Vehicle Detection & Tracking

Gagan Bansal
Johns Hopkins University
gagan@cis.jhu.edu

Sandeep Mullur
Johns Hopkins University
sandeepmullur@gmail.com

Abstract

Object tracking is a field of intense research due to its varied applications leading to several algorithms based on environment. Our proposed algorithm is suitable for the given problem wherein several moving objects exist in a scene obtained with little camera motion. Objects are detected based on motion using adaptive background subtraction. Morphological techniques are used to obtain proper segmentation. Tracking is then done using the kalman-filter approach. A discussion on our approach to solve the problems of merge and split and large camera jerk is also given.

1. Introduction

Object tracking is the problem of estimating the positions and other relevant information like trajectory, shape, size and number of moving objects in an image sequence. So, a tracker assigns consistent labels to the tracked objects in different frames of a video.It has several important applications such as security and surveillance, annotation of videos, traffic management, motion-based video compression and interactive games.

Object tracking, in general, is a challenging problem. Difficulties in tracking objects can arise due to abrupt object motion, changing appearance patterns of the object and the scene, nonrigid object structures, object-to-object occlusions, and camera motion. The complex task of tracking can be simplified by imposing constraints on the motion and/or appearance of objects. For the given video sequence, we assume that the object motion is smooth with no abrupt changes. The object motion is further constrained to be of constant velocity. Prior knowledge about the number and the size of objects, object appearance and shape,is also made use of to simplify the problem. It is also assumed that the camera doesn't move significantly except for minor jerks.

Object tracking involves four main steps- detection, segmentation, tracking and recognition. Object detection and segmentation can be done by several ways:

Color-based– An object can be segmented based on its color in the RGB or HSV space. This is ideal if the object color is distinct from the background. An object can also be recognised or classified based on its color. In the given video sequence, object colors are not distinct as they take a large range of colors. Object recognition based on color would work for some objects like rickshaws having a distinct yellow color, but would fail for cars that can take several colors.

Edge-based– Object boundaries usually generate strong changes in image intensities. Edge detection is used to identify these changes. An important property of edges is that they are less sensitive to illumination changes compared to color features. Algorithms that track the boundary of the objects usually use edges as the representative feature. The method is not suitable when there are several cluttered objects in a frame.

Texture-based–Texture is a measure of the intensity variation of a surface which quantifies properties such as smoothness and regularity. Compared to color, texture requires a processing step to generate texture descriptors.Similar to edge features, the texture features are less sensitive to illumination changes compared to color.

Point detectors–Point detectors are used to find interest points in images which have an expressive texture in their respective localities. A desirable quality of an interest point is its invariance to changes in illumination and camera viewpoint. Commonly used interest point detectors include Harris interest point detector,KLT detector and SIFT detectors. The method is very sensitive to noise and computationally expensive.

Background subtraction–Object detection can be achieved by building a representation of the scene called the background model and then finding deviations from the model for each incoming frame.Any significant change in an image region from the background model signifies a moving object. The pixels constituting the regions undergoing change are marked for further processing like connected component labeling. The method is ideal for segmenting moving objects when the camera motion is negligible.

Once objects have been detected in a frame, tracking is to

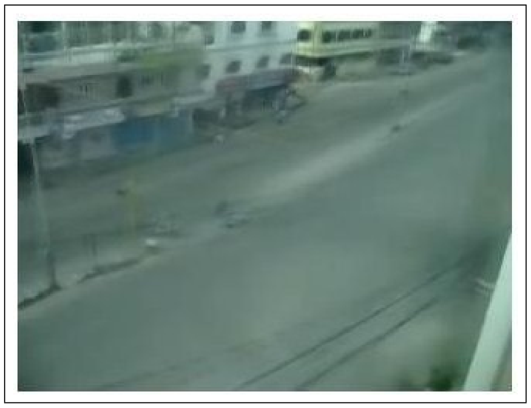


Figure 1. Initial Background Image

be done to achieve correpondence with objects detected in previous frames. Several methods exist for tracking based on context:
Point tracking– Deterministic and Probabilistic methods. Kalman filter is a commonly used statistical method for solving correspondence problem.
Kernel tracking– Multi-view and Template based
Silhouette tracking– Contour evolution and Shape matching

2. Problem Description

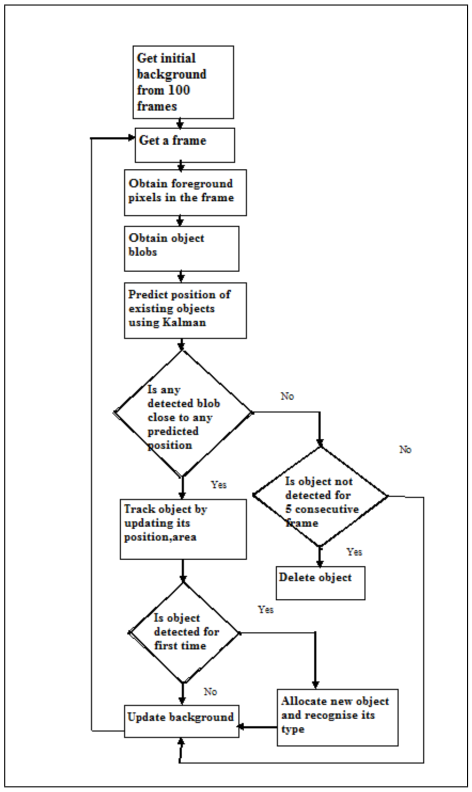
We are given a traffic video shot in India. The aim is to detect,track and recognise vehicles and pedestrians from the video. The problem becomes non trivial given the large volume of traffic it captures. Large number of objects leads to some objects fully visible and many objects partly visible due to occlusions from other objects. The problem of detection and tracking partially visible or hidden objects becomes really challenging. The camera is not fixed and there is lot of camera shake which further increases the complexity of the problem. The poor resolution of the video also compounds the detection problem.

3. Proposed Solution

We use background subtraction for segmentation and detection of objects and use Kalman Filter for tracking the objects.
We note that this approach has been widely used before in applications such as surveillance and automatic video monitoring. ([1]) use the same approach for detecting,tracking and counting people from an overhead camera. Figure (2) summarizes the algorithm in a flowchart.

3.1. Background Subtraction

Background subtraction is an approach for detecting changes in the background of the scene. It assumes an



Get initial background from 100 frames
Get a frame
Obtain foreground pixels in the frame
Obtain object blobs
Predict position of existing objects using Kalman
Is any detected blob close to any predicted position
No
Yes
Track object by updating its position,area
Is object not detected for 5 consecutive frame
No
Yes
Delete object
Is object detected for first time
Yes
No
Allocate new object and recognise its type
Update background

Figure 2. Flowchart of the algorithm

initially static scene where there is no motion and builds mean and standard deviation images for the scene during this phase.Since the input video to our system has no such initial frames with no motion, we initialise the mean image by taking the median of the first 100 frames and the standard deviation image as the difference be the frame and the initial median image averaged over the first 100 frames.Figure (1) shows the initial background image.

$$|I(x,y) - MeanImage(x,y)| > c \times StdDevImage(x,y) \quad (1)$$

Once we have our initial background images, we label a pixel as being in foreground if for any channel Eq. (1) holds true. We chose the constant $c = 2.1$ in Eq. (1).Hence we obtain a mask for the foreground pixels in the scene.
Another issue that needs to be addressed here is that the input video in our case has some camera shake and the camera moves considerably in one or two frames. The sudden camera movement results in some background objects also coming into the foreground. Our approach to solving this problem was trying to absorb static objects into the background. Hence we maintained a counter for each pixel as to if some pixel was continuously in the foreground for a threshhold number of frames, we removed it from the foreground mask. We chose this threshhold to be 15 frames. Once we removed the pixel from the foreground mask, we

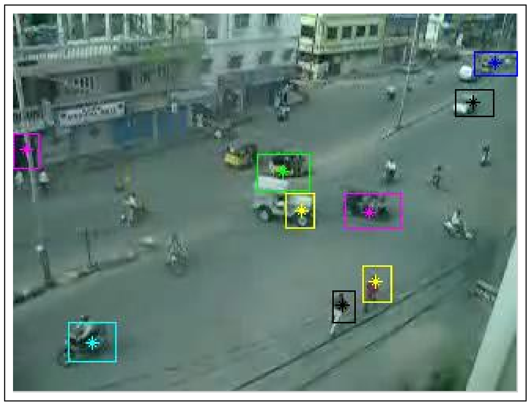


Figure 3. Detected Objects in a frame

updated our background mean and standard deviation images. This was done as a running average between the previous mean image and the background in the current frame. The background standard deviation image was updated similarly as a running average between previous standard deviation image and the difference between current frame and mean background image.

3.2. Detection of Objects

Background subtraction gives us candidate foreground pixels. We further need to cluster the foreground pixels into candidate objects.We initially use morphological operations of image opening and closing. Image closing operation fills the holes,if any, in some cluster of foreground pixels. Image opening operation removes potential noise foreground pixels that connect two clusters. After these operations we remove the foreground clusters having area less than 200 pixels and use connected component labelling which gives us potential objects in the scene. We maintain areas, bounding boxes and centroids of each such object in the foreground.Figure (3) shows the detected objects in some frame.

3.3. Tracking of objects-Kalman filter

Once we have detected some objects, we need to find which of these objects correspond to previously detected objects and those which are detected for the first time.The tracking is done by fitting a discrete time linear dynamical system(LDS) to each object.Each object has an associated state which is a vector $[x, y, v_x, v_y]^T$, where x, y is taken as the center of the bounding box of the object and v_x, v_y are the velocities along x and y directions. We assume that the vehicles move with constant velocities and hence do not take into account the accelerations of the objects. This assumption is valid considering the fact that the velocities of objects in the video do not change considerably between two consecutive frames. The output of the LDS is the actual centers of the bounding boxes of the detected objects given by $[x, y]^T$. In giving the equations for the Kalman filter, we use a similar notation as used by ([2]).

$$state[k] = Astate[k-1] + w[k-1] \tag{2}$$

$$output[k] = Hstate[k] + v[k] \tag{3}$$

$$x[k] = x[k-1] + v_x \times TimeStep \tag{4}$$

$$A = \begin{bmatrix} 1 & 0 & T & 0 \\ 0 & 1 & 0 & T \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5}$$

$$H = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \tag{6}$$

Eq. (2) relates the state of the system at step k in terms of the state of the sytem at step k-1. Eq. (3) relates the output of the system at step k in terms of the state of the sytem at step k.
$w[k-1]$ and $v[k]$ are the process and measurement noise respectively which are taken to be gaussian with 0 mean and covariance matrices Q and R respectively.Since our LDS follows equations of motion as per the Newtonian mechanics, we can write Eq. (4). Hence we can write the A and H matrices in Eq. (2) and Eq. (3) as per Eq. (5) and Eq. (6).

$$\widehat{State[k]} = AState[k-1] \tag{7}$$

Hence given the state of the sytem at step k-1, we predict the state at step k for all previously existing objects in the system using Eq. (7). Then we search in a window of size 30×30 as to if some object is detected in current frame. If some object is detected, we correspond that detected object to the previously existing object. We allocate a new object for all detected objects that do not get a correspondance to a previously existing object. For all the previously existing objects that do not get detected in current frame we update their "NotDetected" counters and update their present state as being that predicted by the Kalman.If some object is not detected for a continuous 5 frames, we delete that object from existing object list and append its complete information including its complete trajectory into "allobjects" list which is list of all objects that existed and were later deleted.

$$State[k] = \widehat{State[k]} + GainMat[k] \times (Output[k] - H\widehat{State[k]}) \tag{8}$$

$$GainMat[k] = \frac{CovApriori[k] \times H^T}{HCovApriori[k]H^T + R} \tag{9}$$

$$CovApriori[k] = ACovariance[k]A^T + Q \qquad (10)$$

$$Covariance[k] = (I - GainMat[k]H) \times CovApriori[k] \qquad (11)$$

For all previously existing objects that are also detected in current frame, we update their state using Eq. (8) and then modify the location of center of bounding boxes as that being of actually detected corresponding object.The Gain-Matrix in Eq. (8) is defined by Eq. (9).

3.4. Occlusions - Attempt to handle merge and splits

In a video sequence with clutter, when two objects merge together, the tracking algorithm fails to associate the previous known single objects while a new bigger blob is detected. Similarly, the tracking algorithm may fail when a larger blob splits into smaller objects.
A merge is detected in our method when the Kalman estimates for the centroids of the previous objects lie within a threshold distance of the centroid of the new bigger object. In our experiment, the threshold was chosen to be 15 pixels. If a merge is detected, the new blob is associated with a new ”parent object” and the information of the previous ”child objects” is linked to it. A split is detected when the Kalman estimate of the centroid of a ”parent object” lies within a threshold distance of the centroid of two or more smaller objects. Once a split is detected, the new objects should be associated with the ”child objects” that were linked to the ”parent object”. This correspondence between objects before merge and after split is done based on area metric.
We could get the method working only for the case when two objects merge and then split into two objects. But, we could not include this into the final tracking algorithm as few other cases need to be handled, *e.g.* a new object merging into a ”parent object”. The counting process would improve significantly once this is included.

3.5. Recognition of objects

Our system recognises three kinds on objects Car,Pedestrian and Bike. We used the aspect ratios of the bounding boxes of the objects to recognise them.Since most of the motion is along one axis, we can use such a metric.We observed that cars are wider than either pedestrians or bikes. So we used if for some object, $Height < 1.2 \times Width$ it is a car, if $Height > 1.5 \times Width$ it is a pedestrian and if $1.2 \times Width < Height < 1.5 \times Width$ it is a bike.One advantage of using such a metric is that it is invariant of scale.

3.6. Counting of objects

Each time we detect an object and that detected object does not correspond to a previously existing object, we recognise that object as being either car,pedestrian or bike and increment the counter for the respective type.

Cars			Bikes			Pedestrians			Objects		
A	B	C	D	E	F	G	H	I	J	K	L
5	0	6	1	1	3	1	2	2	10	13	7
5	0	6	1	1	3	1	2	2	10	13	7
5	0	6	1	1	3	1	1	2	9	13	7
5	0	6	1	1	3	1	1	2	9	13	7
5	0	6	1	1	3	1	1	2	9	13	7
5	0	6	1	1	3	1	1	2	9	13	7
5	0	6	1	1	3	1	1	2	9	13	7
5	0	6	1	1	3	1	1	2	9	13	7
5	0	6	1	1	3	1	1	2	9	13	7
5	0	6	1	1	3	1	1	2	9	13	7

Table 1. Results for Frames 151-160

Cars			Bikes			Pedestrians			Objects		
A	B	C	D	E	F	G	H	I	J	K	L
5	6	10	0	2	5	0	1	1	14	15	5
5	5	10	0	3	4	0	1	1	14	18	5
5	5	10	0	3	4	0	0	1	14	18	5
5	5	10	0	3	4	0	0	1	14	18	5
4	6	10	0	2	3	0	0	1	12	18	4
5	7	10	0	2	4	0	0	1	14	17	5
5	6	10	0	3	3	0	0	1	14	18	5
3	5	10	0	2	4	0	0	1	8	15	3
3	5	10	0	2	4	0	0	1	8	15	3
3	5	10	0	2	4	0	0	1	8	15	3

Table 2. Results for Frames 552-561

4. Results

Figure (4) shows the trajectories of detected objects in two frames.

Our algorithm can process frames at around 4-5 fps if we stop the display of objects and their trajectories.Thus this approach can be useful if some application requires near real time processing.We present the detection,tracking and recognition results of our system taken on a window of 10 frames in Tables (1) and (2). Table (1) shows the results for Frames 151-160 when the objects exist at reasonably sparse locations.Table (2) shows the results for Frames 552-561 when the camera has had a sudden jerk and there are considerable occlusions among the objects.The notations used in both the Tables (1), (2) are as follows:
Column A:Number of objects recognised as cars and actually cars.
Column B:Number of objects recognised as cars and actually other objects.
Column C:Actual number of cars in the scene.
Column D:Number of objects recognised as bikes and

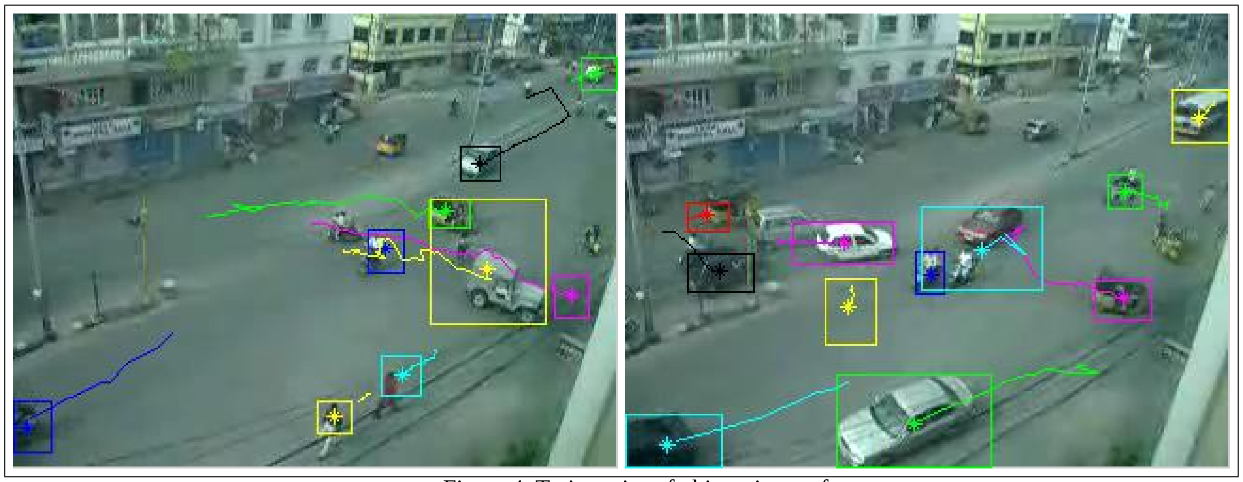


Figure 4. Trajectories of objects in two frames

actually bikes.
Column E:Number of objects recognised as bikes and actually other objects.
Column F:Actual number of bikes in the scene.
Column G:Number of objects recognised as pedestrians and actually pedestrians.
Column H:Number of objects recognised as pedestrians and actually other objects.
Column I:Actual number of pedestrians in the scene.
Column J:Total number of objects detected in the scene.
Column K:Actual number of objects detected in the scene.
Column L:Number of objects correctly recognised.

Thus we see that for frames 151-160, we correctly detected and tracked on average 9 out of a total 13 objects. and we correctly detected,tracked and recognised on average 7 out of a total 13 objects.So we recognised 7 out of 9 objects correctly.On an average there was 1 false positive and 4 false negatives.
For frames 552-561, we correctly detected and tracked on average 13 out of a total 18 objects. and we correctly detected,tracked and recognised on average 4 out of a total 13 objects.So we recognised 4 out of 13 objects correctly.This can be attributed to some background objects being detected into foreground due to the sudden camera shake. Also some of the objects are completely occluded in some of the frames.On an average there were 6 false positives and 10 false negatives.

We count the total cars in the video to be 728, bikes to be 128 and pedestrians to be 123.
The number of recognised objects are greater than the actual number of objects because one object may be detected more than once when we lose track of that object due to occlusion.

5. Caveats and Future Work

Our recognition framework is based only on the initial appearance of the object, but when an object is entering the scene its bounding box may not depict its true shape. This can be resolved by recognising some object when it has been detected for some threshhold number of frames.Some other metrics such as area or shape priors can be used to recognise the objects though at the cost of computation time.
Camera shake though handled by absorbing static objects into background can be dealt better with aligning the images assuming affine motion of the camera.The problem with our approach is that we do detect some objects as false positives for few frames until they are absorbed into the background. Also if some car is stationary at some positions for considerable time, we would absorb it into the background.
As we mentioned before, we attempted to handle merges and splits only if two objects merged and then split.This did not work well and instead spoiled our results. But merges and splits needs to handled efficiently because it can immensely improve the counting results, especially in cluttered videos. Merges can be handled in a way similar to that mentioned in Section (3.4) but for more than two objects. Splits need to be handled more carefully so as to get the right correspondance between the objects before merge and after split.

References

[1] M. C. Snidaro, Lauro and L. J. Martin. Video security for ambient intelligence. In *IEEE Transactions on Sytems, Man and Cybernetics*, 2005. 2

[2] G. Welch and B. Gary. An introduction to kalman filter. In *SIGGRAPH 2001 course*, 2001. 3