Plane-Based Optimization for 3D Object Reconstruction from Single Line Drawings

Jianzhuang Liu, *Senior Member*, *IEEE*, Liangliang Cao, Zhenguo Li, and Xiaoou Tang, *Senior Member*, *IEEE*

Abstract—In previous optimization-based methods of 3D planar-faced object reconstruction from single 2D line drawings, the missing depths of the vertices of a line drawing (and other parameters in some methods) are used as the variables of the objective functions. A 3D object with planar faces is derived by finding values for these variables that minimize the objective functions. These methods work well for simple objects with a small number N of variables. As N grows, however, it is very difficult for them to find the expected objects. This is because with the nonlinear objective functions in a space of large dimension N, the search for optimal solutions can easily get trapped into local minima. In this paper, we use the parameters of the planes that pass through the planar faces of an object as the variables of the objective function. This leads to a set of linear constraints on the planes of the object, resulting in a much lower dimensional null space where optimization is easier to achieve. We prove that the dimension of this null space is exactly equal to the minimum number of vertex depths that define the 3D object. Since a practical line drawing is usually not an exact projection of a 3D object, we expand the null space to a larger space based on the singular value decomposition of the projection matrix of the line drawing. In this space, robust 3D reconstruction can be achieved. Compared with the two most related methods, our method not only can reconstruct more complex 3D objects from 2D line drawings but also is computationally more efficient.

Index Terms—3D object reconstruction, degree of reconstruction freedom, line drawing, null space, singular value decomposition.

1 Introduction

In this paper, a line drawing is defined as a 2D projection of the edges and vertices of a 3D object in a generic view, with or without hidden edges and vertices. It is the simplest and most direct way of illustrating a 3D object. The human vision system has the ability to interpret 2D line drawings as 3D objects without difficulty. Emulating this ability is an important research topic for machine vision. Its applications include flexible sketching input for conceptual designers who tend to prefer pencil and paper over mouse and keyboard in current CAD systems, conversion of existing industrial wireframe models to solid models, interactive generation of 3D objects from images, and user-friendly query input interface for 3D object retrieval from large 3D object databases and the Web.

Interpretation of line drawings has spanned more than three decades. The earliest work is about line labeling and testing the correctness/realizability of a line drawing [1], [2], [3], [4], [5], [6], [7]. It does not give explicit 3D reconstruction from a line drawing. More recently, symbolic computation with the Grassmann-Cayley algebra is used to analyze the realizability of a polyhedral scene [8]. Three-dimensional reconstruction from multiple views of a wireframe model tries to recover a 3D CAD model from its three orthographic projections [2]. More information can be found from these orthographic views for the reconstruction task than from a single generic view. The work on face identification from single line drawings [9], [10], [11], [12], [13], [14], [15], [16], [17] discusses how to find from a line drawing the circuits that represent the faces of the object. When the face circuits of the object is known before reconstructing its 3D geometry, the complexity of the reconstruction can be reduced significantly.

The ultimate target of line drawing interpretation is to reconstruct 3D objects from 2D line drawings. To this end, the reconstruction is usually formulated as an optimization problem, and many methods have been proposed [9], [18], [19], [20], [21], [22], [23], [24], [25], [26], [27], [28], [29], [30], [31]. These methods are most related to the work in this paper and will be reviewed in more detail in the next section. In these optimization-based methods, the variables of the objective functions are the missing depths of the vertices in a line drawing (plus other parameters in some methods). A 3D planar object (that is, an object with planar faces only) is recovered by deriving the values for these variables that minimize the objective function. In general, these methods work well for simple objects with a small number N of the variables. As N grows, however, it is very difficult for them to find the expected objects. This is because with the nonlinear objective function in a high-dimensional space R^N, the search for optimal solutions can easily get trapped into local minima. Fig. 1b shows an example that the method in [21] cannot handle.

In this paper, we tackle the reconstruction problem in another way. Instead of the depths of the vertices, we use only the parameters of the planes that pass through the planar faces of an object as the variables of the objective function. This method leads to a set of linear constraints, resulting in a much lower dimensional null space where optimization is easier to achieve. We prove a theorem that shows that the

- *J. Liu, Z. Li, and X. Tang are with the Department of Information Engineering, The Chinese University of Hong Kong, Hong Kong. E-mail: {zliu, zgli5, xtang}@ie.cuhk.edu.hk.*
- *L. Cao is with the Department of Electrical and Computer Engineering, University of Illinois at Urbana-Champaign, Urbana, IL 61801. E-mail: cao4@uiuc.edu.*

Manuscript received 13 July 2006; revised 21 Jan. 2007; accepted 13 Mar. 2007; published online 3 May 2007.
Recommended for acceptance by R. Basri.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number TPAMI-0518-0706.
Digital Object Identifier no. 10.1109/TPAMI.2007.1172.

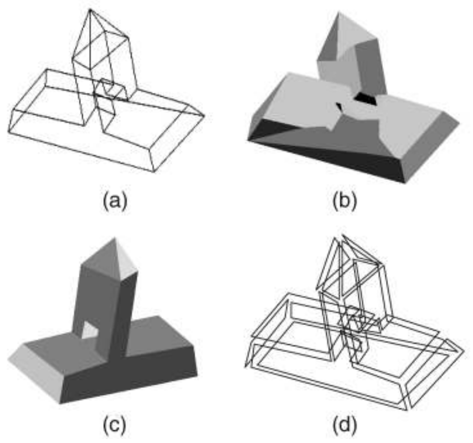

Fig. 1. (a) A line drawing. (b) A failed object reconstructed by the method in [21]. (c) A successfully reconstructed object. (d) Face circuits of the line drawing in (a).

minimum number of variables defining a 3D object that corresponds to an ideal line drawing is equal to the dimension of the null space with respect to this line drawing. We also develop an algorithm to find this minimum number (or its upper bound) from a practical line drawing. Due to vertex position errors in a practical line drawing, an expected 3D object may not be contained in the null space, so we expand it to a larger space based on this minimum number (or its upper bound) and the singular value decomposition (SVD) of the projection matrix of the line drawing. This new space is still much smaller than R^N, in which robust 3D reconstruction can be achieved. The objects reconstructed by our algorithm include polyhedra, nonmanifold solids, and nonsolid objects, with or without hidden edges and vertices given and with or without through holes.

The rest of this paper is organized as follows: We review the most related work in Section 2. Some assumptions are given in Section 3. Section 4 briefly discusses the previous formulations of the reconstruction problem, and then, Section 5 presents our formulation. Section 6 proves that the minimum number of variables defining a 3D object is equal to the dimension of the null space with respect to the line drawing. An algorithm for finding the upper bound of this minimum number is developed in Section 7. How to obtain the search space for the 3D reconstruction based on SVD is described in Section 8. A set of experimental results are provided in Section 9. Finally, Section 10 concludes this paper.

2 Related Work on 3D Reconstruction

So far, there has been little work on 3D reconstruction from single line drawings representing objects with curved faces. In this paper, we focus on 3D planar object reconstruction from single line drawings. To interpret 2D line drawings as 3D objects, an effective approach is to formulate the 3D reconstruction as an optimization problem with an objective function. In this research, it should be noted that there are always vertex position errors in a practical line drawing, which is either extracted from an image or drawn by a person. It is often impossible to find a 3D object whose projection is exactly the line drawing. This is because small deviations of some vertices from their precisely projected 2D positions may cause the 3D vertices on the same planar

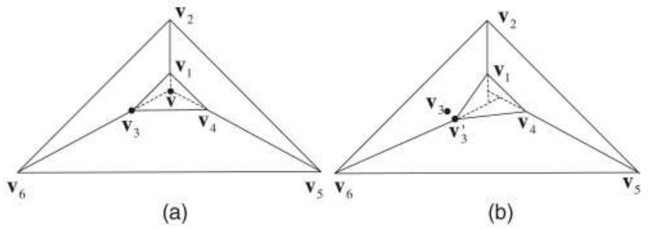

Fig. 2. (a) An ideal line drawing of a truncated pyramid. (b) A practical line drawing representing the same truncated pyramid.

face to be noncoplanar. Another typical example comes from a truncated pyramid, as shown in Fig. 2, where the extensions of the three edges $\overline{\mathbf{v}_2\mathbf{v}_1}$, $\overline{\mathbf{v}_5\mathbf{v}_4}$, and $\overline{\mathbf{v}_6\mathbf{v}_3}$ in the ideal line drawing (Fig. 2a) should meet at a point $\mathbf{v}$. However, if $\mathbf{v}_3$ is deviated a little to $\mathbf{v}_3'$ (Fig. 2b), the three lines $\overline{\mathbf{v}_2\mathbf{v}_1}$, $\overline{\mathbf{v}_5\mathbf{v}_4}$, and $\overline{\mathbf{v}_6\mathbf{v}_3'}$ will not meet at one point, resulting in no 3D truncated pyramids corresponding to this practical line drawing. This problem is termed *superstrictness*.

Assuming that the line drawings of polyhedra have been extracted from images and labeled correctly, the methods in [25], [26], and [27] all use shading information on the surfaces of the polyhedra to define the objective functions and constraints. The limitations of line labeling are that multiple consistent labeling solutions for one line drawing are possible [27], and most of the line drawings given in this paper cannot be labeled because the line labeling algorithms are generally suitable for polyhedra without hidden edges. These three methods also attempt to circumvent the superstrictness problem. In [25], Sugihara proposed a scheme to detect and remove redundant equations that cause the superstrictness. The drawback of Sugihara's method is that substantially large deviations may be introduced at the recovered vertices associated with the deleted equations, as pointed out by the authors in [7], [26], and [27]. In [26], Shimshoni and Ponce used a different method to deal with the superstrictness problem. They modeled vertex position errors with new variables without deleting any redundant equations. The main disadvantage of this method is that the number of variables in the optimization is too large even for a simple line drawing. For example, there are 51 variables for the truncated pyramid in Fig. 2a or 2b. In optimization, the likelihood of getting trapped into local minima increases as the number of variables increases. Both methods in [25] and [26] need to find a good initial shape that is close to the optimal one, but obtaining such a shape is itself a difficult reconstruction problem for a complex line drawing. In [27], Shimodaira treated each face of a polyhedron as a distinct object, and the gaps betweens faces are minimized during the optimization. The main limitations of this method are that it is applicable only to very simple polyhedra and we need to know in advance the horizontality and verticality of particular faces and the convex-concave properties of all the edges.

In the three methods mentioned above, the projected vertex positions of the reconstructed polyhedron are usually changed from their original 2D positions. Other methods [9], [18], [19], [20], [21], [22], [23], [24], including the one proposed in this paper, do the 3D reconstruction from line drawings only (without the use of shading information) and do not change the original 2D vertex positions. As a result, the faces are not required to be exactly planar.

Marill [18] presented his method based on a simple criterion: minimizing the standard deviation of the angles in the reconstructed object, which is called the MSDA principle. Motivated by MSDA, Brown and Wang [19] proposed to minimize the standard deviation of the segment magnitudes (MSDSM), and Shoji et al. [20] presented the criterion of minimizing the entropy of angle distribution (MEAD). MSDA, MSDSM, and MEAD can only recover simple objects from line drawings. Leclerc and Fischler's method considers not only MSDA but also the planarity constraint on the faces of the object [9]. This method performs better than MSDA, MSDSM, and MEAD. Later, Lipson and Shpitalni [21] extended Leclerc and Fischler's method by adding more constraints such as line parallelism, line verticality, isometry, corner orthogonality, skewed facial orthogonality, and skewed facial symmetry. This method can reconstruct more complex objects than Leclerc and Fischler's. Piquer et al. focused on recovering mirror symmetry objects using a symmetry constraint [22]. Based on the work in [18], [9], and [21], Turner et al. recovered simple planar 3D objects from scenes [28]. Shesh and Chen applied Lipson and Shpitalni's algorithm to their sketching system in [23]. Cao et al. focused on the 3D reconstruction of trihedral polyhedra from line drawings without hidden lines [24].

Among these previous methods [9], [18], [19], [20], [21], [22], [23], [24], [25], [26], [27], [28], Lipson and Shpitalni's [21] can handle the widest range of planar objects. Our method in this paper is most related to Leclerc and Fischler's [9] and Lipson and Shpitalni's [21] methods because 1) the three methods all use MSDA and planarity to formulate the reconstruction problem, 2) they neither need to label a line drawing nor use shading information, and 3) the original 2D vertex positions are not changed in the recovered 3D object.

3 Assumptions

In this paper, a line drawing is assumed to be an orthogonal projection of a 3D planar object in a generic view, with or without hidden lines and vertices. A line drawing in a generic view means that no two vertices appear at the same position and no two lines overlap in the 2D projection plane. If a line drawing is drawn with a mouse or tablet PC pen on the screen, the hidden lines and vertices can also be given, which allows the reconstruction of complete and more complex objects. When a recovered 3D object looks reasonable to human observers, the reconstruction is considered successful. Fig. 1c is a successful result obtained from the line drawing in Fig. 1a, but Fig. 1b is not.

Given a line drawing, its face topology is assumed to be known before the reconstruction of its 3D geometry. Here, the face topology denotes the set of circuits that represent all the faces of the 3D object. The line drawing in Fig. 1a has 15 faces, as shown in Fig. 1d. Finding the face topology from a line drawing is not a trivial problem due to the fact that the number of circuits is exponential in the number of edges in a general line drawing [16]. Fortunately, our previous work [15], [16] can be used for this purpose.

4 Formulations in the Most Related Methods

In this section, we briefly review the formulations of the reconstruction problem in the most related methods. These methods inflate a flat 2D line drawing by searching for suitable depths (z-coordinates) for all the vertices of the line drawing. These z-coordinates are obtained by minimizing an objective function that consists of some constraints such as MSDA, face planarity, and line parallelism. These constraints try to emulate the human perception of a 2D line drawing as a 3D object. The objective functions to be optimized take the following form:

$$\Phi(z_1, z_2, \dots, z_{N_v}) = \sum_{i=1}^{N_c} w_i \phi_i(z_1, z_2, \dots, z_{N_v}), \qquad (1)$$

where N_v denotes the number of the vertices of a line drawing, $z_1, z_2, \dots, z_{N_v}$ are the N_v z-coordinates to be determined, ϕ_i, $1 \leq i \leq N_c$, are the N_c constraints used, and w_i, $1 \leq i \leq N_c$, are weighting factors that give different weights to the constraints.

After $z_1, z_2, \dots, z_{N_v}$ are obtained by minimizing Φ, a 3D object is completely defined if the face topology is known and the line drawing is an orthogonal projection of the 3D object. The assumption of orthogonal projection makes the x and y-coordinates of the vertices of the 3D object available from the line drawing.

From our experiments, we consider that MSDA and face planarity are the two most important constraints for 3D object reconstruction from line drawings. Let ϕ_1 be the standard deviation of the angles in the reconstructed object:

$$\phi_1(z_1, z_2, \dots, z_{N_v}) = SD(\theta_1, \theta_2, \dots, \theta_k), \qquad (2)$$

where $\theta_1, \theta_2, \dots, \theta_k$ are all the angles formed by every two adjoining lines in the 3D object, and SD denotes the standard deviation. Minimizing ϕ_1 is the MSDA.

Let constraint ϕ_2 be the total distance of the vertices from their corresponding planes that pass through the faces of the 3D object. Minimizing ϕ_2 forces face planarity. Given the 3D coordinates of all the vertices, the parameters representing these planes can be obtained by a least square best fit plane algorithm [21]. Other constraints can be found from [19], [20], [21], [22], and [24].

From the definitions of ϕ_1 and ϕ_2, we can see that $\Phi(z_1, z_2, \dots, z_{N_v})$ is nonlinear. Minimizing it is carried out in the space R^{N_v}. Our experiments show that this optimization can easily get trapped into local minima when N_v is large with these formulations.

5 Our Formulation

We first define a new term and then present a new formulation of the reconstruction problem.

Definition 1. *Let a line drawing be a projection of a 3D object. The minimum number of depths (z-coordinates) that can uniquely define this 3D object is called the degree of reconstruction freedom (DRF) for the line drawing.*

Now, let us analyze the DRF for a simple line drawing shown in Fig. 3. In the previous depth-based optimization methods, the dimension of the search space is six since there are six vertices in the line drawing.[1] However, the DRF for this line drawing can be less than six with new geometric constraints taken into account.

1. In an implementation, we can arbitrarily specify the depth of one vertex, making the dimension of the search space be five. For simplicity of the description, we assume that the values of all the vertices can vary.

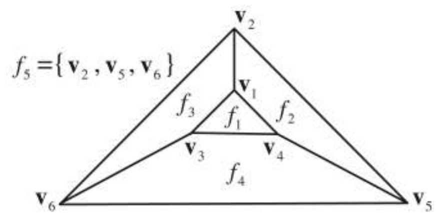

Fig. 3. The line drawing of a truncated pyramid with six vertices $\mathbf{v}_{1-6}$ and five faces f_{1-5}.

Here, we assume that the line drawing is a precise orthogonal projection of a truncated pyramid. Thus, all the 3D vertices on the same face are coplanar. For example, all the four vertices $\mathbf{v}_{3-6}$ are on the plane defined by $a_4x + b_4y + c_4 - z = 0$, which passes through the face $f_4 = \{\mathbf{v}_3, \mathbf{v}_4, \mathbf{v}_5, \mathbf{v}_6\}$. Next, we can show that the 3D object is defined if z_3, z_4, z_5, and z_1 are given.

When z_3, z_4, and z_5 are known, the 3D plane $a_4x + b_4y + c_4 - z = 0$ is defined. Then, z_6 can be calculated by $z_6 = a_4x_6 + b_4y_6 + c_4$. Since $\mathbf{v}_1$, $\mathbf{v}_3$, and $\mathbf{v}_6$ are known now, $\mathbf{v}_2$ is determined because it is on the plane defined by $\mathbf{v}_1$, $\mathbf{v}_3$, and $\mathbf{v}_6$. Thus, all the 3D vertices of the object are derived.

On the other hand, it is obvious that any three known depths cannot define a unique 3D object whose projection is this line drawing. Therefore, the DRF for this line drawing is four, which is smaller than six, the number of vertices. This example suggests that other formulations of the reconstruction problem in a lower space for optimization are possible.

In our formulation, instead of the depths of the vertices, we use the faces of a line drawing to be the variables of the objective function. More exactly, we use the parameters a_i, b_i, and c_i defining plane i by $a_ix + b_iy + c_i - z = 0$ to be the variables of the objective function, where plane i passes through face i. We call this formulation a *plane-based optimization* formulation.

In what follows, for simplicity, if a plane is defined by $ax + by + c - z = 0$, we may use the triple (a, b, c) to denote the plane. If this plane passes through a face, we may also use (a, b, c) to denote the face. Besides, a face may also be represented by the vertices it passes through, such as $f_5 = \{\mathbf{v}_2, \mathbf{v}_5, \mathbf{v}_6\}$ in Fig. 3.

Now, we show how to formulate the geometric constraints with the line drawing given in Fig. 3. In this line drawing, three faces, f_1, f_2, and f_3, pass through vertex $\mathbf{v}_1 = [x_1, y_1, z_1]^T$. Thus, we have these linear constraints $z_1 = a_1x_1 + b_1y_1 + c_1$, $z_1 = a_2x_1 + b_2y_1 + c_2$, and $z_1 = a_3x_1 + b_3y_1 + c_3$, which can be reduced to two equations by eliminating z_1: $a_1x_1 + b_1y_1 + c_1 - a_2x_1 - b_2y_1 - c_2 = 0$ and $a_2x_1 + b_2y_1 + c_2 - a_3x_1 - b_3y_1 - c_3 = 0$. For other vertices $\mathbf{v}_{2-6}$, we can obtain similar constraints without the z-coordinates. Rewriting all these linear equations in matrix form, we have

$$\mathbf{Pf} = \mathbf{0}, \tag{3}$$

where $\mathbf{P}$ is a 12×15 matrix in this example, and $\mathbf{f} = [a_1, b_1, c_1, a_2, b_2, c_2, a_3, b_3, c_3, a_4, b_4, c_4, a_5, b_5, c_5]^T$ consists of all the parameters of the five faces of the truncated pyramid.

For a general line drawing with N_v vertices and N_f faces, we can obtain the same matrix representation as in (3), with $\mathbf{f} = [a_1, b_1, c_1, a_2, b_2, c_2, \cdots, a_{N_f}, b_{N_f}, c_{N_f}]^T$, and $\mathbf{P}$ being a matrix of size $M \times (3N_f)$, where M depends on the structure of the line drawing. If only one face passes through a vertex, this vertex contributes nothing to $\mathbf{P}$; if n faces pass through it, it contributes $n - 1$ rows to $\mathbf{P}$. We call $\mathbf{P}$ and $\mathbf{f}$ the *projection matrix* and *face parameter vector* of the line drawing, respectively.

Usually, there are an infinite number of solutions to (3). However, the number of the independent solutions is limited. All the solutions to (3) compose a null space, denoted by $Null(\mathbf{P})$, with a dimension $D_{Null(\mathbf{P})} = 3N_f - Rank(\mathbf{P})$, where $Rank(\cdot)$ denotes the rank of the matrix [32]. We will see later that $D_{Null(\mathbf{P})} \ll N_v$ for a complex line drawing. This suggests that searching for the optimal $\mathbf{f}$ in $Null(\mathbf{P})$ would be much easier than the searching for the optimal $z_1, z_2, \ldots, z_{N_v}$ in R^{N_v}.

Given a line drawing, now, our target is to find the optimal $\mathbf{f}^* \in Null(\mathbf{P})$ such that $\mathbf{f}^*$ minimizes an objective function $\Psi(\mathbf{f})$, which is modified from $\Phi(z_1, z_2, \ldots, z_{N_v})$ defined in (1). Let

$$\Phi'(z_1, z_2, \ldots, z_{N_v}) = \Phi(z_1, z_2, \ldots, z_{N_v}) - w_2\phi_2(z_1, z_2, \ldots, z_{N_v}), \tag{4}$$

where $\phi_2(z_1, z_2, \ldots, z_{N_v})$ is the constraint of face planarity. Since all the depths can be calculated by $z_p = a_ix_p + b_iy_p + c_i$ if vertex $\mathbf{v}_p = [x_p, y_p, z_p]^T$ is on the plane (a_i, b_i, c_i), we can convert the depth-based representation $\Phi'(z_1, z_2, \ldots, z_{N_v})$ into a plane-based representation by

$$\Psi(\mathbf{f}) = \Phi'(z_1(\mathbf{f}), z_2(\mathbf{f}), \ldots, z_{N_v}(\mathbf{f})). \tag{5}$$

The optimization problem is now to

$$\text{minimize} \quad \Psi(\mathbf{f}), \tag{6}$$

$$\text{subject to} \quad \mathbf{f} \in Null(\mathbf{P}). \tag{7}$$

In $\Phi'(z_1, z_2, .., z_{N_v})$ (also $\Psi(\mathbf{f})$), the constraint ϕ_2 is removed because $\phi_2 = 0$ when $\mathbf{f} \in Null(\mathbf{P})$.

Given a line drawing, it is straightforward to obtain $\mathbf{P}$ and $Null(\mathbf{P})$, from which it seems that the problem defined in (6) and (7) can be solved easily. However, a practical line drawing is usually not a precise projection of a 3D object, causing the $Null(\mathbf{P})$ obtained from this line drawing to be smaller than the null space of the projection matrix of a corresponding ideal line drawing. Thus, the search in this smaller space may miss the expected solutions.

Let us analyze this problem in more detail. In what follows, a line drawing that is exactly the projection of a 3D object is called an *ideal* line drawing. A practical line drawing is not necessarily an ideal line drawing.

Suppose that LD_0 is an ideal line drawing of a 3D object and LD is a practical line drawing representing the same object but with some vertices deviating a little from their corresponding vertices in LD_0. Let $\mathbf{P}_0$ and $\mathbf{P}$ be the two projection matrices of LD_0 and LD, respectively. The dimension $D_{Null(\mathbf{P}_0)}$ of $Null(\mathbf{P}_0)$ and the dimension $D_{Null(\mathbf{P})}$ of $Null(\mathbf{P})$ can be obtained by

$$D_{Null(\mathbf{P}_o)} = 3N_f - Rank(\mathbf{P}_o), \tag{8}$$

$$D_{Null(\mathbf{P})} = 3N_f - Rank(\mathbf{P}) \tag{9}$$

with N_f being the number of faces of LD_0 (or LD).

We have found that the ranks of the projection matrices of most practical line drawings are larger than those of their corresponding ideal line drawings. Fig. 2 shows an example with the ideal line drawing LD_0 and the practical line

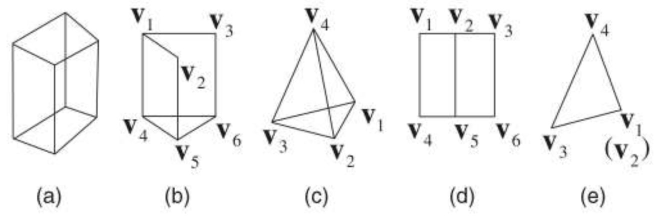

Fig. 4. Line drawings used to explain the changes of the ranks of their projection matrices.

drawing LD representing a truncated pyramid. Both $\mathbf{P}_0$ and $\mathbf{P}$ are of size 12×15, and we have $Rank(\mathbf{P}_0) = 11$ and $Rank(\mathbf{P}) = 12$. The constraint that the three lines $\overline{\mathbf{v}_2\mathbf{v}_1}$, $\overline{\mathbf{v}_5\mathbf{v}_4}$, and $\overline{\mathbf{v}_6\mathbf{v}_3}$ meet exactly at a point $\mathbf{v}$ in LD_0 implies that the 12 vectors, each of which is a row in $\mathbf{P}_0$, are not independent. When this constraint is not imposed as in LD, the 12 row vectors in $\mathbf{P}$ become independent, making the rank of $\mathbf{P}$ larger by one.

Another example is given in Fig. 4a. If the line drawing is an ideal one, the rank of its projection matrix will be 14. The constraint that all the six faces are planar in a 3D space implies that some of the row vectors of the projection matrix are dependent. However, if it is a practical line drawing, usually, this constraint cannot be satisfied, causing the rank to increase by one. In all the examples given in Section 9, the ranks increase.

There are line drawings for which the ranks of their projection matrices remain unchanged. Figs. 4b and 4c show two examples. They are always ideal line drawings, the projections of an infinite number of 3D objects, with the ranks being 5 and 8, respectively.

We have not found that the rank of the projection matrix of a line drawing representing an object in a generic view may reduce, but we do find that if a line drawing is the projection of an object in a special view, the rank may reduce. For example, if the two line drawings in Figs. 4b and 4c are degraded into the two shown in Figs. 4d and 4e, the ranks will reduce to 2 and 0, respectively. Since we consider line drawings that are the projections of 3D objects in generic views, it is our belief that the ranks of the projection matrices of these line drawings do not reduce. Although we have not been able to prove this observation, the reduction of the rank for a line drawing in a generic view, if it is indeed possible, does not affect our approach to 3D reconstruction (see Section 8 for the explanation).

Now, the questions are "What is the dimension of $Null(\mathbf{P}_0)$ when we do not have an ideal line drawing LD_0 but only a practical line drawing LD?" and "How can we find a space, from which an expected 3D object with respect to LD can be obtained?" We will answer these questions in the next three sections.

6 Relation between the Dimension of the Null Space and the DRF

In Section 5, we can find the DRF for the line drawing shown in Fig. 3 no matter whether it is an ideal line drawing of a 3D object or not. Let $\mathbf{P}_0$ be the projection matrix of an ideal line drawing LD_0. We will prove that the dimension of $Null(\mathbf{P}_0)$ is equal to the DRF for LD_0, which implies that even if we do not have this ideal line drawing LD_0 (but a practical line drawing LD), it is still possible to find the dimension of $Null(\mathbf{P}_0)$ from LD. This is the key to finding a space for the search for the expected 3D objects. Before giving the proof, we consider a lemma first.

Lemma 1. *Let $\mathbf{P}_0$ be the projection matrix of an ideal line drawing of a 3D object. Then, 1) some of the vertices of the 3D object satisfy $\mathbf{P}_0\mathbf{Q}^{-1}\mathbf{z} = \mathbf{0}$, where $\mathbf{z}$ is formed by the z-coordinates of these vertices, and $\mathbf{Q}$ is formed by the x and y-coordinates of these vertices; and 2) the dimensions of $Null(\mathbf{P}_0)$ and $Null(\mathbf{P}_0\mathbf{Q}^{-1})$ are the same.*

Proof.

1. Suppose that the 3D object has N_f faces. From face i, we can find three noncollinear vertices $\mathbf{v}_{ij} = [x_{ij}, y_{ij}, z_{ij}]^T$, $j = 1, 2, 3$, which define the plane (a_i, b_i, c_i) passing through this face. Thus, we have

$$z_{ij} = a_i x_{ij} + b_i y_{ij} + c_i, \; j = 1, 2, 3 \tag{10}$$

or

$$\begin{bmatrix} z_{i1} \\ z_{i2} \\ z_{i3} \end{bmatrix} = \begin{bmatrix} x_{i1} & y_{i1} & 1 \\ x_{i2} & y_{i2} & 1 \\ x_{i3} & y_{i3} & 1 \end{bmatrix} \begin{bmatrix} a_i \\ b_i \\ c_i \end{bmatrix} = \mathbf{Q}_i \begin{bmatrix} a_i \\ b_i \\ c_i \end{bmatrix}. \tag{11}$$

The vertices $\mathbf{v}_{i1}$, $\mathbf{v}_{i2}$, and $\mathbf{v}_{i3}$ are not collinear, neither are their projections on the plane where the line drawing is shown in a generic view. Thus, it holds that $\det(\mathbf{Q}_i) \neq 0$.

From each of the N_f faces, we can obtain an equation similar to (11). Combining all these N_f equations, we have

$$\mathbf{z} = [z_{11}, z_{12}, z_{13}, z_{21}, z_{22}, z_{23}, \cdots, z_{N_f1}, z_{N_f2}, z_{N_f3}]^T = \mathbf{Qf}, \tag{12}$$

where $\mathbf{f} = [a_1, b_1, c_1, a_2, b_2, c_2, \cdots, a_{N_f}, b_{N_f}, c_{N_f}]^T$, and

$$\mathbf{Q} = \begin{bmatrix} \mathbf{Q}_1 & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{Q}_2 & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{Q}_{N_f} \end{bmatrix}.$$

The fact that $\det(\mathbf{Q}_i) \neq 0$, $1 \leq i \leq N_f$, implies that $\det(\mathbf{Q}) \neq 0$ and $\mathbf{f} = \mathbf{Q}^{-1}\mathbf{z}$. Hence, from $\mathbf{P}_0\mathbf{f} = \mathbf{0}$ in (3), we have

$$\mathbf{P}_0\mathbf{Q}^{-1}\mathbf{z} = \mathbf{0}. \tag{13}$$

2. Since $\mathbf{Q}$ is invertible, it holds that $Rank(\mathbf{P}_0) = Rank(\mathbf{P}_0\mathbf{Q}^{-1})$ [32], which indicates that the dimensions of $Null(\mathbf{P}_0)$ and $Null(\mathbf{P}_0\mathbf{Q}^{-1})$ are both equal to $3N_f - Rank(\mathbf{P}_0)$. □

It is worth noting that in general, the $\mathbf{z}$ in (12) and (13) does not include all the z-coordinates of the vertices of the object. However, if $\mathbf{z}$ is a solution of (13), all the other z-coordinates not in $\mathbf{z}$ can be derived from their x and y-coordinates and $\mathbf{z}$, because all the 3D faces have been determined by $\mathbf{z}$. It also should be emphasized that some of the z-coordinates in $\mathbf{z}$ may be chosen more than once. In this case, (13) can be represented in another form:

$$\mathbf{P}_0\mathbf{Q}^{-1}\mathbf{M}\mathbf{z}' = \mathbf{0}, \tag{14}$$

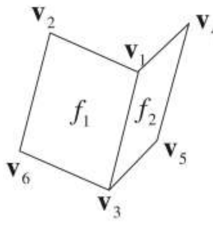

Fig. 5. A line drawing with two faces.

where $\mathbf{z} = \mathbf{M}\mathbf{z}'$, all the z-coordinates in $\mathbf{z}'$ represent different vertices, and the elements in $\mathbf{M}$ are either 1 or 0. This can be clear after the explanation with a line drawing given in Fig. 5. If $\mathbf{v}_1$, $\mathbf{v}_2$, and $\mathbf{v}_3$ are chosen to define face f_1 and $\mathbf{v}_4$, $\mathbf{v}_1$, and $\mathbf{v}_3$ are chosen to define face f_2, then $\mathbf{z} = [z_1, z_2, z_3, z_4, z_1, z_3]^T$, $\mathbf{z}' = [z_1, z_2, z_3, z_4]^T$, and

$$\mathbf{M} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}.$$

Theorem 1. *Let $\mathbf{P}_0$ be the projection matrix of an ideal line drawing of a 3D object. The dimension of $Null(\mathbf{P}_0)$ equals the DRF for this line drawing.*

Proof. If $\mathbf{z}$ is a solution of (13), then $\mathbf{z}'$ is also a solution of (14); if $\mathbf{z}'$ is a solution of (14), then $\mathbf{z} = \mathbf{M}\mathbf{z}'$ is also a solution of (13). This equivalence between (13) and (14) indicates that the dimensions of $Null(\mathbf{P}_0\mathbf{Q}^{-1})$ and $Null(\mathbf{P}_0\mathbf{Q}^{-1}\mathbf{M})$ must be the same. By Lemma 1, the dimensions of $Null(\mathbf{P}_0\mathbf{Q}^{-1}\mathbf{M})$ and $Null(\mathbf{P}_0)$ are also the same.

Let D be the dimension of $Null(\mathbf{P}_0)$ (or $Null(\mathbf{P}_0\mathbf{Q}^{-1}\mathbf{M})$). Then, there are D free variables in $\mathbf{z}'$, and the other variables in $\mathbf{z}'$ can be derived from these free variables [32]. From the discussion in the paragraph following the proof of Lemma 1, we know that the z-coordinates of all the vertices of the object can be obtained from these D free variables. Therefore, by definition, the DRF for the line drawing is equal to the dimension D of $Null(\mathbf{P}_0)$. □

7 Finding the DRF

Given a practical line drawing LD with its projection matrix $\mathbf{P}$, $Null(\mathbf{P})$ is often shrunk as discussed in Section 5. From Theorem 1, we further know that the dimension of the shrunk $Null(\mathbf{P})$ is less than the DRF for the line drawing. For such a null space, it is probable that the expected 3D objects will not be contained in this space. An example is given in Fig. 2b, where no 3D truncated pyramids are available in that $Null(\mathbf{P})$. However, if we can find the DRF for the line drawing, we can expand $Null(\mathbf{P})$ to a space with a dimension not less than the DRF so that the expected 3D objects can be obtained. In the following, we first define two terms, *partial line drawings* and *neighboring faces*, and then develop an algorithm to find an upper bound of the DRF.

Definition 2. *Denote a line drawing by $LD = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ and $\mathcal{E}$ are the sets of the vertices and edges of LD, respectively. A partial line drawing of LD is denoted by $LD_p = (\mathcal{V}_p, \mathcal{E}_p)$, with $\mathcal{V}_p \subseteq \mathcal{V}$, and $\mathcal{E}_p \subseteq \mathcal{E}$. A neighboring face of LD_p is a face that has only one edge or has two or more collinear edges in $\mathcal{E}_p$.*

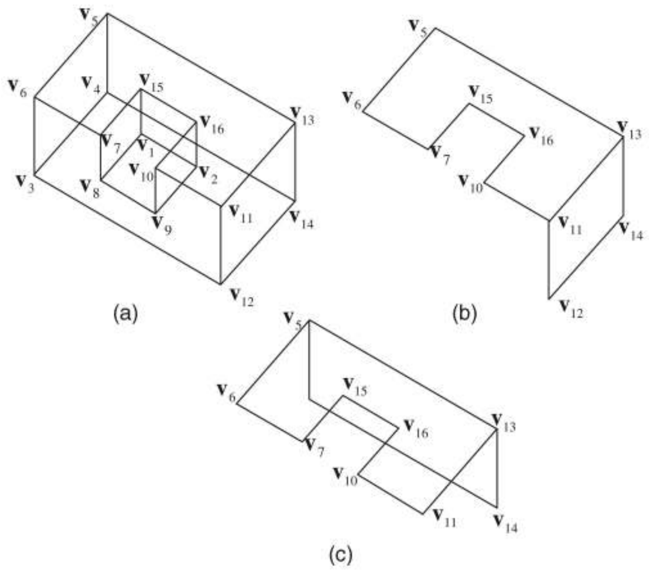

Fig. 6. (a) A line drawing LD. (b) A partial line drawing LD_{p1} of LD. (c) Another partial line drawing LD_{p2} of LD.

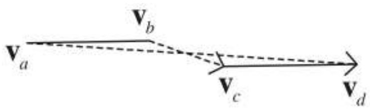

Fig. 7. Detection of collinearity.

By definition, a neighboring face of LD_p has not been defined completely by LD_p, but its degree of freedom is only one since it passes through one edge (or more than one collinear edge) in LD_p. Figs. 6b and 6c show two partial line drawings LD_{p1} and LD_{p2} of the line drawing LD in Fig. 6a. The face $\{\mathbf{v}_3, \mathbf{v}_4, \mathbf{v}_5, \mathbf{v}_6\}$ is a neighboring face of LD_{p1}, but the face $f = \{\mathbf{v}_3, \mathbf{v}_6, \mathbf{v}_7, \mathbf{v}_8, \mathbf{v}_9, \mathbf{v}_{10}, \mathbf{v}_{11}, \mathbf{v}_{12}\}$ is not since it has two noncollinear edges $\overline{\mathbf{v}_{10}\mathbf{v}_{11}}$ and $\overline{\mathbf{v}_{11}\mathbf{v}_{12}}$ in LD_{p1}. However, f is a neighboring face of LD_{p2} because all its edges in LD_{p2}, $\overline{\mathbf{v}_6\mathbf{v}_7}$ and $\overline{\mathbf{v}_{10}\mathbf{v}_{11}}$, are collinear.

The intersection of two planar faces in the 3D space is a line or collinear lines. The projections of these collinear lines are also collinear in the 2D line-drawing plane. However, two lines that should be collinear may not be exactly collinear in a practical line drawing. We have to allow some degree of inaccuracy for the detection of collinearity. Let the two vertices of edge $\overline{\mathbf{v}_a\mathbf{v}_b}$ be $\mathbf{v}_a$ and $\mathbf{v}_b$ and the two vertices of edge $\overline{\mathbf{v}_c\mathbf{v}_d}$ be $\mathbf{v}_c$ and $\mathbf{v}_d$, as shown in Fig. 7. Suppose that $\mathbf{v}_a$ and $\mathbf{v}_d$ are the two farthest vertices between the two edges. Then, we have two vectors $\overrightarrow{\mathbf{v}_a\mathbf{v}_d}$ and $\overrightarrow{\mathbf{v}_b\mathbf{v}_c}$. The two edges are considered as collinear if $\alpha_1 < \beta_1$ and $\alpha_2 < \beta_2$, where β_1 and β_2 are two angle thresholds, α_1 is the smaller angle between $\overline{\mathbf{v}_a\mathbf{v}_b}$ and $\overline{\mathbf{v}_c\mathbf{v}_d}$, and α_2 is the angle between $\overrightarrow{\mathbf{v}_a\mathbf{v}_d}$ and $\overrightarrow{\mathbf{v}_b\mathbf{v}_c}$. In our experiments, β_1 and β_2 are chosen to be 8 degrees and 5 degrees, respectively.

Algorithm 1. (Finding an upper bound ub of the DRF for a line drawing $LD = (\mathcal{V}, \mathcal{E})$. Let $LD_p = (\mathcal{V}_p, \mathcal{E}_p)$ be a partial line drawing of LD.)

```
1. Initialization:
  (a) $F \leftarrow$ all the faces of $LD$;
  (b) Select randomly one face $f_0 \in F$; $F \leftarrow F \setminus \{f_0\}$;
      $\mathcal{V}_p \leftarrow$ all the vertices of $f_0$; $\mathcal{E}_p \leftarrow$ all the edges of $f_0$;
  (c) $ub \leftarrow 3$;
```

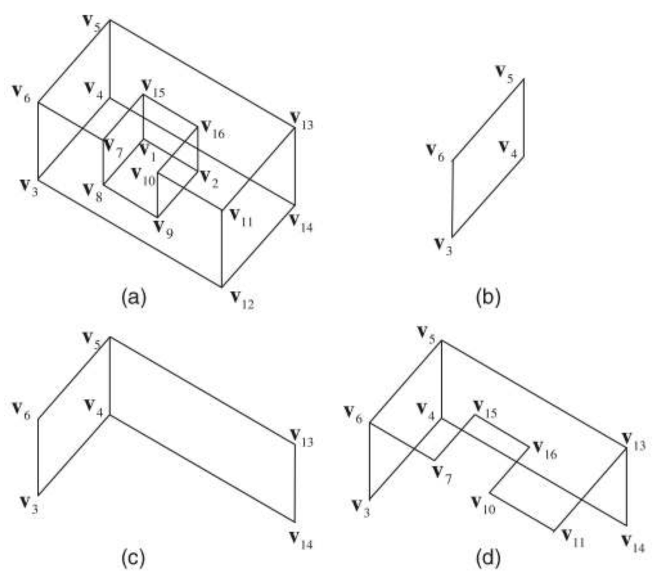

Fig. 8. (a) A line drawing. (b) The first selected face with $ub = 3$. (c) LD_p with two faces and $ub = 4$. (d) LD_p with three faces and $ub = 4$.

```
2. Select randomly one face $f_1 \in F$ that is a neighboring face
   of $LD_p$; $F \leftarrow F \setminus \{f_1\}$; $\mathcal{V}_p \leftarrow \mathcal{V}_p \cup \{\text{all the vertices of } f_1\}$;
   $\mathcal{E}_p \leftarrow \mathcal{E}_p \cup \{\text{all the edges of } f_1\}$; $ub \leftarrow ub + 1$;
3. if $F \neq \emptyset$ then for every face $f_2 \in F$ do
   (a)  if $f_2$ has two noncollinear edges in $\mathcal{E}_p$ then
   (b)     $F \leftarrow F \setminus \{f_2\}$; $\mathcal{V}_p \leftarrow \mathcal{V}_p \cup \{\text{all the vertices of } f_2\}$;
           $\mathcal{E}_p \leftarrow \mathcal{E}_p \cup \{\text{all the edges of } f_2\}$; goto Step 3;
4. if $F \neq \emptyset$ goto Step 2 else return $ub$.
```

The idea of this algorithm is to expand LD_p gradually by adding the vertices and edges of the selected faces into LD_p until $LD_p = LD$. With the first selected face f_0, we set $ub = 3$ in Step 1c because the three vertices defining f_0 can vary independently. In Step 2, f_1 is a neighboring face of LD_p, meaning that it has been defined by LD_p partially and its remaining degree of freedom is one. In other words, with one more vertex of f_1 not in the current LD_p, f_1 can be defined completely. Therefore, ub is increased by one. In Step 3, whenever the vertices and edges of a new face are added to LD_p, the remaining faces in F are checked again to see whether or not there exists such faces that have been defined completely by LD_p. If yes, the vertices and edges of these faces are added to LD_p. In this case, ub is not changed since these faces have no freedom in the 3D space.

Our experiments show that the found ub is exactly the DRF for a line drawing in many cases. In other cases, however, it is an upper bound of the DRF. By definition, the DRF is the minimum number of the z-coordinates that can define an object whose projection is the line drawing. When the algorithm stops, it finds the number ub of the z-coordinates that can define the object. Thus, ub is not less than the DRF for this line drawing. Through the following two examples, we demonstrate how the algorithm may or may not find a value of ub equal to the DRF.

The first example is given in Fig. 8. Suppose that the face in Fig. 8b is selected initially. It is also the initial LD_p with $ub = 3$. In Step 2, if the face $\{\mathbf{v}_4, \mathbf{v}_5, \mathbf{v}_{13}, \mathbf{v}_{14}\}$ is chosen as a neighboring face of the current LD_p, adding the vertices and edges of this face to LD_p results in a new LD_p as shown in Fig. 8c with ub increased by one. In Step 3, the algorithm searches for faces that have been defined completely by the current LD_p. Suppose that the face $\{\mathbf{v}_5, \mathbf{v}_6, \mathbf{v}_7, \mathbf{v}_{15}, \mathbf{v}_{16}, \mathbf{v}_{10}, \mathbf{v}_{11}, \mathbf{v}_{13}\}$ is checked now. Since it has three noncollinear vertices, $\mathbf{v}_5$, $\mathbf{v}_6$, and $\mathbf{v}_{13}$, in the current LD_p, this face has been defined completely. We then add all its vertices and edges to the LD_p in Fig. 8c, forming a new LD_p as indicated in Fig. 8d without increasing ub. Next, Step 3 is repeated again until no new faces can be found that have been defined completely. For this example, by repeating Step 3, all the other faces not in the LD_p in Fig. 8d can be found completely defined. Therefore, $ub = 4$ finally. The value of $ub = 4$ is the DRF for this line drawing because it is impossible to use only three vertices to determine the object.

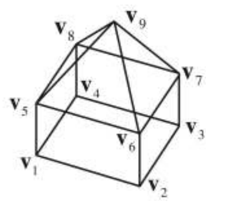

Fig. 9. A line drawing used to show that the found ub may not be the DRF.

Fig. 9 shows a line drawing, for which Algorithm 1 may return a ub that is larger than the DRF for this line drawing. If the first face selected by Algorithm 1 in Step 1b is $\{\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3, \mathbf{v}_4\}$, it is easy to verify that the algorithm will return $ub = 5$. In another case, if the selected faces by the algorithm are in the order

$$\begin{aligned}\{\mathbf{v}_6, \mathbf{v}_7, \mathbf{v}_9\} &\rightarrow \{\mathbf{v}_6, \mathbf{v}_9, \mathbf{v}_5\} \rightarrow \{\mathbf{v}_5, \mathbf{v}_8, \mathbf{v}_9\} \rightarrow \{\mathbf{v}_7, \mathbf{v}_8, \mathbf{v}_9\}\\ &\rightarrow \{\mathbf{v}_2, \mathbf{v}_3, \mathbf{v}_7, \mathbf{v}_6\} \rightarrow \{\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_6, \mathbf{v}_5\}\\ &\rightarrow \{\mathbf{v}_3, \mathbf{v}_4, \mathbf{v}_8 \mathbf{v}_7\} \rightarrow \{\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3, \mathbf{v}_4\}\\ &\rightarrow \{\mathbf{v}_1, \mathbf{v}_4, \mathbf{v}_8, \mathbf{v}_5\},\end{aligned}$$

then $ub = 6$. In fact, the DRF for this line drawing is 5. This is because the rectangular block itself has a DRF $= 4$, which will be increased by one after the pyramid is added to the block.

We hope to find a ub that is as close to the DRF as possible. Due to the random selection of faces in the algorithm, it is possible that the algorithm returns different ub's in two executions when handling the same line drawing. Since the algorithm is fast enough, we can run it n times. Then, we choose $ub = \min\{ub_1, ub_2, \ldots, ub_n\}$. In this way, we have a larger probability of obtaining the DRF for a line drawing. For all the line drawings in our experiments, the smallest upper bounds found by Algorithm 1 are all the DRFs for their corresponding line drawings when we set $n = 10$.

It should be mentioned that there exist line drawings whose DRFs cannot be found by applying Algorithm 1 many times. One example is a line drawing that is obtained by adding one edge to the line drawing shown in Fig. 9, connecting vertices $\mathbf{v}_2$ and $\mathbf{v}_4$. The DRF for this new line drawing is still 5, but Algorithm 1 can only find an upper bound $ub = 6$.

In [1] and [3], Sugihara and Whiteley developed a formula that can be used to compute the degree of freedom directly from a line drawing if the line drawing represents a *generically reconstructible* object. Whiteley showed that if the condition for a line drawing to be generically reconstructible is satisfied, the degree of freedom of the object is $|V| + 3|F| - |R|$, where

$|V|$, $|F|$, and $|R|$ denote the numbers of vertices, faces, and incidence pairs of the line drawing, respectively. However, many common objects (such as a hexahedron) are not regarded as generically reconstructible. For most of the line drawings in this paper, this formula $|V|+3|F|-|R|$ cannot be used to obtain the degrees of freedom because they are considered as nongenerically reconstructible by Sugihara and Whiteley's scheme.

To finish this section, we analyze the computational complexity of Algorithm 1. Let a line drawing LD have N_v vertices, N_e edges, and N_f faces. Assume that every face has less than k edges. The main computation is carried out by Steps 2 and 3. In Step 2, the algorithm tests whether a given face is a neighboring face of LD_p by looking for an edge in LD_p that is also an edge in the face, which is bounded by $O(kN_e)$. Thus, the computation at this step is $O(kN_fN_e)$. In Step 3a, to test if f_2 has three noncollinear vertices in $\mathcal{V}_p$, we first find all the vertices in f_2 that are also in $\mathcal{V}_p$ and then check if there are three noncollinear vertices among them. The former and the latter are bounded by $O(kN_v)$ and $O(k)$, respectively. Therefore, Step 3a takes less than $O(kN_v)$ time. Step 3b needs less than $O(N_f+k)$ time. Then, the computation of one execution of Step 3 is bounded by $O((kN_v+N_f+k)N_f)=O(kN_fN_v)$. Note that when Step 2 or 3 is passed once, one face is deleted from F. Therefore, the complexity of the algorithm is bounded by $O(l_1kN_fN_e+l_2kN_fN_v)$, where l_1 and l_2 are the numbers of times Steps 2 and 3 are visited, respectively, with $l_1+l_2=N_f-1$. Here, l_1+l_2 equals N_f-1 instead of N_f because Step 1b already removes one face from F. Assume that $N_e \approx N_v$, and k is a constant. Then, the complexity of the algorithm is bounded by $O(N_f^2N_v)$.

8 Finding a Space for Optimization

It has been emphasized that a practical line drawing LD is usually not a precise projection of a 3D object, and the solutions to $\mathbf{Pf}=\mathbf{0}$ may not contain one corresponding to an expected 3D object, where $\mathbf{P}$ is the projection matrix of LD, and $\mathbf{f}$ is the face parameter vector. This is because $Null(\mathbf{P})$ has been shrunk compared with $Null(\mathbf{P}_0)$, that is, $D_{Null(\mathbf{P})} < D_{Null(\mathbf{P}_0)}$, where $\mathbf{P}_0$ is the projection matrix of an ideal line drawing LD_0 representing the same object as LD does, and $D_{Null(\mathbf{P})}$ and $D_{Null(\mathbf{P}_0)}$ denote the dimensions of $Null(\mathbf{P})$ and $Null(\mathbf{P}_0)$, respectively. Since the objects whose projections are LD_0 exist in $Null(\mathbf{P}_0)$ with a dimension $D_{Null(\mathbf{P}_0)}$, it is quite reasonable to search for an expected object in a space with a dimension equal to $D_{Null(\mathbf{P}_0)}$ or larger. Although we do not have an ideal line drawing in general, fortunately, we have proved that $D_{Null(\mathbf{P}_0)}$ equals the DRF for LD_0, and developed Algorithm 1 to find an upper bound of the DRF from LD. By running Algorithm 1 n (say, 10) times, the minimum upper bound ub is often equal to $D_{Null(\mathbf{P}_0)}$. Given LD only, whether or not it is an ideal line drawing, we believe that searching for an expected object in a space spanned from $Null(\mathbf{P})$ is the best way for 3D reconstruction, and the dimension of the space $\geq ub$ is necessary.

As discussed in Section 5, it is possible that $Rank(\mathbf{P}) = Rank(\mathbf{P}_0)$ for some line drawings (say, the two in Figs. 4b and 4c), resulting in $D_{Null(\mathbf{P})} = D_{Null(\mathbf{P}_0)}$. When LD represents an object in a generic view, we have not found that $Rank(\mathbf{P}) < Rank(\mathbf{P}_0)$ leads to $D_{Null(\mathbf{P})} > D_{Null(\mathbf{P}_0)}$ by (8) and (9). If it did happen that $D_{Null(\mathbf{P})} > ub \geq D_{Null(\mathbf{P}_0)}$, we would not expand $Null(\mathbf{P})$ but search for a 3D object in $Null(\mathbf{P})$ directly. In Sections 8.1 and 8.2, we first discuss how to find the optimal spaces under the condition that $D_{Null(\mathbf{P})} \leq ub$ and then give the algorithm for 3D reconstruction.

8.1 Spaces Obtained by SVD

Now, we want to find a space with a dimension $\geq ub$, which is expanded from $Null(\mathbf{P})$. This space can be obtained with the help of the SVD of $\mathbf{P}$. By SVD, we obtain

$$\mathbf{P} = \mathbf{USV}^T, \tag{15}$$

where $\mathbf{U} = [\mathbf{u}'_1, \mathbf{u}'_2, \cdots, \mathbf{u}'_{3N_f}]$ is a column-orthogonal matrix, $\mathbf{S} = \mathrm{diag}(\delta_1, \delta_2, \cdots, \delta_{3N_f})$ is a diagonal matrix with $\delta_1 \geq \delta_2 \geq \cdots \geq \delta_{3N_f} \geq 0$, $\mathbf{V} = [\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_{3N_f}]$ is an orthogonal matrix, and N_f is the number of faces of LD. Furthermore, it follows from [32] that

$$\mathbf{P}[\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_{3N_f}] = [\delta_1\mathbf{u}'_1, \delta_2\mathbf{u}'_2, \cdots, \delta'_{3N_f}\mathbf{u}'_{3N_f}], \tag{16}$$

$$\delta_1 \geq \delta_2 \geq \cdots \geq \delta_{Rank(\mathbf{P})} > 0, \tag{17}$$

$$\delta_{Rank(\mathbf{P})+1} = \delta_{Rank(\mathbf{P})+2} = \cdots = \delta_{3N_f} = 0, \tag{18}$$

$$Null(\mathbf{P}) = \mathrm{span}\{\mathbf{u}_{Rank(\mathbf{P})+1}, \mathbf{u}_{Rank(\mathbf{P})+2}, \cdots, \mathbf{u}_{3N_f}\}, \tag{19}$$

$$D_{Null(\mathbf{P})} = 3N_f - Rank(\mathbf{P}), \tag{20}$$

where $\mathrm{span}\{\mathbf{u}_{Rank(\mathbf{P})+1}, \mathbf{u}_{Rank(\mathbf{P})+2}, \cdots, \mathbf{u}_{3N_f}\}$ denotes the space spanned by the set of vectors:

$$\{\mathbf{u}_{Rank(\mathbf{P})+1}, \mathbf{u}_{Rank(\mathbf{P})+2}, \cdots, \mathbf{u}_{3N_f}\}.$$

Let

$$S_{Null(\mathbf{P})} = \{\mathbf{u}_{Rank(\mathbf{P})+1}, \mathbf{u}_{Rank(\mathbf{P})+2}, \cdots, \mathbf{u}_{3N_f}\}, \tag{21}$$

$$H_1 = \mathrm{span}\{\{\mathbf{u}_{Rank(\mathbf{P})}\} \cup S_{Null(\mathbf{P})}\}, \tag{22}$$

$$H_2 = \mathrm{span}\{\{\mathbf{u}_{Rank(\mathbf{P})-1}, \mathbf{u}_{Rank(\mathbf{P})}\} \cup S_{Null(\mathbf{P})}\}, \cdots, \tag{23}$$

$$\begin{aligned} H_i = \mathrm{span}\{\{&\mathbf{u}_{Rank(\mathbf{P})-i+1}, \mathbf{u}_{Rank(\mathbf{P})-i+2}, \cdots, \\ &\mathbf{u}_{Rank(\mathbf{P})}\} \cup S_{Null(\mathbf{P})}\}, \cdots, \end{aligned} \tag{24}$$

$$H_{Rank(\mathbf{P})} = \mathrm{span}\{\{\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_{Rank(\mathbf{P})}\} \cup S_{Null(\mathbf{P})}\}. \tag{25}$$

Since $\{\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_{3N_f}\}$ is a set of orthogonal vectors, we have

$$Null(\mathbf{P}) \subset H_1 \subset H_2 \subset \cdots \subset H_{Rank(\mathbf{P})} = R^{3N_f} \tag{26}$$

and the dimension of H_i

$$D_{H_i} = D_{Null(\mathbf{P})} + i, \quad 1 \leq i \leq Rank(\mathbf{P}). \tag{27}$$

Now, if we want to expand $Null(\mathbf{P})$ to a larger space with a dimension equal to $D_{Null(\mathbf{P})}+i$, $1 \leq i \leq Rank(\mathbf{P})$, we can choose the H_i defined in (24).

An infinite number of spaces of a fixed dimension can be chosen to search for 3D objects from a given line drawing. The question is "Which space is the best one?" We claim that the space spanned as in (24) is the best when the dimension of H_i is fixed.

Without loss of generality, suppose that $D_{Null(\mathbf{P})} = ub-1$, and we want to expand $Null(\mathbf{P})$ to a space $H_{k_1,k_2,\ldots,k_{Rank(\mathbf{P})}}$ of dimension $D_{Null(\mathbf{P})}+1$. An infinite number of such spaces can be obtained by setting different values of k_i, $1 \leq i \leq Rank(\mathbf{P})$, $\sum_{i=1}^{Rank(\mathbf{P})} k_i^2 = 1$, $k_i \in R$, in

$$\begin{aligned} H_{k_1,k_2,\ldots,k_{Rank(\mathbf{P})}} = \mathrm{span}\{\{&k_1\mathbf{u}_1 + k_2\mathbf{u}_2 + \cdots + \\ &k_{Rank(\mathbf{P})}\mathbf{u}_{Rank(\mathbf{P})}\} \cup S_{Null(\mathbf{P})}\}. \end{aligned} \tag{28}$$

We claim that the H_1 defined in (22) among these spaces is the best in terms of *similarity* to $Null(\mathbf{P}_0)$.

For any $\mathbf{f} \in Null(\mathbf{P}_0)$, it holds that $\mathbf{P}_0\mathbf{f} = \mathbf{0}$. Since LD is an approximation of LD_0, we relax the requirement that $\mathbf{Pf} = \mathbf{0}$ for $\mathbf{f} \in H_{k_1,k_2,\ldots,k_{Rank(\mathbf{P})}}$. Although we do not require $\mathbf{Pf} = \mathbf{0}$, $\mathbf{f} \in H_{k_1,k_2,\ldots,k_{Rank(\mathbf{P})}}$, we do hope to find such a space with $\mathbf{Pf}$ as close to zero as possible, because a large value of $\|\mathbf{Pf}\|$ implies that the planarity constraint imposed on the faces of the 3D object is violated badly.

Definition 3. *Let H_a and H_b be the two spaces defined in (28). If*

$$\sup_{\mathbf{f}\in H_a,\|\mathbf{f}\|\le 1} \|\mathbf{Pf}\| < \sup_{\mathbf{f}\in H_b,\|\mathbf{f}\|\le 1} \|\mathbf{Pf}\|,$$

H_a is called more similar to $Null(\mathbf{P}_0)$ than H_b.

Let the size of $\mathbf{P}$ be $m \times n$. Then the geometrical explanation of $\sup_{\mathbf{f}\in H_a,\|\mathbf{f}\|\le 1} \|\mathbf{Pf}\| = K_a$ is that the linear transformation $\mathbf{P}$ maps the points inside the unit hypersphere in H_a to points bounded by the hypersphere with its radius $= K_a$ in R^n. The transformation $\mathbf{P}_0$ maps all the points in $Null(\mathbf{P}_0)$ to $\mathbf{0}$ in R^n (bounded by the hypersphere with its radius $= 0$). In the definition, H_a is called more similar to $Null(\mathbf{P}_0)$ since K_a is closer to 0 than $K_b = \sup_{\mathbf{f}\in H_b,\|\mathbf{f}\|\le 1} \|\mathbf{Pf}\|$.

Theorem 2. *H_1 is most similar to $Null(\mathbf{P}_0)$ among all the spaces defined in (28).*

Proof. For any $\mathbf{f} \in H_1$ and $\|\mathbf{f}\| \le 1$, we have $\mathbf{f} = \mathbf{f}_1 + \mathbf{f}_2$, where $\mathbf{f}_1 \in Null(\mathbf{P})$, and $\mathbf{f}_2 = \lambda\mathbf{u}_{Rank(\mathbf{P})}$, $\lambda \in R$, $|\lambda| \le 1$. Thus, $\mathbf{Pf} = \mathbf{Pf}_2 = \lambda\mathbf{Pu}_{Rank(\mathbf{P})} = \lambda\delta_{Rank(\mathbf{P})}\mathbf{u}'_{Rank(\mathbf{P})}$. Since $\mathbf{u}'_i$, $1 \le i \le Rank(\mathbf{P})$, are a set of orthogonal unit vectors, we have $\|\mathbf{Pf}\|^2 = \lambda^2\delta^2_{Rank(\mathbf{P})}$, and further,

$$\sup_{\mathbf{f}\in H_1,\|\mathbf{f}\|\le 1} \|\mathbf{Pf}\| = \delta_{Rank(\mathbf{P})}.$$

Now, for any space $H_{k_1,k_2,\ldots,k_{Rank(\mathbf{P})}}$ except H_1, we have $\sum_{i=1}^{Rank(\mathbf{P})} k_i^2 = 1$ and $|k_{Rank(\mathbf{P})}| \ne 1$. Let $\mathbf{f} = \sum_{i=1}^{Rank(\mathbf{P})} k_i\mathbf{u}_i$. Then, $\mathbf{f} \in H_{k_1,k_2,\ldots,k_{Rank(\mathbf{P})}}$, $\mathbf{f} \notin H_1$, $\|\mathbf{f}\| = 1$, $\mathbf{Pf} = \sum_{i=1}^{Rank(\mathbf{P})} k_i \mathbf{Pu}_i = \sum_{i=1}^{Rank(\mathbf{P})} k_i\delta_i\mathbf{u}'_i$, and $\|\mathbf{Pf}\|^2 = \sum_{i=1}^{Rank(\mathbf{P})} k_i^2\delta_i^2$. From (17), we know that $\delta_1 \ge \delta_2 \ge \cdots \ge \delta_{Rank(\mathbf{P})} > 0$. When $\delta_1 \ge \delta_2 \ge \cdots \ge \delta_{Rank(\mathbf{P})-1} > \delta_{Rank(\mathbf{P})}$, it follows that

$$\|\mathbf{Pf}\|^2 > \sum_{i=1}^{Rank(\mathbf{P})} k_i^2\delta^2_{Rank(\mathbf{P})} = \delta^2_{Rank(\mathbf{P})}$$

and, therefore,

$$\sup_{\mathbf{f}\in H_{k_1,k_2,\ldots,k_{Rank(\mathbf{P})}},\|\mathbf{f}\|\le 1} \|\mathbf{Pf}\| > \sup_{\mathbf{f}\in H_1,\|\mathbf{f}\|\le 1} \|\mathbf{Pf}\|.$$

When $\delta_1 \ge \delta_2 \ge \cdots \ge \delta_j > \delta_{j+1} = \delta_{j+2} = \cdots = \delta_{Rank(\mathbf{P})}$ or $\delta_1 = \delta_2 = \cdots = \delta_{Rank(\mathbf{P})}$, similarly, we can verify that

$$\sup_{\mathbf{f}\in H_{k_1,k_2,\ldots,k_{Rank(\mathbf{P})}},\|\mathbf{f}\|\le 1} \|\mathbf{Pf}\| \ge \sup_{\mathbf{f}\in H_1,\|\mathbf{f}\|\le 1} \|\mathbf{Pf}\|.$$

Thus, H_1 is most similar to $Null(\mathbf{P}_0)$ among all the spaces defined in (28). □

8.2 An Algorithm for 3D Reconstruction

Using Algorithm 1, we obtain an upper bound ub of the DRF for a line drawing. If we expand $Null(\mathbf{P})$ to H_i with $i = ub - D_{Null(\mathbf{P})}$, then $D_{H_i} = D_{Null(\mathbf{P}_0)}$ when $ub = \text{the DRF}$ ($= D_{Null(\mathbf{P}_0)}$). Since $\mathbf{P}$ is usually not the projection matrix of an ideal line drawing, searching for $\mathbf{f}$ in H_i may be too limited to obtain the expected 3D objects. Our experiments show that if we search for a 3D object in H_i whose dimension equals its DRF, the object is often inflated badly (that is, some noncoplanar faces result in being coplanar and/or some coplanar vertices deviate much from planarity); however, if $Null(\mathbf{P})$ is expanded to a larger space H_i with $i = ub - D_{Null(\mathbf{P})} + 1$ or $i = ub - D_{Null(\mathbf{P})} + 2$, we can obtain satisfactory results. Therefore, we choose $i = ub - D_{Null(\mathbf{P})} + 2$ for all the experiments given in the next section. Another reason to choose it instead of $i = ub - D_{Null(\mathbf{P})}$ is given as follows: If the collinearity between two lines in a face is detected wrongly, the upper bound ub may be increased or decreased by one. When ub is increased, it is still an upper bound. When it is decreased, which may happen when two collinear lines in a face are detected as noncollinear because they deviate much from collinearity, it can be smaller than the DRF. In this case, choosing $i = ub - D_{Null(\mathbf{P})} + 2$ can still provide a large space for the reconstruction. Now, we give the algorithm for 3D object reconstruction.

Algorithm 2. (3D object reconstruction from a line drawing LD)

1. Compute the projection matrix $\mathbf{P}$ and the dimension $D_{Null(\mathbf{P})}$ of $Null(\mathbf{P})$ from LD;
2. Find the smallest upper bound ub of the DRF for LD by running Algorithm 1 n times;
3. Find $\mathbf{V}$ satisfying $\mathbf{P} = \mathbf{USV}^T$ by SVD;
4. Construct a space $H_{ub-D_{Null(\mathbf{P})}+2}$ spanned by the last $ub + 2$ column vectors of $\mathbf{V}$;
5. Search for $\mathbf{f}^*$ in $H_{ub-D_{Null(\mathbf{P})}+2}$ such that
$$\mathbf{f}^* = \arg\min_{\mathbf{f}\in H_{ub-D_{Null(\mathbf{P})}+2}} \Psi(\mathbf{f}); \qquad (29)$$
6. Compute all the depths z_p by $z_p = a_ix_p + b_iy_p + c_i$ if $\mathbf{v}_p = [x_p, y_p, z_p]^T$ is on the face (a_i, b_i, c_i);
7. For any vertex, if there are two or more different depths obtained from the faces passing through it, take the average of these depths to be its final depth.

In Step 2, n is set to 10 in all the experiments. Next, we explain the implementation of Steps 4 and 5 in more detail. Since the set of the last $ub + 2$ column vectors of $\mathbf{V}$ is $\{\mathbf{u}_{3N_f-ub-1}, \mathbf{u}_{3F_f-ub}, \cdots, \mathbf{u}_{3N_f}\}$, we have

$$H_{ub-D_{Null(\mathbf{P})}+2} = \text{span}\{\mathbf{u}_{3N_f-ub-1}, \mathbf{u}_{3N_f-ub}, \cdots, \mathbf{u}_{3N_f}\} \qquad (30)$$

or

$$H_{ub-D_{Null(\mathbf{P})}+2} = \left\{\mathbf{f} = \sum_{i=3N_f-ub-1}^{3N_f} k_i\mathbf{u}_i \,\middle|\, k_i \in R,\ 3N_f - ub - 1 \le i \le 3N_f\right\}. \qquad (31)$$

Searching for $\mathbf{f}^*$ in (29) is then transformed to searching for a set $\{k^*_{3N_f-ub-1}, k^*_{3N_f-ub}, \cdots, k^*_{3N_f}\}$ such that

$$\{k^*_{3N_f-ub-1}, k^*_{3N_f-ub}, \cdots, k^*_{3N_f}\} = \arg \min_{\substack{k_i \in R \\ 3N_f-ub-1 \le i \le 3N_f}} \Psi\left(\sum_{i=3N_f-ub-1}^{3N_f} k_i \mathbf{u}_i \right). \tag{32}$$

Many optimization algorithms can be used to find $\{k^*_{3N_f-ub-1}, k^*_{3N_f-ub}, \cdots, k^*_{3N_f}\}$, such as the hill-climbing algorithm in [9], the quasi-Newton search algorithm [33], and genetic algorithms [34]. Algorithm 2 works well using any of them. It is not guaranteed that a global optimal solution can be found. In fact, it is possible that there are an infinite number of expected 3D objects in $H_{ub-D_{Null(\mathbf{P})}+2}$. If the 3D object found is the one expected, the algorithm is considered successful in dealing with the line drawing.

From this algorithm, we see that the 2D coordinates of the line drawing are not changed. As a result, the faces of the reconstructed object are not required to be strictly planar (see Step 7). Two remarkable points in our method are that 1) the dimension of $H_{ub-D_{Null(\mathbf{P})}+2}$ is in general much smaller than the number of vertices of the line drawing, which is the dimension of the search space in the previous most related methods and 2) even though the objective function $\Psi(\mathbf{f})$ uses only one constraint (MSDA), our method can perform very well when handling complex line drawings (see the next section).

9 Experimental Results

In this section, we illustrate 3D object reconstruction from a number of line drawings and compare the results using Algorithm 2, Leclerc and Fischler's algorithm [9], and Lipson and Shpitalni's algorithm [21]. The three algorithms are abbreviated to POA (plane-based optimization algorithm), LFA, and LSA, respectively. All the algorithms are implemented in Visual C++, running on a 2.5 GHz Pentium IV PC. The line drawings are inputted using either a mouse or the pen of a tablet PC.

LFA and LSA are most related to POA as explained in Section 2. In POA, we use only one constraint, MSDA, in the objective function $\Psi(\mathbf{f})$, but the constraint face planarity is also implied since the search space is created based on it. In LFA, the two constraints, MSDA and face planarity, are used in the objective function. In LSA, more constraints, including MSDA and face planarity, are combined in the objective function. Of all the previous methods, LSA can handle the widest range of objects. To find the optimal solutions to the three objective functions, we tried using the hill-climbing optimization in [9], the quasi-Newton search algorithm [33], and genetic algorithms [34]. The quasi-Newton search algorithm appears to work best as a whole for optimization in POA, LFA, and LSA, so it is used in the experiments described here.

We tested all the line drawings given in the experimental sections in [9] and [21]. POA can successfully reconstruct the 3D objects from all these line drawings, some of which are shown in Fig. 10 (line drawings a-d), together with two new line drawings e and f. In Fig. 10, a1-f1, a2-f2, and a3-f3 are the reconstruction results obtained by LFA, LSA, and POA, respectively. The sign "$\surd$" or "$\times$" in Fig. 10 under each reconstruction result indicates whether the result is an expected one or not. LFA can produce the expected objects from only two line drawings: e and f. LSA is better than LFA and is successful in the reconstruction for line drawings a-e, but it fails when dealing with line drawing f. On the contrary, POA can handle all these line drawings.

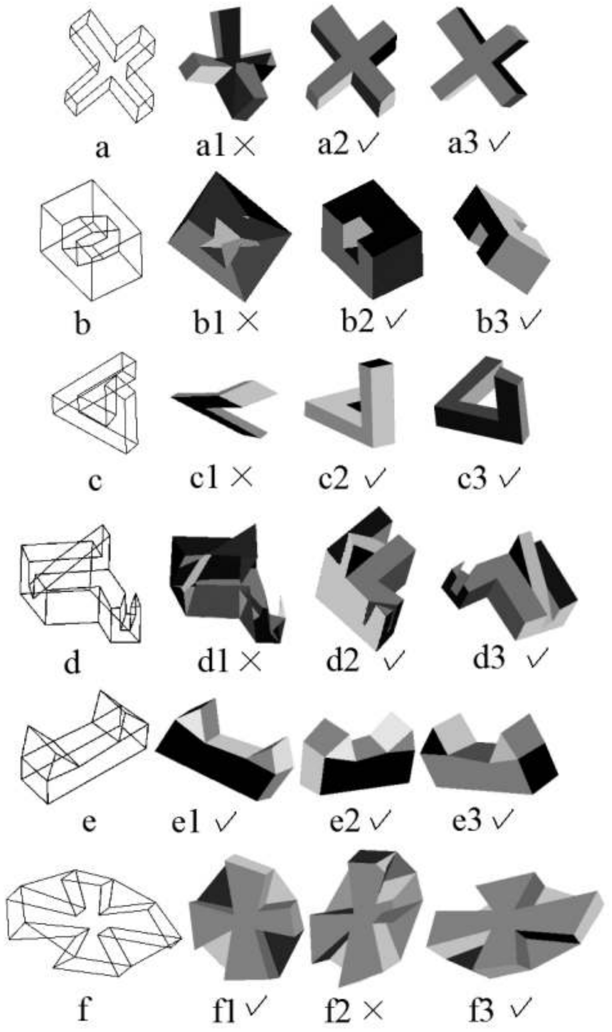

Fig. 10. Reconstruction results from the line drawings a-f: a1-f1 by LFA, a2-f2 by LSA, and a3-f3 by POA.

It should be mentioned that a result may be affected by the initial settings of the z-coordinates of the vertices in LFA and LSA and by the initial settings of the face parameter vector $\mathbf{f}$ in POA. When we say that an algorithm fails in the reconstruction from a line drawing, we mean that it cannot generate an expected result with random initializations in many trials (10 in this case).

Fig. 11 shows another set of more complex line drawings and the reconstruction results by the three algorithms. Both LFA and LSA fail in the reconstruction from these line drawings. We have found that with random initializations, LFA never generated one expected 3D object from this set of line drawings and neither did LSA from the line drawings j-l. However, POA is successful in reconstructing the 3D objects from all these line drawings. The examples given in Figs. 10 and 11 clearly demonstrate that POA is more powerful than LFA and LSA. In addition to these line drawings, we have

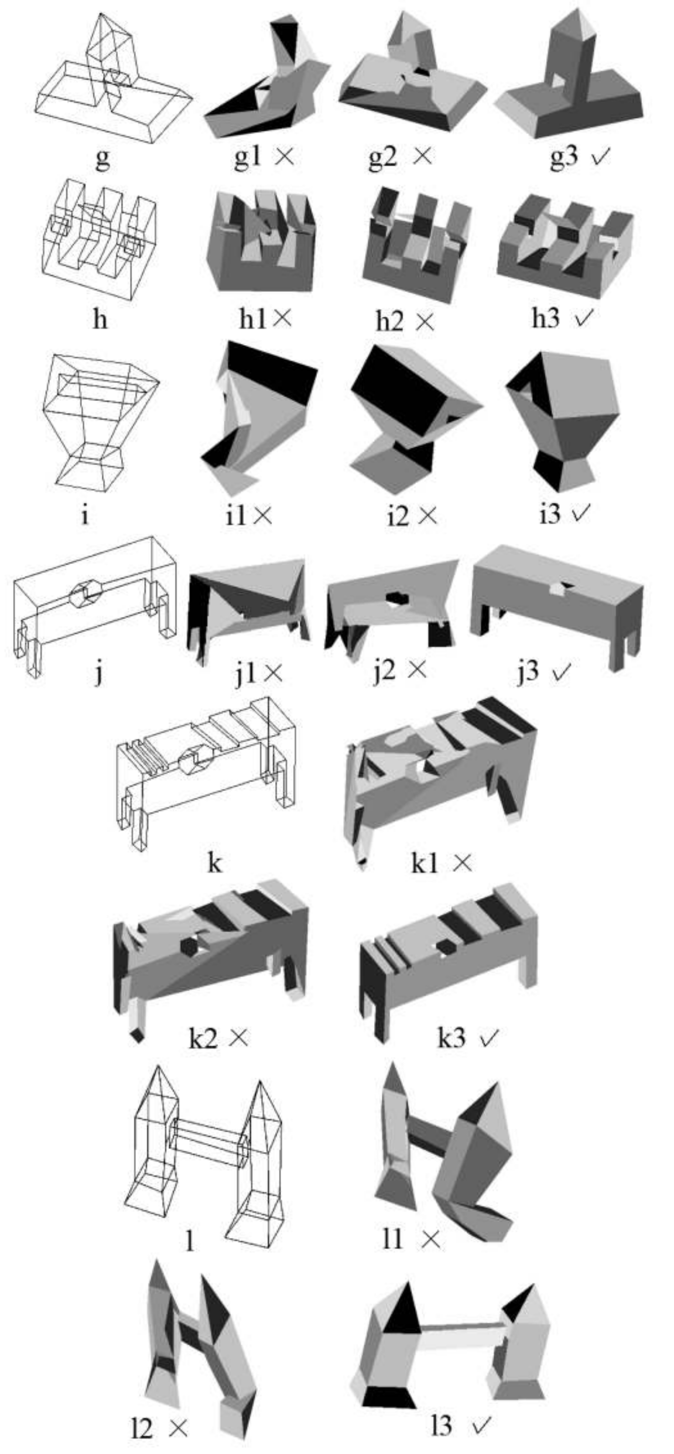

Fig. 11. Reconstruction results from the line drawings g-l: g1-l1 by LFA, g2-l2 by LSA, and g3-l3 by POA.

tried more than 50 other examples, among which neither LFA nor LSA outperforms POA in the reconstruction.

In the experiments, the same sets of face circuits are used as the input to the three algorithms. It is obvious that the success of reconstruction relies on the correctly found face circuits. In the line drawing shown in Fig. 11l, four artificial lines are added to connect the bridge with the two towers. They are necessary to provide spatial positions between the bridge and the two towers.

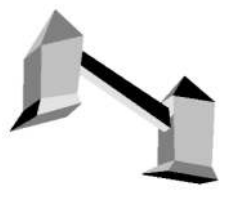

Fig. 12. A distorted reconstruction result by POA from the line drawing l shown in Fig. 11.

TABLE 1
The Upper Bounds ub and the Numbers N_v of Vertices for the Line Drawings a-l

Line drawing	a	b	c	d	e	f	g	h	i	j	k	l
N_v	20	20	16	30	16	24	25	50	20	44	76	42
ub	4	4	4	4	6	4	5	5	6	4	4	10

TABLE 2
The Average Distances of the Vertices to Their Best Fit Planes for Objects a3-l3

Object	a3	b3	c3	d3	e3	f3
Distance	1.02	1.69	0.85	1.96	2.51	0.56
Object	g3	h3	i3	j3	k3	l3
Distance	2.45	0.83	1.21	1.02	2.35	1.23

When POA does the reconstruction from line drawing l in Fig. 11, it sometimes produces distorted objects like the one shown in Fig. 12, in which the bridge is not perpendicular to the two towers. The objective function $\Psi(\mathbf{f})$ in POA includes only one constraint, MSDA. If we add the constraint line parallelism into $\Psi(\mathbf{f})$, the problem can be solved. This constraint requires that parallel lines in a line drawing should be also parallel in the reconstructed 3D object.

Table 1 shows the upper bounds ub found by Algorithm 1 from line drawings a-l and the numbers N_v of vertices of these line drawings. As seen from Algorithm 2, Algorithm 1 is run 10 times for one line drawing, and then, the smallest ub is chosen. We have examined these upper bounds and found that all of them are the DRFs for their corresponding line drawings. The dimension of the search space $H_{ub-D_{Null(\mathbf{P})}+2}$ in POA is $ub+2$, whereas the dimension of the search space R^{N_v} is N_v in LFA and LSA. It is obvious that $H_{ub-D_{Null(\mathbf{P})}+2}$ is much smaller than R^{N_v}. From $H_{ub-D_{Null(\mathbf{P})}+2}$, POA can find the expected objects effectively.

As mentioned before, the faces of a reconstructed object are not required to be strictly planar. To demonstrate the deviation of vertices from planarity, Table 2 shows the average distances of the vertices to their best fit planes for the objects a3-l3 in Figs. 10 and 11. The data are obtained by 1) normalizing a reconstructed object into a cube of size $100 \times 100 \times 100$, 2) finding the best fit planes in the sense of the least square error based on the 3D vertices, and 3) computing the average distance of the vertices to their corresponding best fit planes.

Since $H_{ub-D_{Null(\mathbf{P})}+2}$ is a much smaller space compared with R^{N_v}, it is expected that POA converges faster than LFA and LSA. Fig. 13 shows the numbers of iterations the three algorithms take before convergence in reconstruction. One iteration denotes one evaluation of the objective function.

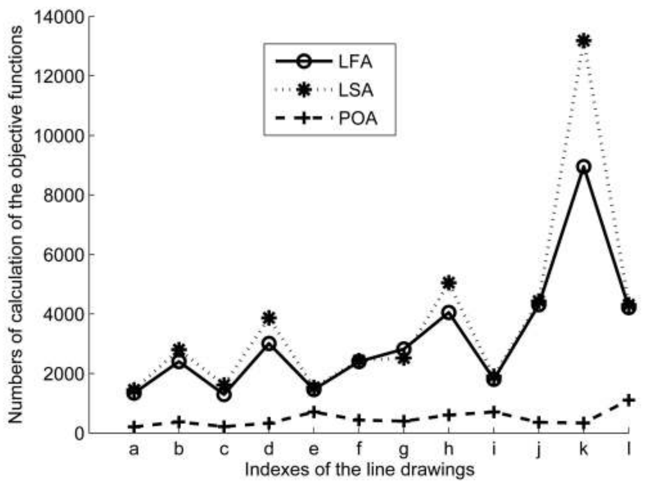

Fig. 13. Comparison of numbers of iterations in the three algorithms.

The convergence criteria in the three algorithms are the same: stopping when the difference between the values of the objective function in two consecutive iterations is smaller than 0.001. Fig. 13 clearly indicates that POA converges much faster than both LFA and LSA. Here, the convergence of an algorithm does not imply that it has generated an expected object. When dealing with line drawing h in Fig. 11, POA needs only about 1 second, whereas LFA and LSA take 9 and 51 seconds, respectively.

We can apply POA to the 3D reconstruction of an object in an image by sketching a line drawing along the edges of the faces of the object. Fig. 14 shows such an example. In Fig. 14a, some hidden lines are also drawn. When the hidden lines are removed, we obtain a 3D object with the visible faces only (Fig. 14d).

10 Conclusions

We have proposed a novel method for 3D object reconstruction from single 2D line drawings in this paper. As opposed to the depth-based formulation of the problem in previous methods, our formulation is plane based, from which a set of linear constraints can be obtained in the form of $\mathbf{Pf} = \mathbf{0}$, with a solution in the null space $Null(\mathbf{P})$ for an ideal line drawing. The dimension of $Null(\mathbf{P})$ is much smaller than the dimension of the space R^{N_v} where the optimization is carried out in most of the previous methods. Since a practical line drawing often causes the problem that $Null(\mathbf{P})$ may not contain the expected solutions, we expand $Null(\mathbf{P})$ to a larger space $H_{ub-Null(\mathbf{P})+2}$ by SVD. The dimension of $H_{ub-Null(\mathbf{P})+2}$ is still much smaller than that of R^{N_v} for general line drawings, and the expected 3D objects are more easily obtained from $H_{ub-Null(\mathbf{P})+2}$. Although only one criterion, MSDA, is used in our objective function (besides the planarity that is implicit in the search space), a number of experimental results show that our method outperforms the previous ones both in 3D reconstruction and in computational efficiency.

Future work includes using POA as the technique for the query input interface of 3D object retrieval and 3D reconstruction from line drawings representing objects with curved faces.

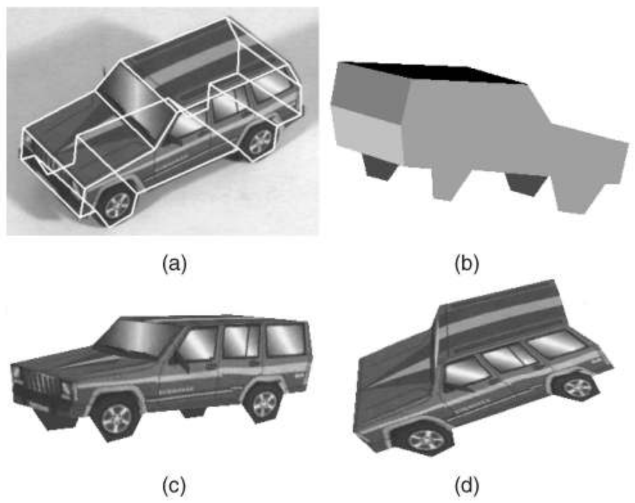

Fig. 14. (a) An image of a jeep on which a line drawing with hidden lines is superimposed. (b) The 3D jeep reconstructed by POA. (c) The 3D jeep with the texture mapped, where the two invisible wheels in the original image are filled with some gray level. (d) The 3D jeep reconstructed from the line drawing without the hidden lines.

Acknowledgments

This work was supported by the Research Grants Council of the Hong Kong SAR (Project No. CUHK 414306).

References

[1] K. Sugihara, *Machine Interpretation of Line Drawings.* MIT Press, 1986.
[2] P. Company, A. Piquer, M. Contero, and F. Naya, "A Survey on Geometrical Reconstruction as a Core Technology to Sketch-Based Modeling," *Computers & Graphics,* vol. 29, pp. 892-904, 2005.
[3] W. Whiteley, "A Matroid on Hypergraphs with Applications in Scene Analysis and Geometry," *Discrete & Computational Geometry,* vol. 4, pp. 75-95, 1989.
[4] M.C. Cooper, "Wireframe Projections: Physical Realisability of Curved Objects and Unambiguous Reconstruction of Simple Polyhedra," *Int'l J. Computer Vision,* vol. 64, no. 1, pp. 69-88, 2005.
[5] M.C. Cooper, "The Interpretations of Line Drawings with Contrast Failure and Shadows," *Int'l J. Computer Vision,* vol. 43, no. 2, pp. 75-97, 2001.
[6] A. Heyden, "On the Consistency of Line-Drawings, Obtained by Projections of Piecewise Planar Objects," *J. Math. Imaging and Vision,* vol. 6, pp. 393-412, 1996.
[7] L. Ros and F. Thomas, "Overcoming Superstrictness in Line Drawing Interpretation," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 24, no. 4, pp. 456-466, Apr. 2002.
[8] H. Li, L. Zhao, and Y. Chen, "Polyhedral Scene Analysis Combining Parametric Propagation with Calotte Analysis," *Lecture Notes in Computer Science,* vol. 3519, pp. 383-402, 2005.
[9] Y. Leclerc and M. Fischler, "An Optimization-Based Approach to the Interpretation of Single Line Drawings as 3D Wire Frames," *Int'l J. Computer Vision,* vol. 9, no. 2, pp. 113-136, 1992.
[10] G. Markowsky and M. Wesley, "Fleshing Out Wire-Frames," *IBM J. Research and Development,* vol. 24, no. 5, pp. 582-597, 1980.
[11] S. Courter and J. Brewer, "Automated Conversation of Curvilinear Wire-Frame Models to Surface Boundary Models: A Topological Approach," *Computer Graphics,* vol. 20, no. 4, pp. 171-178, 1986.
[12] S. Agarwal and J. Waggenspack, "Decomposition Method for Extracting Face Topologies from Wireframe Models," *Computer-Aided Design,* vol. 24, no. 3, pp. 123-140, 1992.
[13] M. Shpitalni and H. Lipson, "Identification of Faces in a 2D Line Drawing Projection of a Wireframe Object," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 18, no. 10, pp. 1000-1012, Oct. 1996.
[14] J. Liu and Y. Lee, "A Graph-Based Method for Face Identification from a Single 2D Line Drawing," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 23, no. 10, pp. 1106-1119, Oct. 2001.

[15] J. Liu, Y. Lee, and W. Cham, "Identifying Faces in a 2D Line Drawing Representing a Manifold Object," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 24, no. 12, pp. 1579-1593, Dec. 2002.
[16] J. Liu and X. Tang, "Evolutionary Search for Faces from Line Drawings," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 27, no. 6, pp. 861-872, June 2005.
[17] H. Li, "nD Polyhedral Scene Reconstruction from Single 2D Line Drawing by Local Propagation," *Lecture Notes in Artifical Intelligence,* vol. 3763, pp. 169-197, 2006.
[18] T. Marill, "Emulating the Human Interpretation of Line-Drawings as Three-Dimensional Objects," *Int'l J. Computer Vision,* vol. 6, no. 2, pp. 147-161, 1991.
[19] E. Brown and P. Wang, "3D Object Recovery from 2D Images: A New Approach," *SPIE Proc. Robotics and Computer Vision,* vol. 2904, pp. 138-145, 1996.
[20] K. Shoji, K. Kato, and F. Toyama, "3-D Interpretation of Single Line Drawings Based on Entropy Minimization Principle," *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* vol. 2, pp. 90-95, 2001.
[21] H. Lipson and M. Shpitalni, "Optimization-Based Reconstruction of a 3D Object from a Single Freehand Line Drawing," *Computer-Aided Design,* vol. 28, no. 8, pp. 651-663, 1996.
[22] A. Piquer, R. Martin, and P. Company, "Using Skewed Mirror Symmetry for Optimisation-Based 3D Line-Drawing Recognition," *Proc. Fifth IAPR Int'l Workshop Graphics Recognition,* pp. 182-193, 2003.
[23] A. Shesh and B. Chen, "Smartpaper: An Interactive and User Friendly Sketching System," *Proc. Ann. Conf. European Assoc. Computer Graphics,* 2004.
[24] L. Cao, J. Liu, and X. Tang, "3D Object Reconstruction from a Single 2D Line Drawing without Hidden Lines," *Proc. 10th IEEE Int'l Conf. Computer Vision,* vol. 1, pp. 272-277, 2005.
[25] K. Sugihara, "An Algebraic Approach to Shape-from-Image Problems," *Artificial Intelligence,* vol. 23, pp. 59-95, 1984.
[26] I. Shimshoni and J. Ponce, "Recovering the Shape of Polyhedra Using Line-Drawing Analysis and Complex Reflectance Models," *Computer Vision and Image Understanding,* vol. 65, no. 2, pp. 296-310, 1997.
[27] H. Shimodaira, "A Shape-from-Shading Method of Polyhedral Objects Using Prior Information," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 28, no. 4, pp. 612-624, Apr. 2006.
[28] A. Turner, D. Chapman, and A. Penn, "Sketching Space," *Computers & Graphics,* vol. 24, pp. 869-879, 2000.
[29] P. Company, M. Contero, J. Conesa, and A. Piquer, "An Optimisation-Based Reconstruction Engine for 3D Modelling by Sketching," *Computers & Graphics,* vol. 28, pp. 955-979, 2004.
[30] P.A.C. Varley and R.R. Martin, "Estimating Depth from Line Drawings," *Proc. Seventh ACM Symp. Solid Modeling and Applications,* pp. 180-191, 2002.
[31] P.A.C. Varley, R.R. Martin, and H. Suzuki, "Frontal Geometry from Sketches of Engineering Objects: Is Line Labelling Necessary?" *Computer-Aided Design,* vol. 37, pp. 1285-1307, 2005.
[32] G. Strang, *Introduction to Linear Algebra.* Wellesley-Cambridge Press, 1998.
[33] W. Press, S. Teukolsky, W. Vetterling, and B. Flannery, *Numerical Recipes in C++: The Art of Scientific Computing.* Cambridge Univ. Press, 2002.
[34] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs.* Springer, 1996.

Jianzhuang Liu received the BE degree from the Nanjing University of Posts and Telecommunications, China, in 1983, the ME degree from the Beijing University of Posts and Telecommunications, China, in 1987, and the PhD degree from the Chinese University of Hong Kong in 1997. From 1987 to 1994, he was an academic member in the Department of Electronic Engineering, Xidian University, China. From August 1998 to August 2000, he was a research fellow in the School of Mechanical and Production Engineering, Nanyang Technological University, Singapore. Then, he was a postdoctoral fellow in the Chinese University of Hong Kong for several years. He is now an assistant professor in the Department of Information Engineering, The Chinese University of Hong Kong. His research interests include image processing, computer vision, pattern recognition, and graphics. He is a senior member of the IEEE.

Liangliang Cao received the BE degree from the University of Science and Technology of China, Hefei, China, in 2003 and the MPhil degree from the Chinese University of Hong Kong, Hong Kong, in 2005. After spending one year as a research assistant in the Department of Information Engineering, The Chinese University of Hong Kong, he is now a PhD student in the Department of Electrical and Computer Engineering, University of Illinois at Urbana-Champaign. His research interests include computer vision and machine learning.

Zhenguo Li received the BS and MS degrees from the Department of Mathematics, Peking University, China, in 2002 and 2005, respectively. He is currently a PhD candidate in the Department of Information Engineering, The Chinese University of Hong Kong. His research interests include computer vision and machine learning.

Xiaoou Tang received the BS degree from the University of Science and Technology of China, Hefei, China, in 1990, the MS degree from the University of Rochester, Rochester, New York, in 1991, and the PhD degree from the Massachusetts Institute of Technology, Cambridge, in 1996. He is a professor and the director of the Multimedia Laboratory in the Department of Information Engineering, The Chinese University of Hong Kong. He is also the group manager of the Visual Computing Group at the Microsoft Research Asia. He is a local chair of the IEEE 10th International Conference on Computer Vision (ICCV '05), an area chair of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 2007, a program chair of ICCV '09, and a general chair of the ICCV International Workshop on Analysis and Modeling of Faces and Gestures 2005. He is a guest editor of the special issue on underwater image and video processing of the *IEEE Journal of Oceanic Engineering* and the special issue on image and video-based biometrics of *IEEE Transactions Circuits and Systems for Video Technology*. He is an associate editor of the *IEEE Transactions on Pattern Analysis and Machine Intelligence*. His research interests include computer vision, pattern recognition, and video processing. He is a senior member of the IEEE.