Road Shape Classification for Detecting and Negotiating Intersections

Christopher Rasmussen
Department of Computer & Information Sciences
University of Delaware
Newark, DE 19716

Abstract

This paper presents an approach to visually classifying the high-level geometry of the road ahead of a vehicle as a section *with continuous curvature parallel edges or an* intersection *containing right-angled legs. The default behavior of the system is snake-based tracking of parallel road edges for curvature estimation. A separate process segments the road surface from the background using color appearance characteristics, then classifies the segmented road shape in a rectified view as either a standard section or one of several intersection types ("four-way," "T," "right angle," etc.). Recognition of the proper shape class of the approaching road is a prerequisite for switching between shape templates in order to successfully track road edges as the vehicle travels through intersections.*

1. Introduction

Most work on road following for autonomous driving and driver assistance (e.g., lane departure warning) assumes the the vehicle is traveling on a highway-like road defined by parallel edges with some curvature, allowing the road shape to be estimated as a single, low-dimensional state [1, 2, 3]. Urban driving is in many ways more difficult than highway driving, but one factor preventing the direct application of techniques developed for highway driving is the many intersections encountered in city driving which cause large, often bilateral gaps in lane lines and road edges.

Detecting intersections robustly allows a steering system to switch to a more appropriate geometric model of the road edges in order to continue tracking successfully. Driver assistance and navigation modules may also use the knowledge that an intersection of a particular type is approaching to register the visual scene with GPS map data for more accurate localization. Sign detection and recognition modules may be invoked or given more cycles near intersections in order to take advantage of the proliferation of semantic information often found in their vicinity. Finally, algorithms for detecting crossing pedestrians and cars should be given greater weight due to the prevalence of such hazards in these areas.

To date there has not been a large body of work on intersection detection. In [4], the authors detect a single fork in the road with virtual views aligned with the predicted orientation of the branches. Though detection works at low speeds, they have trouble navigating through the intersection. They suggest that active camera methods will help with this. There has been some similar work on detecting forks in color-segmented roads [5]. Both of these approaches seem to favor detecting fork-type intersections on relatively narrow paths, rather than grid-like urban street intersections.

A paper with more similarities to the approach described here is [6]. The objective of their work was to guide a small robot around a table-top model of an urban environment by following streets and turning appropriately at intersections. Color and edge cues were both used to guide matching of rectified, live road views with a library of road shape templates. Simplifying the problem was the availability of an *a priori* map of the toy world annotated with intersection types to help with prediction, and minimization of typical vision difficulties by studio-type indoor lighting, uniform surface characteristics, and identical intersection geometry.

This paper presents a classification approach to intersection detection that guides switching of the road shape model to best reflect the current visual environment and ensure continuous tracking. We describe a two-level method that efficiently and robustly combines edge- and region-based road following algorithms with higher level shape analysis using a multi-category classifier. In order to afford an unoccluded wide-angle view for better intersection classification at closer distances and to keep more road edges in view while cornering, we use a wide-angle polycamera [7].

Figure 1: Camera arrangement

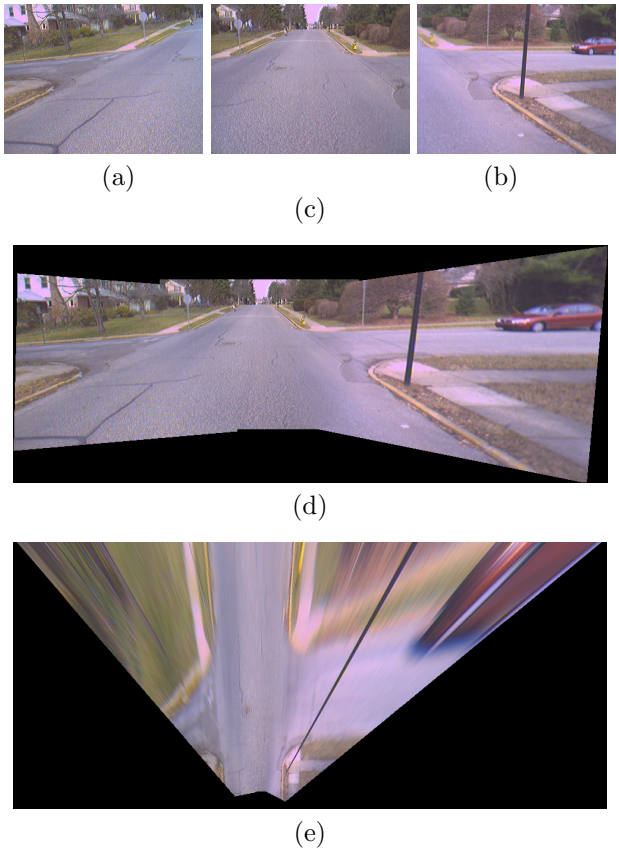
The organization of the rest of the paper is as follows. First, we will discuss the default mode, a strongly model-based, snake-type road edge tracker which operates directly on the image. Second, we will review an appearance-based method for road segmentation (which can be used to initialize edge-based trackers). Third, we will discuss details of the methods we have tested to detect different types of intersections, and the results we have obtained. Finally, we will talk about some ongoing work we are carrying out.

2. Camera and Road Geometry

A 3-camera polycamera was mounted on the roof of a vehicle at a height h of 1.9 m above the road surface and a tilt angle of $\theta = 10°$ below level. Camera resolution was 320×240 with 16 bits of color per pixel. The horizontal field of view (FOV) obtained for each camera after calibration [8] was 42° and the vertical FOV was 32°, with very little radial distortion. The combined horizontal FOV of the polycamera was about 93°. The camera configuration is shown in Figure 1.

The left and right camera centers were each separated from the middle camera center by about 0.1 m, yielding a small enough minimum working distance [7] to construct good panoramic mosaics given the typical distribution of object depths in road scenes. The homographies $\mathbf{H}_{L\to C}$ and $\mathbf{H}_{R\to C}$ for the left-to-center and right-to-center image transformations, respectively, were computed with a linear algorithm from manually chosen point correspondences [9]. The linear blending technique for seam removal described in [10] was used to composite the images. Three sample camera images of an intersection and the mosaic made from them are shown in Figure 2(a-d).

Assuming that the road surface is planar, we can compute a rectifying homography $\mathbf{H}_{C\to T}$ that maps the center camera image to a "bird's-eye" point of view. By first transforming the lateral camera images to the center coordinate frame and then applying $\mathbf{H}_{C\to T}$, we obtain a rectified mosaic of the road scene. An example is given in Figure 2(e); the scale is 0.5 m per pixel. Some departures from the planarity assumption are visible in Figure 2(d): straight across the intersection the road ascends slightly, and the intersecting road descends to the left. These altitude changes introduce distortions into the rectified mosaic, as do passing cars that occlude road edges.



(a) (b) (c) (d) (e)

Figure 2: Sample polycamera road images, mosaic, and rectified mosaic

3. Edge-based Road Tracking

We can use the mosaic image described in the previous section to perform traditional snake-based road edge tracking [1, 2, 3]. The advantage of the polycamera's wide-angle view in this domain is an enhanced ability to "see" the left and right road edges, both for wider roads in general and in situations where road curvature is great enough that one edge tends to leave the field of view.

Formally, a vehicle-centric coordinate system is chosen so that $+Z$ is forward with respect to the direction the vehicle is pointing, $+X$ is right, and $+Y$ is up. The X position of the center of the road is modeled as a quadratic in the ground plane such that

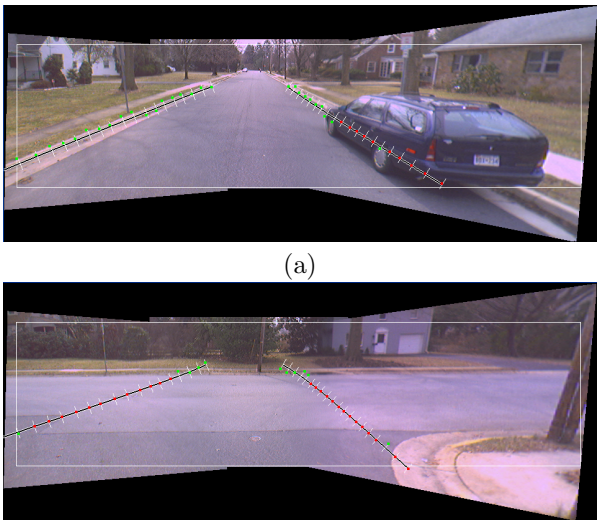
$X_C(Z) = aZ^2 + bZ + c$. Given the width of the road w and the inverse rectifying homography $\mathbf{H}_{T\to C}$, we can immediately derive the image locations x, y of the left and right road edges for any Z.

For urban/suburban driving environments, it is typical to assume that there is a sharp intensity contrast distinguishing the road surface from the background, or a painted line between road lanes. Initial values for the road shape model may be derived from region segmentation methods such as those described in a later section. In subsequent iterations, the tracker predicts left and right image intensity boundaries by projecting the current estimate of the road shape into the image, and measures the discrepancy to update its model. A set of N depth values $\{Z_i\}$ in the range $[Z_{near}, Z_{far}]$ are chosen such that their image projections $L = \{(x_{L_i}, y_{L_i})\}$ and $R = \{(x_{R_i}, y_{R_i})\}$ are roughly equally spaced vertically when $a = b = c = 0$. For each i in L and R, Sobel edge detection is performed along a line of length l bisected by and orthogonal to the curve at that point. The strongest edge location whose edge strength in the direction of the search line exceeds a threshold t is taken to indicate the actual road boundary at that point. The sign of the apparent intensity difference between the road surface and the abutting non-road surface affects this calculation, so that light-to-dark or dark-to-light edges are preferred consistently.

The edge-based, planar road state model $\mathbf{X} = (a, b, c)$ is updated via a Kalman filter [11]. Each measurement $\mathbf{Z}$ consists of a vector of the found left and right edge locations $\{(\hat{x}_{L_i}, \hat{y}_{L_i})\}$ and $\{(\hat{x}_{R_i}, \hat{y}_{R_i})\}$, respectively. w may also be estimated online as a slowly-varying term constrained by some upper and lower bound.

A missing data problem arises when an edge search line is outside of the image or no edge exceeding the threshold t is found. Rather than applying a full Expectation-Maximization-style algorithm [12] when this occurs, we simply fill in missing values in $\mathbf{Z}$ from the filter's predicted measurement $\hat{\mathbf{Z}}$.

This road tracking method is efficient and robust as long as most of its assumptions are satisfied. Disturbances such as passing cars or short breaks in the road edge due to driveways are generally overcome by the least-squares power of many combined edge observations from both sides of the road and along its length. An example of the edge tracker working despite a parked car occluding the road edge is shown in Figure 3(a). The edge tracker fails, however, when a preponderance of its edge measurements are bad, and this occurs regularly in urban environment at intersections (Figure 3(b)). In these situations, the road shape estimate is free to wander. For autonomous driving applications this is unacceptable, and for driver monitoring applications false alarms may result, with no guarantee that the road shape estimate will return to normal even after the vehicle completes the turn or clears the intersection.



(a)

(b)

Figure 3: Edge-based road tracker

4. Road Segmentation

Region-based method for road segmentation are often more robust than edge-based approaches [5, 13]. They are somewhat less sensitive to occlusions and work even without sharp edges (such as may occur on dirt roads).

Following [14], we frame road segmentation as a classification problem in which we wish to identify small patches in the field of view as either road or non-road on the basis of a number of appearance properties, or *features*, that we compute from them. Patches are manually labeled for a representative set of images, and a support vector machine (SVM) [15] is trained to learn a decision boundary in feature space. This model is used to classify pixels in novel images, from which we can derive road shape parameters directly by recursively estimating curvature, width, etc. from the edges of the road region and control steering accordingly (analogous to [3]). Due the relative expense of segmentation, however, it is more desirable to use the edge-based method whenever possible.

The features we use are an "independent" color histogram consisting of 8 bins per channel (8×3 total bins) computed over 31×31 subimages. We have found this histogram technique to be more feasible for small

subimages than a standard joint color histogram over all three channels, while achieving comparable performance. 31×31 subimages also outperformed smaller sizes such as 15×15 and 7×7, though they do tend to smooth the segmented road edges somewhat. Integrating other segmentation cues such as texture and structure is possible, but [14] found that color histograms alone are very strong cues and most likely sufficient for the purposes of this paper.

For training, 31 video frames were chosen at equal intervals from a sequence of 6,150 frames collected at 15 fps over about 7 minutes of driving in a residential neighborhood of Newark, DE. This sampling percentage of about 0.5% was selected based on the good results achieved in [14] with a smaller fraction for a much more diverse set of road types than the asphalt surface variants encountered here. The sequence was collected within a few minutes of noon on a partly cloudy mid-March day.

Road regions were manually marked in each camera image with polygons. Feature vectors were computed for each image at 10-pixel intervals vertically and horizontally ($\Delta x = \Delta y = 10$), with enough margin to ensure that histogram subimages remained entirely within the image. This resulted in $29 \times 21 = 609$ feature vectors per camera image. The SVM^{light} package [16] was used for SVM classification of the feature vectors with a radial basis function (RBF) kernel. A separate road color model was created for each of the polycamera's three cameras in order to avoid photometric calibration issues. With a value of 10 for the RBF spread parameter γ, 98.8% of the 18,879 training samples from each of the center and left cameras images were correctly classified, and 97.5% of the samples from the right camera were.

For efficiency, image neighborhoods are classified at intervals of $\Delta x = \Delta y = 5$, resulting in a decimated "road likelihood" image for each camera in which brightness is proportional to the strength of the SVM's road classification. Thresholding the road likelihood image appropriately results in a binary segmentation image for each camera. These are combined via the mosaicing method previously described to obtain a single segmentation mosaic. It should be stressed that in this work we do not carry out segmentation on the mosaic image, though this could also be done.

An example of road segmentation on the images from Figure 2 is shown in Figure 4 (the constituent camera images were not in the training set), along with rectified versions of the various mosaic types.

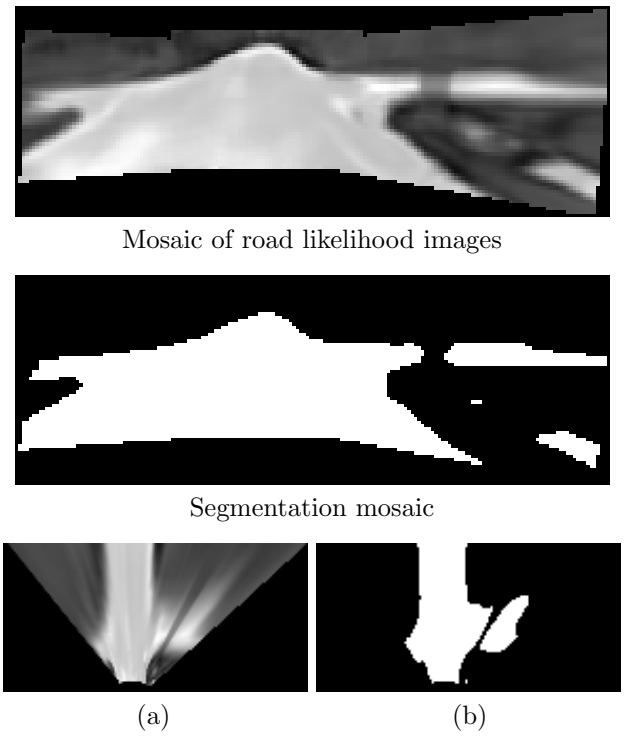


Figure 4: Segmented road scene: (a) Rectified road likelihood mosaic; (b) Rectified segmentation mosaic

5. Shape Classification

When intersections are encountered, the performance of the edge tracker may degrade or completely fail due to the lack of edge information where adjoining roads enter the one the vehicle is travelling on. If these intersections can be anticipated, the assumption of quadratically curved, parallel edges may be changed to a different template in order for tracking to continue successfully. Thus, we attempt to detect intersections beyond and slightly overlapping the most distant locations on the edge snakes.

A natural precursor to analyzing the high-level shape of the road ahead when it is unknown *a priori* is to use strictly appearance-based information to segment it from the background. We do this by running the color histogram classifier described in the previous section on the current polycamera image and examining the results. It is unnecessary to look at the entire image, as the ground plane assumption and knowledge of h, θ, and the internal camera calibration tell us where the horizon line is, above which there is no road. Moreover, our current estimate of the road width and the lookahead distance at which we hope to detect intersections place constraints on the left, right, and bottom portions of the image that we need to examine.

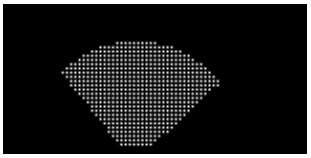


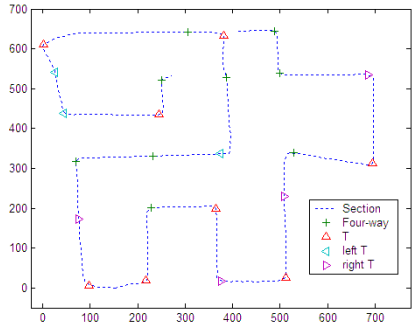
Figure 5: Mask used to learn rectified intersection shapes

In order to exactly specify these metric quantities, it is convenient to work with a rectified view of the road rather than the raw camera image. This also allows us to change the camera geometry relative to the road without relearning any shape classification, because we have moved to a common coordinate system.

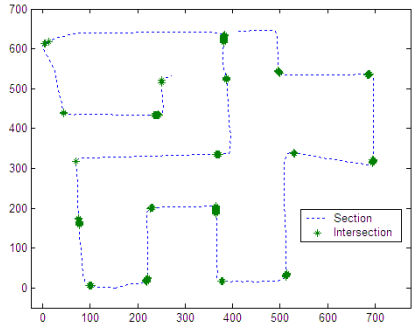
Let the vehicle be on a road that intersects at most one other road at a time at roughly a 90-degree angle. This leads to the following four road shape types: a single, possibly curved *section*, schematically represented by the | symbol (a bird's-eye-view of the road with the vehicle traveling up the page); and three *intersections*: the "four-way," "T," and "right angle," denoted by the symbols +, $\top$, and $\rceil$, respectively. The first general road shape, a *section*, is the geometric assumption made by the default edge tracker described in the first part of the paper. Rotations of T intersections result in three viewpoint-dependent subtypes corresponding to $\top$, $\dashv$, and $\vdash$, and right-angle intersections have left ($\lceil$) and right ($\rceil$) subtypes.

There are many possible methods for classifying the shape of the approaching road. An obvious one is a simple nearest-neighbor criterion: road likelihood templates corresponding to the various intersection types are scaled by the estimated current road width and correlated with a rectified view of the segmented road, and the best match is chosen to represent the current shape of the road. However, this approach is vulnerable to distortions in the apparent intersection shape due to departures of the road surface from planarity.

Our method for road shape determination is learning-based in order to quantify and compensate for violations of the initial assumptions. A set of segmented mosaics of road scenes are labeled by their section or intersection type, and a multi-category classifier is used to learn a model for them in order to categorize future scenes. Specifically, we use all pixels in the rectified road likelihood mosaic corresponding to scene locations less than or equal to than 25 m away from the camera radially, decimated by 2 both vertically and horizontally, as a feature vector. The mask for this set of pixels is shown in Figure 5; there are 604 per rectified mosaic.



(a)



(b)

Figure 6: Intersection shape data: (a) Plot of vehicle positions and true intersection locations (the start point is in the upper right and the scale is in meters); (b) Locations of frames classified as section vs. intersection by rectified shape (from testing data).

Vehicle speed and position were recorded at 1-second intervals with a GPS unit. All GPS-derived quantities were estimated using only points with a position dilution of precision (PDOP) of 8 or less. A total of 394 data points (95.9% of those recorded over the driving course) met this criterion. The total distance driven was 3.18 km, with a mean speed of 27.9 km/h ($\sigma = 11.1$). The course was fairly flat but not completely level: the range of altitudes was 8.7 m ($\sigma = 2.0$). Nearly all streets were straight, with two short, moderately curved sections. The path driven was planned to be non-self-intersecting and to achieve a fairly even distribution on the number of intersection types encountered and directions turned. In all, 18 turns were made: 9 right and 9 left. 6 intersections were driven straight through, for a total of 24 intersections traversed. Two intersection types were present along the course: + and $\top$. Figure 6(a) plots the vehicle's path with the true intersection locations and subtypes overlaid.

To label rectified mosaic frames for learning, we first manually indicated the center of each intersection in the GPS data set. Then we automatically labeled all video frames captured both less than or equal to 25 m away from an intersection subtype and *before* it was reached in the sequence (in order to ensure intersection visibility). All remaining video frames were labeled as sections. Of 6,148 rectified mosaics in the sequence, the distribution of labels was as follows: $|$ (5,184), $+$ (374), $\top$ (463), $\dashv$ (59), and $\vdash$ (68).

Training a binary SVM classifier with an RBF kernel ($\gamma = 0.01$) to simply distinguish between sections and intersections yielded an accuracy rate of 89.0% (averaged over 3 trials of training on $\frac{2}{3}$ of the data and testing on the remaining $\frac{1}{3}$). The classifications match ground truth fairly well, as can be seen in Figure 6(b). Performing SVM regression (again with an RBF kernel and $\gamma = 0.01$) to simulate a multi-category classifier (each label was designated with a integer in the range $[1, 5]$), and interpreting categorization in the nearest neighbors sense, an accuracy of 97.1% was achieved on the entire testing set. Of the 176 misclassified frames, 56 should have been labeled $\dashv$ and 65 should have been $\vdash$. There may have been too few examples of these types in the training set, but they are also intuitively the most difficult to distinguish.

Because our road shape classifier is keyed to a particular distance ahead along the road, it is properly a detector, and thus we can easily initialize an edge-based Kalman filter with a different shape profile at the correct location based on the current state of the parallel edge tracker before turning off the latter, and vice versa after the intersection has been passed. Because of the variability of corner profiles at intersections (i.e., sharp or rounded), region-based tracking methods may work best over these transitions. Angle, scale, and shift information as the vehicle performs turns may be derived from the rectified segmentation mosaics in a straightforward fashion. We are still experimenting with this component of the system.

6. Conclusion

We have described a system for road intersection detection via shape analysis of segmented polycamera images. The road shape classifier interacts with an efficient snake-type edge tracker to govern when it is turned on and off. While the vehicle is traveling through intersections, a region-based mechanism may be used to estimate road shape. Preliminary results are encouraging, but much work remains to be done. In particular, online methods for estimating departures of the ground from planarity would help reduce distortions and disambiguate the shape of intersections. Increased variability in time of day, season, and weather should also be included in the training data in order to obtain a more robust model of the road appearance.

Many other cues can also be used to reinforce intersection classification. For example, signs, traffic lights, and symbols painted on the road such as crosswalks and arrows. Along these lines, we are currently studying how changes in visual conditions affect the efficacy of different cues for road following applications.

References

[1] E. Dickmanns, "Vehicles capable of dynamic vision," in *Proc. Int. Joint Conf. on Artificial Intelligence*, 1997, pp. 1577–1592.

[2] B. Southall and C. Taylor, "Stochastic road shape estimation," in *Proc. Int. Conf. Computer Vision*, 2001, pp. 205–212.

[3] C. Taylor, J. Malik, and J. Weber, "A real-time approach to stereopsis and lane-finding," in *Proc. IEEE Intelligent Vehicles Symposium*, 1996.

[4] T. Jochem, D. Pomerleau, and C. Thorpe, "Vision-based neural network road and intersection detection and traversal," in *Proc. Int. Conf. Intelligent Robots and Systems*, 1995, pp. 344–349.

[5] J. Crisman and C. Thorpe, "SCARF: A color vision system that tracks roads and intersections," *IEEE Trans. Robotics and Automation*, vol. 9, no. 1, pp. 49–58, 1993.

[6] T. Kyriacou, G. Bugmann, and S. Lauria, "Vision-based urban navigation procedures for verbally instructed robots," in *Proc. Int. Conf. Intelligent Robots and Systems*, 2002.

[7] R. Swaminathan and S. Nayar, "Non-metric calibration of wide-angle lenses and polycameras," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 22, no. 10, 2000.

[8] J. Bouguet, "Camera Calibration Toolbox for Matlab," Available at `www.vision.caltech.edu/bouguetj/calib_doc/`. Accessed May 11, 2001.

[9] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, Cambridge University Press, 2000.

[10] R. Szeliski, "Video mosaics for virtual environments," *IEEE Computer Graphics and Applications*, vol. 16, no. 2, pp. 22–30, 1996.

[11] Y. Bar-Shalom and T. Fortmann, *Tracking and Data Association*, Academic Press, 1988.

[12] A. Dempster, N. Laird, and D. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *J. Royal Statistical Society B*, vol. 39, pp. 1–38, 1977.

[13] J. Fernandez and A. Casals, "Autonomous navigation in ill-structured outdoor environments," in *Proc. Int. Conf. Intelligent Robots and Systems*, 1997.

[14] C. Rasmussen, "Combining laser range, color, and texture cues for autonomous road following," in *Proc. Int. Conf. Robotics and Automation*, 2002.

[15] N. Cristiani and J. Shawe-Taylor, *An Introduction to Support Vector Machines and other kernel-based learning methods*, Cambridge University Press, 2000.

[16] T. Joachims, "Making large-scale SVM learning practical," in *Advances in Kernel Methods: Support Vector Learning*, B. Schölkopf, C. Burges, and A. Smola, Eds. MIT Press, 1999.