Incorporating Long-Term Observations of Human Actions for Stable 3D People Tracking

Daisuke Sugimura, Yoshinori Kobayashi, Yoichi Sato
Institute of Industrial Science, The University of Tokyo
Tokyo 153–8505, Japan
{sugimura,yosinori,ysato}@iis.u-tokyo.ac.jp

Akihiro Sugimoto
National Institute of Informatics
Tokyo 101–8430, Japan
sugimoto@nii.ac.jp

Abstract

We propose a method for enhancing the stability of tracking people by incorporating long-term observations of human actions in a scene. Basic human actions, such as walking or standing still, are frequently observed at particular locations in an observation scene. By observing human actions for a long period of time, we can identify regions that are more likely to be occupied by a person. These regions have a high probability of a person existing compared with others. The key idea of our approach is to incorporate this probability as a bias in generating samples under the framework of a particle filter for tracking people. We call this bias the environmental existence map (EEM). The EEM is iteratively updated at every frame by using the tracking results from our tracker, which leads to more stable tracking of people. Our experimental results demonstrate the effectiveness of our method.

1. Introduction

The rapid increase of computing power and the reduction in cost of producing digital cameras have increased the number of potential applications of computer vision. In particular, vision-based tracking has become a key element for automated visual surveillance. However, real-life application, such as video surveillance, also require very accurate and stable tracking algorithms. In order to establish a more accurate and stable tracking algorithm, we must address several challenging issues such as: low image quality, changes in the appearance of targets, changes in image-intensity under varying illumination conditions, cluttered backgrounds and occlusion.

In recent years, a particle filter [3] has become a standard algorithm for vision-based tracking. A particle filter is a Bayesian sequential importance sampling technique, which recursively approximates the posterior probability distribution using a finite set of weighted samples. Recently, a number of methods to improve the robustness of a particle filter have been reported. Most of approaches attempt to prevent tracking failures by using various image features obtained from the observed images [5, 6, 9, 12].

However, in a real world situation it is inevitable that the tracking will fail. For example, when a person being tracking is completely occluded and then reappears in a different location, the tracker will usually lose the target and will not be able to restart tracking quickly. Thus we observe that it is important to devise a method for immediately re-initializing the tracker when targets are lost.

In order to address the issue of lost targets, other approaches that combine a particle filter with a re-initialization scheme have also been proposed [1, 4, 7, 11]. These methods use other image cues, different from those used in generating samples for tracking, to adjust or re-initialize tracking. In the case of 3D tracking, however, accurately reconstructing 3D position of the target is difficult because the image cues are not always properly obtained from all the images for various reasons such as occlusions. Therefore, it is necessary to introduce other types of information besides those based on observed images.

A method using non-image-based cues was proposed by [9] that incorporates the environmental restrictions of the observation space into the framework of a particle filter. The environmental restrictions are manually defined as the probability of a person existing in a region derived from the physical constraints of the observation space. This method generates new samples in regions that are not rejected by the environmental restrictions. However, this method cannot be used when physical constraints of the observation space are not known in advance. Furthermore, the probability of a person existing at a particular location cannot be fully represented from physical constraints alone. Even though the 3D characteristics of a certain location may allow for a human to occupy that space, it does not always indicate that a person will be found there with a high probability. For example, when two walkways exist in a room, it is possible that one walkway is used more frequently.

We reason that using the history of human movement through a scene is a better indicator of the probability of a person existing in certain locations, in comparison to using only 3D shape cues of an observation space. By using the history of human movement in a scene, we can acquire a more realistic probability map of a person existing at a certain location, which can also be used as a powerful cue to address the problem of re-initialization in 3D tracking.

In our work, we incorporate this probability map as the importance function into our particle filter-based 3D tracking. We call the probability of a person existing at different locations in the observation space as the *environmental existence map* (EEM). Our system simultaneously tracks people while dynamically updating the EEM via the online EM algorithm [8] using the online results of the tracker. In addition, we introduce a human action state transition model for adaptively controlling the updating of the EEM. Our proposed method is able to adapt to any environment and quickly re-initalize tracking when a target is lost.

2. Tracking people with the EEM

The system configuration is shown in Figure 2.1. The system consists of two interacting units: (i) a 3D tracker using the framework of a particle filter and (ii) a dynamic EEM updater which uses the history of basic human actions. In this section, we describe our tracking framework.

2.1. Tracking using a particle filter

Here we briefly review the basics of particle filter-based tracking. Suppose that a state of a target at time t is denoted by the vector $\mathbf{X}_t$, and the observation sequence up to time t is $\mathcal{Z}_t = \{\mathbf{z}_1, ..., \mathbf{z}_t\}$. A particle filter is a time-series filter for tracking the target by recursively estimating the probability distribution $p(\mathbf{X}_t|\mathcal{Z}_t)$ of the target. It approximates $p(\mathbf{X}_t|\mathcal{Z}_t)$ using a set of samples $\{\mathbf{s}_t^{(1)}, ..., \mathbf{s}_t^{(N)}\}$. Each sample $\mathbf{s}_t^{(n)}$ represents a hypothesis of the target and has a weight $\pi_t^{(n)}$ representing the value of $p(\mathbf{X}_t|\mathcal{Z}_t)$ at $\mathbf{s}_t^{(n)}$. A particle filter iterate the following steps:
(i) Sampling: Select samples $\{\mathbf{s'}_{t-1}^{(1)}, ..., \mathbf{s'}_{t-1}^{(N)}\}$ in proportion to weights $\{\pi_{t-1}^{(1)}, ..., \pi_{t-1}^{(N)}\}$ corresponding to samples $\{\mathbf{s}_{t-1}^{(1)}, ..., \mathbf{s}_{t-1}^{(N)}\}$.
(ii) Propagation: Propagate $\{\mathbf{s'}_{t-1}^{(1)}, ..., \mathbf{s'}_{t-1}^{(N)}\}$ according to the state transition probability $p(\mathbf{X}_t|\mathbf{X}_{t-1})$ and construct a prior $p(\mathbf{X}_t|\mathcal{Z}_{t-1})$. Then, generate new samples $\{\mathbf{s}_t^{(1)}, ..., \mathbf{s}_t^{(N)}\}$ from $p(\mathbf{X}_t|\mathcal{Z}_{t-1})$.
(iii) Weight Evaluation: Calculate the weights $\{\pi_t^{(1)}, ..., \pi_t^{(N)}\}$ corresponding to $\{\mathbf{s}_t^{(1)}, ..., \mathbf{s}_t^{(N)}\}$ using the observation model $p(\mathbf{Z}_t|\mathbf{X}_t)$ obtained from the observed images.

Our tracker tracks multiple human heads using multiple cameras in the world coordinate system $O-XYZ$ in which

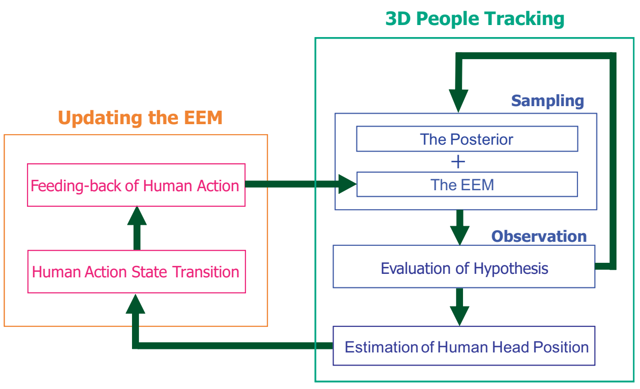

Figure 2.1: System configuration

the XY plane is assigned to the floor surface and the Z axis to the height. The human head is modeled by an ellipsoid whose center position is (x, y, z). The n-th hypothesis $\mathbf{s}_t^{(n)}$ at time t in the particle filter is represented by 3-D vector state, $\mathbf{s}_t^{(n)} = [x_t^{(n)}, y_t^{(n)}, z_t^{(n)}]^T$.

2.2. Incorporating the EEM as the importance function

The EEM is incorporated as the importance function into the framework of a particle filter. When sampling hypotheses, the prior $p(\mathbf{X}_t|\ \mathcal{Z}_{t-1})$ and the EEM $g_t(\mathbf{X}_t)$ are used as sampling sources at the time t. Under the framework of the ICONDENSATION [4], we consider the following three sampling methods. These sampling methods are chosen with a fixed ratio.
(a) Standard Tracking: standard sampling used in a particle filter. Hypotheses are sampled from the prior.
(b) Tracking Initialization: importance sampling for initializing tracking. Hypotheses are sampled from the EEM.
(c) Tracking Adjustment: importance sampling for adjusting tracking. Hypotheses are sampled from the EEM. In the weight evaluation step, the weight will be defined by using the EEM.

2.3. Evaluation of hypothesis

Each hypothesis is evaluated using a combination of multiple cues that come from foreground images and edge images. For hypothesis $\mathbf{s}_t^{(n)}$, we compute $\pi_{i,t}^{(n)}$ using the observed image acquired by the i-th camera and then take the product of weights $\pi_{i,t}^{(n)}$ to obtain the weight $\pi_t^{(n)}$ for $\mathbf{s}_t^{(n)}$. For the hypothesis $\mathbf{s}_t^{(n)}$ sampled from the EEM $g_t(\mathbf{X}_t)$ for the tracking adjustment (case (c)), the weight $\pi_t^{(n)}$ is defined by

$$\pi_t^{(n)} = \frac{p(\mathbf{X}_t = \mathbf{s}_t^{(n)}|\mathcal{Z}_{t-1})}{g_t(\mathbf{s}_t)} \prod_i \pi_{i,t}^{(n)} \ . \tag{2.1}$$

The 3D human head position $\mathbf{s}_t^{head}$ is then estimated as the expectation of the posterior $p(\mathbf{X}_t|\mathcal{Z}_t)$.

3. Updating the EEM

The EEM is modeled as the mixture of Gaussians,

$$g_t(\mathbf{X}_t) = \sum_{i=1}^{K} \omega_{i,t} \mathcal{N}(\mathbf{X}_t|\boldsymbol{\mu}_{i,t}, \Sigma_{i,t}) \ , \tag{3.1}$$

where $\mathcal{N}(\cdot)$ denotes the normal distribution, K denotes the number of Gaussians and $\omega_{i,t}, \boldsymbol{\mu}_{i,t}, \Sigma_{i,t}$ denotes the weight, mean and variance of the i-th normal distribution at time t, respectively. The mixture of Gaussians can represent a complex distribution and can easily generate random samples.

3.1. Dynamically updating the EEM

The EEM is iteratively updated at every frame by using human head position $\mathbf{s}_t^{head}$ estimated by our tracker using the online EM algorithm [8].

The EM algorithm is used to find the maximum likelihood estimates of the parameters of a probabilistic model, defined by a set of unobserved variables [2]. The EM algorithm iterates the expectation (E) step and the maximization (M) step. The former step computes the expectation of the likelihood (including the latent variables) and the latter step maximizes the expectation of the likelihood calculated in the former step to find the maximum likelihood estimates of the parameters. However, enormous computation is required as the number of observed data increases. The online EM algorithm has been proposed to overcome the computational requirements of batch processing, by computing the parameters of the model online [8]. The method uses the currently observed data $\mathbf{x}(t)$ and the estimated parameter $\theta^{(t-1)}$ for fast parameter estimation.

By using the online EM algorithm, the parameters of the mixture of Gaussians representing the EEM are updated with

$$\omega_{i,t} = \langle\langle 1 \rangle\rangle_i(t) \ , \quad \boldsymbol{\mu}_{i,t} = \frac{\langle\langle \mathbf{s}_t^{head} \rangle\rangle_i(t)}{\langle\langle 1 \rangle\rangle_i(t)} \ ,$$

$$\Sigma_{i,t} = \frac{\langle\langle \mathbf{s}_t^{head} \left(\mathbf{s}_t^{head}\right)^T \rangle\rangle_i(t)}{\langle\langle 1 \rangle\rangle_i(t)} - \boldsymbol{\mu}_{i,t} \left(\boldsymbol{\mu}_{i,t}\right)^T \ . \tag{3.2}$$

We note that $\langle\langle \ \cdot \ \rangle\rangle$ is defined as

$$\langle\langle \ f(\mathbf{x}) \ \rangle\rangle_i(t) = (1-\alpha)\langle\langle \ f(\mathbf{x}) \ \rangle\rangle_i(t-1) + \alpha f(\mathbf{x}) p(i \mid \mathbf{x}(t), \theta^{(t-1)}) \ , \tag{3.3}$$

where α denotes a learning weight and $p(i \mid \mathbf{x}(t), \theta^{(t-1)})$ denotes a posterior that the i-th normal distribution is selected. The online EM algorithm dynamically changes the contribution of the current observation by a weight α. This allows us to adaptively update the EEM based on the current state of the target.

3.2. Learning the EEM using the human action state transition model

To learn an EEM that is representative of all the different actions that occur in a scene, we have to consider both the temporal duration of an action as well as the importance of that action. For example, the action of a person entering a room is short in duration but a very important action when calculating the probability of a person existing in the observation space. Therefore, we have to devise a scheme to adaptively adjust the weight α according to the importance of the action that is currently taking place. We introduce a human action state transition model as shown Figure 3.1 to control α to ensure that we learn a representative EEM.

We describe details of the human action states and conditions on their transitions. The states of human actions that we consider in this paper are, Out-of-View, Appear, Active and Standing-Still. We assume that a person is always in one of these states and that the action state changes during tracking.

Out-of-View

Out-of-View is the state in which a person is not observed by the camera(s). Thus the learning weight in this state is set zero. In order to distinguish Out-of-View state from other states (determine whether people are in the field of view or not), we use the magnitude of the changes of likelihood in a particle filter as follows. If $\sum_{i=1}^{N} \pi_t^{(i)} < T_{exist}$, we consider that a person is not in the view ($\boldsymbol{c.1}$), whereas if $\sum_{i=1}^{N} \pi_t^{(i)} \geq T_{exist}$, a person is considered to be present ($\boldsymbol{c.2}$) (where N denotes the number of the hypotheses, $\pi_t^{(n)}$ represents the weight of n-th hypothesis in a particle filter and T_{exist} is the threshold). By using these conditions, the transition conditions are as follows. If the condition "$\boldsymbol{c.1}$" is satisfied, remain in the **Out-of-View** state. If the condition "$\boldsymbol{c.2}$" is satisfied, transit to the **Appear** state.

Appear

Appear is the state in which a person enters into the field of view of the camera(s). The learning weight in this state is set to $\alpha = k\alpha_{act}$, where $k > 1$ is an arbitrary constant and α_{act} is the learning weight in the **Active** state. The learning weight is set to a relatively high value because the regions where entering is observed are important for tracking initialization. The transition conditions are as follows. If the

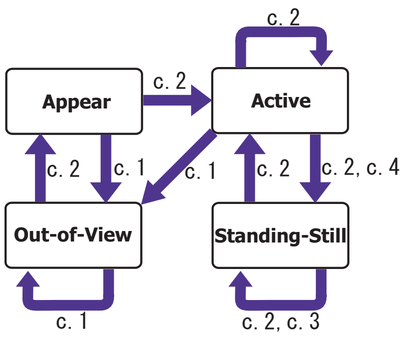

Figure 3.1: Human action state transition model: The state transition occurs when these conditions are satisfied.

condition "$c.1$" is satisfied, transit to the **Out-of-View** state. If the condition "$c.2$" is satisfied, transit to the **Active** state.

Active

Active is the state in which a person is moving within the field of view of the camera(s). We set the learning weight in this state as $\alpha = \alpha_{act}$, which represents a baseline weight for updating. In order to distinguish the **Active** state from the **Standing-Still** state, we introduce the following two conditions: (1) The human traveling speed v_t is less than T_{vel} [cm/frame] ($c.3$) and (2) condition "$c.3$" has been satisfied for more than T_{time} [frame] ($c.4$). By using these conditions, the transition conditions are as follows. If the condition "$c.1$" is satisfied, transit to the **Out-of-View** state. If the condition "$c.2$" is satisfied, remain in the **Active** state. If the both conditions "$c.2$" and "$c.4$" are satisfied, transit to the **Standing-Still** state.

Standing-Still

Standing-Still is the state in which a person is standing still at a certain location. The learning weight is set to reduce the value in proportion to the staying time t_{stay}, i.e., $\alpha = \alpha_{act}/t_{stay}$. This weight is set to prevent the probability of the EEM from concentrating in a particular location, thus making the initialization unit ineffective. The transition conditions are as follows. If only the condition "$c.2$" is satisfied, transit to the **Active** state. If the both conditions "$c.2$" and "$c.3$" are satisfied, remain in the **Standing-Still** state.

4. Experimental results

In order to test the effectiveness of the EEM for initialization, we performed one preliminary experiment to verify the construction of a reliable EEM and a second experiment to measure the improvement in inialization speed using the EEM. In both experiements we used two overhead cameras with overlapping fields of view. Image sequences were captured in VGA at 30 frames per second. Our system was composed of one server PC (Pentium4 3.2GHz, RAM 1GByte) for unified data processing and two client PCs (Pentium4 2.8GHz, RAM 2GByte) for each camera. These PCs were connected via Gigabit Ethernet at 1Gbps. We used the framework of a mixture particle filter [10] for tracking multiple human heads and the number of samples was 150 for each head. We empirically set the number of Gaussians of the EEM to $K = 7$ and the parameters of the human action state transition to $\alpha_{act} = 0.0005$, $k = 2$, $T_{exist} = 2.5$, $T_{vel} = 0.1$[cm/frame] and $T_{time} = 200$ [frame], respectively.

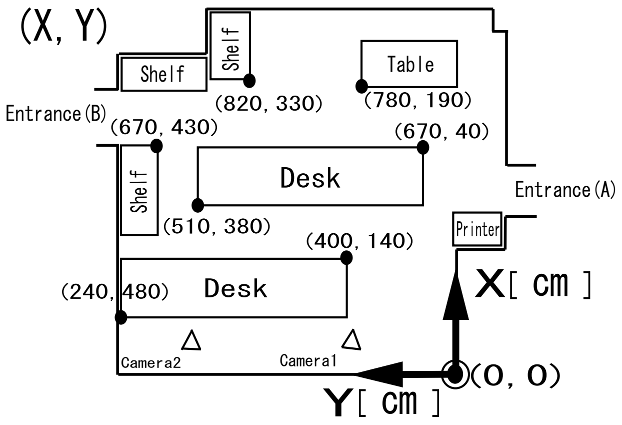

Figure 4.1: Layout of the observation space: Each pair of the numbers (x, y) in this figure means the coordinate value on the each point.

4.1. Constructing the EEM

We performed an experiment to construct the EEM by observing human actions within the scene (illustrated in Figure 4.1) for about 30 minutes. In this experiment, the two actions, walking through a particular walkway (Figure 4.2-(a)) and standing still while chatting around the table (Figure 4.2-(b)) were frequenty observed.

The result of the constructed EEM is shown in Figure 4.3. In Figure 4.3, we can see a peak over the region ($650 < X < 750$, $0 < Y < 100$). The peak was caused by two people chatting around the table in the back of the room for a couple of minutes in the observed image sequence. We can also see another smooth peak along the walkway ($650 < X < 750$, $0 < Y < 450$) in the same figure. This region corresponds to the walkway where people frequently walked. In contrast, the value of the EEM is almost 0 in the region ($500 < X < 650$, $50 < Y < 400$). From the layout of the observation space (Figure 4.1), we can easily see that this is due to the desk located in that region. This

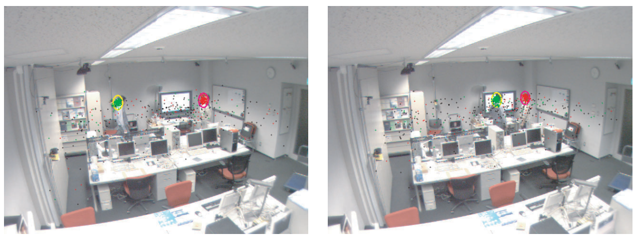
(a) (b)

Figure 4.2: Frequently observed actions: (a) shows that people frequently walked through the particular walkway and (b) shows that two people were chatting around the table

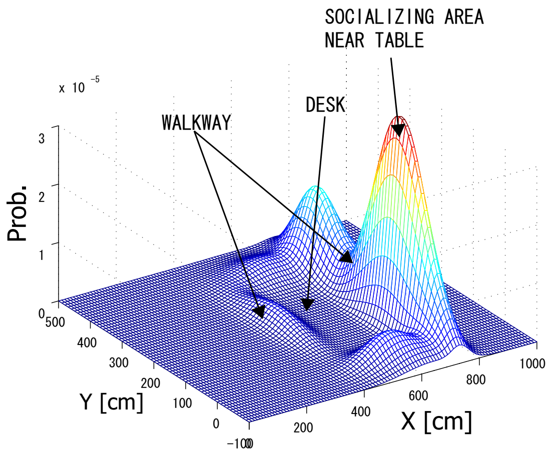

Figure 4.3: Result of the constructed EEM

represents the environmental restrictions in the observation space. From the characteristics of this experiment, we can observe that the constructed EEM properly represents the probability of a person existing in the scene.

4.2. Effectiveness of the EEM for initialization

4.2.1 Performance of initialization speed

We verified the performance of the initialization speed of our tracker with the EEM. More specifically, we calculated the difference between the frame when a person (re-)appears, f_{gt}, and the frame when our tracker (re-)starts tracking the person, f_{init}, in various cases (e.g. a person enters into the field of view, a person reappears from behind the board, etc.). The tracking (re-)initialization rapidly works when the difference, $f_{diff} \equiv |f_{gt} - f_{init}|$, becomes close to 0. An example of this experiment is shown in Figure 4.4. In order to observe the initialization speed improvements of our method using the learned EEM, we compared

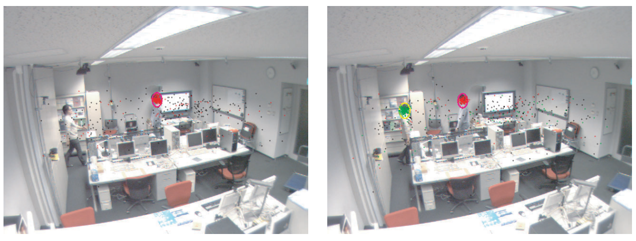
frame 1038 frame 1054

Figure 4.4: Example of the verification experiment of the performance of initialization speed: In the frame 1038 (left image), a new person appears but isn't tracked yet. In the frame 1054 (right image), tracking initialization is performed and tracking starts. We set the frame 1038 f_{gt}, and the frame 1058 f_{init}, then calculate $f_{diff} = |f_{gt} - f_{track}|$.

Table 4.1: Comparison results of the speed of initialization: The mean and the standard deviation are of f_{diff}.

	Mean [frame]	SD [frame]
Learned EEM	30.50	39.10
Uniformly dist. EEM	87.90	72.70

our method with the same tracker using an uniformly distributed EEM.

Table 4.1 shows the comparison results of the speed of the initialization. The mean indicates the average of time required for the (re-)initialization and the SD (standard deviation) indicates the range of the required time. Comparing the mean of the learned EEM to the uniformly distributed EEM, we can see that the learned EEM allows the tracker to (re-)initialize tracking in one third of the time required for the uniformly distributed EEM. The comparison results of the standard deviation shows that the range of the required time for (re-)initialization using the learned EEM is about half of using the uniformly distributed EEM. The improvement in the standard devation using the learned EEM is slightly smaller compared to the improvement in the mean. This is due to the difference of the (re-)initialization speed in each region. In fact, in the regions where the probability of a person existing has a high value ($650 < X < 750$, $0 < Y < 450$), the (re-)initialization rapidly performs. In contrast, in the other regions where the probability has a much lower value ($400 < X < 500$, $50 < Y < 400$), the (re-)initialization requires a longer time. These results show that the learned EEM can more stably and rapidly (re-)initialize tracking compared to the uniformly distributed EEM.

Table 4.2: Comparison results of tracking errors

		Average error [cm]		Standard deviation [cm]	
		Learned EEM	Uniformly dist. EEM	Learned EEM	Uniformly dist. EEM
Person A	XY plane	9.5617	-	4.9673	-
	Z axis	3.6692	-	2.7517	-
Person B	XY plane	6.2543	6.3623	3.4518	4.1039
	Z axis	3.9966	4.4390	2.5914	2.1907

4.2.2 Tracking results

Figure 4.5 shows the results of tracking two human heads using the learned EEM and uniformly distributed EEM with the groud truth. The ground truth was obtained by triangulation with target positions manually marked in the input images. Each tracked person moved as follows. Person A enters from entrance A near the printer and then walks through the walkway towards other entrance B. At the same time, Person B enters from entrance B and walks through the same walkway towards the back of the room. The tracking results of each person corresponds to “tracking result A” and “tracking result B”, respectively. The error measurements shown in Table 4.2 are the average Euclidean distance between the estimated position and the ground truth both on XY plane and Z axis. The standard deviations are also shown in Table 4.2. Compared to the tracking result B using the learned EEM (shown as “tracking result B with EEM” in Figure 4.5) and the tracking result B using the uniformly distributed EEM (shown as “tracking result B with Uniform Dist” in Figure 4.5), both of the results show that once the tracking initialization works successfully, tracking performs properly. For the speed of the tracking initialization, the tracker using the learned EEM started to track in frame 816, while using the uniformly distributed EEM started in frame 858. Compared with the frame in which Person B appeared (frame 810), the tracking initialization using the learned EEM worked faster than using the uniformly distributed EEM. From tracking result A, we can observe that tracking using the learned EEM (shown as “tracking result A with EEM” in Figure 4.5) performs properly. On the other hand, in the case of using the uniformly distributed EEM (shown as “tracking result A with Uniform Dist” in Figure 4.5), tracking completely fails because the tracking initialization does not work. From the results, we can see that the use of the learned EEM effectively works to reduce the required time for the tracking initialization and to achieve stable 3D tracking.

Figure 4.6 shows the tracking results in the case where a person is occluded by a board. The person being tracked walks through the walkway toward the board (frame 345) and then becomes occluded by the board (frame 360). The tracker has lost the target due to complete occlusion (from frame 360 to frame 719). When the person reappears (frame 720), however, the tracker immediately restarts tracking (frame 723).

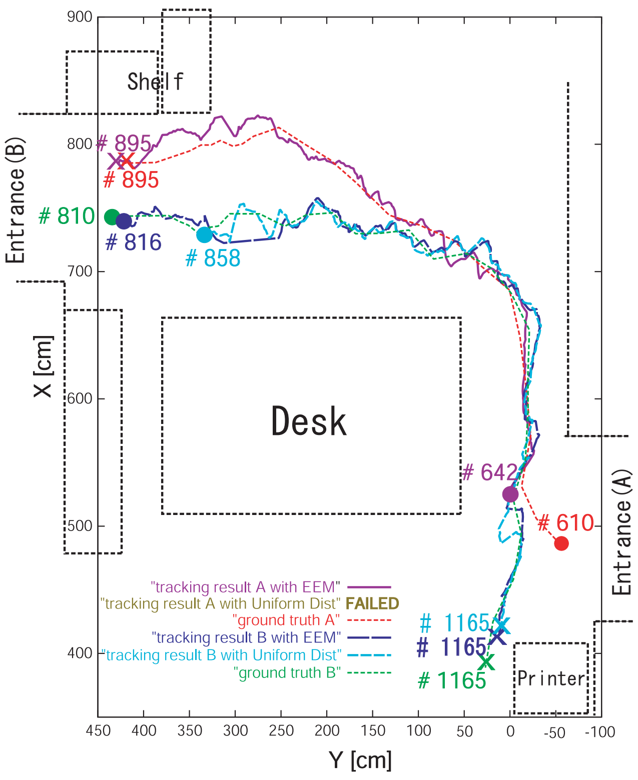

Figure 4.5: Tracking results of two human heads: The circle marked on the edge of each trajectory denotes the point which tracking started and the cross marked on the other edge of each trajectory denotes the point which tracking ended. Note that the legend “tracking result A with Uniform Dist” does not have a corresponding result. Because the tracking initialization did not work.

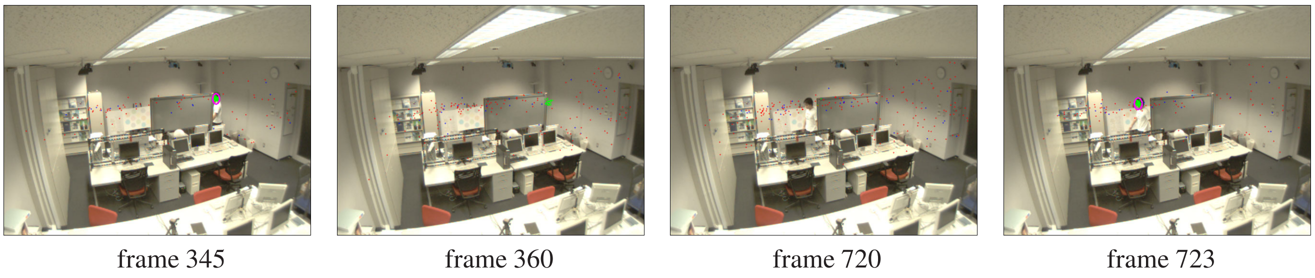

Figure 4.6: Tracking result: Rapidly recovering tracking from static occlusion

5. Conclusions

Certain human actions, such as walking or standing still are frequently observed at specific locations in an indoor environment. This implies that locations, where specific human actions are frequently observed, have a high probability of a person existing compared to other locations. We called this probability the environmental existence map (EEM). The EEM was iteratively updated at every frame by using the tracking results from our tracker and was adaptively incorporated into the learning scheme using a human action state transition model. These contributions indicate that our system automatically address changes of the EEM in time (morning, noon, evening and night) and in a layout in the scene. Through the verification experiments, we have shown that the flexible and immediate 3D tracking initialization can be performed by using the EEM.

In this paper we used a mixture of Gaussians and empirically set the number of Gaussians to model the EEM. The parameter setting of a mixture of Gaussians is data dependent and the EEM will not perform successfully for a different scene. In the future work, we will consider a nonparametric probability density representation, such as the parzen window or the nearest neighbor estimation, for the EEM.

References

[1] Y. Cai, N. Freitas, and J. Little. Robust visual tracking for multiple targets. In *European Conference on Computer Vision*, volume 4, pages 107–118, 2006.

[2] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum-likelihood from imcomplete data via the em algorithm. *Journal of Royal Statistical Society B*, 39:1–22, 1977.

[3] M. Isard and A. Blake. Condensation - conditional density propagation for visual tracking. *International Journal of Computer Vision*, 29(1):5–28, 1998.

[4] M. Isard and A. Blake. Icondensation: Unifying low-level and high-level tracking in a stochastic framework. In *European Conference on Computer Vision*, volume 1, pages 893–908, 1998.

[5] Y. Jin and F. Mokhtarian. Data fusion for robust head tracking by particles. In *IEEE workshop on Visual Surveillance and Performance Evaluation and Tracking*, pages 33–40, 2005.

[6] Y. Kobayashi, D. Sugimura, Y. Sato, K. Hirasawa, N. Suzuki, H. Kage, and A. Sugimoto. 3d head tracking using the particle filter with cascaded classifiers. In *British Machine Vision Conference*, pages 37–46, 2006.

[7] K. Okuma, A. Taleghani, N. Freitas, J. Little, and D. Lowe. A boosted particle filter: Multitarget detection and tracking. In *European Conference on Computer Vision*, volume 3021, pages 28–39, 2004.

[8] M. Sato and S. Ishii. On-line em algorithm for the normalized gaussian network. *Neural Computation*, 12(2):407–432, 2000.

[9] T. Suzuki, S. Iwasaki, Y. Kobayashi, Y. Sato, and A. Sugimoto. Incorporating environment models for improving vision-based tracking of people. *Systems and Computers in Japan*, 38(1):1592–1600, 2007.

[10] J. Vermaak, A. Doucet, and P. Perez. Maintaining multi-modality through mixture tracking. In *IEEE International Conference on Computer Vision*, volume 2, pages 1110–1116, 2003.

[11] Y. Wang, J. Wu, and A. Kassim. Particle filter for visual tracking using multiple cameras. In *IAPR Conference on Machine Vision Applications*, pages 298–301, 2005.

[12] C. Yang, R. Duraiswami, and L. Davis. Fast multiple object tracking via a hierachical particle filter. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 212–219, 2005.