Pattern Recognition 60 (2016) 543–553

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/pr

An example-based approach to 3D man-made object reconstruction from line drawings

Changqing Zou[a,b], Tianfan Xue[c], Xiaojiang Peng[a], Honghua Li[d], Baochang Zhang[e,*], Ping Tan[b], Jianzhuang Liu[f]

[a] Hengyang Normal University, China
[b] Simon Fraser University, Canada
[c] Massachusetts Institute of Technology, United States
[d] Shandong University, China
[e] School of Automation Science and Electrical Engineering, Beihang University, Beijing, China
[f] Huawei Technology Co. Ltd., China

ARTICLE INFO

Article history:
Received 30 July 2015
Received in revised form
27 May 2016
Accepted 28 May 2016
Available online 11 June 2016

Keywords:
3D reconstruction
Example-based
Line drawing
Probabilistic graphical model

ABSTRACT

3D reconstruction from a single 2D line drawing is an important but challenging problem in computer vision. Existed methods usually fail when line drawings contain large degree of noise named sketch errors. In this paper, we present an example-based approach to reconstructing 3D object, either planar or curved, from a single-view line drawing with sketch errors. Our method is to first decompose the input line drawing into primitive components and cluster them into local groups, then turn each group into 3D shapes via a novel example-based algorithm, and lastly integrate those recovered 3D shapes from all groups to build a final complete 3D model. Comprehensive experiments on a wide range of line drawings depicting man-made objects show that the proposed approach outperform previous work, especially for line drawings containing large degree of sketch errors.

1. Introduction

Line drawing is one of the simplest and most direct means to illustrate 3D objects. The human vision system is able to interpret 2D line drawings as 3D objects without endeavor. The emulation of this ability is an important and long-standing research topic in computer vision and graphics. Three dimensional reconstruction from 2D line drawings benefits various applications such as reconstruction from conceptual sketches [1–4], interactive 3D modeling from single images [5–7], and sketch-based 3D shape retrieval [8–10].

The problem of 3D reconstruction from single 2D line drawing is challenging since it is intrinsically ill-posed due to the missing of the depth information. In order to resolve the ambiguity, some researchers use additional information such as local regularity cues, reference models, and gestures [11–16]. Differently, the rule-based methods [1,17–23] usually reconstruct 3D objects by optimizing an objective function built from a set of heuristic image-based rules that summarize human visual perceptions. Generally, rule-based methods can success for a large range of line drawings, but might fail to obtain “good” results, e.g., vertices on a planar face might not exactly lie on the same plane in the result model. This is mainly due to the fact that heuristic rules cannot cover all cases [24,25]. Imperfect line drawings and sketch errors also make those rules to be less useful. Moreover, there is no principled way to tune the parameters that balance each type of heuristic rules.

In this paper, we propose an example-based 3D reconstruction approach, which is the comprehensive version of our preliminary work published in [24]. We extend the previous algorithm to handle wider range of 3D man-made objects, which are composed of both planar and/or curved surfaces. Comprehensive experiments are performed to evaluate the algorithm's performance on line drawings consisting of both straight and curved segments. The key insight of the proposed approach is that a complex 3D object, especially a man-made object, can usually be decomposed into a set of simpler primitive 3D shapes. For instance, the mechanical object in Fig. 1(a) is a composition of 3D shapes shown in Fig. 1(c), which can be generated from the six 3D parametric templates shown in Fig. 1(d). This insight motivates us to adopt an example-driven approach to reconstruct 3D objects from line drawings.

Given a line drawing with both straight and curved lines, our method reconstructs the 3D object in four steps: (1) decompose the input line drawing into simple components, (2) group these components into part groups based on a set of structure

* Corresponding author.
E-mail address: bczhang@buaa.edu.cn (B. Zhang).

http://dx.doi.org/10.1016/j.patcog.2016.05.031
0031-3203/© 2016 Elsevier Ltd. All rights reserved.

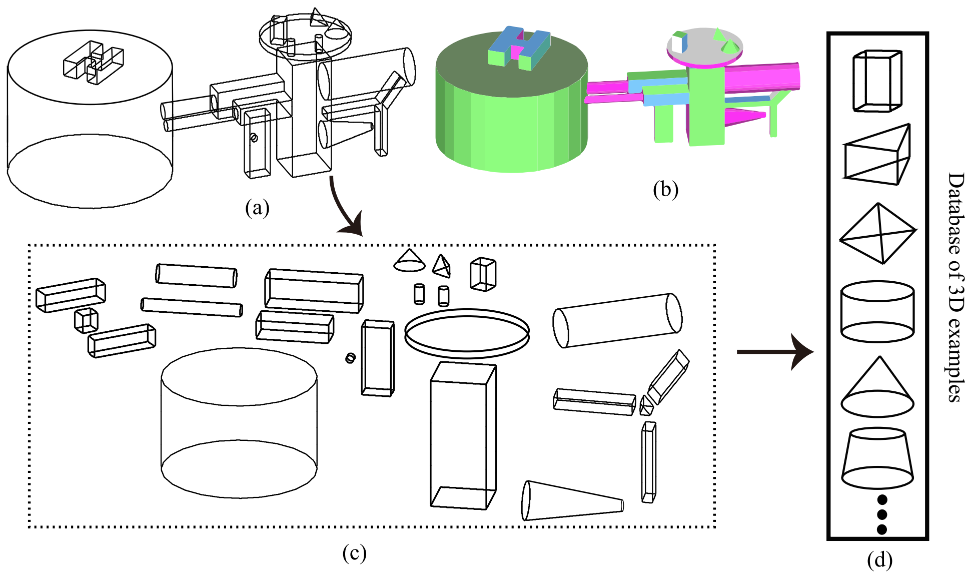

Fig. 1. Illustration of example-based 3D reconstruction from single line drawing. The input 2D line drawing (a) is decomposed into a set of simpler components (c), each of which can be reconstructed by fitting parametric templates from a database (d). These recovered 3D shapes are finally integrated to form the complete 3D model (b).

constraints, (3) reconstruct the 3D geometry for each part group using a novel example-based reconstruction algorithm, and (4) integrate obtained 3D shapes into a final complete 3D object. The core component of the proposed approach is the example-based 3D reconstruction algorithm, where the reconstruction problem is solved with a probabilistic graphical model. The proposed framework is flexible to exploit other reconstruction algorithms (e.g., rule-base approaches) if our example-based algorithm fails to produce satisfactory results.

Our most prominent advantage over previous rule-based methods is the robustness to sketch errors. The ruled-based methods tend to fail to reconstruct plausible 3D geometries from imperfect line drawings, because these rules only based on local information of line drawings and sketch errors may easily violate their assumptions. In the contrary, the proposed method does not directly reconstruct 3D geometry of a line drawing with sketch errors. Instead, we reconstruct 3D geometries by exploiting the most plausible 3D examples (templates) that correspond to the line drawing from the database with a global optimization strategy. Naturally, the proposed approach is more robust than previous rule-based methods.

2. Related work

Here we generally discuss works related to computational interpretation of line drawings, rule-based 3D reconstruction, and 3D reconstruction of curved objects.

Computational interpretation of line drawings: This topic has spanned more than four decades. Generally, existed work can be roughly classified into three categories: (1) line labeling, (2) linear programming based reconstruction, and (3) optimization-based (rule-based) reconstruction. Interested readers may refer to [26–28] for insightful surveys on earlier researches.

Line labeling focuses on finding a set of consistent labels from a line drawing to test the correctness and/or realizability of the line drawing [4,29], but it does not explicitly recover 3D objects. The methods based on *linear programming* [30,31] reconstruct 3D models by solving a linear system which is built from a set of geometrical conditions that the model must fit. In general, linear programming has difficulty to tolerate sketching errors that often exist in a line drawing. Modern methods of 3D reconstruction from line drawings are often *optimization-based*, which usually determine the 3D geometry of a line drawing from the solution that optimizes a certain objective function. Most previous optimization-based 3D reconstruction methods [1,17–19,21,23,32] only focus on planar objects. Being different from those methods mentioned above, the proposed method in the work is a temptation to infer plausible 3D objects, either planar or curved, from single line drawings using a data-driven solution.

Rule-based 3D reconstruction: Rule/regularity based algorithms [7,33–35] are extensively used to reconstruct 3D objects or scenes from single images. A recent survey [36] can be referred to explore that direction. Here, we only discuss algorithms with single drawing as input.

The recent advances [18,19,23] can reconstruct more complex objects than previous works. Ref. [18] finds desired objects in search space of much lower dimensions. This method works well on complex line drawings with low degree of reconstruction freedom (DRF), but increased DRF would make it less robust. Refs. [19,23,37] tried to solve the hard optimization problem caused by the dimensional disaster of a complex line drawing using a divided-and-conquer strategy: first decompose the complex line drawing into a set of parts, then reconstruct 3D shapes from these parts with a rule-based algorithm, finally merge these 3D shapes together to generate the final 3D object.

Our proposed method also utilizes a divided-and-conquer strategy with individual parts reconstructed using example-based algorithm. The example 3D shapes serve as template with internal configurations, which usually lead to better results from line drawings with sketch errors than rule-based algorithms. Most rule-based 3D reconstruction methods assume that the faces are given before the 3D reconstruction. In fact the face identification problem is not trivial [38]. Without such assumption the proposed example-based algorithm infers candidate 3D examples from a pre-defined database, which avoids failure cases caused by unsuccessful face identification.

3D reconstruction of curved objects: Recent research on this topic are mainly rule based on [1,16,22,39]. Lipson and Shpitalni in [1] solved the problem of reconstructing cylindricality by three steps: first approximate a cylindrical face with a set of rectangular faces; then reconstruct 3D geometry of rectangular faces using a rule-based algorithm; finally fit curved faces to these rectangular faces. Wang et al. [22] extended the idea in [1] to reconstruct more complex curved objects. They first distinguish curved faces from planar faces, then convert the line drawings depicting curved objects into generalized polyhedrons, lastly utilize optimization-based approach to produce the final curved 3D objects. Xu et al. presented a sketch-based modeling system in [16] to reconstruct 3D curve networks from 2D sketches, which also employed a rule-based algorithm and require the design drawing to be of good quality. The proposed method in this work uses a similar strategy as in [22] to deal with line drawing parts representing curved shapes: transforming the reconstruction of 3D curved shapes into that of generalized polyhedrons. However we reconstruct the 3D geometry of generalized polyhedrons with an example-based algorithm rather than the rule-based one. Our experiments show that the proposed method can produce better results for line drawings with large degree of sketch errors.

3. Overview

Generally, the main strategy is a procedure of divide-and-conquer, similar to [40]. Refer to Fig. 2 when reading content in this section.

On the *divide* stage, the input line drawing (a) is first decomposed into simpler parts with structural constraints between parts specifying their relative spatial layout (b). Three types of structural constraints are considered in this work – coplanariry C_{planar}, curved touching C_{curved}, and proximity $C_{\text{proximity}}$, which are illustrated as black, red, and green edges respectively. We cluster line drawing parts into *part groups* so that edges within a group are coplanarity constraints while curved contacting and proximity constraints only occur between groups (c). The grouping of parts aims at a bottom-up solution for the followed reconstruction problem.

On the *conquer* stage, we use an example-based algorithm to complete the 3D reconstruction for each part group with candidate 3D examples. While for those without candidate 3D examples (purple blocks) we use a rule-based algorithm [18] to perform the reconstruction. Example-based and rule-based algorithms correspond to $\mathbb{R}_1$ and $\mathbb{R}_2$ respectively. The projection of those 3D shapes obtained from part groups (d) is similar to the input line drawing; however their depth positions might be wrong. Lastly, we obtain the final complete 3D object by adjusting the depth of the 3D shapes corresponding to each part group to satisfy the C_{curved} and $C_{\text{proximity}}$ constraints, see (e). We clarify some important modules in our method as follows.

Input: Our method, with a coverage of both planar and curved objects, operates on an edge-vertex graph representation for the input line drawing. Specifically, the edge-vertex graph representation provides *x*- and *y*-coordinates of all vertices and the information for edges and artificial lines. In our implementation, straight edges are represented by pairs of vertices, and curved edges are represented by Bezier curves.

Decomposition: We use the split face algorithm [41] to perform the line drawing decomposition. Compared to the algorithm in [23] that cannot ensure each separated part depicts a manifold, the split face algorithm produces manifolds with simpler geometry than those in [24]. Though the split face algorithm was proposed for planar objects, it can be extended to separate the curved components from the input line drawing.

3D reconstruction: This is the major module as well as the main contribution of the proposed method. From above discussion we know that 3D reconstruction is a bottom-up process which contains two steps: (1) reconstructing 3D shapes from part groups; then (2) adjusting (translating along the *z*-direction) the positions of those 3D shapes to generate the final objects.

In this paper, we use a bold upper-case letter $\mathbf{X}$ to denote a 3D point, and a bold lower-case letter $\mathbf{x}$ for its 2D projection on the line drawing plane. We denote a 2D line drawing L without curved edges by an undirected graph $G_l = (\{\mathbf{x}^i\}, E)$, where $\{\mathbf{x}^i\}$ are the 2D vertices of L, and E are edges connecting these vertices. The recovered 3D shape S from L retains its topology while vertices become their 3D counterparts, denoted as $G_s = (\{\mathbf{X}^i\}, E)$. In the database, we define a 3D example object M as $G_M = (\{A^i\boldsymbol{\alpha}\}, E_M)$, where $\boldsymbol{\alpha}$ is the parameter vector, A^i is the linear coefficient matrix specifying the position of the ith vertex, and E_M are the example's edges. Under rigid transformation $(\mathbf{R}, \mathbf{t})$, the ith vertex in the 3D example will be transformed to a new location $\mathbf{R}A^i\boldsymbol{\alpha} + \mathbf{t}$.

4. 3D examples

We manually build a 3D example database consisting of 72 planar examples and 40 curved examples, covering most common basic shapes composing man-made objects. For each line drawing part, we find a set of candidate 3D examples that share the same topology. A part with curved edges must match with curved 3D examples. We use the algorithm in [42] to check the isomorphism between two graphs.

To achieve a good generalization ability, we use a set of parameters for each example to represent wider range of candidate 3D shapes. For instance, *cuboid* is determined by three parameters – width a, height b, and depth c; *square frustum* is determined by five parameters – length b and d, width a and c, and height e; and *cuboid* is determined by two parameters – radius r and height d, see Fig. 3.

Given a parametric 3D example in the database, the 3D coordinates of each of its vertices can usually be expressed as a linear function of the parameter vector. For example, vertices $\mathbf{X}_1$ and $\mathbf{X}_2$ on the *square frustum* in Fig. 3(b) can be represented as $[2a, 2b, 0]^\top$ and $[a + c, b + d, e]^\top$, both of which can be reformulated as a product of a 3×5 matrix and the parameter vector $\boldsymbol{\alpha} = [a, b, c, d, e]^\top$ of the example. In general, for a 3D example with m vertices and n parameters, its 3D shape can be determined by m fixed matrices whose elements have a form of $3 \times n$ and an n-dimensional vector containing all the parameters. Therefore, it is easy to know that we can obtain an instance 3D shape of a 3D example by fixing its parametric vector. Note that the representation using a parametric vector is also applicable to a 3D curved object in this paper since it has been approximated by a polyhedron.

For curved examples in the database, we use parametric polyhedrons as alternative representations so that they can fit into our preliminary pipeline in [24]. Accordingly we need to transform the 2D line drawing depicting a curved solid into a new line drawing before the reconstruction. Following [22], a curved edge is approximated by two polygonal lines, where the new *virtual vertices* locate at the curve's *singular points*.[1] For instance, the cylinder in Fig. 4 is approximated by a cuboid, where v_1 and v_2 are two virtual vertices. The main reason to use alternative polyhedron representation for a curved object is that the reconstruction of a polyhedron is much easier than a curved object whose geometry can be uniquely determined by the reconstructed polyhedron.

5. Constraints for reconstruction

Two types of constraints are considered in this work for solving the ill-posed reconstruction problem. First, *projection constraint* states that the projection of the reconstructed 3D objects should

[1] The singular points of a curve are defined as the points having the maximal distance to the straight line passing through the curve's two endpoints.

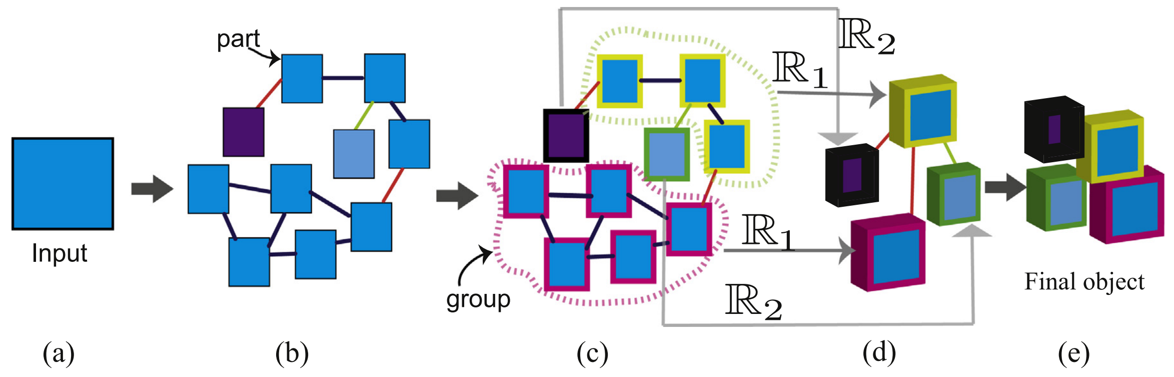

Fig. 2. Overview of the proposed reconstruction framework, see details in Section 3. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this paper.)

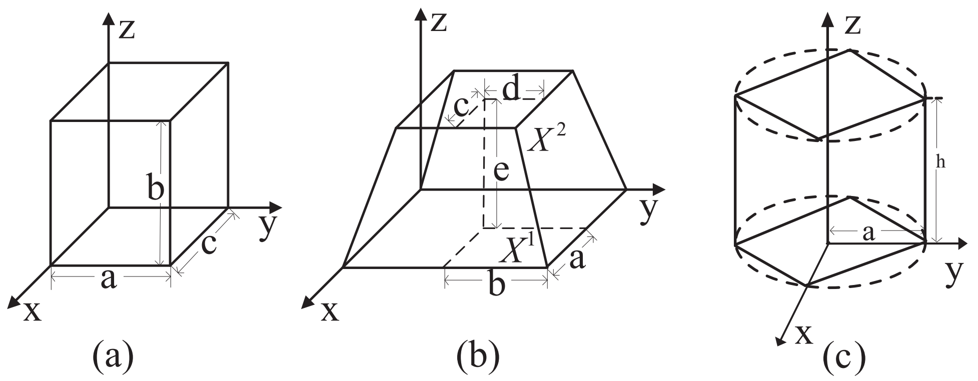

Fig. 3. Three instances from the database of 3D examples: (a) cuboid, (b) square frustum, and (c) cylinder.

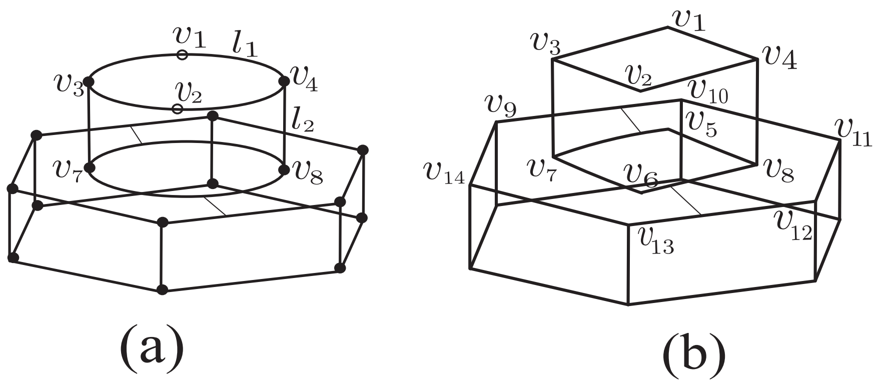

Fig. 4. (a) A line drawing whose edge-vertex graph consists of two connected components, a cylinder on top of a hexagonal prism. (b) By fitting curves in the line drawing with polygonal lines (v_1 and v_2 are virtual vertices), the cylinder is approximated by a cuboid. Artificial lines (thin straight lines) are added to the line drawing to indicate coplanar constraints between faces.

be consistent with line drawings. Second, *structural constraint* are relationships between parts specifying their relative spatial layout in the complete 3D model. We define three types of structural constraints: *coplanarity* (C_{planar}), *curved contacting* (C_{curved}), and *proximity* ($C_{\text{proximity}}$). Both coplanarity and curved contacting constraints are defined between a pair of faces from two different parts: C_{planar} describes the coplanarity between two planar faces, while C_{curved} specifies the contacting constraint between two curved faces or a planar face and a curved face. Fig. 5 shows instances of structural constraints on various line drawings.

Given a planar polygon P and a point $\mathbf{q}$ coplanar with P, we know that $\mathbf{q}$ can be expressed as a linear combination of vertices in P, e.g., Green coordinates [24]. Suppose that two 3D examples, M_1 and M_2, have two coplanar faces, C_1 and C_2. A point on the coplanar plan can be expressed as a linear combination of vertices in C_1 (or C_2), consequently as $B_1\boldsymbol{\alpha}_1$ (or $B_2\boldsymbol{\alpha}_2$), where B_i is the coefficient matrix and $\boldsymbol{\alpha}_i$ is the parameter vector for example M_i. In the 2D line drawing, we encode coplanar vertices in both coplanar faces. Given the assumption of isometric projection, the 3D position of a coplanar vertex can be calculated using the encoded linear representation.

The proximity constraint $C_{\text{proximity}}$ is defined between a pair of parts, and in our implementation is introduced only for isolated parts with no coplanarity or curved contacting constraints with others. Our algorithm automatically detects isolated parts and their closest peer, between which a proximity constraint is added.

The projection and coplanarity constraints will be respected for reconstruction within each part group, see Section 6; while the proximity and curved contacting constraints will guide the integration of 3D shapes recovered from part groups into the final complete 3D model, see Section 7.

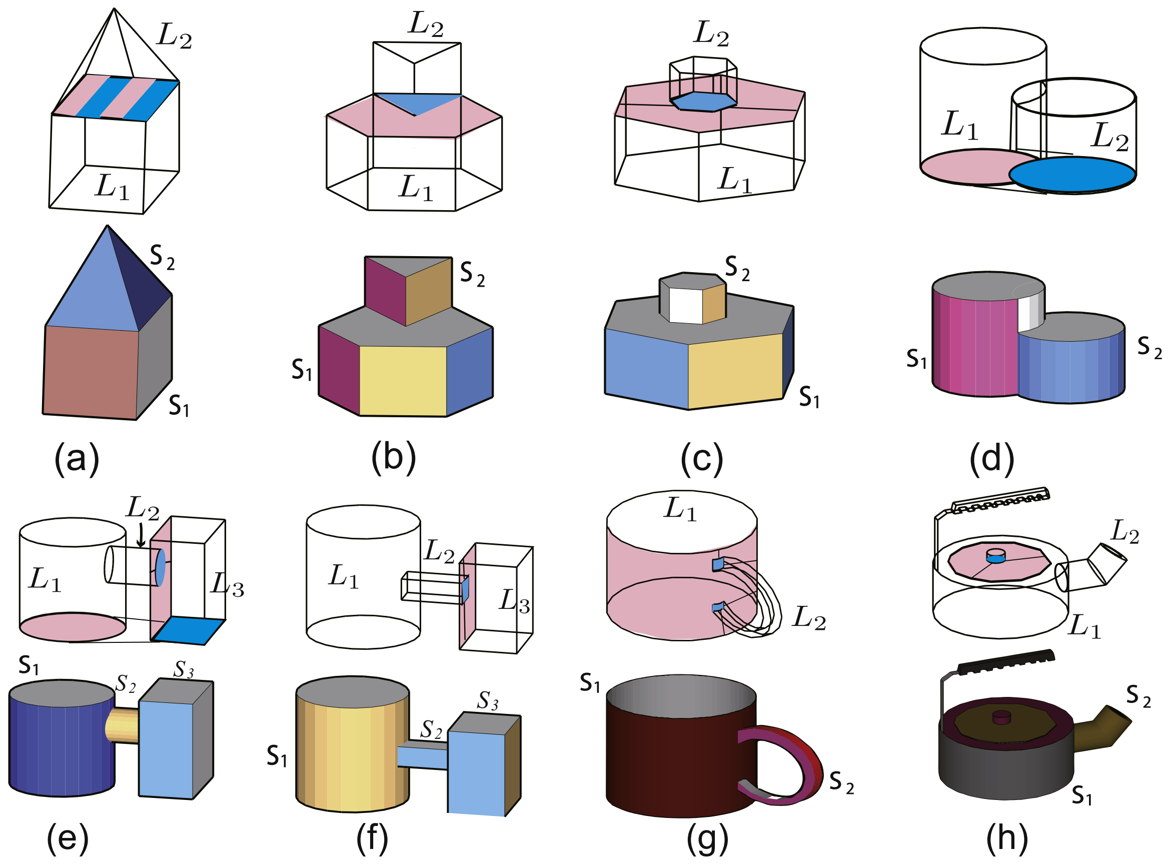

Fig. 5. Illustrations of constraints between line drawing parts. Face pairs constrained by C_{planar} (a)–(f) and C_{curved} (g)–(h) are specified by artificial line pairs and are shadowed by pink–blue color pairs. (L_1, L_2) in (h) are constrained by $C_{\text{proximity}}$. We also show the reconstructed 3D shapes beneath the corresponding line drawings. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this paper.)

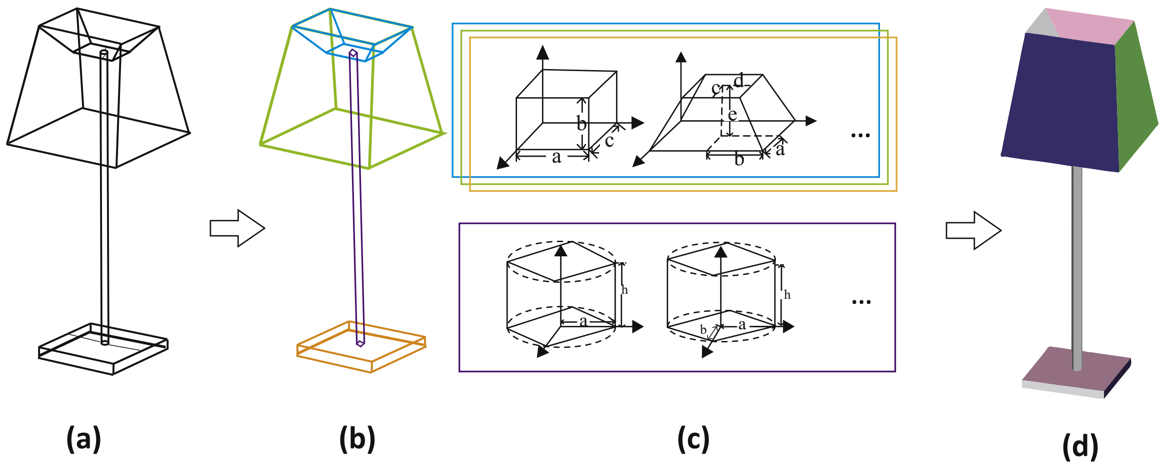

Fig. 6. A line drawing depicting a lamp (a) is decomposed into simpler parts (b), which form a single part group. 3D example candidates (c) are matched from the database to recover individual parts w.r.t. the coplanar constraints between parts. The final result 3D model is shown in (d).

6. Example-based 3D reconstruction

The reconstruction algorithm takes one part group as the input, see Fig. 6. For each part in the group, a set of 3D examples with the same topology as the line drawing part is retrieved from the database. An *undirected graphical model* is proposed to find the best 3D examples that fit all line drawing parts (projection constraint) and follows the coplanarity constraints between parts. The output of this step is a set of 3D examples with corresponding parameters – $\boldsymbol{\alpha}$, $\mathbf{R}$, and $\mathbf{t}$ (Fig. 7).

6.1. Undirected graphical model

The task of the example-based 3D reconstruction is to infer the shape of a 3D part corresponding to each line drawing part L_i. We assume that each 3D part S_i is determined by a set of variables $q_i = \{c_{ik}, \mathbf{R}_{ik}, \mathbf{t}_{ik}\}$, $k = 1, 2, \ldots, n_i$, where n_i is the number of candidate 3D examples for the ith line drawing part, c_{ik} has a boolean value indicating whether the kth example is selected, and ($\mathbf{R}_{ik}$, $\mathbf{t}_{ik}$, $\boldsymbol{\alpha}_{ik}$) are the rotation matrix, the translation vector and the parameter vector for the kth candidate example, respectively.

Given a part group $L = \{L_i\}$, the best choice of $\{q_i\}$ is to

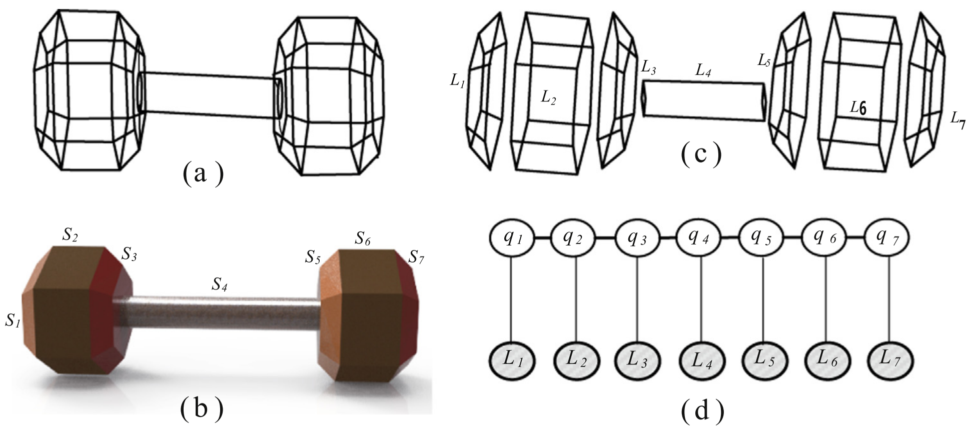
(a)
L_1
L_2
L_3
L_4
L_5
L_6
L_7
(c)
S_2
S_3
S_4
S_1
S_5
S_6
S_7
(b)
q_1 q_2 q_3 q_4 q_5 q_6 q_7
L_1 L_2 L_3 L_4 L_5 L_6 L_7
(d)

Fig. 7. (a) The input 2D line drawing. (b) Reconstructed 3D object. (c) Line drawing parts L_{1-7}. (d) Graphical model for the line drawing where observed nodes L_{1-7} are marked by shadow.

maximize the posteriori probability $P(\{q_i\}|L) \propto P(L|\{q_i\}) \cdot P(\{q_i\})$. To formulate this probability, we assume $\{q_i\}$ possess Markov property and build an undirected graphical model, where each observation node L_i denotes a line drawing part and each latent node q_i denotes the corresponding 3D example S_i. Our graphical model contains two kinds of edges: one type of edge connects L_i and q_i, which ensures the projection constraints, the other type connects two 3D parts q_i and q_j, which ensures the coplanarity constraints.

Given undirected graphical model with Markov property, we have $P(L|\{q_i\}) = \prod_i P(L_i|q_i)$ and $P(\{q_i\}) = \prod_i \phi(q_i) \cdot \prod_{(i,j)\in Q} \psi(q_i, q_j)$, where Q is the set of edges among $\{q_i\}$, $\phi(\cdot)$ and $\psi(\cdot,\cdot)$ are potential functions [43]. Then we can reformulate the posterior probability as

$$P(\{q_i\}|L) \propto \prod_i P(L_i|q_i) \cdot \prod_{(i,j)\in Q} \psi(q_i, q_j) \cdot \prod_i \phi(q_i), \tag{1}$$

whose maximizing procedure is equivalent to

$$\min_{\{q_i\}} \left(\lambda_1 \sum_i E(L_i|q_i) + \lambda_2 \sum_{\{i,j\}\in Q} E(q_i, q_j) + \lambda_3 \sum_i E(q_i) \right), \tag{2}$$

where $E(L_i|q_i) = -\log P(L_i|q_i)$, $E(q_i, q_j) = -\log \psi(q_i, q_j)$, $E(q_i) = -\log \phi(q_i)$, and λ_i controls the weight for each term. For all experiments in this paper, we set $\lambda_1 = 30$, $\lambda_2 = 1$, and $\lambda_3 = 0.05$.

The first term $E(L_i|q_i)$ in (2) is the negative log likelihood term corresponding to the projection constraint, which is defined as

$$E(L_i|q_i) = \sum_{k=1}^{n_i} \left(c_{ik} \sum_{v\in V_i} \| K\mathbf{X}_{ik}^v - \mathbf{x}_i^v \|^2 \right), \tag{3}$$

where V_i is the set of vertices in the basic line drawing part L_i, $\mathbf{X}_{ik}^v = \mathbf{R}_{ik} A_{ik}^v \boldsymbol{\alpha}_{ik} + \mathbf{t}_{ik}$ is the 3D position of the vertex v after rotation and translation, $A_{ik}{}^v$ is the matrix of the candidate 3D model M_{ik} that corresponds to the vertex v, and $\mathbf{x}_i^v$ is the 2D coordinate of the vertex v, and K is the orthogonal projection matrix with the form $K = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}$.

The second term $E(q_i, q_j)$ in (2) corresponds to the coplanarity constraints. Recall that the 3D position of a coplanar vertex can be computed by using encoded linear coefficients on either one of involved 3D examples. Ideally, a coplanar vertex shared by q_i and q_j has exactly the same 3D positions produced by q_i and q_j. Ideally, $E(q_i, q_j)$ would be formulated as a set of equation constraints like $\mathbf{X}_{ik} = \mathbf{X}_{jl}$, which are 3D positions for a shared coplanar vertex produced by q_i and q_j with their kth and lth 3D examples respectively. However, these non-linear equation constraints usually make it hard to find a feasible solution, thus we instead convert them into soft constraint using the following quadratic form:

$$E(q_i, q_j) = \sum_{k=1}^{n_i} \sum_{l=1}^{n_j} \left(c_{ik} c_{jl} \sum_{v\in V(i,j)} \| \mathbf{X}_{ik}^v - \mathbf{X}_{jl}^v \|^2 \right), \tag{4}$$

where $V(i, j)$ is the coplanar vertices shared by q_i and q_j.

The third term $E(q_i)$ denotes the negative prior of q_i. According to Gestalt psychology, one of the most influential theories with a long history asserts that human beings are innately driven to perceive objects as simple as possible [44]. Therefore, it is reasonable to define $E(q_i)$ by the number of parameters η_{ik} in the kth 3D example for q_i as

$$E(q_i) = \sum_{k=1}^{n_i} c_{ik} \eta_{ik}. \tag{5}$$

6.2. Solution for the graphical model

The objective function in (2) is a six-order polynomial and subject to a binary constraint $c_{ik} \in \{0, 1\}$ and a orthogonal constraint $\mathbf{R}_{ik}^{\mathsf{T}} \mathbf{R}_{ik} = I_{3\times3}$. It is not easy to solve this problem directly. To obtain an optimal solution, we design an alternative minimization algorithm and relax the binary constraint $c_{ik} \in \{0, 1\}$ to be a continuous linear inequality constraint $0 \le c_{ik} \le 1$. We first present how to initialize parameters $(c_{ik}, \mathbf{R}_{ik}, \mathbf{t}_{ik}, \boldsymbol{\alpha}_{ik})$ and then detail the steps of alternative minimization.

6.2.1. Initial setting

We initial equal weights for all the n_i candidates for L_i as $c_{i,k} = 1/n_i$. The projection constraint is used to calculate initial values of $\mathbf{R}_{ik}$, $\mathbf{t}_{ik}$ and $\boldsymbol{\alpha}_{ik}$ by solving the optimization problem as follows:

$$\min \sum_{v\in V} \| K(\mathbf{R}A^v\boldsymbol{\alpha} + \mathbf{t}) - \mathbf{x}^v \|^2,$$
$$\text{s. t.} \quad \mathbf{R}^{\mathsf{T}}\mathbf{R} = I, \tag{6}$$

where the subscripts i and k for $\mathbf{R}$, A^v, $\boldsymbol{\alpha}$ and $\mathbf{t}$, and the subscript i for $\mathbf{x}^v$ and V are omitted for conciseness, V is the set of the vertices in L_i, $\mathbf{x}^v$ denotes the 2D vertices in L_i, and I is the identity matrix.

Algorithm 1. Initialization of parameters $\mathbf{R}$, $\mathbf{t}$ and $\boldsymbol{\alpha}$.

```
Input: Generate random values for $\mathbf{R}^{(0)}$, $\mathbf{t}^{(0)}$ and $\boldsymbol{\alpha}^{(0)}$, $i \leftarrow 0$;
  1. Fix $\mathbf{R}^{(i)}$, find optimal $\mathbf{t}^{(i+1)}$ and $\boldsymbol{\alpha}^{(i+1)}$ by solving
       $\begin{cases} f'_{\boldsymbol{\alpha}}(\mathbf{R}, \mathbf{t}, \boldsymbol{\alpha}) = 0, \\ f'_{\mathbf{t}}(\mathbf{R}, \mathbf{t}, \boldsymbol{\alpha}) = 0, \\ \mathbf{t}(3) = 0; \end{cases}$
  2. Fix $\mathbf{t}^{(i+1)}$ and $\boldsymbol{\alpha}^{(i+1)}$, and find optimal $\mathbf{R}^{(i+1)}$ using [45];
  3. If $|f(\mathbf{R}^{(i)}, \mathbf{t}^{(i)}, \boldsymbol{\alpha}^{(i)}) - f(\mathbf{R}^{(i+1)}, \mathbf{t}^{(i+1)}, \boldsymbol{\alpha}^{(i+1)})| > 1e^{-5}$, then
     $i \leftarrow i+1$ and goto step 1;
Return $\mathbf{R}^{i+1}$, $\mathbf{t}^{i+1}$, and $\boldsymbol{\alpha}^{i+1}$;
```

The four steps used to solve Eq. (6) are summarized in Algorithm 1, which utilizes an alternative minimization strategy. Denote the object function in Eq. (6) as $f(\mathbf{R}, \mathbf{t}, \boldsymbol{\alpha})$, which is a quadratic function of $\mathbf{t}$ and $\boldsymbol{\alpha}$ when $\mathbf{R}$ is fixed. We can obtain the minimal value for $\mathbf{t}$ and $\boldsymbol{\alpha}$ by setting two corresponding partial derivatives equal to 0 in step 3. With the assumption of orthogonal projection, the translation along the z-axis ($\mathbf{t}(3)$) is irrelevant to $f(\cdot)$, therefore we set it to 0. In step 4, we alternatively fix $\mathbf{t}$ and $\boldsymbol{\alpha}$ and update $\mathbf{R}$ using the method in [45]. Given obtained initialization of $\mathbf{c}$, $\mathbf{R}$, $\boldsymbol{\alpha}$, and the first two entries of $\mathbf{t}$, we further estimate the initial value for $\mathbf{t}(3)$ by setting the derivative of function (2) w.r.t. $\mathbf{t}(3)$ to 0, which results in a set of linear equations. Then the optimal solution for $\mathbf{t}(3)$ can be computed by solving these linear equations.

Algorithm 2. Finding the optimal solution $\tilde{\mathbf{c}}$, $\widetilde{\mathbf{R}}$, $\tilde{\mathbf{t}}$, and $\tilde{\boldsymbol{\alpha}}$.

```
Input: Initialize $\tilde{\mathbf{c}}^{(0)}$, $\widetilde{\mathbf{R}}^{(0)}$, $\tilde{\mathbf{t}}^{(0)}$, and $\tilde{\boldsymbol{\alpha}}^{(0)}$ using Algorithm 1; $i \leftarrow 0$.
  1. Fix $\widetilde{\mathbf{R}}^{(i)}$, $\tilde{\mathbf{t}}^{(i)}$, and $\tilde{\boldsymbol{\alpha}}^{(i)}$, and find $\tilde{\mathbf{c}}^{(i+1)}$ by solving the quadratic
     programming problem:
       $\min_{\tilde{\mathbf{c}}} g(\tilde{\mathbf{c}}, \widetilde{\mathbf{R}}^{(i)}, \tilde{\mathbf{t}}^{(i)}, \tilde{\boldsymbol{\alpha}}^{(i)}), \quad \text{s. t.} \mathbf{0} \le \tilde{\mathbf{c}} \le \mathbf{1}, \sum_{k=1}^{n_i} c_i(k) = 1.$
  2. Fix $\tilde{\mathbf{c}}^{(i+1)}$, $\tilde{\mathbf{t}}^{(i)}$, and $\tilde{\boldsymbol{\alpha}}^{(i)}$, and find $\widetilde{\mathbf{R}}^{(i+1)}$ by solving
       $\min_{\widetilde{\mathbf{R}}} g(\tilde{\mathbf{c}}^{(i+1)}, \widetilde{\mathbf{R}}, \tilde{\mathbf{t}}^{(i)}, \tilde{\boldsymbol{\alpha}}^{(i)}), \quad \text{s. t. } \widetilde{\mathbf{R}}\widetilde{\mathbf{R}}^{\top} = I.$
  3.Fix $\tilde{\mathbf{c}}^{(i+1)}$ and $\widetilde{R}^{(i+1)}$, and find $\tilde{\mathbf{t}}^{(i+1)}$ and $\tilde{\boldsymbol{\alpha}}^{(i+1)}$ by solving the
    linear equations:
       $\begin{cases} g'_{\tilde{\mathbf{t}}}(\tilde{\mathbf{c}}^{(i+1)}, \widetilde{\mathbf{R}}^{(i+1)}, \tilde{\mathbf{t}}, \tilde{\boldsymbol{\alpha}}) = 0, \\ g'_{\tilde{\boldsymbol{\alpha}}}(\tilde{\mathbf{c}}^{(i+1)}, \widetilde{\mathbf{R}}^{(i+1)}, \tilde{\mathbf{t}}, \tilde{\boldsymbol{\alpha}}) = 0. \end{cases}$
  4.If $|g(\tilde{\mathbf{c}}^{(i)}, \widetilde{\mathbf{R}}^{(i)}, \tilde{\mathbf{t}}^{(i)}, \tilde{\boldsymbol{\alpha}}^{(i)}) - g(\tilde{\mathbf{c}}^{(i+1)}, \widetilde{\mathbf{R}}^{(i+1)}, \tilde{\mathbf{t}}^{(i+1)}, \tilde{\boldsymbol{\alpha}}^{(i+1)})| > 1e^{-5}$,
    then $i \leftarrow i+1$ and go to step 1.
Return $\tilde{\mathbf{c}}^{(i+1)}$, $\widetilde{\mathbf{R}}^{(i+1)}$, $\tilde{\mathbf{t}}^{(i+1)}$, and $\tilde{\boldsymbol{\alpha}}^{(i+1)}$.
```

6.2.2. *Alternative minimization*

After initialization, the solution to (2) is found as follows. For ease description, let $\tilde{\mathbf{c}} = \{c_{ik}\}$, $\widetilde{\mathbf{R}} = \{\mathbf{R}_{ik}\}$, $\tilde{\mathbf{t}} = \{\mathbf{t}_{ik}\}$ and $\tilde{\boldsymbol{\alpha}} = \{\boldsymbol{\alpha}_{ik}\}$, and denote the objective function in (2) as $g(\tilde{\mathbf{c}}, \widetilde{\mathbf{R}}, \tilde{\mathbf{t}}, \tilde{\boldsymbol{\alpha}})$. Although $g(\cdot)$ is a six-order polynomial, it is a quadratic function if any three of $\tilde{\mathbf{c}}$, $\widetilde{\mathbf{R}}$, $\tilde{\mathbf{t}}$, and $\tilde{\boldsymbol{\alpha}}$ are fixed. Using this property, we design an alternative minimization algorithm listed in Algorithm 2, where we alternatively fix three variables in each of the first three steps. After solving (2) in the continuous relaxation version, we obtain the binarized vector $\mathbf{c}_i$ by setting its maximum component to 1 and all the other components to 0.

7. Generation of the final 3D object

After reconstructing the 3D shapes from each part group, we may still not achieve a complete reconstruction. This is because those 3D shapes may not connect with each other in 3D space, despite the fact that the projection of these 3D shapes composes a line drawing similar to the input (the projection of these 3D shapes is ideally the same with the input). Therefore we need to translate those 3D shapes along the z direction to make them connect with each other to generate the final 3D object. Two types of structural constraints, curved contacting and proximity, are considered for adjusting the layout of obtained 3D shapes. The positions $\{o_i\}$ of 3D shapes obtained from part groups are optimized as follows:

$$\min \sum_{(i,j)\in C_{\text{curved}}} \{|\mathcal{M}(o_i, o_j) - \mu| + \alpha\cdot\mathcal{S}(o_i, o_j)\} + \sum_{(i,j)\in C_{\text{proximity}}} |\mathcal{M}(o_i, o_j)|, \tag{7}$$

where $\mathcal{M}(\cdot, \cdot)$ is the minimum translational distances (MTD) [46], defined as the length of the shortest relative translation that results in the objects being in contact, μ is a predefined parameter which controls a desired (normalized) amount of the intersections in C_{curved} structures, $\mathcal{S}(O_i, O_j)$ is a measure for the symmetry, α is the weight to balance the intersection and the symmetry.

Generally, minimizing $|\mathcal{M}(\cdot, \cdot)|$ is to make two shapes to touch and minimizing $|\mathcal{M}(\cdot, \cdot) - \mu|$ is to make two shapes to intersect by a penetration amount μ. In the implementation $\mathcal{S}(\cdot, \cdot)$ is approximated with the angle between normal directions of two contacting faces (at least one is curved). In our experiments, μ is set to 0.05 times of the average size of 3D shapes constrained by $C_{\text{proximity}}$, and the balancing weight is $\alpha = 0.1$. We fix the position of o_1 by setting its depth to 0 and optimize Eq. (7) using a hill climbing algorithm [47].

Existing algorithms for computing MTD of two 3D shapes mainly focus on convex polyhedra, so they cannot be directly used in this work since our shapes might be curved or concave. In our implementation, we approximate curved shapes by their convex hulls and use the algorithm in [48] to compute MTD. The result 3D shape from a complex part group is usually composed of more than one primitive shapes. To compute MTD in this case, we compute MTD to each of its constituent shape and use the minimum MTD as the final result.

8. Experiments and discussion

In this section, we demonstrate the effectiveness of our example-based 3D (E3D) reconstruction algorithm on line drawings that depict various man-made objects. We implemented the proposed method with MATLAB based on our preliminary work published in [24]. Compared to the preliminary version, the proposed method mainly expands the coverage of the 3D reconstruction algorithm from only planar objects to a wider range of objects: both planar and curved objects. Since our preliminary version has conducted some experiments to evaluate the performance (robustness to different levels of sketch errors) of E3D on planar objects, in this paper, we conduct major experiments to evaluate the performance of E3D on curved objects.

The first experiment is designed to evaluate the effectiveness and robustness of the proposed method on line drawings with only coplanarity constraints C_{planar}. The rule-based algorithm [22]

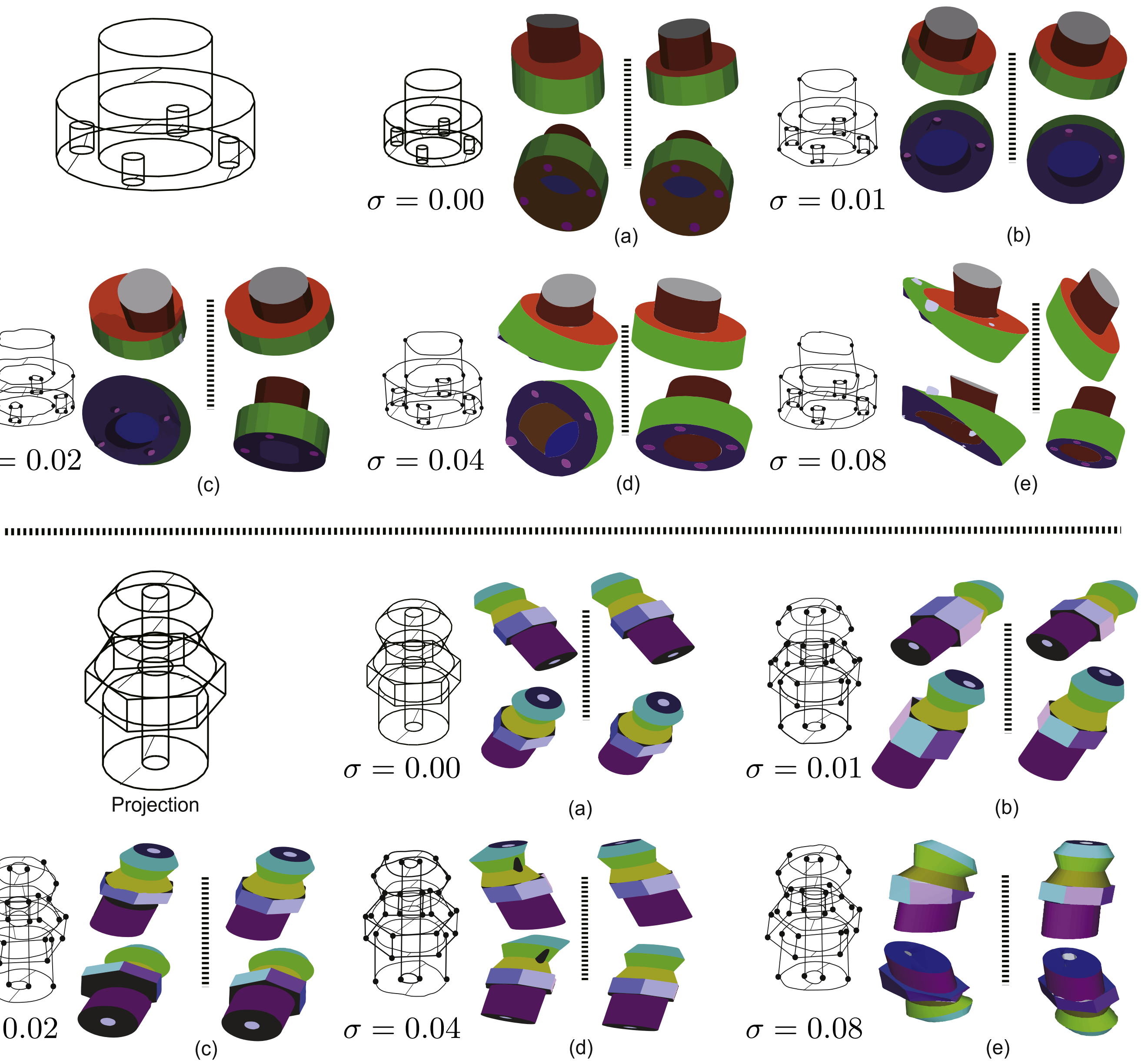

Fig. 8. Comparison between the proposed E3D and rule-based algorithm [22] on two line drawings with only coplanar constraints C_{planar}. We run each test with different noise level σ indicated at the bottom of the disturbed line drawing. Reconstruction results are organized in two columns with two views: rule-based on the left, and E3D on the right.

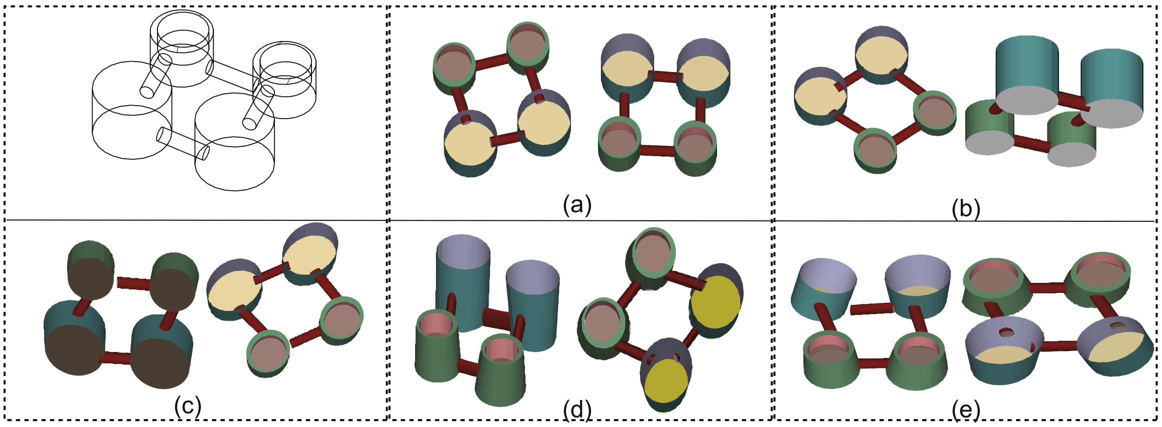

Fig. 9. Reconstruction results on a line drawing with only curved contacting and proximity constraints, C_{curved} and $C_{\text{proximity}}$. The noise level σ for tests in (a)–(e) are 0.00, 0.0.1, 0.02, 0.04, and 0.08, respectively.

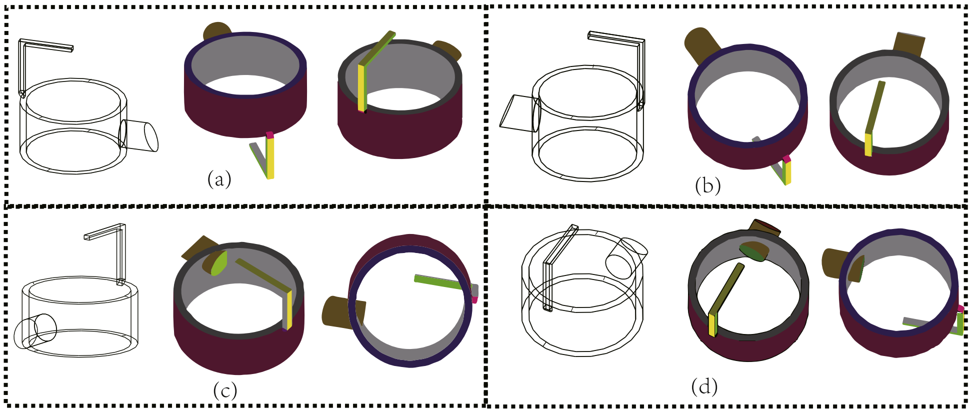

Fig. 10. The reconstruction results for a teapot line drawing from four different perspectives. The teapot body and spout are connected by a proximity constraint $C_{\text{proximity}}$.

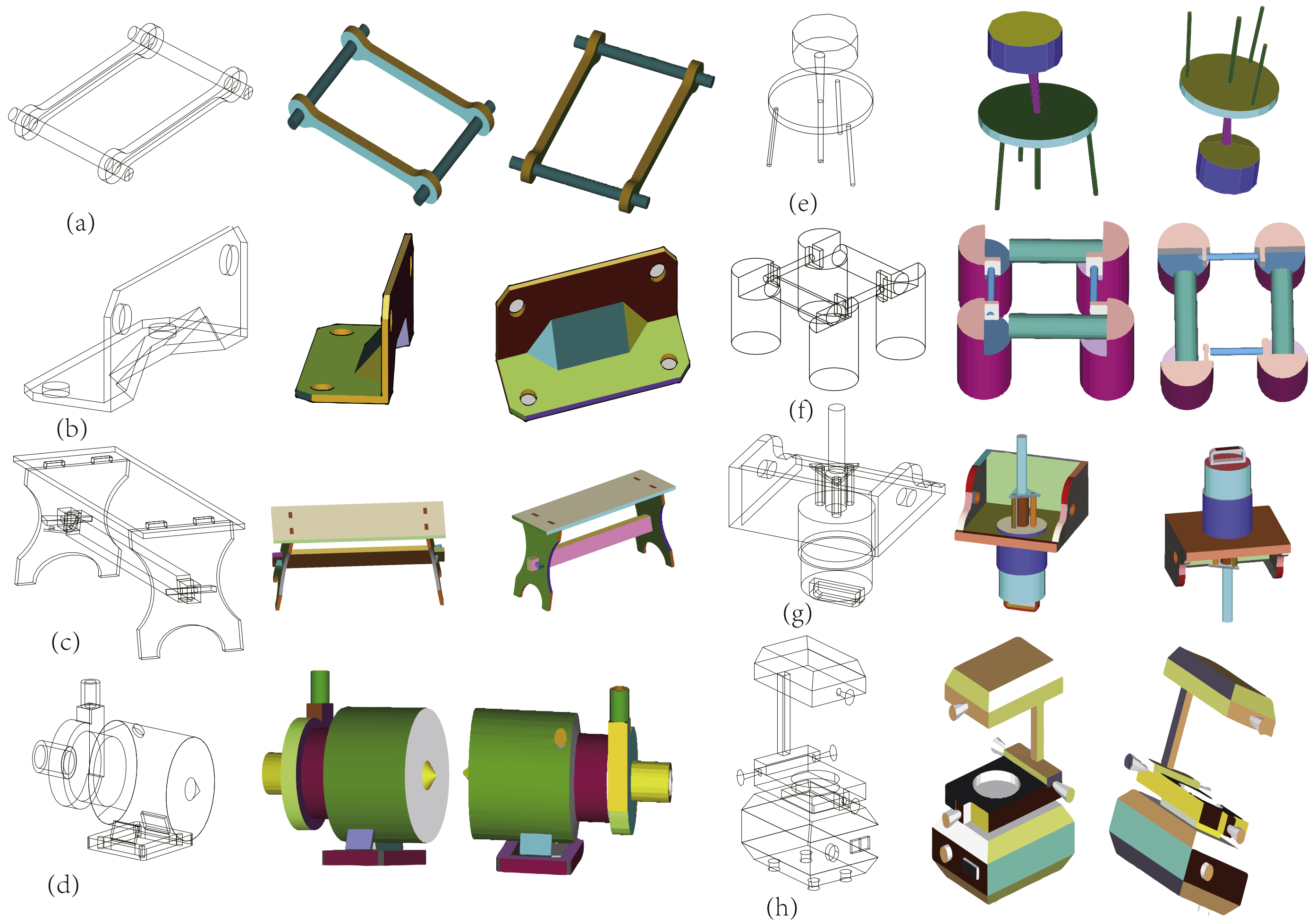

Fig. 11. Reconstruction results of our method on various complex line drawings.

is selected as the baseline. We manually built two 3D objects using AutoCAD, and generated their orthogonal projections as shown in Fig. 8. Both of those objects can be regarded as a combination of a set of simple 3D shapes that have the corresponding 3D examples in the database in our method. We add Gaussian noise to the line drawing with zero mean and five levels of variance $\sigma = \{0.00, 0.01, 0.02, 0.04, 0.08\}$.

Fig. 8 illustrates the reconstruction results of two methods for two line drawings with various levels of noise. Both methods can generate successful reconstructions with sketch error $\sigma \leq 0.02$. However, the rule-based approach fails to obtain plausible reconstructions with sketch error $\sigma \geq 0.04$. In contrast, E3D can still achieve good results even with sketch error $\sigma = 0.08$. This experiment shows that the proposed method is more robust to sketch error on than the rule-based method.

The second experiment is designed to evaluate the performance of E3D on line drawings only with curved contacting and proximity constraints, C_{curved} and $C_{\text{proximity}}$. The rule-based algorithm [22] cannot reconstruct a complete 3D object from such a line drawing due to the existence of isolated parts. In this experiment, we create a 3D object with AutoCAD and generated five line drawings by adding various levels of Gaussian noise. The

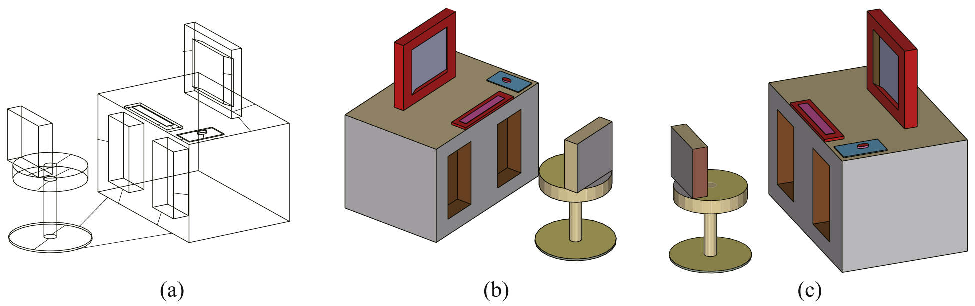

Fig. 12. A 3D scene synthesized by the proposed method.

reconstruction results are shown in Fig. 9. We can see that the proposed method can achieve plausible results even with high level of sketch error $\sigma = 0.08$.

The third experiment studies the influence of view directions for line drawings with only proximity constraints $C_{\text{proximity}}$. We draw a 3D teapot and generate line drawings by projecting the 3D wireframe from four different perspectives. These four line drawings as well as their reconstruction results are shown in Fig. 10. We can see that the connection between teapot body and spout are correctly reconstructed in (a) and (b), while results in (c) and (d) show less plausible connection. The experiment indicates that the proposed E3D approach is more likely to produce plausible results from line drawings with joints shown from a side view.

The forth experiment tests the performance of the proposed method on various line drawings depicting complex man-made objects, see Fig. 11. Some (not all) line drawing parts in (c), (f), and (g) have no corresponding 3D examples in our current database, so their 3D shapes were obtained by the algorithm [22]. The final 3D object is achieved from 3D shapes generated by both the example-based and rule-based algorithms, which show that the framework proposed in this work is compatible.

The last experiment demonstrates the capability of our example-based approach for reconstructing larger scale scenes. Fig. 12 shows the line drawing and reconstructed 3D models of a scene composed of a chair, a desk, and objects on the desk. With the help of a pair of artificial lines on the ground between chair and desk, our method treats the two line drawings depicting a chair and a desk as two parts. Moreover, the proposed method can also be used to reconstruct 3D models from images, which has been demonstrated in our preliminary work [24].

9. Conclusions

The proposed method is the comprehensive version of the work published in [24]. In our preliminary work, we have proposed an example-based 3D reconstruction method to recover planar objects from single line drawings, and demonstrated that the example-based algorithm is much more robust than previous methods for the reconstruction of planar objects. Our contributions in this work are two folds: (1) we extend the example-based algorithm to handle both curved and planar objects; and (2) we proposed a flexible framework that can integrate both example-base and rule-based algorithms into a unique pipeline, which inherently takes the advantages of each integrated reconstruction algorithm. We have designed comprehensive experiments to evaluate the performance of our method . The results in both this work and the preliminary version show that the proposed algorithm can robustly reconstruct both curved and planar complex objects from single line drawings. Moreover, our algorithm also has the potential of reconstructing large scale objects (e.g., scenes) from single line drawings.

Acknowledgments

This work is partially supported by the National Natural Science Foundation of China (Grant nos. 61502153, 61572064, 61272052, and 61473086), the National Basic Research Program of China (2015CB352501), the Program for New Century Excellent Talents in University of Ministry of Education of China, Beijing Municipal Science & Technology Commission (Z161100001616005), the Natural Science Foundation of Hunan Province of China (2016JJ3031), the Construct Program of the Key Discipline in Hunan Province, the Doctorial Start-up Fund of Hengyang Normal University (15B22), the CICAEET fund, and the PAPD fund. The authors are thankful to Dr. Huixuan Tang for her help on proofreading. The authors thank all reviewers for their thorough and constructive suggestions.

References

[1] H. Lipson, M. Shpitalni, Optimization-based reconstruction of a 3D object from a single freehand line drawing, Comput. Aided Des. 28 (8) (1996) 651–663.
[2] H. Lipson, M. Shpitalni, Correlation-based reconstruction of a 3d object from a single freehand sketch, in: ACM SIGGRAPH 2007 Courses, 2007.
[3] M. Masry, D. Kang, H. Lipson, A freehand sketching interface for progressive construction of 3d objects, Comput. Graph. 29 (4) (2005) 563–575.
[4] A. Piquer, M. Contero, F. Naya, A survey on geometrical reconstruction as a core technology to sketch-based modeling, Comput. Graph. 29 (6) (2005) 892–904.
[5] Z. Li, J. Liu, X. Tang, A closed-form solution to 3d reconstruction of piecewise planar objects from single images, in: CVPR, 2007, pp. 1–6.
[6] J. Zheng, Y. Wang, Z. Tang, Solid geometric object reconstruction from single line drawing image, in: GRAPP, 2015, pp. 391–400.
[7] C. Zou, X. Peng, H. Lv, S. Chen, H. Fu, J. Liu, Sketch-based 3-d modeling for piecewise planar objects in single images, Comput. Graph. 46 (2015) 130–137.
[8] L. Cao, J. Liu, X. Tang, 3D object retrieval using 2D line drawing and graph based relevance feedback, in: Proceedings of ACM Multimedia, 2006, pp. 105–108.
[9] T. Funkhouser, P. Min, M. Kazhdan, J. Chen, A. Halderman, D. Dobkin, D. Jacobs, A search engine for 3D models, ACM Trans. Graph. 22 (1) (2003) 83–105.
[10] S. Hou, K. Ramani, Sketch-based 3D engineering part class browsing and retrieval, in: Proceedings of EuroGraphics Workshop on Sketch-Based Interfaces and Modeling, 2006, pp. 131–138.
[11] P. Debevec, C. Taylor, J. Malik, Modeling and rendering architecture from photographs: A hybrid geometry- and image-based approach, in: Proceedings of ACM SIGGRAPH, ACM, New York, 1996, pp. 11–20.
[12] D. Jelinek, C. Taylor, Reconstruction of linearly parameterized models from single images with a camera of unknown focal length, IEEE Trans. Pattern Anal. Mach. Intell. 23 (7) (2001) 767–773.
[13] B. Xu, W. Chang, A. Sheffer, A. Bousseau, J. McCrae, K. Singh, True2form: 3d curve networks from 2d sketches via selective regularization, ACM Trans. Graph. 33 (4) (2014) 131:1–131:13.
[14] A. Shesh, B. Chen, Smartpaper: an interactive and user friendly sketching system, Comput. Graph. Forum 23 (3) (2004) 301–310.
[15] A. Shesh, B. Chen, Peek-in-the-pic: flying through architectural scenes from a single image, Comput. Graph. Forum 27 (8) (2008) 2143–2153.
[16] B. Xu, W. Chang, A. Sheffer, A. Bousseau, J. McCrae, K. Singh, True2form: 3d curve networks from 2d sketches via selective regularization, ACM Trans. Graph. 33 (4) (2014) 131:1–131:13.
[17] Y. Leclerc, M. Fischler, An optimization-based approach to the interpretation of single line drawings as 3D wire frames, Int. J. Comput. Vis. 9 (2) (1992) 113–136.
[18] J. Liu, L. Cao, Z. Li, X. Tang, Plane-based optimization for 3D object

reconstruction from single line drawings, IEEE Trans. Pattern Anal. Mach. Intell. 30 (2) (2008) 315–327.
[19] J. Liu, Y. Chen, X. Tang, Decomposition of complex line drawings with hidden lines for 3d planar-faced manifold object reconstruction, IEEE Trans. Pattern Anal. Mach. Intell. 33 (1) (2011) 3–15.
[20] T. Marill, Emulating the human interpretation of line-drawings as three-dimensional objects, Int. J. Comput. Vis. 6 (2) (1991) 147–161.
[21] K. Shoji, K. Kato, F. Toyama, 3-D interpretation of single line drawings based on entropy minimization principle, in: CVPR, 2001, pp. 90–95.
[22] Y. Wang, Y. Chen, J. Liu, X. Tang, 3d reconstruction of curved objects from single 2d line drawings, in: CVPR, pp. 1834–1841.
[23] C. Zou, S. Chen, H. Fu, J. Liu, Progressive 3d reconstruction of planar-faced manifold objects with drf-based line drawing decomposition, IEEE Trans. Vis. Comput. Graph. 21 (2) (2015) 252–263.
[24] T. Xue, J. Liu, X. Tang, Example-based 3d object reconstruction from line drawings, in: CVPR, 2012, pp. 302–309.
[25] C. Zou, J. Liu, Optimization-based precise reconstruction of 3d objects from line drawings, J. Comput.-Aided Des. Comput. Graph. 24 (12) (2012) 1586–1591.
[26] M. Cooper, Line Drawing Interpretation, Springer, London, 2008.
[27] G. Johnson, M.D. Gross, J. Hong, E.Y. Do, Computational support for sketching in design: a review, Found. Trends Hum.-Comput. Interact. 2 (1) (2009) 1–93.
[28] W. Wang, G.G. Grinstein, A survey of 3d solid reconstruction from 2d projection line drawings, Comput. Graph. Forum 12 (2) (1993) 137–158.
[29] K. Sugihara, Machine Interpretation of Line Drawings, The MIT Press, Cambridge, MA, 1986.
[30] L. Ros, F. Thomas, Overcoming superstrictness in line drawing interpretation, IEEE Trans. Pattern Anal. Mach. Intell. 24 (4) (2002) 456–466.
[31] S. Kyratzi, N.S. Sapidis, From sketch to solid: an algebraic cross-section criterion for the realizability of a wireframe sketch, Computing 86 (2) (2009) 219–234.
[32] E. Brown, P. Wang, 3d object recovery from 2d images: a new approach, in: SPIE Proceedings of Robotics and Computer Vision, 1996, pp. 138–145.
[33] P. Luo, J. He, L. Lin, H. Chao, Hierarchical 3d perception from a single image, in: ICIP, 2009, pp. 4265–4268.
[34] L. Lin, K. Zeng, Y. Wang, W. Hu, 3d structure inference by integrating segmentation and reconstruction from a single image, IET Comput. Vis. 2 (1) (2008) 15–22.
[35] X. Liu, Y. Zhao, S. Zhu, Single-view 3d scene parsing by attributed grammar, in: CVPR, 2014, pp. 684–691.
[36] M.R. Oswald, E. Töppe, C. Nieuwenhuis, D. Cremers, A review of geometry recovery from a single image focusing on curved object reconstruction, in: Innovations for Shape Analysis: Models and Algorithms, 2013, pp. 343–378.
[37] L. Yang, J. Liu, X. Tang, Complex 3d general object reconstruction from line drawings, in: ICCV, 2013, pp. 1433–1440.
[38] J. Liu, Y.T. Lee, A graph-based method for face identification from a single 2d line drawing, IEEE Trans. Pattern Anal. Mach. Intell. 23 (10) (2001) 1106–1119.
[39] K. Cai, H. Zhang, Efficient depth restoration from 2d line drawings with line segments and curves, in: CAD/Graphics, 2015, pp. 49–56.
[40] J. Liu, Y. Chen, X. Tang, Decomposition of complex line drawings with hidden lines for 3d planar-faced manifold object reconstruction, IEEE Trans. Pattern Anal. Mach. Intell. 33 (1) (2011) 3–15.
[41] C. Zou, H. Yang, J. Liu, Separation of line drawings based on split faces for 3d object reconstruction, in: CVPR, 2014, pp. 692–699.
[42] D. Corneil, C. Gotlieb, An efficient algorithm for graph isomorphism, J. ACM 17 (1) (1970) 51–64.
[43] C. Bishop, Pattern Recognition and Machine Learning, Springer, New York, 2006.
[44] S. Palmer, Vision Science: Photons to Phenomenology, The MIT Press, Cambridge, MA, 1999.
[45] Z. Wen, W. Yin, A Feasible Method for Optimization with Orthogonality Constraints, Technical Report, Rice University, 2010.
[46] S. Cameron, R. Culley, Determining the minimum translational distance between two convex polyhedra, in: ICRA, 1986, pp. 591–596.
[47] J. Borghoff, L.R. Knudsen, K. Matusiewicz, Hill Climbing Algorithms and Trivium, vol. 6544, Springer, Berlin, Heidelberg, 2010.
[48] Y.J. Kim, M.C. Lin, D. Manocha, Incremental penetration depth estimation between convex polytopes using dual-space expansion, IEEE Trans. Vis. Comput. Graph. 10 (2) (2004) 152–163.

Changqing Zou received the B.E. degree from Harbin Institute of Technology, China, in 2005, and the M.E. degree from Institute of Remote Sensing and Digital Earth, Chinese Academy of Sciences, China, in 2008, and the Ph.D. degree at the Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, China, in 2015. He is now a Postdoctoral Research Fellow in Simon Fraser University, Canada. He is also currently an Assistant Professor in the College of Computer Science and Technology, Hengyang Normal University, China. His research interests include computer vision and computer graphics.

Tianfan Xue received the B.E. degree in Computer Science and Technology from Tsinghua Universtiy, Beijing, China, in 2009, and M.Phil. degree in Computer Vision from The Chinese University of Hong Kong, Hong Kong, in 2011. He has been working toward the Ph.D. degree in Massachusetts Institute of Technology (MIT), USA since 2012. His current research interests include computer vision and machine learning.

Xiaojiang Peng received his B.S. degree in Communication Engineering at Kunming University of Science and Technology in 2009. Then he completed Ph.D. degree in Computer Science at Southwest Jiaotong University in 2014. He was an internship student at Shenzhen Institute of Advanced Technology, Chinese Academy of Sciences from 2013 to 2014. His research focus is in the area of action recognition and deep learning. Currently, he is an Assistant Professor in Hengyang Normal University in China.

Honghua Li received his B.S. degree in Computing Science at National University of Defence Technology in 2008. Then he completed his Ph.D. degree in Computer Science at Simon Fraser University in 2015. His research focus spans from geometry processing, semantic shape analysis, to 3D modeling and printing. He currently holds a visiting scholar position in Shandong University.

Baochang Zhang received the B.S., M.S., and Ph.D. degrees in Computer Science from the Harbin Institute of Technology, Harbin, China, in 1999, 2001, and 2006, respectively. From 2006 to 2008, he was a Research Fellow with the Chinese University of Hong Kong, Hong Kong, and Griffith University, Natant, QLD, Australia. He is an Associate Professor with the Science and Technology on Aircraft Control Laboratory, School of Automation Science and Electrical Engineering, Beihang University, Beijing. He was selected by the Program for New Century Excellent Talents at the University of Ministry of Education of China in 2013.

Ping Tan received his Ph.D. degree in Computer Science & Engineering from the Hong Kong University of Science and Technology in 2007, and his Master and Bachelor degrees Shanghai Jiao Tong University, China, in 2000 and 2003 respectively. And now he is an Assistant Professor at the School of Computing Science at Simon Fraser University (SFU).

Jianzhuang Liu received the Ph.D. degree in computer vision from The Chinese University of Hong Kong, Hong Kong, in 1997. From 1998 to 2000, he was a research fellow with Nanyang Technological University, Singapore. From 2000 to 2012, he was a Postdoctoral Fellow, then an Assistant Professor, and then an Adjunct Associate Professor with The Chinese University of Hong Kong. He joined Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, as a Professor, in 2011. He is currently a Chief Scientist with Huawei Technologies Co. Ltd., Shenzhen, China. He has published more than 100 papers, most of which are in prestigious journals and conferences in computer science. His research interests include computer vision, image processing, machine learning, multimedia, and graphics.