Cardiff University

School of Computer Science & Informatics

Automatic Semantic and Geometric Enrichment of CityGML 3D Building Models of varying Architectural Styles with HOG-based Template Matching

Jonathan Slade

Submitted in part fulfilment of the requirements for the degree of

Doctor of Philosophy in Computer Science at Cardiff University, January 31st, 2018

DECLARATION

This work has not been submitted in substance for any other degree or award at this or any other university or place of learning, nor is being submitted concurrently in candidature for any degree or other award.

Signed ………………………………………… (candidate) Date ………………….

STATEMENT 1

This thesis is being submitted in partial fulfillment of the requirements for the degree of PhD……(insert MCh, MD, MPhil, PhD etc, as appropriate)

Signed ………………………………………… (candidate) Date ………………….

STATEMENT 2

This thesis is the result of my own independent work/investigation, except where otherwise stated, and the thesis has not been edited by a third party beyond what is permitted by Cardiff University's Policy on the Use of Third Party Editors by Research Degree Students. Other sources are acknowledged by explicit references. The views expressed are my own.

Signed ………………………………………… (candidate) Date ………………….

STATEMENT 3

I hereby give consent for my thesis, if accepted, to be available online in the University's Open Access repository and for inter-library loan, and for the title and summary to be made available to outside organisations.

Signed ………………………………………… (candidate) Date ………………….

Abstract

While the number of 3D geo-spatial digital models of buildings with cultural heritage interest is burgeoning, most lack semantic annotation that could be used to inform users of mobile and desktop applications about the architectural features and origins of the buildings. Additionally, while automated reconstruction of 3D building models is an active research area, the labelling of architectural features (objects) is comparatively less well researched, while distinguishing between different architectural styles is less well researched still. Meanwhile, the successful automatic identification of architectural objects, typified by a comparatively less symmetrical or less regular distribution of objects on façades, particularly on older buildings, has so far eluded researchers.

This research has addressed these issues by automating the semantic and geometric enrichment of existing 3D building models by using Histogram of Oriented Gradients (HOG)-based template matching. The methods are applied to the texture maps of 3D building models of 20th century styles, of Georgian-Regency (1715-1830) style and of the Norman (1066 to late 12th century) style, where the amalgam of styles present on buildings of the latter style necessitates detection of styles of the Gothic tradition (late 12th century to present day).

The most successful results were obtained when applying a set of heuristics including the use of real world dimensions, while a Support Vector Machine (SVM)-based machine learning approach was found effective in obviating the need for thresholds on match-scores when making detection decisions.

Dedication

Dedicated to the memory of my sister, Emma, to my partner Siobhan and to my parents Kay and Roger. You never lost faith, despite the ups and downs.

Acknowledgements

Huge thanks must go to my wonderful supervisors Christopher Jones and Paul Rosin. I have valued enormously your wealth of knowledge, experience and patience, plus your ability to drill into the detail while also setting things in a broader context. You kept encouraging me and pushing me, and I can't thank you enough. Big thanks also to Isabel Sargent, my Ordnance Survey, Great Britain supervisor. You've been a great support, and I've really enjoyed the time at headquarters in Southampton.

This work was jointly funded by the Engineering and Physical Sciences Research Council (EPSRC) and the Ordnance Survey, Great Britain, under an EPSRC *Industrial Cooperative Awards in Science and Technology* (iCASE) award (1407944). Special thanks go to the following individuals for their assistance: David Holland, Jon Horgan and Debbie Wilson at Ordnance Survey; Yinda Zhang at Princeton, first author on the HOG-based template-matching paper used in this work (Zhang *et al.*, 2013b); Olivier Teboul, creator of the *ECP Facades Database* (Teboul, 2010); Stefan Buss at 3DIS, the producer of the Trimble *SketchUp CityEditor* plug-in; Catherine Bankhurst and Charles O'Brien, part of the team at Yale University Press, the producer of the Pevsner architectural guides; Neville Mcauley-Churchill at extraordinarybookofdoors.com; Neil Guiden at Historic England; Jackie Chadwick at the National Museum Wales; Susan Fielding at the Royal Commission on Ancient and Historical Monuments of Wales; Kathryn Betts at Meldreth Local History Group; and Yukun Lai and Oriel Prizeman at Cardiff University.

Finally, thank you to my PhD and research colleagues over the years, for your support, especially Kelvin Wong at University College London.

Quote

“Things are not always what they seem; the first appearance deceives many; the intelligence of a few perceives what has been carefully hidden.”

Plato in *Phraedus* c.370 BC

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation

3D building models are of increasing importance in a number of fields. These include: cultural heritage augmented reality tours; facilities and asset management; smart cities; 3D real-estate registration; modelling the energy requirements for a building; solar planning; noise modelling; and emergency response planning (e.g. see: Bernardini *et al.*, 2016; Boyes *et al.*, 2017; Bullivant, 2017; Drobež *et al.*, 2017; Duguleana *et al.*, 2016; Kumar *et al.*, 2017; Lilis *et al.*, 2016; Murtiyoso *et al.*, 2017; Romero Rodríguez *et al.*, 2017). With respect to cultural heritage it has, for example, been estimated that Britain's built heritage attracts 9 million overseas visits and £6.5 billion of foreign spending per year (Dawe, 2013).

While the advent of *Web 2.0* has led to an increasing number of publicly available 3D building models, the vast majority of these models lack much, if any, semantic content which might be used to inform users about the building, or to enable rich geo-data applications (Jones *et al.*, 2014). In many cases detailed geometry, to which labelling could be attached, is also lacking. Additionally, many of the available 3D representations of buildings are point or mesh-based. While such representations do have their uses, the attaching of semantic content to points or meshes presents, arguably, potential spatial accuracy issues. In contrast, spatial analysis of a 3D building model formed from structured geometry is both more straightforward and potentially more accurate (see: Holland *et al.*, 2016; Xie *et al.*, 2017).

So, what type of semantic enrichment might be useful? Knowledge regarding the locations of windows and doors on a building enables a number of important use cases, including (i) more refined energy modelling, (ii) better noise modelling, (iii) so-called ‘right to light’ planning, (iv) emergency response planning, (v) automated building control in a smart city, (vi) entrance-locating for delivery companies, and, by inferring the storeys in a building, (vii) 3D real-estate registration. (i)-(iii) are increasingly required during local authority planning applications for buildings.

Consequently, a large body of literature exists regarding automated window and door detection using façade images, e.g. see: Cohen *et al.* (2014); Liu *et al.* (2017); Morago *et al.* (2016); Müller *et al.* (2007); Riemenschneider *et al.* (2012); Wolff *et al.* (2016); Zhang *et al.* (2013a); Zhang *et al.* (2015). However, such methods are restricted to architectural styles typified by the symmetrical arrangement and regular spacing of architectural objects on façades – for older buildings, which are more likely to be an amalgam of styles, such arrangement and spacing is less likely to be the case.

Those methods which do not rely on such symmetry or distribution do not tend to label the architectural style of the detected windows or doors, e.g. see: Horne *et al.* (2016); Kim *et al.* (2016a); Martinović *et al.* (2015); Teboul *et al.* (2013); Wenzel and Förstner (2016); Yu and Wang (2016). While some methods do detect architectural style, they tend to do so for the entire building and not for the individual windows and doors, e.g. see: Chu and Tsai (2012); Ippolito and Attenni (2015); Obeso *et al.* (2016a); Strobbe *et al.* (2016); Zhang *et al.* (2010).

All of the abovementioned methods detect windows or doors on façade images. Those few approaches which do so directly on existing 3D building models, such as Henn *et al.* (2012), only do so for architectural styles typified by symmetry and regular distribution of objects. Equally, methods which do detect the architectural style of individual window or door objects do so on façade images, and for the same types of styles.

Knowledge of the architectural style of window and door objects enables some exciting use cases, including the aforementioned augmented reality cultural heritage tours. Imagine, for example, a fun educational app which uses a 3D building model and poses questions regarding the style of a window or door at which the phone’s camera is directed. Additionally, such information could be used to reduce the time-consuming process of cataloguing the style of objects on a cultural heritage property, something which is of importance for local authorities and building preservation organisations.

Using the architectural styles of the windows and doors, a potentially dominant style for a building, or its component parts, could be inferred. Style can be used as surrogate for

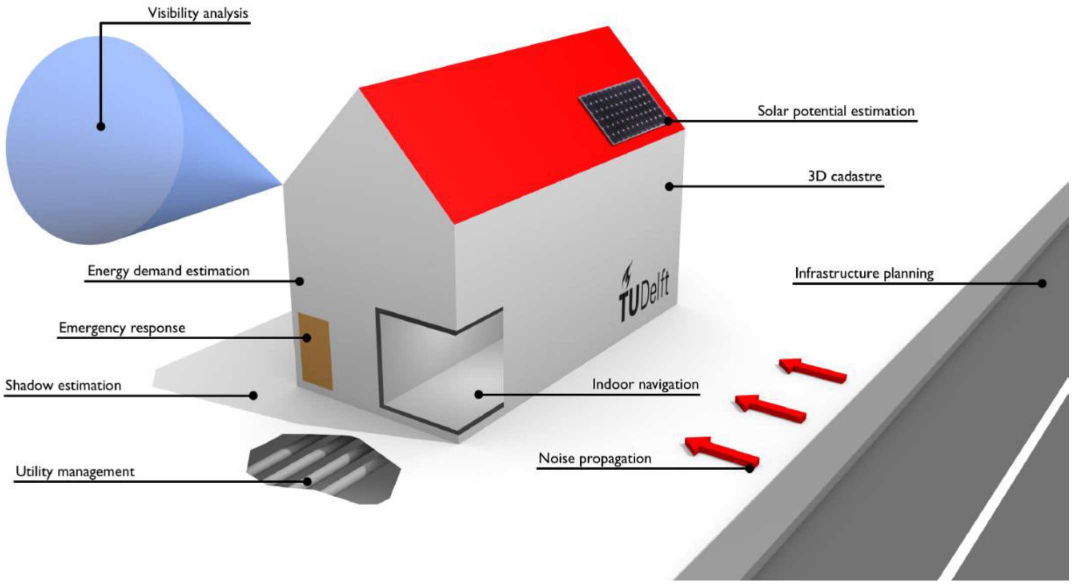


Figure 1.1: Selection of Use Cases for 3D City Models
Source: Biljecki *et al.* (2015). Additionally, cultural heritage has a number of use cases – see Section 1.1.

building age (though this is not without its challenges – see Chapter 2). As a result of this, a number of other use cases become possible. These include: more refined energy and noise modelling; insurance risk assessment; property valuation modelling; and sewerage and water pipe infrastructure replacement planning.

1.2 Use Cases for 3D Building Models

From the time of the Greeks through the Renaissance to the present day, ‘tangible’ 3D building models (such as those in clay and wood) have been an important part of the practice of architecture and building management, helping to provide an understanding of a building and its space (Mezzino *et al.*, 2016). Ammon (2017) discuss the so-called ‘digital turn’ in the Architecture, Engineering and Construction (AEC) industry, that is away from paper methods and toward 3D digital Computer-aided Design (CAD) approaches, which took place from the early 1990s. Urban planning makes extensive use of 3D building models, including as an important aid during public engagement e.g. see: Berck (2017); Brasebin *et al.* (2016); Seifert *et al.* (2016). On this last point, Münster *et al.* (2017) stress how useful the modern digital equivalent of Mezzino’s ‘tangible’ 3D building models can be for imparting a building’s design and architectural character to a user. illustrates some example use cases, key examples of which are described below.

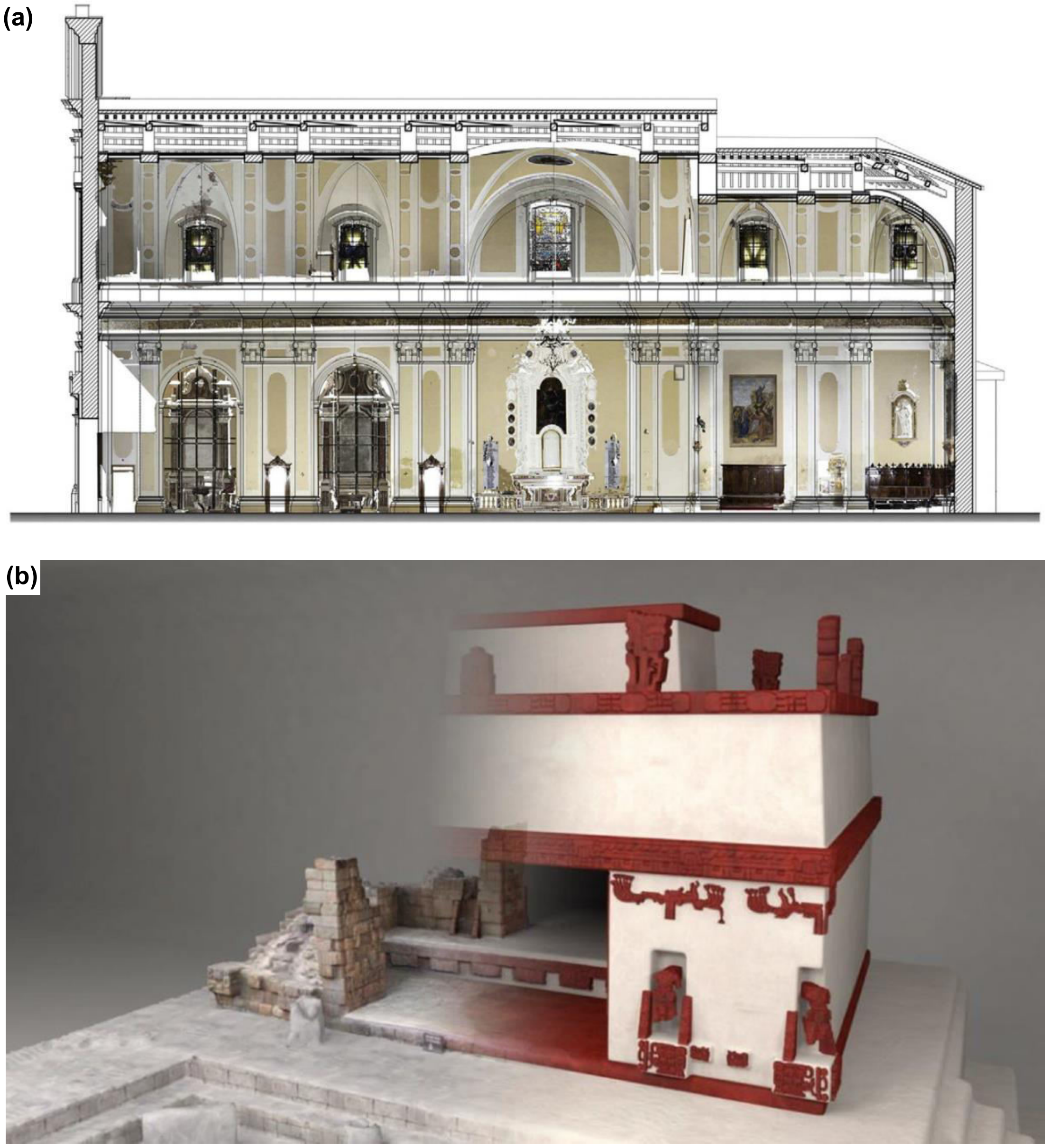


Figure 1.2: Examples of 3D Building Models used for Cultural Heritage
(*a*) Plan view from 3D building model of Santissimo Nome di Maria church, Poggio Rusco, Mantua, Italy, built 1748, created to assist with restoration following earthquake damage in 2012 (Biagini *et al.*, 2016). (*b*) 3D building model, in semi-transparency, of a no longer extant temple in the ancient Mayan city of Copán, Honduras, a UNESCO site, inhabited between the 5th to 9th centuries, overlaid on a laser scan (von Schwerin *et al.*, 2016).

1.2.1 3D Building Models in Cultural Heritage, BIM & HBIM

Of particular relevance to this work is the use of 3D building models for cultural heritage preservation. Figure 2.3 illustrates some examples of the use of 3D building models for cultural heritage. Indeed, the creation and study of 3D building models for cultural heritage purposes is a highly researched field (Münster *et al.*, 2016). Such preservation and associated cataloguing, often mandated nationally e.g. in Great Britain by Historic

England (2017), Historic Environment Scotland (2017) and Cadw (2017), and supported internationally by the likes of UNESCO, has clear benefits. For example, with the maintenance of a catalogue of building components, restoration following malicious damage or theft becomes more straightforward (Blake, 2008). Preservation of cultural heritage buildings can also encourage tourism (McKercher and Du Cros, 2002). Note once again that it been estimated Britain's built heritage attracts 9 million overseas visits and £6.5 billion of foreign spending per year (Dawe, 2013).

Studies using 3D building models for preservation purposes include: Biagini *et al.* (2016); Brusaporci (2017); Dhonju *et al.* (2017); Murtiyoso *et al.* (2017); Oreni *et al.* (2014). Relatedly, 3D building models have been used to reconstruct buildings which are no longer fully extant i.e. for archaeological purposes (e.g. see: Apollonio, 2016; Deggim *et al.*, 2017; Verhoeven, 2016; von Schwerin *et al.*, 2016).

Building Information Modelling (BIM) is an increasingly common, and increasingly mandated, AEC approach for digitally recording the design and development of buildings during the project lifecycle, in a collaborative manner, often using CAD technologies (RIBA, 2016). BIM 3D building models are also handed over for facilities and asset management e.g. see: Boyes *et al.* (2017); Jiang *et al.* (2017); Maltese *et al.* (2016). Heritage BIM (HBIM) is a similar approach to BIM but for heritage buildings, whereby polyhedral models are mapped to point cloud data from surveys (Murphy *et al.*, 2009b).

If the architectural style of the individual components in a building is known, then the dominant architectural style for the building could be inferred. Understanding the architectural style (and hence the approximate age) of a building and its individual objects can help construct a cultural heritage record for a building, for which a 3D building model can form the basis. Such a model could provide a baseline for change detection and be used to create a virtual tour for users (e.g. see: Kersten *et al.*, 2017; Kiourt *et al.*, 2016; Napolitano *et al.*, in press). Augmented Reality (AR) tours, such as those on smartphones, also make use of 3D building models e.g. see: Bostanci *et al.* (2015); Duguleana *et al.* (2016); Li and Fan (2014); Verykokou *et al.* (2014). Virtual Reality (VR) is a further application for 3D building models e.g. see: Koeva *et al.* (2017); Lin (2017); Murphy *et al.* (2017); Thomopoulos *et al.* (2016).

Using the architectural style of the individual components, a 'transformation sequence' could also be determined i.e. the architectural evolution of the building through time (Agudo *et al.*, 2016). Such a sequence could be of great utility in an educational cultural-heritage virtual or augmented reality app. See also Section 2.5 for further discussion of some key stages in the evolution of virtual cultural heritage tours, achieved as a result of 3D reconstruction.

1.2.2 3D Building Models for 3D Cadastre

3D building models are also used for 3D cadastre (real-estate registry) by local and national cadastral survey organisations and again a significant body of work exists on the topic e.g. see: Atazadeh *et al.* (2016b); Drobež *et al.* (2017); Dsilva (2009); Hamid *et al.* (2016); Ho and Rajabifard (2016); Shojaei *et al.* (2016); Stoter *et al.* (2016a). illustrates an example use of a 3D building model for 3D cadastre. Clearly, with increasingly high-rise societies, legal ownership boundaries are no longer solely 2D. As such, 3D building models can provide real benefit to the survey organisations, when compared to 2D, including the automation of spatial queries such as how many units exist within a certain geographic area within a complex block.

Sargent *et al.* (2015) also highlight how having a 3D building model containing doors and floors, the latter perhaps derived from the positions of windows, can be useful to other organisations such as delivery firms and insurers. Indeed, Wong and Ellul (2017) state that as many as half of all potential commercial users of a geospatial 3D building model data product would require the number of storeys present to be included in the data. The use of 3D building models for historic 3D cadastre, such as Gatta *et al.* (2017), represents a cross-over with the cultural heritage use cases mentioned in Section 1.1, providing insights into the historic urban and societal fabric, for example.

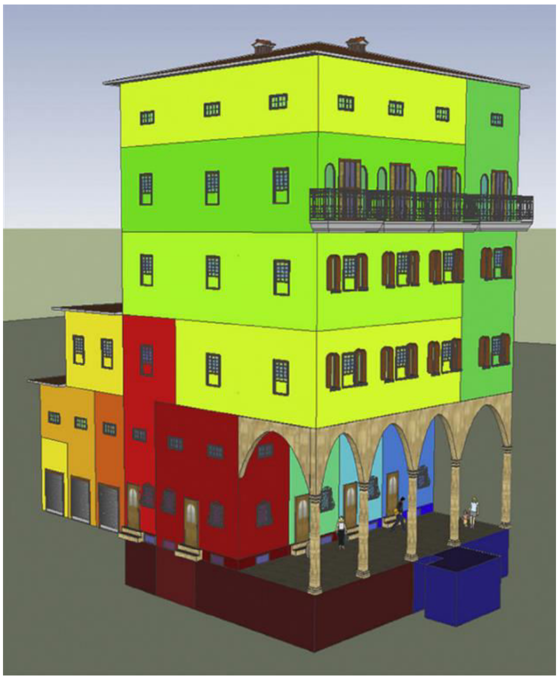


Figure 1.3: Example of 3D Building Model used for 3D Cadastre
Colours represent different historic legal owners of the units within a building on Piazza Maggiore, Bologna, Italy c.1831 (Gatta *et al.*, 2017).

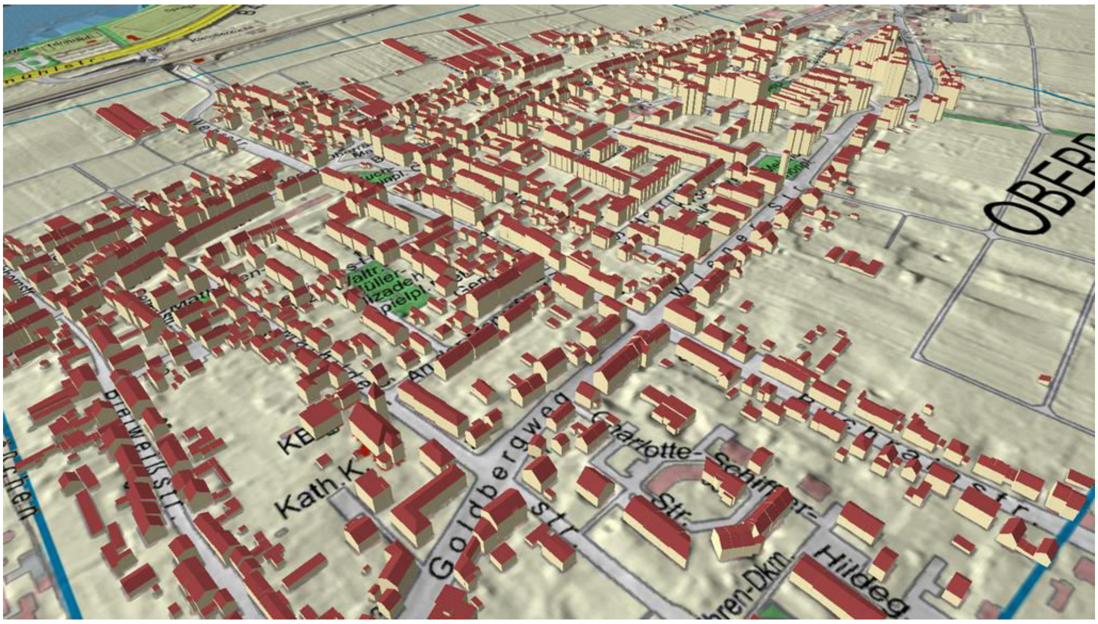


Figure 1.4: Visualisation of a Regional Mapping Agency 3D Building Data Product Produced for the City Surveying Office of Frankfurt, Germany; the 3D building models have been placed onto a 2D topographic map which has been draped over a 3D terrain model (virtualcitySYSTEMS, 2017).

1.2.3 National Mapping Agencies & 3D Building Models

NMAs such as the Ordnance Survey in Great Britain are endeavouring to create 3D city models for cities, regions and, in some cases, whole countries. provides a visualisation of a regional mapping agency 3D building data product. In Europe, for example, some NMAs have 'already built solid databases to maintain 2.5D and 3D topographic data covering their whole country' such as for the Netherlands (Stoter *et al.*, 2016b). In some countries the NMA is combined with cadastral survey and land registry, which means that 3D building models provide the opportunity for 3D cadastre. For a selection of NMA 3D city / 3D building model or cadastre initiatives see: El-Mekawy *et al.* (2014); Felus *et al.* (2014); Guan *et al.* (2016); Gulliver *et al.* (2016); Holland *et al.* (2016); Kaden and Kolbe (2013); Streilein *et al.* (2016); Van Oosterom *et al.* (2012).

1.2.4 Other Use Cases for 3D Building Models including Building Age

The size and orientation of building surfaces, the positions and sizes of windows and doors, and volume estimations for interiors allow for a further suite of use cases. These include Building Energy Modelling (BEM) e.g. see: Del Giudice *et al.* (2014); Ham and Golparvar-Fard (2015); Kim *et al.* (2016b); Lilis *et al.* (2016). For more information on

BEM see later in this section. Another is solar planning such as the optimal positioning of photovoltaic cells, right-to-light considerations and solar shading and irradiance analysis e.g. see: Chatzipoulka *et al.* (2016); Liang and Gong (2017); Martínez-Rubio *et al.* (2016); Romero Rodríguez *et al.* (2017). Emergency response planning can also be undertaken using 3D building models. This can include the training of personnel, the planning of evacuation routes within and around buildings and the potential to automatically pilot Unmanned Aerial Vehicles (UAVs) to survey a dangerous location quickly; knowing the locations of windows and doors may allow the survey to assess interiors and the number of people escaping or even allow the UAV to be piloted through openings (e.g. see: Bernardini *et al.*, 2016; Boguslawski *et al.*, 2016; Diez *et al.*, 2016; Tashakkori *et al.*, 2016). Using the locations and sizes of apertures on a 3D building model, noise modelling can be carried out, and is often required for planning applications, such as in

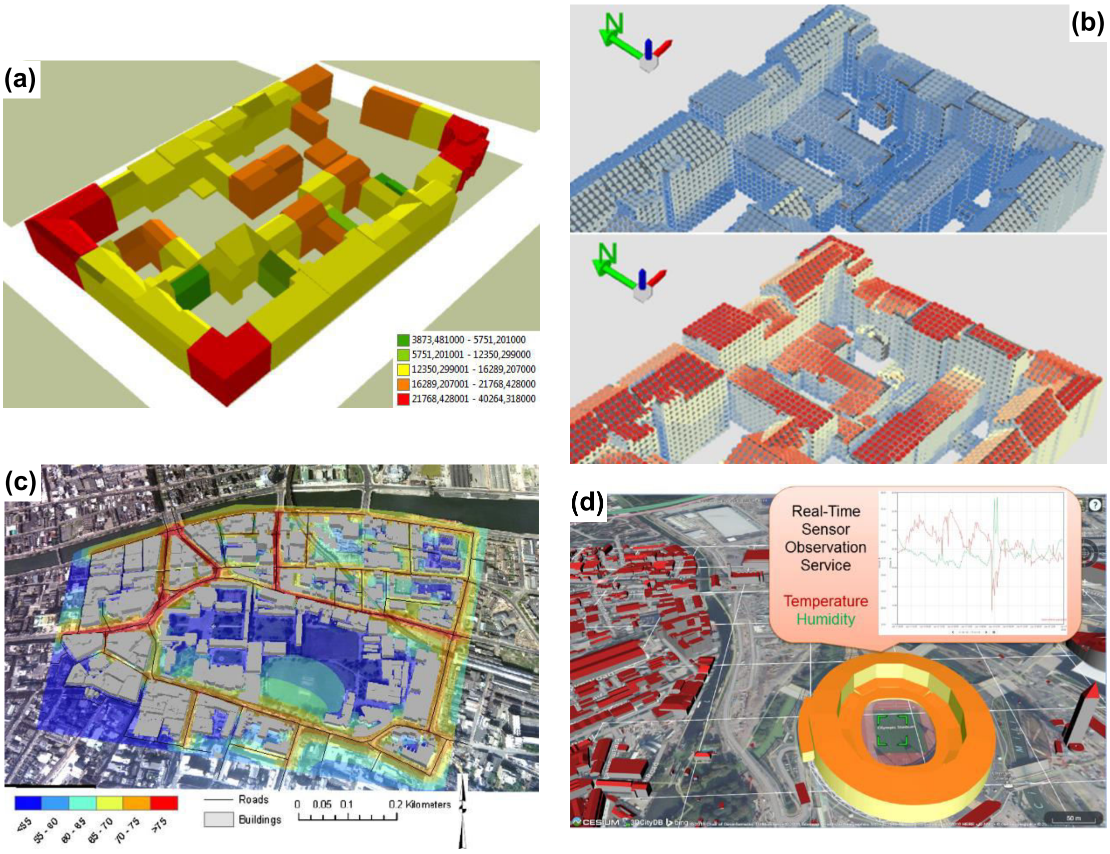


Figure 1.5: Examples of Other 3D Building Model Use Cases
(*a*) Building Energy Modelling(BEM) / utility company use cases i.e. annual warm water demand in kWh/a for a block of buildings in Berlin, Germany (Kaden and Kolbe, 2013). (*b*) Solar irradiation use case showing January (*top*) and July (*bottom*) with high values in red, low in blue (Chaturvedi and Kolbe, 2016). (*c*) Noise modelling use case for part of Dublin, Ireland (Murphy *et al.*, 2009a). (*d*) Smart cities / Internet of Things (IoT) use case showing sensor observations for temperature and humidity for a room in a building (Chaturvedi and Kolbe, 2016).

the UK. Noise modelling can attempt to forecast the noise either within a building, resulting from a proposed new transport network for example, or outside a building, such as around a proposed new factory (e.g. see: Kumar *et al.*, 2017; Murphy *et al.*, 2009a; Zubala and Sadurska, 2016). illustrates some of these, plus the following, use cases.

Smart cities, incorporating a feed from Internet of Things (IoT) sensor data, can also make use of 3D building models to provide outcomes such as real-time utility, transport and health management (e.g. see: Brundu *et al.*, 2017; Bullivant, 2017; Howell *et al.*, 2016; Kamel Boulos *et al.*, 2017). 3D building models can be used for personal navigation, especially indoor navigation e.g. see: Ballardini *et al.* (2016); Liao and Dong (2017); Sattler *et al.* (2017); Xiong *et al.* (2016). Further uses of 3D building models include Global Navigation Satellite System (GNSS) / radio network planning (e.g. Ellul *et al.*, 2016), microclimate modelling (e.g. see: Garcia-Dorado *et al.*, 2017; Kardinal Jusuf *et al.*, 2017), flood planning (e.g. Amirebrahimi *et al.*, 2016) and population estimation (e.g. Biljecki *et al.*, 2016a). Biljecki *et al.* (2015) provide a useful further discussion of 3D building model use cases.

The rationale for the *UKBuildings* data product from GeoInformation Group (2016) states that there are a number of commercial 3D building model use cases for building age, as follows:

(1) The insurance industry can use building age when assessing building risk.

(2) BEM calculates the thermal capacitance of a building, as derived from the construction materials used, their positioning and their volume in the building (Menberg *et al.*, 2016). Different construction materials possess different thermal properties and therefore different insulating capabilities: for example, modern insulating foams used in rigid thermal insulation can be up to 50 times more effective at insulating than brick alone (e.g. see: Autodesk, 2018; Menberg *et al.*, 2016). Knowledge of the age of a building could be used as a proxy for the likely materials used in its construction, where an estimation of the building materials can be used as an input to BEM.

(3) With respect to noise modelling, different construction materials possess different sound insulating capability too (Hopkins, 2012, pp. 608-609). Once again, modern foam insulation can be more effective than brick alone.

(4) Building age can be used in property valuation modelling.

(5) Lastly, water and sewerage utility infrastructure providers might use a knowledge of building age to infer the likely state of the pipe network in the vicinity.

As mentioned in Section 1.1 architectural style can be used as an indicator of approximate building age, or as a surrogate for age, though using such an approach is not without its challenges – see Section 2.4.

1.3 Objectives & Research Question

To ensure that the output from this thesis is useful to the Ordnance Survey, the approach would need to be rolled-out across Great Britain. To reiterate, the Ordnance Survey are one of the sponsors of this work. As such, the methods in this thesis would be a useful addition to the other techniques that Ordnance Survey have employed successfully for the automated detection of roof-type in imagery (Orlowski, 2017). The approach described in this thesis supports the creation of 3D building models with Level of Detail 3 (LOD3) – see Chapter 2. As such, it underpins not just visualisation, but also analysis of the semantic content and geometric content for component objects within 3D building models. Expanding upon those use cases described above, such content can drive cultural heritage applications such as the aforementioned educational augmented reality apps, in addition to the supporting the work of architectural historians and building preservation authorities, as follows.

Such an augmented reality app might take the form of a quiz and may ask users to 'guess the architectural style' of windows and doors.

The work of architectural historians might include deriving how a building has evolved over time and calculating the proportions of the building which comprise different architectural styles, based on the architectural style of its architectural components.

As part of more detailed geometry in a 3D building model of a cultural heritage building, and associated architectural labelling of that geometry, knowledge of the architectural style of building components could assist building preservation authorities repair damage. Such damage might include that resulting from vandalism or following an earthquake. In such a scenario, 3D building models would allow geospatial analysis of the building, where examples of such analysis could be as follows. Firstly, by using imagery of damaged areas as texture maps on a pre-existing 3D building model, architectural objects within the model could be assessed for damage – crucially, the use of such an approach, versus the use of a purely tabular record, has the potentially to significantly reduce the time taken to assess the damage, including automatically determining the precise location of the damage in the building. Subsequently, geospatial analysis could be used to plan, and consequently to expedite, the work of specialist renovation teams, noting that older buildings are often an amalgam of styles and that different restoration specialists may have differing specialities with respect to both the type of object and architectural style. For example, some renovation teams may specialise in Norman window restoration and not in other types of architectural feature or styles.

It is for the reasons above that the approach taken in this work therefore explored methods for algorithm efficiency and scalability to unseen data.

The objectives of this work were therefore as follows:

(1) Geometrically and semantically enrich existing 3D building models with windows and doors.

(2) Detect the architectural style of the windows and doors, including for older buildings i.e. those buildings more likely to be an amalgam of styles.

(3) Detect the windows and doors irrespective of the presence of symmetry or regularly spaced objects on the building façades.

(4) Make the method scalable to unseen 3D building models.

The research question was as follows:

> *Is it possible to automatically enrich 3D building models of cultural heritage buildings by extracting window and door information – both geometry and specific architectural style?*

1.4 Approach Summary

Based on the above objectives, methods were created in this research to automatically detect window and door geometry for existing 3D building models, including labelling the objects as windows and doors and specifying their architectural style. The 3D building models were obtained from the internet. To ensure that the methods performed in the absence of symmetry and regularity of the distribution of architectural components on the building façades, the data used covered a range of architectural styles from the 12th century to the present day. This contrasts with the vast majority of the existing methods, which focus on buildings from the 17th century onwards. By testing the methods on older buildings, multiple styles were detected on the same building, providing insights into the development of the building over time.

In order to achieve the objectives, techniques from the fields of computer vision and machine learning were used, utilising the texture maps on the surfaces of the 3D building models. In particular, a recent variant of template matching was used i.e. HOG-based template matching (see: Xiao, 2013; Zhang *et al.*, 2013b). Additionally, part of this work used a support vector machine (SVM) to learn what represented a window or door.

In order to remove false matches, and thus enhance performance, a number of heuristics were used, including those which made use of real-world measurements taken from the 3D building model.

Lastly, three trials were conducted as part of the work, including: the initial use of a standard template matching approach; an image matching trial (to enable future work); and the clustering of templates.

1.5 Challenges

The work presented a number of challenges, as summarised below.

In the absence of existing suitable reference datasets, a wide-ranging data collection exercise was undertaken. In particular, obtaining sufficient image data of the correct architectural style and image quality proved significantly time-consuming.

On the last point, image quality was a challenge across the project, requiring a small proportion of the texture map images on the 3D building models to be replaced. As such, one is at the mercy of the person who originally created the 3D model.

The complex structure of many of the 3D building models used also presented a number of challenges. Specifically, the models can use: complex component nesting; multiple syntactic approaches for the same geometry; a range of texture mapping methods; and a substantial number of components (see: Gröger and Plümer, 2012; Nguyen *et al.*, 2017). In order to overcome these challenges quality checking and optimisation of the 3D building models was carried out in advance and accuracy-validation checks were conducted during processing.

1.6 Main Contributions

Contribution (1) is the creation of a pipeline which can detect, using multiple architectural designs, the styles of windows and doors on existing 3D building models. This was achieved with 4 different architectural styles, on 24 3D building models comprising a total of 17,433 surfaces and 956 window and door objects. While the individual steps in the pipeline are not novel, the combining of these steps and the application of the methods to the detection of the architectural style of window or door objects on existing 3D building models is novel.

Contribution (2) is the semantic and geometric enrichment of existing 3D building models. In particular, the texture mapping inherent to the 3D building model is used to create the geometric enrichment, by determining the geometry of window and door objects that had not been present in the original model geometry. The semantic enrichment comprises the type of object ('window' or 'door') and its architectural style. Moreover, the work indicates that with images of sufficient quality, and suitable templates, an even greater level of architectural detail can potentially be detected.

Contribution (3) is the lack of a reliance on symmetry or regularity of object distribution on façades, plus the use of a range of styles which include those pre-17th century. Methods which focus on older buildings are rare in the literature (see above). Importantly, this contribution means that the methods in this thesis are more suitable for older buildings.

Contribution (4) is the use of heuristics, including those which make use of real-world measurements within the 3D building model, to reduce the number of false matches. Following an extensive evaluation of the heuristics, it is clear that their use consistently improves the results (by a mean of 73%), while for some 3D building models this improvement can be as much as 30-fold.

Contribution (5) is the use of machine learning to teach the pipeline to recognise windows and doors of particular architectural styles, using the match-scores from HOG-based template matching. In particular, parity of performance was obtained between (a) the part of the method which used manually determined match-score thresholds, and (b) that which effectively did so using machine learning. Crucially, the parity achieved by (b) enables the scalability of the solution to both unseen 3D building models and, provided suitable data is available, to other architectural styles. While the runtime achieved with (b) was higher than (a), steps for future work have been identified which should bring the two runtimes more in line.

1.7 List of Relevant Publications

Published

- Jones, C.B., Rosin, P.L., Slade, J., 2014. Semantic and geometric enrichment of 3D geo-spatial models with captioned photos and labelled illustrations, in: Belz, A., Moens, M.-F., Smeaton, A.F. (Eds.), Proc.5th International Conference on Computational Linguistics (COLING) – 3rd Workshop on Vision and Language (VL). ACL, Dublin, Ireland, pp. 62-67.
- Slade, J.D., Jones, C.B., Rosin, P.L., 2014. Semantic and geometric enrichment of 3D geo-spatial building models with captioned photos and labelled illustrations, in: Stoter, J. (Ed.), Proc. EuroSDR / ISPRS Workshop – Efficient Capturing of 3D Objects at a National Level: with a Focus on Buildings and Infrastructure. EuroSDR / ISPRS, Southampton, UK.
- Slade, J., Jones, C.B., Rosin, P.L., 2015. Semantic and geometric enrichment of 3D geo-spatial building models with photo captions and illustration labels using template matching & SIFT, in: Malleson, N. (Ed.), Proc.3rd GIS Research UK (GISRUK). University of Leeds, Leeds, UK.
- Slade, J., Jones, C.B., Rosin, P.L., 2017. Automatic Semantic and Geometric Enrichment of CityGML Building Models using HOG-based Template Matching, in: Abdul-Rahman, A. (Ed.), Lecture Notes in Geoinformation and Cartography (LNGC) – Advances in 3D Geoinformation. Springer International Publishing, Cham, pp. 357-372.

In preparation

- Slade, J., Jones, C.B., Rosin, P.L., Sargent, I., 2017. Automatic Semantic and Geometric Enrichment of CityGML Building Models using HOG-based Template Matching, for varying Architectural Styles, Computers, Environment and Urban Systems (journal)

In addition, this work highlighted necessary development in a Trimble *SketchUp* (Trimble, 2018) plug-in used to process 3D building models, resulting in two dedicated updates to the software by the vendor – see Slade and Buss (2017).

1.8 Thesis Outline

Chapter 2 discusses previous work in the fields of window, door and architectural style detection, plus background to key concepts and approaches. Regarding the content of Chapter 2, note also the comments in Section 1.9.1 and Section 1.9.2, below.

Chapter 3 describes the data, including its pre-processing, plus the challenges faced during the data gathering process.

Chapter 4 details the method and outcome of two early trials, namely the image matching and standard template matching trials.

Chapter 5 explains the method used to create the pipeline into which HOG-based template matching, the heuristics and the semantic and geometric enrichment were placed.

Chapter 6 describes an extension to the pipeline described in Chapter 5, namely the method used to replicate match-score thresholding using the SVM machine learning technique.

Chapter 7 provides an extensive presentation and evaluation of the results of the work. This includes an evaluation of the effectiveness of the heuristics, plus a comparison of the results obtained when match-score thresholds were determined without the SVM, versus when the SVM was used. An evaluation of the use of the methods in this work, with a reference dataset of façade images containing windows and doors, is presented, although the evaluation is qualitative on the basis that the reference data comprised stand-alone façade images and not 3D building models.

Chapter 8 concludes this work by summarising the achievements, the potential applications of the work, shortfalls of the methods used here, and future research to be carried out.

Appendix D details the method and outcome of a further trial i.e. that of template clustering.

1.9 Considerations

A number of considerations should be borne in mind when reading the thesis, as described below.

1.9.1 Interdisciplinary Nature

This thesis is multi-disciplinary, i.e. it covers computer science including computer vision and machine learning, geo-information science and architectural history. Consequently, a greater level of background detail is provided than might normally be expected in a thesis, for the following areas: 3D building models; architectural style, computer vision; and machine learning. This is especially true for Chapter 2.

1.9.2 Architectural Style Focus

Part of the funding for this work was provided by Great Britain's national mapping agency, Ordnance Survey. The architectural style focus here is therefore on British architectural styles. That said, due to the historic nature of architectural style development, a number of the style groupings used are also present across Europe and beyond, albeit with local variations.

1.9.3 Development Framework

The majority of the new code created for the methods in Chapters 3-6, used MATLAB (MathWorks, 2017b). This was on the basis that the code which was reused from existing methods also tended to use MATLAB i.e. that used for the image-matching and standard template matching trials (Chapter 4), HOG-based template matching (Chapters 5 and 6) and the template-clustering trial (Appendix D).

1.9.4 Data Referencing

Lastly, a note on navigating the thesis with respect to the data used. Each item of data used here has a reference code e.g. ‘BM_2_1’. Those reference codes are used in the text, figures and tables. The sources of the data items can be found in Appendix A, using the reference code as a look-up. Data items of the following type, have a reference code that begins with the following letters:

• 3D Building Models	‘BM’
• Façade Images (testing)	‘FI_’
• Templates (clustering)	‘CD’ and ‘CW’
• Photos (only used in figures)	‘PH’
• Templates (non-clustering)	‘TD’ and ‘TW’
• Training Images	‘TZ’

Chapter 2

Background

2.1 Introduction

To recap, while the availability of 3D building models is increasing, most lack much, if any, semantic content, which could be used to inform users about the building or to support rich geo-data applications (Jones *et al.*, 2014). The necessary geometry for the attaching of such labels is also often missing.

3D building models have a wide variety of uses, including: for cultural heritage purposes such as within augmented reality apps; during the planning process; for building energy modelling; for noise modelling; for solar planning; and for the cataloguing of complex legal ownership boundaries (Biljecki *et al.*, 2015). A number of the use cases require window and door geometry and semantics, or features which can be derived from the presence of windows and doors, such as the number of floors. Knowledge of the architectural history of the building could also greatly assist in the creation of e.g. augmented reality cultural heritage apps for a building or a neighbourhood. Furthermore, the approximate age of the building can be of significant utility for insurance risk profiling, energy and noise modelling and utility network replacement.

Structured, solid models for a building, as opposed to point clouds or mesh surfaces, mean that the attaching of semantic content is more straightforward and accurate (e.g. see: Holland *et al.*, 2016; Xie *et al.*, 2017). Where is the edge of a window in a point cloud or a triangulated mesh? Structured models are therefore more suitable for many of the use cases mentioned above.

This chapter provides a background to the work undertaken in this thesis. It alludes to the challenges presented by the data requirements in this study, as described in more detail in Chapter 3. This chapter also outlines the rationale for the choice of methods used in Chapters 4-6. Lastly, a number of concepts plus a selection of terms, including those of architectural components, are defined here on the basis that they are used elsewhere in this work, including in the evaluation of the results in Chapter 7. The structure and content of this current chapter are outlined below.

Within this chapter the use cases for 3D building models are explained. The different approaches to the digital 3D representation of buildings, including their semantics, are also described. Noting that the majority of the existing methods for detecting windows and doors tend to focus on the use of symmetry or regularity of the distribution of architectural components on building façades, the work here includes older buildings and styles. Such older buildings are generally typified by more asymmetry and irregularity of component distribution. Consequently, a background to the architectural styles used in this work is provided, including comparisons of the complexity and variability of their architectural components.

The chapter also summarises and compares, qualitatively, existing methods, though not before providing background to various computer vision and machine learning approaches, which are used in both the existing methods and the approach in this work. Finally, background is provided on existing reference datasets and available code.

To reiterate, the nature of the work in this thesis is interdisciplinary i.e. it makes use of techniques and concepts from computer science including computer vision and machine learning, geo-information science and architectural history. For this reason, the background provided in this chapter is beyond what might normally be provided in a thesis.

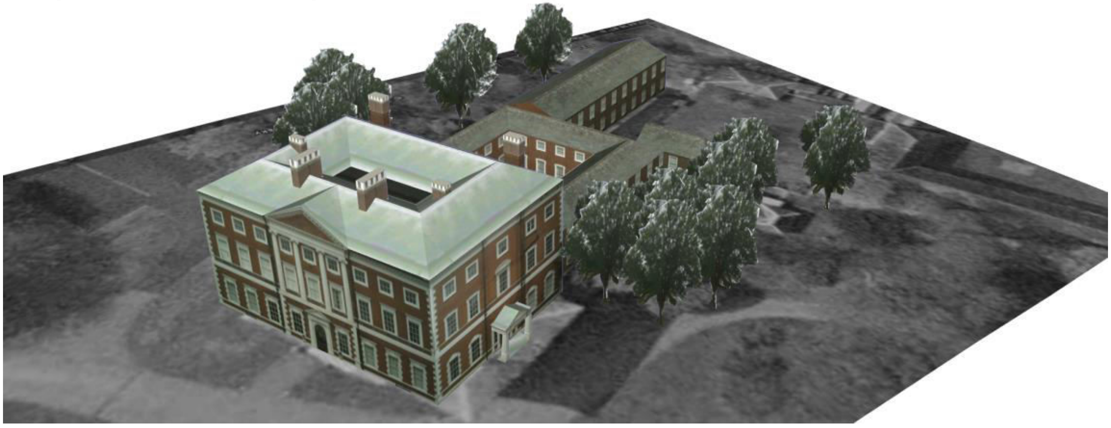
2.2 3D City Models & 3D Building Models Definition

3D building models are a component of 3D city models, alongside terrain, vegetation and transportation models (Döllner *et al.*, 2006). 3D city models comprise structured geometry with polyhedral form and are not point clouds or meshes, although somewhat confusingly, 3D city modelling tools often allow for meshes to be produced as part of a model. Semantic 3D city models additionally include 'ontological structure including thematic classes, attributes, and their interrelationships' for the geometry components (Kolbe *et al.*, 2009). Such semantic content may include 'wall', 'floor' or 'roof' surface labels, in

(PH_1)

Image cannot be displayed due to copyright

(BM_GR_3)
***With* provided texture maps**



(BM_GR_3)
With provided texture maps *removed*

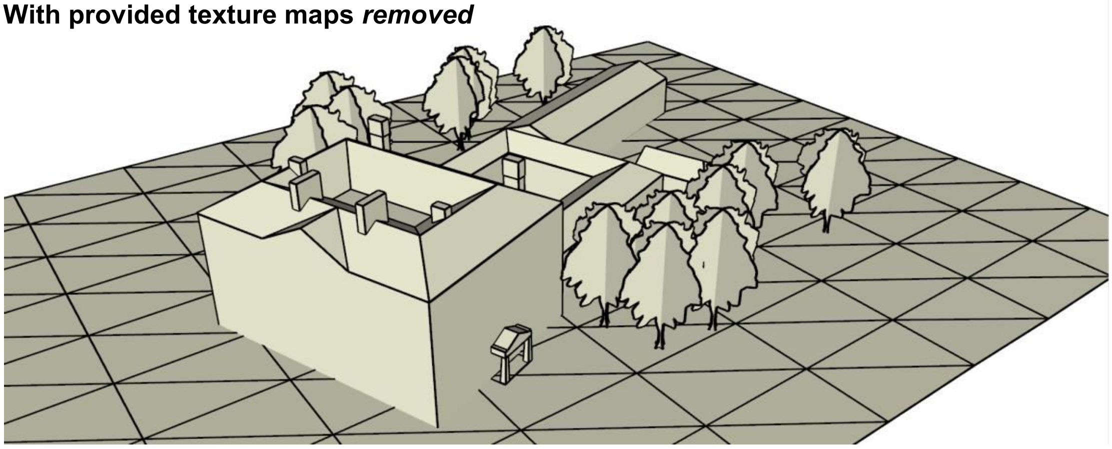


Figure 2.1: Example 3D City Model
(*PH_1*) Photograph of Lytham Hall, Lancashire, UK. (*BM_GR_3*) *'With provided texture maps'*: Corresponding textured 3D city model, including 3D building, terrain and vegetation model. A Trimble *SketchUp* model has been exported to the CityGML format. (*BM_GR_3*) *With provided texture maps removed'*: the same 3D city model but with textures removed. Note the lack of window geometry.

addition to ‘window’ or ‘door’ (Stadler and Kolbe, 2007). While point cloud or mesh representations of buildings can be useful, semantic annotation of geometry is more straightforward and potentially more accurate when a building model is used (e.g. see: Holland *et al.*, 2016; Xie *et al.*, 2017). Imagine, for example, the challenges presented when attempting to define the boundary of a window in a mesh representation of a building. Additionally, with semantics added, the possibilities for spatial analysis such as the locations and total area of windows provide a number of important use cases for 3D building models. Indeed, Sargent *et al.* (2015) found that, for potential users of a 3D building data product from Great Britain’s ‘National Mapping Agency’ (NMA), knowing the positions and shapes of windows and doors were key requirements for a number of use cases. Due to the semantic potential this work focusses on 3D building models and not point cloud or mesh representations. shows an example city model, for a cultural heritage building in the UK – note the lack of window geometry, despite the presence of windows on the textures.

2.3 3D Building Models & Semantic Information

2.3.1 Semantic Information Models

CAD models used by the AEC industry invariably contain semantic content for their geometric components. However, proprietary software such as Autodesk’s *AutoCAD* (Autodesk, 2017) or Bentley System’s *MicroStation* (Bentley Systems, 2017) is often needed to view, create and edit such models. Moreover, that semantic content may, arguably, be focussed on the needs of the AEC industry rather than those of other domains. CAD models from the AEC industry are also more likely to be the intellectual property of the initiating organisation rather than freely available (McGlinn *et al.*, in press).

An alternative approach is provided by the extensible markup language (XML)-based CityGML file format (Gröger *et al.*, 2012) from the Open Geospatial Consortium (OGC), based as it is on OGC’s ‘Geography Markup Language’ (GML). CityGML is a geo-spatial, geometric and common semantic information model used to represent 3D urban objects (Gröger *et al.*, 2012). It is an ontology for the ‘urban landscape’, based on International Organization for Standardization (ISO) 191xx (Gröger *et al.*, 2012). CityGML uses a Levels of Detail (LOD) concept as follows:

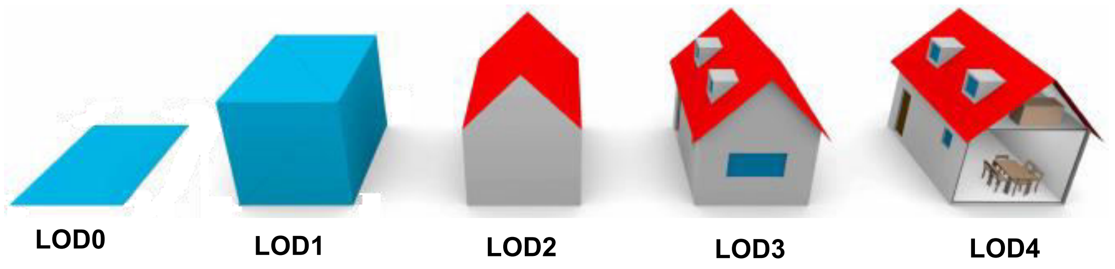


Figure 2.2: CityGML Levels of Detail (LOD)
Source: Robles-Ortega *et al.* (2017)

- LOD0 – regional, landscape
- LOD1 – city, region
- LOD2 – city districts, projects
- LOD3 – architectural models (outside), landmarks
- LOD4 – architectural models (interior)

(Gröger *et al.*, 2012)

See for an illustration of LOD. Note that LOD3 is the minimum LOD to allow for the inclusion of windows and doors (Gröger *et al.*, 2012) – consequently, the building models used in this work are LOD3.

In contrast to CityGML, many of the formats from graphic design, such as KML, X3D, U3D, VRML or legacy CAD geometry formats, do not provide the ability to add semantic attributes (e.g. see: Smart *et al.*, 2011; Stadler and Kolbe, 2007). Ross *et al.* (2009) argue that GML (on which CityGML is based) should become the data interchange standard for urban planning. Döllner and Hagedorn (2007) and Zhu *et al.* (2009) conclude that CityGML is the most effective standard for semantic content.

Isikdag and Zlatanova (2009), Kolbe (2009) and Gröger and Plümer (2012) emphasise the need for CityGML to become interchangeable with the standard BIM data format of Industry Foundation Classes (IFCs). The mapping between the IFC and CityGML ontologies is not without its challenges, including different representations of LOD, but efforts have been undertaken to attempt to automate the conversion e.g. see: Kang and Hong (2017); Safe Software (2015); Vilgertshofer *et al.* (2017). Nonetheless, there really needs to be a standard transformation approach rather than the multiple approaches that exist currently (Arroyo Ohori *et al.*, 2017), although the differences in topology between the two standards make the creation of any transformation a challenging task.

CityGML is not without its drawbacks either, including a lack of explicitness with regard to the representation of LOD; nonetheless, it remains a good choice for the representation of semantically rich 3D city / building models (e.g. see: Löwner *et al.*, 2016; Wong and Ellul, 2016). Also, efforts are underway to address the drawbacks in the next CityGML release (again, e.g. see: Löwner *et al.*, 2016; Wong and Ellul, 2016). Currently, CityGML is at version 2. Löwner and Gröger (2017) outline the plans for version 3. While it is noted that the BIM IFC formats may be more suitable for some use cases, such as cadastre (e.g. see: Atazadeh *et al.*, 2016a; Stoter *et al.*, 2017), CityGML is, for the reasons mentioned above, used in this work.

While CityGML represents an ontology for the urban landscape it is, arguably, not entirely suited to the detailed representation of a multi-layered taxonomy of building structure in the field of cultural heritage. This is primarily because the CityGML ontology lacks the sort of low-level features which might make up, say, the components of a Classical column – see Section 2.4. The 'International Committee for Documentation – Conceptual Reference Model' (CIDOC-CRM) is an ISO standard which allows for detailed, nested cataloguing of cultural heritage components, including those on buildings (ICOM, 2016). It has, for example, recently been implemented for buildings by Soler *et al.* (2017) and in a CIDOC-CRM compatible ontology by Garozzo *et al.* (2017). Due to the absence of an available CIDOC-CRM to CityGML mapping, and on the basis that this work is only identifying window and door objects (which *are* part of CityGML's ontology), no attempt was made to extend the CityGML ontology as part of this work.

Some examples of the use of CityGML 3D building models are outlined below. Chaturvedi and Kolbe (2016) integrated real-time sensors into a smart city model. Lilis *et al.* (2016) combined CityGML and IFC files for BEM. Dsilva (2009) and Ying *et al.* (2017) investigated the use of CityGML for cadastre. Soon *et al.* (2014) investigated CityGML for land use. Kardinal Jusuf *et al.* (2017) predicted microclimate using CityGML. Xiong *et al.* (2016) used CityGML for indoor navigation.

2.3.2 CityGML Semantic & Geometric Representation

CityGML is composed of a core module and thematic extension modules, where 'Building' is one of the thematic extension modules (Gröger *et al.*, 2012). CityGML uses a number of stereotypes including 'Geometry' and 'Feature', where 'windows' and 'doors' are examples of features, which have associated geometry. The relationships between the CityGML 3D building model components is shown in the Unified Modelling Language (UML) diagram in Figure 2.3.

CityGML uses boundary representation (B-REP) for its geometry i.e. point, line, surface and body geometry components (Nguyen-Gia *et al.*, 2017). Other geometry representations include voxels and constructive solid geometry (CSG). Voxels are simple blocks, often regularly sized and spaced. CSG uses cube, cylinder, cone, prism and sphere components. CSG is favoured by the AEC industry and commonly used in CAD. Note that due to its use of B-REP, as opposed to, say, CSG, CityGML faces are always planar according to the specification (Gröger *et al.*, 2012). Such planarity is crucial to the rendering approach used in this work – see Chapter 5 for more on this.

The following is based on Gröger *et al.* (2012). When opening a CityGML file, a viewer will calculate real-world coordinates for the model's geometric components via a transformation from their model-space 3D Cartesian coordinate system to the spatial reference system attribute *srsName* (Whiteside, 2009). Within CityGML each polygon face is represented by a *gml:LinearRing*. CityGML specifies that texture map image files can appear as covers for each *gml:LinearRing*; each *gml:LinearRing* represents a building

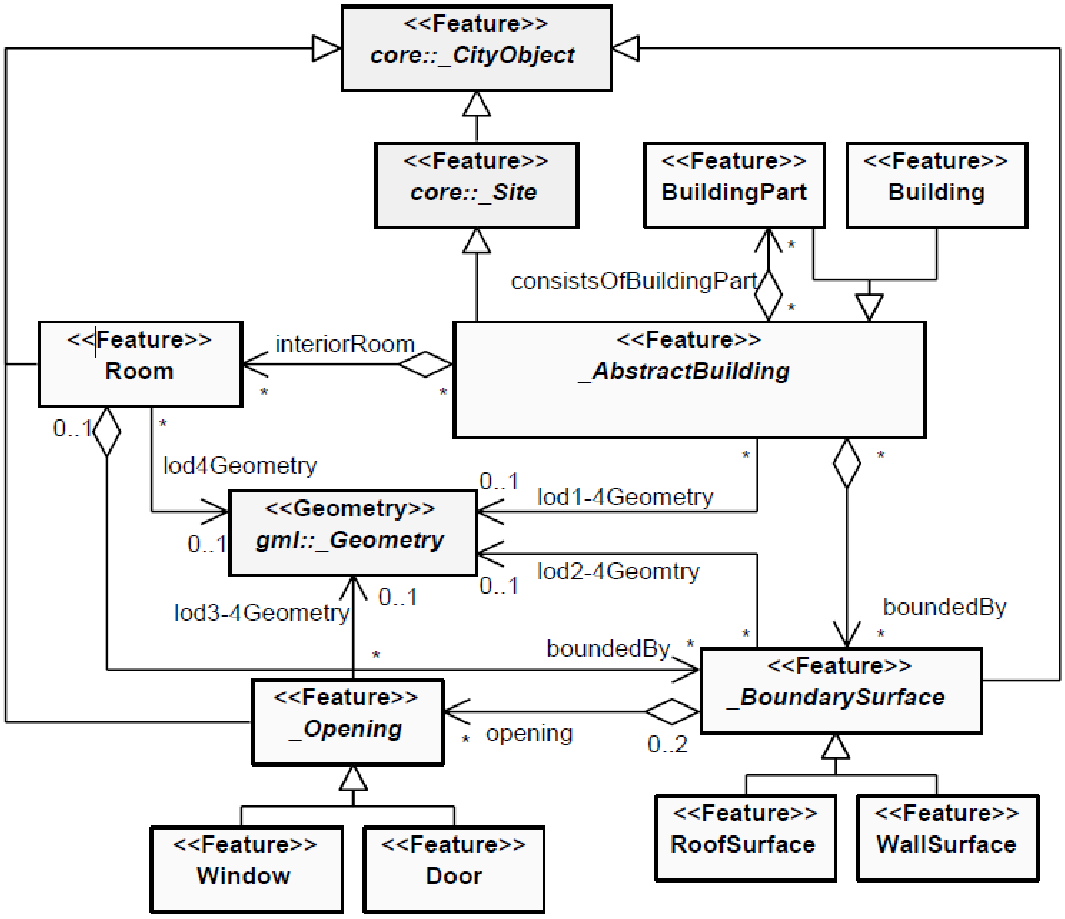


Figure 2.3: UML Diagram for Simplified CityGML 3D Building Model.
Prefixes indicate XML namespaces for associated model elements (Gröger *et al.*, 2012).
Note window and door features are shown *bottom left*. Source: Kolbe (2009)

model polygon. A single texture map image is often re-used across multiple *gml:LinearRing*s. It is conceivable that the creator of the model cropped (or masked) the image file, perhaps only selecting part of that file as the texture map image for the corresponding *gml:LinearRing*. CityGML stores such cropping parameters within 2D 'texture space', using *textureCoordinates* in the [0,1] interval regardless of aspect ratio. (s,t) denotes a *textureCoordinates* coordinate pair, where s is the horizontal and t the vertical texture axis. A polygon with four vertices will have five coordinate pairs, the fifth ensuring that the polygon is closed. As such, CityGML uses the COLLAborative Design Activity (COLLADA) standard for texture mapping (Khronos Group, 2008), which includes the option to 'wrap', 'mirror', 'clamp' or add a 'border' to a texture map image as specified by the value for the element *wrapMode*. The CityGML approach to texture mapping is summarised in .

The texture mapping inherent to a CityGML model can be achieved via translation, rotation, scaling or shear of a texture map image (Gröger *et al.*, 2012). The HOG-based template matching approach was used in this thesis to detect window and door objects

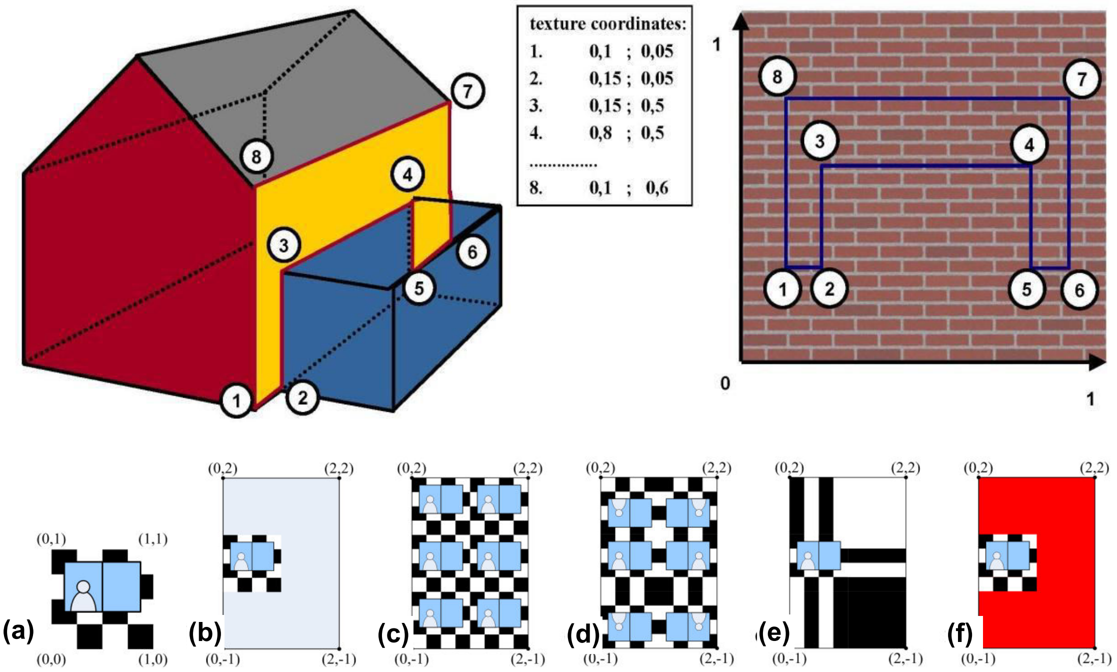


Figure 2.4: CityGML (COLLADA) Approach to Texture Mapping
(*Top*) the texture map image on the right is mapped to the yellow face of a CityGML 3D city model, using the texture coordinates shown. The texture map that appears on the face is a mask, defined by points 1-8, of the original texture map image. (*Bottom*) CityGML texture mapping also allows for different 'wrap' modes for a texture. (*a*) is the original texture map image applied to a façade using different wrap modes: (*b*) none, (*c*) wrap, (*d*) mirror, (*e*) clamp and (*f*) border, with border colour in red. Numbers are texture coordinates.

on the texture map images *prior* to rendering onto the *gml:LinearRing*(s). Rendering of candidate window and door matches onto the *gml:LinearRing*(s) was then carried out, taking account of any transformations expressed in the *textureCoordinates*. Due to the geometry techniques used here it was not possible to accurately carry out shear during rendering. However, in practice few examples of the use of shear were found. Nonetheless, because the standard allows for shear, a shear check was carried out, removing from the final accuracy calculation (F-measure – see Section 2.6) any *gml:LinearRing*s which failed the check. See Chapter 5 for more detail.

The method used in this work used real-world units (and position). As such, the method calculated real-world 3D coordinates for the vertices of the newly detected windows or doors. Heuristics were also incorporated which used real-world units to filter the results. Again, see Chapter 5 for more detail.

Within CityGML a geometric component can often be represented through a number of syntactic approaches (e.g. see: Gröger and Plümer, 2012; Nguyen *et al.*, 2017). Such ambiguity presented a challenge for the methods in this work, not least when considering whether to inject new CityGML for the detected windows and doors into the pre-existing CityGML file. Furthermore, CityGML is a 'complex hierarchical structure containing multi-level deep associations' (Nguyen *et al.*, 2017). Consequently, it was determined that calculating the 3D coordinates of the vertices of the newly identified windows and doors without the injection of new CityGML was more than sufficient. Note also that the work in this thesis was a proof-of-concept. Future work could of course address the latter, having been enabled by this work.

A decision was taken to parse the CityGML file using the Document Object Model (DOM) from W3C (2004) as opposed to another parser such as Simple API for XML (SAX) from Megginson (2004). In practice, despite the assertion from Mezzino *et al.* (2016) that SAX is comparatively faster for parsing CityGML, early experiments in this work determined that DOM was sufficiently quick. DOM also has the advantage of native client support for the particular implementation used here. Nevertheless, if the implementation was to be scaled-up to significantly more 3D building models, and if the injection of new CityGML were to be carried out, then SAX might be a consideration.

2.3.3 CityGML Conversion, Quality, Accuracy & Errors

The CityGML specification proposes a ±0.5m geometric accuracy for LOD3 (Gröger *et al.*, 2012). In this sense the geometric accuracy of a model versus that in the real-world is a representation of geometric quality (Sargent *et al.*, 2015). The accurate representation of geometry within CityGML is not without its own challenges though (e.g. see: Biljecki *et al.*, 2016b; Wong and Ellul, 2016). See below for mention of semantic quality. Note too that the work here could be seen as an enabler for improved quality of 3D building models, i.e. with regard to the completeness of the model, through the inclusion of previously missing windows and doors.

Broadly, quality assessment can be conducted on an intrinsic basis, using just the model itself, or on an extrinsic basis, using some external metric such as a point cloud of the building (Wong and Ellul, 2016). Due to a common lack of additional data (Wong and Ellul, 2016) the methods described in this thesis focussed on intrinsic methods and did not make use of other data to assess accuracy.

Wong and Ellul assessed geometric quality intrinsically using metrics that: provided an indication of the complexity of the building; determined how efficiently the building had been modelled e.g. with the minimum number of vertices; and performed a check on minimum size, including checking vertices were at least 0.5m long. They concluded that quality of the building model (and therefore accuracy) could be affected by '1) choice of modelling tools; 2) model optimization; 3) conversion; 4) and semantics editing'.

Regarding (1) and (3) the 3D building models used in this work were downloaded from the Trimble *3D Warehouse* (Trimble, 2014) and then converted from the Trimble *SketchUp* format (Trimble, 2018) into CityGML using 3DIS' *CityEditor* (3DIS, 2017). *CityEditor* was used, as opposed to a fully automated solution such as *FME* from Safe Software (2017) because it operates as a plug-in for *SketchUp* and therefore allows easy viewing, editing and export to CityGML. During export to the CityGML format the highest precision available was used, although of course this did not have a bearing on accuracy if the model was originally produced with low accuracy. Regarding (2) and (4) some optimisation and semantics editing was conducted using *CityEditor*. A separate CityGML validation tool such as CityDoctor (2017) was not used to correct or reject poor quality 3D building models. The rationale for such an approach was to prove that the method was scalable to the variety of 3D building models available. Ledoux (2013) and Wagner *et al.* (2013) also made recommendations regarding validation of CityGML models, the latter including semantic validation. During the conversion processes used here a CityGML syntactic structure was enforced, which effectively validated the semantic structure of the 3D building model. See Appendix C for details on the *SketchUp* to CityGML conversion, including more detail on model optimisation and semantic editing.

Biljecki *et al.* (2016b) determined that non-planarity of the polygon faces in a model was one of the most common errors in CityGML datasets. They also concluded that the error was generally only a few cm at most, i.e. well within LOD3's ±0.5m proposed accuracy standard. Consequently, while initial experiments with planarity checking were conducted, an assumption was subsequently made that all faces in the 3D building models used here were planar. Nonetheless, a more general positional accuracy check was still included in the method used here, in order to reject those faces with a variation beyond 0.5m – see Chapter 5 for more here.

2.4 Architectural Style

Four architectural styles were used in this work (see below), spanning the period from 1066 to the present day. Following the Renaissance, which for British architecture began around 1530 (Pevsner and Sambrook, 2010), the design concepts of Roman architecture were extensively reintroduced to architectural design. Consequently, the evolution of architectural style from Roman times to the present day are summarised below. In doing so, definitions are given for some well-used terms used to describe architecture, including describing architectural components. Such terms are used to discuss the results of this work. That said, in using detail when describing the architectural components this thesis does so for the purpose of later illustrating the challenges for the method, rather than necessarily being able to detect such fine-scale component detail *per se*. See Chapter 3 for an explanation of the level of detail that methods detect.

British architectural historians describe much of the country's architecture as conforming to one of three architectural *traditions*:

- Gothic (late C12-present)
- Classical (early C16-present)
- Modernism (1925-present)

  (see: Drolet, 2004, p. 63; Gorst, 2003, p. 63; Pevsner Architectural Guides, 2013; Wheeler and Whiteley, 1992, p. 18)

Admittedly, use of the phrase 'modernism tradition' could appear to be a contradiction in terms, and indeed its use is not particularly commonplace. However, for the purposes of consistency and ease of reference the phrase is used in this work.

Of course, these designations are not unique to Britain, being in use across Europe and beyond. The majority of architectural styles in Britain belong to one of these traditions, though there are some exceptions, including the stand-alone pre-Gothic styles of:

- Roman (43-c.410 A.D.)
- Anglo-Saxon (C7-mid-C11)
- Saxo-Norman (c.1060-1100)
- Norman (1066-late C12)

  (Pevsner Architectural Guides, 2013)

There are also some later, stand-alone styles including the 20th century Arts and Crafts, Art Noveau, Art Deco styles and neo-vernacular (Pevsner Architectural Guides, 2013). Again, these stand-alone styles are often present in Europe and beyond. In continental Europe, Norman is referred to as Romanesque.

The four styles used in this work relate to the traditions as follows:

- 20th century styles (*referred to from this point forth as* 'C20')

  The first architectural 'style' is 20th century, with buildings belonging to any of the three traditions or a stand-alone style of the period. Buildings from the 20th century were chosen on the basis that they are the most prevalent buildings in Britain, based on housing figures for England which show that 80% of homes were built post-1919 (see: DCLG, 2016; IHBC, 2016).

- 'Georgian-Regency' (1715-1830)

  The second is the combination of two Classical tradition styles: Georgian and Regency, which overlap in time and span the period 1715-1830.

- 'Gothic' (late C12-present)

  The third is actually a tradition: Gothic.

- 'Norman' (1066-late C12)

  The fourth is a stand-alone style: Norman.

It was mentioned in Section 2.2 that architectural style can be used as an indicator of building age. To do so is not without its challenges though: while the detection of the styles used in this work will give an indication of oldest likely age, the continuance of the traditions to the present day means it is not possible to more accurately determine the age of the buildings depicted in the 3D building models. Think Gothic Revival buildings

(a)

Image cannot be displayed due to copyright

Ritual Complex,
Fengchu
Shaanxi Province,
China
(c. 1100-1000 B.C)

(b)

Image cannot be displayed due to copyright

Great Stupa
Sanchi
Madhya Pradesh
India
(c. 100 B.C.)

Figure 2.5: Examples of Different Types of Symmetry in Architecture
(*a*) Bilateral symmetry, where 'only one plane can divide the whole into identical halves' on a median axis i.e. reflectional symmetry via that axis. Note that the building skin has been removed on the left-hand side of the symmetry plane, for the purposes of the illustration only. Also, note that not all reflectional symmetry is bilaterally symmetric. (*b*) Rotational, also known as radial, symmetry is the property of a shape when it appears unchanged despite partial rotation. Source: Ching (2014, pp. 360-361).

(mid-17th century onwards) and neo-Georgian buildings of the 20th century (Pevsner Architectural Guides, 2013). Additionally, as Lee *et al.* (2015) assert, the boundaries between styles are not always clear, making detection of style challenging. Nonetheless, the range of construction dates for the buildings in each of the four styles were chosen in order that each of the four date ranges did not overlap. The rationale for this was to enable future work which may detect more detailed style and hence provide a more accurate age determination.

Stylistically, there are some key differences between the pre-Gothic styles, Gothic, Classical and Modernism. The use of symmetry and regularity, for both overall building form and in the distribution and design of architectural components such as windows, is a design strand which runs through many architectural styles, although the type of symmetry used does tend to vary over time (Williams, 1999). Indeed, Dehbi *et al.* (2016a) use the automated detection of symmetry in 2D building footprints to infer the symmetrical properties of likely façades, essentially using the type of symmetry as a proxy for architectural style, though their work falls short of detecting the style itself. illustrates the different types of symmetry found in the four styles.

Arguably, older buildings demonstrate more asymmetry or more irregularity in the distribution of architectural components. The reason for this is that they are more likely to have been altered over time, often becoming an amalgam of styles as a result. The 12th century Norman churches at Kilpeck, Moccas, Heckingham and Askham Bryan in the dataset for this work demonstrate a combination of original features and newer, irregularly placed Gothic windows and doors (see BM_N_1, BM_N_3 and BM_4 respectively). This is in contrast to the newer Georgian-Regency dataset buildings, which demonstrate a purer maintenance of the original style, including a more regular distribution of windows, despite the passage of 200-250 years. For more details regarding the 3D building models used in this work see Chapter 3.

Architectural historians use the term ‘rhythm’ to describe the repetition or alternation of components such as columns or windows, and ‘interval’ to describe the sizes of the spacing between them (Ching, 2011, p. 55). ‘Order’ meanwhile is the ‘logical, harmonious or comprehensible arrangement in which each element of a group is properly disposed with reference to other elements and its purpose’ (Ching, 2011, p. 54).

Unlike the majority of current window detection approaches (on façade images), which rely on the symmetry or regularity of their distribution, or use styles that comprise standardised designs (see Section 2.8), the method in this work was not restricted by such symmetry, regularity or standardisation constraints. As a result, the method was able to detect individual windows and doors whose distribution on a façade followed no pattern. Together with the pre-Gothic styles, the stylistic differences between the three traditions are described below. This includes a discussion on symmetry and regularity.

The HOG-based template matching approach (see: Xiao, 2013; Zhang *et al.*, 2013b) used in this work histograms edge orientations within cells of an 8×8 pixel grid. Edges are lines marked out by having similar intensity, such as the edge represented in an image by a line in the real world, which defines the outer part of a window frame. Intensity is an approximate analogue for brightness (Sonka *et al.*, 2014). The strength of an edge, also termed ‘magnitude’, is represented by its gradient, where the gradient is oriented in the direction of the greatest change in pixel intensity (Sonka *et al.*, 2014). See Section 2.6 for more background to computer vision, including HOG-based template matching.

The gridding approach used by HOG-based template matching, together with the potential for the texture maps provided with the 3D building models to be of varying quality, may present a challenge for the HOG-based template matching method to detect design details. The below therefore also summarises key changes in form and complexity in the design of windows and doors between the four styles used in this work, i.e. whether the design of different windows and doors of the same style is standardised or highly varied.

2.4.1 Pre-Gothic Styles & Norman (1066-late C12)

Roman architecture is a derivative of earlier Greek architecture. It is typified by a prevalence of an axially symmetric arrangement of building components, by the repetition of components spaced at regular intervals and by the use of proportion for their sizing and distribution (see: Jones, 2003, pp. 40-41; Williams, 1999). Roman architecture introduced orders in the combined form of columns with horizontal entablatures. There are five styles of Roman order: Doric, Tuscan, Ionic, Corinthian and Composite, where the latter combines Ionic and Corinthian components (Pevsner and Sambrook, 2010). See Figure 2.6 for an illustration of the orders. All but the Tuscan are originally derived from Greek orders (Berning, 2011). These Roman and Greek orders were reintroduced as part of the Classical tradition (see: Berning, 2011; Waters, 2017a) hence their mention here.

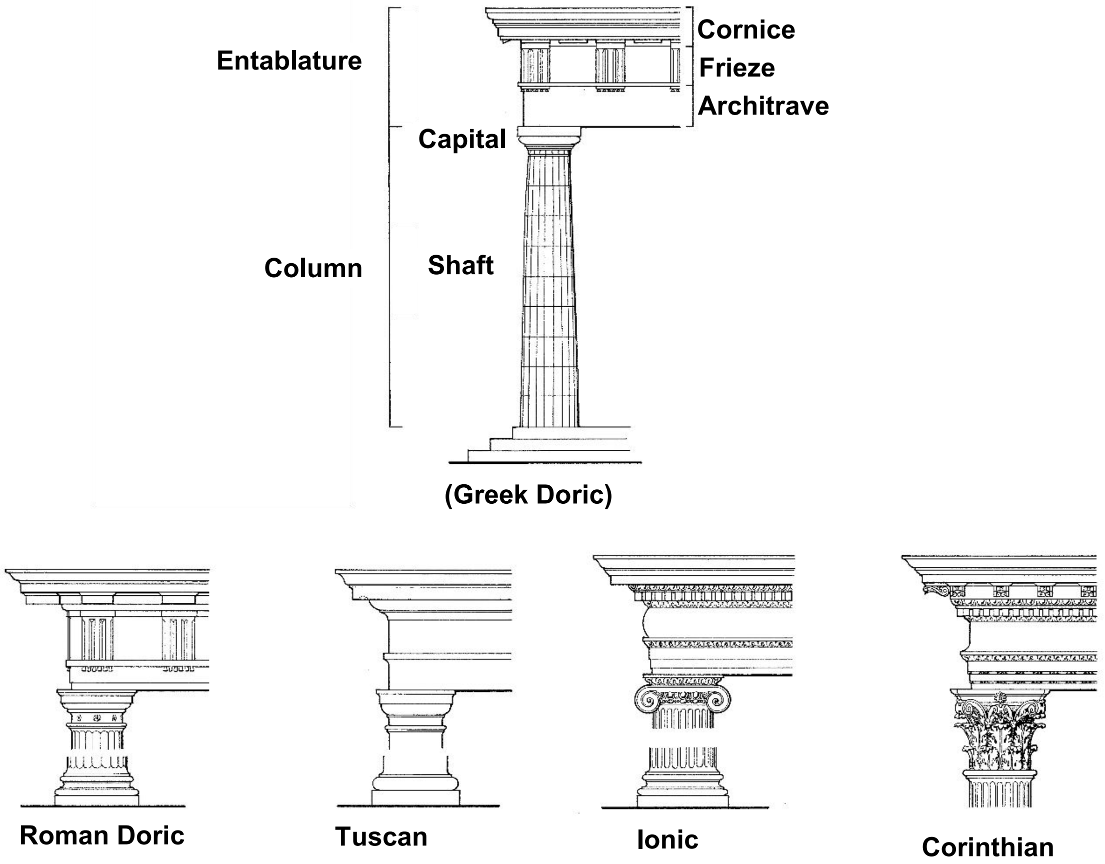


Figure 2.6: Orders used in Roman Style & Georgian-Regency Style
Source: Pevsner and Sambrook (2010).

The Norman style turned instead to the use of bilateral symmetry (Williams, 1999). Because this work does not include Anglo-Saxon or Saxo-Norman, they are not discussed here, nor are Roman windows or doors for the same reason.

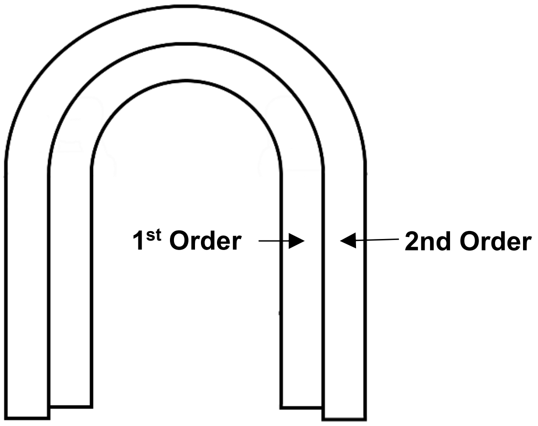


Figure 2.7: Concept of Orders in Norman Style Windows & Doors
Example shown is 2-order. Windows or doors with 1, 3, 4 or more orders are also possible.

The Norman style is typified by the use of a round arch for both windows and doors (Fernie, 2002, p. 268). After the Romans left in the 5th century it was not until 675 that glazing reappeared, though its use in Norman times was rare, in addition to which barely any glazing from then until the 16th-century Reformation still exists (Marks, 2006). Consequently, the glazing in extant Norman windows is overwhelmingly of the Gothic tradition or other styles. Doors tended to be constructed of a number of interconnecting vertical wooden planks, often held together by

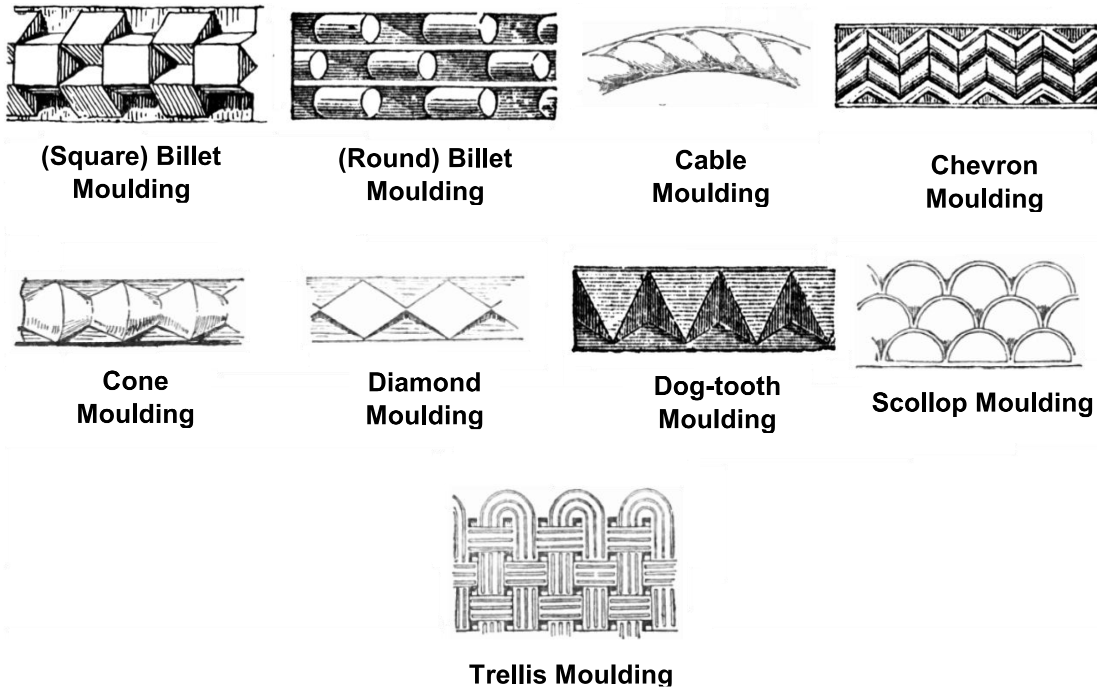


Figure 2.8: Examples of Types of Carving on Norman Window & Door Orders & Arches
Sources: Bell (1914, pp. 28-29); Hamlin (1916, pp. 261, 453).

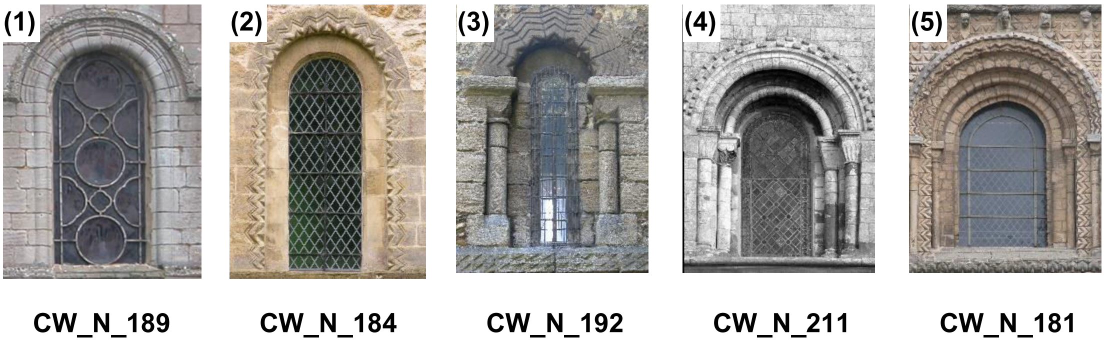


Figure 2.9: Norman Style Window Examples & Descriptions of their Orders
(*1*) One order. Plain, continuous, hood mould above arch. Cathedral of the Holy Trinity, Ely, Cambridgeshire. (*2*) Two orders. 1st (inner) order plain, continuous. 2nd (outer) order continuous chevron moulding. Church of St Micheal, Stewkley, Buckinghamshire. (*3*) Two orders. 1st order plain, continuous. 2nd order has shafts on bases with capitals with chevron moulding in arch. Leper Chapel, Cambridge. (*4*) Two orders. Both have shafts with capitals and, plain, continuous arches. Billet moulding string course. The Parish Church of St Andrew, Steyning, West Sussex. (*5*) Four orders. 1st order plain, continuous. 2nd order has shafts and capitals, plain, continuous arch. 3rd order continuous chevron moulding. 4th order has shafts and capitals, chevron moulded continuous arch. Above the arch is a billet moulded chamfer. Cathedral of the Holy Trinity, Ely, Cambridgeshire.

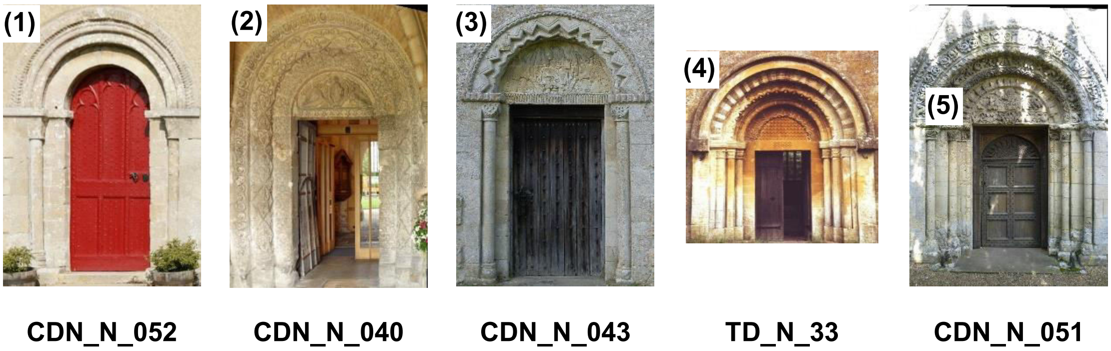


Figure 2.10: Norman Style Door Examples
(1) Church of St Swithun, Martyr Worthy, Hampshire. *(2)* Malmesbury Abbey, Wiltshire. *(3)* St. Giles Church, Water Stratford, Buckinghamshire. *(4)* St. Michael and All Angels Church, Guiting Power, Gloucestershire. *(5)* St Mary's Church, Patrixbourne, Kent.

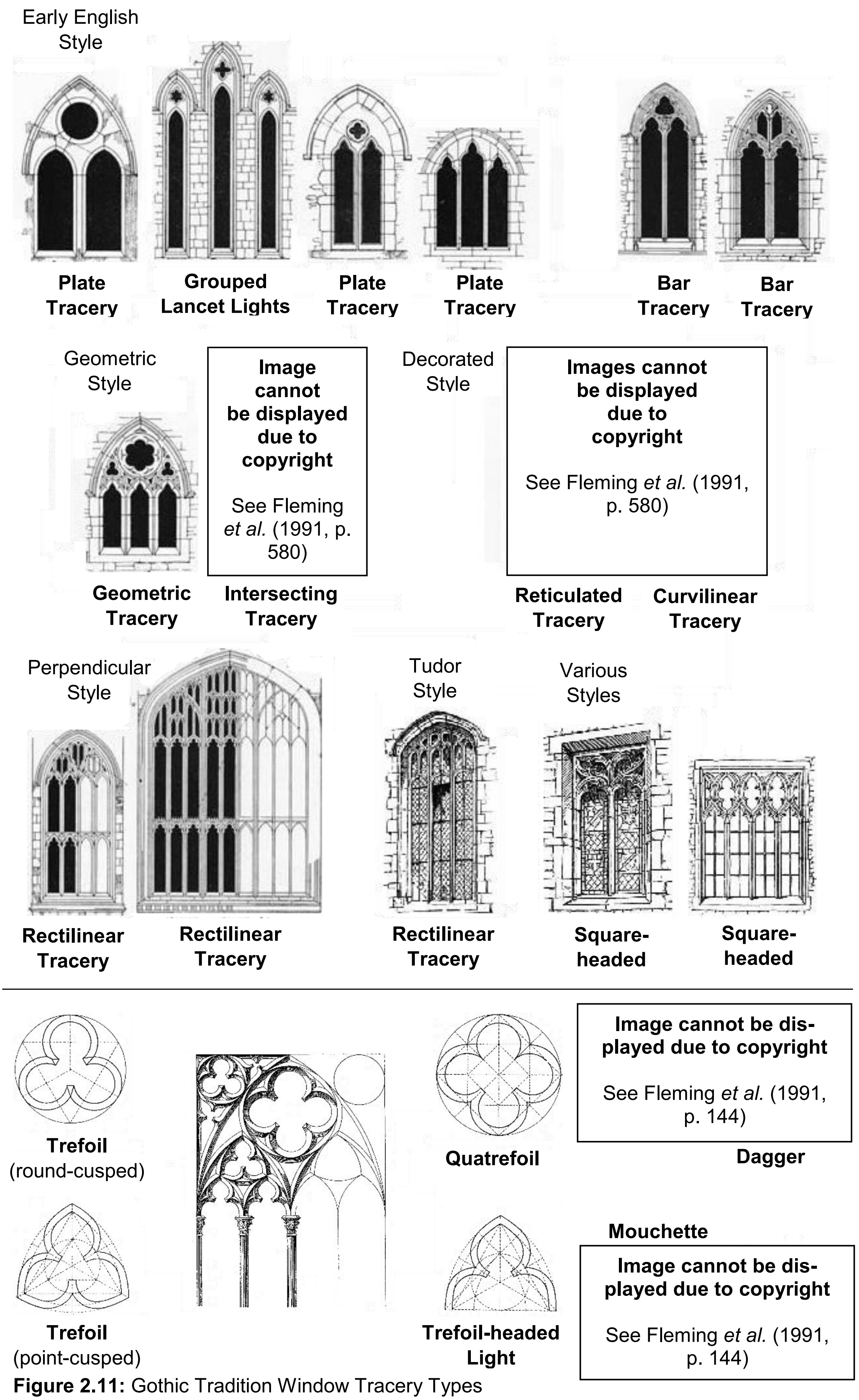


Figure 2.11: Gothic Tradition Window Tracery Types
Top: tracery types. *Bottom*: examples of features. Sources: Fleming *et al.* (1991); Fletcher (1905); Gill (1916); Pevsner and Sambrook (2010); Rovira y Rabassa (1897).

wide hinges (see: Hendricks, 1999; King's College London, 2016). Norman arches also used the principle of orders, in this case to describe the series of recessed arches beside the window or door (see: King's College London, 2016; Pevsner and Sambrook, 2010). For example, a window with ‘2 orders’ will have two parallel bands running from the bottom left corner of the glazing, up around the edge of the glazing, over the arch of the window and all the way down to the bottom right corner of the glazing. See Figure 2.7. These bands can include columns and can contain a variety of often intricate stone carving styles, as illustrated in Figure 2.8. Figure 2.9 shows examples of Norman window designs and describes the orders on the examples. Note that a ‘hood mould’ is a ‘projecting moulding over the arch of a window or door’, and that a ‘string course’ is a ‘horizontal course of brick or stone’ (Ching, 2011, pp. 159, 274).

Figure 2.10 shows examples of Norman door designs. Note that, as mentioned above, neither the doors themselves nor the glazing are original and may be replacements of a later architectural style.

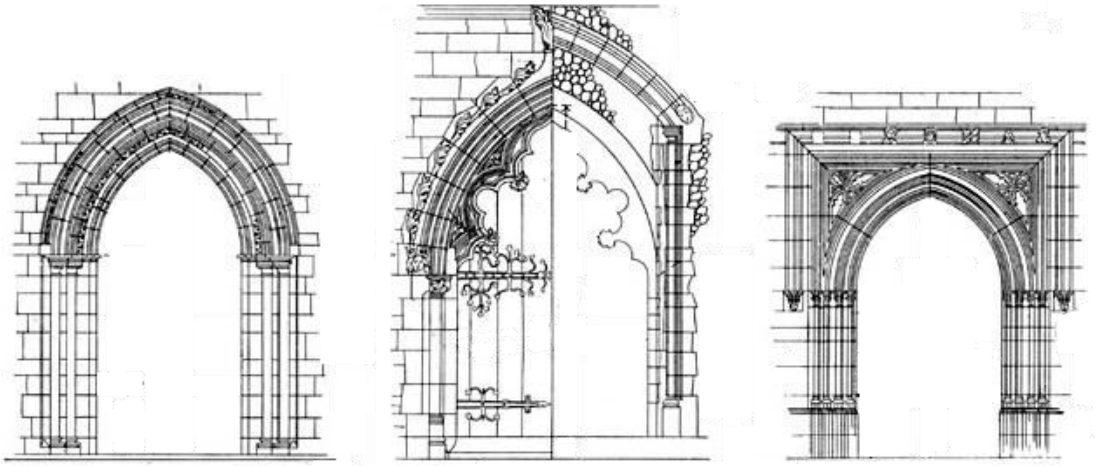
2.4.2 Gothic Tradition (late C12-present)

Gothic continued the use of bilateral symmetry for design, however overall the design can give the appearance and distribution of components an impression of irregularity (see: Waters, 2017b; Williams, 1999). Gothic was marked by the emphasis of the vertical, by the introduction of the pointed, not round-headed, arch and by the use of tracery (Waters, 2017b). Tracery is an ‘openwork pattern of masonry or timber in an opening’ such as a window (Pevsner and Sambrook, 2010). The type of tracery, in association with other design features such as the use of a particular type of pointed arch define a number of Gothic styles.

The styles of Gothic are:

- Early English (c.180-c.1250)
- Geometric (c.1240-1290)
- Decorated (late C13-late C14)
- Perpendicular (1320s-early C16)
- Tudor (1485-1603)

(Pevsner Architectural Guides, 2013)



Early English Style

Decorated Style

Perpendicular Style

Figure 2.12: Gothic Tradition Door Examples
Tudor sub-style design similar to Perpendicular style design except with lower, flattened arch. Note that the Decorated style example is split vertically to show two possible arch structures. Actual doors excluded for sake of illustration. Source: Fletcher (1905).

Note that there are two revival styles, namely 'Gothic Revival' (mid-17th century to present day) and 'Gothick' (c.1730-80), as described by Pevsner Architectural Guides (2013).

illustrates some examples of window tracery, described below. See also Appendix B for more background to the various sub-styles of tracery. Note from the figure how there can be great variety in the design, including the number of 'lights' (the vertical window panes or glazing panel divisions) and the number of tiers (horizontal stages, separated by dividers called 'transoms') and in the components of the ornamental carving that forms the mullions, such as cusps, foils, daggers and mouchettes (Pevsner and Sambrook, 2010). Note too that Gothic windows can also be square-headed.

Doors of the Gothic tradition also follow the pointed-arch form. As with the Norman style the doors tend to be constructed of interconnecting vertical planks, held together by horizontal hinges (Sharpe, 2011, pp. 121-123). illustrates examples of Gothic doors.

2.4.3 Classical Tradition (early C16-present) including Georgian-Regency (1715-1830)

From around 1530, as a result of the Renaissance, the majority of new buildings and renovations in Britain began to move away from what had been almost entirely Gothic design (Pevsner and Sambrook, 2010). While the symmetry was generally a departure from Roman axial-symmetry, to rotational and reflectional symmetry, design cues for the

Classical tradition were very much drawn from the works of the influential Roman architect Vitruvius (see: Jones, 2003, p. xi; Williams, 1999). Consequently, designs were typified by a return to an emphasis on the Roman concepts of proportion, symmetry, repetition, especially of windows, plus the use of Roman *and* Greek orders (see: Berning, 2011; Waters, 2017a).

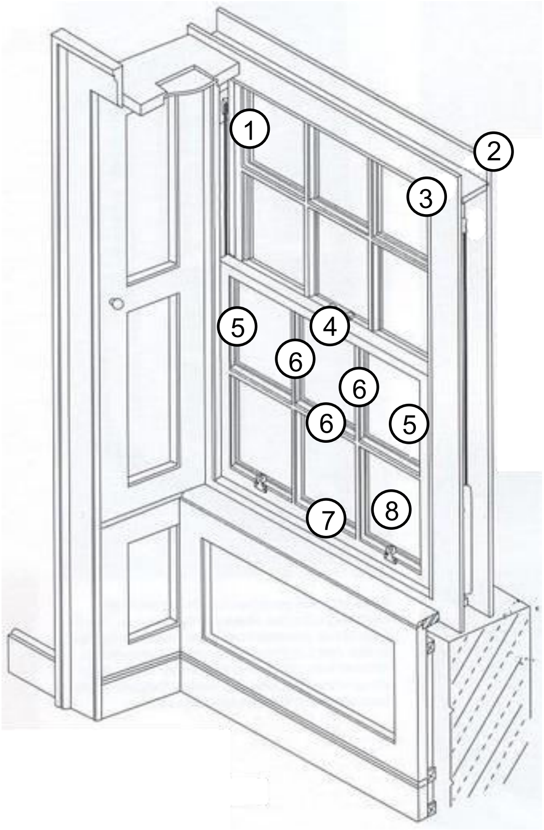
Georgian and Regency are, arguably, two of the most prevalent and well-known of the approximately 15 styles of the Classical tradition (Pevsner Architectural Guides, 2013). Strictly, 'Georgian' is a style grouping rather than a style in itself (Pevsner and Sambrook, 2010), though, arguably, it has become synonymous with the definition of a style. The Regency style is distinguished by 'thinner or more summary classical detail than the 18th-century norm' (Pevsner and Sambrook, 2010).

Sash windows have their origin in the late 17th century, after the start of the Classical tradition (Louw, 1983). The use of sash windows became a key feature of many of the styles of the Classical tradition, including Georgian and Regency. During that time the sashes comprised a number of glazing panels, arranged in grids. A window of two sashes, each with three glazing panels horizontally and two glazing panels vertically, is referred to as a '6-over-6' window (Pickles *et al.*, 2017). Windows were often square-headed but could also be arched. illustrates the structure of sash windows, including terms used to describe components and differing configurations of glazing panels.

The Classical tradition also witnessed the reintroduction of the panelled door from the Roman style (Hendricks, 1999) either with a single or double door in one doorway. Porticoes (porches) were also reintroduced from the Roman style, supported by columns,

either '*in antis'* or 'prostyle', i.e. flush with the building façade or standing away from the building façade respectively (Pevsner and Sambrook, 2010). The porticoes were often topped with a pediment including designs known as 'open' (triangular), 'segmental' (curved) or 'broken' (Pevsner and Sambrook, 2010). Fanlights were often above the door and were either semi-circular or rectangular (Pevsner and Sambrook, 2010). illustrates various Georgian-Regency doors including possible portico, pediment and fanlight combinations.

With the emphasis on the preservation of proportion, borrowed from Vitruvius, the aspect ratios of building components, including the spacing and sizing of windows and doors



KEY

1 **Parting Bead**
2 **Outer Lining**
3 **Inner Lining**
4 **Meeting Rail**
5 **Sash Stile**
6 **Glazing Bar**
7 **Bottom Rail**
8 **Glazing Panel / Pane**

(a)

Image cannot be displayed due to copyright

TZ_GR_23

(b)

Image cannot be displayed due to copyright

TZ_GR_14

(c)

Image cannot be displayed due to copyright

TZ_GR_6

(d)

Image cannot be displayed due to copyright

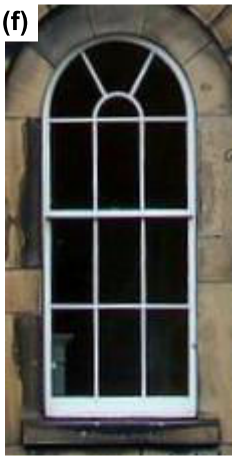
TZ_GR_7

(e)

Image cannot be displayed due to copyright

TZ_GR_5

(f)



TZ_GR_10

Figure 2.13: Georgian-Regency Style (Sash) Window Structural Description & Examples *Left*: Structural description of sash window. *Right*: Example sash window showing different combinations of glazing panels. Source: Allen (1994). (*a*) 3-over-3, Micklegate, York. (*b*) 3-over-6 sash window, Bedford Square, London. (*c*) 6-over-6 sash window, Sydney Place, Bath. (*d*) 9-over-6, Queen Square, Bristol. (*e*) 2-over-4, Roman Baths frontage, Bath. (*f*) Arch sash window, Charlotte Square, Edinburgh.

Images cannot be displayed due to copyright

Portico *in antis*, Open Pediment

Portico *in antis*, Open Pediment & Semi-circular Fanlight

Portico *in antis*, Open Pediment & Semi-circular Fanlight

Prostyle Portico & Rectangular Fanlight

Images cannot be displayed due to copyright

Portico *in antis*, Semi-circular Fanlight

Portico *in antis*, Rectangular Fanlight

Portico *in antis*

Portico *in antis*, Segmental Pediment

Images cannot be displayed due to copyright

Portico *in antis*, Broken Pediment

Portico *in antis*, Open Pediment

Portico *in antis*, Open Pediment & Semi-circular Fanlight

Portico *in antis*

Figure 2.14: Georgian-Regency Style Door Examples & their Design Components
All examples are distyle porticoes, i.e. two columns supporting the portico. Orders are Greek or Roman derivatives. Note how the doors themselves can be double or single. Sources: Getty Images (2017); Pevsner and Sambrook (2010).

and their panels, tended to be standardised (Curl and Wilson, 2015, p. 843). While demonstrates how Georgian-Regency windows follow standardised designs, clearly shows that Georgian-Regency door designs possess greater variety than their window counterparts. These are important points for this work. The HOG-based template matching method used here is somewhat insensitive to the size and positioning of the component parts of door and window instances in texture map images. Georgian-Regency windows, with their more standardised design, might be detected more readily than Georgian-Regency doors, and windows and doors of the other styles.

2.4.4 Modernism ‘Tradition’ (1925-present) & Neo-vernacular

As a general rule, Modernism heralded a departure from the symmetry and regularity of the Classical tradition toward asymmetry and irregularity (Tournikiotis, 2001, p. 76). Furthermore, the style tends to emphasise the ‘elimination of ornament’ including the use of swathes of smooth, unadorned concrete or render across façades (see: Pevsner and Sambrook, 2010; Waters, 2017c). Note that British Modernism mutated somewhat by reintroducing brickwork and timber (Fiegel, 2011). Indeed, some of the C20 buildings used in this work possess both brickwork and cues of Modernism.

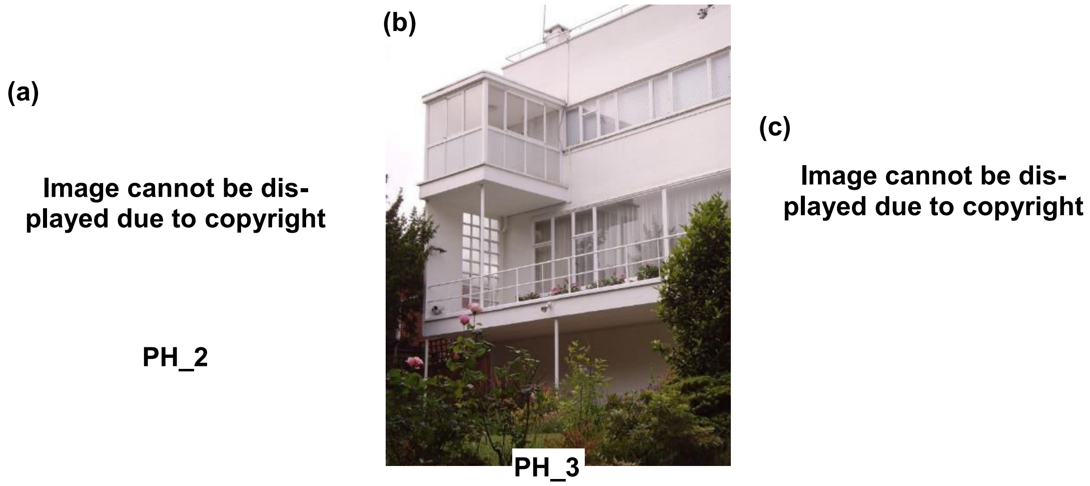

(a)
Image cannot be displayed due to copyright
PH_2
(b)
PH_3
(c)
Image cannot be displayed due to copyright

Figure 2.15: Modernism Window Examples
(*a*) Kensal House, London. See Fiegel (2011) for style designation. (*b*) Sun House, London. See Louw (2015, pp. 46-47) for style designation. (*c*) Old Church Street, London. Image source and style designation are from Waters (2017c).

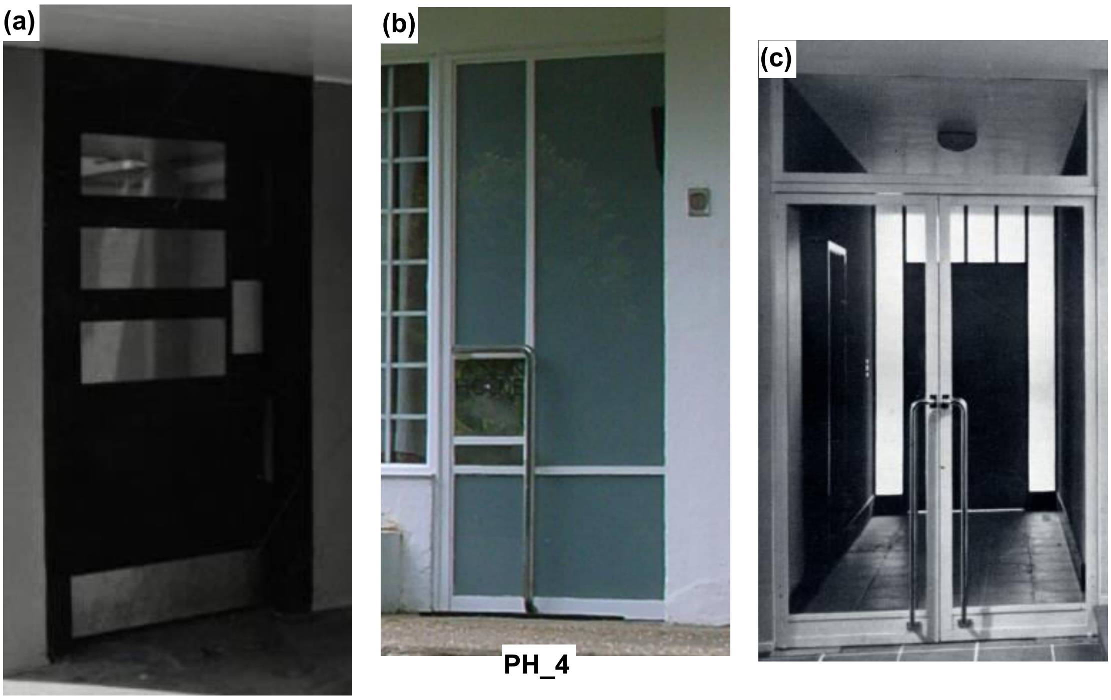


Figure 2.16: Modernism Door Examples
(*a*) Kensal House, London. Image source: Walker (2015a). (*b*) Sun House, London. (*c*) Old Church Street, London. Image source: Walker (2015b). For style designations for all images see the caption for .

Aside from one 3D building model (built at the turn of the 20th century), the construction dates for the C20 3D building models used in this work coincide with dates for Modernism. However, note that many 20th century British buildings lack any formal architectural style designation, despite the existence of style categories into which to place them. Such a designation would often have been carried out for older buildings by authorities or academics due to their cultural heritage significance. In contrast, some of the C20 buildings in this work lack such a formal style designation. Relatedly, note that some of the C20 style buildings in the dataset could also represent the stand-alone neo-vernacular style, which has been a tendency, especially in the last century, particularly in the 1970s (Pevsner Architectural Guides, 2013). Neo-vernacular 'drew on brick, tile and other traditional materials' during design, often using local material and approaches and was dubbed, perhaps unfairly, 'architecture without architects' (Curl and Wilson, 2015, pp. 517, 806). English Heritage (2017) states that the buildings in the new town of Milton Keynes tend to be examples of Modernism and neo-vernacular styles. The dataset in this work contains C20 buildings from Milton Keynes which may therefore represent Modernism, or the neo-vernacular style – other C20 buildings in the dataset have a similar appearance so might also be of the same tradition / style.

(a) PH_5

Image cannot be displayed due to copyright

(b)

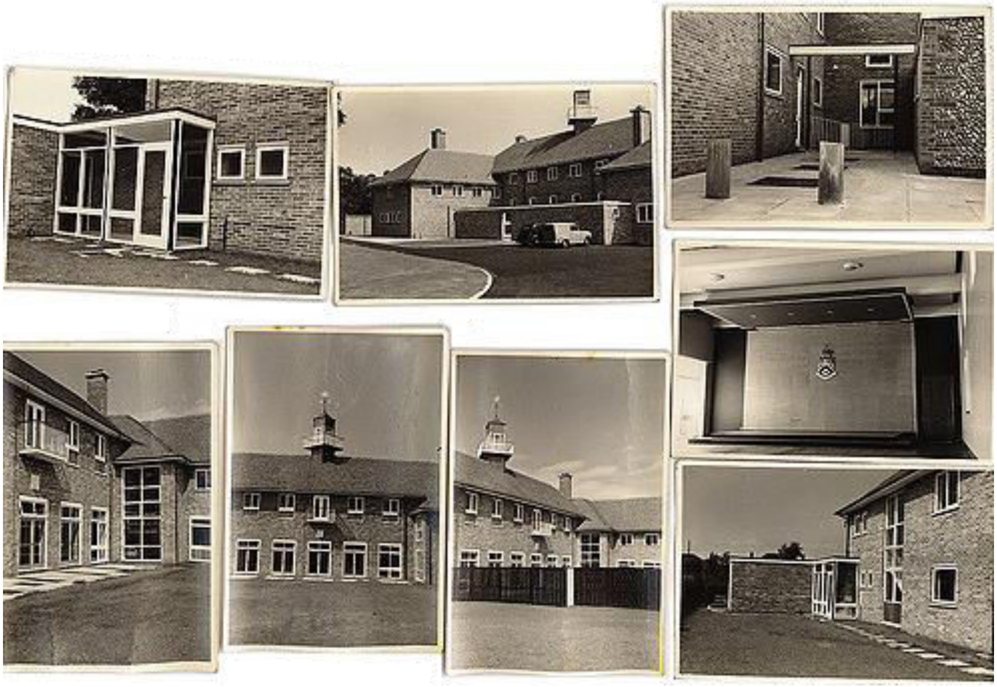


Figure 2.17: Neo-Vernacular Style Window & Door Examples
(*a*) Aviva Offices, Tanner's Moat, York. See City of York Council (2011) for style designation.
(*b*) 8 images of Gresham's School, Holt, Norfolk. See Roberts (2009) for style designation.

Windows in the architecture of Modernism, driven in part by a desire to bring large amounts of light into rooms, tend to use large expanses of glass, stretched either horizontally or vertically or in both directions (Louw, 2015, p. 44). Glazing panel configurations tended to be based on grids, i.e. a tendency toward regularity and symmetry within the window itself, but also simple asymmetric splits of rectangular panels (Louw, 2015, pp. 45, 48). See . Door designs within Modernism also tend toward large expanses of glass (or large, plain expanses of other material) as shown in Louw (2015). See for examples of doors from the Modernism tradition. Essentially, the component make-up of both windows and doors demonstrated a wide variety of configurations.

There is a dearth of literature that exists regarding the window types used in the neo-vernacular style, perhaps because it is such a relatively recent, common and diverse style. Nonetheless, the examples of Roberts (2009) and the Aviva building on Tanners Moat (City of York Council, 2011) also suggest a level of complexity and variety in the configurations of rectangular glazing panels, and a grid arrangement of glazing panels respectively. Door styles for neo-vernacular are also poorly researched, yet, as with its windows, the example of Roberts (2009) indicates a similar tendency toward an arrangement of components. See Figure 2.17 for examples of neo-vernacular windows and doors.

2.5 Reconstruction Techniques

It was mentioned in Section 2.2 that this work focusses on 3D building models and not point cloud or mesh representations of buildings. Nonetheless, their inclusion in this chapter is of relevance, for the following reasons. (1) Firstly, point clouds or meshes can be stages in a pipeline to reconstruct a 3D building model. (2) Moreover, the identification of windows and doors can also take place within such pipelines (see Section 2.8 for more discussion on this). (3) Perhaps most importantly, it should be noted that the enrichment that this work enables is essentially that of reconstruction, albeit reconstruction on an existing 3D building model. (4) Furthermore, it is clearly not feasible for NMAs to create 3D city models for large areas entirely manually. (5) Lastly, note that it is unclear whether the 3D building models used in this work were originally created for the Trimble *3D Warehouse* manually or via automated reconstruction. Consequently, methods in use for the automated creation of 3D building models are summarised below.

Automated methods for creating 3D building models include: the use of remotely sensed Light Detection and Ranging (LiDAR) point clouds; the employment of ground-based laser-scan point clouds; and so-called photogrammetric ‘image-based’ means, which use images to then create a point cloud (Biljecki *et al.*, 2016c). LiDAR and laser-scanning methods are *active* reconstruction techniques because they use range data, emitting light and recording the response to create a depth map (Pears *et al.*, 2012, p. 88). Image-based approaches represent *passive* reconstruction in that they ‘do not emit any illumination’, merely perceive it via an image sensor (Pears *et al.*, 2012, p. 88).

2.5.1 Point Clouds & Meshes

Structure-from-motion (SfM) and multi-view stereo (MVS) are two popular image-based approaches i.e. methods which derive a point cloud from multiple images (Özyeşil *et al.*, 2017). The following is based on Goesele *et al.* (2007) and Seitz *et al.* (2006). SfM produces comparatively sparse point clouds but deduces camera parameters including pose. MVS meanwhile requires the parameters but produces denser point clouds. Consequently, SfM is often used as a precursor to MVS, especially on images lacking camera parameters such as those in collections of internet photos. SfM obtains the parameters by determining correspondence of the same feature between two or more images (see Section 2.6), after which the homography of the transformation between the two images is calculated using their epipolar geometry (see: Fisher *et al.*, 2014, p. 90; Khan *et al.*, 2015).

SfM applications include: Agarwal *et al.* (2011); Fritsch and Klein (2017); Lingua *et al.* (2017); Sinha *et al.* (2008). MVS applications include: Furukawa and Ponce (2010); Goesele *et al.* (2007); Li and Li (2017); Thonat *et al.* (2016). However, it is the highly cited *Photo Tourism* work of Snavely *et al.* (2006) which, arguably, set the scene. The work of Snavely *et al* is also relevant to the work described in this thesis because it created a virtual cultural heritage experience of buildings, using tags applied to areas of the reconstruction, although, admittedly, it did so on point clouds. The ideas behind *Photo Tourism* were, arguably, continued, and extended with the work of Simon and Seitz (2008), and then the *3D Wikipedia* work of Russell *et al.* (2013). Appendix B provides more background to these methods.

From point clouds, triangulated meshes can be derived using readily available software, such as that from AEC software providers Autodesk and Bentley. Open-source tools are available too, such as *MeshLab* (ISTI - CNR, 2017) and those summarised by Remondino *et al.* (2017). Methods also exist which use image-based approaches to create a point cloud and then auto-derive a mesh e.g. see: Barrile *et al.* (2017); Ley *et al.* (2017); Waechter *et al.* (2017).

2.5.2 3D City Models

As Fritsch and Klein (2017) assert, no method currently exists which can automatically produce high quality 3D city models, either from imagery or point clouds. Indeed, Wang *et al.* (2015) discuss how many of the current methods actually require some level of manual intervention. Nonetheless, attempts have been made to fully automate the process including Yang *et al.* (2016) and Wang *et al*'s point-cloud methods, but once again both concede that some manual intervention is still, currently, unavoidable. Henn *et al.* (2013) do reconstruct CityGML, from point clouds, but only for LOD2 models i.e. with no windows or doors.

Musialski *et al.* (2013) provide an extensive survey of 3D building reconstruction techniques, including those which use procedural modelling approaches to reconstruct 3D city models using single façade images or building footprints. In their summary they note that procedural modelling methods tend to rely on previously derived shape grammars, or similar, to do so. Grammars express the 3D structure of the component parts of an object (Gkeli *et al.*, 2017), where the component parts in the case of this work would be elements such as windows and doors.

Procedural modelling has a number of advantages, including: being scalable to multiple grammars, especially where the grammars can be inversely modelled from existing buildings of the same type; and the potential to model many, large buildings more quickly than non-procedural methods (Dore and Murphy, 2017). However, a significant drawback is that they are ineffective when realistic and detailed geometry for small building features is required (Dore and Murphy, 2017).

The history of the use of grammars and procedural modelling in architecture is summarised in Müller *et al.* (2007). The work described in this thesis did not involve procedural modelling methods, so these methods are not described further here. That said, note that section 2.8.2, which critiques methods for detecting windows and doors during reconstruction, does include methods which use procedural modelling.

Gkeli *et al.* (2017) attempt to categorise reconstruction methods into: data-driven approaches (non-parametric); and model-driven approaches (parametric), such as procedural modelling, where such approaches require *a priori* information regarding the building's shape which then forms a grammar. Gkeli *et al* also note that a number of hybrid methods exist, which combine data and model-driven approaches. Additionally, Gkeli *et al* further categorise data-driven methods into those which use: plane fitting e.g. Arikan *et al.* (2013); filtering and thresholding including edge detection; segmentation e.g. Wang *et al.* (2016); and supervised classification. Note that while these examples of previous work result in 3D building models, the abovementioned categories also apply to reconstruction where the end result is: a new or enhanced point cloud or a mesh; or new line segments e.g. Ni *et al.* (2017); or one or more planes such as façades e.g. Seo *et al.* (2016).

To the best of this author's knowledge 'filtering and thresholding', or supervised classification methods alone, have not been used to reconstruct 3D building models. Nevertheless, as with methods which mix data and model-driven approaches, the aforementioned hybrid approaches are common. Lastly, note that the methods used in this work would be categorised as being 'filtering and thresholding including edge detection' (i.e. computer vision), and indeed 'supervised classification' (an aspect of machine learning) is also applied to the match-score results (though not on the texture map images themselves). See Section 2.7 for background to machine learning. See Section 2.8 for a discussion on the use of 'filtering and thresholding including edge detection' on imagery for the detection of windows or doors.

2.6 Computer Vision

2.6.1 Background & Challenges

Computer vision aims to 'duplicate the effect of human vision by electronically perceiving and understanding an image' (Sonka *et al.*, 2014, p. 1). It is used in a wide range of applications including: the interpretation of medical imaging; inspection during manufacturing; in retail such as at automated check-outs; automotive safety including detecting obstacles such as people; surveillance; optical character recognition (OCR) including within postal systems and for number plate recognition; face detection; finger print recognition; and reconstruction such as SfM (Szeliski, 2010, pp. 2, 5). In terms of this work it is important to have an understanding of the challenges faced by computer vision techniques, not least in order to be able to explain the results obtained. Sonka *et al* outline the challenges for computer vision as follows (Sonka *et al.*, 2014, pp. 4-5).

The first challenge is that of interpretation i.e. that there are potentially a number of different ways to interpret an image, where in the real world the object in the image does not have more than one interpretation. The real-world object is said to be of a certain 'class' (Fisher *et al.*, 2014, p. 48) where in the work here a class might be either 'window' or 'door' or, at a lower level, 'Georgian-Regency style window' or 'Norman style door'. 'Class' is essentially used for the same purpose in both computer vision and in machine learning (see Section 2.7). In this work an object in the real world is either a door or a window class, not both. This is despite potentially confusing descriptions such as 'French windows' – architecturally such objects are still doors, albeit with a secondary use as a window. The availability of *a priori* information about the domain area can help with interpretation. Machine learning can also be used during interpretation, including the training of a classifier with known examples of the objects in images in order to predict the interpretation of unknown objects in other images.

The second challenge, as determined by Sonka *et al*, is the existence of noise in an image, noting that all real-world measurements are inherently noisy.

The third is that of having large amounts of data – images are often large – which, dependent upon the algorithm and implementation, can present runtime challenges despite advances in processing power and memory.

Fourth is the challenge presented by brightness in an image, where the brightness measured and represented in image pixels is a function of complex image-formation physics. Many computer vision techniques use variations in pixel intensity across an image. Intensity is an approximate analogue for radiance. Radiance is a function of irradiance (i.e. the type, intensity and position of the light source), and is based on the position of the camera and on surface geometry and reflectance. One might therefore conclude that such sensitivity to radiance could present its own challenges for a camera sensor and subsequently for computer vision algorithms. For example, variations in brightness across an image might make detection of the edges, such as window frames, difficult if any of the following scenarios occur: if the sun was behind cloud meaning that lighting was poor; if the camera was obliquely positioned with respect to the building façade; if the façade was curved; or if the window panes contained significant amounts of reflection.

The fifth, and final, challenge summarised by Sonka *et al* is that it is common for image analysis algorithms used in computer vision to focus *locally* e.g. around a pixel and its neighbourhood, when in fact an absence of the global view, i.e. the whole image, can make the correct detection of objects in the image problematic. In practice, most current methods tend to operate locally (Khan *et al.*, 2015).

2.6.2 Computer Vision Methods

This section provides background to various computer vision methods, specifically with regard to object detection. One such approach uses the concept of image segmentation, a field that includes work by: Gadde *et al.* (2016); Jampani *et al.* (2015); Yu and Wang (2016). Relatedly, saliency detection methods can be used, including those of Lin *et al.* (2014) and Xu *et al.* (2014a). Visual saliency can be described as how well an object stands out from its surroundings (Xu *et al.*, 2014a). The Bag-of-Words or Bag-of-Visual-

Words (BoW / BoVW) approach can also be employed to object detection, where the method is summarised by Sivic *et al.* (2005) as follows. Feature descriptors are first obtained from images, for example using the highly cited and still extremely popular, SIFT (Lowe, 2004). Next, a codebook is created, containing visual codewords representing the image patches, from which a histogram of codewords for each image is calculated. Examples of its implementation include the work of dos Santos *et al.* (2016) and Zheng *et al.* (2014). Separately, various template and tile matching approaches can also be used for object detection, such as those of: Korman *et al.* (2017); Lian *et al.* (2017). Many of these approaches use machine learning of some form (see section 2.7 for background to machine learning).

With respect to this project there is a lineage of work concerned with identifying architectural components using computer vision – early undertakings are discussed in detail within Koutamanis and Mitossi (1993). Section 2.5 has already mentioned some methods which use computer vision during reconstruction of buildings. An early foray into the use of SfM to recreate structured 3D building models was that of Debevec *et al.* (1996) while Iqbal and Aggarwal (2002) used edge-detection to infer architectural features. Of particular relevance to the work here is the method of Johansson and Kahl (2002) and subsequent work by Mayer and Reznik (2005) and Reznik and Mayer (2007), which all had varying levels of success detecting detailed architectural features such as windows.

At this stage it is also worth mentioning that the remote sensing and photogrammetry community undertake a considerable amount of study into object detection – of some relevance here is work on detecting building outlines, often with feature and object detection methods mentioned herein, such as recent studies by: Du *et al.* (2015); Verykokou and Ioannidis (2016).

Note that the pre-processing of images may improve object detection by creating a more readily detectable abstraction of the object. Possible pre-processing approaches include the use of projective transform or distance transform, edge detection and blurring. Distance transform provides the distance from some subset of the image where the subset location(s) have a pixel value of zero, the pixels closest have low values and the pixels the furthest away have high values (Sonka *et al.*, 2014). Edge detection might be used to attempt to make otherwise disjoint lines more coherent such as Kang *et al.* (2007). Blurring can be achieved by applying a smoothing filter such as Gaussian (Sonka *et al.*, 2014, p. 140). The use of pre-processing in this work is detailed in Chapter 3.

2.6.3 Features, Detectors & Descriptors

Within computer vision 'features' represent either a 'distinctive part of something', where the eyes would be distinctive features on a face, or they correspond to an 'attribute derived from an object' (Fisher *et al.*, 2014, p. 96). As a reminder, the latter definition is akin to the use of the term within machine learning. Broadly, feature detectors can be categorised as edge detectors, corner detectors, and blob detectors where edges, corners and blobs are types of feature (Li *et al.*, 2015). In the context of computer vision, a descriptor is a set of vectors (also called a feature vector) which provide 'a summary description that can be compared with other descriptors in a database to obtain matches according to some distance metric' (Fisher *et al.*, 2014, p. 129). Once a feature has been detected a descriptor can then be extracted. Popular descriptors include SIFT, HOG, speeded up robust features (SURF) from Bay *et al.* (2006), binary robust invariant scalable keypoints (BRISK) from Leutenegger *et al.* (2011) and fast retina keypoint (FREAK) from Alahi *et al.* (2012). Note that some methods, such as SIFT, SURF and HOG, include both a detector and descriptor, and that it is often possible to mix different detectors and descriptors in an implementation. Incidentally, SIFT, SURF and HOG are all histogram based. Compared to methods which do not bin the information in an image, the use of histograms can result in a shorter runtime when performing calculation on descriptors (Cevikalp and Triggs, 2017).

As mentioned, SIFT is highly cited and remains popular, forming the basis of many SfM approaches for example. Using eight reference datasets, including façade images, Khan *et al.* (2015) evaluated and compared SIFT's performance for both image matching and homography. They concluded that SIFT still outperformed other popular methods, which included HOG, SURF, BRISK and FREAK.

SIFT computes a descriptor at interest points only, and not across the whole image or image region (Szeliski, 2010). It is therefore, arguably, more suited to image matching and the locating of specific instances of objects rather than generic object detection. HOG meanwhile computes a descriptor over the entirety of a regular grid. As such it is able to capture 'subtle variations in orientation' and position in object outlines and structure (Szeliski, 2010, p. 585). HOG is also one of the most popular and successful descriptors – *the* most for pedestrian detection (Sonka *et al.*, 2014, p. 512). The use of histogrammed gradients within grid cells also means that it is also somewhat insensitive to the exact location of edges in an image. Consequently, HOG should, compared to SIFT, be more able to deal with, for example, variations in the positioning of component parts of windows and doors which are likely to occur due to subtle changes in architectural style and building techniques. Broadly, SIFT is invariant to rotation of entire objects compared to HOG (which is not rotation invariant), both methods are scale invariant and HOG is generally faster due to the higher dimensionality of the SIFT descriptor (see: Hassaballah *et al.*, 2016; Takahisa *et al.*, 2013). Arguably, for the purposes of this work, the poorer performance of HOG when applied to rotated objects is less important. Moreover, windows and doors are generally likely to be vertically inclined on the images used. By way of further support for the use of HOG, its use of histograms of gradients across a grid provides some local insensitivity to the rotation of component parts of the window or door objects. It is for these reasons that SIFT was chosen for the image matching part of this work, and HOG for object detection, albeit within a HOG-based template matching approach. Sections 2.6.4, 2.6.5 and 2.6.6 provide more technical background to SIFT, HOG and template matching approaches respectively.

2.6.4 Scale-Invariant Feature Transform (SIFT)

SIFT allows for the identifying of matching features between images (correspondence) through the creation and then comparison of keypoint (feature) descriptors. These are obtained through a pipeline of: scale-space (Witkin, 1984) ‘extrema detection’ via a difference of Gaussian (DoG) approach; keypoint localisation; and finally the creation of a keypoint descriptor representing the local image gradients for the region (patch) around a keypoint (Lowe, 2004). Each keypoint has 128 elements (dimensions). These are formed from 4×4 keypoint histograms, each of which has 8 bins i.e. 4×4×8 = 128 (Hassaballah *et al.*, 2016). shows a schematic representation of the SIFT descriptor. The computation of keypoints and descriptors is achieved as follows, based on Lowe (2004) and Desolneux and Leclaire (2017).

Keypoints are computed from the detection of local extrema, after which their positions are refined to sub-pixel precision. Subsequently, the extrema of the Laplacian of Gaussian (LoG) with low magnitude that are located on edges are discarded. In order to compute the SIFT local descriptor for a keypoint, one or more dominant orientations are computed. For each detected orientation a histogram angle is computed, quantised on eight values, using a grid of 4×4 square regions around the keypoint. The histogram angle is computed in each grid cell. The square regions have one side parallel to the detected orientation. When forming the histogram each pixel effectively votes using a weight that is a function of the gradient norm value at the current scale, and of the distance to the keypoint centre. Finally, the 16 histograms are concatenated to form the feature vector and normalised by quantising to 8-bit integers. This last step also includes linear splitting of the vote angle ‘between the two adjacent quantised angle values’.

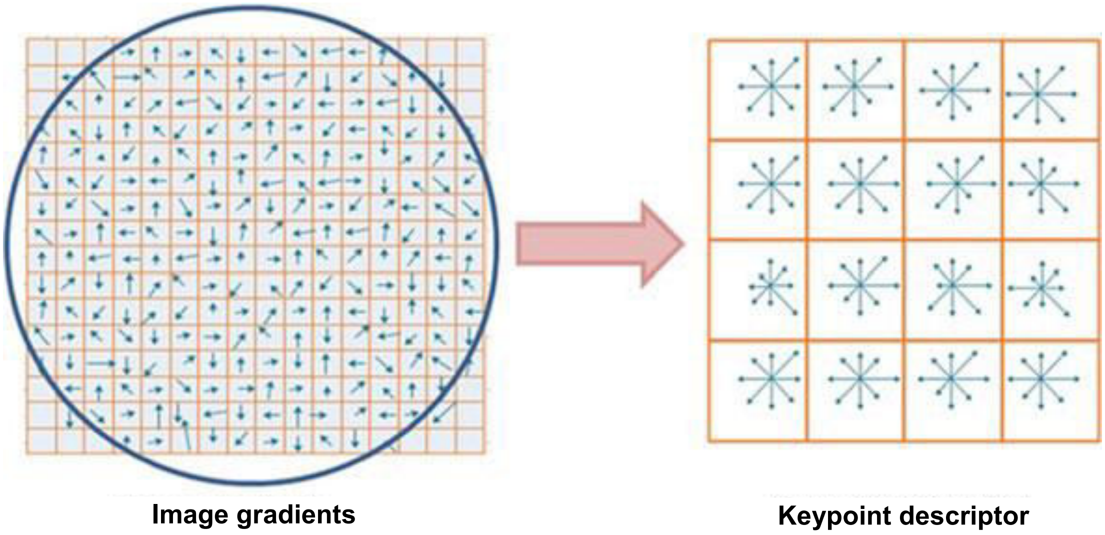


Figure 2.18: Schematic Representation of SIFT Descriptor
16×16 pixel patch and a 4×4 descriptor array (Hassaballah *et al.*, 2016).

As mentioned in Section 2.6.3 a distance metric can then be used to determine correspondence. This can commonly be achieved using a nearest-neighbour approach in what is called ‘feature space’ (see also Section 2.7), based on the Euclidean norm between descriptors. Such an approach can be supplemented with thresholding to reduce the number of features to be processed, thus reducing subsequent runtime. One such method includes thresholding on the ‘ratio between the distances to the nearest and the next nearest’ descriptor (Hassaballah *et al.*, 2016). Essentially, the ratio between the distances needs to be above a threshold to ensure a good and unique match.

Methods such as Random Sample Consensus (RANSAC) from Fischler and Bolles (1981) can be used to reduce outliers which may occur – this is achieved through the generation of a fundamental matrix for the perspective transformation between the two images, where those matches which are inconsistent with the transform can then be removed. The use of thresholding in advance can be advantageous to RANSAC, on the basis that outliers are likely to be more clear-cut following thresholding. shows an example of image matching and feature correspondence, which has included the use of RANSAC in an attempt to remove outliers.

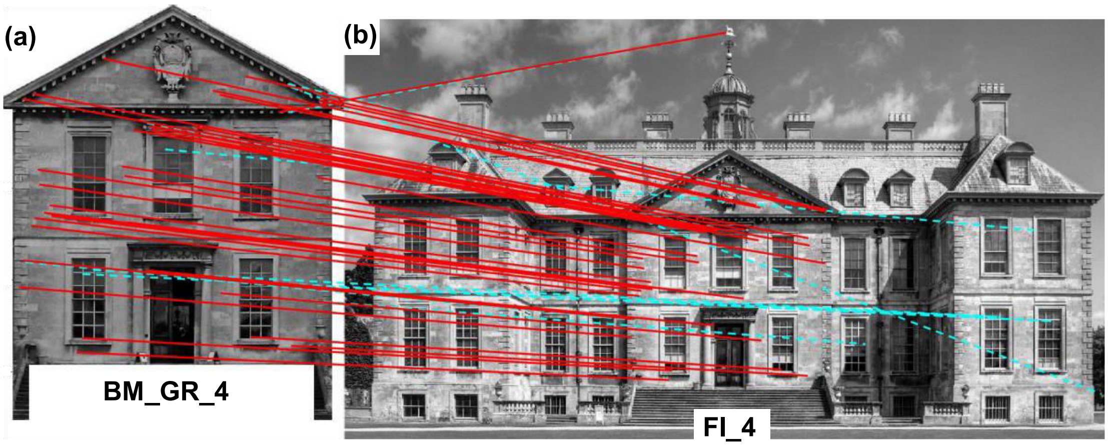


Figure 2.19: Image Matching & Feature Correspondence with SIFT & RANSAC
(*a*) Texture map image for 3D building model of Belton House, Lincolnshire. (*b*) *Flickr* image for the same building. Red lines are feature correspondence achieved with SIFT descriptors i.e. the same region on the actual building, in each image. Cyan lines are outliers removed with RANSAC. Note however that not all red lines show correct correspondence, demonstrating that the method is not perfect, as with all computer vision approaches.

2.6.5 Histogram of Oriented Gradients (HOG)

The following description is based on Dalal and Triggs (2005). In order to form a HOG descriptor an image is divided into a grid. Each grid cell contains an accumulation of 1D histogram of gradient directions over the pixels of that cell. The descriptor is of dimensionality 2×2×9 = 36 (in unsigned form, see below). The formation of the descriptor is described in more detail below.

There are 9 evenly spaced 'unsigned' gradient orientations between 0°-180°. Dalal and Trigg's implementation of HOG provides for an 'unsigned' and a 'signed' version of the descriptor, where the dimensionality component for each is 9 and 18 respectively, the latter representing gradient orientations from 0°-360°. In practice the original work found that an unsigned approach gave the best results for pedestrian detection. The gradients are those of local pixel intensity and correspond to edge directions. When combined, the histogram contents for all grids cells form the descriptor.

The descriptor is contrast-normalised (with 2×2 square HOG blocks) which gives 'better invariance to illumination'. This is achieved by 'accumulating a measure of local histogram energy' over somewhat larger spatial regions' known as 'blocks', the results of which are then used to normalise all cells in the block. Consequently, the descriptor makes uses of 'fine-scale gradients, fine orientation binning' and 'relatively coarse spatial binning' in an attempt to enable detection of the same type of feature in two separate

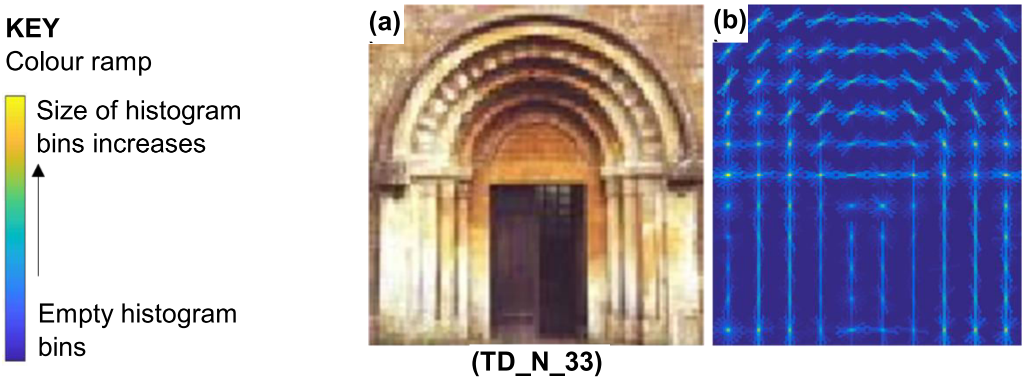


Figure 2.20: 31D HOG Descriptor for Example Norman Style Door Visualised as Polar Plots (*a*) A photograph of an example Norman style door. (*b*) A corresponding pictorial representation comprised of a polar plot for each grid cell in the descriptor. The polar plot represents the histogram. The orientations of the lines in the polar plots correspond to the HOG orientations. Each orientation represents a bin in the histogram. Lengths (and colour – see 'Key') of the lines in the polar plots correspond to the size of the bin in question i.e. the number of edges (gradients) in the orientation range in question. Note how the appearance of coherent edges in the photos can also be seen in the corresponding grid of polar plots.

images when the shape and structure of the feature varies between the two images. An alternative way of stating this property is that HOG descriptors are somewhat insensitive to the exact distribution of edges, locally – as mentioned in Section 2.6.4. Specifically, local object appearance is 'characterized … by the distribution of local intensity gradients or edge directions, even without precise knowledge of corresponding gradient or edge positions'. In practice, this means that 'translations or rotations make little difference if they are much smaller than the local spatial or orientation bin size'.

HOG may struggle to provide a match between two images which are of the same type of object but whose shape and structure vary at a more global level. For the sake of illustration imagine two slightly different, but still similar, designs of Gothic window: (i) a Perpendicular window, with a flatter, more depressed design of arch; (ii) and a Geometric window with a taller, more elongated type of arch. Local level features might include the type of leadwork in small glazing panels. The differences in the arches (larger features) would be at a more global level, and could mean that (i) may not be matched with (ii).

Felzenszwalb *et al.* (2010) produced a modification of the original HOG descriptor, instead creating a 31D descriptor which incorporates both unsigned and signed HOGs, as opposed to 36 or 72 (2×2×18) dimensions for the unsigned only or signed only versions respectively of the original HOG implementation. The 31 dimensions are made up from 9+18+4 components, comprising the unsigned orientation bins, the signed orientation bins and normalisation via texture-energy features respectively. The rationale for the modified descriptor was two-fold. Firstly, by reducing dimensionality runtime could be reduced. Secondly, including signed orientation 'can help differentiate light-to-dark versus dark-to-light transitions within an image region' (MathWorks, 2017a), which could be useful when attempting to detect objects with subtle variations in texture.

See for an illustration of the 31D HOG descriptors for the templates used in this work. The figure shows a polar plot, for each grid cell in the descriptor, each polar plot representing the histogram for that grid cell. The orientations of the lines in the polar plots correspond to the HOG orientations, where each orientation represents a bin. The lengths (and colour) of the lines in the polar plots correspond to the size of the bin in question. Note from the figure how the orientation of coherent edges in the photos can often also be picked out in the corresponding gridded polar plot of the HOG descriptor.

Note that the original HOG implementation used a trained Support Vector Machine (SVM) classifier to detect objects using their descriptors. The work in this thesis does not use a classifier in that way, instead choosing to detect objects using a more recent variant of a HOG method, namely HOG-based template matching (see: Xiao, 2013; Zhang *et al.*, 2013b) and only using a classifier to, in effect, learn thresholds for the match-scores which result from that method.

2.6.6 Template Matching including HOG-Based

In computer vision, template matching is a simple, standard approach for detecting objects in images (Sonka *et al.*, 2014, p. 233). A template image (e.g. containing a single window object) is scanned across a test image (in the case of this work the test image would be a texture map image from a CityGML building model) to detect matching occurrences in the test image of the object contained in the template. Traditionally, good matches are identified as locations which result in large correlation values between the template and the corresponding test image windows, where the correlation is often achieved using Fast Fourier Transform or FFT (Sonka *et al.*, 2014, p. 234). Essentially, the method is comparing pixel intensities between a test image and template. A standard template matching method is found in the code of (Kroon, 2011) which uses either normalised cross correlation (NCC) or sum of squared differences (SSD) of the pixel intensities in the template and test image, followed by FFT to obtain correlation. The output is a match-score for each pixel in the test image, which indicates the likelihood that the pixel is the centre of an object which corresponds to that in the template. Such standard template matching approaches are inherently highly sensitive to even very small variations in rotation and scale though (Sonka *et al.*, 2014). Nonetheless, as an early part of the work here the Kroon method was trialled – see Chapter 4.

As mentioned in Section 2.6.3 HOG descriptors were chosen for (generic) object detection. Specifically, a recent alternative was chosen (see: Xiao, 2013; Zhang *et al.*, 2013b) which performs template matching using HOG descriptors, and employs the enhanced

HOG descriptor from Felzenswalb *et al* – see Section 2.6.5. The matching is achieved by convolving the descriptors of the template and the test image, at each of the grid cells of the descriptor. As with standard template matching the output is a match-score, in this case for each grid cell (each grid cell is 8×8 pixels in size).

Descriptors are centred on the centroid of the template or test image. For a template, the number of grid cells, both horizontally and vertically, is defined by Equation 2.1. In practice this means that even a pixel width or height which is divisible by 8 results in the loss of 1 grid cell all around the edge of the image, e.g. a 64-pixel width results in a grid 7 cells wide, not 8. Also, a minimum of either a 20-pixel width or a 20-pixel height are needed to form a descriptor. However, descriptors for test images are *not* cropped in the same way as those of templates, but as in Equation 2.2. The descriptors for test images are padded, as follows. The padded area is the region represented by the symmetrical difference of (1) the outcome of Equation 2.1 (with a test image as input) and (2) the result of Equation 2.2. The region that results will therefore comprise whole grid cells around the boundary of (1), filled with zero values for the descriptor contents.

$$cells_{template} = round\left(\frac{pixels_{template}}{8}\right) - 2$$

Equation 2.1

$$cells_{test\ image} = round\left(\frac{pixels_{test\ image}}{8}\right)$$

Equation 2.2

Note: the above equations calculate the number of columns or rows.

The HOG-based template matching method performs matching over an image pyramid, progressively reducing the size of the test image at each level of the pyramid. This overcomes one of the shortfalls of standard template matching, namely that the template needs to have been previously sized to the dimensions of instances of the object on the test image to optimise the matching (Sonka *et al.*, 2014, p. 233). A shortfall of the HOG-based template matching method is that the implementation does not allow comparison of match-scores achieved with different templates. For this reason, the method was extended in the work here by the creation of a new pipeline into which HOG-based template matching was placed, where the pipeline included the normalising of match-scores to allow such comparison. In addition, the non-maximum suppression approach used in the HOG-based template matching approach was replaced. Non-maximum suppression is used for ‘suppressing multiple responses’ (Fisher *et al.*, 2014, p. 189), which in the case of this work might be multiple possible window detections within the same region of the image, each of which has a match-score. The alteration was to take account of what were termed ‘partial matches’ whereby smaller but higher scoring candidate matches were suppressed, leaving slightly lower scoring but larger matches (see Chapter 5).

To determine the effectiveness of the method used here, the commonly used F-measure calculation was used to compare the locations of the rectangles from ground truth window and door locations with candidate matches. A micro-averaging as opposed to macro-averaging method was used to calculate precision and recall, based on the guidance of Sebastiani (2002), their text classification approach being equally relevant for computer vision. To determine if a candidate match was a true positive or a false positive, with respect to its position on a texture map image, the 'Bounding Box Evaluation' method from the well-known *PASCAL Visual Object Classes* (VOC) Challenge (Everingham *et al.*, 2010) was employed. That method applies a threshold of 50% to the ratio between the intersection and union of ground truth and candidate match. See Equation 2.3, Equation 2.4 and Equation 2.5 for the F-measure calculation. Precision here is the proportion of retrieved items that are relevant. Recall is the proportion of relevant items that are successfully identified.

$$precision = \frac{true\ positives}{true\ positives + false\ positives}$$ **Equation 2.3**

$$recall = \frac{true\ positives}{true\ positives + false\ negatives}$$ **Equation 2.4**

$$Fmeasure = 2\frac{precision \times recall}{precision\ + recall}$$ **Equation 2.5**

2.7 Machine Learning

Machine learning attempts to teach computers to 'learn from experience' by learning 'information directly from data without relying on a predetermined equation as a model', instead using algorithms which adapt to improve performance as 'the number of samples available for learning increases' (MathWorks, 2016d). The algorithms locate 'natural patterns in data' thus generating insights which might help make improved predictions and decisions (MathWorks, 2016d). They are increasingly used in areas such as object detection and recognition e.g. facial recognition, credit scoring, stock trading, medical diagnosis, energy forecasting, in manufacturing and for natural language processing (see: Alpaydin, 2014, p. 3; MathWorks, 2016d).

There are two types of machine learning algorithms: supervised learning, where either classification or regression techniques are commonly used; and unsupervised learning, where clustering is the most common approach (see: Fisher *et al.*, 2014, p. 279; MathWorks, 2016d; Sonka *et al.*, 2014, p. 406). Broadly, supervised learning should be used when a prediction, such as a value, is required; unsupervised learning should be used when an idea of the structure of data is required, including splitting the data up based on some *a posteriori* knowledge (MathWorks, 2016c). For this reason, supervised learning was used in this work to, in effect, learn thresholds for the match-scores resulting from HOG-based template matching. Meanwhile, unsupervised learning was used to find representative template examples for template cluster centres during the trialling of clustering HOG descriptors of templates.

2.7.1 Supervised Learning & Support Vector Machines (SVM)

Supervised, or classifier, learning is founded on the principle that, using learned knowledge in the form of a trained classifier, it is possible to predict (also known as 'to classify') the class of each of a set of observations (also known as 'examples'). In this work the classes are window and door templates of various architectural styles. Each observation is defined by a feature vector made up of a predictor value for every possible class. In this work features are the match-scores for each of the template classes, where the feature vector might be '0.3, 0.2, 0.1' with three template classes in total. The classifier is trained using a set of different observations, each formed of a similarly structured feature vector but where each observation in the training data is manually (hence the 'supervised' tag) assigned a class label before the classifier is trained. The training data is supposedly representative of positive classes and negative classes and is sufficiently discriminative between classes – this is where having large amounts of data can be key (MathWorks, 2016e). The classifier is essentially learning a 'set of patterns and classes' in order to attempt to predict the class of the observations which lack a class label (Sonka *et al.*, 2014, p. 396). Such initially classless observations are known as test data. Visualising the data, one might find it helpful to think of each observation as a row in a table, where each column is a feature and each cell then contains a predictor value. In that case training data would have an additional column which contains a class label for each observation, but the test data (before classification) would not. See Chapter 6 for how the feature vector used in this work was constructed.

Having large amounts of training data is often crucial for successful classification, generalising the classifier for new data as a result. However, the classifier learning approach used must also be inductive i.e. it should be nearly optimal for all feasible patterns and not just those in the training data (Sonka *et al.*, 2014, p. 396). Being able to distinguish between these different patterns is crucial for classification. Sonka *et al.* (2014) describe the concept of such separation as follows. The set of all possible patterns forms 'feature space' (see also Section 2.6). Within feature space the classes ideally tend to form clusters. These clusters are separated by a discrimination curve or a hyper-surface in multidimensions. In practice though, the separation is never perfect, and some objects will ‘always be misclassified’.

A number of supervised learning algorithms are available. Those which are known as classifiers include the previously mentioned SVM, in addition to naïve Bayes and nearest neighbour algorithms (MathWorks, 2016d). Common regression approaches include regression trees, which are a type of decision tree, and neural networks (MathWorks, 2016d). In the case of the problem posed in this work, classifiers, as opposed to regression techniques, are more suitable. The reasons for this are that the data can be split into specific classes and because a discrete rather than a continuous response is required (MathWorks, 2016a).

The choice of classifier can be a complex decision in itself, dependent on factors such as: the size and structure of the data; the desired runtime performance; and the type of insights required. Consequently, the best approach is to trial a few algorithms on the data before choosing (see: Kaehler and Bradski, 2016, p. 777; MathWorks, 2016d). The problem in this thesis requires fast runtime during classification (classification is also known as ‘testing’) due to the potentially large amount of, and size of, individual texture map images. Essentially, as image size increases so does the number of HOG descriptor 8×8-pixel grids, where each grid cell will have a feature vector requiring classification. Conversely, training needs to only be undertaken once. The time taken for training during this work is therefore not so crucial. A requirement for a fast runtime during testing, but flexibility over training time, means that the SVM or naïve Bayes algorithms may be best suited, while naïve Bayes can be suited to smaller training datasets (see: Kaehler and Bradski, 2016, p. 777; MathWorks, 2016b). Consequently, early experiments used SVM and naïve Bayes classifiers. The SVM gave the best results, so it was chosen as the classifer for this work.

Best practice within machine learning is that *separate* data should be used for each of:

(i) Training Data

Training data is used to ‘fit the model’.

(ii) Validation Data

Validation data is used to check the generalisation ability of the model by estimating the prediction error. Essentially, this set of data is used to tune the classifier model in order to optimise the results so that the best model is produced, in preparation for the final stage (testing).

(iii) Test Data

Test data is used to assess the generalisation error for the final model and should ideally be set aside and not used until the final stage of data analysis.

Note that the principle of separating validation and test data also applies to the production of statistical models which do *not* use machine learning.

(see: Alpaydin, 2014, pp. 37-41; Hastie *et al.*, 2009, p. 222)

To illustrate the above point, using test data for training, or test data for validation, would potentially bias the model toward the test data, reducing the model’s inductive ability and scalability to accurately predict on unseen data. Additionally, supervised learning should avoid, or at least minimise, ‘overfitting’. Overfitting is where the classifier is ‘so closely aligned to training datasets that it does not know how to respond to new situations’ i.e. how to respond to test data (MathWorks, 2016e). The standard approach to reducing overfitting is to use more training data (MathWorks, 2016e) which is, as already mentioned, a challenge in this work. Selecting the right features for the feature vector, transforming the features into new features and the use of hyperparameter tuning can be used to optimise the result (MathWorks, 2016b). Hyperparameters ‘control how a machine learning algorithm fits’ the trained classifier to the data (MathWorks, 2016b). Lastly, further optimisation might be achieved by augmenting training data to provide training data which has a more representative spread of data variations than might be expected in unseen test data. Chapter 6 covers in detail the optimisation steps during training in this work.

2.7.2 Unsupervised Learning including *k*-Means

As a reminder, supervised learning approaches generally use clustering of some form. Clustering techniques can be split into ‘hard’ and ‘soft’ clustering i.e. each data point belongs to only one cluster and each data point can belong to more than one cluster respectively (MathWorks, 2016a). Noting the comments in Section 2.6, i.e. that the objects in this study (windows or doors) can only have one class, hard clustering was chosen.

Hard clustering approaches include *k*-means, *k*-medoids, hierarchical clustering and the neural network-based self-organising map (MathWorks, 2016a). *k*-means partitions data into *k* number clusters, which are mutually exclusive, and where the membership of a cluster is determined by the distance from that point to the cluster centre (MathWorks, 2016a). *k*-medoids is similar but the cluster centres correspond to points in the data. Affinity propagation (Frey and Dueck, 2007) is a more recent but similar approach to *k*-medoids in that it finds cluster centres which are points in the data. Hierarchical clustering lends itself to visualising clusters which are nested. Relationships between the clusters in this work needed to be flat, not nested, where, for example, clustering on the template examples of the class ‘Norman style windows’ required only one level of cluster. Hierarchical clustering was therefore not chosen as an approach. Meanwhile, the self-organising map is suited to high-dimensional data (MathWorks, 2016a). In addition to being used as a regression approach for supervised learning, neural networks (and so-called deep learning) can be used for unsupervised learning (MathWorks, 2016d). However, they were not considered in this study due to the requirement for large amounts of training data (Liptak, 2005, p. 166).

Consequently, it was decided that *k*-means and affinity propagation would be trialled. In choosing *k*-means it was noted that the number of clusters needed to be known in advance (MathWorks, 2016d). The silhouette clustering evaluation criterion was therefore used prior to clustering to evaluate the optimal number of clusters (Rousseeuw, 1987). Affinity propagation includes a similar step to pre-determine the optimal cluster configuration.

2.8 Window & Door Detection

2.8.1 Window & Door Detection on Imagery

The identification of window and door objects on images of buildings is predominantly achieved through some combination of image segmentation and feature detection (Iman Zolanvari and Laefer, 2016). Neuhausen *et al.* (2016) expand upon that and summarise the approaches taken to image-based window detection (which would also apply for door detection) as falling into one of three categories, summarised as follows. (1) Grammar-based approaches where the challenges can be: a reliance on experts to generate the grammar; that architectural styles can vary by country and within a country; that the methods tend to only result in simple segmentation. (2) Pattern recognition approaches where the challenges can be: that window components other than frames tend to only possess small changes in pixel intensity, which can then lead to detection problems; and that they often rely on symmetry, regular placement and standardisation of design of objects on façades. (3) Supervised machine learning, which is seen as a better approach in that it doesn’t rely on the arrangement of components on a façade, although such methods still rely on the quality of the provided machine learning features and the often time-consuming creation of a training dataset. The method used in this work sits within (2) but attempts to obtain some of the benefits of (3), namely a non-reliance upon the nature of the arrangement of façade component objects. As a reminder, note that the methods used here do also include machine learning.

There is a rich seam of research that attempts to identify and label windows or doors on façade images. Table 2.1 and Table 2.2 summarise a selection of those methods, highlighting the datasets used and the detection performance obtained. As noted in the tables, for those methods which do not provide quantitative results, their inclusion is on the basis that: they were early works in the field, or which demonstrated particular novelty at the time; or they have been applied to a particular application area or type of dataset of note; or have a finding which is of relevance to this work. The methods in the tables are categorised with respect to whether they possess 'symmetry / regularity / standardisation constraints' or not. All the methods are grouped by this binary distinction, below. The methods use a variety of datasets and performance metrics. To allow direct comparison of methods and results, some key methods, which use the most popular reference dataset, are discussed in more detail, below. Some key findings for some of the other methods, of particular relevance for the work in this thesis, are also drawn-out, below. Finally, a summary of the evolution of detection performance on the most popular reference dataset and of the methodological approach for all the methods is provided.

Method	Symmetry / Regularity / Standardisation Constraint in Method & / or Architectural Style?	Dataset(s)	Window Detection Accuracy	Door Detection Accuracy	Notes
			('Pixel Accuracy' unless stated otherwise)		
Liu *et al.* (2017)	Yes	*ECP*, *eTRIMS*	93% (ECP)	91% (ECP)	
Horne *et al.* (2016)	No	Own dataset	N/A (does not detect windows)	N/A (no quantitative results)	Prosthetic vision application (personal navigation)
Kim *et al.* (2016a)	No	Own dataset	90%	N/A (does not detect doors)	
Morago *et al.* (2016)	Yes	*ZuBuD* (Shao *et al.*, 2003), Ceylan *et al.* (2014), Hauagge and Snavely (2012)	N/A (no quantitative results)	N/A (no quantitative results)	Image registration application
Wenzel and Förstner (2016)	No	*eTRIMS*	84-87%	N/A (does not detect doors)	
Wolff *et al.* (2016)	Yes	Own dataset (Google *StreetView*)	Results at façade-level	Results at façade-level	
Yu and Wang (2016)	No	*LM+SUN* (Tighe and Lazebnik, 2013), own dataset	0.54 (F-measure)	0.52 (F-measure)	Includes images captured from vehicles; 10,000s of images
Martinović *et al.* (2015)	No	*Monge2014* (Riemenschneider *et al.*, 2014)	N/A (no quantitative results)	N/A (no quantitative results)	Detects during SfM pipeline
Zhang *et al.* (2015)	Yes	Own dataset	N/A (no quantitative results)	N/A (no quantitative results)	Oblique aerial imagery
Cohen *et al.* (2014)	Yes	*ECP*, additional *ECP* website data, *eTRIMS*, *ZuBuD*	87% (ECP)	82% (ECP)	
Teboul *et al.* (2013)	No	*ECP*, own dataset	81% (ECP)	84% (ECP)	
Zhang *et al.* (2013a)	Yes	Own dataset	N/A (no quantitative results)	N/A (no quantitative results)	Preliminary work

Table 2.1: Window & Door Detection in Imagery – Summary of Symmetry / Regularity, Datasets used & Detection Accuracy for Other Methods (Table 1 of 2)
Note: table is sorted by year then first author surname from 'Method'. See Section 2.12 for details of the '*ECP*' and '*eTRIMS*' datasets. See also Table 2.2.

Method	Symmetry / Regularity / Standardisation Constraint in Method & / or Architectural Style?	Dataset(s)	Window Detection Accuracy	Door Detection Accuracy	Notes
			('Pixel Accuracy' unless stated otherwise)		
Riemenschneider *et al.* (2012)	Yes	Own dataset (*Graz50*), *Paris2010* (Teboul *et al.*, 2010)	60%	41%	
Macák and Drbohlav (2011)	No	*eTRIMS*	0.58-0.7 (F-measure)	N/A (does not detect doors)	Results improve with rectified images
Recky and Leberl (2010)	Yes	Own dataset	94% (% windows detected)	N/A (does not detect doors)	
Teboul *et al.* (2010)	Yes	*ECP* (own dataset)	81% (ECP)	71% (ECP)	
Ripperda and Brenner (2009)	Yes	Own dataset	N/A (no quantitative results)	N/A (no quantitative results)	Combines with manual façade reconstruction
Drauschke and Förstner (2008)	No	*eTRIMS*	41% for windows (error rate) 10% for window panes (error rate)	N/A (does not detect doors)	Principle focus of output is window pane detection
Korah and Rasmussen (2008)	Yes	*PSU NRT* (Lee, 2007)	N/A (no quantitative results)	N/A (no quantitative results)	Detection of occluded windows
Ali *et al.* (2007)	No	*TSG-20* (Joanneum, 2004a), *TSG-60* (Joanneum, 2004b), *ZuBuD* (Shao *et al.*, 2003), own dataset	30-57% (positive true accuracy)	N/A (does not detect doors)	
Müller *et al.* (2007)	Yes	Own dataset	N/A (no quantitative results)	N/A (no quantitative results)	Procedural modelling step late in pipeline.
Van Gool *et al.* (2007)	Yes	Own dataset	N/A (no quantitative results)	N/A (no quantitative results)	Procedural modelling step late in pipeline. Related to Müller *et al.* (2007)
Alegre and Dellaert (2004)	Yes	Own dataset	N/A (no quantitative results)	N/A (no quantitative results)	Early example in the field

Table 2.2: Window & Door Detection in Imagery – Summary of Symmetry / Regularity, Datasets used & Detection Accuracy for Other Methods (Table 2 of 2)
Note: table is sorted by year then first author surname from 'Method'. See Section 2.12 for details of the '*ECP*' and '*eTRIMS*' datasets. See also Table 2.1.

Methods to match whole windows include: Alegre and Dellaert (2004); Cohen *et al.* (2014); Korah and Rasmussen (2008); Morago *et al.* (2016); Müller *et al.* (2007); Recky and Leberl (2010); Riemenschneider *et al.* (2012); Ripperda and Brenner (2009); Teboul *et al.* (2010); Van Gool *et al.* (2007); Wolff *et al.* (2016); Zhang *et al.* (2013a); Zhang *et al.* (2015).

Of these methods, those which provide results for what is the most commonly used reference façade dataset, i.e. the *ECP Facades Database* of Teboul *et al.* (2010), all use a segmentation approach, and carry-out detection as follows. Teboul *et al.* (2010) use procedural and shape grammars and a random forest classifier with features formed from patches centred on pixels, obtaining 81% and 71% pixel-accuracy for window and door detection respectively. Random forest classifiers are formed from multiple decision trees, which each use a random subset of training observations, combining predictions from each decision tree to improve accuracy (Alpaydin, 2014, p. 235). Cohen *et al.* (2014) meanwhile use *TextonBoost* (Shotton *et al.*, 2006) features in a boosted decision tree, obtaining 81% and 84% pixel-accuracy. The current state of the art in window and door detection on images is, to the best of this author’s knowledge, Liu *et al.* (2017). Liu *et al* use a 'feature map' to form features, and a combination of faster recursive convolutional neural networks (R-CNN), deep CNN and region proposals (bounding boxes) for detection, obtaining 93% and 91% pixel-accuracy. See Section 2.12 for background to the *ECP Facades Database* and other key reference datasets.

However, none of the abovementioned window and door detection methods label the architectural style of the identified object. And as Neuhausen *et al.* (2016) observed generally, the methods also require a symmetrical layout of objects on façades, or the regular spacing or standardisation of the design of those objects.

While the following methods do not require symmetry, regularity of object distribution or standardisation of object design on a façade in order to detect and label windows or doors, they do not attempt to identify architectural style of the objects either: Ali *et al.* (2007); Horne *et al.* (2016); Kim *et al.* (2016a); Martinović *et al.* (2015); Teboul *et al.* (2013); Wenzel and Förstner (2016); Yu and Wang (2016). Of those methods, only Teboul *et al.* (2013) provide results for the *ECP Facades Database*, doing so through use of a random forest classifier with reinforcement learning, with features comprising 'red, green and blue' (RGB) patches – they obtained 81% and 84% pixel-accuracy for window and door detection respectively. Reinforcement learning aims to determine the best sequence of actions to achieve a goal by using rewards and repeated attempts to achieve that goal (Alpaydin, 2014, p. 13). Drauschke and Förstner (2008) and Macák and Drbohlav (2011) also do not require symmetry but only attempt to match component parts of windows, rather than whole windows.

Incidentally, note from Table 2.1 and Table 2.2 how Zhang *et al.* (2015) used oblique imagery. This contrasts with the other methods, which tend to use data captured at street-level. Meanwhile, Macák and Drbohlav (2011) found that their results improved when rectified, as opposed to unrectified, imagery was used. Clearly, the pose within imagery used in this work will therefore need to be a consideration. These points are of interest to the Ordnance Survey, noting that application of a window and door detection method to imagery for the whole of Great Britain will likely necessitate automated capture of imagery. In that respect note that Yu and Wang (2016) included images captured from vehicles – see below for more discussion on their method and results.

To reiterate, the work described in this thesis faced a challenge with respect to the availability of data. This problem was caused, in part, by the highly variable nature of the designs for different windows and doors encountered in the real world. See Chapter 3 for more on this challenge. As such, the reduced performance of the method from Yu and Wang (2016), compared to other methods of the last few years in Table 2.1, is, arguably, a function of the highly variable nature of the designs of windows and doors likely to be encountered in the 'tens of thousands' of images used in their work. Indeed, Yu and Wang chose not to label all window and door instances in their training data and suggest that this may explain the lower performance in their results when compared to other classes of object. Incidentally, note that their labelling is not publicly available. Contrast the size of the dataset used by Yu and Wang with the hundreds of images in each of the popular reference datasets, such as the *ECP Facade Database* and the *eTRIMS Image Database* (Korč and Förstner, 2009). Moreover, the architectural styles within those popular reference datasets tend to belong to the equivalent of the Classical tradition. Furthermore, other reference datasets used by the methods in Table 2.1 and Table 2.2 tend to contain images of buildings from the Classical tradition, or of modern office blocks or apartment blocks. Buildings of the Classical tradition, or of such a block-type, tend to contain windows and doors with more standardised designs.

In addition, when different datasets are used comparing the results from any of the methods in Table 2.1 and Table 2.2 can only be indicative. The use of differing metrics amongst those methods also prevents direct comparison of their results. For example, direct comparison of Yu and Wang's results with those of most of the other methods of the past few years is not possible, on the basis that the former use F-measure and the latter use pixel accuracy.

Nonetheless, note from Table 2.1 and Table 2.2 how, in the course of a little over 5 years, pixel accuracy for window and door detection for those methods which used the *ECP Facade Database* has improved from 70-80% to 90+%. For all of the methods outlined in the tables, the evolution of the methodological approaches used since the earliest work, i.e. that of Alegre and Dellaert (2004), to the present-day state-of-the-art approach from Liu *et al* has been typified by:

- An early emergence of machine learning to complement computer vision.
- A certain movement away from the use of grammars (though their use persists) and away from the spatial constraints that they can impose on the distribution of window and door objects on façades.
- The very recent emergence of methods which are beginning to embrace deep learning (noting once again the requirement for substantial amounts of data for such approaches).

Note that none of the above methods attempt to create new labelled geometry for the detected objects on a 3D building model (although Müller *et al.* (2007) do attempt to procedurally model buildings).

2.8.2 Window & Door Detection during Reconstruction

Several building reconstruction methods attempt to identify and label window and door objects during the reconstruction process. Those which do so by segmenting a point cloud include the work of Aijazi *et al.* (2016), Beetz *et al.* (2016), Cohen *et al.* (2016), Dimitrov and Golparvar-Fard (2015), Koch *et al.* (2016) and Schmittwilken and Plümer (2010) while Gadde *et al.* (in press) and Jampani *et al.* (2015) generate segmentation of point cloud and of images. Bassier *et al.* (2016) attempt to label the objects in a mesh.

Affara *et al.* (2016) and Frommholz *et al.* (2015) attempt to identify the geometry of windows or doors and then label them during the reconstruction of a 3D building model from imagery, while Chu *et al.* (2016) and Loch-Dehbi and Plümer (2015) attempt to do the same from a combination of imagery and ground plans. Arikan *et al.* (2013); Li *et al.* (2011), Dehbi *et al.* (2016b), Iman Zolanvari and Laefer (2016), Li *et al.* (2017), Wang *et al.* (2015), Yu and Wang (2016) and Zhou *et al.* (2016) undertake to do the same during building reconstruction of a 3D building model from point clouds. Dore and Murphy (2014) use a combination of point clouds and imagery. Lastly Sugihara and Shen (2016) attempt to detect windows during a reconstruction process which uses a combination of ground plans and building structure priors.

However, once again, none of the reconstruction methods outlined above identify the architectural style of the detected objects. Moreover, aside from a few exceptions, such as Ley and Hellwich (2016), all of the above methods are only employed on buildings whose architectural styles are typified by a comparatively more symmetrical or more regular distribution or standardisation of design of the objects on the façades.

2.9 Architectural Style Detection

2.9.1 Architectural Style Detection for Whole Building

Strobbe *et al.* (2016) attempt to automatically determine architectural style from floor plans, but only do so for the whole building and for the modern Portuguese *Malagueira* style. They also provide useful background to the variety of approaches for automatic style derivation. Meanwhile, Ippolito and Attenni (2015) attempt to identify the architectural style of pre-Renaissance buildings, but for the Italian Roman style, only for the whole building, and on point clouds, not 3D building models. Chu and Tsai (2012), Obeso *et al.* (2016a) and Zhang *et al.* (2010), also attempt to detect older less symmetrical building types or architectural styles with less standardised design of architectural components, this time in images of Gothic, Korean Josean dynasty and Islamic, Mexican and historic Chinese buildings respectively, but do so for the whole building not for individual objects on the building. Xu *et al.* (2014b) attempt to identify probabilities that different areas of images from a variety of world locations may be of a particular architectural style, but again stop short of doing so for consistently discrete objects such as windows and doors. Meixner *et al.* (2011) do attempt to automatically infer architectural style of a building, but only identify the style of the roof, and do so for point clouds. While Henn *et al.* (2012) do attempt to determine the style for existing 3D building models, they only do so for one style, the Classical Wilhelminian style. Llamas *et al.* (2017) identify the type of architectural feature, such as bell tower, dome, flying buttress and gargoyle, but not the architectural style. Lastly, although they do not attempt to judge the architectural style, Biljecki and Sindram (2017) automatically determine the construction dates for buildings using 3D building models, using machine learning, but only on buildings post-1860. That said, to the best of this author's knowledge it is the only method which attempts to automatically determine the construction dates from 3D building models.

2.9.2 Architectural Style Detection for Building Objects

Agudo *et al.* (2016) assign a 'transformation sequence' to a 9th-10th century medieval church in Spain, labelling individual objects on a 3D building model created from photogrammetry with architectural styles using the categorisation method of Caballero Zoreda (2010), but the style is assigned manually. Müller *et al.* (2015) do identify individual windows and doors in images and assign them a 'style', but this label is more of a structural description for the object rather than an architectural style. While Doersch *et al.* (2015) do not label objects of an architectural style they do present an interesting proposition: the clustering of geo-referenced images based on the prevalence of objects of the same visual appearance, resulting in an identification of the transitions between building types, including style transitions, across neighbourhoods of Paris. Gadde *et al.* (2016) suggest another interesting approach: to learn a grammar for an architectural style from a training set. Lee *et al.* (2015) and Mathias *et al.* (2016) do detect windows of a particular architectural style on images of European buildings, but again the styles are all typified by symmetry or regularity or standardisation of design of architectural components. Shalunts *et al.* (2011) identify the architectural style of the individual windows and doors but their methods require some user intervention during processing. They do employ their methods on Gothic buildings, but the method still relies on symmetry.

2.10 HOG for Building & Object Detection in Imagery

While HOG is a highly cited method, commonly used for its original purpose of pedestrian detection (Dalal and Triggs, 2005), and referenced by thousands of other articles, its use in the field of buildings is less common. HOG has been used to identify buildings in aerial imagery, such as: Ok *et al.* (2012) who use HOG to infer edges, including in buildings, in overlapping stereo images; and Benedek *et al.* (2015) who use HOG to identify storage tankers and sports ground complexes. With respect to the use of HOG to detect objects within building façade imagery, Lia *et al.* (2016) use HOG as part of a feature vector, but to detect entire buildings, while the aforementioned methods of Lee *et al.* (2015), Morago *et al.* (2016) and Zhang *et al.* (2010) use Pyramid of HOG (PHOG), regional HOG and whitened HOG space respectively. The HOG-based template matching approach (Xiao, 2013; Zhang *et al.*, 2013b) which was used in this work was devised to address the problem of image completion. To the best of this author's knowledge, that HOG-based template matching approach has not been used for building or building component object detection.

2.11 Window & Door Detection in Existing 3D Building Models

While, in theory, image-based feature-detection methods could be employed on existing 3D building models (using the buildings' texture map images), there seems to be a surprising lack of research attempting to do so. Most of the effort appears to be focussed on identifying missing windows or doors using SfM, on pre-existing point clouds or on façade images. Rook *et al.* (2016) attempt to semantically enrich existing CityGML models but only label CityGML *GroundSurface*s, *WallSurfaces* and *RoofSurface*s and not windows or doors. The detection of missing windows and doors on existing 3D building models has been attempted by Demir *et al.* (2016) and Diakité *et al.* (2014), but rather than using texture map images to do so they rely on either the inherent topology of the model or the pre-existence of labelled geometry for similar window or door objects in the model.

To the best of this author's knowledge no research has been carried out which attempts to replace missing windows and doors in 3D building models with labelled geometry, irrespective of the presence of symmetry, regularity of distribution or standardisation of the design of objects on the buildings' façades.

2.12 Reference Datasets & Reference Implementations

While detailed AEC 3D building models have existed in the world of CAD for some decades, the arrival of *Web 2.0* has led to the advent of building models generated by members of the public. Aside from the *3D Warehouse*, which is free, *TurboSquid* provides commercially sourced models for a fee.

A number of reference datasets marked up with windows and doors do exist, such as the *eTRIMS Image Database* (Korč and Förstner, 2009), the *ECP Façades Database* (Teboul, 2010) and the *CMP Façade Database* (Tyleček, 2013). However, they are for façade images rather than texture maps on 3D building models. Moreover, the datasets do not differentiate the architectural style of the objects. Xu *et al.* (2014b) and Obeso *et al.* (2016b) provide façade image datasets marked up by architectural style but only for the entire building, not individual objects. While, for example, Niemeyer *et al.* (2014) and Rottensteiner *et al.* (2013) provide ISPRS benchmark datasets of buildings, the datasets are point clouds (though they contain ground truth polygons for roof shape) and the ground truth is only at the resolution of building, façade or roof and not for objects on the building. Consequently, to the best of this author's knowledge no benchmark building dataset exists with ground truth mark-up of the boundaries and architectural styles of windows and doors. As a result, all 3D building models, and templates, were collected by the author for this work – see Chapter 3 for details of those datasets.

To the best of this author's knowledge, the only publicly and freely available application for window or door detection is that provided by Teboul *et al.* (2013). However, following discussions with the principle author of the work, it was determined that the provided application is not that used to obtain the results in the paper.

2.13 Summary

In this chapter a background to 3D city and building models has been provided, their representation and reconstruction has been outlined and the methods available for computer and machine vision have been outlined. Existing methods for window, door and architectural style detection have been summarised, as has the availability of reference datasets. It is noted that there is no existing research which detects missing windows or doors on 3D building models irrespective of the presence of symmetry or regularity of the distribution or standardisation of the design of the windows or doors. This is especially evident for older buildings i.e. those which are more likely to be an amalgam of styles. Consequently, the following methods / standards were chosen for this work: CityGML for the 3D city models; C20, Georgian-Regency, Norman and Gothic styles; SIFT for an image matching trial; HOG-based template matching for window and door detection; *k*-means and affinity propagation to trial the clustering of HOG descriptors of templates; and an SVM for, in effect, learning match-score thresholds to filter out poor candidate matches. It is also noted that the application of HOG-based template matching to window and door detection has not previously been attempted, nor has a pipeline been placed around the method, as created here, to compare the match-scores between different templates. Lastly, attention is drawn to the use of heuristics, including those based on real-world units from the 3D building models, to filter candidate matches. Chapter 3 now describes the data used and the challenges associated with its collection. Chapters 5 and 6 then detail the newly created methods used in this work.

Chapter 3

Data & Data Challenges

3.1 Introduction

As mentioned in Chapter 2, no suitable reference datasets were available for this work (aside from a small trial conducted on the *ECP Façades Database* images – see Chapter 7). To reiterate, suitable data comprised: façade images marked up with ground truth windows and doors of the architectural styles used (i.e. the data used in Chapter 4, initially in Chapter 5 and as training images in Chapter 6); 3D building models marked up in the same fashion (i.e. the data used in Chapter 5 and Appendix D); and templates of the styles in this work (i.e. the data used in Chapters 4, 5 and 6 and Appendix D). Consequently, suitable image data needed to be collected, which was achieved primarily using internet searches, or created.

To ensure the integrity of the architectural history component of the work undertaken in this thesis, the collection of data frequently required significant cross-checking with reference literature, including for the confirmation of the architectural style of buildings and, in many cases, of individual windows and doors. The assignment of style to a building or to a window or door can be contentious though, even among the experts in the field. Furthermore, many buildings in Great Britain do not possess a formal categorisation of architectural style. Clearly then, determining the style of a building might present a challenge!

The time taken to collect data was further compounded by a dearth of actual buildings of particular styles, and by a lack of available imagery for some styles and for buildings. The following restrictions further reduced the availability of suitable data, and meant that many images had to be rejected: (1) repetition of the same view of a building in many online photo libraries; (2) the requirement not to mix the buildings used in training and testing; (3) and the trade-off between (a) attempting to increase the amount of training data in an attempt to reduce overfitting and (b) the impact of (2).

For these reasons, this chapter describes both the data collected and created and expands upon the abovementioned challenges. The chapter also outlines the potential consequences of the data shortages, including during validation. See Appendix C for details on the *SketchUp* to CityGML conversion.

3.2 Data – Potential Sources, Formats & Cataloguing

Image data (including templates) was generally collected from internet image repositories such as *Flickr* and *Geograph* (Geograph Project, 2017), and Google web searches. All of the 3D building models were from the Trimble *3D Warehouse* and as such are stored there in Trimble *SketchUp* format. The Trimble *3D Warehouse* includes a limited model-search facility, which searches for words tagged or used as a title or description for a model. Appendix C contains more information on the data collection strategies, data formats and cataloguing.

3.3 Data – Shortages & Challenges

The work undertaken for this thesis presented a number of challenges regarding the scarcity of suitable data. Broadly, the reasons for those shortages can be summarised as: a shortage of suitable, available data compounded by the paucity of definitive documentation of the architectural styles for some buildings, by the requirement to avoid mixing training and testing data, by the need to find templates and training data with representative designs and by the time required to pre-process the data. These are explained in more detail below, beginning with the requirement to confidently judge architectural style for the windows and doors in the images and templates, and for the 3D building models.

3.3.1 Judging Architectural Style

It was, important to this work to use reference literature to pre-assign architectural style to a building or to an individual window or door, noting that image and 3D building model repository and web users in general may not necessarily tag the architectural style correctly, or at all. The assignment of a style by an authority can also be challenging and contested because an amount of subjectivity is required to assign the style. In some cases the use of a term for a style at all can be contentious e.g. with reference to the uncertainty regarding categorisation of the Gothic styles – see Hart (2010). Nonetheless, in that one could at least use a consistent approach, a small, discrete grouping of recognised reference authorities was used for judging architectural style. Generally, the well-respected and popular (see e.g. Hart, 2010, p. 2) works of Pevsner were used, complemented with the reference guides from Great Britain's listed building authorities Historic England (2017), Historic Environment Scotland (2017) and Cadw (2017), noting that much of their work in turn cites Pevsner. When no style designation was available from these sources, reference works from the Royal Institute of British Architects (RIBA) were used. For Norman buildings *The Corpus of Romanesque Sculpture in Britain & Ireland* (King's College London, 2016) was used to check the style of buildings, windows and doors. No equivalent resource was found for the other styles used in this work. The literature used does not contain a style assignment for every building in Great Britain. It was therefore necessary to reject some potential buildings, windows and doors. While it would have been possible to attempt to judge style if a date of construction was available, where that date only fell in one style category, and if the building, window or door contained archetypal designs, such judgement was not attempted. The rationale for avoiding such a step was that the author of this work is not an architectural historian.

Consequently, the identification of architectural style, especially for the trial of the clustering of HOG descriptors for hundreds of individual templates (see Section 3.6 and Appendix D), was a labour-intensive task. The collection process, not just for that trial, was initially compounded by the need to sometimes investigate which building an image contained (if not clear from the internet description). When searching for templates it was also often necessary to determine which actual window or door the image contained, noting the comments in Chapter 2 that older buildings have often had some replacement windows or doors of later styles. Lastly, many of the images returned from searches were in fact of an unsuitable 'neo' or revival style. For these reasons a large number of potential images, templates and 3D building models had to be excluded from the dataset.

3.3.2 Quantity of Potential Data for each Style

Aside from The Office of the Deputy Prime Minister (2001) and IHBC (2016), which cover housing only and assign all buildings constructed pre-1850 into one date band, to the best of this author's knowledge there is no aggregated public record of the construction date, or even date range, for all buildings in Great Britain. Even for buildings which are 'listed' by the heritage preservation bodies, no comprehensive record of construction date is available – see Appendix C for a discussion on this point.

With respect to style classification, the abovementioned database from King's College London (2016) holds a record for all of the 2,400 buildings in Great Britain (98% of which are in England) which contain Norman sculptural features, but it can often be the case that these buildings have been rebuilt in a later style and that only small Norman features, which are not always windows or doors, remain. For the other styles used, again to the best of this author's knowledge, no similar record exists. Despite efforts by the likes of Historic England (2017), Historic Environment Scotland (2017) and Cadw (2017) in Great Britain, it has been determined during the work conducted for this thesis that, in general, the older the building the more likely it is to have been demolished, usually before such protections were in place.

Nonetheless, a very approximate indication of the availability of potential images and templates by style can be obtained by running *Flickr* searches for (1) 'Norman architecture England', (2) 'Gothic architecture England', (3) 'Georgian architecture England OR Regency architecture England' and (4) 'Modernism architecture England OR Neo-vernacular architecture England' – 42,000, 37,000, 23,000 and 4,000 results respectively were returned. A search using the terms (1), (3) and (4) on the Trimble *3D Warehouse* produced 8, 12 and 20 results respectively. Note that, aside from the image matching trial, no Gothic 3D building models were used, only Norman ones that contain some added Gothic features. However, using these search terms for 3D building models these terms proved too specific for finding potential models. Splitting the search phrases and using additional terms yielded more results, though rarely more than a few dozen potentially suitable buildings per style. Many of these potential models needed to be rejected for the reasons given above and below.

With respect to the results of image searches, the same building, or even the same architectural component, has often been photographed multiple times. For example, while a *Flickr* search for 'Kilpeck church' (3D building model BM_N_1, as used in this work) will yield almost 3,000 results, the same window or door appears again and again in the images retrieved. In general terms, reducing the number of duplicate instances of architectural components (i.e. the same window or door on the same building) used during training was an important part of the approach in this work, in order to reduce the problems of overfitting described in Chapter 2. The same was also true for the clustering trial – again see Chapter 2 for background here. One duplicate of the same window or door instance was therefore permitted in the training images, and in the templates used for the clustering trail.

Images were also rejected if they did not contain windows or doors, if the windows or doors were occluded, or if the image was of insufficient quality. Here, a low-quality image is one that is too small, or lacking sufficient sharpness

Particular difficulty was encountered when searching for images of buildings of the Modernism tradition, where it was concluded that such buildings are not only less photographed but also less tagged by users. Such a paucity of photographs may be due to the tradition being less researched, broader in scope and more difficult to judge style-wise (see Chapter 2). Compared to the other styles, a more manual search of a larger number of images using broader search terms was therefore required, which took more time.

In Chapter 2 it was mentioned that there are a wide variety of potential designs for windows and doors for some of the styles – this presented a particular challenge for finding templates and training images (façade images) which covered the spread of design variations encountered in the test images. Consequently, the hand-crafting of images and templates was sometimes required (which took time). The exceptions to this were the templates used for clustering, and the training images, both of which were used un-edited because it was desirable for the machine learning to attempt to identify real-world patterns which might not be readily discernible to the naked eye.

Through the use of multiple search terms on the Trimble *3D Warehouse*, it was determined that suitable Georgian-Regency 3D building models were approximately an order of magnitude more prevalent compared to suitable Norman 3D building models. Some potential Norman 3D building models were excluded due to a lack of authentically Norman windows and doors, the originals having been replaced. The paucity of suitable 3D building models meant that only five suitable Norman 3D building models could be found. While more Georgian-Regency 3D building models were available a large number had to be rejected for the reasons outlined above, meaning seven were used in total. Lastly, as with searching for imagery, finding suitable C20 3D building models of the Modernism tradition or neo-vernacular style proved particularly labour intensive. Due to the broad scope of the C20 style, and the use of a wide range of search terms, 150 potentially suitable 3D building models were obtained. All but 12 were rejected, for the reasons covered above.

Finally, the pre-processing of the data could also be labour intensive, taking, for example, approximately half a day to pre-process a 3D building model – see Section 3.8 for more on the pre-processing conducted.

3.3.3 Data Gathering for State-of-the-Art Methods

By way of a reminder, the current state-of-the-art window and door detection method, on façade images, not 3D building models, is Liu *et al.* (2017). In order to obtain a strong performance Liu *et al* used a deep-learning pipeline of pre-trained 'network layers' which used 1.3M images for training (Simonyan and Zisserman, 2014) from the publicly available *ImageNet* (Deng *et al.*, 2009). However, due to a lack of the necessary architectural style class granularity in *ImageNet*, a pre-trained network would not be suitable for this work. Nor, to the best of this author's knowledge, is such class granularity available, in any similar large public datasets. Consequently, such datasets, or pre-trained networks, were not used for this study.

3.3.4 Data Shortages – Consequences for Validation

To reiterate, using a separate test and validation dataset is best practice when producing statistical models, with or without machine learning. The core of the work described in this thesis was essentially to produce a statistical model which could detect or classify objects. However, shortages of suitable data meant validation and test data were the same in the following cases. (i) For HOG-based template matching without an SVM, the match-score thresholds for the Norman 3D building models were obtained using the test data (see Chapter 5). (ii) In addition, test data was used for validation of the HOG-based template matching approach with an SVM (see Chapter 6). Some 3D building models which were ultimately rejected from the dataset due to the prevalence of very low-quality texture map images were used in the early stages of validation.

While the above mixing of validation and test data was not ideal, and biased the models produced to the test data, the shortage of suitable available data made doing so essential.

3.4 Data – Façade Images including Training Images

In this work 'façade images' are photographs of building façades which are not a texture map image from a 3D building model. Four façade images were obtained from a combination of *Flickr*, *Geograph* and *Wikipedia* searches. The façade images were used to trial the standard template matching approach and for initial validation of the HOG-based template matching approach, hence only four were used – see Appendix D and Chapter 5 respectively for the methods used. Only four were obtained, on the basis that they would only be used for trials. The early experiments in question did not attempt to detect the architectural style, though the buildings, windows and doors in the façade images were all Georgian-Regency style or similar. The exception was the Palace of Westminster, which is Gothic Revival (see: Historic England, 2017; Pevsner and Sambrook, 2010). See Figure 3.1 for an illustration of the façade images used.

Façade images were also used as training images, where 20, 24 and 25 individual façade images were used for training for the C20, Georgian-Regency and Norman / Gothic styles respectively. This translated to 20, 23 and 22 unique buildings respectively, noting the comments in Section 3.3 regarding an allowance for some duplication of instances. See Section 3.7 for more detail on the number of instances of windows and doors used in the training data.

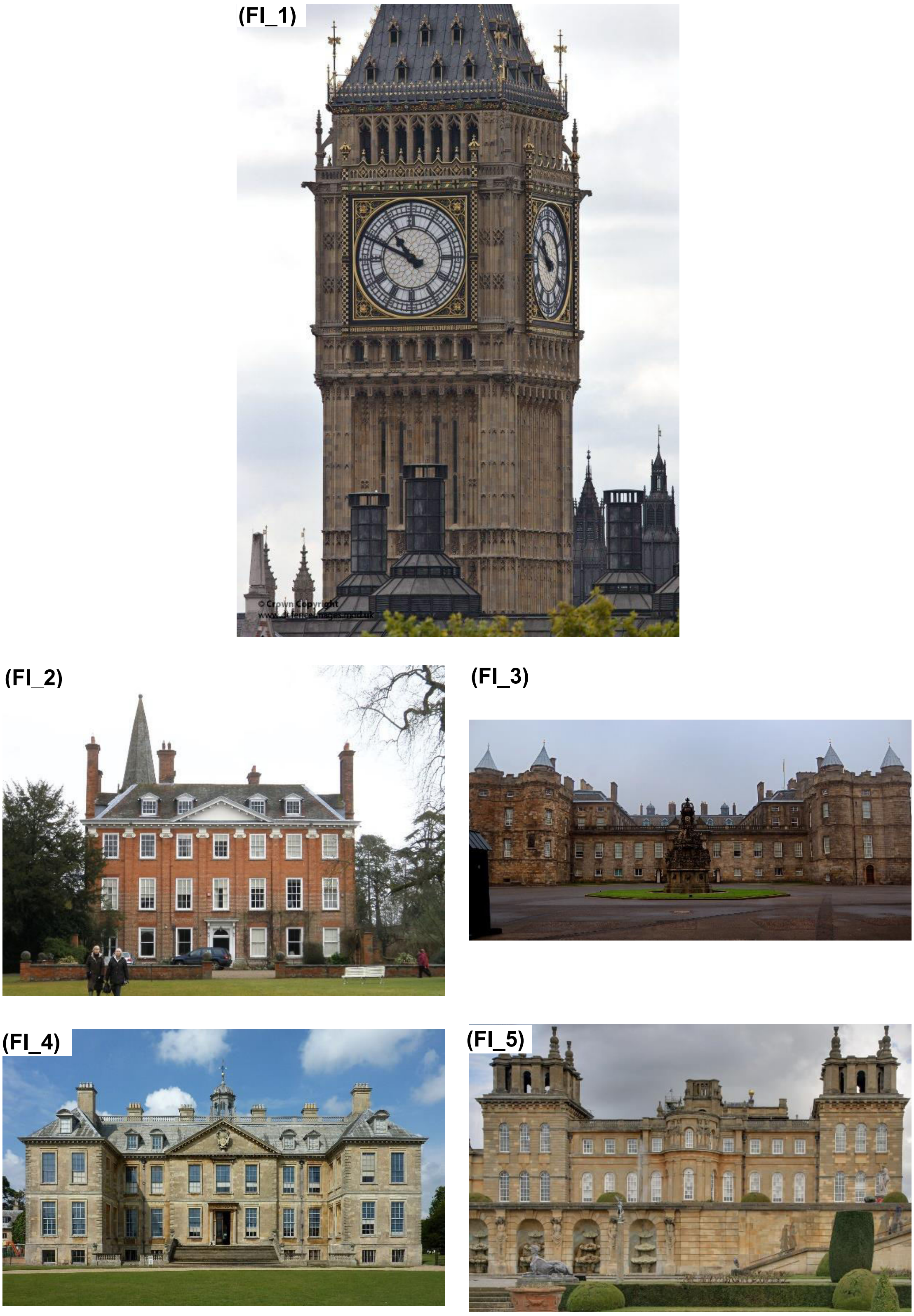


Figure 3.1: Façade Images
(*FI_1*) Palace of Westminster, London, built 1835-60. (*FI_2*) Welford Park, Berkshire, built late 17th century. (*FI_3*) Palace of Holyroodhouse, Edinburgh, built 1671-8. (*FI_4*) Belton House, Lincolnshire, built early 18th century. (*FI_5*) Blenheim Palace, Oxfordshire, built early and mid-18th and earlier 20th centuries. Construction date sources: Historic England (2017) and Historic Environment Scotland (2017).

3.5 Data – 3D Building Models including Texture Map Images

Figure 3.2 shows a 3D building model used in the image matching (SIFT) trial. The model in question was only used in this trial, therefore no further details on it are provided here. Figure 3.3, Figure 3.4 and Figure 3.5 illustrate the C20, Georgian-Regency and Norman style 3D building models, respectively. Table 3.1, Table 3.2 and Table 3.3 detail the C20, Georgian-Regency and Norman style 3D building models respectively. See Section 3.7 for the ground-truthing approach used. Note the variety of complexity of the building models, where *gml:LinearRings* represent a polygon face in a model (see Chapter 2). Table 3.4 shows the total number of texture map images, *gml:LinearRing*s, ground truths marked up and ground truths rendered for the 24 3D building models i.e. 12, 7 and 5 3D building models for C20, Georgian-Regency and Norman styles respectively.

Note that '*rendered*' means the result of the *rendering* process, which here is used to describe process of repeating windows or door candidate matches across the *gml:LinearRing*s of the 3D building model. Bear in mind that a candidate match on a texture map image could be repeated both within a *gml:LinearRing* if texture wrapping is used (see Chapter 2) and across different *gml:LinearRing*s if the texture map image is used on more than one *gml:LinearRing* (see Chapter 5 for more detail on the implementation of rendering).

Note also that while the texture map images used were generally the originals from the Trimble *3D Warehouse* 3D model, 19 (4%) of the texture map images were replaced because they were of very low quality. See Section 3.8 for more background on the process of replacement. Note that some of the 3D building models used already had some form of basic window or door geometry – 34 windows and 13 doors, or 5% of all window or door instances. However, applying the methods described in this thesis to those 3D building models was still seen as a valid test for the methods to identify similar windows or doors on the texture map images for other 3D building models where the window or door geometry was initially absent.

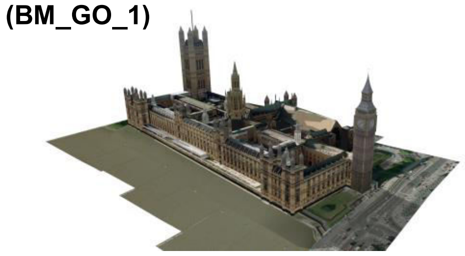


Figure 3.2: Palace of Westminster 3D Building Model (Gothic Revival)
Note that this 3D building model is only used in the image-matching trial (Chapter 4).

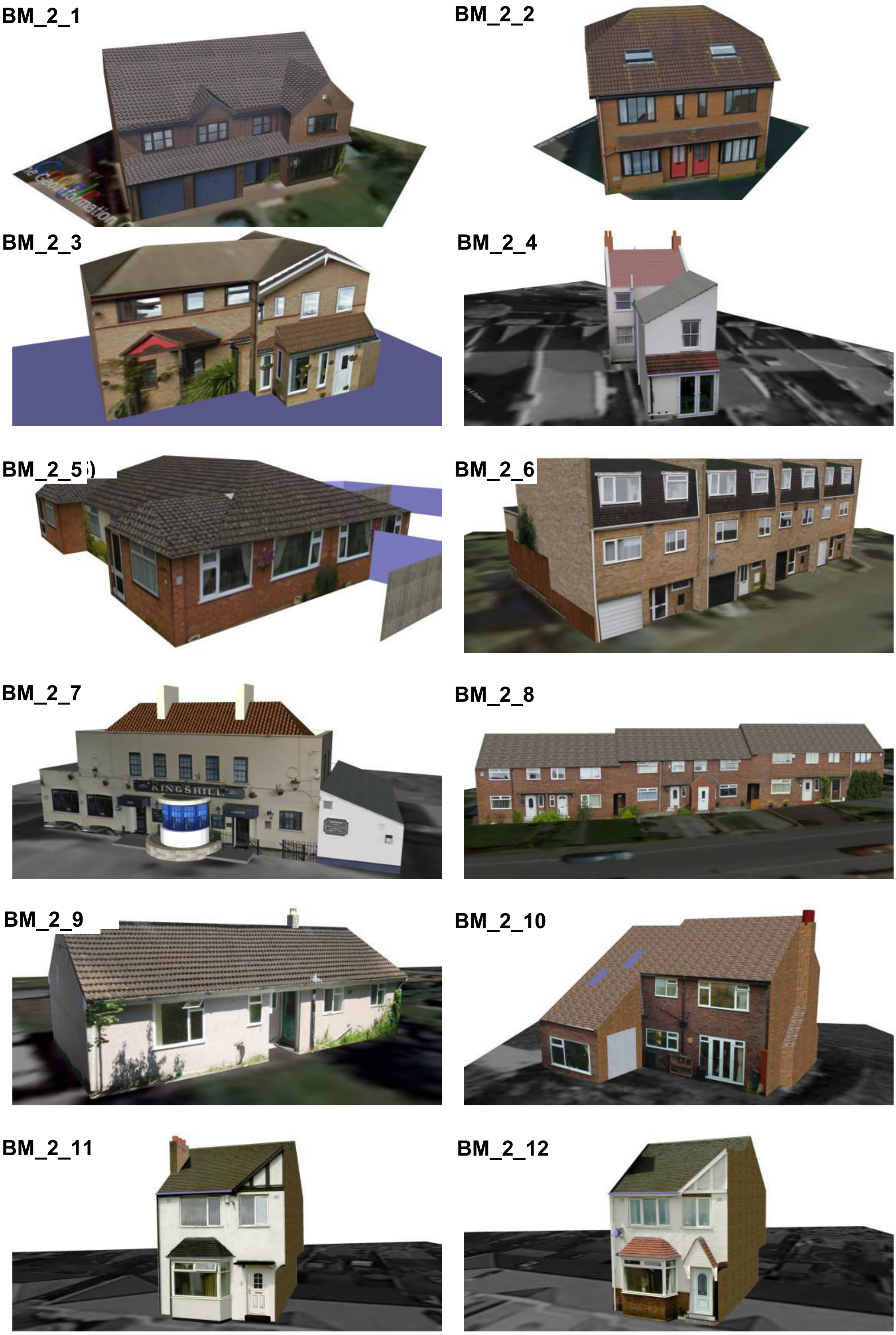


Figure 3.3: C20 3D Building Models
For details on the buildings see Table 3.1, using ‘BM_2_n’ as a reference.

Reference	**Building**	**Date Built**	**Num. Texture Map Images**	**Num. Texture-Mapped *gml: Linear Rings***	**Num. of Ground Truths Marked up**	**Num. Ground Truths after Render**
BM_2_1	3 Garland Court, Milton Keynes, Buckinghamshire	1998	24	66	30	16
BM_2_2	16 Bergamot Gardens, Milton Keynes, Buckinghamshire	1984	24	66	29	12
BM_2_3	25 Silicon Court, Milton Keynes, Buckinghamshire	1988	11	18	18	13
BM_2_4	15 Lowther, Road, Brighton, East Sussex	1900	11	581	11	11
BM_2_5	3-5 Berwick Road, Bishops Cleeve, Gloucestershire	Unclear. Between 1954-71	22	30	39	16
BM_2_6	7-7C, Station Road Cam, Gloucestershire	1976	26	5098	34	54
BM_2_7	Kingshill, Dursley, Gloucestershire	1930s	11	259	20	15
BM_2_8	24-34 Fairfield Avenue, Ormesby, North Yorkshire	1960s	6	43	31	27
BM_2_9	Meadow View, Stone Allerton, Somerset	1965	31	271	14	12
BM_2_10	89 Maidavale Crescent, Coventry, Warwickshire	1960	11	237	26	11
BM_2_11	71 Darby Road, Wednesbury, West Midlands	1930s	29	975	10	6
BM_2_12	75 Darby Road, Wednesbury, West Midlands	1930s	26	330	13	6
		TOTALS	232	7974	275	199

Table 3.1: C20 Style 3D Building Models
The table gives an indication of the complexity of the 3D building models. 'Architectural Style' and 'Date Built' sources: Historic England (2017); Mouseprice (2017); old-maps.co.uk (2017); Pevsner Architectural Guides (2013); Pevsner (2001); Pevsner *et al.* (1989); Pevsner and Sambrook (2010); RIBA (2017)

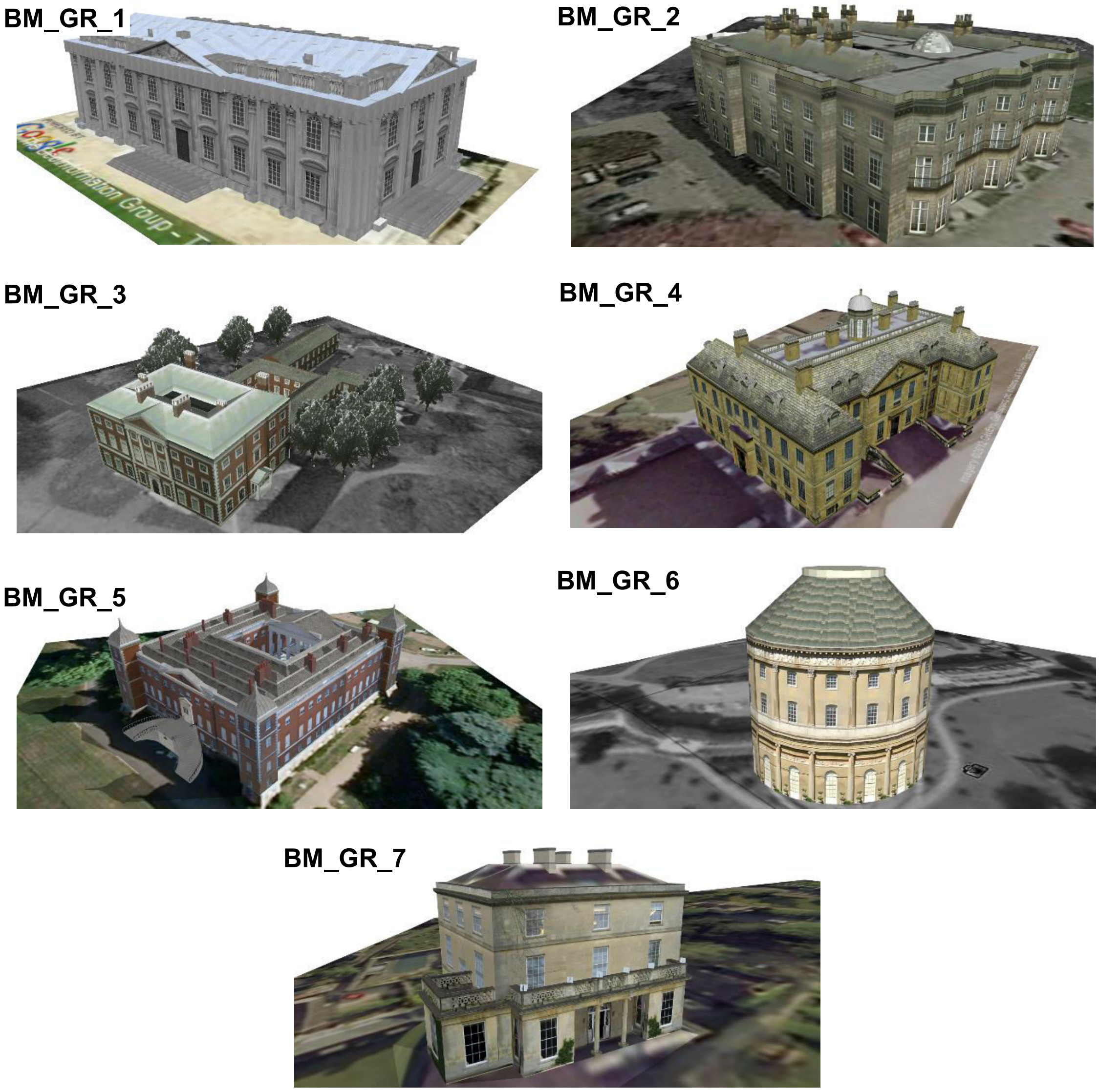


Figure 3.4: Georgian-Regency 3D Building Models
For details on the buildings see Table 3.2, using ‘BM_GR_n’ as a reference.

Reference	Building	Date Built	Num. Texture-Map Images	Num. Texture-Mapped *gml: Linear Ring*s	Num. of Ground Truths Marked up	Num. Ground Truths after Render
BM_GR_1	Senate House, Cambridgeshire	1728-68	13	1,830	8	55
BM_GR_2	Haigh Hall, Greater Manchester	1827-40	44	1,040	47	119
BM_GR_3	Lytham Hall, Lancashire	1752-64 altered early C17 parts	23	3,721	108	216
BM_GR_4	Belton House, Lincolnshire	1685-88 altered 1777-78	32	1,363	37	172
BM_GR_5	Osterley House, London	c1577 altered 1761-65	17	7,052	31	226
BM_GR_6	Ickworth House, Suffolk	1795-1829	3	272	3	36
BM_GR_7	Box House, Wiltshire	1810-20 with C19 new storeys & C20 door	68	478	27	35
		TOTALS	**200**	**15,756**	**261**	**859**

Table 3.2: Georgian-Regency Style 3D Building Models
See the caption for Table 3.1.

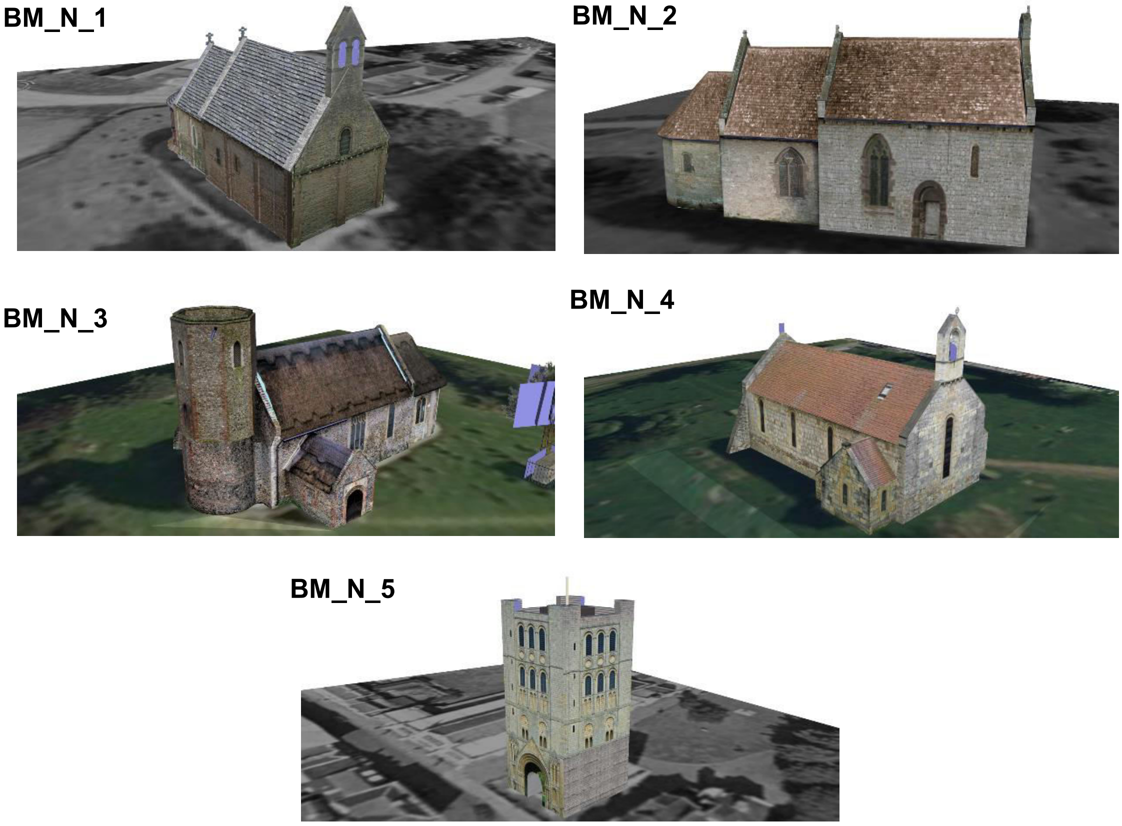


Figure 3.5: Norman 3D Building Models
For details on the buildings see Table 3.3, using ‘BM_N_n’ as a reference.

Ref	Building	Date Built	Num. Texture-Map Images	Num. Texture-Mapped *gml: Linear Rings*	Num. of Ground Truths Marked up	Num. Ground Truths after Render
BM_N_1	St Mary and St David's church, Kilpeck, Herefordshire	11-1143 Chancel C13 Bellcote c1864	77	581	9	13
BM_N_2	St Michaels All Angels' church, Moccas, Herefordshire	Mid-C12 Some windows C14	32	414	10	13
BM_N_3	St Gregory's church, Heckingham, Norfolk	C12/13 Porch C15	88	483	19	25
BM_N_4	St Nicholas' church, Askham Bryan, North Yorkshire	Late C12 Some parts altered later	35	143	14	14
BM_N_5	Norman Tower, Bury St Edmunds, Suffolk	1120-48	10	56	24	32
		TOTALS	**200**	**242**	**1,677**	**76**

Table 3.3: Norman Style 3D Building Models
See the caption for Table 3.1.

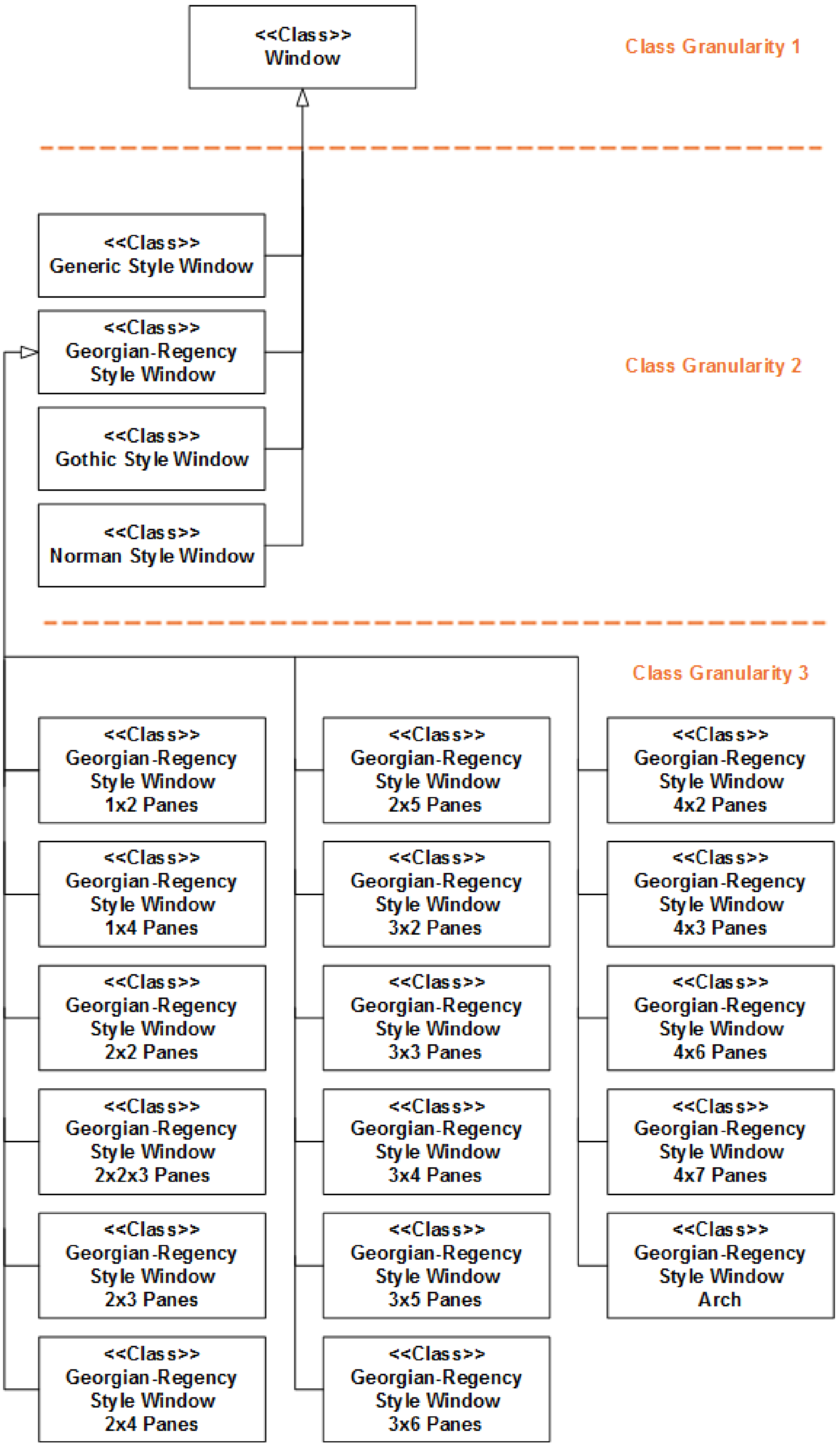
Num. Texture-Map Images	Num. Texture-Mapped *gml: Linear Ring*s	Num. of Ground Truths Marked up	Num. Ground Truths after Render
442	**17,433**	**337**	**956**

Table 3.4: Summary of Complexity of 3D Building Models
The summary is for all 3D building models used in this work i.e. covering all styles. See Table 3.1, Table 3.2 and Table 3.3 for more detail.

3.6 Data – Templates, Choice of Classes & Class Granularity

Templates were chosen on the basis of the occurrence of window or door instances of similar designs on the façade images, and on the texture map images of the 3D building models. It could be argued that to do so would encourage overfitting to the test data. Adopting such an approach is common for template matching strategies though. Crucially, the templates were taken from *different* buildings to those in the façade and texture map images. Note also that some of the templates used had designs not present on any of the façade or texture map images, which therefore reduced the risk of overfitting. Additionally, it could also be argued that, while many template and instance pairs had broadly similar designs, there were often subtle variations between the designs of templates and instances, a consequence of the high levels of variability in the designs for some styles. As such, the template sets were *not* entirely tuned to the test data.

Figure 3.6 and Figure 3.7 show a UML diagram of the choice of window and door template classes respectively, which are described below. In that the detection of the architectural style of window and door objects on the imagery was required, objects were first split into ‘window’ and ‘door’ classes. In the taxonomy of classes used, this was termed ‘granularity 1’. Within each of those classes four sub-classes were selected for each of the styles e.g. ‘C20 Style Window’, ‘Georgian-Regency Style Window’, ‘Norman Style Window’ and ‘Gothic Style Window’ for all windows. This level in the taxonomy was named class ‘granularity 2’. For the non-clustering HOG-based template matching part of this work, this resulted in eight classes at granularity 2. The standard template matching trial was only undertaken using Georgian-Regency style (or similar) images and templates. The trial used one template from the set described above, plus one template (TW_GR_3x5E) of the same glazing-panel configuration as a template from the main set (a ‘3x5 pane’ window – see below). The latter template was from a different source and was only used in the earliest part of that trial.



<<Class>> Window
Class Granularity 1
<<Class>> Generic Style Window
<<Class>> Georgian-Regency Style Window
<<Class>> Gothic Style Window
<<Class>> Norman Style Window
Class Granularity 2
Class Granularity 3
<<Class>> Georgian-Regency Style Window 1x2 Panes
<<Class>> Georgian-Regency Style Window 1x4 Panes
<<Class>> Georgian-Regency Style Window 2x2 Panes
<<Class>> Georgian-Regency Style Window 2x2x3 Panes
<<Class>> Georgian-Regency Style Window 2x3 Panes
<<Class>> Georgian-Regency Style Window 2x4 Panes
<<Class>> Georgian-Regency Style Window 2x5 Panes
<<Class>> Georgian-Regency Style Window 3x2 Panes
<<Class>> Georgian-Regency Style Window 3x3 Panes
<<Class>> Georgian-Regency Style Window 3x4 Panes
<<Class>> Georgian-Regency Style Window 3x5 Panes
<<Class>> Georgian-Regency Style Window 3x6 Panes
<<Class>> Georgian-Regency Style Window 4x2 Panes
<<Class>> Georgian-Regency Style Window 4x3 Panes
<<Class>> Georgian-Regency Style Window 4x6 Panes
<<Class>> Georgian-Regency Style Window 4x7 Panes
<<Class>> Georgian-Regency Style Window Arch

Figure 3.6: UML Diagram of Template Classes – Windows

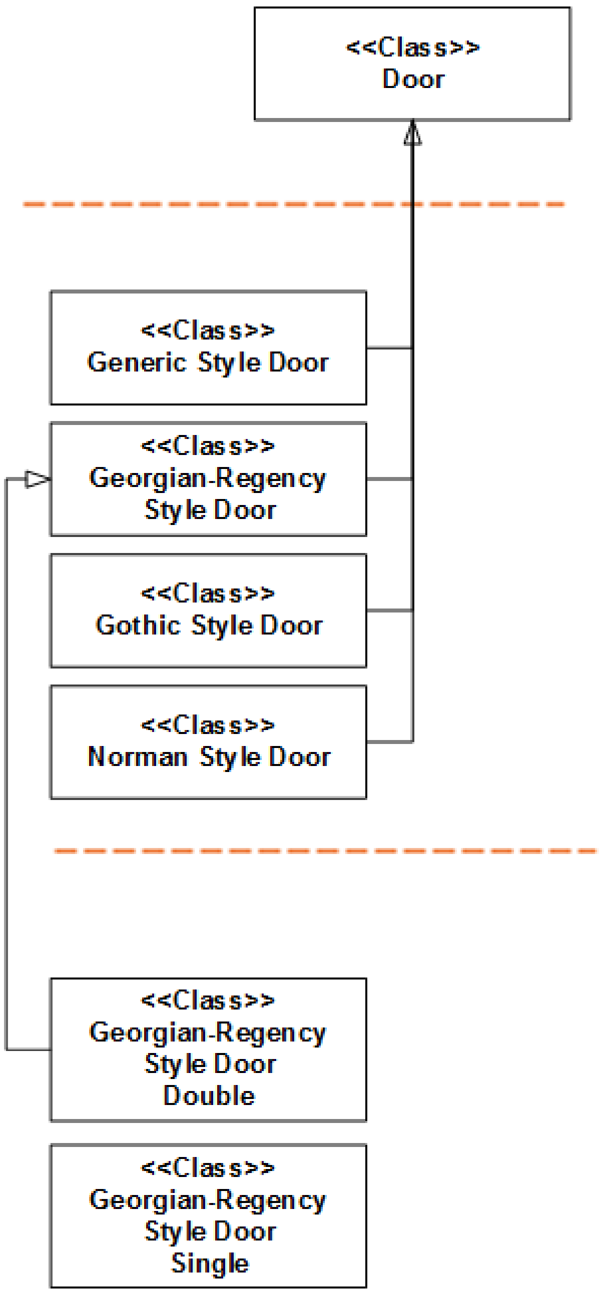


Figure 3.7: UML Diagram of Template Classes – Doors

During the non-clustering HOG-based template matching stage, experiments were also undertaken with a lower level of class granularity for the Georgian-Regency windows, based on the various glazing panel configurations horizontally and vertically. Example window class designations at this level of granularity included '3x3 pane Georgian-Regency Style Window' and '3x4 pane Georgian-Regency Style Window'. A similar class split was adopted for doors but based on whether the door feature had one ('single') or two ('double') actual doors. Door class designations at this level of granularity were 'Georgian-Regency Style Door Double' and 'Georgian-Regency Style Door Single'.

This resulted in a total of 17 window sub-sub-classes, and two door sub-sub-classes. This level in the class-taxonomy was referred to as class 'granularity 3'. However, while the approach used here included this lowest level of granularity for the Georgian-Regency style, granularity 3 classes were *not* used for the other styles. The rationale for such an approach was that those other styles possessed comparatively greater variety of window and door design i.e. their design was less standardised (see Chapter 2). F-measure was calculated at each of the above class granularities (see Chapter 5).

Table 3.5 lists the number of templates, for each sub-class, used in the non-clustering HOG-based template matching part of this work. Most of the templates had a cropped and a 'not-cropped' version, though note that even 'not-cropped' versions were usually crops of the original source images – see Section 3.8 for detail on the cropping process. Both cropped and 'not-cropped' templates were used on the basis that early experiments determined that including the brick or stone area around the window or door, i.e. when using a not-cropped template, could result in a candidate match with a higher match-score when compared to the equivalent cropped template. The reason that 'not-cropped' templates were sometimes more successful was either due to fact that the cropped template was too tightly cropped for the particular texture map image instance of a window or a door, or because the HOG descriptor of the material surrounding the window or door in the template was a good match for the HOG descriptor of the window or door instance surround on the texture map image. In total 78 unique templates were used, and 148 when 'not-cropped' versions of templates were added. Figure 3.8, Figure 3.9, Figure 3.10 and Figure 3.11 illustrate the C20, Georgian-Regency, Gothic and Norman style templates respectively. Crucially, the templates were taken from imagery of buildings which was *not* in the 3D building model or façade imagery datasets.

Cropped templates only

Style	Num. of Window Templates	Num. of Door Templates	Total Num. of Templates
C20	23	4	27
Georgian-Regency	17	5	21
Norman	10	2	12
Gothic	15	2	17
TOTALS	**65**	**13**	**78**

Cropped and 'not-cropped' templates

Style	Num. of Window Templates	Num. of Door Templates	Total Num. of Templates
C20	46	8	54
Georgian-Regency	34	5	37
Norman	19	2	21
Gothic	30	4	34
TOTALS	**129**	**19**	**148**

Table 3.5: Summary of the Split of Classes of Templates used (Non-Clustering)
Note that most templates have a cropped and 'not-cropped' version.

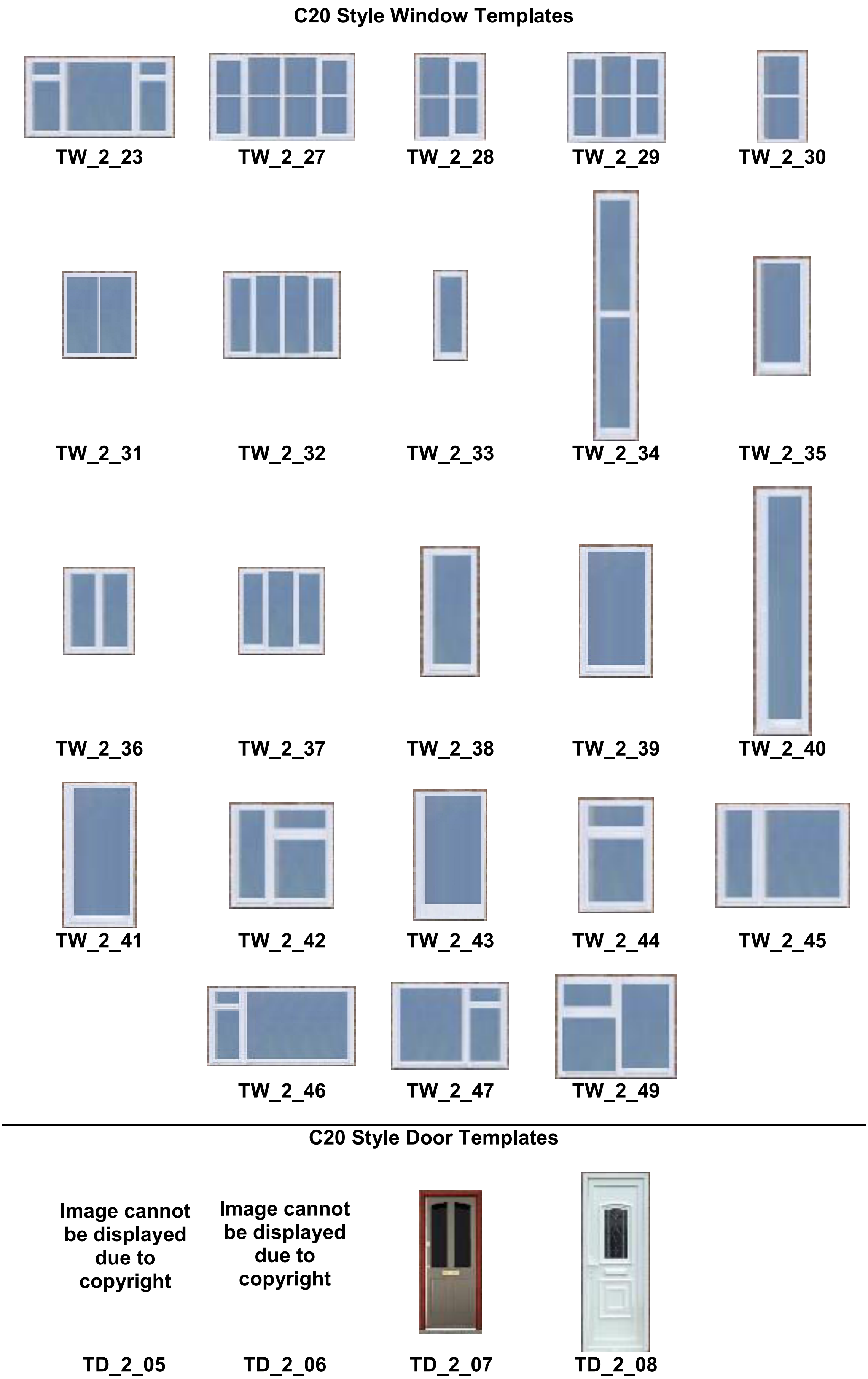


Figure 3.8: C20 Style Window Templates
'Cropped' versions are shown. Note: relative sizes of the templates shown here are as used in the methods within this work.

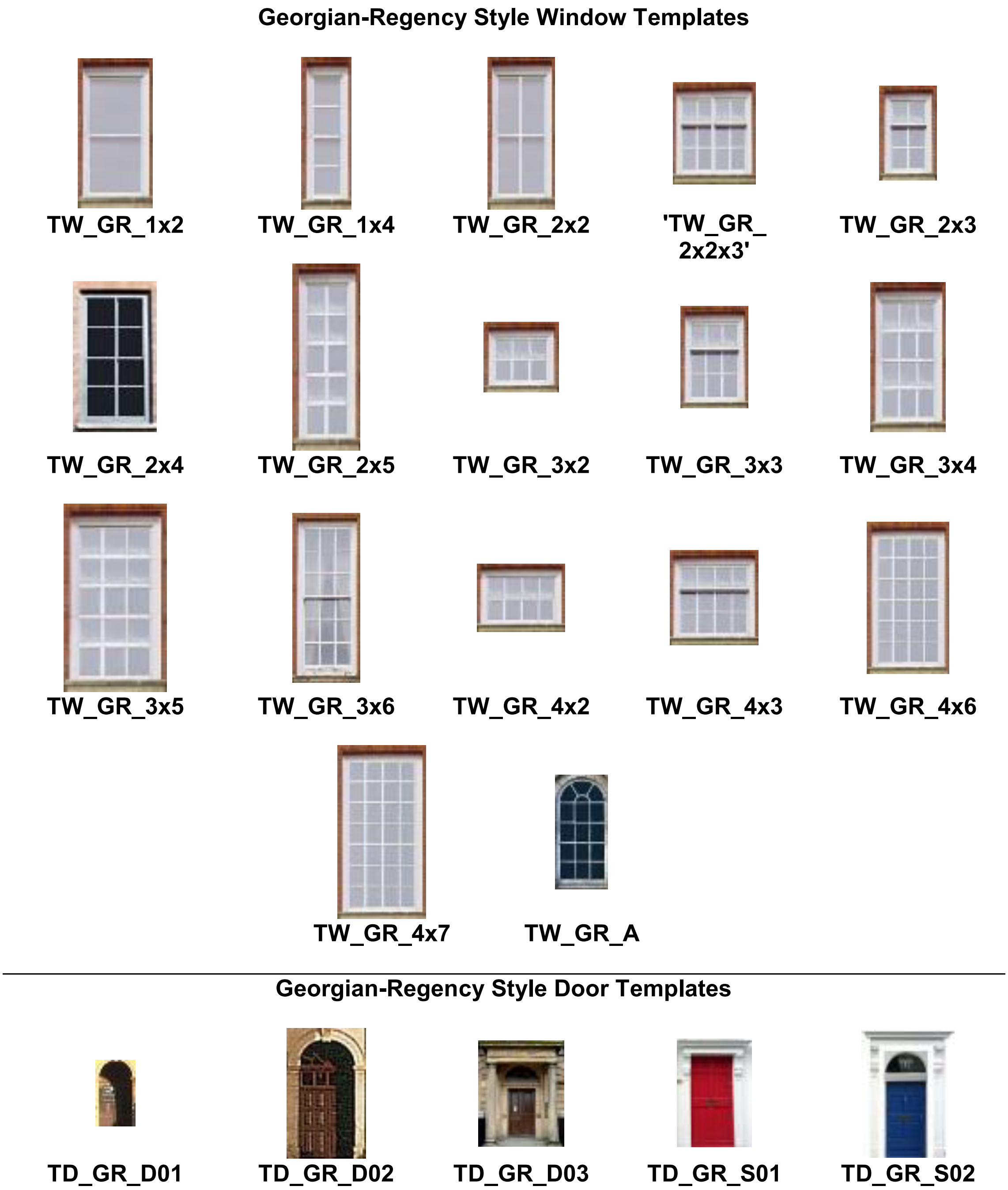
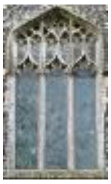
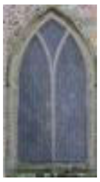


Figure 3.9: Georgian-Regency Style Templates
See the caption for Figure 3.8.

Gothic Style Window Templates


TW_GO_01


TW_GO_02

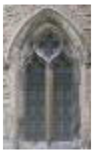

TW_GO_05

Image cannot be displayed due to copyright

TW_GO_06

Image cannot be displayed due to copyright

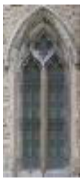
TW_GO_07

Image cannot be displayed due to copyright

TW_GO_08

Image cannot be displayed due to copyright

TW_GO_11


TW_GO_14

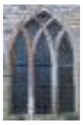

TW_GO_15

Image cannot be displayed due to copyright

TW_GO_16

Image cannot be displayed due to copyright

TW_GO_17

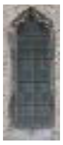

TW_GO_18

Image cannot be displayed due to copyright

TW_GO_20

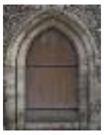
Image cannot be displayed due to copyright

TW_GO_27

Image cannot be displayed due to copyright

TW_GO_29

Gothic Style Door Templates


TD_GO_01

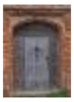
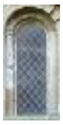
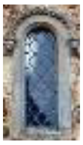
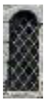

TD_GO_02

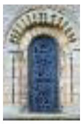
Figure 3.10: Gothic Style Templates
See the caption for Figure 3.8.

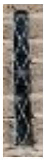
Norman Style Window Templates

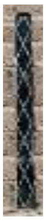

TW_N_02

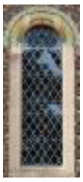

TW_N_03

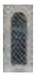

TW_N_06

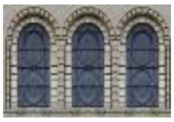

TW_N_07

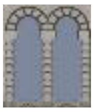

TW_N_08


TW_N_09

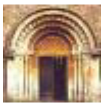

TW_N_10


TW_N_13


TW_N_15


TW_N_18

Norman Style Door Templates


TD_N_33

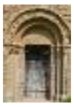

TD_N_37

Figure 3.11: Norman Style Templates
See the caption for Figure 3.8.

Note that the relative sizes of the templates shown in the figures are as used in this work. To achieve this in Figure 3.8, Figure 3.9, Figure 3.10 and Figure 3.11 the resolution of the files was made the same, even though the original files had varying resolutions in their EXIF *XResolution* and *YResolution* tags. See Appendix C for background to 'EXIF'.

With respect to the sizing of the templates used in the method, it was determined during early experiments that, for at least one of template width or template height, the minimum size to facilitate matching with no rescaling of the test image was 75 pixels. Templates were also approximately sized relative to one another according to the sizes of instances of the objects found in real buildings. The size of a template was also increased if it was determined that the complexity of the design of the window or door in the template was high. If this was not carried out then potentially important detail for the matching process might otherwise be lost. For tall, thin templates the height was made as large as was required in order for the width to be around 75 pixels, while preserving the aspect ratio (and vice versa for short, wide templates). Consequently, the minimum, maximum and mean pixel widths for the templates were 30, 218 and 106 respectively. The same values for template height were 69, 365 and 162. Lastly, note that the HOG-based template matching approaches in this work used an image pyramid, which effectively progressively increased the size of the template and ran matching at each level in the image pyramid – see Chapters 2 and 5.

The trial of the clustering of template HOG descriptors was initially conducted on a subset of styles: Norman and Gothic. To determine the efficacy of the clustering approach a different, larger, set of templates was used. For the purposes of consistency with the non-clustering approach, the abovementioned class granularity 1 and class granularity 2 scheme was still used as a basis for clustering. Note that the clustering approach required that the observations for one clustering run needed to be of the same dimensions. For this reason, each clustering run needed to be performed on HOG descriptors of the same size i.e. on templates of the same pixel dimensions. As already stressed, Norman and Gothic windows and doors possess a wide variety of design, which can also manifest itself as variation in the aspect ratio of the window frame or door region. This translated to the need to split observations for granularity 2 into a number of discrete aspect ratios. In total this resulted in 800 templates being collected. See Table 3.6 for a summary of the split (and pixel sizes for each split), noting that each row represents one clustering run.

Template Class (Granularity 2)	Discrete Aspect Ratio Name	Num. of Window Templates used in Single Cluster Run	Pixel Dimensions for each Template (Width×Height)
Gothic Style Window	'Width Narrow'	100	123×199
Gothic Style Window	'Width Medium'	100	201×199
Gothic Style Window	'Width Wide'	100	199×140
Norman Style Window	N/A	100	143×199
	SUBTOTAL	**400**	
Gothic Style Door	'Width Narrow'	100	122×199
Gothic Style Door	'Width Wide'	100	205×199
Norman Style Door	'Width Narrow'	100	140×199
Norman Style Door	'Width Wide'	100	199×202
	SUBTOTAL	**400**	
	TOTAL	**800**	

Table 3.6: Summary of the Split of Classes of Templates used (Clustering)

3.7 Data – Ground Truths

Façade images used for training, and the texture map images of the 3D building models, were marked up manually with rectangular ground truths, where the vertices were recorded in pixel coordinates. Table 3.1, Table 3.2 and Table 3.3 show the total number of ground truths recorded for the 3D building models, which are summarised in Table 3.4.

The dictionary definition of a door or a window is that they represent an opening. Consequently, the ground truths for the texture map images included: windows with no glazing, such as the louvre windows in the bellcote of St Gregory's church, Heckingham (BM_N_3); and windows that were permanent openings to the building interior, such as the twin-light examples in the Norman Tower (BM_N_5). Windows or doors that included a mesh or grill covering added for the purposes of security, such as on BM_N_3, or as an original adornment, such as on BM_N_5, were also found on some of the texture map images. As a result, they were also ground-truthed. From an architectural point of view the categorisation of such objects as 'window' is consistent with reference literature for the buildings (Historic England, 2017). Across all 3D building models there were 23 instances of such objects (7% of all window or door instances). Note that no window templates were used that had an absence of glazing, due to the low incidence of such objects.

Window or door instances on the 3D building models, or on façade images used for training, which had been entirely bricked up were *not* recorded as ground truths (the instances no longer represent openings). That said, Historic England (2017) does sometimes refer to such objects as 'blind', 'blank', 'blocked' or 'bricked-up' windows or doors. Across all the buildings used in this work there were three instances of windows or doors which were blocked after construction was completed, and 12 instances of windows where it is not clear whether the windows were originally blank or were blocked-up after construction was completed (Historic England, 2017).

Ground truths were marked up on the façade images used for training, in the same manner, though in order to send to training a balanced spread of design types not all window and door instances were marked up. The choice of ground truths was achieved by running HOG-based template matching on the façade images used for training, and then selecting the strongest candidate matches.

Points representing the location of negative class were also marked up, to represent locations which were not the centre of a window or door instance. Initial experiments were conducted using a pseudo-randomised method to locate the points representing observations for the negative class, but better results were achieved through manually placing such points. These manually located points were placed in regions on the façade images on brickwork and roof tiles, and also just inside window and door frames, thereby training the classifier to *not* recognise potentially lower-scoring candidate matches as windows or doors.

Table 3.7 shows the total number of positive class and negative class ground truths marked up in the training images. Note that the inclusion of class granularity 3 meant that comparatively more Georgian-Regency objects were marked up. Also, note that for Norman and Gothic the counts are combined because instances of windows and doors for these styles only occur on the Norman 3D building models. Aside from BM_GO_1, which was only briefly used for the image matching (SIFT) trial, no Gothic 3D building models were used.

Style	Num. of Façade Images used for Training	Num. of Unique Buildings	Num. of Positive-Class Ground Truths Marked up	Num. of Negative-Class Ground Truths Marked up
C20	20	20	35	210
Georgian-Regency	24	23	52	312
Norman & Gothic	25	22	35	210
TOTALS	**69**	**65**	**122**	**732**

Table 3.7: Counts of Total Façade Images, Unique Buildings, Positive-Class & Negative-Class Ground Truths used for Training

3.7.1 Data –Training Observations – Positive-Class:Negative-Class Ratio

It is a generally accepted principle within the field of machine learning that the split of training observations for the positive class and observations for the negative class should either be balanced, or representative of the split likely to be encountered in the real world. The selection of which approach is used will depend on the nature of the problem being addressed.

SVMs are inherently binary (two-class) classifiers. Compared to multi-class classifiers, binary classifiers can be more sensitive to an imbalance of training observations. Consequently, the SVM implementation used employed a 'one-vs-one' learning strategy, thus creating a 'learner' for each class pair, to address the multi-class problem presented in this work (Anthony *et al.*, 2007). The one-vs-one strategy has been found to be effective when using an imbalance of classes (Anthony *et al.*, 2007).

In the case of building façades, it was determined empirically that the ratio of 'window or door area' to 'non-window or non-door area' was approximately 1:6. This was achieved using the façade images, for each image in turn, as follows: (i) calculating the pixel area for the ground truth bounding boxes; (ii) drawing a rectangle around the façade boundaries in the image and calculating their pixel area; (iii) subtracting (i) from (ii). The mean for each of (i) and (iii) across all façade images was then calculated, resulting in the 1:6 ratio. Early experiments determined that using the same number of positive-class and negative-class observations (where the latter were sampled randomly, programmatically, from the full set of negative-class observations) resulted in a negligible change in the performance of the classifier. For every positive-class training observation (a window or door instances) six negative-class training observations were therefore used.

3.8 Data – Pre-processing

3.8.1 Projective Transform

When a 3D building model is produced, texture map images should, in theory, be rectified by the modeller, so that they appear to be posed square-on to the façade in question. Consequently, templates, façade images used in training and any replacement texture maps were projective-transformed manually. This was achieved by first marking out the four vertices of what should end up as rectangular area. Using these control points, the transformation parameters were obtained and the transformation applied thus correcting the perspective distortion.

3.8.2 EXIF Tags

See Appendix C for background to ‘EXIF’. All texture map images and façade images had any missing *IFD0:XResolution* and *IFD0:YResolution* tags added to their EXIF data using *exiftool* (Harvey, 2017). These tags were needed to ensure that the export of figure objects (images marked up with candidate matches) were the same pixel dimensions as the original images.

3.8.3 Texture Map Image Replacements

During this work it was determined that a number of 3D building models available on the Trimble *3D Warehouse* used imagery taken from Google *StreetView*, something which, in older versions of Trimble *SketchUp*, could be achieved in a semi-automated fashion using the real-world coordinates of the model being constructed. However, such imagery can often suffer from distortions, including a reduction in the coherency of lines, resulting from poor rectification within Google *StreetView*. Consequently, such images were replaced prior to processing. To facilitate template matching on texture map images that were very small, the pixel size of the image was automatically doubled, using bicubic interpolation and anti-aliasing (see Chapter 5). Where doing so would still have resulted in instances of windows or doors on the texture map image that were smaller than half the pixel dimensions of the smallest template, i.e. 37.5 pixels wide or high, the texture map image was also replaced in advance. Lastly, if the texture map image lacked sharpness then it was replaced with a higher quality image, prior to processing. Prior to this, experiments were conducted with various auto-sharpening techniques, but the results proved unsuccessful. Replacement texture map images were obtained using Google *Image* and *Flickr* searches and corrected for perspective distortion using projective transform.

These replacement texture map images (and also some templates) were sometimes constructed by editing original images in *paint.net* (dotPDN, 2017). Such editing could include splicing portions from two or more images, plus copy-paste and manual editing within a single source image. For example, where occlusions (gravestones, persons, etc) occurred in the replacement images, *paint.net* was used to restore the occluded portion.

Note that the images chosen as replacements never contained an occluded window or door.

Replacing poor quality texture map images can be very labour intensive as it requires searching online for photos of what are often obscure parts of a building, especially when the required aspect is not available on *StreetView*. It could be argued that this is the reason the quality of some of the texture map images in original models may be poor in the first place i.e. it can be hard, or in some cases impossible, to find good photos. Note once again that 19 (4%) of the texture map images were replaced.

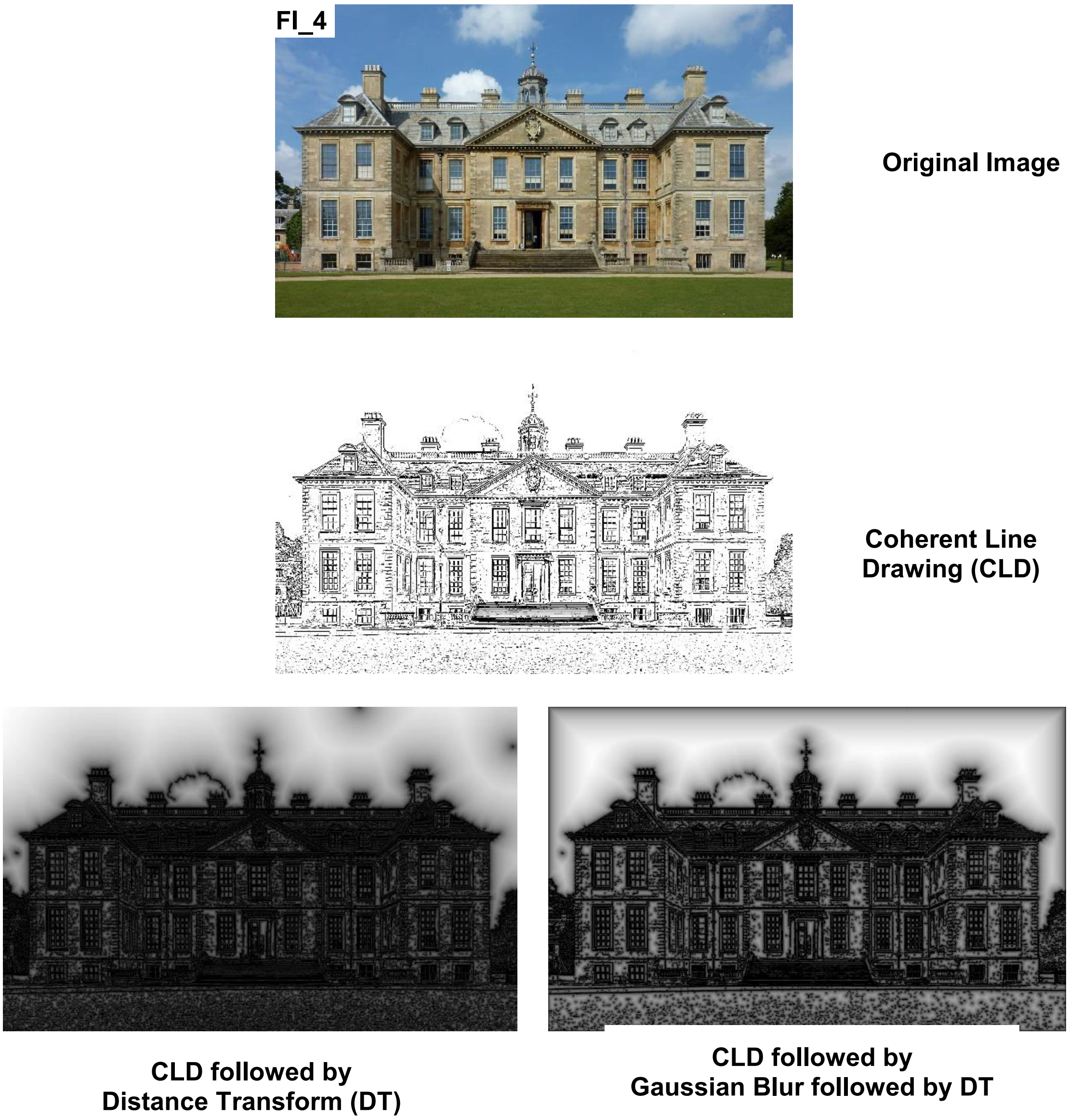
3.8.4 Template-Specific Pre-Processing

The boundaries of any templates created or cropped from existing imagery (aside from the 'not-cropped' versions) were consistently defined by the boundary of the window sill, the door itself or any surrounding orders, whichever were wider or taller.

Georgian-Regency and C20 style templates proved more effective when the glazing was blanked out. In doing so, reflections, the contents of which might otherwise have been picked up as edges, were removed. Early experiments to remove the glazing from Gothic or Norman templates were not so successful though. Arguably, the successful response for Georgian-Regency was the result of comparatively more standardised glazing style versus the greater variety of glazing for the Gothic and Norman styles. Essentially, the glazing style is more discriminative for Gothic and Norman windows.

3.8.5 Coherent Line Drawing, Gaussian Blur & Distance Transform

The standard template matching trial (see Chapter 4) included initial experiments using templates and façade images that had been pre-processed using the coherent line drawing method of Kang *et al* (see Chapter 2). The rationale for the use of the Kang *et al* method was that it was generally effective in extracting long, curvilinear lines while minimising the amount of clutter and noise in the output. Early HOG-based template matching experiments also included this pre-processing step. During the standard template matching trial, experiments were also conducted that included the application of Gaussian blur or distance transform (or both, in that order) to the coherent line drawing versions of the façade images and templates (again, see Chapter 2). The kernel (filter) size for



FI_4

Original Image

Coherent Line Drawing (CLD)

CLD followed by Distance Transform (DT)

CLD followed by Gaussian Blur followed by DT

Figure 3.12: Examples of Coherent Line Drawing, Distance Transform & Gaussian Blur Note that, purely for illustration purposes, each pixel intensity value in the above two distance transform images have been converted to $\ln(value + 1)$.

$$Filter\ (Kernel)\ Size = 2\lceil 2\sigma \rceil + 1$$

Equation 3.1

$$D_{Eu} = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

Equation 3.2

the Gaussian blur was kept at the default i.e. using $\sigma = 0.5$ (standard deviation) as calculated using the standard calculation, as shown in Equation 3.1. The standard Euclidean distance transform equation (as used) for the distance between two points that have coordinates (x_1, y_1) and (x_2, y_2) is shown in Equation 3.2.

The Gaussian blur was carried out in an attempt to consolidate scattered disjoint components of what should be solid lines, and to extend the influence of the highly localised features i.e. lines, thus increasing the tolerance of the matching to geometric differences between the template and the texture map image. Gaussian blur was also carried out as part of data augmentation for the supervised classification stage (see Chapter 6). Figure 3.12 demonstrates the effect of combining coherent line drawing, distance transform and Gaussian blur in various combinations.

3.9 Summary

This chapter has detailed the challenges faced as a result of the requirement to gather the majority of the data used in this work, due to a lack of suitable reference datasets. In addition, the data used in the work has been described, including the nature of any pre-processing. Chapters 4-6 describe the use of that data in the methods employed by this work and expand upon some of the potential implications for the dearth of suitable available data.

Chapter 4

Image Matching (SIFT) & Standard Template Matching – Trials

4.1 Introduction

Having introduced the data used in this work in Chapter 3, this chapter represents the first use of that data, namely in the trials of image matching and standard template matching. The process of image matching, using the highly popular SIFT method explained in Chapter 2, attempts to detect the same feature in two images of the same scene. Such correspondence between two images can pave the way for a number of applications, including the linking of features in a texture map image of a 3D building model, and a richly captioned photo in an internet image repository such as *Flickr*. In the case of this thesis, the trial was intended to be a precursor to future work which could then use such captions to produce even richer semantics, and thus potentially enable an even richer virtual cultural heritage experience.

The trial of standard template matching was a proof-of-concept of the efficacy of template matching for addressing the problem of architectural object detection. Specifically, the trial needed to determine if template matching could detect the sort of window and door detail required for its application to be worthwhile. The trial included a comparison with the HOG-based template matching approach (the HOG-based template matching approach is described in Chapter 5).

The methods and trial results for each of the image-matching and standard template matching trials are described below.

4.2 Image Matching (SIFT) – Trial

4.2.1 Approach

As mentioned above, the purpose of the trial was to enable what will now be future work to auto-generate more detailed captions for specific instances of window, door or other architectural components. For example, rather than simply captioning a particular window or door as a 'Norman style window' or a 'Gothic style door', one might, using captioned online photo collections or guide texts, caption a door as: 'Nave door … semi-circular headed of two richly ornate and moulded orders, outer order with chevron ornament and grotesque beasts and fishes in linked medallions, inner order with similar radially arranged animal heads, curious beasts and fishes, and beakheads; Tree of Life motif to tympanum. Jambs of two orders with outer attached shafts decorated with twisting serpents and inner shaft to left with two warrior-type figures entwined in vine, the shaft to the right has twisting vine and palmette motifs' for BM_N_1, Kilpeck church (Historic England, 2017). Such detail could enable even richer geo-data experiences within augmented reality and virtual cultural heritage tours for example.

FI_1 and FI_4 were used for this trial. In each case a corresponding 3D building model for the building from the Trimble *3D Warehouse* was used, i.e. BM_GR_4 and BM_GO_1 respectively. SIFT-based matching was carried out with RANSAC, matching between the façade image and a texture map image from the 3D building model for the same part of the building.

Using the SIFT matching implementation (Lowe, 2005), which corresponds to the original work (Lowe, 2004), RANSAC was used to filter outlier matches. Lowe's method initially applies a distance threshold to the dot products of the keypoint descriptor vectors, between each of the two images. The RANSAC approach used creates 'homogeneous feature coordinates for fitting of the fundamental matrix' and then sets a second 'distance threshold for deciding outliers' (Kovesi, 2005b). The fundamental matrix was calculated using Kovesi (2005a) to split inliers from outliers using a second distance-threshold.

In order to visualise the output (a set of matches) a graphic was also produced, showing the two input images side by side, with inliers represented as a straight red line linking the same feature in each image, and outliers with a cyan line. Note that the method converts the images to grayscale before processing. The visualisation of the output also

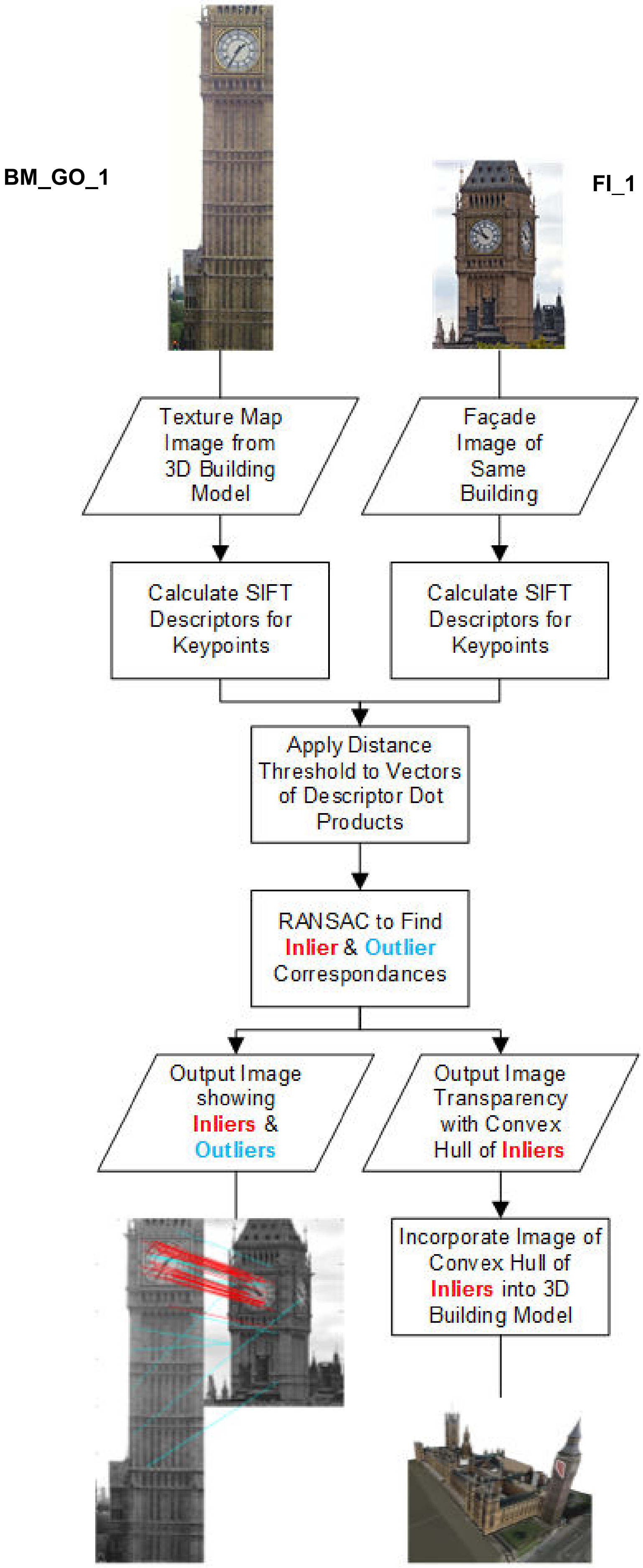

BM_GO_1
FI_1
Texture Map Image from 3D Building Model
Façade Image of Same Building
Calculate SIFT Descriptors for Keypoints
Calculate SIFT Descriptors for Keypoints
Apply Distance Threshold to Vectors of Descriptor Dot Products
RANSAC to Find Inlier & Outlier Correspondances
Output Image showing Inliers & Outliers
Output Image Transparency with Convex Hull of Inliers
Incorporate Image of Convex Hull of Inliers into 3D Building Model

Figure 4.1: Image-Matching (SIFT) Trial – Process Flow

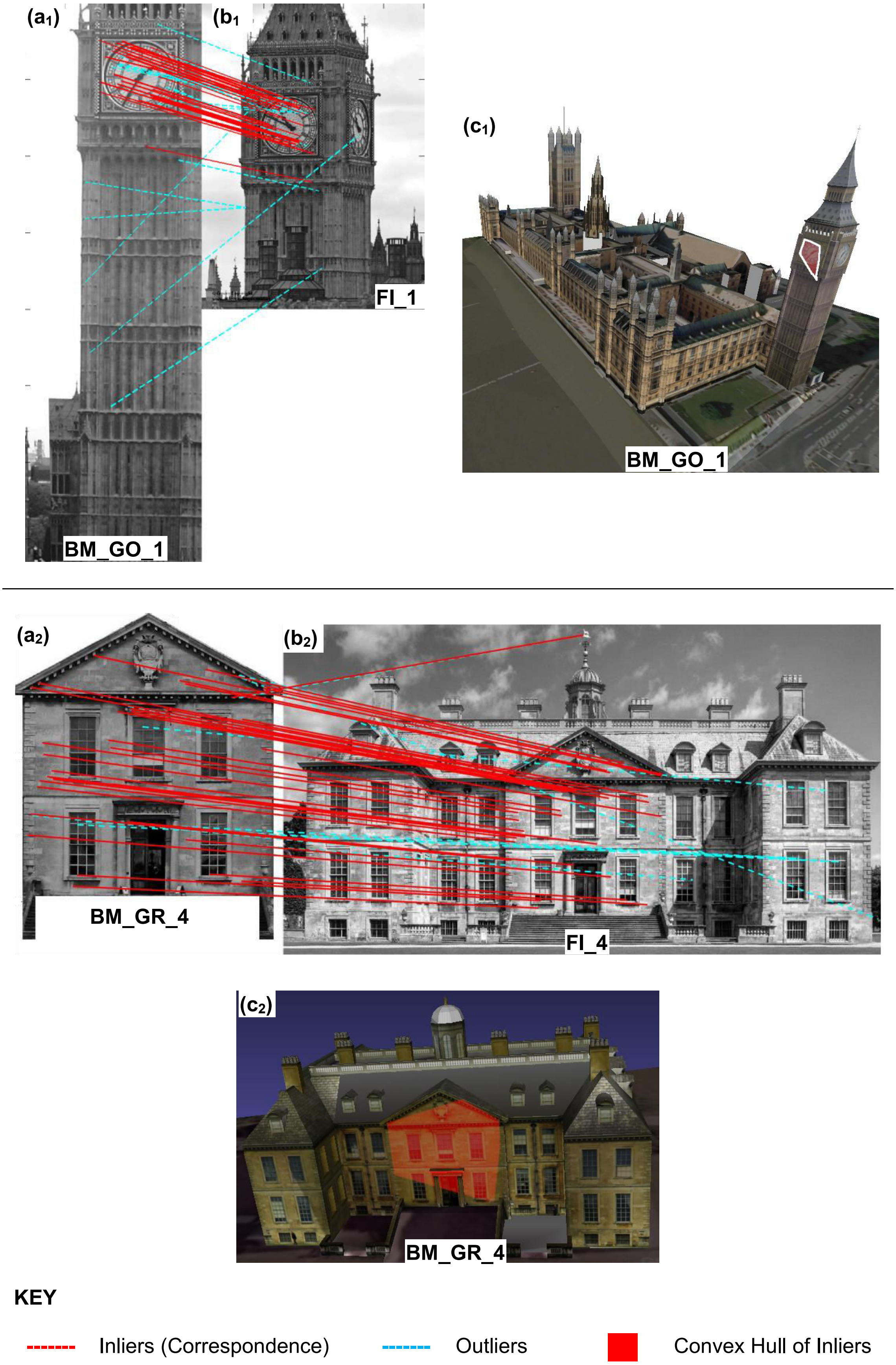


Figure 4.2: Image matching trial – Trial Results
(a_1 & a_2) Texture map image from the 3D building model, with inliers and outliers shown between each texture map image and (b_1 & b_2) each of the façade images respectively. (c_1 & c_2) Convex hull of the inliers shown on the texture map image of the 3D building model.

showed a convex hull around the inliers to illustrate the region of correspondence. This was achieved through the creation of a mask, where pixel values of 255 were applied as a red transparency to the original texture map image. Pixel values of 0 were not processed. The application of the mask was programmed in C. The resulting image was then used to replace the original texture map image in the 3D building model in order to illustrate the convex hull area. The region represented by the convex hull is an example of a region to which a detailed caption, taken from the caption in the internet image repository, could be assigned.

See Figure 4.1 for an illustration of the process flow for the image-matching (SIFT) trial. The trial of SIFT-based image matching represents a small portion of the overall work in this thesis, therefore the method is not discussed further here.

4.2.2 Trial Results

Figure 4.2 shows the results of the image matching trial. As can be seen, inliers generally demonstrate correspondence. However, a few inliers show an incorrect result e.g. the inlier that terminates on the top of the dome (‘cupola’) on the roof on FI_4. Clearly, if the method were to be used in future work this would need to be investigated. As with the method description above, no further discussion of the trial results is conducted here.

4.3 Standard Template Matching – Trial

Kroon’s standard template-matching code operates on a single template and a single test image. For the work here façade images FI_ 2, FI_ 3, FI_ 4 and FI_ 5 were used in the trial, and, toward the end of the trial, validation runs were conducted with texture map images from BM_GR_4. Kroon’s code was placed into a pipeline, to which an image pyramid process was added. The image pyramid progressively increased the size of the template, whereupon matching was conducted at each level of the pyramid. The scales for the image pyramid were determined empirically and set at 90-110% scale in steps of 1%. The range was chosen by measuring the pixel heights of the window and door instances on the façade images. This then indicated the maximum and minimum resizing of the template and the approximate steps between those limits, from which the maximum and minimum scaling and steps for the image pyramid could then be calculated.

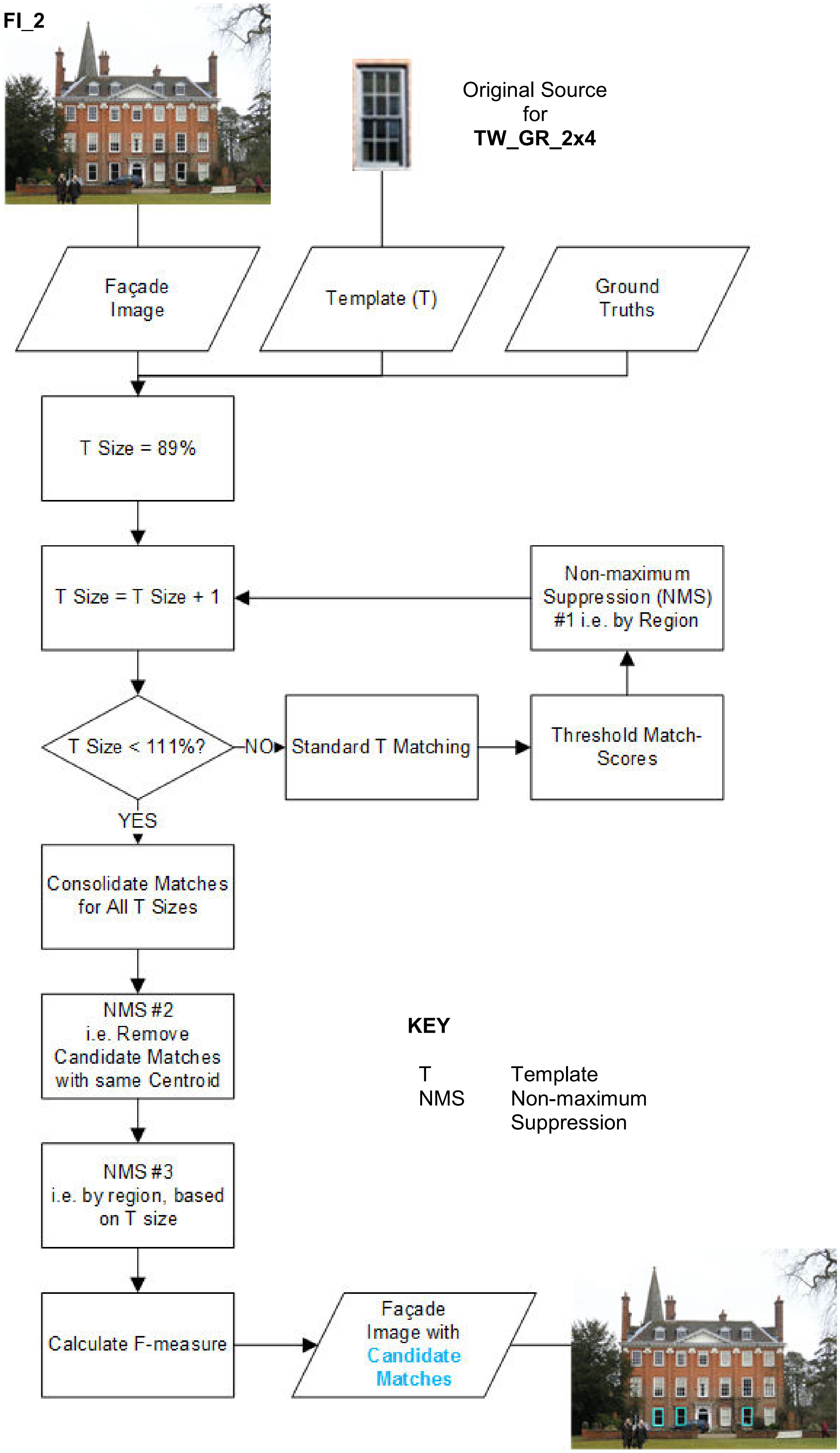


Figure 4.3: Standard Template matching trial – Process Flow

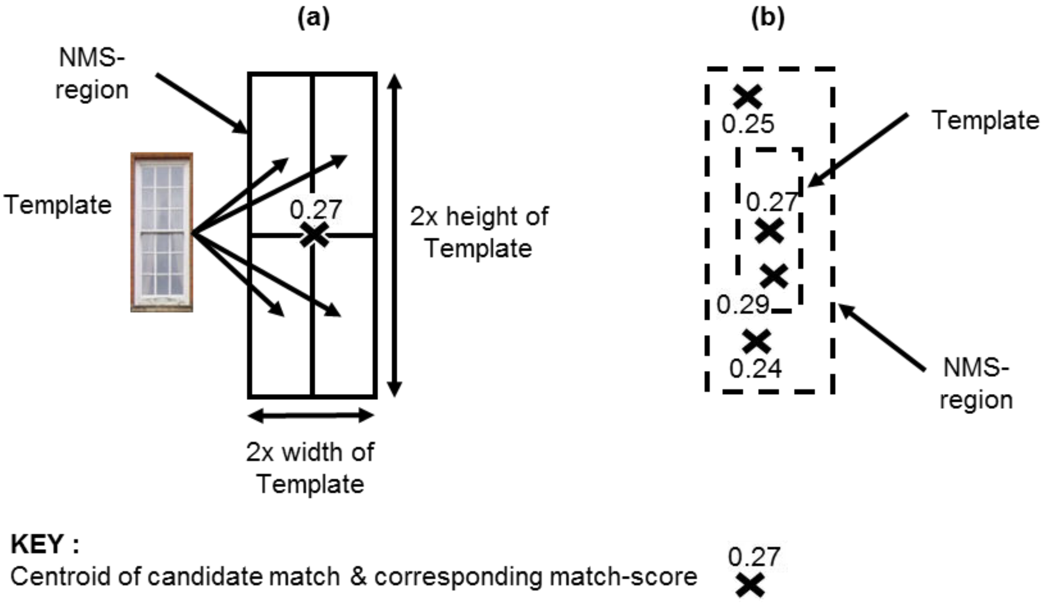


Figure 4.4: Non-maximum Suppression Approach – Standard Template Matching Trial (*a*) Illustrates how the NMS-region, centred on the current pixel, is sized i.e. 4x the area of the template. The current pixel is the location with a match-score of 0.27. (*b*) The match-score for the current pixel, i.e. 0.27, would have been set to zero by the non-maximum suppression approach (it was not the maximum match-score in the NMS-region).

To reiterate, using test data to tune a parameter in this manner is not generally acceptable. However, since this was an early trial it was deemed an acceptable approach. In that this was a trial, using only a small number of templates and façade images, the potential issues regarding small instances of windows or doors on test images were not considered, unlike the approach in Chapter 5. Thresholding and non-maximum suppression were run on the candidate matches. Finally, the F-measure was calculated for the result. Below, the standard template matching method and the associated pipeline are described in more detail. Figure 4.4 illustrates the process flow used.

Kroon's code calculates correlation between the template and test image as SSD and NCC values, using FFT to achieve the correlation. This provides a match-score for every pixel in the test image. To achieve this the test image is first padded by the size of the template minus 1 pixel. Essentially, a match-score for a pixel represents the likelihood that location is the centroid of an instance of an object which is a match for the template.

4.3.1 Thresholding

The pipeline also included an added step to threshold the match-scores, thereby reducing match-scores below the threshold to zero. The threshold for windows and doors was determined empirically, using F-measure (see below) and a separate set of façade images. This set of images included FI_5, which explains why that image was not used in the comparison. The threshold value was determined as follows. Using each of the abovementioned ground-truthed façade images in turn as validation data, the full collection of templates used for the trial were used to run standard template matching, with no thresholding in place. The maximum and minimum match-scores were then determined from those results. Next, standard template matching was executed with the same data, in an iterative fashion, with the threshold set in the range [min match-score, max match-score] at increments of 0.1. In this manner, by using F-measure, the optimal second decimal-place threshold-range was determined e.g. [0.7, 0.8], i.e. higher F-measures informed the decision on the optimal range. The iterative process was then repeated in that range, in increments of 0.01, to arrive at the threshold value. In practice this meant that the threshold was set at 0.77.

4.3.2 Non-maximum Suppression (NMS)

Non-maximum suppression (NMS) was also added to the pipeline, and operated in three stages, namely: (1) on candidate matches obtained with the template at a single size; (2) on candidate matches for all sizes of template combined, to ensure that only one match-score existed for each (x, y) location in an image; and then (3) once all remaining candidate matches had been combined for all template sizes. (1) and (3) operated as follows. Using the matrix of match-scores, padded by template size and width with zeros, every pixel not set to zero by thresholding was processed. Specifically, if a pixel had a match-score greater than zero then a neighbourhood ('NMS-region') was dynamically constructed, sized twice the template width and twice the template height, centred on the current pixel. If the match-score at the current pixel was not the maximum match-score in the NMS-region then the match-score at the current pixel was zeroed. Figure 4.4 illustrates the non-maximum suppression approach used for stages (1) and (3).

4.3.3 F-measure

As mentioned above, F-measure was used to tune thresholds. It was also used to review the results of the trial. Due to the sensitivity of the standard template matching approach to the placement of the component parts of objects (see below), the F-measure results were highly variable between different combinations of template and façade image. Consequently, no F-measure results are given here. Nonetheless, for completeness, the method used to calculate F-measure is described below.

The method used here only operated on a single template, meaning there was no concept of class granularity in the F-measure calculation. Ground truths were marked up on the façade images as centroids of true instances. For each candidate match that remained after NMS had completed, the pixel-distance from the centroid of the candidate match to the centroid of the ground truth was calculated. If the centroids of a candidate match and ground truth were horizontally within a distance equal to half of the width of the template that achieved the candidate match, and vertically within half the height of the same template then the candidate match was deemed a true positive. If more than one ground truth met these criteria then the closest ground truth was chosen. The downside of the above approach was that no consideration was given to the relative sizes of the ground truth and the candidate match, e.g. a candidate match could be significantly smaller or larger than the ground truth instance (arguably a poor result) and still be a true positive if the centroid proximity conditions above were met. The centroid proximity approach was therefore replaced by the method used in Chapters 5 and 6, that is with a standard area-overlap method in the field i.e. the *PASCAL VOC* area-overlap approach mentioned in Chapter 2.

4.3.4 Trial Results – Standard Template Matching versus HOG-based Template Matching

This section describes the approach and the results of an extension to the trial: the comparison of (a) the standard template matching approach described above with (b) the HOG-based template matching approach described in Chapter 5. On the basis that (b) was ultimately chosen as the approach for the core of the work in this thesis, the method for (b) is described in Chapter 5, not here.

The threshold for (b) was set at 0.26, in a similar way to the approach used for (a). Because the image pyramid used in (a) was less extensive than that for (b), templates were resized manually in advance using the sizes of the instances on the façade images. While tailoring the size of the templates to specific instances on the façade images is not an ideal approach, for this trial it was deemed acceptable. For this comparison the same templates were used for the trial runs on (a) and (b). shows the result of running (a) and (b).

In summary of the results, (a) detected 5% of the windows and 3% of the windows with the same glazing-panel configuration as the template. The equivalent results for (b) were

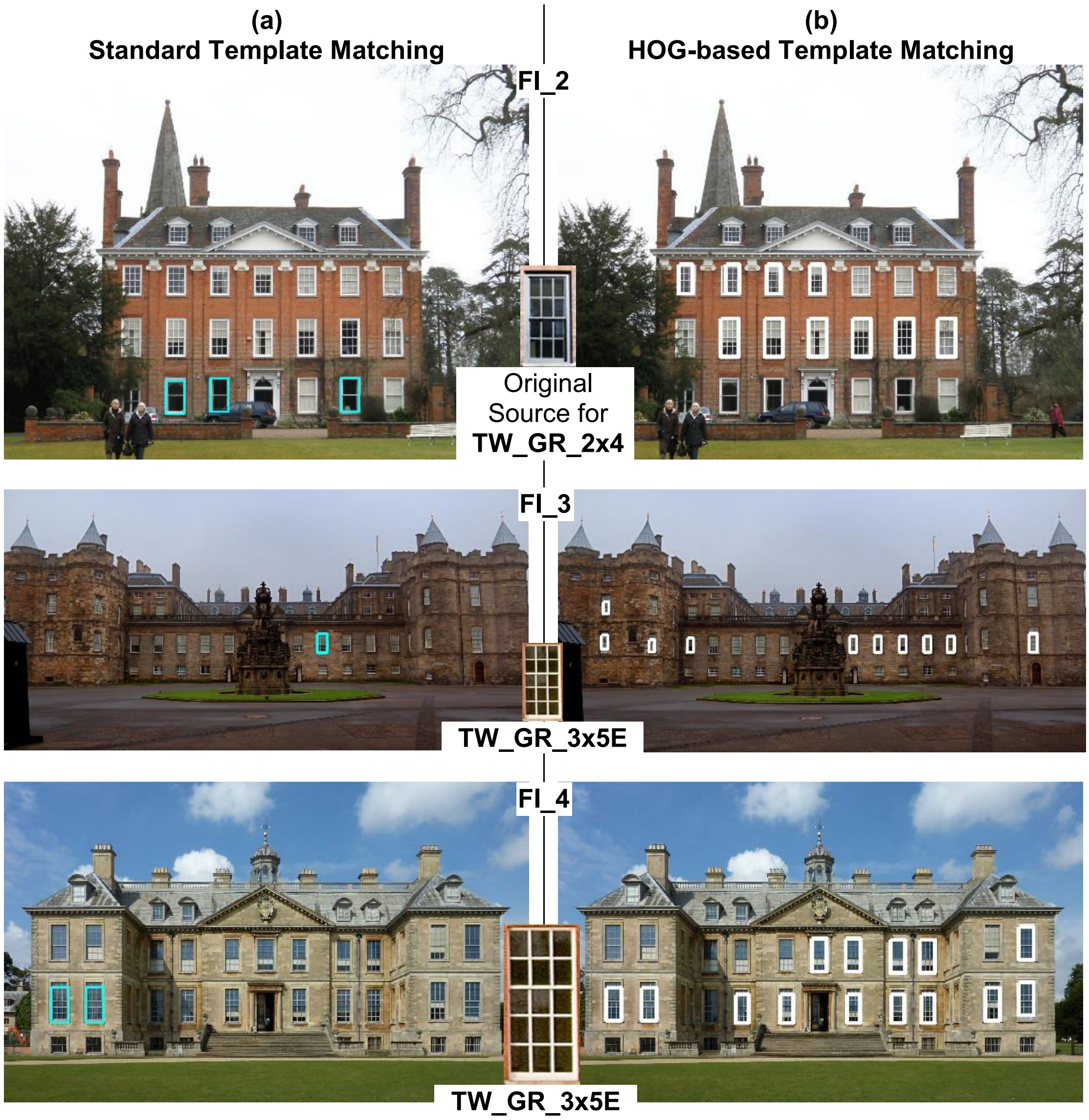


Figure 4.5: Standard Template Matching versus HOG-based Template Matching – Trial Results Rendered onto Façade Images
(*a*) Candidate matches shown as **cyan** bounding boxes. (*b*) Candidate matches shown as white bounding boxes. A single template was used for each façade image – the templates used are shown in the centre. For illustration purposes the templates above are sized relative to the actual sizes of the ones used during processing.

32% and 60% respectively. Additional early experiments were conducted using (a) with façade images and templates that had both been pre-processed using various combinations of coherent line drawing, Gaussian blur and distance transform. The results generally improved when using such pre-processing, especially when coherent line drawing was used, but they were still significantly worse than the results obtained with (b). Further improvements were achieved with (a) but using unique thresholds tuned to each façade image. At this point it should also be noted that the result from using (a) proved highly sensitive to the threshold used – an alteration at the 1st, 2nd or even the 3rd decimal place could sometimes make a significant difference to the result. The findings here suggest that (a) was, compared to (b), more sensitive to the location of the features within an object.

While the above result of 32% for (b) is low, it was obtained with just one template, with a particular glazing-panel configuration. To put this in context, the window instances on the façade images have windows with a variety of glazing-panel configurations. The result is more impressive when one notes that 20% of the window instances on the façade images were of the same glazing-panel configuration as the templates used. As such, (b) identified 60% of those window instances. This suggests that (b) has the better potential to detect fine levels of detail in templates and object instances, compared to (a). It is for the above reasons that the trial of (a) was taken no further, and that (b) was chosen as the core method to be used in this thesis.

The 32% detection rate for (b) could, arguably, be improved by extending the approach to use of multiple templates, including multiple glazing-panel configurations.

4.4 Summary

This chapter has expanded upon the method used for the image matching (SIFT) trial, and upon the method used for the standard template matching approach. The former produced insights which might be useful for future work and even greater levels of semantic enrichment. The latter included a comparison with the HOG-based template matching approach (Chapter 5) which determined that HOG-based template matching would be used for the core of the work in this thesis, i.e. the work carried out using the methods not only in Chapter 5, but also in Chapter 6.

Chapter 5

HOG-based Template Matching (no SVM)

5.1 Introduction

So, how to identify missing windows and doors in existing 3D building models? Chapter 3 determined that the HOG-based template matching method of Xiao (2013) and Zhang *et al.* (2013b) was more scalable to a variety of images than a standard template matching approach (with the implementation of Kroon, 2011) and should therefore form the basis of the approach used in this thesis. That chapter also noted that the HOG-based template matching approach might achieve stronger results if it were added to a pipeline that used multiple window and door templates. In order to enable semantic and geometric enrichment of existing 3D building models, the method would need to use the coordinate reference system of those 3D building models. This would facilitate the inclusion of some heuristics, based on real-world coordinates, to filter out poor candidate matches. Lastly, to ensure the spatial accuracy of detected windows, validation would be needed. After all, to enable geometric enrichment the new windows and doors would need to be correctly located.

The chapter is therefore split into the following sections, each of which describes the methods used: (1) CityGML parsing; (2) pre-processing; (3) HOG descriptor density considerations; (4) the use of multiple templates using match-score normalisation; (5) non-maximum suppression; (6) cropping considerations; (7) the calculation of the 3D real-world coordinates for the new windows and doors, achieved via transformations and ren-

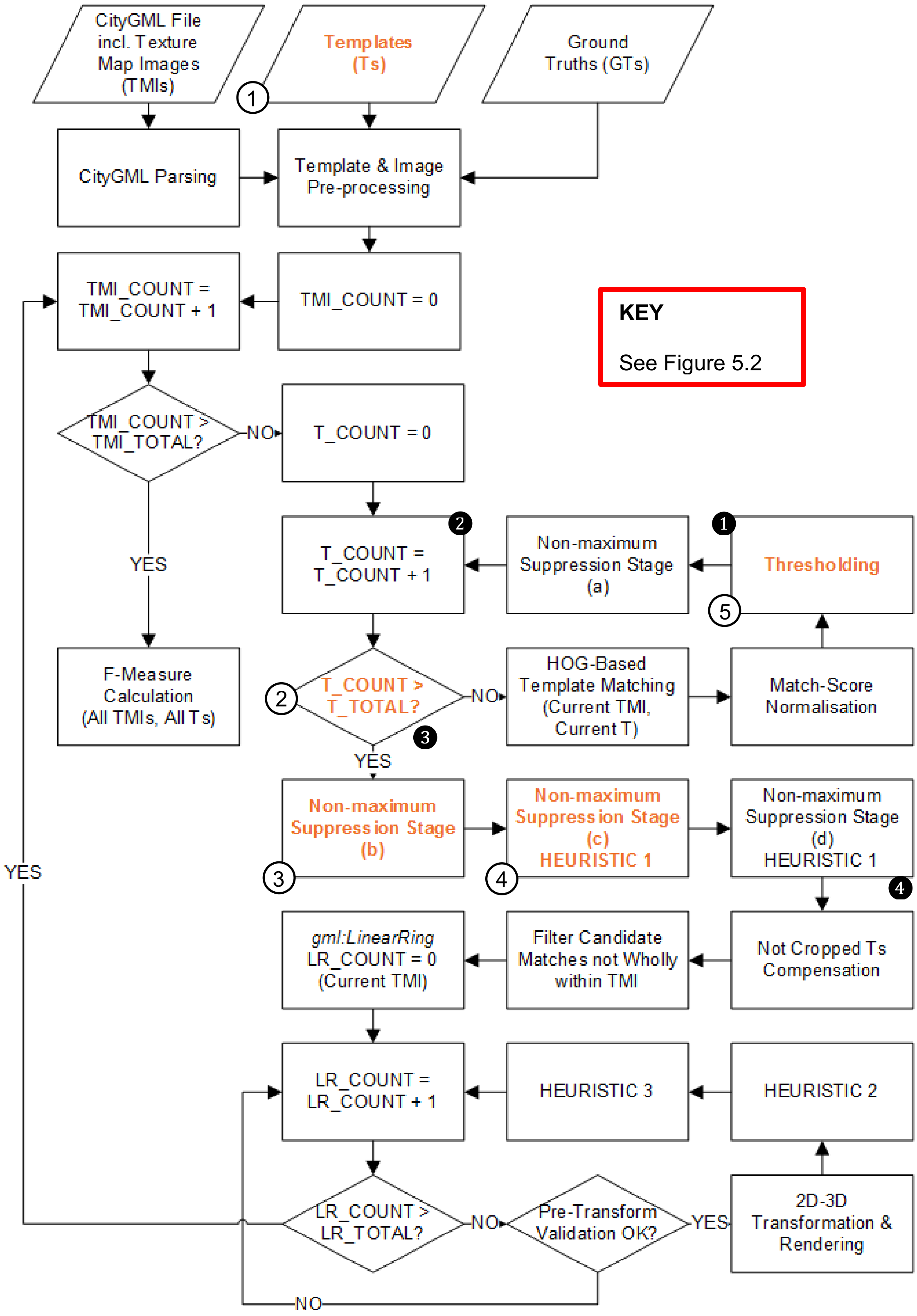

CityGML File incl. Texture Map Images (TMIs)
Templates (Ts)
Ground Truths (GTs)
1
CityGML Parsing
Template & Image Pre-processing
TMI_COUNT = TMI_COUNT + 1
TMI_COUNT = 0
KEY
See Figure 5.2
TMI_COUNT > TMI_TOTAL?
NO
T_COUNT = 0
YES
2
T_COUNT = T_COUNT + 1
Non-maximum Suppression Stage (a)
1
Thresholding
5
F-Measure Calculation (All TMIs, All Ts)
2
T_COUNT > T_TOTAL?
NO
3
HOG-Based Template Matching (Current TMI, Current T)
Match-Score Normalisation
YES
Non-maximum Suppression Stage (b)
Non-maximum Suppression Stage (c) HEURISTIC 1
Non-maximum Suppression Stage (d) HEURISTIC 1
YES
3
4
4
gml:LinearRing LR_COUNT = 0 (Current TMI)
Filter Candidate Matches not Wholly within TMI
Not Cropped Ts Compensation
LR_COUNT = LR_COUNT + 1
HEURISTIC 3
HEURISTIC 2
LR_COUNT > LR_TOTAL?
NO
Pre-Transform Validation OK?
YES
2D-3D Transformation & Rendering
NO

Figure 5.1: HOG-based Template Matching – Process Flow (no SVM)
Links to Figure 5.4, Figure 5.6, Figure 5.7 and Figure 6.6 in Chapter 6.

KEY

Flow-chart Shapes Containing Text in Orange	Indicates that the process step in question is either linked to or from Figure 6.6 in Chapter 6, or is replaced by a process step in that figure. Use references ⓝ beside each shape to trace the linkage to that figure.
🅝	Indicates that the process step either links to or from Figure 5.6.
TMI	Texture Map Image
T	Template
GT	Ground Truth
LR	*gml:LinearRing*

Figure 5.2: Key for Figure 5.1

dering; (7) validation of geometric accuracy; (8) the heuristics used; (9) class granularity considerations; and (10) a summary of how semantic and geometric enrichment have been enabled.

Figure 5.1 (for which Figure 5.2 is the key) summarises the process flow for the HOG-based template matching approach used in this chapter. The steps are described in detail below. Note that this chapter is the core of the method used in this thesis, and is the method on which Chapter 6 builds. As such, this chapter does not describe the use of an SVM (Chapter 6) or trial of the use of clustering of HOG descriptors for templates (Appendix D).

5.2 CityGML Parsing

The CityGML file (XML) was read into memory and parsed as a DOM in order to extract the content required to process all the *gml:LinearRing*s associated with texture mapped polygons in the CityGML file. The process used is shown in Figure 5.4 and described in more detail below.

Firstly, all *app:ParameterizedTexture* elements were extracted. Figure 5.3 shows an example of CityGML for an *app:ParameterizedTexture* element, according to the CityGML standard (Gröger *et al.*, 2012). CityGML stores all texture map images in a folder (which for Figure 5.3 is a folder called ‘textures’). Each *app:ParameterizedTexture* represents a single texture map image used as a front or rear side of a face, where the texture map image is referenced by the element *app:imageURI*. URI is an abbreviation for Uniform Resource Identifier. For Figure 5.3 *app:imageURI* corresponds to texture map image ‘TextureMapImage1.jpg’. In this way an *app:imageURI* can be repeated once i.e. representing (1) a front, and (2) a rear texture. Rear textures have an additional element *app:isFront*, which has a value of ‘false’.

```
<app:ParameterizedTexture>
      <app:imageURI>textures/TextureMapImage1.jpg</app:imageURI>
      <app:wrapMode>wrap</app:wrapMode>
      <app:target uri="#_Filename_BD.BuildingName_PG.Polygon1">
            <app:TexCoordList>
                  <app:textureCoordinates ring=
                  "#_Filename_BD.BuildingName_PG.Polygon1_LR.LR1">
                        2.0 0.0
                        4.0 0.0
                        4.0 1.0
                        2.0 1.0
                        2.0 0.0
                  </app:textureCoordinates>
            </app:TexCoordList>
      </app:target>
      …
<app:ParameterizedTexture>
```

Figure 5.3: CityGML *app:ParameterizedTexture* Example
Note: multiple *app:target*s can use the same *app:imageURI* within the same *app:ParameterizedTexture*. This example is for front textures only. Note, for this example: *uri* corresponds to *gml:id* for *gml:Polygon* in Figure 5.5; *ring* corresponds to *gml:id* for *gml:LinearRing* in Figure 5.5; the number of decimal places for the *app:textureCoordinate* coordinates has been reduced for illustration purposes only. Each pair of coordinates is referred to as (s,t).

The *ring* URI for *app:textureCoordinates* is the URI for the *gml:LinearRing* onto which the texture map image will be mapped. Each *app:ParameterizedTexture* only applies to a single texture map image (*app:imageURI*) but that texture map image can be used on more than one *gml:LinearRing* i.e. there can be more than one *app:target* element for each *app:ParameterizedTexture*. The method used here only worked with the texture map wrap modes of ‘wrap’ or ‘none’ and not the remaining three modes that the CityGML standard allows (see Chapter 2). However, in practice all CityGML used in this work used ‘wrap’, or occasionally ‘none’, for texture mapping. To reiterate, *app:wrapMode* and *app:textureCoordinates* define the nature of texture wrapping and the texture coordinates used to apply the texture mapping, respectively. Also, note that, as per Chapter 2, each pair of coordinates in *app:textureCoordinates* is referred to as (s,t).

The steps carried out to convert to CityGML from the *SketchUp* format (see Chapter 3) included the removal of all textures on the rear of faces. Due to the complexity of some of the 3D building models, any *app:ParameterizedTexture*s that represented texture on rear faces were filtered out, as a precaution and to ensure the method was scalable to other 3D building models.

All remaining *app:textureCoordinates* plus the associated 3D real-world coordinates for the corresponding *gml:posList* were then extracted for all remaining *gml:LinearRing*s. Consequently, only *gml:LinearRing*s that were texture mapped were processed. For the 3D building models used for HOG-based template matching, a total of 87% of *gml:LinearRing*s were texture mapped. The real-world 3D coordinates in *gml:posList* were in the Cartesian coordinate reference system EPSG:32630 (see Chapter 3).

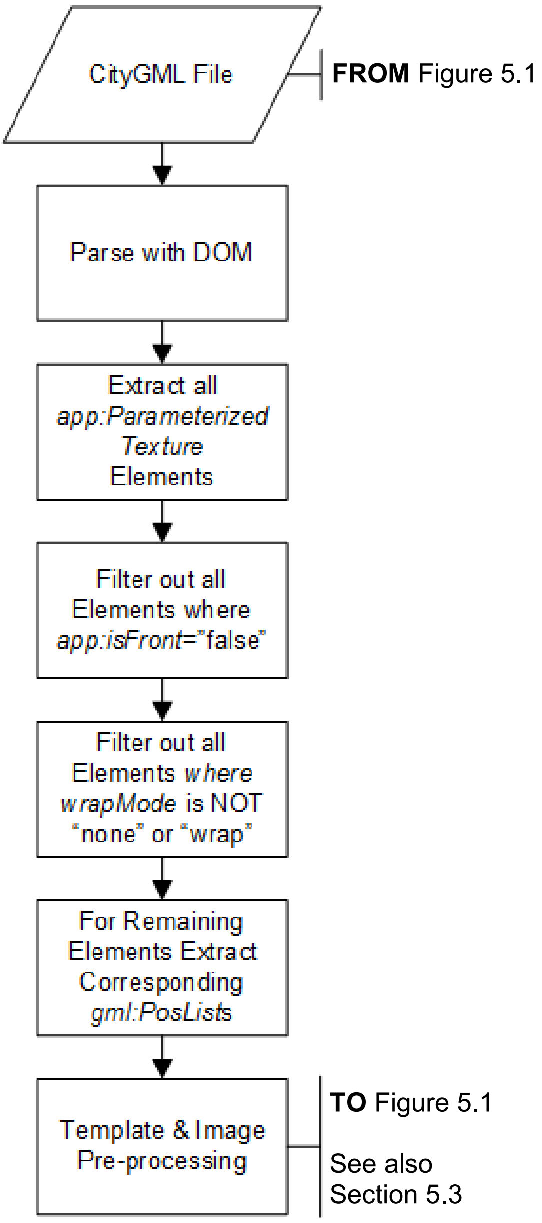

CityGML File
FROM Figure 5.1
Parse with DOM
Extract all app:Parameterized Texture Elements
Filter out all Elements where app:isFront="false"
Filter out all Elements where wrapMode is NOT "none" or "wrap"
For Remaining Elements Extract Corresponding gml:PosLists
Template & Image Pre-processing
TO Figure 5.1
See also Section 5.3

Figure 5.4: CityGML Parsing – Process Flow For linkage to overall process flow see Figure 5.1. The above shows additional process steps not shown in that figure.

Figure 5.5 shows an example CityGML *gml:Polygon*, including the corresponding *gml:LinearRing* and *gml:posList*. The figure links to the example *app:ParameterizedTexture* in Figure 5.3.

MATLAB allows for numbers with a precision up to 15 digits. Here the precision for *app:textureCoordinates* and *gml:posList* coordinates was set to the maximum possible in MATLAB, while the pre-processing into a CityGML file allowed for a greater number of decimal places. Some precision was therefore lost. Bearing in mind that CityGML LOD3 has an accuracy recommendation of 0.5m, i.e. one decimal place, this was deemed acceptable. The use of 15-digit scaled fixed point numbers was maintained due to the need to preserve precision during the transformation from texture to real-world coordinates, which underpins the method used here – see Section 5.9.

Note that the example in Figure 5.3 and Figure 5.5 is for a *gml:exterior gml:LinearRing*. GML allows for *gml:interior* and *gml:exterior gml:LinearRing*s. Strictly, any candidate matches on the *gml:exterior gml:LinearRing* should be clipped before calculating F-measure, by using the *gml:interior gml:LinearRing* as the clipping region i.e. as a 'cookie cutter'. Without doing so, window or door candidate matches or ground truths might be double-counted. Specifically, if a candidate match was a true positive and occurred on both *gml:interior* and *gml:exterior gml:LinearRing*s, the candidate match would be double-counted *and* the corresponding ground truths might also be double-counted. Subject to the relative positioning of the *gml:interior* and *gml:exterior gml:LinearRing*s the net effect on F-measure could therefore be zero. When a candidate match was a false positive and double-counted, the F-measure could be degraded incorrectly.

```
<gml:Polygon gml:id="_Filename_BD.BuildingName_PG.Polygon1">
      <gml:exterior>
            <gml:LinearRing gml:id=
                  "_Filename_BD.BuildingName_PG.Polygon1_LR.LR1">
                  <gml:posList srsDimension="3">
                        501514.13 5955056.18 -0.46
                        501510.88 5955065.32 -0.46
                        501510.88 5955065.32 5.21
                        501514.13 5955056.18 5.21
                        501514.13 5955056.18 0.46
                  </gml:posList>
            </gml:LinearRing>
      </gml:exterior>
</gml:Polygon>
```

Figure 5.5: CityGML Polygon Example
gml:id for *gml:Polygon* corresponds to *uri* in Figure 5.3; *gml:id* for *gml:LinearRing* corresponds to *ring* in Figure 5.3. *gml:LinearRing* is a *gml:exterior*. The number of decimal places for the *gml:posList* coordinates has been reduced for illustration purposes only. Each row of *gml:posList* (x,y,z) coordinates uses the EPSG:32630 (URN 30N) coordinate reference system.

For the 3D building models used for HOG-based template matching a total of 2% of the *gml:LinearRing*s were *gml:interior*s. Based on the small proportion of *gml:interior*s in the models processed it was chosen to treat *gml:interior* and *gml:exterior gml:LinearRing* in the same manner. Future work might include the elimination of one set of candidate match vertices by only keeping the set that corresponded to the *gml:interior gml:LinearRing*. See Section 5.8 and Section 5.11 for more detail on the rendering of candidate matches onto *gml:LinearRing*s and the F-measure calculation approach, respectively.

The final step during the CityGML parse was to find the lowest real-world z-coordinate in preparation for the operation of heuristic 3 – see Section 5.11.

5.3 Template & Image Pre-Processing Steps

Having determined the test images (texture map images) to be used from the CityGML parse, the size of each test image was doubled in order to deal with small instances of windows or doors. Doing so effectively decreased the size of the template at the outset, though only in relative terms i.e. without any of the degradation in quality that resulted from actually decreasing the size of the template. Note once again that during early experiments it was determined that at least one of a template's width or height needed to be at least 75 pixels.

Early experiments included the pre-processing of templates and texture map images into (pp1) coherent line drawings or (pp2) distance transforms, both with and without (pp3) Gaussian blur. However, compared to the results when using images that had not had (pp1), (pp2) or (pp3) applied, the F-measure degraded. As a result, none of these pre-processing steps was used in later experiments. HOG-based template matching was more successful than standard template matching and was most successful with images that had *not* had (pp1), (pp2) or (pp3) applied. The reason for the improved performance was likely due to the reduced sensitivity of the HOG descriptor to the precise location and appearance of the particular features of an object, or to variations in lighting. In turn, this meant that test image objects were more likely to be matched when they were compressed, stretched by a small amount, or where their component parts were distributed with differing proportions, when compared to the template. Example component parts might include glazing panels. Such variety in the make-up of component parts is likely to occur in the real world, when one considers the diversity of building methods and architectural design. In the case of the HOG descriptor, the abovementioned pre-processing steps, which were intended to emphasise line coherency and link disjoint line segments, were not only less successful, but reduced the chances of a successful match occurring.

5.4 HOG Descriptor Density

In spite of the advantages of using HOG-based rather than standard template matching, early experiments demonstrated that identifying very fine detail in Norman or Gothic window or doors objects could sometimes still be a challenge for the method. The HOG descriptor was created with histograms formed from oriented gradients in 8×8-pixel cells across the image. The resultant reduction in sensitivity to precise object-component placement was an advantage when attempting to use a template that was similar, but not the same, as an instance on a test image. A disadvantage was that the detection of fine detail could be lost in the HOG descriptor abstraction. Consider, for example, that when using the 8×8 gridded descriptor the slight difference between a subtly pointed arch of a Gothic window and a rounded arch of a Norman window runs the risk of being lost in the HOG descriptor abstraction.

As a result, a more densely gridded version of the original, higher-dimensionality Dalal and Triggs (2005) descriptor was initially trialled. However, the processing time needed to form such a dense descriptor proved unworkable, taking up to a day to form a descriptor for a single, admittedly large, example texture map image. This result was achieved on a virtual machine running a single processor from an Intel® *Xeon*® E5-1620 3.6GHz host processor, and 24GB of RAM from the host. While more processing power and memory could be obtained, the benefits of using a denser grid will be a trade-off:

Reference	Source	Source Resized by %	Width x Height (Pixels)	Area (Pixels)	Maximum Possible Score (no normalisation)
T1	TD_GR_S02_C	50%	88×122	10736	54.17
T2	TD_GR_S02_C	200%	352×488	171776	1174.39
T3	TW_GR_3x5E	50%	81×141	11421	49.56
T4	TW_GR_3x5E	200%	324×562	182088	986.76

Table 5.1: Normalisation Scheme Trials – Templates

a denser grid provides greater capability for discerning detail, but an increased sensitivity to positioning of component parts of an object. The consequence of the latter would mean that matching a similar but not identical template could become more challenging, when compared to the use of a descriptor formed on a less dense grid. Consequently, the approach chosen was to proceed with the 8×8-pixel grid 31D descriptor variant (as discussed in Chapter 2).

5.5 Match-Score Normalisation

To allow comparison of match-scores achieved with different templates, match-scores were normalised. Initially, two normalisation schemes were trialled: (1) divide the match-score by the maximum possible match-score achievable with the template that resulted in the candidate match; and (2) divide the match-score by the area of the template (in pixels) that produced the candidate match. For (2) the normalised score was multiplied by 100 to obtain a more useable value. To perform the trial of these two schemes the templates in Table 5.1 were used. The trials comprised running HOG-based template matching by matching a template with a template, including matching a template with itself, at two scales. Following normalisation, in theory, matching a template with itself, regardless of scale, should have given the same match-score for any combination of scales (aside from those cases where the template was bigger than the 'image', in which case the match-score should have been zero).

Table 5.2 shows the results of the trial of normalisation scheme (1). The rows in green should have resulted in a score of 1, which, to within 0.03, is the case. Table 5.3 shows the results of the trial of normalisation scheme (2). The order of template and image combinations is the same as in Table 5.2. Unlike the equivalent values in Table 5.2, the normalised match-scores in Table 5.3, i.e. for (2), are not in the [0,1] range. A non [0,1] range of data can still, in theory, be normalised. In Table 5.3 the (a) 'T1 and T1' and (b) 'T2 and T1' image and template combinations result in the same match-score (as expected), as do the (c) 'T3 and T3' and (d) 'T4 and T3' image template combinations.

Image	Template	Non-normalised Match-Score	Maximum Possible Match-Score	Normalised Score
T1	T1	53.40	54.17	0.99
T1	T2	0.00	1174.39	0.00
T1	T3	22.29	49.56	0.45
T1	T4	0.00	986.76	0.00
T2	T1	52.60	54.17	0.97
T2	T2	1164.61	1174.39	0.99
T2	T3	23.02	49.56	0.46
T2	T4	281.30	986.76	0.29
T3	T1	27.21	54.17	0.50
T3	T2	0.00	1174.39	0.00
T3	T3	49.49	49.56	1.00
T3	T4	0.00	986.76	0.00
T4	T1	26.82	54.17	0.50
T4	T2	425.22	1174.39	0.36
T4	T3	49.41	49.56	1.00
T4	T4	974.20	986.76	0.99

Table 5.2: Normalisation Scheme Trial (1) – Maximum Possible Score – Trial Outcome
See Table 5.1 for details of the 'Image' and 'Template' used. Normalisation has been achieved by dividing the non-normalised match-score by the maximum possible match-score. Rows in **green** show a normalised score, following HOG-based template matching using the template as both template and image i.e. the normalised score should be in the range [0,1]. Rows in grey correctly result in a match-score of zero (the template is larger than the image).

Image	Template	Non-normalised Match-Score	Template Area (Pixels)	Normalised Score
T1	T1	53.40	10736	0.50
T1	T2	0.00	171776	0.00
T1	T3	22.29	11421	0.20
T1	T4	0.00	182088	0.00
T2	T1	52.60	10736	0.49
T2	T2	1164.61	171776	0.68
T2	T3	23.02	11421	0.20
T2	T4	281.30	182088	0.15
T3	T1	27.21	10736	0.25
T3	T2	0.00	171776	0.00
T3	T3	49.49	11421	0.43
T3	T4	0.00	182088	0.00
T4	T1	26.82	10736	0.25
T4	T2	425.22	171776	0.25
T4	T3	49.41	11421	0.43
T4	T4	974.20	182088	0.54

Table 5.3: Normalisation Scheme Trial (2) – Template Area (x100) – Trial Outcome
See Table 5.1 for details of the 'Image' and 'Template' used. Normalisation has been achieved by dividing the non-normalised match-score by template area, then multiplying by 100 to generate a more useable value. Rows in **green** correspond to rows in green in Table 5.2. Rows in grey correctly result in a match-score of zero (the template is larger than the image).

However, the match-scores for (a)-(d) are not all the same. Additionally, the ‘T2 and T2’ and ‘T4 and T4’ image and template combinations have different match-scores to each other and to (a)-(d). The conclusion drawn from this was that normalisation scheme (2) gave preference to larger templates – see Table 5.1 and note that the larger the template size, the higher the normalised match-score.

Based on the results in Table 5.2 and Table 5.3, one might consider scheme (1) should be the most effective. After all, it was normalising the match-scores in consistent fashion, i.e. in the range [0,1], while scheme (2) was not.

To test the hypothesis that (1) was the better approach, the trial was extended, as follows. For each normalisation schemes (1) and (2) the HOG-based template matching machine learning approach (see Chapter 6) was carried out. The data used comprised all seven of the Georgian-Regency 3D building models, and all the Georgian-Regency templates. At class granularity 1 and 2, normalisation scheme (1) resulted in a mean F-measure of 0.39, while normalisation scheme (2) resulted in a mean F-measure of 0.86. These results were achieved using all heuristics (see Section 5.11 for a description of the heuristics used).

Experiments were also conducted with normalisation scheme (1) and the removal of heuristic 1, but this made the match-scores worse still. So, in practice, granting more weight to candidate matches achieved with larger templates was preferable for the HOG-based template matching approach used here. Consequently, normalisation scheme (2) was chosen for both the non-machine learning approach (this chapter) and the machine learning approach (Chapter 6).

5.6 Threshold Setting & Thresholding

In advance of running HOG-based template matching, thresholds were tuned empirically (see below), using the following data. Note that the number of instances of windows and doors used to tune the thresholds for each style was approximately proportional to the number of unique classes of templates for a style. For example, if the ratio of Georgian-Regency template classes to Gothic template classes was 2:1, then approximately twice as many Georgian-Regency window and door instances as Gothic instances would be used to tune the thresholds.

- For C20 templates the texture map images from a 3D building model not processed elsewhere in this work were used – see Table 5.4 for details of the model in question. The model comprised 75 texture map images, representing nine properties in a terraced block with a total of 67 windows and doors.
- For Georgian-Regency templates, early experiments used façade images from the standard template matching trial. Three façade images were used: FI_2, FI_3 and FI_4. In order to obtain a similar total number of window and door instances across the three building styles in the images uses for tuning, FI_5 was not used. Together these three façade images comprised 118 windows and doors. A larger number of instances was used compared to the other styles to account for design variation at class granularity 3. To recap, that granularity was not used for the other classes.
- For Norman and Gothic templates, due to the dearth of suitable available data and the challenges with respect to determining architectural style, the test 3D building models were used to tune thresholds in advance. This therefore comprised 76 windows and doors.

Additionally, ground truth data (rectangular ground truths) was marked up. This then allowed F-measure to be used in order to tune the threshold value empirically. The thresholds were tuned as follows. Using the ground-truthed images as validation data, for each threshold at granularity 2, the full collection of templates for the granularity 2 class were used to run HOG-based template matching, with no thresholding in place. The maximum and minimum match-scores were then determined from those results. Next, HOG-based template matching was executed with the same data, in an iterative fashion, with the threshold set in the range [min match-score, max match-score] at increments of 0.1. In this manner, by using F-measure at granularity 2, the optimal threshold-range to the second decimal place was determined e.g. [0.2, 0.3], i.e. higher F-measure scores informed the decision on the optimal range. The iterative process was then repeated in that range, in increments of 0.01, resulting in a value for the granularity 2 class threshold.

On the basis that thresholds were applied at class granularity 2, there were eight different thresholds, where each window and each door of each style had a threshold value, making 2×4=8 thresholds. Thresholds were split in this manner to reflect what appeared to be differing levels of variability between windows and doors and styles. The thresholds were set as shown in Table 5.5. As the table notes, in practice all thresholds, barring those for the Georgian-Regency style, were given the same values. The visual perception of inconsistent variability between windows and doors and styles was therefore not generally reflected in the need to differentiate between classes when thresholding.

Reference	Building	Date Built	Num. of Property Units	Num. Texture Map Images	Num. Windows & Doors
BM_2_13_V	93, 95, 97, 101, 103, 105 & 107 Westward Road & 1, 3, 5 & 7 Hilly Orchard, Stroud, Gloucestershire	2002	9	75	67

Table 5.4: C20 Templates Threshold-Setting – Tuning Data

Template Class (Granularity 2)	Match-Score Threshold
C20 Style Door	0.26
C20 Style Window	0.26
Georgian-Regency Style Door	0.3
Georgian-Regency Style Window	0.3
Gothic Style Door	0.26
Gothic Style Window	0.26
Norman Style Door	0.26
Norman Style Window	0.26

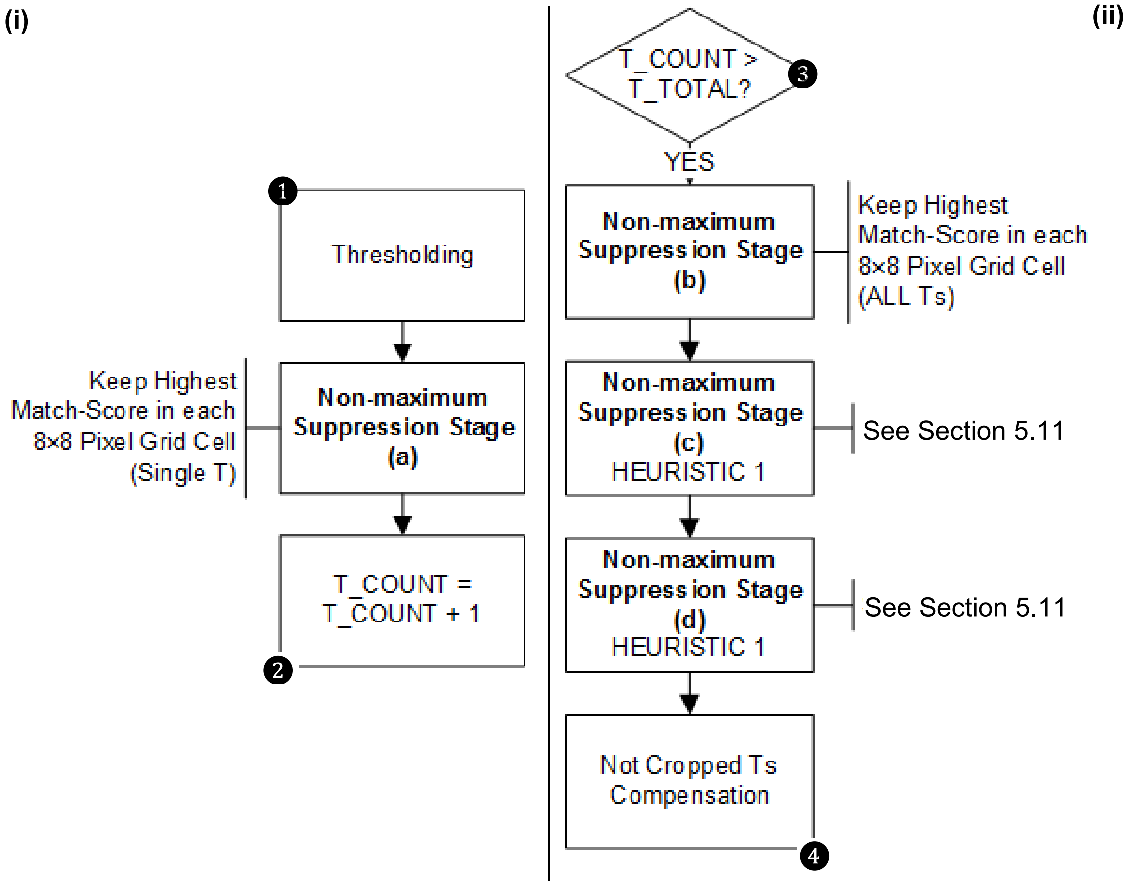
Table 5.5: Match-Score Thresholds (no SVM)

Having run HOG-based template matching to obtain match-scores across the test image, the thresholds were then applied.

5.7 Non-maximum Suppression (NMS)

Following thresholding, the non-maximum suppression approach then operated in three stages, as follows: (a) on candidate matches for a single template for all levels of the image pyramid combined, to ensure that only one match-score existed for each 8×8 pixel grid square in the test image; (b) on candidate matches for all templates and for all levels of the image pyramid, to once again ensure that only one candidate match existed for each 8×8 pixel grid square in the test image; and (c) on remaining candidate matches, using the NMS-region approach from (1) and (3) in the standard template matching trial (see Chapter 4). Note that step (b) was removed for machine learning (see Chapter 6) on the basis that the feature vector used the match-scores for all templates at a location. The process flow for non-maximum suppression is shown in Figure 5.6. Note that heuristic 1 extended non-maximum suppression stage (c) further, and also added a final stage, (d).

Note that a drawback of running non-maximum suppression for candidate matches achieved with multiple templates was that no consideration of threshold was undertaken during non-maximum suppression. Should the match-scores have been consistently normalised in advance of non-maximum suppression then this would not, theoretically, have been an issue. However, the choice of normalisation scheme meant that preference was given to larger templates during non-maximum suppression (see also the comments in Section 5.6). Note once again the comments regarding the early trial with the other normalisation scheme and the corresponding degradation of mean F-measure.


(i)
(ii)
1
Thresholding
Keep Highest Match-Score in each 8×8 Pixel Grid Cell (Single T)
Non-maximum Suppression Stage (a)
T_COUNT = T_COUNT + 1
2
T_COUNT > T_TOTAL?
3
YES
Non-maximum Suppression Stage (b)
Keep Highest Match-Score in each 8×8 Pixel Grid Cell (ALL Ts)
Non-maximum Suppression Stage (c) HEURISTIC 1
See Section 5.11
Non-maximum Suppression Stage (d) HEURISTIC 1
See Section 5.11
Not Cropped Ts Compensation
4

KEY

T	Template
ⓝ	Indicates that the process step either links to or from Figure 5.1.

Figure 5.6: Non-maximum Suppression (HOG-based Template Matching) – Process Flow
Note that the above figure is an annotated version of sub-sections of the process-flow in Figure 5.1. (*i*) Non-maximum suppression stage (a). (*ii*) Non-maximum suppression stages (b), (c) and (d).

5.8 Compensation for Not-Cropped Templates & Texture Map Image Boundaries

Following non-maximum suppression, candidate matches that resulted from 'not-cropped' templates were automatically cropped, proportionally, to the size of the corresponding cropped template. This ensured that candidate matches were as tightly cropped as possible to the window or door instances on the test images. In addition, any candidate matches not wholly contained within the boundaries of the texture map image were also removed.

5.9 2D-3D Transformation & Rendering

Having obtained, and non-maximally suppressed, candidate matches for all templates for a texture map image, the following steps were undertaken. For each *gml:LinearRing* for a texture map image the remaining candidate matches were transformed from 2D pixel coordinates to 2D texture coordinates and then to 3D real-world coordinates. The remaining candidate matches were then rendered onto the *gml:LinearRing*. These steps are summarised in Figure 5.7 and detailed below.

With regard to the level of detail provided below for the 2D-3D transformation and rendering method, note the following. The transformation and rendering in question formed the basis of the geometric enrichment for this work, and was therefore of critical importance to one of the two principle aims (semantic and geometric enrichment) of the project. Essentially, the method results in the calculation of the world-space coordinates for the vertices of candidate matches. Consequently, the methods used are given extensive coverage below, including validation steps to check the quality of the geometric enrichment.

5.9.1 2D-3D Transformation Parameters

To reiterate, the CityGML standard states that all surfaces must be planar. The method used here for finding the transformation parameters therefore assumed planarity. The derivation of 2D-3D transformation parameters was achieved with the following steps, including a number of 2D-2D transformations.

Firstly, the number of columns and rows of grid cells (tiles) over which the texture map image was repeated (tiled), for the current *gml:LinearRing*, was obtained using the corresponding *app:textureCoordinates*. Note that this represented whole grid cells, not fractions of grid cells at the edges of the texture map region. As such, the grid represents texture space for the *gml:LinearRing* but before any masking due to *app:textureCoordinates* was carried out (see Section 5.9.2). The calculation of the number of texture-space grid columns and grid rows for a *gml:LinearRing* was achieved using Equation 5.1 and Equation 5.2 respectively. s_{max} and s_{min} represent the maximum and minimum s value in *app:textureCoordinates* for the *gml:LinearRing*, and t_{max} and t_{min} are the equivalent for t. Note that when s_{max} and s_{min} were both at least half way along one grid cell of texture space, e.g. with values of 0.5 and 0.6 respectively, Equation 5.1 or Equation 5.2 could result in a value of zero. In this case the number of columns or rows was set to 1.

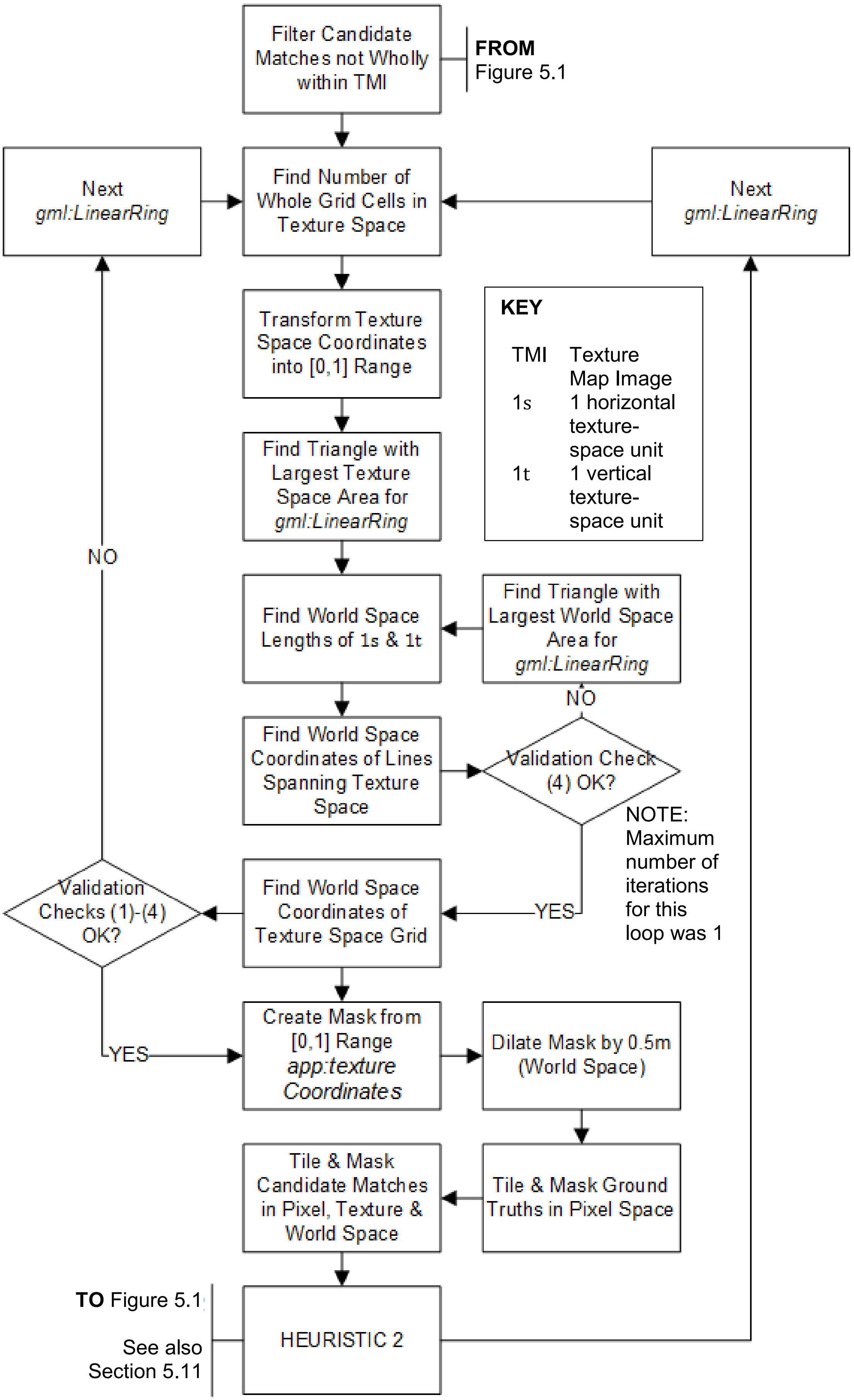

Filter Candidate Matches not Wholly within TMI
FROM Figure 5.1
Next gml:LinearRing
Find Number of Whole Grid Cells in Texture Space
Next gml:LinearRing
Transform Texture Space Coordinates into [0,1] Range
KEY
TMI Texture Map Image
1s 1 horizontal texture-space unit
1t 1 vertical texture-space unit
Find Triangle with Largest Texture Space Area for gml:LinearRing
NO
Find World Space Lengths of 1s & 1t
Find Triangle with Largest World Space Area for gml:LinearRing
NO
Find World Space Coordinates of Lines Spanning Texture Space
Validation Check (4) OK?
NOTE: Maximum number of iterations for this loop was 1
Validation Checks (1)-(4) OK?
Find World Space Coordinates of Texture Space Grid
YES
YES
Create Mask from [0,1] Range app:texture Coordinates
Dilate Mask by 0.5m (World Space)
Tile & Mask Candidate Matches in Pixel, Texture & World Space
Tile & Mask Ground Truths in Pixel Space
TO Figure 5.1
See also Section 5.11
HEURISTIC 2

Figure 5.7: 2D-3D Transformation & Rendering – Process Flow (Summary)
See Figure 5.1 for process linkage. The above shows additional process steps not shown in that figure.

$$Columns_{LR\,TS\,[0,1]} = |\lceil s_{max} \rceil - \lfloor s_{min} \rfloor|$$ **Equation 5.1**

$$Rows_{LR\,TS\,[0,1]} = |\lceil t_{max} \rceil - \lfloor t_{min} \rfloor|$$ **Equation 5.2**

app:textureCoordinates for the current *gml:LinearRing* were then transformed into the [0,1] range, using Equation 5.3 and Equation 5.4. This allowed for more straightforward interpolation later in the method. Note that, unless mentioned otherwise below, all *app:textureCoordinates* take this transformed form.

$$s_{[0,1]} = \frac{s - \lfloor s_{min} \rfloor}{Columns_{LR\,TS\,[0,1]}}$$ **Equation 5.3**

$$t_{[0,1]} = \frac{t - \lfloor t_{min} \rfloor}{Rows_{LR\,TS\,[0,1]}}$$ **Equation 5.4**

The vertices of the candidate matches, which were in pixels, are in ‘pixel space’. The size of $1\mathrm{s}$ and $1\mathrm{t}$ in pixel space was derived by dividing the length of the grid axes in pixel space by the corresponding length in texture space. The length of the grid axes in pixel units was calculated by multiplying the width of the texture map image by $Columns_{LR\,TS\,[0,1]}$ and the height of the texture map image by $Rows_{LR\,TS\,[0,1]}$ for axis s and t respectively.

The 2D-3D transformation method, was based on the calculation of the real-world (‘world space’) coordinates for the four corners of the grid, and the use of s and t world-space values to then derive world-space coordinates anywhere in that grid. The calculation of the size of $1\mathrm{s}$ and $1\mathrm{t}$ unit in world-space units was achieved as follows.

Firstly, by looping through all possible combinations of *app:textureCoordinates* vertices (using the original values from the CityGML i.e. not the [0,1] range transformation) for the current *gml:LinearRing*, the set of three vertices that formed the triangle with largest area were found. The basis for doing so was that the largest triangle should have provided the highest possible accuracy for the current *gml:LinearRing* in the subsequent calculations. The area calculation was based on Heron’s formula (see Equation 5.5 and Equation 5.6) where the triangle has lengths, a, b and c, and a semi-perimeter m. By expansion Equation 5.7 can be derived (florin, 2010). This was the method used, not Equation 5.5 and Equation 5.6, which simply illustrate the basis of the derivation. The triangle is defined by vertices A, B and C. The texture-space to world-space transformation for the horizontal and vertical axes is not comparably isometric for both axes. However, texture space is isometric, in that Heron’s formula can be applied in texture space.

$$Area = \sqrt{m(m-a)(m-b)(m-c)}$$ **Equation 5.5**

$$m = \frac{a+b+c}{2}$$ **Equation 5.6**

$$Area = \frac{A_x(B_y - C_y) + B_x(C_y - A_y) + C_x(A_y - B_y)}{2}$$ **Equation 5.7**

A series of validation checks were carried out in advance of the 2D-3D transformation (see Section 5.10). If the fourth of those checks failed then world space was used, as follows, to find the three vertices of the *gml:LinearRing* which formed the largest triangle (and if it still failed then the *gml:LinearRing* was not processed). The area of a triangle is commonly given by Equation 5.8, where vectors AB and AC are denoted as $\bar{B}$ and $\bar{C}$ respectively, and $\bar{B} \times \bar{C}$ is the 2D cross-product. Then, by finding the cross-product in real coordinate space in 3D, i.e. in $\mathbb{R}^3$, the area of a triangle in 3D was obtained using Equation 5.9 (Manzoni, 2012).

$$Area = \frac{1}{2}|\bar{B} \times \bar{C}|$$ **Equation 5.8**

$$Area = \frac{1}{2}\sqrt{(x_2 \cdot y_3 - x_3 \cdot y_2)^2 + (x_3 \cdot y_1 - x_1 \cdot y_3)^2 + (x_1 \cdot y_2 - x_2 \cdot y_1)^2}$$ **Equation 5.9**

The size of $1\mathrm{t}$ (and subsequently $1s$) in world space was determined as follows, and as summarised in Figure 5.8. The approach was based on Mann (2016) and the principle that linear interpolation is a ‘weighted average’ (i.e. a weighted mean). Note that for the rest of Section 5.9.1, array and object notation are also used, as opposed to the purely mathematical notation above. The additional notation comprises the use of square braces and dots respectively.

For sake of illustration, assume that the s values for the three vertices A, B and C are defined by the relationship represented in Equation 5.10.

$$\mathrm{B}.s < \mathrm{A}.s < \mathrm{C}.s$$ **Equation 5.10**

Therefore, placing the vertices into an array, sorted by s, results in the relationships shown in Equation 5.11, Equation 5.12 and Equation 5.13.

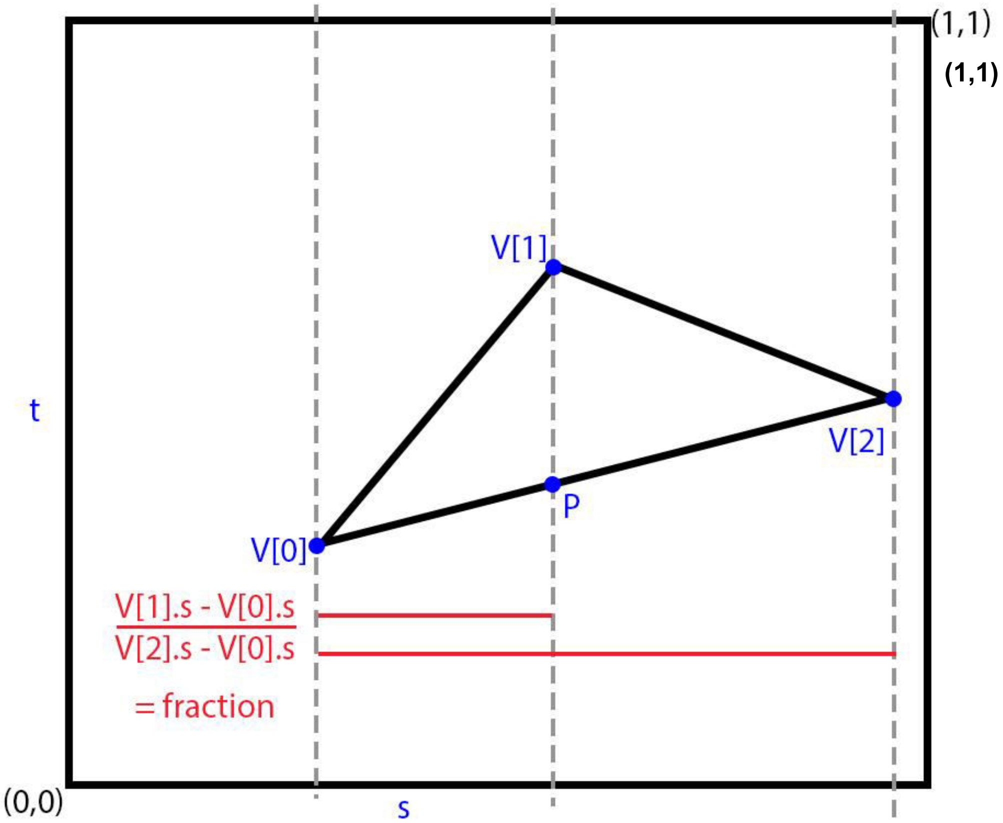


Figure 5.8: Calculation of Length of 1t (Texture Space) in World-Space Units
Having determined the three vertices of the *gml:LinearRing* which form the triangle with the largest area, the vertices were placed into an array, sorted by s, i.e. vertices V[0] ,V[1] ,V[2]. The length between P and V[1] was determined using linear interpolation, using the fraction.
Source: Mann (2016).

$$\mathrm{V[0] = BC}$$ **Equation 5.11**

$$\mathrm{V[1] = AB}$$ **Equation 5.12**

$$\mathrm{V[2] = CA}$$ **Equation 5.13**

The fraction along the edge opposite to V[1], that is where it intersects, projected in the direction of t, was then calculated using Equation 5.14.

$$\mathrm{fraction} \;=\; \frac{\mathrm{V[1].s - V[0].s}}{\mathrm{V[2].s - V[0].s}}$$ **Equation 5.14**

The coordinates of that point on the edge were then calculated using linear interpolation between V[0] and V[2], interpolating all scalars in the object, as shown in Equation 5.15.

$$\mathrm{P = (V[0]\;x\;(1 - fraction)) + (V[2]\;x\;fraction)}$$ **Equation 5.15**

This provided the end points of a vertically aligned span in t, which allowed the length of the span to be obtained using Equation 5.16.

$$\mathrm{dt} = |\mathrm{V}[1].\mathrm{t} - \mathrm{P.t}|$$ **Equation 5.16**

Using 3D Pythagoras, the length of that span was then calculated, denoted dw. This allowed the length of $1\mathrm{t}$ in world space to be calculated using Equation 5.17.

$$1\mathrm{t} = \frac{\mathrm{dw}}{\mathrm{dt}}$$ **Equation 5.17**

The equivalent for $1\mathrm{s}$ was then calculated in a similar manner. If $\mathrm{V}[0]$ and $\mathrm{V}[1]$ were above each other and vertically aligned, or horizontally aligned then $1\mathrm{t}$ or $1\mathrm{s}$ in world space were obtained by dividing dw by the t length or s length of the edge in question. An initial check for this condition was first carried out to reduce possible processing time, noting that some 3D building models have thousands of *gml:LinearRing*s.

Lastly, the coordinates of the four world-space corners of the bounds of the texture-space grid were calculated as follows. The above calculations resulted in two intersecting lines, one aligned vertically in texture space and the other aligned horizontally in texture space. For each of those lines three coordinates were known, in both texture (i.e. in 2D) and world space (i.e. in 3D). At this stage of the pipeline the lines in question were oriented vertically and horizontally in texture space, and the size of the texture-space grid was known. This meant that extending each line until it spanned the texture-space grid provided texture-space coordinates for the ends of each span. Consequently, using interpolation once again, the world-space coordinates of the end points and intersection of the two lines, were calculated. The positions of these points on the texture-space grid is shown in Figure 5.9.

The above process resulted in four tessellating rectangles (formed by the spans, not by the texture-space grid), in a 2×2 arrangement, where each rectangle had three known vertices in both texture space and world space. The world-space coordinates for each of the four missing vertices were then calculated, using the principle that for each rectangle two vectors were parallel, as were the other two. x, y and z for each of the vertices were calculated using Equation 5.18, Equation 5.19 and Equation 5.20 respectively (Vahe, 2014), where W is the missing vertex and T, U and V are the known vertices. The world-space bounds of texture space were then known, where Q, H and K denoted the origin of the texture-space grid (bottom left), the top-left and bottom-right points respectively, as shown in Figure 5.9.

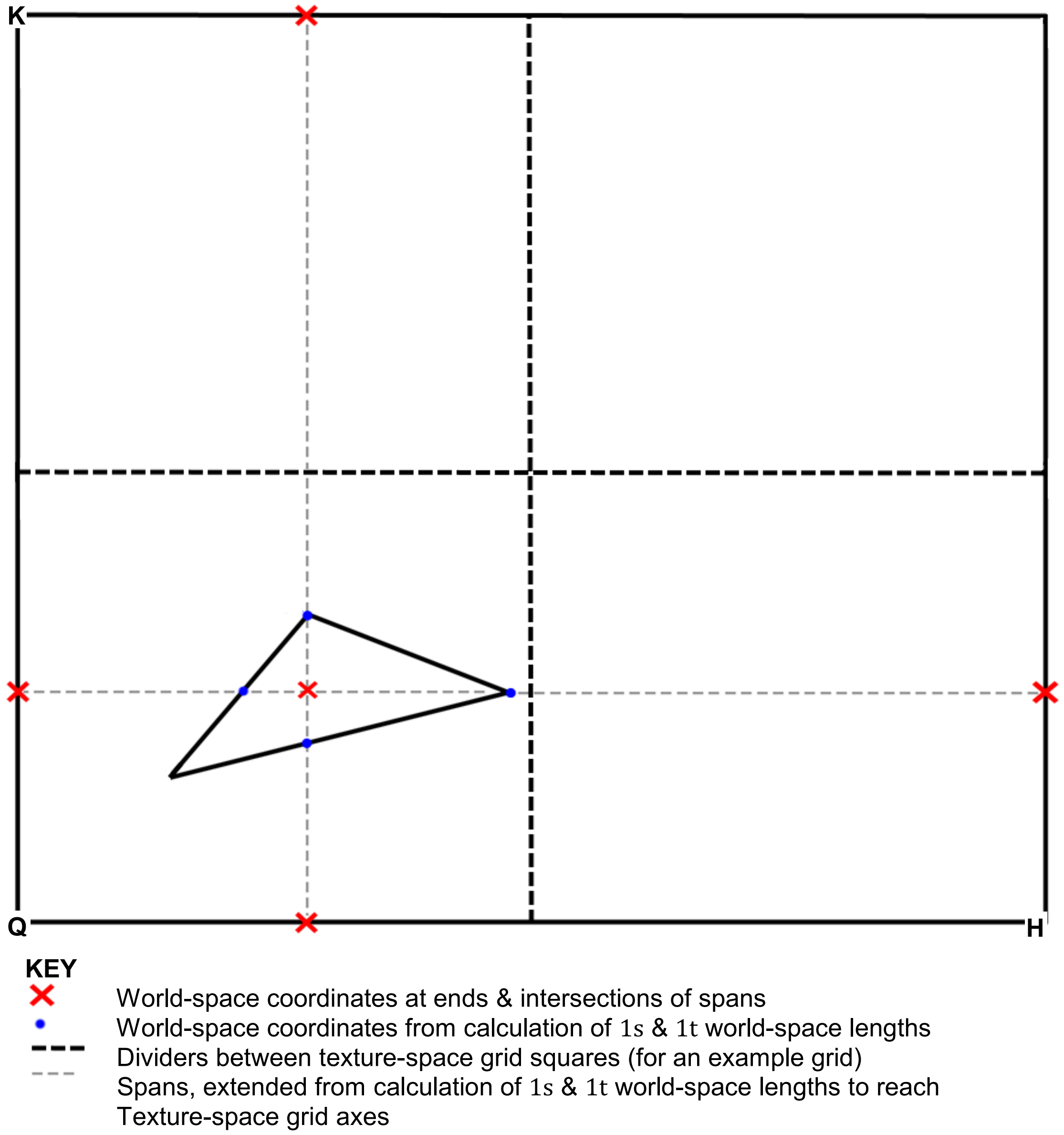

K
Q
H
KEY
✕ World-space coordinates at ends & intersections of spans
● World-space coordinates from calculation of 1s & 1t world-space lengths
▬ ▬ ▬ Dividers between texture-space grid squares (for an example grid)
- - - Spans, extended from calculation of 1s & 1t world-space lengths to reach Texture-space grid axes

Figure 5.9: Calculation of Bounds of Texture Space in World-Space Units
Note the presence of Q, H and K at the corners of the texture-space grid – these are used in the 2D-3D transformation and are obtained using the world-space coordinates at the ends and intersections of the spans. Q, H and K were calculated using Equation 5.18, Equation 5.19 and Equation 5.20 and the world-space coordinates at ends and intersections of spans (above). The triangle is that from Figure 5.8. Note that for the sake of illustration the triangle shown is assumed to be that with the largest area using the *gml:LinearRing* vertices, for the calculation of both $1\mathrm{s}$ and $1\mathrm{t}$ – this may not necessarily be the case with real *gml:LinearRings*. Note also that the number of grid cells shown for texture space is only for illustration purposes – any combination of row and column numbers is possible.

$$x_W = x_T + x_V - x_U$$

Equation 5.18

$$y_W = y_T + y_V - y_U$$

Equation 5.19

$$z_W = z_T + z_V - z_U$$

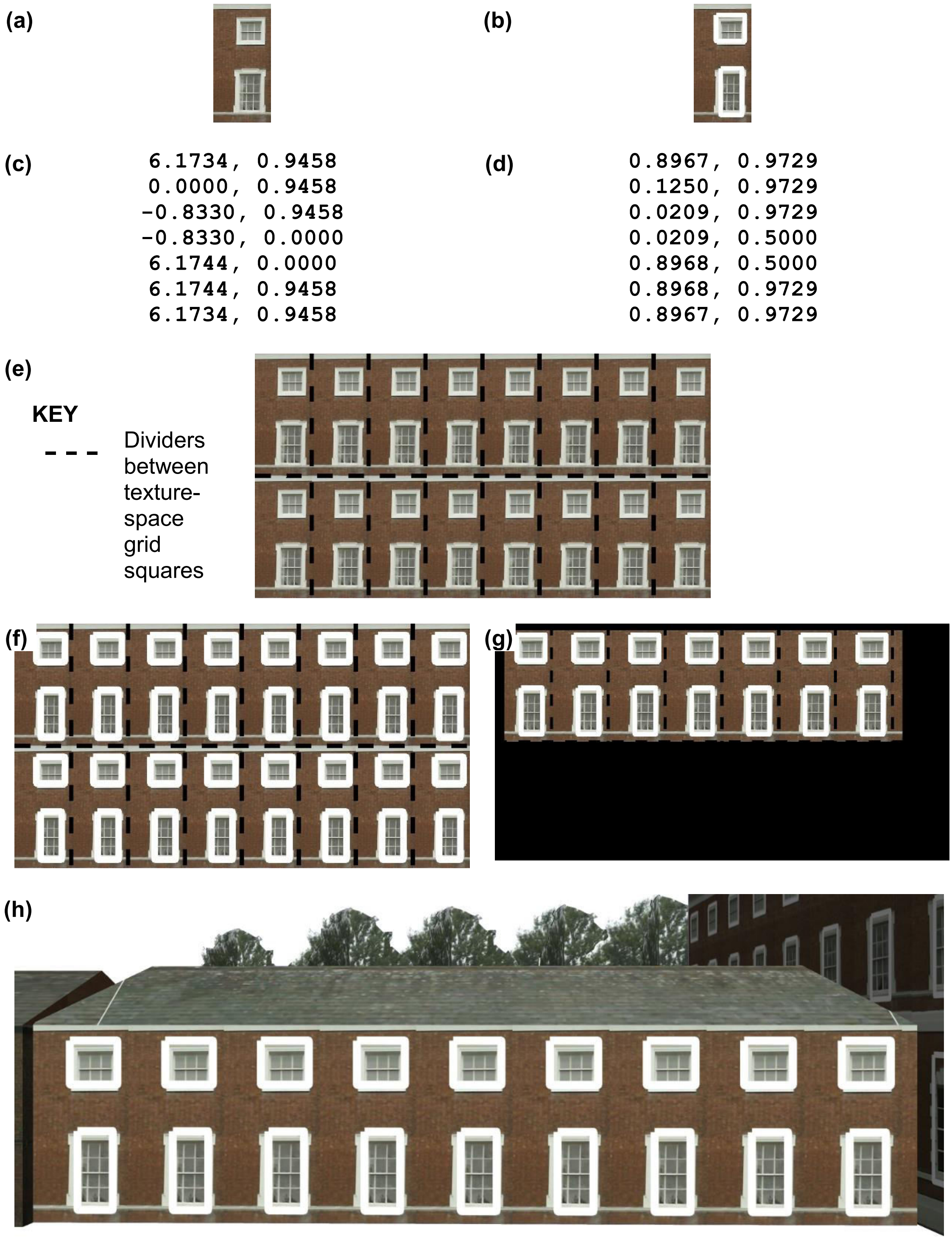
Equation 5.20

5.9.2 Rendering – Candidate Matches & Ground Truths

As a reminder, the candidate matches and ground truths were obtained or recorded, in pixel space, on a texture map image, before any tiling or masking defined by *app:textureCoordinates* and *app:wrapMode* for the current *gml:LinearRing* was carried out. The method for performing that tiling and masking (collectively, 'rendering') is summarised in Figure 5.10 and detailed below.

With the number of columns and rows in the texture-space grid known, the ground truths were tiled across the texture-space grid. The pixel coordinates of the ground truths (where the reference frame was the bounds of the texture map image) were transformed by multiplying x-pixel coordinates by the grid column and y-coordinates by the grid row. The pixel-space coordinates for the candidate matches on the current texture map image were transformed to pixel space for the current *gml:LinearRing* in the same manner. No further transformation of the ground truths was required. This was on the basis that the determination of true and false positives was carried out in pixel space (see Section 5.12).

In order to transform the vertices of the candidate matches into [0,1] range texture space for the current *gml:LinearRing*, two transformations were required. (1) The first transformation step was carried out on the pixel coordinates for the candidate matches on the current texture map image i.e. before any tiling. The transformation was from pixel space to texture space, where the bounds of texture space were defined by the texture map image boundary. This transformation was achieved by dividing the x-direction and y-direction candidate-match coordinates by the pixel width and height of the texture map image, respectively. These candidate-match coordinates were then tiled across each of the grid cells of texture space. (2) The second transformation step was to transform the texture coordinates for the candidate matches from (1) into the [0,1] range for texture space for the current *gml:LinearRing*. This was carried out using Equation 5.21 and Equation 5.22, with input from Equation 5.1 and Equation 5.2. 'CM' equals 'candidate match', 'TMI' equals 'texture map image', 'LR' is '*gml:LinearRing*' and 'TS' is 'texture space'.



(c)

6.1734,	0.9458
0.0000,	0.9458
-0.8330,	0.9458
-0.8330,	0.0000
6.1744,	0.0000
6.1744,	0.9458
6.1734,	0.9458

(d)

0.8967,	0.9729
0.1250,	0.9729
0.0209,	0.9729
0.0209,	0.5000
0.8968,	0.5000
0.8968,	0.9729
0.8967,	0.9729

Figure 5.10: Tiling & Masking of Candidate Matches & Ground Truths
Note that the 3D building models is **BM_GR_3**. (*a*) Texture map image *before* HOG-based template matching. (*b*) Texture map image *after* HOG-based template matching, with candidate matches shown as white bounding boxes. (*c*) *app:textureCoordinates* from a *gml:LinearRing*, before any transformation. Note that a reduced number of decimal places is shown for the purposes of illustration. Also note that the *gml:LinearRing* is (not quite) rectangular hence there being greater than five vertices. (*d*) The *app:textureCoordinates* transformed to the [0,1] range using Equation 5.3 and Equation 5.4. (*e*) The texture map image from (a) tiled across texture space, with the number of tiles determined from (c) using Equation 5.1 and Equation 5.2. (*f*) Candidate matches from (b) tiled across texture space, before any masking. The equivalent was carried out for ground truths. (*g*) Masking applied to candidate matches and ground truths from (f), based on (d), dilated by 0.5m. (*h*) The *gml:LinearRing* in the 3D building model, marked up with candidate matches. Note that terrain model and part of vegetation model have been removed for the purposes of illustration.

$$s_{CM\ Vertex\ LR\ TS\ [0,1]} = \frac{Column_{CM\ LR\ TS\ [0,1]} - 1}{Columns_{LR\ TS\ [0,1]}} + \frac{s_{CM\ Vertex\ TMI\ [0,1]}}{{Columns_{LR\ TS\ [0,1]}}^{2}}$$

Equation 5.21

$$t_{CM\ Vertex\ LR\ TS\ [0,1]} = \frac{Row_{CM\ LR\ TS\ [0,1]} - 1}{Rows_{LR\ TS\ [0,1]}} + \frac{t_{CM\ Vertex\ TMI\ [0,1]}}{{Rows_{LR\ TS\ [0,1]}}^{2}}$$

Equation 5.22

Subsequently, using the *app:textureCoordinates* a mask was created in pixel space, representing the region *outside* of the texture map region. This mask was used to remove any of the tiled candidate-match and ground truth bounding boxes that it overlapped, including those touching the edges of the mask.

In order to deal with *gml:LinearRing* boundaries that were close to or overlapped a candidate match, the mask was first dilated by 0.5m (i.e. the accuracy recommendation for LOD3) before it was applied. The size of 0.5m in pixel units was first determined, using the aforementioned transformations between pixel, texture and world space (but in reverse). The dilation step did introduce the potential for erroneous double-responses or ground truths where the boundary of a *gml:LinearRing* cut through a window or door instance on a texture map image, or where the boundary was especially close to the edge of a window or door instance. Analysis of the results for all the 3D building models was performed and it was determined that the impact of the double-count generally resulted in a lower F-measure, i.e. by leaving the potential double-count in the processing pipeline the overall results were *not* generally being artificially improved.

Heuristics 2 and 3 were then applied (see Section 5.11). The candidate matches and ground truths that remained were carried forward, in pixel space of the texture-space grid, to the F-measure calculation (see Section 5.11).

5.9.3 2D-3D Transformation – Candidate Matches

At this point in the process flow the coordinates of the world-space vertices of the texture-space grid, and of the texture-space vertices for each remaining candidate match, had both been obtained. Consequently, interpolation was used to obtain the world-space coordinates $(\mathrm{x,y,z})$ for the vertices of the candidate matches, using Equation 5.23, Equation 5.24 and Equation 5.25 respectively. $\overline{H}$ and $\overline{K}$ are the vectors for points H and K respectively, with *Q* as the origin. The coordinates of a texture-space candidate-match vertex (P) are denoted as $(\mathrm{u,v})$. The process of interpolation is illustrated in Figure 5.11.

$$P_x = Q_x + u\bar{H}_x + v\bar{K}_x$$ **Equation 5.23**

$$P_y = Q_y + u\bar{H}_y + v\bar{K}_y$$ **Equation 5.24**

$$P_z = Q_z + u\bar{H}_z + v\bar{K}_z$$ **Equation 5.25**

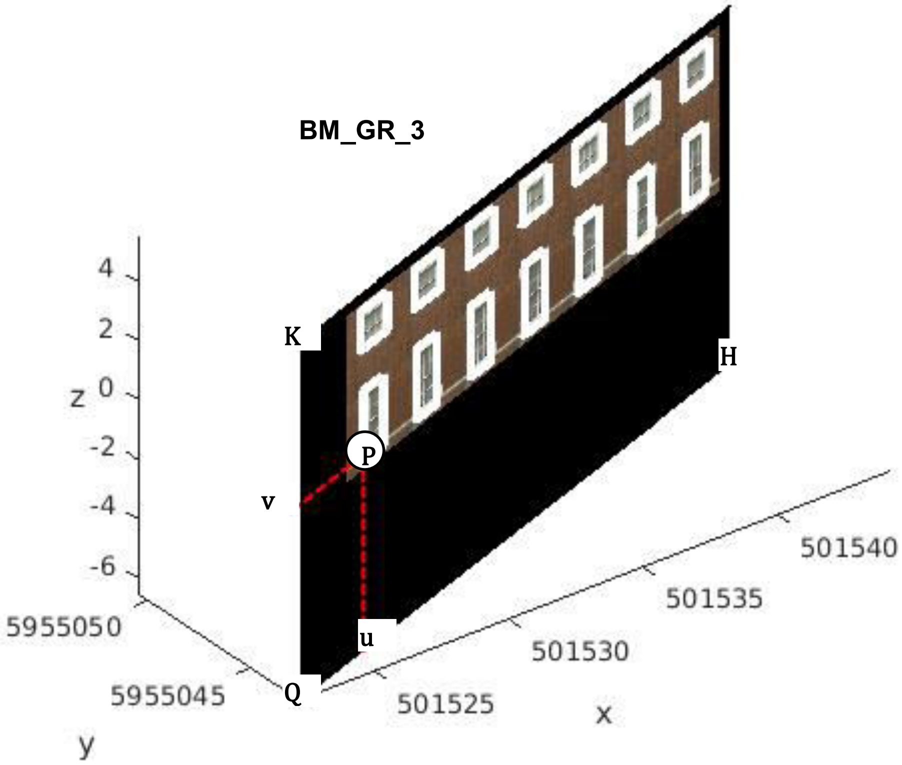


Figure 5.11: Calculation of World-Space Coordinates for Candidate-Match Vertex
Note the presence of Q, H and K at the corners of the texture-space grid, each of which are known in world space. The world-space coordinates of candidate-match vertex P were calculated using Equation 5.23, Equation 5.24 and Equation 5.25, using texture-space coordinates (u,v). The *gml:LinearRing* shown is that from Figure 5.10. (x,y,z) are in the Cartesian geospatial coordinate system EPSG:32630 (URN 30N).

The candidate matches were validated for spatial accuracy qualitatively (visually) in a 3D plotting tool, by projecting the *gml:LinearRing* in EPSG:32630 (URN 30N), applying the texture map to it and then checking that the coordinates of the candidate matches were coplanar with the *gml:LinearRing*. This was achieved by manually rotating a *gml:LinearRing*, marked up with candidate matches, about the z-axis. To visualise this imagine the *gml:LinearRing*, marked up with candidate matches and the axes in Figure 5.11. Specifically, coplanarity was checked when: the viewer attitude was along the edge Q-H, i.e. the azimuth was the angle of Q-H in the x-y plane; the inclination with respect to the x-y plane was zero; and the inclination with respect to the Q-K edge was also zero. An automated coplanarity check could have been used here. However, noting the 0.5m accuracy recommendation for the CityGML standard, this was not deemed necessary.

5.10 Pre-2D-3D Transformation Validation Checks

Four validation checks were performed to ensure that the 2D-3D transformation could be correctly applied. These checks were as follows. (1) A check that the world-space height or world-space width of a texture-space grid cell was greater than 0.5m. (2) A check that the *SketchUp* to CityGML conversion had not resulted in known texture-mapping errors. (3) A check to ensure that the method for finding the world-space length of *1s* and *1t* was solvable. (4) A check to determine the accuracy of the *1s* and *1t* world-space lengths. These four checks are detailed below.

5.10.1 Validation Check (1)

Validation check (1) was achieved by checking that the world-space size of 1s and 1t were both greater than 0.5m. For any *gml:LinearRing*s where this check failed the *gml:LinearRing* was not processed – any candidate matches were likely to be too small to represent real window or door instances. For the 3D building models used in HOG-based template matching, 4% of *gml:LinearRing*s were filtered out by validation check (1).

Prior to choosing this approach for validation check (1), an alternative approach was trialled. This comprised checking whether either the world-space height or the world-space width of the minimum bounding rectangle (MBR) of the *gml:LinearRing* was less than 0.5m. The MBR approach was achieved as follows. Firstly, two 3D vectors (based on the *gml:LinearRing* world-space vertices) with an angle between them that was not zero, were obtained. The rationale for doing so was that it is possible to have a *gml:LinearRing* where more than two vertices are colinear in 3D. The vectors were obtained by finding the cross product of two vectors that were not zero. Secondly, the 3D coordinate system was rotated to a new 2D coordinate reference system. Lastly, the MBR in 2D was calculated using the code provided by Diener (2011). These steps are not described in any more detail here, on the basis that the method was not ultimately used.

However, the MBR approach had a higher processing overhead for each *gml:LinearRing*. It was therefore decided that the texture-space grid-cell size check provided a suitable analogue for the MBR approach. Note too that the approach used would either process the same number of, or more, *gml:LinearRing*s, compared to the MBR approach. This was on the basis that the region being checked for size would either have the same dimensions, if the *gml:LinearRing* filled only one entire texture-space grid cell, or be larger.

5.10.2 Validation Check (2)

During preliminary experiments, the *SketchUp* to CityGML conversion could result in occasional 'not a number' (NaN) or 'Infinity' values for *app:textureCoordinates*. Validation check (2) was performed to check for such occurrences. The validation check comprised searching the *app:textureCoordinates* for a *gml:LinearRing* for NaN or Infinity values. Any *gml:LinearRing*s containing such values were not processed – the method used in this work relied upon the presence of numeric *app:textureCoordinates*. Subsequently, through working with the *CityEditor* vendor, it was identified that there was a bug in the vendor's code. Using the information provided by the author the bug was then corrected by the vendor in a dedicated new public release of the *CityEditor* plug-in (Slade and Buss, 2017). The new release of the plug-in removed the occurrence of the problem for the 3D building models used in this work and was used in the experiments in Chapter 7. The validation check was maintained though, for scalability purposes i.e. for unseen 3D building models.

5.10.3 Validation Check (3)

Validation check (3) was performed on the basis that long, very thin *gml:LinearRing*s could result in a triangle (see Section 5.9) where, due to a shortage of precision, the triangle's vertices became colinear. In such a scenario the method used to find $1s$ and $1t$ in world-space units would not yield the correct result. Consequently, a check was performed to determine if $V[1]$ and P had the same t value (when finding $1t$) or if $V[1]$ and P had the same s value (when finding $1s$). If the check found colinearity then the *gml:LinearRing* was not processed. In effect, the check worked as a proxy for identifying possibly long, very thin *gml:LinearRing*s. As with the filtering of *gml:LinearRing*s where the texture-space grid cell width or length was less than 0.5m, such *gml:LinearRing*s would not have been likely to have contained any candidate matches due to the thinness of the *gml:LinearRing*.

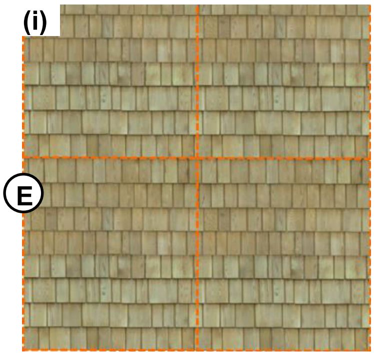
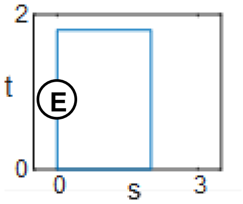
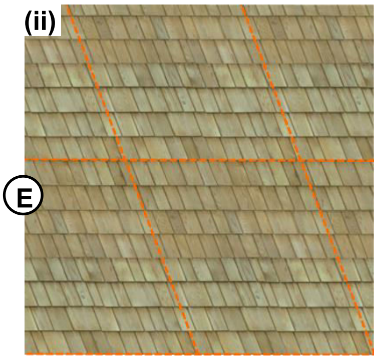
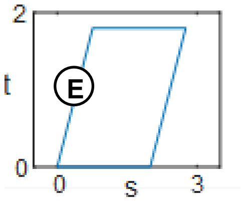
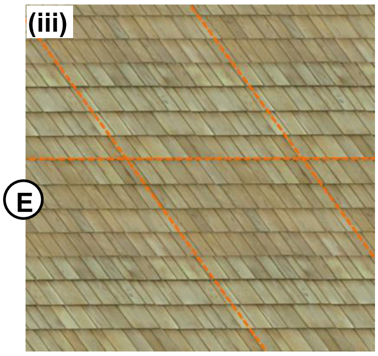
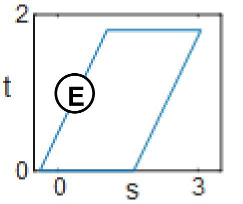
5.10.4 Validation Check (4)

Validation check (4) calculated the lengths of all edges of the current *gml:LinearRing* using two methods, as follows. The first method (a) used the *gml:posList* world-space coordinates using 3D Pythagoras. The second method (b) used the *app:textureCoordinates* texture-space coordinates to find s and t components of each edge length, then multiplied them by $1s$ and $1t$ respectively, followed by 2D Pythagoras to find the edge lengths in world space. If, for any of the *gml:LinearRing*'s edges, the difference between (a) and (b) was greater than 0.5m then the *gml:LinearRing* was not processed.

The application of shear to the texture map, using the *app:textureCoordinates*, can lead to the rejection of a *gml:LinearRing* by validation check (4). Such rejection would be on the basis that the 2D-3D transformation method used here relies on isometric scaling. The difference in lengths (a) and (b) will increase as shear displacement increases. This is because the angle of an edge in texture space will increase with the amount of shear, while the angle for the same edge in world space will remain fixed. Consequently, method (b) will give a different result than method (a) when shear is applied to a texture map image. Note that there will be a threshold of shear displacement beyond which validation check (4) would reject the *gml:LinearRing* i.e. *gml:LinearRing* textures with lesser amounts of shear *will* be processed. The threshold would vary due to the positioning of the vertices of the *gml:LinearRing* in world and texture space. See Figure 5.12 for an illustration of how shear affects the calculation of (b).

The HOG-based template matching approach used here takes place on texture map images, that is before texture mapping. Consequently, any texture map image shearing would also result in the shearing of any candidate matches. With the 2D-3D transformation method used here, this would result in world-space vertices for the candidate matches that were non-rectangular. The templates used were all rectangular, in addition to which it was intended that the candidate matches in world space would also be rectangular. As such, shearing was generally undesired. In practice, visible shear was identified on only four texture mapped *gml:LinearRing*s (<1%) that resulted in candidate matches. Nonetheless, the maintenance of this validation check promoted the scalability of the method to unseen 3D building models.

Note that validation check (4) did not exclusively detect shear. Despite the use of the maximum precision achievable, when very small triangles (not picked up by the other validation checks) were used to find the world-space lengths of $1s$ and $1t$, precision issues could result in a failure of the check. Moreover, the check may also identify errors in texture mapping generated during the *SketchUp* to CityGML conversion. This would

	Texture Mapped *gml:LinearRing* (World Space)	*app:textureCoordinates* (Texture Space)		Validation Check (3) Δ Lengths for Edge Ⓔ
No Shear		2.00, 0.00 2.00, 1.80 0.00, 1.80 0.00, 0.00 2.00, 0.00		0m
Shear Factor ~3s		2.00, 0.00 2.76, 1.80 0.76, 1.80 0.00, 0.00 2.00, 0.00		0.26m
Shear Factor ~6s		1.62, 0.00 3.06, 1.80 1.06, 1.80 -0.38, 0.0 1.62, 0.00		0.82m

KEY

Shear Factor ~ns	Shear Displacement where '*s*' is 1s in texture space
---	Tiling of original texture map image (also shows shear displacement)
Δ Lengths	Delta in Length for Ⓔ calculated with methods (a) (b) – Section 5.10
Ⓔ	Edge. Always vertical in world space, but angle varies in texture space

Figure 5.12: Impact of Shear of Texture Map on Pre-Transformation Validation Check (4) All shear displacement is in the horizontal direction. The world-space dimensions for the *gml:LinearRing*s (*i*), (*ii*) and (*iii*) are *all* 1.9mx1.9m. *gml:LinearRing* (*ii*) would pass the validation check (4) but *gml:LinearRing* (*iii*), with a larger shear displacement, would fail and would not be processed. Note: the number of decimal places shown for the *app:textureCoordinate* coordinates has been reduced for illustration only. Source of image: Slade and Buss (2017).

be despite adequate steps having been taken during conversion, where the texture mapping errors (perhaps of as yet unknown basis) might be a result of the potentially high complexity and variability of CityGML models. On which note, during early experiments the author used validation check (4) to highlight to the developers of the *CityEditor* plug-in the erroneous existence of projective transform during the conversion process for *app:textureCoordinates* (which only allow Affine transformations). As with the NaN and Infinity issue identified above, the bug was therefore fixed, with a further dedicated new release of the plug-in (Slade and Buss, 2017). The revised version of the plug-in, containing both fixes, was used in the experiments in Chapter 7.

Separate to the use of *app:textureCoordinates*, the CityGML standard does allow for projective transform of textures. This can be achieved using the class *TexCoordGen* and a transformation matrix in a *worldToTexture* element. However, relying as it does on isometric texture mapping, the method used in this work does not process such projected textures – indeed no projected textures existed in any of the 3D building models used in this work.

5.10.5 Impact of Validation Checks (3) & (4)

The percentage of *gml:LinearRing*s, for all 3D building models used in the HOG-based template matching experiments, not processed as a result of validation checks (3) and (4) was 1% and less than 1% respectively. These results, including the corresponding number of *gml:LinearRing*s, are summarised in Table 5.6. Validation check (1) is not shown in the table – the reason for its use is seen to be quite straightforward (see above). Validation check (2) is not shown either – no *gml:LinearRing*s failed as a result of the check.

Despite the low percentage of *gml:LinearRing*s that failed to process, the number of failures for some 3D building models was anomalous, as shown in Table 5.7. Of the 25 3D building models used for HOG-based template matching, 15 had some *gml:LinearRing*s that were not processed due to validation check (3) or (4). For four of those 3D building models, greater than 5% (rounded up) of their *gml:LinearRing*s were not processed due to those checks.

Num. Texture-Mapped *gml:Linear Ring*s	**Check (3) Fail Num. *gml:Linear Ring*s**	**Check (3) Fail % *gml:Linear Ring*s**	**Check (4) Fail Num. *gml:Linear Ring*s**	**Check (4) Fail % *gml:Linear Ring*s**
17,433	231	1.33%	58	0.33%

Table 5.6: Total *gml:LinearRing*s Removed by Pre-Transformation Validation Checks (3) & (4) – Total
The total number of *gml:LinearRing*s is based on all 3D building models used in the HOG-based template matching experiments.

As discussed above, the reason for validation check (3) failing was due to *gml:LinearRing*s being long and very thin, and as a result not processing those *gml:LinearRing*s is entirely appropriate. However, the failures due to validation check (4) bear more investigation, not least because the reason for the failure can be varied. A visual inspection of the texture mapped *gml:LinearRing*s that failed the check shows that the majority represent parts of terrain models in the CityGML. The images in question were Google *Earth* aerial images, where the ability to add this imagery as terrain is an automated feature within *SketchUp*. Note that none of the *gml:LinearRing*s that failed the check resulted in any candidate matches. As mentioned above, no visible shear was apparent for any *gml:LinearRing*s that resulted in any candidate matches. For those *gml:LinearRing*s that did not result in any candidate matches, shear *might* still have been present. If it was, the check could have failed as a result. Additionally, the failures could have been due to as yet unknown texture mapping errors resulting from the *SketchUp* to CityGML conversion process. Consequently, the check was maintained.

Reference	**Num. Texture-Mapped *gml: Linear Rings***	**Check (3) Fail Num. *gml: Linear Rings***	**Check (3) Fail % *gml: Linear Rings***	**Check (4) Fail Num. *gml: Linear Rings***	**Check (4) Fail % *gml: Linear Rings***
BM_2_2	66	0	N/A	1	1.52%
BM_2_4	581	1	0.17%	0	N/A
BM_2_6	5098	12	0.24%	1	0.02%
BM_2_11	975	0	N/A	1	0.10%
BM_2_12	**330**	0	N/A	**35**	**10.61%**
BM_GR_1	1830	6	0.33%	0	N/A
BM_GR_2	**1,040**	2	0.19%	**60**	**5.77%**
BM_GR_4	1363	6	0.44%	0	N/A
BM_GR_5	7,052	3	0.04%	2	0.03%
BM_GR_6	272	0	N/A	14	5.15%
BM_GR_7	**478**	5	1.05%	**102**	**21.34%**
BM_N_1	581	5	0.86%	0	N/A
BM_N_2	414	1	0.24%	0	N/A
BM_N_3	483	1	0.21%	2	0.41%
BM_N_4	**143**	**16**	**11.19%**	**13**	**9.09%**

Table 5.7: *gml:LinearRing*s removed by Pre-Transformation Validation Checks (3) & (4) – by 3D Building Model
3D building models where *gml:LinearRing*s were filtered out due to validation checks (3) or (4). No other 3D building models used in this work failed validation check (3) or (4). Items in **red** show where the % of *gml:LinearRing*s which failed the validation check was greater than 5%. Note that validation check (3) is correctly filtering *gml:LinearRing*s which were unlikely to contain candidate matches.

5.11 Heuristics

Results obtained in early experiments revealed the prevalence of false positives across façades. Consequently, heuristics were added to the pipeline to address those false positives. HOG-based template matching resulted in object boundaries that were mapped to the 3D building model coordinates. The heuristics were applied to boundary represented objects to select or eliminate them. The heuristics are summarised in Figure 5.13, and as follows. Heuristic 1: reject higher scoring overlapping candidate matches if they were too small. Heuristic 2: candidate matches that were too tall or too short were rejected. Heuristic 3: candidate 'door' matches which were too high up the building were rejected. The rationale and the method for each of the three heuristics is expanded upon below.

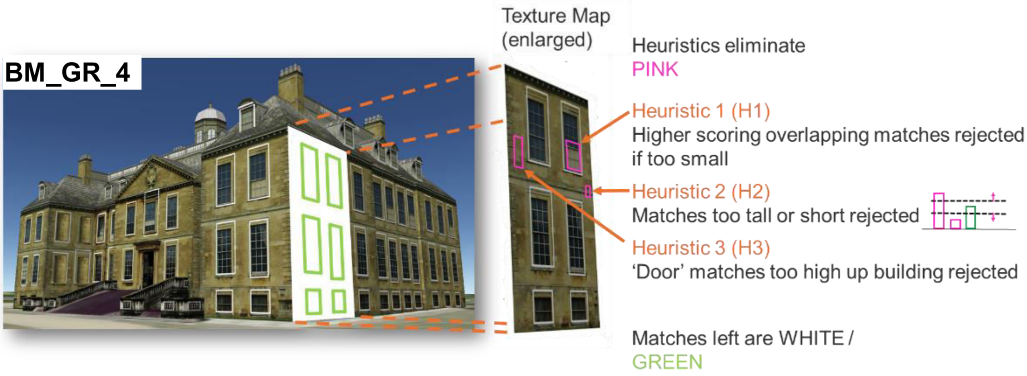

BM_GR_4
Texture Map (enlarged)
Heuristics eliminate PINK
Heuristic 1 (H1)
Higher scoring overlapping matches rejected if too small
Heuristic 2 (H2)
Matches too tall or short rejected
Heuristic 3 (H3)
'Door' matches too high up building rejected
Matches left are WHITE / GREEN

Figure 5.13: Summary of the Heuristics Applied to an Example *gml:LinearRing*
HOG-based template matching has resulted in candidate-match bounding boxes. The heuristics operated on the candidate-match bounding boxes, their aim being to increase the number of true positives and reduce the number of false positives.

Heuristic (Hn)	Heuristic Threshold Description	Value
H1	Score Proximity	0.04
	Overlap %	89%
H2	Doors Min Length	1.5
	Doors Max Length	10
	Windows Min Length	0.3
	Windows Max Length	20
H3	Max Height at Door Base	6

Table 5.8: Heuristics Threshold Values
H2 and H3 values are in metres, the unit of real-world measurement in the 3D building models.

Variable Name (Section 5.11)	Description
CITYGML	CityGML XML file for the current building
CITYGML.LR	All *gml:LinearRing*s for CITYGML
CITYGML.LR.POS_LIST	*gml:posList*s for CITYGML.LINEAR_RING, in 3D real-world coordinates
CITYGML.LR.POS_LIST.MIN_Z	Lowest z value (height) in CITYGML.LR.POS_LIST
CM_COLLECTION	All candidate matches for current *gml:LinearRing*
CM_COLLECTION.BB_PX	Bounding boxes for CM_CURRENT, in pixel coordinates
CM_COLLECTION.BB_PX.CENT	Centroids for CM_CURRENT.BB
CM	Candidate match
CM.CR	Candidate match for current iteration of for-loop
CM.CR.BB_PX	Bounding box, in pixel coordinates, for CM.CR
CM.CR.BB_PX.AREA	Bounding box area, in 2D pixel coordinates, for CM
CM.CR.BB_RW	Bounding box, in 3D real-world coordinates, for CM.CR
CM.CR.BB_RW.BASE_COORDS	The two base coordinates (i.e. bottom vertices of a window or a door bounding box) for CM.CR.BB_RW
CM.CR.BB_RW.HEIGHT	Height, in texture-space orientation, in real-world units, of CM.CR_BB_RW
CM.CR.BB_RW.VERTICES	Bounding box vertices for CM.CR.BB_RW
CM.CR.CLASS	Class e.g. 'Georgian-Regency Window 3x3 panes', for CM.CR
CM.CR.SCORE	Match-score for CM.CR
NR	Non-maximum suppression region – see Fig. 6 (a)
NR.MAX	Candidate match in NR with the highest score
NR.MAX.BB_PX	Bounding box, in pixel coordinates, for NR.MAX
NR.MAX.BB_PX.AREA	Bounding box area for NR.MAX.BB_PX
NR.MAX.BB_PX.VERTICES	Bounding box vertices for NR.MAX.BB_PX
NR.MAX.SCORE	Maximum match-score in NR

Table 5.9: Description of Variable Names used in Pseudo-Code (Section 5.11)

5.11.1 Heuristic 1 – Reject Higher-Scoring Overlapping Candidate Matches if too Small

The aforementioned early experiments revealed the prevalence of what might be termed ‘partial matches’. Such a scenario might include e.g. an instance of a Georgian-Regency ‘3x6 pane’ window on a texture map image resulting in a candidate match from a smaller ‘3x3 pane’ template. Thus, candidate matches resulting from a lower score for a ‘3x6 pane’ template, and a higher score for a ‘3x3 pane’ template, were resolved to leave a ‘3x3 pane’ candidate match after non-maximum suppression. In fact, it is desirable for the candidate match resulting from the larger ‘3x6 pane’ template to be left. The addition of heuristic 1 (H1) addressed this problem.

Heuristic 1 comprised an alteration to stage (c) from Section 5.7, plus (d) the addition of a second, greedy, NMS stage based on area. The extended stage (c) operated as follows, operating in turn on each *gml:LinearRing* and each of the candidate matches for a *gml:LinearRing*. As with the standard NMS approach used here, the ‘NMS region’ (NR) was an area twice as wide and twice as high as the width and height respectively of the

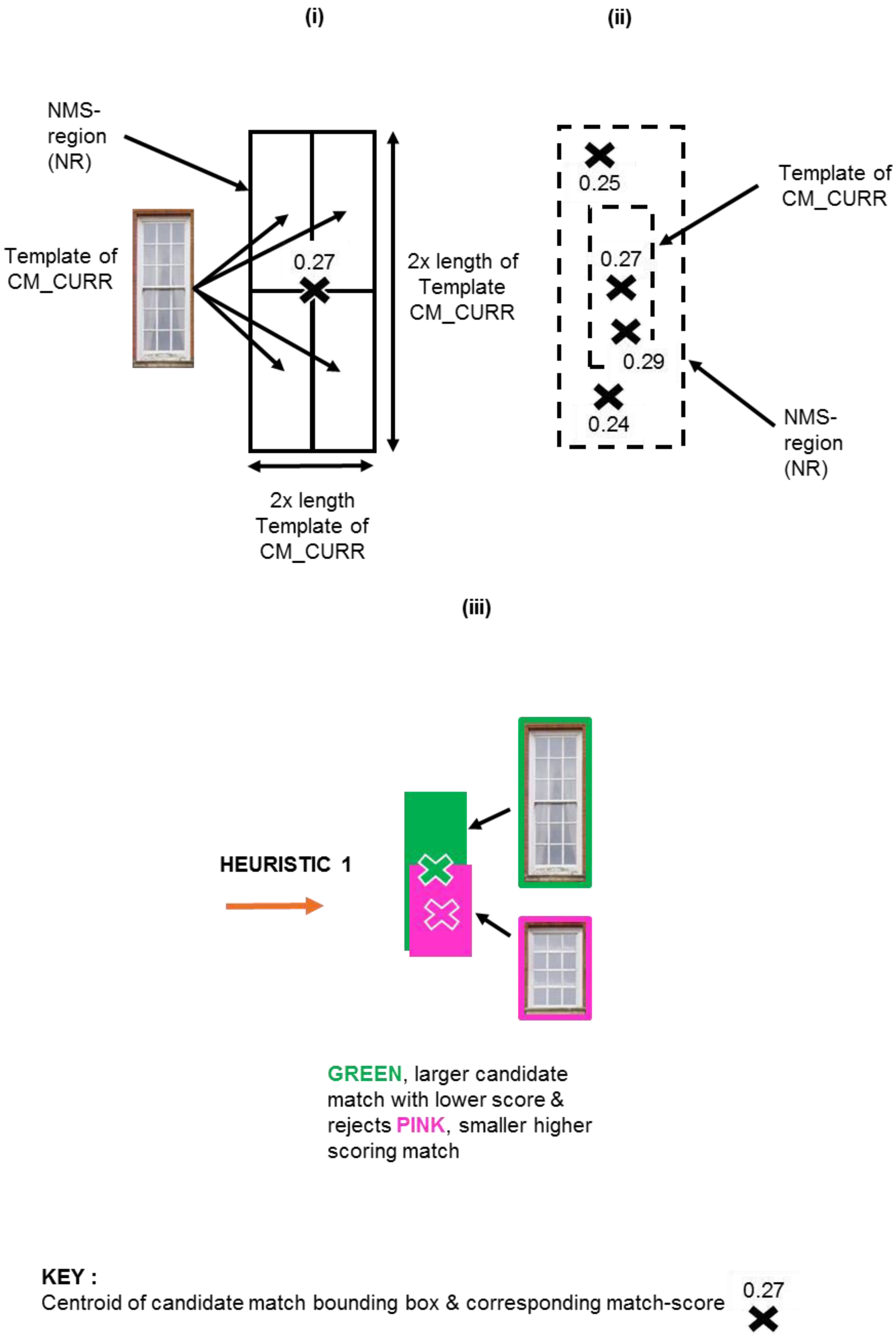

(i)
(ii)
NMS-region (NR)
Template of CM_CURR
0.27
2x length of Template CM_CURR
2x length Template of CM_CURR
0.25
0.27
0.29
0.24
Template of CM_CURR
NMS-region (NR)
(iii)
HEURISTIC 1
GREEN, larger candidate match with lower score & rejects PINK, smaller higher scoring match
KEY :
Centroid of candidate match bounding box & corresponding match-score
0.27

Figure 5.14: Illustration of Stage (c) of Heuristic 1 – HOG-based Template Matching (i) Illustrates the manner in which the non-maximum suppression (NMS) region (NR) for the candidate match of the current iteration (CM_CURR) was formed, in this case for the candidate match (CM_CURR) with the match-score of 0.27. (ii) Illustrates the candidate matches within NR, for CM_CURR which were considered during NMS. (iii) Illustrates the following: the result of stage (c) of heuristic 1 (which is part of NMS) whereby the larger, slightly lower scoring candidate match was kept and the smaller 0.29 scoring candidate match, and all other candidate matches in NR, were rejected. The templates that resulted in the candidate matches with the score of 0.27 and 0.29 are shown to left, ringed in green and pink respectively.

template that resulted in the candidate match, centred on the centroid of that candidate match. An illustration of the NR is shown in Figure 5.14 (i). As can be observed in Figure 5.14 (ii), the previous method would simply have rejected all candidate matches apart from the highest scoring one (0.29), regardless of the size of the template which resulted in that candidate match. Heuristic 1 kept larger candidate matches with a slightly lower score, as illustrated in Figure 5.14 (iii) and as described below. Stage (d) was intended to remove any overlapping smaller matches which still remained after stage (c). See Table 5.8 for the threshold values used in the heuristic. ‘Score proximity’ was the difference allowed between scores to trigger elimination by heuristic 1, i.e. the ‘3x6 pane’ window candidate match could have a score of up to 0.04 less than the ‘3x3 pane’ window candidate match from the example above. Note that match-scores tended to be in the range 0-0.4 where 0.04 represents 10% of the highest likely match-score.

The overlap percentage was determined empirically, using the façade images, as follows. For each façade image in turn, HOG-based template matching was run, using all Georgian-Regency templates, with the overlap percentage in the range of [50%, 100%] and with a step of 10%. The 50% starting value was chosen because the two candidate matches being compared should be mostly overlapping to resolve the 'partial match' issue. Mean highest F-measure across all the façade images was used to determine the overlap percentage which was ultimately used.

The process flow for heuristic 1, including stage (d), is shown below. See Table 5.9 for a description of the variables used.

```
input CM_COLLECTION (the set of candidate matches)
STAGE (c) i.e. suppression of strong, smaller matches
for each CM in CM_COLLECTION
        compute NR (non-maximum suppression region)
        if CM.CR.SCORE = NR.MAX.SCORE then
                do add CM.CR to CM_ COLLECTION_TO_KEEP
        else if
                -0.04 >= (CM.CR.SCORE – NR.MAX.SCORE) <=0
                and (area(NR.MAX.BB_PX ∩ CM.CR.BB_PX) / …
                        (min(NR.MAX.BB_PX.AREA, …
                        CM.CR.BB_PX.AREA))) > 0.9
                and CM.CR.BB_PX.AREA > …
                        NR.MAX.BB_PX.AREA then
                        do add CM.CR to …
                                CM_ COLLECTION_TO_KEEP
do CM_COLLECTION = CM_ COLLECTION_TO_KEEP
STAGE (d) i.e. suppression of smaller overlapping matches
sort CM_COLLECTION by y then x of …
        CM_COLLECTION.BB_PX.CENTROID
j = 0 inds = [];
for each remaining CM in CM_COLLECTION
        j = j + 1
        compute NR (non-maximum suppression region)
        if CM.CR.BB_PX.AREA < NR.MAX.BB_PX.AREA
                do append j to inds
do empty CM_COLLECTION where index = inds
```

5.11.2 Heuristic 2 – Reject Candidate Matches too Tall or too Short

Early experiments also revealed that particularly small (or large) false-positive candidate matches could sometimes occur, matching edges representing window glazing bars against edges representing, for example, the mortar lines in brickwork. Heuristic 2 (H2) addressed this issue.

Heuristic 2 used the coordinates of the vertices of the bounding boxes of the candidate matches in the 3D world-space coordinates of the CityGML. Using 3D Pythagoras, the length of the bounding box was obtained, and any candidate matches that were too tall or were too short were rejected. See Table 5.8 for the threshold values used in the heuristic. These threshold values were determined empirically using the buildings contained in façade images, aside from FI_4, since it was of the same building used in BM_GR_4 i.e. one of the 3D building models used in testing. The thresholds were determined by conducting a manual survey of the range of real-world heights of windows and doors contained in 3D building models BM_GO_1, BM_OR_1_V, BM_OR_2_V and BM_OR_3_V i.e. the buildings found in FI_1, FI_2, FI_3 and FI_5 respectively. The Trimble *SketchUp* Tape Measure tool was used to measure the heights.

The process flow for heuristic 2 is shown below. See Table 5.9 for a description of the variables used.

```
input CM_COLLECTION (the set of candidate matches)
for each CM in CM_COLLECTION
    compute CM.CR.BB_RW.VERTICES
    compute CM.CR.BB_RW.HEIGHT
    if CM.CR.CLASS = ‘door’ then
        if CM.CR.BB_RW.HEIGHT< 1.5m
            or CM.CR.BB_RW.HEIGHT> 10m then
                do reject CM.CR
    if CM.CR.CLASS = ‘window’ then
        if CM.CR.BB_RW.HEIGHT< 0.3m
            or if CM.CR.BB_RW.HEIGHT> 20m then
                do reject CM.CURR
```

5.11.3 Heuristic 3 – Reject Candidate Door Matches too High up Building

Finally, early experiments also identified the prevalence of false-positive candidate door matches occurring above the ground floor. Heuristic 3 addressed this issue. Heuristic 3 also used the real-world 3D coordinates of the vertices of the bounding boxes of the candidate matches. The building model’s lowest z-coordinate value in real-world units was then used to reject candidate door matches that had base vertices above one storey height i.e. 3m based on Biljecki *et al.* (2017). To accommodate exposed basement substructures, including buildings set into slopes, the value was extended to 6m, as noted in Table 5.8. The process flow for heuristic 3 is shown below. See Table 5.9 for a description of the variables used.

```
input CITYGML,
        CM_COLLECTION (the set of candidate matches)
compute find CITYGML.LR.POS_LIST
for each CM in CM_COLLECTION
        if CM.CR.CLASS = 'door' then
                if either of …
                        CM.CR.BB_RW.BASE_COORDS  > …
                        (CITYGML.LR.POS_LIST.MIN_Z + 6) then
                                do reject CM.CR
```

5.12 F-measure & Class Granularity

The judgement of whether a candidate match was a true positive was conducted in pixel space for the current *gml:LinearRing*. Consideration was given to the use of world space instead, and the extra precision it would allow. HOG-based template matching was conducted in pixel space. As such, the use of world space for determining true and false positives would not have provided any more accuracy than the size of one pixel in world space. Before calculating F-measure, the number of true positives, false positives, total relevant and total retrieved were recorded across all *gml:LinearRing*s in the CityGML that passed the four validation checks.

F-measure was calculated at each of the three class granularities (although for C20 and Norman style 3D building models class granularity 3 was not used). The method of determining true and false positives for the standard template matching trial used a centroid proximity approach, with no consideration of the amount of overlap or relative sizes of the ground truth and candidate match. Here the *PASCAL VOC* area overlap method was used instead – it has become a standard approach in the field. A candidate match was judged to be a true positive if the 'overlap ratio a_o between the predicted bounding box B_p and ground truth bounding box B_{gt}' exceeded 0.5, as calculated using Equation 5.26 (Everingham *et al.*, 2010). $B_p \cap B_{gt}$ is the intersection of the candidate-match and ground truth bounding boxes, and $B_p \cup B_{gt}$ is their union. Everingham *et al* chose to set the threshold at 50% to account for inaccuracies in the bounding boxes of ground truth data. While an option in this work would have been to increase the threshold (meaning correct detections were more challenging) it was decided to keep the 50% threshold on the basis that it was a standard in the field.

$$a_o = \frac{area(B_p \cap B_{gt})}{area(B_p \cup B_{gt})}$$ **Equation 5.26**

Note that the method used here addressed a detection rather than a classification problem, whereby F-measures of zero were possible.

5.13 Semantic & Geometric Enrichment

The generation of the world-space coordinates for the vertices of window or door rectangles, plus the associated class of window or door, represent the semantic and geometric enrichment achieved. While this work did not inject new CityGML content for the windows and doors into the existing CityGML, an example of the necessary new CityGML is nonetheless illustrated in Figure 5.15, for a new window.

New XML for a window or door would need to be inserted into an existing *bldg:WallSurface* element (or perhaps, for a skylight, into a *bldg:RoofSurface*). Another consideration would be whether to specify the new window or door just with a *gml:exterior gml:LinearRing*, or both a *gml:exterior* and *gml:interior*. Note too that the CityGML standard defines that the *app:textureCoordinates* and corresponding *gml:posList*. *gml:exterior gml:LinearRing*s 'have to be specified in reverse order … counter-clockwise … when looking in the opposite direction of the surface's normal vector' while those for *gml:interior gml:LinearRing*s are specified clockwise (Gröger *et al.*, 2012). Lastly, any additional *gml:LinearRing*s would require additional *app:target* elements for the *app:ParameterizedTexture* that corresponded to the texture map image. This might necessitate clipping, i.e. cookie cutting of the *gml:exterior gml:LinearRing* and corresponding *app:textureCoordinates* and *gml:posList*. Note that all this assumes correct syntactic structure of the pre-existing CityGML. While the conversion from the *SketchUp* format to CityGML in this work has enforced correct syntactic structure, in this author's experience such correct semantic syntactic structure is not always present in other 3D city models. The challenge of enriching the CityGML would also be compounded by the variety of syntactic approaches that the standard allows for the same semantic and geometric features.

Note also that the methods presented here include an automated output of all texture map images, marked up with candidate matches as white bounding boxes, for a processed 3D building model. As such the enrichment *can* be visualised, by importing the original CityGML model into a viewer, having replaced the original texture map image files with the marked-up files. The marking-up can include text describing the class of the candidate match and also the match-score.

```
<bldg:opening>
      <bldg:Window gml:id="_Filename_BD.BuildingName_W.Window1">
            <gml:name>Georgian-Regency Style Match-Score 0.33</gml:name>
            <bldg:lod3MultiSurface>
                        <gml:MultiSurface>
                              <gml:surfaceMember>
                                    <gml:Polygon gml:id="_Filename_BD.BuildingName_PG.Polygon2">
                                                <gml:exterior>
                                                      <gml:LinearRing gml:id=
                                                            "_Filename_BD.BuildingName_PG.Polygon1_LR.LR1">
                                                            <gml:posList srsDimension="3">
                                                                  713012.97 5788489.48 1.09
                                                                  713014.56 5788489.48 1.09
                                                                  713014.56 5788489.48 3.55
                                                                  713012.97 5788489.48 3.55
                                                                  713012.97 5788489.48 1.09
                                                            </gml:posList>
                                                      </gml:LinearRing>
                                          </gml:exterior>
                                    </gml:Polygon>
                              </gml:surfaceMember>
                  </gml:MultiSurface>
            </bldg:lod3MultiSurface>
      </bldg:Window>
</bldg:opening>
```

Figure 5.15: Hypothetical Semantic & Geometric Enrichment of CityGML with a New Window Object
While this work did not create new CityGML for the detected missing windows or doors the above is a hypothetical example of what such new XML might look like. This could be achieved using the window vertices which *were* calculated with the HOG-based template matching method described here, plus the associated class of the candidate match, and the associated match-score. The world-space coordinates of the vertices would correspond to the coordinates under *gml:posList*. *gml:name* value corresponds to example window class, followed by a match-score, hypothetically achieved with the method. Note that number of decimal places shown for the *gml:posList* coordinates has been reduced for illustration only. Also, note that a CityGML door would follow an identical format, aside from *bldg:Window* being replaced with *bldg:Door*.

5.14 Summary

This chapter has detailed the approach used for HOG-based template matching. HOG-based template matching from Xiao (2013) and Zhang *et al.* (2013b) was placed into a pipeline. To that pipeline was added a number of steps, as described in this chapter and summarised below. The approach to parsing the CityGML file, including the determination of the list of texture map images to be processed, has been detailed. For each of those images and for each template, HOG-based template matching was run. Candidate matches were then rendered onto each *gml:LinearRing* that used the image. Two match-score normalisation schemes were trialled. Non-maximum suppression was also carried out on the candidate matches. For each *gml:LinearRing* the candidate matches were then transformed into 3D world-space coordinates, and spatial accuracy validated. Heuristics, including the use of world-space units, were used to improve the results. F-measure was then calculated for the entire 3D building model, at each class granularity. Finally, this chapter described the enabling of the semantic and geometric enrichment, including possible future work for injecting new CityGML into the existing 3D building model file.

Chapter 6

SVM to Replicate Thresholding

6.1 Introduction

Thresholding of match-scores was a step in the method described in Chapter 5, where the thresholds in question had been determined empirically beforehand. However, what does such an empirical approach mean for the scalability of the methods in this thesis to unseen 3D building models and new architectural styles? Is it possible to automate the empirical process of choosing thresholds, by using machine learning, thus improving scalability? This is where using the SVM comes in, namely to replicate the thresholding approach. Incidentally, the potential capability to scale to new architectural styles assumes the availability of sufficient and suitable training data.

The basis of the machine learning approach used here was that the combination of match-scores for different templates was representative of the object at the location of the corresponding candidate match. Essentially, one set of match-scores could suggest an object was e.g. a ‘Norman Style Window’, while a different set of match-scores could suggest an object was a ‘Gothic Style Door’. In addition, it would be important to be able to detect negative classes i.e. locations that were neither a window nor a door. A set of match-scores for these types of ‘object’ would also need to be included in the approach.

The data used comprised façade images not used for the methods in Chapters 4 or 5, and templates used for the method in Chapter 5. The façade images were used to gather the training observations.

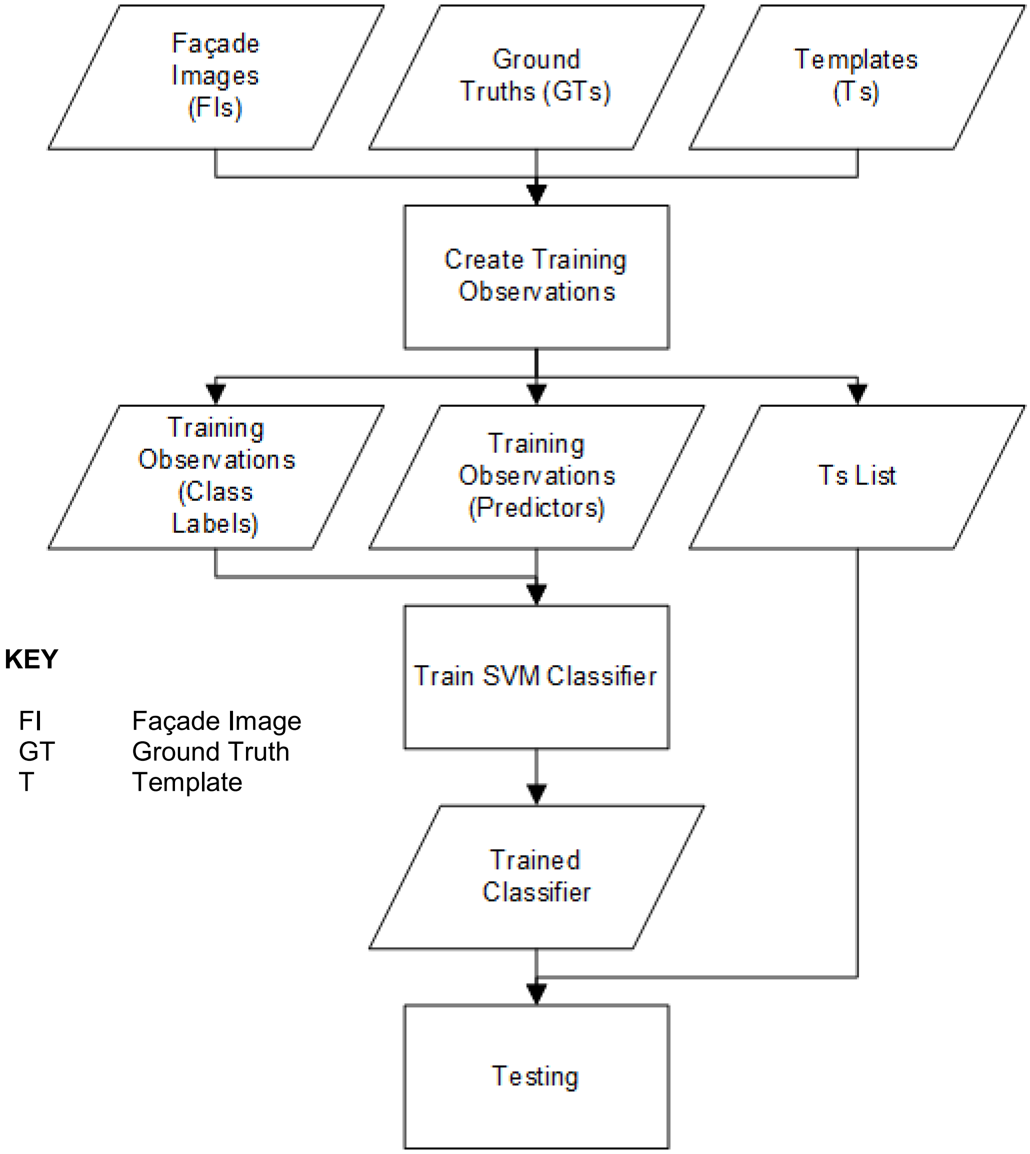
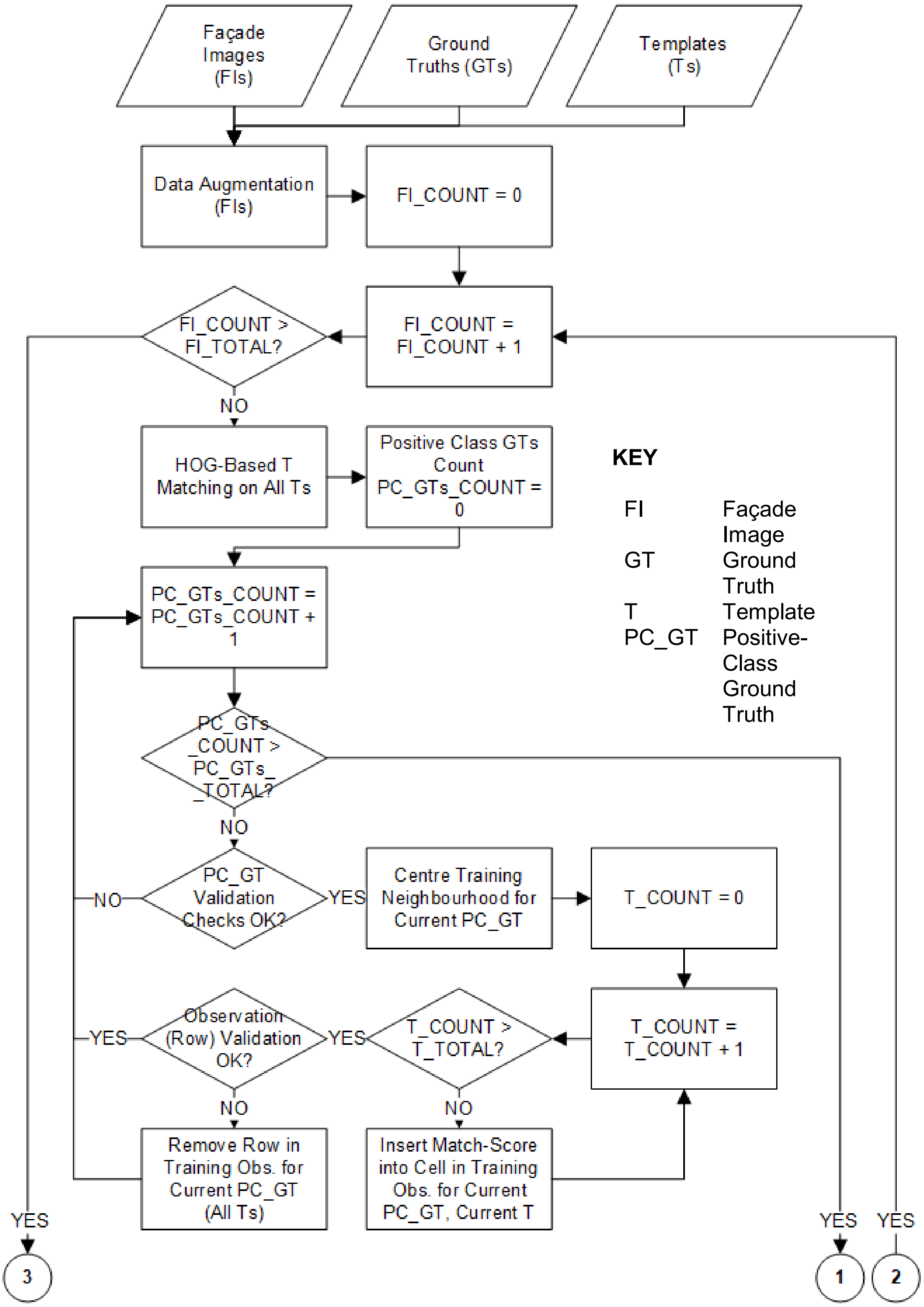


Figure 6.1: Process Flow (Summary) for use of SVM to Replicate Match-Score Thresholding This process flow links to the process flows in Figure 6.2, Figure 6.3, Figure 6.5 and Figure 6.6.

The process steps carried out in order to gather training observations, to train the SVM and then to conduct testing are detailed below. This includes steps to optimise the performance of the classifier model. Figure 6.1 summarises the process flow used for training and testing with an SVM, which is described in the rest of this chapter.

6.2 Training Observations & Classification Model Improvement

The process flow for the creation of training observations is shown in Figure 6.2 and Figure 6.3, and described below, beginning with an explanation of the approaches taken to optimising the performance of the classification model.


Façade Images (FIs)
Ground Truths (GTs)
Templates (Ts)
Data Augmentation (FIs)
FI_COUNT = 0
FI_COUNT = FI_COUNT + 1
FI_COUNT > FI_TOTAL?
NO
HOG-Based T Matching on All Ts
Positive Class GTs Count PC_GTs_COUNT = 0
PC_GTs_COUNT = PC_GTs_COUNT + 1
PC_GTs _COUNT > PC_GTs_ _TOTAL?
NO
PC_GT Validation Checks OK?
NO
YES
Centre Training Neighbourhood for Current PC_GT
T_COUNT = 0
T_COUNT = T_COUNT + 1
T_COUNT > T_TOTAL?
YES
Observation (Row) Validation OK?
YES
NO
NO
Remove Row in Training Obs. for Current PC_GT (All Ts)
Insert Match-Score into Cell in Training Obs. for Current PC_GT, Current T
YES
3
YES
1
YES
2

KEY

FI	Façade Image
GT	Ground Truth
T	Template
PC_GT	Positive-Class Ground Truth

Figure 6.2: Process Flow for Creating Training Observations (1 of 2)
See also Figure 6.3. Figure 6.2 and Figure 6.3 are an expansion of the 'Create Training Observations' process step in Figure 6.1.

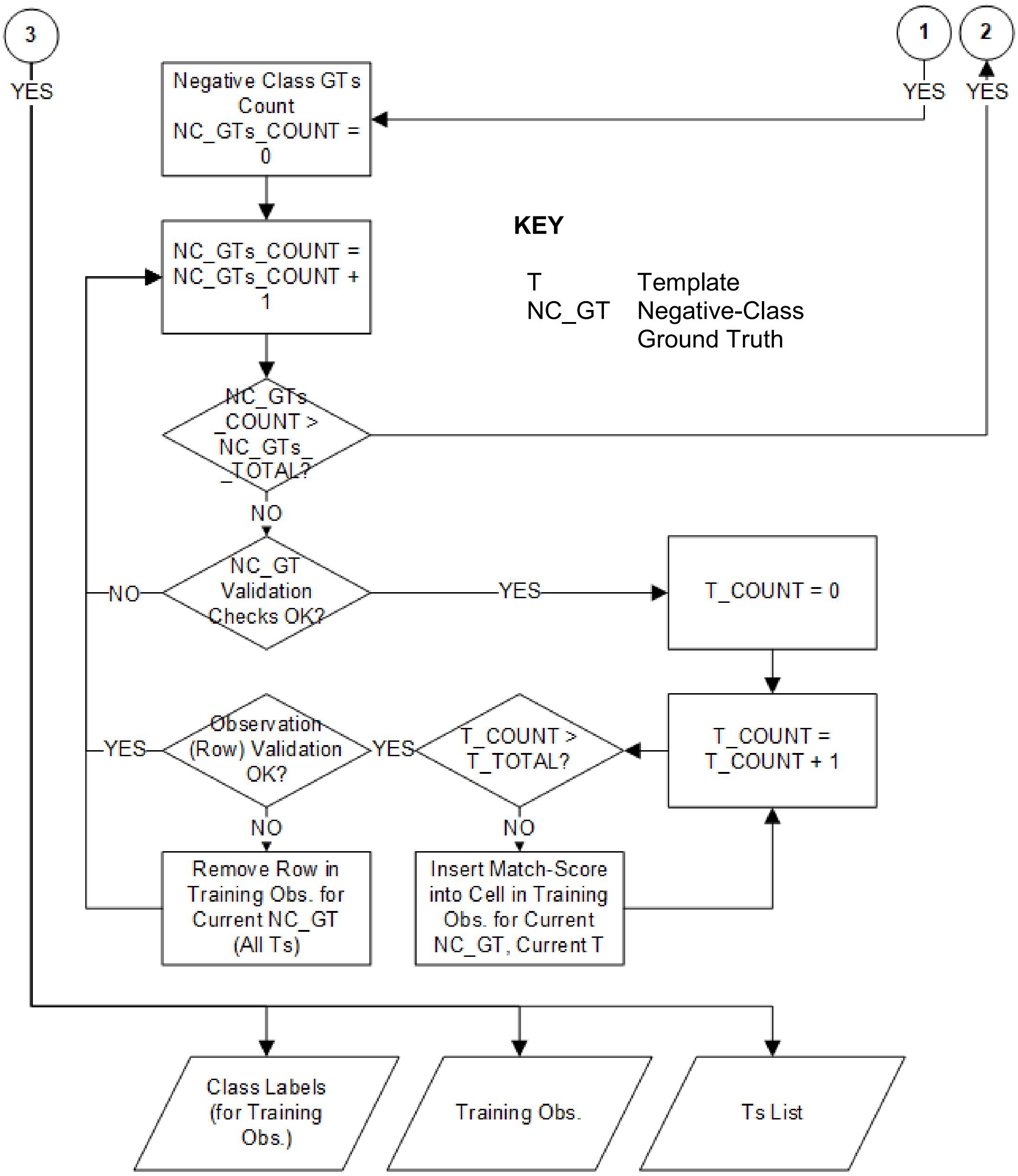

3
YES
1
2
YES
YES
Negative Class GTs Count NC_GTs_COUNT = 0
NC_GTs_COUNT = NC_GTs_COUNT + 1
KEY
T Template
NC_GT Negative-Class Ground Truth
NC_GTs _COUNT > NC_GTs _TOTAL?
NO
NC_GT Validation Checks OK?
NO
YES
T_COUNT = 0
YES
Observation (Row) Validation OK?
YES
T_COUNT > T_TOTAL?
T_COUNT = T_COUNT + 1
NO
NO
Remove Row in Training Obs. for Current NC_GT (All Ts)
Insert Match-Score into Cell in Training Obs. for Current NC_GT, Current T
Class Labels (for Training Obs.)
Training Obs.
Ts List

Figure 6.3: Process Flow for Creating Training Observations (2 of 2)
See also Figure 6.2, noting that its caption applies to this figure.

Training observations comprised a vector of match-scores at locations across the training images, for both true (positive class) and false (negative class) instances of window and door objects. Positive-class ground truths were marked up in advance as rectangles, with a template class, on training (façade) images. Locations representing the negative class were also marked up on every training image in advance, as points (as described in Chapter 3). To mimic the non-SVM approach (see Chapter 5) training images were initially all doubled in size to improve the detection of small instances of windows or doors on texture map images. Then, for each ground truth in each training image, the centroid pixel for the ground truth rectangle was calculated and a ‘training neighbourhood’ derived. The method for obtaining training observations, including the use of the training neighbourhood, is illustrated in Figure 6.4, and described below.

The training neighbourhood was initially centred on the centroid of the ground truth rectangle. The neighbourhood was given a width and height, in pixels, of 20% of the width of the ground truth rectangle. As shown in Figure 6.4, the ground truth centroid location was not always in the 8×8-pixel grid cell that contained the highest score within the training neighbourhood. Consequently, the neighbourhood was re-centred onto the location, within the neighbourhood, which contained the highest count of maximum match-scores for templates representing the ground truth class. This was carried out in order to obtain match-scores which were representative of the centre of the strongest candidate match for the ground truth class. For example, for sake of illustration, assume that templates T_1, T_2 and T_3 represent all the templates for the 'Norman Style Window' class, where the ground truth class was the same. If grid cell (1) then had match-scores of 0.25, 0.33 and 0.23 for T_1, T_2 and T_3 respectively, and grid cell (2) had match-scores of 0.22, 0.31 and 0.24 for T_1, T_2 and T_3 then (1) would be chosen over (2). If (1) had the highest count of maximums within the training neighbourhood, among all the grid cells, then the neighbourhood would be re-centred onto grid cell (1).

Note that Figure 6.4 (a), (b) and (c) only show one ground truth. This is purely for the sake of illustration – other training images included multiple ground truths. Furthermore, note how Figure 6.4 (d) shows variability of match-scores within a training neighbourhood. Due to such variability, only the match-scores for the grid cell at the new centre of the training neighbourhood were used to compile training observations. During early experiments, F-measure was used to refine the number of training neighbourhood grid cells used to compile training observations. Ultimately, it was decided to only use one grid cell, though not before smoothing of the match-scores was trialled (see below).

Before calculating the location of the central grid cell in the training neighbourhood, three validation checks were carried out on the ground truth, as follows.

(a) If the location of the highest count of maximums (as described above) was not the same for both the training neighbourhood and within the entire rectangle of the ground truth, then the ground truth was rejected and not used to produce a training observation.

(b) If the maximum match-score for any template in the training neighbourhood was not from a template of the ground truth class, then the ground truth was rejected, and not used to produce a training observation.

The reason for validation checks (a) and (b) was to ensure that the positive-class training observations were representative of strong candidate matches.

(c) Additionally, any ground truths that had grid cells at the centre of the re-centred training neighbourhood, which did not contain a full complement of match-scores for all templates, were also rejected.

As a reminder, the HOG descriptor was masked out at the periphery of the image due to the method of descriptor formation. Once again, note that differently sized templates would result in differently-sized HOG descriptors. Therefore, the extent to which match-scores resulting from convolution were masked out depended on the sizes of the HOG descriptors for the image and for the template, noting that templates used were of varying sizes. In practice, due to their location away from the boundaries of the training images, no positive-class ground truths were rejected as a result of validation check (c). Nonetheless, for the sake of scalability to other training images, the use of validation step (c) for positive-class ground truths was maintained.

A table of positive-class training observations was compiled (to which training observations representing the negative class were added, as described below). Each row was a training observation, representing one ground truth instance, and each column represented a template. Each row had a template class label, and the vector for each row comprised a match-score for each template, where those match-scores were the predictors.

Negative-class training observations were then compiled as follows. Using the negative-class ground truth points for a training image, the nearest 8×8-pixel grid cell was determined using Euclidean distance. The match-scores within that grid cell, for all templates, were then used to compile training observations in a similar way to those for positive class. Having determined the closest grid cell, validation check (c) was then carried out on that grid cell. Unlike the positive-class observations, the class label given to each negative-class training observation row was ‘null’.

For the negative-class training observations, a training neighbourhood approach was not used. Instead, the grid cell closest to the marked-up point was required, for the following reason. Positive-class instances, i.e. windows or doors on a training image, had a centre which the method was able to locate, based on match-score. Attempting to determine the centre of a negative-class instance would be entirely arbitrary– after all, it could be argued, choosing an object that represents the negative class would be subjective.

Preliminary experiments included the use of the coherent line drawing method as a pre-processing step on the training images, to emphasise disjoint lines. However, as with its

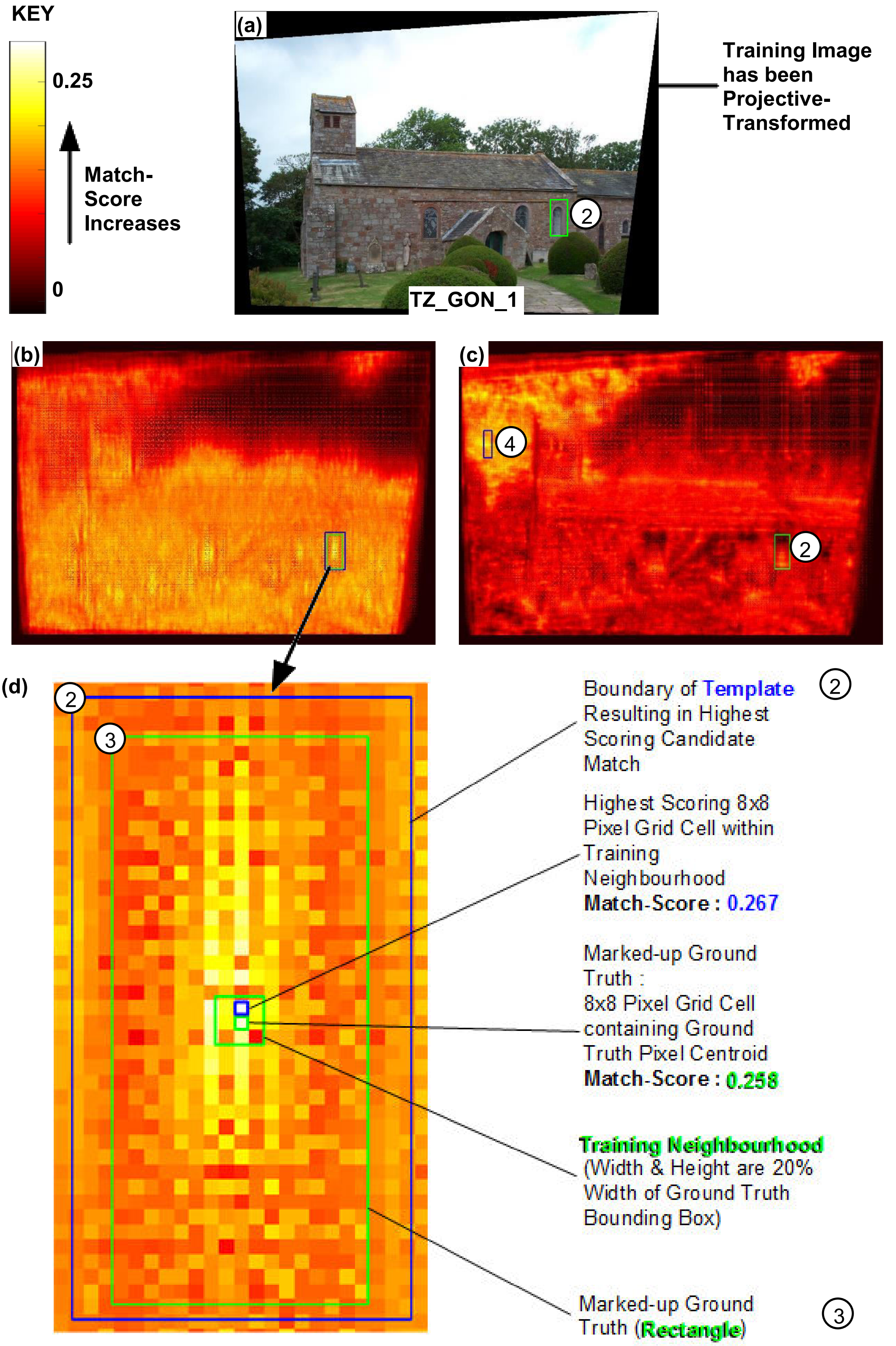


Figure 6.4: Example of Creation of Training Observation from Ground Truths
(*a*) Source image of a Norman church. (*b*) Heat map of match-scores with a 'Norman Style Window' template, showing location of 2 and 3 in (b). (*c*) Heat-map of match-scores for same source image but with a 'C20 Style Window' template. 4 is the location highest scoring candidate match for that template. (*d*) Zoomed-in region of (b).

use during early experiments with the no SVM approach in Chapter 5, a worsening of F-measure resulted during testing. Data augmentation, i.e. using both the unaltered training image plus a coherent line drawing version as inputs to the collection of training data, was also trialled early on. Once again, the F-measure results from testing were inferior compared to just using the unaltered training images. Coherent line drawing was therefore not taken further as a result.

Figure 6.4 shows the high variability of match-scores within a region as small as a training neighbourhood. In an attempt to smooth these match-scores a Gaussian blur was initially applied to the match-scores (not to the actual images) as a trial. As with the experimentation with coherent line drawing, the F-measure from testing deteriorated as a result. While blurring had the effect of smoothing the match-scores across the training neighbourhood, it lessened the highest match-score as a result. Training observations would then include positive-class entries with lower match-scores. During classification (testing) this led to more false positives. As a result, the blurring of the match-scores was not taken any further either.

On a positive note, early experiments determined that results were improved when combining Gaussian blurred (σ = 1) training images with the unaltered training images. Doing so therefore doubled the number of training images and ground truths used to gather training observations. This data augmentation step was therefore included in the pipeline. Other Gaussian kernel sizes (σ = 0.5, 1.5, 2, 3) were also trialled, but σ = 1 proved the most effective.

To recap, the ratio of positive-class to negative-class examples was intentionally set at a ratio of 1:6, based on the approximate ratio of 1:6 for ‘window or door’ to ‘non-window or non-door’ regions on façades (see Chapter 3). Again, by way of a reminder, note that early experiments with an equal number of positive-class and negative-class observations resulted in a negligible change in performance (see Chapter 3).

Table 6.1 shows the effect on the number of positive-class and negative-class ground truth instances of validation checks (a), (b) and (c) on the unaltered and Gaussian blurred façade images used for training. In summary, 37 positive-class ground truths (15%) were rejected because of the combined use of validation checks (a), (b) and (c), and four negative-class ground truths (<1%) were rejected as a result of validation check (c).

Style	Image Type	Num. of Positive-Class Ground-Truths Marked up	Num. of Negative-Class Ground Truths Marked up	(i) Num. of Positive-Class Ground Truths After Validation Checks (a), (b) & (c)	(ii) Num. of Negative-Class Ground Truths After Validation Check (c)
C20	Unaltered	35	210	26	208
C20	Gaussian Blur	35	210	25	208
	SUBTOTAL	**70**	**420**	**51**	**416**
Georgian-Regency	Unaltered	52	312	45	312
Georgian-Regency	Gaussian Blur	52	312	47	312
	SUBTOTAL	**104**	**624**	**92**	**624**
Norman & Gothic	Unaltered	35	210	33	210
Norman & Gothic	Gaussian Blur	35	210	31	210
	SUBTOTAL	**70**	**420**	**64**	**420**
	TOTALS	**244**	**1464**	**207**	**1460**

Table 6.1: Ground Truths on Façade Images used for Training After Validation Checks
Ground truths used in training are those in columns (*i*) and (*ii*).

The process flow for training the classifier is shown in Figure 6.5, and described below. Sequential feature selection was conducted, by iteratively adding or removing predictor variables and then using F-measure to evaluate the effectiveness of the selection step (MathWorks, 2016a). The result of this approach was that adding two shrinkage steps, '(1)' and '(2)', further improved F-measure. The rationale behind using a shrinkage approach, i.e. reducing the number of predictor variables, was to improve the generalisation of the classifier and thus reduce overfitting. (1) and (2) are illustrated in Table 6.2, and described below.

Note that the term shrinkage, as a statistical definition, is used here to describe: (i) the setting of some predictor values for an observation to zero, which effects feature dimensionality reduction for an observation; and (ii) the removal of features i.e. dimensionality reduction for all of the observations (Li, 2017). These two definitions correspond to steps (1) and (2) respectively, where the implementation of those steps is defined as follows:

(1) Firstly, for each training observation, each predictor value was zeroed if it was not the maximum i.e. if it was not the maximum in that row in the set of training observations. Based on the validation steps (a) and (b), above, the highest match-score for a row was for a template that corresponded to the class label of the training observation.

(2) Secondly, the dimensionality of the training vector was reduced by reducing the number of predictors (columns in the table of training observations) for the same template class to one. In practice this step reduced the dimensionality of the feature vector from 149 to 25, when using the template set covering all styles.

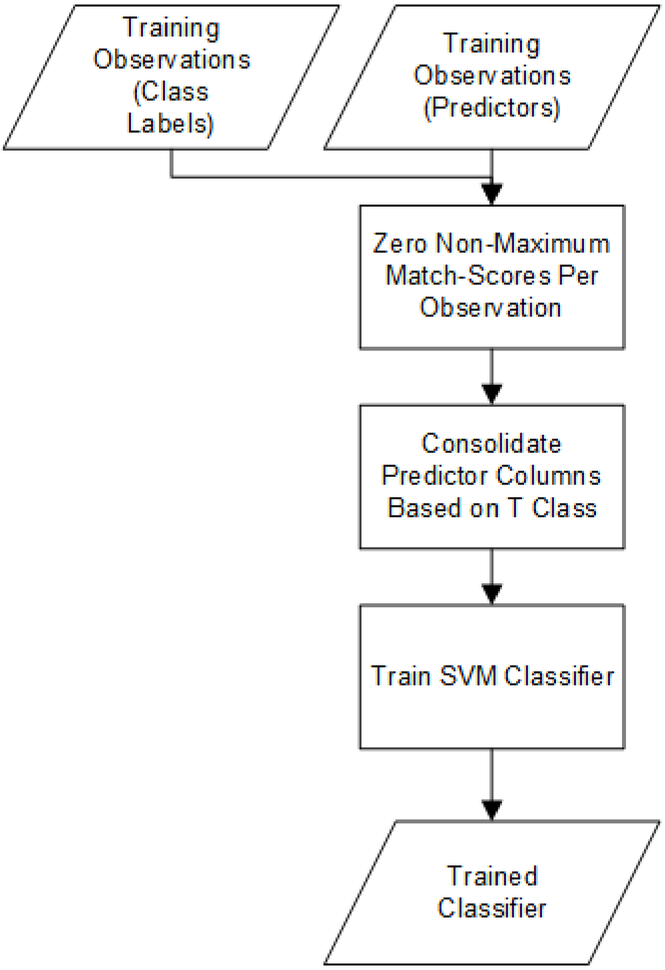

Training Observations (Class Labels)
Training Observations (Predictors)
Zero Non-Maximum Match-Scores Per Observation
Consolidate Predictor Columns Based on T Class
Train SVM Classifier
Trained Classifier

Figure 6.5: Process Flow for Training SVM Classifier
This figure is an expansion of the ‘Train SVM Classifer’ process step in Figure 6.1.

Before Feature Selection Shrinkage Steps (1) or (2)

Class Label	**Template (T_n) & Corresponding 'Template Class ('Window …')**									
	T_1	T_2	T_3	T_4	T_5	T_6	T_7	T_8	T_9	T_{10}
	Window Style C20			Window Style Gothic			Window Style Norman			
Window Style C20	0.33	0.28	0.22	0.19	0.15	0.18	0.17	0.16	0.14	0.13
Gothic Style Window	0.18	0.17	0.19	0.34	0.28	0.29	0.16	0.14	0.12	0.15
Window Style Norman	0.14	0.19	0.17	0.18	0.12	0.16	0.25	0.28	0.35	0.22

After Feature Selection Shrinkage Step (1)

Class Label	**Template (T_n) & Corresponding 'Template Class ('Window …')**									
	T_1	T_2	T_3	T_4	T_5	T_6	T_7	T_8	T_9	T_{10}
	Window Style C20			Window Style Gothic			Window Style Norman			
Window Style C20	0.33	0	0	0	0	0	0	0	0	0
Gothic Style Window	0	0	0	0.34	0	0	0	0	0	0
Window Style Norman	0	0	0	0	0	0	0	0	0.35	0

After Feature Selection Shrinkage Step (2)

Class Label	**Template Class ('Window …')**		
	Window Style C20	Window Style Gothic	Window Style Norman
Window Style C20	0.33	0	0
Gothic Style Window	0	0.34	0
Window Style Norman	0	0	0.35

Table 6.2: Example of Effect of Feature Selection Shrinkage Steps (1) & (2)
Each row is a training observation (comprising a feature vector of match-scores plus a corresponding class label) and columns are predictors. The feature vector passed to training is that after Feature Selection Shrinkage Step (2).

Instead of using shrinkage to reduce some predictor values to zero, another approach would have been to use regularisation i.e. defining weights for each predictor variable. Shrinkage produced a good improvement in F-measure versus using no shrinkage, meaning regularisation was not used.

Another form of dimensionality reduction (of the training vector) is feature transformation (MathWorks, 2016a), including the use of approaches such as principle component analysis (PCA). On the basis that approximate parity with the non-SVM F-measure results was achieved using the feature selection steps described above, feature transformation was not attempted.

The final step involved in tuning the classifier model to optimise the results was the use of automated optimisation of hyperparameters (see Chapter 2) during the training of the model.

6.3 Testing

Figure 6.6 illustrates the process flow used during testing, which is described below. To begin, the list of templates used for training were passed to the HOG-based template matching method ready for testing. Using the same set of templates would also ensure that the size of the feature vector for training and testing was the same, noting that a classifier is trained with a feature vector, and that the trained classifier uses a feature vector for classification.

A number of different classifier models were compiled: (1) using templates for all styles; (2) using all C20 style templates; (3) using all Georgian-Regency style templates; (3) using all Gothic and Norman style templates. This allowed comparison with the results of the experiments conducted using the method from Chapter 5.

Having run HOG-based template matching on a texture map image, and therefore derived match-scores for the 8×8-pixel grid cells across the image, the following two steps from the method in Chapter 5 were *skipped*: thresholding of match-scores, and non-maximum suppression step (b). The reason for removing the thresholding step was that the SVM was attempting to, in effect, mimic thresholding. In addition, a match-score for every template was needed in order to form the feature vector used for classification. Therefore, non-maximum suppression step (b) needed to be removed from the process flow.

An equivalent to validation step (c), from the process used for compiling training observations, was used during testing. If the centroid location for any candidate matches did not possess a full complement of match-scores the candidate match was filtered out, and not passed to the classification stage.

The match-scores were formed into the predictor table format used in training, though, of course, without any template class label for each row. Each row represented a grid cell, i.e. an (x,y) pixel location on the texture map image, at pixel location (5,5) within the grid cell. Classification was then run, resulting in each row being labelled with either a positive-class label, e.g. ‘Norman Style Window’, or ‘null’ if it was a negative

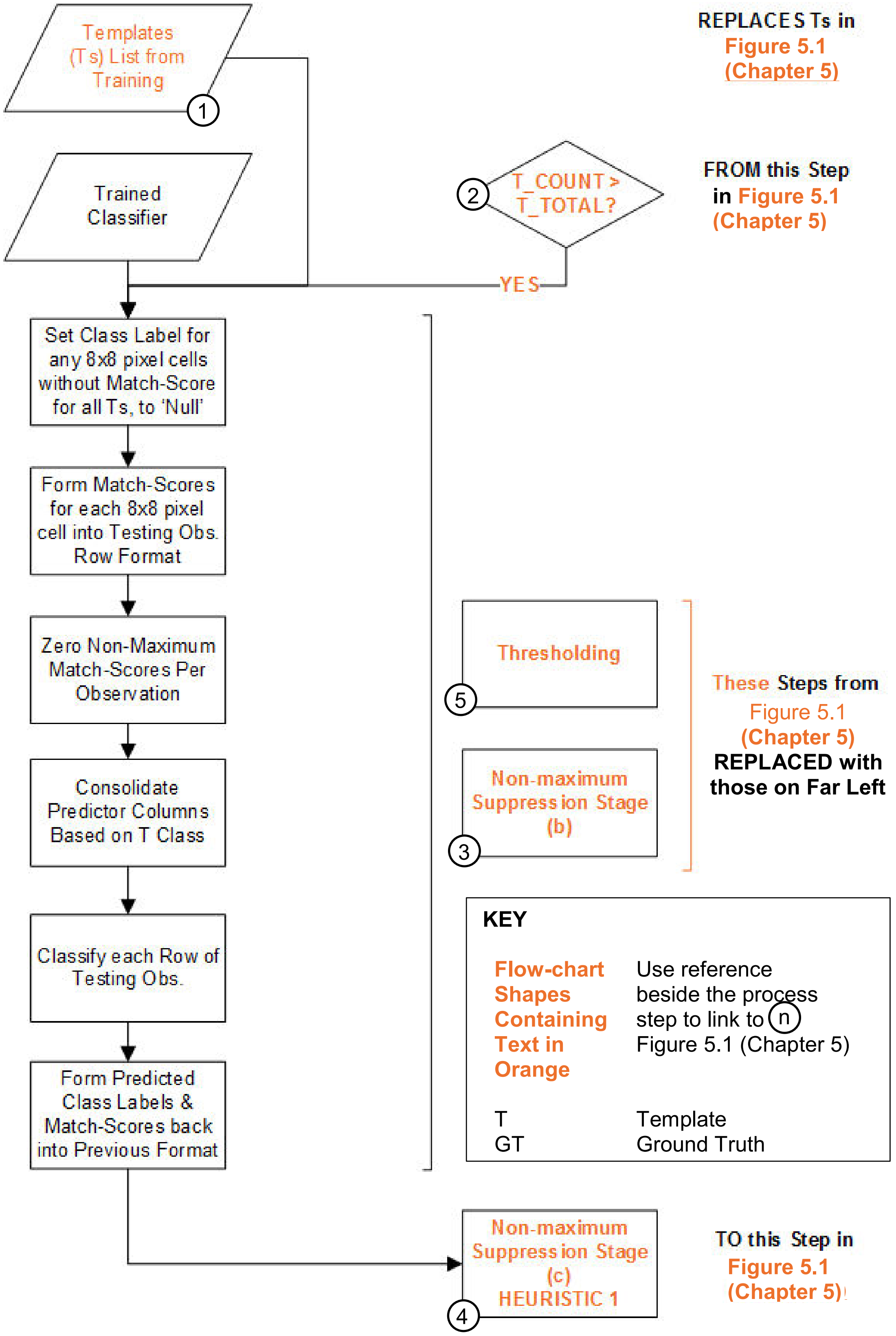

Templates (Ts) List from Training
1
Trained Classifier
REPLACES Ts in Figure 5.1 (Chapter 5)
2
T_COUNT > T_TOTAL?
FROM this Step in Figure 5.1 (Chapter 5)
YES
Set Class Label for any 8x8 pixel cells without Match-Score for all Ts, to 'Null'
Form Match-Scores for each 8x8 pixel cell into Testing Obs. Row Format
Zero Non-Maximum Match-Scores Per Observation
Consolidate Predictor Columns Based on T Class
Classify each Row of Testing Obs.
Form Predicted Class Labels & Match-Scores back into Previous Format
Thresholding
5
Non-maximum Suppression Stage (b)
3
These Steps from Figure 5.1 (Chapter 5) REPLACED with those on Far Left
KEY
Flow-chart Shapes Containing Text in Orange
Use reference beside the process step to link to n Figure 5.1 (Chapter 5)
T
Template
GT
Ground Truth
Non-maximum Suppression Stage (c) HEURISTIC 1
4
TO this Step in Figure 5.1 (Chapter 5)

Figure 6.6: Process Flow for Testing using Trained SVM Classifier
This figure is an expansion of the 'Testing' process step in Figure 6.1. Note also the linkage to Figure 5.1 from Chapter 5.

class. Any rows with negative-class labels were removed from the data. The match-score associated with the (x,y) pixel location of the corresponding candidate match for the predicted template class was then added back to each row. Where there was more than one template for a template class, the highest of the match-scores at the same (x,y) pixel location for the template class was chosen.

As shown in Figure 6.6 (specifically, its linkage to Figure 5.1 from Chapter 5), the above process operated on texture map images. Following classification, the pipeline had produced a matrix of match-scores, template classes and candidate-match template-boundary pixel locations. At this stage the process steps in Figure 5.1 in Chapter 5 could be picked up again. The first of these was non-maximum suppression stage (c).

6.4 Summary

This chapter has detailed the SVM approach, specifically its use for replacing the match-score thresholding step from the method in Chapter 5. The approach was based on the hypothesis that the combination of match-scores from different templates is representative of the class of the object on which the candidate match is centred. Training data, in the form of additional façade images, was used to gather training observations by conducting the HOG-based template matching approach using the templates already used in the methods in Chapter 5. By using an SVM in place of a thresholding approach the method becomes more scalable to unseen 3D building models and to other architectural styles. For the latter, this assumes sufficient suitable data is available.

Having now presented the methods for all of the work in this thesis, Chapter 7 will now present the results of this study, plus an evaluation.

Chapter 7

Experiments & Evaluation

7.1 Introduction

Chapters 3-6 described the methods used to enable semantic and geometric enrichment of existing 3D building models in the detection of windows and doors and their architectural style. This chapter presents the experimental results and an evaluation of those results. The pertinent questions are: have the heuristics improved the results, and has the use of machine learning matched or improved the results? Equally, how will the methods fare on unseen reference datasets? Below, the author details the strengths and challenges of the methods in practice and, also reveals how the method has the potential to identify detailed architectural design features in objects, provided that the image and template are consistent.

Note that the results of the following three trials: (a) image matching (SIFT), (b) standard template matching and (c) the clustering of templates, are not covered in this chapter on the basis that they remained trials and were not taken further in this work. For the results of these trials see Chapter 4 for (a) and (b), and Appendix D for (c).

7.2 HOG-based Template Matching (no SVM)

7.2.1 Experimental Results

The *lefthand-side* of each of the figures below show the candidate matches resulting from HOG-based template matching with no SVM, rendered onto the 3D building models, for the C20 (Figure 7.1 and Figure 7.2), Georgian-Regency (Figure 7.3 and Figure 7.4) and Norman styles (Figure 7.5). Note that the widths of the lines forming the bounding boxes representing candidate matches vary due to differing EXIF *IFD0:XResolution* or *IFD0:YResolution* in the source images. Note also, in Figure 7.2, that the pre-existence of detailed window geometry in BM_2_11 and BM_2_12 has cropped out some candidate match bounding boxes. Within the corresponding texture map images for those two 3D building models, the majority of window and door instances were successfully matched.

Table 7.1, Table 7.2 and Table 7.3 use bar charts to illustrate a comparison of the F-measure obtained with the different heuristics, and at different class granularities, for the 3D building models of the C20, Georgian-Regency and Norman styles respectively. For each 3D building model, the results are split into two sets: (R1) results obtained using templates covering all four styles and (R2) results obtained using templates covering just the style of objects found on the 3D building model. Within each of those sets, results are provided with no heuristics, with each of heuristics 1, 2 and 3 individually and with all three heuristics used together. Table 7.4 presents the equivalent information from the preceding tables, but as means for the C20, Georgian-Regency and Norman styles and as means for all 3D building models combined. In theory, for unseen 3D building models (R1) could initially be used to determine the potentially dominant architectural style, thus informing the choice of templates for obtaining (R2). In doing so (R2) should provide a more accurate detection of the architectural style of the windows and doors than (R1).

Table 7.5 shows the values for mean F-measures by style, and adds a delta value for each of ‘heuristic 1 only’, ‘heuristic 2 only’, ‘heuristic 3 only’ and ‘all heuristics’. Each delta value represents the difference in F-measure versus the use of no heuristics. Appendix E provides the equivalent to Table 7.5 but for each 3D building model. Table 7.6 provides the equivalent to Table 7.5 but for all 3D building models combined.

Note that the remainder of this chapter uses the following terms:

- ‘Template set (i)’

  Refers to the set of templates that comprises all four styles.

- ‘Template set (ii)’

  Refers to the set of templates of styles only found on the 3D building model. For the Georgian-Regency 3D building models this comprised just the Georgian-Regency template set. For the Norman 3D building models this constituted the Gothic and Norman template sets combined. For 10 of the C20 style 3D building models this comprised the C20 templates. For the remaining two C20 style 3D building models this constituted the Georgian-Regency templates. For one of those two 3D building models (BM_2_4), Georgian-Regency templates were used – the building was transitional Victorian era, where the Georgian-Regency template set included sash windows and doors similar to some of those typical of the Victorian era (1837-1901). For the other 3D building model in question, the architectural style was neo-Georgian (BM_2_7).

- ‘G1’, ‘G2’ and ‘G3’

  Refer to class granularity 1, class granularity 2 and class granularity 3 respectively.

- ‘H1’, ‘H2’ and ‘H3’

  Refer to heuristic 1, heuristic 2 and heuristic 3 respectively.

- Delta

  Refers to the difference between F-measure, where an example delta would be the difference between an F-measure obtained without heuristics versus an F-measure obtained with the use of all heuristics. Delta can be expressed as a percentage or as an absolute value.

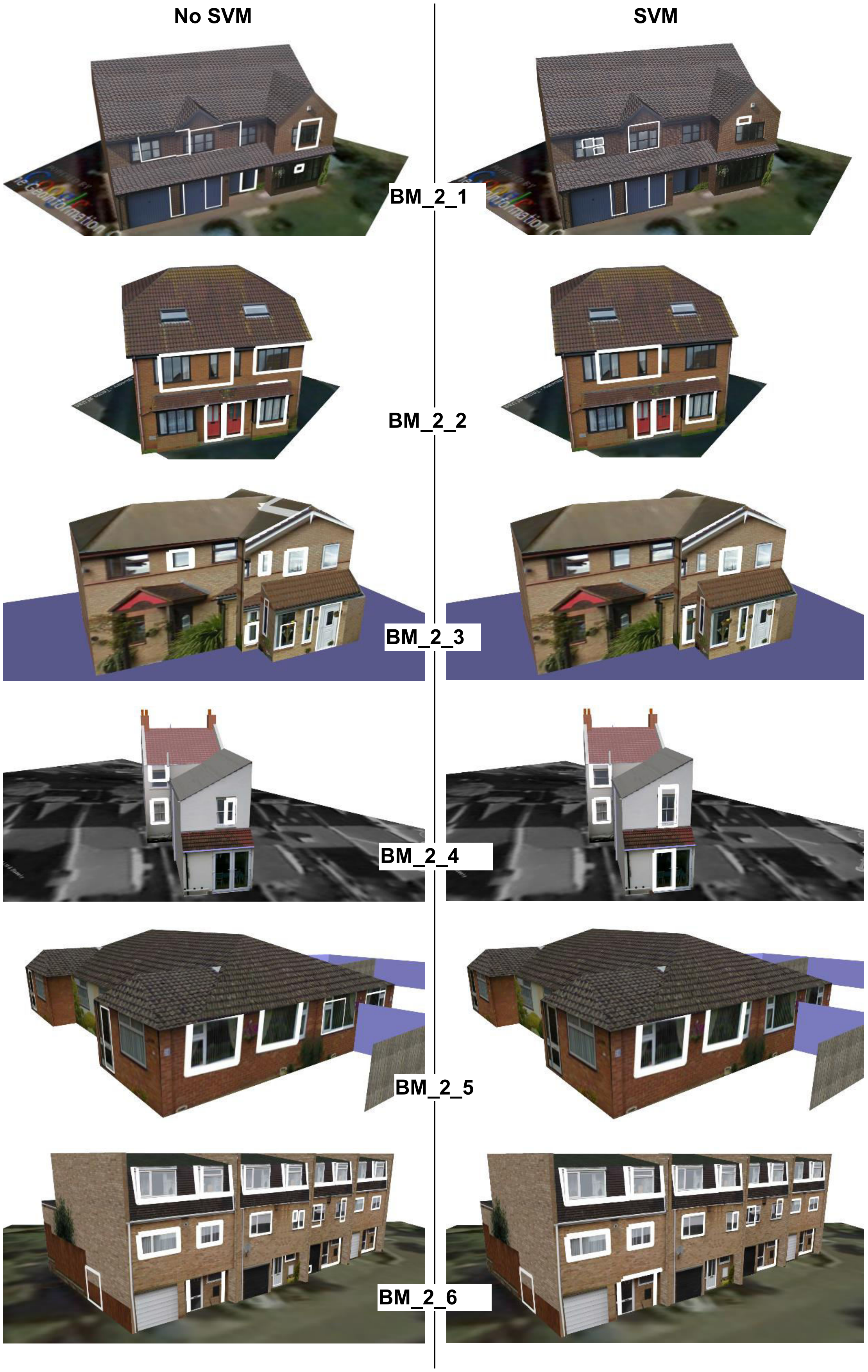


Figure 7.1: HOG-based Template Matching Results ('No SVM' Versus SVM) Rendered onto C20 3D Building Models (Figure 1 of 2)

Results above are using all heuristics and templates of a style only found on the 3D building models. Results for 'No SVM' on the left and results for SVM on the right. Candidate matches are rendered as white bounding boxes (or **red** bounding boxes for readability). See also Figure 7.2. See Figure 3.3, Chapter 3, for the same 3D building models but with no candidate matches.

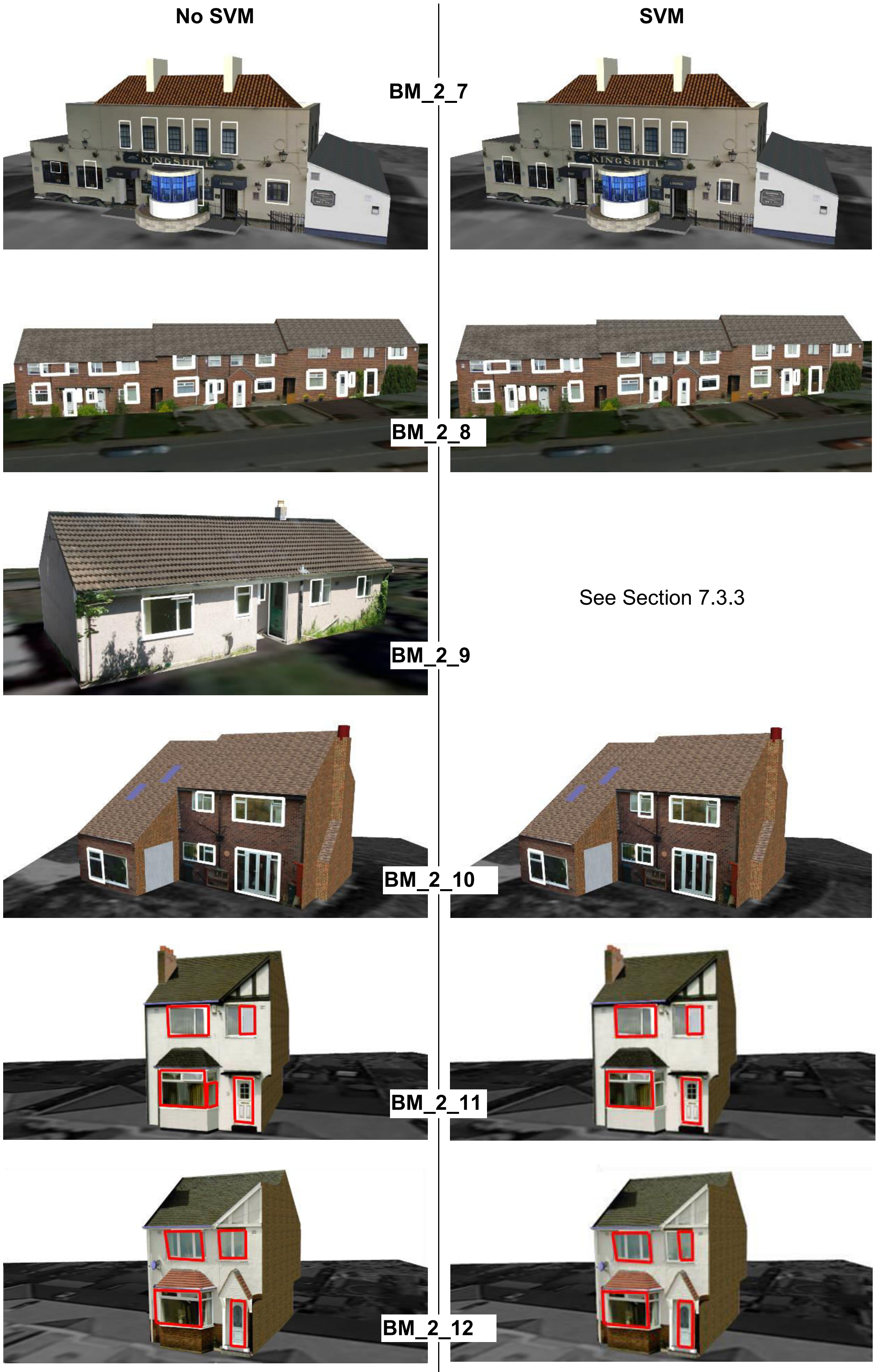
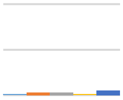
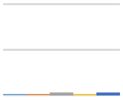
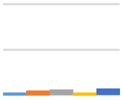
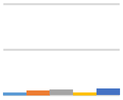
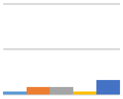
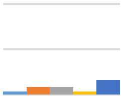
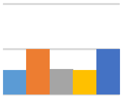
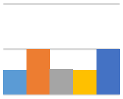
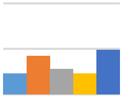
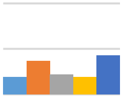
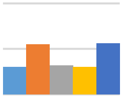
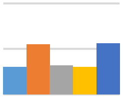
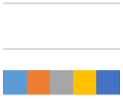
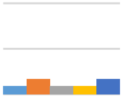
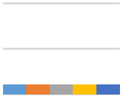
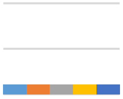
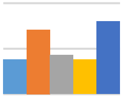
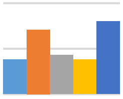
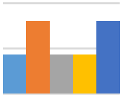
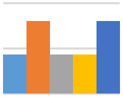
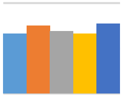
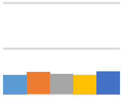
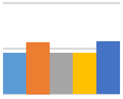
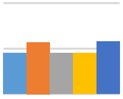
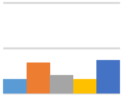
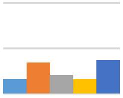
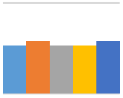
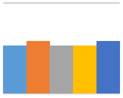
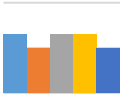
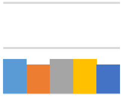
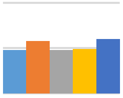
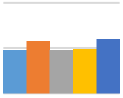
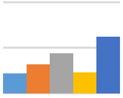
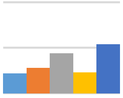
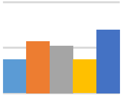
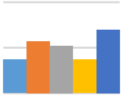
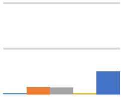
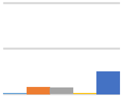
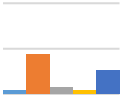
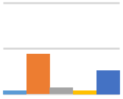
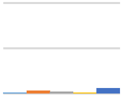
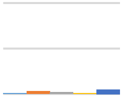
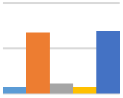
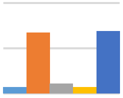
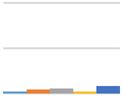
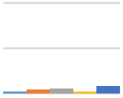
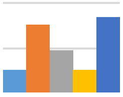
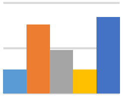
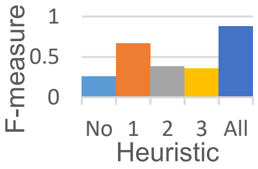


Figure 7.2: HOG-based Template Matching Results ('No SVM' Versus SVM) Rendered onto C20 3D Building Models (Figure 2 of 2)
See the caption for Figure 7.1. See also Figure 7.1. See Figure 3.3, Chapter 3, for the same 3D building models but with no candidate matches.

C20 style 3D Building Model	Template Set (i)			Template Set (ii)		
	G1	G2	(No G3)	G1	G2	(No G3)
BM_2_1						
BM_2_2						
BM_2_3						
BM_2_4						
BM_2_5						
BM_2_6						
BM_2_7						
BM_2_8						
BM_2_9						
BM_2_10						
BM_2_11						
BM_2_12						

KEY G Class Granularity (G1, G2, G3)



‘No’	No Heuristics
‘1’	Heuristic 1 Only
‘2’	Heuristic 2 Only
‘3’	Heuristic 3 Only
‘All’	All Heuristics

Table 7.1: Heuristics – Experimental Results – C20 Style 3D Building Models
See the caption for Table 7.2.

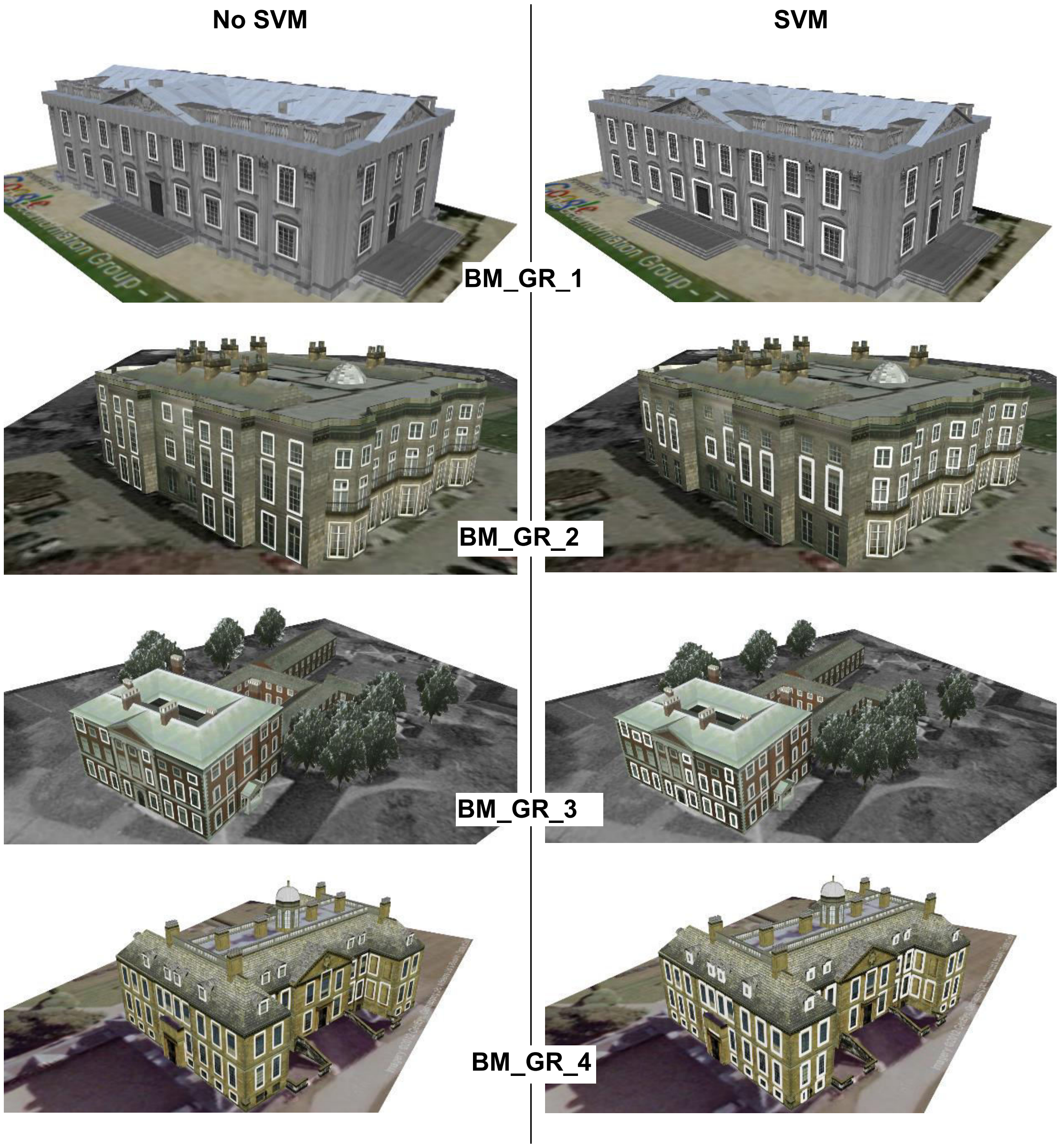


Figure 7.3: HOG-based Template Matching Results ('No SVM' Versus SVM) Rendered onto Georgian-Regency 3D Building Models (Figure 1 of 2)
See the caption for Figure 7.1. See also Figure 7.4. See Figure 3.4, Chapter 3, for the same 3D building models but with no candidate matches.

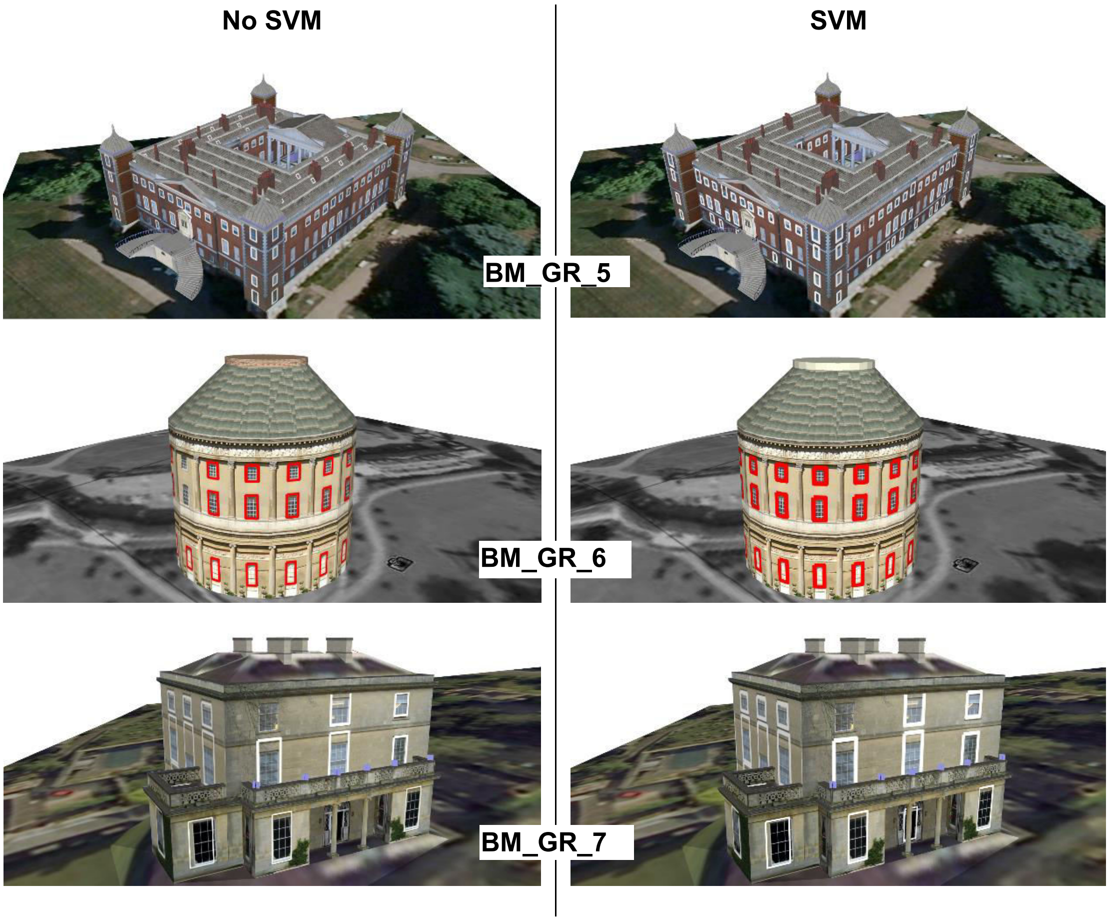
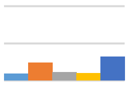
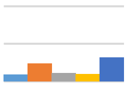
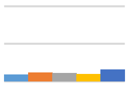
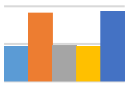
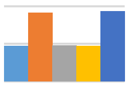
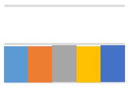
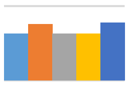
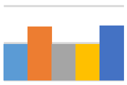
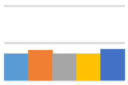
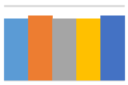
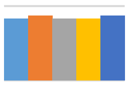
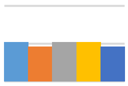
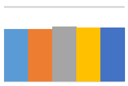
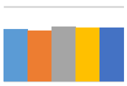
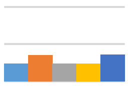
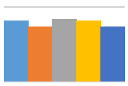
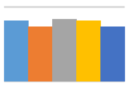
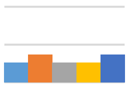
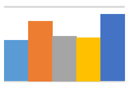
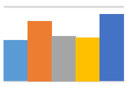
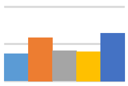
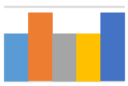
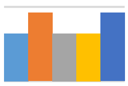
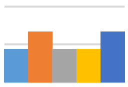
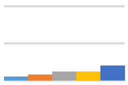
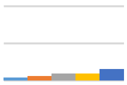
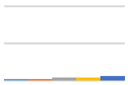
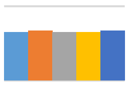
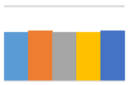
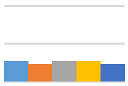
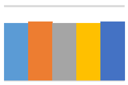
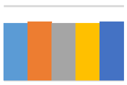
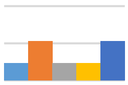
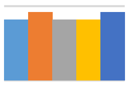
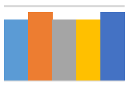
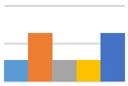
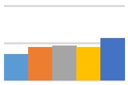
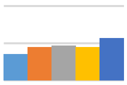
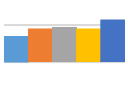
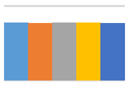
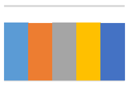
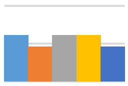
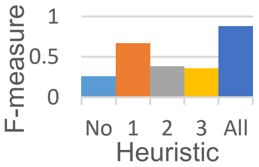


Figure 7.4: HOG-based Template Matching Results ('No SVM' Versus SVM) Rendered onto Georgian-Regency 3D Building Models (Figure 2 of 2)
See the caption for Figure 7.1. See also Figure 7.3. See Figure 3.4, Chapter 3, for the same 3D building models but with no candidate matches.

Georgian-Regency Style 3D Building Model	Template Set (i)			Template Set (ii)		
	G1	G2	G3	G1	G2	G3
BM_GR_1						
BM_GR_2						
BM_GR_3						
BM_GR_4						
BM_GR_5						
BM_GR_6						
BM_GR_7						

KEY

G Class Granularity (G1, G2, G3)



‘No’	No Heuristics
‘1’	Heuristic 1 Only
‘2’	Heuristic 2 Only
‘3’	Heuristic 3 Only
‘All’	All Heuristics

Table 7.2: Heuristics – Experimental Results – Georgian-Regency Style 3D Building Models Values are provided for running the method with template set (i), and with template set (ii). Each of those sets of results provides values at each class granularity. G1 and G2 for C20 and Georgian-Regency 3D building models are the same (only one style of template was used). There is no G3 for the C20 or Norman style 3D building models.

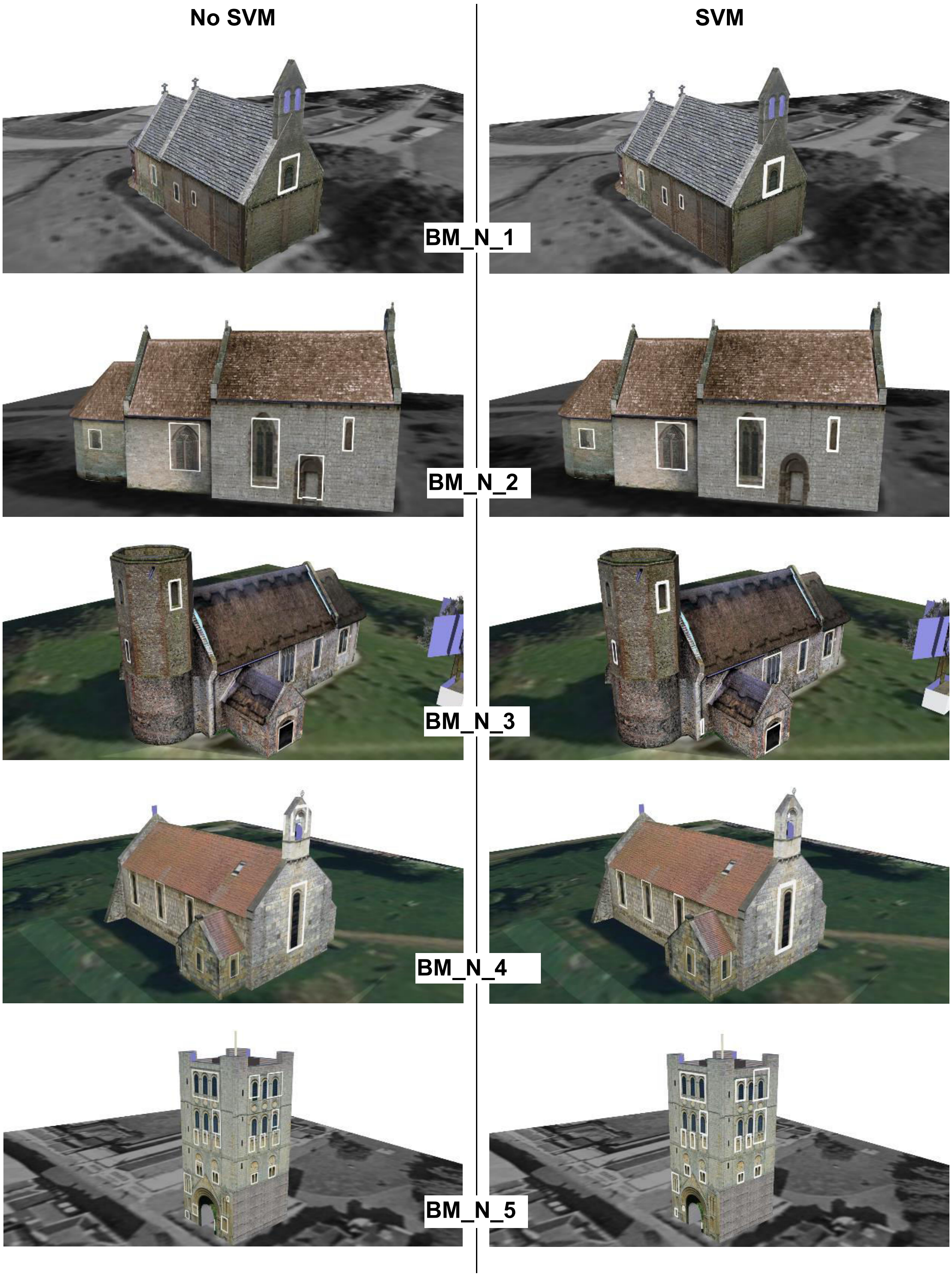
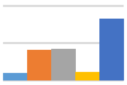
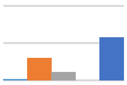
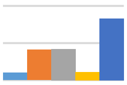
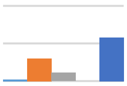
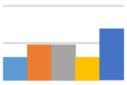
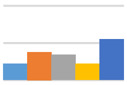
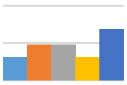
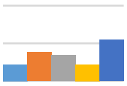
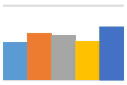
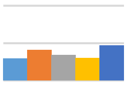
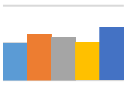
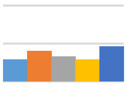
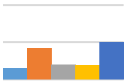
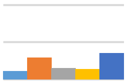
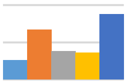
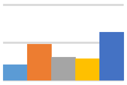
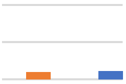
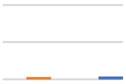
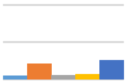
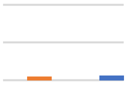
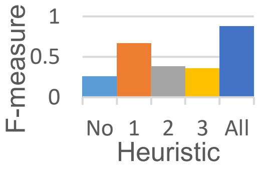


Figure 7.5: HOG-based Template Matching Results ('No SVM' Versus SVM) Rendered onto Norman 3D Building Models
See the caption for Figure 7.1. See Figure 3.5, Chapter 3, for the same 3D building models but with no candidate matches.

Norman Style 3D Building Model	Template Set (i)			Template Set (ii)		
	G1	G2	(No G3)	G1	G2	(No G3)
BM_N_1						
BM_N_2						
BM_N_3						
BM_N_4						
BM_N_5						

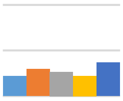
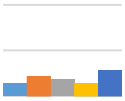
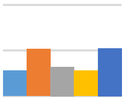
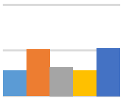
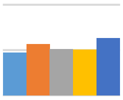
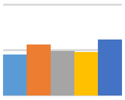
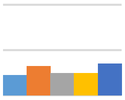
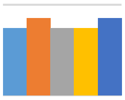
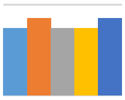
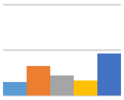
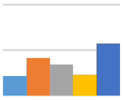
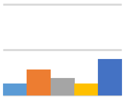
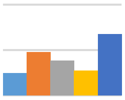
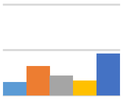
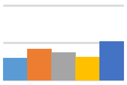
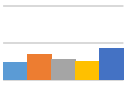
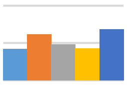
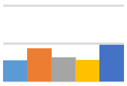
KEY

G Class Granularity (G1, G2, G3)



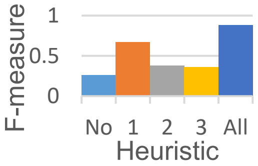
‘No’	No Heuristics
‘1’	Heuristic 1 Only
‘2’	Heuristic 2 Only
‘3’	Heuristic 3 Only
‘All’	All Heuristics

Table 7.3: Heuristics – Experimental Results – Norman Style 3D Building Models
See the caption for Table 7.2.

3D Building Model Style	Template Set (i)			Template Set (ii)		
	G1	G2	G3	G1	G2	G3
C20			**No G3**			**No G3**
Georgian-Regency						
Norman			**No G3**			**No G3**
All Styles						**No G3**

KEY

G Class Granularity (G1, G2, G3)



‘No’	No Heuristics
‘1’	Heuristic 1 Only
‘2’	Heuristic 2 Only
‘3’	Heuristic 3 Only
‘All’	All Heuristics

Table 7.4: Heuristics – Experimental Results – Mean by Style & Mean All Styles Combined See the caption for Table 7.2.

(a)

F-measures

	Heuristic (H)	Template Set (i)			Template Set (ii)	
		G1	G2		G1	G2
C20	No Hs	0.22	0.14		0.28	0.28
	H1 only	0.3	0.22		**0.52**	**0.52**
	H2 only	0.27	0.19		0.32	0.32
	H3 only	0.22	0.14		0.28	0.28
	All Hs	**0.37**	**0.29**		**0.52**	**0.52**
		G1	**G2**	**G3**	**G1 & G2**	**G3**
Georgian-Regency	No Hs	0.47	0.45	0.22	0.74	0.41
	H1 only	0.57	0.56	0.32	0.85	**0.47**
	H2 only	0.51	0.49	0.25	0.74	0.41
	H3 only	0.5	0.48	0.24	0.74	0.41
	All Hs	**0.63**	**0.62**	**0.35**	**0.86**	**0.47**
		G1	**G2**		**G1**	**G2**
Norman	No Hs	0.21	0.13		0.24	0.15
	H1 only	0.41	0.28		0.48	0.32
	H2 only	0.34	0.19		0.38	0.22
	H3 only	0.23	0.13		0.27	0.16
	All Hs	**0.57**	**0.40**		**0.67**	**0.46**

(b)

∆F-measure
'Hs' vs 'No Hs' (%)

	Heuristic (H)	Template Set (i)			Template Set (ii)	
		G1	G2		G1	G2
C20	No Hs					
	H1 only	35%	56%		85%	85%
	H2 only	21%	30%		15%	15%
	H3 only	0%	1%		0%	0%
	All Hs	67%	102%		88%	88%
		G1	**G2**	**G3**	**G1 & G2**	**G3**
Georgian-Regency	No Hs					
	H1 only	21%	25%	44%	15%	16%
	H2 only	9%	9%	11%	0%	0%
	H3 only	7%	7%	9%	0%	0%
	All Hs	34%	38%	57%	15%	16%
		G1	**G2**		**G1**	**G2**
Norman	No Hs					
	H1 only	91%	120%		96%	118%
	H2 only	59%	47%		57%	47%
	H3 only	6%	3%		11%	8%
	All Hs	166%	211%		176%	209%

(c)

∆F-measure
'Hs' vs 'No Hs' (Abs.)

	Heuristic (H)	Template Set (i)			Template Set (ii)	
		G1	G2		G1	G2
C20	No Hs					
	H1 only	0.08	0.08		0.24	0.24
	H2 only	0.05	0.04		0.04	0.04
	H3 only	0.00	0.00		0.00	0.00
	All Hs	0.15	0.15		0.25	0.25
		G1	**G2**	**G3**	**G1 & G2**	**G3**
Georgian-Regency	No Hs					
	H1 only	0.10	0.11	0.10	0.11	0.06
	H2 only	0.04	0.04	0.02	0.00	0.00
	H3 only	0.03	0.03	0.02	0.00	0.00
	All Hs	0.16	0.17	0.13	0.11	0.07
		G1	**G2**		**G1**	**G2**
Norman	No Hs					
	H1 only	0.19	0.15		0.23	0.17
	H2 only	0.13	0.06		0.14	0.07
	H3 only	0.01	0.00		0.03	0.01
	All Hs	0.36	0.27		0.43	0.31

KEY
Per building per granularity (G1, G2, G3) **bold** F-measure is highest

KEY
Green text: ∆ > 0 Black text: ∆ = 0 **Red** text: ∆ < 0

Table 7.5: HOG-based Template Matching (no SVM) – Heuristics Experimental Results – Mean by Style
See the caption for Table 7.6. G1 and G2 for C20 and Georgian-Regency 3D building models are the same (only one style of template was used). There is no G3 for the C20 or Norman style 3D building models (see Chapter 3).

(a)

F-measures

	Heuristic	Template Set (i)		Template Set (ii)	
	(H)	G1	G2	G1	G2
All Styles	No Hs	0.3	0.24	0.42	0.28
	H1 only	0.42	0.35	0.62	0.44
	H2 only	0.37	0.29	0.48	0.32
	H3 only	0.32	0.25	0.43	0.28
	All Hs	**0.52**	**0.43**	**0.68**	**0.49**

(b)
△F-measure
'Hs' vs 'No Hs' (%)

	Heuristic	Template Set (i)		Template Set (ii)	
	(H)	G1	G2	G1	G2
All Styles	No Hs				
	H1 only	410%	48%	46%	57%
	H2 only	24%	20%	15%	14%
	H3 only	5%	5%	2%	2%
	All Hs	74%	81%	63%	75%

(c)
△F-measure
'Hs' vs 'No Hs' (Abs.)

	Heuristic	Template Set (i)		Template Set (ii)	
	(H)	G1	G2	G1	G2
All Styles	No Hs				
	H1 only	0.12	0.11	0.19	0.16
	H2 only	0.07	0.05	0.06	0.04
	H3 only	0.02	0.01	0.01	0.00
	All Hs	0.22	0.20	0.26	0.21

KEY
Per building per granularity (G1, G2, G3) **bold** F-measure is highest

KEY
Green text: △ > 0 Black text: △ = 0 **Red** text: △ < 0

Table 7.6: HOG-based Template Matching (no SVM) – Heuristics Experimental Results – Mean All Styles Combined
Values are provided for running the method with template set (i), and with template set (ii). Each of those sets of results provides values at each class granularity i.e. G1 and G2. Sub-table (*a*) contains F-measure results (vertically) per style (or for all styles combined) and per granularity. The highest F-measure for a per-style (or for all styles combined), per-granularity set is highlighted in **bold**. Sub-tables (*b*) and (*c*) show the delta (△) in F-measure versus the use of no heuristics, in percent and as an absolute value respectively.

7.2.2 Results Summary – F-measures

The mean F-measures, with template set (ii), with all three heuristics were:

- 0.68 (G1) and 0.49 (G2) – All 3D building models
- 0.52 (G1 and G2) – C20 style 3D building models
- 0.86 (G1 and G2) and 0.47 (G3) – Georgian-Regency 3D building models
- 0.67 (G1) and 0.46 (G2) – Norman 3D building models

7.2.3 Results Summary – Template Choice

Compared to template set (i), the use of template set (ii) consistently resulted in an equal or higher F-measure. The improvement in F-measure when using template set (ii) was most marked for Georgian-Regency 3D building models, whereby the results for all seven 3D building models improved. For Norman, three out of five improved. For C20, 7 out of 12 improved.

7.2.4 Results Summary – Heuristic 1

H1 rejected higher scoring overlapping candidate matches if they were too small. Based on mean delta, H1 was the most effective heuristic: the mean delta for all styles of 3D building model was 0.15 (48%). H1 was the most successful heuristic across all styles, generally resulting in a positive delta. H1 was generally most successful for the C20 style. The highest absolute delta across all 24 3D building models was 0.6, for BM_2_11 (857%, template set (ii), G1 and G2). Appendix F contains an extended results summary for H1.

7.2.5 Results Summary – Heuristic 2

H2 rejected candidate matches that were too tall or too short. H2 was the second most successful heuristic, with a mean absolute F-measure delta of 0.05 (18%). H2 was the second-most successful heuristic across all the styles and did not result in any negative deltas. H2 was generally most successful for the C20 style. The highest absolute delta obtained for any of the 24 3D building models was for BM_2_9 (0.22, template set (i), G1 and G2). Appendix F contains an extended results summary for H2.

7.2.6 Results Summary – Heuristic 3

H3 rejected candidate ‘door’ matches that were too high up the building. H3 was, based on mean F-measure, the least successful of the heuristics for all the styles. The mean absolute delta obtained with H3 was 0.01 (3%).The highest absolute delta was 0.1, for BM_N_4 (38%, template set (ii), G1 and G2). For BM_N_1, H3 resulted in a small negative absolute delta: -0.02 (-100%, for each set of templates at G2), but this is a negligible reduction when one considers that an F-measure will always be in the range [0,1]. Aside from this, H3 did not degrade the results. Appendix F contains an extended results summary for H3.

7.2.7 Results Summary – All Heuristics

Through comparison of the mean F-measure when using (S1) each of the heuristics individually and using (S2) all three heuristics together, (S2) was the most successful. It had a mean F-measure of 0.22, versus 0.15, 0.05 and 0.01 for the sole use of each of H1, H2 and H3 respectively. Compared to (S1), (S2) was also the most successful for each style, generally resulting in a positive absolute delta. The combination of all heuristics was the most successful for the Norman style, with a mean delta of 0.34, compared to 0.2 and 0.16 respectively for C20 and Georgian-Regency. The largest positive absolute delta for any of the 3D building models was 0.73 for BM_N_1 (730%, template set (i), G1). Appendix F contains an extended results summary for the use of all three heuristics in combination.

7.2.8 Runtime

Processing all 24 3D building models took just over 4.5 hours. These timings were obtained using template set (ii). Runtimes using templates in template set (i) were higher (see below). Appendix H contains details of the runtimes split by architectural style and by 3D building model.

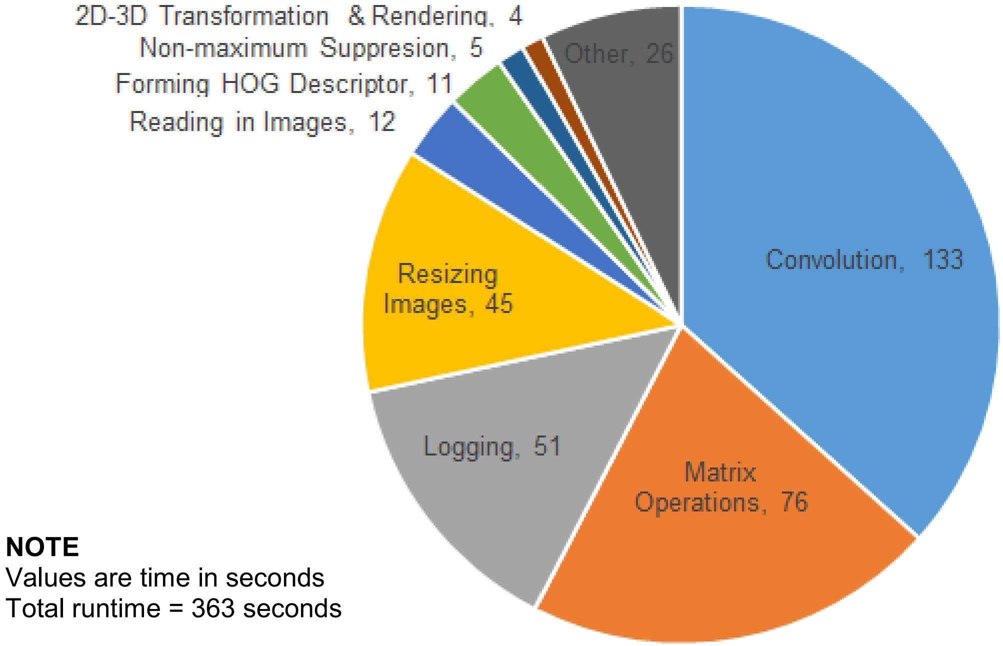


Figure 7.6: Split of Runtime by Process – no SVM
The 3D building model used for the timings was **BM_N_4**, with template set (ii). Compare with Figure 7.11 i.e. the process timings when no SVM was used, for the same 3D building model. Note that total runtime is not consistent with that in Appendix H due to the additional overhead resulting from timing the processes.

Figure 7.6 illustrates the approximate proportion of time spent in the various processes of the pipeline, using one of the 3D building models – note how the convolution, followed by matrix processing, use the largest proportions of runtime. Once again, all the results were achieved on a virtual machine running a single processor from an Intel® *Xeon*® E5-1620 3.6GHz host processor, and 24GB of RAM from the host. Outputting texture map images marked up with the candidate matches, ready for use in a viewer, would take longer.

In general, runtime was a function of 3D building model complexity, as defined by the number of *gml:LinearRing*s and the number of texture map images. See Chapter 3 for an indication of the complexity of the 3D building models used. Texture map image size also resulted in a longer runtime. To a lesser extent, using a greater number of templates increased runtime.

When a high number of false positives with match-scores above the thresholds occurred, this could also increase runtime. Such a scenario was found to occur where a texture map image and a much smaller template contained similarly spaced, repeating horizontal and vertical edges. Specifically, the scenario comprised lines in the texture map image which were *not* from window or door objects, where that pattern of lines was greatly repeated across the image. Examples of such a scenario include texture map images of tiled roofs with lots of small tiles, or of brickwork with coherent mortar lines and lots of small bricks, when combined with smaller versions of Georgian-Regency window templates (see below).

7.2.9 Discussion – Overview

To reiterate, the use of all three heuristics together generally gave the best results. When used in isolation H1 was generally the strongest of the three heuristics, compared to the use of either no heuristics or each of the other heuristics in isolation. Importantly, using all three heuristics together produced the strongest F-measure, or very close it, for each 3D building model. Arguably, by running the combined set of heuristics in the pipeline, the occasional tendency of the heuristics (when used in isolation) to remove true positives in error is diminished. Note that the pipeline processed H1 first, followed by H2 then H3.

F-measures when using template set (i), with all three heuristics in combination, were either the same or worse compared to the use of template set (ii). This therefore supports the theory that running the method first with template set (i), and then again using template set (ii), is the best strategy. That said, results for template set (i) for some 3D building models, such as BM_GR_1 and BM_GR_5, would make it a challenge to determine the architectural style of templates to include in template set (ii).

With regard to the quality of texture map images, Figure 7.8 demonstrates the positive effect of replacing an original low quality (small, not sharp) texture map image with a higher quality (larger, sharper) one. Specifically, where the original texture map image for the 3D building model (BM_GR_6) did not result in any candidate matches, the use of a higher quality one resulted in candidate matches. The G3 F-measure obtained with template set (ii) for BM_GR_6 is, incidentally, the second highest G3 F-measure for any 3D building model. The practicalities of replacing texture map images, and the effect of doing so on the scalability of the method to unseen 3D building models, is discussed in the following sections.

Even when using an optimal set of templates with all three heuristics, the F-measures obtained between the different styles of 3D building models were still variable. This was especially true for the C20 and Georgian-Regency 3D building models versus the Norman ones. In general terms, the method was most successful on window objects on Georgian-Regency style 3D building models – this was essentially due to the standardisation of Georgian-Regency window designs. Specifically, such standardisation was represented in common aspect ratios of the boundaries of the sash window's outer lining and glazing panels, the configurations of the glazing panels and the use of plain glazing.

In contrast, other objects contained greater variation within their designs. The variability of F-measure was also present within each style-set of 3D building models. The variation in designs is discussed in the following sections.

The following phenomena can reduce F-measure, and can therefore explain the variability in the results:

(1) False negatives

(2) False positives for all class granularities

(3) True positives only at higher class granularities

Note that G1 is the highest granularity, and G3 the lowest. Figure 7.7 illustrates a selection of examples of (1) and (2), using the images and HOG descriptor gridded polar plots for texture map images and templates. The causes of the phenomena are discussed below. Note that Georgian-Regency and Norman 3D building models are used to illustrate and discuss the phenomena in question, although many of the effects described below also hold true, broadly, for the C20 3D building models. For the C20 style, in summary, false positives and false negatives could occur due to the variability of some C20 style window designs found in the 3D building models. Such variability was especially apparent in the configuration of the asymmetric splits of the glazing bars, in addition to the presence of a wide variety of aspect ratios for window frame proportions.

7.2.10 Discussion – False Negatives

False negatives are essentially the failure to obtain a match where there should be a match. Examples of this can be seen in Figure 7.7 for the following examples: (a) for the two right-most window instances on the texture map image, and (b) for the central window instance on the texture map image.

For (a) note how the gridded polar-plot of the HOG descriptor of the template shown indicates three roughly coherent vertical lines. These lines are similar in appearance to the left-most window instance (which achieved a successful match). However, they are less similar in appearance to the gridded polar plot of the HOG descriptor for the other two window instances in that they indicate fewer than three coherent vertical lines.

With respect to (b) note how the gridded polar plot of the HOG descriptor for the texture map image is not a good match for the template on the far right (which should, ideally,

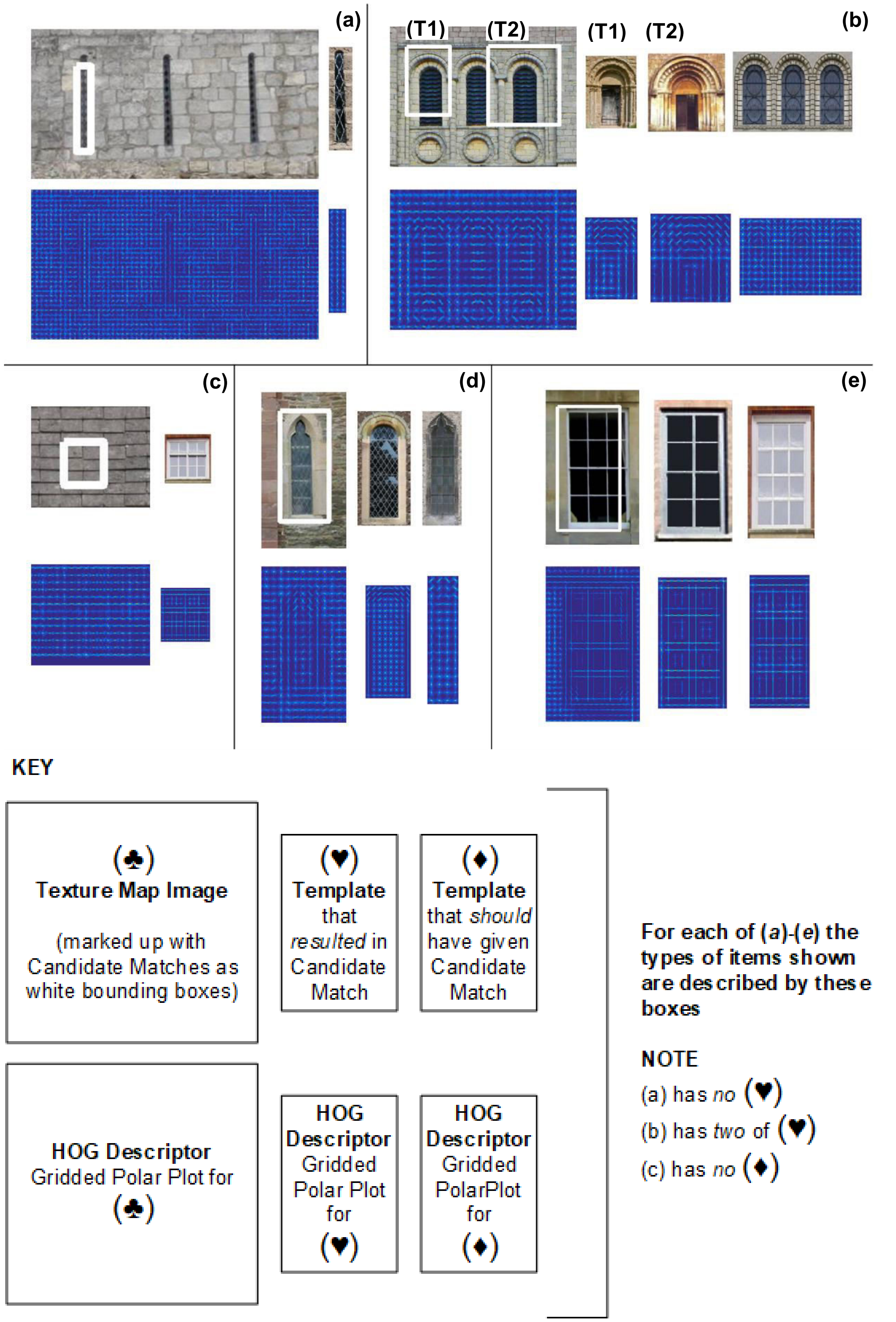

(a)
(T1)
(T2)
(T1)
(T2)
(b)
(c)
(d)
(e)
KEY
(♣)
Texture Map Image
(marked up with Candidate Matches as white bounding boxes)
(♥)
Template that *resulted* in Candidate Match
(♦)
Template that *should* have given Candidate Match
HOG Descriptor Gridded Polar Plot for (♣)
HOG Descriptor Gridded Polar Plot for (♥)
HOG Descriptor Gridded PolarPlot for (♦)
For each of (*a*)-(*e*) the types of items shown are described by these boxes
NOTE
(a) has *no* (♥)
(b) has *two* of (♥)
(c) has *no* (♦)

Figure 7.7: Example Reasons for False Positives & False Negatives
The following 3D building models and templates are used above. (*a*) **BM_N_4**, TW_N_09. (*b*) **BM_N_5**, TD_N_37, TD_N_33, TW_N_15. (*c*) **BM_GR_5**, TW_GR_2x2x3. (*d*) **BM_N_1**, TW_N_10, TW_GO_18. (*e*) **BM_GR_7,** TW_GR_2x4, TW_GR_3x4. Note that the HOG descriptor gridded polar plot for the texture maps was selected from a set for the corresponding image pyramid.

have resulted in a match) in the centre region of the windows. Specifically, the grating on the texture map image window instances does not have a HOG descriptor sufficiently like the HOG descriptor for the style of the leadwork in the glazing of the template image. In the case of (b) the non-maximum suppression step may also have removed overlapping matches. As a general principle, the method used in this work was designed to filter out overlapping window or door candidate matches, which would of course be a highly unusual occurrence in the real world. Such overlaps would also conflict with CityGML standards.

At this point it is worth discussing the general effect of glazing types. As a reminder, some of the 'Georgian-Regency Style Window' templates had their glazing manually blanked out during pre-processing (see Chapter 3). Doing so improved the results of HOG-based template matching compared to those using the original template images. Broadly, it could be argued, blanking should also have been conducted on the regions that could represent glazing on any corresponding candidate matches on the façade or texture map images. To do so, one might use the regions of the glazing on the template to form a mask during pre-processing, which would then be applied to the candidate matches as part of the pipeline. Specifically, pixel intensity values in the mask area would be zeroed.

However, there could be possible complications to the blanking-out of glazing. Firstly, the gridding of the HOG descriptor could lead to issues of over- or under-extending the mask. Secondly, notwithstanding the points in Chapter 3 regarding the lack of success with Gothic or 'Norman Style Window' templates using blanked-out glazing, the application of a mask to window objects for those styles could be complex. This is because such window designs often possess multi-tiered, multi-light structure and intricate tracery designs for Gothic, and non-plain glazing for Norman. While the variability in the designs of the glazing for those two styles clearly presents challenges for the method, it could be argued that these (which, unlike Georgian-Regency glazing, is generally *not* plain) can sometimes be usefully discriminative. As such, this might explain the poor performance of the early experiments mentioned in Chapter 3, with manually blanked-out glazing for those styles.

In summary, if the magnitude and position of edges that form a window (or door) instance on a texture map image do not sufficiently match the magnitude and position of edges in the template, then a false negative may result. This inconsistency between edges in window (or door) instances on the texture map image, and in the template, can be due to the variability in the design or construction method of the window (or door). In turn this can mean that the template is not sufficiently like the instance on the texture map image. Further examples of such variability are revisited in the following sections. Note too that the likelihood of a false negative can increase if the texture map image or template are low quality – essentially a lower quality image loses the design details.

7.2.11 Discussion – False Positives at all Class Granularities

The most common reasons for false positives, at all F-measure granularities, were: (Rsn1) where a door or window template resulted in a candidate match on a non-window or non-door architectural feature; (Rsn2) where a door template resulted in a candidate match against an instance of a window, or vice versa; and (Rsn3) where the candidate match was too small, meaning it failed the *PASCAL VOC* overlap check.

Figure 7.7 (c) shows an example of (Rsn1) as a false positive on a roof tile texture map image for BM_GR_5. Due to repetition of the candidate match across *gml:LinearRings*, 12 false positives resulted across the 3D building model. This explains the lower F-measure when compared to the other Georgian-Regency buildings. These false positives can be seen on the pitched roofs on BM_GR_5 in Figure 3.4 in Chapter 3. Note in Figure 7.7 (c) how the gridded polar plots both for the texture map image in the region of the false positive, and for the template that resulted in the false positive, are similar in appearance. Specifically, the vertical and horizontal lines of the outer lining / parting bead / stile[1], meeting rail, bottom rail and glazing bars of the sash window in the template produced a similar HOG descriptor to the lines formed by the boundaries of the roof tiles in the texture map image.

False positives were also observed on the blind windows in BM_N_5, though these were originally blind at the point of construction (Historic England, 2017), so were not marked up as ground truths. See Chapter 3 for an explanation of why such a ground-truthing approach was adopted.

Figure 7.7 (b) shows an example of (Rsn2). The two templates that resulted in the matches were door templates, when in fact the texture map image contains instances of an arcaded triple window (see below). Again, note in Figure 7.7 (b) that the gridded polar plots for both the texture map image in the region of the window instances, and for the templates that resulted in the false positive, are similar in appearance. Note also how the gridded polar plots for those templates are more similar to the gridded polar plot for the region in the texture map image, than to the plots for the template that should have

[1] A component part of a sash window, as defined in Chapter 2, and not to be confused with ‘style’.

resulted in the match. It could be argued that the biggest difference, visually at least, between the instances in the texture map image and the templates is that the former are windows and the latter are doors! Nevertheless, one suggests that, to the human observer, Norman windows and Norman doors can appear to have similar structure and designs in their orders, as noted in Chapter 3. However, in the case of (Rsn2) there are still design differences in the orders found in the instances and the orders in the template. These include: a different number of orders (three and four, versus two); different moulding styles (chevron versus plain and ‘roll’); differing decorations on the outer boundary of the arch of the outer order (hoodmoulds and chamfers versus no decoration); differing brickwork configurations in the shafts; and different designs of capitals (see Appendix A for a link to the original source images). Despite these differences the false positives still occurred. Essentially, the detail in the design variations within the orders is not apparent in the HOG descriptor, perhaps as a consequence of the small size of the template for what are complex designs. Note also how the grating on the window instances produces a HOG descriptor similar to the planks used for the Norman doors.

One texture map image for BM_N_3 (not used in Figure 7.7) also produced a false positive of a door on a window instance. In this case, the texture map image was of lower pixel dimensions and sharpness.

Examples of (Rsn3) were found on BM_N_1 and BM_N_3. Both of those 3D building models also contained texture map images that were of lower pixel dimensions and sharpness.

In summary, where the HOG descriptor for a template is sufficiently similar to that on a region on a texture map image, then the potential exists to either: correctly identify the instance (a true positive); or to incorrectly identify an instance as the wrong class or style or to achieve a match that is too small (false positives).

7.2.12 Discussion – True Positives only at Higher Class Granularities

Scenarios also occurred in which a true positive at G1 was achieved, whereby the instance was correctly identified as a window or door, but a false positive at G2 or at G3 was obtained. Such a scenario could include failing to identify the style of the window or door (G2) or failing to detect a sub-sub-class such as ‘3x3 pane’ window correctly (G3). Figure 7.7 (d) and (e) illustrate such scenarios.

Figure 7.7 (d) demonstrates how a Norman window template has incorrectly matched to a Gothic window instance. The example illustrates how distinguishing between Norman and Gothic objects can sometimes prove problematic. In the case of (d) the HOG descriptors for the templates and window instance fail to pick out the subtle distinction between a pointed arch, and a round-headed arch at the top of Gothic and Norman windows respectively. Moreover, while the number of lights has been correctly detected, the HOG descriptors do not pick out the distinction between the round-headed light in the Norman window template and the trefoil-headed light in the Gothic template and window instance. Indeed, as Lee *et al.* (2015) observed generally, the distinction between architectural styles is not always clear cut.

Figure 7.7 (e) illustrates how lines which are less coherent, as indicated in the HOG descriptor gridded polar plot of the texture map image, can still result in a partial (and offset) candidate match. In this case the window instance on the texture map image was a ‘6-over-6’ sash window, and the template a ‘4-over-4’ sash window. Once again, scenarios such as those found in Figure 7.7 (d) and (e) are more likely to occur where the texture map image (or template) pixel dimensions are small, or when image sharpness is low.

7.2.13 Discussion – Other Challenges

There are a number of other challenges for the method used here:

(C1) The potential to double-count candidate matches used in the F-measure calculation due to the relative positions of ground truths and candidate matches

(C2) Architectural rhythm leading to repeating window instances with very small intervals between the windows

(C3) A rise in unexpected false positives if template choice was not carefully considered

(C4) The potential uniqueness of Gothic and Norman window or door instances (therefore not matching any template)

(C5) The time overhead for replacing texture map images

(C6) The time taken to produce templates

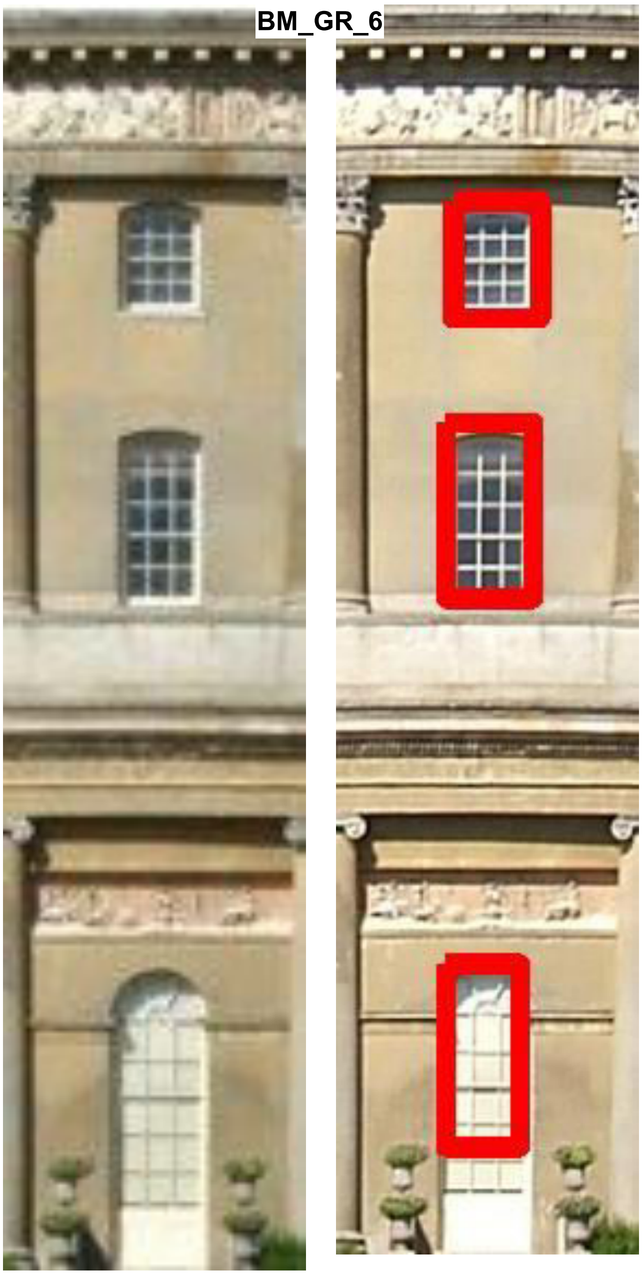
These challenges are discussed below.

(C1) In theory, the method used in this work still has the potential for overlapping candidate matches to persist. If this overlap does occur then the F-measure calculation may double-count, depending on the overlap amount, under two scenarios. The first is the potential to double-count a true positive if a single match overlaps two ground truths. The second scenario is the potential to double-count true positives if two matches overlap a single ground truth. To put this in context, only two examples of double-counting due to overlap occurred: (E1) an overlapping pair of matches on a façade texture map image for BM_GR_2 and (E2) an overlapping pair of matches on a lead-flashing texture map image for BM_N_4. Each candidate match in (E1) was only counted once for the F-measure calculation on the basis that, for each match, the 50% overlap threshold was only met for one ground truth. Both matches in (E2) were filtered out by heuristic 2, so were not considered during the F-measure calculation.

(C2) Overlapping candidate matches occurred in early experiments on BM_N_5 as a result of the small intervals between the stages / registers (rows) of windows, and shared colonettes (columns). Windows of this design are often referred to architecturally as arcaded windows. Such an overlap would have been an issue for the calculation of F-measure, in that matches or ground truths could have been counted twice in error. The size of the non-maximum-suppression neighbourhood was therefore reduced in early experiments, from twice the width and twice the height of the template to 1.5 times the width and height. In practice, doing so reduced F-measure for most of the 3D building models, even those without overlaps, noting that only BM_N_5 initially resulted in overlaps. Subsequently, templates representing arcaded windows were included in the template sets used in the results in this chapter. While this did not consistently achieve matches, one instance on BM_N_5 was successfully matched.

(C3) Using more templates in an attempt to achieve a better F-measure often actually made the results worse, due to an increase in the occurence of false positives. Consequently, a relatively conservative approach was adopted when selecting the template sets used for the experiments in this chapter.

(C4) In general, the component make-up of Gothic, and to an extent Norman, windows and doors is, as mentioned in Chapter 2, more complex than for C20 or Georgian-Regency windows and doors. Indeed, architectural historians have stated that, for many older Gothic and Norman buildings, individual instances of windows and doors on a building are unique at the lowest level of design detail. An example of this in the work undertaken for this thesis is the unique porch on BM_N_5 (Tymms and Thompson, 1855). This uniqueness presents challenges for the method, both in terms of finding or generating templates and achieving matches.



(Original)

ORIGINAL
TEXTURE MAP IMAGE

Dimensions: 66×276 px

(Replacement)

REPLACEMENT
TEXTURE MAP IMAGE

Dimensions: 170×697 px

Figure 7.8: Effect of Texture Map Image Replacement on Object Detection
(*Original*) The original texture map image as downloaded with the model from the Trimble *3D Warehouse.* When using the original texture map image no windows were detected. (*Replacement*) The texture map image used to replace the original, and the detection of window instances, shown as **red** bounding boxes. When using the replacement texture map image the top two window instances were successfully detected, while the bottom window instances obtained a less strong match. Note the larger size and correspondingly higher quality, of the replacement.

(C5) As outlined in Chapter 3, some texture map images were replaced, which was often a labour-intensive task. Optimising all texture map images would likely improve the results, as indicated by Figure 7.8. Unfortunately, to do so would negate the potential for the method to be scalable to other, unseen, 3D building models, due to the time penalty.

(C6) As a reminder, some templates needed to be hand-crafted due to the large variety and specificity of designs found on the 3D building models in testing. Note once again that choosing templates in this manner is a standard approach. Thanks to their more standardised design, when compared to the other styles, it was easier to modify Georgian-Regency templates. New variants could be created through the use of a simple copy-paste operation to add rows or columns of glazing panels. However, it was not as straightforward to do so for C20, Norman and Gothic templates, due to their comparatively greater component variability.

Equally, the greater variability in Georgian-Regency door designs, versus Georgian-Regency window designs made it harder to produce new Georgian-Regency door templates. Such variation includes: the use of the different orders (Doric, Tuscan, Ionic, Corinthian or Composite) and the associated variability in the designs of entablatures, columns, shafts, capitals, cornices, friezes and architraves; the presence of a portico and whether it was *in antis* or prostyle; the presence of a pediment and whether its design is open, segmental or broken; and the presence and design of a fanlight. Finally, the nature of the panelling configuration in different examples of Georgian-Regency doors also shows variation. A further consequence of this variability was that Georgian-Regency doors were consistently poorly detected with the methods in this work, compared to windows of the same style.

7.2.14 Discussion – Unexpected Findings

Despite not choosing to calculate F-measure at what might be considered G3 for Norman 3D building models, the method used in this work did actually achieve some strong G3 matches, especially for some instances of windows on BM_N_2, and BM_2_3. See Figure 7.9, and a description of the (U1) and (U2) examples in the figure, below. Note that the match-scores for each example are well above the Gothic style window threshold of 0.26, where a match-score nearer 0.3 tends toward the highest match-scores achieved for any texture map images for any of the 3D building models.

Example (U1) in the figure is successfully detecting a Gothic bar tracery window with: the same number of lights; the same style of hoodmould; the presence of a central point-cusped foil in the arched region of the tracery; and even similar brickwork at the edges of the window. That said, the glazing style is different between instance and template. In addition, the arch contains a *tre*foil while the template uses a *quatre*foil. However, arguably, due to the correct detection of the overall structure and design of the window, the match is still a strong one.

Meanwhile, example (U2) in the figure demonstrates the correct detection of a Gothic Y-tracery window (two-light intersecting tracery). The match was achieved despite the presence of a window on the opposite aspect of the building, seen through the glazing in the instance, and a difference in glazing between the instance and template.

In addition, while there were some challenges in doing so (see above), the method used in this work is matching some windows where glazing has been replaced with covers, such as wooden boards, wooden mesh, metal panels or louvres. This is being achieved even without the use of templates where the glazing has been replaced. For those instances that have been missed, it is fair to say that there is such a variety of architectural

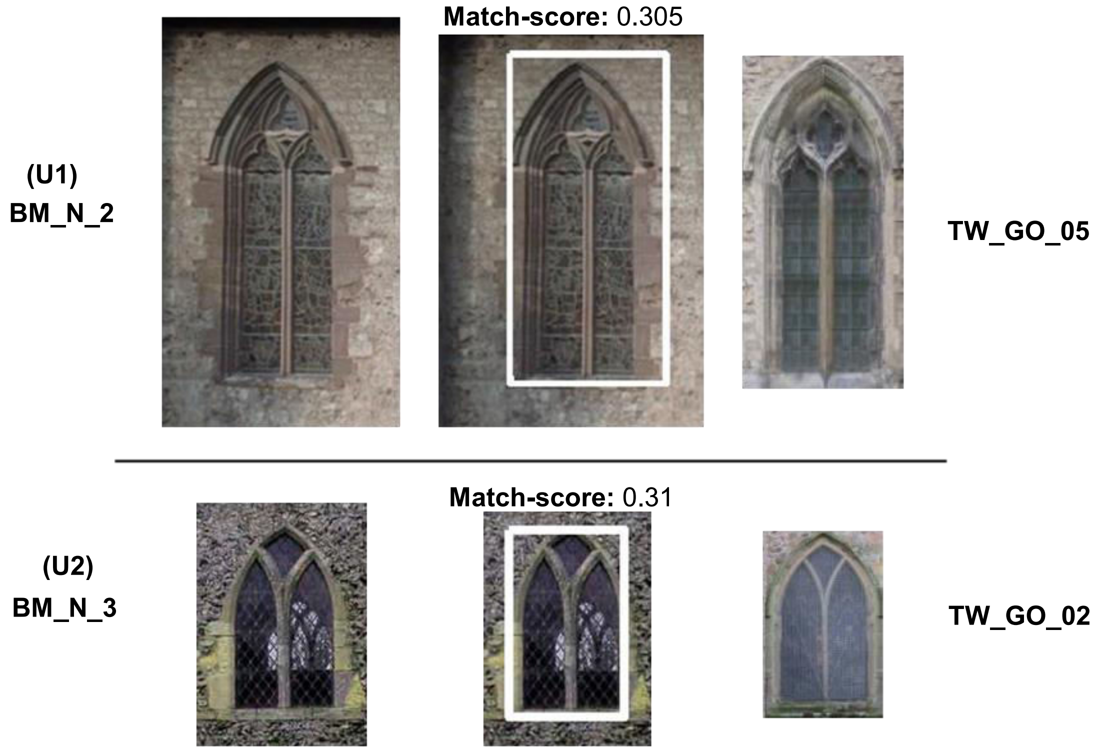


Figure 7.9: Examples of Unexpected Findings – Correct 'Granularity 3' Detection for Gothic Style
For each of (*U1*) and (*U2*) the left-hand image shows the unmarked-up texture map image; the middle image shows the same but marked up with a candidate match using a white bounding box; the right-hand image shows the template that resulted in the candidate match.

forms, that models for templates have not been built for them all – this could be achieved if one had unlimited time.

Lastly, despite the invariance to rotation of the HOG descriptor, the method used here showed some ability to deal with window instances on texture map images that were distorted to the extent that they did not appear square-on. See Figure 7.10. The gridding of the HOG descriptor, and the associated reduction in sensitivity to the exact placement of component parts within objects, might explain such behaviour, as might the use of a range of orientations for the bins.

Admittedly, all the training images and templates were corrected for perspective distortion. Furthermore, an assumption was made that the creator of the 3D building model carried out the texture mapping in such a way that textures appeared square-on.

As a reminder, visible shear was identified on only four texture mapped *gml:LinearRing*s (<1%). Remember also that pre-2D-3D transformation validation check (4) was conducted on all texture map images, filtering out those with shear above a *gml:LinearRing*-

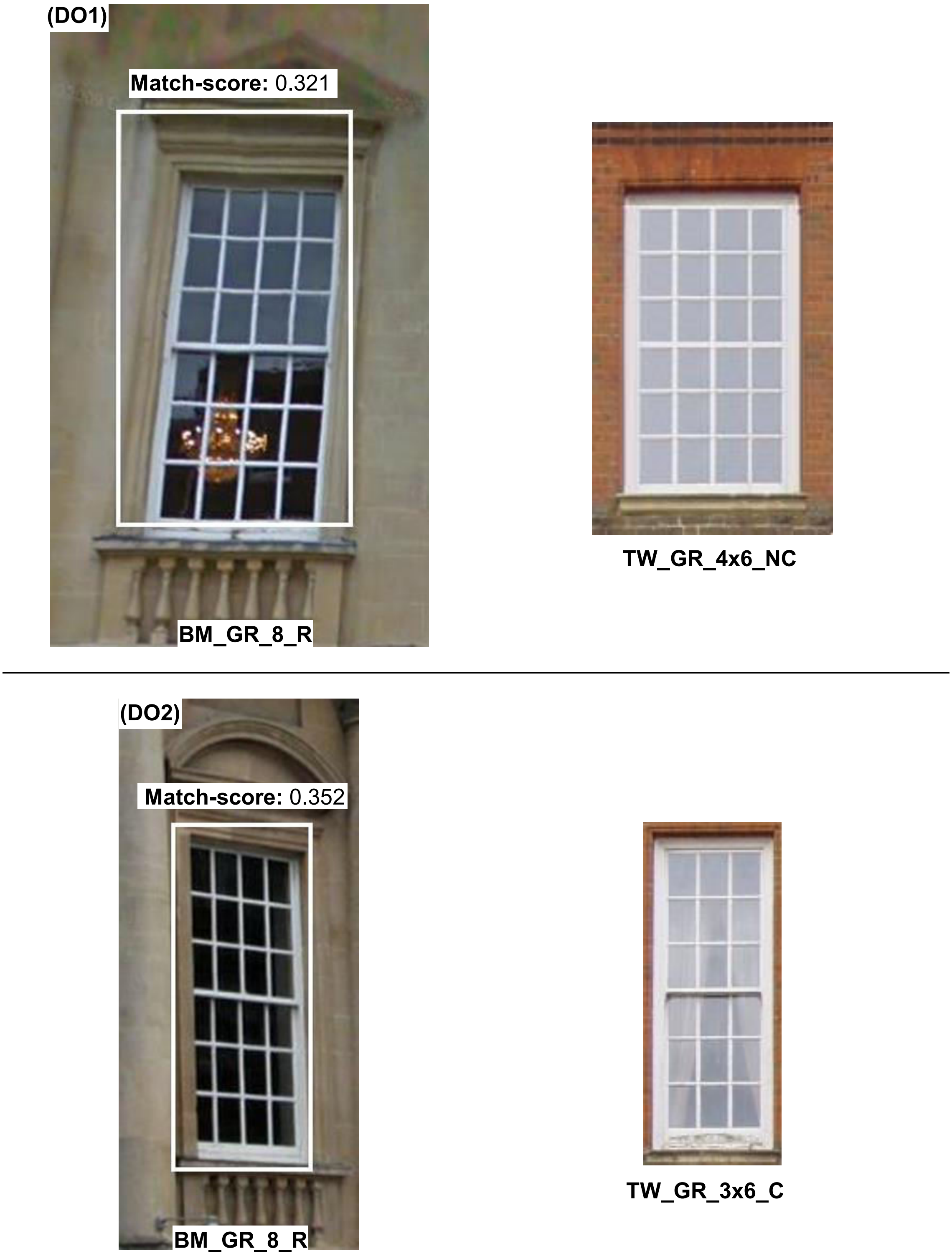


Figure 7.10: Capability of HOG-based Template Matching to Detect Distorted Objects
Each of (*DO1*) and (*DO2*) shows a portion of a texture map image (on the left) marked up with a white bounding box showing a candidate match, where the template that resulted in the match is shown on the right. '*DO1*' achieved a G3 match. '*DO2*' obtained a G1 match (and a G2 one, but only because template set (ii) was used). The 3D building model used here was rejected from the dataset owing to the use of obvious shear-mapping in a number of the *gml:LinearRing*s that used the texture map images shown above. Also note that this model was only used to illustrate distortion, and not in any other experiments.

specific threshold. However, the ability to still successfully detect objects that have undergone some distortion bodes well for the scalability of the HOG-based template matching method to unseen 3D building models.

As a caveat, if an object in a texture map image possesses non-shear distortion, such as perspective distortion, and a correctly centred candidate match is obtained on the image, then the 2D-3D transformation method will *incorrectly* identify orthogonality around the object in the texture map image. This is because *app:textureCoordinates* elements in CityGML can only apply texture mapping using affine, and not projective, transformation. In the experience of this author, perspective distortion is not uncommon in texture map images found on other 3D building models. To reiterate, CityGML *does* allow for projective transform, through a different mechanism to *app:textureCoordinates*, namely *TexCoordGen* (though this was not present for any of the 3D building models used here). One is essentially at the mercy of the person who created the 3D building model, specifically as regards whether they chose to correct perspective distortion using this approach. Note that the methods used in this work do not support the *TexCoordGen* approach.

Equally, if the object only possesses shear distortion, then the method used in this work will *not* apply any shear mapping contained in the CityGML. As mentioned earlier in this chapter, there is a very low incidence of visible shear mapping in the 3D building models used here.

7.2.15 Inferring Dominant Architectural Style, Architectural Development in a Building & Assessing Building Damage

By inference, the identification of the architectural style for the detected windows and doors also provided an indication of the potentially dominant architectural style, plus an indication of how the building has developed architecturally over time. Therefore, the correct distinction between the detected architectural style of the windows and doors in the Norman 3D building models, along with the calculation of the positions of those objects on the building, indicates the areas of the building that witnessed the incursion of the Gothic tradition. For example, the Gothic windows successfully identified on the texture map images in Figure 7.9, plus their calculated positions on the building, indicate that those regions of what are originally Norman buildings underwent modification to the Gothic style.

The methods in this thesis might also be used to detect which architectural components in a building were damaged, such as damage following an earthquake or vandalism. By using new imagery, i.e. that showing the damage, as texture map images on a 3D building model which already includes window and door geometry and corresponding semantic labels, the methods described here could be used to detect the damage, as follows. Firstly, the 'detection' of false negatives with window or door templates on parts of the model known to be windows or doors could indicate damage i.e. a window or door should have been detected but was not because it was damaged. Secondly, damaged-window or door templates could be used to detect true positives to indicate damage.

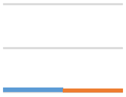
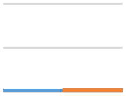
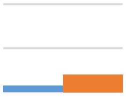
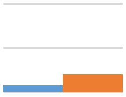
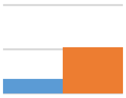
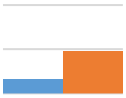
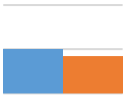
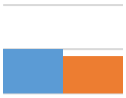
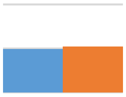
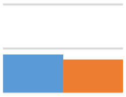
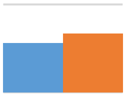
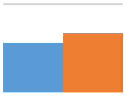
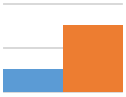
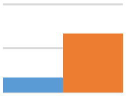
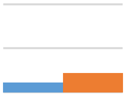
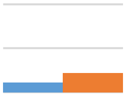
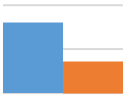
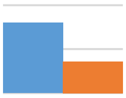
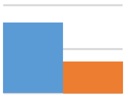
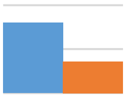
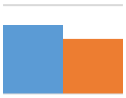
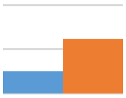
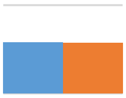
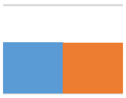
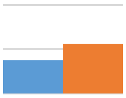
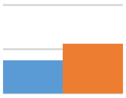
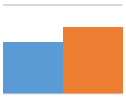
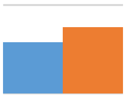
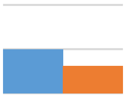
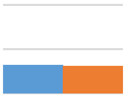
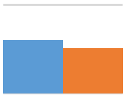
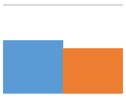
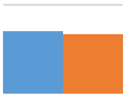
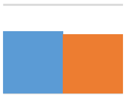
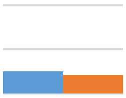
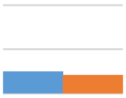
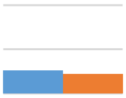
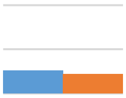
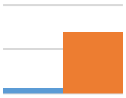
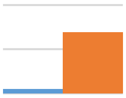
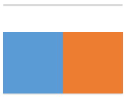
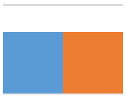
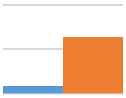
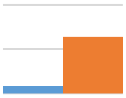
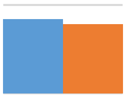
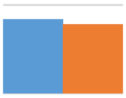
7.3 HOG-based Template Matching – SVM vs No SVM

7.3.1 Experimental Results

The results discussed to this point have been obtained without the use any machine learning classification. To recap, the HOG-based template matching pipeline produced in this work (see Chapter 5) was extended to include a machine-learning classification step, using a Support Vector Machine or SVM (see Chapter 6). The feature vector used with the SVM consisted of match-scores for the various templates. Experiments were conducted using the classification step, with the same 3D building models used in the experiments discussed above. The exception to this was BM_GR_8_R, which was only used to illustrate texture map distortion (see above). The results for use of the SVM, and a comparison with the results obtained *without* the use of the SVM, are provided below.

Figure 7.1 and Figure 7.2 compare the candidate matches resulting from HOG-based template matching without the SVM ('no SVM') and with the SVM, rendered onto the 3D building models. Figure 7.3 and Figure 7.4 show the equivalent for Georgian-Regency, and Figure 7.5 for Norman.

Table 7.7, Table 7.8 and Table 7.9 use bar charts to illustrate a comparison of the F-measure obtained with no SVM and with the SVM, at different class granularities, for the 3D building models of the C20, Georgian-Regency and Norman styles respectively. For each 3D building model, the results are split into template set (i) and template set (ii). Appendix G shows the equivalent information, but as values. Table 7.10 shows the equivalent information (as bar charts), but as means per style and as means for all 3D building models combined. Finally, Table 7.11 shows the delta values for using the SVM compared to the use of no SVM. Note that no results are given for BM_2_9 for template set (i) – see Section 7.3.3.

C20 style 3D Building Model	Template Set (i)			Template Set (ii)		
	G1	G2	(No G3)	G1	G2	(No G3)
BM_2_1						
BM_2_2						
BM_2_3						
BM_2_4						
BM_2_5						
BM_2_6						
BM_2_7						
BM_2_8						
BM_2_9	See Section 7.3.3					
BM_2_10						
BM_2_11						
BM_2_12						

KEY G Class Granularity (G1, G2, G3)

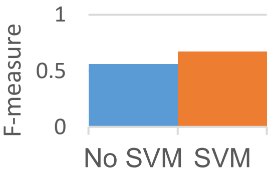


Table 7.7: SVM vs No SVM – Experimental Results – C20 Style 3D Building Models See the caption for Table 7.8.

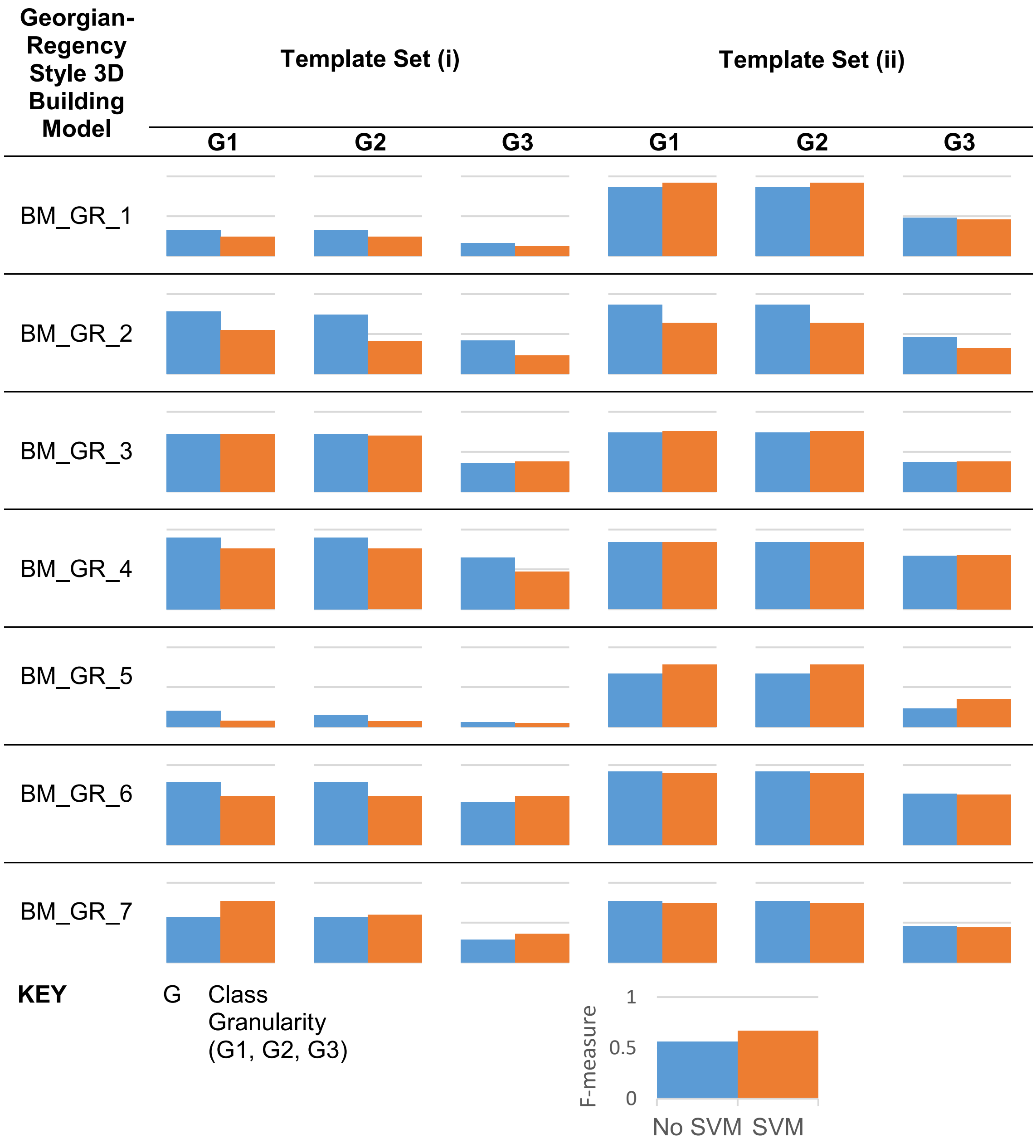
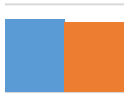
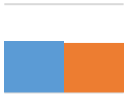
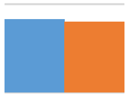
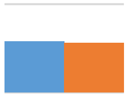
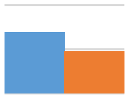
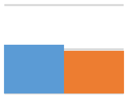
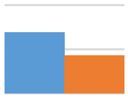
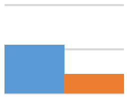
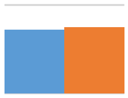
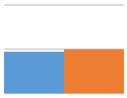
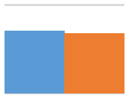
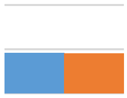
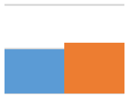
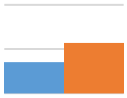
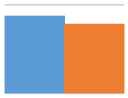
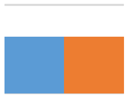
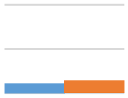
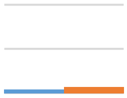
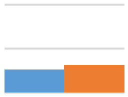
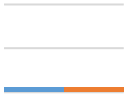


Table 7.8: SVM vs No SVM – Experimental Results – Georgian-Regency Style 3D Building Models
Values are provided for running the method with template set (i), and for running the method with template set (ii). Each of those sets of results provides values at each class granularity. G1 and G2 for C20 and Georgian-Regency are the same (only one style of template was used). There is no G3 for the C20 or Norman styles.

Norman Style 3D Building Model	Template Set (i)			Template Set (ii)		
	G1	G2	(No G3)	G1	G2	(No G3)
BM_N_1						
BM_N_2						
BM_N_3						
BM_N_4						
BM_N_5						

KEY	G	Class Granularity (G1, G2, G3)

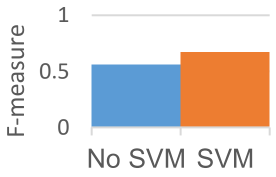
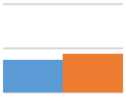
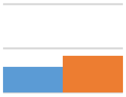
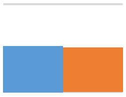
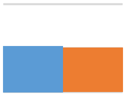
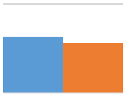
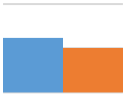
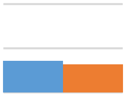
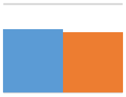
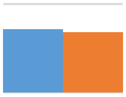
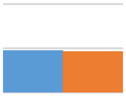
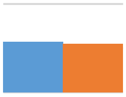
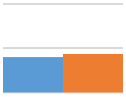
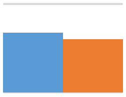
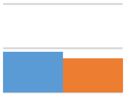
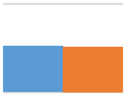
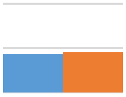
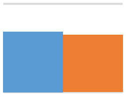
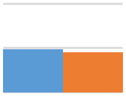


Table 7.9: SVM vs No SVM – Experimental Results – Norman Style 3D Building Models
See the caption for Table 7.8.

3D Building Model Style	**Template Set (i)**			**Template Set (ii)**		
	G1	**G2**	**G3**	**G1**	**G2**	**G3**
C20			**No G3**			**No G3**
Georgian-Regency						
Norman			**No G3**			**No G3**
All Styles						**No G3**

KEY G Class Granularity (G1, G2, G3)

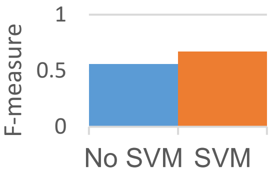


Table 7.10: SVM vs No SVM – Experimental Results – Mean by Style & Mean All Styles Combined
See the caption for Table 7.8.

(a)
F-measures

3D Building Model Style	Use SVM?	Template Set (i)			Template Set (ii)	
		G1	G2		G1	G2
C20	No SVM	0.37	0.29		**0.52**	**0.52**
	SVM	**0.44**	**0.41**		0.51	0.51
		G1	G2	G3	G1 & G2	G3
Georgian-Regency	No SVM	**0.63**	**0.62**	**0.35**	**0.86**	**0.47**
	SVM	0.55	0.51	0.32	0.84	0.47
		G1	G2		G1	G2
Norman	No SVM	**0.57**	0.40		**0.67**	**0.46**
	SVM	0.55	**0.44**		0.6	0.39
		G1	G2		G1	G2
All Styles	No SVM	**0.52**	0.43		**0.68**	**0.49**
	SVM	0.51	**0.45**		0.65	0.45

(b)
△F-measure vs No SVM (%)

3D Building Model Style	Use SVM?	Template Set (i)			Template Set (ii)	
		G1	G2		G1	G2
C20	No SVM					
	SVM	19%	43%		-4%	-4%
		G1	G2	G3	G1 & G2	G3
Georgian-Regency	No SVM					
	SVM	-12%	-18%	-10%	-2%	-2%
		G1	G2		G1	G2
Norman	No SVM					
	SVM	-4%	10%		-11%	-16%
		G1	G2	G3	G1	G2
All Styles	No SVM					
	SVM	-2%	4%		-5%	-7%

(c)
△F-measure vs No SVM (Abs.)

3D Building Model Style	Use SVM?	Template Set (i)			Template Set (ii)	
		G1	G2		G1	G2
C20	No SVM					
	SVM	0.07	0.13		-0.02	0.07
		G1	G2	G3	G1 & G2	G3
Georgian-Regency	No SVM					
	SVM	-0.08	-0.11	-0.04	-0.02	-0.01
		G1	G2		G1	G2
Norman	No SVM					
	SVM	-0.02	0.04		-0.07	-0.07
		G1	G2	G3	G1	G2
All Styles	No SVM					
	SVM	-0.01	0.02		-0.04	-0.03

KEY
Per style per granularity (G1, G2, G3) **bold** F-measure is highest

KEY
Green text: △ > 0 Black text: △ = 0 **Red** text: △ < 0

Table 7.11: HOG-based Template Matching – SVM versus No SVM – Experimental Results – Mean by Style & Mean All Styles Combined
Values are provided for running the method with template set (i), and with template set (ii). Each of those sets of results provides values at each class granularity, namely G1, G2 and G3. G1 and G2 for C20 and Georgian-Regency are the same (only one style of template was used). There is no G3 for the C20 or Norman styles (see Chapter 3). Sub-table (*a*) contains F-measure results (vertically) per style and per granularity. The highest F-measure for a per-style, per-granularity set is highlighted in **bold**. Sub-tables (*b*) and (*c*) show the delta (△) in F-measure versus the use of no SVM, in percent and as an absolute value respectively.

7.3.2 Results Summary

Please note that delta in this section refers to the difference between the F-measure obtained with the SVM, and that obtained without the SVM. The mean delta in F-measure achieved for all 3D building models, across both template sets and for all granularities, for the use of the SVM compared to the use of no SVM, was:

(a) -0.02 (all 3D building models) Interpretation: the SVM mean is 0.02 lower

The equivalent comparisons by style were:

(b) 0.04 (C20 3D building models) Interpretation: the SVM mean is 0.04 higher

(c) -0.05 (Georgian-Regency 3D building models)

(d) 0.03 versus 0.50 (Norman 3D building models)

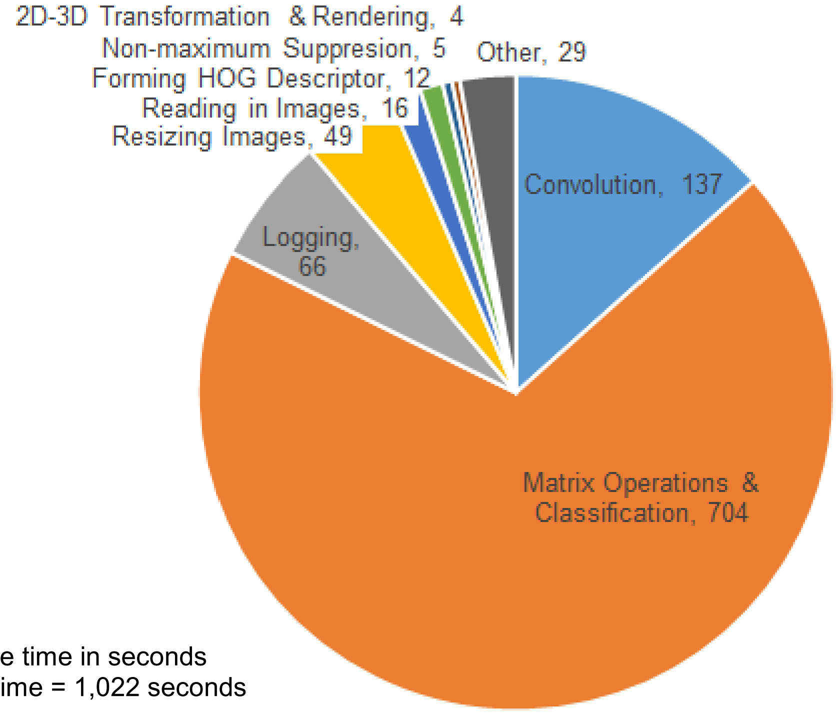
Looking in more detail at the results, the F-measures achieved with the SVM for template set (i) on Georgian-Regency style 3D building models were somewhat anomalous, in that they included the most-negative mean deltas (for G1 and G2) for any of the styles (-12% and -18%, -0.08 and -0.11). Nevertheless, the mean negative deltas for G1 and G2 for the Norman 3D building models, with template set (ii), are quite close to these values. However, a mean negative delta of -0.11 is not particularly high when one considers that each pair of values in each of (a), (b) and (c) are within 0.05 of one another (and noting that a perfect F-measure has a value of 1).

BM_2_11 achieved the highest positive absolute delta (0.64, 1,280%, template set (i), G2), while the most-negative absolute delta (-0.44, -55%, both template sets, G1 and G2) was obtained with BM_2_5. That negative delta was anomalous when one considers the mean delta per 3D building model, for all template sets and for all granularities, as follows. For the C20 style the mean in question was above zero for 8 out of the 12 3D building models and, aside from BM_2_5, the most-negative mean delta for a 3D building model was -0.1. While four out of seven Georgian-Regency 3D building models obtained a negative mean delta, aside from one mean of -0.28, the most-negative mean delta was -0.1. Lastly, for the Norman 3D building models two of five obtained a negative mean delta, where one was -0.22 and the other -0.03.

7.3.3 Runtime

When using the SVM in the pipeline, mean runtime for all 3D building models with template set (ii) increased by 11 minutes (94%). Appendix H contains detail of the split of runtimes by architectural style and by 3D building model. Particular processes took longer when using the SVM – compare Figure 7.6 (no SVM) and Figure 7.11.

The principle reason for the general increase in runtimes was two-fold. Firstly, the SVM approach required the construction and manipulation of a matrix, containing the candidate match data for all 8×8-pixel grid cells across the image and all templates. The contents of the matrix included: match-score, template class, and candidate match dimensions. Classification required that the matrix needed to contain the match-scores for *all* templates at *every* grid cell across a texture map image. In contrast, the no SVM approach thresholded match-scores, significantly reducing the size of the matrix before any manipulation was required.



NOTE
Values are time in seconds
Total runtime = 1,022 seconds

Figure 7.11: Split of Runtime by Process – SVM
The 3D building model used for the timings was **BM_N_4**, with template set (ii). Compare withFigure 7.6, which shows process timings for the same 3D building model when no SVM was used. Due to the nature of the coding approach it is not possible to split the 'Matrix Operations & Classification' process into two process timings. Note that total runtime is not consistent with that in Appendix H due to the additional overhead resulting from timing the processes. Also note that processes which are unchanged versus running with no SVM, such as the convolution step, have slightly differing durations due to small time fluctuations inherent to the application used to write the code.

Secondly, time was then required to conduct classification for each of those grid cells. As a result, the larger the texture map image and the more templates used, the larger the matrix and the longer the runtime. For example, using the largest template set, set (i), on the largest texture map image across all the 3D building models (one of those in BM_2_9), the texture map image would not process due to memory issues. See Appendix H for more background to this issue. Initial attempts were made to optimise the code to allow the texture map image to process. A decision was taken not to attempt any further code optimisation on the basis that mean parity of F-measure was at least achieved for the 23 other 3D building models (see below).

Clearly, the runtime for some of the 3D building models when using the SVM was high. One potential solution to both the memory issue and high runtime would be to add a step, at the start of the pipeline, which automatically reduces the size of texture map images above a certain size. Any candidate matches could then be scaled to the original size as a later step in the pipeline.

7.3.4 Discussion

Based on (a), (b), (c) and (d) in Section 7.3.2, the use of the SVM is broadly achieving parity of F-measure compared to the use of no SVM. There are some anomalies though, both positive and negative. It is likely that such anomalies are the result of two phenomena, described below.

(1) The first is the potential inconsistency between the spread of the window or door designs ground-truthed in the training image set, and the window or door designs found in some of the texture map images. Crucially, the inconsistency may not be visible to the naked eye. As such, windows that appear similar visually may not have similar HOG descriptors, for the reasons already discussed in this chapter. In theory, a greater number of training examples could reduce the anomalies, though note the challenges described in Chapter 3 regarding data collection.

(2) The second consideration is the potential for poor distribution within and between the match-scores for the positive- and negative-class training observations. Figure 7.12 shows the distribution of match-scores for the training observations used in this work. If present, poor distribution could have two possible effects, as follows.

Before discussing the potential effects of (2) it is worth reiterating that a ratio of 1:6 was used for the number of positive-class to negative-class training observations. The ratio was chosen to reflect the approximate split of ‘window or door’ to ‘not-window or not-door' regions on building façades. Early experiments with a 1:1 ratio resulted in no improvement in F-measure compared to those using the 1:6 ratio.

The first potential effect (E1) of (2) would be the inability to pick out poor (but positive class) window or door instances in testing. Essentially, E1 might occur if the match-scores used for the positive-class training observations were generally at the higher end of the possible range of match-scores for positive instances of the type of window or door object in question. To recap, the use of validation checks (a) and (b) during the selection of training ground truths promoted strong candidate matches as an input to training (see Chapter 6). However, the use of those two validation checks could also truncate the distribution of the match-scores for the object instance for the positive-class object.

As a further reminder, the rationale behind the two validation checks was to differentiate between the following two types of object. (Objs1) The first type of object corresponded to positive instances of window or door objects, on the training images, which *would* result in a correct candidate match at the end of the pipeline. (Objs2) The second type of object corresponded to objects which were windows or doors according to a human observer, but would *not* result in a correct candidate match at the end of the pipeline – their match-score was too low. Based on the approach used here, the match-scores for (Objs2) were more likely to be similar to the match-scores for negative-class ground truth locations (more on which later in this section).

At this point it is worth noting that there is a discord between the training and testing criteria used in the method. It is otherwise commonly the case that an application of machine learning uses the same criterion for training and testing. The training criterion in this work was, strictly, to minimise label assignment error, while the testing criterion was different, namely F-measure. Nevertheless, an approach loosely akin to the use of F-measure to optimise training input *was* used. That approach comprised the initial offline selection of training ground truth instances using correct candidate matches from running HOG-based template matching on the training images. Perhaps the tuning of the training input using F-measure would have improved the distribution of the match-scores, although note the comments below regarding the possible problems with the normalisation scheme employed.

The second potential effect (E2) of (2) would be the overlap in the distribution of match-scores for object instances of the negative class ('null' class) with the match-scores for object instances of the positive class. As Figure 7.12 illustrates, there is a small amount of such overlap in the training observations. With positive-class match-scores in the range of 0.24-0.44 with a mean of 0.33, versus negative-class match-scores of 0.13-0.3 with a mean of 0.21, the trend is for satisfactory separation of match-scores (noting that the overall range is only 0.13-0.4). Perhaps those training observations for the negative class that were anomalously distributed (see Figure 7.12) could be removed from the training observations.

This presents something of a quandary. The proverbial 'rob Peter to pay Paul' dilemma occurs: choose a larger number of poor positive-class instances, but suffer from less separation of positive-class and negative-class match-scores and potentially worse classification as a result. Perhaps the solution lies in a further review of the normalisation scheme used. With better normalised match-scores the separation would be greater, allowing poor positive class examples to be chosen. In addition, a refinement of the features used may also bring improved performance (see later in this section).

More crucially perhaps, poor separation of the match-scores for the positive- and negative-class test observations could also lead to poor classification. There are two possible causes for this. (PC1) Firstly, the training data might falsely represent the spread of match-scores resulting from testing (see above). (PC2) Secondly, the normalisation scheme might be flawed, despite the undertakings described in Chapter 5. (PC2) would be more fundamental than (PC1) because it would affect training too. Figure 7.13 illustrates the distribution of the positive-class and negative-class testing observations. Compare the distribution of the test data shown in Figure 7.13 with the equivalent distribution of the training observations in Figure 7.12. Table 7.12 shows statistical measures for each of the training and testing observations distributions, split by positive class and negative class. These results are discussed below.

The statistical measures for the positive class in Table 7.12 demonstrate that the training and testing observations for the positive class have similar distributions. The same measures for the negative class indicate that the negative-class testing observations have a wider range, a lower mean and higher standard deviation than those for the negative-class training observations. Consequently, there is more overlap of the testing observations for the positive-class and negative-class, as shown in Figure 7.13, when compared to Figure 7.12. This suggests that using a distribution similar to that of the negative-class testing observations for the training observations *could* improve performance. Doing so has the potential to *increase* the incidence of (E1) and (E2) though, in addition to presenting further opportunity for manifestation of the abovementioned quandary. At this juncture it is worth restating the potential utility of a further review of the match-score normalisation scheme used – a better distribution of match-scores might resolve the issues of the overlapping distributions.

That aside, noting the differing distributions, one potential future avenue might be the use of techniques from the emerging field of 'transfer learning'. Essentially, transfer learning allows the 'domains, tasks, and distributions used in training and testing to be different'; it also allows the borrowing of (i) labelled data and extracted knowledge from one or more related domains to help improve classification in (ii) the domain of interest (Pan, 2015; Yamada *et al.*, 2018, p. 1). As such, transfer learning could also potentially overcome the dearth of data available for the task of classifying architectural style of windows and doors (see Chapter 3), using pre-existing trained classifiers for the 'window' or 'door' class, followed by a second stage to detect the architectural style of those objects.

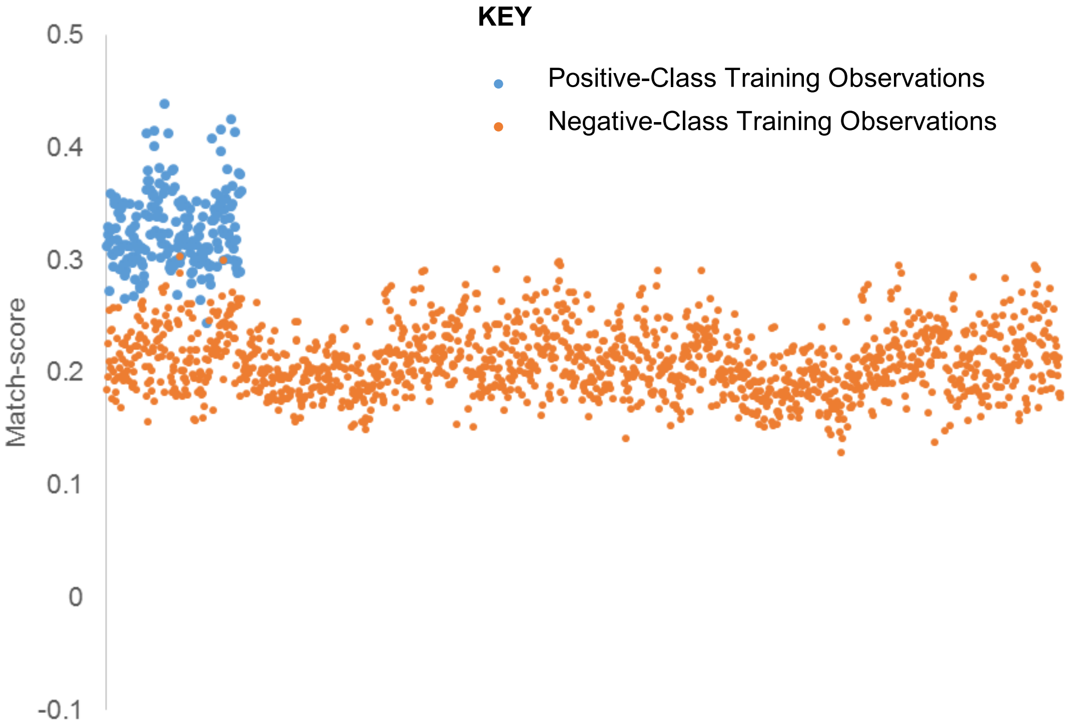


Figure 7.12: Distribution of Match-Scores for Positive-Class & Negative-Class Training Observations

The training observations shown cover all four styles of template and therefore represent all training observations used in this work. For each of the positive-class and negative-class training observation sets, observations are distributed with equal spacing horizontally. This is purely for the purposes of illustration and does not imply any distribution based on value. Compare this figure with the equivalent results for the testing observations in Figure 7.13.

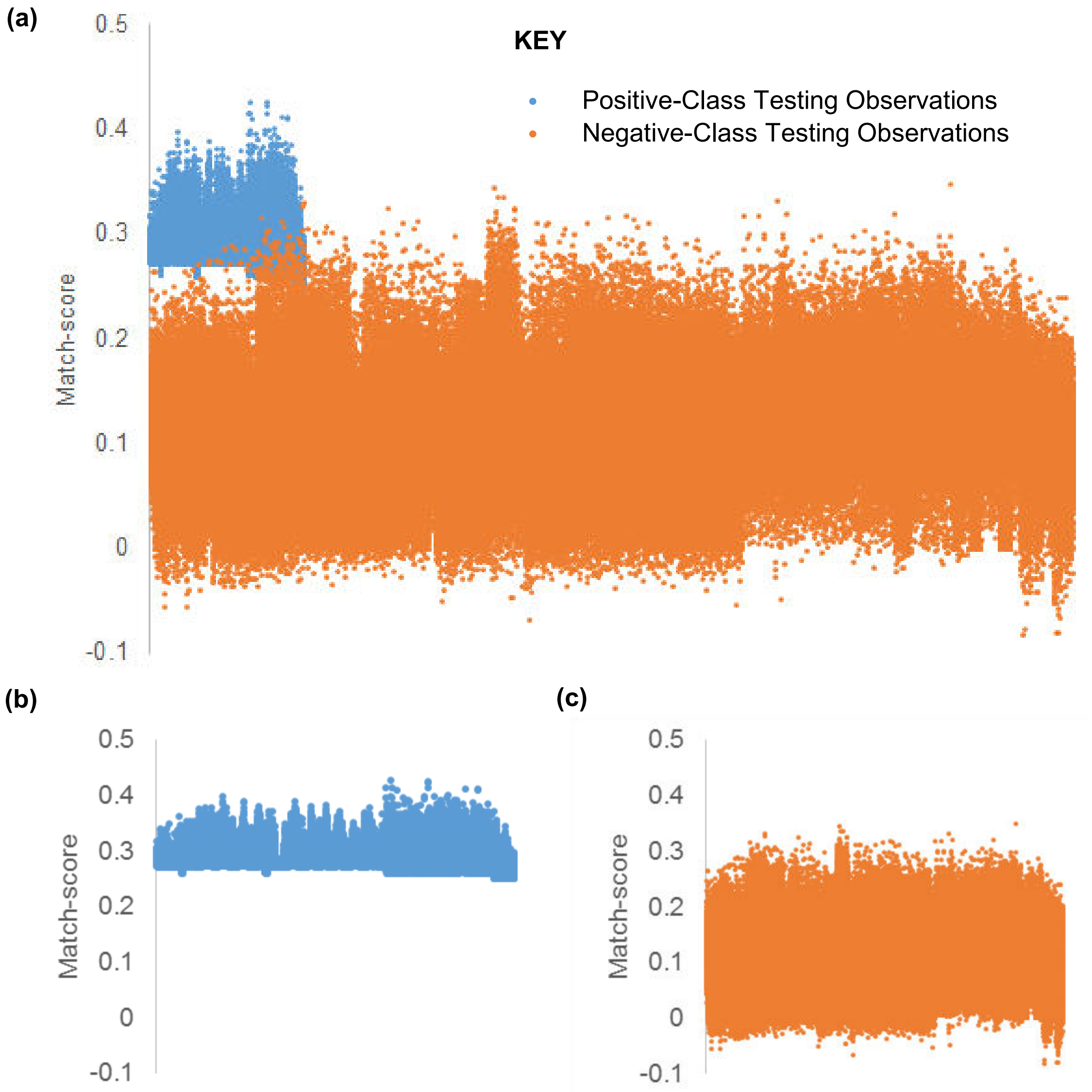


Figure 7.13: Distribution of Match-Scores for Positive-Class & Negative-Class Testing Observations

See the caption for Figure 7.12, but substitute the word 'training' for the word 'testing'. Note, the results in this figure were achieved with template set (ii). (*a*) Positive-class and negative-class testing observations shown together. (*b*) Positive-class testing observations only, to allow any overlap between positive-class and negative-class data points in (a) to be resolved. (*c*) The equivalent for (b), but for negative-class testing observations. Note that the horizontal scales for (b) and (c) are different, but only for the purposes of illustration. The results shown above are post-classification but pre-non-maximum suppression. For the purposes of illustration, and in order to allow visual comparison with Figure 7.12, a sub-set of the negative-class testing observations is shown. This uses the 1:6 ratio of positive-class to negative-class observations used in training to conduct the sampling of the negative-class testing observations above. The sample members were determined using a pseudo-random number generator.

Measure	Positive-class Observations		Negative-class Observations	
	Training	Testing	Training	Testing
Minimum	0.24	0.25	0.13	-0.08
Maximum	0.44	0.43	0.30	0.35
Range	0.19	0.18	0.17	0.43
Mean	0.33	0.29	0.21	0.11
Standard Deviation	0.03	0.02	0.03	0.06

Table 7.12: Statistical Measures for Distribution of Positive- & Negative-Class Training & Testing Observations
The data represented here is from Figure 7.12 and Figure 7.13.

Returning to the question of obtaining parity with the no SVM result, such parity *could* be lauded. After all, it promotes the scalability of the method to new styles and unseen 3D building models (runtime issues aside). However, methods such as SVM tend to be used to *improve* results, compared to methods that do not use machine learning. To recap, steps were taken during this work to optimise the classification model, including the use of data augmentation with Gaussian-blurred training images, and feature selection through shrinkage. In addition to reviewing the choice of training data and the normalisation scheme, as outlined above, performance might be improved through further feature selection and adding new features. Additional features might include those suggested in Chapter 6.

7.4 Comparison with Existing 3D Building Model Window & Door Geometry

As a reminder, only a small proportion (5%) of the window or door instances on the texture map images for the 3D building models had existing window or door geometry. In this section such geometry is termed ‘(a)’, while the world-space geometry for candidate matches is termed ‘(b)’. As Chapter 3 noted, models containing (a) were still processed in the main body of this work. Here, an analysis of the extent of the superposition of corresponding window and door instances in (a) and (b) was conducted using the Tape Measure Tool in *SketchUp*. (a) and (b) for two 3D building models are shown in Figure 7.14. The analysis confirmed that, shear texture-mapping issues aside, the positions of each object in (a) and (b) are within the ±0.5m LOD3 accuracy recommendation.

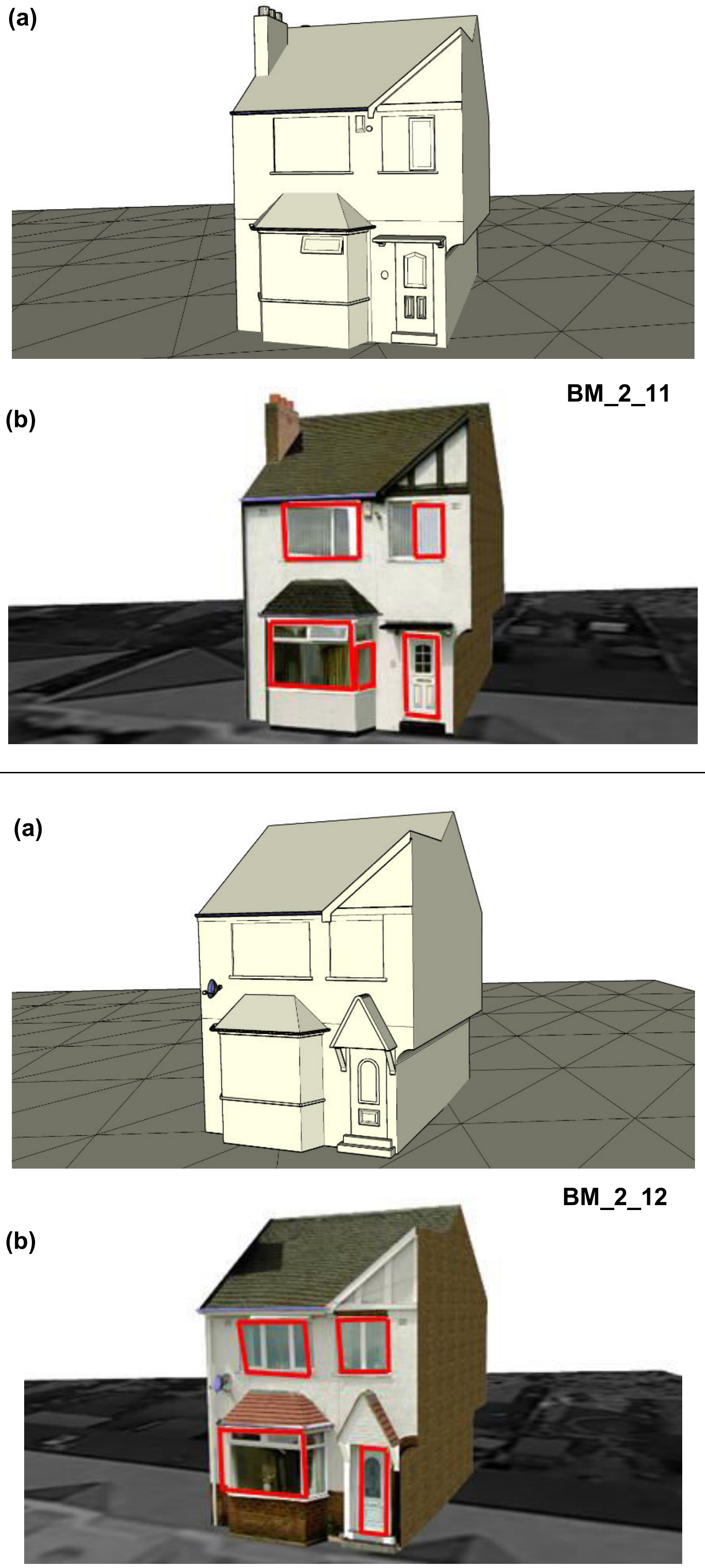


Figure 7.14: Comparison with Existing 3D Building Model Window & Door Geometry
(*a*) 3D building model with textures removed, showing pre-existing window and door geometry. (*b*) The 3D building model marked up with successful candidate matches as **red** boundaries. Note that shear has been applied during texture mapping, and that the viewer has introduced perspective distortion. Also note that each building is approximately 5m wide.

7.5 Comparison with State-of-the-Art & Reference Data

7.5.1 Comparing State-of-the-Art Window & Door Detection

To reiterate, no existing window or door detection method has been found to have a publicly available implementation. Similarly, no existing implementation to detect windows and doors on texture map images within 3D building models is available. Façade image databases *are* available such as (1) *ECP Façades Database* (Teboul, 2010; Teboul *et al.*, 2010), (2) *eTRIMS Image Database* (Korč and Förstner, 2009) and (3) *CMP Façade Database* (Tyleček, 2013). However, the supplied ground truths for (1) and (2) are recorded for every pixel, and not as rectangular boundaries as used here. The use of rectangular boundaries is necessitated by the decision to use the HOG-based template matching approach in this work. In addition, all three of those datasets include a variety of images which include occlusions, especially balconies in front of windows. As a result, the datasets only tend to mark up the non-occluded portions of window instances as a window class, something which the method in this work does not consider. Note that the 3D building models used do *not* contain such occlusions. (3) marks up ground truths as rectangles and also includes the option for overlapping ground truths. Compared to (1) and (2), which have become the standard for façade segmentation, its use in the field is significantly lower.

So, while the methods used here cannot be quantitatively compared with state-of-the-art approaches, be it via use of available implementations or the use of popular reference datasets, a qualitative comparison of the results and challenges is conducted below.

As a reminder, Liu *et al.* (2017) is the state of the art for window and door detection on façade images, as conducted on datasets (1) and (2) and using a grammar and deep learning approach. As Gadde *et al.* (2016) observe, grammar-based segmentation approach such as that of Liu *et al* are less suitable for 'façades with little regularity, e.g., with few windows, highly uneven layouts and strong architectural inconsistencies'. While Liu *et al* do supplement their segmentation with bounding box detection, the method is still inherently reliant upon symmetry. Due to the potential for inconsistencies between a template and an image, or between an architectural grammar used in façade detection and real-world architecture, identifying less symmetrical, less standardised architectural objects presents challenges. Indeed, as Georgopoulos and Stathopoulou (2017) note in their review of state-of-the-art detection methods within the cultural heritage domain, objects such as architectural components can present particular detection challenges due to the potential complexity and variability of their component parts.

Therefore, due to the lack of a publicly available implementation and a conflicting ground-truthing standard (pixel versus bounding-box based) it is not possible to say whether the methods in this work would better those of Liu *et al*, on the data from this work. Likewise, the methods from this work cannot be used to compare the results of Liu *et al* on datasets (1) and (2). Nonetheless, in order to gain insights to the potential for the methods in this work to scale to unseen images a trial using (1) was conducted, as described below.

7.5.2 Reference Data Trial – ECP Façades Database

Dataset (1), contains rectified façade images, as opposed to the unrectified images in (2). (1) contains buildings which are of two Classical French styles: *Restauration*[2], which lasted from 1814-1830, and Hausmannian, which was present from 1853-1870 (Hanser, 2006, p. xiv; Teboul, 2010; Teboul *et al.*, 2010; Watkin, 2005, p. 452). Note that the British Georgian-Regency style prevailed from 1715-1830.

The HOG-based template matching (no SVM) method described in this thesis was run on (1) using the Georgian-Regency template set (‘TS1’) and the C20 style template set (‘TS2’). While the styles in (1) are Classical, none of the template in (TS1) were of the same design as window or door instances on the façade images. Hence, the highly varied (TS2) was also used.

Initial experiments did not include any prior ground-truthing, noting once again that pixel-based ground truths are supplied with the dataset. Example results are shown in Figure 7.15. The figure also illustrates the classes used in the ground truths provided with the dataset. Moreover, it demonstrates examples of the aforementioned occlusion of windows by balconies.

Following a visual inspection of all 104 images marked up with candidate matches, it was determined that only one correct match was obtained with (TS1) and (TS2) combined. The *PASCAL VOC* area-overlap criterion was applied manually. No count of the number of window and door instances is provided with the dataset. However, based on the visual inspection, the mean number of instances per image is around 15, indicating that there are around 1,500 instances in the dataset.

Due to the poor result, all match-score thresholds were reduced by 70% and the experiments were re-run. This resulted in 48 true positives. This would still only be a ~3%

[2] ‘Restoration’ in English.

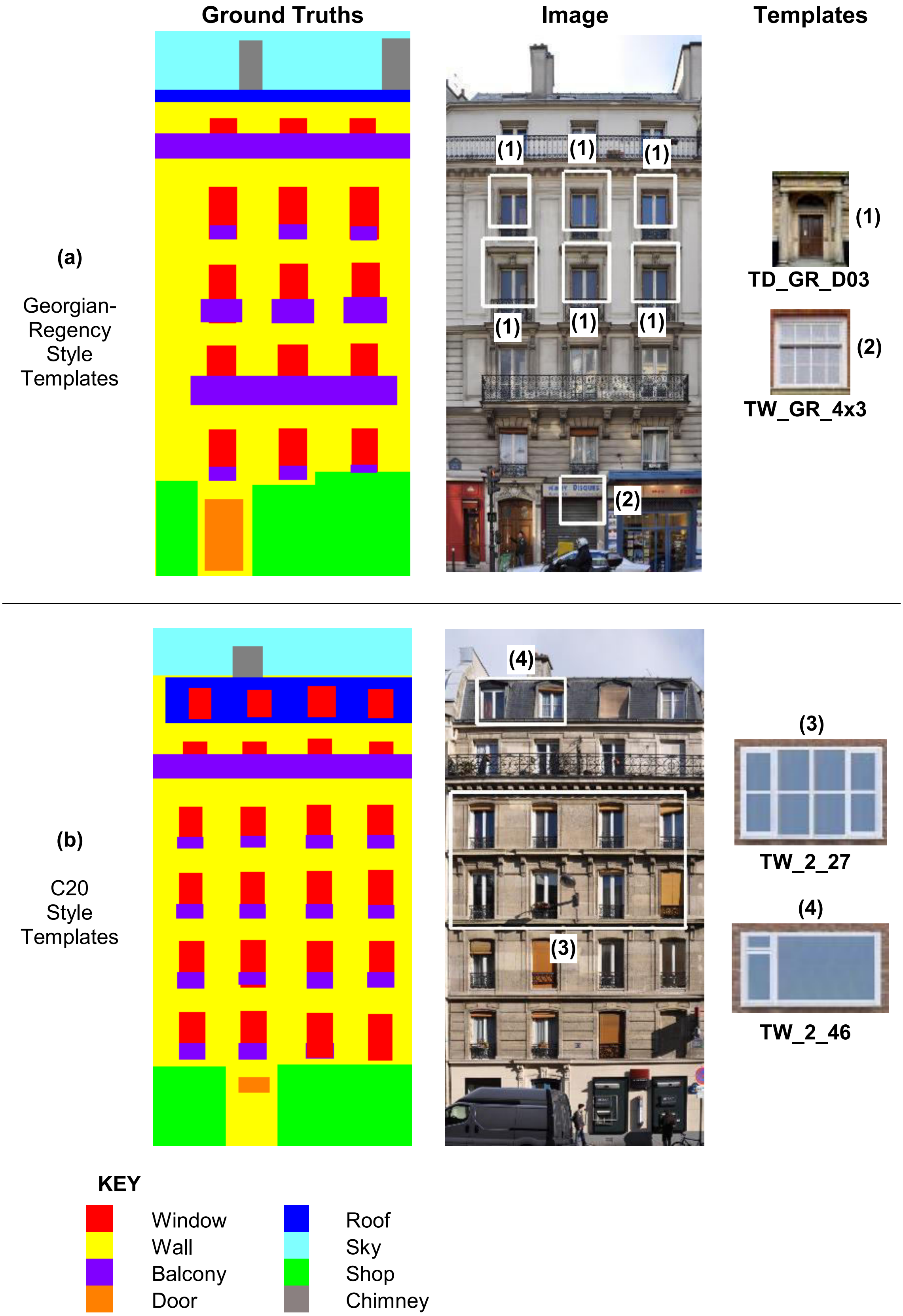


Figure 7.15: Candidate Matches on Selection of Images from *ECP Façades Database* Each of (*a*) and (*b*) show a façade image from the *ECP Façades Database*, marked up with candidate matches as white bounding boxes. The template that resulted in the candidate match is numbered and shown on the right. Note that for (a) heuristic 3 would likely filter out the candidate match if the image were part of the 3D building model. Ground truth and image sources: Teboul (2010).

detection rate though. The number of false positives also increased. As a consequence of the poor results obtained with (1), no further experiments were conducted with that dataset.

Not achieving any successful matches with (1) might seem wholly disappointing. The upside is that, by using template sets that were the incorrect architectural style and did not contain any windows or doors of the designs in the images, the results for architectural style detection are actually as expected. Essentially, the methods used in this work are more successful at detecting specific architectural styles of window or door than simply detecting windows or doors. By way of example, the no SVM method was run with Georgian-Regency 3D building models with the Gothic and Norman template set ('BT1'), and with Norman 3D building models with the Georgian-Regency template set ('BT2'). The mean F-measure for (BT1) was 0.1 for G1, and zero for (BT2) at the same granularity, thus further supporting the hypothesis that the method is a better architectural style window and door detector than a general window and door detector.

However, note in Figure 7.15(a) that, even with the threshold at 100%, the correct architectural tradition is at least being detected (Classical), on the basis that a ‘Georgian-Regency Style Door’ template resulted in the candidate matches in question. Perhaps then, with the right templates, good results could be achieved on dataset (1), the *ECP Façades Database*.

7.6 Summary

This chapter has detailed and discussed the experimental results. The focus has been on determining whether the heuristics improved the results, and whether the SVM matched or bettered the results without the SVM. For the former, the answer is yes. For the latter the answer is that parity has been achieved, but an improvement has not been obtained. Also, while runtime when using the SVM increased, a potential method for improving that situation has been identified.

The methods have shown prowess, particularly in the identification of Georgian-Regency windows, on Georgian-Regency 3D building models and, to a lesser extent, on Norman 3D building models. The challenges presented with respect to object detection can be summarised as taking the form of discrepancies between the HOG descriptor for the template and the texture map image instance. This can manifest itself as either a loss or a misrepresentation of design detail in the HOG descriptor, and can lead to false positives or false negatives. The problem is exacerbated by high levels of variability between different instances of the same class, by highly intricate designs and by poor quality imagery.

Lastly, the methods have been trialled on a reference dataset, the results of which indicate that the approach used in this thesis is better at detecting windows and doors of a particular architectural style than detecting general windows and doors.

In the next chapter, the work is concluded, potential applications of the work are highlighted and a summary of further work is provided

Chapter 8

Conclusion

8.1 Introduction

In this thesis, methods for the automated semantic and geometric enrichment of existing 3D building models with window or door objects, including a classification of their architectural style, were described. In order to achieve these goals HOG-based template matching (see: Xiao, 2013; Zhang *et al.*, 2013b) was employed, and a pipeline added which used multiple templates and thresholding (see: Slade *et al.*, 2015; Slade *et al.*, 2017) or machine learning. The application of the HOG-based template matching approach to 3D building models is novel. Parity of performance was achieved between the thresholding and machine learning approaches, thus promoting the scalability of these methods to unseen 3D building models. The architectural style of windows and doors can provide insights into the development of the building and its dominant architectural style.

A common restriction of the existing methods, including state-of-the-art window and door detection methods, is that they require the buildings in the input images to be of a style typified by symmetry, repeating intervals between architectural objects and standardisation of design, such as Classical styles. Examples of such methods include: Cohen *et al.* (2014); Liu *et al.* (2017); Morago *et al.* (2016); Müller *et al.* (2007); Riemenschneider *et al.* (2012); (Wolff *et al.*, 2016); Zhang *et al.* (2013a); Zhang *et al.* (2015).

While methods do exist that are not reliant upon such symmetry, regular intervals or design standardisation, invariably they do *not* detect architectural style. See, for example, the approaches of: Horne *et al.* (2016); Kim *et al.* (2016a); Martinović *et al.* (2015);

Teboul *et al.* (2013); Wenzel and Förstner (2016); Yu and Wang (2016). Methods that detect architectural style tend to do so for entire buildings, not individual windows and doors, and do so only on façade images, not on 3D building models. In this regard, note the approaches of: Chu and Tsai (2012); Ippolito and Attenni (2015); Obeso *et al.* (2016a); Strobbe *et al.* (2016); Zhang *et al.* (2010). Because older buildings tend to be an amalgam of architectural styles, such methods are therefore limited in scope.

Very few methods exist that attempt to determine architectural style for existing 3D building models. One is Henn *et al.* (2012) but their approach is restricted to a single Classical style. Lee *et al.* (2015) and Mathias *et al.* (2016) do detect the architectural style of windows and doors but *only* for Classical styles, and on façade images not 3D building models (Llamas *et al.*, 2017).

Conversely, the methods in this thesis did not rely on symmetry, nor require regular intervals between objects or design standardisation. In addition, the methods were employed on a range of pre-Classical, Classical and more modern styles, including buildings that are an amalgam of the highly varied Norman and Gothic styles. Such buildings and styles tend to be far less standardised in their design than Classical styles.

In the absence of suitable reference data, a challenging data gathering exercise was undertaken, and data was pre-processed as detailed in Chapter 3.

The fully automated pipeline enabled the processing and evaluation of 3D building models (as described in Chapters 5 and 6). A number of trials were also conducted, including image matching, standard template matching and template clustering (see Chapter 4 and Appendix D). A detailed evaluation of the output from the pipeline was conducted (see Chapter 7). The results indicate the potential for the method to identify an even finer granularity of architectural description than simply the style of the object, provided that the input images are of sufficient quality and suitable templates are used.

8.2 Thesis Achievements

The objectives of this work were as follows:

(1) Geometrically and semantically enrich existing 3D building models with windows and doors.

(2) Detect the architectural style of the windows and doors, including for older buildings i.e. those buildings more likely to be an amalgam of styles.

(3) Detect the windows and doors irrespective of the presence of symmetry or regularly spaced objects on the building façades.

(4) Make the method scalable to unseen 3D building models.

The research question was as follows:

> *Is it possible to automatically enrich 3D building models of cultural heritage buildings by extracting window and door information – both geometry and specific architectural style?*

Objective (1) was met through the creation of a pipeline which: incorporated HOG-based template matching to detect windows and doors on the texture maps of existing 3D building models which lack pre-existing window and door geometry; and transformed the corners of the detected windows and doors from 2D pixel locations into the 3D real-world coordinate reference system of the model. Poor matches were filtered out using heuristics, including those which used real-world units and position in the 3D building model. The normalisation of the match-scores output from HOG-based template matching allowed for multiple templates to be used. 24 3D building models were used, comprising a total of 17,433 surfaces and 956 window and door instances on the texture map images. A mean F-measure of 0.68 was achieved for window and door detection i.e. at class granularity 1 ('has the object been correctly identified as a window or a door'), using templates of styles found on the buildings and the non-SVM approach. The result with the SVM was 0.65. For the Georgian-Regency style, the mean F-measure was 0.86.

In successfully using two geometry-based heuristics (those which use real-world units and position) this work has opened-up the possibility of applying additional geometry-based heuristics to the overall pipeline, with a view to improving the results from computer vision still further (see Section 8.7 for details of future work). The heuristics resulted in a mean F-measure improvement of 73%, and in one case a 30-fold improvement.

In addition, in the context of this work, it has been demonstrated that SIFT-based image matching and the standard template matching approach (Kroon, 2011) are not successful for window and door architectural style detection.

Objective (2) has been met through the use of multiple templates (148) taken from four architectural styles, thus defining a lower level of class granularity for window and door detections. Those four architectural styles were: Norman, Gothic, Georgian-Regency and 20th century styles. This resulted in F-measures at granularity 2 ('is the window or door of the right class') of up to 0.64 for Norman buildings which are an amalgam of Norman and Gothic window and door objects. In addition, provided enough suitable data is available then the results in this thesis also indicate that an even greater level of granularity of architectural style description is possible using the approach described herein.

This work has also created the potential, subject to agreement to use those images which are copyrighted (see Appendix A), for a test dataset that improves on the existing test datasets (for the reasons outlined in Section 8.4). Such a dataset would be suitable not only for generic window and door detection methods, but, crucially, for those methods which identify the architectural style of windows and doors and do so on existing 3D building models. To reiterate, no such publicly available dataset currently exists.

Objective (3) as been met through the use of a template matching approach, and an absence of a grammar-based or other spatial-constraint step, where the presence of such spatial constraints typifies many of the methods in the field. A lack of such spatial constraints in the approach here meant that the distribution of window and door objects on façades did not need to adhere to a symmetry or regularity. As such, windows and doors on older buildings, i.e. those which tend to be an amalgam of styles and thus have windows and doors which are distributed on the façades less symmetrically or more irregularly, such as Norman and Gothic objects, have been detected. Existing methods tend to use buildings which are either of Classical styles or are modern office or apartment blocks i.e. those typified by symmetrical or regular distribution of objects on façades. Furthermore, the styles used in this work include those which contain high levels of variability and low levels of standardisation of window and door design, unlike existing methods.

Objective (4) has been met by using an SVM, which removed an empirical step to derive match-score thresholds based on architectural style. The results, when using the SVM, provided parity with the results achieved using the empirically derived thresholds. As such, provided sufficient suitable data are available the SVM-based method has the potential to scale to unseen 3D building models and different architectural styles. Objective (4) is of particular importance to the Ordnance Survey, noting, once again, that for the methods described in this thesis to be of use they would need to be rolled-out across Great Britain, supplementing their existing work on automated detection of roof-type in imagery (Orlowski, 2017).

Separately, the work here has also formed best practice for (a) the capture of imagery which will be used by modellers when applying texture maps to a 3D building model, and (b) the modelling of CityGML with a view to the insertion of detected window and door objects. (a) and (b) are detailed in Section 8.6.

In meeting all the above objectives, the research question has also been answered: yes, it is it possible to automatically enrich 3D building models of cultural heritage buildings by extracting window and door information – both geometry and specific architectural style.

In summary, the work described in this thesis has made the following contributions to the field:

- The novel and effective application of HOG-based template matching to 3D building model window and door object detection, with multiple templates (Slade *et al.*, 2015; Slade *et al.*, 2017)
- The successful use of heuristics to filter candidate matches, including the use of geometry-based heuristics based on real-world object position, demonstrating the potential to add further geometry-based heuristics in future
- Determination that, in the context of this work, HOG-based template matching should be used over SIFT-based image matching or standard template matching
- The detection of those objects irrespective of symmetry, the interval between them or the standardisation of their design (Slade *et al.*, 2015; Slade *et al.*, 2017)
- Detection of the architectural style of those window and door objects (not just of the entire building) including:
    - Architectural style detection for the window or door objects for 20th century and Classical styles, in addition to the far more challenging Norman and Gothic styles
- Parity between the results obtained with (1) empirically determined thresholds and (2) machine learning which, in effect, learnt the thresholds
- Semantic and geometric enrichment for existing 3D building models with objects of the classes of window and door and their architectural style (Slade *et al.*, 2017)

- Enabled the potential for Ordnance Survey to extend their existing automated roof-type detection to window and door detection, potentially across Great Britain
- The potential, subject to copyright issue resolution, to release a dataset for use detecting both generic windows and doors and the architectural style of the windows and doors on existing 3D building models – such a dataset does not currently exist
- The potential to infer the possible dominant architectural style for a building, based on the spread of architectural styles for the newly identified windows and doors
- The potential to readily infer the possible architectural history of a building, how it developed over time, from the architectural styles of the identified window or door objects and their locations on the building
- Formed best practice for the process of image-capture for texture mapping onto 3D building models
- Formed best practice for the process of creating CityGML for 3D building models, including the manner in which to insert new CityGML for detected windows and doors

8.3 Thesis Achievements in Context of Cultural Heritage

By enabling the creation of 3D building models enriched to LOD3, with window and doors labelled by architectural style this work makes possible the analysis of semantic and geometric content within such models, for the purposes of cultural heritage. Arguably, the abovementioned results for the correct identification of architectural style were strongest and most consistent for buildings whose styles were typified by standardised designs of window and door, i.e. Georgian-Regency (F-measure 0.86, versus 0.68 for all 3D building models). That said, the results have also shown that, with sufficient suitable data, a fine granularity of architectural style detection for styles not typified by such standardised designs is possible. On which note, see Section 8.6 for future work which would include a focus on bringing consistently improved performance for such styles.

To put the above results into a cultural heritage context, note once again that knowledge of the architectural style of windows and doors, within a 3D building model, could potentially drive a number of applications. These include: augmented reality such as in an educational 'guess the style' mobile app; the work of architectural historians and assisting them to analyse how buildings have evolved over time, and also what proportions of buildings are of particular architectural styles; and building preservation authorities, who, as part of a more detailed record of architectural component geometry and associated architectural semantic content, could use the architectural style of windows and doors in a 3D building model to assist with renovation or replacement following damage to a building, such as that caused by earthquakes or vandalism.

For the latter point this might include using imagery of damaged areas, placing it onto existing 3D building models, for the geospatial analysis of damage. Automated damage assessment in such a manner would, arguably, be far quicker than using purely tabular data. Such analysis might facilitate, and ultimately expedite, repair to an older building, which is therefore likely to be an amalgam of architectural styles, by enabling detailed planning of the work of restoration teams with different specialisms e.g. Norman-window repair specialists. Crucially, the object detection methods in this thesis might also be used to detect the damage on such texture map images, if they are mapped to 3D building models which *already* contain geometry and labelling for architectural components, as follows. Firstly, damage could be 'detected' as false negatives on known object locations, e.g. using a window or door template on what was originally a window or door instance on the building. Secondly, templates for damaged window or door objects could also be used – true positives would then reveal which windows and doors were damaged. Further use cases for this work are summarised in Section 8.5.

8.4 Comparison with Other Methods

8.4.1 Quantitative Comparison

Putting aside for a moment the issues regarding direct comparison of the results achieved by the work described in this thesis with the results from existing methods, and also the issues with comparing between the results of those existing methods (on both counts see Section 8.4.2), the results achieved here are compared with those achieved with relevant existing methods, below.

Firstly, taking the state-of-the art in window and door detection in imagery, i.e. Liu *et al.* (2017), the methods in this thesis achieved a mean F-measure of 0.68 for window and door detection combined (0.65 for the SVM), compared to Liu *et al*'s 93% and 91% pixel accuracy for window and door detection respectively. While the result value from this thesis is lower, note that, unlike the work in this thesis, Liu *et al* used a symmetry constraint in their pipeline and only applied their methods to (i) the *ECP Facades Database* i.e. a dataset which only contains Classical-style buildings. The work undertaken here did not impose a symmetry constraint and covered four different architectural styles, where only one of those was a Classical style (Georgian-Regency). As such, the mean F-measure for window and door detection on Georgian-Regency buildings here was 0.86 (0.84 for the SVM), which compares much more favourably with Liu *et al*'s result.

In addition, note that Liu *et al* do not mention another dataset used for training and that (i) only contains 104 images. Consequently, arguably, a percentage of the images in (i) will have been set aside by Liu *et al* for training i.e. not all the images will have been used to provide the results. While not apparent from their work, if the commonly used 1:1 ratio for the number of training to test images was used then only 52 images would have been used for testing, but of course the proportion of training images used could be higher. Contrast this with the 442 images used across the 3D building models used in the work described in this thesis. Note, that (i) contains approximately 15 instances of windows or doors per image, or approximately 750 test instances in total if a 1:1 split of training to test images was used. If the proportion of training images used by Liu *et al* were higher, the number of window and door instances would, of course, be lower. Compare this to 337 window or door instances used in this work (956 when rendered across the 3D building models). Of course, the 337 instances in this work covered the four aforementioned architectural styles though, unlike Liu *et al*.

The work of Teboul *et al.* (2013) is, arguably, more relevant to this work because, unlike Liu *et al*, they did not impose a symmetry constraint as part of their pipeline. Teboul *et al* also used (i), achieving 81% and 84% for window and door detection respectively, which is bettered by the 0.86 achieved here. While no symmetry constraints were used in either method, the architectural styles of the buildings used to obtain each of these results were of the Classical tradition i.e. typified by symmetrical arrangement of objects on façades. That said, one would argue that the lack of symmetry constraint meant that the use of Classical architectural styles was of less relevance to the strong results, but that the standardisation of the designs of windows (and to an extent doors) within Classical styles plays a key part in the high performance (and for the results of Liu *et al*, above). Incidentally, it would be interesting to see how Teboul *et al*'s method performs on older buildings and architectural styles typified by less symmetry and regularity and less standardisation of window and door designs (such as pre-Classical styles).

Lastly, and, in one sense, of most relevance for this work is the approach of Yu and Wang (2016) on the basis that they used *tens of thousands* of street-view images (captured from vehicles) which, arguably, therefore demonstrated higher levels of variability in the designs of windows and doors than (i). By way of illustration, contrast this with the *hundreds* of images of similar architectural styles in each of the two most popular reference datasets i.e. (i) and (ii) *eTRIMS*. Yu and Wang obtained F-measures of 0.54 and 0.52 for window and door detection respectively, which are bettered by the 0.68 F-measure, for all 3D building models, achieved by the work described in this thesis, i.e. for all four styles. While Yu and Wang used a significantly larger dataset than that used here, their lower performance, compared to Liu *et al* and Teboul *et al*'s results, illustrates the point, once again, that variability of window and door design is one of the biggest challenges to detecting window and door instances 'in the wild', successfully. Such a consideration would need to be borne in mind should the Ordnance Survey roll-out the methods from this thesis across Great Britain.

8.4.2 Issues with Comparing Results

To reiterate, to the best of this author's knowledge, no reference dataset exists of texture-mapped 3D building models marked-up with ground truths for window and door objects, and their architectural style. Consequently, comparison of the results achieved in this thesis with those obtained using other methods can only ever be indicative. In addition, while (i) and (ii) are the most popular reference façade datasets in the field, other datasets are used too – see Chapter 2. As such, comparison between the results obtained with other methods can sometimes only be indicative too.

Moreover, the comparison of methods which use different performance metrics can only be indicative. For example, this method used F-measure, while the methods which use (i) and (ii) use pixel accuracy. Even the abovementioned F-measure results of Yu and Wang cannot be directly compared with those described in this thesis since the former split the result into window and door values, while the work here aggregates the two. So, caution needs to be applied when comparing results with other methods in the field.

8.5 Applications

The above achievements present exciting possibilities for a number of use cases, which are outlined below. As such, the following is a summary of the use cases covered in Chapter 2, taking into account the findings of this work.

Knowledge regarding the architectural style of the window and door objects on a 3D building model could enable augmented and virtual reality applications within the cultural heritage domain. Imagine, for example, being able to incorporate a fun educational test into an augmented reality application for a building or neighbourhood of architectural and historic significance: "Is this window (A) Gothic or (B) Norman?" Or, picture being taken on a tour in such an application whereby the user is informed of the different stages of a building's architectural development. Equally, a record of the architectural styles and ages of windows and doors could assist local authorities and building preservation agencies in the cataloguing of cultural heritage, where the identification of style can be time consuming to deduce manually. See Section 8.3 for an expansion of some of the cultural heritage applications for this work. Lastly, the methods could be used to judge the prevailing style for a region, which could be of use to architectural historians.

Two particular use cases of this work are the potential to refine a BEM or building noise model with knowledge of the construction materials used. BEM and noise modelling are of increasing importance in urban planning, including in local authority planning applications. In that architectural style can be used as a proxy of building age, the likely construction materials of a building could be inferred, based on existing knowledge regarding changes in the use of construction materials over time. Different construction materials possess different thermal and sound insulating capabilities – modern insulating foams can be up to 50 times more effective as a thermal insulator than brick alone (see: Autodesk, 2018; Hopkins, 2012, pp. 608-609). Consequently, older buildings, which tend to have an absence of modern insulating materials, are usually poorer insulators for both heat and sound. An estimate of the construction materials used in a building, as inferred from its age, can therefore enable a potentially more refined building energy or noise model, versus an approach that makes generalised assumptions about construction materials.

Another use case for building age is for building insurance risk determination. Furthermore, knowing the age of a building would enable more accurate property valuation modelling. Finally, building age could be useful to water and sewerage infrastructure providers, assisting them to prioritise their pipework renewal program.

The identification of windows and doors in this work has been in the context of detecting their architectural style, and not generic window or door detection. That said, in identifying the style of a window or a door, clearly one is also identifying the more generic class of ‘window’ or ‘door’. Knowledge of this generic class enables some additional important use cases.

Knowing the positions and sizes of apertures (windows and doors) on a building makes it possible to produce a more refined BEM. This is on the basis that (i) windows and doors tend to lose more heat than walls or roofs, and (ii) total aperture size can be used to refine the model further. Equally, such knowledge enables more accurate noise modelling, on the basis that windows and doors tend to be poorer insulators of sound.

A further use case relating to windows is the ‘right to light’ impact on existing buildings, for example due to the potential construction of a new building nearby. Such impacts can be modelled accurately using viewsheds if the positions and dimensions of windows in the existing buildings are known.

Emergency response planning can be improved through access to a 3D building model containing window and door geometry and semantics. Consider the possibility of using the 3D building model in the cab of an emergency vehicle while approaching an incident to plan a building search, based on door and window locations. In some scenarios it could be possible to use the model to create a flight path for a UAV. This would be useful at particularly hazardous incidents or ones where an urgent assessment is required.

The location of windows and doors can also infer the number of storeys in a building. In turn this can enable 3D cadastre cataloguing, provided any intra-floor ownership divisions are also known. Moreover, knowledge of the location of the entrance to a property is crucial for the emergency services, as indicated above, and also useful for delivery companies.

Additionally, automated smart city building controls, such as temperature and ventilation management using sensors and associated servo-assisted window opening, can be enabled if a 3D building model, which forms part of the control system, includes window geometry and semantics.

8.6 Best-Practice Recommendations

(1) Image-capture best practice. This work has, in facing challenges regarding image suitability, determined the following best practices for image-capture with regard to texture-mapping a 3D building model (although the points below are equally applicable to façade imagery), as follows:

- Images should be taken square-on to the façade, with as little perspective distortion or skew of the façade boundary in the image as possible. If necessary images containing perspective distortion should be corrected in advance using projective transform tools. On this point note, once again, for example, how the window and door detection results from Macák and Drbohlav (2011) improved when rectified images were used, a finding which is also applicable for the methods used here.
- Images should be as 'in focus' on the façade region as is possible – a lack of sharpness in the image can cause issues for object detection methods, such as HOG-based template matching.
- For the purposes of object detection, the image pixel dimensions should be such that window and door objects on the image are large enough to be picked-up by the detection method used (noting that object detail is lost as the pixel count reduces). For HOG-based template matching at least one of the width or height of object instances should be at least 75 pixels high.
- Again, for object detection, objects should not be occluded.

The above points relate not only to image capture carried-out manually, but also to the automated capture of imagery from vehicles. The latter is of particular relevance to the Ordnance Survey, noting, once again, that the methods in this thesis would need to be applied to the whole of Great Britain such that image capture from vehicles would be highly likely (noting the points above regarding imagery needing to be square-on to the façade, as opposed to UAV-captured oblique imagery for example).

(2) CityGML best-practice. When creating a CityGML 3D building model, noting that the CityGML standard allows different syntactic structures to be used to represent the same object, this work has determined the following best practices:

- All surfaces should be categorised as either *GroundSurface*, *WallSurface* or *RoofSurface*, which could be achieved with an automated surface analyser such as that in 3DIS' *SketchUp CityEditor* plug-in (3DIS, 2017).

- To promote automated enrichment, and to facilitate automated injection of new CityGML content resulting from object detection, CityGML for new windows and doors should be placed into the CityGML *bldg:opening* syntax, within a *WallSurface* or, possibly, for windows, a *RoofSurface* element. The same would also apply if producing window or door geometry manually.

- A window or a door detected via object detection (or if modelling manually) should be inserted into a *gml:interior gml:LinearRing*, which in turn should be placed into the existing *gml:exterior gml:LinearRing* on which the object was detected. This will necessitate clipping the corresponding texture map image. Specifically, the *app:textureCoordinates* and *gml:posList* must be modified to remove the hole in the existing *gml:exterior gml:LinearRing*. The *app:textureCoordinates* and *gml:posList* for the new *gml:interior gml:LinearRing* must take the coordinates of the hole.

8.7 Shortfalls & Future Work

While this work has achieved successful detection of window and door objects including their architectural style, especially for Georgian-Regency, and obtained parity between the SVM and 'no SVM' approaches, two principle shortfalls with the methods in this thesis remain apparent:

(i) Detection of fine granularity architectural style is not consistent and is sometimes poor

(ii) The use of machine learning, specifically the SVM (but also the clustering of templates) has *not* improved performance compared to the use of no SVM

Consequently, a number of areas of future work have been identified with the intention of addressing those shortfalls. A selection of other less significant areas of future work have also been identified. See below.

8.7.1 Replace HOG-based Template Matching with Different Detection Approach (High Significance)

In order to improve the detection of fine detail, serious consideration may need to be given to whether HOG-based template matching is the best approach. A state-of-the-art deep learning method, such as that from Liu *et al.* (2017), though modified to remove the requirement for symmetry, could be a worthwhile avenue. Acknowledging the need for substantial amounts of training data for such methods, transfer learning could be used to overcome both a shortage of data by learning from another domain area, and an imbalance of the distributions of training and testing observations (Pan, 2015; Yamada *et al.*, 2018, p. 1). A two-stage approach could be adopted, as follows. The first stage would be a detection step, based on the use of a pre-trained and publicly available network, marked up with generic classes such as 'window' and 'door'. Such a pre-trained network might be based on the millions of images from *ImageNet* (Deng *et al.*, 2009). Stage two would be a recognition phase, using far fewer, but more specific, examples, namely windows and doors marked up with architectural style (for which, as noted in this thesis, there is a paucity of suitable available data).

8.7.2 Feature Selection (High Significance)

If the HOG-based template matching approach is maintained, then improvements in performance might be obtained through the use of more features within the SVM, and perhaps further feature selection on the existing features. The latter might include possible regularisation and transformation steps. Additional features could include those used to improve the classification of 'window' versus 'door', or to remove false positives. Features such as these might include: real-world measurements equivalent to those used in heuristics 2 and 3; and contextual proximity, such as the relative positions of types of objects to other types of object and their positions on a façade. Other features could be used to detect more detail, such as particular architectural styles. Features which might be of use in that respect could include descriptors, noting, for example, that Dalal and Trigg's original HOG implementation used the HOG descriptor itself as a feature in classification using an SVM.

8.7.3 Match-Score Normalisation Scheme (High Significance)

While additional features could improve performance, there remains a question about the suitability of the match-score normalisation scheme used. As such, even if additional features were added (see above) the existing match-score features may be unsuitable, based on: (i) the tendency for the scheme to favour larger templates and (ii) the overlapping of the distributions of the positive-class and negative-class observations. A review of the match-score normalisation scheme would therefore be prudent, in that it could lead to a greater differentiation between match-scores for poor positive instances and negative instances, and therefore lead to more true positives, fewer false positives and fewer false negatives as a result. To clarify, the steps outlined in Section 8.7.2 could be *complementary* to a review of the normalisation scheme.

8.7.4 Additional Heuristics (High Significance)

The inclusion of more heuristics might improve the performance. One such heuristic could be to 'snap' rows of window candidate matches per floor to a horizontal alignment and to give the candidate matches a common size and interval spacing. This would only be exercised for styles typified by such arrangements, such as Classical styles. The exercising of such a heuristic would therefore require either *a priori* style information, or a high number of objects detected for a style.

Another heuristic might include the detection of a relative geometry incline, or perhaps the extraction of the type of surface from the CityGML, to infer if the surface is tilted and would therefore lack doors in the real world. Perhaps material detection strategies could also be employed to filter out false positives on brickwork and tiled regions. Lastly, the coherent line drawing method from Kang *et al* might be used to eliminate candidate matches with few strong edges.

8.7.5 Templates – Clustering, Cropping & Masking (High Significance)

Improving the distribution of the sort of architectural designs found in the real world, but represented in the template examples used for clustering, could improve the selection of exemplars, and lead to performance gains during HOG-based template matching. If the number of clusters reduced as a result, then runtime would also be improved. In addition, the 'cropped-ness' of the candidate matches could be refined, perhaps using the coherent line drawing method to detect dominant outer edges. Exemplar templates could also be cropped in advance using a similar approach. Lastly, the input parameters for clustering might be reviewed to see if optimisation can be achieved.

Further template considerations are as follows.

Despite the potential for glazing to be discriminative for some styles, dominant edges could be used to produce a glazing mask from the templates. HOG-based template matching would be carried out using the masked template on texture map images, where the mask would be applied to each grid cell.

Two sets of additional templates to collect are Hausmannian and *Restauration* style, which would allow comparison with state-of-the-art façade segmentation approaches. That said, a pixel-based ground-truthing approach would need to be implemented to allow comparison with those state-of-the-art results.

8.7.6 Geometry Processing (Low Significance)

An analysis of the reasons why some *gml:LinearRing*s were filtered out by Chapter 5's validation check (4) might indicate additional pipeline steps that could reduce the number filtered out. Such a reduction would improve the scalability of the methods to unseen 3D building models. Bear in mind, however, that the proportion of *gml:LinearRing*s processed for the 3D building models in this work was high. Separately, geometry processing could also be improved by adding steps to process *gml:interior gml:LinearRings*.

8.7.7 Texture Mapping & Texture Map Image Quality (Low Significance)

The ability to deal with shear-based mapping would also improve scalability to unseen 3D building models, as would the inclusion of projected textures using *TexCoordGen*. That said, note that the incidence of sheared and projected textures in the 3D building models used here was <1% and zero respectively.

Perhaps an image quality check could be added, maybe to check for sharpness. However, this could mean that fewer 3D building models would be processed as a result. Perhaps therefore the 'higher significance' steps outlined above would be more worthwhile. Nonetheless, image quality would always need to be a consideration for any computer vision approach.

8.7.8 Automated Injection of new Windows & Doors into CityGML (Low Significance)

Clearly, while this work has produced semantic and geometric enrichment, the automated injection of the new XML content into the existing CityGML file could be key for any applications of the methods used here. Moreover, perhaps a plug-in for the Trimble *SketchUp* could be created which would automatically detect missing windows and doors.

8.7.9 Richer Semantic Content (Low Significance)

Finally, image matching between texture map images and captioned images such as *Flickr* images could be conducted, from which the *Flickr* tags would be used to produce more detailed, instance-based captioning for objects. Guide book texts could then be mined to produce even richer semantic content, while alignment with 2D floor plans could introduce the possibility for interior semantic and geometric enrichment.

Bibliography

3DIS, 2017. Der CityEditor, https://www.3dis.de/loesungen/3d-stadtmodelle/cityeditor/. (Accessed 30 October 2014)

Affara, L., Nan, L., Ghanem, B., Wonka, P., 2016. Large Scale Asset Extraction for Urban Images, in: Leibe, B., Matas, J., Sebe, N., Welling, M. (Eds.), Lecture Notes in Computer Science (LNCS) - Computer Vision - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016 - Proceedings, Part III, first ed. Springer, Cham, pp. 437-452.

Agarwal, S., Furukawa, Y., Snavely, N., Simon, I., Curless, B., Seitz, S.M., Szeliski, R., 2011. Building Rome in a day. Communications of the ACM 54, (10), 105-112.

Agudo, M.d.I.Á.U., Fragero, J.I.M., Talaverano, R.M., 2016. Virtual models for archaeological research and 2.0 dissemination: The early medieval church of San Cebrián de Mazote (Spain). SCIentific RESearch and Information Technology - Ricerca Scientifica e Tecnologie dell'Informazione 6, (2), 93-108.

Aijazi, A.K., Checchin, P., Trassoudaine, L., 2016. Automatic Detection and Feature Estimation of Windows from Mobile Terrestrial LiDAR Data, in: Menegatti, E., Michael, N., Berns, K., Yamaguchi, H. (Eds.), Intelligent Autonomous Systems 13, first ed. Springer International Publishing, Berlin Heidelberg, pp. 659-674.

Alahi, A., Ortiz, R., Vandergheynst, P., 2012. FREAK: Fast Retina Keypoint, in: Mortensen, E.N. (Ed.), Proc. 24th Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, Providence, USA, pp. 510-517.

Alegre, F., Dellaert, F., 2004. A probabilistic approach to the semantic interpretation of building facades, Proc. International Workshop on Vision Techniques Applied to the Rehabilitation of City Centres. CIPA, Lisbon, Portugal, pp. 25-27.

Ali, H., Seifert, C., Jindal, N., Paletta, L., Paar, G., 2007. Window Detection in Facades, in: Werner, B. (Ed.), Proc. Image Analysis and Processing, 2007. ICIAP 2007. 14th International Conference on. IEEE, Modena, Italy, pp. 837-842.

Allen, C., 1994. Performance Standards for Timber Sash and Case Windows, Report Tecnical Advice Note, Historic Scotland, http://www.buildingconservation.com/articles/sashwindows/sash_windows.htm. (Accessed 06 November 2017)

Alpaydin, E., 2014. Introduction to Machine Learning, third ed. MIT Press, Cambridge.

Amirebrahimi, S., Rajabifard, A., Sabri, S., Mendis, P., 2016. Spatial Information in Support of 3D Flood Damage Assessment of Buildings at Micro Level: a Review. ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences IV-2/W1, 73-81.

Ammon, S., 2017. Epilogue: the rise of imagery in the age of modeling, in: Ammon, S., Capdevila-Werning, R. (Eds.), Philosophy of Engineering and Technology - The Active Image - Architecture and Engineering in the Age of Modeling, first ed. Springer, Cham.

Anthony, G., Gregg, H., Tshilidzi, M., 2007. Image classification using SVMs: one-against-one vs one-against-all, in: Negara, P.R.S. (Ed.), Proc. 28th Asian Conference on Remote Sensing (ACRS). Pusat Remote Sensing Negara, Kuala Lumpar, Malaysia, pp. 801-806.

Antibody Software, 2017. Bulk Image Downloader, http://bulkimagedownloader.com/. (Accessed 19 November 2017)

Apollonio, F.I., 2016. Classification Schemes for Visualization of Uncertainty in Digital Hypothetical Reconstruction, in: Münster, S., Pfarr-Harfst, M., Kuroczyński, P., Ioannides, M. (Eds.), 3D Research Challenges in Cultural Heritage II: How to Manage Data and Knowledge Related to Interpretative Digital 3D Reconstructions of Cultural Heritage, first ed. Springer International Publishing, Cham, pp. 173-197.

Arikan, M., Schwärzler, M., Flöry, S., Wimmer, M., Maierhofer, S., 2013. O-snap: Optimization-based snapping for modeling architecture. ACM Transactions on Graphics (TOG) 32, (1), 1-15.

Arroyo Ohori, K., Biljecki, F., Diakité, A., Krijnen, T., Ledoux, H., Stoter, J., 2017. Towards an integration of GIS and BIM data: what are the geometric and topological issues? ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences IV-4/W5, 1-8.

Atazadeh, B., Kalantari, M., Rajabifard, A., 2016a. Comparing Three Types of BIM-based Models for Managing 3D Ownership Interests in Multi-level Buildings, in: van Oosterom, P. (Ed.), Proc. 5th International Workshop on 3D Cadastres. FIG, Athens, Greece, pp. 183-198.

Atazadeh, B., Kalantari, M., Rajabifard, A., Ho, S., Ngo, T., 2016b. Building Information Modelling for High-rise Land Administration. Transactions in GIS 21, (1), 91-113.

Autodesk, 2017. AutoCAD, https://www.autodesk.co.uk/products/autocad/overview. (Accessed 02 October 2017)

Autodesk, 2018. Insulation, https://sustainabilityworkshop.autodesk.com/buildings/insulation. (Accessed 24 Janaury 2018)

Ballardini, A.L., Cattaneo, D., Fontana, S., Sorrenti, D.G., 2016. Leveraging the OSM building data to enhance the localization of an urban vehicle, in: Antoniou, C., Çelikoğlu, H.B. (Eds.), Proc. 19th International Conference on Intelligent Transportation Systems (ITSC). IEEE, Rio de Janeiro, Brazil, pp. 622-628.

Barrile, V., Bilotta, G., Lamari, D., 2017. 3D models of cultural heritage. International Journal of Mathematical Models and Methods in Applied Sciences 11, 1-8.

Bassier, M., Vergauwen, M., Van Genechten, B., 2016. Automated Semantic Labelling of 3D Vector Models for Scan-to-BIM, in: Anderson, M.S.T., Anderson, P. (Eds.), Proc. 4th Annual International Conference on Architecture and Civil Engineering (ACE). GSTF, Singapore, pp. 93-100.

Bay, H., Tuytelaars, T., Van Gool, L., 2006. SURF: Speeded up robust features, in: Leonardis, A., Bischof, H., Pinz, A. (Eds.), Lecture Notes in Computer Science (LNCS) - Computer Vision - ECCV 2006 - 9th European Conference on Computer Vision, Graz, Austria, May 7-13, 2006 - Proceedings, Part I, first ed. Springer, Berlin, Heidelberg, pp. 404-417.

Beetz, J., Blümel, I., Dietze, S., Fetahui, B., Gadiraju, U., Hecher, M., Krijnen, T., Lindlar, M., Tamke, M., Wessel, R., Yu, R., 2016. Enrichment and Preservation of Architectural Knowledge, in: Münster, S., Pfarr-Harfst, M., Kuroczyński, P., Ioannides, M. (Eds.), 3D Research Challenges in Cultural Heritage II: How to Manage Data and Knowledge Related to Interpretative Digital 3D Reconstructions of Cultural Heritage, first ed. Springer, Cham, pp. 231-255.

Bell, A., 1914. Architecture, second ed. T.C. & E.C. Jack, London.

Benedek, C., Shadaydeh, M., Kato, Z., Szirányi, T., Zerubia, J., 2015. Multilayer Markov Random Field models for change detection in optical remote sensing images. ISPRS Journal of Photogrammetry and Remote Sensing 107, 22-37.

Bentley Systems, 2017. MicroStation, https://www.bentley.com/en/products/brands/microstation. (Accessed 02 October 2017)

Berck, C., 2017. Aug City: The Cyber-Spatial Impacts of Augmented Reality on the Field of Urban Planning, MCRP Thesis, University of Nebraska, Lincoln

Bernardini, G., Azzolini, M., D'Orazio, M., Quagliarini, E., 2016. Intelligent evacuation guidance systems for improving fire safety of Italian-style historical theatres without altering their architectural characteristics. Journal of Cultural Heritage 22, 1006-1018.

Berning, D., 2011. Georgian Architecture: how to Identify the Greek Orders, The Guardian, London, 11 September 2011

Biagini, C., Capone, P., Donato, V., Facchini, N., 2016. Towards the BIM implementation for historical building restoration sites. Automation in Construction 71 Part 1, 74-86.

Biljecki, F., Arroyo Ohori, K., Ledoux, H., Peters, R., Stoter, J., 2016a. Population Estimation Using a 3D City Model: A Multi-Scale Country-Wide Study in the Netherlands. PLOS ONE 11, (6), 1-22.

Biljecki, F., Ledoux, H., Du, B., Stoter, J., Soon, K.H., Khoo, V.H.S., 2016b. The Most Common Geometric and Semantic Errors in CityGML

Datasets. ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences IV-2/W1, 23-30.

Biljecki, F., Ledoux, H., Stoter, J., 2016c. Generation of Multi-LOD 3D City Models In CityGML with the Procedural Modelling Engine RANDOM3DCITY. ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences IV-4/W1, 51-59.

Biljecki, F., Ledoux, H., Stoter, J., 2017. Generating 3D city models without elevation data. Computers, Environment and Urban Systems 64, 1-18.

Biljecki, F., Sindram, M., 2017. Estimating Building Age with 3D GIS. ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences IV-4/W5, 17-24.

Biljecki, F., Stoter, J., Ledoux, H., Zlatanova, S., Çöltekin, A., 2015. Applications of 3D City Models: State of the Art Review. ISPRS International Journal of Geo-Information 4, (4), 2842-2889.

Blake, J., 2008. On Defining the Cultural Heritage. International and Comparative Law Quarterly 49, (1), 61-85.

Boguslawski, P., Mahdjoubi, L., Zverovich, V., Fadli, F., 2016. Automated construction of variable density navigable networks in a 3D indoor environment for emergency response. Automation in Construction 72 Part 2, 115-128.

Bostanci, E., Kanwal, N., Clark, A.F., 2015. Augmented reality applications for cultural heritage using Kinect. Human-centric Computing and Information Sciences 5, 1-18.

Boyes, G.A., Ellul, C., Irwin, D., 2017. Exploring BIM for Operational Integrated Asset Management – a Preliminary Study Utilising Real-World Infrastructure Data. ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences IV-4/W5, 49-56.

Brasebin, M., Christophe, S., Jacquinod, F., Vinesse, A., Mahon, H., 2016. 3D Geovisualization & Stylization to Manage Comprehensive and Participative

Local Urban Plans. ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences IV-2/W1, 83-91.

Brundu, F.G., Patti, E., Osello, A., Del Giudice, M., Rapetti, N., Krylovskiy, A., Jahn, M., Verda, V., Guelpa, E., Rietto, L., Acquaviva, A., 2017. IoT Software Infrastructure for Energy Management and Simulation in Smart Cities. IEEE Transactions on Industrial Informatics 13, (2), 832-840.

Brusaporci, S., 2017. Advances in Media, Entertainment, and the Arts (AMEA) - Digital Innovations in Architectural Heritage Conservation, first ed. IGI Global, Hershey.

Bullivant, L., 2017. 4D Hyperlocal: A Cultural Toolkit for the Open-Source City, first ed. John Wiley & Sons, Oxford.

Caballero Zoreda, L., 2010. Una experiencia en Arqueología de la Arquitectura. Arqueología de la Arquitectura 3, 103-119.

Cadw, 2017. Historic Environment - Records - Cof Cymru - National Historic Assets of WalesCof Cymru - National Historic Assets of Wales, http://cadw.gov.wales/historicenvironment/recordsv1/cof-cymru/?lang=en. (Accessed 10 January 2014)

Cevikalp, H., Triggs, B., 2017. Visual Object Detection Using Cascades of Binary and One-Class Classifiers. Int J Comput Vis 123, (3), 334–349.

Ceylan, D., Mitra, N.J., Zheng, Y., Pauly, M., 2014. Coupled structure-from-motion and 3D symmetry detection for urban facades. ACM Transactions on Graphics (TOG) 33, (1), 1-15.

Chaturvedi, K., Kolbe, T.H., 2016. Integrating Dynamic Data and Sensors with Semantic 3D City Models in the Context of Smart Cities. ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences IV-2/W1, 31-38.

Chatzipoulka, C., Compagnon, R., Nikolopoulou, M., 2016. Urban geometry and solar availability on façades and ground of real urban forms: using London as a case study. Solar Energy 138, 53-66.

Ching, F.D.K., 2011. A Visual Dictionary of Architecture, 1st ed. Wiley, Hoboken.

Ching, F.D.K., 2014. Architecture: Form, Space, and Order, fourth ed. John Wiley & Sons, Hoboken.

Chu, H., Wang, S., Urtasun, R., Fidler, S., 2016. HouseCraft: Building Houses from Rental Ads and Street Views, in: Leibe, B., Matas, J., Sebe, N., Welling, M. (Eds.), Lecture Notes in Computer Science (LNCS) - Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016 - Proceedings, Part VI, first ed. Springer, Cham, pp. 500-516.

Chu, W.-T., Tsai, M.-H., 2012. Visual pattern discovery for architecture image classification and product image search, in: Ip, H.H.S., Rui, Y. (Eds.), Proc. 2nd International Conference on Multimedia Retrieval (ICMR). ACM, Hong Kong, China, pp. 1-8.

City of York Council, 2011. York Central Historic Core Conservation Area Appraisal (HCCAA) - 22 Railway Area, Report 22.Railway area, City of York Council, https://www.york.gov.uk/downloads/file/5915/22railway_area. (Accessed 05 October 2017)

CityDoctor, 2017. CityDoctor for 3D City Models, http://www.citydoctor.eu/index.php/citydoctor_for_3d_city_models.html?language=en. (Accessed 30 October 2017)

Cohen, A., Schönberger, J.L., Speciale, P., Sattler, T., Frahm, J.-M., Pollefeys, M., 2016. Indoor-Outdoor 3D Reconstruction Alignment, in: Leibe, B., Matas, J., Sebe, N., Welling, M. (Eds.), Lecture Notes in Computer Science (LNCS) - Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016 - Proceedings, Part III, first ed. Springer, Cham, pp. 285-300.

Cohen, A., Schwing, A.G., Pollefeys, M., 2014. Efficient Structured Parsing of Facades Using Dynamic Programming, in: Mortensen, E., Fidler, S. (Eds.), Proc. 26th Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, Columbus, USA, pp. 3206-3213.

Curl, J.S., Wilson, S., 2015. The Oxford Dictionary of Architecture, third ed. Oxford University Press, Oxford.

Dalal, N., Triggs, B., 2005. Histograms of oriented gradients for human detection, in: Schmid, C., Soatto, S., Tomas, C. (Eds.), Proc. 17th Computer Society Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, San Diego, USA pp. 886-893.

Dawe, S., 2013. King of the Castles: Britain's Built Heritage Rules Huffington Post, New York, 02 October 2013

DCLG, 2016. English Housing Survey - Housing Stock Report, 2014-15, Report 978-1-4098-4874-5, DLCG, https://www.gov.uk/government/uploads/system/uploads/attachment_data/file/539600/Housing_Stock_report.pdf. (Accessed 15 August 2017)

Debevec, P.E., Taylor, C.J., Malik, J., 1996. Modeling and rendering architecture from photographs: a hybrid geometry-and-image-based approach, in: Spencer, S. (Ed.), Proc. 23rd International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH). ACM, New Orleans, USA, pp. 11-20.

Deggim, S., Kersten, T.P., Lindstaedt, M., Hinrichsen, N., 2017. The Return Of The Siegesburg – 3D-Reconstruction of a Disappeared and Forgotten Monument. International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences XLII-2/W3, 209-215.

Dehbi, Y., Gröger, G., Plümer, L., 2016a. Identification and Modelling of Translational and Axial Symmetries and their Hierarchical Structures in Building Footprints by Formal Grammars. Transactions in GIS 20, (5), 645-663.

Dehbi, Y., Hadiji, F., Gröger, G., Kersting, K., Plümer, L., 2016b. Statistical Relational Learning of Grammar Rules for 3D Building Reconstruction. Transactions in GIS 21, (1), 134-150.

Del Giudice, M., Osello, A., Patti, E., 2014. BIM and GIS for district modeling, in: Mahdavi, A., Bob, M., Scherer, R.J. (Eds.), eWork and eBusiness in Architecture, Engineering and Construction: ECPPM 2014, first ed. CRC Press, Croydon.

Demir, İ., Aliaga, D.G., Benes, B., 2016. Proceduralization for Editing 3D Architectural Models, in: Su, H. (Ed.), Proc. 4th International Conference on 3D Vision (3DV). IEEE, Stanford, USA, pp. 194-202.

Deng, J., Dong, W., Socher, R., Li, L.J., Kai, L., Li, F.-F., 2009. ImageNet: A large-scale hierarchical image database, in: Flynn, P., Mortensen, E. (Eds.), Proc. 21st Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, Miami Beach, USA, pp. 248-255.

Department for Culture Media and Sport, 2010. Principles of Selection for Listing Buildings, Report Department for Culture Media and Sport, https://www.gov.uk/government/uploads/system/uploads/attachment_data/file/137695/Principles_Selection_Listing_1_.pdf. (Accessed 28 January 2018)

Desolneux, A., Leclaire, A., 2017. Stochastic Image Reconstruction from Local Histograms of Gradient Orientation, in: Lauze, F., Dong, Y., Dahl, A.B. (Eds.), Scale Space and Variational Methods in Computer Vision: 6th International Conference, SSVM 2017, Kolding, Denmark, June 4-8, 2017, Proceedings, first ed. Springer International Publishing, Cham, pp. 133-145.

Dhonju, H.K., Xiao, W., Sarhosis, V., Mills, J.P., Wilkinson, S., Wang, Z., Thapa, L., Panday, U.S., 2017. Feasibility Study of Low-Cost Image-Based Heritage Documentation. International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences XLII-2/W3, 237-242.

Diakité, A.A., Damiand, G., Gesquiere, G., 2014. Automatic semantic labelling of 3D buildings based on geometric and topological information, in: Breunig, M., Al-Doori, M., Butwilowski, E., Kuper, P.V., Benner, J., Haefele, K.H. (Eds.), Proc.

9th International 3DGeoInfo Conference. Karlsruhe Institute of Technology, Dubai, United Arab Emirates, pp. 48-62.

Diener, J., 2011. 2D minimal bounding box, https://uk.mathworks.com/matlabcentral/fileexchange/31126-2d-minimal-bounding-box. (Accessed 10 June 2016)

Diez, H.V., García, S., Mujika, A., Moreno, A., Oyarzun, D., 2016. Virtual training of fire wardens through immersive 3D environments, in: Adcock, M., Bednarz, T. (Eds.), Proc. 21st International Conference on Web3D Technology. ACM, Anaheim, USA, pp. 43-50.

Dimitrov, A., Golparvar-Fard, M., 2015. Segmentation of building point cloud models including detailed architectural/structural features and MEP systems. Automation in Construction 51, 32-45.

Doersch, C., Singh, S., Gupta, A., Sivic, J., Efros, A.A., 2015. What makes Paris look like Paris? Communications of the ACM 58, (12), 103-110.

Döllner, J., Baumann, K., Buchholz, H., 2006. Virtual 3D city models as foundation of complex urban information spaces, in: Schrenk, M. (Ed.), Proc. 11th International Conference on Urban Planning and Spatial Development in the Information Society (CORP) - Sustainable Solutions for the Information Society. CORP, Vienna, Austria, pp. 107-112.

Döllner, J., Hagedorn, B., 2007. Integrating urban GIS, CAD, and BIM data by service based virtual 3D city models, in: Rumor, M., Coors, V., Fendel, E.M., Zlatanova, S. (Eds.), Urban and Regional Data Management - Annual, first ed. Taylor & Francis, Leiden, pp. 157-160.

Dore, C., Murphy, M., 2014. Semi-Automatic Techniques for Generating BIM Façade Models of Historic Buildings. Journal of Information Technology in Construction 19, (2), 20-46.

Dore, C., Murphy, M., 2017. Current State of the Art Historic Building Information Modelling. International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences XLII-2/W5, 185-192.

dos Santos, J.M., de Moura, E.S., da Silva, A.S., da Silva Torres, R., 2016. Color and texture applied to a signature-based bag of visual words method for image retrieval. Multimed Tools Appl 76, (15), 16855–16872.

dotPDN, 2017. paint.net, https://www.getpaint.net/. (Accessed 19 November 2017)

Drauschke, M., Förstner, W., 2008. Selecting appropriate features for detecting buildings and building parts. International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences XXXVII Part B3b, 447-452.

Drobež, P., Fras, M.K., Ferlan, M., Lisec, A., 2017. Transition from 2D to 3D real property cadastre: The case of the Slovenian cadastre. Computers, Environment and Urban Systems 62, 125-135.

Drolet, M., 2004. The Postmodernism Reader: Foundational Texts, first ed. Routledge, Padstow.

Dsilva, M.G., 2009. A feasibility study on CityGML for cadastral purposes, Masters Thesis, Eindhoven University of Technology, Eindhoven

Du, S., Zhang, F., Zhang, X., 2015. Semantic classification of urban buildings combining VHR image and GIS data: An improved random forest approach. ISPRS Journal of Photogrammetry and Remote Sensing 105, 107-119.

Duguleana, M., Brodi, R., Girbacia, F., Postelnicu, C., Machidon, O., Carrozzino, M., 2016. Time-Travelling with Mobile Augmented Reality: A Case Study on the Piazza dei Miracoli, in: Ioannides, M., Fink, E., Moropoulou, A., Hagedorn-Saupe, M., Fresa, A., Liestøl, G., Rajcic, V., Grussenmeyer, P. (Eds.), Digital Heritage - Progress in Cultural Heritage - Documentation, Preservation, and Protection - 6th International Conference, EuroMed 2016, Nicosia, Cyprus, October 31 – November 5, 2016 - Proceedings, Part I, first ed. Springer International Publishing, Cham, pp. 902-912.

El-Mekawy, M., Paasch, J., Paulsson, J., 2014. Integration of 3D Cadastre, 3D Property Formation and BIM in Sweden, in: van Oosterom, P. (Ed.), Proc. 4th International Workshop on 3D Cadastres. FIG, Dubai, United Arab Emirates, pp. 1-18.

Ellul, C., Adjrad, M., Groves, P., 2016. The Impact of 3D Data Quality on Improving GNSS Performance Using City Models Initial Simulations. ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences IV-2/W1, 171-178.

English Heritage, 2017. Modern: Architecture, http://www.english-heritage.org.uk/learn/story-of-england/modern/architecture/. (Accessed 05 October 2017)

Esri, 2017. ArcMap, http://desktop.arcgis.com/en/arcmap/. (Accessed 20 November 2017)

Everingham, M., Van Gool, L., Williams, C.I., Winn, J., Zisserman, A., 2010. The PASCAL Visual Object Classes (VOC) Challenge. Int J Comput Vis 88, (2), 303-338.

Felus, Y., Barzani, S., Caine, A., Blumkine, N., Van Oosterom, P., 2014. Steps towards 3D cadastre and ISO 19152 (LADM) in Israel, in: van Oosterom, P. (Ed.), Proc. 4th International Workshop on 3D Cadastres. FIG, Dubai, United Arab Emirates, pp. 1-20.

Felzenszwalb, P.F., Girshick, R.B., McAllester, D., Ramanan, D., 2010. Object Detection with Discriminatively Trained Part-Based Models. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) 32, (9), 1627-1645.

Fernie, E., 2002. The Architecture of Norman England, first ed. Oxford University Press, Oxford.

Fiegel, L., 2011. Modernism: a concrete utopia, The Guardian, London, 11 September 2011

Fischler, M.A., Bolles, R.C., 1981. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Communications of the ACM 24, (6), 726-740.

Fisher, R.B., Breckon, T.P., Dawson-Howe, K., Fitzgibbon, A., Robertson, C., Trucco, E., Williams, C.K.I., 2014. Dictionary of Computer Vision and Image Processing, 2nd ed. Wiley, Chichester.

Fleming, J., Honour, H., Pevsner, N., 1991. The Penguin Dictionary of Architecture, 5th ed. Penguin Books, London.

Fletcher, B.F., 1905. A History of Architecture on the Comparative Method for the Student, Craftsman, and Amateur, fifth ed. Batsford, London.

Flickr, 2017. Flickr, http://www.flickr.com. (Accessed 10 January 2014)

florin, 2010. Checking to see if 3 points are on the same line, https://stackoverflow.com/a/3813755. (Accessed 28 June 2014)

Frey, B.J., Dueck, D., 2007. Clustering by Passing Messages Between Data Points. Science 315, (5814), 972-976.

Fritsch, D., Klein, M., 2017. 3D preservation of buildings – Reconstructing the past. Multimed Tools Appl, 1-18.

Frommholz, D., Linkiewicz, M., Meissner, H., Dahlke, D., Poznanska, A., 2015. Extracting Semantically Annotated 3D Building Models with Textures from Oblique Aerial Imagery. International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences XL-3/W2, 53-58.

Furukawa, Y., Ponce, J., 2010. Accurate, Dense, and Robust Multiview Stereopsis. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) 32, (8), 1362-1376.

Gadde, R., Jampani, V., Marlet, R., Gehler, P.V., in press. Efficient 2D and 3D Facade Segmentation using Auto-Context. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) PP, (99), 1-1.

Gadde, R., Marlet, R., Paragios, N., 2016. Learning Grammars for Architecture-Specific Facade Parsing. Int J Comput Vis 117, (3), 1-27.

Garcia-Dorado, I., Aliaga, D.G., Bhalachandran, S., Schmid, P., Niyogi, D., 2017. Fast Weather Simulation for Inverse Procedural Design of 3D Urban Models. ACM Transactions on Graphics (TOG) 36, (2), 1-19.

Garozzo, R., Murabito, F., Santagati, C., Pino, C., Spampinato, C., 2017. Culto: an Ontology-Based Annotation Tool for Data Curation in Cultural Heritage. International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences XLII-2/W5, 267-274.

Gatta, G., Arioti, E., Bitelli, G., 2017. Geomatics science applied to cartographic heritage and archive sources: A new way to explore the XIXth century Gregorian Cadastre of Bologna (Italy), an ante-litteram 3D GIS. Journal of Cultural Heritage 23, 68-76.

Geograph Project, 2017. Geograph, http://www.geograph.org.uk/. (Accessed 02 February 2014)

GeoInformation Group, 2016. UKBuildings Dataset, http://www.geoinformationgroup.co.uk/#!ukbuildings/c1his. (Accessed 10 June 2015)

Georgopoulos, A., Stathopoulou, E.K., 2017. Data Acquisition for 3D Geometric Recording: State of the Art and Recent Innovations, in: Vincent, M.L., López-Menchero Bendicho, V.M., Ioannides, M., Levy, T.E. (Eds.), Quantitative Methods in the Humanities and Social Sciences - Heritage and Archaeology in the DigitalAge: Acquisition, Curation, and Dissemination of Spatial Cultural Heritage Data, first ed. Springer International Publishing, Cham, pp. 1-26.

Getty Images, 2017. Georgian Door Details - A selection of Georgian front door details, https://www.gettyimages.co.uk/detail/illustration/georgian-door-details-royalty-free-illustration/165905484. (Accessed 07 November 2017)

Gill, H., 1916. The Church Windows of Nottinghamshire. Transactions of the Thoroton Society 20, 1-4.

Gkeli, M., Ioannidis, C., Potsiou, C., 2017. Review of the 3D Modelling Algorithms and Crowdsourcing Techniques. An Assessment of their Potential for 3D Cadastre, in: Potsiou, C., Halme, P., Friis-Hansen, L. (Eds.), Proc. Working Week - Surveying the World of Tomorrow - From Digitalisation to Augmented Reality. FIG, Helsinki, Finland, pp. 1-24.

Goesele, M., Snavely, N., Curless, B., Hoppe, H., Seitz, S.M., 2007. Multi-View Stereo for Community Photo Collections, in: Scheirer, W. (Ed.), Proc. 11th International Conference on Computer Vision (ICCV). IEEE, Rio de Janeiro, Brazil, pp. 1-8.

Gorst, T., 2003. The Buildings Around Us, first ed. Taylor & Francis, Cambridge.

Grilli, E., Menna, F., Remondino, F., 2017. A Review of Point Clouds Segmentation and Classification Algorithms. International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences XLII-2/W3, 339-344.

Gröger, G., Kolbe, T., Nagel, C., Häfele, K., 2012. OGC City Geography Markup Language (CityGML) En-coding Standard, Report OGC 12-019, Open Geospatial Consortium (OGC), https://portal.opengeospatial.org/files/?artifact_id=47842. (Accessed 19 May 2015)

Gröger, G., Plümer, L., 2012. CityGML–Interoperable semantic 3D city models. ISPRS Journal of Photogrammetry and Remote Sensing 71, 12-33.

Guan, L., Ding, Y., Feng, X., Zhang, H., 2016. Digital Beijing construction and application based on the urban three-dimensional modelling and remote sensing monitoring technology, in: Wang, C., Weng, Q. (Eds.), Proc. 36th International Geoscience and Remote Sensing Symposium (IGARSS) - Advancing the Understanding of Our Planet. IEEE, Beijing, China, pp. 7299-7302.

Gulliver, T., Haanen, A., Goodin, M., 2016. A 3D Digital Cadastre for New Zealand by 2021: Leveraging the Current System and Modern Technology, in: van Oosterom, P. (Ed.), Proc. 5th International Workshop on 3D Cadastres. FIG, Athens, Greece, pp. 473-490.

Ham, Y., Golparvar-Fard, M., 2015. Mapping actual thermal properties to building elements in gbXML-based BIM for reliable building energy performance modeling. Automation in Construction 49 Part B, 214–224.

Hamid, Q., Chauhdry, M.H., Mahmood, S., Farid, M.S., 2016. Arc GIS and 3D Visualization of Land Records: A Case Study of Urban Areas in Punjab. National Academy Science Letters 39, (4), 277–281.

Hamlin, A.D.F., 1916. A History of Ornament - Ancient and Medieval, first ed. The Century Co., New York.

Hanser, D.A., 2006. Architecture of France, first ed. Greenwood Press, Westport.

Hart, S., 2010. Medieval Church Window Tracery in England, first ed. Boydell Press, Woodbridge.

Harvey, P., 2015. Re: PNG Pixels Per Unit value not updating - Reply #4, http://u88.n24.queensu.ca/exiftool/forum/index.php/topic,5950.msg29287.html?PHPSESSID=ref52016vii792t8at9n11faq6#msg29287. (Accessed 20 June 2014)

Harvey, P., 2017. EXIF Tool by Phil Harvey, https://www.sno.phy.queensu.ca/~phil/exiftool/. (Accessed 20 June 2014)

Hassaballah, M., Abdelmgeid, A.A., Alshazly, H.A., 2016. Image Features Detection, Description and Matching, in: Awad, A.I., Hassaballah, M. (Eds.), Image Feature Detectors and Descriptors : Foundations and Applications, first ed. Springer International Publishing, Cham, pp. 11-45.

Hastie, T., Tibshirani, R., Friedman, J., 2009. The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Second Edition, second ed. Springer-Verlag, New York.

Hauagge, D.C., Snavely, N., 2012. Image matching using local symmetry features, 24th Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, Providence, USA, pp. 206-213.

Hendricks, S., 1999. Point of Entry: an Overview of Historic Doors. Traditional Building 12, (5), 1-4.

Henn, A., Gröger, G., Stroh, V., Plümer, L., 2013. Model driven reconstruction of roofs from sparse LIDAR point clouds. ISPRS Journal of Photogrammetry and Remote Sensing 76, 17-29.

Henn, A., Römer, C., Gröger, G., Plümer, L., 2012. Automatic classification of building types in 3D city models. GeoInformatica 16, (2), 281-306.

Historic England, 2006. The Senate House, Cambridge, Cambridgeshire, http://www.heritage-explorer.co.uk/web/he/searchdetail.aspx?id=5851. (Accessed 20 November 2017)

Historic England, 2017. Listing - Search the List, https://historicengland.org.uk/listing/the-list/. (Accessed 10 January 2014)

Historic Environment Scotland, 2017. Listing, scheduling and designations - Listed buildings - Search for a listed building https://www.historicenvironment.scot/advice-and-support/listing-scheduling-and-designations/listed-buildings/search-for-a-listed-building/. (Accessed 10 Jaunary 2014)

Ho, S., Rajabifard, A., 2016. Towards 3D-enabled urban land administration: Strategic lessons from the BIM initiative in singapore. Land Use Policy 57, 1-10.

Holland, D.A., Pook, C., Capstick, D., Hemmings, A., 2016. The Topographic Data Deluge – Collecting & Maintaining Data in a 21st Century Mapping Agency. International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences XLI-B4, 727-731.

Hopkins, C., 2012. Sound Insulation, first ed. Taylor & Francis, Oxford Burlington.

Horne, L., Alvarez, J., McCarthy, C., Salzmann, M., Barnes, N., 2016. Semantic labeling for prosthetic vision. Computer Vision and Image Understanding 149, 113-125.

Howell, S., Hippolyte, J.-L., Jayan, B., Reynolds, J., Rezgui, Y., 2016. Web-based 3D urban decision support through intelligent and interoperable services, in: Macii, D., Solanas, A., Zhang, G. (Eds.), Proc. 2nd International Smart Cities Conference (ISC2). IEEE, Trento, Italy, pp. 1-4.

ICOM, 2016. What is the CIDOC CRM?, http://new.cidoc-crm.org/. (Accessed 11 August 2016)

IHBC, 2016. English housing stock age, https://www.designingbuildings.co.uk/wiki/English_housing_stock_age. (Accessed 15 August 2017)

Iman Zolanvari, S.M., Laefer, D.F., 2016. Slicing Method for curved façade and window extraction from point clouds. ISPRS Journal of Photogrammetry and Remote Sensing 119, 334-346.

Ippolito, A., Attenni, M., 2015. Measurements, geometries and proportions in Roman archaeological architecture, in: Grifa, C., Lubritto, C., Mercurio, M., Santoriello, A., Tomay, L. (Eds.), Proc. 1st International Conference on Metrology for Archaeology (MetroArcheo). MetroArcheo, Benevento, Italy, pp. 95-100.

Iqbal, Q., Aggarwal, J.K., 2002. Retrieval by classification of images containing large manmade objects using perceptual grouping. Pattern Recognition 35, (7), 1463-1479.

Isikdag, U., Zlatanova, S., 2009. Towards Defining a Framework for Automatic Generation of Buildings in CityGML Using Building Information Models, in: Lee, J., Zlatanova, S. (Eds.), Lecture Notes in Geoinformation and Cartography (LNGC) - 3D Geo-Information Sciences, first ed. Springer, Berlin Heidelberg, pp. 79-96.

ISTI - CNR, 2017. MeshLab, http://www.meshlab.net/. (Accessed 17 October 2017)

Jampani, V., Gadde, R., Gehler, P.V., 2015. Efficient Facade Segmentation Using Auto-context, in: Mortensen, E. (Ed.), Proc. 15th Winter Conference on Applications of Computer Vision (WACV). IEEE, Waikoloa Beach, USA, pp. 1038-1045.

JEITA, 2017. Exchangeable image file format for digital still cameras : Exif Version 2.31, http://www.jeita.or.jp/cgi-bin/standard_e/list.cgi?cateid=1&subcateid=4. (Accessed 19 November 2017)

Jiang, Z., Messner, J.I., Dubler, C.R., 2017. Defining a Taxonomy for Virtual 3D City Model Use Cases with a Focus on Facility Asset Management—A Virtual Campus Case Study, in: Lin, K.-Y., El-Gohary, N., Tang, P. (Eds.), Computing in Civil Engineering 2017 : Information Modeling and Data Analytics, first ed. American Society of Civil Engineers, Reston, pp. 43-49.

Joanneum, 2004a. TSG-20, http://dib.joanneum.at/cape/TSG-20/. (Accessed 15 July 2016)

Joanneum, 2004b. TSG-60, http://dib.joanneum.at/cape/TSG-60/. (Accessed 15 July 2016)

Johansson, B., Kahl, F., 2002. Detecting Windows in City Scenes, in: Lee, S.-W., Verri, A. (Eds.), Lecture Notes in Computer Science (LNCS) - Pattern Recognition with Support Vector Machines - First International Workshop, SVM 2002 Niagara Falls, Canada, August 10, 2002 - Proceedings, first ed. Springer, Berlin Heidelberg, pp. 388-396.

Jones, C.B., Rosin, P.L., Slade, J., 2014. Semantic and geometric enrichment of 3D geo-spatial models with captioned photos and labelled illustrations, in: Belz, A., Moens, M.-F., Smeaton, A.F. (Eds.), Proc. 25th International Conference on Computational Linguistics (COLING) - 3rd Workshop on Vision and Language (VL). ACL, Dublin, Ireland, pp. 62-67.

Jones, M.W., 2003. Principles of Roman Architecture, second ed. Yale University Press, Singapore.

JPEG, 2017. Overview of JPEG, https://jpeg.org/jpeg/. (Accessed 19 November 2017)

Kaden, R., Kolbe, T.H., 2013. City-Wide Total Energy Demand Estimation of Buildings Using Semantic 3D City Models and Statistical Data. ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences II-2/W1, (1), 163-171.

Kaehler, A., Bradski, G., 2016. Learning OpenCV 3: Computer Vision in C++ with the OpenCV Library, first ed. O'Reilly Media, Sebastopol.

Kamel Boulos, M.N., Lu, Z., Guerrero, P., Jennett, C., Steed, A., 2017. From urban planning and emergency training to Pokémon Go: applications of virtual reality GIS (VRGIS) and augmented reality GIS (ARGIS) in personal, public and environmental health. International Journal of Health Geographics 1-16, (1), 7.

Kang, H., Lee, S., Chui, C.K., 2007. Coherent line drawing, in: Agrawala, M., Deussen, O. (Eds.), Proc. 5th International Symposium on Non-photorealistic Animation and Rendering (NPAR). ACM, San Diego, USA, pp. 43-50.

Kang, T.W., Hong, C.H., 2017. IFC-CityGML LOD mapping automation using multiprocessing-based screen-buffer scanning including mapping rule. KSCE Journal of Civil Engineering, 1-11.

Kardinal Jusuf, S., Mousseau, B., Godfroid, G., Soh Jin Hui, V., 2017. Integrated modeling of CityGML and IFC for city/neighborhood development for urban microclimates analysis. Energy Procedia 122, 145-150.

Kersten, T.P., Tschirschwitz, F., Deggim, S., 2017. Development of a Virtual Museum Including a 4D Presentation. International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences XLII-2/W3, 361-367.

Khan, N., McCane, B., Mills, S., 2015. Better than SIFT? Machine Vision and Applications 26, (6), 819–836.

Khronos Group, 2008. COLLADA – Digital Asset Schema Release 1.5.0 - Specification, Report COLLADA 1.5.0 Specification, Khronos Group, https://www.khronos.org/files/collada_spec_1_5.pdf. (Accessed 28 January 2018)

Kim, H., Kim, K., Kim, H., 2016a. Data-driven scene parsing method for recognizing construction site objects in the whole image. Automation in Construction 71 Part 2, 271-282.

Kim, H., Shen, Z., Kim, I., Kim, K., Stumpf, A., Yu, J., 2016b. BIM IFC information mapping to building energy analysis (BEA) model with manually extended material information. Automation in Construction 68, 183-193.

King's College London, 2016. The Corpus of Romanesque Sculpture in Britain & Ireland, http://www.crsbi.ac.uk/. (Accessed 23 August 2016)

Kiourt, C., Koutsoudis, A., Pavlidis, G., 2016. DynaMus: A fully dynamic 3D virtual museum framework. Journal of Cultural Heritage 22, 984-991.

Koch, T., Korner, M., Fraundorfer, F., 2016. Automatic Alignment of Indoor and Outdoor Building Models Using 3D Line Segments, in: Mortensen, E., Fidler, S. (Eds.), Proc. 29th Conference on Computer Vision and Pattern Recognition Workshops (CVPRW). IEEE, Las Vegas, USA, pp. 10-18.

Koeva, M., Luleva, M., Maldjanski, P., 2017. Integrating Spherical Panoramas and Maps for Visualization of Cultural Heritage Objects Using Virtual Reality Technology. Sensors 17, (4), 829-843.

Kolbe, T.H., 2009. Representing and Exchanging 3D City Models with CityGML, in: Lee, J., Zlatanova, S. (Eds.), Lecture Notes in Geoinformation and Cartography (LNGC) - 3D Geo-Information Sciences, first ed. Springer, Berlin Heidelberg, pp. 15-31.

Kolbe, T.H., Nagel, C., Stadler, A., 2009. CityGML–OGC Standard for Photogrammetry, in: Fritsch, D. (Ed.), Proc. 52nd Photogrammetric Week (PhoWos). University of Stuttgart, Stuttgart, Germany, pp. 265-277.

Korah, T., Rasmussen, C., 2008. Analysis of Building Textures for Reconstructing Partially Occluded Facades, in: Forsyth, D., Torr, P., Zisserman, A. (Eds.), Lecture Notes in Computer Science (LNCS) - Image Processing, Computer Vision, Pattern Recognition, and Graphics Computer Vision – ECCV 2008 - 10th European Conference on Computer Vision, Marseille, France, October 12-

18, 2008 - Proceedings, Part I, first ed. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 359-372.

Korč, F., Förstner, W., 2009. eTRIMS Image Database for interpreting images of man-made scenes, Report TR-IGG-P-2009-01, University of Bonn, http://www.ipb.uni-bonn.de/projects/etrims_db/. (Accessed 09 February 2017)

Korman, S., Reichman, D., Tsur, G., Avidan, S., 2017. Fast-Match: Fast Affine Template Matching. Int J Comput Vis 121, (1), 111-125.

Koutamanis, A., Mitossi, V., 1993. Computer vision in architectural design. Design Studies 14, (1), 40-57.

Kovesi, P., 2005a. RANSACFITFUNDMATRIX - fits fundamental matrix using RANSAC, http://www.peterkovesi.com/matlabfns/Robust/ransacfitfundmatrix.m. (Accessed 10 June 2014)

Kovesi, P., 2005b. testfund.m, http://www.peterkovesi.com/matlabfns/Robust/example/testfund.m. (Accessed 10 June 2014)

Kroon, D.-J., 2011. Fast/Robust Template Matching, http://uk.mathworks.com/matlabcentral/fileexchange/24925-fast-robust-template-matching. (Accessed 01 September 2014)

Kumar, K., Ledoux, H., Commandeur, T.J.F., Stoter, J.E., 2017. Modelling Urban Noise in CityGML ADE: Case of The Netherlands. ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences IV-4/W5, 73-81.

Ledoux, H., 2013. On the Validation of Solids Represented with the International Standards for Geographic Information. Computer-Aided Civil and Infrastructure Engineering 28, (9), 693-706.

Lee, S., Liu, Y., 2007. PSU Near-Regular Texture Database, http://vivid.cse.psu.edu/texturedb/gallery/. (Accessed 15 July 2016)

Lee, S., Maisonneuve, N., Crandall, D., Efros, A.A., Sivic, J., 2015. Linking Past to Present: Discovering Style in Two Centuries of Architecture, in: Veeraraghavan, A., Favaro, P., Pulli, K. (Eds.), Proc. 5th International Conference on Computational Photography (ICCP). IEEE, Houston, USA, pp. 1-10.

Leutenegger, S., Chli, M., Siegwart, R.Y., 2011. BRISK: Binary Robust invariant scalable keypoints, in: Grau, A., Rocha, A. (Eds.), Proc. 13th International Conference on Computer Vision (ICCV). IEEE, Barcelona, Spain, pp. 2548-2555.

Ley, A., Hänsch, R., Hellwich, O., 2017. Automatic Building Abstraction from Aerial Photogrammetry. ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences IV-2/W4, 243-250.

Ley, A., Hellwich, O., 2016. Depth Map Based Facade Abstraction from Noisy Multi-View Stereo Point Clouds, in: Rosenhahn, B., Andres, B. (Eds.), Lecture Notes in Computer Science (LNCS) - Pattern Recognition - 38th German Conference, GCPR 2016, Hannover, Germany, September 12-15, 2016 - Proceedings, first ed. Springer International Publishing, Hannover, pp. 155-165.

Li, H., 2017. How to use regularization to prevent model overfitting, https://blogs.sas.com/content/subconsciousmusings/2017/07/06/how-to-use-regularization-to-prevent-model-overfitting/. (Accessed 05 January 2018)

Li, J., Fan, X., 2014. Outdoor augmented reality tracking using 3D city models and game engine, in: Nan, J., Lin, S. (Eds.), Proc. 7th International Congress on Image and Signal Processing (CISP). IEEE, Dalian, China, pp. 104-108.

Li, Y., Li, Z., 2017. A Multi-View Stereo Algorithm Based on Homogeneous Direct Spatial Expansion with Improved Reconstruction Accuracy and Completeness. Applied Sciences 7, (5), 446-466.

Li, Y., Wang, S., Tian, Q., Ding, X., 2015. A survey of recent advances in visual feature detection. Neurocomputing 149, 736-751.

Li, Y., Zheng, Q., Sharf, A., Cohen-Or, D., Chen, B., Mitra, N.J., 2011. 2D-3D fusion for layer decomposition of urban facades, in: Grau, A., Rocha, A. (Eds.), Proc. 13th International Conference on Computer Vision (ICCV). IEEE, Barcelona, Spain, pp. 882-889.

Li, Z., Zhang, L., Mathiopoulos, P.T., Liu, F., Zhang, L., Li, S., Liu, H., 2017. A hierarchical methodology for urban facade parsing from TLS point clouds. ISPRS Journal of Photogrammetry and Remote Sensing 123, 75-93.

Lia, B., Chenga, K., Yub, Z., 2016. Histogram of Oriented Gradient based Gist Feature for Building Recognition. Computational Intelligence and Neuroscience 2016, 1687-5265.

Lian, Y., Shen, X., Hu, Y., 2017. Detecting and inferring repetitive elements with accurate locations and shapes from façades. The Visual Computer, 16.

Liang, J., Gong, J., 2017. A Sparse Voxel Octree-Based Framework for Computing Solar Radiation Using 3D City Models. ISPRS International Journal of Geo-Information 6, (4), 106-120.

Liao, H., Dong, W., 2017. An Exploratory Study Investigating Gender Effects on Using 3D Maps for Spatial Orientation in Wayfinding. ISPRS International Journal of Geo-Information 6, (3), 60-79.

Lilis, G.N., Giannakis, G.I., Katsigarakis, K., Costa, G., Sicilia, A., Garcia-Fuentes, M.A., Rovas, D.V., 2016. Simulation model generation combining IFC and CityGML data, in: Christodolou, S., Scherer, R.J. (Eds.), eWork and eBusiness in Architecture, Engineering and Construction: ECPPM, first ed. CRC Press, Croydon, pp. 215-224.

Lin, Y., Kong, S., Wang, D., Zhuang, Y., 2014. Saliency Detection within a Deep Convolutional Architecture, in: Stracuzzi, D., Gunning, D. (Eds.), Proc. 28th Conference on Artificial Intelligence (IAAI) - Cognitive Computing for Augmented Human Intelligence Workshop. AAAI, Québec City, Canada, pp. 31-37.

Lin, Y.C., 2017. Application of Integration of HBIM and VR Technology to 3D Immersive Digital Management—Take Han Type Traditional Architecture as an Example. International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences XLII-2/W5, 443-446.

Lingua, A., Noardo, F., Spanò, A., Sanna, S., Matrone, F., 2017. 3D Model Generation using Oblique Images Acquired by UAV. International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences XLII-4/W2, 107-115.

Liptak, B.G., 2005. Instrument Engineers' Handbook, Volume Two: Process Control and Optimization, fourth ed. CRC Press, Boca Raton.

Liu, H., Zhang, J., Zhu, J., Hoi, S.C., 2017. DeepFacade: A Deep Learning Approach to Facade Parsing, in: Sierra, C. (Ed.), Proc. 26th International Joint Conference on Artificial Intelligence (IJCAI). ijcai.org, Melbourne, Australia, pp. 2301-2307.

Llamas, J., M. Lerones, P., Medina, R., Zalama, E., Gómez-García-Bermejo, J., 2017. Classification of Architectural Heritage Images Using Deep Learning Techniques. Applied Sciences 7, (10), 992.

Loch-Dehbi, S., Plümer, L., 2015. Predicting building façade structures with multilinear Gaussian graphical models based on few observations. Computers, Environment and Urban Systems 54, 68-81.

Louw, H., 2015. The Development of the Window, in: Tutton, M., Hirst, E., Pearce, J. (Eds.), Windows: History, Repair and Conservation, first ed. Taylor & Francis, Abingdon, pp. 8-112.

Louw, H.J., 1983. The Origin of the Sash-Window. Architectural History 26, 49-150.

Lowe, D., 2005. Demo Software: SIFT Keypoint Detector, http://www.cs.ubc.ca/~lowe/keypoints/. (Accessed 02 June 2014)

Lowe, D.G., 2004. Distinctive Image Features from Scale-Invariant Keypoints. Int J Comput Vis 60, (2), 91-110.

Löwner, M.-O., Gröger, G., 2017. Das neue LoD Konzept für CityGML 3.0, in: Bill, R., Golnik, A. (Eds.), Proc. 13th GeoForum MV. GeoMV eV, Warnemünde, Germany, pp. 1-8.

Löwner, M.O., Gröger, G., Benner, J., Biljecki, F., Nagel, C., 2016. Proposal for a New LOD and Multi-Representation Concept for CityGML. ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences IV-2/W1, 3-12.

Macák, J., Drbohlav, O., 2011. Hierarchical shape model for windows detection, in: Mayer, H. (Ed.), Proc. 35th Austrian Association for Pattern Recognition Workshop. OAGM / AAPR, Graz, Austria, pp. 1-8.

Maltese, S., Fradegrada, G., Moretti, N., Dejaco, M.C., Cecconi, F.R., 2016. GIS Application in Urban Districts Mainteinance, in: Tadeu, A., Abrantes, V., Ural, D. (Eds.), Proc. 41st World Congress on Housing - Sustainability and Innovation for the Future. IAHS, Albufeira, Portugal, pp. 1-10.

Mann, J., 2016. Find length of one (s, t) texture coordinate unit in world space units on a plane, https://stackoverflow.com/a/40367081. (Accessed 07 November 2016)

Manzoni, R., 2012. how to calculate area of 3D triangle?, https://math.stackexchange.com/a/128999. (Accessed 28 June 2014)

Marks, R., 2006. Stained Glass in England During the Middle Ages, first ed. Taylor & Francis, Abingdon.

Martínez-Rubio, A., Sanz-Adan, F., Santamaría-Peña, J., Martínez, A., 2016. Evaluating solar irradiance over facades in high building cities, based on LiDAR technology. Applied Energy 183, 133-147.

Martinović, A., Knopp, J., Riemenschneider, H., Van Gool, L., 2015. 3D all the way: Semantic segmentation of urban scenes from start to end in 3D, in: Mortensen, E., Fidler, S. (Eds.), Proc. 20th Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, Boston, USA, pp. 4456-4465.

Mathias, M., Martinović, A., Van Gool, L., 2016. ATLAS: A Three-Layered Approach to Facade Parsing. Int J Comput Vis 118, (1), 22-48.

MathWorks, 2016a. Applying Supervised Learning, in: MathWorks (Ed.), Machine Learning with MATLAB, first ed. MathWorks, Natick, pp. 1-20.

MathWorks, 2016b. Applying Unsupervised Learning, in: MathWorks (Ed.), Machine Learning with MATLAB, first ed. MathWorks, Natick, pp. 1-15.

MathWorks, 2016c. Getting Started with Machine Learning, in: MathWorks (Ed.), Machine Learning with MATLAB, first ed. MathWorks, Natick, pp. 1-16.

MathWorks, 2016d. Introducing Machine Learning, in: MathWorks (Ed.), Machine Learning with MATLAB, first ed. MathWorks, Natick, pp. 1-11.

MathWorks, 2016e. Machine Learning Challenges: Choosing the Best Classification Model and Avoiding Overfitting, Report 92974v00 03/16, MathWorks, https://uk.mathworks.com/content/dam/mathworks/tag-team/Objects/m/87371_92974v00_Machine_Learning_Whitepaper.pdf. (Accessed 27 October 2017)

MathWorks, 2017a. extractHOGFeatures - 'UseSignedOrientation' — Selection of orientation values, https://uk.mathworks.com/help/vision/ref/extracthogfeatures.html#UseSignedOrientation. (Accessed 26 October 2017)

MathWorks, 2017b. MATLAB, https://uk.mathworks.com/products/matlab.html. (Accessed 19 November 2017)

Mayer, H., Reznik, S., 2005. Building facade interpretation from image sequences. International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences XXXVI-3/W24, 55-60.

McGlinn, K., Debruyne, C., McNerney, L., O'Sullivan, D., in press. Integrating Ireland's Geospatial Information to Provide Authoritative Building Information Models, in: de Boer, V. (Ed.), Proc. 13th International Conference on Semantic Systems (SEMANTiCS). SEMANTiCS, Amsterdam, The Netherlands, pp. 57-64.

McKercher, B., Du Cros, H., 2002. Cultural tourism: The partnership between tourism and cultural heritage management, second ed. Routledge, Binghamton.

Megginson, D., 2004. About SAX, http://sax.sourceforge.net/. (Accessed 27 October 2017)

Meixner, P., Leberl, F., Brédif, M., 2011. Interpretation of 2D and 3D building details on facades and roofs. International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences XXXVIII-3/W22, 137-142.

Menberg, K., Heo, Y., Choudhary, R., 2016. Sensitivity analysis methods for building energy models: Comparing computational costs and extractable information. Energy and Buildings 133, 433-445.

Mezzino, D., Pei, C.W., Santana-Quintero, M.E., 2016. Interpretation of Sensor-Based 3D Documentation, in: Münster, S., Pfarr-Harfst, M., Kuroczyński, P., Ioannides, M. (Eds.), 3D Research Challenges in Cultural Heritage II: How to Manage Data and Knowledge Related to Interpretative Digital 3D Reconstructions of Cultural Heritage, first ed. Springer International Publishing, Cham, pp. 119-135.

Morago, B., Bui, G., Duan, Y., 2016. 2D Matching Using Repetitive and Salient Features in Architectural Images. IEEE Transactions on Image Processing 25, (10), 4888-4899.

Mouseprice, 2017. Mouseprice, https://www.mouseprice.com/. (Accessed 14 November 2017)

Müller, P., Zeng, G., Van Gool, L., 2015. Computer System and Method for Generating a 3D Geometric Model, US Patent 20,150,324,496, United States, http://www.freepatentsonline.com/20150324496.pdf. (Accessed 01 December 2015)

Müller, P., Zeng, G., Wonka, P., Van Gool, L., 2007. Image-based Procedural Modeling of Facades. ACM Transactions on Graphics (TOG) - Proceedings of ACM SIGGRAPH 2007 1-26, (3), 10.

Münster, S., Kröber, C., Weller, H., Prechtel, N., 2016. Researching Knowledge Concerns in Virtual Historical Architecture, in: Ioannides, M., Fink, E., Moropoulou, A., Hagedorn-Saupe, M., Fresa, A., Liestøl, G., Rajcic, V., Grussenmeyer, P. (Eds.), Digital Heritage - Progress in Cultural Heritage - Documentation, Preservation, and Protection - 6th International Conference, EuroMed 2016, Nicosia, Cyprus, October 31 – November 5, 2016 - Proceedings, Part I, first ed. Springer International Publishing, Cham, pp. 362-374.

Münster, S., Kröber, C., Weller, H., Prechtel, N., 2017. Virtual Reconstruction of Historical Architecture as Media for Knowledge Representation, in: Ioannides, M., Magnenat-Thalmann, N., Papagiannakis, G. (Eds.), Mixed Reality and Gamification for Cultural Heritage, 1st ed. Springer International Publishing, Cham, pp. 313-330.

Murphy, E., King, E.A., Rice, H.J., 2009a. Estimating human exposure to transport noise in central Dublin, Ireland. Environment International 35, (2), 298-302.

Murphy, M., Chenaux, A., Keenaghan, G., Gibson, V., Butler, J., Pybusr, C., 2017. Armagh Observatory – Historic Building Information Modelling for Virtual Learning in Building Conservation. International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences XLII-2/W5, 531-538.

Murphy, M., McGovern, E., Pavia, S., 2009b. Historic building information modelling (HBIM). Structural Survey 27, (4), 311-327.

Murtiyoso, A., Koehl, M., Grussenmeyer, P., Freville, T., 2017. Acquisition and Processing Protocols for UAV Images: 3D Modeling of Historical Buildings using Photogrammetry. ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences IV-2/W2, 163-170.

Musialski, P., Wonka, P., Aliaga, D.G., Wimmer, M., van Gool, L., Purgathofer, W., 2013. A Survey of Urban Reconstruction. Computer Graphics Forum 32, (6), 146-177.

Napolitano, R.K., Scherer, G., Glisic, B., in press. Virtual tours and informational modeling for conservation of cultural heritage sites. Journal of Cultural Heritage PP, 1-1.

Neuhausen, M., Koch, C., König, M., 2016. Image-based window detection: an overview, in: Borrmann, A., de Wilde, P., Hartmann, T., Smith, I., Li, H., O'Brien, W., Suter, G., Geyer, P. (Eds.), Proc. 23rd International Workshop of the European Group for Intelligent Computing in Engineering (EG-ICE). EG-ICE, Krakow, Poland, pp. 217-225.

Nguyen-Gia, T.-A., Dao, M.-S., Mai-Van, C., 2017. A Comparative Survey of 3D GIS Models, in: Anh-Hoang, T., Dang, T.H. (Eds.), Proc. 4th Conference on Information and Computer Science (NICS). NAFOSTED, Hanoi, Vietnam, pp. 126-131.

Nguyen, S.H., Yao, Z., Kolbe, T.H., 2017. Spatio-Semantic Comparison of Large 3D City Models in CityGML Using a Graph Database. ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences IV-4/W5, 99-106.

Ni, H., Lin, X.G., Zhang, J.X., 2017. Applications of 3D-Edge Detection for ALS Point Cloud. International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences XLII-2/W7, 277-283.

Niemeyer, J., Rottensteiner, F., Soergel, U., 2014. Contextual classification of LiDAR data and building object detection in urban areas. ISPRS Journal of Photogrammetry and Remote Sensing 87, 152-165.

Obeso, A.M., Benois-Pineau, J., Acosta, A.Á.R., Vázquez, M.S.G., 2016a. Architectural style classification of Mexican historical buildings using deep convolutional neural networks and sparse features. ELECTIM 26, (1), 011016.

Obeso, A.M., Reyes, L.M.A., Rodriguez, M.L., Cruz, M.H.M., Vázquez, M.S.G., Benois-Pineau, J., Fuentes, L.M.Z., Martinez, E.C., Secundino, J.A.F., Martinez, J.L.R., 2016b. Image annotation for Mexican buildings database, in: Iftekharuddin, K.M., Awwal, A.A.S., Vázquez, M.S.G. (Eds.), Proc. 10th Conference on Optics and Photonics for Information Processing SPIE, San Diego, USA, pp. 99700Y-99700Y-99708.

Ok, A.O., Wegner, J.D., Heipke, C., Rottensteiner, F., Soergel, U., Toprak, V., 2012. Matching of straight line segments from aerial stereo images of urban areas. ISPRS Journal of Photogrammetry and Remote Sensing 74, 133-152.

old-maps.co.uk, 2017. old-maps.co.uk, https://www.old-maps.co.uk/. (Accessed 14 November 2017)

Oreni, D., Brumana, R., Georgopoulos, A., Cuca, B., 2014. HBIM Library Objects for Conservation and Management of Built Heritage. International Journal of Heritage in the Digital Era 3, (2), 321-334.

Orlowski, A., 2017. UK's map maker Ordnance Survey plays with robo roof detector, https://www.theregister.co.uk/2017/12/14/ordnance_survey_ml_experiments/. (Accessed 15 April 2018)

Özyeşil, O., Voroninski, V., Basri, R., Singer, A., 2017. A survey of structure from motion. Acta Numerica 26, 305-364.

Pan, S.J., 2015. Transfer Learning, in: Aggarwal, C.C. (Ed.), Data Classification: Algorithms and Applications, 1st ed. CRC Press, Boca Raton, pp. 537-570.

Pears, N., Liu, Y., Bunting, P., 2012. 3D Imaging, Analysis and Applications, first ed. Springer, London.

Pevsner Architectural Guides, 2013. Looking at Buildings - Timeline, http://www.lookingatbuildings.org.uk/timeline.html. (Accessed 06 September 2016)

Pevsner, N., 2001. Herefordshire, ninth ed. Penguin Books, London.

Pevsner, N., Harris, J., Antram, N., 1989. Lincolnshire, 2nd ed. Yale University Press, London.

Pevsner, N., Sambrook, J., 2010. Pevsner's Architectural Glossary, 1st ed. Yale University Press, London.

Pickles, D., McCaig, I., Wood, C., 2017. Traditional Windows - Their Care, Repair and Upgrading, Report HEAG039, Historic England, https://content.historicengland.org.uk/images-books/publications/traditional-windows-care-repair-upgrading/heag039-traditional-windows-revfeb17.pdf/. (Accessed

Recky, M., Leberl, F., 2010. Windows Detection Using K-means in CIE-Lab Color Space, in: Mortensen, E.N., Yang, M.-H. (Eds.), Proc. 22nd Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, San Francisco, USA, pp. 356-359.

Remondino, F., Nocerino, E., Toschi, I., Menna, F., 2017. A Critical Review of Automated Photogrammetric Processing of Large Datasets. International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences XLII-2/W5, 591-599.

Reznik, S., Mayer, H., 2007. Implicit shape models, model selection, and plane sweeping for 3D facade interpretation. International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences XXXVI-3/W49A, 173-178.

RIBA, 2016. National Building Specification (NBS) National BIM Report 2015, Report RIBA, https://www.thenbs.com/~/media/files/pdf/nbs-national-bim-report-2015.pdf?la=en. (Accessed 19 April 2016)

RIBA, 2017. RIBApix, http://www.ribapix.com. (Accessed 11 September 2014)

Riemenschneider, H., Bódis-Szomorú, A., Weissenberg, J., Van Gool, L., 2014. Learning Where to Classify in Multi-view Semantic Segmentation, in: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (Eds.), Computer Vision – ECCV 2014:

13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V. Springer International Publishing, Cham, pp. 516-532.

Riemenschneider, H., Krispel, U., Thaller, W., Donoser, M., Havemann, S., Fellner, D., Bischof, H., 2012. Irregular lattices for complex shape grammar facade parsing, in: Mortensen, E.N. (Ed.), Proc. 24th Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, Providence, pp. 1640-1647.

Ripperda, N., Brenner, C., 2009. Application of a formal grammar to facade reconstruction in semiautomatic and automatic environments, in: Haunert, J.-H., Kieler, B., Milde, J. (Eds.), Proc. 12th Conference on Geographic Information Science. AGILE, Hannover, Germany, pp. 1-12.

Roberts, D., 2009. Fifties Neo-Vernacular, http://www.msa.mmu.ac.uk/continuity/index.php/2009/06/14/fifties-neo-vernacular/. (Accessed 05 October 2017)

Robles-Ortega, M.D., Ortega, L., Feito, F.R., 2017. Efficient Visibility Determination in Urban Scenes Considering Terrain Information. ACM Transactions on Spatial Algorithms and Systems (TSAS) 3, (3), 1-24.

Romero Rodríguez, L., Duminil, E., Sánchez Ramos, J., Eicker, U., 2017. Assessment of the photovoltaic potential at urban level based on 3D city models: A case study and new methodological approach. Solar Energy 146, 264-275.

Rook, M., Biljecki, F., Diakité, A.A., 2016. Towards Automatic Semantic Labelling of 3D City Models. ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences IV-2/W1, 23-30.

Ross, L., Bolling, J., Döllner, J., Kleinschmit, B., 2009. Enhancing 3D city models with heterogeneous spatial information: Towards 3D land information systems, in: Sester, M., Bernard, L., Paelke, V. (Eds.), Lecture Notes in Geoinformation and Cartography (LNGC) - Advances in GIScience - Proceedings of the 12th AGILE Conference, first ed. Springer, Berlin Heidelberg, pp. 113-133.

Rottensteiner, F., Sohn, G., Gerke, M., Wegner, J.D., 2013. ISPRS Test Project on Urban Classification and 3D Building Reconstruction, Report International Society for Photogrammetry and Remote Sensing (ISPRS), http://www2.isprs.org/tl_files/isprs/wg34/docs/ComplexScenes_revision_v4.pdf. (Accessed 09 February 2017)

Rousseeuw, P.J., 1987. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. Journal of Computational and Applied Mathematics 20, 53-65.

Rovira y Rabassa, A., 1897. Estereotomía de la Piedra, first ed. Librería y Estampería Artística, Barcelona.

Russell, B.C., Martin-Brualla, R., Butler, D.J., Seitz, S.M., Zettlemoyer, L., 2013. 3D Wikipedia: using online text to automatically label and navigate reconstructed geometry. ACM Transactions on Graphics (TOG) 32, (6), 1-10.

Safe Software, 2015. BIM to GIS (Advanced) | IFC LOD 200 to LOD 3 CityGML https://knowledge.safe.com/articles/1024/bim-to-gis-advanced-ifc-lod-200-to-lod-3-citygml.html. (Accessed 02 October 2017)

Safe Software, 2017. Convert, transform, and integrate data with FME Desktop, http://www.safe.com/fme/format-search/. (Accessed 20 June 2012)

Sargent, I., Holland, D., Harding, J., 2015. The Building Blocks of User-Focused 3D City Models. ISPRS International Journal of Geo-Information 4, (4), 2890-2904.

Sattler, T., Torii, A., Sivic, J., Pollefeys, M., Taira, H., Okutomi, M., Pajdla, T., 2017. Are Large-Scale 3D Models Really Necessary for Accurate Visual Localization?, in: Mortensen, E. (Ed.), Proc. 30th Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, Honolulu, USA, pp. 6175-6184.

Schmittwilken, J., Plümer, L., 2010. Model-based reconstruction and classification of façade parts in 3D point clouds. International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences XXXVIII Part 3A, 269-274.

Sebastiani, F., 2002. Machine learning in automated text categorization. ACM Computing Surveys (CSUR) 34, (1), 1-47.

Seifert, N., Mühlhaus, M., Petzold, F., 2016. A Parametric 3D City Model: Basis for Decision Support in Inner-City Development, in: Yabuki, N. (Ed.), Proc. 16th International Conference on Computing in Civil and Building Engineering (ICCCBE). Osaka University, Osaka, Japan, pp. 1-8.

Seitz, S.M., Curless, B., Diebel, J., Scharstein, D., Szelisk, R., 2006. A Comparison and Evaluation of Multi-View Stereo Reconstruction Algorithms, in: Kakadiaris, I.A. (Ed.), Proc. 18th Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, New York, USA, pp. 519-528.

Seo, D., Kang , H.-D., Hernández , D.C., Jo, K.-H., 2016. Building facade detection using geometric planar constraints, in: Hu, H., Zhang, D. (Eds.), Proc. 9th International Conference on Human System Interactions (HSI). IEEE, Portsmouth, United Kingdom, pp. 393-396.

Shalunts, G., Haxhimusa, Y., Sablatnig, R., 2011. Architectural Style Classification of Building Facade Windows, in: Bebis, G., Boyle, R., Parvin, B., Koracin, D., Wang, S., Kyungnam, K., Benes, B., Moreland, K., Borst, C., DiVerdi, S., Yi-Jen, C., Ming, J. (Eds.), Lecture Notes in Computer Science (LNCS) - Advances in Visual Computing - 7th International Symposium, ISVC 2011, Las Vegas, NV, USA, September 26-28, 2011 - Proceedings, Part II, first ed. Springer, Berlin Heidelberg, pp. 280-289.

Shao, H., Svoboda, T., Van Gool, L., 2003. ZuBuD-Zurich buildings database for image based recognition, Report Technical Report No.260, Swiss Federal Institute of Technology, http://www.vision.ee.ethz.ch/datasets/downloads/report-db.ps. (Accessed 15 July 2016)

Sharpe, G.R., 2011. Historic English Churches: A Guide to Their Construction, Design and Features, first ed. I. B. Tauris, London New York.

Shojaei, D., Olfat, H., Rajabifard, A., Darvill, A., Briffa, M., 2016. Assessment of the Australian digital cadastre protocol (ePlan) in terms of supporting 3D building subdivisions. Land Use Policy 56, 112-124.

Shotton, J., Winn, J., Rother, C., Criminisi, A., 2006. TextonBoost: Joint Appearance, Shape and Context Modeling for Multi-class Object Recognition and Segmentation, in: Leonardis, A., Bischof, H., Pinz, A. (Eds.), Lecture Notes in Computer Science book series (LNCS) - European Conference on Computer Vision - ECCV 2006: Computer Vision – ECCV 2006. Springer, Berlin, Heidelberg, pp. 1-15.

Simon, I., Seitz, S.M., 2008. Scene Segmentation Using the Wisdom of Crowds, in: Forsyth, D., Torr, P., Zisserman, A. (Eds.), Lecture Notes in Computer Science (LNCS) - Computer Vision – ECCV 2008 - 10th European Conference on Computer Vision, Marseille, France, October 12-18, 2008 - Proceedings, Part II, first ed. Springer, Berlin, Heidelberg, pp. 541-553.

Simonyan, K., Zisserman, A., 2014. Very deep convolutional networks for large-scale image recognition. arXiv:1409.1556, [cs.CV].

Sinha, S.N., Steedly, D., Szeliski, R., Agrawala, M., Pollefeys, M., 2008. Interactive 3D architectural modeling from unordered photo collections. ACM Transactions on Graphics (TOG) - Proceedings of ACM SIGGRAPH Asia 2008 27, (5), 1-10.

Sivic, J., Russell, B.C., Efros, A.A., Zisserman, A., Freeman, W.T., 2005. Discovering objects and their location in images, in: Lin, S. (Ed.), Proc. 10th International Conference on Computer Vision (ICCV). IEEE, Beijing, China, pp. 370-377.

Slade, J., Buss, S., 2017. Texture map projection problem, https://forum.3dis.de/viewtopic.php?f=4&t=9. (Accessed 20 January 2017)

Slade, J., Jones, C.B., Rosin, P.L., 2015. Semantic and geometric enrichment of 3D geo-spatial building models with photo captions and illustration labels using template matching & SIFT, in: Malleson, N. (Ed.), Proc. 23rd GIS Research UK (GISRUK). University of Leeds, Leeds, UK, pp. 1-7.

Slade, J., Jones, C.B., Rosin, P.L., 2017. Automatic Semantic and Geometric Enrichment of CityGML Building Models using HOG-based Template Matching, in: Abdul-Rahman, A. (Ed.), Lecture Notes in Geoinformation and Cartography

(LNGC) - Advances in 3D Geoinformation, first ed. Springer International Publishing, Cham, pp. 357-372.

Smart, P.D., Quinn, J.A., Jones, C.B., 2011. City model enrichment. ISPRS Journal of Photogrammetry and Remote Sensing 66, (2), 223-234.

Snavely, N., Seitz, S.M., Szeliski, R., 2006. Photo tourism: exploring photo collections in 3D. ACM Transactions on Graphics (TOG) - Proceedings of ACM SIGGRAPH 2006 25, (3), 835-846.

Soler, F., Melero, F.J., Luzón, V.M., 2017. A complete 3D information system for cultural heritage documentation. Journal of Cultural Heritage 23, 49-57.

Sonka, M., Hlaváč, V., Boyle, R., 2014. Image Processing, Analysis, and Machine Vision, 4th ed. Cengage Learning, Boston.

Soon, K.H., Thompson, R., Khoo, V., 2014. Semantics-based fusion for CityGML and 3D LandXML, in: van Oosterom, P. (Ed.), Proc. 4th International Workshop on 3D Cadastres. FIG, Dubai, UAE, pp. 1-16.

Stadler, A., Kolbe, T.H., 2007. Spatio-semantic coherence in the integration of 3D city models. International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences XXXVI-2/C43, 1-8.

Stoter, J., Ploeger, H., Roes, R., Riet, E.v.d., Biljecki, F., Ledoux, H., Kok, D., Kim, S., 2017. Registration of Multi-Level Property Rights in 3D in The Netherlands: Two Cases and Next Steps in Further Implementation. ISPRS International Journal of Geo-Information 6, (6), 158-175.

Stoter, J., Ploeger, H., Roes, R., van der Riet, E., Biljecki, F., Ledoux, H., 2016a. First 3D Cadastral Registration of Multi-level Ownerships Rights in the Netherlands, in: van Oosterom, P. (Ed.), Proc. 5th International Workshop on 3D Cadastres. FIG, Athens, Greece, pp. 1-14.

Stoter, J., Vallet, B., Lithen, T., Pla, M., Wozniak, P., Kellenberger, T., Streilein, A., Ilves, R., Ledoux, H., 2016b. State-of-the-Art of 3D National Mapping in 2016. International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences XLI-B4, 653-660.

Streilein, A., Remondino, F., Pfeifer, N., Trollvik, J.A., Stoter, J., Crompvoets, J., Potůčková, M., 2016. EuroSDR – The Pan-European Network for Mapping Agencies & Academia. International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences XLI-B4, 661-668.

Strobbe, T., wyffels, F., Verstraeten, R., Meyer, R.D., Campenhout, J.V., 2016. Automatic architectural style detection using one-class support vector machines and graph kernels. Automation in Construction 69, 1-10.

Sugihara, K., Shen, Z., 2016. Automatic generation of 3D house models with solar photovoltaic generation for smart city, in: Pandey, J. (Ed.), Proc. 3rd

International Conference on Big Data and Smart City (ICBDSC). IEEE, Muscat, Oman, pp. 1-7.

Szeliski, R., 2010. Computer Vision: Algorithms and Applications, first ed. Springer, London.

Takahisa, K., Sun, Z., Micheletto, R., 2013. A Fast and Precise HOG-Adaboost Based Visual Support System Capable to Recognize Pedestrian and Estimate Their Distance, in: Petrosino, A., Maddalena, L., Pala, P. (Eds.), New Trends in Image Analysis and Processing - ICIAP 2013: ICIAP 2013 International Workshops, Naples, Italy, September 9-13, 2013. Proceedings, first ed. Springer, Berlin, Heidelberg, pp. 20-29.

Tashakkori, H., Rajabifard, A., Kalantari, M., 2016. Facilitating the 3D Indoor Search and Rescue Problem: an Overview of the Problem and an Ant Colony Solution Approach. ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences IV-2/W1, 233-240.

Teboul, O., 2010. Ecole Centrale Paris (ECP) Facades Database, http://vision.mas.ecp.fr/Personnel/teboul/data.php. (Accessed 09 February 2017)

Teboul, O., Kokkinos, I., Simon, L., Koutsourakis, P., Paragios, N., 2013. Parsing Facades with Shape Grammars and Reinforcement Learning. IEEE Transactions on Pattern Analysis and Machine Intelligence 35, (7), 1744-1756.

Teboul, O., Simon, L., Koutsourakis, P., Paragios, N., 2010. Segmentation of building facades using procedural shape priors, in: Mortensen, E.N., Yang, M.-H. (Eds.), Proc. 22nd Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, San Francisco, USA, pp. 3105-3112.

TGRMN, 2017. Bulk Rename Utility, http://www.bulkrenameutility.co.uk/Main_Intro.php. (Accessed 19 November 2017)

The Office of the Deputy Prime Minister, 2001. English House Condition Survey 2001, Report 15 August 2017, Queen’s Printer and Controller of Her Majesty’s Stationery Office,, http://doc.ukdataservice.ac.uk/doc/6102/mrdoc/pdf/6102main_report.pdf. (Accessed 15 August 2017)

Thomopoulos, S.C.A., Doulgerakis, A., Bessa, M., Dimitros, K., Farazis, G., Georgiou, E., Kanellos, T., Karafylli, C., Karafylli, M., Kyriazanos, D.M., Kountouriotis, V.I., Lampropoulos, V., Margonis, C., Maroglou, C., Motos, D., Papagianni, A., Paterakis, M., Skroumpelou, K., Thanos, G.K., Theodorou, I.-E., Thomopoulos, C.P., Tsimpiridis, P., Zacharakis, D., Zalonis, A., 2016. DICE: Digital Immersive Cultural Environment, in: Ioannides, M., Fink, E., Moropoulou, A., Hagedorn-Saupe, M., Fresa, A., Liestøl, G., Rajcic, V., Grussenmeyer, P. (Eds.), Digital

Heritage - Progress in Cultural Heritage - Documentation, Preservation, and Protection - 6th International Conference, EuroMed 2016, Nicosia, Cyprus, October 31 – November 5, 2016 - Proceedings, Part I, first ed. Springer International Publishing, Cham, pp. 758-777.

Thonat, T., Shechtman, E., Paris, S., Drettakis, G., 2016. Multi-View Inpainting for Image-Based Scene Editing and Rendering, in: Su, H. (Ed.), Proc. 4th International Conference on 3D Vision (3DV). IEEE, Stanford, USA, pp. 351-359.

TIG, 2010. Fix Reversed Face Materials, https://sketchucation.com/forums/viewtopic.php?t=30107. (Accessed 20 June 2014)

Tighe, J., Lazebnik, S., 2013. Finding Things: Image Parsing with Regions and Per-Exemplar Detectors, in: Mortensen, E., Fidler, S. (Eds.), 25th Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, Portland, USA, pp. 3001-3008.

Tournikiotis, P., 2001. The Historiography of Modern Architecture, first ed. MIT Press, Cambridge.

Trimble, 2014. Trimble 3D Warehouse, https://3dwarehouse.sketchup.com. (Accessed 10 July 2014)

Trimble, 2018. SketchUp, https://www.sketchup.com/. (Accessed 22 January 2018)

Tyleček, R., 2013. The CMP Facade Database, Report CTU-CMP-2012-24, Czech Technical University, ftp://cmp.felk.cvut.cz/pub/cmp/articles/tylecek/Tylecek-TR-2012-24.pdf. (Accessed 24 November 2016)

Tymms, S., Thompson, J.R., 1855. Handbook of Bury St. Edmund's, fifth ed. Oxford University, Oxford.

Vahe, 2014. Finding the missing vertex (x,y,z) of a rectangle whose other vertices are defined., https://math.stackexchange.com/a/678757. (Accessed 07 November 2016)

Van Gool, L., Zeng, G., Van den Borre, F., Müller, P., 2007. Towards mass-produced building models. International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences XXXVI-3/W49A, 209-220.

Van Oosterom, P.J.M., Vandysheva, N., Ivanov, A., Pakhomov, S., Spiering, B., Stoter, J.E., Zlatanova, S., 2012. Design of 3D cadastre model in the Russian Federation, https://www.geospatialworld.net/article/design-of-3d-cadastre-model-in-the-russian-federation/. (Accessed 19 September 2016)

Verhoeven, G.J., 2016. Mesh Is More—Using All Geometric Dimensions for the Archaeological Analysis and Interpretative Mapping of 3D Surfaces. Journal of Archaeological Method and Theory 24, (4), 999-1033.

Verykokou, S., Ioannidis, C., 2016. Automatic Rough Georeferencing of Multiview Oblique and Vertical Aerial Image Datasets of Urban Scenes. The Photogrammetric Record 31, (155), 281-303.

Verykokou, S., Ioannidis, C., Kontogianni, G., 2014. 3D Visualization via Augmented Reality: The Case of the Middle Stoa in the Ancient Agora of Athens, in: Ioannides, M., Magnenat-Thalmann, N., Fink, E., Žarnić, R., Yen, A.-Y., Quak, E. (Eds.), Digital Heritage. Progress in Cultural Heritage: Documentation, Preservation, and Protection: 5th International Conference, EuroMed 2014, Limassol, Cyprus, November 3-8, 2014. Proceedings, first ed. Springer International Publishing, Cham, pp. 279-289.

Vilgertshofer, S., Amann, J., Willenborg, B., Borrmann, A., Kolbe, T.H., 2017. Linking BIM and GIS Models in Infrastructure by Example of IFC and CityGML, in: Lin, K.-Y., El-Gohary, N., Tang, P. (Eds.), Computing in Civil Engineering 2017: Information Modeling and Data Analytics, first ed. ASCE, Reston, pp. 133-140.

virtualcitySYSTEMS, 2017. Products - Building Reconstruction - Media - Gallery - 3d_city_model_frankfurt_main_1024px_10, http://www.virtualcitysystems.de/en/products/buildingreconstruction#media. (Accessed 31 October 2017)

von Schwerin, J., Lyons, M., Loos, L., Billen, N., Auer, M., Zipf, A., 2016. Show Me the Data!: Structuring Archaeological Data to Deliver Interactive, Transparent 3D Reconstructions in a 3D WebGIS, in: Münster, S., Pfarr-Harfst, M., Kuroczyński, P., Ioannides, M. (Eds.), 3D Research Challenges in Cultural Heritage II: How to Manage Data and Knowledge Related to Interpretative Digital 3D Reconstructions of Cultural Heritage, first ed. Springer International Publishing, Cham, pp. 198-230.

W3C, 2003. Portable Network Graphics (PNG) Specification (Second Edition), https://www.w3.org/TR/2003/REC-PNG-20031110/. (Accessed 19 November 2017)

W3C, 2004. Document Object Model (DOM), https://www.w3.org/DOM/. (Accessed 27 October 2017)

Waechter, M., Beljan, M., Fuhrmann, S., Moehrle, N., Kopf, J., Goesele, M., 2017. Virtual Rephotography: Novel View Prediction Error for 3D Reconstruction. ACM Transactions on Graphics (TOG) 36, (1), 1-11.

Wagner, D., Wewetzer, M., Bogdahn, J., Alam, N., Pries, M., Coors, V., 2013. Geometric-Semantical Consistency Validation of CityGML Models, in: Pouliot, J., Daniel, S., Hubert, F., Zamyadi, A. (Eds.), Lecture Notes in Geoinformation and Cartography book series (LNGC) - Progress and New Trends in 3D Geoinformation Sciences, first ed. Springer, Berlin, Heidelberg, pp. 171-192.

Walker, D., 2015a. Better living through gas: Kensal House, https://rbkclocalstudies.wordpress.com/2015/12/03/better-living-through-gas-kensal-house/. (Accessed 07 November 2017)

Walker, D., 2015b. From Bauhaus to our house: Old Church Street 1936, https://rbkclocalstudies.wordpress.com/2013/11/14/from-bauhaus-to-our-house-old-church-street-1936/. (Accessed 07 November 2017)

Wang, C., Cho, Y.K., Kim, C., 2015. Automatic BIM component extraction from point clouds of existing buildings for sustainability applications. Automation in Construction 56, 1-13.

Wang, J., Xu, Y., Remil, O., Xie, X., Ye, N., Yi, C., Wei, M., 2016. Automatic Modeling of Urban Facades from Raw LiDAR Point Data. Computer Graphics Forum 35, (7), 269-278.

Waters, S., 2017a. RIBA: Classical / Classical Revival / Neo-Classical https://www.architecture.com/knowledge-and-resources/knowledge-landing-page/classical-classical-revival-neo-classical. (Accessed 03 October 2017)

Waters, S., 2017b. RIBA: Gothic / Gothic Revival / Neo-Gothic, https://www.architecture.com/knowledge-and-resources/knowledge-landing-page/gothic-gothic-revival-neo-gothic. (Accessed 13 February 2017)

Waters, S., 2017c. RIBA: Modernism, https://www.architecture.com/knowledge-and-resources/knowledge-landing-page/modernism. (Accessed 07 November 2017)

Watkin, D., 2005. A History of Western Architecture, fourth ed. Laurence King Publishing, London.

Welford Park, 2014. Welford Park - The History of Welford Park, http://www.welfordpark.co.uk/the-history. (Accessed 10 June 2015)

Wenzel, S., Förstner, W., 2016. Facade Interpretation using a Marked Point Process. ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences III-3, 363-370.

Wheeler, M., Whiteley, N., 1992. The Lamp of Memory: Ruskin, Tradition, and Architecture, first ed. Manchester University Press, Manchester.

Whiteside, A., 2009. Definition identifier URNs in OGC namespace, Report 07-092r1, Open Geospatial Consortium (OGC), https://portal.opengeospatial.org/files/?artifact_id=24045. (Accessed 08 May 2015)

Williams, K., 1999. Symmetry in Architecture. Visual Mathematics 1, (1), 1-10.

Witkin, A.P., 1984. Scale-space filtering: A new approach to multi-scale description, in: Elliot, D.F. (Ed.), Proc. International Conference on Acoustics, Speech, and Signal Processing (ICASSP). IEEE, San Diego, pp. 150-153.

Wolff, M., Collins, R.T., Liu, Y., 2016. Regularity-Driven Facade Matching Between Aerial and Street Views, in: Mortensen, E., Fidler, S. (Eds.), Proc. 29th Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, Las Vegas, USA, pp. 1591-1600.

Wong, K., Ellul, C., 2016. Using Geometry-based Metrics as part of Fitness-for-purpose Evaluations of 3D City Models. ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences IV-2/W1, 129-136.

Wong, K., Ellul, C., 2017. User Requirements Gathering for 3D Geographic Information in the United Kingdom. ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences IV-4/W5, 125-132.

Xiao, J., 2013. HOG-based Template Matching, http://3dvision.princeton.edu/code.html#templateMatching. (Accessed 10 January 2015)

Xie, L., Hu, H., Zhu, Q., Wu, B., Zhang, Y., 2017. Hierarchical Regularization of Polygons for Photogrammetric Point Clouds of Oblique Images. International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences XLII-1/W1, 35-40.

Xiong, Q., Zhu, Q., Du, Z., Zlatanova, S., Zhang, Y., Zhou, Y., Li, Y., 2016. Free multi-floor indoor space extraction from complex 3D building models. Earth Science Informatics 10, (1), 69-83.

Xu, L., Zeng, L., Duan, H., Sowah, N.L., 2014a. Saliency detection in complex scenes. EURASIP Journal on Image and Video Processing 2014, (1), 1-13.

Xu, Z., Tao, D., Zhang, Y., Wu, J., Tsoi, A., 2014b. Architectural Style Classification Using Multinomial Latent Logistic Regression, in: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (Eds.), Lecture Notes in Computer Science (LNCS) - Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014 - Proceedings, Part I, first ed. Springer, Cham, pp. 600-615.

Yamada, M., Chen, J., Chang, Y., 2018. Transfer Learning: Algorithms and Applications, 1st ed. Morgan Kaufmann, Burlington.

Yang, B., Dong, Z., Liang, F., Liu, Y., 2016. Automatic registration of large-scale urban scene point clouds based on semantic feature points. ISPRS Journal of Photogrammetry and Remote Sensing 113, 43-58.

Ying, S., Guo, R., Yang, J., He, B., Zhao, Z., Jin, F., 2017. 3D Space Shift from CityGML LoD3-Based Multiple Building Elements to a 3D Volumetric Object. ISPRS International Journal of Geo-Information 6, (1), 1-18.

Yu, T., Wang, R., 2016. Scene Parsing using Graph Matching on Street-view Data. Computer Vision and Image Understanding 145, 70-80.

Zhang, B., Song, Y., Guan, S., Zhang, Y., 2010. Historic chinese architectures image retrieval by SVM and pyramid histogram of oriented gradients features. International Journal of Soft Computing 5, (2), 19-28.

Zhang, H., Xu, K., Jiang, W., Lin, J., Cohen-Or, D., Chen, B., 2013a. Layered analysis of irregular facades via symmetry maximization. ACM Transactions on Graphics (TOG) 32, (4), 1-13.

Zhang, J., Yao, H., Jiang, W., Shen, X., 2015. Hierarchical Repetition Extraction for Building Façade Reconstruction from Oblique Aerial Images. International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences XL-4/W5, 183-187.

Zhang, Y., Xiao, J., Hays, J., Tan, P., 2013b. FrameBreak: Dramatic Image Extrapolation by Guided Shift-Maps, in: Mortensen, E., Fidler, S. (Eds.), Proc. 25th Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, Portland, USA, pp. 1171-1178.

Zheng, Z., Zhang, Y., Yan, L., 2014. Global and local exploitation for saliency using bag-of-words. IET Computer Vision 8, (4), 299-304.

Zhou, K., Gorte, B., Zlatanova, S., 2016. Exploring Regularities for Improving Façade Reconstruction from Point Clouds. International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences XLI-B5, 749-755.

Zhu, Q., Hu, M., Zhang, Y., Du, Z., 2009. Research and practice in three-dimensional city modeling. Geo-spatial Information Science 12, (1), 18-24.

Zubala, T., Sadurska, U., 2016. The use of Noise Maps & Planning Tools in the Management of an Acoustic Climate in an Urban Area. Journal of Ecological Engineering 17, (4), 209-217.

Appendix A Data Sources

NOTE: Each ‘Reference’ in this appendix follows approximate alphabetical order.

Reference	Source
BM_2_1	“3 Garland Court House” by ninjaartmonster
BM_2_2	“MK77NG Model” by RayJay
BM_2_3	“House at the center of england” by John K
BM_2_4	“15 Lowther Road” by 3dbrighton
BM_2_5	“Bishops Cleeve – Berwick Road Part 9” by James .L
BM_2_6	“Housing #34, Cam” by John
BM_2_7	“'The Kingshill', Dursley” by John
BM_2_8	“Fairfield Avenue” by Mclaren606
BM_2_9	“Meadow View” by Chris D.
BM_2_10	“House” by bella99_uk
BM_2_11	“71 Darby Road. Darby Road Project from earthHD” by earthHD
BM_2_12	“75 Darby Road. Darby Road Project from earthHD” by earthHD
BM_2_13_V	“93-109 Westward Road with 1,3,5&7 Hilly Orchard” by Tony Wilton

Table A.1: Sources of 3D Building Models – C20 Style
These models are licensed for public use by Trimble. Note: BM_2_13_V was used for validation and threshold tuning only, and not in any experiments.

3D Building Model Reference	Source of Texture Map Images used as Replacements
BM_2_3	5 texture map images replaced using images downloaded from Google *StreetView* on 15/09/2017
BM_2_7	3 texture map images replaced using images downloaded from Google *StreetView* on 15/09/2017

Table A.2: Sources of Texture Map Images Replaced in 3D Building Models – C20 Style
For the above 3D building models, texture map images were replaced and then edited manually, using the above images, in advance of processing.

Reference	Source
BM_GO_1	"Houses of Parliament" by Damo

Table A.3: Sources of 3D Building Models – Gothic (Revival)
These models are licensed for public use by Trimble.

Reference	Source
BM_GR_1	"Senate House in Cambridge" by Aldara
BM_GR_2	"Haigh Hall" by peteralwyn@walsh143@orangehome.co.uk
BM_GR_3	"Lytham Hall" by peteralwyn@walsh143@orangehome.co.uk
BM_GR_4	“Belton House” by Johan
BM_GR_5	"Osterley Park, Greater London" by Johan
BM_GR_6	"ickworth house rotunda" by jason
BM_GR_7	"Box House" by Damo
BM_GR_8_R	“Bristol Old Vic” by 3DBristol

Table A.4: Sources of 3D Building Models – Georgian-Regency Style
These models are licensed for public use by Trimble. Note that aside from Figure 7.10 in Chapter 7, BM_GR_8_R was not used. More detailed architectural styles are as follows. BM_GR_1: neo-classical style. BM_GR_3: Palladian style. BM_GR_4: Palladian style. Style sources: Historic England (2006, 2017); Pevsner *et al.* (1989); RIBA (2017). These styles fall within or cross-over with the Georgian style grouping according to Pevsner Architectural Guides (2013) and Pevsner and Sambrook (2010). See Chapter 2.

3D Building Model Reference	Source of Texture Map Images used as Replacements
BM_GR_6	“03s17b6” by OpenBuildings, is licensed for public use by OpenBuildings

Table A.5: Sources of Replacements Texture Map Images in 3D Building Models – Georgian-Regency Style
For the above 3D building models, texture map images were replaced and then edited manually, using the above images, in advance of processing.

Reference	Source
BM_OC_1_V	“Welford House, Berkshire” by VassilisP.
BM_OC_2_V	“Palace of Holyroodhouse. Edimburg. UK” by wysywyg
BM_OC_3_V	“Blenheim Palace” by Richard

Table A.6: Source of 3D Building Models – Other Classical Styles (used for Validation Only)
These 3D building models were only used for determining the thresholds for heuristic 2 – see Section 5.11.2. Construction dates and architectural styles for the corresponding buildings are as follows: BM_OC_1_V built c.1652, frontage remodelled in 1700 in possibly Baroque style (Watkin, 2005, p. 352; Welford Park, 2014); BM_OC_2_V built 1671-8 in combined Baroque and Renaissance styles (Historic Environment Scotland, 2017); and BM_OC_3_V built in 1706-29 in Baroque style (Historic England, 2017).

Reference	Source
BM_N_1	"Kilpeck Church" by Tom
BM_N_2	"St Michael All Angels Church, Moccas" by Tom
BM_N_3	"Church of St.Gregory, Heckingham" by Jonathan G
BM_N_4	"St Nicholas, Askham Bryan" by Geoffrey H.
BM_N_5	"Norman Tower, Bury St Edmunds, Suffolk, UK" by Pat_Spillane

Table A.7: Sources of 3D Building Models – Norman Style
These models are licensed for public use by Trimble.

3D Building Model Reference	Source of Texture Map Images used as Replacements
BM_N_1	"Kilpeck: Church of SS Mary and David (Herefordshire)" by Michael Day, is licensed under CC BY-NC 2.0
BM_N_2	"SO3543 Moccas Church" by Philip Halling is licensed under CC BY-SA 2.0
BM_N_4	**(1)** **"**SE5548 East wall, St Nicholas, Askham Bryan" by Rich Tea is licensed under CC BY-SA 2.0 **(2)** "St Nicholas' Church, Askham Bryan" by Richard Masters is licensed under CC BY-NC-SA 2.0
BM_N_5	"The Norman Tower and Gatehouse" by Tour Suffolk provides no copyright information on the website

Table A.8: Sources of Replacements Texture Map Images in 3D Building Models – Norman Style
For the above 3D building models, texture map images were replaced and then edited manually, using the above images, in advance of processing.

Reference	Source
CDN_GO_026	“Muchelney, Somerset” by Martin Beek is copyright
CDN_GO_062	“Langley Chapel” by Marios Hadjianastasis is copyright
CDN_GO_071	“3rd day in England 007” by Juli Elliot is copyright
CDN_N_025	“Norman doorway, Norwich Cathedral, Norwich, Norfolk, England” by Spencer Means is licensed under CC BY-SA 2.0
CDN_N_035	“Stewkley Church” by R~P~M is licensed under CC BY-NC-ND 2.0.
CDN_N_040	“ST9387 Door, Malmesbury Abbey” by Derek Harper is licensed under CC BY-SA 2.0
CDN_N_043	“SP6534 South Door, St Giles Parish Church” by David Hillas is licensed under CC BY-SA 2.0
CDN_N_045	“TR1041 Norman Priest's Door” by Julian P Guffogg is licensed under CC BY-SA 2.0
CDN_N_044	“SU8504 Romanesque door and window, Chichester Cathedral” by Julian P Guffogg is licensed under CC BY-SA 2.0
CDN_N_051	“TR1855 Norman south doorway at St Mary's Church, Patrixbourne” by Marathon is licensed under CC BY-SA 2.0
CDN_N_052	“SU5132 Red Door on the Church” by Bill Nicholls is licensed under CC BY-SA 2.0
CDW_GO_004	"Coughton Court" by Liz & Johnny Wesley Barker is licensed under CC BY-NC-ND 2.0.
CDW_GO_005	"5589" by Ben Abel is copyright
CDW_GO_013	"Tudor door and windows" by dreamstime is copyright
CDW_GO_016	“TM4287 Brick Doorway – Weston Church” by Ashley Dace is licensed under CC BY-SA 2.0
CDW_GO_018	"Coughton Court" by Beth Gibbons is copyright
CDW_GO_020	"Admiring the old graffiti on the back door to the Tudor House in Blue Anchor Lane" by Mike is copyright
CDW_GO_026	"Aldsworth-250 St Bartholomew North porch" by Guy Thornton is copyright
CDW_GO_027	"barn Doors" by Stu@ is copyright
CDW_GO_034	"Christ College" by nicholas_january is copyright
CDW_GO_036	"Church Door" by allyhook is licensed under CC BY-NC-ND 2.0.
CDW_GO_041	"Deanery tower (15th century) – Hadleigh, Suffolk" by ronin65 is copyright
CDW_GO_047	"DSC_0231.JPG" by ntlakesoutdoors.org.uk is copyright
CDW_GO_059	"Love this door!!" by beachrumpunch is copyright
CDW_GO_065	"Lavenham Front Door 2" by iRobbo (aka Boycotts) is copyright
CDW_GO_072	"Main gate at Hampton Court" by Specklett is copyright
CDW_GO_076	"Ornate façade, Oxford" by Monceau is copyright
CDW_GO_078	“Paycocke's Coggeshall” by Amanda Slater is licensed under CC BY-SA 2.0
CDW_GO_079	"https://www.flickr.com/photos/7343172@N08/2291842547/" by picturesofian is copyright
CDW_GO_082	"Speke Hall, Liverpool" by fredparkins is copyright
CDW_GO_095	"Tudor Gateway – Pilgrims Way, Detling, Kent, UK. Image" by Dragontree is copyright
CDW_GO_100	"Westminster Abbey Entrance" by Sue Martin is copyright

Table A.9: Sources of Clustering Templates used in Figures – Table 1 of 3
See also Table A.11 and Table A.10. Note: not all templates used are listed above, only those used in figures.

CDW_N_001	"Bromyard Church (St. Peter)" by Hugh Llewelyn is licensed under CC BY-SA 2.0.
CDW_N_007	"Peterborough Cathedral" by .Martin. is licensed under CC BY-ND 2.0
CDW_N_024	"southwell minster west portal" by damian entwistle is licensed under CC BY-NC 2.0.
CDW_N_028	"The Norman south doorway, the Church of St Mary, Thornham Parva, Suffolk" by Spencer Means is licensed under CC BY-SA 2.0.
CDW_N_030	"Through the round window" by bishib70 is licensed under CC BY-NC-ND 2.0
CDW_N_032	"ST4226 Huish Episcopi: south door, St Mary's church" by Martin Bodman is licensed under CC BY-SA 2.0.
CDW_N_049	"Burton Dassett, Warwickshire" by Martin Beek is copyright
CDW_N_058	"Bradbourne, Derbyshire, All Saints" by John Hawes is copyright
CDW_N_062	Since downloading the image has been removed from the internet. Previously found here.
CDW_N_067	"Heckingham Norman Door" by Cam Self is copyright
CDW_N_069	"Masham Norman door" by Annie-Sue Jyelra is copyright
CDW_N_071	"Norman Door at Adel Church in Leeds" by David is copyright
CDW_N_073	"Norman Door at Hales Church in Norfolk" by David is copyright
CDW_N_074	"Norman door at Peterborough Cathedral" by Peter Bester is copyright
CDW_N_085	"Norman Door" by UEA Landscape is copyright
CDW_N_098	"south door" by Broads Marshman is copyright
CDW_N_101	"SU8504 Romanesque door and window, Chichester Cathedral" by Julian P Guffogg is licensed under CC BY-SA 2.0.
CDW_N_105	"W front, central door." by "photographer not known" is copyright
CDW_N_106	"N doorway, general view" by Harry Sunley copyright.
CDW_N_110	"W doorway, external view" by Harry Bodenham copyright.
CDW_N_112	"Nave, N doorway." by Allan Brodie is copyright
CDW_N_181	"SW transept, W face, Gallery S window" by Ron Baxter is copyright
CDW_N_184	"Tower lower lateral windows" by Ron Baxter is copyright
CDW_N_189	"S transept, W wall, S aisle, window" by Ron Baxter is copyright
CDW_N_192	"Nave, N window" by Ron Baxter is copyright
CDW_N_211	"S clerestory windows W-E" by Ron Baxter is copyright

Table A.10: Sources of Clustering Templates used in Figures – Table 2 of 3
See also Table A.11 and Table A.9. Note: not all templates used are listed above, only those used in figures.

Reference	Source
CWN_GO_005	“Tudor window (3)” by david.robarts is copyright
CWN_GO_037	“Charlecote Park” by Alan Rust is copyright
CWN_GO_056	"Eglwys St Cristiolus" by steveandthedogs is copyright
CWN_GO_097	"The Gatehouse-Charlecote Park" by Lilla~Rose is copyright
CWM_GO_022	"Base Court, Hampton Court Palace" by Rob Telford is copyright
CWM_GO_034	"Charlecote Park (east elevation)" by jacquemart is copyright
CWM_GO_068	“King Edmund's church, Greensted” by Webrarian is copyright
CWM_GO_101	“Rufford Old Hall, Lancashire” by Rex Harris is copyright
CWW_GO_003	"04 Packwood House – Warwickshire – Main house 20110723" by Davids Unusual Destinations is copyright
CWW_GO_010	"2015.08.30 Athelhampton House (34)" by Kotatsu Neko 808 is copyright
CWW_GO_015	Believed to be a *Flickr* image but source location not recorded (in error)
CWW_GO_021	"Baddesley Clinton" by Owl lover is copyright
CWW_GO_027	"Baddesley Clinton" by peet-astn is copyright
CWW_GO_037	Since downloading the image has been removed from the internet. Previously found here.
CWW_GO_041	"Coughton Court Warwickshire. 5B0_0007" by Bill Shakespeare is copyright
CWW_GO_048	"Godmanchester, 5117.jpg" by dinkarsabnis is copyright
CWW_GO_068	“oak house” by Sean Garrett is licensed under CC BY-NC-SA 2.0
CWW_GO_071	"Packwood House 31-05-2009-5244" by Kerry Garratt is licensed under CC BY-SA 2.0.
CWW_GO_074	"Packwood House 31-05-2009-5244" by Kerry Garratt is licensed under CC BY-SA 2.0.
CWW_GO_077	"pl15sept11 308" by Pierre Langlois is copyright
CWW_GO_083	"The Guildhall, Lavenham (NT)" by Mike Jones is copyright
CWW_GO_087	"Trinity College Cambridge" by Neasan O'Neill is copyright
CWW_GO_090	"Tudor Life at Kentwell Hall 1535, June 2009, Suffolk, England" by Niko S90 is copyright
CWW_GO_096	"Wythenshawe hall building" by Andrew Duxbury is copyright
CW_N_079	“Colchester castle window” by Marion CW (Marion in Cornwall) is copyright
CW_N_109	“Broughton Poggs, Oxfds” by extraordinarybookofdoors.com is copyright, used with permission
CW_N_150	“Exterior, N transept, W wall, aisle windows” by Ron Baxter is copyright
CW_N_193	“The late Norman doorway, All Saints Church, Sutton Courtenay” by Bressons_Puddle is copyright

Table A.11: Sources of Clustering Templates used in Figures – Table 3 of 3
See also Table A.9 and Table A.10. Note: not all templates used are listed above, only those used in figures.

Reference	Source
FI_1	“Big Ben in Westminster, London” by Defence Images is licensed under CC BY-NC 2.0
FI_2	“Welford Park, Newbury, Berkshire” by Amanda Slater is licensed under CC BY-SA 2.0
FI_3	“Holyrood palace” by Asif Musthafa is licensed under CC BY-NC-ND 2.0
FI_4	“South facing front to Belton House” by Wehha is licensed under CC BY-SA 3.0
FI_5	“Blenheim_Palace: Unprocessed (img_8298_hdr)” by Peter Gawthrop is licensed under CC BY-NC 2.0

Table A.12: Sources of Façade Images

Reference	Source
PH_1	“Lytham Hall” by Alan Cards is copyright
PH_2	“MTJLondon5028” by Mark Tewdwr-Jones is copyright
PH_3	“Sun House Frognal Way Hampstead 1935: London modernism” by mermaid is licensed under CC BY-NC-ND 2.0
PH_4	“Sun House” by Steve Cadman is licensed under CC BY-SA 2.0
PH_5	“Architecture Old & New” by Robin Denton is copyright

Table A.13: Source of Photos (used in Figures but not Elsewhere)

Reference	**Source**
TD_2_05	“Part L Regent Oak door” by howdens.com is copyright
TD_2_06	“H2XG glazed door” by howdens.com is copyright
TD_2_07	“Richmond M” by Milli Richards has an unknown licence
TD_2_08	Since downloading the image has been removed from the internet. Previously found here.
TW_2_23	Derivative of: “January 24th – Home of the Future” by Stephen Train is licensed under CC BY-NC 2.0.
TW_2_27	Derivative of: “January 24th – Home of the Future” by Stephen Train is licensed under CC BY-NC 2.0.
TW_2_28	Derivative of: “January 24th – Home of the Future” by Stephen Train is licensed under CC BY-NC 2.0.
TW_2_29	Derivative of: “January 24th – Home of the Future” by Stephen Train is licensed under CC BY-NC 2.0.
TW_2_30	Derivative of: “January 24th – Home of the Future” by Stephen Train is licensed under CC BY-NC 2.0.
TW_2_31	Derivative of: “January 24th – Home of the Future” by Stephen Train is licensed under CC BY-NC 2.0.
TW_2_32	Derivative of: “January 24th – Home of the Future” by Stephen Train is licensed under CC BY-NC 2.0.
TW_2_33	Derivative of: “January 24th – Home of the Future” by Stephen Train is licensed under CC BY-NC 2.0.
TW_2_34	Derivative of: “January 24th – Home of the Future” by Stephen Train is licensed under CC BY-NC 2.0.
TW_2_35	Derivative of: “January 24th – Home of the Future” by Stephen Train is licensed under CC BY-NC 2.0.
TW_2_36	Derivative of: “January 24th – Home of the Future” by Stephen Train is licensed under CC BY-NC 2.0.
TW_2_37	Derivative of: “January 24th – Home of the Future” by Stephen Train is licensed under CC BY-NC 2.0.
TW_2_38	Derivative of: “January 24th – Home of the Future” by Stephen Train is licensed under CC BY-NC 2.0.
TW_2_39	Derivative of: “January 24th – Home of the Future” by Stephen Train is licensed under CC BY-NC 2.0.
TW_2_40	Derivative of: “January 24th – Home of the Future” by Stephen Train is licensed under CC BY-NC 2.0.
TW_2_41	Derivative of: “January 24th – Home of the Future” by Stephen Train is licensed under CC BY-NC 2.0.
TW_2_42	Derivative of: “January 24th – Home of the Future” by Stephen Train is licensed under CC BY-NC 2.0.
TW_2_43	Derivative of: “January 24th – Home of the Future” by Stephen Train is licensed under CC BY-NC 2.0.
TW_2_44	Derivative of: “January 24th – Home of the Future” by Stephen Train is licensed under CC BY-NC 2.0.
TW_2_45	Derivative of: “January 24th – Home of the Future” by Stephen Train is licensed under CC BY-NC 2.0.
TW_2_46	Derivative of: “January 24th – Home of the Future” by Stephen Train is licensed under CC BY-NC 2.0.
TW_2_47	Derivative of: “January 24th – Home of the Future” by Stephen Train is licensed under CC BY-NC 2.0.
TW_2_49	Derivative of: “January 24th – Home of the Future” by Stephen Train is licensed under CC BY-NC 2.0.

Table A.14: Sources of Templates – C20 style (Non-Clustering)

Reference	Source
TD_GO_01	"Wickhampton, Church of St. Andrew" by Gary Trouton is licensed under CC BY-NC 2.0.
TD_GO_02	"Tudor Gateway – Pilgrims Way, Detling, Kent, UK. Image" by Groundspeak, Inc is licensed for public use by Groundspeak, Inc.
TW_GO_01	Amalgam of: "15th C. transitional window tracery, the Church of St Mary, Wilby, Suffolk" by Spencer Means is licensed under CC BY-SA 2.0; and "The west window (c.1460), the Church of St Mary, Wilby, Suffolk, England" by Spencer Means is licensed under CC BY-SA 2.0.
TW_GO_02	"Y-tracery ... Cossington" by Leicestershire Victoria County History Trust is licensed for public use by Leicestershire Victoria County History Trust
TW_GO_05	Derivative of: "northwest window of the chancel" by Meldreth Local History Group is copyright, with permission
TW_GO_06	Derivative of: “Wickhampton” by Simon K is copyright
TW_GO_07	Derivative of: “Wickhampton” by Simon K is copyright
TW_GO_08	Derivative of: “Wickhampton” by Simon K is copyright
TW_GO_11	Derivative of: “St Andrew's Church, Tredunnock, Gwent” by ChurchCrawler is copyright
TW_GO_14	Derivative of: "northwest window of the chancel" by Meldreth Local History Group is copyright, with permission
TW_GO_15	"window at east end of the south chancel aisle" by Church History Project provides no copyright information on the website.
TW_GO_16	Amalgam of: TW_GO_08 and TW_GO_11.
TW_GO_17	Amalgam of: TW_GO_08 and TW_GO_11.
TW_GO_18	Derivative of: "northwest window of the chancel" by Meldreth Local History Group is copyright, with permission
TW_GO_20	“St. Nicholas – Iford – East Sussex – 11th Century” by Badger of the Bank is copyright
TW_GO_27	Derivative of: “Idbury St Nicholas Large Perpendicular south-east chancel window -231” by Guy Thornton is copyright
TW_GO_29	Amalgam of: TW_GO_27 and TW_GO_08.

Table A.15: Sources of Templates – Gothic Style (Non-Clustering)

Reference	Source
TW_GR_1x2	Derivative of: “Winslow Hall" by Lesley is licensed under CC BY-SA 2.0.
TW_GR_1x4	Derivative of: “Winslow Hall" by Lesley is licensed under CC BY-SA 2.0.
TW_GR_2x2	Derivative of: “Winslow Hall" by Lesley is licensed under CC BY-SA 2.0.
TW_GR_2x2x3	Derivative of: “Winslow Hall" by Lesley is licensed under CC BY-SA 2.0.
TW_GR_2x3	Derivative of: “Winslow Hall" by Lesley is licensed under CC BY-SA 2.0.
TW_GR_2x4	Since downloading the Image has been removed from the internet. Previously found here.
TW_GR_2x5	Derivative of: “Winslow Hall" by Lesley is licensed under CC BY-SA 2.0.
TW_GR_3x2	Derivative of: “Winslow Hall" by Lesley is licensed under CC BY-SA 2.0.
TW_GR_3x3	Derivative of: “Winslow Hall" by Lesley is licensed under CC BY-SA 2.0.
TW_GR_3x4	Derivative of: “Winslow Hall" by Lesley is licensed under CC BY-SA 2.0.
TW_GR_3x5	Derivative of: “Winslow Hall" by Lesley is licensed under CC BY-SA 2.0.
TW_GR_3x6	Derivative of: “Winslow Hall" by Lesley is licensed under CC BY-SA 2.0.
TW_GR_4x2	Derivative of: “Winslow Hall" by Lesley is licensed under CC BY-SA 2.0.
TW_GR_4x3	Derivative of: “Winslow Hall" by Lesley is licensed under CC BY-SA 2.0.
TW_GR_4x6	Derivative of: “Winslow Hall" by Lesley is licensed under CC BY-SA 2.0.
TW_GR_4x7	Derivative of: “Winslow Hall" by Lesley is licensed under CC BY-SA 2.0.
TW_GR_A	“Arched window” by psyberartist is licensed under CC BY 2.0.

Table A.16: Sources of Templates – Georgian-Regency Style (Non-Clustering)

Reference	Source
TD_GR_D01	"St Chad" from Pevsner Architectural Glossary App by Yale University Press is copyright, with permission
TD_GR_D02	"Lytham Hall" from Pevsner Architectural Glossary App by Yale University Press is copyright, with permission
TD_GR_D03	"Portico" by Arthur John Picton is licensed under CC BY-NC 2.0.
TD_GR_S01	Amalgam of: "Dublin Doors" by Jim McDougallis licensed under CC BY 2.0; and "Dublin yellow and red Georgian doors" by hugovk is licensed under CC BY-NC-SA 2.0.
TD_GR_S02	Amalgam of: "Dublin Doors" by Jim McDougallis licensed under CC BY 2.0; and "Dublin yellow and red Georgian doors" by hugovk is licensed under CC BY-NC-SA 2.0.
TW_GR_3x5E	"IBM Hursley" by Alexis Birkill is licensed under CC BY-NC-SA 2.0
TD_N_33	"SP0924 West door, St. Michael and All Angels, Guiting Power, Glos" by nick macneill is licensed under CC BY-SA 2.0.
TD_N_37	"The door – Heath Chapel" by R Garbett is licensed under CC BY-NC-ND 2.0.
TW_N_02	"normanwindow" by leonardstanley.org.uk provides no copyright information on the website.
TW_N_03	"Close up of the Norman window in the south wall of the nave February 2010" by Bedfordshire Archives and Records Service provide unclear copyright information on their website.
TW_N_06	"Penmon, Anglesey" by extraordinarybookofdoors.com is copyright, with permission
TW_N_07	"east window" by Gerry Cordon provides no copyright information on the website.
TW_N_08	Derivative of: "SK6655 Norman window" by Richard Croft is licensed under CC BY-SA 2.0.
TW_N_09	Derivative of: "SK6655 Norman window" by Richard Croft is licensed under CC BY-SA 2.0.
TW_N_10	Derivative of: "Norman Window" by Martin Stone is licensed under CC BY-SA 2.0.
TW_N_13	Derivative of: "St George's Church, Eastergate (Herringbone Brickwork)" by Hassocks5489 is licensed under CC0 1.0.
TW_N_15	Derivative of: "SU8504 Romanesque door and window, Chichester Cathedral" by Julian P Guffogg is licensed under CC BY-SA 2.0.
TW_N_18	Derivative of: "SU8504 Romanesque door and window, Chichester Cathedral" by Julian P Guffogg is licensed under CC BY-SA 2.0.

Table A.17: Sources of Templates – Norman Style (Non-Clustering)

Reference	Source
TZ_GR_5	"[5040] Bath Roman Baths" by Budby is copyright
TZ_GR_6	"Bath – Sydney Place" by Tranter Dewy is copyright
TZ_GR_7	"Regency House, Queen's Square, Bristol" by Mark Woodland is copyright
TZ_GR_10	"Edinburgh: Charlotte Square" by James Stringer is licensed under CC BY-NC 2.0
TZ_GR_14	"1 Bedford Square WC1" by Jamie Barras is copyright
TZ_GR_23	"[40231] York 118 Micklegate" by Budby is copyright
TZ_GON_01	"Church of All Saints, Bolton" by Jhsteel is licensed under CC BY-SA 3.0

Table A.18: Sources of Training Images
Note: not all training images used are listed above, only those used in figures.

Appendix B Extended Background

B.1 Architectural Style – Gothic Styles & Tracery

The categorisation of tracery types and the assignment of tracery type to Gothic style in can be broadly summarised as follows, based on: Ching (2011); Hart (2010); Pevsner and Sambrook (2010). Lancets are single light windows with a pointed arch. Plate tracery is the earliest form, introduced c.1200, with shapes cut through solid masonry. Bar tracery, introduced c.1250, comprises patterns formed by 'intersecting moulded ribwork'. Note that bar tracery is not an architectural style, and that post c.1250 almost all tracery is constructed using bar tracery. When there are two lights intersecting tracery is called 'Y-tracery'. Reticulated tracery is a 'netlike arrangement of repeated geometrical figures'. Curvilinear tracery is characterised by irregular curved forms in a 'flowing' manner. Rectilinear tracery, also called panel tracery, often has more than one tier. The Tudor substyle is differentiated from the Perpendicular by the lower flattened arch. Stained glass incorporating leadwork and plainly glazed diamond-shaped leadwork were common (Marks, 2006,p. xxiv).

B.2 3D Reconstruction – Key Stages in the Development of Methods for Creating Virtual Tours

Using a set of images of a location from both private collections and *Flickr* photos (Flickr, 2017), Snavely *et al* used the Scale-Invariant Feature Transform (SIFT) feature extraction, detection and matching algorithm (Lowe, 2004) to match keypoints from which camera parameters for each image were then obtained and point clouds derived. While Snavely's work allowed for manual labelling of objects in photos and alluded to the possibilities of linking to photos with pre-existing tags it, arguably, did not major on the semantic attribution. A further shortfall of the method was its inability to enable applications which require polygonal- or polyhedral-based spatial analysis, on the basis that the tour was point-cloud based. Nonetheless, the work represented an important moment in the evolution of cultural heritage building reconstruction and gave a hint of the potential for rich cultural heritage geo-data applications that might result from semantic attribution.

Subsequent work from Simon and Seitz (2008), itself an evolution of *Photo Tourism*, determined to extract useful semantic content from mined *Flickr* tags and automatically label objects in a SfM image-based reconstructed 3D scene. Once again though, the method was point-cloud based. A further evolution still was that of Russell *et al.* (2013) who attached semantic content to architectural scenes with their *3D Wikipedia* work. That study employed SfM-MVS on internet photo collections and used descriptive text for a location (such as from *Wikipedia*) within reconstructed scenes. The resulting application was navigable via the descriptive text, jumping to objects on a click or through an animated tour. The user experience was also point-cloud based though. Essentially, *Photo Tourism*, Simon and Seitz (2008) and *3D Wikipedia* segment point clouds, a technique on which Grilli *et al.* (2017) provide a useful summary.

Appendix C Extended Method – Data & Data Challenges

C.1 Data Collection Strategies, Data Formats & Cataloguing

In order to collect large numbers of images for consideration, a screen-scraping program called *Bulk Image Downloader* (BID) from Antibody Software (2017) was used. See Section 3.3 for an explanation of why it was necessary to collect large numbers of images. The methods described in Chapters 4-6 made extensive use of string manipulation of image and template filenames, including the extraction of class granularity structure (see Section 3.6). Consequently, the *Bulk Rename Utility* from TGRMN (2017) was used for renaming files in bulk, including for the generation of sequential template version numbers (examples) for the clustering trial – see Section 3.6 and Appendix D.

The image files that were downloaded from the internet, and also the texture map images for the 3D building models, were Joint Photographic Experts Group (JPEG) and, to a lesser extent, Portable Network Graphics (PNG) formats from JPEG (2017) and W3C (2003) respectively. All imagery was therefore rectangular, although some PNG texture map images used transparency to simulate non-rectangular shapes. All templates and façade images were JPEG format.

The methods in Chapter 4-7 made use of exchangeable image file format (EXIF) tags, where EXIF is a set of standards, including tags, for digital camera images, as specified by JEITA (2017). Of the two image formats used, only JPEG officially supports EXIF, however PNGs can hold EXIF tags (Harvey, 2015), and it was chosen to do so to enable a consistent approach with the method for processing JPEGs (see Section 3.8).

C.2 Construction Dates for Listed Buildings in Great Britain & Challenges

Of the approximately 455,000 listed buildings in Great Britain, 83%, 10% and 7% are in England, Scotland and Wales respectively (source: Cadw, 2017; Historic England, 2017; Historic Environment Scotland, 2017). As a result of this split, the below discussion focusses on listed buildings in England, although the challenges presented are similar for Scotland and Wales. Most buildings in England built pre-1840 are on the 'listed' preservation register maintained by Historic England (Department for Culture Media and Sport, 2010). However, listed buildings have often been altered over time, due to their age. As such no record exists providing a comprehensive single construction date for every English listed building, although the dates for the phases of alterations are usually available. In turn however, it is often challenging for the preservation authorities to assign a principal construction date. Attempting to do so can often require subjective judgement, such as how much of the core fabric of a building is of a certain age. A judgement as to *what* represents the core fabric of a building is also, arguably, subjective, not to mention that it might vary between different buildings. Nonetheless, an indication of the spread of the 'age' of England's listed buildings can at least be obtained as follows. Based on the oldest architectural component in a building, 0.5%, 2% and 26% of English listed buildings fall within the C11-C12, C13-C15 and C16-C18 date ranges (Historic England, 2017). Those date ranges correspond, approximately, to those for each of the oldest three styles used in this work. Note that the three styles in question do possess some date overlap.

C.3 CityGML – Conversion from SketchUp Format

As mentioned in Chapter 2, Trimble *SketchUp* 3D building models from the Trimble *3D Warehouse* were converted to CityGML using Trimble *SketchUp Pro* with the *CityEditor* plug-in. During the conversion process the model was optimised, the semantics were edited, and correct syntactic structure enforced. The conversion process also used the TIG *Reverse Face Tools* plug-in (TIG, 2010) and Esri's *ArcMap* (Esri, 2017) and was carried out as follows.

In *SketchUp* all model components were exploded and unlocked and then grouped and classified into individual buildings, terrain, vegetation and transport. A surface analyser was then run on the building polygons to classify them as *GroundSurface*, *WallSurface* and *RoofSurface*. These steps optimised the model and enforced correct syntactic structure in the CityGML at export. Because this study was only concerned with the exterior of the building, textures on the reverse side of all faces were removed. *SketchUp* uses the geodetic coordinate reference system WGS 84, that is, it uses latitude and longitude. CityGML requires Cartesian coordinates. Consequently, the WGS84 location of the 3D model was transformed into the geo-spatial projected coordinate reference system EPSG:32630 (WGS 84 / UTM zone 30N) using *ArcMap*. Finally, the 3D building model was exported from *SketchUp* format to CityGML, at LOD3. A number of validation checks were then conducted on the CityGML XML using a text editor, after which the CityGML was imported into *SketchUp Pro* to compare with the original *SketchUp* model, including a check of the geospatial location.

Appendix D Additional Method – Clustering of Templates – Trial

D.1 Introduction

The method described in Chapter 5 used manually selected templates as an input to HOG-based template matching (where the method then converted the template images to HOG descriptors in advance of running template matching). As an alternative approach, a trial was conducted using clustering of templates, specifically their HOG descriptors. The HOG descriptors for the cluster centres were then used as an input to HOG-based template matching. The rationale for doing so was that clustering had the potential to reduce the number of templates used in the HOG-based template matching pipeline. If this were the case, then runtime for the pipeline should be reduced. Furthermore, clustering could potentially result in a better selection of templates, noting that there could be a difference between what appears to be a distinguishing feature to the human eye, and what represents a distinguishing feature to a HOG descriptor.

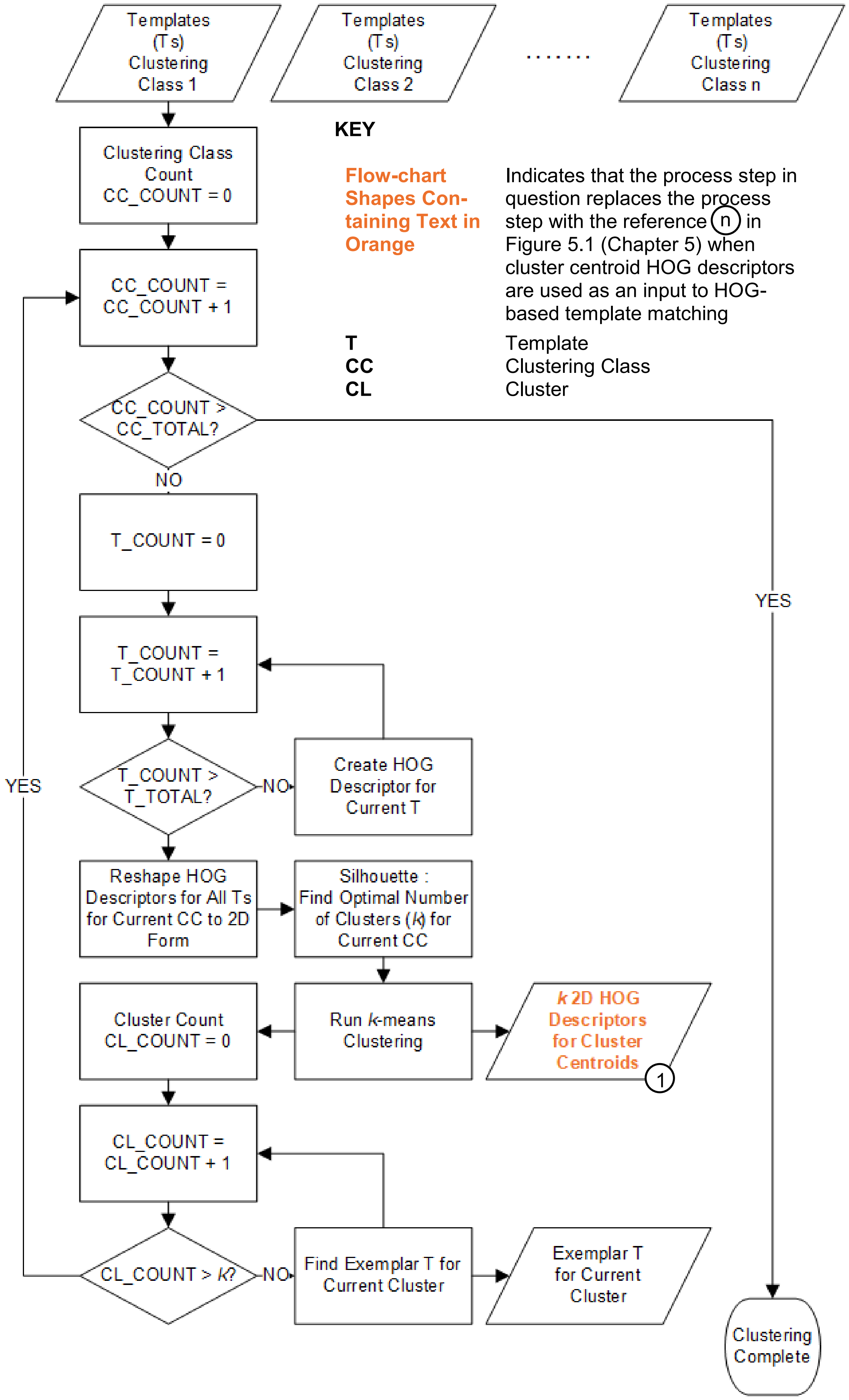
Note that aside from a linkage to Chapter 3 (data) and Chapter 5 (HOG-based template matching) this appendix is stand-alone with regard to the method chapters. However, the insights revealed by the output of the clustering process support the abovementioned trend that threads through much of this work, namely that HOG descriptors can detect edges that are either more, or less, apparent to the human observer. The approach taken for the clustering trial is detailed below.

D.2 Cluster Centroids of Template HOG Descriptors – Trial

The use of templates that were more distinct from one another could result in more true positives and fewer false positives during HOG-based template matching. By way of illustration, imagine a person choosing from a pool of 100 example 'Norman Style Window' templates. They may, for example, choose 10 visually distinct Norman window templates to use for processing, perhaps basing their selection on a combination of the number of orders and the carving style. Conversely, by clustering the HOG descriptors for the same set of 100 example 'Norman Style Window' templates, the type of carving may be less of a distinguishing feature within a HOG descriptor. This might, for example, result in four cluster centres, based only on the number of orders. In turn, runtime for HOG-based template matching on a test image would be reduced if 4 versus 10 templates were used. For sake of illustration, assume that partitioning solely on the number of orders was indicative that the number of orders was the optimal distinguishing feature for Norman windows. If that was the case, then more true positives and fewer false positives should result on the texture map images of the 3D building models, compared using the 10 templates chosen by the person. This would be on the basis that clustering had chosen the best examples, by selecting features represented by the corresponding HOG descriptor i.e. ones that were more distinct than those selected by the person. Of course, this is just a possible scenario (clustering wouldn't necessarily partition the clusters solely based on orders) but it illustrates the point.

Figure D.1 shows the process flow for the clustering approach used. The classes of template used in clustering were as described in Chapter 3. To reiterate, this meant eight classes (i.e. $\mathrm{n} = 8$ in Figure D.1). Clustering was run on each class at a time, and each class resulted in *k* clusters. Unlike the templates used for the method described in Chapter 5, aspect ratio was used to further split classes e.g. the 'Gothic Style Window' class was split into 'Width Narrow', 'Width Medium' and 'Width Wide'. To distinguish this approach, a class on which clustering was run is referred to here as a 'clustering class'.

For each clustering class, 31D-HOG descriptors were created for each template example. The 31D HOG descriptor for each 8×8-pixel grid cell across a template example was reshaped into a 2D form that *k*-means required, as follows and as shown in Figure D.2. Chapter 2 described how the pixel dimensions of a template determine the size of the descriptor grid – in the case of the example 28×36-pixel template in the figure, a descriptor grid of 2×3 grid cells will result due to masking at the periphery of the template,


Templates (Ts) Clustering Class 1
Templates (Ts) Clustering Class 2
....
Templates (Ts) Clustering Class n
Clustering Class Count CC_COUNT = 0
CC_COUNT = CC_COUNT + 1
CC_COUNT > CC_TOTAL?
NO
YES
T_COUNT = 0
T_COUNT = T_COUNT + 1
T_COUNT > T_TOTAL?
NO
Create HOG Descriptor for Current T
Reshape HOG Descriptors for All Ts for Current CC to 2D Form
Silhouette : Find Optimal Number of Clusters (k) for Current CC
Cluster Count CL_COUNT = 0
Run k-means Clustering
k 2D HOG Descriptors for Cluster Centroids
1
CL_COUNT = CL_COUNT + 1
YES
CL_COUNT > k?
NO
Find Exemplar T for Current Cluster
Exemplar T for Current Cluster
Clustering Complete

KEY

Flow-chart Shapes Containing Text in Orange	Indicates that the process step in question replaces the process step with the reference (n) in Figure 5.1 (Chapter 5) when cluster centroid HOG descriptors are used as an input to HOG-based template matching
T	Template
CC	Clustering Class
CL	Cluster

Figure D.1: Process Flow – Cluster Centroids of Template HOG Descriptors – Trial

i.e. the descriptor for the template will be of dimensions 2×3×31. Here the 2×3×31 descriptor was reshaped into 1D 186×1 form required for clustering. *k*-means will only cluster if each example is of the same size. All template examples for a single clustering run therefore had the same pixel and 1D HOG descriptor dimensions. Also, note that each 186×1 1D descriptor will represent a single row in the table of data that is used for clustering, so that each row of that table represents a template example.

The silhouette clustering evaluation criterion (Rousseeuw, 1987) was then used to determine the optimum number of clusters in advance of clustering. The silhouette method requires a list of the number of clusters to evaluate e.g. [1, 2, 3], whereupon the method then checks if each number represents the optimum number of clusters (*k*). Preliminary experiments determined that a list comprising [1:20] achieved the best results. Subsequently, *k*-means clustering was run for each clustering class. Within each clustering class, each cluster centroid resulted in a HOG descriptor, there being *k* cluster centroids.

The template example within a cluster with the shortest Euclidean distance to the cluster centroid HOG descriptor was chosen as the cluster exemplar. This provided visual insights regarding the manner in which the clustering had partitioned the examples, which would otherwise be impossible to discern from the cluster centroid HOG descriptor alone.

The number of clusters and the number of examples in each cluster are shown in Table D.1. The exemplar templates that represent each cluster centre are shown in Figure D.3, Figure D.4 and Figure D.5 for the following clustering class templates respectively: ‘Gothic Style Window’ and ‘Norman Style Window’; ‘Gothic Style Door’; and ‘Norman Style Door’.

Template Class (Granularity 2)	Clustering Class	Num. of Clusters	Num. of Examples in each Cluster
Gothic Style Window	‘Width Narrow’	2	64, 36
Gothic Style Window	‘Width Medium’	2	36, 64
Gothic Style Window	‘Width Wide’	15	5, 7, 2, 7, 5, 9, 6, 2, 17, 3, 4, 20, 11, 1, 1
Norman Style Window	N/A	2	64, 36
	SUBTOTAL	**21**	
Gothic Style Window	‘Width Narrow’	2	55, 45
Gothic Style Window	‘Width Wide’	19	2, 9, 2, 11, 6, 3, 4, 6, 3, 3, 4, 2, 1, 3, 6, 4, 11, 6, 14
Norman Style Door	‘Width Narrow’	2	57, 43
Norman Style Door	‘Width Wide’	19	2, 5, 4, 2, 9, 2, 8, 3, 5, 5, 2, 12, 4, 3, 10, 6, 4, 8, 6
	SUBTOTAL	**42**	
	TOTAL	**63**	

Table D.1: Number of Template Clusters by Clustering Class (*k*-means) – Trial

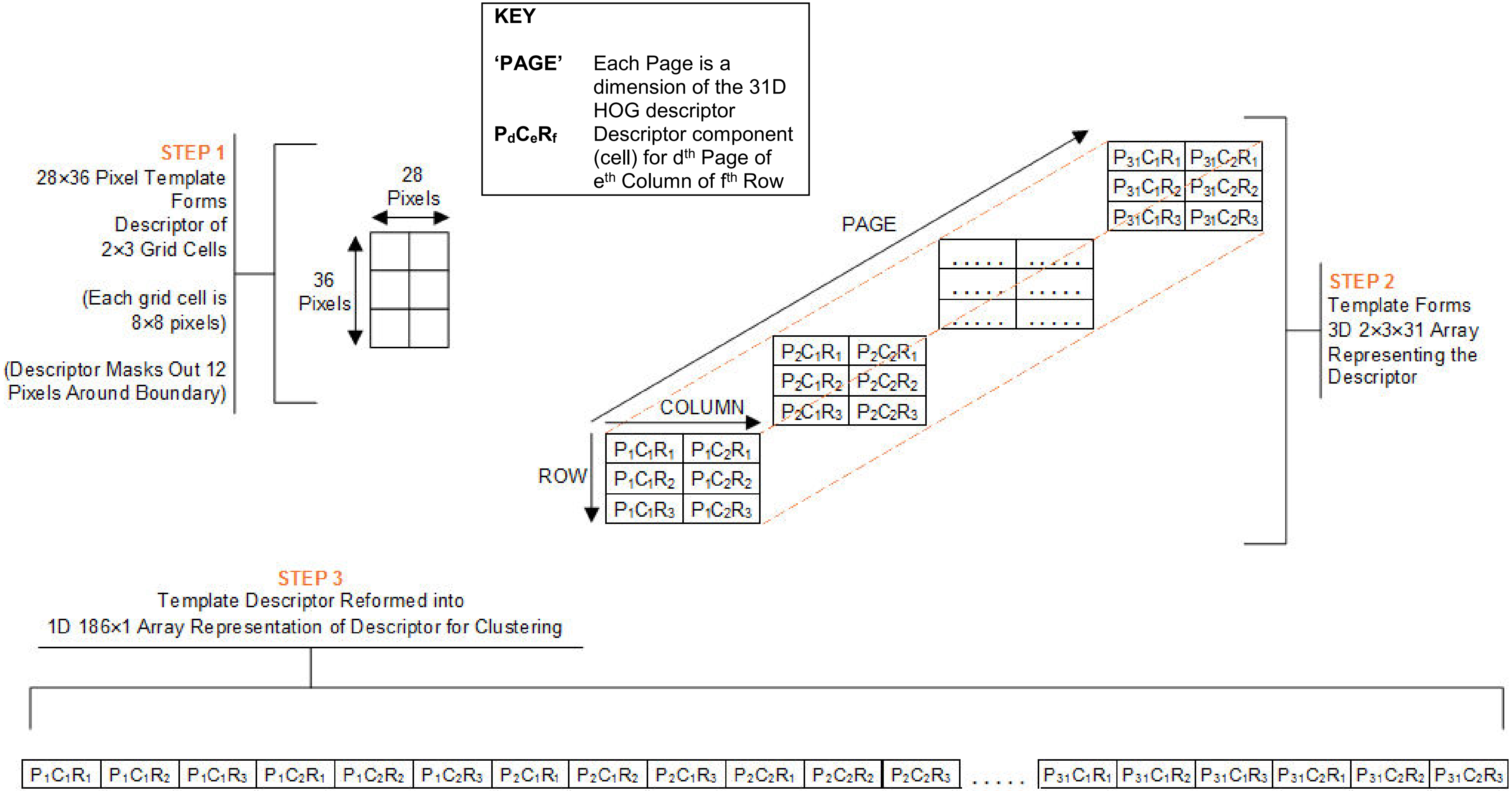

KEY
'PAGE' Each Page is a dimension of the 31D HOG descriptor
$P_dC_eR_f$ Descriptor component (cell) for dth Page of eth Column of fth Row
STEP 1
28×36 Pixel Template Forms Descriptor of 2×3 Grid Cells
(Each grid cell is 8×8 pixels)
(Descriptor Masks Out 12 Pixels Around Boundary)
28 Pixels
36 Pixels
PAGE
COLUMN
ROW
$P_1C_1R_1$ $P_1C_2R_1$
$P_1C_1R_2$ $P_1C_2R_2$
$P_1C_1R_3$ $P_1C_2R_3$
$P_2C_1R_1$ $P_2C_2R_1$
$P_2C_1R_2$ $P_2C_2R_2$
$P_2C_1R_3$ $P_2C_2R_3$
....
....
....
$P_{31}C_1R_1$ $P_{31}C_2R_1$
$P_{31}C_1R_2$ $P_{31}C_2R_2$
$P_{31}C_1R_3$ $P_{31}C_2R_3$
STEP 2
Template Forms 3D 2×3×31 Array Representing the Descriptor
STEP 3
Template Descriptor Reformed into 1D 186×1 Array Representation of Descriptor for Clustering
$P_1C_1R_1$ $P_1C_1R_2$ $P_1C_1R_3$ $P_1C_2R_1$ $P_1C_2R_2$ $P_1C_2R_3$ $P_2C_1R_1$ $P_2C_1R_2$ $P_2C_1R_3$ $P_2C_2R_1$ $P_2C_2R_2$ $P_2C_2R_3$ ..... $P_{31}C_1R_1$ $P_{31}C_1R_2$ $P_{31}C_1R_3$ $P_{31}C_2R_1$ $P_{31}C_2R_2$ $P_{31}C_2R_3$

Figure D.2: HOG Descriptor Reshaping Required for Clustering
Note that the template shown is smaller than the templates used – template dimensions are purely for the purposes of illustration.

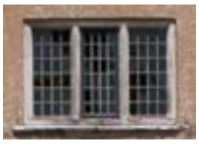
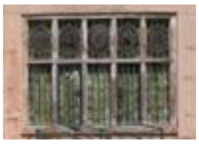
Gothic Style Window 'Width Narrow' (2 Clusters)

Image cannot be displayed due to copyright	Image cannot be displayed due to copyright
CWN_GO_056	CWN_GO_097

Gothic Style Window 'Width Medium' (2 Clusters)

Image cannot be displayed due to copyright	Image cannot be displayed due to copyright
CWM_GO_022	CWM_GO_034

Gothic Style Window 'Width Wide' (15 Clusters)

Image cannot be displayed due to copyright	Image cannot be displayed due to copyright	Image cannot be displayed due to copyright	Image cannot be displayed due to copyright	Image cannot be displayed due to copyright
CWW_GO_003	CWW_GO_010	CWW_GO_015	CWW_GO_021	CWW_GO_027
Image cannot be displayed due to copyright	Image cannot be displayed due to copyright	Image cannot be displayed due to copyright		
CWW_GO_037	CWW_GO_041	CWW_GO_048	CWW_GO_071	CWW_GO_074
Image cannot be displayed due to copyright	Image cannot be displayed due to copyright	Image cannot be displayed due to copyright	Image cannot be displayed due to copyright	Image cannot be displayed due to copyright
CWW_GO_077	CWW_GO_083	CWW_GO_087	CWW_GO_090	CWW_GO_096

Norman Style Window (2 Clusters)

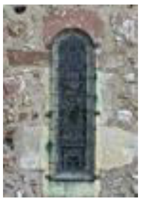
	Image cannot be displayed due to copyright
CW_N_109	CW_N_193

Figure D.3: Gothic & Norman Window Clustering Class Exemplars (*k*-means) – Trial
Note that the relative sizes of the templates shown here, in Figure D.3, Figure D.4, in Figure D.5, and in Figure 3.8, Figure 3.9, Figure 3.10 and Figure 3.11 in Chapter 3 are as used in the methods within this work. Also note: some template examples were rejected prior to clustering, hence references contain numbering above 100.

Gothic Style Door 'Width Narrow' (2 Clusters)

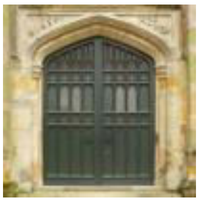
Image cannot be displayed due to copyright

CDN_GO_026

Image cannot be displayed due to copyright

CDN_GO_062

Gothic Style Door 'Width Wide' (19 Clusters)



CDW_GO_004

Image cannot be displayed due to copyright

CDW_GO_005

Image cannot be displayed due to copyright

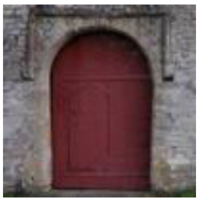
CDW_GO_013

Image cannot be displayed due to copyright

CDW_GO_018

Image cannot be displayed due to copyright

CDW_GO_020



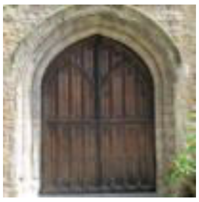
CDW_GO_026

Image cannot be displayed due to copyright

CDW_GO_027

Image cannot be displayed due to copyright

CDW_GO_034



CDW_GO_036

Image cannot be displayed due to copyright

CDW_GO_041

Image cannot be displayed due to copyright

CDW_GO_047

Image cannot be displayed due to copyright

CDW_GO_059

Image cannot be displayed due to copyright

CDW_GO_065

Image cannot be displayed due to copyright

CDW_GO_072

Image cannot be displayed due to copyright

CDW_GO_076

Image cannot be displayed due to copyright

CDW_GO_079

Image cannot be displayed due to copyright

CDW_GO_082

Image cannot be displayed due to copyright

CDW_GO_095

Image cannot be displayed due to copyright

CDW_GO_100

Figure D.4: Gothic Door Clustering Class Exemplars (*k*-means) – Trial
See the caption for Figure D.3.

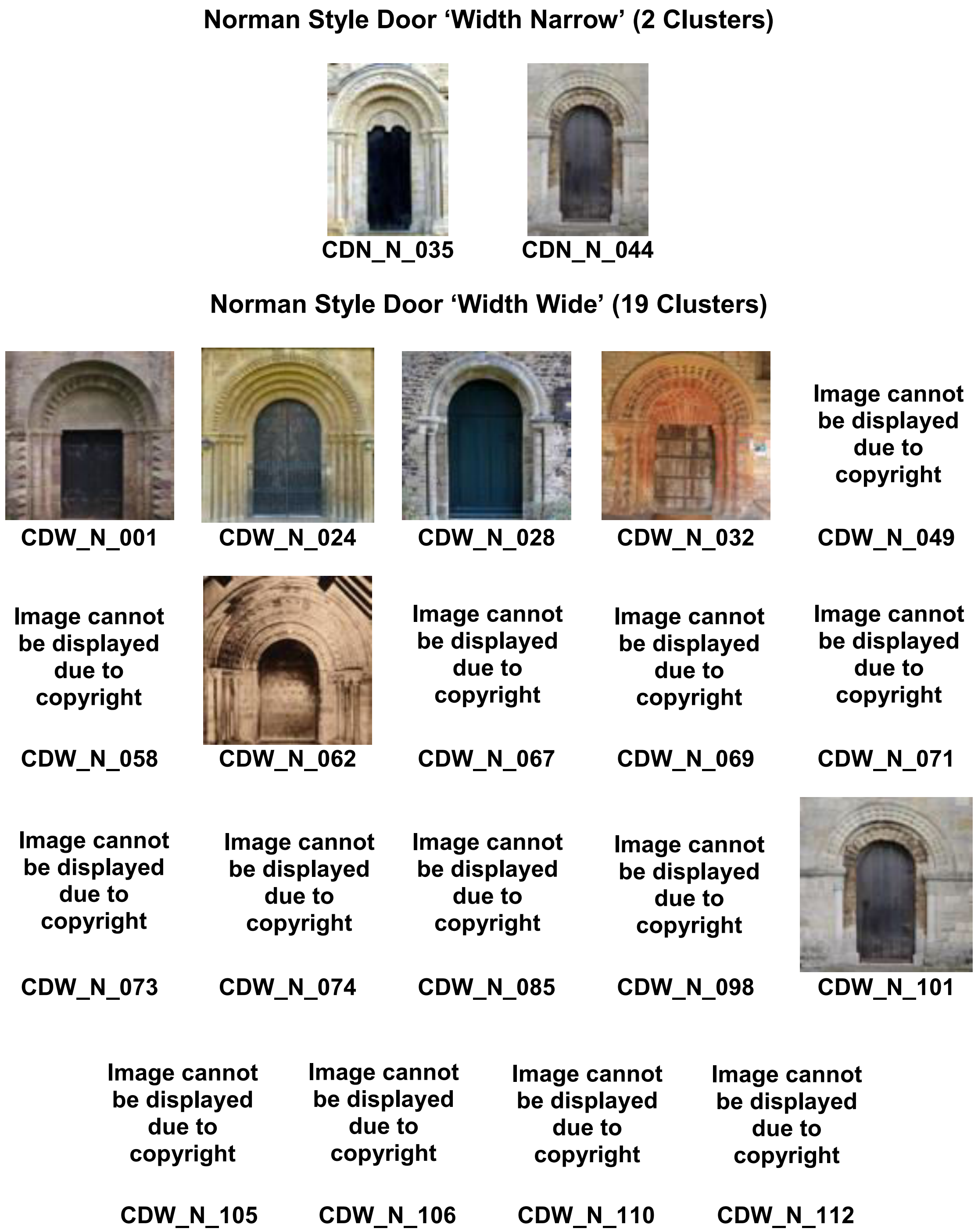

Norman Style Door 'Width Narrow' (2 Clusters)
CDN_N_035
CDN_N_044
Norman Style Door 'Width Wide' (19 Clusters)
CDW_N_001
CDW_N_024
CDW_N_028
CDW_N_032
Image cannot be displayed due to copyright
CDW_N_049
Image cannot be displayed due to copyright
CDW_N_058
CDW_N_062
Image cannot be displayed due to copyright
CDW_N_067
Image cannot be displayed due to copyright
CDW_N_069
Image cannot be displayed due to copyright
CDW_N_071
Image cannot be displayed due to copyright
CDW_N_073
Image cannot be displayed due to copyright
CDW_N_074
Image cannot be displayed due to copyright
CDW_N_085
Image cannot be displayed due to copyright
CDW_N_098
CDW_N_101
Image cannot be displayed due to copyright
CDW_N_105
Image cannot be displayed due to copyright
CDW_N_106
Image cannot be displayed due to copyright
CDW_N_110
Image cannot be displayed due to copyright
CDW_N_112

Figure D.5: Norman Door Clustering Class Exemplars (*k*-means) – Trial
See the caption for Figure D.3.

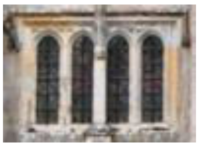
Note that the clustering class width designations ('Narrow', 'Medium', 'Wide') were removed during F-measure calculations e.g. a 'Gothic Style Window Width Narrow' template became a 'Gothic Style Window' template.

The template examples used for clustering included the templates used in the non-clustering approach, or ones visually similar. The intention was to create a similar breadth of design styles in both the non-clustering and clustering template sets. That said, with a larger number of template examples required for clustering, it was necessary to broaden the design of templates used. This meant that designs not found on any of the 3D building models occurred more frequently in the cluster templates sets, compared to those templates used for the non-clustering approach. A consequence of this was that some of the Gothic clustering templates were, in hindsight, perhaps vernacular or possibly Elizabethan style. Elizabethan style is a Classical tradition style, although the style occurred in the later 16th century i.e. very early in the Classical tradition (Pevsner Architectural Guides, 2013; Pevsner and Sambrook, 2010). Nonetheless, note again the comments above regarding a similar spread of design style in the template sets for the clustering and non-clustering approaches.

As can be seen in Table D.1, the number of clusters (and therefore the number of template exemplars that would be used for HOG-based template matching), when using *k-means* clustering, was 63. This was higher than the 55 templates used in the non-clustering approach. The pre-processing of the templates into HOG descriptors (which the clustering approach does) should, in theory, reduce HOG-based template matching runtime. The reason for this is that the formation of the HOG descriptors for the templates is effectively moved offline i.e. as a pre-processing step. However, the time required to form HOG descriptors for all 800 clustering templates and then cluster will be longer than simply forming the HOG descriptors for the 55 Gothic and Norman non-clustering templates. In practice, when running HOG-based template matching (without the SVM – see Chapter 6) only 11% of the overall runtime corresponded to the forming of the HOG descriptors (see Chapter 7). The majority of the time to form HOG descriptors will be the time taken to do so for the façade or texture map images, noting that those images are almost always larger than the templates. As a consequence, the anticipated runtime gains from removing the step to form the HOG descriptors for templates from the pipeline were negligible (and the runtime for clustering and HOG-based template matching combined would be *higher*, as noted above).

As can be seen in Figure D.3, Figure D.4 and Figure D.5, to the human observer the exemplars are, arguably, somewhat visually distinctive, though note that the templates shown are exemplars for the HOG descriptor cluster centroids. For an illustration of which template examples were placed into which cluster, for one of the clustering classes, see Figure D.6, Figure D.7, Figure D.8, Figure D.9, Figure D.10. As with the aforementioned exemplars, those figures once again appear to indicate somewhat visually distinctive separation of the clusters. Partitioning appears to be based on some combination of: the different shapes of window heads (e.g. square, round or pointed); the different numbers of lights and tiers in windows; the different design of the leadwork (diamond-shaped, square-shaped or more ornate stained-glass designs); and possibly

Cluster 1 (5 Examples)


CWW_GO_013

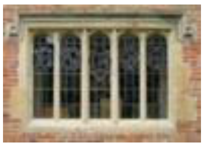
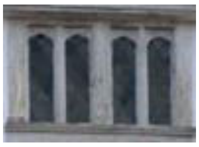
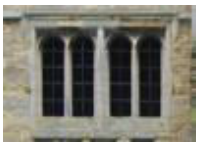
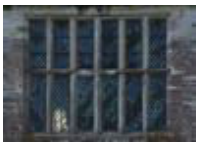

CWW_GO_040

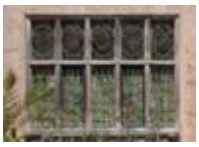
Image cannot be displayed due to copyright
CWW_GO_041

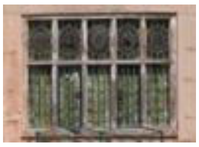

CWW_GO_051

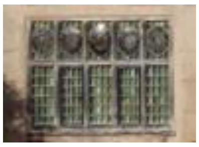

CWW_GO_079

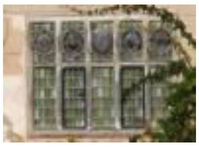
Cluster 2 (7 Examples)

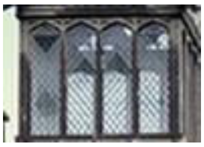

CWW_GO_030

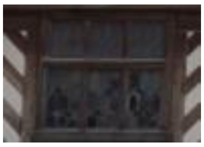

CWW_GO_073


CWW_GO_074


CWW_GO_075


CWW_GO_076


CWW_GO_078


CWW_GO_097

Cluster 3 (2 Examples)

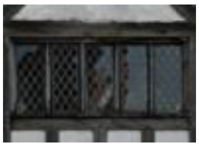
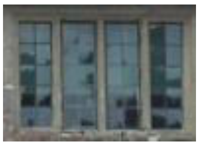
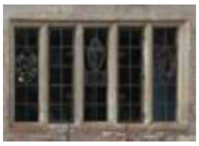
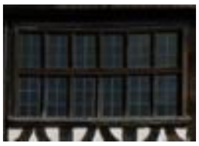

CWW_GO_001

Image cannot be displayed due to copyright
CWW_GO_083

Cluster 4 (7 Examples)


CWW_GO_018


CWW_GO_031


CWW_GO_036

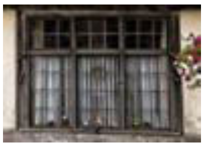
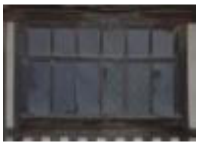

CWW_GO_086

Image cannot be displayed due to copyright
CWW_GO_096


CWW_GO_099

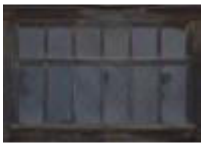
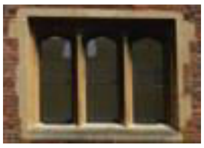
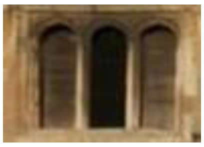
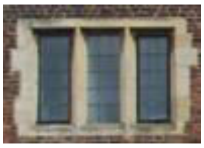

CWW_GO_100

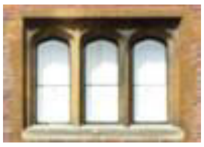
Figure D.6: *k*-means Template Cluster Members – Gothic Style Window 'Width Wide' (1 of 5) See Figure D.7, Figure D.8, Figure D.9 and Figure D.10 for the remaining four figures of the five-figure set.

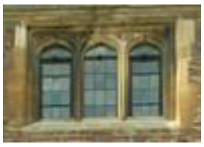
Cluster 5 (5 Examples)


CWW_GO_015

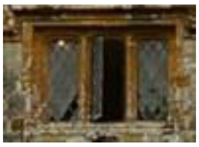

CWW_GO_043


CWW_GO_054


CWW_GO_055


CWW_GO_060

Cluster 6 (9 Examples)


CWW_GO_007

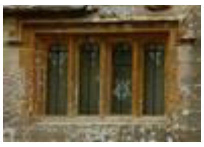
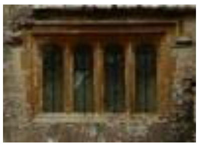
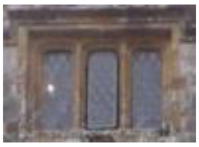
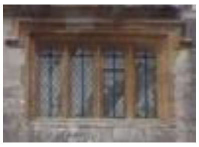

CWW_GO_009

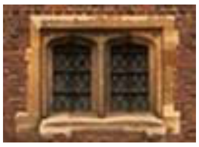
Image cannot be displayed due to copyright

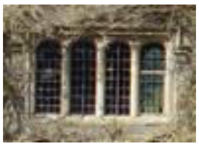
CWW_GO_010

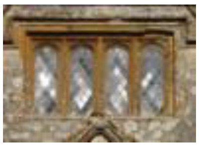

CWW_GO_011


CWW_GO_044


CWW_GO_045


CWW_GO_050


CWW_GO_053


CWW_GO_061

Cluster 7 (6 Examples)

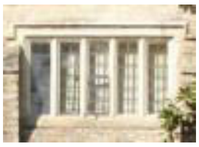
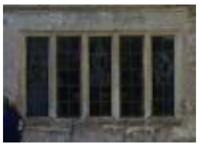
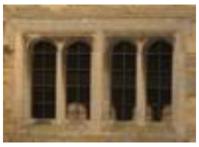
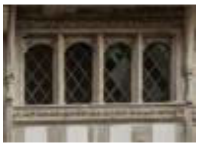

CWW_GO_014

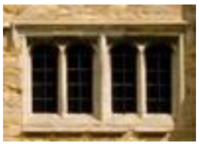
Image cannot be displayed due to copyright

CWW_GO_021

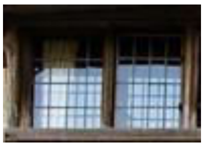

CWW_GO_029


CWW_GO_042


CWW_GO_082


CWW_GO_094

Cluster 8 (2 Examples)


CWW_GO_037

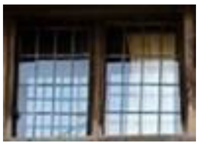

CWW_GO_038

Figure D.7: *k*-means Template Cluster Members – Gothic Style Window 'Width Wide' (2 of 5) See Figure D.6, Figure D.8, Figure D.9 and Figure D.10 for the remaining four figures of the five-figure set.

Cluster 9 (17 Examples)

CWW_GO_008

CWW_GO_012

CWW_GO_039

CWW_GO_046

CWW_GO_062

CWW_GO_063

CWW_GO_064

CWW_GO_065

CWW_GO_066

CWW_GO_067

CWW_GO_080

CWW_GO_081

CWW_GO_084

CWW_GO_085

Image cannot be displayed due to copyright
CWW_GO_087

CWW_GO_088

CWW_GO_093

Cluster 10 (3 Examples)

CWW_GO_035

CWW_GO_058

Image cannot be displayed due to copyright
CWW_GO_077

Cluster 11 (4 Examples)

CWW_GO_002

Image cannot be displayed due to copyright
CWW_GO_048

CWW_GO_049

CWW_GO_089

Figure D.8: *k*-means Template Cluster Members – Gothic Style Window 'Width Wide' (3 of 5) See Figure D.6, Figure D.7, Figure D.9 and Figure D.10 for the remaining four figures of the five-figure set.

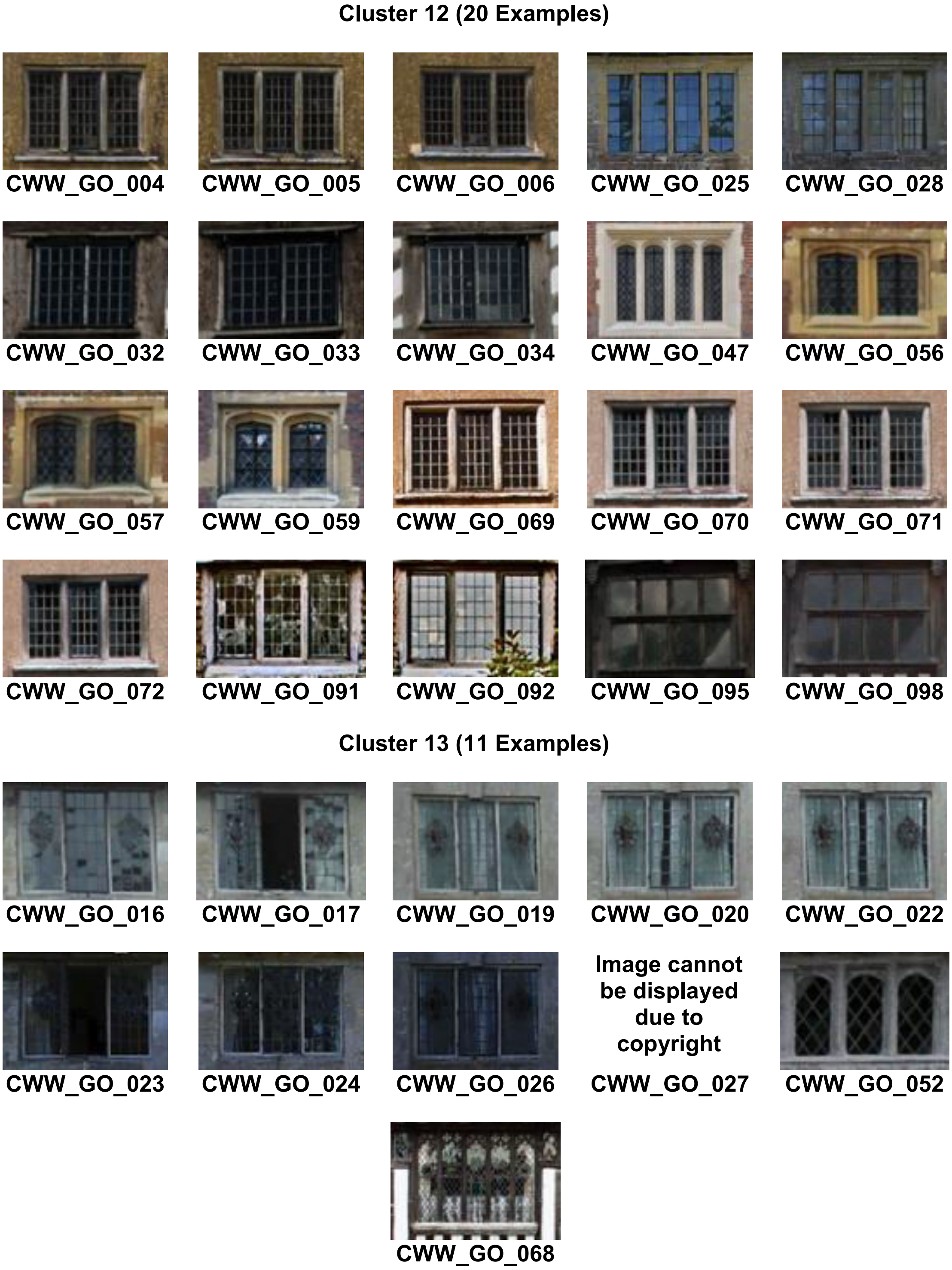


Figure D.9: *k*-means Template Cluster Members – Gothic Style Window ‘Width Wide’ (4 of 5) See Figure D.6, Figure D.7, Figure D.8 and Figure D.10 for the remaining four figures of the five-figure set.

Cluster 14 (1 Example)

Image cannot be displayed due to copyright

CWW_GO_090

Cluster 15 (1 Example)

Image cannot be displayed due to copyright

CWW_GO_003

Figure D.10: *k*-means Template Cluster Members – Gothic Style Window 'Width Wide' (5 of 5)
See Figure D.6, Figure D.7, Figure D.8, Figure D.9 and Figure D.10 the remaining four figures of the five-figure set.

even variations in lighting (which might affect edge coherency). Such distinguishing features do not consistently split the clusters, however i.e. some cluster members break these rules. The use of the HOG descriptor for clustering could explain such inconsistency, i.e. edges picked up by the descriptor could be more or less strong than the same edge perceived by the human eye.

By using the cluster centroid HOG descriptors from *k*-means, a mean F-measure for the Norman 3D building models of 0.04, for class granularity 1, was obtained using the HOG-based template matching approach. When using the exemplar templates from each *k*-means cluster, instead of the cluster centroid HOG descriptors, a marginal improvement in mean F-measure resulted but only to 0.05. In an attempt to improve performance a different approach, affinity propagation clustering (Frey and Dueck, 2007), was then trialled, using the same template set and clustering class splits. Frey and Dueck's method measures similarity between data point pairs while simultaneously checking whether all data points are potential exemplars. They claim both a faster runtime and lower clustering error. Unlike *k*-means, affinity propagation *is* an exemplar method.

The process flow for using affinity propagation was broadly as per that in Figure D.1, with the following three exceptions. (1) '*k*-means' was replaced with affinity propagation. (2) 'Silhouette' was replaced with an equivalent step, one native to the affinity propagation method. (3) The exemplar step was removed. Euclidean distance between points was used as the similarity measure. While Frey and Dueck originally used negative squared Euclidean distance as a similarity measure, they state that the choice of similarity measure is dependent upon the specifics of the clustering problem. As a starting point, Euclidean distance was used here instead. No further detail on the specifics of the process flow for the use of the affinity propagation method is given here – the method was not ultimately used.

Template Class (Granularity 2)	Clustering Class	Num. of Clusters	Num. of Examples in each Cluster
Gothic Style Window	'Width Narrow'	2	70, 30
Gothic Style Window	'Width Medium'	2	38, 62
Gothic Style Window	'Width Wide'	1	100
Norman Style Window	N/A	2	61, 39
	SUBTOTAL	**7**	
Gothic Style Window	'Width Narrow'	2	68, 32
Gothic Style Window	'Width Wide'		
		2	49, 51
Norman Style Door	'Width Narrow'	2	23, 77
Norman Style Door	'Width Wide'	2	70, 30
	SUBTOTAL	**7**	
	TOTAL	**14**	

Table D.2: Number of Template Clusters by Clustering Class (Affinity Propagation) – Trial

The number of cluster centres obtained with affinity propagation across the template set was lower than *k*-means: 14 versus 63. Table D.2 shows the partitioning of the examples, including the number of clusters and the number of examples in each cluster, for each clustering class. Figure D.6 and Figure D.7 show the exemplars for each clustering class. None of the template examples were exemplars resulting from *both* *k*-means and affinity propagation. Nevertheless, there was some commonality for some clustering classes, e.g. the 'Gothic Style Window Width Narrow' and 'Norman Style Door Width Narrow' clustering classes had the same number of cluster centres and some visual similarity in the exemplars.

The mean F-measure at granularity 1 improved when using affinity propagation for clustering: 0.2 versus 0.05. That said, there was one anomalous granularity 1 F-measure result, 0.5 for BM_N_1, though this was still worse than the 0.83 achieved without clustering (see Chapter 7).

The F-measure of 0.2 for affinity propagation is still a poor result though, when one considers that the mean F-measure achieved for the Norman 3D building models (with no SVM) without clustering of templates was 0.67. See Chapter 7 for the results of the HOG-based template matching with no clustering.

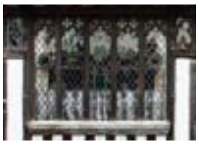
Gothic Style Window 'Width Narrow' (2 Clusters)

Image cannot be displayed due to copyright	**Image cannot be displayed due to copyright**
CWN_GO_005	**CWN_GO_037**

Gothic Style Window 'Width Medium' (2 Clusters)

Image cannot be displayed due to copyright	**Image cannot be displayed due to copyright**
CWM_GO_068	**CWM_GO_101**

Gothic Style Window 'Width Wide' (1 Cluster)



CWW_GO_068

Norman Style Window (2 Clusters)

Image cannot be displayed due to copyright	**Image cannot be displayed due to copyright**
CW_N_079	**CW_N_150**

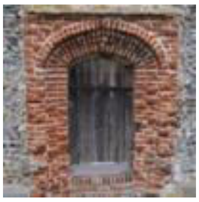
Figure D.11: Gothic & Norman Window Clustering Class Exemplars (Affinity Propagation) – Trial
Note that the relative sizes of the templates shown in Figure D.6 and Figure D.7, and in Figure 3.8, Figure 3.9, Figure 3.10 and Figure 3.11 in Chapter 3 are as used in the methods within this work. Also note: some template examples were rejected prior to clustering, hence references contain numbering above 100.

Gothic Style Window 'Width Narrow' (1 Cluster)

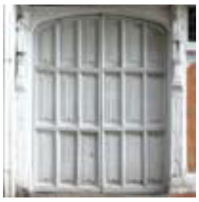
Image cannot be displayed due to copyright

CDN_GO_071

Gothic Style Window 'Width Wide' (2 Clusters)

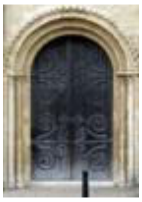


CDW_GO_016

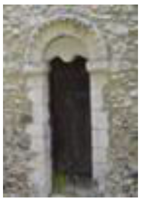


CDW_GO_078

Norman Style Door 'Width Narrow' (2 Clusters)

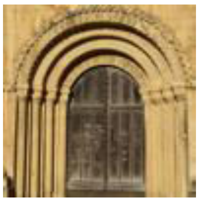


CDN_N_025

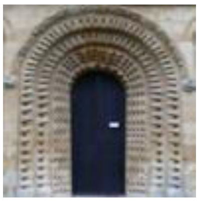


CDN_N_045

Norman Style Door 'Width Wide' (2 Clusters)



CDW_N_007



CDW_N_030

Figure D.12: Gothic & Norman Door Clustering Class Exemplars (Affinity Propagation) – Trial See the caption for Figure D.6.

On a positive note, the time taken for clustering using affinity propagation was quicker, by an order of magnitude: 15 seconds versus 155 seconds. In addition, and as expected, the processing time for HOG-based template matching, with 14 not 63 templates, improved too.

Due to the poor results obtained with clustering, no trial was conducted that combined clustering and the use of the SVM. While the F-measure results with the use of cluster templates from affinity propagation were an improvement on those obtained with *k*-means, they were still not good enough to carry the method forward in this work. For this reason, template examples in each affinity propagation cluster for a clustering class are not shown here. The equivalent example for *k*-means is intended to give an indication here of how clustering in general operated on the templates in this work.

The likely causes of the poor F-measure obtained with clustering could be as follows. (1) The template examples were not sufficiently representative of the instances of windows or doors found on the 3D building models. The poor result may have been compounded by any inconsistency between the visual appearance of a template and its HOG descriptor, i.e. the HOG descriptor may accentuate or lessen the coherency of lines compared to the apparent visual appearance. Note that such an inconsistency would not be unique to the use of templates resulting from clustering. (2) Non-exemplar HOG descriptors, i.e. from *k*-means, might not be sufficiently similar to HOG descriptors for real-world window or door instances. (3) The presence of large amounts of surrounding brickwork at the periphery of some template examples (more than the 'not-cropped' templates used for the non-clustering approach) may have added edges, in those regions, which were less representative of window or door instances found on the 3D building models. The existence of large expanses of surround for some template examples was a result of the requirement for each template example in a clustering class to have the same pixel dimensions. If a candidate match of the correct class *had* been achieved with such a template, then it could be judged as a false positive due a failure of the overlap criterion used for calculating F-measure.

D.3 Summary

This appendix has detailed the method used for the trial of clustering of the HOG descriptors of templates. The intention was that the use of HOG descriptors representing the cluster centres as input to HOG-based template matching might improve the results of HOG-based template matching as a result of a better selection of templates, and also reduce runtime if fewer templates resulted from clustering. However, the outcome of the template clustering trial was that the approach would not be taken forward, and that template images would continue to be used as input to HOG-based template matching. Nonetheless, the trial gave some insights into how the HOG descriptor represents different designs within templates, i.e. from the manner in which clustering chose to partition the set of templates. In the next chapter, machine learning is applied to HOG-based template matching itself, specifically with the use an SVM with the intention of achieving parity, or ideally bettering, the results obtained without an SVM. If successful then the overall method in this work would become more scalable to unseen 3D building models, and potentially to new architectural styles, provided enough suitable data was available.

Appendix E Extended Experiments – F-measure Comparisons – Heuristics

See also Table E.2, Table E.3 and Table E.4

C20 Style 3D Building Model	Heuristic (H)	**(a) F-measures**				**(b) ∆F-measure ‘Hs’ vs ‘No Hs’ (%)**				**(c) ∆F-measure ‘Hs’ vs ‘No Hs’ (Abs.)**			
		Template Set (i)		**Template Set (ii)**		**Template Set (i)**		**Template Set (ii)**		**Template Set (i)**		**Template Set (ii)**	
		G1	**G2**	**G1**	**G2**	**G1**	**G2**	**G1**	**G2**	**G1**	**G2**	**G1**	**G2**
BM_2_1	No Hs	0.01	0.01	0.03	0.03								
	H1 only	0.03	0.01	0.05	0.05	**200%**	0%	**67%**	**67%**	**0.02**	0.00	**0.02**	**0.02**
	H2 only	0.03	**0.03**	0.06	0.06	**200%**	**200%**	**100%**	**100%**	**0.02**	**0.02**	**0.03**	**0.03**
	H3 only	0.01	0.01	0.03	0.03	0%	0%	0%	0%	0.00	0.00	0.00	0.00
	All Hs	**0.05**	**0.03**	**0.07**	**0.07**	**400%**	**200%**	**133%**	**133%**	**0.04**	**0.02**	**0.04**	**0.04**
BM_2_2	No Hs	0.03	0.03	0.27	0.27								
	H1 only	0.08	0.08	**0.5**	**0.5**	**167%**	**167%**	**85%**	**85%**	**0.05**	**0.05**	**0.23**	**0.23**
	H2 only	0.08	0.08	0.28	0.28	**167%**	**167%**	**4%**	**4%**	**0.05**	**0.05**	**0.01**	**0.01**
	H3 only	0.03	0.03	0.27	0.27	0%	0%	0%	0%	0.00	0.00	0.00	0.00
	All Hs	**0.16**	**0.16**	**0.5**	**0.5**	**433%**	**433%**	**85%**	**85%**	**0.13**	**0.13**	**0.23**	**0.23**
BM_2_3	No Hs	0.23	0.19	0.3	0.3								
	H1 only	0.42	0.37	0.55	0.55	**83%**	**95%**	**83%**	**83%**	**0.19**	**0.18**	**0.25**	**0.25**
	H2 only	0.28	0.22	0.32	0.32	**22%**	**16%**	**7%**	**7%**	**0.05**	**0.03**	**0.02**	**0.02**
	H3 only	0.23	0.19	0.3	0.3	0%	0%	0%	0%	0.00	0.00	0.00	0.00
	All Hs	**0.49**	**0.43**	**0.56**	**0.56**	**113%**	**126%**	**87%**	**87%**	**0.26**	**0.24**	**0.26**	**0.26**

KEY
Per building per granularity (G1, G2) **bold** F-measure is the highest

KEY
Green text: △ > 0 Black text: △ = 0 **Red** text: △ < 0

Table E.1: HOG-based Template Matching (no SVM) – Heuristics Experimental Results – C20 Style 3D Building Models (Table 1 of 4)
Values are provided for running the method with template set (i), and with template set (ii). Each of those sets of results provides values at each class granularity i.e. G1, G2, G3. G1 and G2 for C20 and Georgian-Regency 3D building models are the same (only one style of template was used). There is no G3 for the C20 or Norman style 3D building models – see Chapter 3. Sub-table (*a*) contains F-measure results (vertically) per building and per granularity. The highest F-measure for a per-building, per-granularity set is highlighted in **bold**. Sub-tables (*b*) and (*c*) show the delta (△) in F-measure versus the use of no heuristics, in percent and as an absolute value respectively.

See also Table E.1,
Table E.3 and Table E.4

		(a) F-measures				(b) ∆F-measure 'Hs' vs 'No Hs' (%)				(c) ∆F-measure 'Hs' vs 'No Hs' (Abs.)			
C20 Style 3D Building Model	**Heuristic (H)**	**Template Set (i)**		**Template Set (ii)**		**Template Set (i)**		**Template Set (ii)**		**Template Set (i)**		**Template Set (ii)**	
		G1	**G2**	**G1**	**G2**	**G1**	**G2**	**G1**	**G2**	**G1**	**G2**	**G1**	**G2**
BM_2_4	No Hs	**0.26**	0.09	**0.11**	**0.11**								
	H1 only	**0.26**	**0.17**	**0.11**	**0.11**	0%	**89%**	0%	0%	0.00	**0.08**	0.00	0.00
	H2 only	**0.26**	0.09	**0.11**	**0.11**	0%	0%	0%	0%	0.00	0.00	0.00	0.00
	H3 only	**0.26**	0.09	**0.11**	**0.11**	0%	0%	0%	0%	0.00	0.00	0.00	0.00
	All Hs	**0.26**	**0.17**	**0.11**	**0.11**	0%	**89%**	0%	0%	0.00	**0.08**	0.00	0.00
BM_2_5	No Hs	0.38	0.38	0.43	0.43								
	H1 only	0.71	0.71	**0.8**	**0.8**	**87%**	**87%**	**86%**	**86%**	**0.33**	**0.33**	**0.37**	**0.37**
	H2 only	0.43	0.43	0.43	0.43	**13%**	**13%**	0%	0%	**0.05**	**0.05**	0.00	0.00
	H3 only	0.38	0.38	0.43	0.43	0%	0%	0%	0%	0.00	0.00	0.00	0.00
	All Hs	**0.8**	**0.8**	**0.8**	**0.8**	**111%**	**111%**	**86%**	**86%**	**0.42**	**0.42**	**0.37**	**0.37**
BM_2_6	No Hs	0.66	0.21	0.45	0.45								
	H1 only	0.75	0.24	0.57	0.57	**14%**	**14%**	**27%**	**27%**	**0.09**	**0.03**	**0.12**	**0.12**
	H2 only	0.69	0.22	0.45	0.45	**5%**	**5%**	0%	0%	**0.03**	**0.01**	0.00	0.00
	H3 only	0.66	0.21	0.45	0.45	0%	0%	0%	0%	0.00	0.00	0.00	0.00
	All Hs	**0.77**	**0.25**	**0.58**	**0.58**	**17%**	**19%**	**29%**	**29%**	**0.11**	**0.04**	**0.13**	**0.13**

KEY
Per building per granularity (G1, G2) **bold** F-measure is the highest

KEY
Green text: ∆ > 0 Black text: ∆ = 0 **Red** text: ∆ < 0

Table E.2: HOG-based Template Matching (no SVM) – Heuristics Experimental Results – C20 Style 3D Building Models (Table 2 of 4)
See caption for Table E.1.

See also Table E.1,
Table E.2 and Table E.4

		(a) **F-measures**				(b) **∆F-measure 'Hs' vs 'No Hs' (%)**				(c) **∆F-measure 'Hs' vs 'No Hs' (Abs.)**			
C20 Style 3D Building Model	**Heuristic (H)**	**Template Set (i)**		**Template Set (ii)**		**Template Set (i)**		**Template Set (ii)**		**Template Set (i)**		**Template Set (ii)**	
		G1	**G2**	**G1**	**G2**	**G1**	**G2**	**G1**	**G2**	**G1**	**G2**	**G1**	**G2**
BM_2_7	No Hs	0.16	0.16	0.53	0.53								
	H1 only	0.34	0.34	**0.58**	**0.58**	**113%**	**113%**	**9%**	**9%**	**0.18**	**0.18**	**0.05**	**0.05**
	H2 only	0.2	0.2	0.53	0.53	**25%**	**25%**	0%	0%	**0.04**	**0.04**	0.00	0.00
	H3 only	0.16	0.16	0.53	0.53	0%	0%	0%	0%	0.00	0.00	0.00	0.00
	All Hs	**0.37**	**0.37**	**0.58**	**0.58**	**131%**	**131%**	**9%**	**9%**	**0.21**	**0.21**	**0.05**	**0.05**
BM_2_8	No Hs	**0.65**	**0.38**	0.48	0.48								
	H1 only	0.5	0.32	0.58	0.58	**-23%**	**-16%**	**21%**	**21%**	**-0.15**	**-0.06**	**0.10**	**0.10**
	H2 only	**0.65**	**0.38**	0.48	0.48	0%	0%	0%	0%	0.00	0.00	0.00	0.00
	H3 only	**0.65**	**0.38**	0.49	0.49	0%	0%	**2%**	**2%**	0.00	0.00	**0.01**	**0.01**
	All Hs	0.5	0.32	**0.6**	**0.6**	**-23%**	**-16%**	**25%**	**25%**	**-0.15**	**-0.06**	**0.12**	**0.12**
BM_2_9	No Hs	0.22	0.22	0.37	0.37								
	H1 only	0.32	0.28	0.57	0.57	**45%**	**27%**	**54%**	**54%**	**0.10**	**0.06**	**0.20**	**0.20**
	H2 only	0.44	0.44	0.52	0.52	**100%**	**100%**	**41%**	**41%**	**0.22**	**0.22**	**0.15**	**0.15**
	H3 only	0.23	0.23	0.37	0.37	**5%**	**5%**	0%	0%	**0.01**	**0.01**	0.00	0.00
	All Hs	**0.62**	**0.54**	**0.7**	**0.7**	**182%**	**145%**	**89%**	**89%**	**0.40**	**0.32**	**0.33**	**0.33**

KEY
Per building per granularity (G1, G2) **bold** F-measure is the highest

KEY
Green text: ∆ > 0 Black text: ∆ = 0 **Red** text: ∆ < 0

Table E.3: HOG-based Template Matching (no SVM) – Heuristics Experimental Results – C20 Style 3D Building Models (Table 3 of 4)
See caption for Table E.1.

See also Table E.1, Table E.2 and Table E.3

(a)

F-measures

C20 Style 3D Building Model	**Heuristic (H)**	**Template Set (i)**		**Template Set (ii)**	
		G1	**G2**	**G1**	**G2**
BM_2_10	No Hs	0.01	0.01	0.04	0.04
	H1 only	0.08	0.08	**0.44**	**0.44**
	H2 only	0.07	0.07	0.07	0.07
	H3 only	0.01	0.01	0.04	0.04
	All Hs	**0.25**	**0.25**	0.26	0.26
BM_2_11	No Hs	0.01	0.01	0.07	0.07
	H1 only	0.03	0.03	0.67	0.67
	H2 only	0.02	0.02	0.11	0.11
	H3 only	0.01	0.01	0.07	0.07
	All Hs	**0.06**	**0.05**	**0.69**	**0.69**
BM_2_12	No Hs	0.02	0.02	0.26	0.26
	H1 only	0.04	0.04	0.76	0.76
	H2 only	0.05	0.05	0.48	0.48
	H3 only	0.02	0.02	0.26	0.26
	All Hs	**0.08**	**0.08**	**0.84**	**0.84**

(b)

△F-measure ‘Hs’ vs ‘No Hs’ (%)

C20 Style 3D Building Model	**Heuristic (H)**	**Template Set (i)**		**Template Set (ii)**	
		G1	**G2**	**G1**	**G2**
BM_2_10	No Hs				
	H1 only	**700%**	**700%**	**1000%**	**1000%**
	H2 only	**600%**	**600%**	**75%**	**75%**
	H3 only	0%	0%	0%	0%
	All Hs	**2400%**	**2400%**	**550%**	**550%**
BM_2_11	No Hs				
	H1 only	**200%**	**200%**	**857%**	**857%**
	H2 only	**100%**	**100%**	**57%**	**57%**
	H3 only	0%	0%	0%	0%
	All Hs	**500%**	**400%**	**886%**	**886%**
BM_2_12	No Hs				
	H1 only	**100%**	**100%**	**192%**	**192%**
	H2 only	**150%**	**150%**	**85%**	**85%**
	H3 only	0%	0%	0%	0%
	All Hs	**300%**	**300%**	**223%**	**223%**

(c)

△F-measure ‘Hs’ vs ‘No Hs’ (Abs.)

C20 Style 3D Building Model	**Heuristic (H)**	**Template Set (i)**		**Template Set (ii)**	
		G1	**G2**	**G1**	**G2**
BM_2_10	No Hs				
	H1 only	**0.07**	**0.07**	**0.40**	**0.40**
	H2 only	**0.06**	**0.06**	**0.03**	**0.03**
	H3 only	0.00	0.00	0.00	0.00
	All Hs	**0.24**	**0.24**	**0.22**	**0.22**
BM_2_11	No Hs				
	H1 only	**0.02**	**0.02**	**0.60**	**0.60**
	H2 only	**0.01**	**0.01**	**0.04**	**0.04**
	H3 only	0.00	0.00	0.00	0.00
	All Hs	**0.05**	**0.04**	**0.62**	**0.62**
BM_2_12	No Hs				
	H1 only	**0.02**	**0.02**	**0.50**	**0.50**
	H2 only	**0.03**	**0.03**	**0.22**	**0.22**
	H3 only	0.00	0.00	0.00	0.00
	All Hs	**0.06**	**0.06**	**0.58**	**0.58**

KEY
Per building per granularity (G1, G2) **bold** F-measure is the highest

KEY
Green text: △ > 0 Black text: △ = 0 **Red** text: △ < 0

Table E.4: HOG-based Template Matching (no SVM) – Heuristics Experimental Results – C20 Style 3D Building Models (Table 4 of 4)
See caption for Table E.1.

See also Table E.6 and Table E.7		(a) F-measures					(b) ∆F-measure 'Hs' vs 'No Hs' (%)					(c) ∆F-measure 'Hs' vs 'No Hs' (Abs.)				
Georgian-Regency Style 3D Building Model	**Heuristic (H)**	**Template Set (i)**			**Template Set (ii)**		**Template Set (i)**			**Template Set (ii)**		**Template Set (i)**			**Template Set (ii)**	
		G1	**G2**	**G3**	**G1 & G2**	**G3**	**G1**	**G2**	**G3**	**G1 & G2**	**G3**	**G1**	**G2**	**G3**	**G1 & G2**	**G3**
BM_GR_1	No Hs	0.09	0.09	0.09	0.47	0.47										
	H1 only	0.24	0.24	0.12	0.91	0.47	**167%**	**167%**	**33%**	**94%**	0%	**0.15**	**0.15**	**0.03**	**0.44**	0.00
	H2 only	0.11	0.11	0.11	0.48	**0.48**	**22%**	**22%**	**22%**	**2%**	**2%**	**0.02**	**0.02**	**0.02**	**0.01**	**0.01**
	H3 only	0.1	0.1	0.1	0.47	0.47	**11%**	**11%**	**11%**	0%	0%	**0.01**	**0.01**	**0.01**	0.00	0.00
	All Hs	**0.32**	**0.32**	**0.16**	**0.93**	**0.48**	**256%**	**256%**	**78%**	**98%**	**2%**	**0.23**	**0.23**	**0.07**	**0.46**	**0.01**
BM_GR_2	No Hs	0.63	0.49	0.36	0.83	**0.52**										
	H1 only	0.76	0.72	0.41	**0.87**	0.46	**21%**	**47%**	**14%**	**5%**	**-12%**	**0.13**	**0.23**	**0.05**	**0.04**	**-0.06**
	H2 only	0.63	0.49	0.36	0.83	**0.52**	0%	0%	0%	0%	0%	0.00	0.00	0.00	0.00	0.00
	H3 only	0.63	0.49	0.36	0.83	**0.52**	0%	0%	0%	0%	0%	0.00	0.00	0.00	0.00	0.00
	All Hs	**0.78**	**0.74**	**0.42**	**0.87**	0.46	**24%**	**51%**	**17%**	**5%**	**-12%**	**0.15**	**0.25**	**0.06**	**0.04**	**-0.06**
BM_GR_3	No Hs	0.85	0.85	0.24	0.91	0.26										
	H1 only	0.85	0.84	0.35	0.87	**0.37**	0%	**-1%**	**46%**	**-4%**	**42%**	0.00	**-0.01**	**0.11**	**-0.04**	**0.11**
	H2 only	**0.87**	**0.87**	0.24	**0.92**	0.26	**2%**	**2%**	0%	**1%**	0%	**0.02**	**0.02**	0.00	**0.01**	0.00
	H3 only	0.86	0.86	0.24	0.91	0.26	**1%**	**1%**	0%	0%	0%	**0.01**	**0.01**	0.00	0.00	0.00
	All Hs	0.86	0.86	**0.36**	0.87	**0.37**	**1%**	**1%**	**50%**	**-4%**	**42%**	**0.01**	**0.01**	**0.12**	**-0.04**	**0.11**

KEY
Per building per granularity (G1, G2, G3) **bold** F-measure is highest

KEY
Green text: ∆ > 0 Black text: ∆ = 0 **Red** text: ∆ < 0

Table E.5: HOG-based Template Matching (no SVM) – Heuristics Experimental Results – Georgian-Regency Style 3D Building Models (Table 1 of 3)
See caption for Table E.1.

See also Table E.5
and Table E.7

		(a) F-measures					**(b) ∆F-measure 'Hs' vs 'No Hs' (%)**					**(c) ∆F-measure 'Hs' vs 'No Hs' (Abs.)**				
Georgian-Regency Style 3D Building Model	**Heuristic (H)**	**Template Set (i)**			**Template Set (ii)**		**Template Set (i)**			**Template Set (ii)**		**Template Set (i)**			**Template Set (ii)**	
		G1	**G2**	**G3**	**G1 & G2**	**G3**	**G1**	**G2**	**G3**	**G1 & G2**	**G3**	**G1**	**G2**	**G3**	**G1 & G2**	**G3**
BM_GR_4	No Hs	0.55	0.55	0.37	0.64	0.44										
	H1 only	0.81	0.81	0.59	**0.92**	**0.67**	**47%**	**47%**	**59%**	**44%**	**52%**	**0.26**	**0.26**	**0.22**	**0.28**	**0.23**
	H2 only	0.61	0.61	0.41	0.64	0.44	**11%**	**11%**	**11%**	0%	0%	**0.06**	**0.06**	**0.04**	0.00	0.00
	H3 only	0.59	0.59	0.4	0.64	0.44	**7%**	**7%**	**8%**	0%	0%	**0.04**	**0.04**	**0.03**	0.00	0.00
	All Hs	**0.9**	**0.9**	**0.65**	**0.92**	**0.67**	**64%**	**64%**	**76%**	**44%**	**52%**	**0.35**	**0.35**	**0.28**	**0.28**	**0.23**
BM_GR_5	No Hs	0.05	0.04	0.02	0.65	**0.27**										
	H1 only	0.08	0.06	0.02	**0.67**	0.23	**60%**	**50%**	0%	**3%**	**-15%**	**0.03**	**0.02**	0.00	**0.02**	**-0.04**
	H2 only	0.12	0.09	0.04	0.65	**0.27**	**140%**	**125%**	**100%**	0%	0%	**0.07**	**0.05**	**0.02**	0.00	0.00
	H3 only	0.12	0.09	0.04	0.65	**0.27**	**140%**	**125%**	**100%**	0%	0%	**0.07**	**0.05**	**0.02**	0.00	0.00
	All Hs	**0.2**	**0.15**	**0.06**	**0.67**	0.23	**300%**	**275%**	**200%**	**3%**	**-15%**	**0.15**	**0.11**	**0.04**	**0.02**	**-0.04**
BM_GR_6	No Hs	0.77	0.77	0.23	0.91	0.28										
	H1 only	**0.79**	**0.79**	**0.53**	**0.96**	**0.64**	**3%**	**3%**	**130%**	**5%**	**129%**	**0.02**	**0.02**	**0.30**	**0.05**	**0.36**
	H2 only	0.77	0.77	0.23	0.91	0.28	0%	0%	0%	0%	0%	0.00	0.00	0.00	0.00	0.00
	H3 only	0.77	0.77	0.23	0.91	0.28	0%	0%	0%	0%	0%	0.00	0.00	0.00	0.00	0.00
	All Hs	**0.79**	**0.79**	**0.53**	**0.96**	**0.64**	**3%**	**3%**	**130%**	**5%**	**129%**	**0.02**	**0.02**	**0.30**	**0.05**	**0.36**

KEY
Per building per granularity (G1, G2, G3) **bold** F-measure is highest

KEY
Green text: ∆ > 0 Black text: ∆ = 0 **Red** text: ∆ < 0

Table E.6: HOG-based Template Matching (no SVM) – Heuristics Experimental Results – Georgian-Regency Style 3D Building Models (Table 2 of 3) See caption for Table E.1.

See also Table E.5 and Table E.6

		(a) F-measures					(b) △F-measure ‘Hs’ vs ‘No Hs’ (%)					(c) △F-measure ‘Hs’ vs ‘No Hs’ (Abs.)				
Georgian-Regency Style 3D Building Model	**Heuristic (H)**	**Template Set (i)**			**Template Set (ii)**		**Template Set (i)**			**Template Set (ii)**		**Template Set (i)**			**Template Set (ii)**	
		G1	**G2**	**G3**	**G1 & G2**	**G3**	**G1**	**G2**	**G3**	**G1 & G2**	**G3**	**G1**	**G2**	**G3**	**G1 & G2**	**G3**
BM_GR_7	No Hs	0.35	0.35	0.26	**0.78**	**0.61**										
	H1 only	0.45	0.45	0.24	0.77	0.46	**29%**	**29%**	**-8%**	**-1%**	**-25%**	**0.10**	**0.10**	**-0.02**	**-0.01**	**-0.15**
	H2 only	0.47	0.47	**0.35**	**0.78**	**0.61**	**34%**	**34%**	**35%**	0%	0%	**0.12**	**0.12**	**0.09**	0.00	0.00
	H3 only	0.45	0.45	0.34	**0.78**	**0.61**	**29%**	**29%**	**31%**	0%	0%	**0.10**	**0.10**	**0.08**	0.00	0.00
	All Hs	**0.57**	**0.57**	0.29	0.77	0.46	**63%**	**63%**	**12%**	**-1%**	**-25%**	**0.22**	**0.22**	**0.03**	**-0.01**	**-0.15**

KEY
Per building per granularity (G1, G2, G3) **bold** F-measure is highest

KEY
Green text: △ > 0 Black text: △ = 0 **Red** text: △ < 0

Table E.7: HOG-based Template Matching (no SVM) – Heuristics Experimental Results – Georgian-Regency Style 3D Building Models (Table 3 of 3)
See caption for Table E.1.

See also Table E.9

Norman Style 3D Building Model	Heuristic (H)	(a) F-measures				(b) ∆F-measure 'Hs' vs 'No Hs' (%)				(c) ∆F-measure 'Hs' vs 'No Hs' (Abs.)			
		Template Set (i)		Template Set (ii)		Template Set (i)		Template Set (ii)		Template Set (i)		Template Set (ii)	
		G1	G2	G1	G2	G1	G2	G1	G2	G1	G2	G1	G2
BM_N_1	No Hs	0.1	0.02	0.1	0.02								
	H1 only	0.41	0.3	0.41	0.3	310%	1400%	310%	1400%	0.31	0.28	0.31	0.28
	H2 only	0.42	0.11	0.42	0.11	320%	450%	320%	450%	0.32	0.09	0.32	0.09
	H3 only	0.11	0	0.11	0	10%	-100%	10%	-100%	0.01	-0.02	0.01	-0.02
	All Hs	**0.83**	**0.58**	**0.83**	**0.58**	730%	2800%	730%	2800%	0.73	0.56	0.73	0.56
BM_N_2	No Hs	0.31	0.22	0.31	0.22								
	H1 only	0.48	0.38	0.48	0.38	55%	73%	55%	73%	0.17	0.16	0.17	0.16
	H2 only	0.48	0.34	0.48	0.34	55%	55%	55%	55%	0.17	0.12	0.17	0.12
	H3 only	0.31	0.22	0.31	0.22	0%	0%	0%	0%	0.00	0.00	0.00	0.00
	All Hs	**0.69**	**0.55**	**0.69**	**0.55**	123%	150%	123%	150%	0.38	0.33	0.38	0.33
BM_N_3	No Hs	0.51	0.29	0.5	0.29								
	H1 only	0.63	0.41	0.62	0.4	24%	41%	24%	38%	0.12	0.12	0.12	0.11
	H2 only	0.6	0.34	0.58	0.33	18%	17%	16%	14%	0.09	0.05	0.08	0.04
	H3 only	0.52	0.3	0.51	0.29	2%	3%	2%	0%	0.01	0.01	0.01	0.00
	All Hs	**0.72**	**0.47**	**0.71**	**0.46**	41%	62%	42%	59%	0.21	0.18	0.21	0.17

KEY
Per building per granularity (G1, G2) **bold** F-measure is the highest

KEY
Green text: ∆ > 0 Black text: ∆ = 0 **Red** text: ∆ < 0

Table E.8: HOG-based Template Matching (no SVM) – Heuristics Experimental Results – Norman Style 3D Building Models (Table 1 of 2)
See caption for Table E.1.

See also Table E.8

		(a) F-measures				(b) △F-measure 'Hs' vs 'No Hs' (%)				(c) △F-measure 'Hs' vs 'No Hs' (Abs.)			
Norman Style 3D Building Model	Heuristic (H)	Template Set (i)		Template Set (ii)		Template Set (i)		Template Set (ii)		Template Set (i)		Template Set (ii)	
		G1	G2	G1	G2	G1	G2	G1	G2	G1	G2	G1	G2
BM_N_4	No Hs	0.15	0.11	0.26	0.21								
	H1 only	0.42	0.29	0.67	0.48	180%	164%	158%	129%	0.27	0.18	0.41	0.27
	H2 only	0.2	0.15	0.38	0.31	33%	36%	46%	48%	0.05	0.04	0.12	0.10
	H3 only	0.19	0.14	0.36	0.29	27%	27%	38%	38%	0.04	0.03	0.10	0.08
	All Hs	**0.5**	**0.35**	**0.88**	**0.64**	233%	218%	238%	205%	0.35	0.24	0.62	0.43
BM_N_5	No Hs	0	0	0.05	0								
	H1 only	0.1	0.03	0.21	0.05	N/A	N/A	320%	N/A	0.10	0.03	0.16	0.05
	H2 only	0	0	0.06	0	N/A	N/A	20%	N/A	0.00	0.00	0.01	0.00
	H3 only	0	0	0.07	0	N/A	N/A	40%	N/A	0.00	0.00	0.02	0.00
	All Hs	**0.11**	**0.04**	**0.26**	**0.06**	N/A	N/A	420%	N/A	0.11	0.04	0.21	0.06

KEY
Per building per granularity (G1, G2) **bold** F-measure is the highest

KEY
Green text: △ > 0 Black text: △ = 0 **Red** text: △ < 0

Table E.9: HOG-based Template Matching (no SVM) – Heuristics Experimental Results – Norman Style 3D Building Models (Table 2 of 2)
See caption for Table E.1.

Appendix F Extended Experiments– Extended Heuristics Results Summary

F.1 Extended Results Summary – Heuristic 1

Based on mean delta, H1 was the most effective heuristic: the mean delta for all styles of 3D building model was 48%, or 0.15. H1 was generally most successful for the C20 style.

For C20 style 3D building models, H1 was the most successful heuristic, generally providing the best improvement in F-measure for 10 out of the 12 3D building models. The highest percentage delta achieved was 1,000% for BM_2_10 (0.4, template set (ii), G1 and G2), and the highest absolute delta was 0.6, for BM_2_11 (857%, template set (ii), G1 and G2). BM_2_8 resulted in a negative delta for template set (i), G1 and G2, though the mean delta was only -0.11 (-19%).

When used on Georgian-Regency 3D building models, H1 was the most successful heuristic, based on mean F-measure per 3D building model, for five out of seven 3D building models. BM_GR_1 provided the highest absolute delta: 0.44 (94%, template set (i), G1 and G2). H1 resulted in a degradation in F-measure for three buildings, though aside from a 0.15 (25%) drop for BM_GR_7 (template set (i), G3) the reductions were small.

Based on mean F-measure per 3D building model, H1 was the strongest heuristic for all five Norman 3D building models. BM_N_1 resulted in the highest positive percentage increase for any style: 1,400% (0.28, for both sets of G2 results). BM_N_4 had the highest absolute increase for the Norman style: 0.41 (158%, templates set (ii), G1). Apart from BM_N_1 (which saw a reduction of only 0.02 for template sets (i) and (ii) for G2), H1 always achieved a positive delta among the Norman style 3D building models.

F.2 Extended Results Summary – Heuristic 2

H2 was the second most successful heuristic, based on mean F-measures, with a mean absolute delta of 0.05, and a mean percentage delta of 18%. H2 was generally most successful for the C20 style. H2 did not result in any negative deltas across any of the 24 3D building models.

For the C20 style, H2 was the second most successful heuristic, based on mean F-measure, resulting in the highest mean F-measure per 3D building model for 2 out of 12 3D building models. The highest percentage delta was for BM_2_10 (600%, 0.06, template set (i), G2), and the highest absolute delta was for BM_2_9 (0.22, template set (i), G1 and G2).

For the Georgian-Regency style, H2 was the equal second most successful heuristic, based on mean F-measure per 3D building model. It resulted in the highest mean F-measure for two of the seven 3D building models, and the second highest for three of seven. BM_GR_5 provided the highest percentage delta: 140% (template set (i), G1). This improvement was small in absolute terms, though (0.07). Box House resulted in the highest absolute delta: 0.12 (34%, template set (i), G1 and G2).

For Norman buildings, H2 was also the second most successful heuristic, based on mean F-measure per 3D building model, obtaining the second highest percentage delta for four out of five buildings. The highest percentage delta was 450% for BM_N_1 (0.09, templates set (i), G2). In absolute terms the largest positive delta was 0.32, also for BM_N_1 (320%, template set (i), G1).

F.3 Extended Results Summary – Heuristic 3

H3 was, based on mean F-measure, the least successful of the heuristics: using the same measure it was not the strongest heuristic for any of the styles. The mean absolute delta obtained with H3 was 0.01 (3%).

When used on C20 style 3D building models, H3, using mean F-measure per building, only resulted in any delta for two 3D building models, BM_2_8 and BM_2_9, where the mean for the two buildings was 0.01.

Used in isolation on Georgian-Regency buildings, H3 was the least successful of the three heuristics, though based on mean F-measure per building, it was the second or equal second best heuristic for four out of seven.

On Norman buildings H3 was also the weakest heuristic, based on mean F-measure per 3D building model, placing third for all five 3D building models. That said, it still improved some of the F-measures for four out of five buildings. The highest absolute delta was 0.1, for BM_N_4 (38%, template set (ii), G1 and G2). For BM_N_1, H3 resulted in a small negative absolute delta: -0.02 (-100%, for each set of templates at G2), but once again these are negligible considering that a good result tends to 1. Aside from these drops, H3 did not degrade the results.

F.4 Extended Results Summary – All Heuristics

Based on mean F-measure, comparing the results of (1) using each of the heuristics individually, with the results of (2) using all three heuristics together, (2) was the most successful, with a mean F-measure of 0.22 (versus 0.15, 0.05 and 0.01 for the sole use of each of H1, H2 and H3 respectively). Compared to (1), (2) was also the most successful for each style.

For the C20 style, the use of all three heuristics together gave the highest mean F-measure for 9 out of 12 of the 3D building models. The highest percentage delta was obtained with BM_2_10: 2,400% (0.24, template set (i), G1 and G2) and the highest absolute delta was for BM_2_5 (0.42, 111%, template set (i), G1 and G2). For BM_2_8 negative deltas resulted: -0.15 (-23%) and -0.06 (-16%), both for template set (i), for G1 and G2 respectively. That, said, bearing in mind the overall improvement in F-measure these negative deltas were somewhat rare.

On Georgian-Regency 3D building models, the use of all three heuristics together resulted in the highest mean F-measure per building in five out of seven cases (and joint second for the other two). Amongst the Georgian-Regency 3D building models, the highest percentage delta was 300% (0.15, BM_GR_5, templates set (i), G1) and the highest absolute positive delta was 0.36 (129%, BM_GR_6, templates set (ii), G3). Using all three heuristics together did result in some negative deltas but, generally, these were small (ranging from -0.01 to -0.06, or -1% to -15%). The exception to this was BM_2_7 which saw a drop of 0.15 (-25%, template set (ii), G3), though as a G3 result this was arguably the most demanding of the metrics used in this work.

The combination of all heuristics was the most successful for the Norman style, with a mean delta of 0.34, compared to 0.2 and 0.16 respectively for C20 and Georgian-Regency. The largest positive percentage delta was for BM_N_1: 2,800% (0.56, both template sets, G2). The largest absolute delta was for the same building: 0.73 (730%, template set (i), G1). Absolute delta values for all results for three of the 3D building models were consistently above 0.25.

Appendix G Extended Experiments – F-measure Comparisons – SVM vs No SVM

See also Table G.2

(a)
F-measures

C20 Style 3D Building Model	**Use SVM?**	**Template Set (i)**		**Template Set (ii)**	
		G1	**G2**	**G1**	**G2**
BM_2_1	No SVM	**0.05**	0.03	0.07	0.07
	SVM	0.04	**0.04**	**0.2**	**0.2**
BM_2_2	No SVM	0.16	0.16	**0.5**	**0.5**
	SVM	**0.52**	**0.48**	0.42	0.42
BM_2_3	No SVM	0.49	**0.43**	0.56	0.56
	SVM	**0.52**	0.37	**0.67**	**0.67**
BM_2_4	No SVM	0.26	0.17	0.11	0.11
	SVM	**0.76**	**0.67**	**0.22**	**0.22**
BM_2_5	No SVM	**0.8**	**0.8**	**0.8**	**0.8**
	SVM	0.36	0.36	0.36	0.36
BM_2_6	No SVM	**0.77**	0.25	**0.58**	**0.58**
	SVM	0.62	**0.62**	0.57	0.57

(b)
△F-measure ‘Hs’ vs ‘No Hs’ (%)

Template Set (i)		**Template Set (ii)**	
G1	**G2**	**G1**	**G2**
-20%	**33%**	**186%**	**186%**
225%	**200%**	**-16%**	**-16%**
6%	**-14%**	**20%**	**20%**
192%	**294%**	**100%**	**100%**
-55%	**-55%**	**-55%**	**-55%**
-19%	**148%**	**-2%**	**-2%**

(c)
△F-measure ‘Hs’ vs ‘No Hs’ (Abs.)

Template Set (i)		**Template Set (ii)**	
G1	**G2**	**G1**	**G2**
-0.01	**0.01**	**0.13**	**0.13**
0.36	**0.32**	**-0.08**	**-0.08**
0.03	**-0.06**	**0.11**	**0.11**
0.50	**0.50**	**0.11**	**0.11**
-0.44	**-0.44**	**-0.44**	**-0.44**
-0.15	**0.37**	**-0.01**	**-0.01**

KEY
Per building per granularity (G1, G2) **bold** F-measure is the highest

KEY
Green text: △ > 0 Black text: △ = 0 **Red** text: △ < 0

Table G.1: HOG-based Template Matching – SVM versus No SVM – Experimental Results – C20 Style 3D Building Models (Table 1 of 2)
Values are provided for running the method with template set (i), and with template set (ii). Each of those sets of results provides values at each class granularity i.e. G1, G2, G3. G1 and G2 for C20 and Georgian-Regency 3D building models are the same (only one style of template was used). There is no G3 for the C20 or Norman style 3D building models – see Chapter 3. Sub-table (*a*) contains F-measure results (vertically) per building and per granularity. The highest F-measure for a per-building, per-granularity set is highlighted in **bold**. Sub-tables (*b*) and (*c*) show the delta (△) in F-measure when using the SVM versus the use of no SVM, in percent and as an absolute value respectively.

See also Table G.1

C20 Style 3D Building Model	Use SVM?	(a) F-measures				(b) ∆F-measure ‘Hs’ vs ‘No Hs’ (%)				(c) ∆F-measure ‘Hs’ vs ‘No Hs’ (Abs.)			
		Template Set (i)		Template Set (ii)		Template Set (i)		Template Set (ii)		Template Set (i)		Template Set (ii)	
		G1	G2	G1	G2	G1	G2	G1	G2	G1	G2	G1	G2
BM_2_7	No SVM	0.37	0.37	0.58	0.58								
	SVM	**0.56**	**0.56**	**0.75**	**0.75**	**51%**	**51%**	**29%**	**29%**	**0.19**	**0.19**	**0.17**	**0.17**
BM_2_8	No SVM	**0.5**	**0.32**	**0.6**	**0.6**								
	SVM	0.31	0.31	0.51	0.51	**-38%**	**-3%**	**-15%**	**-15%**	**-0.19**	**-0.01**	**-0.09**	**-0.09**
BM_2_9	No SVM	0.62	0.54	**0.7**	**0.7**								
	SVM	*	*	0.67	0.67	*	*	**-4%**	**-4%**	*	*	**-0.03**	**-0.03**
BM_2_10	No SVM	**0.25**	**0.25**	**0.26**	**0.26**								
	SVM	0.21	0.21	0.22	0.22	**-16%**	**-16%**	**-15%**	**-15%**	**-0.04**	**-0.04**	**-0.04**	**-0.04**
BM_2_11	No SVM	0.06	0.05	**0.69**	**0.69**								
	SVM	**0.69**	**0.69**	**0.69**	**0.69**	**1050%**	**1280%**	0%	0%	**0.63**	**0.64**	0.00	0.00
BM_2_12	No SVM	0.08	0.08	**0.84**	**0.84**								
	SVM	**0.64**	**0.64**	0.78	0.78	**700%**	**700%**	**-7%**	**-7%**	**0.56**	**0.56**	**-0.06**	**-0.06**

KEY
Per building per granularity (G1, G2) **bold** F-measure is the highest

KEY
Green text: ∆ > 0 Black text: ∆ = 0 **Red** text: ∆ < 0

Table G.2: HOG-based Template Matching – SVM versus No SVM – Experimental Results – C20 Style 3D Building Models (Table 2 of 2)
See caption for Table G.1. For entries marked ‘*’ see Appendix H.

		(a) F-measures					(b) △F-measure vs 'No SVM' (%)					(c) △F-measure vs 'No SVM' (Abs.)				
		Template Set (i)			Template Set (ii)		Template Set (i)			Template Set (ii)		Template Set (i)			Template Set (ii)	
Georgian-Regency Style 3D Building Model	**Use SVM?**	**G1**	**G2**	**G3**	**G1 & G2**	**G3**	**G1**	**G2**	**G3**	**G1 & G2**	**G3**	**G1**	**G2**	**G3**	**G1 & G2**	**G3**
BM_GR_1	No SVM	**0.32**	**0.32**	**0.16**	0.93	**0.48**										
	SVM	0.24	0.24	0.12	**0.96**	0.46	-25%	-25%	-25%	3%	-4%	-0.08	-0.08	-0.04	0.03	-0.02
BM_GR_2	No SVM	**0.78**	**0.74**	**0.42**	**0.87**	**0.46**										
	SVM	0.55	0.41	0.23	0.64	0.32	-29%	-45%	-45%	-26%	-30%	-0.23	-0.33	-0.19	-0.23	-0.14
BM_GR_3	No SVM	**0.86**	**0.86**	0.36	0.87	0.37										
	SVM	**0.86**	0.85	**0.38**	**0.88**	**0.38**	0%	-1%	6%	1%	3%	0.00	-0.01	0.02	0.01	0.01
BM_GR_4	No SVM	**0.9**	**0.9**	**0.65**	**0.92**	0.67										
	SVM	0.76	0.76	0.47	**0.92**	**0.68**	-16%	-16%	-28%	0%	1%	-0.14	-0.14	-0.18	0.00	0.01
BM_GR_5	No SVM	**0.2**	**0.15**	**0.06**	0.67	0.23										
	SVM	0.08	0.07	0.05	**0.78**	**0.35**	-60%	-53%	-17%	16%	52%	-0.12	-0.08	-0.01	0.11	0.12
BM_GR_6	No SVM	**0.79**	**0.79**	0.53	**0.96**	**0.64**										
	SVM	0.61	0.61	**0.61**	0.95	0.63	-23%	-23%	15%	-1%	-2%	-0.18	-0.18	0.08	-0.01	-0.01
BM_GR_7	No SVM	0.57	0.57	0.29	**0.77**	**0.46**										
	SVM	**0.77**	**0.6**	**0.36**	0.74	0.44	35%	5%	24%	-4%	-4%	0.20	0.03	0.07	-0.03	-0.02

KEY
Per building per granularity (G1, G2, G3) **bold** F-measure is highest

KEY
Green text: △ > 0 Black text: △ = 0 **Red** text: △ < 0

Table G.3: HOG-based Template Matching – SVM versus No SVM – Experimental Results – Georgian-Regency Style 3D Building Models
See caption for Table G.1.

		(a) F-measures				**(b) △F-measure vs 'No SVM' (%)**				**(c) △F-measure vs 'No SVM' (Abs.)**			
Norman Style 3D Building Model	**Use SVM?**	**Template Set (i)**		**Template Set (ii)**		**Template Set (i)**		**Template Set (ii)**		**Template Set (i)**		**Template Set (ii)**	
		G1	**G2**	**G1**	**G2**	**G1**	**G2**	**G1**	**G2**	**G1**	**G2**	**G1**	**G2**
BM_N_1	No SVM	**0.83**	**0.58**	**0.83**	**0.58**								
	SVM	**0.8**	**0.56**	**0.8**	**0.56**	**-4%**	**-3%**	**-4%**	**-3%**	**-0.03**	**-0.02**	**-0.03**	**-0.02**
BM_N_2	No SVM	0.69	**0.55**	**0.69**	**0.55**								
	SVM	0.48	**0.48**	0.43	0.22	**-30%**	**-13%**	**-38%**	**-60%**	**-0.21**	**-0.07**	**-0.26**	**-0.33**
BM_N_3	No SVM	0.72	0.47	**0.71**	**0.46**								
	SVM	**0.75**	**0.5**	0.68	0.45	**4%**	**6%**	**-4%**	**-2%**	**0.03**	**0.03**	**-0.03**	**-0.01**
BM_N_4	No SVM	0.5	0.35	**0.88**	**0.64**								
	SVM	**0.57**	**0.57**	**0.79**	**0.64**	**14%**	**63%**	**-10%**	0%	**0.07**	**0.22**	**-0.09**	0.00
BM_N_5	No SVM	0.11	0.04	0.26	**0.06**								
	SVM	**0.14**	**0.07**	**0.31**	**0.06**	**27%**	**75%**	**19%**	0%	**0.03**	**0.03**	**0.05**	0.00

KEY
Per building per granularity (G1, G2, G3) **bold** F-measure is highest

KEY
Green text: △ > 0 Black text: △ = 0 **Red** text: △ < 0

Table G.4: HOG-based Template Matching – SVM versus No SVM – Experimental Results – Norman Style 3D Building Models
See caption for Table G.1.

Appendix H Extended Experiments – Runtimes

H.1 Runtime – HOG-based Template Matching (no SVM)

Table H.1 details the runtimes for processing each 3D building model. The mean runtimes for processing a 3D building model were 12 minutes, 13 minutes, 7 minutes and 12 minutes for C20, Georgian-Regency, Norman and for all 3D building models respectively. The lowest runtime was for BM_GR_6 (2 minutes), and the highest was for BM_GR_5 (44 minutes). Aside from BM_GR_5 and BM_2_6 and BM_2_9 the highest runtime was 16 minutes.

	(a)		(b)	(c)
	Runtime (seconds)		Decrease in Runtime (%)	Decrease in Runtime (Abs.)
	Templates			
3D Building Model	Template Set (i)	Template Set (ii)		
BM_2_1	2180	955	56%	1225
BM_2_2	529	258	51%	271
BM_2_3	297	143	52%	154
BM_2_4	538	250	54%	288
BM_2_5	529	271	49%	258
BM_2_6	2940	2420	18%	520
BM_2_7	392	202	48%	190
BM_2_8	392	213	46%	179
BM_2_9	4650	2,520	46%	2130
BM_2_10	706	423	40%	283
BM_2_11	1410	878	38%	532
BM_2_12	832	413	50%	419
MEAN	**1283**	**746**	**42%**	**537**
TOTAL	**15395**	**8946**	**42%**	**6449**
BM_GR_1	742	331	55%	411
BM_GR_2	664	329	50%	335
BM_GR_3	1540	796	48%	744
BM_GR_4	1160	535	54%	625
BM_GR_5	,190	2610	18%	580
BM_GR_6	178	135	24%	43
BM_GR_7	2210	805	64%	1405
MEAN	**1383**	**792**	**43%**	**592**
TOTAL	**9684**	**5541**	**43%**	**4143**
BM_N_1	1140	566	50%	574
BM_N_2	517	251	51%	266
BM_N_3	2,073	789	62%	1285
BM_N_4	851	323	62%	528
BM_N_5	709	284	60%	425
MEAN	**1058**	**443**	**58%**	**616**
TOTAL	**5291**	**2213**	**58%**	**3078**
For all 3D Building Models				
MEAN	**1265**	**696**	**45%**	**570**
TOTAL	**30370**	**16700**	**45%**	**13670**

Table H.1: Runtimes by 3D Building Model – no SVM
(*a*) Runtimes (in seconds) per 3D building model, including mean and total, split by style, then, for all 3D building models combined. (*b*) Percentage decrease in runtime between running with templates for all four styles and with templates specific to the building style only. (*c*) Absolute decrease represented by (b). For an indication of the complexity of each 3D building model see Table 3.1, Table 3.2 and Table 3.3 in Chapter 3 for an indication of the complexity of the 3D building models.

H.2 Runtime – HOG-based Template Matching – SVM

Table H.2 details the runtimes for processing each 3D building model, with the use of the SVM, and also shows the increases in runtime compared to running with no SVM, as shown in Table H.1. There was a consistent increase in runtime when using the SVM. The mean increases in runtimes for using the SVM, compared with no SVM, were 13 minutes (a 103% increase), 7 minutes (54%), 11 minutes (155%) and 11 minutes (94%) for C20, Georgian-Regency, Norman and for all 3D building models respectively. These results were for template set (ii). Using the larger number of templates in template set (i) meant that runtimes were higher still. The mean increases between running template set (i) and template set (ii), both with the SVM, were: 490%, 141%, 149% and 338% for C20, Georgian-Regency, Norman and for all 3D building models respectively. This compares to equivalent mean increases of 72%, 75%, 139% and 82% when using no SVM. Note also that there were some anomalies in these increases (with use of the SVM) at a 3D building model level, the most marked of which was BM_2_1, which saw a 3,905% increase in runtime from 36 minutes to 24 hours!

Regarding the inability of the pipeline to process BM_2_9 when using template set (i), with dimensions of 3,872×2,592 pixels, the resulting matrix was almost 3GB and 89 million rows in size. Such a size meant that this particular texture map image would not process due to memory issues, despite the generous memory quota on the virtual machine. This is the reason why there are no F-measure results or runtimes for BM_2_9 with template set (i).

	(a)		(b)		(c)	
	Runtime (seconds)		Increase in Runtime (%)		Increase in Runtime (Abs.)	
	Templates					
3D Building Model	**Template Set (i)**	**Template Set (ii)**	**Template Set (i)**	**Template Set (ii)**	**Template Set (i)**	**Template Set (ii)**
BM_2_1	87300	2540	3905%	166%	85120	1585
BM_2_2	1130	443	114%	72%	601	185
BM_2_3	738	270	148%	89%	441	127
BM_2_4	1320	430	145%	72%	782	180
BM_2_5	1280	456	142%	68%	751	185
BM_2_6	7390	3770	151%	56%	4450	1350
BM_2_7	870	315	122%	56%	478	113
BM_2_8	1140	423	191%	99%	748	210
BM_2_9		6900		174%		4380
BM_2_10	1630	681	131%	61%	924	258
BM_2_11	2400	1190	70%	36%	990	312
BM_2_12	1770	725	113%	76%	938	312
MEAN	**8914**	**1512**	**595%**	**103%**	**7631**	**766**
TOTAL	**106968**	**18143**	**595%**	**103%**	**91573**	**9197**
BM_GR_1	1500	542	102%	64%	758	211
BM_GR_2	1030	423	55%	29%	366	94
BM_GR_3	3320	1300	116%	63%	1780	504
BM_GR_4	2460	905	112%	69%	1300	370
BM_GR_5	4810	3120	51%	20%	1620	510
BM_GR_6	309	167	74%	24%	131	32
BM_GR_7	7200	2100	226%	161%	4990	1295
MEAN	**2947**	**1222**	**113%**	**54%**	**1564**	**431**
TOTAL	**20629**	**8557**	**113%**	**54%**	**10945**	**3016**
BM_N_1	2030	944	78%	67%	890	378
BM_N_2	856	403	66%	61%	339	152
BM_N_3	5890	2300	184%	192%	3817	1511
BM_N_4	2670	1030	214%	219%	1819	707
BM_N_5	2570	956	263%	236%	1861	672
MEAN	**2803**	**1127**	**165%**	**155%**	**1745**	**684**
TOTAL	**14016**	**5633**	**165%**	**155%**	**8725**	**3420**
For all 3D Building Models						
MEAN	**5901**	**1347**	**366%**	**94%**	**4635**	**651**
TOTAL	**141643**	**32333**	**366%**	**94%**	**111243**	**15633**

Table H.2: Runtimes by 3D Building Model – SVM & Differences with No SVM
(*a*) Runtimes (in seconds) per 3D building model, including mean and total, split by style, then, for all 3D building models combined. (*b*) Percentage increase in runtime between running with no SVM (shorter run time) and running with the SVM (longer runtime). (*c*) Absolute decrease represented by (b). For an indication of the complexity of each 3D building model see Table 3.1, Table 3.2 and Table 3.3 in Chapter 3.

Appendix I Code – Summary

This code has been redacted for copyright / IP reasons

The same is true for that which was in Appendices J, K, L, M, N and O.