URBAN FABRICS AS A TOOL OF URBAN PERFORMANCE MITIGATION

Algorithmic approach for bridging building geometry with urban performance optimisation

Anas Moustafa Hosney Elsayed Lila

B.Sc (Arch), M.Sc. (Arch)

A THESIS SUBMITTED TO CARDIFF UNIVERSITY FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

2020

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 Doctor of Philosophy

Signed …….. (Candidate) Date ……

STATEMENT 2

This thesis is the result of my own independent work/investigation, except where otherwise stated.

Other sources are acknowledged be explicit references.

Signed …….. (Candidate) Date ……

STATEMENT 3

I hereby give consent for my thesis, if accepted, to be available for photocopying and for inter-library loan, and for the title and summary to be made available to outside organizations.

Signed …….. (Candidate) Date ……

STATEMENT 4: PREVIOUSLY APPROVED BAR ON ACCESS

I hereby give consent for my thesis, if accepted, to be available for photocopying and for inter-library loans **after expiry of a bar on access previously approved by the Academic Standards & Quality Committee.**

Signed …….. (Candidate) Date ……

In the name of Allah the most compassionate and the most Merciful

Acknowledgement

First of all, I am very grateful to Allah Almighty for granting me the strength, patience and health to accomplish.

I want to express my genuine gratitude to everyone who has supported me during the years of working on this thesis. First and foremost, I will always be in debt to my first supervisor, Dr. Simon Lannon, for his continouse care and guidence and mentoring since the beginning of this research. He was always there for shaping the direction of this thesis and providing professional guidance at all times.

My appreciation also goes to my second supervisor Dr. Wassim Jabi for his concern and thoughtful advice. My career and research would not be the same without his generous help and worthy experience.

Moreover, I pass my thanks to Dr Heba El Sharkawy, who started the first steps of this project.

I want to extend my recognition to my colleagues and friends, with whom I shared this unique experience and supported me over this long journey. I mainly mention my friends Yasser Mehanna, Youmna El-ghazi, Reem Okasha, and the whole PGR community in WSA.

I am thankful for the newton fund represented in the Newton Musharaf scholarship for financing this doctorate research at Welsh School of Architecture, Cardiff University.

Dedication

I dedicate this humble work to

My mother

whose support and encouragement and prayers were the reason I feel loved and alive.

My Father's soul.

I wish I could make him proud of his youngest son and live up to any of his standards.

My brothers,

Mohamed, Ahmed and Osama

For the endless safety and peace, they have been providing at all times, and for the moments that we will never forget and for the principles we will try our best to communicate and deliver to our next generation.

My Daughters

Zeina & Nour

The peak of god's blessings.

And for all fellow scholars of science and peace.

Abstract

The advancements in methods of built environment design have led to the rise of computational methods in urban modelling and environmental simulation to aid the early stages of the design process. Computational urban modelling and simulation methodologies can use a parametric approach to enable geometrical dynamic modelling and investigate urban environmental performance.

This thesis aims to understand the effect of urban geometry on urban performance in an algorithmic approach to reach for an efficient optimisation approach based on this impact. To achieve this goal, a framework was developed to enable a time-efficient performance-based optimized design to guide the urban design process in the early stages. This framework establishes a new environmental data-driven approach for designing urban neighbourhoods.

A preliminary sensitivity analysis measured the relative importance of geometrical variables, their impact on performance aspects and the computational time. It was conducted in two locations, Aswan, Egypt, London, UK, for cooling and heating demands. The results of this analysis quantified relative importance for the tested geometrical variables' impact on energy demand. It was clear that computational and time cost is limiting the capability of conducting general performance optimisation on the urban scale. This led the research to the classification of geometry to optimise urban geometry based on its solar radiation performance.

The parametric workflow presents a methodology to break down the neighbourhood model into its geometrical variables: location, orientation, building's area, height, typology, and the surrounding geometry context. Then, a database of text annotations for generated buildings was attached to its solar radiation simulation results. These annotations are used as indicators to match the following geometry generations to save simulation time in similar geometrical scenarios.

The framework was used to optimize solar radiation for a neighbourhood geometry in Aswan, Egypt. Machine learning principles were adopted to provide the framework with prediction capabilities of solar radiation performance with accepted prediction accuracy and reduced time consumption. A positive linear correlation was found between machine learning

principles and its equivalent simulation results–for architectural and urban scales. The proposed prediction approach succeeded to achieve significant time savings compared to the traditional simulation process with acceptable accuracy. These reported findings shed new light on the capability of optimisation in the early design stages.

The Genetic Algorithm's optimisation principles show a significant capability to find optimal or near-optimal solutions for hypothetical and existing neighbourhood context tests while saving more than 80% of the computational time needed. These results present a template for using data-driven urban design to inform environmental decisions in the early design stage at the neighbourhood scale.

List of publications

1. Lila, A.M.H., Jabi, W. and Lannon, S. 2021. Predicting solar radiation with Artificial Neural Network based on urban geometrical classification. In: *IBPSA 2021*. bruges, belgium

2. Lila, A.M.H. and Lannon, S. 2019. Classifying urban geometry impact on solar radiation. In: Corrado, V., Fabrizio, E., Gasparella, A., and Patuzz, F. eds. *the International Building Performance Simulation Association (IBPSA) 2019 16th Conference*. Rome, Italy

3. Chatzivasileiadi, A., Lila, A.M.H., Lannon, S. and Jabi, W. 2018. The Effect of Reducing Geometry Complexity on Energy Simulation Results. In: *36th annual Education and research in Computer Aided Architectural Design in Europe (eCAADe)*. Lodz, Poland.

4. Lila, A.M.H. and Lannon, S. 2017. A parametric sensitivity analysis of the impact of built environment geometrical variables on building energy consumption. In: *PLEA*. Edinburgh, UK.

5. Lila, A.M.H., Lannon, S. and Jabi, W. 2017. Holistic sensitivity analysis on urban geometry and its effect on building performance in hot arid zones. In: ElSharkawy, Heba, Zahiri, Sahar and Clough, J. ed. *International Conference for Sustainable Design of the Built Environment (SDBE)*. london: University of East London, pp. 193–204.

Table of Contents

List of Acronyms and Abbreviations

AI	Artificial Intelligence
ANN	Artificial Neural Networks
BIM	Building Information Models
CEA	City Energy Analyst
CFD	Computational Fluid Dynamics
CGA	Computer Generated Architecture
CIBSE	Chartered Institution of Building Services Engineers
DOE	Department of Energy
EBP	Energy Performance of Buildings
EEP	Energy and Environmental Prediction model
EEPFD	Evolutionary Energy Performance Feedback for Design
EPC	Energy Performance Certificate
ETMY	Egyptian Typical Meteorological Year
FAR	Floor Area Ratio
FSI	Floor Space Index
GA	Genetic Algorithms
GBS	Green Building Studio
GDFD	Grammars Derived Form Design
GIS	Geographic Information System
HB	Honeybee
HVAC	Heating, Ventilation and Air Conditioning
IES	Illuminating Engineering Society
IRRGA	Implicit Redundant Responsive Genetic Algorithm
LCA	Life Cycle Assessment
LCZ	Local Climate Zones
LoD	Level of Details
LT-Model	Lighting and Thermal Method
MIT	Massachusetts Institute of Technology
MSE	Mean Square Error
NMT	Non-Manifold Topology
NURBS	Non-Uniform Rational Basis Spline
SAP	Standard Assessment Procedure for residential buildings
SOM	Self-Organizing Map
UD	Urban Daylight
UMI	Urban Modeling Interface
UNFPA	United Nations Population Fund
VPL	Visual Programing Language
WSA	Welsh School of Architecture
WWR	window-to-wall ratio

List of Figures

List of Tables

1 Introduction

1.1 Research context

Urban sustainable design has become the focus of heightened attention due to continuous alerts from United Nations Population Fund (UNFPA) reports about world urban growth and the increasing percentage of urbanised environments compared to past years' statistics. These statistics were also accompanied by predictions of the continuity of this increase. It is envisaged that the scale of cities and urbanised communities will reach limits never seen before globally (Martine and Marshall 2007). This concern highlighted the importance of investigating urban growth to mitigate its environmental impact on the planet. The environmental burden comes from the effect of Greenhouse Gas emissions caused by the construction industry. Also, the built environment is responsible for consuming almost 40% of the energy demands. All this encouraged multiple studies to experiment and analyse different ways to reduce and hopefully eliminate these impacts on the environment.

New urban design methodologies have evolved to develop different environmental and sustainable approaches. They have aimed at various sustainable goals, such as reducing urban and architectural energy consumption, mitigating the urban heat island effect, and reducing the urban carbon footprint. Urban geometry has gained some attention based on its key role in controlling the built environment performance. The urban geometry controls the sun penetration within the urban environment and therefore controls the energy consumed for lighting and thermal performance. In addition, the urban heat island phenomenon, which affects the urban microclimate, is also mainly caused by urban geometry and its density and its loads on the thermal performance of the built environment.

Urban modelling and simulation are considered part of the investigative approaches that can provide a clear understanding of urban environmental performance and its links to urban geometry. Urban modelling has gained a recognisable advancement from the development of the parametric approach. Parametric design involves not only the software application, which is broadly known and used currently, but it is an algorithmic procedure to break down design elements to control them individually and reproduce a broader range of options for the design problem. This allowed urban models to address and handle urban geometrical complexity and analyse and come up with enhanced and performance-based urban models. Urban performance optimisation has been under investigation through different angles, some of which addressed the link between geometry and performance and how to apply

optimisation methods and even artificial intelligence principles to look for the optimal geometrical solution that performs in favour of the local climate conditions. There are continuous explorations of implementing these advancements in the early stages of urban design to inform and enhance the urban design decision-making process.

1.2 Statement of the problem

The rapid continuous urban growth creates increasing environmental challenges. In some cases, urban designers and decision-makers find it hard to create environmentally informed designs to overcome environmental challenges and burdens due to lack of time. One of the environmental impacts of increasing urbanism is the urban heat island which can be scaled down to the urban microclimate and urban context. The change in the urban microclimate due to urbanisation is directly linked to the energy consumption to control the indoor environment either by cooling or heating indoor spaces (Landsberg 1981). Different ways of implementing urban microclimate mitigation in the urban context reduce adverse effects on temperature, energy consumption, and demands, promoting environmental quality and air quality and protecting human health. The urban heat island can be controlled and its impacts reduced by either treating building envelopes, increasing vegetation ratio, implementing sustainable landscape materials and designs and modifying urban fabric and geometry as an approach of futuristic policies and efforts (Nunez 1974; Che-Ani et al. 2009; Stewart and Oke 2012). The rate of urban growth drives decision-makers and urban planners to investigate new urban strategies to help improve the quality of life in the expected new urban communities. This research tries to find better environmentally performing urban patterns and built environments that will avoid or reduce the effect of the urban context on architectural thermal performance. This research is trying to provide an algorithmic framework that classifies urban geometrical features and links them to environmental performance and use this link in optimising the following generated urban patterns to save time consumed in running simulations to inform the environmental decision-making process in the early stages of urban design.

1.3 Research aim, questions, and objectives

The research investigates the effect of urban geometry on urban performance in an algorithmic approach that targets the generation of urban geometrical variables and classifies its individual buildings based on the shared geometrical features. It will utilise this effect to create a database of classified buildings with their simulation performance results to allow for artificial intelligence-assisted optimisation that will inform the decision-making process in the early stages of neighbourhood design. This will reduce the time needed to search for the optimal performing urban geometry, which will help enhance the design process and inform further aspects of environmental analysis.

The main scope of this research involves the study of urban geometry, urban performance simulation and optimisation as it attempts to answer the following questions:

- What is the relative importance of geometrical built environment variables on environmental performance?
- How can parametric data flow help generate and analyse urban different geometries patterns and performance and utilise urban modelling complexity?
- To what extent does the geometry classification prediction reduce the time consumed for parametric simulation methods, and how far would it affect the results' accuracy?
- What kind of optimisation method would help search for a timely, efficient optimal solution for urban neighbourhood's geometry with acceptable accuracy?

To achieve this goal and to answer the questions, the research has set the following objectives:

- To provide a better understanding of the relative importance of geometrical variables to performance on an urban scale
- Analysis of the impact resulting from altering urban geometry on its environmental performance and try to find a way to utilise this impact in optimising urban geometry
- Establishing an urban geometry generation framework that will feed into a solar radiation prediction-based design process

- Utilising this framework in a parametric approach to handle urban modelling complexity and extract performance data and visualisations dynamically
- Utilise urban geometry impact on performance to search for an optimisation method to facilitate data-driven design approach included in the early stages of design

1.4 Methodology and research framework

Urban design studies have dealt with urban geometry and its impact on urban performance from different angles according to the variables most related to the problem under investigation. Urban geometry research mainly focuses on addressing urban complexity to visualise and model this in complex urban environments to assess and simulate its performance from different aspects. Parametric modelling and generative design have recently gained some attention in urban geometry research. This is added to the challenging cost of time for performance-based decision-making processes in the early design stages. The optimisation of urban geometry performance is addressed by different approaches to be included in the early stage of design to help with the decision-making process. However, urban geometry was rarely utilised in linking urban geometry to the performance by applying artificial intelligence optimisation methods. Given how this research is a multi-disciplinary project, breaking down the methods into multiple phases was needed to achieve the research objectives and answer the research questions.

The first phase of the research is the theoretical study in chapter two, which reviews the literature regarding the following points:

- Investigating urban computational modelling approaches and their complexity
- Studying urban parametricism and generation tools and methods
- Investigating urban performance simulation tools and methods
- Studying urban geometrical variables and their relative importance on urban performance
- Analysing optimisation methods based on geometry impact on urban performance.

This will be followed by the next analytical section starting In chapter four to chapter eight, which consists of the following stages:

1.4.1 Preliminary study

This will be in chapter four, starting with a sensitivity analysis to better understand the relative importance of the geometrical variables on multiple performance aspects like energy demand, solar radiation, and lighting for indoor performance. This analysis will run for different climate conditions to validate the results of the study. This sensitivity analysis is still in the preliminary stage of the project; it aims to be conducted on a simple grid urban form and simple testing for optimisation methods.

1.4.2 Parametric and generative urban modelling

In chapter five, the research will build a parametric framework utilising a shape grammar sequence that can generate different neighbourhood geometrical iterations representing a wide range of alternatives controlled by a commonly used visual programming platform.

1.4.3 Geometrical classification and data retrieval

The geometrical classification is done based on specific categories found in the literature review and preliminary studies. Chapter six will show that it is made in a parallel sequence to the generation sequence, making it flexible and responsive to the geometrical iterative process and allowing for better recognition for data retrieval. Collecting the classified data also is a different sequence to make it ready for different methods of recognition or prediction based on its three-dimensional features. The urban neighbourhoods were classified on a building level, and buildings were classified based on their urban context and architectural typologies.

1.4.4 Empirical study

This parallel workflow allowed the algorithm to sort, label and classify buildings in each generated urban configuration by a text label saved along with its performance analysis results collected from the simulation brute force process. In the optimisation stage, the simulation results and geometrical building features will act as a lookup database. The algorithm will search for similar features between what is saved in it and the newly tested geometries. The second layer of artificial intelligence is added to this study. The classified annotations were used as an input for a simple neural network node as training data. As shown in chapter seven, this neural network predicts the performance based on the input data. This prediction result is combined with the lookup retrieving outcomes, to sum up neighbourhood performance results. Following chapter eight, a genetic algorithm is used to

determine the optimal performing neighbourhood geometry in the tested pool. This artificial intelligence application is aimed to reduce the time of simulation and optimisation of neighbourhoods' geometry without risking the needed accuracy.

1.4.5 Testing

Also, in chapter eight, a comparison test will be done between the actual full brute force simulation results for the urban configuration with no classification or approximation for individual buildings against results from the framework prediction and approximation for the same urban configuration. Simultaneously, testing the framework setting until it reaches an acceptable accuracy rate for both the saved neighbourhood's geometry and newly generated ones. This will be followed by a series of tests for the optimisation capabilities of the framework by testing it with different sizes of iteration pools to investigate how near could it be to the optimal solution.

The following stage of testing is to apply this generation and prediction framework on an existing case study in the same climate zone tested in the previous stages to get a clearer understanding of how far this framework could be applied to existing urban geometry optimisation.

The theoretical background of this study is based on reviewing the state-of-the-art urban modelling approaches and the available links between these approaches and available simulation tools. Moreover, it investigates the various optimisation methods for urban geometry in the early stages of design. The recommendations and conclusions of these earlier studies were considered to form the scope and foundation of the presented study. The preliminary study of this research has aimed to better understand the relative importance of geometrical features in an urban context and their impact on different aspects of performance. It aimed to investigate the time constraints of simulation in an urban context and identify how much time each performance aspect can cost to be included in the early design stages. Another goal was to figure the level of dependency between different aspects of performance. The preliminary study was conducted in three different climate conditions to compare the findings and establish the rational reasoning of the thesis hypothesis, which will be discussed in the relative chapter. The generative classification framework was built up based on the preliminary study's findings to break down the urban models' complexity into its core characteristics, making it easier to control, identify, and link to their impact on urban

performance. This generative framework was built to accommodate different inputs and produced a unified classification process of urban geometry. The empirical study stage utilised this classification process in simulation and optimisation methods of urban performance. It

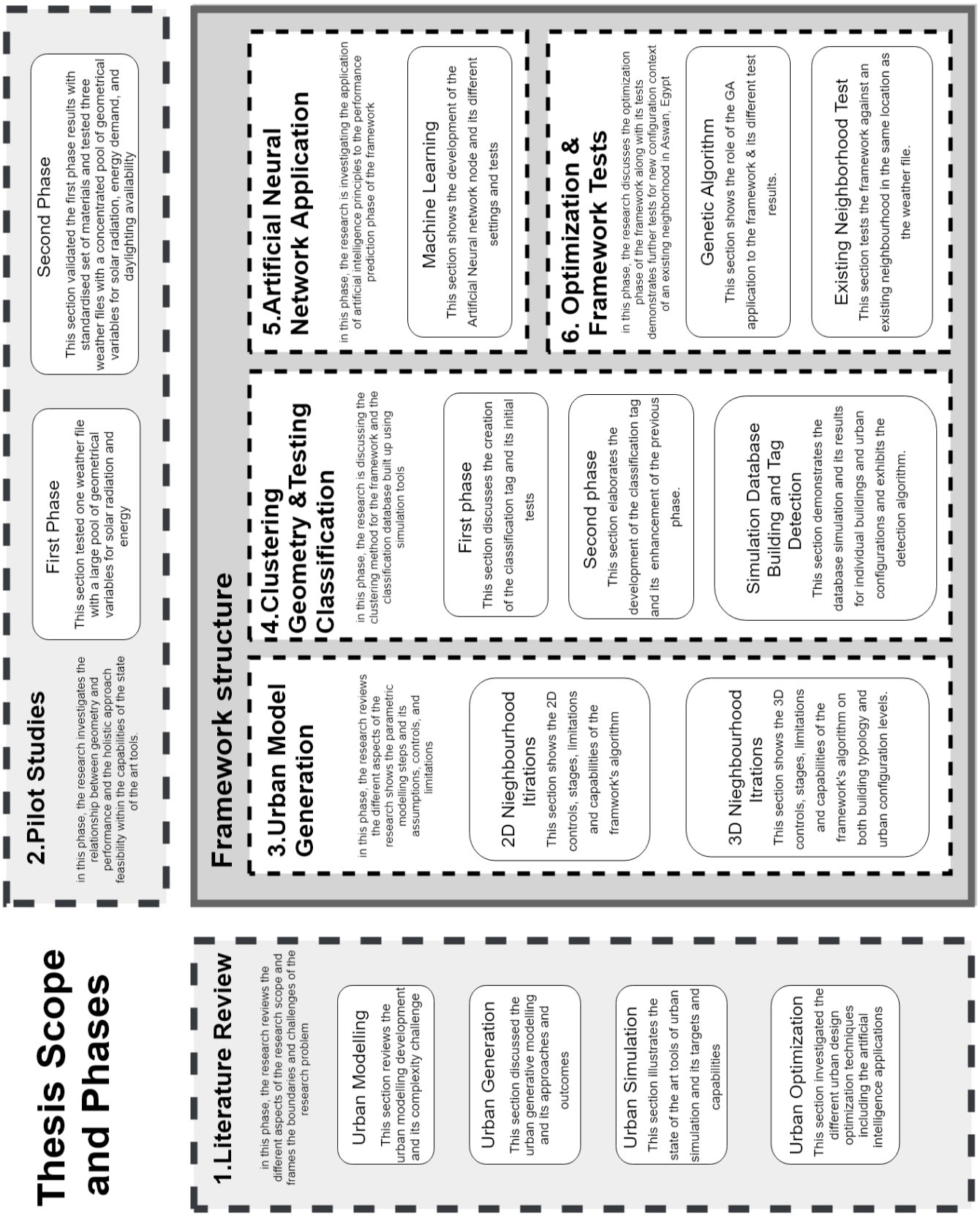


Figure 1-1 Thesis scope and different phases flow

tested the potentiality of applying artificial intelligence principles in optimising the performance of these classified neighbourhood models. The final testing stage aimed to test the findings of this framework against a conventional urban simulation and optimisation process using the same modelling tools and the ability to implement this framework to different case study inputs. The final stage was to test the applicability of this framework on an existing urban neighbourhood, test its classification capabilities and test how the optimisation can enhance performance for these case studies. Figure 1-1 shows the structure of the thesis and the different stages of the methods.

1.5 Scope and limitation

The modelling scale of this study is the neighbourhood scale which allows the highlighting of geometrical features like the orientation of buildings, heights, density and also a brief investigation of building typologies like window-to-wall ratios and the addition of central courtyards. The targeted functionality of this study is aimed at midrise residential buildings. The framework deals with a large variety of urban geometries due to the capabilities allowed by the classification techniques, focusing on the individual buildings rather than classifying each neighbourhood as a whole. This allowed more flexibility in the recognition and prediction. Due to the computational cost of running such a number of iterations on this scale, the research focuses on urban geometry and its impact on performance. It does not include other urban performance drivers like vegetation and albedo, for example.

The modelling platform used in this study is a visual programming language enabled tool called "Grasshopper". It acts as an add-on to a conventional modelling software called "Rhinoceros". The literature shows that Grasshopper is widely used to implement different parametric modelling and simulation approaches. It accommodates various plug-ins to conduct built environment simulation and optimisation on various scales and principles. However, it is a capable and widely used tool; Grasshopper and the simulation plug-ins implemented in it are not the fastest way to conduct simulations on an urban scale due to the initial cost of processing time on machines running Windows. This is also another limitation this research is dealing with, limiting down the study's options and parameters to have the framework built on a widely used and recognised platform that allows for more accessible future enhancement and knowledge transfer (Martin 2014).

1.6 Case study locations

The preliminary study is conducted for three climate characteristics for three cities with different urban geometry contexts and growth strategies. The cities are as follows:

- **Aswan, Egypt**: It is one of the major cities in the southern part of Egypt. It is targeted as one of the future expanding cities of the country's strategic plan for new cities (Egyptian Ministry of State for Administrative Development 2016). Its climate is categorised as a hot arid zone according to the ASHRAE classification of climate (ANSI/ASHRAE/IESNA 2010). The study of this city focused on cooling demand. To balance it with window-to-wall ratios, an analysis of daylighting availability was added to the later stages of the study.

- **London, UK**: It is one of the largest metropolitans globally and is expected to have its largest population by 2020 (Greater London Authority 2016). According to the ASHRAE classification, London weather is classified as mixed humid weather conditions (ANSI/ASHRAE/IESNA 2010). Due to the climate, the preliminary study highlighted heating demand for residential buildings while keeping the daylighting balance testing.

- **Birmingham, UK**: It is one of the largest cities in the UK, and it has a different urban nature and growth plan than London (Birmingham City Council 2017). The study of the city also has heating focused analysis for the preliminary study simulation, as the city is classified as experiencing cold, humid weather conditions. This analysis results can be found in(Appendix A (Preliminary Study))as it was found similar to London results for heating demand.

The preliminary study results have led the scope of the research to create a proof of concept to a framework capable of breaking down urban geometry through a dynamic classification method that enables the prediction of selected performance aspects and geometrical optimisation based on these predictions. This emphasises the potentiality of searching for a performance-based decision-making process in the early stages of design. Achieving this will drive the future work to guide the early stage of urban design on different performance aspects like daylighting, energy demand and solar radiation. It will also make it possible to investigate the capability of applying these methods to different climatic conditions. The

empirical study was done only on the hot arid zone and solar radiation to prove the concept about the framework's capability and its optimisation ability.

2 Literature Review

2.1 Introduction

The urbanization of the world's population is expected to continue to grow; this urban growth introduces environmental impacts on the air temperature and building energy consumption (Martine and Marshall 2007). Therefore, it is critical to developing urban environments that provide human comfort while reducing the environmental burden of urbanization. Urban context mitigation and enhancement are among the key goals of research investigating the reduction of environmental impacts caused by urban growth. Urban climatology, if compared to building climatology, can be recognized as climate features outside the building (Radfar 2012). The urban microclimate generally affects the use of air conditioning in the pursuit of thermal comfort (Landsberg 1981). Urban microclimate mitigation may be implemented in different ways to reduce adverse effects on temperature, energy demands, promoting environmental quality and air quality and protecting human health. Some approaches to mitigate urban microclimate effects can include treating building envelopes, increasing vegetation ratio, implementing and improving the sustainable landscape infrastructure and modifying the urban fabric and geometry (Nunez 1974; Che-Ani et al. 2009; Norton et al. 2015).

Urban performance computational modelling and simulation gained attention due to increasing awareness of urban growth outcomes and mitigating urban microclimate environmental impacts. This attention focused on the early stages of urban design to develop evidence-based approaches to provide better performing urban environments. During the research, critical issues are being addressed to create computationally modelled urban environments and environmentally optimized design decisions on an urban scale for the early design stages. The complexity and level of details of urban modelling are among the issues widely investigated by different studies (Martin and March 1972; Schwarz 2010; Stewart and Oke 2012; Biljecki et al. 2014; Picco and Marengo 2015). Other challenges were the performance simulation methods and techniques. Research is still investigating and developing different approaches to the urban performance simulation process and its inputs, outputs and visualizations (Greenberg and Erdine 2014; Naboni 2014; Trigaux et al. 2014; Nault et al. 2016). Moreover, another interesting aspect involves dealing with the resultant data from these processes and the way to feed it into the urban design and decision-making

process, either for optimisation goals or for evidence-based computationally generated urban designs (Pinto Duarte et al. 2005; Hillier 2007; Koenig 2011; Beirão 2012).

This chapter reviews the landscape of contemporary urban modelling techniques and simulation approaches. The discussion reveals the challenges and potentialities of the current state of the art in the field of urban energy demand simulation. Also, it will show it is possible to achieve optimal performance for a holistic urban environment with the utilization of some of the existing integrated tools and frameworks and overcome some of the limitations of conventional methods.

2.2 Urban computational modelling and complexity

Not far from Christopher Alexander's work (Alexander et al. 1977; Alexander 1979) and his concept of translating architecture into pattern languages and defining the vocabularies used by users, architects or planners to form the built environment. In his *Elementary Models of Built Forms* (March 1971), March introduced the ''built forms'' defined as mathematical or quasi-mathematical models of buildings. This definition produced a new direct way of geometric translation of buildings which formed a solid ground to facilitate computational geometric understanding. These built forms were used to represent buildings to provide different levels of complexity in theoretical studies. Through this approach, March provided a framework that can answer built environment-related enquiries mathematically. These questions relate to building heat loss, cost, urban compactness and primary urban forms. He illustrated more investigation of the mathematical relations between different geometries and their transformational processes in his book *The Geometry of Environment* (March and Steadman 1971). He introduced various ways to conceive the built environment through certain geometrical relations and interpretations (like modules, proportions, transformations, etc.). Adding this perception of the built environment to how the urban environment was introduced in Urban Space and Structures (Martin and March 1972) will present a clear understanding of the built environment as a series of geometrical mathematically represented variables. The same book considered that urban space could be conceived as a grid or a framework that comprises the urban form and its components (like streets, buildings, etc.). They discussed the built environment form and its relation to land use efficiency regarding the potentiality of recognizing the urban environment as a grid with different

iterations of solids and voids. Also, this idea presented what they called “speculations” of buildings and how these buildings’ efficiency can be assessed.

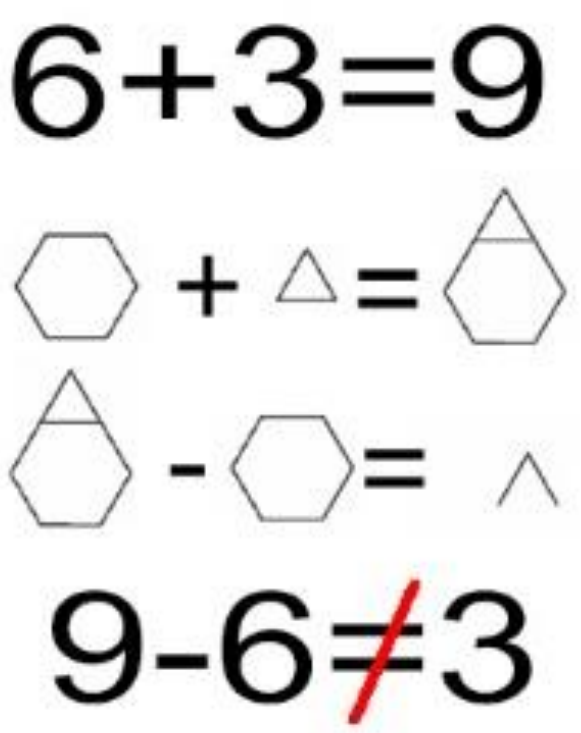


Figure 2-1 shows the difference between numerical and geometrical simple methods as discussed by (March,2011)

As mentioned previously, the idea of mathematically translating geometry and turning it into what we now know as computer modelling was based on the principles of Alexander’s theory of patterns in the built environment (March 1971; March and Steadman 1971; Alexander et al. 1977; Alexander 1979). More recently, in the editorial to volume 13 of the Nexus Network Journal in 2011 (March 2011), March provides a milestone introduction to geometrical mathematics and shape grammars. In addition to his and Alexanders’ work, he covered the work of Stiny and Gips in establishing the foundation of more practical ways to build shape grammars in built environment modelling (Stiny and Gips 1972; Gips 1975; Stiny 1982; Stiny 2006). In this same editorial, March provides a simple example to differentiate between mathematical logics and shape grammars to show how subtraction and addition do not have the same results when applied in shapes, as shown in (Figure 2-1) illustrating the zero-dimension point as the basis of shape grammar as the start of the set of rules. Furthermore, March’s editorial adapted different shape grammar definitions. It introduced a comprehensive definition of shape grammar as it consists of a set of terminal shapes that

forms the vocabulary along with another set of markers which is a different set of shapes, a set of shape rules of the form and the initial shape that starts a tuple of 4 that represents the shape grammar (March, 2011). Based on these ideas and discussions, shape grammar has been applied in many aspects of built environment modelling. In Mitchell's *Logic of Architecture* (Mitchell and J. 1990), he provides an approach to translate the architectural relations into that set of roles "shape grammar". He introduced more generic shape grammars that can be applied in different cases. The book briefly discussed the debate of user/computer agency in the justification of using grammars to analyse and generate architecture. According to this discussion the design process is usually led by designer experience and decisions which are a set of orders, trial and error that can be translated into a shape grammar done by a computer. The translation and control of this process was yet to be carried out by an experienced designer back then.

Besides the facilitation of computational perception of the built environment, the research has investigated the interpretation of different urban patterns and features into computational models and the different ways it has been represented to designers and architects to allow computational applications and software platforms to handle urban design. With that understanding, the process of urban modelling can be categorized between analyzing the built environment and generating the built environment geometry in computer-aided modelling. Analysis of the urban built environment can be understood better by what (Hillier 2007) introduced as "space syntax". This terminology can be defined as analyzing the space to explain the human behaviour within it. It is a human-focused process that deals with the activities in the space from the spatial perspective. This approach has many applications in urban modelling research (Nourian et al. 2010; Karimi 2012; Vasku 2013), some of which was aiming to only analyse the urban space in order to enhance its performance in some aspects, while others have used this analysis to be part of the urban model generation process.

These aspects varied with modelling techniques and tools, such as Tsamis (2017), who used shape grammar to regenerate shape boundaries recreating the model's way of division. Muslimin (2017) used shape grammar to analyse ornaments and allow knowledge transfer for culturally restricted designs and motifs. Jowers et al. (2017) addressed the complexity of multiplication using shape grammar on basic two-dimensional shapes. On the scale of

architecture, Dortdivanlioglu and Economou (2017) analysed two plans by the same architect to reach his common design strategy aspects from analyzing the two projects based on their shape grammars. Shekhawat and Duarte (2017) further added different floor plan boundaries using shape grammars. The study took into consideration functional aspects such as zones adjacency, relations and sizes.

Biljecki et al. (2014) have argued even more about computational modelling by discussing these models' Level of Details (LOD). The study introduced six metrics for model LOD identification, which will help identify models' LOD based on these criteria and each metric.

This way of defining three-dimensional (3D) modelling LOD provides an established guide for urban modelling. It improves the decision for each object's LOD following the model's primary purpose in the first place. This will help the research improve how to model each object and categorize its LOD based on these metrics.

Furthermore, different approaches to analyzing and providing a better understanding of urban modelling characteristics and criteria have been different. (Stewart and Oke 2012) provided a new way of categorizing different urban microclimates in relation to their capacity and primary occupancy. They called it Local Climate Zones (LCZ). Their article published by the American Meteorological Society tries to link between urban heat island causes and characteristics and the urban building capacity and land cover. Stewart & Oke (2012) provide ten different LCZ based on building types and another seven based on land cover, opening the door to a large number LCZs by different combinations between both. This way of illustrating urban microclimates could help simplify the way meteorological data is interpreted in urban modelling and simulation. It will help create new ways of analyzing the urban microclimate and urban context with some criteria of its own, without loading too much ineffective meteorological data about the city or urban broader boundaries into the simulation.

The emerging aspect when it comes to LOD and simulation is the time constraint of the tool. Time becomes more critical when the simulation is utilized in the decision-making process in the early stages of design. Another modelling perception enhancement method is Non-Manifold Topology (NMT), recently reintroduced by Jabi (2015) to replace the widely used polyhedral modelling approach. The main aspect of this topology is its flexibility to allow

sharing faces, edges and vertices between two different model entities, the absence of which was considered a modelling fault in polyhedral modelling. This allowed more spaces to be easily perceived in the same model and even free walls to be assessed, and the integration between indoor and outdoor performance analysis in the same model. This method was studied and analysed by (Picco and Marengo 2015; Chatzivasileiadi et al. 2018). These studies highlighted the need for balancing LOD in simulation models in a way that keeps the results relevant and efficient and for the time consumed to be bearable within the limits of an early stage of design. This balance was mainly found near the simplified geometries, not the extreme limits of turning the model into bounding boxes, but before this simplification stage.

There are many different approaches of realizing, categorizing and redefining the built environment and climate context based on the urban fabric, density and clustering (Schwarz 2010) or based on weather variables (Virk et al. 2015). This ongoing development of the definition and understanding of urban modelling will enhance the general research approach to urban simulation and modelling.

Table 2-1 Contributions to computational modelling complexity and level of details

Researchers	**Contribution**
March, 1971	Mathematical geometry
March and Steadman, 1971	Mathematical geometry patterns
Martin and March, 1972	Urban geometrical patterns
Biljecki et al., 2014	Urban computational modelling level of details
Jabi, 2015	Non-Manifold Typologies
Picco and Marengo, 2015	Energy model simplifications and their impact on heating and cooling results
Chatzivasileiadi et al., 2018	Balancing LOD simplification for simulation goals
Virk et al., 2015	Classifying urban models through weather variables

Table 2-1 shows the continuous development by different researchers to provide more computationally perceived geometry and facilitate how computers handle the built environment and allow designers and architects to utilize such applications. This helped other research to tackle urban complexity from the angle of its physical features and its spatial patterns: not just its mathematical representation. Furthermore, it enhances the accessibility of computational models and representations and the interoperability between different

modelling situations, whether for environmental simulation or modelling for design representations, or the illustration of some design data visualizations. This highlighted the importance of accessible, capable modelling tools that make the design process flow more easily between these situations.

These different interpretations and attempts to address urban complexity and the continuous efforts of quantifying it into logical patterns and flows allowed the rise of what became later called parametric urban design. Here urban models could be understood as a set of input parameters that generate the form of desired urban models. A change consequently follows the change of these parameters in the end product of urban models, all running under a set of rules or shape grammar flows, as mentioned earlier.

The literature states that complexity challenges the inclusion of simulation in the early stages of design. This challenge has been widely investigated on various scales. The simplification of models also impacts the accuracy of the results, which might have a consequent risk on the efficiency of the modelling and simulation results. Considering this, the literature argues that there is a threshold where this effect on results is insignificant. The model level of detail should consider the time constraints to inform the design process at its early stage.

2.3 Parametric urban design and its capabilities

As discussed earlier, parametricism was developed and proposed under the umbrella of quantifying the built environment into computational models generated by following a sequence of rules. Woodbury (2010) has discussed this by comparing the conventional and parametric design and building processes. According to his "*Elements of Parametric Design,*" Woodbury states that conventional models are easy to make and erase partially as the designer may directly sculpture the models. However, parametric models are easier to edit and adapt to changes requested in different design stages. They create a set of rules and relationships between the model's elements; the editing is taking place in these rules and relationships themselves rather than editing the models directly. This was discussed by Jabi (2013) in his definition of the parametric design system and how it is based on its inputs. Wortmann and Tunçer (2017)provided a more profound argument on the different parametric design methods and the qualities and merits of each approach. It was dependable on basic textual programming, visual programming or coupling of both the two approaches.

This research went through a practical comparison of how different projects had been exposed to parametric design principles and methods. It concluded with the significance of parametric design, not just as a way of iterating different variables but also as providing a more flexible collaboration medium, especially in the conceptual stage of design projects.

Parametric design has proved its benefits to creativity in design. Creating and modifying countless iterations with the change of parameters and algorithms gave the design a powerful capability. These algorithms have been developed to be genetically enhanced. The application of genetic algorithms strengthens this capability by widening the cognitive ability of the designers and enhancing the output solutions from these parametric algorithms (Lee et al. 2014; Lee et al. 2015).

Duarte et al.(2007) worked on developing a generative tool based on the analysis of the urban form and features of the old city of Marrakech, Morocco. The tool was used to analyse the street structure and buildings with courtyards to regenerate and interpret the old city growth and provide a clear understanding.

Pinto Duarte et al. (2005) and Duarte and Beirão (2011) took some steps further by implementing shape grammars in parametric model generation in both of their educational design studios. They tested the application of different shape grammars with different student groups after introducing urban contextual analysis and discussing the different scales of the built environment and the necessity of prioritizing shape grammar rules in relation to them. In these studies, student groups were made aware of a specific tool to be used in their

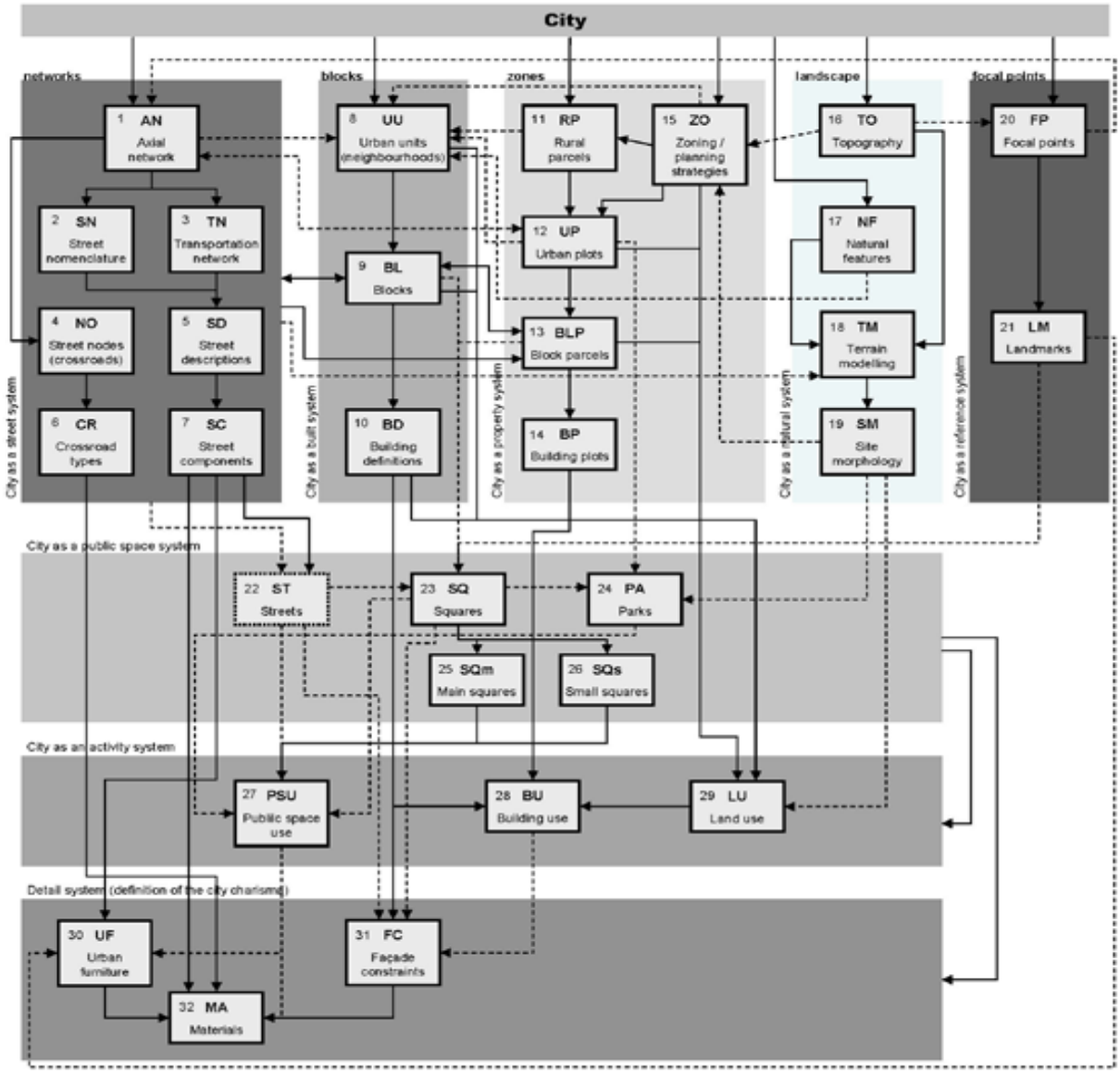


Figure 2-2 City Maker dismantling the inherited urban built environment (Biero 2012)

design process for a neighbourhood scale project. This research showed that introducing these tools enhanced the capability of dealing with urban complexity and empowered students with better design skills. Stouffs and Sariyildiz (2013) discussed the research concept for developing a tool for urban modelling. It consisted of different urban modelling related

platforms. In the methodology of this tool, CIM City Information Modelling was merged with a Geographic Information System (GIS) to add flexibility and better handling to the complexity of urban modelling. Beirão(2012) introduced the final version in the development process of this tool, City Maker, which dismantled the built environment on a city level to provide the ability of urban generation. Figure 2-2shows the methodology introduced for this tool in understanding the categorization of the urban built environment. Applying this tool in a visual programming language interface such as Grasshopper (Mcneel 2014) links it to be an application of the urban parametricism ideas introduced by Schumacher (2012). In his book, *The Autopoiesis of Architecture*, Patrick Schumacher introduced parametric urbanism and some examples from his teaching studios and practice at Zaha Hadid Architects. In these examples, they used Maya (Autodesk Inc. 2012) to build the urban models of these projects. He drove these urban geometrical experiments using specific features for each project, such as fluidity or a network of paths generated based on Frie Otto's minimal paths ideas. According to the book author, these examples deliver a new urban geometry with queries and concerns about its rationale.

Moreover, he tried to define these concerns by addressing them and illustrating the working methods within the design team, emphasizing the suitability of dealing with urban modelling complexity through a parametric approach. There was further discussion about these conventional urban design approaches in (Leach 2009), along with other ideas of interpreting the urban built environment by discussing urbanism as an organic biostructure (Roche 2009) or breeding fractal networks (Batty 2009) or relating urban economics to its spatial features

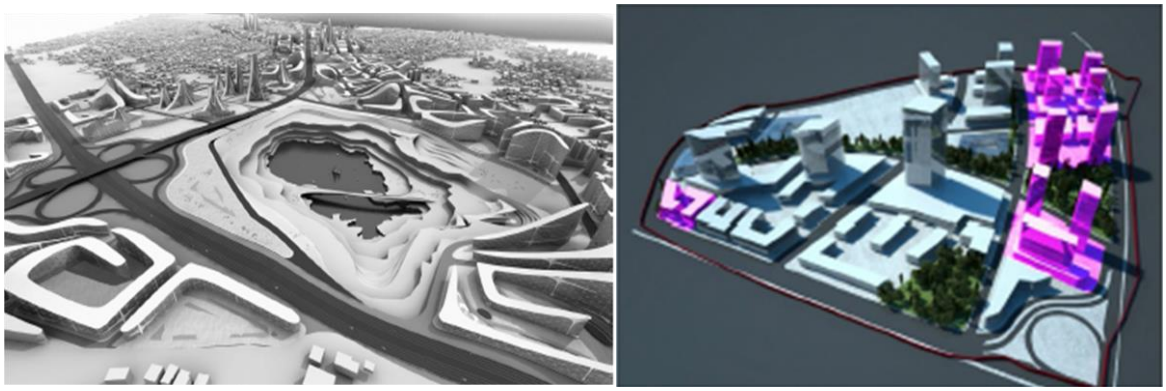


Figure 2-3 The difference between designer's initial proposal and final stages of the Kartal project as published in Caiskan, 2017 inherited from Kartal Municipality, Istanbul, Turkey

(Chiaradia 2009). On the other hand, Çalışkan (2017) expressed concern about parametric urbanism when provided with the result of one of the examples above after it went through

a modification process between the designer and the municipality to comply with building and construction laws, as shown in Figure 2-3

The resulting project differed somewhat from the original proposal. It affected the original concepts discussed in Schumacher's book about site heights and how they are derived by plot locations and interrelationship with the surrounding buildings.

Mandić and Tepavčević (2015) briefly review different approaches to shaping shape grammar in urban design and how these approaches built their generation logic and roles. Also, this study introduced Computer Generated Architecture (CGA) shape grammar which is the analytical application in urban design and its development sequence used in City Engine, which is another tool introduced by Parish and Müller (2001); with the addition to the generation of Grammars Derived Form Design (GDFD). The research compared their work to the City Maker work approach, claiming City Engine to be more flexible but limited to exposure in context variations.

Another way of data automation was introduced by Vidmar (2013). This framework was hosted on the SketchUp (Trimble 2016) modelling platform. It provides quick access to urban modelling data like Floor Area Ratio (FAR), land use distribution… etc., enabling a visual and fast way of data extraction in a simple modelling tool. However, it lacks the modelling automation needed to save time on the modelling process itself, and it inherits the automation limitations of the host platform.

Schneider et al. (2011) investigated the idea of developing a system of automating a layout in different scales of the built environment. The research introduced a system of generating zones on a generic scale attentive to a set of functions. Moreover, the study discussed different computational approaches to this generation, like evolutionary algorithms, agent-based systems, constraint-based systems and cellular automata. Koenig (2011) discussed a process of cellular automata generating urban scale geometries based on six typologies that experimented with a road network generation method. It showed the investigation of building-level generation and discussed the irregularity of the resulting mixed iterations and ways to avoid it. Koenig et al. (2013) developed a generic way to generate street networks on an urban scale.

Furthermore, König (2015) combined this investigation to introduce an open library for generating urban geometry, called CPlan. This library reintroduced Grasshopper in the visual programming language, adding more access (Koenig et al. (2017)). The tool is named Space Decoding. It is divided into analytical and generative urban geometry and provided a multi-scale generation from street to building plot control. This paved the way for more computer agency to have a more reliable role in the urban design process. As proposed by (König et al. 2017), the urban cognitive design consisted of a mixture of user interaction and machine learning trained in the evolutionary algorithm generation of urban geometry, as shown in Figure 2-4. Furthermore, it discussed the improvement of using a Self-Organizing Map (SOM) to classify and categorize iterations to reduce time and enhance efficiency in searching for the optimal solution.

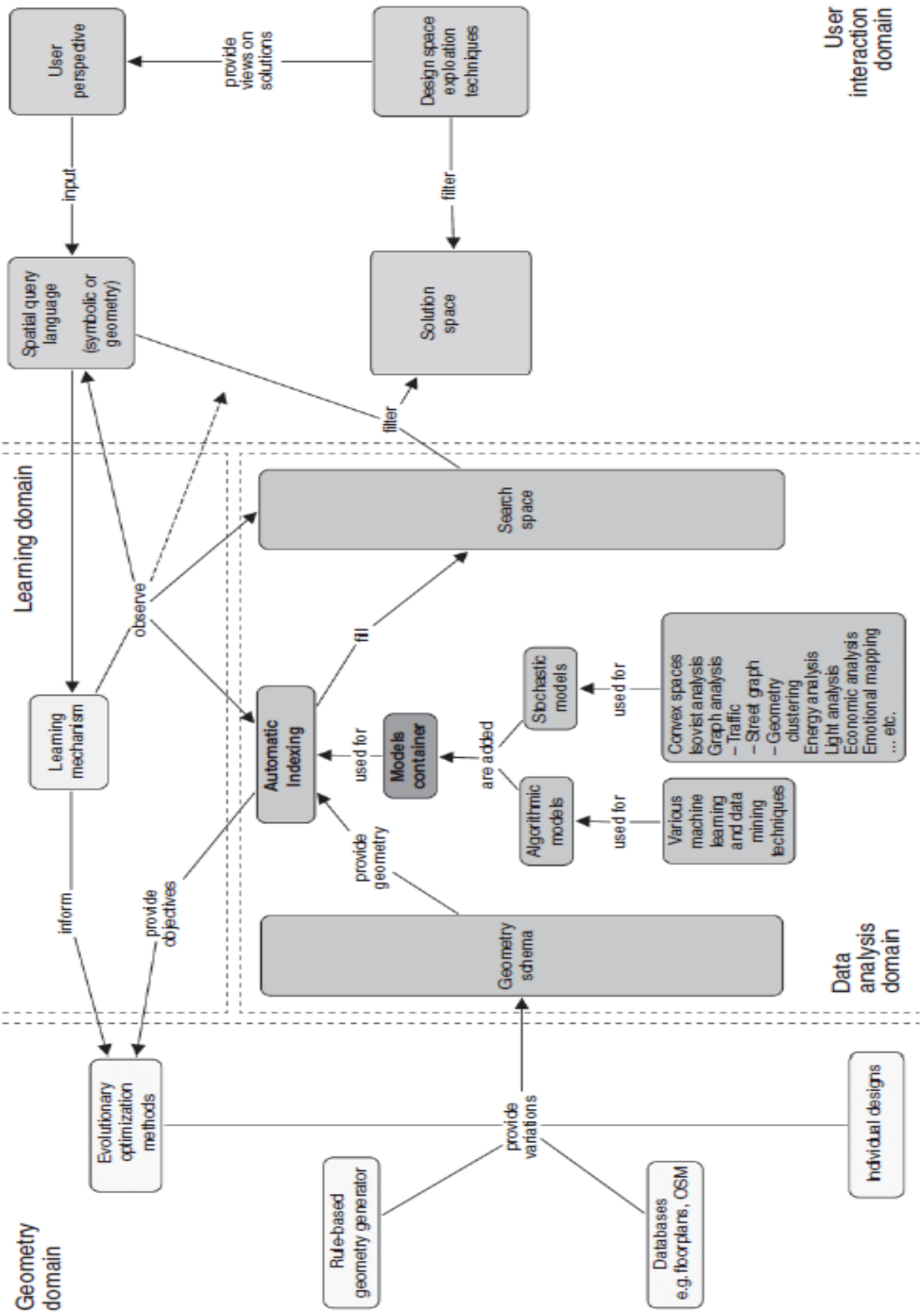


Figure 2-4: Cognitive urban design method structure (König et al. 2017)

In summary, the literature shows that computational methods have managed to quantify built environments and allowed for more control in their modelling techniques. Parametric design has emerged as a development of the algorithmic approach of quantifying the built environment geometry. The parametric design helped address modelling complexity at an urban scale by allowing for more control and flexibility in modelling. This led to more innovative approaches in design and urban forms. Some of these innovative outcomes clashed with functions and compliance with municipality laws. On another aspect of the development of parametric urban design, parametric urban design tools paved the way for more implementation of computational and algorithmic methods. Artificial intelligence and its application were widely introduced and tested within the platforms that host parametric design principles. This allowed for more investigation in urban design and urban performance optimisation to the limit that some literature considers the cognitive urban design approach, as mentioned earlier.

2.4 Built environment computational climate-based simulation and optimisation

One of the earliest simulation concepts which could be considered as one of the direct impacts of Martin and March (1972) in the field of built environment modelling and simulation is the Lighting and Thermal Method (LT-Model) introduced by Baker and Steemers (2003) as a tool of energy consumption simulation. This approach tried to estimate the effect and assess urban morphology and context on indoor energy consumption depending on MATLAB image processing. In terms of energy simulation, several methods have been developed to enhance the geometrical perception of space in the simulation model. In thermal models the simulation usually ignores the complexity of the space as long as the space model is introduced to the simulation process as a series of attached surfaces linked to each other to form one enclosed spatial configuration. On the other hand, solar radiation and wind assessment give a great deal of attention to the three-dimensional nature of spaces. This made it easier to reduce model complexity in thermal simulation to cut down the simulation processing time and even grouping similar zones into a larger one as they share the same thermal style. Dogan and Reinhart (2013) proposed a new technique called “shoebox” to do this simplification process automatically instead of doing it manually. This technique was used to create the Urban Modelling Interface (UMI) (Reinhart et al. 2013). On

an urban scale, Rodríguez-Álvarez (2016) came up with a similar way to simplify urban modelling for the energy analysis of existing urban cities. This is by creating a geometric interpretation of the urban geometry to create what he named the “notional grid”. This grid represents the most common geometric features of the urban fabric. It enhances the urban model's perception to do the energy analysis needed for the selected urban geometries. Bassett et al. (2012) illustrated a method for calculating urban solar analysis. This method was mainly by ray tracing the sky vault reflection on each building’s faces in the studied urban configuration. This was conducted using a plug-in for SketchUp (Trimble 2016) called Virvil. It also worked as a bridge between SketchUp and its energy simulation engine developed by Cardiff University, named HTB2 (Jones et al. 2013).

Although urban modelling and simulation have been the subject of investigation for a long time now, it is still developing and introducing new tools. The LT-Model is one of the earliest simulation tools developed by (Steemers 2003) to link indoor energy consumption to conditions in the outdoor urban context. Ratti et al. (2005) further illustrated this tool in a study conducted over three different sites to link urban geometry and indoor energy consumption by using the LT-Model applied along with an image processing approach using MATLAB (The MathWorks 2017). More development in urban modelling and simulation tools has been developed to introduce reachable and understandable user interfaces and more comprehensive analysis. ENVI-MET® is one of the most recognized simulation tools available, especially for the urban context, as they have enhanced their ENVI-MET 4 BETA II version to include indoor energy calculations enhancement (Bruse et al. 2016). With its sensitivity to vegetation along with its Computational Fluid Dynamics (CFD) simulation, it gained much recognition. It was used for simulation at an urban scale to assess different variables of urban geometry. It could also be used as a comprehensive tool to assess urban geometry and its thermal performance (Bouchahm et al. 2012; Taleghani et al. 2014; Taleghani et al. 2015). Bourbia et al. (2010) used it specifically to assess the role of albedo on urban canyon geometry. It is important to mention that this application has a drawback regarding its modelling flexibility. It only allows a model to be drawn on a pixelated grid, making it hard to build curved or free-form models.

Furthermore, it is not an open-source application that can only use the functionalities provided by the licensed version. This deprives users of developing their simulation process

or from coding a multi-objective simulation differently from those available by the application. The UMI is another comprehensive tool developed by the sustainable media lab at the Massachusetts Institute of Technology (MIT) (Reinhart et al. 2013). This tool is a plug-in on a Non-Uniform Rational Basis Spline (NURBS) modelling platform called Rhinoceros (McNeel 2014). This simulation tool depends on an energy modelling technique called "show box" (Dogan and Reinhart 2013). This technique simplifies the energy model by grouping similar spaces together and reduces the simulation running time. Another added value to this tool is its "walkability" feature (Rakha and Reinhart 2012; Rakha and Reinhart 2013). This simply allows the architects and urban planners to assess the walkability within a specific urban configuration, linking this to the case study's carbon footprint and adding the "walkability" as a variable for urban sustainable assessment and optimisation. UMI is used in urban energy assessment and optimisation once the "show box" method gets validated and finishes its ongoing development stage. It is important to mention that it gives results in square metres and cannot give the individual assessment to units in one building; the assessment works on the whole building as one entity that consists of a number of floors given as input to the simulation model. Also, the same team have developed a tool for an urban daylighting assessment called Urban Daylight (UD) as a plug-in for the visual programming language platform Grasshopper hosted by Rhino as a modelling interface. UD was incorporated later in a more extensive set of components called Archisim (Dogan et al. 2014; Dogan et al. 2016), which included more energy analysis capabilities on an urban scale while it still needs more validating studies. Another tool using the same platform is Diva for Rhino (Jakubiec and Reinhart 2011) and its more parametric extension to Diva for Grasshopper (Lagios et al. 2010). Diva is mainly for indoor lighting analysis based on Radiance (Ward 1994). Its earlier versions had an energy assessment capability based on Energy Plus engine (U.S. Department of Energy's (DOE) 2016) but for only one zone.

From a planning point of view, City Energy Analyst (CEA) is a new tool developed to assess energy on an urban scale (Fonseca et al. 2016). This tool is written using PYTHON coding language to be hosted in the GIS application ArcGIS (Esri 2016). This tool opens the door to infrastructure assessment regarding energy efficiency and carbon emissions at an urban scale. Also, it analyses implementing photovoltaics as part of an energy source to the urban configuration. It allows assessing future scenarios of urban growth and the cost of

implementing these scenarios due to its GIS nature. CEA was not yet been made available for users by the time this literature review was done. The first available version dates to February 2016. its capabilities and limitations were not yet evident as it was still in the development stage.

Dallal and Visser (2015) introduced another tool programmed with Python programming language to generate a climate-responsive urban morphology for hot arid zones. Their study was based on giving almost three choices for each variable, and the algorithm merges the optimal solution to reach the optimal performance for the targeted urban community. The investigation included different urban geometrical parameters, such as the distribution of green areas, compactness of urban configuration, spacing between buildings and the distribution of heights in the configuration. The results of this study are only the percentage of enhancement the tool provided for a specific design considered as the benchmark for the study. Also, it was concluded that there was a need to add more parameters to the process to achieve a more comprehensive approach.

Nault et al. (2013) presented a framework of urban solar potentiality analysis based on sensitivity analysis to create a more comprehensive simulation approach. This analysis linked the building performance and urban neighbourhood geometrical features. Peronato et al. (2015) further elaborated on the same framework, mainly based on Rhinoceros and Grasshopper for modelling the geometry and the MIT developed simulation tools (UMI, UD, Diva for Rhino and Diva for Grasshopper). They used MATLAB (The MathWorks 2017) for post-simulation data processing. A research study by (Nault et al. 2015a) evaluated the broader application of this framework. It provided a review of the commonly used tools for simulation related to heating and cooling and solar potentiality. The results of this study emphasized the impact the compactness and fenestration ratio have on building performance and solar potentiality within an urban configuration. This provided help and aid for decision-making regarding the thermal performance of the urban configuration in the early stages of design.

With regard to a comprehensive and integrative approach in simulation methods, Sadeghipour and Pak (2013) have introduced new comprehensive simulation tools based on the Rhino 3D modelling platform and its Visual Programing Language (VPL) interface, Grasshopper. This set of tools started with two plug-ins, Ladybug for climate analysis and visualization (for example, sun path and shadow visualization) and Honeybee for building

performance simulation (for example, thermal and lighting analysis). These tools are based on Day-Sim (Reinhart 2017), Radiance (Ward 1994) and Energy Plus (U.S. Department of Energy's (DOE) 2016) simulation engines. These tools have the added value of being open-source tools improving the capability of flexibility in the simulation models and adding a great deal of Energy Plus capabilities than allowed by other simulation tools, such as DIVA for Rhino. Ladybug and Honeybee enhanced integrative simulation and design methods by avoiding the use of multiple platforms for the same projects and combining multi-objective parametric performance tools into one method and process.

Furthermore, Ladybug allows the designer to import back the geometry analysed and visualized in the Rhino/Grasshopper platform. In addition to its comprehensive integrative approach, it can benefit from working on a parametric VPL as Grasshopper to generate further iteration from the simulation outputs. Some studies have implemented Ladybug as a climate visualization tool conducting the simulation over another simulation engine (ANSYS for CFD) with the benefits of the parametric modelling of Grasshopper, as in (Taleb and Musleh 2015). Others used it within a series of simulation tools (Anton and Tănase 2016) or mainly depended on it to conduct the simulation analysis while applying a genetic algorithm (Naboni 2014; Calcerano and Martinelli 2016).

Jones et al. (2007) introduced a tool for assessing the scale of urban performance called the Energy and Environmental Prediction model (EEP). This tool was developed in the Welsh School of Architecture (WSA), Cardiff University. Regarding its energy analysis, this tool was based on the UK Government Standard Assessment Procedure for residential buildings (SAP), which is an energy assessment methodology (BRE and DECC 2013). EEP included different sub-models for analysis, such as traffic, health and, of course, energy performance for domestic and non-domestic use. It was mainly built upon a GIS model. In addition to the regular geometrical variables investigated by EEP like the number of stories, storey heights and window-to-wall ratios, it added the age of buildings as a variable of study to assess the insulation capability and the heat gain/loss dependent on this variable. Later on, this group developed an early stage design energy performance assessment tool named HTB2 (Jones et al. 2013). This tool opened the door for a multi-objective comprehensive urban simulation in the early design stage, either on a single building scale or on an urban scale. Jones et al. (2009) illustrated that HTB2 could investigate urban geometry orientation, overshadowing between

buildings, thermal insulation, and internal heat gain from appliances and lighting. It is important to mention that this study has investigated these variables separately to assess their potentiality regarding energy optimisation for a proposed urban master plan. The WSA team also developed a plug-in for HTB2 and named it Virvil (Bassett et al. 2012). It is a bridging plug-in to link HTB2 as a simulation engine to SketchUp (Trimble 2016) as a simple modelling interface. Virvil simply works by ray tracing the sky vault reflection on each face of the building, either it is a façade or a roof. This reflection represents the solar radiation on the building. With the help of HTB2, the energy performance can be simulated from this reflection and the buildings' variables as an input to the process. Then, Virvil translates the simulation outputs back again on the SketchUp model. Dealing separately with each face of the model gives this framework more resilience and compliance to each model's needed specifications. Its hourly-based simulation capability and the control of ray angle separation for each face give the user much more control. With the expectation of adding more urban context features to the tool's recognition, such as wind breeze and the enhancement of its sensitivity, this framework can be highly promising with regard to the need to reduce the time consumed to run the simulation.

One of the leading developers of software services in the built environment and construction industry is Autodesk (Autodesk inc. 1985). It has multiple applications for simulation in the early stages of design. By 2015, Ecotect, one of the most commonly used tools for built environment performance analysis, stopped selling licenses. It became partially integrated within other applications developed by the same company as Revit. This integration was developed further to be what we now know as Autodesk Insight (Autodesk.inc [no date]), a plug-in that enables building performance analysis within the Revit platform. Another performance simulation is Green Building Studio (GBS) (Autodesk Inc. 2017b) which was released by Autodesk close to the discontinuation time of Ecotect to be its fourth runner for building performance simulation afterwards. It is promoted as a holistic tool for the simulation of multiple performance features not restricted to thermal performance but also including basic CFD to simulate wind energy; it reacts around the building's geometry. Although it is not an open-source application, GBS is based on the DOE-2 (Simulation Research Group 2017) simulation engine, an open-source tool for energy performance calculation. Its access to Building Information Models (BIM) generated by Revit gives it many advantages

along with its web-based simulation version. Its validation studies were done and submitted to the United States Department of Energy (U.S. Department of Energy 2008), yet peer-reviewed publications utilizing these capabilities are less promoted by the developers, unlike its rivals.

Attia et al. (2012) tried to create selection criteria for performance simulation tools by surveying architects and engineers. The study showed some differences between the preferences of the two selected groups. The main features considered in this study were the tool's integration within the design process, interoperability with building modelling, accuracy and level of details, usability and information management and integration of knowledge-based design intelligence. These features can be varied between the available state-of-the-art tools for urban modelling and performance simulation tools. The key feature that can be added to these criteria is the capability of conducting multi-objective optimisation in a complex urban scale model. The major challenge, continuously mentioned in most of the studies introducing the simulation performance tool, is its utilization in the early stage of urban design, as simulations tend to consume time that is not always affordable. There is still a need at the early design decision-making stage for a comprehensive tool that can conduct performance simulation at an urban scale within a reasonable time.

Table 2-2 Summary of the simulation software and its capabilities and limitations regarding urban simulation

	Analysis capabilities	**Name of product**	**Modelling platform**	**Availability**	**Source code availability**
Baker and Steemers, 2003; Ratti et al., 2005	Microclimate and indoor energy consumption	LT-Model with MATLAB image processing	stand alone		
Reinhart et al., 2013	Urban performance (energy, walkability, etc.)	UMI and "Shoebox" simplification technique	Rhino	Available for Free	Not open source
Rodríguez-Álvarez, 2016	Urban energy demand	UEIB &UEIB-GIS with "Notional Grid" simplification technique			
Bassett et al., 2012	Urban solar & energy analysis	HTB2	stand alone	Available for free	Not open source
Jones et al., 2013	Urban solar & energy analysis	Virvil	SketchUp	Available for free	Not open source
Jakubiec and Reinhart, 2011	Urban daylight analysis	Urban daylight	Grasshopper	Embedded in another tool	
(Dogan et al. 2014; Dogan et al. 2016)	Urban energy performance	Archi sim	Grasshopper	Available for free	Not open source
Lagios et al., 2010	Indoor lighting analysis	DIVA	Rhino & Grasshopper	Available for free for educational use	Not open source
Lagios et al., 2010	Urban energy and infrastructure analysis	City Energy Analyst on python coding	Arc GIS		
Dallal and Visser, 2015	Urban energy	PYTHON coding	blender		
Nault et al., 2013, Nault et al., 2015 & (Peronato et al. 2015)	Urban solar analysis	Urban solve	Grasshopper		
Sadeghipour and Pak, 2013	Holistic performance simulation	Ladybug tools	Grasshopper	Available for free	Open-source
ENVI-MET GmbH et al., 2016	Holistic performance simulation	Envi-Met	stand alone	Available for free	Not open source
Trigaux, Allacker and Troyer, 2014	Urban energy performance and solar analysis	EPB+	Stand alone		
(Autodesk Inc. 2017b)	Holistic performance simulation	Green building studio	Stand alone	Available for free for educational use	Not open source

Lastly, it can be argued that recent developments of simulation tools have provided a gateway for the leading simulation engines to be introduced to some parametric modelling platforms. In addition, there is a consistent development of these tools to achieve a holistic simulation approach that can deal with the built environment as a whole on both the architectural and urban levels, as shown in Table 2-2. On the level of urban performance simulation, different tools are trying to simplify urban models to allow the simulation process to be conducted reasonably, allowing the decision-making process to stay efficient in the early design stages. Although some of the developed tools accommodate the parametric iterative approach, the time cost of conducting these simulations still requires more simplified or limited iterations to be done within the early stages of design.

2.5 Urban geometrical variables impact on urban performance

The early design stage has recently gained much attention, especially in urban performance simulation and optimisation. In addition to the research on natural meteorological effects on the urban context, researchers have investigated the role of urban geometry and its effect on the urban context. Urban geometry is formed by various variables, each of which affects the urban context on different scales. These variables need more investigation to understand better their relative importance on different aspects of urban performance. There is a recognized amount of ongoing research studying this relation between urban geometry variables and their effects on the urban context and its thermal status. This relation has been studied from different approaches and perspectives. Some researchers focused on identifying this relation and providing a better understanding of these different effects. Others tried to link it to other urban aspects, such as density, walkability, or indoor environment quality.

Trigaux et al. (2014) introduced the Life Cycle Assessment (LCA) calculation framework. This framework aimed to evaluate the energy consumption of a medium-density simple urban configuration on a neighbourhood scale based upon the solar exposure of each unit in the configuration. Following this, Trigaux et al. (2015) linked the urban context to indoor energy consumption by developing an enhanced simulation tool based upon the Flemish Energy Performance of Buildings (EBP). They named it EPB+ and used the tool to analyse the solar irradiation received by each unit to state its heat gain and consequently the energy consumed to reach thermal comfort for each unit. The authors compared the results between EPB+ and Energy Plus (U.S. Department of Energy's (DOE) 2016) to validate them. This experiment was

done on one fixed urban courtyard configuration with other studies on different urban geometrical configurations to follow. Also, it is important to mention that it was done monthly and ignored the building shadowing on the ground and its effect on solar irradiation in the urban geometrical selected case study.

2.5.1 Building typology

Taleghani et al. (2014) have carried out further investigation related to the configuration of the urban courtyard. They used ENVI-Met (Bruse et al. 2016)as the simulation tool to run different iterations of urban courtyard orientations and 2D dimensions. The research investigated the urban mean heat radiant. In addition to these iterations, the study added vegetation, water and albedo as additional variables of urban geometry, based upon a 2050 weather forecast as inputs to the simulation. This study has provided a clear vision for studying the orientation of urban courtyards. In addition, it discussed which element to add to enhance the outdoor thermal status in a closed urban configuration. On the other hand, height was not included in this study, although it has an important influence on outdoor solar irradiation and thermal status. Moreover, Taleghani et al. (2015) added more simplified geometrical configuration alternatives in their study based on existing standard urban fabrics in the Netherlands with further investigation on the height iteration effect on these alternatives.

A link between height and space floor area was provided by Panão et al. (2008). This study investigated the height through its relation to the number of floors and area covered by the buildings in the site, floor space index (FSI) as introduced in the paper, considering the spacing between buildings as canyons and pavilions. In addition to these variables, the study analysed the iteration of orientation for the urban configuration and the building length in relation to its FSI. This study was done using a genetic algorithm to enhance the alternatives to reach the optimum solution. Applying this approach in urban geometry analysis to energy consumption optimisation opened the door for further research to be done along these lines with more urban geometrical variables and alternatives, rather than simplified grid configuration as the study has done.

Kampf and Robinson (2010) introduced a randomized methodology to get new shapes of buildings' roofs that led to a new way to determine heights in urban geometries. This research

targeted the optimisation of solar energy applications in urban geometry. This explains a lot about their concentration on roofs and their direction in response to solar irradiation within defined site constraints. They used Radiance to ray trace predicted solar irradiation on roofs in the urban geometrical form.

2.5.2 Urban layout

Yi and Kim (2015) also used a genetic algorithm tool, Grasshopper (Mcneel 2014), Galapagos (Rutten 2013), to analyse solar irradiation for the urban geometrical configuration of high-rise buildings. This study targeted optimizing the solar irradiation exposure for a configuration of six high-rise buildings. They did this by setting several geometrical variables, not restricted to height or floor areas, but by adding scaling, tilting, and building orientation. This study controlled the basic geometrical aspects of the buildings in this configuration through a pivot point. This means the change applied to this point will be applied to all the buildings. This idea helps in accelerating the analysis, especially with the application of the genetic algorithm approach as a major part of the optimisation process will be computerized. On the other hand, it limits the alternatives that could be generated through the randomized iteration of geometrical variables.

Although orientation has been investigated previously by Panão et al. (2008), Taleghani et al. (2014) and Taleghani et al. (2015), Vermeulen et al. (2015) analysed it along with heights, scaling and positioning. The study applied an algorithm to investigate the rotational capabilities of each building in the urban geometrical configuration with limitations for the maximum iteration. Also, the context was added as a grid of buildings surrounding the site. However, it should be noted that this study depended only on direct solar irradiation on surfaces and compensated for the absence of the inter-shadowing effect by Atherton et al. (1978) method. This method is only applicable on polygonal lines, which adds more geometrical limitations to this simulation process as it weakens the opportunity to examine curve shapes with the same methodology. This result, considering the clear sky conditions and the fact that it has not been based upon meteorological data, shows that many factors can be considered if this method was applied in a comprehensive urban geometrical urban energy simulation study.

The positioning of buildings within the urban geometrical configuration was examined with a genetic algorithm (Conceição António et al. 2014). This study aimed to evaluate solar irradiation, taking into consideration building shadowing interrelationships. They studied several positioning scenarios between two buildings in an urban grid, and the resulting inter-shadowing in these scenarios generated different urban grid layouts with different positioning alternatives. Then they investigated grouping heights in the configuration to determine the optimum solution for the proposed site. This study suggests considering the indoor environment and comfort needs in further future simulations.

Moreover, it is recommended by the research to consider more variables such as size, the shape of buildings and facades' geometrical modifications. This provides a better comprehensive understanding of urban geometry and its variables which can be considered in urban simulation modelling that can be briefed as shown in Figure 2-5

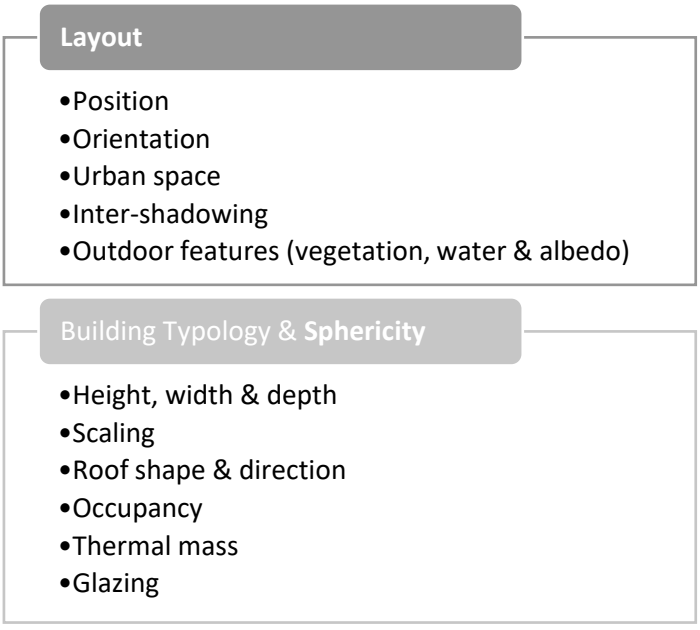


Figure 2-5 Summary of urban geometrical variables as illustrated through the literature review of previous studies

As shown in Table 2-3, each variable has been covered individually or in groups in one or more research approaches. Most of the variables have been assessed concerning their impact on thermal performance or solar irradiation. However, most research outcomes have recommended further investigation with more case studies and more comprehensive examination.

Table 2-3 Geometrical variable investigated based on its performance impact

References	Geometrical Variable	Performance Aspect
Taleghani et al., 2014, 2015	Building typology (urban courtyards) & orientations	Urban mean heat radiant
Panão et al., 2008	Floor space index & orientation & urban spaces	Energy consumption
Kampf & Robinson, 2010	Building typology (roofs)	Solar irradiation
Yi and Kim, 2015	Building typology (tilting), orientation, height & scale	Solar irradiation
Vermeulen et al., 2015	Positioning & scale & height & individual building orientation	Solar irradiation
Conceição António, Monteiro and Afonso, 2014	Positioning, inter-shadowing and indoor thermal comfort	Solar irradiation
Trigaux et al., 2014	Urban courtyards and heights	Solar radiation and energy demand
Nault et al., 2013, 2015 & Peronato et al., 2015	Different hypothetical urban configurations	Solar radiation and energy demand
Dallal and Visser, 2015	Urban voids, heights, orientation, Massing	Energy consumption and passive strategies

As shown in the literature (see Table 2-3), geometrical variables have been tested and simulated at the architectural and urban scales. These simulation studies dealt with different performance aspects from urban mean heat radiant and buildings' solar radiation to indoor energy demand. It can be noticed that the geometrical variables can be grouped into two categories, as shown in Figure 2-5. This categorization also followed the scale of simulation conducted for these studies. In addition, it is repeatedly recommended from different literature to conduct a comprehensive study that includes a larger pool of iterations and geometrical variables from the two different scales. Also, there is a need for a holistic analysis

of different performance aspects and their correlation to understand better the balance between these aspects and the effect it gets from varying geometry on an urban design scale. However, most of these recommendations are bounded by the limitations of the time cost on conducted research and the practical capabilities of the simulation engines utilized in these studies.

2.6 Built environment performance optimisation

Optimisation is looking for the best iteration in a group of iterations or options (De wilde 2018). This definition is introducing optimisation as a decision-making process taken within the project's design and analysis phase. In his book, *Building Performance Analysis*, (De wilde 2018) discusses the utilization of this decision-making process, either to choose a specific element like an optimal material or to find some optimal performing designs or settings of design parameters based on a preset benchmark or to combine both approaches in the search for the optimal solution for a design problem. This design nature had an open discussion in Cross's (2001) review of design discipline versus design science. He reviews the nature of design problems in the built environment and the debate about ways to “scientize” them. This research can partially relate to this dialogue as “quantifying” the design problem. This review also discusses design methods and definitions and how they can be addressed in different ways. This differentiated between the knowledge of design that needs to be transferred, copied and inherited and the design act itself, which must not be copied or inherited from previous practices. As mentioned earlier, the urban context is a complex topic. However, analyzing and optimizing urban performance is a quantifiable process in nature which can be reused and inherited between different design practices. Given the complex nature of urban models, urban numerical optimisation would be exposed to ideas like hierarchal optimisation techniques (Choudhary et al. 2005), which allow breaking down the design problem resolution into a multi-stage process of decision-making and optimizing individual simple parts of the dialogue about dealing with the design problem. However, Hazelrigg (1998) recommended that this might lead to misleading results in the optimisation process and that the optimisation process should look holistically for a comprehensive objective. Urban performance optimisation has multiple applications in different studies. Nguyen et al. (2014)provide a literature review of the simulation-based optimisation methods and tools that also discusses the different algorithms used for the simulation-based

performance optimisation and how frequently each algorithm or method is applied. These algorithms represent a way of looking into a consistently changing search space due to the generative nature of the simulation inputs. This generative computational added some flexibility because traditional optimisation is limited by the iteration pool, which gained more solutions by being computationally generated. This computational assisted process attempts to overcome the challenge of time cost in the early stages of design. Furthermore, it discussed the efficiency of artificial intelligence and different applications and solvers assisting the simulation-based optimisation process computationally.

The investigation of applying artificial intelligence in the built environment computational methods has been ongoing for decades now. This application varied between the different tasks and roles artificial intelligence can conduct within the design and construction processes of the built environment. Referring to artificial intelligence systems as expert systems, Kim et al. (2012) identify these expert systems like computer programs that apply artificial intelligence (AI) principles to solve a predefined clear problem. They continue to explain this definition by breaking down the expert into the knowledge base, which is the set of definitions and facts that the system is dependent on for taking actions, listed as the control mechanism. The control mechanism is the action the system is set to take by conducting it computationally; the most straightforward example for this is the logic of “if [event], then [action]”. According to this review, the data resulting from these actions during the generation process is called the working memory. The study also summarised the roles that artificial intelligence can perform, focusing on urban planning and design. These roles were 1) interpretation, which is providing understandable data to users by utilizing these systems as a mediator for the generic data output of the process; 2) diagnosis as an application of these systems to detect specific patterns, design and planning to evaluate iteration based on analysis; 3) monitoring and control which means the capability to control the behaviour of the tested variables based on a series of variant inputs; and, 4) instruction which is the utilization of these systems to transfer a set of knowledge to prospective users. (Koenig et al. 2020) have introduced another historical review of computational optimisation methods in the built environment. They highlighted the challenges facing evolutionary-based optimisation tools. Some of these challenges are the aesthetics of proposed optimal solutions and the computational and time costs threatening the feasibility of these methods. The

research also proposed assistance using image processing techniques to overcome some of these challenges. Additionally, they did not specify a clear pathway of handling such a database or the exact scope of using their data-driven optimisation framework.

Genetic algorithms can be explained as the generation of a first population based on the initial input enhanced by changing parameters to meet the desired target. Then, the first population get assessed in compliance with the desired target. If this does not meet the target, the algorithm will populate the next enhanced generation to reach a closer result by changing the parameters entered in the base model. After that, the better alternative will be the base from which the next generation will start. It will continue to generate more alternatives until it reaches the target or the pre-installed number of generations, whichever is closer.

With this impressive ability for the design in general, parametric urban design can be represented as a group of arranged buildings and urban geometrical variables shaped by scripted algorithms. This interpretation provides a different vision and capability for investigating urban design, geometry and performance (Schumacher 2009; Schumacher).

Genetic algorithms were applied and linked to environmental analysis earlier than these interpretations, as in (Caldas and Norford 2002), who applied them on Autodesk AutoCAD (Autodesk Inc. 2017a), linking it with the DOE thermal and lighting analysis program developed by the United States Department of Energy (Simulation Research Group 2017) to study the sizing and placing of windows in office buildings. (Panão et al. 2008) introduced the application of a genetic algorithm to the process of performance analysis on an urban scale. The study linked urban geometrical variables in an urban grid configuration to energy optimisation. It used this link to search for the optimal solution regarding the number of floors, heights and orientation of the urban configuration using the help of a genetic algorithm. Furthermore, (Mackey et al. 2015) emphasized the impact of climatic conditions on the geometry represented in outdoor shade. They applied a genetic algorithm to search for the best bus stop shade in three different sites with different climatic conditions. (Rakha and Nassar 2011) avoided this geometrical limitation while applying a genetic algorithm for a model of indoor ceiling optimisation with natural lighting. Also, due to the unfeasibility of investigating all the possibilities and alternatives, they limited the number of iterations and populations generated from these initial iterations presenting a methodology to search near an optimum solution but not defining it strictly. While (Yi and Kim 2015) used the genetic

algorithm capabilities to investigate a bigger model consisting of the urban configuration of six high-rise buildings. This study manipulated different geometrical variables to search for the optimal performance solution for the urban configuration. (Suyoto et al. 2015) used this parametric genetic approach to conduct a more comprehensive study for office towers. This study extended the use of parametric genetic algorithm analysis to analyse and optimize more than just the environmental aspects, such as heat gain and glazing skin material and design and a building's overshadowing. They included the structural analysis of different massing iteration along with the pedestrian walking path through the urban configuration. (Calcerano and Martinelli 2016) applied a simple genetic algorithm to find the optimal tree positioning around a building in order to search for the optimal energy consumption used for cooling. (Song et al. 2016) introduced Implicit Redundant Responsive Genetic Algorithm (IRRGA). This study implemented a genetic algorithm approach to present a form-finding approach for apartment buildings with a multi-objective fitness criteria. IRRGA was mainly controlled by symmetry, structure, connectivity and cost as a fitness criterion for the optimally selected forms with no interest in the indoor plan design for the generated forms.

(Gerber and Lin 2014; Lin 2014; Lin and Gerber 2014) introduced a comprehensive genetic architectural framework called Evolutionary Energy Performance Feedback for Design (EEPFD). This framework is a multidisciplinary design optimisation framework for the early stages of design. It is coded by MATLAB (The MathWorks 2017) and based on the building information modelling tool Autodesk Revit (Autodesk Inc. 2017c) as its modelling interface. It depends on Autodesk GBS (Autodesk Inc. 2017b) to conduct further analysis. This framework extends the link between form-finding and energy consumption simulation to include more than just the geometrical variables and decide the number of floors allocated for each occupancy in a complex multi-functional building and where to position these occupancies relatively. As powerful as it seems to be, EEPFD needs to be validated through various design scenarios. It is also not clear if it has the capability of conducting an urban scale genetic analysis. Moreover, it inherits the modelling limitations Revit has because of its family/massing data structure.

Grasshopper provides an alternative regarding modelling limitations because it is not based on the family data structure. Furthermore, there are various genetic algorithm solvers available as plug-ins for Grasshopper. Galapagos (Rutten 2013; Jin and Jeong 2014; Yi and Kim

2015) is the default genetic algorithm solver in Grasshopper. Octopus (University of Applied Arts Vienna Bollinger+Grohmann Engineers. 2014a) is a more flexible solver regarding the fitness control and evaluation and history saving within Grasshopper but is not yet an open-source plug-in. It has a heavy load on computational resources due to its visual nature. Biomorpher (Harding 2017) is one of the latest released solvers for Grasshopper that adds more user interaction to the genetic algorithm process by allowing users to choose the parents of each generation from a group of clusters distributed by the solver based upon relativity to the fitness. Moreover, this is an open-source tool that allows other designers to add/modify its main code. There are also other solvers which use Grasshopper as a platform to apply genetic algorithm approaches. Another approach to enhancing the application of genetic algorithm principles in the parametric environment was to allow more control over the variation breeding and mating in different generations. This enhancement was achieved by introducing an interface called “Wallacei” (Makki and Showkatbakhsh 2018; Makki et al. 2019; Showkatbakhsh and Makki 2020) to conduct such a process and analyse the optimisation process outputs. It started as an add-on to the commonly used genetic algorithm solver interface, Octopus, and in its recent releases, the team has implemented their genetic algorithm code. This allowed more visualization of the genetic algorithm process in different stages, leading to more user agency over the optimisation process to overcome a challenge known about genetic algorithm solvers, of being uncontrolled “Black box” tools (Wortmann and Nannicini 2016; Wortmann 2017). Surrogate models are about approximations done in the design space, so instead of the random searching for optimal solutions, the design spaces are just approximated, and the approximated first solution is the one that gets simulated. Then the design space becomes the “approximate” design space. This approximated space is defined mathematically based on some chosen design iterations within the design space (Wortmann et al. 2015). This method can be claimed to be a middle ground between fully simulating the looked-up iterations in the design space and approximating the space to some local area with a smaller number of simulations and consequent saving of time. This method is also implemented in a plug-in in the Grasshopper parametric platform under “Opossum” (Wortmann 2017; Wortmann and Tunçer 2017). An analysis of multiple Multi-Objective Optimisation methods was introduced in (Wortmann and Natanian 2020). This research highlighted the importance of using machine learning-related methods to optimise energy performance on the urban scale. It showed the potentiality of utilizing data analytical

techniques to optimise urban geometry compared to depending solely on algorithmic evolutionary methods.

Other computational solvers try to deal with the optimisation process in the parametric environment, such as swarming, which is a naturally inspired algorithm that resembles the bird flocking biological system. It is based on multiple searching agents which exchange data between each other while avoiding colliding and closing up towards the location of the optimal solution. The solver interface for swarming principles in the Grasshopper environment has been named “Silvereye” (Cichocka et al. 2017).

Moreover, there has been continuous development to include machine learning methods and principles under the notion of built environment optimisation. Generally, the significant difference in machine learning is where the problem or parts of the problem are predefined and pre-introduced to the solver. This means the algorithm mainly predicts results for the problem based on preset “training data” (Krarti 2003; Zhao and Magoulès 2012; Dounis et al. 2014). Machine learning methods have different mathematical models, each of which follows a particular way of predicting the results of the proposed problem. One of the models, Artificial Neural Networks (ANN), aims to imitate how the human neural system works (Cui and Cai 2013). This method is gaining attention to predict and optimize different urban performance aspects such as radiation and life cycle assessment of carbon footprint (Płoszaj-Mazurek et al. 2020; Rahbar et al. 2020). These studies highlighted the potential of using this method of handling such a large set of data attributes of environmental performance aspects on the urban design scale due to its ability to reach acceptable accuracies efficiently. An overview and comparison of the different mathematical models are presented in Ali et al.'s (2018) study utilizing different models to predict the Energy Performance Certificate (EPC) for a large number of residential buildings.

Quantifying the design process is an approach that has been developed over a long time, and it has multiple frames of reference. One avenue of research is to seek better performing design through the optimisation process. Optimisation also has been investigated through different paths. Artificial intelligence assisted optimisation has been utilized through different studies to find optimal performing solutions. The literature shows that the mathematical models for artificial intelligence optimisation can be categorized into groups. One group is where the problem is unknown and unresolved, and in this case, the optimisation solver is

searching the pool of iteration with no guidance. The other group is to give the solver a set of pre-tested results to guide it through the process of searching for an optimal solution. The third group is where those two methods are merged. This is directly linked to the built environment performance when it comes to face the time challenge of benefiting from the performance simulation results to guide the early stages of design. The literature also shows that some multiple applications and interfaces may be adapted in the parametric environment allowing more generative designs to be simulated and optimized.

2.7 Summary

Quantifying the built environment can be linked to the theoretical framework of categorizing it into patterns and grids. This process emerged from theoretical to mathematical phases until it could be applied visually in computational modelling. Although attempts have been made to make urban computational modelling accessible and more capable from earlier stages in developing urban modelling and the trials to quantify it, some challenges face urban modelling and its integration into urban design. One of these challenges is urban complexity and the level of details for the models and determining it based on the model's aimed function. Taking urban performance simulation as one of these aims, the literature repeatedly shows that time also forms another challenge to the capability of modelling the urban environment and simulating its performance. Most of the research that deals with urban modelling for simulation purposes aims to simplify these models to make it easier to inform the design decision-making in its early stages. This simplification varied, either by classifying the different models based on similarity; by limiting the geometrical features of the model itself; by assigning different levels of details and attributes for different functions to models; or by finding a new mathematical representation for models that are faster to simulate and model by reducing the number of faces or number of functions the model needs to get to its final stage.

In recent years, urban modelling exceeded the stage of static modelling to achieve iterative and generative modelling. This has been provided by the application of parametricism in urban modelling. This led to the control of urban models on a better-detailed level. The model is generated through dynamic and responsive parameters that generate a pool of iterations by using these parameters. In this way, urban models can be recognized as parameters rather than dealing with their complexity as a whole entity. Another benefit from generative urban

modelling is to be able to draw out the impact of individual parameters on performance within a contextual setting. This gave a better insight into urban performance and its drives. The literature also discusses the conflict between parametric urban modelling and design, in some cases, where the local laws and functions have not been implemented in the decision-making process. This has emphasized the importance of adding functionality to the geometrical forming process.

The investigation of urban geometrical parameters and their relative impact on urban performance has been demonstrated in various studies. The literature has illustrated different trials to mitigate urban performance by varying geometrical parameters. This chapter showed and discussed the categorization of urban geometrical variables into two groups based on this scale. It has shown research studies in the literature dealing with different aspects of urban performance and optimizing it. The limited number of limited iterations is one of the repeated challenges of the reviewed studies. Another challenge is the lack of urban holistic simulation ability due to the lack of capable tools and the simulation time cost.

Furthermore, the quantification of the design process also helped enhance the application of optimisation principles based on specific performance goals. The chapter presented the optimisation principles in the design decision-making process. It explained some of its different artificial intelligence mathematical models and reviewed some studies that tried to apply them. In this chapter, the study has also displayed some tools that act as an end-user interface in the parametric environment to provide easier access for the urban and architectural designer to get the aimed guidance from these tools in their decision-making process. The literature illustrated that a mathematical optimisation model could be categorized into three groups based on previous knowledge of the design problem at hand.

This chapter has shown the current gap in finding a way to fulfil the research aim of utilising urban geometry in optimizing urban performance. The literature has revealed the need to classify urban models and classify them into groups highlighting the capabilities provided by generative design to handle this task. Moreover, the literature can argue that classifying urban geometry and linking it to performance can provide a pathway to overcoming the time cost challenge when implementing available methods of artificial intelligence assisted optimisation methods. This will allow for the inclusion of the simulated results, where there

is a need to get better data-driven designs in the early design stages. The following chapter will discuss the thesis method in detail and build a framework that will implement the previously identified goals and fill the framed gaps.

3 Methodology

3.1 introduction

As shown in chapter two, the nature of linking geometry to performance at an urban scale remains unclear. It can be generally concluded that there is a need for a faster and more comprehensive way to conduct performance simulation on an urban scale. The research aims to fill this gap. The first stage of the research methodology was to conduct a preliminary study on iterative urban simulation to understand the problem's nature better and form clear aims for the research to help fill this gap. The results of this preliminary study have shown the limitations and opportunities of creating a framework that guides the urban design process in its early stage based on environmental performance. Limitations like computational cost and time directed the research to aim for a proof of concept of this framework that focuses on the solar radiation performance of the urban configuration. This aspect is less time-consuming than other environmental performance simulation aspects on the urban modelling scale.

The critical lesson learned from the preliminary studies was the significance of geometrical features relative importance on performance which led the research to investigate a method to classify urban geometry and utilise this classification towards more efficient environmentally driven urban early stage designs. This method of classification aimed at sorting each building based on its geometrical features. This goal was achieved by creating a generative modelling code that works on classifying urban models by attaching a text tag as an attribute for each building in the generated urban configuration. The automation of this classification was also part of the framework's development scope to provide flexible and dynamic modelling capabilities within the current limitations. This was done by developing the iteration controlling code during different phases of the research.

This classification created a set of data consisting of the building's classification tags. There was a need to develop a method to retrieve, save, and compare this dataset and investigate this geometrical classification's available opportunities and outcomes. This approach opened the door to utilise this dataset in applying machine learning principles to predict the solar radiation of the newly generated urban configuration based on the similarities of its geometrical features when attributed to its solar radiation performance. Using machine learning, ANN helped overcome the time limitation of running solar radiation simulation on

an urban scale and develop the framework towards an optimisation capability to guide the urban design process in its early stage.

Creating an optimisation framework for neighbourhood design was finally met by adding the genetic algorithm add-on, which provided the needed optimal solution for neighbourhood geometry based on its solar radiation and Floor Area Ratio. This phase was tested for different iterations' pool sizes and tested for an existing neighbourhood boundary with the same weather file.

The chapter is discussing the methodological reasoning of these different stages of the framework. The tools of each stage and its different metrics are discussed within this chapter sections with a summary of the final proof of concept version of the framework.

3.2 Preliminary study

The preliminary investigation aimed to present a sensitivity analysis of multiple variables of urban geometry to test their relative importance in different aspects of performance on an urban scale. The targeted geometrical features were height, built area ratio, orientation and window-to-wall ratio (WWR). These variables were selected based on the literature review results to cover both sides of the categorisation mentioned in the literature and allow testing of the indoor performance and the outdoor performance results caused by changing the geometry inputs. This was also due to a comprehensive vision of the relative importance of these variables and the broader scope of the framework in future work.

3.2.1 Simulation metrics and parameters

The performance aspects tested were solar radiation, energy demand for heating and cooling and indoor daylighting availability. The aspects were selected to cover both indoor and outdoor aspects; the daylighting availability was added to balance the energy demand results and show the correlation between the two aspects resulting from changing the geometrical variables. The metrics used to measure these performance aspects were based on the literature as solar radiation is measured by kilowatt-hours per square metre (kWh/sqm) which combines the amount of energy/heat received from the sun on the outer surface of the building. The same metric also measures the energy demand for cooling and heating by combining it with the indoor floor space (ANSI/ASHRAE/IESNA 2010). Daylighting autonomy

is measured by the percentage of the indoor floor space reached for a preset illuminance for a certain period (Illuminating Engineering Society (IES) 2017).

The preliminary study had two phases, and through each phase, the simulation process was enhanced to be adapted to newly found limitations or to validate results.

The first phase had the largest number of iterations but only analysed solar radiation and energy demand for cooling as it was conducted in a hot arid zone. The final phase of the preliminary studies was divided into two stages. The first stage added daylighting availability as a balancing feature of the simulation to ensure that optimal solutions for energy demand do not exclude the daylighting availability in the indoor environment. This addition was prompted by the fact that lower values of WWR showed better energy demand performance. Thus, an assessment of daylighting availability was needed to balance the performance results. Also, some of the literature have advised duplicating the simulations for multiple performance aspects to get more balanced results (Sabry et al., 2014)**.** The preliminary study's last stage was to verify results against different climate conditions for heating energy demand. This step provided a better understanding of the preliminary study findings and results.

3.2.2 Tools

Recent studies have provided important information on the implementation of parametric principles in urban modelling and simulation. The generative models used in these studies showed the capabilities of iterating models within the same simulation settings. It is established that generative models can overcome conventional simulation methods in a tested number of models. The second advantage of using generative models to conduct urban simulation is to utilise the generative capabilities to set up the pool of tested iterations and automatically gather the data. However, there are certain drawbacks associated with using generative models. They do not always run as smoothly as they should due to the implemented tools being relatively new when this study was conducted.

The research has used the Grasshopper (Davidson, 2017) tool for visual programming for generative modelling. It enables dealing with urban geometry on a generative basis. It is also an add-on for a conventional modelling platform called Rhinoceros (McNeel 2014). This compound of tools formed the base that hosted all the modelling done in this research. The

preliminary study in Grasshopper used a local algorithm for the generative models. The following phases used a component called Colibri, which is part of a set of components underpinned in the TT TOOLBOX plug-in (CORE studio 2017a). This tool continued for the following phases of the thesis until it was replaced by a more custom code programmed using PYTHON programing language (Python Software Foundation, 2001).

The research used a suit interface hosted in Grasshopper called Ladybug Tools (Sadeghipour and Pak 2013) for simulation. It acts as a parametric interface for widely used simulation engines, allowing Grasshopper users to conduct their simulation in the platform without transferring the model and collecting the simulation results from another software application. Ladybug allowed access for engines like Energy Plus (US Department of Energy, 2016) to simulate energy performance in the Grasshopper environment. Another link is provided for Daysim (Reinhart 2017) and Radiance (Ward 1994) which are simulation engines for daylighting performance. It is important to note that Ladybug tools are not the only suit that provides such an interoperability option. Nevertheless, it is one of the rare suits with an open-source licence and connects multiple engines, other than those mentioned earlier, in the same core code.

To collect and save the simulation results, the research used another plug-in to transfer data from Grasshopper to Microsoft Excel (Microsoft Corporation 2010), a data analysis and visualisation software. The plug-in is called Bumblebee (Mans 2016).

The preliminary study flow, conditions, limitations and results will be discussed thoroughly in the following chapter. Concerning the methodology, the preliminary study has clarified some challenges and opportunities that led to the formation of the current research aim. It provided several lines of evidence to allow for a practical application for sequencing the simulation performance instead of running a holistic simulation process at once. Several published studies have encouraged such an approach to conducting multi-objective simulation and optimisation, especially for cases as complex as urban neighbourhood models (Krippendorff, 2005; Lawson, 2006; Maver, 1970; Tschetwertak et al., 2017). Another trend of the results from the preliminary study was the clear difference in the relative importance of geometrical variables and their impact on performance especially. This showed the benefits of breaking down urban complexity into classifying its models into its core geometrical variables and classifying its performance based on that classification technique. A number of authors have

considered the effects of classification on simplifying the simulation process and reducing its time cost (Bassett et al., 2012; Robinson, 2006; Robinson et al., 2007; Vartholomaios, 2017).

These two principles shaped the current aim to meet the need for a fast and comprehensive urban simulation and optimisation framework by taking advantage of the relative importance of the urban geometrical variables to apply classification of neighbourhood models the geometrical nature of buildings included. Then, the framework will adopt these classified databases of buildings and their performance into a generative optimisation process that saves more time than the conventional brute force simulation and optimisation methods.

The first stage of building this classifying framework was to build its generative logic that can adapt to different inputs in a parametric way that does not lead to anomalies or system crashes.

3.3 Parametric and generative urban modelling

As mentioned earlier, Grasshopper/Rhino suits formed the platform that hosted all the parametric modelling processes in this study. The initiation of the street networks and buildable areas in this framework was conducted using a tool for urban network generation called Decoding Spaces (Koenig et al., 2017).

3.3.1 Generative modelling classes

After generating the base of buildable areas and street networks based on an input polygon boundary shape, the framework started a series of generative classes developed locally in Grasshopper to introduce variations in the urban morphology and sort each building on its contextual features. These modelling code classes control different geometrical features on urban and architectural levels of the neighbourhood model. Some code classes are intended to sort and classify buildings based on other features, and the third group to both sort and classify the buildings based on the modelling changes that occur in them.

For example, the coding class for creating urban voids is a class for modelling that does not directly impact the classification tag for buildings. At the same time, another class sorts each building based on its exposure to these generated urban voids. On the other hand, the class that generates building courts is doing both as it sorts the buildings in the model based on their status of having courts or not.

The product of this series is a 3D model that is a ready simulation while it has an implemented data classification for each of the included buildings. This classification tags the buildings to their simulated performance results.

3.3.2 Generative modelling control

The control of the generative process is achieved by varying the different input parameters. Colibri handled the initial stage of this controlling system, the plug-in to handle multiple cross-matching between a set of parameters (CORE studio 2017b), to create the expected pool of iterations. This was changed later to a PYTHON code (Python Software Foundation. 2001) embedded in a Grasshopper platform by another plug-in that allows calling for Python libraries and functions to be used live in the Grasshopper platform without compiling it outside (giulio 2017). This capability comes as a default in the recent release of Rhino 6 and its Grasshopper add-on. Utilising Python took into consideration the time cost made by Colibri. Thus, the combinatorial calculations were done only once in this version. The variation of the iteration took place by changing only one slider, which saved some time in calculating the whole process for each iteration generation.

This process worked when the control of the generation was limited to the iteration of the neighbourhood configuration scale. However, since the classification aim was to sort and analyse buildings and deal with the rest of the neighbourhood as its context, there was a challenge iterating both building and urban levels. As a modelling and Visual programming tool, Grasshopper cannot create two sliders corresponding to each other with different starting and ending values. This meant all configurations had to have the same number of buildings which would have limited the diversity of the tested, analysed and optimised neighbourhood. The study tried to utilise multiple open-source codes to gain control over Grasshopper's slider nature. However, the process had to be automated due to the large amount of variation handled in the case study, which required the inherited codes to keep refreshing on the right spot to change the configuration. The last version of the generation control class included remapping the urban configuration trigger values to fit into an almost equal number of buildings for each configuration. The only casualty of this system is that sometimes some buildings were simulated more than once. This did not contribute to the accuracy of the process as the neighbourhoods' solar radiation performance was simulated as a whole. This iteration technique used only to create the database of classification tags and

their performance to be recalled as a benchmark later in the prediction and optimisation stages.

Another limitation for Grasshopper is that its slider's animation option, responsible for automating the process, had a limit for animation, which meant that automating the generation and classification process was bounded by the number of buildings for each simulation and database building stage. However, this helped, in a way, to limit the causality of overrunning individual buildings because it means a new value is estimated for every 10,000 buildings. This solution resulted in groups of 25 to 40 configurations for each run, mainly relative to the number of buildings that limited the repeated cases.

The Decoding Space limitation inherited another challenge as it did not maintain the same configuration for the same parameter inputs. This means that every time the trigger moves to analyse a different building, it might generate a different configuration. Therefore, the control for generating the configuration had to be frozen with no triggering caused by the change due to variations between different individual buildings for analysis and classification. This solution was done with the help of another plug-in for Grasshopper, called Heteropetra (Bahrami 2018). It is used to pause the signal triggering the change of urban geometrical configuration until the other signal controlling the individual buildings by its index finishes running through it.

This stage of the framework will be discussed in chapter four with detailed figures and examples on creating the generative model and its controls.

3.4 Geometrical classification and data retrieval

This section will focus on the data handling of urban geometry, the first phase of geometry classification. The second phase is the database building, the analogue lookup, and the different development stages it passed through to reach its final current state.

3.4.1 Geometrical classification

classifying the geometry in this research was achieved by a classification label for each building included in the neighbourhood geometrical iterations. This label acts as a lookup tag for similar buildings with the same geometrical features. In this way, performance can be approximated based on the similarity of geometrical features, and urban performance can be

calculated based on this approximation. This method forms the basis for the latter stage of machine learning optimisation discussed in the next chapter. The application of classifying is proved to help with the time cost of handling large amounts of data. Many studies in the literature have depended on machine learning to overcome the time cost challenge in conventional simulation methods that require a long time to search for an optimal solution.

The classification process takes place in multiple classes in a parallel sequence to the generation classes. The basic idea of this classification is to take advantage of the nature of

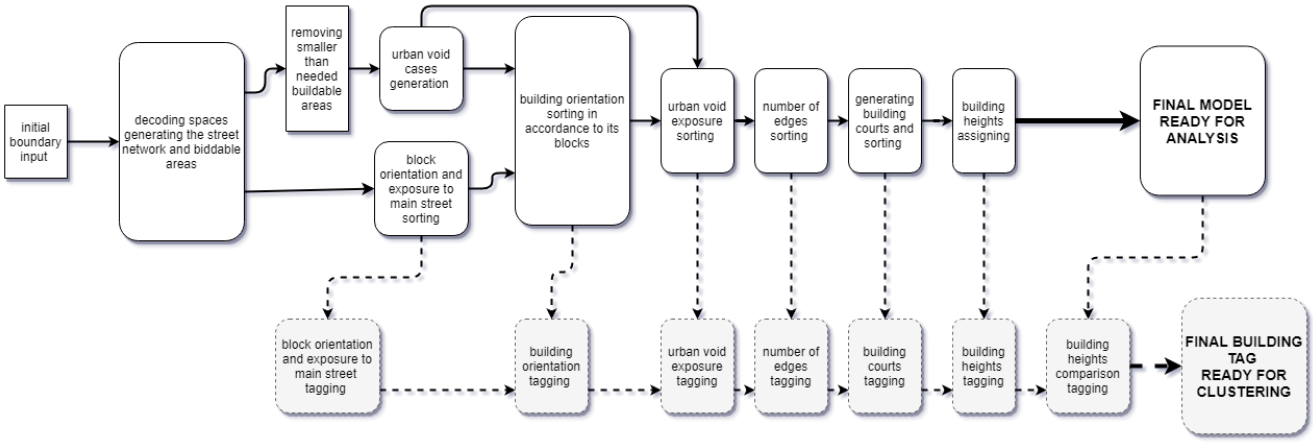


Figure 3-1 shows the parallel sequence of the generation model and its classes, upper flow of components, and the classification tag , and lower set of component sequence

Grasshopper being a visual programming language to deal with the neighbourhood model as a list of data. Each index in this list represents a building, and the index gets tagged and classified based on a function or an analysis of the building's geometry.

The classification tag had two phases of development. The first phase product was a mix between numbers (integers) and text data types. This phase included the following features of the building geometry:

- Urban block orientation
- Urban block exposure to the main street
- Building orientation within the block
- Building urban void exposure
- Number of edges of the building
- Building area

- Building court condition
- Number of surrounding buildings and height comparison. For example, three buildings higher, and none are equal, and one building is of lower height than the tagged building.

These tags formed a database saved in an excel sheet format using the plug-in for communicating Grasshopper to the Microsoft excel data analysing program Bumblebee. The lookup used in this phase was conducted using text similarity components locally available in Grasshopper. This phase of the tag was tested in multiple stages. The testing aimed to investigate the accuracy of the approximation against actual results collected from simulation. Moreover, it aimed to assess the difference in time to get the prediction against the time consumed to conduct the simulation. These multiple tests are detailed and discussed in chapter four. The results of these tests led to the development of phase two of creating the classification tag and developing a different bespoke way for the lookup.

The second phase of the classification tag had to be fully numerical to enhance the lookup results. In addition to that, there are some added features to the tag to enhance its resemblance test and therefore enhance the lookup's accuracy. Besides the features mentioned in the previous phase, the following features are included in the second and final phase of the tag:

- Building height
- Orientation of the surrounding buildings to the tagged building and its height comparison to it. For example, a building higher to the east and another one equal to the south-east and another higher one to the south ..etc.
- Wider perimeter of surrounding buildings for comparison. This is to check for higher buildings in a further perimeter to the first tested one. This feature locates the highest building on this broader perimeter. It indicates its relation and orientation to the tagged building—for example, a higher building with 30 metres in the northeast.

3.4.2 Data retrieval

A set of 400,000 tags were generated from this tag to start the final testing and investigate the lookup learning curve. The lookup for this tag was a bespoke method made to detect each

of these features algorithmically to make sure it does not depend on any of the ready-made components in Grasshopper. This database was brought back from excel sheets to Grasshopper utilising the components for reading excel sheet data in Bumblebee.

Such a large scale of data needed to be cleaned from data anomalies caused by the limitations of the used tools generating odd or unpleasant cases of some recorded features. This was done by a series of masks testing the nature of each data entry and removing all non-numerical entries from the different lists. For example, a building might get an unacceptable analysis in the test for the second perimeter surrounding the building due to its location or a geometrical generation anomaly inherited from Decoding Spaces. This building entry must be deleted from this list, and the other 11 recorded features' lists. A series of selection masks have conducted this cleaning to the lists that showed it contained non-numerical data.

The lookup in this stage was also made using the same technique. Each feature list was isolated for each building in the configuration at hand, then the lookup match for all the similar options for each one in the database. Then, only the indices with approved values continue to be tested in the second phase. At the last stage of the lookup, it finds the recorded performance of the surviving similar buildings from the database.

Subsequently, a building with unique tags needed to be predicted using a machine-learning application and the sum of the two groups will be fed a fitness for a multi-objective genetic algorithm to find the optimal solution for the tested pool of iterations, which is the third and last stage of the framework

3.5 Empirical study

The optimisation and machine learning prediction are the final stages of this framework. In addition, this section will discuss the simulation used to build up the database used in the prediction. The whole framework optimises the neighbourhood geometry to its optimal solar radiation as a starting point for future work, including other performance aspects like energy demand and daylighting availability. A code developed by Ladybug Tools did the solar radiation simulation, and it is part of its components' library.

The machine learning application chosen to be implemented is Artificial Neural Network. Initially, a neural network component in the Lunch Box plug-in (Proving Ground Inc, 2018) was used to predict the tags' performance. This plug-in aims to implement machine learning

applications for the users of Grasshopper. Due to some limitations, this tool could not provide an efficient result, either in time or accuracy.

The following step was to use an open-source neural network code adapted to Grasshopper by the researcher with some adjustments to enhance its performance to the needed prediction. The adaptation process was undertaken with the help of my second supervisor. The code went through multiple stages of testing and enhancement to get it to its latest version. These tests varied in the scale of the database fed to the neural network and the neural network structure itself and its settings. Also, these tests included data that were never seen by the neural network to assess its capability for prediction towards new data entries.

This led to a series of edits of the nature of the code to search for a neural network that can save its training, deal with multiple numbers of entries for predictions, and save time by cross-validating the prediction results within the training time, which led to high performing neural network.

The consecutive stage of this was to use this performance prediction as a fitness function for a genetic algorithm to find an optimal solution for the tested neighbourhood models' solar radiation and floor area ratios. The research used Biomorpher (Harding 2017) to conduct the optimisation part of the framework. Biomorpher is a cluster-oriented genetic algorithm with an interface that allows control over different finesses individually and gives a visual result for each generation, allowing more control on the progress of the process and the ending point. This allowed more control over the optimisation process to decide in each stage whether to continue or stop based on the time cost and expected accuracy decided by the user.

The validation of the outcomes will be discussed in chapter seven to show the different stages of validation to test.

3.6 Framework testing

As shown in each stage, there are different tests and enhancement phases. The results are also validated on different scales and stages. The first stage of validation will test the accuracy of its prediction and optimisation compared to pre-saved results. Moreover, to test the time saved by the framework compared to running the whole iterations based on simulation rather than NN prediction.

This validation phase uses different scales of databases and testing pools of iteration to understand the capability of the prediction with preloaded and newly introduced data.

The following stage of validation has considered the classification tag's capability to understand different neighbourhood boundary options. In addition, it also considers the NN capability of maintaining its prediction accuracy rate with a different set of geometry parameters that changes the whole pool of iterations and creates a different set of neighbourhood models.

The last phase of the validation tests the efficiency of this framework against an existing neighbourhood geometry. This phase will test the generation's ability to create multiple iterations for an existing case and the classification of its buildings. Furthermore, this has provided an insight into the optimisation capacity based on an existing case study and the amount of efficiency achieved by optimising the solar radiation performance for an existing neighbourhood using this framework.

The validation results will be shown in chapter eight, followed by the thesis outcomes, discussions, and future work.

3.7 Overview of methods and framework structure

As discussed earlier, the scope of the research has been reshaped based on the preliminary study's findings to have a proof of concept of the framework capacity for one performance aspect, which is solar radiation. The framework, focusing on solar radiation, has gone through different stages to handle the neighbourhood model. As shown in Figure 3-2, the initial phase of the framework involves modelling and generation, including setting the urban model parameters and preferences, site boundary and weather file or location. This is followed by the simulation phase with the database build-up. This is formed by a portion of the iteration pool acting as the training data for the next phase, the Artificial Neural Network prediction phase. In this phase, the ANN will be trained and prepared to predict different model entries. The endpoint of the framework is where the genetic algorithm takes over the flow, controls the input parameters' variation, and assesses the value received from the ANN predictions.

Finally, validation against the pre-saved data is conducted to show the difference; this framework proved to have compared to conventional ways of simulation-based optimisation.

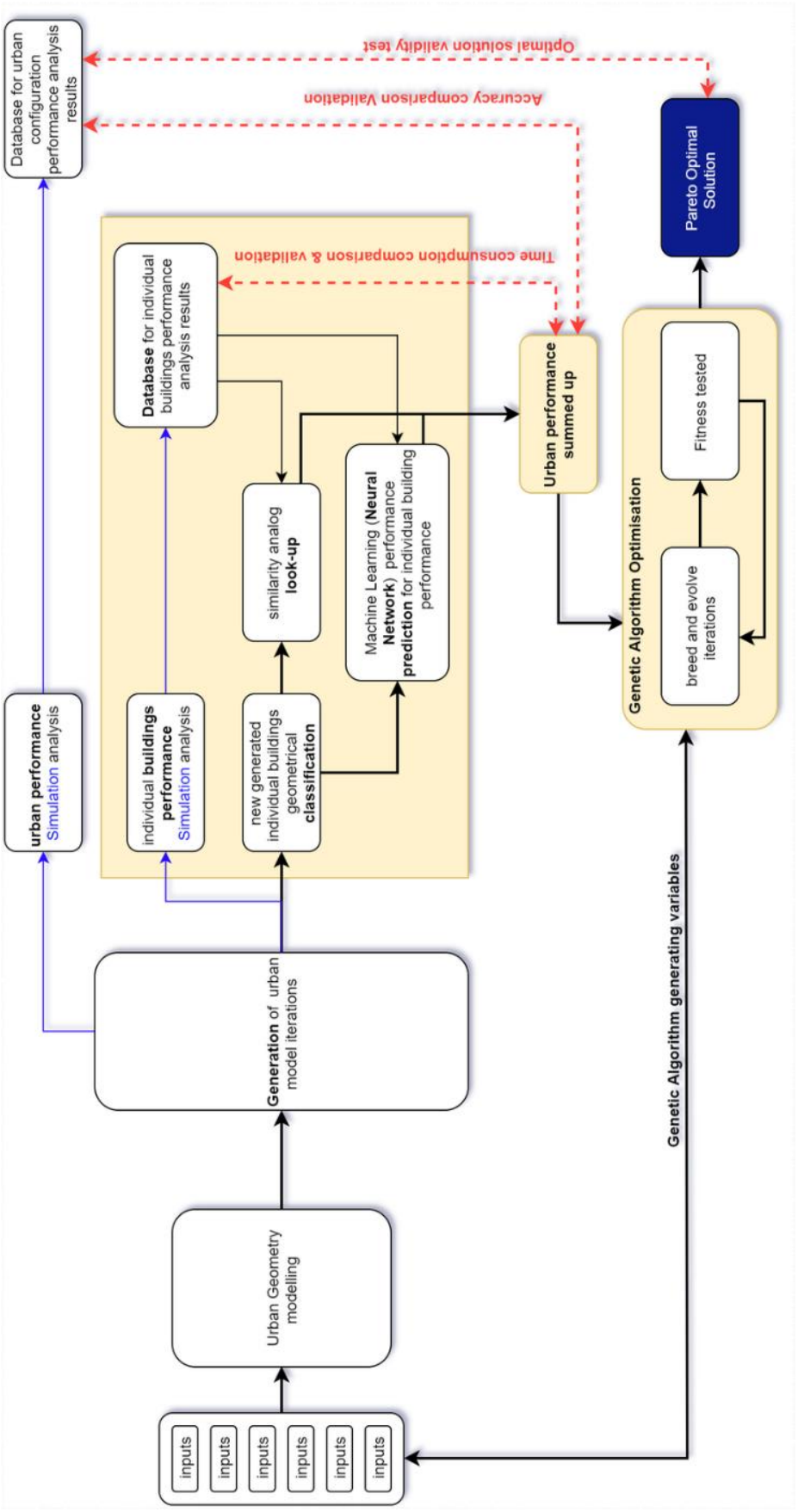


Figure 3-2 Diagram of the workflow scope and stages

3.8 Summary

As shown in this chapter, the research aims have been identified by the literature review and preliminary study of the urban geometrical relative impact on different aspects of performance. The research aims to utilise the geometrical impact on urban performance and solar radiation to build an efficient framework that can inform urban design in its early stages. This was done by applying geometrical classifying to an urban neighbourhood case study in hot arid zones. The method shows how the research fulfils this aim through the different stages of building this framework. In terms of geometrical challenges, the research has depended on parametric modelling tools to create the starting point of the framework to generate the multiple tested models. In addition, the parametric nature of the model allowed the researchers to accommodate the geometry classification to be more accepting of the different variations in the generative process. A classification technique was applied to simplify the urban geometry to its core elements providing a more straightforward way to overcome the complexity of urban modelling. In addition, it had a significant impact on performance prediction and optimisation, which has been based on coupling between the neural network and cluster-oriented genetic algorithm principles and solvers.

To summarise this proof of concept version of the framework, it can be divided into five main stages. The first stage is related to the generation of the parametric model and setting up the inputs for the desired neighbourhood design. This stage uses the tool Decoding Spaces to generate the geometrical variables. The following stage was to create an iteration control code responsible for iterating the different available options on the urban configuration and individual building levels. A developed python code (Python Software Foundation. 2001) is in charge of this iteration process. At the same time, it was building up the database with the help of some heteropetra's (Bahrami 2018) components to control the iteration of individual buildings. Another stage for building the database is to run the solar radiation simulations and collecting their results. The main tool used for this stage is the ladybug tool (Sadeghipour and Pak 2013) to simulate the direct solar radiations of the tested geometries. The stage of applying ANN for predicting solar radiation results utilised a python code developed for this research scope. The final stage of optimising the outputs and weighing between the generated iterations used cluster oriented genetic algorithms tool. This tool was also available

on the grasshopper platform set of add-ons and was named Biomorpher (Harding 2017). The set of used tools and their related stages is shown in Figure 3-3.

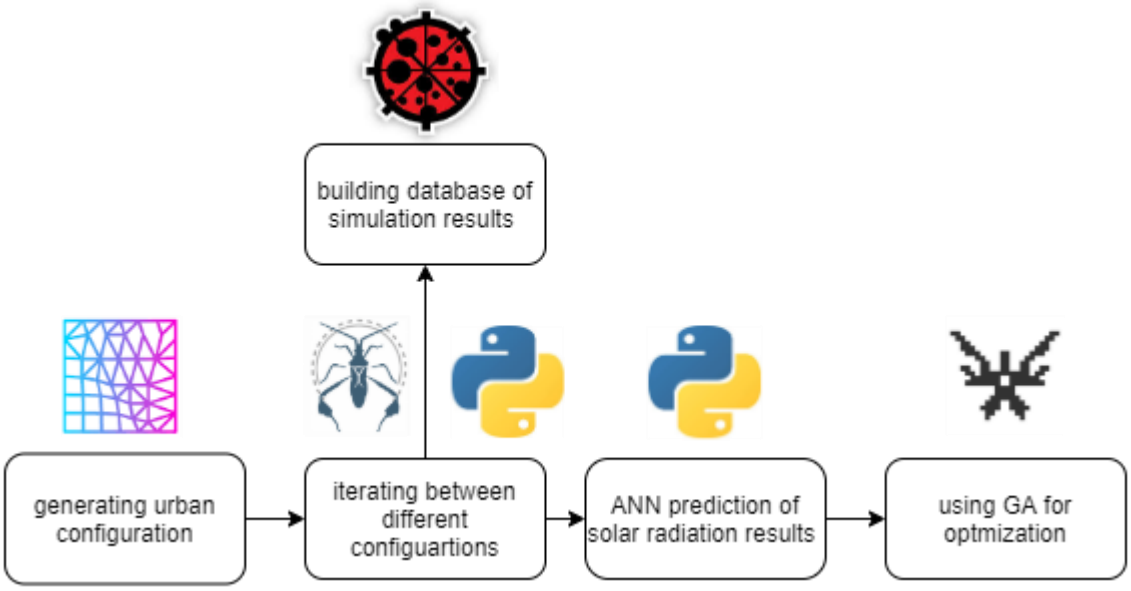


Figure 3-3 the final version of the framework stages and its used tools and coding languages.

The methodology of this research provides a novel method to classify urban geometry and utilise its impact to predict urban solar radiation for a wide range of varied urban models faster, more accurately and efficiently than conventional simulation-based optimisation. This outcome will allow the data-driven performance-based urban geometrical design to be included in the early design stages.

4 Preliminary studies

4.1 Introduction

This phase of the research sought to clearly understand the multiple variations of the study's key aspects. Different preliminary study stages provided a better understanding of the correlation between geometry and performance within an urban context. The early stage of the study looked at the performance aspects of energy demand and solar radiation, and daylighting availability was added as a balancing feature for the last phase. The preliminary study stages also differed in the number of tested iterations based on the challenges faced in the early stage. The preliminary study was a sensitivity analysis of the variation of geometrical features in a simple urban grid context to overview the simulation and testing results. Looking for optimal solutions was not the aim of this study. Instead, the main aim was to analyse the impact of geometrical variables on performance. Thus, the parametric approach of the tools was utilised to generate a brute force simulation with no selection from the planned variables.

The literature review has shown that modelling and simulation are under ongoing investigation to address urban complexity and link it to enhance decision-making in the design process, especially in the early design stage. The time cost is one of the most challenging obstacles for gaining performance simulation data to inform the design accurately. This challenge is emphasised with the scale of the urban models and the data handled with them regarding multi-objective simulation and optimisation. These challenges need to be further investigated by breaking down urban geometry into its variables and testing each factor's relative importance on the urban performance mentioned aspects. Urban performance simulation aspects also need to be analysed regarding their impacts on design decisions and the time consumption to be included in the early design phase, which is usually faced with the challenge of insufficient time. Using simulation on an urban scale to advise for early-stage design decisions is critical to timing and accuracy and the amount of analysis needed to inform the design decision.

Several studies investigate the relationship of the urban geometrical variable on energy demand (Hemsath & Bandhosseini, 2015; Hosney Lila & Lannon, 2017; Lin, 2014; Nault et al., 2015; Ratti et al., 2005; Robinson, 2006; Rodríguez-Álvarez, 2016; Vartholomaios, 2017; Yi & Kim, 2015). This sensitivity analysis on urban geometry was conducted with a holistic and integrative approach. Urban context has a significant influence on the building heat loss/gain that determines the energy demand needed to achieve indoor thermal comfort. The scope of

this study was to investigate the probabilities of conducting a holistic simulation in an urban context that includes different performance features for the same model, such as testing the same model for energy demand, solar radiation and lighting performance. It also aimed to investigate the possibilities of sequencing these performance aspects to reduce the time consumed in its simulation and analysis and the effect this sequencing might have on the accuracy of the results. Different studies have put forward and tested the design problem and analysis into a multistage decision-making process as a solution to tackle the complexity of urban design and optimisation (Maver 1970; Krippendorff 2005; Lawson 2006; Tschetwertak et al. 2017)**.**

The urban geometrical variables analysed were height, built area ratios, orientation and WWR. The trial included variables from different scales to analyse their effect on the building performance. The study included daylight availability sensitivity analysis by changing the lighting control systems. This study included three different climate conditions to refine and compare the energy demand for cooling and heating energy performance between different conditions. The study investigated the effect of changing lighting controls (standard ON/OFF controls vs dimmers) on cooling energy consumption. In addition, it provides an insight into the buildings' inter-shadowing effect by adding the context of the buildings' built area ratio in the tested grid. The tests investigated the relative importance of each geometrical variable and the correlation between the change of energy demands based on the change of lighting controls.

As mentioned in previous sections, this preliminary study was conducted using the modelling tool of Rhinoceros and its visual program language interface, Grasshopper. The Ladybug Tools Package did the simulation for solar radiation and energy demand, and lighting availability. Honeybee, one part of that simulation package, acts as a geometry mediator between Grasshopper as a modelling tool and simulation engines that conduct the targeted simulation, which are Energy Plus, Radiance and Daysim, in this case. For solar radiation, the primary member of the package used was Ladybug. The algorithm used for this sensitivity analysis will also be discussed in this section. Adopting this mix of tools and packages allowed for automating the process through the basic principles of parametric modelling. This provided a better insight into applying these tools on a large scale of data and helped develop ways to

overcome the challenges of time consumption and enhanced the efficient use of available computational facilities. This will be further discussed in this section.

The preliminary study can be divided into two phases:

- The initial phase was to analyse the set of parameters on energy demand and solar radiation within various contexts. This was conducted using the weather file of Aswan, Egypt, as representative of a hot arid zone climate. Although the results of this phase were influential in shaping the second phase, these test details and results are discussed in appendix A of this thesis to focus on the following systematic testing phase.
- The final phase was to verify the first phase results adding the daylight availability as a balancing feature which provided a better understanding of the WWR impact, especially in a hot arid zone climate. Moreover, the aim of repeating the process in different climatic conditions in London, United Kingdom (UK) and Birmingham, UK. This added another aspect of verification for the results for the heating demand, this time due to the similarity in results for the two selected climates; Birmingham results were moved to appendix A.

The three different climate zones and materials used in each phase are shown in Table 4-1

Table 4-1 Different cities and their related data in each phase

City	Coordinates	Class	Climate	Study Phase	Materials Used
Aswan, Egypt	24.0889° N, 32.8998° E	1B	Dry	1	Based on literature
London, UK	51.5074° N, 0.1278° W	4A	Marine	2	ASHRAE BASED

4.2 Preliminary studies algorithms

The algorithm used to run these simulations was a simple trial to test the available tools and methods to iterate the different geometry variations in simulations. Figure 4-1 shows a simple illustration of its stages. The arrows in red represent the first phase of the preliminary study where energy was the main aim of the simulation, while the blue arrows represent the added daylighting availability balancing the method.

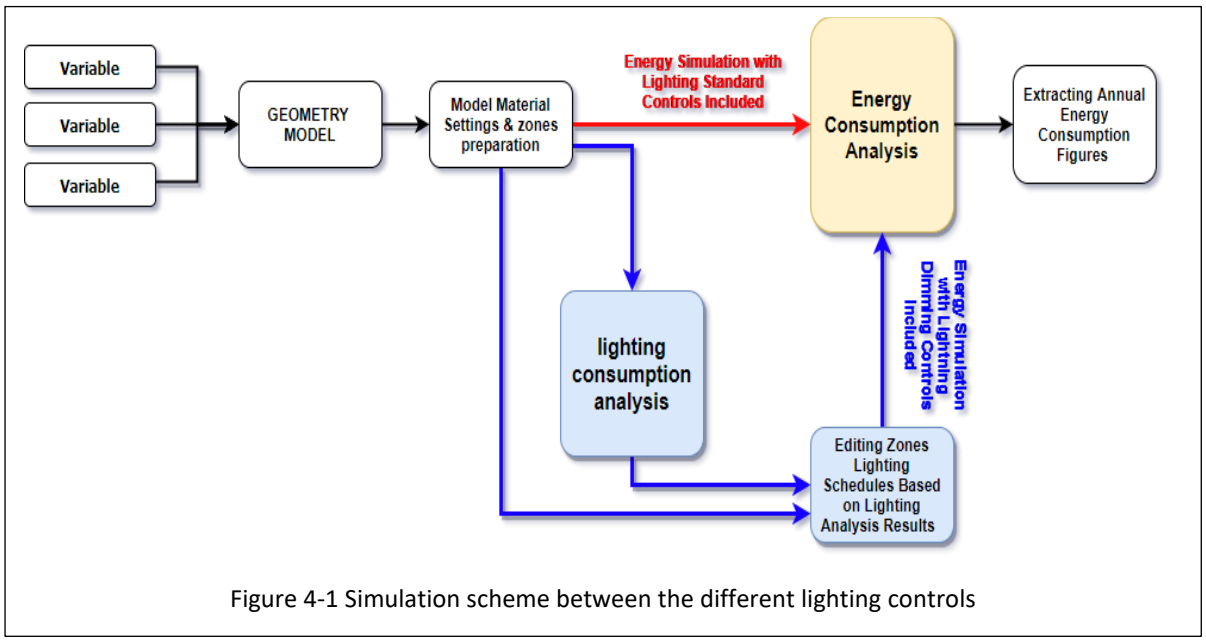


Figure 4-1 Simulation scheme between the different lighting controls

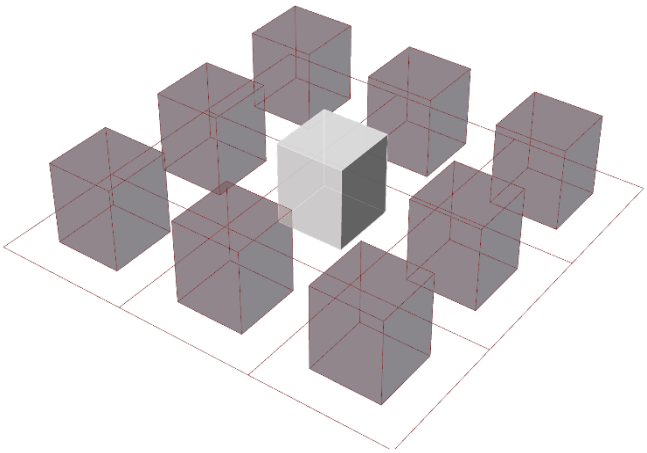


Figure 4-2 The grid and the selected middle building

The model creation is a mix between Grasshopper components and some components from Ladybug Tools. The model is based on a simple "three by three" grid while the simulation is conducted for the middle building while the geometrical variations occur on the nine included buildings in the same way.

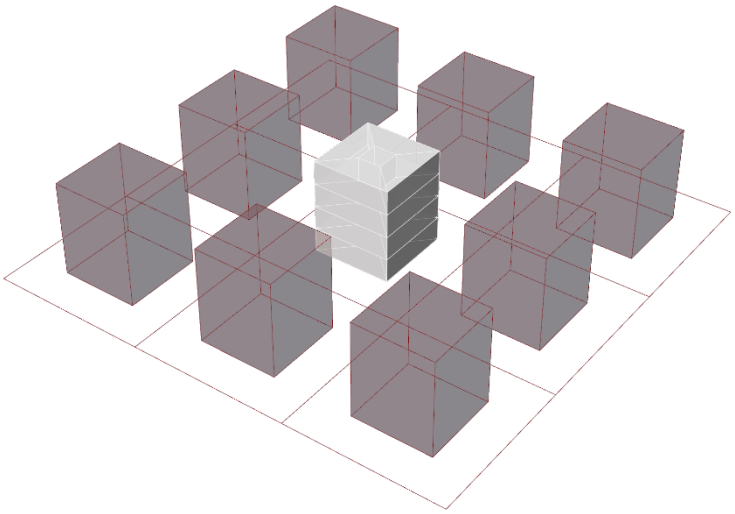


Figure 4-4 The middle building after creating zones

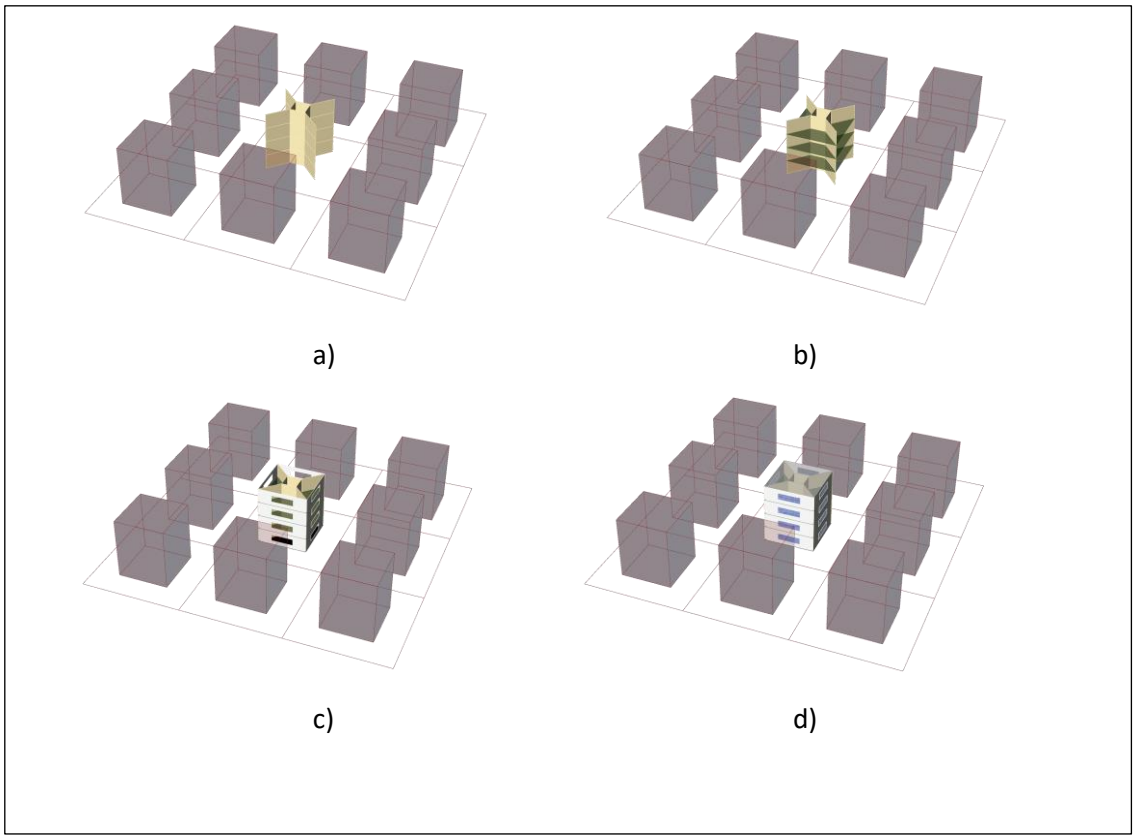


Figure 4-3 shows different elements of the model a) interior walls, b) interior floors added, c) exterior walls and ground added and d) where the windows and the roof are added as transparent only to show the inside of the model.

This is shown in Figure 4-2. Although Ladybug Tools package has a component that deals with creating the thermal zones with controls of the perimeter to core ratios, in this algorithm, these zones were modelled locally in Grasshopper to save the time consumed for each run. The Ladybug component did consume more time than the regular way of modelling in its early versions, especially when it came to automated generation, as shown in Figure 4-4

The following stage is to turn these closed zones into Honeybee zones, or, as it is named in the definition, HB zones. These zones are meshes that are readable to the simulation engines used in this simulation. Adding the WWR follows this stage as in this study, the distribution of glazing was equal on all four orientations. The Honeybee component is used for this function. Then the assignment of materials for different faces is also done by the Honeybee component. Then, the final stage of simulation is done either for energy in the first phase or for daylighting and energy in the second phase. The last stage of the algorithm is about getting data out to excel sheets to get the results visualised, as shown in Figure 4-6, Figure 4-7 and Figure 4-8. The different elements of the model are shown in Figure 4-3.

This algorithm allowed dynamic control of the model generation and automatic run for multiple iterations without adjusting the simulation settings. Controlling the automation of this process was mainly developed in local Grasshopper components in the earlier stages of the analysis. Near the end of the preliminary study, it was controlled through a component downloaded from the package of a plug-in called TT ToolBox the component name was Colibri. Later there are some open-resource codes available online for the same function. This process will be discussed in more detail in chapter six. Appendix B is showing the grasshopper components and code used for this study.

4.3 Preliminary study final phase

The initial phase studied Aswan, Egypt, to test the variation effect on daylighting and energy performance. This was followed by enhanced simulation with more generic materials from the ASHRAE standard for each climate condition to ensure that the study results were not affected by any elements other than the geometry.

4.3.1 Aswan simulation with literature-based materials

4.3.1.1 *Location*

This study phase used the Egyptian Typical Meteorological Year (ETMY) weather file for Aswan city in southern Egypt (24.0889° N, 32.8998° E). Aswan targets future urban growth in Egypt (Egyptian Ministry of State for Administrative Development 2016). It also represents an important sustainable development node for Egypt in hosting the high dam of the Nile as one of the old national development projects. It has a hot, dry climate. There are governmental plans for its growth with a twin new city. According to Kottek et al. (2006), Aswan falls in the hot arid zone classification.

4.3.1.2 *Geometrical variables*

Learning from the initial testing phase, the number of variables for this test was reduced than the initial one. Table 4-2 shows the number of iterations for each variable. The reduction took place mainly with some orientation angles tested and the WWR. In this study, the iterations are less, but the spectrum tested is the same to avoid affecting the relative importance for each variable. For orientation, it had two iterations for either 0 degrees with a building facing north and 45 degrees rotation of the whole configuration. WWR had three iterations in this case, including the two bounding values of 20% and 80%, along with a 50% option to have a marker in the middle of the change (see Figure 4-5)

This algorithm did a cross-matching for these geometrical variables producing 210 different iterations. The study can be divided into six groups with 35 iterations each. These groups included two sets of orientations of 0 and 45 degrees and three sets of WWR .2, .5 and .8 with the total original variations of heights and building scales.

Table 4-2 Geometrical parameters for the second phase of the sensitivity analysis

Geometrical Variables							
Height (metres)	3.5	7	10.5	14	17.5	21	24.5
Scale (built area ratios)	50%	60%	70%	80%	90%		
Window-to-wall ratio	20%	50%	80%				
Orientation (degrees)	0	45					

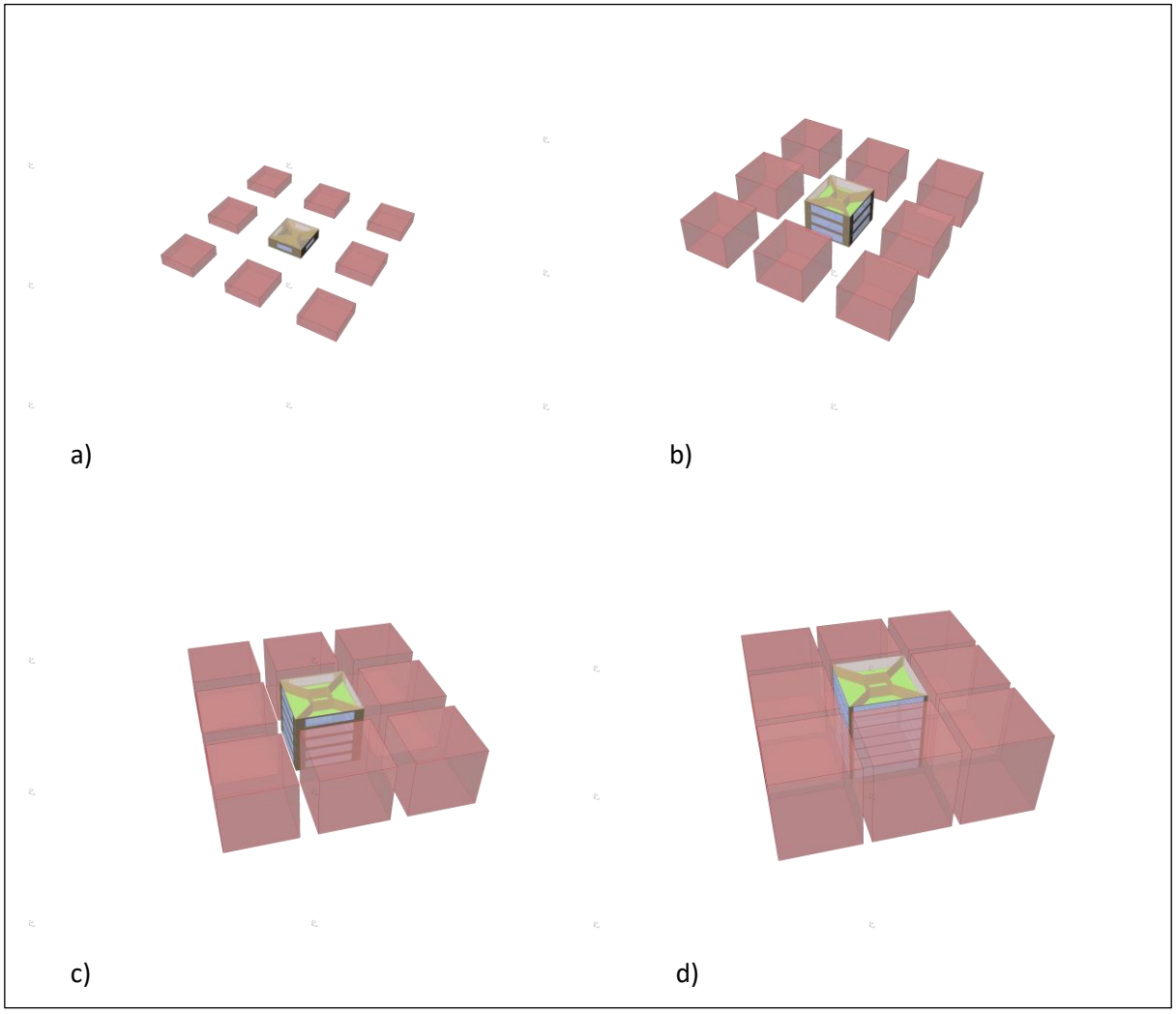


Figure 4-5 Geometrical variables examples a) one-floor height, 20% WWR, 50% built area ratio and zero degrees rotation. b) Three-floor height, 50% WWR, 70% built area ratio and zero degrees rotation. c) Five-floor height, 50% WWR, 80% built area ratio and 45 degrees

4.3.1.3 *Material input*

The inputs for material were adjusted based upon some studies made in the same geographical context (El-deep et al. 2012; Attia and Evrard 2013). The material properties were fixed for all the iterations and designed based upon the specification of the Chartered

Institution of Building Services Engineers (CIBSE) Guide for environmental design (Butcher 2006). Table 4-3 shows the material parameters used in this test.

Table 4-3 The material parameters used in the study

CUSTOMISED MATERIALS			
External Wall		Internal Wall	
U-Value	3.10	U-Value	5.29
Materials	CEMENT PLASTER BRICK (EXPOSED) CEMENT PLASTER	Materials	CEMENT PLASTER BRICK INTERIOR (EXPOSED) CEMENT PLASTER
Internal Floor		External Roof	
U-Value	1.43	U-Value	0.36
Materials	CERAMIC-FLOOR-TILES CEMENT-MORTAR(MOIST) CONCRETE CAST(HEAVYWEIGHT) GYPSUM-PLASTER	Materials	CEMENT-MORTAR(MOIST) EXPANDED POLYSTYRENE (EPS) CONCRETE, CAST (HEAVYWEIGHT)
Single Glazed Window			
U-Value	5.4		
Materials	CLEAR GLASS 12MM		

4.3.1.4 *Daylighting availability analysis settings*

The main goal of adding daylight availability to the study was to balance the energy consumed for thermal comfort and lighting the zones. Some of the dense configurations caused little sun penetration, which was beneficial to the cooling energy consumption, but, on the other hand, there was a need to know the effect this might have on lighting consumption.

For the daylight, the annual analysis of each zone was divided into a mesh of a 0.6-metre cell with one sensor point in the centre of it with a 0.7-metre height from the floor. The lighting control system used is auto-dimming with a switch-off occupancy sensor with 300 lux target illuminances for each zone.

As shown in Figure 4-1, the addition of the daylighting availability test took place before simulating energy demand. This was done because the idea of balancing was to test the impact of changing the lighting controls between two systems. The first system is one with

standard ON/OFF controls, while the other system is a dimming light control system based on achieving the desired 300 lux illuminance in the space. The dimming control system simulation is used to edit the lighting schedule before it feeds to the energy simulation. In this way, the energy demand will show the difference in its patterns due to this change of control systems. The test has been conducted twice to compare these results, utilising one control system in each run.

4.3.1.5 *Results showcase*

The results for each case were provided separately for each zone. This is shown in the examples illustrated in Figure 4-6, Figure 4-7 and Figure 4-8. The group results are summed up for each case. The example shown for illustration is an intermediate case with 45 degrees' rotation, 20% WWR, four-floor height and 70% built area ratio. Heating consumption varied

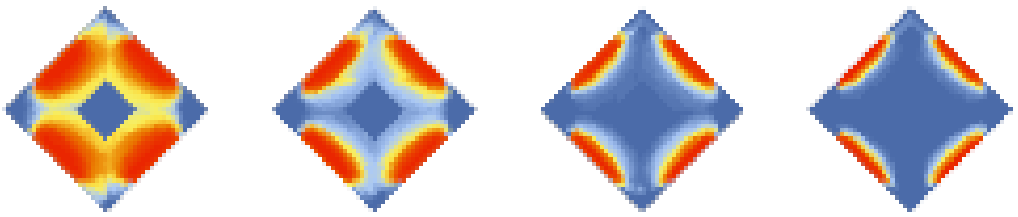


Figure 4-6 Daylight autonomy results illustration for 45 degrees rotation, 20% WWR, 4-floor height and 70% built area ratio showing the results of different zones. The floors order begins with the ground floor to the right up to the 5th floor to the left

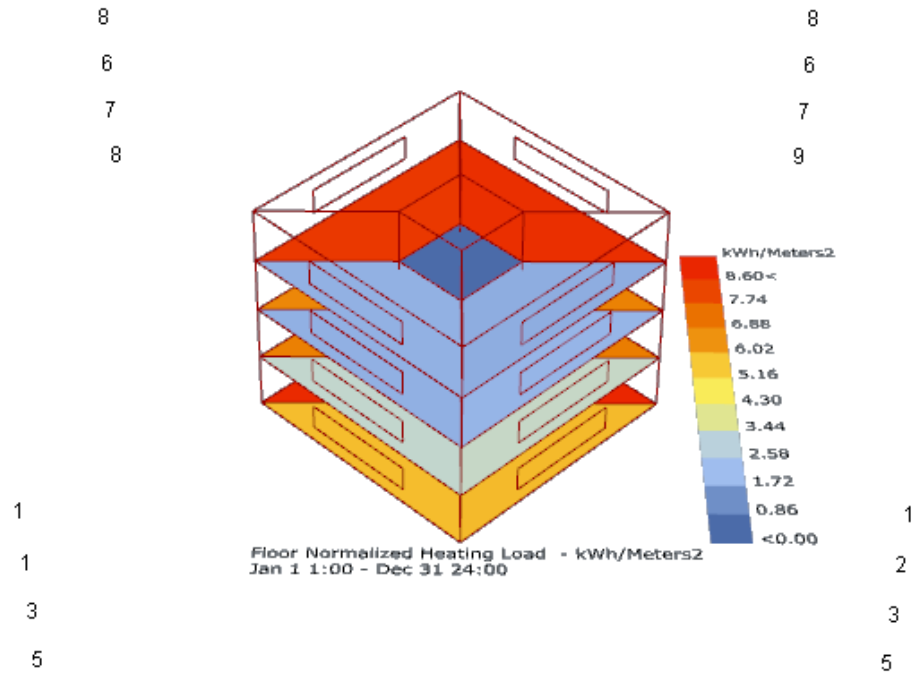


Figure 4-7 Heating consumption in kWh/m2 for each zone for the mentioned case

between 1.7–8.5 kWh/m2 for the whole run, so it lacked significance to be added to the current study results as the cooling results have much more significant variance.

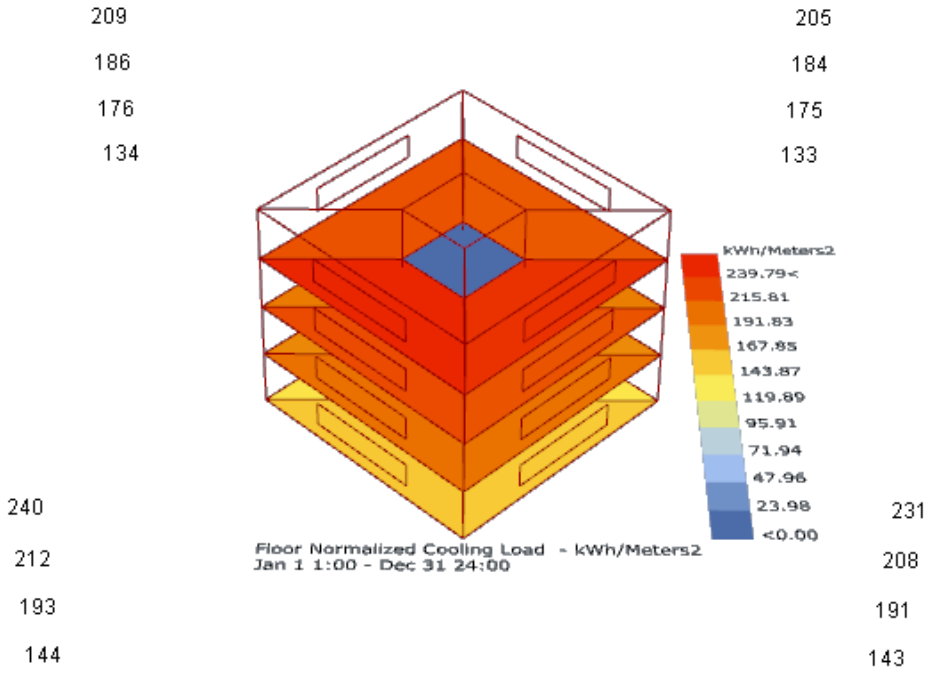


Figure 4-8 Heating consumption in kWh/m2 for each zone for the mentioned case

4.3.1.6 *Summary*

As shown in appendix A, these test results showed that the pattern in cooling consumption is repetitive through different variables. At the same time, the significance is mainly driven by the variation of built-up area ratios and heights. While the values are changing due to the change of WWR, the pattern of consumption is relatively close between the different six groups sorted by WWR and rotation. The relative importance of this geometry will be discussed in further detail in the following section and the ASHRAE material simulation results for Aswan.

There is also a significant finding of differentiating lighting control systems on cooling consumption for the six different groups. As shown in appendix A, there is a continuous linear correlation for the two lighting control systems on the results of cooling consumption. These linear correlations are a further reason for rerunning the simulation with a set of materials

based on the ASHRAE benchmark recommendation for this climate zone and other climate zones with different cooling and heating requirements.

4.3.2 Different climate zones analysis with ASHRAE-based material

4.3.2.1 *Simulation settings*

Following the previous testing phases and results, The research aimed to analyse and compare these findings and correlation with a different systematic approach. This was done by conducting the same brute force sensitivity analysis with the same settings yet assigning ASHRAE materials and using different climate conditions.

Table 4-4 Assigned material properties

Aswan, Egypt		**London, UK**	
Material name	**U-value**	**Material name**	**U-value**
ASHRAE 90.1-2010 Extwall Mass Climate Zone 1	3.69	ASHRAE 90.1-2010 Extwall Mass Climate Zone 4	**0.64**
Interior Wall	2.58	Interior Wall	2.58
Interior Floor	1.44	Interior Floor	1.44
ASHRAE 90.1-2010 Extroof lead Climate Zone 1	5.84	ASHRAE 90.1-2010 Extroof lead Climate Zone 2-8	0.28
ASHRAE 90.1-2010 Extwindow Nonmetal Climate Zone 1	5.84	ASHRAE 90.1-2010 Extwindow Nonmetal Climate Zone 4	2.27

The study inspected the potentialities of the sequential design process to be applied in environmental optimisation, especially with the challenge of urban complexity and time-consuming environmental analysis. The study used the climatic data of the cities of Aswan in Egypt and London and Birmingham in the UK. However, due to similarity in findings, the results of Birmingham, UK is shown in appendix A. The three cities have different climate condition classifications, as shown in Table 4-4(ANSI/ASHRAE/IESNA 2010). Exterior materials were set to ASHRAE recommendations for each climatic condition (Table 4-4)

(ANSI/ASHRAE/IESNA 2010). Moreover, interior materials were normalised to the default interior wall constructions from the Ladybug tools set of materials.

The study of Aswan, Egypt, was concerned with cooling energy demand patterns and the results. The results for heating energy demand in London will focus on the following section, along with cooling results from the Aswan analysis when needed.

4.3.2.2 *Results*

4.3.2.2.1 Variables relative importance for Aswan (cooling) two sets of material

4.3.2.2.1.1 Heights:

In the conducted study in Aswan, there were seven height variations. The results imply that there was a noticeable difference in the cooling energy consumption within the building. The study shows that the relationship between height and cooling energy consumption negatively correlates in both phases. It could be said that this is due to the arid conditions of the specified zone. A comparison of the calculations of the extremities (highest and lowest blocks) showed

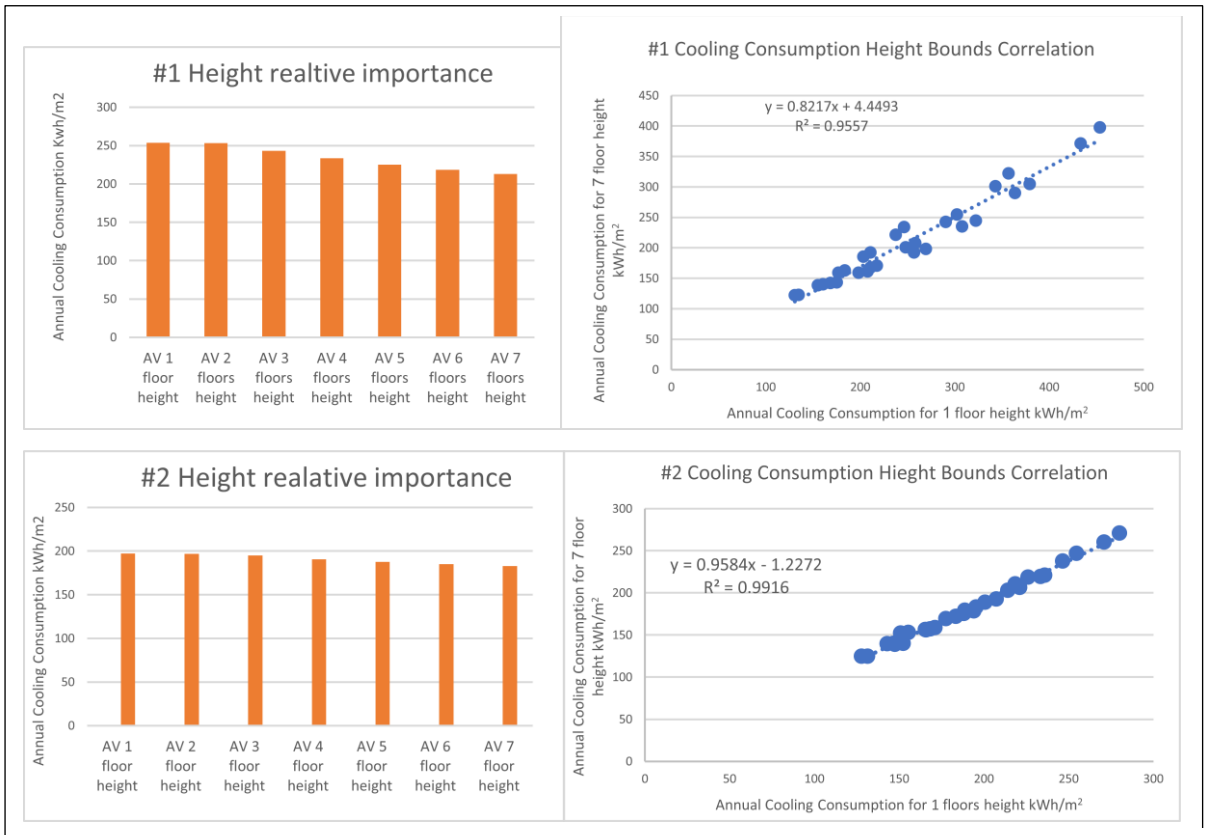


Figure 4-9 (#1) To the left: height variations' cooling average consumption comparison in kWh/m^2, to the right: height variations' bound correlated in kWh/m^2, (#2) To the left: height variations' cooling consumption average comparison in kWh/m^2, to the right: height variations' bound correlated in kWh/m^2

an 18 % difference in cooling consumption for the first phase of the study and nearly 4% for the second phase (see Figure 4-9).

4.3.2.2.1.2 Built area ratio:

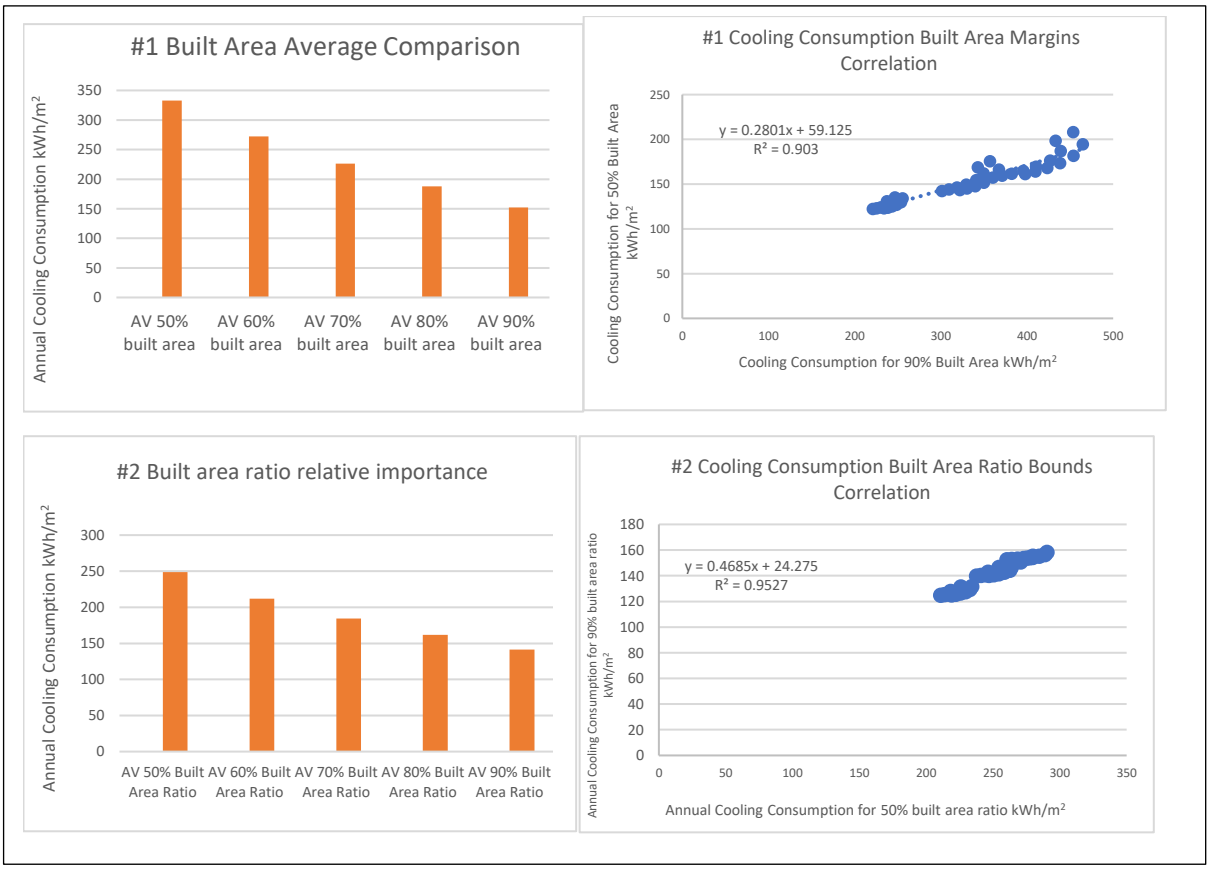


Figure 4-10 (#1) To the left: built area ratio variations' cooling consumption average comparison in kWh/m^2, to the right: built area ratio variations' bound correlated in kWh/m^2 for the first phase, (#2) To the left: built area ratio variations' cooling consumption average comparison in kWh/m^2, to the right: built area ratio variations' bound correlated in kWh/m^2 for the second phase

As mentioned before, there were five built area ratio variations. The results imply that there was a further change in the cooling energy consumption within the building. The study shows that the relationship between the built area ratio and the cooling energy consumption is a clear negative correlation: the denser the configuration, the more prevention of sun penetration. Therefore, cooling consumption is reduced heavily. A comparison of the calculations of the bounds (most and least dense group configurations) showed that there is a 72% difference in cooling consumption for the first phase, and, with the change of material for the second phase, the difference reaches a 54% change between the bounds of the five groups (see Figure 4-10).

4.3.2.2.1.3 WWR

For WWR variations, the relationship with cooling energy consumption for the first phase is a positive correlation that becomes less steady in the second phase. This can be caused by the climate zone chosen for the study. The difference in the cooling energy consumption is more significant than that shown for heights but still less than that shown for the built area ratios effect. Comparing these variable limitations indicates that there is almost a 66% difference in cooling consumption for the first phase. However, this number decreases to a 22% difference in consumption (Figure 4-11).

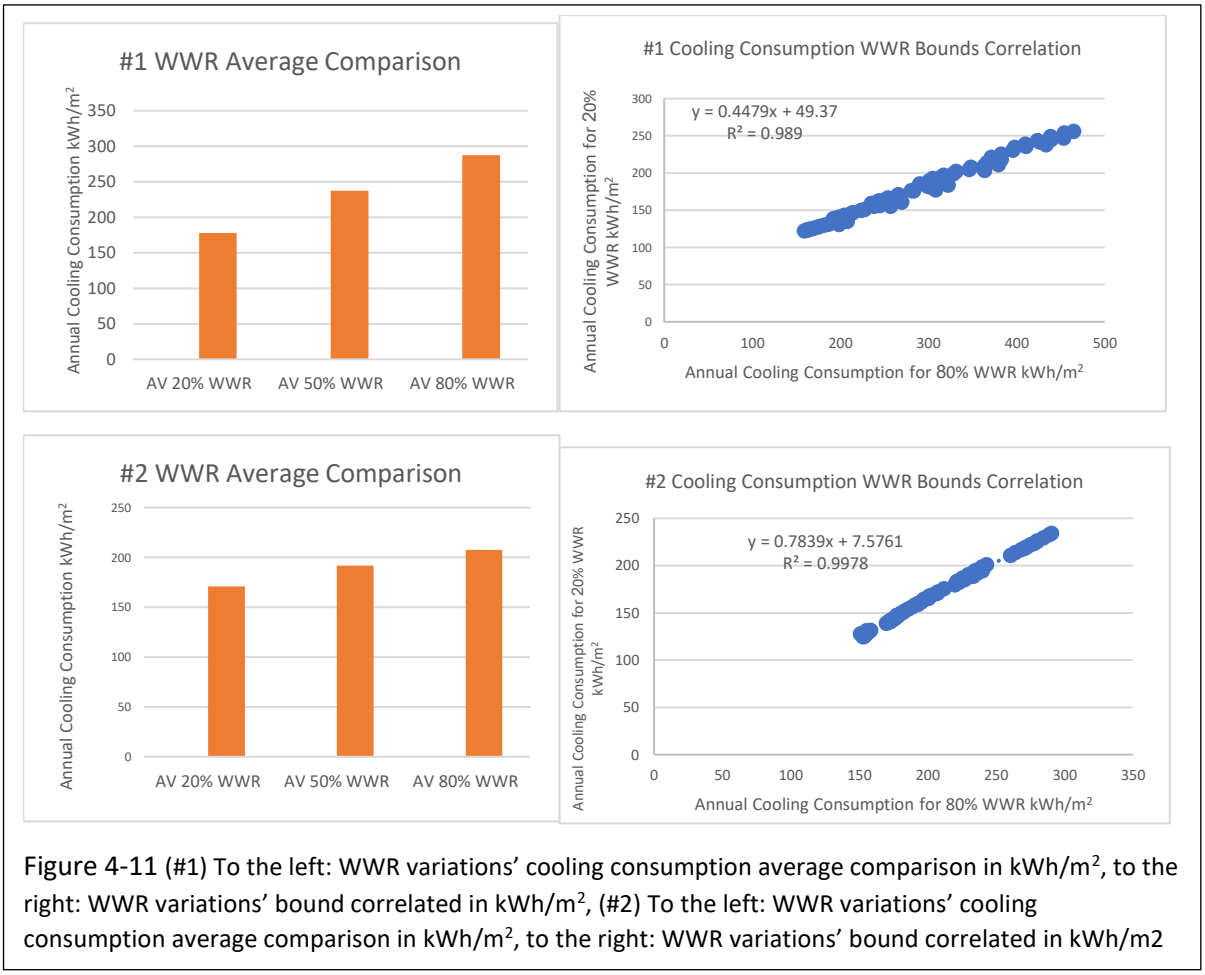


Figure 4-11 (#1) To the left: WWR variations' cooling consumption average comparison in kWh/m^2, to the right: WWR variations' bound correlated in kWh/m^2, (#2) To the left: WWR variations' cooling consumption average comparison in kWh/m^2, to the right: WWR variations' bound correlated in kWh/m2

4.3.2.2.1.4 Orientation:

There is a slight positive correlation for this variable, according to the results. Comparing the two variations for both tests, it can be argued that there is an 8% cooling energy consumption difference that exists between the two different angles for the first phase. While, for the second phase, it decreases to a 6% difference in consumption between the two different angle groups (Figure 4-12).

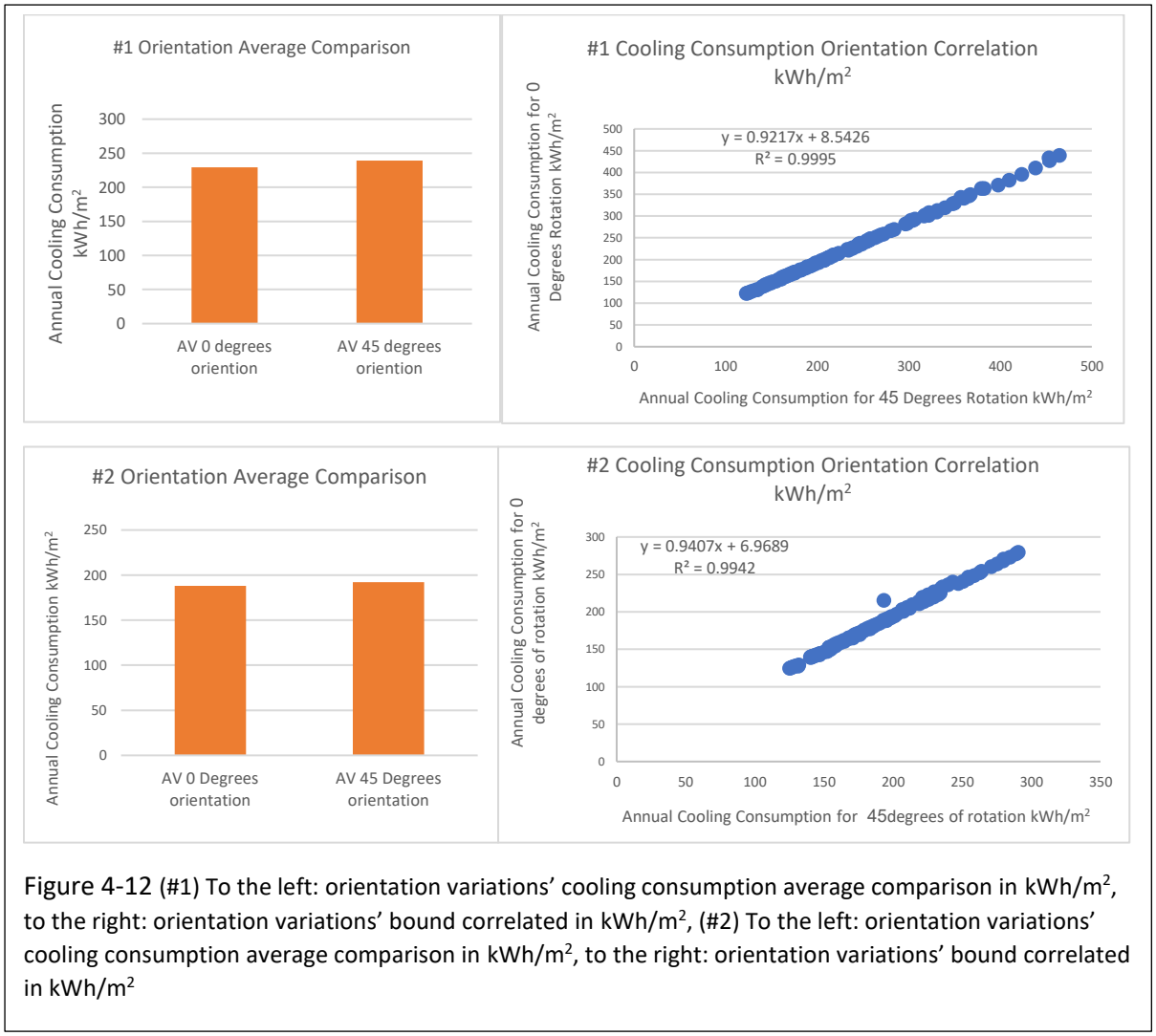


Figure 4-12 (#1) To the left: orientation variations' cooling consumption average comparison in kWh/m², to the right: orientation variations' bound correlated in kWh/m², (#2) To the left: orientation variations' cooling consumption average comparison in kWh/m², to the right: orientation variations' bound correlated in kWh/m²

4.3.2.2.2 Variables' relative importance for London, UK (Heating)

The relative importance of geometrical variables will be discussed in this section, with the same structure for heating demand in London, UK.

4.3.2.2.2.1 Heights:

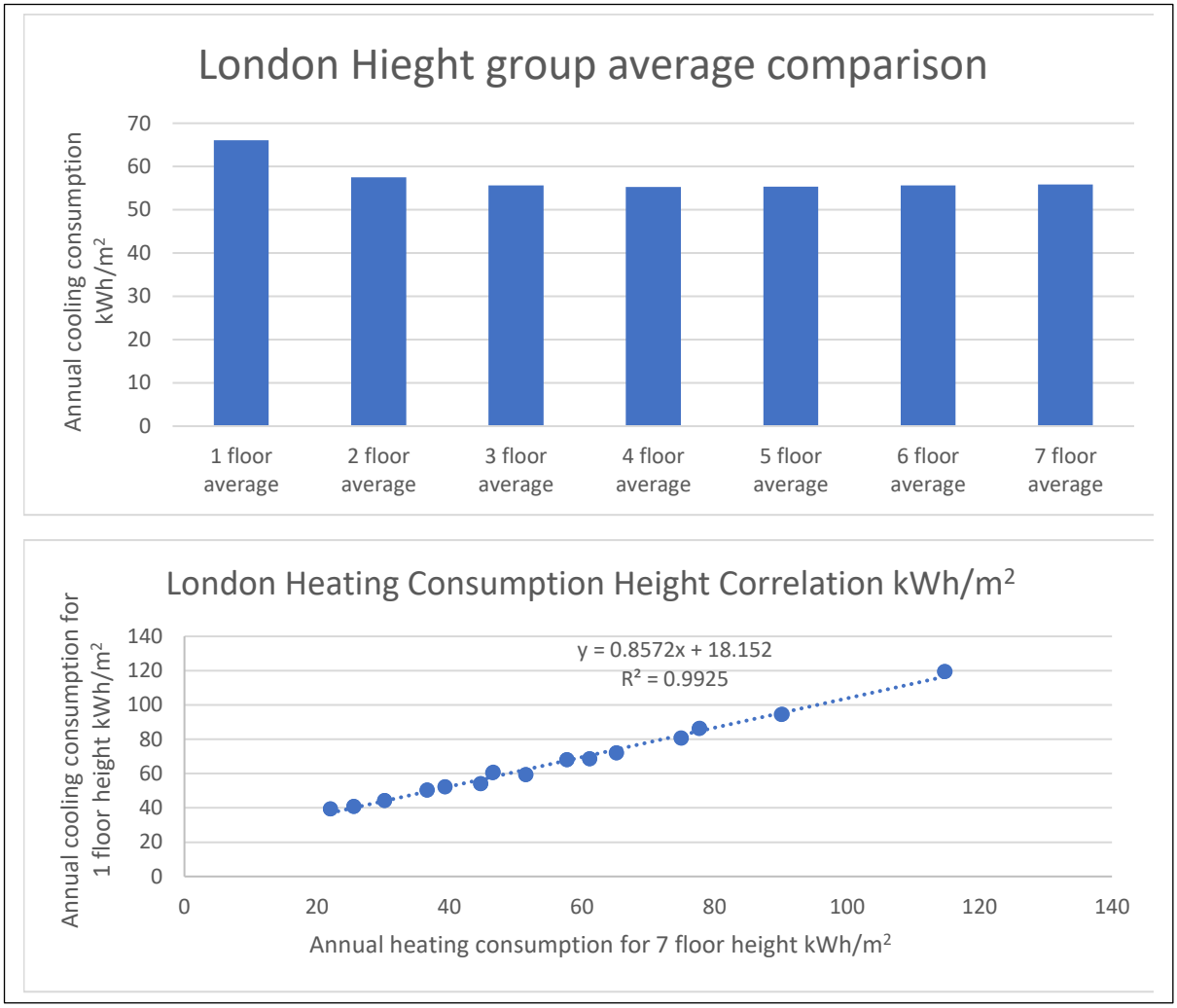


Figure 4-13 Different height groups' relative importance for heating, cooling and lighting (with dimming) energy consumption for London, UK

The average value for heating consumption in the groups for one and two floors has decreased with more floors. While it kept on decreasing for the rest of the groups, There was a minimal decrease in values when it came to the average. As shown in Figure 4-13, the gap between the maximum and minimum value for heating consumption followed the same pattern where it became more considerable for the first two groups. Then the change appeared to be un-noticeable. The heating consumption correlation between the highest and lowest group was almost 16% for London simulation results

4.3.2.2.2.2 Built area ratio

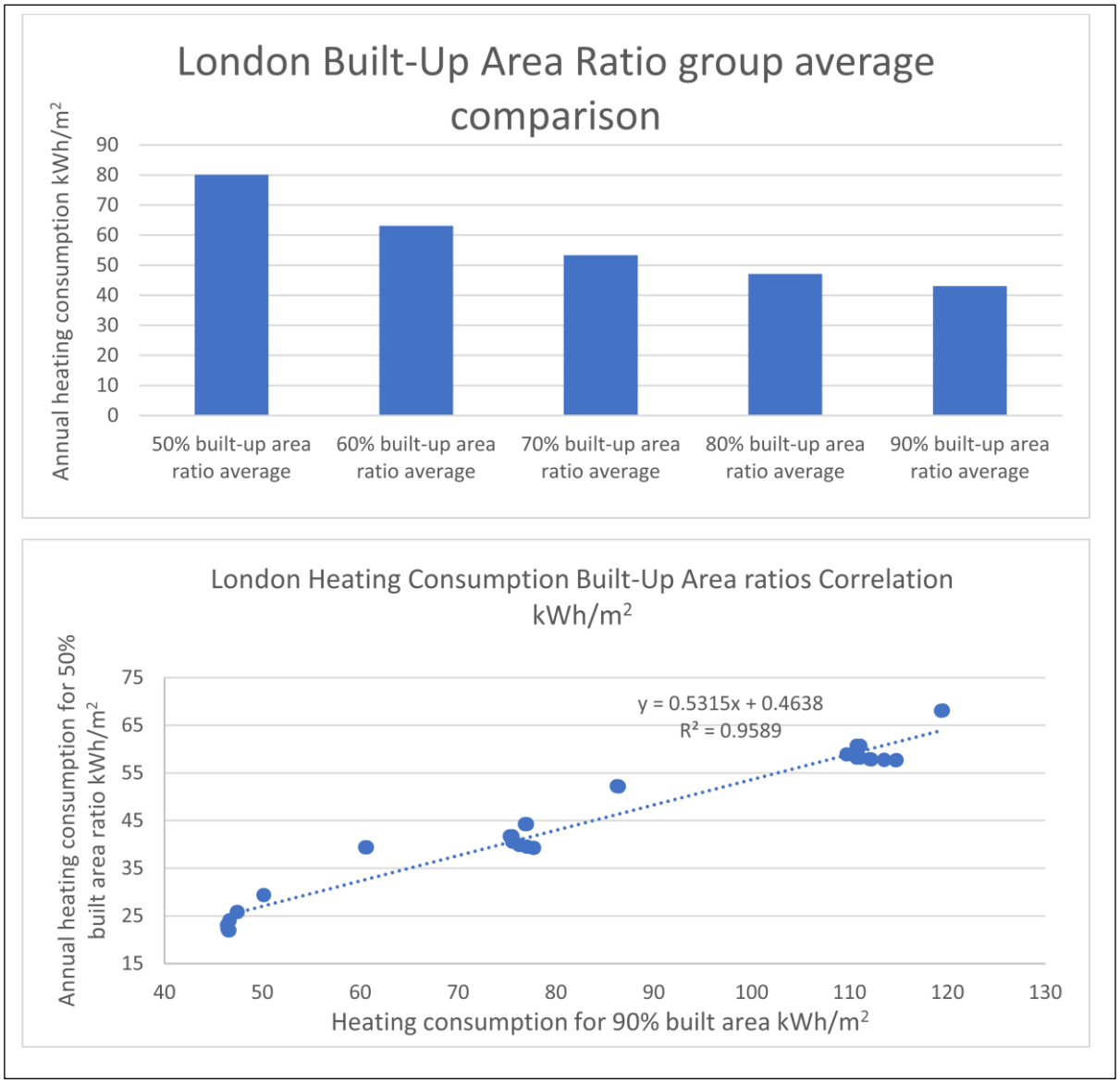


Figure 4-14 Built-up area ratio variations' heating consumption average comparison in kWh/m², built-up area ratio variations' bound correlated in kWh/m2 for London, UK

Built-up area ratios also had the most significant impact on heating consumption as the difference occurred in varying the largest and least ratios; they scored 47% for London. This is a significantly higher impact than the height variable impact. The average for heating consumption showed a descending pattern with the rise of built-up area ratios as shown in Figure 4-14, for London analysis results.

4.3.2.2.2.3 WWR:

WWR has a correlated increase impact on heating consumption. This is expected as the larger the ratio; the more exposure zones will have and consequently more heating demand. This impact was also shown in the bound's correlations for the two weather files with a 56% (see Figure 4-15).

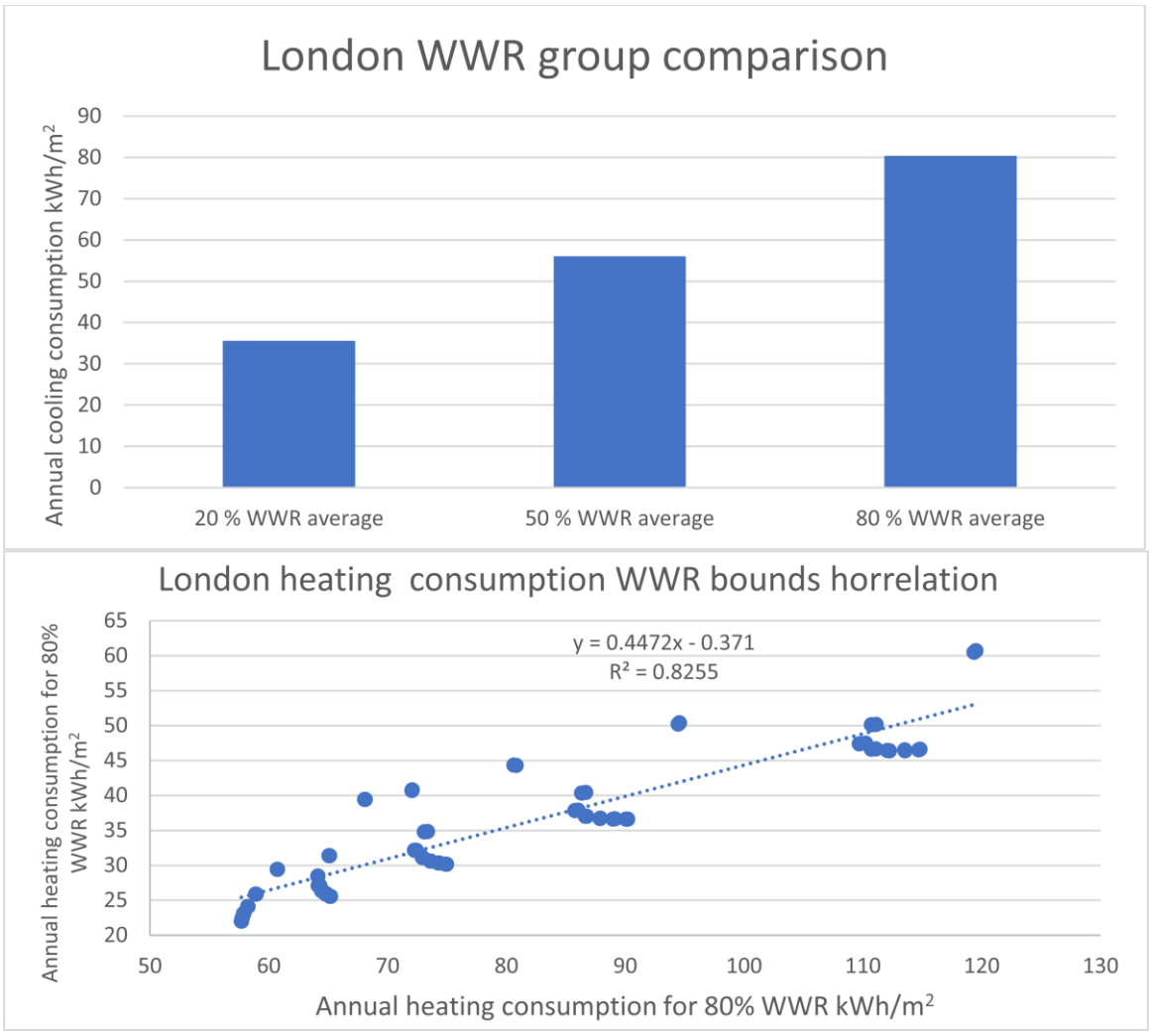


Figure 4-15 WWR variations' heating consumption average comparison in kWh/m2, WWR variations' bound correlated in kWh/m2 for London, UK

4.3.2.2.2.4 Orientation:

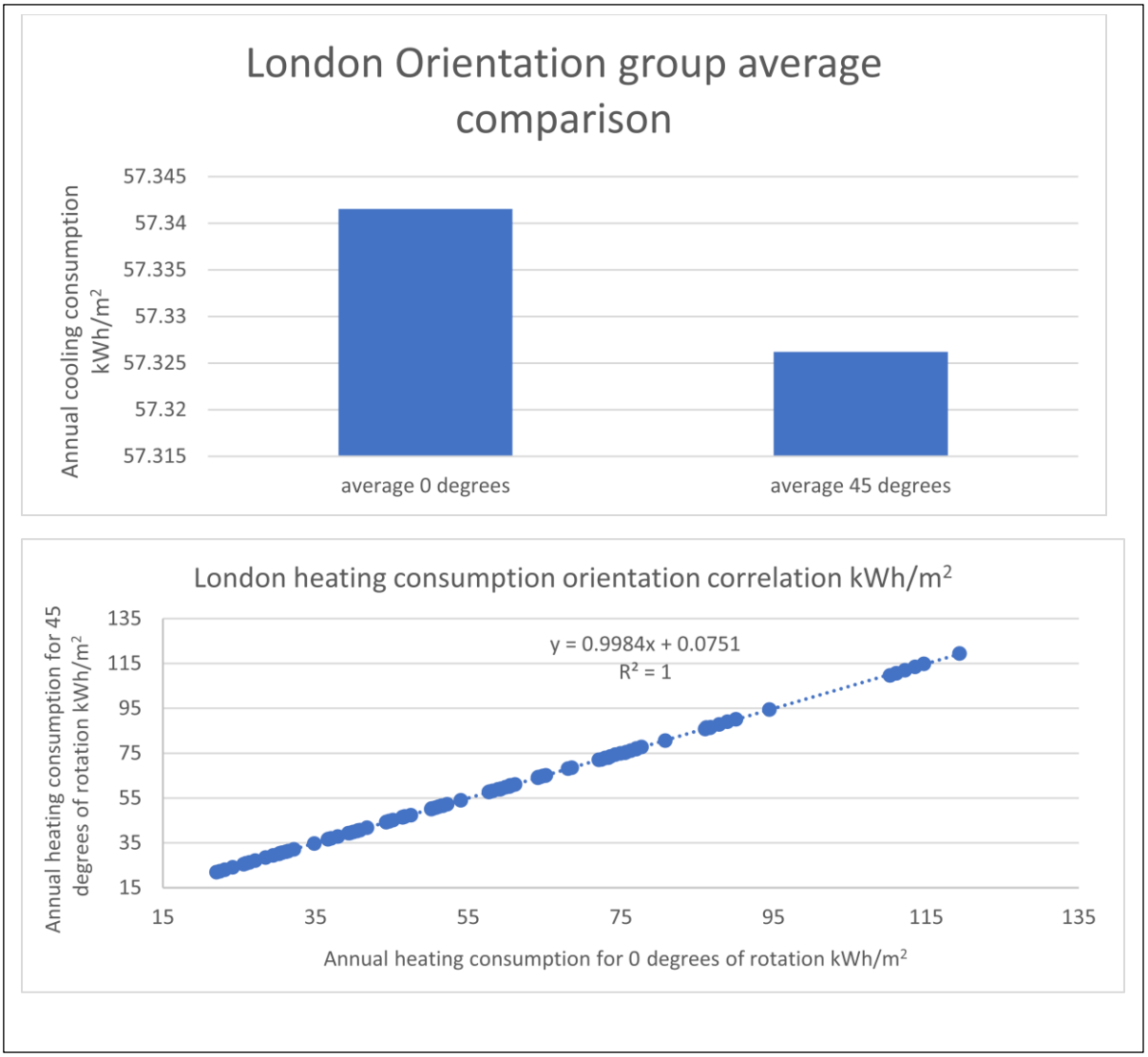


Figure 4-16 Orientation variations' heating consumption average comparison in kWh/m2, orientation variations' bound correlated in kWh/m2 for London, UK

The change of orientation had a minimal impact on performance, as shown in Figure 4-16. This might be caused by the equal distribution of fenestrations on the four different facades of the tested buildings. This minimal impact is clearly shown in the groups' correlation graphs, which were less than .01% different between the two orientation degrees for each tested iteration.

4.3.2.2.3 Summary

The initial comparison of the results between Aswan literature-based material and the ASHRAE set of materials has proved the evident similarity of the consumption patterns for cooling. Also, it has been proved that changing the lighting control system did not significantly impact energy demand for cooling through the two assigned sets of materials. This led to the next stage of the study, which focuses on different climatic conditions with higher expected heating energy consumption to investigate the validity of the findings in the first stage. The insignificant impact of lighting control changes on energy performance persisted in being apparent for the heating performance for London and Birmingham, as it appeared in the Aswan results (see appendix A) .

When it comes to the relative importance of the geometrical variables on heating and cooling demand for the tested climates, the results had shown a clear difference between each variable's impact on the demand. The geometrical variables' relative importance summary is shown in Figure 4-17, as a radar scale that starts from 0% to 60% at its top value. According to the tested samples, the lowest impact was the change of orientation for heating focused zones in London with values of 1%. The variation of heights scored the lowest impact percentage for Aswan as it had an impact near to 5%, with a 6% change of cooling demand caused by the variation of orientation in the Aswan weather file. Heights' variation for London had a second to lowest impact with a 15% change of energy demand for the bounding groups of variants. Then the highest two impacts were also different for the cooling demand zones.

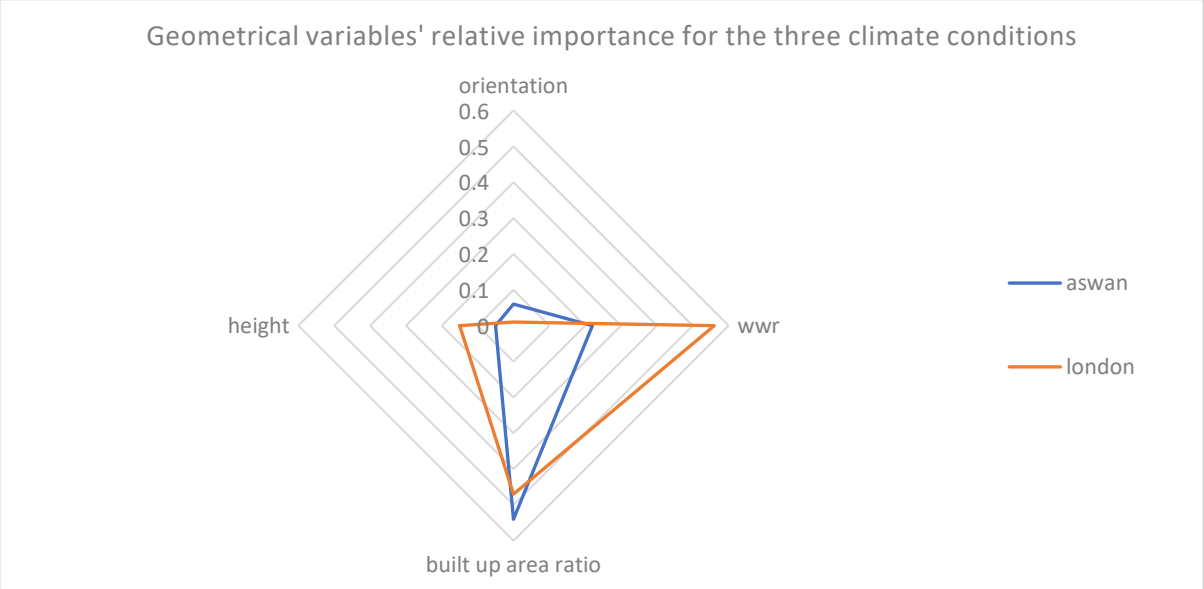


Figure 4-17 Geometrical variables' relative importance for the two tested weather files

Built-up area ratios were first with a 54% possible change of cooling, followed by the WWR, which showed a difference of 22% of cooling demand. WWR had a significant impact on the heating demand zones in London with 56%, followed by the built-up area ratios causing an impact of 47% in London's heating demand.

As shown in the results showcase, this analysis was conducted on the scale of zones in the tested buildings. The ASHRAE materials simulation results will be discussed on a zone level in the following section. This section will illustrate both cooling results for Aswan, Egypt, ASHRAE applied results and heating demand for London, UK, ASHRAE applied results. This provides a clearer understanding of the geometrical variations' impact on performance for the zone scale.

4.3.3 Detailed zone energy demand analysis

In this section, the results of zones will be discussed based on zones instead of the whole building performance. This provides a clear illustration of the performance for heating and cooling demand based on orientation and how the height difference will affect the performance for each zone and in each orientation. This simply meant calculating the results based on the zone's area instead of the whole building's area. So, each zone had its consumption normalised to its area. For example, one-floor iterations had five-zone results, one for the core and four for the perimeter. The core result was excluded from this discussion as it mainly was zero or near zero because there was no fenestration assigned to it. These results were calculated by sorting the zone results into seven sheets of results based on the number of floors and the number of zones. Then, these sheets consisted of two groups, each for the orientation of the iterations. An average is calculated for each zone orientation. This provided the results of zones in an azimuth original four orientation and the rotated diagonal four orientations. For each of those eight orientations, the results are merged to have a readable illustration for each orientation and each floor (see Figure 4-18, Figure 4-21, Figure 4-24, Figure 4-27, Figure 4-31, Figure 4-33 and Figure 4-36). The results in this section will only focus on cooling demand for Aswan and heating demand for London results.

4.3.3.1 *One-floor results analysis*

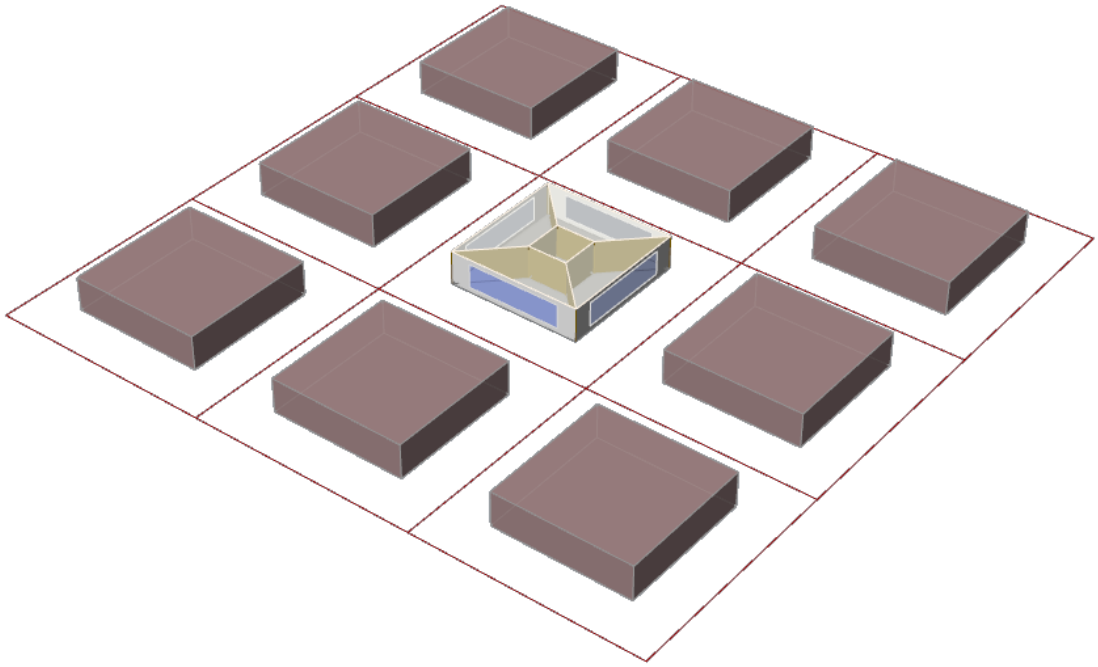


Figure 4-18 One-floor prototype with 50% WWR and 60% built area ratio and no rotation

As shown in Figure 4-19Figure 4-20, there is a clear difference between the two demands for heating and cooling overall consumption for the different orientations. For the Aswan cooling demands, the consumption peaks for the west and southwest orientations. Also, there are some high cooling consumption demands over the south, east and southeast zones. The consumption declines significantly for zones with north and northeast, and northwest

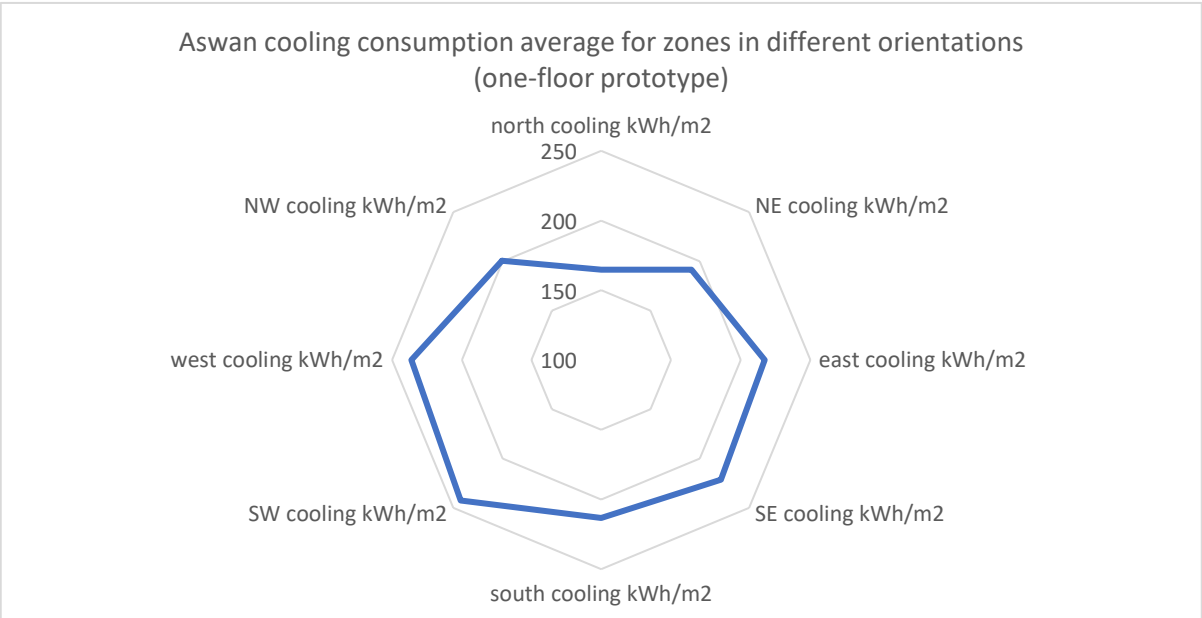


Figure 4-19 One-floor prototype cooling consumption results average per orientation for Aswan, Egypt

orientations, which is expected due to the sun exposure and the hours of high temperature in the location, from noon till sunset. For London's heating demand, It is illustrated in Figure 4-20 that north-oriented zones have peak heating demands for one-floor buildings iterations. The more south-oriented, the less demand the zones get for heating consumption. This is illustrated in the results of zones oriented towards north, northwest and northeast with the highest heating consumption demand values. However, zones with south, southeast, and southwest had the lowest radar graph values. This indicates its low consumption values for heating demand. Zones with direct east and west orientation are the middle grounds for the two high and low consumption orientation groups.

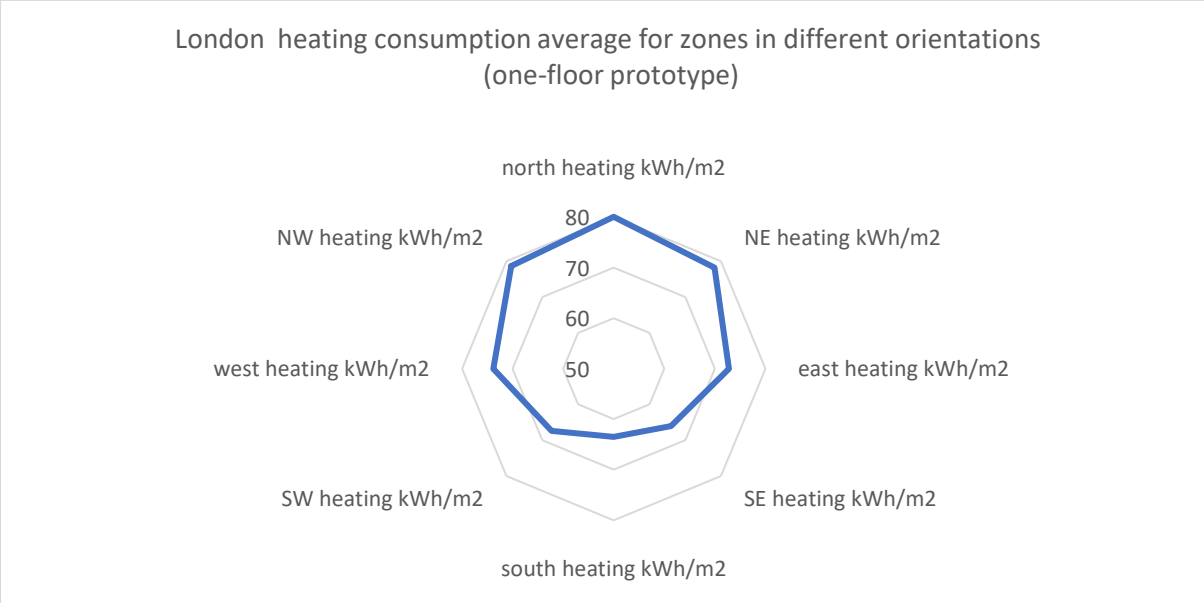


Figure 4-20 One-floor prototype heating consumption results average per orientation for London, UK

4.3.3.2 *Two-floor results analysis*

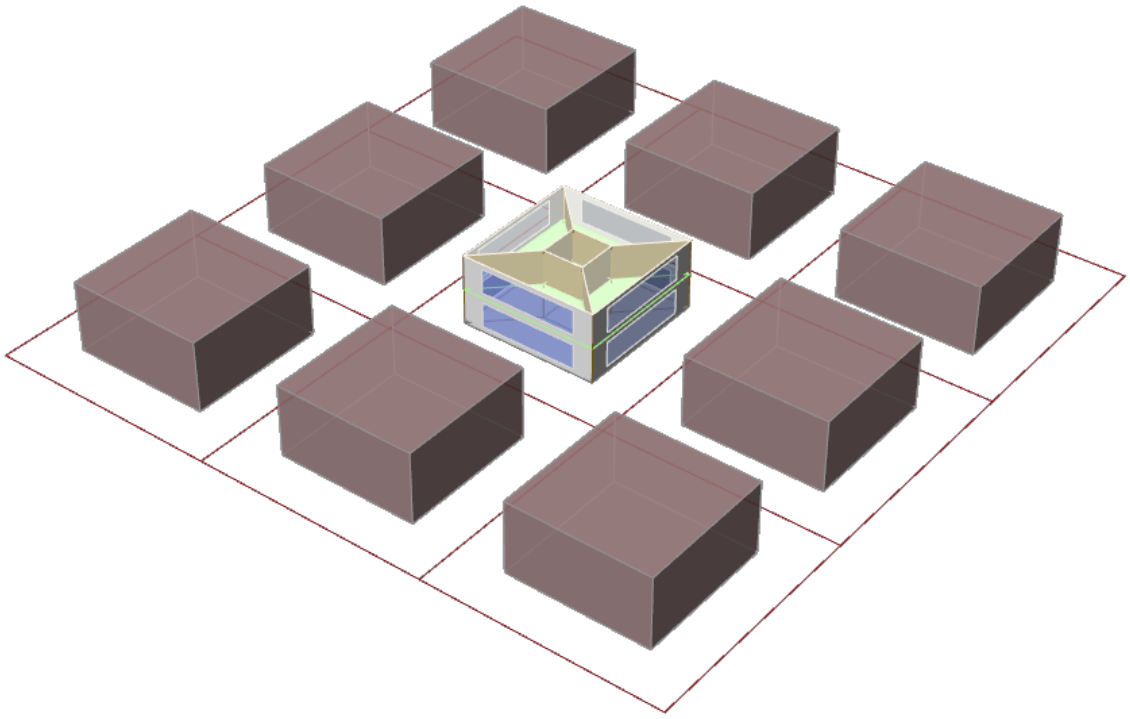


Figure 4-21 Two-floor prototype with 50% WWR and 60% built area ratio and no rotation

Iterations with two-floor height had another layer of consumption for the first-floor zones with the eight orientations. For the Aswan cooling demand, the first floor echoes the performance of the ground floor, and both do not show a difference of pattern in consumption from the one-floor prototypes. One important note is that the cooling demand

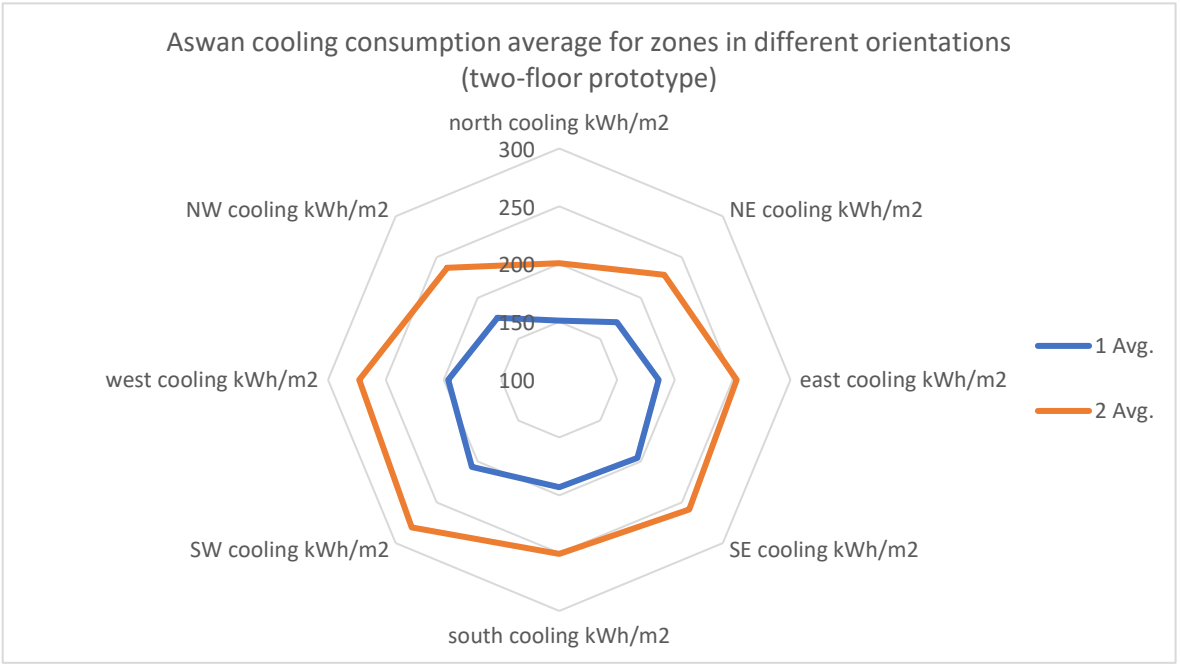


Figure 4-22 Two-floor prototype cooling consumption results average per orientation for Aswan, Egypt

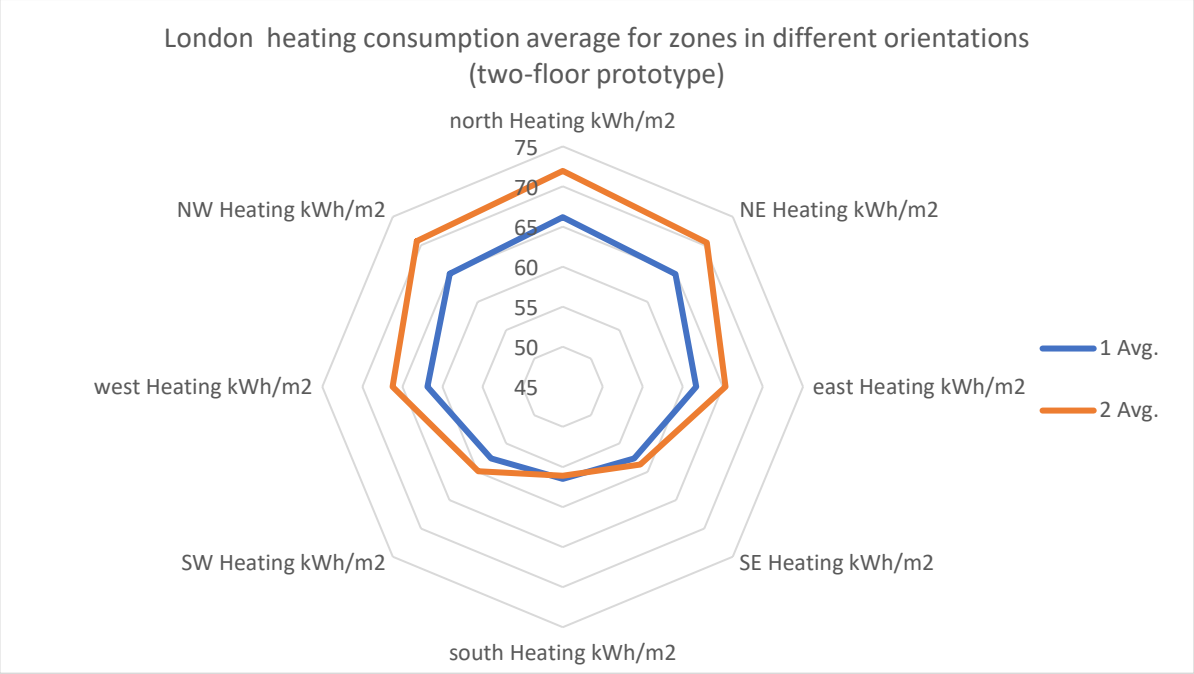


Figure 4-23 Two-floor prototype heating consumption results average per orientation for London, UK

values for the ground floor are less than the first-floor cooling demand values by a clear margin. Also, the peak values in the west and southwest zone orientations are less in the ground floor than their equivalent in the one-floor prototype zone with the same orientation, while the lower values stayed close to values for both prototypes. This is caused by the effect of the first floor reducing the exposure for the ground floor while it is exposed directly to the sun and having higher values in cooling demand, but with the same order when it came to orientation, creating almost a parallel line to the ground floor cooling consumption values (see Figure 4-22). For heating demands in London, there was no parallelism between the two-floor consumption pattern based on orientation. However, high and low consumption patterns remained the same when it came to north-oriented zones being the peak of heating demand and south oriented zones being the lowest heat demanding zones. The exposure to the sun did play a role in reducing the heating demand in the lower consumption zones, as the south floor-oriented zone got a slightly lower average of performance when it was compared to ground floor heating. This does not continue for the southeast and southwest oriented zones as the first-floor zone average are higher than the ground zone for the same orientation, and the gap is more significant for the southwest than the southeast. The gap of increasing heating demand between the first and ground floors keeps rising until it reaches its peak, with the heating demand for both floors average for north-oriented zones. The graphs also illustrate the difference in values between the two heating demand zones. In

these cases, exposure to a cold climate played a contrasting role to what had happened in the hot zone. The first floor needed more heating consumption than the ground floor, as shown in Figure 4-23.

4.3.3.3 *Three-floor results analysis*

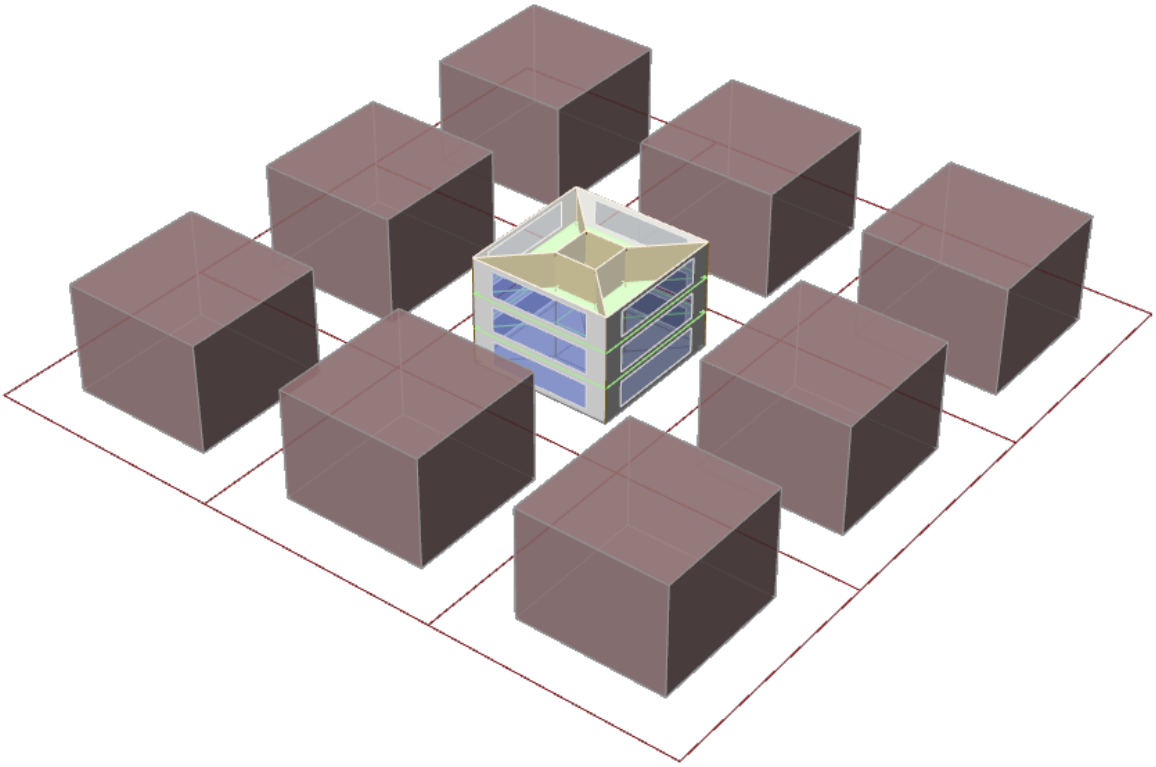


Figure 4-24 Three-floor prototype with 50% WWR and 60% built area ratio and no rotation

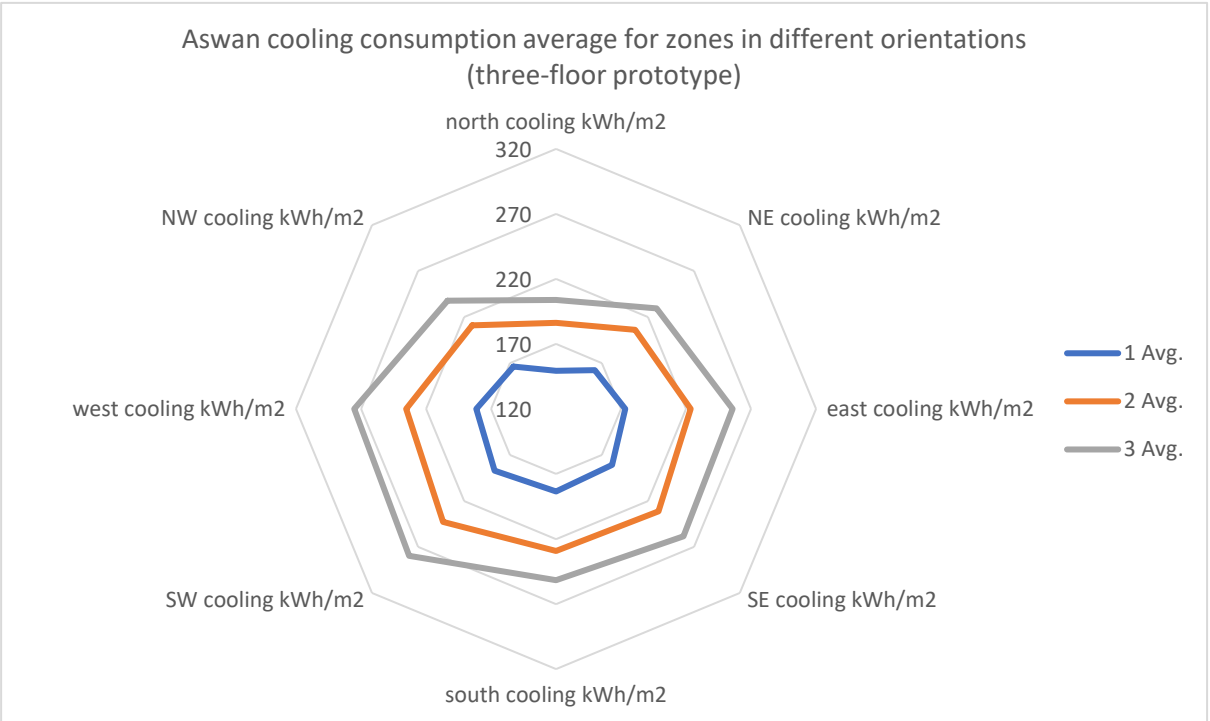


Figure 4-25 Three-floor prototype cooling consumption results average per orientation for Aswan, Egypt

Adding a third stack of zones to the model has almost the same impact as the effect caused by the first floor. As shown in Figure 4-25, this time, the ground floor cooling consumption reduced by a clear difference to the one-floor prototype. The same offset results for upper floors still occur in the average results for this prototype. There is a more significant difference between the ground and first-floor average cooling consumption than the difference for first- and second-floor averages for the eight different orientations. For London's results, the ground floor orientation averages seem to have closer results to each other. This might be caused by the context overshadowing neutralising the effect of orientation on the heating demand for each zone. Marginal differences in heating demands for zone per orientation begin to show for the first-floor averages that keep the previous pattern of a northern high demand zone and southern zones with less heating demand. However, all the eight orientation averages are less than the ground floor results averages. The second-floor results contradict this as the northern zones (north, northwest and northeast) have a clear difference in heating demand average values than the same values for the ground floor.

Moreover, the southern zones (south, southwest and southeast) still fall below the values of the same orientation for the ground floor. The east and west orientations are slightly more than ground floor values. (see Figure 4-26)

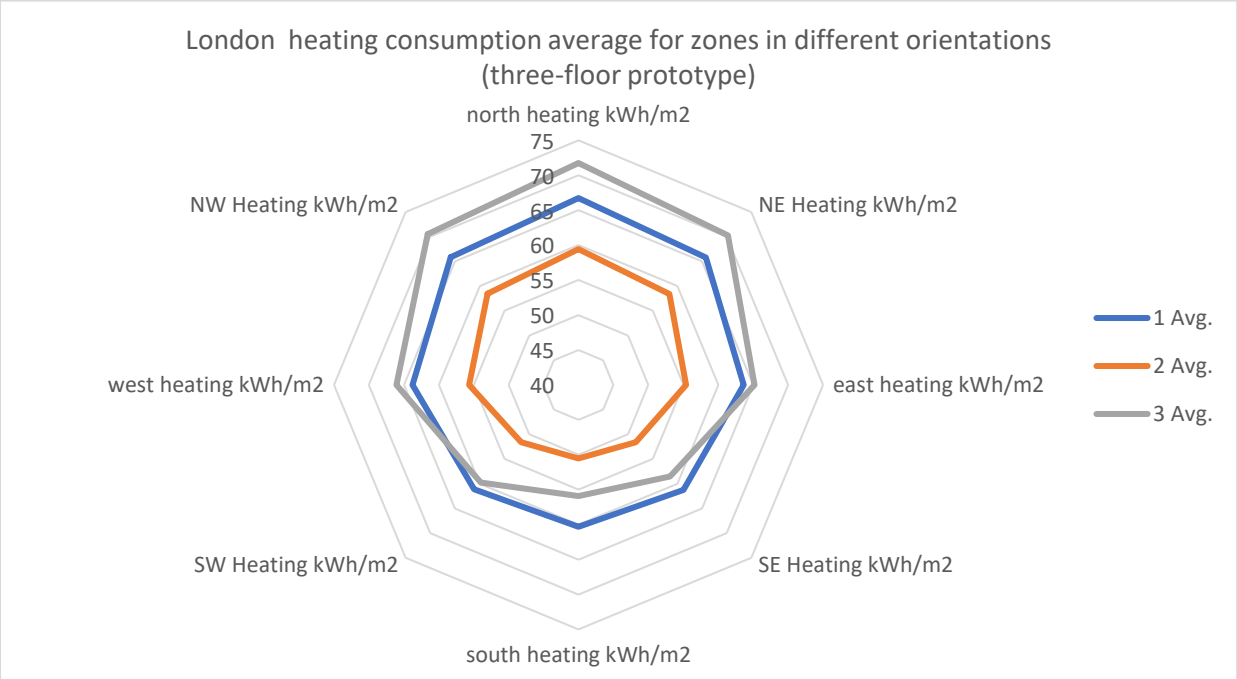


Figure 4-26 Three-floor prototype heating consumption results average per orientation for London, UK

4.3.3.4 *Four-floor results analysis*

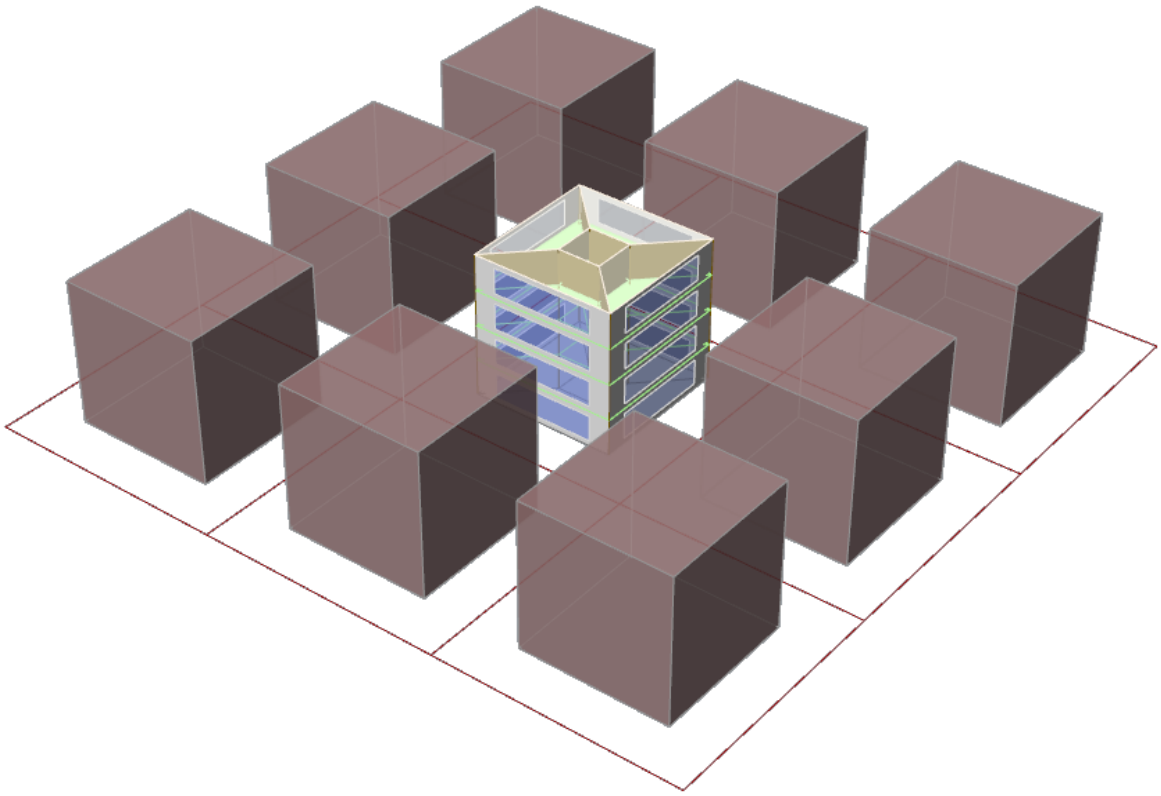


Figure 4-27 Four-floor prototype with 50% WWR and 60% built area ratio and no rotation

The pattern of cooling consumption differs between floors for this prototype in a different way than the previous prototypes. In this prototype, the average cooling demand values for

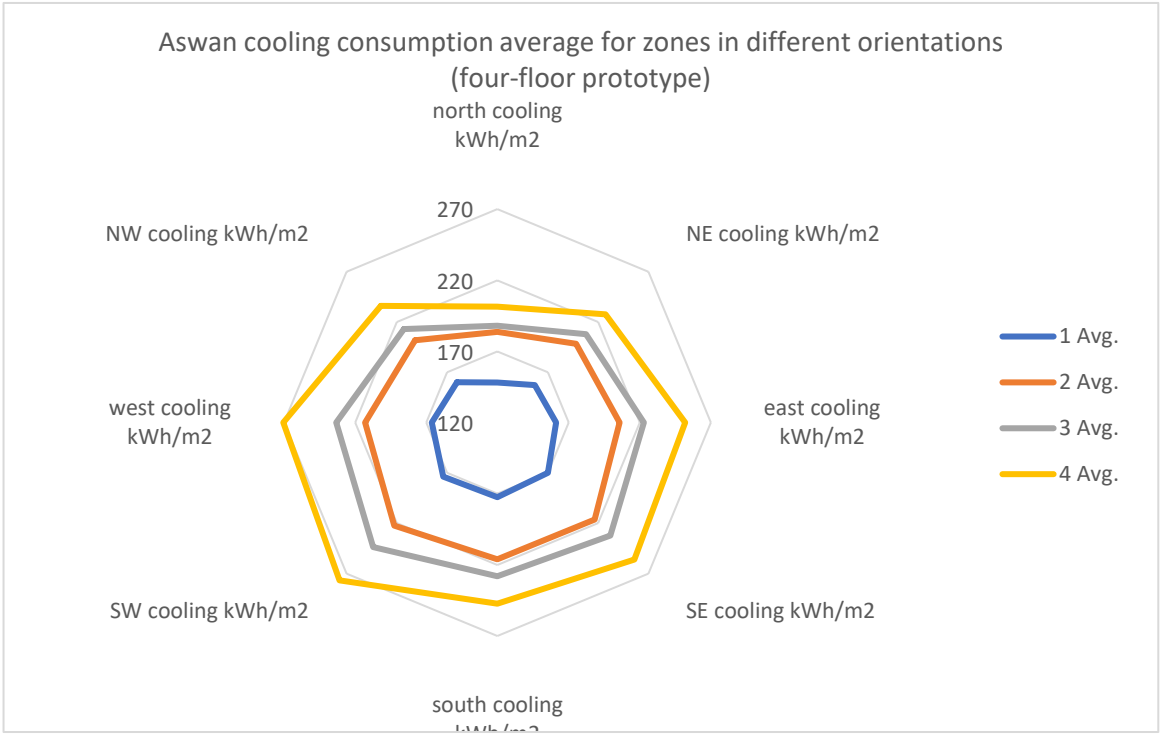


Figure 4-28 Four-floor prototype cooling consumption results average per orientation for Aswan, Egypt

the ground floor have a significant decrease from the first and second floor, while the third floor also has a clear boost from its lower floors (see Figure 4-28). This boost is more noticeable with the peaking of west and southwest zone averages. At the same time, it decreases for the averaged results of the north zone and almost diminishes for the difference between the first and second floors. Nevertheless, the gap between the ground floor and the rest of the floors stays almost consistent. As shown in Figure 4-29, a similar relationship also emerges for the heating demands in London, in the sense that the ground and third floors seem to have different averaged results, being exposed either to ground or sky, than the first and second and third floors as non-exposed floors. The close averaged results for the eight orientation in the eight directions continues to occur for this prototype but with a more evident rise for the northern expected higher consumption values than the southern zone results. Although the northern zones averaged results for heating demand show similar values for the first and second floors, the southern zones have a difference in heating demand as the higher floor gets less heating demand in the southern oriented zones. However, this does not continue for the third floor, as its southern oriented zones have higher consumption values than the second and first floors but still less than the ground floor. The third floor in the northern zones reached 72.25 kWh/m2 for average heating demand in a north orientation for London's results and 66.33 kWh/m2 for Birmingham's results. This made the north third

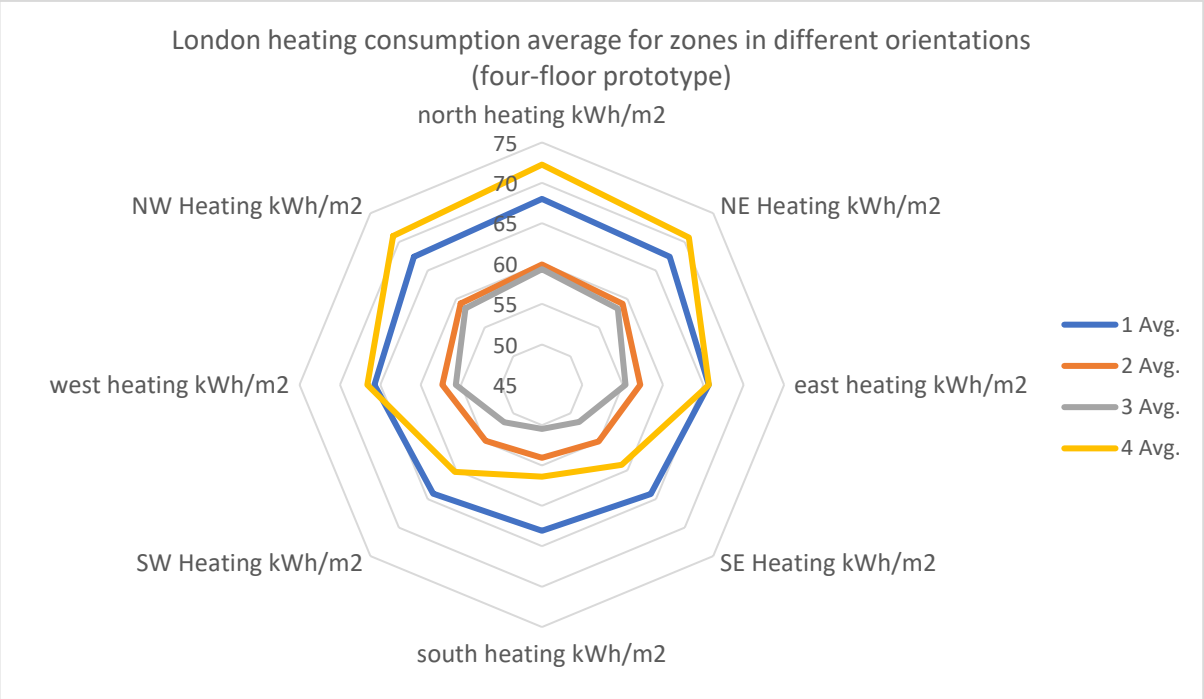


Figure 4-29 Four-floor prototype heating consumption results average per orientation for London, UK

floors' average heating consumption the highest in all four floors and eight orientations in this prototype.

4.3.3.5 *Five-floor results analysis*

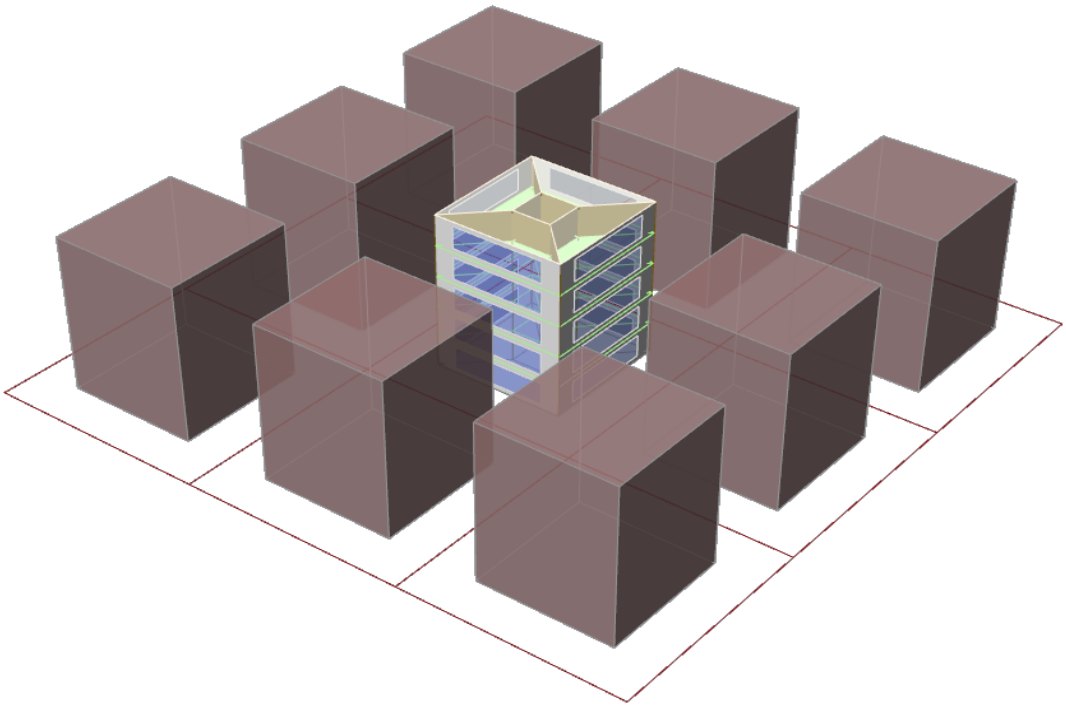


Figure 4-31 Five-floor prototype with 50% WWR and 60% built area ratio and no rotation

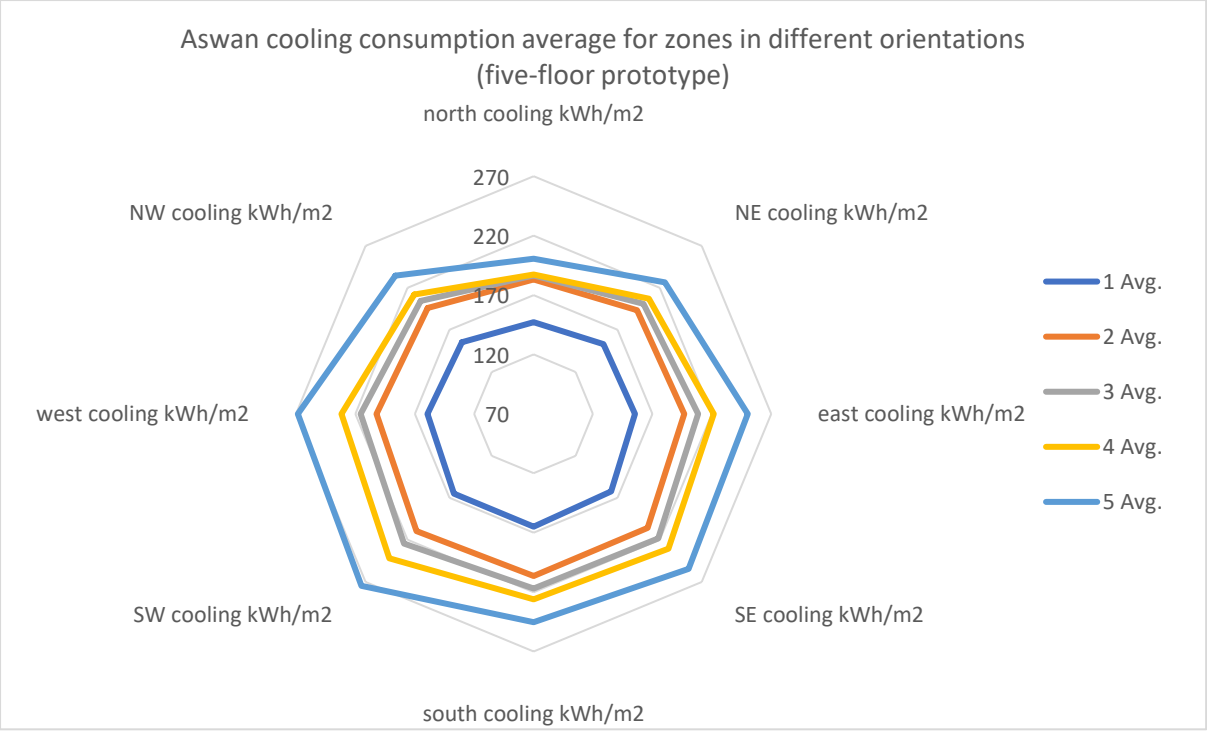


Figure 4-30 Five-floor prototype cooling consumption results average per orientation for Aswan, Egypt

This five-floor prototype is more evidence that there are two performance groups according to floors based on exposure. The ground floor had minor cooling consumption by a clear margin with values close to equal on the eight different orientations. However, the peak of performance is still visual for the southwest and south zones' cooling averages. This peak in performance is continuous between different floors. However, the north cooling averages for the non-exposed floors in this prototype, first, second and third, are almost equal according to the graph in Figure 4-30, while the consumption differs for the southwest zones for the same three floors. There is a leap in cooling demand for the fourth floor eight oriented zones with a consistency of peaking in the southwest zone cooling average. The north has the most negligible value of cooling consumption compared to the same floor zones.

As expected, the heating demands for London are different from the Aswan cooling demand for this prototype. The only resemblance is the difference that appears in exposed and non-exposed floor performance. On the contrary to Aswan's cooling results, heating consumption does not show an evident rise with the increase of floor order as shown in the Aswan cooling demand. The ground floor starts with an overall relatively high heating demand in all eight orientation averages with a marginal difference between north and south results. This difference becomes apparent with the non-exposed floors. For the three floors, northern and

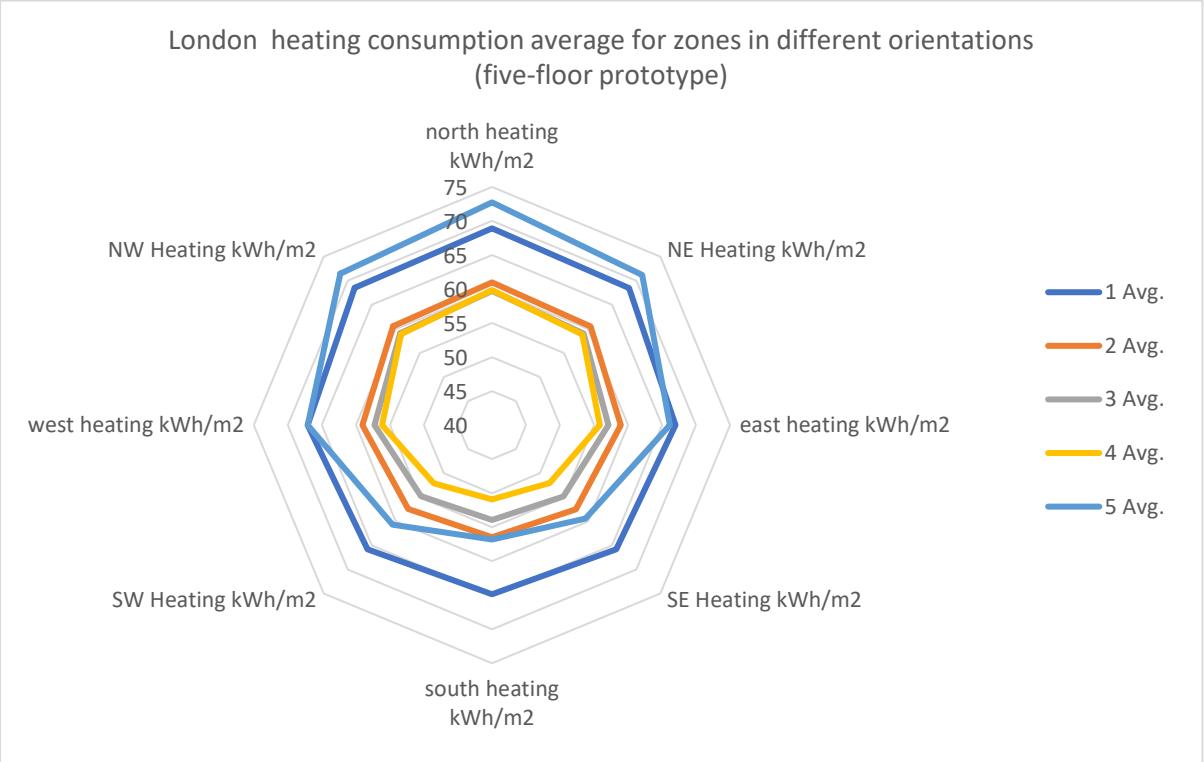


Figure 4-32 Five-floor prototype heating consumption results average per orientation for London, UK

southern zone heating consumption decreased with the increase of height. However, this decrease in values was more considerable for southern zones and barely noticeable for northern zones heating average demand values. East and west zones had the median values for each floor as the same consumption pattern is shown in different prototypes. For the fifth-floor heating demand, heating averages in all eight zones increase with the non-exposed floors. In comparison, the averages of the three southern oriented zones fall below the ground floor similar oriented zones and the northern orientation peaks over the ground floor averages for heating demand. All this is with a marginal difference in east and west heating demand values for both ground and fifth floors (see Figure 4-32).

4.3.3.6 *Six-floor results analysis*

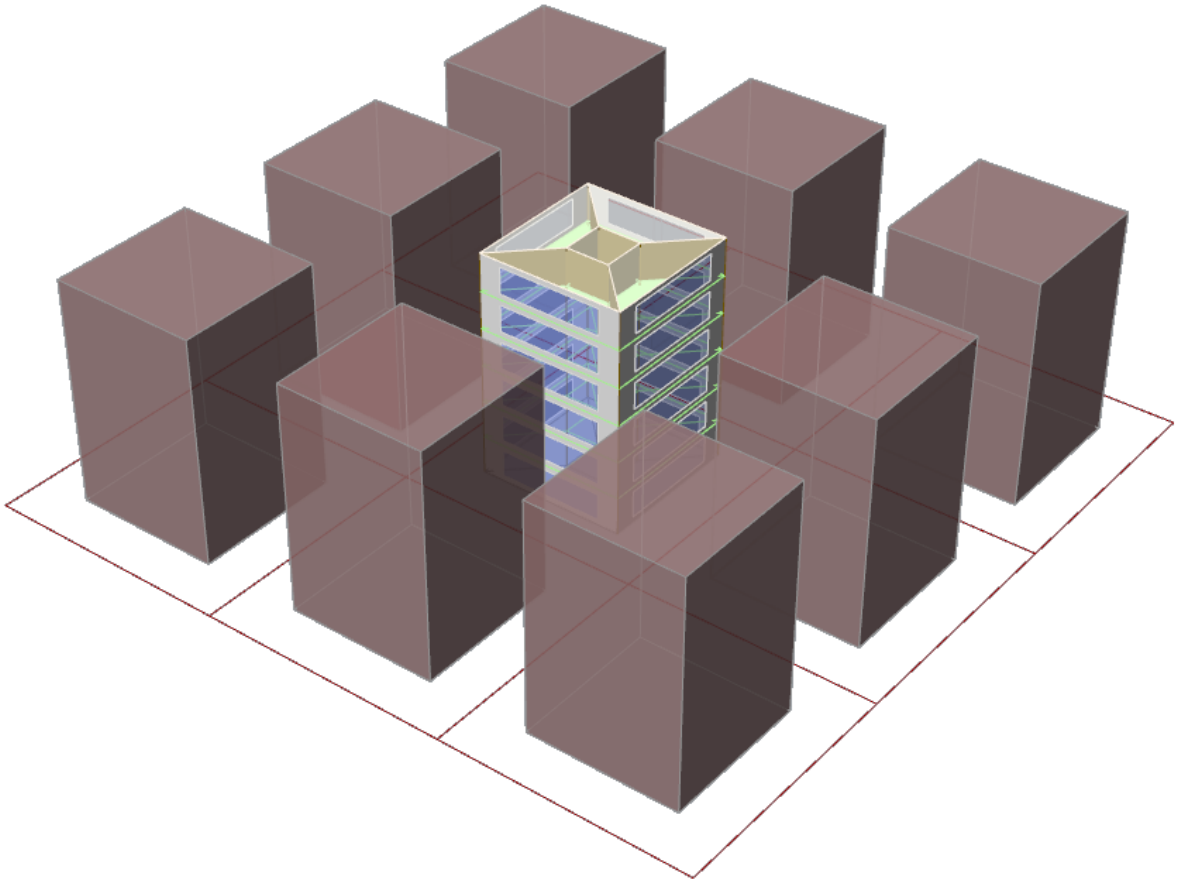


Figure 4-33 Six-floor prototype with 50% WWR and 60% built area ratio and no rotation

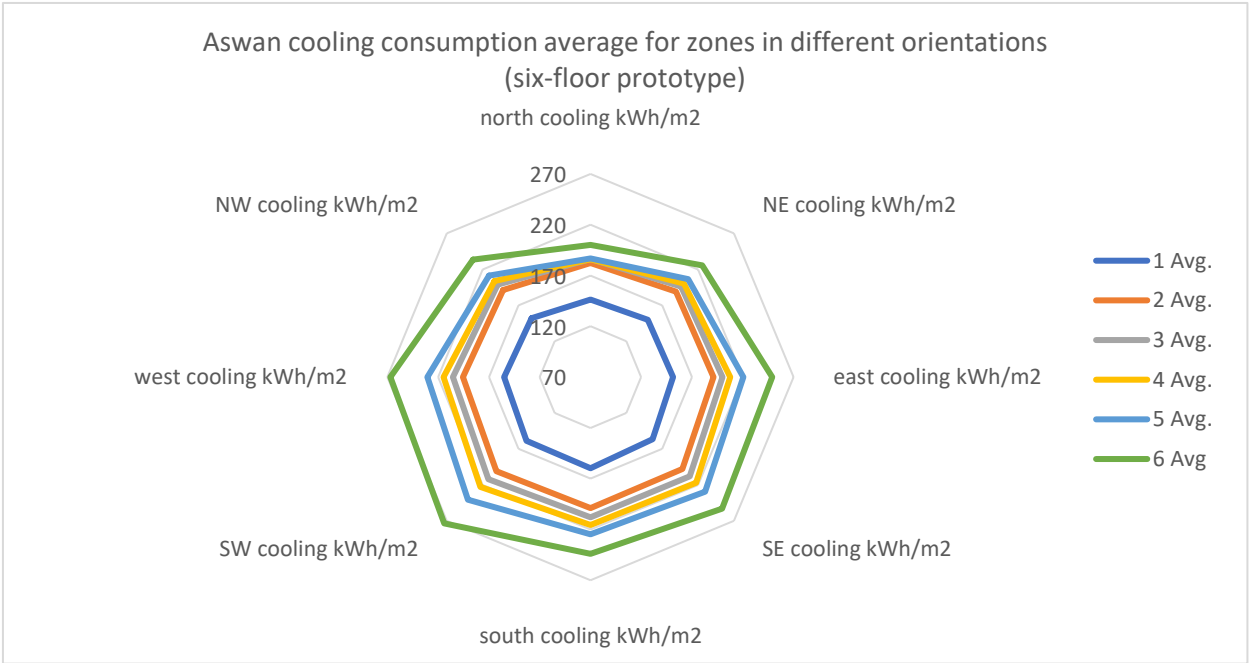


Figure 4-34 Six-floor prototype cooling consumption results average per orientation for Aswan, Egypt

Looking into the cooling consumption for this prototype, it is shown that the pattern of demand stayed the same as the five-floor prototype. The ground floor values were slightly reduced than in the previous prototype, while having more than three floors in the non-exposed stack of floors has shown that the proximity of results for the north-oriented zones for the first, second, third and fourth floors are still existing. Differently, the cooling demand average for the fourth floor in the southwest zone, being the highest non-exposed floor, records a further higher value of cooling consumption with a fair difference from the rest of the non-exposed floors. This is followed by the leap of cooling consumption for the fifth floor in the eight oriented zones following the same order of higher and lower zone consumption as previously discussed for other prototypes. (see Figure 4-34)

As shown in Figure 4-35, The results for London showed similar patterns as the five-floor typology in heating demand consumption. As previously illustrated, graphs show that the ground floor still has a high demand for consumption on all eight orientations. This is followed by a steep decrease of heating consumption for the first-floor averages in all eight zones, then keeps on slightly declining for the results of the following non-exposed floors, clearer for the southern oriented zones' heating demand averages than the results of heating consumption for the north-oriented zones. The fifth and final floor exhibits a higher heating demand average for the north-oriented zone while its south-oriented result is less than the south zone

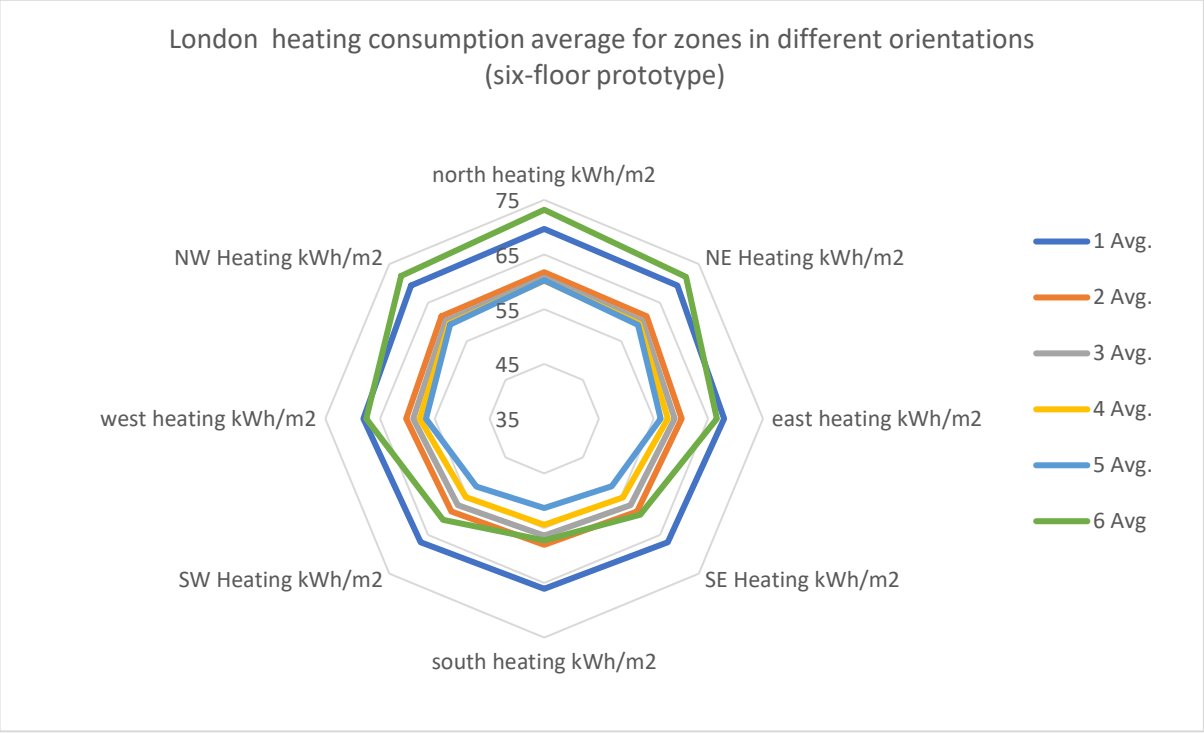


Figure 4-35 Six-floor prototype heating consumption results average per orientation for London, UK

heating average of the first floor. It can be argued that this effect is due to the overshadowing and its effect on the lower floors. In contrast, the full exposure of the last floor led its performance to follow the sun's exposure, lowering the demand for heating in the south over the year while it peaked towards the north-oriented zones.

4.3.3.7 *Seven-floor results analysis*

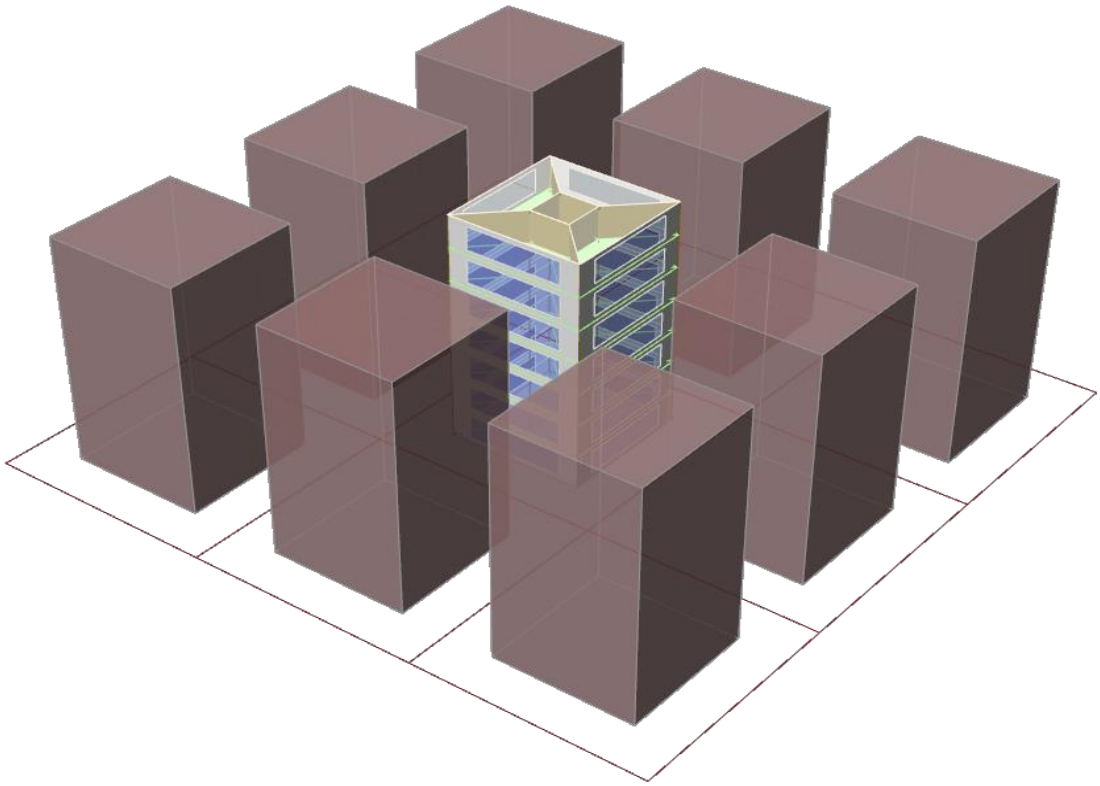


Figure 4-36 Seven-floor prototype with 50% WWR and 60% built area ratio and no rotation

The exposure effect is still apparent for both heating and cooling. For findings on cooling energy consumption, the ground floor still has the minor consumption with marginal peaking towards the direct south rather than the expected southern west zone average cooling

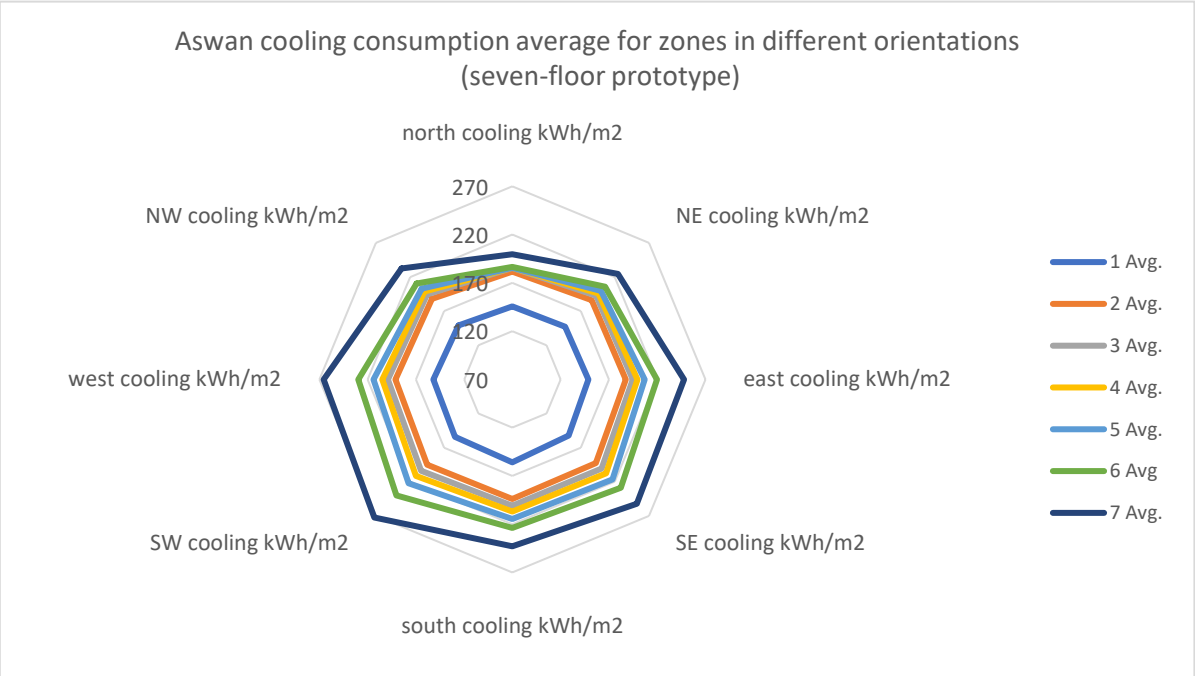


Figure 4-37 Seven-floor prototype cooling consumption results average per orientation for Aswan, Egypt

demand. The increase of demand from non-exposed floors in all eight directions happens for the averaged results of the floors from the first to the fifth, with the regular peaking of southwest and west zones cooling averages. Lastly, the sixth floor shows the performance gap due to its roof exposure, although it has relatively close values to the exposed fifth floor in the five-floor prototype. This can be reasoned due to the similarity of the context condition and its overshadowing effect for both averages (see Figure 4-37)

On the other hand, heating demand shows a difference in consumption pattern and the results of relativity between floors. As shown in Figure 4-38, the ground floor difference and rise of performance persists to appear in this prototype. Following this, the non-exposed floors have a slight decline in heating demand averages for the first, second and third floors. The previously discussed southern oriented zones decrease in heating demand averaged values are indicated clearer for the fourth and fifth floors. This can also signify the overshadowing effect and its control over sun penetration to different floors in a building. The final sixth floor shows a similar pattern of performance and similar values for heating demand to the fifth floor of the five-floor prototype discussed earlier. Although the north directed zone of the top floor has the peak value, the south directed zone of the same floor has an average heating demand that falls below the value of its similar oriented zone in the

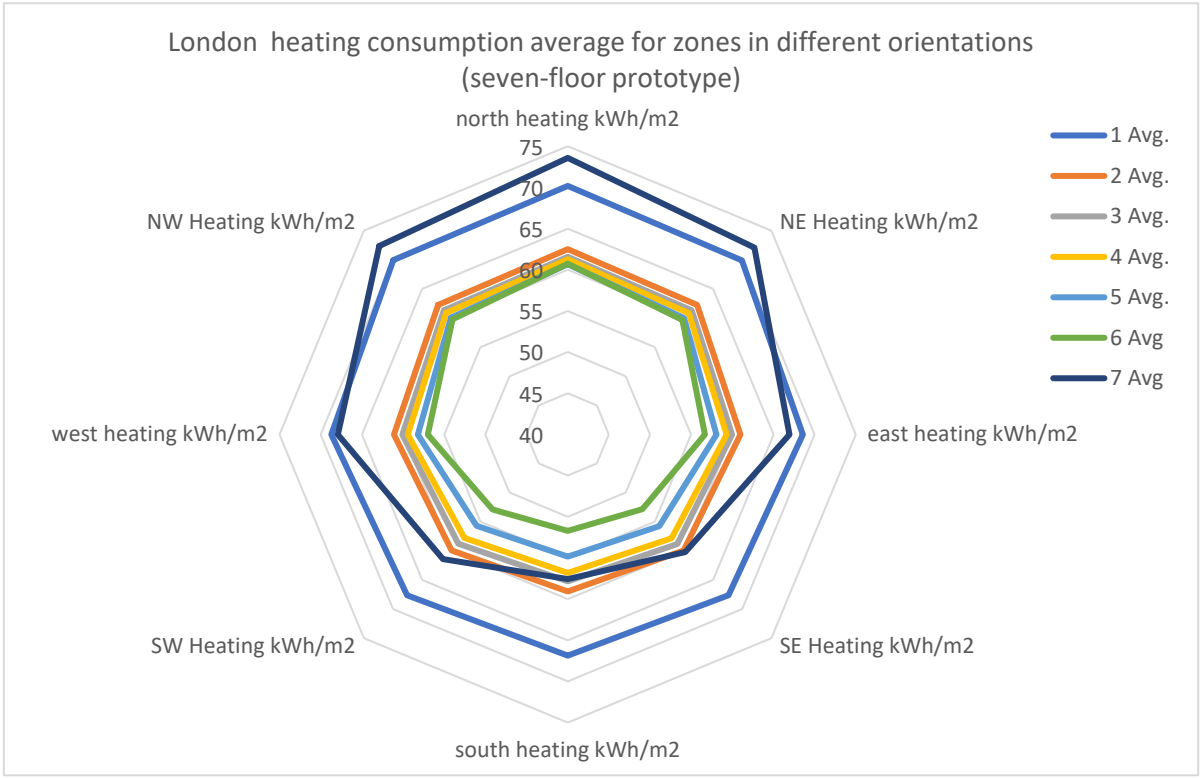


Figure 4-38 Seven-floor prototype heating consumption results average per orientation for London, UK

first and second floors for both climate conditions. This can show the importance of exposure and overshadow and control the energy demand on the zone scale.

4.3.3.8 *Summary*

This section has discussed the results of the weather files used for both heating and cooling demands accordingly. Grouping results based on the number of floors and orientation has shown a coherent illustration of performance patterns for zones at different floor levels and orientations. Although these results are averaged for each floor orientation, the impact of context variation is evident through the different results for different floor number prototypes. Another general remark on the overall prototype zone results is the repetitiveness of the consumption pattern between different floor prototypes for both cooling and heating. Even more, the value of the close results for peak zone consumption is an interesting finding. This can be related to the earlier finding of the height impact on the whole building performance. It can lead to further research on indoor environmental performance and its relation to urban context variations.

Furthermore, it is noted that overshadowing played an essential role in the heating and cooling demand for different orientations and floor levels on the zone performance level exposure and urban context. This has enabled the research to clearly understand the relationship between the geometrical variables and indoor zone cooling and heating demand, leading to enhancing the simulation models and the way they are built and controlled. These similarities in patterns and values for the demand on this scale also verified the geometrical classification importance as an approach to deal with geometrical complexity and the general impact of geometry on performance at both urban and architectural levels.

4.4 Summary

The preliminary study conducted a sensitivity analysis to understand better the effect of geometrical variables on different aspects of performance, solar radiation, energy, and, as shown in appendix A, daylighting availability. This was conducted utilising integrative parametric state-of-the-art simulation tools. In carrying out these multi-objective multi-zone simulation tests, time was one of the most significant constraints. The tools used needed more optimisation regarding the analysis in urban scale multi-objective simulation without compromising its current integrating capabilities.

The nature of the tools used in these preliminary studies also must be addressed. The first thing to note is the dynamic updating process for the tools used. This study started with Ladybug version 0.0.62 and Honeybee 0.0.59, and at the time of writing this work, the study is working on versions Ladybug version 0.0.67 and Honeybee 0.0.64. while currently, the last version is named Ladybug Tools 1.2.0, containing the whole set of packages. They have been updated multiple times during the run of this study and the following ones. Although this continuous development has enhanced the tool's performance in many aspects, it was also challenging to keep updated during the testing and analysing process.

The aim to run holistic simulation and optimisation on an urban scale at one click seems troublesome with this set of tools and their time-consuming features, not to mention the hardship of dealing with such a complex goal within this vast set of components. This made it clear that there is a gap for enhancement on the time-consuming challenge and the integration of data-driven design decisions within the early design stages. The study has shown in appendix.A. that there is a potential for sequential environmental optimisation in the early stages of urban design. It can be argued that this correlation can lead to establishing a multistage environmental optimisation framework that enables environmentally data-driven design decisions in the early stages of urban design within an acceptable time cost.

The preliminary study investigates breaking down urban geometry complexity by quantifying its impact on different environmental aspects for the tested climatic conditions. This relative importance of geometrical variables showed different impacts on both cooling and heating energy demand. The findings of this analysis show that there is a clear order of variables' impact on performance when it comes to urban geometrical iteration. As shown in Figure 4-17, this relative importance between the tested weather files was based on its primary demand type, either cooling demand or heating demand. For cooling demand in Aswan, Egypt, height variation is the least influential geometrical variable followed by the orientation's relative importance to cooling demand by a close margin of difference. WWR are the second-highest impacting variable on cooling demand for Aswan with the ASHRAE set of materials. At the same time, built area ratios can impact the cooling demand the most out of the four tested variables.

This order had changed when it came to heating demand in London, UK. The relative importance of orientation falls behind heating this time by being the least impacting feature

on heating demand for both weather files, followed by height variation impact with a more significant gap of change percentage. WWR has the most impact on heating demand for both weather files, leaving the built area ratios in second place in affecting the heating demand. This enabled quantifying the impact of variables on performance to break down the complexity of this relationship. This provided a fair understanding of the geometrical variables' connection to energy demand in urban scale modelling and simulation. This connection added a novel insight into the geometry as a series of high and low impacting variables, which can be used to classify the urban configuration models based on these variables to develop better frameworks that overcome the time cost and the complexity issues associated with urban modelling.

More work needs to be done to reduce time consumption in such multi-objective urban modelling and simulation. There is still an actual difficulty in conducting an urban multi-objective iterative simulation promptly. The correlation between solar gains and cooling consumption and the repetitive pattern of cooling results encouraged daylighting analysis to the study. This addition provides further understanding of the building performance within this context and more reliable results as it might change the optimal performance solution. Also, it will represent a straightforward quantification of the urban geometrical variables' effect on the building performance.

The discussion of zone level results added a clearer understanding of this connection with more evidence on connecting the geometrical variables to energy performance. The findings of the zone scale analysis have emphasised the role of the urban context in controlling performance, even on performance in indoor zones. As the exposure of zones was one key issue of this discussion, one other major key conclusion was the repetitiveness in performance patterns between different floor prototypes, highlighting the relative importance of the geometrical variables to performance. This illustrated the possible capabilities of geometrical classification based on variables as an approach for dismantling the complexity of urban models and enabling a timely mannered simulation to influence design decisions in the early stages of urban design in an efficient way.

Despite its exploratory nature regarding issues of time and complexity, the preliminary study offers some insight into the probability of having a comprehensive, holistic performance analysis on an urban scale. The preliminary study certainly adds to our understanding of the

relationship between geometry and performance on an urban scale. The correlations between solar radiation and energy demand and daylighting availability help expand the understanding of conducting a holistic urban performance analysis within the current limitations of time cost and encourage the investigation of a sequential approach to search for efficient frameworks that are capable of conducting this holistic analysis within the time frame of the early stages of design.

Moreover, this new understanding of the impact of different geometrical variables on energy demand is helping to implement classification principles on urban geometry and performance analysis to overcome the challenge posed by complexity.

As discussed earlier in Chapter 3, the thesis scope provides a proof of concept for applying classification to investigate the relationship between geometry and performance at the urban design level. Due to the time mentioned earlier limitation, the study will concentrate on urban solar radiation to reduce the hazard caused by this limitation. The following chapter will illustrate the flow of a classification framework that generates urban models and iterations with different controls on different scales. This is the initial stage of a framework based on classifying geometry and using its relationship with performance, solar radiation specifically, to search for faster, accurate results by applying basic machine learning principles leading to the optimisation stage to have the data-driven urban design models.

5 Urban Model Generation

5.1 Introduction

As shown in Figure 5-1, this is the first chapter discussing the initial stage of the framework. The literature has shown the different approaches to overcome the challenges of design decision-making in its early stages of urban design. One of these challenges was urban

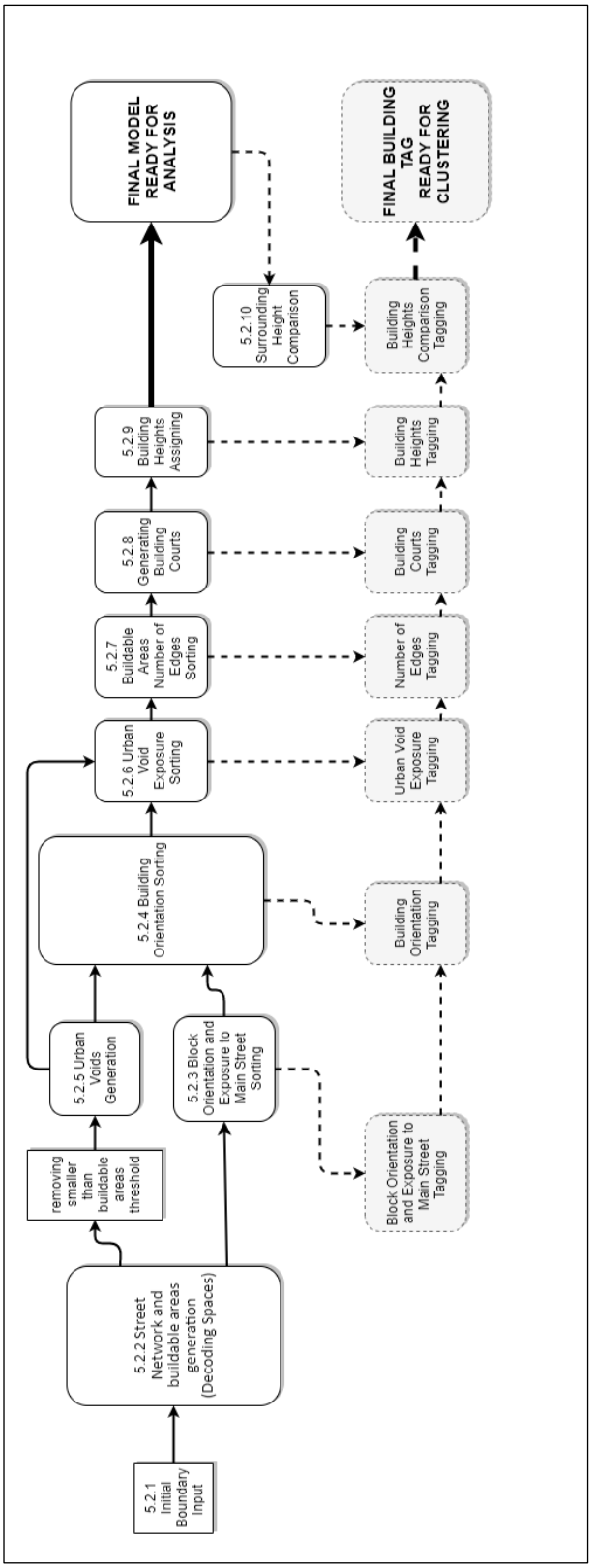


Figure 5-1 Framework flow chart explaining the flow of modelling and tag creation

complexity. Multiple studies were conducted to break down urban models into their geometrical variables to search for a clearer link between geometry and performance at an urban scale. The framework aims to respond to the challenge posed by urban complexity by classifying these geometries based on their features. This needed a generation process that accommodates the different variables while classifying and annotating them systematically. The parametric modelling platform, Grasshopper, provided the path for accomplishing this generation process based on the literature review.

This chapter discusses the shape grammar algorithm for creating a model that is parametrically generated and controlled. This allows the model to adapt to different inputs and widens the limits of geometrical variation of each stage. As discussed earlier, this algorithm is built to generate different urban iterations for the same urban contextual status, and the annotation also classifies each building in each generated urban configuration model. This means that the classification tag records the geometrical features of each building, creating a database of classification tags that can recall building data when asked for a comparison with new sets of buildings. In this framework, the data attached to the buildings will be its performance saved from simulation. This will allow for the recall of the building performance when needed, based on its similarity to a database of saved building performances. This classification process is a series of algorithmic classes working parallel to the model generation that ends up as a list of annotations or tags for the buildings' generated neighbourhood model. These different stages of parallel model generation, along with tag annotation to the model, is illustrated in Figure 5-1. This chapter will detail the model generation part of the framework, while the following chapter will demonstrate the classification tag creation and development along with the different testing phases for its capabilities.

5.2 Model generation stages

The first stage of creating the model is to generate the road network and the buildable areas within the given boundaries introduced by the user. The framework depends on a plug-in developed by a group of European universities. It is called Decoding Spaces (Koenig et al. 2017). This tool allows the generation of urban networks within a specific boundary with a certain amount of input and control. The major drawback is the limitation it imposes on the design process by limiting some geometrical aspects, and the framework had to overcome

these limitations when necessary. Although the capabilities of this tool allow generating a whole urban geometry, including buildings, the framework used it for generating the buildable areas as a start point for the needed urban geometry generation in this framework. The framework had to create a parallel text tagging system for each building to identify buildings based on their features to prepare the classification phase.

5.2.1 Initial boundary input

The framework starts with two boundary options. The first option is a typical rectangular

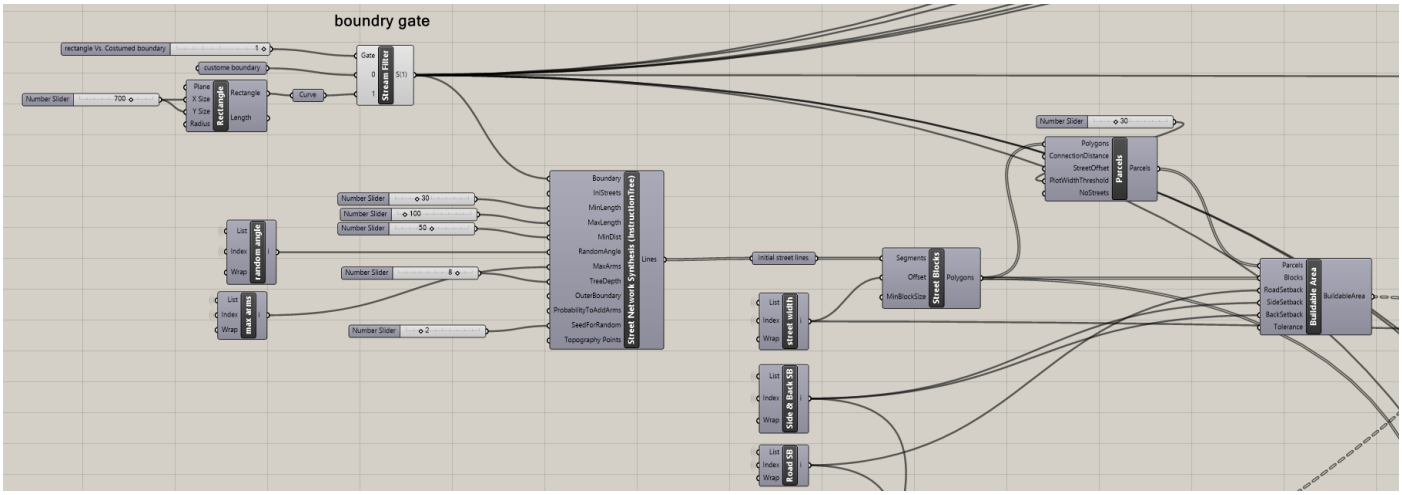


Figure 5-2 The initial input of model's boundary and the gate controling the two mentioned options. Decoding Spaces used components to generate street networks and buildable areas for the model's initial start

boundary created by the rectangle component in Grasshopper. The second option is to draw a custom boundary by the designer and transfer it from Rhino to the Grasshopper curve component. The choice is made by a gate component. This gate component's role is to allow the user to choose between the two options by moving a slider to the aimed value, allowing this option to be created as a polyline as shown in Figure 5-2. This becomes the first user-decided variation in the framework allowing limitless generations of boundaries. The following figures of the generated models show two different settings of boundaries to illustrate the applicability of this option. The database of tested geometries was built only for a rectangular boundary.

5.2.2 Generation of the street network and buildable areas

Then this boundary serves as an input to the series of four components inherited from the Decoding Spaces plug-in. The first component, "street network synthesis," generates the street network then transfers the curves to the next component, "street blocks", which

creates the blocks for the urban configuration, then to “parcels” and divides them into the needed “buildable areas”. As this tool is still in progress, not all the inputs are effective regarding the expected outputs. For example, the minimum block size does not change the output of the urban configuration. The logic of creating the network is published by the research group that illustrates creating street patterns (Koenig et al., 2013). For the following stages of the framework, it was necessary to create a different generation logic for the urban morphology. The tool is oriented towards general urban controls, not specific controls to the buildings. For example, it gives Floor Area Ratio (FAR) to control density instead of controlling the heights of buildings. These tools need some inputs defined by the user to create the neighbourhood street networks. The inputs actively used by the framework are listed as following:

- Boundary, which is a simple polyline that states where the neighbourhood is going to be located.
- Minimum and maximum lengths, which are the limits of the street length within the configuration.
- Random angle, which is the allowed angles between streets in the created junctions of the neighbourhood. This input has a high effect on the urban fabric pattern for the neighbourhood.
- Maximum arms: this input states the maximum allowed streets in a junction in the configuration, but it does not mean that all junctions have to follow this number. It acts as a maximum that does not have to be met if the algorithm could not generate such a number of streets from the same junction point.
- Seed for random: this input controls the random generation of streets every time the component receives a signal, even if it has the same values. This was one of the tool limitations as it was not as responsive at the research time as it is now in the recent releases of Decoding Spaces. This caused some challenges when building a database of configurations and buildings with their performance attached because it was hard to compare the saved data and the newly generated models with the same input due to this random effect.

- Street blocks' offset: this input mainly controls the width of the created street network.
- Parcels' plot width threshold: a threshold for the width of each parcel that the algorithm should not exceed.
- Buildable areas' setbacks: this input controls the setback from the parcel's front, side, and back.

5.2.3 Block orientation and exposure to main street sorting

The next stage is to order the geometry by the blocks' orientations. The orientation is defined by creating a vector from the boundary centre point to the centre point of each block. Then

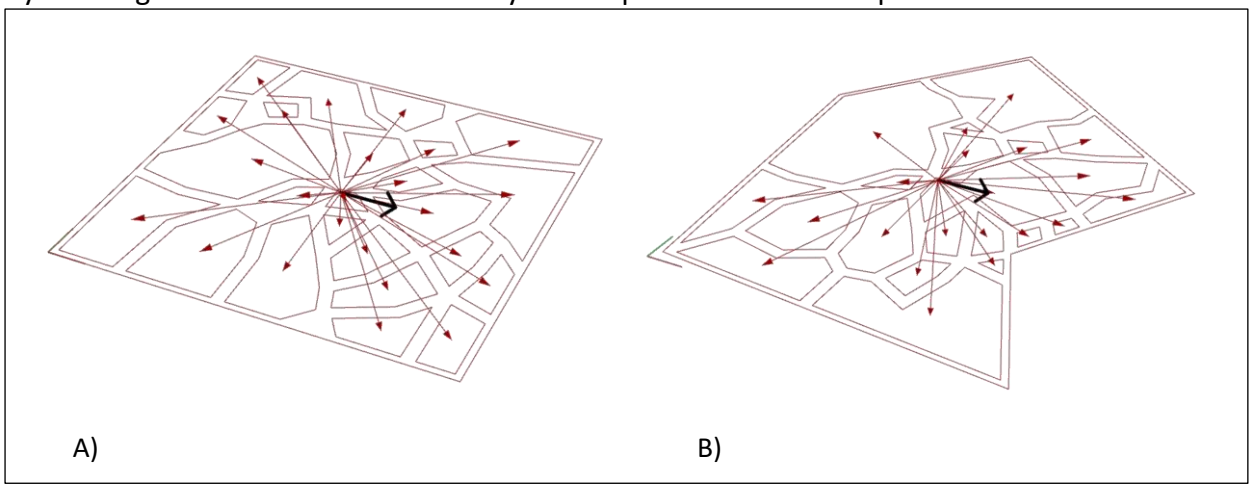


Figure 5-3 The vectors generated by the framework for each block to identify its orientation. A) the first option with typical rectangular boundary and B)a custom-shaped boundary

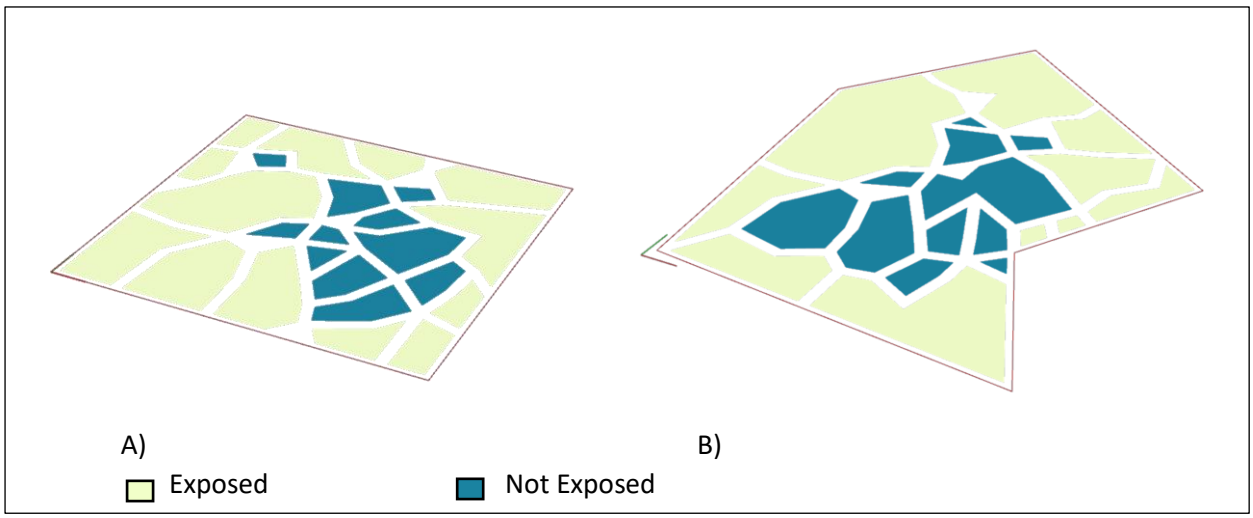


Figure 5-4 Blocks' main street exposure done by the framework sorting. A) the first option with typical rectangular boundary and B) a custom-shaped boundary

the angle difference from the vector in the direction X is measured. These angles fall into either of the eight azimuth orientations; these orientations can then be tagged to each block.

There is a limitation to the accuracy of this process due to the way of automatically calculating the geometrical centre of the boundary and assuming that it always follows the original world orientation set by the software. So, the user must note that the boundary input must follow the same orientation as the existing north in the software, especially if this framework is tried with an existing urban context.

Afterwards, blocks are sorted by their exposure to the main street. This creates two groups, one with exposure to the boundary's main street and another with no exposure to the main street. This sorting is done by scaling down the boundary curve and testing if the blocks intersect with it, then it is exposed. Otherwise, it will an inner block with no exposure to outer main streets.

The way Grasshopper deals with multiple entities is through lists of these entities. The framework reorders these lists to allow them to be tagged in the parallel tagging algorithm. It keeps this order until it changes for both processes with the following detected feature.

This will be illustrated in more detail in the following chapter. As shown in Figure 5-3 and Figure 5-4, this is an analytical phase that simply assesses the generated geometry based on the Decoding Spaces algorithm. The user controls the standard inputs needed for the components to perform their functions.

5.2.4 Building orientation sorting

The next phase is to do this orientation sorting, but this time for each buildable area. The process starts with creating a vector for each buildable area. The vectors were created based

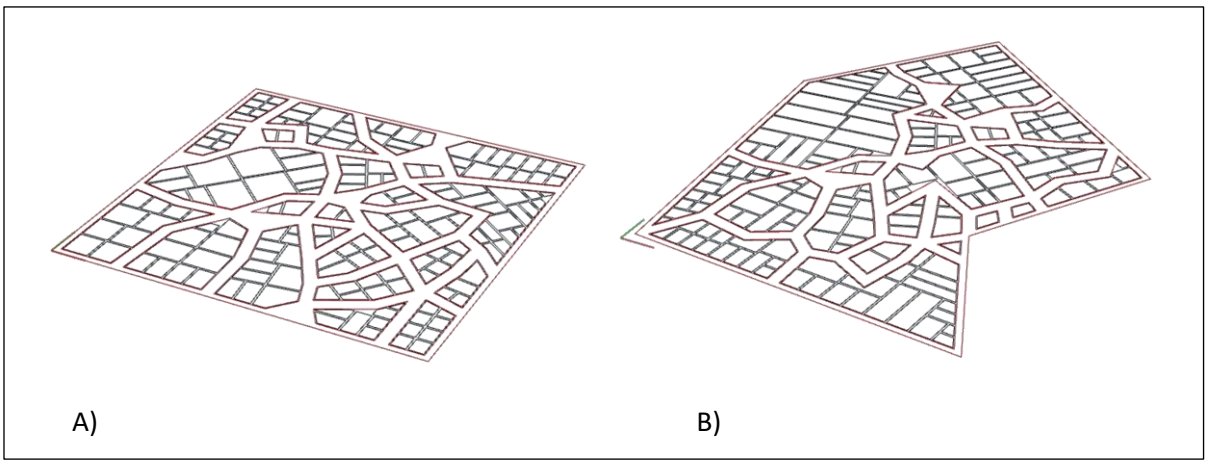


Figure 5-5 The buildable areas generated by the framework in its initial stage. A) the first option with typical rectangular boundary and B) a custom-shaped boundary.

on the nearest distance from each buildable area's centre to the generated street network. This allowed the algorithm to detect the exposed facade of each buildable area.

Then a comparison study is made for the angle of each of these vectors from vector X in the centre point of each block. This gives each building a value of its angular comparison to the x vector acting as a zero-base angle. This value is used to categorize a buildable area based on its orientation within each of its blocks. Moreover, as in the previous feature, the order of the geometry lists is reorganized based on this new setting. Classification tag lists follow the same order with the resembling value of each buildable area orientation. This is a further analytical phase where the algorithm only acts in accordance with the geometry it receives with no further generation or modelling conducted. (see Figure 5-5, Figure 5-7 and Figure 5-6)

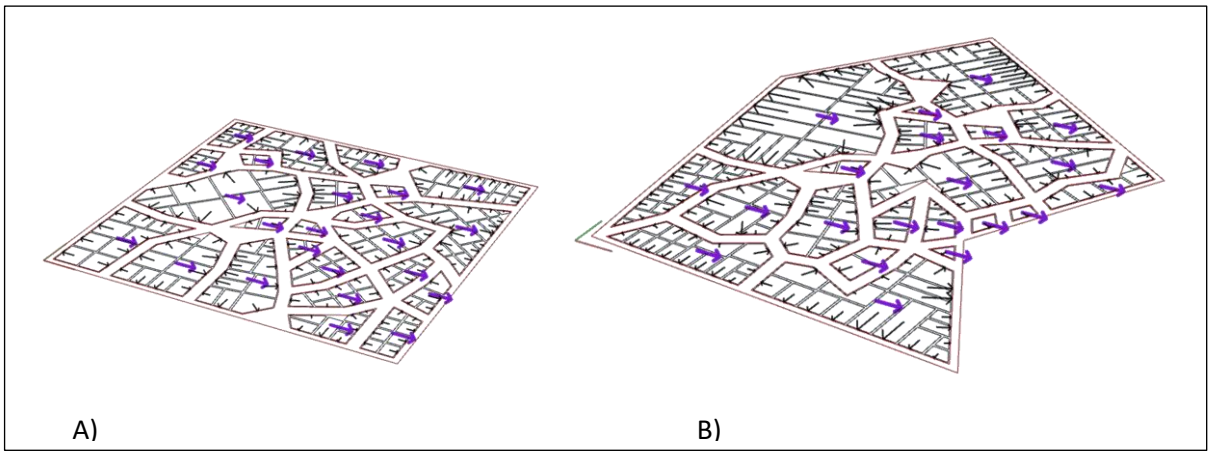


Figure 5-6 Vector X for each urban block and the buildable areas' vectors that help in sorting and tagging each building based on its location in the urban block. A) the first option with typical rectangular boundary and B) a custom-shaped boundary.

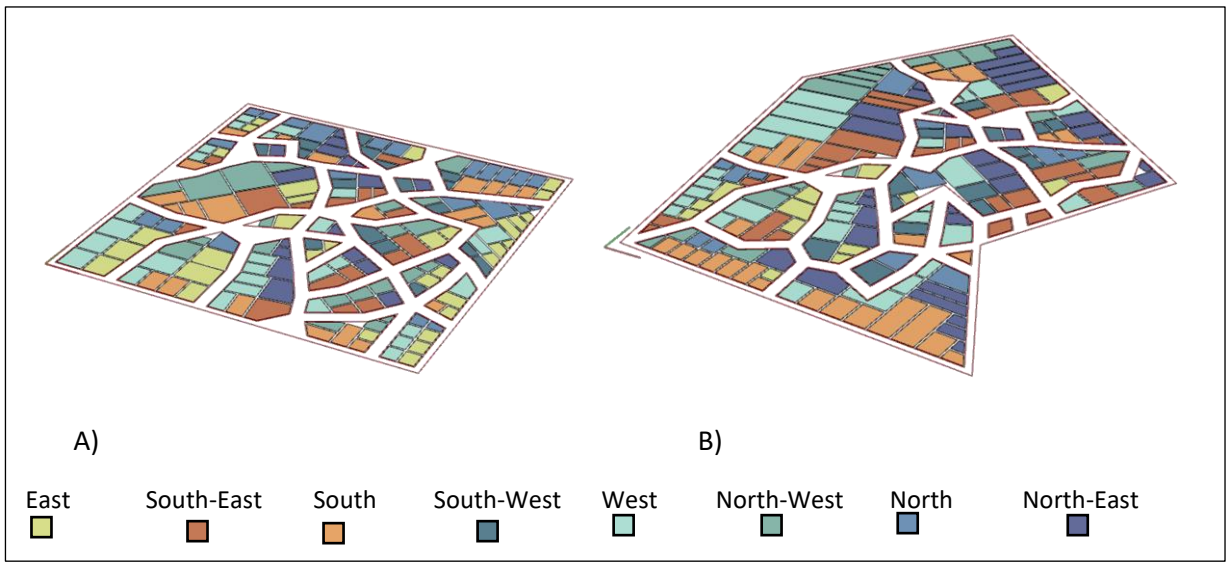


Figure 5-7 Buildable areas colour-coded based on orientation. A) the first option with typical rectangular boundary and B) a custom shape boundary.

5.2.5 Urban voids generation

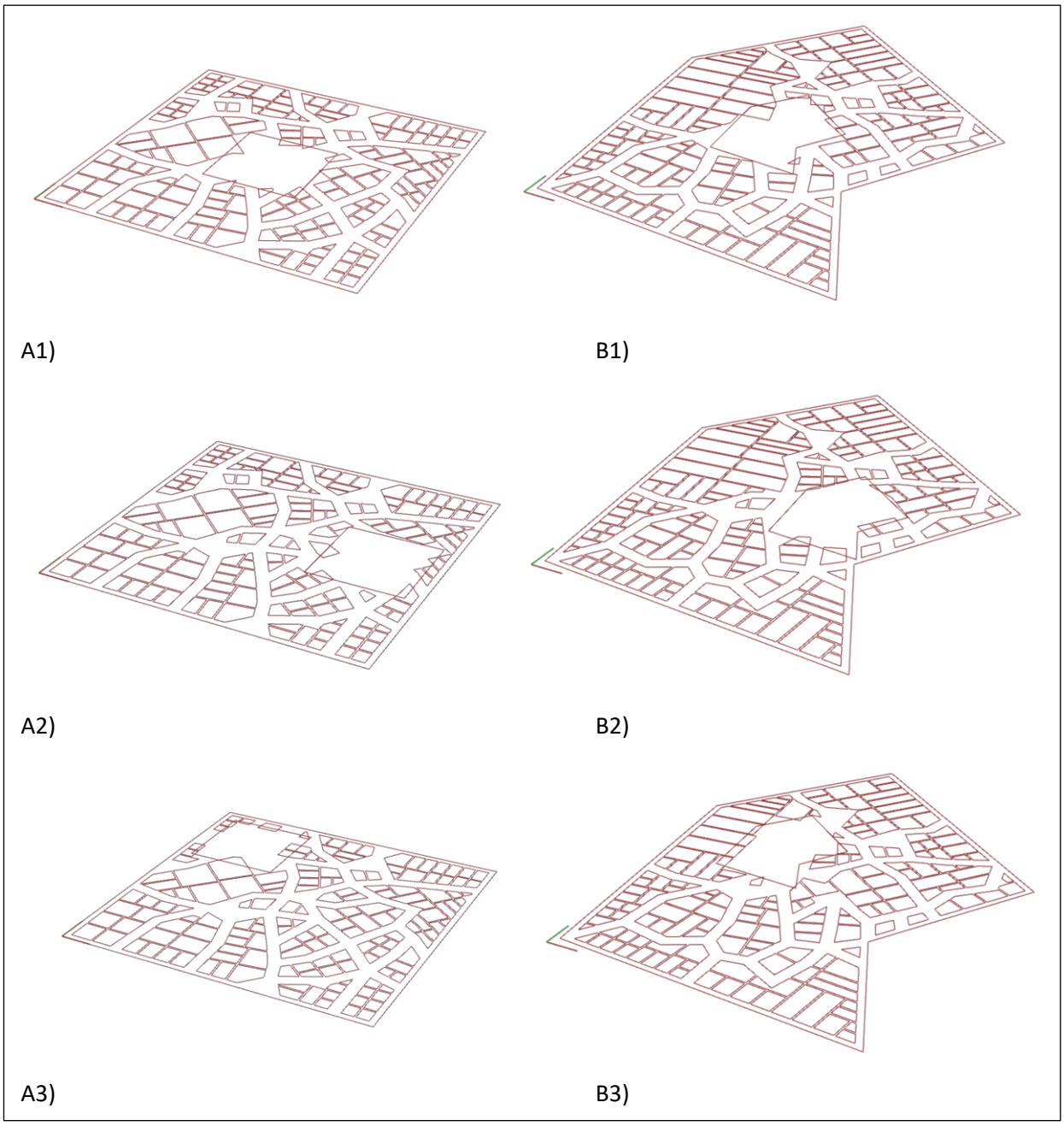


Figure 5-8 Different cases for urban voids in the rectangular boundary option, from left to right, no voids, central urban void and southern west singular urban void.

The next stage is to allow the existence of urban voids or plazas within the generated configuration. The framework allows 12 cases of urban voids to be created based on the user's decision. An urban void is determined as a scale of the outer boundary, and the user determines its percentage.

Urban voids are created by testing the location of the buildable areas to the created urban voids. This happens by comparing the centre points of the buildable areas to the

boundary/boundaries of the created urban void areas. Then, selecting the buildable areas' centre points that are located within the allocated urban void curve. Then, these selected areas are removed from the two lists of geometry and tags for the following stages. The twelve cases consist of eight cases based on the eight azimuth directions, one case for a centric urban void and another for no urban void to let the generation be as it is, and the last two cases for multiple urban voids. (see Figure 5-8)

The first multiple urban void case is randomly located based on the Grasshopper random component. The user can control the number of them and the ratio of each one from the actual allocated urban voids percentage.

For example, as shown in Figure 5-9, if the user needs to create a 30% urban void in the whole configuration and chooses to distribute this percentage equally in five multiple locations, then

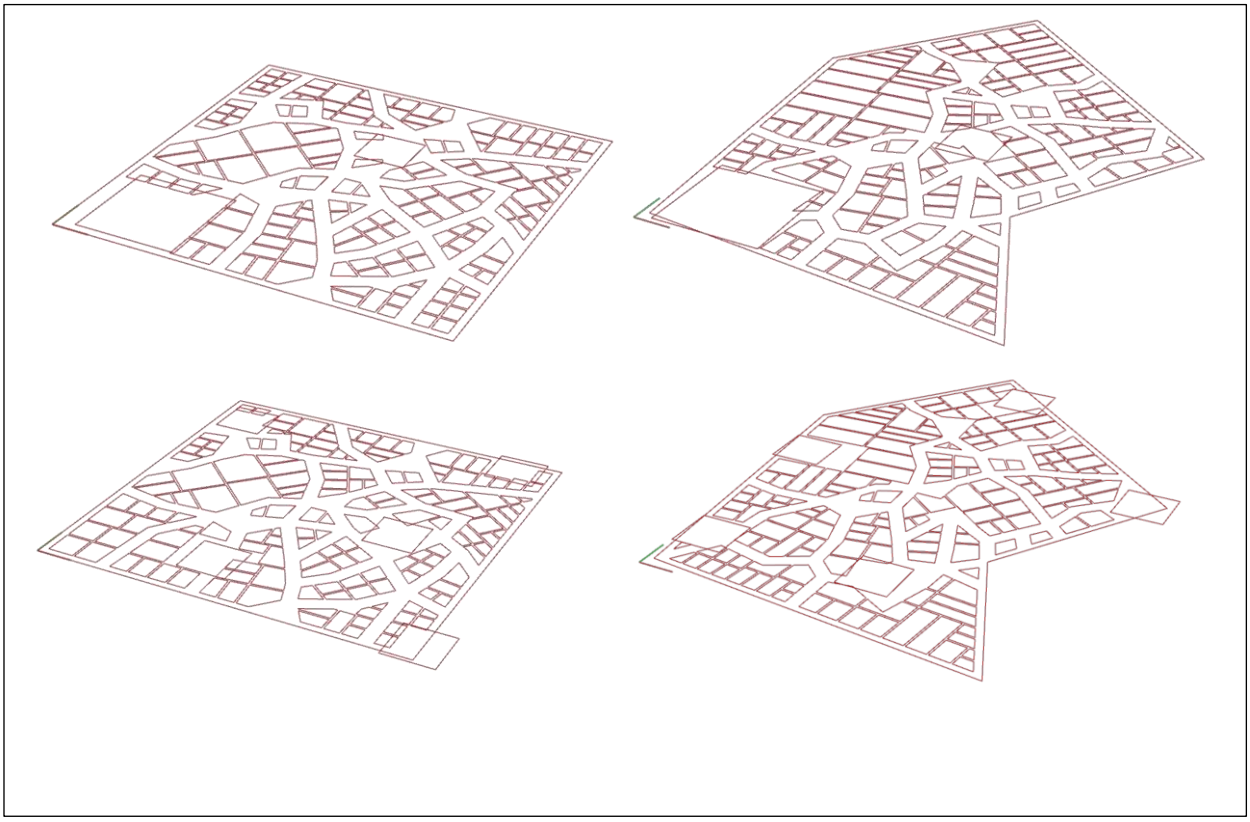


Figure 5-9 Different cases of multiple urban voids in rectangular boundary option, to the left, controlled multiple urban voids, to the right, automated multiple urban voids.

the ratio of each urban void will be 20% of the allocated 30%. Furthermore, if the choice is made to distribute it unequally, the user can allocate this percentage through some number slider controlling this distribution. If the user needs to generate a specified number of urban voids in specified locations, it can be done by drawing points in the needed locations to act as

centre points for the custom-made urban voids. This feature is mostly model generation and, as discussed, it only affects the tags list by removing the tags attached to removed buildable areas within the urban void.

5.2.6 Urban void exposure sorting

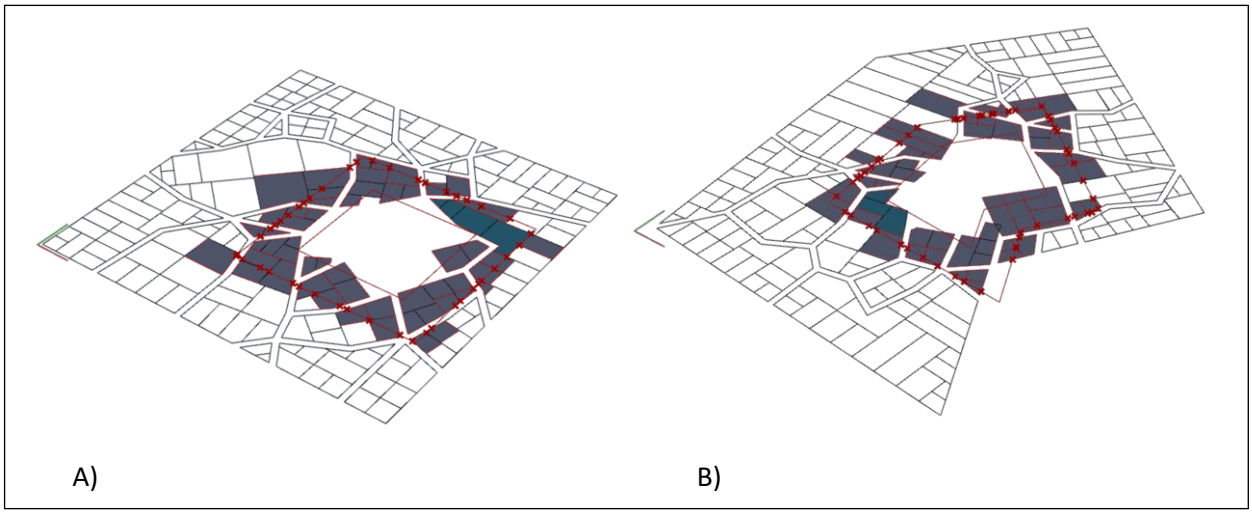


Figure 5-10 Urban void exposure clustering first trial by using urban void offset intersection to label the exposed buildable areas

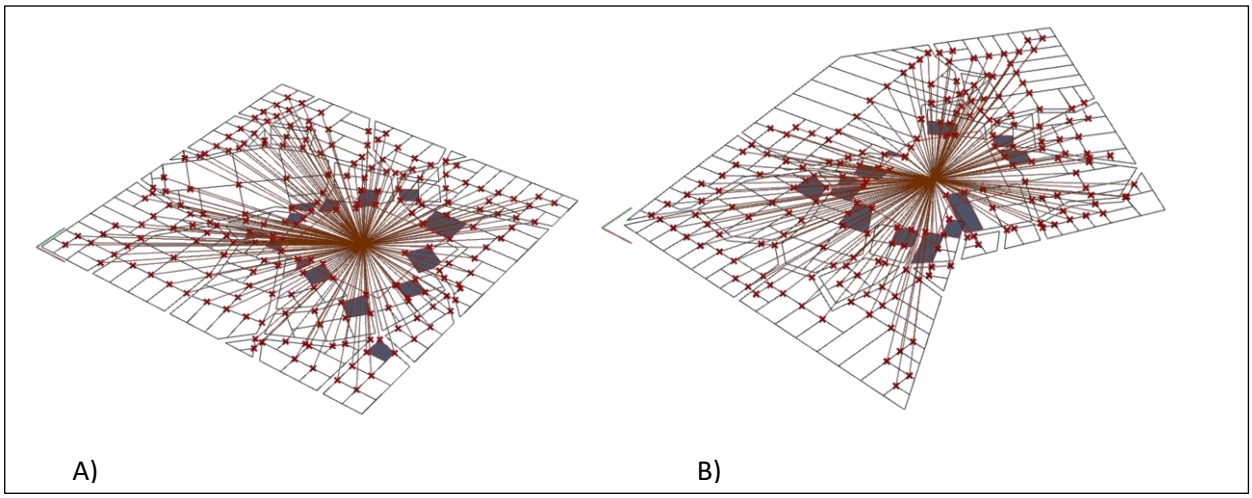


Figure 5-11 Urban void eposure clustering second trial by using lines from urban void centre to the buildable areas' centres and using lines with less than two point intersections to highlight exposed buildable areas

The next phase involves conducting the analytical part for the previous generation phase. This algorithm class aims to differentiate between the buildable areas exposed to the newly created urban voids and those without direct exposure or near to them. This is done because the solar penetration within the urban configuration affects the amount of solar radiation received by each building, consequently affecting its performance (Hosney & Lannon, 2017; Hosney et al., 2017). Therefore, it was necessary to highlight this feature when it comes to

classifying the building and relating it to its performance. This section will discuss different methods of applying this classification as an example to show the process of trial and error while building this framework's algorithm.

The first trial, Figure 5-10, was to try an offset of the created urban voids. Then, the framework tests the remaining buildable areas against their intersections with this offset. Then this new feature is added to the tagging ID for the buildable area. The main issue with

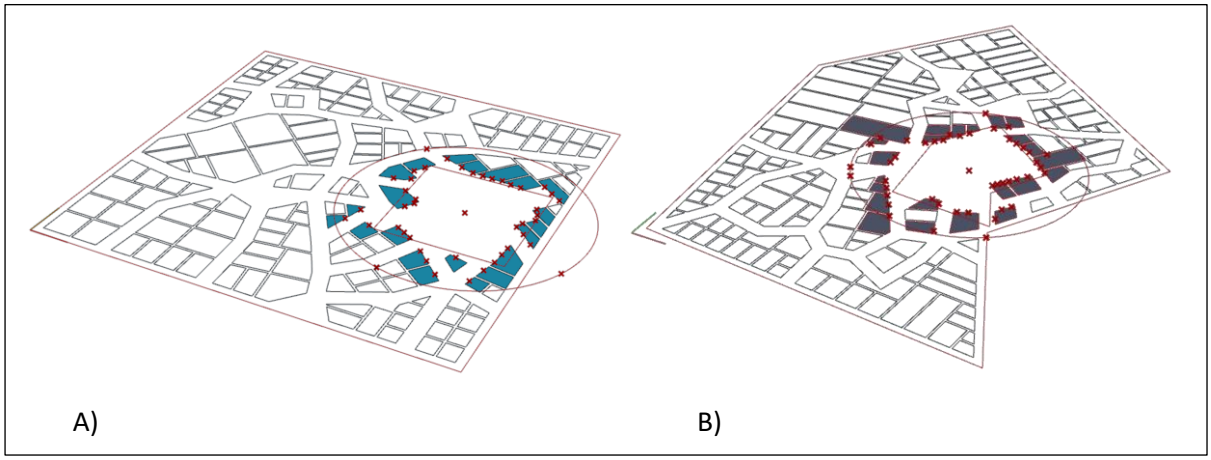


Figure 5-12 Urban void exposure clustering using Isovest component with a boundary for the urban void to highlight the urban void exposed buildable areas

this approach was the needed adjustment for the offset amount to make sure that all exposed buildable areas are included. The second trial, Figure 5-11, aimed to include all the needed buildable areas without adjustment for each iteration. The framework has created a line from the centre point of the urban void to each buildable area centre, and the lines with only a one-point intersection were chosen to create a mask for the buildable areas to highlight the exposed ones.

Nevertheless, with the nonlinear relationship between the urban void and its exposed buildable areas, the lines to some exposed buildable areas intersected deeper into the voids before reaching their centre, which also affected the accuracy. The next idea was to evaluate the centre point on all the buildable areas curves and apply the same line intersection. However, the same issue occurred and added to the inaccuracy of adding areas with no intersections to the centre as their streets are oriented towards the centre point.

The final stage, as shown in Figure 5-12, was to use Isovist. This Grasshopper component can test a set of points visible from the urban void centre point in space and with respect to the rest of the buildable areas acting as obstacles. Due to street openings within the

configuration, the framework had to add a boundary for Isovist to stop at to prevent it from including non-exposed curves because the street orientation did not have any obstacles.

5.2.7 Sorting number of edges of buildable areas

The next phase of the framework building is differentiating each buildable area based on its number of edges, which will be the number of facades this building has. This was done by creating two groups of buildable areas based on the number of segments for each buildable area. One group has four sides or fewer, and the other group has geometries with more than four sides. The reason for limiting this feature into only two groups was based on the limitations caused by the generation tool used initially to generate the buildable areas that made it rare to get a buildable area with seven sides, for example. Some of the sorted buildable areas with more than four edges have this number due to a marginal break in its area boundary polyline that causes a break and counts the line in two segments. This is a further detailing analytical process that impacts the order of the neighbourhood buildings list and its parallel classification tag list by adding this value indicator and its related tags while reordering it.

5.2.8 Generating building courts

Another detail of the building typology added through this phase of building the model is a building court. The building court is decided based upon the area of each buildable area, and

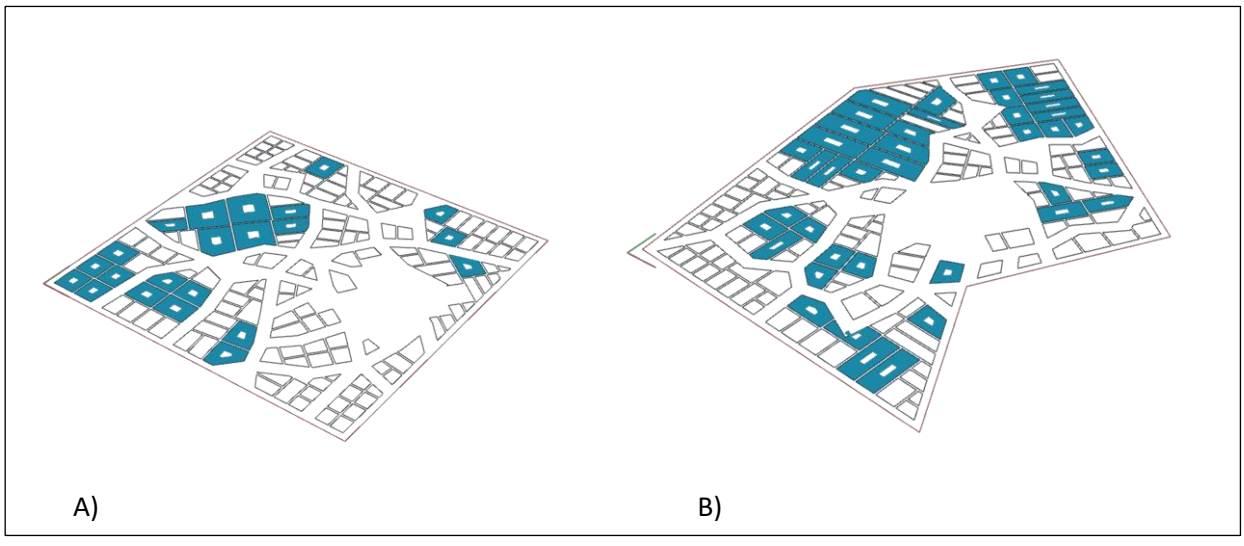


Figure 5-13 Building courts generated for buildable areas that are larger than the preset area threshold

the user can control its threshold. For this model, the area threshold is set to 2,500 sq m which means any buildable area larger than this value will have a centric court in it with a scale of 30% of its area, and this ratio is controllable. This allows the user to determine the scale of

centric courts in each building. The analytical part of this class is simple as it determines if the building has a court or not (see Figure 5-13). Then, the algorithm indicates this in the related tag for each building. The limitation of this feature is that there is no control for the user over the court location within the buildable area. Although this is considered a future enhancement of the algorithm, this limitation is beneficial to narrow down the available options and avoid the overload of time costs, while the goal is only to test the proof of concept in this stage.

5.2.9 Assignment and comparison of building heights

This algorithm class introduces heights to this generated 2D configuration. Heights had three options to be decided by the user, but initially, the algorithm will need an input of desired heights for this configuration. The heights are defined by two entries. The first data needed to be determined is the number of heights per configuration and this is acquired from the number of indices in the final buildable areas list to be matched with number of heights introduced to the configuration. The second entry is the value of these heights; these values are defined by a domain set by the user to determine the lowest and highest value needed in the designed configuration and then how many values are introduced between these two bounding values. The maximum number of heights to be introduced to the configuration is 10 values. Once the list of height values is ready to be assigned to the 2D buildable areas, the user will choose from three options of assigning these heights as shown in Figure 5-14. These three options are listed as follows:

- **Height by attractors**: in this option heights are assigned by user preference. The user must determine one or multiple points to act as an attractor of height. This simply means that the nearer the buildable area to these attractor locations it will receive a higher value in height rather than a further buildable area plot. It is important to note that in the case of multiple attractors it is an average of distances between each buildable area and the attractor points. This means the more attractors there are it will diminish the effect of this distribution.

- **Height by area**: in this case, heights are assigned based on area. This means the

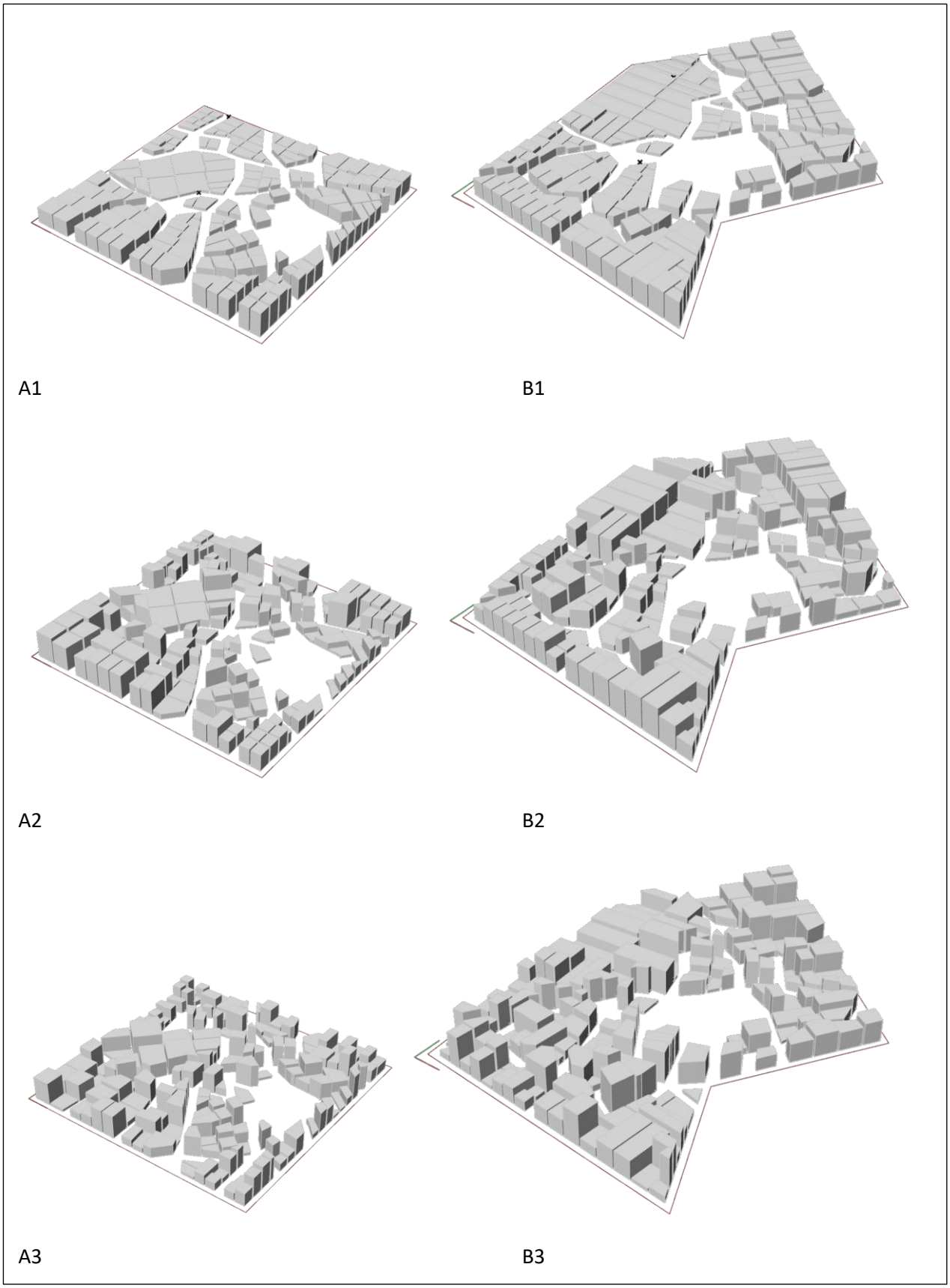


Figure 5-14 Different height option results on the two predefined urban boundaries: A1 and B1 heights by attractor points, A2 and B2 heights by area, A3 and B3 heights by random selection

smaller areas get lower heights and larger areas get the highest height value.

- **Height by random selection**: this option is mainly based on randomizing the list of heights and assigning them to the buildable areas. This is based on a component that conducts this function automatically in the Grasshopper platform.

As in previous cases, these different options are controlled by a slider feeding into a gate to decide which option is the one that will function for the following stages. This stage is the final generation class in the modelling algorithm. The direct analytical impact of it is the height of each building to be stated in the tag in its final stage. Yet, it is essential to define the context for each building, which is the following class in the classification tag creation parallel algorithm.

5.2.10 Surrounding height comparison

The last phase of identifying the model is to tag each building based on its surrounding heights. It is important to add this identification due to its effect on the amount of solar exposure. As ahown in Figure 5-15, The first step is to group every building with its surrounding buildings creating a tree of lists; each list contains the surrounding buildings of each individual building. The same sorting is done for the list of tags by branching it into a series of lists for groups of buildings.

To create these groups, the first step is to offset all the buildable areas, then test the intersected areas with these offsets and based on the status of the intersection, the groups are created. This means that when the offset intersects with an actual buildable area, it gets attached to a centric tagged one until all intersected surrounding areas are listed. Moreover, this process is repeated for all the buildable areas. The same filter of grouping the buildable areas' list is repeated to group the heights' list.

Then a comparison is made to sort out the heights to three groups of higher, lower and equal heights. Then to add more specifications, a line is drawn from the centre of the surrounding buildings to the centre of the tagged building. By the orientation comparison done before, each surrounding buildings is identified based on its orientation to the tagged building. The tag splits this information into two categories. The first one identifies the number of buildings

higher, lower and equal to the tagged building. The other one provides the orientation of each surrounding building accompanied by its height comparison result individually.

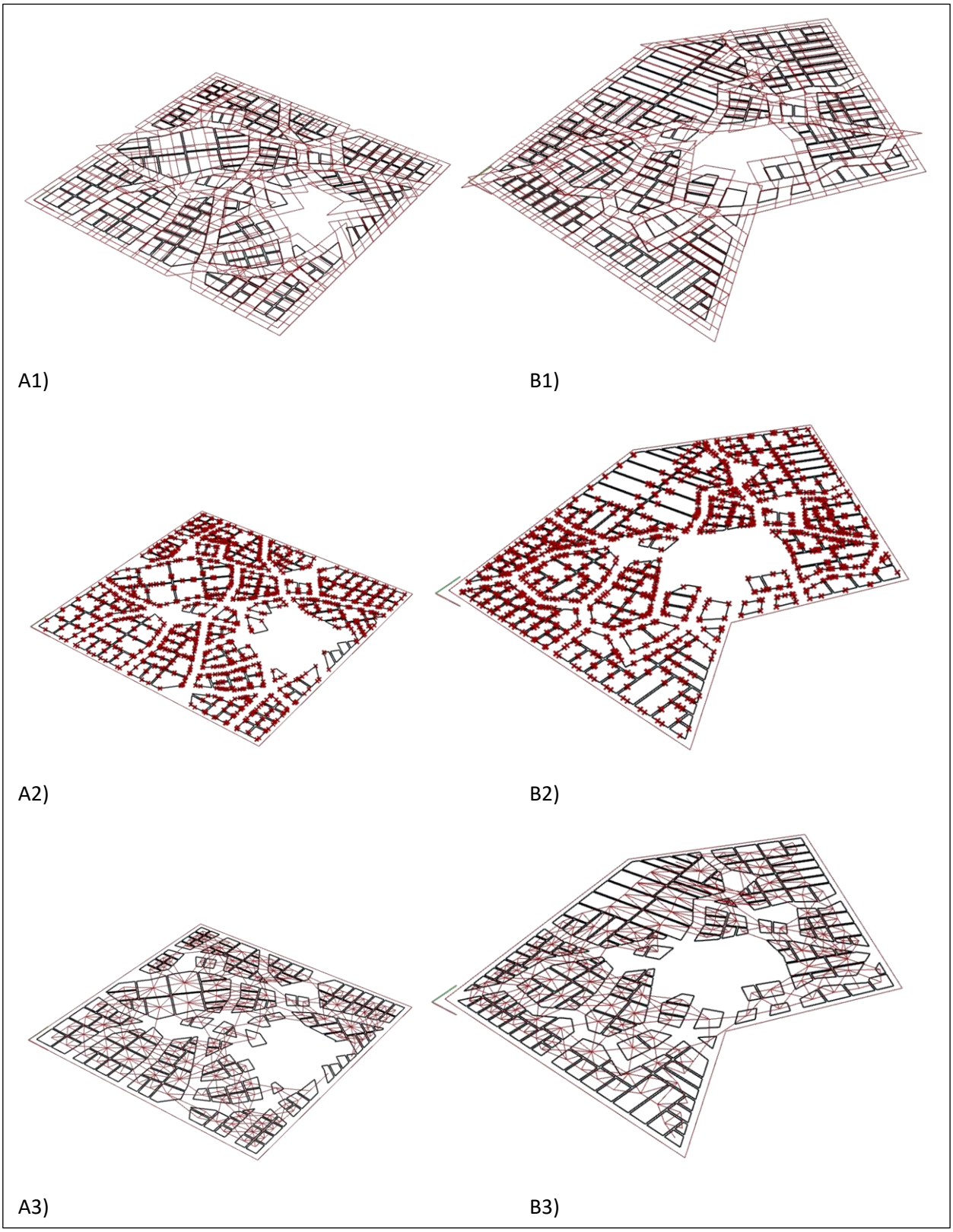


Figure 5-15 Surrounding height comparison different stages. A1 &B1, the buildable areas offsets. A2&B2, shows points of intersections between the offsets and the surrounding buildings. A3&B3,, drawing lines from each buildable area to its surrounding buildings centre points.

Another layer of this test was added in the final stage of enhancing the classification tag. In this layer, a further offset was done to identify if higher buildings at a further distance may cause inter-shadowing on the tagged building and affect its solar radiation status without being noted in the classification. The same process did this, only this time, the highest building in the surrounding buildings list was noted and added to the tag with its orientation and height comparison relationship with the tagged building.

At this stage, the model covered most of the geometrical variables of the generated urban configuration, and each building was ready for analysis and classification by unique name tags.

5.3 Summary

This chapter has illustrated different classes of the generation algorithm in the framework. These classes are responsible for generating the 3D model and its parallel phases of creating the classification tags. The main theme for creating these models is to be as dynamic and flexible as possible. This was shown in the different graphs along the chapter sections. These classes staged creating the model allowing more control over each variable contributing to the neighbourhood 3D model. Further, they provided a systematic way to classify buildings within this neighbourhood on their related urban and architectural features.

The generation classes and phases vary between being analytical and generative-based phases. It is clear that analytical phases contributed to determining buildings' geometrical features to allow for the classification process to be systematic. The generation-based phases mainly dealt with creating a shape grammar for the 3D model generated from inputs that are not restricted to a specific pool of options. This enabled the framework to handle endless model generations with minimal limitation to its geometrical capabilities. As discussed in this chapter, there is room for enabling these limitations for each class. However, due to time considerations and the scope of the thesis, these features are postponed for future development of the framework.

As shown in Table 5-1 Different modelling stages and their contribution to scale and role type, the generation algorithm generates and classifies models over the two levels' scale between urban scale and architectural typology scale. This was considered due to the findings of the preliminary study on geometrical relative importance and based on the literature review of

the geometrical impact on performance on an urban scale. The algorithm considered user control's capability to either minimize or maximize the generated pool of iterations. This has been enabled in all the generation related phases. This grants the framework the capability to adapt to user preferences and different design aims without losing the capability of classifying geometry to enhance its performance assessment time limitation.

This chapter demonstrated a clear innovative way to break down urban neighbourhood models and overcome the limitations of urban modelling complexity through this algorithmic classification shape grammar approach. The following chapter will continue discussing the resulting classification tags and the different tests and development stages occurring throughout this research scope.

Table 5-1 Different modelling stages and their contribution to scale and role type

	Analytical	Generation
Initial boundary input		urban , user input enabled
Street network & buildable areas		urban , user input enabled
Urban blocks orientation	urban	
Building orientation	typology	
Urban void generation		urban , user input enabled
Urban void exposure	urban	
Number of edges	typology	
Building courts	typology	typology, user input enabled
Height distribution		typology, user input enabled
Surrounding heights comparison	urban	

6 Classification Framework and Testing Stages

6.1 Introduction

Aiming to break down urban models into a set of buildings with geometrical variables, this framework creates a text annotation to these buildings, alongside the geometrical generation process. These annotations aim to classify the buildings based on their features and contextual status.

This chapter discusses the development of the classification tag resulting from the generation process. This is the second stage of developing the framework. Classifying a building's geometry by annotating its geometrical and contextual features has been developed and tested in multiple phases to enhance its capabilities. The benefit of classifying geometry is already embedded in the generation process due to the parametric modelling platform used in the process. The development and testing of the classification tag aimed for further advantage. The development process aimed to utilize the classification tag as a lookup database of buildings. This database consisted of two elements. First, the generated buildings' classification tags in text format. The second element was the solar radiation simulation results of these buildings as a performance indicator of these urban configurations. The testing was mainly to establish the accuracy and time saved by utilizing this database lookup compared to obtaining the solar radiation results by conventional ways of simulation.

The development of the text tag can be sorted into two phases. The first phase was the initial generation of the tags with less numerical data, having more consideration to the readability of the text tag. The following phase leaned into more numerical status of the tags to enhance their readability from the computational agency side aiming for better accuracy. The start of these tests was to determine the boundaries of the pool of iterations to build up the proof of concept case study for this innovative method.

6.2 Database settings

To test the classification process, some boundaries must be set for the sake of building this database. These settings are the variables based on which the database was set. Some of the variables were set to one value to limit the number of tested iterations, and other values had multiple options to generate various urban model iterations. As shown in Table 6-1, fixed preset variables are for both the analytical and generation processes. This means that these

variables can be changed and tested for either better results or just user preference for different studies.

Table 6-1 Model geometrical variables and the number of cases for each variable

Geometrical category	Number of cases
Urban Void Cases	13
Building Courts	2
Maximum Arms	3
Random Angles	4
Height Distribution	3
Street Width	3
Road Setback	3
Side & Back Setback	3
Total Iterations	23328

In addition to the model variable cases mentioned earlier, the framework depended on the allowed features of the initial generating tool to expand the variables pool with different geometrical conditions. As shown in Table 6-2, these variables are more related to geometrical change of the urban configuration and building typology based on what was discussed earlier in chapter 2. This included the 13 cases of urban voids discussed in section 5.2.5, the two cases of building courts illustrated in section 5.2.8 and three height distributions created by the framework shown in section 5.2.10. Furthermore, five other categories are depending on Decoding Spaces generation logic. Maximum Arms is a category for the maximum arms for the street intersections. The framework has three cases for this category of four, five and six arms for the intersection. This helps bring up a wide variation in the configuration of the street network iteration. Another feature allowed by the tool is the provision of random angles, and it controls the angle variation for the newly created streets after the intersections. The framework assigned 0, 30, 60 and 90 values for this category to reduce the number of iterations in total and gain the optimal use possible from this category.

The street width is also generated by the tool. As this framework concentrates on urban neighbourhoods, the values of 4, 8 and 12 metres offset for street width were assigned for this category. The rest of the categories are on the scale of building typologies and control

the setback as either road setback or side and back setback for each buildable area. Both of those two categories were assigned 0, 2 and 4-metre variables for each category.

Table 6-2 Fixed database variables

Variable name	Variable settings
Boundary area	Set to be 700 by 700 metres to fall within walking distance of neighbourhoods (Carret al., 2010, 2011)
Urban void percentage	30% of the total area of the boundary
Number of random urban voids	Two random urban voids were set to be 20% and 80% of the allocated urban voids area
Building court threshold	Buildings with more than 2,500 square metres were set to have a central court
Height maximum & minimum	Heights varied between 7.5 to 75 metres
Height number of values	There were 10 allocated values of heights with 7.5 metres step
Urban void exposure Isovist radius	It is equal to the radius of a circle with the same area of the urban void multiplied by 1.7
Surrounding buildings 1st test radius	It equals the total street width added to the street set back added to one metre
Surrounding buildings 2nd test radius	It equals the first test radius multiplied by 2.5

This resulted in 23328 urban configuration iterations. These iterations formed the database pool of iteration for the following tests of the classification tags and the following chapter where a neural network was tested to assist the lookup process with machine learning performance predictions. This large number of iterations has allowed different stages of testing for both the lookup code and the neural network prediction code. Also, it allows a wide range of variability for the tested buildings' tags and configurations.

6.3 Database build-up

This section discusses the creation of the database of building tags and the solar radiation of these buildings. Solar radiation simulation was conducted on two levels. The first is the building tag and performance level. The other is to simulate the performance of the urban configuration to help with time and accuracy comparison. As shown in Figure 3-2, the urban

scale simulation will be used to compare the sum of lookup and prediction results to the simulation results and understand the time aspect. Although the database lookup and machine learning prediction will be conducted on a building level, the timing of urban scale simulation will cause a challenge when some performance aspects are simulated.

6.3.1 Framework control development for urban scale iterations

Controlling the generation process is a key aspect to understand the workflow, not only for the generation and database establishment but also to learn about the framework iteration process when it comes to iterating different alternatives automatically for the optimisation stage.

The first version of the iteration controls the process for the research started as early as the preliminary study (chapter four). The need to save time consumed to run simulations one by one, especially with many iterations, led to the creation of a custom Grasshopper code to iterate through the different variables.

The preliminary study had four variables to iterate options. Grasshopper does not allow the animation of sliders all at once. It only allows for animating every slider individually. To overcome this limitation, the control system had to be controlled by one slider. Each variable had a set of options as a list. Then, a series of lists of numbers had to be created to represent each of these variable options index in the variable lists. Then all these indices are stacked using a component called Cross Reference. This component stacks and repeats the indices in their original order until they all meet the maximum number of combinatorial options available between those different variables. For example, if there are two lists, one with three indices and one with four indices, Cross Reference repeats the values of both lists to create two new lists with twelve indices each, with the same values of the two original lists but stacked. This creates all the expected numerical options from the input lists. Entwine is another Grasshopper component that is used to combine the stacked list into one tree. Then, this tree is fed to a “list Item” component that calls the same index from all the input lists simultaneously and from one slider.

Moreover, this is how to overcome the limitation of Grasshopper being unable to animate no more than one slider. The last stage of this version of the control system is to break down the

tree into four lists, but this time each list contains only one index called the “Last Item”. The component “Explode Tree” is doing this tree breakdown.

The second stage of controlling the iteration process mainly depends on a plug-in for Grasshopper called TT Tool Box (CORE studio 2017a). TT Toolbox developed an iteration package Colibri that has simplified all the steps mentioned above in one component. It also allowed the control and selection of which iterations in the pool can be run automatically for each simulation run, making this package convenient for small-scale testing before running the total number of expected iterations, either for the preliminary study or for the database creation stage.

The use of Colibri iteration components continued for the initial stage of building and testing the database. The major limitation of this was that it caused some computational conflict with the larger number of iterations and other plug-ins used in the framework. One of these difficulties was a conflict between the automation process of Colibri continuously triggering Decoding Spaces generation component, which sometimes missed the configuration’s street network, noting that the random seed input for the component was not fully robust in its

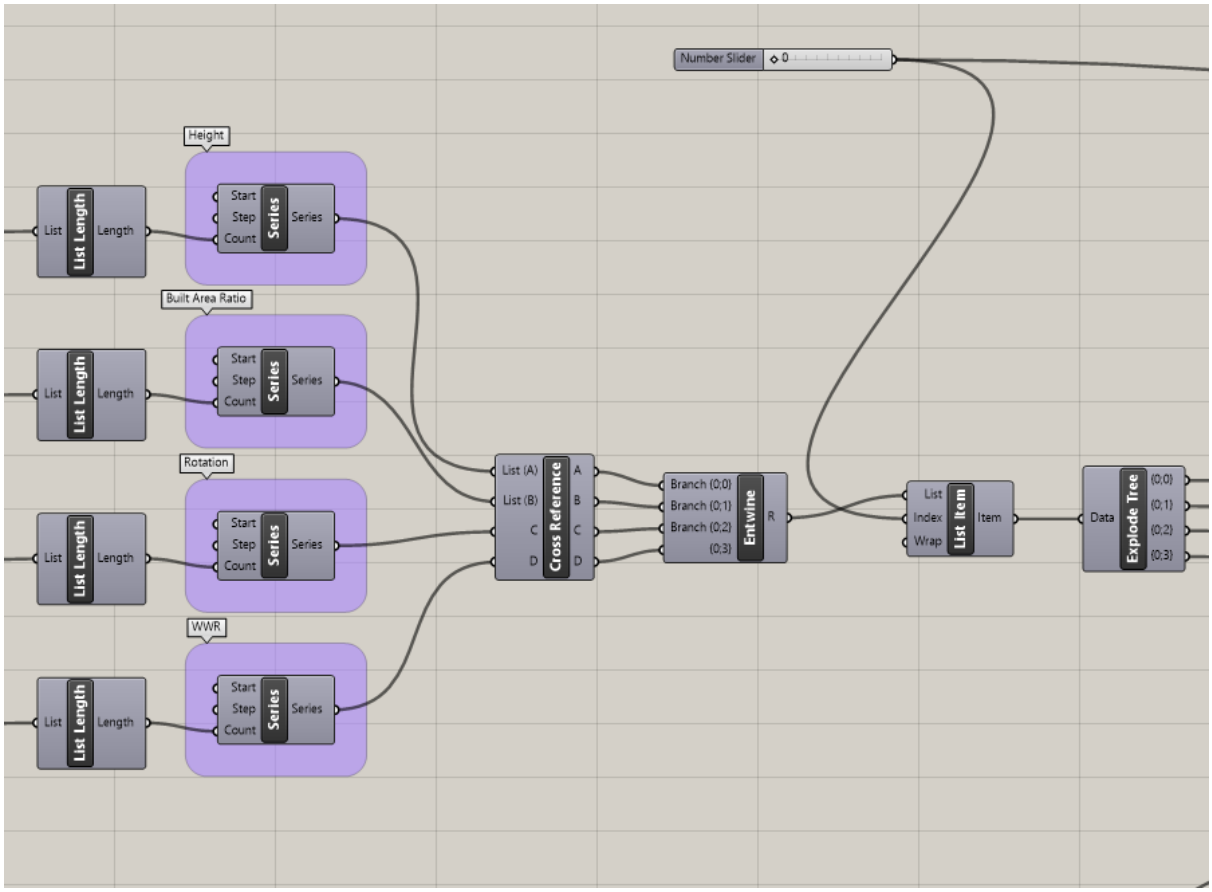


Figure 6-1 First version of controlling the iterations for the pilot studies

early stages. Moreover, as Colibri is computing the whole combinatorial options and keeps triggering the workflow, each time these triggers took their lead from different sliders, which caused some time consumption, especially with the larger scale of establishing the urban scale performance database. Another limitation of Colibri, when used for such a large-scale iteration process, is that it was not an open-source code and did not allow the process to be adjusted and controlled, especially when considering the hardship of pausing the iteration process once it became triggered(see Figure 6-1)

Colibri could not provide a way to iterate for both urban scale and individual building scale regarding this framework database creation. This was due to the change of the number of buildings at each configuration which means the number of available iterations is changing based on each configuration generated. This was a struggle as Colibri must iterate between sliders only, and it cannot receive any other sort of numeric value and iterate based on it. A specific code was created to control the iteration process on both urban and individual building scales to overcome this challenge. This was the last version of the iteration control code, and it utilized Python coding and another plug-in for controlling Grasshopper limitations called Metahopper (Heumann 2018). This control code is the one that was used to build the database for both individual buildings and urban neighbourhood configurations.

6.3.1.1 *Urban scale Iteration control definition*

```
import rhinoscriptsyntax as rs

u = range (UV)
r = range (RA)
m = range (MA)
s = range (SW)
x = range (RSB)
y = range (SBSB)
b = range (BC)
h = range (HD)

for i in u:
    for j in r:
        for k in m:
            for g in s:
                for t in x:
                    for d in y:
                        for w in b:
                            for e in h:
                                print i,j,k,g,t,d,w,e
```

Figure 6-2 The Python code used to calculate the combinatorial options

The control code for urban neighbourhood iterations mainly was based on the same logic as the first stage of the control system. A simple Python node was used only in this version to calculate the expected combinations between the input variables. Then it outputs all the combinations, each with an index that can be again controlled by one slider in Grasshopper. This method benefits of saving the calculation time as it happens once, and this does not take additional time whenever the slider triggers a new iteration to be generated. Moreover, due to its simple function as a Python code, time is saved for the computation force needed to automatically run either the simulations or predictions. The main limitation of running the simulations of urban scale solar radiation was that the aimed database was larger than the limit of animating sliders allowed by Grasshopper. The Grasshopper animation limit for sliders stops at 10,000 runs per animation. Running the whole database of 23328 urban scale solar radiation on one PC was already a time-consuming process. Therefore, distributing the database simulation run on different PCs to collect the results in a parallel manner was an efficient way to do such a large-scale simulation. This shown in Figure 6-2, and the whole input and output controls are shown in Figure 6-3 to illustrate how the slider controls the

iteration process to generate different geometries of neighbourhoods to simulate its solar radiation.

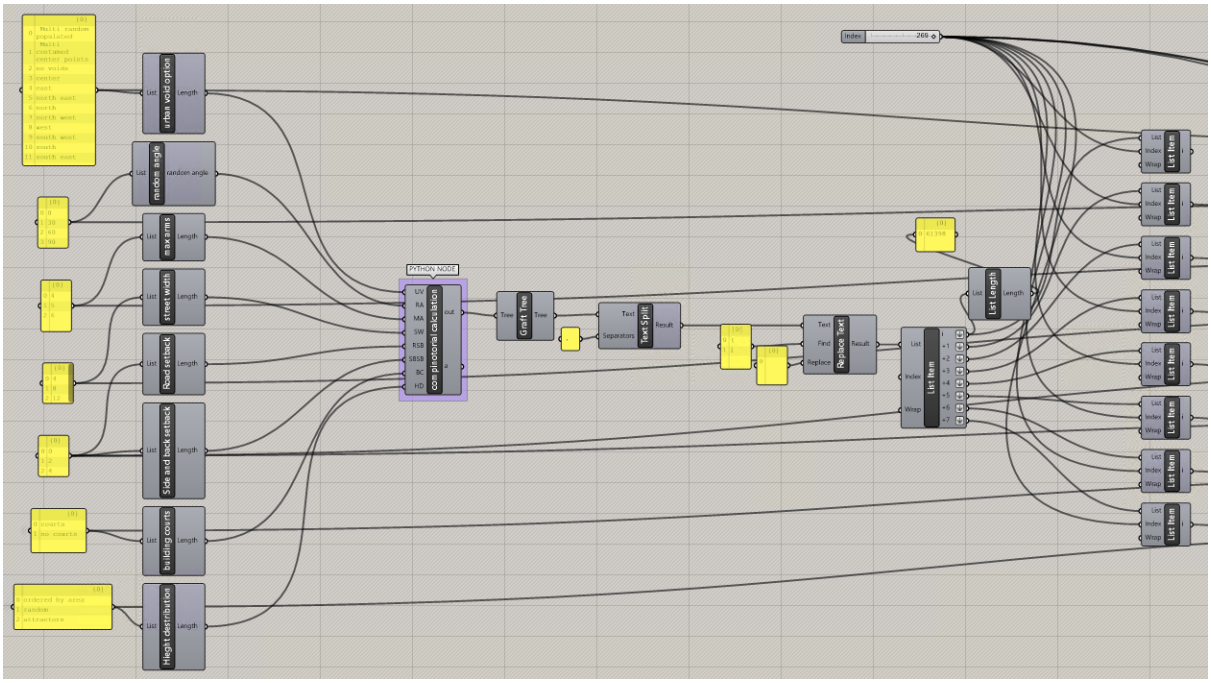


Figure 6-3 Urban scale iteration Grasshopper definition

The database was built on four PCs. The time consumed to run the solar radiation analysis of the 10,000 urban neighbourhoods was between 24 and 30 hours for each of these runs. The database took more time in total, considering the tools' challenges and some access issues to the PCs allowed for this research. Each of the computers used contained intel i7 (8 cores, 3.4 GHz) processors with 32 GB. The operating system was Windows 7 and 64-bit version.

The data recorded was based on the variables and the simulation results, and the time it took to record each entry. In this way, simulating the whole urban configuration is calculated by the difference between these entries. This was done using Grasshopper components. The time component can state the immediate time when it has an input of a text panel with “Now” as an input. This needed to be refreshed with a trigger related to the moment when the

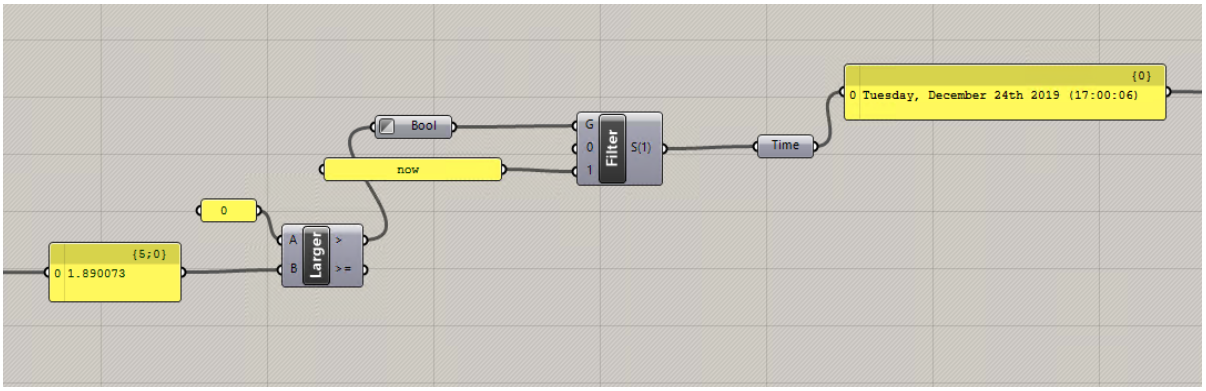


Figure 6-4 The time recording definition

simulation has ended, using the simulation result as a trigger to generate an actual value by comparing it with zero value. As long as the simulation had some results calculated, the trigger will be a “true” value that triggers a gate with a now value for both its options to make sure that no simulation gets recorded more times than it consumed due to faulty results that may occur over the vast number of iterations. In later stages, both gate entries included the time trigger to avoid accumulating the fault pausing time to the first working database log entry. Figure 6-4 shows this simple definition and how it works.

6.3.1.2 *Individual buildings Iteration control definition*

Creating the database of simulation results for individual buildings has aimed for more than just the validation of the framework results. It was collected to be used as a training dataset for the neural network node used to predict the simulation results based on that training.

The creation of these datasets had two major challenges. The first was the dual control of two sliders, one for the urban scale iteration and the other for the individual building’s iterations. As mentioned earlier, there is a limitation in the Grasshopper platform to automate the process with more than one slider. There are some open-source codes available online to solve this limitation. Some of these codes were written in Python language and could update the slider’s start and end. The one used briefly to control the iteration of the individual building was downloaded from the Grasshopper forum and Rhinoceros Discourse website (Deleuran 2018; Piacentino 2018). The main idea of this code was to break down the Grasshopper script responsible for building the slider and redefine it based on maximum and minimum inputs defined by the user. In the framework, this was used to create a slider that gets its maximum value automatically changed based on the changing number of buildings in every newly generated neighbourhood.

This solution worked for redefining the slider selecting the building to run the simulation automatically. However, once the whole process is set to the automatic iteration of neighbourhoods and running each building to set up a database of tags and the building’s solar radiation results, it failed to update the number. The failure occurred as it picks up only the first number input and iterates within the limits of the first number input as a maximum number of building indices to be selected. This means if the first generated has 100 buildings,

and the following one has 112 buildings, then the second neighbourhood will not have the last 12 buildings in it simulated.

Another trial to utilize this technique in the controlling code for iterating both urban neighbourhoods and the individual buildings to get simulated and build a database of tags and solar radiation results is to use the Python code shown earlier. However, this time calculate the combinatorial options for iterating the neighbourhood's variables and the number of buildings set to be an input to the process. This way attempted to use the maximum animation limitation mentioned earlier and aim for a smaller number of urban iterations for each run with less than 10,000 buildings in all of them combined. This new Python node was meant to be before the urban control code. It generates all the neighbourhoods' options accompanied by the building indices for each of these neighbourhoods. The slider for generating urban neighbourhoods is changing based on the change in the length of this list. This solution failed to be updated for automatic simulation runs. It is clear that it is challenging to have dynamic automatic simulation runs for a variable number of buildings, which meant it was necessary to run each neighbourhood on its own, then change the slider controlling the building selection for each generated neighbourhood which seemed to be a time-consuming task. The other way to overcome this is to find a representative average number of buildings for each group of neighbourhoods, which seemed a reasonable solution, especially with the fact that this database is used only to train the neural network and verify its prediction on the building level.

Another challenge was the fact that random seeds in Decoding Spaces are not responsive to its fixed input. This means that each time the controlling slider changes its value, it will trigger the generation node to generate a different street network. For later stages of building the database, this was amended with a better version of the plug-in, making it easier to collect the data. However, there was an issue due to the nature of the generation code being dependent on the list order. Each slider change triggers the generation node to get the same street network for this updated version, but not necessarily the same order of buildable areas generated. This does not mean that the tag will not describe the geometry correctly but only means that the order of the tags is not the same during the simulation time. This was solved by a component from the Metahopper plug-in called Freeze. This component's task was to freeze the value coming out of the slider towards generating the urban neighbourhood's

geometry until it finishes running through the individual buildings one by one. Then, it triggers only when the value of the neighbourhood generation changed to the following tested one. This required the remap of the values coming from the slider to be convenient for both triggers, one for the urban scale generation and the other for iterating between the individual buildings within these urban configurations. Moreover, that is what formed the final version of controlling the framework simulation and database building stage.

The basic concept of this technique was to redevise the decimal distance between two numeric values to the approximated number of buildings for each group of urban configurations for each simulation run. Simply, the same slider will have the value of 1.000 and keep changing its decimal value to be 8.005, 8.020, 8.100...etc using these decimal values to run individual buildings after it gets to be remapped as index numbers of these buildings in its list to be 1, 2, 3...etc. This takes place while the value of (8) is given once to the urban generation process. Then, once the slider reaches a value starting (9), it charges the urban iteration control definition to generate the following urban configuration and stands still for the next trigger.

This method provided a convenient way to conduct the simulation automatically and simultaneously, utilizing multiple PCs to collect the database for buildings tags. The database at first in the initial state aimed at 2,000 urban configurations to be simulated on an individual building level. The first database building stage was stopped at 1,200 urban configurations collection, around 180,000 building tags with their individual performance, due to the challenges mentioned earlier that caused repetitive tags and occasionally geometrically non-conforming urban configurations. This data set was used later for neural network training purposes only, but not for verification, as it was challenging to recall the exact buildings due to the difficulties. Then another run was designed and aimed for the same number of urban configurations, resulting in a total of almost 200,000 building tags. The two sets of data resemble around 5% of the total pool of iterations which acts as a reasonable resource for training within the available resources. The collection of this database was utilizing 10 PCs in the media lab of the Welsh School of Architecture, Cardiff University. The specification is the same as mentioned before for the urban scale database collecting.

6.4 Classification tag first stage

To understand the classification process, it is helpful to explain the building tags generated by the framework to explain the framework capabilities. The tag starts with initials for the block orientation, followed by its exposure to the main street status. Then, the building orientation initial is followed by the urban void exposure status. After this comes the number of edges for the buildable area next to it, whether the building has a court or not, and the built area comes after this in numbers. Furthermore, it ends with the height comparison, which starts with the height status of each surrounding building and its orientation. Between brackets, it summarises the height comparison status, how many buildings are higher, equal or lower. Figure 6-5 shows the initial stage of the tag created by the framework. This shows the initial stage of the classification tag and what gets recorded to classify the buildings. Moreover, it shows that the first tag version varied between text and numbers due to seeking more user-

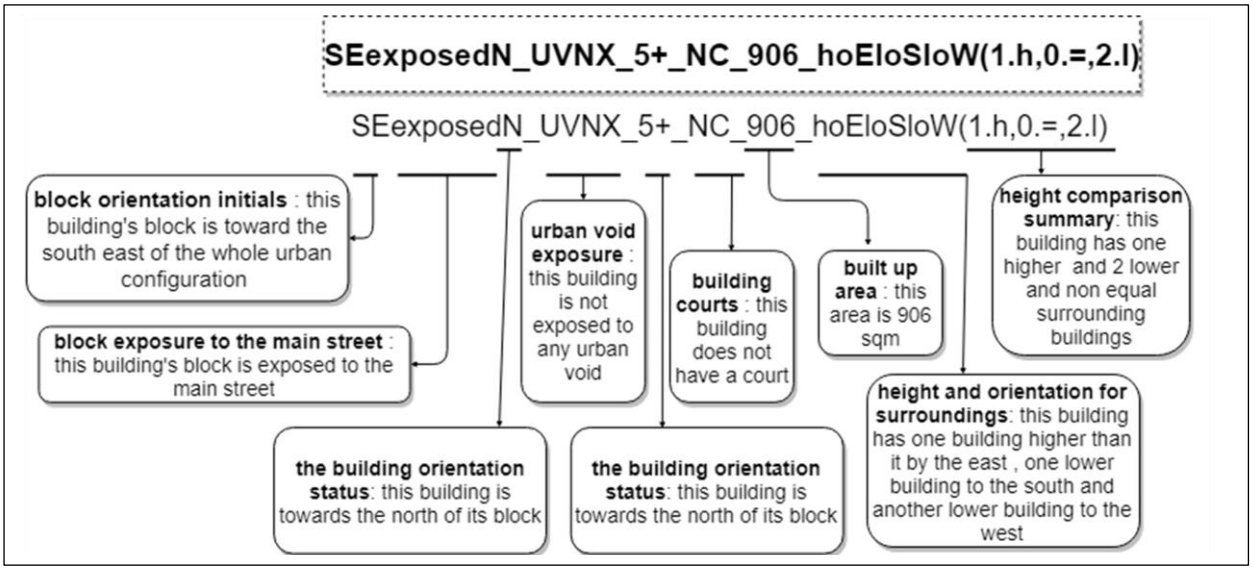


Figure 6-5 Explanation of building's text tag

readable classification tags. This was done using initials for orientations and court typologies, urban void exposure, and height comparison.

6.4.1 Initial testing accuracy for stage one

The first testing process aimed to investigate the current status of a tag regarding its capability to find similarities and detect similar buildings with the same tag. So, it was a simple trial to test if the Grasshopper code can detect buildings within the same. After generating the tags detection or lookup runs, this means testing the extent the tag needs to find most of the

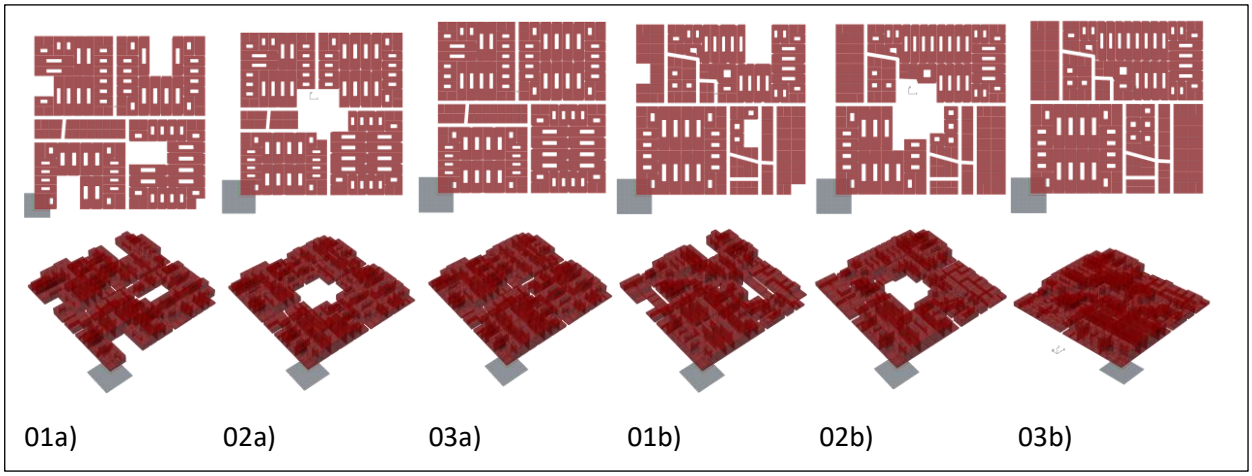


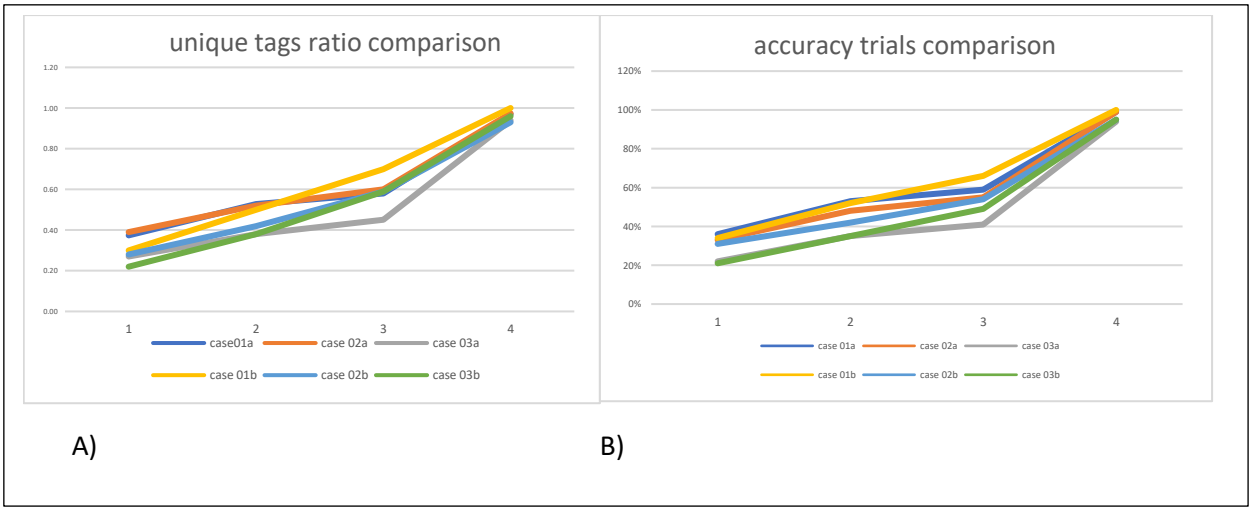
Figure 6-6 Six different cases that were tested using the described clustering technique

configuration's buildings matches. The classifying tests were done for solar radiation simulation for an urban configuration as it is one of the minor time-demanding analyses in the environmental aspects. Six iterations were assigned to do this classification to test the accuracy of the tags. As shown in Figure 6-6, those six iterations were generated from two street networks with the change of key urban void status. Then a detailed run for each building was done in all six iterations. Then the Grasshopper definition was developed to locate the distinct buildings with distinct tags. By calculating the average of those distinct tags based on their number of repetitions each, speculation can be done summing these averages to represent the whole configuration's solar radiation. This Grasshopper definition was based totally on Grasshopper "Data Set" components and how it compares different sets of data, including text and number data formats. This definition is built from two components. The first component is to set the duplicates that the configuration might have. It delivers this new set of building tags for another Grasshopper text comparison component to set the text tag comparison between the tags saved database and the newly generated tags. The tags' index is used to find its saved performance and use it to compare it with the result of direct simulation. The data in these tests are saved in excel sheet format (Microsoft Corporation 2010). It was exported to Grasshopper using Bumblebee, a plug-in for linking excel sheets to Grasshopper (Mans 2016).

Table 6-3 Case accuracy in comparison with the number of distincts in every classification trial

Case 01a		Case 02a		Case 03a		Case 01b		Case 02b		Case 03b	
Accuracy	No. of distinct	Accuracy	No. of distinct	Accuracy	No. of distinct	Accuracy	No. of distinct	Accuracy	No. of distinct	Accuracy	No. of distinct
36%	46/123	33%	45/114	22%	36/132	34%	47/148	31%	41/142	21%	34/157
53%	65/123	48%	60/114	35%	51/132	52%	74/148	42%	60/142	35%	59/157
59%	72/123	55%	69/114	41%	60/132	66%	104/148	54%	84/142	49%	92/157
99%	120/123	99%	111/114	94%	124/132	100%	148/148	95%	133/142	95%	151/157

For each case, a multistage classification has been made. The first stage was with tags including all orientation aspects for buildings and blocks added to the building court status and number of edges and the urban void exposure. The second trial added heights only to the



classification tags. Moreover, the next trail added the built-up area for each building. Finally, the comparison of the surrounding height was added for the final and optimal accuracy trial.

Figure 6-7 A) six case comparisons for distinct tags ratio B) accuracy through different trials

As shown in Table 6-3, Multi-populated urban voids have higher accuracy in all trials due to the effect of urban void exposure status on the performance and consequently on the classification proximity.

On the other hand, iterations with no voids at all have a lower accuracy level. Although removing an urban void should have improved the similarity and the accuracy of classification, it did not. Furthermore, the correlation between the unique ratios and accuracies through the different trials is positive in both aspects. The difference between the final stage and the prior stages is clearly higher. The optimal accuracy reached to more than 95%, but on the

other hand, it is not expected to save time as the unique tag ratios are also higher (see Figure 6-7)

6.4.2 Second phase of testing accuracy for stage one

Height surrounding comparison (described in Figure 5-15) was the first change added to the classification tag for the next testing stage. A larger number of buildings was saved in a small tags database for this stage of testing. This database of saved results consisted of 40,000 building tags with solar radiation results generated from around 380 urban configurations. The second stage of testing, described in Figure 6-8, was done by running a detection test on this database of building tags. This was carried out in two phases. In the first phase, 100 random configurations were selected within the previously run configurations. The second stage runs a detection test on the tags against another 100 configurations selected from the initial larger pool of configurations that did not have any saved tags.

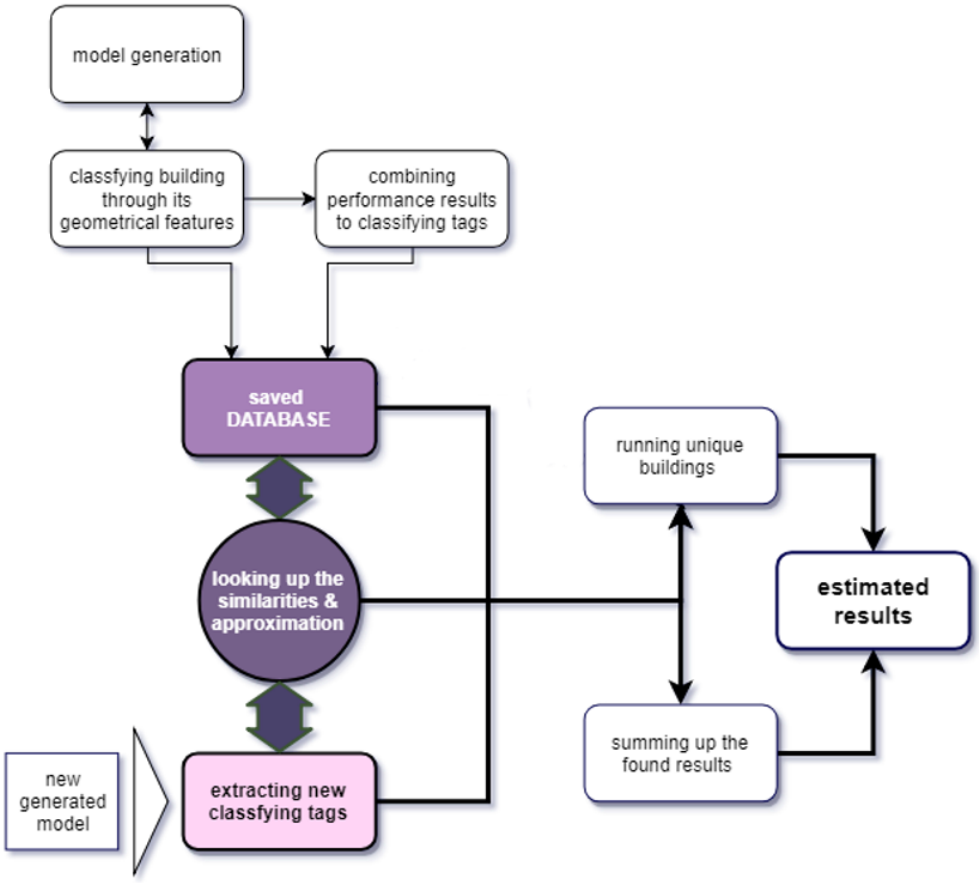
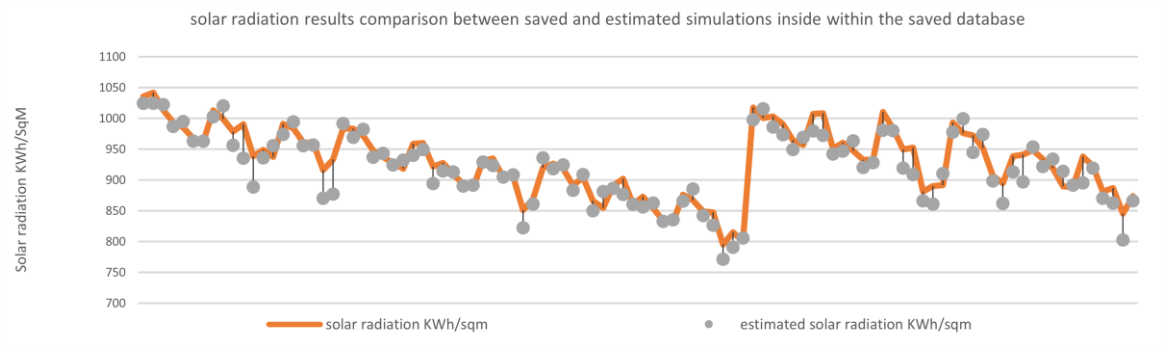
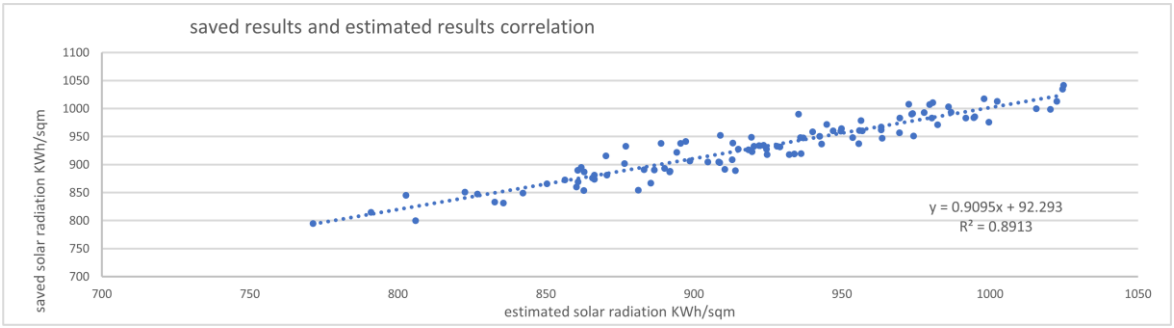


Figure 6-8 Second phase tag testing flow chart

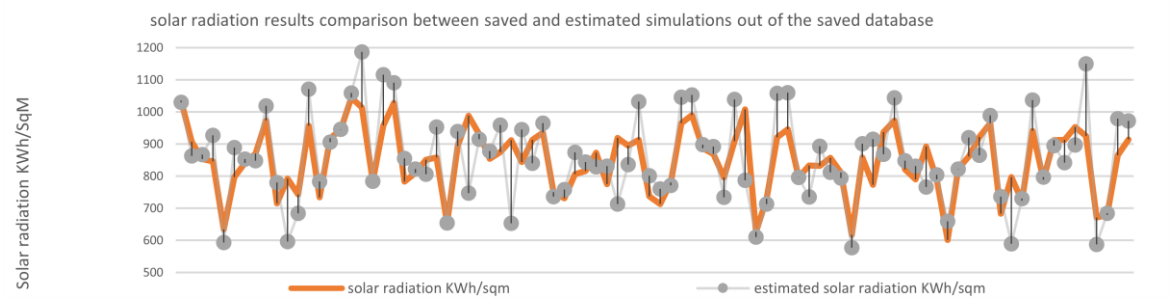
The solar radiation for urban configurations was also saved from simulations for the urban configurations without breaking them down into their constituent buildings. This allowed comparing the results from the lookup summed results with the urban configuration results to test their accuracy. The framework was set to look up similar tags in the saved database and get their solar radiation. The total result of the urban configuration is calculated by summing these looked up saved results. It is important to note that this initial detection process had some tolerance of the tag parameters. For example, it looked for the same number of surrounding buildings with the same height comparison relationship but not necessarily the same area or height if they were not available. If a building or block orientation was north-west and was not found, it can be replaced with either north or west with the same urban void and boundary street exposure conditions. It is important to note that these simulations were conducted on regular computational facilities. The number of computers dedicated for simulations varied from four to six computers over the time of conducting these runs. The computers used had the same specifications as those used for the preliminary study and previous framework testing stages (see section 6.3.1.1). All solar radiation analyses used the weather file of Aswan city in southern Egypt (24.0889° N, 32.8998° E), and it was done using Ladybug tools components for solar radiation calculations and visualizations.



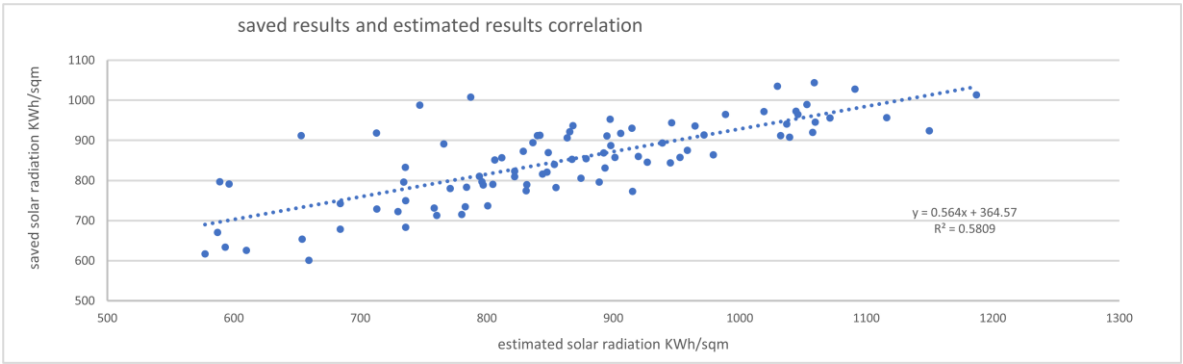
a)



b)



c)



d)

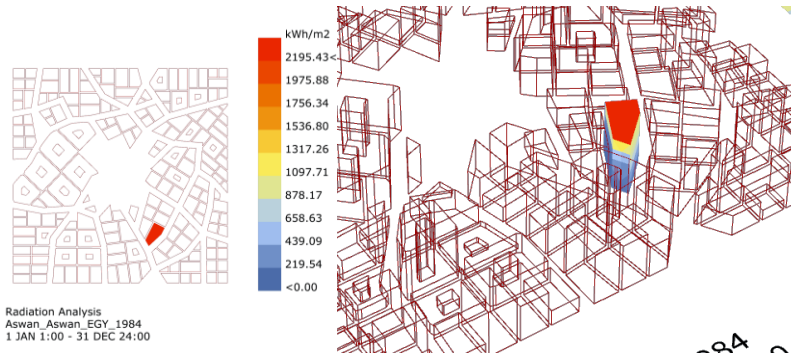
Figure 6-9 a) the graph shows the results comparison between the saved and estimated results, b) shows the correlation between saved and estimated results for the first phase of this analysis c) the graph shows the results comparison between the saved and estimate

As shown in Figure 6-9, The first phase of testing in the second stage, which was within the saved database of tags, got an average accuracy of 98% with the lowest accuracy of 93%. The time consumed was an average of five to six minutes for each configuration to get its results against double this time when the test was run in the same way for each building on its own, then summing the results to get the total performance calculation. It is important to highlight that the initial runs of the total configuration as a whole did not take more than 30 seconds for each. However, the importance of this is discussed in the following section.

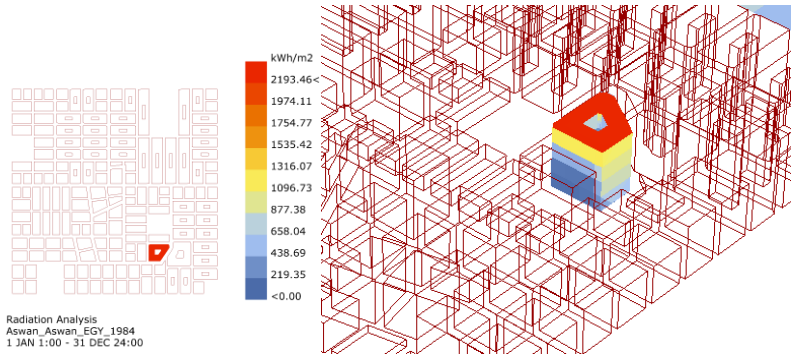
Table 6-4 A comparison for this stage of the detection process between the lookup and saved results for both the text and solar radiation results

Case number	Text tag detail	Solar radiation kWh/sqm
A1	SE_notexposed_SE_UVNX_5+_NC_2146_75.0_LoNWLoNLoNE_(0*H,0*=,3*L)	9.55
A2	SE_notexposed_SE_UVNX_5+_C_2996_60.0_LoWLoNWLoN_(0*H,0*=,3*L)	11.97
B1	SE_notexposed_W_UVX_-=4_NC_1173_7.5_HoEHoSEHoSHoSHoN_(5*H,0*=,0*L)	1.44
B2	SE_notexposed_W_UVX_-=4_NC_2030_7.5_HoEHoSHoSWHoWHoN_(5*H,0*=,0*L)	2.40

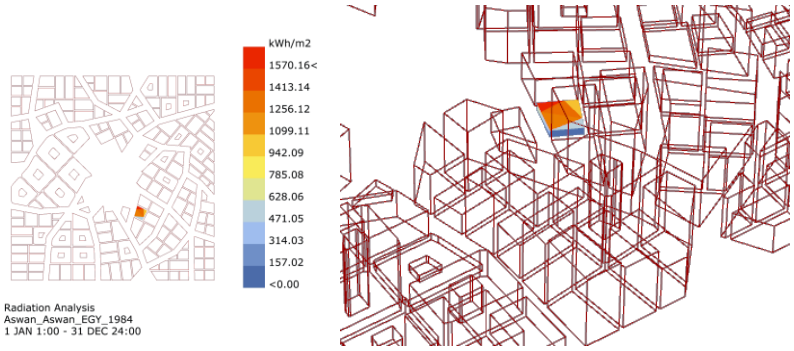
While the same time was consumed during the second phase, the accuracy dropped to around 86%, with 71% of the original model for saved solar radiation results. Also, this phase failed to look up eight out of the aimed at 100 configurations. An example of tag detection at this stage is shown in Table 6-4. It shows the low effect of numbers used to tag areas and heights against letters used to different features of the tagging text. Figure 6-10 shows a sample showcase for the detection process results. It shows the deviation between the actual results compared with the estimated results but in individual building scale. This deviation is the main reason the following stage of developing the classification tag consisted of just numerical fragments, as shown in the following section.



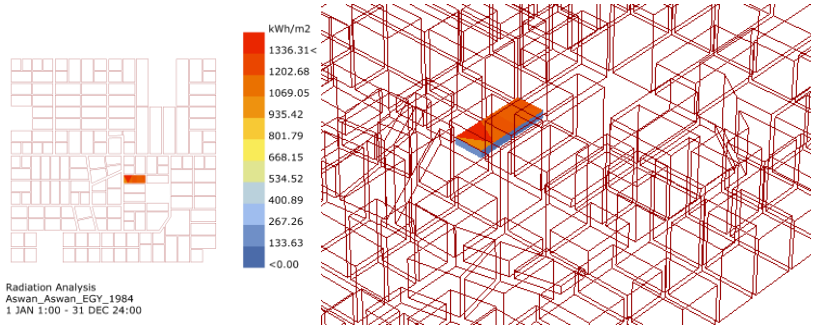
T-A1) P-A1)



T-A2) P-A2)



T-B1) P-B1).....



T-B2) P-B2)

Figure 6-10 shows a selected showcase for the detection process results. Two buildings were selected from the same configuration (A1 and B1) and the detected equivalent buildings that have the nearest tags are shown as A2 and B2. Each case has a top view (T) and perspective view (P)

During testing, the study was faced with multiple challenges. Some of these were inherited by the limitations of the tools used. Others were caused by the limited capabilities of the computational facilities used to handle such a large amount of data and environmental performance simulation. This has limited the inputs for this stage of testing to the numbers as mentioned above, which are still compelling compared to common practices for such analysis. This testing stage has shown a great deal of time-saving while accuracy did not show the same significance of the development. So, more development was necessary for the tag and its detection process to reach for better results and closer to accurate results.

6.5 Classification tag final stage:

Table 6-5 The interpretation of geometrical features in numeric values

Block orientation		Boundary street exposure		Building orientation within the configuration		Urban void exposure		Area's number of edges		Court condition		Height comparison	
0	East	0	Exposed to main street	0	East	0	Not	0	5+	0	Has	0	Higher
1	South-east			1	South-east							1	Equal
2	South			2	South								
3	Southwest			3	South-west								
4	West	1	Not	4	West	1	Exposed to main street	1	4 and less	1	Not	2	Lower
5	Northwest			5	North-west								
6	North			6	North								
7	Northeast			7	North-east								

Learning from the previous tests, this final stage of the tag contained only numerical classification data. This means it had to interpret the data from the text as numbers. The orientation aspects of the tags had to be changed, starting with east equaling a value of 00 until it ends with north-east with the value of a 07. Then the rest of the aspects had a 00 or 01 interpretation as it was just a status of whether it has the feature or not. Height and area were recorded as their original values. Then the three cases of height comparison were 00 for higher and 01 for equal height in surrounding buildings, and 02 for lower heights. Table 6-5 illustrates the details of this interpretation process. The numeric way of saving the surrounding height comparison is saved by the dedication of four digits for each surrounding

building. The first two digits state the height status as either higher, lower or equal. The other two digits report the orientation of that building. For example, if the four digits state 0204, the building is lower than the central tagged building (02), and its orientation is to the west (04). This is explained in Figure 6-11 and Figure 6-12 for the selected showcase. The same process is made for the broader radius of comparison but only for the highest building found in this radius, and it adds the height of it to this tag fragment.

The information on the tag still follows the same order as for the previous stage. The data recorded on this tag was saved to excel sheets recording each tag segment individually, making it easier to control and call it back to Grasshopper when needed.

This tag version was enhanced in its classification and annotation process to enhance the different fragments and geometrical features. Following the detection, the definition needed development to enhance its functionality in finding similarities between the saved and generated tags. This lookup process is discussed in the following section.

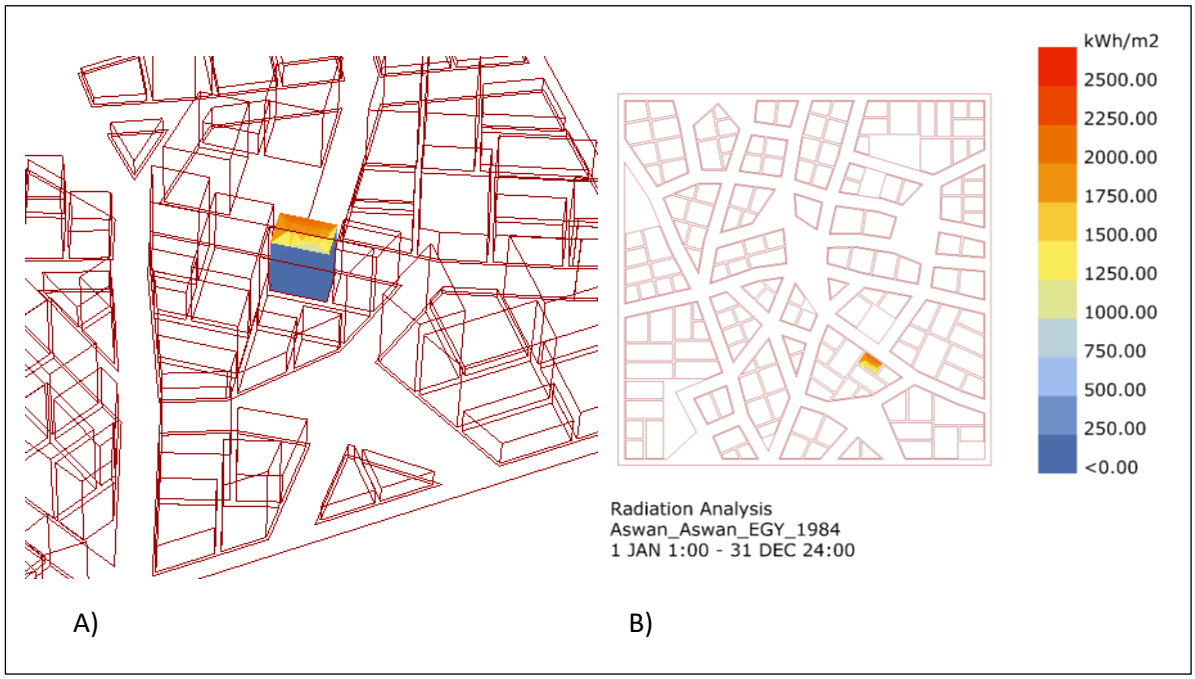


Figure 6-11 The geometrical example for the tag showcase

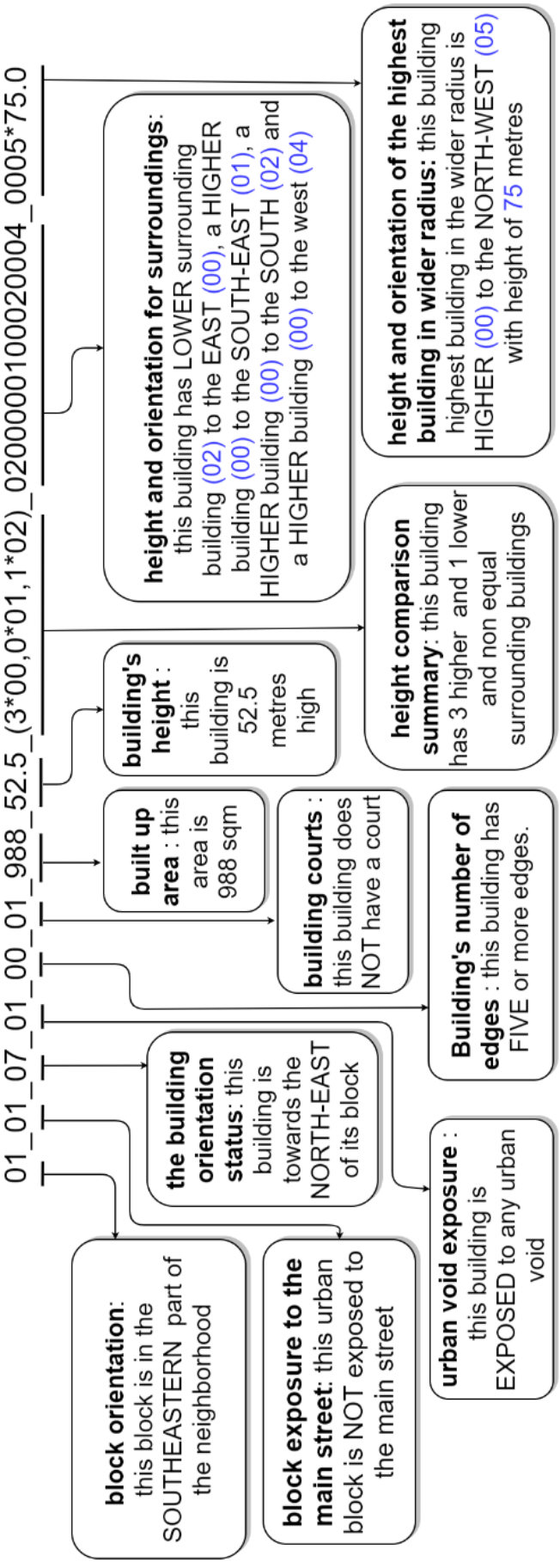


Figure 6-12 Final stage of the text tag with the surrounding height comparison explained in colour

6.6 Lookup process development

The first stage of the detection process was mainly dependent on the Grasshopper component to search for text similarities. As this version of the tag is numerical, a series of numerical equality tests were built to investigate the similarities between the saved database tags and the new neighbourhood configuration generated tags. This process was sequential, meaning each stage was dependent on the previous one. This means the database options become narrowed down at each testing stage, limiting the similar tags to their minimum until it isolates the exact duplicate of the generated tags. Also, to avoid time consumption and adding more controls to the process, this detection process was done on the whole generated tags at once. This means that the generated tags are inputted into the process as a list, not one by one.

The first stage of this detection process was to break down the tags into their fragments, allowing each stage to conduct the similarity test on one fragment at a time. Following this, the first comparison test was to find database tags that have areas equal to the ones that the generated tags have recorded. To do this, the framework used a simple equality testing component in Grasshopper. This component's role was to create a series of lists based on the number of generated tags. Each list consists of the same number of values as the database number of tags with a true or a false in each index. This true or false value is determined based on the probability if this index is for a database tag that equals the generated tag or not. Then, this process acts as a gateway over the database to allow the tags with a true value to continue to the next stage and remove the ones with a false value. (see Figure 6-13) For example, suppose the generated tags are for 100 buildings in the configuration, and the saved database consisted of 1,000 saved tags. In that case, the equality component will create 100 lists with 1000 true or false values in each of the lists in this example. This means that a generated tag will have every database tag that has an equal area to it gathered in one list at the end of this stage. For instance, if a building has an area of 250 square metres, then all the saved database tags with areas equal to 250 square metres will be gathered in one list due to this process. Moreover, this happens for all the generated tags simultaneously; thus, this results in a series of lists with available area similarities.

These lists, or as it is called in Grasshopper definitions, “Tree”, are the input for the following test, which is the same as the area test, but in this stage of the lookup, it is done for the height. The only difference in this stage is that instead of testing the equality over the whole database tags, it is only performed over the tags that passed through the first stage.

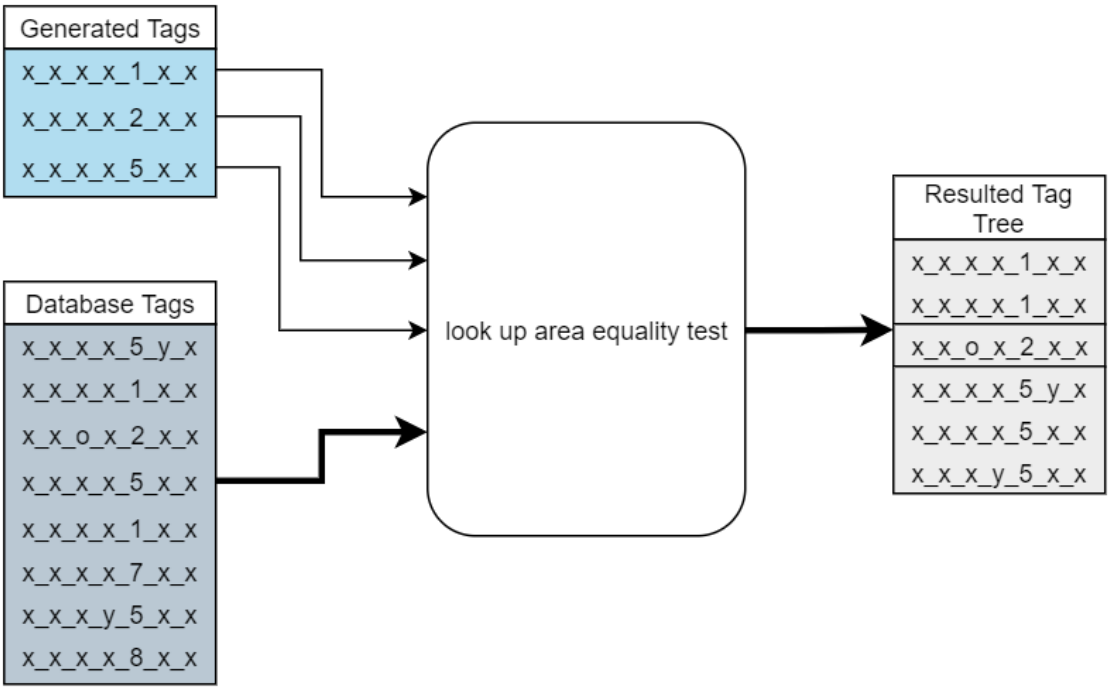


Figure 6-13 Simplified diagram of the tag similarity test

After the height test, the lookup algorithm reduces the similarity options based on each tag fragment. The order of these tests follows the tag’s recorded order to be as follows:

- Block orientation tag fragment
- Block exposure to the main street tag fragment
- Building orientation tag fragment
- Building number of edges tag fragment
- Building court status
- Building exposure to urban voids.

Then due to the different nature of the tag fragments that recorded the height comparisons, the lookup process had to follow a different way of comparison rather than the straightforward equality test.

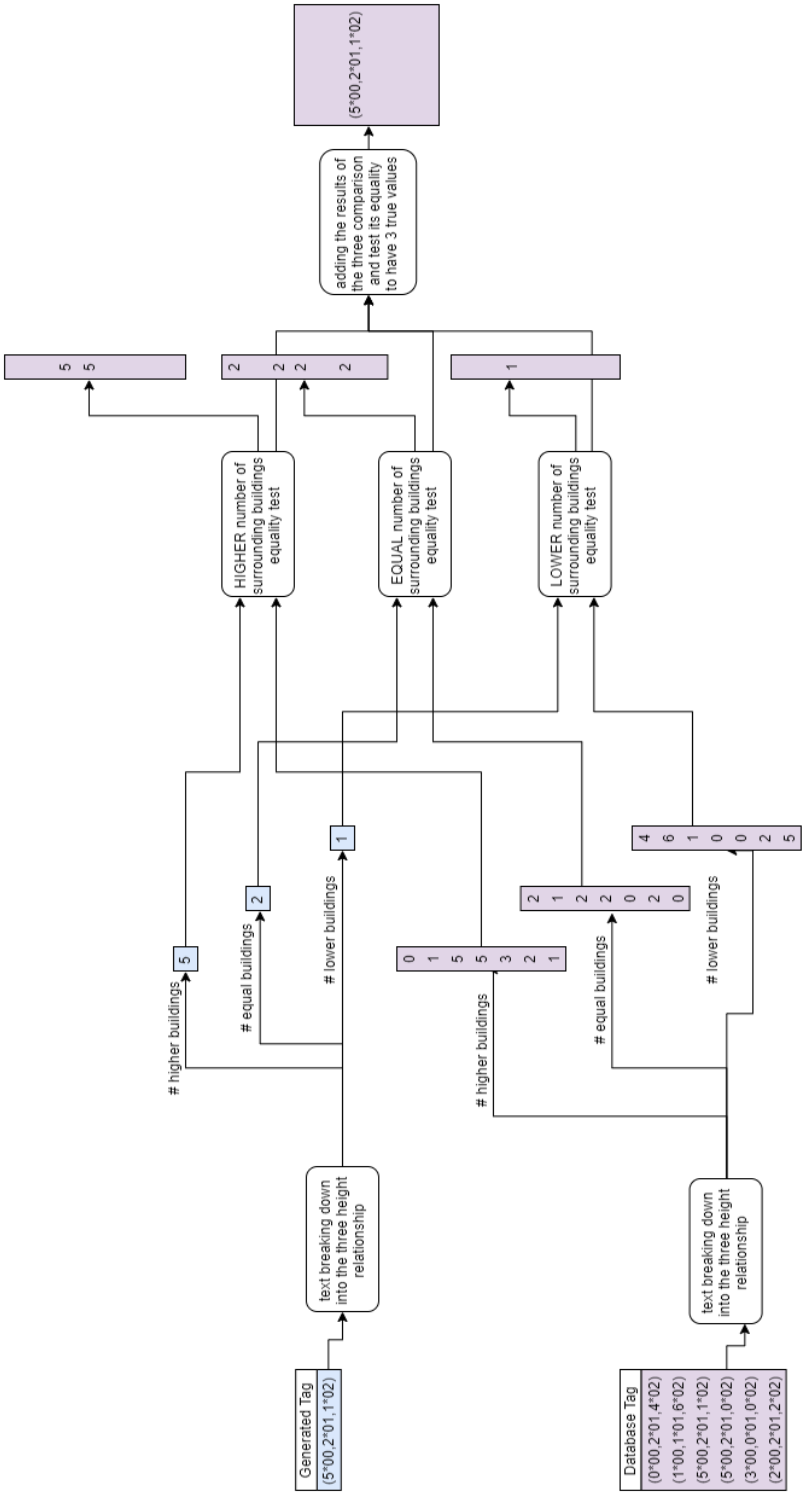


Figure 6-14 An example of surrounding building height status summary lookup similarity test

As seen in Figure 6-12, the text tag fragment that annotates the summary of the surrounding buildings is not just one number. It was hard to simplify this into one number annotation due to the nature of the data generated, recording the geometrical features. The major goal of

the lookup process is to find a way to create a mask that approves the exact similar tags and stops the non-similar tags from continuing to the next stage, as discussed in the earlier fragment lookup explanation. This was done by breaking this tag fragment into three different tests based on the summary status for higher, equal and lower surrounding buildings. So, in this case, the lookup process ensures that similar database tags annotate the same number of buildings for each relation. This was done by first breaking down the fragment into three streams, one for each height relation. After that, each database tag is tested to determine whether it has three true values from the three tests, and those which have three actual values are selected as matches and continue to the next phase. This happens for the generated tags and what remains of the database tags up until this stage. This allowed the interpretation of the surrounding buildings' height status to be quantified and helps the following phase of the lookup process, which deals with the details of the surrounding buildings' height status by adding orientation to each building. Figure 6-14shows a simplified diagram of the data flow of the lookup test for one generated tag.

The next tag fragment to be tested is the detailed height relationship and orientation of the surrounding buildings. From Figure 6-12, it can be indicated that the way of annotating this data needed to be formalized. The length of this fragment is based on the number of the surrounding buildings. This resulted in an unformalized nature to this fragment as nine buildings and others might surround some buildings might have only two surrounding buildings. This led to a difference in the length of this fragment. The process broke it down into three parts based on height relationship to the tagged building to formalise this fragment.

This led to three streams of testing like the previous one. The fragment was coded to annotate the orientation starting with east interpreted as 00 and then continues anti-clockwise recording of the orientations ending with north-east as 07. These interpretations are shown in Table 6-5. This made it hard to deal with the text as a number for the equality test, as some text annotations might start with a higher building to the east and a lower building to the south; this means the fragment will record this as 00000202. This as a number will be 202, which will cause lots of misleading results. To fix this, the numbers were reversed to ensure that all the starting zeros are at the right side of the fragment before it becomes numerically tested against the database saved fragment, which becomes broken down and reversed to meet the needs of the fragment lookup process.

The last tag fragment to be tested is the second wider area height comparison. In this stage, as shown in Figure 6-12, the fragment is only annotated for the highest building on the broader comparison area just to enhance the inter-shadowing effect on the tagged building. The fragment states the building height and orientation regarding the tagged building as a reference. The same process of breaking up and running the similarity test on each part of the fragment and combining the result is conducted for this stage. The similarity test has two parts, one for height and the other for the orientation and relationship – either it is equal or higher. Instead of using four digits to indicate the height relationship and orientation, it was reduced to two digits as the use of "zero" is useless in this case. If the building has a higher building to the southeast, it will be annotated as 0001. The first two digits will be used for the relationship description, 00 as the building is higher than the tagged building. The last two digits are for the orientation, which is 01 as it is to the southeast of the tagged building; in this case, the lookup process reduces this to be just 01.

This process has shown a multistage check for similarity to overcome the lack of lookup accuracy seen in the early stages of the text tag tests. This also gave more clarity to check the accuracy of the process and enhance it when needed, rather than depending on ready-made components that could not allow change to the process. The numerical nature of this tag made it easy to run the similarity tests and increase the lookup's accuracy. The data retrieval also had some edits as it had to be cleaned due to some tags being recorded with false values from incomplete runs. Moreover, some bugs in the generated code might cause some illegal plots to occur, like buildable areas intersecting with street boundary. Sometimes this was due to the nature of the plug-in Decoding Spaces being in its early developing stages.

Moreover, this was challenged by some limitations of the generated code of the Decoding Spaces plug-in. This caused some challenges to find similarities in the database for configurations with the same inputs. This will be discussed in further detail in the following section, along with the preparation of the database for training the neural network node.

6.7 Summary

This chapter focused on the data generation part of the framework. After discussing generating the neighbourhood geometry and its modelling process, this chapter discussed the parallel generation of classification tags and the development of the whole framework

iteration control code, database building and recalling the database and the similarity lookup. The discussion also focused on its time and the accuracy of the results.

This research is aiming at a proof of concept result for this framework. It is the reason the database had to have some geometrical limitations where it is set to be generated with variables based on the literature review. The testing was aimed at the least time-consuming performance aspect, which is solar radiation.

Building the database and automating the iteration process for later optimisation stages needed a simple bespoke definition that combined Python coding and some Grasshopper plug-ins to generate different iteration of urban neighbourhoods' geometry and select buildings for simulation and database setup. This controlling definition is not aimed only at building the database, but it is further used for iterating optimisation alternatives.

The discussion on the development of the classification tags illustrated the details of the text tags and their geometrical annotation relationships. Moreover, it showed the different testing results for the initial stages of the classification tag until it reached its final refined version that was used to build the database. The results showed clear time-saving gains while the accuracy is enhanced by the change of the nature of the tag from being a mix of text and integer indications to be solely a numeric-based tag.

The database lookup and data retrieving have been explained, likewise, in this chapter. Each segment of the tag had a different logic for conducting its similarity test based on the nature of each segment. This has led to an enhanced lookup process that does not allow for approximation of similarity to enhance the earlier accuracy of the results.

This far, the research at hand has explained the building of a database that includes classification tags and solar radiation results. This chapter has described the methods used in building this database and the enhancement of the tag features and retrieving the data results back to the framework, in addition to illustrating the different levels of the control system to iterate alternatives.

Having discussed how to construct the database of classified geometries and how to generate geometrical variables of urban neighbourhoods, the following chapter of this research addresses ways of implementing Artificial Intelligence (AI) applications on this database to

save the simulation time, maintaining a high ratio of accuracy and optimizing the performance.

7 Artificial Neural Network Application

7.1 Introduction

The aim of building this framework is to provide an innovative, efficient way to optimize urban geometry. The previous chapters have described the methods used to build an innovative iterative modelling and classification framework capable of classifying urban geometry based on its buildings geometry and attaching it to its solar radiation performance. This proved to save time in rerunning each urban geometrical alternative. It achieved considerable accuracy in its initial tests considering the limitations mentioned earlier. The last version of the classification tag was enhanced to limit the approximation and enhance the similarity lookup. This far, the lookup finds an exact match for previously saved buildings that have the same geometrical features as the ones in the simulation. This limited the framework's capabilities to open up to different geometrical options, like plots with a different set of areas and heights, for example.

As mentioned earlier, the broad definition of optimisation is to find an optimal solution for a design problem. This research looks into optimizing the urban geometry based on its environmental performance regarding the solar radiation on the neighbourhood. Another aspect of this design problem is to achieve the highest possible Floor Area Ratio (FAR) for the tested neighbourhood. This means that this version of the framework reduces the solar radiation in an urban neighbourhood in hot arid weather conditions (Aswan, Egypt) without sacrificing the FAR of the total urban configuration. The benefits and capabilities of applying Artificial Intelligence to solve problems go as far back as the work of (Turing 1948)in his famous publication “Intelligent Machinery”. Machine learning was defined simply by Turing to be “mimicking education” to allow the machine to learn and interact with definite commands. Turin’s definition did not mention the capabilities of a machine interacting with new undefined commands or problems. However, he hinted at this in earlier sections of the paper while comparing human education and problem-solving behaviour with the prospective machine behaviour.

This chapter discusses the implementation of Artificial Intelligence to enable the framework to recognise urban geometries that are not typical to the saved database and allow it to optimise urban geometries based on their solar radiation performance without sacrificing the accuracy achieved by the conventional simulation methods.

7.2 Neural network node development

The literature has illustrated the use of machine learning techniques in architectural and urban design problems. An Artificial Neural Network (ANN) is one of these techniques. It is clear from its name that it is a mathematical way of imitating how the human brain neurons work. A concise guide to ANN concepts and core models can be found in Jian's "Artificial Neural Networks: A Tutorial" (Jain et al., 1996). According to the paper, the main idea is to mimic the human neuron's act between receiving and transmitting signals based on the history of a mathematical threshold equation. The equation outputs a certain value based on its inputs, and the multiplication of these equations or nodes creates what is known as layers, and a series of layers creates the neural network structure. The paper also categorises ANNs into two main categories: feed-forward and recurrent/feedback networks. The major difference is that in recurrent networks, the node in later layers can give earlier nodes to enhance the prediction based on this feedback.

On the other hand, a feed-forward network has a simple one-way flow of data prediction, which is why the node developed for this problem depends on this method to avoid complexity. ANN has shown its potential to be used as a prediction and classifying method for different datasets in different fields (Paliwal and Kumar 2009). ANN principles have been applied in urban design and planning to predict different urban-related features such as urban growth, spatial factors, outdoor thermal comfort, etc. It was also recommended for use when compared against other methods of machine learning for these prediction studies on urban features (Azari et al., 2016; Chan & Chau, 2019; Lin et al., 2020; Lin et al., 2011; Samardžić-Petrović et al., 2017; Shafizadeh-Moghadam et al., 2017). In this chapter, ANN is tested to assess its accuracy in predicting the solar radiation performance of urban configurations based on the classified geometry of buildings. The first trial in this investigation was to use an available ready-made tool for ANN principles within the platform of Grasshopper. This tool was named LunchBox ML. ANN is one of the ML developed methods in this tool kit embedded in Grasshopper visual programming language.

7.2.1 LunchBox ANN node tests

LunchBox (PROVING GROUND, 2018) is a Grasshopper plug-in used in the first test stages of implementing the ANN application in this framework to test its efficiency to predict solar radiation results based on the training data of the classified geometry database. LunchBox

nodes for ANN request the dataset of training which has two inputs. The first needed input is “training inputs”, which is the database entries used to train the NN. For this test, the data used for training was a margin of 2,000 building tags collected from around 125 urban configurations. Another input is the new generated entry that is aimed to be predicted by the ANN. There has to be a unified boundary or maximum and minimum value for each of the database training entries for the ANN to work. This was done by remapping the database entries from 0 to 1, meaning the remapping process revalued each index of the list to fit within 0 and 1. So, if the entry was, for example, 50 in a boundary that starts with 0 and ends with 200, then the value after remapping will be 0, 25. This helps the ANN to define and relocate the new generations within a unified design space.

Another input to the LunchBox ANN node was “labels”. Then comes the “hidden neurons” input which is the number of neurons in a single layer network created by this ANN node. The node also takes a value of Sigmoid’s alpha under the name “alpha”. The mathematical role of the Sigmoid function in the neural network is to remove the threshold that triggers education in the node, regenerate itself, and allow the network to work on just enhancing its weights (Winston 2010). The Sigmoid function is not the only mathematical way to do this, yet it is one of the most frequently used (Jain et al., 1996; Winston, 2010).

“Iteration” is also a value of the number of revisions each node will have to learn to enhance its accuracy to meet the training data input with its prediction. This process is named the “backpropagation”. It is mainly concerned with calculating the difference between the predicted results and the aimed results. This difference consequently will affect the node’s weight and impact the overall result. The input, in this case, controls the maximum number of repetitions of this process to allow the NN to look for a nearer to optimal prediction with more iterations for each node to learn and enhance its prediction. The last input of the node is “random seed”, and it is not fully clear which part of the network structure this random seed will change (Jain et al., 1996).

The flow of testing at this stage was planned to use the lookup algorithm to find similar buildings with tags then use these found tags and their performance as a benchmark for accuracy. This allowed the lookup process to be tested against newly generated urban configurations. This process can be considered a part of the continuous testing and development of the lookup process. The lookup limitations highlighted in section 6.6 are

discovered during this testing phase, especially for the building iteration control algorithm. The lookup process in this stage mainly dealt with simple tag fragments as this was an early stage of testing the ANN application within the framework. The tag fragments used for the similarity test were as follows:

- Block orientation tag fragment
- Block exposure to the main street tag fragment
- Building orientation tag fragment
- Building number of edges tag fragment
- Building court status
- Building exposure to urban voids
- Building height
- Building buildable area

These simple fragments consist of one numeric value each, which allowed the algorithm to find approximate tags that match the generated tags for these features.

The initial use of the lookup showed that it found similar tags. Thus, for the first stage of testing, the lookup was skipped while testing the neural network on its own to have a clear understanding of its impact.

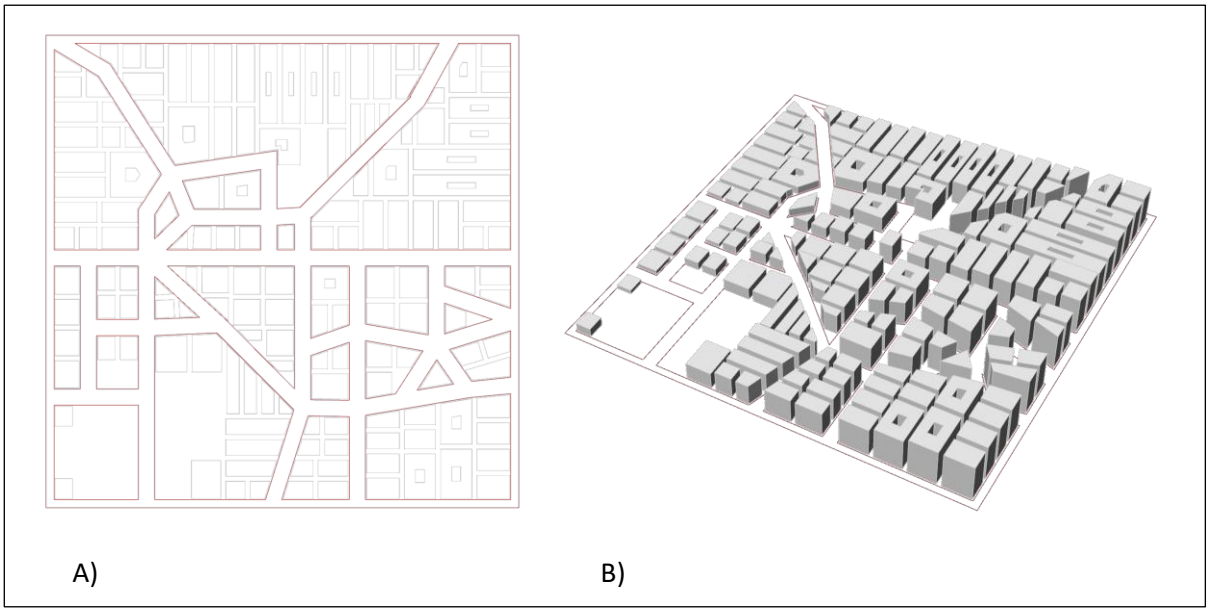


Figure 7-1 The tested urban configuration showcase A) Top view & B) perspective view

The configuration tested for this ANN application consisted of 152 buildings. It had two urban voids, one with 25% of the configuration area to the southwest and another void to the north with a 6% area. The street width in this configuration was 12 metres. The random angle allowed for the junctions was set to 0, and the maximum arms set to six arms. This has caused this semi-fragmented street network shown in Figure 7-1. The height distribution is based on two attractor points to the north-west and south-west of the configuration causing the higher buildings to be on the east side of the configuration. Each of the buildings was selected on its own and had a simulation running while the rest of the list of buildings was fed to the simulation as context. The results of these simulations are the results compared to the predictions of this stage of ANN application testing.

7.2.1.1 *LunchBox testing results*

To test its applicability, a quick test with a single ANN layer was structured to test its capability of prediction and the time it consumes to get the results for a range of 10 urban configurations.

The ANN layer consisted of five neural nodes, and the test was conducted on another PC with ten nodes for a single layer.

Time was a key element in structuring this test. The time consumed for the ten-node layer was between two to six minutes to predict one tag, which led to a whole day needed to predict only one configuration. The five-node ANN took only 20 minutes to predict a configuration with almost 150 buildings in it. Almost the same time was consumed when the whole list of tags was fed into the node. This raised a question about editing the code to avoid consuming time to repeat the training time and enable the prediction time to be faster and more dynamic if applied to different generated entries. It is important to note that the neural network setting was to only ten iterations, a minimal number to allow the learning process to occur within the different database inputs.

Considering this time limitation, the testing focused on predicting urban configuration to test its accuracy. The testing was focused only on the five neurons in the hidden layer. Then, the accuracy was calculated by comparing the database results for individual tags with the ANN output prediction for each tag based on the eight tag fragments mentioned earlier.

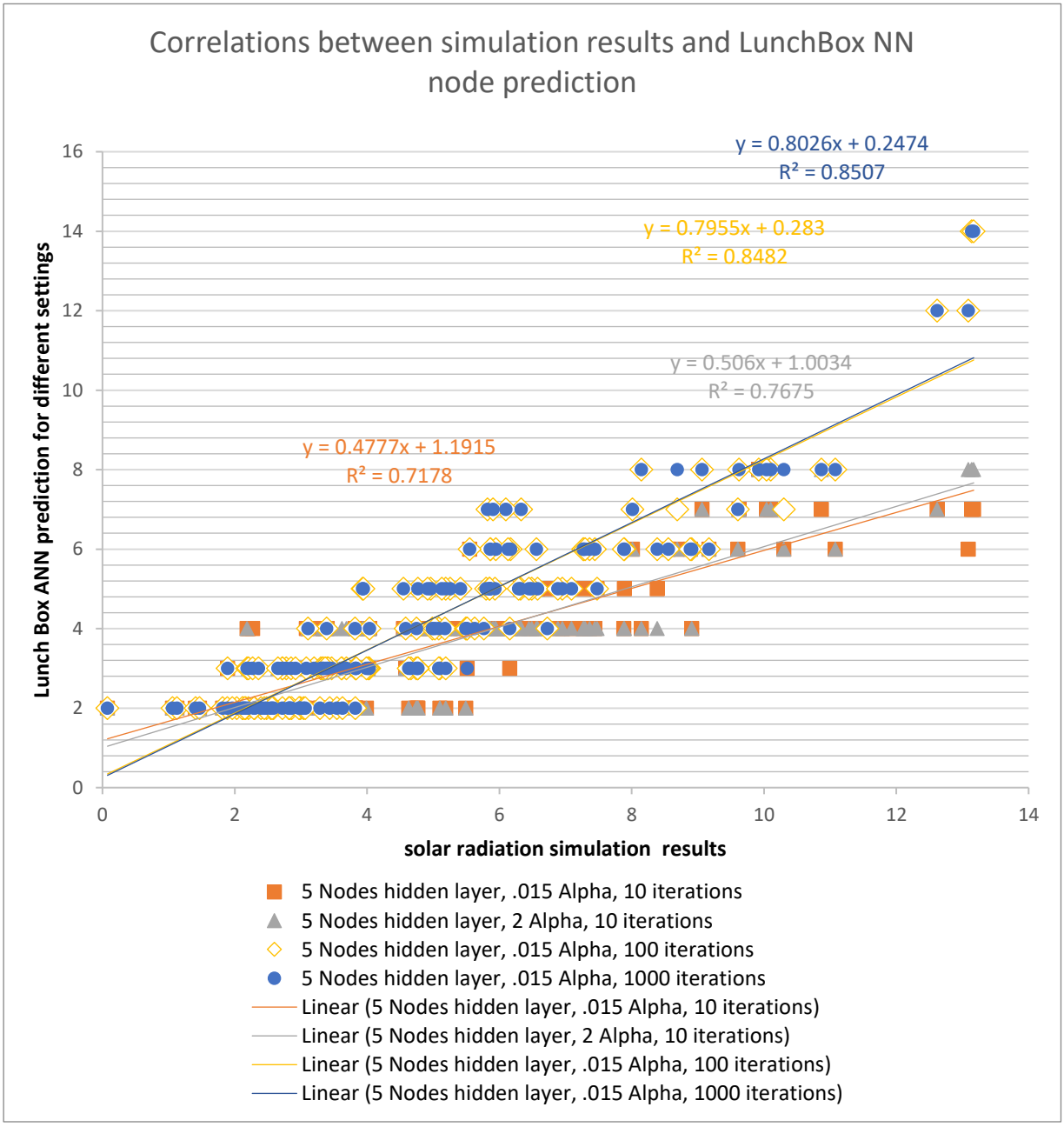


Figure 7-2 Acccuracy results for different LunchBox testing phases

The first test achieved around 50% accuracy in the direct value correlation between the prediction and the simulation of the same urban configuration. That was with an alpha of 2, and this took around five minutes of training and prediction. Then, when reduced to .015 to reduce the prediction error, this resulted in almost 48% accuracy with the same amount of time consumed. The subsequent trial was to raise the iterations to 100 and reached up to 79.5%, yet it took 52 minutes to train and predict the 152 building tags of this configuration at hand. The last trial was to raise the iterations to be 1,000 iterations. This increase did not result in a significant accuracy change as it reached 80%, while the training and prediction consumed two hours and 42 minutes. The results of these tests are shown in Figure 7-2.

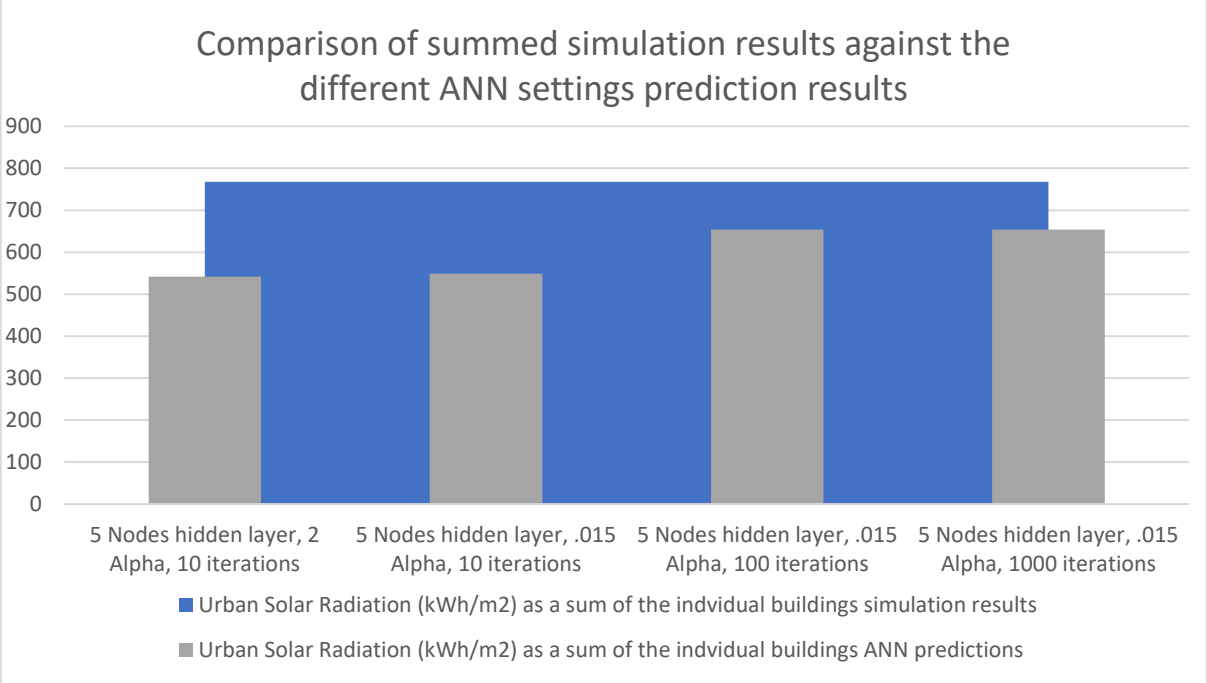


Figure 7-3 Urban scale results for different LunchBox testing phases

To calculate the urban scale results, the individual building results were summed for each trial compared to the sum of the simulation results. As shown in Figure 7-3, the difference of changing the alpha value is not as significant as the increase of iterations from 10 to 100. The first test was 70% close to the simulation results summary, while the change of alpha value increased that to 71.5%. The similar accuracy levels for the 100 and 1,000 iteration tests were clearer in the summed results. Both of the tests achieved 85% accuracy for the whole configuration solar radiation prediction in kWh/m^2.

These tests have provided a clear understanding of the extent of applying ANN on predicting solar radiation performance for an urban configuration using a classified database of buildings. It is important to note that this training has only included the 20,000 entries of the actual 400,000 database of building tags with solar radiation performance. This showed that applying ANN prediction using this extensive database allows for a novel way to save the time consumed for simulations. The time consumption in these tests is one of the limitations of using this LunchBox ANN node. The fact that the number of requested predictions does not significantly affect the time for training and prediction has indicated that this can be enhanced by separating the training from the prediction process. In this way, the training of the ANN will take place only once while the time consumed for it can be reutilized in predicting different entries. Although the LunchBox node has an open-source code available, the code uses unclear classes and does not have enough resources to learn the source code core

principles (Proving Ground, 2017). This led the research to look for another resource to apply ANN principles within this framework scope. The idea is to find an open-source code for ANN that can be edited to enable it to separate the training and the predictions.

7.2.2 Open-source Python ANN node

One of the benefits of the previous phase of testing the ANN application is the fact that the ANN node has accepted multiple inputs for prediction, which allowed the framework to avoid the limitation of having a dual control system for iterating the geometry and tags on the buildings' and the urban configuration's scale. This meant that the control part of the framework in this stage was responsible only for iterating the urban configurations.

The ANN node used for this framework is an open-source Python code initially inherited from a blog website that teaches how to build a neural network in a simple method with no loaded libraries (The Codacus, 2017) and goes through different stages of enhancements until it reached to the final version used. In this section, the parts added during development to the original code will be discussed, along with the initial and final stages of testing the utilization of ANN principles on the database at hand to get the optimal time-saving and accuracy needed. The original code was divided into a series of classes, each of which defines some functions to build the ANN simply.

The first class defines the connection between the neurons. This is to define the neuron weights and their inter-connection in the network. The neuron class follows that by defining each neuron's place in the layers and its static value, learning triggers and learning rate. This is followed by the code lines that define the network-related values as the Sigmoid Value and its derivation through learning and defining the error rate for each neuron to be calculated later in the code to calculate the backpropagation between different layers of the ANN. This leads to the last section of this class focused on the data flow between neurons and layers. Feed-forward is defined in a section to handle the data received by the neuron, either as directly from the ANN inputs or internally between layers. It gets to be multiplied by the weight of the connection connecting this neuron to its previous layer. The last section in this class is the definition of backpropagation. As mentioned earlier, this process is responsible for the neuron learning process. This happens by changing the weights to reduce the error of

prediction. This allows it to enhance the learning for each neuron based on its weight and collectively for the whole ANN error to be reduced.

The following class is the one that builds the network out of these neurons. This class is responsible for structuring the neurons based on the input of layers and their included neurons. This structure is named in the code by “Topology”. It is a list input of the number of neurons in each layer of the ANN. The dataset used to train the network is handled in the following section by calculating the initial error of the network in comparison to the input data targets. This is followed by the sections responsible for calling the feed-forward functions for every neuron in the network from the input layer to the output layer. The section is doing the same calling process for the backpropagation function for every neuron in the network from the output to the input layer. The last section of this class is the results collecting section in the network class. Then, the original state of the code had a testing set of data at the end to test the whole ANN process after the training code class.

7.2.2.1 *ANN Python customization*

The main idea of editing this code was to separate the training process from the prediction process. This took place by creating two Grasshopper Python components. The first component was used for the training part of the ANN application. It consisted of the original code classes for Connection, Neuron and Network. They added one more class instead of the original training class in which the training took place and then extracted the trained network as an output to be fed into the second component. As shown in Appendix C, this class of code identifies different parts in the ANN. It starts with calling the neuron eta and alpha static values identified earlier in the parts of code adapted from the original code as static values and obtained from an integer input to the component. Then it identifies “maxiterations”, which is the maximum number of iterations allowed for learning over the whole ANN before it starts to output predictions. This number is set as an input to the Grasshopper component to be set by the user. Then it calls the feed-forward and backpropagation process over the input data and its targeted results. These are also fed to the Grasshopper component as a list of target data. The input name is “Classification Data”. Another input is the tree of lists containing the data used to train the network. Each list contains one column of the training database parameters and is named “Training Data” when input into the component. Another input is the “Error Threshold”, which is used to break the training process by comparing it

with the ANN calculated error. Then the network is defined from these inputs and built based upon the “topology” input, which is a list with the neuron number of each layer. The last lines of the code export the trained ANN as an output with the name “nn”.

The following component is responsible for receiving the trained ANN and new data to be predicted. The code sets the input data under the name “test data” and creates a predicted list. Then it activates the trained ANN. The rest of the code is about activating the “feed-forward” function and obtaining the prediction results of the input data, either as one entry or a list of entries to be predicted at the same time. Then, the component exports the output predictions. This code is shown in Appendix C.

It is important to note that the output of the prediction, in this case, follows the same remapping process as that of the training data targets. If the training data had its targets remapped to be between 0 and 1, then the output prediction will fall between the same bounds. Therefore, another remapping process had to take place to get the results back to the target original status.

Through this two-component sequence the time for iterating the prediction has been reduced and the training time is accounted as just one time, and the trained ANN is reused for different inputs following the training time. This saved the overall time and allowed for predicting different urban configurations in less time. As discussed earlier, the size of the input training data has a significant impact on the training time. However, this method of separating the training enabled the loading of larger datasets and enhanced the possibility of testing it against different urban iterations.

This version of the ANN code had some training to consume around four hours to train a dataset of 50,000 entries and 24 minutes for a dataset of 500 entries. It took around 20 milliseconds to predict one urban configuration with three configurations per minute with an average of 80 tags in each one. The training settings also needed several trials for testing and adjustment to reach the settings for each dataset size. This led to having a rather generic way of enhancing the training and having more control over the accuracy rate achieved in the training rate. The direct feature that can give such control was to trigger the learning error rate by detecting the mean square error achieved by each learning iteration. Based on this,

the ANN will break or continue learning. There was a need to get an output of this feature to understand the nature of training that took place to get the ANN ready for prediction.

The first trial undertaken for this objective was to extract the mean square error for each iteration. Then, the ANN training stops based on the outcomes of the extraction of these mean square errors. The basic idea is to have a case that when the mean square error reaches a certain value for a number of iterations, this will stop the training. Another case was for the training to be stopped due to reaching the maximum number of iterations. Furthermore, the

```
import sys
import math
import random
import pickle

sys.path.append(pythonPath)
import network

tempList = [list(i) for i in trainingData.Branches]
trainingInputs = []
for i in range(len(tempList[0])):
    newList = []
    for j in range(len(tempList)):
        newList.append(tempList[j][i])
    trainingInputs.append(newList)
tempList = [list(i) for i in trainingP.Branches]
trainingOutputs = []
for i in range(len(tempList[0])):
    newList = []
    for j in range(len(tempList)):
        newList.append(tempList[j][i])
    trainingOutputs.append(newList)

tempList = [list(i) for i in validationData.Branches]
validationInputs = []
for i in range(len(tempList[0])):
    newList = []
    for j in range(len(tempList)):
        newList.append(tempList[j][i])
    validationInputs.append(newList)
tempList = [list(i) for i in validationP.Branches]
validationOutputs = []
for i in range(len(tempList[0])):
    newList = []
    for j in range(len(tempList)):
        newList.append(tempList[j][i])
    validationOutputs.append(newList)

nn = network.Network(topology)
nn = nn.trainNetwork(trainingInputs, trainingOutputs, validationInputs,
validationOutputs, errorThreshold, maxIterations, eta, alpha)
pickledNetwork = pickle.dumps(nn)
iterations = nn.iterations
mseList = nn.mseList
stop = nn.stop
```

Figure 7-4 Final version of the component of calling and training the Pickled ANN

last case was to achieve the error threshold without reaching the required number of iterations with the square mean error value to stop the training.

A more structured way to control the training was to enable what is known as “cross-validation” of the ANNs. Cross-validation is simply a method to hold part of the training dataset and use it for testing the predictions to enhance the predictions for new data entries. This way prevents over-optimistic results of testing the prediction results against itself as it was used once before in training. So the preserved set of data used for the testing set of data within the training process ensures the ANN is not trained only for the input data set. Thus, it allows the predictions to be more accurate when it predicts data that it did not see before during the training phase. There are different ways of conducting this method within the training phase (Arlot & Celisse, 2010; Pedregosa et al., 2010; Varoquaux et al., 2015). The most straightforward application of cross-validation was chosen to be implemented in this research. These cross-validation techniques are about holding a part of the training data and testing the ANN prediction to ensure the prediction error from the validation data set is less than the one calculated from the training dataset (Arlot and Celisse 2010).

The final version of the code regarding this included three cases in which the training was stopped. The first reason to stop the training and extract the ANN was that the maximum iterations allowed had been reached. The code exports a value of number 1 to indicate that the training has stopped because of this reason. Then, in the second case, the validation error achieves better results than the error threshold defined as an input to the node: here, the code exports a value of a number 2. The last case where the training is stopped is when the validation error is less than the training error, and this cause has an indication of number 3 as an output from the Grasshopper node. The calculation of the training error is not collected before the ANN runs 25% of the determined maximum number of iteration to give the ANN a meaningful time to train before initiating any of the three causes where training is stopped.

The code was also edited to export a list of mean square errors for each iteration to show the development over time. Also, another output is the number of iterations that took place until the ANN reached its final training to indicate in case the reason training was stopped was due to not reaching the maximum number of iterations.

As shown in Appendix C, the code starts with calling the inputs and defining them. Then it starts the ANN training for the training data and collecting the error ratios for the training. Following this, the prediction starts on the cross-validation dataset, followed by collecting the validation error by comparing it with the validation target inputs. The last section of this class is to determine when to stop the training based on the three cases mentioned earlier.

This version of the ANN was capable of handling the whole dataset with iterations reaching 1,000. Although the training still consumes some time, lasting in some instances to seven and a half hours to load 100,000 entries and train the ANN for 1,000 iterations on an ANN with two hidden layers. This time is still reasonable if factored to iterating different urban configurations and getting the performance prediction in almost 20 seconds maximum. This capability was further developed by importing a Python library that enables the saving of the trained ANN to be recalled and used for different iterations and on different PCs (Python Software Foundation, 1990). The library name is “Pickle”. It has the function of calling for a version of the ANN code saved outside Grasshopper in plain text and using it to train the ANN on the dataset inputs. It generates a version of the trained ANN that can be saved in text format to be reused in predicting different newly generated entries.

Application of this library to the ANN nodes resulted in having the ANN part of the framework consisting of three parts. The first part is the actual ANN code saved in text format containing the last version of the ANN code, including the cross-validation application. Another part is the call-out node in Grasshopper. This component is now responsible for calling the ANN code and training it and extracting the trained ANN and an optimal Pickled version to reuse it, and the regular ANN outputs mentioned earlier. The last component is the prediction component. It stays a Grasshopper component that receives the trained ANN, either directly from the Grasshopper training component or a pickled ANN after being called into the Grasshopper platform.

As shown in Figure 7-4 Final version of the component of calling and training the Pickled ANN, the code calls the training text code-named “network,” then it runs the training and validation using the input datasets for each function. Finally, the last lines of code identify the node outputs, which consist of the pickled ANN named “nn”, the number of iterations, Mean Square Error (MSE) list for each iteration “mseList” and the training stop reason named “stop”.

7.2.3 ANN node testing results

The previous section has discussed the different versions and development of the application of ANN principles through Python coding to be hosted on the Grasshopper platform and embedded within the framework to predict solar radiation performance on an urban level to save simulation time. The different versions of this node have been tested to record its timing and accuracy compared to the conventional simulation results saved in the tag's database.

7.2.3.1 *First phase of testing ANN node*

The first phase of the test has been set to have the ANN predicting the solar radiation based on the classification tags found by the lookup definition. This way allows testing the validity of the lookup process and the accuracy of the ANN predictions.

The early trials of the ANN code with its first adopted version have shown some indications to enhance the structure of the hidden layers and the input settings to reach close to optimal settings that can produce reasonably accurate predictions.

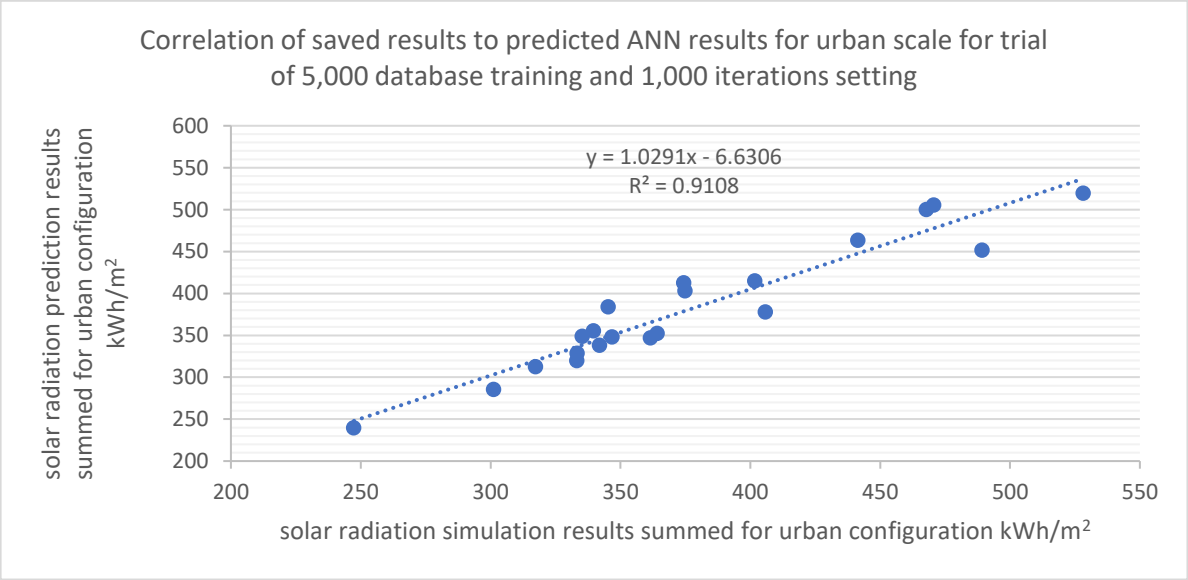


Figure 7-5 ANN prediction correlation with saved database simulation results for ANN with 5,000 training dataset and 1,000 iterations

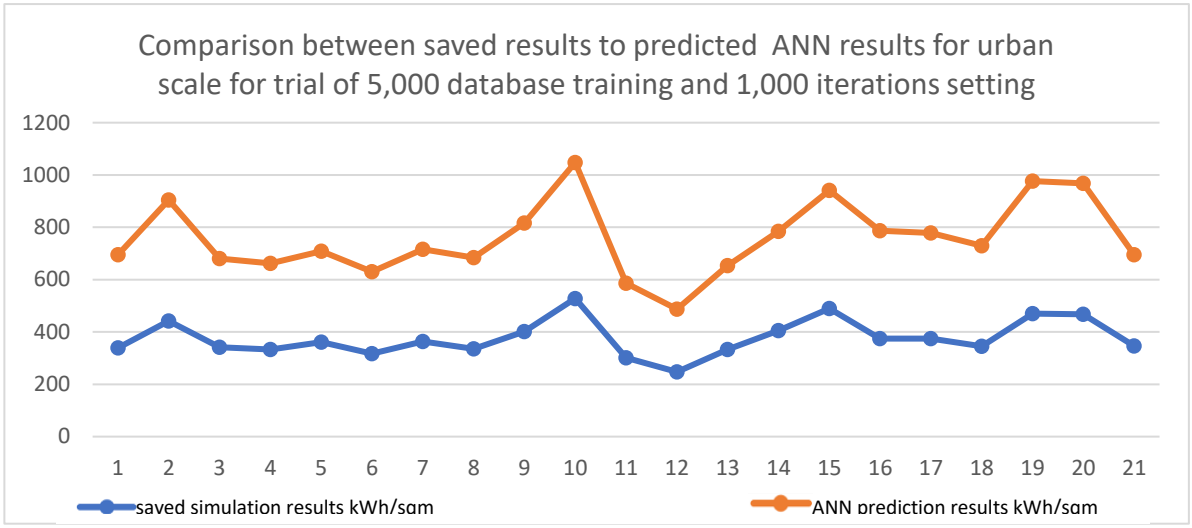


Figure 7-7 ANN prediction comparison with saved database simulation results for ANN with 5,000 training dataset and 1,000 iterations

The first trial of training the first version of the ANN was to use 5,000 entries from the database with 1,000 iterations on a three-hidden layer structure for it. The layers contained 10, 7 and 4 neurons each, as in the input order to output direction. The Alpha was set to .015, and error threshold to be .0001, and eta to be .01. This training has taken around 40 minutes to have a trained ANN as an output. Then the subsequent trial took a sample of 20 urban configurations to test the lookup and ANN training. As mentioned earlier, the lookup process

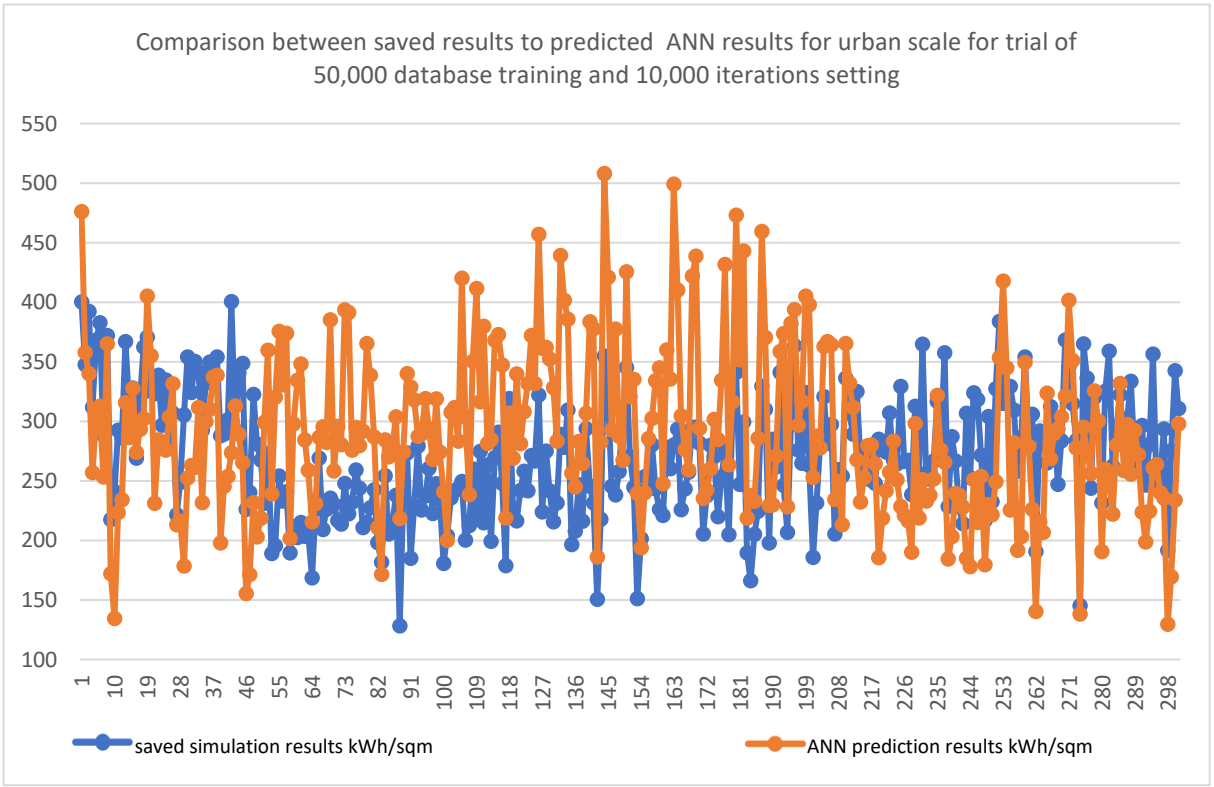


Figure 7-6 Comparison between simulation and prediction results for 300 configurations with ANN trained on 50,000 entries and 10,000 iterations setting.

is tested to look for similar tags and extract their solar radiation results from the database; then, it feeds these tags to be predicted by the ANN. The predictions are collected for each configuration and compared with the sum of database saved results for the same configurations. The lookup in this stage found around a 50% average of similar tags for each configuration. As shown in Figure 7-5 & Figure 7-7, The results for the 20 urban configurations were around 88% of value difference, and the prediction took less than 20 seconds to predict around 70 tags and record it in Excel.

The following test to this one aimed to investigate the enlargement of the dataset and maximize the iterations to understand the ANN training behaviour. It had 50,000 entries from the database with two layers of seven and four nodes from input to output order. This trial used 10,000 iterations to test the need for a large number of iterations and to know its time cost. The alpha and eta and error threshold was set as in the previous test trial. This training took 37 hours and 30 minutes to finish. Then the testing took place on 300 iterations. The configurations' solar radiation was calculated by summing the simulation results and predictions for each configuration. The lookup in this stage was not in its last updated version, but it followed the same structure and methods mentioned in section 6.6. Therefore, the lookup managed to find around 30% similar tags in the database, which was fed into the prediction component. A minute was enough to finish the prediction of three urban configurations and record it in Excel sheets. The accuracy achieved a 42% value correlation

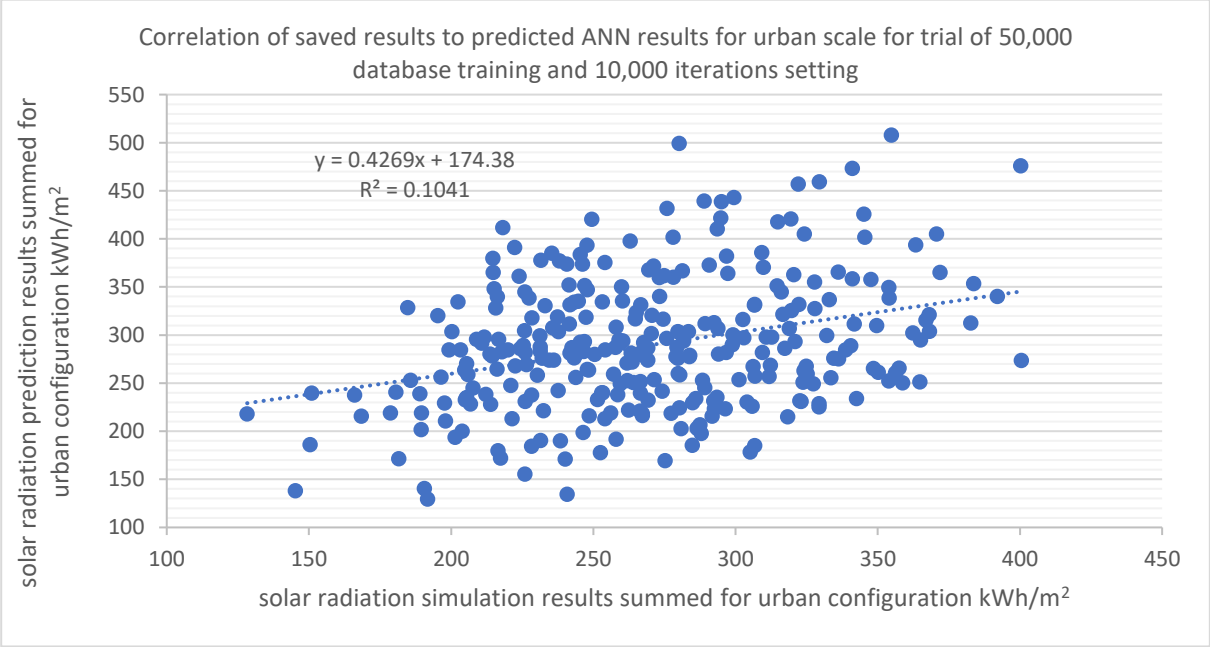


Figure 7-8 Correlation between simulation and prediction results for 300 configurations with ANN trained on 50,000 entries and 10,000 iterations setting

between the summed simulation results and the summed prediction results. This ratio is driven by a similar ratio in most comparisons between the two results on the individual building scale (see Figure 7-8 & Figure 7-6).

Following this, the first edit of the ANN code was done and tested for different sample sizes. The test again was for the found tag sign in the database. The settings were the same as for the training component. The only changed parameters were for the iterations, and the topography was two hidden layers with seven and four neurons from input to output. It was tested for 300 urban configurations.

Table 7-1 showing the difference in time and correlation accuracy in different training data sizes and maximum iterations

Training data size	Iterations	Time consumed for training (hours)	Correlation accuracy for trained data
50,000	1,200	3.9	83%
75,000	1,200	5.8	92.7
5,000	1,000	2	38.3%
75,000	1,000	5.5	87.7%
100,000	1,000	7.4	94.5%

The iteration change has shown a clear difference in the time consumption and accuracy, as shown in Table 7-1. Moreover, the increase of sample size has shown a clear enhancement in the prediction results. It is important to note that the high accuracy ratio is due to the comparison between the prediction and the dataset introduced for the training. These results led to the development of the node to have a cross-validation part of making sure that ANN is not biased towards the training data only.

Table 7-2 showing the difference in time and correlation accuracy in different ANN settings

Training data size	Hidden layers from input to output	iterations	Error threshold	Eta	Time for training (minutes)	Correlation accuracy for trained data
500	7	8,000	.0001	.2	16	92%
2,000	7	8,000	.0001	.2	66	89 %
2,000	7	8,000	.0001	.15	66	88 %
500	3	5,000	.0001	.01		84%
500	12	5,000	.0001	.01		86%
500	2	5,000	.0001	.01		84.7%
500	5	5,000	.0001	.01		86.2%
500	7	5,000	.0001	.2		93.1%
500	7	5,000	.0001	.5		92.2%
500	7	5,000	.0001	.3		92.7%
2,000	7	5,000	.0001	.2	40	89.5%
500	12, 9, 5, 3, 2	5,000	.0001	.2	31.5	Faulty results
500	12, 9, 3, 2	5,000	.0001	.2		Faulty results
500	12, 3	5,000	.0001	.2	19	90.6%
500	12, 3	5,000	.01	.2	18.4	Faulty results
500	12	5,000	.0001	.01		86.2%
500	7	5,000	.0001	.01		86.1%
1,000	7	5,000	.0001	.2	17.3	98.2%
10,000	7	5,000	.0001	.2	42	84.6%
10,000	9, 3	5,000	.0001	.2		87.7%

This was followed by a series of tests of the training with different settings to understand better the parameters and the continuous editing of the code. These tests were conducted on only one configuration with small-sized training data. The tags tested for that configuration were around 140 tags out of 160 total buildings in the neighbourhood. The test did not use a fixed number due to the change in the lookup process and the limitations of the generation algorithm mentioned earlier.

As shown in Table 7-2, the variation of the training data size tested was between 500 and 10,000 entries. The topography of the ANN has been changed between a different number of

neurons from a single layer and reached five hidden layers with different values shaping a structure of pyramidical shape from input to output direction. Guided by the previous tests, there was a maximum of 8,000 iterations with mostly 5,000 iterations to balance the sample's small size. The error threshold was set to be either .01, .001 or .0001 in most cases. Changing the eta varied between .01 and .5 and was .2 in most cases. Not all time consumption was documented, yet the time recorded indicated the impact caused by the change of parameters and provided an insight into the boundaries of the time consumption cost. The recorded time varied between 66 and 16 minutes for the least training parameter settings. The accuracy being tested against known data to the ANN achieved optimistic ratios. The accuracy achieved a high of 98.2% and a low 84% accuracy to compare the prediction and simulation results of data that were part of the training dataset.

These tests illustrated the impact of parameters on the training time consumption for the training and the expected accuracy for ANNs predicting the solar radiation of the database at hand.

7.2.3.2 *Second phase of testing ANN node (testing data that was not part of the training)*

The tests against the unseen data followed this stage in different scales of sample sizes. In the unseen data tests, the accuracy was calculated for database tags that were not introduced by the training component.

The tests with unseen data were conducted with different sample sizes. Predictions were also made on the scale of the tags and for the whole configuration for each training sample size to show the impact of changing the sample size on the accuracy. This helped to show the importance of the sample size to reach a reasonable accuracy of predictions and the impact of changing settings on the training consumed time with the larger datasets.

The first data sample size was a 10,000-entry dataset. To minimize the time consumption, there was only one hidden layer with seven nodes in this ANN. The training parameters were set to 250 iterations with an error threshold of .001. The MSE list output started with the value of 0.002739 and ended up with 0.002726. The reason for stopping the training was because the maximum number of iterations allowed had been reached. The training time was a little more than four minutes.

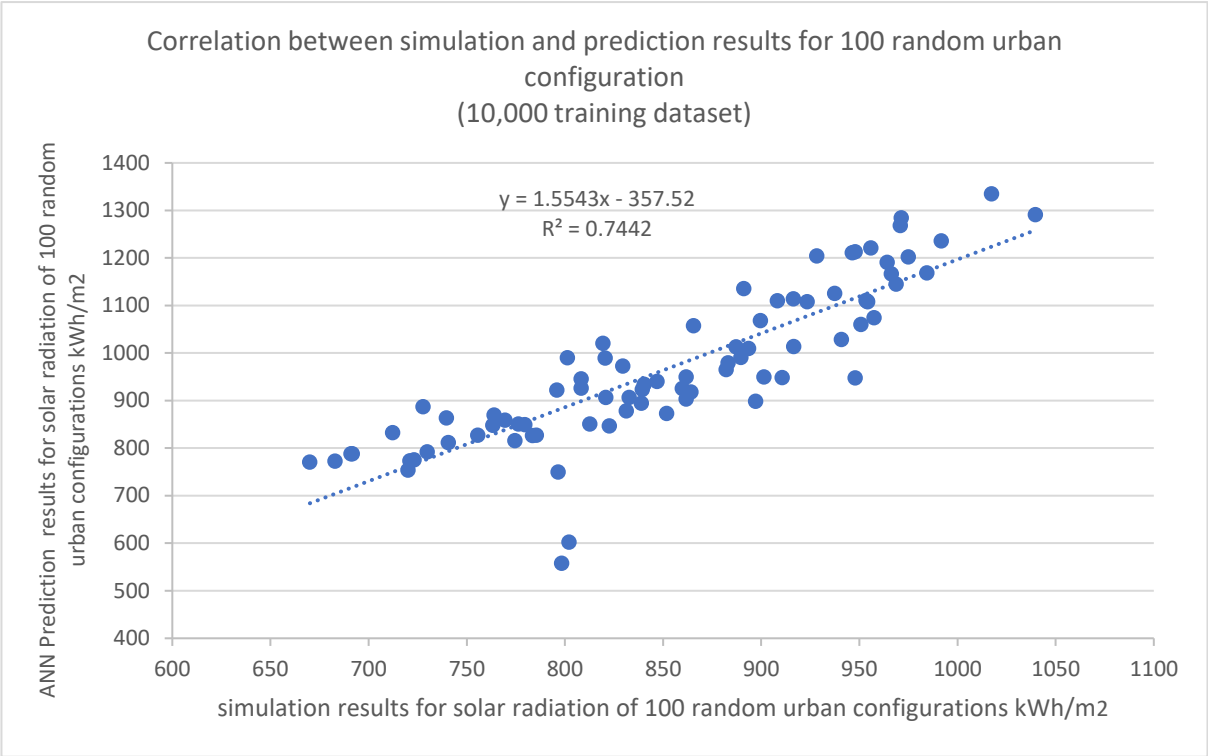


Figure 7-9 Correlation of simulation and prediction for 100 random urban configuration for ANN 10,000 training dataset

This ANN was tested on 100 randomly selected configurations from the larger pool of configurations for the framework testing. This test resulted in a high coefficient of determination (R-squared value) with 74.4% for the positive linear correlation between the two results (see Figure 7-9).

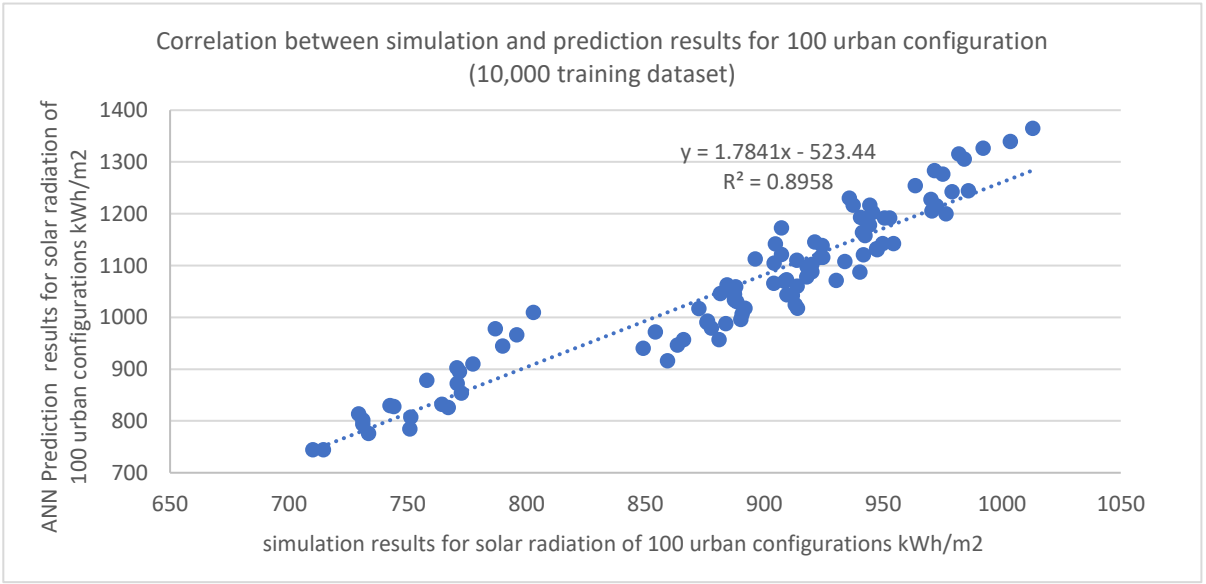


Figure 7-10 Correlation of simulation and prediction for 100 urban configuration for ANN 10,000 training dataset

Another test was done for the same ANN, but this time for a set of 100 configurations that share some features with the training database. This test had a better result due to the relativity of the features. It resulted in an 89% R^2 value which illustrates a high correlation factor (see Figure 7-10). This difference in results comes from the role played by the lookup in finding more similar tags from the database and reducing the number of classification tags fed to the ANN prediction node.

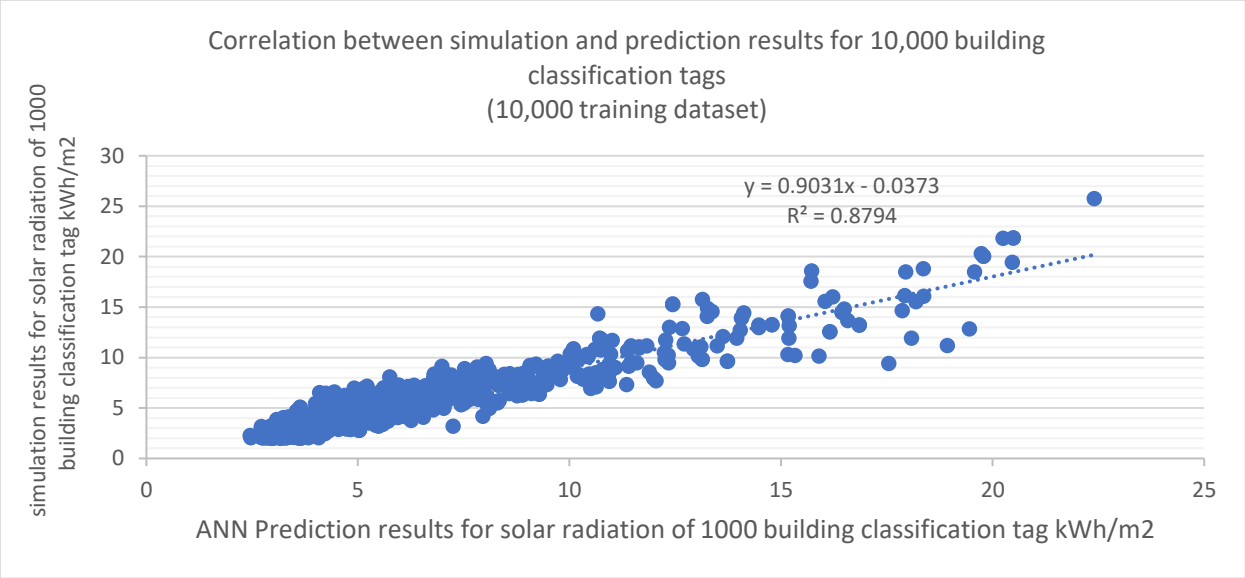


Figure 7-11 Correlation of simulation and prediction for 1,000 building classification tag for ANN 10,000 training dataset

The last test, shown in Figure 7-11, was to have this correlation between the prediction and simulation results on the building level. This was conducted by loading 1,000 tags from the database not included in the training dataset for this ANN and comparing its saved results with the prediction. The correlation got an 87.9% R2 value for this linear regression test.

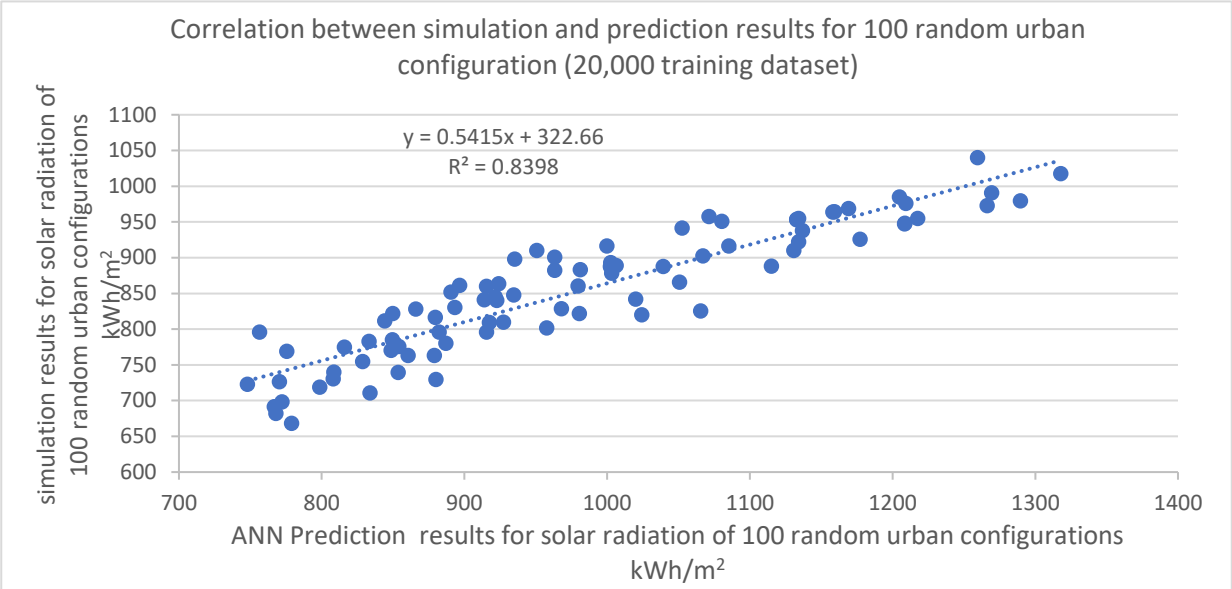


Figure 7-12 Correlation of simulation and prediction for 100 random urban configuration for ANN 20,000 training dataset.

The following phase of these tests was in respect of the same three groups of predictions for the ANN node that was trained on 20,000 dataset entries. It is important to note that these performance predictions were new to the ANN node and were not part of the loaded training datasets. The training ended after one iteration with the same settings as the previous dataset sample size due to the cross-validation data achieving an MSE that was less than the training MSE dataset. The time consumed for this short training run was two minutes. As shown in Figure 7-12, the first group of random 100 configurations achieved a value of 83%.

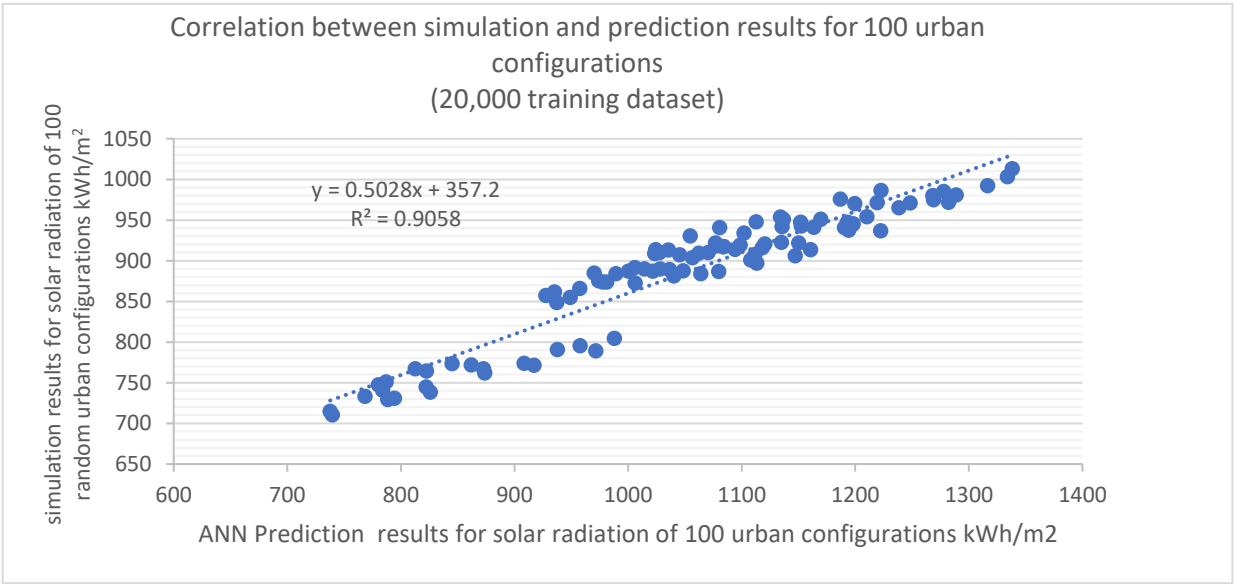


Figure 7-13 Correlation of simulation and prediction for 100 urban configuration for ANN 20,000 training dataset

The second group test achieved better results with a 90% R^2 value, a considerably high coefficient of determination for such a linear correlation graph (see Figure 7-13). Following those two groups, the classification tag test of 1,000 building performance predictions achieved a 93.4% R^2 value unexpectedly with fewer training iterations (see Figure 7-14).

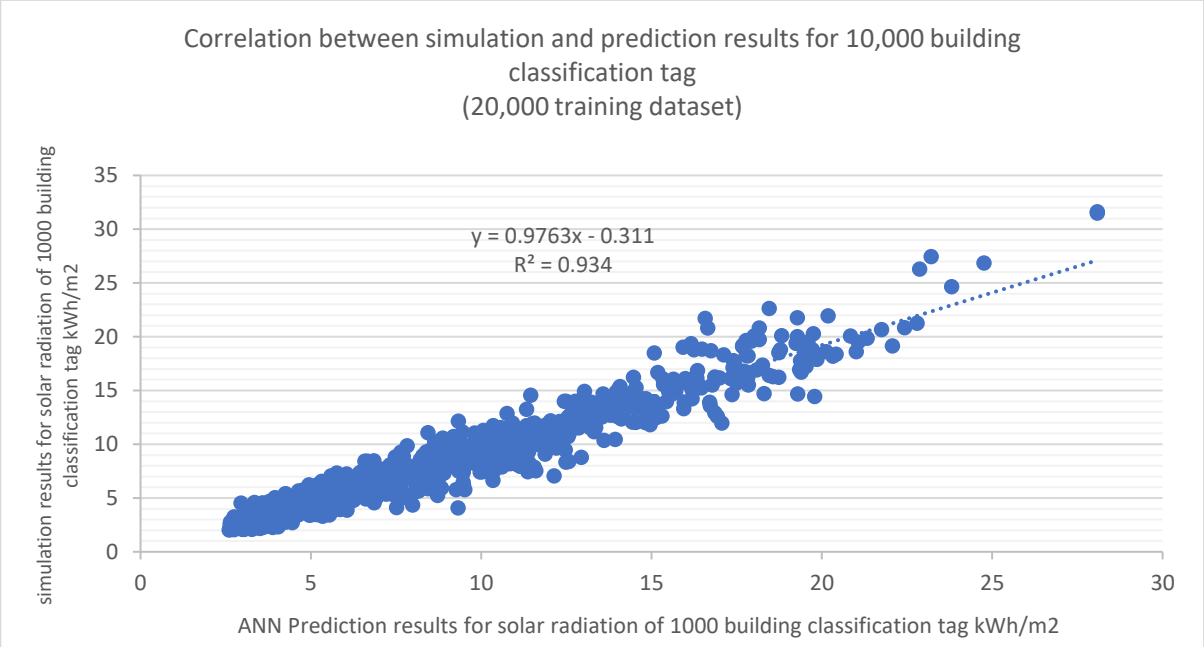


Figure 7-14 correlation of simulation and prediction for 1000 building classification tag for ANN 20000 training dataset

Following those two sample sizes, the research enlarged the sample size to 150,000 tags for a training dataset.

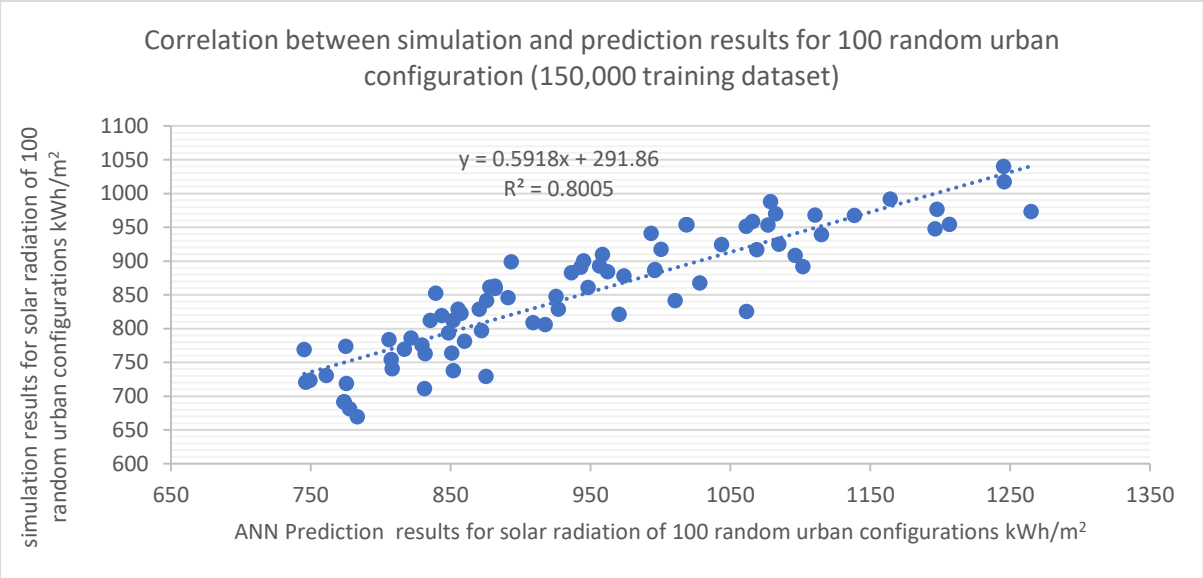


Figure 7-15 Correlation of simulation and prediction for 100 random urban configuration for ANN 150,000 training dataset

The 150,000 entry ANN had the same settings as previous ones. Only the error threshold was changed for more iterations in the training phase. The error threshold has been reduced to be .0001. This resulted in running the whole 250 allocated iterations of training which started

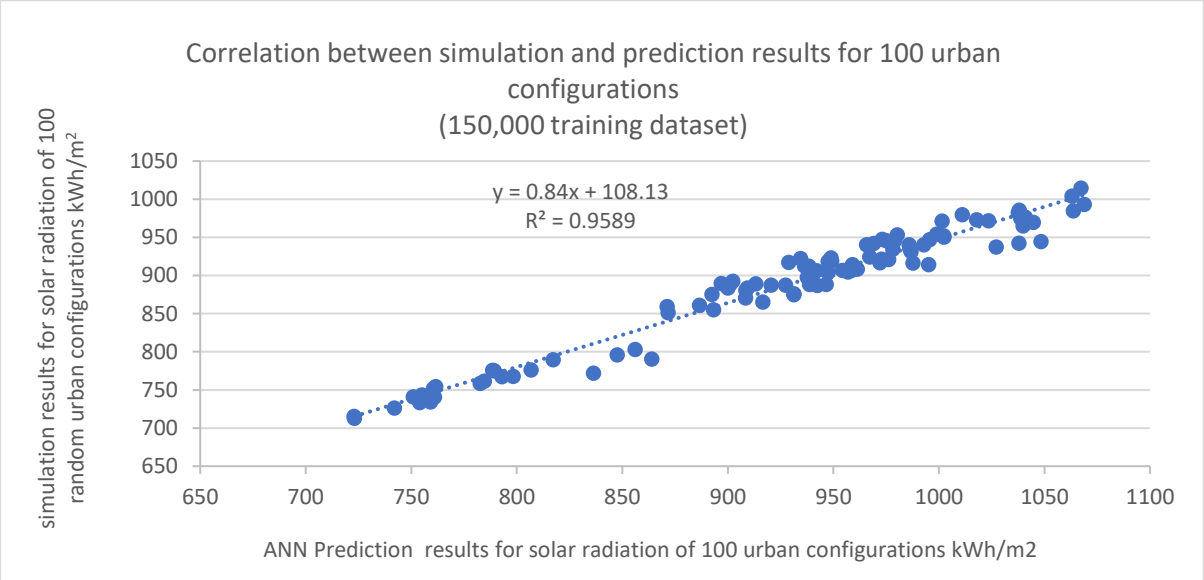


Figure 7-16 Correlation of simulation and prediction for 100 urban configuration for ANN 150,000 training dataset

with an error of 0.001877 and ended up with an error value of 0.001588. The time consumed for training this ANN was almost 38 minutes. The same three groups of testing have been introduced to prediction also for this version of ANN testing.

The first group of 100 random configurations achieved an 80% R^2 value for this correlation between simulation results and ANN predictions (see Figure 7-15. The 100 configurations which shared similar features to the database achieved 95.8% for the same value (see Figure 7-16). This high value is due to the larger number of tags found by the lookup due to the larger size of the used sample. As shown in Figure 7-17, The last group of testing was for the 1,000 individual tags comparison from the database. This correlation reached 94% for the R^2 value.

The last test of unseen data has been set for a sample of 200,000. To better understand the ANN's performance on this sample, multiple settings were tested against the same three

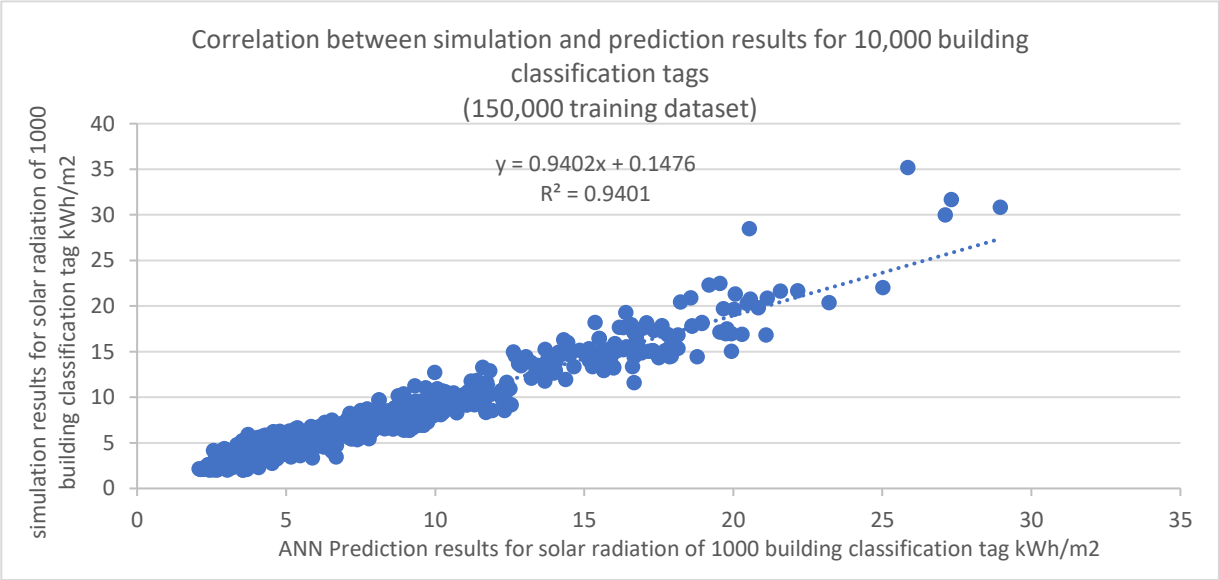


Figure 7-17 Correlation of simulation and prediction for 1,000 building classification tag for ANN 150,000 training dataset

groups of predictions. The first ANN setting had the same settings as the previously tested ANNs. For this one, the MSE list started with .00151 and reached a low of .001408 for the last trained iteration error threshold. Another ANN was set to have two hidden layers of seven and four neurons, respectively, from input to output direction. The maximum number of iterations was set to 1,000 iterations. The rest of the settings have not been changed. The MSE list started with .001248 and ended the training with a value of 0.001115. It is important to note that the MSE list only starts recording after 25% of the iterations have been done. This means that in this case, the MSE list has only 750 iterations, and the first ANN had 187

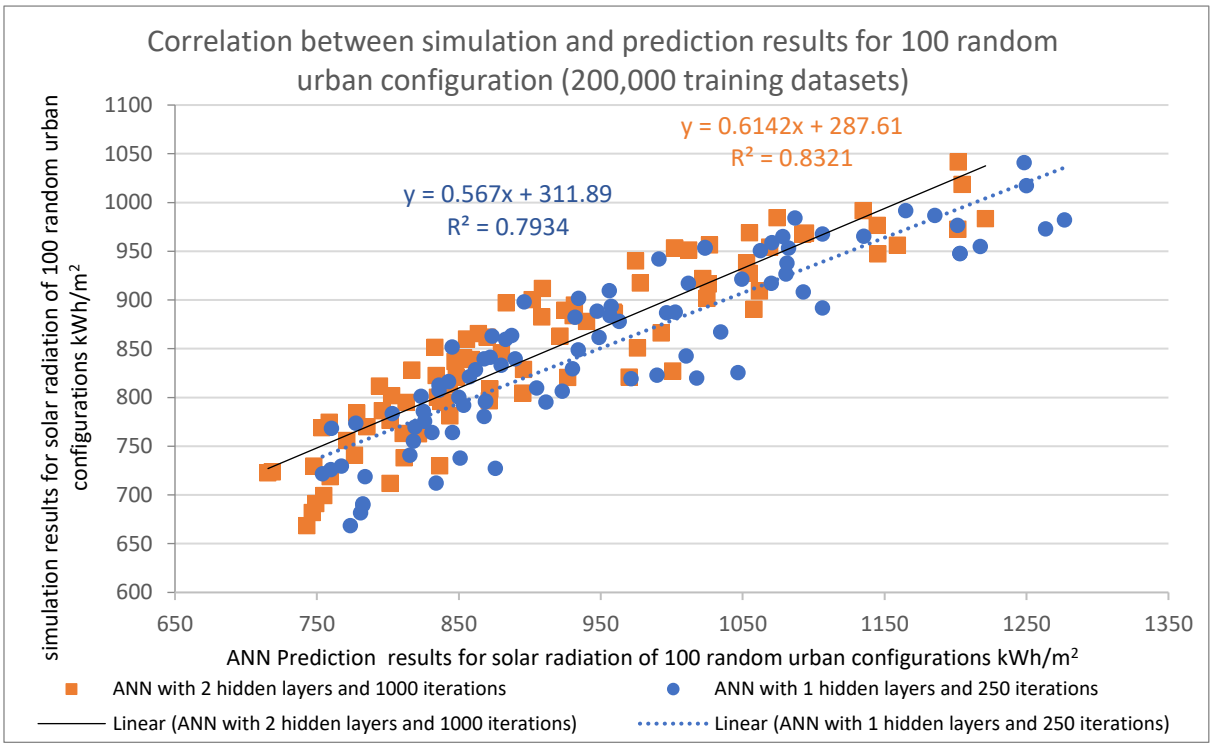


Figure 7-19 Correlation of simulation and prediction for 100 random urban configuration for 2 ANNs with 200,000 training dataset

values. This is also the reason behind the MSE lists, in this case starting at different values of MSE for its first recorded iteration. The time consumed for the first ANN was five hours of training, and the ANN with two hidden layers took six hours and 24 minutes.

As shown in Figure 7-19, The first group of random selected urban configurations achieved an R^2 value of 79.3%. This was enhanced with the added hidden layer to be 83.2% for the

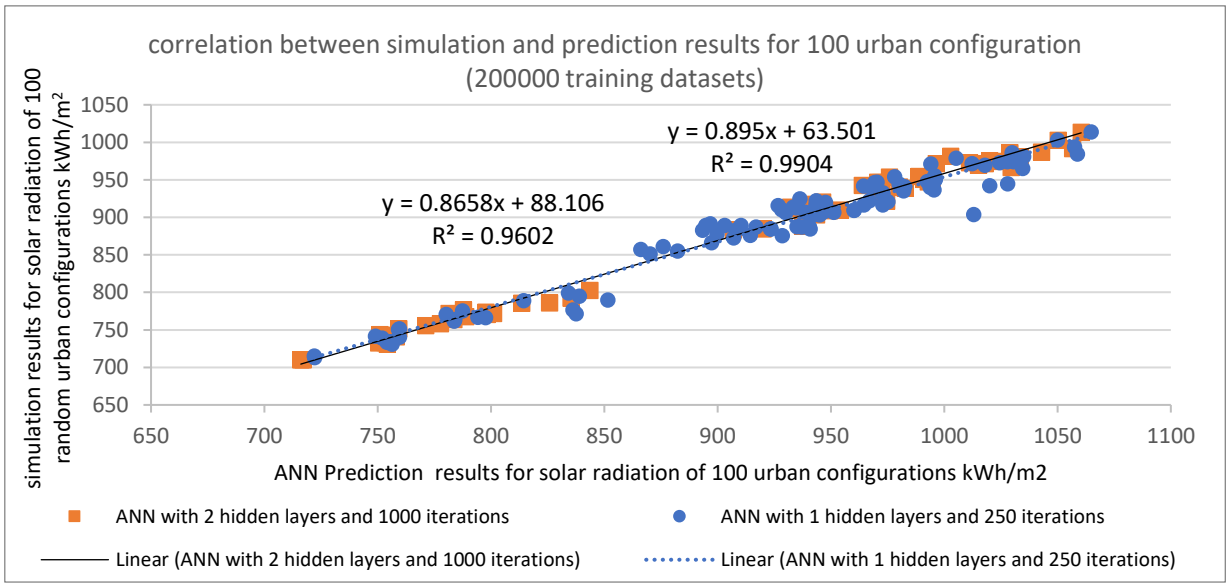


Figure 7-18 Correlation of simulation and prediction for 100 urban configuration for 2 ANNs with 200,000 training dataset

correlation of predictions and simulation results.

The 100 configurations that had similar features of the training database were already included within the training data by enlarging it. Thus, the test resulted in a high correlation coefficient reaching 99% for the ANN with two hidden layers and 96% R^2 value for the ANN with one hidden layer. (see Figure 7-18

Finally, the 1,000 Tag results, shown in Figure 7-20, had a closer variation between the results of the two tested ANNs. The R^2 for the ANN with one hidden layer was 93.7%. This was slightly enhanced for the ANN with two hidden layers by getting a value of 93.9% for the R^2 of the tested correlation. It is important to note that this high value is not due to the lookup assistance as it was not used for the individual classification tags' predictions in any of the unseen tests.

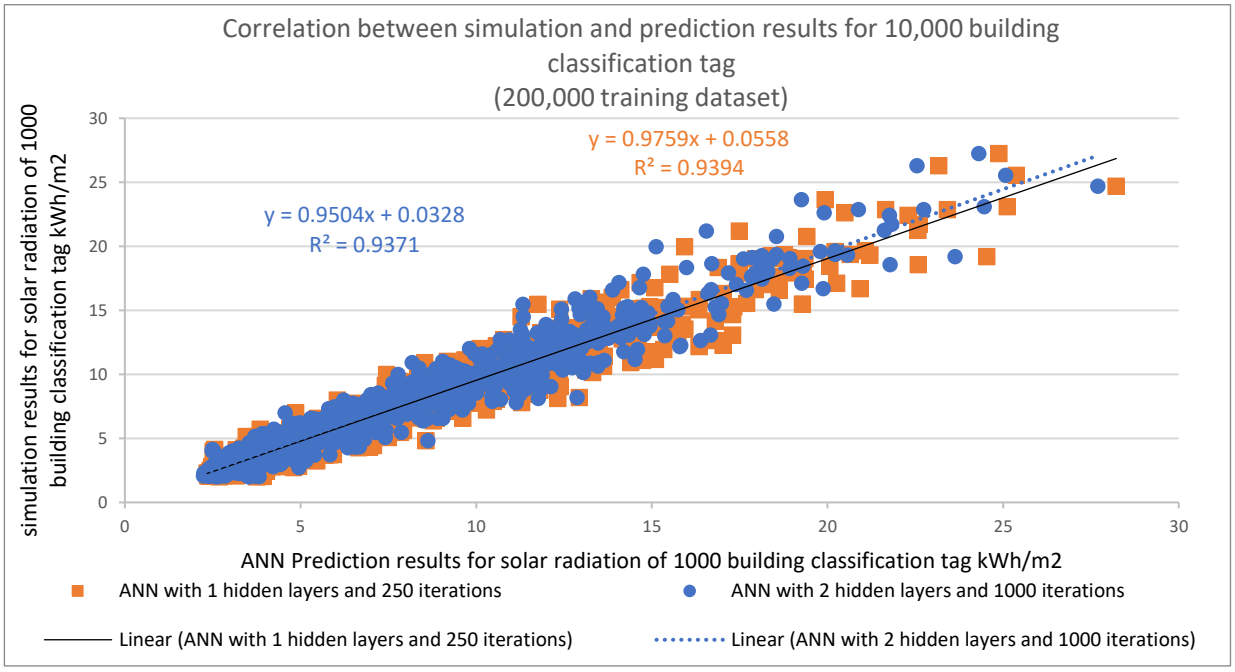


Figure 7-20 Correlation of simulation and prediction for 1,000 building classification tag for ANN 150,000 training dataset

7.3 ANN results discussion

In this investigation, the aim was to assess the ANN addition to the framework and apply its principles in utilizing the classified database to predict performance on an urban level. This was conducted through various studies with different types of datasets and different sizes; the prediction was done for individual classification tags and summed up the urban configuration performance. These tests have shown the benefit of reducing the time of getting the performance, allowing for the performance information to be included within the

early stages of design. The accuracy achieved with the applied ANN code provided significant results for the individual building tags predictions.

7.3.1 Time performance

The assessment of time-saving performance is calculated by comparing the time consumed by the ANN node with the time consumed by the conventional simulation methods. On a building level, ANN predictions consume 3% of the simulation time. This is due to the capability of predicting the whole set of building tags for a configuration in one run instead of simulating the buildings with their different context and getting the results for it

It is important to note that the training time is not factored into these comparisons due to the difference in the number of buildings and configurations that can be done using these trained ANNs. Moreover, solar radiation simulation is one of the least time-consuming simulation aspects compared to other simulation aspects like energy demand or daylighting performance. This finding result shows the potential to utilise this classification method accompanied by ANN predictions to get even better time-performing frameworks that will reach even better time-saving results.

7.3.2 Prediction accuracy comparison to saved simulation results

In this chapter, the ANN node has been tested and developed through different phases. The ANN node developed for this framework had a simple cross-validation method embedded in the training class of coding in its last version. This was linked to the stop of training after reaching an error threshold. The last phase of testing was also conducted over datasets that were not used for training. Then it was also done over different sample sizes to find the better sample size performance. The test for each sample size investigated three groups of predictions. One of these prediction tests was for predicting the classification building tags. The comparison for this test was to feed in 10,000 tags from the saved database, with these tags not being a part of the training dataset. Then the prediction results were compared with the saved simulation results.

These tests have shown a positive linear correlation between the prediction and the solar radiation simulation in kWh/m^2. This correlation testing was looking mainly for the accuracy of prediction in total. Coefficient of determination (R^2) is recommended to measure the ANN prediction results with a clear indicator of accuracy (Zikmund et al. 2003). Further

investigation of the accuracy and relation to individual variables and prediction nodes will be considered in future work using other statistical indicators. As shown in Figure 7-11, Figure 7-14, Figure 7-17 and Figure 7-20, the coefficient of determination (R^2) accuracy for this comparison starts with 87% for the smallest dataset sample size of 10,000 training entries. It then is enhanced to be 93% with double that size. This value is slightly enhanced for the 150,000 training entry sample size to 94%. The last two tests with the 200,000 sample size had similar results for their correlation R^2. These high results were also accompanied by a similar correlation in the actual values between the prediction and simulation at the building level. This means that the error in prediction values follows almost the same percentage as shown in the figures. The high value of correlation with this simple ANN has shown the capability of utilizing this method of classification with ANN. It is worth noting that there is a role for the lookup process in achieving this high level of accuracy. The tags were not part of the tested database for each sample size. The same sample size was also applied for the lookup process, yet being part of the large database gave it the benefit of finding some identical tags in the database. The percentage of found tags did not exceed 20% in any case of the different sample size tests. This highlights the importance of coupling between the two processes of finding similarities between the available database and the input testing data and the prediction outcome of the ANN node. This combination of findings supports the conceptual premise that applying ANN principles can achieve such high accuracy of predictions to save time for the desired simulation and the reuse of the trained ANN nodes to be utilized with different case studies using the same weather file and location.

The impact of the lookup was clearer in the urban scale testing of the last version of the ANN node. The urban scale testing had two groups of testing, each of which had 100 urban configurations in it. The first group comprised urban configurations selected with shared urban inputs as included in the database for training. The other group was randomly generated within the pool of iterations for this case study as the framework generation in Table 6-1. The accuracy for predicting the solar radiation on the urban configuration was investigated by comparing the urban configuration simulation results against the outcome of the framework combining the lookup process and the ANN prediction. This was done by feeding all the tags of the buildings to the lookup process to find similarities within the training data sample. The unique tags continue to be the input to the ANN prediction. The tag

predictions and the found tags are summed to get the final prediction for the solar radiation on the urban configuration. This number is compared with the simulation outcome.

The R^2 results for the selected 100 configurations were 89%, 90%, 95%, and 99% for the four different tested training sample sizes in the same order from smallest to largest as shown in respectively (see Figure 7-10, Figure 7-13, Figure 7-16 and Figure 7-18). These values were lower for the randomly generated 100 configurations predictions' correlations. Figure 7-9 shows that R^2 values started with 74% for the smallest training sample size, followed by 83% and 80% for the 20,000 and 150,000 dataset samples, as shown in Figure 7-12 and Figure 7-15, respectively. The last result for the 200,000 training dataset was 83%, as shown in Figure 7-19. This difference in correlation coefficient values for the two groups of urban configuration predictions can show the impact of the lookup process to enhance the overall result of the predicted urban configuration. These accuracy levels can be found similar to other investigations of applying ANN methods on different data sets and for different prediction goals (Chan & Chau, 2019; Lin et al., 2020). Although these findings for prediction results on an urban scale seem in line with the building prediction scale's findings, it shows a positive linear correlation. However, these correlations are not at the same level of accuracy. This can be noted from the difference in values between prediction and simulation outcomes for the tested urban configurations, especially the random ones. A possible explanation of this might be the selection of the training database. The training database was built from similar groups of urban configurations, and it was not selected from different feature groups in the available pool of iterations. This was due to the limitation of automating a random selection of the buildings in the database, leading to more geometrically unacceptable solutions in the database. Another explanation for this is the expected aggregation of the prediction error becoming clearer with the addition of the classification tag prediction results. These findings show that there is still room for improvement in using the prediction on an urban scale applying this classification method. However, there are some immediately reliable conclusions for the framework aiming for a proof of concept. These are based on the found linear positive correlation to utilize this node of ANN in the following optimisation stage applying genetic algorithm principles to highlight the optimal solution in this pool of iterations.

7.4 Summary

This chapter has shown different stages utilizing ANN principles within this framework to test its potential in predicting solar radiation performance compared to simulation results and the accuracy of predicting the performance at building and urban level, and the potential time saved in the process. The Artificial Neural Network principles have been applied to the classified database of building classification tags and the solar radiation results. These principles were introduced and discussed within the first trial of utilizing ANNs on this database by using a ready-made plug-in (LunchBox). This tool was tested for different neural network settings. The different tests have shown the need for developing an Artificial Neural Network that separates the training time and the prediction time to reduce the time consumed and enable the prediction to take place for different data inputs without reconducting the training.

This outcome drives the research to depend on a basic open-source code for a neural network to create a sequence that separates the training and prediction time. This open-source Python code was selected to be simple and not depend on ready-made libraries that apply ANN principles to reduce the dependencies and allow the editing of the code to occur. The separation of training and prediction codes happened by creating two Grasshopper Python components for each function. Another phase of developing the code was by adding cross-validation principles to the training component. This allowed the training to check its prediction error before releasing the trained ANN. The training had three different settings for the process to be stopped based on the maximum iterations set for the training, the relation between the validation error and the error threshold or the relation between the validation error and the training error. The last used version of this ANN class in the framework allowed the Pickle Python library to be utilized. This package allowed for the reuse of the trained ANN, even for different files and in parallel times. This is due to its capability of saving and recalling the trained ANN from a saved directory.

The testing of this ANN sequence took place during and after the development time. The multiple tests shown in this chapter varied from the early tests, including data entries that were part of the training data or testing data that was part of the classification tags database but was not part of the training data. This was for the first stage of testing during the time of developing the ANN nodes. These different tests have shaped the direction of developing the

ANN nodes and have shown the initial potential of utilizing ANN prediction methods on the database at hand. Another stage of testing followed this one using only the last updated version of the ANN code with its capabilities of Pickling. This stage tested different training data sizes. The tests aimed to test the time saved by the ANN prediction compared to the time consumed for the simulation. It aimed to compare the results for both methods to measure the accuracy achieved by the ANN prediction compared to the simulation results acting as the benchmark for this comparison. To do this, each training database size predicted 100 configuration solar radiation results based on the sum of predictions for the buildings in it. This was done twice, once for a 100 configuration that shares some of the same urban generation settings, and another 100 was selected randomly over the whole database of testing. Each of the training sizes had been tested the same way for 10,000 building tags.

The results of these tests have shown the significance of the size of data entry. Moreover, the accuracy for the largest training dataset size with a 200,000 classification tag database has shown an R^2 of 93% when comparing the prediction against the simulation results. Although some smaller datasets show similar R^2 values as the largest dataset sample size, the error ratio is less for the largest training data size. For the urban configuration prediction results, the 100 urban configurations have achieved almost 90% for the R^2 correlation value, while the 100 random selected configurations achieved 83% for the same value. The effectiveness of the lookup process in the framework was shown by an impact on the selected urban configurations that share related settings with the training data input, which led to fewer predictions and closer results recalled by the lookup process. The high value for an individual building is also due to the elimination of the summing factor. The urban configuration results for both the simulation and the predictions were summed from the individual building results, while the building correlation was calculated directly without intermediate processes that might affect the accuracy. This method has already shown acceptable accuracy. It also opens the door for other simulation aspects to be considered utilizing the same method of depending on the individual building results to build a database for predicting urban configuration results.

8 Genetic Algorithm Application and Framework Testing

8.2 Introduction

The following chapter discusses adding the last node to the framework to allow for optimisation between different iterations and benefit from the ANN application on a larger scale than the training input datasets.

The literature has shown the influence of applying genetic algorithms (GA) in solving design problems, especially with finding an optimal solution for environmental performance. GA is looking for optimal performing solutions in a pool of iterations without running the whole available iterations. It is known as the mathematical way of mimicking the evolution theory. This might be the reason some of the literature name it the “evolutionary algorithm”. A simple definition of GA can be to mix the different parameters or chromosomes (for example, x, y) to create different solutions to a problem (for example, xxyy or xxxy). Then, it evaluates the performance of these generated solutions compared to the set or desired solution output. The solutions that are closer to the desired performance or the optimal performance than the rest of the generated solutions form the start of the following step of this process, which is called the “generation”. The number of solutions in each generation is mainly called the “population” of this generation. This process continues until the GA finds an optimal solution or it reaches a set threshold for the generation’s number. The mixing of these parameters is conducted through two methods. The first method is “cross over”. This is the simple way of mixing or mating the different parameters with each other to create a different solution for the set problem. The other method is called “mutation”. This is related more to the change of the chromosome itself from its original status, so it does not remain in its original input status through the process of multiple generations (Whitley 1994; Winston 2010).

There are different ways to implement the GA principles in problem-solving and optimisation in general. This was interpreted by different tools and interfaces hosted on the Grasshopper platform. The literature has shown different applications for different tools that apply GA to optimisation in enhancing the environmental performance for the built environment (Rutten 2013; University of Applied Arts Vienna and Bollinger+Grohmann Engineers 2014). These tools widened the reach of this method by being available in Grasshopper as it is widely used for built environment modelling and performance analysis.

8.3 Genetic algorithm application and results

After developing the ANN capable of utilizing the classified database to predict the performance for both the building and urban scale, GA is used in this framework to conduct the optimisation part to look for the optimally performing urban geometry based on the prediction inputs coming from the ANN output. The criteria chosen to determine this optimal solution was solar radiation performance for the urban scale geometry and the Floor Area Ratio (FAR) of these configurations. The two factors should be conflicting as the test site (Aswan, Egypt) falls in a hot arid zone. This causes the solar radiation to be higher with higher buildings that need to have a high FAR value. This is why the optimisation problem is defined as a multi-objective optimisation. The tool used to conduct this optimisation in the framework is called "Biomorpher" (Harding 2017). Biomorpher is a tool that applies a method of GA that is named a cluster-oriented genetic algorithm. The major difference of this method is its capability of approximating the available pool of iterations into regions to find the closest optimal region instead of running the GA evaluation on each individual in its population. This helps save time for the generation (Bonham and Parmee 2004). Biomorpher has been embedded into the framework to implement the GA optimisation because of its capability of controlling the process and visualizing the options. Additionally, the classifying approach of the GA makes the implementation computationally efficient.

8.3.1 Initial framework test

The first test to implement the GA was set to have a 1,000 pool of iterations as the limit for the testing. It aimed to reach the largest possible FAR from these iterations with the lowest solar radiation possible. These two values were input from the framework's different classes. The FAR for each configuration was calculated by extracting the buildable area and multiplying it by the number of buildings. Each building had a floor height equal to 3 metres high. Then the sum of this process is divided by the total area of the buildable plots in the configuration at hand. The performance prediction is calculated by running the buildings' classification tags through the lookup process to find similar tags to extract their performance from the database. Then the rest of the unique tags with no similarities are predicted by the ANN class of the framework. The sum of the two values is the prediction fed to the GA component to act as fitness for the generation process.

The other inputs to Biomorpher's GA component are the slider responsible for controlling the iteration process as a "genome" and the geometry of the urban configuration itself after converting it to a "mesh" geometry. The genome is the name used to refer to the identifying features of each tested iteration. The mesh is one of the geometrical modelling representations used by the software platform.

The first trial was to test the use of Biomorpher, finding an optimal solution within 1,000 urban configurations. This sample of 1,000 urban configurations was part of the database for this stage of testing. The generation settings avoided all mutations, and the crossover was set to be 10% of the population. A 100 population for each generation was set for this phase of the testing. This test aimed to find the result of using Biomorpher as a typical GA application where the generation and offspring selection is conducted automatically based on the fitness preference. As mentioned earlier, the fitnesses were set to reduce the solar radiation while achieving the maximum FAR possible.

The test has aimed to get only five successive generations to test the time consumption too. The whole GA run took around 500 minutes. This test result came up with a high value of the FAR within the highest 15% of the iteration pool. However, the solar radiation was not within the same ratio as the lowest solar radiation saved results of the iteration pool.

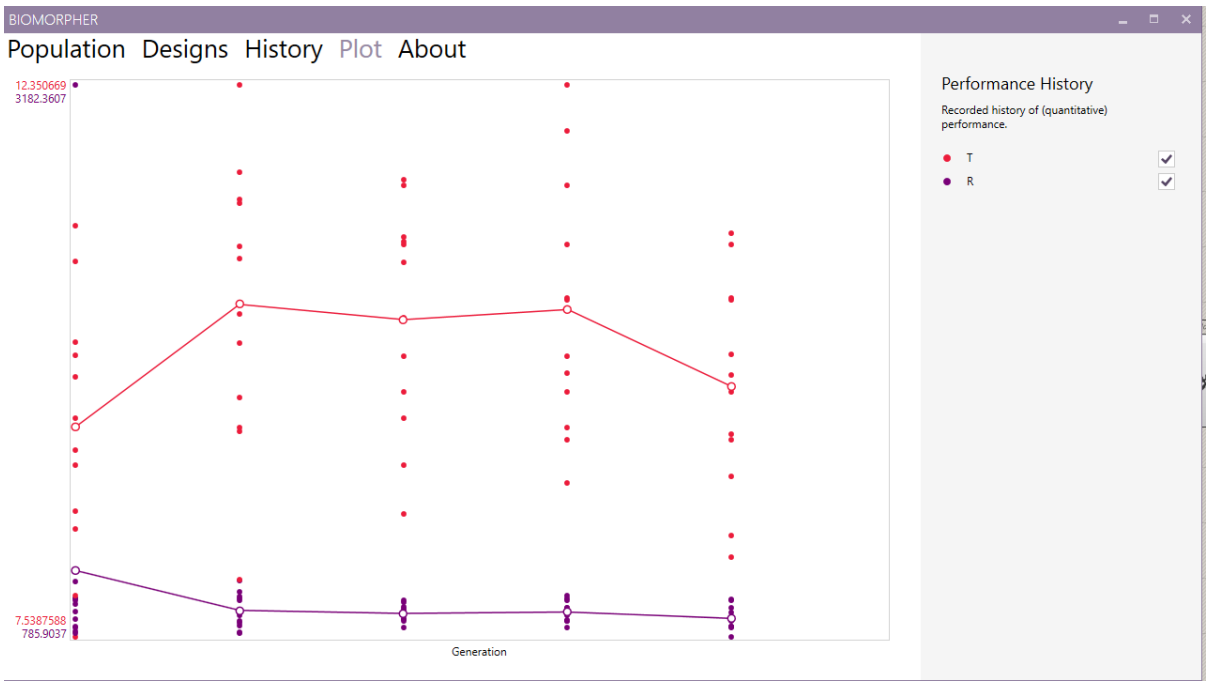


Figure 8-1 The generation plot of the two fitnesses in the first GA testing

However, this run has shown the capabilities of the tool. The sequence of development through the five generations is shown in Figure 8-1 for both the fitnesses. It shows the continuous reduction of the solar radiation fitness in blue. Furthermore, it shows the increase of the FAR value, in red, then it declined for the 5th generation, which was led by the prioritization of lower solar radiation value.

This process gave a clear understanding of the performance of the tool. Another trial was conducted with the same settings.

This time the generation was done by selecting the optimal clusters for each generation to create the offspring of the following generation. This test was done for a greater number of generations and consequently for a longer time. It had ten generations, and in each generation, the largest FAR and the lowest solar radiation value clusters were selected to continue the process. It took around 12 hours to finish this trial. The final 12 clusters of the last generation of this test are shown in Figure 8-2; each geometry is basically a representative of a cluster.

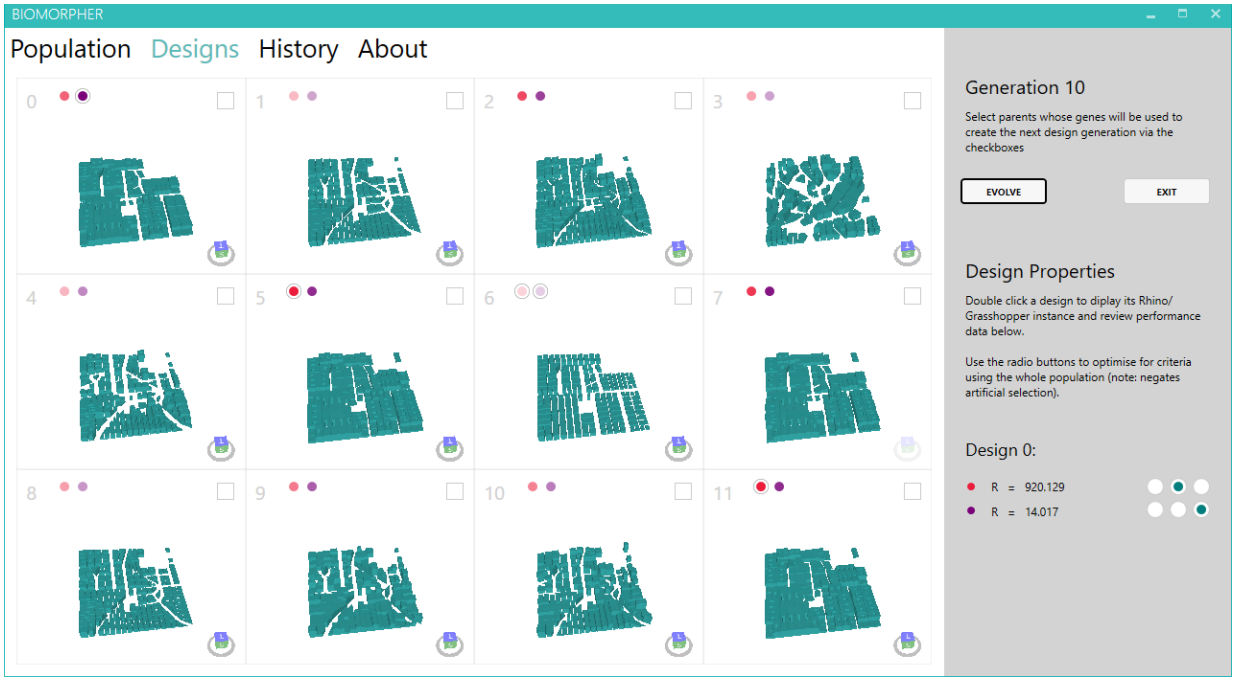


Figure 8-2 The last 12 clusters of the GA 10th generation for the 1,000 iteration pool test

The representative iteration with the highest FAR value was shown in Figure 8-2: cluster 0. It had a higher value of FAR of 14.01707 with solar radiation of 900.5328 kWh/m^2. As shown in Table 8-1, this iteration falls in the four highest FAR values in the whole pool of 1,000 iterations. One of the higher iterations with a higher value of FAR has a higher value of solar

radiation. This excludes it from being an optimal choice for the two performing aspects. Figure 8-3 shows the visual difference between the GA found result and the top two optimal choices with priority to FAR values over the 1,000 results pool.

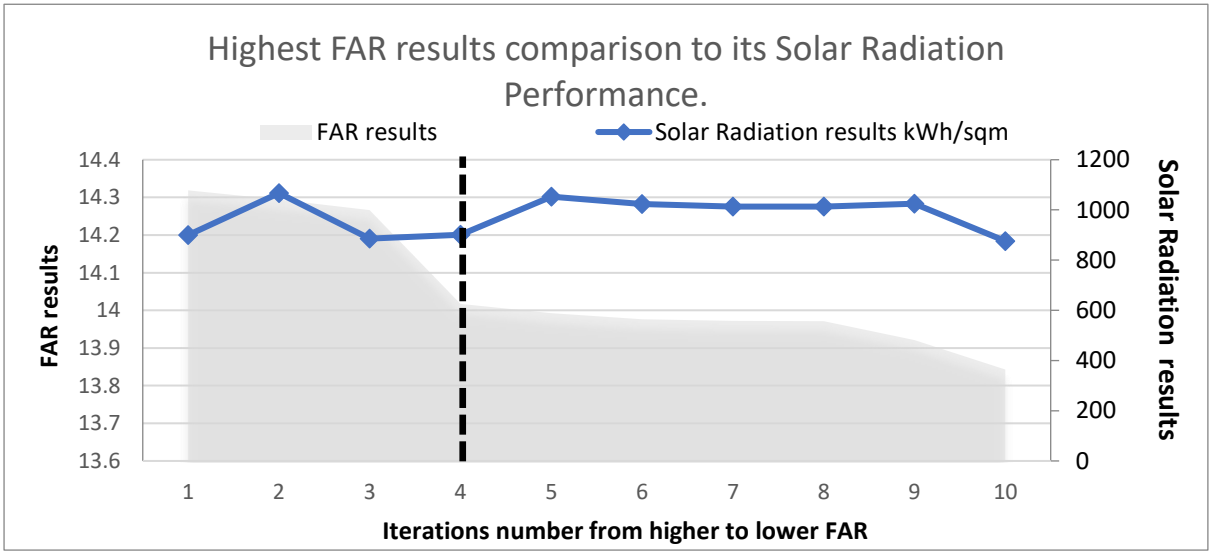


Figure 8-3 Highest 10 FAR values with their solar radiation results and the GA found iteration (in dashed line)

Table 8-1 showing the highest nine FAR values in the 1,000 pool of iterations with the GA chosen iteration highlighted in blue

Configuration ID	Urban Void Case	Random Angle Case	Maximum Arm Case	Street Width Case	Street Setback Case	Side & Back Setback Case	Building Court Case	Highest Distribution Case	Solar Radiation Results in kWh/m^2	FAR Results
4,485	1	0	0	2	0	0	1	0	900.2714	14.31869
4,050	0	3	1	0	0	0	0	0	1067.043	14.29413
4,320	0	3	2	2	0	0	0	0	885.4684	14.26613
4,482	1	0	0	2	0	0	0	0	900.5329	14.01707
4,068	0	3	1	0	1	0	0	0	1052.692	13.99231
4,212	0	3	2	0	0	0	0	0	1024.055	13.97646
4,698	1	0	2	0	0	0	0	0	1013.996	13.97238
4,860	1	0	2	0	0	0	0	0	1013.889	13.97159
4,377	1	0	0	0	0	0	1	0	1024.955	13.9209
4,503	1	0	0	2	1	0	1	0	876.2165	13.84315

8.3.2 Second framework test with 12,000 available iterations

The next test followed the same procedure to find an optimal solution within a larger pool of iterations. In this test, the pool of iterations was half of the available iterations by the generation class. The optimal solution chosen from this test took 13 generations with an hour for each generation, and the setting was 50 population per generation with no mutation and 10% crossover.

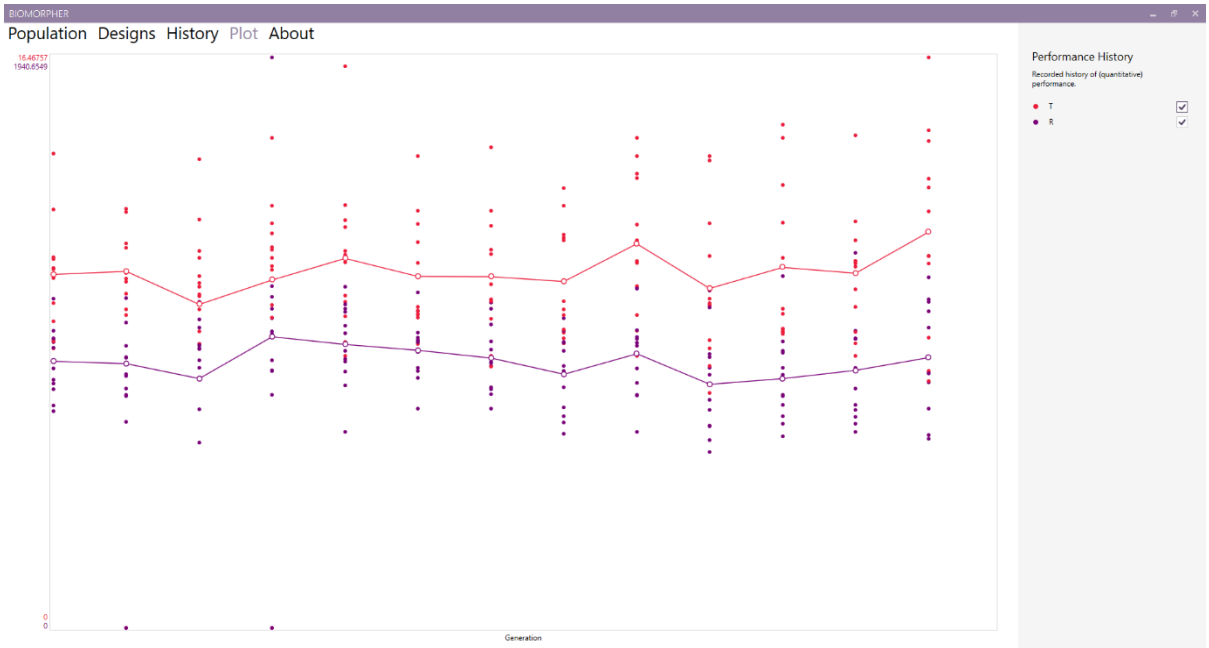


Figure 8-5 Second test results' plot with 13 generations.

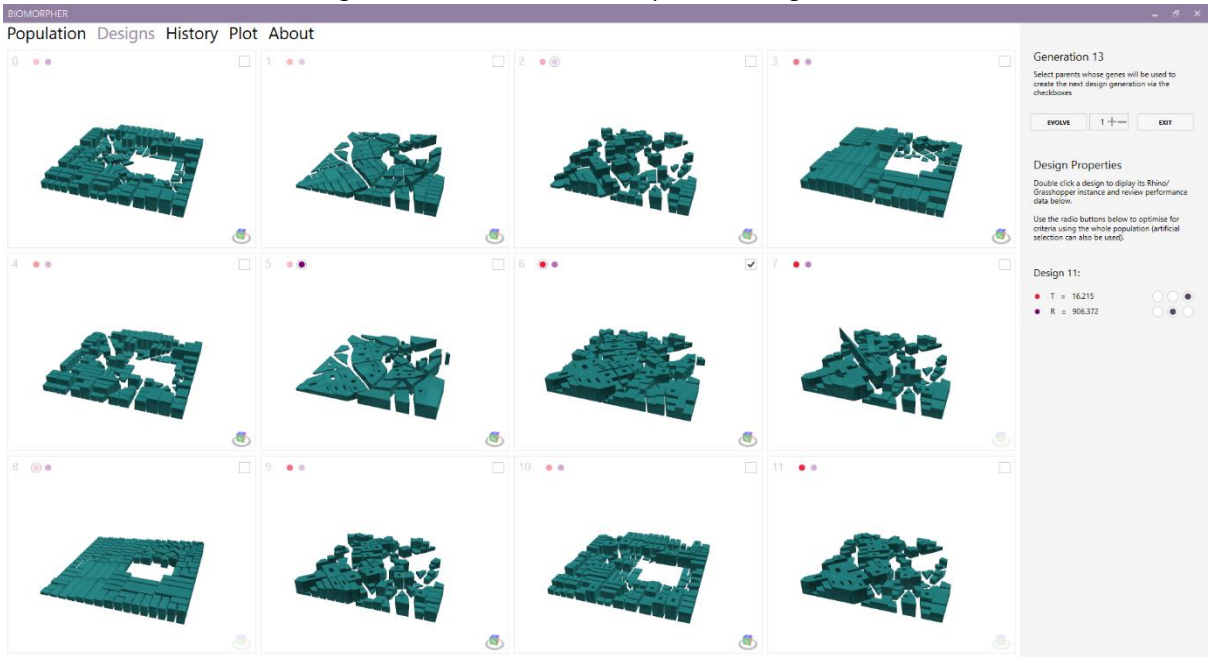


Figure 8-4 The last 12 clusters of the GA 13th generation for the 12,000 iteration pool test

The plot of the GA process shows fluctuating behaviour in looking into the design space of the available iterations. The gap between the two fitnesses throughout the plot shows the

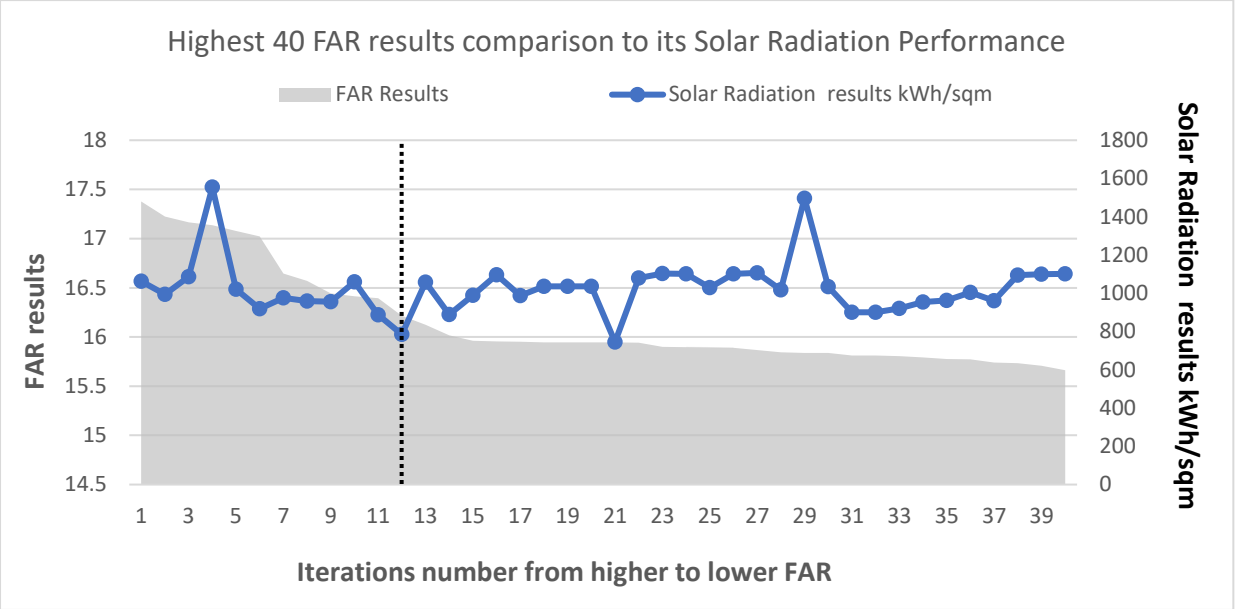


Figure 8-6 Highest 40 FAR values with its solar radiation results and the GA found iteration (in dashed line)

progress the GA is trying to make by looking for the highest FAR, in red, and the lowest solar radiation result, in blue. Although the FAR reached a peak in the 9th generation, yet the relook of the next generation's clusters resulted in a slightly lower solar radiation option by the 13th generation, as shown by the difference in the gap between the two fitnesses in Figure 8-5.

Another note on the results of this test is that the highest FAR representative cluster does not have the best performing solar radiation. The higher FAR cluster had solar radiation of 1294.7 kWh/m^2 with a FAR of 16.64; this was cluster number 6 within the last generated 12 clusters, while cluster 11 had better performance for both fitnesses, not just FAR. The representative cluster 11 had a significantly lower solar radiation of 906.3 kWh/m^{2,} while the FAR value is also lower with a slight difference. This representative had a 16.2 FAR value, which is not that different from the highest FAR achieved by this generation. This was the cluster with the number 11, as shown in Figure 8-4

When it came to comparing this result with the saved simulation result, this urban configuration had an optimal performance considering the two fitness performances. The list of the saved performances and FAR values was sorted in order from larger to smaller based on the FAR values. This is done to prioritize the FAR value for neighbourhood design which is a usual constraint for the designer to achieve higher FAR, allowing for more economic value for the land use. As shown in Figure 8-6, the selected representative cluster achieved the lowest solar radiation result within the highest 40 FAR values in the pool of iterations for this

test, marked with a dashed line. In contrast, its FAR value is the 12th highest value in a pool of more than 12,000 iterations. The following higher FAR values in the graph do not have lower solar radiation than the selected representative cluster. **Error! Reference source not found.** shows the top 20 iterations with the highest FAR values in the saved pool of iterations. It is important to note a difference in values between the GA selection of optimal performing solar radiation predictions and the saved database of solar radiation simulation results. This is due to the difference between ANN predictions and the actual simulation result and the fact that the database of urban configuration performance results were saved out of simulations conducted over the whole configuration, while the prediction results are the sum of the predictions of each building in that configuration. Its number does not just select the configuration, but also the urban generation parameters are checked to be the same features for discussed configurations. Also

Table 8-2 The highest 20 FAR values in the save results

Order based on higher FAR	Urban Void Case	Random Angle Case	Maximum Arm Case	Street Width Case	Street Setback Case	Side& Back Setback Case	Building Court Case	Height Distribution Case	Solar Radiation Results in kWh/m^2	FAR Results
1	2	0	1	0	1	0	1	0	1,063.583	17.37746
2	2	0	1	1	1	0	1	0	994.0439	17.22447
3	2	0	1	0	0	0	1	0	1,085.866	17.16701
4	2	2	1	1	1	1	1	0	1,554.55	17.13628
5	2	0	1	1	0	0	1	0	1,020.48	17.07668
6	2	0	1	2	1	0	1	0	919.1503	17.0214
7	2	2	2	0	0	0	0	0	975.5042	16.64382
8	2	0	1	1	2	0	1	0	959.0805	16.57062
9	2	0	1	2	0	0	0	0	955.3986	16.44351
10	2	0	3	1	0	0	1	0	1,058.776	16.41231
11	2	0	1	2	2	0	1	0	886.6475	16.39362
12	2	2	2	2	0	0	0	0	785.624	16.216
13	2	0	1	0	1	0	0	0	1,057.446	16.12415
14	2	2	2	1	0	0	0	0	889.4906	16.0172
15	2	0	1	1	1	0	0	0	989.7215	15.96211
16	2	2	1	0	1	0	0	0	1,096.213	15.95576
17	2	0	3	2	0	0	1	0	987.7892	15.95179
18	2	0	2	1	0	0	1	0	1,036.448	15.94575
19	2	0	2	1	0	0	1	0	1,036.205	15.94575
20	2	0	2	1	0	0	1	0	1,036.851	15.94575

as shown in Figure 8-4, the limitation of the geometry generation tool produced some configurations that need to be deselected from the GA selection process. Furthermore, some configurations do not fully restrict the boundary rectangle due to some limitations in the package used in the geometry generation process.

The optimal solution for this test did not fully cover the rectangular boundary, not just due to the generation of urban voids but also due to some generation process limitations. This allowed for its high FAR value, although this did not count as a bad result for the buildings as every buildable area does not intersect with its surrounding ones, nor do the buildings have any unacceptable results regarding their extrusions. Nevertheless, the following test tried to find an optimal solution covering the whole area of the given boundary and committed to the generated urban voids.

8.3.3 Framework test for total available iterations

The last stage of this test was to test the optimisation capability within the boundaries of the whole pool of iterations allowing Biomorpher to look for an optimal solution within the large database of configurations. In this test, the Biomorpher setting was 100 populations for each generation, and the same settings as the previous test came to cross over, which was 10% and no mutation was permitted.

This larger number of populations made the generation take a longer time than previously. In this test, each generation took around one hour and 40 minutes. The 4th generation showed a near-optimal performing representative cluster in comparison with the previous test

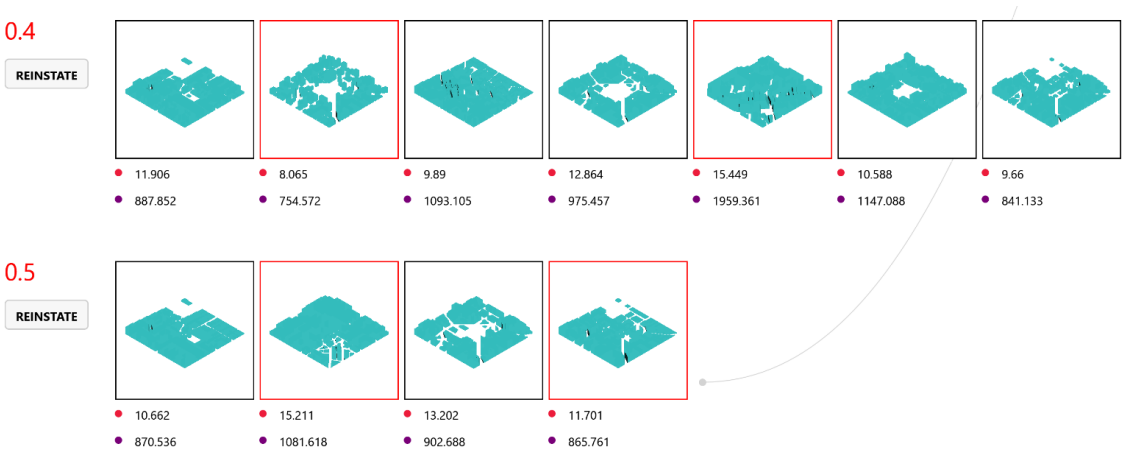


Figure 8-7 The representative clusters for the 4th and 5th generations of the full database test (red dots for FAR results and blue dots for solar radiation prediction in kWh/sqm)

results. This iteration has a solar radiation value of 1081.61 kWh/m^2 and a FAR value of 15.21, as shown in Figure 8-7.

Biomorpher did not find a better performing solution until generation 20. Figure 8-8 shows the representative clusters of the last four generations of the process and their results for

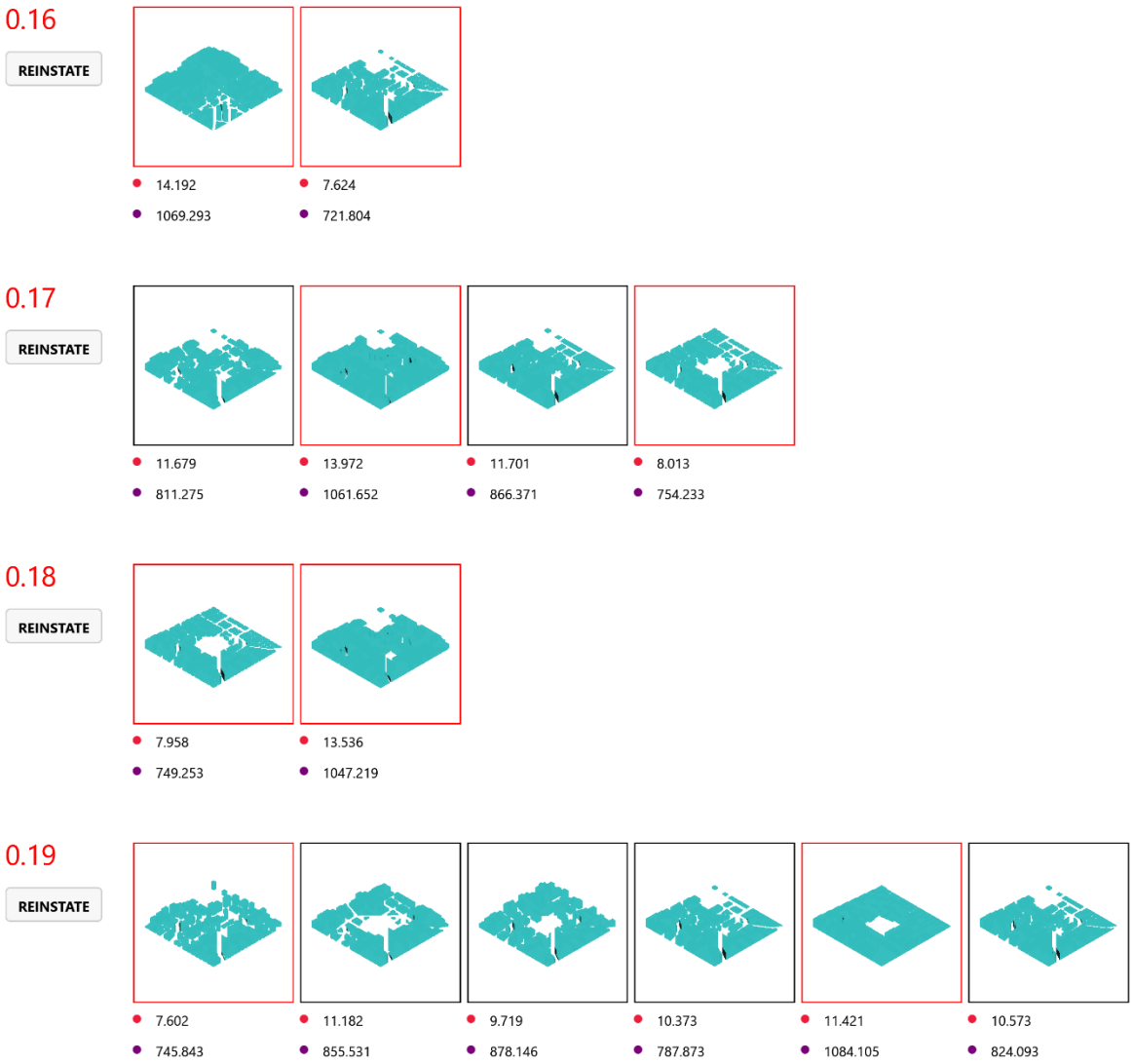


Figure 8-8 The last four generations in the last full database test (red dots for FAR results and blue dots for solar radiation prediction in kWh/sqm)

solar radiation in kWh/m^2 with a blue dot beside it and the FAR values of the iterations with a red dot beside it. For this reason, the test reinstated the generation 4 results and tried to drive the generation process towards a better performing option by only selecting the most optimal representative clusters. Reinstating means getting Biomorpher back to the stage of generating a certain generation. After this, the test continued for another four generations with this new selection of clusters. However, it has shown a slightly better performing

iteration. This iteration has a solar radiation value of 1204.8 kWh/m^2 and a FAR value of 15.84, as shown in Figure 8-9.

The final trial was to start a new GA test, but this time instead of selecting a random set of the population for the initial generation, it was set to start with this solution.

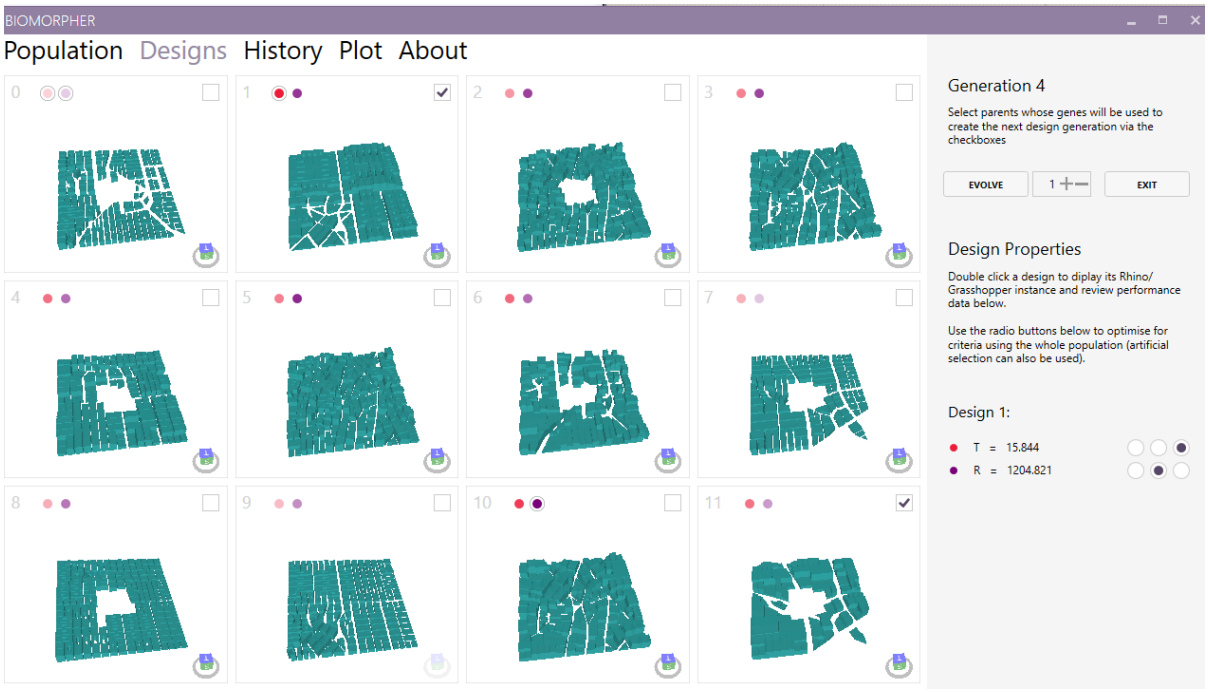


Figure 8-9 The representative clusters of the reinstating trial showing the optimal performing representative cluster in cluster 1

This time it took three generations to get a better performing iteration. It is important to know that these three generations did not try to look far from the starting point of the first

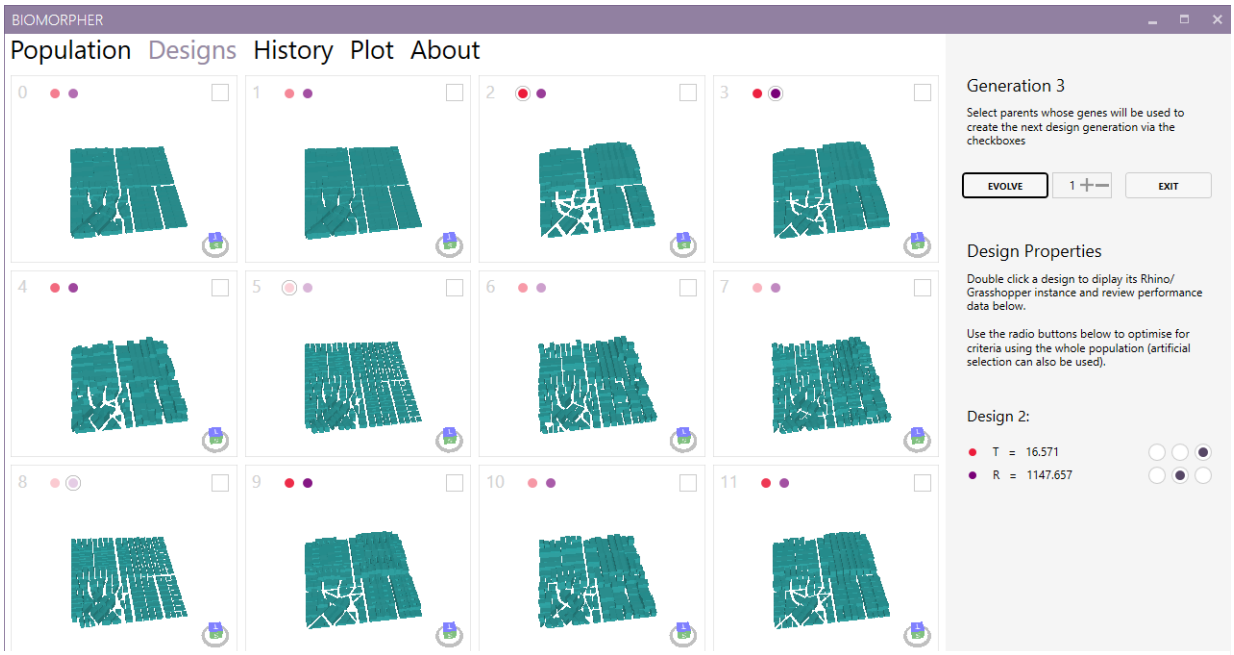


Figure 8-10 The third generation of the directed retest for the whole database test

set of populations. This might raise queries about the tool's capability to examine the selected iterations and the diversity of its generation process if it only focuses on iterations with relative settings. The third generation had a cluster with a solar radiation value of 1147.65 kWh/m^2 and a FAR value of 16.57, as shown in Figure 8-10.

This phase of testing has shown the importance of the user interaction with the GA generation stage of the framework. As shown in the multiple trials, the test took different directions to look for better performing iterations while reducing the time consumed by directing the generation breeding process, which is a rare feature in the available tools applying GA principles in the same platform. Moreover, it has shown a limitation of the tool. When starting the proceedings with a given set of iterations, it does not look further within the pool of available iterations. It is important to note that this generation process was done without reviewing the simulations in the saved database of urban configurations. After finishing this optimisation process, the comparison was made as done in previous tests with the saved simulation results to get the location of the GA optimal solution against the optimal solution of the saved results.

After these different trials, the GA pointed out the configuration with the 14th highest FAR value. This is the 5th lowest solar radiation in the highest 20 FAR values in the pool of 23,328 urban geometrical iterations. The highest 20 FAR iteration results are shown in Table 8-3, with the GA selected iteration highlighted in colour. Its order as the 5th optimal solution can be seen more clearly in Figure 8-11. This result shows the potential of using this framework to provide guidance through the early stages of urban design, being capable of reaching this efficient level of accuracy considering the time saved from simulating the total number of iterations to reach the optimal solution.

The following step of testing the framework was to utilize this set of classifications, prediction and optimisation on an existing neighbourhood boundary and test its effectiveness of being integrated to optimize an already designed neighbourhood. This test will be discussed in the following section.

Table 8-3 The highest 20 FAR values in the save results of the total database with the pointed out iteration in blue

Order based on higher FAR	Urban Void Case	Random Angle Case	Maximum Arm Case	Street Width Case	Street Setback Case	Side& Back Setback Case	Building Court Case	Highest Distribution Case	Solar Radiation Results in kWh/m^2	FAR Results
1	5	2	1	0	0	0	0	0	1,048.129	17.60369
2	3	2	1	1	0	0	0	0	1,013.694	17.5942
3	2	0	1	0	1	0	1	0	1,063.583	17.37746
4	2	0	1	1	1	0	1	0	994.0439	17.22447
5	2	0	1	0	0	0	1	0	1,085.866	17.16701
6	2	2	1	1	1	1	1	0	1,554.55	17.13628
7	2	0	1	1	0	0	1	0	1,020.48	17.07668
8	2	0	1	2	1	0	1	0	919.1503	17.0214
9	5	2	1	2	0	2	0	0	1,046.329	17.01338
10	0	2	1	1	0	0	0	0	1,009.887	16.95517
11	5	2	1	2	1	0	0	0	922.0213	16.77168
12	4	2	1	2	1	0	0	0	932.8184	16.71138
13	2	2	2	0	0	0	0	0	975.5042	16.64382
14	2	0	1	1	2	0	1	0	977.3871	16.56767
15	2	2	2	2	1	0	0	0	778.3127	16.55688
16	2	0	1	2	0	0	0	0	955.3986	16.44351
17	2	0	0	1	0	0	1	0	1,080.064	16.4123
18	2	0	1	2	2	0	1	0	886.6475	16.39362
19	0	2	1	1	0	1	0	1	1,157.974	16.30851
20	2	2	2	2	0	0	0	0	785.624	16.216

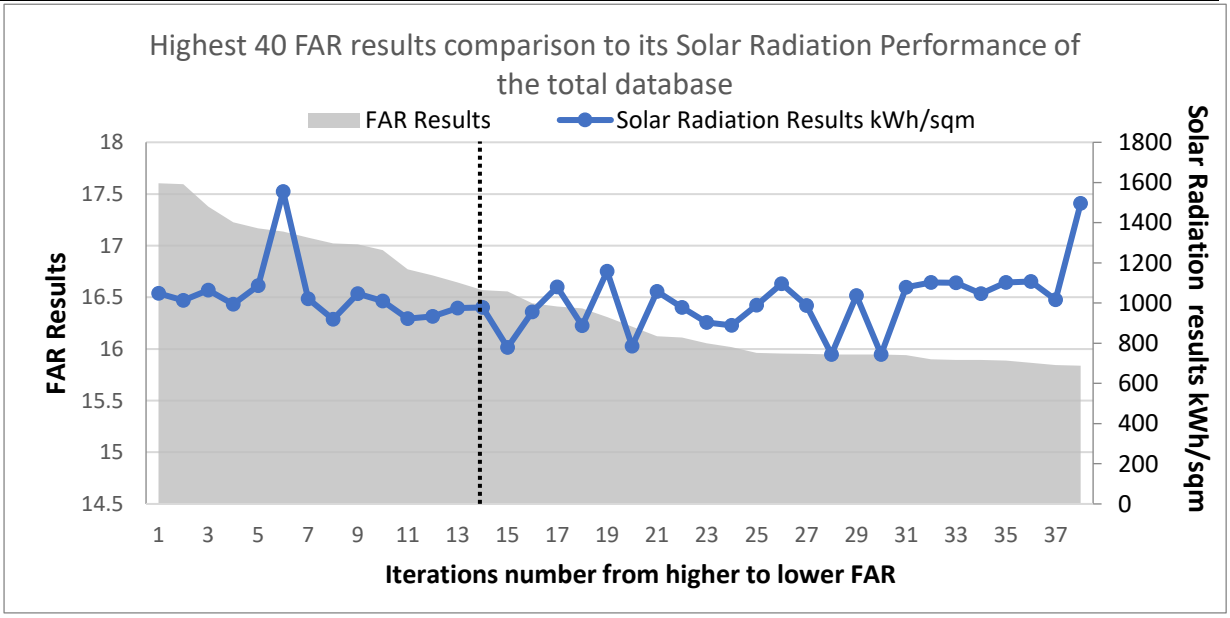


Figure 8-11 Highest 40 FAR values with their solar radiation results from the total database and the GA found iteration (in dashed line)

8.4 Framework testing on existing neighbourhood

The testing in this phase aimed to investigate the different phases of the framework. Each phase of the framework was tested by applying a new boundary to the generation process, generating new classified geometries, predicting the solar radiation for the newly generated urban iterations, optimizing the pool iterations and comparing the optimal solution with the actual case study status. This test will highlight the strengths and weaknesses of the framework at its current status as a proof of concept, and it provides a clear path for the future work of this research.

8.4.1 Case study location

The location of an existing case study selection was made based on the weather file used to build the training database. The location of the testing neighbourhood boundary was selected from the city of New Aswan in southern Egypt. This is one of the new cities commissioned by the government to accommodate the Egyptian population growth. Being a twin city and an extension to the original city, it falls within the same climate and weather conditions. The total planned area of the city is 91.532112 km^2. It targets 850,000 inhabitants by 2023 (New Urban Communities Authority at The Ministry of Housing. Utilities & Urban Communities [no date]). The residential sector has different types and classes of housing among other different land-use types in the city prospectus land use map (see Figure 8-13). The existing status of

Figure 8-12 The current status of New Aswan, Egypt (Gorelick et al. 2017)

the city is as shown in Figure 8-12. It shows that the city is still under ongoing construction. The satellite photos were taken from the Google Earth tool (Gorelick et al. 2017).

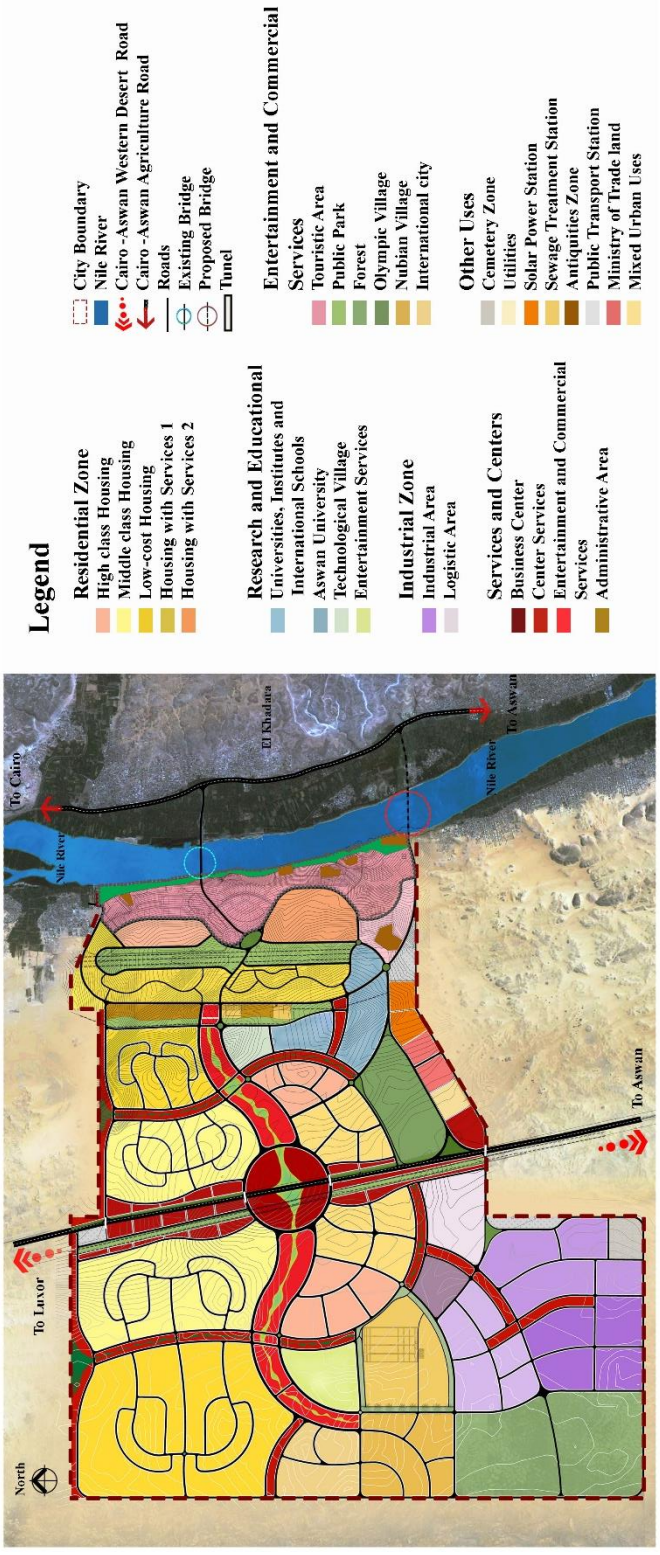


Figure 8-13 New Aswan land use design with legend edited by the researcher (New Urban Communities Authority at The Ministry of Housing, Utilities & Urban Communities. [no date]).

The neighbourhood case study is located in the built part of the city. The location of the

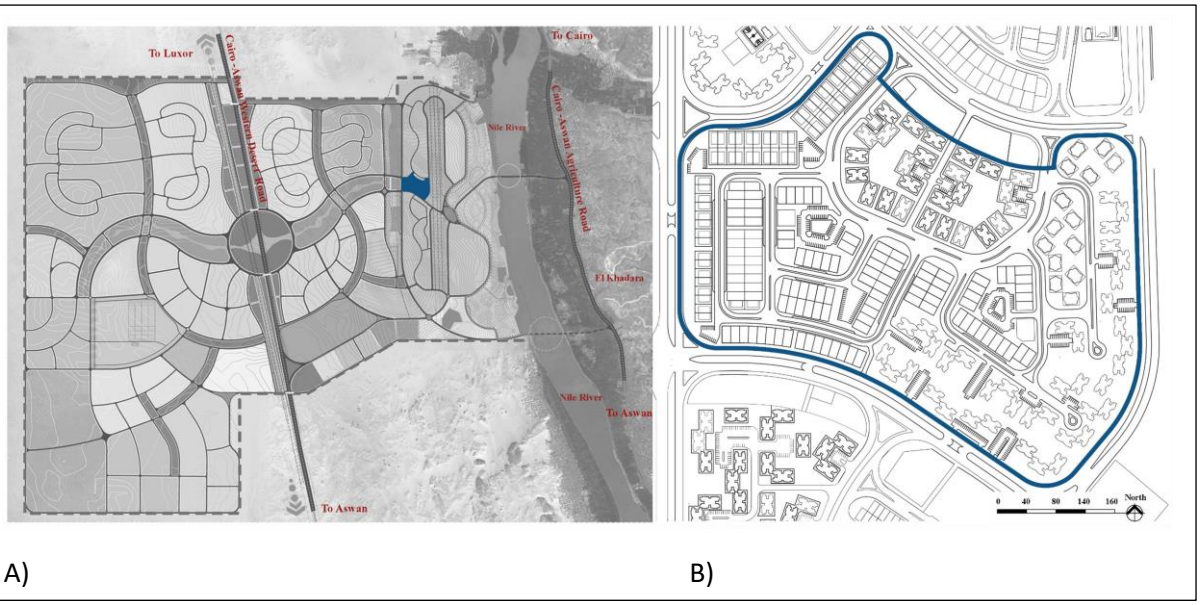


Figure 8-14 A) case study neighbourhood location in the city land use map. B) detailed urban design of the selected neighbourhood and boundary highlighted in colour edited by the researcher. (New Urban Communities Authority at The Ministry of Housing. Utilities & Urban Communities. [no date])

selected neighbourhood and its detailed urban design plan with building plots are shown in Figure 8-14. The heights of the buildings were approximated from the Google Earth images of the site. This is shown in the satellite image of the site (see Figure 8-15). Heights were

Figure 8-15 Selected neighbourhood current status (Gorelick et al. 2017)

approximated from the number of floors shown in this image to model the existing status of the neighbourhood.

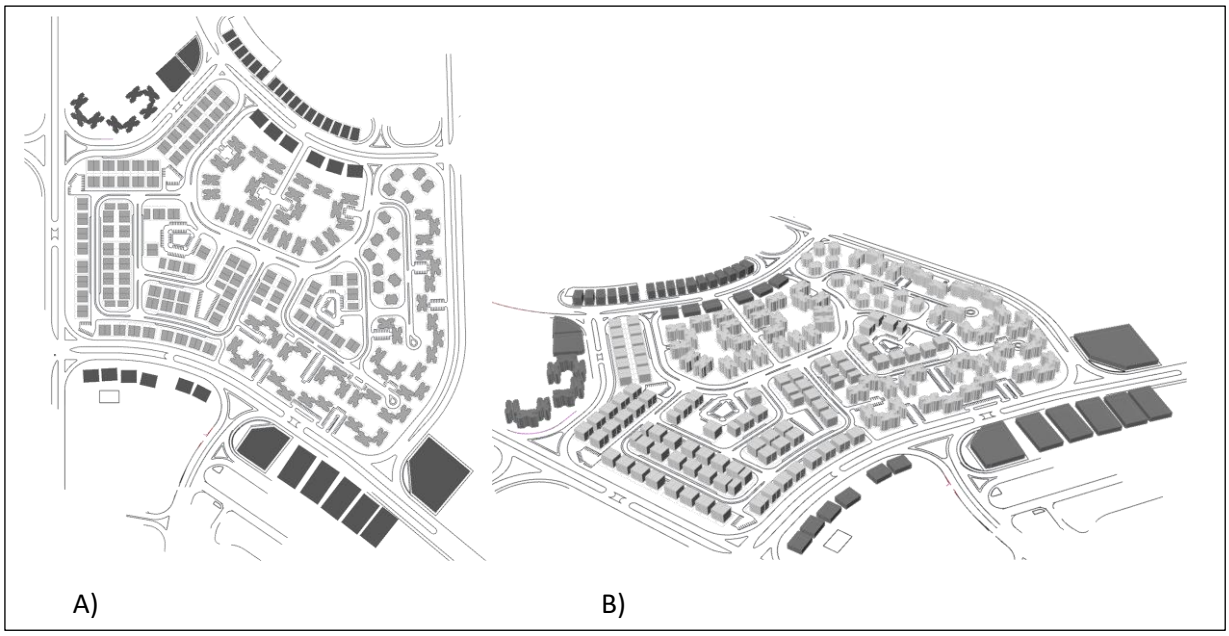


Figure 8-16 Model of the designed status of the neighbourhood A) the top view of the existing status of the neighbourhood. B) perspective view for the model.

This helped create a model for the existing status of the neighbourhood (see Figure 8-16). The selected neighbourhood has an area of 237,866.71 $m^{2,}$ with 60% of this area dedicated to open spaces. The buildings in this neighbourhood came with two heights. Two typologies of the buildings had a four-floor height, and the other two typologies had a five-floor height. Assuming that one floor was 3.5 metres high based on the literature, these heights could be assumed to be 14 and 17.5 metres high for each group of typologies (Attia and Evrard 2013;

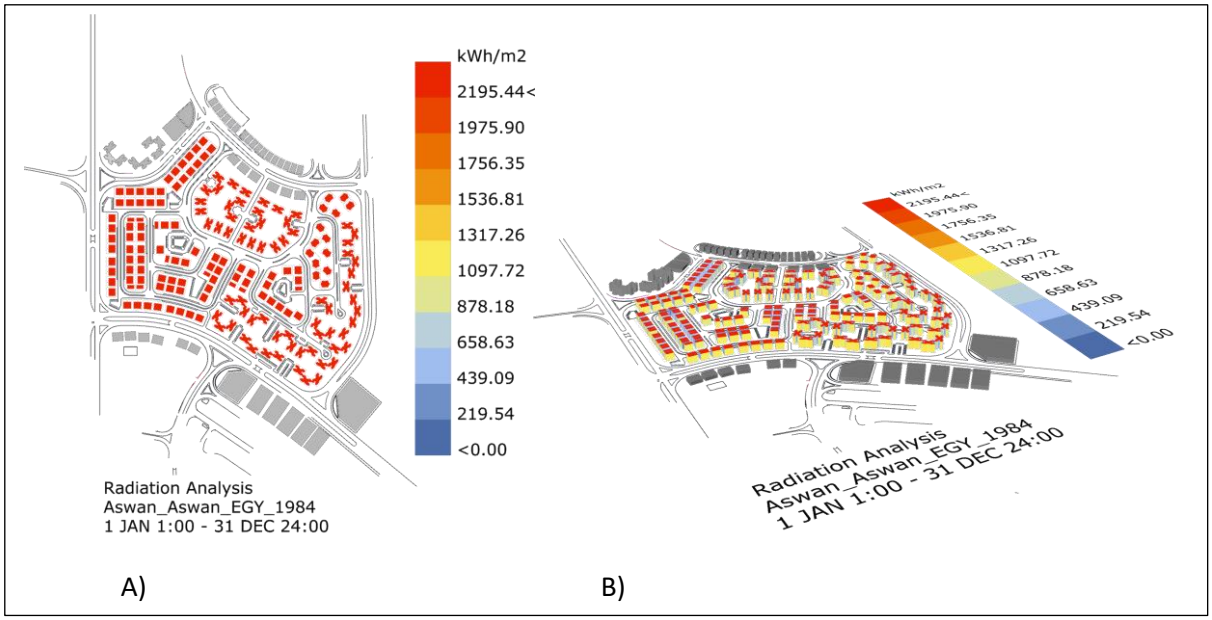


Figure 8-17 Model of simulation results for selected neighbourhood A) the top view of simulation results the existing status of the neighbourhood. B) perspective view for the model results.

Sabry et al. 2014). This low density of built-up areas and low-rise buildings resulted in a low value of FAR. As shown in Figure 8-17, This neighbourhood has a 1.27 FAR value. The total built-up area for all the buildings calculated from the model was 51,339.91 m^2. The total number of building prototypes was 256 buildings in the model. The model had a simulation of its solar radiation to act as a benchmark for the framework results. This simulation of solar radiation result was 224.8 kWh/m^2.

8.4.2 Framework generation testing for the existing case study

The boundary and the existing context of the selected neighbourhood were used as a fixed input to the framework. Another input consisted of the three main streets entering the neighbourhood. The streets were input as lines to be a starting point of the street generation process of the decoding spaces component. The generation process needed some editing to get the building footprints and urban voids closer to the model of the existing status of the neighbourhood. This included the changing of the urban voids percentage to match the one of the existing model. Another edit was to remove the entire built-up neighbourhood from the options of urban voids for the same reason. Another adjustment was to reduce the minimum distance between the streets to comply with the model distances. This changed from 50 metres in the previous studies to be 30 metres in this test. Also the built-up area width was reduced from 30 to 20 metres to allow for smaller building plots to be created and compared with the model of the existing case study. The classification of urban geometry went with no obstacles as the framework has followed its normal data flow.

8.4.3 ANN testing for the existing case study

The framework prediction was tested by running a solar radiation simulation for 1,000 random urban configurations and comparing its results with the framework prediction results for the same configurations. The results have shown a high correlation between the two results (see Figure 8-18). It had a 94% R^2 value for the correlation, which is a significant positive correlation noting that the training data of this ANN node is generated from different settings, neighbourhood boundary and context conditions. The error was decreased in this prediction test due to the reduced number of buildings generated for each configuration after

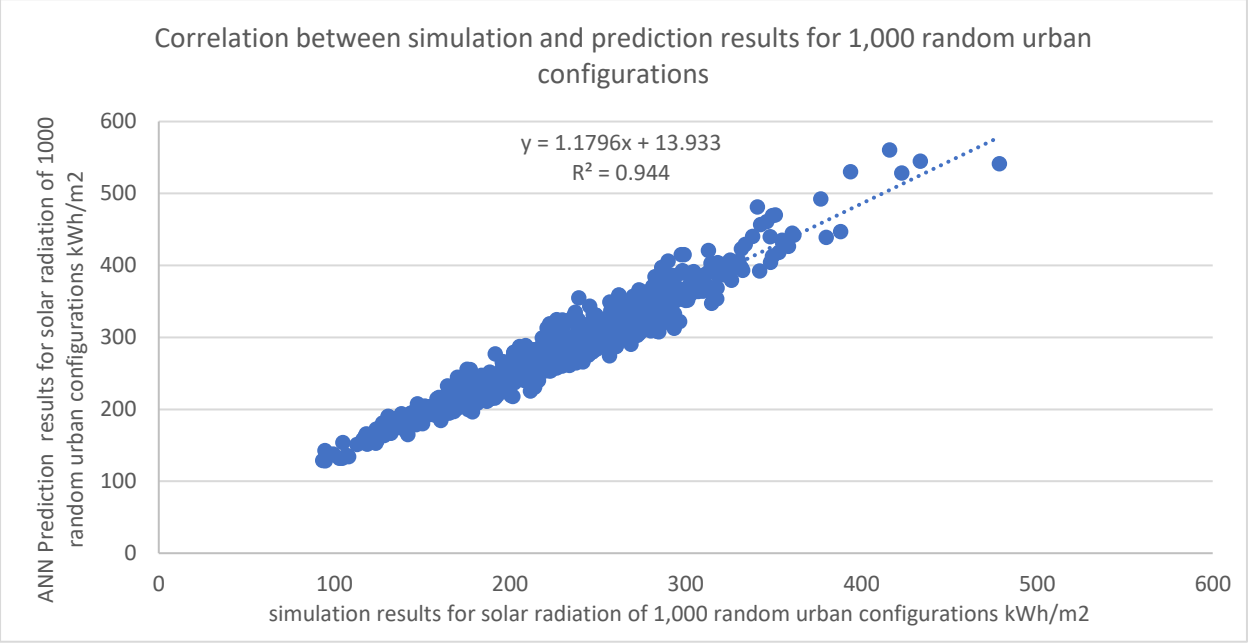


Figure 8-18 Framework prediction testing results

increasing the urban void percentage. This led to a reduction of the error aggregation in the addition of tags prediction.

8.4.4 GA testing for the exiting case study

The prediction test result led to the continuation of testing for the final stage of the framework. GA optimisation using Biomorpher was conducted on the whole pool of iterations using the simulated model as a threshold for the framework to achieve or enhance. The same procedure of testing the GA application previously was set to be conducted in this testing

Figure 8-19 Existing case study GA optimization plot

phase. The time consumption took around 52 minutes for each generation. This was done for six generations. At the 6th generation, there was more than one cluster that had better performing iterations for solar radiation results and FAR values. The plot of these six generations shows the development of the GA optimisation for the two fitnesses. It shows that the solar radiation values are decreasing throughout the six generations of this test, shown by the blue line, while the FAR was fluctuating in the red line (see Figure 8-19). Two representative clusters were performing better for both fitnesses. A solar radiation simulation

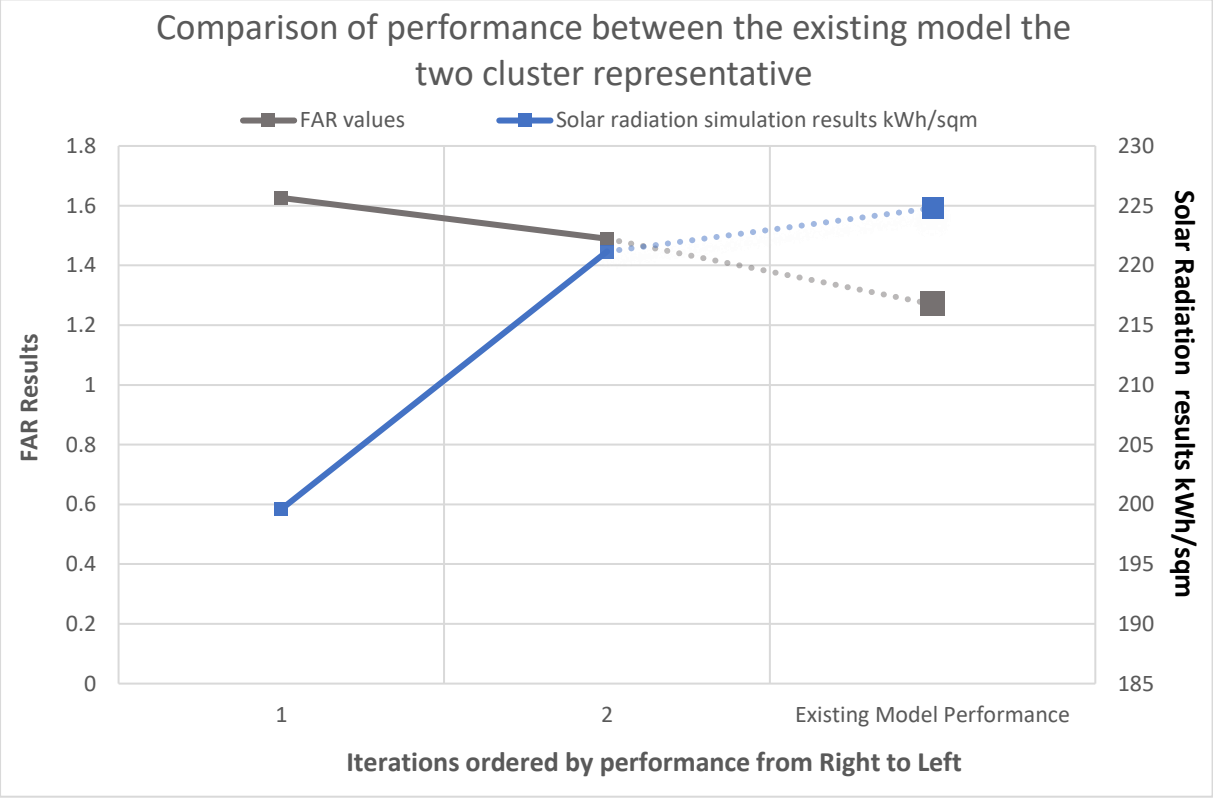


Figure 8-20 Comparison of the fittest 2 representative clusters with the existing model results

was conducted for these two cluster representatives to get their actual simulation results. The simulation results are compared with the existing model results in Figure 8-20 and the FAR results. It can be seen that the performance for the two fitnesses are better performing for the two GA selected iterations.

The final models of the two iterations are shown in Figure 8-21. It shows the top view and the perspectives of both iterations and the visualized results of the simulation. Iteration 2 had a FAR value of 1.4 and solar radiation of 221.2 kWh/m^2. It has 108 buildings with a total built area of 64,295.4 m^2. Iteration 1 had better-performing fitnesses. It had lower solar radiation with only 200 kWh/m^2 and a 1.6 FAR value. It had fewer numbers of buildings with only 98 buildings and a total built-up area of 55,128.4 m^2.

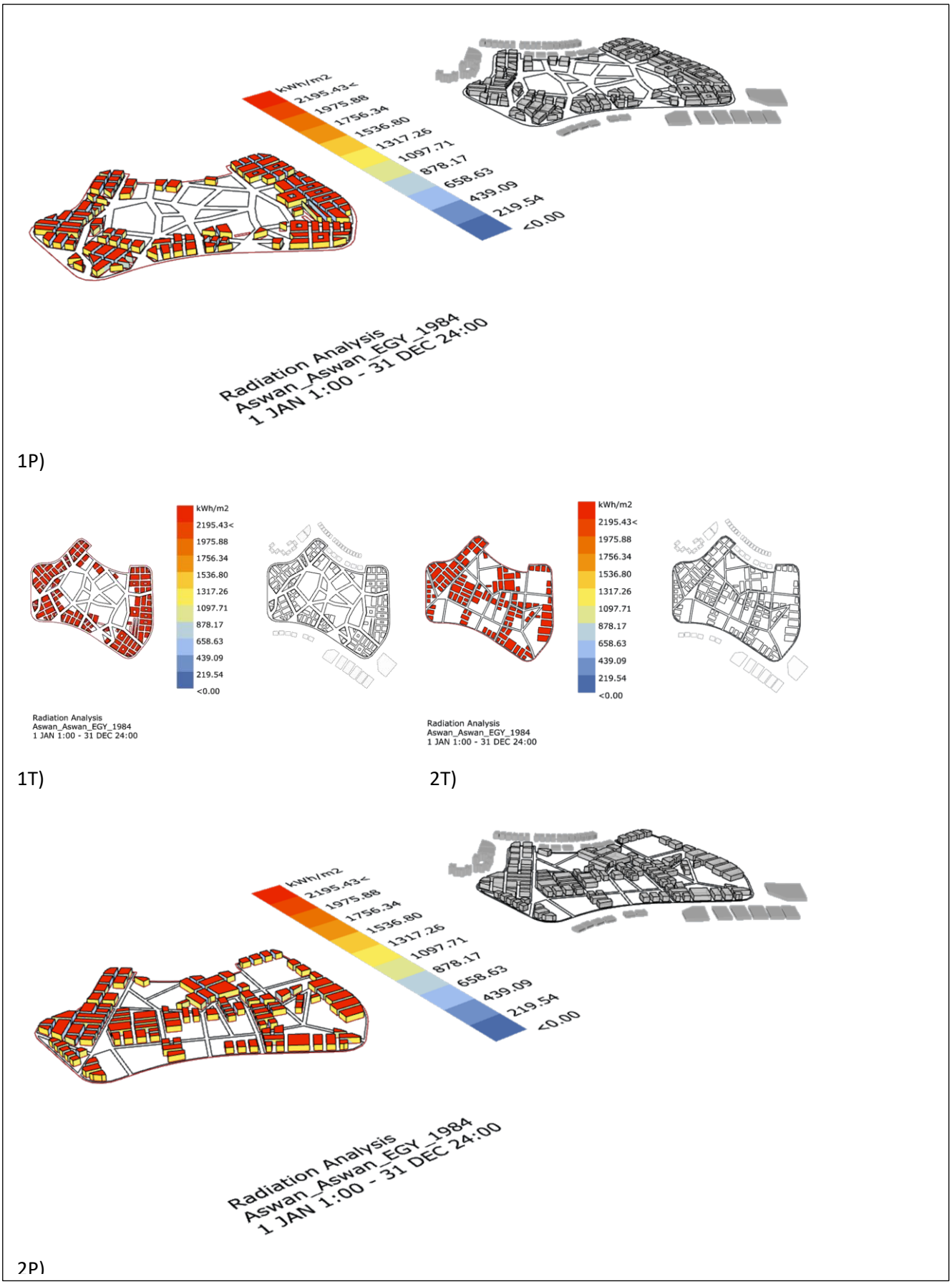


Figure 8-21 Two GA selected models and simulation visualization results shown in perspectives (P)and top views (T) for iterations 1 and 2

Although both models had different settings regarding the urban void status, iteration 1 had

one central urban void space while option 2 had the randomly generated five urban voids scattered through the neighbourhood. The two provide further compactness than the existing model. Both of the GA models had fewer buildings with a larger built area, while the existing model determined the urban void percentage. Another note is that the typology of buildings in iteration 1 had courts in some buildings, which can be one reason for the lower solar radiation results. This highlights the importance of further investigation of the building typologies and their impact on this neighbourhood scale. The street network for the selected models has a different set of rules in its creation. Thus it resulted in the fragmented network shown for iteration one and the almost perpendicular one for iteration 2. This is another opportunity to develop the framework further to provide more defining parameters, shape grammar and classes that provide more control over the network, and test its impact when introducing some street network principles to the framework, like hierarchy. This optimisation test was done in two days when considering the time consumed to edit the framework setting to match the existing neighbourhood model. These two optimally performing iterations can guide further development in the later design stages, considering that this is set to be an early stage of the design optimisation framework.

This simple test shows the capability of the framework at its current status as a proof of concept. The framework at this stage provided a method of simplifying the geometry through classification, applying ANN to predict performance and GA for searching for a better performing alternative than the conventional design method.

8.5 Summary

This chapter discussed the last phase of the framework: to apply the last stage of optimising the geometry based on its solar radiation results and floor area ratios. The framework used a Grasshopper plug-in that applies Genetic Algorithm (GA) methods, Biomorpher, That allows the control of iteration process limiting the time consumed for optimisation stage. Moreover, it has a clustering nature due to its cluster-oriented genetic algorithm that enhanced its time consumption and computational cost.

The first stage of testing this application of GA principles within the framework was based on comparing its optimized results to the saved database of FAR and solar radiation results. This was for three sizes of the database. The first test was conducted with 1,000 available urban

configurations. Compared with the saved simulation database, the framework got the third optimal solution. At the same time, it ran for five generations taking around 4 hours using the ANN prediction with the Biomorpher clustering GA application. The following test was conducted with over half of the saved database. This test the importance of selecting the generation clusters feature in Biomorpher because of the need to eliminate the iterations that were not affected by the generation process limitations. The results of this test have reached the lowest possible solar radiation in the 20 highest FAR from the testing sample.

The final test of the database was on the whole saved urban configurations. This test has shown a limitation of the GA tool about its capability to overcome its local optimal solution and look further for better performing results with different settings. This limitation can be evaded by the user interaction and choice of alternatives. The final database testing phase has led to finding the 5^{th} optimal solution for higher FAR and lower solar radiation in a pool of 23,383 different urban configurations taking around 24 hours of running. This genetic algorithm runs with modification of cluster selection done by the researcher (see section 8.3.3).

Following these tests, the framework was utilized in an existing context. The selection of the existing neighbourhood was based on location in New Aswan, Egypt. This is one of the new Egyptian cities that was planned to host the expected urban growth. The neighbourhood design was modelled and simulated to get an insight into its density and solar radiation performance. The framework settings were edited to adapt to the existing case study geometry status. These edits included the change of available options of urban voids and the percentage of it.

Moreover, the street network initial start was an input to the framework and the existing urban context model and site boundary. The geometry classification and tag creation for the buildings did not have any challenges because the flow of the framework did not change. The prediction testing achieved a high value of correlation compared to simulated results of the generated sample of the new generated pool of iterations. The last phase of optimisation utilizing Biomorpher was done with the same settings used in the database testing phase. The testing went for six generations with six hours to reach the two iterations that had a better performance for solar radiation and FAR values for the same neighbourhood boundary and context. The framework has proved its capability of finding a better performing iteration in a

short time that is convenient to guide and inform the decision-making process for the early stages of design. However, there is room for further investigation regarding some aspects, like the inclusion of building typologies and street network hierarchy as framework parameters. In its status as a proof of concept, the framework has shown a significant time-saving capability along with accurate results in comparison with traditional methods of simulation. It has also shown its capability of classifying geometry on an urban scale, predicting its solar radiation performance and searching for optimally performing iterations in a large pool of iteration on a neighbourhood scale. This chapter has shown the potentiality of the methods used in this framework if implemented in further detailed neighbourhood scale models with even more performance aspects like energy demand and lighting availability, and outdoor thermal comfort, for example.

These outcomes will have a clear impact on the decision making process for similar weather conditions and globally, if modelled and trained, to further implement a data-driven approach in urban design for neighbourhood scale. This will benefit different stakeholders and decision-makers to have performance-based decisions in the early stages of design which will help guide climate-resilient neighbourhood designs. For example, taking the Egyptian case study, applying such a method in Egypt's early neighbourhood design stages will provide some assistance to meet the local government’s aims and promises to achieve the United Nations Sustainable Development Goals. (United Nations Development Program UNDP 2015; Sustainable Development Goals Knowledge Platform 2021). This framework will also assist private sector owners, and real estate developers comply with these government plans and what it can save regarding the post-occupancy costs.

On the one hand, Egyptian urbanised new cities and settlements are growing. It is important to highlight that there is a significant portion of this Egyptian urbanization owned and managed by the private sector. (Metwally and Soliman Abdalla 2013; Mohammed and Hammad 2019). While, prices for utilities, especially energy, are continuously rising. This means this urban growth has an economic challenge in addition to its environmental burden. All these factors are adding to the economic and environmental feasibility for novel performance-based design approaches provided by this framework at its final stage(Hongyun and Radwan 2021). Applying data-driven design solutions in early neighbourhood design stages allow different stakeholders to have a capable, clear and feasible design approach to

designing climate-resilient neighbourhoods. This research will guide urban design approaches and explain urban geometry dynamics to different designers, decision-makers, and policymakers. It will also interact with different design aspects like socio-spatial integration in design and informing participatory practices while informing the design stakeholders' agendas with quantitative numerical performance data.

9 Conclusion

9.1 Introduction

This chapter discusses the main findings and conclusions of this research. It also explores its innovative approach for performance-based design and optimisation and the possibility of its inclusion in the early stages of urban design, exemplified by the hot arid climate zone represented by Aswan, Egypt. It will also discuss the findings from testing this framework as a proof of concept and its capabilities and the opportunities for further development of this framework to be comprehensive and robust in guiding the neighbourhood early stage of design with performance-based data.

The study in hand determined the effect of urban geometry on urban performance, specifically urban direct solar radiation, in an algorithmic approach
The second aim of this study was to investigate the possibility of utilising this relationship in building a data-driven optimisation framework for the neighbourhood scale. To achieve this, the research investigated the geometrical impact on different aspects of performance in two different climatic conditions focusing on energy demand (see Chapter Four). The results of these preliminary studies and their analysis shaped the research scope towards creating a novel workflow that breaks urban neighbourhood models into a classified set of variables that relates to both building and urban levels (see Chapters Five & Six). The following was to implement ANN method on this geometry classification to predict its solar radiation and overcome the simulation computational and time costs (see Chapter Seven). The last part of this paper moved on to investigate and test the application of GA to optimise urban geometry based on its solar radiation results and FAR in Aswan, Egypt (see Chapter Eight).

9.2 The relative importance of geometrical variables on urban energy demand and solar radiation.

The literature has discussed the continuous development of computational modelling for the built environment and highlighted the complexity of creating urban models. Several studies have reviewed other aspects of urban geometry modelling and its impact on urban performance aspects using different scale models for simulation. Some of these studies have investigated the possibilities of utilising this impact in an optimisation process or method. As pointed by literature (see chapter two) , time consumption is also another challenge to search for performance-based design on an urban scale. This directed the research started with a

preliminary study that investigates the limitations of modelling and simulation state-of-the-art tools. Also, it aimed to achieve a better understanding of the relative importance of geometrical variables within an urban context on the performance outcomes.

This investigation used a simulation engine, named Ladybug tools, to run a sensitivity analysis on four geometric variables in an urban context using the parametric notion of Grasshopper platform and its VPL nature. The environmental performance aspects were chosen for this sensitivity analysis focused mainly on energy demand and, as shown in appendix A, solar radiation and daylighting availability. This was tested for one cooling-oriented weather file, Aswan, Egypt, and another heating-oriented one, London, UK. The test for Aswan was with materials based on literature settings then repeated with ASHRAE's recommended set of materials for the tested locations. The settings and results of these preliminary studies are shown and discussed in Chapter Four.

These sensitivity analyses clarified the tested variables' impact on energy demand for heating and cooling. This was calculated by grouping iterations based on its variables and measure the impact caused by the variation of each geometrical variable. It showed the order of this impact on energy demand between the tested variables for both tested locations. The order of this relative importance changed between London and Aswan, as shown in Figure 4-17. This was done on the scale of the tested building by summing up all zones results.

These results have achieved the goal of understanding the relative geometrical importance of energy demand and provided a clear understanding of quantifying these differences between geometrical variables for both tested locations. Although there is a difference in impact order between the two chosen locations, the parameters with contextual impact, height, and built area ratio had more significance than the other two variables related to building typology and orientation. Another stage of analysing energy demand followed this analysis stage to investigate these results on the zones' level and orientation. This analysis showed a clear pattern of energy demand for each of cooling and heating. It was proved that zones with more exposure to the outdoor had significantly different energy demand profiles than zones that fall on more adiabatic floors. In addition, this study illustrated the challenges and limitations of conducting such a comprehensive simulation on an urban scale. It also provided a clear understanding of the needed computational power to run this amount of simulations. These results lay the groundwork for the following steps of the research scope.

The preliminary studies utilised parametric modelling tools to investigate the capability of generating different models with a set of variables to break down the challenge of urban modelling complexity into its essential geometrical variables. This approach proved helpful in expanding our administration of urban modelling complexity by breaking it down to its geometrical parameters. These results added to the growing body of research that highlights the relationship between geometry and performance. Moreover, it helped to establish the framework's classification approach to achieve the research aim of optimising neighbourhood geometry based on performance aspects. As to urban performance aspects, preliminary studies compelled the research scope to investigate optimising direct solar radiation in neighbourhoods due to its low computational costs and time consumption. This thesis scope is a proof of concept for the framework capabilities and objectives. This study has shaped the methods used in this research to create this framework as a proof of concept to generate neighbourhood models, classify it based on geometrical features, predict its solar radiation performance to overcome the time constraint and provide an optimal solution to fulfil the aim and guide different stakeholders in the early stages of design.

9.3 Urban complexity modelling and generative design.

The research scope started with setting out the pool of iterations available for testing. Based on literature and preliminary studies, generating these iterations focused on emphasising the chosen platform of modelling (Mcneel 2014). This generative modelling stage was done by dividing the workflow into two parallel routes, one for the generated geometry and another for creating a classification attribute. The initial generation of the model used generation components from a ready-made plug-in for the modelling platform, Decoding Spaces (Koenig et al. 2017). The choice of both the modelling platform and the plug-in as the initial generation step of the framework was to fulfil the literature recommendation of creating more accessible tools for users, allowing for more user agency. It was recommended to reduce dependency on ready-made plug-ins, known as "black box" tools that did not allow for much user interaction. This reason was also the drive to add more capabilities to the ones that were available in Decoding Spaces.

The final version of the generation and classification workflow consisted of ten classes that varied between generative and analytical modelling tasks. The generative modelling tasks received multiple user inputs to control the model and its possible variations. At the same

time, the analytical classes were responsible for providing the geometrical analysis needed to classify each building characteristics as an attribute to the geometrical models. The whole process is discussed in detail in Chapter Five. This work also showed some limitations of the used tool. Some of the generated options had irregular configuration forms due to some short comes of the tool. Dependency on a continuously developed tool was another limitation for this plug-in and other plug-ins during the development and testing of the framework.

This work contributes to existing knowledge of urban parametric modelling methods by providing a novel way of generating geometrically classified urban modelling iterations. This approach has proved helpful in expanding our understanding of how to handle urban modelling complexity by analysing and classifying geometrical characteristics. This complexity breakdown provided an addition to the initially used plug-in and the conventional practice of parametric urban modelling by adding this novelty of producing a list of text classification tags of the neighbourhood buildings' geometrical attributes, in addition to its capability of controlling generated model in terms of height distribution, urban voids placement, and percentage. The generation framework also proved its capability of accepting different geometrical contextual conditions by testing its generative and classification capabilities with varying boundaries of neighbourhoods.

9.4 Urban geometry classification and application of neural networks

Chapter Six is discussing the classification database setup and the iteration pool settings. Besides, it goes through multiple stages of investigation and development for the classification tag as data texts. The principal theoretical implication of this classification is that its capability to find similarities to overcome the simulation time consumption limitation, highlighted by the preliminary studies. Thus, research has also developed a text-based lookup to compare the generated tags to the saved database to find similar buildings in the generated neighbourhoods. This data handling and automation of the iteration process have allowed additional capabilities to allow the dynamic generation of different neighbourhood models.

The generation of the classification tags into clear numeric textual nature preserved accessibility to users while allowing for applying machine learning methods to these datasets. This was fulfiling the purpose to investigate the potentiality of predicting simulation results

utilising ANNs principles to enhance the time consumption of the process. Chapter Seven shows the phases of implementing these ANN principles on a dataset of classification tags attached to its saved direct solar radiation simulation results. It started with a trial of a ready-made plug-in that offered the application of ANN in Grasshopper platform. Following this, the research developed its own bespoke ANN node. This ANN node is initially an open-source python code (The Codacus 2017). The code had fewer library dependencies to allow for future development and further access to this framework component. This process went through multiple phases of development and testing of this node and its capabilities of predicting solar radiation results for the classified database.

Testing this python coded component was shown with different dataset scales and was conducted over three different testing groups. The first group was on buildings level, comparing building solar radiation results directly to predictions from trained ANN for 10,000 buildings and its classification tags. Another group was for 100 urban configurations that we are sharing some geometrical generation settings to the configurations used in training the ANN. The final group was for another 100 urban configurations selected randomly from the total pool of available iterations. Urban accuracy tests were done by summing the complete ANN prediction results with the tags found by the lookup code to compare them with urban solar radiation simulation results.

These tests were done on unseen datasets for four different sizes of training to examine the appropriate size of the needed training to reach for a significant accuracy. The different dataset sizes were 10000, 20000, 150000 and 200000 entry databases. The final test was repeated with varying training settings, and the testing was for both trained ANNs. These test results are shown in section 7.2.3.

The assessment of time-saving performance is calculated by comparing the time consumed by the ANN node with the time consumed by the conventional simulation methods. On a building level, ANN tags' prediction consumes 3% of the simulation time. This is due to predicting the whole set of building tags for a configuration in one run instead of simulating each building with its different context and getting its results.

It is important to note that the training time is not factored into these comparisons due to the difference in the number of buildings and configurations used by these trained ANNs.

Moreover, solar radiation simulation is one of the least time-consuming simulation aspects compared to other simulation aspects like energy demand or daylighting performance. This finding result shows the potential of utilising this classification method accompanied by ANN predictions to get even better time-performing frameworks that will reach even better time-saving results for other more time-consuming aspects of environmental performance.

For accuracy assessment, the aim focused on the prediction accuracy without giving attention to the impact of the individual parameters on the prediction results. Therefore accuracy was measured by R^2 value results (Zikmund et al. 2003). The final test of 200000 dataset results for the similarly selected 100 configurations was 0.99 R^2 value for the tested training sample. As shown and discussed in section 7.3, This value was lower for the randomly generated 100 configurations predictions' correlations shows that R^2 value result was 0.83. This difference in correlation coefficient values for the two groups of urban configuration predictions can show the impact of similarities between the tested configurations and the ones used in training. The accuracy achieved in these tests has passed the significance level suggested by (Henseler et al. 2009), where it is essential to mention that the level of an acceptable R^2 value is rarely determined. However, these achieved accuracy levels can be found similar to other investigations of applying ANN methods on different data sets and for different prediction goals in the urban design context (Chan & Chau, 2019; Lin et al., 2020). These correlation findings are in line with the building accuracy results, which achieved a 0.93 R^2 value.

These results have shown the significance of applying ANNs to predict solar radiation on the neighbourhood scale. Moreover, it emphasised the impact of investigating the relationship between geometry and environmental performance on the urban level. It demonstrated a direct benefit of exploring this relationship on enhancing the early stages of the design decision-making process and promising optimisation of both its function and product. This enhancement was done by illustrating the impact of this classification method when utilised in predicting solar radiation performance and its capability of saving simulation time and cost while achieving acceptable accuracy. It also provided a novel way of tackling the design problem through technology and highlighted some of the potentialities and challenges of the data-driven design methods.

These findings laid the ground for further investigation on other aspects of environmental performance, energy demand, daylighting availability, etc.., to allow for faster, more efficient and informed decision making in the early stages of urban neighbourhood design.

9.5 Optimisation and framework prototype testing

The final stage of this research was to investigate the potentialities of optimising neighbourhood geometry based on its solar radiation prediction achieved by the developed ANN. Optimisation of the design was by selecting the optimal solution within the available pool of iterations. This was done with attention to the importance of tackling the time constraint to provide an efficient inclusion of data-driven solutions in the early stages of design. Another aspect to take into consideration was the aim of keeping the framework accessible to its users. For these goals, the framework used a ready-made open-source tool available on Grasshopper platform and named Biomorpher. This tool allowed for user interaction within the generative process to assess and determine the direction of the optimisation process. Chapter Eight illustrates the different tests conducted using Biomorpher as a tool for optimisation within the framework scope. The tool input was live from ANN prediction results. The framework had FAR added as an extra fitness to ensure this multi-objective optimisation process had less bias in determining its optimal solution based on one fitness while ignoring any other, sometimes contradicting, fitnesses.

The first phase of testing continued the same pool of iterations set at the beginning of developing the framework. The tests varied in size by testing the framework to find optimal solutions within smaller portions of this pool, simply reducing the iteration number to make it easy for the framework to get to the optimal solution within what is available for it. These tests were on 1000, 12000 and 23328 (total number of available urban configurations in the pre-set pool of iterations).

The final and largest test resulted in finding the 5^{th} optimal solution based on the two determining fitnesses in around 24 hours, saving more than 80% of the time needed to run the 23328 simulations to get the optimal solution through conventional ways.

The following and final phase of testing was conducted on an existing neighbourhood to examine the framework's capability of running a generative optimisation process on an existing neighbourhood boundary and the extent of optimisation it can achieve against a

conventionally designed neighbourhood sharing the same climatic conditions. The tests were done on a selected neighbourhood in the city of new Aswan, Egypt, after building the exiting model and running its solar radiation simulation to benchmark the framework results. In this test, ANN predictions were also tested for 1000 generated buildings in this neighbourhood geometry to test the classification capability and its achieved accuracy. This test resulted in a 0.94 R2 value for positively correlated predictions compared to the simulation results of the same generated buildings. These results came in line with the previous ANN findings showing consistent results while changing the context and boundary, yet following the same generative and classification methods. Biomorpher ran for six generations for optimisation capability testing, taking less than 6 hours while reaching more than one better-performing iteration than the benchmarked existing neighbourhood design. Two of the optimal solutions were simulated to compare the solar radiation results to the benchmark. Both results of the simulation were again better performing than the existing design while providing higher FAR results. It is important to note that the optimal solutions' building typologies were different from the existing ones, resulting in lower buildings with larger plots and denser urban configurations.

9.6 Framework contribution to neighbourhood design

These concluded results emphasised this proof of concept capability of optimising urban neighbourhood geometry based on its solar radiation predictions. It also provides a constructive method of handling urban complexity through the classification of its geometrical features. This outcome responds to the research scope of investigating the relationship between geometry and its performance and utilising this relationship into a novel data-driven optimisation framework that enhances the process and the product of early stages of neighbourhood design. Although it is still a proof of concept, this framework allows decision-makers, planners, architects and real estate developers to make performance-based decisions in the early stages of neighbourhood design as it provides a time saving, accurate, accessible and efficient process of solar radiation optimisation.

Currently, the framework's can contribute to providing a roadmap to benefit from the link between geometry and performance on the urban level. The framework managed to illustrate a method of overcoming urban complexity by classifying geometrical features allowing for the

implementation of ANN application and GA optimisation to alleviate the computational and time cost challenge facing the data-driven approach in the early stages of urban design.

This research lays the ground for further development of the framework and its methods. Once this approach of data-driven design gets validate will influence the decision process in the early stages in design within the local policy and regulation of the designed project. For the Egyptian case, there is a continuous increase in urbanisation (World Bank group 2019). The current government plans more urban growth to accommodate this grown urban population in new cities and communities (Egyptian New Urban Communities Authority 2020). These newly developed cities have a significantly large portion of it planned with private sector developers to be gated communities or private owned neighbourhoods (Metwally and Soliman Abdalla 2013; Mohammed and Hammad 2019). At the same time, Egypt is proposing its 2030 vision which aims to fulfil the United Nations Sustainable Development Goals (SDGs) (United Nations Development Program UNDP 2015; Sustainable Development Goals Knowledge Platform 2021; United Nations Development Program UNDP [no date]). This tool assists different stakeholders in analysing these new urban expansions and communities with evidence-based design decisions in the early design stages. In Egypt's case, the benefits of using such a framework will help governmental decision-makers and planners and extend this benefit to private sector real estate developers and architects and urban designers. This method of including data-driven design in the early stages of urban neighbourhood design complies with the current movement towards climate-resilient new urban communities in Egypt.

The findings also open the door for further development to build a robust and better visually presented tool capable of conducting urban geometry optimisation for other aspects like daylighting availability and different climates, as long as it is thoroughly tested and prepared. It can be argued that these outcomes would lead to a significant change of the early stage of neighbourhood design as it will allow for more climate-based performance to guide the geometry and urban configuration across the neighbourhood scope.

9.7 Future work

This thesis has provided a deeper insight into the relationship between geometry and environmental performance on an urban scale. The contribution of this study confirmed the

capability to enable the inclusion of climate-based optimisation in the early stages of design for a neighbourhood scale as a proof of concept in Aswan, Egypt. This was achieved through the different stages of parametric generation and classification data flows. Moreover, it has shed new light on the implementation of Artificial Intelligence to enhance the time consumption of the simulation process to include performance-based decisions in the early stages of design on an urban scale. This took place in predicting the performance of urban configurations utilising Artificial Neural Networks and using a cluster-oriented genetic algorithm tool to conduct geometry optimisation on an urban scale for newly designed and existing contexts. However, it is essential to note that it is still a proof of concept. Therefore, The future work of developing this framework aims for a robust tool capable of efficiently conducting optimisation to include performance-based knowledge in the early stages of urban design with a user-friendly interface. So far, the framework is hosted on Grasshopper due to its accessibility to architects and designers. Yet, it will get an even more accessible and user-friendly interface in its final versions to reach more users and hopefully reach the stage of having a framework capable of generating optimised models for neighbourhood configurations with acceptable visualised performances.

This framework has shown the potential benefits of following these methods of prediction and optimisation. Future studies may investigate the same techniques on more complex and computational costly simulation aspects like energy demand, lighting availability, outdoor thermal comfort, etc., which will give a further comprehensive analysis of the urban geometry in the early stages of design.

The framework also will be developed to have the capabilities to be conducted on different climate conditions and weather files. This will be done by building the needed dataset for training the neural network to predict urban performance in these climate results.

Another methodological advancement is to add weights to the classification fragments based on their relative impact on investigated performance aspects. This will be done through more data analysis and engineering phases that will determine the used fragments impact on performance and assign weights on it while training the ANN for prediction. This will build an even more accurate flow of generation, classification, prediction and optimisation for neighbourhood designs in these different climate zones.

10 References

Alexander, C. 1979. *The timeless way of building*. New York: Oxford University Press.

Alexander, C., Ishikawa, S. and Silverstein, M. 1977. *A pattern language: towns, buildings, construction*. Oxford University Press.

Ali, U., Shamsi, M.H., Alshehri, F., Mangina, E. and Donnell, J.O. 2018. Application of Intelligent Algorithms for Residential Building Energy Performance Rating Prediction UCD Energy Institute , University College Dublin , Belfield , Dublin 4 , Ireland School of Computer Science and Informatics , University College Dublin (U.

ANSI/ASHRAE/IESNA 2010. *ANSI/ASHRAE/IESNA 90.1-2010, Energy Standard for Buildings Except Low-Rise Residential Buildings. American Society of Heating, Refrigerating and Air-Conditioning Engineers*. Atlanta, Georgia.

Anton, I. and Tănase, D. 2016. Informed Geometries. Parametric Modelling and Energy Analysis in Early Stages of Design. In: *Energy Procedia*., pp. 9–16. doi: 10.1016/j.egypro.2015.12.269.

Arlot, S. and Celisse, A. 2010. A survey of cross-validation procedures for model selection. *Statistics Surveys* 4, pp. 40–79. doi: 10.1214/09-SS054.

Atherton, P., Weiler, K. and Greenberg, D. 1978. Polygon shadow generation. In: *ACM SIGGRAPH Computer Graphics*., pp. 275–281. Available at: internal-pdf://95.150.71.55/p275-atherton.pdf.

Attia, S. and Evrard, A. 2013. Benchmark Models for Air Conditioned Residential Buildings in Hot Humid Climate. In: *13th Conference of International Building Performance Simulation Association*. Chambry, France. Available at: http://orbi.ulg.ac.be/handle/2268/164047 [Accessed: 22 March 2017].

Attia, S., Hensen, J.L.M., Beltrán, L. and De Herde, A. 2012. Selection criteria for building performance simulation tools: contrasting architects' and engineers' needs. *Journal of Building Performance Simulation* 5(3), pp. 155–169. Available at: http://www.tandfonline.com/doi/abs/10.1080/19401493.2010.549573 [Accessed: 2 April 2019].

Autodesk.inc [no date]. Insight - High performance and sustainable building design analysis. Available at: https://insight360.autodesk.com/oneenergy/ [Accessed: 1 April 2019].

Autodesk inc. 1985. Autodesk | 3D Design, Engineering & Construction Software. Available at: https://www.autodesk.com/ [Accessed: 31 March 2019].

Autodesk Inc. 2012. Maya | Computer Animation & Modelling Software | Autodesk. Available at: https://www.autodesk.co.uk/products/maya/overview?mktvar002=716541&mkwid=sRGNSTFw8%7Cpcrid%7C212217280476%7Cpkw%7Cautodesk maya%7Cpmt%7Ce%7Cpdv%7Cc%7Cslid%7C%7Cpgrid%7C43332388520%7Cptaid%7Ckwd-298358543755%7C&intent=&utm_medium=cpc&utm_source=google&utm_campaign=GGL_Maya_UK_BR_SEM& [Accessed: 23 February 2018].

Autodesk Inc. 2017a. AutoCAD For Mac & Windows | CAD Software | Autodesk. Available at: https://www.autodesk.co.uk/products/autocad/overview [Accessed: 14 June 2017].

Autodesk Inc. 2017b. Green Building Studio. Available at: https://gbs.autodesk.com/GBS/ [Accessed: 14 June 2017].

Autodesk Inc. 2017c. Revit Family | BIM Software | Autodesk. Available at: https://www.autodesk.com/products/revit-family/overview [Accessed: 14 June 2017].

Azari, M., Tayyebi, A., Helbich, M. and Reveshty, M.A. 2016. Integrating cellular automata, artificial neural network, and fuzzy set theory to simulate threatened orchards: Application to Maragheh, Iran. *GIScience and Remote Sensing* 53(2), pp. 183–205. doi: 10.1080/15481603.2015.1137111.

Bahrami, A. 2018. HETEROPTERA. Available at: https://www.food4rhino.com/app/heteroptera.

Baker, N. and Steemers, K. 2003. *Energy and environment in architecture: a technical design guide*. Taylor & Francis.

Bassett, T., Lannon, S.C., Waldron, D. and Jones, P.J. 2012. Calculating the solar potential of the urban fabric with SketchUp and HTB2. In: *Solar Building Skins,* . Bressanone, Italy

Batty, M. 2009. A Digital Breeder for Designing Cities. *Architectural Design* 79(4), pp. 46–49. Available at: http://doi.wiley.com/10.1002/ad.916 [Accessed: 24 February 2018].

Beirão, J. 2012. *CItyMaker; Designing Grammars for Urban Design*. Delft University of Technology, Faculty of Architecture, Department Architectural Engineering+ Technology, Department of Urbanism]. Available at: http://abe.tudelft.nl/article/view/beirao/ [Accessed: 13 February 2018].

Biljecki, F., Ledoux, H., Stoter, J. and Zhao, J.Q. 2014. Formalisation of the level of detail in 3D city modelling. *Computers Environment and Urban Systems* 48, pp. 1–15. Available at: internal-pdf://211.145.144.190/Biljecki CEUS 2014.pdf.

Birmingham City Council 2017. *Adopted Birmingham Development Plan 2031 | Birmingham City Council*. Available at: https://www.birmingham.gov.uk/downloads/file/5433/adopted_birmingham_development_plan_2031 [Accessed: 20 September 2019].

Bonham, C.R. and Parmee, I.C. 2004. Developments of the cluster oriented genetic algorithm (COGA). *Engineering Optimisation* 36(2), pp. 249–279. Available at: http://www.tandfonline.com/doi/abs/10.1080/03052150410001650160 [Accessed: 2 April 2020].

Bouchahm, Y., Fatiha, B. and Bouketta, S. 2012. numerical simulation of effect of urban geometry layouts on wind and natural ventilation under mediterranean climate. In: *ASCAAD*. Manama (Kingdom of Bahrain), pp. 195–202. Available at: http://papers.cumincad.org/cgi-bin/works/Show?_id=ascaad2012_020&sort=DEFAULT&search=ascaad 2012&hits=745 [Accessed: 23 November 2016].

Bourbia, F., Bouchahm, Y. and Mansouri, O. 2010. The influence of Albedo on the urban microclimatic street canyon. In: *Proc., 6th Int. Conf. Arab Society for Computer Aided Architectural Design, ASCAAD, Manama, Kingdom of Bahrain*., pp. 159–169. Available at: internal-pdf://244.238.78.255/ascaad2012_016.content.pdf.

BRE and DECC 2013. *The Government's Standard Assessment Procedure for Energy Rating of Dwellings 2012 edition*. Available at: www.bre.co.uk/sap2012 [Accessed: 13 June 2016].

Bruse, M., Huttner, S., Hofmeyer, J., Seeger, D. and Simon, H. 2016. The Hitchhiker's Guide

to ENVI-met. Available at: http://www.model.envi-met.com/hg2e/doku.php?id=files:downloadv4 [Accessed: 3 May 2016].

Butcher, K. 2006. *CIBSE Guide A Environmental Design*. CIBSE Publications.

Calcerano, F. and Martinelli, L. 2016. Numerical optimisation through dynamic simulation of the position of trees around a stand-alone building to reduce cooling energy consumption. *Energy and Buildings* 112, pp. 234–243. doi: 10.1016/j.enbuild.2015.12.023.

Caldas, L.G. and Norford, L.K. 2002. A design optimisation tool based on a genetic algorithm. *Automation in Construction* 11(2), pp. 173–184. Available at: http://linkinghub.elsevier.com/retrieve/pii/S0926580500000960 [Accessed: 14 June 2017].

Çalışkan, O. 2017. Parametric Design in Urbanism: A Critical Reflection. *Planning Practice and Research* 32(4) 8 August, pp. 417–443. Available at: https://www.tandfonline.com/doi/full/10.1080/02697459.2017.1378862 [Accessed: 23 February 2018].

Carr, L.J., Dunsiger, S.I. and Marcus, B.H. 2010. Walk Score™ as a global estimate of neighborhood walkability. *American Journal of Preventive Medicine* 39(5), pp. 460–463. doi: 10.1016/j.amepre.2010.07.007.

Carr, L.J., Dunsiger, S.I. and Marcus, B.H. 2011. Validation of Walk Score for estimating access to walkable amenities. *British Journal of Sports Medicine* 45(14), pp. 1144–1148. doi: 10.1136/bjsm.2009.069609.

Chan, S.Y. and Chau, C.K. 2019. Development of artificial neural network models for predicting thermal comfort evaluation in urban parks in summer and winter. *Building and Environment* 164, p. 106364. doi: 10.1016/j.buildenv.2019.106364.

Chatzivasileiadi, A., Lila, A.M.H., Lannon, S. and Jabi, W. 2018. The Effect of Reducing Geometry Complexity on Energy Simulation Results. In: *36th annual Education and research in Computer Aided Architectural Design in Europe (eCAADe)*. Lodz, Poland. Available at: file:///C:/Users/anas_/Downloads/AS675140174090245153797736923_content_1.pdf [Accessed: 11 December 2018].

Che-Ani, A.I., Shahmohamadi, P., Sairi, A., Mohd-Nor, M.F.I., Zain, M.F.M. and Surat, M.

2009. Mitigating the urban heat island effect: some points without altering existing city planning. *European Journal of Scientific Research* 35, pp. 204–216.

Chiaradia, A. 2009. Spatial Design Economies. *Architectural Design* 79(4), pp. 80–85. Available at: http://doi.wiley.com/10.1002/ad.921 [Accessed: 24 February 2018].

Choudhary, R., Papalambros, P.Y. and Malkawi, A. 2005. A hierarchical design optimisation approach for meeting building performance targets. *Architectural Engineering and Design Management* 1(1), pp. 57–76. doi: 10.1080/17452007.2005.9684584.

Cichocka, J.M., Migalska, A., Browne, W.N. and Rodriguez, E. 2017. SILVEREYE – The Implementation of Particle Swarm Optimisation Algorithm in a Design Optimisation Tool. Springer, Singapore, pp. 151–169. Available at: http://link.springer.com/10.1007/978-981-10-5197-5_9 [Accessed: 16 January 2019].

Conceição António, C.A., Monteiro, J.B. and Afonso, C.F. 2014. Optimal topology of urban buildings for maximization of annual solar irradiation availability using a genetic algorithm. *Applied Thermal Engineering* 73, pp. 424–437. Available at: internal-pdf://134.2.137.64/1-s2.0-S1359431114006693-main Optimal topology.pdf LB - not a detailed model, only solar irradiation, capacity decided by the designer for the plot of the case study model,.

CORE studio 2017a. TT Toolbox | Food4Rhino. Available at: https://www.food4rhino.com/app/tt-toolbox [Accessed: 30 September 2019].

CORE studio 2017b. TT Toolbox | Food4Rhino.

Cross, N. 2001. Designerly Ways of Knowing: Design Discipline Versus Design Science. *Design Issues* 17(3), pp. 49–55. doi: 10.1162/074793601750357196.

Cui, Z. and Cai, X. 2013. Artificial Plant Optimisation Algorithm. In: *Swarm Intelligence and Bio-Inspired Computation*. Elsevier Inc., pp. 351–365. Available at: http://dx.doi.org/10.1016/B978-0-12-405163-8.00016-8 [Accessed: 26 September 2019].

Dallal, N. El and Visser, F. 2015. A climate responsive urban design tool: a platform to improve energy efficiency in a dry hot climate. *International Journal of Sustainable Energy* . Available at: http://www.tandfonline.com/doi/abs/10.1080/14786451.2015.1091836 [Accessed: 23 November 2016].

Deleuran, A. 2018. Problem Controlling Range of Sliders with Python - Scripting - McNeel Forum. Available at: https://discourse.mcneel.com/t/problem-controlling-range-of-sliders-with-python/61908/7 [Accessed: 24 December 2019].

Dogan, T. and Reinhart, C. 2013. Automated conversion of architectural massing models into thermal 'shoebox' models. *BS2013: 13th Conference of International Building Performance Simulation Association* , pp. 3745–3752. Available at: http://www.ibpsa.org/proceedings/BS2013/p_1123.pdf.

Dogan, T., Reinhart, C. and Michalatos, P. 2014. Automated multi-zone building energy model generation for schematic design and urban massing studies. In: *IBPSA eSim conference, Ottawa, Canada*.

Dogan, T., Reinhart, C. and Michalatos, P. 2016. Autozoner: an algorithm for automatic thermal zoning of buildings with unknown interior space definitions Autozoner: an algorithm for automatic thermal zoning of buildings with unknown interior space definitions. *Journal of Building Performance Simulation* 9(2), pp. 176–189. Available at: https://www.tandfonline.com/action/journalInformation?journalCode=tbps20 [Accessed: 28 March 2019].

Dortdivanlioglu, H. and Economou, A. 2017. Outlining Terragni. Springer, Singapore, pp. 381–394. Available at: http://link.springer.com/10.1007/978-981-10-5197-5_21 [Accessed: 12 February 2018].

Dounis, A.I., Science, A. and Piraeus, T.E.I. 2014. Artificial intelligence for energy conservation in buildings Artificial intelligence for energy conservation in buildings. *Advanes in Building Energy Research* . doi: 10.3763/aber.2009.0408.

Duarte, J.P. and Beirão, J. 2011. Towards a Methodology for Flexible Urban Design: Designing with Urban Patterns and Shape Grammars. *Environment and Planning B: Planning and Design* 38(5), pp. 879–902. Available at: http://journals.sagepub.com/doi/10.1068/b37026 [Accessed: 11 February 2018].

Duarte, J.P., Rocha, J.M. and Soares, G.D. 2007. Unveiling the structure of the Marrakech Medina: A shape grammar and an interpreter for generating urban form. *AI EDAM* 21(04), pp. 317–349. Available at:

http://www.journals.cambridge.org/abstract_S0890060407000315 [Accessed: 11 February 2018].

Egyptian Ministry of State for Administrative Development 2016. New Urban Communities Authority. Available at: http://www.newcities.gov.eg/english/default.aspx [Accessed: 22 March 2017].

Egyptian New Urban Communities Authority 2020. new urban cities. Available at: http://www.newcities.gov.eg/know_cities/default.aspx [Accessed: 23 June 2021].

El-deep, K., El-Zafarany, A. and Sheriff, A. 2012. Effect of building form and urban pattern on energy consumption of residential buildings in different desert climate. In: *PLEA 2012- 28th Conference, Opportunities, Limits & Needs Towards an environmentally responsible architecture*. lima, Peru

Esri 2016. ArcGIS. Available at: https://www.arcgis.com/features/.

Fonseca, J.A., Nguyen, T.A., Schlueter, A. and Marechal, F. 2016. City Energy Analyst (CEA): Integrated framework for analysis and optimisation of building energy systems in neighborhoods and city districts. *Energy and Buildings* 113, pp. 202–226. Available at: http://dx.doi.org/10.1016/j.enbuild.2015.11.055.

Gerber, D.J. and Lin, S.-H.E. 2014. Designing in complexity: Simulation, integration, and multidisciplinary design optimisation for architecture. *SIMULATION* 90(8), pp. 936–959. Available at: http://sim.sagepub.com/cgi/doi/10.1177/0037549713482027 [Accessed: 14 June 2017].

Gips, J. 1975. Shape Grammars and their Uses: Artificial Perception, Shape Generation and Computer Aesthetics (Interdisciplinary Systems Research series 10). *Basel and Stuttgart: Birkhäuser Verlag*

giulio 2017. GhPython. Available at: https://www.food4rhino.com/app/ghpython [Accessed: 2 October 2019].

Gorelick, N., Hancher, M., Dixon, M., Ilyushchenko, S., Thau, D. and Moore, R. 2017. Google Earth Engine: Planetary-scale geospatial analysis for everyone. *Remote sensing of Environment* 202, pp. 18–27.

Greater London Authority 2016. *The London Plan 2016 PDF | London City Hall*. Available at: https://www.london.gov.uk/what-we-do/planning/london-plan/current-london-plan/london-plan-2016-pdf [Accessed: 20 September 2019].

Greenberg, E. and Erdine, E. 2014. Computing the Urban Block - Local Climate Analysis and Design Strategies. In: *Thompson, Emine Mine (ed.), Fusion - Proceedings of the 32nd eCAADe Conference - Volume 1, Department of Architecture and Built Environment, Faculty of Engineering and Environment, Newcastle upon Tyne, England, UK, 10-12 September 2014, pp. 145-152*.

Harding, J. 2017. Biomorpher. Available at: https://github.com/johnharding/Biomorpher/blob/master/README.md.

Hazelrigg, G.A. 1998. A framework for decision-based engineering design. *Journal of Mechanical Design, Transactions of the ASME* 120(4), pp. 653–658. doi: 10.1115/1.2829328.

Hemsath, T.L. and Bandhosseini, K.A. 2015. Sensitivity analysis evaluating basic building geometry's effect on energy use. doi: 10.1016/j.renene.2014.11.044.

Henseler, J., Ringle, C.M. and Sinkovics, R.R. 2009. The use of partial least squares path modeling in international marketing. *Advances in International Marketing* 20, pp. 277–319. doi: 10.1108/S1474-7979(2009)0000020014.

Heumann, A. 2018. MetaHopper | Food4Rhino. Available at: https://www.food4rhino.com/app/metahopper [Accessed: 20 December 2019].

Hillier, B. 2007. *Space is the machine: a configurational theory of architecture*. Space Syntax.

Hongyun, H. and Radwan, A. 2021. Economic and social structure and electricity consumption in Egypt. *Energy* 231, p. 120962. doi: 10.1016/J.ENERGY.2021.120962

Illuminating Engineering Society (IES) 2017. ANSI/IES RP-16-17.

Jabi, W. 2013. *Parametric design for architecture*.

Jabi, W. 2015. Linking Design and Simulation Using Non-manifold Topology. *Architectural Science Review* 8628(January), p. In press. Available at: http://dx.doi.org/10.1080/00038628.2015.1117959.

Jain, A.K., Mao, J. and Mohiuddin, K.M. 1996. Artificial neural networks: A tutorial. *Computer* 29(3), pp. 31–44. doi: 10.1109/2.485891.

Jakubiec, J.A. and Reinhart, C.F. 2011. DIVA 2.0: Integrating daylight and thermal simulations using rhinoceros 3D, DAYSIM and EnergyPlus. In: *Proceedings of Building Simulation 2011: 12th Conference of International Building Performance Simulation Association*., pp. 2202–2209. Available at: http://www.scopus.com/inward/record.url?eid=2-s2.0-84870189464&partnerID=tZOtx3y1 [Accessed: 11 May 2016].

Jin, J.-T. and Jeong, J.-W. 2014. Optimisation of a free-form building shape to minimize external thermal load using genetic algorithm. *Energy and Buildings* 85, pp. 473–482. Available at: http://linkinghub.elsevier.com/retrieve/pii/S0378778814008196 [Accessed: 14 June 2017].

Jones, P., Patterson, J. and Lannon, S. 2007. Modelling the built environment at an urban scale—Energy and health impacts in relation to housing. *Landscape and Urban Planning* . Available at: http://www.sciencedirect.com/science/article/pii/S0169204607001430 [Accessed: 13 June 2016].

Jones, P.J., Lannon, S.C., Li, X., Bassett, T. and Waldron, D. 2013. Intensive building energy simulation at early design stage. In: *13th International Conference of the International Building Performance Simulation Association*. Chambéry, France: International Building Performance Simulation Association (IBPSA), pp. 862–869.

Jowers, I., Earl, C. and Stiny, G. 2017. Shape Computations Without Compositions. Springer, Singapore, pp. 348–365. Available at: http://link.springer.com/10.1007/978-981-10-5197-5_19 [Accessed: 12 February 2018].

Kampf, J.H. and Robinson, D. 2010. Optimisation of building form for solar energy utilisation using constrained evolutionary algorithms. *Energy and Buildings* 42, pp. 807–814. Available at: internal-pdf://76.184.125.97/1-s2.0-S0378778809003156-main.pdf.

Karimi, K. 2012. A configurational approach to analytical urban design: 'Space syntax' methodology. *URBAN DESIGN International* 17(4), pp. 297–318. Available at: http://link.springer.com/10.1057/udi.2012.19 [Accessed: 13 February 2018].

Kim, T.J., Wiggins, L.L. and Wright, J.R. 2012. *Expert systems: Applications to urban planning*.

Springer Science & Business Media.

Koenig, R. 2011. Generating urban structures: a method for urban planning supported by multi-agent systems and cellular automata. *Przestrzeń i Forma* (nr 16), pp. 353–376. Available at: https://www.infona.pl/resource/bwmeta1.element.baztech-article-BPS1-0047-0065 [Accessed: 24 February 2018].

Koenig, R., Knecht, K. and Miao, Y. 2020. *The Development of Optimisation Methods in Generative Urban Design: A Review*. Available at: https://www.researchgate.net/publication/344460745 [Accessed: 3 July 2021].

Koenig, R., Miao, Y., Knecht, K., Buš, P. and Mei-Chih, C. 2017. Interactive Urban Synthesis. Springer, Singapore, pp. 23–41. Available at: http://link.springer.com/10.1007/978-981-10-5197-5_2 [Accessed: 24 February 2018].

Koenig, R., Treyer, L., Schmitt, G. and Zurich, E. 2013. Graphical Smalltalk with My Optimisation System for Urban Planning Tasks. In: *Generation, Exploration and Optimisation - Volume 2 - Computation and Performance - eCAADe 31*. Delft, The Netherlands, pp. 195–203. Available at: http://www.ia.arch.ethz.ch/ [Accessed: 24 February 2018].

König, R. 2015. CPlan: An Open Source Library for Computational Analysis and Synthesis. Available at: https://e-pub.uni-weimar.de/opus4/frontdoor/index/index/docId/2503 [Accessed: 24 February 2018].

König, R., Schmitt, G., Standfest, M., Chirkin, A. and Klein, B. 2017. Cognitive computing for urban planning. In: *The virtual and the real in planning and urban design : perspectives, practices and applications*. Routledge, pp. 93–111. Available at: https://www.research-collection.ethz.ch/handle/20.500.11850/230314 [Accessed: 24 February 2018].

Kottek, M., Grieser, J., Beck, C., Rudolf, B. and Rubel, F. 2006. World map of the Köppen-Geiger climate classification updated. *Meteorologische Zeitschrift* 15(3), pp. 259–263. Available at: http://www.schweizerbart.de/papers/metz/detail/15/55034/World_Map_of_the_Koppen_Geiger_climate_classificat?af=crossref [Accessed: 6 March 2017].

Krarti, M. 2003. An overview of artificial intelligence-based methods for building energy systems. *Journal of Solar Energy Engineering, Transactions of the ASME* 125(3), pp. 331–342.

doi: 10.1115/1.1592186.

Krippendorff, K. 2005. *The semantic turn: A new foundation for design*. crc Press.

Lagios, K., Niemasz, J. and Reinhart F, C. 2010. Animated Building Performance Simulation (Abps) – Linking Rhinoceros / Grasshopper With Radiance / Daysim. *conference proceedings of SimBuild 2010* (August), p. 7. Available at: http://www.ibpsa.us/pub/simbuild2010/papers/SB10-DOC-TS06A-03-Lagios.pdf.

Landsberg, H.E. 1981. *The urban climate*. Academic press.

Lawson, B. 2006. *How Designers Think: The Design Process Demystified*. Oxford.

Leach, N. 2009. Digital Cities. *Architectural Design* 79(4), pp. 6–13. Available at: http://doi.wiley.com/10.1002/ad.911 [Accessed: 24 February 2018].

Lee, J., Gu, N. and Williams, A. 2014. Parametric design strategies for the generation of creative designs. *International Journal of Architectural Computing* 12(3), pp. 263–282. Available at: http://jac.sagepub.com/lookup/doi/10.1260/1478-0771.12.3.263 [Accessed: 14 June 2016].

Lee, J.H., Gu, N. and Ostwald, M.J. 2015. Creativity and parametric design? Comparing designer's cognitive approaches with assessed levels of creativity. *International Journal of Design Creativity and Innovation* 3(2), pp. 78–94. Available at: http://www.tandfonline.com/doi/abs/10.1080/21650349.2014.931826 [Accessed: 14 June 2016].

Lila, A.M.H. and Lannon, S. 2017. A parametric sensitivity analysis of the impact of built environment geometrical variables on building energy consumption. In: *PLEA*. Edinburgh, UK. Available at: https://orca.cf.ac.uk/102918/1/Lila PLEA 2017.pdf [Accessed: 20 September 2017].

Lin, J., Wan, H. and Cui, Y. 2020. Analyzing the spatial factors related to the distributions of building heights in urban areas: A comparative case study in Guangzhou and Shenzhen. *Sustainable Cities and Society* 52, p. 101854. doi: 10.1016/j.scs.2019.101854.

Lin, S.-H.E. 2014. *Designing-in performance: Energy simulation feedback for early stage design decision making*. UNIVERSITY OF SOUTHERN CALIFORNIA,. Available at:

http://gradworks.umi.com/36/28/3628225.html [Accessed: 14 June 2017].

Lin, S.-H.E. and Gerber, D.J. 2014. Designing-in performance: A framework for evolutionary energy performance feedback in early stage design. *Automation in Construction* 38, pp. 59–73. Available at: http://linkinghub.elsevier.com/retrieve/pii/S0926580513001647 [Accessed: 14 June 2017].

Lin, Y.P., Chu, H.J., Wu, C.F. and Verburg, P.H. 2011. Predictive ability of logistic regression, auto-logistic regression and neural network models in empirical land-use change modeling - a case study. *International Journal of Geographical Information Science* 25(1), pp. 65–87. doi: 10.1080/13658811003752332.

Lipp, M., Wonka, P., Wimmer, M., Lipp, M., Wonka, P. and Wimmer, M. 2008. Interactive visual editing of grammars for procedural architecture. In: *ACM SIGGRAPH 2008 papers on - SIGGRAPH '08*. New York, New York, USA: ACM Press, p. 1. Available at: http://portal.acm.org/citation.cfm?doid=1399504.1360701 [Accessed: 13 February 2018].

Mackey, C., Sadeghipour, M. and Samaras, P. 2015. ComfortCover: A Novel Method for the Design of Outdoor Shades. In: *SimAUD*. Alexandria, VA, USA: Society for Modeling & Simulation International, pp. 197–204.

Makki, M. and Showkatbakhsh, M. 2018. CONTROL OF MORPHOLOGICAL VARIATION THROUGH POPULATION BASED FITNESS CRITERIA. In: *Learning, Adapting and Prototyping, Proceedings of the 23rd International Conference of the Association for Computer-Aided Architectural Design Research in Asia (CAADRIA)2018*., pp. 153–162. Available at: https://www.researchgate.net/publication/325405944_CONTROL_OF_MORPHOLOGICAL_VARIATION_THROUGH_POPULATION_BASED_FITNESS_CRITERIA [Accessed: 25 September 2019].

Makki, M., Showkatbakhsh, M., Tabony, A. and Weinstock, M. 2019. Evolutionary algorithms for generating urban morphology: Variations and multiple objectives. *International Journal of Architectural Computing* 17(1), pp. 5–35. doi: 10.1177/1478077118777236.

Mandić, M. and Tepavčević, B. 2015. Analysis of shape grammar application as a tool for urban design. *Environment and Planning B: Planning and Design* 42(4), pp. 675–687. Available at: http://journals.sagepub.com/doi/10.1068/b130084p [Accessed: 13 February

2018].

Mans, D. 2016. Bumblebee: Grasshopper + Excel | Food4Rhino. Available at: https://www.food4rhino.com/app/bumblebee-grasshopper-excel [Accessed: 30 September 2019].

March, L. 1971. *Some elementary models of built forms*. University of Cambridge School of Architecture.

March, L. 2011. Forty Years of Shape and Shape Grammars, 1971 – 2011. *Nexus Network Journal* 13(1), pp. 5–13. Available at: http://link.springer.com/10.1007/s00004-011-0054-8 [Accessed: 7 February 2018].

March, L. and Steadman, P. 1971. *The geometry of environment: an introduction to spatial organization in design*. Riba London. Available at: http://library.wur.nl/WebQuery/clc/355952 [Accessed: 26 April 2016].

Martin, D. 2014. Rhino & Grasshopper soon even for Mac And Linux? - Grasshopper. Available at: https://www.grasshopper3d.com/forum/topics/rhino-grasshopper-soon-even-for-mac-and-linux?id=2985220%3ATopic%3A1167231&page=1#comments [Accessed: 20 September 2019].

Martin, L. and March, L. 1972. Urban Space and Structures. Cambridge Urban and Architectural Studies, No. 1.

Martine, G. and Marshall, A. 2007. UNFPA State of world population 2007: unleashing the potential of urban growth. In: *State of world population 2007: unleashing the potential of urban growth*. UNFPA

Maver, T.W. 1970. *Emerging Methods in Environmental Design and Planning Appraisal in the Building Design Process*. MIT Press. Post-occupancy Evaluation.

Mcneel, R. 2014. Grasshopper 3D. Available at: http://www.grasshopper3d.com/.

McNeel, R. 2014. Rhinoceros 5. Available at: https://www.rhino3d.com/.

Metwally, M. and Soliman Abdalla, S. 2013. Major Trends of the Gated Communities Development in Egypt an Approach to Urban Sustainability. In: *Privet Urbana Governance and Gated Communities*. Brighton,UK

Microsoft Corporation 2010. Microsoft Excel.

Mitchell, W.J. (William J. and J., W. 1990. *The logic of architecture : design, computation, and cognition*. MIT Press. Available at: https://dl.acm.org/citation.cfm?id=533511 [Accessed: 23 February 2018].

Mohammed, A. and Hammad, E. 2019. THE FUTUER OF URBAN GROWTH IN EGYPT ASSESSMENT OF CURRENT TRENDS AND FORGING NEW APPROACHES FOR SUSTAINABLE DEVELOPMENT. *journal of Engineering Sciences* 47(5), pp. 752–764.

Muslimin, R. 2017. EthnoComputation: An Inductive Shape Grammar on Toraja Glyph. In: *CAAD Futures*. turkey, pp. 329–347. Available at: https://link.springer.com/content/pdf/10.1007/978-981-10-5197-5_18.pdf [Accessed: 12 February 2018].

Naboni, E. 2014. Integration of Outdoor Thermal and Visual Comfort in Parametric Design. In: *30th International PLEA Conference*., pp. 1–10.

Nault, E., Peronato, G., Rey, E. and Andersen, M. 2015a. Review and critical analysis of early-design phase evaluation metrics for the solar potential of neighborhood designs. *Building and Environment* 92, pp. 679–691. doi: 10.1016/j.buildenv.2015.05.012.

Nault, É., Rastogi, P., Rey, E. and Andersen, M. 2015b. The sensitivity of predicted energy use to urban geometrical factors in various climates. In: *PLEA 2015*. Available at: http://infoscience.epfl.ch/record/211101/files/PLEA2015_Final.pdf [Accessed: 13 May 2016].

Nault, E., Rey, E. and Andersen, M. 2016. A Multi-Criteria Decision-Support Workflow for Early-Stage Neighborhood Design based on Predicted Solar Performance. In: *PLEA 2016, 36th International Conference on Passive and Low Energy Architecture. Cities, Buildings, People: Towards Regenerative Environments*. los angeles,USA. Available at: https://infoscience.epfl.ch/record/218840/files/1093-Nault01.pdf?version=1 [Accessed: 17 March 2017].

Nault, E., Rey, E., Andersen, M., Nault, É., Rey, E. and Andersen, M. 2013. Early design phase evaluation of urban solar potential: Insights from the analysis of six projects. In: *IBPSA 2013-13th International Conference of the International Building Performance Simulation*

Association.

New Urban Communities Authority at The Ministry of Housing. Utilities & Urban Communities. [no date]. Home - New Aswan. Available at: http://www.newcities.gov.eg/english/New_Communities/Aswan/default.aspx [Accessed: 22 March 2020].

Nguyen, A.T., Reiter, S. and Rigo, P. 2014. A review on simulation-based optimisation methods applied to building performance analysis. *Applied Energy* 113, pp. 1043–1058. doi: 10.1016/j.apenergy.2013.08.061.

Norton, B.A., Coutts, A.M., Livesley, S.J., Harris, R.J., Hunter, A.M. and Williams, N.S.G. 2015. Planning for cooler cities: A framework to prioritise green infrastructure to mitigate high temperatures in urban landscapes. *Landscape and Urban Planning* 134, pp. 127–138. Available at: internal-pdf://135.136.230.62/Norton land 2015.pdf.

Nourian, P., Rezvani, S. and Sariyildiz, S. 2010. Designing with Space Syntax. In: *eCAADe 31*., pp. 357–366.

Nunez, M. 1974. *The energy balance of an urban canyon*. University of British Columbia.

Paliwal, M. and Kumar, U.A. 2009. Neural networks and statistical techniques: A review of applications. *Expert Systems with Applications* 36(1), pp. 2–17. doi: 10.1016/j.eswa.2007.10.005.

Panão, O., N, M.J., Gonçalves, H.J.P. and Ferrão, P.M.C. 2008. Optimisation of the urban building efficiency potential for mid-latitude climates using a genetic algorithm approach. *Renewable Energy* 33, pp. 887–896. Available at: internal-pdf://136.136.71.94/Optimisation of the urban building efficiency.pdf.

Parish, Y.I.H. and Müller, P. 2001. Procedural modeling of cities. *Proceedings of the 28th annual conference on Computer graphics and interactive techniques - SIGGRAPH '01* (August), pp. 301–308. Available at: http://portal.acm.org/citation.cfm?doid=383259.383292.

Pedregosa, F., Varoquaux, G., Gramfort, A. and Michel, V. 2010. 3.1. Cross-validation: evaluating estimator performance — scikit-learn 0.22.1 documentation. Available at:

https://scikit-learn.org/stable/modules/cross_validation.html#cross-validation [Accessed: 20 January 2020].

Peronato, G., Nault, É., Cappelletti, F., Peron, F. and Andersen, M. 2015. A parametric design-based methodology to visualize building performance at the neighborhood scale. In: *Building Simulation Applications BSA 2015 2nd IBPSA-Italy conference Bozen-Bolzano, 4th – 6th February 2015*. Bozen-Bolzano University Press, pp. 351–358.

Piacentino, G. 2018. The only safe way to Update Sliders in GhPython - Grasshopper Developer - McNeel Forum. Available at: https://discourse.mcneel.com/t/the-only-safe-way-to-update-sliders-in-ghpython/61995 [Accessed: 24 December 2019].

Picco, M. and Marengo, M. 2015. On the Impact of Simplificationson Building Energy Simulation for Early Stage Building Design. *Journal of Engineering and Architecture* 3, pp. 66–78. Available at: file:///C:/Users/anas_/Downloads/On_the_Impact_of_Simplificationson_Building_Energy.pdf [Accessed: 11 December 2018].

Pinto Duarte, J., Ducla-Soares, G. and Sampaio, A.Z. 2005. Urban Grammars: Towards Flexible Urban Design. In: *Digital Design: The Quest for New Paradigms [23nd eCAADe Conference Proceedings*. Lisbon (Portugal): Education in Computer Aided Architectural Design in Europe, pp. 491–500. Available at: http://papers.cumincad.org/cgi-bin/works/Show?2005_491 [Accessed: 23 February 2018].

Płoszaj-Mazurek, M., Ry, Z., Grochulska-Salak, M., Ryńska, E. and Grochulska-Salak, M. 2020. Methods to Optimize Carbon Footprint of Buildings in Regenerative Architectural Design with the Use of Machine Learning, Convolutional Neural Network, and Parametric Design. *Energies* 13(20). Available at: www.mdpi.com/journal/energies.

Proving Ground 2017. archinate / lunchboxml / ProvingGround.LunchBoxML-grasshopper / Components / nodeNeuralNetwork.cs — Bitbucket. Available at: https://bitbucket.org/archinate/lunchboxml/src/master/ProvingGround.LunchBoxML-grasshopper/Components/nodeNeuralNetwork.cs [Accessed: 14 January 2020].

PROVING GROUND 2018. Lunch-Box. Available at: https://provingground.io/tools/lunchbox/ [Accessed: 3 December 2019].

Python Software Foundation. 2001. Python. Available at: https://www.python.org/about/legal/ [Accessed: 19 January 2019].

Python software foundation 1990. 11.1. pickle — Python object serialization — Python 2.7.17 documentation. Available at: https://docs.python.org/2/library/pickle.html [Accessed: 20 January 2020].

Radfar, M. 2012. *Urban Microclimate, Designing the Spaces Between Buildings*.

Rahbar, M., Nejad, M.J.M., Bemanian, M.R. and Davaie-Markazi, A. hossein 2020. Artificial neural network for outlining and predicting environmental sustainable parameters. *Journal of Sustainable Architecture and Urban Design* 7(2), pp. 169–182. Available at: https://jsaud.sru.ac.ir/article_1264.html [Accessed: 4 July 2021].

Rakha, T. and Nassar, K. 2011. Genetic algorithms for ceiling form optimisation in response to daylight levels. *Renewable Energy* 36, pp. 2348–2356. Available at: internal-pdf://244.234.15.130/1-s2.0-S0960148111000826-main.pdf.

Rakha, T. and Reinhart, C. 2012. Generative urban modeling: a design work flow for walkability-optimized cities. *Proceedings of SimBuild* . Available at: http://www.tarekrakha.com/s/SB12_TS04b_3_Rakha.pdf [Accessed: 10 May 2016].

Rakha, T. and Reinhart, C.F. 2013. A Carbon Impact Simulation-Based Framework for Land Use Planning and Non-Motorized Travel Behaviour Interactions. *13th Conference of International Building Performance Simulation Association* . Available at: http://static1.squarespace.com/static/53d65c30e4b0d86829f32e6f/t/53d667c2e4b07404fe20051c/1406560194389/CarbonImpact_BS2013_TR_CR.pdf.

Ratti, C., Baker, N. and Steemers, K. 2005. Energy consumption and urban texture. *Energy and Buildings* 37, pp. 762–776. Available at: internal-pdf://0350249394/1-s2.0-S0378778804003391-main.pdf.

Reinhart, C. 2017. Daysim. Available at: http://daysim.ning.com/ [Accessed: 17 March 2017].

Reinhart, C.F., Dogan, T., Alstan, J.J., Rakha, T. and Sang, A. 2013. Umi - an Urban Simulation Environment for Building Energy Use , Daylighting and Walkability. *Proceedings of BS2013:*

13th Conference of International Building Performance Simulation Association , pp. 476–483.

Robinson, D. 2006. Urban morphology and indicators of radiation availability. *Solar Energy* 80(12), pp. 1643–1648.

Robinson, D., Campbell, N., Gaiser, W., Kabel, K., Le-Mouel, A., Morel, N., Page, J., Stankovic, S. and Stone, A. 2007. SUNtool–A new modelling paradigm for simulating and optimising urban sustainability. *Solar Energy* 81(9), pp. 1196–1211.

Roche, F. 2009. I've Heard About … (A Flat, Fat, Growing Urban Experiment): Extract of Neighbourhood Protocols. *Architectural Design* 79(4), pp. 40–45. Available at: http://doi.wiley.com/10.1002/ad.915 [Accessed: 24 February 2018].

Rodríguez-Álvarez, J. 2016. Urban Energy Index for Buildings (UEIB): A new method to evaluate the effect of urban form on buildings' energy demand. *Landscape and Urban Planning* 148, pp. 170–187. Available at: http://linkinghub.elsevier.com/retrieve/pii/S0169204616000025.

Rutten, D. 2013. Galapagos: On the Logic and Limitations of Generic Solvers. *Architectural Design* 83(2)March, pp. 132–135. Available at: http://doi.wiley.com/10.1002/ad.1568 [Accessed: 16 June 2016].

Sabry, H., Sherif, A., Gadelhak, M. and Aly, M. 2014. Balancing the daylighting and energy performance of solar screens in residential desert buildings: Examination of screen axial rotation and opening aspect ratio. *Solar Energy* 103, pp. 364–377. Available at: http://linkinghub.elsevier.com/retrieve/pii/S0038092X14001030 [Accessed: 17 March 2017].

Sadeghipour, M. r and Pak, M. 2013. Ladybug: a Parametric Environmental Plugin for Grasshopper To Help Designers Create an Environmentally-Conscious Design. In: *13th Conference of International building Performance Simulation Association*. Chambéry, France, pp. 3129–3135. Available at: http://www.ibpsa.org/proceedings/bs2013/p_2499.pdf.

Samardžić-Petrović, M., Kovačević, M., Bajat, B. and Dragićević, S. 2017. Machine Learning Techniques for Modelling Short Term Land-Use Change. *ISPRS International Journal of Geo-Information* 6(12), p. 387. Available at: http://www.mdpi.com/2220-9964/6/12/387 [Accessed: 19 March 2020].

Schneider, S., Fischer, J.-R. and König, R. 2011. Rethinking Automated Layout Design: Developing a Creative Evolutionary Design Method for the Layout Problems in Architecture and Urban Design. In: *Design Computing and Cognition '10*. Dordrecht: Springer Netherlands, pp. 367–386. Available at: http://www.springerlink.com/index/10.1007/978-94-007-0510-4_20 [Accessed: 24 February 2018].

Schumacher, P. 2009. Parametric Patterns. *Architectural Design* 79(6)November, pp. 28–41. Available at: http://doi.wiley.com/10.1002/ad.976 [Accessed: 14 June 2016].

Schumacher, P. 2012. *The autopoeisis of architecture. Vol. II, A new agenda for architecture*. Wiley. Available at: https://books.google.co.uk/books?hl=en&lr=&id=IceUD0HqQr4C&oi=fnd&pg=PT13&dq=the+autopoiesis+of+architecture&ots=-wbd_bSbIJ&sig=mnIbeB_YozpGczvoKb2PUVQyxZ8#v=onepage&q=the autopoiesis of architecture&f=false [Accessed: 23 February 2018].

Schumacher, P. *Parametricism: A New Global Style for Architecture and Urban Design*. John Wiley & Sons, Ltd. 79. Available at: http://doi.wiley.com/10.1002/ad.912 [Accessed: 14 June 2016].

Schwarz, N. 2010. Urban form revisited-Selecting indicators for characterising European cities. *Landscape and Urban Planning* 96(1), pp. 29–47. Available at: http://dx.doi.org/10.1016/j.landurbplan.2010.01.007.

Shafizadeh-Moghadam, H., Asghari, A., Tayyebi, A. and Taleai, M. 2017. Coupling machine learning, tree-based and statistical models with cellular automata to simulate urban growth. *Computers, Environment and Urban Systems* 64, pp. 297–308. doi: 10.1016/j.compenvurbsys.2017.04.002.

Shekhawat, K. and Duarte, J.P. 2017. Rectilinear Floor Plans. Springer, Singapore, pp. 395–411. Available at: http://link.springer.com/10.1007/978-981-10-5197-5_22 [Accessed: 12 February 2018].

Showkatbakhsh, M. and Makki, M. 2020. Application of homeostatic principles within evolutionary design processes: adaptive urban tissues. *Journal of Computational Design and Engineering* 7(1), pp. 1–17. Available at:

https://academic.oup.com/jcde/article/7/1/1/5809434 [Accessed: 3 July 2021].

Simulation Research Group, U.S.D.O.E. 2017. DOE-2 | Simulation Research. Available at: http://simulationresearch.lbl.gov/projects/doe2 [Accessed: 14 June 2017].

Song, H., Ghaboussi, J. and Kwon, T. 2016. Architectural design of apartment buildings using the Implicit Redundant Representation Genetic Algorithm. *Automation in Construction* . Available at: http://www.sciencedirect.com/science/article/pii/S0926580516302102 [Accessed: 22 November 2016].

Steemers, K. 2003. Energy and the city: density, buildings and transport. *Energy and buildings* 35, pp. 3–14. Available at: internal-pdf://189.126.176.96/1-s2.0-S0378778802000750-main.pdf.

Stewart, I.D. and Oke, T.R. 2012. Local Climate Zones for Urban Temperature Studies. *Bulletin of the American Meteorological Society* 93, pp. 1879–1900. Available at: internal-pdf://120.3.66.3/Stewart BAMS 2012.pdf.

Stiny, G. 1982. Spatial Relations and Grammars. *Environment and Planning B: Planning and Design* 9(1), pp. 113–114. Available at: http://epb.sagepub.com/lookup/doi/10.1068/b090113 [Accessed: 7 February 2018].

Stiny, G. 2006. *Shape: Talking about Seeing and Doing*. Available at: https://books.google.com/books?hl=en&lr=&id=VC0TDgAAQBAJ&oi=fnd&pg=PP8&ots=dsTlngDjU-&sig=OwHUyGorKg6rizSLg3Pd0vJ0_zk [Accessed: 7 February 2018].

Stiny, G. and Gips, J. 1972. Shape grammars and the generative specification of painting and sculpture. *Information Processing 71 Proceedings of the IFIP Congress 1971. Volume 2* 71, pp. 1460–1465. Available at: https://www.researchgate.net/profile/James_Gips/publication/221329330_'Shape_Grammars_and_the_Generative_Specification_of_Painting_and_Sculpture'/links/569f87db08aee4d26ad264e4.pdf [Accessed: 7 February 2018].

Stouffs, R. and Sariyildiz, S. 2013. Theme Computation and Performance. In: *Computation and Performance – Proceedings of the 31st eCAADe Conference - Volume 1 eCAADe 2013*., p. 5. Available at: https://www.researchgate.net/profile/Rudi_Stouffs/publication/280931559_Computation_

Performance_Proceedings_of_the_31st_International_Conference_on_Education_and_research_in_Computer_Aided_Architectural_Design_in_Europe/links/55e6592b08aecb1a7ccd6998/Comp [Accessed: 23 February 2018].

Sustainable Development Goals Knowledge Platform 2021. Egypt Voluntary National Review 2021 . Available at: https://sustainabledevelopment.un.org/memberstates/egypt [Accessed: 23 June 2021].

Suyoto, W., Indraprastha, A. and Purbo, H.W.H. 2015. Parametric Approach as a Tool for Decision-making in Planning and Design Process. Case study: Office Tower in Kebayoran Lama. In: *Procedia - Social and Behavioral Sciences*., pp. 328–337. Available at: http://www.sciencedirect.com/science/article/pii/S1877042815033479 [Accessed: 29 May 2016].

Taleb, H. and Musleh, M.A. 2015. Applying urban parametric design optimisation processes to a hot climate: Case study of the UAE. *Sustainable Cities and Society* 14, pp. 236–253. doi: 10.1016/j.scs.2014.09.001.

Taleghani, M., Kleerekoper, L., Tenpierik, M. and van den Dobbelsteen, A. 2015. Outdoor thermal comfort within five different urban forms in the Netherlands. *Building and Environment* 83, pp. 65–78. doi: 10.1016/j.buildenv.2014.03.014.

Taleghani, M., Tenpierik, M., van den Dobbelsteen, A. and Sailor, D.J. 2014. Heat in courtyards: A validated and calibrated parametric study of heat mitigation strategies for urban courtyards in the Netherlands. *Solar Energy* 103, pp. 108–124. Available at: internal-pdf://133.131.175.100/1-s2.0-S0038092X14000516-main.pdf.

The Codacus 2017. Build Neural Network From Scratch in Python (no libraries) | The Codacus. Available at: https://thecodacus.com/neural-network-scratch-python-no-libraries/ [Accessed: 15 January 2020].

The MathWorks, I. 2017. MathWorks - Makers of MATLAB and Simulink. Available at: https://www.mathworks.com/ [Accessed: 14 June 2017].

Trigaux, D., Allacker, K. and Troyer, F. De 2014. A simplified approach to integrate energy calculation in the Life Cycle Assessment of neighbourhoods. In: *30th International PLEA Conference,Volume: Sustainable habitat for developing societiesPLEA*. Ahmedabad, India.

Available at: https://lirias.kuleuven.be/handle/123456789/481910 [Accessed: 22 November 2016].

Trigaux, D., Oosterbosch, B., Allacker, K. and De Troyer, F. 2015. A design tool to optimize solar gains and energy use in neighbourhoods. In: *PLEA 2015-Architecture in (R) Evolution*. Bologna, Italy: Ass. Building Green Futures, p. 305. Available at: internal-pdf://2.138.19.252/pu_337.pdf.

Trimble 2016. SketchUp. Available at: http://www.sketchup.com/products/sketchup-pro/new-in-2016?gclid=CInUn8WDpc0CFfUV0wodPL8LMQ.

Tsamis, A. 2017. The Marching Shape. Springer, Singapore, pp. 366–380. Available at: http://link.springer.com/10.1007/978-981-10-5197-5_20 [Accessed: 12 February 2018].

Tschetwertak, J., Schneider, S., Hollberg, A., Donath, D. and Ruth, J. 2017. A Matter of Sequence Investigating the Impact of the Order of Design Decisions in Multi-stage Design Processe. In: Editors: Çağdaş, G., Özkar, M., Gül, L.F., Gürer, E. ed. *Computer-Aided Architectural Design. Future Trajectories*. Istanbul, Turkey. Available at: https://books.google.co.uk/books?hl=en&lr=&id=SNgoDwAAQBAJ&oi=fnd&pg=PA100&dq=a+matter+of+sequence&ots=3t7-cpvDaN&sig=QWXvmN-WgU3UARYHNOx_viGiDTw#v=onepage&q=a matter of sequence&f=false [Accessed: 15 June 2018].

Turing, A.M. 1948. Intelligent machinery.

U.S. Department of Energy's (DOE) 2016. EnergyPlus™. Available at: https://energyplus.net/ [Accessed: 10 May 2016].

U.S. Department of Energy 2008. *Tax Deduction Qualified Software - Green Building Studio Web Service version 3.4*. Available at: www.buildings.energy.gov/qualified_software.html [Accessed: 1 April 2019].

United Nations Development Program UNDP 2015. Sustainable Development Goals | United Nations Development Programme. Available at: https://www.undp.org/sustainable-development-goals [Accessed: 23 June 2021].

United Nations Development Program UNDP [no date]. Sustainable Development Goals |

UNDP in Egypt. Available at: https://www.eg.undp.org/content/egypt/en/home/sustainable-development-goals.html [Accessed: 23 June 2021].

University of Applied Arts Vienna Bollinger+Grohmann Engineers. 2014a. octopus | Food4Rhino. Available at: http://www.food4rhino.com/app/octopus [Accessed: 14 June 2017].

University of Applied Arts Vienna Bollinger+Grohmann Engineers. 2014b. octopus | Food4Rhino.

Varoquaux, G., Buitinck, L., Louppe, G., Grisel, O., Pedregosa, F. and Mueller, A. 2015. Scikit-learn. *GetMobile: Mobile Computing and Communications* 19(1), pp. 29–33. doi: 10.1145/2786984.2786995.

Vartholomaios, A. 2017. A parametric sensitivity analysis of the influence of urban form on domestic energy consumption for heating and cooling in a Mediterranean city. *Sustainable Cities and Society* 28, pp. 135–145. Available at: https://www.sciencedirect.com/science/article/pii/S2210670716303572?via%3Dihub [Accessed: 11 January 2019].

Vasku, M. 2013. Generative Improvement of Street Networks Based on Space Syntax: Applied in a case study on an informal settlement in Jeddah. In: *ecaade 31*. Available at: https://repository.tudelft.nl/islandora/object/uuid%3Ac09440e6-99c8-4411-a2d3-23523b939349 [Accessed: 13 February 2018].

Vermeulen, T., Knopf-Lenoir, C., Villon, P. and Beckers, B. 2015. Urban layout optimisation framework to maximize direct solar irradiation. *Computers, Environment and Urban Systems* 51, pp. 1–12. Available at: internal-pdf://128.93.22.176/Vermeulen CEUS 2015.pdf.

Vidmar, J. 2013. Parametric Maps for Performance-Based Urban Design, A lateral method for 3D urban design. In: *eCAADe 31, City Modelling - Volume 1 - Computation and Performance*. Delft, Netherlands, pp. 311–316. Available at: http://www.modelur.com [Accessed: 10 December 2018].

Virk, G., Mylona, A., Mavrogianni, A. and Davies, M. 2015. Using the new CIBSE design summer years to assess overheating in London: Effect of the urban heat island on design.

Building Services Engineering Research and Technology 36, pp. 115–128. Available at: internal-pdf://205.132.173.209/BUILDING SERV ENG RES TECHNOL-2015-Virk-115-28.pdf.

Ward, G. 1994. The RADIANCE lighting simulation and rendering system. In: *Proceedings of the 21st annual conference on Computer graphics and interactive techniques*. Orlando, Florida: ACM., pp. 459–472. Available at: http://dl.acm.org/citation.cfm?id=192286 [Accessed: 11 May 2016].

Whitley, D. 1994. A genetic algorithm tutorial. *Statistics and Computing* 4(2), pp. 65–85. doi: 10.1007/BF00175354.

De wilde, P. 2018. *Building Performance Analysis*. Wiley. doi: 10.1002/9781119341901.

Winston, P. 2010. 6.034 Artificial Intelligence. Fall 2010. Massachusetts Institute of Technology: MIT OpenCourseWare, https://ocw.mit.edu. License: Creative Commons BY-NC-SA. Available at: https://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-034-artificial-intelligence-fall-2010/# [Accessed: 13 January 2020].

Woodbury, R. 2010. *Elements of parametric design*. Taylor & Francis Group. Available at: http://cw.routledge.com/textbooks/9780415779876/ [Accessed: 25 March 2019].

World Bank group 2019. Urban population - Egypt, Arab Rep. | Data. Available at: https://data.worldbank.org/indicator/SP.URB.TOTL?locations=EG [Accessed: 22 June 2021].

Wortmann, T. 2017. OPOSSUM Introducing and Evaluating a Model-based Optimisation Tool for Grasshopper. In: P. Janssen, P. Loh, A. Raonic, and M. A. Schnabel eds. *Protocols, Flows and Glitches, Proceedings of the 22nd International Conference of the Association for Computer-Aided Architectural Design Research in Asia (CAADRIA)*. Hong Kong, pp. 283–293. Available at: www.food4rhino.com [Accessed: 16 January 2019].

Wortmann, T., Costa, A., Nannicini, G. and Schroepfer, T. 2015. Advantages of surrogate models for architectural design optimisation. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing: AIEDAM* 29(4), pp. 471–481. doi: 10.1017/S0890060415000451.

Wortmann, T. and Nannicini, G. 2016. Black-box optimisation methods for architectural design. In: *CAADRIA 2016*. Melbourne, AU, p. Volume: 177-186. Available at:

https://www.researchgate.net/publication/299594622_Black-box_optimisation_methods_for_architectural_design [Accessed: 25 September 2019].

Wortmann, T. and Natanian, J. 2020. Multi-Objective Optimisation for Zero-Energy Urban Design in China: A Benchmark. In: *SIMAUD 2020*. Available at: https://www.researchgate.net/publication/341592609 [Accessed: 4 July 2021].

Wortmann, T. and Tunçer, B. 2017. Differentiating parametric design: Digital workflows in contemporary architecture and construction. *Design Studies* 52, pp. 173–197. Available at: https://www.sciencedirect.com/science/article/pii/S0142694X17300352 [Accessed: 25 March 2019].

Yi, Y.K. and Kim, H. 2015. Agent-based geometry optimisation with Genetic Algorithm (GA) for tall apartment's solar right. *Solar Energy* 113, pp. 236–250. Available at: internal-pdf://153.227.9.90/1-s2.0-S0038092X14005398-main.pdf.

Zhao, H.X. and Magoulès, F. 2012. A review on the prediction of building energy consumption. *Renewable and Sustainable Energy Reviews* 16(6), pp. 3586–3592. doi: 10.1016/j.rser.2012.02.049.

Zikmund, W.G., Babin, B.J., Carr, J.C. and Griffin, M. 2003. Business research methods 7th ed. *Thomson/South-Western*

11 Appendix A (Preliminary Study)

11.2 Preliminary study Aswan first phase

11.2.1 Geometrical parameters and simulation settings

The first stage of the preliminary study was conducted on energy demand for heat loss and gains while lighting consumption was normalized by setting the lighting controls to a static ON/OFF controls system. The building areas varied from 50% to 90% of each cell's area with a 10% differentiation for each group. In addition to scale, the height was a feature of geometrical variation in the study. Building heights varied between 3.5 metres and 24.5 metres with variation of a 3.5 metres floor height for each group. Moreover, the whole configuration is rotated by 11.25 degrees for each group of rotations creating five groups. Finally, the case study building has a fixed WWR for its four direction facades. The WWR varied between 20% and 80% with a 10% variation for each group iteration. These variables are shown in the following table

Geometrical parameters for the first phase of the sensitivity analysis

Geometrical Variables							
Height (metres)	3.5	7	10.5	14	17.5	21	24.5
Scale (built area ratios)	50%	60%	70%	80%	90%		
Orientation (degrees)	0	11.25	22.5	33.75	45		
Window-to-wall ratio	20%	30%	40%	50%	60%	70%	80%

These parametric variations created 1,224 iterations. The simulation was done for heights then repeated with a changed built area ratio then with a different orientation and finally with the change of WWR.

The analysed building has a 30% core to perimeter ratio. Energy Plus midrise apartment zone programs were chosen for the building zones with the apartment program for the perimeter zones and corridor program for the core zones. All zones are conditioned with the default set ideal air loads system for Heating, Ventilation and Air Conditioning (HVAC). The grid cell size is 23 by 23 metres as representing the common size of land in Egypt (El-deep et al., 2012)

11.2.2 Material inputs

The inputs for material were adjusted based upon some studies made in the same geographical context (El-deep et al. 2012; Attia and Evrard 2013). The material properties were fixed for all the iterations and designed based upon the specification of the Chartered Institution of Building Services Engineers (CIBSE) Guide for environmental design (Butcher 2006). Following table shows the material parameters used in the study.

The material parameters used in the study

CUSTOMIZED MATERIALS			
External Wall		Internal Wall	
U-Value	3.10	U-Value	5.29
Materials	CEMENT PLASTER BRICK (EXPOSED) CEMENT PLASTER	Materials	CEMENT PLASTER BRICK INTERIOR (EXPOSED) CEMENT PLASTER
Internal Floor		External Roof	
U-Value	1.43	U-Value	0.36
Materials	CERAMIC-FLOOR-TILES CEMENT-MORTAR(MOIST) CONCRETE CAST(HEAVYWEIGHT) GYPSUM-PLASTER	Materials	CEMENT-MORTAR(MOIST) EXPANDED POLYSTYRENE (EPS) CONCRETE, CAST (HEAVYWEIGHT)
Single Glazed Window			
U-Value	5.7		
Materials	CLEAR GLASS 3MM		

11.2.3 Results

As shown in the following figure,The results show a repetitive pattern of consumption for cooling. Where the consumption is reduced with the variation of heights the consumption continues descending for the next built area ratio group. There is a minimal effect to this loop from the

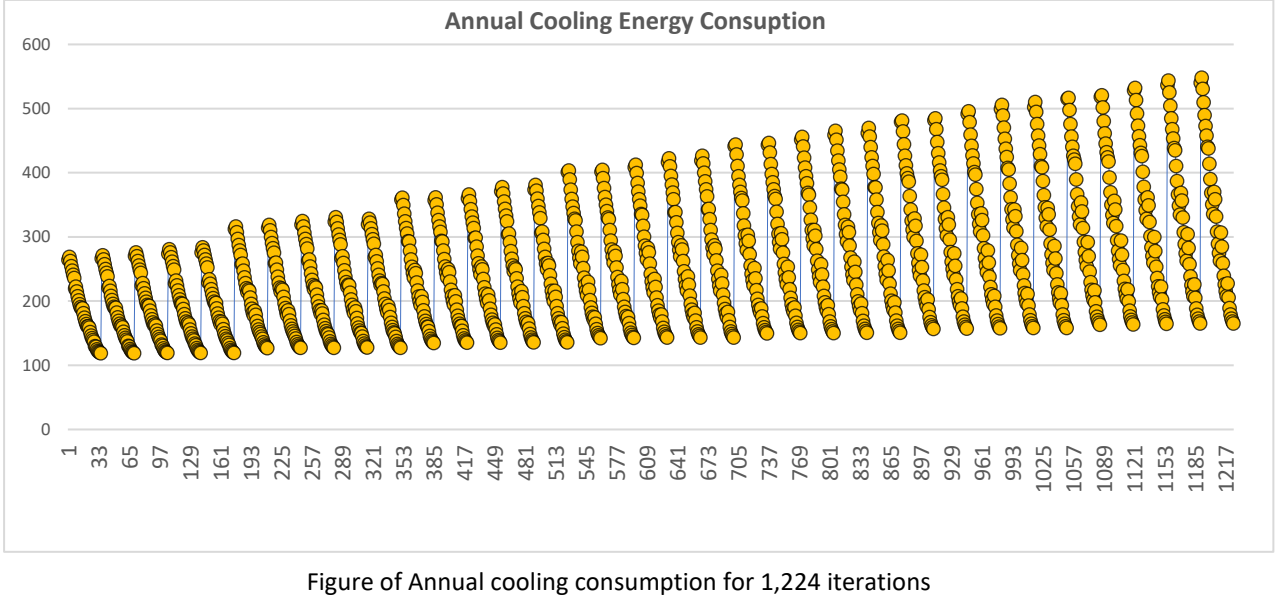


Figure of Annual cooling consumption for 1,224 iterations

orientation changes until the first group of WWR changes, and then there is a clear step in the consumption pattern, starting the next loop of readings. This step of change gets smaller with higher values of WWR. Furthermore, the results illustrated the clear correlation between solar gains and cooling consumption.

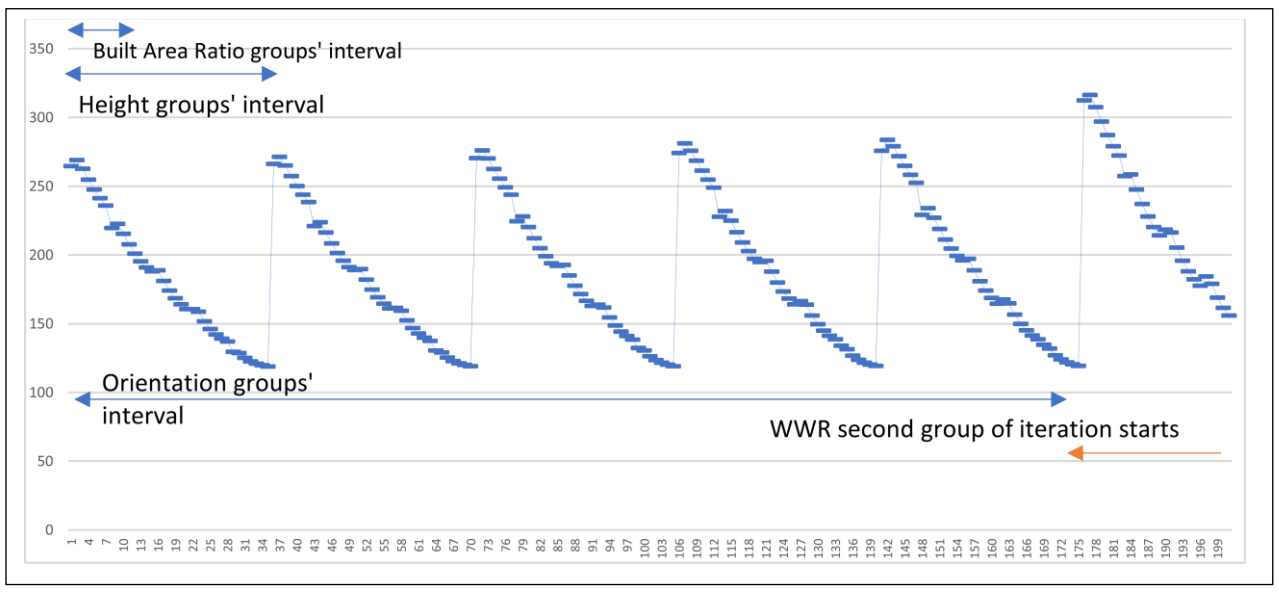


Figure of Group intervals for each parameter on the chart for the first 200 cases

This repetitive nature of the results encouraged the addition of another performance aspect to get a better insight about the performance. Therefore, the following phase added daylight availability as a core performance aspect to get this balance of performance between the energy demand and the daylighting availability, especially with the fact that WWR is affecting the results pattern significantly.

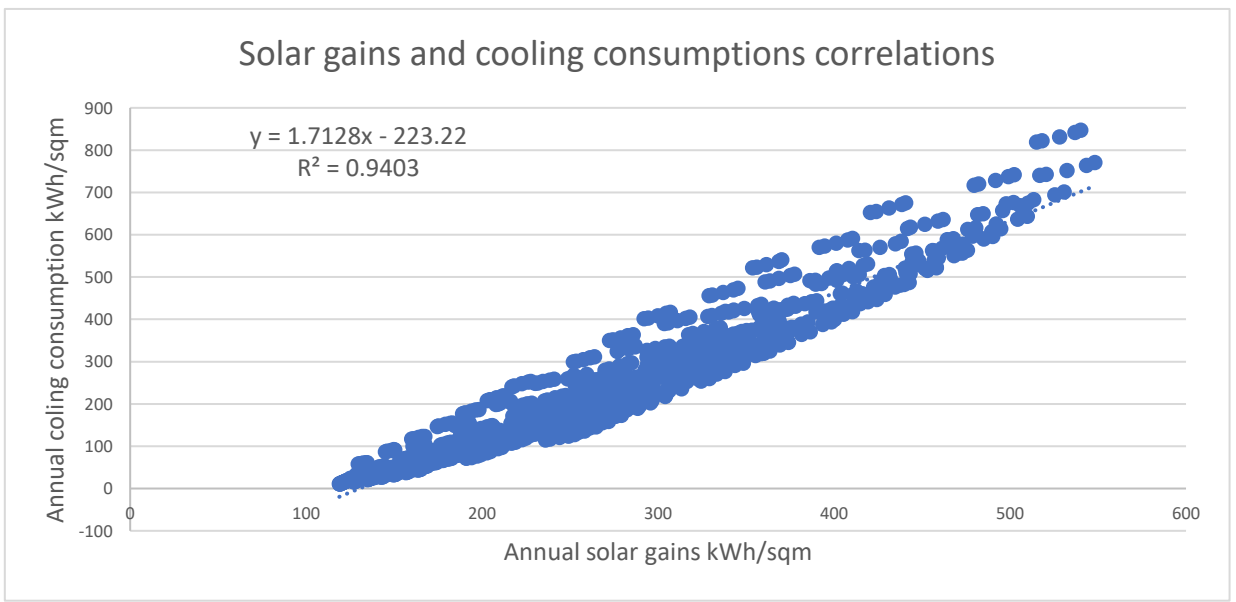


Figure of Annual solar gains and cooling energy consumption correlation

11.3 Results of daylighting availability for cooling daylight availability and energy comparison.

The comparison is for cooling and lighting energy consumption, once with lighting dimming controls and the other with standard on/off lighting controls.

The six groups are categorized by rotation angle and WWR for each group and each group has full heights and scale ranges mentioned earlier (see following figure). The groups are as shown in the following table.

Table of Features of the six groups

Group name	Rotation angle	WWR
A1	0	20%
A2	0	50%
A3	0	80%
B1	45	20%
B2	45	50%
B3	45	80%

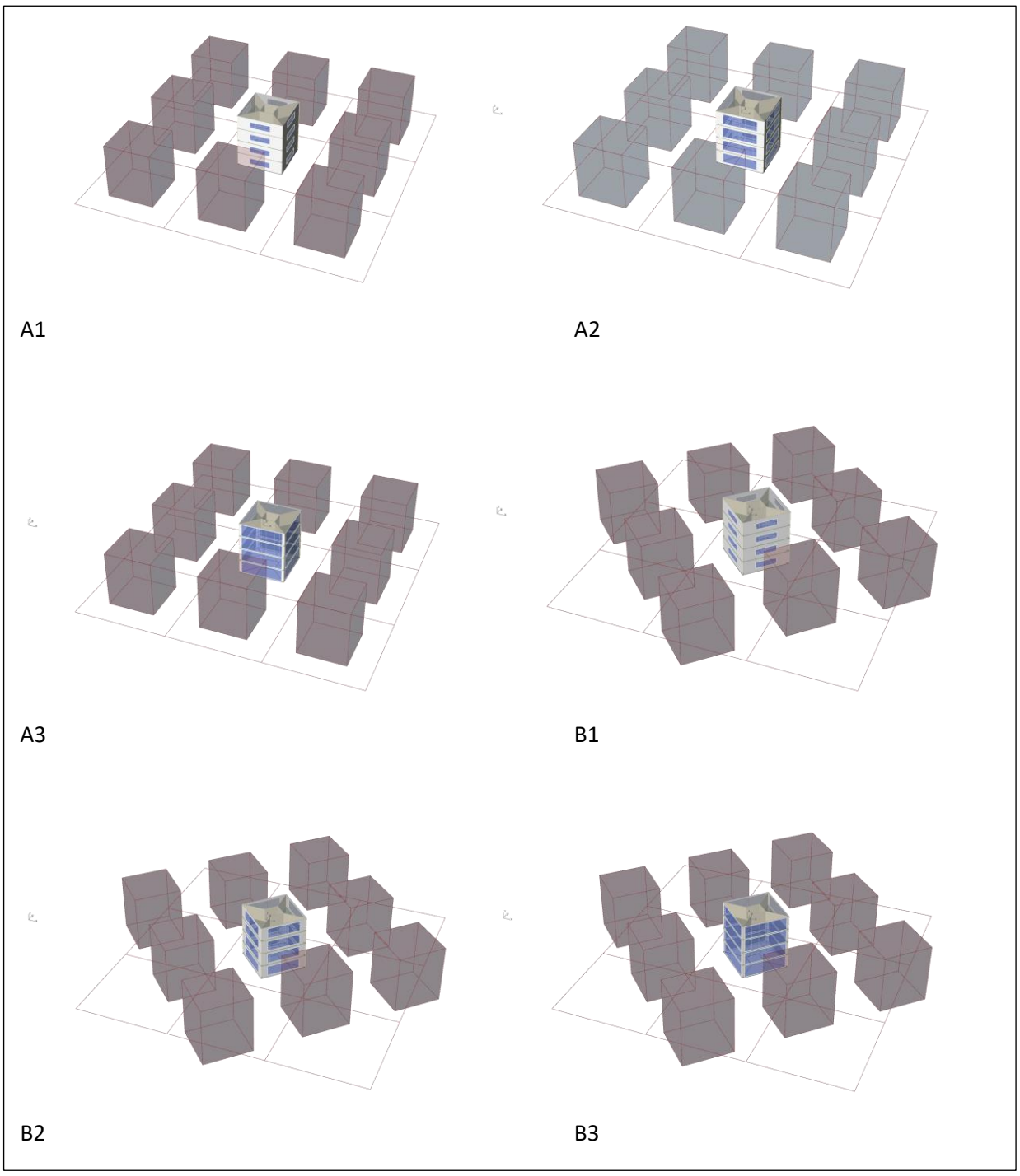


Figure shows six example cases of the six groups

11.3.1 Remarks on general lighting results

The lighting energy consumption results for the six groups are shown in the following figure, Each group of seven high buildings is shown by the major vertical guidelines. Each group has a different built-up area ratio increasing from 50% in the group from index 1 to 6 until it reached a 90% built-up ratio for the group starting at index 29 to 35. The initial observation shows a direct effect on daylighting availability distribution. The first notice of the result is that the WWR grouping has a pairing effect on the results. This is shown in that the A trend lines have a close trend line with the B groups. This shows that the orientation does not have as much significance as the WWR. After the direct effect of WWR the scale and the context proximity also show a great effect, especially in the lower floors in the higher cases which leads to the rise of lighting energy consumption. The margin of consumption for each height group increases simultaneously by the increase of built-up area ratios. Further, there is a gap between the consumption of the seven-floor-height prototype and its following one-floor-height prototype: the gap increases and becomes clearer with the increase of built-up area ratios for the groups A3 and B3 of .8 WWR, while it does not show as significant for groups A1 and B1 with .2 WWR which have a smoother trend line. For each height group, the increase of height also brings an increase of lighting energy consumption. This can caused by the increase of lit area and the increase of shaded areas, due the context getting higher and the lower floors receiving less daylighting. It is important to note that the whole margin of the lighting energy consumption falls in a small range from almost 34 kWh/sqm to almost 48 kWh/sqm for the whole range of iteration. As per cooling consumption, the pattern of consumption does not have the same baseline of applying the on/off controls as the one existing for lighting, so it will be included in the discussions of the groups' results.

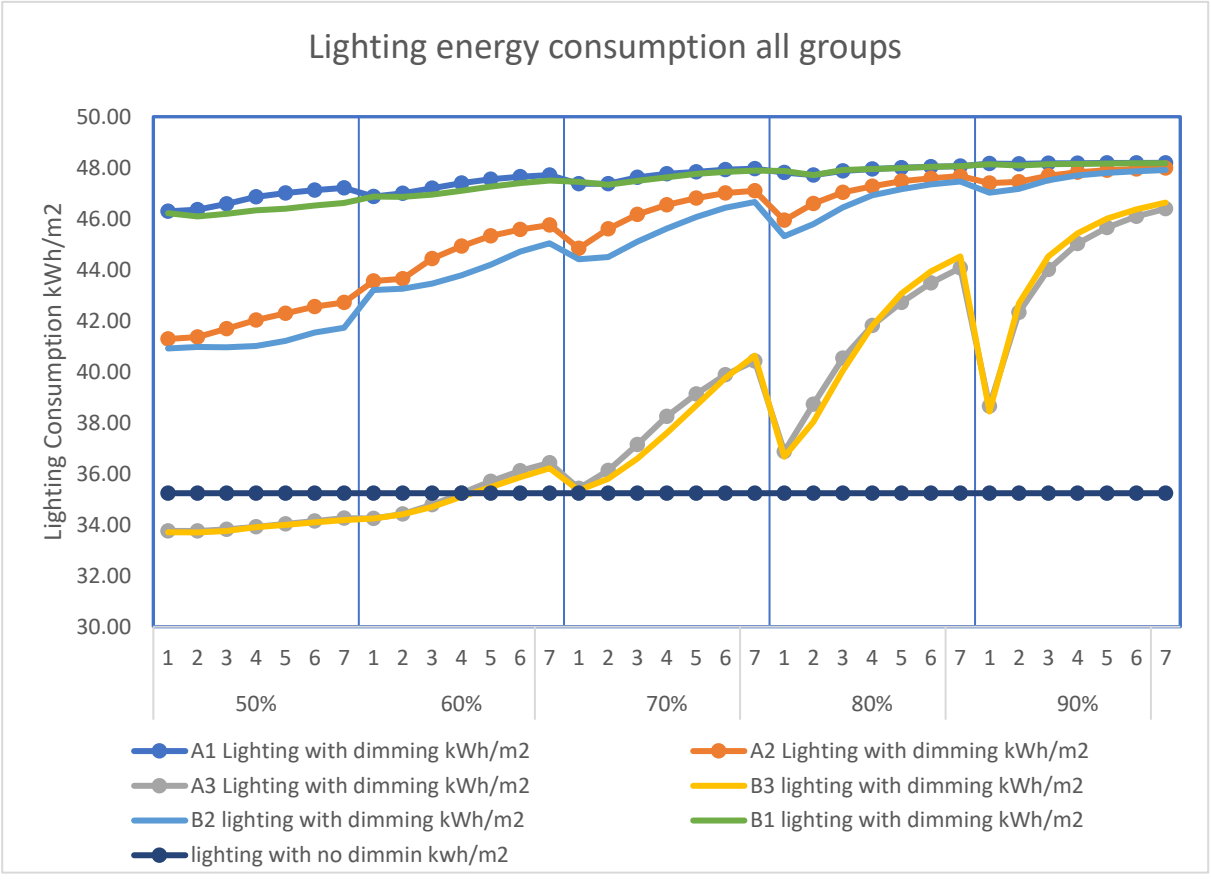


Figure shows the Lighting consumption patterns of the six groups compared with no dimming kWh/m2 results

11.3.2 Group A1 (rotation = 0, WWR = 20%)

The general note here in the following figure is the general decline of cooling consumption over the whole group due to the increase of built-up area ratios for the results of both diming and no dimming lighting controls, while the light energy consumption does not show the same impact occurring due to the iteration either for height or built-up area ratios. Another note for the cooling energy consumption within the height groups is that it starts with a slight increase between one- and two-floor-height iterations then starts the continuation of its general decline of the trend line.

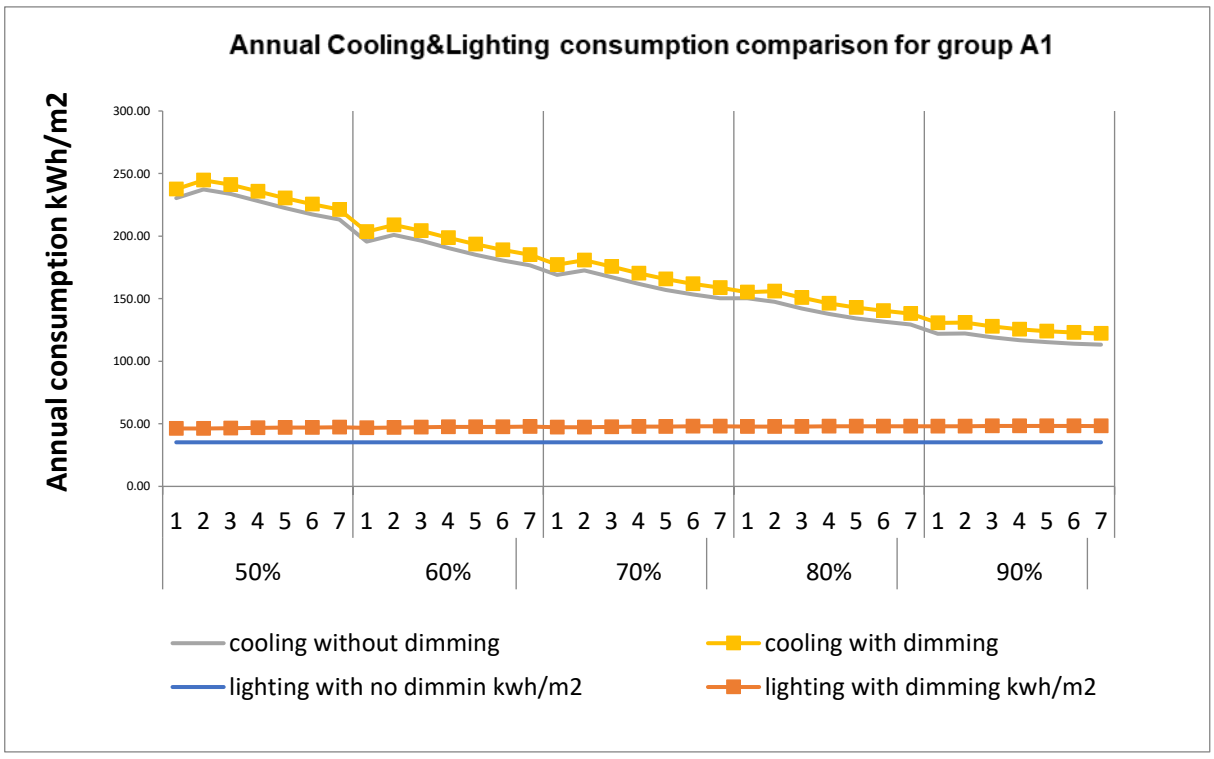


Figure of Annual cooling & lighting consumption results comparison for group A1

11.3.3 Group A2 (rotation = 0, WWR = 50%)

The change in values for cooling is different from the previous group as the bigger the built area and higher the configuration the more apparent the difference. With regard to lighting energy consumption, the change of urban configuration has more effect in this case on the dimming controls due to the bigger WWR ratio. The decline pattern is steep with the lower values of built area ratios which means the height is now as effective when it comes to larger built-up area ratios. This can be reasoned by the fact that less sun penetration is allowed to the configuration which minimizes the effect of height in comparison with built-up area ratios due to the proximity of buildings. This change in cooling consumption pattern is still consistent.

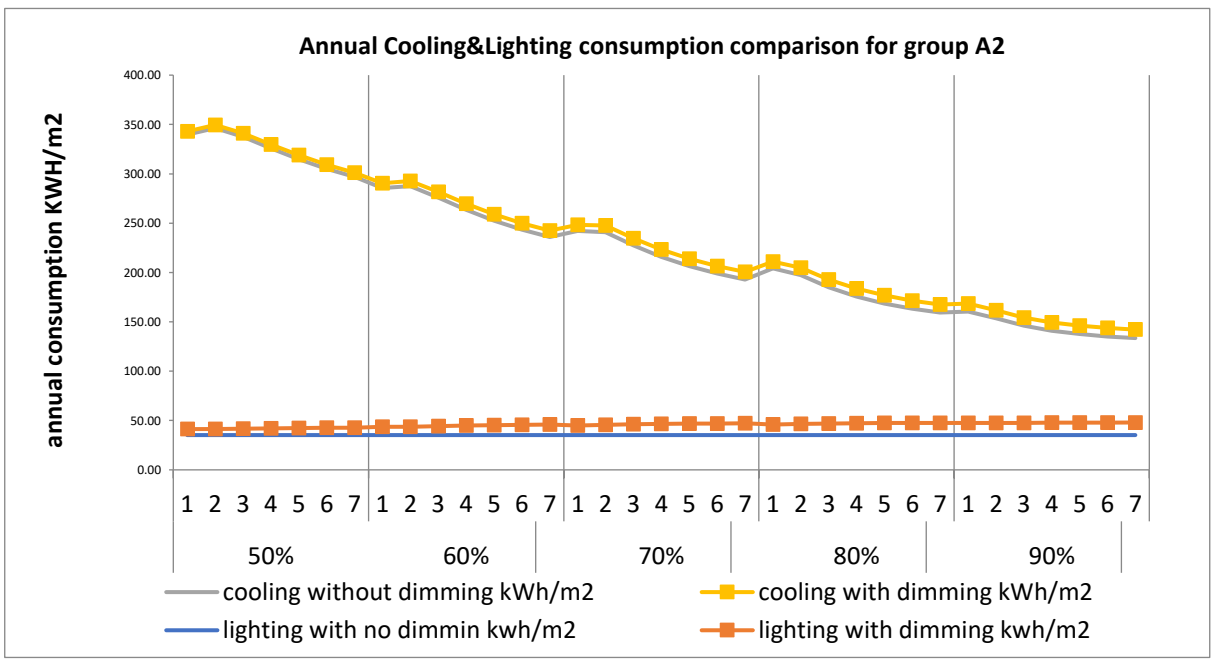


Figure of Annual cooling & lighting consumption results comparison for group A2

11.3.4 Group A3 (rotation = 0, WWR = 80 %)

The effect of outdoor geometrical changes is more apparent in this group on lighting consumption with dimming controls applied. On the other hand, the cooling values are getting reduced for the densest configurations. Then, the dimming-applied model begins to have a fraction of higher values of cooling consumption, but this did not change the pattern of cooling consumption consistency between different lighting controls.

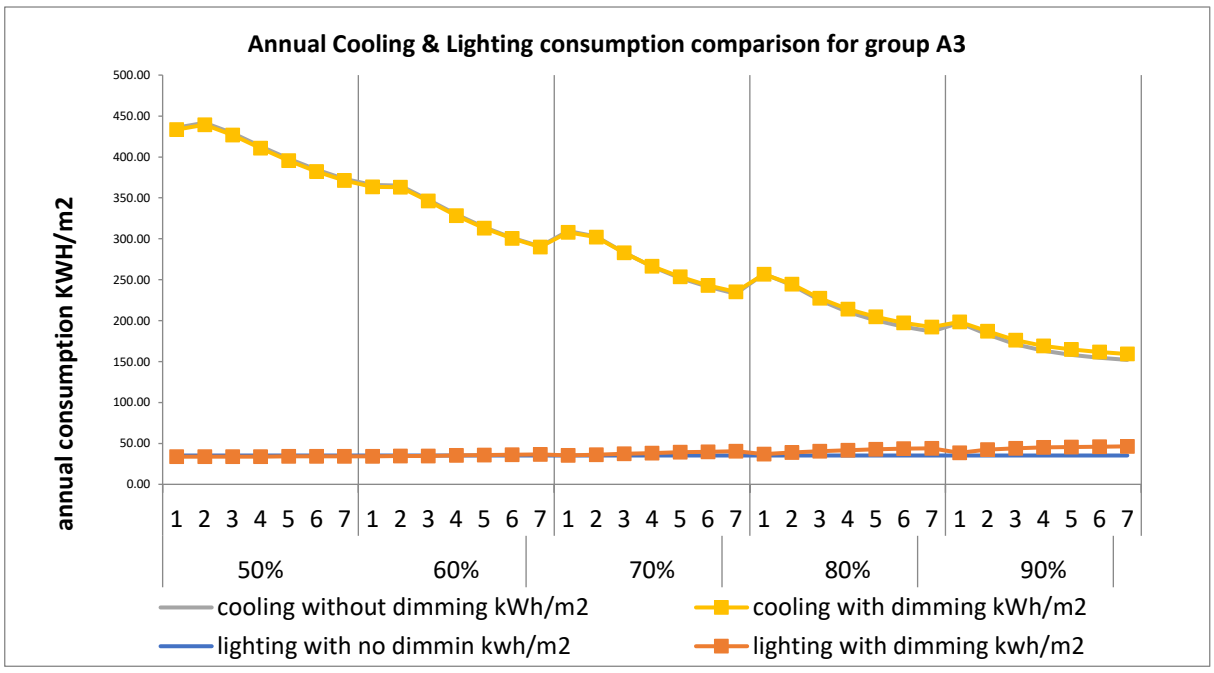


Figure of Annual cooling & lighting consumption results comparison for group A3

11.3.5 Group B1 (rotation = 45, WWR = 20%)

In general, the values of energy consumption for cooling and lighting are getting closer to group A1 when the configuration gets dense. Furthermore, the values are closer when it comes to one-floor runs and then continues to repeat the pattern again with the repetition of heights with different scales.

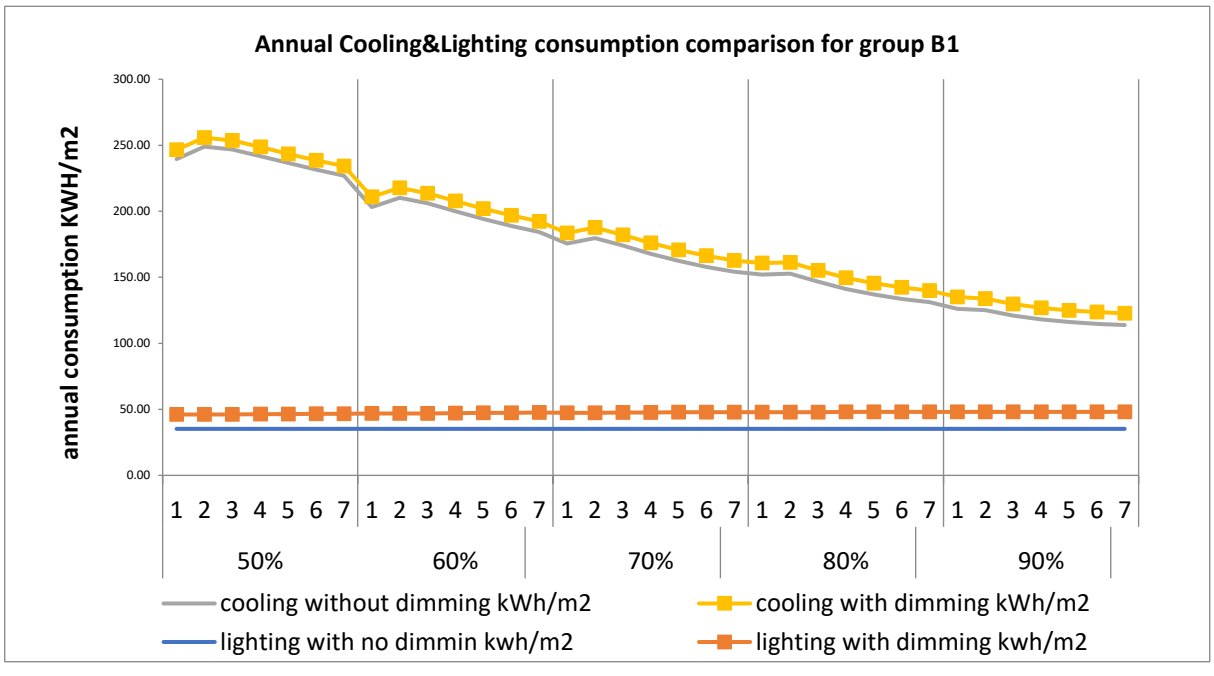


Figure of Annual cooling & lighting consumption results comparison for group B1

11.3.6 Group B2 (rotation = 45, WWR = 50%)

The pattern and almost the same difference in consumption values are repeated in this group again with the same proximity in the lower heights lighting consumption values of group A2.

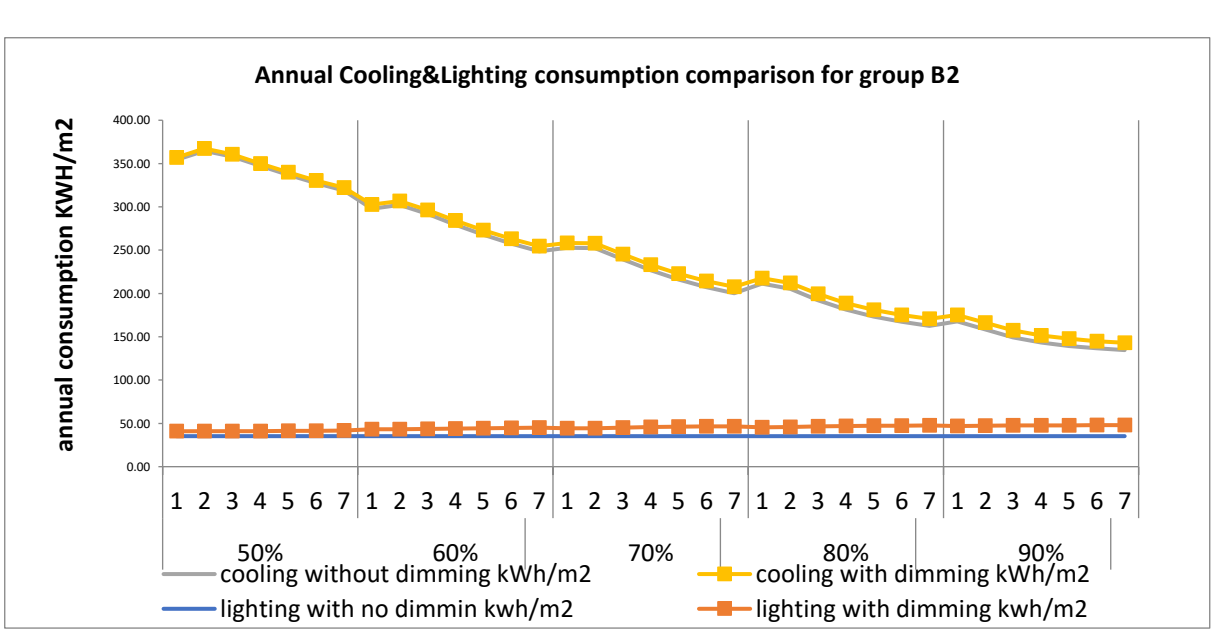


Figure of Annual cooling & lighting consumption results comparison for group B2

11.3.7 Group B3 (rotation = 45, WWR = 80%)

The main difference in this group is that the lighting values are almost like group A3 due to the high value of WWR. But the cooling still has the same pattern of proximity to group A3.

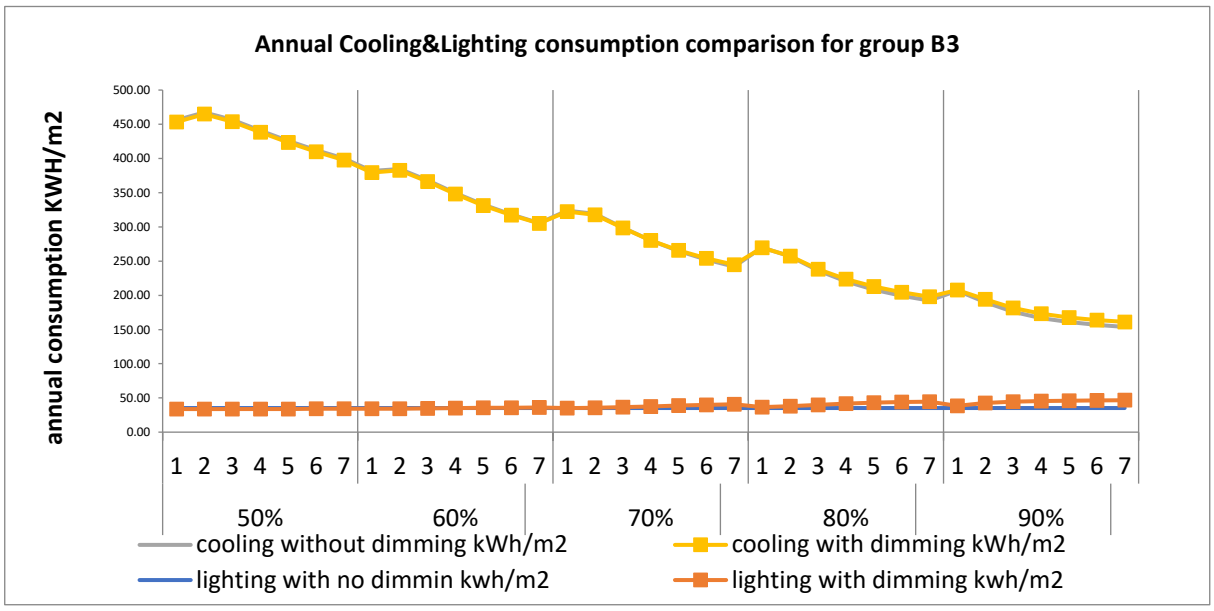
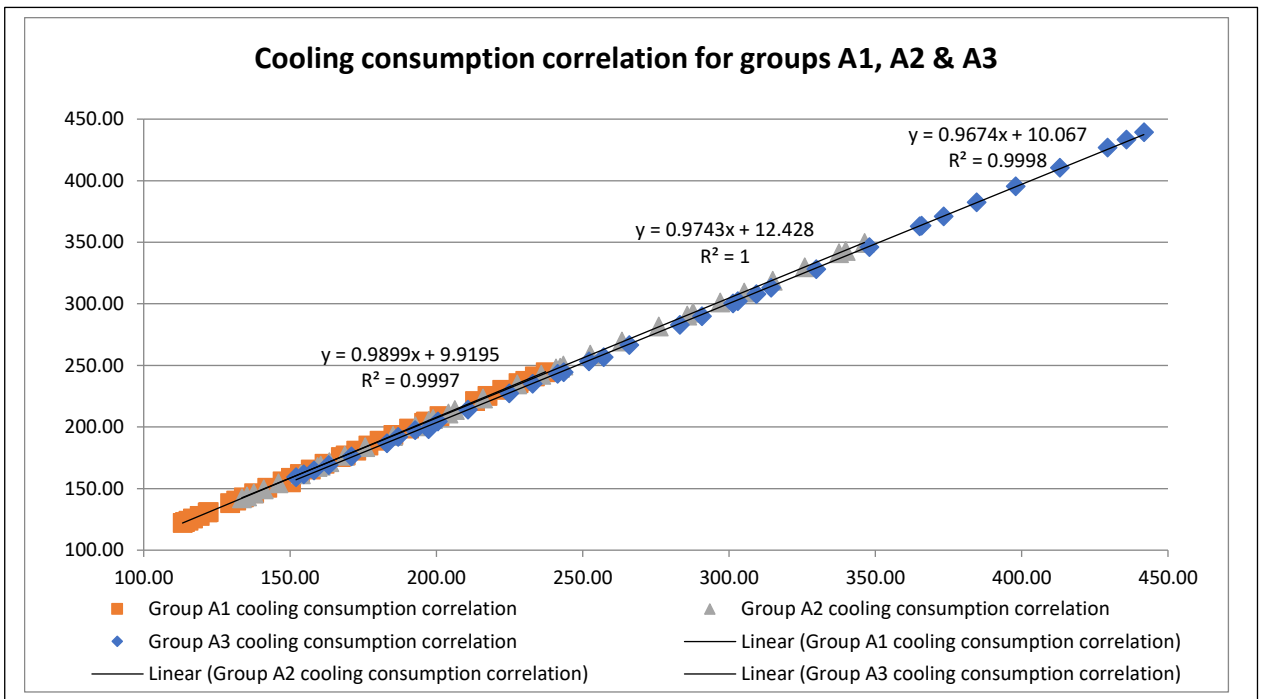


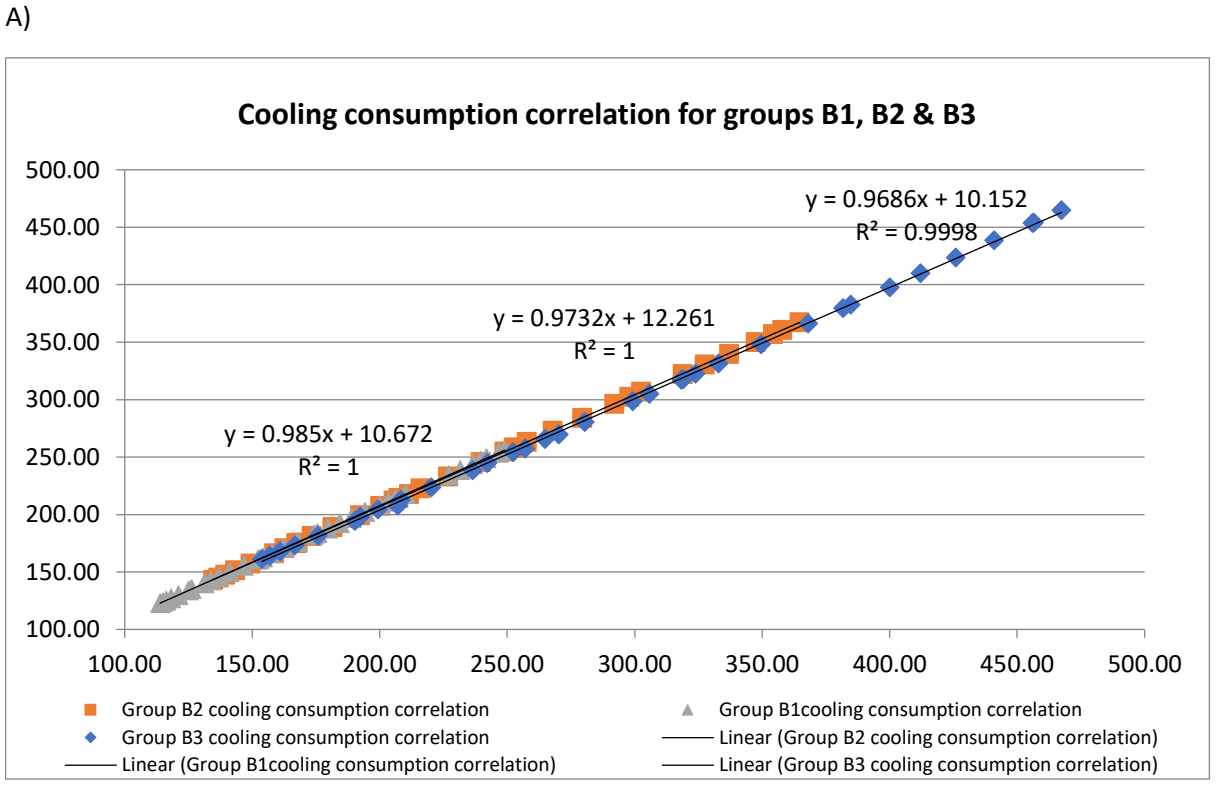
Figure of Annual cooling & lighting consumption results comparison for group B3

11.3.8 cooling lighting availability and heating energy demand correlation

There is also a significant finding about the impact of differentiating lighting control systems on cooling consumption for the six different groups. There is a continuous linear correlation for the two lighting control systems on the results of cooling consumption.These linear correlations are a further reason for rerunning the simulation with a set of materials that is based on the ASHREA benchmark recommendation for this climate zone and other climate zones with different cooling and heating requirements.



A)



B)

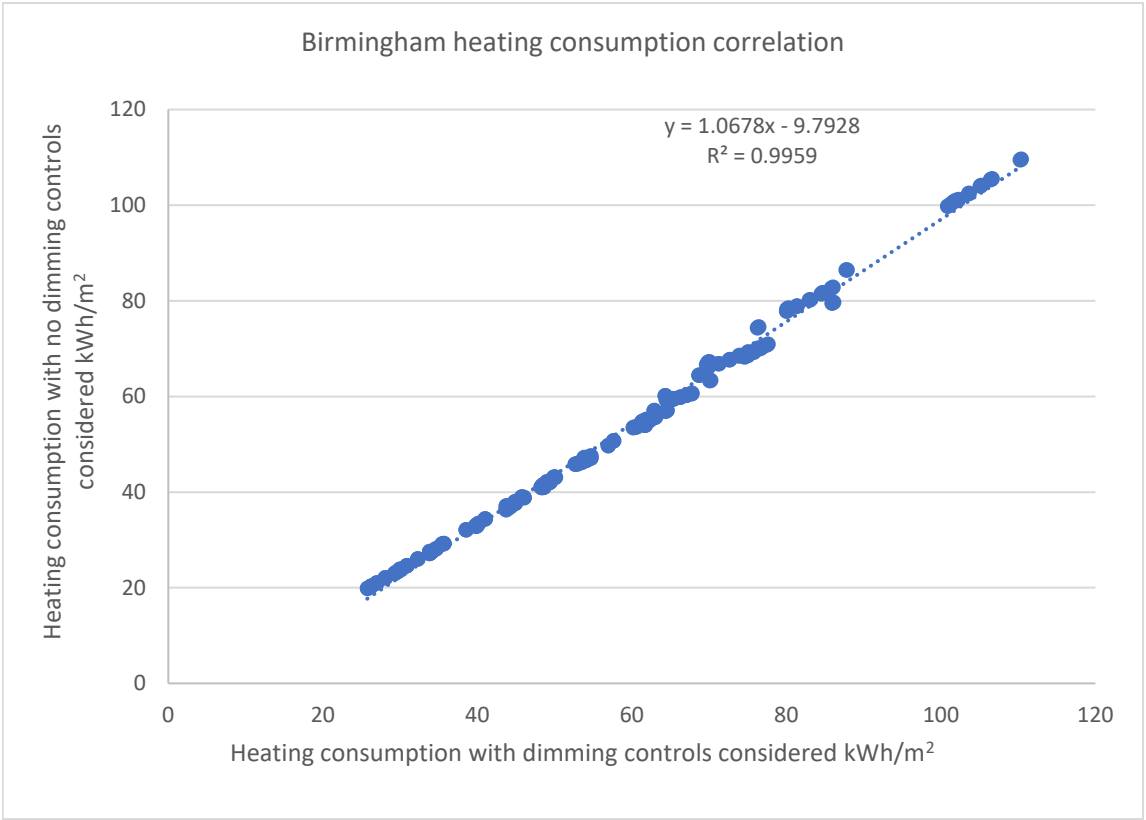
Figure of The linear correlation of the comparison between the cooling consumption with standard ON/OFF lighting controls (vertical axis) and the cooling consumption with Dimming Lighting controls (horizontal axis) for A) groups A1, 2 and 3 and B) groups B1, 2 and 3

11.3.9 Heating lighting availability and heating energy demand correlation

Heating energy demands in the London and Birmingham cases have shown a clear linear correlation between the two ways of calculating the energy performance with the change of lighting controls. The change of lighting control systems affected the values of calculated energy consumption for heating demands without changing the patterns of this consumption.

The correlation between lighting and energy demand proved to exist throughout the three selected climate conditions. This linear correlation shows that there is a balance between energy demand and the lighting analysis. The patterns of energy consumption for heating and cooling purposes did not show a significant change in response to the essential change of lighting controls and its sensitivity to the actual daylight. This means that the change of lighting controls did not affect optimal solutions by adding more sensitivity to the daylight provision in the zones. This emphasized the opportunity to run a holistic simulation through a sequential approach as a way of breaking down the environmental performance simulation on an urban scale in a multistage framework.

A)



B)

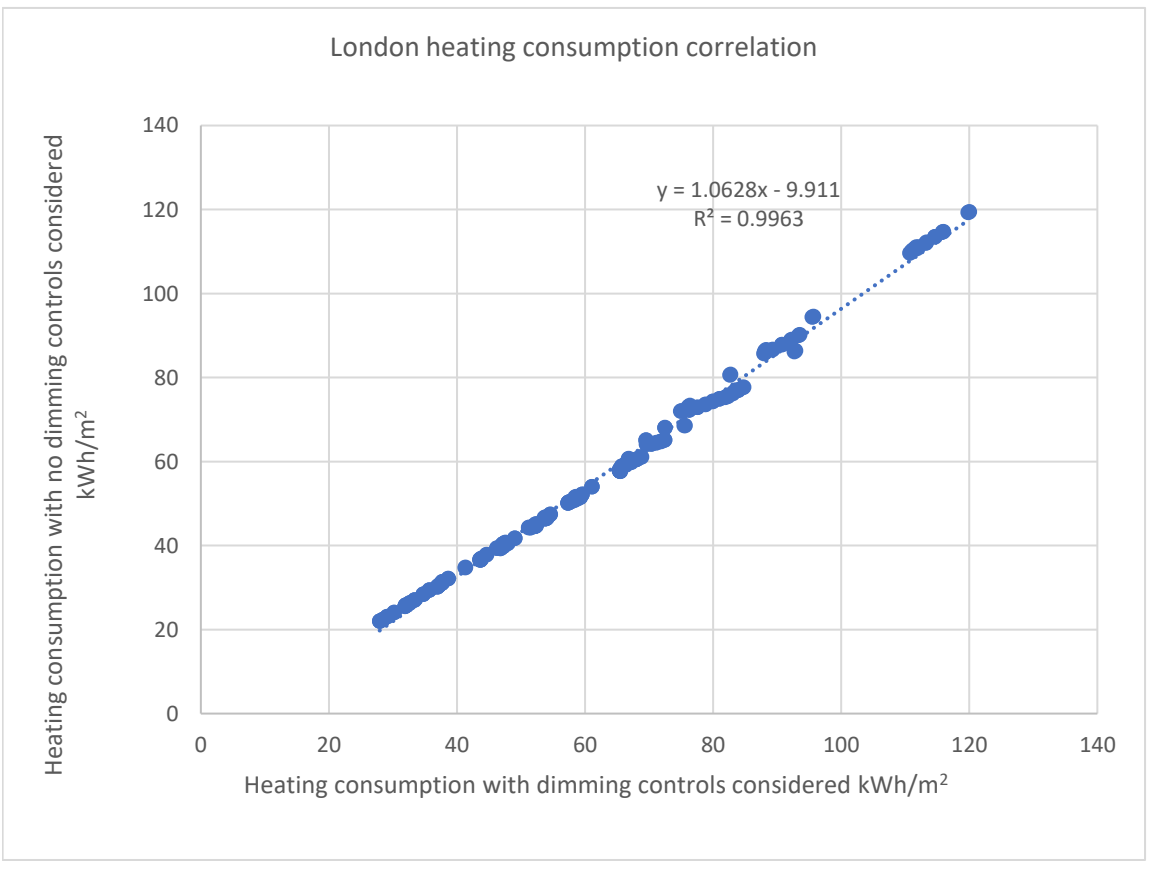


Figure of shows the linear correlation between heating energy demand for the two different lighting methods used in A) Birmingham, UK, B) London, UK

11.4 Birmingham analysis results

11.4.1 Material inputs

Birmingham, UK	
Material name	**U-value**
ASHRAE 90.1-2010 Extwall Mass Climate Zone 5	0.55
Interior Wall	2.58
Interior Floor	1.44
ASHRAE 90.1-2010 Extroof lead Climate Zone 2-8	0.28
ASHRAE 90.1-2010 Extwindow Nonmetal Climate Zone 5-6	1.98

11.4.2 Relative importance results

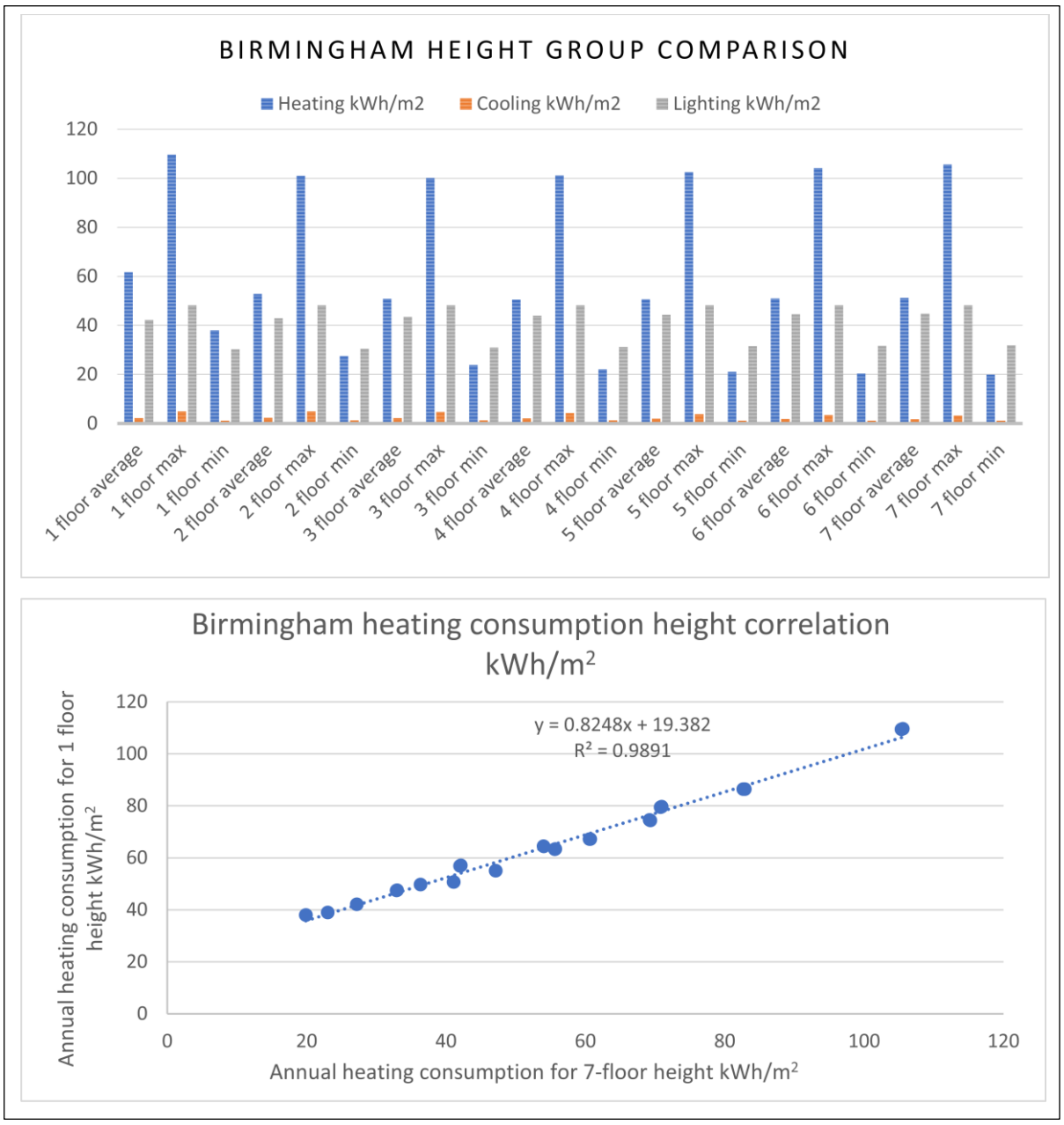


Figure of Different height groups' relative importance for heating, cooling and lighting (with dimming) energy consumption for Birmingham, UK

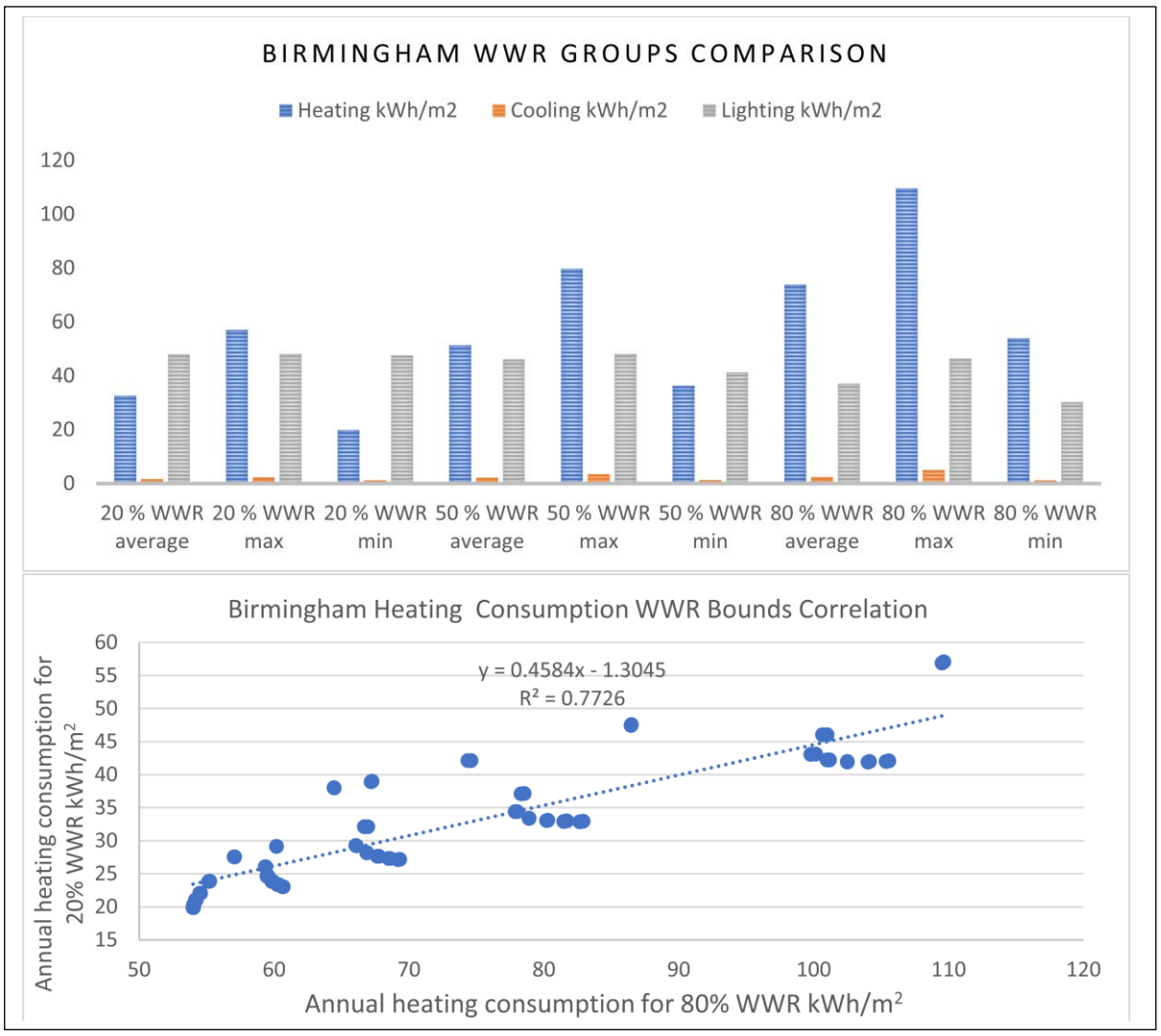


Figure of WWR variations' heating consumption average comparison in kWh/m2, WWR variations' bound correlated in kWh/m2 for Birmingham, UK

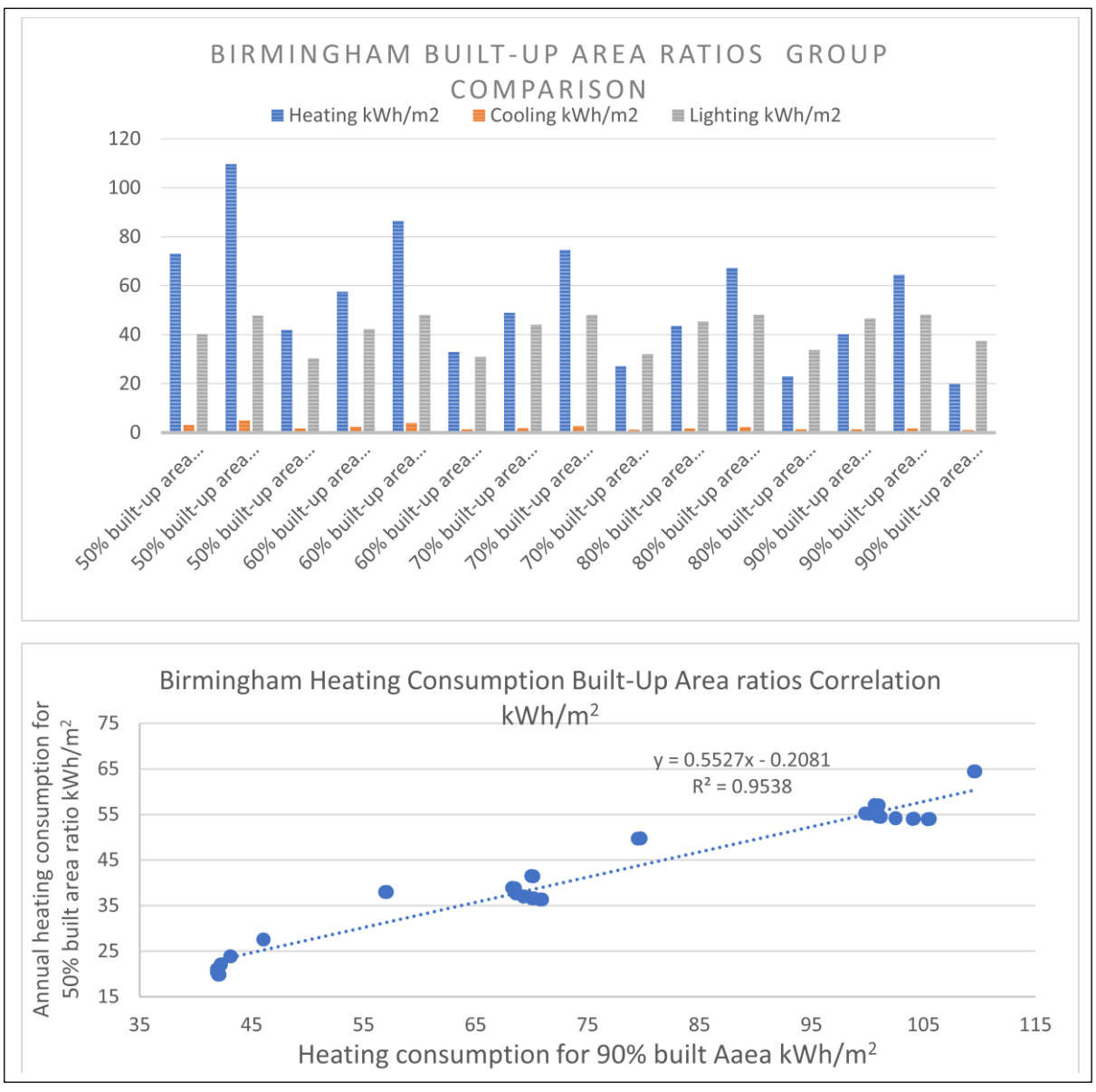


Figure of Built-up area ratio variations' heating consumption average comparison in kWh/m^2, built-up area ratio variations' bound correlated in kWh/m2 for Birmingham, UK

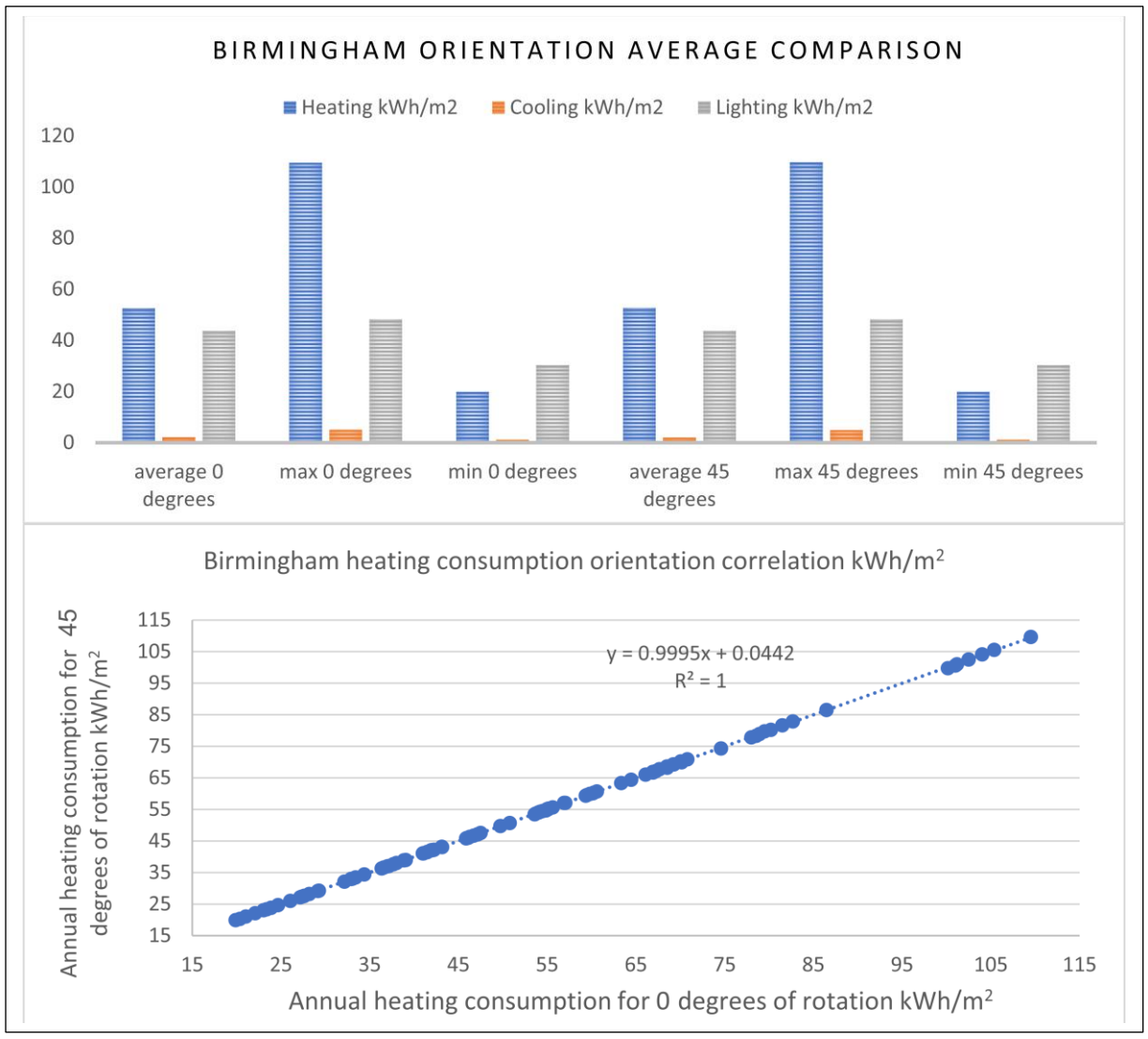


Figure of Orientation variations' heating consumption average comparison in kWh/m2, orientation variations' bound correlated in kWh/m2 for Birmingham, UK

11.4.3 Detailed zone energy demand analysis

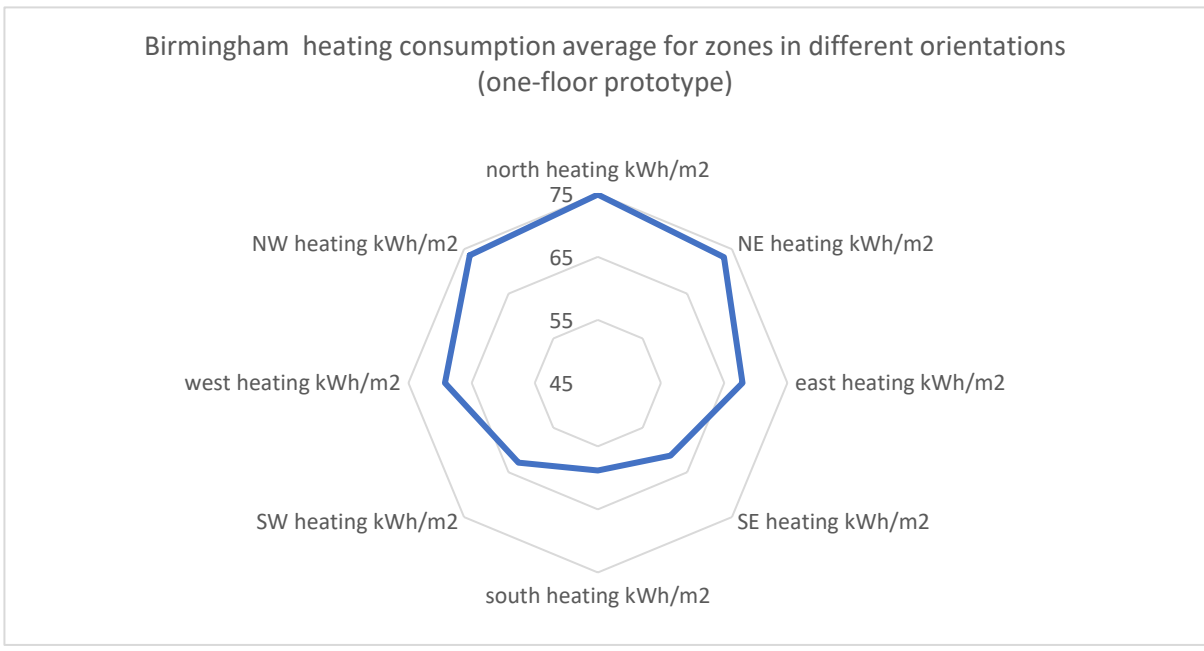


Figure of One-floor prototype heating consumption results average per orientation for Birmingham, UK

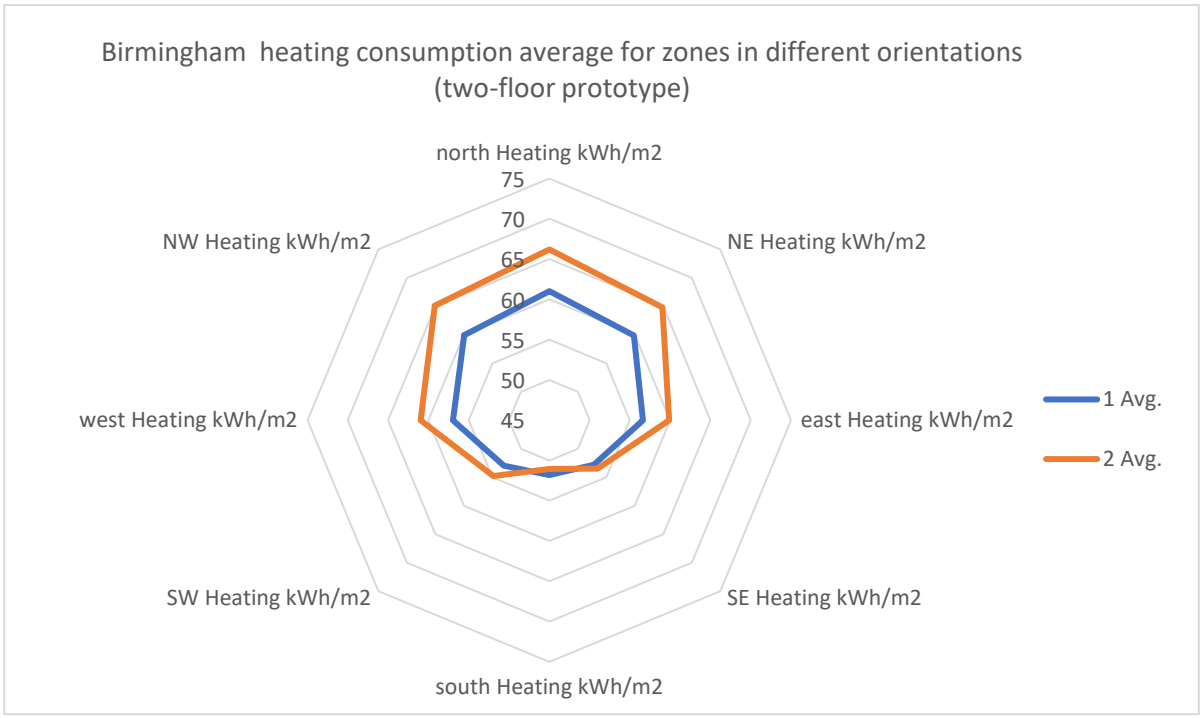


Figure of Two-floor prototype heating consumption results average per orientation for Birmingham, UK

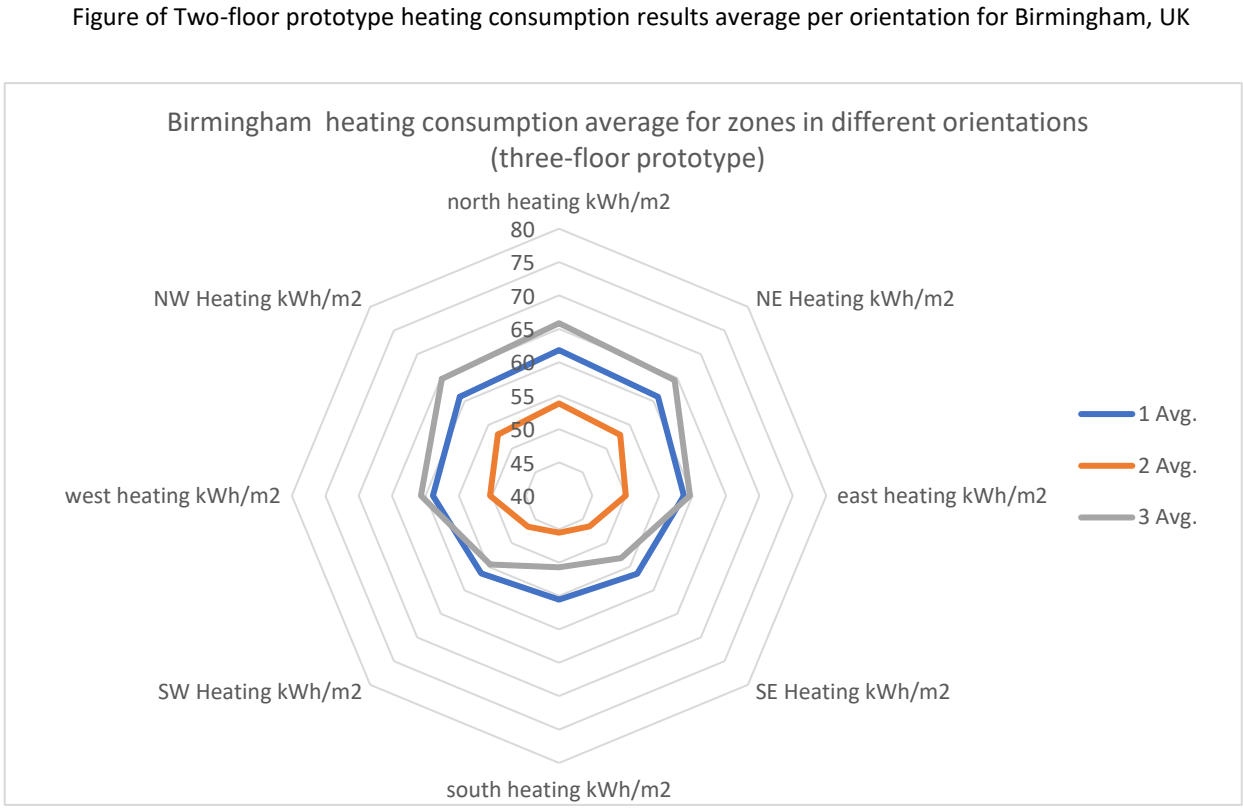


Figure of Three-floor prototype heating consumption results average per orientation for Birmingham, UK

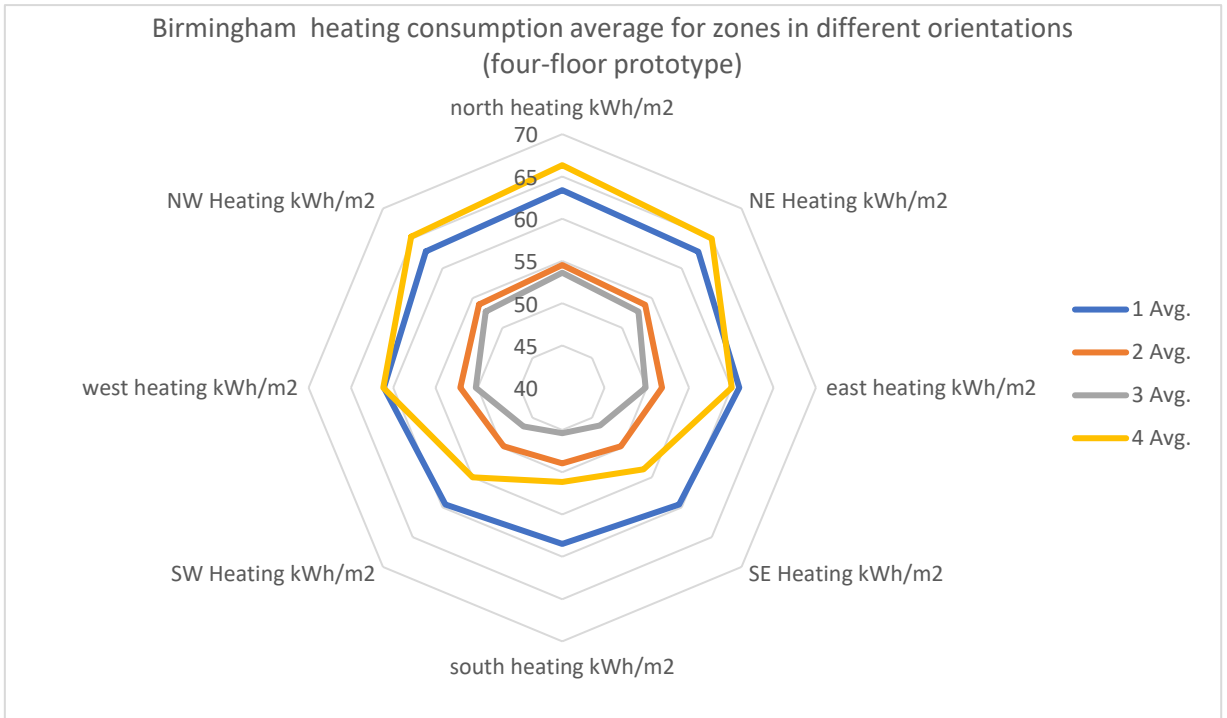


Figure of Four-floor prototype heating consumption results average per orientation for Birmingham, UK

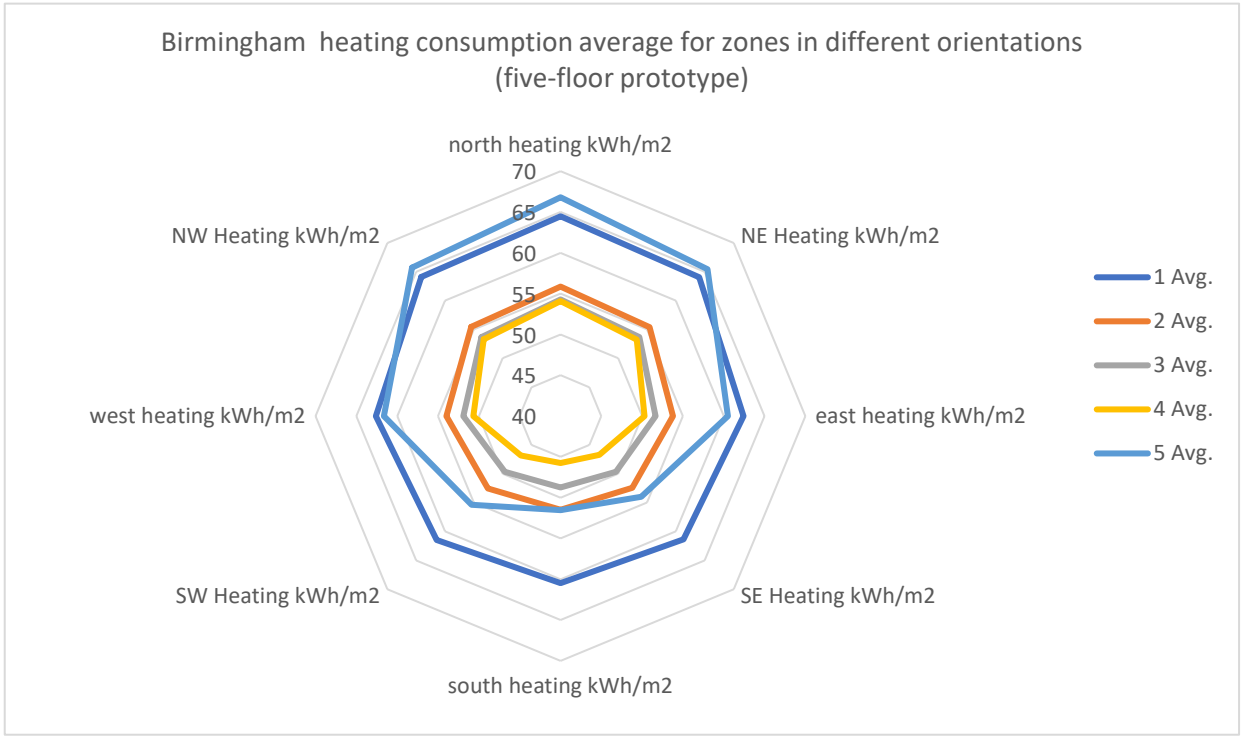


Figure of Five-floor prototype heating consumption results average per orientation for Birmingham, UK

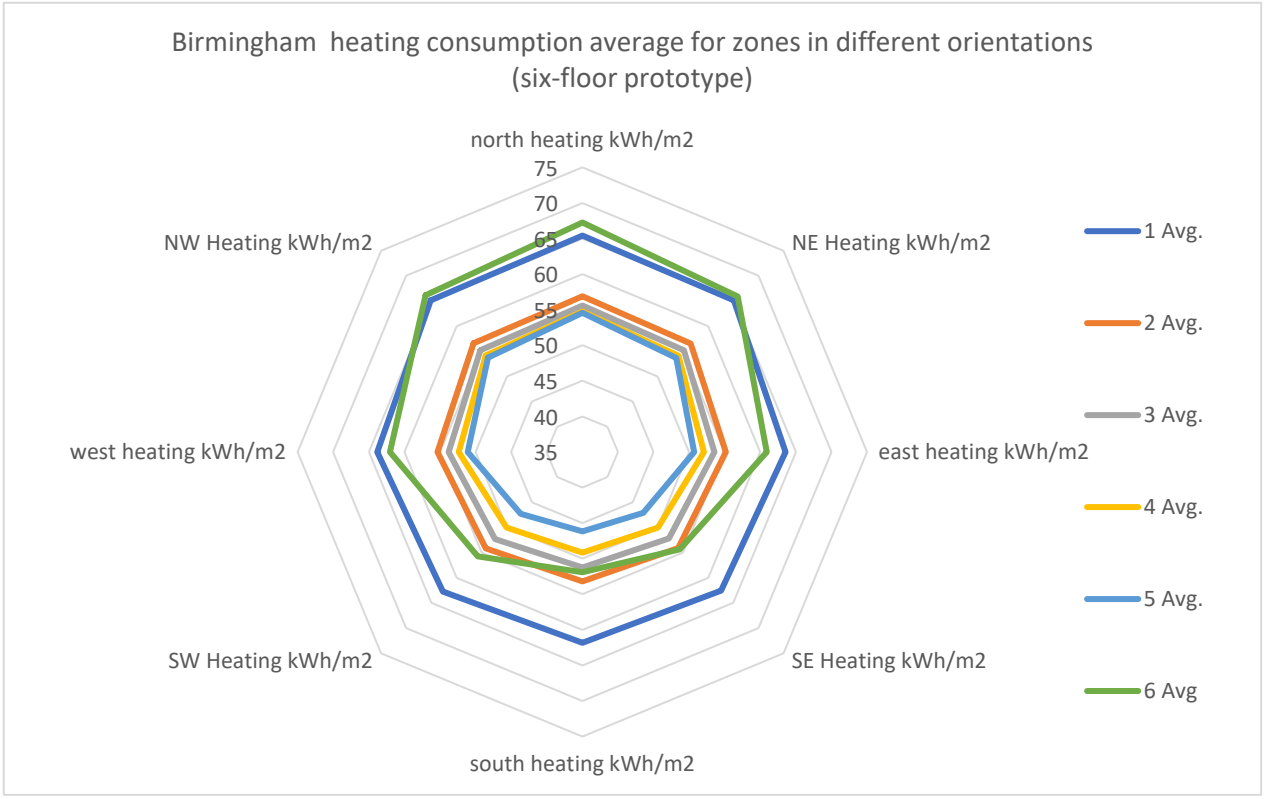


Figure of Six-floor prototype heating consumption results average per orientation for Birmingham, UK

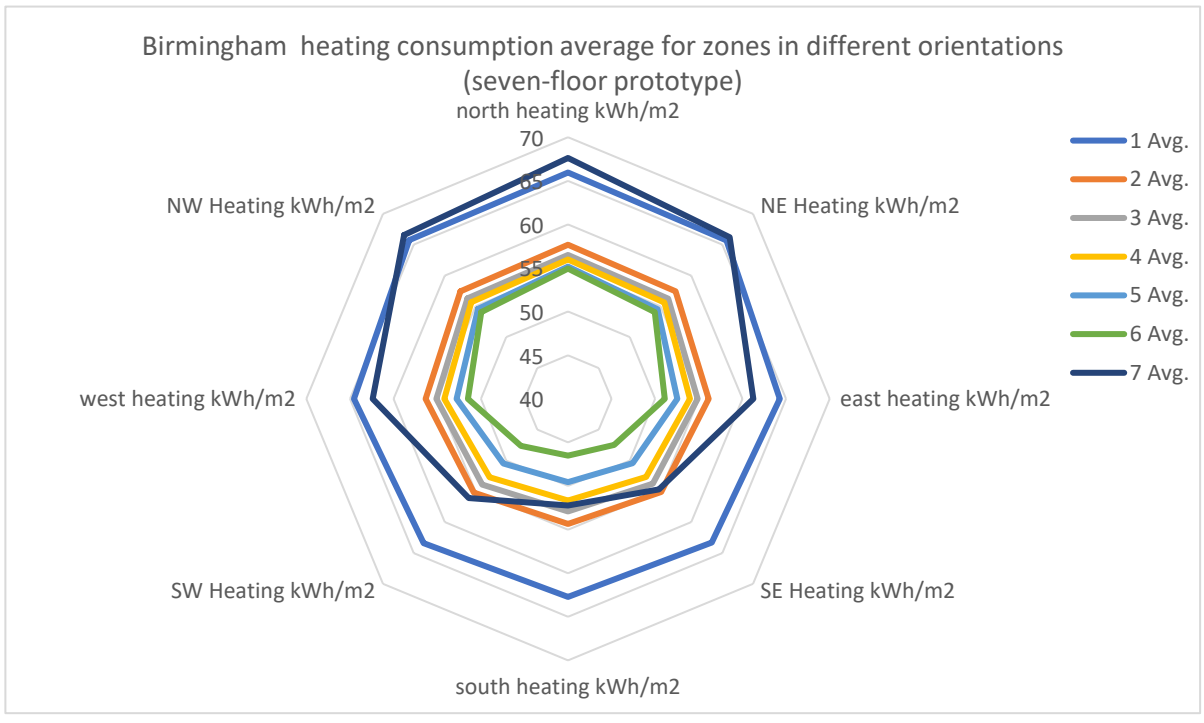
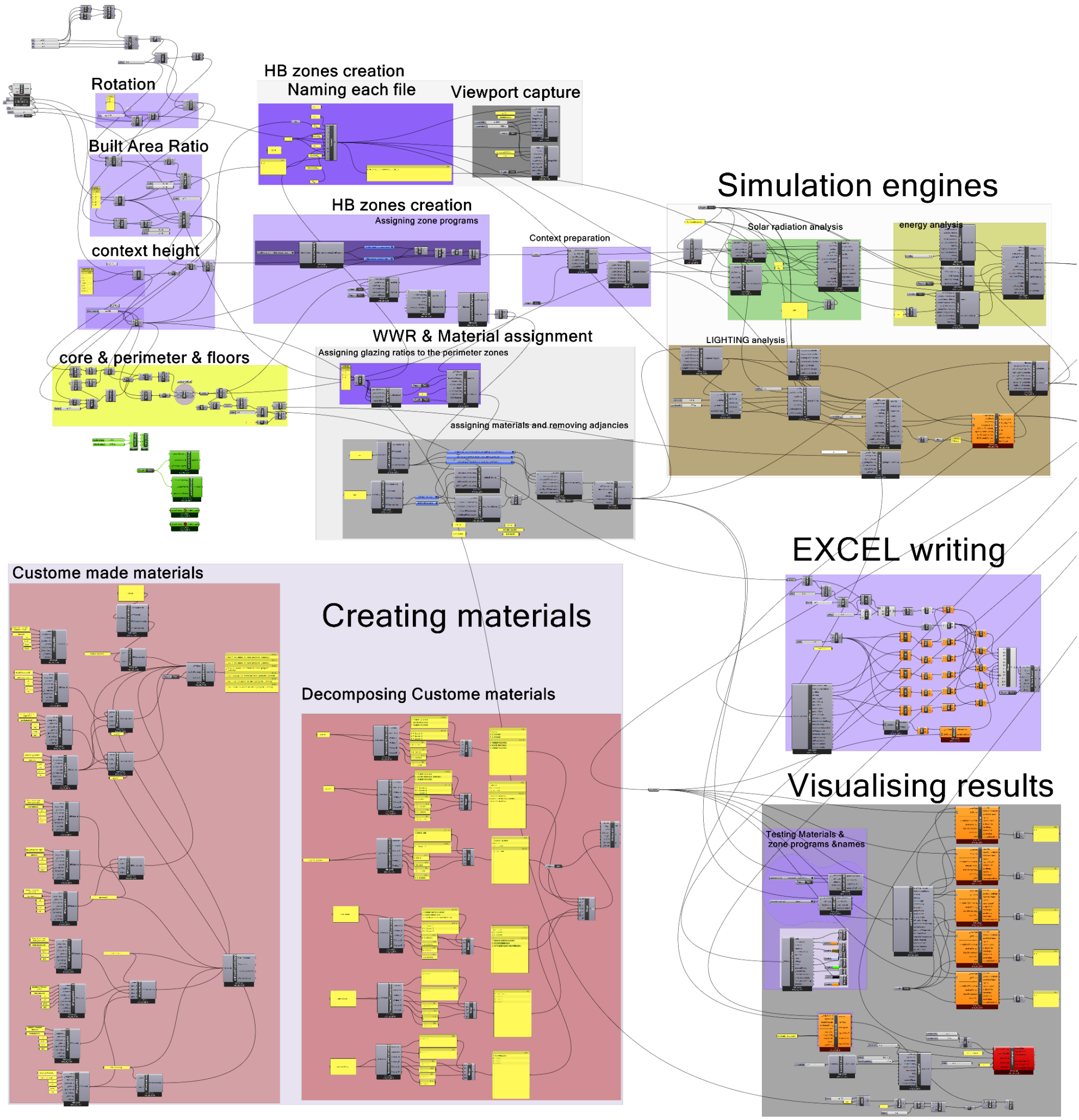


Figure of Seven-floor prototype heating consumption results average per orientation for Birmingham, UK

12 Appendix B (preliminary study code)

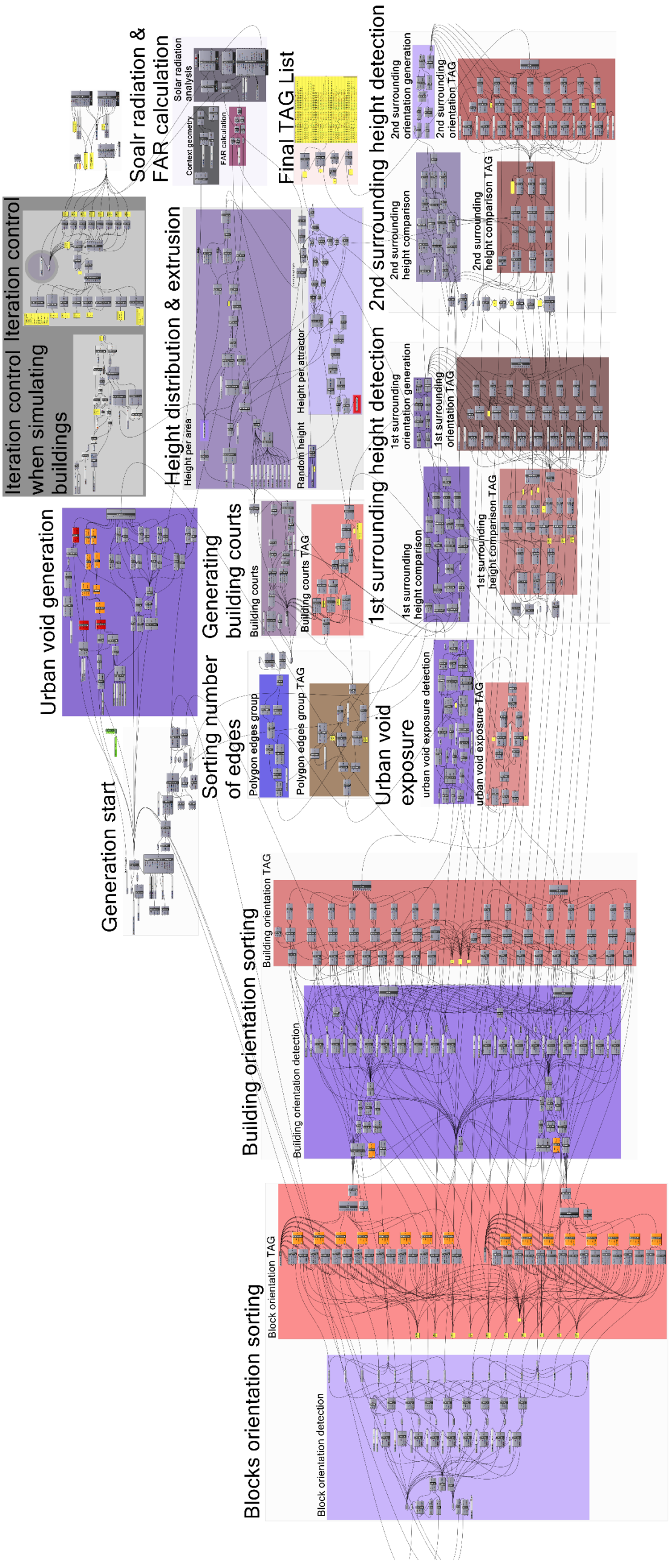
Preliminary study full Grasshopper canvas definition



Grasshopper canvas high resolution screen shot for the pilot study including custom materials and ASHRAE materials (source : author)

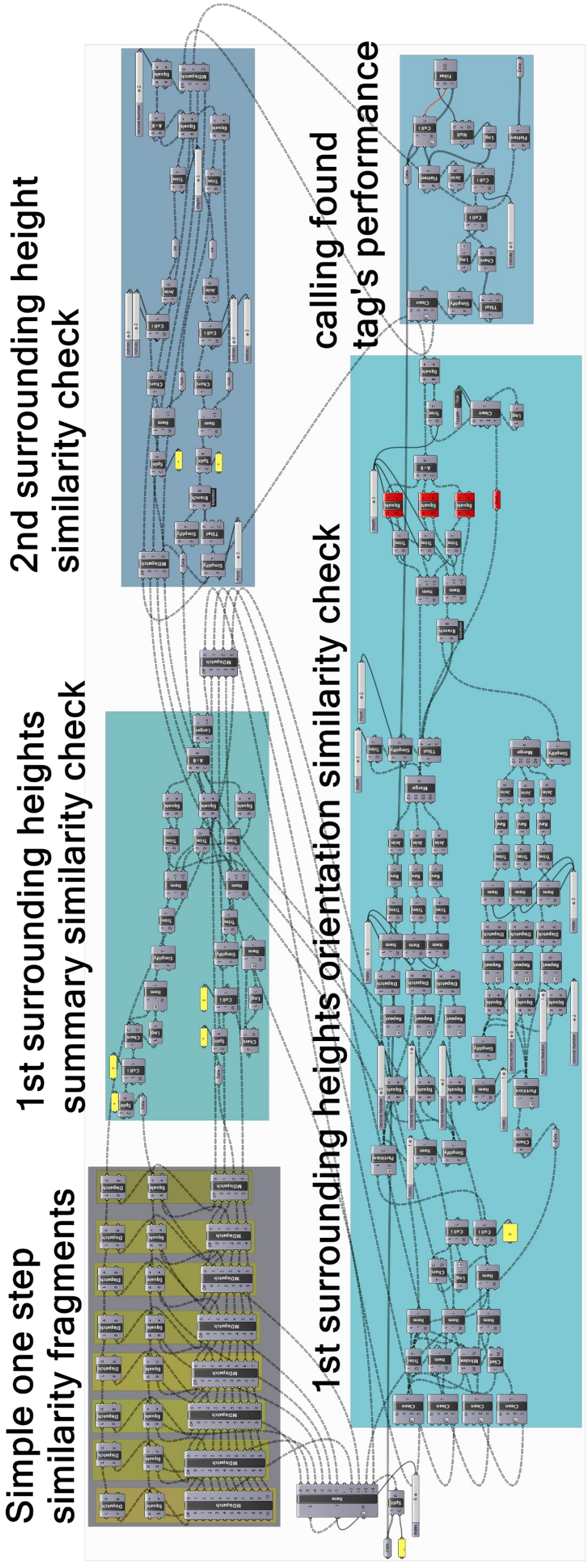
13 Appendix c (Geometry & classification tag)

Geometry and classification tag Grasshopper definition



Grasshopper canvas high resolution screen shot of the geometry and classification tag creation algorithm (source : author)

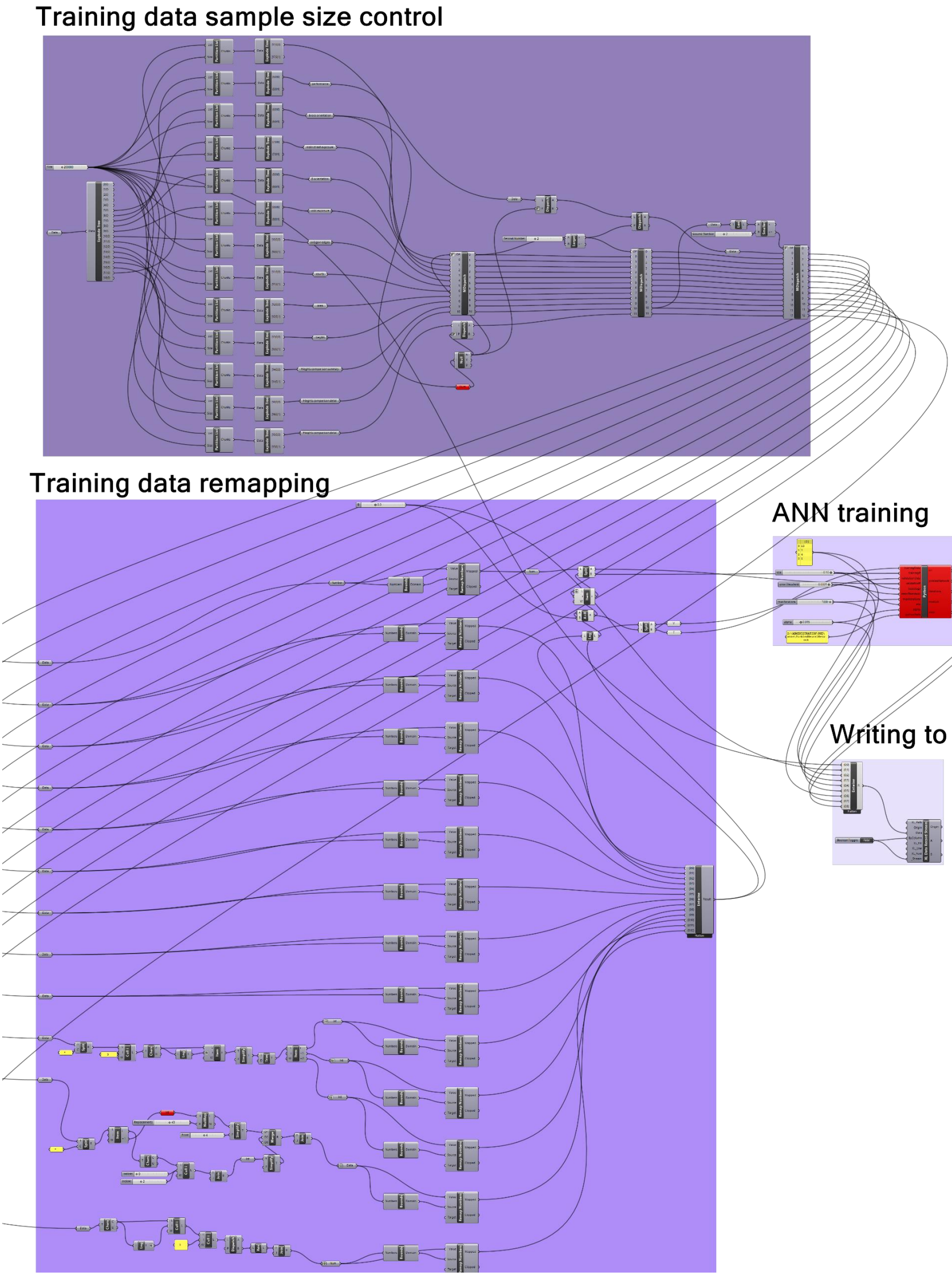
Tag look up & matching Grasshopper definition



Look up process Grasshopper canvas high resolution screen shot

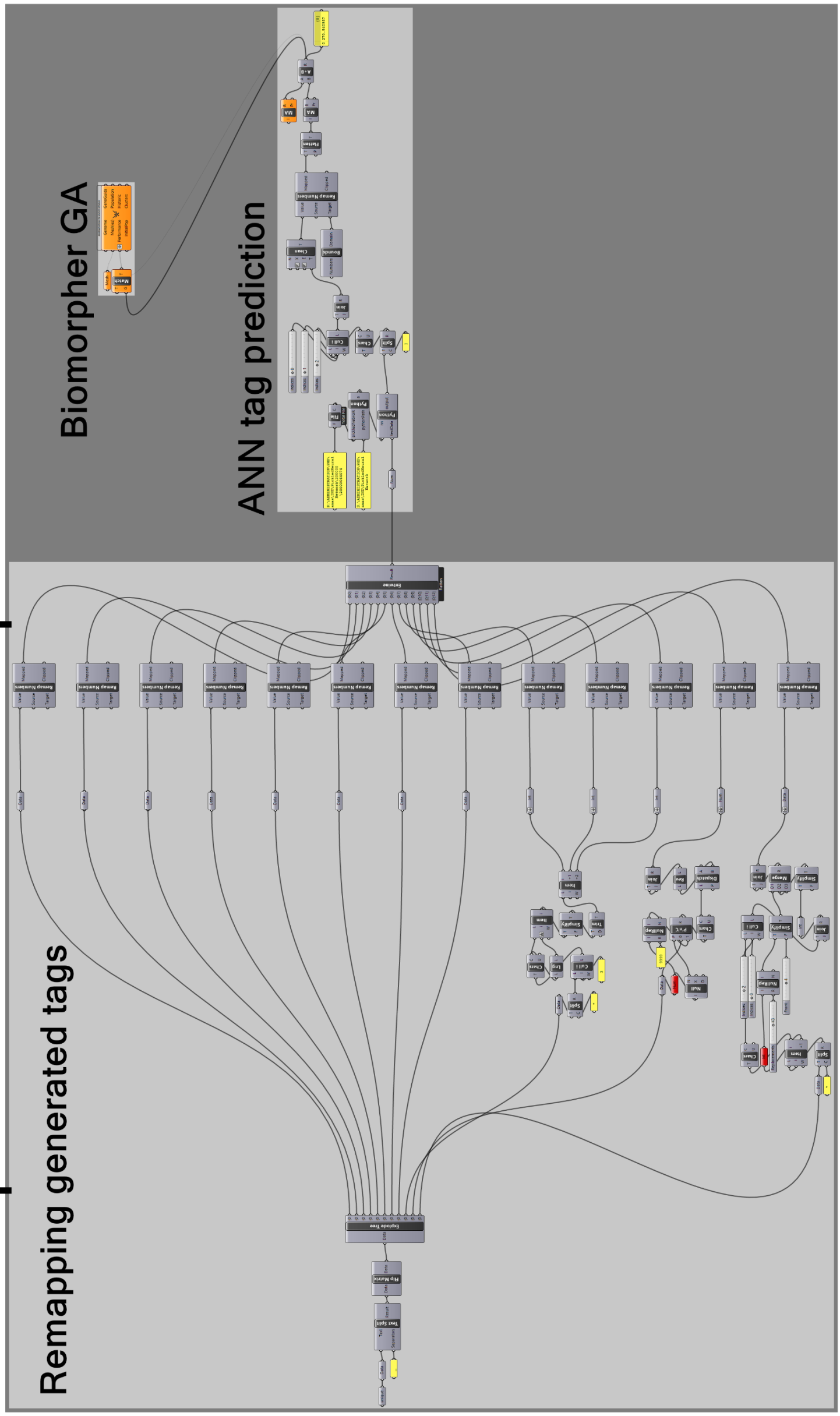
14 Appendix D (ANN prediction)

ANN training Grasshopper definition



Grasshopper canvas high resolution screen shot for ANN training class (source : author)

ANN prediction with GA optimisation Grasshopper definition



initial python addition to the ANN node

```
def trainNetwork(net, inputs, outputs, errorThreshold, maxIterations, eta, alpha):
    Neuron.eta = eta
    Neuron.alpha = alpha
    for i in range(maxIterations):
        err = 0
        for j in range(len(inputs)):
            net.setInput(inputs[j])
            net.feedForward()
            net.backPropagate(outputs[j])
            err = err + net.getError(outputs[j])
        # print "error: ", err
        if err < errorThreshold:
            break
    return net
# Build Network
net = Network(topology)
net = trainNetwork(net, inputs, outputs, errorThreshold, maxIterations, eta, alpha)
nn = net
```

The training network added class (source : author)

ANN prediction Python code

```
tempList = [list(i) for i in testData.Branches]
input = []
for i in range(len(tempList[0])):
    newList = []
    for j in range(len(tempList)):
        newList.append(tempList[j][i])
    input.append(newList)
net = nn
output = []
if isinstance(input[0], list):
      for anItem in input:
            net.setInput(anItem)
            net.feedForward()
            output.append(net.getResults())
else:
      net.setInput(input)
      net.feedForward()
      output = net.getResults()
print output
```

Prediction component code

ANN Training final Python code

```
def trainNetwork(self, trainingInputs, trainingOutputs, validationInputs, validationOutputs, errorThreshold,
maxIterations, eta, alpha):
        Neuron.eta = eta
        Neuron.alpha = alpha
        iter = 0
        stop = 0
        while (stop == 0):
            trainingError = 0
            validationError = 0
            for i in range(len(trainingInputs)):
                self.setInput(trainingInputs[i])
                self.feedForward()
                self.backPropagate(trainingOutputs[i])
                trainingError = trainingError + self.getError(trainingOutputs[i])
            trainingError = trainingError / len(trainingInputs)
            validationError = trainingError*2
            if iter >= maxIterations*0.25:
                validationError = 0
                for i in range(len(validationInputs)):
                    self.setInput(validationInputs[i])
                    self.feedForward()
                    validationError = validationError + self.getError(validationOutputs[i])
                validationError = validationError / len(validationInputs)
                self.mseList.append(validationError)
            iter = iter + 1
            if iter >= maxIterations:
                stop = 1
            elif validationError < errorThreshold:
                stop = 2
            elif validationError < trainingError:
                stop = 3
        self.iterations = iter
        self.stop = stop
        return self
```

Training the neural network class in its final version