Robust Feature-based Object Tracking

Bing Han, William Roberts, 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

Object tracking is an important component of many computer vision systems. It is widely used in video surveillance, robotics, 3D image reconstruction, medical imaging, and human computer interface. In this paper, we focus on unsupervised object tracking, i.e., without prior knowledge about the object to be tracked. To address this problem, we take a feature-based approach, i.e., using feature points (or landmark points) to represent objects. Feature-based object tracking consists of feature extraction and feature correspondence. Feature correspondence is particularly challenging since a feature point in one image may have many similar points in another image, resulting in ambiguity in feature correspondence. To resolve the ambiguity, algorithms, which use exhaustive search and correlation over a large neighborhood, have been proposed. However, these algorithms incur high computational complexity, which is not suitable for real-time tracking. In contrast, Tomasi and Kanade's tracking algorithm only searches corresponding points in a small candidate set, which significantly reduces computational complexity; but the algorithm may lose track of feature points in a long image sequence. To mitigate the limitations of the aforementioned algorithms, this paper proposes an efficient and robust feature-based tracking algorithm. The key idea of our algorithm is as below. For a given target feature point in one frame, we first find a corresponding point in the next frame, which minimizes the sum-of-squared-difference (SSD) between the two points; then we test whether the corresponding point gives large value under the so-called Harris criterion. If not, we further identify a candidate set of feature points in a small neighborhood of the target point; then find a corresponding point from the candidate set, which minimizes the SSD between the two points. The algorithm may output no corresponding point due to disappearance of the target point. Our algorithm is capable of tracking feature points and detecting occlusions/uncovered regions. Experimental results demonstrate the superior performance of the proposed algorithm over the existing methods.

Keywords: Object Tracking, Sum-of-Squared Difference (SSD), Harris Criterion, Tomasi-Kanade's model, Occlusion.

1. INTRODUCTION

Accurate tracking of feature points over image sequences is a critical and essential process for vision systems ranging from unmanned aerial vehicles to medical devices. Study has shown that establishing correspondence between image patches, through correlation-based measures or sum-of-squared differences (SSD), can achieve effective feature tracking.

A feature-based tracking algorithm must first assume a form to model an object's motion. Traditionally, motion has been represented as translational, which indeed proves reliable for small, linear movements. When tracking over a longer image sequence, however, more complex models are needed as geometric deformations of objects become significant. Shi and Tomasi noted in [1] that translational models are poor representations when an object has undergone an affine transformation. In their approach, translational models are used for tracking, which provides higher reliability and accuracy over smaller inter-frame camera motions. To monitor the image sequence for occlusions via comparisons between the first and current frames, affine models are utilized, which serve to better account for longer object deformations that potentially could have occurred.

Once a motion model has been established, an object can be tracked over frames through SSD or correlation methods. To ensure that the accuracy of tracked features is maintained and to reject outliers that arise through

occlusion, a successful tracking algorithm must adopt a criterion through which to monitor a feature's quality. In [1], Shi and Tomasi used a measurement of features' rms residue between the initial and current frame, which they described as "dissimilarity", to quantify the change in a feature's appearance over frames. Jin, Favaro, and Soatto [2] proposed a rejection rule based on a normalized cross-correlation function which furthermore compensated for differences in intensity variation among the patches of interest. In order to achieve stability and robustness against occlusions, several other tracking methods [3], [4] have used Kalman filters to smooth the object trajectory and monitor the object's motion.

Feature correspondence presents a challenging problem for feature-based object tracking, as ambiguity often arises when a feature point in one image has many similar points in another image. To alleviate ambiguity, algorithms often perform an exhaustive search or compute pixel correlations over large windows. As a result, the computational complexity of the algorithms is considerably increased. In contrast, Tomasi and Kanade in [5] use small windows to track the translational motion of objects by minimizing a residue error, thus reducing complexity. However, their approach may lose a significant percentage of tracked points over longer sequences.

To prevent this loss of feature points, our algorithm couples the approach taken by Tomasi and Kanade with an SSD criterion. Furthermore, to ensure robustness but still avoid computational complexity, our method employs the combined motion model described in [1].

Depending on the application, not all feature points in an image sequence necessarily provide "useful" information to the researcher. To obtain segmentation of objects and the image background, our algorithm uses the greedy learning procedure developed by Williams and Titsias in [6]. Their learning model sequentially extracts object parameters from a sequence using a robust statistical approach, and thus avoids the complexity that often results from segmentation algorithms. The approach taken by Williams and Titsias stems from an earlier model described by Jojic and Frey in [7], which conversely learns objects simultaneously using a variational method. By first segmenting the objects from the sequence, our algorithm is able to filter out undesirable features and thus ensure that all of the tracked points are indeed meaningful.

The remainder of this paper is organized as follows. In Section 2, we introduce the translational and affine image motion models, and Shi-Tomasi's [1] combined motion model. Section 3 explains the object segmentation method used in our algorithm. In Section 4, we present our point feature tracking algorithm. Section 5 shows the experimental results and a performance evaluation of the proposed algorithm versus previous approaches. Finally, we conclude the paper and describe future work in Section 6.

2. IMAGE MOTION MODELS

In order to track a point feature in a video sequence, an image motion model should first be defined. The image motion model relates information about the image's deformation. Generally, either a translational model or an affine motion model is used.

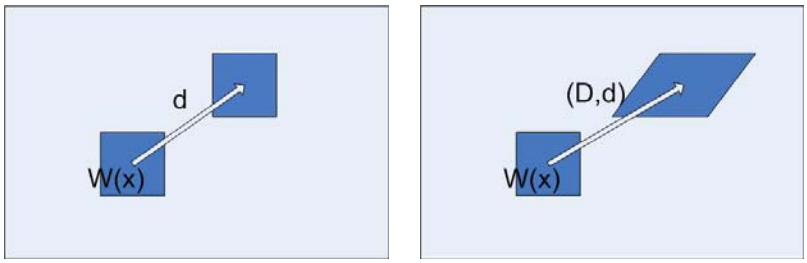
2.1. Translational motion model

Image intensity patterns change in an intricate way. In the translational motion model, however, we only concentrate on the "regions" of the image that can be modeled as undergoing a linear transformation. Translational motion models assume that each point in the window undergoes identical motion. Translational models for point feature tracking are easy to implement and computationally costless. Fig. 1(a) shows the translational motion of a fixed window $\mathbf{W(x)}$.

Let $\mathbf{x}$ be the central point of the concentrated "region". Let $\mathbf{W(x)}$ be the window around $\mathbf{x}$. Let the function $\mathbf{h}$ describe the transformation of the image motion. We have

$$\mathbf{h}(\tilde{\mathbf{x}}) = \tilde{\mathbf{x}} + \Delta\mathbf{x}, \tag{1}$$

where, $\forall\tilde{\mathbf{x}} \in \mathbf{W(x)}$, $\Delta\mathbf{x} \in \mathbb{R}^2$. This model is valid for portions of a scene that are flat and parallel, and whose movements are parallel, to the image plane. The approximation applies only locally in space and in time. Although coarse, the model is at the core of most feature matching or tracking algorithms due to its simplicity and the efficiency of its implementation.



(a) (b)

Figure 1. (a) Translational motion deformation of local domain **W(x)**, and (b) affine motion deformation of local domain **W(x)**.

2.2. Affine motion model

When variations of the displacement vector are noticeable even within the small windows used for tracking, translational models fail to describe the event, as different motions occur within the same window. An affine motion model, shown in Fig. 1(b), is thus introduced.

More precisely, an affine motion model is defined as:

$$\mathbf{h}(\tilde{\mathbf{x}}) = \mathbf{D}\tilde{\mathbf{x}} + \mathbf{d}, \tag{2}$$

where $\mathbf{D} \in \mathbb{R}^{2\times 2}$ and $\mathbf{d} \in \mathbb{R}^2$. $\mathbf{D}$ is a deformation matrix and $\mathbf{d}$ is the translation of the center of the window. The quality of this estimation depends on the size of the feature window, the texture of the image within it, and the amount of camera motion between frames. This model serves as a good approximation for small planar patches parallel to the image plane that undergo an arbitrary translation and rotation about the optical axis, and only a modest rotation about an axis orthogonal to the plane. The affine model represents a convenient tradeoff between simplicity and flexibility. In the prevailing algorithms, an affine motion model is used to identify the good features.

2.3. Combined motion model

Both translational models and affine models have limitations, and thus a combination of the two models will be established. In Shi and Tomasi's paper [1] , they monitor the quality of image features during tracking by measuring features' rms residues between the first and the current frame. When the dissimilarity becomes too large, the feature will be abandoned. For this case, affine models, rather than translational, prove more suitable. When the window is small, the matrix $\mathbf{D}$ is harder to estimate, as the variations of motion within the window are smaller and therefore less reliable.

Generally, smaller windows are preferable for tracking because they are less likely to straddle a depth discontinuity. Whereas an affine motion is used for comparing features between the first and the current frame, a translational model is preferable during tracking for its improved accuracy and reliability.

3. OBJECT SEGMENTATION

In this section, we will briefly describe the approach to object segmentation taken by Titsias and Williams in [6]. For our algorithm, object segmentation will be used to isolate features of interest for tracking.

3.1. Segmentation model

The goal of the algorithm is to learn appearance-based models to describe the background and foreground objects in an image. The variable $\mathbf{f}$ will be used to describe the appearance of the foreground object, where, for brevity, we will assume in this explanation that only one object is present. A vector of probabilities, $\boldsymbol{\pi}$, will be assigned

to the image, where $\pi_j \approx 1$ when pixel j is part of the foreground object and $\pi_j \approx 0$ otherwise. The background appearance will be described by $\mathbf{b}$. A single frame can then be described by the following mixture distribution:

$$p(\mathbf{x}) = \prod_{p=1}^{P} [\pi_p p_f(x_p; f_p) + (1 - \pi_p) p_b(x_p; b_p)], \tag{3}$$

where P denotes the number of pixels in the image, $p_f(x_p; f_p) = N(x_p; f_p; \sigma_f^2)$, $p_b(x_p; b_p) = N(x_p; b_p; \sigma_b^2)$, and $N(x; \mu; \sigma^2)$ represents a Gaussian distribution with mean μ and variance σ^2.

To account for the object motion over frames, we let j_f and j_b denote the object and background transformation, respectively. If, assuming only translations, we then let the matrix $\mathbf{T}_{j_f}$ represent the transformation j_f and $\mathbf{T}_{j_b}$ represent j_b, then:

$$\begin{aligned} p(\mathbf{x}|j_f, j_b) \quad = \quad & \prod_{p=1}^{P} [(T_{j_f}\boldsymbol{\pi})_p p_f(x_p; (T_{j_f}\mathbf{f})_p) + \\ & (\mathbf{1} - T_{j_f}\boldsymbol{\pi})_p p_b(x_p; (T_{j_b}\mathbf{b})_p)]. \end{aligned} \tag{4}$$

The parameters needed to model the scene, given by $\theta = \{\mathbf{f}, \boldsymbol{\pi}, \mathbf{b}, \sigma_f^2, \sigma_b^2\}$, can be obtained by maximizing the likelihood $L(\theta) = \sum_{n=1}^{N} \log p(\mathbf{x}^n|\theta)$ using the EM algorithm, where j_f and j_b are the unknown parameters. To reduce complexity, Williams and Titsias extract the background and foreground appearances sequentially.

3.2. Learning the background

To obtain the background of the sequence, the foreground mask $\boldsymbol{\pi}$ is effectively set to zero so that (4) becomes:

$$p(\mathbf{x}) = \prod_{p=1}^{P} [p_b(x_p; b_p)]. \tag{5}$$

The log likelihood of the background is then:

$$L_b = \sum_{n=1}^{N} \log \sum_{j_b=1}^{J} P_{j_b} p(\mathbf{x}^n|j_b), \tag{6}$$

which is then maximized using the EM algorithm to obtain j_b.

3.3. Learning the foreground

After finding the background, the foreground mask $\boldsymbol{\pi}$ in (4) is then allowed to take on non-zero values. As a direct maximization over the new likelihood would require a search over $J_f \times J_b$ possibilities, Williams and Titsias instead use the constrained EM algorithm presented in [8]; the computational complexity is reduced to J_f by using the background transformation already obtained. By introducing a distribution $Q^n(j_f, j_b)$ and using Jensen's inequality, they produce a lower bound on the likelihood:

$$\begin{aligned} F \quad = \quad & \sum_{n=1}^{N} \sum_{j_b j_f} Q^n(j_f|j_b) Q^n(j_b) \{\log[P_{j_f} P_{j_b} \\ & \times \quad \prod_{p=1}^{P} p(x_p^n|j_b, j_f)] - \log[Q^n(j_f|j_b) Q^n(j_b)]\}, \end{aligned} \tag{7}$$

which can be tightened by setting $Q^n(j_b, j_f) = P(j_b, j_f|\mathbf{x}^n)$ for every image $\mathbf{x}^n$. Maximization can then be performed, where in the expectation step F is maximized with respect to the Q^n distribution and in the maximization step F is maximized with respect to the object parameters $\{\mathbf{f}, \boldsymbol{\pi}, \sigma_f^2\}$. The update equations are omitted here, but can be found in [6].

Figure 2. An example of the response of the Harris feature detector using $\mathbf{p} = 6$.

4. TRACKING ALGORITHMS

Robustness and accuracy are two important criterions for a tracking algorithm. In addition, feature tracking demands a computationally efficient approach. In this section, we will first describe several basic feature-based algorithms. Then, we will present our new algorithm, which will seek to achieve better efficiency and performance than previous approaches.

4.1. Feature selection

In the first step of feature selection, candidate features in one or more frames from the given sequence of images are selected. No feature-based algorithm can work unless good features can be identified first. For the case of point features, a popular algorithm is the well-known Harris corner detector. The quality of a point with coordinates $\mathbf{x} = [x, y]^T$, as a candidate feature, can be measured by

$$C(\mathbf{x}) = \det(\mathbf{G}) + k \times \text{trace}^2(\mathbf{G}), \tag{8}$$

computed on the window $\mathbf{W}(\mathbf{x})$. In the equation, $\mathbf{G}$ is a 2×2 matrix which depends on $\mathbf{x}$, given by

$$\mathbf{G} = \begin{pmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{pmatrix}, \tag{9}$$

where I_x and I_y are the gradients obtained by convolving the image I with the derivatives of a pair of Gaussian filters and k is a constant parameter that can be chosen by the user.

A point feature is selected if C($\mathbf{x}$) exceeds a certain threshold τ. In this way, a feature is selected only if the window contains "sufficient texture". In order to achieve tracking efficiency, we do not want the features to be concentrated in a small region within the whole image. However, in some specific feature-rich regions, more than one feature may be selected out, which does not prove efficient for tracking. To mediate this problem, we define a minimal space $\mathbf{p}$ between two selected features, such that a candidate feature point should be sufficiently distanced from other selected points.

Fig. 2 and Fig. 3 show two sets of initial features selected from the same image. A small minimal space $\mathbf{p}$ results in more features which serve to describe the same region. Generally, we choose $\mathbf{p}$ to be 5 or 6 in order to achieve a better tradeoff between robustness and efficiency.

4.2. Sum-of-squared-difference criterion

Under the assumption of the simple translational deformation model, tracking a point feature $\mathbf{x}$ is the process of finding out the location $\mathbf{x} + \triangle\mathbf{x}$ on the frame at time $t + \tau$ whose window is most similar to the window $\mathbf{W}(\mathbf{x})$.

A common way of measuring similarity is by using the sum-of-squared-difference(SSD) criterion. The SSD criterion compares the image window $\mathbf{W}$ centered at the location (x, y) at time t and other candidate locations

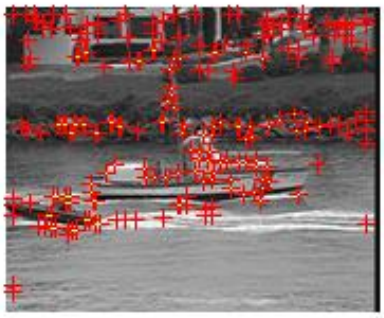


Figure 3. An example of the response of the Harris feature detector using $\mathbf{p} = 3$.

$(x + dx, y + dy)$ on the frame at time $t + dt$, where the point could have moved between the two frames. The displacement $\mathbf{d}$ is obtained by minimizing the SSD criterion

$$E_t(dx, dy) \doteq \sum [I(x + dx, y + dy, t + dt) - I(x, y, t)]^2, \tag{10}$$

where the subscript t indicates the translational deformation model.

One alternative way to compute the displacement $\mathbf{d}$ is to evaluate the function at each location and choose the one that gives the minimum error. This formulation is due to Lucas and Kanade [9], and was originally proposed in the context of stereo algorithms. The algorithm was later refined by Tomasi and Kanade [5] in a more general feature-tracking context.

4.3. Pyramidal decomposition

Multi-scale decomposition is a widespread tool in image processing. The typical recursive form of the algorithm, which decomposes signals into information at different levels, leads to large improvements in computational efficiency. Simoncelli and Freeman [10] proposed a steerable pyramid for efficient and accurate linear decomposition of an image into scale and orientation. Bouguet developed an algorithm to implement pyramidal image scales of the Lucas-Kanade feature tracker.

The proposed tracking scheme is implemented in a multi-scale fashion by constructing a pyramid of images through smoothing and downsampling of the original image. For instance, let I^0 be the original image and I^{L-1} represent the image at level $L - 1$. The L^{th} level image is then defined as follows:

$$\begin{aligned} I^L(x, y) = & \frac{1}{4} I^{L-1}(2x, 2y) + \\ & \frac{1}{8}(I^{L-1}(2x - 1, 2y) + I^{L-1}(2x + 1, 2y) \\ & + I^{L-1}(2x, 2y - 1) + I^{L-1}(2x, 2y + 1)) + \\ & \frac{1}{16}(I^{L-1}(2x - 1, 2y - 1) + I^{L-1}(2x + 1, 2y + 1) \\ & + I^{L-1}(2x - 1, 2y - 1) + I^{L-1}(2x + 1, 2y + 1)). \end{aligned} \tag{11}$$

For a given feature central point $\mathbf{x}$, its corresponding coordinates on the pyramidal images are defined by

$$\mathbf{x}^L = \frac{\mathbf{x}}{2^L}. \tag{12}$$

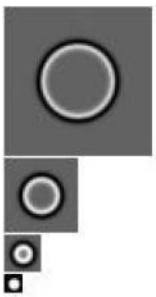


Figure 4. A 3-level, $k = 1$ steerable pyramid(non-oriented). Shown are the three bandpass images at each scale and the final lowpass image [10].

Then we could compute each motion direction vector $\mathbf{d}^L$ on each level of pyramidal images. Finally, we could sum up all levels of motion vector $\mathbf{d}$ as

$$\mathbf{d} = \sum 2^L \mathbf{d}^L. \tag{13}$$

A key problem to any feature tracker is the tradeoff between accuracy and robustness. Intuitively, accuracy requires a small tracking window in order to preserve details in the image while robustness prefers a bigger intergration window to handle larger motions. The advantage of pyramidal implementation is that, while each motion vector $\mathbf{d}^L$ is obtained by way of smaller integration windows, the overall displacement vector $\mathbf{d}$ can account for larger pixel motions, thus achieving both accuracy and robustness.

Fig. 4 shows a 3-level steerable pyramid decomposition of a disk image, with $k = 1$, where k represents the number of orientation bands. Fig. 5 shows a 3-level steerable pyramid decomposition with $k = 3$. The filters are directional second derivatives oriented at $\theta \in \{-2\pi/3, 0, 2\pi/3\}$.

4.4. Improved algorithm

From our experimental data, we find out that both minimal SSD criterion or Tomasi-Kanade's algorithm [5] will lose track of more than half of the initial selected features after 15 to 30 frames. In order to increase the robustness of the tracking algorithm, we combine the two tracking methods and propose a more robust feature-based tracking algorithm.

First, we will use Harris's criterion to select the set of initial candidate features ($\mathbf{S}_1$) in the first frame of the video sequence. Each feature is then tracked using two methods of detection. Minimal SSD criterion is applied in the first step to find the most similar region of the target feature in the time adjacent frame. If the quality $C(\mathbf{x})$ of the region $\mathbf{W}(\mathbf{x} + \mathbf{d})$ exceeds the chosen threshold τ_H, it will be updated to the set of tracking features. However, when the motion becomes more complicated or after the feature is tracked through a long sequence of frames, the most similar region that fit minimal SSD criterion may not be a good feature to keep on tracking. In this case, the feature will be declared lost of tracking in the former SSD criterion algorithm. In the experiment, more than ten percent of features are lost because the quality degrades through the tracking sequence. In order to solve this problem, we will introduce Tomasi-Kanade's [5] algorithm as a complement to minimal SSD criterion.

Once the quality $C(\mathbf{x})$ of the tracked feature is lower than the threshold τ_H, we will define another set of candidate features ($\mathbf{S}_2$) to be

$$\mathbf{S}_2 = \mathbf{S}_3 - \mathbf{S}_1, \tag{14}$$

where $\mathbf{S}_3 = \{\mathbf{x}_1, ...\mathbf{x}_j, ...\mathbf{x}_m\}$ are the potential features in the tracking window $\mathbf{W}_2(\mathbf{x})$. After subtracting the points that are already in the set of initial candidate features, other features will be calculated to find the one that gives the minimal squared difference. If the similarity between the two regions is smaller than the threshold τ_E, it will also be taken as a successfully tracked feature.

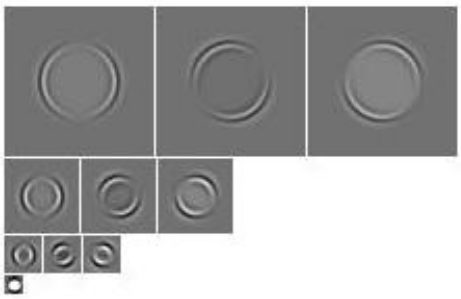


Figure 5. A 3-level, $k = 3$ steerable pyramid(second derivative). Shown are the three bandpass images at each scale and the final lowpass image [10].

In this way, we can increase the robustness when tracking point features along a long image sequence. The algorithm could also perform well when the motion between two time adjacent images is not purely translational, such as through scattering or scaling.

5. EXPERIMENT RESULTS

5.1. Algorithm

The full algorithm is presented in Fig. 6. In the code, the feature quality, $C(\mathbf{x})$, is calculated by Harris's criterion. In the event that the object's motion is known to be translational and void of scaling, the use of the purely translational model yields improved results versus the model which accounts for scaling.

5.2. Results

We have tested the algorithm on video clips consisting of various types of motion, including translational, affine, and scaling deformations. Features are identified in the initial frame as the candidate features to be tracked through the whole image sequence. The performance of the algorithm depends on the content of the test video.

In the experiment, parameters could be adjusted according to the type of motion in the test image sequence. For each of our simulations, the threshold for selection is $\tau_S = 0.01$, the threshold for SSD criterion in the tracking process is $\tau_E = 0.1$, and the threshold for Harris's criterion in the tracking process is $\tau_H = 10$.

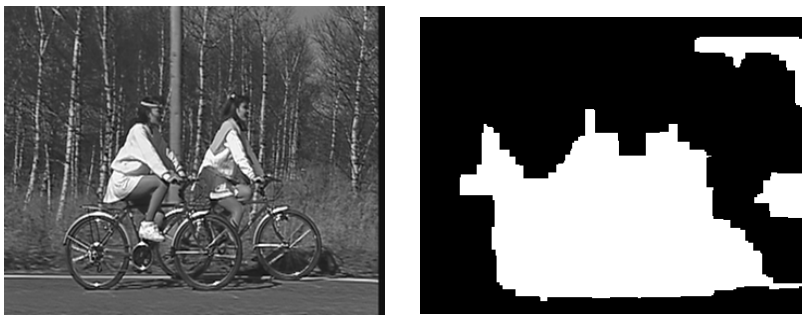
To evaluate the proposed algorithm, we applied the algorithm developed by Tomasi and Kanade in [5] and the improved algorithm which combines the SSD criteria and Tomasi-Kanade's algorithm.

Fig. 7(a) shows the initial frame of a sequence displaying two bikers traveling at approximately the same speed. The sequence undergoes translational motion, affine motion, and scaling. In Fig. 7(b), the object mask created using the greedy learning approach [6] is shown, where noise in the mask has been removed by way of morphological filters. As evidenced, the mask successfully recovers both bikers, but also erroneously contains some of the background on the right side of the frame. After applying the mask to the initial frame of the sequence, the combined tracking algorithm and Tomasi-Kanade's algorithm were applied to the 50-frame video. The results from the first and last frame are shown in Fig. 8(a) and Fig. 8(b), respectively, using the combined method. Furthermore, Fig. 9 illustrates the number of features tracked during each frame of the sequence using the combined algorithm, shown with the open circles, and the Tomasi-Kanade approach, shown with the filled squares. The combined algorithm improves the number of successfully tracked features by over 10 percent.

Fig. 10(a) displays the initial frame of a sequence containing a coastguard boat on a river. The object mask created [6] is shown in Fig. 10(b). Unlike the previous video, the coastguard sequence contains only translational motion. The first frame of the tracked sequence, after applying the mask, is shown in Fig. 11 for the combined algorithm. However, as evidenced in Fig. 12, the combined algorithm proves equivalent to the results obtained using the Tomasi-Kanade approach. Thus, as expected, the combined algorithm only offers improved performance for image sequences containing affine or scaling deformations.

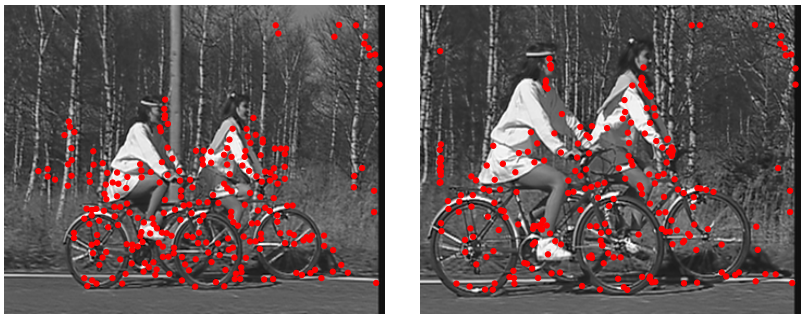
```
1)  function FeatureSelection(I)
2)      Choose window W1(x)
3)      Compute the quality C(x) of each pixel using eq 8
4)      Choose threshold τs
5)      Sort x in decreasing order of C(x)
6)      if C(xi) > τs
7)          if xi − xj <minimal space(j < i)
8)              S1 ← xi
9)          end
10)     end
11) end function
12) function FeatureTracking(x')
13)     Build a pyramid of k levels of images
14)     for level = 1 : k
15)         Compute d^k = −G^{-1}b for the image pair (I1^k, I2^k)
16)         Move the window W(x) by 2d^k through warping the next level
        image I2^{k-1}(x)
17)         Update the displacement d ← d + 2d^k and the index k ← k - 1
18)     end
19)     evaluate the quality of the feature by Harris's criterion defined by
        equation 8
20)     if C(x) > τE
21)         update xi ← xi+1
22)     else
23)         Find all the candidate features in the tracking window W2(x)
        on image I2
24)         Calculate and compare SSD for each candidate feature and the
        target feature, find the one gives minimal SSD
25)         if MSSD < τE
26)             update xi ← xi+1
27)         else
28)             returnlost
29)         end
30)     end
31) end function
```

Figure 6. Combined feature-based tracking algorithm.



(a) (b)

Figure 7. (a) 1^{st} frame in the bike sequence. (b) Mask for bike sequence created using [6].



(a) (b)

Figure 8. Image in tracked bike sequence: (a) 1^{st} frame. (b) 50^{th} frame.

6. CONCLUSION

In this paper, we have proposed a new algorithm for feature tracking based on SSD criterion and Tomasi-Kanade's algorithm. To isolate features of interest, we applied a method for object segmentation developed by Williams and Titsias. The algorithm has proven computationally inexpensive and robust to various types of object motion.

For many computer vision systems, an algorithm that operates in real-time is generally preferable. In the future, we will seek to adopt a method of segmenting objects that can perform in real-time.

Acknowledgement

This work was supported in part by the Air Force Research Laboratory under grant FA8650-06-1-1027.

REFERENCES

1. J. Shi and C. Tomasi, "Good features to track," in *Proceedings of Computer Vision and Pattern Recognition (CVPR 1994)*, pp. 593–600, (Seattle, USA), June 1994.
2. P. F. H. Jin and S. Soatto, "Real-time feature tracking and outlier rejection with changes in illumination," in *Proceedings of International Conference on Computer Vision (ICCV 2001)*, pp. 684–689, July 2001.
3. M. W. H. T. Nguyen and R. V. D. Boomgaard, "Occlusion robust adaptive template tracking," in *Proceedings of International Conference on Computer Vision (ICCV 2001)*, pp. 678–683, July 2001.
4. J. G. Legters and T. Young, "A mathematical model for computer image tracking," in *Proceedings of Pattern Analysis and Machine Intelligence (PAMI 1982)*, pp. 583–594, Nov 1982.
5. C. Tomasi and T. Kanade, "Factoring image sequences into shape and motion," in *Proceedings of the IEEE Workshop on Visual Motion*, pp. 21–28, (Princeton, USA), Oct 1991.
6. C. K. I. Williams and M. K. Titsias, "Fast unsupervised greedy learning of multiple objects and parts from video," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, pp. 79–87, 2004.
7. N. Jojic and B. Frey, "Learning flexible sprites in video layers," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. I:199–206, (Kauai, Hawaii), 2001.
8. R. M. Neal and G. E. Hinton, "A view of the em algorithm that justifies incremental, sparse and other variants," in *Learning in Graphical Models*, pp. 355–368, Kluwer Academic, 1998.
9. B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *Proceedings of Image Understanding Workshop*, pp. 121–130, April 1981.
10. E. P. Simoncelli and W. T. Freeman, "The steerable pyramid: A flexible architecture for multi-scale derivative computation," in *Proceedings of International Conference on Image Processing (ICIP 1995)*, pp. 444–447, Oct 1995.

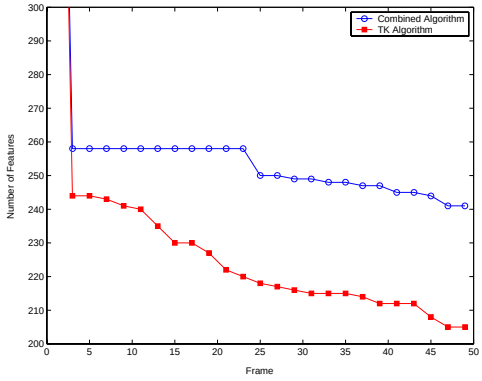


Figure 9. Number of features selected from the bike sequence using the combined algorithm and the Tomasi-Kanade(TK) algorithm.

(a) (b)

Figure 10. (a) 1^{st} frame in the coastguard sequence. (b) Mask for coastguard sequence created using [6].

Figure 11. 1^{st} tracked frame in the coastguard sequence using the combined algorithm.

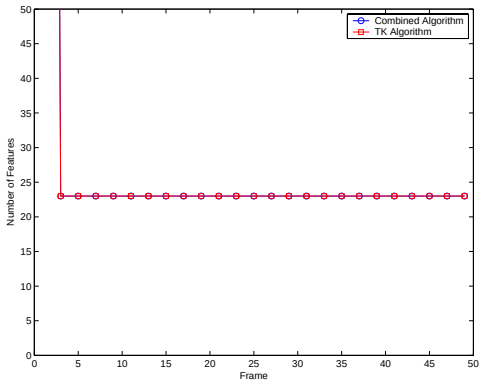


Figure 12. Number of features selected from the coastguard sequence using the combined algorithm and the Tomasi-Kanade (TK) algorithm.