Symmetric Piecewise Planar Object Reconstruction from a Single Image

Tianfan Xue[1,2], Jianzhuang Liu[1,2], and Xiaoou Tang[1,2]
[1]Department of Information Engineering, The Chinese University of Hong Kong
[2]Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, China
{xtf009,jzliu,xtang}@ie.cuhk.edu.hk

Abstract

Recovering 3D geometry from a single view of an object is an important and challenging problem in computer vision. Previous methods mainly focus on one specific class of objects without large topological changes, such as cars, faces, or human bodies. In this paper, we propose a novel single view reconstruction algorithm for symmetric piecewise planar objects that are not restricted to some object classes. Symmetry is ubiquitous in manmade and natural objects and provides rich information for 3D reconstruction. Given a single view of a symmetric piecewise planar object, we first find out all the symmetric line pairs. The geometric properties of symmetric objects are used to narrow down the searching space. Then, based on the symmetric lines, a depth map is recovered through a Markov random field. Experimental results show that our algorithm can efficiently recover the 3D shapes of different objects with significant topological variations.

1. Introduction

3D object reconstruction from a 2D image is an important and challenging problem in computer vision. It has wide applications in virtual reality, 3D object retrieval, and object detection. Many methods have been proposed to recover 3D models from images taken from known camera viewpoints, known as multi-view stereo [9], [23], and some have achieved very good results. However, its applications are limited by the multi-view requirement. It is often difficult to have multiple images that satisfy the multi-view setting for reconstruction. Besides, the majority of images on the internet and in personal photo albums are single views. Generally, recovering 3D shape from a single image is an ill-posed problem, and additional information (or constraints) is needed to handle it. Methods in [6], [28], and [11] try to solve this problem with the help of user's input, which often requires intensive manual effort. In order to automatically recover 3D shapes from single views, some strong constraints on object shapes or pre-trained models are introduced in [1], [13], [15], [24], and [27]. These methods obtain good results on a specific object class such as faces [1], human bodies [24], and cars [15], or on some specific scenes like indoor scenes with planar walls, a ceiling, and a floor [13], and planar surfaces with repetitive patterns [27]. However, a method focusing on one class cannot be used to deal with another class. Therefore, finding good constraints for single view reconstruction is a necessary step. Ideal constraints should be as general as possible to fit for more objects and also be as strict as possible to make the problem well-posed.

Figure 1. Examples of symmetric and piecewise planar objects.

In this paper, we propose to recover 3D shapes from single views of reflectionally symmetric[1] and piecewise planar objects. Symmetry is a good constraint for single-view object reconstruction, because it ubiquitously exists in not only manmade but also natural objects. The "piecewise planar" constraint also exists in many manmade objects such as those in Figure 1. These two constraints introduce useful information to handle the reconstruction problem.

It should be mentioned that there is much work on symmetry detection from 2D images such as [4], [14], and [18]. These algorithms focus on objects lying on a single plane and do not take the 3D geometry of the objects into consideration. They cannot be applied to 3D object reconstruction.

The rest of this paper is organized as follows. Section 2 gives the assumptions and the camera model used in this work. Section 3 describes useful geometric properties of symmetric objects. Based on these properties, an algorithm for plane and symmetric lines detection is presented in Section 4. Depth map estimation with a Markov random field (MRF) is proposed in Section 5. Section 6 shows our experimental results, and Section 7 concludes this paper. Figure 2 illustrates the main steps of our method.

[1]In the rest of this paper, we simply use symmetry to denote reflectional symmetry for conciseness.

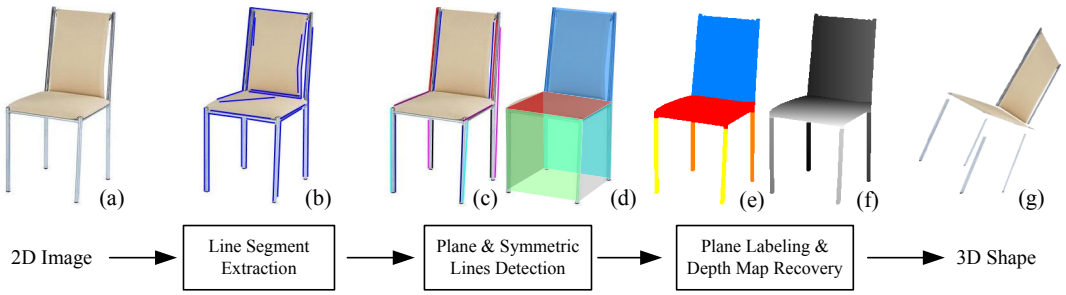

Figure 2. Reconstruction flowchart. Given a 2D image (a), line segments (see (b)) are first extracted from the image. Then, a set of symmetric line pairs (marked with the same colors in (c)) and planes (see (d)) are detected. Next, each pixel is assigned to one plane through a MRF (plane labeling (e)), and a depth map (f) is recovered based on the labeling result. The last image (g) is a rendering result of the 3D object in a novel view.

2. Assumptions and Camera Model

2.1. Assumptions

In addition to the assumption that an object is symmetric and piecewise planar, we assume that the object is photoed from a generic view, which means that no plane of the object is projected to a line in the image and the image plane is not perpendicular to any dominant axis of the object. We assume that the object structure satisfies the Manhattan world assumption, which has been used in many recent works related to 3D reconstruction such as [8] and [10]. In the original Manhattan world assumption, objects should contain three dominant and mutually orthogonal directions and most of the object planes are perpendicular to one of these directions. In this work, we use a weaker assumption that allows the planes to deviate from the main directions but less than 25°.

Same as most prior works, in this paper, we do not address the issue of object segmentation. For experiments, we use images of objects, without the background, which are abundant on the internet, such as those in Flickr and Google image. For objects with background, some interactive segmentation algorithms [20] can be used to obtain the clear objects.

2.2. Camera Model

We use a simplified camera model in this work. The camera has zero skew and does not have radial distortion, and the aspect ratio of the pixel equals 1. In this model, the projection matrix is:

$$M = [K|\mathbf{0}], \quad K = \begin{pmatrix} -f & 0 & u_0 \\ 0 & -f & v_0 \\ 0 & 0 & 1 \end{pmatrix}, \qquad (1)$$

where f is the focal length of the camera and (u_0, v_0) is the position of the principle point in the camera coordinate system.

In the rest of the paper, a bold upper-case letter (say, $\mathbf{X}$) denotes the homogeneous coordinate of a 3D point, and its 2D projection on the image plane is denoted by the corresponding bold lower-case letter $\mathbf{x}$. A plane $n_x x + n_y y + n_z z + d = 0$ is represented by $\boldsymbol{\pi} = (n_x, n_y, n_z, d)^\top = (\mathbf{n}^\top, d)^\top$, where $\mathbf{n} = (n_x, n_y, n_z)^\top$ is the normal of the plane. If not specified, homogeneous coordinates are used, and variables in Euclidean coordinates are represented by letters with a tilde above them, such as $\widetilde{\mathbf{X}}$.

3. Single View Geometry of Symmetric Objects

In this section, we show properties used to reduce the searching space in symmetry detection. One is that the symmetric point of a given point must lie on a special line called epipolar line. The other is that the position of the symmetry plane of the object does not affect the reconstruction result as long as the normal of the plane is not changed. We first define the epipole and epipolar lines in Definition 1. Note that these two terms have often appeared in 3D geometry related literature in computer vision [9], [21]. For completeness and easy understanding of our work, we show their definitions again here.

Definition 1. *Let $\boldsymbol{\pi} = (\mathbf{n}^\top, d)^\top$ be the symmetry plane of a symmetric object. The epipole is the vanishing point of the lines parallel to $\mathbf{n}$. The epipolar lines are the lines passing through the epipole.*

Figure 3 gives an example of the epipole and epipolar lines. The following Property 1 is well known [21] and used in our work to find pairs of symmetric points.

Property 1. *Let $\mathbf{x}'$ be the symmetric point of a point $\mathbf{x}$ in a 2D image. Then $\mathbf{x}'$ must lie on the epipolar line passing through $\mathbf{x}$.*

For the detection of pairs of symmetric lines, we use the *angle span* defined as follows:

Definition 2. *Let O be the epipole, A and B be the endpoints of a line l, and $\tan\theta_1$ and $\tan\theta_2$ be the slopes of lines OA and OB. The angle span $span(l)$ of l is defined as the interval $[\theta_1, \theta_2]$ if $\theta_1 > \theta_2$ or $[\theta_2, \theta_1]$ if $\theta_2 \geq \theta_1$.*

Property 2. *Two symmetric lines have the same angle span.*

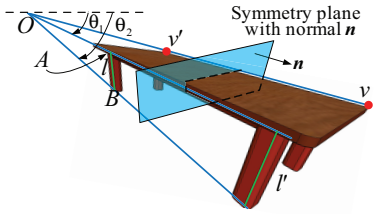

Figure 3. Illustration of an epipole, epipolar lines, and angle spans. O is the epipole and the three blue lines passing through O are epipolar lines. The symmetric point v' of v lies on the epipolar line passing through v. Symmetric lines l and l' have the same angle span.

Figure 3 illustrates the angle span of l with endpoints A and B. Property 2 is a direct corollary of Property 1. In Section 4.5, we will discuss how to exploit this property to find a set of possible pairs of symmetric lines.

From [9], the vanishing point of a line l is at $K\mathbf{n}_l$ in the image plane, where $\mathbf{n}_l$ is the direction of l. In our case, this vanishing point is the epipole and $\mathbf{n}_l = \mathbf{n}$. Thus the epipole is at $K\mathbf{n}$. For a pair of symmetric points $\mathbf{x}$ and $\mathbf{x}'$, they are collinear with the epipole, and if $\mathbf{x} \neq \mathbf{x}'$, it is easy to prove that there exist unique α and β such that $K\mathbf{n} = \alpha\mathbf{x} + \beta\mathbf{x}'$.

Before showing that the position of the symmetry plane does not affect the reconstruction result if its normal $\mathbf{n}$ is fixed, we prove the following property.

Property 3. *Let $M = [K|\mathbf{0}]$ be the projection matrix, $\pi = (\mathbf{n}^\top, d)^\top$ be the symmetry plane, and $\mathbf{x}$ and $\mathbf{x}'$ ($\mathbf{x} \neq \mathbf{x}'$) be a pair of symmetric points in the 2D image plane. If $K\mathbf{n} = \alpha\mathbf{x} + \beta\mathbf{x}'$, then the corresponding 3D points of $\mathbf{x}$ and $\mathbf{x}'$ in Euclidean coordinates are:*

$$\widetilde{\mathbf{X}} = \frac{\alpha d K^{-1}\mathbf{x}}{1/2 - \beta\mathbf{n}^\top K^{-1}\mathbf{x}'}, \tag{2}$$

$$\widetilde{\mathbf{X}}' = -\frac{\beta d K^{-1}\mathbf{x}'}{1/2 - \beta\mathbf{n}^\top K^{-1}\mathbf{x}'}. \tag{3}$$

Proof. Let $\mathbf{X}^\top = (\mathbf{X}_{xyz}^\top, X_r)$ and $\mathbf{X}'^\top = (\mathbf{X}_{xyz}'^\top, X_r')$. Since $M\mathbf{X} = \mathbf{x}$, $M\mathbf{X}' = \mathbf{x}'$, and $M = [K|\mathbf{0}]$, we have

$$\mathbf{X}_{xyz} = K^{-1}\mathbf{x}, \;\; \mathbf{X}'_{xyz} = K^{-1}\mathbf{x}'. \tag{4}$$

As $\mathbf{X}$ and $\mathbf{X}'$ are a pair of symmetric points with respect to the plane $\pi = (\mathbf{n}^\top, d)^\top$, according to [22], there is a scalar h such that

$$\begin{cases} h\mathbf{X}_{xyz} = \mathbf{X}'_{xyz} - 2\mathbf{n}\mathbf{n}^\top\mathbf{X}'_{xyz} + 2dX'_r\mathbf{n}, \\ hX_r = X'_r. \end{cases} \tag{5}$$

Suppose that $||\mathbf{n}|| = 1$. Then, (4) and (5) lead to

$$K\mathbf{n} = \frac{h\mathbf{x}}{2dX'_r - 2\mathbf{n}^\top K^{-1}\mathbf{x}'} - \frac{\mathbf{x}'}{2dX'_r - 2\mathbf{n}^\top K^{-1}\mathbf{x}'}. \tag{6}$$

From (5), (6) and $K\mathbf{n} = \alpha\mathbf{x} + \beta\mathbf{x}'$, we obtain:

$$\widetilde{\mathbf{X}} = \frac{1}{X_r}\mathbf{X}_{xyz} = \frac{\alpha d K^{-1}\mathbf{x}}{1/2 - \beta\mathbf{n}^\top K^{-1}\mathbf{x}'}, \tag{7}$$

$$\widetilde{\mathbf{X}}' = \frac{1}{X'_r}\mathbf{X}'_{xyz} = -\frac{\beta d K^{-1}\mathbf{x}'}{1/2 - \beta\mathbf{n}^\top K^{-1}\mathbf{x}'}. \tag{8}$$

□

Suppose that the projection matrix $M = [K|\mathbf{0}]$ is given and the normal $\mathbf{n}$ of symmetry plane is fixed. Given a set of pairs of symmetric points, the 3D positions of these points can be calculated with (2) and (3). Furthermore, from (2) and (3) we can see that, if the symmetry plane moves along its normal direction $\mathbf{n}$ (only d changes in this case), the 3D coordinates of these points will only proportionally increase/decrease (i.e., the shape of the object changes only up to a scale). Therefore, we can arbitrarily choose d.

4. Plane Detection

In our method, detecting a set of planes on the object consists of four steps: 1) line segment extraction, 2) vanishing point detection and camera calibration, 3) symmetric line pair detection and hypothesis plane generation, and 4) real plane selection from the set of hypothesis planes. The details of the steps are described in the following.

4.1. Line Segment Extraction

We first use the bilateral filter [26] to reduce the noise in the input image. Then we generate the edge map of the filtered image with Canny edge detector [19]. Line segments are extracted from the edge map through the Matlab toolbox by Kovesi [12]. Short edges and duplicated edges (edges that are very close to each other) are eliminated. Line segments on the boundary of the object are also extracted using the algorithm in [12].

4.2. Vanishing Point Detection and Camera Calibration

The camera calibration matrix K (see Section 2.2) is estimated through vanishing points. Under the Manhanttan world assumption, there are three dominant and mutually perpendicular directions in the object. Each dominant direction correspond to a vanishing point. We use the method in [25] to detect the three vanishing points corresponding to the three dominant directions. Based on the vanishing points, K is calculated using the method in [9].

4.3. Position of the Symmetry Plane

To find the position of the symmetry plane $\boldsymbol{\pi} = (\mathbf{n}, d)$, we only need to derive its normal $\mathbf{n}$, as d does not affect the reconstruction result (Property 3). Based on the Manhattan world assumption, $\mathbf{n}$ is parallel to one of the dominant directions. For each of the three dominant directions, we apply our reconstruction algorithm under the assumption

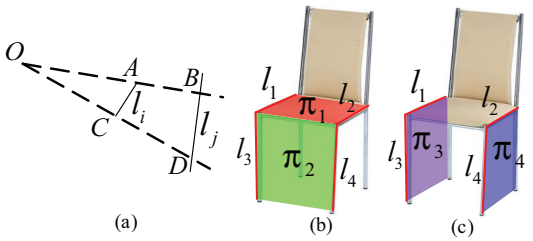

Figure 4. (a) A potential pair of symmetric lines. (b) Illustration of the detection of hypothesis planes perpendicular to the symmetry plane. (c) Illustration of the detection of planes not perpendicular to the hypothesis planes.

that $\mathbf{n}$ is parallel to this direction, resulting in three reconstructed 3D objects. Then the best object is selected. This selection process will be explained at the end of Section 5. Without loss of generality, in Sections 4.4, 4.5, and 5, we present our reconstruction algorithm based on the assumption that $\mathbf{n}$ is parallel to the z axis.

4.4. Hypothesis Plane Detection

We first find out all the line pairs which have the potential to be pairs of symmetric lines. According to Property 2, if two lines are symmetric, their angle spans are the same in the ideal case. Considering the inaccuracy of line extraction and occlusion, all the line pairs with large overlapping angle spans are considered to be potential pairs of symmetric lines. The overlapping ratio of the angle spans of two lines l_i and l_j is computed by:

$$overratio(l_i, l_j) = 2\frac{|span(l_i) \cap span(l_j)|}{|span(l_i)| + |span(l_j)|}, \quad (9)$$

where $|span(l)|$ is the length of the interval $span(l)$.

For each potential pair (l_i, l_j), their 3D coordinates are deduced as follows: First, two rays from the epipole which pass through the endpoints A and C of l_i are added, and these rays intersect l_j or its extension at B and D in the 2D image plane, as shown in Figure 4(a). According to Property 1, the symmetric points of A and C are B and D, respectively. Then, we calculate the 3D coordinates of A, B, C, and D using Property 3, which are coplanar in the 3D space. Notice that when l_i or l_j passes through the epipole (which means that the length of the angle span of l_i or l_j is zero), we cannot determine its 3D position. Therefore, those lines with small angle spans are not considered in the detection of symmetric line pairs.

Each pair found by the above method determines a hypothesis plane perpendicular to the symmetry plane. For example, in Figure 4(b), there are two pairs of symmetric lines: (l_1, l_2) and (l_3, l_4), which determine planes $\boldsymbol{\pi}_1$ and $\boldsymbol{\pi}_2$, respectively, and $\boldsymbol{\pi}_1$ and $\boldsymbol{\pi}_2$ are perpendicular to the symmetry plane. How to find planes not perpendicular to the symmetry plane will be discussed at the end of Section 4.5.

Algorithm 1 Detection of potential symmetric line pairs and hypothesis planes perpendicular to the symmetry plane

Input: A set of line segments L

Initialization: The set of planes $\Pi \leftarrow \phi$, the set of symmetric line pairs $P \leftarrow \phi$

1. **for** $l \in L$
2. Remove l from L if $|span(l)| < T_{span}$
3. **for** each line pair (l_i, l_j) in L
4. **if** $D(l_i, l_j) > T_{dis}$ **and** $overratio(l_i, l_j) > T_{over}$
5. Calculate the 3D positions of l_i and l_j (with Property 3) and the plane $\boldsymbol{\pi} = (\mathbf{n}, d)$ passing through l_i and l_j. Add (l_i, l_j) to P
6. **if** $\langle \mathbf{n}, \mathbf{n}_x \rangle < 25^\circ$ or $\langle \mathbf{n}, \mathbf{n}_y \rangle < 25^\circ$, **then** add $\boldsymbol{\pi}$ to Π
7. Cluster the hypothesis planes in Π using mean-shift to obtain the groups of the planes

Return P and the set of the representative hypothesis planes (centers of the groups)

The procedure of the detection of potential symmetric line pairs and hypothesis planes perpendicular to the symmetry plane is listed in Algorithm 1. In step 2, lines with small angle spans are removed. In step 4, two lines which have a large overlap in angle spans and are not close to each other are considered to be potential symmetric lines. Here, we require two lines not to be close, otherwise an inaccurate hypothesis plane may be generated from them. $D(l_i, l_j)$ is the average distance between lines l_i and l_j. In step 6, $\mathbf{n}_x$ and $\mathbf{n}_y$ are the directions of the x and y axes, and $\langle \cdot, \cdot \rangle$ denotes the angle between two vectors. Based on the weak Manhattan world assumption, the planes perpendicular to the symmetry plane do not deviate too much from the x or y axis (recall that the symmetry plane is assumed to be perpendicular to the z axis). Those planes that do not satisfy this constraint are removed in step 6. In step 7, we cluster the hypothesis planes through mean-shift [7] to remove duplicated planes.

4.5. Hypothesis Plane Selection

To select real planes from the set of hypothesis planes found by Algorithm 1, we first assign each potential pair of symmetric lines to one hypothesis plane, and select the real pairs of symmetric lines based on this assignment. Then those hypothesis planes containing at least one such pair are considered as real planes. The detail is described as follows.

In the symmetric line pair assignment, to decide whether a pair should be assigned to a hypothesis plane, we first project the two lines of the pair from the 2D image plane back to the 3D hypothesis plane. If the two projected lines are symmetric with respect to the symmetry plane $\boldsymbol{\pi}$, we assign this pair to this plane. For example, suppose there are

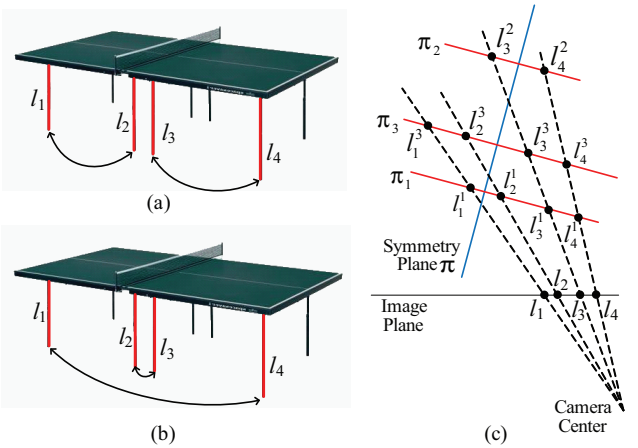

Figure 5. An example of hypothesis plane selection. (a) Input image with four line segments l_{1-4} shown in red (other line segments are not shown for clarity). Two lines pointed by a two-way arrow is a pair of potential symmetric lines. (b) Another two potential pairs of symmetric lines. (c) Geometry of the imaging and line pair assignment, where a point denotes a line in (a) or (b). The assignment result is $\{(l_1, l_2, \boldsymbol{\pi}_1), (l_3, l_4, \boldsymbol{\pi}_2), (l_1, l_4, \boldsymbol{\pi}_3), (l_2, l_3, \boldsymbol{\pi}_3)\}$.

three hypothesis planes $\boldsymbol{\pi}_{1-3}$ from the image in Figure 5(a) or (b). The geometry of the imaging is shown in Figure 5(c). If l_1 and l_2 are projected from the 2D image plane back to $\boldsymbol{\pi}_1$, they are assigned to $\boldsymbol{\pi}_1$ because they are symmetric with respect to the symmetry plane $\boldsymbol{\pi}$. However, they are not assigned to $\boldsymbol{\pi}_2$ or $\boldsymbol{\pi}_3$, since they are not symmetric on $\boldsymbol{\pi}_2$ or $\boldsymbol{\pi}_3$ with respect to $\boldsymbol{\pi}$. Similarly, l_3 and l_4 are assigned to $\boldsymbol{\pi}_2$, l_2 and l_3 to $\boldsymbol{\pi}_3$, and l_1 and l_4 to $\boldsymbol{\pi}_3$.

The following function is used to judge whether l_i and l_j are symmetric when they are projected on a hypothesis plane $\boldsymbol{\pi}_k$:

$$w_{\boldsymbol{\pi}_k}(l_i, l_j) = \exp(-D(sym(l_i^k), l_j^k)^2), \qquad (10)$$

where l_i^k is the projected line of l_i on $\boldsymbol{\pi}_k$, $sym(l_i^k)$ is the computed symmetric line of l_i^k with respect to the symmetry plane, and $D(\cdot,\cdot)$ is the average distance between two 3D lines. If $w_{\boldsymbol{\pi}_k}(l_i, l_j)$ is larger than a threshold T, we assign (l_i, l_j) to $\boldsymbol{\pi}_k$, and if there are more than one hypothesis plane with $w_{\boldsymbol{\pi}_k}(l_i, l_j) > T$, we assign (l_i, l_j) to the plane with the maximum value.

In what follows, each potential line pair together with their assignment is denoted by $(l_i, l_j, \boldsymbol{\pi}_k)$, and the set of all these line pairs with their assignments are denoted by P. Besides, $P^r \subset P$ is used to denote a subset that contains real symmetric line pairs. How to find P^r is described in the following.

The real symmetric line pairs in P^r should not contain two pairs that have a common line, as one line can only have one symmetric line. For example, $(l_1, l_4, \boldsymbol{\pi}_3)$ and $(l_1, l_2, \boldsymbol{\pi}_1)$ should not coexist in P^r. Based on this constraint, the selection of real symmetry line pairs is formulated as the following optimization problem:

$$\max_{P' \subset P} \left(\left(\sum_{(l_i, l_j, \boldsymbol{\pi}_k) \in P'} w_{\boldsymbol{\pi}_k}(l_i, l_j) \right) - \alpha |plane(P')| \right), \qquad (11)$$

$$\text{subject to:} \quad \text{no common lines in any two pairs} \in P',$$

where $plane(P') = \bigcup_{(l_i, l_j, \boldsymbol{\pi}_k) \in P'} \{\boldsymbol{\pi}_k\}$ contains all the hypothesis planes that line pairs in P' are assigned to, and α is a weighting factor. The first term in (11) requires the lines in each pair are as symmetric as possible. The second term is to minimize the number of planes in the object. This is consistent with the human visual system, as human beings usually try to interpret an object as simple as possible. For example, in Figure 5, $P_1' = \{(l_1, l_4, \boldsymbol{\pi}_3), (l_2, l_3, \boldsymbol{\pi}_3)\}$ is better than $P_2' = \{(l_1, l_2, \boldsymbol{\pi}_1), (l_3, l_4, \boldsymbol{\pi}_2)\}$, because the symmetry line pairs in P_2 are assigned to two planes $\boldsymbol{\pi}_1$ and $\boldsymbol{\pi}_2$, while the symmetric line pairs in P_1 are assigned to only one plane $\boldsymbol{\pi}_3$.

Finding the optimal solution to (11) is an NP-hard problem. To find a convex relaxation of (11), we introduce two variables x_i and y_k, where $x_i = 1$ if the ith symmetric line pair is selected and $x_i = 0$ otherwise; $y_k = 1$ if plane $\boldsymbol{\pi}_k$ is selected and $y_k = 0$ otherwise. Then (11) is reformulated as:

$$\max_{\{x_i, y_k\}} \left(\left(\sum_{(l_{a_i}, l_{b_i}, \boldsymbol{\pi}_{c_i}) \in P} x_i w_{\boldsymbol{\pi}_{c_i}}(l_{a_i}, l_{b_i}) \right) - \alpha \sum_{k=1}^{K} y_k \right),$$

subject to:

$$\begin{aligned} &1.\ \forall j, \sum_{(l_{a_i}, l_{b_i}, \boldsymbol{\pi}_{c_i}) \in P,\ \text{where } l_{a_i} = l_j \text{ or } l_{b_i} = l_j} x_i \leq 1 \\ &2.\ \forall k, i, \text{ such that } \boldsymbol{\pi}_{c_i} = \boldsymbol{\pi}_k, y_k \geq x_i, \\ &3.\ x_i \in \{0, 1\}, y_k \in \{0, 1\}. \end{aligned} \qquad (12)$$

Different from previous notation, a triple $(l_{a_i}, l_{b_i}, \boldsymbol{\pi}_{c_i})$ is used in (12). Note the only non-convex constraint is the third one. So we first remove this constraint and the problem (12) becomes a linear programming, which can be efficiently solved by the simplex method [5]. Suppose the solution to this relaxed problem is $\{\widetilde{x_i}\}$ and $\{\widetilde{y_k}\}$. Then an approximate solution to the original problem (12) is obtained by setting $x_i = 1$ if $\widetilde{x_i} > 0.5$ and $x_i = 0$ otherwise, with y_k obtained the same way.

The planes not perpendicular to the symmetry plane is detected as follows. First, the 3D positions of the line pairs in P^r are calculated by projecting them from the 2D image plane to the planes they are assigned to. Then from every two pairs (l_i, l_j) and (l_m, l_n) in P^r, we have four sets $\{l_i, l_m\}$, $\{l_i, l_n\}$, $\{l_j, l_m\}$, and $\{l_j, l_n\}$. If any of these sets can approximately form a plane in the 3D space, this plane is added to the real plane set Π^r. For example, in Figure 4(c), suppose $\{l_1, l_2\}$ and $\{l_3, l_4\}$ are two line pairs in P^r. Then planes $\boldsymbol{\pi}_3$ and $\boldsymbol{\pi}_4$ are formed by the sets $\{l_1, l_3\}$ and $\{l_2, l_4\}$ and added to Π^r. Planes deviated from the z axis more than 25° are not added to Π^r due to the weak

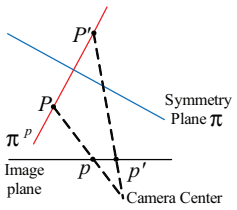

Figure 6. Geometry to define the data term.

Manhattan world assumption. Besides, duplicated planes are removed by mean-shift (see step 7 in Algorithm 1).

5. Depth Map Generation

Given a set of 3D planes Π^r found by the method in Section 4, we assign each foreground (object) pixel a label indicating which plane the pixel belongs to (background is ignored). Then the depth map is computed by projecting these pixels from the 2D image plane to the planes they are assigned to. Finding the best labels for the pixels is formulated as a minimization problem with an MRF. The energy function to be minimized is defined as:

$$E = \sum_{p} E_d(\boldsymbol{\pi}^p) + \sum_{(p,q)\in N_4} E_{ls}(\boldsymbol{\pi}^p, \boldsymbol{\pi}^q) + \sum_{(p,q)\in N_s} E_s(\boldsymbol{\pi}^p, \boldsymbol{\pi}^q), \tag{13}$$

where $\boldsymbol{\pi}^p \in \Pi^r$ is the plane pixel p is assigned to, N_4 is the 4-neighborhood system, N_s is another pixel set defined in Section 5.3, and the three terms, E_d (data term), E_{ls} (local smoothness term), and E_s (symmetry term), will be described in Sections 5.1–5.3, respectively.

5.1. Data Term

This term is used to determine which plane a pixel should be assigned to. Let p be a pixel in the image plane (see Figure 6(a)). Suppose p is assigned to plane $\boldsymbol{\pi}^p$. Then the projection P of p on $\boldsymbol{\pi}^p$ is computed. With the symmetry plane $\boldsymbol{\pi}$ and P, we can find the symmetric point P' of P in the 3D space. Then P' is projected to the image plane, resulting in p'. The similarity between p and p' reflects the possibility of p belonging to $\boldsymbol{\pi}^p$. Let

$$E_d(\boldsymbol{\pi}^p) = \alpha_{dc} E_{dc}(\boldsymbol{\pi}^p) + \alpha_{de} E_{de}(\boldsymbol{\pi}^p), \tag{14}$$

where α_{dc} and α_{de} are weighting factors, and $E_{dc}(\boldsymbol{\pi}^p)$ and $E_{de}(\boldsymbol{\pi}^p)$ are color and edge consistency terms, respectively, defined as:

$$E_{dc}(\boldsymbol{\pi}^p) = \begin{cases} R(||I_p - I_{p'}||/50), & \text{if } p' \in I_F, \\ 2, & \text{otherwise}, \end{cases} \tag{15}$$

$$E_{de}(\boldsymbol{\pi}^p) = \begin{cases} R(g(p')/15), & \text{if } p \in I_E \text{ and } p' \in I_F, \\ 2, & \text{if } p \in I_E \text{ and } p' \notin I_F, \\ 0, & \text{otherwise}, \end{cases} \tag{16}$$

where I_F is the set of all foreground pixels, I_E is the set of all edge pixels, I_p is a 3D vector representing the RGB values of p, $g(p')$ is the the distance between p' and its nearest edge pixel, and R is a kernel function defined as $R(x) = \min\{x^2, 1\}$. In these two terms, we assign an extra punishment 2 if p' is not on the object. In this paper, $\alpha_{dc} = 1$ and $\alpha_{de} = 600$, where α_{dc} is much larger due to two reasons: First, the edge consistency term is only evaluated on the edge pixels while the color consistency term is evaluated on all pixels. Second, the edge information is more reliable than the color information because the colors of two symmetric points are not always the same.

5.2. Local Smoothness Term

The local smoothness constraint enforces neighboring 2D pixels of the object are also close in the 3D space. The local smoothness term used in this paper is similar to the one in [8]. The detail is omitted here to save space.

5.3. Symmetry Term

If the symmetric pixel of a pixel p is q, then the symmetric pixel of q is also p. To enforce this constraint, the symmetry term is used to punish inconsistent pairs, defined as:

$$E_s(\boldsymbol{\pi}^p, \boldsymbol{\pi}^q) = \begin{cases} \alpha_s, & \text{if } p' = q \text{ and } q' \neq p, \\ \alpha_s, & \text{if } p' \neq q \text{ and } q' = p, \\ 0, & \text{otherwise}, \end{cases} \tag{17}$$

where $\alpha_s = 10$ is the punishment constant, p' and q' are found by the method described in Section 5.1 (also see Figure 6). N_s in (13) contains all the pixel pairs that have the potential to be symmetric, i.e., $N_s = \{(p,q) \,|\, p' = q \text{ or } q' = p \text{ for at least one plane}\}$.

The minimization of E in (13) is solved through graph cuts [2]. After labeling all the pixels of the object, the depth map is computed by projecting each pixel to its assigned 3D plane.

As discussed in Section 4.3, we obtain three 3D objects under the assumption that the symmetry plane is perpendicular to the x, y, and z axes, respectively. Finally, we choose the object with the minimum energy.

6. Experiments

In this section, a set of examples is presented to demonstrate the performance of our algorithm. We have collected a number of images of symmetric piecewise planar objects, such as chairs and cabinets, etc. Our algorithm is implemented in Matlab, and tested on a PC with 2.4GHz Intel Core 2 CPU. The average reconstruction time is 45 seconds per object (not including segmentation). The objects, their recovered depth maps, and rendering results on novel views are shown in Figure 7. We can see that the depths of almost all of the object pixels are correctly recovered.

Table 1. Plane detection and depth map recovery

Object		A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
Plane Detection	# of planes	4	7	7	6	4	9	5	4	5	8	3	6	6	7	4	4	9
	# of errors	0	1	0	0	0	1	0	0	1	1	0	0	0	2	0	0	3
Depth Recvoery	# of planes	3	4	5	4	3	6	3	3	4	5	3	4	4	4	3	2	4
	# of errors	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0

Table 1 gives the details of plane detection and depth map recovery. The first and the second rows show the numbers of all planes and incorrect planes found in the plane detection, respectively. The third and fourth rows are the numbers of all planes and incorrect planes used in the depth map recovery, respectively (here a used plane is a plane with at least one pixel assigned to it). On average, for each object, there are 5.8 planes found in plane detection with 0.53 incorrect plane. After the depth map recovery, the number of incorrect planes per object drops to 0.24.

Reconstructing 3D shapes from single view images is a challenging problem. Some of our results are not so perfect. Figure 8(b) shows the incorrect plane of object B found in plane detection which affects a small part (patch) of the object. Another problem is that when some parts of an object have multiple explanations, our algorithm may obtain an incorrect result in these parts. For example, the leg marked by a red circle in Figure 8(c) should belong to the plane for the high back of the chair, but our algorithm assigns this leg to the front vertical plane. This is because assigning this leg to the front plane does not affect the symmetry of the object (note that this leg is considered as located on the symmetry plane in this case).

7. Conclusions and Future Work

We have proposed to tackle the problem of 3D object reconstruction from single-view images using the symmetric and piecewise planar constraint. Our algorithm mainly consists of two steps: plane detection and depth map recovery. Several geometric properties are presented to help find the planes of the object and the depth map is recovered using MRFs. Experimental results show that our algorithm can deal with different 3D objects with significant topological variations, without the need of specific object shape priors.

In the future, we plan to extend this approach to handling general symmetric objects by approximating the curved surface of an object with a piecewise planar surface. Another way of dealing with more complex symmetric objects is to add some user interactions, such as combining this work with shape from line drawings [3], [16], [17].

Acknowledgement

This work was supported by grants from Natural Science Foundation of China (No. 60975029, 61070148), Shenzhen Science and Technology R&D Foundation (No. JC200903180635A, JC201005270378A), and the Research Grants Council of the Hong Kong SAR, China (Project No. CUHK 415408).

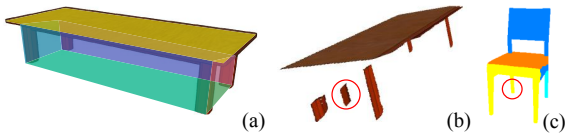

Figure 8. (a) Six correct planes of object B. (b) One incorrect plane of object B (marked by the cycle). (c) Faults in the depth map of object I (marked by the cycle).

References

[1] V. Blanz and T. Vetter. A morphable model for the synthesis of 3D faces. *SIGGRAPH*, 1999.

[2] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE T-PAMI*, 23(11):1222–1239, 2001.

[3] L. Cao, J. Liu, and X. Tang. What the back of the object looks like: 3D reconstruction from line drawings without hidden lines. *IEEE T-PAMI*, 30(3):507–517, 2008.

[4] M. Chertok and Y. Keller. Spectral symmetry analysis. *IEEE T-PAMI*, 32(7):1227–1238, 2009.

[5] G. Dantzig, A. Orden, P. WOLFE, and R. C. S. M. CA. *The generalized simplex method for minimizing a linear form under linear inequality restraints.* Defense Technical Information Center, 1954.

[6] A. R. J. François, G. Medioni, and R. Waupotitsch. Mirror symmetry $\Rightarrow$ 2-view stereo geometry. *Image and Vision Computing*, 21, 2003.

[7] K. Fukunaga and L. Hostetler. The estimation of the gradient of a density function, with applications in pattern recognition. *IEEE T-IT*, 21(1):32–40, 2002.

[8] Y. Furukawa, B. Curless, S. Seitz, and R. Szeliski. Manhattan-world stereo. *CVPR*, 2009.

[9] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge Univ. Press, 2003.

[10] D. Hoiem, A. A. Efros, and M. Hebert. Automatic photo pop-up. *ACM SIGGRAPH*, 2005.

[11] N. Jiang, P. Tan, and L. Cheong. Symmetric architecture modeling with a single image. *ACM SIGGRPHA Asia*, 28(5):1–8, 2009.

[12] P. D. Kovesi. MATLAB and Octave functions for computer vision and image processing. School of Computer Science & Software Engineering, The University of Western Australia. Available from: <http://www.csse.uwa.edu.au/∼pk/research/matlabfns/>.

[13] D. C. Lee, M. Hebert, and T. Kanade. Geometric reasoning for single image structure recovery. *ICCV*, 2009.

[14] S. Lee and Y. Liu. Curved glide-reflection symmetry detection. *CVPR*, 2009.

[15] M. J. Leotta. Predicting high resolution image edges with a generic, adaptive, 3-D vehicle model. *CVPR*, 2009.

[16] 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.

[17] 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.

[18] G. Loy and J. O. Eklundh. Detecting symmetry and symmetric constellations of features. *ECCV*, 2006.

[19] J. R. Parker. *Algorithms for Image Processing and Computer Vision*. John Wiley & Sons, 1997.

[20] C. Rother, V. Kolmogorov, and A. Blake. Grabcut: Interactive foreground extraction using iterated graph cuts. *ACM Transactions on Graphics*, 23:309–314, 2004.

[21] C. A. Rothwell, D. A. Forsyth, A. Zisserman, and J. L. Mundy. Extracting projective structure from single perspective views of 3D point sets. *ICCV*, 1993.

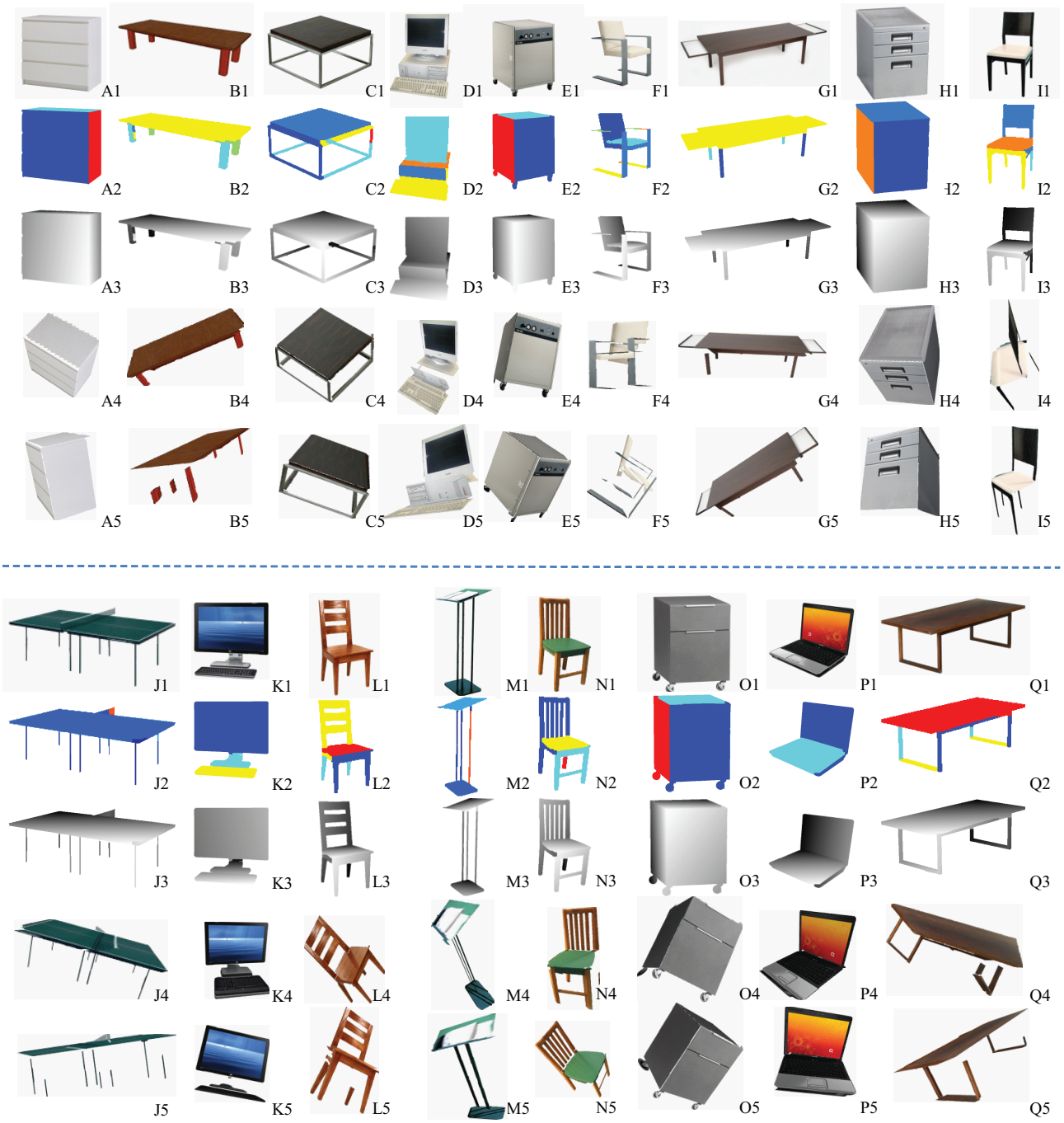

Figure 7. 3D Reconstruction results. A1–Q1: input images; A2–Q2: labeling results; A3–Q3 recovered depth maps; A4–Q4 and A5–Q5: rendering results in two novel views.

[22] P. J. Schneider and D. H. Eberly. *Geometric Tools for Computer Graphics*. Morgan Kaufmann, 2002.

[23] S. M. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. *CVPR*, 2006.

[24] L. Sigal, A. Balan, and M. J. Black. Combined discriminative and generative articulated pose and non-rigid shape estimation. *NIPS*, 2007.

[25] J. P. Tardif. Non-Iterative Approach for Fast and Accurate Vanishing Point Detection. *ICCV*, 2009.

[26] C. Tomasi and R. Manduchi. Bilateral filtering for gray and color images. *ICCV*, 1998.

[27] A. Y. Yang, K. Huang, S. Rao, W. Hong, and Y. Ma. Symmetry-based 3-D reconstruction from perspective images. *Computer Vision and Image Understanding*, 99(2):210–240, 2005.

[28] L. Zhang, G. Dugas-Phocion, J. S. Samson, and S. M. Seitz. Single-view modelling of free-form scenes. *CVPR*, 2001.