Extraction of Line Segments and Circular Arcs From Freehand Strokes Based on Segmental Homogeneity Features

Xiwen Zhang, *Member, IEEE*, Jiqiang Song, *Member, IEEE*, Guozhong Dai, and Michael R. Lyu, *Fellow, IEEE*

***Abstract*—The extraction of component line segments and circular arcs from freehand strokes along with their relations is a prerequisite for sketch understanding. Existing approaches usually take three stages to segment a stroke: first identifying segmentation points, then classifying the substroke between each pair of adjacent segmentation points, and, finally, obtaining graphical representations of substrokes by fitting graphical primitives to them. Since a stroke inevitably contains noises, the first stage may produce wrong or inaccurate segmentation points, resulting in the wrong substroke classification in the second stage and inaccurately fitted parameters in the third stage. To overcome the noise sensitivity of the three-stage method, the segmental homogeneity feature is emphasized in this paper. We propose a novel approach, which first extracts graphical primitives from a stroke by a connected segment growing from a seed-segment and then utilizes relationships between the primitives to refine their control parameters. We have conducted experiments using real-life strokes and compared the proposed approach with others. Experimental results demonstrate that the proposed approach is effective and robust.**

***Index Terms*—Segmental homogeneity feature, sketch understanding, stroke recognition, stroke segmentation.**

I. Introduction

GRAPHICAL objects are essential and frequently used for information representation in daily life and work. However, the input of graphical objects into computers is not convenient for most users, of course, not including proficient CAD drawers through much training. In common CAD software tools, users have to memorize many command strings, select and type one of them, or search through menus or buttons to choose specific ones, to generate their desired graphical objects, and it is awkward or confusing [1]. Alternatively, by taking advantage of pen-based computing [2], they could simply sketch graphical objects using an electronic pen on an electronic tablet or a paper, and the objects are recognized automatically by computer. It is natural and fluent. More importantly, it is identical with the right way people sketch on paper by pen. Due to its potential for use as a more natural alternative for entering graphical objects into computers, sketch recognition has attracted more research attention recently [3].

In fact, a sketch is composed of strokes. A stroke is a set of temporally ordered sampling points captured in a single sequence of pen-down, pen-move, and pen-up. To allow pen-input in a more natural way, one stroke may contain more than graphical primitives [4]–[6]. We define a substroke as a component of a stroke, which corresponds to a graphical primitive. According to the number of graphical primitives, a stroke can be classified into two categories: 1) a stroke with a single primitive (a single primitive stroke) and 2) a stroke with multiple primitives (a multiprimitive stroke). The graphical primitives are fitted to substrokes from a multiprimitive stroke or a single primitive stroke. The points representing graphical primitives are defined as their control points.

The primitives contained in strokes include line segments and circular arcs. The two types of primitives in a stroke may concatenate at their endpoints to form corners. They may smoothly connect with each other, e.g., a circular arc is tangent with a line segment or a circular arc. Thus, extracting component line segments and circular arcs from a stroke along with their relations is prerequisite for achieving the best sketch understanding.

Existing approaches [4]–[6] to extracting graphical primitives from a stroke by the following three stages. 1) Segmentation point detection: a stroke is divided into several substrokes by locating segmentation points. 2) Substroke classification: each substroke is classified as a type of graphical primitive. 3) Substroke fitting: each substroke is fitted to a graphical primitive according to its type to obtain its control parameters.

A stroke inevitably contains noises, which come from two sources. One is the minor hand fluctuation during sketching, and the other is the digitizing error of the input device. The noise causes many false positives in segmentation point detection [6], whereas current approaches [4]–[9] are not robust enough to effectively extract graphical primitives from strokes with many noises. However, human can easily identify each substroke of a stroke, since each substroke is a homogeneous one regarding to its graphical feature. Based on this observation,

Manuscript received September 26, 2004; revised April 18, 2005. This work was supported in part by the National Key Basic Research and Development Program of China under Grant 2002CB312103, in part by the National Natural Science Foundation of China under Grants 600373056 and 60033020, and in part by the Research Grants Council of the Hong Kong Special Administrative Region, China, under Grant CUHK4182/03E. This paper was recommended by Associate Editor V. Govindaraju.

X. Zhang is with the Laboratory of Human-Computer Interaction and Intelligent Information Processing, Institute of Software, The Chinese Academy of Sciences, Beijing 100080, China, and also with the Department of Computer Science and Engineering, The Chinese University of Hong Kong, Shatin, N.T., Hong Kong (e-mail: xwzhang@cse.cuhk.edu.hk).

J. Song is with the Hong Kong Applied Science and Technology Research Institute, Science Park, Shatin, N.T., Hong Kong (e-mail: jqsong@astri.org).

G. Dai is with the Laboratory of Human-Computer Interaction and Intelligent Information Processing, Institute of Software, The Chinese Academy of Sciences, Beijing 100080, China (e-mail: guozhong@admin.iscas.ac.cn).

M. R. Lyu is with the Department of Computer Science and Engineering, The Chinese University of Hong Kong, Shatin, N.T., Hong Kong (e-mail: lyu@cse.cuhk.edu.hk).

Digital Object Identifier 10.1109/TSMCB.2005.857288

the segmental homogeneity feature is emphasized in this paper. Suppose a substroke is divided into equal-length segments and each segment is characterized using its homogeneity features, the difference among these segments becomes much smaller, which means the homogeneity features of segments within the same substroke are similar. These observations can help extract, fast and accurately, substokes from a stroke. Consequently, this paper proposes a novel approach to extract graphical primitives from strokes based on the segmental homogeneity feature. The relations between primitives are determined based on a set of domain rules and are exploited to refine the control parameters of extracted graphical primitives for perfect visual representation.

The rest of this paper is organized as follows. Section II reviews the previous work related to graphical primitive extraction from strokes and presents our analyses. Section III describes the definition of a seed-segment and the segment-growing algorithm based on it. Section IV applies the proposed approach to the extraction of graphical primitives from strokes and utilizes their relations to refine their control parameters. In Section V, detailed experimental results and performance analyses are reported, and some comparisons with other approaches are also given. Section VI draws our conclusions.

II. Related Work

Existing three-stage approaches to stroke segmentation first identify segmentation points. Approaches for detecting segmentation points can be classified into three classes according to their detection features.

1) Approaches based on speeds and curvatures of stroke points.

   Calhoun *et al.* [7] identify segmentation points using both speeds and distance sign changes of stroke points. The distance sign is estimated using a window of points on either side of a current point.

   The above approach works well for sketches with relatively low noise levels, but yields many false positives for noisy sketches. Thus, Davis [6] first employs an average-based filtering to reduce (but not eliminate) false positives, and then combines speed and curvature to identify segmentation points [8]. The average-based filtering looks for extremes only in areas of the speed and curvature that exceed the average value.

   Although strokes can be usually smoothed with appropriate filters to suppress noisy points, it is difficult to select filters that work equally well for different strokes [9]. Too little smoothing leads to superfluous corners whereas excessive smoothing causes the disappearance of true corners [7].

2) Fuzzy logic approaches based on speeds, accelerations, directions, and angles of stroke points.

   Strokes are segmented through finding obtuse and acute corner points and inflection points [4], [5], which are identified as segmentation points. These approaches exploit properties and fuzzy heuristic knowledge of segmentation points, which are related to speed and acceleration. The obtuse corner points are identified based on directional deviations of stroke points. The acute corner points are detected between two adjacent obtuse points, by an adaptive threshold and fuzzy knowledge, with respect to speeds, accelerations and linearity of stroke points. Inflection points are identified in terms of changes in convexity of stroke points.

3) Scale-space-based approaches.

   Sezgin *et al.* [9] propose a scale-space based approach to increase segmentation point detection accuracy in noisy sketches. The approach is based on curvature and speed. It is to look at the number of segmentation points presented at different scales in the scale space, and to select a scale where most of the noise is filtered out. The feature count graph is modeled by fitting two lines to it: one to the region with the steep drop corresponding to places where the segmentation points disappear due to noise and the other to the flat region where real segmentation points disappear. The scale corresponding to the intersection of these two lines is taken. The approach is computationally intensive [7] and, thus, cannot satisfy the rapid requirement for stroke segmentation.

After identifying segmentation points, substrokes are classified to representing lines, circular arcs, or other graphical primitives. There are three kinds of approaches to classify substrokes according to their underlying techniques.

1) Approaches based on the least-squares fitting error (LSFE).

   Calhoun *et al.* [7] compute LSFEs to a line and a circle for each substroke. A substroke is typically classified into the graphical primitive that matches with the least error. To be a valid circular arc, the subtending angle of the arc must be at least 15°.

   Shpitalni and Lipson [10], based on the linear least-squares fitting to a conic section equation, utilize the equation's natural property to classify strokes and identify lines, elliptic arcs, and corners composed of two lines with an optional fillet. The hyperbola form of the equation is used for corner detection.

2) Approaches based on length ratio.

   For a substroke, Davis [6] compares the Euclidean distance between a pair of segmentation points and the accumulated arc length covered by the original substroke points. The ratio of the arc length to the Euclidean distance is very close to 1 for a linear region and significantly higher than 1 for a curved region. They approximate substrokes with the ratio higher than 1 using Bezier curves and do those with the ratio very close to 1 using line segments.

3) Approaches based on shape.

   Qin *et al.* [5] classify a substroke as a line, a conic curve, or a free-form curve according to its linearity, convexity and complexity. A conic curve is further classified into a circle, a circular arc, an ellipse, or an elliptical arc using the weighted least-square fitting with some normalization techniques based on algebraic distances. When free-form curves are identified, B-spline curves are used to fit a set of stroke points.

There is much investigation [11]–[19] for segmenting the other kind of digital curves consisting of connected pixels. Those curves are detected from images scanned from paper drawings [11], [12], [16] or captured from other objects [13]–[15], [17]–[19]. These methods are not applicable to stroke segmentation because strokes consist of rather sparse points and contain more noisy points due to freehand sketching and digitization errors.

Previous related work on graphical primitive extraction from strokes does not provide complete information. They identify corners [1], [4], [6], [7], [9] and smooth joints of curves [5], but not tangent points between circular arcs and line segments or circular arcs, which are common and important in sketches. Line segments, circular arcs, Bezier curves, and B-spline curves are extracted from a stroke [4]–[7], [9], but their relations are not provided. Igarashi *et al.* [20] identify relations among strokes, including perpendicular, congruence, and symmetry. The strokes are all single primitives and only corresponding to line segments. Landay and Myers [21] also identify relations among strokes, including containing, near, vertical, and horizontal relations. In fact, the relations between the primitives in a stroke and the relations among strokes [9] are useful for sketch understanding. We think *a priori* knowledge in segmental homogeneity features should be incorporated into the extraction process of substrokes. Obviously, this will improve the entirety of extraction results and greatly reduce noise sensitivity. Furthermore, the relationship between graphical primitives can be used to refine their control parameters. Thus, our approach is designed on the following principles.

- Line segments and circular arcs are extracted in their entireties, even if they are cornered or tangent to each other.
- *A priori* knowledge of graphical primitives can be incorporated effectively to improve the reliability of the graphical primitive extraction.
- The criteria of homogeneity should be noise insensitive.
- The relations between graphical primitives can be extracted and used to refine control parameters of the primitives.
- The extracted information should be represented as a compact structure for the subsequent sketch understanding.

III. Definition and Growing of Seed-Segment

Due to digitizing errors and minor hand fluctuations, there are some unnecessary and inaccurate points in strokes, as shown in Fig. 1(a), which contains two strokes, i.e., $A'B'$ and $C'D'$, points A' and C' are points of pen-down, points B' and D' are ones of pen-up. Current graphical primitive extraction approaches based on segmentation point detection are prone to over-segmenting or under-segmenting strokes due to noisy points.

However, the human recognizes graphical primitives from a stroke mainly depending on their homogeneity of graphical features; that is to say, any two equal-length segments of the same substroke have very similar graphical features, including a slope and an intercept for a line or a radius and a center for a circle or an arc. Following this observation, we propose to use a seed-segment, which can best represent the graphical feature of a sub-

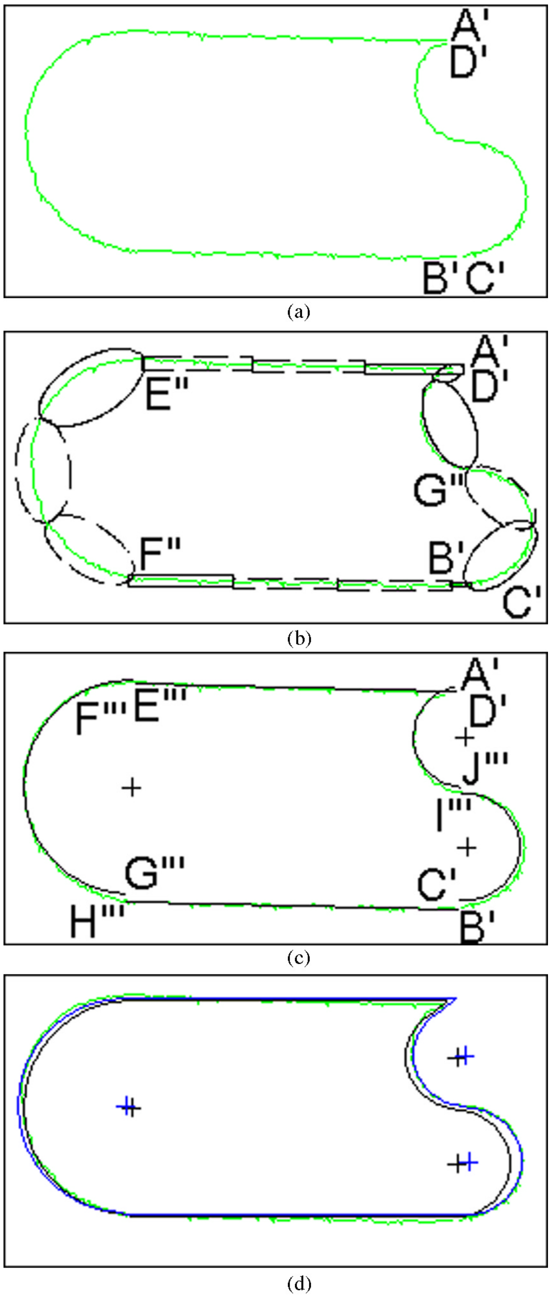


Fig. 1. Graphical primitives are extracted from a sketch. (a) A sketch with two strokes $A'B'$ and $C'D'$. (b) Substrokes $A'E''$, $E''F''$, $F''B'$ are extracted from stroke $A'B'$ and $C'G''$ and $G''D'$ are from $C'D'$. (c) Fitted results of substrokes are line segments $A'E'''$ and $H'''B'$ and arcs $F'''G'''$, $C'I'''$ and $J'''D'$. (d) Refined graphical primitives are label in black, and ground-truth ones manually extracted in blue. (Color version available online at http://ieeexplore.ieee.org.)

stroke, to identify the homogeneity feature of a substroke, and then extract the substroke based on the seed-segment growing.

A. Definition of a Seed-Segment

A seed-segment is in fact a segment of the original stroke and is used to extract a substroke. To define whether a segment can be a seed-segment, three homogeneity-related estimations for a segment are defined first.

1) The LSFE: LSFE is the smaller one between the line fitting error and the circle fitting error of the segment. It

indicates the membership extent of the segment being a straight-line segment or a circular arc.

2) Graphical Attributes (GAs): If the line fitting error is smaller, k and b are calculated according to formulas in the reference [11]; they are the slope and the intercept of a line equation $y = kx + b$, respectively. Otherwise, the circle fitting error is smaller, and x_C, y_C, and r are calculated according to formulas in the [11], they are coordinates of center and the radius of a circle, respectively. k and b or x_C, y_C, and r are graphical attributes of a segment.
3) Graphical Homogeneities (GHs): After dividing a segment into three equal-length subsegments, GHs are the maximum absolute differences of GAs (k and b for a line, or x_C, y_C, and r for a circular arc) of the segment over those of the three subsegments. They reflect the extent of homogeneity of the segment's graphical attributes. GHs for a line segment consist of $\mathrm{GH_k}$ and $\mathrm{GH_b}$. GHs for a circular arc consist of $\mathrm{GHx_c}$, $\mathrm{GHy_c}$, and $\mathrm{GH_r}$.

Definition 1: A seed-segment is a segment of a stroke that can well represent the graphical feature of a substroke. It must satisfy the following two constraints simultaneously.

1) LSFE is smaller than or equal to a preset threshold, i.e., $\mathrm{LSFE} \le \max\_\mathrm{LSFE}$.
2) GHs are smaller than or equal to a set of preset thresholds, i.e., $\mathrm{GH_k} \le \max\_\mathrm{GH_k}$, $\mathrm{GH_b} \le \max\_\mathrm{GH_b}$, or $\mathrm{GHx_c} \le \max\_\mathrm{GHx_c}$, $\mathrm{GHy_c} \le \max\_\mathrm{GHy_c}$, $\mathrm{GH_r} \le \max\_\mathrm{GH_r}$.

The thresholds, max_LSFE and max_GHs including $\max\_\mathrm{GH_k}$, $\max\_\mathrm{GH_b}$, or $\max\_\mathrm{GHx_c}$, $\max\_\mathrm{GHy_c}$, $\max\_\mathrm{GH_r}$, are preset by analyzing the substroke samples which represent graphical primitives. To determine these thresholds, we first manually collect many samples that are substrokes cropped from real-life strokes. Then, LSFE and GHs of each substroke are automatically calculated using the above definitions. Finally, max_LSFE is set to be the maximum of all LSFEs, and Max_GHs is set to be the maximum of all GHs. Therefore, for each type of substrokes, there is a set of preset thresholds. We provide two sets of thresholds for the substrokes corresponding to line segments and the circular arcs, respectively, in Section IV.

According to the positional relationship between a segment and its corresponding substroke, a segment can be classified into three categories: the internal segment, the boundary segment, and the external segment, as shown in Fig. 2. In the figure, there are two substrokes $\mathrm{C'G''}$ and $\mathrm{G''D'}$ in stroke $\mathrm{C'D'}$ and the external segment is for the substroke $\mathrm{C'G''}$. The definitions are as follows.

Definition 2: An internal segment is entirely inside the current substroke, which means all its subsegments are inside the substroke as well. A seed-segment must be an internal segment. A current substroke is the substroke being processed.

Definition 3: A boundary segment is partially inside the current substroke. There are at least one and at most two subsegments locating entirely inside the substroke.

Definition 4: An external segment is entirely outside the current substroke, but inside the stroke, which means all its subsegments are outside the current substroke, but inside the stroke.

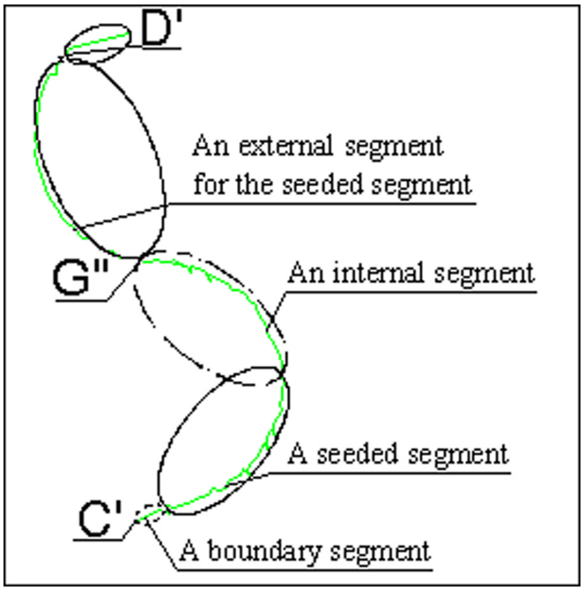


Fig. 2. Four categories of segments are for the substroke $\mathrm{C'G''}$. (Color version available online at http://ieeexplore.ieee.org.)

Since LSFEs and GHs of different substrokes vary, after identifying a seed-segment which corresponding to a substroke, the classification thresholds of three categories of segments are calculated based on the seed-segment. Considering the impact from noisy points, the conditions for these segments are not as strict as those for the seed-segment. The thresholds are relaxed as $\max\_\mathrm{LSFE'} = \mathrm{LSEF} + \varepsilon_\mathrm{L}$, $\max\_\mathrm{GHs'} = \mathrm{GHs} + \varepsilon_\mathrm{Gs}$, where ε_L and ε_Gs are small offsets from the LSFE and GHs of the seed-segment, respectively. ε_Gs for a line includes $\varepsilon_\mathrm{Gk}, \varepsilon_\mathrm{Gb}$, and for a circular arc include $\varepsilon_\mathrm{Gxc}, \varepsilon_\mathrm{Gyc}, \varepsilon_\mathrm{Gr}$. The internal segments should have similar graphical attributes with the seed-segment. For the line fitting process, $\max\_\mathrm{k'} = \mathrm{k} + \varepsilon_\mathrm{Sk}$, $\min\_\mathrm{k'} = \mathrm{a} - \varepsilon_\mathrm{Sk}$, and $\max\_\mathrm{b'} = \mathrm{b} + \varepsilon_\mathrm{Sb}$, $\min\_\mathrm{b'} = \mathrm{b} - \varepsilon_\mathrm{Sb}$ where ε_Sk and ε_Sb are small offsets from k and b of the seed-segment, respectively. For the circle fitting process, $\max\_\mathrm{x'_c} = \mathrm{x_c} + \varepsilon_\mathrm{Sxc}$, $\min\_\mathrm{x'_c} = \mathrm{x_c} - \varepsilon_\mathrm{Sxc}$, $\max\_\mathrm{y'_c} = \mathrm{y_c} + \varepsilon_\mathrm{Syc}$, $\min\_\mathrm{y'_c} = \mathrm{y_c} - \varepsilon_\mathrm{Syc}$, $\max\_\mathrm{r'} = \mathrm{r} + \varepsilon_\mathrm{Sr}$, and $\min\_\mathrm{r'} = \mathrm{r} - \varepsilon_\mathrm{Sr}$, where ε_Sxc, ε_Syc and ε_Sr are small offsets from $\mathrm{x_c}$, $\mathrm{y_c}$, and r of the seed-segment, respectively. Thus, during the seed-segment growing, a new segment (or subsegment) grows on the following three conditions: 1) $\mathrm{LSEF} \le \max\_\mathrm{LSFE'}$, 2) $\mathrm{GHs} \le \max\_\mathrm{GHs'}$, and 3) $\min\_\mathrm{k'} \le \mathrm{k'} \le \max\_\mathrm{k'}$, $\min\_\mathrm{b'} \le \mathrm{b'} \le \max\_\mathrm{b'}$, or $\min\_\mathrm{x'_c} \le \mathrm{x'_c} \le \max\_\mathrm{x'_c}$, $\min\_\mathrm{y'_c} \le \mathrm{y'_c} \le \max\_\mathrm{y'_c}$, $\min\_\mathrm{r'} \le \mathrm{r'} \le \max\_\mathrm{r'}$.

There are many parameters and thresholds to extract substrokes from strokes. They are listed in Table I.

B. Seed-Segment Growing

To extract a substroke from a stroke, we first detect a seed-segment from the stroke and determine which type of graphical primitive it represents. Then, the seed-segment grows by merging successive connected internal segments and boundary segments until no more satisfactory segments can be found. After the growing terminates, the seed-segment, the internal segments and all the internal subsegments of the

TABLE I
PARAMETERS AND THRESHOLDS FOR EXTRACTING GRAPHICS PRIMITIVES FROM STROKES

Seed-segment detection	**Segment Growing**		
$LSFE \leq max\_LSFE = 4$	$LSEF' \leq max\_LSFE' = LSEF + \varepsilon_L$	$\varepsilon_L = 2$	
$GH_k \leq max\_GH_k = 0.01$	$GH_k' \leq max\_GH_k' = GH_k + \varepsilon_{Gk}$	$\varepsilon_{Gk} = 0.01$	
$GH_b \leq max\_GH_b = 6$	$GH_b' \leq max\_GH_b' = GH_b + \varepsilon_{Gb}$	$\varepsilon_{Gb} = 2$	
$GHx_c \leq max\_GHx_c = 3$	$GHx_c' \leq max\_GHx_c' = GHx_c + \varepsilon_{Gxc}$	$\varepsilon_{Gxc} = 2$	
$GHy_c \leq max\_GHy_c = 3$	$GHy_c' \leq max\_GHy_c' = GHy_c + \varepsilon_{Gyc}$	$\varepsilon_{Gyc} = 2$	
$GH_r \leq max\_GH_r = 4$	$GHr' \leq max\_GHr' = GHr + \varepsilon_{Gr}$	$\varepsilon_{Gr} = 3$	
k	$min\_k' \leq k' \leq max\_k'$	$max\_k' = k + \varepsilon_{Sk}$ $min\_k' = a - \varepsilon_{Sk}$	$\varepsilon_{Sk} = 0.02$
b	$min\_b' \leq b' \leq max\_b'$	$max\_b' = b + \varepsilon_{Sb}$ $min\_b' = b - \varepsilon_{Sb}$	$\varepsilon_{Sb} = 4$
x_c	$min\_ x_c' \leq x_c' \leq max\_ x_c'$	$max\_x_c' = x_c + \varepsilon_{Sxc}$ $min\_ x_c' = x_c - \varepsilon_{Sxc}$	$\varepsilon_{Sxc} = 3$
y_c	$min\_ y_c' \leq y_c' \leq max\_ y_c'$	$max\_y_c' = y_c + \varepsilon_{Syc}$ $min\_y_c' = y_c - \varepsilon_{Syc}$	$\varepsilon_{Syc} = 3$
r	$min\_ r' \leq r' \leq max\_ r'$	$max\_r' = r + \varepsilon_{Sr}$ $min\_r' = r - \varepsilon_{Sr}$	$\varepsilon_{Sr} = 4$

boundary segments recorded during the growing form the substroke corresponding to a graphics primitive. The detailed algorithm of seed-segment growing is described as follows.

Step 1: Set the seed-segment as the active segment.
Step 2: Extract a forward segment candidate connected with the active segment.
Step 3: If the segment candidate is an internal segment, then set it as the active segment and return to Step 2.
Step 4: If the segment candidate is a boundary segment, then extract its internal subsegments.
Step 5: If the backward direction has not been processed, then repeat Steps 1–4 with changing the forward direction to the backward direction. Otherwise, go to Step 6.
Step 6: A complete substroke corresponding to a graphical primitive is attained.

IV. EXTRACTION OF LINE SEGMENTS, CIRCULAR ARCS, AND THEIR RELATIONS

The segment length is critical to the overall performance since it directly affects the accuracy of resulting substrokes, the processing time and the boundary precision. Because the speed of pen varies when one sketches in a free way, each stroke is resampled using its minimum gap between original sample points. According to the relationship between the segment length and the substroke extraction performance (provided in Section V), we set the length of a segment to be 16 points, which achieves the best performance result. Furthermore, the thresholds for seed-segment growing are determined. The thresholds max_LSFE and max_GHs for line segments and circular arcs are calculated, respectively, after analyzing many samples. The thresholds for seed-segment detection are set as follows: For the line fitting, $\max_{\mathrm{LSFE}} = 4$, $\max\_\mathrm{GH}_\mathrm{k} = 0.01$, and $\max\_\mathrm{GH}_\mathrm{b} = 6$; for the circle fitting, $\max\_\mathrm{LSFE} = 4$, $\max\_\mathrm{GHx}_\mathrm{c} = 3$, $\max\_\mathrm{GHy}_c = 3$, and $\max\_\mathrm{GH}_\mathrm{r} = 4$.

A. Extraction of Line Segments and Circular Arcs Based on Seed-Segment Growing

We first identify a seed-segment using point-by-point detection in the stroke. After attaining a segment that satisfies two conditions of a seed-segment for line segments or for circular arcs, we set it as the seed-segment characterizing a substroke. Then, the LSFE, GHs, and GAs of the seed-segment are calculated. Thus, they are adaptive to the current seed-segment growing. For the line fitting process, $\varepsilon_\mathrm{L} = 2$, $\varepsilon_\mathrm{Gk} = 0.01$, $\varepsilon_\mathrm{Gb} = 2$, $\varepsilon_\mathrm{Sk} = 0.02$, and $\varepsilon_\mathrm{Sb} = 4$. For the circle fitting process, $\varepsilon_\mathrm{L} = 2$, $\varepsilon_\mathrm{Gxc} = 2$, $\varepsilon_\mathrm{Gyc} = 2$, $\varepsilon_\mathrm{Gr} = 3$, $\varepsilon_\mathrm{Sxc} = 3$, $\varepsilon_\mathrm{Syc} = 3$, and $\varepsilon_\mathrm{Sr} = 4$. Finally, we extract the substroke by growing the seed-segment using the algorithm stated in Section III-B. A group of connected internal segments and boundary segments are attained after the growing terminates. Therefore, the substroke is extracted as a group of connected homogeneous segments.

After extracting a substroke, we continue to scan the stroke for the next substroke by the above process until no seed-segment is detected from the stroke. Fig. 1(b) shows the detected substrokes from the strokes of Fig. 1(a), where the segments marked by solid rectangles and ellipses are the seed-segments of line segments and circular arcs, respectively. The internal segments and internal subsegments of boundary segments before a seed-segment are marked as dashed ones, those after the seed-segment marked as dotted ones. Although the substrokes contain some noisy points, they are all correctly detected since the extraction parameters are adaptive to each substroke. The substrokes $\mathrm{A'E''}$, $\mathrm{E''F''}$ and $\mathrm{F''B'}$ are extracted from stroke $\mathrm{A'B'}$, and substrokes $\mathrm{C'G''}$ and $\mathrm{G''D'}$ from stroke $\mathrm{C'D'}$ as shown in Fig. 1(b). If the fitted parameters of the seed-segment are used to achieve the corresponding graphics primitive, computational cost can be saved, but the not good result is attained. So, the fitted parameters of the detected segments and starting points and end points of the substroke are used to achieve the fitted result of the substroke. Fig. 1(c) shows the fitted results of the extracted substrokes in black overlapping the original strokes in green. The fitted line segments $\mathrm{A'E'''}$ and $\mathrm{H'''B'}$, and the fitted arcs $\mathrm{F'''G'''}$, $\mathrm{C'I'''}$ and $\mathrm{J'''D'}$ are shown in Fig. 1(c).

B. Determination of the Relations Between the Primitives

After extracting graphical primitives from a stroke, the relationship between two adjacent primitives can be determined using some rules [11].

1) If two adjacent primitives are both straight lines, and their inclinations is equal or close to 90°, the two lines are perpendicular and their relation is labeled as LLPer; otherwise, they intersect each other and their relation is labeled as LLI. The inclinations tolerance is set as $\pm 4°$ in terms of statistical analysis. We manually collected twenty samples that are pairs of perpendicular lines. Their inclinations are automatically calculated. The differences between 90 and the inclinations are calculated. The tolerance is set to be the maximum absolute value of all differences.
2) If they are both circular arcs, the distance between their centers is calculated. If the distance is equal or close to the sum of their radii, the two arcs are externally tangent to each other and their relation is labeled as AATE. The distance tolerance is set as ± 5 pixels in terms of statistical analysis. If the distance is equal or close to the difference of their radii, the two arcs are internally tangent to each other and their relation is labeled as AATI. The distance tolerance is set as ± 5 pixels in terms of statistical analysis. Otherwise, they intersect each other and their relation is labeled as AAI. We manually collected thirty samples that are externally tangent circles, circular arcs. Ten samples are ten pairs of circles, ten samples are ten pairs of circular arcs, and ten samples are ten pairs of circles and circular arcs. Their distances of pairs of centers are automatically calculated. Let r1 be the radius of one circle or arc, and r2 be the other, in a pair of circles or arcs. Let r12 be their sum. The differences between distances (between centers) and their r12s are calculated. The tolerance is set to be the maximum of all absolute differences. The distance tolerance for internally tangent is attained using the same way. But in this case, the differences of pairs of radiuses are used. The difference is between a bigger radius and a smaller one.
3) If one primitive is a straight line and the other is a circular arc, the distance from the center of the circular arc to the straight line is calculated. If the distance is equal or close to the radius of the circular arc, the straight line is tangential to the circular arc and their relation is labeled as LAT. Otherwise, they intersect each other and their relation is labeled as LAI. The distance tolerance is set as ± 5 pixels in terms of statistical analysis. We manually collected twenty samples that are pairs of a straight line and a circular arc with tangential relation. Their distances between centers and lines are automatically calculated. The tolerance is set to be the maximum of absolute differences between distances and radius.

The relationship between two not adjacent primitives can be determined using the following rules.

1) If they are both straight lines and the inclinations between them is equal or close to 180°, the two lines are parallel and their relation is labeled as LLPar. The inclinations tolerance is set as $\pm 4°$ in terms of statistical analysis. We manually collected twenty samples that are pairs of parallel lines. Their inclinations are automatically calculated. The differences between 180 and the inclinations are calculated. The tolerance is set to be the maximum absolute value of all differences.
2) If they are both circular arcs (circles), the distance between their centers is calculated. If the distance is smaller than a distance threshold, the two arcs (circles) are concentric and their relation is labeled as AAC, ACC, CAC, or CCC. The threshold is set as five pixels in terms of statistical analysis. We manually collected 20 samples that are pairs of concentric circles or arcs. Their distances between centers are automatically calculated. The tolerance is set to be the maximum of distances.

The line segments close to the horizontal or vertical direction are identified according to an inclination tolerance. The tolerance is set as $\pm 4°$ in terms of statistical analysis. We manually collected twenty samples that are horizontal line segments. Their inclinations to x axis are automatically calculated. The tolerance is set to be the maximum of the absolute values of inclinations. The vertical inclinations tolerance is set by analyzing line segments close to the vertical direction as the same way.

Control parameters of adjacent primitives in a stroke can be refined according to their relations and characteristics. The control parameters of a line are starting coordinates and end coordinates, while those of an arc are the coordinates of its center, radius, starting coordinates, end coordinates, starting angle to the $+\mathrm{X}$ axis (let the direction of arc be counter-clockwise), and end angle to the $+\mathrm{X}$ axis. They are adjusted with the following rules.

1) Horizontal and vertical lines are first adjusted. If a line is horizontal, then y values of two end points are averaged, the average value is used to replace y values of two end points. If a line is vertical, then x values of two end points are averaged, the average value is used to replace x values of two end points.
2) Each pair of intersection graphics primitives is then adjusted. Two adjacent end points of a pair of primitives are replaced using their intersection point.
3) Each pair of concentric circles or arcs or a circle and an arc is then adjusted. Two center points are replaced using their averaged point.
4) If a line is tangent to a circular arc, let D be the distance between the line and the center point of the arc, let R be the radius of the arc, let P be the perpendicular point from the center to the line. If the difference between D and R is 0, then the line is extended or\and shorten to P, and the arc also does so. If the difference is not 0, then the arc is moved with a minimum distance to a new center to make the D is equal to R. and then we extend or\and short the line or\and the arc. If an arc is tangent to a line, the line is moved in order to avoiding move its previous primitive.

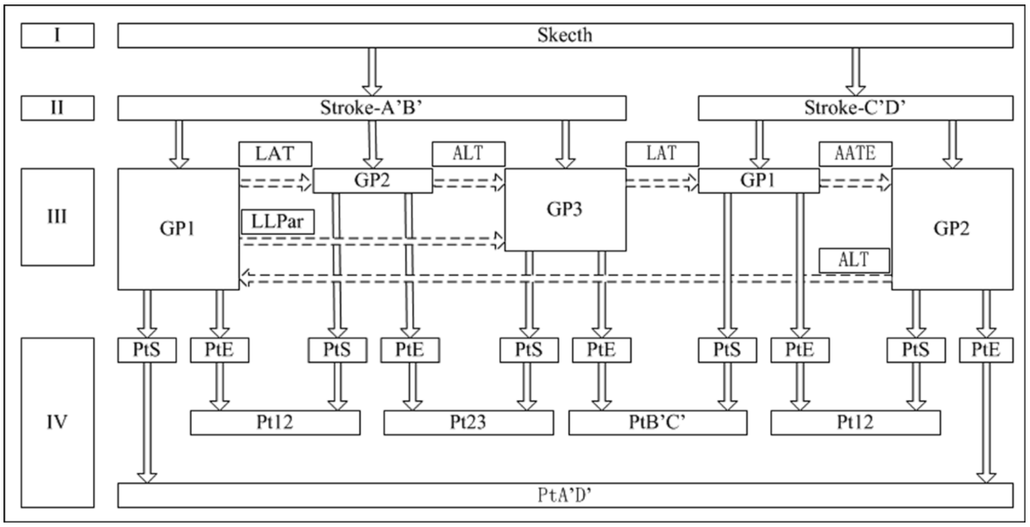


Fig. 3. Hierarchical network for the sketch shown in Fig. 1.

5) If an arc is external tangent to another arc, let R1 and R2 be their radiuses, and let D be the distance of their center points, let P be the tangent point. If the D is equal to the sum, then the arcs extended or shorten to P. If the D is not equal to the sum of R1 and R2, then the second arc is moved to a new center to make the D is equal to the sum of R1 and R2.
6) If an arc is internal tangent to another arc, let R1 and R2 be their radiuses, and let D be the distance of their center points, let P be the tangent point. If the D is equal to the difference of R1 and R2, then the arcs extended or shorten to P. If the D is not equal to the difference of R1 and R2, then the second arc is moved to a new center to make the D is equal to the difference.

After extracting the information of substrokes from each stroke, the relations between two adjacent boundary graphical primitives in two neighboring strokes with a smaller gap and those between two not adjacent primitives in different strokes can be determined using the above-described rules. The gap tolerance is set as four pixels according to statistical analysis. We manually collected twenty samples that are pairs of adjacent boundary graphical primitives in two neighboring strokes. Their distances between adjacent points of adjacent boundary graphical primitives are automatically calculated. The tolerance is set to be the maximum of distances. Two involved boundary graphical primitives are adjusted with rules 2, 4, and 5 mentioned above to adjust graphics primitives in strokes. Fig. 1(d) shows the refined graphical primitives in black for fitted ones of Fig. 1(c).

C. Constructing the Hierarchical Network of a Sketch for Compact Representation

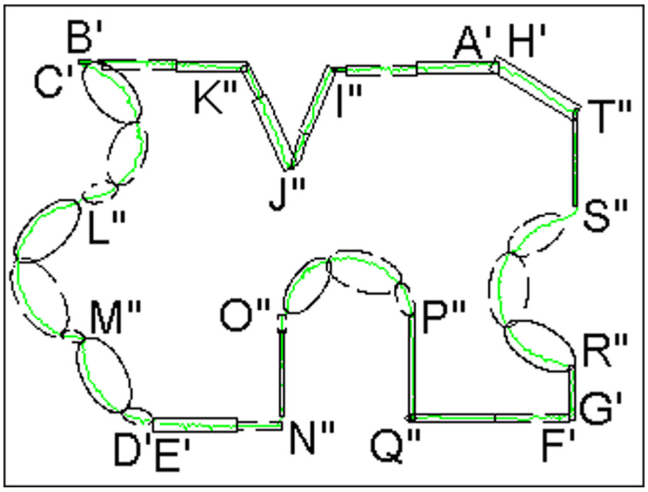
With substrokes' types, fitted parameters, control parameters, attributes, and the relations between graphical primitives, a stroke can, thus, be compactly represented using an associated network. In the network of a stroke, a node represents the type, fitted parameters, control parameters, and attributes of a substroke while a link between two nodes represents the relations between the two graphical primitives. The type of a substroke is either line or arc. The fitted parameters for a line include k and b, while those for a circle include $\mathrm{x_c}$, $\mathrm{y_c}$, and r. The control parameters for a line segment include starting coordinates and end ones, while those for a circular arc include center, radius, starting coordinates, end coordinates, starting angle, and end angle. The relations between two graphical primitives include: 1) LLI: nonperpendicular intersection between lines; 2) LLPer: perpendicular intersection between lines; 3) LLPar: parallel between straight lines; 4) AAI: intersection between arcs; 5) AATI: internal tangency between two arcs; 6) AATE: external tangency between two arcs; 7) AAC (ACC, CAC, or CCC): concentric between circular arcs (circles); 8) LAI (ALI): intersection between a line and an arc; 9) LAT (ALT): tangency between a line and an arc.

A sequence of strokes composes a sketch, which can, thus, be represented as a hierarchical structure after constructing the associated network of strokes. In the network, the top-level object is the sketch, which in turn consists of strokes and their relations locating the second level. Substrokes extracted from strokes become the third-level objects. Two adjacent graphical primitives in the same stroke share a common point, by which they are associated. The two adjacent strokes are also associated at a common boundary point. The fourth-level objects in the network are control points. Fig. 3 shows an associated network constructed from the sketch shown in Fig. 1.

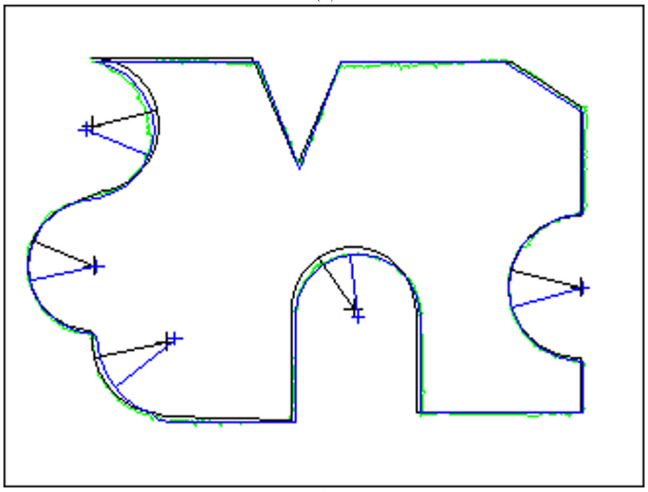
Therefore, the associated network of a sketch satisfies the requirement of a compact representation of a sketch as a bridge from stroke recognition to sketch understanding. It can provide sufficient information for extracting lexical, syntactic, and semantic information in subsequent phases for sketch understanding [22].

V. Experiments and Performance Analysis

Based on the proposed approach, a software prototype has been developed in Visual C++ R6.0 by authors. This section presents performance evaluation in terms of experimental results with real-life sketches and quantitative analyses based on ground-truth data.

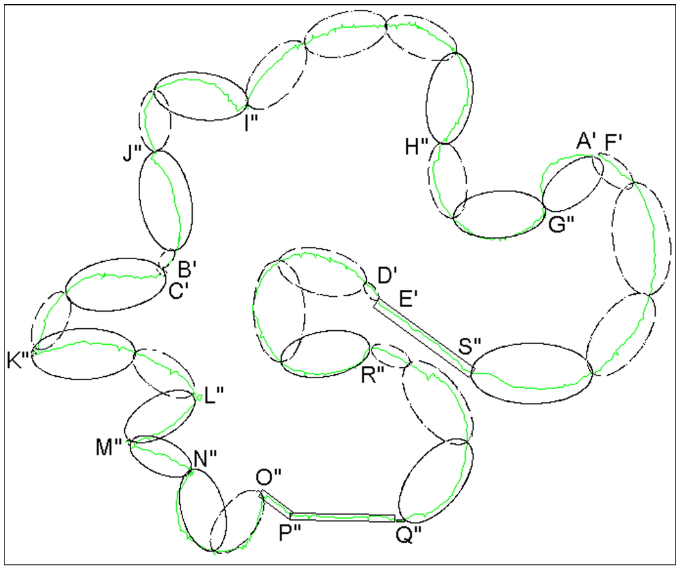


(a)

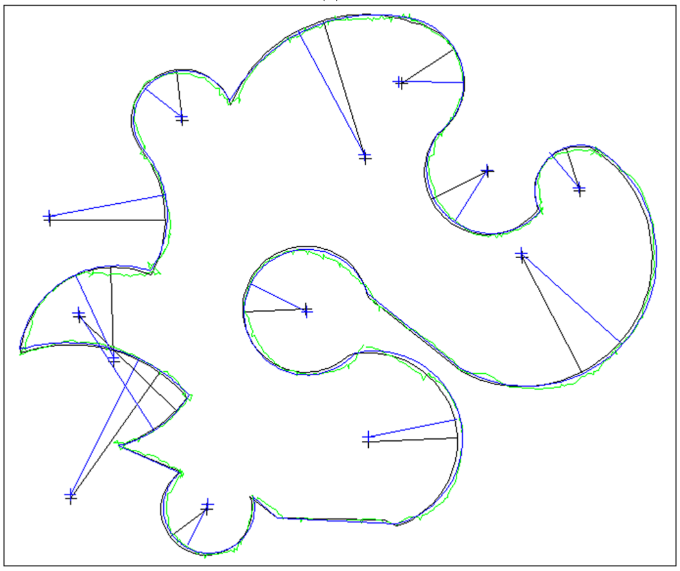


(b)

Fig. 4. Graphical primitives extracted from a sketch with part shape. (a) A sketch with four strokes $A'B'$, $C'D'$, $E'F'$, $G'H'$. Substrokes $A'I''$, $I''J''$, $J''K''$, $K''B'$ are extracted from stroke $A'B'$, $C'L''$, $L''M''$, $M''D'$ from $C'D'$, $E'N''$, NO'', $O''P''$, $P''Q''$, $Q''F'$ from $E'F'$, and $G'R''$, $R''S''$, $S''T''$, $T''H'$ from $G'H'$. (b) Refined, ground-truth graphical primitives of the sketch are labeled in black and blue, respectively. (Color version available online at http://ieeexplore.ieee.org.)



(a)



(b)

Fig. 5. Graphical primitives extracted from a sketch with flower shape. (a) A sketch with three strokes, $A'B'$, $C'D'$, $E'F'$. Substrokes $A'G''$, $G''H''$, $H''I''$, $I''J''$, $J''B'$ are extracted from $A'B'$, and $C'K''$, $K''L''$, $L''M''$, $M''N''$, $N''O''$, $O''P''$, $P''Q''$, $Q''R''$, $R''D''$ from $C'D'$, $E'S''$, $S''F'$ from $E'F'$. (b) Refined, ground-truth graphical primitives of the sketch are labeled in black and blue, respectively. (Color version available online at http://ieeexplore.ieee.org.)

A. Experimental Results

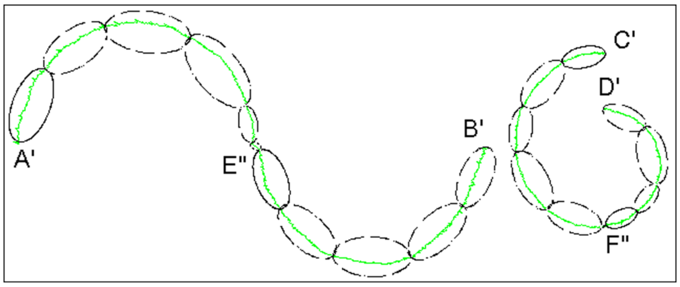
Many sketches have been tested on our software prototype. The SoundWay Pen [23], from Beijing ChineseStar Cyber Technology Limited, is used to sketch graphical objects on paper, and the point coordinates for each stroke are captured by the prototype. Pen positions are sampled at 60 points/s and at 100 dpi. More experimental results are shown in Figs. 4–6.

To make a quantitative evaluation on the extracted results, the ground-truth data are used as a reference. As the ground-truth graphical primitives cannot be obtained by any automatic processing, they [Fig. 1(d), Fig. 4(b), Fig. 5(b), and Fig. 6(b)] are manually detected by a professional engineer. They are displayed in blue color overlapping on the original stroke in green.

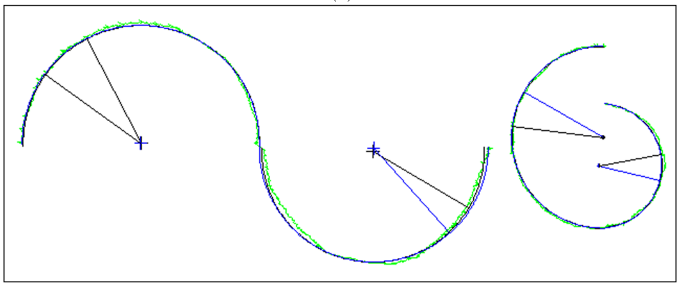
The discrepancy between the extracted graphical primitives and the ground-truth ones can be used to evaluate the performance of the proposed approach. A substantial disagreement of the object number indicates a large discrepancy between the ground-truth ones and the detected ones. The graphical primitive detection error (GPDE) indicating the relative percent of the wrongly detected graphical primitives over the ground-truth ones is employed. The GPDE is defined as

$$\mathrm{GPDE} = \frac{|N_r - N_c|}{N_r} \times 100\%$$

where N_c is the number of the correctly detected graphical primitives and N_r is the number of the detected ones. The smaller the GPDE is, the better the detection rate is. Distance errors of control points of detected primitives over those of reference ones also provide useful discrepancy measures. The maximum control point discrepancy $(\mathrm{CPD_M})$ characterizing the maximum



(a)



(b)

Fig. 6. Graphical primitives extracted from a sketch with externally and internally tangent arcs. (a) A sketch with two strokes, A′B′ and C′D′. Substrokes A′E″, E″B′ are extracted from A′B′, C′F″, F″D′ from C′D′. (b) Refined, ground-truth graphical primitives of the sketch are labeled in black and blue, respectively. (Color version available online at http://ieeexplore.ieee.org.)

distance error between the control points of detected primitives and those of ground-truth ones is employed. The $\mathrm{CPD_M}$ is defined as

$$\mathrm{CPD}_M = \max(D_i)$$

where D_i is the distance between a control point of a correctly detected graphical primitive and its corresponding control point of the ground-truth one. If a substroke is wrongly detected, D_i is the maximum distance between all points on a line segment or a circular arc detected and a circular arc or a line as referenced one. The smaller $\mathrm{CPD_M}$ is, the higher the extraction accuracy is.

The evaluation results of extracted graphical primitives, e.g., Fig. 1(d), Fig. 4(b), Fig. 5(b), and Fig. 6(b), compared with the ground-truth graphical primitives, e.g., Fig. 1(d), Fig. 4(b), Fig. 5(b), and Fig. 6(b), are given in Table II.

Table II demonstrates that the detection error and extraction discrepancy of the proposed approach is low. Our test data set not listed in Table II has 83 sketches containing 1023 strokes and 4320 substrokes. Actually, their GPDE and $\mathrm{CPD_M}$ are below 15% and eight pixels, respectively.

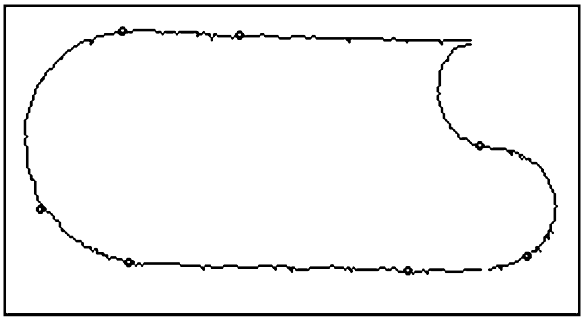
The processing time is another important performance index. Table II lists the processing time (tested on a PC with PIII 1150-Mhz CPU and 256 M RAM) of extracting graphical primitives in several sketches. Our test data set, maximum time for extracted primitives is smaller than 0.012 s. We find that the processing time is proportional to the number of graphical primitives, but not the number of strokes. This is because we use the scanning scheme to detect the seed-segments, which passes the noninteresting segments quickly.

TABLE II
EVALUATION OF OUR APPROACH

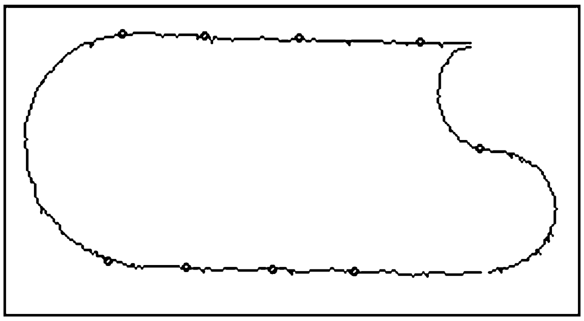
Item / Label	Number of strokes	Number of primitives	GPDE	$\mathrm{CPD_M}$ (pixels)	Time for each primitive (seconds)
1(Figure 1)	2	5	0%	7	0.008
2(Figure 4)	4	16	0%	6	0.011
3(Figure 5)	3	14	10%	6	0.009
4(Figure 6)	2	4	0%	4	0.005

TABLE III
PERFORMANCE PROFILES OVER DIFFERENT SEGMENT LENGTHS

Segment length (points)	GPDE	$\mathrm{CPD_M}$ (pixels)	Processing time (seconds)	Boundary scale (points)	SSP
13	0%	3	0.05	5	0.230
16	0%	3	0.04	6	0.290
19	0%	4	0.04	7	0.185
22	0%	5	0.05	8	0.100



(a)



(b)

Fig. 7. Segmentation point detection results using two other approaches. (a) Segmentation points, marked by small circles, identified with the speed-threshold approach. (b) Segmentation points, marked by small circles, identified with the distance sign change approach.

B. Segment Length Selection

The segment length is a major parameter in the proposed approach. It impacts the performance in three aspects: the accuracy of resulting graphical primitives, the processing time, and the boundary scale. Generally, a smaller segment (as long as it is) will increase accuracy of detected graphical primitives, but it takes much more processing time and is more sensitive to noisy points. Since the boundaries of detected substrokes are approximated using the segments and subsegments, smaller segments are sure to produce more precise boundaries. To determine which segment size is the best, we design a segment size priority (SSP) computation as follows:

$$\mathrm{SSP} = 1 - (\alpha \times \mathrm{Error} + \beta \times \mathrm{Time} + \gamma \times \mathrm{Boundary\ Scale})$$

TABLE IV
COMPARISON AMONG FIVE GRAPHICAL PRIMITIVE EXTRACTION APPROACHES

Approach / Characteristic	Speed-threshold approach [7]	Distance sign change approach [7]	Fuzzy logical approach [4, 5]	Multi-scale approach [9]	The proposed approach
Processing level	Point	A window of Points	Points	Point	Segment (A group of points)
Utilization of a priori knowledge	Little	Little	Much	Moderate	Moderate
Result type	Points	Points	Points	Points	Graphical primitives
Post processing burden	Much	Much	Much	Much	No
Noise sensitivity	High	High	High	Very low	Very low
Computational cost (seconds of each primitive)	0.003	0.004	0.007	0.02	0.012
Primitive recognition accuracy (%)	74	69	82	86	85
Primitive distance accuracy (pixels)	9	11	10	9	8

where α, β and γ are the performance weights of the substroke error, the processing time, and the boundary approximation scale, respectively. The higher the SSP, the higher the extraction accuracy. Since the aim of our application is to identify the substrokes and to fit them and determine their relations, the substroke accuracy is important. The processing time is also important to meet the rapid processing requirement for sketch recognition application. However, the boundary approximation scale is less important because small approximation errors do not affect the fitting parameter estimation and relationship determination. Therefore, we set $\alpha = 0.4$, $\beta = 0.4$, and $\gamma = 0.2$ in our segment size selection. When calculating SSP, each performance value is divided by the maximum of its column to be normalized into [0,1].

Table III shows the performance profiles for Fig. 1 when the subsegment length varies from five to eight points, where the subsegment length must be larger than three since at least three points not lying on the same line determine a circle. The boundary approximation scale is the same as the length of a subsegment. The result is that the SSP corresponding to six points reaches the highest. Many other profiles of sketches validate this. Therefore, we set the subsegment length to be six points in our approach, and the segment length is 16 points.

C. Comparisons With Other Approaches

Fig. 7 shows graphical primitive extraction results using two other methods. Fig. 7(a) shows the segmentation points detected from the sketch of Fig. 1(a) using the speed-threshold approach [7]. In order to extract the substrokes correctly, a post-processing is necessary to delete some false positives. For example, Davis [6] uses an average-based filter to reduce false positives. But it is difficult to choose an appropriate filter that will achieve satisfactory results for all strokes in a sketch. Fig. 7(b) shows the segmentation points detected from the sketch of Fig. 1(a) using the distance sign change approach [7]. A post-processing is performed to delete some false positives. Due to the point-level operation, the two substrokes contain some noisy points and have less accurate boundaries. Comparing with the graphical primitives shown in Fig. 1(d), we find that our proposed approach produces more satisfactory results.

The proposed approach and four other approaches are further compared in eight aspects, including processing level, utilization of *a priori* knowledge, result type, post processing burden, noise sensitivity, computational cost, primitive recognition accuracy, and primitive distance accuracy. The other four approaches are the speed-threshold approach [7], the distance sign change approach [7], the fuzzy logical approach [4], [5], and the multiscale-based approach [9]. The comparison results among these approaches and our proposed approach are listed in Table IV, which shows that our proposed approach possesses more advantages than other approaches, especially in the substroke accuracy and the noise insensitivity. Therefore, the proposed approach can achieve satisfactory results for extracting line segments and circular arcs in strokes, even in noisy conditions.

D. Discussions

With the experimental results and performance analyses, we conclude that the proposed approach has four major advantages.

1) Using segmental homogeneity features instead of using the local feature of points, this approach is neither sensitive to the inhomogeneous graphical features in the point level nor sensitive to the local noises.
2) The segment growing is able to directly obtain the entire substroke; therefore, no post processing is needed.
3) The extraction thresholds are adaptive to each substroke; therefore, the proposed approach can extract different types of substrokes in the same stroke.
4) The control parameters of primitives are attained using fitted results and their relations. So, the accuracy of graphical primitive extraction is greatly improved and a

perfect visual representation for rough sketches can be attained.

VI. Conclusion

This paper introduces the segmental homogeneity feature, including the LSFE, the graphical attributes, and the graphical homogeneity, to improve the adaptability to the graphical nonhomogeneity in the point level. This paper further proposes a seeded-segment growing algorithm based on the segmental graphical homogeneity feature. With this algorithm, one can successfully detect substrokes in their entireties even with the interference of noisy points. The proposed approach is applied to the extraction of graphical primitives corresponding to substrokes in strokes. Identifying graphical primitives in the segment level rather than in the point level improves the extraction accuracy and efficiency. The accurate substrokes act as the starting points for their fitting. Fitted results and their relations are used to calculate the control parameters of graphical primitives, which further improves accuracy. Complete information of graphical primitives is finally engaged to construct a hierarchical network for a sketch.

We have tested the proposed approach using many real-life sketches on our software prototype. The performance analyses are reported in detail, including the experimental results, the performance evaluation, and the comparison with other methods. The analyses confirm that the proposed approach is effective and robust.

References

[1] P. Agar and K. Novins, "Polygon recognition in sketch-based interfaces with immediate and continuous feedback," in *Proc. 1st Int. Conf. Computer Graphical and Interactive Techniques in Australasia and South East Asia*, Melbourne, Australia, Feb. 2003, pp. 147–150.

[2] L. Schomaker, "From handwriting analysis to pen-computer applications," *Electron. Commun. Eng. J.*, vol. 10, no. 3, pp. 93–102, 1998.

[3] A. Apte, V. Vo, and T. D. Kimura, "Recognizing multi-stroke geometric shapes: An experimental evaluation," in *Proce. ACM Symp. User Interface Software and Technology*, Atlanta, GA, Nov. 1993, pp. 121–128.

[4] S. F. Qin, D. K. Wright, and I. N. Jordanov, "On-line segmentation of freehand sketches by knowledge-based nonlinear thresholding operations," *Pattern Recognit.*, vol. 34, no. 10, pp. 1885–1893, 2001.

[5] ——, "From on-line sketching to 2-D and 3-D geometry: A system based on fuzzy knowledge," *Comput.-Aided Design*, vol. 32, no. 14, pp. 851–866, 2000.

[6] R. Davis, "Sketch understanding in design: Overview of work at the MIT Lab," in *Proc. Amer. Assoc. Artificial Intelligence Spring Symp.*, Palo Alto, CA, Mar. 2002, pp. 24–31.

[7] C. Calhoun, T. F. Stahovich, T. Kurtoglu, and L. B. Kara, "Recognizing multi-stroke symbols," in *Proc. Amer. Assoc. Artificial Intelligence Spring Symp.*, Palo Alto, CA, Mar. 2002, pp. 15–23.

[8] T. M. Sezgin, T. Stahovich, and R. Davis, "Sketch based interfaces: early processing for sketch understanding," in *Proc. Perceptive User Interfaces Workshop*, Lake Buena Vista, FL, Nov. 2001, pp. 1–8.

[9] T. M. Sezgin, "Feature Point Detection and Curve Approximation for Early Processing of Freehand Sketches," M.S. thesis, Dept. EECS, Mass. Inst. Technol., Cambridge, May 2001.

[10] M. Shpitalni and H. Lipson, "Classification of sketch strokes and corner detection using conic sections and adaptive clustering," *Trans. ASME J. Mech. Design*, vol. 119, no. 2, pp. 131–135, 1997.

[11] K. Z. Chen, X. W. Zhang, Z. Y. Ou, and X. A. Feng, "Recognition of digital curves scanned from paper drawings using genetic algorithms," *Pattern Recognit.*, vol. 36, no. 1, pp. 123–130, 2003.

[12] J. Song, F. Su, C. Tai, and S. Cai, "An object-oriented progressive-simplification based vectorization system for engineering drawings: model, algorithm and performance," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 8, pp. 1048–1060, Aug. 2002.

[13] M. Marji and P. Siy, "A new algorithm for dominant points detection and polygonization of digital curves," *Pattern Recognit.*, vol. 36, no. 10, pp. 2239–2251, 2003.

[14] G. Dudek and J. K. Tsotsos, "Shape representation and recognition from multiscale curvature," *Comput. Vis. Image Understand.*, vol. 68, no. 2, pp. 170–189, 1997.

[15] P. L. Rosin and G. A. W. West, "Nonparametric segmentation of curves into various representations," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 17, no. 12, pp. 1140–1153, Dec. 1995.

[16] N. S. Netanyahu, V. Philomin, A. Rosenfeld, and A. J. Stromberg, "Robust detection of straight and circular road segments in noisy aerial images," *Pattern Recognit.*, vol. 30, no. 10, pp. 1673–1686, 1997.

[17] H.-T. Sheu and W.-C. Hu, "Multiprimitive segmentation of planar curves—a two-level breakpoint classification and tuning approach," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 21, no. 8, pp. 791–797, Aug. 1999.

[18] C. Ichoku, B. Deffontaines, and J. Chorowicz, "Segmentation of digital plane curves: a dynamic focusing approach," *Pattern Recognit. Lett.*, vol. 17, pp. 741–750, 1996.

[19] A. Albano, "Representation of digitized contours in terms of conic arcs and straight-line segments," *Comput. Graph., Image Process.*, vol. 23, pp. 860–870, 1974.

[20] T. Igarashi, S. Matsuoka, S. Kawachiya, and H. Tanaka, "Interactive beautification: a technique for rapid geometric design," in *Proc. 10th Annu. ACM Symp. User Interface Software and Technology*, Banff, AB, Canada, Oct. 1997, pp. 105–114.

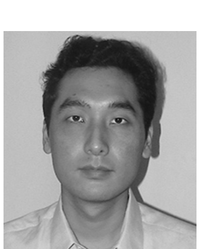
[21] J. A. Landay and B. A. Myers, "Sketching interfaces: Toward more human interface design," *IEEE Computer*, vol. 34, no. 3, pp. 56–64, Mar. 2001.

[22] R. W. Ferguson and K. D. Forbus, "A cognitive approach to sketch understanding," in *Proc. Amer. Assoc. Artificial Intelligence Spring Symp.*, Palo Alto, CA, Mar. 2002, pp. 45–50.

[23] Beijing ChineseStar Cyber Technology, Ltd.*http://www.cstar.com.cn/soundway/shwb.asp* [Online]

Xiwen Zhang (M'01) received the B.S. degree in chemical equipment and mechanics from the Fushun Institute of Petrol (the Liaoning University of Petroleum and Chemical Technology since March 2002), Fushun, China, in 1995, and the Ph.D. degree in mechanical manufacturing and automation from the Dalian University of Technology, Dalian, China, in 2000.

Currently, he is a Postdoctoral Fellow with the Department of Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong. Since October 2002, he has been with Institute of Software, The Chinese Academy of Sciences, Beijing, and has been an Associate Research Professor since January 2003. From October 2000 to September 2002, he was a Postdoctorate in computer science and technology at the State Key Laboratory of Novel Software Technology, Department of Computer Science and Technology, Nanjing University, Nanjing, China. His research interests include human computer interaction, sketch understanding, computer vision, artificial intelligence, medical image analysis, and understanding of scanned engineering drawings.



Jiqiang Song (S'99–M'02) received the B.S. degree in computer science and the Ph.D. degree in computer science and applications from Nanjing University, Nanjing, China, in 1996 and 2001, respectively.

He is currently a member of Professional Staff of the Hong Kong Applied Research and Technology Institute, Hong Kong. He was a Postdoctoral Fellow with the Department of Computer Science and Engineering, The Chinese University of Hong Kong, from 2001 to 2004. His research interests include graphics recognition, automatic interpretation of engineering drawings, and image/video processing and video compression, and he has published over 30 papers in these areas.

Guozhong Dai graduated from the Department of Application Mathematics, University of Science and Technology of China, Beijing, in 1967.

From 1982 to 1985, he was a Visiting Scholar at the University of Maryland, College Park. He is currently a Research Professor at Institute of Software, The Chinese Academy of Sciences, Beijing, where he is an expert member of Chinese High-Tech Program. His major research interests cover human computer interaction and software engineering. He pioneered the HCI research in China. He has been working on multimodal user interfaces, especially pen-based user interfaces, for more than ten years, and has extensive experience in this area. So far, he holds one patent and has published more than 100 papers.

Dr. Dai has won many awards from the Chinese Government for his excellence in HCI research. He is the Chairman of SIGCHI China. He was the program Chair for the 5th Asia Pacific Conference on Computer Human Interaction (APCHI 2002) and has served on program committees for many conferences, including the 8th World Conferences on Integrated Design and Process Technology in 2005, the First International Conference on Affective Computing and Intelligent Interaction in 2005, the 6th Asia-Pacific Conference on Computer-Human Interaction Incorporating the 5th ACM SIGCHI-NZ Symposium (APCHI 2004), Virtual Reality and its Application in Industry (VRAI2003), ACM Symposium on Virtual Reality Software and Technology 2002, and he will serve on the program committee for the 2006 International Conference on Intelligent User Interfaces.

Michael R. Lyu (S'84–M'88–SM'97–F'04) received the B.S. degree in electrical engineering from National Taiwan University, Hsinchu, Taiwan, R.O.C., in 1981, the M.S. degree in computer engineering from University of California, Santa Barbara, in 1985, and the Ph.D. degree in computer science from University of California, Los Angeles, in 1988.

He is currently a Professor at the Department of Computer Science and Engineering, The Chinese University of Hong Kong, Shatin, NT, Hong Kong. He is also the Director of the Video Over Internet and Wireless (VIEW) Technologies Laboratory. He was with the Jet Propulsion Laboratory, Pasadena, CA, as a Technical Staff Member from 1988 to 1990. From 1990 to 1992, he was with the Department of Electrical and Computer Engineering, The University of Iowa, Iowa City, as an Assistant Professor. From 1992 to 1995, he was a Member of the Technical Staff in the applied research area of Bell Communications Research (Bellcore), Morristown, NJ. From 1995 to 1997, he was a Research Member of the Technical Staff at Bell Laboratories, Murray Hill, NJ, which was first part of AT&T and later became part of Lucent Technologies. His research interests include software reliability engineering, distributed systems, fault-tolerant computing, wireless communication networks, Web technologies, digital libraries, and E-commerce systems. He has published over 200 refereed journal and conference papers in these areas. He has participated in more than 30 industrial projects, and helped to develop many commercial systems and software tools. He was the editor of two book volumes: *Software Fault Tolerance* (New York: Wiley, 1995) and *The Handbook of Software Reliability Engineering* (Piscataway, NJ: IEEE and New York: McGraw-Hill, 1996). He has been frequently invited as a keynote or tutorial speaker to conferences and workshops in U.S., Europe, and Asia. He is an Associate Editor of the *Journal of Information Science and Engineering.*

Dr. Lyu received the ISSRE Best Paper Awards in 1998 and 2003, respectively. He served on the Editorial Board of IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, and has been an Associate Editor of IEEE TRANSACTIONS ON RELIABILITY. He initiated the First International Symposium on Software Reliability Engineering (ISSRE) in 1990. He was the program Chair for ISSRE1996, and has served in program committees for many conferences, including ISSRE, SRDS, HASE, ICECCS, ISIT, FTCS, DSN, ICDSN, EUROMICRO, APSEC, PRDC, PSAM, ICCCN, ISESE, and WWW. He was the General Chair for ISSRE2001, and the WWW10 Program Co-Chair. He has been frequently invited as a keynote or tutorial speaker to conferences and workshops in the U.S., Europe, and Asia.