Stereo Vision, 3D Reconstruction and Object Detection

Radhika Deshpande (ES14BTECH11006), K Sai Niranjan (ES14BTECH11011),
Sanjan Prakash (EP14BTECH11007))

***Abstract*—This project aims at creating a system which includes a pair of stereo cameras for 3-dimensional reconstruction, object detection and depth analysis. Further, it uses disparity maps for more accurate object detection.**

I. Introduction

STEREO vision refers to the extraction of 3-dimensional information from digital images. A major motivation for development in the domain of stereo vision is our very own pair of stereo cameras - our eyes. By simply viewing an area before us, we are able to gauge distances and depths and have a sound sense of orientation about our surroundings. Following the model of our eyes, stereo vision usually involves the use of two cameras, with the images captured of a scene using them being stereoscopic complements of each other. Analysis of the relative positions of objects in these two frames (disparity) leads to the extraction of 3-dimensional information about the real-world points, most notably the distance of a point along the optical axis of the stereo cameras. This data can be extrapolated to generate a collection of real-world points, known as point clouds, that will be used to simulate or reconstruct the scene captured.

Stereo vision has been on a rapid rise in this age of automation. It can be found profoundly useful in the production of automated automobiles. The vehicles would detect obstacles, stationary, approaching or receding, and dodge them, all in the absence of a human being. SLAM (Simultaneous Localization and Mapping) is vital for autonomous robot navigation. The robot builds a map of its environment while tracking its motion through that map. 3D reconstruction of the locality of interest for this purpose can be carried out using stereo vision. The booming gaming industry is also another avenue where stereo vision can be seen as an emerging solution for building maps of regions where these games are set in. Further, object detection, which spawns as an offspring of stereo vision, can be used as a tool to aid the visually-challenged in order to simulate a sense of vision for them.

II. Hardware Setup

We used two cameras (Logitech C270) for building our own stereo camera. We have used a flat wooden plank to attach the cameras. We have kept them 6.7 cm which would be close to the average separation between the human left and right eyes. Our initial data was collected using two cameras. The major problem we found with this setup was that the exposure rate of both the cameras was different and hence it resulted in a bad disparity map. We have later turned to using a physics slider with only one camera. This provided the advantage of having the same exposure. However we could only work with images which did not have motion as it would take time to move from left to right position. We have used the slider to take photos from one camera with a baseline difference of $5cm$.

III. Calibration and Rectification

We used $Matlab's$ stereo calibrator app to calibrate the cameras. From the stereo calibrator app, we obtained the intrinsic matrix of both the cameras and also extrinsic parameters (rotation and translation matrices). We tried rectification with two approaches :

1) We used radial and tangential coefficients to undistort the images captured. We used intrinsic and extrinsic parameters to find the respective rotation matrices for each camera. We applied the respective rotation matrices to the images to make their epipolar lines parallel. We also obtained disparity to depth matrix.
2) We tried to rectify the images without any calibration. We have used 8 point algorithm to find the fundamental matrix. We initially applied Scale-invariant Feature Transform (SIFT) on both the left and right images. We matched the feature points using FlannBasedMatcher from OpenCV. Once we extracted the corresponding points in both the images, we calculated the fundamental matrix using the 8-point algorithm. The fundamental matrix relates two points lying on corresponding epipolar lines through the following equation :

$$XFX' = 0,$$

where X is the point from the left image and X' is corresponding point from the right image. We used the fundamental matrix to obtain the final rectified images.

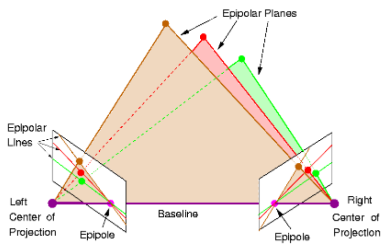

Fig. 1. Diagram displaying the epipolar planes and lines with the orientation of the camera

IV. 3D Reconstruction and Point Clouds

Post calibration and rectification, we used the disparity matrix obtained from calibration to reproject to 3 dimensional space. Initially, we used the dataset from Middlebury to experiment with our code. The accuracy and precision of the 3D projection depends crucially on disparity and calibration matrices. We used the the PLY file format (Polygon File Format) to visualize the 3D point cloud. The resulting point clouds are rendered in 3D by using a point could viewer software MeshLab. The results using the Middlebury dataset are as in Fig1 :

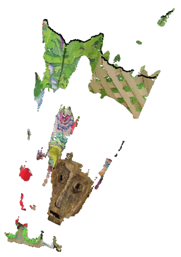

Fig. 2. 3D point cloud of Middlebury dataset visualized in MeshLab

We then went on to visualize the 3D projections of the images obtained from our cameras. Unfortunately, even after careful tuning of disparity parameters we did not get desirable results. The reason behind undesirable results is probably our setup. The system is not as robust as required for projection to 3D.

V. Disparity

The relative shift in the location of an object in corresponding images captured from left and right positions of the stereo camera is referred to as disparity. A matrix of disparity values for each of the pixels of an image with respect to its stereoscopic complement is called a disparity map.

We used Semi-Global Block Matching (SGBM) method to find disparity from rectified images. The cost function that SGBM uses is sum of absolute differences of pixel values in a block. There are other cost functions that could be used too, such as absolute gradient difference, squared intensity difference etc. We tried to use a non-parametric cost function census. Census uses relative difference between the pixels around it's pixels. Census transform gives a string of zeros and ones depending upon the pixels around the main pixel.

$$C(i+k,j+l) = \begin{cases} 1, if img[i+k,j+l] > img[i,j] \\ 0, otherwise \end{cases}$$

All the values are concatenated to form a string. The cost function is the hamming distance between the strings of corresponding pixels. Normally we take minimum hamming distance for the disparity. However there are many other optimization techniques for selecting the optimal disparity such that the cost function is minimized and also it doesn't affect the smoothness of the disparity.

All the values are concatenated to form a string. The cost function is the Hamming distance between the strings of corresponding pixels. Normally we take minimum Hamming distance for the disparity. However, there are several other optimization techniques for selecting the optimal disparity such that the cost function is minimized and also it does not affect the smoothness of the disparity.

SGBM uses scanline optimization to select the optimal disparity. Other optimization techniques include graph cut and adaptive belief propagation.

VI. Object Detection

The biggest advantage of Stereo Vision is the depth information we get. We wanted to use this information to obtain more accurate object detection. For object detection, we adopted the Graph Cut Method. This method has two modes - cv2.GC_INIT_WITH_RECT or cv2.GC_INIT_WITH_MASK or both simultaneously.

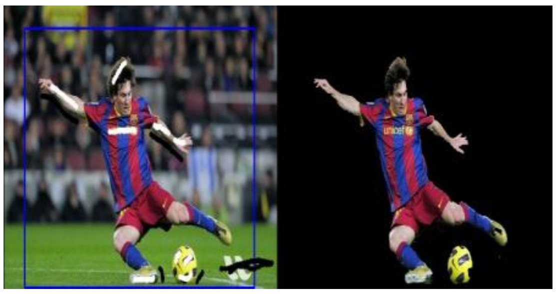

Fig. 3. Grabcut uses 2 modes cv2.GC_INIT_WITH_RECT or cv2.GC_INIT_WITH_MASK or both simultaneously

A. Algorithm

In this method, the user initially inputs the foreground and background through either of the modes. For the rectangle mode, everything outside this rectangle will be taken as sure background. Everything inside rectangle is unknown. Similarly any user input specifying foreground and background through mask mode are considered as hard-labelling which means they won't change in the process. Computer does an initial labelling depending on the data we gave. It hard-labels the foreground and background pixels. A Gaussian Mixture Model (GMM) is then used to model the foreground and background.
Depending on the data we gave, GMM learns and create new pixel distribution. That is, the unknown pixels are labelled either probable foreground or probable background depending on its relation with the other hard-labelled pixels in terms of color statistic.
A graph is built from this pixel distribution. Nodes in the graphs are pixels. Additional two nodes are added, Source node and Sink node. Every foreground pixel is connected to Source node and every background pixel is connected to Sink node.
The weights of edges connecting pixels to source node/end

node are defined by the probability of a pixel being foreground/background. The weights between the pixels are defined by the edge information or pixel similarity. If there is a large difference in pixel color, the edge between them will get a low weight.
Then a mincut algorithm is used to segment the graph. It cuts the graph into two separating source node and sink node with minimum cost function. The cost function is the sum of all weights of the edges that are cut.
After the cut, all the pixels connected to Source node become foreground and those connected to Sink node become background.
The process is continued until the classification converges.

We first used the rectangle mode for object detection. The results were not satisfying. So,we thought of using disparity to mark the sure background and foreground in the mask mode. The background has lower disparity while the foreground has higher disparity. The disparity map can further be thresholded in a manner such that the different objects of interest can be isolated from the background.

The images clearly show how masking with disparity gives a more accurate object isolation. Using the depth map, the average distance of the object from the cameras is also calculated.

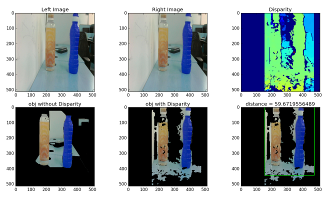

Fig. 4. Result 1

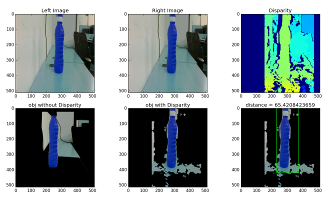

Fig. 5. Result 2

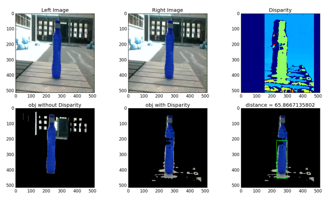

Fig. 6. Result of Live Demo conducted during presentation

The disparity thresholds have to be hard coded in the current version of our code. Our future work includes creating a GUI for a more robust and accurate system.

VII. Depth Estimation

Disparity simply works on the principle of parallax of the eye. According to this, objects that are closer to the eye or the camera appear to move at a faster rate when compared to objects that lie in the background. This implies that, objects in the foreground seem to be shifted by a greater magnitude than those in the foreground. This demarcation in disparities can be exploited to evaluate the distance of objects along the optical axis of the camera in the real-world.

As has been made clear thus far, distance along the optical axis is inversely proportional to the corresponding disparity value obtained from the disparity map. This can be supported with mathematical proof, using the principle of similar triangles.

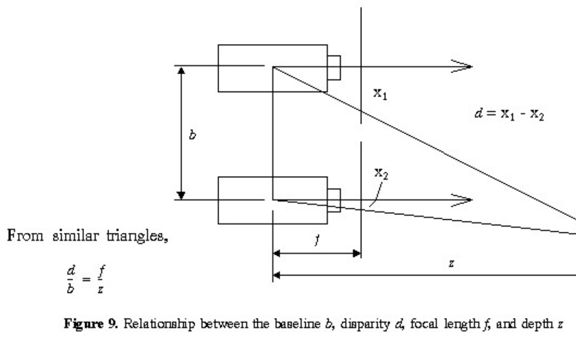
x_1
$d = x_1 - x_2$
b
x_2
P
From similar triangles,
$\frac{d}{b} = \frac{f}{z}$
f
z
Figure 9. Relationship between the baseline b, disparity d, focal length f, and depth z

Fig. 7. Principle of similar triangles used for depth estimation

VIII. Conclusion

We successfully obtained disparity from the images we have taken. We also improved the graph cut algorithm by using the depth knowledge we acquired from two images. We had achieved much better results using disparity at separating foreground from background. We have been able to reconstruct to 3D point clouds from disparity map.

IX. Future work

We believe that there is a lot of scope for stereo vision in self driving cars. We have just used stereo vision for detection objects in stationary images. Finding out various objects at different depths in a video would be very crucial for self driving cars. There is also a lot of work involved in stabilizing a stereo video. One more future goal we plan to achieve is stitching of point clouds.

X. Work Distribution

Niranjan built the initial setup of cameras. He did calibration for the cameras and applied rectification for the images. He tried rectification through both the approaches using calibrated way and also uncalibrated way. He obtained the disparity for the rectified images. He wrote code to tweak the parameters of the disparity. He also tried to find disparity by applying census transform. When there was difficulty in getting disparity due to change in exposure of cameras, Sanjan suggested to use one camera and slider to take images so that we can get better disparity.

Sanjan worked on obtaining disparity maps and using them to evaluate the real-world depth of pixels in the image. Initially, we went with a Global Matching algorithm to obtain the disparity map. On comparison, it was found that SGBM was delivering better results and thus, we decided to use this for disparity map generation.

Radhika worked on visualization of 3D point clouds for datasets from Middlebury and our own images. After a lot of reading, Niranjan suggested the Graph Cut method for object detection. Radhika experimented using datasets and came up with the algorithm/code for more accurate object detection using disparity. She also organized the scripts for a more structured and robust system.

References

[1] A Global 3D Map-Building Approach Using Stereo Vision http://www.dccia.ua.es/~jmsaez/3D_Robot_Vision_Database/Papers/Global\%203D\%20Mapping\%20ICRA\%202004.pdf

[2] MASTER'S THESIS High Accuracy Volume Measurement of Small Shapes using Stereo Vision http://ltu.diva-portal.org/smash/get/diva2:1029709/FULLTEXT02.pdf.

[3] Interactive Foreground Extraction using GrabCut Algorithm http://docs.opencv.org/3.1.0/d8/d83/tutorial_py_grabcut.html.