

Identification of Faces in a 2D Line Drawing Projection of a Wireframe Object

M. Shpitalni and H. Lipson

*Laboratory for Computer Graphics and CAD, Faculty of Mechanical Engineering,
Technion, Haifa 32000, Israel*

Abstract - An important key to reconstructing a three-dimensional object depicted by a two-dimensional line drawing projection is face identification. Identification of edge circuits in a 2D projection corresponding to actual faces of a 3D object becomes complex when the projected object is in wireframe representation. This representation is commonly encountered in drawings made during the conceptual design stage of mechanical parts. When non-manifold objects are considered, the situation becomes even more complex. This paper discusses the principles underlying face identification and presents an algorithm capable of performing this identification. Face-edge-vertex relationships applicable to non-manifold objects are also proposed. Examples from a working implementation are given.

Index Terms - Line drawing interpretation, face identification, line labeling, 3D object reconstruction, non-manifold geometry, image understanding.

1. INTRODUCTION

Restoration of three-dimensional information from a single image is required by many vision-based applications. One such application is the reconstruction of a three-dimensional object from a single projection given in the form of a line drawing. An important key to understanding the line drawing and accelerating the reconstruction process is the initial phase of analysis and identification of the edge circuits corresponding to the actual faces comprising the 3D object. Those circuits should be identified and distinguished from the many other existing circuits in the drawing. This identification phase is complex when the object is displayed in wireframe representation, so that all of its edges appear in the projection. Such cases are common in the initial conceptual design stage of mechanical parts. Identification becomes even more complex when general objects including non-manifolds are considered. This type of line drawing input has been used in a new user interface for conceptual CAD proposed by the authors [1, 2].

In this paper, a method is proposed for identifying the relevant faces in a 2D projection of a general (manifold or non-manifold) object. In Fig. 1, for example, the original input consisting of two-dimensional edge-vertex graphs appears on the left. On the right, these graphs have been broken down (by the proposed algorithm) into the edge-circuits corresponding to the actual faces of the depicted 3D object. It is important to note that both the graphs on the left and those on the right are all two-dimensional and that no information was available regarding the three-dimensional objects they represent.

This work strives to simplify reconstruction of an object from its 2D wireframe projection by analyzing the 2D projection. This analysis significantly reduces the number of degrees of freedom in the reconstruction phase. Since depth information is missing, a line drawing representing the projection of a three-dimensional object does not correspond uniquely to a

single object. Each vertex can be moved freely along an axis perpendicular to the drawing plane, giving rise to an infinite number of possible 3D line configurations corresponding to various objects. A projection of a wireframe box, for instance, has eight vertices; therefore, the reconstruction problem has eight degrees of freedom related to the unknown depth of the vertices. Had the vertices been classified into the six box faces, then six relationships requiring planarity could be formulated, thus reducing the number of degrees of freedom from eight to two. Because the depth of one vertex may arbitrarily be set at 0, the number of degrees of freedom is reduced in practice to 1. Face identification can thus be used to significantly reduce the complexity of the reconstruction problem.

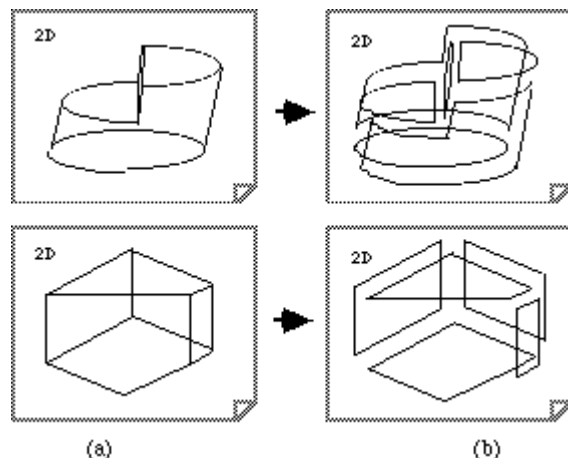


Fig 1. (a) The given 2D projection of some object, and (b) the identified projection of the faces in the 2D projection plane.

In this paper, the problem of face identification is defined. Next, related work regarding analysis of line drawings is reviewed. Two approaches to the solution are then proposed. The first is applicable for a specific subset of manifold objects. The second approach, derived from general vertex-edge-face relationships, is applicable for general objects including both manifold and non-manifold geometries with planar and curved surfaces. Heuristic rules are applied to reduce the problem search space to a manageable size as well as to select among alternative solutions. In the course of the mathematical development, new face-edge-vertex relationships for non-manifold objects are proposed. Finally, examples of a working implementation are given. A short glossary of topology and graph-theory terms is provided in appendix A. The reader is also referred to references [3] and [4] for a more elaborate definition and discussion of the terms used in this paper.

2. PROBLEM AND APPROACH

In general, the problem can be defined as follows: given a single, two-dimensional line drawing representing a projection of a three-dimensional wireframe model, it is required to identify those edge circuits that correspond to the actual faces comprising the object that would most likely be understood by a human observer (most plausible solution).

However, further clarification is needed to define the problem more precisely, especially regarding the assumptions made.

- * The input to the system consists only of the single two-dimensional line drawing which is entered as a graph of connected entities. (This graph may be the result of previous processing of a rough input sketch, see [1, 5]). Each edge of the graph corresponds to exactly one edge of the depicted object. Each vertex of the graph corresponds to exactly one vertex of the depicted object.
- * The input projection represents a wireframe model of a general object that may be manifold, non-manifold or an assembly of both types of objects. No information is provided to the system about the three-dimensional object itself, its type, or its position relative to the viewpoint.
- * The projection is drawn from a general - non-accidental - viewpoint that reveals all edges and vertices. That is, none of the object's edges or vertices coincide accidentally, and none of them accidentally appear as joined in the projection. This assumption also ensures that the topology of the projected edge-vertex graph will not change if the viewpoint is perturbed slightly within its neighborhood.
- * All drawn lines and curves in the projection represent real edges, silhouette curves (refer to Fig. 2) or intersections of faces in the 3D object. No shadow lines or surface marks are allowed.

2.1 Overview of approach

Following is a brief description of the overall approach of the proposed algorithm for identifying faces of objects in their projection. Input to the algorithm is an edge-vertex graph. This graph typically originates from a raw preprocessed source (e.g., a raster image or a rough sketch), as described briefly in the next section. Once the graph is obtained, there are two ways of proceeding.

If the object depicted in the graph is known a priori to be a manifold of genus 0, a simple analytic approach can be taken. The graph is brought to a planar configuration in which the easily identifiable faces of the graph correspond uniquely to the faces of the depicted object. An exception to this rule is a case when the graph is less than three-vertex-connected. In this case, various alternative configurations may arise. This case is discussed, and a heuristic technique for selecting the most plausible solution is provided.

In the other case, where no a priori knowledge about the depicted object is available (a general object type is presumed), the algorithm proceeds as follows. After the input data has been transformed into a graph form, as discussed below, the algorithm generates a pool of various edge circuits which potentially correspond to faces of the depicted object. Various considerations in generating this pool are discussed. Then, the algorithm tries to match a selection of these potential faces in a way that complies with some basic geometrical and topological constraints. The constraints and the matching process are discussed in detail. Here again, several alternative solutions may be found, and the same technique for selecting the most plausible solution is applied.

2.2 Converting a rough input sketch to an edge-vertex graph

Since the input to the application (in our case) is in the form of a rough freehand sketch, some means of converting this sketch into an edge-vertex graph are required. Although this stage is

not the main issue of this paper, a brief description is provided for completeness. A full and detailed description of this stage can be found in [5].

The sketch is entered by means of on-line sketching onto a graphical device. Each sketch stroke is assumed to correspond either to a line, to an elliptic arc or to a corner. Strokes are classified by fitting each one to a conic section equation which can take the form of any of these three entity types (the corner entity corresponds to a hyperbola). Once this classification is complete, the scattered entities are linked at their endpoints to form an edge-vertex graph. Linking is achieved by clustering entity endpoints according to various adaptive criteria, based mainly on the amount of details and properties of the sketch in the close vicinity of the endpoint. The sketches shown in Table 1 were analyzed using this technique before the actual algorithms described in this paper were applied.

2.3 Faces and circuits

In a projection from a non-accidental viewpoint, the edges of a 3D face form a closed circuit of entities. Boundaries of planar faces are projected into non-self-intersecting circuits, while boundaries of non-planar faces are projected into one or more non-self-intersecting circuits divided by the projection of silhouette curves, as is illustrated in Fig. 2. Thus, boundaries of all faces of a general 3D object are projected into a set of closed, non-self-intersecting circuits of entities. Note, however, that not every non-self-intersecting circuit of entities in the projection necessarily corresponds to an actual face of the 3D object. See, for instance, the circuit highlighted in the projection of a block in Fig. 3(b). In fact, the vast majority of the non-self-intersecting circuits do not correspond to actual faces. Therefore, the circuits corresponding to actual faces must be distinguished from those which do not.

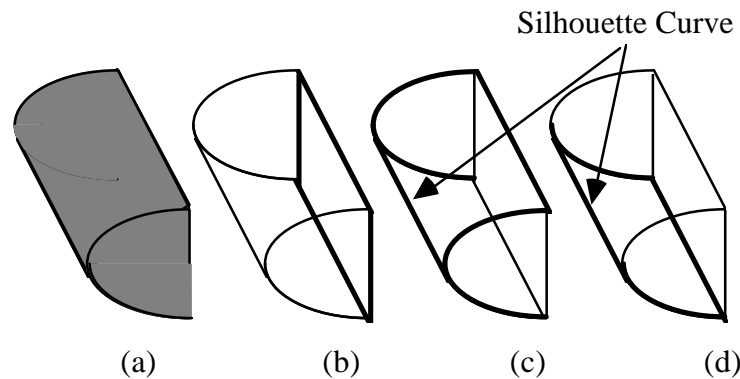


Fig 2. (a) The original 3D object; (b) planar faces are projected into circuits; (c) and (d) non-planar faces are projected into a group of circuits divided by silhouette curves.

Since a single projected graph does not necessarily correspond to unique 3D topology for a general object, several feasible alternatives for face configurations may result. The algorithm should choose the most plausible one, that is, the solution that would most likely be selected by a human observer.

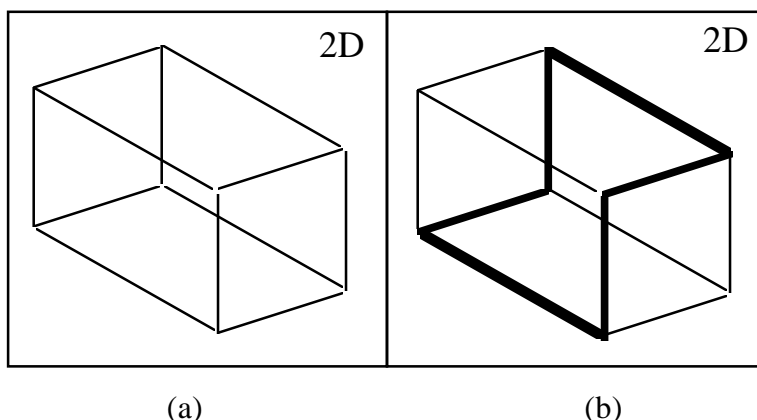


Fig 3 (a) A projection of some object; (b) can the non-self-intersecting highlighted circuit be a face of the depicted object ?

The problem can now be stated as follows: Given is a single graph representing a projection of a 3D wireframe model as viewed from a non-accidental viewpoint. It is required to generate a list of circuits of the graph corresponding to the actual faces of the most plausible 3D object depicted by the graph. Each such face must be defined by an ordered set of edges comprising its contour.

The proposed approach is based upon identifying those circuits corresponding to actual faces of the 3D object from all the possible circuits in the two-dimensional edge-vertex graph. In the following sections, a general identification method is proposed. The proposed method is capable of dealing with graphs depicting general geometries of manifold and non-manifold objects, possibly with a non-zero genus. A simpler method is also proposed for cases in which it is known in advance that the object represented in the line drawing is a manifold object of genus zero.

3. RELATED WORK

Interpretation of 2D drawings as 3D topologies is related to image-analysis and computational geometry research. Most of the work has concentrated on analyzing object drawings with hidden lines removed and is often termed 'line labeling.' In such drawings, the projection is by definition already a *plane graph*, and the task of face (or partially occluded face) identification is reduced to detecting edge circuits enclosing the graph faces. Moreover, the objects considered are typically confined to plane faced objects from the trihedral or Origami world, for which simpler line-labeling procedures can be applied.

Single projection line drawing interpretation has been treated qualitatively by Huffman [6] and Clowes [7] and quantitatively using line labeling techniques by, for example, Kanade [8] and Sugihara [9]. According to these techniques, an edge in a drawing can represent one of three states: concave face connection, convex face connection, or occluding-face edge. A consistent set of line labels is searched for, using a library of possible line junction configurations. Many such sets may be found. Marti et al [10] analyzed drawings with hidden edges represented by dashed lines, using a line labeling method with an expanded junction library. The expanded library produced a very large number of possible solutions and relied on the line font (dashed/solid) for extracting spatial information. This attribute is not available in our input.

Another approach to face identification, proposed by Leclerc et al [11], searches for all non-self-intersecting closed circuits of entities. Faces are identified when a circuit is free of internal lines and vertices or when it forms a convex shape and is free of any internal circuits, except those ending on two non-adjacent vertices of the convex circuit. This approach is easy to implement but operates only for a limited range of objects. It fails if the object contains concave faces or holes. In addition, it does not deal with ambiguities and cases with more than one solution.

4. FACE IDENTIFICATION METHOD FOR MANIFOLD OBJECTS OF GENUS ZERO

Euler-Poincare's formula for all 2-manifolds [3] states that

$$v - e + f = 2 (s - g) \tag{1}$$

where v , e and f respectively represent the number of vertices, edges and faces of an object, s represents the number of discrete objects, and g represents the sum of the genres of each object. For a manifold object of genus 0, $g=0$, and the number of objects s can be uniquely determined by counting the separable *components* of the graph. Thus, the number of faces can be easily computed after counting the vertices and edges in the given edge-vertex graph. In a manifold object, every edge belongs to exactly two faces. Therefore, the face suggested in Fig. 3(b) is ruled out because had it been selected, not enough edges would be left to account for the remaining five faces, as required by Euler's formula.

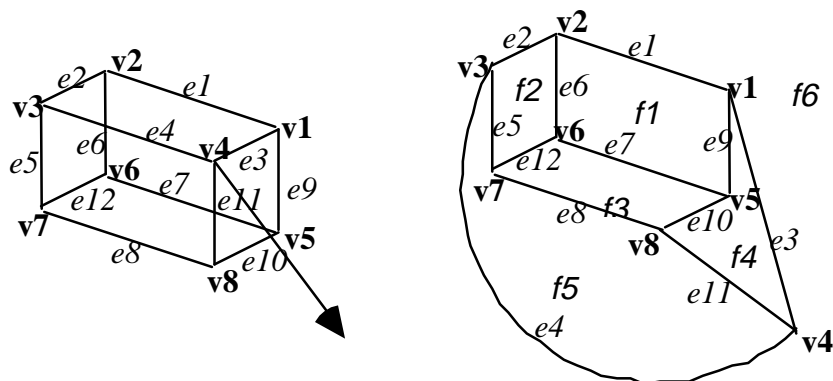


Fig 4. (a) Original edge-vertex graph representing a block, and (b) the corresponding equivalent planar graph with faces identified.

While the Euler-Poincare formula specifies the *number* of faces comprising the object depicted by the graph, it cannot identify the *actual faces*. These can be identified using a planar configuration of the graph. The correspondence between the planar graph and the topology of the 3D object is termed a *plane model*, a labeled plane figure whose edges and vertices correspond to those of the real 3D object. A formal definition of plane models is presented in [3]. The given edge-vertex graph is, in fact, a plane model of the object it represents. Faces are identifiable when the graph is re-drawn with no intersection between edges, i.e., when the input drawing is embedded as a *plane graph*. At this stage, all the faces of the graph (minimal, empty and closed circuits of edges) correspond to faces of the object represented by the current plane model. In addition, the *exterior* face of the plane graph defined by the outermost circuit of entities also corresponds to a face of the object. Fig. 4 illustrates this face identification process

for a wireframe block. The given edge-vertex graph, Fig. 4(a), consists of 8 vertices ($v1..v8$) and 12 edges ($e1..e12$). For face identification, vertex $v4$ and its connected edges $e3$, $e4$ and $e11$ are moved so that the graph contains no intersections. In the resulting graph, shown in Fig. 4(b), faces $f1..f6$ can be easily detected, with $f6$ the exterior face.

An examination of the edges surrounding each face in Fig. 4(b) reveals that

$$\begin{aligned} f1 &= \{e1, e6, e7, e9\} & f2 &= \{e2, e5, e12, e6\} \\ f3 &= \{e7, e12, e8, e10\} & f4 &= \{e3, e9, e10, e11\} \\ f5 &= \{e4, e5, e8, e11\} & f6 &= \{e1, e2, e4, e3\} \end{aligned}$$

and thus a solution has been obtained.

Two issues limit the applicability of this method: existence and uniqueness. The first issue is whether a plane embedding *exists*, or, in other words, whether the given graph is *planar*. Since any manifold object of genus 0 is homotopically equivalent to a sphere (i.e., can be deformed continuously to a sphere) [3], and a graph on a sphere can be embedded in a plane with one exterior face, the *plane model* of such an object is always planar. The same does not apply for objects of a higher genus nor for non-manifold objects. Therefore, the proposed method is not suitable for these types of objects.

The second issue concerns uniqueness: whether the face-circuit relationship in the given planar graph is unique and independent of the graph embedding. Whitney [12] asserts that the embedding of a planar, 3-vertex-connected graph G into a plane (or a 2-sphere) is topologically unique. Thus, the set of faces of G is well defined. According to Steinitz's theorem [13], a graph which is 3-vertex-connected and planar is isomorphic to the edge graph of a 3-dimensional convex polytope. Since a 3-dimensional convex polytope is isomorphic to any 2-manifold of genus 0, it can be concluded that the faces of a manifold object of genus 0 can be uniquely identified in its projected edge-vertex graph (which is always planar) as long as the graph is 3-vertex-connected.

If the graph is planar but not 3-vertex-connected, several different embeddings corresponding to different face configurations can be found. The number of such embeddings is exponentially proportional to the number of components in the decomposition of the given graph into 3-vertex-connected components that contain circuits. The relationship is exponential because each component may be individually flipped while the planarity of the graph is retained, thus giving rise to many permutations (provided that the component contains at least one circuit). However, the interior faces of each 3-vertex-connected component are well defined and unchanged by the flipping process. Therefore, they can be accepted without further processing. In a further step, the most plausible embedding should be selected.

4.1 Examples

In this section, examples of face identification for manifold objects of genus zero are discussed.

Example 1: Consider again the graph in Fig. 3(a). This graph is both planar and 3-vertex-connected and therefore has unique face topology. The circuit highlighted in Fig. 3(b) does not enclose a single face when the graph is brought to a plane embedding; therefore, it cannot correspond to a face of the depicted object. The unique face topology has been determined, as

shown in Fig. 4(b). Note that the faces were identified without assuming any knowledge about the object depicted by the graph.

Example 2: Consider the graph shown in Fig. 5(a) which represents the projection graph of some polyhedron. Without prior knowledge about the object depicted by the graph, it is evident that the graph is not 3-vertex-connected: the graph can be separated into two components if it is disconnected at two vertices, one on each of the two highlighted edges. Therefore, the graph can be brought to two plane embeddings. The first planar configuration is shown in Fig. 5(b) with the inner component enclosed in a dashed rectangle. The second planar configuration is shown in Fig. 5(c), which was created by flipping the inner component of the previous configuration. The two corresponding object topologies are illustrated respectively in Fig. 5(d), the subtraction of a small block from a large block, and in Fig. 5(e), the union of a large block with a small block. Note that other *non-manifold* interpretations can be given to these pictures; however, only those representing manifold objects are relevant here.

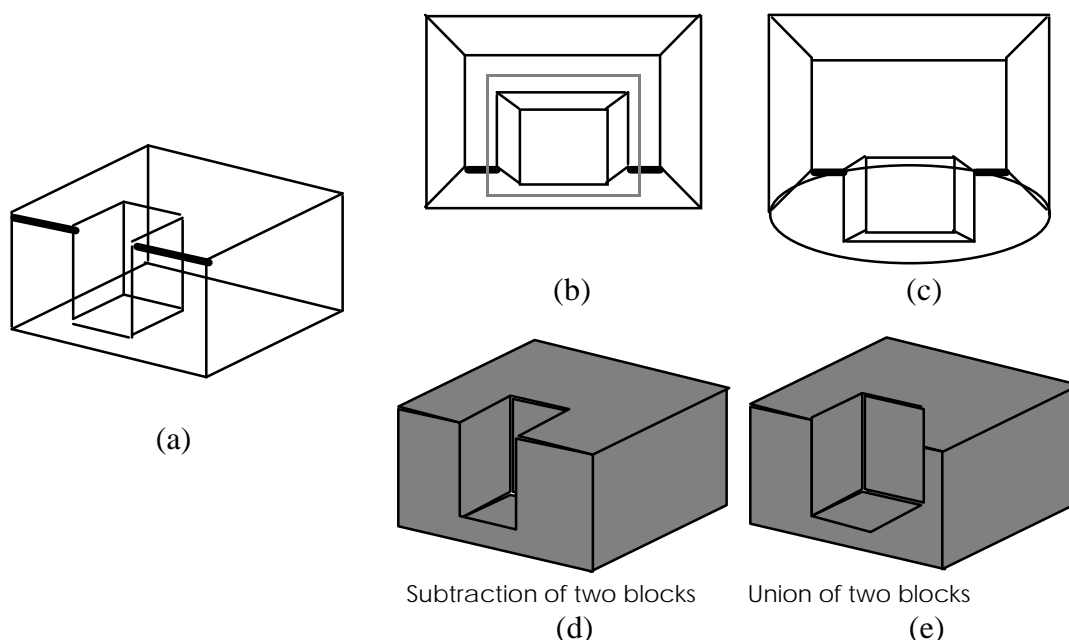


Fig 5 (a) Given edge-vertex graph; (b) first planar configuration; (c) second planar configuration; (d,e) the two resulting 3D objects.

Example 3: A more elaborate example is shown in Fig. 6. The graph in Fig. 6(a) is the projection of a wireframe cylinder with two silhouette curves. Note that this object has non-planar faces. Without any prior knowledge about the depicted object, we notice that the planar graph may be decomposed into two components by disconnecting it, for example, at vertices $v1$ and $v2$; therefore it is *not* 3-vertex-connected. The graph can be re-embedded with two different configurations, as shown in Fig. 6(b) and (c), where the subsequent embedding is created by flipping over one of the separated components. Any other embedding is isomorphic to one of these two configurations. The circuits enclosing the four faces of each graph are listed. In both embeddings, $f1=\{e1,e2\}$ and $f3=\{e5,e6\}$, because these faces are interior to the 3-vertex-connected components. However, in Fig. 6(b), $f2=\{e2,e3,e5,e4\}$ and $f4=\{e1,e3,e6,e4\}$, whereas in the configuration shown in Fig. 6(c), where the component containing the edges $e1$ and $e2$ has

been flipped over, faces f_2 and f_4 now correspond to different circuits, namely, $f_2=\{e_1,e_3,e_5,e_4\}$ and $f_4=\{e_2,e_3,e_6,e_4\}$. Once the face-circuit relationships in both plane configurations have been listed, the graph in Fig. 6(a) can be disassembled into its faces. The resulting faces corresponding to the two embeddings are shown below the graphs. Any other embedding of the graph will have the topology and face-circuit relationship of either of the above two.

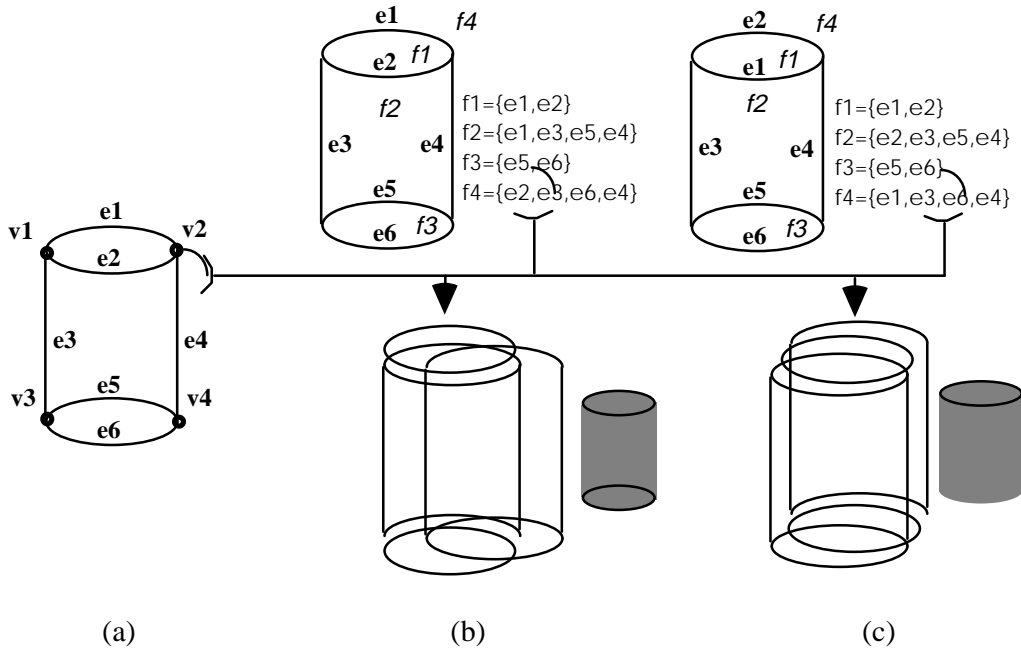


Fig 6 (a) A graph representing a projection of a cylinder; (b) the first plane embedding, with corresponding face list and interpretation; and (c) the second.

4.2 Summary of the approach for manifold with genus-zero

The proposed face identification method is summarized below. The method is applicable when the depicted object is known a priori to be a 2-manifold object of genus zero.

- *Assuming the graph represents a manifold object of genus zero, bring it to a plane embedding.*
- *Check whether the graph is 3-vertex-connected (see [4]).*
- *If the graph is 3-vertex-connected, the faces of the graph are the faces of the depicted object. End.*
- *If the graph is not 3-vertex-connected, decompose the graph into 3-vertex-connected components containing circuits (see [4]).*
- *The interior faces of each component are faces of the depicted object.*
- *Find all permutations of plane embeddings created by flipping 3-vertex-connected components.*

- *The set of faces identified in each embedding of the graph corresponds to a specific interpretation.*
- *Select a single (most plausible) interpretation using heuristics, as described in section 5.7 below. End.*

For manifold objects of a non-zero genus or for objects that are not manifold at all, a different approach must be taken.

5. IDENTIFICATION METHOD FOR GENERAL OBJECTS

The case of non-manifold objects is much more complex. Sometimes it is impossible to find a plane model configuration with no edge intersections (see Fig. 7(a)). That is, the face-edge-vertex graph is not *planar*. In other instances, several plane embeddings can be found, each representing a different solution (see Fig. 7(b)), and none necessarily corresponding to faces of the plausible object. Consequently, a plane embedding for edge-vertex graphs of non-manifold objects is either nonexistent or not necessarily unique. A different approach should therefore be taken, one which will also permit selection of the most plausible face configuration.

The term *general object* refers to an object of an unknown type. This object can be either manifold or non-manifold and is composed of solids, surfaces and skeletal structures with or without holes and perhaps with several detached components. This generality precludes defining a unique mathematical solution. The goal is to find the most plausible solution that would be identified by a human observer. Heuristics must therefore be introduced into the solution method to account for psychological reasoning in picture understanding.

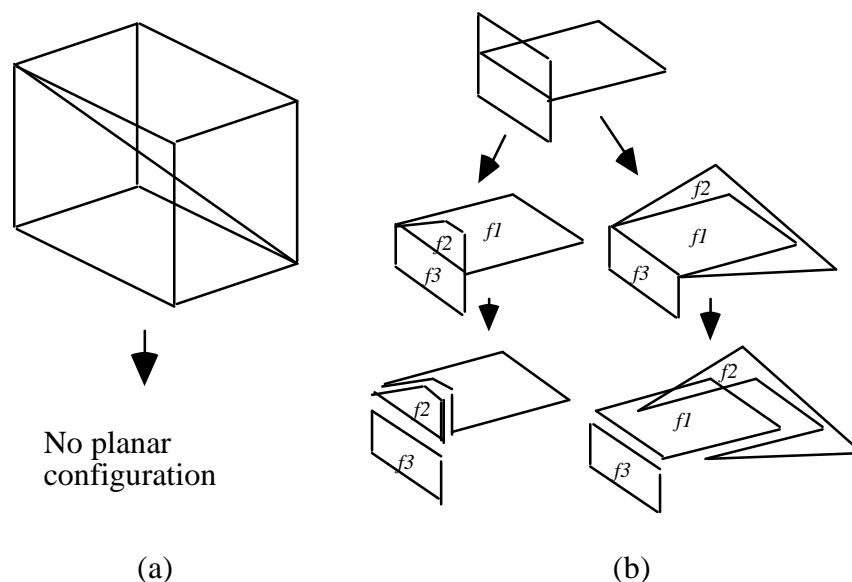


Fig. 7. Plane models of **non-manifold objects**: (a) a non-intersection configuration cannot be found for a wireframe box with a diagonal link; (b) different conflicting planar embeddings can be found for a T-shaped object.

5.1 Potential faces and circuit space

A general object with n vertices has $O(n)$ edges and $O(n)$ faces. Its projection has m non-self-intersecting circuits of edge-entities, where m is an exponential function of n . We shall term these circuits as *potential faces* because the faces of the object are a subset of k circuits, where $0 < k < m$. Although there are exactly 2^m possible combinations of *potential faces*, human observers seem to be able to easily identify this subset. Fig. 8 shows an example of 15 potential faces in the projection of a wireframe block.

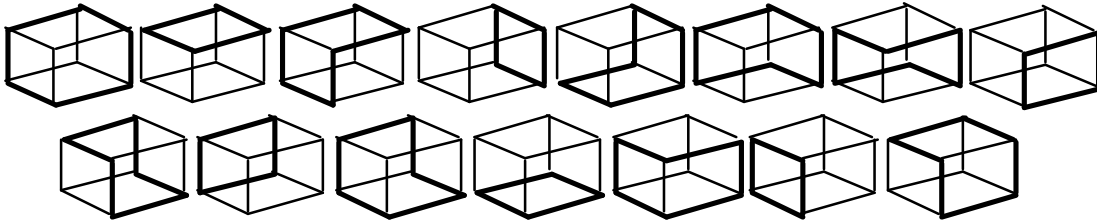


Fig 8. A projection of a wireframe block has 15 non-self-intersecting circuits of edge entities. Each is a potential face of the depicted object.

Potential faces can be identified using several approaches. One approach involves using the so-called *circuit space* [4]. The circuit space of a graph contains all the possible circuits embedded in the graph. This space is spanned by the *fundamental circuits* of the graph, which form the basis of this space. The addition of a chord to a spanning tree of a graph creates exactly one circuit. Thus, a collection of these circuits with respect to a particular spanning tree forms a set of fundamental circuits. An efficient $O(n^3)$ time algorithm for finding such a basis for a graph containing n vertices can be found in [4]. The full circuit space can then be constructed using integer combinations (ring sums) of the basis circuits.

5.2 Maximum rank equations

Identifying faces of an unknown general object in a given projection can be formulated as a selection problem, i.e., selecting k faces among the m potential faces such that the k faces represent a valid and most plausible object. Once the problem space has been defined, the search target must be defined as well as constraints for reducing search time and domain.

The Euler-Poincare equation (1) cannot provide any information about the number of faces expected in a general object because the object is not necessarily manifold. Therefore, more general relationships are required.

In formulating the identification method, the following notation is used:

- * The *Rank* $R(e)$ of an edge e denotes the number of faces whose boundary contains that edge.
- * The *Rank* $R(v)$ of a vertex v denotes the number of faces whose boundary contains that vertex.
- * The degree $d(v)$ of a vertex v denotes the number of edges meeting at that vertex.

The following analysis is based on the assumption that two different smooth surfaces cannot share two non-collinear edges meeting at a vertex. This assumption is based on the face adjacency theorem discussed in the next section. The specific case of two edges meeting collinearly at a vertex, for which this assumption does not hold, will be dealt with separately in a subsequent section.

Local analysis of a general edge e with endpoints at two vertices $v1$ and $v2$ reveals information regarding the upper bounds of the ranks (see Fig. 9).

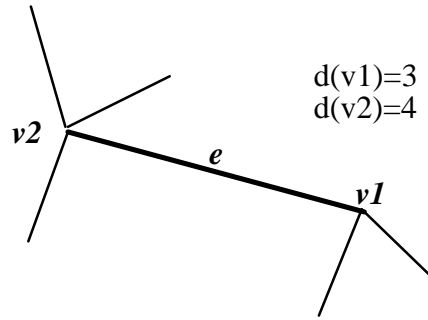


Fig 9. Local analysis of a general edge e in an edge-vertex graph, with its two endpoints at vertices $v1$ and $v2$.

Every face boundary passing through vertex v must also pass through two edges meeting at v . Thus, the number of faces passing through vertex v is limited by the number of possible edge pair permutations at that vertex. More specifically,

$$R(v) \cdot 1/2 [d(v) * (d(v) - 1)] \quad (2)$$

Similarly, since every face boundary passing through edge e must also pass through one of the other edges meeting at its endpoint, the maximum number of faces bounded by e is limited by the number of edges joining it at its endpoint. More specifically,

$$R(e) \cdot d(v1) - 1 \quad \text{and} \quad R(e) \cdot d(v2) - 1$$

or, more compactly,

$$R(e) \cdot \min [d(v1), d(v2)] - 1 \quad (3)$$

Once $R(e)$ has been determined, it is also evident that every face boundary passing through a vertex also passes through two of the edges meeting at that vertex, and thus,

$$R(v) = 1/2 \cdot R(e), \quad \text{for all edges meeting at vertex } v \quad (4)$$

Similarly, every face boundary passing through an edge e must also pass through both its endpoints, and thus

$$R(e) \cdot \min [R(v1), R(v2)] \quad (5)$$

In the following discussion, Equations (2), (3), (4) and (5) will be termed the **maximum rank** equations.

In a given edge-vertex graph, the degree $\mathbf{d}(\mathbf{v})$ of a vertex can easily be determined. This information can be used along with Equations (2) and (3) to compute a preliminary estimation of the upper bounds of the ranks of all vertices and edges in the graph. Next, relaxation is used to further adjust the upper bounds of the ranks. Equations (4) and (5) are applied iteratively until "equilibrium" is reached, that is, until the ranks of all vertices and edges in the graph comply with the maximum rank equations. This formulation and computation provides a preliminary understanding of face configuration in the object. The computed values represent only the upper bound of the ranks and are denoted by $R^+(v)$ and $R^+(e)$. The actual ranks, in an interpreted solution, are denoted simply by $R(v)$ or $R(e)$. Consider, for example, a trihedral polyhedron; the solution of the maximum-rank equations reverts to the known relationship $R^+(e)=2$ and $R^+(v)=3$ for all vertices and edges.

Fig. 10 depicts a simple projection of some non-manifold object. The maximum ranks $R^+(v)$ and $R^+(e)$ computed for this graph are illustrated in Fig. 10(a). Two subsets of potential faces agreeing with these ranks have been found and are illustrated in Fig. 10(b) and 10(c). Note that these ranks were obtained from a two-dimensional line drawing, and no information was provided about the three-dimensional object it represents.

The advantage of *maximum rank* equations is that they are applicable to non-manifold objects as well as to polyhedra and multi-component combinations of the two. Thus, an upper bound of edge and vertex ranks may be derived *without* prior knowledge about the type of object depicted in the line drawing. This formulation provides a good basis for selecting a subset of potential faces.

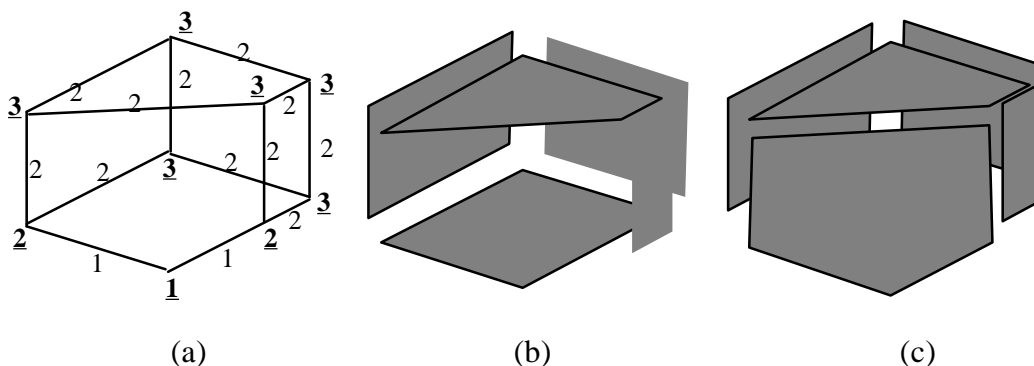


Fig 10. Upper bounds of ranks for a non-manifold object, x denoting vertex ranks, and y denoting edge ranks. (a) Original graph with maximum ranks; (b) solution one: a rectangular box with a diagonal opening; (c) solution two: a trapezoid prism without a bottom.

The rank bounds obtained by the equations (2-5) should be integers. Therefore, equation (4) must be rounded down in case it results in a non-integer value. Such intervention is possible if the equations are solved by means of relaxation. Furthermore, the relaxation solution approach is robust when handling inaccurate line drawings. (See Section 5.9 below.)

The rank bounds obtained by the above formulation can be used to check validity of a given potential face configuration. The computed rank bounds can be used to govern the search process and constrain the search domain. However, to find the *best* solution, a target function must be defined, as discussed in the next section.

5.3 Compliance Function

Since the maximum rank equations only establish an upper bound on the ranks, they allow for a large number of solutions, including the trivial solution of zero faces, implying that the object is a skeletal structure with no faces at all. This solution is mathematically correct but is not likely to be selected by a human observer. To overcome this difficulty, heuristics are used to try to imitate the psychological reasoning of a human observer. According to our observations, a human subject typically selects a configuration in which as many edges as possible participate in as many faces as possible. The target solution, then, is the one agreeing most closely with the upper bounds derived from the maximum rank equations.

For a given subset of potential faces \mathbf{x} , the ranks of the edges and vertices can be computed according to their definition, and a compliance coefficient $\mathbf{g}(\mathbf{x})$ may be computed by minimizing the sum of the deviations of the actual ranks from the upper bounds of the ranks,

$$\mathbf{g}(\mathbf{x}) = \sum \cdot |R^+(e) - R(e)| + \sum \cdot |R^+(v) - R(v)| \rightarrow \min \quad (6)$$

where summation is performed on all vertices and edges in the edge-vertex graph. The compliance function is low for a good fit and high for a bad fit. Also note that this function increases for cases of over-configuration, where the number of faces configured causes the actual rank of an edge or a vertex to exceed the upper bound. The compliance function discourages such a situation, and the search process can be set to prohibit such occurrences.

5.4 Search Methods

Several algorithms are available for searching a combination tree while attempting to minimize a target function such as $\mathbf{g}(\mathbf{x})$. Two of these algorithms were examined.

The **A-star** (A^*) search algorithm [14] searches the problem domain by advancing from one solution state to the next according to a cost function. In the face assignment problem, a state is a particular selection of potential faces complying with the constraints. A^* can advance to the next state by adding another potential face while maintaining the validity of the state. In order to eliminate unnecessary permutations, potential faces can only be added in a predefined order. The cost function evaluated at each state consists of two terms: The 'cost so far' coincides with the compliance function $\mathbf{g}(\mathbf{x})$. This function starts at a high level and decreases monotonously as faces are added to the configuration. The second function—the 'heuristic'—represents an optimistic forecast of the maximum amount by which the compliance function would be further reduced should a given branch of the search tree be followed. This function is denoted by $\mathbf{h}(\mathbf{x})$. If each edge in a face may reduce the compliance function by 1 at best, then an optimistic forecast is that the current cost will be reduced by the total number of edges on the boundaries of the potential faces remaining to be tested, which do not conflict with the already assigned faces. The algorithm to compute $\mathbf{h}(\mathbf{x})$ is, hence, the sum of the number of edges in unassigned non-conflicting faces; this information is obtained using considerations discussed in the next section. As search tree nodes are gradually unfolded, priority is given to the node with the lowest *current + forecast costs*. The drawback of the A^* approach, however, is that it requires a list of open nodes sorted by their *current + forecast costs*. In problems with a wide tree span, maintaining this list drains many resources. Alternative variants of the A^* method, for example, *Iterative deepening A^** [14], can be used.

A second algorithm used in this implementation is **Branch-and-Bound** [14]. This algorithm uses the *current + forecast cost* functions to determine whether the current path is worth pursuing. That is, it checks whether the final outcome of this path has a chance to be better than the best found to that point.

With both of these algorithms, a search path is aborted if the rank of any vertex or edge exceeds its computed upper bound, unless the input graph is known to originate from an inaccurate source, in which case some slack is necessary.

5.5 Face adjacency theorem

The enormous search domain (2^m combinations) calls for further reduction. Two approaches are proposed for reducing the problem's search domain:

- Reducing the size of the potential faces pool (reducing m).
- Limiting unfolding of the combination tree.

When two planar faces in an object share common edges, these edges must be collinear in space because they all lie on the line created by the intersection of the two planar face planes. Therefore, the edges are also collinear in the projection plane. Thus, according to the *adjacency theorem*:

Two adjacent planar faces may coexist in the same object if and only if their common edges are collinear.

The adjacency theorem can be expanded to support non-planar, smooth (non-creased) faces. Any edge or other curve on a general smooth surface must satisfy the surface equation. An edge curve common to two faces must satisfy the equation for both surfaces. If a curve is not smooth at some breaking point, the two vectors tangent to the curve at its breaking point define a plane \mathbf{p} . A smooth surface contains that curve only if it is tangent to the plane \mathbf{p} at the breaking point. Two surfaces sharing a non-smooth edge must both be tangent to the plane \mathbf{p} at the tangent point. This occurs only in two cases: (a) the faces are tangent at their common edges meeting at the breaking point, so that they connect smoothly and no edge is visible, or (b) the two surfaces merge into one, so that the edge is no longer common to *two* faces.

The expanded theorem therefore states that

Two adjacent general faces may coexist in the same object if and only if their common edges are smooth.

The term "smooth" implies a continuous first derivative. Note that if two faces share several, non-continuous edges, as illustrated in Fig. 11, then smoothness is required of each edge separately.

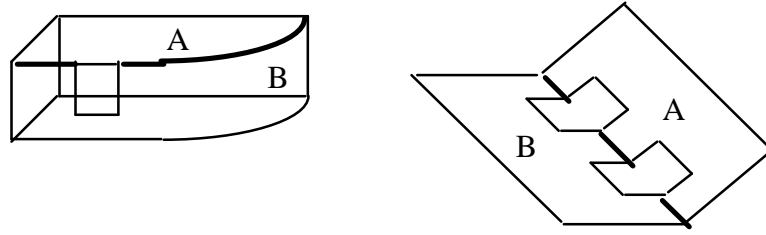


Fig 11. In both models, faces A and B meet along smooth, non-continuous edges.

Smooth edges in the three-dimensional object also appear smooth in the projection. This theorem may therefore be used to disallow certain combinations of potential faces. Whenever a potential face is added while unfolding the combination tree, its boundaries are examined to determine whether it can co-exist with its predecessors. If not, the branch is pruned from that point. To enhance the effect of this theorem, the potential face list is sorted so that larger faces are attempted first. Since larger faces have longer boundaries, they are more likely to conflict with each other; therefore, the search tree branches are pruned first. Implementing the face-adjacency constraint reduces the size of the unfolded search domain by several orders of magnitude and eliminates some implausible solutions (see examples in Table 1). For practical considerations, it is best to compute a binary matrix B with ones and zeroes at position $B(i,j)$ indicating whether face i and j can or cannot co-exist, respectively. This matrix provides fast access to the co-existence information required by the search process.

5.6 Elimination of implausible faces

The adjacency theorem can also be used to reduce the original pool of potential faces by eliminating some implausible faces. The potential face list is searched to find groups in which one face completely encloses the other faces and shares a non-smooth boundary with each of them. For practical purposes, it is sufficient to search for a pair of faces sharing a smooth common edge, with each face sharing a non-smooth edge with a third enclosing face. Fig. 12 illustrates a group of 3 potential faces in which the large face in Fig. 12(a) completely encloses the two others, 12(b) and 12(c), and shares a non-collinear boundary with each of them. According to the adjacency theorem, the enclosing potential face may not co-exist with the two others. It remains to decide whether it is the enclosing face or the two enclosed faces that will remain in the potential face list. Selecting the two enclosed faces can be more helpful in reducing the compliance function than selecting the larger face: together, they raise the ranks of more edges and vertices because of their common boundary which is not part of the larger face. This local analysis makes it possible to determine in advance that the search process will discard the larger face, making it safe to omit it from the potential face pool in advance. Reducing the number of potential faces directly reduces the search domain. This procedure can be applied to larger groups of potential faces.

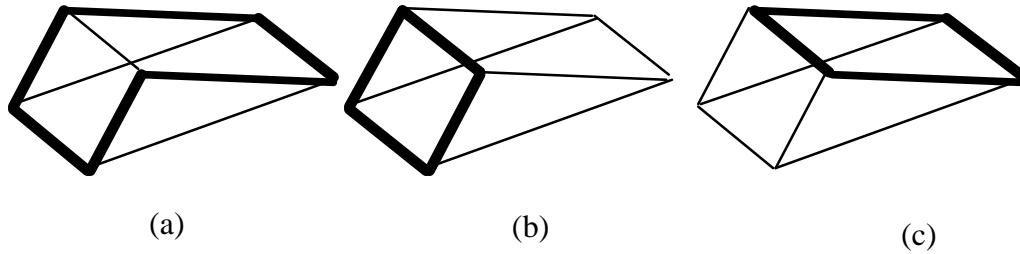


Fig 12. Potential face (a) shares a common edge and encloses face (b) and face (c).

After reduction, the potential face list contains only *minimal* non-self-intersecting circuits of entities.

5.7 Selecting the most plausible solution

Although we must select faces corresponding to the most plausible 3D object, it should be noted again that discussion and decision making is still performed in the projection plane. The search can result in more than one solution. Typically, the number of solutions is very low (up to ten). In addition to the solution that would have been selected by a human observer, the suggested solutions also include other face configurations describing possible objects that are not easy to imagine. Fig. 13 shows a projection of a body with two solutions, displayed with hidden lines removed to illustrate their spatial configuration, with a principal planar face highlighted. The object illustrated in Fig. 13(c) is not apparent at first glance, but it is a valid solution.

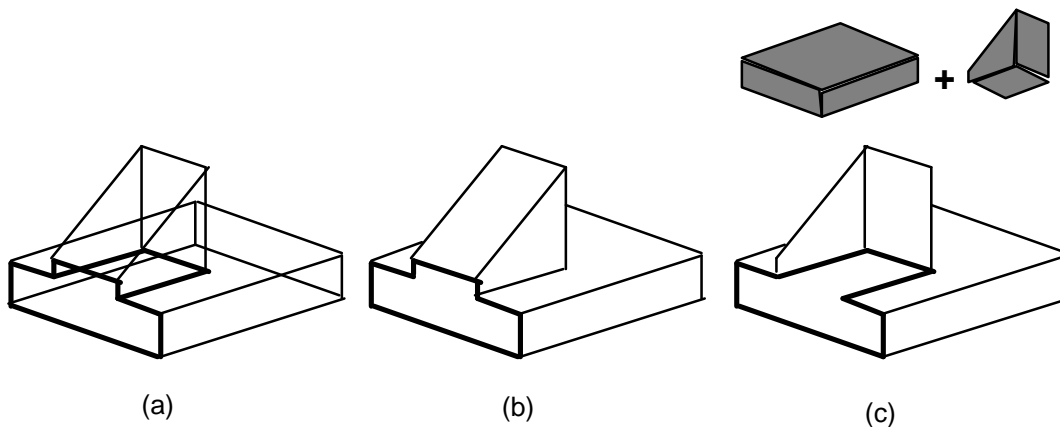


Fig 13. Multiple solutions: (a) original projection, with two potential faces highlighted, (b) natural solution (c) a second valid but unlikely solution.

Selecting the most plausible solution is a matter of determining what makes solution 13(b) more apparent to the human eye. This determination is based on perception of image regularities. The concept of image regularities [8] specifies that certain geometrical relationships between entities in the projection plane suggest spatial relationships. For example, the fact that two lines are parallel in the projection plane *usually* implies that they are also spatially parallel. The solution in Fig. 13(c) is not plausible because lines parallel in the projection 13(a) are not parallel in the 3D object shown in 13(c). Unfortunately, this regularity cannot be used for selecting the plausible face configuration because it relies on comparison with spatial information which is

not available in the 2D projection. However, other regularities may be used. For example, one regularity observable in the face shown in Fig. 13(b) is *skewed orthogonality*. When entities around a face contour alternate between two distinct angles, the face is perceived by a human observer as being planar with contour entities perpendicular to their neighbors. The face highlighted in Fig. 13(c) does not comply with this regularity and therefore is less likely to be perceived as planar. Fig. 14 shows some examples of skewed orthogonality. The statistical behavior of alternating values produced by multiplying the scalar-product and the cross-product of adjacent lines in the projection plane is used to identify these cases. Consistent behavior is likely to represent skewed orthogonality. The degree of skewed orthogonality is evaluated by a weighting coefficient W as follows:

$$w_{\substack{\text{skewed} \\ \text{orthogonality}}} = \sigma(\beta_{i=1..N}) \quad \beta_i = (-1)^i \cdot [\hat{l}_i \cdot \hat{l}_{i+1}] [\hat{l}_i \times \hat{l}_{i+1}] \hat{\mathbf{k}} \quad (7)$$

where N represents the number of lines along the face contour, \hat{l}_i denotes the two-dimensional unit direction vector of a line of the contour in the projection plane, and $\hat{\mathbf{k}}$ represents a unit direction vector perpendicular to the projection plane.

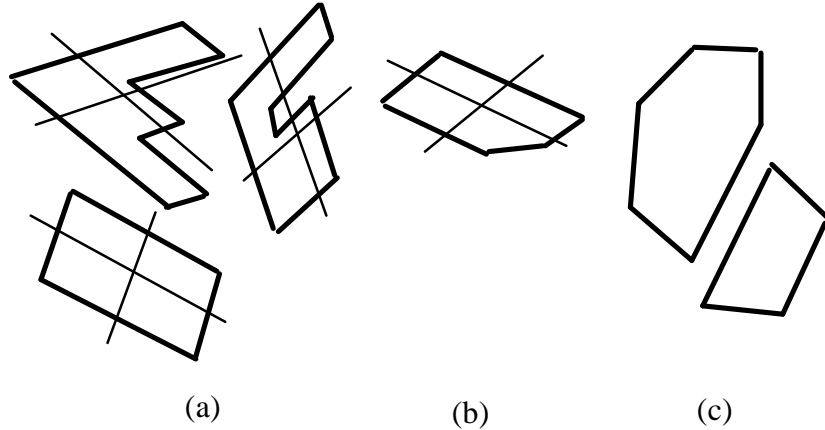


Fig. 14. Faces showing skewed orthogonality: (a) with their skewed principal axis; (b) partial orthogonality; (c) none.

Other image regularities (*skewed symmetry*, for instance) may also be used to determine whether a given face seems to be planar to a human observer. See Fig. 15 for some examples of skewed symmetry. Skewed symmetry detection is more complex and is a research topic on its own. Some techniques are described in [2], [15] and [16].

Note that the above criteria should only be used as means of 'tie breaking' between alternative optimal solutions to the face assignment problems. After all the faces in a solution have been tested for regularities, a total factor is evaluated. The solution with the highest factor is the most plausible alternative.

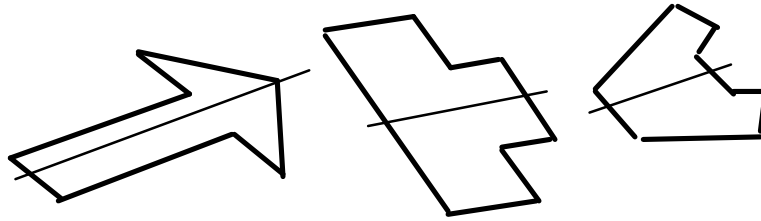


Fig 15. Faces showing skewed symmetry.

5.8 Smooth entity chains

Equations (2) and (3) of the maximum rank equations are based on the assumption that any pair of edges meeting at a vertex may jointly participate in no more than one face boundary. This assumption is a consequence of the face adjacency theorem: that is, two edges that are not joined smoothly at the vertex cannot be shared by any two faces coexisting in the object. However, in cases where two edges meeting at a vertex are collinear (i.e., are joined smoothly), this assumption does not hold, and an extended version of the equation should be used. First, let us define a pair of edges meeting smoothly at a vertex as forming a *smooth entity chain*. A smooth entity chain may span several vertices, as is illustrated by chain A-B-C-D in Fig. 16. Using the same reasoning that led to equation (3), it may be observed that any face boundary passing through an edge which is part of a smooth entity chain must also pass through two edges breaking off that chain, one on each side. (For illustration purposes, these sides will be denoted as *left* and *right*, however they correspond to the two directions of traveling along the curve.) The graph in Fig. 16 indicates that any face boundary passing through edge BC must pass through one edge breaking off the chain on the left, namely AE, AF or BG. Similarly, any face boundary passing through edge BC must also pass through one edge breaking off the chain on the right, namely CH, DI or DJ. Thus, the number of face boundaries passing through a general edge e that is part of an entity chain is limited by the number of edges breaking off the chain on both sides of that edge. More specifically,

$$R(e) \bullet \bullet d(v) - 2n_L + 1 \quad \text{for all the } n_L \text{ vertices to the } \underline{\text{left}} \text{ of edge } e$$

$$R(e) \bullet \bullet d(v) - 2n_R + 1 \quad \text{for all the } n_R \text{ vertices to the } \underline{\text{right}} \text{ of edge } e$$

The rank is limited by the lower of the two, so that

$$R(e) \bullet \min [\bullet d(v_L) - 2n_L, \bullet d(v_R) - 2n_R] + 1 \quad (8)$$

where v_L denote all the n_L vertices along the smooth entity chain on the left of the edge in question (including its left endpoint), and v_R denote all the n_R vertices on its right side (including its right endpoint). Note that when the smooth entity chain consists of one entity alone, i.e., no edges meet smoothly around edge e , n is 1 and equation (8) reverts to equation (2), derived earlier.

The extended equation (8) and equation (4) replace equations (2) and (3) of the maximum rank equations when collinear (smooth) edges meet at a vertex. A correct bound may be computed for the edge segments along the smooth chain, as with the rank of edge BC in the graph in Fig. 16.

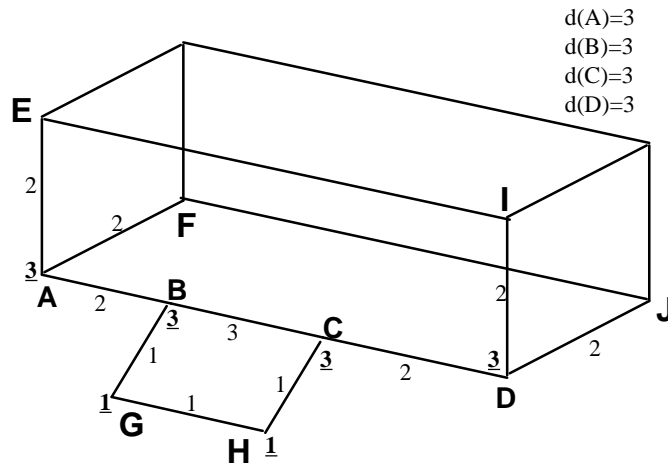


Fig 16. Smooth entity chains (A-B-C-D) must be analyzed with the extended equation (8). The resulting edge and vertex rank-bounds are shown.

5.9 Adaptation to inaccurate drawings

The treatment of inaccurate input is important in applications which process rough drawings such as freehand sketches. Rough input data requires several preprocessing stages that involve smoothing and classifying entities and aligning them at common vertices to form an edge-vertex graph (see [1] for more detail). The terms used to compute maximum ranks have been assumed to be integers. These ranks may become real numbers when inaccuracies are introduced, thus supporting ambiguous cases. For instance, when applying the face adjacency theorem, it may not be possible to determine whether a chain of boundary entities is smooth or not. In such a case, ranks of the participating entities can be set to a value between the rank corresponding to accurate smoothness and the rank corresponding to a non-smooth boundary. Since the identification problem is formulated as a search for optimal cost function, full compliance is not required for a solution to be found. Moreover, the function does not evaluate to zero at its optimum, since estimation is based on local connectivity and does not always hold for all possible configurations. Experiments have shown that this formulation performs correctly even when an arbitrary entity is accidentally omitted from the drawing.

5.10 Algorithm Compendium

The algorithm proposed for general objects may be summarized as a combination tree search problem seeking the optimum of a cost function $g(\mathbf{x})$. The stages are summarized below:

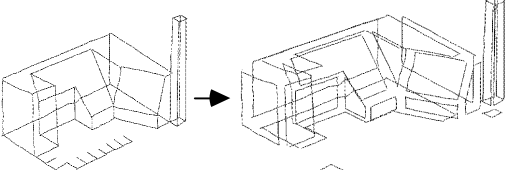
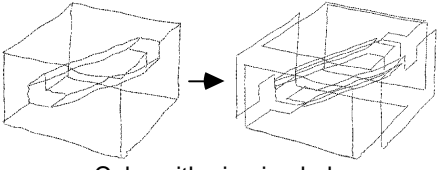
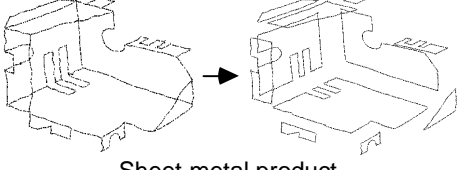
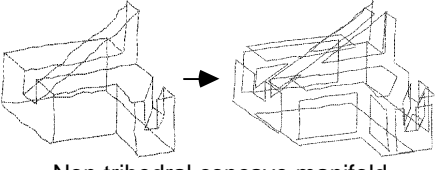
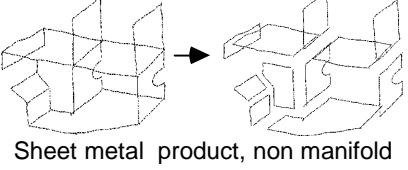
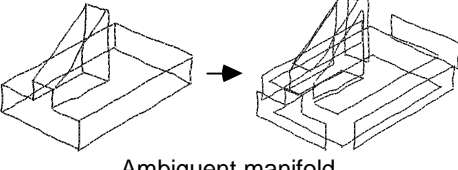
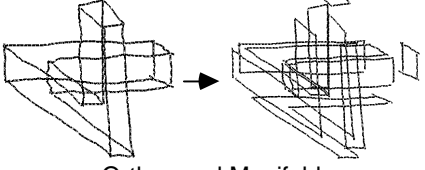
- *Search the input projection to find all possible non-self-intersecting circuits of entities (see section 5.1 for details). This stage can be combined with the criteria described in section 5.6 to construct a reduced set of circuits.*
- *Eliminate the larger face of self-contained groups and obtain a minimal list of potential faces according to the criteria described in section 5.6.*
- *Compute and store whether face pairs may or may not co-exist in the same object, according to the criteria described in section 5.5*

- *Compute upper bounds of ranks for each edge and vertex in the graph, using the equations listed in sections 5.2 and 5.8. Non-accurate graphs can be considered using the approach described in section 5.9.*
- *Use the A* or branch-and-bound algorithms [14], with the target function and setup described in sections 5.3 and 5.4, to search for the optimal configuration of potential faces satisfying the constraints set by the maximum rank equations (section 5.2 and 5.8) and the constraints posed by the face adjacency theorem (sections 5.5 and 5.6).*
- *If more than one solution is obtained, use image regularities such as those described in section 5.7 to select the most plausible solution.*

6 RESULTS

Although the original combination space is extremely large, the above criteria narrow down the search domain to a manageable size. For projections with approximately 30 edges, the correct solution can be found within a few seconds (using an old SGI 4D workstation). Table 1 summarizes principal parameters for some sample projections tested. Each row of the table shows an initial sketch and the resulting face decomposition in the left column (In the more general setup of an application reconstructing a three dimensional object from a rough sketch [1], this raw sketch is analyzed and transformed into an edge-vertex graph using fitting techniques described in [5].) The second column shows the number of vertices and edges in the input graphs. Then, the initial potential face pool is generated (shown in the third column) according to sections 5.1 and 5.6, and analyzed using the algorithm described above. First, minimal potential faces are derived. Column 3 shows the theoretical number of potential faces and the number of minimal faces actually determined. Next, the search algorithm is applied to seek the optimal configuration. Column 4 lists the theoretical number of face configurations in the search domain and the number of configurations actually spanned during the heuristic search. Note that the number of configurations actually searched is indeed a small fraction of the theoretical space, mainly due to the coexistence criteria. Finally, Column 5 lists the number of optimal solutions obtained and the elapsed time. When more than one solution was found, criteria listed in section 5.7 were used to automatically determine the most plausible solution. The selected solution was then visualized by the implementation by translating the input sketch strokes to form a two-dimensional "exploded-view" of the face decomposition.

Examination of the results shown in Table 1 and of other trials as well indicate that indeed the most plausible face configuration is determined even for cases of non manifold, mixed dimensions and higher genus models. Execution time, however, varies and it appears that the exponential nature of the circuit space is the main cause for this behavior.

Input Projection (Freehand Sketch format) and Output Face Decomposition (Actual output of implementation shown)	Input Edge-Vertex Graph Size	Faces	Face Configurations	Solutions & Elapsed Time*
 <p>Building/parking: Mixed dim., non manifold</p>	68 Edges 49 Vertices	194 Potential Faces 53 Minimal Faces 22 Finally Selected	10 ⁵⁸ Face Configurations 254 Actually Tested	1 Solution (shown) 3 [Secs]
 <p>Cube with piercing hole Manifold, genus 1</p>	30 Edges 20 Vertices	73 Potential Faces 49 Minimal Faces 10 Finally Selected	10 ²² Face Configurations 87 Actually Tested	4 Solutions (automatically selected solution shown) 1 [Sec]
 <p>Sheet metal product Non manifold, with holes</p>	70 Edges 57 Vertices	77 Potential Faces 25 Minimal Faces 11 Finally Selected	10 ²³ Face Configurations 122 Actually Tested	4 Solutions (automatically selected solution shown) 4 [Secs]
 <p>Non trihedral concave manifold</p>	46 Edges 30 Vertices	72 Potential Faces 28 Minimal Faces 18 Finally Selected	10 ²¹ Face Configurations 183 Actually Tested	1 Solution (shown) 2 [Secs]
 <p>Sheet metal product, non manifold</p>	39 Edges 28 Vertices	156 Potential Faces 41 Minimal Faces 11 Selected	10 ⁴⁶ Face Configurations 70 Actually Tested	1 Solution (shown) 3 [Secs]
 <p>Ambiguous manifold</p>	24 Edges 16 Vertices	37 Potential Faces 19 Minimal Faces 10 Finally Selected	10 ¹¹ Face Configurations 72 Actually Tested	2 Solutions (automatically selected solution shown) 1 [Sec]
 <p>Orthogonal Manifold</p>	24 Edges 16 Vertices	21 Potential Faces 15 Minimal Faces 10 Finally Selected	10 ⁶ Face Configurations 56 Actually Tested	1 Solution (shown) 1 [Sec]

* Elapsed times were obtained on an SGI Iris 4D Workstation 30MHz and include the time required to find the potential circuits.

Table 1. Summary of parameters for various test cases.

7 CONCLUSIONS

This paper presents a method for identifying the actual faces of a wireframe object depicted by a single two-dimensional projection. Two methods were proposed: (a) an analytical procedure for analyzing drawings of manifold objects of genus zero, and (b) an optimization-based procedure for analyzing projections of general objects. In addition, a formulation of maximum-rank equations is proposed, based on the face adjacency theorem. The maximum-rank equations are applicable to a wide variety of object types, including non-manifold objects. Using this formulation, a preliminary understanding of the object depicted by the two-dimensional projection can be derived **without** any prior knowledge about the object itself or its type. This method provides important information for the process of reconstructing three-dimensional geometry from single line drawings, and can thus aid in automatic interpretation of line drawings.

A basic drawback of the proposed algorithm is that it is not output sensitive in its execution time. While the complexity of the solution of the face assignment problem (i.e. the number of faces) is proportional to the complexity of the input graph (n vertices), the algorithm has to search a domain which is generally exponentially related to this size. It is not clear whether this stage is inevitable in pursuing the most plausible solution, and further research is required to examine whether some of the constraints established here can be used to further reduce the initial face pool to a polynomial size. Nevertheless, the algorithm is appropriate for interpreting hand-sketched scenes, certainly within an application in which the scene is gradually sketched in several stages.

Further research is also required to attempt to find the optimal face configuration using techniques other than heuristic search. One promising direction is the integer linear programming (ILP) approach. All the constraints derived in this paper take an integer form; the target function is linear and the variables are integers. Thus, the problem resides in a strictly integer space (as opposed to mixed integer). Several new and highly efficient methods are now available for solving this type of problems (see, for example, [17]).

ACKNOWLEDGMENT

This work was carried out in partial fulfillment of the requirements for the degree of D.Sc. in Mechanical Engineering at the Technion [Hod Lipson]. This research has been supported in part by the Fund for the Promotion of Research at the Technion, Research No. 033-028.

REFERENCES

- [1] H. Lipson and M. Shpitalni, "A new interface for conceptual design based on object reconstruction from a single freehand sketch," *Annals of the CIRP*, Vol. 45/1, pp. 133-136, 1995.
- [2] Lipson, H. and Shpitalni, M., "Optimization-Based Reconstruction of a 3D Object From a Single Freehand Line Drawing," *Journal of Computer Aided Design*, in print, 1995.
- [3] M. Mantyla, *An Introduction to Solid Modeling*, Computer Science Press, 1988.
- [4] A. Gibbson, *Algorithmic Graph Theory*, Cambridge University Press, 1985

- [5] Shpitalni, M. and Lipson, H., "Classification of Sketch Strokes and Corner Detection using Conic Sections and Adaptive Clustering," submitted to *Trans of the ASME, Journal of Mechanical Design*, 1995.
- [6] D. A. Huffman, "Impossible objects as nonsense sentences," *Machine Intelligence 6*, Edinburgh Univ. Press., pp. 295-323, 1971.
- [7] M. B. Clowes, "On seeing things," *Artificial Intelligence*, Vol. 2, pp. 79-116, 1971.
- [8] T. Kanade, "Recovery of the three-dimensional shape of an object from a single view," *Artificial Intelligence*, Vol. 17, pp. 409-460, 1980.
- [9] K. Sugihara, *Machine interpretation of line drawings*, MIT Press, 1986.
- [10] E. Martí, J. Regincós, J. López-Krahe and J.J. Villanueva, "Hand line drawing interpretation as three-dimensional object," *Signal Processing*, Vol. 32, pp. 91-110, 1993.
- [11] Y. G. Leclerc and M. A. Fisler, "An optimization based approach to the interpretation of single line drawings as 3D wire frames," *Int. J. of Computer Vision*, Vol. 9, No 2, pp. 113-136, 1992.
- [12] W. Whitney, "Motions and stresses of projected polyhedra", *Structural Topology* 7, pp. 13-38, 1982.
- [13] E. Steinitz, "Polyeder und Raumeinteilungen", *Encyclopadie der Mathematischen Wissenschaften*, Band 3 (Geometrie) Teil 3AB12, pp. 1-139, 1922.
- [14] E. Rich and K. Knight, *Artificial Intelligence*, McGraw-Hill Inc., 1991.
- [15] Friedberg, S A 'Finding axes of skewed symmetry' *Computer Vision, Graphics, and Image Processing*, No 34, pp. 138-155, 1986.
- [16] Yip, R K K, Tam, P K S and Leung, D N K 'Application of elliptic Fourier descriptors to symmetry detection under parallel projection' *IEEE PAMI*, Vol. 16, No 3, pp. 277-286, 1994.
- [17] P. Conti and C. Traverso, "Buchberger algorithm and integer programming", *Proceedings AAECC-9*, Springer LNCS, 539: pp. 130-139, 1991.

APPENDIX A

GLOSSARY OF TOPOLOGY AND GRAPH THEORY TERMS

Genus	A genus of an object corresponds to the number of non-intersecting closed curves that can be drawn on it without separating it into two regions [3]. For example, a sphere has genus 0, and a torus has genus 1.
Graph	Geometrically, a graph is defined as a set of points (vertices) in space which are interconnected by a set of lines (edges) [4].
Planar graph	A planar graph is a graph which can be embedded (drawn) in the plane without any two edges crossing each other.
Manifold	A 2-manifold is an object in which every point on its boundary belongs to (is surrounded by) a 2-dimensional region [3].
3-vertex-connected	A graph is said to be three-vertex-connected if there is a set of three vertices at which the graph can be disconnected and partitioned into two components.
Graph Face	A graph face is an empty area in a planar graph surrounded by a loop of edges.
Circuit	A circuit is a closed loop of edges, not passing through the same vertex more than once.