Multiple Object Tracking System with Three Level Continuous Processes

Kazuhiro FUKUI, Hiroaki NAKAI, and Yoshinori KUNO
Research and Development Center, Toshiba Corporation
1, Komukai Toshiba-cho, Saiwai-ku, Kawasaki 210, Japan

Abstract

In this paper we report a system for detecting human like moving objects in time-varying images. We show how it is possible to detect the image trajectories of people moving in ordinary indoor scenes. The system consists of three subprocesses: changing region detection, moving object tracking and movement interpretation. The processes are executed in parallel so that each one can recover from the others' errors. This ensures the reliable detection of the trajectories in difficult cases such as movement across complicated backgrounds. We have built a trial detection system using a parallel image processing system. The details of the trial system and experimental results of walking person detection are described.

1 Introduction

Tracking multiple objects has many applications. An example is to examine the movements of customers in a supermarket. The information obtained can be useful for analyzing the customers' flow to improve commodity display and layout in the store. The technique can be applied to analyze traffic flow to detect accidents.

Several systems detecting moving objects are commercially available[1, 2]. Their primary application is to detect intruders by finding any change in the subtraction result between the current image and the reference image or between two or more consecutive frame images. Although such systems can recognize the presence of moving objects in a given scene, they cannot interpret the motion of multiple moving objects. In addition, such systems are sensitive to noise caused by a change in environment: a change in the weather, sun position, swaying leaves in the wind, etc, often yielding false alarms.

In computer vision research, motion analysis is one of the most popular areas. Many references can be found in conference proceedings and journals. Many of them are theoretically sound but are not robust against noise and errors that are inevitable in real applications. In other words, they assume that the image processing can give a perfect result all the time. Therefore, such methods only work in a limited condition where the situation of scene is simple and constant in time: the scene background color is unique, the number of moving objects is small and moving objects do not cross each other.

We aim to develop a system for detecting and tracking multiple moving objects that can work in real environments. Our starting point is the assumption that a complex vision process cannot always yield successful results but that it is unlikely that all subprocesses of the process are failing all the time. This assumption has lead us to the following concepts in designing the system.

1. The whole process is divided into several subprocesses.
2. Each subprocess is working all the time using a current image or data extracted from it.
3. The subprocesses are organized so that they can watch others' results and can control others' behaviors.

A certain subprocess might make a mistake at some moment. However, it might give correct results at the next moment. Or, other subprocesses might detect its failure and could correct its behavior. The combination of such error recovery mechanisms can help the system to work properly.

We have developed a system for tracking multiple moving objects based on the above idea. The main target objects are humans. As the first step of the development, we have restricted the function of the system to detecting and tracking each target as one

0-8186-2840-5/92 $03.00

object. This means that we do not care about the partial movements of the target obtained, such as the movements of people's legs and arms. The purpose of the system is to find people in a given scene and to obtain the trajectory of each person. We leave more detailed motion analysis for the next research stage. This simplification of the problem can add one more assumption that all parts of a target object cannot be tracked all the time but that at least some portion of the objects can be detected and tracked at each moment. This assumption can be used in the same way listed above as basic concepts, making the system much more robust.

The system is implemented on a multi processor system[3]to achieve a real time system. This paper presents the details of the system and experimental results of tracking the customers in a store.

2 Overview

2.1 Basic Concept

The basic idea for the system development is to divide the process into several subprocesses working continuously and to organize them so that they can corroborate each other. We divide the process into three subprocesses: the detection, tracking, and interpretation. We call these the detector, tracker and interpreter, respectively. The detector detects regions with large brightness changes. The system considers them as candidates of moving objects, and the trackers start tracking these regions. The interpreter determines each object's motion from the tracking results.

This division is often used in motion analysis[4]. Conventional systems connect these subprocesses sequentially as shown in Fig.1(a). In such systems, however,the whole process fails even if one of the subprocesses make any small error at some moment. To reduce such failures, motion continuity is often introduced[5, 6, 7]; that is, the current result(result2 in Fig.1(a)) is compared with the previous result(result1 in Fig.1(b)). If the results are not continuous, the current result is discarded as a failure.

Although this continuity verification is often useful in motion analysis, it is not sufficient. Fig.1(b) shows the process flow of the proposed system. The data basically flow from the detector to the interpreter as in conventional systems. However, it differs in that each subprocess is working continuously and that the results of an individual subprocesses are sent to the other subprocesses to corroborate each others' results, or to check for the failure of the others.

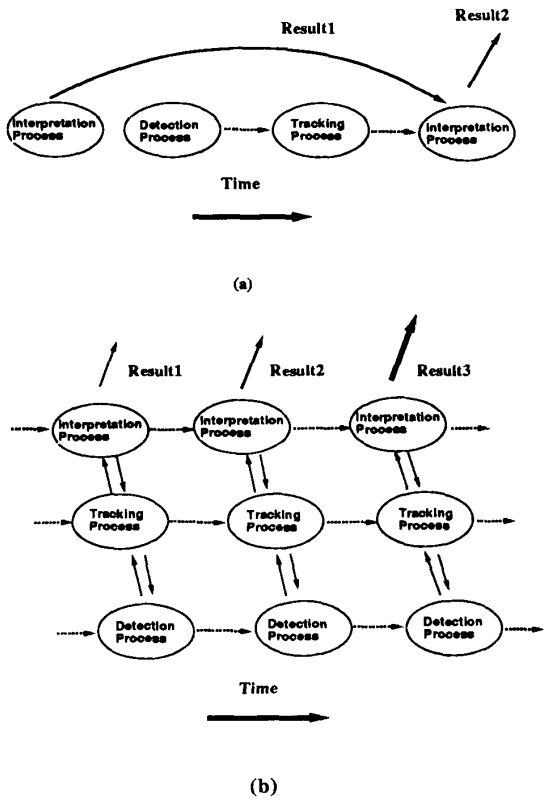

Figure 1: Process for Interpretation of Motion

Figure2 shows the relationship between the subprocesses. The number of trackers depends on the situation. Note that several trackers might track different parts of a single object.

When the detector detects a region with a large brightness change, the system checks all active trackers. If no process is tracking around the detected region, a new tracker is initiated to examine the region. The tracker continues tracking, reporting the current position to the detector and the interpreter. The detector checks whether there is any large change around the received position. Since the detector keeps working independent of the tracker, it should detect the large change region around the position that the tracker is indicating if both processes are properly working. If this is not the case and such a situation continues for a while, the detector terminates the tracker.

The tracker can detect the failure of the detector and can control the detection sensitivity. Several other fac-

tors besides moving objects such as fluorescent lamp flickering may cause a brightness change. The detector might choose some regions whose brightness change is not caused by moving objects as moving objects candidates, initiating trackers. However, these trackers cannot detect any movement in the specified regions in such cases. (Actually, this is detected by the interpreter). This can save the system from failure when the detector makes such mistakes.

Errors in the detector mentioned above may happen often in the same location of the image. In this case, the tracker decreases the sensitivity of the detector in that part of the image. Such cases are often observed with shiny objects, because the brightness on such object faces greatly changes even with slight illumination condition change.

The interpreter receives all tracking results to obtain the trajectory of each moving object. We cannot expect that all trackers correctly chase their target objects. They might sometimes fail during tracking. Multiple trackers might be tracking the same object. Thus, the information that the interpreter receives is a collection of small fragments of trajectories for various object parts. The interpreter combines the fragments coming from the same object into a continuous trajectory. This allows the system to overcome problems arising from the tracking failures.

The interpreter can control the trackers. If it finds a tarcker that is not moving for a while, it terminates the process. If it discovers that multiple trackers are chasing almost same part, it merges them into a new tracker. In addition, we are planning that the interpreter can give some positional constraints between multiple trackers when it finds them chasing the different parts of the same object (for example, a head and a body of a person). This has not been implemented in the experimental system described later.

The final ability of the system depends on the correctness of the operation of combining fragmented tracking results, since the output of the interpreter is the output of the system and there is no recovery method in the current implementation. In order to improve this capability, we are experimenting on adding a new function requesting to examine attributes in the tracked regions from the interpreter to the trackers when the interpreter cannot have confidence in its judgment. Color and texture are examples of attributes. When multiple objects are crossing and multiple trackers are active, the attributes of such tracked regions might be helpful for the final decision. Although we have experimented with color, the full implementation has not been completed.

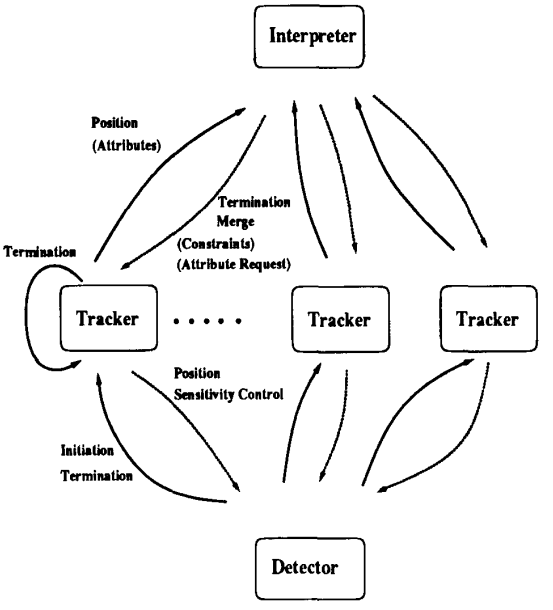

Figure 2: Relation of Processes

2.2 Vision Processor

We have used the vision processor(VP)[3]recently developed to achieve a real time implementation of the idea described in the previous section. Fig.3 shows the configuration of the vision processor. The VP is a kind of multi-window system proposed by[8]. It consists of 16 local modules with a host computer, Sparc Station, by Sun Microsystems. Each module has a local processor, m68030 and three local image memories. The local modules are connected by special image buses. Each module can input any portion of an image from the image buses at video rate and carries out an assigned task. Each local module can communicate to the host workstation by a 16 bit parallel port. The detector is carried out by 2 local modules. 14 local modules are reserved for the tracker. The interpreter runs on the host workstaion. Details of implementation are described in section 4.

3 Three Level Processes

We describe the computation methods of the three processes in this section. Emphasis has been placed on fast computing in developing each process, because continuous watching is very important to realize a robust

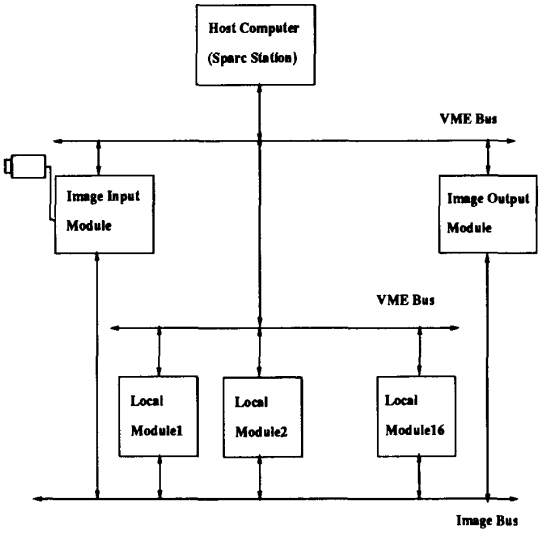

Figure 3: Configuration of Vision Processor

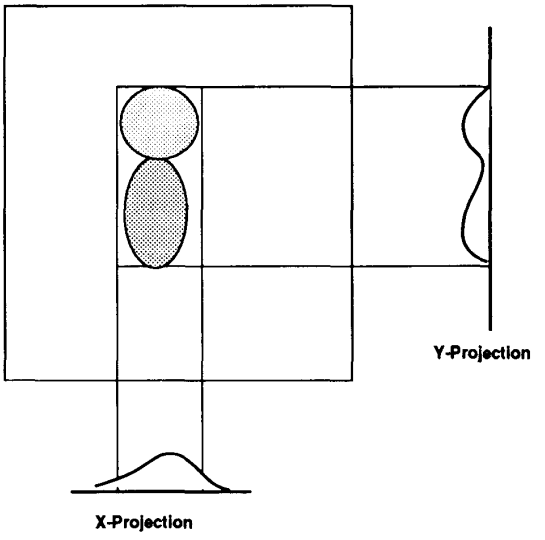

Figure 4: Detector

system based on our idea. The shorter the time interval between successive processes, the higher is the possibility of recovery from any error at the next time. Thus, each process of the system is simple, but the combination of these simple processes achieves a robust system.

3.1 Detector

The detector detects candidate regions for moving objects. It uses subtraction between successive images or between a current image and a reference image without any moving objects. Both methods are available in the current system. In the latter methods, the reference image is updated at certain time interval. Binarization, X-projection, Y-projection, thresholding operation, and computation of a circumscribed rectangular region follow to extract regions with large brightness change[10], as shown in Fig.4. The detector sends the position and the size of the region to initiate a new tracker. If the region size is large, the region is divided into multiple small regions and for each region a tracker is initiated.

3.2 Tracker

Many tracking methods have been proposed, Examples are cross correlation[11, 9],and snakes[12]. In this system, we use a simple method based on a subtraction result between two successive frame images.

When the detector detects a changing region, a local module of the VP is assigned to track the region. The local module inputs the rectangular region in the current image as shown in Fig.5. We call this region a tracking window. The center position of the tracking window is the center of the changing region detected in the detector, and the window size is determined by the changing region size. Then, the local module inputs the same portion of the next frame image, computing subtraction between the tracking windows at the two time points. Actually, subtraction is operated only on 16 small cell regions in the tracking windows as shown in Fig.5 to reduce processing time. Then, the movement vector$(vx, vy)$is calculated by:

$$vx = \frac{\sum_{i=1}^{16} w_i x_i s(i)}{\sum_{i=1}^{16} s(i)} \quad (1)$$

$$vy = \frac{\sum_{i=1}^{16} w_i y_i s(i)}{\sum_{i=1}^{16} s(i)} \quad (2)$$

where, i is the cell number(from 1 to 16), $s(i)$ is the sum of the absolute values above a certain threshold R, in the subtraction result for the cell i, and $w(i)$ is weight(usually 1.0) which can be changed (if desired) to constrain the motion in the window.

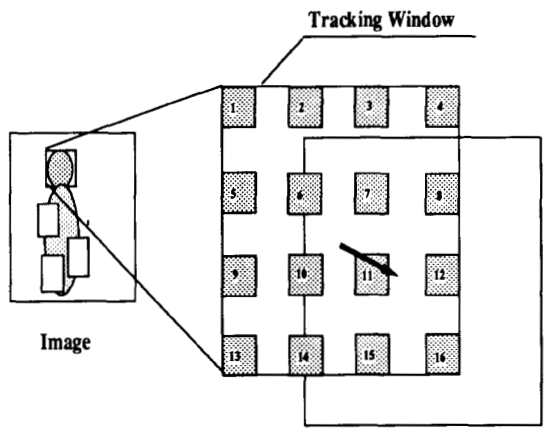

Figure 5: Tracker

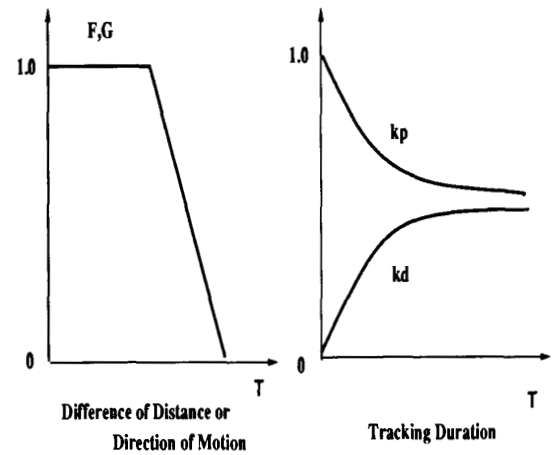

Figure 6: Functions F,G, and the Weights Kp, Kd

The tracking window is translated by the movement vector at the next frame time and the same operation is carried out. Roughly speaking, this operation moves the tracking window towards the center of gravity of the pixels greater than the threshold, R, in the subtraction result.

Although the operation does not guarantee correct tracking, its output is good enough for the interpreter, because the processing interval is short.

3.3 Interpretation Process

The interpreter consists of two subprocesses: the grouping process and the correspondence process. The interpreter receives the position data from all active trackers at every time step. Then, the grouping process classifies the trackers into groups, each of which consists of the trackers chasing the same object. Each group represents an object candidate at a given time. The corresponding process detects correspondent object candidates among those of successive time steps to obtain the trajectories of the individual objects.

3.3.1 Grouping Process

In order to extract candidates of moving objects, an imaginary connection strength between the trackers is considered. The groups of trackers linked by the connection with strength above a threshold value are extracted as object candidates. The strength $S_{ij}(T)$ of the connection between the trackers i and j, after the time duration T during which both trackers are active is given by:

$$S_{ij}(T) = k_p(T)F(p_i(T) - p_j(T)) + k_d(T)G(d_i(T) - d_j(T)) \quad (3)$$

where, F and G are functions of the position difference and the motion direction difference, respectively. p_i,d_i is the position and direction of the tracker i, respectively. Fig.6 shows F,G,k_p and k_d. They are determined so that at the beginning of tracking, pairs of positionally closer trackers are given larger connection strengths and that after some tracking duration, pairs of trackers which are moving to the same direction are given larger connection strengths.

Groups of trackers which are joined by a connection strength above a threshold value are extracted as candidates of moving objects. A list of the candidates is transferred to the correspondence process.

3.3.2 Correspondence Process

The correspondence process selects corresponding object candidates using the successive time step results of the grouping process. In motion analysis, smoothness of motion or continuity of motion is often used[13, 14]. The proposed method is also based on this assumption.

Let an objects candidate be X_i and its component tracker m_{ik} at time t. At the next time step, t+ Δ t, we denote these by X_i' and m_{ik}'. The purpose of the correspondence process is to find the pair of X_i and X_j' with the best matching score. The matching score

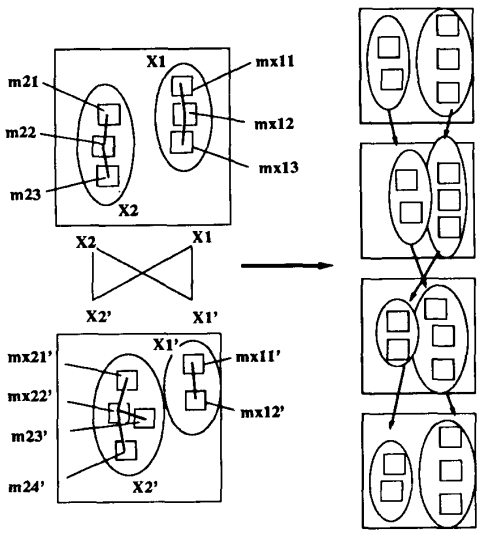

Figure 7: Correspondence Process

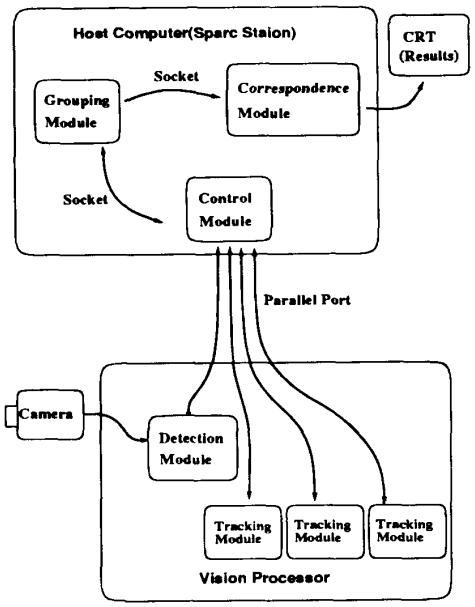

Figure 8: Configuration of Implementation

M_{ij} between the object candidates X_i and X'_j is given by:

$$M_{ij} = \sum_{k,l} H(m_{ik}, m'_{jl}), \tag{4}$$

where H is a function indicating the degree of correspondence between the trackers. The degree of correspondence is determined by a heuristic function considering the position of trackers and the direction of the motion.

The correspondence process selects the best matching score pairs among those with the scores above the threshold, recording the results as the object motion trajectories.

The above description explains the basic computation in the correspondence process. In the real implementation, the process examines the results for the last few time steps if it cannot find good corresponding pairs. In addition, the process does not output the motion interpretation result at every time step. It outputs an individual trajectory result after it has observed good corresponding results for a certain time duration. If the process detects some trackers which do not satisfy the above two cases, that is, the trackers for which the process cannot find the corresponding ones for some time, and whose movements are confined in a small region, the interpreter terminates the trackers. These trackers fail in tracking because of environmental noise, such as illumination changes.

4 Implementation in Vision Processor

We have implemented the method in the VP as shown in Fig.8. The control process monitors the status of the local modules, assigning tasks to appropriate local modules. Fig.9 shows an example of the control flow. The details about the local module management are given in [3].

The detector uses 2 local modules of the VP. They cover the right part and the left part in the whole image. The size of each detection window is 260pixels × 280pixels. In order to achieve real time processes, the local modules sample image data at every four pixels. The position data of detected regions are transferred to the control process through the 16 bit parallel port.

The other 14 local modules can be used for the tracker. The initial position of a tracking window is sent from the control process. The results of tracking is transferred to the interpreter through the control process.

The interpreter is located in the host workstation. The socket function in Unix is used for communication between the processes in the host workstation.

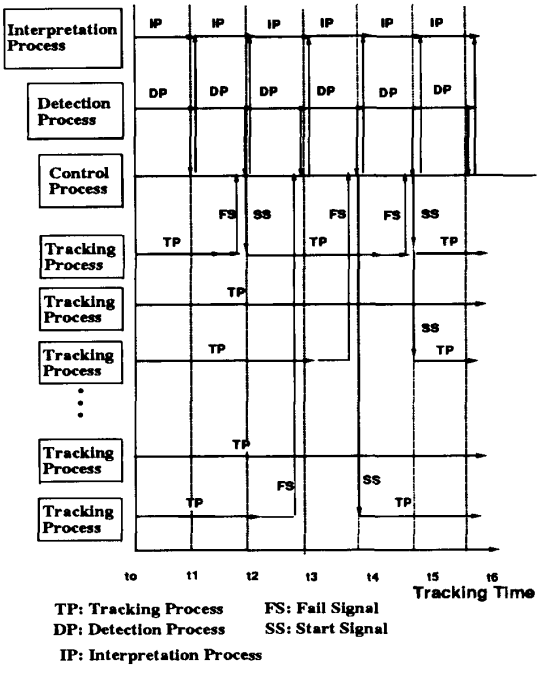

Figure 9: Sequential Flow of Process

5 Experimental Results

5.1 Preliminary Experiment on a Workstaion

First, we have carried out an experiment by simulating the VP on the workstation to confirm validity of the method. We have used 15 frame sequences in which two people cross in a passage. Figs.10(a)-(c) show the results of tracking. In the figures the rectangles represent the tracking windows. Figs.10(d)-(h) show the results of the motion interpretation. The rectangles, circles and triangles represent the tracking windows. The same marked windows indicate that they are classified as a group in the grouping process. The lines between tracking windows represent connection strengths in the grouping process. In the figures, at T=4 the grouping process is excuted, at T=5 the result of the interpreter is note complete, at T=9 several new trackers are initiated, at T=12 the result of the interpreter is complete.

5.2 Experiment in Real Time

We have carried out experiments with the VP using indoor image sequences in real time. Figs.11(a)-(c) show the sequence of images in a store where several customers are walking. In Fig.11(d) the two large white rectangle windows represent the detection windows and the small white rectangle regions represent tracking windows. Fig.11(e) shows the intermediate results of the interpretation. Fig.11(f) shows the image trajectories of people moving in the store. The elliptic region represents the last position of each person. In this experiment we limited the number of people in the scene up to five. We ran the experiment for several hours. By checking the tracking results displayed on the CRT motion as in Fig.11(f), the system was found to yield good results, although quantitative evaluation has not been made.

6 Conclusions

We have proposed a system detecting human like moving objects in time-varying images. The system consists of three subprocesses: changing region detection, moving object tracking and movement interpretation. The subprocesses are executed in parallel so that each one can recover the others' errors. We have built a trial detection system using a parallel image processing system. Experimental results of walking person detection show that the system can reliably detect the trajectories in complicated cases such as movement across difficult backgrounds.

In the current implementation, the interpreter uses only motion information to discriminate individual objects based on the motion continuity and rigid body assumptions. We are planning to add a process checking other attributes such as color and other feature properties to improve the interpretation accuracy for much more complex cases. Also, we are planning to apply the system to outdoor scenes where there are much more condition changes than indoor cases. Quantitative evaluation of the system is another issue left for future work.

References

[1] Valdis A.Dunis, "PRO-1600: The Adpro Computer Command Technical Guide", July 1987.

[2] Watch Sensor ASM-100PS, Intelligent motion detecter YS-D100 Sony Corporation.

[3] Hiroaki Kubota, Yasukazu Okamato, Hiroshi Mizoguchi, Yoshinori Kuno, "Vision processor for moving object analysis," Proc. of Computer Architecture for Machine Perception '91(CAMP'91) pp.461-470, 1991.

[4] Hsi-Jian Lee, Lung-Fa Huang, Zen Chen, "Multiframe ship detection and tracking in an infrared image sequence," Pattern Recognition, Vol.23, No.7, pp.785-798, 1990.

[5] R.D.Horton, "A traget cueing and tracking system(TCATS) for smart video processing," International Carnahan Conference on Security Technology, pp68-72, 1990.

[6] Y.Yagi, N.Mine, M.Yachida, "Finding and tracking moving objects from image sequences," Proc. of First Korea-Japan Joint Conference on Computer Vision, pp192-198, October 1991.

[7] T.J.Ellis, P.L.Rosin, P.Golton, "Model-based vision for automatic alarm interpretation," International Carnahan Conference on Security Technology, pp62-67, 1990.

[8] H.Inoue, H.Mizoguchi, "A flexible multi window vision system for robots," Proc. of Second International Symposium on Robotics Research(ISRR2), pp.95-102, 1985.

[9] M.Inaba, H.Inoue, "Observing motion by nultiple visual tracking agents," Proc. of International Workshop on Intelligent Robots and Systems '91(IROS'91),Vol.1, pp.128-133, 1991.

[10] A.L.Gilbert, M.K.Giles, G.M.Flachs, R.B.Rogers. Y.Hsun U, "A real-time video tracking system," IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.2, No.1, pp.47-56, January 1980.

[11] N.Sasaki, I.Namikawa, "The measuring system of marathon runner," Proc. of IAPR Workshop on Machine Vision Applications, pp181-184, November 1990.

[12] M.Kass, A.Witkin, D.Terzopoulos, "Snakes: Active contour models," Proc. of the 1st International Coference on Computer Vision, pp259-268, 1987.

[13] M.J.Fletcher, K.Warwick, R.j.Mitchell, "The appliaction of a hybrid tracking algorithm to motion analysis," Proc. of IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp.84-89, 1991.

[14] K.Rangrajan, M.Shah, "Establishing motion correspondence," CVGIP:Image Understanding, Vol.54, No1, pp.56-73, 1991.

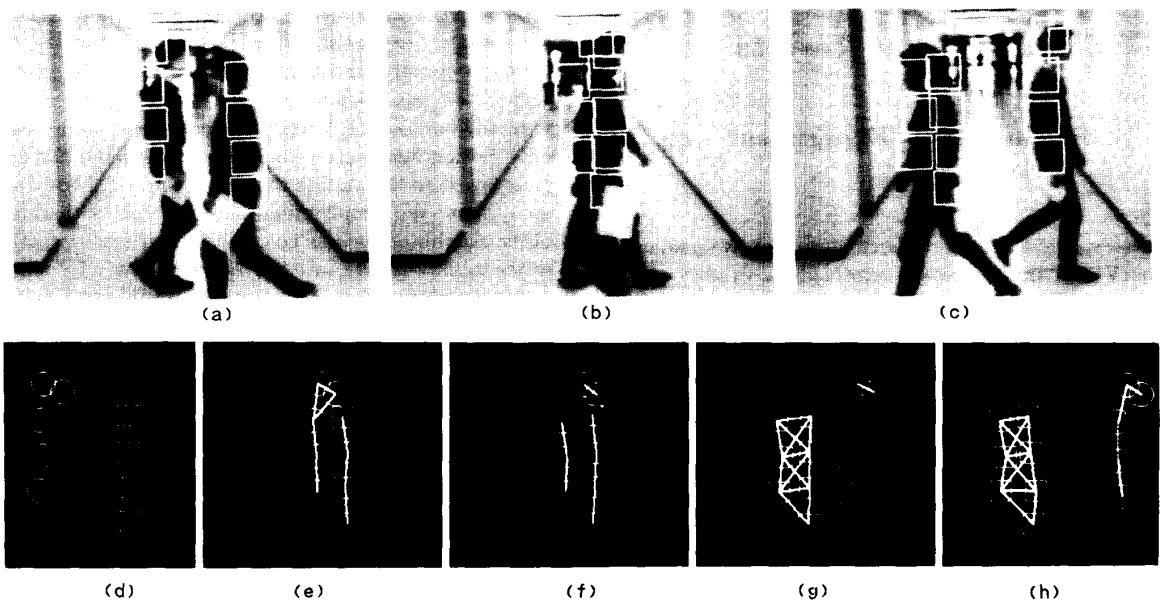

Figure 10: Preliminary Experiment on a Workstaion

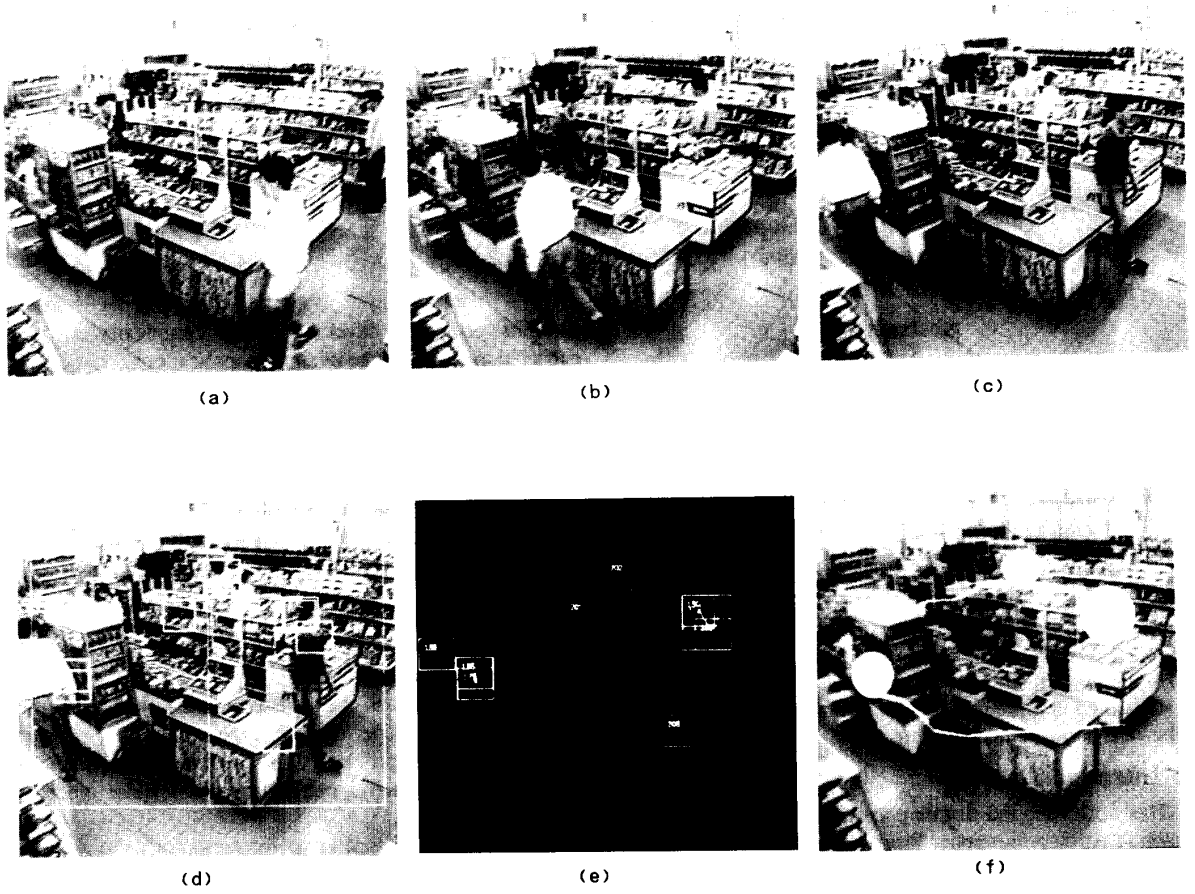

Figure 11: Experiment Results in Real Time