Hindawi
Mathematical Problems in Engineering
Volume 2020, Article ID 7439341, 13 pages
https://doi.org/10.1155/2020/7439341

Research Article

Multistroke Grouping of Online Freehand Axonometric Sketches for Mechanical Models

Shouxia Wang, Shuxia Wang, and Weiping He

School of Mechanical Engineering, Northwestern Polytechnical University, Xi'an, China

Correspondence should be addressed to Shuxia Wang; 2008wangshuxia@163.com and Weiping He; weiping@nwpu.edu.cn

Received 19 December 2019; Revised 23 March 2020; Accepted 13 April 2020; Published 7 May 2020

Academic Editor: Paolo Crippa

Multistroke drawing occurs frequently in conceptual design sketches; however, it is almost unsupported by the current sketch-based user interfaces. We proposed a sketch recognition system based on the multistroke primitive grouping method. Based on grouping the strokes that lie within the mutual boundaries between adjacent regions, we create line drawings from online freehand axonometric sketches of mechanical models. First, closed regions and their boundary bands of the sketch were extracted. Then, the strokes that cross the boundary bands of two or more closed regions are segmented, and the strokes that lie within the intersection of two adjacent boundary bands are grouped. Finally, grouped strokes are simplified into a new single stroke and then fitted as a geometric primitive; thus, the input sketches are recognized to the line drawings. We developed a prototype of the sketch recognition system to evaluate the proposed method. The results showed that the input sketches are simplified into the accurate line drawings efficiently. The proposed method can be applied to both multistroke overtracing and nonovertracing sketches.

1. Introduction

Conceptual design is an early phase of the design process in which freehand sketching is used to capture and communicate ideas quickly. These sketches are expected to be drawn using a computer-aided design (CAD) tool to improve their utilization later in the detailed design phase. However, current CAD tools are not well suited for drawing freehand sketches; thus, the conceptual sketches would have to be manually transformed into 3D models. In recent years, many researches on sketch-based interfaces for modelling (SBIM) have been extensively carried out to extend CAD technologies to support the complete design process. Sketch-based modelling generally starts with axonometric drawings, that is, a kind of stereoscopic graph made by 2D lines. SBIM systems may obtain sketches drawn with pencil on paper or digital tools, namely, offline or online sketches, respectively. In online sketching, the locations of vertices, corners, and edges of the model can be determined while the stroke is being drawn [1]. Unlike offline sketching, stroke points are unordered and blended. Therefore, current researches mainly focus on online sketching technology [2].

Converting rough freehand sketches into clean line drawings is a crucial step in the development of both online and offline sketch-based modelling [3]. Primitive-based methods are commonly used to convert online freehand sketches to clean line drawings, which involve sketch segmentation [4, 5], stroke grouping [6, 7], primitive fitting [8, 9], and endpoint fusion and sketch beautification [10].

Multistroke sketches are commonly found during the sketch design stage when the designers tend to use several overlapped or disconnected strokes to draw a longer curve. Sometimes, additional strokes are drawn over the completed sketches to correct them. The interpretation of such sketches requires the grouping of strokes that form a single curve. A sketch-based interface for modelling should support multistroke sketching to provide designers with greater freedom. Although recognition of intended shapes from many discontinuous, overtraced, or unordered strokes can be intuitively achieved by human vision, it continues to be a challenge for computer vision. Multistroke sketching ability is one of the main challenges in online sketch recognition.

Stroke grouping is a critical step in both online sketch recognition and artistic sketch simplification, and it has

attracted many researchers' attention over the past few years. Gestalt psychologists [11] introduced the laws of perceptual organization to explain the way people make sense of visual scenes, which include proximity, similarity, closure, continuation, and common fate. The work of Liu et al. [12] is the first attempt to use the law of closure for sketch simplification by grouping strokes that lie in the same region boundary. Sketches in different drawing domains may contain different line types; for example, technical drawings are usually composed of geometric primitives (straight lines and circular arcs), while fashion design sketches and cartoon images are commonly composed of freeform splines. Therefore, strokes could be grouped together by utilizing domain-specific knowledge.

In this paper, we proposed a multistroke primitive grouping method based on mutual boundaries of adjacent regions formed by strokes to convert online axonometric sketches of mechanical models into clean line drawings. First, closed regions formed by input strokes with their boundary bands were obtained. Input strokes across two or more closed regions were splitted. Then, strokes that lied in mutual boundaries of two adjacent regions were clustered and simplified into single strokes. Finally, adjacent strokes that form an effective edge of the model were merged and fitted into a geometric. An online freehand sketch recognition system for multistroke sketches (FSR-MG) is developed to evaluate the feasibility of the proposed method. The work presented in this paper forms a foundation for future research on geometrical reconstruction.

Our main contribution is a novel algorithm that uses mutual boundaries of adjacent regions to group strokes in online multistroke axonometric sketches of mechanical models. The advantages of the proposed method are given as follows: (1) More strokes are grouped simultaneously compared to the existing methods that merely group two strokes iteratively. (2) It does not limit the overlapping ratio between strokes and works well for both multiple overtracing sketches and nonovertracing sketches. (3) It does not limit the geometric types of the grouped strokes, thus providing users with greater drawing freedom and producing a highly robust sketching system.

The rest of this article is organized as follows. In Section 2, we review the related work. Section 3 gives the overview and the details of the proposed method. Section 4 reports the implementation of the proposed method and discusses its merits and limitations. Finally, Section 5 concludes this article.

2. Related Work

Sketch-based interfaces are emerging as a way to release users from a maze of menus, toolbars, and many complicated commands [13]. Many research studies have investigated the use of sketch-based interfaces, but strict restrictions on the drawing manner do not allow users to sketch naturally. Most of the recognizers assume that any primitive shapes are drawn with a single stroke [14, 15]. However, there are many instances in which users choose to draw a shape with more than one stroke [16].

There are a few sketch-based reconstruction systems that support overtracing and multistroke. Some sketch-based systems like 3D Sketch [17] and ILoveSketch [18] approximated overlapping strokes as a core curve, but there was no prescription offered about how to group overlapping strokes. SMARTPAPER [19] simplified sketches composed of straight line segments by replacing strokes sharing similar slopes and close endpoints with an average straight line segment. Ku et al. [6] and our previous research [7] used a similar procedure that classified input strokes into 2D geometric primitives first and then grouped strokes based on their fitted features. Consequently, geometric primitives drawn with different stroke types cannot be clustered, which is a common event in the users' design sketches. Moreover, the curve grouping method [6] may result in overgrouping cases because curves would be grouped together if there was an overlap of the bounding boxes of strokes.

Many research studies have focused on the simplification of overtraced lines in artistic vector, where lines are clustered based on their degree of proximity, shape similarity, continuity, or directional consistency. Barla et al. [20] used proximity as the main constraint to group overtraced strokes. The method is based on an " -line" that contains a user-specified width to define the strokes to be grouped. However, the " -line" cannot fold onto itself; hence, it is not applicable for folding or self-intersecting curves. Pusch et al. [21] used hierarchical space partitioning to divide strokes into locally orientable segments and fitted a B-spline curve that passes through these segments. The method is suitable for nonintersecting curves. Shesh and Chen [22] merged strokes according to their proximity, color, local gradient, and the extent of overlap. Their overlapping concept might be inadequate because of the lack of overlap between the contour curves. Chien et al. [23] used a low-pass filter to assign a weight to every stroke, moved the strokes to the position of the higher weight, and paired strokes based on the endpoints' position and their tangent value. However, the method cannot simplify the strokes locally.

Some line drawing simplification methods used the regions formed by strokes as guidance for stroke clustering. Liu et al. [12] made the first attempt to incorporate the law of closure into the semantic analysis of sketches. Based on the notion of regions formed by strokes, the method combines stroke grouping with region interpretation and the proposed iterative cyclic refinement approach and solves the problem of the mutual influence of regions and strokes. However, it fails to explicitly handle the closed strokes. Liu et al. [24] simplified a vector sketch by simplifying its geometric structure, which is extracted from the input vector graph and consists of various geometric primitives like nodes, curves, and patches. The method extracts regions from sketch by nodes of interacting curves rather than from raster images.

Some studies used machine learning algorithms to simplify line drawings. Orbay and Kara [25] proposed a trainable stroke clustering method that learns the rules for stroke grouping from the users' sketches. The method can handle self-intersecting and bifurcating curves that are difficult to distinguish using a purely local analysis. Ogawa et al. [26] proposed a machine learning approach that can be

trained without the use of manual annotations. The difficulty in the application of machine learning is the acquisition of training samples.

Most of the artistic line simplification methods mentioned above cannot directly solve long tortuous strokes because they were designed to calculate the geometric relationships between short, smooth strokes. In addition, the stroke segmentation process will break the strokes' raw information, such as closure and geometric types, while the stroke clustering process might fail to restore such original information. Therefore, they are not fully applicable to multistroke primitive grouping in mechanical drawings where the contour lines are mainly made up of regular curves.

Many approaches have been proposed to extract simplified drawings from scanned pencil sketches [1, 27–36]. Kara et al. [28] proposed a sketch recognizer that allows overtracing symbol input, but it relies entirely on the symbol libraries where the sketch is examined in pixels; hence, no work has been done on the interpretation of individual strokes. Sezgin and Davis [27] applied moving least squares to move a circular window along the overtraced strokes of non-self-intersecting polylines, obtaining several small segments before linking to form a single line. Chansri and Koomsap [1] created line drawings from paper-based overtracing freehand sketches by image processing techniques such as dilation and thinning. Bartolo et al. [29, 30] described approaches based on Gabor and Kalman filtering to convert rough strokes into vectorized representations. Simo Serra et al. [32, 33] proposed approaches based on Convolutional Neural Networks to directly simplify rough raster sketches, using their dataset for large-scale learning of sketch simplification. Favreau et al. [31] proposed the first vectorization algorithm that explicitly balances the fidelity of the input bitmap with the simplicity of the output, as measured by the number of curves and their degree. Chen et al. [34] proposed an improved topology extraction approach for vectorization of sketches. It has been proven to be efficient and robust, but high-level understanding of the line drawing was needed. Parakkat et al. [35] used the Delaunay triangulation to group adjacent strokes; the method can handle discontinuous and broken curves. Donati et al. [36] used Pearson's multiresolution cross correlation application to extract lines form noisy hand-drawn sketches and then used an unbiased thinning algorithm to obtain clean 1-pixel lines. These approaches build directly upon the overtraced strokes for line drawing creation because they are blended together "for free" as the user draws [37].

3. Freehand Sketch Recognition System for Multistroke Sketches

In hand-drawn axonometric drawings of mechanical models, edges and profile outlines are represented by a cluster of strokes. The edge is the mutual boundary between two adjacent faces and the profile outline is the mutual boundary between visible and invisible regions of a curved surface.

We focus on online freehand axonometric projections of mechanical models with hidden lines. We assume that the hidden lines are drawn with solid lines. This kind of drawings has two important characteristics: (1) the edges and profile outlines are mostly composed of line segments or curves, and (2) the boundary projections of two adjacent regions mostly contain only one geometric primitive, such as line segments, polylines, and conic curves. For example, the axonometric drawing shown in Figure 1 has 5 closed regions, and region 6 is the background area. In Figure 1, the arc segments AG, CD (upper arc), and AB are the mutual boundaries of the adjacent regions (1, 2), (2, 3), and (1, 6), respectively. These three segments are circular arcs in 3D space but are projected as ellipse arcs in axonometric drawing. We call these intersection points as branch points, which could be the intersection of two edge lines, two profile outlines, or an edge line and a profile outline. For example, points A to G shown in Figure 1 are branch points. Therefore, the mutual boundaries of two adjacent regions end at two corresponding branch points.

3.1. General Idea. The general idea of the proposed multistroke primitive grouping method based on mutual boundaries between adjacent regions is illustrated in Figure 2. We assume that all valid closed regions (Figure 2(b)) of the sketch (Figure 2(a)) are extracted. First, input strokes are segmented into shorter strokes at the branch points. Next, strokes inside the same mutual boundaries of two adjacent regions are grouped, as shown in different colors in Figure 2(c). Then, grouped strokes are simplified into single strokes (Figure 2(d)). Finally, adjacent strokes that form a single primitive are combined to a new stroke (Figure 2(e)) and are fitted into a parameterized primitive to get a 2D line drawing. We adopted the trapped-ball method [38] to extract the closed regions of sketches and used the fuzzy classification method to fit single strokes to geometric primitives, as proposed in our previous work [39].

3.2. Multistroke Grouping. The input of our online multistroke sketch recognition system is a set of digital strokes drawn by the user through a tablet stylus or a mouse. A stroke is defined as a time-ordered sequence of 2D points that are sampled along the trajectory of the stylus. The flowchart of our system is presented in Figure 3. The common regional boundaries based multistroke grouping method consists of six procedures as follows. The former steps are depicted in detail in sections 3.2.1–3.2.4:

(1) Closed regions of the sketches are extracted by using the trapped-ball method. Then the mathematical morphology operations, such as dilation and erosion, are used to obtain a ring-shaped boundary band for each region.

(2) All cross-regional strokes are segmented into shorter strokes at two adjacent branch points and a copy of the original transregional strokes is saved before segmentation.

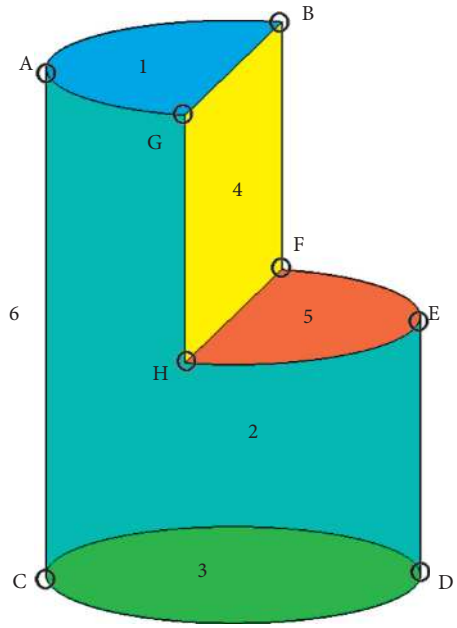


Figure 1: A sample of axonometric drawing.

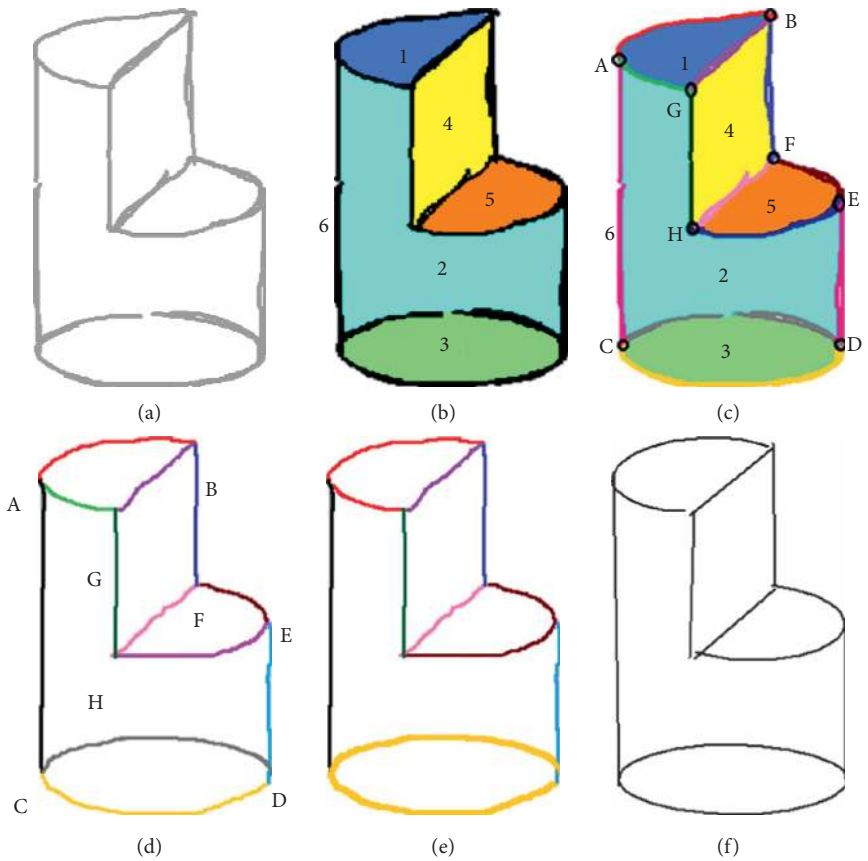


Figure 2: General idea of the multistroke grouping method based on common boundaries of regions: (a) input sketch; (b) sketch regions; (c) stroke segmentation and grouping; (d) multiple strokes simplification; (e) adjacent strokes combination; (f) single-stroke fitting.

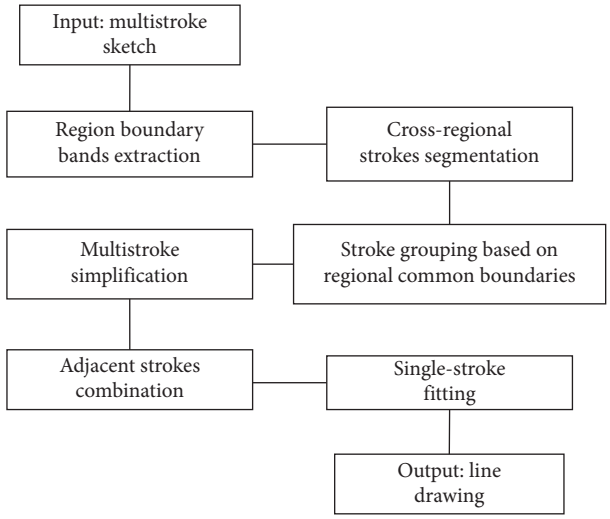


Figure 3: Flowchart of line drawing creation from multistroke freehand sketching in FSR-MG system.

(3) The strokes that lie in the intersection area of the boundary bands of two adjacent regions are grouped together.

(4) Grouped multiple strokes are simplified into single strokes.

(5) The adjacent strokes that form a single primitive are combined as new single strokes. At this point, a single stroke sketch is obtained.

(6) Each single stroke is fitted to a geometric primitive using our previous fuzzy classification method [39].

3.2.1. Region Boundary Bands Extraction. The commonly used flood-fill method might cause a leaking problem when the sketch has large interstroke gaps. We adopt the trapped-ball region segmentation method [38] to extract the regions of multistroke sketches. The method is a leak-proof region extraction to process images with discontinuous edges. We manually adjust the maximal trapped-ball radius (from 4 to 8 pixels) until the optimal closed regions of the sketch are extracted.

A region boundary band is a toroidal zone around the outside boundary of a closed region or the overall outline of the entire sketch, as shown in Figure 4. Given a closed region from the original sketch image, two morphology operations, dilation and erosion, are performed by using a circular structure element. Then the region boundary band is obtained by subtracting the eroded region from the dilated region. The width of the boundary band of a region or the whole sketch l_r is positively correlated to the scale of the region, as shown in the following equation:

$$l_r = \alpha\sqrt{m_r}, \tag{1}$$

where α is an experienced coefficient, set to 1.2 and 0.95 for the boundary bands of the closed regions and the whole sketch. The parameter m_r is the length of the short semiaxis radius of the region's minimum enclosing ellipse. The region boundary band is represented by its constituent pixels in the common regional boundaries based multistroke grouping method, namely, $\mathbf{B} = \{p_j = (x_j, y_j)\}$.

3.2.2. Cross-Regional Strokes Segmentation. A cross-regional stroke is defined as a stroke that is partially within any region boundary band, as shown in Figure 5. It is divided into multiple stroke segments, where each of them is mostly located inside the common boundary of two adjacent regions and between two adjacent branch points. If more than 90% of the sampling points of a stroke are inside the boundary band of a region R, then the stroke is thought to belong to that region; otherwise, the stroke needs to be segmented further. An original input stroke is saved in a predefined linked strokes list $\{\mathbf{Sc}_j\}$ before splitting, which is used to merge adjacent strokes to avoid losing joint information of the original input strokes.

Given an original input stroke list $\mathbf{S} = \{p_i = (x_i, y_i, t_i); 0 \le i < n\}$, and a region boundary band or a whole sketch boundary band $\mathbf{B} = \{p_j = (x_j, y_j); 0 \le j < m\}$, where n is the number of the sampling points of $\mathbf{S}$. The method that segments the stroke $\mathbf{S}$ according to the region boundary band $\mathbf{B}$ is as follows.

(i) *Step 1*. Count the number of sampling points of $\mathbf{S}$ that lie in the region boundary band $\mathbf{B}$, and denote it as n_f.

(ii) *Step 2*. Calculate the ratio of n_f and n; if $n_f/n \le 0.9$, proceed to Step 7.

(iii) *Step 3*. In turn, judge whether the point p_i in $\mathbf{S}$ is inside the region boundary band $\mathbf{B}$; and remove the successive sampling points that are all inside (or not inside) the region boundary band $\mathbf{B}$ from $\mathbf{S}$ to establish a new stroke; thus, $\mathbf{S}$ is divided into several stroke segments.

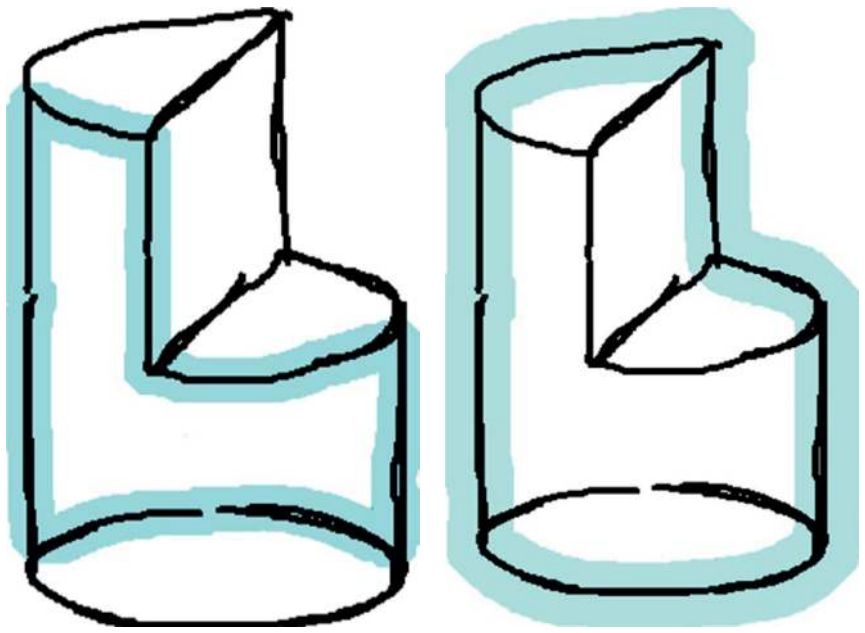
(iv) *Step 4*. Mark the first point of each segment as well as the last point of $\mathbf{S}$ as "initial split points" and save them in the point list $\{\mathbf{si}_k\}$, as shown in Figure 5(a).

(v) *Step 5*. Along the drawing direction of $\mathbf{S}$, some initial split points indicate the trajectory in which $\mathbf{S}$ starts entering or leaving the boundary band $\mathbf{B}$, and they are called entry points or exit points, respectively, such as $\mathbf{si}_2$ and $\mathbf{si}_3$ in Figure 5(a) separately. Assume that lr is the width of $\mathbf{B}$; along the stroke direction, we move the entry points forward $lr/2$ pixels and move the exit points back $lr/2$ pixels to obtain real split points, which are saved in a point list $\{\mathbf{sr}_k\}$, as shown in Figure 5(b).

(vi) *Step 6*. Divide $\mathbf{S}$ into several shorter strokes at the real split points $\mathbf{sr}_k$.

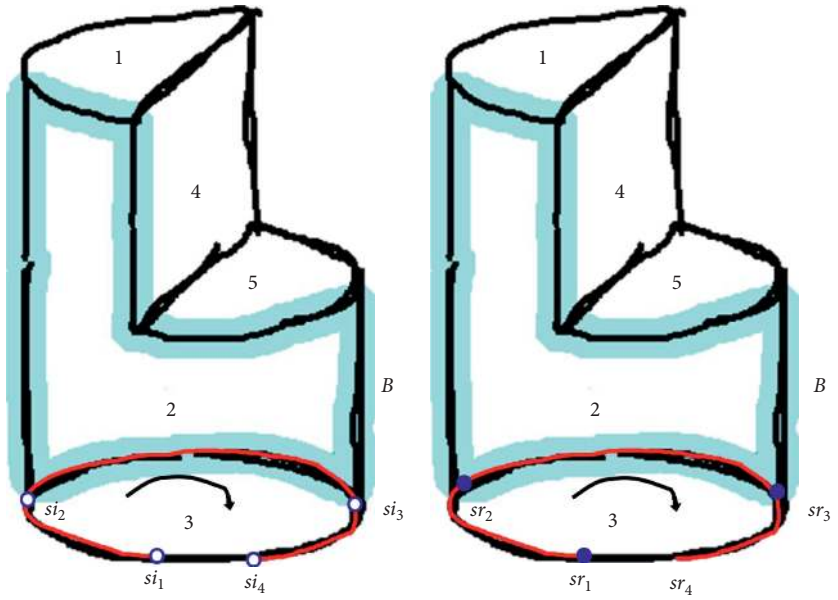
(vii) *Step 7*. The end.

3.2.3. Stroke Grouping. When a stroke is mostly inside the boundary bands of two adjacent regions simultaneously, it is considered to belong to the common boundary of the two regions. Particularly, if a stroke is mostly inside the boundary bands of three or more regions at the same time,



(a) (b)

Figure 4: Region boundary bands for (a) a closed region and (b) the entire sketch.



(a) (b)

Figure 5: Cross-regional strokes and their split points. (a) Initial split points. (b) Real split points.

then it is assigned to the common boundary of every two regions. The common boundary of the regions i and j is denoted by $\mathbf{B}(i, j)$.

The stroke grouping process is illustrated in Figure 6. Stroke $\mathbf{S}_1$ (shown in red) is mostly inside the boundary bands of region 2 and region 4 simultaneously, so it is assigned to the common boundary of region 1 and region 4, denoted by the relationship $\mathbf{S}_1 \in \mathbf{B}\{2, 4\}$. Stroke $\mathbf{S}_2$ (shown in green) is short and is located in the boundary bands of regions 1, 2, 3, and 4 simultaneously, so it is assigned to multiple regional common boundaries; that is, $\mathbf{S}_2 \in \mathbf{B}\{1, 2\}\&\mathbf{B}\{1, 3\}\&\mathbf{B}\{1, 4\}\&\mathbf{B}\{2, 3\}\&\mathbf{B}\{2, 4\}\&\mathbf{B}\{3, 4\}$. Invalid common regional boundaries, like $\mathbf{B}\{1, 2\}$ and $\mathbf{B}\{3, 4\}$, are removed if the length of the perimeter of the bounding boxes of the strokes inside them is smaller than the width of their constituent region boundary bands.

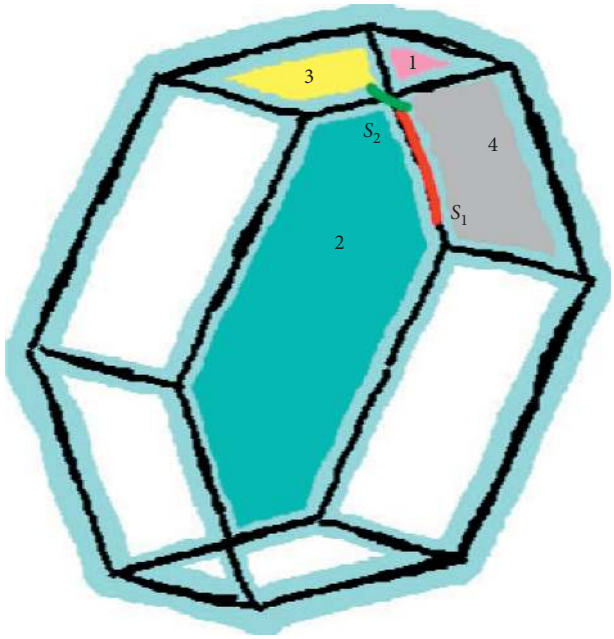


Figure 6: Assignment of strokes to common regional boundaries.

3.2.4. Multistroke Simplification. A bottom-up greedy algorithm is adopted to simplify grouped strokes into single strokes. The possible configurations of two strokes are shown in Figure 7. Two overtraced or disconnected stroke pairs are iteratively simplified into a new stroke until no new strokes can be created. We calculate the overtracing ratio of two overtraced strokes based on stroke tolerance zone, which is defined as a region around a stroke and is formed by equidistant curves, as shown in Figure 8(a). In Figure 8(b), a stroke tolerance zone is formed by a set of rectangles and circles built according to the polygonal approximate points of the stroke, shown as small blue circles, which are obtained using the split and merge algorithm of polygonal approximation [40]. The overtracing ratio of two strokes is defined as the percentage of the sample points of the smallest stroke which are located inside the tolerance zone of the largest one. As shown in Figure 8(c), the smallest stroke is that whose perimeter of the bounding box is shorter. The percentage of points that fall in a stroke tolerance zone is calculated by computing the number of points located inside the continuant rectangles and circles of the stroke tolerance zone. The width w of the stroke tolerance zone depends on the scale of the stroke, as shown in the following equation:

$$w = \frac{2}{3\sqrt{c} + 2b}, \tag{2}$$

where c and b are the perimeter of the bounding box and the width of the stroke, respectively. We set the value of b to 5 pixels.

During the simplification process, two strokes with a higher overtracing ratio are merged first. If the overtracing ratio of two strokes is close to 0 (zero), then the distances among their endpoints pair are further calculated.

Two nonovertraced strokes $\mathbf{S}_1$ and $\mathbf{S}_2$, which have close endpoints, are combined if the minimum distance between their endpoints is smaller than 20 pixels: (1) adjust two strokes in the same drawing direction, as shown in Figure 9(a), where the closest endpoints are the first point of $\mathbf{S}_1$ and the last point of $\mathbf{S}_2$; (2) link the list of sampling points of $\mathbf{S}_1$ and $\mathbf{S}_2$ end to end.

The simplification process of two overtraced strokes $\mathbf{S}_1$ and $\mathbf{S}_2$ is illustrated in Figure 10. First, each stroke is splitted into segments based on the other's tolerance zone. Then, the sampling points of the stroke segments in the overlapping area are interpolated to create a new stroke segment at the center line of the overlapping area. Finally, the interpolating stroke segment is combined with other stoke segments outside the overlapping area to form a longer stroke.

3.2.5. Adjacent Strokes Combination. After the steps above, the original input sketch is converted to a nonovertracing sketch, and each stroke is located between two adjacent branch points, as shown in Figure 2(d). Then the adjacent strokes that form a single primitive are combined to generate a new single stroke and finally to get a single stroke sketch. If the distance between their closest endpoints is less than the given threshold (we set this value to 20), then the two strokes are considered to be adjacent to each other. Adjacent single strokes are clustered according to the shapes of the new stroke generated by combining them and the original input cross-regional strokes. The detailed approach to decide whether two adjacent strokes $\mathbf{S}_1$ and $\mathbf{S}_2$ should be combined is described as follows.

(i) *Step 1.* Link the sampling points of $\mathbf{S}_1$ and $\mathbf{S}_2$ end to end and generate a longer stroke $\mathbf{S}_N$.

(ii) *Step 2.* Recognize the geometric type of $\mathbf{S}_N$ by using the fuzzy classification method [39]. If $\mathbf{S}_N$ is a line segment or a conic curve, proceed to Step 4.

(iii) *Step 3.* Traverse the original input cross-regional strokes list $\{\mathbf{Sc}_j\}$ and calculate the overtracing ratio of $\mathbf{Sc}_j$ with S_1 and S_2, respectively. If there is an original cross-regional stroke overtraced with both $\mathbf{S}_1$ and $\mathbf{S}_2$ and the overtracing ratios are both smaller than a threshold (we set this value to be 0.8), then proceed to Step 4.

(iv) *Step 4.* $\mathbf{S}_1$ and $\mathbf{S}_2$ can be combined together, and replace them with $\mathbf{S}_N$.

(v) *Step 5.* The end.

4. Experiment Results

4.1. Results. We implemented the common regional boundaries based multistroke recognition system using the Visual C++ programming language on a computer with a Windows 7 operating system. The input device can be a mouse or a tablet stylus. To validate the effectiveness of our method, the multistroke sketch recognition system is evaluated against various multistroke axonometric sketches, such as those shown in Figures 11–16. We executed the entire experiments on a 3.5 GHz computer with 10 GB of RAM. Currently, the closed regions and the region boundary bands of the sketch are extracted using MATLAB and they are manually inputted to the multistroke sketch recognition system.

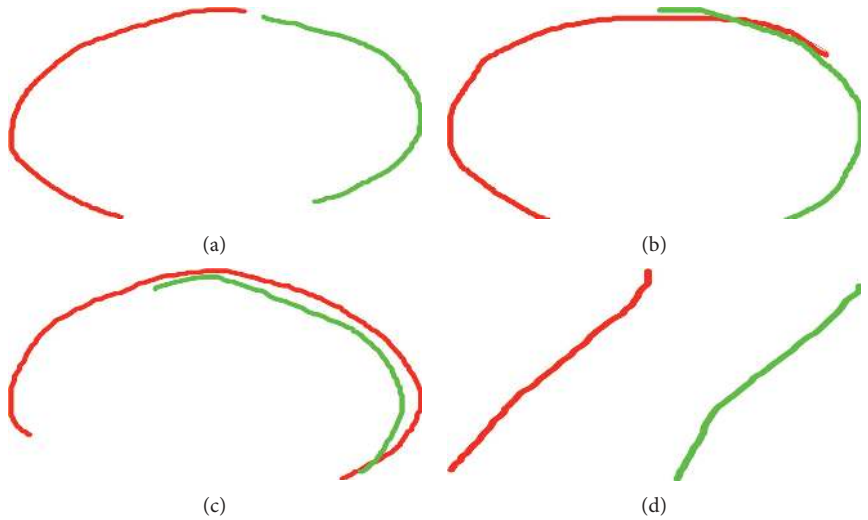


Figure 7: Possible structures of two strokes: (a) nonovertracing continuous strokes, (b) partly overtraced strokes, (c) fully overtraced strokes, and (d) nonovertracing disconnected strokes.

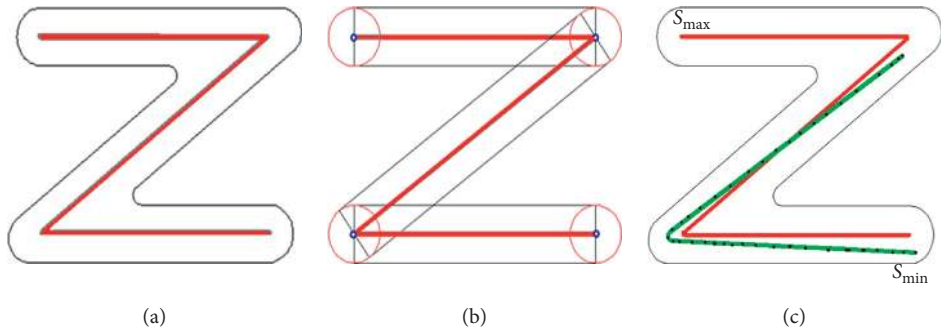


Figure 8: Tolerance zone of strokes and overtracing ratio computation between two strokes.

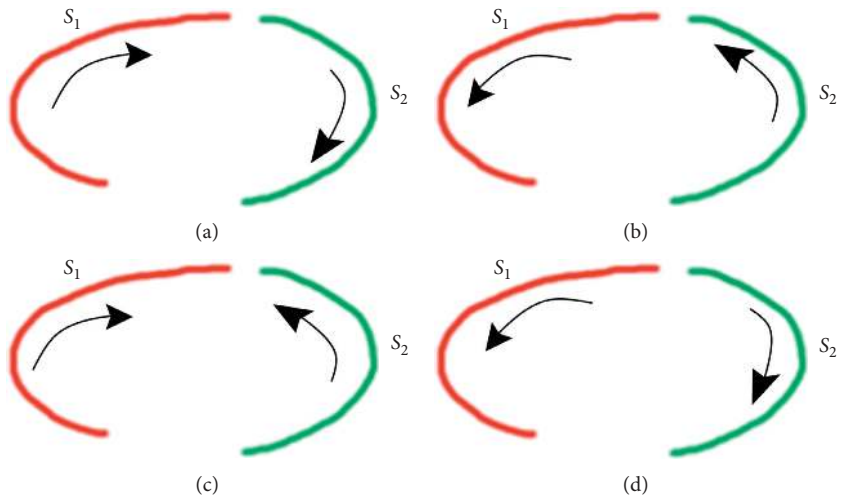


Figure 9: The relative direction of two nonovertraced strokes with close endpoints.

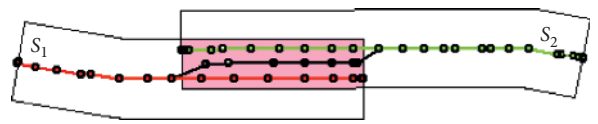


Figure 10: Simplification of two overtraced strokes.

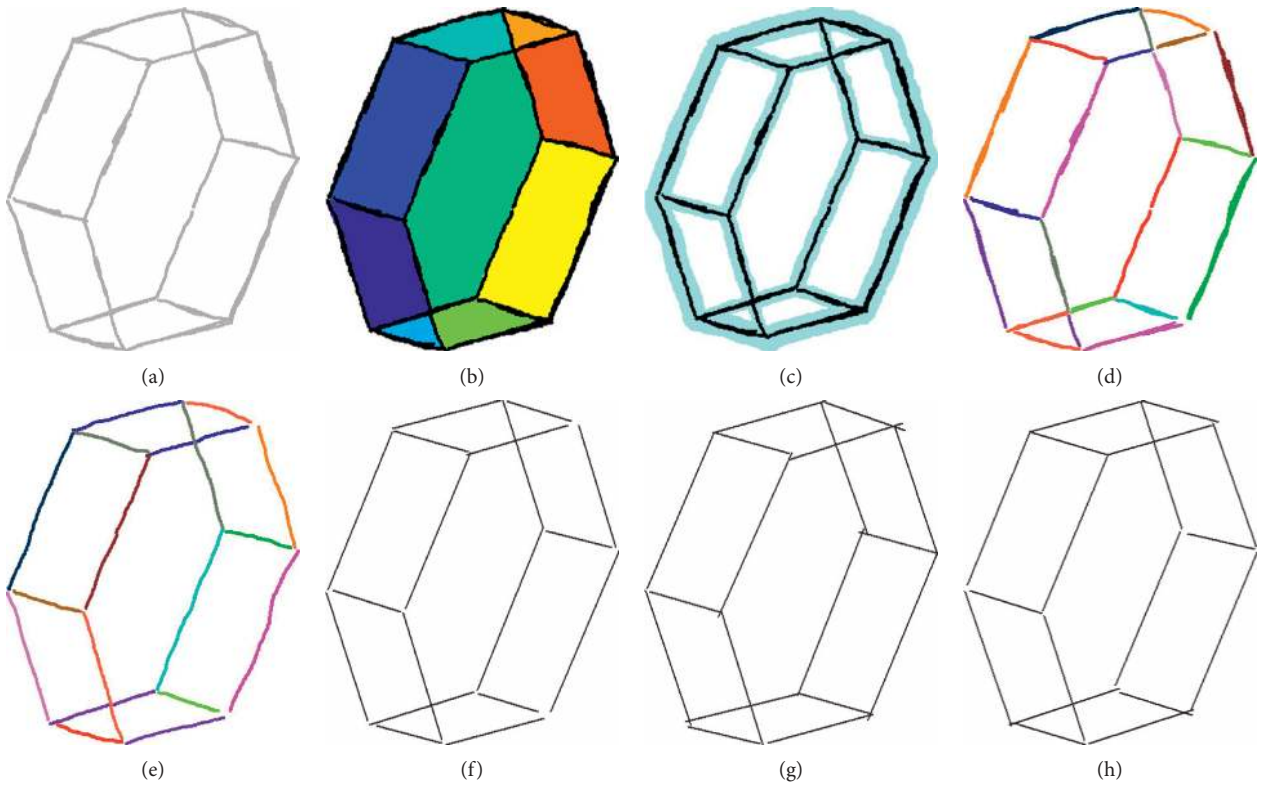


Figure 11: Axonometric sketch consists of line segments with hidden parts drawn in solid lines. (a) Input. (b) Closed regions. (c) Region boundary bands. (d) Stroke grouping. (e) Multistroke simplification. (f) Final results. (g) Ku et al. [6]. (h) Our previous multistroke grouping method [7].

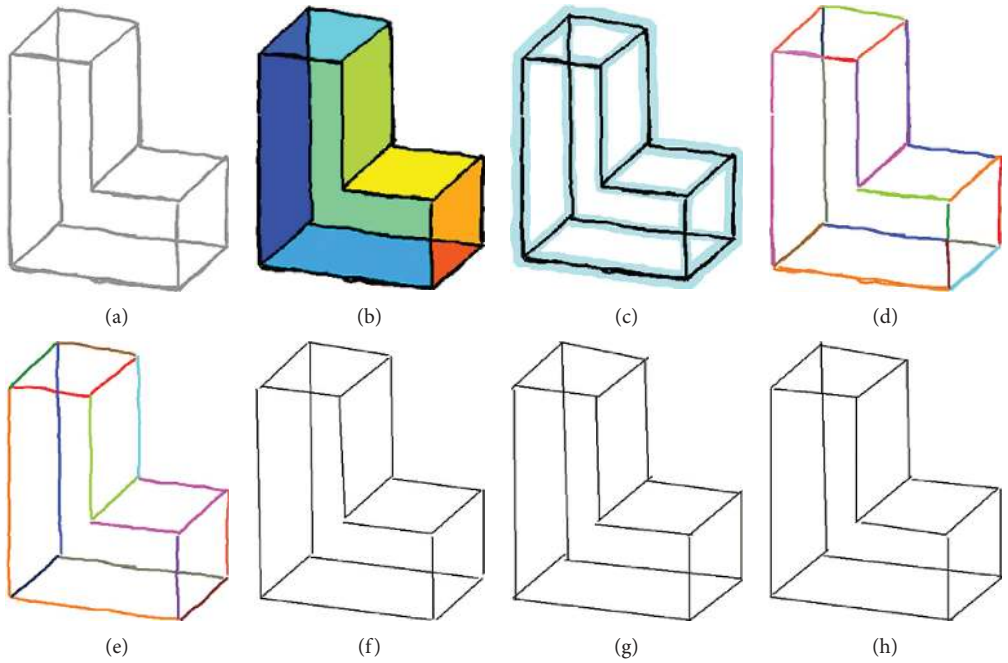


Figure 12: Axonometric sketch consists of line segments and polylines with hidden parts drawn in solid lines. (a) Input. (b) Closed regions. (c) Region boundary bands. (d) Stroke grouping. (e) Multistroke simplification. (f) Final results. (g) Ku et al. [6]. (h) Our previous multistroke grouping method [7].

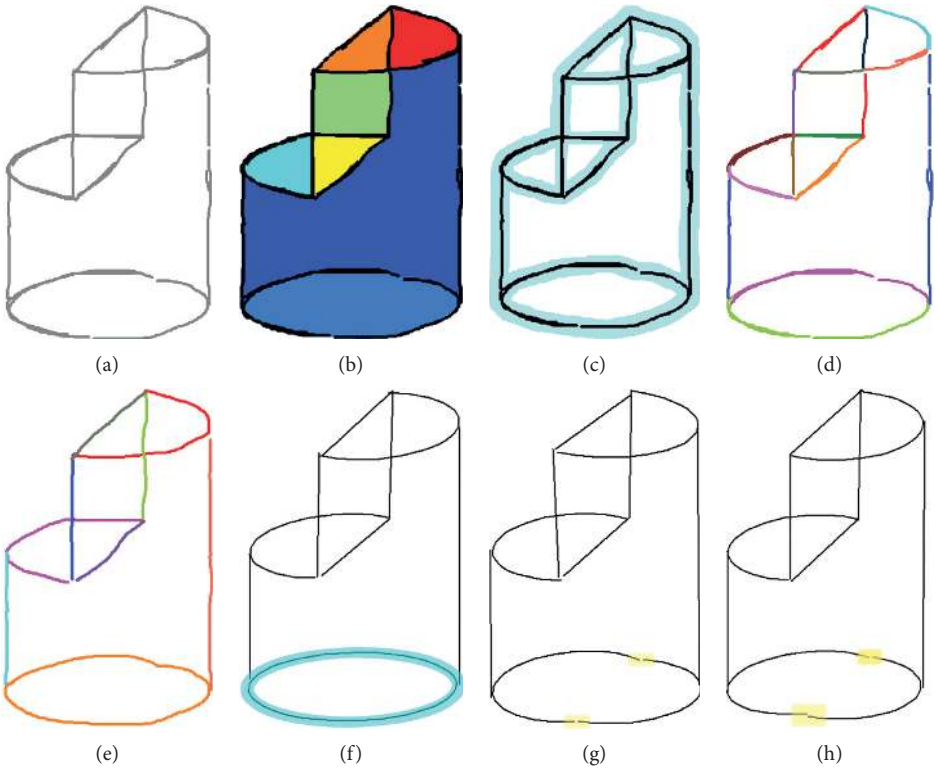


Figure 13: Axonometric sketch consists of line segments and conic curves with hidden parts drawn in solid lines. (a) Input. (b) Closed regions. (c) Region boundary bands. (d) Stroke grouping. (e) Multistroke simplification. (f) Final results. (g) Ku et al. [6]. (h) Our previous multistroke grouping method [7].

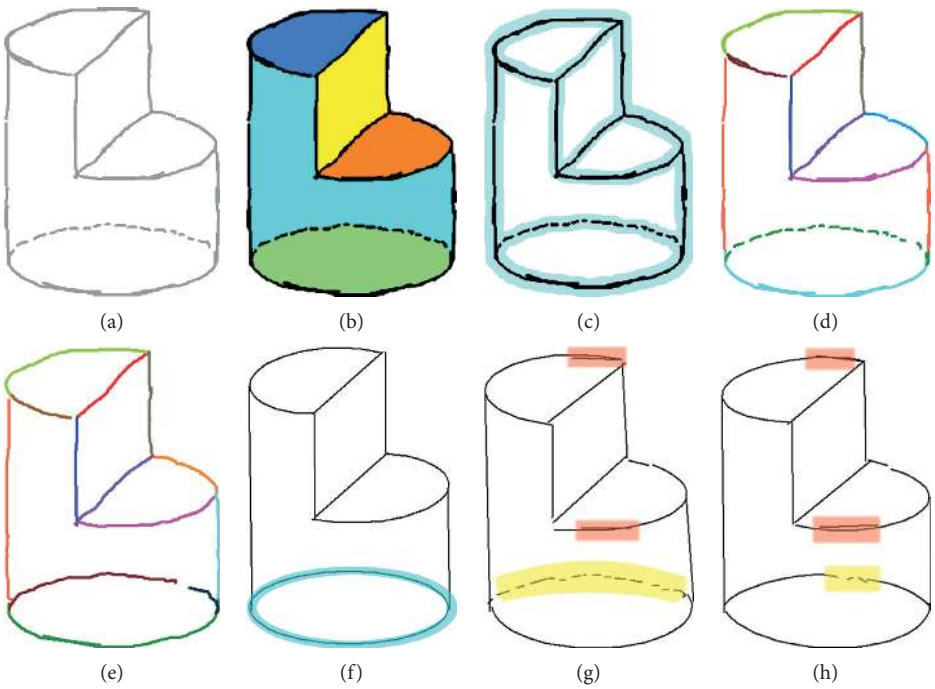
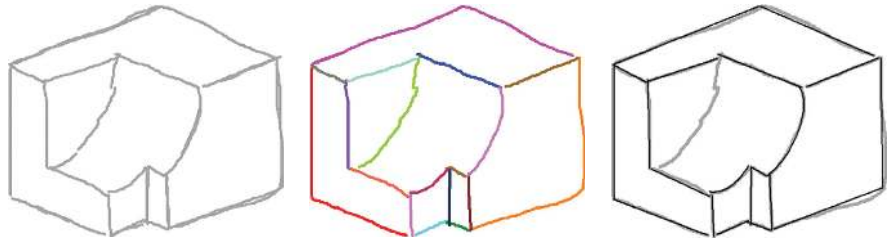
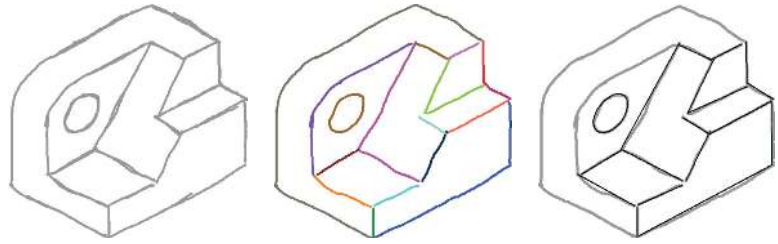


Figure 14: Axonometric sketch consists of line segments and conic curves with hidden parts drawn in dashed lines. (a) Input. (b) Closed regions. (c) Region boundary bands. (d) Stroke grouping. (e) Multistroke simplification. (f) Final results. (g) Ku et al. [6]. (h) Our previous multistroke grouping method [7].



(a) (b) (c)

Figure 15: Erroneous results caused by stroke recognition method. (a) Input. (b) Simplified strokes. (c) Final results.



(a) (b) (c)

Figure 16: Axonometric sketch without hidden lines. (a) Input. (b) Simplified strokes. (c) Final results.

Figures 11–14 show various multistroke axonometric sketches with hidden parts drawn in solid lines or dashed lines. All the sketches contain the overtraced strokes and nonovertraced strokes, which form a single primitive together. Each input stroke represents one geometric primitive, such as a line segment, a polyline, or a conic curve. The sketch in Figure 11 only contains line segments, while the sketch in Figure 12 consists of both line segments and polylines. The sketch in Figure 13 is composed of line segments and conic curves, and the multiple strokes forming a primitive are of the same geometric types. The sketch in Figure 14 has hidden lines drawn in dashed lines, which is actually a collection of disconnected and disordered short strokes.

The processes of simplifying input sketches into line drawings of Figures 11–14 are shown in detail as follows: (a) input sketch, (b) closed regions, (c) region boundary bands, (d) stroke groups from stroke segmentation and grouping, (e) single strokes from multiple strokes simplification, and (g) final results from single stroke fitting.

Moreover, we compared the common regional boundaries based stroke grouping method to the related grouping method of Ku et al. [6] and our previous stroke grouping method [7], in which the result is shown in parts (g) and (h) of Figures 11–14. The two methods are designed to create line drawings from multistroke sketches for geometric reconstruction.

The features of the sketches in Figures 11–14, such as the number of input strokes, the number of stroke groups, the total computational time, and the number of the simplified primitives, of the proposed method and the existing methods are listed in Table 1.

4.2. Discussions. From Table 1, the number of simplified primitives of the common regional boundaries based stroke grouping method is less than the two existing methods in all cases, which are mainly designed for overtracing sketches. The methods in [6, 7] grouped line segments based on endpoints location and slopes, so they can group nonovertraced line segments. The method [6] only deals with multistroke line segments and overtraced curves, while the method in [7] can also deal with multistroke polylines. The proposed method produced the same stroke grouping results as the methods in [6, 7] in Figure 11 and also produced the same stroke grouping results as [7] in Figure 12.

The differences between our method and the two existing stroke grouping methods [6, 7] mainly lie in the grouping results of different stroke types. As shown in Figures 14(g) and 14(h), the method in [6] cannot group overtraced strokes of different geometric types (shown in orange) or multiple nonovertracing strokes that form a longer line (shown in orange). Our previous method [7] groups short strokes that are drawn over consecutive periods of time and recognises them into longer curves, so it performed better than the method in [6] on grouping of short nonovertracing strokes (shown in yellow). As shown in Figures 13(g) and 13(h) and Figures 14(g) and 14(h), both methods in [6, 7] fail to group overtraced strokes of different geometric types (shown in orange). However, as shown in Figures 13(e) and 14(e), the proposed method groups short strokes that formed a longer line of other geometric types and overtraced strokes of different geometric types successfully, because it treats proximal strokes as a whole regardless of their geometric types, thus providing the users with a higher degree of freedom and creating a highly robust sketching system. Moreover, compared with the existing methods that iteratively group two strokes based on the local geometric properties, our method groups strokes at a higher level and more strokes can be grouped together at once.

Table 1: Sketch specifications and computational performance statistics of the sketches.

Figure		11	12	13	14
# of strokes		44	29	22	50
# of region		9	9	7	5
# of groups		18	22	15	11
# of simplified primitives	Proposed method	18	13	9	9
	Ku et al. [6]	18	18	10	28
	Wang et al. [7]	18	13	10	16
Total time (s)		60.411	70.186	43.616	37.274

4.3. Limitations. The proposed method partly depends on the results of the single-stroke recognition [39], which might cause wrong results in some cases. For example, as shown in Figure 15, the input multistroke sketch is simplified into a single-stroke sketch accurately, but some strokes are not recognized correctly by the single-stroke recognition method.

The proposed method is applicable to axonometric sketches with hidden parts drawn in solid lines. However, axonometric sketches in engineering domains sometimes do not contain hidden lines. As shown in Figure 16, the proposed method produces erroneous results on an axonometric sketch without hidden lines. For such drawings, the stroke segmentation process is required before the stroke fitting process. We plan to continue the research in this field for the near future.

According to Table 1, we can see that the proposed approach took about 1 minute to vectorize an input sketch. We observed that the running time of the proposed method highly depends on the size and complexity of the input sketches and the number of their closed regions. Therefore, the proposed method may not be suitable for interactive applications or as a repeated component of a larger process.

5. Conclusion

In this paper, we propose a stroke grouping method based on common boundaries between adjacent regions to create line drawings from online multistroke axonometric sketches of mechanical models. The method consists of five parts, namely, (1) region boundary band extraction, (2) cross-regional stroke segmentation, (3) stroke grouping based on common regional boundaries, (4) multistroke simplification, and (5) adjacent strokes combination. The proposed method does not limit the overlapping ratio and the geometric types of grouped strokes. It can handle both multiple overtracing and nonovertracing sketches. However, currently, it only considers stroke grouping of the axonometric sketches with hidden lines. For the axonometric sketches without hidden lines, the complex strokes that represent the common boundaries of two adjacent regions need to be further segmented into single primitives. We will study and address these limitations in future work.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was partly supported by National Key R&D Program of China (Grant no. 2019YFB1703800), Fundamental Research Funds for the Central Universities (Grant no. 3102020gxb003), Natural Science Basic Research Plan in Shaanxi Province of China (Grant no. 2016JM6054), and Programme of Introducing Talents of Discipline to Universities (111 Project), China (Grant no. B13044).

References

[1] N. Chansri and P. Koomsap, "Automatic single-line drawing creation from a paper-based overtraced freehand sketch," *The International Journal of Advanced Manufacturing Technology*, vol. 59, no. 1-4, pp. 221–242, 2012.

[2] H. Tian and S. Qin, "Exploring local regularities for 3D object recognition," *Chinese Journal of Mechanical Engineering*, vol. 29, no. 6, pp. 1104–1113, 2016.

[3] P. Company, M. Contero, J. Conesa, and A. Piquer, "An optimisation-based reconstruction engine for 3D modelling by sketching," *Computers & Graphics*, vol. 28, no. 6, pp. 955–979, 2004.

[4] T. M. Sezgin and R. Davis, "HMM-based efficient sketch recognition," in *Proceedings of the 10th International Conference on Intelligent User Interfaces*, Citeseer, San Diego, CA, USA, January 2005.

[5] T. M. Sezgin, T. Stahovich, and R. Davis, "Sketch based interfaces: early processing for sketch understanding," in *Proceedings of the ACM SIGGRAPH 2006 Courses on-SIGGRAPH'06*, ACM, Los Angeles, CA, USA, July 2006.

[6] D. C. Ku, S. F. Qin, and D. K. Wright, "Interpretation of overtracing freehand sketching for geometric shapes," in *Proceedings of the 14th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision 2006, WSCG'2006*, 263–270, WSCG, Plzen, Czech Republic, January 2006.

[7] S. Wang, S. Qin, and M. Gao, "New grouping and fitting methods for interactive overtraced sketches," *The Visual Computer*, vol. 30, no. 3, pp. 285–297, 2014.

[8] A. Fitzgibbon, M. Pilu, and R. B. Fisher, "Direct least square fitting of ellipses," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 21, no. 5, pp. 476–480, 1999.

[9] P. Company, R. Plumed, and P. A. C. Varley, "A fast approach for perceptually-based fitting strokes into elliptical arcs," *Visual Computer*, vol. 31, no. 6-8, pp. 775–785, 2015.

[10] S. Qin and H. Tian, "On the Algorithm for Reconstruction of Polyhedral Objects from a Single Line Drawing," in *Proceedings of the 2015 21st International Conference on Automation and Computing (ICAC)*, IEEE, Glasgow, UK, September 2015.

[11] R. Arnheim, "Untersuchungen zur Lehre von der Gestalt," *Psychologische Forschung*, vol. 11, no. 1, pp. 2–119, 1928.

[12] X. Liu, T.-T. Wong, and P.-A. Heng, "Closure-aware sketch simplification," *ACM Transactions on Graphics (TOG)*, vol. 34, no. 6, p. 168, 2015.

[13] B. Yu and S. Cai, "A domain-independent system for sketch recognition," in *Proceedings of the 1st international conference on Computer graphics and interactive techniques in Austalasia and South East Asia - GRAPHITE'03*, ACM, Melbourne, Australia, February 2003.

[14] D. J. Kang, M. Masry, and H. Lipson, "Reconstruction of a 3D object from a main axis system," in *Proceedings of the AAAI Fall Symposium Series: Making Pen-Based Interaction Intelligent and Natural*, AAAI, Menlo Park, CA, USA, January 2004.

[15] M. Masry, D. Kang, and H. Lipson, "A freehand sketching interface for progressive construction of 3D objects," *Computers & Graphics*, vol. 29, no. 4, pp. 563–575, 2005.

[16] T. Hammond and B. Paulson, "Recognizing sketched multistroke primitives," *ACM Transactions on Interactive Intelligent Systems (TiiS)*, vol. 1, no. 1, p. 4, 2011.

[17] J. Mitani, H. Suzuki, and F. Kimura, "3D sketch: sketch-based model reconstruction and rendering," in *From Geometric Modeling to Shape Modeling*, Springer, 2000.

[18] S.-H. Bae, R. Balakrishnan, and K. Singh, "ILoveSketch: as-natural-as-possible sketching system for creating 3d curve models," in *Proceedings of the 21st annual ACM symposium on User interface software and technology-UIST'08*, ACM, Monterey, CA, USA, October 2008.

[19] A. Shesh and B. Chen, "SMARTPAPER: an interactive and user friendly sketching system," *Computer Graphics Forum*, vol. 23, no. 3, pp. 301–310, 2004.

[20] P. Barla, J. Thollot, and F. X. Sillion, "Geometric clustering for line drawing simplification," in *Proceedings of the ACM SIGGRAPH 2005 Sketches on-SIGGRAPH'05*, ACM, New York, NY, USA, July 2005.

[21] R. Pusch, F. Samavati, A. Nasri, and B. Wyvill, "Improving the sketch-based interface - forming curves from many small strokes," *Visual Computer*, vol. 23, no. 9-11, pp. 955–962, 2007.

[22] A. Shesh and B. Chen, "Efficient and dynamic simplification of line drawings," *Computer Graphics Forum*, vol. 23, no. 2, pp. 537–545, 2008.

[23] Y. Chien, W. C. Lin, T. S. Huang, and J. H. Chuang, "Line drawing simplification by stroke translation and combination," in *Proceedings of the Fifth International Conference on Graphic and Image Processing (ICGIP 2013)*, SPIE, Hong Kong, China, October 2013.

[24] Y. Liu, X. Li, P. Bo, and X. Gao, "Sketch simplification guided by complex agglomeration," *Science China Information Sciences*, vol. 62, no. 5, p. 52105, 2019.

[25] G. Orbay and L. B. Kara, "Beautification of design sketches using trainable stroke clustering and curve fitting," *IEEE Transactions on Visualization and Computer Graphics*, vol. 17, no. 5, pp. 694–708, 2011.

[26] T. Ogawa, Y. Matsui, T. Yamasaki, and K. Aizawa, "Sketch simplification by classifying strokes," in *Proceedings of the 2016 23rd International Conference on Pattern Recognition (ICPR)*, IEEE, Cancun, Mexico, December 2016.

[27] T. M. Sezgin and R. Davis, "Handling overtraced strokes in hand-drawn sketches," in *Proceedings of the Making Pen-Based Interaction Intelligent and Natural*, AAAI, Arlington, VA, USA, October 2004.

[28] L. B. Kara and T. F. Stahovich, "An image-based, trainable symbol recognizer for hand-drawn sketches," *Computers & Graphics*, vol. 29, no. 4, pp. 501–517, 2005.

[29] A. Bartolo, K. P. Camilleri, S. G. Fabri, J. C. Borg, and P. J. Farrugia, "Scribbles to vectors: preparation of scribble drawings for CAD interpretation," in *Proceedings of the 4th Eurographics workshop on Sketch-based interfaces and modeling-SBIM'07*, ACM, Riverside, CA, USA, August 2007.

[30] A. Bonnici and K. P. Camilleri, "Scribble vectorization using concentric sampling circles," in *Proceedings of the 2009 Third International Conference on Advanced Engineering Computing and Applications in Sciences*, IEEE, Sliema, Malta, October 2007.

[31] J.-D. Favreau, F. Lafarge, and A. Bousseau, "Fidelity vs. simplicity," *ACM Transactions on Graphics*, vol. 35, no. 4, pp. 1–10, 2016.

[32] E. Simo-Serra, S. Iizuka, K. Sasaki, and H. Ishikawa, "Learning to simplify: fully convolutional networks for rough sketch cleanup," *Acm Transactions on Graphics*, vol. 35, no. 4, p. 121, 2016.

[33] E. Simo-Serra, S. Iizuka, and H. Ishikawa, "Mastering sketching: adversarial augmentation for structured prediction," *ACM Transactions on Graphics*, vol. 37, no. 1, 2017.

[34] J. Chen, M. Du, X. Qin, and Y. Miao, "An improved topology extraction approach for vectorization of sketchy line drawings," *The Visual Computer*, pp. 1–12, 2018.

[35] A. D. Parakkat, U. Bondi Pundarikaksha, and R. Muthuganapathy, "A Delaunay triangulation based approach for cleaning rough sketches," *Computers & Graphics*, vol. 74, pp. 171–181, 2018.

[36] L. Donati, S. Cesano, and A. Prati, "A complete hand-drawn sketch vectorization framework," *Multimedia Tools and Applications*, vol. 78, no. 14, pp. 19083–19113, 2019.

[37] M. J. Fonseca and J. A. Jorge, "Using fuzzy logic to recognize geometric shapes interactively," in *Proceedings of the Ninth IEEE International Conference on Fuzzy Systems. FUZZ- IEEE 2000 (Cat. No.00CH37063)*, IEEE, San Antonio, TX, USA, May 2000.

[38] S.-H. Zhang, T. Chen, Y.-F. Zhang, S.-M. Hu, and R. R. Martin, "Vectorizing cartoon animations." visualization and computer graphics," *IEEE Transactions on*, vol. 15, no. 4, pp. 618–629, 2009.

[39] S. X. Wang, G. F. Wang, M. T. Gao, and S. H. Yu, "Using fuzzy hybrid features to classify strokes in interactive sketches," *Advances in Mechanical Engineering*, vol. 5, 2013.

[40] P. L. Rosin, "Techniques for assessing polygonal approximations of curves," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 6, pp. 659–666, 1997.