Accurate 3D Ground Plane Estimation from a Single Image

Anoop Cherian Vassilios Morellas Nikolaos Papanikolopoulos
cherian@cs.umn.edu morellas@cs.umn.edu npapas@cs.umn.edu
Department of Computer Science and Engineering,
University of Minnesota, Minneapolis, MN 55455

Abstract— **Accurate localization of landmarks in the vicinity of a robot is a first step towards solving the SLAM problem. In this work, we propose algorithms to accurately estimate the 3D location of the landmarks from the robot only from a single image taken from its on board camera. Our approach differs from previous efforts in this domain in that it first reconstructs accurately the 3D environment from a single image, then it defines a coordinate system over the environment, and later it performs the desired localization with respect to this coordinate system using the environment's features. The ground plane from the given image is accurately estimated and this precedes segmentation of the image into ground and vertical regions. A Markov Random Field (MRF) based 3D reconstruction is performed to build an approximate depth map of the given image. This map is robust against texture variations due to shadows, terrain differences, etc. A texture segmentation algorithm is also applied to determine the ground plane accurately. Once the ground plane is estimated, we use the respective camera's intrinsic and extrinsic calibration information to calculate accurate 3D information about the features in the scene.**

I. Introduction

Simultaneous Localization And Mapping (SLAM) is the classic problem where a robot in an unknown terrain has to build a map with respect to the Euclidean cordinates of features in its environment and has to localize itself with respect to this map. Inferring accurately the 3D positional information of features with respect to the robot is thus of primary importance in solving SLAM. Traditionally, a laser scanner or a stereo vision system has been used to estimate the location of these features. The high cost of 3D scanners and the need for proper calibration of stereo cameras render both sensor modalities unfavorable for this task and thus this motivates research for better alternatives.

In recent years there have been a number of techniques reported in the literature focusing on the use of monocular cues for depth estimation. Depth from shading [10], depth from blur/focus/defocus [11], depth from magnification [13], etc. are representatives of these efforts. A problem with these approaches is that they impose explicit assumptions on the environment such as Lambertian reflectance, single source of light direction, etc. In the work of [3], texture gradient based methods for inferring the depth are suggested, but they are limited to a noise free environment. Three dimensional reconstruction from a single image is discussed in [1] whereby availability of perspective planes is required. In [7], a dynamic Bayesian model for learning depth is introduced but it is limited to indoor images.

In the work reported in [4], Homiem et. al use a texture segmentation algorithm based on [5] to divide the environment into ground, vertical, and sky. But the results not only do not seem to be generalizable but this technique is not accurate enough for localization. In [8], [2] and [14], Saxena et. al proposed a different algorithm for building a depth map from a single image. This algorithm is robust to texture differences on the ground plane, it is relatively immune to shadows, and it works well in finding discontinuities in the image. This last capability dictates its use for obstacle detection among others. One problem associated with their method is that the output is continuous and finding boundaries or edges of planes is infeasible, especially when considering irregularities of the reconstructed surface. The authors in [2] suggest machine learning to directly determine the plane parameters, but the algorithm was found to be less efficient when operating in real-time. Also, one may be more interested in building a high accuracy ground map for robot localization rather than a visually pleasing 3D reconstruction.

Our approach introduces a novel way of combining the depth map results from Saxena et. al in [8], with the texture segmentation algorithm reported in [5] to provide an accurate ground plane estimation. The main idea of our methodology is to build an accurate ground/vertical image representation from the given image. Our primary assumptions are that no objects are hung in the air and the depth of an object can be estimated by the distance between the point at which an object touches the ground and the position of the camera. We also assume in our derivations that the optical axis of the camera embedded on the robot is parallel to the ground plane. Our method starts by first reconstructing the 3D depth map of the image using an MRF model as described in [8]. We consider the smoothing of the MRF to be very important as indiscriminate smoothing can lead to loss of valuable information regarding obstacles in the robot path. As a result of this we introduce a novel smoothing parameter estimation method based on Principal Component Analysis (PCA).

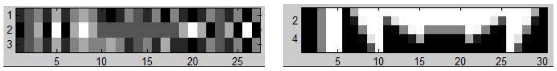
In order to find the boundaries of the ground plane, a texture segmentation of the image is performed next following the method outlined in [5]. This algorithm returns segments (called super-pixels from here on) in the image whose encompassing pixels exhibit similar textures. A problem with this approach is that small texture differences like shadows or changes in lighting conditions result in separate super-pixels. In order to build an accurate ground plane, we introduce a novel approach by combining the depth maps with the superpixels. Once the ground plane is found, we use the calibration of the camera along with information regarding its Euclidean position to build a 3D coordinate system for the image. We intend to address in the future the intricacies associated with the simultaneous movements of the robot and the change of feature coordinates due to the robot's moves.

The rest of the document is organized as follows: We begin with algorithms for estimating the ground plane outlined in Sections II.A, II.B and II.C. In Section II.D, we define a reference frame for estimating the depth at a pixel in the image. Finally, we discuss our experiments, a few sample reconstructions, and empirical results in Section III.

II. Ground plane estimation

A. 3D depth map construction

Humans seldom have any difficulty in inferring the 3D structure of the scene from a single image. This is not only attributed to prior knowledge about the environment but also to using monocular cues such as texture variations, occlusion, known object sizes, haze, de-focus, etc. For example, the texture of many objects will look different at different distances from the viewer. Texture gradients capture the distribution of the direction of the edges and it is a valuable source of depth cues. Haze, which is caused by atmospheric light scattering can also provide texture information to define depth. For example, the haze of the sky is very different from the haze of something closer to the camera. This research stems from the approach described in [8]. It uses texture energies, texture gradient, and haze as the sources of depth information. Since we deal with RGB images in which the texture is distributed over the three color channels, it is better to consolidate the texture information to a single intensity channel by converting the image into the YCbCr format. This pixel intensity information is convolved with 9 Laws mask filters [6], 2 local averaging filters, and 6 Nevatia-Babu texture gradient filters [9]. The Laws and Nevatia-Babu filters are shown in Figure 1. Typical outputs of these filters for a sample image are shown in Figure 2. These 17 filter outputs combined with their squared energies form a 34-dimensional feature vector. In order to include information regarding depth at neighboring pixels, we include the feature vectors from the four neighbors as well to form a 34x5=170-dimensional feature vector. We call it the *absolute feature vector* since its components directly map to the depth at the pixel. The absolute feature vector is complemented with another feature vector, called the *relative feature vector* to represent relative depths between two given pixels. It is formed using the 17-dimensional feature vector introduced earlier by creating a 10-bin histogram for every dimension. We assume that the relative depth between two pixels is proportional to the difference between the respective relative feature vectors.



(a) (b)

Fig. 1. (a) Laws masks filters (9 3x3 filters), (b) Nevatia-Babu texture gradient filters.

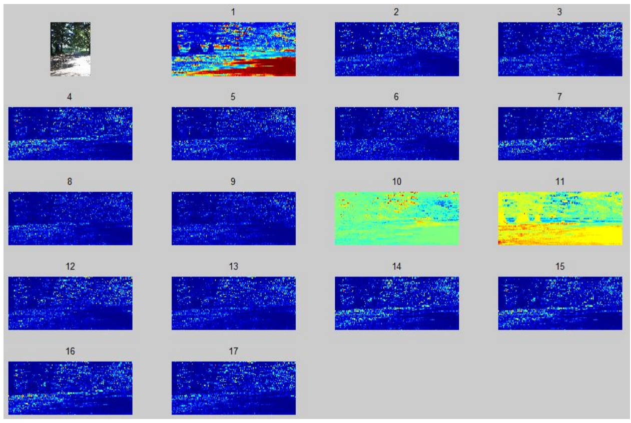


Fig. 2. First subplot shows the original image. Images 1-9 are outputs of the Laws mask filters. These filters do local averaging, edge detection, and spot detection. Images 10-11 are the results of local averaging filters applied to the Cb and Cr channels respectively. Images 12-17 are the outputs of the Nevatia-Babu filters. They provide information regarding the direction of texture gradients.

Now that depth information from a single image has been represented by a composite feature vector (using the absolute and relative feature vectors), a supervised learning algorithm applied on a Gaussian Markov Random Field (MRF) [12] is deployed to estimate the posterior distribution of the depth for every pixel in the image. We do not use the feature vectors at multiple scales as discussed in the original paper. This is a compromise we have made as the extra computational burden introduced is not justified by the accuracy achieved. We model the posterior distribution of depth d, given the feature vectors X parameterized by σ and θ as:

$$P(d|X;\sigma,\theta) = \frac{1}{Z}exp(-E_{\sigma,\theta}(d,X)) \tag{1}$$

where

$$E_{\sigma,\theta}(d,X) = \sum_{i=1}^{n}\frac{(d_i - X_i\theta_r)^2}{\sigma_{1r}^2} + \sum_{i=1}^{n}\sum_{j\in N_s(i)}\frac{(d_i-d_j)^2}{\sigma_{2r}^2}. \tag{2}$$

Here, Z is a normalization constant, $E_{\sigma,\theta}(d,X)$ defines a Gibbs energy function, and X_i is the feature vector at pixel i as explained earlier. The first term in Equation (2) models the absolute depth at the pixel i in terms of feature vectors through the parameter θ_r. As it is apparent from the equation, we use a linear relationship between the feature vectors and the depth at a pixel. A different θ_r parameter is used for each row of the image because the laser depth maps we use for training the system were captured using a horizontally mounted camera. To estimate the θ_r parameters we use a linear least squares solution over the training data. The second term in (2) models the relative depth at pixel i in terms of the depths at its four neighboring pixels $N_s(i)$.

The parameter σ_r defines the smoothness of the depth map. Similarly to the values of the θ_r parameters, we estimate different σ_r for each row of the image. Estimation of these parameters is quite important for the depth map creation since it defines the smoothness of the depth variations. We introduce a PCA based approach to find the σ^2_{1r} values. We find the parameters U_r of a hyperplane such that $\sigma^2_{1r} = \|U_r X_i\|$. Eigen analysis is used to estimate U_r. We find the covariance matrix between the difference in $d_i - X_i\theta_r$ across all the pixels in the row and across all the training images. The eigenvector corresponding to the largest eigenvalue is selected and this forms the vector U_r for a given row.

Estimation of the σ^2_{2r} parameters is achieved by following a similar approach. Since this parameter defines relative smoothness across pixel depths, the relative feature vectors introduced earlier are considered here. We estimate the plane parameter V_r from the eigen analysis of the covariance matrix of $(d_i - d_j)$ and compute $\sigma^2_{2r} = \|V_r(Y_i - Y_j)\|$, where Y_i and Y_j are the relative feature vectors at depth pixels i and j, respectively.

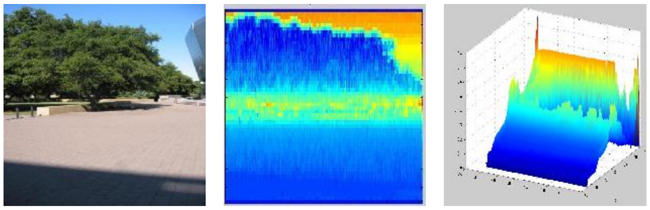
Once the parameters are estimated from the training data, given a test image, we use the Maximum A Posteriori (MAP) estimate to find the depth map. MAP is implemented as an iterative gradient descent algorithm, where the depth at i^{th} pixel in the $(s+1)^{st}$ iteration is given by:

$$d_i^{s+1} = d_i^s - \lambda \nabla E_{\sigma,\theta}(d_i, X_i) \tag{3}$$

$$\nabla E_{\sigma,\theta}(d_i, X_i) = \left(\frac{1}{\sigma^2_{1r}} + \frac{1}{\sigma^2_{2r}}\right)^{-1} \left[\frac{X_i\theta_i}{\sigma^2_{1r}} + \sum_{j \in N_{s(i)}} \frac{(d_i - d_j)}{\sigma^2_{2r}}\right] \tag{4}$$

where λ is the learning rate.

Further smoothing is achieved by sampling the depth points from each row, applying linear interpolation and fitting a spline. Figure 3 shows a typical output of this algorithm.

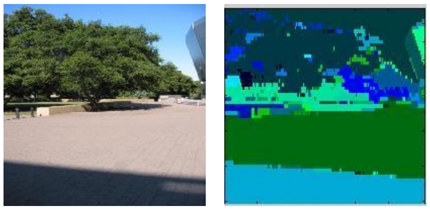


(a) (b) (c)

Fig. 3. (a) Original. (b) Estimated depth map. Note the smoothing of the ground plane irrespective of the shadowed region. (c) It shows the same plot from the side to show explicitly the smoothing of the ground plane texture.

B. Texture segmentation

A problem with the depth-map created from the above algorithm is the loss of information regarding the plane boundaries. We can't differentiate if a particular variation in the depth is due to an error in the estimation or simply due to depth discontinuities. Furthermore, even after multiple smoothing of the planes, it still has irregularities as is seen in Figure 3(c). In order to counteract this problem, we divide the original image into regions of perceptually similar textures called super-pixels based on the algorithm described in [5]. Our assumption is that regions having similar textures are more likely to lie on the same plane and thus even if the depth map gives irregularities in estimation, this could be corrected from the information resulting from image segmentation. Figure 4 shows the output of superpixelation. In the next section, we propose a strategy to merge coplanar superpixels, thereby determining the ground plane.



(a) (b)

Fig. 4. (a) Original Image. (b) The Superpixels. Each superpixel is a uniquely colored region in the graph. Note in (b) that the shadow in the image falls in a separate superpixel than that for the ground plane.

C. Inferring the ground plane

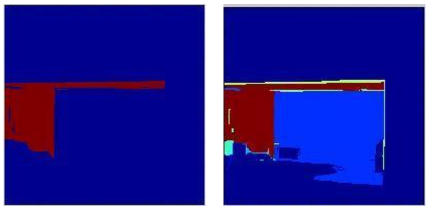
As discussed in the previous sections, the MRF based algorithm gives us an estimate of the depth at each pixel, but we lose information regarding the planes in the image. The superpixel algorithm provides planes in the image, but it lacks information regarding coplanarity. In this section we describe an algorithm to combine both these approaches to get a more robust estimation of the ground plane.

We consider the superpixels as being the nodes of a graph and adjacent superpixels are linked with an edge in the graph. Since the output of the superpixeling algorithm is a set of complex shaped uniquely colored regions, we devise a strategy to get an adjacency matrix from the superpixel graph. The algorithm is depicted below:

```
Input: superpixel_image
Output: adj_matrix
$edges$ := EdgeDetector($superpixel\_image$);
foreach $edge$ in $edges$ do
    foreach $nibor\_color$ in niborhood of $edge$ do
        if $nibor\_color \neq color\_of\_edge$ then
            adj_matrix($color\_of\_edge$, $nibor\_color$) = 1;
        end
    end
end
```

Algorithm 1: Finds the adjacency matrix from the superpixel graph

In Algorithm 1, EdgeDetector() takes the superpixel image and produces an edge map from the superpixels. Since these edges are boundaries of the superpixels, a check of the colors of the regions around an edge will provide information regarding the neighborhood. The algorithm operates in O(E) time, where E represents the number of edges in the graph. Figure 5 shows an output of a superpixel and the neighboring superpixels identified by the above algorithm. Once the



(a) (b)

Fig. 5. (a) shows a typical super pixel and (b) shows its detected neighbors from the adjacency matrix.

adjacency matrix is found, we perform a Breadth First Search (BFS) over the graph moving across superpixel planes and checking if neighboring planes are coplanar. We assume that the robot is located on the superpixel plane at the bottom center of the image. Thus, it is highly likely that the best path for robot navigation will be coplanar to this plane[1]. We start the BFS from the superpixel at the bottom center of the segmented graph, and work our way upwards through the neighbors from the adjacency matrix. For each neighboring superpixel, we sample n points, map these points to the corresponding points in the depth map we found in Section II.A, and then we fit a 3D plane into these n points using the Moore-Penrose pseudoinverse. Once the normal unit vectors to these planes are found, we check the cosine similarity between these vectors. Those planes having high cosine similarity are more likely to be coplanar, and thus those superpixels are merged. Figure 6 shows the schematic of the scenario.

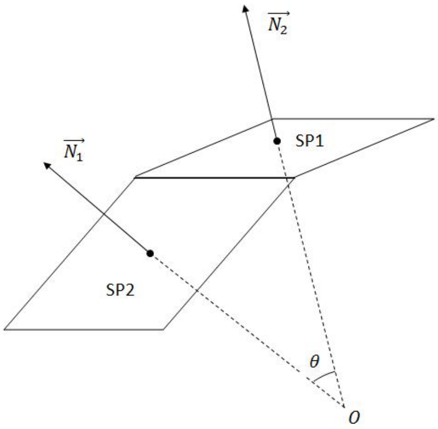


Fig. 6. SP1, SP2 are two superpixel planes and $\vec{N_1}$ and $\vec{N_2}$ are normals to these planes. To check if two super pixels are coplanar, we compute the inner product between $\vec{N_1}$ and $\vec{N_2}$ checking if it is closer to unity.

D. Building the reference plane from ground plane

Now that the ground plane has been identified, we can use the calibration of the camera and assumptions about its Euclidean coordinates, to find the correct distance map. Figure 7 depicts the scenario.

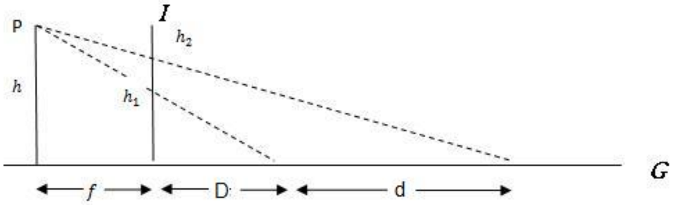


Fig. 7. Schematic of the camera structure.

Here P is the optical center, h the height of the camera, I the image plane, G the ground plane, f the focal length, and D the actual depth of the bottom most row of the image on the ground plane from the image plane. In addition, d is the depth, h_1 and h_2 show the heights of the projected ground points on the image plane.

Once the ground plane is determined, the given image is first inverse calibrated using the camera calibration matrix. We use the following equation to determine the actual depth of each pixel:

$$d = D * \frac{h}{h_1} * \frac{h_1 - h_2}{h_2 - h}. \qquad (5)$$

III. Experiments and results

In the experimental results discussed below we used an MIT implementation of the superpixeling algorithm for texture segmenting the image. All other algorithms were implemented in Matlab. We also implemented the MRF based depth map creation algorithm and used the Stanford Make3D 400 images/depthmaps data set for training and testing our implementation. Out of the 400 images in the set, we used 200 of the images for training the system and used the rest for testing. Each image was of size 1704 x 2272 and each depth map of size 55 x 305. On a PC running on a 2GHz Pentium processor and 2GB RAM, our implementation took 1-2 seconds per image for producing the 3D reconstruction.

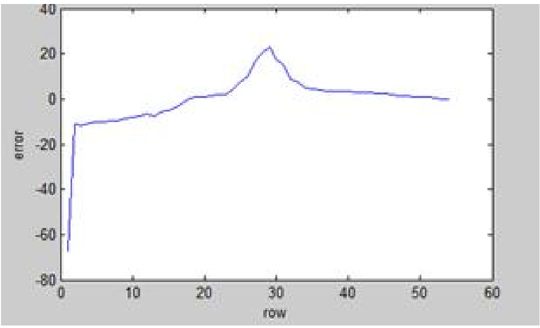


Fig. 8. It shows the mean error (in meters) between the real and predicted depths against image rows.

We found that the output of our implementation gave very good depth maps for the rows above 35 and below 25. A plot of the error in the estimation is shown in Figure 8. As it is

[1] This assumption is not entirely valid when the pose of the robot changes. In this situation we need to find the entropy of the superpixels to determine the best path suitable for robot navigation.

category	*mean rms error*(m)	*standard deviation*
Campus	0.374	0.082
Forests	0.707	0.246
Roads	0.316	0.099
Overall	0.397	0.269

TABLE I

MEAN RMS ERROR AND STANDARD DEVIATION COMPUTED BETWEEN THE RECONSTRUCTED 3D AND THE CORRESPONDING LASER DEPTH MAP.

seen from Figure 8, the error is quite high around the mid row of the images. This is due to the robot mounted camera and the variation in depth across the training images is quite high along the horizon where the ground and vertical planes meet. Thus our algorithm will not provide a good estimate of depths at these points. However this is a problem that we can accommodate, since we are only interested in building the ground plane and therefore we do not need to consider the depth map this far.

An empirical comparison of our algorithm with the given laser data is provided in Table I. Since we do not know the Euclidean coordinates of the camera, we made an estimate of them from the images and the corresponding laser data. The data set was segregated into images of campus, roads, and forests. As seen from the table, the forest group shows the maximum error, because of the noisy environment with too many texture variations. In our experiments we found that in many other instances the output of our algorithm looked superior to the laser depth maps. This is because the laser depth maps degrade with distance from the camera, but our algorithm uses the predicted depth map and the plane similarities to increase the accuracy in estimating the ground plane. Figures 9 through 11 show the estimation of the ground plane in various scenarios. The predicted ground plane in each of these situations is shown in green. Figure 12 shows two samples of the 3D environment reconstructed from the image. Figure 13 provides a comparison of our method with the raw output of the MAP estimation. Figures 14 and 15 show comparisons of the predicted depth maps with the laser range data.

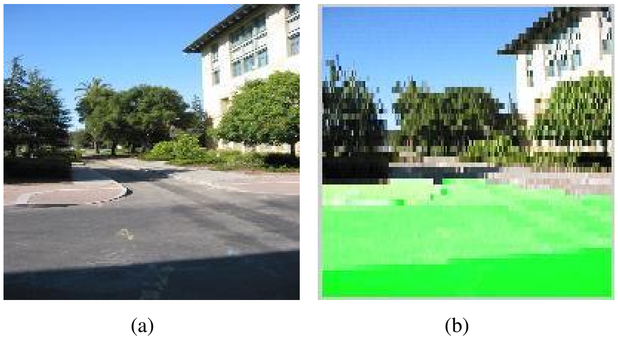


Fig. 9. (a) Original. (b) The image shows the estimated ground plane in green color. The robustness of the algorithm to shadows is depicted here.

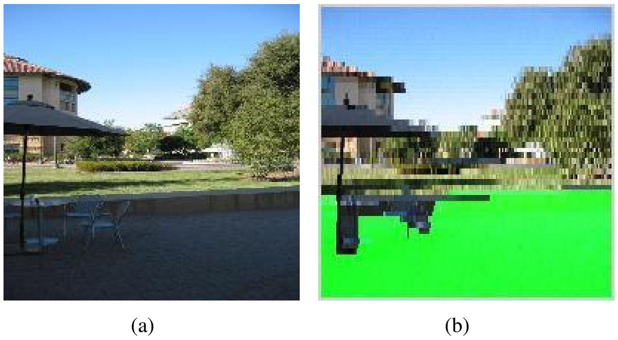


Fig. 10. (a) Original. (b) The image shows the estimated ground plane in green color. The algorithm finds out obstacles in the image and considers that in the ground plane estimation.

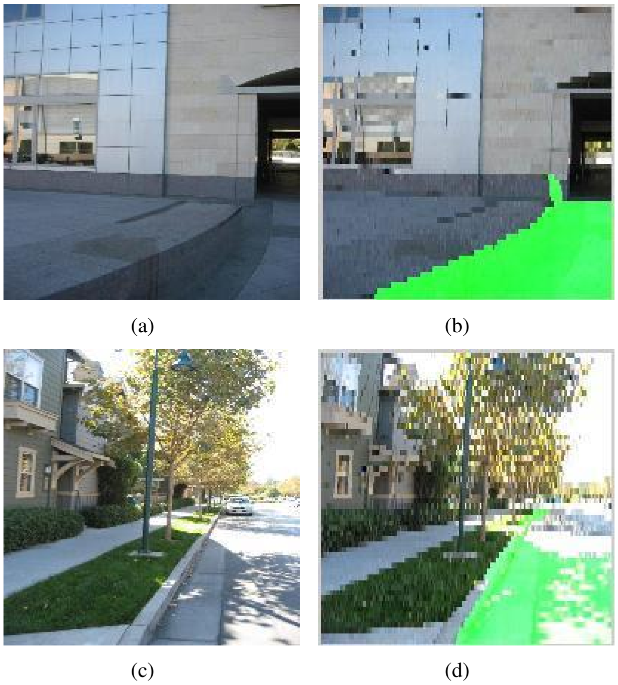


Fig. 11. (a) and (c) show the original images. (b) and (d) show the estimated ground plane in green color. This exemplifies the robustness to texture differences in the ground plane and also the identification of obstacles in the path.

IV. Conclusion and future work

In this work, we introduced a framework for estimating the ground plane accurately and showed how to map the pixels in a single image to 3D Euclidean coordinates with respect to the camera center. It was found that combining the predicted depth map with the superpixel algorithm results in better ground plane estimation. The new algorithm was found to be robust against shadows or texture differences on the ground plane while simultaneously it facilitated detection of obstacles in the robot navigation path. The algorithm could be improved for irregularly illuminated regions like forests over which its performance was not very impressive. In the present work we did not consider the intricacies associated with robot motion, nor have we investigated on how to combine 3D reconstructions from multiple views. These are

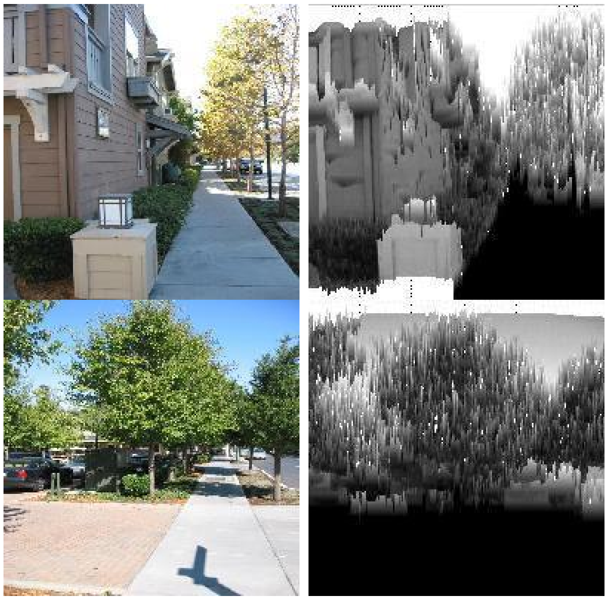


Fig. 12. Left figures show the original images. Right figures show the 3D reconstructions based on the estimated ground plane. The black region in the images on the right shows the estimated ground plane.

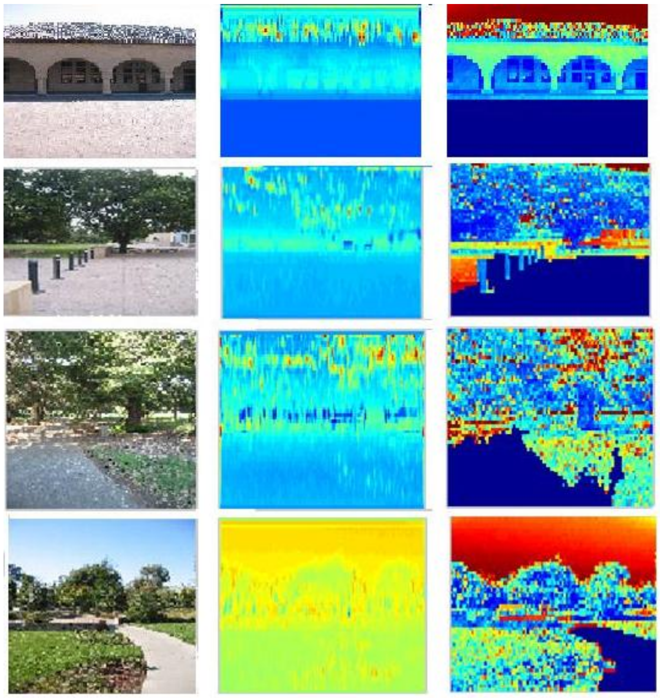


Fig. 13. Comparison of final predicted depth map with the raw output of the MAP estimation (before using superpixel information). Each row shows the original image, the image predicted after the MAP estimation, and the final depth map after using the superpixel information respectively. The region marked in blue shows the ground plane identified as navigable.

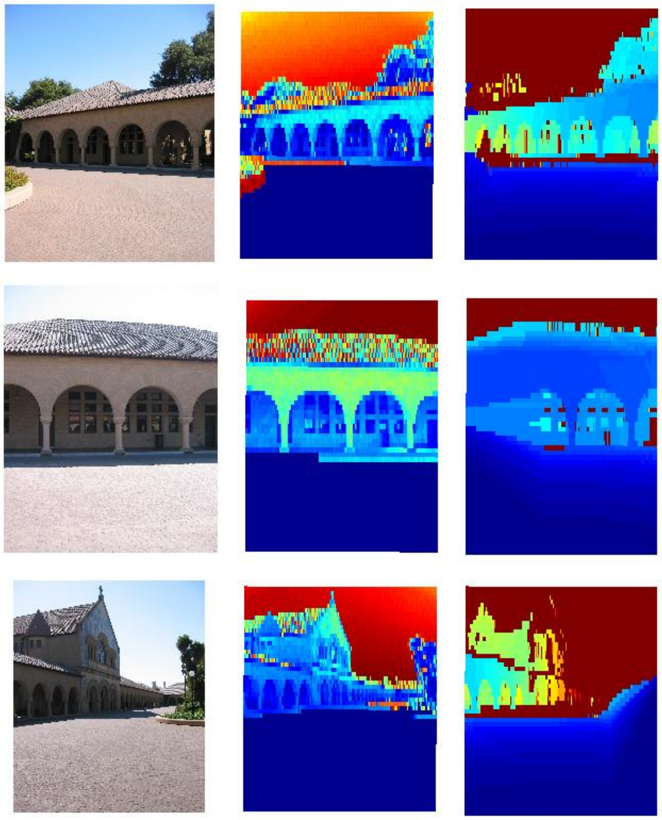


Fig. 14. Comparison of the 3D reconstruction with the laser depth data. Each row shows the original image, the predicted depth map, and the laser depth map, respectively.

topics for future research.

V. Acknowledgements

This work was supported in part by the U.S. ARMY (ARO) through contract #W911NF-08-1-0463 (Proposal 55111-CI), and the National Science Foundation through Grants #CNS-0324864, #CNS-0420836, #IIP-0443945, #IIP-0726109, and #CNS-0708344.

References

[1] A. Criminisi, I. Reid, and A. Zisserman. Single view metrology. *IJCV, 40:123-148*, 2000.

[2] A. Saxena, S. Chung, and A. Ng. 3-D depth reconstruction from a single still image. *IJCV*, Aug 2007.

[3] A.M. Loh, and P. Kovesi. Estimation of surface normal of a curved surface using texture. *In Proceedings of the Digital Image Computing: Techniques and Applications Conference*, 2003.

[4] D. Hoiem, A. Efros, and M. Hebert. Automatic photo pop-up. *In ACM SIGGRAPH*, 2005.

[5] D. Huttenlocher, and F. Felzenszalb. Efficient graph based image segmentation. *IJCV. Vol 59*, 2004.

[6] E. R. Davies. Machine vision-theory, algorithms and applications. *Morgan Kauffman Press*, 2005.

[7] E. Delage, H. Lee, and A.Y. Ng. A dynamic Bayesian network model for autonomous 3d reconstruction from a single indoor image. *CVPR*, 2006.

[8] M. Sun, A. Saxena, and A. Ng. 3-D scene structure from a single still image. *In ICCV workshop on 3D Representation for Recognition (3dRR-07)*, 2007.

[9] R. Nevatia, and K. Babu. Linear feature extraction and detection. *Computer Vision, Graphics, and Image Processing, 13:257- 269*, 1980.

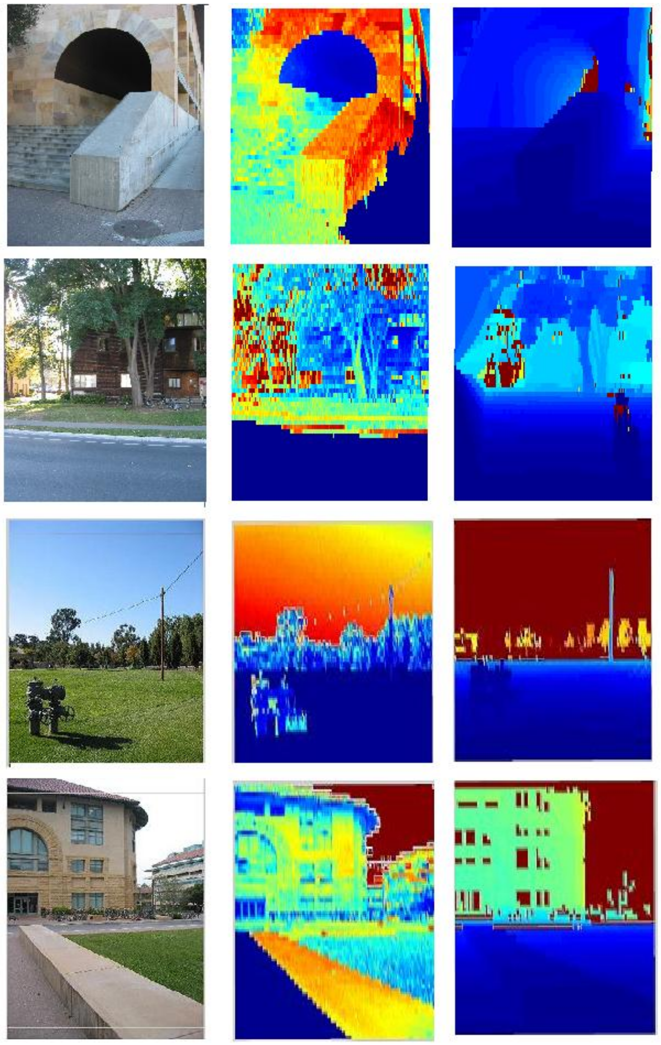


Fig. 15. Comparison with laser depth data. Each row shows the original image, the predicted depth map and the laser depth map respectively.

[10] R. Zhang, P. Tsai, J. Cryer, and M. Shah. Shape from shading: A survey. *IEEE PAMI, 21:690-706*, 1999.

[11] S. Chaudhuri, and A. N. Rajagopalan. Depth from defocus: a real aperture imaging approach. *Springer Verlag*, 1999.

[12] S. Geman, and D. Geman. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 6:721-741*, 1984.

[13] S. Lee, S.C. Ahn, and A. Meyyappan. Depth from magnification and blurring. *IEEE International Conference on Robotics and Automation Proceedings*, 1997.

[14] S.H. Chung, A. Saxena, and A. Ng. Learning depth from single monocular images. *NIPS*, 2005.