Complex 3D General Object Reconstruction from Line Drawings

Linjie Yang[1], Jianzhuang Liu[2,1,3], and Xiaoou Tang[1,2]
[1]Department of Information Engineering, The Chinese University of Hong Kong
[2]Shenzhen Key Lab of Computer Vision and Pattern Recognition
Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, China
[3]Media Lab, Huawei Technologies Co. Ltd., China
{yl012,jzliu,xtang}@ie.cuhk.edu.hk

Abstract

An important topic in computer vision is 3D object reconstruction from line drawings. Previous algorithms either deal with simple general objects or are limited to only manifolds (a subset of solids). In this paper, we propose a novel approach to 3D reconstruction of complex general objects, including manifolds, non-manifold solids, and non-solids. Through developing some 3D object properties, we use the degree of freedom of objects to decompose a complex line drawing into multiple simpler line drawings that represent meaningful building blocks of a complex object. After 3D objects are reconstructed from the decomposed line drawings, they are merged to form a complex object from their touching faces, edges, and vertices. Our experiments show a number of reconstruction examples from both complex line drawings and images with line drawings superimposed. Comparisons are also given to indicate that our algorithm can deal with much more complex line drawings of general objects than previous algorithms.

1. Introduction and Related Work

A 2D line drawing is the most straightforward way of illustrating a 3D object. Given a line drawing representing a 3D object, our visual system can understand the 3D structure easily. For example, we can interpret without difficulty the line drawing shown in Fig. 1(a) as a castle with four walls and one door. Imitating this ability has been a longstanding and challenging topic in computer vision when a line drawing is as complex as this example. The applications of this work include 3D object design in CAD and for 3D printers, 3D query generation for 3D object retrieval, and 3D modeling from images.

In this paper, same as the majority of related work, a line drawing is defined as the orthogonal projection of the edges and vertices of a 3D object in a generic view, and objects with planar surfaces are considered. A line drawing is represented by an edge-vertex graph. It can be obtained by the user/designer who draws on the screen with a tablet pen, a mouse, or a finger (on a touch sensitive screen), with all, with some, or without hidden edges and vertices.

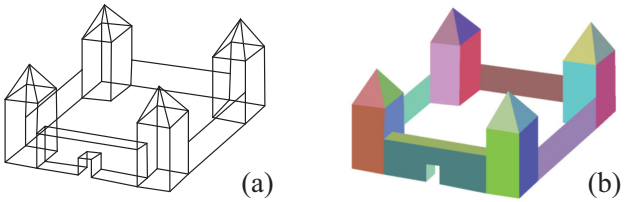


Fig. 1. (a) A line drawing representing a castle. (b) The 3D model of the line drawing.

Line labeling is the earliest work to interpret line drawings [1], [17]. It searches for a set of consistent labels such as convex, concave, and occluding from a line drawing to test its correctness and/or realizability. Line labeling itself cannot recover 3D shape from a line drawing. Later, 3D reconstruction from the contours (line drawings) of objects in images is studied [19], [14], [13], which handles simple objects only. Model-based 3D reconstruction [2], [3], [20] can deal with more complex objects, but these methods require to pre-define a set of parametric models.

Recently, popular methods of 3D reconstruction from line drawings are optimization based, which are most related to our work and are reviewed next. These methods can be classified into two categories: one dealing with manifolds and the other dealing with general objects. A general object can be a manifold, non-manifold solid, or non-solid. Manifolds are a subset of solids, defined as follows:

A manifold, or more rigorously 2D manifold, is a solid where every point on its surface has a neighborhood topologically equivalent to an open disk in the 2D Euclidean space.

In this paper, a solid is a portion of 3D space bounded by planar faces, and a manifold is also bounded by planar faces. Each edge of such manifolds is shared exactly by two faces [4].

Most 3D reconstruction methods from a line drawing assume that the face topology of the line drawing is known in advance. This information can reduce the reconstruction complexity greatly. Algorithms have been developed to find faces from a line drawing in [16], [10], and [9], where [16] and [10] are for general objects and [9] for manifolds.

Optimization-based 3D reconstruction depends on some critera (also called image regularities) that simulate our visual perception. Marill proposes a very simple but effective criterion to reconstruct a simple object: minimizing the standard deviation of the angles (MSDA) in the object [11]. Later, other regularities are proposed to deal with more complex objects such as face planarity, line parallelism, isometry, and corner orthogonality [5], [6], [15], [18]. In these methods, an objective function

$$\Phi(z_1, z_2, ..., z_{N_v}) = \sum_{i=1}^{C} \omega_i \phi_i(z_1, z_2, ..., z_{N_v}) \qquad (1)$$

is minimized to derive the depths $z_1, z_2, ..., z_{N_v}$, where N_v is the number of vertices in the line drawing, $\phi_i, i = 1, 2, ..., C$, are the regularities, and $\omega_i, i = 1, 2, ..., C$, are the weights. The main problem in this approach is that these algorithms are easy to get trapped into local minima (obtaining failed results) when a line drawing is complex with many vertices, due to the search in a high-dimensional space (N_v dimensions) with the non-convex objective function. For example, the search space is of 56 dimensions for the object in Fig. 1(a).

To alleviate this problem, Liu et al. formulate 3D reconstruction in a lower dimensional space so that the optimization procedure has a better chance to find desired results [7]. For the complex object in Fig. 1(a), however, the search in a space with 18 dimensions is still too difficult for it to obtain a satisfactory 3D object (see Section 3).

The methods in [5], [6], [15], [18], and [7] reconstruct general objects, and the one in [7] can deal with more complex objects than the other four. But these algorithms cannot avoid the local minimum problem in a high dimensional search space when a line drawing is complex.

In [8], a divide-and-conquer (D&C) strategy is used to tackle this problem. It first separates a complex line drawing into multiple simpler ones, then independently recovers the 3D objects from these line drawings, and finally merges them to form a complete object. Since the separated line drawings are much simpler than the original one, the 3D reconstruction from each of them is an easy task.

This D&C approach handles manifolds only. Based on known faces found by the face identification algorithm in [9], it uses manifold properties to find internal faces

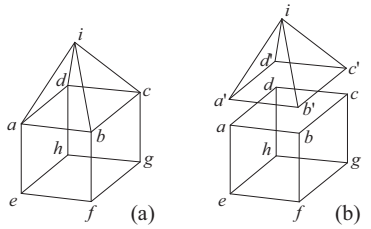
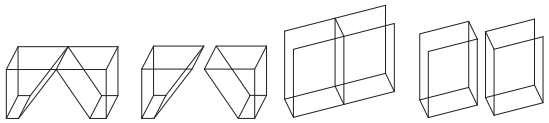


Fig. 2. (a) A simple manifold with nine faces and one internal face (a, b, c, d). (b) Decomposition result from the internal face.



(a) (b) (c) (d)

Fig. 3. (a) A non-manifold solid. (b) Expected decomposition of (a). (c) A sheet object. (d) Expected decomposition of (c).

from a line drawing and then separates the line drawing from the internal faces. An internal face is defined as an imaginary face lying inside a manifold with only its edges visible on the surface [8]. It is not a real face but can be considered as two coincident real faces of identical shape belonging to two manifolds. For example, Fig. 2(a) shows a manifold with nine faces. The D&C first finds the internal face (a, b, c, d) and then decomposes the line drawing from this internal face (Fig. 2(b)).

However, handling manifolds only limits the applications of [8]. In many applications in computer vision and graphics such as 3D object matching, retrieval, and rendering, it is unnecessary to represent objects as manifolds, in order to facilitate data processing and reduce data storage. For example, a flat ground can be represented by a sheet (one face), but if it is represented by a manifold, a thin box with six faces has to be used. Fig. 1(a), Fig. 3(a), and Fig. 3(c) are line drawings of three non-manifolds.

In this paper, we propose a novel approach to 3D reconstruction of complex general objects based on visual perception, object properties, and new line drawing decomposition. Compared with previous methods, ours can deal with much more complex line drawings of general objects. It can handle not only manifolds but also non-manifold solids and non-solids, and is insensitive to sketching errors.

2. General Object Reconstruction

We also use the D&C strategy to deal with 3D reconstruction from a line drawing representing a complex general object. The key is how to decompose a complex line drawing of any objects into multiple simpler line drawings. These decomposed line drawings should represent objects that are in accordance with our visual perception, which makes the 3D reconstruction from these line drawings easier and better because the regularities used to build an objective function for reconstruction follow human perception of

common objects [11], [5], [6], [15], [18].

Before the decomposition of a line drawing, we assume that all the real and internal faces of the object have been obtained from the line drawing using a face identification algorithm. For example, the algorithm in [10] finds 10 faces from the line drawing in Fig. 2(a) (including the internal face), and obtains 12 and 7 faces from the line drawings in Figs. 3(a) and (c), respectively.

2.1. Decomposing line drawings of solids

In this subsection, we consider the line drawings of solids first. The decomposition method will be extended to the line drawings of general objects in the next subsection.

It is not difficult to see that in general, a complex object, especially a manmade complex object, can be considered as the combinations of multiple smaller objects. The most common combination is the touch of two faces from two different objects such as the one in Fig. 2. Other combinations are the touches among lines, faces, and vertices. Our target is to decompose a complex solid into multiple *primitive solids*. Before the definition of a primitive solid, we introduce a term called the *degree of freedom* of a solid.

Definition 1. *The degree of freedom (DoF) of a 3D solid represented by a line drawing is the minimal number of z-coordinates that can uniquely determine this 3D solid.*

This is the first time that the concept of DoF is used to decompose line drawings. Now let us consider a simple object in Fig. 4(a). The cube has six faces: (v_1, v_2, v_3, v_4), (v_1, v_2, v_6, v_5), (v_1, v_4, v_8, v_5), (v_2, v_3, v_7, v_6), (v_4, v_3, v_7, v_8), and (v_6, v_7, v_8, v_5). We can show that the cube is determined if the z-coordinates of its four non-coplanar vertices are known. Without loss of generality, suppose z_1, z_2, z_4, and z_5 are known. Since the 3D coordinates of v_1, v_2, and v_4 are fixed (remind that the x- and y-coordinates of all the vertices are known under the orthogonal projection), the 3D plane passing through the face (v_1, v_2, v_3, v_4) is determined, and thus z_3 can be calculated. Similarly, z_6 and z_8 can be obtained. Finally, z_7 can be computed with the 3D coordinates of v_3, v_4, and v_8 known, which determine the plane passing through the face (v_4, v_3, v_7, v_8). So the 3D cube can be determined by the known four z-coordinates, z_1, z_2, z_4, and z_5. Further, it can be verified that three 3D vertices cannot determine this object uniquely because they can only define one face in 3D space. Therefore, the DoF of the cube is 4.

Similar analysis allows us to know that the solids in Fig. 2(b), Fig. 3(b), and Fig. 4(b) all have DoF 4, while the two solids in Fig. 2(a) and Fig. 3(a) have DoFs 5 and 6, respectively. From these analysis, we can have the intuition that solids with DoF 4 serve as the building blocks of more complex solids whose DoFs are more than 4. Besides, we have the following property:

Property 1. *There is no solid with DoF less than 4.*

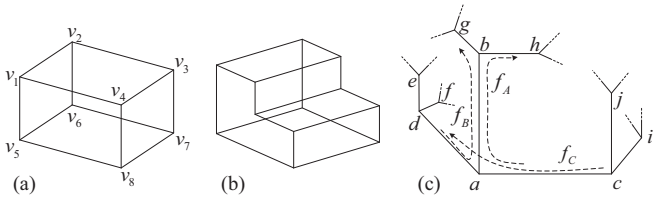


Fig. 4. (a) A cube whose DoF is 4. (b) Another solid whose DoF is also 4. (c) Part of a line drawing with each vertex of degree 3.

This property is easy to verify. A solid with fewest faces is a tetrahedron. Every two of its four faces are not co-planar. Three 3D vertices of a tetrahedron can only determine one 3D face. Next, we define primitive solids.

Definition 2. *A 3D solid represented by a line drawing is called a primitive solid if its DoF is 4.*

Property 2. *If every vertex of a 3D solid represented by a line drawing has degree 3*[1]*, then it is a primitive solid.*

Proof. Let part of such a line drawing be the one as shown in Fig. 4(c). At each vertex, every two of the three edges form a face, because a solid is bounded by faces without dangling faces and edges. Let the three paths f_A, f_B, and f_C in Fig. 4(c) denote the three faces at vertex a.

Without loss of generality, suppose that the four z-coordinates (and thus the four 3D coordinates) of vertices a, b, c, and d are known. Then the three planes passing through f_A, f_B, and f_C are determined in 3D space. With the two known 3D planes passing through f_A and f_B at vertex b, the 3D coordinates of vertices g and h connected to b can be computed. Similarly, the 3D coordinates of vertices e and f connected to d and the 3D coordinates of vertices i and j connected to c can be obtained. Furthermore, all the 3D coordinates of the other vertices connected to e, f, g, h, i, and j can be derived in the same way. This derivation can propagate to all the vertices of this solid. Therefore, the DoF of this solid is 4 and it is a primitive solid. □

Property 3. *The DoF of a solid is 5 which is obtained by gluing two faces of two primitive solids.*

Proof. Let the two primitive solids be PS_1 and PS_2 and their corresponding gluing faces be f_1 and f_2, respectively. The DoFs of PS_1 and PS_2 are both 4. Suppose that PS_1 is determined in 3D space, which requires four z-coordinates. Then f_1 and f_2 are also determined in 3D space. When the z-coordinates of three vertices on PS_2 are known based on f_2, one more z-coordinate of a vertex not coplanar with f_2 on PS_2 can determine PS_2 in 3D space. Therefore, the DoF of the combined solid is 5. □

Fig. 2 is a typical example of two primitive solids gluing together along faces. Fig. 3(a) is an example of two primitive solids gluing together along edges. Two primitive solids may also connect at one vertex. The following property is easy to verify.

[1] The degree of a vertex is the number of edges connected to this vertex.

Property 4. *The DoF of a solid is 6 which is obtained by gluing two edges of two primitive solids. The DoF of a solid is 7 which is obtained by gluing two vertices of two primitive solids.*

From the above properties, we can see that primitive solids are indeed the "smallest" solids in terms of DoF and they can serve as the building blocks to construct more complex solids. Therefore, our next target is to decompose a line drawing representing a complex solid into multiple line drawings representing primitive solids. Before giving Definition 3, we define some terms first.

Vertex set of a face. The vertex set $Ver(f)$ of a face f is the set of all the vertices of f.

Fixed vertex. A fixed vertex is one with its z-coordinate (thus its 3D coordinate) known.

Unfixed vertex. An unfixed vertex is one with its z-coordinate unknown.

Fixed face. A fixed face is one with its 3D position determined by its three fixed vertices.

Unfixed face. An unfixed face is one with its 3D position undetermined.

Definition 3. *Let the vertex set and the face set of a line drawing be $V = \{v_1, v_2, ..., v_n\}$ and $F = \{f_1, f_2, ..., f_m\}$, respectively, where n and m are the numbers of the vertices and the faces, respectively. Also let V_{fixed}, F_{fixed}, $V_{unfixed}$, and $F_{unfixed}$ be the sets of fixed vertices, fixed faces, unfixed vertices, and unfixed faces, respectively. Suppose that an initial set of two fixed neighboring faces sharing an edge is $F_{initial}$ with all their fixed vertices in $V_{initial}$. The final F_{fixed} in Algorithm 1 is called the maximum extended face set (MEFS) from $F_{initial}$.*

In Algorithm 1, a face f that satisfies the condition in step 3 is a face that has been determined in 3D space by the current fixed vertices in F_{fixed}. When this face is found, it becomes a fixed face and all its vertices become fixed vertices. The DoF of the initial two fixed faces combined is 4. It is not difficult to see that the algorithm does not increase the initial DoF, and thus the final object represented by the MEFS also has DoF 4. Next, let us consider a simple example shown in Fig. 2(a) with the following three cases:

Case 1. Suppose that $F_{initial} = \{(e, f, g, h), (e, f, b, a)\}$, $V_{initial} = \{e, f, g, h, b, a\}$, and the algorithm adds the faces into F_{fixed} in this order: $(f, g, c, b) \rightarrow (a, b, c, d) \rightarrow (e, h, d, a) \rightarrow (g, h, d, c)$. Then the final object found by the algorithm is the cube. Note that the algorithm does not add any triangular faces into F_{fixed} because they do not satisfy the condition in step 3.

Case 2. If $F_{initial} = \{(b, i, a), (b, i, c)\}$, then the final object found is the pyramid, and the algorithm does not add any rectangular faces except (a, b, c, d) into F_{fixed}.

Case 3. If $F_{initial} = \{(b, a, i), (e, f, b, a)\}$, the algorithm cannot find any other faces to add to F_{fixed}. Thus, it fails to find the cube or pyramid.

Algorithm 1 Face extending procedure

Initialization: $F_{unfixed} = F \setminus F_{initial}$, $F_{fixed} = F_{initial}$, $V_{fixed} = V_{initial}$, $V_{unfixed} = V \setminus V_{initial}$.

1. **do** the following steps **until** no face satisfies the condition in step 3;
2. Find a face $f \in F_{unfixed}$ that satisfies
3. the number of non-collinear vertices in $Ver(f) \cap V_{fixed}$ is more than 2;
4. Add face f into F_{fixed} and delete it from $F_{unfixed}$;
5. For each vertex $v \in Ver(f)$, if $v \in V_{unfixed}$, add v into V_{fixed} and delete it from $V_{unfixed}$;

Return The final F_{fixed}.

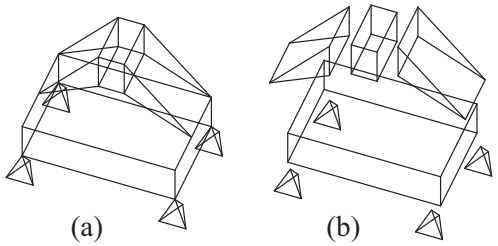


Fig. 5. (a) A complex line drawing of non-manifold solid. (b) The decomposition result by our algorithm.

In case 3, the object represented by the MEFS has only two initial faces and this object is discarded. In order not to miss a primitive solid, we run Algorithm 1 multiple times each with a different pair of neighboring faces in $F_{initial}$. Then, we can always have $F_{initial}$ with its two faces from one primitive solid. For the object in Fig. 2(a), we can always find the cube and the pyramid. Note that the same primitive solid may be found multiple times from different $F_{initial}$, and finally we keep only one copy of each different object (cube and pyramid in this example).

When a complex solid is formed by more than two primitive solids, Algorithm 1 can still be used to find the primitive solids, which is the decomposition result of the complex line drawing. More complex examples are given in Section 3. Besides, Algorithm 1 can also deal with complex solids formed by gluing primitive solids between edges and vertices. Fig. 5(a) is a solid constructed by gluing eight primitive solids between faces, edges, and vertices. Running Algorithm 1 multiple times with different pairs of neighboring faces in $F_{initial}$ generates the primitive solids as shown in Fig. 5(b).

2.2. Decomposing line drawings of general objects

A general object can be a manifold, non-manifold solid, or non-solid. Given a line drawing representing a general object, it is unknown whether this object consists of only primitive solids. However, we can always apply Algorithm 1 to the line drawing multiple times, each with a

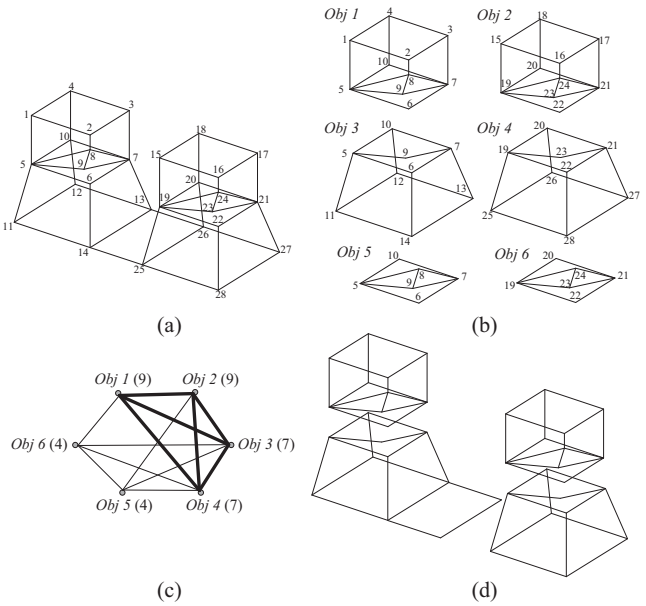


Fig. 6. Illustration of our decomposition method. (a) A line drawing. (b) The set of MEFSs from (a). (c) The weighted object-coexistence graph where the maximum weight clique is shown in bold. (d) The decomposition of (a).

different pair of neighboring faces in $F_{initial}$, generating a set S_{MEFS} of MEFSs (recall that an MEFS with only two initial neighboring faces is discarded). In what follows, we also call an MEFS an object, which is represented by the MEFS. Note that an MEFS generated from a general line drawing may not be a primitive solid, but its DoF must be 4. Objects of DoF 4 have relatively simple structures and are easy to be reconstructed. A number of decomposition examples of complex general line drawings can be seen from the experimental section.

One issue existing in this decomposition method is that two different MEFSs may share many faces. For example, from the line drawing in Fig. 6(a), all different MEFSs found by running Algorithm 1 multiple times are shown in Fig. 6(b), where *Obj 1* and *Obj 5* share four faces, and so do *Obj 2* and *Obj 6*. Obviously, *Obj 5* and *Obj 6* are not necessary. Next we define *object coexistence* and a rule to choose objects.

Definition 4. *Two objects are called coexistent if they share no face or share only coplanar faces.*

Rule 1. *Choose a subset of S_{MEFS} such that in the subset, all the objects are coexistent and the number of total faces is maximized.*

From Definition 4, *Obj 1* and *Obj 5* are not coexistent in Fig. 6, and *Obj 2* and *Obj 6* are not either. If *Obj 5* and *Obj 6* are kept with *Obj 1* and *Obj 2* discarded, many faces in the original object will be missing. Rule 1 guarantees that *Obj 1* and *Obj 2* are kept but not *Obj 5* and *Obj 6*.

Algorithm 2 Decomposition of a general line drawing

Input: A Line Drawing: $G = (V, E, F)$.
Initialization: $S_{MEFS} = \emptyset$, $S_{MWC} = \emptyset$.

1. **for** each pair of neighboring faces $\{f_a, f_b\}$ in F **do**
2. Call **Algorithm 1** with $F_{initial} = \{f_a, f_b\}$ and $V_{initial} = Ver(f_a) \cup Ver(f_b)$;
3. **if** the returned F_{fixed} from Algorithm 1 contains more than two faces **do**
4. $S_{MEFS} \leftarrow F_{fixed}$;
5. Construct the object-coexistence graph G_{obj} with S_{MEFS};
6. $S_{MWC} \leftarrow$ the maximum weight clique found from G_{obj};
7. **for** each face f not contained in S_{MWC} **do**
8. Attach f to the object in S_{MWC} that contains the maximum number of the vertices of f;

Return S_{MWC}.

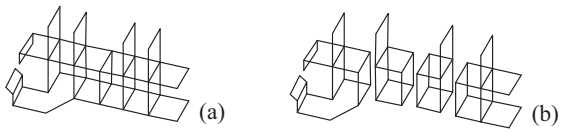


Fig. 7. (a) A sheet object with 23 faces. (b) Decomposition result by Algorithm 2 with the modification in Algorithm 1.

We formulate Rule 1 as a maximum weight clique problem (MWCP), which is to find a clique[2] of the maximum weight from a weighted graph. First, we construct a weighted graph, called the object-coexistence graph, in which a vertex denotes an object in S_{MEFS} and there is an edge connecting two vertices if the two objects represented by the two vertices are coexistent. Besides, each vertex is assigned a weight equal to the number of the faces of the corresponding object. The MWCP is a well-known NP-hard problem. In our application, however, solving this problem is fast enough since an object-coexistence graph usually has less than 20 objects (vertices). We use the algorithm in [12] to deal with this problem.

Fig. 6(c) is the object-coexistence graph constructed from the six objects in Fig. 6(b), where the weights of the vertices are denoted by the numbers in the parentheses. The maximum weight clique is shown in bold.

From Fig. 6, we see that the face (14, 13, 26, 25) is not contained in S_{WMC}, which is used to store the objects in the maximum weight clique. This face is finally attached to *Obj 3*. In general, each of the faces not in S_{WMC} is attached to an object that contains the maximum number of the vertices of this face. If there are two or more objects that contain the same number of the vertices of this face, this face is assigned to any of them.

[2]A clique is a subgraph of a graph such that every two vertices in the subgraph are connected by an edge.

Algorithm 2 shows the complete algorithm to decompose a general line drawing. Steps 7 and 8 attach the faces not in S_{MWC} to some objects in S_{MWC}.

A common complex object usually consists of primitive solids and sheets, and Algorithm 2 works well for the decomposition of most complex line drawings. However, there are still some line drawings the algorithm cannot deal with. Such an example is shown in Fig. 7(a) which is a sheet object with 23 faces. In Algorithm 1, with any pair of initial neighboring faces, there is no any other face satisfying the condition in step 3, thus no object of DoF 4 will be found. The following scheme can solve this problem.

Given a line drawing, steps 1–6 in Algorithm 2 are used to decompose it into multiple objects of DoF 4. If there are separate groups of faces not in S_{MWC}, where the faces in each group are connected, then attach the groups each with less than four faces to some objects in S_{MWC}[3] (the attachment method is similar to steps 7 and 8 in Algorithm 2). For a group with four or more connected faces, Algorithm 2 is applied to it with a minor modification in Algorithm 1. The modification is to set $F_{initial}$ to contain three connected faces whose combined DoF is 5. This modification allows the search of objects of DoF 5. Suppose the object in Fig. 7(a) is such a group. Applying Algorithm 2 to it with the minor modification generates the decomposition result as shown in Fig. 7(b).

2.3. 3D Reconstruction

A complex line drawing can be decomposed into several simpler ones using the method proposed in Sections 2.1 and 2.2. The next step is to reconstruct a 3D object from each of them, which is an easy task because the decomposed line drawings are simple. The method in [6] or [7] can carry out this task very well. We use the one in [6] for our work with the objective function $\Phi(z_1, z_2, ..., z_{N_v})$ constructed by these five image regularities: MSDA, face planarity, line parallelism, isometry, and corner orthogonality. The details of the regularities can be found from [6].

After obtaining the 3D objects from all the decomposed line drawings, the next step is to merge them to form one complex object. When merging two 3D objects, since they are reconstructed separately, the gluing parts (face or edge) of them are usually not of the same size. Then one object is automatically rescaled according to the sizes of the two gluing parts, and the vertices of the gluing part of this object are also adjusted so that the two parts are the same. After merging all the 3D objects, the whole object is fine-tuned by minimizing the objective function Φ on the object.

We can also apply our method to reconstruct 3D shapes from objects in images. First, the user draws a line drawing along the visible edges of an object and he/she can also guess (draw) the hidden edges. Then from this line drawing, our approach described above reconstructs the 3D geometry of the object in the image.

[3]The reason to attach a group with less than four faces to an object in S_{MWC} is that this group is small and is not necessary to be an independent object to reconstruct.

3. Experimental Results

In this section, we show a number of complex 3D reconstruction examples from both line drawings and images to demonstrate the performance of our approach. The first set of experiments in Fig. 8 has nine complex line drawings. Fig. 8(a) is a manifold, and the others are non-manifold solids or non-solids. The decompositions of the line drawings are also given in the figure, from which we can see that the results are in accordance with our visual perception very well. All the primitive solids are found by our algorithm. It is the successful decompositions that make the 3D reconstructions from these complex line drawings possible. The expected satisfactory reconstruction results are shown also in Fig. 8 each in two views.

Fig. 9 shows another set of 3D reconstructions from objects in images with line drawings drawn on the objects. The decomposition results are omitted due to the space limitation. Each reconstruction result obtained by our algorithm is shown in two views with the texture from the image mapped onto the surface. We can see that the results are very good. The details of the objects and the line drawings can be shown by enlarging the figures on the screen.

Among all the previous algorithms for general object reconstruction, the one in [7] can deal with most complex objects. Due to the local minimum problem in a high dimensional search space, however, this algorithm cannot handle line drawings as complex as those in Figs. 8 and 9. For example, Fig. 10(a) shows its reconstruction result from the line drawing in Fig. 8(c), which is a failure.

The reader may wonder what happens if the 3D reconstruction is based on an arbitrary decomposition of a complex line drawing, instead of the proposed one. Fig. 10(b) shows such a decomposition from Fig. 8(c). Based on this decomposition, the 3D reconstruction result obtained by the scheme described in Section 2.3 is given in Fig. 10(c), which is a failure. The failure is caused by two reasons: (i) An arbitrary decomposition usually does not generate common objects, which makes the image regularities less meaningful for the 3D reconstruction. (ii) The gluing of 3D objects from the decomposition in Fig. 10(b) is difficult because of the irregular touches between the objects. The fine-tuning processing (see Section 2.3) cannot reduce the large distortion to an acceptable result.

Note that since our algorithm is not limited to manifolds, it can deal with line drawings with some or without hidden lines. The third line drawing in Fig. 9 is an example where some hidden lines are not drawn.

Most of the line drawings in this paper look tidy. This

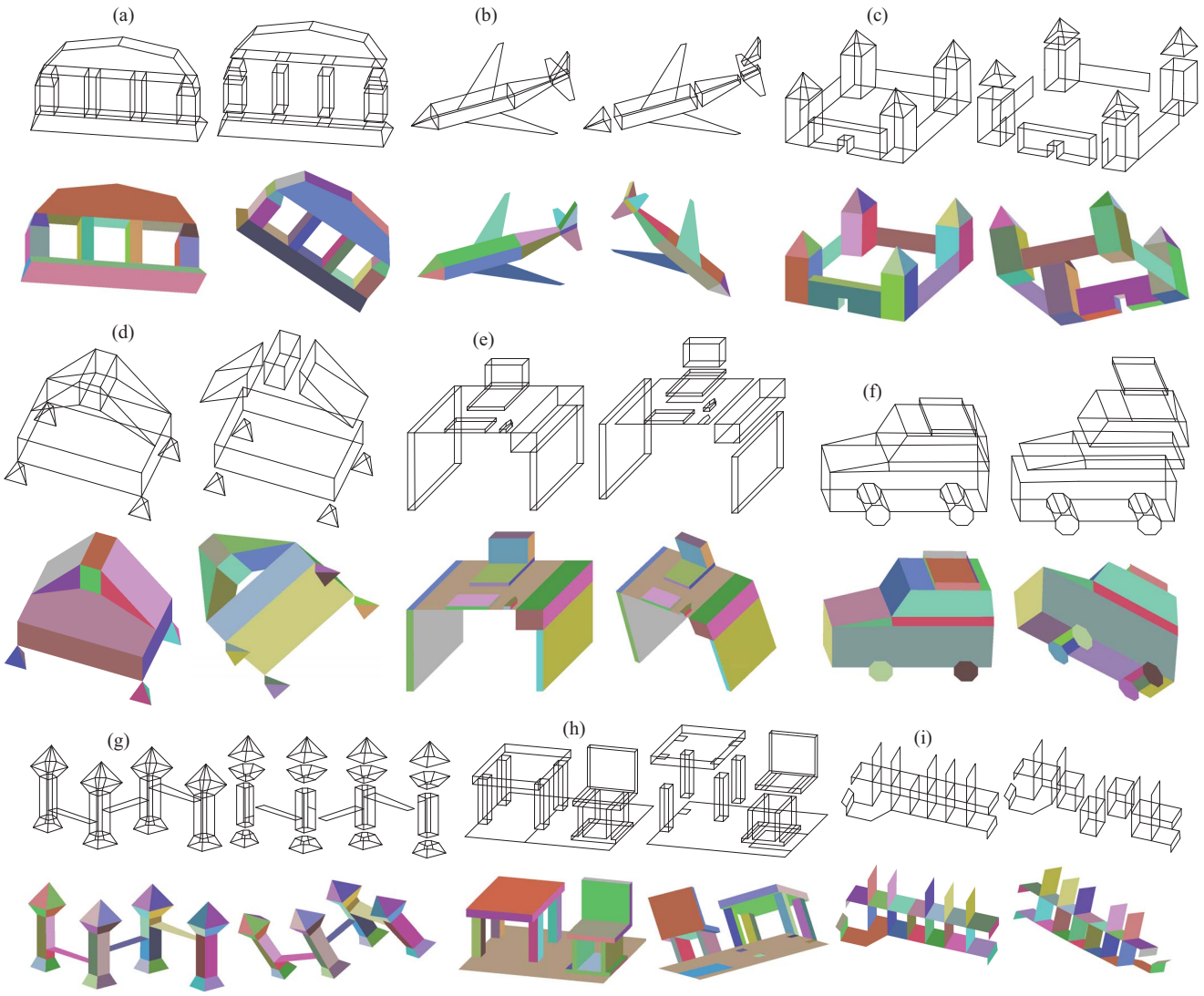


Fig. 8. Nine complex line drawings, their decompositions, and 3D reconstruction results in two views where different colors are used to denote the faces (better viewed on the screen).

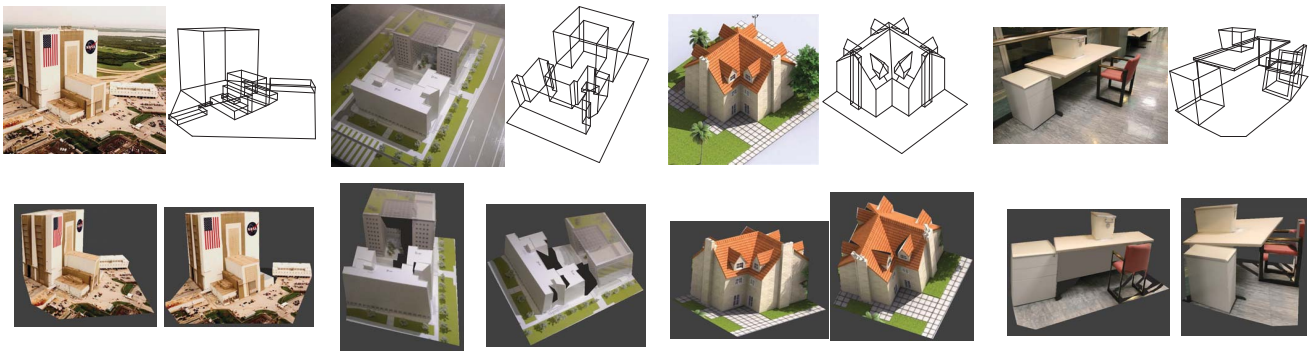
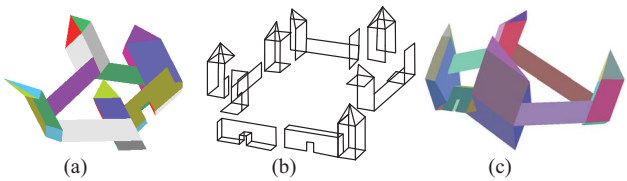
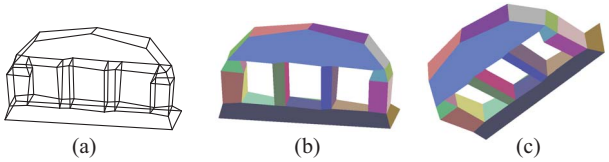


Fig. 9. Four images, the corresponding line drawings, and the reconstructed 3D objects with texture mapped, each shown in two views. The details can be seen by enlarging the figures on the screen.



(a) (b) (c)

Fig. 10. (a) A failed reconstruction by the algorithm in [7]. (b) An arbitrary decomposition of the line drawing in Fig. 8(c) without using our decomposition method. (c) Failed 3D reconstruction based on the decomposition in (b).



(a) (b) (c)

Fig. 11. (a) A line drawing with strong sketching errors. (b) (c) Two views of the successful reconstruction result by our algorithm.

is for easy observation of the objects. In fact, our algorithm is not sensitive to sketching errors. Take Fig. 8(a) as an example and assume it is an accurate projection of the 3D object. Then, random variations are generated with the Gaussian distribution $N(0, \sigma^2)$ on the 2D locations of the vertices. Fig. 11(a) is a resulting noisy line drawing with $\sigma = W/200$ where W is the width of the line drawing in Fig. 8(a). From Fig. 11, we see that even for this line drawing with strong sketching errors, our algorithm can still obtain the good reconstruction result.

Our algorithm is implemented in C++. The computational time includes two parts: line drawing decomposition and 3D reconstruction. The main computation is consumed by the second part. On average, a common PC takes about one minute to obtain the reconstruction from each of the line drawings in Figs. 8 and 9.

4. Conclusion

Previous algorithms of 3D object reconstruction from line drawings either deal with simple general objects or are limited to only manifolds (a subset of solids). In this paper, we have proposed a novel approach that can handle complex general objects, including manifolds, non-manifold solids, and non-solids. It decomposes a complex line drawing into simpler ones according to the degree of freedom of objects, which is based on the developed 3D object properties. After 3D objects are reconstructed from the decomposed line drawings, they are merged to form a complex object. We have shown a number of reconstruction examples with comparison to the best previous algorithm. The results indicate that our algorithm can tackle much more complex line drawings of general objects and is insensitive to sketching errors.

The future work includes (i) the correction of the distortions of 3D objects reconstructed from images caused by the perspective projection, and (ii) the extension of this work to objects with curved faces.

Acknowledgements

This work was supported by grants from Natural Science Foundation of China (No. 61070148), Science, Industry, Trade, and Information Technology Commission of Shenzhen Municipality, China (No. JC201005270378A), and Guangdong Innovative Research Team Program (No. 201001D0104648280). Jianzhuang Liu is the corresponding author.

References

[1] M. Clowes. On seeing things. *Artificial Intelligence*, 2:79–116, 1971.

[2] P. Debevec, C. Taylor, and J. Malik. Modeling and rendering architecture from photographs: A hybrid geometry- and image-based approach. *Proc. ACM SIGGRAPH*, pages 11–20, 1996.

[3] D. Jelinek and C. Taylor. Reconstruction of linearly parameterized models from single images with a camera of unknown focal length. *IEEE T-PAMI*, 23(7):767–773, 2001.

[4] D. E. LaCourse. *Handbook of Solid Modeling*. McGraw-Hill, 1995.

[5] Y. Leclerc and M. Fischler. An optimization-based approach to the interpretation of single line drawings as 3D wire frames. *IJCV*, 9(2):113–136, 1992.

[6] H. Lipson and M. Shpitalni. Optimization-based reconstruction of a 3d object from a single freehand line drawing. *Computer-Aided Design*, 28(7):651–663, 1996.

[7] J. Liu, L. Cao, Z. Li, and X. Tang. Plane-based optimization for 3D object reconstruction from single line drawings. *IEEE T-PAMI*, 30(2):315–327, 2008.

[8] J. Liu, Y. Chen, and X. Tang. Decomposition of complex line drawings with hidden lines for 3d planar-faced manifold object reconstruction. *IEEE T-PAMI*, 33(1):3–15, 2011.

[9] J. Liu, Y. Lee, and W. Cham. Identifying faces in a 2D line drawing representing a manifold object. *IEEE T-PAMI*, 24(12):1579–1593, 2002.

[10] J. Liu and X. Tang. Evolutionary search for faces from line drawings. *IEEE T-PAMI*, 27(6):861–872, 2005.

[11] T. Marill. Emulating the human interpretation of line-drawings as three-dimensional objects. *IJCV*, 6(2):147–161, 1991.

[12] P. R. J. Östergård. A new algorithm for the maximum-weight clique problem. *Nordic J. of Computing*, 8(4):424–436, Dec. 2001.

[13] H. Shimodaira. A shape-from-shading method of polyhedral objects using prior information. *IEEE T-PAMI*, 28(4):612–624, 2006.

[14] I. Shimshoni and J. Ponce. Recovering the shape of polyhedra using line-drawing analysis and complex reflectance models. *Computer Vision and Image Understanding*, 65(2):296–310, 1997.

[15] K. Shoji, K. Kato, and F. Toyama. 3-d interpretation of single line drawings based on entropy minimization principle. *CVPR*, 2001.

[16] M. Shpitalni and H. Lipson. Identification of faces in a 2d line drawing projection of a wireframe object. *IEEE T-PAMI*, 18(10), 1996.

[17] K. Sugihara. Machine interpretation of line drawings. *MIT Press*, 1986.

[18] A. Turner, D. Chapman, and A. Penn. Sketching space. *Computer and Graphics*, 24:869–879, 2000.

[19] F. Ulupinar and R. Nevatia. Shape from contour: straight homogeneous generalized cylinders and constant cross-section generalized cylinders. *IEEE T-PAMI*, 17(2):120–135, 1995.

[20] T. Xue, J. Liu, and X. Tang. Example-based 3d object reconstruction from line drawings. *CVPR*, 2012.