Title:
Learnable CAD for Reconstructing 3D Models from Sketches

Authors:
Masaji Tanaka, tanaka@mech.ous.ac.jp, Okayama University of Science
Ryosuke Yamaoka, t13t101yr@ous.jp, Okayama University of Science

Keywords:
Inductive Learning, Sketch, Line Drawing, Reconstruction, Learning Interface

DOI: 10.14733/cadconfP.2017.420-424

Introduction:
Generally, sketches as line drawings are important tools for designers when they create new mechanical parts and so forth. A human can draw a sketch of a 3D object, and the other human would recognize the object from the sketch. This human behavior fascinated a lot of researchers who long for realizing artificial intelligence systems. If a system that can automatically reconstruct a 3D object as a solid model from a sketch drawn in CAD is realized, it would be useful for designers and so forth. Since decades, this reconstruction system has been researched by many researchers, e.g. [1]. Although their proposed methods could automatically reconstruct solid models from sketches, it was difficult for them to handle sketches including curved lines. Recently we proposed a method that could handle curved lines in sketches drawn in CAD, e.g. [6]. In [6], firstly several sketches of simple objects such as a cuboid and a hole (see Fig. 1) were defined as sketch features, and when a sketch of a complex object was input, each sketch feature was detected and extracted repeatedly from the sketch until all lines were deleted in the sketch. After a feature was extracted from a sketch, the sketch would be broken and meaningless. So some restoration process would be required. Also, since the restoration process could exist unlimitedly for many kinds of sketches, we applied our inductive learning technique, e.g. [7]. In the technique, each restoration process could be learned precisely and mathematically. Although deep learning systems in neural networks have been developed magnificently in recent years, e.g. [5], it is not clear how a deep learning method constructs a procedure out of example sequences of steps unless the method produces each step out of example steps and some other method puts the steps into a sequence of the steps. Our learning technique can construct a procedure from example sequences of steps. In [7], although the effectiveness of the learning technique for [6] could be explained in detail, the organization of the user interface for the learning was difficult and unclear. In this paper, we propose a new user interface for the learning of those restoration processes by using simple CAD operations such as copy & paste, and also propose more simplified and clarified algorithm than our past methods [6-7].

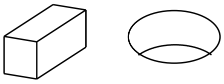


Fig. 1: Two sketch features, a cuboid and a hole.

Main Idea:
Fig. 2 shows our proposed algorithm in this paper. The algorithm is made from integrating and simplifying our past methods [6-7]. In Step 9 of that, a broken sketch could be restored automatically or some user could teach restoration process in CAD. In this teaching, a user can only use simple CAD operations. Therefore, our proposed user interface would be much more effective than [7]. In our past methods, we defined classes only of basic geometric elements such as points and lines. Therefore, the

introduction of our IFOG technique [8] to our past methods became uncompleted. In our proposed algorithm, much more classes which could cover many kinds of restoration processes are defined. Here, to solve Example 1 illustrated in Fig. 3(a), eight classes are defined as follows.

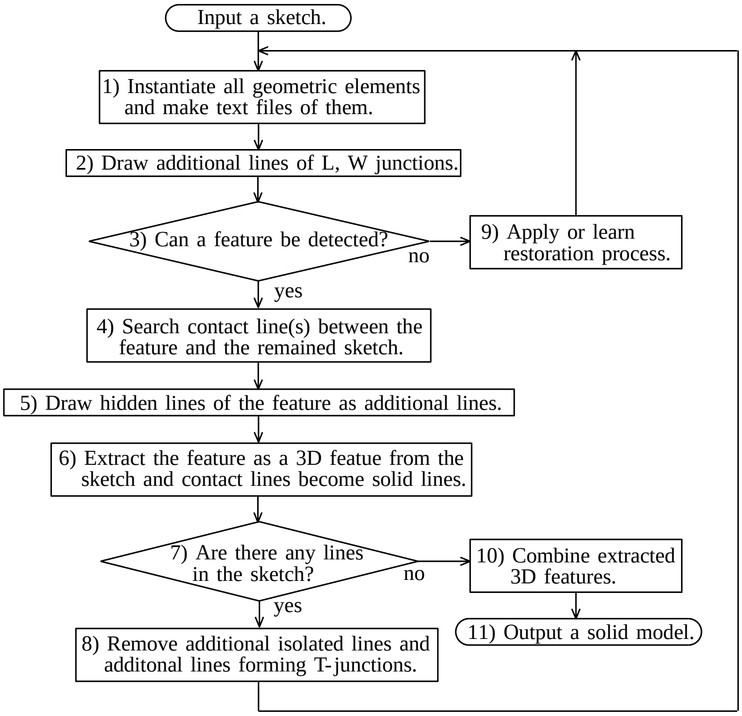


Fig. 2: Our proposed algorithm in this paper.

Class Point: 1) Number 2) x-y coordinates 3) Contact lines;

Class Straight Line: 1) Number 2) Two terminals 3) additional line? 4) Contact line of two features?;

Class Elliptical Arc: 1) Number 2) x-y coordinates of center point 3) Major and Minor radiuses 4) Initial and terminal points (counterclock) 5) Slope of major axis (deg) 6) Ellipse? 7) Semi ellipse? 8) additional line? 9) Contact line of two features?;

Class Rectangle: 1) Number 2) Two pairs of two lines which are parallel 3) Four points;

Class Relationship of Two Straight Lines: 1) Number 2) Two lines 3) Connection point 4) Parallel?;

Class Relationship of Two Elliptical Arcs: 1) Number 2) Two elliptical arcs 3) Connection point(s) 4) Same shape? 5) Do they form T-junction(s)? 6) Which arc has some terminal in T-junction(s)?;

Class Cuboid Sketch: 1) Number 2) Three parallelograms 3) Three lines forming T-junction 4) Connection point of T-junction 5) Contact line(s) to the other feature(s);

Class Hole Sketch: 1) Number 2) Ellipse 3) Elliptical arc in 2) 4) Two connection points 5) Same shape?;

Each class consists of properties. For example, Class Point consists of three properties. For solving Example 1, only cuboids and holes are defined as two classes here. When Example 1 is input to the algorithm, firstly each geometric element is recognized and instantiated from its class, and then instance files as text files are made or updated in Step 1. If two straight lines are intersected, they are divided at their intersection but curved lines are not divided in any cases. In Step 2, L-junctions and W-junctions are detected, and their both sides lines are extended as additional lines. They are drawn as dotted lines. In Fig. 3(b), four additional lines are drawn from an L-junction and a W-junction. In Step 3, a hole sketch can be detected as in this figure. The hole consists of P_1, P_2, L_1 and L_2. They can be defined as instances as follows.

Point: 1) P1 2) 18.5 39.2 3) L1 L2; 1) P2 2) 38.3 23.8 3) L1 L2;

Elliptical Arc: 1) L1 2) 25.0 28.5 3) 25.0 20.2 4) N/A 5) -14 6) yes 7) no 8) no 9) N/A;

1) L2 2) 30.0 32.5 3) 25.0 20.2 4) P1 P2 5) -14 6) no 7) yes 8) no 9) N/A;

Relationship of Two Elliptical Arcs: 1) R1 2) L1 L2 3) P1 P2 4) yes 5) yes 6) L2;

Hole Sketch: 1) F1 2) L1 3) L2 4) P1 P2 5) yes;

In Step 4, L_1 can be the contact line between the hole and the remained sketch. In Step 5, one elliptical arc and two straight lines can be drawn as additional lines as in Fig. 3(c). In Step 6, all lines of the hole (F_1) are deleted as in Fig. 3(d). In this figure, to emphasize the contact line, L_1 is drawn as a dotted circle in the remained sketch. In Step 8, four straight additional lines are deleted but they are drawn again in Step 2 as in Fig. 3(e). In this figure, a cuboid sketch can be detected in Step 3, and since L_3 is an additional line made from the L-junction and shares the cuboid and the remained sketch, L_3 becomes the contact line in Step 4. In Step 5, three hidden lines are drawn as in Fig. 3(f). In Step 6, all solid lines of the cuboid are deleted, and L_3 is changed into a solid line as in Fig. 3(g). In Step 8, L_7, L_8 and L_9 are deleted because they form T-junctions. Also, L_4 and L_5 are deleted because they are additional isolated lines. An isolated line has some terminal connecting to no lines in the algorithm. The detailed explanations of extracting features are indicated in [6]. After this deletion, L_6 becomes an additional isolated line and it is deleted. As the result, Fig. 3(h) is obtained. In this figure, it is clear that L_3 is shared between the cuboid (F_2) and the remained sketch. Therefore, if M_1 and M_2 are two 3D faces, they are connected tangentially and L_3 is disappeared in the 3D model of Example 1. The data of L_3 in Fig. 4(d) becomes as follows. **Straight Line**: 1) L3 2) (omitted) 3) no 4) yes;

Here, suppose that a hole sketch is detected in Step 9) as in Fig. 4(a). The detailed explanation of this step is indicated later. In Fig. 4(b), the hole (F_3) is extracted, and F_4 could be detected in Step 9). Fig. 4(c) shows the sketch after the extraction of F_4. In this figure, there are two parallelograms. If a cuboid sketch can be restored as in Fig. 4(d) from these parallelograms, Step 10 can be executed. When two 3D features are combined step by step (F_5+F_4+F_3+F_2+F_1) in accordance with their contact lines, finally the solution of Example 1 can be obtained as in Fig. 4(e).

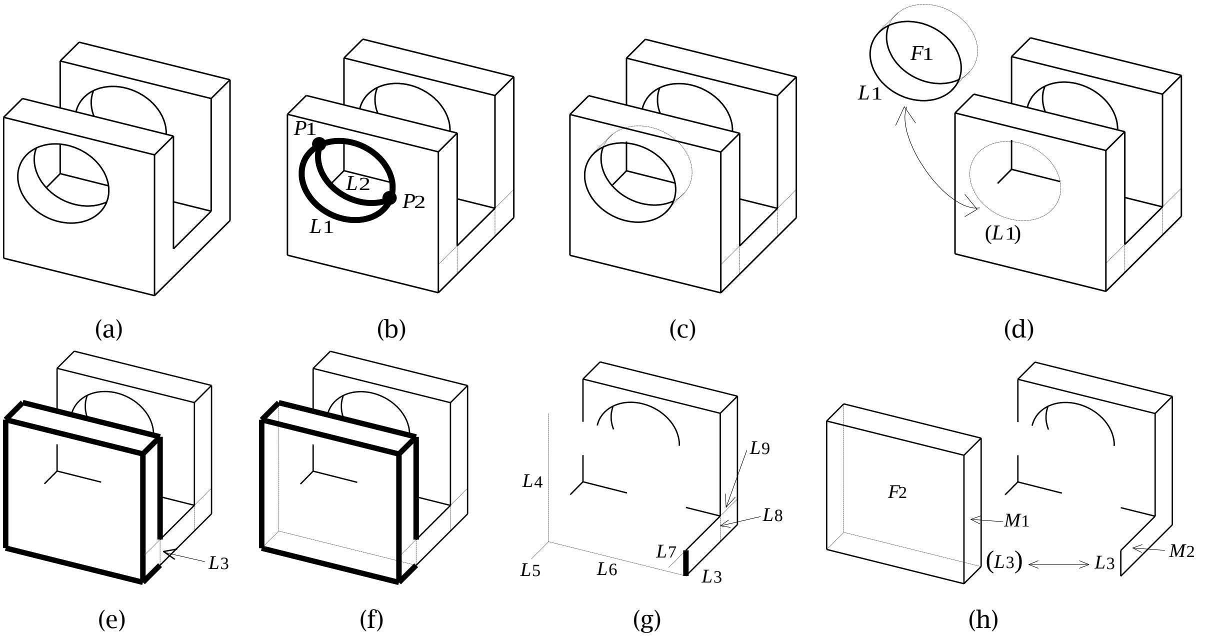
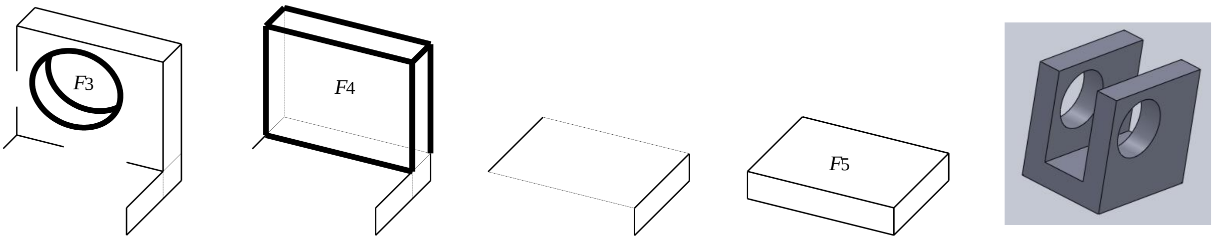


Fig. 3: Extraction of two features from Example 1: (a) Example 1, (b) Four additional lines and the detection of a hole sketch, (c) Hidden lines of the hole, (d) Extraction of the hole as F_1. (e) Detection of a cuboid sketch, (f) Hidden lines of the cuboid, (g) Extraction of the cuboid as F_2, and (h) Relationship among M_1, M_2 and L_3.



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

Fig. 4: Making solid model of Example 1 from Fig. 3: (a) Detection of F_3, (b) Detection of F_4, (c) Extraction of F_4, (d) Detection of F_5, and (e) An overview of the solution.

Since many kinds of learning processes to restore parallelograms were indicated in detail in [4], in this paper, the learning of restoring hole sketches and a difficult restoration case of a cuboid sketch are indicated as follows. Fig. 5(a) shows Problem 1 consisting of two elliptical arcs. This problem can be expressed as their properties as follows. Here, useless properties are omitted.

Problem 1: **L1** 4) P2 P4 6) no 7) yes; **L2** 4) P3 P1 6) no 7) yes; **R1** 2) L1 L2 3) P1 4) yes 5) yes 6) L2;

When a user extends *L*1 to make an ellipse as in Fig. 5(b), *P*2 and *P*4 are removed. Then the user can extend *L*2 to make a hole sketch as in Fig. 5(c). As the result, the above properties can be changed as Solution 1 as follows.

Solution 1: **L1** 4) N/A 6) yes 7) no; **L2** 4) P3 P1 6) no 7) yes; **R1** 2) L1 L2 3) P1 P3 4) yes 5) yes 6) L2;

In the same way, the user can teaches Problem 2 (Fig. 5(d)) which is very different from Problem 1 and Solution 2 (Fig. 5(e)) as follows.

Problem 2: **L3** 4) P6 P8 6) no 7) yes; **L4** 4) P7 P5 6) no 7) yes; **R2** 2) L3 L4 3) P8 4) yes 5) yes 6) L3;

Solution 2: **L3** 4) P6 P8 6) no 7) yes; **L4** 4) N/A 6) yes 7) no; **R2** 2) L3 L4 3) P8 P6 4) yes 5) yes 6) L3;

When these two problems are compared, if two numbers of the same property are different, a variable can be made such as *L*x1. The numbering process of variables is executed step by step. As the result, a generalized problem and a generalized solution can be made step by step as follows.

Generalized problem:

Lx1 4) Px1 Px2 6) no 7) yes; **Lx2** 4) Px3 Px4 6) no 7) yes; **Rx1** 2) Lx1 Lx2 3) Px1 4) yes 5) yes 6) Lx2;

Generalized solution:

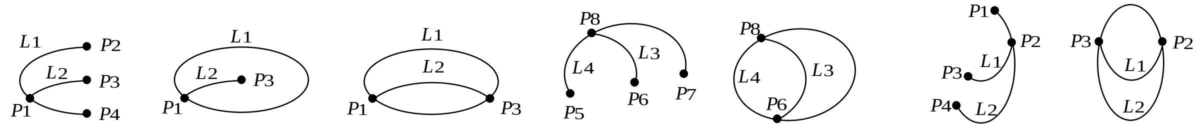
Lx1 4) N/A 6) yes 7) no; **Lx2** 4) Px1 Px2 6) no 7) yes; **Rx1** 2) Lx1 Lx2 3) Px1 Px2 4) yes 5) yes 6) Lx2;

Fig. 5(f) is a new problem. When the generalization is applied, Problem 3 and its solution (Fig. 5(g)) can be obtained as follows.

Problem 3: **L2** 4) P4 P1 6) no 7) yes; **L1** 4) P3 P2 6) no 7) yes; **R3** 2) L1 L2 3) P2 4) yes 5) yes 6) L1;

Solution 3: **L2** 4) N/A 6) yes 7) no; **L1** 4) P2 P3 6) no 7) yes; **R3** 2) L1 L2 3) P2 P3 4) yes 5) yes 6) L1;

In the same way, to restore Fig. 4(c), a user can teach the following three operations. 1) Copy & paste an oblique line. 2) Copy & paste a vertical line. 3) Change all additional lines into solid lines.

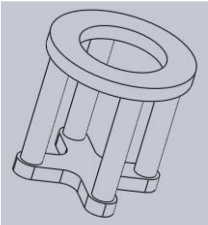


(a) (b) (c) (d) (e) (f) (g)

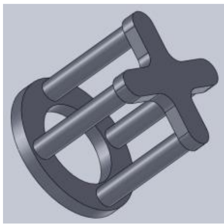
Fig. 5: Learning of restoring hole sketches: (a) Problem 1, (b) Extension of L_1 in (a), (c) Solution 1, (d) Problem 2, (e) Solution 2, (f) New problem as Problem 3, and (g) Solution 3.

Conclusions:

In this paper, firstly our new algorithm is indicated. In the algorithm, the handling of additional lines and contact line(s) between two 3D features could be explained more clearly than our past methods [6-7]. Also, since all geometric entities are classified in the algorithm, clearer learning could be realized especially in elliptical arcs. As the result, a user could teach restoration processes to make sketch features by using simple CAD operations. Therefore, our automatic restoration system might realize a learnable CAD including a smart user interface. Finally the vision of our method is described as follows. Recently many 3D sketching systems has been developed. For example, SOLIDWORKS has operations for that, e.g. [2]. SketchUp is a product to make 3D models from sketches, e.g. [3]. Igarashi has developed 'SmoothTeddy' that is a quick 3D modeling and painting system, e.g. [4]. Although they use sketches to make 3D models, they seem to be particular solid modelers or CG systems. Their systems have automatic parts to aid 3D modeling from sketches. However, it would be difficult for users to select the best system among such systems. As a result, our system might be more useful than those systems in the future. The ability of our system would be shown in Fig. 6. This figure illustrates Example 2 that is a mechanical part. To solve Example 2, 'Fillet' must be defined as a sketch feature. The handling of fillets was described in [6]. It would be difficult to recognize such mechanical parts by conventional image retrieval techniques because most mechanical parts are not standardized. Moreover, generally the shapes of creational parts would be strange. Therefore, our system would be effective for that.



(a)



(b)

Fig. 6: Example 2: (a) Example 2 and (b) An overview of the solid model of Example 2.

References:

[1] Company, P.; A. Piquer, A.; Contero, M.; Naya, F.: A survey on geometrical reconstruction as a core technology to sketch-based modeling, Computers & Graphics, 29(6), 2005, 892–904, http://dx.doi.org/10.1016/j.cag.2005.09.007.
[2] http://help.solidworks.com/2016/English/SolidWorks/sldworks/t_Beginning_a_3D_Sketch.htm
[3] http://www.sketchup.com/
[4] Igarashi, T.; Shono, N.; Kin, T.; Saito, T.: Interactive Volume Segmentation with Threshold Field Painting, Proceedings of the 29th Annual Symposium on User Interface Software and Technology (UIST '16), 2016, 403-413, https://doi.org/10.1145/2984511.2984537
[5] Schmidhuber, J.: Deep learning in neural networks: An overview, Neural Networks, 61, 2015, 85-117, http://dx.doi.org/10.1016/j.neunet.2014.09.003.
[6] Tanaka, M.; Kaneeda, T.: Feature Extraction from Sketches of Objects, Computer-Aided Design & Applications, 12(3), 2014, 300-309, http://dx.doi.org/10.1080/16864360.2014.981459
[7] Tanaka, M.; Takamiya, Y.; Tsubota, N.; Asanuma, S.; Iwama, K.: Reconstruction of Solid Models from Sketches Including Curves with Inductive Learning Technique, Computer-Aided Design & Applications, 14(5), 2017, 632-641, http://dx.doi.org/10.1080/16864360.2016.1273580.
[8] Tanaka, M.; Takamiya, Y.; Tsubota, N.; Iwama, K: IFOG: Inductive Functional Programming for Geometric Processing, Computer-Aided Design & Applications, 13(3), 2015, 389-396, http://dx.doi.org/10.1080/16864360.2015.1114397.