The 2009 IEEE/RSJ International Conference on
Intelligent Robots and Systems
October 11-15, 2009 St. Louis, USA

Entropy-based Motion Segmentation from a Moving Platform

Hyeun Jeong Min and Nikolaos Papanikolopoulos
Department of Computer Science and Engineering
University of Minnesota
Minneapolis, MN 55455
Email: {hjmin|npapas}@cs.umn.edu

Abstract— **The forward moving target segmentation from a moving platform with a pinhole camera is an important and relatively unexplored problem in visual tracking. This paper proposes a novel segmentation algorithm for extracting a moving target when using a moving platform (that follows a similar trajectory and has a camera that is not calibrated for the particular scene). When the target has unpredictable motions, we are unable to model it and the pertinent backgrounds are very different. We introduce a new entropy-based clustering algorithm in order to find a bounding box representing the target. A target model based on graph representation is used for matching the moving target. To demonstrate the robust target segmentation scheme, we apply the method to a team of miniature robots (the Explorers developed at the University of Minnesota) in real tracking missions.**

I. Introduction

Cameras provide rich information, especially in identifying or differentiating particular objects in real-world applications. This strength provides moving platforms like robots the possibility to be used in various high-level applications in search, rescue, surveillance, security, amongst others. Visual tracking is one of the popular applications for both static and moving cameras. Visual tracking with a moving camera includes the possibility that a moving platform follows a moving target. Following a target includes different scenarios based on the trajectories followed. In particular, the followers may have a similar trajectory with the target or they may follow a different one. Conventional tracking research using cameras has mainly dealt with cases where the target and the tracking system follow different trajectories. A significant difference when similar trajectories are involved is that when the following platform has almost the same movement as the moving target, the follower's camera should detect the target moving in front of it. This requires forward motion segmentation in order to extract a moving target from the following camera. However, the detection of a forward moving object has been relatively ignored in the object segmentation research field, especially when the target model is unknown.

A prevailing assumption in tracking is generally that a target has constant velocity or acceleration. This assumption in the estimation theory allows algorithms based on prediction models to effectively work in traditional tracking problems. As robots are more and more required to function in complex environments, the assumption regarding constant velocity or acceleration may not apply. For instance, when a human or a moving object suddenly stops or moves due to the requirements of its mission, prediction models are not sophisticated enough. This requires an accurate analysis of the moving target from images to provide proper information for executing the tasks without relying on prediction models.

While static cameras frequently apply background subtraction, moving cameras are unable to do this since the images grabbed from a moving camera have variant background views. Moving cameras therefore need to correct continuously changing views for object segmentation. This issue worsens when cameras move quickly but have a low frame rate. Currently applied methods for image correction are commonly based upon optical flow or matched features. Image correction is challenging since similar motions between a moving target and a moving camera result in similar changes in optical flow. This work tries to correct image differences by combining optical flow with feature matching.

After correcting the images, we first apply a Temporal Difference (TD) method on consecutive images, and then use a clustering method for the selected features on the resulting image to find candidate clusters. The clustering method saves computational time by reducing the number of candidates as compared to other pertinent algorithms. The created clusters, grouped together consist of the occupied pixels and highlight the differences between consecutive images. The clusters are parts of the target on the image. Our goal is to find these clusters and then compute a bounding box representing the target. While most image classification methods focus on distinguishing similar objects after a training process takes place, our image classification is applied to the features produced by the TD that may belong to the target, other moving objects, and possibly noise. Since any unsupervised classification requires the number of clusters, we need to find this number k. However, this is not realistic in real-world situations with an unknown number of moving objects. We, therefore, propose an entropy-based algorithm to find the best number of clusters and the other pertinent parameters. The goal in our clustering is to find 'good clusters' like those mentioned in [1] that provide a proper number of candidates for target matching.

We also describe a target modeling scheme that uses a graph representation for the extracted clusters. In the graph, nodes include feature information and edges represent the connectivity between clusters. To distinguish a target from noise or other moving objects, we apply a color histogram along with a similarity measure to determine whether or

not extracted features represent the target. For real-world experiments, we use the Explorer and the eROSI robots [2], developed in the Center for Distributed Robotics at the University of Minnesota.

II. Related Work

This section considers previous work on object segmentation from a moving camera system. When using a static camera for a moving object segmentation, it is often required to adjust to changing environmental conditions. Peterfreund [3] uses snakes to extract an object's silhouette while a static camera is utilized. Stein *et. al.* [4] use the combination of appearance cues with motion cues after over-segmentation to find object boundaries. A representative algorithm for moving objects' extraction from a static camera is background or foreground subtraction. Ke *et. al.* [5], on the other hand, propose a correlation of spatio-temporal shapes to video clips without background subtraction. The authors utilize a learning method to track motion by using the target model. A segmentation algorithm based on both pixel and region analyses for moving objects is presented by Fujiyoshi [6] to detect multiple overlapping objects from a static camera.

While there are plenty of promising algorithms on moving object segmentation from a static camera, moving object segmentation from a moving camera has been relatively unexplored. Representative methods for image correction from a moving camera use optical flow associated with the Lucas-Kanade algorithm [7] or matching features. Jung *et. al.* compensate for the movement of a camera using selected features which are represented by corners [8]. The authors deal with visual tracking from a moving camera similarly to our approach. There, however, exist several significant differences: (i) the moving object has a different trajectory than the moving camera platform, (ii) the camera's frame rate is relatively high, and (iii) the moving object exhibits sideways movements as it is seen from the camera. Talukder *et. al.* also use real-time optical flow to find moving objects between consecutive images [9]. The authors describe both monocular-based moving object detection and stereo-based moving object detection; however, their detection of sideways movement is limited. Braillon *et. al.* utilize optical flow on each pixel to find ground motion from consecutive images by using a similarity measure [10]. Du *et. al.* detect a forward moving car [11], but the work is limited to the segmentation of symmetric shapes for the target through edge detection.

III. IMAGE CORRECTION

A. Optical Flow

This section considers how the objects in the $3D$ world map on the image plane when the camera is moving due to the motion of the moving platform that carries it. The velocity when a camera or a target is moving is discussed in [12], but our interest is the case when both the camera and the target are moving. Let $T = (\nu_x, \nu_y, \nu_z)^T$ and $R = (\omega_x, \omega_y, \omega_z)^T$ denote the linear and angular velocity of a camera platform, and $P = (p_x, p_y, p_z)^T$ be the position of objects in $3D$ space. We assume $Q = (q_1, q_2, q_3)^T$ is the velocity of moving objects ($q_1, q_2,$ and $q_3 \in \Re$). Then, the velocity of moving objects w.r.t the moving camera frame is

$$\dot{P} = -T - RP + Q. \tag{1}$$

The perspective projection (x, y) on the $2D$ image plane from the position P is represented as $x = \frac{p_x}{p_z}$ and $y = \frac{p_y}{p_z}$, and the derivatives of (x, y) are

$$\dot{x} = (\frac{\dot{p_x}}{p_z} - x\frac{\dot{p_z}}{p_z}), \quad \text{and} \quad \dot{y} = (\frac{\dot{p_y}}{p_z} - y\frac{\dot{p_z}}{p_z}). \tag{2}$$

From the Eqs. (1) and (2), the derivatives of x and y on image plane are represented as follows:

$$\dot{x} = (\frac{x\nu_z}{p_z} - \frac{\nu_x}{p_z}) + (xy\omega_x - (x^2+1)\omega_y + y\omega_z) + O_x \tag{3}$$

$$\dot{y} = (\frac{y\nu_z}{p_z} - \frac{\nu_y}{p_z}) + ((1+y^2)\omega_x - xy\omega_y - x\omega_z) + O_y, \tag{4}$$

where $O_x = (\frac{q_1}{p_z} - \frac{xq_3}{p_z})$ and $O_y = (\frac{q_2}{p_z} - \frac{yq_3}{p_z})$ in the case that both the camera and the target are moving.

B. Image Correction

Image correction is the process of compensating for the differences between consecutive images, due to the camera's motion. This requires appropriate factors for image translation and scaling during the process. From Eqs. (3) and (4), we know that the derivatives of the positions on an image plane for both the moving object and the moving camera are not proportional to p_x or p_y. When we assume known linear and angular velocities for the moving camera, we can estimate the depth in the case of static objects. We can also estimate it when we have predictable motions of a moving object. However, we assume that moving objects exhibit sudden accelerations or decelerations. Under these assumptions, the equations regarding $\dot{x}$ and $\dot{y}$ are unsolvable due to the four unknown variables ($q_1, q_2, q_3,$ and p_z). Matching features promises to estimate the values of the variables $q_1, q_2,$ and q_3 to solve Eqs. (3) and (4).

For image correction at time t, we apply translation and scaling on the image. The translation and scaling factors are based on the average differences of $\dot{x}$ and $\dot{y}$. Let f_i be the matching features and $f_i(\vec{u}) = (f_i(\vec{u}), f_{i-1}(\vec{u}))^T$, where $\vec{u} = (\dot{x}, \dot{y})^T$ and $i = 1, ..., N$, and N is the number of features. We now classify the matching features into moving or static objects, and $S_i = \{\vec{u} \| \|f_i(\vec{u})\| < \frac{1}{N}\Sigma_{i=1}^{N}\|f_i(\vec{u})\|\}$ represents a set of static object features. The norm of the vector u has outstanding differences that correspond to moving features that have a different moving direction from the one associated with the camera platform. The slope for each matched features is also used. We classify matched features into 4 classes represented by $\{SE, NW, NE, SW\}$.

$$I_{comp}(\vec{x}) = A\vec{x} + \vec{z},$$

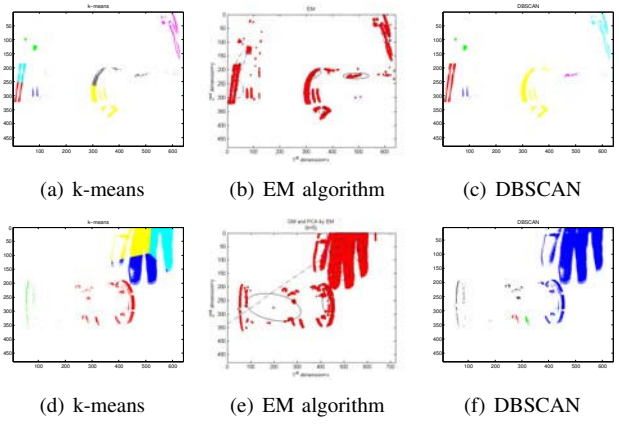
where $A = \begin{pmatrix} \delta_x & 0 \\ 0 & \delta_y \end{pmatrix}$, $\vec{z} = \begin{pmatrix} \alpha_x - \epsilon_1 \\ \alpha_y - \epsilon_2 \end{pmatrix}$, and $\vec{x} = (x \ y)^T$. The translational and scaling factors are $\alpha_x = \frac{1}{|S|}\Sigma_{i=1}^{|S|}S_i$ and $\delta_x = m_W(\vec{x}) - m_E(\vec{x})$, where $m_W(\vec{x})$ and $m_E(\vec{x})$ are the mean of the classes $\{SW, NW\}$ and $\{SE, NE\}$ on the x-axis, respectively, and $\epsilon_1, \epsilon_2 \approx 0$.

IV. OBJECT SEGMENTATION

Although we apply the image correction as in Section III, the corrected images still include differences irrelevant to the target due to various reasons such as low resolution, other moving objects, and so on. This section describes the entropy-based clustering algorithm where the goal is to find candidate clusters, characterized by a Gaussian distribution. These clusters make a target blob. Since features involve not only target features but also others such as noise or other objects, a target matching algorithm finding the correct clusters amongst the selected candidates is also applied.

A. Entropy-based Clustering

Finding clusters with high density, approximately drawn as Gaussian correlations in the Euclidean space, needs unsupervised classification algorithms such as the k-means, the mixture model based EM (Expectation Maximization), or the density-based DBSCAN (Density-Based Spatial Clustering of Applications with Noise). Prior to the explanation of the proposed clustering algorithm, we discuss the most important issues in motion segmentation. Fig. 1 shows the example images resulting from the TD. As we mentioned in Section I, a forward moving object corresponds to a certain boundary in the image differences. Figs. 1(a) $\sim$ 1(c) include a moving robot and noise, and Figs. 1(d) $\sim$ 1(f) have a moving robot with forward movement and another moving object (hand) with sideways movement. The examples show that the hand's entire shape is perfectly visible, while the moving robot is represented by a faint boundary. In Figs. 1(b) and 1(e), the dashed blue ellipses denote candidate clusters for the selected attribute. For these images, knowing the number of clusters is an essential issue in segmenting a particular target. However, although we know the number of clusters, the results from the various clustering algorithms show some differences.



(a) k-means (b) EM algorithm (c) DBSCAN

(d) k-means (e) EM algorithm (f) DBSCAN

Fig. 1. Comparative examples when different classification algorithms are applied with the same number of clusters $k = 7$ (up) and $k = 5$ (down) on the example images including other moving obstacles.

Since the DBSCAN algorithm is based on density, it can be effective to find a density-based object such as in Fig. 1(f), but it is not promising when the features' characteristics have discontinuities, only showing parts of an object or when disjoint objects have connected features. The results of both the k-means and EM algorithms vary as the choice of initial parameters, and the number of clusters are subject to change. An important issue for unsupervised classification is how to know or choose the proper number of clusters before applying the algorithm. A representative method that identifies the best number of clusters is involving human subjectivity based on a similarity measure. However, target clustering on images including noise or other obstacles is less informative than the one required by a similarity measure. Issues arise when the only information available is the position of features through the TD on images, the number of clusters is unknown, the number of clusters is flexible at each image, and features represent parts of a target.

1) Choose the principal attribute: For these issues, we consider an uncertainty measure from the concept of the entropy in information theory to find candidates representing a target and also reduce the overall number of candidates. The choice of the principal attribute can reduce the number of candidate clusters required for a matching algorithm. We want to choose these clusters in order to have less uncertainty and simultaneously reduce computational time. The amount of uncertainty represents how much information is involved in the selected attribute. If we assume that $A = \{a_1, ..., a_n\}$ is a set of attributes and $B = \{b_1, ..., b_k\}$ is a set of possible values, then the entropy of each attribute is as follows:

$$E(a_i) = -\sum_{j=1}^{k}(p_{a_i}(b_j) log_2 p_{a_i}(b_j)), \tag{5}$$

where $p_{a_i}(b_j)$ is the probability mass function of a_i for each b_j. From Eq. (5) we choose the best informative attribute with minimum entropy as in Eq. (6). Since the entropy shows the amount of uncertainty, we choose the attribute with minimum uncertainty.

$$a_{max} = \arg\max_{a_i \in A}(1 - \frac{E(a_i)}{\sum_{j=1}^{n}(E(a_j))}). \tag{6}$$

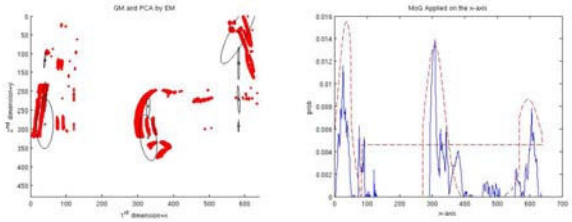
2) Find candidate clusters for each attribute: The objective of the clustering algorithm is to find candidate clusters consisting of a target's boundary, which is represented by a bounding box. From this requirement, we apply the clustering algorithm in each attribute separately to find clusters consisting of a square box. The process is started from the selected attribute in Eq. (6). This attribute is used for the principal attribute (axis) of the bounding box of a target and is changing for each image. After choosing the attribute with the minimum entropy, we first apply Gaussian Distributions (GDs) to the histogram of features from the TD result with respect to a chosen attribute. There is an iteration until the condition in Eq. (7) is satisfied. The increasing condition for the number k_{a_i} of clusters for the attribute a_i is based on the user-defined minimum threshold T_{min} of the variance (or standard deviation). This is considered to distinguish the features overlapped from other moving objects or corresponding to the inside of the target shape. As discussed before, forward movements preserve parts of the

object. The threshold can vary depending upon the object's motion. We utilize the k-means clustering algorithm for the selected number k_{a_i}, and there is an iteration until the following condition is satisfied:

$$\sigma_i < T_{min} \quad \text{for} \quad \forall i \leq k, \tag{7}$$

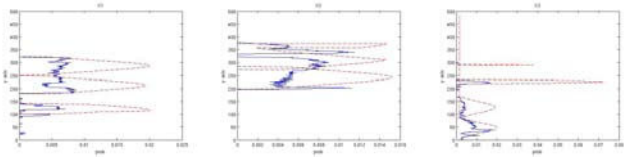
where $p_i(x) = \frac{1}{\sqrt{2\pi}\sigma_i} \exp(-\frac{1}{2\sigma_i^2}(m_i - x)^2)$, and σ_i is the Standard Deviation (SD) of the $p_i(x)$. We note that x is the variable for the histogram applied on the selected attribute (axis). For example, x and y are the variables for the x-axis and the y-axis, respectively.

After finding clusters associated with the selected principal attribute, we apply a similar algorithm to the histogram corresponding to each chosen cluster. For instance, if we find n clusters by applying for the algorithm finding the GDs on the principal attribute, we need to apply for a similar clustering algorithm for the histogram corresponding to the other attributes within each cluster. We then find the number k_{x_j} of clusters for the other attribute $x_j (x_i \neq x_j)$.



(a) The features through the TD and the clusters.

(b) The GDs applied on the x-axis.



(c) The 3 GDs on the y-axis for the 1^{st} cluster.

(d) The 2 GDs on the y-axis for the 2^{nd} cluster.

(e) The 4 GDs on the y-axis for the 3^{rd} cluster.

Fig. 2. An example to show the process of the choice of the number of clusters and the parameters. There are three candidate clusters on the x-axis.

3) Combine each cluster to derive the number of clusters and the parameters: Through the process of finding k_{a_i} and k_{b_j} for each i, we find the number of clusters along with the parameters. Eq. (8) represents the choice of the parameters resulting from the GDs on each axis. Eq. (8) corresponds to the case that the principal attribute is the x-axis, and k_x is the number of clusters on the x-axis. The parameters and the number of clusters can be used by any clustering algorithm such as the EM or the k-means algorithm as the initial information. However, we utilize the entropy to choose a principal attribute, and thus use these parameters without requiring clustering algorithms for the mixture model:

$$M = \{(m_{x_i}, m_{y_j}) | i \leq k_x, j \leq k_y(i)\} \tag{8}$$

$$k_y = \sum_{i=1}^{n} k_y(i), \tag{9}$$

where m_{x_i} and m_{y_j} are the mean of $p_i(x)$ and $p_j(y)$, respectively. Here, $|M| = k_y$, and k_y is the number of total clusters aggregated for each $k_y(i)$ which is the number of clusters on the y-axis resulting from the i^{th} cluster x_i.

Example 1: Fig. 2 shows an example of the selection of the number of clusters and the parameters. The principal attribute through the entropy computation is the x-axis in this case, and three GDs on the x-axis by Eq. (7) are found ($k_x = 3$) and shown in Fig. 2(b). For each GD on the x-axis, other GDs with respect to the histogram applied on the y-axis within the i^{th} GD are executed. The three results on each GD are shown in Figs. 2(c) $\sim$ 2(e). The 3, 2, and 4 GDs are chosen as follows: $k_y(1) = 3, k_y(2) = 2$, and $k_y(3) = 4$. The combined clusters for both attributes with the number of the clusters and the means are shown in Fig. 2(a).

B. Target Modeling

After extracting clusters, a target modeling process is required to achieve proper target segmentation. The aim of the target modeling is to choose proper clusters amongst the candidate clusters for further fitting boundaries to the target.

1) Target Representation: We deal with a graph based target representation where the graph consists of clusters as nodes and the connectivity between clusters is represented by links. The graph representation is

$$G = (V, E) \text{ and } \mathrm{G_t} = (\mathrm{V_t}, \mathrm{E}), \text{ where} \tag{10}$$

$$V = V_t \cup V_n = \{c_1, c_2, \cdots, c_k\}, \text{ and} \tag{11}$$

$$E = \{c_{ij} | c_{ij} = (c_i, c_j), 1 \leq i, j \leq k\}. \tag{12}$$

A set of vertices V is composed of extracted clusters processed from the algorithm described in Section IV-A, and V_t is a set of clusters that belong to a target along with the number of clusters (k). We discriminate the target from others by a color matching algorithm along with a similarity measure. The graph G_t represents a graph model for a target, and each node includes the following information:

- *Parameters* - The mean and variance are associated with the position of the centroid in each cluster.
- *Direction* - The information here is the location of each cluster and is represented by $D = \{E, W, N, S\}$, which are east, west, north, and south directions.
- *Connectivity* - The connectivity of edges is represented by a matrix, $C_e = \{E\}_{ij}$, where $e_{ij} (\in E) = 1$ if c_i and c_j are connected and $e_{ij} = 0$ otherwise.
- *Length* - The edges having a connection also include the distance of the two clusters.

2) Target Matching: For the target matching we choose 2 clusters for each attribute representing a target's boundary. A set of target clusters V_t is chosen from a set of nodes V by considering the position, size, and direction of clusters. We first choose the clusters from the principal attribute as the entropy indicates. If the chosen attribute is the y-axis, the clusters represent N and S. All remaining clusters are candidates for the other directions. Then a target matching algorithm through a color histogram is applied. We compute the color histogram for an extracted bounding box to represent a target in each step, and measure the similarity between the extracted targets at $t-1$ and t. The color histogram has

m-bins for the RGB values which consist of $\sqrt[3]{m}$-bins in each one of the R, G, or B values. The target model extracted from the image at time t is represented by:

$$p_t = \{p_i(x)\}_{i=1,\ldots,m}, \text{ where } \sum_{i=1}^{m} p_i(x) = 1. \quad (13)$$

We use the Bhattacharyya distance to measure the similarity of the two target models at times $t-1$ and t. It is represented by the Bhattacharyya coefficient,

$$BC(p_{t-1}, p_t) = \sum_{i=1}^{m} \sqrt{p_{t-1} p_t}. \quad (14)$$

V. EXPERIMENTAL RESULTS

For the experiments, we used images taken from the eROSI and Explorer robots in executing real-time tracking missions [13]. Each robot is equipped with a camera and differentially driven wheels. During the experiments, the target leading other moving platforms moves forward or rotates with certain linear and angular velocities. We show the results of image correction and the choice of the clusters along with the parameters used by the proposed methods.

A. Results from Image Correction

This section shows the result from the image correction by using the proposed method. In Fig. 3, (a) and (b) show the consecutive images along with the features extracted by the Harris corner operator. Fig. 3(c) shows the result of the matched features for the consecutive images (shown in Figs. 3(a) and 3(b)). We compute the translational and scaling factors from these matching features (based on the distance and slope of the differences). Figs. 3(d) and 3(e) correspond to the TD result for the original image and the corrected image, respectively.

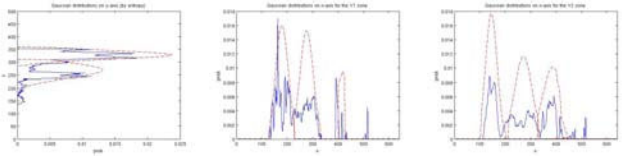
(a) At time t. (b) At time $t+1$. (c) Matched features.

(d) The original TD. (e) The corrected TD.

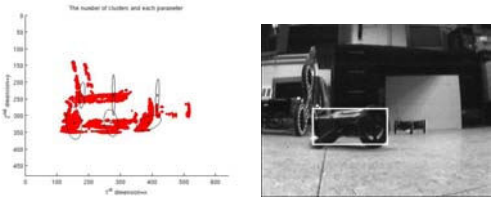
Fig. 3. The results of the image correction. The original image and the extracted corner features are shown in 3(a) and 3(b).

B. Results from Target Segmentation

This section shows the results for target segmentation by using the proposed algorithm with the target modeling method. The target moves continuously, and the following platforms equipped with a camera try to keep the target on the center of the image while they maintain a pre-defined distance between them. If the algorithm picks a small number of clusters, we may not find these corresponding to the target model. If the algorithm, however, picks a large number of clusters, it will provide many candidate clusters which impacts the computational time. Therefore, we show that our proposed algorithm can find the proper number of clusters, which means that it supplies the minimum information needed to construct a target model.



(a) The GDs applied on the histogram of the y-axis (principal axis). (b) The GDs applied on the histogram of the x-axis for the 1^{st} cluster. (c) The GDs applied on the histogram of the x-axis for the 2^{nd} cluster.



(d) The TD result and the chosen clusters. (e) The bounding box for the target.

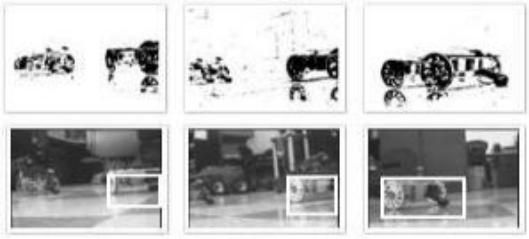
Fig. 4. The results of target segmentation from multiple moving robots.

Fig. 4 shows the result of the target segmentation from the second following robot. The principal attribute by the entropy computation (Eqs. (5) and (6)) is the y-axis in this case. The GDs on the y-axis are generated through the process of finding the number of k_y, and the results are shown in Fig. 4(a). There are 2 clusters on the y-axis ($k_y = 2$), and each cluster has found the GDs on the x-axis ($k_x(1) = 3$ and $k_x(2) = 3$). The clusters on the x-axis for each cluster from the y-axis are shown in Figs. 4(b) and 4(c). The number of the clusters and the parameters are represented on the TD features in Fig. 4(d). From these clusters (generated on the principal attribute), we can find the N and S directions, and by applying the color histogram we can find the E and W directions from the clusters on the x-axis. The bounding box for the target is shown in 4(e), and it could be segmented by the moving robots.

Fig. 5 shows the sequences of the extracted targets from the followers. In the experiment, there are three robots following each target moving in a line or a circle, where the first leader had constant linear and angular velocities ($0.02m$ and $0.02rad/sec$, respectively). The tracking task is repeated for about 30 minutes in the uncontrolled environment in our laboratory. Each image in Fig. 5(a) is grabbed from the first follower. Fig. 5(b) shows the selected features from the TD and the segmented target's boundaries (represented by the white bounding boxes) as seen from the second follower.

Fig. 6 shows the comparison between the EM and the proposed algorithm when the candidate clusters are chosen.

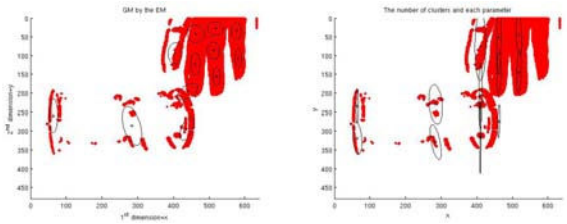
(a) From the first follower.



(b) From the second follower.

Fig. 5. A sequence of extracted features from the TD (up) and the segmented target (down).

Since most of the features from the TD are located in the hand, the clusters from the EM are more concentrated on the hand than the target. However, the result from the proposed algorithm includes enough clusters to find the target. With respect to the processing time, the proposed clustering algorithm is significantly faster than the EM. Fig. 7 shows the comparison between the proposed algorithm and the template matching applied on the sequence of images taken from the first following robot. The upper three figures are the results of the template matching for the target robot, and the lower figures are the results of the proposed method.

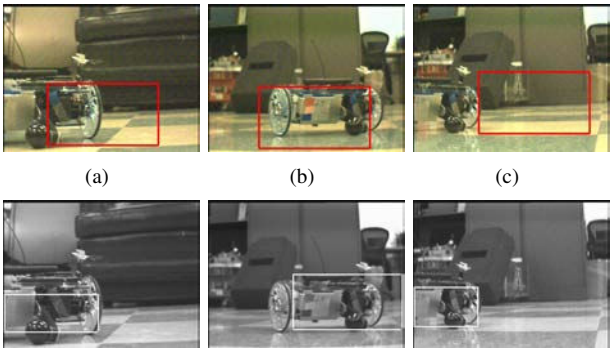


(a) The EM clustering. (b) The proposed clustering.

Fig. 6. The comparison of the clustering results between the EM with the defined 12 clusters and the proposed algorithm.

VI. Conclusions and Future Work

This paper presents a forward moving target segmentation algorithm from a camera following the target. This camera exhibits a similar trajectory. To segment a moving target, we propose an entropy-based clustering algorithm with the choice of the number of clusters and the parameters automatically done. This algorithm also reduced the number of candidates for the target model. In addition, this is applicable for moving platforms like robots having limited resources or low resolution cameras. We applied the proposed algorithm to segment each target from each robot executing a tracking task in the team. Future work can be extended to segment non-rigid body objects or to estimate 3D shapes.



(a) (b) (c)

(d) (e) (f)

Fig. 7. The comparison of the matching results between the proposed algorithm and the template matching algorithm.

VII. Acknowledgements

This material is based upon work supported under a National Science Foundation through grants #IIS-0219863, #CNS-0224363, #CNS-0324864, #CNS-0420836, #IIP-0443945, #IIP-0726109, and #CNS-0708344.

References

[1] C. Böhm, C. Faloutsos, J.-Y. Pan, and C. Plant, "Robust information-theoretic clustering," in *Proc. of the 12th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, 2006, pp. 65–75.

[2] M. Walter, M. Anderson, I. Burt, and N. Papanikolopoulos, "The design and evolution of the eROSI robot," in *IEEE Int. Conf. on Robotics and Automation*, Apr. 2007, pp. 2984–2989.

[3] N. Peterfreund, "Robust tracking of position and velocity with Kalman snakes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 21, no. 6, pp. 564–569, June 1999.

[4] A. Stein, D. Hoiem, and M. Hebert, "Learning to find object boundaries using motion cues," in *IEEE Int. Conf. on Computer Vision (ICCV)*, 2007, pp. 1–8.

[5] Y. Ke, R. Sukthankar, and M. Hebert, "Spatio-temporal shape and flow correlation for action recognition," in *IEEE Conf. on Computer Vision and Pattern Recognition*, 2007, pp. 1–8.

[6] H. Fujiyoshi and T. Kanade, "Layered detection for multiple overlapping objects," *IEICE Transactions on Information and Systems*, pp. 2821–2827, 2004.

[7] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *Proc. of Image Understanding Workshop*, 1981, pp. 121–130.

[8] B. Jung and G. S. Sukhatme, "Tracking multiple moving targets using a camera and laser rangefinder," University of Southern California, Institute for Robotics and Intelligent Systems (IRIS) Technical Report 01-397, 2001.

[9] A. Talukder, S. Goldberg, L. Matthies, and A. Ansar, "Real-time detection of moving objects in a dynamic scene from moving robotic vehicles," in *Proc. Of the 2003 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2003, pp. 1308–1313.

[10] C. Braillon, C. Pradalier, J. L. Crowley, and C. Laugier, "Real-time moving obstacle detection using optical flow models," in *Intelligent Vehicles Symposium*, 2006, pp. 466–471.

[11] Y. Du and N. Papanikolopoulos, "Real-time vehicle following through a novel symmetry-based approach," in *IEEE Int. Robotics and Automation*, 1997, pp. 3160–3165.

[12] N. Papanikolopoulos, P. Khosla, and T. Kanade, "Visual tracking of a moving target by a camera mounted on a robot: A combination of control and vision," *IEEE Transactions on Robotics and Automation*, vol. 9, no. 1, pp. 14–35, February 1993.

[13] H. J. Min, A. Drenner, and N. Papanikolopoulos, "Vision-based leader-follower formations with limited information," in *IEEE Int. Conf. on Robotics and Automation*, 2009.