2012 IEEE/RSJ International Conference on
Intelligent Robots and Systems
October 7-12, 2012. Vilamoura, Algarve, Portugal

Mobile Robot Monocular Vision Navigation Based on Road Region and Boundary Estimation

Chin-Kai Chang Christian Siagian Laurent Itti

Abstract— **We present a monocular vision-based navigation system that incorporates two contrasting approaches: region segmentation that computes the road appearance, and road boundary detection that estimates the road shape. The former approach segments the image into multiple regions, then selects and tracks the most likely road appearance. On the other hand, the latter detects the vanishing point and road boundaries to estimate the shape of the road. Our algorithm operates in urban road settings and requires no training or camera calibration to maximize its adaptability to many environments. We tested our system in 1 indoor and 3 outdoor urban environments using our ground-based robot, Beobot 2.0, for real-time autonomous visual navigation. In 20 trial runs the robot was able to travel autonomously for 98.19% of the total route length of 316.60m.**

I. INTRODUCTION

Ability to navigate in one's environment is important for a fully autonomous mobile robotic system. One critical task in navigation is to be able to recognize and stay on the road. There are many proximity sensors that have been used to detect the road, such as laser range finder (LRF) [1], [2], stereo cameras [3], and Kinect [4]. However, these sensors have limitations. For one, Kinect, which utilizes infra-red technology, does not work outdoors, due to the presence of sunlight. Furthermore, proximity information, although it can be used to estimate the ground plane, it cannot directly recognize the shape and appearance of the road without the presence of bounding structures such as surrounding walls.

Monocular cues from a camera, on the other hand, have the versatility of being applicable in most environments, while readily encode these road information, which are the critical for autonomous navigation. As such it is important to develop robust monocular road recognition techniques to enhance the available proximity-based navigation systems.

There are a few main issues in using monocular vision road recognition. For one, road appearance varies from place to place, which makes it difficult to define what a road is. Consequently, many navigational approaches limit their working environments to just previously visited places through the teach-and-replay paradigm [5], [6]. These systems require the robot to traverse a specific environment during the training process before it can navigate on the same road during testing. In addition, often times they also need to be trained on many lighting conditions to be illumination invariance. This is because these systems use local corner/keypoint features such as SIFT [7] or KLT [8], which are illumination specific.

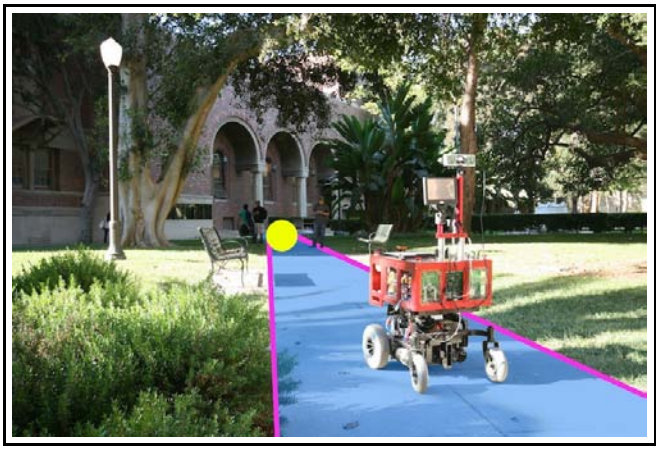

Fig. 1. Beobot 2.0 performing autonomous navigation in an unknown environment. There are two characteristics of the road that are exploited by our system. One is the similarity appearance within the road, color of the concrete, which can be compromised by shadows. The other cue is the vanishing point at the end of the road, noted by the yellow dot. The difficulty of vanishing point-based systems are their instability, as a small location shift can result in large shape changes on the road area closest to the robot, making it less suitable for navigation tasks.

There are, however, a set of approaches that do not require specific training for a particular environment. These systems try to recognize a road by exploiting its visual characteristics. For example, a system by Rasmussen etal [9] estimates the road as a triangular shape, and perform histogram differencing between the road region and its two flanking areas. The histogram is composed of a small number of bins obtained through k-means LAB color clustering on the image. Another system by [10] uses both color and texture features to cluster the road area, and the subsequent classification confidence values to estimate the extent of the road. [11] relies on the salient nature of the road with respect to its surrounding regions, and grows the road region from a saliency map. One problem that region-growing based systems face is that they are less robust to shadows on the road. Observe figure 1 for an illustration of these challenges.

In addition, there is also a group of techniques [12], [13], [14] that try to recognize the road by using the road vanishing point. In general the technique relies on detecting converging

C.-K. Chang is with Department of Computer Science, University of Southern California, Hedco Neuroscience Building - Room 10, 3641 Watt Way, Los Angeles, California, 90089-2520, USA. chinkaic@usc.edu

C. Siagian is with the Division of Biology, California Institute of Technology, Division of Biology 216-76, Caltech, Pasadena, California, 91125, USA. siagian@caltech.edu

L. Itti is with the Faculty of Computer Science, Psychology, and Neuroscience, Univesity of Southern California, Hedco Neuroscience Building - Room 30A, 3641 Watt Way, Los Angeles, California, 90089-2520, USA. itti@pollux.usc.edu

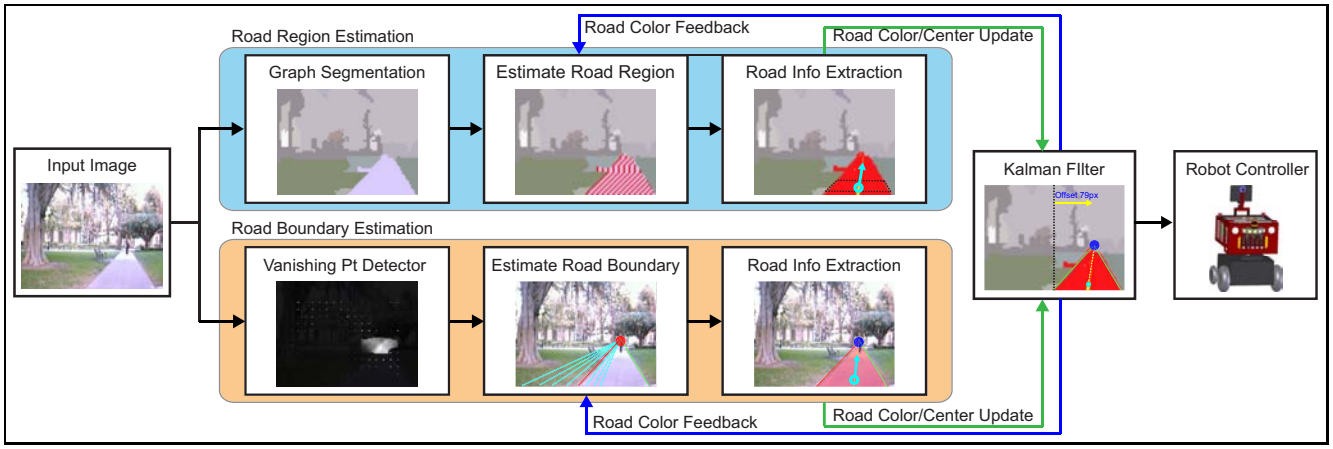

Fig. 2. Overall visual road recognition system. The algorithm starts by inputting the camera image to both the road region and road boundary estimation modules. The road region estimator utilizes a graph-based segmenter, while the road boundary estimator relies on a real time vanishing point detection. Both modules output road color average and the road center location, which are fused together by a Kalman Filter.

edge directions to vote for the most likely vanishing point. The biggest problem we found was that the output location can be unstable if there are several equally likely vanishing points. Furthermore, because the vanishing point is far from the robot, a slight deviation can result in a large change in heading direction, making it unsuitable for navigation. It is also important to note that most vanishing point-based approaches are not real time, and, thus, not tested on a robot for an autonomous navigational task.

Our novel contribution starts by presenting an autonomous navigation approach using only monocular cues, comprised of the two above-mentioned and opposing vision techniques. That is, we designed and implemented real-time and robust region appearance-based and vanishing point-based road recognition. We then propose a navigation framework that fuses both techniques to improve the robustness of the overall system. We tested the system using Beobot 2.0 [15] in one indoor and three outdoor campus environments to demonstrate its capabilities in unconstrained settings. In addition, we also analyze the benefit of using the individual cues in details, as well as comparing our system with other road recognition systems [9], [12].

These two systems represent the two most popular monocular road detection approaches available in the literature.

In the following section II, we describe our presented model, report the testing results in section III, and discuss the findings in section IV.

II. DESIGN AND IMPLEMENTATIONS

The overview of our vision navigation system is illustrated in figure 2. From the input image, we segment out the region that are considered to be a road (described in subsection II-A), while simultaneously estimate the road boundaries by computing the road vanishing point (described in subsection II-B). In section II-C, using a Kalman Filter, we then combine these two algorithms to produce a robust estimation of the road, which is used to create control policy for autonomous navigation.

A. Road Region Estimation

The input image size of 320x240 image is first down-sampled to 80x60 to speed up the process and to filter out spurious noise input. The road region estimator then segments the image to large contiguous regions using [16] and search for the segmented road region. The graph-based algorithm recursively merge image regions if the intra-region differences is lower than the inter-region difference plus a constant k (we set $k = 400$) Initially, the road center is selected from the center of the image, assuming the robot starts out on the road, facing the direction of the road heading.

This is done by creating a small search window W_{fixed} on the bottom center of the image (observe figure 3). The region that covers the largest area in W_{fixed}, by the number of pixels, is chosen to be the road region. The system then computes the road color by averaging the pixel values in the region, which will be used to update an appearance prior estimated the Kalman Filter.

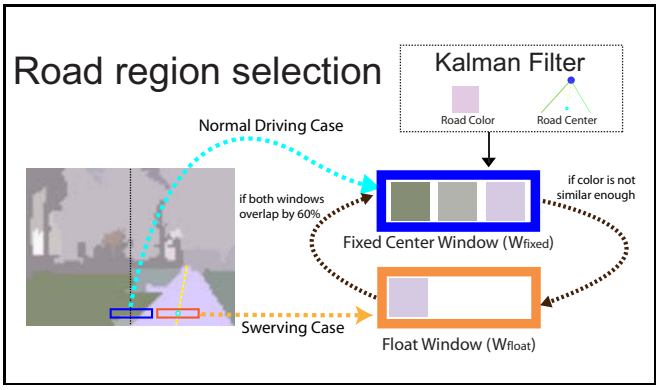

Fig. 3. Road region selection process. The estimator searches for the road region in the segmented input image using the center W_{fixed} window if the robot is moving steadily in the middle of the road. However, if the robot starts to swerve from the center of the road significantly, it then switches to W_{float}. The system switches back to W_{fixed}, when the robot successfully re-centers itself.

In addition, the estimator also computes the road region center because it indicates the direction of the road. This is done by estimating a slope of a line that bisects the bottom portion of the road region (observe the yellow trace in figure 3). The reason of estimating only using the bottom part is because it is in the immediate front of the robot, which is much more critical for determining the current motion command. The far end of the region will be considered in future time steps, when the computation is repeated.

After each frame, the estimator checks if the output road region is similar (in number of pixel, coordinate location, and color) to the one produced in the previous frame. In occurrences where the robot swerves too far away from the center of the road, there is usually a large deviation. In such cases, it switches to an identically sized window W_{float}, located at the previous road center estimation. The switch allows the system to react quickly to a dramatic robot movement. As the robot re-centers itself on the road, W_{float} keeps track of the movement, and the estimator only switches back to W_{fixed} when it sufficiently overlaps with W_{float}. By normally using W_{fixed}, the estimator provides a subtle bias to the select regions that are near the center of the image.

B. Road Boundary Estimation

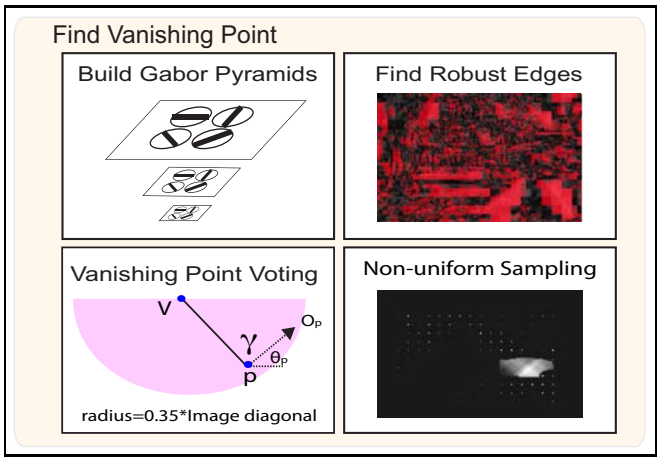

Fig. 4. Vanishing point detection module. The module first compute Gabor pyramids in 4 directions and 3 scales. It then computes the edge direction confidence scores to select only the most robust edges. The module then votes for vanishing point location using robust edges within .35*image diagonal below each respective vertical coordinate [12]. To speed up the process, the module performs a non-uniform sampling of the candidate locations. That is, it densely samples the locations around the vanishing points from the last 10 frames, as well as sparsely samples the rest of the locations.

Figure 4 illustrates the process of finding the road using vanishing point detection. This module is inspired by [12], which does not readily run real-time. We, on the other hand, optimize the implementation and significantly modify many parts of algorithm to make it run 100 ms/frame in our robot. One notable modification is to go from 36 orientations to a more realistic 4, which requires more robust processing throughout the algorithm for a comparable performance. In addition instead of only using static information, we utilize a temporal filtering process to increase robustness.

Using the input image, the estimator computes dyadic Gabor pyramids $G_{\alpha,s}$ for $N_{angle} = 4$ orientations α (equally spaced between 0 to 180 degrees), at $N_{scale} = 3$ scales s. For each orientation, the estimator averages out the responses (observe equation 1) from all scales to create a single map at the largest resolution.

$$\bar{G}_{\alpha}(i,j) = 1/N_{scale} \sum_{s=0}^{N_{scale}-1} G_{\alpha,s}(\frac{i}{2^s}, \frac{j}{2^s}) \qquad (1)$$

It then calculates the confidence value of the edge direction Gabor responses in each location by comparing how dominant the largest response $SA(1)$ is with respect to the second $SA(2)$:

$$Conf(i,j) = 1 - \frac{\bar{G}_{SA(2)}(i,j)}{\bar{G}_{SA(1)}(i,j)} \qquad (2)$$

The pixels with high confidence values are selected to vote for the vanishing point location.

For each location candidate V, the estimator computes the likelihood by accumulating scores from a local neighborhood of pixels in a half-disk region P. The contributing score of each voting pixel $p \in P$ is computed according to:

$$score(V,p) = \begin{cases} \frac{1}{1+\gamma * dist(V,p)} & \gamma \leq 8^\circ \\ 0 & \text{otherwise} \end{cases} \qquad (3)$$

with $\gamma = \angle(\overline{Vp}, \overline{O}_p)$ denoting the angle between line $\overline{Vp}$ and the dominant angle vector at p, illustrated in the Vanishing Point Voting sub-module in figure 4.

To speed up the process, we introduce a non-uniform sampling over the vanishing point locations. We densely sample the locations around (within $1/10$ of the image dimensions) the estimated vanishing points from the last 10 frames, and sparsely sample the rest of the image. Observe the Non-uniform Sampling sub-module in figure 4. On average, the system evaluates around only 3 percent (30 times speedup) of all possible locations.

The estimator then selects the highest scoring vanishing point location and finds its left/right road boundaries by computing the likelihood score for each candidate ray in five degree spacing. We propose two improved and more efficient methods: Gabor Response Variance Score (GRVS) and color difference (CD). Observe figure 5 for illustration.

First, we introduce a novel GRVS method, which measures the consistency of line l using only $N_{angle} = 4$ Gabor orientations. The technique creates N_{angle}-orientation bins B_{α} to store the Gabor response of every pixel on line l based on its highest response angle. The GRVS is then computed based on the bin with the highest count B_{max}:

$$GRVS(l) = \frac{(|B_{max}|/|l|)^2}{std(B_{max})} \qquad (4)$$

Here $|l|$ refers to the total number of pixels on line l, while $|B_{max}|$ and $std(B_{max})$ are the total number of pixels and Gabor magnitude standard deviation in bin B_{max}. The advantage of GRVS is that it does not require many Gabor

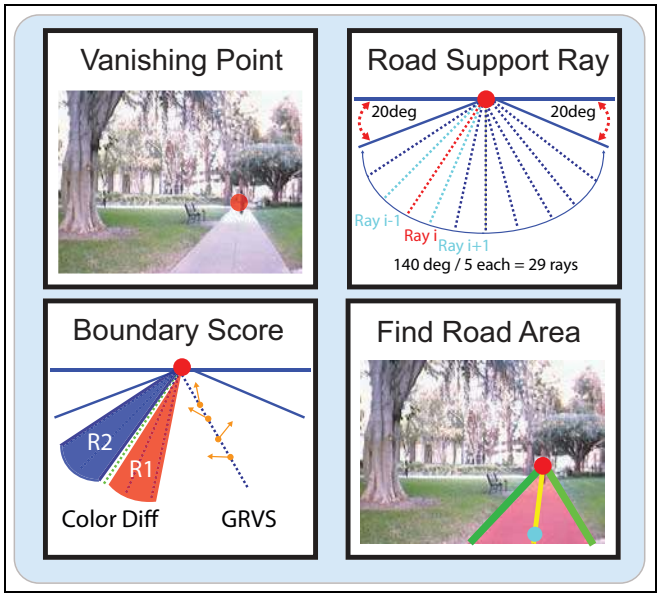

Fig. 5. Finding Road Boundaries. From the vanishing point, we extend ray with 5 degrees spacing and evaluate how likely each ray is a road boundary. Refer to text for details.

orientation bins to establish line consistency. The assumption is that if there exist a line, all the pixels on the line ought to have similar Gabor responses (indicated by low standard deviation), and orientation (by having a clear majority in bin counts) even if the dominant angle is not aligned with the vanishing point ray. Figure 6 illustrates the process.

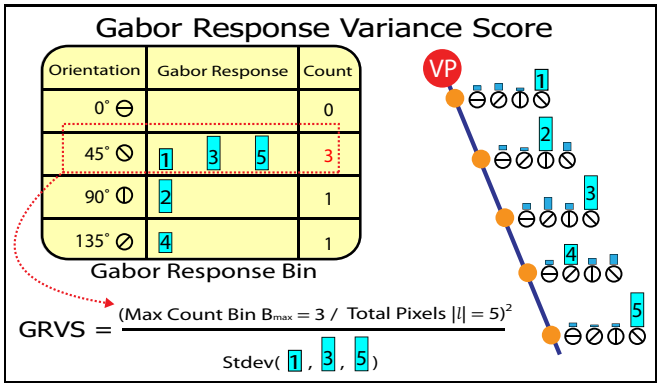

Fig. 6. Gabor Response Variance Score (GRVS). The Gabor responses of the points along a candidate road boundary line l are indicated by with cyan bars. The table to the right tabulates the total count of dominant angles for each orientation. Here, the $45°$ angle wins out, and the result is used to calculate $GRVS(l)$.

The color difference (CD) method calculates RGB color average of pixels within 20 degrees of each side of the ray, with the inner road region denoted as R_1 and the outer flanking area as R_2.

We also tested the system using HSV and LAB color space, and find no significant performance difference.

Observe the Color Difference sub-module in figure 4 for illustration. CD is computed using:

$$CD(R_1, R_2) = diff(R_1, R_2) - diff(R_{Road}, R_1) \quad (5)$$

The first term maximizes the contrast between the road and flanking areas R_1 and R_2, as measured by color euclidean distance. The second term, on the other hand, minimizes the color difference between R_1 and the current Kalman Filter estimated road color R_{Road}.

The GRVS and CD scores are multiplied to obtain the final likelihood score. We found that the combination of these factors are more robust than either one to handle the variety of road appearances.

The process is repeated to find the second road boundary before the module computes the average road color and the road middle point, located at the middle bottom of the area enclosed by the two boundaries.

C. Combining Road Region and Boundary Estimation

Both road region and boundary estimation modules outputs the road color average and road middle point state vector

$$X_k = \left[R_{\{r,g,b\}\ k} \quad x_k\right] \quad (6)$$

where $R_{\{r,g,b\}\ k}$ is RGB color, and x_k is the horizontal location of the middle point. The Kalman filter integrates these state vectors using a Zero-Acceleration prediction model because the robot tends to stay on the road center while moving forward and the road color appearance remain similar to previous frame.

The Kalman Filter estimation is then fed back to each respective module, thus indirectly incorporating information from one module to the other.

Finally, the overall system computes the distance between the image center the road center, and use it as a differential input for a PID robot heading control to align the robot to the center of the road.

III. TESTING AND RESULTS

We test the system accuracy in recognizing the center of the road in one indoor (HNB) and three outdoor (AnF, Equad, and SSL) environments. Observe figure 7 for example scenes and table I for pertinent information about each site. We carry out two different testing procedures: one to assess the accuracy of the system in recognizing the center of the road (section III-A), done offline and compared against ground-truth, while the second is for autonomous robot navigation (section III-B).

A. Visual Road Recognition

We manually drive the robot through the environments to collect testing videos. In each frame, we then annotate the left and right boundaries, and extending them to meet at the vanishing point. We then calculate the road center point using the technique explained at the end of section II-B.

On some frames, it is harder to clearly label the two sides of the road, in cases such as an intersection. For this experiment, we skip these frames altogether because they have more than one heading option. In autonomous navigation, whenever the robot has to make a decision near an intersection, it selects the nearest road to the estimated heading.

Fig. 7. Example images from each testing environment. We display 4 example figures for sites HNB, AnF, Equad, and SSL. Below each figure, we display the system's road segmentation estimation, denoted by the red shading. In addition, we also added the vanishing point estimation, denoted by the blue dot, with the green line extended being the road boundary. Also, we added a yellow line on the bottom of each image that points to the direction of the robot heading. As we can see, the system works on various road appearance, width, and under many lighting conditions. Furthermore, it is also able to overcome various challenges, such as pedestrians and sporadic shadows.

The error is defined as the deviation between the human annotated and system estimated road center point. In our robot camera setup, one pixel at the bottom of the image is equivalent to one centimeter. We report the experiment results in figure 8.

We include the results of our road region and boundary estimation, as well as the combined performance. In addition, we also compare our system with two other road recognition systems: [9], which primarily uses region color appearance to find the road, and [12], which detects the vanishing point in the image. Note that our system runs real time, 100 ms per frame, which is critical for autonomous navigation tasks. As a comparison, our implementation of [9] runs 780ms per frame, while [12] runs 43 seconds per frame. All experiments for this section is run on a 2.2GHz Intel Core2 Duo processor.

Overall, our full system is able to estimate the road center reliably in all testing environments as its average error is under 5 percent when compared to the road width. For example, in AnF, the average accuracy is 9.41cm, while the road width is 2.12m. Furthermore, our full system produces the best overall result, when compared to the individual algorithms within the system. Although the full system may not always produce the best result for every environment, it is the most stable when compared to the individual algorithms.

The road region estimator, which gives the best result in HNB and AnF, tends to perform optimally in environments with uniform road appearance. However, it is less effective on roads where there are more inconsistencies because of factors such as shadows. This case is illustrated by the example in the fifth column of figure 9. On the other hand, the road boundary estimator has higher tolerance to many road appearances, but often with lower accuracy, such as the slightly off right boundary detection in the forth column of figure 9. The overall system, however, balances these characteristics as it approach the accuracy of road region estimator in uniform roads, but with the robustness of road boundary estimator to provide usable navigation results even in very challenging conditions.

TABLE I
EXPERIMENT SITES INFORMATION

	HNB	AnF	Equad	SSL
Traversal Length	36.67m	105.75m	138.27m	326.00m
Total Frames	3919	7217	5371	8569
Road Width	1.5m	2.12m	3.1m	4.47m

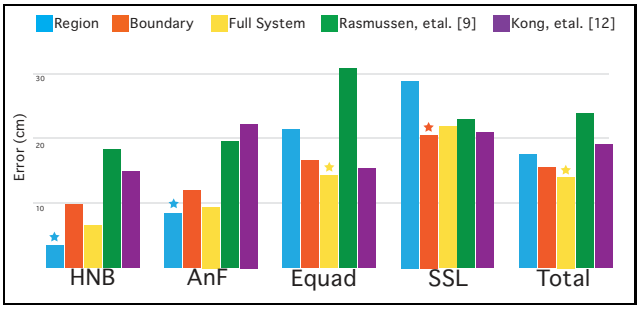

Fig. 8. Road Recognition Results. The bar graphs illustrate the performance of the different algorithms in recognizing road in different environments. Performance is measured by the difference between the ground truth and estimated road center in cm. We report the results by the individual algorithms in our approach, road region and road boundary estimation, and the overall system. In addition, we also include implementations [9] and [12]. Note: shorter bars are better, the shortest in each group denoted by a star.

Fig. 9. System Comparison Examples. The different systems outputs are organized to different rows with the first three rows occupied, in order, by the individual algorithms in our approach, road color region detection and road boundary estimation, and then the overall system. In addition, we also include an implementation of systems by [9] (forth row) as well as [12] (fifth row).

In addition we also compare our individual algorithms with similar alternative methods. Much like our road region estimator, the system by [9] also primarily utilizes color, rather than edge information, to detect the road. However, their system uses a fixed set of triangular templates as shape prior. Ours, on the other hand, segments the road region without a pre-conceived notion of road shape, and directly extract the road middle point for navigation.

The system [9] is accurate if the triangular road shape is sufficiently visible with similar flanking areas. However, in urban environments, the flanking areas are more complex. For example, one side can be similar to the road, such as in column three of figure 9. Our road region estimator, on the other hand, is not constrained with a particular shape and can adapt to this difficulty as can be seen in the example. Its drawback, however, is that it fails when a significant part of the road is covered by a shadow, such as in th fifth column of figure 9. Here, the estimator splits the road into multiple segments and selects the one that appears most similar to the color prior. In this case, a shape prior would be helpful, which in our overall system comes from the road boundary estimator.

We also compare our road boundary estimator with [12]. These two vanishing point-based algorithms differ in that the former utilizes fewer number of edge orientations than the latter, 4 and 36, respectively. In our experiments, we find that even with a small number of orientations, our system maintains the same level of performance, while operating in real-time. This is because a lot of the edge information is repeated between neighboring angles. We also found that the two diagonal Gabor angles ($\alpha = 45°$ and $135°$) contribute the most in the vanishing point voting. Another important factor that we added is temporal information, which we found to be critical because static edge information can be noisy and often times makes it hard for a system to distinguish the true vanishing point from the false ones. By adding temporal continuities our estimator is able to discard sporadic false positives and improve its accuracy.

Both algorithms, however, have a difficulty in specifying the road shape when only one boundary is viewed in the image (sixth column of figure 9) when the robot swerves slightly off course. What usually occurred is that the true road vanishing point does not receive enough support. With the help of the road region estimator the overall system can alleviate the error and still outputs a road center much closer to the ground truth center.

Given these results we proceed to test system for autonomous navigation. In such scenario, any erroneous deviation produced in one frame will be amplified in subsequent frames if the robot does not make the proper correction.

B. Autonomous Navigation

We test the system using our robot, Beobot 2.0 [15], which is 57cm in length and 60cm in width. The camera itself has a regular 60 degree field-of-view and is mounted 117cm above the ground, at the center of the robot. We measure autonomous navigation success by calculating the percentage of distance in which the robot is driving autonomously. Observe figure 10 for an example trajectory.

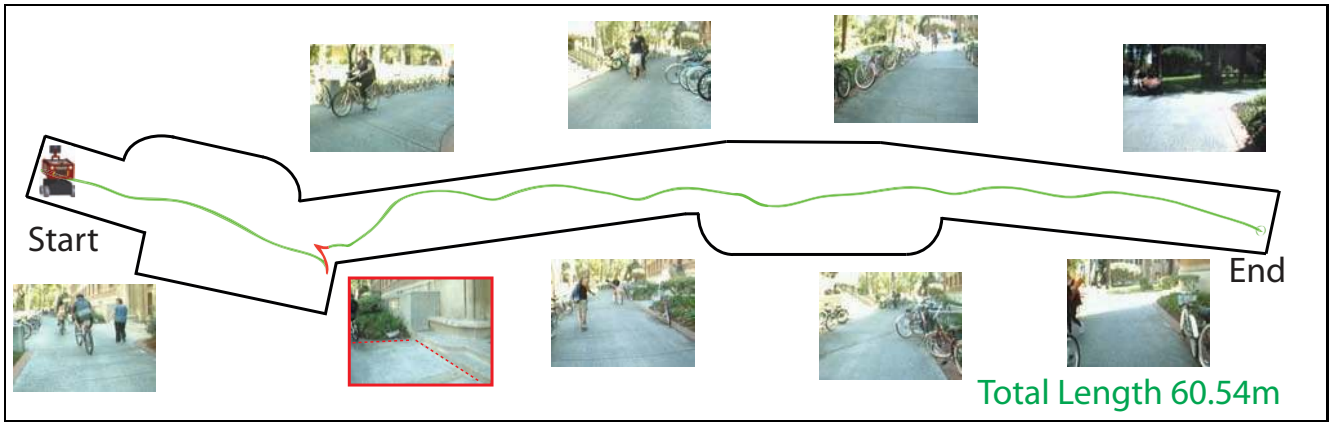

Fig. 10. Example trajectory of autonomous navigation using the presented visual road recognition system. Here Beobot 2.0 has to traverse through a 60.54m route. The autonomous navigation is indicated by the green trail, while the remote control (to rescue the robot from hitting a wall) is colored in red. The reason why the robot moved to the right is because it decided to take the right turn on a road that suddenly decreases its width by a half. In addition, we also the added a few example images to show the different parts of the route, where we see that the robot has to manage its path around people and other vehicles.

TABLE II
AUTONOMOUS NAVIGATION PERFORMANCE RESULTS

	HNB	AnF	Equad	SSL	Total
Path Length	36.67m	59.09m	64.79m	155.24m	316.60m
Auto. Drive	36.67m	57.50m	62.12m	153.45m	309.74m
Manual Drive	0.0m	1.58m	2.38m	1.78m	5.74m
Performance	100%	97.33%	95.87%	98.84%	98.19%

We run the robots five times on each site and average its performance. We selected the longest continuous path within each site so that the robot does not have to make decisions at the intersection. Table II reports the length of the route as well as the percentage of trajectory in which Beobot 2.0 is driving by itself.

From the table, we can see that the system rarely fails, averaging 98.19 percent of autonomous driving. One reason for the failure was that the robot was stuck in a small opening, while going through a door. This occurred because the door frame protrusion was thin, and it quickly went out of the camera field of view. When the robot saw the frame, it tried to avoid it for a short period, until the door was out of the field of view. But then it saw what was beyond the frame, and tried to re-center itself using that portion of the road, not knowing the door frame was in its blind spot. We believe, this case can be improved by better modeling of the environment, using techniques such as SLAM (Simultaneous Localization and Mapping).

Another problem that we encountered was that the road concrete some times extruded out to a bench seating or a bike parking area. Most times, the robot was able to stay on the road and move beyond the extruded area. However, once or twice, the robot would move toward the extruded area such as in figure 7, specifically just prior to the red section of the trajectory. As the robot traversed through the extruded area, its lateral location tend to be biased toward the extrusion to center itself on the road. At some point, the left boundary of the road would be move out of the field of view, while the extrusion itself projected to a triangular shape. The system then fit a road to the corner of the extrusion, making the robot deviate from the road. In this case a wider angle lens or an omni-directional camera would allow the system to maintain a view of the true road, and prevent such situation.

IV. DISCUSSION AND CONCLUSIONS

In this paper we present a novel monocular image-based road recognition system that fuses road region and boundary estimation. By using two complementary ways to characterize the road, we are able to produce a system that is robust in various indoor and outdoor environments. In the road region estimation, we introduce an straight-forward segmentation-based technique to isolate the road without shape constraints. On the other hand, in the road boundary estimation, we propose an efficient vanishing point detection, to allow our system to run in real-time on a robot.

For example, we add a non-uniform sampling mechanism in the vanishing point voting step to reduce computation time. In addition, we demonstrate that even with a reduced number of orientations, our system still maintain its accuracy because of the use of a new Gabor Response Variance Score (GRVS) method. Furthermore, we also compare our algorithm with other approaches, and show that it performs better than others in our dataset.

Finally, our contribution also includes a series of field tests to show that our system is able to autonomous navigate in both indoor and outdoor environments by reliably detecting the navigable road area.

V. ACKNOWLEDGMENTS

The authors gratefully acknowledge the support of NSF, General Motors, ARO and DARPA.

The authors affirm that the views expressed herein are solely their own, and do not represent the views of the United States government or any agency thereof.

REFERENCES

[1] Z. Hu, H. Wang, L. Zhang, and H. Yu, "Laser based sensor boundary recognition of mobile robot," in *International Conference on Networking, Sensing and Control*, Okayama, Japan, March 2009.

[2] K. Wurm, R. Kummerle, C. Stachniss, and W. Burgard, "Improving robot navigation in structured outdoor environmets by identifying vegetation from laser data," in *Intelligent Robots and Systems (IROS)*, oct. 2009, pp. 1217 –1222.

[3] S. Hrabar and G. Sukhatme, "Vision-based navigation through urban canyons," *Journal of Field Robotics*, vol. 26, no. 5, pp. 431 – 452, 2009.

[4] J. Cunha, E. Pedrosa, C. Cruz, A. Neves, and N.Lau, "Using a depth camera for indoor robot localization and navigation," in *RGB-D RSS workshop*, Los Angeles, California, 2011.

[5] Z. Chen and S. Birhfield, "Quantitative vision-based mobile robot navigation," in *International Conference on Robotics and Automation*, May 2006, pp. 2686 – 2692.

[6] C.-K. Chang, C. Siagian, and L. Itti, "Mobile robot vision navigation & localization using gist and saliency," in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct 2010, bb, both first authors contributed equally.

[7] D. Lowe, "Distinctive image features from scale-invariant keypoints," *Intl. Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[8] C. Tomasi and T. Kanade, "Detection and tracking of point features," Carnegie Mellon University, Tech. Rep. CMU-CS-91-132, April 1991.

[9] C. Rasmussen, Y. Lu, and M. Kocamaz, "Appearance contrast for fast, robust trail-following," in *International Conference on Intelligence Robotics and Systems*, October 2009, pp. 3505 – 3512.

[10] T. Kuhnl, F. Kummert, and J. Fritsch, "Monocular road segmentation using slow feature analysis," pp. 800 –806, june 2011.

[11] P. Santana, N. Alves, L. Correia, and J. Barata, "Swarm-based visual saliency for trail detection," in *Intelligent Robots and Systems (IROS)*, oct. 2010, pp. 759 –765.

[12] H. Kong, J.-Y. Audibert, and J. Ponce, "General road detection from a single image," *IEEE Transactions on Image Processing*, vol. 19, no. 8, pp. 2211 – 2220, August 2010.

[13] P. Moghamadam, J. A. Starzyk, and W. S. Wijesoma, "Fast vanishing point detection in unstructured environments," *IEEE Transactions on Image Processing*, vol. PP, no. 99, pp. 1 – 6, July 2011.

[14] M. Nieto and L. Salgado, "Real-time vanishing point estimation in road sequences using adaptive steerable filter banks," *Advanced Concepts for Intelligence Vision Systems Lecture Notes in Computer Science*, pp. 840 – 848, 2007.

[15] C. Siagian, C.-K. Chang, R. Voorhies, and L. Itti, "Beobot 2.0: Cluster architecture for mobile robotics," *Journal of Field Robotics*, vol. 28, no. 2, pp. 278–302, March/April 2011.

[16] P. Felzenszwalb and D. Huttenlocher, "Efficient graph-based image segmentation," *International Journal of Computer Vision*, vol. 59, pp. 167–181, 2004.