Frontal Geometry from Sketches of Engineering Objects: Is Line Labelling Necessary?

P. A. C. Varley

Department of Precision Engineering, The University of Tokyo, Tokyo, Japan
pvarley@den.rcast.u-tokyo.ac.jp

R. R. Martin

School of Computer Science, Cardiff University, Cardiff, Wales, UK
Ralph.Martin@cs.cf.ac.uk

H. Suzuki

Department of Precision Engineering, The University of Tokyo, Tokyo, Japan
suzuki@den.rcast.u-tokyo.ac.jp

Abstract

A tool which can quickly interpret line drawings (with hidden lines removed) of engineering objects as boundary representation CAD models would be of significant benefit in the process of engineering design. Inflation *of the drawing to produce a* frontal geometry*, a geometric realisation of that part of the object visible in the drawing, is an important stage of this process.*

Previous methods of producing frontal geometries have relied on the technique of line-labelling *(labelling edges as convex, concave or occluding). Although restricted subsets of the line-labelling problem have known solutions, reliable methods have not been found for the general line-labelling problem, and traditional methods, when adapted to drawings with non-trihedral junctions, are unacceptably slow.*

Many other papers assume that line-labelling is an essential step. Here we show this is not necessarily true, and that comparable results can be obtained by a novel alternative approach. Firstly, we consider what outputs from line-labelling are essential to the production of frontal geometry. Secondly, we investigate by what other means these outputs can be produced.

Our work indicates that the only essential output from line-labelling for frontal geometry is the determination of which T-junctions in a drawing are occluding and which are non-occluding. This information is required for inflation, and also for detection of symmetry and for constructing hidden topology.

Thus, we propose and analyse a new method which, in the absence of line labels, simultaneously inflates a drawing to produce the frontal geometry and attempts to determine whether each T-junction is occluding or not. For drawings of objects with holes or pockets, and for cases where line-labelling is particularly unreliable, our new method can provide a better alternative.

Keywords

Line Drawing Interpretation, Engineering Design, Conceptual Design, Frontal Geometry

1 Introduction

1.1 Topic

Our long-term goal is a system which can automatically produce a boundary-representation solid model

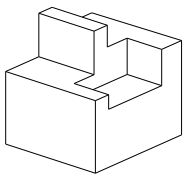

Figure 1.

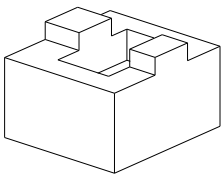

Figure 2.

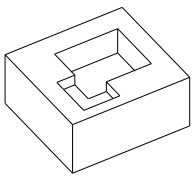

Figure 3.

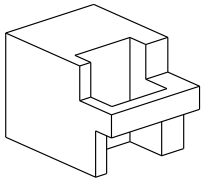

Figure 4.

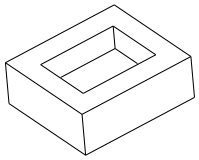

Figure 5.

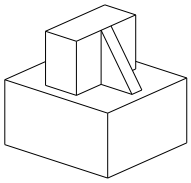

Figure 6.

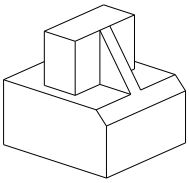

Figure 7.

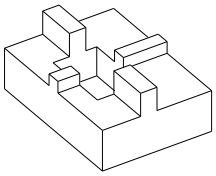

Figure 8.

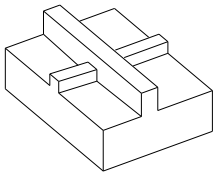

Figure 9.

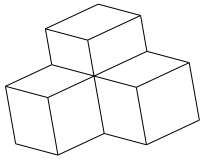

Figure 10.

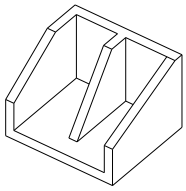

Figure 11.

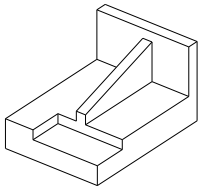

Figure 12.

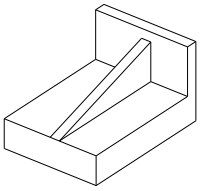

Figure 13.

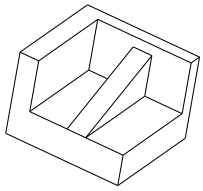

Figure 14.

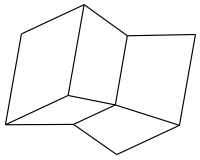

Figure 15.

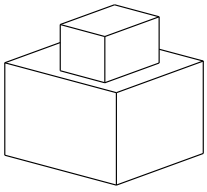

Figure 16.

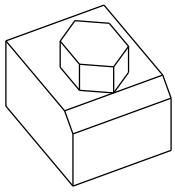

Figure 17.

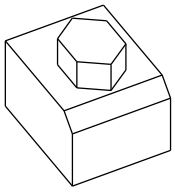

Figure 18.

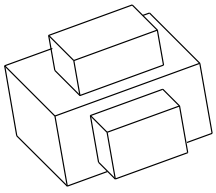

Figure 19.

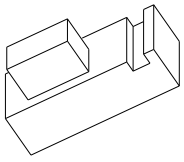

Figure 20.

Line Drawings from Sashikumar et al [26, 27]

from a corresponding input *line drawing* showing the *visible edges* of a *polyhedral object*.

Jenkins [10] and others have observed that a tool capable of producing a solid model from an input line drawing in an acceptably short time—a second or two—would be of significant benefit in the process of engineering design, enabling designers to spend more time on the creative aspects of their job and less on the routine aspects. They may also find a simpler user interface than that provided by current CAD systems less of a distraction.

The particular topic of this paper is whether *line labelling* is a necessary step in that process.

1.2 Assumptions

To make progress with the sketch input problem, we must make various (reasonable) assumptions concerning the drawing. We assume that the drawing is a *natural line drawing*, showing the visible part only (hidden lines removed) of a single object. The reason we prefer natural line drawings to full wireframe drawings is twofold: firstly, with the aim of presenting a simple user interface, drawing natural line drawings is easier than drawing wireframes (there are fewer lines to draw! [5]), and secondly, with the aim of presenting a natural user interface, natural line drawings depict what humans are accustomed to seeing, so we believe drawing them to be more natural than drawing wireframes.

We assume that the line drawing is drawn from

a *general position* viewpoint, meaning that no small change in the viewpoint changes the topological structure of the drawing (i.e. there are no accidental coincidences). This assumption can be justified on grounds other than that it is necessary for our algorithms: Enns and Rensink [4] have shown that drawings in general position are psychologically easier for humans to interpret than drawings with accidental coincidences. Thus, it can be assumed that humans will avoid such accidental coincidences when drawing for their own benefit.

As described in Section 1.5 below, we assume that existing methods can convert freehand sketches to line drawings, and that the resulting line drawings which we take as our input are topologically correct.

However, we cannot assume that the drawings are geometrically perfect. For example, "parallel" lines may be close to, but not exactly, parallel. In this context, it can be noted that none of our test set of twenty drawings discussed later is mathematically perfect, although all appear well-drawn to the naked eye. This tolerance of errors, although necessary for any practical system, reduces the choice of approaches. *Superstrict* methods such as those of Sugihara [29] and Whiteley [41] cannot be used without adaptation. Indeed much of [29], recognising that drawing imperfections are inevitable, is devoted to a discussion of how such superstrictness may be circumvented. Our preference is for methods which inherently allow for imperfections rather than treat them as a problem to be worked around.

1.3 Terminology

Before proceeding, we first need to define the terminology we use. A boundary-representation model of a solid *object* describes the *topology* and *geometry* of its *faces*, *edges* and *vertices* in 3D. *Topology* is discrete (recording connectivity e.g. between vertices and edges), whereas *geometry* is continuous (providing e.g. the spatial coordinates of vertices). A *line drawing* is a 2D pictorial representation of an object, comprising *lines* (which correspond to visible or partially-visible edges) and *junctions* (some of which correspond to the visible vertices of the object). Loops of lines and junctions form *regions*, which correspond to the visible or partially-visible faces of the object. For a general introduction to and overview of these and other CAD concepts, see (for example) Lee [15]. Note the careful distinction between 2D ideas (drawings, regions, lines, junctions) and 3D ideas (objects, faces, edges, vertices).

Junctions of different types are denoted by a letter which is a mnemonic for the configuration of lines meeting there. For example, a T-junction is a meeting of three lines: two of them lie in the same straight line, while the third terminates at the junction.

A polyhedron is *trihedral* if exactly three faces meet at each vertex. It is *extended trihedral* [21] if exactly three planes meet at each vertex; there may be four or more faces provided that some are coplanar. It is *tetrahedral* if no more than four faces meet at any vertex.

An important concept in this paper is the *frontal geometry*, an intermediate stage between the 2D drawing and the corresponding 3D object (for this reason, it is sometimes called "$2\frac{1}{2}$D"). In a frontal geometry, everything visible in the drawing is given a position in 3D space, but the occluded part of the object, not visible in the drawing, is not present.

1.4 Input Data

Figures 1–20 illustrate the kind of line drawings we wish to process. These drawings are also used later to test our ideas; they are the same set of test drawings as in one of our previous papers [38]. They are taken from two papers by Sashikumar et al [26, 27]. We believe them to be more like real engineering objects than some of the simpler objects used in earlier sketch input papers, and we also believe them to be a representative sample of line drawings of engineering objects [38]: for example, features such as K-vertices occur with roughly the same frequency in these drawings as they do in a survey [25]. (*K-vertices* occur when axis-aligned cuboids and triangular prismatic wedges meet; in drawings, these often have the shape of a capital K.) Using realistic test cases is important, as the reported success of previous methods has often been exaggerated through their being tested on drawings of too simple a nature.

Sashikumar et al assume, correctly, that their readers can interpret such drawings as representations of solid objects. The process of visualising a frontal geometry is easy for humans—we do it automatically as a pre-attentive process [4]. It is only when we attempt to replicate an equivalent ability in a computer algorithm that we realise its difficulty. Nevertheless, there is ample anecdotal evidence for the idea that depth perception is a skill which humans learn (see, for example [16]), and that which can be learnt can, at least in principle, be automated.

In addition to these "representative sample" drawings, we also consider two sets of drawings which cause particular problems for existing methods, and in particular for catalogue-based line labelling. Firstly, Figures 21–52 are the 32-drawing test set given in our previous paper on labelling drawings with multiple tetra-

hedral K-vertices [37]. Although the complete catalogue of tetrahedral vertices is known [32], and existing methods can, in principle, label drawings containing multiple K-vertices, the results of doing so are far from satisfactory [37].

Secondly, there are eight new drawings, Figures 54–61, of polyhedral objects with higher-order K-vertices where more than four edges meet; catalogues of such higher-order vertices are too large to be useful, so catalogue-based labelling is clearly inappropriate for these, as will be discussed later.

1.5 Steps in Drawing Interpretation

Conversion of *freehand sketches* to *line drawings* (as in Figure 53) involves, *inter alia*, removal of duplicate lines, straightening of lines, and connection of lines at appropriate junctions. This has been investigated by several groups, [28] being a recent example. We do not discuss this step in the overall process further here, other than to note that some practical implementations ([20], for example) have problems with T-junctions. We assume here that this conversion can be done, and take the resulting *line drawings* as the starting-point for our own work.

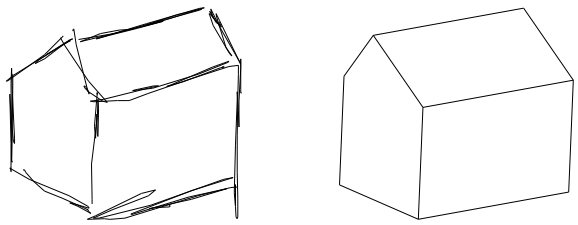

Figure 53. Sketch and Line Drawing [31]

Line drawings can be considered either as pictorial representations, or, in graph-theoretic terms, as sets of junctions (vertices) and lines (edges) where each junction has x- and y-coordinates and each line connects two junctions; the two interpretations, although useful for different purposes, have the same meaning and are interchangeable.

Before attempting to construct the hidden part of an object, it is appropriate to attempt to deduce as much as possible from what is visible in the drawing. Several complementary steps (described in more detail in [34]) exist for inferring information from a drawing. These include:

- *Region identification,* the subdivision of the drawing into 2D areas bounded by loops of lines. This is straightforward to carry out. Each region corresponds to an *entire* face or a *partial* face of the portrayed object.
- *Line labelling:* each line may be labelled as representing a *convex* edge (+), a *concave* edge (−) or an *occluding* edge (→); in the latter case the direction of the arrow indicates which face is attached to the edge, and which is merely occluded by it. The process of determining which label each line has is called *line labelling* and is discussed in Section 2 below.
- *Grouping of parallel lines,* the aim being to decide which lines in the drawing correspond to parallel edges in 3D (not all lines which appear parallel in 2D are necessarily meant to be parallel in 3D!). This is discussed in Section 10.
- *Feature recognition:* a *feature*, or *form feature* ([7, 8]), is a commonly-occurring localised configuration of lines with a standard interpretation. Although useful in practice, recognition of *known* features should not be an essential component of an approach for interpreting drawings of *novel* objects, and we do not discuss it further in this paper.
- *Inflation,* which adds depth to the part of the object visible in the drawing, moving it into 3D space to create a frontal geometry; in polyhedral objects, it is equivalent to adding depth coordinates for each vertex. Inflation is the main subject of this paper.

Although line labelling is a standard and useful technique in many cases, there is no fully-reliable general-case line-labelling algorithm. Four of our twenty drawings representative of engineering objects cannot reliably be labelled using existing approaches [38]. Neither can (for example) Figures 46–52 [37]. Catalogue-based labelling, used in current line-labelling algorithms, cannot be used at all for Figures 54–61.

Since there are drawings we cannot label at present, we have two possible ways forward: we can look for improved line-labelling algorithms, and we can investigate whether line labelling is, in fact, necessary. This paper describes the latter investigation.

1.6 Inflation

This paper presents a new approach to inflation (see Figure 62). We initially recognise and evaluate those components of a drawing which first attract the attention of humans: faces, cubic corners [22], and parallel lines. Information derived from them is encoded using *compliance functions*, which produce equations involving depth coordinates of vertices—see Section 3). We then propagate the implications of these components through the drawing. Although our new approach is

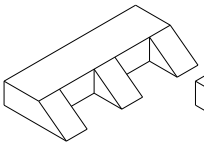

Figure 21.

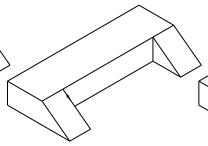

Figure 22.

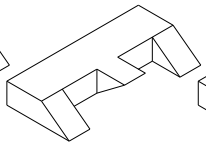

Figure 23.

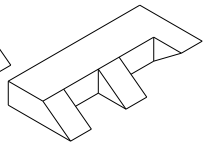

Figure 24.

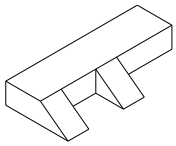

Figure 25.

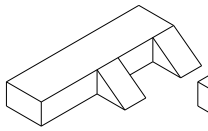

Figure 26.

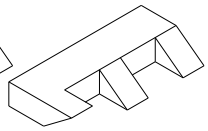

Figure 27.

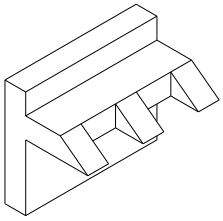

Figure 28.

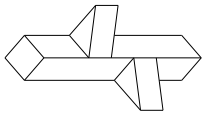

Figure 29.

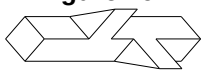

Figure 30.

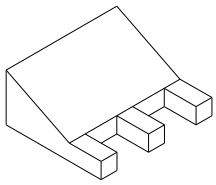

Figure 31.

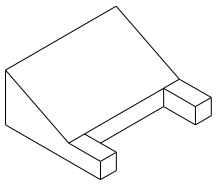

Figure 32.

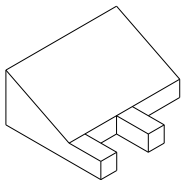

Figure 33.

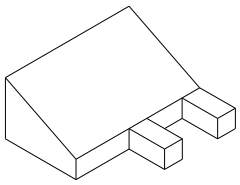

Figure 34.

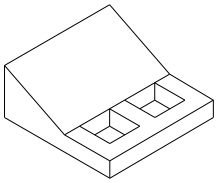

Figure 35.

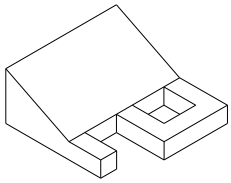

Figure 36.

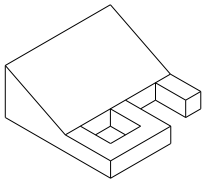

Figure 37.

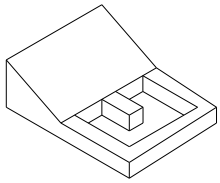

Figure 38.

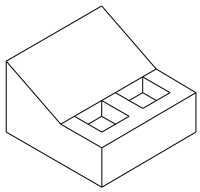

Figure 39.

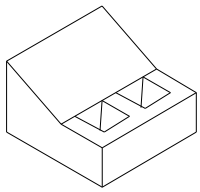

Figure 40.

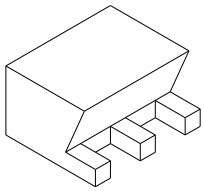

Figure 41.

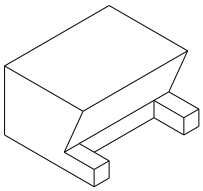

Figure 42.

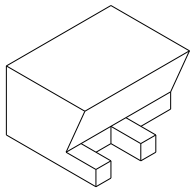

Figure 43.

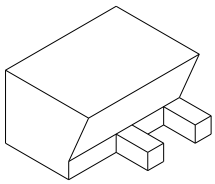

Figure 44.

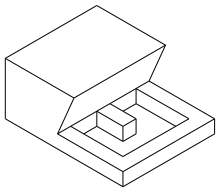

Figure 45.

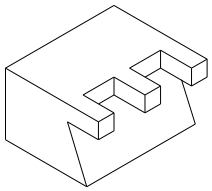

Figure 46.

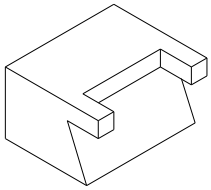

Figure 47.

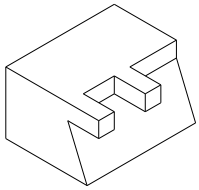

Figure 48.

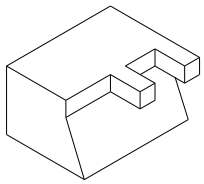

Figure 49.

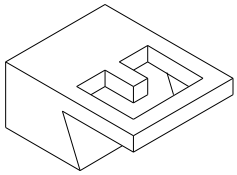

Figure 50.

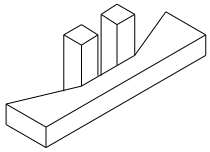

Figure 51.

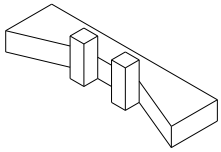

Figure 52.

Line Drawings with K-Vertices [37]

less mathematically-rigorous than those presented elsewhere (e.g. [5, 34, 38]), it may be closer to the way people see things (we suggest that the process used by human vision may be modelled better by progressively-refined belief values than by solution of mathematical equations). Emulating the human vision process is likely to lead to better results if we define the *correct* interpretation of a drawing as the object a human sees in it.

In outline, we use a relaxation approach [24] which:

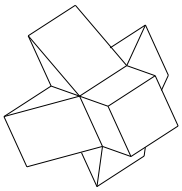

Figure 54.

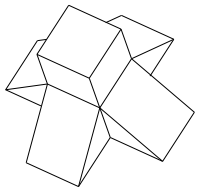

Figure 55.

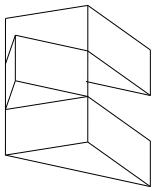

Figure 56.

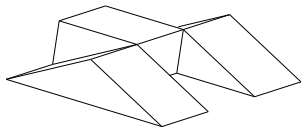

Figure 57.

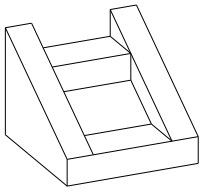

Figure 58.

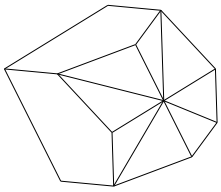

Figure 59.

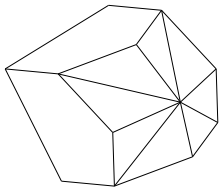

Figure 60.

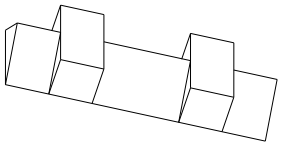

Figure 61.

Line Drawings with Higher-order Vertices

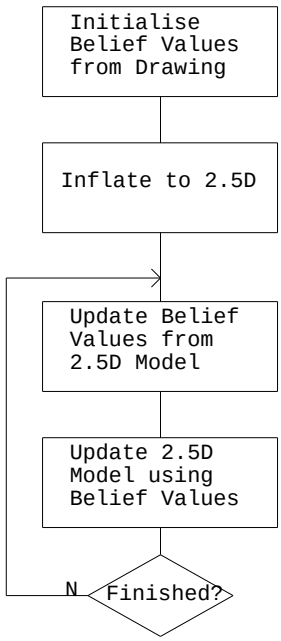

Figure 62. Overview of Approach

- For our chosen *compliance functions*, determines initial *belief values* in the range 0–1. These measure our confidence in the belief that a given encoded idea is correct: a belief value of 1 indicates that something is known to be true, a belief value of 0 indicates that something is known to be untrue, and intermediate values correspond to different degrees of confidence.
- Creates an initial frontal geometry—for each vertex, somewhat arbitrary z-coordinates are created (see Section 4) so that subsequent use of non-inflationary compliance functions will not lead to trivial (non-inflationary) results.
- Iteratively,
  - Updates the belief value of each compliance function according to how well it matches the current state of the frontal geometry. Where the match is good, the belief value is raised; where the match is poor, the belief value is lowered.
  - Updates the frontal geometry using predictions from the compliance functions—for each vertex, a separate z-coordinate prediction is made using each appropriate compliance function, and these predictions are combined by averaging, with each prediction being weighted according to its current belief value.

There are successful precedents for hierarchical relaxation being a useful approach for emulation of human vision processes [13] and for solving geometric constraint satisfaction problems [5].

1.7 Overview

In the rest of the paper, Section 2 discusses the benefits of line labelling, showing why labelling has been popular in the past. Section 3 discusses traditional approaches to inflation. Section 4 describes how we initialise a set of vertex z-coordinates. Sections 5–12 describe our chosen compliance functions; each Section describes how the initial belief value is calculated,

how the belief value is updated, and how the compliance function is used to predict z-coordinates. The parallelogram compliance function in Section 9 is new: although the concept of "deskewing" parallelograms has been used before [11], our use of this concept requires fewer other assumptions; Section 11, enforcing 3D collinearity of edges which derive from the same 2D drawing line, is also new. Section 13 presents the tests carried out on our test set and our results, while Section 14 draws conclusions from these results and makes recommendations for further investigation into the approach we present.

The main goal of this paper is to obtain by other means the benefits which line labelling would provide.

2 Starting Point: Why Label?

2.1 Background

Line labelling is the process of determining, for each line in a drawing, whether the line should be labelled as *convex* (+), *concave* (−) or *occluding* (→). By convention, occluding lines are labelled with the occluded face on the left side of the arrow. Labelling is a well-established preliminary stage of interpreting line drawings [1, 6, 9]. The best-established method of labelling line drawings is by means of a *junction catalogue* [1, 9], a list of which labels may validly meet at a junction given a set of assumptions about the type of object and the nature of the drawing. Combinations of labels for lines meeting at a junction which do not correspond to a valid junction label can be rejected. The task is thus translated into a discrete constraint satisfaction problem, where the constraints are that each line must have the same label throughout its length, and each junction must have a valid labelling from the catalogue.

The Clowes-Huffman [1, 9] catalogue for line drawings of *trihedral* polyhedra is well-established. Although the limitation to trihedral vertices is somewhat restrictive, Clowes-Huffman line-labelling has been used successfully in applications similar to our own [5]. The assumption that the drawing shows a *single* object simplifies labelling, reducing the potential for ambiguous labellings.

Despite the initial success of line labelling methods, the limitation to polyhedra with only trihedral junctions has proved a problem. Real engineering objects are often not trihedral (e.g. eight of our twenty test drawings are not). Various extensions to the junction catalogue have been proposed, including one for simple curved objects [18], one with the six extra junction labels required to label polyhedra with extended trihedral vertices (appearing in [21] and others), and most recently a full tetrahedral polyhedral junction catalogue [32].

Extension of junction catalogues results in a greater proportion of junction labellings being valid rather than invalid, and an inevitable consequence of this is that more labellings of entire objects are valid. This number can quickly become so large as to be result in previous methods being infeasible—for example, Figure 6 has 337 valid labellings if the junction catalogue for tetrahedral K-vertices is used, and nearly 1.4 million if the junction catalogue for all tetrahedral vertices is used. This should be compared to the case for trihedral objects, where often a unique labelling results.

Many such "valid" labellings are not geometrically realisable. We have noted before [34, 37] that although the problems of ambiguity and geometric unrealisability exist even with trihedral drawings, they are far worse for non-trihedral drawings.

2.2 Why Labelling is Useful

Certainly, labelling is desirable. Successful labelling provides useful information about the object drawn. Firstly, line labels indicate which edges bound the visible faces or partial faces of the object and which merely occlude them. For example, labelling Figure 1 correctly shows that three of the faces are partially-occluded, and the visible parts will need extending when creating a complete topology.

Secondly, junction labels can be used to obtain a depth ordering of visible vertices [33]. It is, for example, immediately apparent that, in the most natural interpretation of Figure 1, two of the Y-junctions are all-convex, with the Y-junctions being nearer the viewer than any of their edge-connected neighbours, while the third is all-concave and further from the viewer than its neighbours.

Thirdly, the underlying vertex types implied by junction labels constrain the possibilities when attempting to construct the hidden topology of the object. As an example, when interpreting Figure 6 using our tetrahedral junction catalogue [32], the T-junction corresponding to S in Figure 65 corresponds to a tetrahedral vertex where (a) exactly one more edge is required to complete the vertex and (b) that edge must be concave.

Fourthly, the underlying vertex types implied by junction labels are an aid to symmetry detection. In Figure 6, the underlying vertex type of the T-junction mentioned above is the same as that of the neighbouring K-junction, supplying supportive evidence for the hypothesis that there is a plane of mirror symmetry normal to the edge joining these two vertices.

A further reason for the frequent use of labelling is the speed of Clowes-Huffman labelling of trihedral objects: in practice (but not in theory) it takes $O(n)$ time (n being the number of lines) with a small constant. However, as computers are now much faster than when line labelling was invented, this is becoming less significant. For example, our alternative labelling approach based on probabilistic relaxation [36] also takes much less than a second to interpret drawings of a realistic complexity when run on a modern computer.

2.3 Limitations of Labelling

As yet, we have no perfect solution to the problem of only producing labellings which are geometrically realisable. In previous papers [37, 38] we outlined and extended a method which produces both a provisional frontal geometry and recommendations for line labels. Although frequently successful, it still fails often enough (as do earlier methods which carry out line labelling first) to make it worthwhile considering whether, and why, labelling is necessary at all.

Another open problem in line-labelling is that of distinguishing through holes from pockets. In principle, this is simple: if the depth of the feature is approximately the same as the thickness of the material, the feature is a through hole and visible lines at the bottom are occluding; otherwise, the feature is a pocket and the lines are concave.

A straightforward implementation of this idea would identify correctly that the features in Figures 3 and 5 are pockets. However, two problems are immediately apparent. Firstly, the "thickness of the material" is not always easy to determine. If Figure 5 stood on legs, rather than resting on a flat base, determining the thickness near the hole loop would not be trivial. Secondly, there is the question of when holes and pockets may be present, and the above problem must be solved. This is clearly the case whenever the presence of a hole loop corresponding to a hole or pocket is hypothesised, but not all through holes and pockets possess hole loops. For example, the depression in Figure 2 could, topologically, be either a hole or a pocket, but the drawing contains no hole loop.

Problems such as these take line-labelling far away from the domain of discrete constraint satisfaction problems. Additionally, it is psychologically implausible that line labelling, an NP-complete constraint satisfaction problem, can form an early part of the human vision process [4]. For these reasons, although line-labelling is without doubt useful, we must consider whether it is essential, and whether any truly essential information it produces can be produced by other means.

2.4 Existing Alternatives to Labelling

One alternative approach which has been proposed is Draper's *sidedness reasoning* [3]. This can be illustrated by reference to Figure 63, which has five faces, five regions, and six internal lines (AB, BC, CD, DE, EA, BE). To simplify Draper's idea for illustrative purposes, we assume that these six lines are either convex or concave, and the boundary lines are occluding.

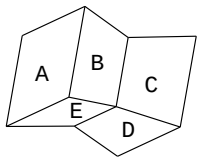

Figure 63. Sidedness Reasoning

Clearly, if line AB is convex, then in region A, the plane of face A is behind the plane of face B; if line AB is concave, the plane of face A is in front of the plane of face B. The converse is true for region B. Similar deductions can be made for the other six internal lines. Furthermore, if line AB is convex, the presence of the Y-junction separating A, B and E implies that in region E, the planes of faces A and B are in front of the plane of face E; if line AB is concave, the planes of faces A and B are behind the plane of face E. The consequences of X-junctions such as the one separating regions B, C, D and E are less easily determined, as are those of apparently-trihedral junctions which in fact correspond to non-trihedral vertices. Labellings which produce direct or cyclic contradictions can be eliminated. For example, it is possible to determine, even without examining lines BC, CD and DE, that lines AB, BE and AE must either be all-convex or all-concave.

Apart from the difficulties of extending Draper's idea to non-trihedral vertices, the major problem remains that, regardless of implementation details which may speed it up in practice, its worst-case performance will never be better than exponential, as each possible cycle of face planes must be checked in each region for all combinations of convex and concave lines.

Thus, in practice, Draper's idea suffers from exactly the same problems as more traditional line-labelling approaches: it does not extended reliably to non-trihedral vertices, and its theoretical execution time (and practical time when extended to non-trihedral vertices) is exponential.

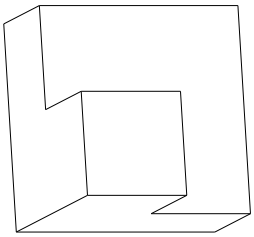

Figure 64. Occlusion without T-Junctions

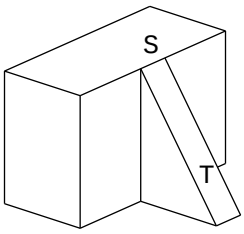

Figure 65. T-Junctions

2.5 Occlusion and T-Junctions

However, while rejecting Draper's method, we can make use of some of its underlying ideas. If non-silhouette lines are limited to corresponding to either convex or concave edges, it is clearly possible to simplify Draper's concept into a polynomial-order approach: create, and iteratively update, the frontal geometry until various constraints are satisfied, and then use this geometry to determine whether the edges are convex or concave. It is the possibility that non-silhouette lines may be occluding which causes difficulties.

Object self-occlusion is a little-studied area. Previous investigations of occlusion, e.g. Cooper [2], have concentrated on multi-object pictures, where one object is occluded by another, and not on the problem of possible self-occlusion; others who touch on occlusion (e.g. [4, 5, 21]), through their reliance on the trihedral junction catalogue of Clowes and Huffman, do not even consider the possibility that T-junctions might not be occluding.

In general, the presence of T-junctions in a drawing suggests occlusion. However, it is neither a necessary nor a sufficient condition. Drawings exist where there are no T-junctions, but some lines are clearly occluding—Figure 64 provides an example—but they are comparatively rare. More common are drawings where T-junctions do not correspond to the occlusion of one line by another. See Figure 65 (part of Figure 6). Junction S is a non-occluding T-junction, a reflection of its non-trihedral neighbour. Junction T is, however, an occluding T-junction, being the point in the drawing where one edge passes behind another.

We believe that the only *essential* function performed by line labelling is to distinguish occluding T-junctions from non-occluding T-junctions. Although labelling's other output is certainly useful, it is possible to obtain the equivalent information by other means at a later stage of processing. For example, although it is helpful to know whether two-face edges are convex or concave, this can be determined geometrically once the drawing has been inflated.

There are several reasons why it is important to make the distinction between occluding and non-occluding T-junctions:

- There is *no vertex* at the x-y location of an occluding T-junction; there *is* a non-trihedral vertex at the x-y location of a non-occluding T-junction. (No other junction type can potentially not correspond to a vertex of the object portrayed.)

- Thus, in detecting symmetry (an important and useful technique—see e.g. [23]) by a process of matching vertices, a non-occluding T-junction must match a non-trihedral vertex of similar underlying type, whereas an occluding T-junction need not match anything.

- Similarly, in constructing the hidden topology, the presence of a non-occluding T-junction gives considerable information concerning the edge(s) which must be added to complete the topology, whereas an occluding T-junction provides far fewer clues.

- Occluded and occluding lines have different z-coordinates at an occluding T-junction, but intercepting lines have the same z-coordinates at a non-occluding T-junction.

Even if labelling is not performed, distinguishing occluding from non-occluding T-junctions is necessary to create a sensible frontal geometry, both for the theoretical reasons just outlined and in practice, as shown by our results in Section 13. In this paper, we investigate an alternative approach, using a probability-based mechanism to classify occluding and non-occluding T-junctions, *without* using line labelling.

However, it is not reasonable to expect any method for analysing T-junctions to be perfect. Even for humans, the implications of T-junctions are harder to establish than those of W-junctions or Y-junctions [4], and humans, as already noted, are "expert" at interpreting line drawings. We can even note one case, that of Figure 58, where a *correct* labelling would be misleading! A junction label which usually corresponds to an occluding T-junction here corresponds to a true vertex where five edges (two of them occluded) meet.

3 Inflation

3.1 Inflation and Compliance Functions

Inflation is the process of adding depth to the part of the object visible in the drawing. This is normally done by solving a system of equations. Any information which may be available or hypothesised is encoded using *compliance functions*, ideas which can be encoded as equations relating depth coordinates. For example, the idea that two edges are parallel can clearly be translated into an equation relating the depth coordinates of the vertices joined by these two lines (see Section 10).

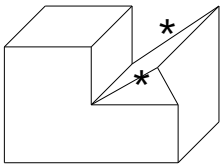

Figure 66. Edges Not Parallel in 3D

Since compliance functions represent hypotheses, and hypotheses are sometimes wrong, the resulting equations can sometimes be wrong too. Figure 66 is one such example: despite appearances, the indicated lines do not correspond to parallel edges. Inflation methods *must* be able to deal with contradictory equations, and should ideally be able to identify which assumptions are erroneous.

Traditional methods of inflation (e.g. [5, 16, 34]) use a variety of compliance functions to create a system of equations relating the depth (z-)coordinates for each visible vertex, and for each point at which a partially-occluded edge disappears from view. The best solution to this (usually overdetermined) system of equations thus creates the frontal geometry.

Table 1 lists various compliance functions; it is based on the list in Lipson and Shpitalni [17], with additions. It indicates what input information the compliance function needs, whether or not the compliance function leads to an equation linear in z-coordinates, and whether or not the compliance function is, of itself, inflationary (some are not: e.g. line parallelism obviously has the trivial solution $z = 0$ for all vertices). Although "face planarity" [17] cannot be translated directly into an equation linear in z-coordinates, it can be used if expressed as several "4-Vertex planarity" compliance functions; skewed facial compliance functions can lead to linear equations provided that components of face equations are present as additional unknowns [5].

We describe our preferred compliance functions in more detail below. See [17] for further description of the others.

3.2 Chosen Compliance Functions

It is neither practical nor, given the requirements for an interactive system, desirable to use *all* of these compliance functions. However, since compliance functions are clues which can at times mislead, it is desirable to use *several* of these compliance functions, in order that any misleading clues are overruled. Accordingly, we select those which we believe to be best in terms of general applicability and reliability, so our investigation utilises the following compliance functions:

- *Face-vertex coplanarity* corresponds to the requirement that the object portrayed is polyhedral—see Section 6. Since our aim is to distinguish occluding from non-occluding T-junctions, determination of which vertices actually lie in the plane of each face is necessary at some point.

- *Corner orthogonality* [22] is based on the idea that any junction of three lines is likely to represent a *cubic corner*, a vertex at which three mutually-orthogonal faces meet—see Section 7. At least one inflationary compliance function is required, and this one has been tested and found reliable. Minimum standard deviation of angles [19], although in principle more general, works well in practice only for special cases such as regular polyhedra [14]. Corner orthogonality works well for any drawing which meets the necessary requirements, which are that it must contain junctions of three lines meeting in a mathematically-correct 2D projection of a cubic corner [22]. Although limited in principle, corner orthogonality is particularly useful in practice as cubic corners are common in engineering objects.

- *Major axis alignment* [37] is based on the idea that the three most common line orientations in the drawing correspond to the three major mutually-perpendicular axes of the object (as is commonly the case with edges of engineering objects [12, 25])—see Section 8. This is another inflationary compliance function. It is a more recent invention than corner orthogonality, and for this reason possibly less reliable. However, unlike corner orthogonality (which can fail entirely in some circumstances), major axis alignment does always inflate a drawing, if not always entirely correctly.

Name	Ref.	Requires	Linear?	Inflates?
face planarity	[17]	Labelling	Y/N	N
4-vertex planarity	[33]	Labelling	Y	N
line parallelism	[17]	Parallel	Y	N
line verticality	[17]	(none)	Y	N
isometry	[17]	(none)	Y	N
corner orthogonality	[22]	(none)	Y	Y
junction label pairs	[33]	Labelling	Y	Y
major axis alignment	[37]	Labelling	Y	Y
skewed facial orthogonality	[11]	(none)	Y/N	Y
skewed facial symmetry	[11]	Symmetry	Y/N	Y
line orthogonality	[17]	(none)	N	Y
minimum std dev. of angles	[19]	(none)	N	Y
face perpendicularity	[17]	(none)	N	Y
line collinearity	[17]	(none)	Y	N
planarity of skewed chains	[17]	Symmetry	N	Y
mirror symmetry	[40]	Symmetry	N	Y

Table 1. Compliance Functions for Inflation

- *Parallelograms*, which we introduce in Section 9, corresponds to the requirement that a parallelogram in the drawing corresponds to a rectangle in 3D. As we explain in Section 9, this can provide useful information beyond that provided by line parallelism and can be used in many instances where major axis alignment is inappropriate. Although the concept of "deskewing" a parallelogram to obtain a rectangle is the same as that in Kanade's "skewed symmetry" [11], our approach makes different (and, in our opinion, more justifiable) assumptions than Kanade's.

- *Line parallelism* [17] assumes that lines parallel in 2D correspond to edges parallel in 3D—see Section 10. This is useful for producing "tidy" output (for example by making lines terminating in occluding T-junctions parallel in 3D to other lines with similar 2D orientation). However, the main reason for its inclusion here is that it also produces belief values for two lines being parallel as a secondary output. Computing such values, as discussed below, is non-trivial, and an approach which produces such information for free as a by-product of its main function is useful.

- *Through lines* corresponds to the requirement that a continuous line intercepted by a T-junction or K-junction corresponds to a single continuous edge of the object—see Section 11. Although a special case of line collinearity, there are mathematical reasons why straightness of through lines can be enforced more strictly than line collinearity.

4 Initialisation

4.1 Overview

As summarised in the introduction, the first stage of our approach to finding frontal geometry uses 2D information for two purposes: to estimate initial belief values for the assertion posited by each compliance function in appropriate places in the drawing, and to estimate initial z-coordinates for each junction. Later Sections describe, individually, how the belief values are initialised; these belief values are, in turn, used to update the z-coordinate estimates. Here, we consider how the frontal geometry is initialised. Because of the two-way nature of the relaxation process, it is desirable to initialise the frontal geometry plausibly, in order that the respective belief values of competing compliance functions are evaluated reliably, and to speed up the process of convergence to a solution.

4.2 Simple Initialisation of Frontal Geometry

In our initial investigations, we assumed that since the frontal geometry would be produced by our compliance functions, all that would be required of initialisation would be to force the z-coordinates away from the trivial solution ($z = 0$ for all vertices) which satisfies parallel line and face-vertex coplanarity compliance. To this end, we found the xy-centre (the midpoint of the extreme junction x- and y-coordinates), assumed that this was the nearest point to the viewer, and initialised all z-coordinates to lie on the surface of a cone having the xy-centre as the apex and with axis in the z-direction (the cone angle was a tuning parameter).

It was found that this was sufficient in the majority of cases, but that in a significant minority of cases a phenomenon occurred which we term *z-reversal*: the z-coordinates have the wrong sense, with vertices which should be convex being concave and vice versa. Although, in practice, it is often easy to detect and correct z-reversal [37, 38], there are cases where correcting z-reversal as a postprocessing step is insufficient: z-reversal can provide misleading information to the compliance functions, the wrong compliance functions can as a consequence be selected, and relaxation can then converge to the wrong minimum. Also, z-reversal is a symptom of deeper problems and requires an explanation.

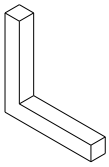

Figure 67. No Central Vertex

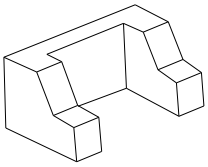

Figure 68. Distant Central Vertex (from [5])

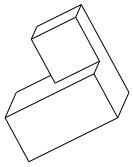

Figure 69. Distant Central Vertex

Figures 67–69 show three typical drawings leading to z-reversal; there are two distinct problems, depending on whether z-reversal occurs as a consequence of initialisation or arises later.

In Figure 67, initialisation to the surface of a cone is inappropriate. All of the junctions in the drawing lie on the same side of the cone, so although the "outer" faces are initialised acceptably, the "inner" faces are not. Whether z-reversal occurs or not is then a matter of chance, and in this case chance is against us. In Figure 68, the problem is similar but worse: initialisation to the surface of a cone is harmful, as the centre of the drawing corresponds to the part of the object which is furthest from the viewer, not nearest.

4.3 Avoiding Initialisation Problems

To circumvent such problems of initialisation of frontal geometry, we currently include in our initialisation process the predictions for relative z-coordinates produced by the cubic corner compliance function (see Section 7), assuming that potential silhouette cubic corners are of the only permitted type and that other potential cubic corners are of the most common type. These are combined with the cone assumption by translating both assumptions into an overconstrained set of linear equations (in which the weighting of the cone assumption is very low) and finding the best solution to the resulting system of equations.

The problem with Figure 69 occurs at a later stage and is explained in Section 8.4.

5 T-junctions

5.1 Initialisation

In Section 2, we noted the importance of distinguishing between occluding and non-occluding T-junctions.

Initially, we do not know whether a given T-junction is occluding or not, so we set the belief value for each T-junction being occluding to the same fixed intermediate value. (This value and other "fixed values" are parameters which can be tuned for optimal performance. Their tuning is discussed in Section 13; later in this and subsequent Sections we give the values used).

An occluding T-junction has two z-coordinates, one where the occluded edge passes out of view and the other on the occluding edge at the same x-y coordinates. Unless and until we determine that a T-junction is non-occluding, we maintain separate z-coordinates for these two. If, when the final frontal geometry is obtained, the two z-coordinates are (approximately) the same, the T-junction is (probably) non-occluding.

5.2 Updating

On each iteration, after the new z-coordinates have been calculated, we update each belief value as shown:

$$b' = kb + (1 - k)\left(d^{-\delta}\right)$$

where k is an under-relaxation (or damping) factor, d is a tuning parameter and δ is the difference (measured in pixels) between the two z-coordinates for the T-junction. If the updated value is outside the "intermediate" range (0.0001–0.9999), we set it to 0 or 1 as appropriate and record that the T-junction is definitely either *non-occluding* or *occluding*.

This belief value is an output, not fed back into the algorithm. However, if at any point it is decided that a T-junction is known to be non-occluding, our algorithm no longer maintains two separate z-coordinates for it.

We use under-relaxation here and elsewhere for a number of reasons, including:

- To model our belief that human evaluation of hypotheses in line drawings is a gradual process, which involves considering the consequences of accepting a hypothesis before doing so.

- To allow compliance functions to vary in importance during the iteration process: some may be more important early on, while others may be more important in the later stages.
- To reduce the likelihood of a plausible but incorrect hypothesis being accepted at an early stage.
- To reduce the likelihood of non-convergence (damping is a standard technique from numerical analysis, most often used to avoid oscillation).

5.3 Tuning

The values of tuning parameters, as used to produce our test results, are: the initial belief value for a T-junction being occluding is 0.26 if K-junctions are visibly present in the drawing, 0.57 otherwise. The under-relaxation factor k is initially 0.97, dropping linearly to 0.95 for the final iteration (the number of iterations used is explained in Section 13). The initial belief values seem rather low; this, together with the high under-relaxation factor, suggests that our approach is perhaps too eager to label T-junctions as occluding on the basis of small, perhaps temporary, differences in z-coordinates.

6 Face-Vertex Coplanarity

6.1 Overview

Although our main objective in this paper is to use inflation to distinguish occluding from non-occluding T-junctions, distinguishing convex from concave edges is also useful. The simplest way of doing this is to compare the computed face normals of the two faces meeting at each edge. Hence, it is useful as well as aesthetically desirable to try to ensure that the loop of vertices for each face is coplanar.

As with the other compliance functions we use, the purpose here is twofold: to determine whether or not the assertion represented by the compliance function is satisfied, and if it is, to constrain the corresponding vertex z-coordinates so that compliance is enforced.

The problem here is knowing which vertices lie in the plane of a face—junctions which do not bound the corresponding region may nevertheless correspond to vertices which lie in the plane of the face, and junctions which *do* bound the region need not necessarily lie in the plane of the face. As an example of the former, two regions may correspond to parts of the same face: Figures 4 and 7 are two drawings in which this is the case. As an example of the latter, the face may be partially-occluded, in which case some junctions bounding the region may correspond to vertices which occlude, rather than form part of, the corresponding face: Figures 1 and 2 are two drawings in which this is the case.

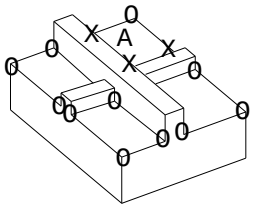

Figure 70. Coplanar Vertices

In extreme cases, such as Face A in Figure 70, it can even be the case that vertices *elsewhere* in the object are more likely to lie in the plane of the face than vertices corresponding to junctions which bound the 2D region.

For this reason, we maintain a matrix of belief values for the possibility of *each* vertex lying in the plane of *each* face.

6.2 Initialisation

We determine the equation of the plane of each face using a weighted linear system where the variables are those (A, B and D) of the face equation $Ax + By + z + D = 0$ (note that, given a face equation of the form $ACx + BCy + Cz + CD = 0$, the general position assumption requires that $C \neq 0$, so C can be removed [5]). There is one equation for each vertex which might lie on the face, and the weight is the belief value that it does lie on the face.

Given this, it is straightforward to make a prediction of the z-coordinate of any vertex which might lie on a face: $z = -(Ax + By + D)$.

Initially:

- Any vertex lying on a fully-visible face is in the plane of the face; the belief value is 1. (We currently assume that if the loop of junctions bounding a region contains one or more T-junctions, the face is not fully-visible; if there are no T-junctions, the face is fully-visible—as shown in Figure 64, there are rare drawings for which this assumption is incorrect.)
- Any vertex at the other end of an edge leaving a fully-visible face is *not* in the plane of the face; the belief value is 0.
- Any vertex lying on a partially-visible face may well be in the plane of the face, but this is not

certain; the belief value, a tuning parameter, is an arbitrary, reasonably high, fixed value.

- Any vertex not falling into any of the above categories might be in the plane of the face, although this is fairly unlikely; the belief value, a tuning parameter, is an arbitrary, reasonably low, fixed value.

Note that for drawings including hole loops, this is not enough in practice to ensure that vertices on the inner loop are coplanar with the outer loop. To provide additional encouragement for this, we reinforce the value calculated by the belief value of any configuration of lines which strongly suggests a hole loop (such configurations are described in [36]).

6.3 Updating

On each iteration, after the new z-coordinates have been calculated and the new 2D axis orientations re-estimated, we update any belief value as shown:

$$b' = kb + (1 - k)\left(d^{-\delta}\right),$$

where δ is the perpendicular distance between the vertex and the face plane (recalculated using the new z-coordinates). If the updated value is outside the "intermediate" range (0.0001–0.9999), we set it to 0 or 1 as appropriate and note that the vertex either *cannot* or *must* be in the plane of the face.

Note that this compliance function is in some sense anti-inflationary, in that for as long as there is the possibility that all vertices lie in the same plane, there will be predictions made that to this effect.

After the belief values have been updated, we re-determine the equation of the plane of each face as described above, using the belief values as the weights in a weighted linear system, and from this make a new estimate of the vertex z-coordinate of each vertex lying in this plane.

6.4 Tuning

The initial belief value for a vertex lying in the plane of a face, optimised and used as described in Section 13, is: 0.52 if the corresponding junction bounds the region and the region includes no T-junctions; 0.04 if the corresponding junction bounds the region and the region includes one or more T-junctions; and 0.49 if the corresponding junction does not bound the region. It seems unlikely that the presence of T-junctions in a region makes the junctions bounding that region less likely to lie in the plane of the face than arbitrary junctions elsewhere in the drawing, although the case of Figure 70 shows that, for some drawings at least, it is conceivable. The under-relaxation factor k is initially 0.03, rising linearly to 0.74 at the final iteration, suggesting that it is important to update these belief values from geometric inflation early on (given our doubts about the initial belief values, this is hardly surprising!). The weighting factor for calculating geometry from these belief values is initially 0.10, dropping to 0.02 later on, implying that this compliance function is of little help in calculating geometry, particularly in the later stages.

7 Cubic Corners

7.1 Overview

A *cubic corner* [22] is a junction where three mutually orthogonal planes meet. There is strong psychological evidence that humans tend to interpret W-junctions and Y-junctions as cubic corners wherever possible. For example, Sugihara's realisations of "impossible objects" [30] are effective because when humans look at a drawing containing three groups of parallel lines we perceive its junctions as representing cubic corners, even when such drawings can only be realised geometrically using non-orthogonal planes. Enns and Rensink [4] have also found that humans find it easier to interpret entire drawings where the trihedral junctions *do* correspond to such cubic corners. Thus, as argued before, if a human will generally interpret a drawing in this way, it was probably this interpretation which the originator intended, and thus the interpretation a computer should aim to produce.

This compliance function is particularly useful in constructing a frontal geometry, as it is possible to determine the 3D direction and relative length of an edge at a cubic corner from the angles of the lines in the 2D drawing [4, 22]. In Figure 71, the trihedral junctions V might be cubic corners, enabling us to predict the depth differences $z_A - z_V$, $z_B - z_V$ and $z_C - z_V$ from the angles E, F and G and the lengths of the lines VA, VB and VC. It can be shown that if the corner is indeed a cubic corner,

$$|z_A - z_V| = l_{VA}/\sqrt{(\tan F \tan G) - 1}$$

where l_{VA} is the 2D length of line VA, and F and G are the 2D angles AVC and AVB [22, 34]. The weight applied to the predictions of z_A, z_B and z_C is a value measuring the confidence in the belief that the junction is indeed a cubic corner.

Only W-junctions and Y-junctions can be cubic corners. A W-junction can only be a cubic corner if the enclosed angles are individually acute, and their sum is obtuse [22].

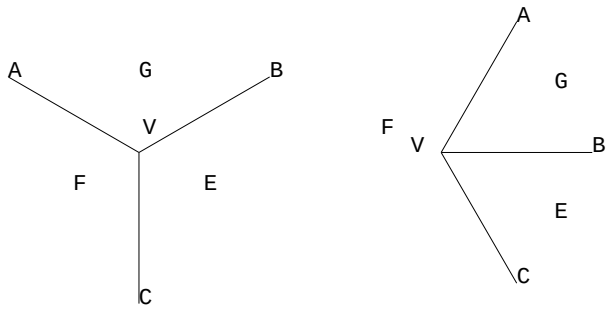

Figure 71. Cubic Corners

Kanade's "skewed symmetry" [11] is an alternative approach to determining relative depth, using similar input information but making different assumptions. Kanade treats cases where there is good reason to believe that two 2D lines (not necessarily meeting at a junction) are perpendicular in 3D. Mirror symmetry is one of many possible clues which could suggest this belief.

Kanade's approach ignores line VC (which need not even be present), so one further assumption is required, and the assumption Kanade makes is equivalent to assuming that equal 2D line lengths correspond to equal 3D edge lengths. If this assumption is made, the corresponding formula becomes

$$|z_A - z_V| = l_{VA}\sqrt{(|\cos G|)}.$$

Clearly, except in the specific case of standard isometric projection, these two formulae produce different results. Since we have not been able to trace any suggestion in the perception literature concerning which approach produces results closer to human perception, we interpolate between the two predictions using a tuning parameter.

(We suggest that interpolating in this way between the two approaches might in any case be for the best. To illustrate this point, consider the five drawings in Figure 72. Which do you perceive as the cube? More specifically, which drawing depicts an object in which all the edges are the same length? According to Perkins's approach, it is the leftmost; according to Kanade's approach, it is the rightmost. The other three drawings are interpolated between the two extreme approaches.)

7.2 Initialisation

We initially set the belief value for a W-junction being a cubic corner to a fixed value if the enclosed angles are individually in the range 15°–75° and their sum is in the range 105°–165°; this value is reduced by a factor $\cos\alpha^p$ for each angle outside this range, where α is the amount by which the angle has strayed into the "danger area".

A Y-junction can only be a cubic corner if all three angles are obtuse [22]. We initially take the belief value for a Y-junction being a cubic corner to be an arbitrary set value (a tuning parameter) if the angles are in the range 105°–165°; this value is reduced by a factor $\cos\alpha^p$ for each angle outside this range, where α is the amount by which the angle has strayed into the "danger area".

Predicting whether A is behind or in front of V is less straightforward, and depends on the variety of the cubic corner, which we now describe.

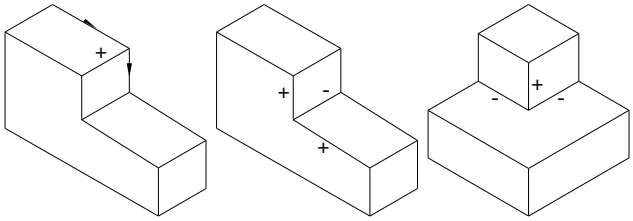

Figure 73. W Cubic Corners

There are three varieties of W-junction cubic corner, shown in Figure 73, corresponding to the three trihedral labels possible at W-junctions. W-junction cubic corners on the object silhouette can only be of the left-hand variety; those elsewhere can be any of the three, and are most commonly of the mostly-convex central variety. In the left-hand and right-hand varieties, the two outer neighbours are further from the viewer than the W-junction and the middle neighbour nearer; in the central variety, the two outer neighbours are nearer the viewer and the middle neighbour further. The three possibilities are treated separately, with separate belief values, but are mutually exclusive: the sums of their belief values cannot exceed 1, and once a junction is known to be of one of these three types, the belief value for the other two types are set to 0.

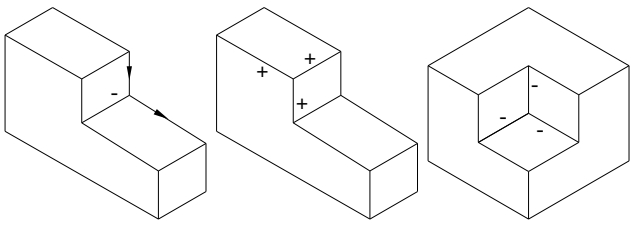

Figure 74. Y Cubic Corners

As with W-junctions, there are three varieties of Y-junction cubic corner, shown in Figure 74, correspond-

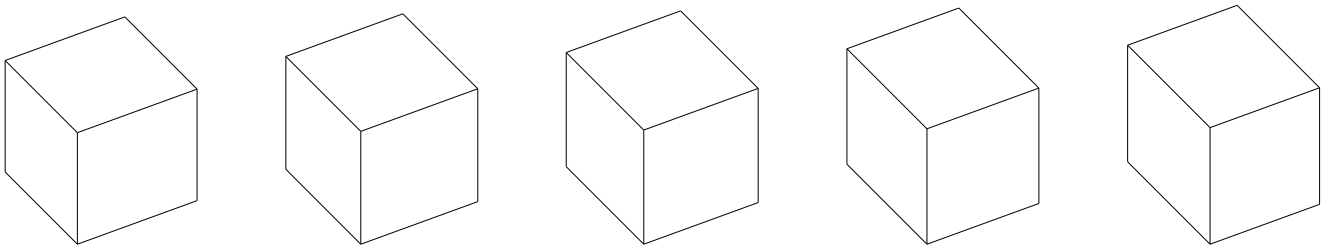

Figure 72. Which is the Cube?

ing to the three trihedral labels possible at Y-junctions. Again, Y-junction cubic corners on the object silhouette can only be of the left-hand variety; those elsewhere can be any of the three, and are most commonly of the all-convex central variety. In the left-hand and right-hand varieties, the three neighbouring vertices are nearer the viewer than the Y-junction; in the central variety, they are further away. Again, the three possibilities are treated separately, with separate mutually exclusive belief values.

Since, for both W-junctions and Y-junctions, the three varieties have labelling implications, when groups of such junctions occur as neighbours there are restrictions on which combinations are possible. Identifying and enforcing such restrictions could, and possibly should, be done. However, to do so would be a tacit return to line labelling, nullifying the question which this paper poses, and so we deliberately do not do it.

7.3 Updating

On each iteration, after the new z-coordinates have been calculated and the new 2D axis orientations re-estimated, we update any belief value as shown:

$$b' = kb + (1 - k)\,(PQR)\,,$$

where P, Q and R are belief values for the edges VA, VB and VC being perpendicular in 3D. The belief that any two vectors $\mathbf{a}$ and $\mathbf{b}$ are parallel in 3D is given by $\left(\hat{\mathbf{a}} \cdot \hat{\mathbf{b}}\right)^p$, where p is a tuneable parameter.

If the updated value is outside the "intermediate" range (0.0001–0.9999), we set it to 0 or 1 as appropriate and note that the vertex either *cannot* or *must* be a cubic corner of the appropriate variety.

We make new estimates of vertex z-coordinates using the equation given above (making a prediction of z_A from the current value of z_V and vice versa), weighting each prediction according to its belief value. There is a theoretical problem here in that, until we have resolved which type of W-junction or Y-junction is present, there will be competing predictions, some placing z_A in front of z_V and some placing z_V behind z_A. However, in practice, such theoretical ambiguity is quickly resolved by other compliance functions.

7.4 Tuning

The initial belief value for an eligible junction being a cubic corner, optimised as described in Section 13, is: 0.31 for an occluding W-junction (all-convex) cubic corner, 0.14 for a majority-convex W-junction cubic corner, 0.06 for a majority-concave W-junction cubic corner, 0.88 for an occluding Y-junction (majority-convex) cubic corner, 0.55 for an all-convex Y-junction cubic corner, and 0.30 for an all-concave Y-junction cubic corner, suggesting, ambiguously, that either Y-junctions, and in particular boundary Y-junctions, are more likely to be cubic corners, or that determining that they are is more helpful when creating frontal geometry. The under-relaxation factor k is initially 0.85, dropping linearly to 0.53 at the final iteration, suggesting, sensibly enough, that it is better to wait until the inflation results are geometrically reliable before reaching a conclusion as to whether a junction is a cubic corner. The weighting given to cubic corner belief values when recalculating geometry is initially 0.15, dropping linearly to 0.03 at the final iteration. We find these low values surprising—one possible explanation is that, since cubic corners are used in initialisation, corners which can be cubic already are, so there is no need to give a strong weighting to reinforce something which is already the case.

8 3D Axes from 2D

8.1 Overview

Another heuristic, based on observations of engineering practice, is that humans tend to interpret the three most prevalent groups of parallel lines in a drawing as representing three mutually-orthogonal axes in 3D [12]. This human bias is understandable: for example, it is clear from Samuel et al [25] that, at least

for the domain of objects which they investigated, a large majority of engineering objects have readily-identifiable major axes.

Compliance functions based on this heuristic have been used with success in constructing the frontal geometry of labelled line drawings [12] and in incorporating geometric information in the labelling process [37, 38]. Since the ideas in the latter approach make no prior assumptions about line labels, they can usefully be adapted to an attempt to construct frontal geometry without line labels. Adaptation is necessary as we do not as yet know which pairs of lines correspond to axes parallel in 3D.

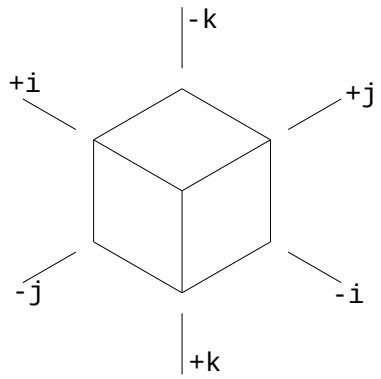

Figure 75. Three Perpendicular Axes in 2D

¿From the general position assumption, the three major axes (i, j, k) of the object cannot be aligned with the (x, y, z) view axes. Initially, we assume that the 2D projections of (i, j, k) onto the x-y plane are equally-spaced, as shown in Figure 75, with the projection of the k-axis coinciding with the y-axis (vertical axis) of the drawing.

8.2 Initialisation

We initialise a belief value for each 3D edge being parallel to each of the three major axes, as $\cos \delta^p$, where δ is the difference in angle between the corresponding 2D line and the initial 2D axis orientation, and p is the same tuneable parameter as before. Since axis alignments are mutually exclusive, if the sum of the three belief values for a given line is greater than 1, they are all reduced proportionately.

8.3 Updating

On each iteration, after the new z-coordinates have been calculated, we re-estimate the 2D axis orientations as the mean of the 2D line angles, weighted by the belief values for the corresponding 3D edges being aligned with the axes.

On each iteration, after the new z-coordinates have been calculated and the new 2D axis orientations re-estimated, we update any belief value as shown:

$$b' = kb + (1 - k)\,(\cos \delta^p)\,.$$

Again, if the sum of the three updated belief values is greater than 1, they are all reduced proportionately. If the updated value is outside the "intermediate" range (0.0001–0.9999), we set it to 0 or 1 as appropriate and note that the edge either *cannot* or *must* be aligned with the axis.

Prediction of relative z-coordinates is based on the cubic corner method described in the previous section: given three 2D lines which are believed to be mutually-orthogonal in 3D, the ratio of depth change to 2D line length of each can be calculated. This idea is applied to any line believed to be parallel to an axis (no predictions are made for lines known not to be parallel to any axis).

8.4 Issues

It can be seen from Figure 75 that the output of this compliance function is not invariant with respect to orientation of the input drawing—if the cube portrayed in the Figure were turned upside down, this compliance function would predict a concave rather than a convex frontal geometry. This appears to be the cause of z-reversal in drawings such as Figure 69. Although initialisation produces a convex object, the repeated predictions this compliance function makes of a concave object eventually take effect.

In order to test how serious this problem is, we have experimented with attempting to alleviate it by running the entire relaxation process four times, with the input drawing being rotated by 30° after each time. This produces four potential sets of z-coordinates. The final output is a collation of the input set of xy-coordinates plus the set of z-coordinates for which the various compliance functions were most in agreement. The results of this experiment are described in Section 13.

8.5 Tuning

The tuning parameters, optimised and used as described in Section 13, are: the under-relaxation factor k is initially 0.12, rising linearly to 0.99 for the final iteration, suggesting that when deciding which edges are axially-aligned it is best to make a decision fairly

early and stick to it. The weighting for using this information when re-calculating geometry is initially 0.43, rising linearly to 0.65 for the final iteration, suggesting that edge axis alignment is a useful clue to $2\frac{1}{2}$D geometry.

9 Parallelograms

9.1 Overview

We have found that there are many occasions where humans strongly believe that four junctions should form a rectangle in 3D but that traditional compliance functions do not explicitly enforce this. As examples of this, we note:

- Opposite "towers" in Figure 8 are clearly intended to be the same height. We can encourage this by making paired vertices at the tops of these "towers" form a rectangle in 3D.
- The large non-axially-aligned face in Figure 35 is clearly intended to be rectangular, and making it rectangular in 3D should be explicitly encouraged.
- The "base" on which Figure 55 rests is clearly intended to be two overlapping rectangles. For each rectangle, all four vertices are visible as junctions in the drawing, but no traditional compliance function would explicitly make them form a rectangle in 3D.

This belief is hardly surprising. As [25] point out, many engineering objects can be constructed as unions of cuboids and axially-aligned wedges, and as a result will contain rectangular groups of four vertices.

Although the concept of "deskewing" a parallelogram to obtain a rectangle is the same as that in Kanade's "skewed symmetry" [11], we noted in Section 7 that the other assumption made by Kanade, that of isometry, is not necessarily justifiable. For this reason, we use an alternative, iterative approach.

Algorithmically, we wish to identify sets of four junctions which are located approximately on the points of a parallelogram in the drawing. Our objective is to make the four vectors between the corresponding vertices form a rectangle in 3D.

9.2 Initialisation

To detect our candidate parallelograms, we start by considering all possible combinations of four junctions. Although inelegant (and $O(n^4)$), this is fast enough in practice. For each such set of four junctions:

- Where three or more junctions are collinear, the set of four junctions is discarded—for example, $ACDB$ in Figure 76 clearly cannot be a rectangle in 3D.
- Where a diagonal line exists, the set of four junctions is discarded—for example, it is clear that more harm than good will result from trying to make $ABGF$ in Figure 76 a rectangle in 3D.
- Otherwise, the initial belief value for the hypothesis that this is a rectangle in 3D is the product of the belief values for the two pairs of opposite sides being parallel in 2D.

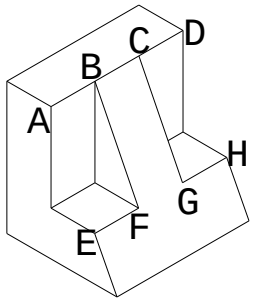

Figure 76. Skewed Parallelograms?

9.3 Updating

Consider A as one of the vertices belonging to a parallelogram with B and C as its neighbours. We wish to enforce

$$(A - B).(A - C) = 0.$$

We can expand this to:

$$(A_z)^2 + P(A - z) + Q = 0$$

where

$$P = -B_z - C_z$$

and

$$Q = (B_z C_z) + (A_x - B_x)(A_x - C_x) + (A_y - B_y)(A_y - C_y).$$

Solving for A_z gives two possible predictions; we choose the one nearer to the current value. On each iteration, we predict a new z-coordinate for each of the four corners of each candidate parallelogram using this method.

On each iteration, after the new z-coordinates have been calculated, we update any belief value as shown:

$$b' = kb + (1 - k)\left(\perp(\mathbf{tl}) \perp (\mathbf{tr}) \perp (\mathbf{bl}) \perp (\mathbf{br})\right),$$

where (for example) $\perp$ (**tl**) is the belief value for the vectors **t** and **l** being perpendicular in 3D, and **t**, **b**, **l** and **r** are the vectors corresponding to the "top" and "bottom" and "left" and "right sides" of the rectangle.

9.4 Tuning

The tuning parameters, optimised and used as described in Section 13, are: the under-relaxation factor k is initially 0.51, rising linearly to 0.95 for the final iteration, suggesting that when deciding which parallelograms should be 3D rectangles it is best to stick to any decisions made. The weighting for using this information when re-calculating geometry is initially 0.41, dropping linearly to 0.29 for the final iteration, suggesting that initially parallelograms are almost as useful as edge axis alignment as a clue to $2\frac{1}{2}$D geometry.

10 Parallel Lines

10.1 Overview

Intuitively, if two lines are parallel in the 2D drawing, the corresponding edges are perceived to be parallel in the 3D object. Although not infallible, this heuristic is generally reliable and has often been used successfully, e.g. [5, 16, 34].

Given the expectation that lines AB and CD are parallel in 3D, and values for z_B, z_C and z_D, computation of z_A is straightforward:

$$z_A = z_B \pm \frac{(z_C - zD)l_{AB}}{l_{CD}}$$

where l is the 2D length of a line.

10.2 Initialisation

The initial value of the belief value (measuring confidence in the belief that two lines are parallel) is:

- 1 if the lines *must* be parallel: i.e. they are "through" lines at a T- or K-junction
- 0 if the lines *cannot* be parallel: i.e. they meet at any other type of junction
- an arbitrary intermediate value in all other cases: we use $\frac{1-q}{2} + q\,(\cos \delta^p)$, where δ is the 2D difference in angle, and p and q are tuneable parameters, so the initial belief value that edges are parallel in 3D when they are not demonstrably *necessarily* parallel but the corresponding lines which are parallel in 2D is $1 - \frac{q}{2}$, and the initial belief value that edges are parallel in 3D when there is no topological consideration making it impossible for them to be parallel but the corresponding lines are perpendicular in 2D is $\frac{q}{2}$.

10.3 Updating

On each iteration, after the new z-coordinates have been calculated, we update any belief value as shown:

$$b' = kb + (1 - k)\,(\cos \epsilon^p)$$

where k is an under-relaxation factor and ϵ is the 3D difference in angle between the lines. If the updated value is outside the "intermediate" range (0.0001–0.9999), we set it to 0 or 1 as appropriate and note that the lines either *cannot* or *must* be parallel.

We make new estimates of vertex z-coordinates using the equation given above, predicting z_A from the current values of z_B, z_C and z_D, and similarly for the other three vertices, to obtain four new z-coordinate predictions for each candidate pair of parallel lines. Each prediction is weighted according to the belief value that this pair of lines is parallel in 3D.

10.4 Tuning

The tuning parameters, optimised and used as described in Section 13, are: q is 0.48 (undetermined initial values thus range from 0.24 to 0.76); p is initially 21.14, rising linearly to 73.65 for the final iteration, corresponding to a parallelism requirement which gradually becomes geometrically stricter. The under-relaxation factor k is initially 0.95, dropping linearly to 0.09 for the final iteration, suggesting that one should delay determining which lines are parallel until the later stages, but that towards the end one can have confidence in the geometric information. This is not necessarily what we should have expected if determination of parallel lines is an early part of the human vision process. Our results suggest, instead, that it is better to identify candidate parallel lines early, but defer judgement as to whether they truly are parallel in 3D; whether or not this matches the human vision process is a matter for speculation. The weighting factor when using parallel line information to re-calculate geometry is initially 0.95, dropping linearly to 0.75 for the last iteration, suggesting that this is an important compliance function to include, particularly at first.

11 Through Lines

11.1 Overview

It is obvious that where a line is split by a T-junction (whether occluding or non-occluding) or K-junction, all parts of the corresponding edge must remain collinear in 3D. All of the Figures 21–52 include such lines, as do several others. This does not appear in Table 1 as it has not traditionally been listed as a compliance function (the problem is absent in the trihedral world since occluding T-junctions do not correspond to vertices).

We include this as an extra compliance function which predicts the new z-coordinate of the T-junction from the old z-coordinates of the two ends of the through edge. The belief value for this compliance function is always 1 (i.e. as high as possible).

Although this could be regarded as simply a special case of collinear lines, adjusting the interpolated T-junction is always mathematically stable (whereas adjusting the extrapolated outer junctions is not), so under-relaxation should not be required for z-coordinate predictions made by this method. The weighting factor for re-calculating geometry using this information is always 1.

12 Cofacial Configurations

12.1 Overview

Cofacial configurations [34] are a method for classifying hole loops as holes/pockets or bosses by attempting to match configurations of lines in the drawing with the configurations of lines shown in Figures 77 and 78.

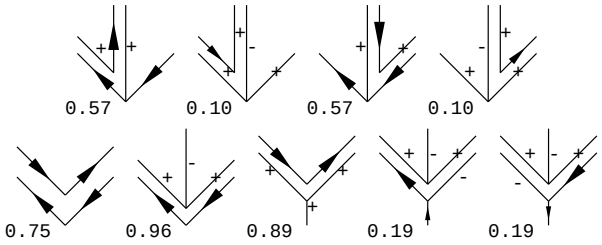

Figure 77. Templates for Holes and Pockets [34]

Cofacial configurations provide a useful method for tying disjoint subgraphs together. Drawings such as those where a hole, pocket or boss sits in the middle of a face usually contain disjoint subgraphs (using the graph-theoretical interpretation of a drawing as sets of junctions and lines). For such drawings, this compliance function is usually necessary; if it is not used, the subgraphs will often drift apart.

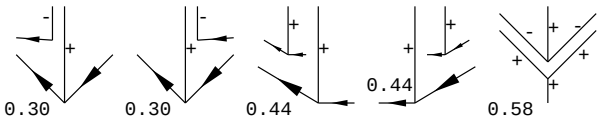

Figure 78. Templates for Bosses [34]

The initial belief values, shown in the figures, are calculated as described in [34]. They are not updated as the object is inflated. These belief values are used as the weightings when re-calculating geometry.

13 Experimental Results

13.1 Tuning

Before carrying out experiments on our test set, optimal values of the various tuning parameters mentioned above were determined by downhill minimisation of a function of the number of incorrectly-determined T-junctions. The tuning dataset used was a subset of our full test set of around 600 drawings [35], excluding drawings with no geometric realisation, drawings with no T-junctions, and (to save time) repetitive drawings with a small number of occluding T-junctions.

Some care was required in choosing the function to minimise. Since about 90% of T-junctions are occluding in our test set, and probably in drawings in general, it is possible to achieve "90% success" simply by labelling all T-junctions as occluding. As occluding T-junctions predominate to this extent, this simple but incorrect approach produces a large plateau in parameter space from which it is difficult for a downhill minimiser to escape.

Instead, the function minimised was $1 - (\frac{O_C}{O_T})(\frac{N_C}{N_T})$, where O_C is the number of occluding T-junctions labelled correctly, O_T is the total number of occluding T-junctions in the dataset, N_C is the number of non-occluding T-junctions labelled correctly, and N_T is the total number of non-occluding T-junctions in the dataset. The results achieved for the optimum tuning parameters were $\frac{O_C}{O_T}$ 88.5% correct and $(\frac{N_C}{N_T})$ 88% correct.

13.2 Experiments

The objectives of our method are twofold: firstly, to provide a set of z-coordinates for each vertex which

corresponds to a reasonable frontal geometry, and secondly and more specifically, to determine which T-junctions are occluding and which non-occluding.

In general, our experience based on preliminary investigation is that the two go together. When our approach determines correctly which T-junctions are occluding and which non-occluding, the resulting frontal geometry is usually geometrically acceptable, and when it does not, the results are unacceptable. Figures 79 and 80 are examples which illustrate this. Section 13.3 discusses this preliminary investigation further.

For this reason, our test results concentrate on the T-junction determination output. Although we tested our approach on other drawings too, for the purpose of presenting results here we concentrate on the output produced when using it to inflate our test set of twenty drawings representative of engineering objects, Figures 1–20. We note, for each drawing, the number of T-junctions identified as occluding and non-occluding, and whether this identification is correct. Section 13.4 presents the results of this investigation.

We also present similar results (albeit in less detail) for the 32 K-vertex drawings from [37], and the 8 new drawings, Figures 54–61, containing uncatalogued junctions.

The number of iterations used in our tests was fixed at 600, which gave results in a reasonable time (a fraction of a second for most drawings).

13.3 Results—Frontal Geometry

In inflating Figures 1–20, we note that the frontal geometry, although satisfactory in most cases, is not as good as can be achieved by other means [33] if a correct labelling is available. Some specific faults can be noted.

Firstly, the quality of the frontal geometry is somewhat better for simpler drawings than for more complex drawings. One possible interpretation of this is that we could obtain better results by allowing relaxation to continue for longer than we do—if this is indeed the case, our approach may become more viable as computing power increases. However, an alternative interpretation is simply that more complex drawings contain more opportunities for misinterpretation.

It is noticeable that through lines are occasionally not converted into straight through edges. Although not a particularly serious problem, the obviousness of a kink introduced into what should be a straight edge means that this should be resolved—the straightness of through edges should be enforced, not merely suggested by one of many competing compliance functions.

Despite the efforts made to avoid the z-reversal we described in Section 4, one of the twenty drawings considered in Table 2, Figure 20, still suffers from this problem. It will be noted that the most-nearly-vertical lines in Figure 20 do not correspond to vertical edges in 3D.

		Correct		Incorrect		Label?
Fig.	Output	Occ.	K	K	Occ.	
	Should be	Occ.	K	Occ.	K	
1		2	0	0	0	Y
2		3	0	1	0	*
3		2	0	2	0	*
4		2	0	4	0	Y
5		2	0	0	0	*
6		3	1	0	0	N
7		3	1	0	0	N
8		5	0	2	0	*
9		5	0	1	0	Y
11		2	1	0	0	N
12		1	0	0	0	Y
13		1	1	0	0	Y
14		1	1	0	0	N
16		1	0	1	0	Y
19		2	0	2	0	*
20		3	0	0	0	Y
Total		38/51	5/5	13/51	0/5	-

Table 2. Determination of T-Junctions I

Finally, it is clear that the geometric results of inflation depend heavily on whether or not T-junctions are identified correctly as occluding or non-occluding. Where T-junctions are identified correctly, the geometric results are nearly as good as can be obtained from labelled drawings (except in those few cases such as Figure 20 where the assumption of major axis alignment is misleading), but when T-junctions are misidentified, the geometric results can be very poor.

13.4 Results—T-Junctions

We now concentrate on the important issue of whether or not T-junctions were identified correctly. See Table 2, where the left column denotes the Figure under test (those without T-junctions are omitted), and the next four columns list, respectively, the number of occluding T-junctions correctly identified, the number of K-type (non-occluding) T-junctions correctly identified, the number of occluding T-junctions misidentified as K-type (non-occluding), and the number of K-type (non-occluding) T-junctions misidentified as occluding. (The final column indicates whether or not the drawing can be reliably interpreted correctly using line-labelling methods; "*" indicates that pockets are misidentified as holes or vice versa but that interpretation is otherwise correct; these results come

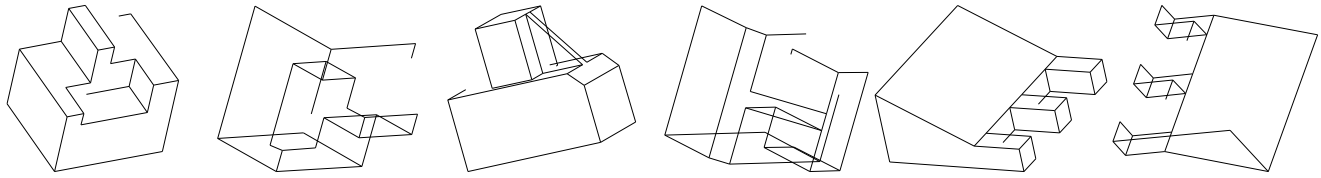

Figure 79. Successful Inflation

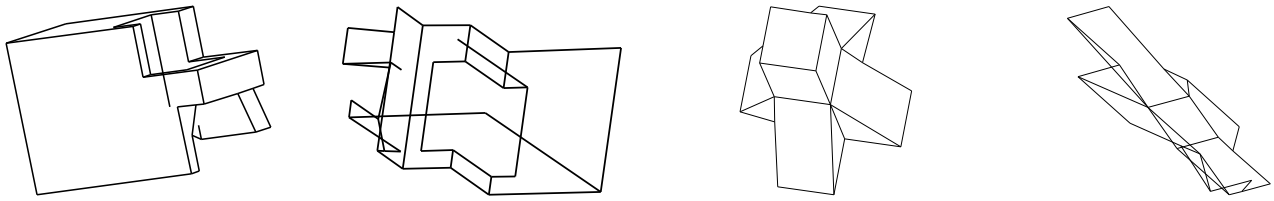

Figure 80. Unsuccessful Inflation

		Correct		Incorrect		Label?
Fig.	Output	Occ.	K	K	Occ.	
	Should be	Occ.	K	Occ.	K	
1		0	0	2	0	Y
2		0	0	4	0	*
3		4	0	0	0	*
4		0	0	6	0	Y
5		0	0	2	0	*
6		0	1	3	0	N
7		2	1	1	0	N
8		7	0	0	0	*
9		6	0	0	0	Y
11		2	1	0	0	N
12		1	0	0	0	Y
13		1	1	0	0	Y
14		1	1	0	0	N
16		0	0	2	0	Y
19		1	0	3	0	*
20		3	0	0	0	Y
Total		28/51	5/5	23/51	0/5	-

Table 3. Determination of T-Junctions Ia

from [39].) It is apparent that the results for this "typical set of engineering drawings" are not as good as for the larger dataset used in optimisation: only 75% of occluding T-junctions are identified correctly (although the few non-occluding T-junctions are all identified correctly). Figure 4 is a particular problem, suggesting that more work is needed for drawings where two regions correspond to the same face.

Processing rotated drawings, as described in Section 8.4, leads to the results shown in Table 3. On the whole, these are noticeably worse than those in Table 2—by allowing more possibilities, we have created more opportunities for mistakes. However, this approach does create correct (non-inverted) geometry for Figure 20, and identifies T-junctions correctly for Figures 3, 8 and 9 (which processing unrotated drawings does not). On the whole, though, the experiment of analysing rotated drawings in this manner cannot be considered a success. It is, of course, also slower by a factor of four.

Table 4 shows the corresponding results for the 32 drawings of objects with K-vertices (see [37]). These are a clear improvement on the more general results in Table 2: 95% of occluding T-junctions and 93% of non-occluding T-junctions are identified correctly. We suggest that the feature which makes it easy to interpret these drawings using our proposed new method is that most, but not quite all, faces and edges are axis-aligned. Identifying the object major axes is thus comparatively straightforward and reliable.

Table 5 shows the corresponding results for the 8 drawings of objects with higher-order vertices (Figures 54–61). These results are clearly poor: there are nearly as many mislabelled T-junctions as correctly-labelled ones. This reinforces the previous remark:

	Correct		Incorrect		Label?
Figs. Output	Occ.	K	K	Occ.	
Should be	Occ.	K	Occ.	K	
21	2	2	0	0	N
22	1	1	0	0	Y
23	2	1	0	0	Y
24	2	2	0	0	N
25	2	2	0	0	Y
26	1	1	0	0	N
27	2	1	0	0	N
28	5	2	0	0	N
29	0	2	2	0	Y
30	0	0	2	0	Y
Subtotal	16/20	13/13	4/20	0/13	5
31	2	2	0	0	Y
32	1	1	0	0	Y
33	2	2	0	0	N
34	1	1	0	0	Y
35	4	2	0	0	Y
36	3	2	0	0	Y
37	3	2	0	0	Y
38	3	1	0	1	*
39	4	2	0	0	N
40	2	2	0	0	Y
Subtotal	25/25	17/18	0/25	1/18	7+*
41	2	2	0	0	N
42	1	1	0	0	N
43	2	2	0	0	N
44	1	1	0	0	N
45	3	2	0	0	N
46	3	0	0	0	N
47	2	0	0	0	N
48	2	0	0	0	N
49	2	0	0	0	N
50	6	0	0	0	N
Subtotal	24/24	8/8	0/24	0/8	0
51	2	1	0	1	N
52	4	1	0	1	N
Subtotal	6/6	2/4	0/6	2/4	0
Total	71/75	40/43	4/75	3/43	12+*/32

Table 4. Determination of T-Junctions II

where there are many faces and edges pointing in a variety of different directions, identifying the object major axes is difficult and unreliable.

		Correct		Incorrect		Label?
Fig.	Output	Occ.	K	K	Occ.	
	Should be	Occ.	K	Occ.	K	
54		3	0	1	0	N
55		0	0	4	0	N
56		0	1	1	0	N
57		1	0	1	0	N
58		2	2	1	0	N
61		0	3	2	1	N
Total		6/16	6/7	10/16	1/7	-

Table 5. Determination of T-Junctions III

13.5 Speed

Although the iterative approach described here is not as fast as the labelling approach described elsewhere [38], interactive time requirements are met. 600 iterations of the relaxation loop typically take a fraction of a second when run on a Dell Optiplex SX270 with Intel Pentium(R) 4 GHz CPU. When the rotation idea described in Section 8.4 is used, this increases to approximately a second on average (approximately 10 minutes for a set of 591 drawings).

13.6 Discussion

For our 20 representative drawings, overall, line-labelling approaches label 11 entirely correctly, misidentify holes as pockets or vice versa in 5 drawings, and mislabels 4 drawings. Our new method inflates 12 drawings entirely correctly, gives z-inversion for one drawing (although T-junctions are identified correctly), and gives reasonable inflation but misidentifies one or more T-junctions for 7 drawings.

These results show why line labelling has survived the test of time: despite its occasional failures, there is no other method which is as successful at creating frontal geometry from typical drawings of engineering objects. In particular, no other method is as successful at identifying occlusion when it occurs.

Nevertheless, the method we present here remains a useful tool in line drawing interpretation. Consider, in particular, Figure 5. Is the hole loop feature a pocket or a through hole? Any attempt to label this runs the risk of introducing errors, as pockets and through holes are labelled differently. Our new approach avoids such errors, producing a good frontal geometry without adding misinformation.

Also, we have noted elsewhere [38] that no known labelling approach comes close to labelling Figure 11 correctly. The output produced for this drawing by our new method is not particularly good, but it meets the requirements of a provisional frontal geometry (relative z-coordinates are acceptable, and T-junctions are identified correctly as occluding and non-occluding), and is better than could be produced from a seriously incorrect labelling.

Similarly, for an entire category of drawings (those with chains of collinear K- and T-junctions, such as Figures 21, 35, 46 etc.), the new approach we propose here produces a more reliable frontal geometry than previous methods such as [37]. Our new method admittedly produces less information, but in doing so avoids producing incorrect information, and what remains is sufficient for our purposes.

Finally, although the results in Table 5 cannot be considered impressive, the method we have outlined is at present the *only* existing approach which can create a frontal geometry for these drawings. Labelling is not an option for these drawings.

14 Conclusions

14.1 Results

The approach we have considered in this paper is, to some extent, extreme, in that ideas which are related to but fall short of full labelling have deliberately been excluded, to see just how far this new approach could stand on its own.

The results are varied. For all but one of our typical set of 20 drawings of engineering objects, we can inflate the drawing to a satisfactory frontal geometries; identification of occlusion is less reliable, resulting in one or more errors for seven of the drawings. By comparison, previous methods cannot reliably create frontal geometries for four of these drawings (although in general, and particularly for trihedral objects, line labelling using the Clowes-Huffman catalogue [1, 9] followed by labelling-based inflation [33], the quality of inflation is better).

The frontal geometries of a more restricted set of objects which include K-vertices, where most faces and edges are aligned with the object major axes, and where those which are not can be easily identified, are better than can be achieved by previous methods. The frontal geometries of objects containing uncatalogued vertices, such as 6-hedral, 7-hedral and 8-hedral extended-K vertices, are poor, but (by avoiding labelling) our new approach can create frontal geometries from these drawings, whereas previous approaches cannot.

The answer to the question *is line labelling necessary* seems to be equally mixed. A perfectionist would say that it is—we want to deduce as much as possible from the drawing, and that includes line labels. However, reasonable and useful (if imperfect) results can be obtained without it for a representative sample of typical engineering objects. In addition, our new method improves on labelling-based methods for some categories of objects, and there are other objects which cannot be labelled at all using existing methods.

A more balanced conclusion would be that although we can often do without line labelling, there are undoubtedly times when it supplies information which is complementary to our approach. This suggests that an investigation into how they two approaches could be combined would be worthwhile.

14.2 Future Work: Improvements

The rotation idea presented in Section 8.4 might produce better results than those shown in Table 3 if:

- we re-optimised the tuneable parameters specifically for this idea, rather than use the same values as for the other tests;
- we had a more sophisticated method of choosing the best of the four possibilities as our final answer (the existing test is simply that the various compliance functions are most in agreement).

14.3 Future Work: Integrated Approach

In considering how further to improve this approach, the next question to be answered is therefore *to what extent is labelling necessary?* For example, can the method presented here be improved by incorporating ideas which, in this investigation, were omitted because they are conceptually similar to labelling?

Clearly, there are cases where a partial labelling is useful but a full labelling could be misleading. Consider, for example, Figure 5. Most of the lines in this drawing can be labelled helpfully and unambiguously, but existing algorithms cannot determine whether the central feature is a pocket (three concave lines) or a through hole (one concave line and two occluding lines). We can make the case that it is better to leave this unresolved than to make a definite decision which may turn out to be wrong.

Even so, *some* labelling is undoubtedly useful. One example would be identification of candidate cubic corners—anything unambiguously labelled with a non-trihedral label cannot be a cubic corner. Another

would be the obvious fact that if the tail of a T-junction is concave, the junction *must* be occluding—incorporating this rule would, for example, lead to a correct interpretation of Figure 3. Even incorporating just the fundamental labelling assumption that a line segment must have the same label at either end would be enough to avoid the error made with Figure 38—a T-junction cannot be occluding if the other end of the "occluding" line segment is clearly convex.

Information can similarly flow in the other direction. Clearly, one potential solution to the line-labelling problem is to produce the frontal geometry first and determine geometrically, from the face planes, whether edges are convex, concave or occluding. This is essentially the approach taken in [38], but the inflation methods we present here are an advance on the simpler approach taken there, and also allow for the possibility of feedback from a labelling mechanism which produces likely, but not certain, line labels.

Such ideas will be investigated in the future.

14.4 Future Work: Symmetry

In principle, we should wish to make use of symmetries implied by the drawing. To illustrate this, it is clear that, when looking at Figure 58, a human will quickly realise that the object depicted has a plane of mirror symmetry. We do not know how humans do this; algorithms we know of for detecting partial symmetries in vertex-edge graphs are nowhere near as reliable as human judgement.

If a good method for quickly finding candidate partial symmetries from vertex-edge graphs could be found, assessing such candidate symmetries using either topological information (line labels) or geometrical information (inflation output)—or both—would be straightforward.

14.5 Summary

In conclusion, the ideas presented in this paper, while not in themselves a full solution to the frontal geometry problem, present pointers as to how two of the major obstacles can be avoided.

Firstly, they indicate how we may avoid the "validation" problem of how to demonstrate that the frontal geometry is the one a human would expect, by basing our approach on configurations humans are known to see in line drawings.

Secondly, they indicate how we may avoid two problems presented by traditional line labelling approaches. The first is the one of ever-larger junction catalogues being required to accommodate ever-rarer drawings containing unusual junctions; the second is the ever-increasing number of alternative labellings deemed to be valid by such higher-order catalogues.

The change of emphasis away from line-labelling as a self-contained problem and towards a more integrated approach to frontal geometry is promising.

15 Acknowledgements

Funding for this investigation was provided by Japan Society for the Promotion of Science Fellowship number P03717; this support is acknowledged with gratitude.

References

[1] M.B. Clowes, *On Seeing Things,* Artificial Intelligence, **2**, 79–116, 1970.

[2] M.C. Cooper, *Efficient Systematic Analysis of Occlusion,* Pattern Recognition Letters **7**, 259–264, 1988.

[3] S.W. Draper, *Reasoning about depth in line-drawing interpretation,* PhD Thesis, Sussex University, 1980.

[4] J.T. Enns and R.A. Rensink, *Preattentive recovery of three-dimensional orientation from line drawings,* Psychological Review, 98, 101-118, 1991.

[5] I.J. Grimstead, *Interactive Sketch Input of Boundary Representation Solid Models,* PhD Thesis, Cardiff University, 1997.

[6] A. Guzman, *Decomposition of a Visual Scene into Three-Dimensional Bodies,* AFIPS Proc. Fall Joint Computer Conference, **33**, 291–304, 1968.

[7] J.H. Han, *Survey of Feature Research,* Technical Report IRIS-96-346, University of Southern California Institute for Robotics and Intelligent Systems, 1996.

[8] J.H. Han, *On Multiple Interpretations,* in Ed. C. Hoffmann and W. Bronsvort, Proceedings: Fourth Symposium on Solid Modeling and Applications, 311–321, ACM Press, 1997.

[9] D.A. Huffman, *Impossible Objects as Nonsense Sentences,* Machine Intelligence **6**, 295–323, New York American Elsevier, 1971.

[10] D.L. Jenkins, *The Automatic Interpretation of Two-Dimensional Freehand Sketches,* PhD Thesis, University of Wales College of Cardiff, 1992.

[11] T. Kanade, *Recovery of the Three-Dimensional Shape of an Object from a Single View,* Artificial Intelligence **17**, 409–460, 1981.

[12] D. Lamb and A. Bandopadhay, *Interpreting a 3D Object From a Rough 2D Line Drawing,* In ed. A.E.Kaufman, Proceedings of the First IEEE Conference on Visualization '90, 59-66, IEEE, 1990.

[13] W.H. Lau and E. Hancock, *Probabilistic relaxation and hierarchical relaxation,* Technical report, Dept of Computer Science, University of York, 1993.

[14] Y.G. Leclerc and M.A. Fischler, *An Optimization-Based Approach to the Interpretation of Single Line Drawings as 3D Wire Frames*, International Journal of Computer Vision **9**(2), 113–136, 1992.

[15] K.W. Lee, *Principles of CAD/CAM/CAE Systems,* Addison Wesley, 1999.

[16] H. Lipson, *Computer Aided 3D Sketching for Conceptual Design*, PhD Thesis, Technion-Israel Institute for Technology, Haifa, 1998.

[17] H. Lipson and M. Shpitalni, *Optimization-based Reconstruction of a 3D Object from a Single Freehand Line Drawing,* Computer-Aided Design **28**(8), 651-663, 1996.

[18] J. Malik, *Interpreting Line Drawings of Curved Objects,* International Journal of Computer Vision **1**, 73–103, 1987.

[19] T. Marill, *Emulating the Human Interpretation of Line-Drawings as Three-Dimensional Objects,* International Journal of Computer Vision **6**(2) 147–161, 1991.

[20] J. Mitani, H. Suzuki and F. Kimura, *3D SKETCH: Sketch Based Model Reconstruction and Rendering,* in Ed. U. Cugini and M.J. Wozny, ¿From Geometric Modeling to Shape Modeling, IFIP TC5 WG5.2 Seventh Workshop on Geometric Modeling: Fundamentals and Applications, October 2-4, 2000, Parma, Italy, 85–98, Kluwer, 2001.

[21] P. Parodi, R. Lancewicki, A. Vijh and J.K. Tsotsos, *Empirically-Derived Estimates of the Complexity of Labeling Line Drawings of Polyhedral Scenes,* Artificial Intelligence **105**, 47–75, 1998.

[22] D.N. Perkins, *Cubic Corners,* Quarterly Progress Report 89, 207–214, MIT Research Laboratory of Electronics, 1968.

[23] A. Piquer, P. Company and R.R. Martin, "Skewed Mirror Symmetry in the 3D Reconstruction of Polyhedral Models", J. Winter School on Computer Graphics **11**(3) 504–511, 2003.

[24] A. Rosenfeld, R.A. Hummel and S.W. Zucker, *Scene Labelling by Relaxation Operations,* IEEE Transactions on Systems, Man and Cybernetics, **6**, 420–433, 1976.

[25] M.M. Samuel, A.A.G. Requicha and S.A. Elkind, *Methodology and Results of an Industrial Parts Survey.* Technical Memorandum 21, Production Automation Project, University of Rochester NY USA, 1976.

[26] V. Sashikumar and M. Sohoni, *Reconstruction of Feature Volumes and Feature Suppression*, in ed. K.Lee and N.Patrikalakis, Proceedings, Seventh ACM Symposium on Solid Modelling and Applications SM'02, 60–71, ACM Press, 2002.

[27] V. Sashikumar, M. Sohoni and R. Rajadhyaksha, *Removal of Blends from Boundary Representation Models*, in ed. K.Lee and N.Patrikalakis, Proceedings, Seventh ACM Symposium on Solid Modelling and Applications SM'02, 83–94, ACM Press, 2002.

[28] T.M. Sezgin, T. Stahovich and R. Davis, *Sketch Based Interfaces: Early Processing for Sketch Understanding*, Proceedings of 2001 Perceptive User Interfaces Workshop (PUI'01), 2001.

[29] K. Sugihara, *Machine Interpretation of Line Drawings,* MIT Press, 1986.

[30] K. Sugihara, *Three-dimentional realization of anomalous pictures—An application of picture interpretation theory to toy design,* Pattern Recognition **30**(7), 1061–1067, 1997.

[31] P.A.C. Varley, H. Suzuki, J. Mitani and R.R. Martin, *Interpretation of Single Sketch Input for Mesh and Solid Models,* International Journal of Shape Modelling **6**(2), 207–241, 2000.

[32] P.A.C. Varley and R.R. Martin, *The Junction Catalogue for Labelling Line Drawings of Polyhedra with Tetrahedral Vertices,* International Journal of Shape Modelling **7**(1), 23–44, 2001.

[33] P.A.C. Varley and R.R. Martin, *Estimating Depth from Line Drawings.* In Ed. K.Lee and N.Patrikalakis, Proc. 7th ACM Symposium on Solid Modeling and Applications, SM02, 180–191, ACM Press, 2002.

[34] P.A.C. Varley, *Automatic Creation of Boundary-Representation Models from Single Line Drawings,* PhD Thesis, Cardiff University, 2002.

[35] P.A.C. Varley, Sketches of Polyhedral Solids. `http://ralph.cs.cf.ac.uk/Data/ Sketch.html`, 2003.

[36] P.A.C. Varley and R.R. Martin, *Deterministic and Probabilistic Approaches to Labelling Line Drawings of Engineering Objects,* International Journal of Shape Modelling **9**(1), 79–99, 2003.

[37] P.A.C. Varley, H. Suzuki and R.R. Martin, *Interpreting Line Drawings of Objects with K-Junctions*, Proc. Geometric Modeling and Processing 2004, Eds. S.-M. Hu, H. Pottmann, 249–358, 2004.

[38] P.A.C. Varley, R.R. Martin and H. Suzuki, *Making the Most of Using Depth Reasoning to Label Line Drawings of Engineering Objects*, in ed. G. Elber, N. Patrikalakis and P. Brunet, 9th ACM Symposium on Solid Modeling and Applications SM'04, 191–202, 2004.

[39] P.A.C. Varley, R.R. Martin and H. Suzuki, *Making the Most of Using Depth Reasoning to Label Line Drawings of Engineering Objects*, to appear in ASME Transactions, Journal of Computing and Information Science in Engineering, special issue on Solid Modeling, 2005.

[40] T. Vetter and T. Poggio, *Symmetric 3D objects are an easy case for 2D object recognition*, Human Symmetry Perception, ed C.W.Tyler, 349–359, 1996.

[41] W. Whiteley, *From a Line Drawing to a Polyhedron*, Journal of Mathematical Psychology **31**, 441–448, 1987.