Available online at www.sciencedirect.com
ScienceDirect
Pattern Recognition 40 (2007) 931–943
PATTERN RECOGNITION
THE JOURNAL OF THE PATTERN RECOGNITION SOCIETY
www.elsevier.com/locate/pr

Accurate object contour tracking based on boundary edge selection☆

Myung-Cheol Roh[a], Tae-Yong Kim[a], Jihun Park[b], Seong-Whan Lee[a,*]

[a]*Department of Computer Science and Engineering, Korea University, Anam-dong, Seongbuk-ku, Seoul 136-713, Korea*
[b]*Department of Computer Engineering, Hongik University, Sangsu-dong, Mapo-ku, Seoul, Korea*

Received 8 November 2005; received in revised form 24 May 2006; accepted 7 June 2006

Abstract

In this paper, a novel method for accurate subject tracking, by selecting only tracked subject boundary edges in a video stream with a changing background and moving camera, is proposed. This boundary edge selection is achieved in two steps: (1) removing background edges using edge motion, and from the output of the previous step, (2) selecting boundary edges using a normal direction derivative of the tracked contour. Accurate tracking is based on reduction of the effects of irrelevant edges, by only selecting boundary edge pixels. In order to remove background edges using edge motion, the tracked subject motion is computed and edge motions and edges having different motion directions from the subjects are removed. In selecting boundary edges using the normal contour direction, the image gradient values on every edge pixel are computed, and edge pixels with large gradient values are selected. Multi-level Canny edge maps are used to obtain proper details of a scene. Multi-level edge maps allow tracking, even though the tracked object boundary has complex edges, since the detail level of an edge map for the scene can be adjusted. A process of final routing is deployed in order to obtain a detailed contour. The computed contour is improved by checking against a strong Canny edge map and hiring strong Canny edge pixels around the computed contour using Dijkstra's minimum cost routing. The experimental results demonstrate that the proposed tracking approach is robust enough to handle a complex-textured scene in a mobile camera environment.

Keywords: Object contour tracking; Boundary edge selection; Optical flow; Contour normal direction; Multi-level edge map

1. Introduction

Tracking of moving objects is a popular issue because of the huge number of potential applications in computer vision, including video coding, video surveillance, augmented reality, and robotics. Although many approaches exist in tracking an object, it still remains difficult to track an object's contour under complex scene environments, where there are many edges around an object's contour. This paper addresses the problem of selecting boundary edges for robust contour tracking in video.

Nguyen et al. proposed a method for tracking a nonparameterized object contour, using the watershed algorithm [1–3]. In Nguyen et al.'s algorithm, a watershed line determined by watershed segmentation and watershed line smoothing energy become a new contour of an object. They removed background edges using object motion. However, many irrelevant edges that prohibit accurate contour tracking are left. To overcome this problem, this paper proposes a method of selecting only the edges in the boundary of the tracked object. In order to increase object contour tracking accuracy, background edges are removed using edge motions. The background edges whose motion directions are different from that of the tracked subject are removed. This is applied in a highly textured scene. After background edge removal, the average intensity gradient is computed in the normal direction of the previous frame contour, and only the

☆A preliminary version of this paper has been presented in the 17th International Conference on Pattern Recognition, Cambridge, UK, 23–26 August 2004.

*Corresponding author. Department of Computer Science and Engineering, Korea University, Anam-dong, Seongbuk-ku, Seoul 136-713, Korea. Tel.: +82 2 3290 3197; fax: +82 2 926 2168.

E-mail addresses: mcroh@image.korea.ac.kr (M.-C. Roh), tykim@image.korea.ac.kr (T.-Y. Kim), jhpark@cs.hongik.ac.kr (J. Park), swlee@image.korea.ac.kr (S.-W. Lee).

0031-3203/$30.00
doi:10.1016/j.patcog.2006.06.014

edges with high gradient values as the boundary edges of the tracked object are considered. Multi-level Canny edges are used to obtain proper detail of a scene. In addition, the contour of a watershed line is verified by checking against a strong Canny edge map around the contour, this is simple and does not include highly textured edges. The edge map can be generated using various control parameters of a Canny edge generator. Final routing is run using the computed contour pixels and strong Canny edge pixels. Thus, the proposed contour tracking method is robust even though an object boundary is not clear due to many textured backgrounds and objects.

The tracking results are compared with Nguyen et al.'s tracking results. In addition, the experimental results show that the proposed contour tracking approach is reliable enough to handle a sudden change of the tracked subject shape in a complex scene.

2. Related works

The methods of representing an object contour can be classified into two categories: parameterized contour and nonparameterized contour. In tracking a parameterized contour, the object contour is represented using parameters. Many of these approaches use snake models [4]. Kalman snake [5] and occlusion adaptive motion snake [6]. In the method of tracking a nonparameterized contour, a contour is represented by the border of the object. The contour generated by these algorithms is represented as a set of pixels. Paragios et al.'s algorithm [7,8] and Nguyen et al.'s algorithm [1] are popular in these approaches.

2.1. Object contour tracking using deformable templates

Zhong et al. proposed a method for object tracking using prototype-based deformable template model [9,10]. In order to track an object in an image sequence, a criterion combining a frame-to-frame deviation of the object shape and a fidelity of the modeled shape to the input image is used. In Zhong et al.'s algorithm, deformable template model is built from the shape information which is extracted from the previous frame along with a systematic shape deformation scheme to model the object shape in a new frame. In tracking stages, (1) edge and gradient information, (2) region consistency, and (3) interframe motion are used. In edge and gradient information, the object boundary consists of pixels whose gradient value is large. In region consistency, for the same object, the color and texture is consistent throughout the video sequence. Finally, in interframe motion, the boundary of a moving object should be characterized by large interframe motion. However, this approach faces difficulty in tracking under fast moving camera and complex scenes, because it is assumed that the boundary of the moving object can be characterized by large interframe motion and the edge gradient is sufficient to be discriminated easily.

2.2. Object contour tracking using geodesic active contour with level set formulation scheme

Paragios and Deriche proposed the geodesic active contour [7]. In this scheme, motion detection is performed using a statistical framework for which the observed interframe difference density function is approximated using a density function. This model is composed of two components, background and a moving object. In this situation, both components represent zero-mean and obey Laplacian and Gaussian law. This statistical framework is used to provide motion detection boundaries.

In this scheme, level set formulation is used. Complex curves can be detected and tracked while topographical changes for the evolving curves are naturally managed. In order to reduce the computational cost required by a direct implementation of the level set formulation scheme, a new approach called Hermes is proposed. Hermes exploits aspects from the well-known front propagation algorithms, narrow band and fast marching.

However, their proposed approach demonstrates good results only in static background situations because the difference image is not reliable in a moving background.

2.3. Object contour tracking using occlusion adaptive motion snake

Fu et al. proposed a method of tracking a contour using occlusion adaptive motion snake [6]. In this scheme, the active contour model is used [4]. In occlusion of the adaptive motion snake, a contour is segmented into several areas based on color, curvature and motion. The segmented areas are separated into object area and background area. The motions of the separated object area and background area are estimated. In this case, one of the two motions is selected. The motion caused by occlusion is not selected. However, the approach requires many parameters to be determined, and furthermore, has difficulty in segmenting the contour.

2.4. Nonparameterized object contour tracking using background edge removal

Nguyen et al. proposed a method for tracking the nonparameterized object contour in a single video stream [1]. In this algorithm, new tracked contour was determined using the watershed algorithm with a watershed line smoothing energy [1–3] added in the energy minimization function. The contour is represented by a border between the tracked object and background areas.

The approach combined outputs of two steps: creating a predicted contour, and removing background edges using object motion, $V_p(t)$. The previous edge map is translated using object motion, $V_p(t)$ and the translated edge map is compared with the current edge map after conducting

distance transform of the translated edge map. Nguyen et al. regarded edges found by looking at the discrepancies between the translated edges map and the current edge map as irrelevant edges.

However, Nguyen et al.'s approach left many irrelevant edges in the following cases: (1) an edge segment that has the same direction as $V_p(t)$ and length that exceeds the length of $V_p(t)$, (2) highly textured background and (3) inner edges of a tracked object. These irrelevant edges prohibit accurate contour tracking.

3. The proposed contour tracking method

3.1. The overview of the tracking method

In the proposed scheme, a nonparameterized object contour-based method in a single video stream is used. In the Nguyen et al.'s algorithm, the new tracked contour is determined by a watershed algorithm with watershed line smoothing energy added in the energy minimization function [1–3]. The contour is the border between a tracked object and background areas. Fig. 1 presents an overview of the proposed tracking method.

In the process of motion tracking, it is required to compute the object motion and edge motion. The object motion can be computed by taking the inside area of the previous contour and computing the offset to match the current image best. The edge motion corresponds to the magnitude and direction of an optical flow for every edge pixel of an image edge map.

To create a new contour, the watersnake model is used in the proposed approach [1,3]. Two edge indicator functions, $h^{(p)}(\mathbf{x})$ and $h^{(I)}(\mathbf{x})$, are defined to decide a new contour in the stage of new contour detection. $h^{(p)}(\mathbf{x})$ is an edge indicator function from the predicted contour, $\partial\Omega^{(p)}$, and $h^{(I)}(\mathbf{x})$ is an edge indicator function computed from the edge map resulting after background edges are removed by object motion vector. A boundary edge map, $\Phi^{(B)}(t)$, and $h^{(B)}(\mathbf{x})$ are derived from edge map, $\Phi^{(R)}(t)$, and $\Phi^{(B)}(t)$, respectively.

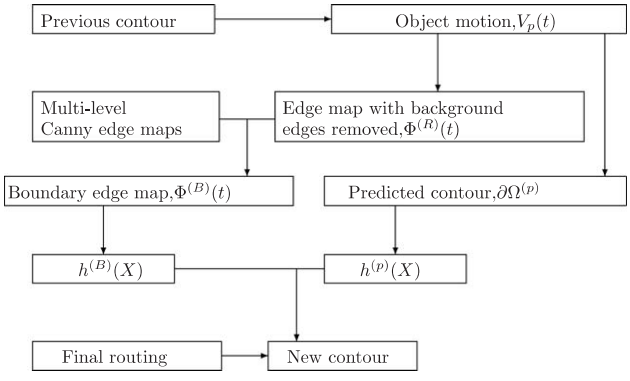

Fig. 1. Overview of our tracking method.

To track a new contour, both Nguyen et al.'s approach [1] and the proposed approach use Canny edges selected. Nguyen et al.'s approach creates an edge map by removing the background edge from a Canny edge map; background edges are removed using object motion, $V_p(t)$. The edge map is created through two steps. Firstly, background edges are removed by edge motion and secondly, boundary edge pixels are selected by the image intensity gradient in the normal direction of the predicted contour. The best part of the proposed algorithm is selecting boundary edge pixels from Canny edges. Although, it is not easy to remove background noisy edges in general, the intensity of the gradient-based method along the predicted contour allows selection of the most reliable pixels while ignoring noisy edges.

3.2. Boundary edge selection

The process of boundary edge selection consists of two steps. First, background edge removal is achieved using edge motions. Second, boundary edge pixel selection is achieved using the gradient in the normal direction of the contour. This section describes how to obtain the boundary edge map, $\Phi^{(B)}(t)$ from the edge map, $\Phi^{(R)}(t)$, by removing irrelevance.

3.2.1. Background edge removal using edge motion

The background edges are removed by comparing motions of object and background. The tracked subject motion and background edge motions are computed. The background edges whose motion directions differ from that of the tracked subject, are removed. Edge motion is computed using optical flow from edge maps generated by the Canny edge generator [11–14].

The tracked subject motion vector is computed to be $V_p(t)$, and each edge pixel's motion vector is tested against $V_p(t)$. If the difference between the two vectors is larger than a specified constant T_e, it is considered to be a background edge pixel. Let $\Phi^{(I)}(t)$ be the edge map detected at the current frame. Vector O_{Edge} is the computed optical flow of an edge pixel in $\Phi^{(I)}(t)$. The dominant translation vector $V_p(t)$ is estimated by

$$V_{(p)}(t) = \arg\min_{V\in\Psi} \sum_{p\in\Omega(t-1)} [I(p,t-1) - I(p+V,t)]^2, \quad (1)$$

where Ψ is the velocity space and $\Omega(t-1)$ is the pixels that belong to an object area in frame $(t-1)$.

$$\Phi_{background}(t) = \{\text{Edge} \in \Phi^{(I)}(t) \mid \|V_p(t) - O_{Edge}\| > T_e\}. \quad (2)$$

$\Phi^{(R)}(t)$ is an edge of $\Phi^{(I)}(t)$ subtracted by $\Phi_{background}(t)$, where $\Phi_{background}(t)$ is a background edge map. The background edge removal method using edge motion removes edges with different motions than that of the tracked

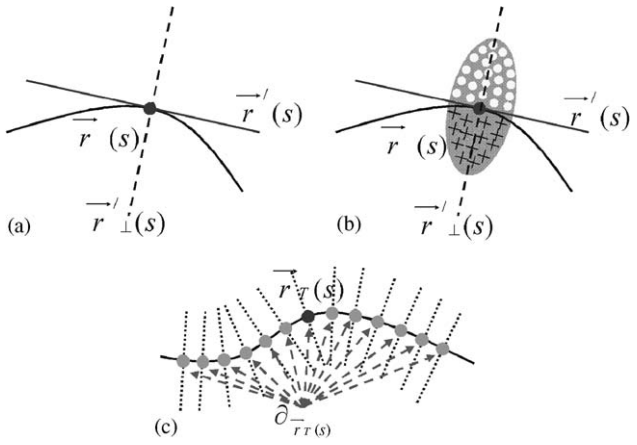

Fig. 2. Calculating the gradient of contour normal direction in a parametric contour. (a) A normal direction of the parametric contour $\vec{r}(s)$. (b) An ellipse with two inside areas separated by a contour for calculating $\widehat{D}$. (c) $\partial_{\vec{r}_T(s_i)}$ and the contour normal directions at the pixels that belong to $\partial_{\vec{r}_T(s_i)}$ for calculating $\widehat{TD}_i(\vec{r}_T(s_i))$.

subject, this method is independent of the degree of complexity in the edge map, while accurately removing all background edge pixels with different motions. The edge map without background edges is used in computing boundary edge selection.

3.2.2. Boundary edge pixel selection

This section explains the method of only selecting boundary edges, $\Phi^{(B)}(t)$, for improving the accuracy of object contour tracking.

- Calculating an image gradient in a contour normal direction.

The intensity gradient to the normal direction of the contour for each pixel is computed, in order to remove noisy and irrelevant edges of the object. Let parametrical representation of a contour be $\vec{r}(s) = \binom{x(s)}{y(s)}$. The tangent direction of $\vec{r}(s)$ is $\vec{r}^{\,/}(s)$ as presented in Fig. 2(a). The orthogonal direction of $\vec{r}^{\,/}(s)$ is $\vec{r}^{\,/}_{\perp}(s)$. The image gradients are considered only in the direction of $\vec{r}^{\,/}_{\perp}(s)$. $I(m,n)$ is an image intensity function.

$$D(\vec{r}(s)t) = \frac{1}{\sqrt{(\mathrm{d}x(s)/\mathrm{d}s)^2 + (\mathrm{d}y(s)/\mathrm{d}s)^2}} \times \left(\frac{\partial I}{\partial n} \frac{\mathrm{d}x(s)}{\mathrm{d}s} - \frac{\partial I}{\partial m} \frac{\mathrm{d}y(s)}{\mathrm{d}s} \right). \quad (3)$$

An average color gradient $\widehat{D}(\vec{r}(s_i))$, along the normal direction at a pixel point $\vec{r}(s_i)$ on the contour is computed by extending Eq. (3). $\vec{r}(s_i)$ is one of the pixel points of $\vec{r}(s)$. The computational process of $\widehat{D}(\vec{r}(s_i))$ is as follows: (i) Make an ellipse with two major axes of $\vec{r}^{\,/}_{\perp}(s)$ and $\vec{r}^{\,/}(s)$ directions. The size is adjusted properly. (ii) Separate the pixels inside the ellipse into two parts using a line along the $\vec{r}^{\,/}(s)$ direction as in Fig. 2(b). (iii) Calculate the mean intensity values of the pixels in two separate areas separated by $\vec{r}(s)$ in Fig. 2(b). The result of the computation is $\widehat{D}(\vec{r}(s_i))$. Fig. 2(c) presents the contour normal directions of the pixels belonging to $\partial_{\vec{r}_T(s_i)}$ for calculating $\widehat{TD}_i(\vec{r}_T(s_i))$.

- Advantages of using a normal direction of a contour.

The gradient values are not affected by intensity changes different from the normal direction of the contour because the gradient is computed only in the normal direction of the contour. This provides the ability to obtain high gradient values around the tracked object boundary, even in a complex scene. Fig. 3 demonstrates that unwanted edge values are weakened and boundary edges values are strengthened. The concept of considering a textured area divided by the boundary contour is applied. In using average intensity values of an image area, small changes and edges coming from noise and small texture patterns are blurred.

- The process of boundary edge pixel selection.

The boundary edge pixels are selected after removing the background edge by considering the object's edge motion. As explained in Section 3.1, $\Phi^{(R)}(t)$ is the edge map after removing background edges by edge motion. $\vec{r}(s)$ is the parametric representation of a predicted contour, $\partial\Omega^{(p)}$, and the total number of pixels on $\partial\Omega^{(p)}$ is N. $\vec{r}(s_i)$ is ith pixel of $\partial\Omega^{(p)}$. The boundary edge pixel selection process is done along $\partial\Omega^{(p)}$. The selection on every pixel point, $\vec{r}(s_i)$, is processed where $i = 1, \ldots, N$, on $\partial\Omega^{(p)}$. $\Phi_i^{(R)}(t)$ is considered, which is a part of the edge map, $\Phi^{(R)}(t)$, along $\partial\Omega^{(p)}$. $\Phi_i^{(R)}(t)$ has edges in a circular area centered at $\vec{r}(s_i)$ with radius c_Φ, a specified constant. $\vec{r}_T(s_i)$ is one of the edge pixels in $\Phi_i^{(R)}(t)$. $\vec{r}_T(s_i)$ can be considered to be one of a pixel-point of $\vec{r}_T(s) = \binom{x_T(s)}{y_T(s)}$, a parametric curve translated from $\vec{r}(s)$ by $(\vec{r}_T(s_i) - \vec{r}(s_i))$. $\partial\Omega_T^{(p)}$ is a contour translated from $\partial\Omega^{(p)}$ by $(\vec{r}_T(s_i) - \vec{r}(s_i))$. The left side of Fig. 4 presents $\partial\Omega^{(p)}$ and noisy edge pixels, $\Phi^{(R)}(t)$. The right side of Fig. 4 presents a close up of a circular area of radius c_Φ centered at $\vec{r}(s_i)$. This circular edge map is denoted as $\Phi_i^{(R)}(t)$. The gradient of a normal direction of $\partial\Omega_T^{(p)}$ at $\vec{r}_T(s_i)$ is computed at the pixel point of every edge on $\Phi_i^{(R)}(t)$.

To detect k possible pixels for boundary edges, a gradient of a normal direction of $\partial\Omega_T^{(p)}$ is computed at every edge pixel point of $\Phi_i^{(R)}(t)$, where k is a specified constant. $\partial_{\vec{r}_T(s_i)}$ is a set of pixels of $\vec{r}_T(s_i)$ on $\partial\Omega_T^{(p)}$ in the circular area of radius c_∂ centered at $\vec{r}_T(s_i)$. $\widehat{TD}_i(\vec{r}_T(s_i))$ is the sum of $\widehat{D}$s computed along the pixels of $\partial_{\vec{r}_T(s_i)}$ with reference at $\vec{r}_T(s_i)$. Fig. 2 (c) presents $\partial_{\vec{r}_T(s_i)}$ and the contour normal

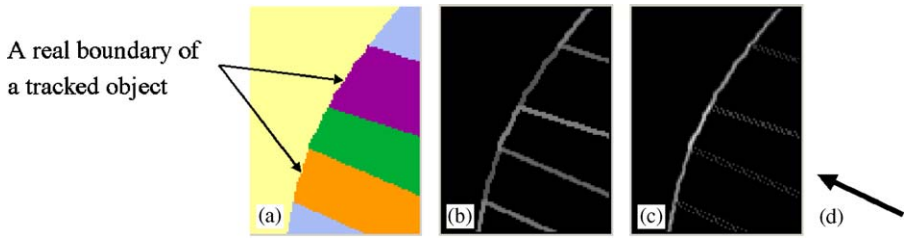

Fig. 3. Advantages of using a normal direction of a contour. (a) An example of image intensity change around a tracked subject. (b) An image gradient computed without considering any special direction. (c) An image gradient computed in the normal direction to the actual boundary of a tracked object. (d) The direction considered as the normal direction.

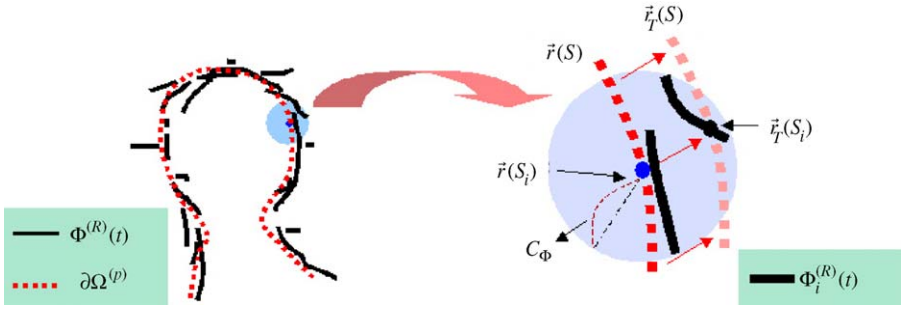

Fig. 4. A predicted contour and image operations along the contour. The operation is done on every edge pixel in a circular area.

directions at the pixels that belong to $\partial_{\vec{r}_T(s_i)}$ for calculating $\widehat{TD}_i(\vec{r}_T(s_i))$.

The biggest $\widehat{TD}_i$'s with large values in selecting boundary edges, is used. Fig. 5 presents the process of selecting pixels for boundary edges.

$$\widehat{TD}_i(\vec{r}_T(s_i)) = \sum_{\partial_{\vec{r}_T(s_i)}} (\widehat{D}_i(\vec{r}_T(s_i))). \tag{4}$$

In order to obtain a clear edge from complex edge image around the boundary of the object, a variety of Canny edge detection depending on a variety of Gaussian parameters is used. For Canny edge maps generated with lower image intensity gradient values, the Canny edge map and $\Phi_i^{(R)}(t)$ are computed at each level. Multi-level Canny edges are results of Canny edge detection depending on the given thresholds. Fig. 6 presents the results of Canny edge detections in three different levels, given a single image. The level of detail of a scene is controlled using multi-level Canny edge maps. The detailed Canny edge map of a scene confuses tracking, and a very simple edge map loses tracking information. Let $Num(\Phi_i^{(R)}(t)_{level(l)})$ be the number of edge pixels in $\Phi_i^{(R)}(t)_{level(l)}$, and $level_{num}$ be the number of level of Canny edges computed. $\Phi_i^{(R)}(t)_{level(l)}$ is $\Phi_i^{(R)}(t)$ calculated at level l. $\Phi_i^{(R)}(t)_{level(l)}$ has edges in a circular area centered at $\vec{r}(s_i)$ with radius c_Φ in level l. At the ith computation loop, if the number of $Num(\Phi_i^{(R)}(t))_{level(l)}$ is smaller than $Canny_{num}$, $\Phi_i^{(R)}(t)_{level(l)}$ is used in one step lower level, where $Canny_{num}$ is a constant. In other words, the detailed Canny edge result is used if the object boundary is not clear. Therefore, robust tracking results can be obtained although the object boundary is not clear. At the ith computation loop, k edge pixels with large $\widehat{TD}_i$ values are selected.

3.3. Contour tracking with selected boundary edges

In the steps of contour detection, using the concept of topographical distance, the watershed segmentation is done by minimizing [1,3]. For this algorithm, two edge indicator functions, $h^{(B)}(\mathbf{x})$ and $h^{(p)}(\mathbf{x})$ are used. Two edge indicator functions, $h^{(B)}(\mathbf{x})$ and $h^{(p)}(\mathbf{x})$, are derived from $\Theta_i^{(B)}(t)$ and $\partial\Omega^{(p)}$, respectively. The detailed algorithm for the edge indicator function is presented in Nguyen et al.'s paper [1]. The boundary edge map, $\Theta_i^{(B)}(t)$, is obtained using the algorithm proposed in Ref. [1]. $\partial\Omega^{(p)}$ is obtained by translating $\partial\Omega(t-1)$ by $V_p(t)$. A watershed line, extracted using two edge indicator functions, $h^{(B)}(\mathbf{x})$ and $h^{(p)}(\mathbf{x})$, becomes a new contour in the current frame.

3.4. Final routing with strong Canny edges

In order to obtain a detailed contour, final routing is conducted. A *strong* Canny edge map has a relatively small

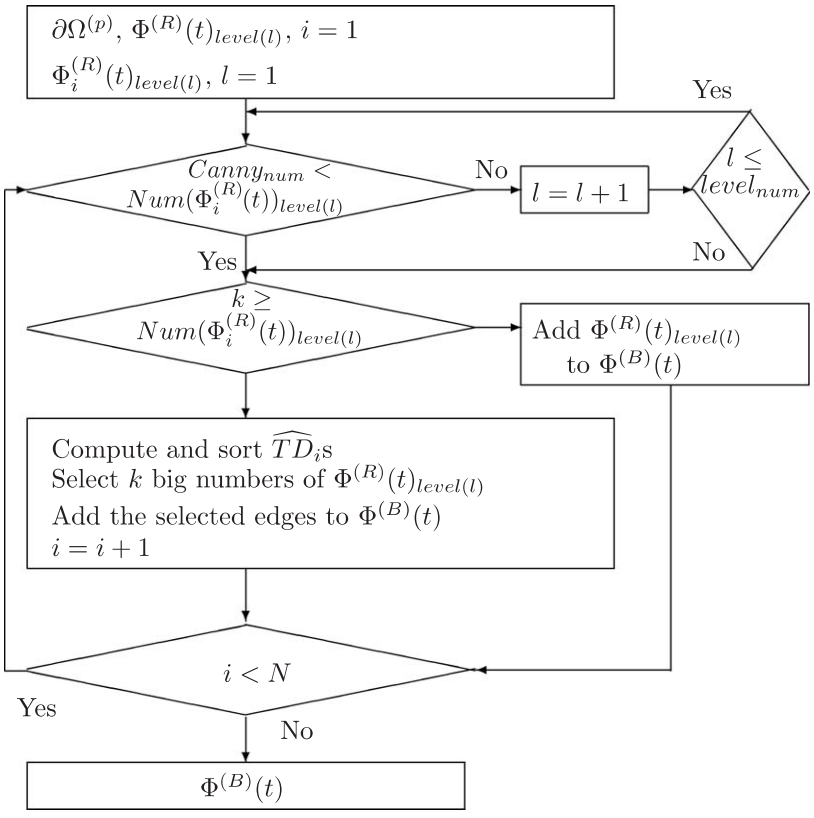

Fig. 5. The process of boundary edge selection.

Fig. 6. Results of Canny edge detections at three different levels given a single image.

number of pixels in the edge map. The edge map is made using only the large gradient value of the intensity changes in a scene. Final routing on strong Canny edges as well as the computed contour of watershed line is achieved. Let I be a pair of (Γ, Υ) consisting of a finite set of pixels, Γ, and a mapping, Υ, that assigns a value $\Upsilon(t)$ to each pixel t in Γ, where $\Upsilon(t)$ is assigned 1 or 2. 1 represents strong Canny edge pixels, and 2 represents computed contour pixels. An adjacency relation A is an irreflexive binary relationship between pixels of Γ. I can be interpreted as a directed graph whose nodes are the pixels and whose arcs are the pixel pairs in A. *sAt* depends only on the four-connected neighbor of the pixels in I, and $(s, t) \in \Gamma \times \Upsilon$. A path is a sequence of pixels $\pi = \langle t_1, t_2, \ldots, t_k \rangle$, where $(t_i, t_{i+1}) \in A$ for $1 \leqslant i \leqslant k-1$. t_1 is the origin, and t_k is the destination of the path. It is assumed that $f(\pi)$ represents a totally ordered set of cost values, where f is a given function which assigns a path cost to each path π. The set of cost values contains a maximum element denoted by $+\infty$. The additive cost function satisfies

$$f_{sum}(\pi \cdot \langle s, t \rangle) = f_{sum}(\pi) + w(s, t),$$

Fig. 7. Final contour fixed using strong Canny edges: (a) computation result of a watershed line; (b) fixed contour using strong Canny edges.

where $(s,t) \in A$, π is any path ending at s, and $w(s,t)$ is a fixed nonnegative weight assigned to the arc (s,t).

$$w(s,t) = \begin{cases} +\infty & \text{if } s \text{ and } t \text{ are not adjacent pixels,} \\ \Upsilon(s) \times \Upsilon(t) \times \Upsilon(t) & \text{if } s \text{ and } t \text{ are adjacent pixels and } \Upsilon(s) \neq \Upsilon(t), \\ 1 & \text{if } (s) \text{ and } (t) \text{ are adjacent pixels and } \Upsilon(s) = \Upsilon(t). \end{cases}$$

This weight function guarantees to take stronger Canny edges in the optimum path routing. The routing is done using Dijkstra's minimum cost routing algorithm. The result of the routing becomes a final contour of a tracked image. Fig. 7(a) is the computed contour of a watershed line created in Section 4, and this is used as an input to the final routing. The resulting final contour is presented in Figs. 7(b) and 8.

Fig. 9 presents an example of final routing. Pixel A is the beginning of the computed contour and pixel Q is the end of the computed contour. Numbers denote LOD values of either strong Canny edges or the computed contour.

Q.2	*P*.2	*O*.	*N*.
M.1	*L*.2	*K*.2	*J*.2
H.2	*G*.1	*F*.	*E*.2
D.	*C*.1	*B*.1	*A*.2

Fig. 9. A LOD Canny edge map and LOD values: an alphabet denotes a Canny edge pixel and its corresponding number denotes LOD value.

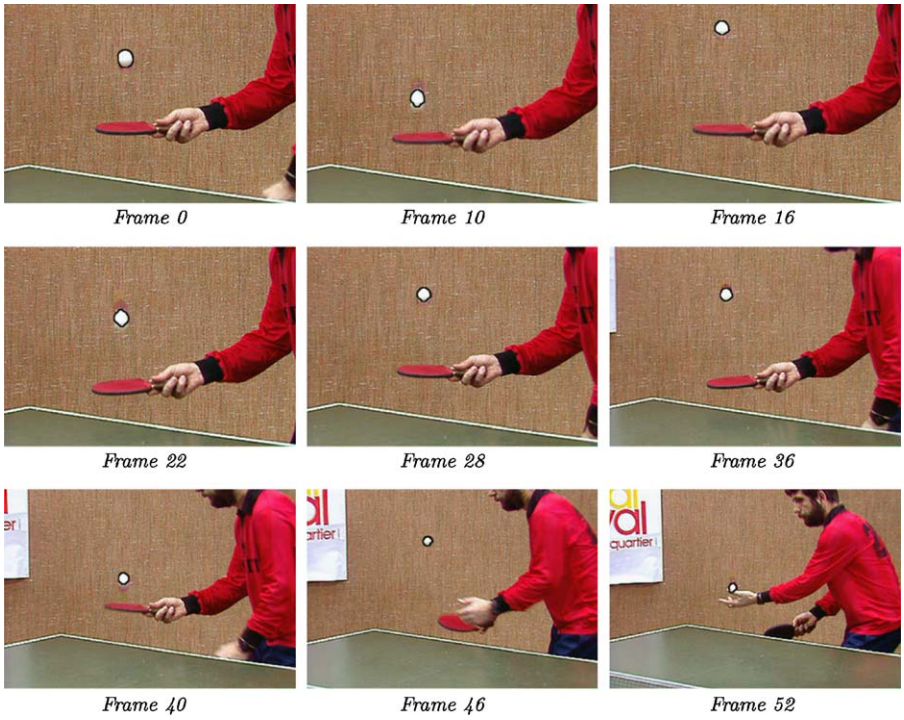

Fig. 8. Tracking results of a ball sequence by the proposed algorithm.

Pixels without any numbers denote pixels which are not the Canny edge. The strongest Canny pixel has level number 1, while the computed contour has level number 2. The computed contour is always four-neighbor connected while there is no guarantee of the strong Canny edge connection. The routing cost between adjacent pixels are as follows: routing cost from pixel A to B is $2 \times 1 \times 1 = 2$, while routing cost from pixels B to C is $1 \times 1 \times 1 = 1$, because the proposed routing favors strong Canny edges. The routing cost from pixels C to D is huge. There is no direct routing cost assigned between nonadjacent pixels. Therefore, the final route determined will be A → B → C → G → L → M → Q.

4. Experimental results and analysis

4.1. Experimental environments

For experiments, some test video sequences are collected from web sites and some are taken under complex scene environments using a hand-held camera. Microsoft Visual C++ 6.0 and the Intel Open Source Computer Vision Library is used. The computational time takes from a few seconds to few minutes, depending on the size and complexity of the contour boundary.

4.2. Results of boundary edge selection

Fig. 10 shows the background edges removed by Nguyen et al.'s approach, and boundary edges selected by setting the parameters of the approach as $Canny_{num} = 10$, $level_{num} = 3$, $c_{\partial} = 20$ and $c_{\Phi} = 15$. Figs. 8, 12–15 present contour tracking results for several video clips by the proposed method. The boundary edges with $k = 2$, is selected. Compared with the Nguyen et al.'s approach (Fig. 11), the proposed approach removes many more irrelevant edges that disturb accurate contour tracking.

4.3. Tracking results with boundary edge selection

Fig. 8 presents ball tracking from a pingpong sequence, using the proposed algorithm. The result demonstrates that accurate tracking is achieved while an object moves rapidly and the size of the object changes. Fig. 12 presents the tracking results with boundary edge selection in a subway foyer. The hall tiles generate many complicated edges as does the man's cross striped shirt. People moving in

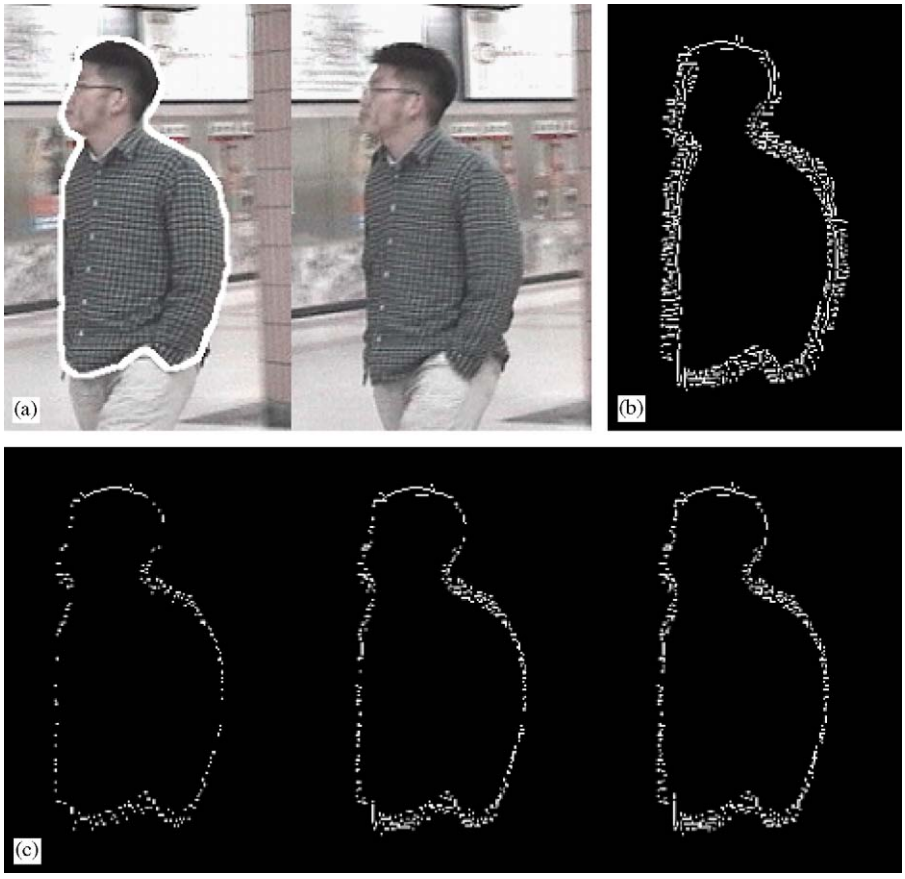

Fig. 10. Results of boundary edge selection. (a) Two consecutive frames and a contour determined at the previous frame. (b) An output of background edge removal by Nguyen et al.'s approach. (c) Outputs of boundary edge selections with $k = 1, 3, 5$.

Fig. 11. Tracking results of human body by Nguyen et al.'s.

Fig. 12. Tracking results of human body by the proposed method.

Fig. 13. Face tracking results by the proposed method: (a) tracking result for a sequence which has complicated background; (b) tracking result for a sequence of several moving objects.

different directions in the background make contour tracking complicated. The contour shape changes as the man wearing a cross striped shirt rotates turning back and the size of the contour changes with the man's location to the camera. It becomes larger when he comes closer to the camera and becomes smaller when he moves away from the camera. Fig. 12 shows that tracking by the proposed method is achieved successfully while the result of Nguyen et al.'s approach fails for the same frames (Fig. 11).

Figs. 13–15 show experimental results for various video clips using the proposed method. In Fig. 13(b), the target woman is walking with other people who are walking in the same direction, and it makes the contour tracking hard because of different speeds of the moving people and the different textures. However, the proposed method achieved good results because many edges around the object were eliminated easily. In Fig. 14(a), accurate contour tracking is achieved although the edges of the bag are complex due to the striped shirt. A fast ball which has two colors is tracked well in Fig. 14(a). Fig. 15(b) shows an example where the proposed contour tracking is not as good as other experimental results shown above. Since the proposed method does not follow the drastic changes of the contour's curvature, boundary selection is not achieved well, while tracking is successful. Some experimental results can be found at http://image.korea.ac.kr/tracking/demo.html.

5. Conclusions and further research

In this paper, a novel contour tracking method for improving the accuracy of a textured object in a video stream with a nonstatic camera is proposed. In order to overcome the noisy edge problem due to a complex scene, the proposed method selects valid edges around a tracked object boundary. The proposed boundary edge selection scheme consists of two steps. First, background edges are removed using the differences between the object's and background edge's motion. The edges that have different motions are regarded as redundant and background edges and are removed. Second,

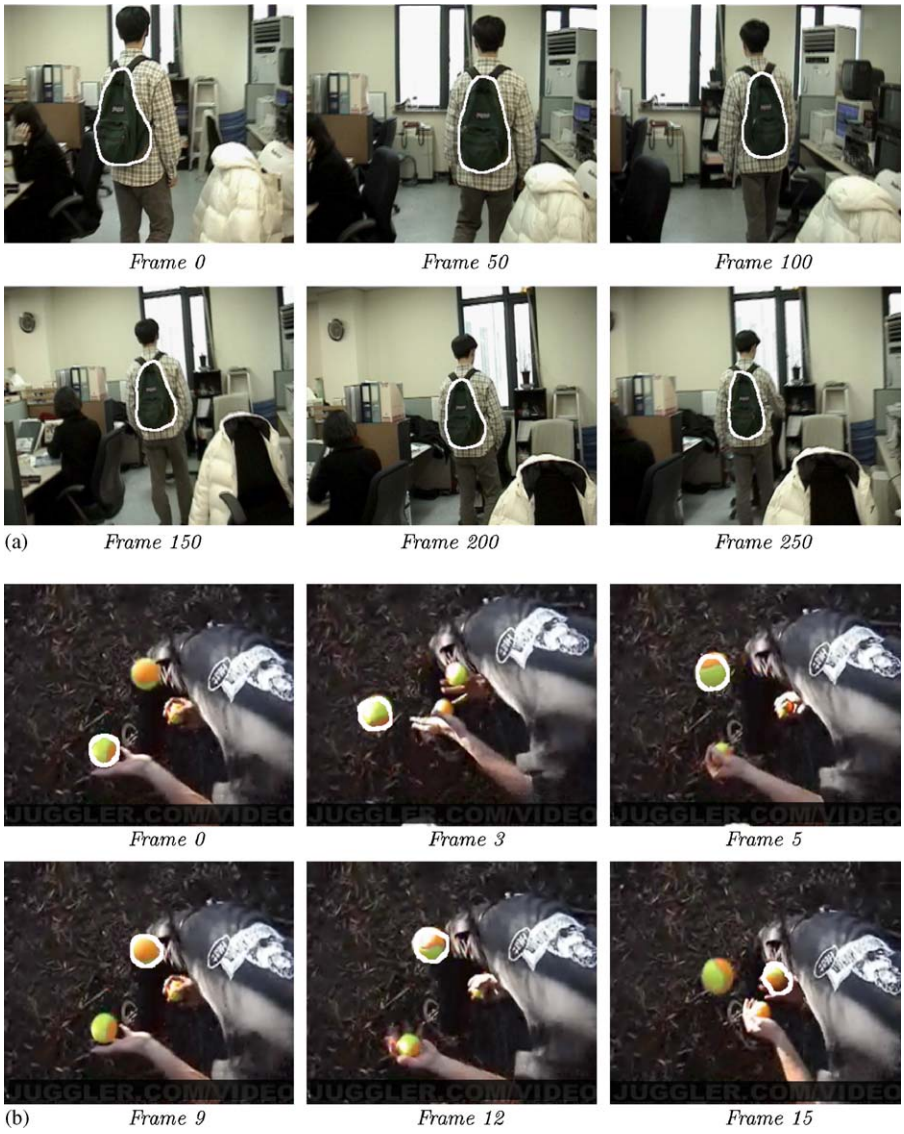

Fig. 14. Object tracking result by the proposed method: (a) tracking results of a nonrigid object with highly textured background; (b) tracking results of a fast moving object.

boundary edges are selected by calculating the normal direction derivative of the tracked contour from the output of the first step. The gradient of image intensity to the normal direction for the contour and the edges which are selected as boundary edges are computed, and the edges with different normal vectors from those of the object boundary's are eliminated considering only the normal direction of the contour.

In the proposed method, the contour's edges are selected accurately without being affected by noisy edges or small cross striped textures. In order to represent the updated contour, the watersnake model is used and detailed edges are extracted using multi-level Canny edges with various Gaussian parameters. Even though object boundaries are complex, the edges are extracted well enough.

The experimental results demonstrate that boundary edge selection by the proposed method is sufficient to handle changes of the object's shape in complex scene and tracking based on boundary edge selection is done successfully. Without interference from the texture of object and background, the method can select the appropriate edges and contour.

Since the proposed method cannot preserve the contour which has high curvature as we presented in Fig. 15(b), we are going to improve the tracking and representation of the unsmoothed object in future work. The proposed method can be also improved by carefully considering the edge motion by statical tracking method e.g. the Kalman filter. The filter can yield a better estimation of the motion vector. Moreover, there is no reliable frame of reference in measuring tracking quality. This paper did not conduct measurements of current work. Therefore, we are going to research on standard measurement and the results will be compared.

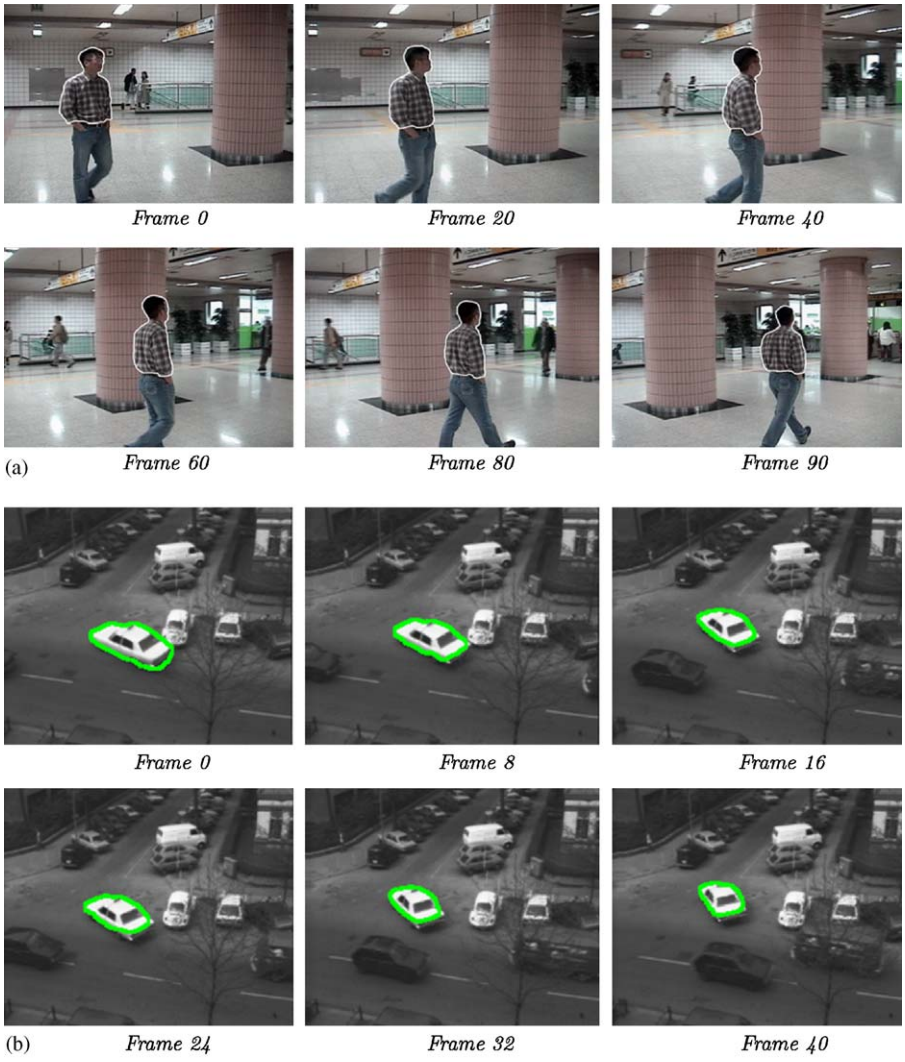

Fig. 15. Tracking results by the proposed method: (a) tracking results of a highly textured object; (b) tracking results of a rigid object.

6. Summary

In this paper, we propose a space-variant image representation model based on properties of magnocellular visual pathway, which performs motion analysis, in human retina. Then, we present an algorithm for the tracking of multiple objects in proposed space-variant model. The proposed space-variant model has two effective image representations for object recognition and motion analysis, respectively. Each image representation is based on properties of two types of ganglion cell, which are the beginning of two basic visual pathway; one is parvocellular and the other is magnocellular. Through this model, we can get efficient data reduction capability with no great loss of important information. And, the proposed multiple objects tracking method is restricted to the space-variant image. Typically, an object-tracking algorithm consists of several processes such as detection, prediction, matching, and updating. In particular, the matching process plays an important role in multiple objects tracking. In traditional vision, the matching process is simple when the target objects are rigid. In space-variant vision, however, it is very complicated although the target is rigid, because there may be deformation of an object region in the space-variant coordinate system when the target moves to another position. Therefore, we propose a deformation formula in order to solve the matching problem in space-variant vision. By solving this problem, we can efficiently implement multiple objects tracking in space-variant vision.

Acknowledgment

This research was supported by the Intelligent Robotics Development Program, one of the 21st Century Frontier R&D Programs funded by the Ministry of Commerce, Industry and Energy of Korea.

References

[1] H.T. Nguyen, M. Worring, R. van den Boomgaard, A.W.M. Smeulders, Tracking nonparameterized object contours in video, IEEE Trans. Image Process. 11 (9) (2002) 1081–1091.

[2] J.B.T.M. Roerdink, A. Meijster, The watershed transform: definition, algorithms and parallelization strategies, Fundam. Inf. 41 (1–2) (2000) 187–228.

[3] H.T. Nguyen, M. Worring, R. van den Boomgaard, Watersnakes: energy-driven watershed segmentation, IEEE Trans. Pattern Anal. Mach. Intell. 25 (3) (2003) 330–342.

[4] M. Kass, A. Witkin, D. Terzopoulos, Snakes: active contour models, Int. J. Computer Vision 1 (4) (1987) 321–331.

[5] N. Peterfreund, Robust tracking of position and velocity with Kalman snakes, IEEE Trans. Pattern Anal. Mach. Intell. 21 (6) (1999) 564–569.

[6] Y. Fu, A.T. Erdem, A.M. Tekalp, Tracking visible boundary of objects using occlusion adaptive motion snake, IEEE Trans. Image Process. 9 (12) (2000) 2051–2060.

[7] N. Paragios, R. Deriche, Geodesic active contours and level sets for the detection and tracking of moving objects, IEEE Trans. Pattern Anal. Mach. Intell. 22 (3) (2000) 266–280.

[8] N. Paragios, O. Mellina-Gottardo, V. Ramesh, Gradient vector flow fast geometric active contours, IEEE Trans. Pattern Anal. Mach. Intell. 26 (3) (2004) 402–407.

[9] Y. Zhong, A.K. Jain, M.-P. Dubuisson-Jolly, Object tracking using deformable templates, IEEE Trans. Pattern Anal. Mach. Intell. 22 (5) (2000) 544–549.

[10] A.K. Jain, Y. Zhong, S. Lakshmanan, Object matching using deformable templates, IEEE Trans. Pattern Anal. Mach. Intell. 18 (3) (1996) 267–278.

[11] D. Gibson, M. Spann, Robust optical flow estimation based on a sparse motion trajectory set, IEEE Trans. Image Process. 12 (4) (2003) 431–445.

[12] A.G. Bors, I. Pitas, Optical flow estimation and moving object segmentation based on median radial basis function network, IEEE Trans. Image Process. 7 (5) (1998) 693–702.

[13] J. Shi, C. Tomasi, Good features to track, IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 1994, pp. 593–600.

[14] J. Shi, C. Tomasi, Performance of optical flow techniques, Int. J. Comput. Vision 12 (1994) 42–77.

About the Author—MYUNG-CHEOL ROH received his BS degree in Computer Engineering from Kang-Won National University, Korea, in 2001, and his MS degrees in Computer Science and Engineering from Korea University, Seoul, Korea, in 2003. He is currently a Ph.D. student at the Department of Computer Science and Engineering in Korea University. He won the best paper award of the 25th annual paper competition by the Korea Information Science Society and Microsoft in 2006. He worked with Prof. Joseph Kittler at the Center for Vision, Speech and Signal Processing in the University of Surrey as a visiting researcher by a support from the Korea Research Foundation for one year in 2004. His present research interests include object tracking, face and gesture recognition and robot vision.

About the Author—TAE-YONG KIM received his BS degree in Computer Science from Sunchon University, Jeonnam, Korea, in 2002, and his MS degrees in Computer Science and Engineering from Korea University, Seoul, Korea, in 2004. His present research interests include object tracking, image segmentation and robot vision.

About the Author—JIHUN PARK received his BEng degree in Mechanical Design and Production Engineering from Seoul National University, Seoul, Korea, in 1983, his MS in Computer Science from KAIST, Seoul, Korea, in 1985, and MSCS and Ph.D. degrees in Computer Science from University of Texas at Austin in 1990 and 1994, respectively. From March 1988 to August 1993, he was an assistant professor in the Department of Computer Engineering at Pusan University of Foreign Studies, Pusan, Korea. In August 1994, he joined the faculty of the Department of Computer Engineering at Hongik University, Seoul, Korea, as an assistant professor, and he is now an associate professor. His research interests include computer vision and computer graphic animation.

About the Author—SEONG-WHAN LEE received his BS degree in Computer Science and Statistics from Seoul National University, Seoul, Korea, in 1984, and his MS and Ph.D. degrees in computer science from KAIST in 1986 and 1989, respectively. From February 1989 to February 1995, he was an assistant professor in the Department of Computer Science at Chungbuk National University, Cheongju, Korea. In March 1995, he joined the faculty of the Department of Computer Science and Engineering at Korea University, Seoul, Korea, as an associate professor, and he is now a full professor. He was the winner of the Annual Best Paper Award of the Korea Information Science Society in 1986. He obtained the First Outstanding Young Researcher Award at the Second International Conference on Document Analysis and Recognition in 1993, and the First Distinguished Research Professor Award from Chungbuk National University in 1994. He also obtained the Outstanding Research Award from the Korea Information Science Society in 1996. He has been the founding co-Editor-in-chief of the International Journal on Document Analysis and Recognition and the associate editor of the Pattern Recognition Journal, the International Journal of Pattern Recognition and Artificial Intelligence, and the International Journal of Computer Processing of Oriental Languages since 1997. He was the Program co-chair of the Sixth International Workshop on Frontiers in Handwriting Recognition, the Second International Conference on Multimodal Interface, the 17th International Conference on the Computer Processing of Oriental Languages, the Fifth International Conference on Document Analysis and Recognition, and the Seventh International Conference on Neural Information Processing. He was the workshop co-chair of the Third International Workshop on Document Analysis Systems and the First IEEE International Workshop on Biologically Motivated Computer Vision. He served on the program committees of several well-known international conferences. He is a fellow of IAPR, a senior member of the IEEE Computer Society and a life member of the Korea Information Science Society. His research interests include pattern recognition, computer vision and neural networks. He has published more than 200 publications in these areas in international journals and conference proceedings, and has authored five books.