A Learning Approach Towards Detection and Tracking of Lane Markings

Raghuraman Gopalan, Member, IEEE, Tsai Hong, Michael Shneier, and Rama Chellappa, Fellow, IEEE

Abstract—Road scene analysis is a challenging problem that has applications in autonomous navigation of vehicles. An integral component of this system is the robust detection and tracking of lane markings. It is a hard problem primarily due to large appearance variations in lane markings caused by factors such as occlusion (traffic on the road), shadows (from objects like trees), and changing lighting conditions of the scene (transition from day to night). In this paper, we address these issues through a learning-based approach using visual inputs from a camera mounted in front of a vehicle. We propose, (i) a pixel-hierarchy feature descriptor to model contextual information shared by lane markings with the surrounding road region, (ii) a robust boosting algorithm to select relevant contextual features for detecting lane markings, and (iii) particle filters to track the lane markings, without the knowledge of vehicle speed, by assuming the lane markings to be static through the video sequence and then learning the possible road scene variations from the statistics of tracked model parameters. We investigate the effectiveness of our algorithm on challenging daylight and night-time road video sequences.

Index Terms—Lane Marking Detection, Context, Boosting, Outlier Robustness, Tracking and Learning.

I. INTRODUCTION

Autonomous navigation of road vehicles is a challenging problem that has wide-spread applications in intelligent systems, and robotics. Detection of lane markings assumes importance in this framework since it assists the driver with details about the road such as, how many lanes are present, are they straight or curved, and so on. It is a hard problem due to variations present in, (i) the appearance of lane markings - solid lines, dotted lines, circular reflectors and their color, yellow or white; (ii) the type of road in which the vehicle is traveling such as highways and city streets, and objects that occlude the lane markings like vehicles and pedestrians; (iii) the time of day in which the scene needs to be analyzed; for instance at night the most visible regions are those that are just ahead of the vehicle, whereas during the day more regions in the field of view of the camera need to be analyzed by the detector; and (iv) the presence of shadows due to objects such as trees that affect the appearance of lane markings. Some examples illustrating these challenges are given in Figure 1.

To deal with the above-mentioned conditions, many approaches have been proposed in the literature. These can be broadly classified based on the type of sensors such as (visual) cameras, internal vehicle state sensors, GPS sensors, laser scanners or radar sensors. Each sensor has its own advantages and limitations. For instance, foggy conditions on the road affect the reliability of cameras, whereas a GPS sensor might be more robust. Although using multiple sensors will certainly help the detection process [1], in this work we are primarily interested in analyzing visual inputs from a single camera mounted in front of a moving vehicle. We now review some related work pertaining to this category.

Prior Work: Detecting road lane markings using image analysis has been an area of active research over the last two decades. The recent survey paper by McCall and Trivedi [2] provides a comprehensive summary of existing approaches. Most of the methods propose a three-step process, (i) extracting features to initialize lane markings such as edges [3], texture [4], color [5], and frequency domain features [6]; (ii) post-processing the extracted features to remove outliers using techniques like Hough transform [7] and dynamic programming [8], along with computational models explaining the structure of the road using deformable contours [9], and regions with piecewise constant curvatures [10]; and then (iii) tracking the detected lane markings using a Kalman filter [11] or particle filters [12], [13] by assuming motion models such as constant velocity or acceleration for the vehicle. There are also methods that use stereo cameras (e.g. [14], [15]) to enforce similarity of points observed from both cameras. More recently, there has been an increased focus on building real-time systems [16] on challenging urban scenarios [17], [18], including night-time driving conditions [19], and on providing functionalities such as lane departure warning [20], and lane reservation on highways [21]. Machine learning methods with
A pixel-hierarchy feature descriptor to model the spatial context information shared by lane markings and the surrounding scene;  
An outlier-robust boosting algorithm to learn relevant contextual features for detecting lane markings, without assuming any prior road model;  
Learning possible variations in the road scene, by assuming that lane markings remain static through the video, and characterizing the tracked model parameters.

**Organization of the paper:** We discuss the detection (localization) of lane markings by extracting contextual features and modeling them with boosting in Section II. Section III deals with tracking and learning the variations in road scene without any prior knowledge of the vehicle’s motion pattern. We then present experimental validation of detection and tracking algorithms in Section IV, using data from both daylight and night-time road sequences. Section V concludes the paper. A block diagram explaining the flow of the proposed approach is given in Figure 2.

II. DETECTION OF LANE MARKINGS

We take a data-driven discriminative approach to classify lane markings $O$ from non-lane markings $O'$. Given a set of $M$ labeled training samples $\{x_i\}_{i=1}^M$ belonging to $O$ and $O'$, we first extract $M_1$ contextual features from every $x_i$ (say, $\{b_{j|x_x}^{x_i}\}_{j=1}^{M_1}$). We then train a classifier on these features using a boosting-based machine learning algorithm, through which the detection of lane markings is performed on test images. More details are provided in the following sub-sections.

A. Modeling the spatial context of lane markings

Context, a loosely defined term in itself, refers to all pertinent information conveyed by the visual scene about the existence of an object [23]. Although the complementary information provided by context has been acknowledged since the early 70’s [26], only in recent years have we seen its explicit modeling in the mainstream object detection literature [27]. Since, (i) the lane markings share a rich neighborhood with the road regions, and (ii) existing methods, at large, characterize the appearance of lane markings in isolation,
understanding the role of context for this problem attains prominence.

Classification of low-level contextual sources, which do not use any higher-level semantic information on the structured grouping of pixels representing different objects, belongs to one of the following categories; (i) top-down methods that compute the gist of the scene by computing some global image statistics, e.g. [27], and (ii) bottom-up methods that correlate the properties of a pixel with its immediate adjoining region, e.g. [28]. Both have relative advantages/disadvantages depending on the application of interest. In this work, we propose a hierarchical descriptor that encodes information of both types.

1) A pixel-hierarchy feature descriptor: Given a pixel \( x \) corresponding to \( O \) or \( O' \), we consider a hierarchy of image regions represented by concentric circles centered at that pixel. Let \( R = \{ R_i \}_{i=1}^{M_2} \) represent the regions\(^2\) enclosed by circles of increasing radius. We now use the ‘visual information’ present in \( R \) to obtain contextual measurements \( h^x = f^x(I_o, I_s) \) for a pixel \( x \), where \( I_o \) denotes the pixel \( x \) in isolation (i.e. the region enclosed by circle of zero radius, centered at \( x \)), \( I_s \) corresponds to the regions enclosed by all other circles of positive radius centered at \( x \), and \( f \) represents a collection of filters that computes context between \( I_o \) and \( I_s \). While the exact definition of ‘visual information’ depends on the application, for this problem, motivated by existing works, we use the intensity image, an edge map output from the Canny operator [29], and texture patterns (obtained from Gabor filters [30] that compute the magnitude and dominant direction of textures at different scales). Let us refer to them as \( F = \{ F_{j=1}^{M_1} \} \). Using this, we compute our basic contextual features for a pixel \( x \)

\[
h^x_j = f^x_j(I_o, I_s) : R \times F \to \mathbb{R}, \forall j = 1 \text{ to } M_1
\]

by analyzing the pattern of \( F \) on different regions \( R \). We used a set of rectangular Haar-like filters [31] for this purpose. These filters have different positive and negative regions, where the positive regions replicate the values of the underlying region \( R \times F \), and the negative regions correspond to a value of zero. The values underneath the positive regions (for each Haar filter) are then added to result in \( h^x_j \), a scalar, and the vector of all such features \( h^x = [h^x_1 \ldots h^x_{M_1}]^T \), denotes the pixel-hierarchy feature descriptor of a pixel \( x \). Typically, the number of features \( M_1 \) is of the order of 1000’s depending on the precision with which the positive and negative rectangular patterns of Haar-filters are varied. An illustration is given in Figure 3.

\[\text{Fig. 3. L-R: An illustration of computing the pixel hierarchy descriptor } h^x \text{ for a pixel } x \text{ on hierarchical circles } R_i, \text{ with the underlying } F_i \text{ corresponding to the intensity image, edge map, and texture response (magnitude patterns of texture response are shown in this figure). 4 wavelet scales with 6 Gabor filter orientations were used. Hence, } M_2 = 26. \) The weak learners \( h_j \) correspond to Haar filters, which when applied on \( R \times F \) result in \( h^x = \{ h^x_j \}_{j=1}^{M_1} = f^x(I_o, I_s) \), \( h^x \) is the pixel-hierarchy descriptor of a pixel \( x \). \( h^x \) computed for \( x \in O, O' \) are used to train the boosting algorithm to compute the strong classifier \( g^*(x) \). This is then used to classify pixels in a test image corresponding to lane markings and others.

\(\text{B. Learning the relevant contextual features through Boosting - Training the classifier}\)

We now require a principled way of selecting relevant features among \( h_j, j = 1 \text{ to } M_1 \), which are most discriminative in classifying pixels \( x \) corresponding to \( O \) from \( O' \). This requirement paves the way to adapting the principles of boosting [24], a machine learning method that determines the optimal set of features to classify objects with provable detection error bounds. Boosting algorithms have been used previously for two-class problems by Viola and Jones [32]
for detecting faces, and Wu and Nevatia [33] for human
detection, where the weak learners $h_j$ modeled the appearance
information of objects in isolation.

1) The Problem of Outliers in Training Set: Before getting
into the details of detecting lane markings using boosting, we
address an important problem in the learning stage of boosting
algorithms: that of the presence of outliers in the training
set. To make the paper self-contained, we present the basic
version of the Adaboost algorithm [24], referred to as Discrete
Adaboost, below.

Given: $(x_1, y_1), \ldots, (x_M, y_M)$ where $x_i \in \mathbb{R}^N$ denotes the $i^{th}$
training sample, and $y_i \in \{-1, +1\}$ denotes its class label, and
classifier pool $\mathbb{H}$ consisting of weak learners $h_j, j = 1, \ldots, M$. Initialize
the training samples with uniform weights $D_1(i) = 1/M, \forall \{x_i\}_{i=1}^M$.

For iterations $t=1,\ldots,T$:

- Train the weak learners using the weight distribution $D_t$.
- Get the weak classifier $\hat{h}_i : \mathbb{R}^N \rightarrow \mathbb{R}$, which, among all
  weak learners $h_j \in \mathbb{H}$, minimizes the error

$$
e_{t} = \frac{1}{D_t} \sum_{i=1}^{M} D_t(i) I(y_i \neq \hat{h}_i(x_i))$$

where $I(\cdot)$ is an indicator function.
- Choose $\alpha_t \in \mathbb{R}$, which is a function of the classification accuracy.
- Update:

$$D_{t+1}(i) = \frac{D_t(i) \exp(-\alpha_t y_i \hat{h}_i(x_i))}{Z_t}$$

where $Z_t$ is a normalization factor chosen such that $D_{t+1}$
will be a distribution.

Output the final classifier:

$$g^*(x) = \text{sign} \left( \sum_{t=1}^{T} \alpha_t \hat{h}_i(x) \right)$$

In summary, given a set of labeled training samples belong-
ing to two classes with the same initial weights, and a pool
of weak learners, the goal of boosting is to find a sequence of
best weak classifiers by adaptively increasing (and decreasing)
the weights of wrongly (and correctly) classified samples in
each stage. However when there are outliers present in the
training set, say due to mislabeled samples or due to samples
that are very different from other neighbors of their class, this
process will result in a substantial increase in their weights and
thereby force the weak learners to concentrate much more on
these samples. This might end up being detrimental to the
performance of Adaboost, as demonstrated convincingly by
[34].

2) Related work: There is a considerable amount of work
addressing this issue. For instance, Friedman et al. [35]
suggested a variant called ‘Gentle Adaboost’ by interpreting
boosting as an approximation to additive modeling on the
logistical scale. Rätsch et al. [36] showed how to regularize
Adaboost to handle noisy data: instead of achieving a hard
margin distribution (by concentrating on a few hard to be
classified samples, like that of Support Vectors), they proposed
several regularization methods to achieve a soft margin that
reduces the effect of outliers. Freund [37] suggested an algo-

$$r_n(x, y) = \frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{m} \mu_{y_{i,j}}$$

which is a function of the distance between a sample and its
class mean in the projected space. Then, if $w_i = 1/M$ denote
the uniform initial weights of all training samples $x_i$, the new

3The idea of different initial weights has been used in Asymmetric boosting
[42] in a different context where the missed detections are penalized more
heavily than false accepts. Hence the positive examples are weighted more
than the negative samples to start with. But, unlike our proposed method, all
positive samples are given the same weight, as are the negative samples. In
some sense this is like uniform weighting, with the weights different for the
two classes.
initial weights \( \tilde{w}_i \) are obtained by,
\[
\tilde{w}_i = w_i \exp(-\delta \epsilon_i)
\]  
where \( \delta \) is the factor controlling the importance of weights learned from (6). \( \delta \) is the total classification accuracy of kernel discriminant analysis on the training samples, which estimates the reliability of the learned weights. For instance, if the classification accuracy is very low (i.e. \( \delta \approx 0 \)), then \( \tilde{w}_i \) reduces to \( w_i = 1/M \) which is the same as the standard boosting setup. This new set of weights is then normalized to make it a distribution, and the classification function is learned using boosting as described in Algorithm 1.

A new cost function for \( \epsilon_t \): The error \( \epsilon_t \) which is minimized to select the weak learners \( h_t \) is, in its basic form (2), a function of the classification rate. However as mentioned before, the problem of outliers leads to a situation where the weights of certain samples become significantly higher than others. Hence to avoid this situation, there have been efforts to modify the cost function (2). The recent work by [40] addresses this issue by defining \( \epsilon_t \) as the relative entropy of the distribution of weights at the \( t^{th} \) iteration, \( D_t \), with that of the uniform initial distribution \( D \). But the problem with this cost is that, \( D_t \) need not be the best reference distribution since not all samples may be of the same quality to compare their current weights \( D_t \) with.

Since the undesirable condition caused by outliers is an uneven distribution of the sample weights, we propose to minimize the following cost function at every \( t^{th} \) iteration, instead of (2),
\[
g_t = \frac{(M - \sum_{i=1}^{M} D_t(i) y_i \hat{h}_t^{x_i})}{M} + \lambda R f_P(D_{t+1})
\]  
where the first term measures the error in classification, and the second term \( f_P(.) \) measures how sparse the distribution \( D_{t+1} \) (3) produced by the weak learner \( \hat{h}_t \) will be. \( \lambda_R \) is a regularization parameter. From the study of the problem of outliers [34], we deduce that \( f_P(.) \) should not be sparse. In other words, the weights should not be concentrated on only a few training samples. Hence we define
\[
f_P(D_{t+1}) = \sum_{i=1}^{M} I(D_{t+1}(i) < \lambda_{cost})
\]
where \( I(.) \) is an indicator function, and \( \lambda_{cost} \) is a threshold. Values of \( \lambda_R \) and \( \lambda_{cost} \) are learned using cross-validation, as explained in the Appendix. With these two modifications, we present our outlier-robust boosting algorithm in Algorithm 1.

At every iteration \( t \), the detection statistic \( g_t \) (8) is evaluated on all the weak learners \( \mathbb{H} \) (the context features \( h_j \), \( \forall j = 1 \) to \( M_1 \)) to select the one with the minimum cost (13). The set of weak learners selected up to \( T_1 \) iterations are then linearly combined to generate the final decision value \( g^* \) (16) that classifies a pixel \( x \) in the test image as a lane marking or otherwise. We empirically evaluated the efficacy of our algorithm on different UCI datasets [48] and present the results in the Appendix. At that point, we also discuss the convergence bounds of the algorithm.

Given: \( \{(x_i, y_i)\}_{i=1}^{M} \), where \( x_i \in \mathbb{R}^N \) is the training data, and \( y_i \in \{-1, +1\} \) its class label, and a pool of weak learners \( \mathbb{H} = \{h_j\}_{j=1}^{M_1} \).

Initialize the weight distribution of training samples \( D_1 \) from the weights learned from (7), i.e.
\[
D_1(i) = \frac{1}{M} \exp(-\delta \epsilon_i), \forall i = 1 \text{ to } M
\]
For iterations \( t=1,...,T_1 \):
(i) \( \forall h_j \in \mathbb{H} \), compute the classification error,
\[
E_{h_j} = \frac{M - \sum_{i=1}^{M} D_t(i) y_i h_j^{x_i}}{M}
\]
(ii) Compute an intermediate weight distribution, \( \tilde{D}_{t+1}^{h_j} \), which the weak classifiers \( h_j \in \mathbb{H} \) will produce,
\[
\tilde{D}_{t+1}^{h_j}(i) = \frac{D_t(i) \exp(-\alpha_t y_i h_j^{x_i})}{Z_t}
\]
where \( \alpha_t \in \mathbb{R} \), and \( Z_t \) is a normalization term to make \( \tilde{D}_{t+1}^{h_j} \) a distribution.
(iii) Select the weak learner \( \hat{h}_t \) that has the minimum cost \( g_t \) (8),
\[
\hat{\epsilon}_t = \min_{h_j \in \mathbb{H}} E_{h_j} + \lambda_R f_P(\tilde{D}_{t+1}^{h_j})
\]
\[
\hat{h}_t = \arg \min_{h_j \in \mathbb{H}} E_{h_j} + \lambda_R f_P(\tilde{D}_{t+1}^{h_j})
\]
(iv) Compute the new weight distribution,
\[
D_{t+1}(i) = \frac{D_t(i) \exp(-\alpha_t y_i \hat{h}_t^{x_i})}{Z_t}
\]
Output the final classifier:
\[
g^*(x) = \text{sign}(\sum_{t=1}^{T_1} \alpha_t \hat{h}_t^{x_i})
\]
which is a binary value corresponding to whether the test pixel \( x \) belong to lane markings or non-lane markings.

Algorithm 1: Proposed boosting algorithm that reduces overfitting and the effect of outliers in the training set.

C. Test phase: Detection (Localization)

We localize lane markings in a test image by computing the pixel-hierarchy descriptor \( f \) for all pixels, and then classifying them using (16). \( g^*(x) = 1 \) if the test pixel \( x \in O \) (lane markings), and \( g^*(x) = -1 \) otherwise. The computations needed to obtain \( f \) are performed efficiently using the concept of integral images and attentional cascade [32]. We provide the implementation details, and validation on images corresponding to daylight and nighttime road scenes in Section IV-A. The subsets of pixels in a test image classified as lane markings are then grouped and parameterized by a second order polynomial using the generalized Hough transform [49] as follows,
\[
L_i = p_2 x^2 + p_1 x + p_0
\]
where $L_i$ denote the $i^{th}$ lane marking, and $\vec{x}$ its horizontal coordinates in the image plane. Such a parameterization provides a coarse description of the structure of lane markings, and can differentiate between curved roads and straight roads. Having said that, it is interesting to see if a non-parametric representation using splines or a piece-wise model would provide more discriminative information about different types of lane markings. This set of $L_i$ denote the final detection (localization) result. An illustration is provided in Figure 4.

III. TRACKING AND LEARNING SOME VARIATIONS IN ROAD SCENE

We use the particle filtering framework [25] to track the localized lane markings in a video sequence. One challenging aspect of this problem comes from the non-availability of knowledge about motion patterns of lane markings. This is because the positions of lane markings in the image plane will depend on how the viewer (the camera/vehicle) is moving, and we do not have this information since we use only the visual input from a camera mounted on the vehicle. This is the main difference between our tracking formulation and existing lane tracking algorithms like [11]–[13], which either assume a motion model for the vehicle, or use information from inertial vehicle sensors to update the state transition model.

A. Formulation of Particle Filters to Track Lane Markings

To handle this situation, we propose a static motion model to represent the state of the particles. In other words, we always expect to see the lane markings at their initial position, and if there are any deviations from this hypothesis, we learn the causes for it. More formally,

\[
\text{State transition model} : x_t = x_{t-1} + u_t \quad (18)
\]

\[
\text{Observation model} : y_t = G(x_t, u_t) \quad (19)
\]

where $x_t$ is the state of the system, a 7 dimensional vector $[p_2, p_1, p_0, \vec{x}_{bl}, \vec{y}_{bl}, \vec{x}_{tr}, \vec{y}_{tr}]^T$, where $p_i$ are the coefficients of the polynomial characterizing a lane marking $L_i$ (17), and $[\vec{x}_+, \vec{y}_+]$ corresponds to the location of the bottom left and top right corners of the window enclosing the lane marking. $u_t$ corresponds to the system noise that has a fixed variance of the form $u_t = R_0 * U_0$, with $R_0$ being a fixed constant measuring the extent of noise, and $U_0$ a standardized random variable/vector. For instance, a larger value of $R_0$ makes the tracker search for the object in a bigger area around the location predicted by the particles. The particles are generated through sequential importance sampling, and we propose around 200 particles to approximate the system dynamics. The observation model (19) is characterized by $v_t$ which corresponds to the observation noise, and $G(.)$ is a function that defines the similarity of the object at the region predicted by the particles, with that of its true (initial) appearance. Let $H_1$ and $H_2$ denote all $H_c$ points sampled uniformly along the reference particle state, and the proposed particle state respectively. Let $H_i = \sum x_\in H_i I(g^*(x) = 1); i = 1, 2,$ where $I(.)$ is an indicator function. We then compute the similarity $G$ as,

\[
G(H_1, H_2) = \frac{\bar{H}_1}{H_c}(1 - \frac{1}{H_c} ||\bar{H}_1 - \bar{H}_2||_1) \quad (20)
\]

B. Learning the Road Scene using Tracked Parameters

We will now discuss what this tracking model conveys. As before, let the parameterized lane markings detected from the initial frame of the video sequence be denoted by $\{L_i\}_{i=1}^k$ (17), where $k$ denotes the number of lane marking groups present in that frame. For instance, in a two lane road there will be three lane markings denoting the left and right edges, and the central median. We define separate particle filters to track each of the $k$ lane markings, and then analyze the variations in its state parameters $[p_2, p_1, p_0, \vec{x}_{bl}(x), \vec{y}_{bl}(x_{}), \vec{x}_{tr}(x_{}), \vec{y}_{tr}(x)]^T$ to understand the causes behind it.

From now on, let us consider one such lane marking and analyze how to interpret its tracked parameters, though this discussion is valid for all lane markings. We also assume that the road is (piecewise) flat, since otherwise the presence of slopes can lead to sudden appearance/disappearance of lane markings that are hard to track. Let the number of frames in the video sequence where the lane marking is successfully tracked be denoted by $\bar{N}$, and the tracked parameters over all the $\bar{N}$ frames be represented by $[p_2, p_1, p_0, \vec{x}_{bl}(x), \vec{y}_{bl}(x), \vec{x}_{tr}(x), \vec{y}_{tr}(x)]^T$, $i = 1, 2, ... \bar{N}$. Let $var(.)$ denote the variance of a set of normalized observations of a state variable, computed at $k'$ equal intervals $\{\bar{N}_j\}_{j=1}^{k'}$. We now analyze the variance of each of the seven state parameters to infer the changes in the road scene. We do this in each of the $k'$ intervals. After every such interval, the reference particle state is updated with that of $H_2$ in that interval with the largest $G$ (20). Let us now consider the first interval, for instance.

1) Static world: If the lane markings are present in almost the same location in the tracked frame as their initial position, then there will not be a substantial variation in any of the seven tracked parameters. Formally, let $p_i = [p_i^1, p_i^2, ..., p_i^{\bar{N}}]$ and if

\[
var(p_i) < \xi_i, \forall i = 0, 1, 2 \quad (21)
\]

it implies that, irrespective of speed, the vehicle is maintaining its relative distance with respect to the lane markings (i.e.

\footnote{The detector is run again if the tracking error, $G(H_1, H_2) < \xi_{th}$, where $\xi_{th}$ is a threshold learned using cross-validation. $\xi_{th} = 0.55$ in these experiments.}
negligible lateral motion), and the road structure is also remaining constant (i.e., a straight road remains straight, and a curved road remain curved). $\xi_t$ is a threshold learned from cross-validation, and $\xi_t = 20$ in these experiments.

2) Change in lateral motion of vehicle: If there are variations only in the first ($p_1$) and zeroth ($p_0$) order coefficients of the parameterized lane marking, i.e.,

$$\left[\text{var}(p_1) > \xi_t\right] \lor \left[\text{var}(p_0) > \xi_t\right] \tag{22}$$

then this is due to the lateral motion of the vehicle with respect to the lane marking, while the road structure remains the same. Specifically, increase in the value of $p_0$ will be caused by the lateral motion of vehicle rightwards of the lane markings, and a decrease in $p_0$ is due to a leftwards lateral movement.

3) Change in road geometry: The second order coefficient ($p_2$) provides some information about the road geometry. For instance, if the road is straight, so will be the lane markings, and hence $p_2$ will be close to zero. If the road begins to curve, the change in the second order coefficient will get significant. Similar variations occur when a curved road becomes straight. Hence, if

$$\text{var}(p_2) > \xi_t \tag{23}$$

then it might be due to changes in the road geometry. This, when coupled with variations in $p_1$ and $p_0$ (22), can be used to infer simultaneous changes in lateral motion of the vehicle.

4) Change in traffic pattern ahead of vehicle: If there is a significant variation only in (any of) the four boundary points of the area enclosing the lane markings, $\{x_{bl}, y_{bl}, x_{tr}, y_{tr}\}$, we analyze the pixels $x$ belonging to the missing area (say, $R_m$) for (16). If

$$\sum_{x \in R_m} I(g^*(x) = 1) < M_4/2 \tag{24}$$

then we classify $R_m$ to belong to non-lane marking. $I(\cdot)$ is an indicator function, and $M_4$ is the number of pixels uniformly sampled in $R_m$. This can be used to alert the driver about the traffic pattern ahead of the vehicle. On the other hand, if there are variations in any of the $p_i$ also, the change in the area of bounding box might be due to the change in road geometry as well. Hence, the analysis of (16) in the missing region $R_m$ provides some information on occluding objects.

We tested our hypotheses by collecting video sequences pertaining to the above four scenarios. Sample results from the tracked sequences illustrating our learning approach are given in Figure 5. We present the results of our experiments in Section IV-B. It is an interesting future work to study the scope of learning applications when the state information of all lane markings are used jointly, and in reasoning non-flat roads using appropriate logic.

IV. EXPERIMENTS

We first evaluate the proposed detection algorithm on day and night time images in Section IV-A, and then discuss the learning applications using our tracking model in Section IV-B.

A. Detection of Lane Markings

We tested our outlier-robust boosting classifier (16) on road images collected during both day and night, over a period of twelve months. Separate classifiers were used for grayscale and color images (since it changes the information contained in $F = \{F_i\}_{i=1}^{M_5}$). We collected a set of 400 images, for both daytime and night time scenarios, and divided them into 5 equal sets. The overlap between these sets was kept to

5We will share our datasets for detection and tracking upon request.
a minimum of around 25%, in terms of the type of road scene, to study the generalization ability. For each of the five trials, one set of images was used for training, and the other four for testing. The detector processes 240 × 320 images at 15 frames per second, on a 4 GHz processor. We then counted the fraction of lane marking pixels in the regions corresponding to boosting results (which intersects with those hand-marked by the user) to determine correct detection rate, and counted the points that are not marked by the user which have been classified as the lane marking class by the algorithm to compute the false positive rate.

Based on this criterion, we present the performance curves in Figure 6 and study the role of spatial context and outlier robust boosting. Accuracy in terms of the position error of detected lane markings is given in Table I. Since there are no standard datasets for this problem, we implemented another machine learning approach based on [13] which learns a single classifier using support vector machines and artificial neural networks trained on both intensity images and contextual features $f(1)$. Best performing kernel parameters were used, and the same experimental setup was followed. From these results, we make the following observations,

![ROC curves for lane marking detection](image)

**Fig. 6.** ROC curves for lane marking detection: comparing different learning methods on an internally collected dataset of 400 day/night-time road images using a 5 fold cross-validation. The detection results correspond to pixel error of detected lane markings within a 3 × 3 neighborhood around the true pixel location.

### TABLE I

**Detection accuracy of Algorithm 1 with context in terms of the position error in the location of detected lane markings. The results show the position error in terms of neighborhood windows around the true lane marking locations in which the detection results occur. Performance across different false positive rates are given.**

<table>
<thead>
<tr>
<th>False positive rate</th>
<th>0</th>
<th>0.1</th>
<th>0.2</th>
<th>0.3</th>
<th>0.4</th>
<th>0.5</th>
<th>0.6</th>
<th>0.7</th>
<th>0.8</th>
<th>0.9</th>
</tr>
</thead>
<tbody>
<tr>
<td>$1 \times 1$</td>
<td>0.85</td>
<td>0.88</td>
<td>0.915</td>
<td>0.94</td>
<td>0.956</td>
<td>0.968</td>
<td>0.979</td>
<td>0.983</td>
<td>0.995</td>
<td>1</td>
</tr>
<tr>
<td>$2 \times 2$</td>
<td>0.872</td>
<td>0.895</td>
<td>0.92</td>
<td>0.948</td>
<td>0.967</td>
<td>0.975</td>
<td>0.982</td>
<td>0.993</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$3 \times 3$</td>
<td>0.91</td>
<td>0.925</td>
<td>0.942</td>
<td>0.957</td>
<td>0.976</td>
<td>0.98</td>
<td>0.99</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$4 \times 4$</td>
<td>0.932</td>
<td>0.94</td>
<td>0.951</td>
<td>0.964</td>
<td>0.985</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$5 \times 5$</td>
<td>0.935</td>
<td>0.947</td>
<td>0.958</td>
<td>0.965</td>
<td>0.985</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

1) Spatial context information helps the detection process. This can be seen from the performance curves with the contextual features (1) and those with only the intensity image;

2) The proposed method for outlier robustness (Algorithm 1) improves detection accuracy of Adaboost. Boosting methods perform better than methods which learn single classifiers, like SVM and neural networks.

These results, overall, support the intuition behind using boosting to learn contextual information for lane marking detection.

1) **Computations involved in determining $f$:** Given the required image representations $F = \{F_i\}_{i=1}^{M}$ pertaining to the original intensity image, edge map (from Canny operator [29]), and texture responses [30], we use the integral images concept [32] to compute the contextual features $f$. We perform an one-time operation to obtain the integral image $I^*$,

$$I^*(a, b) = \sum_{a' \leq a, b' \leq b} F_i(a', b')$$  \hspace{1cm} (25)

where $(a, b)$ denote the pixel location, using which all computations of a rectangular region in Haar-filter can be obtained using the knowledge of $I^*$ belonging to the four corners of rectangle. An illustration is provided in Figure 7. Detection across scale is performed by using different sized Haar-filters, rather than analyzing the image across multiple resolutions. Hence we obtain real-time performance, in line with the first boosting application to object detection [32]. Further increase in computational speed is possible by reducing the image resolution.

![Computing the contextual features](image)

**Fig. 7.** Computing the contextual features $f$ using Integral images [32]. Given an image representation $F_i$, to compute the cumulative information within the region $D$, we only need the value of $I^*$ for the four corner points 1, 2, 3 and 4. The information can be computed according to the pattern of Haar-filters.
B. Learning the Road Scene Variations

We now evaluate our hypotheses about learning variations in the road scene from the tracked model parameters. We collected video sequences pertaining to lateral motion of the vehicle, road curves, and static world models discussed in Sections III-B1 to III-B3. The lane markings are detected (localized) and parameterized (17) in the first frame of the videos using the approach presented in Section II, and then tracked using the particle filtering framework by assuming a static motion model for the lane markings (Section III). The tracker processes 240 × 320 images at 25 frames per second, on a 4 GHz processor. Given in Table II are the statistics of the variance of polynomial coefficients \([p_2, p_1, p_0]\) under different scenarios. Five video segments were used for each of the three scenarios listed below. The numbers indicate how the variance of each parameter computed using all frames in a video varies, indicated by its mean and standard deviation across different videos.

<table>
<thead>
<tr>
<th>Scenario</th>
<th>Mean±standard deviation of the variance on different videos</th>
</tr>
</thead>
<tbody>
<tr>
<td>Static world</td>
<td>1.15±0.11, 0.95±0.08, 1.12±0.17</td>
</tr>
<tr>
<td>Lateral motion</td>
<td>1.85±0.25, 33.12±5.66, 25.93±6.01</td>
</tr>
<tr>
<td>Road geometry</td>
<td>45.22±12.4, 2.77±0.55, 3.91±1.2</td>
</tr>
</tbody>
</table>

**TABLE II**

Statistics of the variance of polynomial coefficients for the scenarios discussed in Sections III-B1 to III-B3.

Occlusion position of the lane marking Determined by its mean and standard deviation across different videos.

<table>
<thead>
<tr>
<th>Occlusion Position of the lane marking</th>
<th>Variance of the bounding box parameters and the polynomial coefficients</th>
</tr>
</thead>
<tbody>
<tr>
<td>Left</td>
<td>1.45 1.62 1.77 1.95 1.12 0.98 1.12</td>
</tr>
<tr>
<td>Center</td>
<td>1.43 1.52 1.67 21.95 1.08 0.92 1.22</td>
</tr>
<tr>
<td>Right</td>
<td>1.22 1.52 1.83 1.65 1.22 1.98 1.43</td>
</tr>
</tbody>
</table>

**TABLE III**

Statistics of the variance of bounding box locations and polynomial coefficients for the occlusion model discussed in Section III-B4.

We then compared our tracking model with the commonly used constant velocity model for vehicles in a particle filter framework [12], [13] and with Kalman filtering [11]. Only the visual inputs were used. We tested the tracking accuracy over 2000 hand-marked frames on both day and night images. We present the detection rate and the false positive rate in Table IV. The criterion used for correct detection was to check if at least TrD\% of tracked points overlap with the ground truth. If the detection is less than TrD\%, we consider it a mis-detection. Whereas the false positive is computed if at least TrF\% of the tracked result is not included in the ground truth. We used the following values for \((T rD, T rF) = \{(80, 20), (90, 10), (80, 10), (90, 20)\}\). Although there are many different ways to validate tracking algorithms [50], we chose this method mainly to understand the performance of our proposed tracking model (Section III). It can be seen that at these operating points the performance of our model is comparable with other models, with a significant reduction in the false positive rate.

**TABLE IV**

Comparison of different tracking models on a set of 2000 frames. Empirical computation times are between 30 FPS to 15 FPS when the number of lane markings change from 1 to 5, with around 25 FPS when there are three lane markings.

<table>
<thead>
<tr>
<th>Tracking model</th>
<th>Performance criteria (in %)</th>
<th>Mean±standard deviation of</th>
</tr>
</thead>
<tbody>
<tr>
<td>Particle filtering</td>
<td>Correct tracking rate</td>
<td>False positive rate</td>
</tr>
<tr>
<td></td>
<td>82.1±3.5</td>
<td>22.3±3.2</td>
</tr>
<tr>
<td>Kalman filtering</td>
<td>76.5±3.7</td>
<td>32±4.8</td>
</tr>
<tr>
<td>Ours</td>
<td>83.2±3.1</td>
<td>15.8±2.5</td>
</tr>
</tbody>
</table>

**V. Conclusion**

Throughout this work we have studied the utility of learning approaches for detection and tracking of lane markings, using visual inputs from a camera mounted in front of a vehicle. We illustrated the advantages of modeling spatial context information through an outlier-robust boosting formulation, and inferring some variations in the road scene from the statistics of tracked model parameters under a static motion model for the lane markings. Without any assumptions on the road structure, or the motion pattern of the vehicle, we demonstrated some results on challenging daylight and nighttime road scenes.

At the core of our approach is the importance placed on the quality of data. Although our data for training and testing had several non-common exemplars, there can be instances such as foggy or rainy road conditions where the visual inputs alone are insufficient to detect lane markings. An illustration is provided in Figure 8. Hence, in order to obtain robust performance under varied road conditions, one could use complementary information from different sensing modalities such as the vehicle’s inertial sensors, GPS information and models for road geometry. Towards that end, we hope that the results from this study will provide some insights into the capabilities of learning contextual information from visual data.
A. Outlier Robustness of Adaboost - Discussion

We now analyze the iteration bounds of the proposed outlier robust boosting algorithm (Algorithm 1) in converging to the output hypothesis with optimal classification margin for the training data. For the class of boosting algorithms that study outlier robustness by modifying the cost function to achieve balanced weight distribution, results pertaining to the maximum achievable margin, and the number of iterations required for it were established by [40], [51]. Specifically, these results apply to methods where the cost function pertaining to weight distribution of samples is generally expressed as the relative entropy between the predicted weight distribution and the desired weight distribution, say $D^\star$.

We now adapt the results of [51] by rewriting our proposed cost function $\bar{f}_P$ (9) in terms of relative entropy follows,

$$f_P(D_{t+1}) = \frac{1}{\lambda_{\text{norm}}} \sum_{i=1}^{M} \bar{D}_{t+1}(i) \log \frac{\bar{D}_{t+1}(i)}{D^\star(i)}$$

where $\bar{D}_{t+1}$ and $D^\star$ are obtained by transforming the predicted weight distribution $D_{t+1}$ and desired weight distribution $D^\star$ (that penalizes sparse non-zero weights using the parameter $\lambda_{\text{cost}}$) as follows: $D^\star(i) = 1/M, \forall i = 1 \text{ to } M$, $\bar{D}_{t+1}(i) \approx 0, \forall i \text{ s.t. } D_{t+1}(i) \geq \lambda_{\text{cost}}$, and $\bar{D}_{t+1}(i) \approx 1/M', \forall i \text{ s.t. } D_{t+1}(i) < \lambda_{\text{cost}}$. $M' < M$ is the number of samples for which $D_{t+1}(i) < \lambda_{\text{cost}}$, and $\lambda_{\text{norm}}$ is a normalization constant whose value equals $M/\log \frac{3/M'}{3/M}$. When $\bar{f}_P$ is used in (13) instead of $f_P$, the optimization problem obtains the form for which the convergence results of [51] apply (since, the main difference between our method and [40], [51] is in the definition of the two distributions whose relative entropy is being computed).

Hence the proposed boosting algorithm terminates after at most $O(\frac{2}{\nu^2} \log(M/\nu))$ iterations with a convex combination $g^\star$ (16) that is at most $\Delta$ below the optimum classification accuracy $\Delta_1$ (available to the system). $\nu$ is a capping parameter that handles hard-to-be classified samples using soft margins. The effect of parameters $\Delta_1$ and $\nu$ on the classification accuracy are studied empirically in the following section.

B. Empirical evaluation

We used ten UCI benchmark datasets [48] to evaluate the proposed boosting algorithm. The data comes in 100 predefined splits, categorized into training and testing sets. For each split, we used 5-fold cross-validation to select the best kernel and its parameters, and the regularization parameters $\lambda_R$ and $\lambda_{\text{cost}}$ (13). This leads to 100 estimates of the generalization error for each dataset. The means and the standard deviations are given in Table V. We experimented with three types of Mercer Kernels, namely - Gaussian RBF $k(x_i, x_j) = \exp(-||x_i - x_j||^2/\sigma_c)$, polynomial $k(x_i, x_j) = (x_i, x_j)^d$ and sigmoid $k(x_i, x_j) = \tanh(c(x_i, x_j) - \delta_c)$, where $x_i$ and $x_j$ are a pair of data points. For each dataset, without the loss of generality, the best performing kernel (7) was used since this step needs to be done separately for every experiment.

It can be seen from Table V that our algorithm gives better performance when compared with the existing approaches on most datasets, and is close to the best algorithm on the others. Based on this study, we have three observations,
1) The weight learning process depends on the classification accuracy obtained from kernel discriminant analysis (the parameter $\delta$ in (7)). It would be interesting to see how the results vary when the bag of kernels is increased, and when a classifier better than kernel discriminant analysis is used;
2) The individual effect of the two components of our algorithm is studied in Table VI. It can be seen that the cost function argument performs slightly better than weight learning, while when used jointly, they produce the least generalization error;
3) Finally, the modifications suggested in our algorithm can be used in tandem with existing methods that focus on other aspects of boosting in handling outliers.

### TABLE V

<table>
<thead>
<tr>
<th>Datasets</th>
<th>Ours - weighting only</th>
<th>Ours - cost function only</th>
<th>Ours - Algorithm 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Banana</td>
<td>10.6 ± 0.3</td>
<td>10.4 ± 0.5</td>
<td>10.1 ± 0.3</td>
</tr>
<tr>
<td>B. Cancer</td>
<td>26.5 ± 3.2</td>
<td>26.45 ± 3.0</td>
<td>26.2 ± 3.2</td>
</tr>
<tr>
<td>Diabetes</td>
<td>24.5 ± 1.2</td>
<td>24.5 ± 1.2</td>
<td>24.5 ± 1.2</td>
</tr>
<tr>
<td>German</td>
<td>23.9 ± 1.1</td>
<td>23.6 ± 0.9</td>
<td>23.4 ± 1.1</td>
</tr>
<tr>
<td>Heart</td>
<td>17.2 ± 2.2</td>
<td>17.15 ± 2.2</td>
<td>16.9 ± 2.2</td>
</tr>
<tr>
<td>Ringnorm</td>
<td>1.8 ± 0.2</td>
<td>1.8 ± 0.2</td>
<td>1.65 ± 0.2</td>
</tr>
<tr>
<td>F.Solar</td>
<td>34.1 ± 1.2</td>
<td>33.75 ± 1.2</td>
<td>33.7 ± 1.2</td>
</tr>
<tr>
<td>Thyroid</td>
<td>4.7 ± 1.6</td>
<td>4.7 ± 1.6</td>
<td>4.6 ± 1.1</td>
</tr>
<tr>
<td>Titanic</td>
<td>21.5 ± 1.0</td>
<td>21.5 ± 1.0</td>
<td>21.5 ± 1.0</td>
</tr>
<tr>
<td>Waveform</td>
<td>9.5 ± 0.5</td>
<td>9.5 ± 0.1</td>
<td>9.12 ± 0.5</td>
</tr>
</tbody>
</table>

#### REFERENCES


Intelligence research program at the Office of Naval Research before returning to NIST.

4) Prof. Rama Chellappa received the B.E. (Hons.) degree from University of Madras, India, in 1975 and the M.E. (Distinction) degree from Indian Institute of Science, Bangalore, in 1977. He received M.S.E.E. and Ph.D. Degrees in Electrical Engineering from Purdue University, West Lafayette, IN, in 1978 and 1981 respectively. Since 1991, he has been a Professor of Electrical Engineering and an affiliate Professor of Computer Science at University of Maryland, College Park. He is also affiliated with the Center for Automation Research (Director) and the Institute for Advanced Computer Studies (Permanent Member). In 2005, he was named a Minta Martin Professor of Engineering. Prior to joining the University of Maryland, he was an Assistant (1981-1986) and Associate Professor (1986-1991) and Director of the Signal and Image Processing Institute (1988-1990) at University of Southern California, Los Angeles. Over the last 30 years, he has published numerous book chapters, peer-reviewed journal and conference papers. He has co-authored and co-edited books on MRFs face and gait recognition and collected works on image processing and analysis. His current research interests are face recognition, clustering and video summarization, 3D modeling from video, image and video-based recognition of objects, events and activities, dictionary-based inference, compressive sensing, and hyper spectral processing.

Prof. Chellappa has received several awards, including an NSF Presidential Young Investigator Award, four IBM Faculty Development Awards, an Excellence in Teaching Award from the School of Engineering at USC, and two paper awards from the International Association of Pattern Recognition. He received the Society, Technical Achievement and Meritorious Service Awards from the IEEE Signal Processing Society. He also received the Technical Achievement and Meritorious Service Awards from the IEEE Computer Society. At University of Maryland, he was elected as a Distinguished Faculty Research Fellow, as a Distinguished Scholar-Teacher, (both university wide awards), received the Outstanding Faculty Research Award and the Poole and Kent Teaching Award for the Senior Faculty from the College of Engineering, an Outstanding Innovator Award from the Office of Technology Commercialization, and an Outstanding GEMSTONE Mentor Award. In 2010, he was recognized as an Outstanding ECE by Purdue University. He is a Fellow of the IEEE, the International Association for Pattern Recognition, the Optical Society of America and the American Association for Advancement of Science. He holds two patents.

Prof. Chellappa served as the associate editor of four IEEE Transactions, as a Co-Editor-in-Chief of Graphical Models and Image Processing and as the Editor-in-Chief of IEEE Transactions on Pattern Analysis and Machine Intelligence. He has also served as a co-guest editor for five special issues that appeared in the Proceedings of the IEEE, the IEEE Transactions on Image Processing, the IEEE Transactions on Circuits and Systems for Video technology and the International Journal of Computer Vision. He served as a member of the IEEE Signal Processing Society Board of Governors and as its Vice President of Awards and Membership. He has served as a General and Technical Program Chair for several IEEE international and national conferences and workshops. He is a Golden Core Member of the IEEE Computer Society and served a two-year term as a Distinguished Lecturer of the IEEE Signal Processing Society. Recently, he completed a two-year term as the President of the IEEE Biometrics Council.