Tracking Many Vehicles in Wide Area Aerial Surveillance

Jan Prokaj, Xuemei Zhao, and Gérard Medioni
University of Southern California
Los Angeles, CA 90089
{prokaj|xuemeiz|medioni}@usc.edu

Abstract

Wide area aerial surveillance data has recently proliferated and increased the demand for multi-object tracking algorithms. However, the limited appearance information on every target creates much ambiguity in tracking and increases the difficulty of removing false target detections.

In this work we propose to learn motion patterns in wide area scenes and take advantage of this additional information in tracking to remove false alarm and reduce tracking error. We extend an existing multi-object tracker for wide area imagery by incorporating the motion pattern data as further probabilistic evidence. Scalability is ensured by dividing the imagery into tiles, processing each tile in parallel, and handing off tracks between tiles when necessary.

Evaluation on sequences from a real wide area imagery dataset shows this approach outperforms a competing tracker not making use of such data.

1. Introduction

The increased deployment of aerial sensors for wide area surveillance has generated huge amounts of imagery, rich with interesting information. This imagery, however, is beyond our ability to analyze manually. It often covers a geographic area of a few square kilometers, which can not be analyzed or monitored at once by one person. Spreading the workload across several people is possible, but at the cost of missing the big picture and more financial resources.

Computer vision offers to solve this problem by automatically processing the imagery into a compact form that is more useful and understandable to a person. One of the fundamental ways that motion imagery is summarized is by estimating the tracks of moving objects. These tracks tell us a lot about the "life" in the region of interest, and are essential cues for higher level reasoning tasks, such as activity recognition. The tracking problem has been studied in many ways for a long time [21], and we have reached a point where excellent results are obtained in many environments [7]. Our domain of wide area surveillance, however, brings its own set of challenges that have not been fully solved, and its own set of opportunities that have not been fully exploited. The challenges are multiple-sensor video capture, which requires accurate registration of the different sensors, low sampling rate, which undermines the common motion smoothness assumption, and limited grayscale resolution of moving targets, which prevents the use of robust and discriminative appearance models.

Current state of the art multi-object tracking methods are detection based [5, 14, 16, 9], which means that target detections are first determined over a sliding window of frames, and these detections are then associated into tracks. These detections are determined in one of two ways: from background subtraction, or using an object detector. In our domain, the small target size makes it difficult to train an object detector with sufficient accuracy. For this reason, background subtraction has been the common way of obtaining target detections [14, 16].

In this work, the moving targets of interest are vehicles. One of the good characteristics of tracking vehicles in our imagery is that the motion of each vehicle is highly predictable most of the time. Vehicles move on roads, at fairly constant velocity, except at intersections. This characteristic induces *motion patterns* in the imagery, which can be exploited to minimize ambiguity in data association during tracking and mitigate the weakness of the appearance model. The use of motion patterns for improving tracking performance has been shown before [3, 22, 24], but it has not been applied to wide area aerial surveillance imagery.

Our primary contribution in this work is a multi-object tracker for wide area imagery that takes advantage of motion patterns in estimating tracks. It accomplishes this by extending the tracker introduced in [14] and incorporating a motion pattern prior in a conditional distribution of the predicted detection state given the current detection state. Our results show that incorporating this motion pattern prior greatly reduces the false alarm rate and ID switches, with a small negative impact on the object detection rate. Furthermore, we discuss a scalable implementation of the tracking algorithm that is able to track vehicles in a region of a few

square kilometers in real-time.

2. Related Work

Tracking targets in wide area aerial surveillance video has recently received a lot of attention [14, 16, 20]. The approach of Xiao *et al*. [20] is detection based, where detections are associated using the classic Hungarian algorithm. The association cost matrix is computed by combining target association cost matrix and the novel target-pair association cost matrix, which allows the inclusion of spatial constraints between neighboring targets. These spatial constraints formalize the idea that a pair of vehicles in the current frame is more likely to match a pair of vehicles in the next frame when the distance and speed difference between the vehicles is preserved. However, this assumption does not always hold, and applying it uniformly to all targets may lead to inappropriate associations. The appearance model in this work is template based, which may have problems with lighting changes and varying backgrounds. Tracks are initialized using three-frame subtraction, but otherwise background subtraction is used for detection.

In [16], Reilly *et al*. also use the Hungarian algorithm for association of detections, but propose to increase its efficiency by dividing the image into cells and computing the associations within each cell. The matching cost between targets takes into account spatial proximity, velocity orientation, orientation of the road, and local context between cars. The use of local context only helps on freeways, where the relative position between targets does not change much over time. On most city roads, where cars change lanes, pass, stop, and turn, this kind of constraint would be less helpful and may cause incorrect associations.

One limitation of using the Hungarian algorithm for associating detections is that it works frame to frame, and does not take advantage of motion and appearance smoothness that holds over several frames. This property has been exploited by several authors [14, 5, 9, 23] to achieve state of the art results. In the domain of wide area aerial surveillance, Prokaj *et al*. [14] show how the space of possible detection associations is significantly reduced by solving a binary labeling problem. This is very useful in reducing false alarm rates and minimizing track fragmentation.

A min-cost flow [23] or Linear Program formulations [5] of multiple-target tracking have been proposed as a promising alternative. The advantage here is that a globally optimal solution can be obtained very efficiently. However, one disadvantage of the Linear Programming formulation is that one must specify a priori locations where objects enter the scene and exit. Furthermore, these methods are less tolerant to false detections and may create many short tracks.

Motion patterns have been studied in previous works from various perspectives. [18, 25] use motion patterns to do scene understanding, and [25] also includes sources and sinks as high-level semantic prior. Since motion patterns contain rich information of the scene, the learning results are used to detect anomalous events [8, 19], and learn traffic rules [11]. [24, 2] use motion pattern as prior information to perform tracking. However, different from general scenes, wide area scenes usually have complex road networks, so motion pattern segmentation [24] is difficult if not impossible. Also, high-rise buildings in aerial images cause serious parallax, which may bring in false trajectories.

3. Approach

Our goal is to track all moving objects in the video from frame to frame. As mentioned in the introduction, wide area aerial surveillance imagery is captured by an array of sensors. Therefore, the input to our algorithm is not one video stream, but rather an array of video streams. Traditionally, there are two ways to handle this. One is to track objects independently in each video and then hand-off the tracks as they cross from the field of view of one sensor to another one. The second way is to first mosaic the array of sensors, and then estimate tracks on the resulting single video stream. In this work we take a hybrid approach, where we first mosaic the sensor array and then divide the stabilized and georeferenced imagery into a number of tiles for parallel processing. This offers an advantage of seeing the "big picture" of the area under surveillance provided by the mosaic and at the same time retains the ability to do parallel processing with an optimal number of tiles. We use the mosaicking algorithm proposed in [15] and briefly describe it in the next subsection.

One of the most successful classes of tracking algorithms is hierarchical [13, 4, 9, 10]. Such approaches first estimate short tracks or tracklets over a brief time period and then iteratively link these tracklets into long tracks. Tracklets are determined using nearest neighbor association [13], affinity measures [9], or particle filtering [10]. We adopt an approach presented in [14], which determines the tracklets optimally by maximizing the joint probability of a set of detections over a temporal window. Here we extend it to take advantage of learned motion patterns during data association by incorporating motion pattern priors in the joint probability distribution.

The tracking algorithm and its extension is described in subsection 3.2. We explain how we learn the motion patterns in subsection 3.3. The approach described in [24] is well suited for our domain and is described there. Brief discussion on scalability is in subsection 3.4.

3.1. Mosaicking

In mosaicking we are concerned with the generation of a single image from the sensor array, as if it were captured by a single (virtual) camera. High quality estimation is a primary requirement. If it is inaccurate (has visible seams),

Figure 1. One frame of the imagery is captured by an array of cameras (left), while it is desirable to work with only one image per frame, as if it were captured by a virtual camera (right).

errors will result in tracking and other algorithms that use this image is input.

In order to generate a high quality mosaic, the model which registers one image onto another needs to have enough degrees of freedom to handle the image deformation. For a true pinhole camera, this model is a homography, which has 8 degrees of freedom. In practice, this model does not hold and must be augmented with additional transformations. In [15], these transformations are modeled with a piecewise affine model.

The piecewise affine model is estimated by minimizing the variance in intensity of overlapping pixels. To make this process more efficient, only a fraction of the overlapping pixels is used in the optimization. These pixels are selected to be spatially distributed and with a high Harris corner measure. Sampling pixels in this manner helps to ensure that all parts of the mosaic are well registered while maximizing the contribution of the few selected ones in the optimization.

Geometrical alignment alone is not sufficient to produce a seamless mosaic, because each camera in the sensor array has different exposure or response function in general (see Figure 1). Therefore we also try to do a coarse intensity alignment of the sensor array. As suggested in [15], we parameterize the differences in intensity as differences in camera gain (scaling factor), and solve for the jointly optimal scaling factors. Any remaining photometric differences or alignment errors are removed by blending the images using [6].

3.2. Tracking

As mentioned earlier, we take a hierarchical approach to tracking. That is, we first estimate short tracks, or tracklets, over a short temporal window and then associate them with an existing set of tracks. The association of tracklets with existing tracks is a relatively easier problem than inferring the tracklets themselves, therefore we focus on the tracklet inference problem here.

The input to our algorithm is a set of object detections (blobs) in each frame, which we estimate using background subtraction. Each detection in the first frame of the window is a potential object. Therefore, we find an optimal tracklet, or a set of tracklets, starting at each detection in the first window frame. This is not a problem, because for detections that are false alarms, the model of a valid tracklet (consistency of motion and appearance) is not satisfied, and the tracklet is discarded. Tracklets that start in the second or later frame of the window are found when the sliding window shifts to that frame.

With the exception of missed detections, given a detection of an object in one frame, we know there must be another detected instance of that object located "nearby" in subsequent frames. By recursively applying this idea, a directed acyclic graph called an association graph, or detection graph is constructed. This graph stores all possible associations of object detections over time. The number of these associations ($>=$ the number of paths from the root down to leaves) is large and intractable to evaluate in entirety. The key idea of [14] is to first remove inconsistent detections in this graph by solving a binary labeling problem, which is very efficient. Once this has been done, the search space of paths in this tree is significantly reduced, and the few (often one) remaining possible paths are easily extracted.

Missed detections, due to occlusion or background subtraction failure, are easily handled in this framework by generating virtual detections in this graph. Any time a detection in frame t does not have anything to link to in frame $t+1$, a virtual detection is generated by predicting the location and appearance of the target in the next frame. This procedure is also recursive, so that when a newly added virtual detection does not have nearby detections in the next frame, the process is repeated. The graph and the following computation does not change.

The binary labeling problem is posed as MAP inference in a Bayesian network, which is constructed directly from the detection graph. This formulation allows easy incorporation of other evidence, such as motion patterns, in to the problem. The joint probability distribution of detection labels $\mathbf{y}$ and observations $\mathbf{o}$ is factorized as

$$p(\mathbf{y},\mathbf{o}) = p(y^0) \prod_{\substack{i,j,t>0 \\ y_i^t \text{near } y_j^{t-1}}} p(y_i^t|y_j^{t-1}) \prod_{i,t>0} p(\mathbf{o}_i^t|y_i^t)\ . \quad (1)$$

where y_i^t is a binary random variable denoting the label of a detection i at frame t, and $\mathbf{o}_i^t$ is the observed properties of each detection (location, appearance, *etc.*). This factorization is illustrated in Figure 2.

To incorporate motion pattern evidence into the probability distribution, only the conditional distributions $p(y_i^t|y_j^{t-1})$ need to be modified. Where previously only

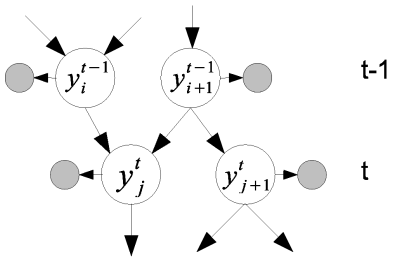

Figure 2. Example of the structure of the graphical model. Each y_i^t is a binary variable that represents a detection label. The shaded nodes represent the measurements associated with each detection (location, appearance, etc.).

the appearance similarity and motion likelihood terms were used, we now add another motion pattern prior term, g, which is assumed to be independent of the previous two terms. The conditional probability table now takes the form shown in Table 1.

As the motion likelihood, the motion pattern prior uses the filtered (posterior) state of a linear Gaussian motion model in estimating a probability. A standard linear Gaussian motion model is defined as follows:

$$\mathbf{z}_{t+1} = \mathbf{A}\mathbf{z}_t + \mathbf{w} \tag{2}$$

$$\mathbf{x}_{t+1} = \mathbf{H}\mathbf{z}_{t+1} + \mathbf{v} \tag{3}$$

where $\mathbf{z}_{t+1}$ is the state vector, which includes the object position and velocity, $\mathbf{x}_{t+1}$ is the measurement vector of the object position, $\mathbf{w} \sim \mathcal{N}(\mathbf{0}, \mathbf{Q})$ is the process noise, and $\mathbf{v} \sim \mathcal{N}(\mathbf{0}, \mathbf{R})$ is the measurement noise.

In our implementation, the motion pattern prior g is estimated using a kernel density, which is learned in the beginning of the sequence. This kernel density is expressed as

$$g(y_i^t) \approx \sum_{n=1}^{M} \exp\left(-\mathbf{e}_n^T \mathbf{\Sigma}^{-1} \mathbf{e}_n\right) \tag{4}$$

where

$$\begin{aligned}\mathbf{e}_n = \big(&\bar{\mathbf{z}}_t^i.x - \mathbf{g}^n.x, \bar{\mathbf{z}}_t^i.y - \mathbf{g}^n.y, \\ &\bar{\mathbf{z}}_t^i.v_x - \mathbf{g}^n.v_x, \bar{\mathbf{z}}_t^i.v_y - \mathbf{g}^n.v_y\big)^T ,\end{aligned} \tag{5}$$

$\bar{\mathbf{z}}_t^i$ is the posterior state estimate, and $\mathbf{g}^n = (x, y, v_x, v_y)^T$ is a motion pattern point (explained below). The covariance $\mathbf{\Sigma}$ is specified in experiments.

3.3. Learning Motion Patterns

A motion pattern is a smooth and compact spatio-temporal structure that describes a set of neighboring objects undergoing coordinated movements. As pointed out in [22, 24], when tracklet points are embedded into (x, y, v_x, v_y) space, manifold structures emerge. In the motion pattern learning framework, the input are tracklets from an arbitrary multi-object tracker. In this work we use tracklets from [14] (acting as a bootstrap). In wide area scenes, many tracklets are not caused by real object movement (positive tracklets), but parallax instead (negative tracklets). However, compared to positive tracklets, negative tracklets are usually short, the movement is usually small and the covered distance is often limited. Making use of these properties, most false tracklets can be removed.

For the remaining tracklets, velocity (v_x, v_y) is calculated on tracklet points, and 2D information is transformed into (x, y, v_x, v_y) space, in which Tensor Voting is used to explore manifold property. Then we filter out outliers based on the local geometric structure. The output is motion pattern information represented by a group of points $\mathbf{g}^n = (x, y, v_x, v_y)\,,\ n = 1, 2, ..., M$.

3.3.1 ND Tensor Voting

The key step in motion pattern learning algorithm is using Tensor Voting [12] to analyze the local structures at input ND points, so we briefly explain it in this section. Tensor Voting is a perceptual organization method enabling us to analyze geometric structure and estimate dimensionality.

For each input ND point, its geometric information is encoded in a second order, symmetric and non-negative tensor $\boldsymbol{T}$. $\boldsymbol{T}$ is a $N \times N$ matrix or an ellipsoid in ND space. Its eigenvectors corresponding to non-zero eigenvalues represent the point's normals, and eigenvectors corresponding to zero eigenvalues represent the point's tangents. $\boldsymbol{T}$ can be decomposed as,

$$\begin{aligned}\boldsymbol{T} &= \sum_{i=1}^{N} \lambda_i \boldsymbol{e_i} \boldsymbol{e_i}^T \\ &= \sum_{i=1}^{N-1} (\lambda_i - \lambda_{i+1}) \sum_{k=1}^{i} \boldsymbol{e_k} \boldsymbol{e_k}^T + \lambda_N \sum_{i=1}^{N} \boldsymbol{e_i} \boldsymbol{e_i}^T\end{aligned} \tag{6}$$

where $\{\lambda_i\}$ are the eigenvalues in descending order, and $\{\boldsymbol{e_i}\}$ are the corresponding eigenvectors. This way, local geometric information such as dimensionality and normal/tangent space at every point can be estimated by examining the eigensystem of its tensor. For instance, the dimensionality d of the structure that a point belongs can be calculated from the gaps between two consecutive eigenvalues.

$$d = N - \arg\max_i (\lambda_i - \lambda_{i+1}) \tag{7}$$

Briefly speaking, input ND points are encoded with tensors. They propagate their information to neighbors', and collect information from them, in a voting process. Then, the local geometric information of each point can be obtained by eigen decomposition analysis of its tensor.

	$y_i^t = 0$	$y_i^t = 1$
$y_j^{t-1} = 0$	0.5	0.5
$y_j^{t-1} = 1$	$1 - a(\mathbf{o}_i^t, \mathbf{o}_j^{t-1})m(\mathbf{o}_i^t)g(\mathbf{o}_i^t)$	$a(\mathbf{o}_i^t, \mathbf{o}_j^{t-1})m(\mathbf{o}_i^t)g(\mathbf{o}_i^t)$

Table 1. Conditional probability distribution of detection i at time t given a detection j at time $t-1$. The motion pattern prior term g mitigates the weakness of the appearance model a and motion model m.

3.3.2 Outlier Filtering

False tracklets bring in outlier in motion pattern space. The saliency of the structure that has i, $i = 1, 2, ..., N-1$ normals is $\lambda_i - \lambda_{i+1}$, and λ_N is the saliency of un-oriented structure. Summing the saliency together, λ_1 is an estimate of the probabilities for all possible manifold structures. Thus, all points are ranked according to their λ_1, and the bottom ones are filtered out as specifed in the experiments.

3.4. Parallel Processing

Wide area imagery often covers a region of several square kilometers, which means the number of vehicles that needs to be tracked is in the thousands. In order to achieve real-time processing at this scale, the tracking task needs to be parallelized and divided among several machines. A natural way to divide up the work is to create a grid of "tiles", each covering an equal area of the monitored region (we assume the imagery has been stabilized and georegistered), and run an independent tracker on each one. This is illustrated in Figure 3. The tiles should have a small overlap region to facilitate track hand off and their size set according to the tracking algorithm's resource requirements. For example, we have experimented with tiles roughly 2100x2100 in size.

Even though each tracker estimates tracks on only a small portion of the region under surveillance, we can avoid track fragmentation arising from the grid by handing off, or linking, tracks for targets that move from the field of view of one tile to the next. To make this process more robust, the tiles have a small overlap region, as mentioned earlier. We have implemented a simple track linking approach that works as follows.

Whenever a new track is initiated in an overlap region, the initiating tile sends a "new-track" message to the overlapping neighbors, which contains the track's ID and its initial trajectory. At the same time, each tile maintains a set of tracks that have terminated in an overlap region. On every frame, the set of terminated tracks is matched against the set of new tracks. Whenever a terminated track matches with a new track, a "hand-off track" message is sent with the trajectory of the terminated track to the tile containing the new track. The historical trajectory is then merged into the new track. Matching of terminated tracks with new tracks can be done with the Hungarian algorithm for significant robustness. However, we have found that having a reasonable overlap region allows a greedy track matching algorithm with no loss in accuracy.

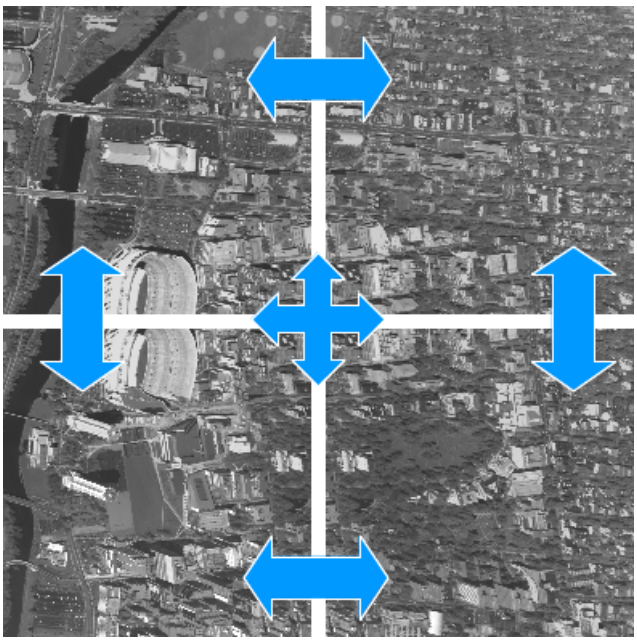

Figure 3. Parallel estimation of tracks on a cluster of computers is enabled by creating an overlapping grid of tiles and linking tracks that move between the tiles.

4. Results

We have evaluated our tracking algorithm on a sequence from a wide area imagery dataset [1]. This dataset is collected from an aerial platform flying in a circular pattern over Ohio State University. The dataset is captured at about 2 frames per second and contains significant parallax from campus buildings and trees. There are more than 4000 frames in the dataset, with each frame roughly 6500x7500 size. We have stabilized and georeferenced the dataset to 0.75 meters per pixel resolution before tracking. For quantitative evaluation we selected a 1312x738 region in the middle of the persistently visible area and manually determined tracking ground truth for 100 frames. The selected sequence has 205 tracks of vehicles, each being about 10x5 pixels in size.

Moving object detection was done using background subtraction. The background is modeled as the mode of a (stabilized) sliding window of frames. A tracking window size of 12 frames, corresponding to about 6 seconds of video was used. In motion pattern learning stage, given an in-

	Without MP	With MP
ODR	0.28	0.23
MODR	0.33	0.29
FAR	0.85	0.19
SWP	0.55	0.35
BRK	0.75	0.46

Table 2. Vehicle tracking performance with and without motion patterns (MP) on wide area imagery. Please see the text for metric definitions.

put tracklet, we first calculate the movement of every tracklet point compared to the starting of the tracklet, and remove the tracklet if the median of the movement is smaller than 6 pixels. That is because the tracklet points caused by parallax often constrained in a small region, while tracklets caused by real moving objects occupy a large area. The covariance $\mathbf{\Sigma}$ in (4) is empirically determined to be $\mathbf{\Sigma} = diag(10, 10, 2, 2)$. In outlier filtering, all points are ranked according to their λ_1, and the bottom 10% are filtered out.

Several metrics were used to evaluate performance: object detection rate (ODR), moving object detection rate (MODR), false alarm rate (FAR), mean cumulative swaps of tracks (SWP), and mean cumulative broken tracks (BRK). To calculate these metrics we determine a 1-1 assignment of estimated detections to ground truth detections using the Hungarian algorithm by maximizing the detection overlap. A correct detection is a detection in an estimated track that overlaps with a detection in a ground truth track.

ODR is defined as the number of correct detections divided by the length of all ground truth tracks. Since our ground truth has annotations for vehicles that come to a stop after initial movement, the ODR is going to be depressed when the input to the tracker is only moving object detections from background subtraction. Therefore, we also define a MODR, where the ground truth detections corresponding to stationary vehicles are removed. The FAR is defined as the number of false positive detections divided by the total number of estimated detections. Finally, BRK and SWP metrics are defined in [17] to measure multi-object tracking performance and are similar to track fragmentation or id switch rates found in the literature.

Quantitative results using the metrics are shown in Table 2. We have compared our proposed approach with motion patterns to one without, as described in [14]. Furthermore, Figure 4 shows the motion pattern learning process. In (a), the initial tracklets are displayed. We can see that there is a large number of false tracklets from parallax or false association. (b) shows the motion pattern learning results, which more or less correspond to the road network.

The results show that motion patterns significantly reduce the false alarm rate with a small corresponding decrease in the object detection rate. This is because most of the false alarms come from moving objects detections due to parallax and these are denoised by tensor voting. Furthermore, the number of ID switches and track fragmentation has also decreased with the use of motion patterns, as is evident in the decrease in the track swap rate and broken tracks rate. It's another indication that ambiguity during tracking has been reduced. While we have not shown this quantitatively, the reduced ambiguity has had another positive effect, and that is an increase in the computational efficiency of the tracker. Tracks were estimated on the evaluation sequence at about 1 frame per second.

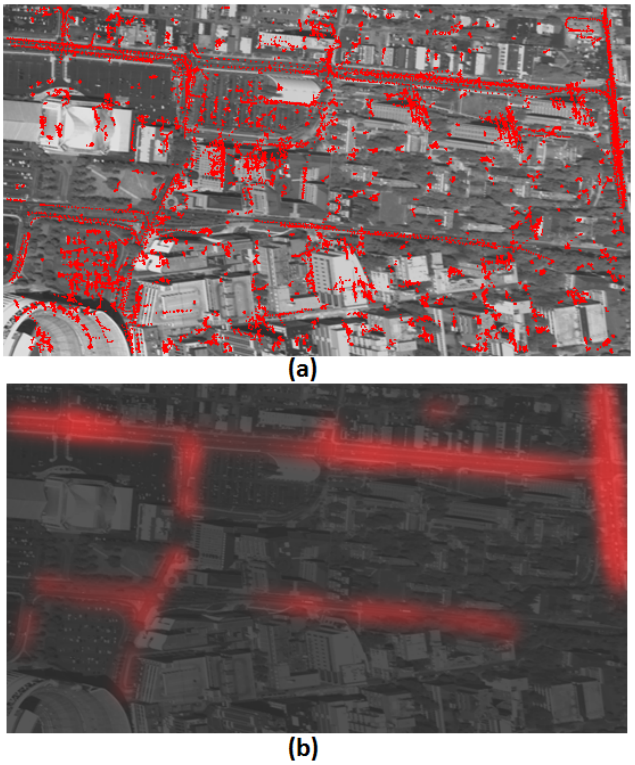

Figure 4. (a) Initial tracklets used to learn motion patterns. (b) Learned motion patterns.

5. Conclusions

Tracking in wide area motion imagery presents several challenges, one of which is the limited appearance information in every target. In this work we have shown how *motion patterns* learned from the sequence help overcome the limited appearance and reduce tracking error. Results on sequences from real wide area imagery show our algorithm outperforms a competing tracker, which does not make use of such motion pattern data, on several metrics.

In the future, we plan to extend this work by learning dynamic motion patterns online and optimize their contribution in the tracker with a motion pattern confidence map. Such change will enable persistent tracking of targets over

long time periods.

6. Acknowledgments

This work was supported in part by grant DE-FG52-08NA28775 from the U.S. Department of Energy and by funding from Lawrence Livermore National Laboratory.

References

[1] CLIF 2006. https://www.sdms.afrl.af.mil/index.php?collection=clif2006. 5

[2] S. Ali and M. Shah. A lagrangian particle dynamics approach for crowd flow segmentation and stability analysis. In *IEEE CVPR*, pages 1 –6, 2007. 2

[3] S. Ali and M. Shah. Floor fields for tracking in high density crowd scenes. In *ECCV*, volume 5303 of *Lecture Notes in Computer Science*, pages 1–14. Springer, 2008. 1

[4] M. Andriluka, S. Roth, and B. Schiele. People-tracking-by-detection and people-detection-by-tracking. In *IEEE CVPR*, pages 1–8, 2008. 2

[5] J. Berclaz, F. Fleuret, E. Turetken, and P. Fua. Multiple object tracking using k-shortest paths optimization. *IEEE PAMI*, 33(9):1806 –1819, sept. 2011. 1, 2

[6] P. J. Burt and E. H. Adelson. A multiresolution spline with application to image mosaics. *ACM Trans. Graph.*, 2:217–236, 1983. 3

[7] T. Dinh and G. Medioni. Context tracker: Exploring supporters and distracters in unconstrained environments. In *IEEE CVPR*, 2011. 1

[8] W. Hu, X. Xiao, Z. Fu, D. Xie, T. Tan, and S. Maybank. A system for learning statistical motion patterns. *IEEE PAMI*, 28(9):1450 –1464, sept. 2006. 2

[9] C. Huang, B. Wu, and R. Nevatia. Robust object tracking by hierarchical association of detection responses. In *ECCV*, volume 5303 of *Lecture Notes in Computer Science*, pages 788–801. Springer, 2008. 1, 2

[10] S. L. Junlian Xing, Haizhou Ai. Multi-object tracking through occlusions by local tracklets filtering and global tracklets association with detection responses. In *IEEE CVPR*, pages 1200–1207, 2009. 2

[11] D. Kuettel, M. D. Breitenstein, L. V. Gool, and V. Ferrari. What's going on? discovering spatio-temporal dependencies in dynamic scenes. *IEEE CVPR*, pages 1951–1958, 2010. 2

[12] P. Mordohai and G. Medioni. Dimensionality estimation, manifold learning and function approximation using tensor voting. *JMLR*, 11:411–450, 2010. 4

[13] A. Perera, C. Srinivas, A. Hoogs, G. Brooksby, and W. Hu. Multi-object tracking through simultaneous long occlusions and split-merge conditions. In *IEEE CVPR*, volume 1, pages 666 – 673, 2006. 2

[14] J. Prokaj, M. Duchaineau, and G. Medioni. Inferring tracklets for multi-object tracking. In *Workshop of Aerial Video Processing Joint with IEEE CVPR*, 2011. 1, 2, 3, 4, 6

[15] J. Prokaj and G. Medioni. Accurate efficient mosaicking for wide area aerial surveillance. In *IEEE WACV*, pages 273 –280, 2012. 2, 3

[16] V. Reilly, H. Idrees, and M. Shah. Detection and tracking of large number of targets in wide area surveillance. In *ECCV*, volume 6313 of *Lecture Notes in Computer Science*, pages 186–199. Springer, 2010. 1, 2

[17] R. L. Rothrock and O. E. Drummond. Performance metrics for multiple-sensor multiple-target tracking. In *Proceedings of SPIE*, volume 4048, pages 521–531, 2000. 6

[18] I. Saleemi, L. Hartung, and M. Shah. Scene understanding by statistical modeling of motion patterns. In *IEEE CVPR*, pages 2069–2076, 2010. 2

[19] X. Wang, X. Ma, and E. Grimson. Unsupervised activity perceptionby hierarchical bayesian models. In *IEEE CVPR*, pages 1–8, 2007. 2

[20] J. Xiao, H. Cheng, H. Sawhney, and F. Han. Vehicle detection and tracking in wide field-of-view aerial video. In *IEEE CVPR*, pages 679 –684, 2010. 2

[21] A. Yilmaz, O. Javed, and M. Shah. Object tracking: A survey. *ACM Comput. Surv.*, 38(4):13, 2006. 1

[22] Q. Yu and G. Medioni. Motion pattern interpretation and detection for tracking moving vehicles in airborne video. In *IEEE CVPR*, pages 2671 –2678, 2009. 1, 4

[23] L. Zhang, Y. Li, and R. Nevatia. Global data association for multi-object tracking using network flows. In *IEEE CVPR*, pages 1 –8, 2008. 2

[24] X. Zhao and G. Medioni. Robust unsupervised motion pattern inference from video and applications. In *IEEE ICCV*, pages 715 –722, 2011. 1, 2, 4

[25] B. Zhou, X. Wang, and X. Tang. Random field topic model for semantic region analysis in crowded scenes from tracklets. In *IEEE CVPR*, pages 3441–3448, 2011. 2