Perpendicularity as a Key to Interpreting Line Drawings of Engineering Objects

Ralph Martin
Cardiff University
Wales, UK
Email: ralph@cs.cf.ac.uk

Peter Varley
The University of Tokyo
Tokyo, Japan
Email: pvarley@den.rcast.u-tokyo.ac.jp

Hiromasa Suzuki
The University of Tokyo
Tokyo, Japan
Email: suzuki@den.rcast.u-tokyo.ac.jp

Abstract

Many recent approaches to the problem of interpreting line drawings as solid objects treat inflation as a two-stage approach, the first stage being to produce a quick initial estimate of vertex z-coordinates, and the second being to refine these initial estimates to produce a "more beautiful" geometry.

By making assumptions about engineering objects and the ways people see and depict them, it is often possible to reproduce a single object which humans will agree is the *correct* interpretation of the drawing.

Of these assumptions, those to do with perpendicularity are most important, in part because perpendicularity is the most common regularity in engineering objects, and in part because of the importance of perpendicularity in the human perception process. In this paper, we catalogue various possible instances of perpendicularity in engineering drawings.

Keywords: Line Drawing, Perpendicularity, Tools for Creative Design

1 Introduction

1.1 Terminology

A solid model of a 3D *object* describes the *topology* and *geometry* of its *faces*, *edges* and *vertices*. *Topology* is discrete (e.g. connectivity between vertices and edges), whereas *geometry* is continuous (e.g. the spatial coordinates of vertices). A *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 in general 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].

Note the careful distinction between 2D ideas (drawings, regions, lines, junctions) and 3D ideas (objects, faces, edges, vertices).

A *frontal geometry* is an intermediate stage between 2D drawing and 3D object (and, for this reason, is sometimes called *2½D*). 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.

A *natural line drawing* shows the *visible edges* of a *polyhedral object*. A *wireframe drawing* shows all edges of a polyhedral object, including *hidden edges*.

A polyhedron is *trihedral* if exactly three faces meet at each vertex. It is *extended trihedral* [Parodi et al] 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. *Higher-order vertices* are also possible, albeit not common.

1.2 Motivation

In principle, it is impossible to interpret a line drawing (either a natural line drawing or a wireframe drawing) as a solid object – an infinite number of possible solid objects could, if viewed from the appropriate viewpoint, result in the same line drawing. In practice, by making assumptions about engineering objects and the ways people see and depict them, it is often possible to reproduce a single object which humans will agree is the *correct* interpretation of the drawing.

Of these assumptions, those to do with perpendicularity are most important, in part because perpendicularity is the most common regularity in engineering objects [Samuel et al], and in part because of the importance of perpendicularity in the human perception process [Perkins].

In this paper, we catalogue various possible instances of perpendicularity in engineering drawings, and ways of representing them as *compliance functions* for use in linear systems for finding z-coordinates. Some are well-known; others have not been investigated before. Section 2 gives an overview of perpendicularity in engineering objects and how the assumption of perpendicularity has been used in recent approaches to line drawing interpretation. Section 3 describes Perkins's *Cubic Corners* [Perkins], the first formalised approach of perpendicularity hypotheses in interpreting line drawings. Section 4 introduces face-edge perpendicularity for trihedral vertices. Section 5 considers the case of junctions of only two lines.

Section 6 considers how such ideas can be applied to interpreting drawings containing *K*-vertices. Section 7 considers how such ideas can be applied to face-vertex coplanarity, and in particular to drawings containing hole loops. Section 8 considers when and how parallelograms in the drawing may be interpreted as rectangles in 3D. Finally, Section 9 presents our conclusions.

This paper does not investigate other common angles such as 60°, although in principle the ideas presented could be extended to such angles. Firstly, there is no other angle which occurs anywhere near so often in engineering objects as the right angle. Secondly, while there is abundant evidence that humans regularly perceive potential cubic corners as junctions of three 90° angles [Perkins], there seems as yet to be no clear evidence whether humans preferentially perceive other angles as subdivisions of 360°, e.g. 15°, 30°, 45°, 60°, 75° or as angles with tangents which are ratios of small numbers, e.g. $\tan^{-1}(1)$, $\tan^{-1}(^1/_2)$, $\tan^{-1}(^1/_3)$, $\tan^{-1}(^1/_4)$.

2 Overview

In considering many approaches taken recently [Company et al, Grimstead, Shesh et al, Varley 2003, Varley 2004a] to inflating drawings (both natural line drawings and wireframe drawings) into the third dimension, there is a similarity which at first sight may appear surprising. All of these approaches use a two-pass approach, the first pass being to produce a quick initial estimate of vertex *z*-coordinates, and the second being to refine these initial estimates to produce a "more beautiful" geometry.

The similarity becomes less surprising when the reasons for doing this are considered. In trying to interpret a drawing as a solid object, it is necessary to make preliminary assumptions (for inflation, these assumptions will be geometric). Some of these assumptions will be correct; others will be incorrect. In order to determine which is which, it is necessary to have a preliminary inflated geometry against which the merits of the various assumptions can be tested.

Since the purpose of the first stage is to obtain quick estimates of depth, it is no surprise that all of the considered approaches use linear methods for preliminary inflation. [Varley 2003] uses overconstrained linear systems of equations where the variables are the vertex *z*-coordinates. [Grimstead] uses a more complicated linear system in which components of the face equations (normals and distances) are additional variables (see Section 7). [Varley et al 2004a] uses three overconstrained linear systems of equations, where the variables are the vertex coordinates in object space. [Shesh et al] uses a "layering" approach, as (for the general case) does [Company et al]. For the specific case of normalons [Company et al], by careful choice of assumptions, obtains depth coordinates directly by propagation.

In general, as the best compromise between speed and effectiveness, we prefer the first approach listed, overconstrained linear systems of equations where the variables are the vertex *z*-coordinates. There are known (and fast) algorithms for finding the best fit to an overconstrained system of linear equations – of these, we prefer [Bauer]. For this reason, in the sections which follow, we note which assumptions can be translated into equations linear in *z*-coordinates.

It is possible to do additional processing between the first and second stages, although of the considered approaches, only [Varley 2003] does this (preliminary inflation is performed on only the part of the object visible in the drawing, but final geometric fitting is performed on the entire object after construction of hidden topology).

Most of the considered approaches treat the second stage as a non-linear optimisation process, and for this reason we include an explicit-equation form of each compliance function. The exception is [Grimstead], which continues to use the *PQzD* linear system approach iteratively, dropping the equation with the largest residual (which, it is assumed, represents the remaining assumption least compatible with the others) until a consistent geometry is obtained.

[Shesh et al] uses iterations of Brent's method [Brent] to refine vertex depth coordinates. [Varley et al 2004a] uses an iterative relaxation process in which, on each iteration, the new values of a vertex's depth coordinate is recalculated from the old values of related vertex depth coordinates; the weighting applied to each compliance function is based on how close it is to matching the previous iteration's geometry. In all of these, since the optimisation process is non-linear, non-linear compliance functions can be used, but the chosen method requires that the equation form of the compliance function is explicit in the vertex depth coordinate.

Finally, and more generally, [Varley 2003] uses a sequential constraint enforcement procedure in which identification and enforcement of compatible constraints uses a downhill optimisation method [Nelder and Mead]. It should be noted that the results were not, in general, successful, although more recently [Langbein et al] has reported greater success with a similar approach using constraint reasoning to a greater extent, and using a more specialised downhill optimisation algorithm.

We note that none of these procedures guarantees convergence. Discussion of convergence is, in general, outside the scope of this paper.

3 Cubic Corners

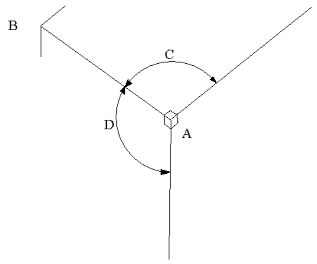

Fig. 1: Cubic Corner

Consider Fig. 1. [Perkins] showed that if *A* is a *cubic corner* (the junction of three mutually-perpendicular edges),

$$|z_B - z_A| = AB\sqrt{\left(\frac{-\cos C \cos D}{\cos E}\right)} \quad (1)$$

(where AB is the 2D length of the line, and E is the complementary angle 180°-C-D). From this, it follows that

$$z_B - z_A = \pm AB\sqrt{(\tan C \tan D - 1)} \quad (2)$$

This is the recommended form for use in linear systems, and is easily rearranged if an explicit equation is required for z_A given z_B or vice versa. Note that in order to use this, there must be a separate mechanism for determining whether *A* is in front of or behind *B*. This is obvious for boundary junctions, but usually not obvious, at least initially, for internal junctions; when used iteratively, the appropriate sign can generally be determined from the previous iteration's output, remembering that the signs alternate for *W*-junctions but are all the same at *Y*-junctions.

The method fails for junctions which do not meet the "Perkins criteria" (either a *W*-junction in which the two internal angles are acute but their sum is obtuse, or a *Y*-junction in which all three angles are obtuse), since the value under the square root becomes negative.

It can be noted that Perkins's interest was psychological: when do humans interpret the junction of three lines as a cubic corner? His experimental results showed that humans interpret junctions as cubic corners (a) when it is mathematically possible (the "Perkins criteria" given above), or (b) when one of the 2D angles is 90° [Perkins].

4 Face-Edge Perpendicularity

It may become apparent, in considering a trihedral vertex, that one of the 3D angles cannot be 90º. It then becomes appropriate to consider whether or not the third edge meeting at the vertex is perpendicular to the other two.

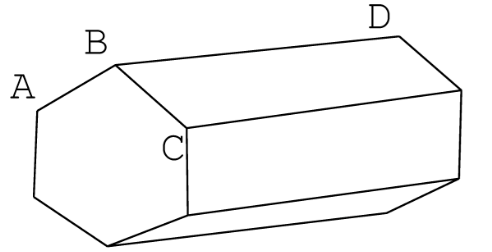

Fig. 2: Face-Edge Perpendicularity

Consider Fig. 2, and assume that we have somehow determined that angle ABC cannot be 90º in 3D. We still wish to make edge BD perpendicular to the plane containing ABC. Let BA, BC, BD be the 2D lengths of the respective edges, α be the angle CBD, and γ be the angle ABD. Clearly,

$$\overrightarrow{BA}.\overrightarrow{BD} = 0 \quad (3)$$

Separating this into the 2D dot product and the change in z-coordinates:

$$BA.BD\cos\gamma + (z_A - z_B)(z_D - z_B) = 0 \quad (4)$$

Similarly,

$$BC.BD\cos\alpha + (z_C - z_B)(z_D - z_B) = 0 \quad (5)$$

Note that BA, BC, BD, α and γ can all be measured directly from the drawing - they do not, for example, need to be recalculated on each iteration when using an iterative approach.

4.1 Out-Of-Plane Edge

Rearranging the above gives two explicit equations for z_D given values for z_B and either z_A or z_C:

$$z_D = z_B + \frac{BC.BD\cos\alpha}{z_B - z_C} \quad (6)$$

and

$$z_D = z_B + \frac{BA.BD\cos\gamma}{z_B - z_A} \quad (7)$$

There is no linear form. This is not a serious limitation, as face-edge perpendicularity is most likely to be used in the iterative refinement stage, not the initial inflation stage - initial inflation will, in most cases, assume a cubic corner.

4.2 In-Plane Edges

Substituting for z_D, as calculated from one of the explicit equations, in the other, and rearranging:

$$(z_B - z_C)BA\cos\gamma = (z_B - z_A)BC\cos\alpha \quad (8)$$

This result can be used either in a linear system or, by rearrangement, to give explicit equations for any of the three depth coordinates in terms of the other two.

5 Edge-Edge Perpendicularity

The traditional approach for obtaining relative depth coordinates by assuming that two vectors are perpendicular is Kanade's *skewed symmetry* [Kanade]. However, Kanade's analysis includes another assumption, that of isometry (equal line lengths in 2D correspond to equal edge lengths in 3D), and this is not necessarily justifiable. For this reason, we present an alternative approach (used for parallelograms in [Varley et al 2004b]).

Our objective is to make two edges perpendicular in 3D. Consider *A* as the vertex at which edges AB and AC meet. We wish to enforce:

$$\overrightarrow{AB}.\overrightarrow{AC} = 0 \quad (9)$$

Separating this into the 2D dot product and the change in z-coordinates:

$$AB.AC\cos\alpha + (z_B - z_A)(z_C - z_A) = 0 \quad (10)$$

(where α is the angle BAC), or

$$z_A^2 - Mz_A + N = 0 \quad (11)$$

where

$$M = (z_B + z_C) \quad (11a)$$

and

$$N = (z_B z_C + AB.AC.\cos\alpha) \quad (11b)$$

Solving for z_A gives two possible predictions: in an iterative process, we can choose the one nearer to the current value with reasonable confidence. There is no linear form.

$$z_A = \frac{1}{2}\left(M \pm \sqrt{M^2 - 4N}\right) \quad (11c)$$

6 Drawings with *K*-Vertices

K-vertices occur when four faces meet at a single vertex, with one of the face normals being a linear combination of two others. Their shape is, typically, that of a capital letter **K**. The three drawings in Figure 3 all include *K*-vertices.

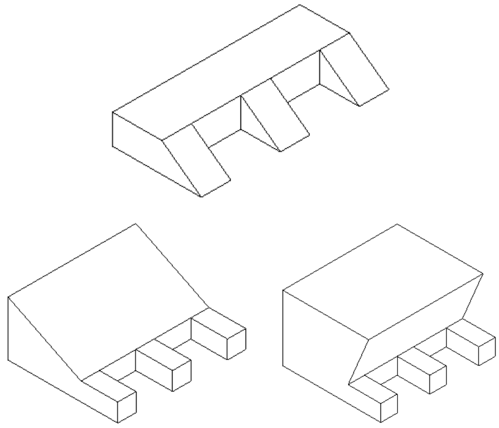

Fig. 3: *K*-Vertices

Two of the edges meeting at a K-vertex must be collinear. Encoding this as a linear equation is straightforward. As can be seen from the drawing, the face normals of the three faces adjacent to one or both of the collinear lines are not linearly independent; this is a consequence of the collinearity, and has no additional significance.

Often, the face normal of the remaining, independent face will be parallel with the two collinear edges; this is the case in all three of the objects in Fig. 3. This hypothesis can be encoded using the equations from Section 4.

Also, the non-collinear edges are often perpendicular in 3D to the collinear edges; this is the case with all the non-collinear edges at *K*-vertices in Fig. 3. This hypothesis can be encoded using the equations from Section 5.

7 Face-Vertex Coplanarity and Hole Loops

Vertices lie on faces (we do not discuss here the problem of determining *which* vertices lie on *which* faces). Given that we have some means of identifying which vertices we believe lie on a particular face, our objective is to make those vertices coplanar. (The inclusion of coplanarity in a paper on perpendicularity is justified theoretically by considering coplanarity in terms of equating vector cross-products; the practical reason for including it is its use in Section 8.)

Determining the equation of the plane of a face from its vertices is straightforward. Following [Grimstead], we recommend using a weighted linear system [Bauer] where the weights reflect confidence that the vertex lies in the plane of the face and the variables are those (*P*, *Q* and *D*) of the face equation

$$Px + Qy + z + D = 0 \quad (12)$$

Note that, given a face equation of the form $PCx + QCy + Cz + CD = 0$, the general position assumption requires that $C \neq 0$, so we can divide through by C [Grimstead].

Having solved for the face equation, it is straightforward to make a prediction of the *z*-coordinate of any vertex *V* which might lie on a face: $z_V = -(Px_V + Qy_V + D)$. This process is, obviously, non-linear, and as such only useful for the iterative refinement stage (by which time we can hope to have reasonable confidence that our judgement of which vertices lie on which faces is correct).

As a special case of vertex coplanarity, we note that *four*-vertex coplanarity does result in a linear equation. Given that we wish to make four vertices *ABCD* coplanar, we can express the vector AD as a linear combination of vectors AB and AC:

$$\overrightarrow{AD} = m\overrightarrow{AB} + n\overrightarrow{AC} \quad (13)$$

where *m* and *n* can be calculated from the known *x* and *y* coordinates of the vertices; rearranging this and extracting only the *z*-components gives

$$(m + n - 1)z_A - mz_B - nz_C + z_D = 0 \quad (14)$$

which can be used directly in a linear system or rearranged to give an explicit equation for any one vertex *z*-coordinate in terms of the other three.

As [Lipson] points out, making groups of four vertices coplanar does not necessarily make the entire face coplanar (except, trivially, for quadrilateral faces). However, four-vertex coplanarity is particularly useful for ensuring that holes, pockets and bosses such as those in Fig. 4 are coplanar with the enclosing face [Varley 2003].

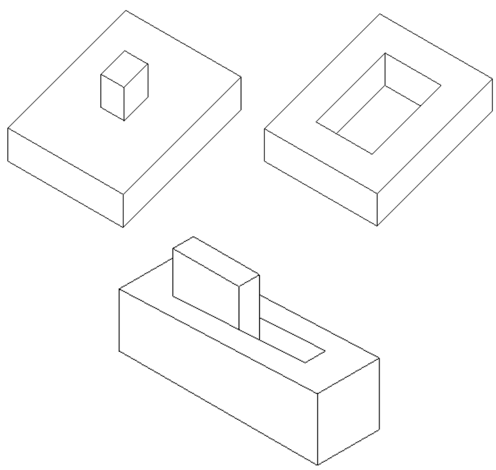

Fig. 4: Bosses, Pockets and Holes

8 Parallelograms as Rectangles

In [Varley et al 2004b], we showed 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. The mirror planes of Figure 2 and the non-axially-aligned faces of Figure 3 illustrate this. Here, we do not discuss how candidate parallelograms are identified.

Given that we wish to make four vertices form a rectangle in 3D, we are specifically aiming for both rectangularity (as in Section 5) and coplanarity (as in Section 7). In the absence of any straightforward way of enforcing both at once, we recommend using four-vertex coplanarity in the initial linear inflation step, and both vertex coplanarity and perpendicularity in the later iterative refinement.

9 Conclusions

This paper collects together several uses of the concept of perpendicularity to inflation of line drawings. All but the most recent (Section 4) have been used with success in practical systems. The encoding of these functions is presented in one or both of two forms: a linear form, suitable for inclusion in a rapid preliminary inflation, and an explicit (possibly non-linear) form, suitable for use in a slower iterative refinement process.

10 Acknowledgements

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

References

L. BAUER, 1971. *Elimination with Weighted Row Combinations for Solving Linear Equations and Least Squares Problems*, Handbook for Automatic Computation **II**, Linear Algebra, eds. J.H.Wilkinson and C.Reinsch, Springer-Verlag.

R.P. BRENT, 1973, *Algorithms for Minimization without Derivatives*, Prentice-Hall.

P. COMPANY, J.M. GOMIS and M. CONTERO, 1999. *Geometrical Reconstruction from Single Line Drawings Using Optimization-Based Approaches*, Proc. of WSCG '99, 361-368.

I.J. GRIMSTEAD, 1997. *Interactive Sketch Input of Boundary Representation Solid Models*, PhD Thesis, Cardiff University.

T. KANADE, 1981. *Recovery of the Three-Dimensional Shape of an Object from a Single View*, Artificial Intelligence **17**, 409-460.

F.G. LANGBEIN, A.D. MARSHALL and R.R. MARTIN, 2004. *Choosing Consistent Constraints for Beautification of Reverse Engineered Geometric Models*, Computer Aided Design **36**(3), 261-278,

K.W. LEE, 1999) *Principles of CAD/CAM/CAE Systems*, Addison Wesley.

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

J.A. NELDER and R. MEAD, 1965. *A Simplex Method for Function Minimization*, Computer Journal **7**, 308-313.

P. PARODI, R. LANCEWICKI, A. VIJH and J.K. TSOTSOS, 1998. *Empirically-Derived Estimates of the Complexity of Labeling Line Drawings of Polyhedral Scenes*, Artificial Intelligence **105**, 47-75.

D.N. PERKINS, 1968. *Cubic Corners,* Quarterly Progress Report 89, 207-214, MIT Research Laboratory of Electronics.

M.M. SAMUEL, A.A.G. REQUICHA and S.A. ELKIND, 1976. *Methodology and Results of an Industrial Parts Survey*, Technical Memorandum 21, Production Automation Project, University of Rochester NY USA.

A. SHESH and B. CHEN, 2004. SMARTPAPER: An Interactive and User Friendly Sketching System, Computer Graphics Forum **23**(3), 301-310.

P.A.C. VARLEY, 2003. *Automatic Creation of Boundary-Representation Models from Single Line Drawings*, PhD Thesis, Cardiff University.

P.A.C. VARLEY, R.R. MARTIN and H. SUZUKI, 2004. *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.

P.A.C. VARLEY, R.R. MARTIN and H. SUZUKI, 2004. *Frontal Geometry from Sketches of Engineering Objects: Is Line Labelling Necessary?* submitted to Computer Aided Design.