3D Imaging using Stereo Matching

A report of Project on Image and Video Processing (EC348)

Submitted by

Chandravaran Kunjeti	181EC156
Saikumar Dande	181EC140
Roshan Rangarajan	181EC139

Under the guidance of

Dr. Shyamlal

in partial fulfilment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING
NATIONAL INSTITUTE OF TECHNOLOGY KARNATAKA
SURATHKAL, MANGALORE - 575025

Contents

1 Abstract

We present the 3D mapping and depth mapping of environments using stereo cameras. This is achieved through first obtaining images through stereo cameras. The framework used first obtains depth and disparity maps through algorithms like Census Transform(CT) as well as Sum of Absolute Difference(SAD) Algorithm to implement block matching. These disparity maps are then converted into point clouds using Open3D. The results obtained from the different methods are also observed, compared, and the results are shown. These algorithms are first tested on stereo datasets before being implemented in the Gazebo simulation environment. A novel method is also proposed which reduces the errors when compared with the existing algorithms. This is further tested on an actual robot. In contrast to common existing methods used to calculate disparity maps, we have used Multi-Block Matching along with SAD and CT to perform block matching. This reduces the bad error when compared with the ground truth.

2 Introduction

Over the years, robots have gained significant importance and have helped alleviate some of the problems faced in many fields such as construction, medicine, and technology. For the robot to interact with its environment, it needs to understand what is around it and where it is. This is achieved through recognition and localization of the environment and recreation of it. To recreate the environment, 3D mapping finds an important application. It finds applications in things such as 3D television, intelligent robotics, medical imaging, and stereo-vision.

One of the first steps is to calculate the depth of the objects present in the environment. There are a variety of methods generally used to obtain depth. These include LIDAR, IR, Stereo, Motion Based, and One Shot Structured light to name a few. LIDAR cameras provide accurate depth but are found to be sparse in vertical and horizontal resolution. IR cameras are often used to detect depth, however they often have limitations in the distance they can measure and also run into problems when the environment contains shiny or transparent objects. Active illumination-based methods are found to either have a good acquisition time or a good resolution of the 3D shapes, but not both [8]. One-shot methods are sensitive to the texture of the surface and can only recreate sparse reconstructions. In this paper, we have used images obtained from a stereo dataset.

While calculating depth, it is often convenient if the two images obtained from the stereo cameras only differ in the horizontal direction. To achieve this, cameras have to go through the processes of rectification and calibration. Stereo calibration is firstly done to remove any lens distortions that may exist. Further rectification is done to ensure that a pixel P if present at (x1,y) in the first image, can be found in the same horizontal line at (x2,y) in the second image. This is often achieved through photogrammetric calibration as shown in Figure 1, by observing a stationary object such as a chessboard and using rotation and projection matrices to rectify and calibrate the camera.

Once the images are obtained from the stereo cameras, they then need to be processed accordingly to obtain a disparity map. A disparity map calculates the apparent motion present for the same pixel in the two stereo images. This is achieved through algorithms such as Census Transform, Sum of Absolute Differences, and Multi-Block Matching. The depth map can then further be calculated using the formula:-

$$depth = \frac{(focal\ length\ of\ camera) * (baseline)}{disparity} \tag{1}$$

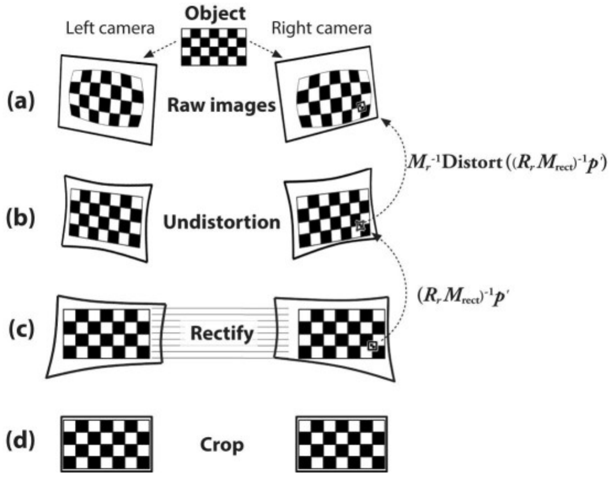
Left camera
Object
Right camera
(a)
Raw images
M_r^{-1}Distort$((R_r M_{\text{rect}})^{-1} p')$
(b)
Undistortion
$(R_r M_{\text{rect}})^{-1} p'$
(c)
Rectify
(d)
Crop

Figure 1: Stereo Camera Rectification

The disparity maps further need to be converted into a 3D point cloud. A point cloud is a collection of pixels in a 3-dimensional space. The rotation and translation matrices are obtained from the images and can be further applied to point clouds to achieve a complete 3D reconstruction of the environment.

3 Literature Survey

In recent years, various methods have been developed to generate sparse and dense disparity maps. In order to generate a disparity map, for every pixel in the left image we need to find the corresponding pixel in the right image, and for that there are various methods. Single block matching techniques uses a certain number of pixels in the given block size to match the left and right pixels.

In the paper[1] instead of using a single block for matching, multiple blocks of different sizes have been used in order to find the better correlated pixel. Multi block matching method gives better results in the cases where the image contains highly slant surfaces for example an image containing a road. In this paper[3] the size of the block changes adaptively based on the surrounding pixel intensities. This method gave much better results as compared to single block matching. There are different block matching methods. Block matching using Sum of Absolute difference, Normalized cross correlation and Census transform. In this paper[4] they proposed a modified census transform method, which gave better results and is more robust as compared to the ordinary census transform method.

4 Backgound(Description of Dataset)

Two datasets have been used to compare and analyse the different models in this paper. These are the 2001 and 2003 Middlebury Stereo Datasets.

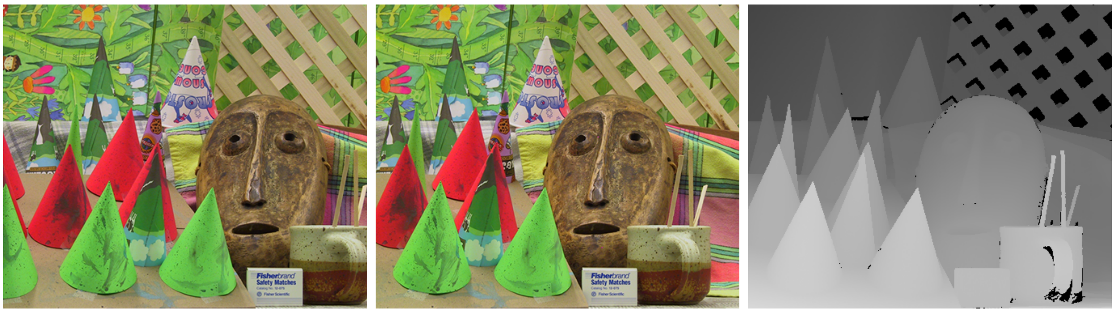

Figure 2: Cones left, right and Ground truth disparity Map image from 2003 Middleberry dataset

4.1 2001 Stereo Vision Middlebury

The 2001 dataset [6] consists of 6 sets of 9 images of piecewise planar scenes with ground-truth disparity maps for images 2 and 6. Every ground-truth disparity map has a total of 32 disparity levels and it is scaled up by a factor of 8.

4.2 2003 Stereo Vision Middlebury

The 2003 dataset [5] consists of 2 sets of 9 color images with ground-truth disparity maps for images 2 and 6. The 9 images have all been rectified so the image motion can be observed purely in the horizontal direction. Every ground-truth disparity map is obtained using structured light and is also scaled down by a factor of 4. The disparity Map has a total of 64 disparity levels. Figure 2 shows the Cones left, right and disparity map data.

5 Methodology

Stereo matching is a 3D reconstruction method that uses two different cameras separated by a fixed distance with their optical axes parallel. An important parameter to note is the **baseline** which is the distance of the line connecting the centre of the cameras, which is perpendicular to the line of sight of cameras. Now let us assume a 3D point is represented as (x,y,z) in world coordinates. A point in the left image is represented by ($\mathrm{x_{left}}$,$\mathrm{y_{left}}$), similarly a right image point is ($\mathrm{x_{right}}$,$\mathrm{y_{right}}$), and the focal length is f and the baseline is b. Then by using similar triangles we have

$$\frac{x_{left}}{f} = \frac{(x + b/2)}{z} \tag{2}$$

$$\frac{x_{right}}{f} = \frac{(x - b/2)}{z} \tag{3}$$

$$\frac{y_{left}}{f} = \frac{y_{right}}{f} = \frac{y}{f} \tag{4}$$

The above equations can be solved us for (x,y,z) which are given by the following equations

$$x = \frac{d(x_{left} + x_{right})}{2(x_{left} - x_{right})} \tag{5}$$

$$y = \frac{b(y_{left} + y_{right})}{2(x_{left} - x_{right})} \tag{6}$$

$$z = \frac{b * f}{(x_{left} - x_{right})} \tag{7}$$

in the above equations the denominator can be observed to be (x_{left}-x_{right}) this quantity is called disparity. We can define **disparity** as the difference of objects in two images as seen by left and right, which occurs due to parallax.

To calculate disparity there have been many methods that have been proposed which have been used by us in out method to get a single combined method. These methods are as follows

5.1 Sum of absolute Difference

Let us assume the images are rectified, essentially meaning the images have their epipolar lines parallel to the x-axis; also the dimensions of the left and right are exact. We will continue to use the same representation of a point in left and right image as mentioned above. In Stereo matching, we need to find for a particular point x_{left} the corresponding point x_{right}, Sum of absolute difference gives way in determining this point [1] . The algorithm is mentioned bellow

```
Algorithm 1: Sum of Absolute difference
Input: I_{left}, I_{right}, disparitylevels
Calculate error between left and right Images
while i < disparitylevels do
    Calculate pixel wise absolute difference and sum
    $error(i) = \sum_{W} I_{left}(u,v) - I_{right}(u+i,v)$
Compute the location of the min value
if error(i) is least value in array then i is disparity value;
Repeat Steps until the whole image is traversed
```

This method uses a single window, to reduce the error more we can use multi block matching [2], where we calculate the error after using for windows of different sizes. We have improved the speed of the python implementation by using vectorizing our code.

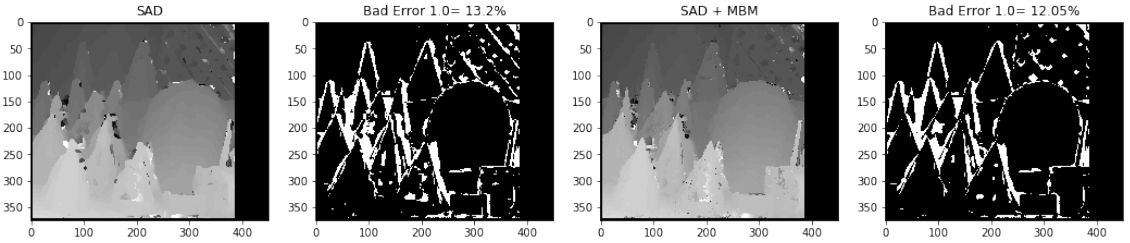

Figure 3: From left to right: SAD, Bad error with SAD, SAD with MBM, Bad error with SAD+MBM

5.2 Normalized correlation

Taking the same assumptions and conventions as before, we similarly need to find the location of the minimum error but the method of calculating this error is different. In algorithm 1 the error calculation step is replaced by the following equation

$$\frac{\sum_{(u,v)\in I} I_{left}(u,v) * I_{right}(u+i,v)}{\sqrt{\sum_{(u,v)\in I} I_{left}^2(u,v) * I_{right}^2(u+i,v)}} \tag{8}$$

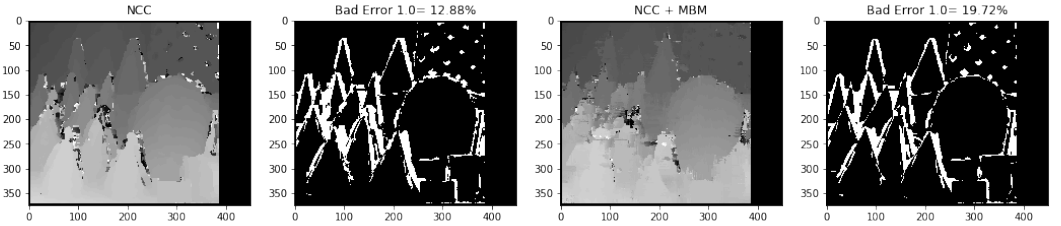

Figure 4: From left to right: NCC, Bad error with NCC, NCC with MBM, Bad error with NCC+MBM

5.3 Census Transform

This method was introduced by Woodfill and Zaibh [9], it was stated that it belongs to the class of non-parametric image matching. The Census Transform[7] C(p) can be considered as a non-linear transformation, where local neighborhood of a pixel p are mapped to a binary string that represents whose intensities are less or greater then that of p. Each Census digit is represented as

$$\upsilon(i,i')$$

where i and i' are the intensity values of the pixel p and neighbor of p

$$\upsilon(i,i') = \begin{cases} 0, & \text{if } i > i' \\ 1, & \text{if } i \leq i' \end{cases}$$

This can be easily understood from fig 5. Where the mapping is performed on both the right and left images, then the bit arrays are compared, the number of bits that are different will determine the hamming distance.

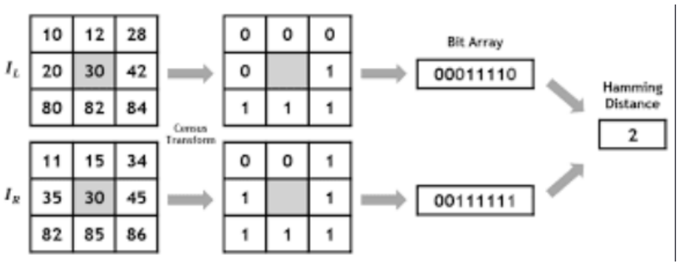

Figure 5: Census transform

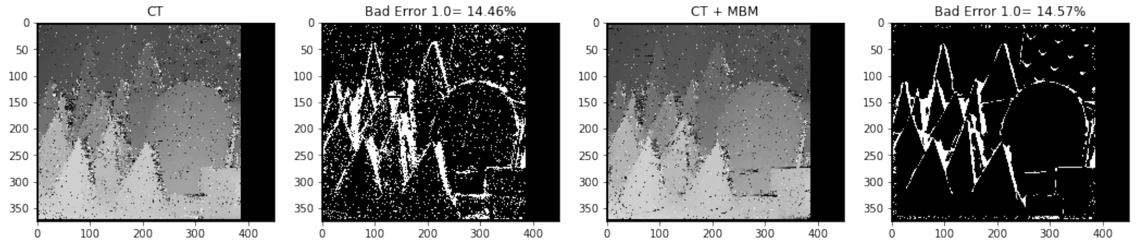

Figure 6: From left to right: CT, Bad error with CT, CT with MBM, Bad error with CT+MBM

5.4 Proposed Method

The model we propose takes the above methods' main components and gets a new and improved model. In this method, the laplacian equation combines the costs of different methods. This final cost is then used to find the disparity levels. The methods incorporated in the model are Sum of absolute difference on gray and image gradients with multi-block matching, Census transform on gray and image gradients with multi-block matching.

The Image gradients are found in 2 directions the x and the y

$$\nabla I(x,y) = \frac{dI}{dx}$$

$$\nabla I(x,y)' = \frac{dI}{dy}$$

I(x,y) and I(x,y)' are combined with the gray scale image and we create an array of images. The error calculation for the these images for stereo matching are found using Sum of absolute difference, and census transform. Therefore we have calculated error in 3 different methods

- Error for gray, using Sum of absolute difference
- Error for gradient, using Sum of absolute difference
- Error for gray and gradient, using Census Transform

Using the laplacian equation it becomes easy to combine the error images to obtain a better result by giving choosing the appropriate weights. We chose SAD and Census transform for the new method as they were the fastest to compute and have least error. The three different Error images are represented as

$\mathrm{C}_{\mathrm{sad}_\mathrm{g}}$ for Sum of absolute difference and gray
$\mathrm{C}_{\mathrm{sad}_\mathrm{grad}}$ for Sum of absolute difference and gradient
$\mathrm{C}_{\mathrm{ct}}$for census transform for gray and gradients.

The weights are respectively λ_{sad_g} $\lambda_{sad_{grad}}$ λ_{CT}

The final laplacian equation is given is as follows

$$C_{final} = 3 - \exp\left\{\frac{-C_{\mathrm{sad}_g}}{\lambda_{sad_g}}\right\} - \exp\left\{\frac{-C_{\mathrm{sad}_{grad}}}{\lambda_{sad_{grad}}}\right\} - \exp\left\{\frac{-C_{\mathrm{CT}}}{\lambda_{CT}}\right\}$$

6 Post Processing

6.1 Locally Consistent Disparity Map

Locally Consistent Disparity Map or also known as LCDM is a widely used filtering technique for disparity maps, as it is observed that disparity maps consist of a lot of salt and peeper noise. We assume that the disparity map does not have any sudden change in the depth over a large regions. The algorithm has 3 main steps which are explained in 2

```
Algorithm 2: Locally Consistent Disparity Map (LCDM)
Step-1 Take a window $\mathbf{W} \in I(u, v)$
Step-2 Count the number of times a value occurs
Step-3 Replace the center pixel with the max occurring value
 Repeat Steps until the whole image is traversed
```

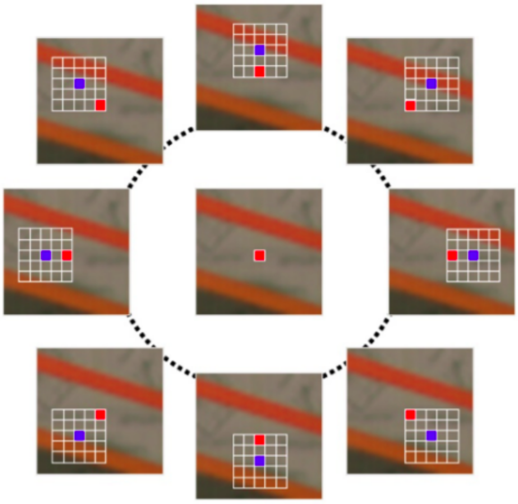

Figure 7: LCDM algorithm representation

7 Simulation Environment and Robot model

The algorithm was implemented on a robot model what was designed by us, it has 2 caster wheels and a 2 driven wheels and it consits of a stereo camera that is mounted on top of a vertical stand to have a higher view point. A model picture can be seen in figure 8

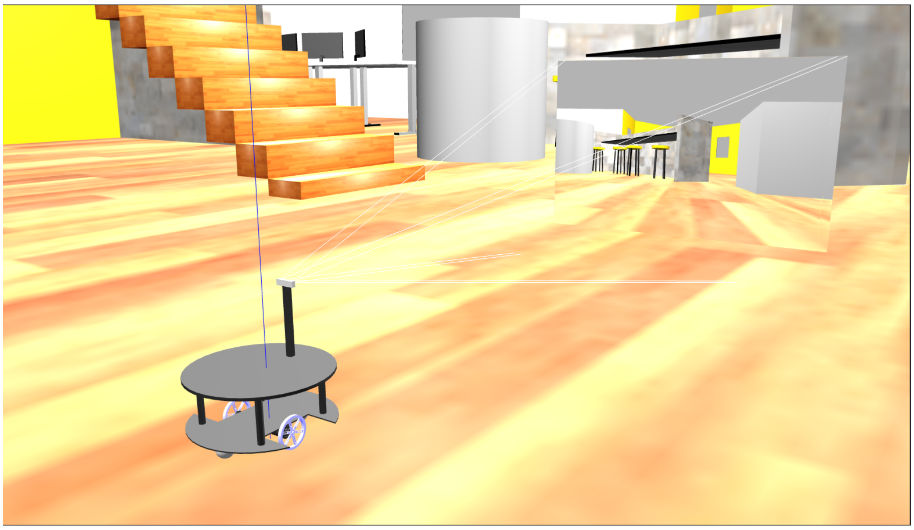

Figure 8: Simulation environment with the robot, includes objects of different shapes and dimensions

Objects such as stairs cylinder, cuboids, chairs, tables were included into the environment to test the robustness of the algorithm. Extra light has been added to the simulation environment to obtain more features, as it is very hard to obtain proper features from none illuminated environment.

8 Results and Discussions

As we can see from the figure 9 and Table 1, our proposed method gives much better results as compared to the other methods. To remove noise from the disparity map, we apply LCDM filter. In the table 2 we can see the time taken for different methods. Our method takes more time compared to other methods because we had to do extra processing to get a denser disparity map. It is a trade off between error and time. The time taken depends on the laptop. This code is ran on a i7 8th gen laptop. Figure 10 shows the implementation of the proposed model in gazebo.

Table 1: Errors calculated using different methods on Middleberry dataset

Method	SAD	SAD+ MBM	NCC	NCC+ MBM	CT	CT+ MBM	Proposed	Proposed+ LCMD
Cones	13.2	12.05	12.88	19.72	14.46	14.57	7.56	6.94
Teddy	15.0	13.29	13.31	24.07	18.55	19.52	8.86	7.53
Sawtooth	5.76	5.63	6.73	12.22	18.39	17.42	5.07	3.53
Venus	9.0	11.05	8.93	13.21	30.59	28.34	10.14	3.0
Barn1	4.15	3.65	4.39	6.76	14.35	13.06	3.65	2.33
Bull	4.61	5.53	3.86	7.52	20.55	19.66	4.24	1.59

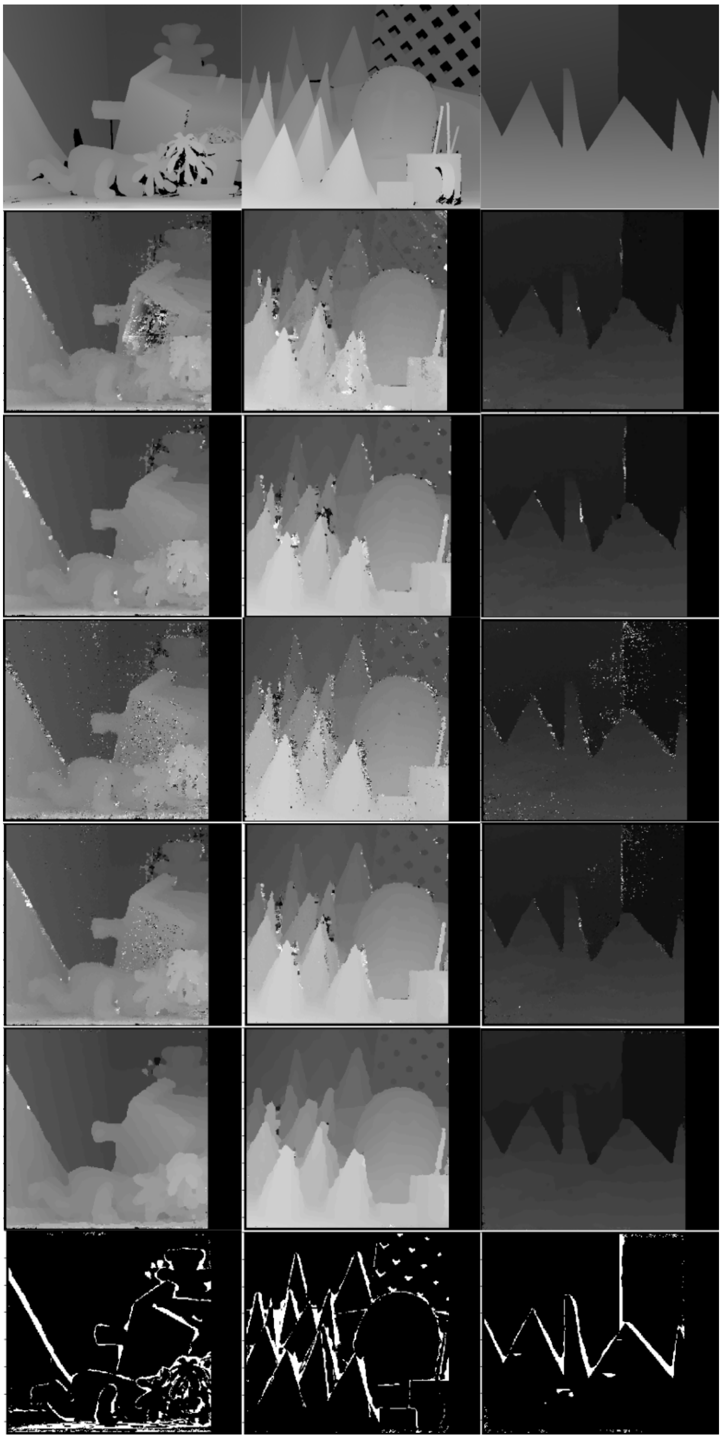

Figure 9: Comparing results of different matching methods. Going from Top to bottom: Ground Truth, SAD+MBM on gray image, SAD+MBM on image gradients, CT+MBM on gray and image gradient, Proposed method, Proposed method with LCDM, Bad error for the proposed result.

Table 2: Time taken(in secs) using different methods on Cone image(Middleberry dataset)

Method	Time Taken(in secs)
SAD	1.31
SAD+MBM	32.29
NCC	0.92
NCC+MBM	30.31
CT	15.7
CT+MBM	35.7
Proposed	51.63

- SAD : Sum of Absolute difference
- NCC : Normalized cross correlation
- CT : Census Transform
- MBM : Multi Block Matching
- LCDM : Locally Consistent Disparity Map

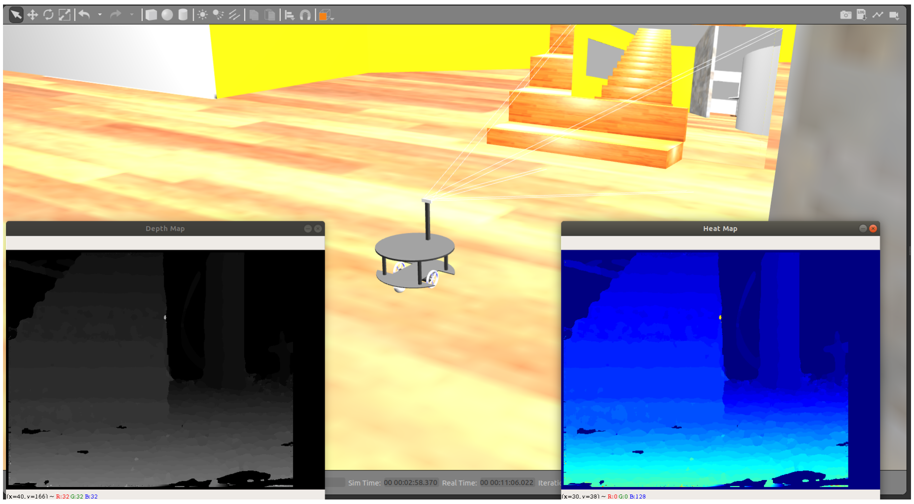
Depth Map
Heat Map
(x=40, y=166) ~ R:32 G:32 B:32
Sim Time: 00 00:02:58.370
Real Time: 00 00:11:06.022
Iteratio
(x=30, y=38) ~ R:0 G:0 B:128

Figure 10: The Gazebo output of the proposed model

9 Conclusion

In this paper, the existing methods used to find the disparity maps such as Census Transform, Sum of Absolute Differences and Normalised Correlation have been implemented and analysed using a stereo dataset. Further, a method has been proposed which combines Sum of Absolute Differences and Census Transform along with Multi Block Matching. This method was applied on gray images by taking gradients in the x and y direction and then using the laplacian equation combine all the methods. The disparity maps acquired through the different methods was compared with the ground truth disparity map and the errors were computed. Further, it was found that the proposed method gave significantly less errors as compared to the existing methods for disparity map calculation. The algorithm was also implemented on a robot model consisting of a stereo camera which was designed for this project. The disparity maps were also converted into a point cloud and visualised using Open3d.

This is link to the Code.

This is link to the Youtube video.

10 Future works

The future works include the following:

- Combining the depth images by using by registering the camera movement and applying homogeneous transform to the point clouds found.
- Improving the Execution time by coding in c++ and using boost library to fastly access the pointers

References

[1] Nils Einecke and Julian Eggert. "A multi-block-matching approach for stereo". In: *2015 IEEE Intelligent Vehicles Symposium (IV).* 2015, pp. 585–592. DOI: 10.1109/IVS.2015.7225748.

[2] Rostam Hamzah, Rosman Abd Rahim, and Zarina Mohd Noh. "Sum of Absolute Differences algorithm in stereo correspondence problem for stereo matching in computer vision application". In: 1 (July 2010). DOI: 10.1109/ICCSIT.2010.5565062.

[3] T. Kanade and M. Okutomi. "A stereo matching algorithm with an adaptive window: theory and experiment". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 16.9 (1994), pp. 920–932. DOI: 10.1109/34.310690.

[4] Li Ma et al. "A Modified Census Transform Based on the Neighborhood Information for Stereo Matching Algorithm". In: *2013 Seventh International Conference on Image and Graphics.* 2013, pp. 533–538. DOI: 10.1109/ICIG.2013.113.

[5] D. Scharstein and R. Szeliski. "High-accuracy stereo depth maps using structured light". In: *2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003. Proceedings.* Vol. 1. 2003, pp. I–I. DOI: 10.1109/CVPR.2003.1211354.

[6] Daniel Scharstein and Richard Szeliski. "A Taxonomy and Evaluation of Dense Two-Frame Stereo Correspondence Algorithms". In: *International Journal of Computer Vision* 47.1 (Apr. 2002), pp. 7–42. ISSN: 1573-1405. DOI: 10.1023/A:1014573219977. URL: https://doi.org/10.1023/A:1014573219977.

[7] Fridtjof Stein. "Efficient Computation of Optical Flow Using the Census Transform". In: *Pattern Recognition.* Ed. by Carl Edward Rasmussen et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 79–86. ISBN: 978-3-540-28649-3.

[8] Minh Vo, Srinivasa G. Narasimhan, and Yaser Sheikh. "Separating Texture and Illumination for Single-Shot Structured Light Reconstruction". In: *2014 IEEE Conference on Computer Vision and Pattern Recognition Workshops.* 2014, pp. 433–440. DOI: `10.1109/CVPRW.2014.70`.

[9] Ramin Zabih and John Woodfill. "Non-parametric local transforms for computing visual correspondence". In: *Computer Vision — ECCV '94.* Ed. by Jan-Olof Eklundh. Berlin, Heidelberg: Springer Berlin Heidelberg, 1994, pp. 151–158. ISBN: 978-3-540-48400-4.