

Mining Patterns of Activity from Video Data

Michael C. Burl

Department of Computer Science

University of Colorado, Boulder

Michael.Burl@colorado.edu

Abstract

In this paper, an algorithm for extracting information from raw, surveillance-style video of an outdoor scene containing a mix of people, bicycles, and motorized vehicles is presented. A feature extraction algorithm based on background estimation and subtraction followed by spatial clustering and multi-object tracking is used to process sequences of video frames into a track set. The resulting track set, which encodes the positions, velocities, and appearances of the various objects as a function of time, are mined to answer user-generated queries that are potentially relevant for surveillance applications and for input to public planning processes.

1 Introduction

Data mining has been defined as the process of extracting implicit, nontrivial, previously unknown and potentially useful information from data in databases [8]. Although much of the effort in the data mining community has indeed focused on traditional databases, there is a growing interest in algorithms that can extract useful information from nontraditional data such as video sequences. This trend is driven both by a heightened emphasis on security and by the ready availability of cheap hardware for quality video capture and analysis. Related work in this area includes [11, 2, 6, 5, 14, 12, 13, 4].

Much of the complexity in mining video data is in the low-level feature extraction steps (computer vision and image processing) that directly handle the raw data. The outputs from these steps can then be used as inputs to higher-level data mining processes. We present an algorithm for extracting information from raw, surveillance-style video of an outdoor scene containing a mix of people, bicycles, and motorized vehicles. The ECOT dataset that we use for testing consists of approximately 18,000 grayscale frames of size (480×640) . The feature-level *track set* data extracted from the image sequence consists of state trajectories (positions, velocities) and appearances versus time for the various objects that appear in the scene.

Higher-level information one may hope to obtain from video includes: detecting trigger events (e.g., any vehicles entering a particular area, people exiting or entering a particular building), determining typical

and anomalous patterns of activity, generating person-centric or object-centric views of an activity, classifying activities into named categories (e.g., walking, riding a bicycle), clustering, and determining interactions between entities.

2 Data Characteristics

The ECOT video data was collected on the University of Colorado, Boulder campus during the summer of 2003 from a vantage point ~ 25 m above the ground plane. Grayscale images (8 bits-per-pixel) of size (480×640) were captured and stored to the disk of a laptop computer with no compression at a rate of approximately 15Hz. The camera was stationary with fixed internal parameters during the data collection. Approximately 20 minutes worth of data, equivalent to 5.5GB or approximately 18,000 frames, were recorded. The field of view of the camera is approximately 50 degrees the horizontal direction so an individual pixel is approximately 1/10th of a degree wide. The camera was tilted downward from the horizon to acquire images of the desired area. Due to the resulting imaging geometry, the spatial resolution is higher toward the bottom of the frames (near-range) and lower toward the top of the frames (far-range). Also, because the line-of-sight becomes closer to horizontal toward the top of the frames there is more potential for occlusion in that region.

3 Feature Extraction via Multi-object Tracking

As shown in Figure 1, our feature extraction from video algorithm consists of the following steps:

1. Background estimation and subtraction.
2. Thresholding and spatial grouping.
3. Gating and Data Association.
4. Birth of New Tracks.
5. Measurement Update.
6. Time Update.

Steps 1 and 2 focus the feature extraction on regions of a frame that differ significantly from what is expected

based on a model of the background. Steps 3–6 implement a multi-object tracker that tries to follow interesting regions across frames. These steps are applied to each frame; the end result is a track set (position, velocity, and appearance vs time for each object).

3.1 Background Estimation and Subtraction

Since the camera is stationary and the background, for the most part, is also stationary, the algorithm attempts to directly model the background appearance and then detect pixels that deviate from the expected background appearance. Various statistical approaches to background estimation have been developed and used before; see [4] for a recent survey and comparison. We have used a temporal median approach in which the background model B_t for a pixel at a particular time t and spatial location (i, j) is simply the median value over the pixel values at this spatial location calculated over a finite historical buffer lagged by τ from the current frame and sampled much slower than the frame rate. Regardless of the quality of the background estimate, the subtraction process may cause fragmentation of an object if parts of the object pass over regions of the background with similar gray-levels.

3.2 Thresholding and Grouping

Given the current frame I_t and the background estimate B_t , the algorithm next identifies pixels significantly different from the background. If $|I_t(i, j) - B_t(i, j)| \geq \Delta$, then pixel (i, j) is flagged as "potentially interesting". Hysteresis thresholding similar to the Canny edge detector [3] is used so that a "potentially interesting" pixel must be part of a group containing a "really interesting" pixel, i.e., a pixel that passes a more rigorous background subtraction threshold. By using hysteresis thresholding, the algorithm can in some cases beneficially "region grow" or bridge across areas with weak contrast against the background.

A spatial clustering algorithm based on classical connected components [9] and single linkage clustering [7] is used to combine pixels into groups. Instead of requiring that a pixel be strictly connected to another, pixels are linked if they are within some pre-defined distance threshold from each other. (Linking is applied transitively.) The result of clustering is a label image in which each pixel in a given cluster receives an integer label (cluster ID). These labels can then be used to compute centroids and other moments for each cluster.

3.3 Linear Dynamical Model

The first two steps of the feature extraction algorithm identify potentially interesting clumps of pixels and group these together

into "objects". The next steps involve linking the objects from different frames together into tracks. For these steps, we have used the machinery of the Kalman filter [10, 1]. Underlying the Kalman filter is a linear dynamical model¹ of the following form (following the notation of [1]):

$$(3.1) \quad \mathbf{x}_{k+1} = \mathbf{F}_k \mathbf{x}_k + \mathbf{G}_k \mathbf{w}_k$$

$$(3.2) \quad \mathbf{z}_k = \mathbf{H}_k^T \mathbf{x}_k + \mathbf{v}_k$$

where \mathbf{x}_k is an $(xr \times 1)$ vector describing the *state* of a particular object at discrete time-step k , \mathbf{w}_k is a $(wr \times 1)$ noise process (assumed Gaussian with zero-mean and covariance \mathbf{Q}_k) that drives the object dynamics, and \mathbf{z}_k is the $(zr \times 1)$ *measurement* (or *observation*) made at time-step k . The number of state variables is x , the number of noise variables is wr , and the number of observatoin variables is zr . Note that \mathbf{z}_k is a function of the system state and the noise process \mathbf{v}_k (assumed to be Gaussian with zero mean and covariance \mathbf{R}_k). The statistics of the initial state \mathbf{x}_0 are assumed to be Gaussian with mean $\bar{\mathbf{x}}_0$ and covariance \mathbf{P}_0 .

The dynamical equations above satisfy the discrete Gauss-Markov property so that the state at time step $k + 1$ is Gaussian distributed and the density $p(\mathbf{x}_{k+1} | \mathbf{x}_k, \dots, \mathbf{x}_0) = p(\mathbf{x}_{k+1} | \mathbf{x}_k)$, i.e., knowing the most recent state \mathbf{x}_k tells us as much about \mathbf{x}_{k+1} as knowing the full state history. Since the various quantities are Gaussian distributed, we only need to keep track of means and covariances to know the full distributions.

Given the sequence of observations $\mathcal{Z}_k = \mathbf{z}_1, \mathbf{z}_2, \dots, \mathbf{z}_k$, we will define two estimators: one estimates the current state and the other predicts the next state (one-step prediction).

$$(3.3) \quad \hat{\mathbf{x}}_{k|k} = E[\mathbf{x}_k | \mathcal{Z}_k]$$

$$(3.4) \quad \hat{\mathbf{x}}_{k+1|k} = E[\mathbf{x}_{k+1} | \mathcal{Z}_k]$$

The notation $\hat{\mathbf{x}}_{k|k}$ means the estimate of the state at time step k given observations up to and including time step k . Similarly, the notation $\hat{\mathbf{x}}_{k+1|k}$ means the estimate of the state at time step $k + 1$ given measurements up to and including time step k . In addition to the state estimators, we also keep track of the covariance of the estimates with $\hat{\Sigma}_{k|k}$ and $\hat{\Sigma}_{k+1|k}$. From the one-step state predictor, we can determine the expected value for the observation and its covariance. Specifically,

$$(3.5) \quad \hat{\mathbf{z}}_{k+1|k} = \mathbf{H}_k^T \hat{\mathbf{x}}_{k+1|k}$$

$$(3.6) \quad \hat{\Sigma}_{k+1|k}^{(z)} = \mathbf{H}_k^T \hat{\Sigma}_{k+1|k}^{(x)} \mathbf{H}_k^T + \mathbf{R}_k$$

¹The material in this section is fairly standard, but is included to keep the paper self-contained.

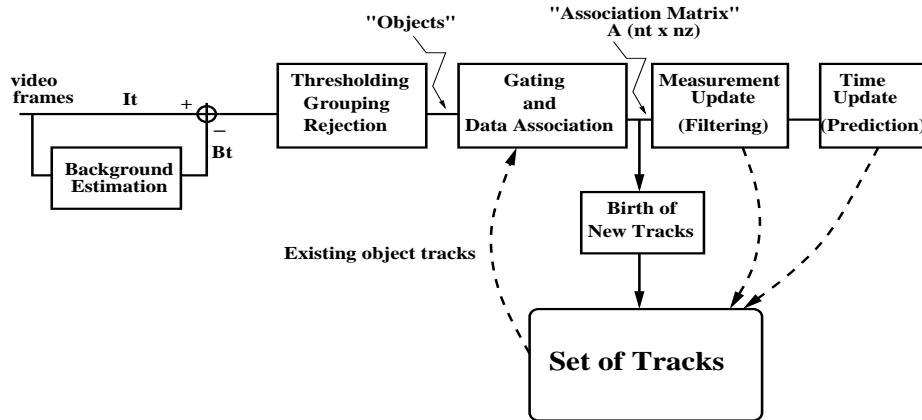


Figure 1: The feature extraction algorithm consists of a background estimation and subtraction step based on temporal median filtering, thresholding and spatial grouping, followed by a multi-object tracker.

These quantities will be useful for deciding which objects in a new frame should be associated with which tracks from the previous frames.

3.4 Gating and Data Association Given that we are tracking an object and have observations up to and including frame k , Equations 3.5 and 3.6 provide a means to predict where the next observation from this track should occur and how much uncertainty from the predicted position we might expect. By setting a bound on the squared Mahalanobis distance from $\hat{\mathbf{z}}_{k+1|k}$, we can establish an elliptical *gate* into which the next measurement for a particular track should fall. Since we will be tracking multiple objects, we expect there to be multiple detected objects (observations) and more than one object may fall into the gate of a particular track. If we have n_t currently active tracks and n_z separate observations (“objects”) in the current frame then the process of gating amounts to determining a $(n_t \times n_z)$ matrix \mathbf{A} in which entry $\mathbf{A}_{i,j} = 0$ if measurement j does not fall inside gate i , but is non-zero if measurement j does fall in gate i . In more recent work, we have examined gating based on both trajectory and appearance, which seems to be more robust.

3.5 Track Births Any measurements which do not fall inside any of the track gates are assumed to be new objects which must be assigned new tracks. A new track is initiated by setting the position components within the state estimator to be the centroid of the new object. The velocity components are set to zero, since we do not know the direction of motion (isotropic model). The initial covariance is taken from \mathbf{P}_0 . Note

that the components of \mathbf{P}_0 that describe the variances of the initial velocity components should be sufficiently large to allow the gate in the next frame to capture the object again. The estimator will then begin to lock on to the true velocity of the object and follow it through subsequent frames.

3.6 Measurement Update Once a specific measurement is associated to a specific track, we can perform the *measurement update* step of the Kalman filter. This step amounts to computing an estimate of the current state using all of the observations up to and including the current time. The equations can be conveniently written in terms of the *innovation*, $\boldsymbol{\nu}_k$, which reflects the difference between the actual measurement and the predicted measurement.

$$(3.7) \quad \hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{L}_k \boldsymbol{\nu}_k$$

$$(3.8) \quad \hat{\boldsymbol{\Sigma}}_{k|k} = \hat{\boldsymbol{\Sigma}}_{k|k-1} - \mathbf{L}_k \mathbf{H}_k^T \hat{\boldsymbol{\Sigma}}_{k|k-1}$$

$$(3.9) \quad \boldsymbol{\nu}_k = \mathbf{z}_k - \hat{\mathbf{z}}_{k|k-1}$$

$$(3.10) \quad \mathbf{L}_k = \hat{\boldsymbol{\Sigma}}_{(\mathbf{x})k|k-1} \mathbf{H}_k \hat{\boldsymbol{\Sigma}}_{(\mathbf{z})k|k-1}^{-1}$$

The matrix \mathbf{L}_k is known as the innovations gain matrix.

The Kalman filter is also able to track through situations in which a measurement is not acquired during a particular frame. In this situation, the state estimate and its covariance are just based on the predictions. Thus, if no measurement is acquired, the tracker uses:

$$(3.11) \quad \hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1}$$

$$(3.12) \quad \hat{\boldsymbol{\Sigma}}_{k|k} = \hat{\boldsymbol{\Sigma}}_{k|k-1}$$

When this happens, the uncertainty in the state estimate grows with each frame, which can eventually lead

to problems with the gating process. In practice, if a measurement is not acquired for a limited number of frames the track is deactivated. If the object is later redetected, it will be assigned a new track.

3.7 Time Update The time update step of the Kalman filter involves predicting the state in the next frame and the covariance of that prediction. Basically, this step reduces to propagating the current estimates through the linear dynamical model.

$$(3.13) \quad \hat{\mathbf{x}}_{k+1|k} = \mathbf{F}_k \hat{\mathbf{x}}_{k|k}$$

$$(3.14) \quad \hat{\Sigma}_{k+1|k} = \mathbf{F}_k \hat{\Sigma}_{k|k} \mathbf{F}_k^T + \mathbf{G}_k \mathbf{Q}_k \mathbf{G}_k^T$$

Similarly for the measurements, we can predict ahead one step to find (same as Equations 3.5 and 3.6):

$$(3.15) \quad \hat{\mathbf{z}}_{k+1|k} = \mathbf{H}_k^T \hat{\mathbf{x}}_{k+1|k}$$

$$(3.16) \quad \hat{\Sigma}_{k+1|k}^{(z)} = \mathbf{H}_k^T \hat{\Sigma}_{k+1|k}^{(x)} \mathbf{H}_k^T + \mathbf{R}_k$$

If the predictions for a given track are outside of the visible image (taking into account the covariance and gate size), then the track is deactivated.

4 Queries over Track Sets

The result of the feature extraction algorithm is a track set. Each track records the history of an object's state estimates (positions and velocities) and appearance over the frames in which the track was active. We next consider some of the types of queries over track sets that are potentially relevant to surveillance applications or as input for planning purposes. As we will see, it is quite easy to establish "virtual fences" around designated areas of the scene. In the ECOT sequence, we might be interested in trajectories that enter the area near the bike racks. Trajectories entering this area can be flagged and the system operator can then request object-centric movies of any persons that entered the area. Similarly, if the system automatically flags unusual trajectories, the operator can view the object-centric movie to see what was actually happening.

4.1 Spatial-temporal queries Using the track set, it is quite easy to establish "virtual fences" around designated areas of the scene, e.g., near the bike racks, so that anyone entering or leaving an area is flagged. Such spatio-temporal queries can be augmented with additional conditions, e.g., if there were an interest in persons exiting a particular building, but not in persons entering the building, the spatial query (or queries if there were multiple exits) can be augmented with additional conditions on the velocity vectors culled from the state estimates. One could also easily incorporate temporal constraints, e.g., if it were known that an event of

interest (say a computer theft) happened in a particular time frame, the set of tracks over which the query would need to be posed could be restricted accordingly. Specialized spatial-temporal indexing structures can be created over the track set to enable these types of queries to be answered efficiently, but we have not taken this step in the current study.

4.2 Source-Destination or Waypoint Queries A slightly more complex query, which can be viewed as an intersection between the results of two spatial-temporal queries, asks for any trajectories that involve a person leaving Building A and going to Building B.

4.3 Interactions between Entities Another type of query involves determining when there were interactions between entities. One might be interested in all encounters between runners and (unleashed) dogs on a particular trail. This query amounts to determining the spatial locations and times when an entity of type A was within a given proximity of an entity of type B. Due to the common timestamping applied to all the trajectories, this query can be answered in brute force fashion by computing the minimum distance between each of the type A tracks and each of the type B tracks. All pairs for which the minimum distance is sufficiently small can be flagged for further evaluation.

5 Experimental Results and Discussion

In the current set of experiments, a time-invariant constant-velocity model in the image plane was used to represent object dynamics. Under this model, the state vector is (4×1) with the x and y position and the v_x and v_y velocities. The observation vector is 2-dimensional and consists only of positional measurements. The dynamical model says that the new position variables are the old position variables plus the velocities (the sample spacing dt can be incorporated into the velocity units) and the new velocities are the old velocities plus white noise.

Figure 2 shows the set of trajectories recovered from a subsequence (724 frames) of the full ECOT dataset. The algorithm, with no effort made toward optimization, runs at ~ 8 Hz on a 2.4GHz Pentium 4 desktop machine. There is no ground truth marking for this dataset (a deficiency noted by several of the reviewers), so it is difficult to give a quantitative assessment of the performance. Some qualitative remarks can be made, however. First, the short trajectories in the tree that appears in the upper left corner of the image are clearly caused by wind effects. In part, this is due to the unimodal background model implicit in the temporal median filter. Other short spurious trajectories can be

seen in the upper right. The results in the bottom three-fourths of the image, however, where the resolution is better due to the viewing geometry, are generally quite good.

The trajectory that originates near the point (0, 425) shows an interesting phenomena that we have seen on other subsets of the sequence. Initially a track is assigned to the person's head. Around (50, 420), the tracker began to also separately track the person's shoes. The clothing blends in too well with the background and hence is not detected, so the two parts of the person are not linked together. As the person nears (80, 360), the head is not detected for several frames and the track assigned to the head jumps down to the shoes. Issues such as this make it difficult to respond to queries about the number of objects. In addition to oversegmentation, we have also seen instances of undersegmentation (or, equivalently, overgrouping) in which pairs of people walking in close proximity are assigned to a single track.

From the track set it is possible (and easy) to generate object-centric views. Figure 3 shows an object-centric view of the person in track #303. This object happens to be the person who is at coordinates (218, 317) in previous figure. The mosaic (presented in row-major order) shows precisely what this person was doing over the previous 20 frames.

6 Conclusions and Future Work

The data association step of the tracker clearly needs to be improved. Currently, each track takes the observation that is closest in Mahalanobis distance to its prediction as its measurement. This method is not robust enough as it does not force the appearance of the object to match the appearance held by the track, which leads to frequent "identity theft" when two objects pass close by each other. One object steals the other's track, but when the two objects eventually separate, the object that lost its track is treated as a "new" object.

References

- [1] B.D.O. Anderson and J.B. Moore, *Optimal Filtering*, T. Kailath (ed), Prentice-Hall. (1979)
- [2] Brickstream, <http://www.brickstream.com/>
- [3] J. Canny, "A Computational Approach to Edge Detection", *TPAMI*, 8(6), 679-698, (1986).
- [4] S.-C. Cheung, C. Kamath, "Robust Techniques for Background Subtraction in Urban Traffic Video", In *Video Comm and Image Proc.*, SPIE Electronic Imaging, San Jose (2004).
- [5] R.T. Collins, A.J. Lipton, H. Fujiyoshi, T. Kanade, "Algorithms for Cooperative Multisensor Surveillance", *Proc. of the IEEE*, Vol. 89, No. 10, (Oct 2001).

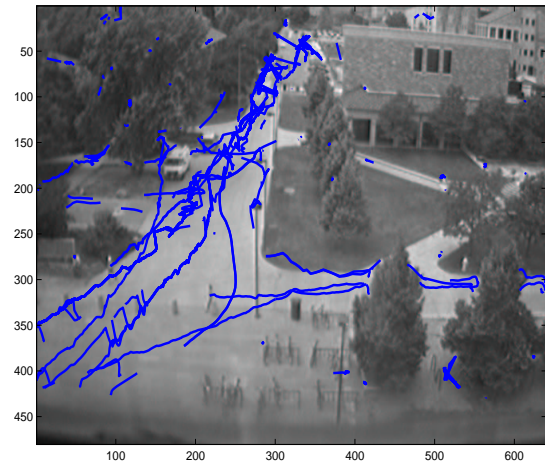


Figure 2: Overlay onto frame 9800 of the trajectories recovered from frames 9075–9800 of the ECOT sequence.

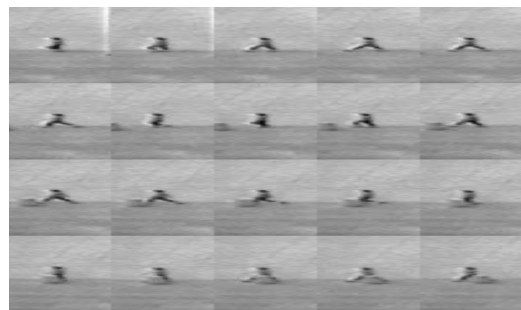


Figure 3: The last 20 frames (row-major order) of an object-centric movie focused on track #303.

- [6] J.-P. Deparis, Y. David, S. Velastin, M. Wherrett, R. Lioni, D. Aubert, D. Sorrenti, "CROMATICA Final Report: Project TR1016, CROwd MAnagement with Telematics Imaging and Communication Assistance", (1999)
- [7] R.O. Duda, P.E. Hart, D.G. Stork, *Pattern Classification*, John Wiley and Sons, (2001).
- [8] U.M. Fayyad, G. Piatetsky-Shapiro, P. Smythm and R. Uthurusamy, *Advances in Knowledge Discovery and Data Mining*, MIT Press, Cambridge, MA (1996).
- [9] B.K.P. Horn, *Robot Vision*, MIT Press, (1986).
- [10] R.E. Kalman, "A New Approach to Linear Filtering and Prediction Problems", *J. Basic Eng.*, ASME, Series D, Vol 82, No. 1, pp. 33–45, (1960).
- [11] C. Rasmussen and G.D. Hager, "Probabilistic Data Association Methods for Tracking Complex Visual Objects", *IEEE Trans. on PAMI*, Vol. 23, No. 6, pp. 560–576, (Jun 2001)
- [12] A. Rosenfeld, D. Doermann, D. DeMenthon, Eds., *Video Mining*, Kluwer (2003).
- [13] C. Stauffer, W.E.L. Grimson, "Adaptive Background Mixture Models for Real-time Tracking", *CVPR*, pp. 246-252, (1999).
- [14] T. Zhao, R. Nevatia, "Bayesian Human Segmentation in Crowded Situations", *CVPR*, pp. 459-466, (2003).