2009-9

Efficient Tracking of Many Objects in Structured Environments

Authors: Nathan Jacobs, Michael Dixon, Scott Satkin, Robert Pless

Corresponding Author: jacobsn@cse.wustl.edu

Abstract: We consider the special case of tracking objects in highly structured scenes. In the context of vehicle tracking in urban environments, we offer a fully automatic, end-to-end system that discovers and parametrizes the lanes along which vehicles drive, then uses just these pixels to simultaneously track dozens of objects. This system includes a novel active contour energy function used to parametrize the lanes of travel based only on the accumulation of spatio-temporal image derivatives, and a tracking algorithm that exploits longer temporal constraints made possible by our compact data representation; we believe both of these may be of independent interest. We offer quantitative results comparing tracking results to ground-truthed data, including thousands of vehicles from the NGSIM Peachtree data set.

Type of Report: Other

Efficient Tracking of Many Objects in Structured Environments

Nathan Jacobs, Michael Dixon
Washington University
St. Louis, MO, USA
(jacobsn|msd2)@cse.wustl.edu

Scott Satkin
The Robotics Institute
Carnegie Mellon University
satkin@cmu.edu

Robert Pless
Washington University
St. Louis, MO, USA
pless@cse.wustl.edu

Abstract

We consider the special case of tracking objects in highly structured scenes. In the context of vehicle tracking in urban environments, we offer a fully automatic, end-to-end system that discovers and parametrizes the lanes along which vehicles drive, then uses just these pixels to simultaneously track dozens of objects. This system includes a novel active contour energy function used to parametrize the lanes of travel based only on the accumulation of spatio-temporal image derivatives, and a tracking algorithm that exploits longer temporal constraints made possible by our compact data representation; we believe both of these may be of independent interest. We offer quantitative results comparing tracking results to ground-truthed data, including thousands of vehicles from the NGSIM Peachtree data set.

1. Introduction

The recent deployment of very large-scale camera networks and persistent aerial imaging sensors has led to a unique version of the tracking problem, with the goal of detecting and tracking every vehicle within a large urban area. In this paper, we exploit the constraints inherent in urban environments – that while there are potentially many vehicles, they follow relatively consistent pathways – to develop a real-time, fully automated system to track a very large number of vehicles.

Long-term persistent surveillance data is increasingly being collected and archived; while there are numerous privacy concerns, efficient algorithms for simultaneously tracking many objects from this data supports a variety of public safety goals. These include goals related to individual vehicles, such as tracking vehicles involved in hit-and-run incidents or finding the origin of illegal dumping operations. Additionally, summary statistics of the trajectories of vehicles as they move through a city and acceleration profiles of vehicles along those tracks are key data elements in the design of modern transportation systems.

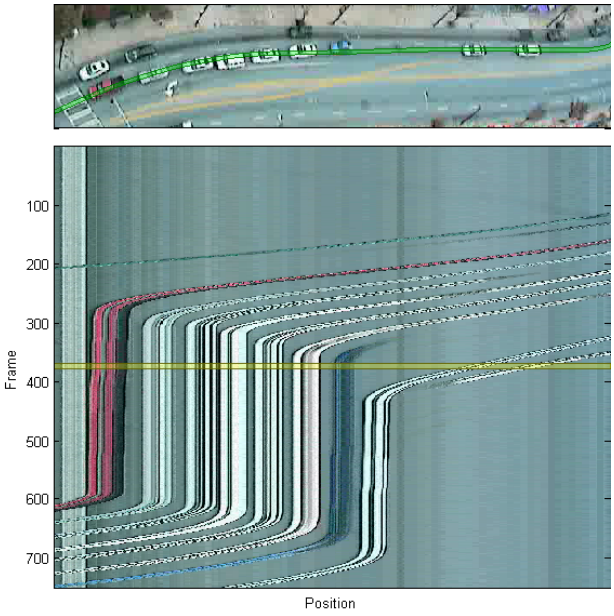


Figure 1. A space-time sheet is an image constructed from image data extracted over time from a curve in image space. We show that in structured scenes, such as cars driving on roads, it is possible to automatically parametrize the paths of consistent motion and obtain accurate tracking results using only the data contained in these sheets.

Continual analysis of these trends can help to determine, for instance, whether gas prices are encouraging people to increase mileage by driving more efficiently (predominantly by accelerating more slowly), and this analysis is possible given an efficient system for collecting trajectories of large sets of vehicles.

Furthermore, tracking cars within a city is a proxy problem for a growing class of important problems requiring the tracking of objects moving along initially unknown, but highly structured paths. One example of this is tracking T-cells moving through the lymph system; these cells can be observed to move along specific paths within the lymph system using two-photon microscopy, a novel in-vivo imaging technique that gives 2- and 3-D movies of fluorescently

tagged cells in living tissue. Observations of the paths T-cells, including how many take each path and how quickly they move through the lymph node are important in detecting immune diseases [12].

One of the key challenges in large scale tracking with modern imaging systems is the amount of data that is generated; this is true both for the case of persistent aerial video where bandwidth and storage are critical limiting issues, and in camera networks where there are additional organizational challenges in dealing with data feeds at different locations. Learning the structure within the scene and capturing the "right" data (that is, the data that is most useful for tracking), is one key approach to allowing these systems to continue to scale up. Furthermore, in some of the biological imaging systems, learning the structures in the scene is itself of biological interest, and sampling fewer voxels within the 3D volume prevents the slow cell death caused by the two-photon imaging systems [4].

Therefore, this paper considers the problem of tracking in highly structured environments, with a focus on learning the scene structure, and tracking using a small fraction of the image pixels. We propose a novel, fully automatic, real-time system specialized for tracking vehicles. Given video data as input, the system learns the road and lane structure of the scene by observing consistent local motion patterns and uses pixel values captured only along the middle of the lane to support multi-object tracking of dozens of vehicles at a time.

Our contributions include the novel overall framework of this system, demonstrating the performance of our Matlab implementation, which is capable of simultaneously tracking dozens of objects at 25 FPS, and evaluating the tracking results against a large, ground-truthed data set prepared by the Next-Generalization Simulation (NGSIM) traffic simulation community. Additionally, this implementation includes a novel active-contour energy formulation, which we use to parametrize the lanes of travel directly from spatio-temporal image derivatives in video data, and the novel use of extended temporal data along possible trajectories to improve and simplify vehicle tracking.

Figure 1 illustrates this paradigm. For each lane, we can summarize the long-term temporal behavior along that line by creating a 2D "sheet" that shows the intensity values along the pixels within the lane over time. This sheet shows the vehicles passing through that lane, reducing a tracking problem from 3D (two spatial dimensions and time) to 2D.

1.1. Related Work

Using Space-Time Sheets Reasoning about spatio-temporal volumes has become extremely popular with the increasing memory capacity and speed of computers. Explicit reasoning about spatio-temporal slices has received attention only in specialized problem domains. Spatio-temporal slices were first used in the context of regularizing structure from motion when the camera motion stays constant [5], and more recently to reason about specular reflections [6].

Within the context of human activity recognition, spatio-temporal slices have been shown to be an effective and compact representation for applications such as person detection and people counting [16], tracking [13, 17] and behavior recognition [9].

Road Extraction The problem of road delineation has been widely studied, since there is a tremendous need to automatically generate geographic information systems. Traditional methods of road extraction utilize static satellite imagery and are based solely on the appearance of roads, though some recent methods have used video data [10, 18].

Numerous approaches to road extraction make use of *Active Contours*, an algorithm originally introduced by Kass et al. [8]. Melonakos et al. [11] redefine Active Contours with an energy term directing a snake's orientation to fit known orientations within the image using Finsler Metrics [3]. These snakes have directional cost functions that make it much cheaper to align along a road in the direction of the roads travel. They illustrate their methods to segment road imagery. We view our work as addressing two key limitations of this previous work. First, although Melonakos et al. optimize open-ended snakes instead of closed contours, these snakes have fixed endpoints; in contrast, we allow the endpoints of the snake to move freely to force lengthening and alignment with roads. Additionally, the "directional data" used in [11] is not derived from motion; rather, it is artificially generated by fitting a static image template along roads. In contrast, we use the accumulation of spatio-temporal image derivatives suggested by [15] to define a mean vector field of observed motion within the scene. This gives both a direction and a magnitude of motion, giving a different type of underlying directional data to which we would like to fit a snake.

Location-Specific Motion Priors Closely related to the present work is research into effective use of motion priors in object tracking [7]. Jacobs et al. propose to condition the motion likelihood term on the tracked objects image location. Our work represents an extreme form of location-specific motion priors, in which the motion priors are introduced as hard constraints rather than a modification of the motion likelihood term. This view greatly decreases the amount of image data that must be considered resulting in a significantly faster algorithm.

Multi-Object Tracking There is a large body of work on multi-object tracking; see the article by Yilmaz [19] for an overview. The two most relevant recent works [20, 14] use

network flow-based methods to extract tracks from lower level primitives. Zhang et al. [20] use two-dimensional object detections as primitives. These primitives are linked as nodes in a min-cost flow problem. Perera et al. [14] use a bipartite matching framework to link tracklets, short segments of objects that are linked prior to data association. To our knowledge, our work is the first to consider the multi-object tracking problem using only image data sampled from a restricted set of curves.

1.2. System Overview

Our proposed system works in four stages. First, we fit a set of curves to the consistent motion patterns in the image (Section 3). Then we extract a set of space-time sheets from a video by sampling the image data along these curves (Section 2). Next, we process the sheets to generate a set of partial object tracks, or tracklets (Section 4). Finally, we perform a data association step to combine tracklets from one or more sheets into complete object tracks (Section 5).

2. Space-Time Sheets

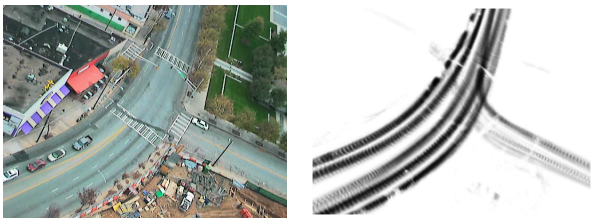
A space-time sheet, or simply a sheet, is an image constructed from video data extracted along a single curve in image space over a temporal interval. In our examples, the curve, which we call a sheet curve, is typically drawn along a lane of traffic. Pixels in a sheet are parametrized by a temporal (in all figures the downward direction is forward in time) and a spatial coordinate (position along the sheet curve in image space). In a single video sequence there may be one or many space-time sheets of interest depending on the application and the scene.

To construct a sheet given a sheet curve, c_k, and a video, I, we simply extract the colors of pixels under the curve, sampling every one pixel width, for each frame of the video, t.

$$S_k(p,t) = I(c_k(p),t).$$

Fig. 2 shows example sheets extracted from video of a traffic scene. The figure highlights the predictable nature of the appearance of cars traveling along sheets.

A significant benefit of working with the sparse sheet representation is the greatly reduced bandwidth requirements per second of video. For example, a 640×480 video with five lanes of traffic can be reduced to approximately 1% of the original data. It is conceivable that this process could be integrated deeper into the imaging system to reduce the image data bandwidth between the camera and the tracking system. This will be especially useful in bandwidth-constrained scenarios, such as gigapixel cameras or low-bandwidth wireless links.



(a) Example frame of video. (b) "Road Score" R.

Figure 3. An example of the road score for a video. Darker pixels indicate a higher value.

3. Road Delineation using Active Contours

Active contours, or "snakes," provide a framework for parametrizing 2D curves (splines) along image boundaries while balancing global smoothness constraints and local image features. In this section, we fit active contours to motion cues accumulated over time in video data. Traditionally, active contours have an energy function defined as the sum of an internal energy function and an external energy function:

$$E_{total} = E_{int} + E_{ext}$$

The classical formulation of the internal energy component encourages the spline to lengthen by maximizing the first derivative, while minimizing curvature by penalizing the second derivative:

$$E_{int}(\vec{s}(t)) = -\alpha \int \left|\left|\vec{s}'(t)\right|\right| dt + \beta \int \left|\left|\vec{s}''(t)\right|\right| dt$$

To specialize these for delineating roads in video, we propose a novel external energy term, which also has two parts. The first part directly encourages the snake to lie on image regions with high "road score" (defined in the next section and shown in Figure 3(b)). The second component of the external energy equation utilizes the optic flow field data and encourages the snake to align with the optic flow vectors. The intuition for this component of the energy function is that the spline should grow in the direction that cars travel.

$$E_{ext}(\vec{s}(t)) = -\gamma \int R(\vec{s}(t))^2 dt + \delta \int \left|\left|\vec{V}(\vec{s}(t)) - \vec{s}'(t)\right|\right| dt$$

These terms are formalized in the next section.

3.1. Motion and Spatio-Temporal Derivatives

To analyze the consistent motion patterns within a scene, we implement the method from [15], which offers a real-time, video-rate method to extract a vector field of the motions observed by a static camera. This method incrementally updates, at each pixel location p, an estimate of the optic flow vector $\vec{V}(p)$ that is most consistent with the spatio-temporal image derivatives observed at that exact location

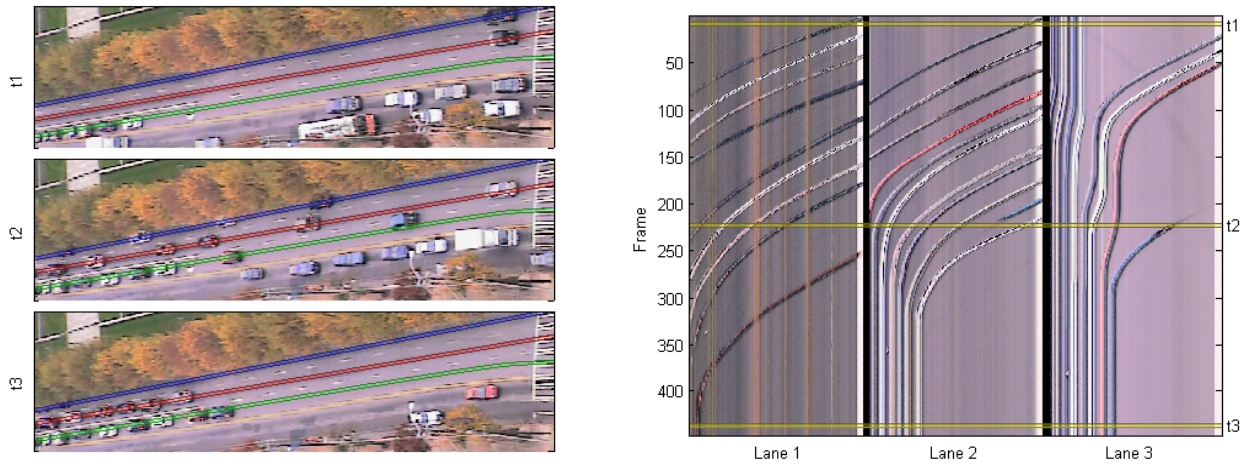


Figure 2. (left) Example frames from a video of a traffic scene at different frames. Three sheet curves are highlighted in red, green and blue, respectively. (right) An example of sheets extracted from the three marked lanes derived for 450 frames of video. The curved streaks visible in the sheets show the cars' progress moving to the left in the image, and coming to a stop for a light (just outside the field of view). The sheet on the left corresponds to the top lane; occlusions in that lane are visible as dominant vertical streaks. A lane change is also visible in this figure: a blue car enters the scene in lane two, its blue streak disappears as it changes lanes at about frame 225 and then appears in the sheet for lane three. Indeed, the original video frame on the left at time t2 shows between the red and green lanes.

over the entire length of the video. These optic flow vectors represent the typical direction of traffic flow at each location in the scene.

For each pixel, we also compute a "road score" $R(p)$, which characterizes how often a significant temporal variation is observed at each pixel

$$R(p) = \sum_{t=2}^{T} |I(x,y,t) - I(x,y,t-1)| > \epsilon_R$$

Since the dominant cause of pixel variation is the motion of vehicles, this term is a good metric for where roads are.

3.2. Implementation

We construct our snakes using a cubic B-spline, which ensures continuity and differentiability of the energy terms. The snake is discretized using uniform sampling with an average of one point per pixel covered. Since the snake lies over a continuous domain, we use bilinear interpolation to estimate $\vec{V}$ and R at non-integer pixel locations. The main algorithm proceeds as follows:

1. Choose a location on the image p with a high "road score" $R(p)$ and initialize a small initial snake in the direction $\vec{V}(p)$.

2. Iteratively update the snake based on the Euler-Lagrange differential equations of the energy function until the gradient descent converges.

3. Set the region of the road score image spanned by the snake to zero, to prohibit additional snakes from overlapping.

4. Repeat steps 1-3, until there are no remaining pixels with a high road score.

Because the length of each road is not known *a priori*, spline control points are dynamically added as the snake lengthens. Additionally, since snakes may get caught in local minima, the gradient descent may terminate before a snake has aligned with the full length of the road. To address this problem, we join any snakes with nearby endpoints.

4. Local Tracking within Space-Time Sheets

This section describes our method of generating a set of tracklets for a single sheet. This set of tracklets, along with tracklets from other sheets, is used as input to a global data association algorithm described in Section 5.

4.1. Detecting Objects on Sheets

The goal of our object detection method is to identify a set of high-quality starting locations that can be used as seeds for a local template-based tracker. Because the local tracker will be used to identify the object's path through the sheet, we do not rely on this detector to find the position of every object in every frame. Instead, our system requires a detector that outputs few false positives and a sufficient number of true positives to find each object at least once.

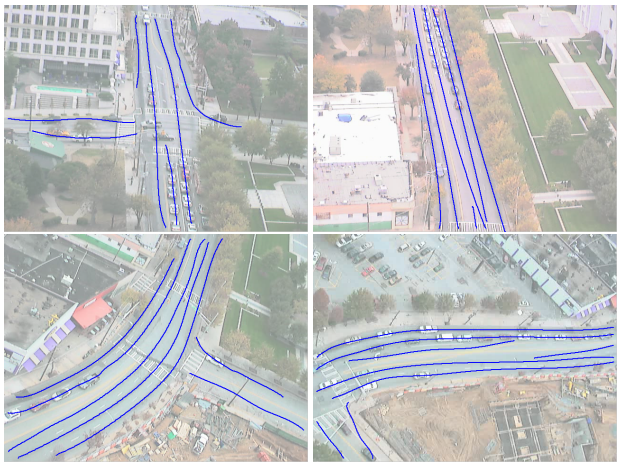
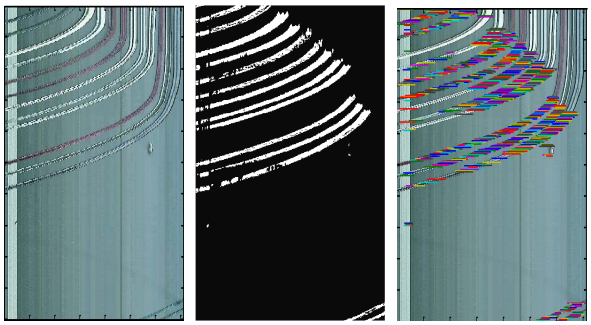


Figure 4. Results of our road delineation algorithm for the first four videos from the NGSIM Peachtree data set.



(a) (b) (c)

Figure 5. Intermediate steps of the detection process. (a) A sheet extracted from a road in which a set of cars are beginning to move. (b) A binary mask that indicates the locations of moving objects. (c) The set of object detections (each horizontal line represents an object detection).

We were guided by the observation that object detection is much simpler when objects are moving. Combined with background modeling, motion estimation can provide very strong cues for discriminating objects from the background. Additionally, in traffic scenes, moving objects are generally farther apart than stationary objects, making it far less likely that two different objects will be grouped into a single detection. Thus, we designed our detection algorithm to find regions of the sheet that have both a high foreground score and locally constant, non-zero velocity (See Fig. 5(b) for an example).

We obtain an estimate of the foreground score at F_k at each spatial location p along the sheet and each time t, using background subtraction, to give $F_k(p,t) = |S_k(p,t) - B_k(p)|$, where B_k is a background model describing the median pixel value at each position in S_k.

Following the work in consistent motion detection [15], we note that regions of consistent motion can be detected through properties of the local spatio-temporal structure tensor. In this case, that is the 2×2 matrix:

$$\vec{ST}(p,t) = \begin{bmatrix} \Sigma S_p^2 & \Sigma S_p S_t \\ \Sigma S_p S_t & \Sigma S_t^2 \end{bmatrix},$$

where S_p is short for the derivative of the intensity moving in the spatial direction of the sheet, S_t is the derivative of the intensity moving through time in the sheet, and the sums are taken over 5×5 pixel regions around the point (p,t).

The eigen decomposition of the structure tensor indicates the local motion consistency. Let $\vec{e_1}(p,t)$ be the first eigenvector of $\vec{ST}(p,t)$ and $\lambda_1(p,t)$ be the first eigenvalue. The first eigenvalue of the structure tensor indicates the magnitude of the dominant linear structure and the first eigenvector provides (the total-least squares) estimate of velocity. With these, we define a binary detector as follows:

$$O(p,t) = \begin{array}{rl} F(p,t) > \theta_{foreground} & \text{and} \\ \lambda_1(p,t) > \theta_{consistent} & \text{and} \\ \vec{e_1}(p,t) \cdot \langle 0,1 \rangle > \theta_{velocity} & \end{array}$$

The binary function $O(p,t)$ will be true in regions of the sheet that correspond to consistently moving foreground objects (see Fig. 5(b)). We convert this to a set of object detections by performing 1-D connected components analysis on each row (each time step) of O, and return the center locations and widths of each detection.

4.2. Tracklet Generation Within Sheets

This section describes our process for taking a set of object detections and converting them into a set of tracklets. To accomplish this, we use a local template tracker, taking advantage of the long temporal window by matching appearance data forward and backward in time. When complete, each tracklet ideally includes the entirety of the motion of a vehicle within a sheet. Tracklets will end when, for example, a vehicle changes lanes, and the data association described in Section 5 will connect those tracklets as well as broken tracklets within a sheet, into a final set of tracks of objects as they move throughout the video.

Essentially, our algorithm takes detections, runs a brute force template tracker, uses an exponential forgetting factor to update the appearance, stops when it runs into an existing track, off then end of the sheet, or when the appearance matches the background. Explicitly, we specify a tracklet by its position $x(t)$, width w, and appearance A over the range of frames $t \in T$. We initialize the template to be the intensity values of those pixels from the sheet at the starting time t_0, so that $A(t_0) = S_k([x(t_0) \ldots x(t_0) + w], t_0)$.

Tracking is then performed in both temporal directions. (Since the forward and backward directions are symmetric

we only describe forward tracking.) To find the next position, we search a range of pixel offsets in the next temporal row t for the offset a^* that minimizes a matching cost function as follows:

$$a^*(t) = \underset{a \in [a_{min} a_{max}]}{\arg\min} \left(D(\widehat{x}_j(t-1) + a, t) + k \cdot a^2 \right),$$

where $\widehat{x}_j(t) = x(t) + (x(t) - x(t-1))$ is the prediction of where the object would be if it continued on a constant velocity, $k \cdot a^2$ is a penalty on acceleration, and $D_j(p,t)$ is the SSD of the template and the sheet at position p and time t:

$$D_j(p,t) = \sum_{i=0}^{w_j} \left(S_k(p+i,t) - A_j(i,t-1) \right)^2.$$

The spatial location is updated with the new object location, $x_j(t) = \widehat{x}_j(t-1) + a^*(t)$, and the appearance template is updated with an exponential forgetting factor $A(i,t) = \alpha \cdot S(x_j(t) + i, t) + (1-\alpha) \cdot A(i, t-1)$.

Tracking then advances to the next temporal row of the sheet and continues until either the edge of the sheet is reached, the average foreground score is too low, $\frac{1}{w_j} \sum_{p \in \tau_j(t)} F_k(p,t) < \theta_{foreground}$, the tracklet overlaps an existing tracklet by more than half its width, or the final temporal row of the sheet is reached.

Once local tracking has terminated in both temporal directions, we remove any unassigned detections that overlap by more than half their width. We then select the remaining detection with the highest foreground score and repeat the above tracking procedure until no further detections remain.

5. Data Association Within and Across Sheets

This section describes the process for combining tracklets that are defined in sheet coordinates into tracks in image space. The input is a set of tracklets from a set of sheets and the world space relationship between the sheets.

We cast this data association problem using the min-cost-flow formulation [20], which minimizes a global cost function if it can be defined in terms of sums of functions defined over individual tracklets, pairs of consecutive tracklets and entry/exit points of tracks. The remainder of this section describes the choices we made for these functions, defined in Eqn 11 of [20].

The tracklet merit score, denoted as C_i in the previous work, defines how likely a given tracklet is to be a true positive. This score should be lower if the tracklet is more likely. We define it as follows:

$$C_i = \begin{cases} \frac{C d_i}{D} & D < d_i \\ C & \text{otherwise} \end{cases}$$

where d_i is the temporal duration of the tracklet, C is a negative number that represents the best possible track merit, and D is a cutoff point above which tracklet merit does not improve solely because it is temporally longer.

We define the tracklet linking score, denoted as $C_{i,j}$ in the previous work, as a combination of a simple appearance term and a motion compatibility term. Given a pair of tracklets, with one identified as the earlier, the goal for the linking score is that it be lower if the pair is more likely to be linked. The appearance term is the Euclidean distance between the average template color of the two tracklets. The motion compatibility term combines both tracklets and computes the likelihood of the combined trajectory using a Kalman filter. The final likelihood of associating the tracklets is the sum of the appearance and the motion log likelihoods.

The final unspecified scores are the track creation and termination scores, $C_{en,i}$ and $C_{ex,i}$ respectively. These scores control track fragmentation and are constant valued parameters. In practice, we found it useful for the track creation and termination scores to be roughly $-2C$ so that tracklets of very high merit are included but tracklets with low merit must be linked with other tracklets to be included in the final set of trajectories.

These scores are used as edge costs in a min-cost-flow problem. The solution is then transformed into image space trajectories, interpolating between associated tracklets to obtain complete tracks.

6. Experiments

We validated our approach on the NGSIM Peachtree data set [1]. This data set was captured through a collaboration of researchers interested in modeling vehicle behavior [2]. A series of cameras were set up viewing consecutive parts of Peachtree Street in Atlanta, Georgia, and 15 minutes of data were simultaneously recorded in each. Extensive ground truth is freely available within the data set, including geo-referenced lane centerlines, intersection locations and traffic light cycles. A total of more than 1,200 vehicles were tracked through the scene using a semi-automated tracking system with extensive hand corrections.

6.1. Tracking Metrics

To provide a quantitative evaluation of our system, we measure the *track completeness factor* and *track fragmentation* as defined by Perera et al. [14]. We compute these metrics as follows:

Given a set of estimated tracks, T, and ground truth tracks, G, we find an optimal-cost association, A^*, from each track in T to a corresponding track in G. The cost of an association is defined as the sum of distances between the estimated tracks and their associated ground truth tracks, where the distance between two tracks is defined as the average Euclidean distance between the estimated object po-

sition and ground truth object position over all frames. The optimal association, A^*, is the association that minimizes this cost. From the optimal association, we compute two performance metrics. The first is the *track completeness factor* (TCF), which provides a measure of how well each object is detected. The second is the *track fragmentation* (TF), which provides a measure of how well the identity of a track is maintained. These are defined as follows:

$$TCF = \frac{\sum_i \sum_{T_j \in A(G_i)} |O(T_j, G_i)|}{\sum_i |G_i|}$$

$$TF = \frac{\sum_i |A(G_i)|}{|\{G_i | A(G_i) \neq \phi\}|},$$

where $A(G_i)$ is the set of estimated tracks associated with track G_i in the optimal association, A^*; $O(T_j, G_i)$ denotes the frames where T_j and G_i overlap; and $|\cdot|$ denotes the cardinality operator.

6.2. Tracking Results

We evaluated our system on seven scenes from the NGSIM Peachtree data set. Because ground truth was not provided for any vehicles initially present in the scene, we discard the first 2,000 frames. The remaining 8,000 frames were divided into four 2,000-frame subsequences and evaluated. The average TCF and TF scores for each camera are reported in Table 1. These results compare favorably to previously reported results in similar traffic scenes [14].

The sheet-curves used for tracking were automatically generatated using the first 1,000 frames of each video. To evaluate the performance of our road delineation method, we compare our results using automatically extracted sheet curves to results obtained using a manually specified set of sheet curves. We found that hand-labeling of the sheet curves led to only a slight improvement in our tracking results.

Note that ground truth positions and identities were only provided for vehicles traveling on or through Peachtree Street. Thus, while vehicles on cross streets are tracked by our system, these tracks do not appear in the ground truth and cannot be empirically validated. Additionally, because these vehicles are not included in the ground truth, they may be erroneously associated with other ground truth tracks during evaluation, which leads to a higher track fragmentation score, even when such vehicles are tracked accurately.

Our system processed each video at approximately 20-30 FPS running in Matlab 2007b on a desktop workstation with a 2.66 GHz Intel Xeon processor. Figure 6 shows some results generated by our method on three of the NGSIM Peachtree cameras. A sequence of three frames (at 5-second intervals) is shown for each video.

	Manual		Automatic	
Camera	TCF	TF	TCF	TF
1	0.36 ± 0.17	1.36 ± 0.32	0.37 ± 0.05	1.36 ± 0.27
2	0.61 ± 0.05	1.25 ± 0.37	0.57 ± 0.05	1.22 ± 0.33
3	0.71 ± 0.12	1.38 ± 0.31	0.68 ± 0.07	1.37 ± 0.24
4	0.77 ± 0.07	1.14 ± 0.15	0.75 ± 0.03	1.14 ± 0.17
5	0.62 ± 0.07	1.35 ± 0.22	0.57 ± 0.06	1.28 ± 0.16
6	0.63 ± 0.04	1.29 ± 0.18	0.54 ± 0.07	1.34 ± 0.21
7	0.46 ± 0.07	1.40 ± 0.10	0.39 ± 0.06	1.44 ± 0.21

Table 1. The track completion factor (TCF) and track fragmentation (TF) for the NGSIM Peachtree data set.

7. Conclusion

We have demonstrated a system that can automatically identify consistent motion patterns in a scene and exploit this structure to track objects. We demonstrated our algorithm running at frame-rate on urban traffic scenes in a large, publicly available and ground-truthed data set and provided quantitative evaluation to support future comparisons.

A key advantage of our method is its ability to track objects using vastly less image data than required by typical approaches. Because imaging systems are often bandwidth-limited, decreasing the number of pixels that must be communicated can reduce restrictions on the resolution and the frame-rate. Our work has shown that, if coupled with an imaging system that can configure which pixels are sampled and transmitted, our system has the potential to enable tracking at higher frame-rates and in larger images than is currently possible.

References

[1] Summary report, ngsim peachtree street (atlanta) data analysis. http://www.ngsim.fhwa.dot.gov/, June 2007. 6, 8

[2] V. Alexiadis, J. Colyar, and J. Halkias. A model endeavor: A public-private partnership is working to improve traffic microsimulation technology. *FHWA Public Roads*, 2007. 6

[3] L. Auslander. On curvature in finsler geometry. *Trans. of the Amer. Math. Soc.*, 79:378–388, 1955. 2

[4] J. B. Beltman, A. F. Marée, J. N. Lynch, M. J. Miller, and R. J. de Boer. Lymph node topology dictates t cell migration behavior. *Journal of Experimental Medicine*, 204(4):771–780, 2007. 2

[5] R. C. Bolles, H. H. Baker, and D. H. Marimont. Epipolar-plane image analysis: An approach to determining structure from motion. *International Journal of Computer Vision*, 1(1):7–55, 1987. 2

[6] A. Criminisi, S. B. Kang, R. Swaminathan, R. Szeliski, and P. Anandan. Extracting layers and analyzing their specular properties using epipolar-plane-image analysis. *Comput. Vis. Image Underst.*, 97(1):51–85, 2005. 2

[7] N. Jacobs, M. Dixon, and R. Pless. Location-specific transition distributions for tracking. In *IEEE Workshop on Motion and Video Computing*, Copper Mountain, CO, Jan. 2008. 2

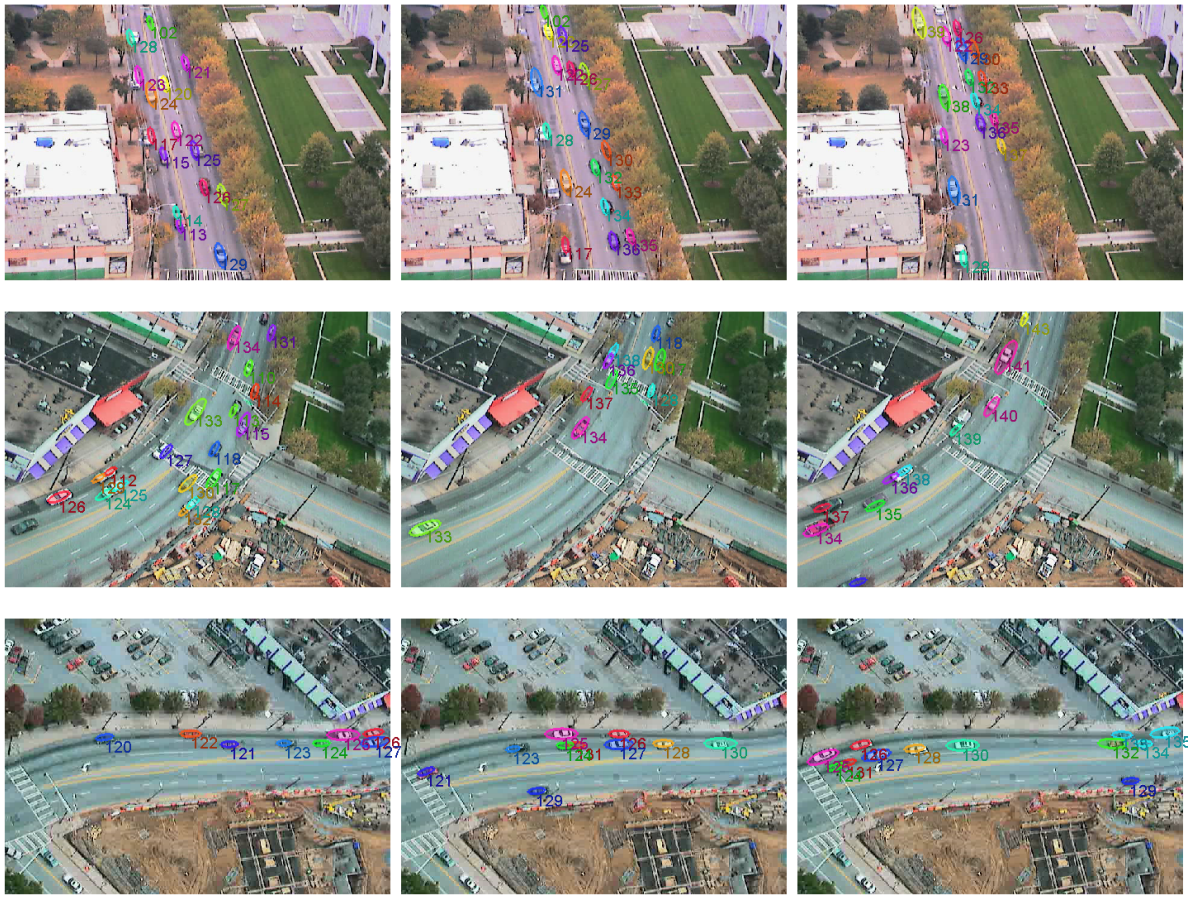


Figure 6. Tracking results obtained using sheet tracking on three videos from the NGSIM Peachtree [1] data set. Columns (from left to right) show tracking results at frames 2500, 2550, and 2600 for each video. Views from the second, third, and fourth cameras are shown.

[8] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *IJCV*, 1(4):321–331, 1988. 2

[9] V. Kellokumpu, G. Zhao, and M. Pietikäinen. Human activity recognition using a dynamic texture based method. In *Proc. of The British Machine Vision Conference (BMVC 2008)*, 2008. 2

[10] J. Melo, A. Naftel, A. Bernardino, and J. Santos-Victor. Detection and classification of highway lanes using vehicle motion trajectories. *Intelligent Transportation Systems, IEEE Transactions on*, 7(2):188–200, June 2006. 2

[11] J. Melonakos, E. Pichon, S. Angenent, and A. Tannenbaum. Finsler active contours. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(3):412–423, 2008. 2

[12] M. J. Miller, S. H. Wei, M. D. Cahalan, and I. Parker. Autonomous t cell trafficking examined in vivo with intravital two-photon microscopy. *Proc. of the National Academy of Science*, 100(5):2604–2609, 2003. 2

[13] S. Niyogi and E. Adelson. Analyzing and recognizing walking figures in xyt. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 469–474, 1994. 2

[14] A. G. A. Perera, C. Srinivas, A. Hoogs, G. Brooksby, and W. Hu. Multi-object tracking through simultaneous long occlusions and split-merge conditions. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 666–673, Washington, DC, USA, 2006. IEEE Computer Society. 2, 3, 6, 7

[15] R. Pless. Detecting roads in stabilized video with the spatio-temporal structure tensor. *Geoinformatica*, 10(1):37–53, 2006. 2, 3, 5

[16] Y. Ran, R. Chellappa, and Q. Zheng. Finding gait in space and time. *Proc. International Conference on Pattern Recognition*, 4:586–589, 2006. 2

[17] Y. Ricquebourg and P. Bouthemy. Real-time tracking of moving persons by exploiting spatio-temporal image slices. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):797–808, 2000. 2

[18] T. Schoepflin and D. Dailey. Dynamic camera calibration of roadside traffic management cameras for vehicle speed estimation. *Intelligent Transportation Systems, IEEE Transactions on*, 4(2):90–98, June 2003. 2

[19] A. Yilmaz, O. Javed, and M. Shah. Object tracking: A survey. *ACM Comput. Surv.*, 38(4):13, 2006. 2

[20] L. Zhang, Y. Li, and R. Nevatia. Global data association for multi-object tracking using network flows. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, June 2008. 2, 3, 6