

Estimating Drivable Collision-Free Space from Monocular Video

Jian Yao
University of Toronto
yaojian@cs.toronto.edu

Srikumar Ramalingam
MERL
ramalingam@merl.com

Yuichi Taguchi
MERL
taguchi@merl.com

Yohei Miki
Mitsubishi Electric Corporation
Miki.Yohei@cb.MitsubishiElectric.co.jp

Raquel Urtasun
University of Toronto
urtasun@cs.toronto.edu

Abstract

In this paper we propose a novel algorithm for estimating the drivable collision-free space for autonomous navigation of on-road and on-water vehicles. In contrast to previous approaches that use stereo cameras or LIDAR, we show a method to solve this problem using a single camera. Inspired by the success of many vision algorithms that employ dynamic programming for efficient inference, we reduce the free space estimation task to an inference problem on a 1D graph, where each node represents a column in the image and its label denotes a position that separates the free space from the obstacles. Our algorithm exploits several image and geometric features based on edges, color, and homography to define potential functions on the 1D graph, whose parameters are learned through structured SVM. We show promising results on the challenging KITTI dataset as well as video collected from boats.

1. Introduction

The holy grail in mobile robotics and automotive research is to be able to perform navigation without any user interaction. Although this problem has been of interest for several decades, the recent successes of self-driving cars (e.g., Google car, competitors in DARPA urban challenge [7]) have demonstrated that this dream might actually happen in the near future. Robotics researchers have mainly focused in building simultaneous localization and mapping (SLAM) systems using both 2D and 3D sensors [32, 35]. In computer vision, research has focused on a wide variety of problems such as stereo [19, 30], scene understanding [24], image-based localization [18, 6], and pedestrian detection [10].

A common assumption has been that if we are able to develop robust and accurate solutions to these problems, we should be able to solve autonomous navigation relying



Figure 1. (Top row) An image captured from the front of a car while driving on a road. The red curve shows the ground truth free space where the car can move without colliding with other cars and obstacles. The yellow curve shows the free space estimated using our algorithm. The blue marked region is the ground truth road classification. The numbers indicate the 3D distance from the camera to the obstacle estimated by our approach. (Bottom row) An image captured from a boat. The single yellow curve represents the free space where our result and the ground truth coincide visually.

mainly on visual information. Geiger et al.[16] developed the KITTI benchmark to test vision algorithms in the context of self-driving cars. Although tremendous progress has been done since its creation two years ago, existing methods are not accurate enough to replace LIDAR-based navigation systems.

In this paper we are interested in addressing two important questions towards autonomous navigation: What is the most critical information necessary to avoid obstacles? Can we come up with a cost-effective approach to obtain this critical information? It can easily be seen that the most critical information is the **drivable free space** that can be immediately reached by the autonomous vehicle **without collision**. In order to estimate the immediate drivable free space, existing self-driving cars typically utilize a LIDAR scanning in every direction. Badino et al. [3] showed that

stereo cameras can be used to replace the LIDAR in solving this task. But, is a stereo pair really necessary?

Our main contribution is to show that this task can be computed using a single camera mounted on a moving platform (i.e., car, boat). When depth information (from stereo or LIDAR) is available, the drivable free-space can be easily computed by identifying the obstacles above the ground plane. This process is, however, non trivial when utilizing a single camera. Monocular approaches typically rely on image edges or geometry to segment the ground plane. Unfortunately, computing reliably obstacle boundaries is a hard problem. In roads, we often observe strong gradients from cross-walks and lane markings, which are not obstacles. In the case of water, there is often strong gradients from the reflection of nearby boats, buildings, and sky. Ground plane estimation might not be reliable due to non-flat roads [23]. Furthermore, moving vehicles pose additional challenges in the monocular setting.

Our approach addresses all these challenges by framing the problem as the one of inference in a 1D Markov random field (MRF), where each node represents a column in the image and its label denotes a position in the ground plane that separates the free space (that can be reached without collision) from the obstacles. Note that if we have an estimate of the ground plane and a calibrated camera, this parametrization can be directly used to estimate the distance to the obstacles (see Fig. 1). Towards this goal, we exploit several image and geometric features based on edges, color, and homography to encode the potential functions that define our energy. As our graph forms a chain, efficient and exact inference can be computed using dynamic programming. Furthermore, we learn the importance of all cues using structured SVMs. We demonstrate the effectiveness of our approach on two scenarios, the challenging KITTI autonomous driving benchmark [16] as well as videos captured while maneuvering a boat near a dock. In the remainder of the paper, we first discuss related work and clarify our contributions. We then present our monocular free space estimation approach, followed by experimental evaluation and conclusions.

2. Related Work

In the past decade, many approaches have been proposed for automatic navigation with obstacle avoidance. The works of [9, 2, 17, 33, 25, 21] have the same goal as us, but the sensors they employed are different. In particular, they utilized laser range finders, accurate pose estimation systems and GPS, while we employ a single low-cost monocular camera.

The concept of occupancy grids, first introduced in [12], is closely related to the free space estimation problem. An occupancy grid refers to a 2D grid where every cell models the occupancy evidence of the environment. They are

typically estimated using a laser range finder or an array of ultrasonic sensors for autonomous navigation of mobile robots [35]. Free space has been also estimated using stereo cameras [14, 3]. In [14], the ground plane as well as obstacles above the ground were estimated using a stereo system. Badino et al. [3] used dynamic programming to solve the free space estimation problem. This was later extended to compute both the free space as well as the height of obstacles in [4]. This is typically referred to as the **stixel world**, which is a simplified model of the world using a ground plane and a set of vertical sticks on the ground representing obstacles. This can be compactly represented on the image using two curves, where the first curve runs on the ground plane enclosing the free space that can be immediately reached without collision, and the second curve encodes the height of all the vertical obstacles at the boundary of the free space. In order to compute the stixel world, semi-global stereo matching algorithm (SGM) was used [19]. Recently, it was shown that stixels can be computed with a stereo camera, but without explicit depth estimation [5]. In [1, 31, 26] free space is estimated using binary classification. They use a different notion of free space which, unlike ours, includes the space behind obstacles. As a consequence efficient and/or exact inference is not possible as the resulting graphical model has loops.

Our work is related to estimating the geometric layout of outdoor scenes. Hoiem et al. [20] showed that it is possible to reliably classify the pixels in a given image into ground, buildings, and sky. Similar to this geometric layout estimation problem, we also explore several appearance and geometric features. Felzenszwalb and Veksler [13] modeled the scene using two horizontal curves that divide the image into three regions: top, middle, and bottom. Exact inference is possible, however the complexity is prohibitive for real-time applications when employing large images.

The general idea of using dynamic programming for column-wise matching was used in [8] for estimating the 3D models of buildings. Gallup et al. [15] generalized this work to several layers of height maps for modeling urban scenes. In contrast, we are interested in scenes with challenging traffic conditions rather than buildings. Furthermore, we exploit temporal information in order to produce smoothly varying results on video sequences. Monocular videos have been used to develop SLAM algorithms [28, 11, 22, 29, 27]. These approaches provide sparse or dense point clouds and do not explicitly estimate the free space.

Our main contributions can be summarized as follows:

- We propose a fast and exact inference algorithm for estimating planar free space using monocular video.
- We show promising experimental results on the KITTI benchmark as well as videos captured from a boat.

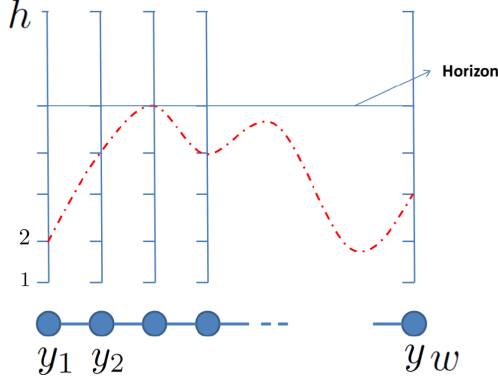


Figure 2. The 1D graph used in estimating the free space in both scenarios related to automobiles and boats. The graph consists of a set of nodes denoted by y_i corresponding to the w columns of the image. Each node represents a discrete variable whose value comes from a set of labels given by the h rows in the image. On labeling these nodes, we obtain a curve that defines the free space in front of the automobile or boat. The labels are ordered in such a manner that the bottom pixel has a label 1 and the top pixel has a label h .

3. Monocular Free Space Estimation

In this paper we are interested in the problem of computing the drivable free space given monocular imagery in the context of both marine and urban navigation. Towards this goal, we express the problem as the one of inference in a Markov random field (MRF), which estimates for each image column, the vertical coordinate of the pixel that separates the free space from obstacles. This defines a 1D curve in the image as shown in Fig. 2. In what follows, we will give more details on the energy functions, potentials, inference, and parameter learning.

3.1. Energy function

Let I_t denote the image at time t in a video. Let the dimensions of the image be $w \times h$, where w and h are the width and height respectively. We model the problem in such a manner that we have w discrete variables $y_i, i \in \{1, \dots, w\}$ and each variable can take a value from h discrete labels, $y_i \in \{1, \dots, h\}$. Let us consider the 1D chain graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, where the vertices are given by $\mathcal{V} = \{1, \dots, w\}$ and the edges are given by $(i, i+1) \in \mathcal{E}, i \in \{1, \dots, w-1\}$ as illustrated in Fig. 2. Note that we can further restrict the states of y_i as it can never be above the horizon (i.e., the horizon is always above the ground plane). Using the training images, we compute a bound for the horizon line and use it to restrict the labels in the inference procedure. Note that more sophisticated horizon estimation algorithms can be used.

In order to compute the curve for image I_t , we make use of the features from images I_t and I_{t-1} . Our energy

function E can be summarized as

$$E(\mathbf{y}, I_t, I_{t-1}) = -\mathbf{w}^T \phi(\mathbf{y}, I_t, I_{t-1}), \quad (1)$$

where $\mathbf{y} = (y_1, \dots, y_w)$, and the potentials decompose into unary and pairwise terms:

$$E(\mathbf{y}, I_t, I_{t-1}) = - \underbrace{\sum_{u \in \mathcal{U}} \sum_i \mathbf{w}_u^T \phi_u(y_i)}_{\text{unary}} - \underbrace{\sum_{(i,j) \in \mathcal{E}} \mathbf{w}_p^T \phi_p(y_i, y_j)}_{\text{pairwise}}. \quad (2)$$

Our unary potentials exploit appearance, edges as well as temporal information (in the form of homography), while our pairwise potentials encode spatial smoothness in the 1D curve. The parameters for our energy are $\mathbf{w} = \{\mathbf{w}_u, \mathbf{w}_p\}$, with \mathbf{w}_u and \mathbf{w}_p the unary and pairwise parameters respectively. As described below, we learn both sets of parameters using structured prediction. We now describe the potentials in more detail.

Appearance: We use two GMMs, each with 5 components, to model the probability for each pixel to be foreground (i.e., road/sea) or background, and use Expectation Maximization (EM) to learn the parameters. Since the GMMs can be noisy, we use a location prior to enforce pixels that are always road to be so. We simply employ the training data to determine this region. Our goal is then to estimate the free space in such a manner that the curve lies on the boundary between road and non-road (or water and obstacles). Towards this, we derive a potential that looks at the entropy of the distribution in patches around the labels:

$$\phi_{\text{appearance}}(y_i = k) = \mathcal{H}(i, k) \sum_{j=k}^h \mathcal{H}(i, j). \quad (3)$$

Here the entropy $\mathcal{H}(i, j)$ is computed in terms of the distribution of road/non-road pixels in a patch centered at pixel location (i, j) . The entropy $\mathcal{H}(i, j)$ should be high near the boundary between road/non-road pixels. Since we are looking for a curve that passes through the boundary between the closest set of obstacles and the road, we use a cumulative sum that prefers pixels that are closer to the car and the ones with non-zero $\mathcal{H}(i, k)$ values.

Edge Information: This potential encourages the curve to be aligned with contours. As there are many contours in the image, we would like to encode that edges that are located at the bottom of the image or closer to the camera in 3D space are preferred. To take this into account, we define a potential which accumulates edge evidence as

$$\phi_{\text{edge}}(y_i = k) = e(i, k) \sum_{j=k}^h e(i, j), \quad (4)$$

where $e(i, j) = 1$ if there is an edge at the pixel (i, j) , and 0 otherwise. The edges are obtained by simply using the Canny edge detector with default parameters. Note that more sophisticated edge detectors could be employed.

Temporal Smoothness: One possibility to encode smoothness is to estimate the curves in two consecutive images jointly by considering pairwise connections between nodes in the two images. The standard approach would be to constrain the labeling of a pixel $p(i, j)$ in image I_t with neighboring pixels of $p(i', j')$ in image I_{t-1} . This would lead to a graph that is not tree-structured. As a result, inference becomes NP-hard. Instead, we use homography to impose smoothness across images and still maintain the 1D chain graph during inference. The basic idea is simple, instead of using smoothness across nearby pixels, we compute a homography matrix based on the ground plane. This gives us a one-to-one mapping from a pixel on the ground in one image to its corresponding pixel on the ground in the previous image. This also provides a mapping between the free space curve in one image to the other, as shown in Fig.4. Let $H(t, t-1)$ be the homography matrix that maps a pixel location (i, j) in image I_t to a pixel location (i', j') in image I_{t-1} , given by

$$\begin{pmatrix} i' \\ j' \\ 1 \end{pmatrix} = H(t, t-1) \begin{pmatrix} i \\ j \\ 1 \end{pmatrix}. \quad (5)$$

We write the potential as

$$\phi_{homography}(y_i = j) = \phi_u(y_{i'} = j'), \quad (6)$$

where $u \in \mathcal{U} \setminus homography$ and $\phi_u(y_{i'} = j')$ is the unary potential in image I_{t-1} , and $y_{i'}$ is computed as in Eq. (5).

Spatial Smoothness: We employ a truncated quadratic penalty to encourage the curve to be smooth. Note that the curve is non-smooth only when there are obstacles, which happens only at a few columns. Thus,

$$\phi_p(y_i, y_j) = \begin{cases} \exp(-\alpha(y_i - y_j)^2) & \text{if } |y_i - y_j| \leq T \\ \lambda_d & \text{otherwise} \end{cases} \quad (7)$$

where α , λ_d , and T are constants.

3.2. Learning and Inference

Inference consists in computing the MAP estimate, or minimum energy configuration. This can be done as follows

$$\max_{\mathbf{y}} \mathbf{w}^T \phi(\mathbf{y}, I_t, I_{t-1}). \quad (8)$$

Our graphical model forms a chain, and thus exact inference can be done using dynamic programming, with a complexity of $\mathcal{O}(wn^2)$, with w the width of the image and n the

number of labels for each variable after imposing the constraint from the horizon.

We employ structured SVMs [34, 36] to learn the parameters of the model by:

$$\begin{aligned} \min_{\mathbf{w}, \{\xi_i\}} \quad & \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_i \xi_i & (9) \\ \text{s.t.} \quad & \xi_i \geq \mathbf{w}^T (\phi_i(\mathbf{y}) - \phi_i(\mathbf{y}^{(i)})) + \Delta(\mathbf{y}, \mathbf{y}^{(i)}), \quad \forall \mathbf{y}. \\ & \xi_i \geq 0, \quad \forall i = 1, \dots, N. \end{aligned}$$

Here, $\mathbf{y}^{(i)}$ is the ground-truth curve for the i -th instance, $\Delta(\mathbf{y}, \mathbf{y}^{(i)})$ the loss function, and N the total number of training examples. The loss function is a truncated version of the relative gap given by

$$\Delta(y_i, y) = \begin{cases} |y - y_i| & \text{if } |y - y_i| \leq T \\ T & \text{if } |y - y_i| > T \end{cases} \quad (10)$$

where T is a constant. We use the cutting plane algorithm of [36], where at each iteration we have to solve the following loss augmented inference to find the most violated constraint:

$$\forall i, \quad \max_{\mathbf{y}} \mathbf{w}^T (\phi_i(\mathbf{y}) - \phi_i(\mathbf{y}^{(i)})) + \Delta(\mathbf{y}, \mathbf{y}^{(i)}). \quad (11)$$

As the loss decomposes into unary potentials, the loss-augmented inference can be solved exactly via dynamic programming.

4. Experiments

In this section we show our experimental evaluation. We begin our discussion by describing the datasets we employ. We then show our quantitative and qualitative evaluation in both marine and urban free space domains.

4.1. Datasets

We used two different datasets for evaluation based on urban and marine scenes.

KITTI: We use the training set from the road challenge in KITTI [16]. The dataset includes 289 monocular video sequences for road detection. The images include road scenes under moderate traffic conditions as shown in Fig. 7. We labeled the drivable collision-free space for all images.

Boat: We mounted a GoPro camera on a boat and collected monocular video sequences while maneuvering the boat near a dock. We manually labeled 200 consecutive images and use the first 100 for training and the remaining 100 images for testing. We extracted images at 5 fps due to the slow motion of the boat.

We compute the homography using SIFT correspondences in a RANSAC framework. In the case of water, we look for correspondences only near the previous free space

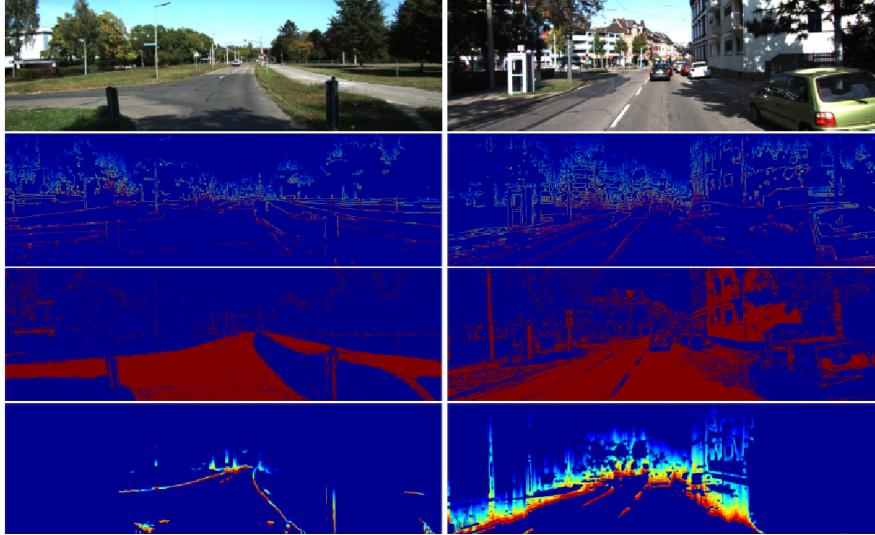


Figure 3. (First row) Two images from KITTI road detection dataset. (Second row) Edge features. (Third row) Segmentation computed using GMM. (Fourth row) Appearance features. All the features are color-coded, where red denotes high value and blue denotes low value.

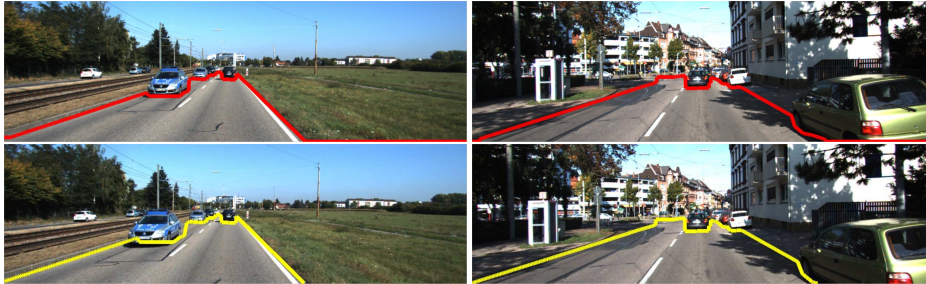


Figure 4. On the top, we show the ground-truth free-space curve in image I_{t-1} . Using the estimated homography, we show the free-space curve on the next image I_t under only the homography transform.

curve as we typically have too many spurious correspondences on water. Note that the edges from reflections of objects on water will not match using the homography matrix, and thus we can filter them from our curve. In the case of road scenes, we use the SIFT matches below the previous free space curve to compute the homography matrix, as reflections are not a problem.

4.2. Quantitative Evaluation

We evaluate our approach using two measures: relative gap (\mathcal{G}) and F1 score ($F1$).

Relative Gap: We first compute the difference between the estimated curve and the ground truth curve. The relative gap is then obtained as the ratio of the average difference with respect to the height of the image h as

$$\mathcal{G}(\mathbf{y}, \mathbf{y}^*) = \frac{\|\mathbf{y} - \mathbf{y}^*\|_1}{n \cdot h}, \quad (12)$$

where \mathbf{y} is the estimated curve and \mathbf{y}^* is the ground truth, h is the height of the image, and n is the normalization con-

stant which takes the number of columns.

F1 Score: As in many vision problems, the ground truth labeling may not be perfect. For autonomous navigation applications, it is not a serious problem if the estimated free space is smaller than the actual one. On the other hand, it is more critical to not have any obstacles inside the free space curve. In regards to this, we propose the $F1$ score to measure the accuracy of classification of pixels under the curve given by

$$F1 = \frac{2 \times P \times R}{P + R}, \quad (13)$$

where P and R refer to the precision and recall, which will be computed only on the pixels that are under the curve.

In order to understand the importance of each feature, we trained and tested our approach using different combinations of features. Tables 1 and 2 show the \mathcal{G} and $F1$ measures using different features for KITTI road dataset and the boat dataset, respectively. As shown in the tables, the appearance term is robust and effectively used together with the smoothing term. The edge potential is also useful

E	A	H	S	$\mathcal{G}(\%)$	$F1(\%)$
✓			✓	13.29	57.18
	✓		✓	7.07	77.08
✓	✓		✓	5.97	80.93
✓	✓	✓	✓	5.45	82.51

Table 1. Drivable free space estimation result on KITTI benchmark dataset [16]. We denote edges, appearance, homography, and smoothness using E, A, H, and S, respectively. Out of the 289 video sequences, we left out four sequences due to problems in ground truth. Here we randomly selected 100 images for training and 100 for testing and 85 for validation. The numbers above are on test images.

E	A	H	S	$\mathcal{G}(\%)$	$F1(\%)$
✓			✓	2.47	96.23
	✓		✓	2.38	96.37
✓	✓		✓	2.36	96.39
✓	✓	✓	✓	2.43	96.3

Table 2. Drivable free space estimation result for boat dataset. We denote edges, appearance, homography, and smoothness using E, A, H, and S, respectively. We randomly select 100 frames for training and 100 for testing from a video.

in combination with the smoothness and appearance. Besides, when the planar assumption of the free space is well satisfied (e.g. road scenario), the homography potential increases performance as well.

We tested different values of regularization parameter C in structured SVM. As shown in Table 3, the accuracy of our algorithm is not very sensitive to the regularization parameter.

We also compared our results with monocular classification algorithms as shown in Table 4 and Table 5. We use [20] for the road and [1, 31] for the water. We computed the free space from these classification algorithms using two approaches: (1) getting the curve directly from the classification result, i.e., we use Eq. 3 to construct the unary feature and then maximize it; (2) estimating the curve when using also the pairwise potentials (smoothness term used in our algorithm). Using smoothness improves performance as it reduces the effect of miss-classified labels in the segmentation results. Our model significantly outperforms the baselines.

C	0.005	0.01	0.1	1	2	5
\mathcal{G}	2.42	2.43	2.42	2.40	2.39	2.43
F1	96.31	96.30	96.32	96.35	96.36	96.30

Table 3. For different values of C , the coefficient of regularization used in the structured SVM, we evaluate our algorithm on the boat dataset with all features.

	$\mathcal{G}(\%)$	$F1(\%)$
[1, 31]	16.33	67.98
[1, 31]+Smooth	15.06	70.98
our model	2.36	96.39

Table 4. Comparison with water classification methods.

	$\mathcal{G}(\%)$	$F1(\%)$
[20]	18.46	62.22
[20]+Smooth	16.42	64.31
our model	5.45	82.51

Table 5. Comparison with monocular road detection methods.

4.3. Distance To Obstacle

Using the calibration matrix K and the height of the camera with respect to ground plane, we can compute the 3D distance from the obstacles (i.e., pixels in the 1D curve) to the camera center. Fig. 5 depicts the distances inferred by our approach on the road scenario.

4.4. Qualitative Evaluation

Qualitatively, we observed the free space using different combinations of features. As shown in Fig. 6, we see an improvement in the result as we add more features. Edges and color are not sufficient to obtain the free space accurately. We also looked at the best and the worst results for the free space estimation for both KITTI (Fig. 7) and the boat dataset (Fig. 8). Besides, we also show some results from the boat video with no ground truth labeled in Fig. 9. A full length video demonstrating our free space estimation results is provided in the supplementary material, for a total of more than 4000 frames.

4.5. Computation Time

Our approach takes 0.1s per image on average for the computation of all features in Matlab. Our un-optimized implementation of the inference algorithm in C++ takes on average 0.1s per image. Note that one can use the optimized DP implementation of [5] which runs at 135Hz.

5. Conclusions and Discussion

We have presented a free space estimation algorithm using monocular sequences. The approach was tested on two different datasets involving road and water scenes. In contrast, prior work used stereo cameras or LIDARs for this purpose and focus mainly on road scenarios. We designed our algorithm using simple and light-weight techniques for real-time feature computation and inference. Note that our approach currently uses features from one previous frame to estimate the free space in the current image. We could extend this easily to multiple images, still maintaining the 1D



Figure 5. The red curve is the result from our model. When the parameters of the camera are known, we can estimate the distance from the obstacle to the camera (yellow text).



Figure 7. Example results from KITTI road dataset. The top four rows show some of the best performing scenes, demonstrating accurate results given by our approach. The last row shows worst performing scenes; failure modes are typically due to road markings and confusion between pavement and road.

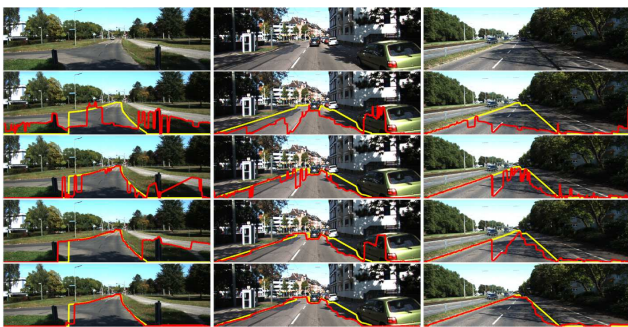


Figure 6. The evolution of the free space curve as we use more and more features. The yellow curve is the ground truth and the red curve is our result. (First row) Original images. (Second row) Edge. (Third row) Appearance. (Fourth row) Edge + Appearance. (Fifth row) Edge + Appearance + Homography. All the results use the smoothness term.



Figure 8. The first row shows top scoring results from the boat sequence, while the second row shows low scoring results. The red curve is our result and the yellow one is the ground truth.



Figure 9. Results selected from the full length of boat video sequences. There are no ground truth labeled for these frames.

chain graph while doing the inference. We notice that the results of our algorithm degrade when there are road marks and shadows in the image. We plan to resolve these issues using depth features that can be extracted using motion stereo algorithms as well as more sophisticated appearance features. Note that however, most segmentation algorithms are not suited for real-time applications, and thus cannot be employed for our purpose. We also plan to use LIDAR to generate ground truth, which can be used for better training and evaluation of our results.

Acknowledgments: We thank Jay Thornton and Shin Miura for useful discussions. We thank Sanja Fidler for the 3D distance computation program. This work was supported by MERL.

References

- [1] S. Achar, B. Sankaran, S. Nuske, S. Scherer, and S. Singh. Self-supervised segmentation of river scenes. In *ICRA*, 2011.
- [2] A. Angelova, L. Matthies, D. Helmick, and P. Perona. Fast terrain classification using variable-length representation for autonomous navigation. In *CVPR*, 2007.
- [3] H. Badino, U. Franke, and R. Mester. Free space computation using stochastic occupancy grids and dynamic programming. In *ICCV Workshop on Dynamical Vision*, 2007.
- [4] H. Badino, U. Franke, and D. Pfeiffer. The stixel world - a compact medium level representation of the 3d-world. In *DAGM*, 2009.
- [5] R. Benenson, R. Timofte, and L. Gool. Stixels estimation without depth map computation. In *ICCV*, 2011.
- [6] M. Brubaker, A. Geiger, and R. Urtasun. Lost! leveraging the crowd for probabilistic visual self-localization. In *CVPR*, 2013.
- [7] M. Buehler, K. Iagnemma, and S. Singh. *The DARPA Urban Challenge: Autonomous Vehicles in City Traffic*. Springer, 2009.
- [8] N. Cornelis, B. Leibe, K. Cornelis, and L. Gool. 3d city modeling using cognitive loops. In *CVPR*, 2006.
- [9] H. Dahlkamp, A. Kaehler, D. Stavens, S. Thrun, and G. Bradski. Self-supervised monocular road detection in desert terrain. In *RSS*, 2006.
- [10] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, 2005.
- [11] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse. MonoSLAM: Real-time single camera SLAM. *PAMI*, 2007.
- [12] A. Elfes. Sonar-based real-world mapping and navigation. *Journal of Robotics and Automation*, 1987.
- [13] Felzenszwalb and Veksler. Tiered scene labeling with dynamic programming. In *CVPR*, 2010.
- [14] U. Franke and I. Kutzbach. Fast stereo based object detection for stop and go traffic. In *IV*, 1996.
- [15] D. Gallup, M. Pollefeys, and J. Frahm. 3D reconstruction using an n-layer heightmap. In *DAGM*, 2010.
- [16] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *CVPR*, 2012.
- [17] R. Hadsell, P. Sermanet, J. Ben, A. Erkan, M. Scoffier, K. Kavukcuoglu, U. Muller, and Y. LeCun. Learning long-range vision for autonomous off-road driving. *JFR*, 2009.
- [18] J. Hays and A. Efros. Im2gps: estimating geographic information from a single image. In *CVPR*, 2008.
- [19] H. Hirschmuller. Stereo processing by semiglobal matching and mutual information. *PAMI*, 2008.
- [20] D. Hoiem, A. A. Efros, and M. Hebert. Automatic photo pop-up. *ACM Trans. Graph.*, 2005.
- [21] D. Kim, J. Sun, S. M. Oh, J. M. Rehg, and A. F. Bobick. Traversability classification using unsupervised on-line visual learning. In *ICRA*, 2006.
- [22] G. Klein and D. Murray. Parallel tracking and mapping for small AR workspaces. In *ISMAR*, 2007.
- [23] R. Labayrade, D. Aubert, and J. Tarel. Real time obstacle detection on non flat road geometry through ‘v-disparity’ representation. In *IV*, 2002.
- [24] L. Ladicky, P. Sturges, C. Russell, S. Sengupta, Y. Bastanlar, W. Clocksin, and P. Torr. Joint optimisation for object class segmentation and dense stereo reconstruction. In *BMVC*, 2010.
- [25] J. Michels, A. Saxena, and A. Y.NG. High speed obstacle avoidance using monocular vision and reinforcement learning. In *ICML*, 2005.
- [26] A. M. Neto, A. C. Victorino, I. Fantoni, and J. V. Ferreira. Real-time estimation of drivable image area based on monocular vision. In *IV*, 2013.
- [27] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison. DTAM: Dense tracking and mapping in real-time. In *ICCV*, 2011.
- [28] D. Nister. An efficient solution to the five-point relative pose. *PAMI*, 2004.
- [29] D. Scaramuzza and R. Siegwart. Appearance-guided monocular omnidirectional visual odometry for outdoor ground vehicles. *IEEE Transactions on Robotics*, 2008.
- [30] D. Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *IJCV*, 2002.
- [31] S. Scherer, J. Rehder, S. Achar, H. Cover, A. Chambers, S. Nuske, and S. Singh. River mapping from a flying robot: state estimation, river detection, and obstacle mapping. *Auton Robot*, 2012.
- [32] R. Siegwart, I. Nourbakhsh, and D. Scaramuzza. *Introduction to Autonomous Mobile Robots, Second Edition*. The MIT Press, 2011.
- [33] D. Silver, J. A. Bagnell, and A. Stentz. Learning from demonstration for autonomous navigation in complex unstructured terrain. *IJRR*, 2010.
- [34] B. Taskar, C. Guestrin, and D. Koller. Max-margin markov networks. In *NIPS*, 2003.
- [35] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics: Intelligent Robotics and Autonomous Agents*. The MIT Press, 2005.
- [36] I. Tsochantaris, T. Joachims, T. Hofmann, and Y. Altun. Large margin methods for structured and interdependent output variables. *JMLR*, 2005.