Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 144 (2018) 2–12

Procedia Computer Science

www.elsevier.com/locate/procedia

INNS Conference on Big Data and Deep Learning 2018

Online video streaming for human tracking based on weighted resampling particle filter

Mukesh Prasad[a,*], Liang-Cheng Chang[b], Deepak Gupta[c], Mahardhika Pratama[d], Suresh Sundaram[d], Chin-Teng Lin[a]

[a]*Centre for Artificial Intelligence, School of Software, FEIT, University of Technology Sydney, Australia*
[b]*Department of Computer Science and Engineering, National Chiao Tung University, Hsinchu, Taiwan*
[c]*Department of Computer Science, National Institute of Technology, Arunachal Pradesh, India*
[d]*School of Computer Science and Engineering, Nanyang Technological University, Singapore*

Abstract

This paper proposes a weighted resampling method for particle filter which is applied for human tracking on active camera. The proposed system consists of three major parts which are human detection, human tracking, and camera control. The codebook matching algorithm is used for extracting human region in human detection system, and the particle filter algorithm estimates the position of the human in every input image. The proposed system in this paper selects the particles with highly weighted value in resampling, because it provides higher accurate tracking features. Moreover, a proportional–integral–derivative controller (PID controller) controls the active camera by minimizing difference between center of image and the position of object obtained from particle filter. The proposed system also converts the position difference into pan-tilt speed to drive the active camera and keep the human in the field of view (FOV) camera. The intensity of image changes overtime while tracking human therefore the proposed system uses the Gaussian mixture model (GMM) to update the human feature model. As regards, the temporal occlusion problem is solved by feature similarity and the resampling particles. Also, the particle filter estimates the position of human in every input frames, thus the active camera drives smoothly. The robustness of the accurate tracking of the proposed system can be seen in the experimental results.

© 2018 The Authors. Published by Elsevier Ltd.
This is an open access article under the CC BY-NC-ND license (https://creativecommons.org/licenses/by-nc-nd/4.0/)
Selection and peer-review under responsibility of the INNS Conference on Big Data and Deep Learning 2018.

Keywords: human tracking; particle filter; color distribution; PID controller; codebook matching

* Corresponding author. Tel.: +61-2-9514-4426.
E-mail address: mukesh.nctu@gmail.com

1877-0509 © 2018 The Authors. Published by Elsevier Ltd.
This is an open access article under the CC BY-NC-ND license (https://creativecommons.org/licenses/by-nc-nd/4.0/)
Selection and peer-review under responsibility of the INNS Conference on Big Data and Deep Learning 2018.
10.1016/j.procs.2018.10.499

1. Introduction

In recent years, visual-based detection and tracking systems have been applied for security surveillance to improve safety and convenience of human's life. The traditional visual-based surveillance system uses human eyes to check monitor all day. This is inefficient and time consuming. In order to solve this problem, automatic and real-time vision based surveillance system is proposed in this paper. In surveillance system, human detection and tracking are important topics. The human detection system has two parts: moving object extraction and human recognition. The moving object extraction extracts object from background and find its related position and size in an image. Human recognition system recognizes object as human or nonhuman. Then, the tracking system tracks the human in continuous frame. The human may be occluded with other objects while tracking. So the tracking system must able to predict the position during and after occlusion.

There are two kinds of camera used in surveillance system, which are fixed and active camera. The advantage of a fixed camera is low cost, but its FOV (field of view) is limited. On the other hand, active camera has good FOV because it has the ability to perform pan-tilt to keep target human in the camera scene. Also, it has good resolution because able to do zoom in/out. This paper integrates human detection, human tracking, and active camera controller to achieve automatic and real-time surveillance systems. Human tracking is used to following target human through the sequence images in terms of changes in scale and position. Fig. 1 shows three based tracking method. First, feature based is the most commonly method, and color, edge, or motion is used as tracking feature. The edge detection method, such as Sobel method [1], Laplacian method [1], and Marr Hildreth method [2], etc., utilize masks to do convolution on the image to detect the edges. Wei Guo et al. [3] proposed human tracking system based on shape analysis. Law et al. [4] designed fuzzy rules in edge based human tracking. This method requires a large and complicated rules set, also needs more computation time, and edge pixels cannot be always detected continuously. Most of the proposed methods mentioned above detect edges using gray level images, and neglect for color images because the representation of a pixel is not only a gray level but a vector in a color space. The edge occurring in the adjacent pixels which have the same values may not be detected. So, edge detection only in gray level image is not sufficient and robust.

Pattern recognition learns the target object and search them in sequence image. Williams et al. [5] extended the approach to the nonlinear translation predictors which learned by Relevance Vector Machine. Agarwal and Triggs [6] used RVM to learn the linear and nonlinear mapping for tracking of 3D human poses from silhouettes. Bohyung Han and Larry Davis [7] used PCA to extract feature from color and used these feature in mean-shift tracking algorithm. Robert T. et al. [8] presented an online feature selection mechanism for evaluating multiple features while tracking and adjusting the set of features to improving tracking performance. The feature evaluation mechanism is embedded in a mean-shift tracking system. It can adaptively select features for tracking. The mean-shift algorithm is originally proposed by Fukunaga and Hostetler [9] for clustering data. The kernel-based object tracking is proposed by Meer et al [10], which tracks an object region represented by a spatial weighted intensity histogram, and computed its similarity value by Bhattacharyya distance using iterative mean-shift algorithm. Later, many variants of the mean-shift algorithm are proposed for various applications [11-14]. Although the mean-shift object tracking algorithm performs well on sequences with relatively small object displacement, however its performance is not guaranteed when objects undergo partially or fully occlusion. In order to improve the performance of mean-shift tracker under partial occlusion, there are some algorithms added to the mean-shift such as Kalman filter [15-16] or particle filter [17-18]. K. Nummiaro et al [18] used the idea of particle filter to apply a recursive Bayesian filter based on sample sets, where color is used as feature. Their work evolved from the condensation algorithm which is developed in the computer vision community. In this paper, the proposed system selects the particle filter to track human, because it has proven very successful for non-linear and non-Gaussian estimation problems.

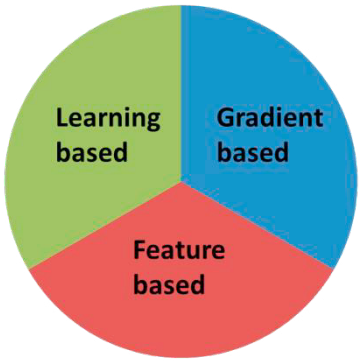


Fig. 1. Three categories of tracking

2. Proposed Human Tracking System

The proposed human tracking system consists four major parts: Image source, Human detection, Human tracking, and Camera control. The input frames are captured by PTZ camera with resolution 720*480. The initial FOV is the scene which need to be monitored and moving object is extracted by background difference. The codebook matching is applied to classify the moving object into human or nonhuman. When a human is detected in initial FOV, it is regards as a target. The Particle filter method tracks the target in each frame and send the target position and size to camera control. The occlusion handler uses the similarity measure value in each frame to solve the temporal full occlusion that sometimes misleads tracking in particle filter. The Target position is sent to PID controller to determine pan-tilt speed and the size is used to decide zoom-in or zoom-out. The PTZ cmd drives PTZ to keep the target in the center of FOV. When PTZ camera is driven, it changes FOV and the human detection is skipped during camera tracking. The diagram of the proposed human tracking system is shown in Fig. 2.

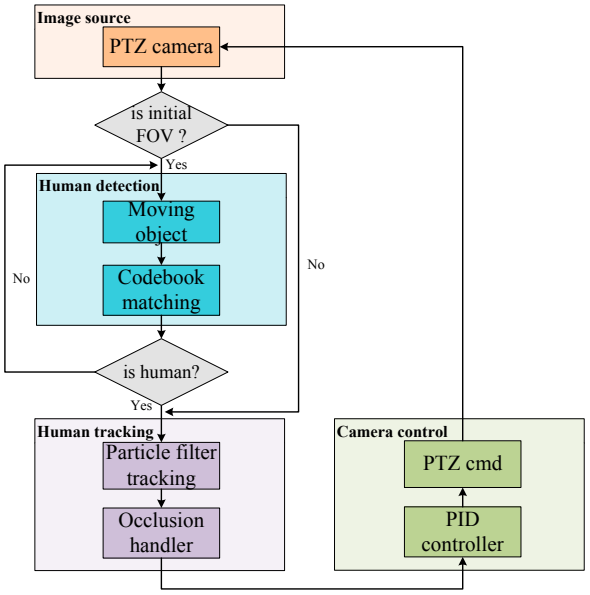


Fig. 2. Proposed system overview

2.1. Human Detection

In surveillance systems, the default position of the camera is mostly fixed, no matter if it is using an active camera or static camera. Because of this, we can use background subtraction to extract moving objects. Our background subtraction only uses the gray level image, so it can decrease the computer power and achieve real-time requirements. With the first image frame adapted over time using Eq. (1), we constructed the background, where I_B^n and I_B^{n-1} represent respectively current and previous background image.

$$I_B^n(x,y) = \begin{cases} \alpha * I_B^{n-1}(x,y) + (1-\alpha) * I_c(x,y), & I_M(x,y) = 0 \\ I_B^{n-1}(x,y), & I_M(x,y) = 1 \end{cases} \quad (1)$$

The background image was updated by scaling factor α(0,1). I_M (x,y) represents the active pixel between both frames. We calculate the difference between current image I_c and background image I_B, as Eq. (2), to determine the moving object, and apply a threshold ths to produce a binary moving object M_obj after all, as described in Eq. (3).

$$I_{BS}(x,y) = |I_c(x,y) - I_B(x,y)| \quad (2)$$

$$M_{obj}(x,y) = \begin{cases} 1, & I_{BS} \geq ths \\ 0, & I_{BS} < ths \end{cases} \quad (3)$$

A dilation process was applied on M_{obj} to enlarge the boundaries and fill the moving objects holes. Our codebook matching algorithm was based on the human-shape information. At first, we normalized the extracted moving object into a 20*40 pixels image. The shape feature extraction extracted the position of the shape pixels in the image, as pointed by the red dots in Fig. 3. We chose Ten Y-axis coordinates for the leftmost and rightmost of the object's boundary. Then, the twenty coordinates of its related X-axis coordinates were arranged as a feature vector as shown as the blue blocks in Fig. 3. We have a total of ten bin in the histogram, as shown as green blocks in Fig. 3. Therefore, a 30-feature vector could represent a human object.

By observation, the top and bottom shape pixels Y-axis were not suitable to choose as feature points, because these pixels are changeable. The way we found these ten specific coordinates at Y-axis was calculating the standard deviation in each value of Y-axis in a training sample, and then get the ten lowest standard deviation values for each side. The codebook represents a list of feature vectors. The feature vector was matched with the vectors in our codebook with the purpose of finding a code vector with the minimum distortion by comparison to the object feature vector. Let a series of features vector denotes as X, and each of X includes M-dimensional data, indicated by $X^0 \dots X^i \dots X^{(M-1)}$. There are N sets of code words V defined as $V_0 \dots V_j \dots V_{(N-1)}$ in codebook C. Each of V_j is just like X that has M-dimensional data defined as $V_j^0 \dots V_j^i \dots V_j^{(M-1)}$. Eq. 4 defines the distortion between feature word and code words.

$$Dis_j = \|X - V_j\| = \sum_{i=0}^{M-1} |X^i - V_j^i| \quad (4)$$

$$Dis_{min} = min(Dis_j) \; j = 0 \dots N-1 \quad (5)$$

If the value of Dis_{min} was smaller than our threshold, we considered the feature word X, or the moving object its represent, as a human. Otherwise, we would consider it as a non-human object.

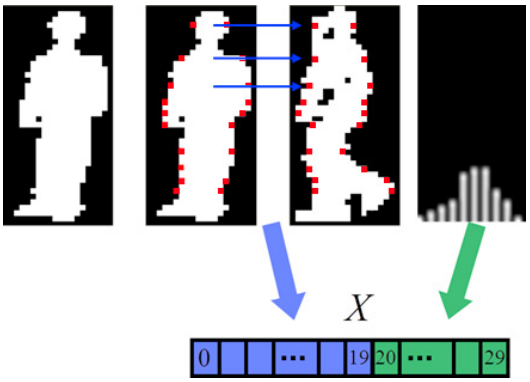
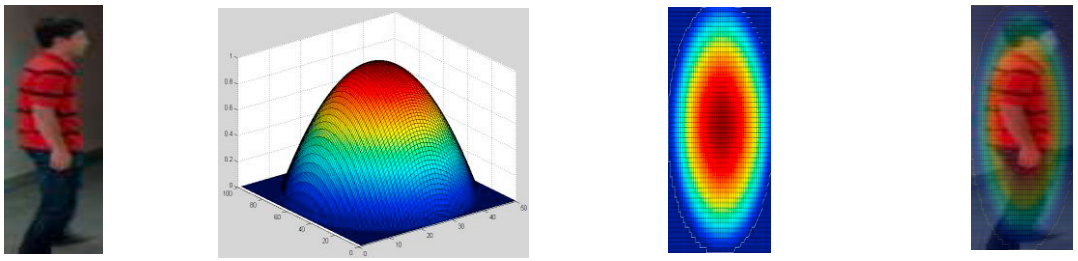


Fig. 3. Example of a feature vector *X*

2.2. Human Tracking

We proposed a particle filter algorithm with a weighted resampling particles method, which just selects high weighted samples, for a human tracking system. The key idea of a particle filter is to approximate the probability distribution by a weighted sample set where each sample represents one hypothetical state of the object with a corresponding discrete sampling probability. When we use color as a feature in a feature-based object racking, colored information is more accurate than grayscale. We chose HSV color space, which gives more performance while tracking than RGB color space, as it reduces illumination or lightness sensitivity. Each color-channel has 8-bits data, which means it produces a (256*256*256) of bins of the color histogram. Without generality loss, the color-data is quantizing into (6*6*6). Therefore, the whole bin of the color histogram is 216 bins. The kernel function was used to represent a target object. Eq. (6) define the Epanechnikov kernel, where x is the normalized pixels in the region defined as our target model. When our kernel function is applied target model, the pixels which are closer to the ROI center contain more critical information, as we can see in Fig 4.

$$k(x) = \begin{cases} \frac{3}{4}(1 - x^2) & if\ \|x\| \leq 1 \\ 0 & otherwise \end{cases} \tag{6}$$



(a) (b) (c) (d)

Fig. 4 (a) Target object (b) Kernel Function (c) Kernel Function (d) Target object and Kernel Function

The particle filter algorithms provide a robust tracking framework, as it models uncertainty. It can keep its options open and consider multiple state hypotheses simultaneously. Since less likely object states have a chance to remain in the tracking process temporarily, particle filters can deal with temporary occlusions. The differences between our proposed tracking method and the original one is the weighted resampling and occlusion handler steps. Define a target model at location y as m-bin histogram $q_y = \{q_y^{(u)}\}_{u=1\ldots m}$ which compute by Eq. (7) with the normalized factor as Eq. (8) where I denotes the number of pixels in the ROI region, δ is the Kronecker delta function, and $a = \sqrt{w^2 + h^2}$ is used to normalize the size of the object region.

$$q_y^{(u)} = f \sum_{i=1}^{I} k\left(\frac{\|y-x_i\|}{a}\right) \delta[h(x_i) - u] \quad (7)$$

$$f = \frac{1}{\sum_{i=1}^{I} k(\frac{\|y-x_i\|}{a})} \quad (8)$$

The sample model $p_y = \{p_y^{(u)}\}_{u=1\ldots m}$ is represented as the same model as the target model. The similarity value ρ between target and sample model computes by Bhattacharyya distance d. A large ρ means that two models are more similar, where ρ equal to 1 means that two histograms are identical.

$$p_y^{(u)} = f \sum_{i=1}^{I} k\left(\frac{\|y-x_i\|}{a}\right) \delta[h(x_i) - u] \quad (9)$$

$$\rho[p,q] = \sum_{u=1}^{m} \sqrt{p^{(u)} q^{(u)}} \quad (10)$$

$$d = \sqrt{1 - \rho[p,q]} \quad (11)$$

In particle filter algorithm, the target model can be represented by a state vector s_target as defined in Eq. (12) where (x, y) specify the center position of ROI, $(v_x,\ v_y)$ object's motion. w and h denote the width and height of ROI, respectively. The initial sample set $S_{initial} = \{s^{(n)}\}_{n=1\ldots N}$ compute as Eq. (13) with N is the number of samples, I is an identity matrix, and $r.v.$ is a multivariate Gaussian random variable. Eq. (14) represents a dynamic model which propagates the sample, where A defines the deterministic component of the model. By using every sample's weight and its state vector, the target human's position and size can be obtained from the estimated vector using Eq. (15). The Bhattacharyya distance uses to update each sample's weight as Eq. (16).

$$s_{target} = \{x, v_x, y, v_y, w, h\} \quad (12)$$

$$s^{(n)} = I s_{target} + r.v. \quad (13)$$

$$s_t = A s_{t-1} + r.v._{t-1} \quad (14)$$

$$E[S_t] = \sum_{n=1}^{N} \omega_t^{(n)} s_t^{(n)} \quad (15)$$

$$\omega^{(n)} = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{d^2}{2\sigma^2}} = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(1-\rho[p_{s^{(n)}},q])}{2\sigma^2}} \quad (16)$$

The resampling step avoids the problem of the degeneracy of the algorithm, in other words, avoiding the situation that most sample weights are close to zero. Eqs. (17), (18) and (19) determine the opportune moment of resampling step, where $rate \in (0,1)$, N_{eff} and N_{ths} are the effective number of samples and given of threshold sample, respectively.

$$N_{eff} < N_{ths} \quad (17)$$

$$N_{eff} = \frac{1}{\sum_{n=1}^{N} (\omega_t^{(n)})^2} \quad (18)$$

$$N_{ths} = rate * N \quad (19)$$

During resampling step, samples with high weight may be chosen several times, leading to identical copies, while others with relatively low weights may not be chosen at all. The initial resample step in particle filter selects samples randomly, sometimes selecting samples with relatively low weight. Consequently, it may track a different object as

the target object, decreasing the tracking accuracy. We proposed a weighted resampling algorithm to cover this problem. First, choose the top sample set S_t^{top} with N_{top} weights from set S_t, as in Eqs. (20) and (21), where top was a top rate which we set to 0.2.

$$N_{top} = top * N \tag{20}$$

$$S_t^{top} = \{s^{top(n)}\}_{n=1\ldots N_{top}} \tag{21}$$

We reproduced N samples in S_t according to the weight of $s^{top(n)}$. To update the target model over time, we applied the Gaussian mixture model (GMM). We chose K Gaussian distributions to approximate any continuous probability distribution where $N(x|\mu_k, \sigma_k)$ is the Gaussian distribution with mean μ_k and standard deviation σ_k. π_k is the weight of the Gaussian distribution, and its sum is equal to 1. Eq. (22) describes all this procedure.

$$p(x) = \sum_{k=1}^{K} p(k)p(x|k) = \sum_{k=1}^{K} \pi_k N(x|\mu_k, \sigma_k) \tag{22}$$

The occlusion handler in our work was a color-based one, which equaled similarities within the target model and the candidate model.

- We created a candidate model $c = \{c^{(u)}\}_{u=1\ldots m}$ from the ROI in current frame.
- We computed the similarity value between the target model $q' = \{q'^{(u)}\}_{u=1\ldots m}$ and the candidate model $c = \{c^{(u)}\}_{u=1\ldots m}$.
- If $similarity < ths_{sim}$, then we did not process the resampling step and assumed another object occluded the candidate model.
- $Count = Count + 1$.
- During tracking process, our algorithm iterated the steps 1~4 until the tracked human has appeared (similarity value larger than ths_{sim}) or $Count \geq 10$, which avoided the samples spread out of the image.

Our active camera was controlled by the Pelco P-protocol [19] through an RS-232 to RS-485 converter. It has to control pan, tilt angle, and zoom's step to achieve tracking purpose. We divided our FOV into 9 regions associated with pan-tilt directions, to keep the tracked object in the center of the FOV, as we demonstrate in Fig. 5. The zoom-in and zoom-out were activated if the target's size becomes smaller or larger than our defined arrangement.

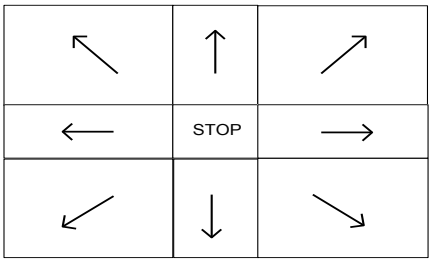


Fig. 5. FOV divided into 9 regions associated with control directions

3. Experimental Results

The proposed system is implemented on PC platform with Intel® Core™ i5 CPU 650 @ 3.20GHz, 4GB RAM, and developed in Borland C++ Builder 6.0 on Windows 7 environment. The system is being tested under several environments in order to verify its performance and stability.

3.1. Track on Video File

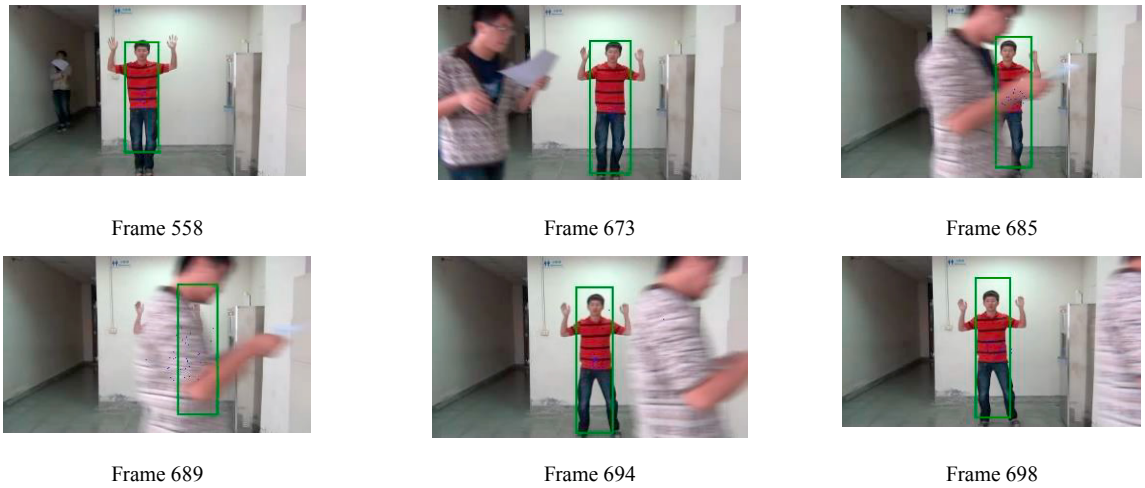
Two videos is used to verify the tracking system, with parameter particle filter as follows:
Number of samples N=30
Number of bins in histogram $m = 6 * 6 * 6 = 216$
State covariance: $\left(\sigma_x, \sigma_{v_x}, \sigma_y, \sigma_{v_y}, \sigma_w, \sigma_h\right) = (2,0.5,2,0.5,0.4,0.8)$

- Video 1 is used to verify the occlusion handler in the proposed system as shown in Fig. 6.
- Video 2 is used to verify the tracking performance in complex situation as shown in Fig. 7.

3.2. Track on Active Camera

The complexity of the environment is enough to verify the system while detecting and tracking moving human as follows:

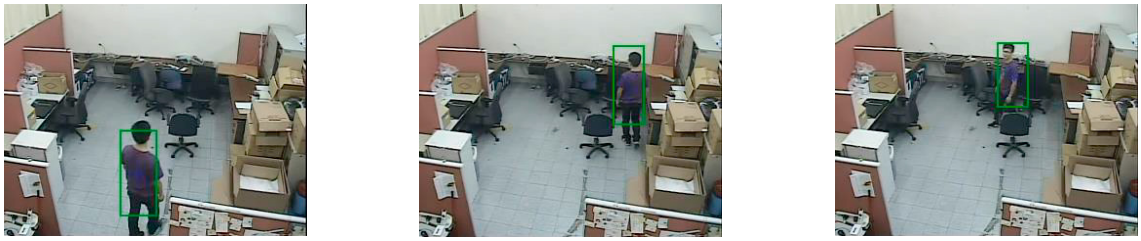
- Tracking by controlling pan, tilt, and zoom, with target human freely walking in the environment as shown in Fig. 8.
- Tracking a target human which more than one person walking in the same environment as shown in Fig. 9.



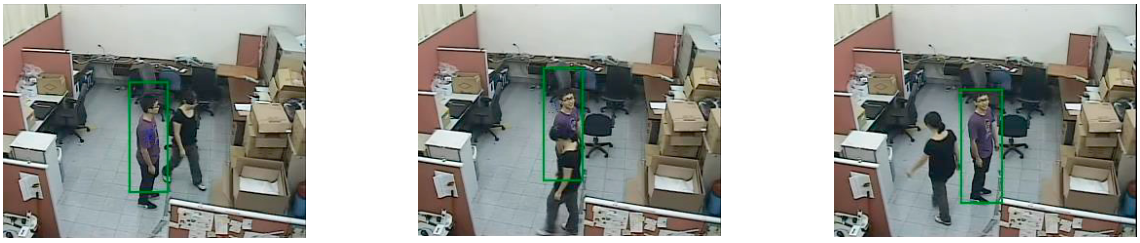
Frame 558 Frame 673 Frame 685

Frame 689 Frame 694 Frame 698

Fig. 6. Tracking with occlusion handler



(a)



(b)

Fig. 7. (a) frame 1436, 1547 and 1605 (b) frame 2757, 2818 and 2838

(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

Fig. 8. Combination of pan, tilt and zoom in / out

(a) (b) (c)

Fig. 9. Human tracking in multiple objects

4. Conclusions

The experiment results show that the proposed system tracks moving human by particle filter algorithm on active camera. Also, the tracking system is able to track the target human among many person walking in the same environment. Moreover, the zoom-in/out adjusts the resolution image of tracking human. There are several contributions on the proposed system, which are as follows: (a) The proposed system exactly distinguishes human and nonhuman. (b) The weighted resampling helps particle filter to preserve the samples with high weights. (c) Occlusion handler solves the temporal full occlusion condition. (d) The proposed system tracks target human smoothly by using the PID controller to determine the motion of camera.

Acknowledgements

This work was supported in part by the Australian Re-search Council (ARC) under discovery grant DP180100670 and DP180100656; in part by the Army Research Laboratory and was accomplished under Cooperative Agreement Number W911NF-10-2-0022 and W911NF-10-D-0002/TO 0023; in part by the Taiwan Ministry of Science and Technology under Grant Number: MOST 106-2218-E-009-027-MY3 and MOST 106-2221-E-009-016–MY2.

References

[1] R. C. Gonzalez, R. E. Woods, Digital Image Processing, Addison-Wesley, New York, 1992.
[2] D. Marr and E. Hildreth, “Theory of edge detection”, Proceedings of the Royal Society, vol. 207, pp. 197–217, London, 1980.
[3] W. Guo, D. L. Bi, L. Liu, “Human motion tracking based on shape analysis”, Proceedings of the International Conference on Wavelet Analysis and Pattern Recognition, pp. 2-4, Beijing, China, November 2007.
[4] T. Law, H. Itoh, and H. Seki, “Image filtering, edge detection, and edge tracing using fuzzy reasoning”, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 18, pp. 481-491, May 1996.
[5] O. Williams, A. Blake, and R. Cipolla, “Sparse bayesian learning for efficient visual tracking”, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 27, pp. 1292–1304, August 2005.
[6] A. Agarwal and B. Triggs, “Recovering 3D human pose from monocular images”, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 28, pp. 44-58, January 2006.
[7] B. Han and L. Davis, “Object tracking by adaptive feature extraction”, International Conference on Image Processing, pp. 1501-1504,

October 2004.

[8] R. T. Collins, Y. Liu and M. Leordeanu, “Online selection of discriminative tracking features”, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 27, pp.1631-1643, October 2005.

[9] K. Fukunaga and L. D. Hostetler, “The estimation of the gradient of a density function, with applications in pattern recognition”, IEEE Transactions on Information Theory, vol. 21, pp. 32-40, January 1975.

[10] D. Comaniciu, V. Ramesh, and P. Meer, “Kernel-based object tracking”, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 25, pp. 564-577, May 2003.

[11] D. Freedman and P. Kisilev, “Fast mean shift by compact density representation”, IEEE Conference on Computer Vision and Pattern Recognition, pp. 1818-1825, June 2009.

[12] F. L. Wang, S. Y. Yu, and J. Yang, “Robust and efficient fragments-based tracking using mean shift”, AEU - International Journal of Electronics and Communications, vol. 64, pp. 614-623, July 2010.

[13] F. Porikli and O. Tuzel, “Multi-kernel object tracking”, IEEE International Conference on Multimedia and Expo, pp. 1234–1237, July 2005.

[14] C. Yang, R. Duraiswami, and L. Davis, “Efficient mean-shift tracking via a new similarity measure”, IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 1, pp. 176–183, June 2005.

[15] R. V. Babu, P. Pe´rez, and P. Bouthemy, “Robust tracking with motion estimation and local Kernel-based color modeling”, Image and Vision Computing, vol. 25, pp.1205–1216, August. 2007.

[16] S. Feng, Q. Guan, S. Xu and F. Tan, “Human tracking based on mean shift and Kalman Filter”, International Conference on Artificial Intelligence and Computational Intelligence, 2009.

[17] P. Pe´rez, C. Hue, J. Vermaak, M. Gangnet, “Color-based probabilistic tracking”, Proceedings of European Conference on Computer Vision, pp. 661-675, 2002.

[18] K. Nummiaro, E. Koller-Meier, and L. V. Gool, “An adaptive color based particle filter”, Image and Vision Computing, vol. 21, pp. 99–110, 2003.

[19] http://www.commfront.com/RS232_Examples/CCTV/Pelco_D_Pelco_P_Examples_Tutorial2.HTM#1