EXTRACTING OBJECTS BY CLUSTERING OF FULL PIXEL TRAJECTORIES

Hisato Aota, Kazuhiro Ota, Yuichi Yaguchi and Ryuichi Oka
Department of Computer and Information Systems, University of Aizu, Tsuruga, Aizu-Wakamatsu, Japan

Keywords: Object tracking, Object segmentation, Pixel-wise matching, Motion feature.

Abstract: We propose a novel method for the segmentation of objects and the extraction of motion features for moving objects in video data. The method adopts an algorithm called two-dimensional continuous dynamic programming (2DCDP) for extracting pixel-wise trajectories. A clustering algorithm is applied to a set of pixel trajectories to determine objects each of which corresponds to a trajectory cluster. We conduct experiments to compare our method with conventional methods such as KLT tracker and SIFT. The experiment shows that our method is more powerful than the conventional methods.

1 INTRODUCTION

Tracking and segmentation of moving objects in video are very important for many tasks, such as video surveillance and event inference. Humans are the principal actors in daily activities, or locomotion (e.g., walking, running and standing), and so human motion is a key class within the field of computer vision. Many approaches for object tracking and segmentation have been proposed. These differ from each other primarily in the way they approach the problem. Approaches depend on such aspects as the target objects, context and environment in which tracking is performed and the end use for which tracking information is being sought. Yilmaz(Yilmaz et al., 2006) declared object tracking as the problem of estimating the trajectories of objects that are moving continuously in an image sequence from initial frame to end frame. In addition, the estimation must maintain a consistent division of objects in the continuous image sequence with segmentation. It must be also able to provide object information such as area, orientation or shape. In previous methods, continuous trajectories can only be generated by detecting objects, which makes it hard to track objects from data. The difficulty of this problem is the difficulty of fixing units of trajectories in time-sequential images. The problem becomes even more difficult when the movement trajectories are extracted from the object area with inadequate region detection. Much of the previous research concentrated on extracting objects from background areas. We felt that an alternative method based on pixel tracking would not be as restricted as the previous methods.

In our laboratory, we developed a method named two-dimensional continuous dynamic programming (2DCDP), which can define full pixel correspondences between two images. By using this approach, it became possible to extract trajectories of all the pixels in an initial frame in time-sequential images. Next, to extract effective trajectory groups from these, we used two strategies. The first is to remove the trajectories of background pixels, and the second is to cluster the trajectories that are not removed and to group similar trajectories. The goal of our work is to prove that our proposed method can track and segment moving objects without object detection or recognition, and without pre-knowledge of the state of an object, its motion or environmental factors such as people walking nonlinearly or complex human motions. To prove the method's benefits, we demonstrate an effective full-pixel matching method and connect its outputs. Experimental results indicate that it is possible to extract pixel trajectories of objects and segment them by clustering trajectories.

2 RELATED WORK

Many methods for object tracking have been proposed. Examples include tracking methods with pre-knowledge of an object's state, using Haar-like features(VIOLA, 2001), HOG features or mean-shift colour features using a statistical classifier (boosting) detector(Mochiki and Katto, 2007) (Nakagawa

Aota H., Ota K., Yaguchi Y. and Oka R. (2010).
EXTRACTING OBJECTS BY CLUSTERING OF FULL PIXEL TRAJECTORIES.
In *Proceedings of the International Conference on Signal Processing and Multimedia Applications*, pages 65-72

et al., 2009)(Takeuchi et al., 2009). In addition, Mikami(Dan et al., 2009) proposed a memory-based particle filter(M-PF), which can visually track moving objects that have complex dynamics. It provided robustness against abrupt head movements and recovered quickly from tracking failure caused by occlusions. Yamashita's(Takayoshi et al., 2008) proposed tracking method with a soft decision feature (EHOG) and online real boosting improved tracking performance in scenes with human poses and posture changes.

Deterministic methods for point correspondence using object tracking based on feature points define a cost of associating each object in frame $t-1$ to a single object in frame t by using a set of motion constraints. Rabaud(V. Rabaud, S. Belongie, 2006) developed a highly parallelized version of the KLT tracker(Shi and Tomasi, 1994) that combined graph cuts and RANSAC clustering and was able to count objects in crowds. Sugimura(Sugimura et al., 2009) proposed a human tracking method with trajectory-based clustering with a gait feature by KLT corresponding points and Delaunay triangulation. Tsuduki(Tsuduki et al., 2007)(Yuji and Hironobu, 2009) used mean-shift searching to track a point based on the information obtained by a SIFT (Lowe, 2004), and obtained a better tracking performance than KLT because a SIFT feature is invariant to changes caused by rotation, scaling, and illumination.

3 SYSTEM OVERVIEW

In this section, we describe each system for this proposed method. First, 2DCDP extracts full pixel corresponding points between frame of t and $t+1$. Then connecting corresponding points with outputs of all frames by 2DCDP, it generate full pixel trajectories. Then thresholding trajectories, it divides trajectories into two groups which is moving objects trajectories and background trajectories. Finally, divided trajectories are extracted and its of moving objects are extracted by incremental clustering (Figure1).

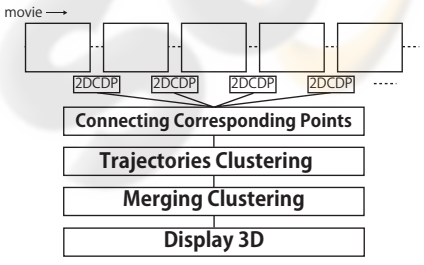

Figure 1: Proposed Work Flow.

3.1 2DCDP Algorithm

2DCDP(Yaguchi et al., 2009) is an extension of CDP(Oka, 1998) to 2D correlation, and is an effective algorithm for full-pixel matching (Figure 3). The pixel coordinates of the input image S and reference image R are defined by:

$$S \triangleq \{(i,j)|1 \le i \le I, 1 \le j \le J\} \quad (1)$$

$$R \triangleq \{(m,n)|1 \le m \le M, 1 \le n \le N\} \quad (2)$$

The pixel value at location (i,j) of the input image S_p is $S_p(i,j) = \{r,g,b\}$, and the pixel value at location (m,n) of the reference image R_p is $R_p(m,n) = \{r,g,b\}$, where r, g, and b are normalized red, green, and blue values respectively, and $(0 \le \{r,g,b\} \le 1)$. We define the mapping $R \to S$, $(m,n) \in R$ and $(\xi(m,n),\eta(m,n)) \in S$ by $(m,n) \Longrightarrow (\xi(m,n),\eta(m,n))$, set the end location for pixel matching as $\hat{i} = \xi(M,N), \quad \hat{j} = \eta(M,N)$ and the point $(\hat{i},\hat{j})$ as a nomination of the spotting point that is determined at the $M+N-1$th iteration of the proposed algorithm. Next, we set the local distance $d(i,j,m,n)$ as the difference value between $S_p(i,j)$ and $R_p(m,n)$, and set $w(i,j,m,n)$ as the weighted value of each local calculation. In this implementation, the local distance is defined as $d(i,j,m,n) = (S_p(i,j) - R_p(m,n))^2$, and weights are set to 1 for all paths (Figure 5). The accumulated local minimum $D(i,j,m,n)$ is used to evaluate the decision sequence, and is defined as:

$$D(\hat{i},\hat{j},m,n) = \frac{1}{W}\min_{\xi,\eta}\{\sum_{m=1}^{M}\sum_{n=1}^{N} w(\xi(m,n),\eta(m,n),m,n) \; d(\xi(m,n),\eta(m,n),m,n)\} \quad (3)$$

Then $\xi^*(m,n)$ and $\eta^*(m,n)$ are used to represent the optimal solutions in $\xi(m,n)$ and $\eta(m,n)$ respectively, where W is the optimal accumulated weight $W = \sum_{m,n} w(\xi^*(m,n),\eta^*(m,n),m,n)$. To ensure continuity and monotonicity, $K(m,n) = \{\xi(m-1,n),\eta(m-1,n)\}$ and $L(m,n) = \{\xi(m,n-1),\eta(m,n-1)\}$ are used to define the sets of points that are movable in the i and j directions in the input image, taken from the movements in the m and n directions in the reference image. The following equation defines the relationship between two corresponding pixels $(m-1,n-1)$ and (m,n) (see Figure 4):

$$(\xi(m-1,n-1),\eta(m-1,n-1)) \in K(m,n) \otimes L(m-1,n) \cap L(m,n) \otimes K(m,n-1) \quad (4)$$

Here, the operator $\otimes$ represents the connection between a set of points on the left and a set of points on the right. To calculate the accumulated local distance,

each accumulated local minimum $D(i,j,m,n)$ is derived from two previous accumulated local minima $D(i',j',m-1,n)$ and $D(i'',j'',m,n-1)$. In this way, we define the rank $l = m+n-1$, as shown in Figure 2 (b), to calculate the accumulated local minimum smoothly. Note that, for the accumulation and backtracking, 2DCDP selects two local paths to check the connection of the four points (m,n), $(m-1,n)$, $(m,n-1)$, and $(m-1,n-1)$ that form a quadrangle (Figure 4).

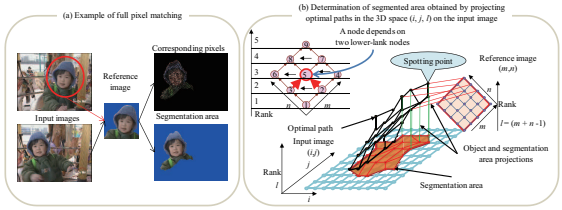

Figure 2: Full pixel matching overview. (a) An example of full pixel matching; (b) Optimal paths are able to explain a 3D space (i,j,l) in the input image. l is the rank used in the expression $l = m+n-1$.

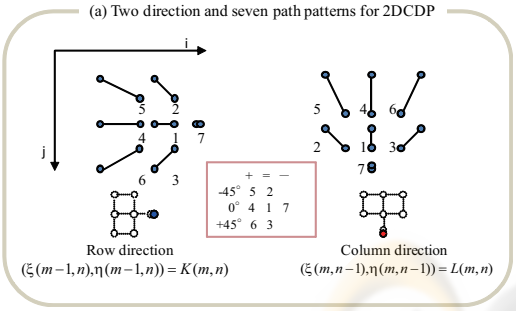

Figure 3: Two directions and seven paths for selecting optimal path to accumulate value.

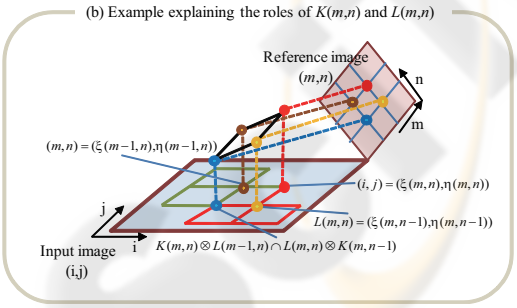

Figure 4: One case (linear matching) among the possible cases for optimal matching of local images, which include many different cases of nonlinear optimal matching of local areas.

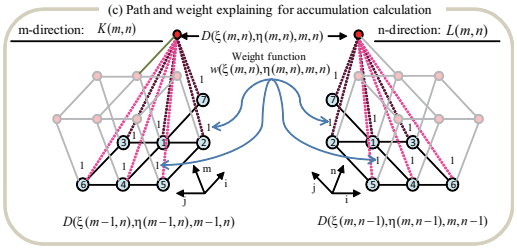

Figure 5: Seven-path map and weight. All weights w are set to 1 in implementation, but this can be changed.

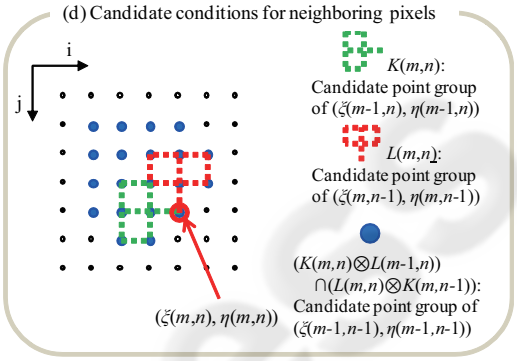

Figure 6: Each i and j direction can connect seven candidate pixels as (a) and (c). 2DCDP selects the node that has a minimal accumulation value from among these paths, but a node has depending on only two lower-rank nodes.

3.2 Connecting Corresponding Points

To extract the trajectories of the pixels in the first frame, we showed that frame (i_k, j_k) at time t and frame (m_k, n_k) at time $t+1$ are the points that correspond by 2DCDP. Therefore, the correspondence points in the following frame are derived for all the pixels in the first frame, and the following frame is processed similarly, the trajectory of the first-frame pixels in the time sequence of images can be determined. However, if both (i_a, j_a) and (i_b, j_b) at time t correspond to (m_c, n_c) at time $t+1$, they are connected. As a result, the number of trajectories decreases.(Figure7)

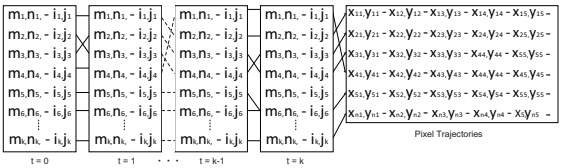

Figure 7: Connecting corresponding points.

3.3 Filtering Trajectories for Clustering

In order to erase stationary trajectories, we calculate spatial moving length for each trajectory. The length is determined by the parameter *mxy* which is defined by $mxy = \max(\max_i x_i - \min_i x_i, \max_i y_i - \min_i y_i)$, where a trajectory is represented by a location sequence,
$(x_1,y_1),(x_2,y_2),...,(x_p,y_p)$. We introduce a threshold value T. The trajectory with a larger length T is erased for clustering. The parameter T is determined by a heuristic method to each time-sequential image.

3.4 Trajectory Clustering

Incremental clustering is based on the basic sequential algorithm scheme (Gonzalez and Woods, 2001). It finds and determines trajectories that have smaller distances than the threshold value between their trajectories and the cluster-center trajectory. Then, it calculates the object-centred trajectory of each cluster. These clusters include only trajectories that match the threshold.
(1) The first cluster includes the first trajectory in the threshold ranges.
(2) The Euclidean distance between the cluster-center trajectory and the newly found trajectory is calculated. Using Euclid norm the distance between trajectories, $C(i) = (c_i x_1, c_i y_1),(c_i x_2, c_i y_2)\ldots(c_i x_k, c_i y_k)$ and $C(j) = (c_j x_1,$
$c_j y_1),(c_j x_j, c_2 y_2)\ldots(c_j x_k, c_j y_k)$, is defined by:

$$d(i,j) = ||C(i) - C(j)|| \quad (5)$$

(3) A new cluster is created if a newly found trajectory is not within a radius of the centered trajectory within which clusters were created before. Otherwise, the newly found trajectory is included in the nearest cluster.
(4) Whenever a trajectory is included in a cluster, the cluster-center trajectory is recalculated as an average of members' trajectories. This incrementally increases the size of the cluster. The algorithm repeats (1)-(4), and ends when the last trajectory is processed (Figure 8, 9).

3.5 Merging Clusters

A cluster, may be separated into two or more clusters during the distance calculation of incremental clustering, and so the clusters are merged in post-processing. If the calculated distance of the average trajectory of two clusters stays constant or decreases, the two clusters are considered to form one. k is the frame number. Using Euclid norm the distance between *cluster*1 and

$m = 1$
$C_m = \{x_1\}$
For $i = 2 to N$
- **IF**$(d(x_i, C_k) \geq \Theta)$**AND**$(m \leq q) then$
  $m = m + 1$
  $C_m = \{x_i\}$
- **Else**
  $C_k = C_k \cup \{x_i\}$
  update mean of C_k.
- **End** {IF}
-End{For}

Figure 8: Algorithm for Incremental Clustering.

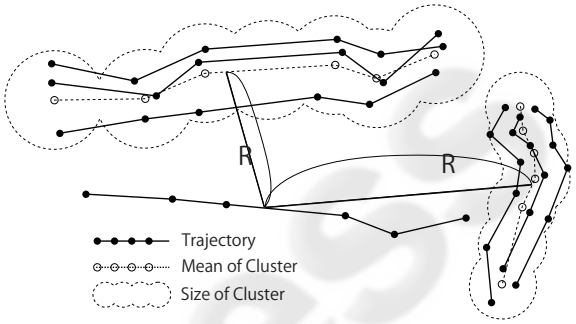

Figure 9: Incremental Clustering.

*cluster*2 average trajectories, describes $Center(i) = (c_i x_1, c_i y_1),(c_i x_2, c_i y_2)\ldots(c_i x_k, c_i y_k)$ and $Center(j) = (c_j x_1, c_j y_1),(c_j x_j, c_j y_2)\ldots(c_j x_k, c_j y_k)$, is defined by:

$$d(i,j) = ||Center(i) - Center(j)|| \quad (6)$$

4 EXPERIMENTS

4.1 Experiment Specification

4.1.1 Machine Environment

The computer used for the experiments was a Dell Precision (CPU.Xeon 3.16 GHz Dual CPU processors; memory, 64 GB DDRAM; operating system, CentOS and a 300 GB HDD). The video camera was a Cannon IXY DIGITAL 920 IS.

4.1.2 Data Sets

Our research was tested on four different datasets that we prepared. The video frame rate was 10 fps. Because of the computer memory limitation, images used in the experiments were resized to low resolution using Lanczos filter. The datasets were as follows.

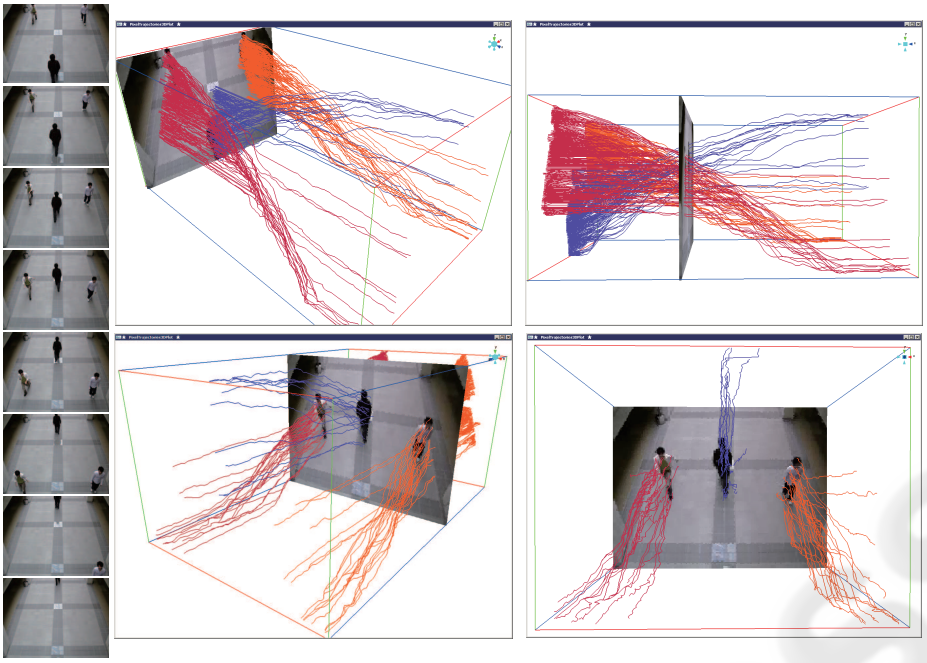

Figure 10: Result of Experiment 1. Three persons are well separated.

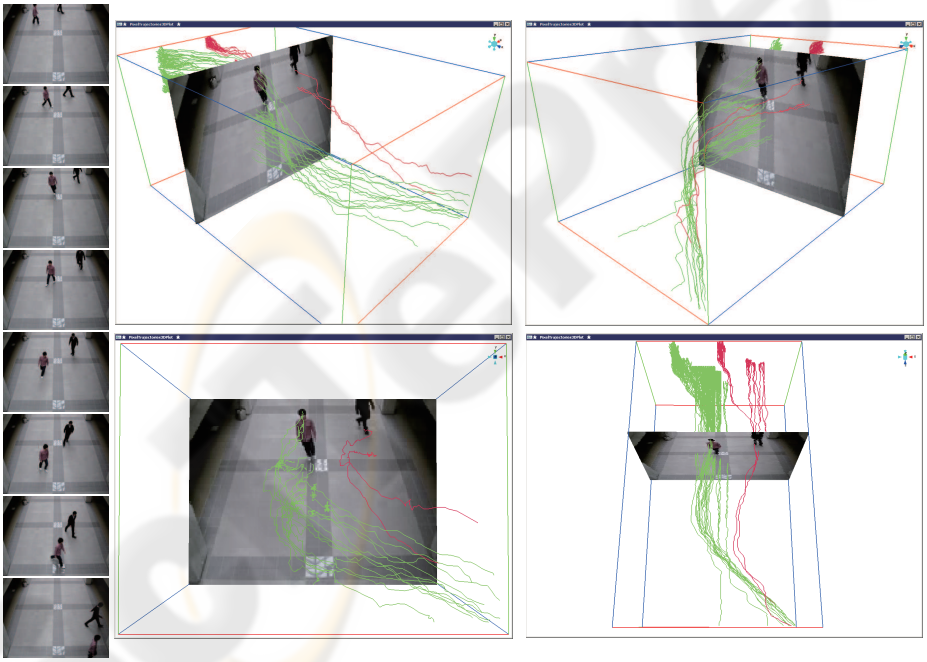

Figure 11: Result of Experiment 2. It can extract complex trajectories well.

- Three people walking in different directions. Frame count is 107 and he image size is 200*150. T is 50.
- Two people walks difference nonlinearly by each direction . Frame number is 183. Image size is 200×150. T is 30.
- Arms moving simultaneously. Frame count is 98, image size is 200×150. T is 40.
- Roller coaster with water splash. Frame count is 17. Image size is 288×192. T is 20.

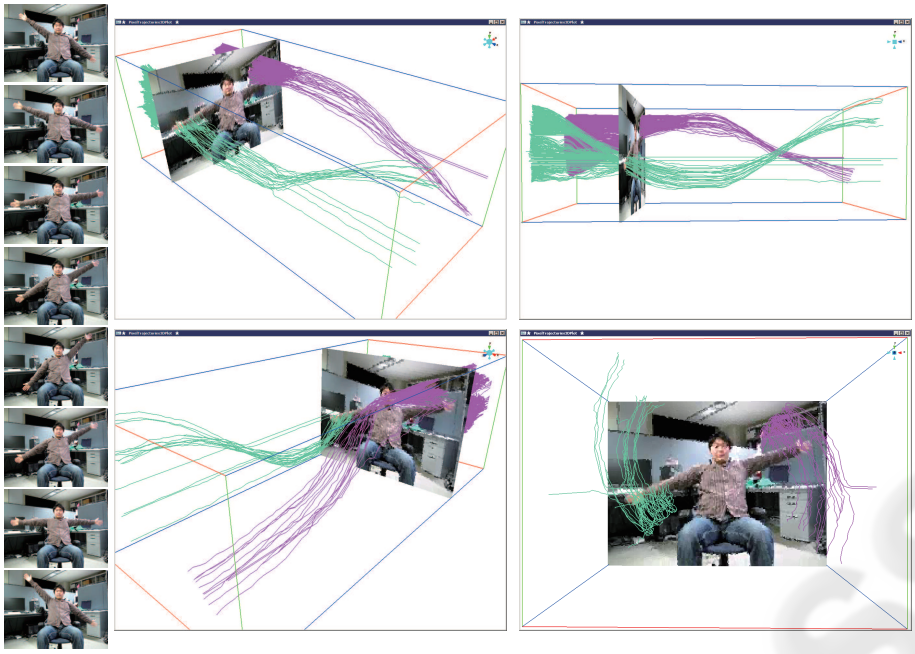

Figure 12: Result of Experiment 3. Two arms are separated.

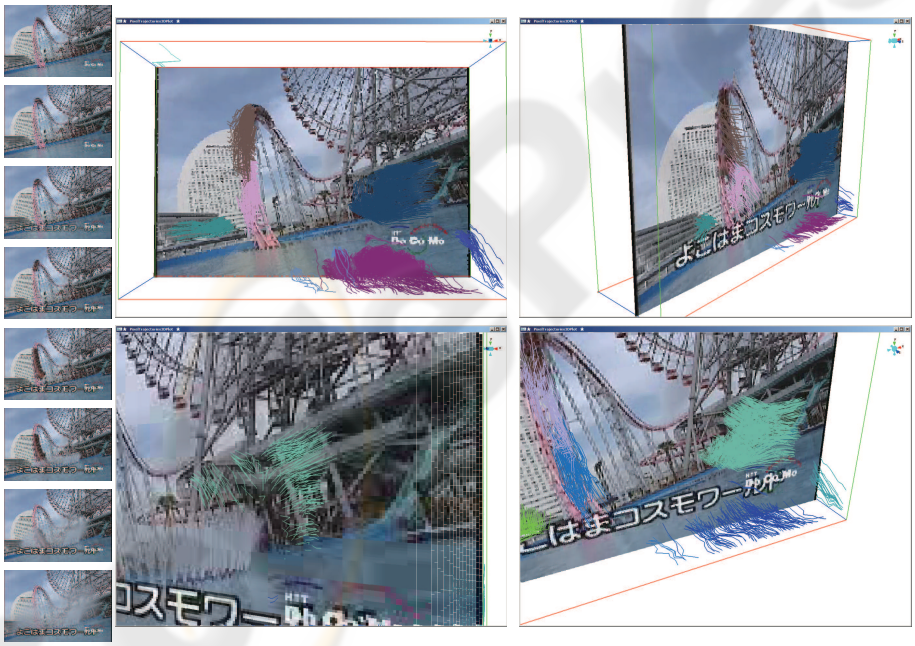

Figure 13: Result of Experiment 4. Roller coaster, water splash and water surface are well separated.

4.2 Results

Below are illustrations of the conditioned trajectories for times t_0 to t_n for each data set(Figure. 10–13). Note that, the amplitude of the z-axis has been changed for convenience. Each different color indicates a different object corresponding to a cluster.

4.3 Comparison with KLT Tracker and SIFT Tracker

Figure(14) shows the corresponding points and trajectories in 3D space in which z-axis represents time. The corresponding points, which are generated by KLT and SIFT, appear discontinuously and the num-

ber of them less than that of 2DCDP, for those reasons, trajectories of those corresponding points cannot be generated by adapting connecting system. In this comparison, we used Stan Birchfield's KLT(Stan, 2009) and Rob Hess's SIFT(Hess, 2009) implementations.

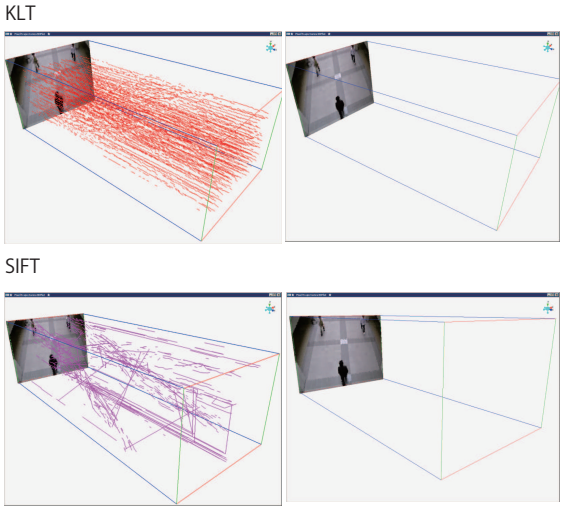

Figure 14: Comparing with KLT Tracker and SIFT. Corresponding points of KLT and SIFT are plotted to the left figure, and trajectories of those coresponding points are plotted to the right figure by adapting connecting system.

4.4 Discussion

In these experiments, density trajectories of simple or complex movements could be acquired even with low-resolution time-sequential images. The proposed method is effective even for different movements by two or more objects. Moreover, even if the movement changed during the sequence, trajectories could still be extracted. Even when objects have varying texture and edges such that features cannot be detected, the proposed method can still acquire trajectories for corresponding points. We also showed that the proposed method is more effective than methods based of feature points such as KLT or SIFT, because 2DCDP can identify many more accurate corresponding points than KLT Tracker or SIFT Flow(Figure14). Furthermore, our method also finds trajectories even for nonlinear, complex, and detailed movement such as wave splashes and the surface of the water (Figure13).

5 CONCLUSIONS

We have created a novel framework for extracting motion features from time-sequential images in a few steps. We also developed an optimal full-pixel matching method called two-dimensional continuous dynamic programming (2DCDP) that achieved image recognition that supports segmentation for free and full pixel matching. We applied 2DCDP to object tracking and utilized data for corresponding points provided by 2DCDP as input for extracting pixel trajectories and could compute density trajectories of moving objects. We were able to extract trajectories and segment objects from various type of motion with low resolution over many frames. Finally, we validated advantages of the proposed method (2DCDP + trajectory Clustering).

6 FUTURE WORK

The present approach has not considered occlusion and decreasing trajectories. Therefore, when objects appear in succession on the screen, the trajectory of a rear object is integrated into that of a front object. Because only a few pixels represent the object trajectories are integrated over time, and so the number of trajectories decreases. If the resolution of the time-sequential images increases so that the number of pixels forming the target object is increased, many trajectories can be calculated more accurately. Further, the time space is divided and clusters can re-form in each time space, and so the decrease in number of trajectories can be reduced if the clustering results are merged through the time axis. It is also necessary to remove outlier noise trajectories to improve the accuracy.

This method is applicable for background detection and subtraction from video captured by a moving camera. By analysing a vector field and extracting the majority 2DCDP vector, our method can divide background and foreground. It is able to apply a presumption of camera motion. The majority vector field is the background, which can be removed to target moving objects. Our experiments indicate that we can also consider applying the method to gesture recognition and to novel human interface such as lip reading and expression recognition.

ACKNOWLEDGEMENTS

I express my gratitude to members of the Image Processing Laboratory for their cooperation in experiments.

REFERENCES

Dan, M., Kazuhiro, O., and Junji, Y. (2009). Memory-based particle filter for face pose tracking robust under complex dynamics. *Proc. of IEEE CVPR2009*, 0:999–1006.

Gonzalez, R. C. and Woods, R. E. (2001). *Digital Image Processing Second Edition*. Prentice-Hall International.

Hess, R. (2009). Sift feature detector. http://web.engr.oregonstate.edu/∼hess/.

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *Int. Journal of Computer Vision*, 60(2):91–110.

Mochiki, R. and Katto, J. (2007). Human tracking by particle filter using haar-like feature. *Proc. of the IEICE General Conference*, 2007(2):121.

Nakagawa, H., Habe, H., and Kidode, M. (2009). Efficient prior acquisition of human existence by using past human trajectories and color of image. *Technical report of IEICE. PRMU*, 108(484):305–312.

Oka, R. (1998). Spotting method for classification of real world data. *The Computer Journal*, 41(8):559–565.

Shi, J. and Tomasi, C. (1994). Good features to track. In *Proc. of IEEE Conference on CVPR1994*, pages 593 – 600.

Stan, B. (2009). KLT: an implementation of the kanade lucas tomasi feature tracker. http://www.ces.clemson.edu/∼stb/klt/index.html.

Sugimura, D., Kitani, K. M., Okabe, T., Sato, Y., and Sugimoto, A. (2009). Tracking people in crowds based on clustering feature trajectories using gait features and local appearances. In *Proc. of MIRU2009*, pages 135–142.

Takayoshi, Y., Hironobu, F., Shihong, L., and Masato, K. (2008). Human tracking based on soft decision feature and online real boosting. In *Proc. of MIRU2008*, pages 12–19.

Takeuchi, D., Ito, Y., Yamashita, A., and Kaneko, T. (2009). Multi-viewpoint person tracking based on face detection of arbitrary pose and mean-shift algorithm. *ITE Technical Report*, 33(11):69–72.

Tsuduki, Y., Fujiyoshi, H., and Kanade, T. (2007). Mean shift-based point feature tracking using sift. *IPSJ SIG Technical Reports*, 49(6):101–108.

V. Rabaud, S. Belongie (2006). Counting Crowded Moving Objects. In *Proc. of IEEE CVPR 2006*, volume Vol. 1, pages 705–711.

VIOLA, P. (2001). Rapid object detection using a boosted cascade of simple features. *Proc. of IEEE CVPR2001*, pages 511–518.

Yaguchi, Y., Iseki, K., and Oka, R. (2009). Optimal Pixel Matching between Images. In *Proc. of PSIVT 2009*, pages 597–610.

Yilmaz, A., Javed, O., and Shah, M. (2006). Object tracking: A survey. *ACM Computing Surveys (CSUR)*, 38(4).

Yuji, T. and Hironobu, F. (2009). A method for visualizing pedestrian traffic flow using sift feature point tracking. In *Proc. of IEEE PSIVT2009*.