Interactive Target Tracking for Persistent Wide-Area Surveillance

Ilker Ersoy[a], Kannappan Palaniappan[a], Guna S. Seetharaman[b] and Raghuveer M. Rao[c]

[a]Department of Computer Science, University of Missouri-Columbia, Columbia, MO, USA
[b]U.S. Air Force Research Laboratory, Rome, NY, USA
[c]U.S. Army Research Laboratory, Adelphi, MD, USA

ABSTRACT

Persistent aerial surveillance is an emerging technology that can provide continuous, wide-area coverage from an aircraft-based multiple-camera system. Tracking targets in these data sets is challenging for vision algorithms due to large data (several terabytes), very low frame rate, changing viewpoint, strong parallax and other imperfections due to registration and projection. Providing an interactive system for automated target tracking also has additional challenges that require online algorithms that are seamlessly integrated with interactive visualization tools to assist the user. We developed an algorithm that overcomes these challenges and demonstrated it on data obtained from a wide-area imaging platform.

Keywords: Tracking, persistent surveillance, wide-area motion imagery.

1. INTRODUCTION

Persistent wide-area motion imagery (WAMI) is a newly emerging technology that provides continuous coverage of several square miles with a temporal sampling of several frames per second.[1,2] Aircraft-based surveillance systems provide live, high resolution images that are collected by an on-board camera array, then mosaiced and geo-registered to create a persistent view of a geographical region. Airborne wide-area imaging technology is becoming more cost effective and has applications across a range of scales measuring the movement of people and vehicles for urban development, traffic analysis, emergency response, law enforcement, and defense applications.[1–8] Wide-area motion imagery poses unique challenges for computer vision algorithms for automated detection and tracking of vehicles. Tracking vehicles of interest in WAMI needs to overcome imaging challenges including changing object appearance, seamless multi-camera mosaicing, geo-registration and stabilization errors due to errors in sensor measurements and limited accuracy of geodetic models, low frame rate, parallax effects and occlusions. A point on the ground is in the FOV of different cameras at different times which may cause considerable distortion and abrupt jumps through camera seams due to geo-registration and stabilization errors. IMU noise can also cause abrupt jumps between frames. These errors hinder moving object detection via background modeling as well as prediction algorithms. Changing viewpoint causes parallax effect in stationary structures and creates occlusions and also hinders background modeling. For these reasons, tracking approaches based on accurate motion detection, static vehicle modeling and smoothly moving vehicles at high frame rates have limited use in this type of imagery. Monochrome imaging and high altitude further complicate accurate modeling of vehicles due to lack of robust features. Also the large volume of data (several terabytes per flight) provides logistic challenges regarding storage, access and online processing for interactive, real-time forensic analysis.

1.1 Persistent Surveillance Data Sets

An eight-camera optical array constructed by Persistent Surveillance Systems Inc. (PSS) was used to collect the aerial imagery used in this paper. Each camera in the camera array produces an 11 megapixel 8-bit visible channel grayscale image at one frame per second and of size 4096×2672 that is orthorectified and geo-registered to a 16K×16K image mosaic. Each mosaicked image frame covers about four square miles. Figure 1 shows the configuration of cameras, focal lengths and overlap regions between adjacent cameras where inconsistencies across the seams are likely to occur. The PSS data sets we report in this paper were from wide-area imagery collected over downtown Philadelphia, PA and over the race track in Charlotte, NC. The airborne imaging platform flies a continuous circular flight path. The varying viewpoints cause a parallax effect of stationary structures and induce

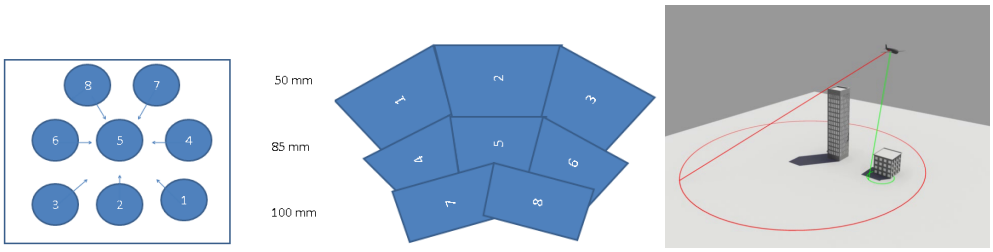

Figure 1. Left: camera array configuration, middle: projected camera views and overlapping regions, right: parallax effect.

an apparent circular motion for those structures that are above the ground plane, with taller buildings having a larger motion. This parallax effect periodically causes major occlusions of roads and vehicles by tall buildings. Figure 1 shows a rendering of the flight path of the platform and the ground plane trajectory of a 3D corner point on the lower building. Figure 2 shows a sample of parallax and the resulting occlusion at an intersection from Philadelphia sequence. Errors in geo-registration due to inaccurate DEM and drift in measurements create abrupt jumps and drifts in the image sequence in both spatial and temporal context. Figure 3 shows two excerpts of ground truth trajectories as marked on each frame for the shown vehicles from Charlotte sequence. It shows the challenges in both accurate motion prediction and abrupt appearance changes across camera seams. As mentioned by Cuntoor et al.,[6] these particular data sets pose major challenges in moving object detection by using motion clues due to many false detections.

2. INTERACTIVE TRACKING WITH ADAPTIVE MODEL OF DENSE DESCRIPTORS

In order to meet the challenges of wide-area motion imagery as exemplified in the presented data set, we developed a tracker that does not rely on motion clues, accurate stabilization, or background modeling for detection. Several trackers in the literature that are developed to track particular objects utilize unique features for detection, where a set of interest points or image patches represented by descriptors are matched against the next frame in order to locate the object.[9–11] For certain applications where image size is considerable small (e.g. webcams), the whole image can serve as the search region. This approach works well if no other similar objects are within the view. In case of wide-area imagery, this is not applicable, because of the sheer size of the image, and multitude of possible matches since many vehicles in an aerial view from high altitude look alike. We opt for feature based detection of the vehicles, but differently from other approaches that use feature detectors to find interest points, we use a dense feature set in order to accommodate wide-area motion imagery. This decision is mainly due to the fact that the vehicles of interest have very small support (about 20×30 pixels), and due to the altitude and camera setup as mentioned before, they lack prominent features that can reliably serve as unique interest points from frame to frame. Factors such as low resolution, distortion, blurry imaging and constant change of viewpoint are exacerbated more by the very low frame rate (1 fps) causing a vehicle to change its appearance vastly even

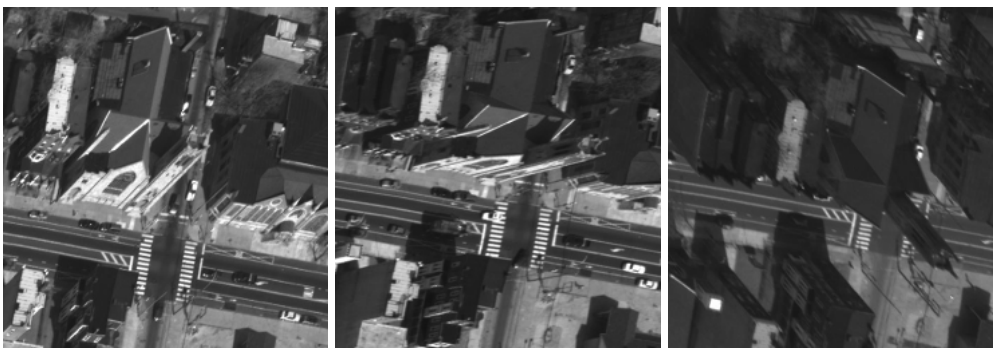

Figure 2. Sample of parallax and resulting occlusions at an intersection in Philadelphia sequence.

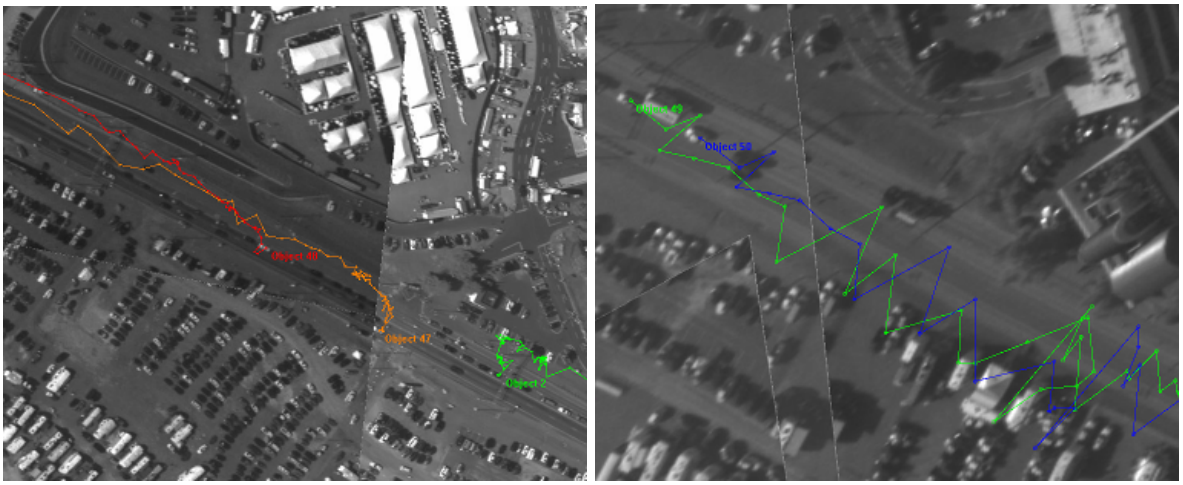

Figure 3. Excerpts of ground truth trajectories of several vehicles from Charlotte sequence. Abrupt jumps due to geo-registration errors as well as inconsistencies across camera seams pose major challenges for tracking.

within a few frames (see Figure 4). This requires an adaptive vehicle model of dense feature descriptors with overlapping support to account for the uncertainties and occlusions in a robust manner. Starting with a model of vehicle, the tracker should search for matches in the next frame in the locality of the predicted location of the vehicle in order to curb many possible matches, and also to maintain real-time response. By basing the model on a multitude of descriptors with overlapping support, we compensate for the fact that even between two consecutive frames the matching will be far from perfect due to aforementioned factors. The following subsection describes the vehicle model and the matching process adapted to wide-area imagery to overcome these problems. Another important design consideration is to be able to use the tracker in an interactive setting. There are many applications of interactive tracking such as forensic analysis of stored imagery (accurate tracking of vehicles forward and backward in time) or monitoring of live events. This does not only impose a constraint for real-time processing, but also imposes a seamless integration with the visualization, storage and access of wide-area motion imagery in order to provide interactive capability to the human analyst. Accessing to stored imagery as well as live imagery from the imaging platform puts a constraint on how much lag time the system and the analyst can tolerate. Also the tracker has to work on whatever quality of imagery is provided without making assumptions about perfectly stabilized background or properly stitched images. Since offline processing of the whole 16K by 16K image sequence may not be possible in a real-time live application, the tracker has to work on local image regions that surround the vehicles of interest and strike a balance between processing time and accuracy of processing. Considering these constraints, we developed an interactive tracker that allows the user to mark vehicles of interest and allow her to observe and validate the automated tracking results in real-time, intervening when necessary in a seamless manner without requiring special hardware or major lag time. The following subsections describe the vehicle model, matching process and stabilization to provide accurate prediction of the vehicle location.

2.1 Vehicle Modeling with Dense Descriptors

We choose an adaptive vehicle model based on a multitude of feature descriptors with overlapping support in order to accommodate the challenges in wide-area motion imagery as discussed in previous sections. Our choice of descriptors are the Speeded-Up Robust Feature (SURF) descriptors as proposed by Bay et al.[12] SURF and SIFT[13] are very popular interest point detectors and descriptors that have many applications for image matching. They have been used in many tracking approaches due to their robust nature of finding unique interest points. Here, we only use the descriptor scheme of SURF, and not the interest point detector. For reasons explained before, finding unique, reliable and repeatable interest points are very challenging and prone to mismatches for vehicles with a very small support in wide-area imagery. This problem was reported by Ma and Grimson[9] for even larger vehicles in a mid-field surveillance framework. By only borrowing the descriptor definition of SURF,

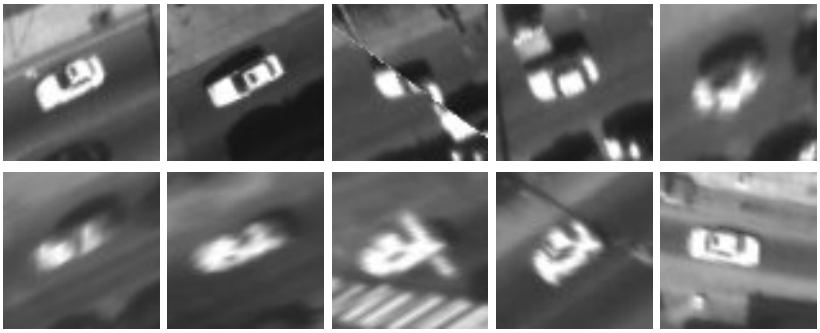

Figure 4. A sample car from Philadelphia sequence going through fast appearance changes.

we represent local image regions by this robust descriptor that is invariant to slight illumination changes as well as small changes in scale and orientation. It is worth noting that scale-invariant nature of SURF is not a result of its descriptor but of its interest point detector. Since the altitude of the imaging platform and camera setup does not change significantly during a collection, we do not require scale-invariant feature descriptors. Similarly we do not utilize rotation-invariant type of SURF descriptor since it would produce too many false matches within a given search window in an urban scene. Based on these considerations, our vehicle model consists of a collection of 64 dimensional SURF descriptors that represent the local image patches around regularly spaced points on the support of the vehicle at a given fixed scale.

Given a bounding box of size $m \times n$ that surrounds the vehicle of interest at frame $t = 0$, a set of descriptors $\mathbf{D} = \{d(x_{ij})\}$ are computed first where i and j are local coordinates with respect to the bounding box and d is the 64 dimensional SURF descriptor of the image patch around the points $\mathbf{X} = \{x_{ij}\}$ at a scale s:

$$\mathbf{X} = \{x_{ij} | i \in \mathbb{Z}_m, i \bmod \mathrm{q} = 0, j \in \mathbb{Z}_n, j \bmod \mathrm{q} = 0\} \tag{1}$$

so that $\mathbf{X}$ are sampled every q pixels in both directions in the bounding box. Given $\mathbf{X}$, $d(x_{ij})$ are computed as

$$\mathbf{D} = \{d(x_{ij}) | x_{ij} \in \mathbf{X}\} \tag{2}$$

Since computing SURF descriptors involves a $20s \times 20s$ image patch, there is a big overlap between the patches around points $\mathbf{X}$ for very small values of q. This redundancy provides the robustness in case of major distortion or occlusion to the vehicle. Overlapping patches produce similar descriptors especially for flat image patches or along the edges. To reduce the amount of computation as well as bias to such patches, we run a mean shift procedure on the set $\mathbf{D}$ to obtain natural clusters in the 64D descriptor space. Mean shift[14,15] is used for clustering since it can detect the modes of a density function given a discrete sampling of that function. The obtained set of cluster centers $\mathbf{C} = \{c_k\}$ serve as the initial vehicle model. To impose spatial coherency on this model, we also compute the relative coordinates $\mathbf{R} = \{r_{ij}\}$ of $\mathbf{X}$ with respect to the center $\mathbf{c}$ of the bounding box which is assumed to be the centroid of the vehicle. Even though the clustering runs on the descriptor space, spatially close descriptors are assigned to similar clusters, hence for each cluster center c_k we compute the median of relative coordinates of all d_{ij} that belong to that cluster and store it. So the model of the vehicle becomes

$$\mathbf{V} = \{(c_k, r_k) | k \in \mathbb{Z}_K\} \tag{3}$$

where K is the number of clusters. Given this model and a search window SW around a predicted vehicle location p in the next frame $t + 1$, detection process becomes finding the most likely position of the vehicle's centroid by matching the descriptors of the model to every possible scene descriptor d_{ij}^{SW} in SW with a similar scale s and grid length q. Matching is accomplished by computing the Euclidean distance between every pair of d_{ij}^{SW} and c_k, ranking the distances of all matches of c_k and computing the distance ratio of the best match to the next best match. Given $e_{ij,k}$ is the Euclidean distance between d_{ij}^{SW} and c_k, the distance ratio of the matching pair (ij, k) is

$$DR_{ij,k} = \frac{e_{ij,k}^{(1)}}{e_{ij,k}^{(2)}}. \tag{4}$$

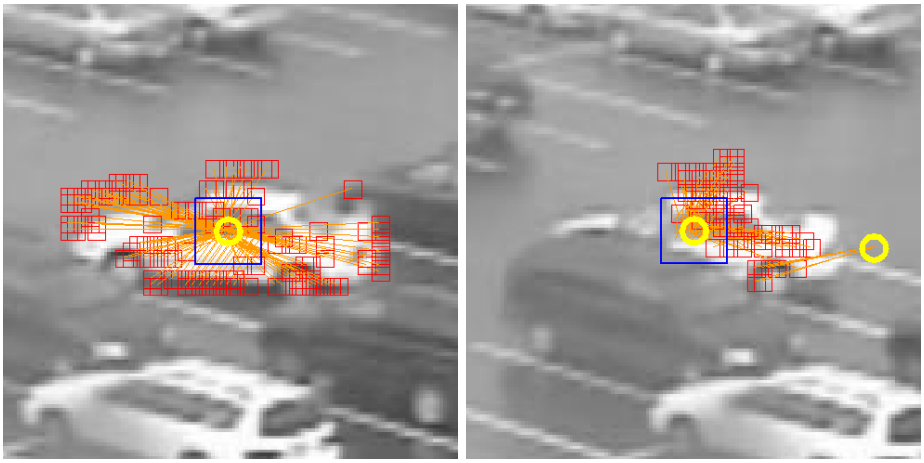

Figure 5. Red small squares show the locations of matching descriptors, orange lines show the tentative votes for the center of the object relative to each matching descriptor, large yellow circles show the cluster centers, large blue square is the center with the largest votes. Left: full view of the tracked object, right: occluded object (shown on non-WAMI data for clarity).

If this ratio is smaller than a threshold, this match is considered as a good match. The rationale behind this is to obtain only those matches that are substantially significant. If a c_k matches to many descriptors in SW with similar Euclidean distance, that particular c_k is not a good descriptor for the vehicle for that frame, but it still remains in the model because it may serve as a better descriptor in a different search window (e.g. a descriptor that represents a particular edge on the vehicle can match to many other descriptors in a given search window with similar edges, but not in an other one where there are no such edges in the background). Those matches (ij, k) that pass the ratio test are retained to compute the most likely location of the vehicle.

$$\mathbf{M} = \{(ij, k) | DR_{ij,k} < T_{DR}\} \tag{5}$$

where the image patch around x_{ij} represented by the descriptor d_{ij}^{SW} is a good match to the model descriptor c_k. $\mathbf{M}$ is the set of matching pairs and its cardinality is expected to be high because of the redundancy in the model as well as the fact that a search window in an urban area can contain very many similar images patches as the ones of a car. Even though they are dispersed through the search window, most of them are expected to be located on the vehicle of interest in frame $t + 1$. A simple approach would be to compute the average of the coordinates of scene descriptors that belong to these matches and provide it as the hypothesized centroid of the vehicle. But since there are many sporadic matches, this approach can bias the centroid and drift it beyond the actual vehicle. Instead, we utilize the mean shift process again, this time on the spatial domain of matching scene descriptors d_{ij}^{SW} to find the mode of their coordinates. This is a robust way to deal with the outliers due to sporadic matches, but it can still produce many clusters with similar cardinality. Instead of taking their absolute coordinates, we add the relative coordinates r_k of each c_k to the coordinate of the corresponding match d_{ij}^{SW}. This enforces the spatial coherency by reducing the number of clusters in the search window, because now every matching descriptor effectively votes towards to most likely centroid of the vehicle regardless of its position.

Figure 5 shows an example of this process. The red small squares are the locations of matching d_{ij}^{SW} after the ratio test. The orange lines represent the relative coordinates r_k of the corresponding c_k that are added to (i, j). If indeed these are good matches, all of them should point towards the most likely location of the centroid. This also handles occlusions gracefully as shown in the same figure. As the vehicle gets occluded, the remaining matches point (vote) towards to correct centroid, eliminating abrupt jumps in the location of the centroid due to the changing support map of the vehicle (demonstrated in this figure on a regular size vehicle for clarity). It is also likely that there may be very few good matches in a given search window. In that case, the tracker relies on the prediction rather than the weak matches to keep the search window on the most likely position of the car in the next frame. If there are no good matches for a predetermined number of frames, the tracker quits in order to let the human analyst to intervene rather than drifting in the scene.

Figure 6. Unstabilized versus stabilized trajectories of two fixed points from Philadelphia sequence. Stabilization successfully eliminates substantial amount of noise associated with the point locations.

2.1.1 Robust model adaptation

It is clear that this model has to be adapted to changing appearance as sampled in Figure 4. To facilitate that, we pick the top best matches in every frame and replace c_k with their corresponding d_{ij}^{SW}. This ensures that the model is only partly adapted, retaining some of the old descriptors, but also quickly adapting to the best matches. For that we compute the average Euclidean distance of matches $\overline{e}_{ij,k}$, if it is less than a threshold, those matches that are better than this average replace the corresponding c_k. This process may still cause a drift in the tracker if there are very few matches in a search window which happen to be sporadic matches due to similar image patches in the background. This is specifically the case when the vehicle is totally occluded yet there are a few good matches in the scene. To avoid that, we also check the number of matches to make sure there are enough good matches to update the model. This is where redundancy in representation again helps to keep the tracker away from confusions in the background.

2.2 Local Stabilization for Accurate Prediction

Predicting the location of the search window in the next frame is a key step of the proposed tracking approach. For the prediction step, we use the Kalman filter[16,17] which is a recursive filter that estimates the state of a linear dynamic system from a series of noisy measurements. Kalman filters are based on linear dynamical systems discretized in the time domain. They are modeled on a Markov chain built on linear operators perturbed by Gaussian noise. The state of the system is represented as a vector of real numbers. At each discrete time increment, a linear operator (dynamic equation or state transition model)

$$\mathbf{x}(k+1) = \mathbf{F}(k)\mathbf{x}(k) + \mathbf{v}(k) \tag{6}$$

is applied to the state to generate the new state, with some noise mixed in, and optionally some information from the controls on the system if they are known. Then, another linear operator (measurement equation or observation model)

$$\mathbf{z}(k) = \mathbf{H}(k)\mathbf{x}(k) + \mathbf{w}(k) \tag{7}$$

mixed with more noise generates the observed outputs from the true ("hidden") state. In Eq. 6 and Eq. 7, k is time step, $\mathbf{x}(k)$ denotes state vector, $\mathbf{v}(k)$ denotes process noise (zero-mean Gaussian with covariance $\mathbf{Q}(k)$), $\mathbf{z}(k)$ denotes measurement vector, and $\mathbf{w}(k)$ denotes measurement noise (zero-mean Gaussian with covariance $\mathbf{R}(k)$). System is modeled by matrices: $\mathbf{F}$,$\mathbf{H}$,$\mathbf{Q}$,$\mathbf{R}$ that are assumed to be known. In this system $\mathbf{F}$ is the state-transition model; $\mathbf{H}$ is the observation model; $\mathbf{Q}$ is the covariance of the process noise; $\mathbf{R}$ is the covariance of the

observation noise. State estimation cycle using Kalman filter consists of two steps: (1) state and measurement prediction; (2) state update. For our tracker we used a constant velocity model. State vector $\mathbf{x}$ contains position and velocity of the tracked car

$$\mathbf{x} = [x \;\; y \;\; v_x \;\; v_y]^T. \tag{8}$$

State-transition model (state matrix) is

$$\mathbf{F} = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{9}$$

Observation Model (measurement matrix)

$$\mathbf{H} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \tag{10}$$

covariance values for the process noise $\mathbf{Q}$ and the observation noise $\mathbf{R}$ are adjusted according to the input image sequence characteristics. To obtain an accurate prediction, a smooth trajectory is required. Even though Kalman filter smooths the trajectory under the assumption of gaussian noise, it is not sufficient in case of persistent wide-area motion imagery because the variance is very large due to large geo-registration errors and inconsistencies due to multiple camera seams. To improve the Kalman prediction, we run the Kalman filter on the stabilized trajectory by stabilizing two consecutive frames in a local neighborhood (such as 512×512) using FTT-based registration. Fourier methods are preferred for this application due to their speed and robustness to noise, occlusion and time varying illumination change. FFT-based registration method is based on the Fourier shift theorem. If two images differ only by a translation, their DFT will be shifted relatively in phase, hence the cross-power spectrum can be computed to factor out the phase difference.

$$R(u,v) = \frac{\mathcal{F}(I_0)\mathcal{F}(I_1)^*}{|\mathcal{F}(I_0)\mathcal{F}(I_1)^*|} = e^{2\pi i(\frac{u\Delta x}{M} + \frac{v\Delta y}{N})} \tag{11}$$

where I_0 and I_1 are consecutive images that differ by a translation of $(\Delta x, \Delta y)$, M and N are image dimensions and R is the normalized cross-power spectrum whose inverse Fourier transform will give peak at $(\Delta x, \Delta y)$. Once translation of the local region around the object is found, the image pair can be stabilized with respect to a reference frame and Kalman predictor performs on the stabilized smooth trajectory. Each new frame is stabilized with respect to previous frame and a cumulative translation is applied to the new centroid in order to put it in the coordinate system of the very first frame. This ensures that the prediction step operates always on the stabilized trajectory. Since the local region may contain structures at different heights causing different degrees of parallax, the translation between frames will be based on a virtual ground plane that may not be accurately on the actual ground. This will cause a slight drift in the stabilized images, hence a fixed point on the ground may follow an elliptical orbit throughout the sequence even after stabilization. While this may be a handicap for other processes such as background modeling and motion detection, it has little consequence in our case because most of the abrubt jumps are eliminated and the resulting trajectories are smooth albeit a slow drift. This stabilized trajectory is smooth enough to enable accurate prediction in Kalman filter. Figure 6 shows stabilized versus unstabilized locations of two fixed points (on the road and on the roof) throughout several frames in the Philadelphia sequence. The point on the roof follows an elliptical orbit due to parallax as expected. The point on the ground also follows a similar orbit in the opposite direction because stabilization vectors are computed with respect to an average virtual plane that effectively lies between the actual ground plane and buildings of different heights. Due to existence of many structures of different height, an accurate stabilization is challenging but the substantial amount of noise can be suppressed successfully as shown. Figure 7 shows a sample of a stabilized versus unstabilized trajectory produced by the tracker for a tracked vehicle in Charlotte sequence. Stabilized trajectory tends to align well with the road that is important for visual interpretation and its smooth varying nature makes it possible for Kalman filter to predict accurately the next location of the vehicle.

2.3 Implementation Details

After thorough experimentation, the model parameters are set as following, scale $s = 2$, grid length $q = 2$, DR threshold $T_{DR} = 0.75$, number of maximum predicted frames before tracker quits is 6. The thresholds to update

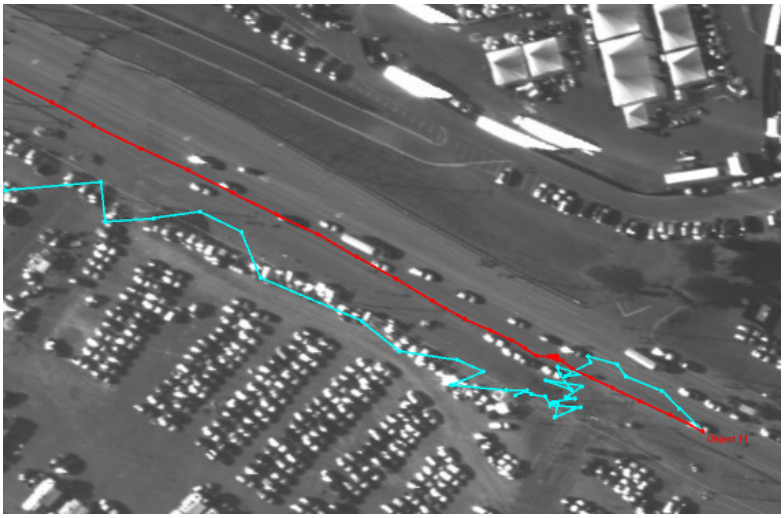

Figure 7. A sample for stabilized vs. unstabilized trajectories produced by the tracker from Charlotte sequence.

the model are set based on the altitude of the imaging platform, this is basically a function of image resolution, the same tracker can be used for different resolutions ranging from WAMI to mounted surveillance cameras (as in Figure 5). The meta data that is available within the sequence allows us to compute the geo-coordinates of the aircraft and the target vehicle, and consequently the height difference between the aircraft and target, the viewpoint of cameras with respect to the vehicle, etc. which both serve as outputs to the user as well as for setting up tracking parameters. In the future, this data will be utilized to provide multiple models with respect to viewpoint and also for occlusion handling together with depth maps. One important aspect to note is that there is no explicit vehicle model in the developed tracker so it can basically track anything that user marks with a bounding box, such as a boat on water or a train on tracks. The tracker is seamlessly integrated with two separately developed visualization systems, and serves human analysts in real time. It can be used on any type or format of image data by setting appropriate parameters for different resolutions.

3. EXPERIMENTAL RESULTS

In this section, we describe the results of automated testing of the interactive tracker on wide-area motion imagery data sets and also comment on the performance on other standard data sets.

3.1 WAMI Data Sets

As described before, the major challenges in the wide-area data set are the small scale of vehicles, abrupt jumps in the image due to geo-registration errors and crowded urban scene rich with man made structures and dense traffic. Our tracker is being actively used in a persistent surveillance system for scene forensic analysis. A major design consideration in the interactive setting is to minimize the number of interventions by the human analyst. After the analyst initializes the tracker to track a particular vehicle that is suspect in some scenario, the tracker has to start running with no substantial lag time and keep up with real time in order to let the analyst intervene whenever the tracker fails. The number of interventions for a given vehicle in a given data set has a large variation due to many factors, including but not limited to: 1) the resolution of the image (altitude of the aircraft and effective focal length), 2) density of the traffic, 3) amount of occlusions (higher structures cause stronger parallax hence increasing occlusions), 4) location of the vehicle with respect to camera array (farther objects incur more distortion and blurring), 5) other effects such as specular reflections, 6) amount of IMU noise, 7) accuracy of the digital elevation model for a given urban area, 8) persistence of the tracked object in the scene (length of trajectory). Because this multitude of parameters causes drastic changes in the performance of a tracker, a meaningful quantitative evaluation is very challenging. For the data sets that our tracker is developed on, the average number of analyst interventions per vehicle varies from 3 to 15 depending on the aforementioned factors.

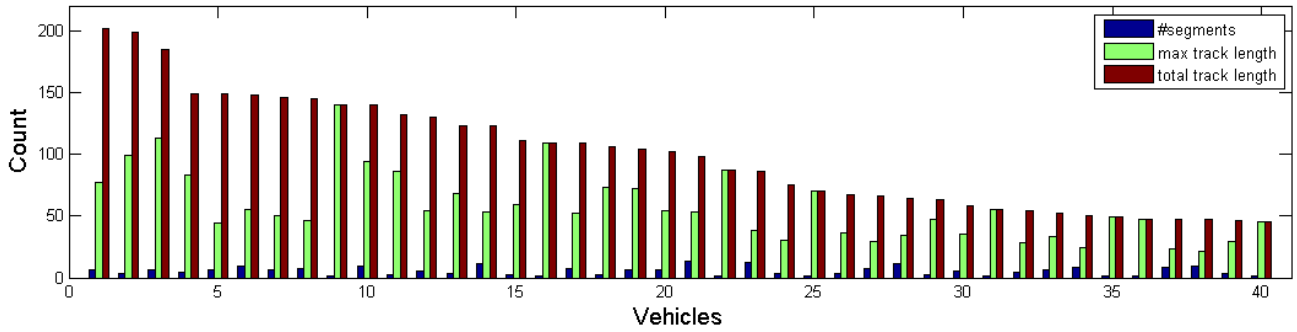

Figure 8. Results of testing with 40 vehicles in Charlotte sequence. Bars from left to right represent number of segments, total (ground truth) track length, and maximum track length sorted with respect to total track length for each vehicle.

Similarly, average length of tracks produced by the tracker has a wide variety in the range of 30-150 frames per vehicle at 1 fps. Vehicles that move slowly or stop for long durations can be tracked for several minutes, but the constant change in the amount of distortion and strong parallax that causes occlusions can eventually cause the tracker to lose these vehicles. Another important performance criterion is the gain in analyst time when doing a full manual track analysis versus assisted tracking for a set of cars involved in a scenario. The developed tracker helps the analysts to analyze a scenario about two to three times faster than manual tracking. Figure 8 summarizes results of automated testing of interactive tracking on Charlotte sequence. This sequence has major abrupt jumps across frames as well as across camera seams. A subset of cars in a local region were manually tracked to obtain ground truth, and an automated testing suite that was developed for this purpose was used to drive the tracker simulating analyst interventions whenever the tracker failed. 40 cars that persist in the selected region of 3000 by 1550 pixels for about 50 frames or more have been chosen for this experiment. They were manually tracked for a total of 3978 track points. Statistics collected from this experiment are number of segments (i.e. interventions) per tracked vehicle, maximum track length among segments per vehicle, average track length per vehicle and average track length percentage with respect to total ground truth per vehicle. A common measure that is used in the evaluation of tracking algorithms is the track fragmentation which is the average number of segments (i.e. tracker failures) per vehicle. In our case, the average number of interventions required by the analyst reflects the same measure and it is 5 for this subset of vehicles. Average track length per segment is 34 frames, and average length of ground truth tracks is 99 frames. Figure 9 shows some results of this set of cars. As seen from the unstabilized trajectories, this is a very challenging sequence in terms of vehicle localization due to major geo-registration and stabilization errors, abrupt jumps and road traffic. The interactive tracker requires on average four or five interventions per vehicle by the analyst.

3.2 Target Detection Performance

In this experiment we evaluate the target detection performance of the proposed tracker independent of the performance of other steps such as stabilization and prediction in a similar way as reported by Palaniappan et al.[4] In order to do that, we feed the tracker with perfect stabilization and prediction of car locations and compare the detection performance to a set of two block correlation-based features and three local histogram-based features. Intensity and gradient magnitude normalized cross correlation (Corr-I and Corr-GradMagI) are chosen for block correlation-based features. Local histogram based features are local intensity histogram (Hist-I), local intensity gradient magnitude histogram (Hist-GradMagI) and histogram of oriented gradients (HOG). HOG has been successfully used in many recent object detection and particularly people detection applications.[18–20] HOG computes the histogram of gradient orientations weighted by corresponding gradient magnitudes over a patch or small region of an image. Five cars that represent several challenges (distortions, occlusions, turns, abrupt jumps, stops, etc.) in Philadelphia sequence are manually tracked for a total of 874 frames for this experiment. Target likelihood maps of cars are computed by using a sliding window histogram differencing scheme. Local maxima in likelihood maps are considered as possible target locations. They are ranked with respect to their likelihood and recall is computed by evaluating their distance to the actual car location. Figure 10

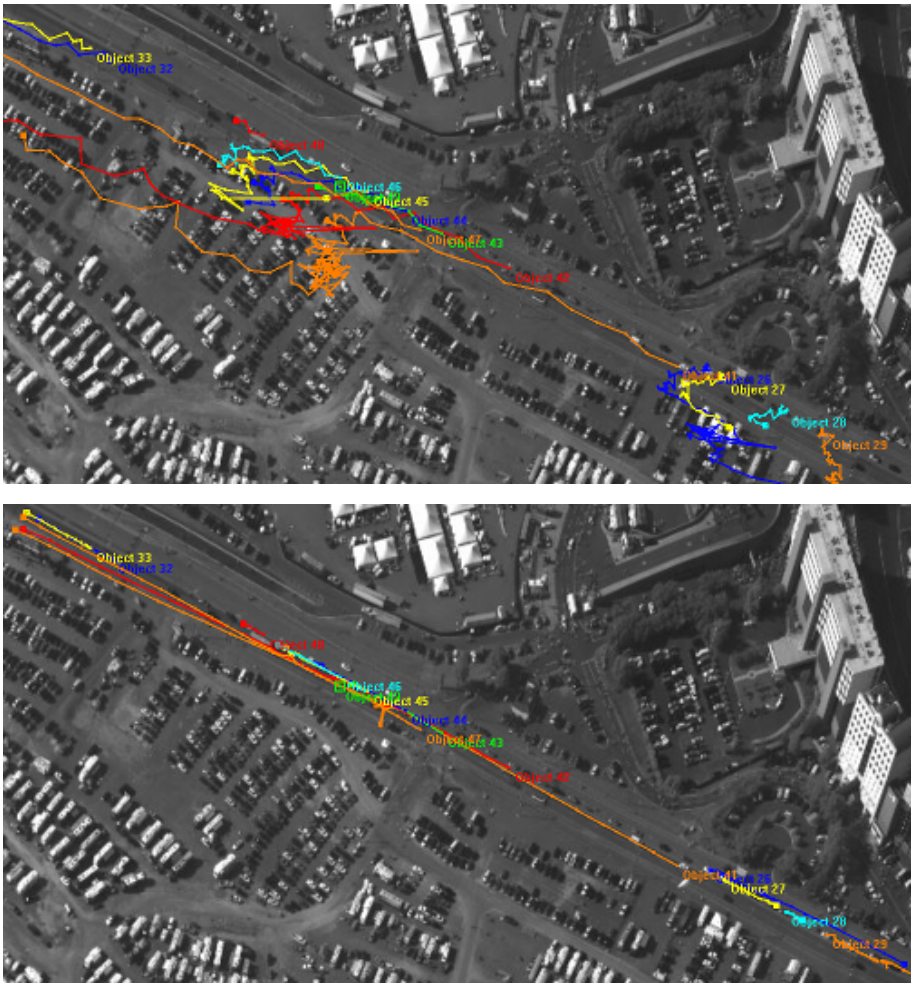

Figure 9. A subset of 40 cars are tracked in Charlotte sequence for testing through the span of 960 frames in a region of 3000 by 1550 pixels (16 cars are visible in this snapshot). top: unstabilized, bottom: stabilized trajectories produced by the tracker.

shows the aggregate detection performance. As shown in the figure, the proposed scheme of adaptive model of dense descriptors outperforms other five features by accurately detecting targets among top ten peaks.

3.3 Standard Data Sets

Even though the proposed tracker is developed for interactive forensic analysis in wide-area surveillance where the objects of interest are very small (20×30 pixels), it is worthwile to compare the performance to other trackers in the recent literature using standard data sets such as tracking faces. In order to do that, we modified our tracker to run on standard resolution data sets reported by Gu et al.[11] The changes made to the tracker were to increase the scale of the dense SURF descriptors and modify the thresholds for model update and lost target decision. We also removed the stabilization. Gu et al.[11] report scores based on Euclidean distance between ground truth tracks and tracker-produced tracks ranging from 13 pixels to 79 pixels. Our tracker was competitive in six out of eight reported sequences ranging from 15 pixels to 43 pixels, obtaining best score in some sequences. We also computed the same Euclidean distance measure for our tracker's results on our test set of 40 cars and obtained an average error of 16 pixels.

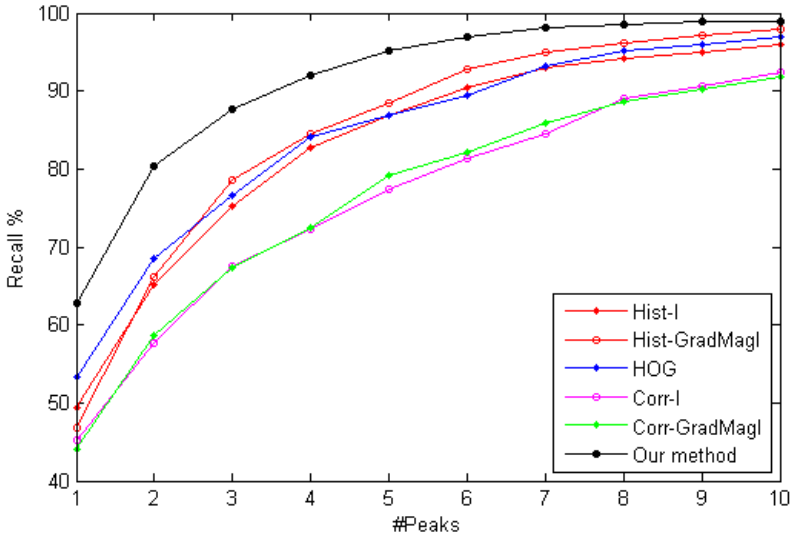

Figure 10. Average recall versus number of peaks selected for each feature vector for five manually tracked cars over a total of 874 frames from Philadelphia sequence.

4. CONCLUSIONS

Developing an interactive tracker for a wide-area surveillance system to track several objects for forensic analysis rather than tracking hundreds of vehicles for aggregate statistical analysis has unique challenges. Every individual vehicle in a scenario has to be tracked trough the whole image accurately In the case of forensic analysis, the interaction with the analyst requires real time response on a vehicle by vehicle basis since every frame has to be validated by the analyst, and it does not permit setup time or any other precomputation that causes major lag in response time. The interactive tracker has to work on image chips which are "on the move", effectively leading to a local virtual moving camera on the whole 16K by 16K image. This requires accurate prediction of the next image chip location and, as described before, an on-the-fly local stabilization step is needed to improve the accuracy of prediction. The switching between automatic and manual tracking modes in a seamless manner is also an important design consideration, this interactive nature of the tracker is especially useful when the analyst gets used to the behavior of the tracker and can anticipate where the tracker will fail so that she can go through those situations manually, and handing over to the tracker again for automated tracking. This makes sure that the tracked vehicle is accounted for in every frame in an unambiguous way. One of the major advantages of the developed tracker is that it does not lose the targets due to lack of motion when they stop, this is valuable especially in urban settings where waves of traffic regularly stop at intersections. Using a dense set of features adds robustness to the vehicle model since at the given scale (20×30 pixels) there are no discernible interest points that an interest point detector can reliably and repeatably find. In addition to small size, distortion and blur also reduces the sharp features that could serve as interest points. Since dense descriptors of vehicles will have an abundance of matches in an urban setting (a small image patch at the edge of a vehicle can match another one at the edge of a window or roof top), careful construction of clustering schemes to filter out these matches plays an important role. By applying mean shift clustering to both vehicle descriptors in descriptor space and coordinates of matched set in spatial domain, we find the major clusters without being confused by the multitude of sporadic matches. The developed tracker is actively being used as a part of an interactive wide-area surveillance system with civilian applications towards scene forensics. It can handle poor image conditions and can keep up with real-time for around 2-4 fps scalable with the number of processor cores. It should be noted that we have not yet pursued any major optimization towards the speedup of the algorithm. Dense SURF descriptors can be computed very efficiently due to substantial overlap of box filter and summation operations. The matching step can be sped up by using KD-trees or other techniques. And finally, all three major bottleneck operations (descriptor computation, matching, stabilization with FFT) are very good candidates for fine grain

parallelization and can be offloaded to GPUs. These optimizations can reduce the computational cost drastically. We also experiment with fusing information from street maps and depth maps of lidar data where available in order to improve tracking performance. Depth maps, if aligned properly with the imagery, provide valuable occlusion information that is computed by utilizing the map and the meta data such as platform geo-coordinates and target vehicle geo-coordinates that we already compute as outputs of our tracker. This occlusion information can be used to inform the tracker for oncoming occlusions and where the vehicle is expected to be visible again. Our future work includes the aforementioned optimizations and increasing the tracking performance of the tracker by improving the vehicle model, augmenting it with meta data, lidar data, and different adaptation models in order to accommodate for more abrupt changes in the target appearance as well as robust handling of occlusions.

ACKNOWLEDGMENTS

The authors wish to thank Dr. Ross McNutt of Persistent Surveillance Systems Inc. for providing the wide-area imagery used in this paper, Anoop Haridas and Joshua Fraser for the new version of the Kolam software tool[21] to support visualization of wide-area airborne video and trajectories produced by the tracker and creating the Maya-based rendering of the airborne imaging platform flight path and geometry and Rengarajan Pelapur for automated testing of tracker performance on PSS data.

This research was partially supported by grants from the U.S. Air Force Research Laboratory (AFRL) under agreements FA8750-11-1-0073, FA8750-10-1-0182, FA8750-11-C-0091 and Leonard Wood Institute (LWI 181223) in cooperation with the U.S. Army Research Laboratory (ARL) under Cooperative Agreement Number W911NF-07-2-0062. Approved for public release (case 88ABW-2012-2498). The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied of AFRL, LWI, ARL or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation thereon.

REFERENCES

[1] Palaniappan, K., Rao, R., and Seetharaman, G., "Wide-area persistent airborne video: Architecture and challenges," in [*Distributed Video Sensor Networks: Research Challenges and Future Directions*], Banhu, B., Ravishankar, C. V., Roy-Chowdhury, A. K., Aghajan, H., and Terzopoulos, D., eds., 349–371, Springer (2011).

[2] Porter, R., Fraser, A. M., and Hush, D., "Wide-area motion imagery," *IEEE Signal Processing Magazine* **27**(5), 56–65 (2010).

[3] Carrano, C. J., "Ultra-scale vehicle tracking in low spatial resolution and low frame-rate overhead video," in [*SPIE Proc. Signal and Data Processing of Small Targets*], Drummon, O. and Teichgraeber, R. D., eds., **7445** (2009).

[4] Palaniappan, K., Bunyak, F., Kumar, P., Ersoy, I., Jaeger, S., Ganguli, K., Haridas, A., Fraser, J., Rao, R., and Seetharaman, G., "Efficient feature extraction and likelihood fusion for vehicle tracking in low frame rate airborne video," in [*13th Int. Conf. Information Fusion*], (2010).

[5] Reilly, V., Idrees, H., and Shah, M., "Detection and tracking of large number of targets in wide area surveillance," in [*11th European Conf. Computer Vision*], 186–199, Springer-Verlag (2010).

[6] Cuntoor, N. P., Basharat, A., Perera, A. G. A., and Hoogs, A., "Track initialization in low frame rate and low resolution videos," in [Proc. IEEE Int. Conf. Pattern Recognition], 3640–3644 (2010).

[7] Jiangjian, X., Hui, C., Sawhney, H., and Feng, H., "Vehicle detection and tracking in wide field-of-view aerial video," in [Proc. IEEE Conf. Computer Vision Pattern Recognition], 679–684 (2010).

[8] Rosenbaum, D., Kurz, F., Thomas, U., Suri, S., and Reinartz, P., "Towards automatic near real-time traffic monitoring with an airborne wide angle camera system," *European Transp. Res. Rev.* **1**, 11–21 (2009).

[9] Ma, X. and Grimson, W. E. L., "Edge-based rich representation for vehicle classification," in [Proc. IEEE Int. Conf. Computer Vision], **2**, 1185–1192 (2005).

[10] Babenko, B., Yang, M. H., and Belongie, S., "Visual tracking with online multiple instance learning," in [Proc. IEEE Conf. Computer Vision Pattern Recognition], 983–990 (2009).

[11] Gu, S., Zheng, Y., and Tomasi, C., "Efficient visual object tracking with online nearest neighbor classifier," in [*10th Asian Conference on Computer Vision (ACCV2010)*], (Nov. 2010).

[12] Bay, H., Ess, A., Tuytelaars, T., and Gool, L. V., "Speeded-up robust features (surf)," *Computer Vision and Image Understanding* **110**, 346–359 (2008).

[13] Lowe, . D. G., "Distinctive image features from scale-invariant keypoints," Intern. J. Comput. Vis. **60**(2), 91–110 (2004).

[14] Fukunaga, K. and Hostetler, L., "The estimation of the gradient of a density function, with applications in pattern recognition," *IEEE Transactions on Information Theory* **21**(1), 32–40 (1975).

[15] Cheng, Y., "Mean shift, mode seeking, and clustering," *IEEE Transactions on Pattern Analysis and Machine Intelligence* **17**(8), 790–799 (1995).

[16] Bar-Shalom, Y., Li, X. R., and Kirubarajan, T., [*Estimation with Applications to Tracking and Navigation*], Wiley-Interscience, 1 ed. (jun 2001).

[17] Welch, G. and Bishop, G., "An introduction to the kalman filter," Tech. Rep. TR 95-041, University of North Carolina at Chapel Hill (may 1995).

[18] Dalal, N. and Triggs, B., "Histograms of oriented gradients for human detection," in [Proc. IEEE Conf. Computer Vision Pattern Recognition], **1**, 886–893 (2005).

[19] Chen, L., Feris, R., Zhai, Y., Brown, L., and Hampapur, A., "An integrated system for moving object classification in surveillance videos," in [*Proc. IEEE Int. Conf. Advanced Video and Signal Based Surveillance*], 52–59 (2008).

[20] Han, F., Shan, Y., Cekander, R., Sawhney, H. S., and Kumar, R., "A two-stage approach to people and vehicle detection with hog-based svm," in [*Performance Metrics for Intelligent Systems Workshop*], (2006).

[21] Haridas, A., Pelapur, R., Fraser, J., Bunyak, F., and Palaniappan, K., "Visualization of automated and manual trajectories in wide-area motion imagery," in [*Information Visualisation (IV), 2011 15th International Conference on*], 288–293, IEEE (2011).