2007 IEEE International Conference on
Robotics and Automation
Roma, Italy, 10-14 April 2007

Multiple Objects Tracking Circuit using Particle Filters with Multiple Features

Jung Uk Cho, Seung Hun Jin, Xuan Dai Pham, and Jae Wook Jeon, *Member, IEEE*

***Abstract*—Object tracking is a challenging problem in a number of computer vision applications. A number of approaches have been proposed and implemented to track moving objects in image sequences. The particle filter, which recursively constructs the posterior probability distributions of the state space, is the most popular approach. In the particle filter, many kinds of features are used for tracking a moving object in cluttered environments. The specific feature for tracking is selected according to the type of moving object and condition of the tracking environment. Improved tracking performance is obtained by using multiple features concurrently. This paper proposes the particle filter algorithm, using multiple features, such as IFD (Inter-Frame Difference) and gray level, to track a moving object. The IFD is used to detect an object and the gray level is used to distinguish the target object from other objects. This paper designs the circuit of the proposed algorithm using VHDL (VHSIC Hardware Description Language) in an FPGA (Field Programmable Gate Array) for tracking without considerable computational cost, since the particle filter requests many computing powers to track objects in real-time. All functions of the proposed tracking system are implemented in an FPGA. A tracking system with this FPGA is implemented and the corresponding performance is measured.**

I. Introduction

THE challenge of tracking moving objects in cluttered environments has been an active research area in the computer vision community in recent years. A number of computer vision applications, such as intelligent robots [1], monitoring and surveillance [2], human computer interfaces [3], smart rooms [4], vehicle tracking [5], biomedical image analysis [6], and video compression [7], require the robust ability to tracking moving objects [8][9][10].

Numerous approaches have been proposed and implemented to track moving objects in image sequences with a camera. Particle filters, which recursively construct the posterior probability distributions of the state space, are the most popular approach, due to their robust tracking performance in cluttered environments. The basic idea of the particle filter is that the posterior density is approximated by a set of discrete samples (called particles) with associated weights [10][11]. In the particle filter, many kinds of features are used for tracking the moving object, such as image contrast, image frame difference, edge-detected silhouette, 2D or 3D contours, gray level, and RGB or HSV color. The feature for tracking is selected according to the type of moving object and the condition of the tracking environment. Improved tracking performance is obtained using multiple features concurrently [12][13].

Particle filters maintain multiple hypotheses at the same time and use a probabilistic motion model to predict the position of the moving objects. Multiple hypotheses allow for the handling of clutter in the background, and recover from failure or temporary distraction. However, expensive computational demands in this approach exist, revealing a bottleneck in the application of particle filters in real-time tracking systems [14].

In order to track a moving object in real-time without delay or loss of the image data, a particle filter algorithm specifically designed for a circuit and the circuit of the object tracking algorithm using the particle filter is proposed. This circuit is designed using VHDL and implemented in an FPGA. An object tracking system with this FPGA is implemented and the corresponding performance is measured.

This paper is organized as follows; in section II, the basic particle filter algorithm and particle filter for the circuit are explained. In section III, a circuit used for the tracking of moving object using a particle filter, is designed using VHDL, and explained in detail using a block diagram. In section IV, the tracking circuit is implemented in an FPGA and the corresponding performance is measured. Finally, the conclusion is described in section V.

II. Particle Filter

Particle filters provide robust tracking of moving objects in a cluttered environment. They are used in the case of non-linear and non-Gaussian problems where the interest lies in the detection and tracking of moving objects. This section explains the particle filter algorithm.

Particle filters use multiple discrete "particles" to represent the belief distribution over the location of a tracked object. In the Bayes filtering paradigm, the posterior distribution is recursively updated over the current state X_t given all observations $Z^t=\{Z_1,..,Z_t\}$ up to and including time t, as

Jung Uk Cho is with the School of information and Communication Engineering, the Sungkyunkwan University, Suwon, Korea (e-mail: ichead@ece.skku.ac.kr)

Seung Hun Jin is with the School of information and Communication Engineering, the Sungkyunkwan University, Suwon, Korea (e-mail: coredev@ece.skku.ac.kr)

Xuan Dai Pham is with the School of information and Communication Engineering, the Sungkyunkwan University, Suwon, Korea (e-mail: pdxuan@ece.skku.ac.kr)

Jae Wook Jeon is with the School of information and Communication Engineering, the Sungkyunkwan University, Suwon, Korea (corresponding author to provide phone: +82-31-290-7129; fax: +82-31-290-7231; e-mail: jwjeon@ yurim.skku.ac.kr).

follows:

$$p(X_t \mid Z^t) = kp(Z_t \mid X_t)p(X_t \mid Z^{t-1}) \quad (1)$$
$$= kp(Z_t \mid X_t)\int_{X_{t-1}} p(X_t \mid X_{t-1})p(X_{t-1} \mid Z^{t-1}),$$

where the likelihood $p(Z_t|X_t)$ expresses the *measurement model* and $p(X_t|X_{t-1})$ is the *motion model*. In a particle filter, the posterior $p(X_{t-1}|Z^{t-1})$ is recursively approximated as a set of N weighted samples $\{X_{t-1}^{(i)}, w_{t-1}^{(i)}\}_{i=1}^{N}$, where $w_{t-1}^{(i)}$ is the weight for particle $X_{t-1}^{(i)}$. Given this equation, a Monte Carlo approximation of the integral can be used and the following can be obtained:

$$p(X_t \mid Z^t) \approx kp(Z_t \mid X_t)\sum_{i=1}^{N} w_{t-1}^{(i)} p(X_t \mid X_{t-1}^{(i)}). \quad (2)$$

One way to view a particle filter is as an importance sampler for this posterior distribution. Specifically, N samples $X_t^{(s)}$ are drawn from the *proposal distribution*

$$q(X_t) = \sum_{i=1}^{N} w_{t-1}^{(i)} p(X_t \mid X_{t-1}^{(i)}), \quad (3)$$

and then weighted by the likelihood, i.e.

$$w_t^{(s)} = p(Z_t \mid X_t^{(s)}). \quad (4)$$

This results in a weighted particle approximation $\{X_t^{(s)}, w_t^{(s)}\}_{s=1}^{N}$ for the posterior $p(X_t|Z^t)$ at time t [11][15].

The general operation of the particle filter tracker for the tracking of objects in video images is illustrated in Fig. 1. Figure 2 shows a flowchart of the general operation of the particle filter. Each particle which tracks moving objects is represented by a point, so the group of particles is referred to as a point cloud. The principal steps in the particle filter algorithm include: [10][16]

1) Initialization

   Draw a set of particle for the prior $p(X_0)$ to obtain $\{X_0^{(i)}, w_0^{(i)}\}_{i=1}^{N}$, let k=1.

2) Sampling

   (a) For i=1,…,N

   Sample $X_k^{(i)}$ from the proposal distribution $p(X_k^{(i)} \mid X_{k-1}^{(i)})$.

   (b) Evaluate new weights

   $$w_k^{(i)} = p(Z_k \mid X_k^{(i)}), \quad i=1,\ldots,N \quad (5)$$

   (c) Normalize weights

   $$w_k^{(i)} = \frac{w_k^{(i)}}{\sum_{j=1}^{N} w_k^{(j)}}, \quad i=1,\ldots,N \quad (6)$$

3) Output

   Output a set of particles $\{X_k^{(i)}, w_k^{(i)}\}_{i=1}^{N}$ that can be used to approximate the posterior distribution as $p(X_k \mid Z^k) \approx \sum_{i=1}^{N} w_k^{(i)} \delta(X_k - X_k^{(i)})$ and the estimate as $E_{p(\cdot|Z^k)}(f_k(X_k)) \approx \sum_{i=1}^{N} w_k^{(i)} f_k(X_k^{(i)})$, where $\delta(\cdot)$ is the Dirac delta function.

4) Resampling

   Resample particles $X_k^{(i)}$ with probability $w_t^{(i)}$ to obtain N independent and identically distributed random particles $X_k^{(j)}$, approximately distributed according to $P(X_k|Z^k)$.

k=k+1, go to step2.

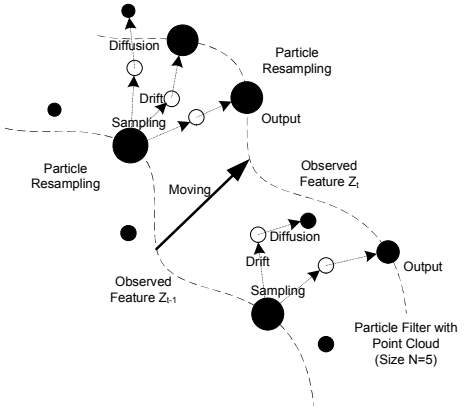


Fig. 1. Visualization of the particle filter.

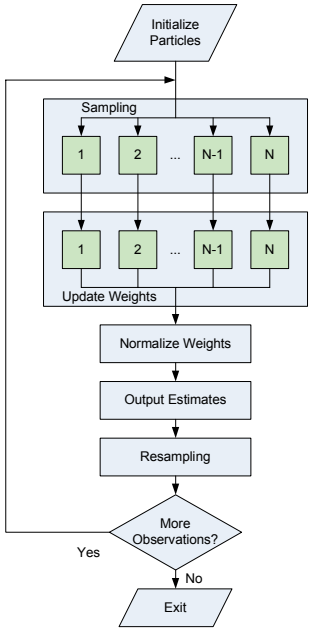


Fig. 2. Flowchart of the particle filter.

III. Circuit Used for Object Tracking

The observations made of the surveillance area after the detection of the objects by the threshold method are used as important features to track the moving object. For object tracking, this paper proposes the particle filter algorithm using multiple features as IFD and gray level. The IFD is used to detect the object and the gray level is used to distinguish the specific object from others. The tracking performance is improved using multiple features concurrently. Due to their

sample-based representation, particle filters have been applied with success to a variety of state estimation problems, including visual tracking. The increased representational power of particle filters, however, comes at the expense of higher computational complexity.

In previous particle filter applications, the prevalent approach in real-time applications was to update the filter as often as possible and to discard the image data that arrives during the update process. This approach is prone to losing valuable image data. At first sight, the sample based representation of particle filters appears to offer an alternative approach, similar to anytime implementation: i.e. whenever a new observation arrives, sampling is interrupted and the next observation is processed. Unfortunately, such an approach can result in too small sample sets, thus causing the filter to diverge [17]. Although the real-time tracking of a moving object is possible using a particle filter, this approach results in the majority of system resources being monopolized by the tracking process.

In order to solve the problem posed by the real-time implementation of object tracking, a specialized circuit is designed using the proposed algorithm. It is impossible for the proposed algorithm to be applied to the hardware circuit without some modification, due to the restrictions imposed by the hardware, such as the timing, resources, size, and power consumption. Therefore, a modified architecture is proposed for circuit implementation of the particle filter. Figure 3 shows the circuit of the object tracking system, which consists of seven modules: the Camera Controller, Image Store Memory, Image Subtractor, Gray Level Comparator, Output, and two Particle Filters. It is designed using VHDL and implemented in an FPGA, in order to track moving objects in real-time. In Fig. 3, the gray box, consisting of the Image Subtractor, Gray Level Comparator, Output and two Particle Filter modules, represent the implemented circuit in an FPGA.

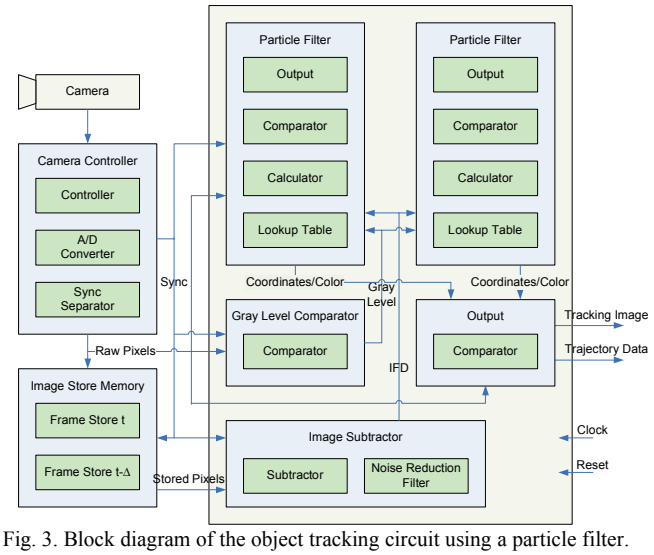


Fig. 3. Block diagram of the object tracking circuit using a particle filter.

A. Camera Controller

In the Camera Controller module, the Sync Separator generates the signals used for controlling the camera and the A/D Converter converts the analog image data into digital image data. The image sync signals, *Sync*, and digital image data, *Raw Pixels*, of the Camera Controller module are used in all modules of the object tracking circuit.

B. Image Store Memory

Based on the sync signals, *Sync*, obtained from the Sync Separator in the Camera Controller module, the Image Store Memory module stores the image data, *Raw Pixels*, arriving from the A/D Converter, in the Camera Controller module frame by frame. This module transfers the image data of the previous and current frames, *Stored Pixels*, to the Image Subtractor module, in order to generate the IFDs, *IFD*, for Particle Filter module. The FIFO (First In, Fist Out) memories are used to store the image of a frame, since it is difficult and inefficient to store the image of a frame in an FPGA.

C. Image Subtractor

The Image Subtractor module generates the IFDs, *IFD*, using pixel by pixel differencing between the images of the previous and current frames, *Stored Pixels*, is obtained from the Image Store Memory module. These IFDs, *IFD*, $\Delta P_t(x,y)$ are determined as

$$\Delta P_t(x,y) = \left| I_t(x,y) - I_{t-\Delta}(x,y) \right|, \tag{7}$$

where $I_t(x,y)$ is the value at the $(x,y)^{th}$ pixel that has the range [0, 255] at time t, and $I_{t-\Delta}(x,y)$ is the value at the $(x,y)^{th}$ pixel that has the range [0, 255] at time t-Δ. The range [0, 255] is the same as that of a gray level image. The IFDs, *IFD*, are used in the Particle Filter module as a tracking feature after the noise is reduced in the Noise Reduction Filter block.

IFDs often contain noise, resulting in errors during image processing, a Noise Reduction Filter block is included to reduce this noise by applying a threshold. The noise reduction technique described by

$$\Delta P_t(x,y) = \begin{cases} \Delta P_t(x,y), & \Delta P_t(x,y) > V_{TH} \\ 0, & \Delta P_t(x,y) \le V_{TH} \end{cases}, \tag{8}$$

where, V_{TH} is a threshold value, which is set in advance, to help reduce the noise.

D. Gray Level Comparator

The Gray Level Comparator module compares the gray level of the image data, *Raw Pixels*, arriving from the A/D Converter in the Camera Controller module with the reference gray level set in advance to distinguish the specific object with the specific gray level. This module transfers the valid signal and gray level, *Gray Level*, $V_t(x,y)$ determined as

$$V_t(x,y) = \begin{cases} 1, & I_L \le I_t(x,y) < I_H \\ 0, & I_t(x,y) < I_L \text{ or } I_H \le I_t(x,y) \end{cases}. \tag{9}$$

The value, $I_t(x,y)$, at the $(x,y)^{th}$ pixel that has the range [0, 255] at time t, is included in the range $[I_L, I_H]$, the value, $I_t(x,y)$,

at the $(x,y)^{th}$ pixel is valid and to be used in the Particle Filter module as a tracking feature. Otherwise, the value, $I_t(x,y)$, at the $(x,y)^{th}$ pixel is invalid for use as a Particle Filter module. The range [0, 255] is the same as that of a gray level image.

E. Output

The Output module receives the trajectory, *Coordinates*, and gray level, *Color*, obtained from Particle Filter modules, and then outputs the trajectory data and trajectory images. This module enables to display these trajectory images to the monitor or LCD directly.

F. Particle Filter

The Particle Filter module in the proposed object tracking circuit is based on the technique described in section 2. The inputs, *IFD*, *Valid*, and *Sync*, represent the IFDs from the Image Subtractor module, the valid signal from the Gray Level Comparator module, and the image sync signals from the Camera Controller module, respectively. The outputs, *Tracking Image* and *Trajectory Data*, represent the tracking image data and the trajectory data of the tracking object, respectively. Therefore, this circuit can obtain the tracking image and trajectory information of the tracked object in real-time.

Figure 4 shows the operation of the particle filter circuit. The principal steps in the particle filter circuit include:

1) Initialization
   Draw a set of particles uniformly.
2) Sampling
   (a) Sample the position of the particles from the proposal distribution.
   (b) Find the features of the moving object (IFD, Gray Level).
   (c) Update the weight of the particles.
   (d) Normalize the weight of the particles.
3) Output
   Output the mean position of the particles that can be used to approximate the posterior distribution.
4) Resampling
   Resample the particles with probability to obtain independent and identically distributed random particles.
5) Go to the sampling step.

Figure 5 shows the architecture of the Particle Filter module. It consists of six blocks: the Particle Initiator, Particle Sampler, Coordinates Comparator, Particle Normalizer, Data Output, and Particle Selector. The Particle Initiator block initializes the particles when it starts to track objects for the first time or when it attempts to find a new moving object after the previously tracked object has disappeared from view. The Particle Sampler block performs the sampling step of the particle filter algorithm. The Coordinates Comparator block is used to update the weights of the particles. The Particle Normalizer block normalizes the weights of the particles. The Data Output block calculates the mean position of the particles. The Particle Selector block performs the resampling step in the particle filter algorithm.

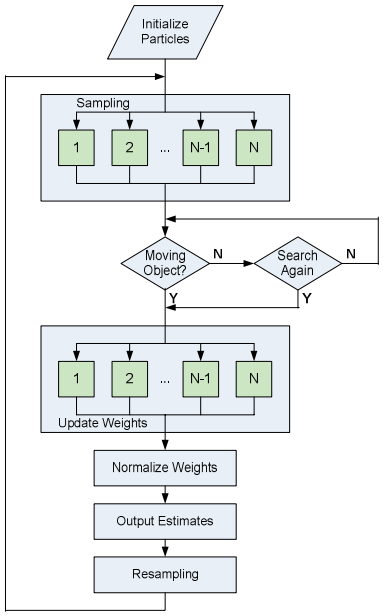


Fig. 4. Flowchart of the particle filter circuit.

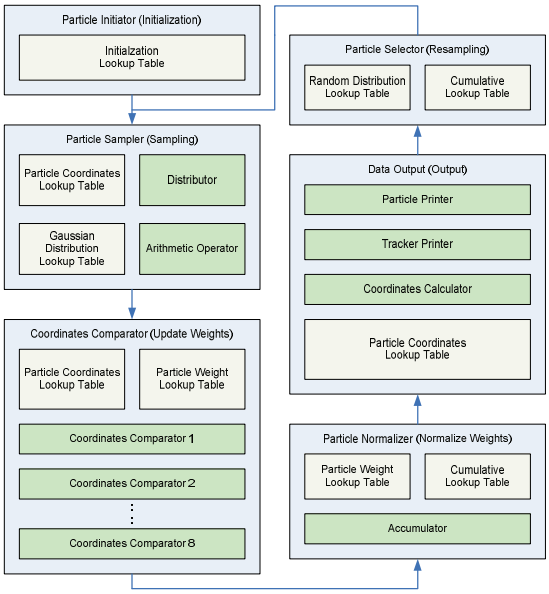


Fig. 5. Block diagram of the particle filter module.

IV. Experiment

The circuit of the object tracking algorithm using the particle filter is designed using VHDL and is implemented in a XC2V8000-4CFF1152 FPGA, with 8M system gates, 93,184 logic cells, and 1,104 I/O pins [18]. Table I presents a summary of the device utilization characteristics for the circuit of the object tracking algorithm using the particle filter. Figure 6 shows a vision system using this FPGA. This FPGA can acquire images from a camera and display both the raw and processed images and the object trajectory data generated by the particle filter to monitor or LCD directly.

A high frame processing rate and low latency are important for many applications that must provide quick decisions based

TABLE I
DEVICE UTILIZATION CHARACTERISTICS FOR THE OBJECT TRACKING CIRCUIT CONSISTING OF PARTICLE FILTERS

DEVICE UTILIZATION CHARACTERISTICS				
Number of Slices:	16776	out of	46592	36%
Number of Slice Flip Flops:	16007	out of	93184	17%
Number of 4 input LUTs:	20385	out of	93184	21%
Number of bonded IOBs:	126	out of	824	15%
Number of MULT18X18s:	2	out of	168	1%
Number of GCLKs:	3	out of	16	18%

Fig. 6. A vision system employing a multiple objects tracking circuit using particle filters with multiple features.

on events in the scene [19]. When the vision system with the circuit of the proposed object tracking using particle filter is applied to an RS-170 camera, which produces images consisting of 640×480 pixels, it can process the images at speeds of up to 56.38 fps (frames per second) to track a moving object and display these processed images to the monitor or LCD. The performance of this circuit is measured by evaluating the average maximum delay time.

Figure 7 and 8 show the results obtained from the tracking of moving objects in a video sequence using the multiple objects tracking circuit. There are some delays between the images shown in Fig. 7.(a) - (f) and Fig. 8.(a) - (f). In Fig. 7, the white points represent the particles to track target object and the white square with the white cross in the center represents the trajectory of the target object. The single object tracking uses only IFD feature to track a moving object. In Fig. 8, moving objects have quit different gray level each other. The multiple objects tracking uses multiple features as IFD and gray level to track multiple objects respectively. In Fig. 8, the black square with the black cross presents the trajectory of the target object that has the specific gray level included in the range [0, 127]. Otherwise, the white square with the white cross represents the trajectory of the target object that has the specific gray level included in the range [128, 255]. Even although there is an occlusion, the tracking specific object is not affected by another in Fig. 8. A set of particles consists of 64 particles. A particle set tracks a target object. The trajectory of the target object is estimated by the mean of the position of particles in a particle set.

V. CONCLUSION

In this paper, a real-time multiple objects tracking circuit is developed to track moving objects. This circuit is designed using VHDL, and implemented in an FPGA, and helps to make smaller, low cost, and low power systems. It is easy to modify and update this circuit as required. In order to implement the objects tracking system, the only additional requirement is an ordinary video camera. No additional systems are needed to implement the objects tracking system. This tracking system solves the expensive computation problem caused by the particle filter. Only a small percentage of the system resources are allocated for objects tracking, the remainder of the resources can be assigned to preprocessing stages or to high level tasks such as recognition, trajectory interpretation, and reasoning. This paper demonstrates that this real-time objects tracking circuit combined with other technologies can produce effective and powerful applications

ACKNOWLEDGMENT

This research was performed for 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] C. Kwok, D. Fox, M. Meila, "Adaptive Real-Time Particle Filter for Robot Localization," *Proceedings of Robotics and Automation*, Vol. 2, pp. 2836-2841, Sept. 2003.

[2] Y. Cui, S. Samarasekera, Q. Huang, M. Greiffenhangen, "Indoor Monitoring Via the Collaboration Between a Peripheral Sensor and a Foveal Sensor," *Proceedings of Visual Surveillance*, pp. 2-9, Jan. 1998.

[3] G. R. Bradski, "Real Time Face and Object Tracking as a Component of a Perceptual User Interface," *IEEE workshop on Applications of Computer Vision*, Princeton, pp. 214-219, 1998.

[4] C. R. Wren, A. Azarbayejani, T. Darrel, A. P. Pentland, "Pfinder: Real-Time Tracking of the Human Body," *IEEE Transactions on Pattern Analysis and Machine intelligence*, Vol. 19, pp. 780-785, July 1997.

[5] M. Betke, E. Haritaoglu, S. S. Davis, "Multiple Vehicle Detection and Tracking in Hard Real-Time," *Proceedings of Intelligent Vehicles Symposium*, pp. 351-356, Sept. 1996.

[6] L. Pan, J. L. Prince, J. A. C. Lima, N. F. Osman, "Fast Tracking of Cardiac Motion Using 3D-HARP," *IEEE Transactions on Biomedical Engineering*, Vol. 52, pp. 1425-1435, August 2005.

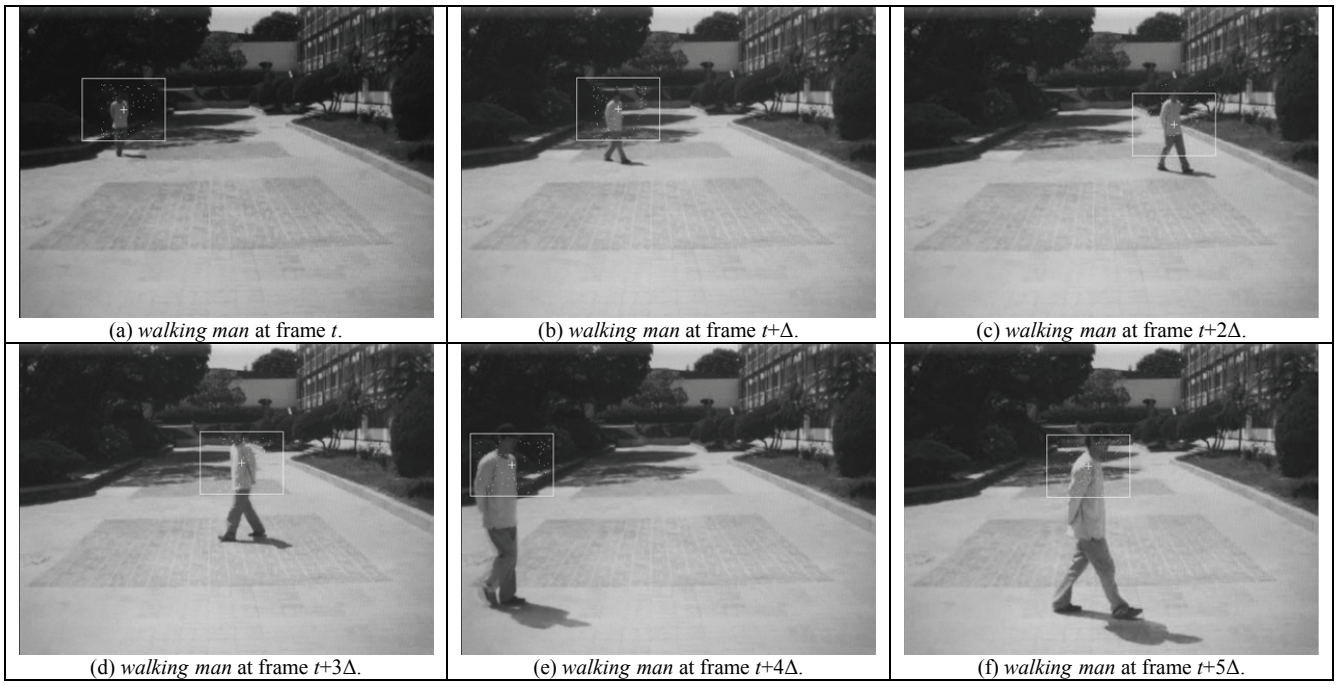
[7] A. Eleftheriadis, A. Jacquin, "Automatic Face Location Detection and Tracking for Model-Assisted Coding of Video Teleconference Sequence at Low Bit Rates," *Signal Processing – Image Communication*, Vol. 7, pp. 231-248, 1995.

[8] A. Blake, M. Isard, *Active Contours*. Springer-Verlag, New York, 1998.

[9] D. Comaniciu, V. Ramesh, P. Meer, "Real-Time Tracking of Non-Rigid Objects Using Mean Shift," *IEEE Conference on Computer Vision and Pattern Recognition*, South Carolina, Vol. 2, pp. 142-149, 2000

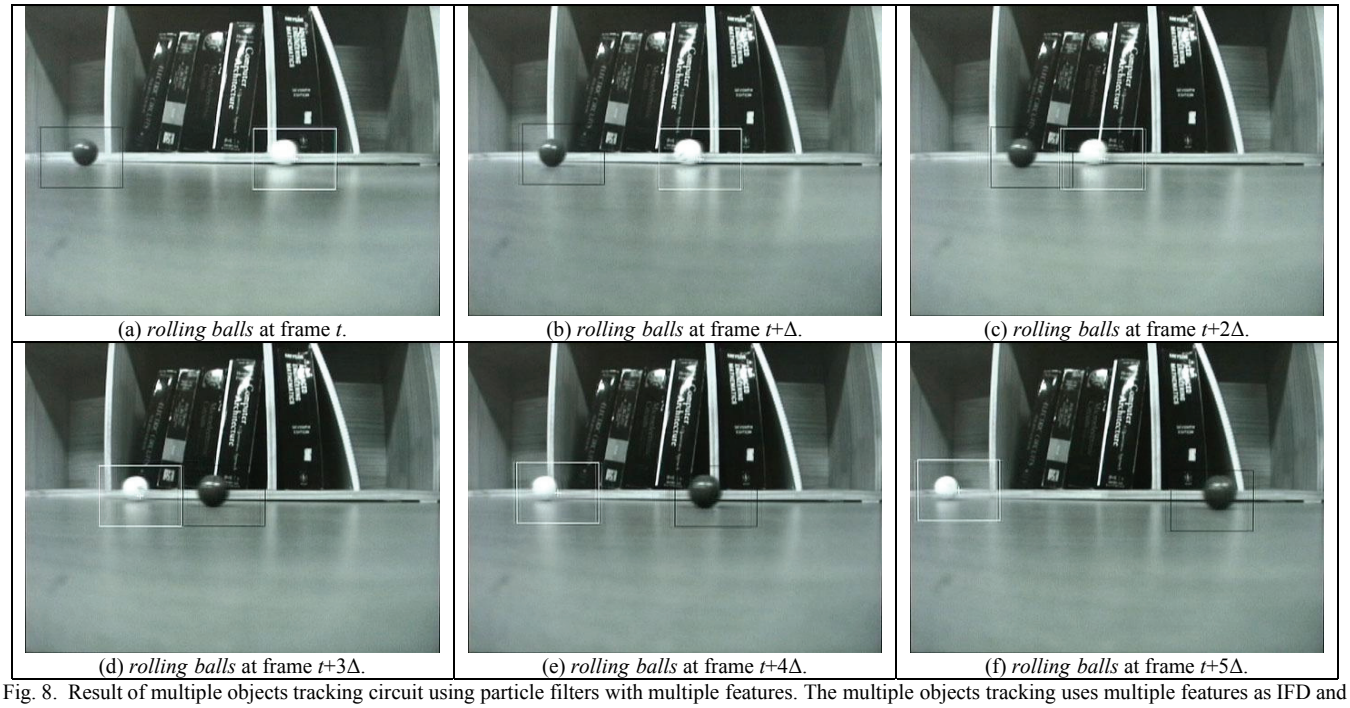
[10] P. Li, T. Zhang, "Visual Contour Tracking Based on Particle Filters," *Proceedings of Generative-Model- Based Vision*, Copenhagen, pp. 61-70, June 2002.

[11] B. Ristic, S. Arulampalam, N. Gordon, *Beyond The Kalman Filter. Boston•London*, Artech House, 2004.
[12] H. Veeraraghavan, P. Schrater, N. Papanikolopoulos, "Robust target detection and tracking through integration of motion, color, and geometry," *Computer Vision and Image Understanding*, Vol. 103, Issue 2, pp. 121-138, August 2006.
[13] P. Perez, C. Hue, J. Vermaak, M. Gangnet, "Color-Based Probabilistic Tracking," *Lecture Notes In Computer Science*; Vol. 2350, pp. 661-675, Springer-Verlag, Berlin Heidelberg, 2002.
[14] C. Shan, Y. Wei, T. Tan, F. Ojardias, "Real Time Hand Tracking by Combining Particle Filtering and Mean Shift," *Proceedings of Automatic Face and Gesture Recognition*, pp. 669-674, May 2004.
[15] C. Hue, J. L. Cadre, P. Perez, "A Particle Filter to Track Multiple Objects," *IEEE Workshop on Multi-Object Tracking*, pp. 61-68, July 2001.
[16] C. Hue, J. L. Cadre, P. Perez, "Sequential Monte Carlo Methods for Multiple Target Tracking and Data Fusion," *IEEE Transactions on Signal Processing*, Vol. 50, No. 2, pp. 309-325, Feb. 2002.
[17] C. Kwok, D. Fox, M. Meila, "Real-Time Particle Filters," *Proceedings of the IEEE*, Vol. 92, Issue 3, pp. 469-484, March 2004.
[18] Xilinx Inc., "Virtex-II Platform FPGAs: Complete Data Sheet," available from www.xilinx.com, March 2005.
[19] J. I. Woodfill, G. Gordon, R. Buck, "Tyzx DeepSea High Speed Stereo Vision System," *Conference on Computer Vision and Pattern Recognition*, pp. 41-45, June 2004.



(a) *walking man* at frame *t*. (b) *walking man* at frame *t*+Δ. (c) *walking man* at frame *t*+2Δ.
(d) *walking man* at frame *t*+3Δ. (e) *walking man* at frame *t*+4Δ. (f) *walking man* at frame *t*+5Δ.

Fig. 7. Result of the object tracking circuit using a particle filter. The single object tracking uses only IFD feature to track a moving object.



(a) *rolling balls* at frame *t*. (b) *rolling balls* at frame *t*+Δ. (c) *rolling balls* at frame *t*+2Δ.
(d) *rolling balls* at frame *t*+3Δ. (e) *rolling balls* at frame *t*+4Δ. (f) *rolling balls* at frame *t*+5Δ.

Fig. 8. Result of multiple objects tracking circuit using particle filters with multiple features. The multiple objects tracking uses multiple features as IFD and gray level to track multiple objects respectively. One has the specific gray level including in the range [0, 127], the other has the specific gray level including in the range [128, 255].