CSE 668
Monocular Depth Estimation for Visual SLAM

Nitin Nataraj (50246850)
Siddharth Sharma (50247321)

May 22, 2018

1 Introduction

Depth estimation refers to the set of techniques and algorithms aiming to obtain a representation of the spatial structure of a scene. In other terms, it is used to obtain a measure of the distance of each point in the visible scene. THe features obtained from both the RGB images and the depth images can be used as the premise for RGBD SLAM. This type of SLAM makes use of these visual features to estimate odometry information from consecutive frames.

Monocular camera settings are cheaper than stereo camera settings and are quite quick to setup. All you need is to strap on a camera on the robot, and you should be able to map the environment. Moreover, LIDAR is more expensive to use in any case, and this technique can serve as an inexpensive replacement for LIDAR. In this project, we worked on a system that could estimate depth information from monocular images using deep learning techniques. We then explore this system's usefulness in mapping the environment by using a type of RGBD SLAM called RTABMap SLAM.

2 Related Work

A lot of work has been on monocular depth estimation. The first breakthrough was from Stanford's Ashutosh Saxena et. al.[2] when they introduced Make3D, an algorithm that could estimate detailed 3D structure from a single still image of an unstructured environment. They use a Markov Random Field (MRF) to infer a set of "plane parameters" that capture both the 3D location and the orientation of the patch. They then use supervised learning to model both the image depth cues as well as the relationships between different parts of the image.

Mayer et al. [3] introduced a fully convolutional deep network called DispNet that directly computes the correspondence field between two images. At training time, they attempt to directly predict the disparity for each pixel by minimizing a regression training loss.

The above methods rely on having large amounts of accurate ground truth

disparity data and stereo image pairs at training time. As an alternative, the authors in [1] pose depth estimation as an image reconstruction problem during training. The intuition here is that, given a calibrated pair of binocular cameras, to learn a function that is able to reconstruct one image from the other, having learned something about the 3D shape of the scene that is being imaged. At training time, they have access to two images Il and Ir , corresponding to the left and right color images from a calibrated stereo pair, captured at the same moment in time. Instead of trying to directly predict the depth, they attempt to find the dense correspondence field dr that, when applied to the left image, would enable the right image to be reconstructed. Given the baseline distance b between the cameras and the camera focal length f, they recover the depth $d*$ from the predicted disparity by using the formula $d* = bf/d$.

3 Approach

Our experiments are based on the work in [1] as we explore monocular depth estimation with left-right consistency in conjunction with RTABMap SLAM.

We attempt the following tasks:

- Run RTABMap [4] on a readily available RGBD dataset, namely the KITTI dataset.
- Apply a pre-trained model, trained on the CityScapes and the KITTI datasets, to RGB images and obtain depth images. We then plan to use these depth images with RTABMap and compare the differences with the ground truth version.
- Implement a ROS publisher to publish three different topics as accepted by RTABMap. The three topics are for RGB images, depth images and camera information respectively.
- Obtain the correct camera calibration parameters for the respective datasets.
- Experiment with different thresholding and scaling methods for conversion between disparity to depth.

4 Challenges Faced

While working on the project, we encountered several challenges, a few of which we have delineated below:

- Visual odometry information was lost during image rescaling.
- Determining camera calibration parameters proved to be very tricky.
- Testing the model was computationally inefficient on the CPU, and there were several GPU images on Ubuntu.
- RTABMap subscriber would not accept data from the published topics. We posit that this could be due to sync issues with published topics.

5 Results

We ran several experiments are the results are displayed below.

5.1 Original KITTI Image

Figure 1: Original grayscale KITTI Image

5.2 RTABMap with KITTI Ground Truth Depth Images

Figure 2: These are the ground-truth depth images obtained from the KITTI dataset.

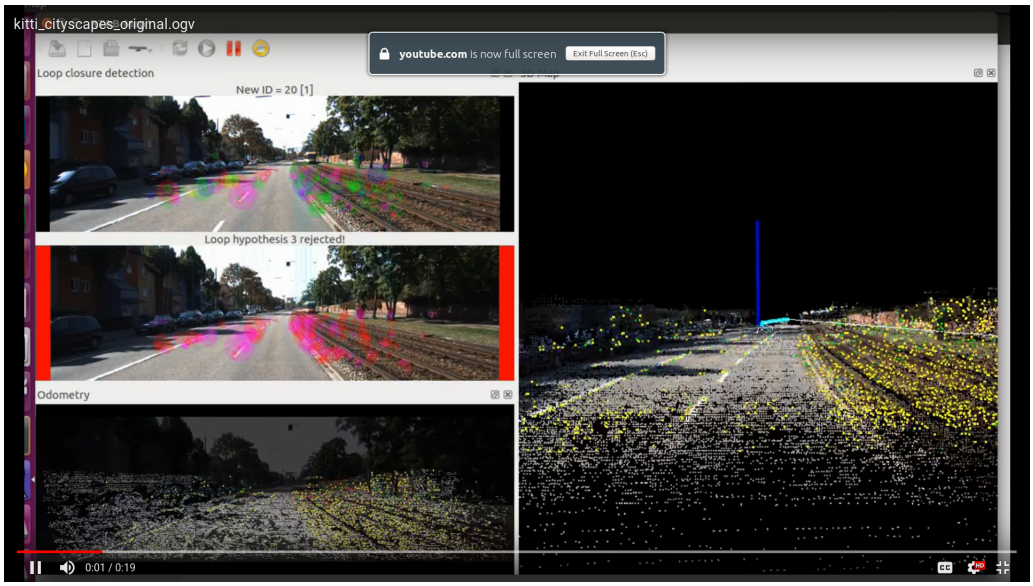

Figure 3: RTABMap mapping is shown on the right with ground truth depth images on the KITTI dataset.

5.3 RTABMap with Depth Images after the Application of Scaling Formula

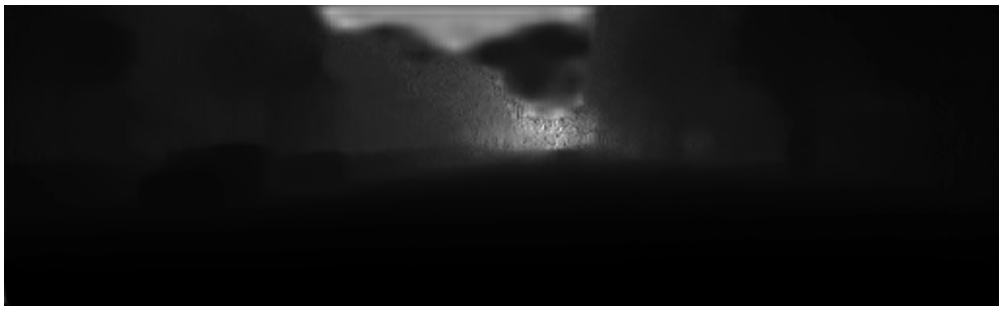

Figure 4: We apply the formula provided for the KITTI dataset. $depth = 0.54 * 721/(1242 * disparity)$

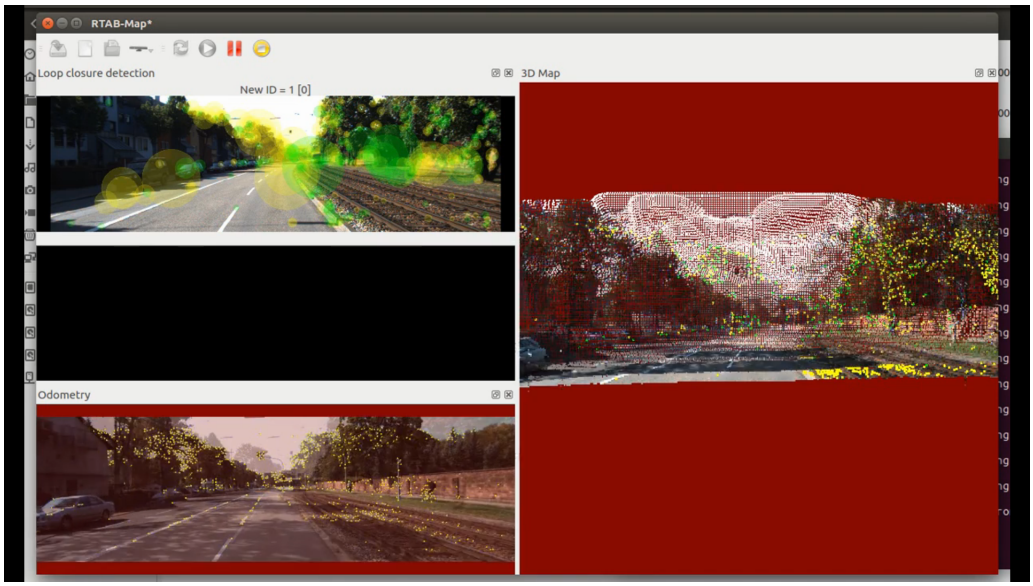

Figure 5: RTABMap Mapping using depth images obtained from first applying the model on the RGB images, and then scaled using the formula $depth = 0.54 * 721/(1242 * disparity)$

5.4 RTABMap with Depth Images from Average Stacked Scaling

Figure 6: We define average stacked scaling as the process of comparing the ground truth depth images with the images obtained from the application of the model; obtaining an average scale map, and applying this as a multiplier to the depth images obtained from the model. We first invert the disparity image to make it proportional to depth.

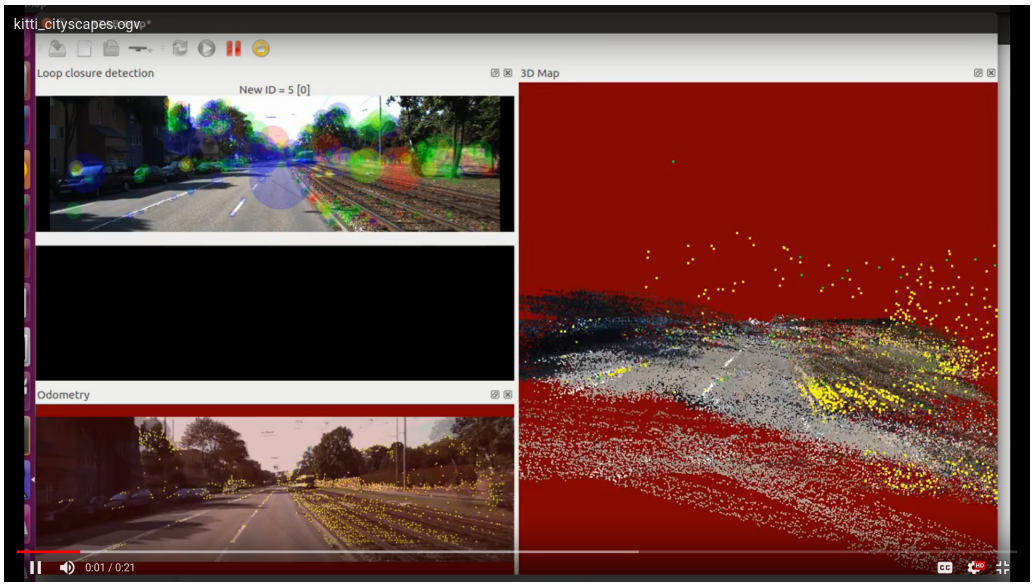

Figure 7: RTABMap Mapping obtained on stacked,scaled depth images obtained from the model.

6 Future Work

A few future directions would be:

- Obtaining real data.
- Syncing topics from ROS publisher.
- Real-time with GPU integration.
- Scaling estimation using Kinect.

7 Running the Code

7.1 Software Downloads and Setup

- Download and setup RTABMap software by following the instructions on this website - https://github.com/introlab/rtabmap/wiki/Installation
- Download and setup tensorflow 1.0 with Python 2.7.
- Clone the monodepth repository from https://github.com/mrharicot/monodepth.

7.2 File Descriptions

These are the desctiptions of the various Python scripts we used to test the different scaling methods.

7.2.1 monodepth_test_multiple_formula.py

This script contains code that applies the direct formula available for the KITTI dataset.

7.2.2 monodepth_test_multiple_temporal.py

This script contains code to run the temporal scaling we tried. It takes the image from the current, previous and future timesteps and determines variances between the pixels at each location. Locations with high variance have their pixels zeroed out.

7.2.3 monodepth_stacked.py

This script contains code for the averaged stacked scaling approach described earlier.

7.2.4 sample_talker.py

This script contains code to publish three ROS topics, RGB images, depth images and camera info. THe respective launch file is within the launch directory as sample_talker.launch

7.3 Running Commands

Run the following commands to execute the depth estimation script and populate depth images into a particular folder. The rgb_kitti folder contains all the RGB KITTI images to be converted into depth images. The depth_kitti folder contains depth images.

python monodepth_test_multiple_formula.py –checkpoint_path models/model_kitti –input_dir rgb_kitti –output_dir depth_kitti

7.4 Running RTABMap

- Assuming you have RTABMap downloaded and setup, you can go the command prompt and type "rtabmap".
- Once there, go to **Window**, select the **Source** tab, select RGB-D as source type, set input rate to 30 Hz or lower, set calibration file to "rgb-ddatasets2.yaml".
- Scroll down to select **Images** for the camera driver. Fill RGB and depth directories (with the respective RGB and Depth image folders).
- Click apply and close the window.
- Click on the "Create new database" icon in the toolbar, click "ok" for the default values and click the "Play" button.

References

[1] Godard, Clément, Oisin Mac Aodha, and Gabriel J. Brostow. "Unsupervised monocular depth estimation with left-right consistency." CVPR. Vol. 2. No. 6. 2017.

[2] Saxena, Ashutosh, Min Sun, and Andrew Y. Ng. "Make3d: Learning 3d scene structure from a single still image." IEEE transactions on pattern analysis and machine intelligence 31.5 (2009): 824-840.

[3] N. Mayer, E. Ilg, P. Hausser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In CVPR, 2016.

[4] Geiger, Andreas, et al. "Vision meets robotics: The KITTI dataset." The International Journal of Robotics Research 32.11 (2013): 1231-1237.