Tracking of Multiple Objects under Partial Occlusion

Bing Han, Christopher Paulson, Taoran Lu, Dapeng Wu, Jian Li
Department of Electrical and Computer Engineering
University of Florida Gainesville, FL 32611
Correspondence author: Prof. Dapeng Wu,
wu@ece.ufl.edu,
http://www.wu.ece.ufl.edu

ABSTRACT

The goal of multiple object tracking is to find the trajectory of the target objects through a number of frames from an image sequence. Generally, multi-object tracking is a challenging problem due to illumination variation, object occlusion, abrupt object motion and camera motion. In this paper, we propose a multi-object tracking scheme based on a new weighted Kanade-Lucas-Tomasi (KLT) tracker. The original KLT tracking algorithm tracks global feature points instead of a target object, and the features can hardly be tracked through a long sequence because some features may easily get lost after multiple frames. Our tracking method consists of three steps: the first step is to detect moving objects; the second step is to track the features within the moving object mask, where we use a consistency weighted function; and the last step is to identify the trajectory of the object. With an appropriately chosen weighting function, we are able to identify the trajectories of moving objects with high accuracy. In addition, our scheme is able to handle partial object occlusion.

Keywords: Object tracking, KLT tracker, object detection, consistency weighted function, object occlusion

1. INTRODUCTION

Object tracking is an important research topic in computer vision. This research is to estimate and analyze the trajectory of an object in the image plane as it moves through an image sequence. Object tracking has been widely used in video surveillance, video indexing, traffic monitoring and motion based recognition.

A tracking algorithm is able to label the tracked object consistently in different frames of a video. Object tracking is not only important but also a very difficult problem because of the arbitrary object shapes, illumination changes, object occlusion, complex object shape and motion, and camera motion. Each problem needs to be solved in order to prevent failure of the tracking algorithm. In an object tracking algorithm there are generally three steps: object detection, tracking objects from frame to frame, and trajectory estimation. The primary difference of different tracking algorithms is the way the algorithms address the three steps. Moreover, different algorithms may impose various assumptions and constraints, such as smooth object motion, rigid object, and priori knowledge of object appearance.

We will briefly review some previous tracking algorithms in three categories. The first group of tracking methods is point tracking algorithms which use a set of points to represent the object being tracked. After finding the feature points, the point tracking algorithm finds the feature correspondence between frames. In order to do this, the point tracker finds the optimal solution of a cost function minimization problem formed to establish the correspondence. Next, Sethi and Jain[1] proposed a greedy approach based on proximity and rigidity constraints. Their algorithm iteratively minimizes the cost function of correspondence in two consecutive frames. However, the method is not able to handle occlusions. Another tracking algorithm proposed by Rangarajan and Shah[2] takes a greedy approach with proximal and uniformity constrains. Therefore, this algorithm is able to obtain the initial correspondences by computing optical flow of the first two frames, and this makes it capable to handle the occlusion problem. Veenman et al.[3] extended the work of Sethi and Jain, and Rangarajan and Shah by introducing a common motion constraint for point correspondence. The main constraint of Veenman's algorithm is that the algorithm assumes that the points on the same object have similar motion directions, which is not suitable for tracking isolated objects. Due to the fact that video sensors induce noise, statistical correspondence methods are proposed to solve the problem of tracking objects in noisy images by taking the measurement of uncertainties into account when estimating the object state. Kalman filter[4,5] has been extensively used for

tracking. It assumes that the state variables are normally distributed. Also, particle filters[6] are used to address other distributions. Since both Kalman filters and particle filters are not suitable for processing multiple object tracking, JPDAF[7,8] and MHT[9,10] are proposed and widely used for multiple data association.

Kernel tracking is another type of tracking algorithm, which utilize a primitive region from the object to compute the motion of the object frame by frame. There are several kernel tracking algorithms in existent and each algorithm is developed to solve one particular problem, such as translation, affine, projective, and estimation. Template matching is the most intuitive approach of kernel tracking because the algorithm finds the region of interest in the current frame and then looks in the next frame until a match is found. In order to reduce the computational cost, Schweitzer et al.[11] proposed an efficient algorithm for template matching. Their algorithm not only uses template matching, but also uses color histograms and mixture models to model the object similarities. Another algorithm that uses kernel tracking is proposed by Comaniciu and Meer[12] which uses a mean-shift tracker to track objects by using a weighted histogram computed from a circular region to represent the object. Comaniciu and Meer used another approach to track a region of interest by using the primitive information to compute the translation. In 1994, Shi and Tomasi[13] proposed the famous KLT tracker which iteratively computes the translation of an image patch centered on an interest point. The KLT tracker is simple and efficient. However, the features will be eliminated if the the sum of squares difference is substantial, because the differences indicate that the selected object is not the same object from the previous frame. Therefore, it is not suitable for tracking an object in a long sequence since this will increase the possibility of error in the video sequence. The greatest advantage of kernel tracking is the real-time applicability, and the goal of such trackers is to estimate the motion of the object, which is usually in forms of translation, affine or projective. However, the limitation is that the primitive geometric shapes for object representation may contain parts of the background. In such cases, the object motion computed by maximizing model similarity will not be accurate due to the affect of partial backgrounds in the model.

The last group of object tracking methods is silhouette tracking. In some cases, the object is too complex to be represented by a set of points or primitive geometry shapes. Therefore, silhouette tracking is used to find the complete object region in every frame with the help of object models computed from previous frames. The main difference between distinctive silhouette algorithms is the method to model the objects. The methods to model the objects can be color histograms, edges, or object contours. One example of a silhouette tracking algorithm was done by Haritaoglu et al.[14] . The algorithm models the objects with edge information from the inner regions of the objects' silhouette. In 2004, Kang et al.[15] proposed an algorithm that uses color histogram and edges to model the objects. Besides, Sato and Aggarwal[16] proposed an object tracking algorithm based on silhouette matching which uses Hough transform to compute the trajectory. Recently, contour evolution is also used by other people to track the objects in consecutive frames. Bertalmío et al. proposed an algorithm[17] that computes the motion vectors on the edge of the silhouette iteratively for each contour position using level set representation. Similarly, Mansouri[18] proposed a contour tracking algorithm based on optical flow constraint which computes the motion vectors for all points inside the silhouette. The most important advantage of silhouette tracking is the ability to track objects of various shapes. However the complexity of those algorithms is high.

In our paper, we propose a new tracking algorithm based both on template tracking and silhouette tracking. The algorithm attempts to adequately track multiple objects of arbitrary shapes in an image sequence that experiences camera motion. In order to successfully complete this task, the algorithm first segments the image to generate the binary object mask and then tracks the features inside the region. To overcome the limitations of the traditional KLT tracker, we propose a novel trajectory estimation method based on a weighting function of tracked feature motion vectors.

The paper is organized as follows: Section 1 presents a review of prior arts in object tracking. In section 2, we will introduce our tracking system. Section 3 presents the object detector and Section 4 discusses the trajectory estimation process. The experiment results are shown in section 5 and conclusions are drawn in section 6.

2. SYSTEM OVERVIEW

The objective of our tracking algorithm is to locate the moving object and compute the trajectory of the object's centroid with the given image sequence. The ability for tracking objects of arbitrary shapes and quantity, handling

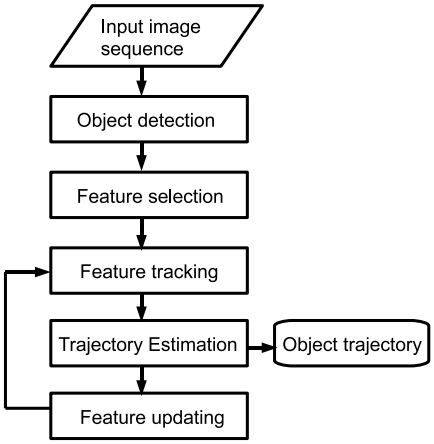

Figure 1. Flow chart of our multiple object tracking system

partial object occlusion, and allowing camera motion should also be taken into consideration. According to these requirements, there are three main steps in our proposed tracking system: object detection, feature tracking and updating, and trajectory estimation. Fig. 1 shows the system scheme.

Object detection is the first step of our algorithm and the goal of object detection is to identify the moving targets and generate a binary object mask for each object being tracked. The purpose of the object mask is to improve the accuracy of features to represent objects. Thanks to the rapid development of object detector algorithms, we are able to find the region of interest with little difficulty. Although there are different algorithms proposed for detecting different shapes of objects, it is imperative that our algorithm is able to detect moving targets with arbitrary shapes. Also our algorithm is able to handle camera motion effectively. Overall, our algorithm is an intuitive and computationally efficient algorithm based on morphology transforms.

After object detection, our algorithm models the moving objects with special image properties within the mask region. Any properties could be used to represent an object, such as edges, silhouette, colors, and primitive information. Then the algorithm uses the traditional KLT detector to select the initial features. As we know, KLT features are invariant to affine transformation, which is able to approximate the global motion caused by the camera movement.

Although we use the KLT detector to select features, the feature tracking and updating process is different from the traditional KLT tracker. The traditional KLT tracker finds features in the whole image and treats all the features the same way, i.e., no feature is important than others. However, our algorithm treats each feature differently according to the tracking performance. This will achieve better tracking performance than the traditional KLT. In our algorithm, the tracked features will be evaluated after tracking in each frame, and the "bad" features will be deleted and new features will be reselected from the rest of the object region. We also propose a weighting function for trajectory estimation, which considers both the quality of the feature and the consistence of the tracking results. As a result, the motion of the features could better represents the motion of the object.

3. OBJECT DETECTION

Object detection is a necessary step for any tracking algorithm because the algorithm needs to identify the moving object. The objective of object detection is to find out the position of the object and the region it occupies in the first frame. This task is extremely difficult due to the complex shapes of the moving targets. In order to simplify the problem, a variety of constrains are imposed to help finding the object in image plane, such as the number of objects to be tracked, types of objects, etc. Currently, there are three popular techniques used in object tracking: image segmentation, background subtraction and supervised learning algorithms.

In general, background subtraction is able to compensate for lighting changes and background clutter,and it is computationally efficient. However, most state-of-the-art background modeling methods are designed for

image sequences from fixed cameras. The image segmentation methods are able to partition the image into perceptually similar regions, but the criteria for a good partition and the efficiency are two problems that the image segmentation algorithms need to address. And the main drawback for supervised learning methods is that they usually require a large collection of samples from each object class and the samples must be manually labeled.

The goal of our object detection algorithm is to find the locations of multiple moving objects in the first few frames with a non-stationary camera. When image segmentation only utilizes spatial correlation of a single image, it is hard to detect the object region due to the different types of objects being tracked. We propose an image segmentation algorithm which considers both spatial and temporal information from the first two frames. Our algorithm obtains the temporal information by computing the optical flow from the first few images. This allows the algorithm to model the motion uniformity inside a region-of-interest. Them the output of the segmentation algorithm consists of several binary masks indicating the region each object occupies in the first image plane. The segmentation process can be expressed by equation (1)

$$\mathbf{Q}(\mathbf{x}) = \mathbf{S}(\mathbf{x}, \mathbf{I_1}) + \lambda * \mathbf{T}(\mathbf{x}, \mathbf{I_1}, \mathbf{I_2}) \,, \tag{1}$$

The final segmentation result depends on two terms, $S(x, I_1)$ and $T(x, I_1, I_2)$. $S(x, I_1)$ is a binary mask computed from primitive correlation in the first frame and $T(x, I_1, I_2)$ represents the uniformity of the motion field by optical flow from the first two frames. Also the λ is a parameter to adjust the weighing between temporal and spatial information. When λ increases, the algorithm gives more and more weight on the temporal information and when λ decreases, the algorithm cares more about the spatial information.

Computing the binary mask using spatial correlation is simple and computational efficient because there are only two steps to achieve the goal: edge detection and morphological connection. For edge detection, we use the Canny edge detector and we assume that the object to be tracked is dominant in the image plane, so we can remove the trivial edges by setting a threshold. We should notice that the detected edges are discontinuous and cannot be directly used to represent the object. Therefore, we need to connect the edges and find the closing boundary of the object. To serve this purpose, we use mathematical morphology which is a theoretical model based on lattice theory and topology. Morphological image processing is generally built on shift invariant operators, which is based on Minkowski addition. There are four basic operators in morphological image processing: opening, closing, erosion, and dilation. In our algorithm, we utilize the dilation and closing operators. With the detected edges, dilation is first applied in order to strengthen the edges as well as to connect the adjacent edges. The purpose of a subsequent closing operation is to remove the small holes inside the object mask.

After segmentation with edge detection and morphological operations, the dominant objects are marked with a binary mask. However, some areas in the background with rich texture may also be marked as a false object, due to the fact that rich texture corresponds to many edges. To solve this problem, we need to compute $T(x, I_1, I_2)$ to remove the false object regions. The correction of the temporal correlation relies on the regional variance of the optical flow field calculated from the first two frames. We use Lucas-Kanade method, which computes the optical flow using partial derivatives with respect to the spatial and temporal coordinates. The image constraint equation is given as:

$$\mathbf{I}(\mathbf{x}, \mathbf{y}, \mathbf{t}) = \mathbf{I}(\mathbf{x} + \delta_{\mathbf{x}}, \mathbf{y} + \delta_{\mathbf{y}}, \mathbf{t} + \delta_{\mathbf{t}},) \,, \tag{2}$$

Removing the higher order terms by Taylor expansion, the equation can be written as:

$$\mathbf{I_x} + \mathbf{I_y} = -\mathbf{I_t} \,, \tag{3}$$

The computed optical flow field will be segmented using the similar morphological operations and the final object regions will be generated. The segmentation result is given in Fig. 2 and Fig. 3.

Figure 2. Segmentation of the 20th frame from the 'coastguard.cif' image sequence.

Figure 3. Segmentation result of the 20th frame after correction.

4. OBJECT TRACKING

Object representation is the next task we need to handle. Most tracking algorithms address the object representation problem with some image properties, such as color, primitive region, shape, and contour, etc. Once the objects are represented by some features, the problem of tracking the objects becomes to track those predefined features or some kinds of structures. In our paper, we use the KLT features as the representation of objects and propose a new trajectory estimation algorithm with a weighting function of tracked features.

4.1 KLT feature selection and tracking

The traditional KLT tracker was first introduced by Lucas and Kanade, fully developed by Tomasi and Kanade, and explained clearly in the paper by Shi and Tomasi. With an image $\mathbf{I}$, the KLT tracker evaluates the variation for each pixel in a small neighborhood.

$$\mathbf{M} = \begin{pmatrix} \sum \mathbf{I_x^2} & \sum \mathbf{I_x I_y} \\ \sum \mathbf{I_x I_y} & \sum \mathbf{I_y^2} \end{pmatrix} \tag{4}$$

Interest points are identified by computing the quality value of each pixel by equation (5):

$$\mathbf{Q} = \det(\mathbf{M}) - \mathbf{k} * \mathbf{tr}(\mathbf{M})^{\mathbf{2}}\,, \tag{5}$$

The feature points selected by the KLT tracker are invariant to both rotation and translation. Once the object mask has been detected, the algorithm uses the KLT tracker to select and track interest points over multiple frames.

4.2 Trajectory estimation and feature update

The output of the traditional KLT tracker is a set of motion vectors of selected feature points. The motion vector represents the distance the object has traveled between two adjacent frames. In our experiment, for each object, we choose 20 to 50 feature points to represent the object. An intuitive idea to compute the trajectory of the object is to average all the motion vectors, so the mean value can be regarded as the motion of the object. Since the original KLT tracker treats all feature points equally, it can be improved by assigning different weights to motion vectors of tracked features associated with their location, quality and consistency.

Our proposed weighting function is given in equation (6)

$$\mathbf{W} = \mathbf{W_p} + \lambda_\mathbf{1}\mathbf{W_q} + \lambda_\mathbf{2}\mathbf{W_c}\,, \tag{6}$$

There are three terms in computing the weighting function. $\mathbf{W_p}$ is a Gaussian weight computed considering the position of the interest point in the object mask, expressed in equation (7)

$$\mathbf{W_p}(\mathbf{x},\mathbf{y}) = \begin{cases} \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{x^2+y^2}{2\sigma^2}} & (x,y) \in \mathbf{W} \\ 0 & elsewhere \end{cases} \tag{7}$$

$\mathbf{W_q}$ represent the weight for feature quality. For each object to be tracked, we use approximately 20 to 50 feature points to represent the object. The feature points have different quality values computed by equation 5. According to the criteria of KLT feature detector, the higher the feature quality, the more accurately the object will be tracked.

The last term in the weighting function is $\mathbf{W_c}$, which stands for the consistency of the feature over multiple frames. The original KLT feature tracker is not able to track through a long image sequence, because a number of feature points will be lost if the tracker cannot find a corresponding one after tracking each frame. In order to overcome the limitation, we propose a feature updating mechanism to find new features once many features are lost and the number of features are not enough to represent an object. By doing this, the algorithm will track a constant number of features for every frame. The consistency of feature is represented by the number of frames

Figure 4. The first frame in the test 'coastguard.cif' image sequence.

it survives, meaning that the longer the feature is present during the feature updating mechanism process, the more stable the feature is. Therefore, the probability that this feature is a good feature to represent the object is high. We use an exponential function to calculate the weight for feature consistency:

$$\mathbf{W_c} = \mathbf{e^n}\,, \tag{8}$$

where n is the number of frames the feature has been detected. The consistency is especially important when object occlusion occurs, which is discussed in the next section.

4.3 Occlusion Handling

In our object tracking scheme, the object is represented by a certain number of feature points. When occlusion happens, part of the object or the whole object is invisible. Once occlusion is occurred, the texture inside the object mask will be different compared to that of the original object. Most of the features will be lost due to the texture change, so the feature updating mechanism will automatically substitute new features selected from the current frame. Also, it is possible that the new features being tracked are not true features for the occluded object. However, these kinds of features will not contribute much to the final decision since their consistency is poor. Our trajectory weighting function ensures that the old features have a larger weight, which means the motion of the object will depend more on the consistent features. Therefore, the weighting function will eliminate the tracking error caused by object occlusion.

5. EXPERIMENTAL RESULTS

We now test our algorithm on several challenging image sequences. All sequences are at 15 fps, $352 * 288$ pixels resolution. In the result figures, we use a white line to represent the trajectory of each object we tracked.

Fig. 4 and Fig. 5 show the first and last frames in the test 'coastguard.cif' image sequence. Fig. 6 shows our tracking result of the two object in the sequence. There are two objects to be tracked in this sequence: a larger ship and a small boat. At first, the boat is in the center of the frame and the ship come into the image from the left. The camera follows the boat, so the camera moves in the image plane and the ship travels from left to right. After several frames, the two objects meet and the ship is occluded by the boat. In our experiment, we are still able to track both objects as long as the ship is not fully occluded.

Figure 5. The last frame in the test image sequence.

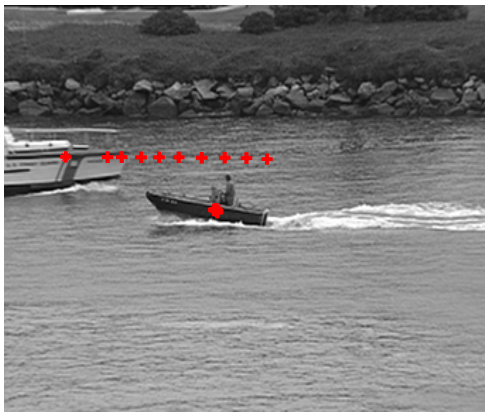

Figure 6. The tracking result of the test 'coastguard.cif' image sequence. Each '+' indicates the position of the ship in one frame.

6. CONCLUSION

In our paper, we propose a multiple object tracking algorithm with a weighting function for different features. Our approach is able to detect the object with the first frames considering both spacial and temporal information and track over multiple frames with partial object occlusion. For each object to be tracked, we use a set of KLT features to represent the object and a weighting function to balance the contribution for different features, according to their position, quality and consistency. Visualized results on several well known test sequences show the good performance of our algorithm.

Disclaimers

The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of AFRL or the U.S. Government.

Acknowledgement

This material is based on research sponsored by AFRL under agreement number FA8650-06-1-1027. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon.

REFERENCES

[1] Sethi, I. K. and Jain, R., "Finding trajectories of feature points in a monocular image sequence," *IEEE Transactions on Pattern Analysis and Machine Intelligence* **9**(1), 56–73 (1987).

[2] Rangarajan, K. and Shah, M., "Establishing motion correspondence," *IEEE Computer Society Conference on Computer Vision and Pattern Recognition* , 103–108 (Jun 1991).

[3] Veenman, C. J., Reinders, M. J. T., and Backer, E., "Resolving motion correspondence for densely moving points," *IEEE Transactions on Pattern Analysis and Machine Intelligence* **23**(1), 54–72 (2001).

[4] Broida, T. J. and Chellappa, R., "Estimation of object motion parameters from noisy images," *IEEE Transactions on Pattern Analysis and Machine Intelligence* **8**(1), 90–99 (1986).

[5] Beymer, D. and Konolige, K., "Real-time tracking of multiple people using continuous detection," *IEEE Frame Rate Workshop* (1999).

[6] Kitagawa, G., "Non-gaussian state-space modeling of nonstationary time series," *Journal of the American Statistical Association* **82**, 1032–1063 (1987).

[7] Chang, Y.-L. and Aggarwal, J., "3d structure reconstruction from an ego motion sequence using statistical estimation and detection theory," *Proceedings of the IEEE Workshop on Visual Motion* , 268–273 (Oct 1991).

[8] Rasmussen, C. and Hager, G. D., "Probabilistic data association methods for tracking complex visual objects," *IEEE Transactions on Pattern Analysis and Machine Intelligence* **23**(6), 560–576 (2001).

[9] Reid, D., "An algorithm for tracking multiple targets," *IEEE Transactions on Automatic Control* **24**, 843–854 (Dec 1979).

[10] Hue, C., Le Cadre, J.-P., and Perez, P., "Sequential monte carlo methods for multiple target tracking and data fusion," *IEEE Transactions on Signal Processing* **50**, 309–325 (Feb 2002).

[11] Schweitzer, H., Bell, J. W., and Wu, F., "Very fast template matching," *Proceedings of the 7th European Conference on Computer Vision* , 358–372 (2002).

[12] Comaniciu, D. and Meer, P., "Mean shift analysis and applications," *The Proceedings of the Seventh IEEE International Conference on Computer Vision* **2**, 1197–1203 (1999).

[13] Shi, J. and Tomasi, C., "Good features to track," *IEEE Computer Society Conference on Computer Vision and Pattern Recognition* , 593–600 (Jun 1994).

[14] Haritaoglu, I., Harwood, D., and David, L. S., "W4: real-time surveillance of people and their activities," *IEEE Transactions on Pattern Analysis and Machine Intelligence* **22**(8), 809–830 (2000).

[15] Kang, J., Cohen, I., and Medioni, G., "Continuous tracking within and across camera streams," *IEEE Computer Society Conference on Computer Vision and Pattern Recognition* **1**, I–267–I–272 (June 2003).

[16] Sato, K. and Aggarwal, J. K., "Temporal spatio-velocity transform and its application to tracking and interaction," *Computer Vision and Image Understanding* **96**(2), 100–128 (2004).

[17] Bertalmío, M., Sapiro, G., and Randall, G., "Morphing active contours," *IEEE Transactions on Pattern Analysis and Machine Intelligence* **22**(7), 733–737 (2000).

[18] Mansouri, A.-R., "Region tracking via level set pdes without motion computation," *IEEE Transactions on Pattern Analysis and Machine Intelligence* **24**(7), 947–961 (2002).