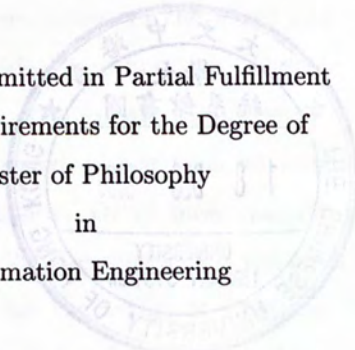


3D Object Reconstruction from Line Drawings

CAO Liangliang

A Thesis Submitted in Partial Fulfillment
of the Requirements for the Degree of
Master of Philosophy
in
Information Engineering



©The Chinese University of Hong Kong
June, 2005

The Chinese University of Hong Kong holds the copyright of this thesis. Any person(s) intending to use a part or the whole of the materials in this thesis in a proposed publication must seek copyright release from the Dean of the Graduate School.



Abstract

A line drawing is defined as the 2D projection of the edges and vertices of a 3D object in a generic view, with or without hidden lines invisible. The line drawings might be the most straightforward and easiest way to illustrate 3D objects. Since the early stage of computer vision, understanding line drawings has been an important research area and invites numerous considerations and efforts. Of all these research topics on line drawings, reconstructing 3D objects from single 2D line drawings is the most important and difficult problem. The human vision system has the ability to interpret 2D line drawings as 3D objects without difficulty. However, such a task is not easy for a computer vision system. Researchers have presented some algorithms to recover 3D objects from line drawings, but the pervious work has two limitations. First, none of the previous methods can recover a complete object if the hidden lines of the object are not shown. Second, most of them can only handle line drawings of simple objects with planar faces. Reconstruction of curved objects remains an unsolved problem. Most of the work in this thesis is devoted to conquer these two limitations existing in the previous work.

To reduce the first limitation, we show that given some constraints, the topology of the hidden parts in a line drawing could be effectively inferred by an algorithm proposed by us. Then we can employ a reconstruction method based on perceptual symmetry and planarity of the object to recover the 3D shape of both the visible parts and the hidden parts of the object.

Our second contribution is to generalize the reconstruction from line drawings of objects with only planar faces to the objects with both planar and curved faces. Our approach includes (1) transforming the line drawing of an object with curved faces into one with planar faces only, (2) identifying the faces of the transformed line drawing, (3) reconstruction of a planar object from the transformed line drawing using the former methods, and (4) generation of the curved faces by developing Bezier curves and surface patches. In addition it also provides an interface to recover 3D models from images. To the best of our knowledge,

our work is the first to provide such reconstruction from single line drawings.

Although our approach can carry out the 3D reconstruction for a large range of objects, the reconstruction of curved objects from single view is not fully solved. To reconstruct more complex objects, the nature and properties of general curved surface model should be explored. In the last part of this thesis, we define and study a new class of curved objects, called Degen generalized cylinders (DGCs) with the help of the surface model proposed by Degen for geometric modelling. The research of DGCs is derived from the discussion of the conditions when straight homogeneous generalized cylinders (SHGCs) and tori have planar limbs. Then, through the rigorous discription based on projective geometry and homogeneous coordinates, we prove that DGCs enjoy invariant properties that reveal the relations among the planar limbs and view directions, which generalize the work of D. Marr and J. Ponce. DGCs include tori, quadrics, cyclides, SHGCs and some other curved objects into a unified model. We expect the research of DGCs will benefit the reconstruction of curved objects in our future work.

摘要

三維物體的頂點和邊在二維平面上的投影構成了綫條畫。在投影中，可以選擇畫出或者不畫出不可見的邊，由此構成的綫條畫作為一種簡捷直觀的顯示三維物體的方法。理解綫條畫是計算機視覺中的一個重要的研究課題。人眼可以輕鬆的從二維綫條畫中感覺到物體的三維形狀，但是計算機視覺要完成類似的功能並非同樣輕鬆。已經有一些研究工作能夠成功的從二維綫條畫中重建三維物體，但這些現有的工作都局限于平面物體，並且要求所有的不可見的邊都在綫條畫中顯示出來。從綫條畫中重建曲面物體，或者從沒有不可見邊的綫條畫中重建三維物體，都超出了現有算法的能力範圍。在本篇論文中，我們集中討論如何克服這兩方面局限。

為了克服第一種局限，我們證明了如果給定一些假設限制，綫條畫中的不可見部分的拓撲形狀可以被一種新算法有效的推斷出來，隨之我們採用了基於視覺對稱性和共面限制的重建算法來同時恢復可見部分和不可見部分的三維形狀。

我們第二方面的貢獻是關於從綫條畫中重建三維物體。這些物體可以同時具有平面和曲面。我們的方法分為四步：（1）將曲面物體的綫條畫變換成只含有平面的綫條畫，（2）識別變換後物體的各個平面，（3）根據以前的方法重建平面物體，（4）最後根據重建的平面物體，再利用貝塞爾曲面元重建原曲面物體。更進一步的，這種新方法提供了一種從照片或者圖像中購造三維物體的工具。據我們所知，我們的工作是第一個將從單個綫條畫中重建平面物體推廣到曲面物體的方法。

儘管我們的算法在很多例子上取得了成功，它並不能徹底地解決所有曲面三維物體的重建問題。為了處理更複雜的曲面物體，我們需要研究廣泛的更有代表性的曲面模型。本論文的最後一部分將討論並定義這樣一種新的曲面模型：德根廣義椎。這個模型推廣自德根的幾何建模工作。這方面的研究來源於關於圓環面和直諧廣義椎產生平面曲綫投影的條件的研究。我們使用投影幾何和廣義坐標證明了德根廣義椎具有一些投影不變量，這些不變量揭示了它們的平面曲綫投影和視角之間的關係。我們的結論推廣了馬爾和龐斯以前的研究工作。德根廣義椎將圓環面，常見二次曲面，四次曲面，直諧廣義椎等統一為一個模型，我們期望這方面的工作會對我們將來的關於重建等方面的工作有所幫助。

Acknowledgement

First and foremost I would like to thank my supervisor Dr. Xiaoou Tang, and Dr. Jianzhuang Liu. It is Dr. Tang's encouragement and direction that makes me be proud in research and not lose impetus of continuous working. During the two years in Multimedia lab I am deeply impressed by Dr. Tang's insight and enthusiasm in the work. I am grateful to work with a supervisor with such a nice personality.

I am also grateful to work with Dr. Liu, who is always glad to talk with students and always like to teach something to the students. I owe much to Dr. Liu, not only because he taught me to write papers words by words and sentences by sentences, not only because of the nice patience and constructive suggestions for almost every ideas of mine, not only because he gave me much instruction on swimming and tennis, but also because he teach me the to be serious and patient in research, which is so important for me with a blundering character.

I am lucky to have the chances to work with a group of ingenious labmates. Many Thanks to Feng Lin, Lifeng Sha, Feng Zhao, Zhifeng Li, Dacheng Tao, Tianqiang Yuan, Xiaogang Wang, Hao Liu, Dahua Lin, Dong Xu, Wei Liu. Being in such a productive group I have the opportunity to get in touch with many aspects of the state-of-art research, and I hope I could be as prosperous as the current labmates in the future. And I would like to thank Shifeng Chen and Peng Du, with the discussion of whom I could understand Graph-cut segmentation and matting quickly, and the collaborating with Shifeng on medical image is such a enjoyable experience.

I should acknowledge the friends in CUHK, to my roommate, dinnermates, badmiton-mates, basketballmates, for the happy time together. At last, eternal thanks to my parents, for always supporting me, no matter what I chose to do. And also, to dear Tangya, meeting you is the most important part of my life.

Contents

1	Introduction and Related Work	1
1.1	Reconstruction from Single Line Drawings and the Applications	1
1.2	Optimization-based Reconstruction	2
1.3	Other Reconstruction Methods	2
1.3.1	Line Labeling and Algebraic Methods	2
1.3.2	CAD Reconstruction	3
1.3.3	Modelling from Images	3
1.4	Finding Faces of Line Drawings	4
1.5	Generalized Cylinder	4
1.6	Research Problems and Our Contribution	5
1.6.1	A New Criteria	5
1.6.2	Recover Objects from Line Drawings without Hidden Lines	6
1.6.3	Reconstruction of Curved Objects	6
1.6.4	Planar Limbs Assumption and the Derived Models	6
2	A New Criteria for Reconstruction	8
2.1	Introduction	8
2.2	Human Visual Perception and the Symmetry Measure	10
2.3	Reconstruction Based on Symmetry and Planarity	11
2.3.1	Finding Faces	11
2.3.2	Constraint of Planarity	11
2.3.3	Objective Function	12
2.3.4	Reconstruction Algorithm	13
2.4	Experimental Results	13
2.5	Summary	18

3	Line Drawings without Hidden Lines: Inference and Reconstruction	19
3.1	Introduction	19
3.2	Terminology	20
3.3	Theoretical Inference of the Hidden Topological Structure	21
3.3.1	Assumptions	21
3.3.2	Finding the Degrees and Ranks	22
3.3.3	Constraints for the Inference	23
3.4	An Algorithm to Recover the Hidden Topological Structure	25
3.4.1	Outline of the Algorithm	26
3.4.2	Constructing the Initial Hidden Structure	26
3.4.3	Reducing Initial Hidden Structure	27
3.4.4	Selecting the Most Plausible Structure	28
3.5	Reconstruction of 3D Objects	29
3.6	Experimental Results	32
3.7	Summary	32
4	Curved Objects Reconstruction from 2D Line Drawings	35
4.1	Introduction	35
4.2	Related Work	36
4.2.1	Face Identification	36
4.2.2	3D Reconstruction of planar objects	37
4.3	Reconstruction of Curved Objects	37
4.3.1	Transformation of Line Drawings	37
4.3.2	Finding 3D Bezier Curves	39
4.3.3	Bezier Surface Patches and Boundaries	40
4.3.4	Generating Bezier Surface Patches	41
4.4	Results	43
4.5	Summary	45
5	Planar Limbs and Degen Generalized Cylinders	47
5.1	Introduction	47
5.2	Planar Limbs and View Directions	49
5.3	DGCs in Homogeneous Coordinates	53
5.3.1	Homogeneous Coordinates	53
5.3.2	Degen Surfaces	54
5.3.3	DGCs	54

5.4	Properties of DGCs	56
5.5	Potential Applications	59
5.5.1	Recovery of DGC Descriptions	59
5.5.2	Deformable DGCs	60
5.6	Summary	61
6	Conclusion and Future Work	62
	Bibliography	64

List of Figures

1.1	An example of Reconstruction from Line drawings. (a) A 2D line drawing. (b) Reconstructed 3D object. (c) Reconstructed 3D object in another view.	1
1.2	Our contributions. (a) Reconstruction from a line drawing without hidden lines. (b) Reconstruction from a line drawing as a curved objects.	6
2.1	Infinite 3D objects that can project to the same line drawings.	10
2.2	Three line drawings that are not included in our experiments.	14
2.3	Reconstruction results for the line drawings in [48], [37], [10], [70].	15
2.3	<i>Continued.</i>	16
2.4	Reconstruction results for a new set of line drawings.	17
2.5	Two different 3D objects that give the same value of the PEMSDA objective function.	18
3.1	(a) A line drawing without hidden lines. (b) The inferred hidden topological structure. (c) Reconstructed complete 3D object.	20
3.2	Illustration of some terms. Here v_{1-9} are complete vertices, v_{10-14} are incomplete vertices, and v_{15} is a broken vertex. Edge (v_{11}, v_9) is a boundary edge, and (v_1, v_{15}) is a zero edge. Cycle $(v_{10}, v_5, v_{11}, v_9, v_{12}, v_8, v_{10})$ is a boundary cycle.	21
3.3	Part of a line drawing where v_1, v_0 and v_2 are collinear.	22
3.4	Walking along boundary edges. (a) A boundary cycle. (b) An impossible path.	25
3.5	(a) A line drawing with a zero edge. (b) Two boundary cycles (bold) with incomplete vertices (filled-in circles) on them.	25

3.6	An example of inferring the hidden topology from line drawing. (a) A line drawing. (b) Finding the boundary edges (the bold cycle) and the incomplete vertices (filled-in circles). (c) Initial hidden vertices (open circles). (d) Reduction of the initial hidden vertices.	26
3.7	Illustration of the cutting and merging procedure where $a-e$ are incomplete vertices, v_{1-5} are hidden vertices, and $(v_1, v_2, v_3, v_4, v_5, v_1)$ is a hidden cycle. (a) The initial hidden structure. (b) Cutting one edge (v_1, v_5) , (c) Merging v_1 to v_2 and v_5 to v_4 . (d) Cutting two edges (v_1, v_5) and (v_3, v_4) . (e) Merging v_1 and v_3 to v_2 ; v_5 and v_4 disappearing after being merged. (f) Cutting two edges (v_1, v_5) and (v_4, v_5) , where v_5 has no place to be merged.	27
3.8	The initial hidden structure and all others derived by the cutting-and-merging procedure.	28
3.9	Results of the inference and reconstruction.	33
3.9	Continued: Results of the inference and reconstruction.	34
4.1	Two line drawings and their reconstructed shapes. Line drawings may be inputted from sketches, images, digitizer tablets, and the screen.	36
4.2	Examples of the transformation of line drawings, where hidden lines are displayed in dashed for easier observation.	38
4.3	Bezier surface patch $S(u, v)$ and its control points.	41
4.4	A set of line drawings and their reconstructed results, each shown from two views.	44
4.5	Reconstruction of a car from a color image. (a) Boundaries drawn by the user as the line drawing. (b) (c) Two views of the recovered 3D car with texture mapped. (d) (e) Two views of the 3D car without texture	45
5.1	2D Projections unable to fully describe the 3D information of the space curve.	49
5.2	(a) Planar limbs. (b) Non-planar limbs.	50
5.3	The coordinate system with a SHGC and the viewing direction \mathbf{v}	50
5.4	A torus with the viewpoint at the z -axis.	52
5.5	Some Degen Surfaces, among which (a), (b), (c) and (d) are a SHGC, an open torus, a cyclide and a quadric, respectively.	55
5.6	A Degen surface with neither u -curves nor v -curves closed.	56
5.7	Illustration of Proposition 5.5.	58
5.8	Illustration of Proposition 5.6.	58
5.9	Illustration of Corollary 5.6.	59

Chapter 1

Introduction and Related Work

1.1 Reconstruction from Single Line Drawings and the Applications

A line drawing is defined as a 2D projection of the edges and vertices of a 3D object in a generic view, with or without hidden lines invisible. The human vision system has the ability to interpret 2D line drawings as 3D objects without difficulty. Emulating this ability is an interesting research topic for enhancing machine vision system. Fig. 1.1 shows an example of such reconstruction.

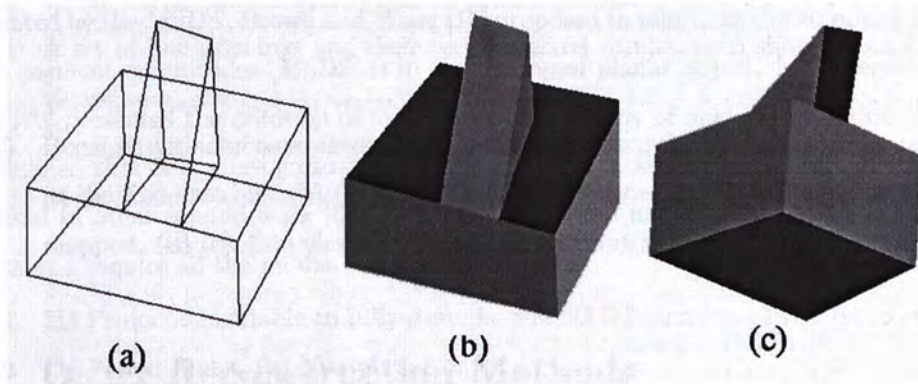


Figure 1.1: An example of Reconstruction from Line drawings. (a) A 2D line drawing. (b) Reconstructed 3D object. (c) Reconstructed 3D object in another view.

With the development of computer vision research and the increasing requirement of interaction between human and computer, the reconstruction from line drawings find more applications, such as

- flexible sketching input for conceptual designers who tend to prefer pencil and paper to mouse and keyboard in current CAD systems [3], [40], [71], [69],

- automatic conversion of existing industrial wireframe models to solid models [3], [2],
- providing rich databases to object recognition systems and reverse engineering algorithms for shape reasoning [2], [3], [78],
- interactive generation of 3D models from images [20], [81], [75],
- user-friendly query interface for 3D object retrieval from large 3D object databases and the Web, which lets users easily draw 3D objects as queries [56], [51], [92].

1.2 Optimization-based Reconstruction

To emulate the human vision system's ability to interpret 2D line drawings as 3D objects, some researchers formulated the 3D reconstruction as an optimization problem based on different objective functions. Marill proposed a criterion, minimizing the standard deviation of the angles (MSDA) in a reconstructed object to emulate human 3D perception of 2D line drawings [48]. This idea is followed by the researchers in [40], [69], [37], [70]. This approach is tolerant of freehand sketching errors, but cannot reconstruct complete 3D objects if their hidden lines are not drawn.

Marill [48] presented his method based on a simple criterion: minimizing the standard deviation of the angles in the reconstructed object, which is called the MSDA principle. Motivated by the MSDA, Brown and Wang [10] proposed to minimize the standard deviation of the segment magnitudes (MSDSM) in the recovered planar object. More recently, Shoji et al. [70] presented the criterion of minimizing the entropy of angle distribution (MEAD), and claimed that it is more general than both the MSDA and the MSDSM. The MSDA is also used in other related work [69], [37], [71]. All these methods can handle only planar objects and require all the hidden lines to be drawn.

1.3 Other Reconstruction Methods

There are still many other papers on the interpretation of line drawings in the literature, which are not so related to our work as those mentioned in the last chapter. We give a brief review of these reconstruction work in this section.

1.3.1 Line Labeling and Algebraic Methods

A large body of the work is about line labeling and 3D reconstruction based on a labeled drawing [30], [17], [90], [28], [46], [18], [77], [76], [31]. Line labeling focuses on finding a set

of consistent labels from a line drawing without hidden lines, and does not explicitly give the 3D structure represented by a line drawing.

Since the early stage of computer vision, a large amount of work called line labeling has been carried out for line drawing interpretation [17], [30], [28], [46]. Line labeling focuses on finding a set of consistent labels from a line drawing without hidden lines, and does not explicitly give the 3D structure represented by a line drawing.

Another body of work on line drawing interpretation is related to judging the correctness of line drawings and give their possible reconstruction based on algebra test with linear equalities and inequalities [45], [77], [76], [88], [64]. The problem of these methods is that such a formulation is superstrict and not robust; an originally correct line drawing will be judged as *impossible* after a little deviation of one or more vertices, causing a 3D reconstruction to fail [64].

1.3.2 CAD Reconstruction

Another body of the work is in the reconstruction of a 3D CAD model from the model's three orthogonal projections [38], [44], [1], [35]. Because more information is available, such work is much easier than the reconstruction from one projection.

1.3.3 Modelling from Images

Tour into the image (TIP) proposed by Horry et al. [29] and later improved by Kang et al. [32] constructs a simple 3D scene model using vanishing points and a spidery mesh. Their works rely on the directly designation of vanishing points and spidery meshes by the user. Although TIP is easy to use and can achieve impressive results, but not strict in geometry and limited to the implied situation where the image plane must be vertical to the ground plane.

More strictly reconstruction work are implored in the spirits of single view metrology (SVM)[39], [75], [36]. Their methods are based on calibration of camera intrinsic parameters, thus do not restrict to the assumption that the image plane is vertical to the image plane. However, such methods requires a lot of human interaction. Such system requires the user not only to set enough constraints to decide the camera intrinsic parameter, but also give hints to every plane or vertex. For example, to define a point, the user must tell the vertical projection of that point on to the ground plane or another reference plane [39],[36]. To define a plane, the user should tell its vanishing line. Such user inputs are burdensome since it is not much easier than input an object using traditional 3D modeler such as AutoCAD. Our

work is different with those work. We requires no human interactions and our reconstruction methods process single line drawings instead of images.

1.4 Finding Faces of Line Drawings

Face identification from a line drawing provide necessary information for reconstructing 3D objects. An object consists of faces. If the face configuration of an object is known before the reconstruction of its 3D geometry, the complexity of the reconstruction will be reduced significantly.

In general, there are many cycles in a line drawing and only a small subset of them represents its faces, and the number of cycles grows exponentially with the number of edges. Thus finding the faces from a line drawing is not a trivial problem. Much effort has been made in this area [71], [42], [69], [3], [2], [37], [43]. Among these techniques, the algorithm presented in [42] and [43] are useful for our work.

1.5 Generalized Cylinder

A challenging problem in reconstructing 3D objects from line drawings is to handle curved objects since almost all the previous optimization-based reconstruction are limited to planar objects. Thus it might be necessary to study the curve surface models. Generalized cylinder (GC) is one of the most important models studied in computer vision.

A generalized cylinder (GC) is a solid obtained by sweeping a planar region along an axis. The planar region is called the cross section of the GC and is not necessarily circular or constant. The axis can also be curved in space. This model was at first proposed by Binford in 1971 [6], and has received extensive attention and become popular in computer vision in the past three decades. Because of their ability to represent objects explicitly and their object-centered coordinate frames derivable from image data, GCs have been applied to shape recovery [65], [23], [87], [66], [61], object modelling [53], [54], [26], [68], model-based segmentation and detection [94], [27], [63], modelling tree branches in computer graphics [7], recovering blood vessels from medical images [55], designing robot vision systems [9], etc.

From previous work on the study of the properties and recovery of GCs, we can roughly divide GCs into two groups: GC with straight axes and GCs with curved axes. In what follows, we call them straight GCs and curved GCs, respectively. Most of the work considers GCs in single views. Straight homogeneous generalized cylinders (SHGCs) are the most important subset of straight GCs, whose sweeping axes are straight and whose cross sections

are scaled along the axes. SHGCs were first defined by Shafer and Kanade [67], and then studied extensively by many researchers [65], [87], [61], [94], [27], [60], [82], [85], [86].

Compared with SHGCs, less work on curved GCs has been carried out. The difficulty is mainly due to two facts: the projection of the axis of a curved GC may not be necessarily the axis of its 2D contours [59], and the angle between the axis and the cross section in the image no longer keeps constant [95]. To interpolate the axis of a curved GC in scattered data, Shani and Ballard proposed an iterative solution of minimizing the torsion of the axis [68]. In [66], Sayd et al. presented a scheme to recover a constrained subset of curved GCs with circular and constant cross sections. Ulupinar and Nevatia focused on a subset of GCs whose axes are planar curves¹ and normal to the constant cross sections [84]. Zerroug and Nevatia studied the invariants and quasi-invariants of a subset of GCs with planar curved axes and with circular (not necessarily constant) cross sections [93]. In [26], Gross considered GCs with planar curved axes or with circular cross sections, and presented an algorithm to recover the GCs using image contours and reflectance information.

1.6 Research Problems and Our Contribution

Although there have been a number of papers discussing the 3D reconstruction from single 2D line drawings [48], [37], [10], [70], [71], [69], all of them handle only line drawings of planar objects, and most of them consider simple objects without holes. In addition, they all requires the hidden lines in the line drawings to be shown.

Our work is mainly devoted to overcome these two constraints. Fig. 1.2 shows the results of our new scheme. To the best knowledge of us, our work is the first try to reconstruct from line drawings of curved objects or from line drawings without hidden lines.

1.6.1 A New Criteria

In Chap. 2, We proposes a new approach to 3D object reconstruction from single 2D line drawings. Our approach is based on the law of symmetry and the constraint of planarity. The former, coming from Gestalt psychology, reveals that human vision trends to interpret an object as symmetrical as possible. Inspired from it, we define a symmetry measure for a planar face in 3D space, which is equal to the ratio of the area to the perimeter squared of the face. The 3D object is reconstructed by maximizing the total symmetry of the object's surface, with the constraint of planarity. A number of examples are given to demonstrate that our approach outperforms the previous ones.

¹A planar curve is a curve lying on a plane in space.

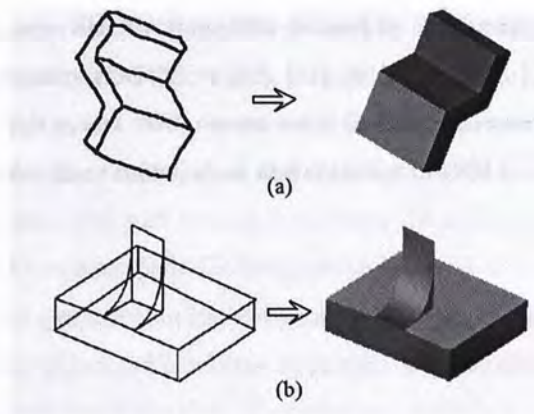


Figure 1.2: Our contributions. (a) Reconstruction from a line drawing without hidden lines. (b) Reconstruction from a line drawing as a curved objects.

1.6.2 Recover Objects from Line Drawings without Hidden Lines

Chap. 3 proposes an approach to the 3D reconstruction from line drawings without hidden lines. Compared with the line drawings with hidden lines visible, these line drawing are easier and more natural to draw. In addition, if we want to recover the complete 3D shape of an object in an image, we do not have the invisible edges to handle. Our approach mainly consists of two elements: inference of the topology of the hidden part of an object and reconstruction of the complete shape of the object. Fig. 1.2(a) shows an example of the the reconstructed results. And this work is also described in [11], [12].

1.6.3 Reconstruction of Curved Objects

Reconstruction of curved objects is a much harder problem. Three non-collinear points determine a plane, but a curved surface often has much more degrees of freedom. Therefore, the reconstruction of curved objects owns a higher underconstrained nature.

We propose an approach to the reconstruction from line drawings of objects with both planar and curved faces. We will show that a line drawing with hidden lines and silhouettes visible makes it possible to reconstruct its complete complex 3D shape. Fig. 1.2(b) shows an example of our reconstruction results. More details can be found in Chap. 4 and our paper [41].

1.6.4 Planar Limbs Assumption and the Derived Models

In our research, starting from the discussion of the conditions when SHGCs and tori (a kind of curved GCs) have planar limbs, we define and study a new class of GCs, with the help

of the surface model proposed by Degen for geometric modeling [21], [22]. We call them Degen generalized cylinders (DGCs), which include SHGCs, tori, quadrics, cyclides, and more other GCs into one model. We present some invariant properties of DGCs that reveal the relations among the planar limbs, axes, and contours of DGCs. This work is summarized in Chap. 5 and [13].

Chapter 2

A New Criteria for Reconstruction

2.1 Introduction

The human vision system can interpret a single 2D line drawing as a 3D wire-frame model without much difficulty. Several reconstruction approaches have been proposed to emulate this ability. All of them are based on minimizing the distribution of angles in the recovered object, except one that is based on minimizing the distribution of segment magnitudes. In this paper, we propose a new method for the same purpose based on the law of symmetry from Gestalt psychology. This law reveals that human beings interpret objects as symmetrical as possible. Inspired from it, we define a symmetry measure for a planar face and reconstruct a 3D object from a line drawing by maximizing the total symmetry of the object's surface with the constraint of planarity. We use an extensive set of experiments to demonstrate the significantly improved performance of the new algorithm over the existing ones.

The human vision system has the ability to interpret 2D line drawings as 3D objects without difficulty. It will be very helpful if a computer can also have this ability. One application is in fast and easy object design where tools are highly desirable that can convert a sketch into a 3D model directly. There have been a few approaches [48], [37], [10], [70] proposed for this purpose, in which the reconstruction is formulated as optimization problems based on different objective functions and a number of successful examples are provided.

Marill [48] presented his approach based on a very simple criterion: minimizing the standard deviation of the angles (SDA) in the reconstructed object, which is called the MSDA principle. Although some successful examples are given in his work, it is easy to find

cases where the MSDA approach fails, as pointed out by Leclerc and Fischler [37]. Leclerc and Fischler's approach considers not only the MSDA principle, but also the planarity constraint exhibited on the faces of the reconstructed object. They showed some examples in [37] where the MSDA fails but their algorithm works well. They also explained why it is necessary to enforce the planarity constraint for the reconstruction.

Motivated by the MSDA, Brown and Wang [10] proposed the principle of minimizing the standard deviation of the segment magnitudes (MSDSM) in the recovered object, as a counterpart of the MSDA. More recently, Shoji et al. [70] presented the criterion of minimizing the entropy of angle distribution, which is claimed to be more general than both the MSDA and the MSDSM.

Of the above four approaches, three are based on minimizing the distribution of angles in the reconstructed object, and one is based on minimizing the distribution of segment magnitudes.

In this chapter, we propose a new approach to 3D object reconstruction from single 2D line drawings. Here a line drawing is defined as the orthogonal projection of a wire-frame object in a generic viewpoint with all the edges and vertices shown. Moreover, only objects having planar faces are considered, as is the case in [48], [37], [10], [70]. Handling curved objects is much harder and is left for future study.

Our approach is based on the law of symmetry and the constraint of planarity. The former, coming from Gestalt psychology, reveals that human vision tends to interpret an object as symmetrical as possible. Inspired from it, we define a symmetry measure for a planar face in 3D space, which is equal to the ratio of the area to the perimeter squared of the face. The 3D object is reconstructed by maximizing the total symmetry of the object's surface, with the constraint of planarity. A number of examples are given to demonstrate that our approach outperforms the previous ones.

The rest of this chapter is organized as follows. In Section 2.2 we discuss human perception of a line drawing and define the symmetry measure for a planar face in 3D space. The scheme of the 3D reconstruction of the whole object from a line drawing is presented in Section 2.3. The large set of reconstruction results generated by our approach and the previous ones are given in Section 2.4. Finally, Section 3.7 concludes this paper.

2.2 Human Visual Perception and the Symmetry Measure

A phenomenon of human vision is that given a 2D line drawing, the perceived 3D shapes from it are limited although there exist infinite possible interpretations. As shown in Fig. 2.1, for the 2D drawing, a human usually accepts the middle 3D object and rejects the other two. We may say that the projection is in a more *generic* view of the middle object. To accord with the human vision's perception, the study on line drawing interpretation mostly uses this generic view assumption. However, this qualitative concept gives little help in the derivation of the 3D geometry of the object from a line drawing.

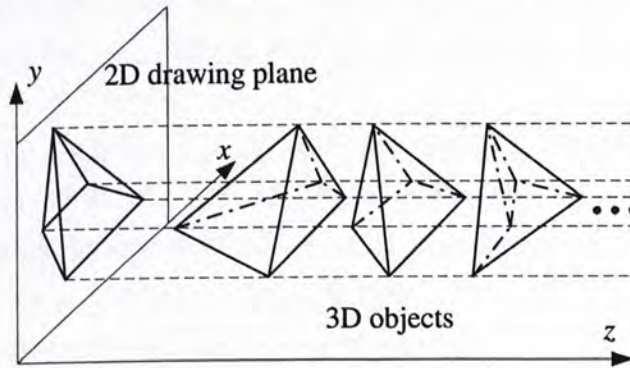


Figure 2.1: Infinite 3D objects that can project to the same line drawings.

Visual psychologists have attempted to discover the laws of perceptual decisions made by the human visual system for hundreds of years. Gestalt psychology, one of the most influential theories with a long history, asserts that human beings are innately driven to experience things as good a whole as possible. Here *good* can mean many things such as symmetry, regularity, simplicity, and order of an object [57], [34]. This nature leads us, when seeing, to strongly favoring certain shapes and configurations over others without high level identification. Gestalt laws of perceptual organization not only are popular in psychology, but also have applications in some fields of computer vision and pattern recognition [57], [97].

The law of symmetry is one of the most important Gestalt laws, which reveals that the human visual system is overwhelmed by symmetry and tends to interpret a figure in such a way as to produce an object that is as symmetrical as possible. Motivated by this law, we build up our reconstruction rule as follows: Given a 2D line drawing, reconstruct its 3D model such that it has as symmetrical faces as possible.

It is necessary to convert this subjective term *symmetry* into a computable measure. Let

us consider a symmetry measure S for a closed planar figure first. It is defined as:

$$S = \frac{A}{P^2} \quad (2.1)$$

where A and P are the area and perimeter of the figure, respectively.

It can be proved that $S \leq \frac{1}{4\pi}$ for any closed planar figure [4]. A circle is the most symmetrical planar object with $S = \frac{1}{4\pi}$. For a polygon with m vertices, its symmetrical measure $S \leq 4m \tan(\frac{\pi}{m})$ [4]. The maximum is achieved if and only if the polygon is the most symmetrical with m equal-length sides. These facts indicate that equation (3.4) is a rather reasonable measure of symmetry and in accordance with human perception.

S is also called a compactness measure. It is easy to be computed and popular in image processing, computer vision and pattern recognition to describe the feature of an object [8], [25], [16]. Although this measure has been used elsewhere, it is new for 3D object reconstruction from line drawings.

An object usually consists of more than one face. We treat the recovered object as the integration of all its 3D planar faces. Therefore, for an object with n faces, we define the whole symmetry measure of the object as

$$S_w = \sum_{i=1}^n \frac{A_i}{P_i^2} \quad (2.2)$$

where A_i and P_i , $1 \leq i \leq n$, are the area and perimeter of face i , respectively. We expect that given a line drawing, maximizing S_w would provide us with the most plausible recovered 3D object.

2.3 Reconstruction Based on Symmetry and Planarity

2.3.1 Finding Faces

To calculate S_w , we must know the topological information of the individual faces in a line drawing, i.e., which circuits of the drawing stand for the faces of the corresponding 3D object. Finding faces from a 2D line drawing is not an easy problem. In general, a drawing has far more circuits than faces, and the number of circuits increases exponentially with respect to the number of faces. Fortunately, two algorithms [71], [42] are available for our use. We adopt the one in [42] because it is much more efficient than the other.

2.3.2 Constraint of Planarity

When we observe a line drawing representing a 3D object, it is obvious that we can identify the circuits that are faces. This face information is very useful in helping our perception of

the shape of the object. Leclerc and Fischler [37] recognized this importance and added the constraint of planarity into Marill's MSDA scheme, obtaining more successful examples. In our approach, we also enforce this constraint to our symmetry-based reconstruction.

For a face with more than three vertices, it is not likely for all the vertices to be located exactly on a plane in 3D space, especially during the first steps of the reconstruction procedure. To compute the deviation from planarity for a face, Leclerc and Fischler used two different ways according to whether the face is convex or not [37]. However, we found that the commonly-used least square fitting technique is more robust than Leclerc and Fischler's method.

Suppose that face i has m vertices (x_{ij}, y_{ij}, z_{ij}) , $1 \leq j \leq m$. The plane that fits these vertices best can be expressed as:

$$a_i x + b_i y + c_i z - 1 = 0. \quad (2.3)$$

Thus, we evaluate DP_i , the deviation from planarity for face i , by the sum of all the distances of the m vertices from the plane:

$$DP_i = \frac{1}{\sqrt{a_i^2 + b_i^2 + c_i^2}} \sum_{j=1}^m |a_i x_{ij} + b_i y_{ij} + c_i z_{ij} - 1|. \quad (2.4)$$

The coefficients a_i, b_i, c_i of the plane can be obtained by the least square best-fit plane algorithm [62].

For an object with n faces, the total deviation from planarity DP is defined as:

$$DP = \sum_{i=1}^n DP_i. \quad (2.5)$$

2.3.3 Objective Function

Based on the above analysis, our objective of reconstruction is to maximize S_w while minimizing DP (imposing the constraint of planarity). In order to combine the two targets into one objective function, S_w is replaced with S'_w that takes the form

$$S'_w = \sum_{i=1}^n \frac{P_i^2}{A_i}. \quad (2.6)$$

S'_w has an inverse trend with S_w . Therefore, the objective function to be minimized is defined as

$$f(z_1, z_2, \dots, z_v; \lambda) = \lambda S'_w + (1 - \lambda) DP \quad (2.7)$$

where $0 \leq \lambda \leq 1$ is a weighting factor, v denotes the number of vertices in a line drawing, and z_1, z_2, \dots, z_v are the z -coordinates to be found. Minimizing f expresses our aim to construct a 3D object with faces as symmetrical as possible under the constraint of planarity.

2.3.4 Reconstruction Algorithm

With a fixed λ in (3.11), there are many optimization algorithms available to obtain the z -coordinates by minimizing f , such as the hill-climbing techniques and generic algorithms. Leclerc and Fischler [37] also provided a scheme, called the *continuation method*, for minimizing their objective function which is in a similar form to (3.11).

In the continuation method, λ is not fixed. Instead, it is a sequence of descent steps applied to f for decreasing values of λ . The sequence begins with the initial condition, $z_1 = z_2 = \dots = z_v = 0$, and some relative large $\lambda_0 \leq 1$. Then λ is reduced by a given value and the descent algorithm is applied again, starting from the solution obtained by the descent algorithm for the previous value of λ . This procedure is repeated until λ goes to a predefined small value. Note that this strategy ensures that the symmetry is favored at the beginning of the optimization, and then the constraint of planarity becomes more and more dominating.

After comparing the continuation method with some hill-climbing algorithms and genetic algorithms, we choose it to be our reconstruction algorithm since we have found that it can generate better results and is fast enough.

2.4 Experimental Results

In this section, we present a number of examples to illustrate and compare the 3D object reconstruction by our approach and the four methods in [48], [37], [10], [70].

For simplicity, Marill's method of *minimizing the standard deviation of angles* [48], Leclerc and Fischler's *planarity enforcing and minimizing the standard deviation of angles* [37], Brown and Wang's *minimizing the standard deviation of segment magnitudes* [10], and Shoji et al.'s *minimizing the entropy of angle distribution* [70] are termed as MSDA, PEMSDA, MSDSM, and MEAD, respectively.

The continuation method given in [37] is used to derive the z -coordinates in the optimization procedures of our approach and the PEMSDA. Marill's hill-climbing scheme [48] is used in the MSDA, MSDSM and MEAD, for which the continuation method is not suitable. The initial values of the z -coordinates of all line drawing are set to zero. All the algorithms are implemented in Visual C++. In our reconstruction tool, a recovered 3D object can be rotated so that we can view it in different directions and easily judge whether it is desirable or not.

The first set of examples contains all the line drawings in [48], [37], [10], [70] except the three shown in Fig. 2.2, where there are edges that can not be used to form a face.

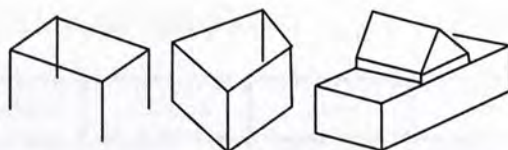


Figure 2.2: Three line drawings that are not included in our experiments.

Fig. 2.3 shows the reconstruction results by the five algorithms. Each 3D object is rotated by an angle from its original position for better examination. The faces of the objects are distinguished by different gray levels. It should be noted that when an algorithm fails to recover an object from a line drawing, a circuit that stands for a planar face may not be located (even nearly) on a plane, rendering a strange 3D shape. In this case, we still use a grey level to show this “face”; an obvious example is the result generated by the MEAD for line drawing (f) in Fig. 2.3. Here, when we say a result is successful, we mean that when the 3D object is viewed from different directions with our tool, the object does not look distorted from our perception. Otherwise, the result is considered failed, and it is displayed in a viewpoint that can clearly indicate the result is not correct. For the successful examples, it will be more convincing if we show each one in several different viewpoints, but the page limitation does not allow it.

There are total 23 line drawings in Fig. 2.3, where we use “√” or “×” to indicate whether a result is successful or not. From the figure, we can find that the three algorithms, MSDA, MEAD and MSDSM, do not work as well as the PEMSDA and our algorithm. The MSDSM produces most failed examples, and the MSDA is a little better than the MEAD. The PEMSDA fails for two examples, but our algorithm successes for all the line drawings.

Fig. 2.4 shows a new set of line drawings which are more complex as a whole than those in Fig. 2.3. It is easy to see that the MSDA, MEAD and MSDSM fail for all the line drawings. The PEMSDA generates two correct results but fails for all the others. On the other hand, our algorithm is successful for all the reconstructions.

We have also tested the stability of these five algorithms. We ran each algorithm on several drawn line drawings of every 3D object in different generic views. The results reveal that these algorithms are quite stable in the sense that if an algorithm fails (succeeds) to reconstruct the shape from one projection, it is very likely for it to be failed (successful) to reconstruct the shape from other projections of the object. All these algorithms are very fast in computation. On a 2 GHz Pentium IV PC, each algorithm takes less than 0.7 second to reconstruct each of the line drawings.

It is not difficult to understand why the MSDA, MEAD and MSDSM do not perform

Line Drawing	Our Algorithm	PEMSDA	MSDA	MEAD	MSDSM
(a)	✓	✓	✓	✓	✓
(b)	✓	✓	✓	✓	✓
(c)	✓	✓	✓	✓	✓
(d)	✓	✓	✓	✓	✓
(e)	✓	✓	✓	✓	✗
(f)	✓	✓	✗	✗	✗
(g)	✓	✓	✗	✗	✗
(h)	✓	✓	✗	✗	✗
(i)	✓	✓	✓	✗	✗
(j)	✓	✗	✗	✗	✗
(k)	✓	✓	✓	✗	✓

Figure 2.3: Reconstruction results for the line drawings in [48], [37], [10], [70].

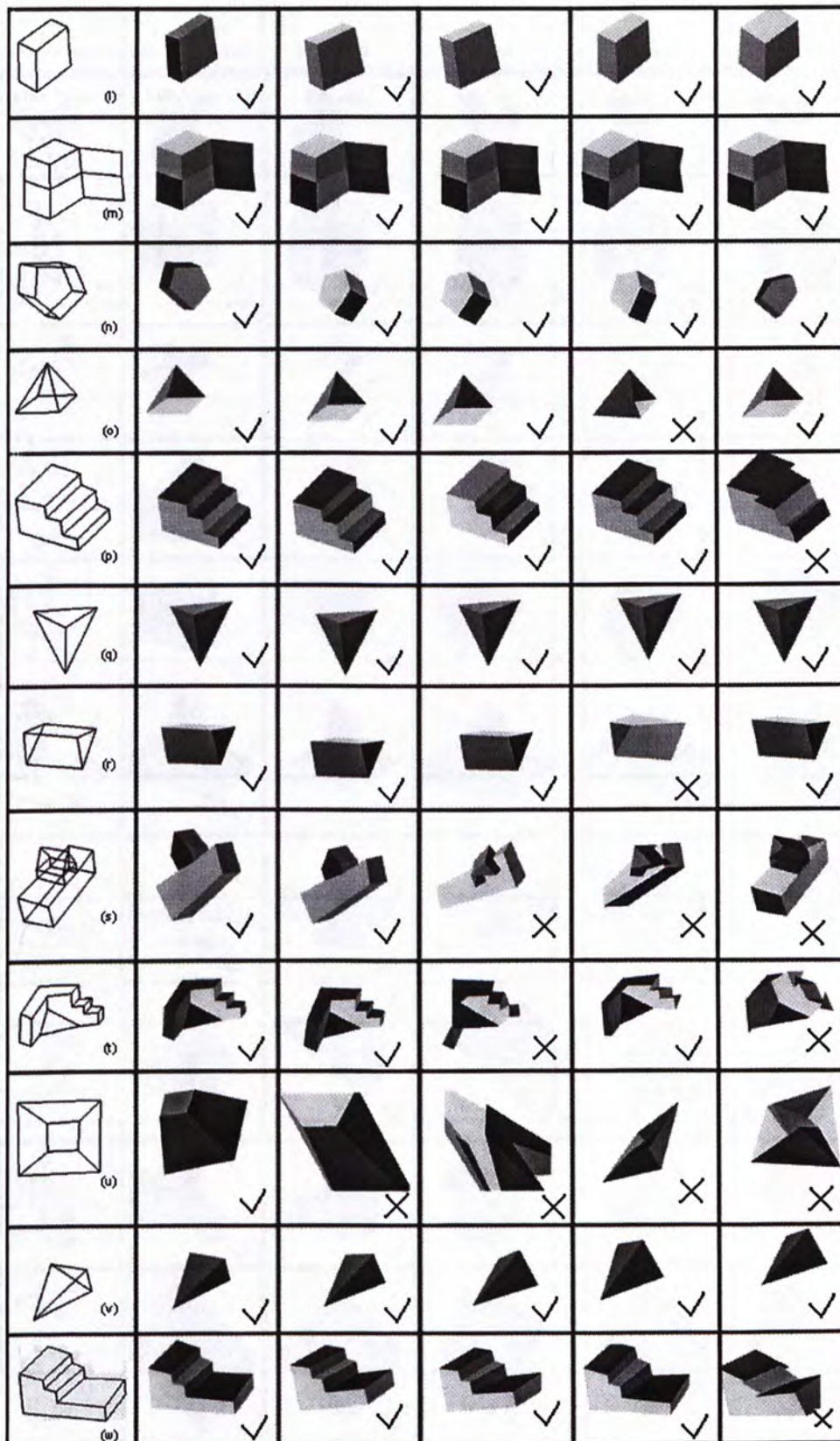


Figure 2.3: Continued.

Line Drawing	Our Algorithm	PEMSDA	MSDA	MEAD	MSDSM

Figure 2.4: Reconstruction results for a new set of line drawings.

well. The main reason is that they do not include the constraint of planarity, which is an important factor in human perception of a line drawing. The reason why our algorithm can work better than the PEMSDA, both of which have the planarity constraint, is not so obvious. After a careful examination of the PEMSDA scheme, we have found that there are two inherent shortcomings in it. The first can be explained with Fig. 2.5. For the two 3D objects each with three faces (one object is the extension of the other), they lead to exactly the same value of the PEMSDA objective function for every λ [37], although they are not the same objects clearly. The second shortcoming is that for a vertex where more than three edges meet at it (say, l edges), all the C_i^2 angles (each formed by two edges) will take part in the calculation of value of the objective function. However, not every such angle is an angle of a planar face. These extra angles might cause the failure of the PEMSDA.

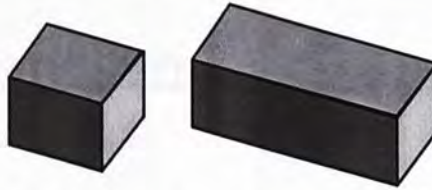


Figure 2.5: Two different 3D objects that give the same value of the PEMSDA objective function.

2.5 Summary

We have presented a novel approach to 3D reconstruction from single 2D line drawings. It is based on the law of symmetry from Gestalt psychology and the constraint of planarity. Both elements reflect how human beings perceive the 3D structure from a line drawing that represents the object. The objective function is quite simple, yet powerful to do the reconstruction work. A large set of examples has been shown to demonstrate that our algorithm outperforms the previous MSDA, MEAD, MSDSM, and PEMSDA algorithms.

The criteria proposed in this chapter will be used in the following chapters. It is with this powerful criteria that we are able to extend the reconstruction of limited objects to the future reconstruction of curved objects and reconstruction from incomplete line drawings.

Chapter 3

Line Drawings without Hidden Lines: Inference and Reconstruction

3.1 Introduction

The human vision system can interpret a single 2D line drawing as a 3D object without much difficulty even if the hidden lines of the object are invisible. Several reconstruction approaches have tried to emulate this ability, but they cannot recover the complete object if the hidden lines of the object are not shown.

This paper proposes an approach to the 3D reconstruction from line drawings without hidden lines. Compared with the line drawings with hidden lines visible, these line drawings are easier and more natural to draw. In addition, if we want to recover the complete 3D shape of an object in an image, we do not have the invisible edges to handle. Our approach mainly consists of two elements: inference of the topology of the hidden part of an object and reconstruction of the complete shape of the object. Fig. 3.1 shows the two steps with an example.

Given a line drawing without hidden lines, there are infinite possible structures being the hidden part of the object. Therefore, we have to impose reasonable constraints for the inference of the hidden structure such that the final recovered 3D object is in accordance with our perception from the line drawing. Besides, in this first work, we restrict our reconstruction to a class of polyhedra where each vertex is of degree 3. This class covers a large set of common objects.

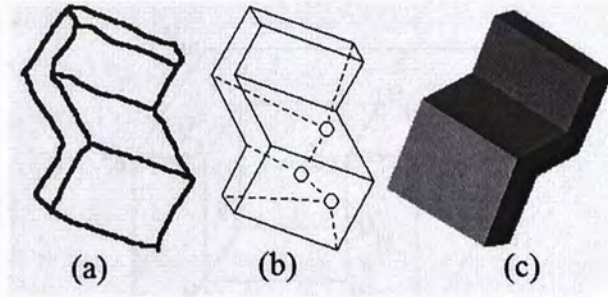


Figure 3.1: (a) A line drawing without hidden lines. (b) The inferred hidden topological structure. (c) Reconstructed complete 3D object.

3.2 Terminology

For easier understanding of the technical content of our approach, we first summarize the terms that will be used in the rest of the paper. Some of these terms are illustrated with Fig. 3.2.

- **Degree.** The degree $d(v)$ of a vertex v is the number of edges meeting at v in a line drawing.
- **Incomplete vertex.** An incomplete vertex v is a vertex of $d(v) = 2$.
- **Complete vertex.** A complete vertex v is a vertex of $d(v) = 3$.
- **Broken vertex.** A broken vertex v is a vertex of $d(v) = 1$.
- **Cycle.** A cycle is a closed trail in a line drawing where all its vertices except the end vertices are distinct.
- **Visible face.** A visible face is a face bounded by a cycle composed of all visible edges in a line drawing.
- **Rank.** The rank $R(e)$ of an edge e is the number of visible faces with edges passing through e .
- **Zero edge.** A zero edge e is an edge of $R(e) = 0$.
- **Boundary edge.** A boundary edge e is an edge of $R(e) = 1$.
- **Boundary cycle.** A boundary cycle is a cycle where all its edges are boundary edges.
- **Hidden cycle.** A hidden cycle is a cycle where all its edges and vertices are hidden.
- N_H . N_H denotes the number of hidden vertices.

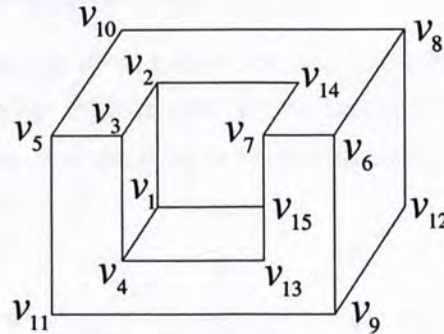


Figure 3.2: Illustration of some terms. Here v_{1-9} are complete vertices, v_{10-14} are incomplete vertices, and v_{15} is a broken vertex. Edge (v_{11}, v_9) is a boundary edge, and (v_1, v_{15}) is a zero edge. Cycle $(v_{10}, v_5, v_{11}, v_9, v_{12}, v_8, v_{10})$ is a boundary cycle.

3.3 Theoretical Inference of the Hidden Topological Structure

3.3.1 Assumptions

Although the 2D projection of a 3D objects has lost the depth information of the 3D vertices and edges, human beings still have the ability to perceive the complete 3D shape from the line drawing even if its hidden lines are invisible. Emulating this function obviously presents a very hard problem. In this first work to tackle the problem, we focus on a class of relatively simple solids and make the following assumptions.

Assumption 3.1 The 3D objects are polyhedra with all the vertices met by three edges and all the edges passed through by two faces.

Assumption 3.2 A line drawing is the parallel or near-parallel projection of the visible edges and vertices of a single polyhedron defined above in a generic view.

Assumption 3.3 Every hidden vertex is connected with at least one visible vertex.

Why do we need Assumption 3.3? If all the three invisible edges meeting at a hidden vertex are allowed to connect to other hidden vertices, there will be infinite possible structures being the hidden part. Assumption 3.3 restricts the infinite structures to limited simpler cases. Gestalt psychology, one of the most influential theories with a long history, asserts that human being are innately driven to perceive things as good a whole as possible. Here *good* can mean many things such as simplicity and symmetry [57]. Assumption 3.3 reflects the perception of simplicity by human beings.

3.3.2 Finding the Degrees and Ranks

Given a line drawing, there are three types of vertices as defined in Section 3.2. Incomplete vertices are easy to find, but to distinguish broken vertices from complete vertices is not obvious. The following theorem allows us to identify the broken vertices.

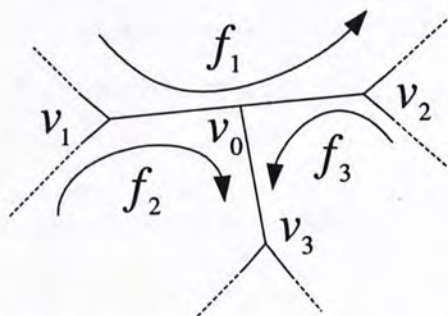


Figure 3.3: Part of a line drawing where v_1, v_0 and v_2 are collinear.

Theorem 3.1 If a vertex v_0 touches a straight line in a line drawing as shown in Fig. 3.3, then v_0 is a broken vertex.

Proof. Suppose, to the contrary, that v_0 is not a broken vertex. Then it is a complete vertex. With the assumption that every edge of the object is passed through by two faces, there are three planar faces passing through v_0 . Let them be $f_1 = (v_1, v_0, v_2, \dots, v_1)$, $f_2 = (v_1, v_0, v_3, \dots, v_1)$ and $f_3 = (v_2, v_0, v_3, \dots, v_2)$. since v_1, v_0, v_2 are collinear, the straight line (v_1, v_2) and the vertex v_3 that is not on this line defines a plane, implying that the two faces f_2 and f_3 are coplanar. Thus the edge (v_3, v_0) should not exist. Therefore, if edge (v_3, v_0) is a visible edge of the object, v_0 is a broken vertex. \square

Knowing all the types of vertices, we can find the degrees of the vertices directly. To find the ranks of the edges, we have to find the visible faces first. Face identification from a line drawing is not a trivial problem. Fortunately, there have been algorithms available for this purpose. We can use one of the algorithms published in [71] and [42] to find the visible faces. For example, the four visible faces found from the line drawing shown in Fig. 3.2 are $(v_1, v_2, v_3, v_4, v_1)$, $(v_6, v_9, v_{12}, v_8, v_6)$, $(v_2, v_3, v_5, v_{10}, v_8, v_6, v_7, v_{14}, v_2)$, and $(v_4, v_3, v_5, v_{11}, v_9, v_6, v_7, v_{13}, v_4)$. $R(v_2, v_3)$, $R(v_3, v_4)$, $R(v_3, v_5)$, $R(v_6, v_7)$, $R(v_6, v_8)$, and $R(v_6, v_9)$ are all equal to 2. $R(v_1, v_2)$, $R(v_1, v_4)$, $R(v_2, v_{14})$, $R(v_7, v_{14})$, $R(v_5, v_{10})$, $R(v_{10}, v_8)$, $R(v_8, v_{12})$, $R(v_{12}, v_9)$, $R(v_9, v_{11})$, $R(v_{11}, v_5)$, and $R(v_7, v_{13})$ are all equal to 1. Since v_{15} is a broken vertex, we cannot find the visible face passing through (v_1, v_{15}) at this stage. Thus, $R(v_1, v_{15}) = 0$. Note that when v_{15} is found to be a broken vertex, the line from v_7 to v_{13}

is one edge but not two.

3.3.3 Constraints for the Inference

The key to the inference of the invisible vertices and edges is to determine the number of hidden vertices N_H and the connections among invisible, incomplete, and broken vertices. The following theorems gives constraints useful for the inference.

Theorem 3.2 Let \mathcal{V}_I and \mathcal{V}_B be the set of incomplete and broken vertices of a given line drawing, respectively. Then we have

$$N_H \leq |\mathcal{V}_I| + |\mathcal{V}_B|, \quad (3.1)$$

where $|\cdot|$ denotes the number of elements in a set.

Proof. A complete vertex does not connect to any hidden vertex. An incomplete vertex connects to one hidden vertex. A broken vertex connect to one hidden vertex too. From Assumption 3, the largest value of N_H appears when all the hidden vertices connect to different visible vertices, which implies the inequality in (3.1). \square

The following Lemma [91] is used to prove Theorem 3.3.

Lemma 3.1 Let \mathcal{G} be any graph, \mathcal{E} be the set of edges and \mathcal{V} be the set of vertices in \mathcal{G} . It holds that

$$\sum_{v \in \mathcal{V}} d(v) = 2|\mathcal{E}|.$$

Theorem 3.3 Given a line drawing, if $|\mathcal{V}_I| + |\mathcal{V}_B|$ is even (odd), N_H of the line drawing must be even (odd).

Proof. Suppose there are N_H hidden vertices and l hidden edges in the object. If we construct a graph using all the hidden edges, hidden vertices, incomplete vertices, and broken vertices (without all visible edges), from Lemma 1 we have

$$\sum_{v \in \mathcal{V}_I} 1 + \sum_{v \in \mathcal{V}_B} 1 + \sum_{v \in \mathcal{V}_H} 3 = 2l,$$

where $\mathcal{V}_I, \mathcal{V}_B$ and \mathcal{V}_H are the sets of incomplete, broken, and hidden vertices, respectively, and $|\mathcal{V}_H| = N_H$. The above equation can be rewritten as $|\mathcal{V}_I| + |\mathcal{V}_B| + N_H = 2(l - N_H)$. Therefore, if $|\mathcal{V}_I| + |\mathcal{V}_B|$ is even (odd), N_H must be even (odd) too. \square

Theorem 3.4 Given a line drawing, we have

$$R(e_i) < d(v), \quad (3.2)$$

$$\sum_{i=1}^{d(v)} R(e_i) = 2F, \quad (3.3)$$

where $e_i, i = 1, 2, \dots, d(v)$, are the edges adjacent to vertex v , and F is the number of visible faces passing through v .

Proof. If v is a broken vertex, then e_i is a zero edge. From Section 3.3.2 we know $R(e_i) = 0$. Thus $R(e_i) < d(v) = 1$. If v is an incomplete vertex, then $R(e_i) \leq 1$ and $R(e_i) < d(v) = 2$. If v is a complete vertex, then $R(e_i) \leq 2$ and $R(e_i) < d(v) = 3$. Thus (3.2) holds.

Since a visible face passing through v passes through two of the $e_i, i = 1, 2, \dots, d(v)$, and $R(e_i)$ is the number of visible faces passing through e_i , we have (3.3) immediately. \square

Theorem 3.5 If a line drawing has no zero edges, an incomplete vertex must be on a boundary cycle.

Proof. We divide the proof into four parts.

(i) We first show that every incomplete vertex is connected with two boundary edges. For an incomplete vertex v_I with its two adjacent edges e_1 and e_2 , $d(v_I) = 2$. From (3.2) we have $R(e_1) < 2$ and $R(e_2) < 2$. Since there are no zero edges, we have $R(e_1) = R(e_2) = 1$. Thus the two edges are boundary edges.

(ii) We prove that each of the two vertices of a boundary edge e is connected with one and only one other boundary edge.

- When a vertex of the boundary edge is an incomplete vertex, the statement is true by (i).
- When a vertex of the boundary edge is a complete vertex, let its other two edges be e_1 and e_2 . From (3.3) we know that $R(e) + R(e_1) + R(e_2)$ is even. Since $R(e) = 1, R(e_1) \neq 0, R(e_2) \neq 0, R(e_1) \leq 2, R(e_2) \leq 2$, there are only two cases for $R(e_1)$ and $R(e_2)$, i.e., $R(e_1) = 1, R(e_2) = 2$ or $R(e_1) = 2, R(e_2) = 1$. Thus one and only one of the two edges e_1 and e_2 must be a boundary edge.

(iii) We verify that all the boundary edges must be on one of the boundary cycles in the line drawing. By the statement in (ii), we can always walk from a boundary edge to another boundary edge (See Fig. 3.4). Since the line drawing has finite vertices, we must meet one of the vertices on the path we have passed through. However, the case in Fig. 3.4(b) cannot happen because the vertex v_i are met by three boundary edges, which contradicts

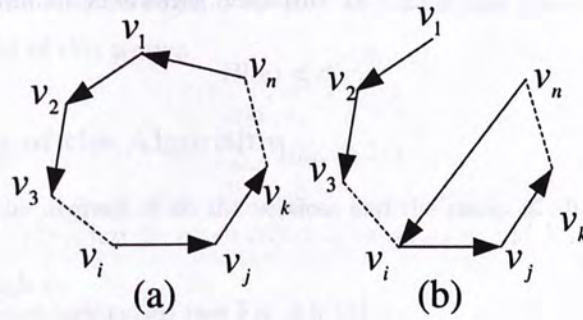


Figure 3.4: Walking along boundary edges. (a) A boundary cycle. (b) An impossible path.

the statement in (ii). Thus we can only have the case shown in Fig. 3.4(a), which forms a boundary cycle.

(iv) Since all the incomplete vertices are on boundary edges, they must be on a boundary cycle by the statement given in (iii). \square

When a line drawing has zero edges, we remove these edges and thus reduce the line drawing to a new one without zero edges. Then Theorem 3.5 applies to this new line drawing. Fig. 3.5 shows a line drawing with $R(v_1, v_2) = 0$. After removing the edge (v_1, v_2) , the reduced line drawing is given in Fig. 3.5 (b), where the bold edges illustrate two boundary cycles. We can also see that all the incomplete vertices are on the cycles.

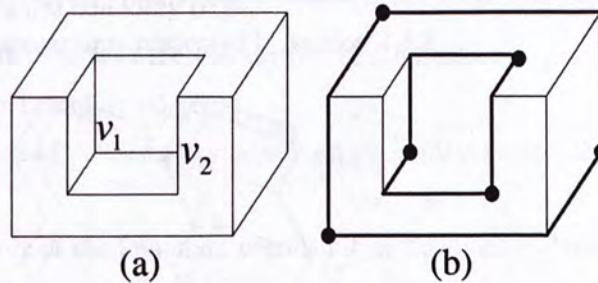


Figure 3.5: (a) A line drawing with a zero edge. (b) Two boundary cycles (bold) with incomplete vertices (filled-in circles) on them.

3.4 An Algorithm to Recover the Hidden Topological Structure

In this section, an algorithm based on the constraints and properties stated in the previous theorems is designed to infer the hidden vertices and edges in a line drawing. We consider

the line drawing without zero edges first. How to handle line drawings with zero edges is discussed at the end of this section.

3.4.1 Outline of the Algorithm

Step 1. Finding the degrees of all the vertices and the ranks of all the edges from a line drawing.

Step 2. Finding boundary cycles (see Fig. 3.6(b)).

Step 3. Constructing an initial hidden topological structure (see Fig. 3.6(c)).

Step 4. Reducing the initial hidden structure to one according to human visual perception of the 3D object (see Fig. 3.6(d)).

Step 1 has been described in Section 3.3.2. Step 2 can be done according to Theorem 3.5. We discuss Steps 3 and 4 in the following two sections.

3.4.2 Constructing the Initial Hidden Structure

Theorem 2 lets us know that the largest $N_H = |\mathcal{V}_I|$ since we consider the line drawings without zero edges (thus $\mathcal{V}_B = \emptyset$). We set $|\mathcal{V}_I|$ hidden vertices and connect each incomplete vertex to a different hidden vertex. Two hidden vertices are connected if their corresponding incomplete vertices are closest on the boundary cycle. One example is given in Fig.3.6(c), where the cycle $(v_1, v_2, v_3, v_4, v_5, v_1)$ is a hidden cycle. Note that the result of this initialization meets the constraints presented in Section 3.3.3.

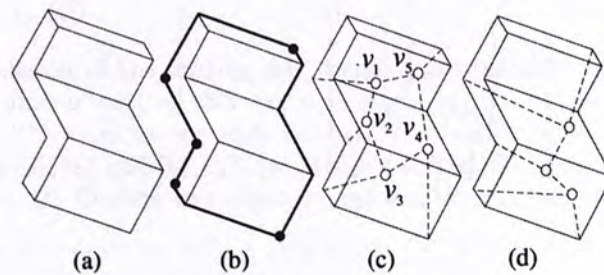


Figure 3.6: An example of inferring the hidden topology from line drawing. (a) A line drawing. (b) Finding the boundary edges (the bold cycle) and the incomplete vertices (filled-in circles). (c) Initial hidden vertices (open circles). (d) Reduction of the initial hidden vertices.

3.4.3 Reducing Initial Hidden Structure

Beginning with the initial hidden structure, we design a procedure to search for other possible hidden structures with fewer hidden vertices. The procedure uses a strategy of cutting-and-merging of edges and vertices. *Cutting* one edge on a hidden cycle removes this edge from the cycle while keeping the two vertices of the edge. After the cutting, the two hidden vertices of the edge are met by only two hidden edges (see Fig. 3.7(b)). To maintain that every vertex is met by three edges, we *merge* the two vertices to their adjacent hidden vertices (see Fig. 3.7(c)). Since each cutting-and-merging reduces two of the hidden vertices, the resulting number of hidden vertices is even (odd) if the initial N_H is even (odd), which satisfies the constraint by Theorem 3.3.

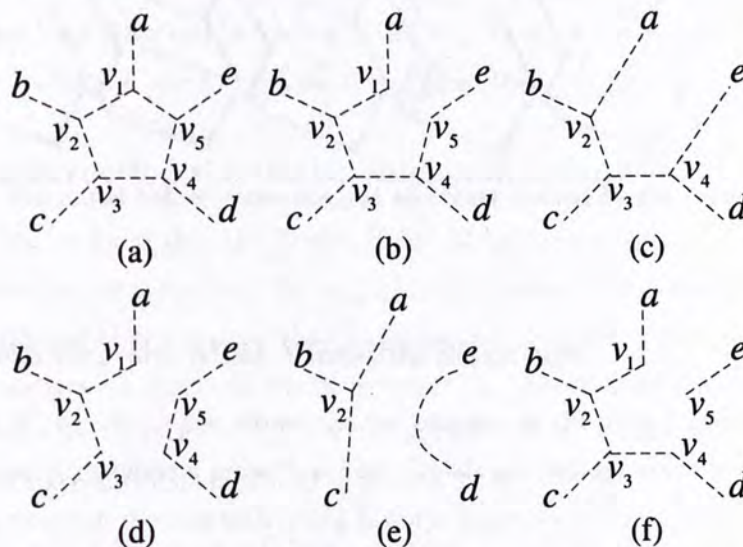


Figure 3.7: Illustration of the cutting and merging procedure where $a-e$ are incomplete vertices, v_1-5 are hidden vertices, and $(v_1, v_2, v_3, v_4, v_5, v_1)$ is a hidden cycle. (a) The initial hidden structure. (b) Cutting one edge (v_1, v_5) , (c) Merging v_1 to v_2 and v_5 to v_4 . (d) Cutting two edges (v_1, v_5) and (v_3, v_4) . (e) Merging v_1 and v_3 to v_2 ; v_5 and v_4 disappearing after being merged. (f) Cutting two edges (v_1, v_5) and (v_4, v_5) , where v_5 has no place to be merged.

At first, the cutting is applied to one hidden edge on the hidden cycle each time, resulting in different hidden structures. The cutting is also used to cut two or more such edges each time. It should be emphasized that not every cutting of two edges or more each time is valid. Fig. 3.7(d) and (f) show two examples of cutting two edges each time. The former is valid but the latter is not.

The cutting-and-merging procedure is always applied to the initial hidden structure, cutting one edge or multiple edges each time. The maximal edges that can be cut each time

is the largest integer $\leq N_H/2$ because removing one edge reduces two hidden vertices. All the hidden structures obtained from the procedure plus the initial structure are kept for the selection of the most plausible one. Fig. 3.8 shows one example.

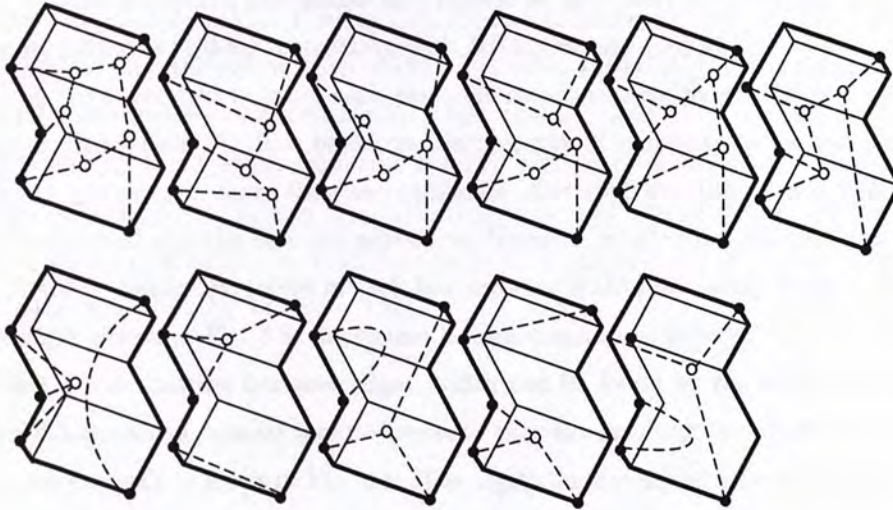


Figure 3.8: The initial hidden structure and all others derived by the cutting-and-merging procedure.

3.4.4 Selecting the Most Plausible Structure

Given a set of possible hidden structures, the selection of the most plausible one is based on some visual psychological properties from Gestalt psychology, which is one of the most influential perception theories with a long history. It asserts that human beings are innately driven to experience things as good a whole as possible. Here *good* can mean many things such as symmetry, simplicity, regularity, and order of an object [57], [34]. This nature leads us, when seeing, to strongly favoring certain shapes and configurations over others without high level identification.

The law of symmetry is one of the most important Gestalt laws, which reveals that the human visual system is overwhelmed by symmetry and tends to interpret a figure in such a way as to produce an object that is as symmetrical as possible. When this law is applied to the inference of the hidden structure of a line drawing, each hidden face should correspond to a similar visible face. At this stage, since we discuss topological structures only, we say that two faces are similar if they have the same number of edges. These two similar faces are also called topologically same. Motivated by this law, the rule of selecting the most plausible hidden structure is: given a set of hidden structures from a line drawing, select the one having as many hidden faces similar to the visible faces as possible (Rule 1).

We define a non-symmetry measure NSM for the selection. Suppose there are n hidden structure, at first, we set $NSM_i = 0, 1 \leq i \leq n$. For every hidden face in the i th structure, we check if there is a corresponding topologically same visible face. If no, increase NSM_i by 1. For each structure, one visible face cannot be used more than once in the checking. The most plausible is the j th structure with $NSM_j = \min_{1 \leq i \leq n} \{NSM_i\}$.

When there are two or more such plausible structures, we choose the one with fewest hidden vertices (Rule 2). It is based on the property of simplicity in Gestalt psychology. If finally there are still more than one candidate after applying Rules 1 and 2, all of them are reconstructed and the user can select one. However, in all of our experiments, only one most plausible hidden structure in each line drawing is obtained using Rules 1 and 2. For the example shown in Fig. 3.8, the second hidden structure is selected.

When a line drawing has zero edges, which can be found by the scheme presented in Section 3.3.2, we can remove these edges such that the resulting line drawing has no zero edges. An example is given in Fig. 3.5. The algorithm developed above can be applied to this new line drawing. In the experimental section, we can see that the hidden structures of many line drawings with zero edges are recovered successfully.

3.5 Reconstruction of 3D Objects

In what follows, we call a line drawing with its recovered hidden structure a complete line drawing. After recovering a complete line drawing, the next important work is to reconstruct its 3D shape. Since we already know the face topology of the complete line drawing, we only need to derive the 3D coordinates of all the visible and hidden vertices. We consider a line drawing is a parallel (or near parallel) projection of a 3D object. The x - and y -coordinates of each visible vertex is thus already known, and only the z -coordinate (depth) has to be derived. However, all the x -, y - and z -coordinates have to be found for hidden vertices. This reconstruction problem is more difficult than those in the previous work [40], [69], [48], [37], where all the vertices and edges of a line drawing are given in the 2D projection. In addition, the previous methods cannot recover the complete 3D object from a line drawing without hidden vertices and edges.

In the following, we propose an optimization-based approach to tackling the reconstruction problem. It inflates a flat line drawing into a 3D object by assigning depths to all the vertices and x - and y -coordinates to all the hidden vertices. The recovered object should be in accordance with our visual perception. We develop an objective function first and then give an algorithm for the reconstruction.

Based on the spirit of the law of symmetry from Gestalt psychology, we consider a symmetry measure S for a closed planar figure first. It is defined as:

$$S = \frac{A}{P^2} \quad (3.4)$$

where A and P are the area and perimeter of the figure, respectively.

It holds that $S \leq \frac{1}{4\pi}$ for any closed planar figure [4]. A circle is the most symmetrical planar figure with $S = \frac{1}{4\pi}$. For a polygon with m vertices, its symmetrical measure $S \leq 4m \tan(\frac{\pi}{m})$ [4]. The maximum is achieved if and only if the polygon is the most symmetrical with m equal-length sides. These facts indicate that (3.4) is a rather reasonable measure of symmetry.

An object consists of more than one face. We treat the recovered object as the integration of all its 3D planar faces. For an object with n faces, we define the whole symmetry measure of the object as

$$S_w = \sum_{i=1}^n \frac{A_i}{P_i^2} \quad (3.5)$$

where A_i and P_i , $1 \leq i \leq n$, are the area and perimeter of face i , respectively. We expect that given a line drawing, maximizing S_w combined with other two criteria would provide us with the most plausible recovered 3D object. This measure S_w plays the main role in the reconstruction, with the other two involved to get better results.

Marill [48] presented his approach to 3D reconstruction based on a simple criterion: minimizing the standard deviation of all the angles (SDA) in the reconstructed object. SDA can be calculated by:

$$SDA = \text{Var}_{i,j}(\cos^{-1}(\mathbf{u}_i \cdot \mathbf{u}_j)) \quad (3.6)$$

where \mathbf{u}_i and \mathbf{u}_j are the unit vectors of two lines meeting at a vertex of the 3D object. This criterion can be regarded as another form of symmetry constraint and is also incorporated into the objective function.

When we observe a line drawing representing a 3D polyhedron, we can clearly identify the cycles representing faces. This face information is very useful in helping our perception of the shape of the object [37]. We also enforce this planarity constraint in the object function.

For a face with more than three vertices, it is not likely for all the vertices to be located exactly on a plane in 3D space, especially during the first steps of the reconstruction. To compute the deviation from planarity for some vertices, the least square fitting technique is employed.

Suppose that face i has m vertices (x_{ij}, y_{ij}, z_{ij}) , $1 \leq j \leq m$. The plane that fits these

vertices best can be expressed as:

$$a_i x + b_i y + c_i z - 1 = 0. \quad (3.7)$$

Thus, we evaluate the deviation from planarity DP_i for face i , by the sum of all the distances of the m vertices from the plane:

$$DP_i = \frac{1}{\sqrt{a_i^2 + b_i^2 + c_i^2}} \sum_{j=1}^m |a_i x_{ij} + b_i y_{ij} + c_i z_{ij} - 1|. \quad (3.8)$$

The coefficients a_i, b_i, c_i of the plane can be obtained by the least square best-fit plane algorithm [62].

For an object with n faces, the total deviation from planarity DP is defined as:

$$DP = \sum_{i=1}^n DP_i. \quad (3.9)$$

Based on the above analysis, our objective of reconstruction is to maximize S_w while minimizing DP and SDA . In order to combine the three targets into one objective function, S_w is replaced with S'_w that takes the form

$$S'_w = \sum_{i=1}^n \frac{P_i^2}{A_i}. \quad (3.10)$$

Finally, the objective function to be minimized is defined as

$$\begin{aligned} f(z_1, z_2, \dots, z_v, x_{h1}, y_{h1}, x_{h2}, y_{h2}, \dots, x_{hu}, y_{hu}) \\ = \lambda_1 S'_w + \lambda_2 SDA + \lambda_3 DP \end{aligned} \quad (3.11)$$

where λ_1, λ_2 and λ_3 are weighting factors, z_1, z_2, \dots, z_v are the depths of all the v visible and hidden vertices, and $(x_{h1}, y_{h1}), (x_{h2}, y_{h2}), \dots, (x_{hu}, y_{hu})$ are the x - and y -coordinates of all the u hidden vertices. Minimizing f expresses our aim to construct a 3D object as symmetrical as possible under the constraint of planarity.

Although there are many optimization methods available, no one can guarantee to find the optimal solution to this multi-dimensional non-linear optimization problem. Genetic algorithms (GAs) have become a popular approach due to its good performance in finding optimal solutions in hard problems [47]. In our work, we also design a GA to minimize f in order to obtain the values of the variables (coordinates) in (3.11). Here the implementation of the GA is straightforward. Each of the variable is encoded with a length of two bytes (16 bits), and all the variables are combined to form a long string. The fitness function of the GA is simply defined as $f_{GA} = 1/(f+1)$, where $(f+1)$ guarantees that the denominator does not vanish. The larger is f_{GA} (i.e., the smaller is f), the fitter is the solution represented by the string. The principle how GAs work can be found in [47].

3.6 Experimental Results

In this section, we present a number of examples to illustrate the proposed approach to the inference and reconstruction of complete objects from line drawings without hidden edges and vertices. The weighting factors λ_1 , λ_2 , and λ_3 are chosen to be 0.65, 0.3, 0.05, respectively. These values are obtained by a few experiments first and then fixed in the reconstruction of all the objects. These parameters are not sensitive in the sense that if they are changed by 10%, the reconstruction results are quite similar. Here λ_3 is set to a value much less than λ_1 and λ_2 because DP is usually much larger than S'_w and SDA in (3.11). The four parameters of the GA, population size, maximum generation, mutation rate, and crossover rate, are set to 100, 30, 0.1 and 0.8, respectively. These parameters are not sensitive either.

Fig. 3.9 shows a set of line drawings and their reconstruction results. The original line drawings are given in the first column. The results of the inference of the hidden structures are illustrated in the second column. The third and fourth columns show the recovered 3D shapes in two views each. From Fig. 3.9, we can see that the reconstructed objects quite accord with our visual perception. The algorithm implemented in C runs fast enough. It takes less than 1 second to do the inference and reconstruction for every line drawing.

Although the objects shown in Fig. 3.9 are not very complex, our work with these encouraging results is the first step towards the research on handling a wider range of objects.

3.7 Summary

We have proposed a novel approach to 3D reconstruction from single 2D line drawings without hidden lines. We first infer the hidden structure of a line drawing with the help of the constraints given in several theorems. Then we present an optimization-based method to recover the 3D shape from the complete line drawing. The objective function is developed based on perceptual symmetry and planarity. A genetic algorithm is designed to carry out the optimization. A number of encouraging results have been obtained, which demonstrate the success of the proposed approach. Our future work will focus on handling more complex polyhedra and curved objects.

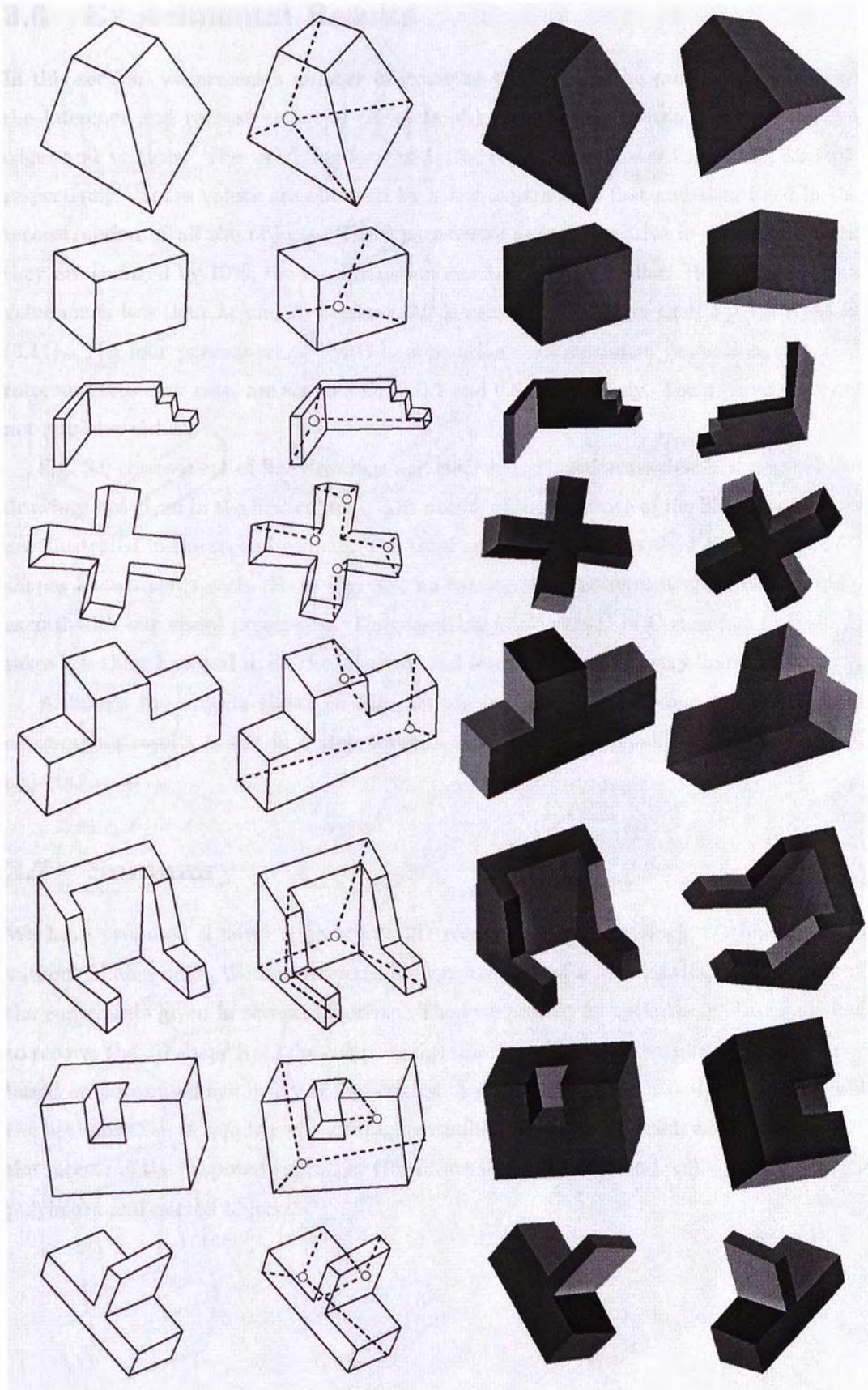


Figure 3.9: Results of the inference and reconstruction.

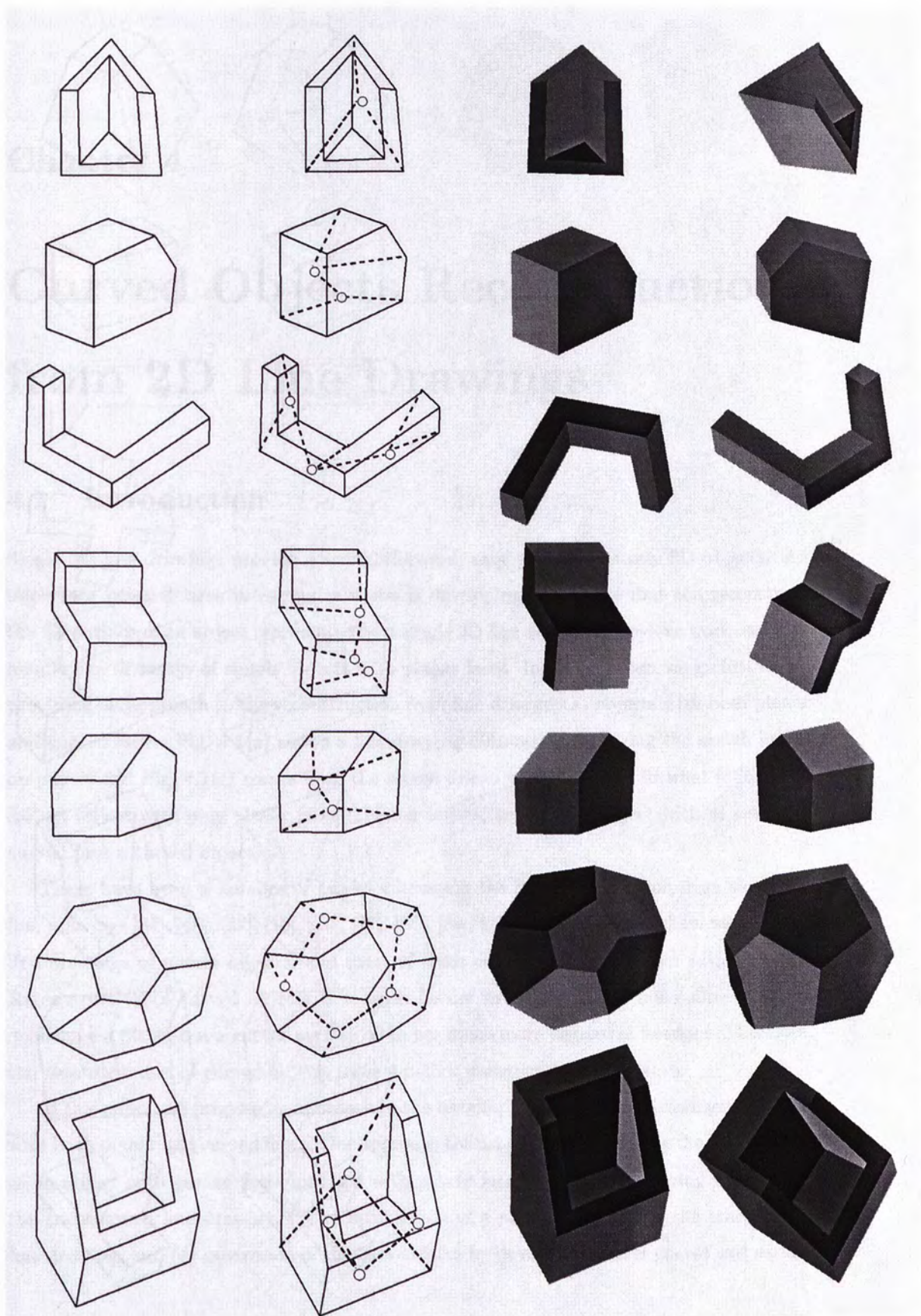


Figure 3.9: Continued: Results of the inference and reconstruction.

Chapter 4

Curved Objects Reconstruction from 2D Line Drawings

4.1 Introduction

Single 2D line drawings provide a straightforward, easy way to illustrate 3D objects. An important research area in computer vision is developing algorithms that can reconstruct the 3D surface of an object represented by a single 2D line drawing. Previous work can only handle line drawings of simple objects with planar faces. In this chapter, we go further by providing an approach to the reconstruction from line drawings of objects with both planar and curved faces. Fig. 4.1(a) shows a line drawing obtained by scanning the sketch image on paper, and Fig. 4.1(c) comes from the screen drawn with a mouse. In what follows, we call an object with only planar faces a planar object, and call an object with at least one curved face a curved object.

There have been a number of papers discussing the 3D reconstruction from single 2D line drawings [81], [48], [37], [10], [70], [77], [64], [89], [71]. However, all of them handle only line drawings of planar objects, and most of them consider simple objects without holes. Reconstruction of curved objects is a much harder problem. Three non-collinear points determine a plane, but a curved surface often has much more degrees of freedom. Therefore, the reconstruction of curved objects owns a higher underconstrained nature.

In this paper, we propose an approach to the reconstruction from line drawings of objects with both planar and curved faces. Our approach consist of (1) transforming the line drawing of an object with curved faces into one with planar faces only, (2) identifying the faces of the transformed line drawing, (3) reconstruction of a planar object from the transformed line drawing, and (4) generation of the curved faces by developing Bezier curves and surface

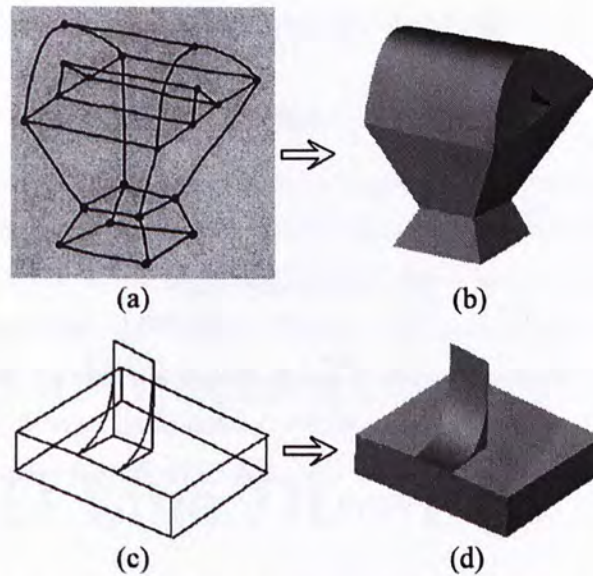


Figure 4.1: Two line drawings and their reconstructed shapes. Line drawings may be inputted from sketches, images, digitizer tablets, and the screen.

patches. A number of results are provided to demonstrate the ability of the approach to performing the reconstruction. Our algorithm can also be used as an interface to recover 3D models from images.

4.2 Related Work

Related work on the interpretation of line drawings can be classified into two groups: (1) line labeling, and (2) face identification and 3D reconstruction from single line drawings with hidden lines visible. Line labeling focuses on finding a set of consistent labels from a line drawing without hidden lines and does not explicitly give the 3D shape represented by the line drawing. Our work belongs to the second group.

4.2.1 Face Identification

Face identification from a line drawing is a necessary step. An object consists of faces. If the face configuration of an object is known before the reconstruction of its 3D geometry, the complexity of the reconstruction will be reduced significantly.

In general, there are many cycles in a line drawing and only a small subset of them represents its faces, and the number of cycles grows exponentially with the number of edges. Thus finding the faces from a line drawing is not a trivial problem. Much effort has been made in this area [71], [42], [69], [3], [2], [37], [43]. Among these techniques, the algorithm

presented in [42] is most suitable and adopted for face identification in our work.

4.2.2 3D Reconstruction of planar objects

Marill [48] presented his method based on a simple criterion: minimizing the standard deviation of the angles in the reconstructed object, which is called the MSDA principle. Motivated by the MSDA, Brown and Wang [10] proposed to minimize the standard deviation of the segment magnitudes (MSDSM) in the recovered planar object. More recently, Shoji et al. [70] presented the criterion of minimizing the entropy of angle distribution (MEAD), and claimed that it is more general than both the MSDA and the MSDSM. The MSDA is also used in other related work [69], [37], [71]. All these methods can handle only planar objects.

4.3 Reconstruction of Curved Objects

This section discusses our work in the reconstruction of curved objects. At first, we give the scheme to transform the line drawing of a curved object into the line drawing of a planar object. Then, we present our new method of reconstructing planar objects. Finally we propose the method for the generation of curved faces with Bezier curves and surface patches based on the reconstructed planar objects.

4.3.1 Transformation of Line Drawings

As mentioned in Section 4.1, the reconstruction of curved objects is a much harder problem than that of planar objects. To reduce the complexity in the problem and to remove ambiguity in some line drawings, when a line drawing is drawn, extra lines may be added on curved faces so that they can be approximated by planar patches. Then the line drawing is transformed into one representing a planar object by straightening all the curves in it.

Let us look at several examples in Fig. 4.2. If the line drawing in Fig. 4.2(a) is transformed into Fig. 4.2(b) by replacing the curves with straight lines, the face configurations of them are the same. For the cylinder shown in Fig. 4.2(d), if the four curved edges are straightened, the transformed line drawing in Fig. 4.2(e) does not represent a solid. However, if two lines bf and dh are added when drawing the object, as shown in Fig. 4.2(f), the new representation of the cylinder has a corresponding line drawing of a polyhedron (Fig. 4.2(g)) with the same face topology. Another advantage of adding the new lines is eliminating the ambiguity in face identification when the algorithm in [42] is used. There

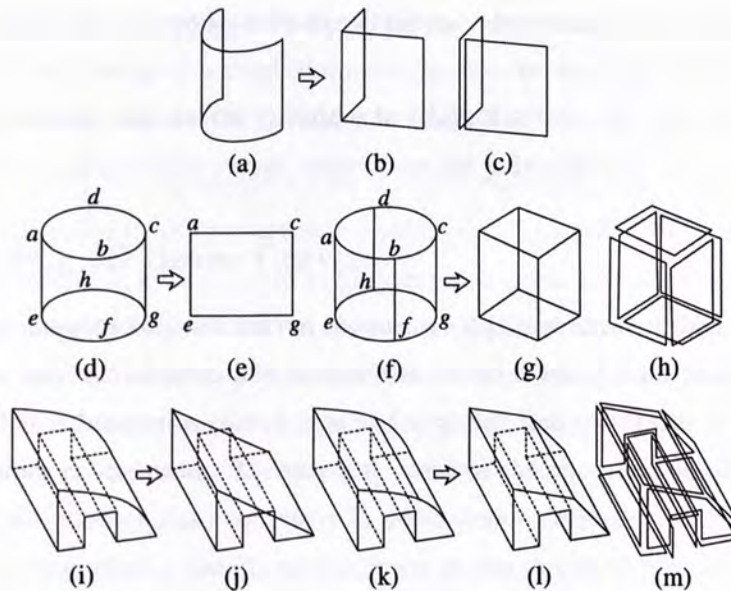


Figure 4.2: Examples of the transformation of line drawings, where hidden lines are displayed in dashed for easier observation.

are two solutions of face identification for Fig. 4.2(d): one set of faces is $\{(a, d, c, b, a), (e, h, g, f, e), (a, b, c, g, f, e, a), (a, d, c, g, h, e, a)\}$, another set is $\{(a, d, c, b, a), (e, h, g, f, e), (a, b, c, g, h, e, a), (a, d, c, g, f, e, a)\}$. For Fig. 4.2(f) or (g), there is only one set of six faces as the solution.

Replacing the two curves in Fig. 4.2(i) by two straight lines, we obtain the line drawing in Fig. 4.2(j). However, this is not a valid representation of a polyhedron, causing both the face identification and the reconstruction to fail. The problem can be overcome by adding one line on the curved faces as shown in Fig. 4.2(k), which has a valid polyhedron approximation (Fig. 4.2(l)).

In this work, we do not define where and how many extra lines must be added to the curved faces of a line drawing; the user has the choice. The basic rule is that the transformed line drawing should be a valid planar object representation of the curved object with the same face topology, and each curve has one vertex or more between its two endpoints. The last condition is for determining the curve in the 3D space (see Section 4.3.2 for why). Adding more lines to a line drawing imposes a little more work on the user (note that the step of straightening the curves is not manual), but doing this can reduce the complexity of the reconstruction, and often provide better visual perception of a line drawing.

Given the line drawing of a curved object, we transform it into a line drawing of a planar object, and then employ the face identification algorithm in [42] to find the face topology. Figs. 4.2(c), (h) and (m) give the found faces from Figs. 4.2(b), (g) and (l). Next we discuss

how to reconstruct the 3D shapes from line drawings representing planar objects.

Given the line drawing of a polyhedron, we can use the algorithm proposed in [43] to find the face topology, and use the technique in Chap. 2 to build the 3D polyhedron. Next we discuss how to generate the curved object from the polyhedron.

4.3.2 Finding 3D Bezier Curves

By observing numerous common curved objects, we find that most of them are composed of both planar and curved faces, and most of the curved edges of these objects are formed by the intersection between a curved face and a planar face. This can be seen from all the objects shown in this paper. Therefore, we focus on the reconstruction of such objects, and make the assumption that every curve in a line drawing is piecewise planar. With this assumption, we first recover the 3D curves based on the recovered planar object and the original 2D line drawing.

Let a 3D curve $\mathbf{C}(t)$ be described in the parameterized form

$$\mathbf{C}(t) = (x(t), y(t), z(t)). \quad (4.1)$$

Under the parallel projection, the 2D image of $\mathbf{C}(t)$ is simply

$$\bar{\mathbf{C}}(t) = (x(t), y(t)). \quad (4.2)$$

Now we need to recover $z(t)$ from $\bar{\mathbf{C}}(t)$. In general, given $\bar{\mathbf{C}}(t)$ and several specific 3D points on $\mathbf{C}(t)$, it is impossible to determine other 3D points on it. As stated in Proposition 5.1, however, this can be achieved in our case where the 3D curve is planar and each curve has one vertex or more between its two endpoints (see Section 4.3.1).

Proposition 4.1 If a 3D curve $\mathbf{C}(t)$ is planar, then its parallel projection $\bar{\mathbf{C}}(t)$ and its three non-collinear points in 3D space define it uniquely.

Proof. Suppose the equation of the plane $\mathbf{C}(t)$ lies on is $z = ax + by + c$. Then $z(t) = ax(t) + by(t) + c$. The three parameters a , b , and c can be obtained uniquely by solving the linear system

$$ax_i + by_i + c = z_i, \quad i = 1, 2, 3, \quad (4.3)$$

with the three available non-collinear 3D points (x_i, y_i, z_i) , $i = 1, 2, 3$. \square

The manipulation of the curve $\mathbf{C}(t)$ is not convenient because we do not have explicit expressions for $x(t)$ and $y(t)$. We use Bezier curves to represent general curves in this paper,

which are convenient to be drawn with a few control points [5]. A Bezier curve of degree n is specified by a sequence of $n + 1$ control points \mathbf{P}_i , $0 \leq i \leq n$. Its equation is

$$\mathbf{C}(t) = \sum_{i=0}^n \mathbf{P}_i B_{i,n}(t), \quad (4.4)$$

where $B_{i,n}(t)$ is the Bernstein polynomial function

$$B_{i,n}(t) = \binom{n}{i} t^i (1-t)^{n-i}. \quad (4.5)$$

From (4.4), we can see that the $n + 1$ control points \mathbf{P}_i define the curve completely. In our reconstruction tool, the user is able to move the control points freely to change the shape of a curve. If a line drawing is obtained by scanning a sketch image, we can find the control points by fitting a Bezier curve to an inputted curve using the technique in [58]. When the control points are collinear, the Bezier curve degenerates to a straight line. Thus, it can be used to represent straight lines too.

Let the control points of a curve $\bar{\mathbf{C}}(t)$ in the 2D projection plane be $\bar{\mathbf{P}}_i = (x_i, y_i)$, meaning that

$$\bar{\mathbf{C}}(t) = \sum_{i=0}^n \bar{\mathbf{P}}_i B_{i,n}(t).$$

Then we can find its corresponding Bezier curve $\mathbf{C}(t)$ in 3D space:

$$\mathbf{C}(t) = \sum_{i=0}^n \mathbf{P}_i B_{i,n}(t)$$

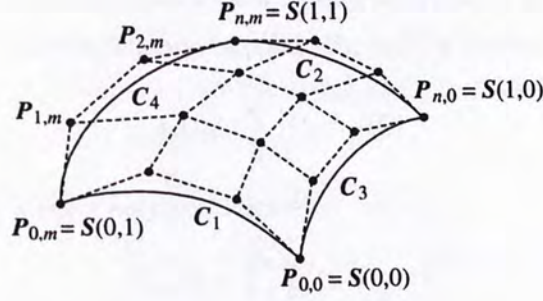
by deriving $\mathbf{P}_i = (x_i, y_i, z_i)$ with $z_i = ax_i + by_i + c$, where a, b, c are obtained from the solution of (4.3).

4.3.3 Bezier Surface Patches and Boundaries

Bezier surfaces are also popular in surface representation. A Bezier surface patch $\mathbf{S}(u, v)$ is defined by a $(n + 1) \times (m + 1)$ array of control points $\mathbf{P}_{i,j}$, as shown in Fig. 4.3. The parametric form of $\mathbf{S}(u, v)$ is

$$\mathbf{S}(u, v) = \sum_{i=0}^n \sum_{j=0}^m \mathbf{P}_{i,j} B_{i,n}(u) B_{j,m}(v). \quad (4.6)$$

Now we derive the equations of the boundaries of the patch $\mathbf{S}(u, v)$. A Bezier patch comprises two families of parameterized curves $\mathbf{C}(u)$ and $\mathbf{C}(v)$. Without loss of generality, the parameters u, v are chosen to lie in the interval $[0, 1]$. Thus the boundaries of $\mathbf{S}(u, v)$ are four curves $\mathbf{C}_1(v)$, $\mathbf{C}_2(v)$, $\mathbf{C}_3(u)$, and $\mathbf{C}_4(u)$, with respect to $u = 0, 1$ and $v = 0, 1$.

Figure 4.3: Bezier surface patch $S(u, v)$ and its control points.

Considering $C_1(v)$ where $u = 0$, we have

$$C_1(v) = S(0, v) = \sum_{j=0}^m P_{0,j} B_{j,m}(v) \quad (4.7)$$

since $B_{0,n}(0) = 1$ and $B_{i,n}(0) = 0$ for all $i \neq 0$. Similarly, we have the other three boundaries

$$C_2(v) = S(1, v) = \sum_{j=0}^m P_{n,j} B_{j,m}(v) \quad (4.8)$$

$$C_3(u) = S(u, 0) = \sum_{i=0}^n P_{i,0} B_{i,n}(u) \quad (4.9)$$

$$C_4(u) = S(u, 1) = \sum_{i=0}^n P_{i,m} B_{i,n}(u). \quad (4.10)$$

Obviously, C_{1-4} are Bezier curves defined by their control points, which are also part of the $(n+1) \times (m+1)$ control points of $S(u, v)$ in (4.6).

4.3.4 Generating Bezier Surface Patches

Given the patch boundaries represented by the 3D Bezier curves C_{1-4} found using the scheme described in Section 4.3.2, our final task is now to generate the Bezier patch. This is equivalent to finding all the other control points $P_{i,j}$ of $S(u, v)$ based on the known control points, $P_{0,j}$, $P_{n,j}$, $P_{i,0}$, and $P_{i,m}$, of C_{1-4} where $0 \leq i \leq n, 0 \leq j \leq m$.

Consider three new surfaces obtained from the boundaries:

$$S_1(u, v) = (1-u)S(0, v) + uS(1, v) \quad (4.11a)$$

$$S_2(u, v) = (1-v)S(u, 0) + vS(u, 1) \quad (4.11b)$$

$$S_3(u, v) = (1-u)(1-v)S(0, 0) + (1-u)vS(0, 1) \\ + u(1-v)S(1, 0) + uvS(1, 1). \quad (4.11c)$$

Here, (4.11a) is a surface passing through curves $S(0, v)$ and $S(1, v)$, (4.11b) is a surface passing through curves $S(u, 0)$ and $S(u, 1)$, and (4.11c) is a surface passing through the four

corners, $\mathbf{S}(0, 0)$, $\mathbf{S}(0, 1)$, $\mathbf{S}(1, 0)$, and $\mathbf{S}(1, 1)$, of the patch $\mathbf{S}(u, v)$. If we consider the surface defined by the sum $\mathbf{S}_1(u, v) + \mathbf{S}_2(u, v)$, we will find that each corner is counted twice. Hence, if we subtract $\mathbf{S}_3(u, v)$ from the sum, we will recover a surface that passes through the four boundaries with each corner counted once:

$$\begin{aligned}\mathbf{S}^*(u, v) &= \mathbf{S}_1(u, v) + \mathbf{S}_2(u, v) - \mathbf{S}_3(u, v) \\ &= (1 - u)\mathbf{S}(0, v) + u\mathbf{S}(1, v) + (1 - v)\mathbf{S}(u, 0) \\ &\quad + v\mathbf{S}(u, 1) - (1 - u)(1 - v)\mathbf{S}(0, 0) \\ &\quad - (1 - u)v\mathbf{S}(0, 1) - u(1 - v)\mathbf{S}(1, 0) - uv\mathbf{S}(1, 1).\end{aligned}\quad (4.12)$$

$\mathbf{S}^*(u, v)$ is called the Coons surface patch [19]. Note that this kind of surface patches gives not only an easy way of interpolation from the boundaries, but also good results in accordance with human vision observation.

Motivated by the Coons patch, we use a similar manner to interpolate all the control points $\mathbf{P}_{i,j}$ of the Bezier patch $\mathbf{S}(u, v)$ based on the known control points, $\mathbf{P}_{0,j}$, $\mathbf{P}_{n,j}$, $\mathbf{P}_{i,0}$, and $\mathbf{P}_{i,m}$, of the boundaries \mathbf{C}_{1-4} where $0 \leq i \leq n, 0 \leq j \leq m$. The interpolation equation is written as

$$\begin{aligned}\mathbf{P}_{i,j} &= (1 - \frac{i}{n})\mathbf{P}_{0,j} + \frac{i}{n}\mathbf{P}_{n,j} + (1 - \frac{j}{m})\mathbf{P}_{i,0} + \frac{j}{m}\mathbf{P}_{i,m} \\ &\quad - (1 - \frac{i}{n})(1 - \frac{j}{m})\mathbf{P}_{0,0} - (1 - \frac{i}{n})\frac{j}{m}\mathbf{P}_{0,m} \\ &\quad - \frac{i}{n}(1 - \frac{j}{m})\mathbf{P}_{n,0} - \frac{i}{n}\frac{j}{m}\mathbf{P}_{n,m}.\end{aligned}\quad (4.13)$$

The following proposition points out that the control points chosen in such a way generate a Bezier patch equivalent to the Coons patch.

Proposition 4.2 The Bezier patch $\mathbf{S}(u, v)$ defined by the $(n + 1) \times (m + 1)$ control points $\mathbf{P}_{i,j}$ in (4.13) is the same as the Coons patch in (4.12).

Proof. Substituting $\mathbf{P}_{i,j}$ in (4.13) into $\mathbf{S}(u, v)$ in (4.6) yields

$$\begin{aligned}\mathbf{S}(u, v) &= \sum_{i=0}^n \sum_{j=0}^m B_{i,n}(u)B_{j,m}(v) [(1 - \frac{i}{n})\mathbf{P}_{0,j} + \frac{i}{n}\mathbf{P}_{n,j}] \\ &\quad + \sum_{i=0}^n \sum_{j=0}^m B_{i,n}(u)B_{j,m}(v) [(1 - \frac{j}{m})\mathbf{P}_{i,0} + \frac{j}{m}\mathbf{P}_{i,m}] \\ &\quad - \sum_{i=0}^n \sum_{j=0}^m B_{i,n}(u)B_{j,m}(v) [(1 - \frac{i}{n})(1 - \frac{j}{m})\mathbf{P}_{0,0} \\ &\quad + (1 - \frac{i}{n})\frac{j}{m}\mathbf{P}_{0,m} + \frac{i}{n}(1 - \frac{j}{m})\mathbf{P}_{n,0} + \frac{i}{n}\frac{j}{m}\mathbf{P}_{n,m}].\end{aligned}\quad (4.14)$$

Further we simplify some terms in (4.14).

$$\sum_{i=0}^n B_{i,n}(u) \frac{i}{n} = \sum_{i=1}^n \binom{n}{i} u^i (1-u)^{n-i} \frac{i}{n} = u.$$

Similarly, we can derive

$$\begin{aligned} \sum_{i=0}^n B_{i,n}(u) \left(1 - \frac{i}{n}\right) &= 1 - u, & \sum_{j=0}^m B_{j,m}(v) \frac{j}{m} &= v, \\ \sum_{j=0}^m B_{j,m}(v) \left(1 - \frac{j}{m}\right) &= 1 - v. \end{aligned}$$

Hence, (4.14) becomes

$$\begin{aligned} \mathbf{S}(u, v) &= \sum_{j=0}^m [(1-u)\mathbf{P}_{0,j} + u\mathbf{P}_{n,j}] B_{j,m}(v) \\ &+ \sum_{i=0}^n [(1-v)\mathbf{P}_{i,0} + v\mathbf{P}_{i,m}] B_{i,n}(u) - (1-u)(1-v)\mathbf{P}_{0,0} \\ &- (1-u)v\mathbf{P}_{0,m} - u(1-v)\mathbf{P}_{n,0} - uv\mathbf{P}_{n,m} \\ &= (1-u)\mathbf{S}(0, v) + u\mathbf{S}(1, v) + (1-v)\mathbf{S}(u, 0) \\ &+ v\mathbf{S}(u, 1) - (1-u)(1-v)\mathbf{S}(0, 0) \\ &- (1-u)v\mathbf{S}(0, 1) - u(1-v)\mathbf{S}(1, 0) - uv\mathbf{S}(1, 1), \end{aligned}$$

which is equal to $\mathbf{S}^*(u, v)$ in (4.12), and thus completes the proof. \square

4.4 Results

We have implemented a complete tool for the object reconstruction from single 2D line drawings, including line drawing input, edge-vertex graph extraction, transformation of a line drawing, face identification, planar object reconstruction, and curved object reconstruction. All the algorithms are implemented in Visual C++, running on a 2 GHz Pentium IV PC. The two parameters m and n in Bezier patches are both set to 3. The weighting factors λ_{1-3} are chosen to be 0.65, 0.1, and 0.25, respectively.

A number of objects with curved faces have been drawn to test our approach. Fig. 4.4 shows part of the line drawings and their reconstruction results. Each result is displayed from two different viewpoints. The last two line drawings come from two scanned sketches drawn on paper, and the others are obtained by the inputs drawn on the screen with a mouse. We can see that the results accord with our visual perception quite well, although we do not have a precise definition of a 3D object corresponding to a given line drawing.

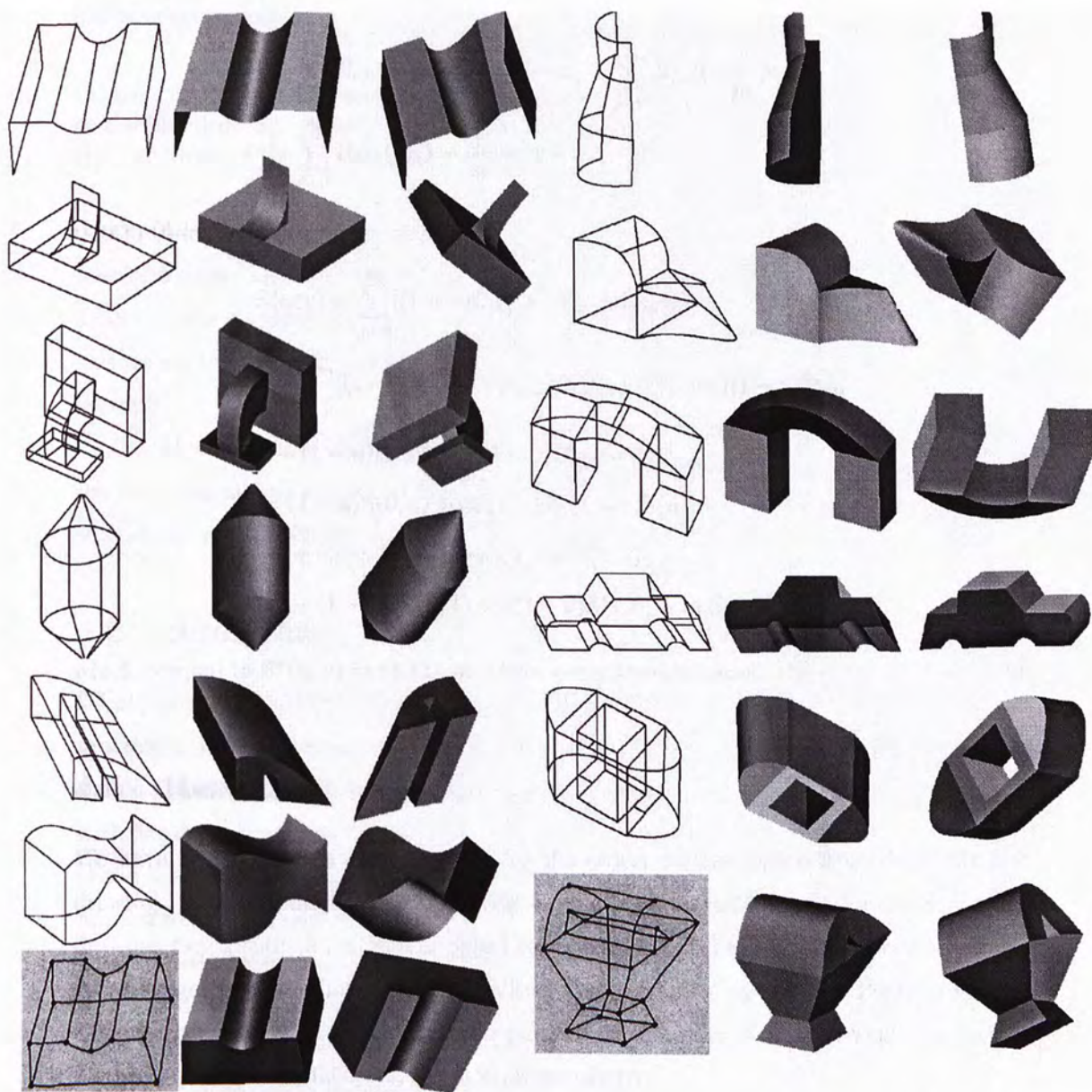


Figure 4.4: A set of line drawings and their reconstructed results, each shown from two views.

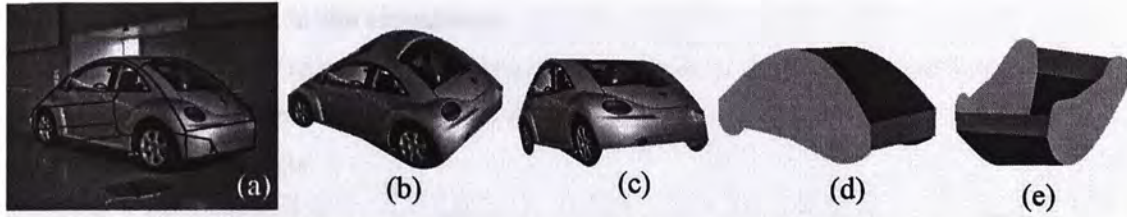


Figure 4.5: Reconstruction of a car from a color image. (a) Boundaries drawn by the user as the line drawing. (b) (c) Two views of the recovered 3D car with texture mapped. (d) (e) Two views of the 3D car without texture

Our tool can also be used for generating photo-realistic 3D models from 2D images. The user first draws lines and curves as the boundaries of an interested object. Then the tool automatically reconstructs the 3D shape of the object from the line drawing. Finally, the texture on the image is mapped onto the surface of the object. Fig. 4.5 shows such an example.

Our algorithm is also efficient and runs fast enough. After a line drawing is available, the tool does all the jobs automatically and can show the reconstructed object within two seconds for each of the line drawings illustrated in the section.

4.5 Summary

3D object reconstruction from single 2D line drawings has been a long-standing challenging problem in computer vision. Some methods exist in the literature but they can only handle simple planar objects. In this paper, we propose a novel reconstruction approach to dealing with line drawings representing objects with curved faces. It includes (1) the transformation of the line drawing of a curved object into the line drawing of a planar one, (2) face identification from the transformed line drawing, (3) reconstruction of the planar object from the transformed line drawing, and (4) reconstruction of the curved faces by the interpolation with the developed Bezier curves and surface patches. To the best of our knowledge, our work is the first attempt to provide a complete tool for the 3D reconstruction of curved objects from single 2D line drawings.

In the experimental section, a number of examples are given to demonstrate the ability of our approach to perform the 3D object reconstruction. The results are quite satisfactory and in accordance with human visual perception of the objects. The developed tool can also be used to create photo-realistic 3D models from images. Our algorithm is efficient; it can finish a reconstruction job within two seconds after a line drawing is available, for each of

the line drawings given in the experiments.

Future work in this research includes tackling objects with more complex surfaces, and developing an algorithm that can do the reconstruction incrementally in order to handle more complicated objects.

Chapter 5

Planar Limbs and Degen Generalized Cylinders

5.1 Introduction

Generalized cylinder (GC) has played an important role in computer vision since it was brought out in the 1970s. While studying GC models in human visual perception of shapes from contours, Marr [49] assumed that GC's limbs are planar curves. Later, Koenderink [33] and Ponce [61] pointed out that this assumption does not hold in general by giving some examples. In this paper, we show that straight homogeneous generalized cylinders (SHGCs) and tori (a kind of curved GCs) have planar limbs when viewed from points on specific straight lines. This property leads us to the definition and investigation of a new class of GCs, with the help of the surface model proposed by Degen for geometric modeling. We call them Degen generalized cylinders (DGCs), which include SHGCs, tori, quadrics, cyclides, and more other GCs into one model. Our rigorous discussion is based on projective geometry and homogeneous coordinates. We present some invariant properties of DGCs that reveal the relations among the planar limbs, axes, and contours of DGCs. These properties are useful for recovering DGC descriptions from image contours as well as for some other tasks in computer vision.

A generalized cylinder (GC) is a solid obtained by sweeping a planar region along an axis. The planar region is called the cross section of the GC and is not necessarily circular or constant. The axis can also be curved in space. This model was at first proposed by Binford in 1971 [6], and has received extensive attention and become popular in computer vision in the past three decades. Because of their ability to represent objects explicitly and their object-centered coordinate frames derivable from image data, GCs have been applied

to shape recovery [65], [23], [87], [66], [61], object modelling [53], [54], [26], [68], model-based segmentation and detection [94], [27], [63], modelling tree branches in computer graphics [7], recovering blood vessels from medical images [55], designing robot vision systems [9], etc.

From previous work on the study of the properties and recovery of GCs, we can roughly divide GCs into two groups: GC with straight axes and GCs with curved axes. In what follows, we call them straight GCs and curved GCs, respectively. Most of the work considers GCs in single views. Straight homogeneous generalized cylinders (SHGCs) are the most important subset of straight GCs, whose sweeping axes are straight and whose cross sections are scaled along the axes. SHGCs were first defined by Shafer and Kanade [67], and then studied extensively by many researchers [65], [87], [61], [94], [27], [60], [82], [85], [86].

Compared with SHGCs, less work on curved GCs has been carried out. The difficulty is mainly due to two facts: the projection of the axis of a curved GC may not be necessarily the axis of its 2D contours [59], and the angle between the axis and the cross section in the image no longer keeps constant [95]. To interpolate the axis of a curved GC in scattered data, Shani and Ballard proposed an iterative solution of minimizing the torsion of the axis [68]. In [66], Sayd et al. presented a scheme to recover a constrained subset of curved GCs with circular and constant cross sections. Ulupinar and Nevatia focused on a subset of GCs whose axes are planar curves and normal to the constant cross sections [84]. Zerroug and Nevatia studied the invariants and quasi-invariants of a subset of GCs with planar curved axes and with circular (not necessarily constant) cross sections [93]. In [26], Gross considered GCs with planar curved axes or with circular cross sections, and presented an algorithm to recover the GCs using image contours and reflectance information.

The analysis of the previous work on SHGCs and curved GCs is explicitly separate, focusing on special classes of GCs. In this chapter, starting from the discussion of the conditions when SHGCs and tori (a kind of curved GCs) have planar limbs, we define and study a new class of GCs, with the help of the surface model proposed by Degen for geometric modeling [21], [22]. We call them Degen generalized cylinders (DGCs), which include SHGCs, tori, quadrics, cyclides, and more other GCs into one model. Our rigorous discussion is based on projective geometry and homogeneous coordinates. We present some invariant properties of DGCs that reveal the relations among the planar limbs, axes, and contours of DGCs. We also discuss how the proposed properties can be used for recovering DGC descriptions from image contours, and for generating good initializations for a new 3D deformable DGC model in 3D data fitting and segmentation.



Figure 5.1: 2D Projections unable to fully describe the 3D information of the space curve.

5.2 Planar Limbs and View Directions

This section discusses two classes of GCs that have planar limbs when viewed from specific directions. These GCs with the property of planar limbs are the motivation of our work.

In this paper, image contours are referred to as the projections of *contour generators* that are curves in space. There are two kinds of contour generators: *limbs* and *edges* [61], [52]. Limb points are the points where the surface turns smoothly away from the observer, and edge points are those where the surface orientation is discontinuous. A limb is sometimes called a rim [49], viewpoint-dependent edge, or virtual edge [96].

Although a curve in 3D space can be formed freely, its projected contours cannot keep all the information of its 3D shape. Fig. 5.1 shows such a limitation. From the projection of a curve, one cannot judge whether it is planar or not in 3D space. To guess the ability of human vision on recovering 3D information from contours, Stevens assumed that one tends to interpret the 2D projection of a space curve as the projection of a planar curve [73], [74]. We can see this tendency from the projections in Fig. 5.1 if the space curve is not shown. In differential geometry, the torsion of a planar curve is zero, which was used by Shani and Ballard as the minimization criterion to recover 3D curved axes [68].

Marr also assumed that limbs are planar in human visual interpretation. With this assumption and other constraints, Marr showed that human beings always interpret the projected surface as part of a GC; limbs being planar is a basic assumption in the study of reconstructing object surfaces in Marr's fundamental vision theory [49].

However, this assumption does not hold generally as pointed out by Koenderink [33]. He showed that the contour of a torus, which is a curved GC, is often the projection of a non-planar limb. Later Ponce and Chelberg revealed that even SHGCs cannot possess planar limbs from all viewing directions [60]. Fig. 5.2 gives such an example, where the bold black curves are the intersection of a plane and the GC's surface. From the two viewing

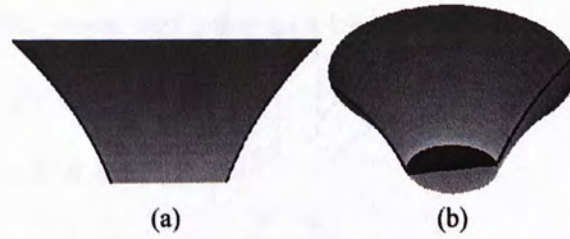


Figure 5.2: (a) Planar limbs. (b) Non-planar limbs.

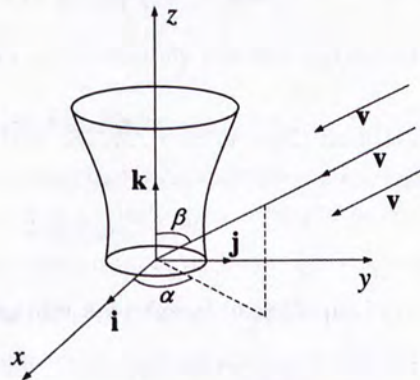


Figure 5.3: The coordinate system with a SHGC and the viewing direction \mathbf{v} .

directions, the limbs in Fig. 5.2(a) are planar, but the limbs in Fig. 5.2(b) are not. Now we discuss in what conditions SHGCs and tori can have planar limbs.

We use the similar notation and the coordinate system as those in [61] and [60]. Suppose that the axis of a SHGC coincides with the z -axis as shown in Fig. 5.3. The surface of a SHGC can be represented in the polar coordinate system by

$$\mathbf{x}(z, \theta) = \rho(\theta)r(z) \cos \theta \mathbf{i} + \rho(\theta)r(z) \sin \theta \mathbf{j} + z \mathbf{k} \tag{5.1}$$

where $z \in [a, b]$, $\theta \in [0, 2\pi]$, and ρ defines the reference cross section on the x - y plane, and r defines the scaling sweeping rule of the SHGC. Let \mathbf{v} be the viewing direction, and \mathbf{n} be the normal vector to the surface at the points on a limb. Then according to the definition of limbs, we have the relation

$$\mathbf{v} \cdot \mathbf{n} = 0. \tag{5.2}$$

Theorem 5.1 A SHGC has planar limbs when the viewing direction is normal to the axis of the SHGC under orthographic projection.

Proof. Assume the viewing direction is given by its spherical coordinate (α, β) (see Fig. 5.3). Then

$$\mathbf{v} = \sin \beta \cos \alpha \mathbf{i} + \sin \beta \sin \alpha \mathbf{j} + \cos \beta \mathbf{k}. \tag{5.3}$$

With (5.2), Ponce [61] proved that points on a limb satisfy

$$\rho^2 r' \cos \beta = [\rho(\theta) \cos(\theta - \alpha) + \rho'(\theta) \sin(\theta - \alpha)] \sin \beta. \quad (5.4)$$

When \mathbf{v} is normal to \mathbf{k} , $\beta = 90^\circ$. Hence

$$\rho(\theta) \cos(\theta - \alpha) + \rho'(\theta) \sin(\theta - \alpha) = 0, \quad (5.5)$$

which implies a function θ of α only (independent of z), i.e., $\theta = f(\alpha)$. We can write the limb equation as

$$\begin{aligned} \mathbf{l}(z) &= \mathbf{x}(z, f(\alpha)) \\ &= r(z)\rho(f(\alpha))(\cos f(\alpha)\mathbf{i} + \sin f(\alpha)\mathbf{j}) + z\mathbf{k} \\ &= r(z)\mathbf{u}(\alpha) + z\mathbf{k}, \end{aligned} \quad (5.6)$$

where $\mathbf{u}(\alpha) = \rho(f(\alpha))(\cos f(\alpha)\mathbf{i} + \sin f(\alpha)\mathbf{j})$. From (5.6),

$$\mathbf{l}''(z) = r''(z)\mathbf{v}(\alpha) \quad (5.7)$$

$$\mathbf{l}'''(z) = r'''(z)\mathbf{v}(\alpha) \quad (5.8)$$

$$\mathbf{l}''(z) \times \mathbf{l}'''(z) = \mathbf{0}. \quad (5.9)$$

Hence $\mathbf{l}'(z) \times \mathbf{l}''(z) \times \mathbf{l}'''(z) = \mathbf{0}$, which indicates that the limb is a planar curve, because the torsion of a planar curve is equal to zero [15]. \square

Although a torus (a curved GC) does not belong to the class of SHGCs, it also has planar limbs when viewed from specific directions. Note that the axis of a torus is a circle inside the torus.

Theorem 5.2 A torus has planar limbs while viewed from a point where the line through the point and the torus center is orthogonal to the torus axis.

Proof. Without loss of generality, we assume that the axis of the torus is located on the x - y plane, the center of it coincides with the origin, and the viewpoint is at lk , as shown in Fig. 5.4. Then the surface of the torus can be parameterized by [50]

$$\begin{aligned} \mathbf{x}(z, \theta) &= (R - r \cos z) \cos \theta \mathbf{i} \\ &\quad + (R - r \cos z) \sin \theta \mathbf{j} + r \sin z \mathbf{k}. \end{aligned} \quad (5.10)$$

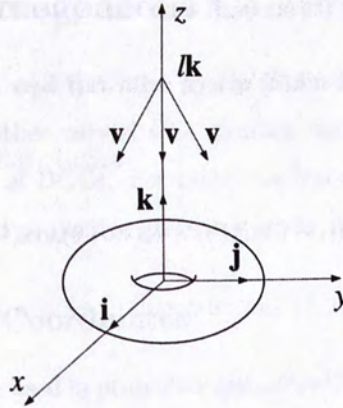


Figure 5.4: A torus with the viewpoint at the z-axis.

The normal to the surface is given by

$$\begin{aligned}
 \mathbf{n}(z, \theta) &= \frac{\partial \mathbf{x}}{\partial z} \times \frac{\partial \mathbf{x}}{\partial \theta} \\
 &= -r \cos z (R - r \cos z) \cos \theta \mathbf{i} \\
 &\quad - r \cos z (R - r \cos z) \sin \theta \mathbf{j} \\
 &\quad + (R + r \sin z)(R - r \cos z) \mathbf{k}.
 \end{aligned} \tag{5.11}$$

The viewing direction from $l\mathbf{k}$ to the surface is

$$\mathbf{v}(z, \theta) = \mathbf{x}(z, \theta) - l\mathbf{k}. \tag{5.12}$$

Substituting \mathbf{v} in (5.12) and \mathbf{n} in (5.11) into (5.2) yields

$$rl \sin z + Rr \cos z - Rr \sin z - r^2 + Rl = 0, \tag{5.13}$$

which implies a function z of l only (independent of θ). i.e., $z = g(l)$. Thus we can write the limb equation as

$$\begin{aligned}
 \mathbf{l}(\theta) &= \mathbf{x}(g(l), \theta) \\
 &= (R - r \cos g(l))(\cos \theta \mathbf{i} + \sin \theta \mathbf{j}) + r \sin g(l) \mathbf{k}.
 \end{aligned} \tag{5.14}$$

It is easy to show

$$\mathbf{l}'(\theta) \times \mathbf{l}''(\theta) \times \mathbf{l}'''(\theta) = \mathbf{0}. \tag{5.15}$$

Thus the limb is planar since its torsion is zero. \square

5.3 DGCs in Homogeneous Coordinates

We have shown that SHGCs and tori have planar limbs when viewed from some specific directions. There are also other curved GCs sharing the same property. This property leads us to the investigation of DGCs. For mathematical convenience, we will mainly use homogeneous coordinates and projective geometry in the following discussion of DGCs.

5.3.1 Homogeneous Coordinates

Homogeneous coordinates are used in projective geometry [24], [72]. They are a useful tool in computer vision and graphics. Points in homogeneous coordinates are represented by vectors $\mathbf{p} = (w, x, y, z)^T \in \mathbb{R}^4 \setminus \{(0, 0, 0, 0)^T\}$. The w, x, y, z are called homogeneous coordinates of \mathbf{p} . \mathbf{p} and $\rho\mathbf{p}$ with $\rho \in \mathbb{R} \setminus \{0\}$ define the same point. Given a point $\mathbf{p} = (w, x, y, z)^T$ with $w \neq 0$ in homogeneous coordinates, its corresponding point $\bar{\mathbf{p}}$ in Cartesian coordinates is

$$\bar{\mathbf{p}} = (\bar{x}, \bar{y}, \bar{z})^T = \left(\frac{x}{w}, \frac{y}{w}, \frac{z}{w}\right)^T. \quad (5.16)$$

If $w = 0$, the point $(0, x, y, z)$ stands for a point at *infinity* (called an *ideal point*).

Orthogonal projection can be treated as a special case of perspective projection when the viewpoint is at infinity. Thus under perspective projection, Proposition 5.1 states that a SHGC has planar limbs when the viewpoint is at infinity and the viewing direction is normal to the axis of the SHGC.

Using homogeneous coordinates, points on a straight line \mathbf{L} can be represented by

$$\mathbf{L} = \alpha\mathbf{a} + \beta\mathbf{b}, \quad (5.17)$$

where $\alpha, \beta \in \mathbb{R}$ and \mathbf{a}, \mathbf{b} are two independent points in the projective space. In what follows, we denote the line \mathbf{L} by $\mathbf{a} \wedge \mathbf{b}$. Similarly, points on a plane \mathbf{P} can be represented by

$$\mathbf{P} = \alpha\mathbf{a} + \beta\mathbf{b} + \gamma\mathbf{c} \quad (5.18)$$

where $\alpha, \beta, \gamma \in \mathbb{R}$ and $\mathbf{a}, \mathbf{b}, \mathbf{c}$ are three independent points. We denote the plane \mathbf{P} by $\mathbf{a} \wedge \mathbf{b} \wedge \mathbf{c}$. Therefore, a curve $\mathbf{C}(s)$ is planar if it can be written in this form

$$\mathbf{C}(s) = p_1(s)\mathbf{a} + p_2(s)\mathbf{b} + p_3(s)\mathbf{c}. \quad (5.19)$$

To verify whether a curve is planar or not, this way is more convenient than calculating the torsion of the curve in Cartesian coordinates.

5.3.2 Degen Surfaces

Degen proposed a novel surface model for geometric modelling in [21] and [22]. We call those surfaces *Degen surfaces*. They cover a wide range of curved surfaces such as those showed in Fig. 5.5. A Degen surface is parameterized by the following equation in homogeneous coordinates

$$\begin{aligned}\mathbf{X}(u, v) &= \alpha(u)\mathbf{a} + \beta(u)\mathbf{b} + \gamma(v)\mathbf{c} + \delta(v)\mathbf{d} \\ &= \mathbf{p}(u) + \mathbf{q}(v),\end{aligned}\quad (5.20)$$

where $\mathbf{p}(u) = \alpha(u)\mathbf{a} + \beta(u)\mathbf{b}$, $\mathbf{q}(v) = \gamma(v)\mathbf{c} + \delta(v)\mathbf{d}$, $u \in [u_1, u_2]$, $v \in [v_1, v_2]$, \mathbf{a} , \mathbf{b} , \mathbf{c} , \mathbf{d} are independent, and $\alpha, \beta, \gamma, \delta$ are certain functions. The two straight lines $\mathbf{a} \wedge \mathbf{b}$ and $\mathbf{c} \wedge \mathbf{d}$ are called the axes of the Degen surface.

5.3.3 DGCs

Before defining DGCs, we show that SHGCs and tori can be represented in the form of Degen surfaces in homogeneous coordinates. The parameterized surface of a SHGC in homogeneous coordinates is simply

$$\mathbf{X}(u, v) = (1, \rho(u)r(v) \cos u, \rho(u)r(v) \sin u, v)^T, \quad (5.21)$$

where the z and θ in (5.1) are replaced by u and v , respectively. Then $\mathbf{X}(u, v) = \mathbf{p}(u) + \mathbf{q}(v)$ with

$$\mathbf{p}(u) = (0, \rho(u) \cos u, \rho(u) \sin u, 0)^T \quad (5.22)$$

$$\mathbf{q}(v) = \frac{1}{r(v)}(1, 0, 0, v)^T. \quad (5.23)$$

Furthermore

$$\mathbf{p}(u) = \rho(u)(\cos u)\mathbf{a} + \rho(u)(\sin u)\mathbf{b} \quad (5.24)$$

$$\mathbf{q}(v) = \frac{1}{r(v)}\mathbf{c} + \frac{v}{r(v)}\mathbf{d} \quad (5.25)$$

with $\mathbf{a} = (0, 1, 0, 0)^T$, $\mathbf{b} = (0, 0, 1, 0)^T$, $\mathbf{c} = (1, 0, 0, 0)^T$, $\mathbf{d} = (0, 0, 0, 1)^T$.

Similarly, replacing the z and θ in (5.10) with u and v , respectively, we can show that a

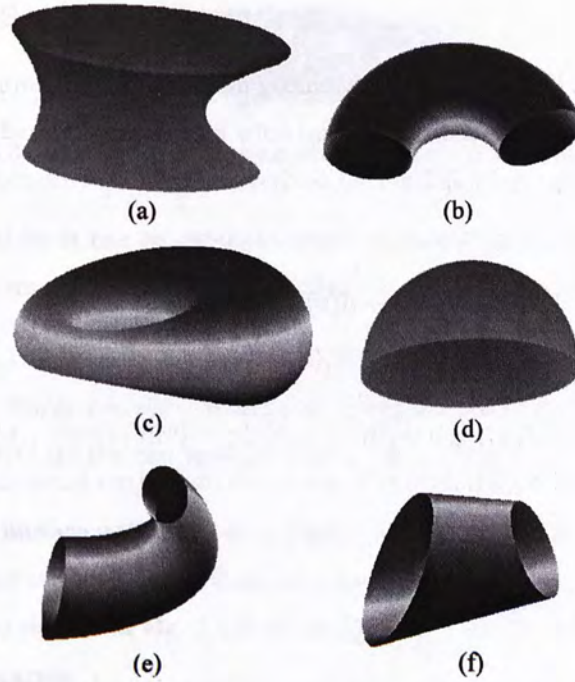


Figure 5.5: Some Degen Surfaces, among which (a), (b), (c) and (d) are a SHGC, an open torus, a cyclide and a quadric, respectively.

torus belongs to a Degen surface by

$$\begin{aligned} \mathbf{p}(u) &= \frac{1}{R - r \cos u} (1/r, 0, 0, \sin u)^T \\ &= \frac{1}{r(R - r \cos u)} \mathbf{a} + \frac{\sin u}{R - r \cos u} \mathbf{b} \end{aligned} \quad (5.26)$$

$$\begin{aligned} \mathbf{q}(v) &= \frac{1}{r} (0, \cos v, \sin v, 0)^T \\ &= \frac{\cos v}{r} \mathbf{c} + \frac{\sin v}{r} \mathbf{d}, \end{aligned} \quad (5.27)$$

with $\mathbf{a} = (1, 0, 0, 0)^T$, $\mathbf{b} = (0, 0, 0, 1)^T$, $\mathbf{c} = (0, 1, 0, 0)^T$, $\mathbf{d} = (0, 0, 1, 0)^T$.

Definition 5.1 On a Degen surface, when $v = v_0$ is fixed, the curve $\mathbf{C}_1(u) = \mathbf{X}(u, v_0)$ is called a u -curve; when $u = u_0$ is fixed, the curve $\mathbf{C}_2(v) = \mathbf{X}(u_0, v)$ is called a v -curve.

In the above examples, the u -curves of a SHGC are $(0, \rho(u) \cos u, \rho(u) \sin u, 0)^T + \mathbf{q}(v_0)$, which are closed when $u \in [0, 2\pi]$; the v -curves of the SHGC are $\mathbf{p}(u_0) + \frac{1}{r(v)}(1, 0, 0, v)^T$. Both the u -curves and v -curves of a torus are circles, which are also closed.

On a Degen surface with $u \in [u_1, u_2]$, $v \in [v_1, v_2]$, the family of u -curves $\{\mathbf{C}_1(u) = \mathbf{X}(u, v_0) \mid v_0 \in [v_1, v_2]\}$ covers the whole surface. Thus a Degen surface can be seen as a surface obtained by sweeping a u -curve when v_0 varies from v_1 to v_2 . If the u -curve is closed,



Figure 5.6: A Degen surface with neither u -curves nor v -curves closed.

the region bounded by it can be regarded as the cross section of a GC. Note that all the u -curves and v -curves are planar as stated in Lemma 5.3 in Section 5.4.

Definition 5.2 A Degen generalized cylinder (DGC) is a solid bounded by a Degen surface $\mathbf{X}(u, v) = \alpha(u)\mathbf{a} + \beta(u)\mathbf{b} + \gamma(v)\mathbf{c} + \delta(v)\mathbf{d}$ with closed u -curves, or closed v -curves, or both. The axes of the DGC are the two straight lines $\mathbf{a} \wedge \mathbf{b}$ and $\mathbf{c} \wedge \mathbf{d}$.

Obviously, the surface of a DGC is a Degen surface. However, a Degen surface with neither u -curves nor v -curves closed does not form a DGC. Fig. 5.6 gives such an example. The Degen surfaces showed in Fig. 5.5 form six DGCs if the cross sections are considered as regions instead of curves.

It should be emphasized that a conventional GC has only one axis and the axis of a conventional curved GC is a curve. It is often more difficult to recover curved axes than to recover straight axes.

5.4 Properties of DGCs

In this section, we present the properties of DGCs that are useful for some computer vision tasks.

Theorem 5.3 The axis of a SHGC coincides with one of the two axes of the DGC that is the corresponding representation of the SHGC in homogeneous coordinates. Another axis of the DGC is a line at infinity.

Proof. When a SHGC is written as (5.1), its axis is the z -axis (Fig. 5.3). The same SHGC can be represented in the form of a DGC as in (5.21)–(5.25). One axis of the DGC is $\mathbf{c} \wedge \mathbf{d}$, i.e., a line passing through $(1, 0, 0, 0)^T$ and $(0, 0, 0, 1)^T$, which denotes the z -axis in homogeneous coordinates. Another axis of the DGC is a line at infinity, which passes through the two ideal points $\mathbf{a} = (0, 1, 0, 0)^T$ and $\mathbf{b} = (0, 1, 0, 0)^T$. \square

It is also easy to find the two axes of a torus when it is represented in the form of a DGC. Suppose a torus in Euclidean geometry is expressed by (5.10). From (5.26) and (5.27), we see that one axis of the torus is $\mathbf{a} \wedge \mathbf{b}$ with $\mathbf{a} = (1, 0, 0, 0)^T$ and $\mathbf{b} = (0, 0, 0, 1)^T$, which

is the z -axis in homogeneous coordinates. Another axis is $\mathbf{c} \wedge \mathbf{d}$ with $\mathbf{c} = (0, 1, 0, 0)^T$ and $\mathbf{d} = (0, 0, 1, 0)^T$, which is a line through the two ideal points \mathbf{c} and \mathbf{d} at infinity.

As pointed out in Propositions 5.1 and 5.2, both SHGCs and tori have planar limbs when viewed from the special directions. Now we show that all DGCs have this property. At first, we give two lemmas that are proved in [21].

Lemma 5.3 All the u -curves and v -curves of a DGC are planar.

Lemma 5.4 All the tangent planes on a u -curve $\mathbf{X}(u, v_0)$ (v -curve $\mathbf{X}(u_0, v)$, respectively) pass through the same point $\gamma'(v_0)\mathbf{c} + \delta'(v_0)\mathbf{d}$ ($\alpha'(u_0)\mathbf{a} + \beta'(u_0)\mathbf{b}$, respectively).

Theorem 5.4 A DGC has planar limbs when viewed from points on its two axes $\mathbf{a} \wedge \mathbf{b}$ and $\mathbf{c} \wedge \mathbf{d}$, and the planar limbs are u -curves and v -curves.

Proof. From Lemma 5.4, we know that all the tangent planes on a u -curve $\mathbf{X}(u, v_0)$ pass through the point $\gamma'(v_0)\mathbf{c} + \delta'(v_0)\mathbf{d}$. All such points with different values of v_0 lie on the axis $\mathbf{c} \wedge \mathbf{d}$. Therefore, if the DGC is observed from one of the points, the viewing directions must lie on these tangent planes at points on the u -curves. Thus the u -curve becomes a limb of the DGC. By Lemma 5.3, the limb is planar. Similarly, the DGC has planar limbs when observed from points on another axis $\mathbf{a} \wedge \mathbf{b}$. \square

Theorem 5.5 For any two contour points from the same u -curve (v -curve, respectively), the tangents to the contours at the two points intersect on the projection of the axis $\mathbf{c} \wedge \mathbf{d}$ ($\mathbf{a} \wedge \mathbf{b}$, respectively).

Proof. From Lemma 5.4, all the tangent planes of the u -curve (v -curve, respectively) meet at the same point on the axis $\mathbf{c} \wedge \mathbf{d}$ ($\mathbf{a} \wedge \mathbf{b}$, respectively). Since the tangent plane at a point of the limb is projected onto the tangent at the corresponding point on the contour generated by the limb [83], this theorem holds. \square

Fig. 5.7 illustrates this invariant property. Note that when a DGC is a SHGC, one axis becomes the axis of the SHGC (Proposition 5.3). Thus the SHGC's invariant property stated in Lemma 4 in Ponce et al.'s work [61] becomes a special case of Proposition 5.5.

Definition 5.5 Let $\mathbf{X}(u, v_i)$ and $\mathbf{X}(u, v_j)$ be two u -curves of a DGC. Two points $\mathbf{X}(u_k, v_i)$ and $\mathbf{X}(u_k, v_j)$ on the two u -curves define a line of correspondence from the two u -curves. Let $\mathbf{X}(u_m, v)$ and $\mathbf{X}(u_n, v)$ be two v -curves of a DGC. Two points $\mathbf{X}(u_m, v_q)$ and $\mathbf{X}(u_n, v_q)$ on the two v -curves define a line of correspondence from the two v -curves.

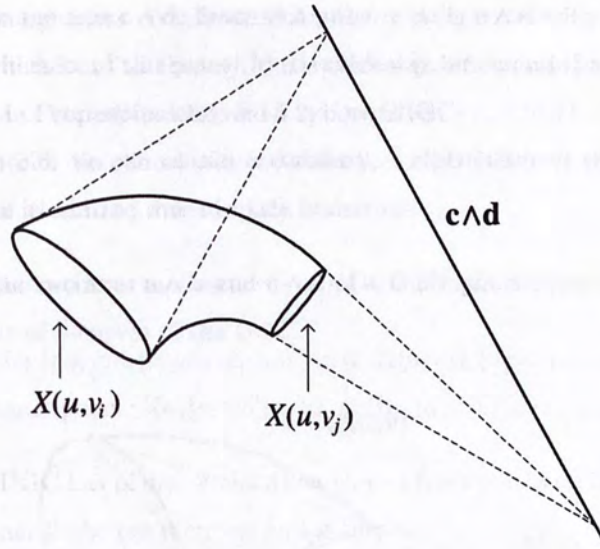


Figure 5.7: Illustration of Proposition 5.5.

Theorem 5.6 All the lines of correspondence from any two u -curves (v -curves, respectively) of a DGC intersect at the same point on the axis $c \wedge d$ ($a \wedge b$, respectively).

Proof. Let $X(u, v_i)$ and $X(u, v_j)$ be two u -curves as shown in Fig. 5.8, the line of correspondence passing through the two points $X(u_k, v_i)$ and $X(u_k, v_j)$ can be expressed as

$$X(u_k, v_i) + \lambda X(u_k, v_j), \quad \lambda \in \mathbb{R}. \tag{5.28}$$

When $\lambda = -1$,

$$\begin{aligned} X(u_k, v_i) - X(u_k, v_j) &= [\mathbf{p}(u_k) + \mathbf{q}(v_i)] - [\mathbf{p}(u_k) + \mathbf{q}(v_j)] \\ &= \mathbf{q}(v_i) - \mathbf{q}(v_j) \\ &= [\gamma(v_i) - \gamma(v_j)]\mathbf{c} + [\delta(v_i) - \delta(v_j)]\mathbf{d}, \end{aligned} \tag{5.29}$$

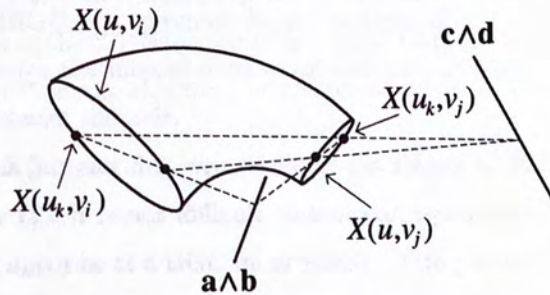


Figure 5.8: Illustration of Proposition 5.6.

which is a point on the axis $\mathbf{c} \wedge \mathbf{d}$. Since this point is independent of u_k , all such lines from the two u -curves intersect at this point. In the same way, we can also prove that the theorem is true for the lines of correspondence from two v -curves. \square

From Theorem 5.6, we can obtain a corollary, the geometry of which is illustrated in Fig. 5.9. The proof is omitted due to space limitation.

Corollary 5.6 The two axes $\mathbf{a} \wedge \mathbf{b}$ and $\mathbf{c} \wedge \mathbf{d}$ of a DGC can be determined from a pair of u -curves and a pair of v -curves of the DGC.

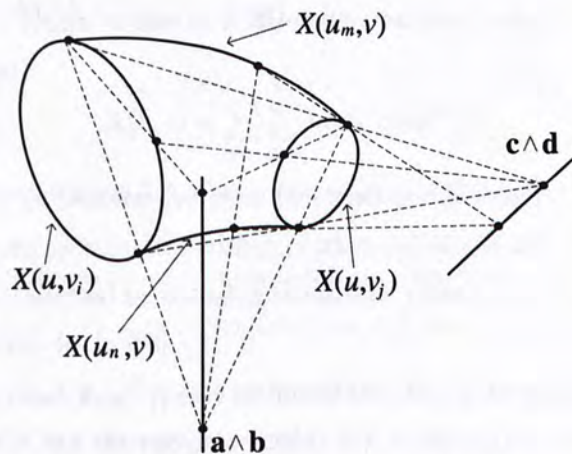


Figure 5.9: Illustration of Corollary 5.6.

5.5 Potential Applications

5.5.1 Recovery of DGC Descriptions

In Lemma 4 in [61], Ponce et al. presented their main finding which states that tangents to the contours at the points of the same cross section of a SHGC intersect at a point on the projection of the axis. With this invariant property, they developed an algorithm to recover the description of a SHGC, i.e., to search for the straight axis of a SHGC in the image. The algorithm first computes the intersections of all tangents at edgels, and then employs the Hough transform to detect the axis.

From Propositions 5.3 and 5.5, we can find that Ponce et al.'s Lemma 4 is a special case of our Theorem ???. It is not difficult to see that Ponce et al.'s algorithm is directly applicable to finding one axis of a DGC in an image. With the edge points that contribute to the axis in the Hough space, the contours of the DGC can also be detected.

5.5.2 Deformable DGCs

A 3D deformable model is useful for model-based segmentation, tracking, recognition, and fitting of objects in 3D images. A 3D image can be a range image or a 3D medical image where an image point denotes its location in 3D space. In [80], Terzopoulos et al. proposed a deformable cylinder model using locally deformable techniques. Later Terzopoulos and Metaxas used superquadrics as the global shape model for more flexible deformation [79]. Since DGCs can represent a wide range of object shapes, they can also be used as a good global deformable model with much freedom and efficient calculation.

The equation of a Degen surface in (5.20) can be written as an equivalent tensor product Bezier representation

$$\mathbf{X}(u, v) = \sum_{i=0}^n \sum_{j=0}^m \mathbf{b}_{ij} B_i^n(u) B_j^m(v), \quad (5.30)$$

where $B_i^n(u)$ and $B_j^m(v)$ are the Bernstein functions, and \mathbf{b}_{ij} are $(n+1) \times (m+1)$ control points in homogeneous coordinates satisfying certain conditions [22]. This expression shows that a DGC can be deformed by changing the control points, and is more flexible than the locally deformable cylinder in [80].

For a point $\mathbf{X} = (w, x, y, z)^T$ on the surface of the DGC in homogeneous coordinates, its corresponding point $\bar{\mathbf{x}}$ in Cartesian coordinates is $\bar{\mathbf{x}} = (x/w, y/w, z/w)^T$. Similarly to the deformation scheme in [79], a point $\bar{\mathbf{r}}$ on the deformable DGC can be expressed by

$$\bar{\mathbf{r}} = \bar{\mathbf{t}} + \mathbf{R}(\bar{\mathbf{x}} + \bar{\mathbf{s}}), \quad (5.31)$$

where $\bar{\mathbf{t}}$ is a reference point at the starting end of the deformable DGC, $\bar{\mathbf{s}}$ is a function that controls the local deformation from the global model, and \mathbf{R} denotes a rotation matrix. The deformable DGC in (5.31) can be manipulated using the similar external forces and numerical simulation as those in [79]. However, the deformation of this model requires linear calculation (see (5.30)) instead of the nonlinear deformation of superquadrics [79].

When a deformable model is applied to image data, a good initialization is often necessary which brings the model to the object of interest as closely as possible in order to produce satisfactory fitting within reasonable time. For the application of our deformable DGC onto a 3D image, if a given set S of points from the image (or part of the image) is known to form an approximate DGC, the properties developed in Section 5.4 can be used to generate a good initialization as described below.

With S , the limbs of the object surface in 3D space can be obtained by setting an arbitrary viewpoint with a simple algorithm. After trying a number of viewpoints (say, 50) in 3D space randomly, we can find a set of limbs. The pair of limbs that exhibit best planarity

at some viewpoint is what we want to have. This pair can be treated as two u -curves of the DGC (Theorem 5.4). Then we use two planes that are perpendicular to the limbs to cut the limbs, resulting in a pair of v -curves approximately. According to Corollary 5.6, these two pairs of curves can be used to find the two axes of the DGC. Therefore, we can approximately generate a DGC with the found axes, u -curves, and v -curves. This DGC is a good initialization for the deformable DGC model. This technique is also useful for the initialization for fitting DGCs to 3D data where the deformation is not performed.

5.6 Summary

GCs have been used in many applications of computer vision. Previous work on GCs focuses on relatively narrow sets of GCs. In this paper, we have proposed a new set of GCs, called Degen generalized cylinders (DGCs). DGCs cover a wide range of GCs, including SHGCs, tori, quadrics, cyclides, and more other GCs into one unified model. We have presented a number of properties existing in DGCs. Our rigorous discussion is based on homogeneous coordinates in projective geometry, which is more general than Euclidean geometry. The invariant properties of DGCs reveal the relations among the planar limbs, axes, and contours of DGCs. We also discuss how the properties can be used for recovering DGC descriptions from image contours, and for generating good initializations for a new 3D deformable DGC model in 3D data fitting and segmentation.

Chapter 6

Conclusion and Future Work

This thesis examines the issues related with reconstructing 3D object from line drawings. The main research contributions are:

1. Presenting a new criteria based on symmetry and planarity for reconstructing 3D objects from line drawings. This criteria is shown to be more robust than former MSDA and other measure, and is necessary for the second step work: reconstruction from line drawings without hidden lines.
2. Firstly exploring possibility to reconstruct 3D objects from line drawings while hidden lines are not shown.
3. Firstly generalizing the reconstruction of 3D objects with planar faces only to the objects with both planar faces and curved faces.
4. Presenting a new curved surface model named DGC. We prove that DGCs enjoy many nice invariants, which can be regarded as a generalization of the work by Marr [49] and Ponce [61]. DGC unified SHGC, tori, and some quadric surfaces into one model, which is expected to be the atoms for future reconstruction work.

With the current results, we can expect many directions for future work. Some of the options are described below, and we hope one or two of them to flourish in the future work.

New Query Interface for 3D Object Retrieval Current work in 3D object retrieval [56], [92] uses an existing models as the query for 3D retrieval. Our next work will be focus on the apply the algorithms in Chap. 4 as providing more comfortable and straight forwards query tools to the retrieval. For this aim, we have done some work recently which is planned to submitted to ECCV [14] later.

Applications of DGCs Another subsequent work is to explore the applications of DGCs based on theoretical work in Chap. 5, such as the reconstruction of complex surfaces, connecting and blending two quadric surfaces using DGCs.

Extracting Line Drawings from Images There have been mature algorithms to detect edges from images. However, to extract a good line drawings from images require more efforts on the edge-grouping and simplifying. Although there has been some elementary work for some simply models [65], this work is still mostly untouched.

Other Directions Other research topics includes present the optimization-based reconstruction into a Bayesian framework, finding a faster and more robust optimization algorithms to perform the optimization, reconstruction under perspective projection, etc.

To conclude, our work on reconstruction 3D objects from line drawings has generalized the previous work to many cases they cannot handel. We believe many interesting and exciting directions still lie ahead.

Bibliography

- [1] S. Ablameyko, A. Gorelik V. Bereishik, and S. Medvedev. 3D object reconstruction from engineering drawing projections. *Computing & Control Engineering Journal*, 10(6):277–284, 1999.
- [2] S.C. Agarwal and J.W.N. Waggenspack. Decomposition method for extracting face topologies from wireframe models. *Computer-Aided Design*, 24(3):123–140, 1992.
- [3] S. Bagali and J.W.N. Waggenspack. A shortest path approach to wireframe to solid model conversion. *Proc. 3rd Symp. Solid Modeling and Applications*, pages 339–349, 1995.
- [4] M. Berger. *Geometrie*. Paris: CEDIC/Fernand Nathan, 1977.
- [5] P. Bezier. Style, mathematics and nc. *Computer-Aided Design*, 22:524–526, 1990.
- [6] T.O. Binford. Visual perception by computer. *IEEE Conf. Systems and Control*, 1971.
- [7] Jules Bloomenthal. Modeling the mighty maple. *ACM Proc. SIGGRAPH '85*, 19(3), 1985.
- [8] M. Brady and A. Yuille. An extremum principle for shape from contour. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 6:288–301, 1984.
- [9] R. Brooks, R. Greiner, and T. O. Binford. The ACRONYM model-based vision system. *Proc. of 6th Int'l Joint Conf. Artificial Intelligence*, pages 105–113, 1979.
- [10] E. Brown and P.S. Wang. 3D object recovery from 2D images: A new approach. *SPIE Proc. Robotics and Computer Vision*, 2904:138–145, 1996.
- [11] L. Cao, J. Liu, and X. Tang. Inference of hidden topology and reconstruction of complete 3D objects from 2D line drawings. *To be submitted to IEEE Trans. Pattern Analysis and Machine Intelligence*.
- [12] L. Cao, J. Liu, and X. Tang. 3D object reconstruction from a single 2D line drawing without hidden lines. *IEEE Proc. Int'l Conf. Computer Vision*, 2005.
- [13] L. Cao, J. Liu, and X. Tang. Degen generalized cylinders and their properties. *to be submitted to European Conference on Computer Vision*, 2006.
- [14] L. Cao, J. Liu, and X. Tang. A novel enquiry interface for 3d shape retrieval. *to be submitted to European Conference on Computer Vision*, 2006.

- [15] Do Carmo. *Differential Geometry of Curves and Surfaces*. Englewood Cliffs, N.J. : Prentice Hal, 1976.
- [16] C. Cheng, W. Liu, and H. Zhang. Image retrieval based on region shape similarity. *Proc. 13th SPIE Symposium on Electronic Imaging — Storage and Retrieval for Image and Video Databases*, 2001.
- [17] M.B. Clowes. On seeing things. *Artificial Intelligence*, 2:79–116, 1971.
- [18] M.C. Cooper. The interpretations of line drawings with contrast failure and shadows. *Int'l Journal of Computer Vision*, 43(2):75–97, 2001.
- [19] A. Davies and P. Samuels. *An Introduction to Computational Geometry for Curves and Surfaces*. New York: Oxford University Press Inc., 1996.
- [20] P.E. Debevec, C. J. Yaylor, and J. Malik. Modeling and rendering architecture from photographs: a hybrid geometry- and image-based approach. *ACM Proc. SIGGRAPH'96*, pages 11–20, 1996.
- [21] Wendelin L. F. Degen. Nets with plane silhouettes. *Proc. IMA Conf. the Mathematics of Surfaces V*, pages 117–133, 1994.
- [22] Wendelin L. F. Degen. Conjugate silhouette nets. *Curve and surface design : Saint-Malo 99*, pages 37–44, 2000.
- [23] M. Dhome, R. Glachet, and J.T. Lapreste. Recovering the scaling function of a SHGC from a single perspective view. *IEEE Proc. Computer Vision and Pattern Recognition*, pages 36–41, 1992.
- [24] E.A. Maxwell. *General Homogeneous Coordinates in Space of Three Dimensions*. Cambridge University Press, 1951.
- [25] R.C. Gonzalez and R.E. Woods. *Digital Image Processing, 2nd Edition*. Prentice Hall, 2002.
- [26] Ari D. Gross. Analyzing generalized tubes. *SPIE Proc. Intelligent Robots and Computer Vision XIII: 3D Vision, Product*, 2354, 1994.
- [27] Ari D. Gross and Terrance E. Boult. Recovery of SHGCs from a single intensity view. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 18(2):161–180, 1996.
- [28] R. Haralick and L. Shapira. The consistent labeling problem: Part 1. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 1(2):173–184, 1979.
- [29] Y. Horry, K. Anjyo, and K. Arai. Tour into the picture: Using a spidery mesh interface to make animation from a single image. *ACM Proc. SIGGRAPH'97*, pages 225–232, 1997.
- [30] D.A. Huffman. Impossible objects as nonsense sentences. *Machine Intelligence*, 6:295–323, 1971.
- [31] T. Kanade. Recovery of the three-dimensional shape of and object from a single view. *Artificial Intelligence*, 17(1-3):409–460, 1981.

- [32] H.W. Kang, S. H. Pyo, K. Anjyo, and S.Y. Shin. Tour into the picture using a vanishing line and its extension to panoramic images. *Proc. EuroGraphics*, 20(3), 2001.
- [33] J.J. Koenderink. What does the occluding contour tell us about solid shape? *Perception*, 13:321–330, 1984.
- [34] K. Koffka. *Principles of Gestalt Psychology*. Routledge & K. Paul, 1963.
- [35] M.H. Kuo. Reconstruction of quadric surface solid from three-view engineering drawings. *Computer-Aided Design*, 30(7):517–527, 1998.
- [36] A. Kushal, G. Chanda, and K. Shrivastava. Multilevel modelling and rendering of architectural scenes. *Proc. EuroGraphics*, 2003.
- [37] Y.G. Leclerc and M.A. Fischler. An optimization-based approach to the interpretation of single line drawings as 3D wire frames. *Int'l Journal of Computer Vision*, 9(2):113–136, 1992.
- [38] R. Lequette. Automatic construction of curvilinear solid from wireframe views. *Computer-Aided Design*, 20(4):171–180, 1988.
- [39] D. Liebowitz, A. Criminisi, and A. Zisserman. Creating architectural models from images. *Proc. EuroGraphics*, 18:39–50, 1999.
- [40] H. Lipson and M. Shpitalni. Optimization-based reconstruction of a 3D object from a single freehand line drawing. *Computer-Aided Design*, 28(8):651–663, 1996.
- [41] J. Liu, L. Cao, and X. Tang. Curved object reconstruction from 2D line drawings. *To be submitted to IEEE Trans. Visualization & Computer Graphics*.
- [42] J. Liu and Y.T. Lee. A graph-based method for face identification from a single 2D line drawing. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 23(10):1106–1119, 2001.
- [43] J. Liu, Y.T. Lee, and W.K. Cham. Identifying faces in a 2D line drawing representing a manifold object. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 24(12):1579–1593, 2002.
- [44] D. Lysak. Interpretation of engineering drawings of polyhedral and nonpolyhedral objects from orthographic projections. *PhD Thesis, Dept. of Electrical & Computer Engineering, The Pennsylvania State University*, 1991.
- [45] A.K. Mackworth. Interpreting pictures of polyhedral scenes. *Artificial Intelligence*, 4(2):121–139, 1973.
- [46] J. Malik. Interpreting line drawings of curved objects. *Int'l Journal of Computer Vision*, 1:73–103, 1987.
- [47] K. F. Man, K. S. Tang, and S. Kwong. *Genetic Algorithms: Concepts and Designs*. New York: Springer, 1999.
- [48] T. Marill. Emulating the human interpretation of line-drawings as three-dimensional objects. *Int'l Journal of Computer Vision*, 6(2):147–161, 1991.

- [49] D. Marr. *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information*. Henry Holt Company, 1982.
- [50] Duncan Marsh. *Applied Geometry for Computer Graphics and CAD*. Springer Undergraduate Mathematics Series. Springer, 1999.
- [51] Patrick Min, Joyce Chen, and Thomas Funkhouser. A 2D sketch interface for a 3D model search engine. *ACM SIGGRAPH'02 Technical Sketches*, page 138, 2002.
- [52] V.S. Nalwa. Line-drawing interpretation: Straight lines and conic sections. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 10(4):514–529, 1988.
- [53] T. O'Donnell, T.E. Boulton, X. Fang, and A. Gupta. The extruded generalized cylinder: A deformable model for object recovery. *IEEE Proc. Computer Vision and Pattern Recognition*, pages 174 – 181, 1994.
- [54] T. O'Donnell, M.-P. Dubuisson-Jolly, and A. Gupta. A cooperative framework for segmentation using 2-D active contours and 3-D hybrid models as applied to branching cylindrical structures. *IEEE Proc. Int'l Conf. Computer Vision*, pages 454 – 459, 1998.
- [55] T. O'Donnell, A. Gupta, and T. Boulton. A new model for the recovery of cylindrical structures from medical image data. *Joint Conf. Computer Vision, Vir. Reality and Robotics in Medicine and Robotics, and Comp.-Assisted Surgery*, 1997.
- [56] S. Ortiz. 3D searching starts to take shape. *Computers*, 37(8):24–26, 2004.
- [57] S.E. Palmer. *Vision Science: Photons to Phenomenology*. The MIT Press, 1999.
- [58] T.A. Pastva. Bezier curve fitting. *Master's Thesis, Naval Postgraduate School, Monterey, CA*, 1998.
- [59] P.J. Giblin and B.B. Kimia. Transitions of the 3D medial axis under a one-parameter family of deformations. *European Conf. Computer Vision*, pages 718–724, 2002.
- [60] J. Ponce and D. Chelberg. Finding the limbs and cusps of generalized cylinders. *Int'l Journal of Computer Vision*, 1:195–210, 1987.
- [61] J. Ponce, D.M. Chelberg, and W.B. Mann. Invariant properties of straight homogeneous generalized cylinders and their contours. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 11(9):951–966, 1989.
- [62] W.H. Press, S.A. Teukolsky, W.T. Vetterling, and B.P. Flannery. *Numerical Recipes in C++: The Art of Scientific Computing*. Cambridge University Press, 2002.
- [63] W. Puech, J.M. Chassery, and I. Pitas. Cylindrical surface localization in monocular vision. *Pattern Recognition Letter*, 18:711–722, 1997.
- [64] L. Ros and F. Thomas. Overcoming superstrictness in line drawing interpretation. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 24(4):456–466, 2002.

- [65] H. Sato and T.O. Binford. Finding and recovering SHGC objects in an edge image. *CVGIP: Graphical Model and Image Processing*, 57(3):346–358, 1993.
- [66] P. Sayd, M. Dhome, and J.M. Lavest. Recovering generalized cylinders by monocular vision. *Object Representation in Computer Vision II*, pages 25–51, 1996.
- [67] Steven Shafer and Takeo Kanade. The theory of straight homogeneous generalized cylinders and a taxonomy of generalized cylinders. *Technical Report CS-083-105, Carnegie Mellon Universit*, 1983.
- [68] U. Shani and D. H. Ballard. Splines as embeddings for generalized cylinders. *Computer Vision, Graphics, and Image Processing (CVGIP)*, 27(2):129–156, 1984.
- [69] Amit Shesh and Baoquan Chen. Smartpaper: An interactive and user friendly sketching system. *Proc. EuroGraphics*, 2004.
- [70] K. Shoji, K. Kato, and F. Toyama. 3-D interpretation of single line drawings based on entropy minimization principle. *IEEE Proc. Computer Vision and Pattern Recognition*, 2:90–95, 2001.
- [71] M. Shpitalni and H. Lipson. Identification of faces in a 2D line drawing projection of a wireframe object. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 18(10):1000–1012, 1996.
- [72] C.E Springer. *Geometry and Analysis of Projective Spaces*. San Francisco : Freeman, 1964.
- [73] K.A. Stevens. The visual interpretation of surface contours. *Artificial Intelligence*, 17(1-3):47–73, 1981.
- [74] K.A. Stevens. Implementation of a theory for inferring surface shape from contours. *MIT AI Memo-676*, 1982.
- [75] Peter Sturm and Steve Maybank. A method for interactive 3d reconstruction of piecewise planar objects from single images. *British Machine Vision Conference*, pages 265–274, 1999.
- [76] K. Sugihara. An algebraic approach to shape-from-image problem. *Artificial Intelligence*, 23:99–95, 1984.
- [77] K. Sugihara. A necessary and sufficient condition for a picture to represent a polyhedral scene. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 6(5):578–586, 1984.
- [78] T. Syeda-Mahmood. Indexing of technical line drawing databases. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 21(8):737–751, 1999.
- [79] D. Terzopoulos and D.N. Metaxas. Dynamic 3D models with local and global deformations: Deformable superquadrics. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 13(7):703–714, 1991.
- [80] D. Terzopoulos, A.P. Witkin, and M. Kass. Constraints on deformable models: Recovering 3D shape and nonrigid motion. *Artificial Intelligence*, 36(1):91–123, 1988.
- [81] A. Turner, D. Chapman, and A. Penn. Sketching space. *Computers & Graphics*, 24:869–879, 2000.

- [82] F. Ulupinar and R. Nevatia. Inferring shape from contour for curved surfaces. *IEEE Proc. Int'l Conf. Pattern Recognition*, pages 147–154, 1990.
- [83] F. Ulupinar and R. Nevatia. Shape from contour: Straight homogeneous generalized cones. *IEEE Proc. Int'l Conf. Pattern Recognition*, pages 582–586, 1990.
- [84] F. Ulupinar and R. Nevatia. Recovering shape from contour for constant cross section generalized cylinders. *IEEE Proc. Computer Vision and Pattern Recognition*, pages 674–676, 1991.
- [85] F. Ulupinar and R. Nevatia. Perception of 3-D surfaces from 2-D contours. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 15(1):3–18, 1993.
- [86] F. Ulupinar and R. Nevatia. Recovery of 3-D objects with multiple curved surfaces from 2-D contours. *Artificial Intelligence*, 67(1):1–28, 1994.
- [87] F. Ulupinar and R. Nevatia. Shape from contour: Straight homogeneous generalized cylinders and constant cross-section generalized cylinders. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 17(2):120–135, 1995.
- [88] P. A. C. Varley and R. R. Martin. A system for constructing boundary representation solid models from a two-dimensional sketch. *IEEE Proc. Geometric Modeling and Processing*, pages 13–30, 2000.
- [89] A.P. Vicent, P.C. Calleja, and R.R. Martin. Skewed mirror symmetry in the 3d reconstruction of polyhedral models. *Journal of WSCG*, 11(3):504–511, 2003.
- [90] D. Waltz. *Understanding Line Drawings of Scenes with Shadows*, chapter Psychology of Computer Vision, pages 19–91. New York: McGraw-Hill, 1975.
- [91] D.B. West. *Introduction to Graph Theory*. Prentice Hall, 1996.
- [92] M. Yu, I. Atmosukarto, W. K. Leow, Z. Huang, and R. Xu. 3D model retrieval with morphing-based geometric and topological feature maps. *IEEE Proc. Computer Vision and Pattern Recognition*, 2:656–661, 2003.
- [93] M. Zerroug and R. Nevatia. Quasi-invariant properties and 3-D shape recovery of non-straight, non-constant generalized cylinders. *IEEE Proc. Computer Vision and Pattern Recognition*, pages 96–103, 1993.
- [94] M. Zerroug and R. Nevatia. Segmentation and recovery of SHGCs from a real intensity image. *European Conf. Computer Vision*, pages 319–330, 1994.
- [95] M. Zerroug and R. Nevatia. Three-dimensional descriptions based on the analysis of the invariant and quasi-invariant properties of some curved-axis generalized cylinders. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 18(3):237–253, 1996.
- [96] Q. Zhu. Virtual edges, viewing faces, and boundary traversal in line drawing representation of objects with curved surfaces. *Int'l Journal of Computers and Graphics*, 15(2):161–173, 1991.
- [97] S.C. Zhu. Embedding gestalt laws in markov random fields. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 21(11):1170–1187, 1999.

CUHK Libraries



004280669