TRACKING THROUGH CHANGES IN SCALE

Shawn Lankton[1] *James Malcolm*[1] *Arie Nakhmani*[2] *Allen Tannenbaum*[1,2]

[1]Georgia Institute of Technology, Atlanta, Georgia
[2]Technion-Israel Institute of Technology, Haifa, Israel

ABSTRACT

We propose a tracking system that is especially well-suited to tracking targets which change drastically in size or appearance. To accomplish this, we employ a fast, two phase template matching algorithm along with a periodic template update method. The template matching step ensures accurate localization while the template update scheme allows the target model to change over time along with the appearance of the target. Furthermore, the algorithm can deliver real-time results even when targets are very large. We demonstrate the proposed method with good results on several sequences showing targets which exhibit large changes in size, shape, and appearance.

Index Terms— Object tracking, template update, scale space, registration

1. INTRODUCTION

This paper addresses the problem of tracking objects that change drastically in size over time. When objects move closer or farther from the camera, significant changes in size, shape, and intensity profile occur, as demonstrated in Figure 1. Methods that do not take changes in target appearance into account can not accurately maintain track in these cases.

Several techniques attempt to address this problem using the popular the mean-shift framework. Collins creates an additional feature space based on target scale characteristics and solves for scale parameters and translation parameters simultaneously [1]. Peng *et al.* and Qian *et al.* adjust the window size and kernel bandwidth of the tracker based on estimations of target scale in successive frames [2, 3]. In these methods, localization is based on an unchanging intensity histogram usually taken from the first frame of the video sequence. When the target is very far, its estimated histogram may be very different from its histogram when the target is closer. Because these methods do not account for this, they can lose track as the appearance changes.

Template tracking is another approach that addresses tracking targets of variable size [4, 5, 6]. The goal of these methods is to register a template image onto the current frame to determine movement. When registration parameters allow scaling, it is possible for the template tracker to follow the changing size of the target. However, this method is defeated if the object changes significantly in appearance as it changes in size.

Based on a review of the literature, there appear to be two sources of error as the target is tracked. *Spatial drift* is the change in the model such that the model and target are misaligned. *Feature drift* is the change of target appearance as it diverges from the appearance of the model over time.

We propose a method that deals with these two sources of error separately. Spatial drift is prevented by registering the previous target representation to the current frame and registering the previous frame to a periodically updated *key model*. This accounts for frame-to-frame movement and incorporates the influence of a stable model to minimize transient affects. Feature drift is accounted for by updating the key model in a way that limits spatial drift and allows smooth feature changes over time. This update allows the method to track features which are specific to the target's recent appearance and is robust against large changes in scale.

Fig. 1. First, middle, and last frame from the LEAVES sequence. Notice the drastic change in size, shape, and appearance as the leaves move from a small blob in the distance to fill the entire frame.

2. PROPOSED ALGORITHM

The proposed method is divided into two main components: template matching and template update. Template matching, described in Section 2.1, prevents spatial drift in the target model by performing two registration procedures. First, translation parameters $\mathbf{p}_1$ are determined, which register the image I at the current frame to the model, $\hat{M}_t$ obtained from the previous frame. This reduces frame-to-frame spatial drift. Next, a second set of translation parameters $\mathbf{p}_2$ is found which align $\hat{M}$ to the key model $\tilde{M}$. This reduces spatial drift in the model. The two sets of translation parameters are summed to determine the final translation parameters $\mathbf{p}$ that specify the location of the target and the next model $\hat{M}_{t+1}$

Template update, described in Section 2.2, is executed every N frames, and allows the key model $\tilde{M}$ to change over time in a manner that limits spatial drift yet allows a smooth change in target appearance. Every N frames, the key model is replaced by the best model from the a set of the previous N models $\{\hat{M}_i\}_{i=1}^{N}$. The diagram in Figure 2 illustrates the algorithm with template matching on the left, and template update on the right.

2.1. Template matching

The goal of template matching is to estimate the transformation parameters $\mathbf{p} = \{p_1 \dots p_n\}$ that best align a reference template $T(\mathbf{x})$ to an image $I(\mathbf{x})$, where $\mathbf{x} = (x, y)^T$ is a vector of template coordinates. Borrowing the notation of Baker and Matthews [6], we assume that $\mathbf{W}(\mathbf{x}; \mathbf{p})$ warps reference coordinates according to $\mathbf{p}$.

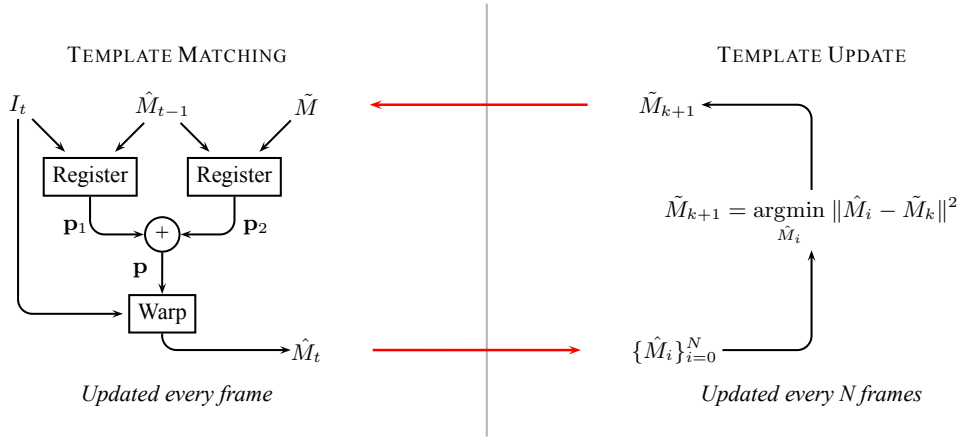

Fig. 2. Flow chart representation of the proposed method

This problem may then be formulated as a sum of squared differences (SSD),

$$\mathbf{p}^* = \underset{\mathbf{p}}{\operatorname{argmin}} \sum_{\mathbf{x}} [I(\mathbf{W}(\mathbf{x};\mathbf{p})) - T(\mathbf{x})]^2, \quad (1)$$

where the sum is computed over template coordinates $\mathbf{x}$. We begin by reforming this expression with the introduction of a step variable $\Delta\mathbf{p}$ and use a truncated Taylor expansion to separate it from the warp:

$$= \sum_{\mathbf{x}} [I(\mathbf{W}(\mathbf{x};\mathbf{p}+\Delta\mathbf{p})) - T(\mathbf{x})]^2 \quad (2)$$

$$\approx \sum_{\mathbf{x}} [I(\mathbf{W}(\mathbf{x};\mathbf{p})) + \nabla I \cdot \frac{\partial \mathbf{W}}{\partial \mathbf{p}} \cdot \Delta\mathbf{p} - T(\mathbf{x})]^2. \quad (3)$$

We now take the gradient with respect to $\Delta\mathbf{p}$ and rearrange terms to yield the step solution:

$$\Delta\mathbf{p} = \sum_{\mathbf{x}} H^{-1} \left[\nabla I \cdot \frac{\partial \mathbf{W}}{\partial \mathbf{p}}\right]^T [T(\mathbf{x}) - I(\mathbf{W}(\mathbf{x};\mathbf{p}))] \quad (4)$$

where H is the $n \times n$ *Hessian* matrix:

$$H = \sum_{\mathbf{x}} \left[\nabla I \cdot \frac{\partial \mathbf{W}}{\partial \mathbf{p}}\right]^T \left[\nabla I \cdot \frac{\partial \mathbf{W}}{\partial \mathbf{p}}\right]. \quad (5)$$

If we examine the Hessian matrix, we notice that the terms that most effect its computation are those with large image gradient ∇I. These coordinate locations $X \subseteq I$ contain the most information affecting the transformation parameters $\mathbf{p}$. As suggested by Dellaert and Collins [7], using only this subset of pixels in the computations can increase speed by orders of magnitude with little loss in accuracy. Following this technique, at the start of each iteration, we compute (4-5) and necessary derivatives only on a dominant subset X. Procedure 1 details the iterative estimation of $\mathbf{p}$ as described in [4, 5].

This process is well suited for arbitrary transformation parameters, but we find that allowing more complex transformations than translation introduces unnecessary degrees of freedom and can lead to inaccurate registrations. Hence, we take $\mathbf{p}$ to be translation in the x and y direction, and

$$\mathbf{W}(\mathbf{x};\mathbf{p}) = \mathbf{x} + \mathbf{p} \quad (6)$$

Procedure 1 Registration

Determine dominant subset $X \subseteq I$
repeat
  Compute $I(\mathbf{W}(\mathbf{x};\mathbf{p}))$
  Compute residual $[T(\mathbf{x}) - I(\mathbf{W}(\mathbf{x};\mathbf{p}))]$
  Compute $\nabla I \cdot \frac{\partial \mathbf{W}}{\partial \mathbf{p}}$ and form the Hessian H as in (5)
  Solve for $\Delta\mathbf{p}$ as in (4)
  $\mathbf{p} \leftarrow \mathbf{p} + \Delta\mathbf{p}$
until $\mathbf{p}$ has converged

2.2. Template Update

The goal of template update is to allow the appearance of the reference image to change in order to reduce feature drift while at the same time preventing spatial drift from entering the system. To achieve this, the key model $\tilde{M}$ is updated every N frames with the procedure outlined in Figure 3.

During the template matching step, we collect the past N models, denoted as $\{\hat{M}_i\}_{i=1}^N$. The new key model, $\tilde{M}_{k+1}$ is chosen as the best representative of this set as determined by a matching error, e

$$e(\hat{M}, \tilde{M}) = \|\hat{M} - \tilde{M}\|^2 \quad (7)$$

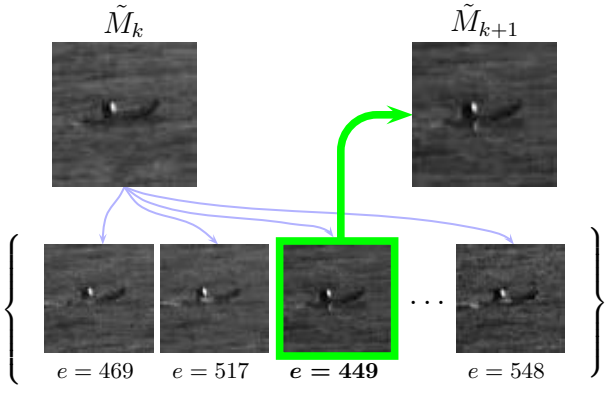

Fig. 3. We update the key model $\tilde{M}_k$ periodically to prevent feature drift. The new key model $\tilde{M}_{k+1}$ is selected by choosing a model from the set $\{\hat{M}_i\}_{i=1}^N$ that has the lowest error e when compared to the previous key model $\tilde{M}_k$

$$\tilde{M}_{k+1} = \underset{\hat{M}_i}{\operatorname{argmin}}\, e(\{\hat{M}_i\}_{i=1}^{N}, \tilde{M}_k) \quad (8)$$

The set of $\{\hat{M}_i\}_{i=1}^{N}$ represents optimally registered image patches from the previous N frames, but many will exhibit image noise or slight misalignments that make them poor choices for a key model. We choose the new key model to be the candidate with the lowest matching energy so that the key model changes smoothly without introducing spatial or feature drift.

Note that the size of the models never varies as the models are updated. Although we maintain track of the target as it changes in scale, we do so without changing the size of our representation. This is accomplished by tracking features available at the current scale of the target. Hence when the target is small, the key model may include the entire target, but if the target grows such that it is larger than the model, the model holds finer details of the target's appearance that allow the algorithm to maintain its track point.

3. EXPERIMENTS

We applied this tracker to several sequences chosen to demonstrate robustness against large changes in appearance of the target especially due to scale change. Additionally, we discuss the parameters that can affect system performance. Videos from all tracking sequences can be found at the author's web site.[1]

3.1. Tracking Results

First, consider the LEAVES sequence shown in Figure 4. Here we use the proposed method to track a small bunch of leaves. In the first frame, when the track is initialized, the leaves appear very small, and little detail can be seen. Over the course of the video, the camera approaches the leaves and their size and detail level increase dramatically. Three full frames are shown with the tracking result on the first row of Figure 4. Rows two and three show models, $\hat{M}_t$ from selected frames. Notice how the information in the model adapts to represent features specific to the current scale of the target. The proposed method maintains a stable track point on the front-most leaf throughout the sequence.

In Figure 5 tracking results for the VEHICLE sequence are shown. This tactical imagery shows a vehicle being tracked starting from a very far distance and closing quickly until the target becomes very large in the frame. In the initial frames, the target is barely visible, and can only be seen in the zoomed-in models shown in the second and third rows of the figure. The model adapts to the changing size of the target, and the sequence is accurately tracked despite the large scale and appearance change.

The BOAT sequence demonstrates the robustness of the technique against factors besides scale change in Figure 6. The size change of the target is not as drastic as those in Figures 4 and 5, but this sequence exhibits poor resolution, sporadic illumination changes, significant image noise, and significant target variability due to crashing waves and the varying pose of the boat. Figure 6 shows three full size frames with tracking result, and eight models from throughout the sequence.

3.2. Parameters

The proposed method has two parameters which control its behavior: percentage of pixels used as the dominant subset $X \subseteq I$ in the registration procedure (Section 2.1) and the length, N of the history used for key model updates (Section 2.2).

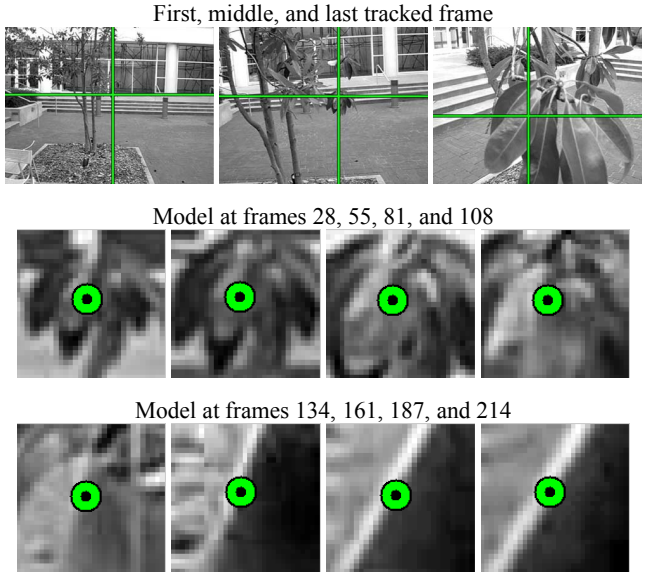

Fig. 4. Tracking results on the LEAVES sequence demonstrating ability to track through large scale change. First row: selected full-size frames. Second and third row: target models from selected frames throughout the sequence.

First, we discuss the pixel percentage used in registration. Using fewer pixels increases the speed of the algorithm significantly. It can also increase the accuracy of registration in some cases. In the BOAT sequence, the background water is noisy and low contrast. If 100% of the pixels are used in registration for this sequence, results are not as accurate due to many ambiguous background pixels being included in the registration. We use 25% of pixels with good results. Alternatively, in the LEAVES sequence the background is very high-contrast. In this case using only a small percentage of the pixels can cause the template matching step to favor registration to the background instead of the target. In this imagery, we use 100% of the pixels to ensure the object is tracked despite the high contrast background.

The second parameter of interest is the length N of the model history used during the template update step. This parameter must be set to reflect how quickly the target is expected to change in appearance. For instance, in the LEAVES sequence, the target undergoes a large appearance and scale change very quickly. For this experiment we use a relatively short history of $N = 5$. In the VEHICLE and BOAT sequences, the target changes more slowly, and a history of 20 frames is used.

3.3. Efficiency

Finally, we point out the efficiency inherent in this algorithm. Other methods that are capable of tracking objects though scale changes may continuously change the size of their tracking window to include the entire object. Hence, the number of pixels analyzed in order to localize the target increases with target size. This increase can lead to slower frame rates when the target appears large. Because our algorithm uses a fixed window size and updates the features tracked within that window, the frame rate remains roughly constant despite

[1] `http://www.shawnlankton.com`

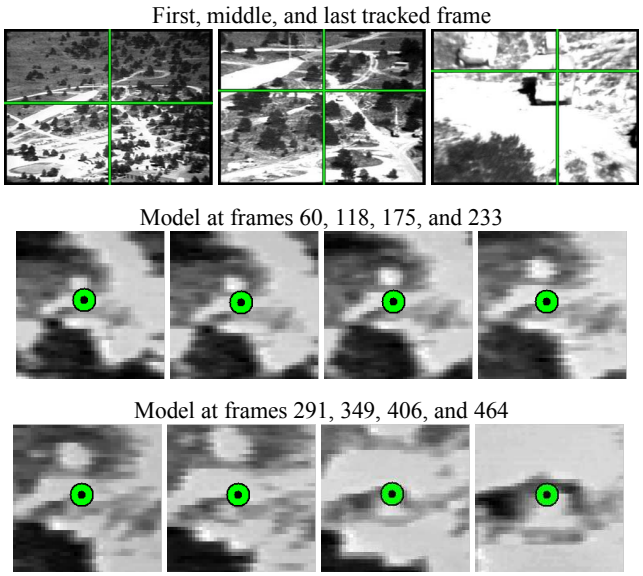

Fig. 5. Tracking results on the VEHICLE sequence demonstrating ability to track through large scale change. First row: selected full-size frames. Second and third row: target models from selected frames throughout the sequence.

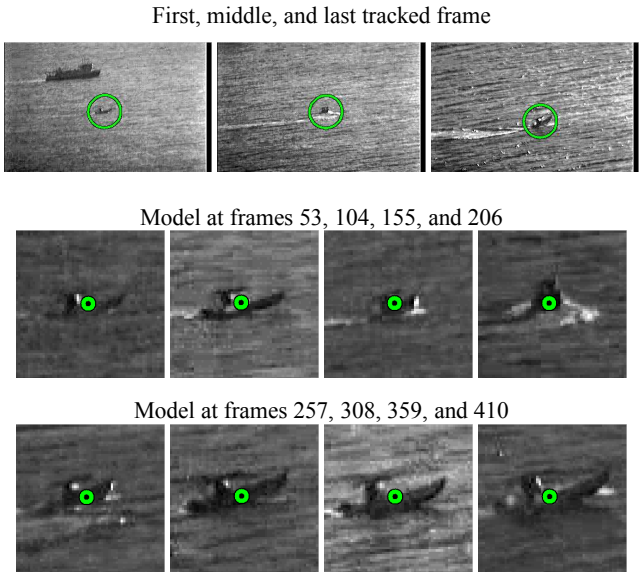

Fig. 6. Tracking results on the BOAT sequence demonstrating ability to track through noise, illumination changes, and target variability. First row: selected full-size frames. Second and third row: target models from selected frames throughout the sequence.

changes in target size. Furthermore, the simplicity of the algorithm allows it to run at real time speeds. In our prototype Matlab implementation we achieve speeds between 17 and 26 Hz. Table 1 shows the frame rates achieved for each of the three experiments shown.

Table 1. Frame rates achieved during experiments

Sequence	Figure	Frame Rate
LEAVES	4	17.54 Hz
VEHICLE	5	25.46 Hz
BOAT	6	24.62 Hz

4. CONCLUSION AND FUTURE WORK

We have shown a novel tracking algorithm capable of tracking targets through significant changes in size and appearance. By using a two-phase template matching scheme and an intelligent template update procedure, the system can prevent spatial drift and feature drift in order to maintain accurate track. Furthermore, the efficiency of the algorithm makes it useful for real-time applications.

Future work may include using robust estimators rather than sum of squared differences to determine correlation between models and targets as well expanding the current approach into a particle filtering framework to make the system more robust against extremely erratic target motion.

Acknowledgements

This work was supported in part by grants from NSF, AFOSR, ARO, MURI, as well as by a grant from NIH (NAC P41 RR-13218) through Brigham and Women's Hospital. This work is part of the National Alliance for Medical Image Computing (NAMIC), funded by the National Institutes of Health through the NIH Roadmap for Medical Research, Grant U54 EB005149. Information on the National Centers for Biomedical Computing can be obtained from `http://nihroadmap.nih.gov/bioinformatics`. Allen Tannenbaum is also with the Department of Electrical Engineering, Technion, Israel where he is supported by a Marie Curie Grant.

References

[1] Robert Collins, "Mean-shift blob tracking through scale space," in *CVPR*, 2003, vol. 2, pp. 234–240.

[2] Ning-Song Peng, Jie Yang, and Jia-Xin Chen, "Kernel-bandwidth adaptation for tracking object changing in size," in *ICIAR*, 2004, vol. 3212, pp. 581–588.

[3] Huimin Qian, Yaobin Mao, Jason Geng, and Zhiquan Wang, "Object tracking with self-updating tracking window," in *PAISI*, 2007, vol. 4430, pp. 82–93.

[4] B. Lucas and T. Kanade, "An iterative image registration technique with and application to stereo vision," in *Int. Joint Conf. on Art. Intel.*, 1981, pp. 674–679.

[5] I. Matthews, Takahiro Ishikawa, and S. Baker, "The template update problem," *IEEE Trans. PAMI*, vol. 26, no. 6, pp. 810–815, June 2004.

[6] S. Baker and I. Matthews, "Lucas-kanade 20 years on: A unifying framework," *IJCV*, vol. 53, no. 3, pp. 221–255, 2004.

[7] F. Dellaert and R. Collins, "Fast image-based tracking by selective pixel integration," in *Workshop on Frame-Rate Vision (ICCV)*, 1999.