Final Report – June 20, 2010

Grant Title: Advanced In-Car Video System

Grant Reference Number: 2007-DE-BX-K007

Executive Summary

The objective of this project was to develop a prototype system with higher image quality and machine-based video analysis in order to detect undesirable (critical) events during a routine stop. A prototype system was designed and tested. The findings and the design were published (or pending to be published) in peer-reviewed scientific venues. Of the specific goals listed in the original project proposal the following were accomplished:

1. Build a computer system to analyze video and identify certain undesirable events. The system was developed as a software prototype on a laptop computer with a digital video recording device connected to the laptop that could be mounted in the passenger front seat of a car for data collection and testing purposes.
2. Deploy the test system in police cars and in simulations in order to capture realistic video recordings of the intended subject matter. Data was collected from real police car videos as well as from simulated situations which were then used to test the developed algorithms.
3. Test the system's ability to determine undesirable situations. The situations included open door of the stopped car, person running out of the stopped car, and officer falling down event. Algorithms were developed and tested for detecting all these events successfully.
4. The results were disseminated with the proper acknowledgement of the NIJ funding in scientific venues. The commercialization of the system is open for anyone interested in it.

Introduction

The objective of this project is to develop an intelligent in-car video system that will be able to analyze the incoming video stream installed in police cars in real-time and detect critical events. The detection of the critical events will result in certain automated decision making, the simplest of which is to alert the police headquarters and call for back up help. Other decisions after the detection of such critical events include recording the scene with higher resolution video camera. The video analyses could also be linked

to data from GPS and ALPR devices to supply supporting information such as location of the incident and the identity of the car stopped.

There were several specific goals:

1. Build a computer system to analyze video and identify certain undesirable events.
2. Test the system in the laboratory with both static and dynamic test targets to determine the technical quality differences between the two video formats.
3. Deploy the test system in police cars and in simulations in order to capture realistic video recordings of the intended subject matter.
4. Test the system's ability to determine undesirable situations.
5. Design and build a video recording system capable of recording both standard resolution and high definition video.

Problem Definition and Goals

The following were the top level goals of this project:

1. From the video camera on board a police car, detect (among many other cars in view) the one being pursued and localize it in the video frame.
2. Detect when the car stops.
3. Detect and track critical events:
    a. Open door;
    b. Person running out of pursued car;
    c. Officer walking to the car;
    d. Officer falling;
4. Integrate these components and raise an alert back to the police headquarters automatically.
5. Include location and car identity information obtained from automatic license plate reader and GPS location device, if possible.

Acquisition of Hardware

In the Spring 2008, we acquired the following hardware to build the infrastructure for this project:

1. Dragonfly Express camera from Pointgrey with high frame rate with a firewire interface.
2. A Dell laptop running Microsoft Windows XP with sufficient memory
3. A HD digital video camera for collecting high definition video data.
4. A portable hard disk (160GB) for keeping and porting the data.

Our prototype software was developed on this Dell laptop using the OpenCV library for image processing and machine vision with real-time operation as our goal.

Data collection

We obtained data mainly from two sources:

1. Real video footage from local police forces in traffic stops under various environmental conditions: daylight, night-time, sunny, suburban, highway traffic, etc. The real video footage we were able to obtain from the local police department was limited to routine traffic stops and did not contain any of the critical events we were interested in. As a result, we had to shoot mock videos for these situations. We attempted to get help shooting more realistic video sequences under controlled situations with the help of the local police academy, but this never materialized.
2. Video footage shot by us to collect data for particular situations to be able to test. The specific footage we shot included:
    a. Door of the pursued car opens.
    b. Car passenger/driver runs out.
    c. Officer approaching the car gets hurt and falls down.
    d. Pursued car makes turns.
    e. Lighting conditions during the pursuit change (for example, alternating shadow and sunny stretches in the video).

    Even though these scenes were simulated mock scenarios shot by us, we were careful to incorporate various lighting conditions (e.g., nighttime, daytime, etc) as well as various traffic and environmental conditions. The camera was placed in our car at the same location and with the same field of view as an in-car video camera in a police car. We drove our video camera equipped car for long periods of time on real streets with various traffic conditions. The test video sequences we used for our algorithms were a representative set reflecting these conditions.

Algorithm Development

Most of our effort was spent on developing the various machine vision and video processing algorithms to work in real-time on the in-car video data. We have successfully developed and tested algorithms for the following tasks:

- Detection and localization in the video frames of the car being pursued. (accomplished)
- Detection of the stopping of the pursued car. (accomplished)
- Detection of the car door opening. (accomplished)
- Detection of a person leaving and running out of the car. (accomplished)
- Detection of the officer in the video frames and tracking his position over time as he/she moves. (accomplished)
- Detection of the officer falling down in the video frames. (accomplished)

We will describe our approach for each of these critical events and how we approached developing and implementing them in more detail below. We have made certain assumptions that hold in the given context of this application in order to be able to do real-time image processing. We briefly describe the overall approach and the general assumptions we have made. Then present the details of each algorithm. Finally, we present some of the results and some performance evaluation of our methods.

The fundamental problem is to identify and track dynamic objects in a changing environment and illumination conditions. The environment is dynamic because the scene changes due to a camera installed in a moving vehicle and the scene containing multiple moving objects independent of the moving camera. The illumination varies due to changing environmental conditions such as shadows due to buildings and

trees along the path, changing cloudiness and changing daylight and nighttime conditions. Although there have been numerous publications on general object recognition and tracking, or combination of them, not many works could be successfully applied in real-time for the in-car video, which has to process the input on-the-fly during the vehicle movement. Typically, motion analysis methods assume a static camera with moving objects to be detected and tracked in the scene or a moving camera with an unchanging environment. The first instance is typically used in various automated video surveillance applications [17, 18]. The second instance is typically used for "structure from motion" types of applications [11] in which the three-dimensional structure of the scene is extracted assuming a non-moving camera or ego-motion estimation applications [9] in which the motion of the camera is extracted assuming a static scene. Normally, combining a moving camera and a scene with moving objects in it is a very difficult problem. In our work, we have tried to tackle this combination of dynamic camera in a dynamic scene is addressed. In order to get our systems to work, we make assumptions that hold true in the particular application of a video camera in a police car. This means that the motion of the camera is typically restricted (e.g., motion in a plane parallel to the ground). The motion of the police car is in the same direction as the car being pursued, and, therefore, this allows simplification of the type of motions in the field of view of the camera due to the oncoming cars, or changing scenes along the sides of the road to be ruled out from being detected and classified. Moreover, once that target vehicle in the front (vehicle of interest) has been stopped and the police car pursuing it has come to a stop, the analysis of the scene reverts to the case of a static camera analyzing a static scene with moving objects in it.

In out context, in-car video is from a camera typically facing forward in a vehicle, which is the simplest and most widely deployed system on police cars and high-tech vehicles. It records various traffic and road situations ahead and is typically used to record events during a routine stop or during a critical incident as

Figure 1: A typical frame of in-car video where multiple vehicles are moving in front of the camera

recorded video data. The objective of this work is to detect vehicles ahead or those being pursued, and track them continuously in the video to facilitate the goals stated above. It is certainly not easy for a single moving camera to yield the information quickly from dynamic scenes without stereo or other sensors' assistance [16]. The main difficulty is, first, the numerous variations of vehicles in color, shape, and type even when they are mainly viewed from back. The vast amount of vehicle samples is extremely difficult to model or learn [13]. Second, the vehicle detection must be done automatically along with the video tracking. Although there have been many works on tracking, most of them assume easily detectable targets or known initial positions. Such approaches cannot satisfy the real time assistance needed during driving of the police vehicle. Third, the in-car video taken on roads may confront drastic variation of environment and illumination [3]. For example, quick transition through shadow and sunny locations in urban areas, dim lighting at night, loss of color on a cloudy day, shining highlight on vehicles, large scale changes of target vehicles due to varied depth, occlusions between vehicles and background, and so on make the feature extraction, recognition, and tracking unstable and difficult.

Our novel methods are along the following lines. First, we select and detect the most common low-level features on vehicles that are robust to changes of illumination, shape, and occlusion. This avoids high-level vehicle and scene modeling and learning that are mostly unstable, time consuming, and not sufficiently general. Second, we focus on the horizontal scene movement for fast processing based on the configuration of the vehicle-borne camera and the position information to obtain. We do data reduction of the video frame content from a two dimensional (2D) information to a one-dimensional (1D) representation. This reduced 1D data is then tracked over time to look for trends in the space-time domain. We track feature trajectories in such temporal profiles so that the real time vehicle and background classification can be done in this reduced dimension. Third, we put more weight on the understanding of motion behavior of the vehicles than object shape analysis in identifying targets. As we mentioned above, the use of shape and color based information for vehicle localization, tracking, and identification is a difficult task due to various environmental reasons. Our method results in the detection and tracking of the target vehicles more robustly, over long periods of time. We use Hidden Markov Model (HMM) to model the continuous movement (and time dependence) of the features so that the results will be influenced less by sensitivity to thresholding during low level preprocessing. Optimal selection of target vehicles will be given in terms of probabilities.

There are many related works of in-car video or images to identify cars [2, 12, 4, 5, 8, 7, 15]. Most of them are shape based methods that usually suffer from brittleness due to the variety of vehicles and backgrounds and have only worked on limited scenes for short periods of time. Because of the lack of robustness in those algorithms, subsequent sophisticated decisions can rarely be applied to video in real time.

The significance of this work lies in its temporal processing rather than shape analysis in target identification. We flip the time and spatial domains to look at problems based on the properties of vehicle generated motion. This extends the processing to long videos and facilitates the processing for many vehicle related recognition tasks. The modeling of motion behavior of scenes with HMM makes the vehicle identification with certainty and this improves the robustness for scene classification as well. Also, the profiled 1D arrays reduce dimensionality of data, thus achieving real-time processing to identify and track vehicles simultaneously.

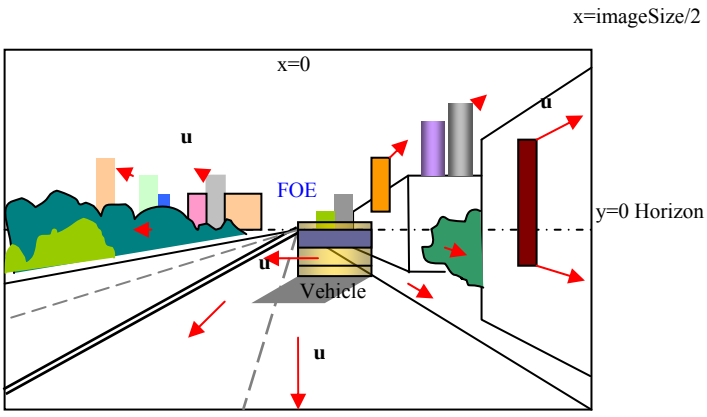
In the following, we will introduce feature detection, selection, and tracking in Section 2, and address the modeling of feature trajectory and the motion properties of scenes in Section 3. Probability based modeling of tracked targets and background is given in Section 4 and HMM based scene identification and tracking is described in Section 5. Experimental results and discussion are given in Section 6.

Vehicle Feature Detection in In-Car Video

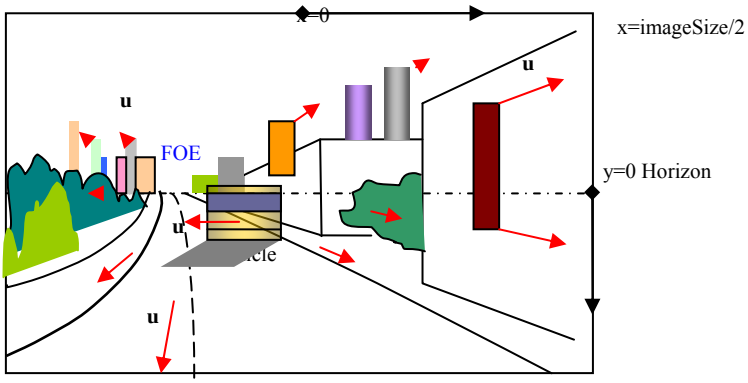
Dynamic Environment of Vehicle Scenes

A general assumption we make is that our own car with a video camera should not turn away completely from target vehicles on the road. For the vehicle-borne camera, a pure rotation without significant translation, e.g., turning at a street corner, does not provide sufficient information to separate the targets and background motion, because such a rotation generates almost the same optical flow in entire frame. Therefore, a translation of the observer vehicle is required. Another reasonable assumption is the continuity of the vehicle and camera motion, which is guaranteed in general according to the driving mechanism of four-wheeled vehicles.

Assume the camera centered coordinate system is O–XYZ with the X axis toward right, Y axis facing down and Z axis in the vehicle moving direction. Denote the 3D position of a scene in the camera centered coordinate system by (X,Y,Z) and its image coordinates by (x,y). A typical view of the in-car video is depicted in Figure 2, where an FOE (focus of expansion) is located in the center part of the image frame $I(x,y)$. In many cases, FOE is overlapped with the vanishing point of the road when the vehicle is pursuing with a forward translation motion without steering (Figure 2a). On a curved road, the steering is not zero and there is a rotation component to the motion of the observer vehicle. This causes the FOE to shift away from the image center as depicted in Figure 2b. A target car that is moving on the road may change its horizontal position and scale when it changes lane and speed, but may still retain its position within the road even when its instantaneous image velocity $\mathbf{u}(x,y)$ changes dramatically. The background, however, has a sort of motion coherence, with the flow spreading out gradually from FOE towards the sides of the image frame. In general, the image velocity increases as the scene moves closer. This cue is sufficient for humans to classify vehicles and background even without knowing the shape of objects. We will model this motion coherence of scenes for automatic background and vehicle separation.



(a)



(b)

Figure 2 Typical views of in-car video with red arrows indicating optical flow or image velocity. (a) Straight road, (b) mildly curved road.

According to perspective projection of the camera, the image position of an object is

$$x(t) = \frac{fX(t)}{Z(t)} \qquad y(t) = \frac{fY(t)}{Z(t)} \tag{1}$$

where f is the camera focal length in the central projection of the video camera. Denote the relative translation of the object to the camera by $(T_x(t),T_y(t),T_z(t))$ in the camera coordinate system, and denote the rotation of the observer vehicle (around the Y axis) by $(R_x(t),R_y(t),R_z(t))$, where the pitch and roll of the observer vehicle have $R_x(t)\approx 0$ and $R_z(t)\approx 0$ on a flat road. The relative speed of the target to the observer vehicle, $(V_x(t),V_y(t),V_z(t))$, is then

$$(V_x(t),V_y(t),V_z(t)) = (T_x(t),T_y(t),T_z(t)) + (X, Y, Z)\times(R_x(t), R_y(t), R_z(t)) \tag{2}$$

where (X,Y,Z) is the object position in 3D space related to the camera. By differentiating Eq. 1 with respect to t, and replacing related terms in the result using Eq. 1 again, the horizontal component of image velocity on an object (or vehicle) becomes

$$v(t) = \frac{\partial x(t)}{\partial t} = \frac{fV_x(t) - x(t)V_z(t)}{Z(t)} \tag{3}$$

Replacing $V_x(t)$ and $V_z(t)$ with Eq. 2, and setting $R_x(t)=0$ and $R_z(t)= 0$, we get

$$v(t) = \frac{fT_x(t) - x(t)T_z(t)}{Z(t)} - \frac{x^2(t) + f^2}{f} R_y(t) = vt(t) + vr(t) \tag{4}$$

where

$$vt(t) = \frac{fT_x(t) - x(t)T_z(t)}{Z(t)} \qquad vr(t) = -\frac{x^2(t) + f^2}{f} R_y(t) \tag{5}$$

are components of horizontal image velocity yield from translation and rotation. If the observer vehicle moves on a straight road, i.e., $R_y(t)=0$, we simply have $V_x(t)=T_x(t)$ and $V_z(t)=T_z(t)$.

Real-time Feature Extraction in Video Frames

The segmentation of vehicles from background is difficult due to the complex nature of the scenes; occlusion by other moving vehicles, complicated shapes and textures, coupled with the ever changing background. The presence of specula reflection on the metallic surfaces (floating on vehicle) and shadows on most cars (deformed and spread) make the color and shape information unreliable. In particular, the reflected scenes (trees, buildings, etc.) from the vehicle back windows always interfere with the shape analysis. These phenomena render ineffective the segmentation that relies on color to detect regions of the vehicles. To cope with these variations, we have examined a variety of video clips and selected three types of low-level features that will result in reliable detection of a vehicle. These features are horizontal line segments, corner points, and intensity peaks.

Corner Detection

An important feature detectable in the video is corner, which may appear on high contrast positions and high curvature points (excluding high contrast edges). In daytime videos, corner points on a vehicle surface or background scenes keep their positions stable over time on surfaces so that they provide coherent motion information of regions. At occluding contours of vehicles (usually on two sides of the vehicles), however, the corner points detected are formed by foreground and background scenes, which do not physically exist and are extremely unstable in the images during the relative motion of vehicles and background scenes.

We use SIFT feature for corners in the video frames in real time [6, 10]. One example is shown in Figure 3 where corners are marked in green circles.

Figure 3. Corner detection in video sequence. Corners marked in green circles.

Line Segment Detection

We have noticed that the shapes of the backs of vehicles typically contain many horizontal edges formed by vehicle tops, windows, bumpers, and shadows. Most of these structural lines are visible during daylight and even in nighttime conditions, which indicate the existence of a vehicle. For vehicles with partially obscured backs, the detection of horizontal line segments on the vehicle is still stable. The vertical line segments, however, are not guaranteed to be visible due to a curved vehicle body, frequent occlusion by other cars, and occlusion over constantly changing background during the vehicle motion.

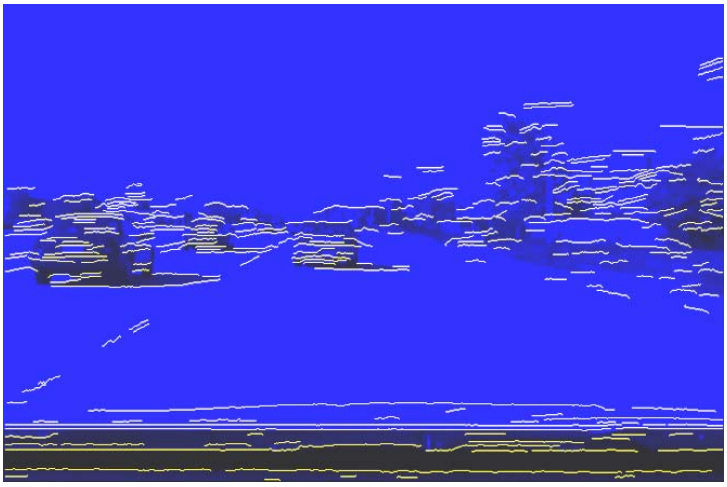


Figure 4. Tracked edge points form horizontal line segments in video frames that can characterize vehicles.

To extract the horizontal line segments, we convolve each video frame with a vertical differential operator $\partial I(x,y)/\partial y$, which results in the y-differential image $I'_y(x,y)$ of the intensity. Then, an edge following algorithm searches peaks in $I'_y(x,y)$ where the contrast is over a threshold δ_1 to produce horizontal line segments. Starting from left end of a line candidate, the horizontal search of edge points has a vertical position tolerance of ±2 pixels, and it selects candidates with the same sign of edge in tracking. It also uses a second threshold lower than δ_1 to bridge line segments at weak-contrast points, which allows a 3-pixel trail further rightward after reaching the end point using δ_1. The tracked edge points form a line segment if they satisfy constraints on the minimum and maximum lengths, high contrast, and near horizontal orientation. These conditions are determined loosely to yield more segments after examining our videos empirically. We pass more information to the later probabilistic processing rather than cutting it off abruptly using tight thresholds at this stage.

Line tracking may break down due to highlights on horizontal structures, insufficient resolution on distant cars, and scale changes when the vehicle changes depth from the camera. The results may also contain random segments from static background such as long roofs, painted marks on road, and other building structure. Figure 4 shows a frame of line extraction overlapped with intensity image.

Intensity Peak Detection for Night Scenes

The intensity peaks from tail and head lights of moving vehicles and static street lamps are used as features when the lighting conditions are poor, edges and corners are mostly invisible, and colors are hard to distinguish. Hence, we detect intensity peaks from vehicle lights in order to obtain more evidence of the vehicle presence. We use a large Gaussian filter to smooth the image and find the local maxima beyond a threshold determined adaptively. These peaks will be further tracked across frames for the direction and speed of moving targets. This creates a history that enhances the vehicle detection and helps in the separation of a vehicle from the background. Figure 3b show examples of the intensity peak detection and tracking.

Figure 5 Intensity Peaks (marked in squares) extracted on traffic lights, tail lights and front lights of vehicles.

After preprocessing and feature extraction, it is noted that a single data source alone cannot guarantee reliable detection results. Different from many other works that put more effort into vehicle shape analysis in individual video frames; this work uses the motion properties of scenes to identify targeting cars. The continuity of a vehicle's motion provides more robust observation than features in complex shape modeling of various vehicles that is constantly affected by occlusion, illumination and background changes. We look into the motion characteristics of tracked objects and separate them as static background or moving cars. By showing the continuous motion of extracted points (without color and shape) to human subjects, we have confirmed that humans are capable of separating vehicles from background using these cues, after knowing that the source is from an in-car video. By adding line segments extracted over time, the identification is even more reliable.

Profiling Features and Temporal Tracking

Vertical Profiling of Features to Realize Data Reduction

To speed up the processing and yield robust results for real time target tracking, we project the intensity $I(x,y)$ in each video frame vertically to form a 1D profile, $T(x)$. Consecutive profiles along the time axis

generate a condensed spatio-temporal image, *T(x,t),* used for analyzing the scene movement. The vertical projection of intensities through a weight mask *w(x,y)* is implemented as

$$T(x,t) = \sum_{y=-h/2}^{h/2} w(x,y) I(x,y,t) \tag{6}$$

where *h* is the image height. The weight distribution, which will be computed in detail in Section 4.3, is high at regions with high probability of containing vehicles and low elsewhere in the image. The traces in the condensed spatio-temporal image show movements of long or strong vertical lines in the video frame. Slanted lines and short edges in individual video frames will not be distinct in the computed 1D intensity profiles. Figure 6a shows such an image, which is a compact and robust representation for recording and estimating vehicle motion.

In addition to the intensity profiling, we also profile features extracted in the video frames vertically to generate feature traces in the spatio-temporal images. For example, we profile the number of horizontal line segments at each position of *x* by accumulating

$$T_l(x,t) = \sum_{y=-h/2}^{h/2} w(x,y) C(x,y,t) \tag{7}$$

where $C(x,y,t)$ takes value 1 on a horizontal line segment and 0 otherwise in frame *t*. Such a result can be found in Figure 6b, where the bright stripes accumulated from many line segments show the motion of vehicles. Many long and horizontal lines such as road paints and wires in the scene add equally to all the positions of the profile without influencing the traces of vehicles. Due to the existence of multiple horizontal lines on a vehicle, the vehicle position appears brighter than other positions of background.

Instant illumination changes happen in many cases when entering a tunnel, penetrating shadow and sunny locations, and lighting up by other vehicles. Such changes alter the intensity of entire frame. In the spatio-temporal profiles, such illumination changes appear obviously horizontal edges over the entire image width. Its influence to the vehicle position is successfully filtered out by taking horizontal derivative of the profiles.

t (frame number)

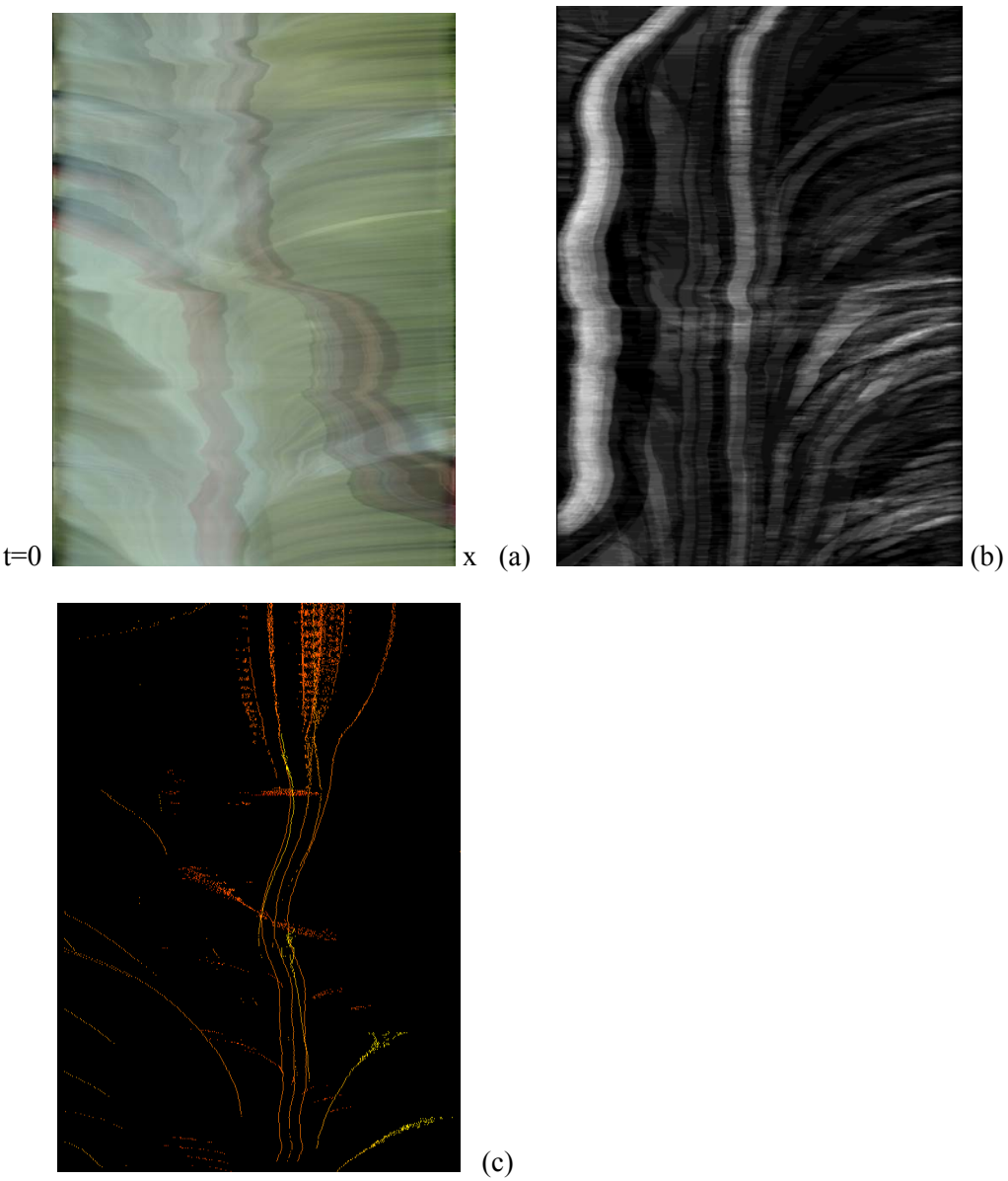


Figure 6. Examples of profiles using (a) intensity (here from tree background and a red car), (b) horizontal line segments, (c) intensity peaks in a night scene before a car stops (tail break light in red).

Compared to the intensity profiles with smooth traces from vertical lines in the video, the profiling of horizontal line segments yields noisy traces due to instability of line detection; a line may fail in extraction by fixed thresholds due to subtle intensity changes in consecutive frames. Because the lighting condition, background, and camera noise vary frame by frame, the line segments found in the images do not always appear at consistent locations and in the same length, which have to be post-processed according to their time coherence. Also, as is evident in Figure 6b, many traces may not belong to a vehicle but to the background. Fortunately, the intensity profile from the numbers of horizontal lines is

significantly less on the backgrounds than on the vehicles. Unlike the traces of a vehicle, background segments lack coherence in time and appear more randomly than those on a vehicle.

Tracking Features in Temporal Profiles for Their Motion Characteristics

The intensity profile shows the horizontal positions of vertical features and ignores the horizontal features at any object height. This produces a compact image for understanding of the horizontal motions in the video. The image changes caused by jitter in camera roll and tilt, $R_z(t)$ and $R_x(t)$, (vehicle roll and pitch) on slanted or uneven roads are significantly reduced in the profiles. We can then develop a robust tracking of significant feature segments in the profiles.

Tracking of intensity profiles is done by horizontal differentiation of $T(x,t)$, i.e., $\partial T(x,t)/\partial x$. The edges are marked as $E(x,t)$ and a maximum span for search is set to search consecutive trace points as time progresses. At the same time, $\partial T(x,t)/\partial t$ is also computed for finding horizontal edges, because a fast moving trace is very slanted with its tangent value $\partial x/\partial t$ in $T(x,t)$. We thus can link horizontal edges in $E(x,t)$ to track the traces with high image velocity $v(t)$. For those very long horizontal segments in $E(x,t)$, they are mainly from the instant illumination changes described above and are ignored in the result. This processing results in the image position $x(t)$ and image velocity $v(t) = x(t) - x(t\text{-}1)$ in pixel.

To preserve the continuity of motion, we compare the image velocity v(t-1) and v(t) from tracked points and new point candidates, and select the successive point from candidates such that |v(t) - v(t-1)| is kept minimum. Besides the requirement of high contrast, the sign of trace, i.e., sign at $E(x,t)$ is used as reference in tracking as well. The approach to examine the motion continuity is also applied in tracking intensity peaks and corner points in the profiles to avoid noise from instant light changes.

Among all types of traces, the horizontal line segment piles provide the most salient clue of vehicle presence. Tracking traces of line segments in $T_l(x,t)$ is done by locating the center of each trace and following its movement continuously. We track the center because the locations of endpoints of each trace are usually unreliable especially for vehicles far away from the camera. In implementation, we filter $T_l(x,t)$ horizontally with a smoothing filter to obtain major traces as shown in Figure 7. These traces are marked at their peaks $x(t)$ above a threshold for the centers (Figure 7). The threshold is decided adaptively at each instance t according to the average and standard deviation of $T_l(x,t)$.

Random line segments on background and instantaneous light changes over the entire frame (tail braking lights, police alarm lights, etc.) will cause a long horizontal line in $T_l(x,t)$. However, these will be ignored in the processes of finding peak (they have no distinct peak) and tracking them over long periods of time. Figure 7c shows some of them.

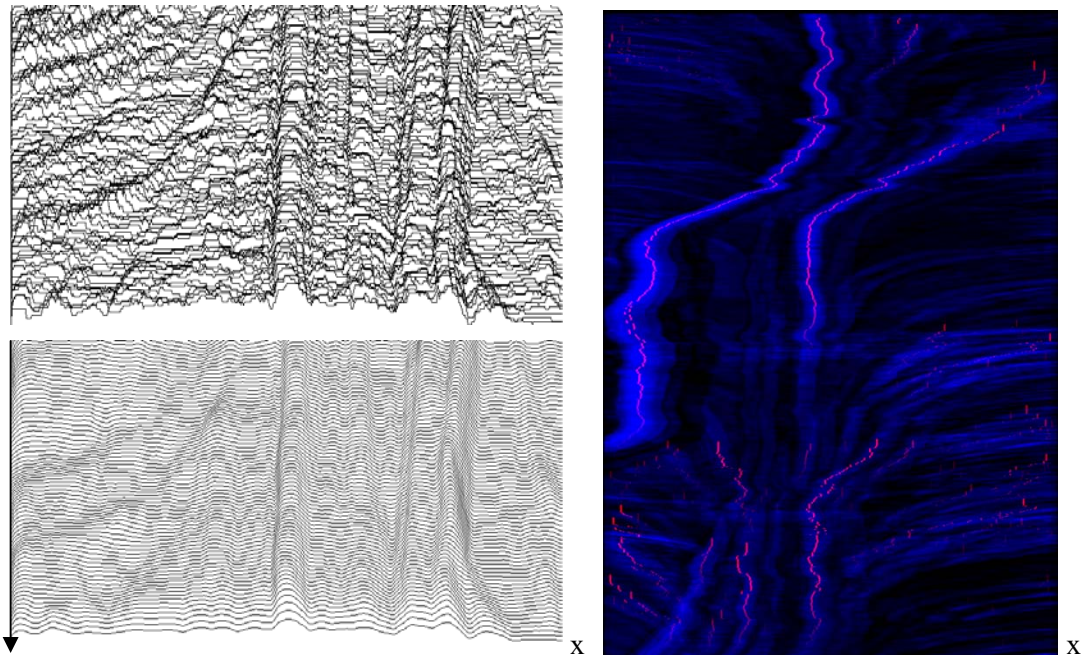


Figure 7 Tracing centers of line segment clusters. Vertical axes are the frame number or time. (a) Profiled distribution of line segments and its smoothed distribution, (b) Center of traces (in purple) overlapped with traces.

Motion Behaviors of Scenes Generated from Vehicle Motion

The motion properties in the spatio-temporal representation are the key evidence in our vehicle identification. The background scenes captured from a forward-looking camera pursue a translation in the Z direction the camera. It is well known in robotics and ITS that, at the image height where the camera rays are horizontal, we can set a sampling line (y=0 for horizontal camera) to collect Epipolar Plane Image (EPI). The motion traces in the EPI has the following properties: (1) Background objects pursue hyperbolic trajectories expanding from the FOE. (2) The curvature of a trajectory is high if the object is close to the road (motion vector of camera) and is low (showing flat traces) if it is far from the road. (3) Their image velocity (tangent of curved trajectories) is high at buildings passing by, and is low at the distance down the street. On the other hand, vehicles tracked within the road may stay in the image frame even they drive irregularly in a zigzag way.

Similarly, our spatio-temporal representation composed of 1D profiles from the image frame shows the same properties as illustrated in Figure 8. The background motion is generated from the motion of camera on observer vehicle; features move sideways in the frame with increasing image velocities. Moreover, the image velocity is proportional to the image position *x(t)*. The vertically condensed profiles give more robust results from long vertical features than that in a single layer EPI because they ignore small features and the changes in camera tilt and roll caused by vehicle shaking on slanted or uneven roads.

Some real background trajectories can be observed in Figure 7, where the image velocity corresponds to the orientation of trajectory; the higher the velocity, the more slanted the trace is in the profiles. In contrast to features in the background, a target vehicle will be kept in the field of view and its horizontal image velocity is relatively low. We can observe curved vehicle traces maintained in the profiles.

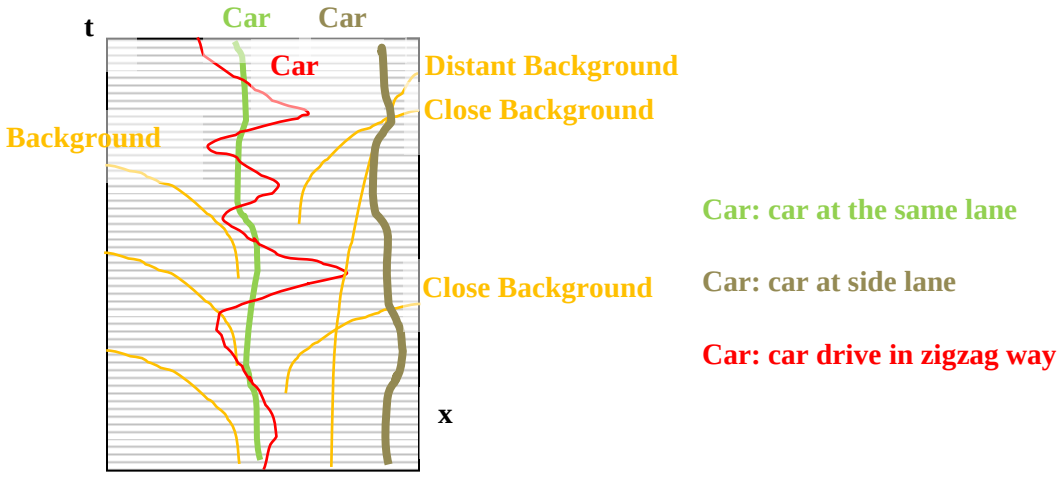


Figure 8 Motion behaviors of vehicles and background in trajectories shown in stacked 1D-profiles along the time axis. The position and tangent of curves show the motion information with respect to the camera.

By tracking the trajectories for a short while, our method will be able to identify the vehicle and background with certain probabilities. Continuous tracking further enhances the confidence of the decision. This is done by computing the probability of a tracked object as a target vehicle or background at each moment using a Hidden Markov Model (HMM). The information used are the horizontal image position $x(t)$ and the velocity $v(t)$ of a tracked trajectory, simply denoted by (x,v). These two parameters are not independent under the vehicle motion model; neither one will determine the object's identity alone. By formulating the classification problem in a probabilistic framework, we can avoid the accumulation of nonlinear decisions from multiple thresholds in segmentation and tracking, and thus improve the quality of vehicle identification.

As one can observe, the traces in the profiles of horizontal line segments provide strong evidence of the vehicle presence, in spite of their inaccurate boundaries that are caused by curved shapes or break lines on the vehicles. On the other hand, the traces from profiling intensity and corner points are much more precise and smooth. Hence, we designed our strategy to combine these features for a correct decision on vehicle detection. By referencing to left and right traces in $T(x,t)$ that bound a vehicle trace in $T_l(x,t)$, the further search of the vertical location of vehicles is narrowed in the video frame. A box hence can be located on a vehicle for display in every video frame.

Computing Forward Probability of Scenes for HMM

Let us model the motion with HMM in order to determine the states of a tracked object as a background component or a moving vehicle. We assign two hidden states at any time instance t: car and background denoted by C_t and B_t to describe every trace extracted. For an object feature, the observations of the two states are image position $x(t)$ and the horizontal image velocity component $v(t)$ of $\boldsymbol{u}(x,y)$, both are continuous distributions rather than discrete events described in conventional HMM. Array $(x(t),v(t))$ is obtained from each trajectory tracked over time in the condensed profiles. We first calculate the forward probability of appearances, P(x,v| B) and P(x,v|C), for background and vehicles in this section in order to estimate likelihood, P(B| (x(t),v(t))) and P(C| (x(t),v(t))), later in the next section based on observation $(x(t),v(t))$. In the 3D space around the camera, X is the horizontal distance of an object from the camera axis (from road if the vehicle is on a straight lane). For background, we have $|X|>D$ where $2D$ is the average road width. By defining the probability distribution of scenes, we can obtain high response to the expected scenes and events.

Probability of Background in Image Domain

Traveling along Straight Roads

For static background, we can assume its probability distribution with respect to the camera as shown in Figure 9, if the observer vehicle is moving on a road. The heights of background features are homogeneously distributed to include high buildings and trees. Assuming a pure translation of observer vehicle first (i.e., R_y=0), the absolute speed V of the vehicle/camera (in Z direction) also follows a normal distribution, i.e., $p(V) \sim G(S, \sigma_V)$ when $V>0$, where S can be set at a proper value such as 50km for pursuing, and otherwise p(V)=0.

If a 3D point is on background, we have its V_x=0 and V_z= -V approaching to the camera, and X is fixed. Then, the image velocity in Eq. 3 becomes

$$v(t) = \frac{fXV}{Z^2(t)} = \frac{Vx^2(t)}{fX} \qquad \text{for } V>0 \qquad (8)$$

This will draw dense function curves in $(x(t),v(t))$ space for various X, as shown in Figure 10. Intuitively, this can also be confirmed from the background trajectory in Figure 6. If the vehicle has roughly a constant speed, $x(t)= fX/(Z_0-Vt)$ is a hyperbola trajectory according to Eq. 1 from an initial depth Z_0 in profile $T(x,t)$. In Figure 10a, an object with a far distance from the road (large $|X|$) has a flat trajectory while an object close to the road has a strongly curved trajectory. As to the distribution of background features along the X direction of the camera/observer-vehicle, we can assume that it follows a flipped Gaussian distribution, i.e., $p(X) \sim 1-\exp(-X^2/2D^2)$ to realize what Figure 9 depicts.

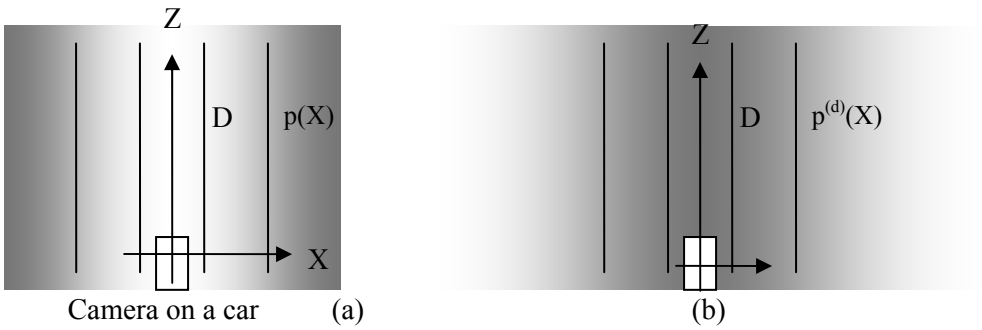


Figure 9 Probability distribution of background aside the road and detectability displayed in dark intensity. The higher the probability, the darker the intensity displays. (a) Background feature distribution on road sides, (b) Detectability of features

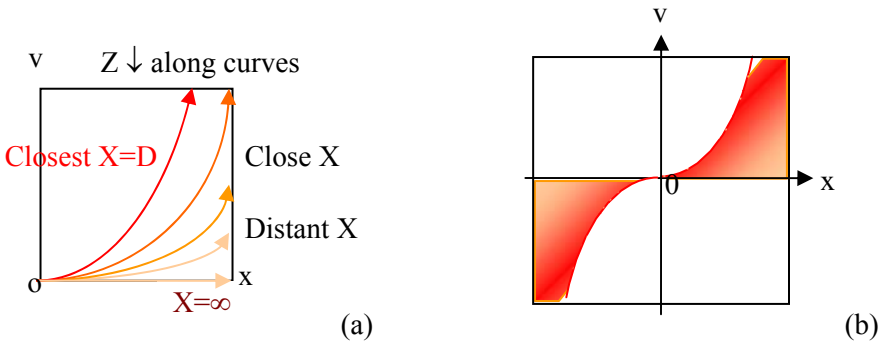


Figure 10 Relation of the horizontal image velocity and the image position of background scenes. (a) Motion traces of background points on right side of road. As the depth Z is reduced, background moves fast in outward direction. (b) Traces for background on both sides of road. The colors correspond to different *X* from the camera axis (or simply road).

In general, if the PDF of a random variable, χ, is $p_\chi(\chi)$, and β is a monotonic function of χ, i.e., $\beta = f(\chi)$, then the PDF of β can be calculated as

$$p_\beta(\beta) = p_\chi\left(f^{-1}(\beta)\right)\left|\frac{\partial f^{-1}(\beta)}{\partial \beta}\right| \quad \text{or} \quad p_\beta(\beta) = \frac{p_\chi\left(f^{-1}(\beta)\right)}{\left|\frac{\partial f(\chi)}{\partial \chi}\right|} \qquad (9)$$

For a multivariate function f (β and χ are vectors of the same length), the PDF of the functions can also be derived similarly in a Jacobian format [14], i.e., $|\cdot|$ in Eq. 9 becomes the Jacobian.

Now, let us compute the forward probability $p(x,v \mid B)$ for background, assuming $(X,Z) \in B$ is a background point. Here the original variables are *X, Z,* and *V* in the 3D space and their functions are dependent image parameters *x* and *v*, described by Eq. 1 and 5. Because the inverse mapping from *x, v* to

X, Z, V is not a unique transformation between the same number of variables, we have to use Bayes theorem and conditional probability to include all the possible cases of *X, Z, V* to produce probability at a pair of (*x,v*). We loop variable X to compute p(*x,v* | B), i.e., for all possible *v* and image position *x*,

$$p(x,v \mid B) = p(x,v \mid (X,Z) \in B) = p(x \mid (X,Z) \in B) \times p(v \mid x,(X,Z) \in B) \quad \text{Bayesian}$$

$$= p(x \mid (X,Z) \in B) \times p\left(X,V \mid x, v = \frac{Vx^2}{fX}, (X,Z) \in B\right) \quad \text{Eq. 9}$$

$$= p\left(X,Z \mid x = \frac{fX}{Z}\right) \times p\left(X,V \mid x, v = \frac{Vx^2}{fX}, (X,Z) \in B\right) \quad \text{Eq. 9}$$

$$= \int_X p(X) \times p\left(Z = \frac{fX}{x} \mid X\right) \times p\left(V = \frac{vfX}{x^2} \mid X\right) \times \left|\frac{fX}{x^2}\right|^2 dX \quad \text{Bayesian and Eq. 9}$$

(10)

In above deduction, we loop X over its whole scope and determine the probability at (*x,v*) by mapping the pair to variables *Z*, and *V* in the 3D space. Input original probability distribution of *Z* and *V*, the background probability p(x,v) becomes

$$p(x,v \mid B) = C \int_X (1 - e^{-\frac{X^2}{2D^2}}) \times e^{-\frac{(\frac{vfX}{x^2} - S)^2}{2\sigma^2}} \times \left|\frac{fX}{x^2}\right|^2 dX \tag{11}$$

where *C* is a constant and a Jacobian | · | is included. Because *p*(*Z*) is irrelevant to *Z* for the background according to the definition in Figure 9a, it is treated as a constant in Eq. 11 and is included in constant *C*.

In real situation, we further consider a visibility or probability of detection by multiplying $p^{(d)}(X)$. We can assume that the visibility of background scenes follows a PDF as

$$p^{(d)}(X) \propto 1/(|X| + 1) \tag{12}$$

where close objects on the side of the road have the highest visibility and scenes away from the road (|X| increases) have greater chances to be occluded. The PDF for background then can be written as

$$p(x,v \mid B) = \int_X p^{(d)}(X) \times p(X) \times p\left(Z = \frac{fX}{x} \mid X\right) \times p\left(V = \frac{vfX}{x^2} \mid X\right) \times \left|\frac{fX}{x^2}\right|^2 dX \tag{13}$$

$$= C_1 \int_X \frac{1 - e^{-\frac{X^2}{2D^2}}}{|X| + 1} \times e^{-\frac{(\frac{vfX}{x^2} - S)^2}{2\sigma^2}} \times \left|\frac{fX}{x^2}\right|^2 dX$$

where C_1 is a constant for normalization, i.e., $\int_v \int_x p(x,v \mid B) dx dv = 1$. After the probability for (*x,v*) is established, we normalize it in the entire scope to determine C_1 and create a 2D PDF as shown in Figure 11. The brightness indicates the probability of background, and the values form a look-up table for referring in real time tracking and computation of HMM.

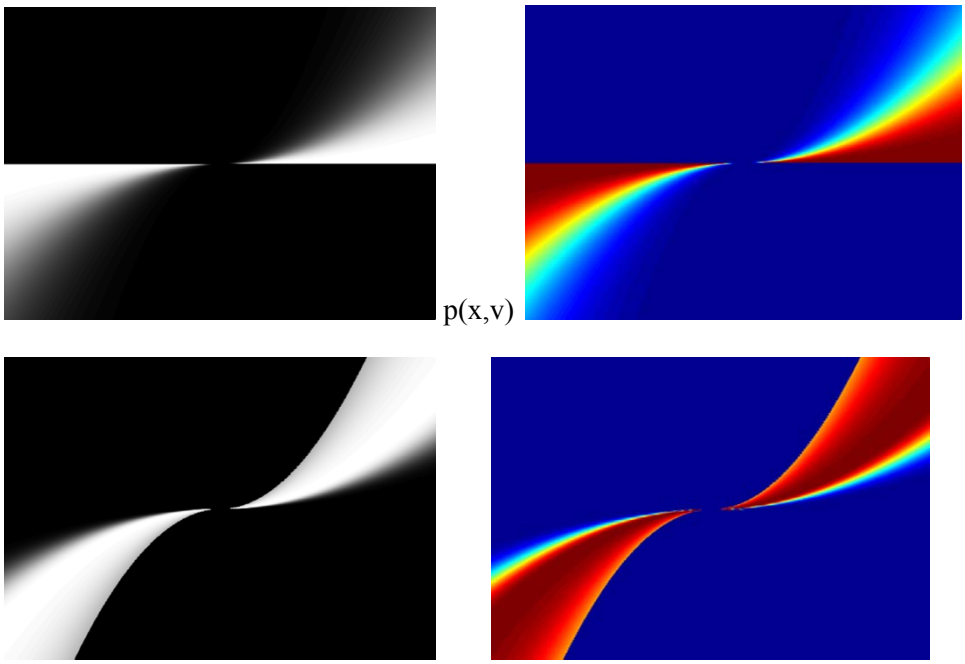


Figure 11 Probability distribution p(*x*,*v*) of background in motion. The intensity corresponding to probability is scaled for display. Top: Wide Street; Bottom: Narrow Street.

Traveling on Curved Roads

During smooth driving of the observer-vehicle, its steering change should be small along a straight and mildly curved road. Also, we are not considering the target identification if the observer vehicle performs pure rotation (e.g., turning a street corner). On a straight and mildly curved road, we can describe the vehicle rotation, $R_y(t)$, in a normal distribution with a small variance. According to Eq. 4 and 5 that also describe the camera rotation, we estimate a general distribution $p(x,v)$ including rotation by adding rotation component $vr(t)$ to the image velocity that was from the translation value $vt(t)$ so far. As illustrated in Figure 12, this is a vertical shift of $p(x,v)$ from $vt(t)$ by a term $vr(t)$ in a quadratic form. The updated scope of feature trajectories is intuitively drawn in Figure 12 for a certain steering angle R_y. If the rotation parameter $R_y(t)$ is not provided from the encoder of the observer vehicle, we have to include all the possible values of rotation of $R_y(t)$ in normal distribution for the probability distribution p(*x*,*v*), i.e.,

$$p(x,v \mid B) = \int_{R_y} p(R_y) \times p(x,v \mid R_y) dR_y \qquad (14)$$

$$= \int_{R_y} p(R_y) \times p\left(x,v \mid v = \frac{x^2 V}{fX} - \frac{x^2 + f^2}{f} R_y\right) dR_y$$

$$= \iint_{R_y X} p(R_y) \times p^{(d)}(X) \times p(X) \times p\left(Z = \frac{fX}{x} \mid X\right) \times p\left(V = \left(v + \frac{x^2 + f^2}{f} R_y\right) \frac{fX}{x^2} \mid X\right) dX dR_y \qquad \text{using (13)}$$

$$= C_{1r} \iint_{R_y X} e^{\frac{-R_y^2}{2\sigma_r^2}} \times \frac{1 - e^{\frac{-X^2}{2D^2}}}{|X|+1} \times e^{\frac{-\left(\left(v + \frac{x^2+f^2}{f} R_y\right)\frac{fX}{x^2} - S\right)^2}{2\sigma^2}} \times \left|\frac{fX}{x^2}\right|^2 dX dR_y$$

where C_{1r} is a constant to be determined in normalization of $p(x,v \mid B)$. Figure 13 shows such a result that is basically a Gaussian blur of the PDF of vehicle translation (Figure 11) in the velocity direction.

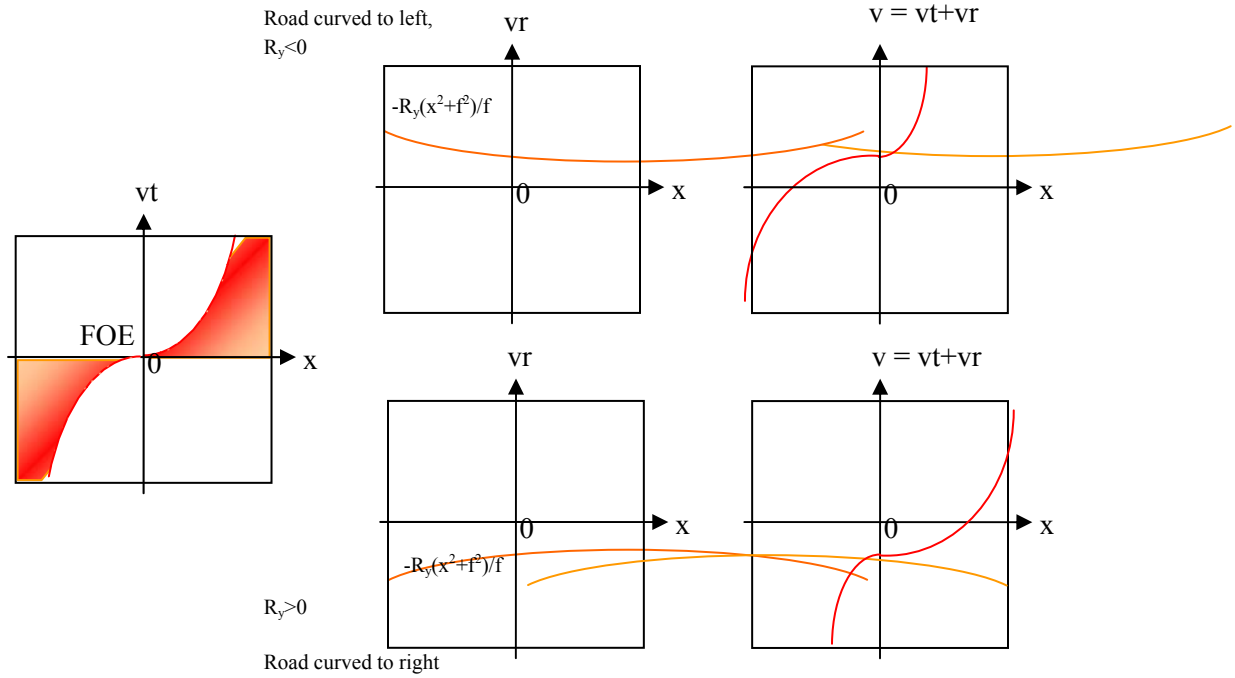


Figure 12 Adding a rotation component to translation to cope with curved road. (left) $p(x,v)$ distribution caused by translation of observer vehicle, (middle) motion component caused by rotation from steering of observer vehicle, (right) Shifting distribution $p(x,v)$ from translation yields a new distribution (area between two curves) including rotation.

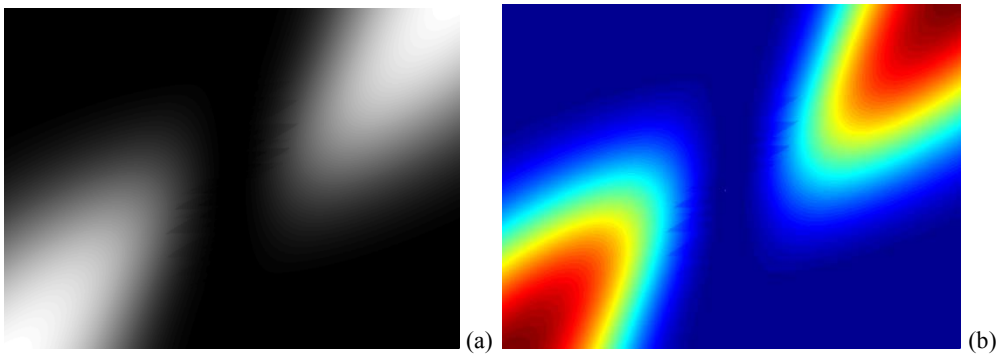


Figure 13 Probability distribution of background when the observer vehicle involves rotation. (a) A fixed steering angle at -10 degree per second, (b) for all possible angles in a Gaussian distribution

Probability of Target Vehicles in Image Domain

Assuming a pure translation of observer vehicle as shown in Figure 14a, we can assume the position of a target vehicle, $(X(t),Z(t))$, following a 2D normal distribution $G((0,F), (D,2F))$, where F is its average distance from the camera (e.g., F=30m). Here, $X(t)$ and $Z(t)$ are independent and D and $|2F|$ are used as the covariance. The projection of the normal distribution onto the image frame is depicted in Figure 14b, and this distribution is piled vertically in the cylinder according to a height probability distribution $H(Y)$ of vehicle features as shown in Figure 14c. In Y direction, the features are guaranteed to be detected near the ground due to the uniformity and high contrast of the vehicle shadow, tire, bumper, etc. against the homogeneous ground. However, features may not be found reliably at a higher Y positions due to a low height of a target vehicle, highlights on its metal top, and changing reflections on the back window of the vehicle. A group of gray cars may have their top color mixed with sky background. Here we design a feature distribution function in Y position as $p(Y) \sim (Y+3)/(Y+4)$, where $Y \in [-3m, 1m]$, assuming the Y axis is facing down and the camera position on the observer vehicle is 1m above the ground. Moreover, we can assume that the relative speed of the target vehicle to the camera, denoted by (V_x, V_z), also follows a normal distribution $G((0,0), (\sigma_x,\sigma_z))$ in a stable pursuing period, where V_x and V_z are independent as well.

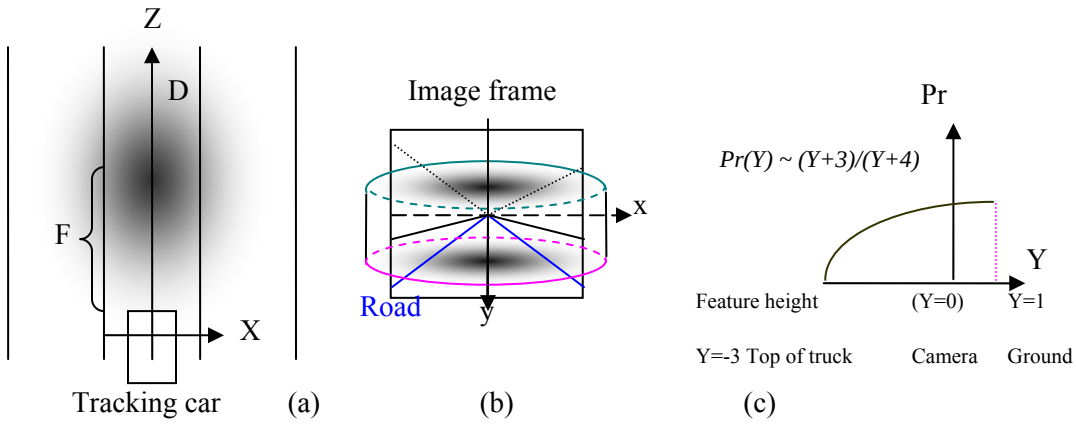


Figure 14 Probability density of target position and relative velocity from the camera is shown as the intensity. The darker the intensity, the higher the probability is. (a) Target vehicle distribution in top view, (b) Projection of the distribution onto image frame, (c) Vehicle feature distribution in height, which is from the ground to the top of highest vehicle in about 4m.

For a tracked car, we can also compute its probability of image behavior $p(x,v \mid C)$. According to Eq. 1, and 3 for a point on the vehicle,

$$p(x,v \mid C) = p(x,v \mid (X,Z,V_x,V_z) \in C) = p\left(x = \frac{fX}{Z}, v = \frac{fV_x - xV_z}{Z}\right) \qquad (15)$$

$$= \int_Z p(Z) \times p\left(x = \frac{fX}{Z}, v = \frac{fV_x - xV_z}{Z} \mid Z\right) dZ \qquad \text{Bayesian}$$

$$= \int_Z p(Z) \times p\left(X = \frac{xZ}{f}, V_z, V_x = \frac{Zv + xV_z}{f} \mid Z \right) \times \left| \frac{Z}{f} \right|^2 dZ \qquad \text{Eq. 9}$$

$$= \int_Z p(Z) \times p\left(X = \frac{xZ}{f} \mid Z \right) \times p\left(V_z, V_x = \frac{Zv + xV_z}{f} \mid Z \right) \times \left| \frac{Z}{f} \right|^2 dZ \qquad X, V_z, V_x \text{ independent}$$

$$= \int_Z p(Z) \times p\left(X = \frac{xZ}{f} \mid Z \right) \times \left\{ \int_{V_z} p(V_z) \times p\left(V_x = \frac{Zv + xV_z}{f} \mid V_z, Z \right) \times \left| \frac{Z}{f} \right|^2 dV_z \right\} dZ \qquad \text{Bayesian}$$

Filling in probability distribution of p(V_x), p(Z), and p(V_z), we can obtain p(x,v | C) as

$$p(x, v \mid C) = C_2 \int_Z \int_{V_z} e^{-\frac{(Z-F)^2}{2F^2}} \times e^{-\frac{\left(\frac{xZ}{f}\right)^2}{2D^2}} \times e^{-\frac{V_z^2}{2\sigma_z^2}} \times e^{-\frac{\left(\frac{Zv + xV_z}{f}\right)^2}{2\sigma_x^2}} \times \left| \frac{Z}{f} \right|^2 dV_z dZ$$

$$= C_2 \int_Z \int_{V_z} e^{-\frac{(Z-F)^2}{2F^2} - \frac{\left(\frac{xZ}{f}\right)^2}{2D^2} - \frac{V_z^2}{2\sigma_z^2} - \frac{\left(\frac{Zv + xV_z}{f}\right)^2}{2\sigma_x^2}} \times \left| \frac{Z}{f} \right|^2 dV_z dZ \qquad (16)$$

where C_2 is a constant for normalization, i.e., $\int_v \int_x p(x,v \mid C) dx dv = 1$. Figure 15 shows the probability distribution in intensity map and 3D plot of vehicles.

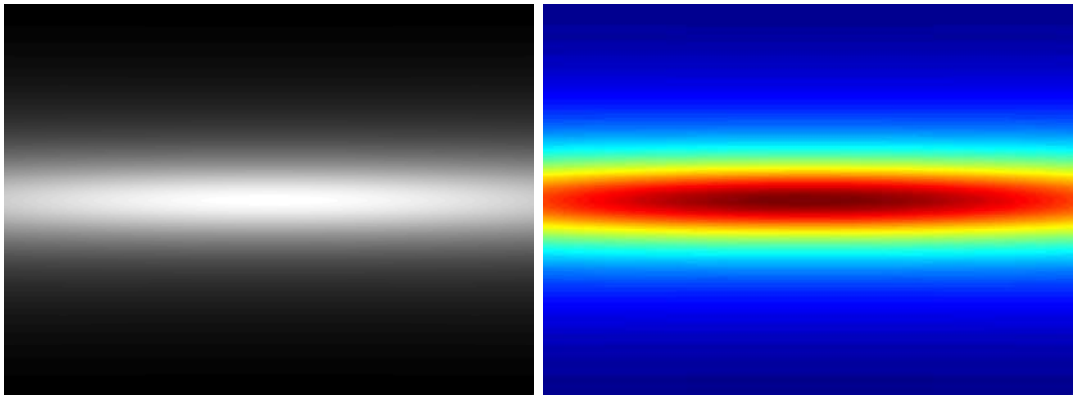


Figure 15 Probability distribution p(x,v) of vehicle features appearing at continuous space (x,v) in gray level and color plot

Estimating Weights for Image Profiling in Vehicle Detection

The vertical profiling of intensity and features has facilitated the target tracking. The set of weights used in the profiling explanation was left for later until the probability was introduced. Here we examine the profiling weight in the image frame such that the spatio-temporal representation obtained will best reflect the motion behaviors of targets. Using the same assumptions of probability distribution given in previous subsection, we will compute the probability of target feature at each image position, i.e., p(x,y | C), or simply p(x,y).

Because the mapping from *X, Y, Z* to *x, y* is not a one-to-one relation, the probability p(*x,y*) is computed by

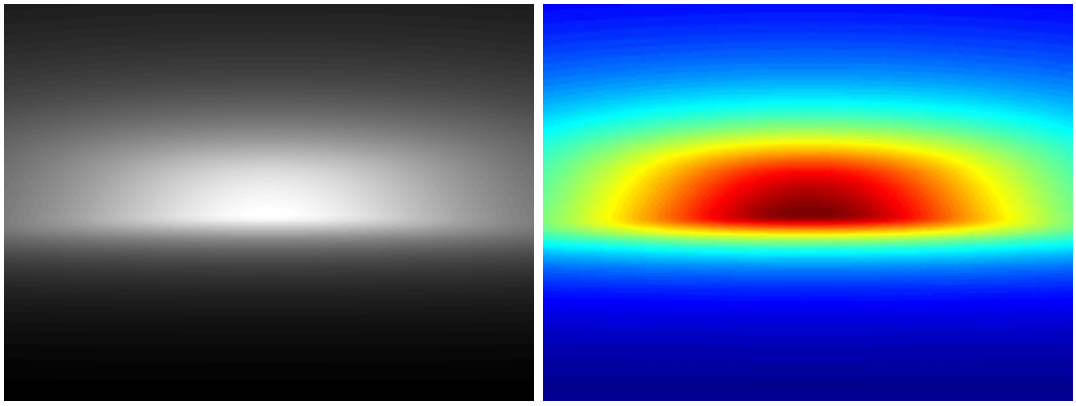
$$p(x,y\,|\,C)=\int_Z p(Z)\times p(x,y\,|\,Z)dZ \qquad \text{Bayesian}$$

$$=\int_Z p(Z)\times p\left(X=\frac{xZ}{f},Y=\frac{yZ}{f}\,|\,Z\right)\times\left|\frac{Z}{f}\right|^2 dZ \qquad \text{Eq. 9}$$

$$=\int_Z p(Z)\times p\left(X=\frac{xZ}{f}\,|\,Z\right)\times p\left(Y=\frac{yZ}{f}\,|\,Z\right)\times\left|\frac{Z}{f}\right|^2 dZ \qquad \text{Independent}$$

$$=C_3\int_Z e^{\frac{(Z-F)^2}{2F^2}}\times e^{-\frac{\left(\frac{xZ}{f}\right)^2}{2D^2}}\times\frac{\frac{yZ}{f}+3}{\frac{yZ}{f}+4}\times\left|\frac{Z}{f}\right|^2 dZ=C_3\int_Z e^{\frac{(Z-F)^2}{2F^2}}\times e^{-\frac{\left(\frac{xZ}{f}\right)^2}{2D^2}}\times\frac{yZ+3f}{yZ+4f}\times|Z|^2 dZ \qquad (17)$$

One result is displayed in Figure 14 and we use it for *w*(*x,y*) in Eq. 6 and 7 in profiling. Using this weight distribution from PDF of target features, we can enhance the extraction of vehicles, and ignore majority of irrelevant features from background.



w(x,y) color plot

Figure 16. The probability of vehicle features *w*(*x,y*) in the image frame for vertical profiling.

Computing Likelihood Using HMM during Tracking

After obtaining the forward probability initially in two tables, p(*x,v*|B) and p(*x,v*|C), we can now estimate the identities of features based on their motion behaviors. The likelihoods, i.e., probabilities of states under observation (*x(t),v(t)*), are denoted by $P(C_t\,|\,x(t),v(t))$ and $P(B_t\,|\,x(t),v(t))$ respectively, or $P(C_t)$ and $P(B_t)$ for short. At any time *t*, we should keep

$$P(C_t) + P(B_t) = 1 \qquad (18)$$

The probability of state transition from frame *t-1* to frame *t* is defined as

$$P(C_t \mid B_{t-1}) = 0.5 \qquad P(B_t \mid B_{t-1}) = 0.5 \tag{19}$$

$$P(C_t \mid C_{t-1}) = 0.8 \qquad P(B_t \mid C_{t-1}) = 0.2$$

The transition from car to car, i.e., $P(C_t \mid C_{t-1})=0.8$, emphasizes the continuity of car motion; as long as a car is captured, it will not be missed easily. A background may be identified as a car later, i.e., $P(C_t \mid B_{t-1}) = 0.5$, since there may not have strong clue to determine a car in the image region near the FOE, where both background and vehicle have small image velocity. When a trajectory is initially detected (t=0), its probabilities as a car and background are set empirically as $P(C_0) = 0.7$ and $P(B_0) = 0.3$ according to our delectability of vehicles using line and corner extraction in the profiled image area.

Using Viterbi algorithm in HMM, the probability of a trace to be assigned as car at time *t* is optimized as

$$P(C_t) = \max[\, P(B_{t-1})P(C_t \mid B_{t-1})p(x(t),v(t)|C_t),\ P(C_{t-1})P(C_t \mid C_{t-1})p(x(t),v(t) \mid C_t)] \tag{20}$$

And the probability as background is

$$P(B_t) = \max[\, P(B_{t-1})P(B_t \mid B_{t-1})p(x(t),v(t) \mid B_t),\ P(C_{t-1})P(B_t \mid C_{t-1})p(x(t),v(t) \mid B_t)] \tag{21}$$

If $P(C_t) > P(B_t)$, the trace is considered as a car at time *t*, and as background otherwise. The identity of a trace is formally output or displayed as the trace is tracked over a minimum duration of time. Otherwise, such a short trace is removed as noise; we assume that a target vehicle will not vanish from the field of view rapidly.

As we track all the traces in the profiles during the vehicle motion, we apply the HMM on each trace to update its state, i.e., car or background. At every moment, the obtained probabilities of a trace, $P(B \mid (x,v))$ and $P(C \mid (x,v))$, are normalized by

$$P(C_t) \leftarrow \frac{P(C_t)}{P(C_t)+P(B_t)} \qquad P(B_t) \leftarrow \frac{P(B_t)}{P(C_t)+P(B_t)} \tag{21}$$

in order to avoid a quick decreasing of $P(C_t)$ and $P(B_t)$ values to zero in Eq. 20 and 21, caused by multiplying a series of probability values less than 1. The processing is in real time as the vehicle moves.

If a new trace is found, it assembles a new HMM. The calculated identity of the trace may be uncertain at the beginning of the trace due to lack of evidence in the short period. The probability will get high as the trace is constantly tracked and updated. Because the probabilities of (x,v) have been computed and stored in lookup tables for car and background, the HMM takes no time in estimating the current status of traces at a new moment.

Critical event detection

The algorithms for detecting critical events all rely on patterns of motion and trends in the flow directions of objects in the scene. Each has its own characteristic motion and how it manifests itself in the video data as well as in the spatio-temporal reduced dimensional representation of the video contents.

Door open event detection

When the car door opens, along the edges it has an optical flow field that gets created. This optical flow field is to be detected after the pursued car has been stopped. The image edges are computed using an edge detection operator such as Sobel or Canny edge detection. Then along these detected edges the flow vectors are computed from frame to frame. The decision that the door of the pursued car has been opened is made when the total flow vector is greater than a certain magnitude. The information that the doors will be on the two sides of the car located in the video frame is used to filter out motion vectors that might exist elsewhere in the video frame due to other moving objects such as oncoming cars.

Detecting person running out of car

We use the reduced data representation given by Eq. (6) to detect a person running out of a stopped car ahead. Such an event is reflected in the 1D projection data as a trace moving toward the edges of the profile in an otherwise smooth and unchanging background (see Fig. 17).

- The distinguishable characteristics of this trace are:
- The trace is close to horizontal, leading away from the position of the vehicle.
- Unlike many other traces such as cars, it originates from the detected vehicle.
- The speed of a running person is limited when compared to other moving vehicles in the background scene.

The features represented in this 1D reduced temporal data are the high gradient points in the 1D data. We compute these high gradient points and track them over time using a simple Kalman filter. The Kalman model assumes a first order, constant acceleration model in 1D. The extracted traces are matched against the position of the vehicle obtained from previous steps. For each time instance t, we identify the candidate points in the 1D profile, and select the ones whose distance to the car boundary is less than a threshold, τ. The threshold is selected based on the average car door size. These selected points are labeled as candidates and their behavior is monitored using their movement vector calculated from $P_t - P_0$, where P_t denotes the location of the trace at time t and P_0 denoted the location where the trace started.

If a trace matches the characteristics of a running person, an alarm is raised and the location of the end point of the trace is indicated as the person running out of the vehicle as shown in Fig. 17.

When we look at the 1D accumulated profiles of the spatio-temporal representation, we can clearly see the trace of the anomaly that the running person creates. This can be seen in the figure below:

Figure 17: The video frame showing the person running out of the car (top image) and the corresponding 1D spatio-temporal representation (bottom image, indicated by the dots in the red ellipse).

Here, the person running out of the car is detected by the set of points in the spatio-temporal representation (bottom image) indicated by the red ellipse that form an almost horizontal long line. Detecting this line allows us to detect the person running in the original video frame indicated by the vertical yellow line in the top image showing where he is and raising a flag at the same time.

Detection of officer falling down

The detection of officer falling down depends on the detection and tracking of the officer in the video images. This is accomplished by using two pieces of knowledge during the video processing. One is the knowledge that the officer will enter the video frame either from the left or right edge of the image (because he is coming from the police car from which the camera is observing the scene). The second is that he will be moving against a relatively unchanging background. Taking advantage of these facts is done by (i) training the video on the background after the car stops and before the officer enters the scene

for a short period of time (30-50 frames or 1-2 seconds) and (ii) using this trained background to perform foreground/background separation using motion information. This allows us to relatively robustly detect moving objects which form a blob in the scene viewed by the camera on board the police car. A falling down event can be recognized by extracting the centroid of a blob and monitoring its movement for trends over time that are indications for falling down. We separated potential blobs that represent the police officer and other blobs in the background (e.g., cars, trees waving, etc.) by analyzing the point of entry of the blob into the field of view, i.e., from the left or right corner of the video frame. Blobs that do not meet these criteria are discarded as unrelated.

A tracking algorithm is initialized for each relevant blob; this tracking monitors the position of the blob at time $t+1$ related to its position at time t, and extracts the movement vector V_t. The trace of the blob centroid over time provides strong evidence for a falling down event with a sharp downwards action; the traces which belong to other moving objects not falling down manifest themselves as mostly linear lines with speed vectors without sudden changes.

We model the motion vectors in a *Falling Down* event with HMM to determine the state of the blobs as falling movement and normal movement. Two hidden states are assigned as Falling (F_i) and Normal Movement (N_i) to describe every candidate blob i. The observable features for each track are the movement gradient angle, a_i, and the magnitude, d_i, of the vector connecting the centroid of a blob in time t to $t+1$, which are both described in a continuous space rather than a discrete one. These two features create our movement vector $V_i(d(t), a(t))$ that represents the blob trace over time.

The likelihoods are denoted by $P(F_t|d(t), a(t))$ and $P\big(N_t\big|d(t), a(t)\big)$ respectively. As parameters of HMM, we keep $P(F_t) + P(N_t) = 1$ at any time *t*. The state transition probabilities from frame $t-1$ to frame t are defined as

$$P(F_t|N_{t-1}) = 0.2 \quad P(N_t|N_{t-1}) = 0.8$$
$$P(F_t|F_{t-1}) = 0.6 \quad P(N_t|F_{t-1}) = 0.4$$

The transition probability values were set empirically.

The Viterbi algorithm was used to estimate the probability of a fall given V_t. As we track each candidate blob, HMM is applied to the trace to update its state, i.e., Falling or Normal Movement. At every t, the obtained probabilities of a trace $P(F|d, a)$ and $P(N|d, a)$ are normalized in order to avoid a quick decrease of probabilities to zero.

The prior probabilities for the HMM model are based on the direction and magnitude of the motion vector in the image. We model this as a Gaussian density in which downwards motion vectors have a higher prior probability for *Falling* down and other directions have a higher probability for *Not Falling down*. The density is governed by the usual Gaussian density $G(x, y;\ x_0, y_0, \sigma_x^2, \sigma_y^2)$, in which (x_0, y_0) is where the Gaussian is centered and (σ_x^2, σ_y^2) are the variances in the *x* and *y* directions, respectively. We use one Gaussian G_F for Falling and one Gaussian G_N for non-falling. For any location (x_0, y_0), the Gaussian is within a window of $x \in [0,40]$ and $y \in [0,40]$ in pixels. We use $\big(\sigma_x, \sigma_y\big) = (25,15)$ for G_N, and $\big(\sigma_x, \sigma_y\big) = (7,15)$ for G_F.

The use of HMM helps to reduce the effects of noise in the data and results in detection of a *Fall* as evidence is accumulated. This process is in real time.

An example of this is shown in Figure 18 in which the officer entering the scene is detected by doing figure/ground separation based on classifying each pixel in the video frame as foreground or background using the knowledge gained as a result of training the algorithms on what is considered background. The red arrows in the figure show the local direction of the motion of the object. The white circle with the line shows the overall direction of the entire object in this particular video frame.

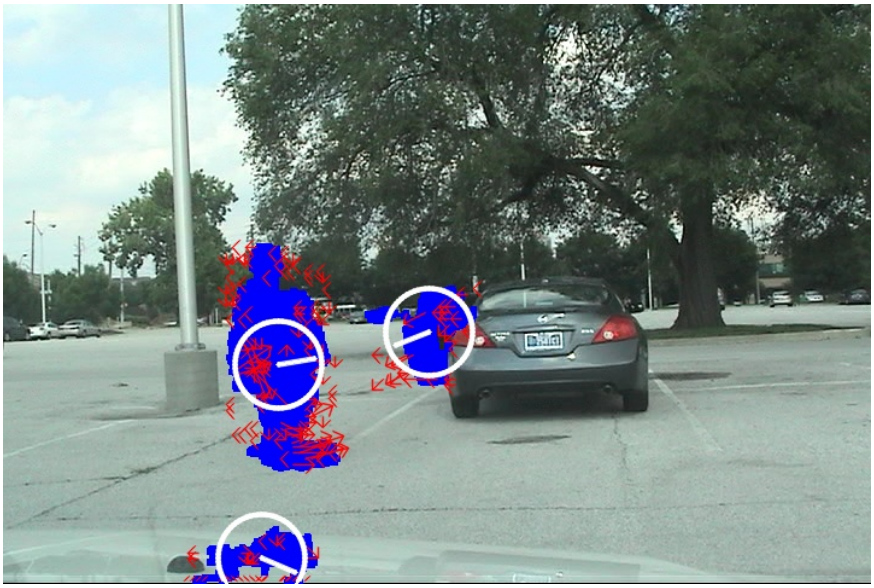


Figure 18: The foreground objects moving against the background are labeled blue. The circles with lines in them show the direction of movement of the objects. This figure shows two moving objects: the officer moving to the right and the door of the car opening to the left. The third object at the bottom is the reflection of the officer on the hood of the observer (police) car.

Figure 19 shows an example in which the sequence of blob centroids during a fall is labeled by an ellipse. This combined with the overall instantaneous object motion downwards (also shown in Figure 20) allows the “Officer Down” event to be detected.

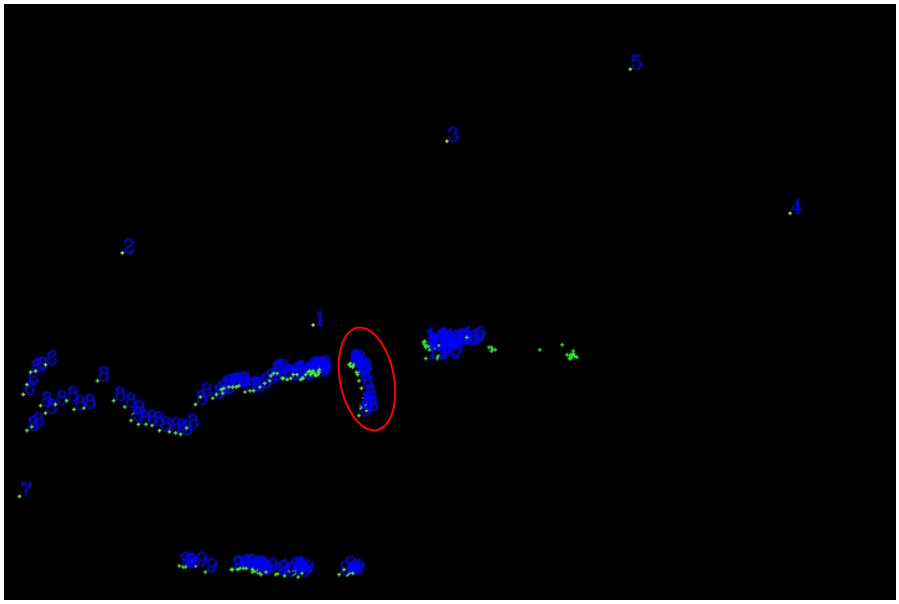


Figure 19: Each dot represents the centroid of the moving object detected in the video in one frame. The object centroids for each video frame (i.e., time instance) are superimposed to obtain this chart which indicates what the motions of the objects are. The collection of object centers moving downwards highlighted by the red ellipse in this figure corresponds to the officer falling down critical event.

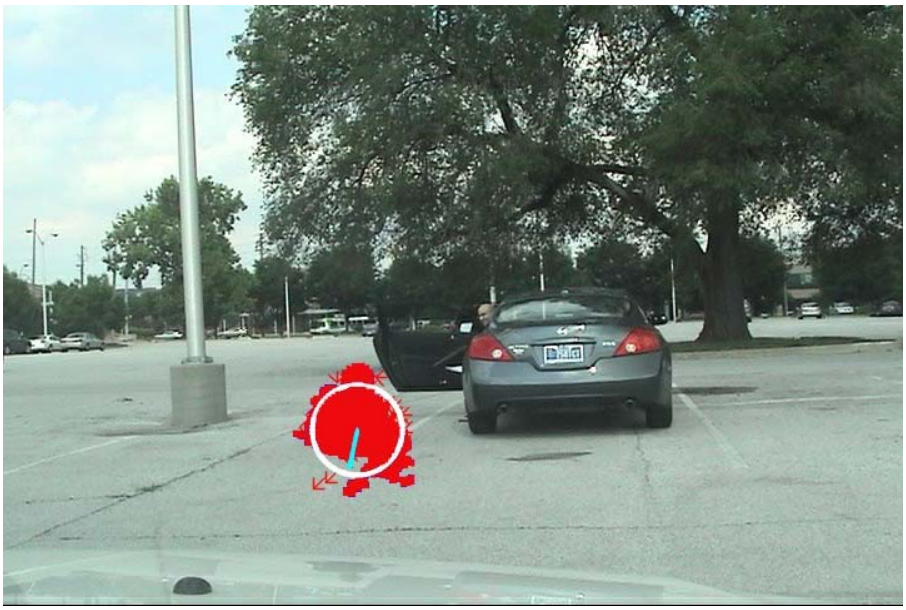


Figure 20: The red color indicates an "Officer down" event being detected.

Experiments and Discussion

Camera Setting, Feature Extraction and Tracking

Because we have introduced probability distribution of scenes to tolerate variations of targets, the precision of feature locations becomes less critical and sensitive to the results. We therefore omit serious

camera calibration by indicating the image position of forward direction, horizon, and the visible portion of self-car in advance for video analysis. These parameters are invariant to the dynamic scenes during vehicle motion.

We have examined our method on videos collected from cameras on board actual vehicles. The sample sets contain videos shot on rural and urban roads with various lighting conditions. The videos consist of both night-time and day-time clips. Table 1 shows examples of captured in-car video we frequently experimented. Some of them were taken with a video camera on board a police car. We implemented our method in Visual C++ environment using the OpenCV software [1] on AVI video format. The forward probability maps were computed offline using MatLab.

Table 1. Part of video sequences for experiment and evaluation

Source	Illumination	Environment	Duration (h:m:s)	Number of Frames
Police Video	Day	Urban	11:07	19990
Police Video	Day	Urban	11:26	20559
Police Video	Dusk	Rural	03:44	6713
Police Video	Night	Rural	09:48	17622
Police Video	Night	Urban	06:38	11928
Experiment Video	Night	Urban	20:09	36251
Experiment Video	Day	Urban	12:23	22293
Experiment Video	Dusk	Urban	02:45	3959
Total			01:18:00	139315

The thresholds used in this work are usually low to allow more candidates to be picked up in the processing; we avoid abrupt cutting off of features and traces in feature detection and tracking. The classification and identification of them to real targets and noises is handled later by HMM with the probability. This will solve the problem of threshold selection that affects the results drastically.

The length of a horizontal line segment is not very important, since we are counting numbers of line segments at each image location x and then find peaks after smoothing; a long line may contribute to everywhere and will not influence the peak locally.

In feature tracking using vertically condensed profiles, we need target vehicles to be present in a certain number of frames before we can conduct a successful identification. This number is calculated empirically and depends on the characteristics of targeted objects. In our tests we defined the minimum tracking duration for a targeted vehicle as 50 frames (which corresponds roughly to 1 second of video). We remove tracking history after a trace is lost for at least 20 frames, and the minimum strength of the trace for thresholding purposes is set to the top 67 percentile; this will remove the noise and weak traces.

Probability Handling

We do not use any real data to train parameters in HMM, rather we define the basic probability distribution according to physical environment and derive the probabilities in images and video, because collecting data for training or learning in this continuous observation is not as efficient and complete as deducing theoretical distributions. Although we can compute ideal target motion from the vehicle ego-motion obtained from vehicle sensor, the results would be affected by noise factors such as vehicle

shaking, slant road, failure of feature detection, etc. For these reasons, our assumptions of background and vehicle are reasonably general and they cover a large scope of variations. The assumption on target position and motion even includes mild turning when the vehicles move on a curved path. The vertical profiling reduces the influence of vehicle roll and pitch on slanted road.

We have used parameters in Table 2 for the forward probability computation. We note that the resulting forward probabilities of the two events (background and car) have major differences in their distributions (Figures 11 and 15) as look-up tables, though some parts at where |x| is small are mutually overlapped. This means the decision to classify an object as background or car made with the current observation that falls in these areas may have low certainty levels, which has to be disambiguated after a period of time as their motion demonstrate sufficient differences.

Table 2 Parameter selection in probability computation

	Parameters	Physical Environment and Condition	Value
D	Average road width	Considering as wide as three lanes on each way	6m
F	Distance to target	Minimum safe distance	10m
σ_F	Variance of target distance		20m
σ_x	Variation of horizontal velocity V_x of target vehicle	Maximum speed in cutting tree lanes, the speed to tolerant for a horizontal camera speed on a curved path, small and sudden steering of the self vehicle	??m per sec
σ_z	Variation of V_z	Constant if target is pursued	??m per sec
V	Absolute speed of observer vehicle	Ranging from 30~70 km	50 km
S σ	Averaging pursuing speed and variance of observer vehicle	50km 10km	15m per sec 5m per sec
f	Camera focal length	Through a simple offline calibration	900 pixel
$\int_z$	Range for integration	From camera position to infinity	0~200m
$\int_x$	Range for integration	Wider than a road to include all backgrounds possibly seen in video frame (wide scene range is included at distant depth)	-50~50m
H	The maximum height of vehicle	As high as a truck, but mostly for cars	4m
σ_r	Variance of steering angle of R_y	Can be computed from the maximum tuning radius of a vehicle and road curvature. Here a simple value is set	10 degree, calculated in radian

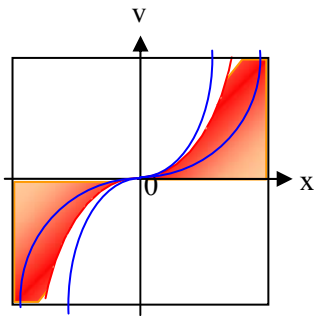


Figure 21 Traces and motion properties of opposite vehicles bounded by blue curves.

The oncoming vehicles traveling on the opposite lanes are treated as background. In addition to their positions that are beyond the boundaries of the lanes of the observer vehicle, their relative velocities, denoted as V_{op}, are faster than that of the background ($-V$). The traces of opposite vehicles will be even more slanted than background traces and, in Figure 10, these traces will be located close to the v axis in a strongly curved way as illustrated in Figure 21. Their probability of appearance is not overlapped with that of tracked vehicles in $p(x,v)$, except the area where $|x|$ is small. Such a special case means the oncoming vehicle is far away and it has a small size in the image and little influence on safety tracking of target vehicles.

The probability and parameters defined in this paper are widely distributed to include all possible cases. If the motion of the observer vehicle could be read from its mechanical and electric system, the PDF of many parameters would be narrowed down to proper ranges as the vehicle moves forward dynamically and the probability maps will be precise for tracking and identifying target vehicles. Figure 21 shows an example for trace identification, from uncertain trace to a certain one; further tracking in consecutive frames becomes confident.

Figure 22 gives examples of vehicle detection and tracking results. As it can be seen in the images, the program can successfully track the vehicles in very complex backgrounds and is invariant to lighting condition. In Figure 22f, the tracking program locates two overlapping boxes over a single car. Because of the close proximity of the camera to the targeted vehicle, the front light of the camera car has created disconnected horizontal line segments; and these lines create two separate traces. In Figure 22b, presences of multiple distant cars and their shadows have created multiple horizontal line segments that seem to move together; the line segmenting algorithm recognizes them as a single vehicle. However, this effect is transient. As the vehicles separate their movements, the mixture of horizontal line segments has dispatched and thus the algorithm separates them as one vehicle. Moreover, the shadow with a vehicle body also creates edges in the video frames and results in a weak trace than the vehicle trace. It is usually very difficult to distinguish these two traces, and we just consider a shadow to be a part of vehicle.

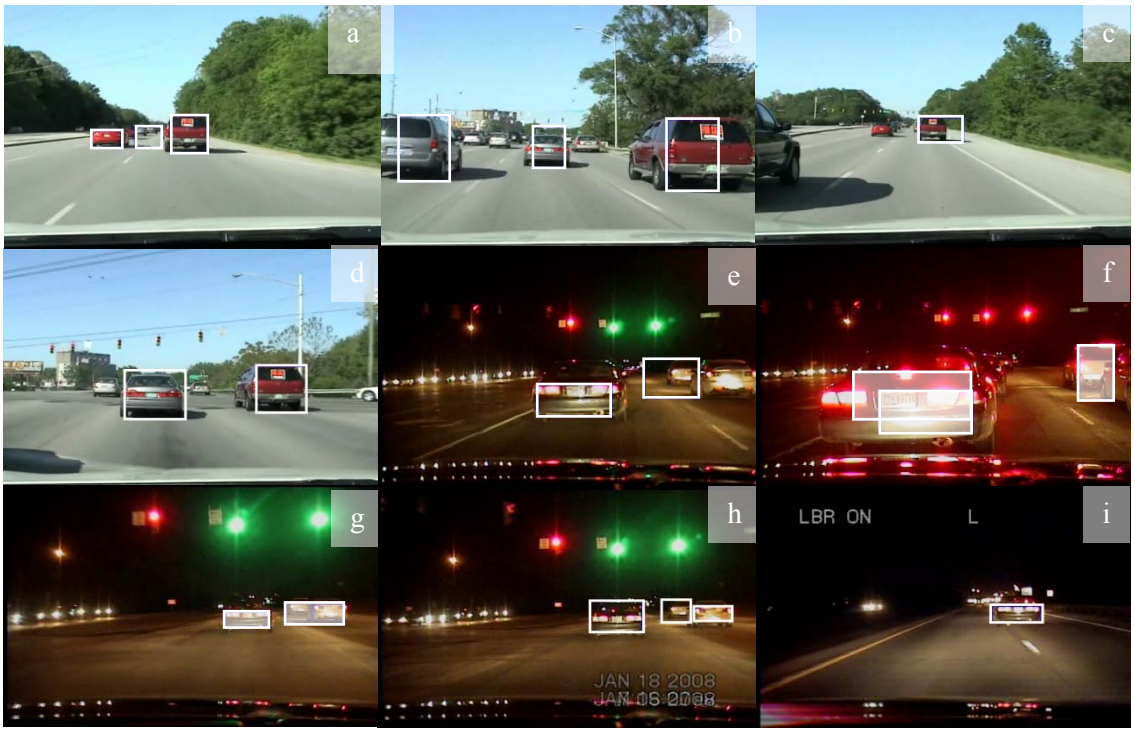


Figure 22 Vehicle detection and tracking results

We found that our method has 89.7% success rate in detecting vehicles. In most cases of correct detection, the duration of the tracking lasts as long as the vehicle remains in the scene. If a detected vehicle moves too far from the observer car (police car), the tracking is stopped. There are instances where the method classifies a non-vehicle region in the video as a vehicle, but the duration of this identification is short and eventually gets corrected. Experimental results show that our method has a success rate of 83% based on our data in the case of falling down detection. Although the false-positive rate for fall detection is high, it is necessary for the police safety. In our design of the system, the officers are equipped with radio worn on their uniforms that they can use to inform the headquarters that it is a false alarm. A high true positive rate, however, is important, especially when the officer is incapacitated for the system to perform as intended.

Fig. 23 shows an example of Running Person detection, whose position is indicated by a vertical red arrow. Fig. 24 shows an example sequence of Fall detection. In Fig. 24(a), a person in the foreground, detected by motion segmentation, is depicted by a blue blobs, and in the last frame of the sequence, the falling down event is indicated by the red blob. Fig. 24(b) shows the probability values computed over time (horizontal axis) for the Falling down event and the normal motion events.

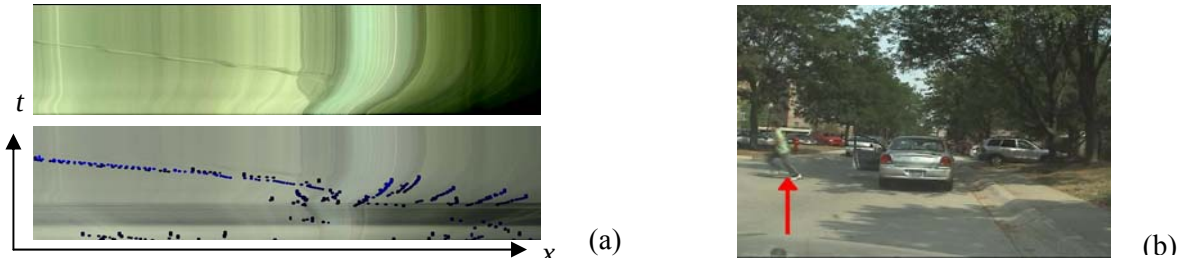


Fig. 23: 1D projections vs. time representation: (a) Top: projection data, bottom: detected running person candidates (blue dots). (b) Final detection result of person running out of car (red arrow)

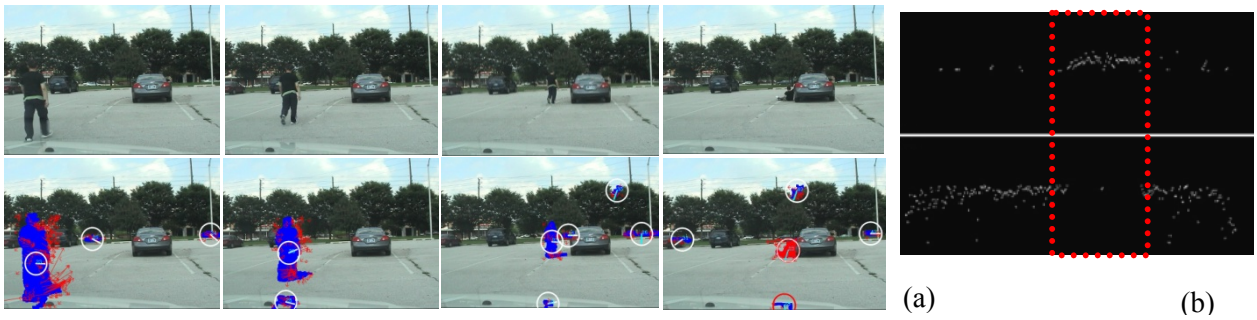


Fig. 24: Detecting falling down event. (a) Top row: selected frames from the original video, bottom row: Candidate blobs (blue) for the corresponding frames; in the last frame, the red blob indicates falling down event. Circles and inside arrows indicate the motion vectors of the blobs. (b) The probabilities for Falling (above white line) and Normal motion (below white line) along time (the x-axis). The red rectangle indicates falling down event.

Table 3 shows the summary of the results of our proposed falling down detection method in the test video clips. In order to calculate the success rate of the method a human observer inspected the video sequence and counted the correct and incorrect *Fall* detection results.

Table 3 Parameter selection in probability computation

		Software Performance	
		F	N
Ground Truth	F	83%	17%
	N	43.75%	56.25%

Processing Time

The computation of HMM is fast and implemented in real time, by referring to look up tables of forward probability calculated in advance. The modules for corner detection and Gaussian use the hardware enhanced Open-CV package. Other processing such as profiling, tracking, high-light detection, and horizontal line segment extraction are also programmed to reach its optimal performance.

Video input along with differentiation for corners and smoothing for peaks can reach a speed of 30 fps with a PC (CPU2.4GHz, Memory 4.0GB). Horizontal line segment extraction can be done at 2Hz. Profiling of intensity and features takes 30ms. After 1D profiles are extracted consecutively, smoothing, tracking, and HMM applied to it takes 0.5ms, since only 1D data are involved in computation.

In the implementation of HMM, an increase of the minimum number of frame for identification results in a better accuracy. The longer we track an object, the higher the probability of a correct identification is. Such a time for identification depends on how significantly a trace shows its motion behavior, which is further related to the observer vehicle (with camera) speed V and how wide the environment or background is surrounding the vehicle. Although the HMM provides probability from very beginning as the trace is initially detected, it will not be very certain if the time for a trace to display its motion tendency is insufficient. Even if the observer vehicle moves fast (i.e., *V* is high), wide scenes (|X| is large) may still be uncertain for a while, since their image motion is slow according to Eq. 8. Fortunately, such a case will not affect the safety of observer vehicle. Close objects and narrow environments (|X| is small) usually show fast image velocity and short appearance duration during the observer vehicle motion. Such background can be identified quickly.

Conclusion

In this project we accomplished designing, implementing, and testing of the algorithms and software to add intelligence to an in-car video system in police cars. Of the original specific goals stated, we have accomplished following, which were technically the most difficult and challenging ones:

1. Build a computer system to analyze video and identify certain undesirable events. The system was developed as a software prototype on a laptop computer with a digital video recording device connected to the laptop that could be mounted in the passenger front seat of a car for data collection and testing purposes.
2. Deploy the test system in police cars and in simulations in order to capture realistic video recordings of the intended subject matter. Data was collected from real police car videos as well as from simulated situations which were then used to test the developed algorithms.
3. Test the system's ability to determine undesirable situations. The situations included open door of the stopped car, person running out of the stopped car, and officer falling down event. Algorithms were developed and tested for detecting all these events successfully.
4. The results were disseminated with the proper acknowledgement of the NIJ funding in scientific venues. The commercialization of the system is open for anyone interested in it.

Because of the complications found in feature extraction over the very wide range of viewing conditions, we did not have time to implement and test the following specific goals in the original project proposal. We believe the system integration functions remaining are more routine tasks at this point, given the algorithmic work successfully completed.

- Designing and building a video recording system capable of recording both standard resolution and high definition video. The most difficult part of accomplishing such a system has been implemented and tested, namely the detection of the stopping of the target vehicle in the front and detection of

critical events. The ability to actually start recording the HD video at these critical times needs to be integrated into the prototype system.

- Integrating the GPS and ALPR into the smart video system has not been done. The integration of this capability was also highly dependent on the successful development of critical event detection algorithms, and, therefore, we did not have time to do the more routine system integration tasks.

Finally, we have a number of papers published on this work and a journal paper that is currently under peer review. All these publications have or will acknowledge the NIJ grant at the time of the publication.

Papers already published or accepted for publication:

1. Amirali Jazayeri, Hongyuan Cai, Jiang Yu Zheng, Mihran Tuceryan, and Herbert Blitzer, "An intelligent video system for vehicle localization and tracking in police cars," SAC '09: Proceedings of the 2009 ACM symposium on Applied Computing, pp. 939-940, Honolulu, Hawaii, 2009.
2. Amirali Jazayeri, Hongyuan Cai, Jiang Yu Zheng, Mihran Tuceryan, "Identifying Vehicles in In-Car Video Based on Motion Model," IEEE Intelligent Vehicles Symposium (IV 2010), San Diego, June 2010. Presented, will be published in the Proceedings.
3. Amirali Jazayeri, Hongyuan Cai, Mihran Tuceryan, Jiang Yu Zheng, "Smart Video Systems in Police Cars," ACM Multemedia 2010 Conference, Firenze, Italy, October 2010. Accepted; to be presented and published in October 2010.

Papers currently under peer review:

1. Amirali Jazayeri, Hongyuan Cai, Jiang Yu Zheng, Mihran Tuceryan, "Vehicle Detection and Tracking in In-Car Video Based on Their Motion," submitted to the IEEE Transactions on Intelligent Transportation Systems.

References

[1] OpenCV Open Source Library. http://sourceforge.net/projects/opencvlibrary/.

[2] D. Alonso, L. Salgado, and M. Nieto. Robust Vehicle Detection Through Multidimensional Classification for on Board Video Based Systems. volume 4, pages IV – 321–IV – 324, 2007.

[3] Margrit Betke and Huan Nguyen. Highway Scene Analysis from a Moving Vehicle under Reduced Visibility Conditions. In *Proc. of the International Conference on Intelligent Vehicles, IEEE Industrial Electronics Society, Stuttgart, Germany*, pages 131–136, 1998.

[4] M.-P. Dubuisson Jolly, S. Lakshmanan, and A. K. Jain. Vehicle segmentation and classification using deformable templates. 18(3):293–308, 1996.

[5] Lei Gao, Chao Li, Ting Fang, and Zhang Xiong. Vehicle Detection Based on Color and Edge Information. In *Image Analysis and Recognition*, volume 5112/2008 of *Lecture Notes in Computer Science (LNCS)*, pages 142–150. 2008.

[6] C. Harris and M. Stephens. A combined corner and edge detector. In *4th Alvey vision conference*, volume 15, page 50. Manchester, UK, 1988.

[7] C. Hoffman, T. Dang, and C. Stiller. Vehicle detection fusing 2D visual features. In *Intelligent Vehicles Symposium, 2004 IEEE*, pages 280 – 285, 14-17 2004.

[8] Chu Jiangwei, Ji Lisheng, Guo Lie, Libibing, and Wang Rongben. Study on method of detecting preceding vehicle based on monocular camera. In *Intelligent Vehicles Symposium, 2004 IEEE*, pages 750 – 755, 14-17 2004.

[9] D. Koller, G. Klinker, E. Rose, D. Breen, R. Whitaker, and M. Tuceryan. Real-time Vision-Based Camera Tracking for Augmented Reality Applications. Lausanne, Switzerland, 1997.

[10] D.G. Lowe. Object recognition from local scale-invariant features. In *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*, volume 2, pages 1150 –1157 vol.2, 1999.

[11] John Oliensis. A Critique of Structure-from-Motion Algorithms. *Computer Vision and Image Understanding*, 80(2):172 – 214, 2000.

[12] P. Parodi and G. Piccioli. A feature-based recognition scheme for traffic scenes. In *Proc. Intelligent Vehicles '95 Symp.*, pages 229–234, 1995.

[13] H. Schneiderman and T. Kanade. A statistical method for 3D object detection applied to faces and cars. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, volume 1, pages 746–751, 2000.

[14] D. Stirzaker. *Elementary probability*. Cambridge Univ Pr, 2003.

[15] G. D. Sullivan, K. D. Baker, A. D. Worrall, C. I. Attwood, and P. M. Remagnino. Model-based vehicle detection and classification using orthographic approximations. *Image and Vision Computing*, 15(8):649 – 654, 1997. British Machine Vision Conference.

[16] H. Takizawa, K. Yamada, and T. Ito. Vehicles detection using sensor fusion. In *Intelligent Vehicles Symposium, 2004 IEEE*, pages 238 – 243, 14-17 2004.

[17] Tao Zhao and R. Nevatia. Tracking multiple humans in complex situations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(9):1208, 2004.

[18] Tao Zhao and R. Nevatia. Tracking multiple humans in crowded environment. volume 2, pages 406–413, 2004.