Tracking of Multiple Moving Objects Using Split-and-Merge Contour Models Based on Crossing Detection

Shoichi Araki†‡, Naokazu Yokoya† and Haruo Takemura†

† Graduate School of Information Science,
Nara Institute of Science and Technology, Nara 630-01, JAPAN

‡ Central Research Laboratories,
Matsushita Electric Industrial Co., LTD., Kyoto 619-02, JAPAN

Abstract

The problem addressed in this paper is the automatic detection and tracking of multiple moving objects. Active contour models (snakes) are useful for detecting and tracking boundaries of deformable objects. However, Snakes can not track multiple objects correctly when they apparently overlap. This paper proposes a split-and-merge contour model based on detecting self- and mutual-crossings of the model. The proposed method can find occlusions of multiple objects by detecting mutual-crossing of different contour models which extract different objects. Multiple different contours which are mutually crossing are merged into a single contour. The merged contour model is split into multiple contour models when overlapping objects leave away from each other. We have implemented the method on DSP boards for real-time tracking of multiple moving objects in a dynamic image.

1 Introduction

Motion tracking is one of the most important problems in computer vision. Active contour models (Snakes)[1] provide a promising approach to the problem. Snake is a parameterized curve $\mathbf{v}(s) = (x(s), y(s))$ $(0 \le s \le 1)$ on an image plane (x, y), which is deformed to detect object boundary by minimizing the following energy functional:

$$E_{Snakes} = E_{int}(\mathbf{v}) + E_{image}(\mathbf{v}) + E_{ext}(\mathbf{v}), \qquad (1)$$

where E_{int} is an internal energy associated with splines, E_{image} is an image energy such as edge potential and E_{ext} is an external energy associated with external forces. Snakes are potentially useful for not only detecting target objects but also tracking them in noisy environments[1][2].

However, Snakes have the following two difficulties in their application to motion tracking in a time-varying image sequence:

1. Automatic extraction of multiple objects,
2. Tracking of apparently overlapping objects.

The first problem primarily concerns initial setting of contour models. In existing methods, initial contours for moving target objects have to be carefully set around the actual boundary of each object because snakes including multiple objects can not extract each of them individually. In most previous works, only semi-automatic motion tracking is realized since initial contours are set manually in such an application. Thus, it is essential to automatic tracking of multiple moving objects that they are automatically extracted in the first frame of an image sequence. The second problem in tracking multiple moving objects is that snakes can not track their boundaries correctly when they apparently overlap, since a contour model is attracted by edges which are extracted by another model.

In order to overcome these difficulties, we propose a split-and-merge contour model based on detecting self- and mutual-crossings of the model. Splitting contour models based on detecting their self-crossings [3] have recently been proposed by the authors for automatic extraction of multiple objects. The self-crossing is caused by employing the area term for energy function[4]. An initial single contour, for which an image frame can be simply selected, iteratively splits into multiple contours by cutting at crossing points for extracting each object individually. The splitting contour model can extract multiple moving objects by using optical flows. A contour is attracted by only pixels with motion cues by setting the image energy E_{image} such as edge potential to be zero when an optical flow calculated at each discrete point of the contour is small. By using the splitting contour model, multiple moving objects can be extracted as some groups of pixels with motion cues, which are smoothly connected.

We propose a methodology of merging multiple contour models for finding apparent overlapping of multiple moving objects by detecting mutual-crossing of different contour models which extract different objects. Multiple different contours which are mutually crossing are merged into a single contour. When overlapping objects leave away from each other, the merged contour is split into multiple contours again.

This paper is structured as follows. In Section 2, we describe the concrete algorithm of split-and-merge contour models. In Section 3, in order to prove the effectiveness of the proposed method, we apply the model to tracking walking persons which are apparently overlapping in a dynamic image of a room scene. The proposed method has been successfully implemented on both a standard workstation and a DSP-based experimental platform. Section 4 summarizes the present work.

2 Split-and-Merge Contour Models

One of the most significant problems in tracking multiple moving objects comes from apparent overlapping of objects; i.e., it is difficult to track multiple objects correctly when they apparently overlap. In order to overcome this problem, we propose a split-and-merge contour model which is based upon detecting self- and mutual-crossings of contours. In this paper, we define a contour model as a set of n discrete points $\mathbf{v}_i(x_i, y_i)$ $(i = 1, 2, \cdots, n)$. In Section 2.1, we describe a method of splitting a contour model based on detecting its self-crossings for automatic extraction of multiple objects. We then explain a method of merging multiple contour models based on detecting mutual-crossing of different contours which is caused by the overlapping of different objects in Section 2.2. In Section 2.3, we explain how to correctly match objects before and after their overlapping. We finally present a concrete procedure for tracking multiple moving objects in Section 2.4.

2.1 Splitting of a contour model

We briefly describe a splitting contour model [3] that is based upon detecting its self-crossings. This contour model is shown to have potentials for automatically detecting multiple moving objects in dynamic images. In the splitting contour model, a contour is divided into multiple closed contours by detecting its self-crossings which are caused by minimizing the energy function including the area term[4]. In the following, we first describe the characteristic of the area term and then explain a method of splitting a contour model.

The area term E_{area} for realizing a "balloon" model[5] can be used as E_{ext}. The area term E_{area} of a contour model $\mathbf{v}_i(x_i, y_i)$ $(i = 1, 2, \cdots, n)$ is defined as

$$E_{area} = \frac{1}{2}\sum_{i=1}^{n}[x_i(y_{i+1} - y_i) - (x_{i+1} - x_i)y_i], \quad (2)$$

where $\mathbf{v}_{n+1}$ $(x_{n+1}, y_{n+1}) = \mathbf{v}_1$ (x_1, y_1). Figure 1 illustrates the calculation of the area term. The first term of the right side of Eq.(2) means a sum of signed areas of rectangles computed on the y axis, whose height is x_i and width is $(y_{i+1} - y_i)$. The second term of the right side of Eq.(2) represents the same calculation for rectangles on the x axis. The area term E_{area} is defined as the average of these two terms. In Fig.1, the sign of E_{area} becomes positive if the sum of rectangular areas is calculated along the contour clockwise. If E_{area} is calculated counter-clockwise, it becomes negative. Figure 2 shows typical behaviors of a contour model with the area term, whose initial contour includes two objects in a synthetic image. The contour is crossed at various parts since E_{area} is computed as negative at crossing parts where contour points are rearranged in counter-clockwise order. If the self-

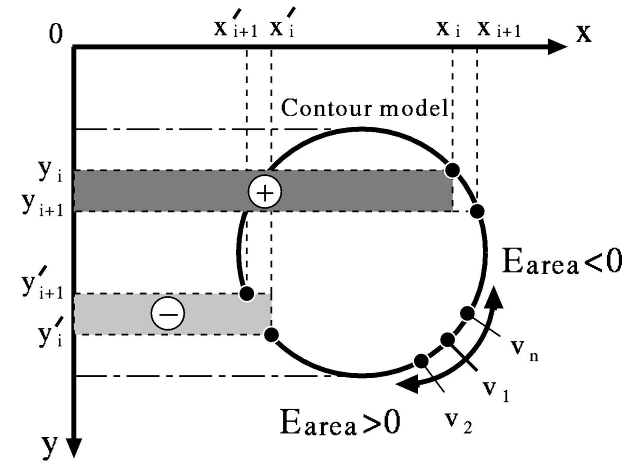
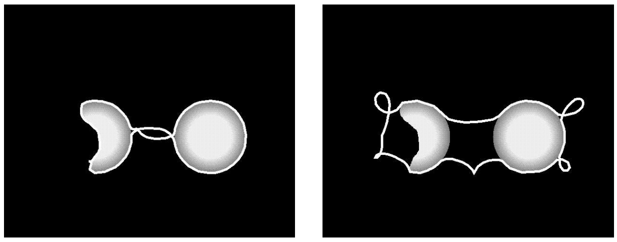


Figure 1: Calculation of the area term.



(a) Crossing between objects (b) Crossing at various parts

Figure 2: Self-crossing of a contour model with area term.

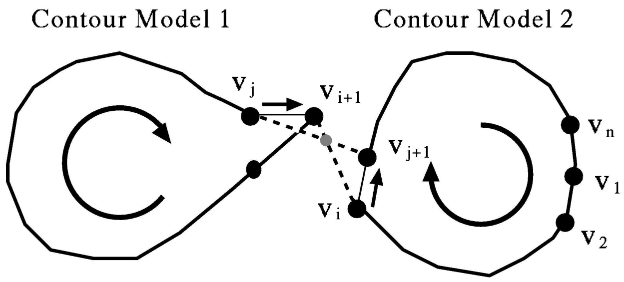


Figure 3: A process of splitting a single contour into two parts.

crossing contour continues to be deformed, crossing parts expand further and fail to detect objects. Splitting contour models detect such self-crossings and split into multiple parts by cutting themselves at crossing points.

Next, a method of splitting a contour model is described. A contour is divided into multiple parts by cutting it at the self-crossing points. Self-crossings of a contour model $\mathbf{v}_i(x_i, y_i)(i = 1, 2, \cdots, n)$ is detected by examining two different segments, say $\mathbf{v}_i\mathbf{v}_{i+1}$ and $\mathbf{v}_j\mathbf{v}_{j+1}$ $(j \neq i-1, i, i+1)$, have a crossing point. For example, a contour model is judged to have a crossing point when there exist real numbers p and q $(0 \leq p \leq 1$ and $0 \leq q \leq 1)$ which satisfy the following equation:

$$p(\mathbf{v}_{i+1} - \mathbf{v}_i) + \mathbf{v}_i = q(\mathbf{v}_{j+1} - \mathbf{v}_j) + \mathbf{v}_j. \quad (3)$$

A self-crossing contour model is cut at crossing points and splits into multiple different contours. Figure 3 illustrates a process of dividing a contour into two parts. Two new contours are then created as $\{\mathbf{v}_1, \cdots, \mathbf{v}_i, \mathbf{v}_{j+1}, \cdots, \mathbf{v}_n\}$ and $\{\mathbf{v}_{i+1}, \cdots, \mathbf{v}_j\}$, respectively. Discrete points of each contour are rearranged in clockwise order to maintain the sign of E_{area} positive.

2.2 Merging of multiple contours

Different contour models which are tracking different objects are merged into a single contour model when a mutual crossing is detected; i.e., different contours are merged when their tracking objects apparently overlap. The process of merging two contours into a single one is illustrated in Fig. 4. Mutual-crossings can be detected by a process similar to that for detecting self-crossings using Eq. (3). In this case, two different segments, $\mathbf{v}_i\mathbf{v}_{i+1}$ belonging to the contour model 1 and $\mathbf{v}_j\mathbf{v}_{j+1}$ belonging to the contour model 2 are examined to have a crossing point. Different contours which are mutually crossing are merged into a single contour. The merged contour is defined as a convex hull including all discrete points of contours which are mutually crossing as shown in Fig. 4.

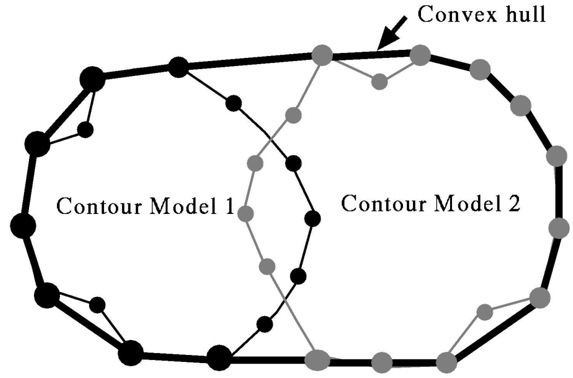


Figure 4: A process of merging two contours into a single contour.

We employ the Graham's method [6] for determining a convex hull from the merged contours.

2.3 Tracking of objects by feature matching

Multiple moving objects can be detected by splitting and merging contour models described above. The problem still remains is how to correctly match objects before and after their overlapping. We employ a simple matching method based on computing a textural feature of an object region. In each frame, cumulative intensity histogram $H_m(k)$ is computed within a region extracted by a converged contour model as a moving object. Note that m $(m = 1, 2, \cdots)$ is the contour model number and k denotes a gray level. At the first frame, features $H_m(k)$ $(m = 1, 2, \cdots)$ of extracted regions are stored in the feature list $\mathcal{H}_l(k)$ $(l = 1, 2, \cdots)$ of extracted moving objects. Note that l is the number associated with an extracted moving object in all previous frames. After the second frame, if a model does not split, $\mathcal{H}_l(k)$ is updated to $H_m(k)$, where $m = l$. When a model splits into multiple models, the difference measure $d_{m,l}$ between features $H_m(k)$ of moving objects extracted at the current frame and features stored in the feature list $\mathcal{H}_l(k)$ is calculated by using the following equation in all combinations of m and l.

$$d_{m,l} = \sum_{k=0}^{255} |H_m(k) - \mathcal{H}_l(k)|. \quad (4)$$

We assume that if the difference $d_{m,l}$ is small, the moving object m extracted in the current frame is similar to the moving object l extracted in past frames. Thus, the object m in the current frame is matched to the object l which minimizes the difference $d_{m,l}$. If the difference $d_{m,l}$ is larger than a threshold, the feature $H_m(k)$

is newly stored in the feature list $\mathcal{H}_l(k)$ as a new moving object extracted by a new split contour. Such a case occurs when multiple moving objects which are overlapping in the first frame leave away from one another in the succeeding frame.

2.4 Procedure for tracking multiple moving objects

An actual procedure for tracking multiple moving objects in an image sequence $I(T)(T = 1, 2, \cdots)$ is described below, where T denotes a frame number. A contour model is represented by $\mathbf{v}_i(x_i(t), y_i(t))$ $(i = 1,2, \cdots, n)$ where t is the number of iterations at each frame.

[STEP1] Inputting images
Input two images $I(T)$ and $I(T+1)$. If it is the first time to do this step then T is set to be 1. Also set the number t of iteration to be 0 at each frame .

[STEP2] Setting an initial contour
When $T = 1$, set an initial contour $\mathbf{v}_i(x_i(0), y_i(0))$ so as to include all moving objects. The image border may be simply selected as an initial contour.

[STEP3] Contour deformation
Move all contour points $\mathbf{v}_i(x_i(t), y_i(t))$ $(i = 1,2, \cdots, n)$ once by minimizing a contour energy E_{Snakes} using the greedy algorithm[7] and $t = t + 1$. The number of actually moved points is stored in C_{move}.

[STEP4] Splitting contours
Split contours at self-crossing points according to the method described in Section 2.1.

[STEP5] Merging contours
Merge mutually crossing contours into a single contour according to the method described in Section 2.2.

[STEP6] Generating new contour points
Generate a new contour point between two adjacent points $\mathbf{v}_i$ and $\mathbf{v}_{i+1}$ which satisfy the condition $|\mathbf{v}_{i+1} - \mathbf{v}_i| > D_{TH}$. D_{TH} is a threshold which means the maximum distance between adjacent discrete points. Such a new point can be generated in the middle of two points or at the location which decreases E_{Snakes} most in a small search space between two points. We can select D_{TH} according to the object shape and the distance between different objects.

[STEP7] Deleting contour points
Delete sharp parts of contours considering that they are trapped by small noises. A discrete point $\mathbf{v}_i$ is deleted when the condition $\theta < \theta_{TH}$ is satisfied, where θ is an angle between $\overrightarrow{\mathbf{v}_i\mathbf{v}_{i-1}}$ and $\overrightarrow{\mathbf{v}_i\mathbf{v}_{i+1}}$. The smaller the angle θ is, the sharper the shape of the contour is. Small contours composed of less than 5 discrete points are also eliminated since such small objects are not our target objects. Contour models are expected to escape from small noises by this deletion process.

[STEP8] Determining the termination of contour deformation
If the condition $C_{move} \leq C_{TH}$ or $t_{max} \leq t$ are satisfied, terminate the contour deformation at the image $I(T)$ and proceed to STEP9 else proceed to STEP3. C_{TH} and t_{max} are predetermined thresholds.

[STEP9] Calculation of image feature of moving objects
Calculate a cumulative intensity histogram within the region surrounded by each contour model as the feature of moving object.

[STEP10] Matching of moving objects
Match moving objects using the similarity measure defined in Eq.(4). Set $T = T + 1$ and proceed to STEP1.

3 Experiments

In this section, we apply the proposed method to tracking of multiple moving objects in a time-varying image sequence. The method has been succesfully implemented on both a standard workstation and a DSP-based experimental platform: First three experiments are carried out on the workstation and final one is on DSP boards. In Section 3.1, we show the result of automatic extraction of multiple moving objects in the first frame of an image sequence of a room scene using the splitting contour model. In Section 3.2, we then present the result of tracking temporarily overlapping objects. We also show the experimental result of extracting a newcomer in a scene in Section 3.3. All the programs in these experiments were written in C and were executed on a standard workstation (DEC 3000), for which all computation times are given. Section 3.4 describes an additional experiment which aims at real-time object tracking using DSP boards.

In all the experiments below, we use the following energy terms.

Internal energy E_{int}:

$$E_{spline}(\mathbf{v}_i) = \frac{1}{2}\sum_{i=1}^{n}(w_{sp1}|\mathbf{v}_i - \mathbf{v}_{i-1}|^2 + w_{sp2}|\mathbf{v}_{i+1} - 2\mathbf{v}_i + \mathbf{v}_{i-1}|^2) \quad (5)$$

$$E_{dist}(\mathbf{v}) = \frac{1}{2}\sum_{i=1}^{n} w_{dist}|d_{av} - |\mathbf{v}_i - \mathbf{v}_{i-1}||^2 \quad (6)$$

Image energy E_{image}:

$$E_{edge}(\mathbf{v}) = -\frac{1}{2}\sum_{i=1}^{n} w_{edge}|\nabla I(\mathbf{v}_i)|^2 \tag{7}$$

External energy E_{ext}:

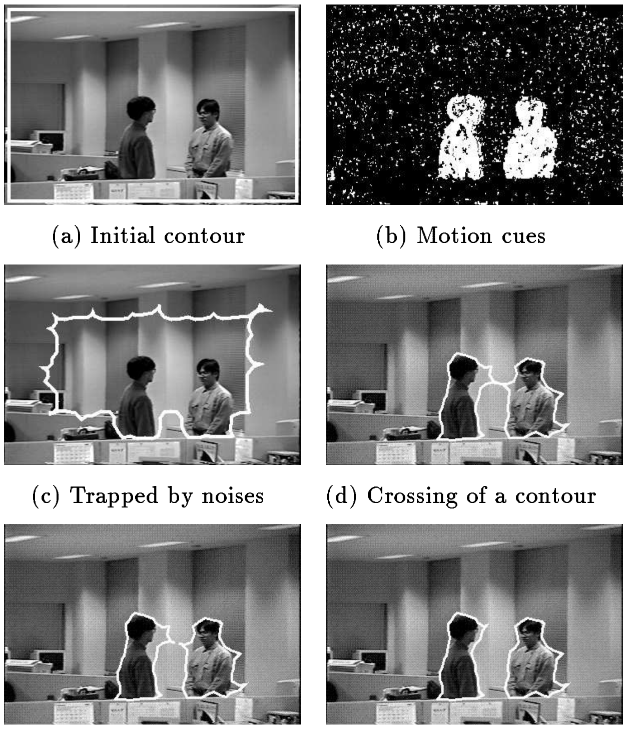
$$E_{area}(\mathbf{v}) = \frac{1}{2}\sum_{i=1}^{n} w_{area}[x_i(y_{i+1} - y_i) - (x_{i+1} - x_i)y_i] \tag{8}$$

Note that $\mathbf{v}_i$ (x_i, y_i) denotes a discrete point of a contour and $\mathbf{v}_0$ and $\mathbf{v}_1$ are equal to $\mathbf{v}_n$ and $\mathbf{v}_{n+1}$, respectively. E_{spline} represents the first and second order continuity constraints [1], E_{area} is an area of a region surrounded by a contour model [4], E_{dist} evenly spaces discrete points in the average distance d_{av} between adjacent discrete points [7], E_{edge} means edge potential defined by spatial gradient of intensity $I(\mathbf{v}_i)$ [1]. Coefficients w_{sp1}, w_{sp2}, w_{area}, w_{dist} and $w_{edge}(\geq 0)$ are weights for the corresponding terms.

3.1 Extraction of multiple moving objects

In Section 1, we have already mentioned that automatic extraction of multiple moving objects at the first frame is essential to automatic motion tracking. The splitting contour model can be easily defined so as to extract multiple moving objects using optical flows. Contour models are attracted only by pixels with motion cues by setting the weight w_{edge} of the image energy E_{edge} to be zero when an optical flow calculated at each discrete point of a contour has a small value. Moving objects can be extracted as groups of pixels with motion cues, which are smoothly connected by the splitting contour model. In experiments, we set $w_{sp1} = 1.0$, $w_{sp2} = 1.0$, $w_{area} = 3.0$ and $w_{dist} = 1.0$, respectively. If optical flows calculated at discrete points are lager than 1.0 pixels / frame, we set $w_{edge} = 2.0$, else $w_{edge} = 0.0$ for extracting only moving objects.

Figure 5 shows an example of extracting walkers in the first frame of time-varying indoor scene images (360×243 pixels, 256 gray levels) taken at intervals of 1/10 second. Figure. 5(a) shows an initial rectangular contour which consists of 96 discrete points. Figure. 5(b) represents a distribution of motion cues whose optical flow vectors estimated from the first and the second frames are larger than 1.0 (pixel/frame). Optical flows are estimated by using the standard least squares method based on the spatial and temporal intensity gradients[8], assuming that motion vectors are uniform in a small region of 3×3 pixels. As shown in Fig.5(c), the contour is trapped by



(a) Initial contour (b) Motion cues
(c) Trapped by noises (d) Crossing of a contour
(e) Split contours (f) Extracted regions

Figure 5: Extraction of walking persons at the first frame in a dynamic image.

other edges except persons since motion cues are incorrectly detected in many places where the intensity changes not by motion but by brightness scattering. However, the contour can escape from such noises because they are small and discontinuous. The contour crossed between two persons and split into two different contours as shown in Figs.5(d) and (e). The splitting contour model successfully extracted two walking persons individually, as shown in Fig.5(f). Though the initial contour includes many edges except for those of walkers, the contour model is not attracted by edges not in motion. Computation time to extract walkers is about 13 seconds. In this experiment, 96 points of initial contour finally reduced to 63 points for extracting two persons.

3.2 Tracking of two temporarily overlapping objects

Figure 6 shows the experimental result of tracking temporarily overlapping walkers in time-varying indoor scene images (360×243 pixels, 256 gray levels) taken at intervals of 1/10 second, which are succeeding frames of the image sequence used in Section 3.1. At the beginning of each frame, the converged contours at the previous frame are moved along optical flow vectors and then expanded by twice minimiza-

(a) 1st frame

(b) Result at 1st frame

(c) 10th frame

(d) Merged at 13th frame

(e) 40th frame

(f) 60th frame

(g) Split at 70th frame

(h) 80th frame

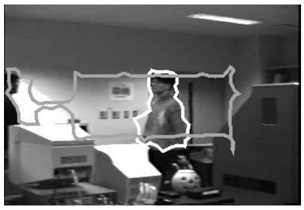
Figure 6: Extraction and tracking of temporarily overlapping walking persons in a dynamic image of a room scene.

tion of E_{Snakes} in which the sign of the weight w_{area} is changed to negative for always including moving objects. The re-deformed contours are used as initial contours in the current frame.

Figures 6(a) and (b) show the first frame with an initial rectangular contour which consists of 96 discrete points and the result of extracting walking persons at the frame, respectively. For visually discriminating two persons, we set the colors of models which extract left and right persons to be white and black, respectively. Figure 6(c) shows the result at the 10th frame just before two different models are mutually crossing. As shown in Fig.6(d), two different contours are merged into a single contour for tracking apparently overlapping persons. Note that the merged contour is represented by dotted lines. Temporarily overlapping persons were correctly tracked by the merged contour as shown in Figs. 6(e) and (f). The merged contour split again at the 70th frame where overlapping persons left away from each other as shown in Fig.6(g). Two split contours successfully tracked two persons through the 80th frame as shown in Fig.6(h). Computation time to extract walkers in the first frame is about 13 seconds same as in the experiment in Section 3.1. In the succeeding frames, it takes about 1 second for tracking at each frame.

(a) First frame

(b) 12th frame



(c) Trapped by a new object

(d) Result at 12th frame

(e) 28th frame

(f) Extracted new object

Figure 7: Extraction of a new person coming into a field of view.

3.3 Extraction of a newcomer in an image sequence

Conventional snakes can not automatically extract newcomers in an image sequence since initial contours have to be set manually for all the target objects. Newcomers can be easily detected by generating a new split-and-merge contour model at each time, where its initial contour is simply selected as an image border. The new contour and existing contourss are controlled not to merge even if they are mutually crossing. A new contour can extract only new objects by setting coefficient $w_{edge} = 0$ when its discrete points are inside the existing contours. In this experiment, we assume that a newcomer is larger than a threshold.

Figure 7 shows the experimental result of extracting a newcomer. Figure 7(a) shows the result in the first frame including a single walking person. A newcomer appeared in the 12th frame from the left side of the image. A new contour model for extracting the newcomer is added along an image border as shown in Fig. 7(b). Though the added contour model extracts a part of the newcomer, it is finally eliminated since the split model is smaller than the predetermined threshold. Such a process is observed in Figs. 7(c) and (d). We also confirm that added contour does not extract an existing person. In all previous frames, an added model is eliminated in the same way as the result of 12th frame. At the 28th frame, the newcomer is successfully extracted because the extracted region becomes larger than the predetermined threshold in the image as shown in Figs. 7(e) and (f).

3.4 Real-time tracking of multiple moving objects using DSP boards

In order to realize real-time tacking of moving objects, we have implemented the method on an experimental image processing system mainly constructed of a video I/O board and six DSP boards "ISHTAR-II"[9] containing TI TMS320C40, which is illustrated in Fig. 8. The video board digitizes NTSC composite signals from a video camera into images which consist of 160×120 pixels, and transfers them to the DSP board 1. Each DSP board processes digitized images and transfers the current result and the image to the next DSP board at every 1/5 second according to a program which is down-loaded from a host computer (SUN SparcStation20) through the MPU board (S-BUS/VME converting interface).

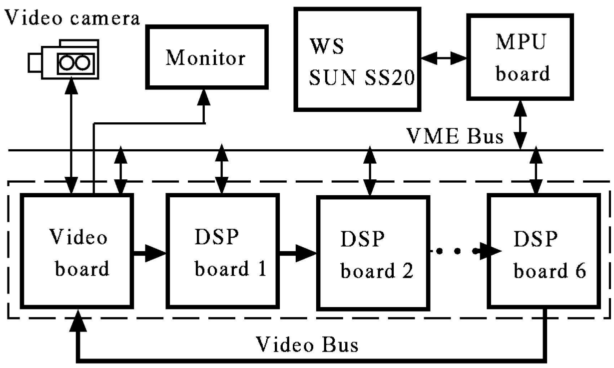
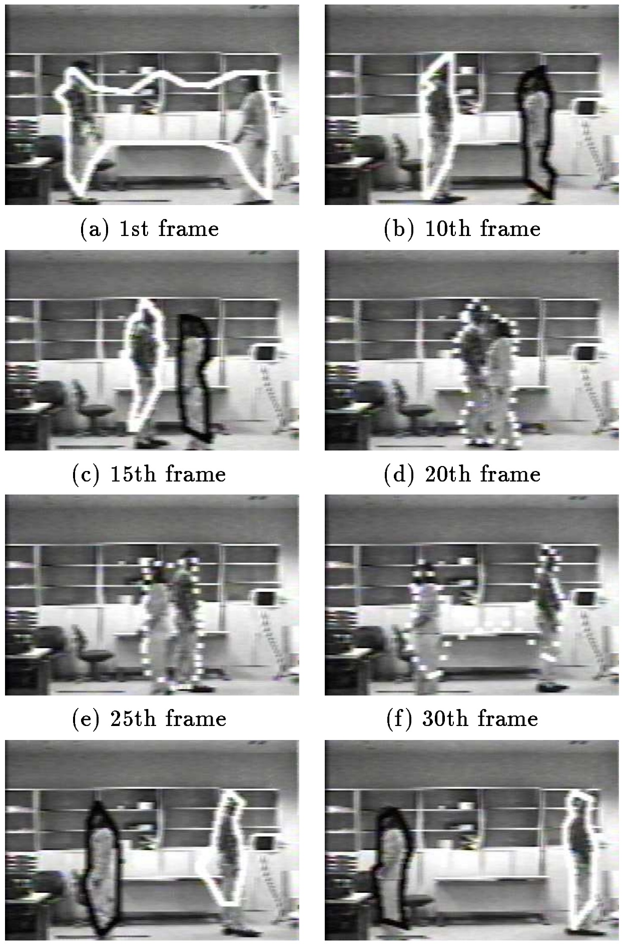


Figure 8: System configuration of a DSP-based experimental platform.

Implementation of the proposed method on DSP boards is a little different from that on a workstation. DSP board 1 stores two succeeding frames and sends them to the second board for calculation of optical flow. The split-and-merge contour model is implemented on DSP board 2 through DSP board 5.



(a) 1st frame (b) 10th frame
(c) 15th frame (d) 20th frame
(e) 25th frame (f) 30th frame
(g) 40th frame (h) 50th frame

Figure 9: Real-time tracking of temporarily overlapping persons using DSP boards.

Each board searches an object boundary with motion 6 times (6 iterations of the greedy algorithm) for every 1/5 second. By pipe-lined processing, object boundaries can be searched 24 times in total in the same frame by these four boards. It should be noted that STEP8 in Section 2.4 is omitted and thus a contour model at earlier frames may not converge to an object boundary. DSP board 2 also expands contour models twice by the same way as described in Section 3.2. They are used as initial contours for the next frame. DSP board 6 matches moving objects when a merged contour is split again. This board also generates an image for displaying the result on a monitor through the video board.

Figure 9 shows the experimental result of real-time tracking using DSP boards. An initial contour shown in Fig. 9(a) consists of 26 discrete points. It takes about 2 seconds to extract two persons individually as shown in Fig. 9(b), since this system searches object boundaries only 24 times in each frame. However, in the succeeding frames digitized every 1/5 second, two persons are successfully tracked by the system. We can observe that temporarily overlapping two persons are successfully tracked as shown in Fig. 9, where the correspondence of contours is indicated by colors and the merged contour is by a dotted line.

We have confirmed that moving objects which have the velocity of less than 10 pixels/sec could be tracked by the proposed method from experiments including the result shown in Fig. 9.

4 Conclusions

We have described a new split-and-merge contour model which can automatically detect and track boundaries of temporarily overlapping multiple moving objects without specifying the number of objects in advance. We have shown the excellent performance of the proposed algorithm for real images of apparently overlapping walkers in a room scene. Real-time tracking has also been demonstrated with the implementation on DSP boards. Our future work includes: 1) tracking of temporarily stopping objects in several frames of an image sequence; 2) improvement of the object matching algorithm by using multiple features of images.

Acknowledgements

This work was supported in part by the Real World Computing Program of the Ministry of International Trade and Industry and also by Grant-In-Aid for Scientific Research under Grant Nos. 08235217 and 08555103 from the Ministry of Education, Science and Culture.

References

[1] M. Kass, A. Witkin and D. Terzopoulos, SNAKES : Active contour models, *International Journal of Computer Vision*, **1**, 321-331 (1988).

[2] K. Fujimura, N. Yokoya and K. Yamamoto, Motion tracking of deformable objects by active contour model using multiscale dynamic programming, *Journal of Visual Communication and Image Representation*, **4**, **4**, 382-391 (1993).

[3] S. Araki, N. Yokoya, H. Iwasa and H. Takemura, A new splitting active contour model based on crossing detection, *Proc. of 2nd ACCV*, **2**, 346-350 (1995).

[4] T. Sakaguchi and K. Ohyama, Snakes with area term, *Proc. IEICE Spring Conf. '91*, D-555, (1991). (in Japanese)

[5] L.D. Cohen and I. Cohen, A finite element method applied to new active contour models and 3D reconstruction from cross sections, *Proc. 3rd ICCV*, 587-591 (1990).

[6] R.L. Graham, An efficient algorithm for determining the convex hull of a finite planar set, *Information Processing Letters*, **1**, 132-133 (1972).

[7] D.J. Williams and M. Shah, A fast algorithm for active contours, *Proc. 3rd ICCV*, 592-595 (1990).

[8] J.K. Kearney, W.B. Thompson and D.L. Boley, Optical flow estimation : An error analysis of gradient-based methods with local optimization, *IEEE Trans. on Patt. Anal. Mach. Intell.*, **PAMI-9**, 229-244 (1987).

[9] M. Shiohara, H. Egawa, S. Sasaki, M. Nagle, P. Sobey and M.V. Srinivasan, Real-time optical flow processor ISHTAR, *Proc. of 1st ACCV*, 790-793 (1993).