World Academy of Science, Engineering and Technology
International Journal of Electrical and Computer Engineering
Vol:2, No:11, 2008

Template-Based Object Detection through Partial Shape Matching and Boundary Verification

Feng Ge, Tiecheng Liu, Song Wang, Joachim Stahl

***Abstract*—**This paper presents a novel template-based method to detect objects of interest from real images by shape matching. To locate a target object that has a similar shape to a given template boundary, the proposed method integrates three components: contour grouping, partial shape matching, and boundary verification. In the first component, low-level image features, including edges and corners, are grouped into a set of perceptually salient closed contours using an extended ratio-contour algorithm. In the second component, we develop a partial shape matching algorithm to identify the fractions of detected contours that partly match given template boundaries. Specifically, we represent template boundaries and detected contours using landmarks, and apply a greedy algorithm to search the matched landmark subsequences. For each matched fraction between a template and a detected contour, we estimate an affine transform that transforms the whole template into a hypothetic boundary. In the third component, we provide an efficient algorithm based on oriented edge lists to determine the target boundary from the hypothetic boundaries by checking each of them against image edges. We evaluate the proposed method on recognizing and localizing 12 template leaves in a data set of real images with clutter back-grounds, illumination variations, occlusions, and image noises. The experiments demonstrate the high performance of our proposed method[1].

***Keywords*—**Object detection, shape matching, contour grouping.

I. INTRODUCTION

OBJECT detection in noisy images is one of the fundamental problems in computer vision. The template based object-detection problem is defined as follows: Given a template of the desirable object, which can be a boundary, a set of feature points, or a learned pattern, locate in an input image the target object that has similar features to the given template. It is well known that detecting objects from real images is extremely difficult even when the template shape is given, due to image noise, occlusion and distortion of target objects, and excessive computational complexity. Therefore, a large amount of works are limited to detecting and matching objects in synthesized shape databases [14] or in special image domains such as military target recognition [13]. Previous works [3, 6, 7, 9, 12, 5, 13, 1, 16] on template-based object detection extract low-level image features such as edges, corners, regions and interest points, and then locate the target boundary by grouping the features and matching them to those on the template. In these works, computational complexity usually is very high, and object occlusion poses a great challenge to the detection algorithms. In [5], space and time bounds on indexing 3D models from 2D images are analyzed in model-based visual recognition systems. It also shows that an index space can be a powerful tool for reducing the image-model match search, but only when accompanied by some mechanism, such as grouping, that prevents the system from having to consider all matches between image group and model group. In [1], the graphical template is proposed to deal with model registration. Landmarks are extracted by local operators to form image graph model, a dynamic programming algorithm on decomposable sub-graphs of the template graph finds the optimal match to a subset of the candidate points in polynomial time. However, this method fails if one of the true locations is not identified, and it is not clear how to construct the graph for matching.

In [13], short edges are extracted from the input image as compact features. With an initial estimate of the target boundary, a gradient-descent algorithm is introduced to compute a locally optimal transform between the template and the target boundaries. This approach, also called Chamfer matching in some literatures [4], relies heavily on the initial estimate of the target boundary, and its performance is degraded by occlusions and false edges resulting from the image noise. In [3, 9, 11], a set of interest points that are invariant to affine transforms are extracted as image features, and object detection is achieved by matching these local features to the template features and evaluating a global transformation model. In these methods, the templates are real images instead of object boundaries, and occlusions affect the performance negatively.

Another related work is the object-detection method developed in [6]. By assuming at least one line segment in the target boundary is accurately and fully detected, exhaustive search, by pairing each model line segment to each detected line segment, is applied to find the target boundary that optimally matches the image information. In [7, 16], the input image is first segmented into small regions with homogeneous intensity, then some random-optimization techniques are used to merge a subset of regions into a larger region and to match the boundary of this larger region to the template boundary. These approaches also have difficulties in dealing with partial occlusions of the target boundaries.

[1] Feng Ge is with the Department of Electrical and Computer Engineering in Virginia Tech, Blacksburg, VA, 24060, USA (e-mail:gef@vt.edu). This work is done when the first author was in University of South Carolina. Tiecheng Liu, Song Wang, and Joachim Stahl are with Department of Computer Science and Engineering in University of South Carolina, Columbia, SC 29208 USA. (e-mail:{tiecheng|songwang|stahlj}@cse.sc.edu).

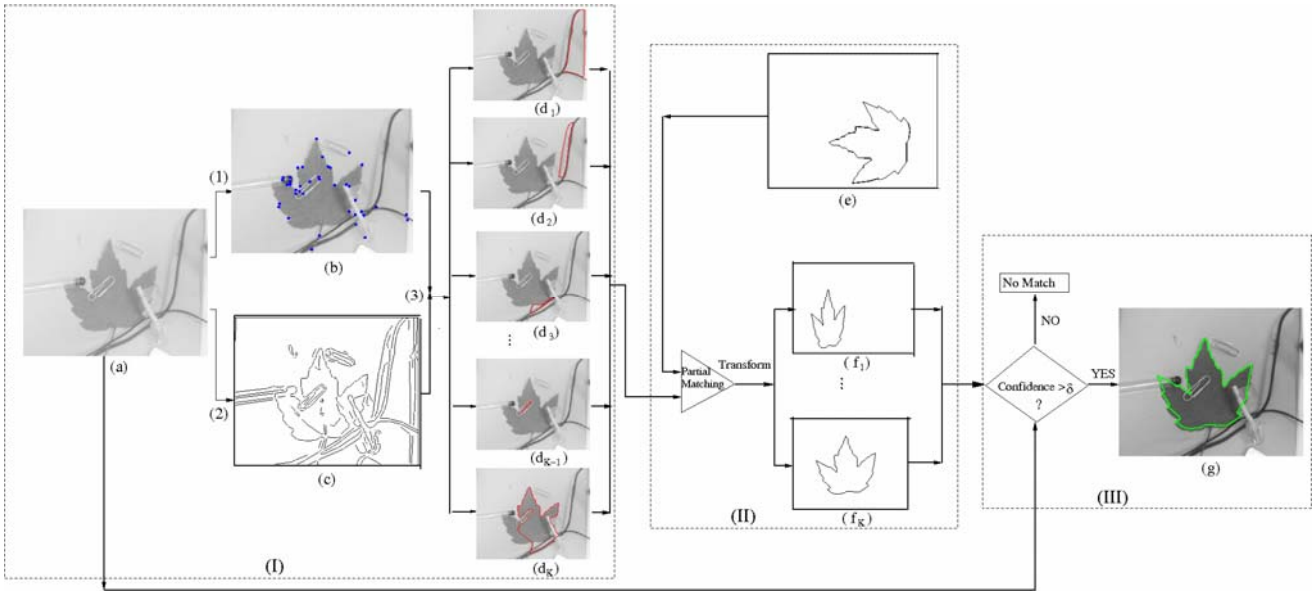

Fig. 1 An illustration of the proposed object-detection process. The three major components are (I) Contour grouping, (II) Partial shape matching, and (III) Boundary verification. In the figure, (a)-(g) are: (a) an original image; (b) detected corners (in blue); (c) detected line segments; (d_1, … , d_K) K contours grouped from image features (in red); (e) a template boundary; (f_1, …, f_K) K hypothetic boundaries; (g) the detected object (in green).

In this paper, we are targeting the object detection problem where only the boundary information of the template is available. We aim to address this specific problem by identifying a set of perceptually-salient closed contours, a typical mid-level vision feature that improves boundary detection [15]. Particularly, these salient closed contours well cover the major structures in the input image and are identified by considering various Gestalt laws, such as proximity, closure, and continuity. This way, the template boundary and the middle-level features extracted from the input image are of the same form: a closed contour. As contour matching can be converted to the matching of landmark sequences, we expect that the matching is more efficient and reliable.

Obviously, it is too strong an assumption that one of the identified contours is exactly the desirable target boundary. Instead, we only assume that some fractions of the extracted salient contours match a fraction of the target boundary, which is actually supported by our experimental observations. Also considering that the target objects may be occluded in real images, we conduct a partial shape matching between the template boundary and the extracted contours to detect the target boundary. To make the partial shape matching more efficient and robust, we represent contours using landmarks and develop a greedy algorithm to search the matched landmark subsequence of contours. The matching is invariant to translation, rotation, scaling, and reflection transforms. Based on the matched contour fractions, we construct a set of hypothetic target boundaries, from which we locate the desirable target boundary by checking each of them against the original image information.

The proposed object-detection method consists of three important components: (I) *contour grouping* for identifying perceptually salient contours from the input image, (II) *partial shape matching* for constructing a set of hypothetic boundaries for the target boundary, and (III) *boundary verification* for locating the final desirable target boundary. Figure 1 illustrates the proposed object detection process. Given an input image (Figure 1 (a)), we first retrieve image features such as corners and lines using standard corner detection and line fitting methods [8, 10], then we group these features into perceptually-salient and closed contours, as shown in images d_1, …, d_K of Figure 1. Based on the identified salient contours, we perform partial shape matching to identify the contour fractions that match the template shapes. From the identified contour fractions, we compute their related affine transforms and construct a set of hypothetic boundaries, as shown in images f_1, …, f_K of Figure 1. Finally, we verify each hypothetic boundary against the input image to locate the final target boundary.

I. Contour Grouping

By considering the important Gestalt laws of proximity, closure, and continuity, the ratio-contour algorithm developed in [17] is a globally optimal contour grouping method that detects perceptually salient contours from real images. However, many salient objects' boundaries are heavily determined by corner information besides continuity, proximity and closure, which are excluded by ratio-contour's results. Therefore, in this paper, we extend the ratio-contour algorithm to combine corner and edge information for salient contour detection.

In the ratio-contour algorithm, a set of line fragments are first detected from the input image using edge detection and line fitting. A contour Γ is then constructed by sequentially connecting a subset of these line fragments with some gap lines into a closed cycle, which satisfies a minimum cost. The cost of this contour (negatively related to its saliency) is

defined by

$$\phi(\Gamma) = \frac{|\Gamma_G| + \lambda \int_{\Gamma} \kappa^2(t)\,dt}{|\Gamma|} \quad (1)$$

where $|\Gamma_G|$ is the total length of the filled gap in connecting line fragments into the closed contour. This term reflects the preference of a contour with good proximity. $\kappa(t)$ is the curvature along the contour. It reflects the preference of a contour with good continuity. The curvature and continuity are actually derived from a smoothed version of Γ which uses Bezier-curve interpolation when connecting two line fragments. $\lambda > 0$ is a regularization factor that balances the proximity and continuity. The normalization over the contour length $|\Gamma|$ in (1) makes the contour grouping unbiased to the contour length. In [17], a globally optimal ratio contour algorithm is developed to find the optimal boundary with the minimum cost (1).

To incorporate the corner information, we introduce a corresponding factor in contour cost (1), which becomes

$$\phi(\Gamma) = \frac{|\Gamma_G| + \lambda \int_{\Gamma} \kappa^2(t)\psi(t)\,dt}{|\Gamma|} \quad (2)$$

where $\psi(t) = 1$ if no corner points are detected near the considered contour point, and $\psi(t) = 0$ otherwise. Given that two line fragments are connected by Bezier-curve interpolation, this cost function indicates that corner points will not introduce high cost for its high curvature. In this paper, we use the Harris corner detector [8] to detect the corners. Since corners are local features, the efficient graph based optimization algorithm developed in [17] can still be used to find the globally optimal contour that minimizes this new cost (2).

With cost (2), we can detect multiple salient boundaries by repeatedly applying it on a set of line fragments and corners and finding the next minimum cycle. However, each real image may contain several salient objects, it is possible that none of a limited number of contours derived from the ratio-contour algorithm matches the desirable target boundary (with the similar shape to the template boundary). In this work, we make a weak assumption: *After iteratively applying the ratio-contour algorithm to detect K salient contours in the image, at least one of these K contours contains a non-trivial fraction of the target boundary, given that K is sufficiently large.*

This assumption is supported by our experiment. In the experiment, we collected 120 real images that always contain an identified salient structure. The boundary of the identified structure in each image is the target boundary. For example, the identified structures in some images are a fish and a pot, as shown in Figure 2. Then we repeat the extended ratio-contour algorithm on each image to detect the first K optimal contours. And we check the percentage of the target boundary that each contour catches. As shown in Figure 3, the experimental results indicate that when 10 contours are detected in each image, more than 96% of the images produce at least one contour that contains at least 20% of the target boundary.

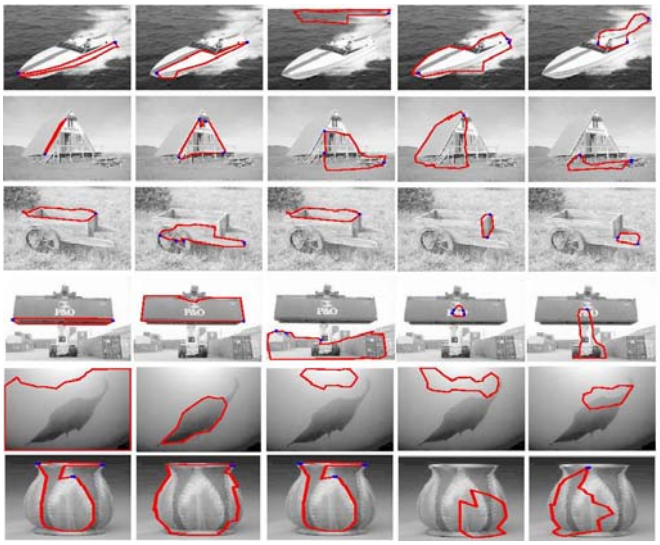

Fig. 2 The first five contours detected from each of the six sample images using the extended ratio contour algorithm. Blue points are the corner points included in the detected contours.

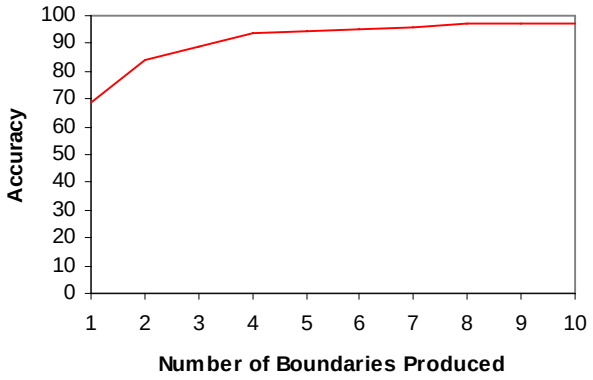

Fig. 3 The percentage of the 120 images, in which at least one out of the K detected contours catches 20% of the target boundary. The curve is obtained by varying K from 1 to 10 continuously.

II. Partial Shape Matching

After we detect K contours by ratio-contour with at least one containing part of the template boundary, we need a partial matching method to identify a correspondence between the detected contour and the template boundary. Let Γ_t, t = 1, …, K be the K contours detected by ratio contour and Ψ be the template boundary. In this component, we develop a partial matching method to measure the similarity between Γ_t and Ψ by finding the possible matched fractions between each contour and the template boundary. Direct matching of all the points in two shapes is both difficult and computationally expensive. Here, we develop a landmark-based algorithm to achieve this goal. In this algorithm, two sequences of landmark points are first sampled from the template boundary Ψ and the considered contour Γ_t, respectively. We then compare these two landmark sequences to identify the best matched subsequence. Particularly, we adopt a transform-invariant metric called "sphericity" to measure the matching

accuracy. The resulting matching is invariant to some affine transforms, including translation, rotation, scaling, and reflection. In addition, the resulting matching can tolerate a certain amount of location variation of the landmarks, which are common in practice.

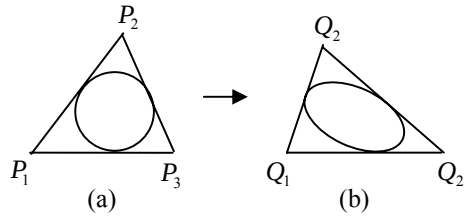

Fig. 4 Use sphericity to measure the mapping from one triangle to another triangle. (a)The original triangle with its inscribed sphere. (b) The transformed triangle with the transformed sphere. This Figure is adapted from Fig.3 in [2]

A. Sphericity and Triangle Matching

"Sphericity" [2] is a measure defined for the matching of two triangles Δ_{P_1,P_2,P_3} and Δ_{Q_1,Q_2,Q_3}. As shown in Fig. 4, an affine transform can be derived to map P_i to Q_i, i = 1, 2, 3. Applying this transform to the circle inscribed in Δ_{P_1,P_2,P_3} brings to us an ellipse inscribed in Δ_{Q_1,Q_2,Q_3}, as shown in Fig. 4(b). The "sphericity" is related to the ratio between the two principle axes of the ellipse. This "sphericity" reflects the "distortion" of the transform and is a suitable local measure of the shape-matching accuracy [2]. Particularly, it is invariant to the transforms of translation, rotation, and scaling. The sphericity of triangle matching can be computed efficiently, as follows. Given two triangles Δ_{P_1,P_2,P_3} and Δ_{Q_1,Q_2,Q_3}, there exists an affine transform T to map one triangle to another:

$$\begin{bmatrix} u \\ 1 \end{bmatrix} = T \begin{bmatrix} x \\ 1 \end{bmatrix} = \begin{bmatrix} A & t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ 1 \end{bmatrix}, \quad (3)$$

where

$$A = \begin{bmatrix} a & b \\ c & d \end{bmatrix}, \quad t = \begin{bmatrix} e \\ f \end{bmatrix},$$

and u and x are the coordinates of the vertices in Δ_{P_1,P_2,P_3} and Δ_{Q_1,Q_2,Q_3}, respectively. The affine transform T can be determined by solving the linear equations (3), and the sphericity of the mapping is then computed as

$$Spericity = 1 - 2(t_2^2 + t_4^2)/(t_1^2 + t_2^2 + t_3^2 + t_4^2) \quad (4)$$

where

$$t_1 = a + d, t_2 = a - d, t_3 = b - c, and \quad t_4 = b + c$$

The value of sphericity is in the range of [-1, 1]. A higher absolute value indicates a better matching. When the sphericity value is 1, only translation, rotation, and scaling transforms exist. The less the absolute sphericity value is, the more transformation distortion there is. Negative values mean that reflection transforms exist.

B. The Model of the Partial Shape Matching

Consider the partial shape matching between the template boundary Ψ to a detected contour Γ, where Ψ has a sequence of n landmarks $\Psi = \{P_i\}$; i = 1, …, n and Γ has a sequence of m landmarks $\Gamma = \{Q_i\}$; j = 1, …, m Every three consecutive landmarks in either Ψ or Γ form a triangle, e.g., $\{P_{i-1}, P_i, P_{i+1}\} \rightarrow \Delta_{P_{i-1},P_i,P_{i+1}}$. In this paper, we denote a triangle formed by three consecutive landmarks as a *consecutive triangle* and the sequence of all such consecutive triangles constructed from a sequence of landmarks as a *consecutive triangle sequence*. Since we only consider closed contours and boundaries, the last landmark in the sequence is the neighbor of the first landmark. P_n is adjacent to P_1, and $\Delta_{P_{n-1},P_n,P_{n+1}}$ is also a consecutive triangle. Therefore, with n landmarks in a boundary, we can construct n consecutive triangles. The goal of partial shape matching is to find the longest landmark subsequences $\Psi' \subset \Psi$ and $\Gamma' \subset \Gamma$ such that the consecutive triangle sequence constructed from Ψ' matches that constructed from Γ'.

As shown in Figure 5, there are three shape matching scenarios. (1) The landmarks on the detected contour are transform-equivalent to those in the template shape. This is the ideal case of shape matching (no occlusion and no false-positive landmarks), and the matching can be easily solved by aligning these two landmark sequences. However, most real-world matching problems do not belong to this category. (2) A more general matching problem is the partial matching with occlusion. As shown in Figure 5(b), the candidate contour is partially occluded, but no irrelevant landmarks are detected. In this case, a subsequence of Ψ is transform-equivalent to Γ. Thus this matching problem can be solved using a m-step dynamic programming process, like the one provided by Ansari and Delp [2]. (3) Most shape matching problems in real images involve both occlusion and false-positive landmarks. In this case, there exist two landmark subsequences $\Psi' \subset \Psi$ and $\Gamma' \subset \Gamma$ such that Ψ' matches Γ'. As shown in Figure 5 (c), the landmark subsequence $\{P_1, P_2, P_3, P_4, P_5, P_6, P_7\}$ matches the subsequence $\{Q_1, Q_2, Q_4, Q_5, Q_7, Q_8, Q_{10}\}$.

In this paper, we propose an approach to solving the partial shape matching problem where the detected contour is partly occluded and contains irrelevant contour parts.

In [2], a dynamic-programming-based algorithm is developed to solve a partial-matching problem. This algorithm, however, cannot solve the above matching problem due to two reasons. First, the dynamic programming algorithm in [2] needs a prior selection of the starting landmark and ending landmark, both of which are unknown. Second, the dynamic-programming algorithm cannot handle the case where

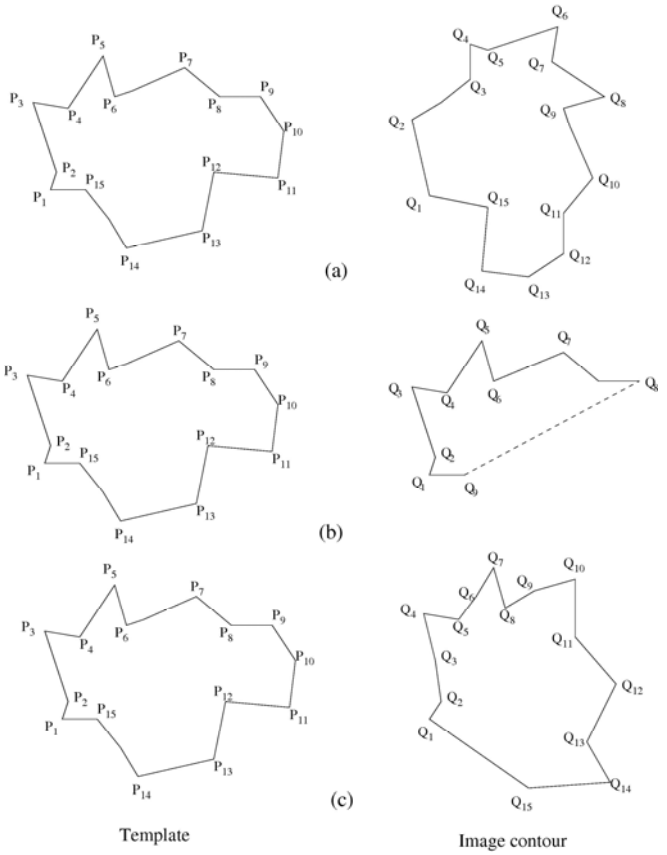

Fig. 5 Three kinds of shape matching problems. (a)The detected contour has the same landmarks as those in the template shape. (b) The detected contour is occluded but no false landmarks are detected. (c) The detected contour has both occluded and false landmarks. Most partial shape matching problems in real images belong to the third category.

the contour landmarks are not exactly a subset of the landmarks of the template boundary. The false-positive landmarks on the contours caused by image noise also degrade the performance of dynamic programming.

C. Landmark Sampling and Matching Table

After reducing partial shape matching to the matching of consecutive triangles, we still need to handle the difficult case where a consecutive triangle in Ψ may not match a consecutive triangle in Γ. For example, in Fig.6, $\Delta_{P_{i\_1},P_i,P_{i+1}}$ matches $\Delta_{Q_{j-4},Q_j,Q_{j+4}}$ instead of $\Delta_{Q_{j-1},Q_j,Q_{j+1}}$. We solve this problem from two aspects. First, we extract template landmarks by including only the most salient feature points, such as the corners and higher curvature points, but retrieve comparatively more contour landmarks with dense sampling. This way, when a contour matches the template boundary Ψ, there always exist some consecutive triangles on Ψ that match the triangles formed by the contour landmarks. Second, we build a landmark matching table that measures the matching from each consecutive triangle in Ψ to $k \times k$ triangles of contour landmarks, and develop an efficient greedy algorithm to search the best matching triangle sequence.

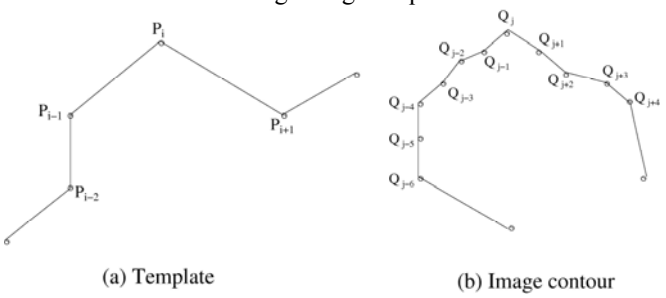

Fig. 6 An illustration of the triangle matching at the (i, j)-th entry in the matching table.

The landmark matching table Π is constructed to have $n \times m$ entries, where each entry $\Pi_{i,j}$ is a $k \times k$ matrix, as shown in Figure 7 (where k = 2). The matrix $\Pi_{i,j}$ describes the local matching between the pair of landmarks P_i and Q_j. As shown in Figure 6, $\Pi_{i,j}^{s,t}$, the (s, t)-th element in matrix $\Pi_{i,j}$ is the sphericity between the triangles $\Delta_{P_{i\_1},P_i,P_{i+1}}$ and $\Delta_{Q_{j-s},Q_j,Q_{j+t}}$ with $1 \le s, t \le k$. We can see that this table considers the case where the neighboring landmarks in the template boundary may correspond to non-neighboring landmarks in the detected contours, i.e., there are more landmarks sampled in the detected contours. Our later experiments indicate that k = 5 is sufficiently large to handle the problem of the different landmark sampling rate in the template boundary and the detected contour.

D. Partial Shape Matching Algorithm

Based on the matching table Π, we develop a greedy algorithm to search the matched subsequences of landmarks in Ψ and Γ. Such matched subsequences correspond to a matching path in the matching table Π. Two parameters δ_H and δ_L are introduced to control this searching process. The higher threshold, δ_H, is used for locating the starting point of the path; the lower threshold, δ_L, is used for determining whether we will continue tracing the path.

Particularly, in the $k \times k$ matrix $\Pi_{i,j}$, we find the element with the maximum magnitude by

$$|\Pi_{i,j}^{s',t'}| = \max_{1 \le s \le k} \max_{1 \le t \le k} |\Pi_{i,j}^{s,t}|$$

If $|\Pi_{i,j}^{s',t'}| > \delta_H$, the (i, j)-th entry is selected as a start point for path search.

Starting from each entry identified as a start point, a greedy search is applied to locate the whole path. Without loss of generality, consider starting from entry $\Pi_{i,j}$ with $\Pi_{i,j}^{s',t'} > \delta_H > 0$. This indicates that $\Delta_{P_{i-1},P_i,P_{i+1}}$ well matches $\Delta_{Q_{j-s'},Q_j,Q_{j+t'}}$ with good sphericity. We continue the matching path in both backward and forward directions.

Forward Search: When $\Delta_{P_{i-1},P_i,P_{i+1}}$ matches $\Delta_{Q_{j-s'},Q_j,Q_{j+t'}}$, we

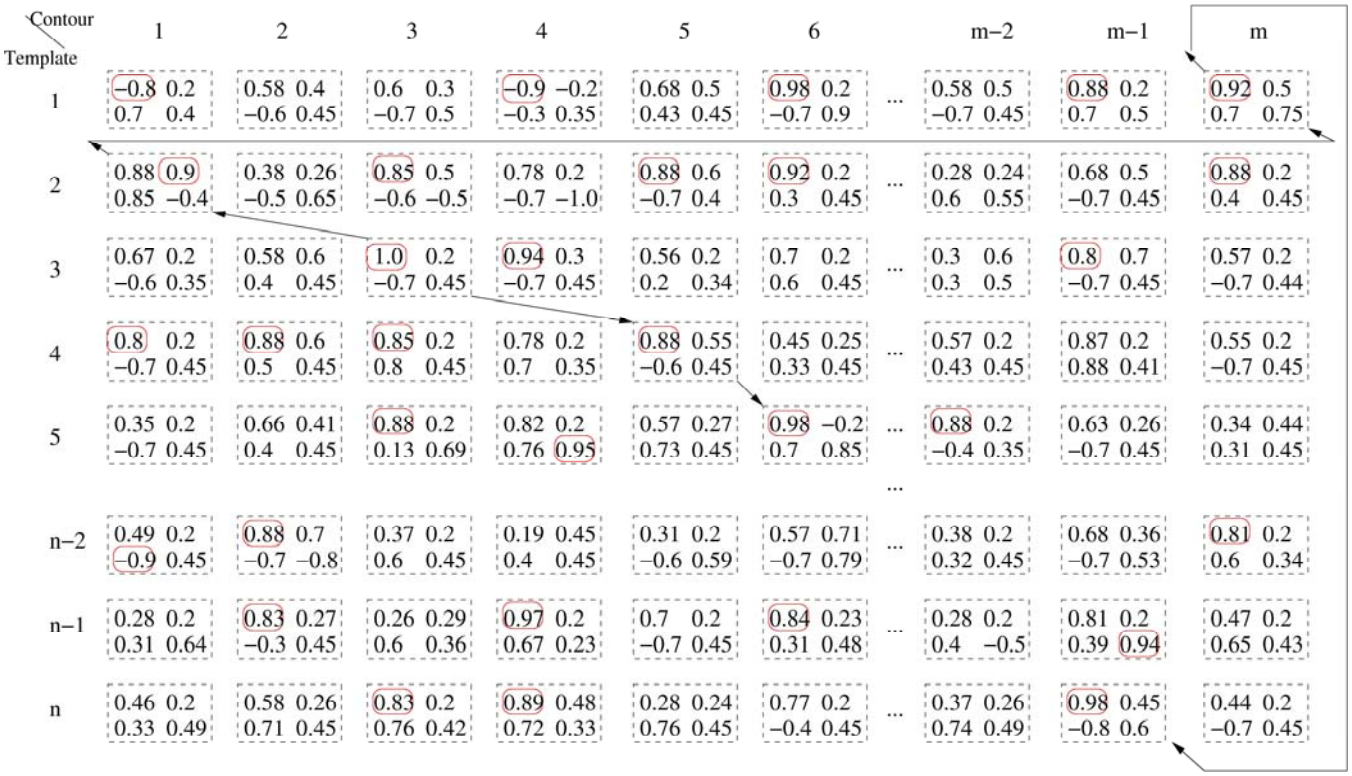

Fig. 7 An illustration of the greedy path searching algorithm. Each entry has 4 elements, i.e. k = 2. The numbers in the circles indicate the entries of the starting points. The arrows show the searching path and direction.

know that the landmark P_{i+1} in Ψ matches the landmark $Q_{j+t'}$ in Γ. So the next entry to be checked is $\Pi_{i+1, j+t'}$. We need to check whether a triangle in $\Delta_{Q_j, Q_{j+t'}, Q_{j+t'+q}}$ $(1 \le q \le k)$ matches $\Delta_{P_i, P_{i+1}, P_{i+2}}$. This can be easily done by inspecting the k elements in the t'-th row of the matrix $\Pi_{i+1, j+t'}$. If the maximum one among these k elements is larger than δ_L, we include this element into the path and continue from it to search the next one by repeating this procedure. Otherwise, the forward search of this path starting from the entry (i, j) terminates.

Note that this path search continues to the other end of the matching table when reaching the boundary of the table, as shown in Fig. 7, because the boundary/contour is closed. In addition, if at the start point of the path we have a negative sphericity value, a new entry will be included only when its sphericity is less than - δ_L. In this case, we expect a reflection transform in the shape matching.

Backward Search: Backward search follows the same procedure as that of the forward search. The only difference is that we locate the next entry (i-1, j-s') with $s' > 0$. Note that both forward and backward search terminate when the length of the matching path is longer than $\min\{m, n\}$.

Figure 7 shows an example this greedy search, where δ_H = 0.9 and δ_L = 0.7. The matching path starts with the entry (3, 3). The forward search ends at the entry (5, 6) as the maximum sphericity values of the next entry is less the δ_L. The backward search continues until the sphericity is lower than δ_L. The resulting path is {(n, m-1), (1,m), (2, 1), (3, 3), (4, 5), (5, 6)}, which corresponds to the subsequences of the matched landmarks { P_n, P_1, P_2, P_3, P_4, P_5} ⊂ Ψ and { Q_{m-1}, Q_m, Q_1, Q_3, Q_5, Q_6} ⊂ Γ.

For each start entry, we extract one matching path using the greedy search algorithm. Among the matching paths starting from different entries, we pick the longest one to retrieve the matched landmark subsequences in the template boundary and the detected contour. From the matched landmarks, we estimate an affine transform from the template boundary to the detected contour by a least-square estimation in equation 3.

A. Computational complexity

The storage usage of the algorithm is obviously $O(mnk^2)$. As for computational time, building the sphericity matching table takes mnk^2 times of sphericity computation; finding the maximum values in all the entries takes mnk^2 comparisons. Since there are at most mn support entries and each support entry has only one matching path, finding the best matching path takes a maximum of $mnk\max\{m,n\}$ times of comparison computation. In real computation, the number of support entry is far less than mn because of the thresholds, and the length of the matching path is usually much less than m and n. So the average-case computation is low. Our experiments show the greedy path searching is computationally efficient.

IV. Boundary Verification

For each candidate contour Γ, the greedy searching method produces a longest matching path. Then a post-processing step evaluates the longest matching paths and chooses the best one when multiple candidate contours exist. The best extracted contour is further compared with the original edge map to verify whether it really matches the object. In our work, a simplified oriented-edge matching algorithm is developed to verify the candidate contours. To measure whether a shape Ψ exists in an image Φ, we compute a matching confidence value and compare it with a predetermined threshold.

For a candidate contour Γ, we calculate the affine transform T based on Eqn. 3. The template shape Ψ is then transformed using T. Let Ψ_T be the transform shape. Each line segment in Ψ_T is sampled every 5 pixels. For the original image Φ, we compute the edge map of I and represent the edges using line segments. Let I be the line approximation of the edge map of image Φ. We measure the distance from each line segment in Ψ_T to the line segments in I. To reduce the negative effect of image noise, we remove all the edges with length less than 10 pixels.

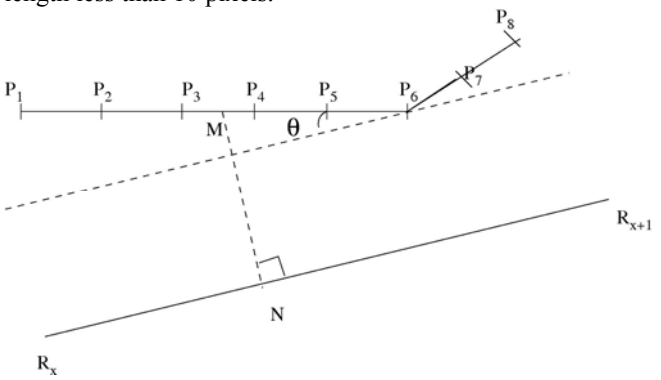

Fig. 8 Illustration of the verification process. The lines on template shape is divided into small line segments, and the distances between the line segments of the template shape and those of the edge map are measured.

The verification is based on line segments for several reasons. (1) The candidate contour may contain false-positive contour part. Thus the original edge information of the image is required. (2) Measuring the proximity of line segments is more computationally efficient than matching all edge points. (3) The line-segment-based verification is more accurate than the landmark-based one because landmarks are sampled sparsely in the template shape.

Suppose Ψ_T is divided into h small line segments $\ell^{(i)}$, i = 1, 2, … , h. Define as the distance from $d^{(i)}$ to the edge map.

$$d^{(i)} = \min_{\ell \in I} \| \ell^{(i)} - \ell \|$$

where $\| \ell^{(i)} - \ell \|$ is the distance between line segment $\ell^{(i)}$ and ℓ The distance between two line segments is computed in the following way. As shown in Figure 8, we project the center point M of line segment $\ell^{(i)}$ (P_3P_4 here) onto the line segment ℓ (R_xR_{x+1} here). If the projection, N, is on ℓ, we define the distance between M and N as the distance between $\ell^{(i)}$ and ℓ; if the projection is not on ℓ, we define the smaller one of $|\overline{MR_x}|$ and $|\overline{MR_{x+1}}|$ as their distance, where R_x and R_{x+1} are the two end points ℓ and | . | measures the length of a line segment.

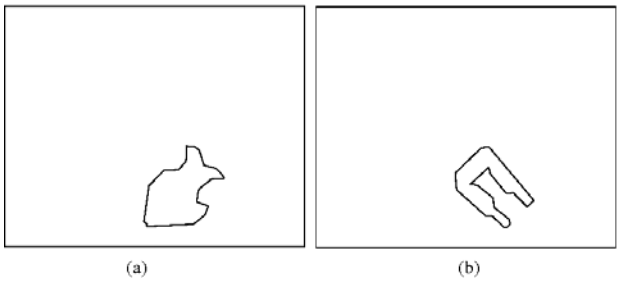

Fig. 9 Two toy templates.

As shown in Figure 8, $\overline{P_1P_6}$ is a line segment in the transformed template shape. We divide that line into small segments with 5 pixel length each. To measure the distance from a line segment $\overline{P_3P_4}$ in template to a line segment $\overline{R_xR_{x+1}}$ in I, we first find the middle point M of $\overline{P_3P_4}$, then project M onto $\overline{R_xR_{x+1}}$. Since the projection N is on the line segment $\overline{R_xR_{x+1}}$, d = $|\overline{MN}|$ is the distance between $\overline{P_3P_4}$ and $\overline{R_xR_{x+1}}$. The angle between these two line segments is also calculated, in a straightforward way.

For a line segment $\ell^{(i)}$ on the transformed template shape, we find on I the line segment that is closest to $\ell^{(i)}$. Assume ℓ^* is the line segment closest to $\ell^{(i)}$, i.e., $\| \ell^{(i)} - \ell^* \| = \min_{\ell \in I} \| \ell^{(i)} - \ell \|$. We measure the angle between $\ell^{(i)}$ and ℓ^*. Denote that angle as $\theta^{(i)}$. For the candidate contour Γ, we compute the transform T. Let Ψ_T be the template shape transformed by the transform T. We compute a matching probability for each contour:

$$\Pr(\Psi_T) = \frac{\sum_{i, d^{(i)} > \delta_d, \theta^{(i)} > \delta_\theta} |\ell^{(i)}|}{\sum_i |\ell^{(i)}|}$$

where | . | measures the length of the line segments, δ_d and δ_θ are the thresholds for matching the line segments in term of their distance and the angle. In our experiments, we choose the values of δ_d and δ_θ to be 3.0 pixels and 0.30, respectively. This verification confidence can be used to decide whether this hypothetic boundary is the target boundary. In this paper, out of K hypothetic boundaries, we find the one with the largest confidence. If this confidence is larger than 0.6, we choose the corresponding hypothetic boundary as the target boundary; if the largest confidence of all hypothetic boundaries is less than 0.6, we conclude that there is no matching of the template object in the image.

V. Experiments and Discussion

To evaluate the performance of the proposed method, we use the following way to construct samples images. First, we put

two toys in different backgrounds and construct input images. These images were taken by a digital camera and normalized to the resolution of 300 x 240. In Fig. 10, we show 10 such sample images. In the data collection, we did not fix the location and perspective of the camera. In addition, these images were taken under different illumination conditions. In some of these images, we intentionally introduce some occlusions on the toys, such as the images (c), (d) and (j) shown in Figs. 10. For templates, we pick one of the images that have no occlusion and manually extract its boundary by identifying and connecting the corner points. After imposing some random translation, rotation, scaling transforms to both boundaries, we set them as the template boundaries. Two such sample template boundaries are shown in Figure 9.

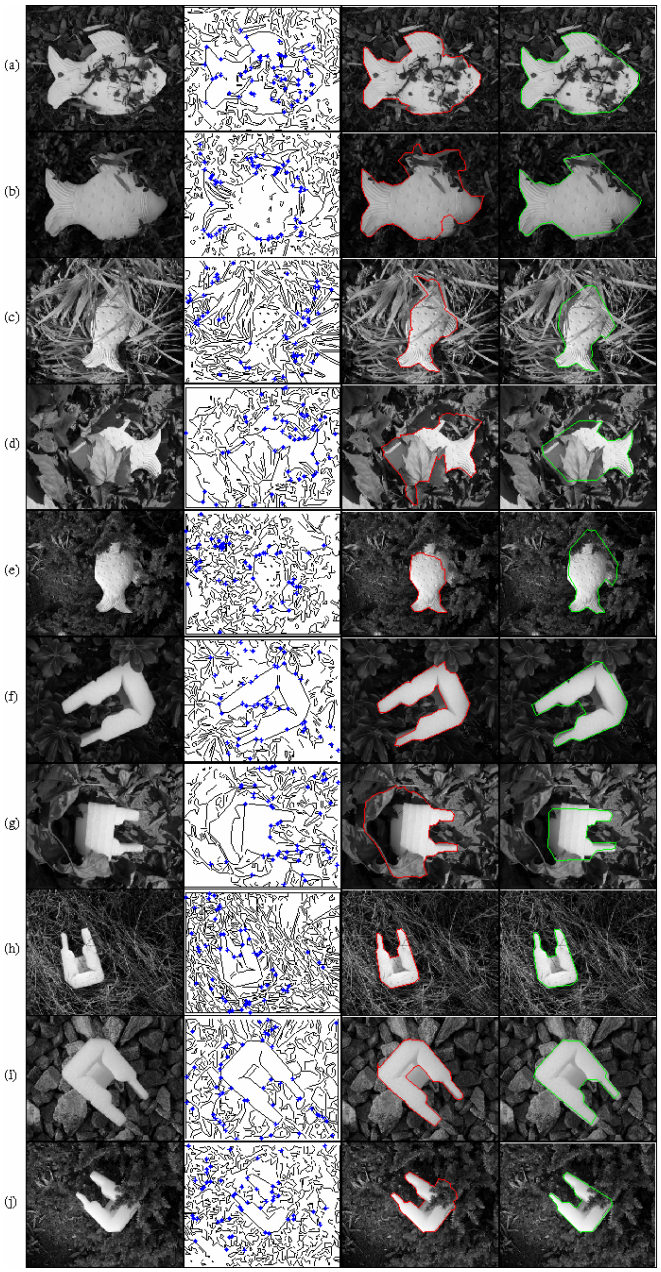

Fig. 10 Object detection results on 10 sample images. Column 1 and 5: 10 original images; column 2 and 6: detected line segments and corners (in blue); column 3 and 7: best extracted contours (in red) using the extended ratio-contour algorithm; column 4 and 8: detected boundaries (in green) in each image.

In implementing the proposed object detection method, we use the Harris corner-detector package by Kovesi [10] to detect the image corners for contour-grouping. With the default parameters in this package for corner detection, each image produces no more than 40 corner points. In our contour grouping, the line segments are generated by the Canny edge detector with default settings in Matlab, followed by a line-fitting process implemented by Kovesi [10]. In our experiments, we only keep the line segments with a length longer than 10 pixels; the segments shorter than 10 pixels are discarded as image noises. In the contour grouping, we always repeat the ratio-contour algorithm to detect the first 10 optimal contours from each image, i.e., the parameter K = 10. In the component of partial shape matching, we set the parameters δ_H and δ_L to be 0.9 and 0.7, respectively. In the boundary verification component, a hypothetic boundary is verified to be the target boundary if its matching confidence is larger than 0.6.Figure 10 illustrates some images from which the proposed method successfully detects the target boundaries. We can see that in almost all of them, the template boundary needs to be scaled, translated, and/or rotated to match the target boundary. The use of "sphericity" as matching measure well handles these transforms. We can also see, from images with clutter background, such as (a) and (c), and images with occlusion, such as (b), (d), (e) and (j), that our contour-grouping algorithm catches substantial fractions of the target boundary, and the later shape matching and boundary verification may recover the desirable boundaries.

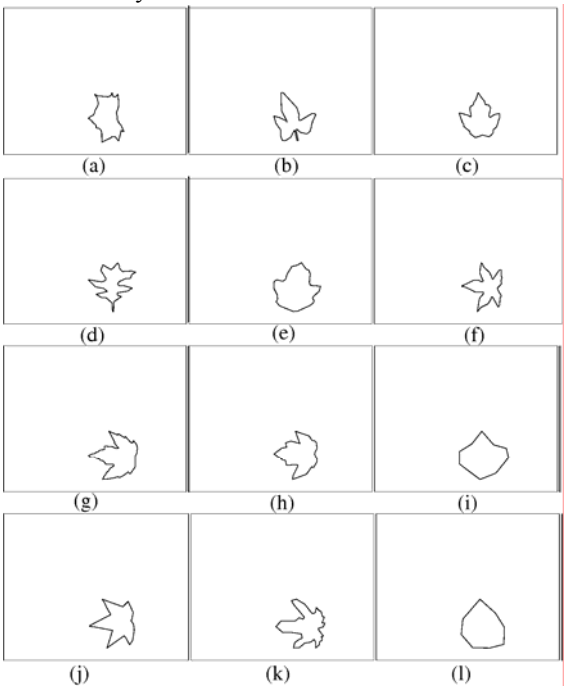

Fig. 11 Twelve template leaf boundaries used in the experiments.

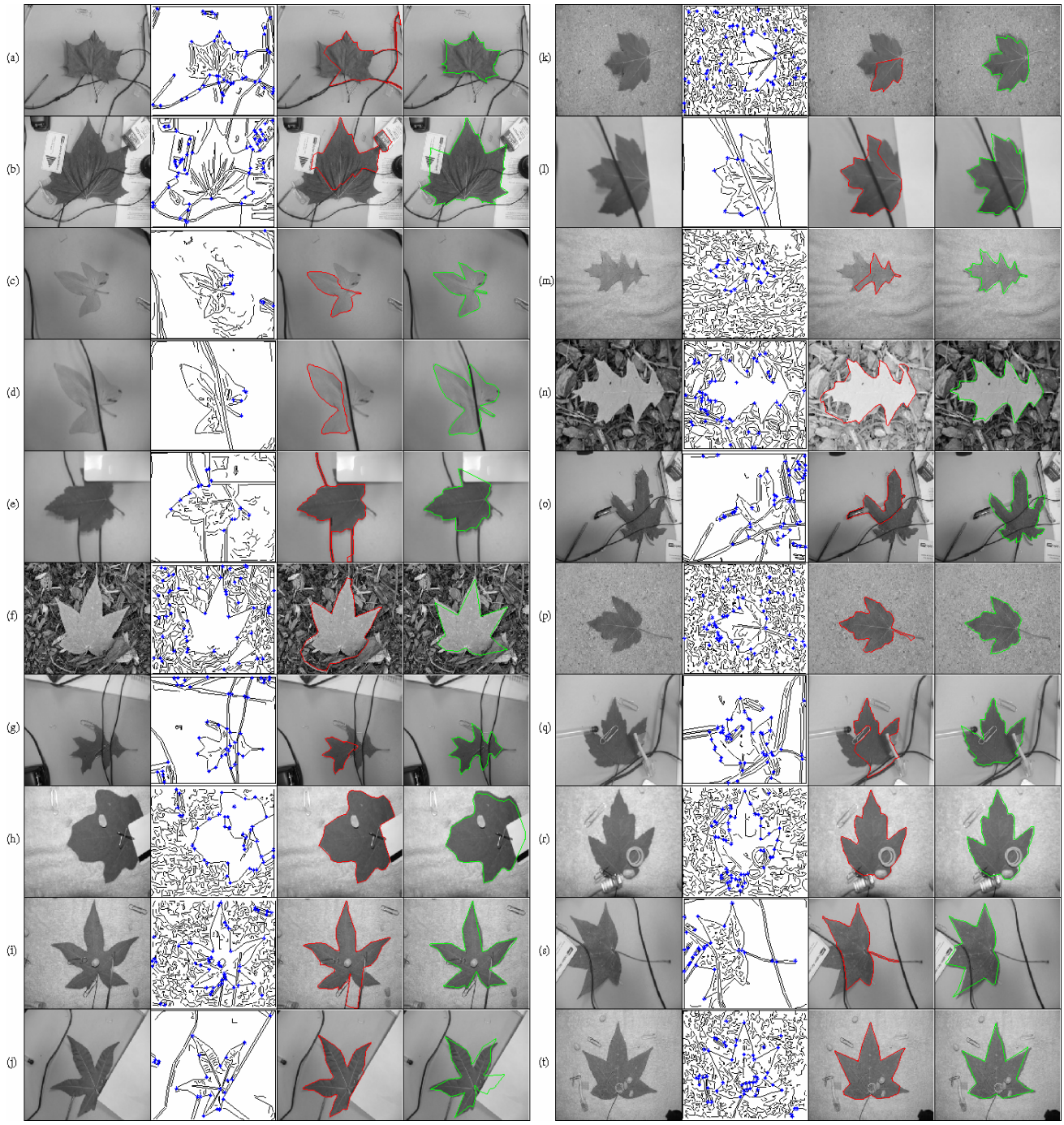

Fig. 12 Object detection results on 20 sample leaf images. Column 1 and 5: 20 original images; column 2 and 6: detected line segments and corners (in blue); column 3 and 7: best extracted contours (in red) using the extended ratio-contour algorithm; column 4 and 8: detected leaf boundaries (in green) in each image.

We also evaluate the shape matching performance using a database of leaf images. We collect 12 leaves with different shapes and put them in different backgrounds to construct 84 leaf images with different settings. 12 template leaf boundaries are shown in Figure 11, and 20 of the leaf images in the database are shown in Figure 12. For each of the 84 collected images, we perform the proposed method to determine whether it contains one of the 12 template leaves. Since we collect the images such that one image only contain one leaf of interest, we can simply compare the matching confidence resulting from each template leaf to achieve the recognition. Among these 84 test images, we find that the proposed method detects the target leaf boundaries in 58 images. In 53 of these 58 images, the detected target boundaries are correctly recognized as one of the 12 template leaves. We also find that the current implementation of the

proposed method produces a very low recall rate (9%) but a relatively high rejection rate (31%). Once the algorithm determines that a target object exists in an image, the target object can be detected and recognized very accurately (91%). These results show that this method may be useful in the applications where the detection accuracy is the primary concern but not all target objects need to be detected. One such example is the template-based image retrieval: It is not necessary to retrieve all of images that contain a certain object, but it is critical not to return many irrelevant images that do not contain this object. In our experiments, the algorithm fails to recognize the template leaves shown in Figure 11 (k) and (l). The major reason is that the method proposed in this paper combines corner and edge information in the partial shape matching but these two leaves show weak corner information. We find that 12 out of the 26 rejected images contain either one of these two special leaves.

VI. CONCLUSIONS

In this paper, we presented an object-detection method to extract the target object boundaries that have a shape similar to a given template. First, we extended the ratio-contour algorithm to globally detect some salient closed contours from the image, by combining both edge and corner information. We then developed a partial shape matching algorithm to identify the fractions of detected contours that partly match the given template boundaries. From each partially matched fraction, we estimated a transform from the template boundary, applied this transform to the whole template, and constructed a hypothetic boundary. Finally, we developed an efficient algorithm to verify each hypothetic boundary against image edges and selected the target boundary based on a verification confidence measure. We evaluated this object-detection method on localizing and recognizing 12 different shape leaves in 84 real images that have clutter backgrounds, illumination variations, occlusions, and image noises. The proposed method correctly detected object boundaries in 53 of the real leaf images.

REFERENCES

[1] Y. Amit and A. Kong. Graphical templates for model registration. IEEE Transactions on Pattern Analysis and Machine Intelligence, 18(3):225–236, 1996.
[2] N. Ansari and E. J. Delp. Partial shape recognition: A landmark-based approach. IEEE Transactions on Pattern Analysis and Machine Intelligence, 12(5):470–483, 1990.
[3] A. C. Berg, T. L. Berg, and J. Malik. Shape matching and object recognition using low distortion correspondence. In IEEE Conference on Computer Vision and Pattern Recognition, 2005.
[4] G. Borgefors. Hierarchical chamfer matching: A parametric edge matching algorithm. IEEE Transactions on Pattern Analysis and Machine Intelligence, 10(6):849–865, 1988.
[5] D. Clemens and D. Jacobs. Space and time bounds on indexing 3d models from 2d images. IEEE Transactions on Pattern Analysis and Machine Intelligence, 13(10):1007–1017, 1991.
[6] P. David and D. DeMenthon. Object recognition in high clutter images using line features. In International Conference on Computer Vision, pages 1581–1588, 2005.
[7] P. F. Felzenszwalb. Representation and detection of deformable shapes. IEEE Transactions on Pattern Analysis and Machine Intelligence, 27(2):208–220, 2005.
[8] C. Harris and M. Stephens. A combined corner and edge detector. In Proc. Fourth Alvey Vision Conference, pages 147–151, 1988.
[9] F. Jurie and C. Schmid. Scale-invariant shape features for recognition of object categories. In IEEE Conference on Computer Vision and Pattern Recognition, pages II–90–96, 2004.
[10] P. D. Kovesi. Matlab functions for computer vision and image analysis. University of Western Australia. Available from: http://www.csse.uwa.edu.au/~pk/research/matlabfns/.
[11] D. Lowe. ”distinctive image features from scale-invariant keypoints”. International Journal of Computer Vision, 60(2):91–110, 2004.
[12] K. Mikolajczyk, A. Zisserman, and C. Schmid. Shape recognition with edge-based features. In British Machine Vision Converence, 2003.
[13] C. F. Olson and D. P. Huttenlocher. Automatic target recognition by matching oriented edge pixels. IEEE Transactions on Pattern Analysis and Machine Intelligence, 6(1):103–113, 1997.
[14] E. G. M. Petrakis, A. Diplaros, and E. Milios. Matching and retrieval of distorted and occluded shapes using dynamic programming. IEEE Transactions on Pattern Analysis and Machine Intelligence, 24(11):1501–1516, 2002.
[15] X. Ren, C. Fowlkes, and J. Malik. Mid-level cues improve boundary detection. Berkeley Technical Report 05-1382, CSD 2005.
[16] S. Sclaroff and L. Liu. Deformable shape detection and description via model-based region grouping. IEEE Transactions on Pattern Analysis and Machine Intelligence, 23(5):475–489, 2001.
[17] S. Wang, T. Kubota, J.M.Siskind, and J.Wang. Salient closed boundary extraction with ratio contour. IEEE Transactions on Pattern Analysis and Machine Intelligence, 27(4):546–561, 2005.

Feng Ge received his B.S. degree in Engineering Mechanics from Tsinghua University, Beijing, in July 2002, and received his M.E. degree in Computer Science and Engineering from University of South Carolina in May 2006. Now he is a PhD student with the Department of Electrical and Computer Engineering in Virginia Tech, Blacksburg, VA, 24060, USA.

Tiecheng Liu received his Ph.D. degree in computer science from Columbia University in 2003. He is an assistant professor in the department of Computer Science and Engineering at the University of South Carolina. His main research interests include computer vision, image and video processing, multimedia and advanced learning technologies. He has published over 30 refereed papers in the area of computer vision and multimedia technology and served as a committee member for IEEE CBAR'04, CIVR'05, and other conferences. He is a member of IEEE and ACM.

Song Wang received his Ph.D. degree in electrical and computer engineering from the University of Illinois at Urbana-Champaign in 2002. From 1998 to 2002, he also worked as a research assistant in the Image Formation and Processing Group of the Beckman Institute, University of Illinois at Urbana-Champaign. Since August 2002, he has been an assistant professor in the Department of Computer Science and Engineering at the University of South Carolina. His research interests include computer vision, medical image processing, and machine learning.

Joachim S. Stahl received BS degree in Computer Science from the Augusta State University in 2003, and the ME in Computer Science and Engineering from the University of South Carolina in 2005. He is currently a PhD student at the University of South Carolina at the department of Computer Science and Engineering. His research is on Computer Vision, in the areas of perceptual organization, segmentation and image understanding. He is a student member of IEEE.