Journal of Signal and Information Processing, **2013, 4, 299-307**
http://dx.doi.org/10.4236/jsip.2013.43038 Published Online August 2013 (http://www.scirp.org/journal/jsip)

Depth-Aided Tracking Multiple Objects under Occlusion

Anh Tu Tran, Koichi Harada

Graduate School of Engineering, Hiroshima University, Higashihiroshima, Japan.
Email: d103714@hiroshima-u.ac.jp, hrd@hiroshima-u.ac.jp

Received May 12th, 2013; revised June 15th, 2013; accepted July 12th, 2013

ABSTRACT

In this paper, we have presented a novel tracking method aiming at detecting objects and maintaining their la-bel/identification over the time. The key factors of this method are to use depth information and different strategies to track objects under various occlusion scenarios. The foreground objects are detected and refined by background subtraction and shadow cancellation. The occlusion detection is based on information of foreground blobs in successive frames. The occlusion regions are projected to the projection plane *XZ* to analysis occlusion situation. According to the occlusion analysis results, different objects' corresponding strategies are introduced to track objects under various occlusion scenarios including tracking occluded objects in similar depth layer and in different depth layers. The experimental results show that our proposed method can track the moving objects under the most typical and challenging occlusion scenarios.

Keywords: Visual Tracking; Multiple Object Tracking; Stereo Tracking; Occlusion Analysis

1. Introduction

Object tracking in the video sequence has played an important role in a research area of computer vision and a wide range of applications, such as video monitoring and surveillance, video conferencing and video summarization. Based on different camera configurations, objects can be tracked by using a single camera or stereo/multiple cameras. Object tracking with a single camera has studied in many literatures and different methods have been developed such as tracking by model-based tracking method [1], appearance-based methods [2-4], feature-based tracking [5], and statistical methods [6-8]. Many algorithms can obtain good results in some cases, such as when the targets are separated. However, multiple object tracking is still a challenging task due to the non-rigid motion of deformable object, persistent occlusion and the dynamic change of object attributes, such as color distribution, shape and visibility. In the real scene, occlusion between objects often occurs. For example, in typical surveillance scenario a person is partially or fully occluded by other people. Unfortunately, these occlusions lead to failed tracking. Some classical frameworks have been extended to track multi objects. In the multi-object tracking system [9], the level set method is used to handle contour splitting and merging. Extensive methods, *i.e.* Monte Carlo based probabilistic methods [10], game theory based approaches [11] and appearance model based deterministic methods [12,13] have been presented to solve the mutual occlusion problem. Another attractive research direction is stereo or multiple camera based method. While object detection and tracking with a single camera are well-explored topics, the use of multi-cameras technology for this purpose has attracted much attentions recently due to the availability and low price of new hardware. A multi-camera system observes the scene from two or more different views, and obtains more comprehensive information than a monocular camera system, which can take the advantage of depth information to improve the tracking system performance. Some tracking methods focus on usage of depth information only [14], or usage of depth information on better foreground segmentation [15], or usage depth information as a feature to be fused in a maximum likelihood model to predict 3D object positions [16].

In this work, we have presented a novel tracking method aiming at detecting objects and maintaining their label/identification across video frame sequence. The main points of this method are to use depth information and different strategies to track objects under various occlusion scenarios. **Figure 1** shows the flowchart of our tracking system.

The rest of this paper is organized as follows: Section 2 presents the proposed tracking method. Section 3

shows experimental results; and, finally, Section 4 concludes this paper.

2. Proposed Synthesis Method

Our proposed tracking system is shown in **Figure 1** and it consists of below main steps.

2.1. Depth Estimation

Depth estimation aims at calculating the structure and depth of objects in a scene from two views or a set of multiple views. This topic has been attracted extensive attentions in research communities. A comprehensive survey and evaluation of dense two-view stereo matching algorithms can be found in [17].

In this work, depth is estimated based on block matching algorithms proposal in [18]. This block matching technique is a one-pass stereo matching algorithm that uses a sliding sums of absolute differences window between pixels in the left image and the pixels in the right image. An example of depth image is shown in **Figure 2**.

2.2. Foreground Segmentation and Shadow Cancellation

Our method performs foreground segmentation to speed up the process of object tracking. There are many fore-

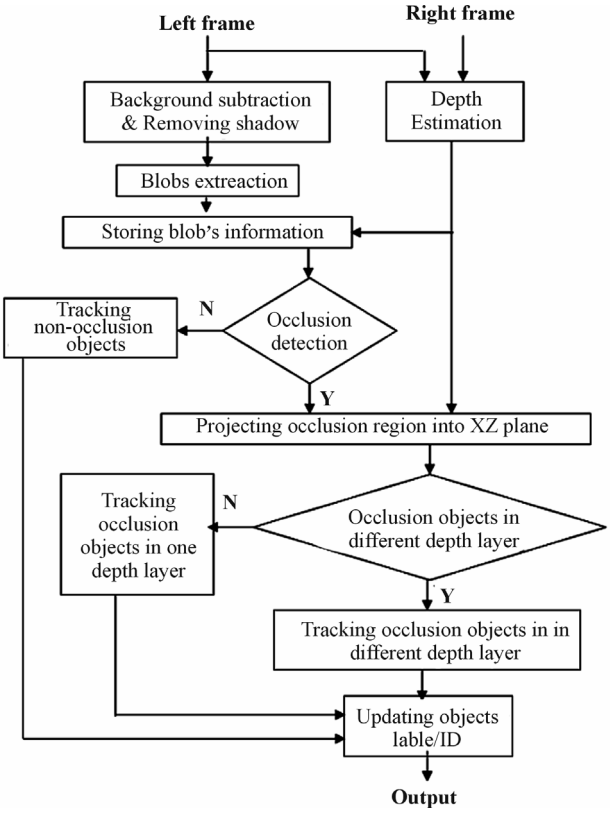


Figure 1. The flowchart of proposed tracking method.

Figure 2. Color image and depth image.

ground segmentation algorithms for instance of Gaussian mixture model [19,20]. In our method, we use simple technique based on absolute differences between current image and background image.

In some cases, we have the fixed cameras observing the scene, so we may have an image of the background of the scene. However, in most case this background is not readily available. Moreover, the background scene often evolves over time because for example the light condition might change or because of new object could be added or removed from the background. Therefore, it is necessary to dynamically build the background model by regularly updating it. This can be accomplished by computing moving average using the following formula:

$$\mu_t = (1-\alpha)\,\mu_{(t-1)} + \alpha\, p_t, \qquad (1)$$

where, p_t is pixel value ate a given time t, $\mu_{(t-1)}$ is the current average value, and α is called the learning rate and it defines the influence of the current value.

In our method, first a color background model is created by computing a moving average for each channel (R, B and G channels of color image) of each pixel of incoming frames (around 10 frames). The decision to define a foreground pixel is simply based on comparing the current frame with background model and then updating this. Specifically,

$$F(p) = \begin{cases} 0 & \text{if } \left|\mathrm{I_c}(p) - I_{bg}(p)\right| < t_H \\ \mathrm{I_c}(p) & \text{otherwise} \end{cases}, \qquad (2)$$

where, $F(p)$ is value of pixel p in foreground image, $\left|I_c(p) - I_{bg}(p)\right|$ represents the absolute color difference between the color value at pixel p of current frame $I_c(p)$ and the color value of pixel p of background frame $I_{bg}(p)$ of R, G, B channels. t_H is threshold and for the each color channel this threshold can be set to $0.3 * I_{bg}(p)$. An example of foreground image is shown in **Figure 3**.

However, the segmented foreground image includes noise affected by the shadow. The shadow regions are the parts of moving objects. Shadow detection and removing will be used to refine the foreground. To avoid the effects

of shadows, a shadow detection described in [21] is employed. More detail about this method, please refer to [21]. The result of the shadow detection is shown in **Figure 4**.

2.3. Blobs Extraction and Blobs' Information Store

First, we try to extract the blobs of objects from segmented foreground image. Blob extraction is performed on the foreground binary image by connected component labeling using CvBlobLib library [22]. The foreground binary image can obtained from simple threshold operation followed by the application of eroded and dilated operation on the segmented foreground image. CvBlobsLib provides two basic functionalities: extracting 8-connected components, referred to as blobs, in binary or grayscale images using Chang's contour tracing algorithm [23], and filtering the obtained blobs to get the objects in the image that we are interested in. In our method, we remove any blobs that have area small than 100 pixels.

For every frame, after extracting blobs, information about blob is stored in a structured record for later processing steps. The blob's information includes total number of blob (NB), blob's center coordination (x, y), blob area (BA), and the average depth value of blob $(Z_{\min})$. Blobs are also given temporally identification (ID).

(a) (b)

Figure 3. Foreground segmentation. (a) input image; (b) segmented foreground image.

(a) (b)

Figure 4. Shadow regions detection. (a) segmented foreground image; (b) shadow detection (the blue color pixels are the detected shadow).

We define two kinds of distance: the distance between blob i and blob j in the same frame t and the distance between blob i of frame t and blob j of frame $(t-1)$.

The distance $d(i_t, j_t)$ between blob i and blob j in the same frame t is computed by:

$$d(i_t, j_t) = \sqrt{\left(x_{i_t} - x_{j_t}\right)^2 + \left(y_{i_t} - y_{j_t}\right)^2}, \quad (3)$$

where $\left(x_{i_t}, y_{i_t}\right)$ and $\left(x_{j_t}, y_{j_t}\right)$ are the center coordination of blob i and blob j at frame t, respectively.

Similarly, a distance $D\left(i_t, j_{(t-1)}\right)$ between blob i of frame t and blob j of frame $(t-1)$ is calculated by:

$$D\left(i_t, j_{(t-1)}\right) = \sqrt{\left(x_{i_t} - x_{j_{(t-1)}}\right)^2 + \left(y_{i_t} - y_{j_{(t-1)}}\right)^2}, \quad (4)$$

where $\left(x_{i_t}, y_{i_t}\right)$ and $\left(x_{j_{(t-1)}}, y_{j_{(t-1)}}\right)$ are the center coordination of blob i at frame t and blob j at frame $(t-1)$ correspondingly.

2.4. Occlusion Detection

We detect the occlusion in current frame t according to blobs' information at frame t and previous frame $(t-1)$. It is based on two clues. The first clue comes from the shortest distance between blobs at the same frame $(t-1)$ and the second one is the difference of number of blobs at frame t and $(t-1)$. First, we find the shortest distance $d\left(i_{(t-1)}, j_{(t-1)}\right)$ between blobs in frame $(t-1)$, assuming that it occurs between blob m and blob n, *i.e.*

$$d_{\min}\left(n_{(t-1)}, m_{(t-1)}\right) = \min\left\{d\left(i_{(t-1)}, j_{(t-1)}\right)\right\}.$$

We define an occlusion flag $f_{\text{occ\_start}}$. This flag gets value t if occlusion is found at frame t and otherwise it gets value -1. Specially (see Equation (5)),

where, $NB_{(t-1)}$ and NB_t are the total number of blobs in frame $(t-1)$ and frame t respectively. $d_{\text{threshold}}$ is the threshold of blob distance at the same frame.

$$f_{\text{occ\_start}} = \begin{cases} t & \text{if } d_{\min}\left(n_{(t-1)}, m_{(t-1)}\right) < d_{\text{threshold}} \text{ and } NB_t < NB_{(t\text{-}1)} \\ -1 & \text{otherwise} \end{cases}, \quad (5)$$

Similarly, we also detect when the occlusion terminates. The end of occlusion is checked based on the shortest distance between blobs at the current frame t and the difference of number of blobs at current frame and previous frame $(t-1)$. We define the end of occlusion flag $f_{\text{occ\_end}}$ as following:

$$f_{\text{occ\_end}} = \begin{cases} t & \text{if } d_{\min}(n_t, m_t) < d_{\text{threshold}} \text{ and } NB_{(t-1)} < NB_t \\ -1 & \text{otherwise} \end{cases}, \tag{6}$$

2.5. Object Tracking

According to the result of occlusion detection, the tracking objects can be dividing into two types: tracking objects without occlusion and tracking objects under occlusion.

2.5.1. Tracking Objects without Occlusion

The video objects correspondence under non-occlusion is obtained through the shorted distance $D(i_t, j_{(t-1)})$ between blobs in previous frame and blobs in the current frame. This distance between blobs in previous frame and blobs in the current frame is calculated by Equation (4). For instance, once a foreground blob m at frame $t(B_t^m)$ finds its corresponding blob $B_{(t-1)}^n$ in frame $(t-1)$, its label or identification (ID) is updated correspondingly to the ID of blob $B_{(t-1)}^n$. Specially,

$$ID \text{ of } B_t^m \equiv ID \text{ of } B_{(t-1)}^n$$
$$\text{if} \begin{cases} D(B_t^m, B_{(t-1)}^n) = \min\left\{D(B_t^m, B_{(t-1)}^j)\right\}, j = 1,2,\cdots,NB_{(t-1)} \\ D(B_t^m, B_{(t-1)}^n) < D_{\text{thres}} \end{cases} \tag{7}$$

where B_j^k denotes the blob k at frame j; $NB_{(t-1)}$ is number of blobs in frame $(t-1)$; D_{thres} is distance threshold.

2.5.2. Tracking Objects under Occlusion

The main idea of our tracking method is that the object label or identification (*ID*) is maintained constantly during occlusion and after they switch their positions.

When occlusion occurs, we can detect and extract the occlusion region. We also can detect and separately extract a list of objects that are non-occlusion objects in previous frame but overlaying each other in current frame.

In order to track the objects under occlusion, depth information is used to analysis the occlusion situation. First, the occluded regions are projected to the ground plane XZ according to their horizontal position and their depth gray level (more detail in the next subsection). Then according to the XZ plane, the occlusion objects can be divided into two types based on the depth ranges: 1) in the different depth layer or 2) in the same depth layer. We are dealing with these situations as following parts.

1) Project Occlusion Regions into Ground Plane *XZ*

Each foreground pixel in occlusion region has $3D$ information obtained from depth map. These pixels are projected to the ground plane XZ according to their horizontal position and their depth gray level, where *X* is the width of the depth map and the range of *Z* is [0,255]. The projected point, which is located at (x, z), is defined as $p(x, z)$. The value at position $p(x, z)$ of projection plane is the total number of points at position *x* in the depth maps that have same gray level (depth value *z*). **Figure 5** illustrates the image plane *XY* and ground plane *XZ*.

In order to remove noisy points, if the value at point $p(x, z)$ is less than threshold T_1 the point $p(x, z)$ will be discarded. Then we also apply morphological operations (dilating and eroding operation) to remove noisy points and connect nearby points. The remaining points in *XZ* plane are grouped in to the blobs that are based on connected component analysis technique using CvBlobLib library [22]. If a projected blob is small than threshold T_2, it is consider a noise and it will be removed. The projected blobs are defined as $\left\{PB_j \middle| (j = 1,2,\cdots,m)\right\}$, where *m* is total number of projected blobs. Each projected blobs PB_j is mask as object regions. **Figure 6** shows an example of projected blobs in *XZ* plane.

As mention before, according to the projected blobs in *XZ* plane, the occlusion objects can be divided into two types based on the depth ranges: 1) in the different depth layers or 2) in the one depth layer.

2) Tracking Occluded Objects in Different Depth Layers

Figure 6 shows the case of occluded objects is in different depth layers. Once occluded objects have different

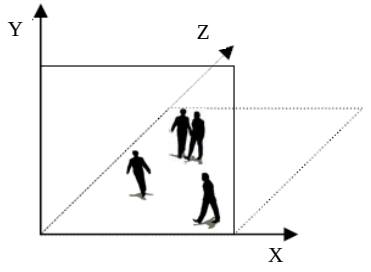
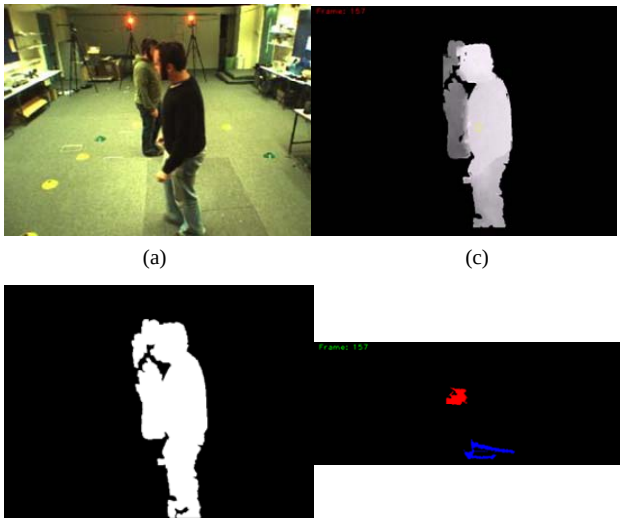


Figure 5. The image plane *XY* and ground plane *XZ*.



(a) (c)

(b) (d)

Figure 6. Projected foreground blobs in *XZ* plane. (a) input image; (b) foreground blob (occlusion region); (c) occlusion region (depth image); (d) projected blobs in *XZ* plane.

depth layer, they can be segmented in color image by means of their depth ranges. An example of color image segmentation by means of depth ranges is shown **Figure 7**.

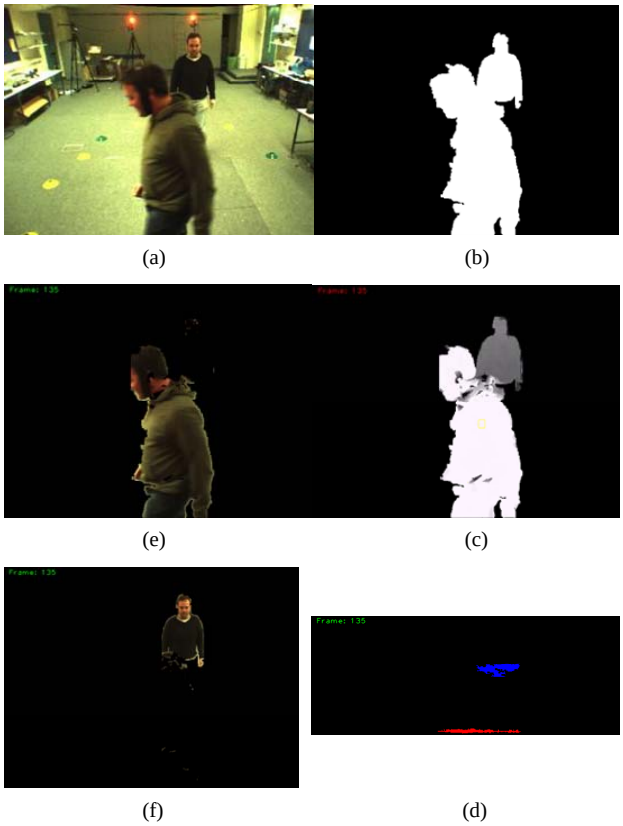
In our method, object correspondence under different layer is based on Bhattacharyya distance [24] between the color histograms. In statistics, the Bhattacharyya distance measures the similarity of two probability distributions. In our case, Bhattacharyya distance represents the similarity between two normalized histograms. The Bhattacharyya distance is calculated by:

$$BD = \sqrt{1 - \sum_{b=1}^{N} \sqrt{p_b q_b}}, \tag{8}$$

where, BD denotes Bhattacharyya distance; p and q are the two normalized color histograms; N is number of bin in histogram.

Let $O^k = \{O_1^k, O_2^k, \cdots, O_m^k\}$ is the occluded objects at frame k and $U^n = \{U_1^n, U_2^n, \cdots, U_m^n\}$ denotes the existing non-occluded objects in previous frame n (the frame before occlusion is found). The color histogram of each occluded objects and existing non-occluded objects are $H_{O_i^k}$ and $H_{U_j^n}$, respectively. In this paper, color histograms are created from hue component of HSV color space. **Figure 8** shows an example of the color histogram.

For each occluded object O_i^k, we calculate the Bhattacharyya distance $BD(O_i^k, U_j^n)$ between color histogram of this object and color histogram of every object U_j^n in U^n and then find the shortest distance $BD_{\min}$.

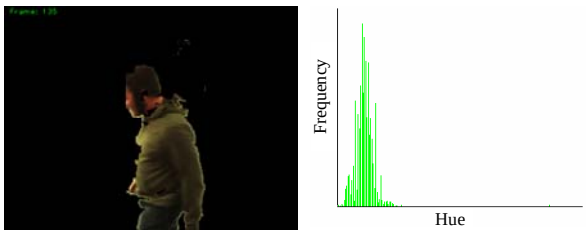


(a) (b)

(e) (c)

(f) (d)

Figure 7. An example of image segmentation by means of depth ranges. (a) input image; (b) foreground blob (occlusion region); (c) occlusion region (depth image); (d) Projected blobs in *XZ* plane (occluded objects in different depth layers); (e) segmented object based on depth layer 1 (red color in d); (f) segmented object based on depth layer 2 (blue color in d).



(a) (b)

Figure 8. An example of color histogram. (a) color image; (b) color histogram.

The Bhattacharyya distance $BD(O_i^k, U_j^n)$ is computed according to Equation (8), specially:

$$BD(O_i^k, U_j^n) = \sqrt{1 - \sum_{b=1}^{N} \sqrt{H_{O_i^k}(b) H_{U_j^n}(b)}}, \tag{9}$$

The occluded object O_i^k will update its ID according to U_j^n if $BD(O_i^k, U_j^n) = BD_{\min}$. **Figure 9** shows the numeric results of calculating Bhattacharyya distance

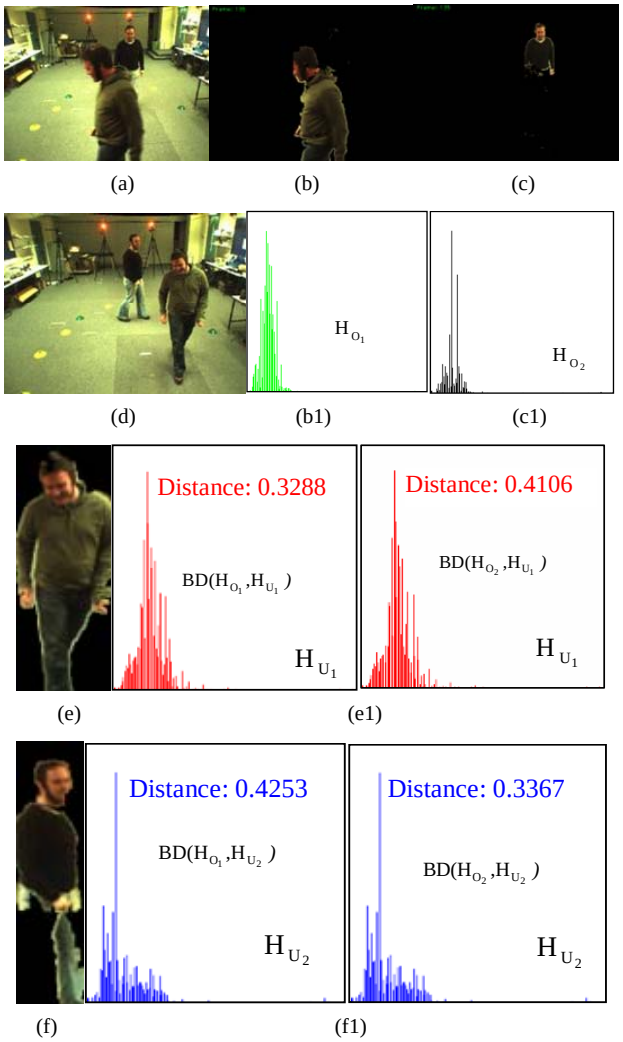


Figure 9. Bhattacharyya distance between two histograms. (a) input frame (having occlusion objects); (b) segmented obj. 1 in input frame (O_1); (c) segmented obj. 2 in input frame (O_2); (d) Previous frame (before occlusion happened); (b1) Histogram of object O_1 in (b) (H_{O_1}); (c1) Histogram of object O_2 in (c) (H_{O_2}); (e) object 1 before occlusion happened (U_1); (e1) Histogram of object U_1 in (e) (H_{U_1}); distance between H_{O_1} and H_{U_1} is 0.3288; distance between H_{O_2} and H_{U_1} is 0.4106; (f) object 2 before occlusion happened (U_2); (f1) Histogram of object U_2 in (f) (H_{U_2}); distance between H_{O_1} and H_{U_2} is 0.4253; distance between H_{O_2} and H_{U_2} is 0.3367.

between two histograms.

Figure 10 illustrates an example of tracking occluded objects in different depth layers.

3) Tracking Occluded Objects in One Depth Layer

Figure 11 shows an example of occluded objects in similar depth layer.

When occlusion objects have similar depth range or

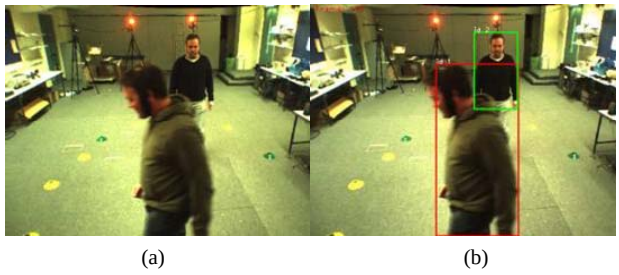


Figure 10. Tracking occluded objects in different depth layers. (a) Input frame (having occlusion objects); (b) Output (tracking occlusion objects in different layers).

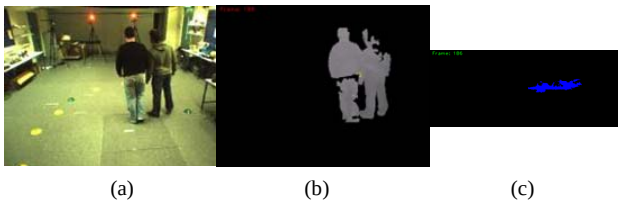


Figure 11. An example of tracking occluded objects in one depth layer. (a) input frame (having occlusion objects); (b) occlusion region (depth image); (c) projected blobs in XZ plane (occlusion objects in one depth layer).

full occlusion, it is difficult to segment and track multiple objects as above technique. To deal with this problem, we propose the tracking method based on camshift (Continuously Adaptive Mean shift) algorithm [25] as following part.

Assuming that there are m occluded object $O_i^k\left(i=1,2,\cdots,m\right)$ in the occlusion region R_{occ}^k, their existing corresponding tracks in the previous frame are $U_j^{(k-1)}$. Our algorithm has following steps:

1) Pre-computing the color histogram H_U for every existing track U_j^n, and the color histogram H_R for occlusion region. Here we calculate a hue histogram from HSV color space.

2) Based on the average depth values, sorting the list of object $U_j^{(k-1)}$ so that the object with biggest depth $U_{\text{foremost}}^{(k-1)}$ (*i.e.* the shortest distance from camera) goes first.

3) Calculating a back projection of a hue plane of occlusion region using the pre-computing histogram of $U_{\text{foremost}}^{(k-1)}$. Based on the back projection image, finding in R_{occ}^k the corresponding object of $U_{\text{foremost}}^{(k-1)}$ using camshift algorithm. Label it as O_{foremost}^k with ID according to the ID of $U_{\text{foremost}}^{(k-1)}$.

Figure 12 shows an example of projection image.

4) Removing the O_{foremost}^k in R_{occ}^k. Selecting the next track in the sorted $U_j^{(k-1)}$ list and running step (3) to

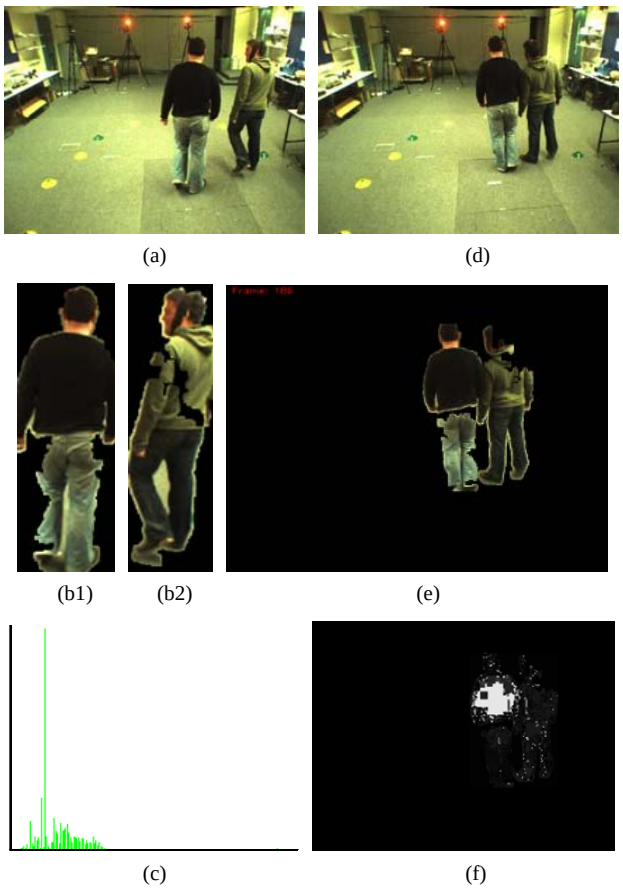


Figure 12. An example of projection image. (a) Previous frame (before occlusion happened); (b1) Object 1 before occlusion happened (with bigger depth); (b2) Object 2 before occlusion happened; (c) Histogram of object 1 in (b1); (d) input frame (having occlusion objects); (e) occluded objects in input frame; (f) Projection image of occluded region in input frame using pre-computing histogram in (c).

find the next O_j^k in R_{occ}^k.

5) Repeating step (4) until all object in O^k in R_{occ}^k finds their corresponding track.

Figure 13 demonstrates a result of tracking occluded objects in one depth layer.

In practice, when projecting partly occluded objects or fully occluded objects with similar depth ranges into XZ plane, we will obtain only one blob/region in XZ plane. In the case, the objects in similar depth range are partly occluded, the above algorithm will work well (see **Figure 13**). However, the fully occluded objects current frame t will reappear as partial occluded objects bind their occluder in the later frame, so it will be tracked by our method (see example in **Figure 14**).

3. Experimental Results

In this section, we show the experimental results to evaluate the proposed tracking method. We evaluate the tracking performance by the capability of detecting and maintaining constant ID of foreground objects during the occlusion and after the occlusion over.

The proposed tracking method has been test on some video sequences. The input of out method is a pair of video sequence and the output is the left video sequence in which has a set of moving objects labeling with ID and bounding boxes with different color.

Figure 15 shows results of tracking non-occlusion objects. In the example 15(a), in the 3 frames (left to right) each object appears one after another and in the fourth frame, object with ID has gone out the scene. These results illustrate the ability of our algorithm in detecting object, assigning and maintaining the objects ID.

We demonstrate the result of tracking of occluded objects in different depth layers in **Figure 16**. Our proposed method can successful detect the object under partial occlusion. All of objects have constantly label over the

Figure 13. An example of tracking occluded objects in one depth layer. (a) Input frame (having occlusion objects); (b) Output (tracking occlusion objects in different layers).

Figure 14. Tracking fully occluded objects. "Room sequence": from left to right, the frame indexes are 67, 72, 73 and 74, respectively.

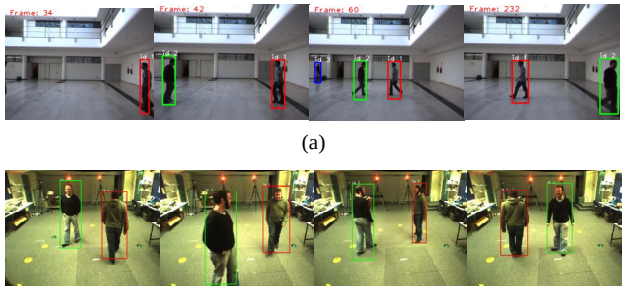


Figure 15. Result of tracking non-occluded objects. (a) Tracking of non-occluded objects in "Gym sequence" (from left to right, the frame number are: 34, 42, 60 and 201); (b) Tracking of non-occluded objects in "Room sequence" (from left to right, the frame number are: 0, 19, 113 and 250, respectively.

time even they are moving in variety of pose and position.

Tracking the occluded object under the similar depth layer is shown in **Figure 17**. These examples show that the proposed algorithm can detect and track a partial occluded object that is equivalent to at least one half a human body.

Figure 14 shows the case an object is fully occluded and in the similar depth layer with its occluder and our system cannot detect it. However, in the future time when this object reappears as partial occluded object, the system can detect and maintain its ID. This example demonstrate the capability of proposed algorithm in term of maintain constant label for object over the video sequence.

4. Conclusions

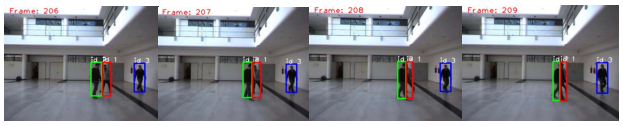
In this paper, we have presented a novel tracking method aiming at detecting objects and maintaining their ID over the time. The key factor of this method is to use depth information to track objects under various occlusion scenarios. Different object tracking strategies are applied according to occlusion situation including finding corresponding objects based on Bhattacharyya distance between two histograms and using a camshift based algorithm with the help of object depth ordering. The experimental results have confirmed the capability of our proposed objects tracking algorithm under the most typical and challenging occlusion scenarios.

However, the proposed algorithm can work only in an indoor or medium sized environment since the reliability of depth information diminishes in proportion to the distance from the camera and only when the moving velocity of objects is slow. In the future work, to construct a robust moving object tracking system in both indoor and outdoor environment, we will study to use more object's features to classify and track the objects.

(a)

(b)

Figure 16. Tracking occluded objects in different depth layers. (a) "Gym sequence": from left to right, the frames indexes are 65, 73, 87 and 145, respectively; (b) "Room sequence": from left to right, the frame indexes are 127, 129, 132 and 135, respectively.



(a)

(b)

Figure 17. Tracking occluded objects in one depth layer. (a) "Gym sequence": from left to right, the frames indexes are 206, 207, 208 and 209, respectively; (b) "Room sequence": from left to right, the frame indexes are 236, 237, 238 and 239, respectively.

REFERENCES

[1] D. Koller, K. Danilidis and H.-H. Nagel, "Model-Based Object Tracking in Monocular Image Sequences of Road Traffic Scenes," *International Journal of Computer Vision*, Vol. 10, No. 3, 1993, pp. 257-281. doi:10.1007/BF01539538

[2] T. Hai, H. S. Sawhney and R. Kumar, "Object Tracking with Bayesian Estimation of Dynamic Layer Representations," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 24, No. 1, 2002, pp. 75-89. doi:10.1109/34.982885

[3] A. D. Jepson, D. J. Fleet and T. F. El-Maraghi, "Robust Online Appearance Models for Visual Tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 25, No. 10, 2003, pp. 1296-1311. doi:10.1109/TPAMI.2003.1233903

[4] D. Comaniciu, V. Ramesh and P. Meer, "Kernel-Based Object Tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 25, No. 5, 2003, pp. 564-577. doi:10.1109/TPAMI.2003.1195991

[5] J. Shi and C. Tomasi, "Good Features to Track," *Proceedings IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (CVPR), Seattle, 21-23 June 1994, pp. 593-600.

[6] Y. Rui and Y. Chen, "Better Proposal Distributions: Object Tracking Using Unscented Particle Filter," *Proceedings of the Computer Vision and Pattern Recognition*, Vol. 2, 2001, pp. 786-793.

[7] M. Isard and A. Blake, "CONDENSATION—Conditional Density Propagation for Visual Tracking," *International Journal of Computer Vision*, Vol. 29, No. 1, 1998, pp. 5-28. doi:10.1023/A:1008078328650

[8] D. Beymer, P. McLauchlan, B. Coifman and J. Malik, "A Real-Time Computer Vision System for Measuring Traffic Parameters," *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, San Juan, 17-19 June 1997, pp. 495-501.

[9] N. K. Paragios and R. Deriche, "A PDE-Based Level-Set Approach for Detection and Tracking of Moving Objects," *The Sixth International Conference on Computer*

Vision, Bombay, 4-7 January 1998, pp. 1139-1145.

[10] Z. Tao, R. Nevatia and W. Bo, "Segmentation and Tracking of Multiple Humans in Crowded Environments," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 30No. 7, , 2008, pp. 1198-1211. doi:10.1109/TPAMI.2007.70770

[11] X. Zhou, Y. F. Li and B. He, "Game-Theoretical Occlusion Handling for Multi-Target Visual Tracking," *Pattern Recognition*, Vol. 46, No. 10, 2013, pp. 2670-2684. doi:10.1016/j.patcog.2013.02.013

[12] V. Papadourakis and A. Argyros, "Multiple Objects Tracking in the Presence of Long-Term Occlusions," *Computer Vision and Image Understanding*, Vol. 114, No. 7, 2010, pp. 835-846. doi:10.1016/j.cviu.2010.02.003

[13] M. Wu, X. Peng, Q. Zhang and R. Zhao, "Segmenting and Tracking Multiple Objects under Occlusion Using Multi-Label Graph Cut," *Computers & Electrical Engineering*, Vol. 36, No. 5, 2010, pp. 927-934. doi:10.1016/j.compeleceng.2009.12.013

[14] E. Parvizi and Q. M. J. Wu, "Multiple Object Tracking Based on Adaptive Depth Segmentation," *Canadian Conference on Computer and Robot Vision*, Windsor, 28-30 May 2008, pp. 273-277.

[15] S. J. Krotosky and M. M. Trivedi, "On Color-, Infrared-, and Multimodal-Stereo Approaches to Pedestrian Detection," *IEEE Transactions on Intelligent Transportation Systems*, Vol. 8, No. 4, 2007, pp. 619-629. doi:10.1109/TITS.2007.908722

[16] R. Okada, Y. Shirai and J. Miura, "Object Tracking Based on Optical Flow and Depth," *International Conference on Multisensor Fusion and Integration for Intelligent Systems* (*IEEE/SICE/RSJ*), Washington DC, 8-11 December 1996, pp. 565-571.

[17] D. Scharstein and R. Szeliski, "A Taxonomy and Evaluation of Dense Two-Frame Stereo Correspondence Algorithms," *International Journal of Computer Vision*, Vol. 47, No. 1-3, 2002, pp. 7-42. doi:10.1023/A:1014573219977

[18] K. Konolige, "Small Vision Systems: Hardware and Implementation," In: Y. Shirai and S. Hirose, Eds., *Robotics Research*, Springer, London, 1998, pp. 203-212.

[19] Z. Zivkovic, "Improved Adaptive Gaussian Mixture Model for Background Subtraction," *Proceedings of the* 17*th International Conference on Pattern Recognition*, Vol. 2, 2004, pp. 28-31.

[20] P. KaewTraKulPong and R. Bowden, "An Improved Adaptive Background Mixture Model for Real-Time Tracking with Shadow Detection," In: P. Remagnino, G. Jones, N. Paragios and C. Regazzoni, Eds., *Video-Based Surveillance Systems*, Springer, Berlin, 2002, pp. 135-144.

[21] A. Prati, I. Mikic, M. M. Trivedi and R. Cucchiara, "Detecting Moving Shadows: Algorithms and Evaluation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 25, No. 7, 2003, pp. 918-923. doi:10.1109/TPAMI.2003.1206520

[22] "OpenCV Wiki, cvBlobLib," 2011. http://opencv.willowgarage.com/wiki/cvBlobsLib

[23] F. Chang, C.-J. Chen and C.-J. Lu, "A Linear-Time Component-Labeling Algorithm Using Contour Tracing Technique," *Computer Vision and Image Understanding*, Vol. 93, No. 2, 2004, pp. 206-220. doi:10.1016/j.cviu.2003.09.002

[24] A. Bhattacharyya, "On a Measure of Divergence between Two Statistical Populations Defined by Their Probability Distributions," *Bulletin of the Calcutta Mathematical Society*, Vol. 35, 1943, pp. 99-109.

[25] G. R. Bradski, "Real Time Face and Object Tracking as a Component of a Perceptual User Interface," *The Fourth IEEE Workshop on Applications of Computer Vision*, Princeton, 19-21 October 1998, pp. 214-219.

