6-1999

On the Geometries of Conic Section Representation of Noisy Object Boundaries

Qiuming Zhu
University of Nebraska at Omaha, qzhu@unomaha.edu

On the Geometries of Conic Section Representation of Noisy Object Boundaries

Qiuming Zhu

Digital Imaging and Computer Vision Laboratory, Department of Computer Science, University of Nebraska at Omaha, Omaha, Nebraska 68182
E-mail: zhuq@unomaha.edu

This paper studies some geometrical properties of conic sections and the utilization of these properties for the generation of conic section representations of object boundaries in digital images. Several geometrical features of the conic sections, such as the chord, the characteristic point, the guiding triangles, and their appearances under the tessellation and noise corruption of the digital images are discussed. The study leads to a noniterative algorithm that takes advantage of these features in the process of formulating the conic section parameters and generating the approximations of object boundaries from the given sequences of edge pixels in the images. The results can be optimized with respect to certain different criteria of the fittings.

1. INTRODUCTION

It has long been known that information about the object shapes is largely conveyed via the curving of the object's boundary edges [27]. Human vision systems rely heavily on the use of the boundary shapes to recognize objects. Many machine vision systems also adopted the same strategy to perform most of the object recognition tasks. Finding an appropriate set of simple and regular curve segments to describe the geometrical appearances of object boundaries in digital images thus is often an indispensable step in a computer vision process. The objective of the step is to use a minimum number of curve pieces to approximate the object's boundary edges with minimum distortion, thus enabling a precise and accurate analysis of the boundaries of the objects.

Modeling object boundaries by conic sections or other kind of curve approximations has been studied by computer vision researchers for many years under the general topic of curve fitting [1, 2, 5, 12, 17–19, 22, 24]. Higher-order curve approximations, such as the cubic bezier and B-splines [3, 8, 13, 21] curves are commonly used. Though a smooth and precise boundary model results, these higher-order curves require complex expressions and intensive computation. They are therefore more often used in computer-aided design

applications. On the other hand, conic section representation of object boundaries has the advantage of simplicity, popularity, and efficiency [9, 14, 20, 23]. The conic sections are, therefore, the most frequently used curve forms, besides the piece-wise straight line segments, used for approximating the object boundaries in computer vision applications.

Conic curves fall into three classes, namely hyperbolas, parabolas, and ellipses (considering the circle as a special case of ellipse). A conic curve is described on a planar surface in an algebraic expression

$$f(x, y) = ax^2 + by^2 + 2hxy + 2ux + 2vy + d = 0. \tag{1.1}$$

A conic section is often defined on Eq. (1.1) with two terminating points $\mathbf{X_0} = (x_0, y_0)$ and $\mathbf{X_1} = (x_1, y_1)$ additionally specified, where $x_0 \le x \le x_1$, $y_0 \le y \le y_1$. Equation (1.1) can be written in matrix form as

$$f(x, y) = \mathbf{X}\mathbf{Q}\mathbf{X}^{\mathrm{t}} = 0, \tag{1.2}$$

where $\mathbf{X} = [x, y, 1]$ and the $\mathbf{X}^{\mathrm{t}}$ denotes the transpose of $\mathbf{X}$. We have in Eq. (1.2) $\mathbf{X_0} \le \mathbf{X} \le \mathbf{X_1}$ and

$$\mathbf{Q} = \begin{bmatrix} a & h & u \\ h & b & v \\ u & v & d \end{bmatrix}. \tag{1.3}$$

It is required that the $\mathbf{Q}$ be a nonsingular matrix for $f(x, y)$ to be a valid conic section. That is, the conditions $\det(\mathbf{Q}) \neq 0$ and $(a+b) \neq 0$ must be satisfied. This leads to an important fact that, although an algebraic equation of the form (1.1) can always be derived from solving the simultaneous equations defined on five distinct data points in a 2D space, the result does not necessarily represent a valid conic curve, or a section [15, 16]. The problem is especially significant to the approximation or recovering of conic sections on object boundaries in digital images. It is because the edge pixels on object boundaries are tessellated in discrete positions of integer values in a digital image. In this tessellation, a large percentage of the points on the curve are displaced from their original positions defined in the mathematical expressions. A valid conic section thus is not always guaranteed from the solutions of Eq. (1.1) with respect to five arbitrarily picked points from the edge pixels that are even on the conical boundary of an object. Though the edges seem smoother in the higher resolution images, as those shown in Fig. 1.1, the increase of image resolution would not eliminate this problem.

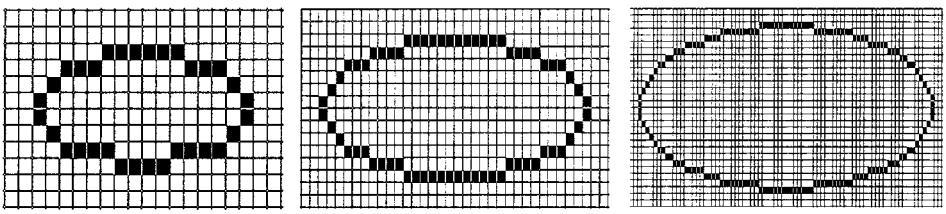

FIG. 1.1. Curve distorted because of pixel tessellation in images: (a) an ellipse in a 20×12 resolution; (b) an ellipse in a 40×16 resolution; (c) an ellipse in a 128×32 resolution.

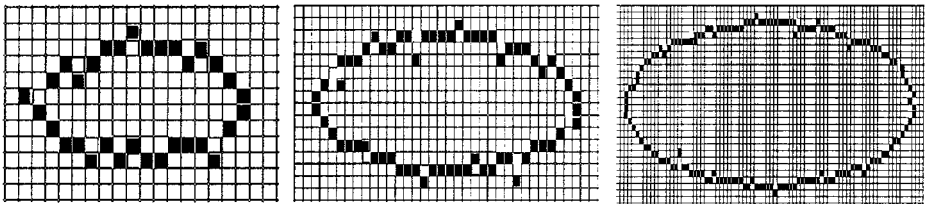

FIG. 1.2. Points on a conic curve are displaced in images because of noise: (a) an ellipse in a 20×12 resolution, (b) an ellipse in a 40×16 resolution; and (c) an ellipse in a 128×32 resolution.

Another problem needs to be considered is the displacement of the edge pixels caused by the noise corruption of the image. The situation is illustrated in Fig. 1.2, where edge pixels are further deviated from their real curve positions because of the effect of noises.

The fact that image pixels on an object boundary are tessellated in integer values and noise displacement makes it difficult and sometimes impossible to identify the conic sections by directly solving the algebraic equations of (1.1), it is therefore necessary to have some algorithms that capture the essential features of the conic sections under tessellation and noise, so their expressions can be derived by applying a certain approximation technique. Several techniques on these aspects were reported. Techniques based on the least-square-error fitting were the most commonly studied ones [6, 14, 17, 25]. Iterative procedures were usually applied in these techniques to derive the parameters of the curve sections from the processes of gradually reducing the fitting errors between the given set of pixels and the resulting curves. Several authors have pointed out the problem of statistical biases inherent in the least-square types of conic fitting [11, 12]. To remedy the problem, Kanatani [11] suggested applying a renormalization technique based on a statistical model of noises. However, problems with respect to the computational complexity and the discontinuity of the resulting object boundaries remain the same. Techniques based on Hough transformation [4, 10] provide an alternative. The techniques tried to fit the boundaries by first converting the edge pixels to a parametric space and then derive the curve parameters from a statistical account of the edge pixel distributions in the space. The major problem with these techniques is the precision of the fitted curve sections that is constrained by the dimension and resolution of the parametric spaces. The scale-up generality of these approaches is therefore very limited.

In this paper, we start with a study of some geometrical properties of the conic sections. Based on the study, an approach for approximating the object boundaries that utilizes some of the specific geometrical properties of the conic sections is described. The paper is organized as follows. Section 2 examines the specific geometrical properties of conic sections. Section 3 discusses how these properties can be used in deriving the feature parameters of conic sections from given sets of image pixels. A generalized guiding triangle (GGT) approach is described. Algorithms for generate the conic approximation of object boundaries are presented in Section 4. Section 5 presents some experimental examples. Section 6 contains conclusion remarks. Proofs of some of the geometrical properties of the conic sections and their applicability in the approximation of object boundaries are included in the appendices.

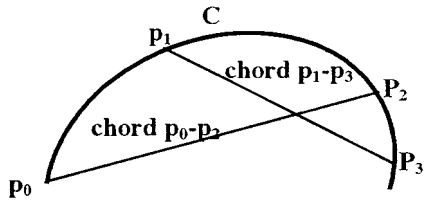

FIG. 2.1. Chords $\mathbf{p_0}$–$\mathbf{p_2}$ and $\mathbf{p_1}$–$\mathbf{p_3}$ on a conic section C.

2. THE CHORD, GUIDING TRIANGLE, AND SHOULDER POINTS OF CONIC SECTIONS

1. *The Chord*

Given any two points on a conic section, say $\mathbf{p_0}$ and $\mathbf{p_1}$, a *chord* is the line segment that connects these two points. Denote the chord as $\mathbf{p_0}$–$\mathbf{p_1}$, it has the property that points on the conic between the $\mathbf{p_0}$ and $\mathbf{p_1}$ all lay at one side of the $\mathbf{p_0}$–$\mathbf{p_1}$. Figure 2.1 shows the chords $\mathbf{p_0}$–$\mathbf{p_2}$ and $\mathbf{p_1}$–$\mathbf{p_3}$ of a conic section C.

Arbitrarily choosing one point $\mathbf{p_0}$ on a conic section and constructing chords that connect $\mathbf{p_0}$ to its consequent points $\mathbf{p_1}, \mathbf{p_2}, \mathbf{p_3}, \ldots, \mathbf{p_n}$ on the conic section, we have a sequence of chords, named as $\mathbf{p_0}$–$\mathbf{p_1}$, $\mathbf{p_0}$–$\mathbf{p_2}$, $\mathbf{p_0}$–$\mathbf{p_3}$, $\ldots$, $\mathbf{p_0}$–$\mathbf{p_n}$. Let the symbol "$\ll$" stands for the geometrical relation "left-of" and "$\gg$" stands for "right-of" between any two chords viewed in a clockwise direction. We see that these chords form either one of two patterns:

(i) in clockwise direction: $\mathbf{p_0}$–$\mathbf{p_1} \ll \mathbf{p_0}$–$\mathbf{p_2} \ll \mathbf{p_0}$–$\mathbf{p_3} \ll \cdots \ll \mathbf{p_0}$–$\mathbf{p_n}$, or (2.1)

(ii) in counterclockwise direction: $\mathbf{p_0}$–$\mathbf{p_1} \gg \mathbf{p_0}$–$\mathbf{p_2} \gg \mathbf{p_0}$–$\mathbf{p_3} \gg \cdots \gg \mathbf{p_0}$–$\mathbf{p_n}$. (2.2)

They are illustrated in Fig. 2.2

2. *The Guiding Triangle*

It is known that any conic section can be defined by a *guiding triangle* (GT) [14] such as the one shown in Fig. 2.3, where $\mathbf{T}_c$ is the guiding triangle defined by the vertices $\mathbf{p_0}$, $\mathbf{p_1}$, and $\mathbf{p_2}$. A conic section C is defined within the guiding triangle $\mathbf{T}_c$ with $\mathbf{p_0}$ and $\mathbf{p_2}$ being its two end points. The two sides $\mathbf{p_0}$–$\mathbf{p_1}$ and $\mathbf{p_2}$–$\mathbf{p_1}$ of $\mathbf{T}_c$ are the tangent vectors of C at the $\mathbf{p_0}$ and $\mathbf{p_2}$ points, respectively. We denote the vectors $\mathbf{p_0}$–$\mathbf{p_1}$ and $\mathbf{p_1}$–$\mathbf{p_2}$ as $\mathbf{p_0^u}$ and $\mathbf{p_2^u}$, and we call the vector $\mathbf{p_0}$–$\mathbf{p_2}$ the baseline of the guiding triangle as well as the baseline of the conic section.

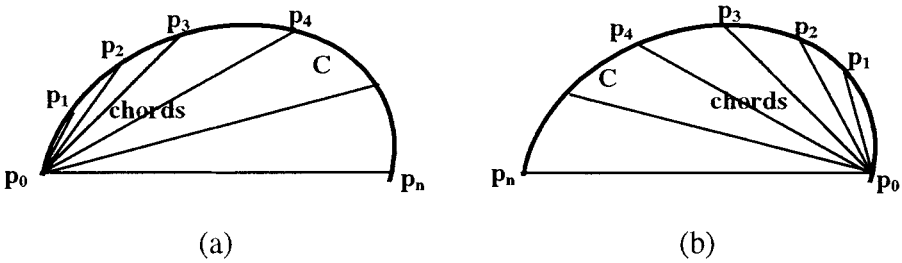

FIG. 2.2. Chords on a conic section C: (a) in the clockwise direction and (b) in the counterclockwise direction.

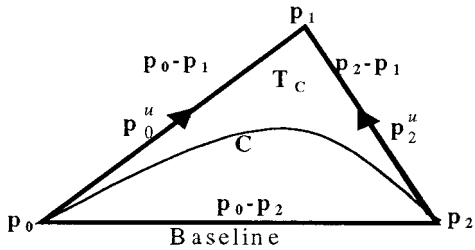

FIG. 2.3. A guiding triangle $\mathbf{T}_c$ defined by vertices $\mathbf{p_0}$, $\mathbf{p_1}$, and $\mathbf{p_2}$ for conic section C.

The geometrical expression of the conic section C can be expressed in a rational quadratic form [6] as

$$C(t) = \frac{B_0(t)\mathbf{p_0} + B_1(t)w\mathbf{p_1} + B_2(t)\mathbf{p_2}}{B_0(t) + B_1(t)w + B_2(t)}, \quad 0 \le t \le 1, \tag{2.3}$$

where $B_0(t) = (1-t)^2$, $B_1(t) = 2(1-t)t$, and $B_2(t) = t^2$, are the Bernstein basis functions [6, 8]. The parameter w of the expression determines the conic classes and controls the "sharpness" of the conic. The conic section is an ellipse when $w < 1$, a parabola when $w = 1$, and a hyperbola when $w > 1$, as shown in Fig. 2.4.

The equation for a conic section defined within a guiding triangle can also be expressed in the algebraic form

$$(a_x x + a_y y + a_0)(b_x x + b_y y + b_0) = K(u_x x + u_y y + u_0)^2, \tag{2.4}$$

where the line $a_x x + a_y y + a_0 = 0$ corresponds to the vector $\mathbf{p_0}$–$\mathbf{p_1}$, $b_x x + b_y y + b_0 = 0$ to the vector $\mathbf{p_2}$–$\mathbf{p_1}$, and $u_x x + u_y y + u_0 = 0$ to the vector $\mathbf{p_0}$–$\mathbf{p_2}$, as they are shown in Fig. 2.5a. The expression (2.4) can actually be directly derived from the geometrical expression (2.3) by some mathematical manipulations. Notice that when the point (x, y) is on vector $\mathbf{p_0}$–$\mathbf{p_2}$, K becomes infinite and the curve degenerates to a straight line coincident to $\mathbf{p_0}$–$\mathbf{p_2}$. The parameter K in the algebraic expression (2.4) has the similar role as the w of the geometrical expression except its value ranges are different. For example, when other conditions are satisfied, for $K = a_y b_y / u_y^2$ the conic is a parabola, $K > a_y b_y / u_y^2$ is an ellipse, and $K < a_y b_y / u_y^2$ is a hyperbola, as shown in Fig. 2.5b.

3. *The Shoulder Point*

In the geometrical expression (2.3) of the conic section, the parameter $t = 0$ represents the end point $\mathbf{p_0}$ and $t = 1$ is the point $\mathbf{p_2}$. Let $\mathbf{p_v}$ be a point on the conic, where the

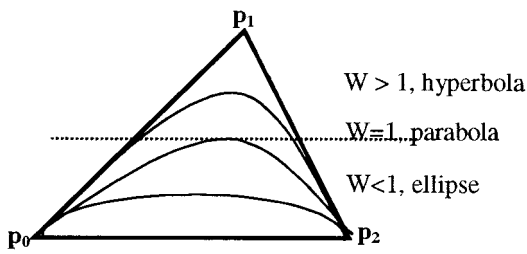

FIG. 2.4. Different types of conic section with respect to parameter w.

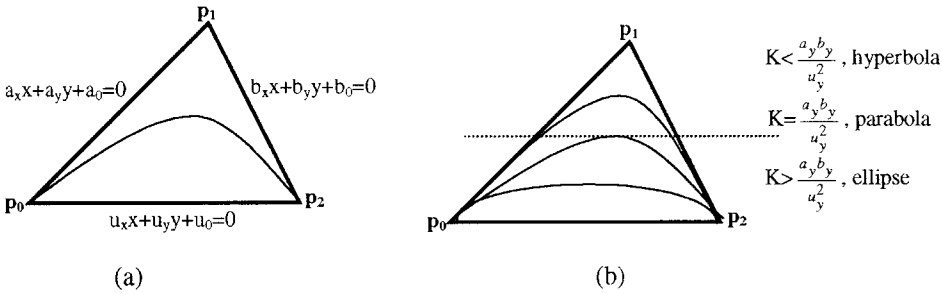

FIG. 2.5. Guiding triangle of conic section in algebraic expression: (a) the annotation of the sides of the triangle and (b) the determination of the conic shapes with respect to the values of K.

parameter $t = 1/2$. This point $\mathbf{p_v}$ is called the *shoulder point*, or the characteristic point [6]. The shoulder point $\mathbf{p_v}$ has the geometrical properties that:

(1) It is the point on a conic section, where the parameter t in its geometrical expression equals $1/2$.

(2) The tangent vector $\mathbf{p_v^u}$ of this point is parallel to the baseline $\mathbf{p_0}$–$\mathbf{p_2}$,

(3) It has the largest distance from it to the baseline $\mathbf{p_0}$–$\mathbf{p_2}$ among all the points on the conic section,

(4) A triangle formed by $\mathbf{p_0}$, $\mathbf{p_2}$, and $\mathbf{p_v}$ has the largest area among all triangles that are formed by $\mathbf{p_0}$, $\mathbf{p_2}$, and a point on the conic section,

(5) It is also on the line $\mathbf{p_1} - (\mathbf{p_0} + \mathbf{p_2})/2$, the line that connects the vertex $\mathbf{p_1}$ and the middle point of the baseline $\mathbf{p_0}$–$\mathbf{p_2}$.

The proof of these properties is included in Appendix A.

3. FROM CONIC GEOMETRIES TO OBJECT BOUNDARIES

1. Pixel Rectification

In discussions of the following, we assume that an edge detection and thinning operation has been applied to the digital images so that the object boundaries are represented in edge segments of one pixel width [28]. The edge segments are distinguished by those called break points such as intersections, corners, and inflection points [7, 26]. Conic sections are then constructed on the edge segments between these break points.

It is possible to apply the conic section's *chord* property to identify the break points as well as the extremely displaced pixels on a given set of edge pixel sequence in a digital image. As we have seen in last section, pixels on a given conic section should always reside at the same side of a chord. Let $\mathbf{p_0}$, $\mathbf{p_1}$, $\mathbf{p_2}$, and $\mathbf{p_3}$ be four pixels in a sequence on an edge segment, where $\mathbf{p_3}$ is the head and $\mathbf{p_0}$ is the tail. Denote the chords as $\mathbf{l_1} = \mathbf{p_0}$–$\mathbf{p_1}$, $\mathbf{l_2} = \mathbf{p_0}$–$\mathbf{p_2}$, and $\mathbf{l_3} = \mathbf{p_0}$–$\mathbf{p_3}$. According to the chord property of Section 2, we should have only two situations: (1) $\mathbf{l_1} \ll \mathbf{l_2} \ll \mathbf{l_3}$ and (2) $\mathbf{l_1} \gg \mathbf{l_2} \gg \mathbf{l_3}$, for a sequence of chords to be valid on a conic section, as shown in Fig. 3.1. Note that these situations are under the assumption that the edge pixels are not displaced by the image tessellation and noise corruption.

By limiting the pixel displacement to one pixel width only, the above situations can be extended to include the cases where two chords could be coincident (overlapping) to each other. Denote the relation of "coincident" by the symbol "//" such that $\mathbf{I}_i \,//\, \mathbf{I}_j$ stands for the

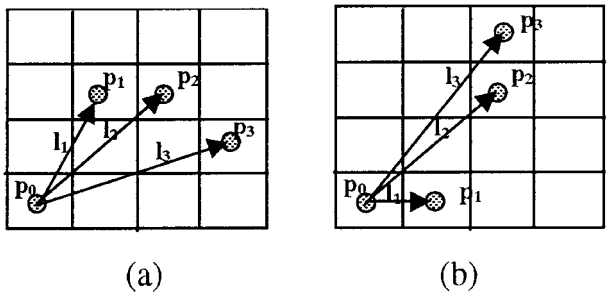

FIG. 3.1. Three chords in a sequence on a conic section: (a) in the clockwise direction and (b) in the counterclockwise direction.

coincidence of two chords $\mathbf{I}_i$ and $\mathbf{I}_j$. We allow for the replacement of any of the "$\ll$" and "$\gg$" relations in the above two situations by the "//" relation. That is, to include the "//" relation between two chords as a valid situation for the conic sections in the digital images. This leads to the valid chord depositions shown by the examples in Fig. 3.2.

The chord sequences that do not comply with the above situations present a violation to the conditions of being a conic section, or say that the edge pixels involved are not situated properly on a conic section. Some of the violation examples are shown in Fig. 3.3. Notice that a violating case may be caused by several reasons, such as the pixel tessellation, the noise corruption, or the existence of a break point among the pixels.

To rectify the pixel sequences with respect to the chord property of conic sections, a quantity called the *accordance value (av)* is computed for each pixel. The result of this computation will be the validation of the pixel being on a conic section. Notice that in a digital image, we allow for the relation between two sequential chords $\mathbf{I}_{i+1}$ and $\mathbf{I}_i$ being any one of the following three possible cases: (1) $\mathbf{I}_{i+1} \gg \mathbf{I}_i$, (2) $\mathbf{I}_{i+1} \ll \mathbf{I}_i$, and (3) $\mathbf{I}_{i+1} \,//\, \mathbf{I}_i$. Let

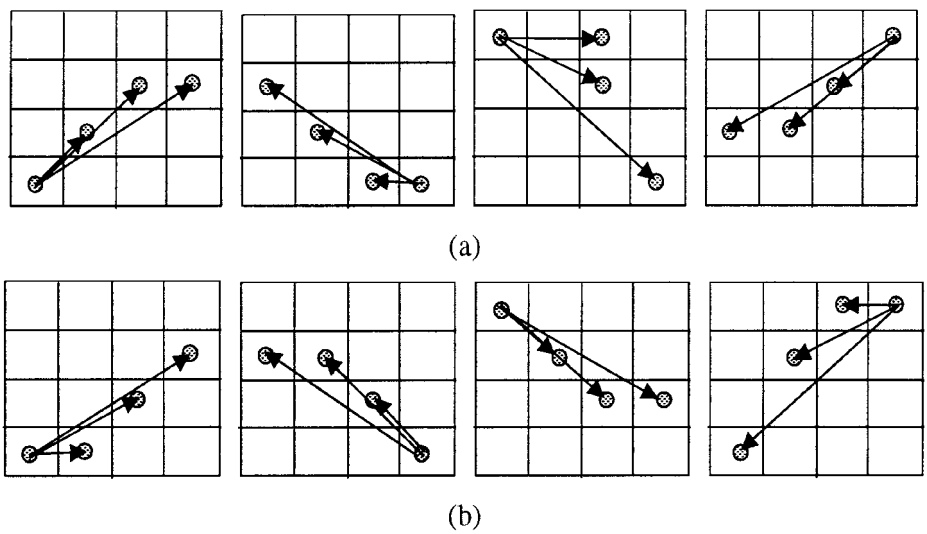

FIG. 3.2. Examples of valid chord relations in a digital image: (a) in the clockwise direction and (b) in the counterclockwise direction.

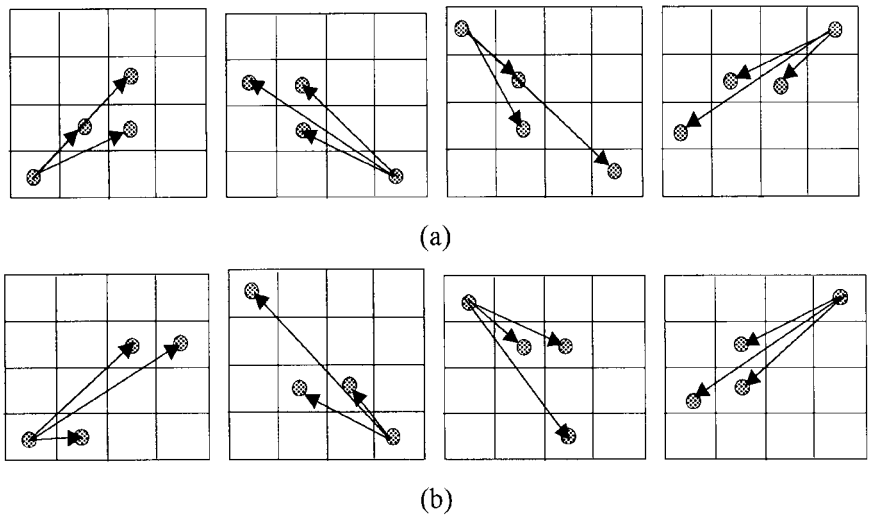

FIG. 3.3. Pixel sequences that violate the chord property of conic sections: (a) in the clockwise direction; (b) in the counterclockwise direction.

$v_{\mathbf{p2}}$ be the *av* assignment for pixel $\mathbf{p}_2$ and $v_{\mathbf{p3}}$ be the *av* assignment for pixel $\mathbf{p}_3$; we define

$$v_{\mathbf{p2}} = \begin{cases} +1, & \mathbf{l_2} \gg \mathbf{l_1}; \\ -1, & \mathbf{l_2} \ll \mathbf{l_1}; \\ 0, & \mathbf{l_2} \,/\!/\, \mathbf{l_1}. \end{cases} \quad v_{\mathbf{p3}} = \begin{cases} +1, & \mathbf{l_3} \gg \mathbf{l_2}; \\ -1, & \mathbf{l_3} \ll \mathbf{l_2}; \\ 0, & \mathbf{l_3} \,/\!/\, \mathbf{l_2}. \end{cases}$$

Figure 3.4 shows some examples of the *av* value assignment, according to the relations presented in the chords $\mathbf{l_1}$, $\mathbf{l_2}$, and $\mathbf{l_3}$. Where (a) shows a case of $\mathbf{I_3} \ll \mathbf{I_2}$ and $\mathbf{I_2} \,/\!/\, \mathbf{I_1}$, (b) a case of $\mathbf{I_3} \ll \mathbf{I_2}$ and $\mathbf{I_2} \gg \mathbf{I_1}$, (c) a case of $\mathbf{I_3} \gg \mathbf{I_2}$ and $\mathbf{I_2} \ll \mathbf{I_1}$, and (d) a case of $\mathbf{I_3} \,/\!/\, \mathbf{I_2}$ and $\mathbf{I_2} \gg \mathbf{I_1}$.

Let $t_{\mathbf{p}}$ denote the total value accumulated on a pixel $\mathbf{p}$. The following procedure traces the pixel sequences, detects inflection points, and eliminates pixels with extremely jagged displacement, according to the evaluation of $t_{\mathbf{p}}$.

PROCEDURE A (Pixel tracing and rectification). Let a set $\{\mathbf{p}_i; i = 0, 1, 2, \ldots, n\}$ be a sequence of edge pixels on an object boundary and let it be traced (checked) in a clockwise

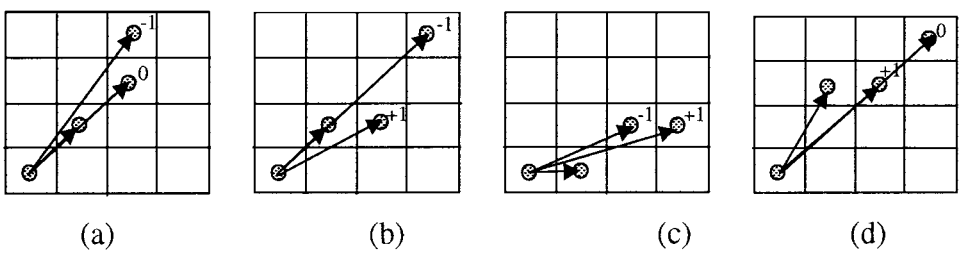

FIG. 3.4. Assignment of accordance value (*av*) to pixel sequences: (a) $\mathbf{I_3} \ll \mathbf{I_2}$ and $\mathbf{I_2} \,/\!/\, \mathbf{I_1}$; (b) $\mathbf{I_3} \ll \mathbf{I_2}$ and $\mathbf{I_2} \gg \mathbf{I_1}$; (c) $\mathbf{I_3} \gg \mathbf{I_2}$ and $\mathbf{I_2} \ll \mathbf{I_1}$; (d) $\mathbf{I_3} \,/\!/\, \mathbf{I_2}$ and $\mathbf{I_2} \gg \mathbf{I_1}$.

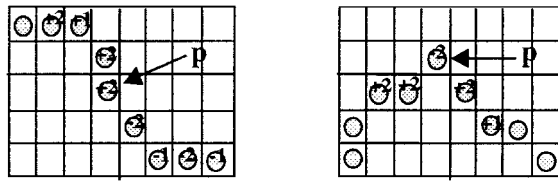

FIG. 3.5. Pixel rectification for conic sections: (a) at inflection point **p**; (b) at a jagged pixel **p**.

direction. For every pixel $\mathbf{p} \in \{\mathbf{p}_i\}$:

Step 1. Calculate the value $v_{\mathbf{p}}$ and add it to the value $t_{\mathbf{p}}$ while the pixels are examined.

Step 2. If three or more edge pixels ahead of **p** have the $t_{\mathbf{p}}$ value in a sign different from that of $t_{\mathbf{p}}$ and the pixels behind **p**, mark **p** an inflection point.

Step 3. If the pixel **p** has a $t_{\mathbf{p}}$ value greater than or equal to $+2$ or -2, and the signs of its neighbors are all different from $t_{\mathbf{p}}$, then **p** is extremely jagged or noise-corrupted. Purge **p** from the sequence.

Figure 3.5 shows the examples of an inflection point and an extremely jagged pixel detectable by the above procedure in a pixel sequence.

The reason that three or more pixels ahead of **p** are examined in Step 2 of above procedure for determining an inflection point is because that is the minimum number of pixels required in the algorithms to be discussed later for constructing a valid conic section.

2. *Generalized Guiding Triangle Formation*

Let the two end pixels of a boundary segment be denoted as $\mathbf{p_0}$ and $\mathbf{p_2}$, and their tangent vectors be denoted as $\mathbf{p_0^u}$ and $\mathbf{p_2^u}$, respectively. We then considered the angles α and β that are formed by the $\mathbf{p_0^u}$ and $\mathbf{p_2^u}$ with respect to the baseline $\mathbf{p_0}$–$\mathbf{p_2}$. They come with three situations:

(a) $\alpha < \beta$, a triangle is formed by $\mathbf{p_0^u}$, $\mathbf{p_2^u}$, and $\mathbf{p_0}$–$\mathbf{p_2}$ (a "real triangle").

(b) $\alpha = \beta$, the tangent $\mathbf{p_0^u}$ is parallel to $\mathbf{p_2^u}$, no real triangle is formed (considering the two vectors converging at the point of infinite, forming a "virtual triangle").

(c) $\alpha > \beta$, a triangle is formed on the opposite side of $\mathbf{p_0}$–$\mathbf{p_2}$ with respect to $\mathbf{p_0^u}$ and $\mathbf{p_2^u}$ (a "mirror triangle").

The situations are illustrated in Fig. 3.6.

It is noted that in any one of these situations, a series of conic sections can be defined by using the $\mathbf{p_0^u}$, $\mathbf{p_2^u}$, and $\mathbf{p_0}$–$\mathbf{p_2}$, as well as one additional point within the geometric region formed by these vectors. We thus call the geometry formed by the points $\mathbf{p_0}$ and $\mathbf{p_2}$ and their tangent vectors $\mathbf{p_0^u}$ and $\mathbf{p_2^u}$ the *generalized guiding triangle* (GGT) of conic sections and have the following theorem.

THEOREM A. *Giving a GGT and a point within the GGT* (*i.e.*, *bound by the lines* $\mathbf{p_0}$–$\mathbf{p_2}$, $\mathbf{p_0^u}$, *and* $\mathbf{p_2^u}$), *a valid conic section is uniquely defined.*

The proof of this theorem is placed in the Appendix B of this paper.

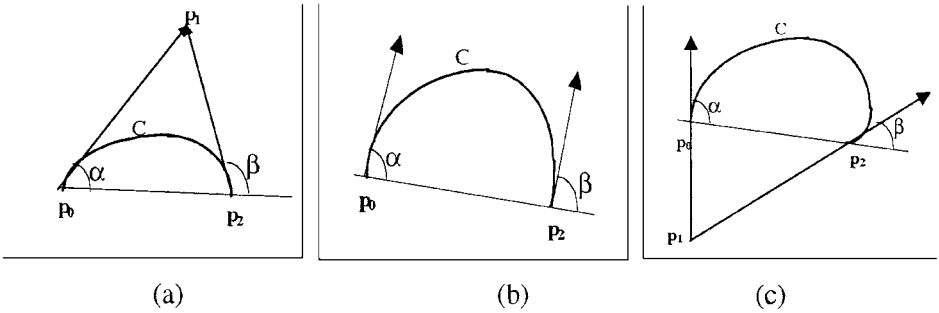

FIG. 3.6. Cases of generalized guiding triangles: (a) $\alpha < \beta$, a real triangle is formed by $\mathbf{p}_0^u$, $\mathbf{p}_2^u$, and $\mathbf{p}_0$–$\mathbf{p}_2$; (b) $\alpha = \beta$, a virtual triangle is formed by assuming the $\mathbf{p}_0^u$ and $\mathbf{p}_2^u$ converge at the point of infinity; (c) $\alpha > \beta$, a mirror triangle is formed on the opposite side of $\mathbf{p}_0$–$\mathbf{p}_2$.

3. Tangent Vector Approximation

To form a GGT, it is necessary to have the tangent vectors $\mathbf{p}_0^u$ and $\mathbf{p}_2^u$ be calculated. Note that $\mathbf{p}_0^u$ and $\mathbf{p}_2^u$ cannot be precisely measured in a digital image because of pixel tessellation and noise corruption. An approximation of the vectors thus is resorted.

In principle, the tangent at point (x_0, y_0) is defined by

$$y - y_0 = f'(x_0)(x - x_0),$$

where

$$f'(x_0) = \lim_{\Delta x \to 0} \frac{f(x + \Delta x) - f(x)}{\Delta x}.$$

This suggests that $\mathbf{p}_0^u$ and $\mathbf{p}_2^u$ can be approximated by taking the direction of the two closest pixels at the $\mathbf{p}_0$ and $\mathbf{p}_2$, respectively. However, this approximation often leads to a large error because of pixel tessellation and noise effect, as shown in Fig. 3.7a.

Let us consider a solution region for the two tangent vectors $\mathbf{p}_0^u$ and $\mathbf{p}_2^u$. It is conceived that the smaller the solution region, the more accurate a solution will be. Therefore rather than trying to directly explore the tangent vector, we could resort to finding a solution region and then attempt to narrow the region to a sufficiently small or acceptable magnitude. The general concept is that once the solution region is small enough, we can use it (or its component) as an approximation to the solution. This leads to the recursive procedure that applies the maximum inner triangle property of the shoulder point discussed in Section 2 to the construction of the solution regions.

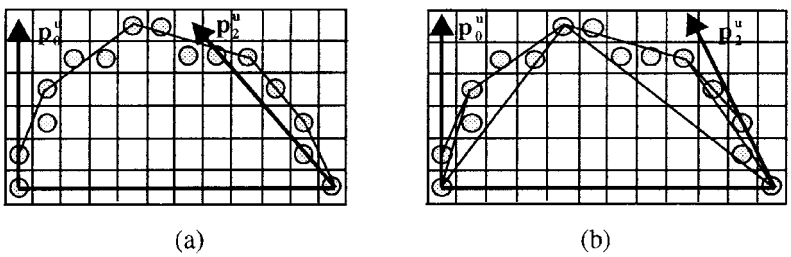

FIG. 3.7. Different ways of approximating tangent vectors: (a) via Δx approach; (b) via maximum inner triangle approach.

PROCEDURE B (Approximation of $\mathbf{p_0^u}$ and $\mathbf{p_2^u}$). Given a pixel set $\{\mathbf{p}_i;\ i = 1, 2, \ldots, n\}$ which is a sequence of edge pixels on an object boundary with $\mathbf{p_0}$ and $\mathbf{p_2}$ been identified as the end points.

Step 1. Find $\mathbf{p_v} \in \{\mathbf{p}_i\}$ such that its distance to line $\mathbf{p_0}$–$\mathbf{p_2}$ is the largest among pixels in $\{\mathbf{p}_i\}$.

Step 2. Divide the conic section into two subdivisions at the point $\mathbf{p_v}$,
Denote these two subdivisions as $\mathbf{p_0}$–$\mathbf{p_v}$ and $\mathbf{p_v}$–$\mathbf{p_2}$.

Step 3. For the subdivision $\mathbf{p_0}$–$\mathbf{p_v}$
Repeat
Find a new $\mathbf{p_v}$ its distance to line $\mathbf{p_0}$–$\mathbf{p_v}$ is the largest among the pixels on conic section $\mathbf{p_0}$–$\mathbf{p_v}$,
Divide the conic section into two subdivisions at the new point $\mathbf{p_v}$,
Until the distance of the new $\mathbf{p_v}$ to $\mathbf{p_0}$–$\mathbf{p_v}$ of the conic subdivision is less than one threshold ε.
Denote the vector $\mathbf{p_0}$–$\mathbf{p_v}$ as an approximation of $\mathbf{p_0^u}$.

Step 4. For the subdivision $\mathbf{p_v}$–$\mathbf{p_2}$
Repeat
Find a new $\mathbf{p_v}$ its distance to line $\mathbf{p_v}$–$\mathbf{p_2}$ is the largest among the pixels on conic section $\mathbf{p_v}$–$\mathbf{p_2}$,
Divide the conic section into two subdivisions at the new point $\mathbf{p_v}$,
Until the distance of the new $\mathbf{p_v}$ to $\mathbf{p_v}$–$\mathbf{p_2}$ of the conic subdivision is less than one threshold ε.
Denote the vector $\mathbf{p_2}$–$\mathbf{p_v}$ as an approximate of $\mathbf{p_2^u}$.

An example of applying the maximum inner triangle approach to the approximation of the tangent vectors is shown in Fig. 3.7b, where three consecutive maximum inner triangles are constructed for the solution regions of the $\mathbf{p_0^u}$ and $\mathbf{p_2^u}$. Justification of the algorithm is contained in Appendix C of this paper.

In the digital image processing, the point $\mathbf{p_v}$ is identified by compare the distances from the pixels on the edge sequence of the conic to the baseline $\mathbf{p_0}$–$\mathbf{p_2}$. The $\mathbf{p_v}$ has the largest distance measure. The process is terminated when no more inner triangles can be formed for the conic subdivisions or the height of the maximum inner triangle is small enough.

The computational complexity of Procedure B can be described in this way. Let n be the number of edge pixels computed. Step 1 of the procedure takes an $O(n)$ complexity. The complexity of Step 2 is a constant. Steps 3 and 4 have the same computational complexity which, in terms of the subdivision nature, is in the order of $O(n \log n)$. Therefore, the overall computational complexity for the procedure is $O(n \log n)$.

4. FORMULATING CONIC SECTIONS ON PIXEL SEQUENCES

In this section we describe the modeling of the boundaries of objects in the digital images in piece-wise approximations that consist of a number of conic pieces connected end-to-end at the break points of the boundary edges.

Let $\{\mathbf{p}_i;\ i = 1, 2, \ldots, n\}$ be a set of edge pixels organized in a sequence with $\mathbf{p_0}$ and $\mathbf{p_2}$ identified as the end points and $\mathbf{p_0^u}$ and $\mathbf{p_2^u}$ their tangent vectors. That is, the pixels in $\{\mathbf{p}_i\}$ are located within the region of the GGT formed by the vectors $\mathbf{p_0^u}$, $\mathbf{p_2^u}$, and $\mathbf{p_0}$–$\mathbf{p_2}$. Convert the $\mathbf{p_0^u}$ and $\mathbf{p_2^u}$ to the parametric forms (a_x, a_y, a_0) and (b_x, b_y, b_0), and $\mathbf{p_0}$–$\mathbf{p_2}$ to (u_x, u_y, u_0),

respectively. For each pixel $\mathbf{p}_i = (x_i, y_i) \in \{\mathbf{p}_i\}$, we then have

$$(a_x x_i + a_y y_i + a_0)(b_x x_i + b_y y_i + b_0) = K_i(u_x x_i + u_y y_i + u_0)^2. \tag{4.1}$$

That is, a unique value K_i can be obtained for any point within the general guiding triangle, such that

$$K_i = \frac{(a_x x_i + a_y y_i + a_0)(b_x x_i + b_y y_i + b_0)}{(u_x x_i + u_y y_i + u_0)^2}. \tag{4.2}$$

Let the K_i values being computed on all the edge pixel in $\{\mathbf{p}_i\}$, we then have $\{K_i, i = 1, 2, \ldots, n\}$. Question now is that how a K value can be derived from the $\{K_i\}$ so that a unique conic section expressed in the form of $(a_x x + a_y y + a_0)(b_x x + b_y y + b_0) - K(u_x x + u_y y + u_0)^2 = 0$ can be obtained. To answer this question, we first define a number of criteria for the evaluation of the fitness of the resulting conic sections to the given set of edge pixels on an object boundary.

Let $\mathbf{p}_i$ denote a pixel point within the set of points that are to be fitted by a conic section. Let $d_0(\mathbf{p}_i)$ be the distance from the point $\mathbf{p}_i$ to the base line $\mathbf{p_0}$–$\mathbf{p}_2$ of the guiding triangle for the fitting conic. According to the distance definition for a point to a line, we have

$$d_0(\mathbf{p}_i) = \frac{|u_x x_i + u_y y_i + u_0|}{\sqrt{u_x^2 + u_y^2}}. \tag{4.3}$$

Let $d_1(\mathbf{p}_i)$ be the distance from the point $\mathbf{p}_i$ to the corresponding fitting point on the fitted conic. Assuming the point set is fitted by the conic section

$$(a_x x + a_y y + a_0)(b_x x + b_y y + b_0) - K(u_x x + u_y y + u_0)^2 = 0. \tag{4.4}$$

We then define

$$d_1(\mathbf{p}_i) = (a_x x_i + a_y y_i + a_0)(b_x x_i + b_y y_i + b_0) - K(u_x x_i + u_y y_i + u_0)^2. \tag{4.5}$$

The examples of $d_0(\mathbf{p}_i)$ and $d_1(\mathbf{p}_i)$ are shown in Fig. 4.1.

Note that if $\mathbf{p}_i$ is on the fitted curve, $d_1(\mathbf{p}_i) = 0$. In other words, we can consider that the value $d_1(\mathbf{p}_i) \neq 0$ represents an error of the fitting. The task of fitting the edge pixels in $\{\mathbf{p}_i\}$ then is modeled as a process of minimizing the fitting errors associated with the $d_1(\mathbf{p}_i)$. This leads to the following theorem.

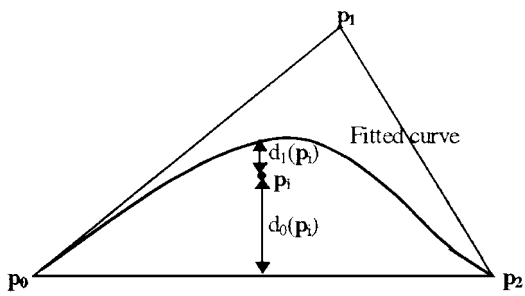

FIG. 4.1. Quantities $d_0(\mathbf{p}_i)$ and $d_1(\mathbf{p}_i)$ for establishing the criteria in conic fitting.

THEOREM B. Given a set of edge pixels $\{\mathbf{p}_i, i = 1, 2, \ldots, n\}$ that fall into the region of a GGT formed by the vectors $\mathbf{p_0^u}$, $\mathbf{p_2^u}$, and $\mathbf{p_0}$–$\mathbf{p_2}$, where $\mathbf{p}_i = (x_i, y_i) \in \{\mathbf{p}_i\}$. Let

$$K_i = \frac{(a_x x_i + a_y y_i + a_0)(b_x x_i + b_y y_i + b_0)}{(u_x x_i + u_y y_i + u_0)^2}.$$

A conic section can be constructed by taking the value K in the equation

$$(a_x x + a_y y + a_0)(b_x x + b_y y + b_0) = K(u_x x + u_y y + u_0)^2$$

in one of the following three ways:

(1) Let

$$K = \frac{1}{\sum_{i=1}^{n}(u_x x_i + u_y y_i + u_0)^4} \sum_{i=1}^{n}(u_x x_i + u_y y_i + u_0)^4 K_i;$$

an error of $\varepsilon(K) = \sum_{i=1}^{n}[d_1(\mathbf{p}_i)]^2$ is minimized.

(2) Let

$$K = \frac{1}{\sum_{i=1}^{n}(u_x x_i + u_y y_i + u_0)^2} \sum_{i=1}^{n}(u_x x_i + u_y y_i + u_0)^2 K_i;$$

an error of $\varepsilon(K) = \sum_{i=1}^{n}[d_1(\mathbf{p}_i)/d_0(\mathbf{p}_i)]^2$ is minimized.

(3) Let $K = (1/n)\sum_{i=1}^{n} K_i$; an error of $\varepsilon(K) = \sum_{i=1}^{n}[d_1(\mathbf{p}_i)/[d_0(\mathbf{p}_i)]^2]^2$ is minimized.

The proof of the theorem is in Appendix D. Applying the above theorem, we have the following procedure for the construction of a conic section for a given set of edge pixels between two break points on an object boundary.

PROCEDURE C (Constructing a conic approximation for an edge segment). Given a sequence of edge pixels $\{\mathbf{P}_i; i = 1, 2, \ldots, n\}$ with $\mathbf{p_0}$ and $\mathbf{p_2}$ identified as the end points and $\mathbf{p_0^u}$ and $\mathbf{p_2^u}$ the approximations obtained by Procedure B.

Step 1. Convert the $\mathbf{p_0^u}$ and $\mathbf{p_2^u}$ to the parameters (a_x, a_y, a_0) and (b_x, b_y, b_0), and the vector $\mathbf{p_0}$–$\mathbf{p_2}$ to (u_x, u_y, u_0) of equation $(a_x x + a_y y + a_0)(b_x x + b_y y + b_0) = K(u_x x + u_y y + u_0)^2$.

Step 2. For each pixel $\mathbf{p}_i = (x_i, y_i) \in \{\mathbf{p}_i\}$, calculate

$$K_i = \frac{(a_x x_i + a_y y_i + a_0)(b_x x_i + b_y y_i + b_0)}{(u_x x_i + u_y y_i + u_0)^2}.$$

Step 3. Let

$$K = \frac{1}{\sum_{i=1}^{n}(u_x x_i + u_y y_i + u_0)^4} \sum_{i=1}^{n}(u_x x_i + u_y y_i + u_0)^4 K_i,$$

or

$$K = \frac{1}{\sum_{i=1}^{n}(u_x x_i + u_y y_i + u_0)^2} \sum_{i=1}^{n}(u_x x_i + u_y y_i + u_0)^2 K_i,$$

or $K = (1/n)\sum_{i=1}^{n} K_i$, in terms of the preference of the error function to be minimized.

Step 4. Replace the K in equation $(a_x x + a_y y + a_0)(b_x x + b_y y + b_0) = K(u_x x + u_y y + u_0)^2$ by the value of step 3.

The computational complexity of Procedure C is clearly in the order of $O(n)$, where n is the number of edge points processed in the process. The GGT-based procedure for the construction of conic section approximations of object boundaries is presented as the following.

PROCEDURE D (Constructing conic approximation for object boundaries).
Input: An edge-detected image with object boundaries represented in pixel sequences.
Output: A set of conic section denotations along the boundaries of an object.

Step 1. Apply a 3×3 window operator [28] to find the break points along the object boundaries.

Step 2. Apply Procedure A to rectify the edge segments and identify the inflection points.

Step 3. For each edge segment in the sequence along the object boundary

3.1. apply Procedure B to find $\mathbf{p}_0^{\mathbf{u}}$ and $\mathbf{p}_2^{\mathbf{u}}$ approximation for an edge segment between two end points $\mathbf{p_0}$ and $\mathbf{p_2}$;

3.2. apply Procedure C to obtain the conic section parameters for the edge segment.

Step 4. Redraw the object boundaries between points s, $\mathbf{p_0}$, and $\mathbf{p}_2$ by the conic sections identified.

Let nd be the total number of edge pixels in an image. The computational complexity of the Procedure D can be derived as the following. First, both Step 1 and Step 2 take $O(n)$ complexity. Step 3 calls Procedure B and Procedure C which then have the $O(n \log n)$ and $O(n)$ complexity, respectively. Since each edge segment is processed only once in Step 3, the overall computational complexity of Step 3 is $O(n \log n)$. The computational complexity of Procedure D thus is $O(n \log n)$.

5. EXPERIMENTS

Experiments on the procedures for conic section approximation described above were conducted with the use of synthetic data and real images, respectively. In the cases of synthetic data, we first assume the existence of an original conic curve. A set of data points are then generated from adding noise and displacement to the selected data points on the original curve. This approach allows us to make a quantitative measurement of the accuracy of the result with respect to the original curve rather than the noise-corrupted data points. In the cases involving the use of real images, the procedures are applied to the object boundaries that have been through a number of image preprocesses to have the edge pixels extracted and the segments properly separated according to the detection of break points.

Usually, a curve-fitting algorithm is evaluated by computing the mean-squared errors between the points on the fitted curve and the given set of data points. The problem with this method is that it only measures how good the fit is with respect to the given data set, but not to the original curve represented by the data points. Since the information about the original curve is available in our synthetic test cases, it is possible to measure the real errors between the fitted points and the true curve positions. Let C_0 be the original conic section and let $C1$ be the fitted curve. We define the fit of the curve as

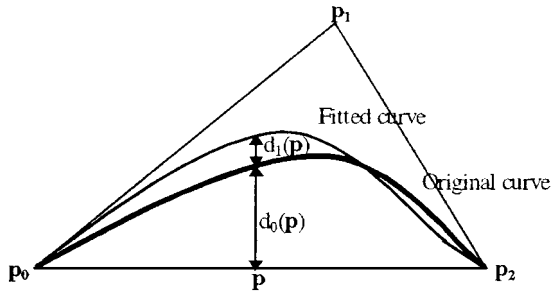

FIG. 5.1. Measurement of fitness on a conic section, where $\mathbf{p}$ is a point on the base line $\mathbf{p_0}$–$\mathbf{p_2}$, $d_0(\mathbf{p})$ is the distance from the point $\mathbf{p}$ to a corresponding point on the original conic, and $d_1(\mathbf{p})$ is the distance from the fitted curve to the corresponding point on the original curve.

$$F(C1, C_0) = \sum_{\mathbf{p}=\mathbf{p_0}}^{\mathbf{p_2}} \frac{(d_0(\mathbf{p}))^2 + (d_1(\mathbf{p}))^2}{(d_0(\mathbf{p}))^2}, \tag{5.1}$$

where, $d_0(\mathbf{p})$ denotes the distance from a point $\mathbf{p}$ on the base line of the guiding traingle to the original conic section. The quantity $d_0(\mathbf{p}) + d_1(\mathbf{p})$ equals the distance from $\mathbf{p}$ to the fitted curve ($d_1(\mathbf{p})$ is the error between the fitted curve and the original one with respect to $\mathbf{p}$). The denotation of these measurements are shown in Fig. 5.1. Notice that a perfect fitting will have $F(C1, C_0) = 1$ according to this definition. A value of $F(C1, C_0) < 1$ will represent a fitting that is under the original curve with respect to the base line, and $F(C1, C_0) > 1$ will represent a fitting over the original curve with respect to the base line. In actual computation, the corresponding $d_0(\mathbf{p})$ and $d_1(\mathbf{p})$ values are first calculated with respect to the common parameter t of the geometrical equations and then converted to the distance measure with respect to the point $\mathbf{p}$ on the base line $\mathbf{p_0}$–$\mathbf{p_2}$.

In the first example of the experiments, we show the approximation of the conic section at the first quadrant of an ellipse. The original ellipse $C0$ is expressed as

$$C0\text{: } 9x^2 + 16y^2 - 144 = 0. \tag{5.2}$$

The points, as shown in Fig. 5.2a, are generated by adding Gaussian noises to a set of pixels originally located on the ellipse section. The end points of the section ($\mathbf{p_0}$ and $\mathbf{p_2}$) d are at (0, 3) and (4, 0), respectively. The tangent vectors $\mathbf{p_0^u}$ and $\mathbf{p_2^u}$ are [1, 0] and [0, 1]. The noise ratio $n(\mathbf{p})$ is measured by comparing the offset of the point away from the original point, say $d_1(\mathbf{p})$, to the distance of the original point to the base line of the conic section, i.e., $d_0(\mathbf{p})$ (refer to Fig. 5.1) such that

$$n(\mathbf{p}) = \frac{(d_1(\mathbf{p}))^2}{(d_0(\mathbf{p}))^2}. \tag{5.3}$$

For data points shown in Fig. 5.2a, the $n(\mathbf{p})$ has a distribution of $G(0.0, 0.02)$, where $G(\,)$ denotes a Gaussian density with 0.0 mean and 0.02 variance.

Figure 5.2b shows the original ellipse, named $C0$, and the conic approximation generated by applying the GGT procedure, named $C1$, as well as that generated by a least-square fitting algorithm [25], named $C2$, and a generalized Hough transformation algorithm [4], named $C3$, respectively. Table 1 shows the computation results of the K_i values obtained on the given point set.

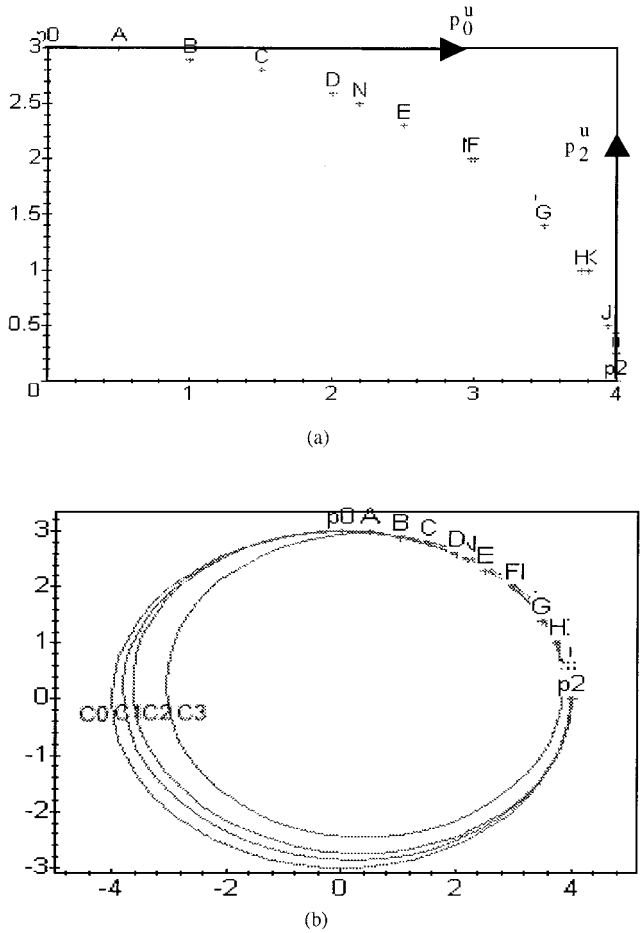

FIG. 5.2. Conic curves generated on a give set of points: (a) the point set; (b) the fitted curves, where $C0$ is the original conic, $C1$ is the conic generated by the GGT procedure, $C2$ is generated by a least-square fitting, and $C3$ is generated by a generalized Hough transformation.

TABLE 1
Computation of K on a Given Point Set for Conic Approximation

	x_p	y_p	K_i
A	0.5	3	
B	1	2.9	0.03866
C	1.5	2.8	0.03652
D	2	2.6	0.04132
E	2.5	2.3	0.04753
F	3	2	0.04
G	3.5	1.4	0.04759
H	3.75	1	0.04734
J	3.94	0.5	0.04528
K	3.8	1	0.03460
L	3.46	1.5	0.04222
M	2.98	2	0.03688
N	2.2	2.5	0.04253
I	4	0.5	

The average value K of 0.04286 is used in obtaining the conic section $C1$. Notice that during the process of deriving the conic section (Procedure C) we used the algebraic form of the conic expression,

$$(a_x x + a_y y + a_0)(b_x x + b_y y + b_0) = K(u_x x + u_y y + u_0)^2.$$

After obtaining the parameter K for the conic section, we rewrite the expression in the form

$$f(x, y) = ax^2 + by^2 + 2hxy + 2ux + 2vy + d = 0 \tag{5.4}$$

with

$$a = a_x b_x - K u_x^2, \tag{5.5}$$

$$b = a_y b_y - K u_y^2, \tag{5.6}$$

$$h = \frac{a_x b_y + a_y b_x}{2} - K u_x u_y, \tag{5.7}$$

$$u = \frac{a_x b_0 + a_0 b_x}{2} - K u_x u_0, \tag{5.8}$$

$$v = \frac{a_y b_0 + a_0 b_y}{2} - K u_y u_0, \tag{5.9}$$

$$d = a_0 b_0 - K u_0^2. \tag{5.10}$$

Applying these equations, we get the fitted conic section $C1$ in the expression

$$C1\text{: } 9x^2 + 16y^2 + 0.7xy - 2x - 2.7y - 136 = 0. \tag{5.11}$$

It is very close to the original ellipse of (5.2). We have the fitness measure $F(C1, C0) = 1.06$. The expressions of $C2$ and $C3$ are obtained as

$$C2\text{: } 9x^2 + 14.4y^2 + 5.8xy - 22x - 24.2y + 55.6 = 0, \tag{5.12}$$

$$C3\text{: } 8x^2 + 14y^2 - 8x - 7y - 109 = 0, \tag{5.13}$$

which have the fitness measure of $F(C2, C0) = 1.09$ and $F(C3, C0) = 1.14$, respectively.

The noise ratio of the given point set is increased in the successive experiments. Figure 5.3a shows the data points generated by adding additional noisy displacement to the data set of Fig. 5.2a. The data set shown in Fig. 5.3a has $n(\mathbf{p})$ of $G(0.0, 0.04)$. The resulting conic approximations to these points are shown in Fig. 5.3b, where the ellipse $C0$ is the original, $C1$ is generated by the GGT-based procedure, $C2$ and $C3$ are from least-square fitting and generalized Hough transformation. The conic expressions for these approximations are

$$C1\text{: } 9x^2 + 16y^2 + 2xy - 9x - 11y - 110 = 0, \tag{5.14}$$

$$C2\text{: } 5x^2 + 6y^2 - 10x - 9y - 24 = 0, \tag{5.15}$$

$$C3\text{: } 8x^2 + 9y^2 - 2.6xy + 6.4x + 25y - 156 = 0. \tag{5.16}$$

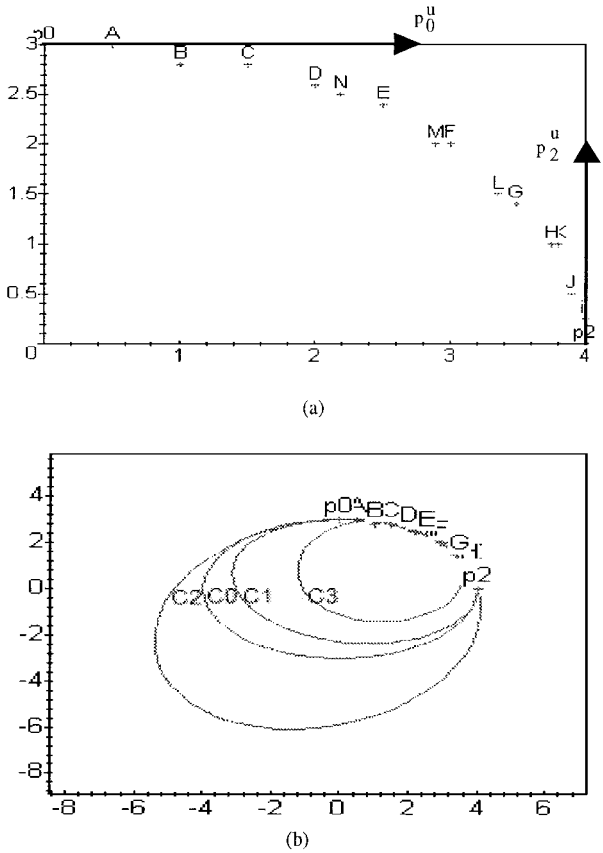

FIG. 5.3. Examples of conic curves generated from noise points: (a) the point set; (b) the fitted curves, where $C0$ is the original conic, $C1$ is the conic generated by the GGT procedure, $C2$ is generated by a least-square fitting, and $C3$ by a generalized Hough transformation.

The fitness measures are $F(C1, C0) = 1.13$, $F(C2, C0) = 1.21$, and $F(C3, C0) = 1.32$, respectively. Figure 5.4 shows a plot of the fitness measurements on a number of test cases with respect to noise distributions from $G(0.0, 0.01)$ to $G(0.0, 0.1)$.

The procedure is also tested on real images with objects in curved boundaries. In the tests, the object boundaries are extracted by first applying an edge detection operation and then an edge linking process that identifies the edge segments and further limits the edge sequences to one pixel width. The set of edge segments, $\{\mathbf{p}_i\}$'s, are identified before the conic approximation algorithms are applied. The examples of Fig. 5.5 show (a) the original images of objects, (b) the edge images after an edge detection operation, (c) the conic sections superimposed on the original object boundaries.

6. CONCLUSION

The conciseness and accuracy of the boundary representation make it preferred for describing the object shapes in many computer vision applications. The conic section representation of object boundaries has several advantages: (1) simplicity, only second-order mathematical expression (quadratic equation) is involved in the representation; (2) popularity, the boundaries of most popular objects (natural and manmade) in computer vision

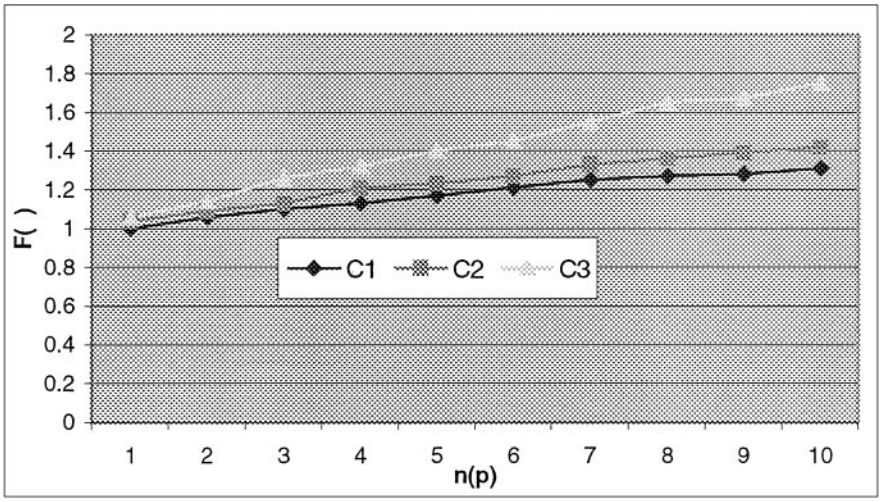

FIG. 5.4. Fitness measurement of the conic approximations vs noise ratio of data points.

applications can be fitted in conic sections; (3) efficiency, some unique geometrical properties of the conic sections make the identification of the parameters of the curve segments possible with the use of noniterative procedures. Even though the conic section is a simple type of curves, it is possible to use them to model and display the more complex curves. The algorithm described here takes advantage of the geometrical properties of conic sections for generating conic section approximations of object boundaries in consideration of image tessellation and noisy effect on the displacement of the edge pixels. The algorithm is efficient in terms of its computational complexity. No iterative computations are needed to derive the

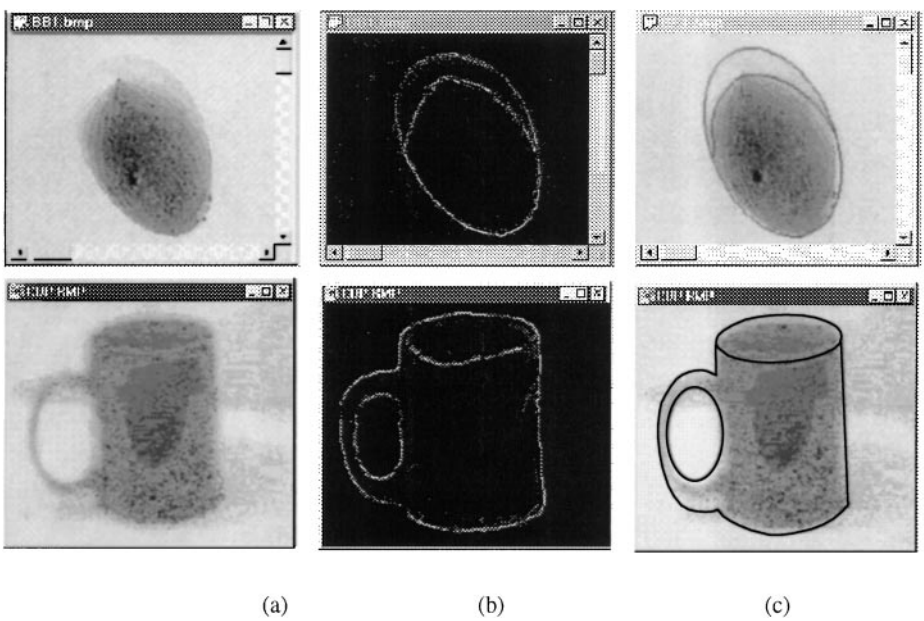

FIG. 5.5. Conic approximation of object boundaries in digital images: (a) original image; (b) boundary edges of the objects in the images; and (c) conic sections on the boundaries of the objects.

parameters of the conic expressions for the boundary segments. While the accuracy of this method relies on the proper detection of the geometrical features represented by the object boundary pixels, the results can be optimized with respect to different criteria in terms of the requirements of the applications.

APPENDIX A

(1) Define the point $\mathbf{p_v} = C(t = 1/2)$, that is, the point where the parameter t of the geometrical equation for the conic section C is equal to $1/2$.

(2) Let $C(t) = [\begin{smallmatrix} C^y(t) \\ C^x(t) \end{smallmatrix}]$ and calculate $C'(t) = dC^y(t)/dC^x(t)$. That is,

$$C'(t) = \frac{dC^y(t)}{dC^x(t)} = \frac{\frac{dC^y(t)}{dt}}{\frac{dC^x(t)}{dt}} = \frac{-2p_0^y + 2tp_0^y + 2wp_1^y - 4twp_1^y + 2tp_2^y}{-2p_0^x + 2tp_0^x + 2wp_1^x - 4twp_1^x + 2tp_2^x}. \tag{A.1}$$

Let $t = 1/2$ in above expression; we get

$$C'(t = 1/2) = \frac{-p_0^y + 2wp_1^y - 2wp_1^y + p_2^y}{-p_0^x + 2wp_1^x - 2wp_1^x + p_2^x} = \frac{p_0^y - p_2^y}{p_0^x - p_2^x}. \tag{A.2}$$

That is, the tangent of the baseline $\mathbf{p_0}$–$\mathbf{p_2}$. So it shows that the point, $\mathbf{p_v} = C(t = 1/2)$ is also the point where the conic has the tangent vector parallel to the baseline $\mathbf{p_0}$–$\mathbf{p_2}$.

(3) Observe that if any point $\mathbf{p_s}$ on the conic section has a greater distance to the baseline than the point $\mathbf{p_v}$, then point $\mathbf{p_v}$ must be located at a position between $\mathbf{p_s}$ and $\mathbf{p_0}$–$\mathbf{p_2}$. It means that the line parallel to $\mathbf{p_0}$–$\mathbf{p_2}$ and passing at $\mathbf{p_v}$ will have to intersect the conic more than once. This contradicts with the fact (1) above that the line parallel to $\mathbf{p_0}$–$\mathbf{p_2}$ and passing at $\mathbf{p_v}$ is the tangent of conic at $\mathbf{p_v}$. Therefore, the point $\mathbf{p_v}$ is the point on the conic that has the largest distance to the baseline $\mathbf{p_0}$–$\mathbf{p_2}$.

(4) Denoted the triangle formed by the points $\mathbf{p_0}$, $\mathbf{p_v}$, and $\mathbf{p_2}$, as $\mathbf{p_0}$–$\mathbf{p_v}$–$\mathbf{p_2}$, it is seen that its area equals the product of the baseline $\mathbf{p_0}$–$\mathbf{p_2}$ and the half the distance from the point on the conic to the baseline. Since the point $\mathbf{p_v}$ has the largest distance from the conic to the baseline, thus the triangle formed by $\mathbf{p_v}$ and $\mathbf{p_0}$–$\mathbf{p_2}$ has the largest area.

(5) To prove that the conic point $\mathbf{p_v}$ is also on the line $\mathbf{p_1} - (\mathbf{p_0} + \mathbf{p_2})/2$, we show that the point $C(t = 1/2)$ is the intersecting point of the conic with the line $\mathbf{p_1} - (\mathbf{p_0} + \mathbf{p_2})/2$. Using the geometrical expression of the conic with $t = 1/2$, we have

$$C(t = 1/2) = \frac{\left(1 - \frac{1}{2}\right)^2 \mathbf{p_0} + 2\left(1 - \frac{1}{2}\right)\frac{1}{2} w \mathbf{p_1} + \left(\frac{1}{2}\right)^2 \mathbf{p_2}}{\left(1 - \frac{1}{2}\right)^2 + 2\left(1 - \frac{1}{2}\right)\frac{1}{2} w + \left(\frac{1}{2}\right)^2} = \frac{\frac{1}{4}\mathbf{p_0} + \frac{1}{2} w \mathbf{p_1} + \frac{1}{4}\mathbf{p_2}}{\frac{1}{4} + \frac{1}{2} w + \frac{1}{4}}$$

$$= \frac{\mathbf{p_0} + 2w\mathbf{p_1} + \mathbf{p_2}}{2 + 2w}. \tag{A.3}$$

The above can be expressed as

$$C(t = 1/2) = \begin{bmatrix} C^y\left(\frac{1}{2}\right) \\ C^x\left(\frac{1}{2}\right) \end{bmatrix} = \frac{1}{2 + 2w} \begin{bmatrix} p_0^y + 2wp_1^y + p_2^y \\ p_0^x + 2wp_1^x + p_2^x \end{bmatrix} \tag{A.4}$$

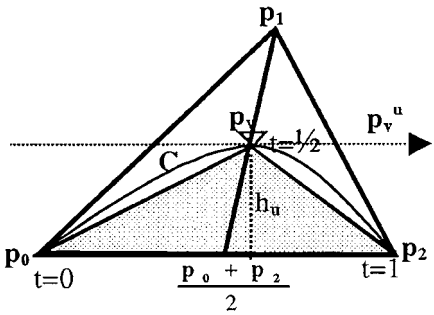

FIG. A.1. Properties of the shoulder point $\mathbf{p_v}$ on a conic section C: (1) parameter $t = 1/2$; (2) tangent vector $\mathbf{p_v^u}$ parallel to $\mathbf{p_0}$–$\mathbf{p_2}$; (3) h_u is the largest among all distances from C to $\mathbf{p_0}$–$\mathbf{p_2}$; (4) triangle $\mathbf{p_0}$–$\mathbf{p_2}$–$\mathbf{p_v}$ has the largest area among all triangles formed by $\mathbf{p_0}$, $\mathbf{p_2}$, and a point on the conic section; (5) $\mathbf{p_v}$ is on the line $\mathbf{p_1} - (\mathbf{p_0} + \mathbf{p_2})/2$.

Let the middle line $\mathbf{p_1} - (\mathbf{p_0} + \mathbf{p_2})/2$ be expressed as

$$\frac{y - p_1^y}{x - p_1^x} = \frac{p_1^y - \frac{p_0^y + p_2^y}{2}}{p_1^x - \frac{p_0^x + p_2^x}{2}}. \tag{A.5}$$

By some arrangement, we have

$$\frac{y - p_1^y}{x - p_1^x} = \frac{2p_1^y - p_0^y - p_2^y}{2p_1^x - p_0^x - p_2^x} \tag{A.6}$$

Replacing x and y in the left hand side of the above equation by $C(t = 1/2) = [\frac{C^y(\frac{1}{2})}{C^x(\frac{1}{2})}]$, we get

$$\begin{aligned}\frac{y - p_1^y}{x - p_1^x} = \frac{\frac{p_0^y + 2wp_1^y + p_2^y}{2+2w} - p_1^y}{\frac{p_0^x + 2wp_1^x + p_2^x}{2+2w} - p_1^x} &= \frac{p_0^y + 2wp_1^y + p_2^y - (2+2w)p_1^y}{p_0^x + 2wp_1^x + p_2^x - (2+2w)p_1^x} \\ &= \frac{2p_1^y - p_0^y - p_2^y}{2p_1^x - p_0^x - p_2^x}\end{aligned} \tag{A.7}$$

So the point $C(t = 1/2)$ is on the line $\mathbf{p_1} - (\mathbf{p_0} + \mathbf{p_2})/2$.

Figure A.1 puts the above properties of the shoulder point $\mathbf{p_v}$ in a single drawing.

APPENDIX B

To prove Theorem **A** we consider the algebraic equation of the conic section,

$$(a_x x + a_y y + a_0)(b_x x + b_y y + b_0) = K(u_x x + u_y y + u_0)^2, \tag{B.1}$$

where $a_x x + a_y y + a_0 = 0$ is defined by vector $\mathbf{p_0^u}$, $(b_x x + b_y y + b_0) = 0$ by vector $\mathbf{p_2^u}$, and $u_x x + u_y y + u_0 = 0$ by vector $\mathbf{p_0}$–$\mathbf{p_2}$. A point (x_p, y_p) within the GGT means that the following equations must be satisfied by the point (x_p, y_p) (only one situation is illustrated

without losing generality):

$$a_x x_p + a_y y_p + a_0 > 0, \tag{B.2}$$

$$(b_x x_p + b_y y_p + b_0) < 0, \tag{B.3}$$

$$u_x x_p + u_y y_p + u_0 \neq 0. \tag{B.4}$$

In any of these cases, a nonzero solution for K,

$$K = \frac{(a_x x_p + a_y y_p + a_0)(b_x x_p + b_y y_p + b_0)}{(u_x x_p + u_y y_p + u_0)^2} \neq 0, \tag{B.5}$$

exists. Replacing the K in (B.1), we see that a valid conic section is obtained.

APPENDIX C

The algorithm for the approximation of $\mathbf{p_0^u}$ and $\mathbf{p_2^u}$ can be justified in the following way. First, it is known that the least error solution in an asymptotic approximation of a conic section by two straight lines (i.e., closest to the conic) is the two sides of the maximum inner triangle. The error is measured as the area between the conic section and the line pieces. Let the two sides of the inner triangle be denoted as $\mathbf{p_0'}$ and $\mathbf{p_2'}$, respectively. Let the angle between $\mathbf{p_0'}$ and $\mathbf{p_0^u}$ be denoted as $\angle\mathbf{p_0'}$–$\mathbf{p_0^u}$ and the angle between $\mathbf{p_2'}$ and $\mathbf{p_2^u}$ be as $\angle\mathbf{p_2'}$–$\mathbf{p_2^u}$. Taking the summation of $\angle\mathbf{p_0'}$–$\mathbf{p_0^u}$ and $\angle\mathbf{p_2'}$–$\mathbf{p_2^u}$ as the solution region for the approximation of $\mathbf{p_0^u}$ and $\mathbf{p_2^u}$, it is obvious that $\mathbf{p_0'}$ and $\mathbf{p_2'}$ of the maximum inner triangle give the least error.

Second, we show that the solution region is reduced in repetitive application of the above approach. Notice that the $\mathbf{p_v}$ chosen in the algorithm is the shoulder point of the conic. It is then possible to construct a maximum inner triangle for each of the conic subdivisions based on $\mathbf{p_0}$–$\mathbf{p_v}$ and $\mathbf{p_v}$–$\mathbf{p_2}$. Repeating the process will reach a point where the inner triangles in the successive subdivisions of the conic section are acceptably small so that no more subdivision is needed. Let us keep using the symbol $\mathbf{p_v}$ to denote the middle points of the successive subdivisions of the conic section; at the end of the repeating process the left-most vector $\mathbf{p_0}$–$\mathbf{p_v}$ and the right-most vector $\mathbf{p_2}$–$\mathbf{p_v}$ can then be used to approximate the $\mathbf{p_0^u}$ and $\mathbf{p_2^u}$.

APPENDIX D

We prove Theorem B in accordance with three cases:

(1) Let $\varepsilon(K) = \sum_{i=1}^{n} [d_1(\mathbf{p}_i)]^2$; that is,

$$\varepsilon(K) = \sum_{i=1}^{n} \left[(a_x x_i + a_y y_i + a_0)(b_x x_i + b_y y_i + b_0) - K(u_x x_i + u_y y_i + u_0)^2\right]^2. \tag{D.1}$$

To minimize $\varepsilon(K)$, we take a differentiation of $\varepsilon(K)$ with respect to K, which yields

$$\varepsilon'(K) = \sum_{i=1}^{n} -2\left[(a_x x_i + a_y y_i + a_0)(b_x x_i + b_y y_i + b_0) - K(u_x x_i + u_y y_i + u_0)^2\right] \times (u_x x_i + u_y y_i + u_0)^2. \tag{D.2}$$

Let $\varepsilon'(K) = 0$; we get

$$\sum_{i=1}^{n} \left[(a_x x_i + a_y y_i + a_0)(b_x x_i + b_y y_i + b_0)(u_x x_i + u_y y_i + u_0)^2 - K(u_x x_i + u_y y_i + u_0)^4 \right] = 0. \tag{D.3}$$

Note that

$$(a_x x_i + a_y y_i + a_0)(b_x x_i + b_y y_i + b_0) = K_i(u_x x_i + u_y y_i + u_0)^2 \tag{D.4}$$

which yields

$$K = \frac{1}{\sum_{i=1}^{n}(u_x x_i + u_y y_i + u_0)^4} \sum_{i=1}^{n}(u_x x_i + u_y y_i + u_0)^4 K_i. \tag{D.5}$$

(2) Let

$$\varepsilon(K) = \sum_{i=1}^{n} \left[\frac{d_1(\mathbf{p}_i)}{d_0(\mathbf{p}_i)} \right]^2;$$

that is,

$$\varepsilon(K) = \sum_{i=1}^{n} \left[\frac{(a_x x_i + a_y y_i + a_0)(b_x x_i + b_y y_i + b_0) - K(u_x x_i + u_y y_i + u_0)^2}{\frac{|u_x x_i + u_y y_i + u_0|}{\sqrt{u_{xi}^2 + u_y^2}}} \right]^2. \tag{D.6}$$

Taking a differentiation of $\varepsilon(K)$ with respect to K, it yields

$$\varepsilon'(K) = \sum_{i=1}^{n} -2\left[(a_x x_i + a_y y_i + a_0)(b_x x_i + b_y y_i + b_0) - K(u_x x_i + u_y y_i + u_0)^2 \right]\left(u_x^2 + u_y^2\right). \tag{D.7}$$

Let $\varepsilon'(K) = 0$; we get

$$\sum_{i=1}^{n} \left[(a_x x_i + a_y y_i + a_0)(b_x x_i + b_y y_i + b_0) - K(u_x x_i + u_y y_i + u_0)^2 \right] = 0. \tag{D.8}$$

Note that

$$(a_x x_i + a_y y_i + a_0)(b_x x_i + b_y y_i + b_0) = K_i(u_x x_i + u_y y_i + u_0)^2; \tag{D.9}$$

that is,

$$\sum_{i=1}^{n} \left[K_i(u_x x_i + u_y y_i + u_0)^2 - K(u_x x_i + u_y y_i + u_0)^2 \right] = 0 \tag{D.10}$$

which yields

$$K = \frac{1}{\sum_{i=1}^{n}(u_x x_i + u_y y_i + u_0)^2} \sum_{i=1}^{n}(u_x x_i + u_y y_i + u_0)^2 K_i. \tag{D.11}$$

(3) Let

$$\varepsilon(K) = \sum_{i=1}^{n} \left[\frac{d_1(\mathbf{p}_i)}{[d_0(\mathbf{p}_i)]^2} \right]^2;$$

that is,

$$\varepsilon(K) = \sum_{i=1}^{n} \left[\frac{(a_x x_i + a_y y_i + a_0)(b_x x_i + b_y y_i + b_0) - K(u_x x_i + u_y y_i + u_0)^2}{\left[\frac{|u_x x_i + u_y y_i + u_0|}{\sqrt{u_{xi}^2 + u_y^2}} \right]^2} \right]^2. \quad \text{(D.12)}$$

Taking a differentiation of $\varepsilon(K)$ with respect to K, it yields

$$\varepsilon'(K) = \sum_{i=1}^{n} -2\big[(a_x x_i + a_y y_i + a_0)(b_x x_i + b_y y_i + b_0) - K(u_x x_i + u_y y_i + u_0)^2\big] \times \frac{\left(u_x^2 + u_y^2\right)^2}{(u_x x_i + u_y y_i + u_0)^2}. \quad \text{(D.13)}$$

Let $\varepsilon'(K) = 0$; we get

$$\sum_{i=1}^{n} \big[(a_x x_i + a_y y_i + a_0)(b_x x_i + b_y y_i + b_0) - K(u_x x_i + u_y y_i + u_0)^2\big] \frac{\left(u_x^2 + u_y^2\right)^2}{(u_x x_i + u_y y_i + u_0)^2} = 0. \quad \text{(D.14)}$$

Again, since

$$(a_x x_i + a_y y_i + a_0)(b_x x_i + b_y y_i + b_0) = K_i(u_x x_i + u_y y_i + u_0)^2, \quad \text{(D.15)}$$

we have

$$\sum_{i=1}^{n} \big[K_i(u_x x_i + u_y y_i + u_0)^2 - K(u_x x_i + u_y y_i + u_0)^2\big] \frac{\left(u_x^2 + u_y^2\right)^2}{(u_x x_i + u_y y_i + u_0)^2} = 0, \quad \text{(D.16)}$$

which yields

$$\sum_{i=1}^{n} [K_i - K]\left(u_x^2 + u_y^2\right)^2 = 0. \quad \text{(D.17)}$$

That is,

$$K = \frac{1}{n} \sum_{i=1}^{n} K_i. \quad \text{(D.18)}$$

ACKNOWLEDGMENTS

The author thanks the reviewers for providing detailed and valuable comments to the earlier version of this paper. The support of the University Committee on Research at the University of Nebraska at Omaha is also acknowledged.

REFERENCES

1. A. Albano, Representation of digitized contours in terms of conic arcs and straight-line segments, *Comput. Graphics Image Process*. **3**, 1974, 23–33.
2. F. Bookstein, Fitting conic sections to scattered data, *Comput. Graphics Image Process*. **9**, 1979, 56–71.
3. W. Chang and K. Chou, A fast algorithm for cubic B-spline curve fitting, *Comput. & Graphics* **18**, No. 3, 1994, 327–334.
4. E. R. Davis, Finding ellipse using the genealized Hough transformation, *Pattern Recognit. Lett.* **9**, No. 1, 1989, 87–96.
5. T. Ellis, A. Abboot, and B. Brillault, Ellipse detection and matching with uncertainty, *Image and Vision Comput.* **10**, No. 5. 1992, 271–276.
6. G. Farin, Curvature continuity and offsets for piecewise conics, *ACM Transactions on Graphics* **8**, No. 2, 1989, 89–99.
7. M. Fischler and H. Wolf, Locating perceptually salient points on planar curves, *IEEE Trans. Pattern Anal. Mach. Intell*. **16**, No. 2, 1994, 113–129.
8. M. Floater, High-order approximation of conic sections by quadratic splines, *Comput. Aided Geom. Design* **12**, 1995, 617–637.
9. J. Hu and T. Pavlidis, Function plotting using conic splines, *IEEE Comput. Graphics Appl*. 1991, 89–94.
10. C. L. Huang, Elliptical feature extraction via an improve Hough transform, *Pattern Recognit. Lett.* **10**, No. 2, 1989, 93–100.
11. K. Kanatani, Statistical bias of conic fitting and renormalization, *IEEE Trans. Pattern Anal. Mach. Intell*. **16**, No. 3, 1994, 320–326.
12. Y. Kanazawa and K. Kanatani, Optimal conic fitting and reliability evaluation, *IEICE Trans. Inf. Systems* **E79-D**, No. 9, 1996, 1323–1328.
13. G. Medioni and Y. Yasumoto, Corner detection and curve representation using cubic B-splines, *Comput. Vision, Graphics Image Process.* **39**, 1987, 267–278.
14. T. Pavlidis, Curve fitting with conic splines, *ACM Transactions on Graphics* **2**, No. 1, 1983, 1–31.
15. M. Richetin, Recognition of conics in contours using their geometrical properties, in *Proceedings of CVPR '85*, 1985, pp. 464–469.
16. P. L. Rosin, Ellipse fitting by accumulating five-point fits, *Pattern Recognit. Lett.* **14**, No. 8, 1993, 661–669.
17. P. L. Rosin, A note on the least squares fitting of ellipse, *Pattern Recognit. Lett.* **14**, No. 10, 1993, 799–808.
18. S. Safaee-Rad, I. Tchoukanov, B. Benhabib, and K. C. Smith, Accurate parameter estimation of quadratic curves from grey-level images, *CVGIP: Image Understanding* **54**, No. 2, 1991, 259–274.
19. P. Sampson, Fitting conic sections to "Very Scattered" data: An iterative refinement of the bookstein algorithm, *Comput. Graphics Image Process.* **18**, 1982, 97–108.
20. E. Saund, Identifying salient circular arcs on curves, *Cvgip: Image Understand.* **58**, No. 3, 1993, 327–337.
21. L. Shao and H. Zhou, Curve fitting with Bezier cubics, *Graph. Models Image Process.* **58**, No. 3, 1996, 223–232.
22. M. Werman and Z. Geyzel, Fitting a second degree curve in the presence of error, *IEEE Trans. Pattern Anal. Mach. Intell.* **17**, No. 2, 1989, 207–211.
23. J. A. Wiseman and P. R. Wilson, A sylvester theorem for conic sections, *J. Disc. Comput. Geometry* **3**, 1988, 295–305.
24. P. Wilson, Conic representations for shape description, *IEEE Comput. Graph. and Appl.*, 1987, 23–29.
25. Z. Zhang, Parameter estimation techniques: A tutorial with application to conic fitting, *Image and Vision Comput.* **15**, 1997, 59–76.
26. P. Zhu and P. Chirlian, On critical point detection of digital shapes, *IEEE Trans. Pattern Anal. Mach. Intell.* **17**, No. 8, 1995, 337–748.
27. Q. Zhu, Virtual edges, viewing facts, and boundary traversal in line drawing representation of objects with curved surfaces, *Comput. & Graphics* **15**, No. 2, 1991, 161–173.
28. Q. Zhu, M. Payne, and V. Riordan, Edge linking by a directional potential function (DPF), *Image and Vision Comput.* **14**, 1996, 59–70.