Undergraduate Project 1
Real Time Obstacle Detection for Vehicles using Stereo Cameras

Prawal Gangwar and Akshat Agarwal

***Abstract*— With the demand of autonomous and driver assisting vehicles, the basic need to detect and estimate the distance of various obstacles on the road in real time is a big challenge. In this report we present a real time robust method to find the solution to this problem. The goal is to find detect obstacles using stereo images and map it with the real world coordinates.**

I. INTRODUCTION

The computation of free space available in the environment is an essential task for many intelligent automotive and robotic applications. The free space is the world region where navigation without collision is guaranteed. In automotive applications, efficient and robust calculation of free space becomes critical when in a dense environment, especially with pedestrians. In robotics, free space in the robot's environment is needed to plan the path between two points. Obstacle detection can be done with the help of input from various kind of sensors, such as RADAR and LIDAR sensors, which are very expensive and prohibit the technology from being scaled up to production level. Since metallic parts of vehicles reflect RADAR much more efficiently than human tissue, RADAR, while working well on sparsely populated roads, can have trouble detecting pedestrians in dense urban environments. Stereo cameras provide a cheaper solution than LIDAR, and a lot of research has been done in the area of obstacle detection and free space computation using forward facing stereo cameras.

Depth estimation from stereo cameras is done by computing the disparity between two images. Disparity refers to the difference in location of an object in corresponding two (left and right) images as seen by the left and right cameras which is created due to horizontal separation between the optical axes of the lenses. The disparity of a pixel is equal to the shift value that leads to minimum sum-of-squared-differences for that pixel. Computation of disparity maps has been done using various methods, such as belief propagation [1] and semiglobal matching [2]. A comprehensive evaluation of the available algorithms has been done on the Middlebury dataset [3], and it indicates that the semiglobal block matching method proposed by H. Hirschmuller is one of the most efficient and fast stereo reconstruction algorithm.

In this project we have implemented a system to compute the free space in front of the vehicle using input from stereo cameras. First, a disparity map is computed by the semi global block matching routine available in the OpenCV library [4]. This is used to compute a columnar occupancy grid, and further a map of the free space in front of the vehicle.

The method presented works at 3Hz on a Core-i5-4200 CPU with 1.6GHz clock frequency. Each frame takes about 330ms to process, and a majority of this time (300ms) goes into the disparity map computation. The method has been evaluated on the Bahnoff dataset [5] which consists of a series of images taken from a car moving in a high density urban environment with pedestrians, park benches, trees and poles being some major obstacles present in its path. The images are rectified and undistorted.

II. ALGORITHM DESCRIPTION

This section describes in detail the procedure followed for arriving at an estimate of the free space in front of the vehicle. The methods have been tested on the Bahnoff dataset.

A. Depth Map Computation

As previously mentioned, the depth map has been computed by the Semi-Global Block Matching method implemented in OpenCV. The method was used due to it's proper balance of speed and accuracy, which is reflected in its high rating on the Middlebury dataset rankings.

The algorithm is based on the idea of pixelwise matching of Mutual Information. The cost aggregation involves searching in multiple directions to enforce a global smoothness constraint, hence taking into account the disparities of neighbouring pixels while calculating the disparity of a particular pixel, thereby reducing noise in the form of false positives.

The SGBM routine offers 11 parameters for creating disparity maps in different environments (dense/sparse), for enforcing smoothness constraints, noise removal and number of directions checked while enforcing the constraint. To get the best results on our dataset, we created a GUI (Fig. 1) with trackbars for tuning these parameters and checking the results in real time. This enabled us to get the best possible results from the algorithm, as shown in Fig. 2.

B. Elimination of Noise Speckles

Even though the algorithm was able to give sufficiently good disparity data it had some issues with the noise. Some parts of the disparity image showed some very high intensity points which had no mapping in the real image. Exact reason for this is unknown but the observation of it's non-repetitive behaviour in the image sequence helped us to solve the problem.

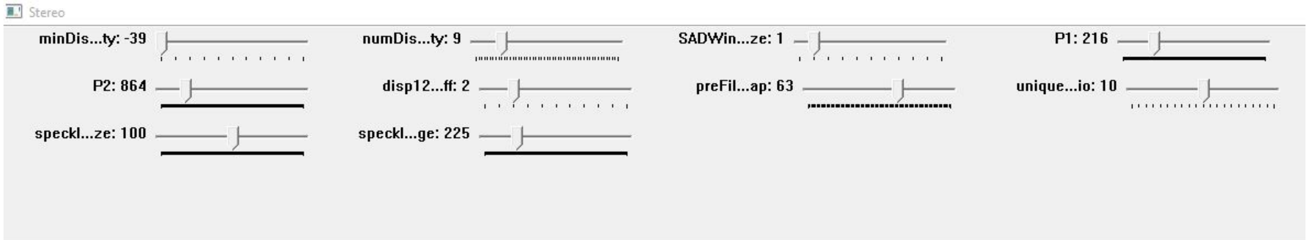


Fig. 1: The GUI used for parameter tuning

(a) Left Image (b) Right Image (c) Disparity Calculation using SGBM

Fig. 2: Diparity Image Calculation form the two Stereo Images

The assumption that full algorithm is fast and will work on a adequate frame rate of the images will let us assume that there has been not much change in the 3 continuous images. Taking a weighted mean of the past 3 disparity maps and eliminating the points(from the current disparity image) whose intensity is greater by a threshold lets us eliminate the noisy data from the images. The results can be seen in Fig. 3.

C. *Detection and Elimination of Road and Sky*

The removal of road from the image data was important because the non-uniform texture and noise of the road was also detected as obstacle and it's removal was necessary to compute the free space where the vehicle had to move with least possibility of collision.

The method used to eliminate the road as well the sky involved use of watershed algorithm [6] to segment the road and sky in the image as shown in the 5b by using the sum of output of multiple Erode and Dilate operations on the original images as markers. The resulting binary image is then inverted and element-wise multiplied to the disparity image to clear off all the details from road as well as the sky.

The Dilate function is basically convolution of an image with a kernel such that this operation causes bright regions within an image to "grow" (therefore the name dilation). Similarly, the Erode function is just the opposite. Instead of causing the bright regions to "grow" it shrinks them and causes the dark zones to get "bigger". The watershed algorithm performs marker based image segmentation of the road as described in [6]. The marker, which needs to be provided manually, was given as the result of Erode+Dilate images, the results of which can be seen in Fig. 4.

D. *Columnar Occupancy Grid*

Traffic scenes typically consist of a relatively planar free space limited by 3D obstacles that have a nearly vertical pose. The idea is that the ground plane has monotonically decreasing depth (disparity) along each row, moving upwards from the bottom of the depth map, and obstacles represent a discontinuity in the depth profile of the ground plane. This concept was used to approximate the obstacles by rectangular blocks of a certain width and height. It offered a significant reduction in data volume while preserving information of interest.

A higher disparity corresponds to a closer obstacle, since the angle of light falling on the camera lenses is greater for objects closer to the lenses. This along with the vertical obstacle approximation was used, by finding the maximum value of disparity in each column, and interpreting it as the disparity (or depth) corresponding to the obstacle closest in that column. This led to a columnar occupancy grid being obtained, which can be seen in Fig. 6.

E. *Computation of Free Space*

The data obtained from the columnar occupancy grid was used to build a free space map (Fig. 6), by using the relationship between disparity in pixels and depth in the real word:

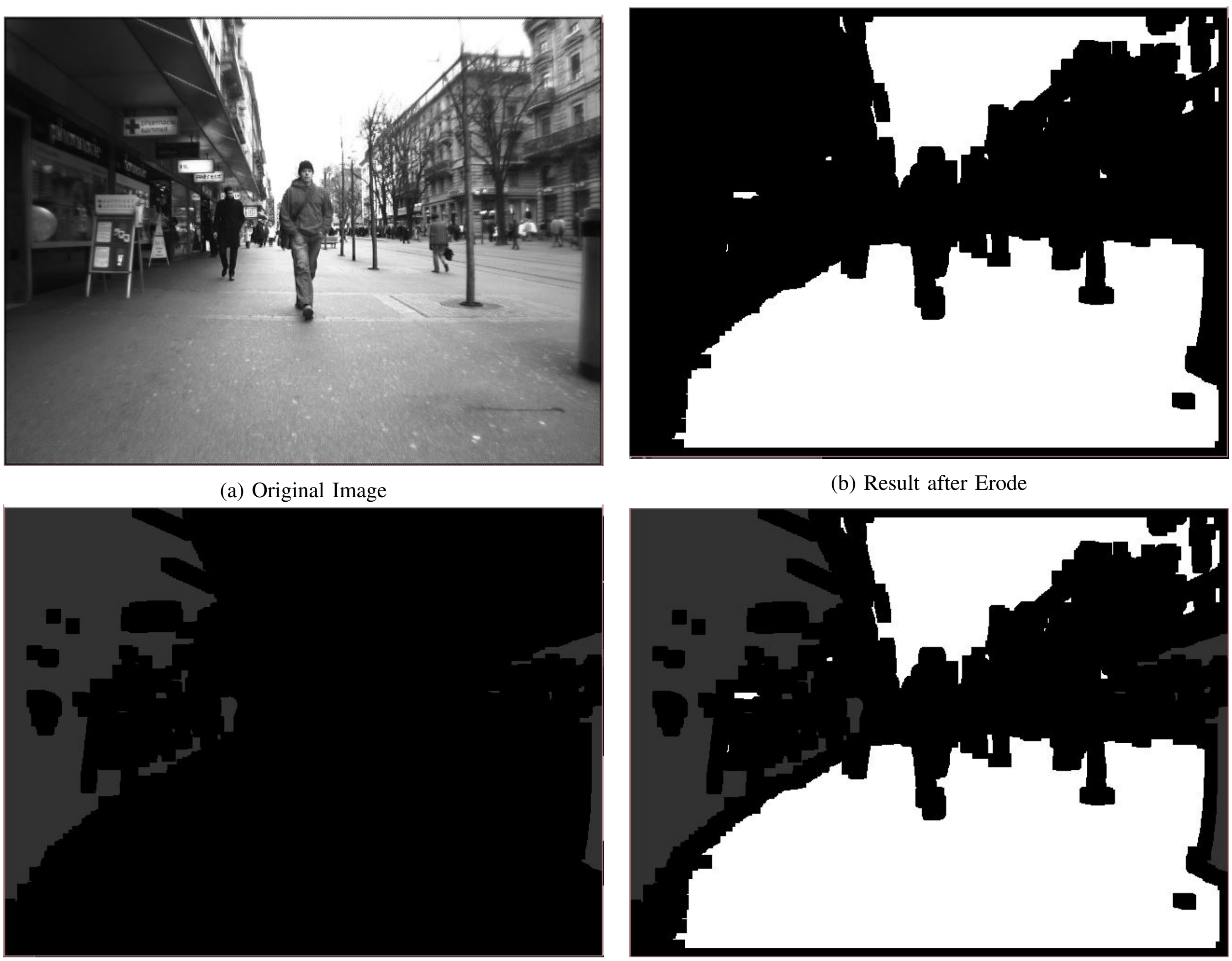
$$z = \frac{f \times b}{d} \tag{1}$$

where z is the real world depth, f denotes the focal length divided by an appropriate scale factor, b the baseline distance and d the disparity. From the camera calibration matrix available, $f = 500.6mm$ and $b = 400mm$. To retain information

(a) Disparity with Noisy Speckles

(b) Disparity with Noisy Speckles Removed

Fig. 3: Speckle Removal from the Disparity Images



(a) Original Image

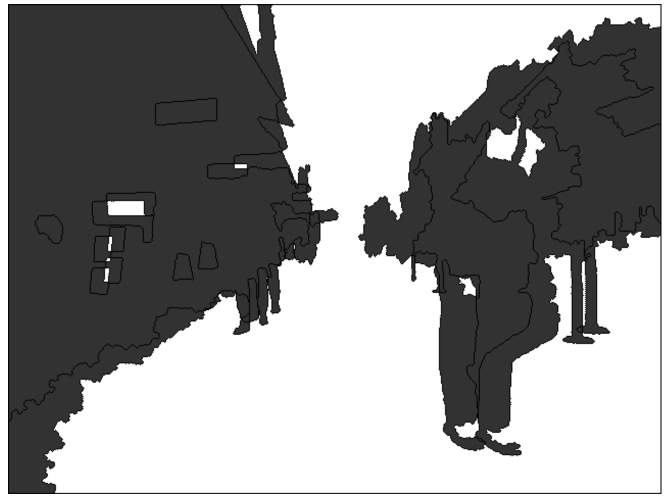
(b) Result after Erode

(c) Results after Dilate

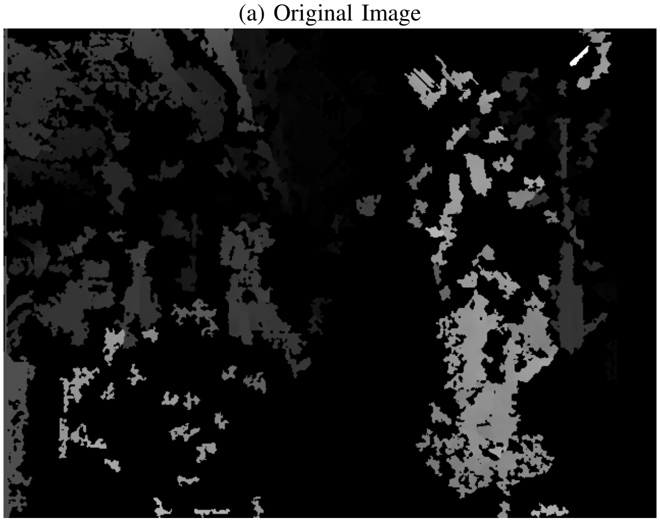
(d) Erode + Dilate

Fig. 4: Application of Erode and Dilate functions in Sky and Road Detection

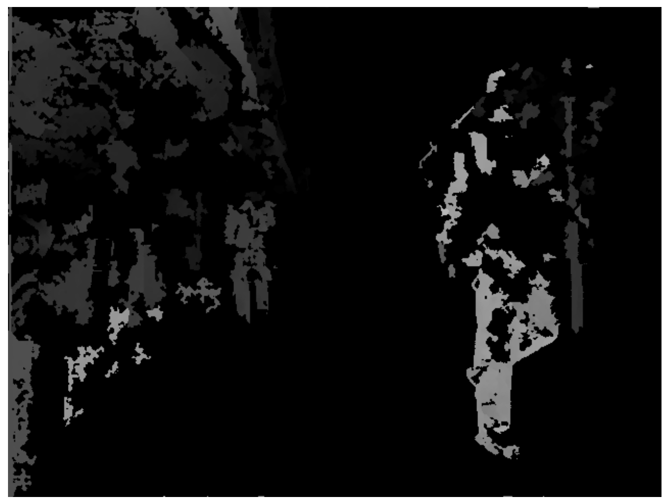
(a) Original Image



(b) Road and Sky marked as White area



(c) Disparity with Road and Sky details

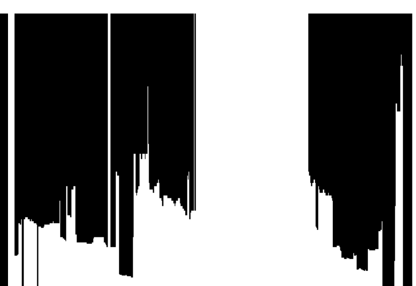


(d) Disparity with Road and Sky details removed

Fig. 5: Road and Sky Detection and Removal from Disparity Image

(a) Original Image

(b) Occupancy Grid



(c) Free Space Plotting

Fig. 6: Free Space and Occupancy Grid Estimation

about obstacles, the depth was saturated at an upper bound of 10m. Since any information about obstacles further than that would not be reliable, they were not considered while mapping the free space.

III. CONCLUSION

The columnar implementation of occupancy grid and free space reduced computational time while preserving information of interest, however the SGBM algorithm took 300ms to run on a CPU. However the algorithm contains some computations that are embarrassingly parallel, and as such a GPU would be able to reduce the running time of the SGBM algorithm by a great margin. The current algorithm can be improved to deal with the following problems to get better details and a more robust detection.

- Untextured background
- Specular surfaces (shiny surface with light reflecting from it)
- Shadows on the ground and roads
- Perspective distortion
- Repetitive patterns
- Non-Uniform Road Surface

REFERENCES

[1] J. Sun, N.-N. Zheng, and H.-Y. Shum, "Stereo matching using belief propagation," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 25, no. 7, pp. 787–800, 2003.

[2] H. Hirschmüller, "Stereo processing by semiglobal matching and mutual information," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 30, no. 2, pp. 328–341, 2008.

[3] D. Scharstein and R. Szeliski, "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms," *International journal of computer vision*, vol. 47, no. 1-3, pp. 7–42, 2002.

[4] G. Bradski, *Dr. Dobb's Journal of Software Tools*.

[5] A. Ess, B. Leibe, K. Schindler, , and L. van Gool, "A mobile vision system for robust multi-person tracking," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'08)*. IEEE Press, June 2008.

[6] J. B. Roerdink and A. Meijster, "The watershed transform: Definitions, algorithms and parallelization strategies," *Fundam. Inform.*, vol. 41, no. 1-2, pp. 187–228, 2000.