A Contour-Based Moving Object Detection and Tracking

Masayuki Yokoyama
Advanced LSI Design & Development Division
SoC Business Center
Sony Corporation
Tokyo, Japan

Tomaso Poggio
Center for Biological and Computational Learning
Computer Science and Artificial Intelligence Lab
Brain Sciences Department
Massachusetts Institute of Technology
Cambridge, MA 02139

***Abstract*— We propose a fast and robust approach to the detection and tracking of moving objects. Our method is based on using lines computed by a gradient-based optical flow and an edge detector. While it is known among researchers that gradient-based optical flow and edges are well matched for accurate computation of velocity, not much attention is paid to creating systems for detecting and tracking objects using this feature. In our method, extracted edges by using optical flow and the edge detector are restored as lines, and background lines of the previous frame are subtracted. Contours of objects are obtained by using snakes to clustered lines. Detected objects are tracked, and each tracked object has a state for handling occlusion and interference. The experimental results on outdoor-scenes show fast and robust performance of our method. The computation time of our method is 0.089 s/frame on a 900 MHz processor.**

I. Introduction

Detecting and tracking moving objects are widely used as low-level tasks of computer vision applications, such as video surveillance, robotics, authentication systems, user interfaces by gestures, and a pre-stage of MPEG4 image compression. Software development of low-level tasks is especially important because it influences the performance of all higher levels of various applications.

Motion detection is a well-studied problem in computer vision. There are two types of approaches: the *region-based approach* and the *boundary-based approach*.

In the case of motion detection without using any models, the most popular region-based approaches are background subtraction and optical flow. Background subtraction detects moving objects by subtracting estimated background models from images. This method is sensitive to illumination changes and small movement in the background, e.g. leaves of trees. Many techniques have been proposed to overcome this problem [1], [2], [3], [4]. The mixture of Gaussians [2] is a popular and promising technique to estimate illumination changes and small movement in the background. However, a common problem of background subtraction is that it requires a long time for estimating the background models. It usually takes several seconds for background model estimation because the speed of illumination changes and small movement in the background are very slow. Optical flow also has a problem caused by illumination changes since its approximate constraint equation basically ignores temporal illumination changes [5].

In the case of boundary-based approaches, many of them use edge-based optical flow [6], [7], [8], level sets [9], and active contours, such as snakes [10], balloons [11], and geodesic active contours [12]. In [6], a method for detecting moving edges is proposed. The zero-crossings at each pixel are calculated from the convolution of the intensity history, with the second-order temporal derivative of the Gaussian function for detecting moving edges. The authors show theoretically and experimentally that this method is insensitive to illumination changes. However, the results of this method are inaccurate when the image is not smoothed sufficiently because it computes velocity without using spatial information of neighborhood pixels. In [13], a color segmentation and a non-parametric approach for computing a dense disparity map are used for detecting initial contours of moving objects for snakes. The parameters for color segmentation have to be tuned to the conditions of illumination. Geodesic active contours and level sets are used in [14], but these techniques are computationally too expensive for real-time applications.

In this paper, we present a method for detecting and tracking moving objects which includes non-rigid objects. Our method is robust because we use edge-based features which are insensitive to illumination changes. The method is also fast because the area of edge-based features is less than region-based features. This paper is organized as follows: In Section 2, we give details of detecting moving edges by using a gradient-based optical flow technique and an edge detector. Methods of contour extraction from detected and restored line segments are described in Section 3. Methods of tracking detected objects are presented in Section 4. Experimental results and conclusions are described in Sections 5 and 6, respectively.

II. Optical Flow

For the computation of optical flow of 2D image motion, the following approximation is used:

$$I(\mathbf{x}, t) \approx I(\mathbf{x} + d\mathbf{x}, t + dt), \tag{1}$$

where $I(\mathbf{x}, t)$ is the spatiotemporal function of image intensity. Intensity of time t and $t + dt$ is assumed to remain the same, i.e., no illumination changes are supposed to occur. Equation

(1) can be expanded using Taylor series and ignoring higher order terms as follows:

$$\nabla^T I \cdot \mathbf{v} + I_t = 0, \tag{2}$$

where $\nabla I = (I_x, I_y)^T$ is the gradient, I_t is the temporal derivative of $I(\mathbf{x}, t)$, and $\mathbf{v} = (u, v)^T$ is the image velocity, respectively. However, one linear constraint equation is insufficient to determine two-dimensional velocity. This problem is known as the *aperture problem*.

Many methods have been proposed to add other constraints. Barron *et al.* [5] evaluated several of them, and they show quantitatively that the method proposed by Lucas and Kanade [15] is one of the most reliable. This method is based on minimizing a weighted sum-of-squares error function of a spatial local region. All pixels in a region are supposed to have the same velocity, and the equation of the error is written as:

$$E = \sum_{i \in R} |w_i| [\nabla^T I(\mathbf{x}_i, t) \cdot \mathbf{v} + I_t(\mathbf{x}_i, t)]^2, \tag{3}$$

where R is a spatial local region and $|w_i|$ is a positive weight of each pixel of the region. To compute the least error of (3), the first-order derivative with respect to $\mathbf{v}$ is computed. Then the estimated velocity is obtained as:

$$\hat{v} = -\mathbf{M}^{-1}\mathbf{b}. \tag{4}$$

where

$$\begin{aligned} \mathbf{M} &= \begin{bmatrix} \sum |w_i| I_x^2 & \sum |w_i| I_x I_y \\ \sum |w_i| I_x I_y & \sum |w_i| I_y^2 \end{bmatrix} \\ \mathbf{b} &= \begin{bmatrix} \sum |w_i| I_x I_t \\ \sum |w_i| I_y I_t \end{bmatrix}, \end{aligned} \tag{5}$$

assuming that $\mathbf{M}$ is invertible. Horn and Schunk [16] applied pre-smoothing to a global region in order to simplify the optical flow problem.

Even though the method proposed in [15] has more reliability than other comparable techniques, the approximation in (1) is problematic when the illumination changes. Simoncelli *et al.* [17] showed that uncertainty of the calculation of (2) decreases nonlinearly in proportion to the trace of M of (5), i.e., the magnitudes of the spatial gradients.

Our method is based on using features which have strong magnitudes of the gradients. These features are extracted by a Canny edge detector [18]. Edges are masked by the thresholding velocity magnitude τ for eliminating background edges with little motion, i.e.,

$$|u| + |v| \leq \tau. \tag{6}$$

The Manhattan magnitude is used for simplifying the computation. Unmasked edges are used for restoring lines and extracting contours of objects in later phases.

III. Contour Extraction

Our method for extracting contours of moving objects consists of 4 steps explained in the following sections: *line restoration*, *line-based background subtraction*, *clustering*, and *active contours*.

A. Line Restoration and Line-based Background Subtraction

Edges are sometimes incorrectly masked by (6) because of the use of local regions for computing optical flow. This locality causes two types of miscalculations:

1) No additional information for solving (2) exists in the region (i.e., the aperture problem).
2) Moving edges are incorrectly considered as stationary edges if stationary edges exist in the same region with moving edges.

These miscalculations make detected edges fragmented (shown in Fig. 1(c)). For restoring these fragments, masked edges by (6) are restored as unmasked edges (i.e. moving edges), if they are connected to points of moving edges without including any cross-points. This restoration can be easily calculated because moving edges are obtained as one pixel wide lines by the edge detector [18]. As a result of this restoration, many lines of moving objects are restored, and their shapes become more clear (shown in Fig. 1(d)).

However, restored lines also include background edges which are incorrectly detected as the motion for the following reasons:

1) Blinked reflection causes miscalculation of the motion. It appears on corners of buildings and waves, for instance, and is detected as moving edges.
2) Background edges around moving objects are detected as moving edges because the region-based optical flow calculates velocity as the same in a region.

The noise as described above can be eliminated by subtracting background edges of the previous frame (except for the reflection of non-rigid objects). The detected and restored line of the current frame is eliminated if the line belongs to a background line (shown in Fig. 1(e)). We define a detected and restored line of the current frame as $\{\mathbf{p}_1, \cdots, \mathbf{p}_n\} \in S_{mv}$, and all background lines of the previous frame as S_{bg}. Then the line of the current frame is considered as a background line when it satisfies the following equation:

$$\frac{\sum_{k=1}^{n} b_k}{n} > \mu, \tag{7}$$

where $\mu (0 \leq \mu \leq 1)$ is a constant, and

$$b_i = \begin{cases} 1 & \text{if } \mathbf{p}_i \in S_{bg} \\ 0 & \text{otherwise.} \end{cases} \tag{8}$$

B. Clustering

We use a nearest-neighbor clustering with respect to the distance and velocity. We label two lines $\{\mathbf{p}_1, \cdots, \mathbf{p}_m\} \in S_1$ and $\{\mathbf{q}_1, \cdots, \mathbf{q}_n\} \in S_2$ as the same, if they satisfy the following constraints:

$$\begin{aligned} \min_{i \in \{1,\cdots,m\}, j \in \{1,\cdots,n\}} \{|x_{p_i} - x_{q_j}| + |y_{p_i} - y_{q_j}|\} &\leq \alpha_d \\ \min_{i \in \{1,\cdots,m\}, j \in \{1,\cdots,n\}} \{|u_{p_i} - u_{q_j}|\} &\leq \alpha_u \\ \min_{i \in \{1,\cdots,m\}, j \in \{1,\cdots,n\}} \{|v_{p_i} - v_{q_j}|\} &\leq \alpha_v, \end{aligned} \tag{9}$$

where $(u_{\mathbf{p}}, v_{\mathbf{p}})$ is the velocity of the position $(x_{\mathbf{p}}, y_{\mathbf{p}})$ of point $\mathbf{p}$, and $\alpha_d, \alpha_u, \alpha_v$ are thresholding constants.

We already eliminated most noise and made scenes low-clutter on outdoor-scenes (Fig. 1(e)). In this case, the nearest-neighbor clustering is the most effective method except for the scenes which include interfering objects [19]. We handled interfering objects in the phase of tracking.

C. Active Contours

Contours of the clustered lines are extracted by using snakes [10], [19]. The discrete energy function of a snake of the contour $\{\mathbf{p}_1, \cdots, \mathbf{p}_N\} \in S$ is defined as:

$$E = \sum_{i=1}^{N} (\alpha_i E_{cont} + \beta_i E_{curv} + \gamma_i E_{image}), \qquad (10)$$

where E_{cont} is the continuity energy, E_{curv} is the smoothness energy, E_{image} is the edge-attraction energy, and $\alpha_i, \beta_i, \gamma_i \geq 0$ are weights of each energy, respectively. These energies are defined as:

$$\begin{array}{lcl} E_{cont} & = & ||\mathbf{p}_i - \mathbf{p}_{i-1}||^2 \\ E_{curv} & = & ||\mathbf{p}_{i-1} - 2\mathbf{p}_i + \mathbf{p}_{i+1}||^2 \\ E_{image} & = & -||\nabla I||. \end{array} \qquad (11)$$

For the computation of snakes, we calculate the convex hull [20] for each cluster as the initial contour (shown in Fig. 1(f)).

IV. Tracking Detected Objects

To solve the correspondence problem for detected ojects in different frames, we defined the similarity between an object of the previous frame $R_{prev}(m)$ and an object of the current frame $R_{cur}(m')$, using estimated positions of lines by optical flow, i.e.,

$$\begin{array}{lcl} \{\hat{\mathbf{p}}_1, \cdots, \hat{\mathbf{p}}_n\} & \in & \hat{S} \\ \{\hat{\mathbf{S}}_1, \cdots, \hat{\mathbf{S}}_{N_m}\} & \in & \hat{R}_{prev}(m). \end{array} \qquad (12)$$

Then the similarity is defined as follows:

$$\rho(m, m') = \frac{\sum_{j=1}^{N_m} \sum_{i=1}^{n_j} l_{ij}}{\sum_{j=1}^{N_m} |S_j|}, \qquad (13)$$

where $|S_j|$ is the number of elements in $|S_j|$, and

$$l_{ij} = \begin{cases} 1 & \text{if } \hat{\mathbf{p}}_i \in R_{cur}(m') \text{ where } \hat{\mathbf{p}}_i \in \hat{S}_j \\ 0 & \text{otherwise.} \end{cases} \qquad (14)$$

Then these two objects $R_{prev}(m)$ and $R_{cur}(m')$ are considered as the same object if the similarity $\rho(m, m')$ is non-zero value, i.e.,

$$R_{prev}(m) \equiv R_{cur}(m'), \quad \text{if} \quad \rho(m, m') \neq 0. \qquad (15)$$

In addition, we defined following states of tracked objects for handling them as special cases:

1) *Occluded*
   If an object disappears without reaching the borders of the frame, the object is considered to be *occluded*. The center of the contour and velocity of the center are registered to the list of occluded objects. Occluded objects are deleted from the list when all estimated positions of points of the contour get out of the frame.
2) *Reappeared*
   Newly detected objects whose contours are inside the frame (i.e., unattached to the borders of the frame) are regarded as *reappeared*. The corresponding occluded object is searched from the list. The center of each occluded object is recalculated by using a Kalman filter [21] from its tracked history of positions. Then the closest object is considered as the same object.
3) *Merged*
   If a detected object of the current frame includes estimated lines of two or more objects of the previous frame, the detected object is regarded as *merged*; i.e., two or more objects of the previous frame satisfy (15) with a single object of the current frame. Each velocity of merged objects is stored for finding corresponding objects when they separate again.
4) *Separated*
   If two or more detected objects include estimated lines of one object of the previous frame, these detected objects are regarded as *separated*; i.e., a single object of the previous frame satisfies (15) with two or more objects of the current frame. If the object of the previous frame is labeled as *merged*, corresponding objects are found according to their velocity.

V. Experimental Results

We tested our method on various sequences including cars (Fig. 2(a)), pedestirans (Figs. 2(b) and 2(e)), approaching cars (Fig. 2(c)), occluded cars (Fig. 2(d)), and interfering objects (Fig. 2(e)). All sequences shown in Fig. 2 are recorded on outdoor-scenes that include the sky, trees, buildings, grounds, and snow. They include several kinds of noise caused by illumination changes, small movement in the background, and reflection. However, our results showed remarkable robustness against these environments. Our method succeeded detecting and tracking moving objects accurately in all video sequences in Fig. 2, even though these sequences had many causes of noise. For instance, Figs. 2(b) and 2(c) include the sky which causes illumination changes. These sequences also include trees and grass which cause noise of small movement in the background. Figs. 2(b) and 2(e) include buildings and ground covered by snow which causes reflection. We also succeeded in tracking occluded cars and interfering pedestrians in Figs. 2(d) and 2(e), respectively.

Table I shows the composition of edges of each video sequence in Fig. 2; i.e., what percentage of edges belongs to detected objects or is removed as noise. The removed edges are classified here as *Masked Edges*, *Background Edges*, and *Others*. The edges masked by (6) are defined as *Masked Edges*. *Masked Edges* do not include restored edges. The edges subtracted as backgrounds from restored lines are defined as *Background Edges*. *Others* represent edges of the removed objects whose area is too small, e.g., fading-out and occluding objects.

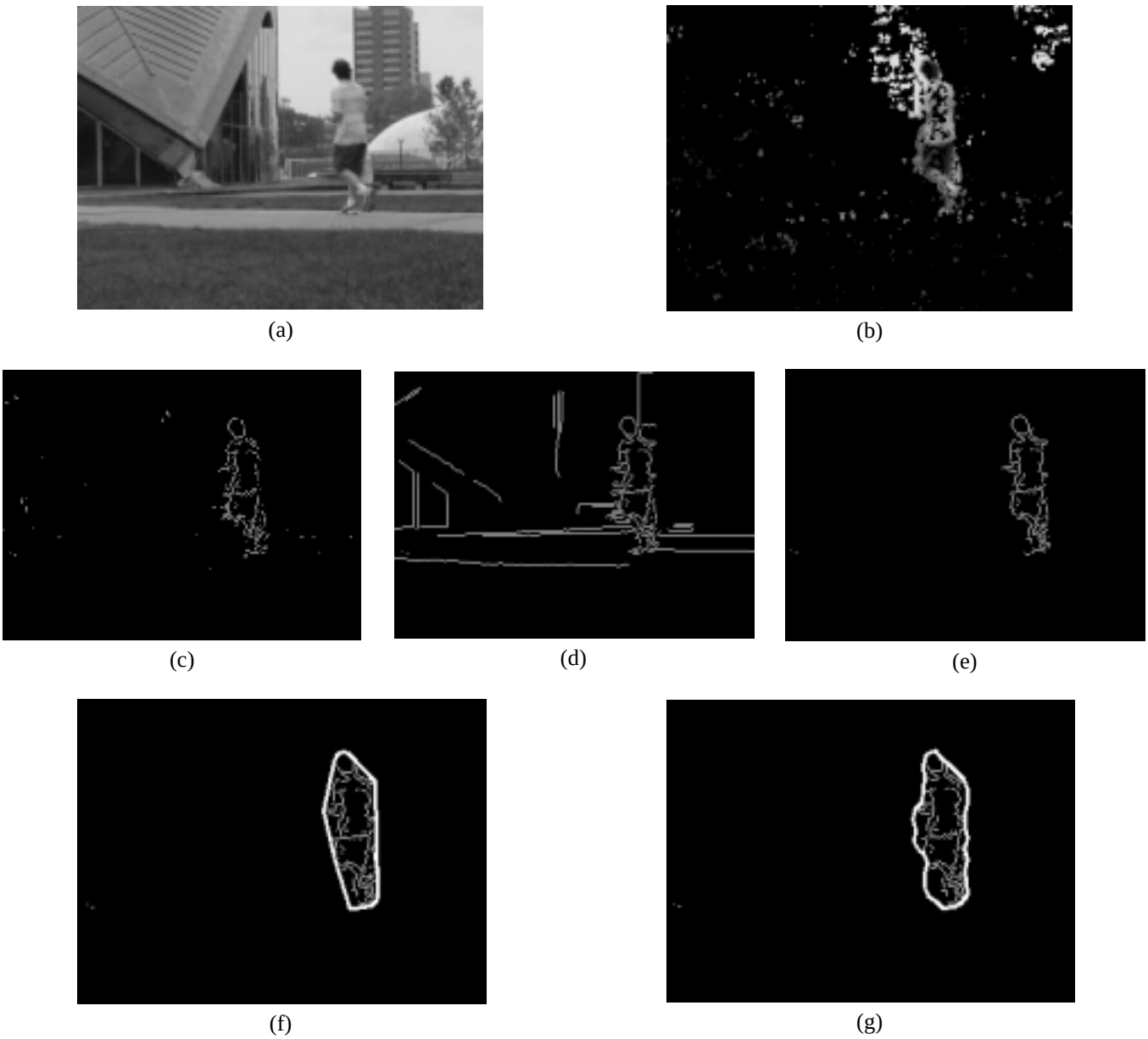

Fig. 1. The flow of detecting objects. (a)Original image. (b)Masked image by (6). Many noise are detected in the image. (c)Masked edges. (d)Restored lines. Several background lines are also restored. (e)Background lines of the previous frame are subtracted. (f)Convex hull. (g)Finally detected contours.

We used video sequences of QVGA(320x240) on a Pentium-M 900MHz and 512MB memory PC for the test. The average computation time of our implementation was 0.089 s/frame.

In our experiments, we chose the following parameters. The region size of optical flow was 5x5. $\tau = 1.0$ in (6). $\mu = 0.5$ in (7). $\alpha_d = 17$, $\alpha_u = 1.0$, and $\alpha_v = 1.0$ in (9), $\alpha = \beta = \gamma = 1.0$ in (10), identical to all points.

VI. Conclusion

We proposed a new system for the detection and tracking of moving objects. Our method is a contour-based detection, which allows users to obtain more accurate information than previous methods using rectangles or ellipses. The computation of our system required 0.089 s/frame on a 900 MHz processor, which satisfies requirement as a real-time system.

References

[1] L. Li, W. Huang, I. Y. H. Gu, and Q. Tian, "Foreground object detection in changing background based on color co-occurrence statistics," in *Proc. IEEE Workshop on Applications of Computer Vision*, 2002.

[2] C. Stauffer and W. E. L. Grimson, "Adaptive background mixture models for real-time tracking," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 1999.

[3] A. Mittal and N. Paragios, "Motion-based background subtraction using adaptive kernel density estimation," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, vol. 2, 2004, pp. 302–309.

TABLE I

THE COMPOSITION OF EDGES(%)

	Objects	Masked Edges	Background Edges	Others
(a)	11.9	74.2	13.5	0.4
(b)	4.0	78.7	17.2	0.0
(c)	5.3	90.1	4.5	0.1
(d)	9.0	78.1	12.6	0.3
(e)	1.7	88.7	9.6	0.0

[4] A. Elgammal, D. Harwood, and L. Davis, "Non-parametric model for background subtraction," in *Proc. European Conf. Computer Vision*, vol. II, May 2000, pp. 751–767.

[5] J. L. Barron, D. J. Fleet, and S. S. Beauchemin, "Performance of optical flow techniques," *Int'l J. Computer Vision*, vol. 12(1), pp. 43–77, February 1994.

[6] J. H. Duncan and T.-C. Chou, "On the detection of motion and the computation of optical flow," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 14, no. 3, pp. 346–352, March 1992.

[7] V. Caselles and B. Coll, "Snakes in movement," *SIAM J. Numerical Analysis*, vol. 33, pp. 2445–2456, 1996.

[8] S. S. Beauchemin and J. L. Barron, "The computation of optical flow," *ACM Computing Surveys*, 1995.

[9] J. Sethian, "Level set methods and fast marching methods," *Cambridge Univ. Press*, 1999.

[10] M. Kass, A. Witkin, and D. Terzopoulos, "Snakes: Active contour models," *Int'l J. Computer Vision*, pp. 321–331, 1988.

[11] D. Cohen, "On active contour models and balloons," *CVGIP: Image Understanding*, vol. 53, pp. 211–218, 1991.

[12] V. Caselles, R. Kimmel, and G. Sapiro, "Geodesic active contours," in *Proc. IEEE Int'l Conf. Computer Vision*, 1995, pp. 694–699.

[13] L. Qiu and L. Li, "Contour extraction of moving objects," in *Proc. IEEE Int'l Conf. Pattern Recognition*, vol. 2, 1998, pp. 1427–1432.

[14] N. Paragios and R. Deriche, "Geodesic active contours and level sets for the detection and tracking of moving objects," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 22, no. 3, pp. 266–280, March 2000.

[15] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *Proc. Int'l Joint Conf. Artificial Intelligence*, 1981, pp. 674–679.

[16] B. K. P. Horn and B. G. Schunk, "Determining optical flow," *Artificial Intelligence*, vol. 17, pp. 185–203, 1981.

[17] E. P. Simoncelli, E. H. Adelson, and D. J. Heeger, "Probability distributions of optical flow," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 1991, pp. 310–315.

[18] J. Canny, "A computational approach to edge detection," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 8, no. 6, pp. 679–698, November 1986.

[19] E. Trucco and A. Verri, *Introductory Techniques for 3-D Computer Vision*. Prentice Hall, 1998.

[20] J. Sklansky, "Measuring concavity on a rectangular mosaic," *IEEE Trans. Computers*, no. 12, pp. 1355–1364, November 1972.

[21] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Trans. ASME - J. Basic Engineering*, vol. 82, pp. 35–45, March 1960.

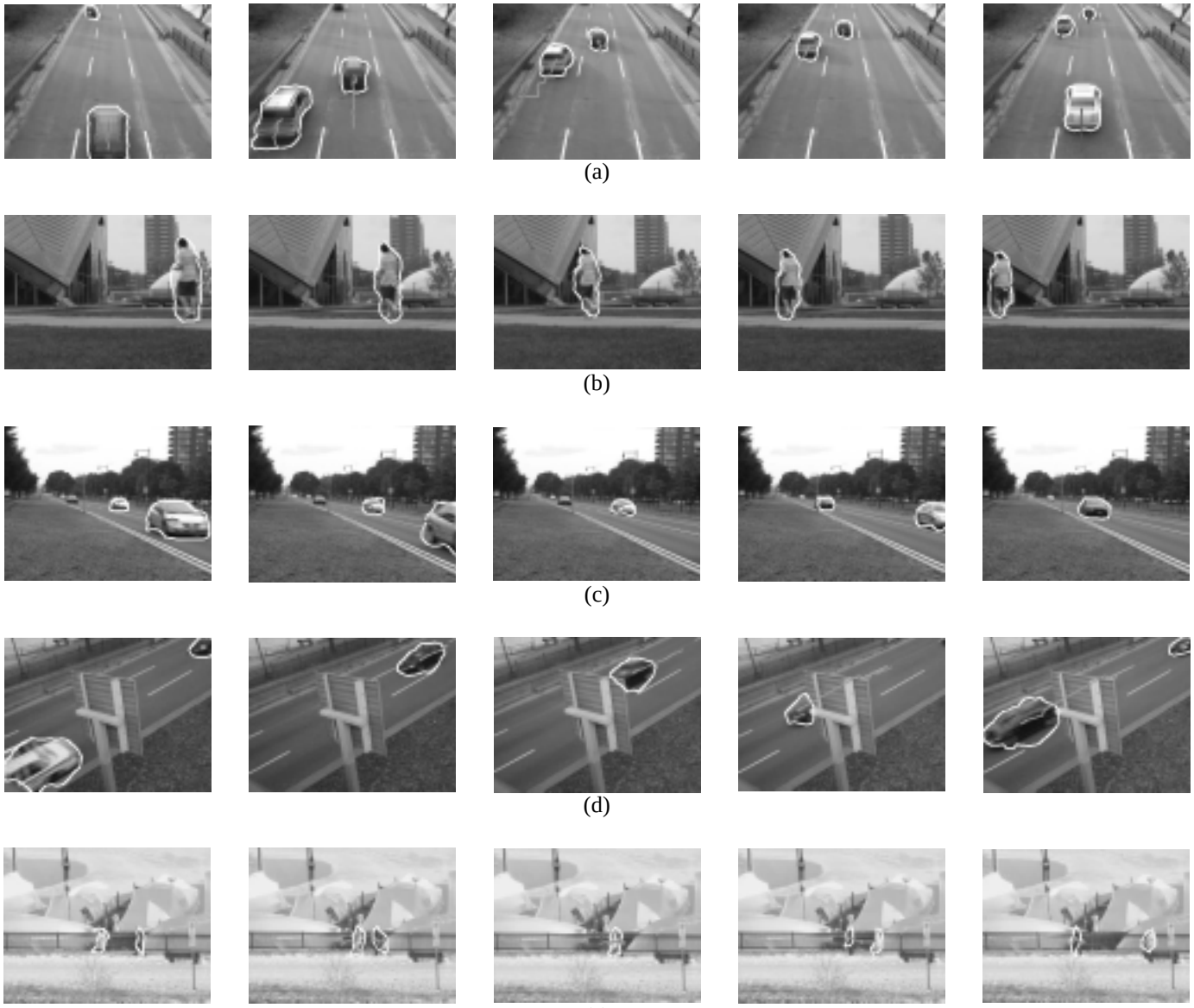

Fig. 2. Results of detection and tracking moving objects. (a)Cars. (b)A pedestrian. (c)Approaching cars. (d)Occluded cars. (e)Interfering pedestrians.