Moving Object Tracking from Airborne Video using Relaxation Labeling Algorithm on Active SIFT points

Kyung min Han and G.N. DeSouza

January 11, 2010

1 Introduction

In this research, we are interested in tracking moving ground objects as they are perceived by airborne cameras. When the motion of the camera increases the dynamic of the background activity and the tasks of target segmentation and tracking become even more daunting. Unfortunately, without such segmentation, the detection, identification, and geolocation of these same targets would just be impossible.

We developed a simple, but accurate and effective method for segmentation and tracking of a moving target from dynamic backgrounds. Our method relies on the calculation of a differential sparse SIFT flow, which can separate the motion of the background (due to the airborne camera) from the moving targets. Once segmented out from its background, the images of possible targets are handled as independent ROIs (regions of interest). Then, each ROI undergoes feature based matching based on the relaxation labeling algorithm [2, 6, 3].

2 Proposed Method

The proposed framework is composed of two phases: the detecting phase and the tracking phase. Figure 1 (a) shows a flow chart of the algorithm.

2.1 Detecting Phase

For given two consecutive images, we can extract a set of SIFT [4] points that matches between two images. Since these two images are collected with a time difference of “d”, if the camera frame rate is fast as real time, we can expect a large set of matching points. Also, if either object in the image or the observing camera were moving in the images, we can create flow vectors that represent the movement of the objects. That is, for two consecutive image frames with moving objects, we can build sparse SIFT flows on the moving object. Not like dense flow estimation [7], we consider matching sparse points only as it suggested in [4].

Once the SIFT flows are determined for the given image set, we build a histogram for both magnitude and phase of the flow. The peak modes found in the magnitude and phase flow of the histogram are considered to be,

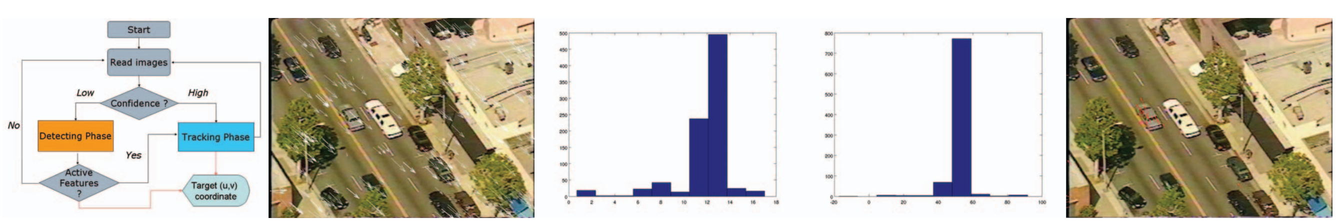

Figure 1: a) A flow chart of the entire algorithm b) Total flow vector created from two consecutive frames. c) Magnitude histogram of the flow d) Angle histogram of the flow (The modes of histogram represent the background flow vector) e) The foreground flow vector can be computed by subtracting the background flow vector from the total flow vectors.

respectively, the magnitude and the phase of the background flow. The moving objects can be segmented out by subtracting the background vector from the entire flow vectors. Figure 1 depicts the sequence of this procedure.

2.2 Tracking Phase

2.2.1 Probabilistic Relaxation Labeling

The template feature group is composed of a set of feature points $A = \{a_1, a_2, \ldots, a_n\}$, and their corresponding labels, $\Lambda = \{\lambda_1, \lambda_2, \ldots, \lambda_n\}$. In the mean time, we have an incoming query image patch with a group of feature points, $B = \{b_1, b_2, \ldots, b_m\}$. The labeling problem is to assign the best labels, Λ, to the feature set of the query image patch, B. Every consistently labeled feature points in B represents the matched target feature points, or the tracked feature points. In order to achieve this goal, we need set up a good confidence measurement for each labeling. A confidence in assigning a label α_i to b_j depends on the strength of interaction with its neighboring feature points. In each iteration, the algorithm performs the confidence measurement based on the compatibility function, and it updates the probability table using the new confidence measurement.

2.2.2 The compatibility function

Let's consider two feature points, b_i and b_j, from the query image patch. A possible two labels for these feature points are, λ_k and λ_l which are the labels for a_k and a_l from the template feature points, respectively. If neither camera nor the object undergo a severe rotation, the distances among all feature points in the template image should be preserved in the query image. Thus, an important clue is to look at the distance between two features. That is,

$$d_s = \frac{1}{\sqrt{2\pi\sigma}} exp(\frac{-s_m}{\sigma^2}) \tag{1}$$

where, s_m is a similarity measurement between two vectors, $\overrightarrow{b_i b_j}$ and $\overrightarrow{a_k a_l}$, in the image coordinate frame. Thus, d_s decreases if the position of b_j w.r.t b_i is similar to the position of a_l w.r.t a_k.

In addition to the distance relationship between feature points, another important factor is comparing the similarity between two features. For example, if the two labels, λ_k and λ_l are the correct labels for query feature points b_i and b_j, they should look very similar to a_k and a_l of the template features. That is,

$$d_m = D(b_i, a_k) + D(b_j, a_l) \tag{2}$$

where, $D(x, y)$ is a distance between a vector x and y. Since we are using SIFT points, the distance between two vectors can be computed by their inner product. Given these two different distances: d_s and d_m, the final compatibility function is defined as

$$r_{ij}(\lambda_k, \lambda_l) = \gamma d_s + (1 - \gamma) d_m \tag{3}$$

where, γ is averaging factor with boundary $|\gamma| \leqq 1$. Therefore, the support for $b_i = \lambda_k$ can be calculated as

$$q^s(b_i = \lambda_k) = \sum r(\lambda_k, \lambda_l) P^s(b_i = \lambda_k) \tag{4}$$

where, $P^s(b_i = \lambda_k)$ is the probability that feature b_i's label is λ_k at the iteration s. Also, it is subject to $\sum P^s(b_i = \lambda_k) = 1$ for any i.

Figure 2: a) Template feature points detected at frame number 300 b) Matching points detected in 330th frame after the labeling. Note that only the red numbers in (b) are successfully matched points.

Algorithm 1 Labeling Process

Iterate

1. Get Support, $Q^s(b_i = \lambda_k) = \sum_j w_{ij} q_j^s(b_j = w_k)$, where $q_j^s(b_i = w_j)$ is calculated as it is defined in equation 4 (w_{ij} is distance weight with unit sum, that is proportional to the distance between the i^{th}feature and the j^{th} feature.)

2. Compute a new probability $P^{s+1}(b_i = w_k) = \frac{P^s(b_i=w_k)Q(b_i=w_k)}{K}$, where $K = \sum_l P^s(b_i = w_l)Q^s(b_i = w_l)$ is a normalizing constant

---

3 Results

The proposed tracking algorithm has been tested with a number of data sets and with different resolution, using different target objects, and scenarios. Images from those videos are shown in figure 3. All image sequences are captured from aerial vehicles.

A square placed on top of the target objects shows when the car is being tracked. Some of the images, for example the Hollywood sequence shown on the top row of figure 3, contained backgrounds with much clutter, such as trees, other cars, buildings, and etc. That could cause momentary disruption of the tracking algorithm. However, as the results show, the two-phase approach of the tracking algorithm could recover the tracking quite well and the overall performance was consistently high.

The data set shown in the 2^{nd}, 3^{th}, and 4^{th}rows are collected for geolocation project [5], and they are easily available in our server, http://vigir.missouri.edu/~gdesouza/TrackAndGeo/. After applying the proposed algorithm to these data, we have observed similarly good result that we achieved with the Hollywood data sequence.

The last data set shown in the last row of the figure 3 is the DARPA sequence available in the CMU data base. This sequence was bit harder than others because there are multiple objects moving with similar speed. Moreover, vehicles frequently undergo severe occlusions. As a result, the tracker jumped around to different targets several times.

4 Conclusion

In this paper, we addressed a robust tracking method to track moving ground vehicles captured from the aerial vehicles. The algorithm calculates sparse SIFT flow to represent the motion of feature points. Based on, the histogram analysis on these flow vectors, we can segment out the moving features only. Later on, these features are tracked by a labeling algorithm. We used SIFT points in our implementation, but for faster performance, we can also use different kinds of features such as SURF [1]. In order to validate our method, we tested it with several challenging airborne video sequences, and the performance of the algorithm was excellent.

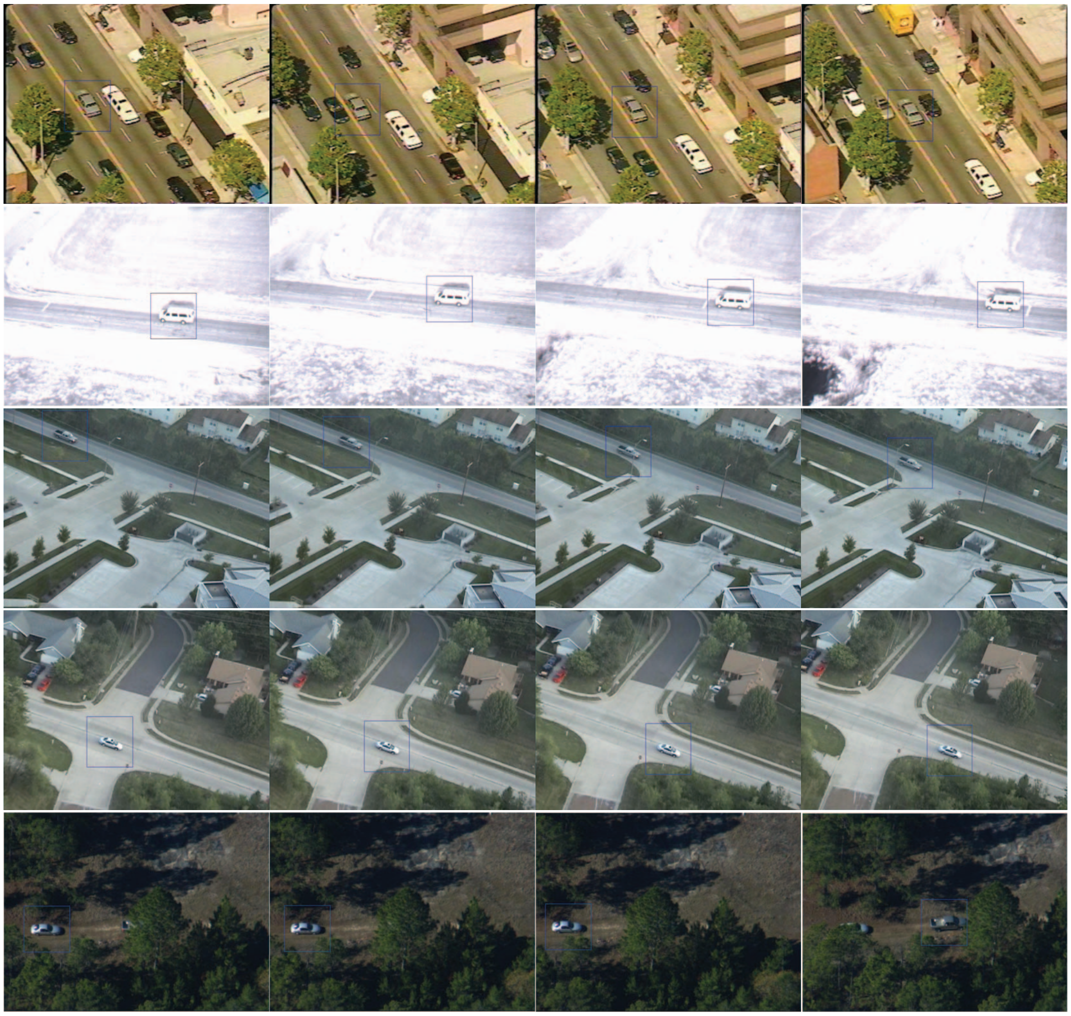

Figure 3: Tracking result achieved with different data set

References

[1] Tinne Tuytelaars Herbert Bay, Andreas Ess and Luc Van Gool. Surf:speeded up robust features. *Computer Vision and Image Understanding*, 10:346–359, 2008.

[2] Azriel Rosenfeld; Robot A. Hummel and Steven W. Zucker. Scene labling by relaxation operations. *IEEE, Trans. Man, And Cybernetics*, 6:421–433, 1976.

[3] R.A. Hummel and S.W Zucker. On the foundations of relaxation labeling process. *PAMI*, pages 267–287, 1983.

[4] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2), 2004.

[5] Kyung min Han and G.N. DeSouza. Multiple targets geolocation using sift and stereo vision on airborne video sequences. In *IEEE, International Conference on Intelligent ROobots and Systems*, 2009. St.Louis, USA.

[6] Rosenfield. Iterative methods in image analysis. *PR*, pages 181–187, 1978.

[7] C. Liu; J. Yuen; A. Torralba; J. Sivic and W. Freeman. Dense correspondence across different scenes. In *ECCV 2008*, pages 28–42, June 2008. Marseile, France.