Holistic Modeling and Tracking of Road Scenes

John J. Sprouse
jsprouse@ri.cmu.edu

October 2nd, 2006

Robotics Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213

Abstract

This thesis proposal addresses the problem of road scene understanding for driver warning systems in intelligent vehicles, which require a model of cars, pedestrians, the lane structure of the road, and any static obstacles on it in order to accurately predict possible dangerous situations. Previous work on using computer vision in intelligent vehicle applications stops short of holistic modeling of the entire road scene. In particular, no lane tracking systems exists which detect and track multiple lanes or integrate lane tracking with tracking of cars, pedestrians, and other relevant objects. In this thesis, we focus on the goal of holistic road scene understanding, and we propose contributions in three areas: (1) the low-level detection of road scene elements such as tarmac and painted stripes; (2) modeling and tracking of complex lane structures, and (3) the integration of lane structure tracking with car and pedestrian tracking.

Contents

Figure 1: A road scene in which the danger associated with the vehicles present depends greatly on the lane structure of the roadway. The vehicle to the right is dangerous in this system because it's lane of travel merges with the ego lane. The oncoming sedan is not dangerous even though it is traveling directly towards the ego-vehicle, because it's lane of travel curves away from that path.

1 Introduction

This thesis proposal is concerned with the development of prediction and warning systems for intelligent vehicles, such as that studied by Broadhurst *et al*. in [5]. Given a model of the present road scene relative to the vehicle the system is mounted in (the *ego-vehicle*), the goal is to predicts likely future states and the danger associated with each, so that the driver can be alerted. The input to such a system must include, at a minimum, descriptions of the state of the ego-vehicle, the location of other vehicles, pedestrians, and static obstacles, and a rich model of the roadway: the extent of the drivable surface, its division into lanes, directions of travel, and semantic tags such as "merge lane," "off-ramp," or "parking lane."

In the past, many sensors have been employed toward detecting and tracking these semantic objects, such as sonar, LIDAR, and light-striping. The use of cameras is attractive as they are relatively inexpensive and easy to install, and there is a rich history of computer vision in domains such as lane keeping and departure warning, driver performance monitoring, collision warning, and automatic cruise control, but to date no systems have taken a holistic approach to road scene understanding. Yet such an approach is necessary for driver assistance systems to be useful in practice.

Consider the example of Figure 1, in which we see a vehicle in an adjacent lane. Since it is not in the immediate path of the ego-vehicle, it would be disregarded by a simple collision detection system. However, if that system takes into account a complete model of the roadway, it would use the fact that the adjacent lane is merging to predict a possible collision. In contrast, the oncoming vehicle in this scene appears to be on a possible collision course, if one only considers its position and velocity. Yet, knowledge of the shape of the lane in which it is traveling makes this collision much less likely.

This thesis will investigate approaches to detection and 3D tracking of semantic objects relevant to driver warning systems using cameras mounted in a moving vehicle[1]. Specifically, we will be concerned with three

main questions:

- How shall we *model* the lane structure of the roadway?

  Previous work has mainly focused on models which describe the *ego-lane* (the lane the ego-vehicle is traveling in) only. A few papers model adjacent lanes, but no work to date has attempted to develop a model capable of describing more complex scenes or assign semantics to each lane, nor to incorporate rules which govern the shapes and relative positions of lanes on a roadway.

- How shall we *track* such a model over time?

  Ego-lane tracking has been accomplished successfully using a variety of methods including Kalman filters [13, 26, 29], particle filters [41, 2], or ad-hoc methods as appropriate to the task at hand. In considering a richer model of lane structure, we must infer not only the shape of each lane, but how many are present. As such, the literature on multiple target tracking (MTT) becomes relevant.

- How can we *exploit* our knowledge of other objects?

  Algorithms for detecting and tracking other vehicles, pedestrians, and static obstacles may operate well in isolation, but there exists rich contextual relationships between their locations and the model of the lane structure. Exploiting these relationships should allow the various detection and tracking modules to mutually assist each other.

We first concentrate on pixel-level detection of road scene elements such as stripes and tarmac. We then discuss the modeling and tracking of complex lane structures. Finally, we ask the question of how this output and the output of trackers for cars and pedestrians might mutually inform each other.

1.1 Low-level cues for lane tracking

We observe that on most modern high-speed roadways, lanes are bounded by painted stripes, explicit detection of which is an important cue in lane tracking. In section 2.1, we present our previous work on detection of stripe-like features in images. Our work builds upon previous work on detecting roadways in aerial images [42].

A few systems before ours have been designed to extract stripes from images, for example [4, 41]. Other systems have operated on myriad different features. The ALVINN and RALPH systems developed at CMU [35, 34] operate directly on image intensity values. Other systems use image gradients [26, 28, 17], detected edges [4, 24], and box filters [13, 2].

We also observe that stripes become increasingly difficult to track the further they are from the ego-lane. However, one can infer the presence of the lane by the extent of the tarmac. With the goal of using such cues in lane tracking, we believe it is important to model the appearance of tarmac, background, and other relevant image surfaces. We discuss the classification of road scene images in section 2.2.

1.2 Modeling and tracking lane structures

Previous works which explicitly model the lane structure of the roadway have focused on the *ego-lane*. They can be categorized by (a) whether they use a 2D model of curves in the image, or a 3D model of curves on

[1]The algorithms discussed in this work were tested on data collected from our test vehicle, a Lexus LS-430 generously provided by DENSO corporation and shown in Figure 1b. We have mounted a Sony XC-555 CCD camera for video collection, and this data is synchronized via a time-stamping system to odometry data including the velocity and steering angle of the vehicle. Before each test run, the camera is calibrated using a checkerboard array so that we know both its intrinsic parameters (focal length, principal point, and skew) and its position and orientation relative to a coordinate system on the tarmac below the midpoint of the rear axle.

the ground plane, and (b) the type of curve used in modeling. The most common types of curves are straight lines [4], parabolas [13, 26], circular arcs [24, 28], and splines [44].

Recently, with the advent of modern in-car geographical information systems (GIS), attempts have been made to match the video data to a given street map. One example is the DARVIN system [17], in which a map in the form of a straight-line network is used in combination with a vision and a GPS sensor. Similarly, the LARA system [21] matches a detailed manually constructed map to image features. Commercial road maps are becoming more sophisticated, to the degree that they include spline curves, and information about the number of lanes on a road. However, they are still not detailed enough to explicitly match to road scene features.

When the lane structure of the roadway is unknown, we must estimate the number of lanes as well as track each of them. Traditionally there are two approaches to tracking a varying number of targets. *Data association* trackers probabilistically assign image measurements to potential targets, and then track each target independently, conditional on these measurements. We present our previous work on multiple target tracking of lane boundaries (painted stripes) in section 3.2. Unfortunately, independent tracking of these lane boundaries makes it difficult to incorporate a model of the global structure of the roadway. Only by tracking them jointly can we integrate contextual constraints between lane boundaries. *Non-parametric* trackers, such as particle filters, are able to track multiple targets as separate modes in the state distribution. We have developed a particle filter-based tracker of a single lane which incorporates a flexible shape model with simple constraints on the boundaries; it is discussed in section 3.3.

Moving forward, we believe that a model of complex lane structures is needed if a prediction and warning system is to become practical. We discuss such a model and propose a stochastic algorithm for tracking it in section 4.1.

1.3 Contextual relationships in road scenes

Finally, we discuss the integration of other relevant semantic objects into our system. The simplest way to achieve this would be to aggregate our lane structure tracking system with the output of detectors and trackers designed for the objects of interest. However, we note that these trackers mutually constrain each other in many ways. For example, the viewpoint of the camera and the 3D shape of the ground in front of the vehicle mutually constrains the 2D shape of the lanes and the 2D location and sizes cars in the image. Hoiem *et al*. discuss the propagation of mutual constraints between the viewpoint and object locations in 2D in [19], assuming a flat ground plane; in this thesis, our goal will be to integrate the tracking of lane structures into such a system. We discuss this goal in section 4.2.

1.4 Thesis statement

In this thesis, we will investigate the modeling of complex lane structures, the tracking of such models over time, and the mutual constraints between the lane structure model and the tracking of other relevant road scene objects such as cars and pedestrians, with the goal of enabling practical prediction and driver warning systems.

2 Lane Feature Extraction

In real-world road scenes, there are a variety of cues which divide the drivable surface into lanes. In some circumstances, lane boundaries are implicit and it is expected that the driver will intuit them based on contextual information such as road boundaries and their expected width. Usually there are explicit markings present. On most modern roadways, we can expect to find painted stripes, which not only demarcate lane boundaries but also add semantic information such as whether a lane boundary is meant to be crossed (dashed versus solid lane lines), and whether the adjacent lanes are meant for opposite directions of travel (yellow versus white lane lines). We can also expect to find tarmac within the boundary of the lane. While noting that there are many other cues which should be used in a complete lane tracking system (see [2] for an example which fuses multiple cues), we turn our focus to the detection of painted stripes and tarmac.

2.1 Stripe detection

Detection of stripes in an image can be viewed as the simultaneous detection of opposing step edges separated by a given width. Canny [8] explored the use of matched filters for this problem, but noted a step edge will produce the same response as a stripe profile of half its magnitude. This effect is demonstrated in Figure 2, and is known to be a property of all linear filters [25]. In [39] the authors note that these responses due to step edges are predictable, and they explicitly check for their presence using multiple filter widths at each possible orientation. However, [25] eliminates the need for post-processing using a nonlinear combination of steerable filters. [42] takes a similar approach but accounts for the bias introduced when the image magnitude is not the same on either side of the stripe. We use a technique similar to those presented in [25, 42]: we first detect the orientation of potential stripes at each pixel in the image, and then we check for the presence of oriented stripe edges at the appropriate distance along the normal direction. Our approach differs in that we detect stripe edges using a narrow bandwidth filter, which biases our technique toward "box-like" profiles.

For a given stripe width w, we start by looking for the orientation where the second derivative of the image magnitude is maximal. This can be found by taking the eigenvector $\vec{e}_1$ corresponding to the largest eigenvalue of the Hessian matrix

$$H(x,y) = \begin{pmatrix} I_{xx} & I_{xy} \\ I_{xy} & I_{yy} \end{pmatrix} ,$$

where I_{xx}, I_{yy}, and I_{xy} are the results of filtering the image by second derivative of Gaussian filters G^{σ}_{xx}, G^{σ}_{yy}, and G^{σ}_{xy} respectively. It can be shown [8] that for a stripe of a given orientation and width, the response will take on its maximum value at $\sigma = \sqrt{3}w/6$.

Given the normal direction $\vec{e}_1$ at each point $\vec{x}$, we can anticipate the location of the stripe edges as

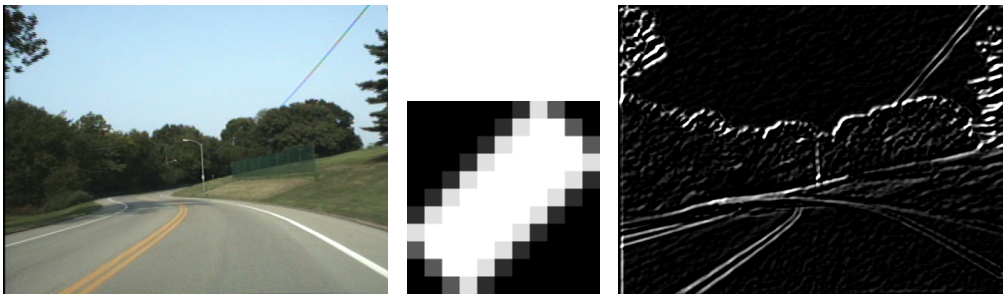
$$\begin{aligned} \vec{x}_l &= \vec{x} - (w/2)\vec{e}_1 , \quad \text{and} \\ \vec{x}_r &= \vec{x} + (w/2)\vec{e}_1 . \end{aligned}$$

We then verify the presence of the stripe by checking the response of a narrow edge detector oriented in the direction of $\vec{e}_1$:

$$R(\vec{x}) = \min([I_x(\vec{x}_l)\ I_y(\vec{x}_l)]\vec{e}_1, -[I_x(\vec{x}_r)\ I_y(\vec{x}_r)]\vec{e}_1) ,$$

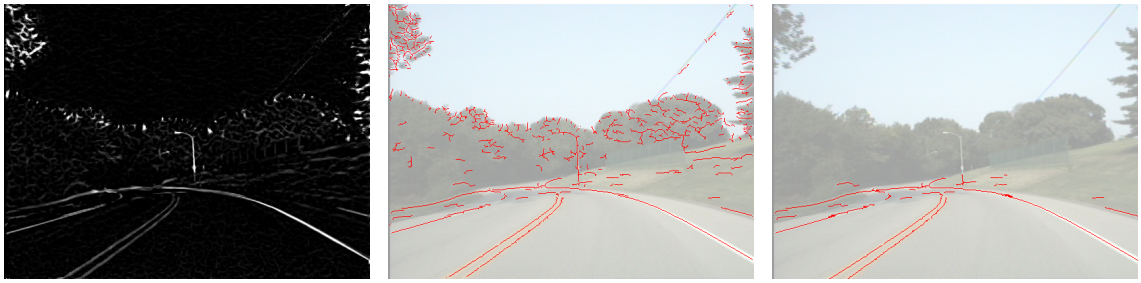
where I_x and I_y are the responses from filtering the image with steerable derivative of Gaussian filters $G^{\hat{\sigma}}_x$ and $G^{\hat{\sigma}}_y$. We set $\hat{\sigma}$ to be narrow enough to match the step edge due to the stripe boundaries in the image; a value of 1.0 usually gives good results. As shown in Figure 3(a), this technique suppresses responses due to unpaired step edges while keeping responses due to stripes of the requested width.

At each pixel location, we record the width and orientation which produced the maximum response. We then perform non-maximal suppression along the recorded orientation in order to suppress non-peaks in the



(a) (b) (c)

Figure 2: Stripe detection with a matched filter. The input image (a) is filtered with the matched filter of width 5 (b) and the result is shown in (c). Note the strong response due to the step edge at the top of the trees.



(a) (b) (c)

Figure 3: (a) Result of filtering using our method. Note that step edges are ignored. (b) The result after non-maximal suppression and hysteresis thresholding. (c) The result after post-processing.

response profile. Classification of the image into binary "stripe" versus "non-stripe" pixels is done using adaptive thresholding. An example of the result at this stage is shown in Figure 3(b).

Once we have classified each image pixel, we can further eliminate responses that are not due to lane-lines. We first eliminate responses that fall above the horizon line (using the assumption of a flat ground plane). Then, we group the remaining responses into connected components by repeatedly choosing an ungrouped pixel and recursively adding adjacent pixels. Next, straight line segments are fit to the grouped components, such that no pixel is greater than five pixels away from the line. Finally, the length of each segment is measured by projecting their endpoints of these segments are projected onto the 3D ground plane. On the assumption that the length of painted stripes and dashes are greater than 2 meters on the ground plane, any pixels associated with segments less than that length are discarded. The final result after post-processing is shown in Figure 3(b).

Better filtering and further post-processing is possible, given the availability of additional cues. In the future, we intend to incorporate such knowledge in specific ways:

- The output of a tarmac detector (see the next section) allows us to eliminate responses which are not near the road surface. In addition, knowledge of objects such as cars, pedestrians, and static obstacles allows us to eliminate measurements due to occlusion.

- Knowledge of predicted lane line locations allows us to lower our classification thresholds in those image regions in order to localize faint stripes.

- A prior model of the color of lane lines allows us to eliminate unlikely stripes.

2.2 Tarmac classification

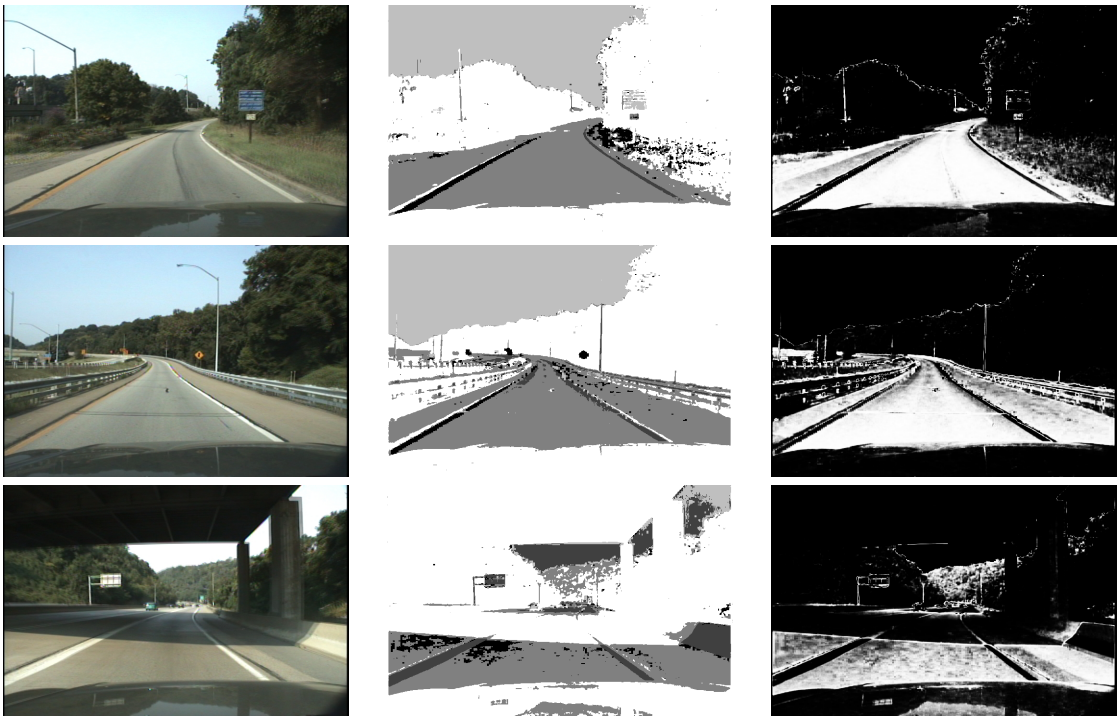
Whereas painted stripes are strong cues for the boundaries of lanes, a second important lane cue is the appearance of the interior region. A model of the appearance of drivable tarmac versus stripes and background would further constrain the locations of potential lanes in an image.

There is much work on appearance modeling in general and tarmac detection in particular which we can take advantage of in order to build a robust classifier. The SCARF system [11] uses four Gaussians each for classifying on-road and off-road pixels, and learns their parameters via K-means clustering of training data. Pixels are then classified by their maximum likelihood due to the model of road pixels versus off-road pixels. In [32], the authors develop a histogramming method which captures a notion of the connectedness of pixels to large segments. A model is learned using this technique and segmentation takes place using histogram matching. In [1], an Adaboost classifier is trained on the output of an oriented filter bank.

The results presented in these papers tend to focus on rural scenes, where the off-road regions are characterized by unstructured textures. They have yet to be evaluated for urban scenes, where there exist many man-made structures similar in appearance to tarmac. We suspect that a scheme will be needed which takes advantage of the texture orientation (e.g. buildings and highway barriers will tend to have more vertical orientations than tarmac). The authors of [18] describe a method for classifying the appearance of image segments as vertical structure, ground plane, or sky, taking advantage of the orientation of their texture, their location in the image, and the shape of the segment boundaries.

Our initial attempt at classification of road scene pixels was tested on a single sequence captured from our test vehicle. We hand-labeled approximately 100 points each for the following classes: white stripes, yellow stripes, tarmac, sky, and background. For each pixel, we computed eighteen features: the HSV values plus the output of fifteen filters taken from the Leung and Malik set [27]. We then trained a linear Gaussian classifier and tested various images from the same sequence. The results are shown in Figure 4. The initial results are promising, but do not generalize well.

We propose to further study classification methods applied to the problem of road scene classification in urban environments. The previous methods mentioned above will serve as a starting point, but if these methods are found lacking, we will investigate methods which incorporate features more appropriate for distinguishing off-road structures from tarmac in urban road scene video sequences. Two possibilities specific to our application present themselves immediately: (1) given a sequence of images collected from our vehicle, we can use features such as optical flow, and (2) we can also set up the test vehicle with a stereo pair of cameras, which will provide strong evidence for 3D structure in each image.



(a) (b) (c)

Figure 4: Results of pixel classification for three images from our test sequence. (a) The input images; (b) the MAP classes for each pixel, ranging from dark to light: yellow stripes, white stripes, tarmac, sky, and background; and (c) the posterior probability for the tarmac class.

3 Modeling and Tracking of Lane Structures

The problem of modeling road scenes (ego-state, tarmac, pedestrians, and other vehicles) presents different challenges for prediction and driver warning systems than it does for applications such as lane keeping, adaptive cruise control, or collision detection. While those applications require information about the ego-lane only, we can identify three necessary extensions: allowance for *flexible road curvatures*, modeling of *all lanes* on the road, and detection and tracking of *lane line properties* such as dashedness and color. These extensions allow us to make queries which mutually constrain our car and pedestrian tracking modules. For example,

- What parts of the ground plane are drivable versus off-road? *We expect to find vehicles on the road, and we expect to find road under vehicles. Similarly, we are more likely to find pedestrians off the road.*
- How is the drivable ground plane divided up into lanes? *A vehicle is more likely to be in the center of a lane than on the boundary.*
- What direction of travel is each lane meant for, and is it a parking lane? *The likely velocity of a vehicle and the presence of a pedestrian are constrained by the type of lane they are in.*

In this section, we first discuss previous work in the literature on these problems. We then describe our attempts to address them.

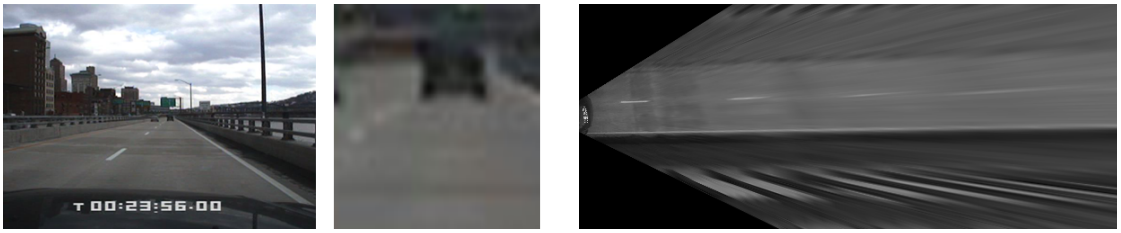
3.1 Previous work

Curvature modeling. In their seminal work, Dickmanns *et al*. argue for modeling the ego lane in 3D using a *clothoid* – a parametric curve whose radius of curvature varies linearly with distance, commonly used in engineering of high-speed roadways – and approximate it using a third-order polynomial [13]. Many subsequent authors [38, 36, 15] have used the same model, while others use parabolic [34, 3, 7, 29] or circular [28] approximations. All of these systems assume a single curve within the sensor range, whereas Jung and Kelber [20] use two curves, a straight-line model plus a parabolic model for the far field of view, under the assumption that the road is approximately straight near the vehicle. Wang et al. [44] model the road using a pair of image coordinate B-snakes [22], which are more flexible in handling the wide variety of lane shapes.

Nonparametric models allow for the greatest flexibility in lane shapes. The GOLD system [4] detects the most likely pixel location of the left, center, and right lane line of the road in each row of a top-down warped image, constraining them to be the same width and smoothly varying in offset. The approach of Chaupuis *et al*. [9] uses straight line interpolation of ten control points in the image for each of the left and right lane lines, constrained by a learned prior covariance matrix.

Multiple lanes. The SCARF system was the first model-based attempt at detecting splits and merges in the tracked road. It assumes that the main road is straight, and models it as an image triangle with its apex on the horizon line. This triangle is fit to a color-based tarmac segmentation. The system then tests for possible branches by adding triangles that extend from the main one, using heuristics to limit the search space. The ground plane location and road width are assumed constant, and the parameters are not tracked over time.

Risack *et al*. describe a system which tracks the ego-lane using a Kalman filter and a parabolic curve model, testing for the presence of additional lanes to the right and left at each time step [38]. In contrast, the GOLD system [4] does not use a curvature model of lanes. Rather, it warps the image to a top-down view, detects stripe pixels, and detects possible widths at each row, assuming a two-lane roadway and that each



(a) (b)

Figure 5: (a) The input image, and an example of the lack of pixel information due to distant stripes. (b) The top-down view created using the homography induced by the ground plane. Notice the extra stripes in the warped image due to the holes in the guardrail.

stripe pixel is due to either the left, right, or middle lane line on the road. The most likely width at each row is computed, allowing for nonparametric roads to be detected in each image.

None of these systems attempt to explicitly model the number of lanes on the roadway, nor do they detect and track that which are not mutually parallel.

Lane line properties. To date, no known system explicitly models lane lines as dashed or solid, or detects the color (yellow versus white) of the lane line. In most countries, a dashed lane line indicates the legality of a lane change, and the yellow lane line indicates opposite direction of travel in the adjacent lane. These are important cues for a prediction system, as they indicate the expected velocity of vehicles in adjacent lanes. They also provide information to a prediction system as to the likely actions of the ego vehicle and other vehicles.

3.2 Multiple hypothesis tracking of road stripes

Tracking an unknown number of lanes can be thought of as a multiple target tracking (MTT) problem: at each time step, we are given a varying number of stripe measurements, and we must (a) assign each of them to a known lane boundary, a new boundary, or noise, and (b) efficiently track the targets based on the assignment. One of the seminal approaches, due to Reid [37, 10], is called the multiple hypothesis tracking (MHT) algorithm. It considers every possible association of measurements to targets, and evaluates the probability of each based on the distance of the measurements to the targets. Given each possible assignment (hypothesis), the targets are tracked using a Kalman filter. Since there is a combinatorially large number of assignments at each time step, various strategies are employed to constrain the size of the hypothesis tree.

We implemented a multiple hypothesis tracker for road stripes. In the following we describe the major components: *measurements*, *target representation*, and *tracking*.

Measurements: We start by warping each input image to a top-down view, as shown in Figure 5(b). Assuming a flat ground plane, we can compute the homography induced by it between the vehicle's calibrated camera and a virtual downward-looking camera. This approach has the advantage of enhancing distance features and making the edges of stripes, which are parallel on the tarmac, parallel in the image, thereby reducing the range of stripe widths we need to detect. However, the assumption of a flat ground plane will create errors on bumpy roads, or when the relative camera pose is not well calibrated. Our technique also has the disadvantage of enhancing features not on the tarmac, often making them look "stripe-like" in the top-down view.

We then run our stripe detector, discussed in section 2.1, on the top-down view and group the resulting pixels. On the assumption that any valid painted road marking will be at least one meter long (e.g. a dash), we

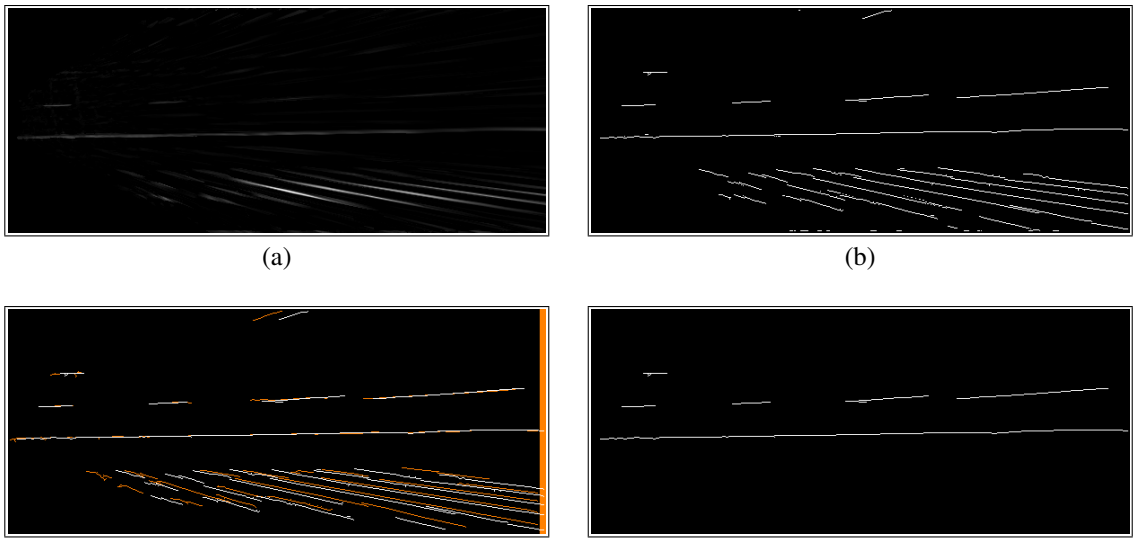


Figure 6: Filtering the top-down view. (a) The result of filtering the input image (Figure 5b). (b) The result after non-maximal suppression and hysteresis thresholding. (c) The thresholded output with small segments removed (white), superimposed on the transformed result from the previous frame (orange). Note that the detected stripes on the road match up with the previous frame, whereas the detected stripes from the guardrails on the sides of the road do not. (d) The final result after matching.

discard any groups which have fewer pixels than would span one meter in the top-down view. The remaining groups, shown in Figure 6(a and b) are stored in a database of observations across time.

Next, we use the ego-vehicle odometry to eliminate stripes due to objects not on the ground plane. We check that each group matches one from the previous time step, transformed according to the steering radius r (positive values indicate steering to the right) and distance d travelled by the car since the last frame:

$$\begin{bmatrix} x_t \\ y_t \end{bmatrix} = \begin{bmatrix} \cos\theta_t & -\sin\theta_t \\ \sin\theta_t & \cos\theta_t \end{bmatrix} \begin{bmatrix} x_{t-1} \\ y_{t-1} \end{bmatrix} + \begin{bmatrix} r_t(1-\cos\theta_t) \\ -r_t\sin\theta_t \end{bmatrix},$$

where $\theta = -d/r$. Scaling the translational components of this transformation according to the pixels per meter in the top-down view allows us to update the pixel-wise positions of the segments from the previous time step to match the current segments (Figure 6(c)). Then, we check the percentage overlap of each segment with the pixels of segments from the previous time step, and we retain only those segments which exceed a threshold. The final result is shown in Figure 6(d).

Tracking: The MHT algorithm takes the view that each detected group of pixels may be a result of (a) a previously known stripe, (b) a new stripe, or (c) a noise process. This leads to a combinatorially large number of possible assignments. In addition, if a target is not assigned a measurement, there is the possibility that it disappeared. For a nontrivial number of stripes and measurements, enumeration quickly becomes intractable.

We use two well-known strategies for managing the number of hypothesis. The first is *gating*: we only consider possibilities in which all the measurements are within a specified distance of the stripe they are assigned to. This eliminates consideration of extremely unlikely associations. The second, *clustering*, is an extension of gating. If a group of stripes is far enough away from all other stripes that all measurements associated with one of its members cannot be associated with a nonmember, then that group is split into a

separate cluster and can be tracked with a separate hypothesis tree. In this way, the problem is factorized into several disjoint tracking problems until a measurement becomes potentially associated with a stripe from two different clusters, in which case they are merged. Finally, at each time step, we *prune* the set of hypothesis by (1) discarding hypothesis with extremely low probability, and (2) keeping only the top K hypothesis.

The probability of each hypothesis at time t, $\Theta_i^t = \{\Theta_{p(i)}^{t-1}, \theta_i^t\}$, where $p(i)$ is the parent index of hypothesis i and θ_i^t is the set of assignments of measurements to stripes, can be calculated using the current set of measurements $Z^t = \{Z^{t-1}, z^t\}$ and Bayes' rule:

$$\begin{aligned} P(\Theta_i^t|Z^t) &= P(\theta_i^t, \Theta_{p(i)}^{t-1}|Z^{t-1}, z^t) \\ &\propto p(z^t|\theta_i^t, \Theta_{p(i)}^{t-1}, Z^{t-1}) P(\theta_i^t|\Theta_{p(i)}^{t-1}, Z^{t-1}) P(\Theta_{p(i)}^{t-1}|Z^{t-1}) \,. \end{aligned} \quad (1)$$

The first term of Equation 1 is the probability of the measurements being generated according to their assignments. In our application, if a measurement z_k^t is due to a known stripe S_i^l, it is modeled as a normal distribution about the sum of squared minimum distances $d(z_k^t, S_i^{l,t})$ between a pixel in the measurement and a pixel in the stripe. If the measurement is due to noise, it is modeled as equally likely to occur anywhere in the image volume (*i.e.* the number of pixels) V. The resultant expression is

$$p(z_k^t|\theta_i^t, \Theta_{p(i)}^{t-1}, Z^{t-1}) = \prod_{k=1}^{M_i} \mathcal{N}(d(z_k^t, S_i^{l,t}); 0, \sigma^d)^{\tau^k} V^{-(1-\tau^k)}$$

where M_i is the number of measurements assigned to hypothesis i and $\tau^k = 1$ if measurement k in hypothesis i is due to a known stripe and 0 if it is due to noise.

The second term of Equation 1 is the probability of the assignment itself, and works out to [10]

$$P(\theta_i^t|\Theta_{p(i)}^{t-1}, Z^{t-1}) = \frac{\phi! v!}{M_i} \mu_F(\phi) \mu_N(v) \prod_{l=1}^{N_i} (P_D)_l^{\delta} (1 - P_D)^{(1-\delta_l)} (P_\chi)^{\chi_l} (1 - P_\chi)^{(1-\chi_l)} \,.$$

Here, v is the number of measurements from known stripes, ϕ is the number of measurements from noise, $\mu_F(\phi)$ is the PDF over the number of noise measurements seen in a given time step, $\mu_N(v)$ is the PDF over the number of new stripes, and δ_l and χ_l are indicator functions set to 1 if stripe l was detected or deleted, respectively, and 0 otherwise. We model $\mu_F(\phi)$ and $\mu_N(v)$ as Poisson processes with parameters λ_F and λ_N. P_D is the probability that a known stripe is re-detected at each time step, while P_χ is the probability that a stripe will disappear.

Finally, the third term of Equation 1 is the probability assigned to the parent hypothesis from the previous time step.

Once each new hypothesis has been created, the system tracks each stripe according to the measurement assigned to it. The Kalman filter provides a theoretically sound way of doing this, by computing a gain matrix based on the variance of the stripe, the measurement noise model, the process noise model, and the distance from the measurement to the target (defined above). The stripe is moved toward the measurement according to the gain factor, and the variance of the stripe is updated accordingly. In practice, we update the variance in the standard way, but because a proper definition of "toward" is difficult to identify with respect to our distance measure, we simply assign the target state to the pixels of the measurement.

The interested reader is directed to [10] for further details of multiple hypothesis tracking.

Results: Quantitative results are difficult to obtain for stripe tracking due to the difficulty of exactly labeling all targets in every frame of a sequence. We show qualitative results on a two-lane highway sequence with an exit ramp in Figure 7. The system is able to quickly detect likely stripes by frame 2, and it detects the stripe which separates the exit ramp while it is still far away. However, the system loses track of the right lane line

Frame 2

Frame 15

Frame 35

Frame 95

Frame 106

Frame 112

Figure 7: Results of multiple hypothesis tracking of stripes, showing the hypothesis with the highest probability at each time step. The exit lane is detected very early at frame 15. The system loses track of the right lane line at frame 95 but recovers by frame 112.

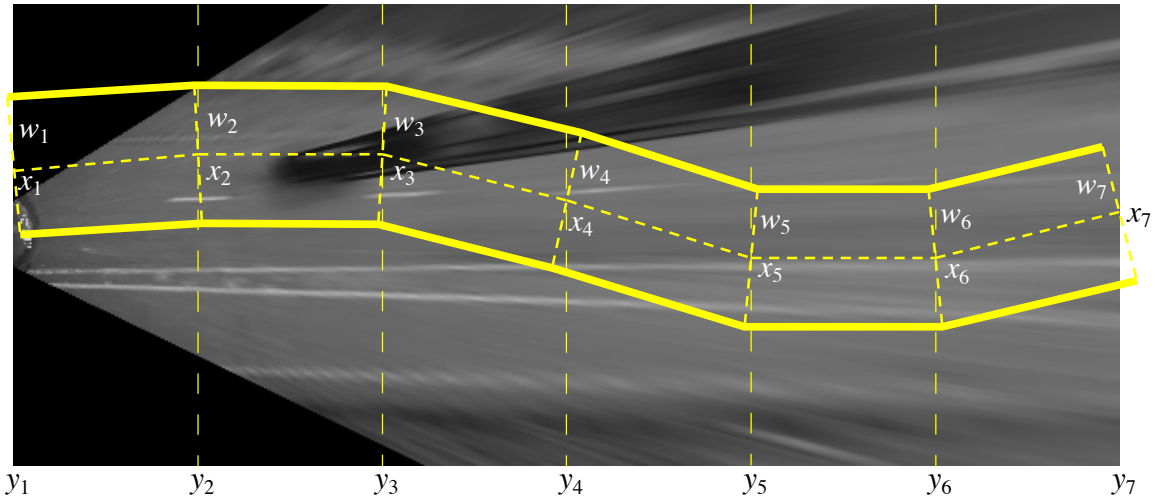


Figure 8: A nonparametric lane model

at frame 95, and does not recover until a more likely hypothesis takes over at frame 112. Since we are not modeling the pitch of the camera relative to the road, the generation of the top-down view often violates the motion model of the stripes when the vehicle goes over a bump. An MPEG movie of these results can be downloaded from `http://www.cs.cmu.edu/~jsprouse/papers/2006_proposal/mht.mpg`.

This system is a promising first step towards tracking of multiple lane lines. However, the main drawback is the necessity of grouping stripes into clusters for efficiency, because there is no clear way of assembling these disjoint hypothesis into coherent global hypothesis which account for the relationships between lane lines. In the next section, we investigate a model which encodes the notion that a lane, while not necessarily conforming to any family of curves, should be of constant width and it's boundaries parallel.

3.3 Stochastic tracking of nonparametric lane shapes

Two main drawbacks of the previous system are the combinatorial complexity of enumerating and evaluating all the possible data associations at each time step and the difficulty of integrating the relationships between stripes. We have developed a particle filter-based system which makes progress toward addressing both of these issues.

In this approach, we have developed a nonparametric model of a single lane, shown in Figure 8. It is more flexible than the parametric curve models used in previous works, but of higher dimension and therefore more challenging to track. We represent the center of the lane as a connected series of straight line segments, with control points at fixed distances $\{y_i\}$ where $i = \{1, \ldots, N_y\}$ on the ground plane in front of the vehicle. The parameters consist of the horizontal offset $\{x_i\}$ and lane width $\{w_i\}$ at each distance. The total dimension of the model is $2N_y$. For our application, we use seven distances $y = \{5m, 10m, \ldots, 65m\}$, for a total of 14 parameters in our state space. This approach is closest to that used in [9], although they model each lane-line separately.

This lane model implicitly encodes constraints on the width of a lane, as well as the notion that the boundaries of a lane should be approximately parallel. It will be fit to the lanes in the top down view, generated in the same manner as our MHT system described in the previous section. However, instead of grouping the thresholded stripe pixels and using them as measurements for a multiple-hypothesis Kalman

filter, we will use them to evaluate each hypothesis of a particle filter [14].

Particle filters eliminate the need to enumerate every possible hypothesis by maintaining a set of N samples from the distribution of possible hypothesis, which is potentially multi-modal due to the presence of multiple lanes. Each sample (particle) represents a possible lane configuration $\mathbf{x} = \{x_1, w_1, \ldots, x_7, w_7\}$. The distribution of lanes in the image at any time step can be approximated by

$$P(\mathbf{x}_t|\mathbf{I}^t) = \sum_{i=1}^{N} \delta(\mathbf{x}_t - \mathbf{x}_t^{(i)}) \,,$$

where $\mathbf{I}^t = \{I_1, \ldots, I_t\}$ is the history of stripe detector outputs at each time step, and $\delta(.)$ is the Kronecker delta function.

At time $t = 0$, the particles are initialized by sampling from a prior distribution over lane shapes, $p_0(\mathbf{x})$. At each subsequent time-step, given a set of particles $\{\mathbf{x}_{t-1}^{(i)}\}$ from the previous time $t-1$ and a new image I_t, a new set of particles is produced as follows:

1. Propagate each particle through a motion model of how lane shapes change over time by sampling from its distribution $p(\mathbf{x}_t|\mathbf{x}_{t-1}^{(i)})$.

2. Re-weight the particles according to a *score function* (how well the proposed lane shape matches the new measurements at time t):
$$\hat{\pi}_t^{(j)} = f(I_t|\mathbf{x}_t^{(j)}) \,.$$

3. Normalize the weights to sum to one:
$$\pi_t^{(j)} = \frac{\hat{\pi}_t^{(j)}}{\sum_{i=1}^{N} \hat{\pi}_t^{(i)}} \,.$$

4. Resample a new set of particles according to the empirical distribution of weights to produce a sample from the filtering distribution.

This particular form of particle filtering is known as *sequential importance re-sampling* (SIR) [14].

Motion Model: The motion model $p(\mathbf{x}|\mathbf{x}_{t-1})$ describes how lane shapes change over time. In our application, we will assume that the vehicle is moving according to the curvature of the lane, such that in the absence of any noise in the measurement process the parameters describing the lane would not change at all. Therefore, we only need to model the noise in the measurement process. We do this using a Gaussian noise process:

$$p(\mathbf{x}|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}; \mathbf{x}_{t-1}, \Sigma) \,,$$

where $\Sigma = \mathrm{diag}(\sigma_{x1}, \sigma_{w1}, \ldots, \sigma_{x7}, \sigma_{w7})$.

In order to prevent the system from getting stuck in a local minimum, we also sample a small number of particles at each time step from the prior distribution $p_0(\mathbf{x})$. This is known as *importance sampling*.

Observation Model: We divide the score function $f(I_t|\mathbf{x}) = f_I(I_t|\mathbf{x}) f_S(\mathbf{x})$ into two components: a *shape score* f_S and an *appearance score* f_I.

The appearance scoring function should measure the distance between a lane hypothesis $\mathbf{x}_t$ and the current stripe detector output. We would like to avoid having this function being too "peaked", as this is known to increase the variance of particle weights, thereby decreasing the effectiveness of the particle filter [14]. We

Top-down view generation		Particle filtering	
Image size	250×600	Num. particles	350
Ground region X	[-12.5m, 12.5m]	Num. from prior	50
Ground region Y	[5m, 65m]	σ_{x_i}	0.4m
		σ_{w_i}	0.2m
		d_{max}	1m
Filtering		$\sigma_{\theta 12}$	$\pi/8$
Stripe widths	$\{1, 2, 3\}$	$\sigma_{d\theta}$	$\pi/8$
Hys. threshold max	0.05	μ_w	3.2m
Hys. threshold min	0.01	σ_w	2.0m
Pct. overlap between frames	50	σ_{dw}	1.0m

Table 1: Parameters for stochastic tracking of nonparametric lane shapes

use an inverted distance transform of the final detected road stripes, cropping it at a cutoff distance d_{max}: the score $L(i)$ at each pixel i is given by

$$L(i) = 1 - \min(DT(i)/d_{max}, 1)\,.$$

.

A good lane hypothesis should meet two requirements: (1) there should be stripes at the predicted lane boundary, and (2) there should not be stripes in the interior of the lane. We therefore further break down the appearance score into two terms:

$$\begin{aligned} f_{lines}(I_t|\mathbf{x}) &= \frac{1}{||S_{lines}||} \sum_{i \in S_{lines}} L(i), \qquad \text{and} \\ f_{mid}(I_t|\mathbf{x}) &= 1 - \frac{1}{||S_{mid}||} \sum_{i \in S_{mid}} L(i)\,, \end{aligned}$$

where S_{lines} and S_{mid} are the pixels in the top down view predicted to be lane boundary points and mid-lane points, respectively. The final appearance score is the product of these two: $f_I(I_t|\mathbf{x}) = f_{lines}(I_t|\mathbf{x}) f_{mid}(I_t|\mathbf{x})$.

Next, the shape score is used to reduce the weight of lane hypothesis of unlikely configuration. Our criteria for this are (1) that the width and curvature of the road should change smoothly, and (2) the closest lane segment should be roughly parallel to the direction of travel, reflecting the assumption that the vehicle is traveling down the middle of the lane. The curvature and initial angle constraints are modeled using a set of Gaussian functions:

$$f_\theta(\mathbf{x}) = \exp\left\{-\frac{(\theta_{12} - \pi/2)^2}{\sigma_{\theta 12}}\right\} \prod_{i=2}^{6} \exp\left\{-\frac{(\theta_i - \theta_{i-1})^2}{\sigma_{d\theta}}\right\}\,,$$

where $\theta_i = \tan^{-1}\left((y_{i+1} - y_i)/(x_{i+1} - x_i)\right)$. The width constraint is also modeled using a set of Gaussians:

$$f_w(\mathbf{x}) = \exp\left\{-\frac{(w_1 - \mu_w)^2}{\sigma_w}\right\} \prod_{i=2}^{7} \exp\left\{-\frac{(w_i - w_{i-1})^2}{\sigma_{dw}}\right\}\,.$$

The final shape score is the product of these two: $f_S(\mathbf{x}) = f_\theta(\mathbf{x}) f_w(\mathbf{x})$. An example of the resultant scored distribution of lane shapes can be seen in Figure 9.

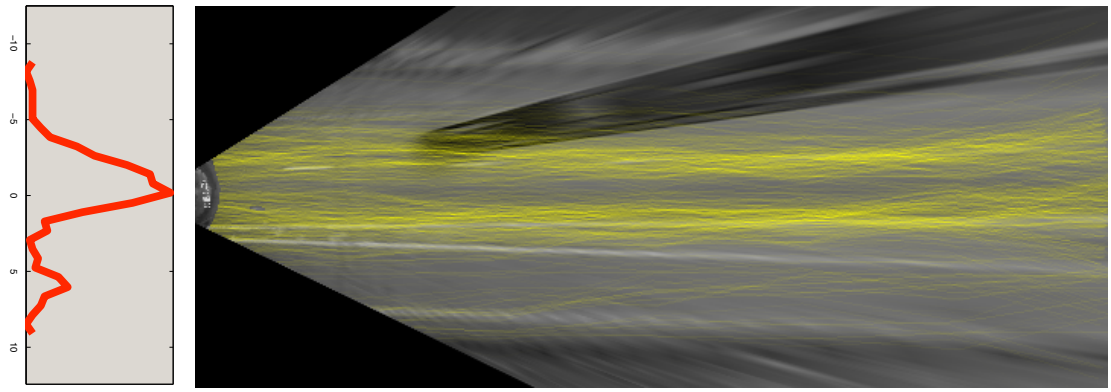


Figure 9: Results of particle filtering the lane mesh after six frames. Left: the distribution of lane centers x_1 closest to the vehicle. Right: the posterior distribution of lane boundaries.

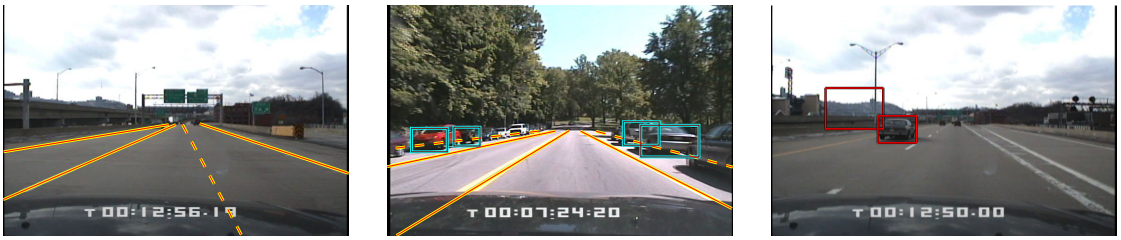
Results: We have tested our system on several video sequences taken using our test vehicle. Example results can be found at `http://www.cs.cmu.edu/~jsprouse/papers/2006_proposal/pflanes.mov`. Our settings for the system parameters are shown in Table 1.

The algorithm was implemented using Matlab and runs at approximately 8.75 seconds per frame on a dual 3.06GHz Xeon PC. Figure 9 shows the posterior distribution represented by the particle filter after six frames. Looking at the marginal distribution of x_1, we can see that multiple modes are present in the posterior corresponding to the ego lane and the exit lane to the right. This shows that the system has the capacity to fit a multi-modal distribution of lanes in an image.

However, as the results show, particle filters are notorious for having trouble in high-dimensional spaces, and therefore will have difficulty fitting the data exactly. As we move toward modeling more of the properties of lanes, roads, and road scenes, the dimensionality of our state space will only increase. In section 4, we discuss our proposed solution.

4 Contextual modeling

Our goal of detecting and tracking all of the objects relevant to a driver warning system could be accomplished by aggregating the output of a lane line tracker, a car tracker, a pedestrian tracker, and algorithms designed for other object classes of interest. However, there are many cases in which evidence for the existence and location of an object is not available from visual data, but we are able to identify and track it nonetheless. For example, when a lane becomes significantly wider than the expected lane width relative to other lanes on a road, we can infer that it has split into two lanes bounded by an unseen lane line, which we can track using the location of the neighboring lane lines, as shown in Figure 10(a). Similarly, when a lane is not visible due to a row of parked cars as shown in Figure 10(b), we nonetheless know it is present since cars are more likely to be found on a lane than off. In contrast, Figure 10(c) shows an example where a hypothesized object can be rejected if it does not make sense in context.



(a) (b) (c)

Figure 10: Examples of context propagation. (a) Some lane lines (solid) can be tracked using visual features, while others (dashed) are *implied* by context – in this case, the width of the right-hand lane. (b) In this case, the lane lines bounding the parking lanes are implied by the presence of cars. (c) The false positives output by a car detector can be rejected if they do not make sense relative to the rest of the scene.

In this thesis, we will focus on modeling two types of contextual relationships: (1) lane structure relationships, and (2) the presence and 3D location of cars relative to the road. Assuming a flat ground plane, the simplest way these two subsystems can be related is through the camera viewpoint, as shown in Figure 11.

4.1 Lane structure relationships

In tracking the lane structure of the road, a set of rules can be enumerated which describe the interrelationships between lane lines, lanes, and roadways.

- Lane lines can be tarmac edges, solid stripes, dashed stripes, or implied (meaning that the lane is bounded implicitly).
- A dashed lane line always divides two driving lanes, while a solid lane line can bound one lane or divide two lanes. A tarmac edge always bounds a lane.
- Lanes on a roadway have a default width, and the default width of a roadway has a universal prior. Lanes should not significantly exceed this default width before being considered split into two lanes. Lanes with width much less than this width are considered ”shoulder” lanes (lanes not meant to be driven on).
- Painted markings in the middle of lanes which can be interpreted as text, directional arrows, crosswalk delimiters, etc., should be ignored by the lane tracker. If they form unknown shapes or there is other noise present in the lane, it is less likely to be a valid driving lane.

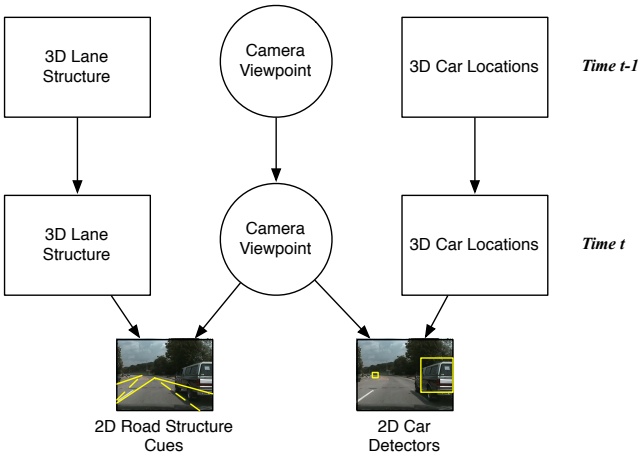


Figure 11: System diagram. Assuming that the ground plane is flat, the only hidden parameter the two subsystems have in common is the camera viewpoint.

Our goal is to build a system which uses these rules in interpreting how the ground plane is divided up into tarmac versus non-drivable surface, and how the tarmac is divided into lanes. For this thesis, we propose to (1) develop a road structure template which can encode these rules, and (2) fit the template using stochastic methods.

In section 3.3 we devised a lane shape template which describes a single lane. For the present problem, we propose to expand this template such that it can describe multiple lanes and roadways. The new template is parameterized by $S = \{\mathbf{x}, N_l, S_l^1, \ldots, S_l^{N_l}\}$, where $\mathbf{x}$ is a vector of x-positions of the lane centerline and the y-positions in front of the vehicle are specified in advance. Each lane i on the road is parameterized by $S_l^i = \{\mathbf{d^i}, \theta_i\}$, where $\mathbf{d^i}$ is a vector of length $N_l - 1$ of perpendicular distances from the centerline at each control point in $\mathbf{x}$, and θ_i describes the color and dashedness of the line. An example is shown in Figure 12.

Fitting this template involves maximizing the posterior $p(S|\mathbf{I})$ which measures how well the template describes the image. Finding the maximum *a posteriori* (MAP) state S^* is difficult due to the high dimensionality of the model, and the presence of local minima in the state space. However, recent works [23, 43] have shown success in simulating posteriors over such state spaces using Markov Chain Monte Carlo search. A set of reversible transition kernels $\mathcal{K}_a(S'|S)$ are constructed which obey the detailed balance condition:

$$p(S|\mathbf{I})\mathcal{K}_a(S'|S) = p(S'|\mathbf{I})\mathcal{K}_a(S|S')$$

At each iteration of MCMC, a kernel is selected with probability π_a, where $\sum_a \pi_a = 1$, and a new state S' is sampled from the it. In the Metropolis-Hastings form of MCMC, sampling from a kernel $\mathcal{K}_a$ is equivalent to sampling from a proposal distribution $Q_a(S'|S)$ and accepting the proposal with probability

$$\alpha(S'|S : \mathbf{I}) = \min\left\{1, \frac{p(S|\mathbf{I})Q_a(S'|S)}{p(S'|\mathbf{I})Q_a(S|S')}\right\}$$

This form of *importance sampling* allows us to generate a sample from $p(S|\mathbf{I})$, which we only need the ability to evaluate but not sample from. This sample can be used to estimate the MAP, or the expected value of a function of the state space.

We propose to implement MCMC for parsing road scenes. For our application, we define three reversible transition kernels.

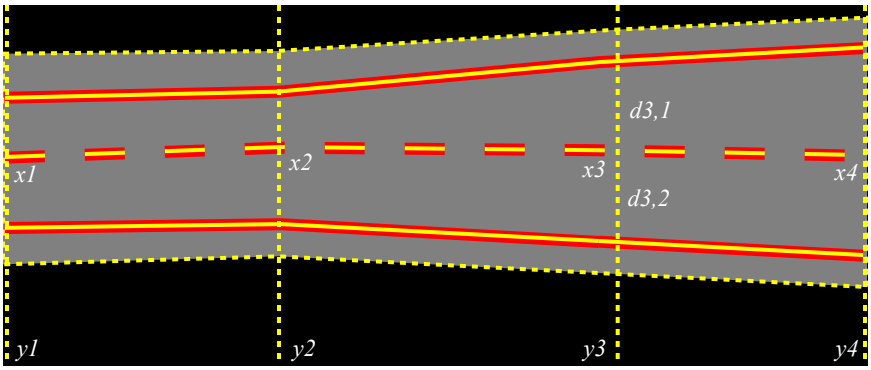


Figure 12: Lane structure template

1. Select a control point of the template and move it (diffusion).

2. Split a lane, or merge two lanes into one.

3. Switch a lane line between dashed and solid.

The probability π_a of selecting each kernel changes over time, with diffusion becoming more likely while changes to the structure of the model become less so.

From a Bayesian viewpoint, the scoring function $p(S|\mathbf{I})$ takes the form $c^{-1}P(\mathbf{I}|S)p(S)$, where c is a normalization constant. The likelihood $P(\mathbf{I}|S)$ should incorporate our notions of the appearance of lanes. Each lane line should be supported by image measurements, whether dashed or solid, and there should be tarmac at the interior of the lane. In addition, there should not be tarmac outside the image region defined by the template; this allows templates with different numbers of lanes to compete on the same footing. The likelihood score will utilize our stripe detection and tarmac classification work from Section 2. The prior $p(S)$ should incorporate our restrictions on lane shape, including its degree of curvature and our constraints on allowable widths.

We will first implement a system to fit our lane template to single road images, followed by an investigation of incorporating priors from the previous time step for tracking. Then, we will consider possible extensions to the model:

- Estimation of the camera viewpoint and ego-vehicle speed and steering angle.

- A model of stripe color and the implied direction of travel of each lane.

- A model of on/off-ramps which branch away from or merge with the main roadway.

- A model of multiple roadways in the same scene.

In developing this system, we will use the output of our car detector/tracker to mask out image regions which are occluded so that they do not have to be explained by the template. However, in tracking the camera viewpoint, the location and size of other vehicles on the roadway provide a strong mutual constraint. We wish to integrate our lane structure tracking with the car tracker in such a manner that the final interpretation is consistent.

4.2 Holistic road scene understanding

State of the art car and pedestrian detectors such as [40] typically produce a set of 2D bounding rectangles. For our purposes, the 3D position of the object can be inferred using the flat ground plane assumption along with the vertical coordinate of the bottom of the box, as discussed in [19]. However, these subsystemsoften produce false positives and negatives. One advantage of holistic modeling of road scenes is that the interrelationships between object classes can be brought to bear on this problem. For example, cars are more likely to be on the tarmac than off it, and more likely to be in the middle of a lane than on a lane line. Also, the location of the camera with respect to the ground plane provides constraints as to the possible locations in the image of pedestrians, given their height in the image.

Many intelligent vehicle systems include modules which detect and track different objects in the scene, the most common being the ego-lane and other vehicles. However, there has been very little work on modeling the relationship between them. The GOLD system [4] detects obstacles by comparing a stereo pair of inverse-perspective warped images of the ground plane. Dellaert *et al*. used a Kalman filter to track a joint model of the ego lane and a single car, with no inter-constraints [12].

More general attempts to model spatial relationships between object categories include [19]. Here, the 3D orientation of each pixel in a single image is labeled as vertical, ground, or sky. This is combined with the output of car and pedestrian detectors into a graphical model [33] which simultaneously estimates the horizon position and the probability that each detection is a type I error. This method brings modern statistical techniques to the scene interpretation of early AI research [31, 16, 6].

We propose a similar system to model cars and pedestrians in our system. Two modifications will need to be studied: first, we will incorporate priors from the previous frame in a tracking context; *i.e*., we will reformulate the problem as a dynamic bayesian network [30]. Second, we will investigate how to incorporate inference of the camera viewpoint with our lane structure tracker in order to produce a mutually consistent result. The end goal is system capable of tracking a consistent interpretation of the road scene over time.

5 Schedule

Topic	Target completion date
LOW LEVEL ROAD SCENE ANALYSIS	
Detecting stripe-like features	Complete
Road scene classifier for tarmac vs. background discrimination	Fall 2006
LANE STRUCTURES	
Particle filter for single-lane template tracking	Complete
MCMC for multi-lane template fitting and tracking	Spring 2007
Extend to multiple roadways, stripe properties, and viewpoint modeling	Spring 2007
HOLISTIC ROAD SCENE MODELING	
Implement car detector	Complete
Implement pedestrian detector and tracker	Summer 2007
Investegate of mutua constraints due to lane, car, and pedestrian tracking	Summer 2007
THESIS	
Write dissertation and defend	Fall 2007

6 Conclusion

In this thesis, we propose to develop a tracker for a model of multi-lane structures on roadways, and integrate it with trackers for other road scene objects in such a way that the mutual constraints between the systems are satisfied. The result, a holistic description of all relevent semantic objects relative to the ego-vehicle, is crucial for the real-world deployment of prediction and driver warning systems for intelligent vehicles.

References

[1] Yaniv Alon, Andras Ferencz, and Amnon Shashua. Off-road path following using region classification and geometric projection constraints. *CVPR*, 1:689–696, 2006.

[2] Nicholas Apostoloff and Alexander Zelinsky. Robust vision based lane tracking using multiple cues and particle filtering. In *IEEE Intelligent Vehicles Symposium*, 2003.

[3] M. Beauvais, C. Ckeucher, and S. Lakshmanan. Building world models for mobile platforms using heterogeneous sensors fusion and temporal analysis. In *IEEE Intelligent Vehicles Symposium*, pages 171–176, 1996.

[4] M. Bertozzi and A. Broggi. GOLD: A parallel real-time stero vision system for generic obstacle and lane detection. *IEEE Transactions on Image Processing*, 7(1):62–81, January 1998.

[5] Adrian E Broadhurst, Simon Baker, and Takeo Kanade. Monte carlo road safety reasoning. In *IEEE Intelligent Vehicle Symposium (IV2005)*. IEEE, June 2005.

[6] R. Brooks, R. Greiner, and T. Binford. Model-based three dimensional interpretation of two-dimensional images. *Proc. Int. Joint Conf. on Art. Intell.*, 1979.

[7] T. Bücher, C. Curio, J. Edelbrunner, C. Igel, D. Kastrup, I. Leefken, Gesa Lorenz, Axel Steinhage, and W. von Seelen. Image processing and behaviour planning for intelligent vehicles. *IEEE Transactions on Industrial Electronics*, 50(1):62–75, February 2003.

[8] J. F. Canny. Finding edges and lines in images. Technical Report 720, MIT Artificial Intelligence Laboratory, 1983.

[9] R. Chapuis, R. Aufrere, and F. Chausse. Accurate road following and reconstruction by computer vision. *Intelligent Transportation Systems, IEEE Transactions on*, 3(4):261–270, 2002.

[10] Ingemar J. Cox and Sunita L. Hingorani. An efficient implementation of Reid's multiple hypothesis tracking algorithm and its evaluation for the purpose of visual tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(2):138–150, February 1996.

[11] J. Crisman and Chuck Thorpe. SCARF: A color vision system that tracks roads and intersections. *IEEE Trans. on Robotics and Automation*, 9(1):49 – 58, February 1993.

[12] Frank Dellaert, Dean Pomerleau, and Chuck Thorpe. Model-based car tracking integrated with a road-follower. In *International Conference on Robotics and Automation*, May 1998.

[13] Ernst D. Dickmanns and Birger D. Mysliwetz. Recursive 3-d road and relative ego-state recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2), February 1992.

[14] Arnaud Doucet, Nando de Freitas, and Neil Gordon, editors. *Sequential Monte Carlo Methods in Practice*. Springer, 2001.

[15] J. Goldbeck and B. Huertgen. Lane detection and tracking by video sensors. In *IEEE Intelligent Transporation Systems*, pages 74–79, 1999.

[16] A. Hanson and E. Riseman. Visions: A computer system for interpreting scenes. *Computer Vision Systems*, 1978.

[17] F. Heimes and H.H. Nagel. Towards active machine-vision-based driver assistance for urban areas. *International Journal of Computer Vision*, 50(1):5–34, 2002.

[18] D. Hoiem, A. A. Efros, and M. Hebert. Geometric context from a single image. In *ICCV*, October 2005.

[19] Derek Hoiem, Alexei A. Efros, and Martial Hebert. Putting objects in perspective. In *CVPR*, 2006.

[20] Claudio Rosito Jung and Christian Roberto Kelber. An improved linear-parabolic model for lane following and curve detection. In *SIBGRAPI*, 2005.

[21] M. Kais, S. Dauvillier, A. De La Fortelle, I. Masaki, and C. Laugier. Towards outdoor localization using GIS, vision system and stochastic error propagation. *Proceedings of the second International Conference on Autonomous Robots and Agents, Palmerston North, New Zealand, December*, 2004.

[22] M. Kass, A. Wirkin, and D. Terzopoulous. Snakes: Active contour models. *International Journal of Computer Vision*, 1:321–331, 1988.

[23] Z. Khan, T. Balch, and F. Dellaert. MCMC-based particle filtering for tracking a variable number of interacting targets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(11):1805–1918, 2005.

[24] K. Kluge and Chuck Thorpe. Representation and recovery of road geometry in yarf. In *Proceedings of the Intelligent Vehicles '92 Symposium*, pages 114 –119, July 1992.

[25] T. M. Koller, G. Gerig, G. Szekely, and D. Dettwiler. Multiscale detection of curvilinear structures in 2-d and 3-d image data. In *ICCV*, 1995.

[26] C. Kreucher and S. Lakshmanan. Lana: A lane extraction algorithm that uses frequency domain features. *IEEE Transactions on Robotics and Automation*, 15(2):343–350, 1999.

[27] T. Leung and J. Malik. Representing and recognizing the visual appearance of materials using three-dimensional textons. *International Journal of Computer Vision*, 43(1):29–44, 2001.

[28] B. Ma, S. Lakshmanan, and A.O. Hero. Pavement boundary detection via circular shape models. In *IEEE Intelligent Vehicles Symposium*, pages 644–649, 2000.

[29] J. C. McCall and M. M. Trivedi. Video based lane estimation and tracking for driver assistance: Survey, system, and evaluation. *IEEE Transactions on Intelligent Transportation Systems*, 7(1):20–37, March 2006.

[30] K.P. Murphy. *Dynamic Bayesian Networks: Representation, Inference and Learning*. PhD thesis, U.C. Berkeley, 2002.

[31] Y. Ohta, T. Kanade, and T. Sakai. An analysis system for scenes containing objects with substructures. In *IJCPR*, pages 752–754, 1978.

[32] M. A. Patricio and D. Maravell. Segmentation of traffic images for automatic car driving. In *EUROCAST*, pages 314–325, 2003.

[33] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 1988.

[34] D. Pomerleau. RALPH: Rapidly adapting lateral position handler. In *Proc. Intelligent Vehicles*, pages 54–59, 1995.

[35] Dean Pomerleau. *Neural network perception for mobile robot guidance*. Kluwer Academic Publishing, 1993.

[36] K.A. Redmill, S. Upadhya, A. Krishnamurthy, and U. Ozguner. A lane tracking system for intelligent vehicle applications. In *IEEE Intelligent Transporation Systems*, pages 273–279, 2001.

[37] Donald B. Reid. An algorithm for tracking multiple targets. *IEEE Transactions on Automatic Control*, AC-24(6):843–854, December 1979.

[38] R. Risack, P. Klausmann, W. Krüger, and W. Enkelmann. Robust lane recognition embedded in a real-time driver assistance system. In *IEEE Intelligent Vehicles Symposium*, pages 35–40, 1998.

[39] Henry A. Rowley and Takeo Kanade. Reconstructing 3-d blood vessel shapes from multiple x-ray images. In *AAAI Workshop on Computer Vision for Medical Image Processing*, San Francisco, CA, March 1994.

[40] H. Schneiderman. Feature-centric evaluation for efficient cascaded object detection. *CVPR 2004*, 2, 2004.

[41] B. Southall and C. Taylor. Stochastic road shape estimation. In *Proc. Int. Conf. Computer Vision*, pages 205–212, 2001.

[42] Carsten Steger. An unbiased detector of curvilinear structures. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(2):113–125, February 1998.

[43] Zhuowen Tu and Song-Chun Zhu. Parsing images into regions, curves, and curve groups. *International Journal of Computer Vision*, 69(2):223–249, 2006.

[44] Y. Wang, E. Teoh, and D. Shen. Lane detection and tracking using b-snake. *Image and Vision Computing*, 22:269–280, 2004.