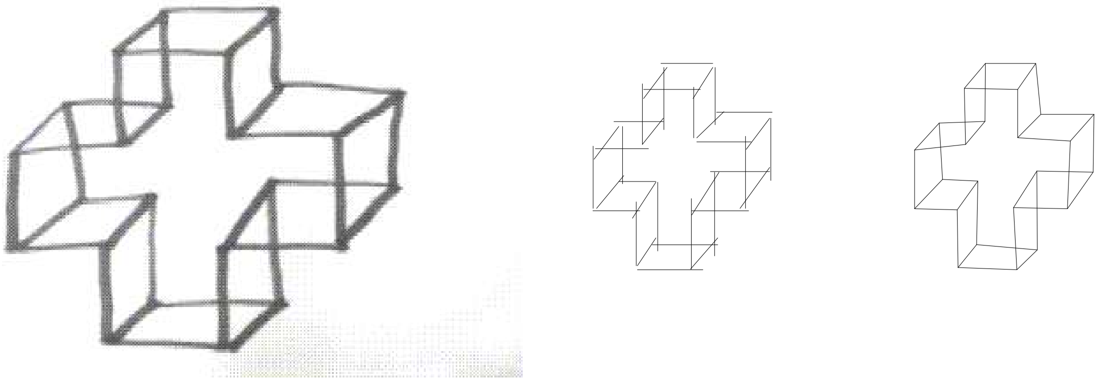
Reconstructing
Rectangular Polyhedra
From
Hand-Drawn Wireframe Sketches

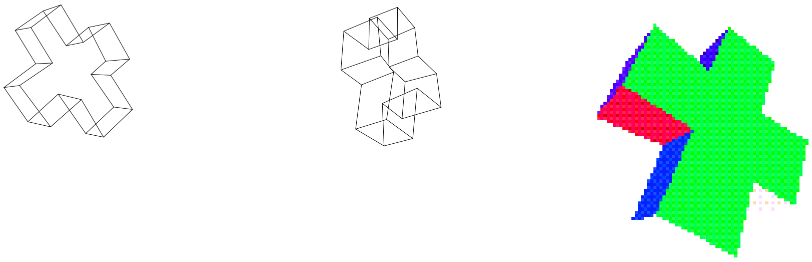
Jeremy S. De Bonet
Computer Graphics 6.837
Final Project
December, 1995

Abstract

Human observers are capable of interpreting hand drawn sketches as three- dimensional objects, despite inconsistencies in lengths, variability in angles, and unconnected vertices. The current system is an attempt to achieve such robust performance in the limited domain of sketches of wireframe rectangular polyhedra. The first version of this system reconstructs three-dimensional objects from perfect drawings, in which all angles and line junctions are consistent with projections of rectangular polyhedron. Ambiguities which are inherent in such drawings are avoided by choosing a *line grammar* which yields only a single interpretation. Next, reconstruction from imperfect drawings, in which all the line segments were randomly perturbed, was then achieved by grouping line endpoints into vertices while simultaneously restricting lines to particular orientations, and recovering three-dimensional form from the corrected line drawing. Finally, when actual hand-drawn sketches were used as input, we found that to successfully perform reconstruction the constraints on line orientations had to be replaced with constraints segment lengths and an additional three-dimensional point clustering process was needed.



Original image, initial line recovery, and lines after 2D clustering



Two views of the recovered 3D lines, and the corresponding surfaces.

Figure 1: *An example of the process through which the system described in this paper reconstructs 3D shape from a hand-drawn sketch.*

Introduction

Past systems which perform polyhedral reconstructions have been based on a variety of methods which attempt to determine a globally consistent interpretation of the scene. The extensive work which has successfully been done in this area can be grouped into two major themes: systems which rely on line labeling techniques, and systems which use gradient information to add additional constraints which are not present in line drawings.

Huffman and Clowes independently developed methods in which they labeled line types and enumerated all physically possible types of vertices in a dictionary, and then searched – initially simply depth-first and the eventually using more sophisticated pruning methods – for configurations in which all of the vertices could be labeled with entries from the dictionary (Huffman, 1971; Clowes 1971). Such techniques are capable of recovering scene geometry through the use large dictionaries and extensive searches though the space of possible legal combinations, however:

> "...the labeling scheme examined here still has problems: syntactically nonsensical scenes [can be] coherently labeled; scenes [and be] given geometrically impossible labels; and scenes that cannot arise from polyhedra are easily labeled." (Ballard and Brown, 1982, p. 298).

Another method, developed by Mackworth (1973), used brightness to determine surface gradients – in a fashion similar to the reflectance map computations later used by Horn (1977). The additional constraints were used to determine the equations of the planes on which the vertices in the line drawing lie. However, surface brightness information is not available in line drawings and yet human observers are capable of perceiving the three-dimensional polyhedra represented by them. This leads us to ask: what alternative methods could be used to solve this problem?

The observing system could assume a "grammatical" structure within the lines, which could be used to unambiguously interpret each line as having some particular three-dimensional orientation, *independently of the global structure within the figure.* Then, if the chosen grammar resulted in a physically impossible three-dimensional interpretation another grammar would be chosen. The polyhedron could be determined by searching over grammar-space, which is relatively small and independent of the scene, instead of over the potentially huge space of vertex classifications. One result of this method is that some drawings would have multiple consistent interpretations, each with a different corresponding grammar – such as is perceived in the bi-stable Necker cube illusion.

By assuming a simple grammar which just uses the line orientation to determine the represented three-dimensional displacement, simple polyhedral scenes such as the fireplace mantel shown in Figure 15-2 of Robot Vision (Horn, 1986) can be interpreted without determining a consistent labeling of each line.

In the case of the mantel drawing, the grammar which results in a reconstruction which conforms to human perception is one which maps vertical (90-180 degree) lines to y displacement, 150-330 degree diagonal lines to x displacement, and 30-210 degree diagonals to z displacement. In fact, any mapping of these orientations to an orthogonal triplet of vectors will result in rotated and scaled versions of the same polyhedron. One could imagine constructing a system which makes an orientation histogram of a drawing and maps the peak orientations to an arbitrary orthonormal basis set.

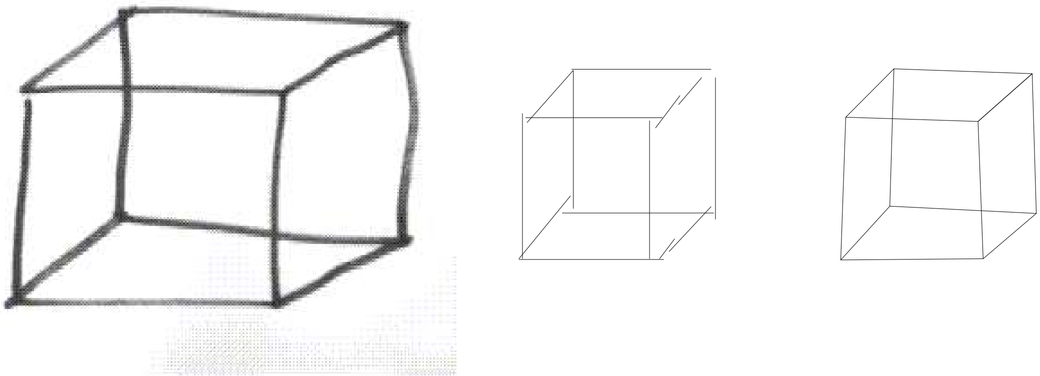
In the current system we circumvent this step by assuming, and restricting the input drawings to containing alignment around only a predetermined set of orientations. These peak orientations were chosen to be along the x axis, the y axis and 45-225 degrees and were mapped to the orthonormal three-space vectors along the x,y and z axes. This mapping (mapping) is used by the present system to parse wireframe sketches of rectangular polyhedral scenes.

Interpreting 3D Structure from Perfect Line Drawings

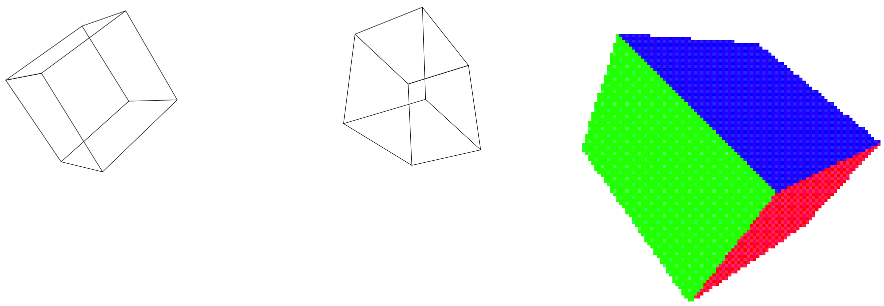
The simplest wireframe rectangular polyhedral scene consists of a single cube. With the grammar described above in place, reconstructing the three-dimensional structure can be accomplished by choosing a vertex to correspond to some arbitrary point in three-space and then tracing along each line and labeling each new vertex with the three-dimensional location of the previous point plus the displacement indicated by the length and orientation of the line between them. Clearly other reconstructions would be possible if another grammar had been chosen.

In most cases the scene perceived by human observers tend to be consistent with grammars which correspond lines in the the x and y directions, when present, with three-dimensional lines at a single depths, and lines along one or multiple diagonals traveling in depth; though this need not be the case: other grammatical rules result in stable interpretations as well.

The simplicity of this reconstruction, however is dependent on all paths between two points in the drawing resulting in the same interpreted three-dimensional displacement. In the case of the simple grammar we have chosen – a mapping three peak orientations to orthogonal three space vectors – only closed loops which consist



Original image, initial line recovery, and lines after 2D clustering

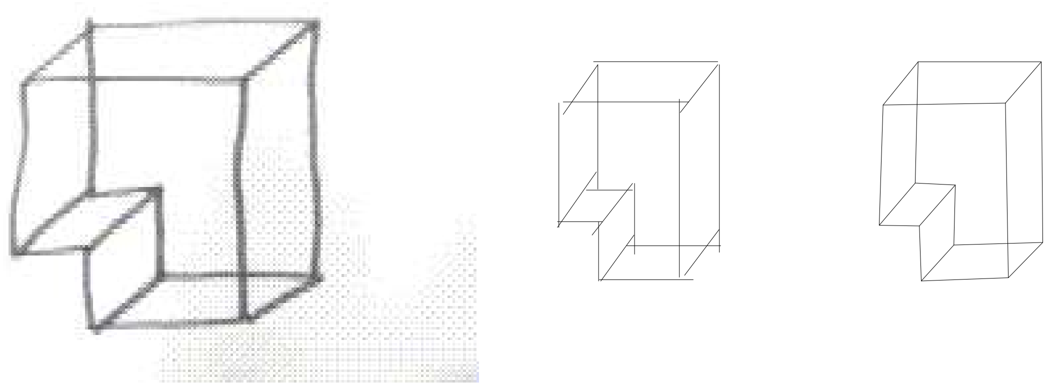


Two views of the recovered 3D lines, and the corresponding surfaces.

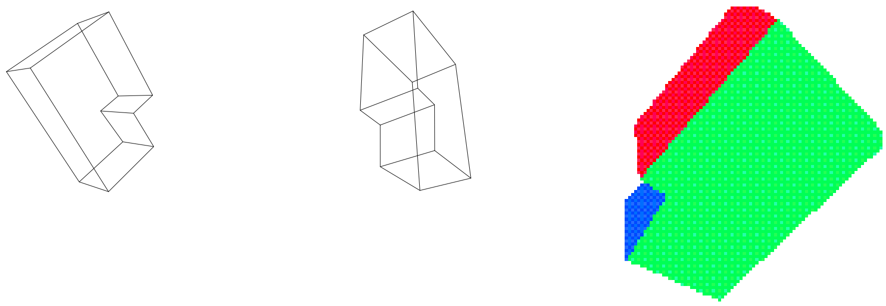
Figure 2: *Reconstruction of the simplest wireframe rectangular polyhedral scene: a cube.*

exclusively of lines of these orientations will be mapped to a closed loop in three dimensions. Closed loops which contain lines of other (non-peak) orientations will result in disconnected paths in three-dimensions, with the distance between the two disconnected ends depending on how we choose to map other orientations into three-space.

Yet when interpreting such drawings, human observers perceive lines of over a range of orientations as mapping to the same direction in three-dimensions, regardless of the fact that this yields a globally inconsistent object. For example consider the haphazardly drawn cube in Figure 2, even though the there are actually lines of many orientations – resulting in a drawing which could not be the orthographic projection of a rectangular polyhedral object – observers tend to ignore the inconsistencies and perceive a locally consistent rectangular object.



Original image, initial line recovery, and lines after 2D clustering



Two views of the recovered 3D lines, and the corresponding surfaces.

Figure 3: *Reconstruction of a more complex wireframe sketch.*

The Addition of Noise to the Input Lines

Once the initial reconstruction system, which was only capable of reconstructing wireframe drawings containing lines of three orientations was built, an additional layer was added to enable it to parse line drawings in which the endpoints each line line segment were perturbed by adding some random displacement. The effect of this noise was not only to disconnect lines which had come together to form vertices, but also to change the orientation of each line.

To find the intended vertices in the noisy line input, we take advantage of the fact that every vertex of wireframe rectangular polyhedral consists of the junction of the endpoints of three line segments. In this definition of vertex we exclude line intersections which occur due to vertices defined by the intersection of lines. Though intersection-defined vertices are not used to constrain the solution, they can be recovered through the recovery of each of its component lines. For an example of a reconstruction of a figure containing such an intersection-defined vertex see Figure 4.)

One approach to clustering the line endpoints into vertices would be to minimizing the total squared displacement necessary to move each endpoint such that it coincides with endpoint from two other lines. When there are no spurious or fragmented lines and the added noise is not so great that line endings are closer to some other vertex than their intended vertex, the global minimization solution results in each point moving to the average of it and its two closest neighbors. However, in practice spurious and fragmented lines result in a minimum displacement configuration which contains extra vertices and lines of orientations which are not intended in the drawing. Furthermore in noisy drawings which a total number of lines which is not divisible by 3, there is no such global solution.

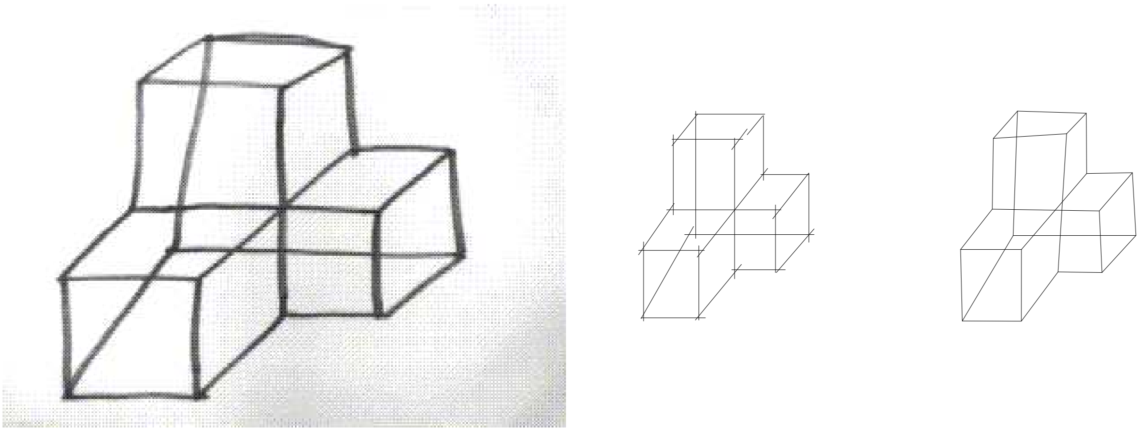
This global minimization can be modified, however, into a local iterative technique which does result acceptable vertex clustering. By assuming that locally these *fragmentless* conditions exit we can cluster the endpoints by iteratively moving each point to the average of it and its two closest neighbors. When three segment endings cluster to form a vertex, they become stable and do not change with additional iterations. Though it is possible that fragmented and spurious lines on of edges can cause problems by generating edges which are not intended, this does not occur often. Fragmented and spurious lines tend to move toward duplicating some other line which is in a stable configuration. These excess lines are easily removed by eliminating lines which are within some small displacement from another line.

This method successfully recovers vertices for drawings with noise of up to about 20 percent of the length of the shortest line segment – drawings which are difficult to interpret by human observers.

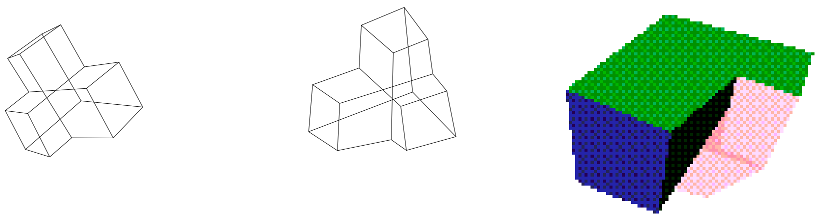
Results of this clustering operation on images acquired from actual hand-draw sketches (discussed later) are shown in the third panel of each figure.

Once the lines segment endpoints have been clustered, the resulting drawing contains lines of many orientations which are distributed around three peak orientations.

To use the grammar technique to attempt to parse such drawings chosen one



Original image, initial line recovery, and lines after 2D clustering

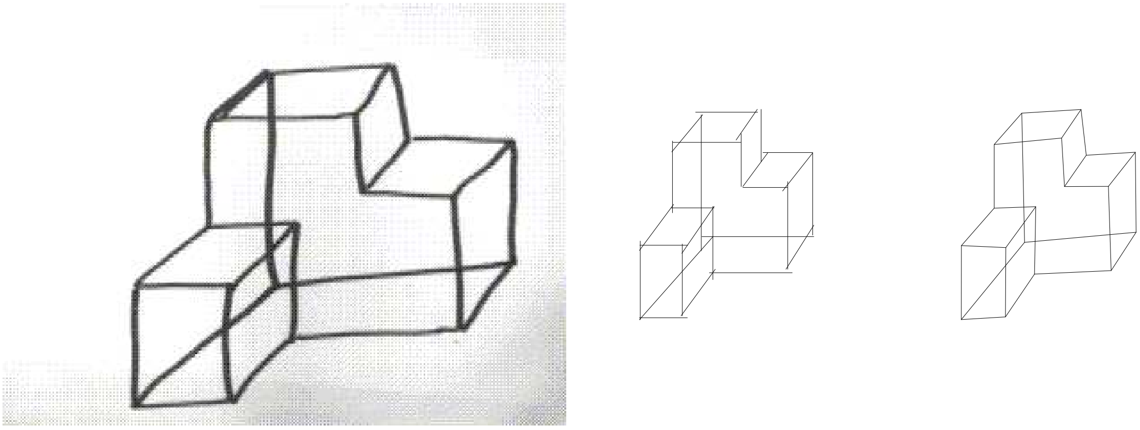


Two views of the recovered 3D lines, and the corresponding surfaces.

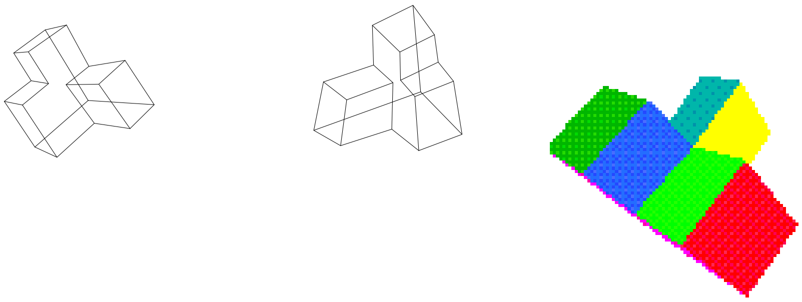
Figure 4: *Though it does not explicitly manipulate vertices defined by line intersections, the system can recover them.*

might consider using some method to project lines onto the three peak orientations. Because in two space any set of three orientations must be linearly dependent, any non-peak orientation can be projected to infinitely many possible combinations of the three peak orientations; therefore some rule must be used to decide which projection to use. Once a projection rule is chosen, the three-dimensional displacement represented by a line is the determined by the linear combination, indicated by its projection, of the displacements associated with the peak orientations.

One could imaging using a projection rule which maps a particular orientation to a linear combination of the two closest peak orientations, however, it is trivial to generate an example of a closed loop drawing which under such a rule maps to an open path.



Original image, initial line recovery, and lines after 2D clustering

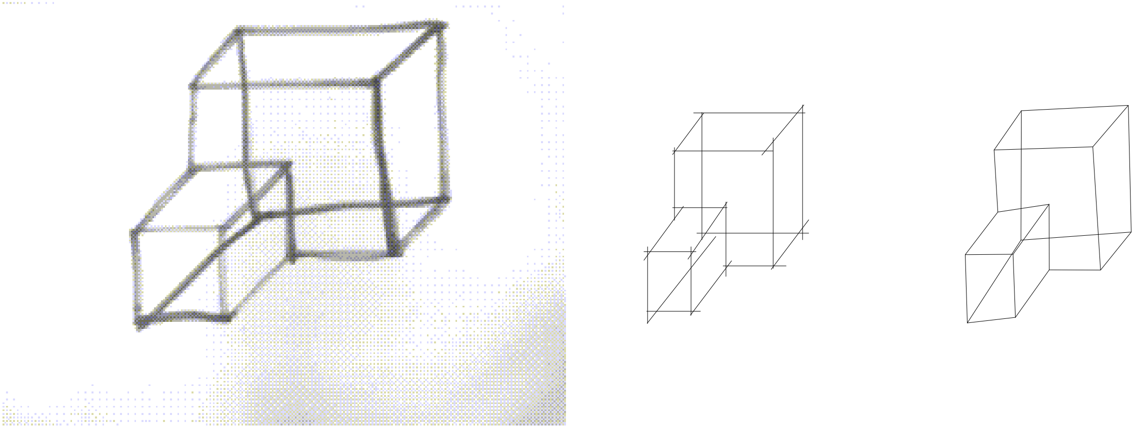


Two views of the recovered 3D lines, and the corresponding surfaces.

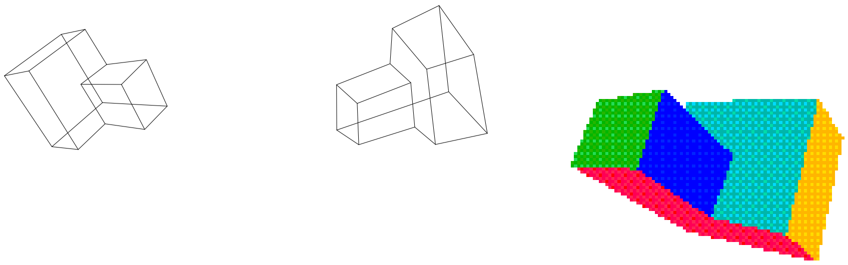
Figure 5: *A sketch reconstruction of a similar polyhedron in which the intersection-defined vertex of Figure reffig:Failhas been replaced with two end-defined vertices.*

Alternatively, one could imaging always projecting non-peak orientations onto the *same* two peak orientations. This mapping rule will result in a consistent grammar which can reconstruct a subset of generalized cylinders (Binford, 1971), in which an arbitrary two-dimensional shape (in this case a shape without lines of the third peak orientation) is swept along the direction in three space which is mapped to the third peak orientation. However, reconstructions based on this grammar would be extremely sensitive to slight perturbations of lines which are supposed to be of the third peak orientation.

To compensate for the distributed orientations resulting from the vertex-clustered drawings, an alternative method was implemented. By rotating each line to be oriented along the closest peak orientation we are insured that any closed path in the two-dimensional drawing will map to a closed path in three dimensions. The effect



Original image, initial line recovery, and lines after 2D clustering



Two views of the recovered 3D lines, and the corresponding surfaces.

Figure 6: *A reconstruction of an object with surfaces at three levels in each dimension.*

of rotating each line is to re-break the connected lines which were acquired in the clustering operation. However, the breaks which this introduces are smaller than the original breaks in the perturbed drawing because, after the initial clustering operation, the line centers – which do not change when the line is rotated – are closer to their intended locations.

To re-acquire endpoints which cluster to form vertices without changing the orientation of the lines, the iterative clustering operation was repeated while constraining the the motion of each line so that it retained the same orientation – at the expense of allowing its length to vary.

For the ideal drawings which were randomly perturbed, this method is able to successfully recover the topological structure of the intended three-dimensional image. Because line lengths were allowed to vary in order to simultaneously retain connectedness and orientation, the relative distances recovered become less accurate with increased perturbation. Eventually, as noise level increases, the recovered lengths degenerate to zero, and three-dimensional recovery fails.

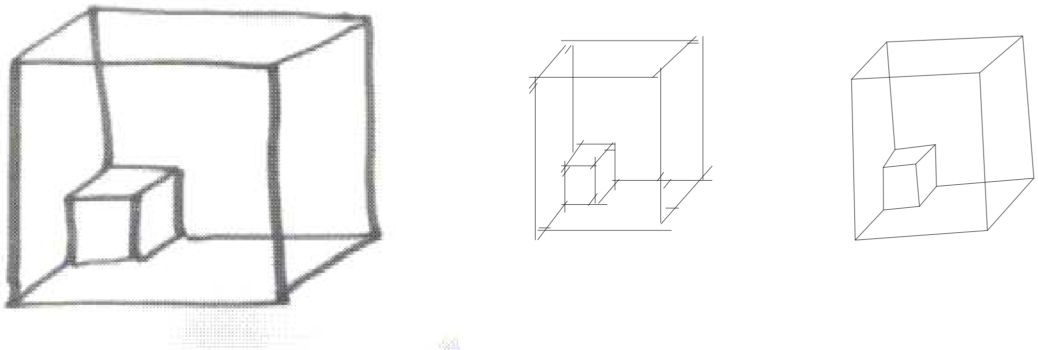
Interpreting Hand-Drawn Sketches

The next phase of the project was to apply this reconstruction technique to line drawings acquired from hand-drawn sketches.

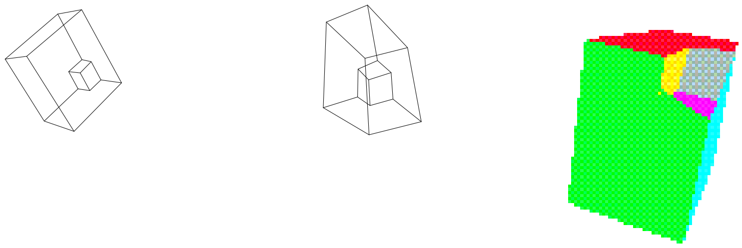
Hand-drawn wireframe sketches were acquired using a black and white CCD camera attached to a Silicon Graphics Indy computer. Examples of the input images are shown in the first panel of each figure. Line segments in the captured images were found with an algorithm which is equivalent to a multi-scale oriented filtering. To reduce computational cost, only orientations near the peak orientations of the grammar were found. Lines of with less than a specified threshold length were discarded, and fragmented lines were minimized by initially swelling the thresholded image before performing the line search. The line segments found are shown in the second panel of each figure.

For carefully drawn sketches the system as currently described was able to reconstruct the intended three-dimensional structure. However, when reconstructions of more haphazard drawings were attempted the failure rate rose rapidly. This was due to the fact that the deviation of the orientations of lines acquired from actual hand-drawn sketches from the peak orientations tended to be systematic, as opposed to uniformly distributed as they were in jittered ideal drawings. For example, a cube is often sketched with one face slightly smaller than the face parallel to it, resulting in every angle of the four adjacent faces deviating slightly. Because of this, the second phase of orientation-constrained iterative endpoint clustering results in line segments with lengths which frequently degenerate to zero. This sensitivity to these systematic orientation deviations is inherent in this clustering process, and as a result this method was abandoned in favor of another, more robust technique.

Even though most hand-drawn wireframe sketches of rectangular polyhedral ob-



Original image, initial line recovery, and lines after 2D clustering

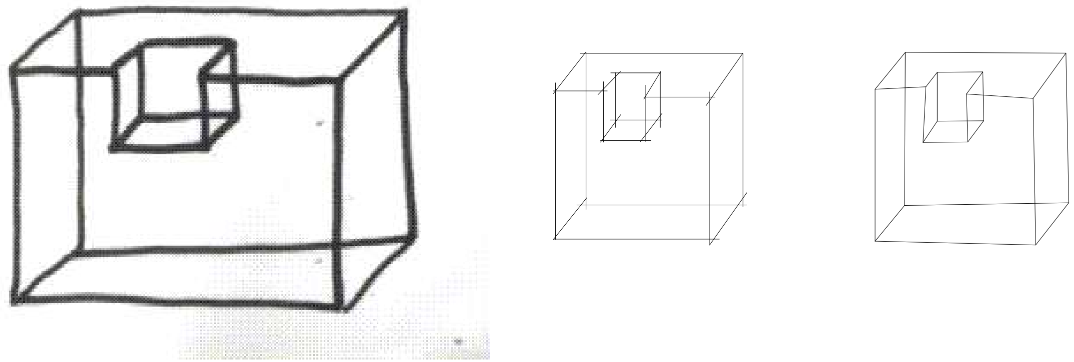


Two views of the recovered 3D lines, and the corresponding surfaces.

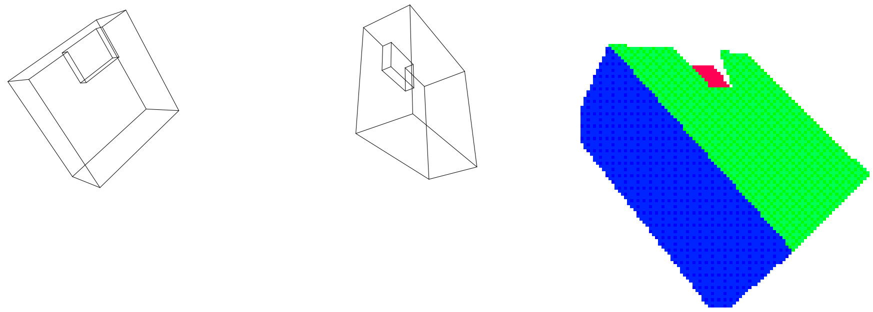
Figure 7: *The first complex polyhedron devised by friends in the AI lab to challenge the system.*

jects are globally inconsistent, they can be interpreted based on the locally consistent information. This tolerance of a global inconsistency was achieved by relaxing the constraint that the recovered polyhedron be truly rectangular.

This was done using three-dimensional version of the initial unconstrained clustering method used in two dimensions. By beginning at a vertex and recursively mapping each line as to a line in three space by interpreting it as if it actually were of the peak orientation to which it is closest, we recover a partially connected set of lines in three space. Even though the resulting set of lines is only partially connected, the situation is far better than the fragmentation which the iterative clustering corrected in two space for three reasons: each line segment is connected to at least one other segment; adding an additional dimension increases the distance between disconnected vertices; and all spurious and fractured lines have already been eliminated. Hence,



Original image, initial line recovery, and lines after 2D clustering



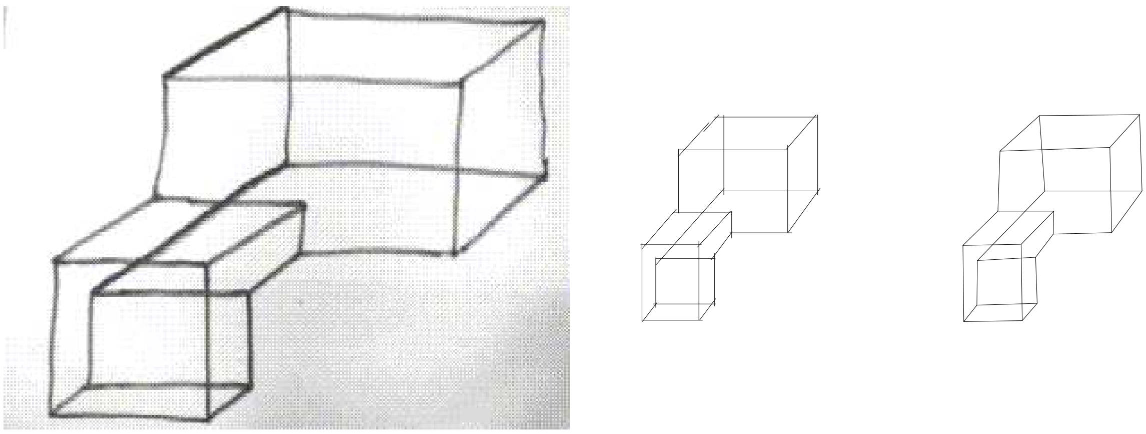
Two views of the recovered 3D lines, and the corresponding surfaces.

Figure 8: *The reconstruction of a another polyhedron with the same "notched-cube" theme as in Figure reffig:CornerNotch.*

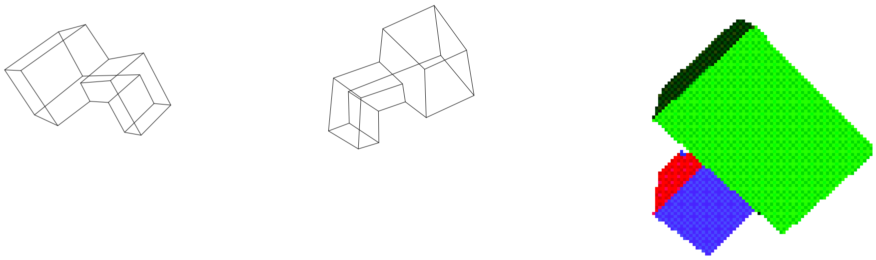
the iterative grouping operation in three-space is able to successfully regroup line segment endpoints into vertices.

Though resulting polyhedral wireframe is not exactly rectangular (except in the case of an ideal input drawing), line lengths are roughly proportional to the line lengths indicated in the drawing. For example see Figures 9 and 6. Further, the angles between lines which meet vertices does not deviate greatly from 90 degrees, resulting in a wireframe which is locally consistent with a rectangular polyhedron. Examples of these three-dimensional line reconstructions, from two points of view, are shown in the fourth and fifth panel of each figure.

Surfaces can be reconstructed from (perfect) rectangular polyhedra by iteratively tracing all of the closed paths from some point lines orthogonal to one of the orthonormal basis vectors, until all the lines have been tested for membership in a



Original image, initial line recovery, and lines after 2D clustering

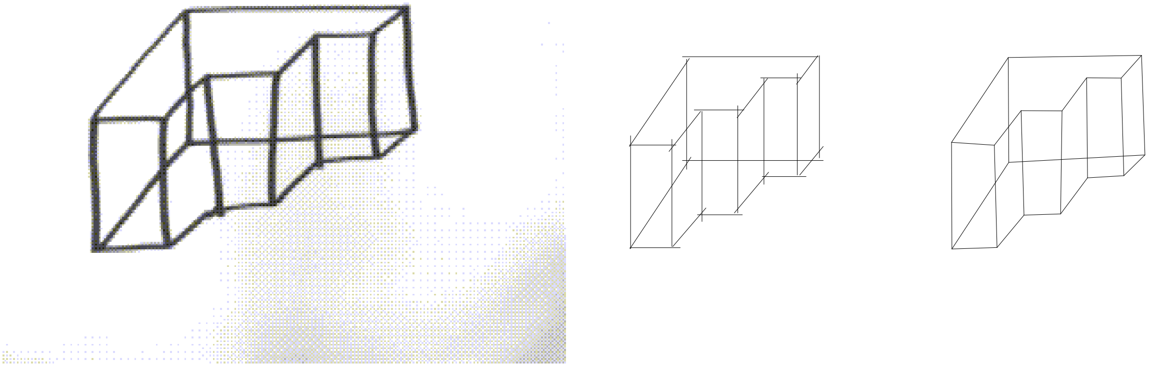


Two views of the recovered 3D lines, and the corresponding surfaces.

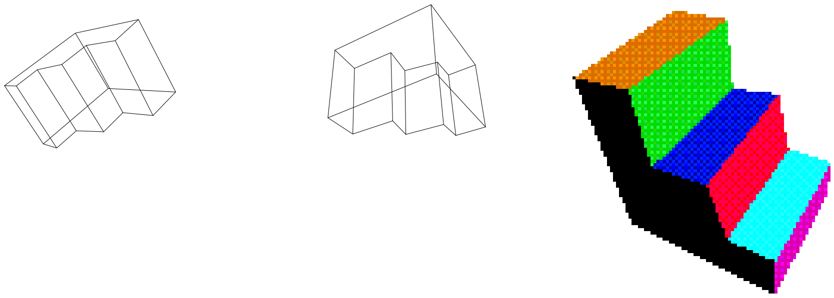
Figure 9: *By relaxing angular constraint of insisting on orthogonality, the system is able to maintain the relative distances indicated in the original sketch.*

surface orthogonal to each vector. Reconstructing surfaces from polyhedra which are only approximately rectangular, was done by only enforcing that the angle between each line in the closed path and a basis vector be less than 45 degrees.

Presumably in many applications of such a system, we want to assume that the polyhedron depicted in the drawing is *supposed* to be rectangular. Therefore it is useful to determine what the rectangular polyhedral shape would be if the sketch had been drawn perfectly. The closest rectangular polyhedron can be easily found by "snapping" the acquired approximately-rectangular polyhedron to a grid by quantizing the line segment endpoints (assuming that the sketch only deviates by some reasonable amount from a projection of a rectangular polyhedral shape).



Original image, initial line recovery, and lines after 2D clustering



Two views of the recovered 3D lines, and the corresponding surfaces.

Figure 10: *Reconstruction of sketches consisting of haphazardly drawn "right" angles is made possible by allowing the recovered 3D lines to deviate from a true rectangular polyhedron.*

Multiple Polyhedra

Because wireframe objects cannot occlude one another, except cases in which lines or vertices are accidentally aligned, there is no additional difficulty associated with multiple polyhedra. Once all of the closed paths beginning from a point have been mapped to three-space, all of the remaining lines which have not yet been mapped belong to different polyhedra. We simply recursively apply the same reconstruction procedure until all lines have been used.

There is no way to determine the relative depth of multiple polyhedra from a sketch, unless their spatial confiuration is made explict by intigrating them into a single wireframe as in Figure 11. Therefore initial depth of the first point is arbitrarily chosen to be zero (though any value could be used); because of this a sketch which contains two overlapping wireframes will result in recovered three-dimensional polyhedra which occupy the same space – this can only be avoided by the application of an arbitrary rule which simply displaces the two polyhedra in depth.

System Limitations and Recovery Failure

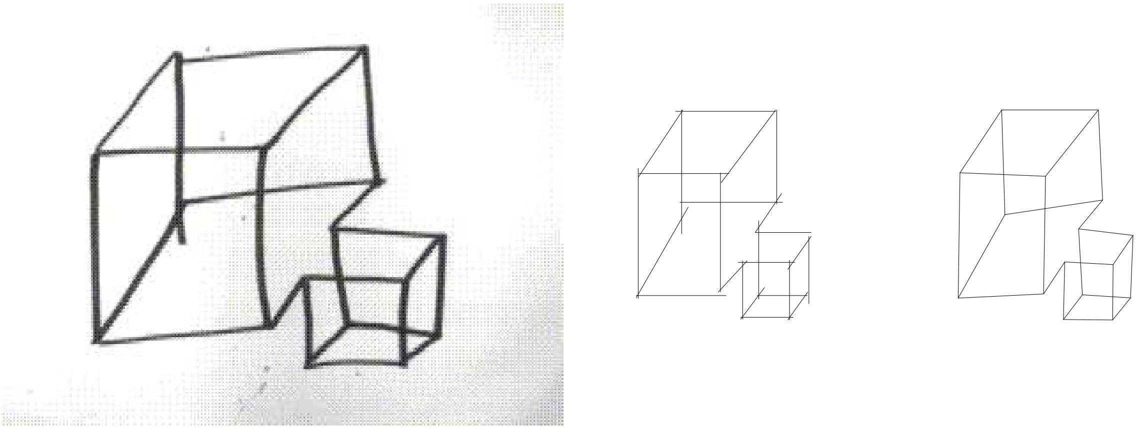
Though this system can successfully recover three-dimensional structure from sketches, it requires human intervention to set parameters about the environment and it is prone to certain problems which cause recovery to fail.

- Spurious lines are reduced by throwing out lines below a threshold length; determining this threshold is a difficult problem when nothing about the thickness and density of the lines in the input sketch are known. Currently this threshold is a system parameter.

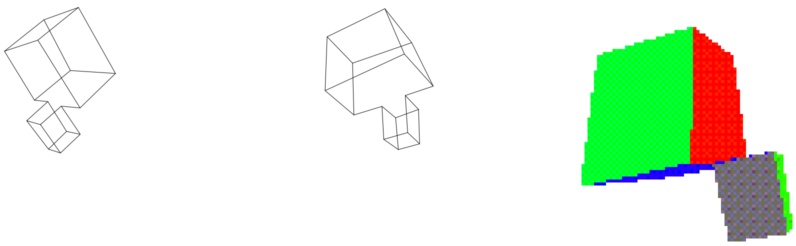
- The gain control level used to determine which image pixels are part of lines is also a system parameter.

- Though this system is quite robust to spurious and somewhat fractured lines, the clustering operation will fail completely if there are any *missing lines*.

- "The accidental alignment in which we happen to be viewing a vertex from a direction that makes it line up with a more distant edge" (Horn, 1986) will cause the recovery of only one line, which will result in failure of the clustering operation.



Original image, initial line recovery, and lines after 2D clustering



Two views of the recovered 3D lines, and the corresponding surfaces.

Figure 11: *Unless explicitly defined by attachment, as in the above example, the system will simply reconstruct each polyhedron in the scene at the x and y positions indicated in the drawing and at a depth of 0.*

References

Ballard, D.H., and C.M. Brown, *Computer Vision*, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1982.

Binford, T.O. (1971) "Visual perception by computer," *Proceedings of the IEEE Systems Science and Cyberntic Conference*, Miami, December.

Clowes, M.B. (1971) "On Seeing Things," *Artificial Intelligence*, Vol. 2, No. 1, pp. 79-116, Spring.

Horn, B.K.P. (1977) "Image Intensity Understanding," *Artificial Intelligence*, Vol. 8, No. 2, pp. 201-231, April.

Horn, B.K.P., *Robot Vision*, MIT Press, Cambridge, Massachusetts, 1986.

Huffman, D.A. (1971) "Impossible Objects as Nonsense Sentences," in *Machince Intelligence 6*, B. Meltzer, and D.M. Michie (eds.), Edinburgh University Press, pp. 295-323.

Mackworth, A.K. (1973) "Interpreting pictures of polyhedral scenes," *Artificial Intelligence*, Vol. 4, No. 2, pp. 121-137, Summer.