# Individual Lab Report

Bikramjot Singh Hanzra
Team B – Auto Pirates
Teammates – Tushar Chugh, Shiyu Dong, Tae-Hyung Kim, William Seto
ILR04

November 13, 2015

# Contents

# List of Figures

# 1 Individual Progress

In the last week, I worked upon writing a ROS package `pirate_filter` to filter radar data and extracting location information of the obstacles using the OpenCV [1] library. The package is hosted on GitHub at `http://github.com/auto-pirates/recboat-ros/tree/master/src/pirate_filter` and is available for download.

The data that comes from the RADAR needs to be filtered before it can be fed to the path planning algorithm. The problem is that a lot of times a single obstacles breaks into multiple smaller obstacles as shown in Figure 1. If this issue is not addressed, the path planning algorithm will try to plan the path through the regions in between these multiple smaller obstacles which will ultimately lead to collisions.

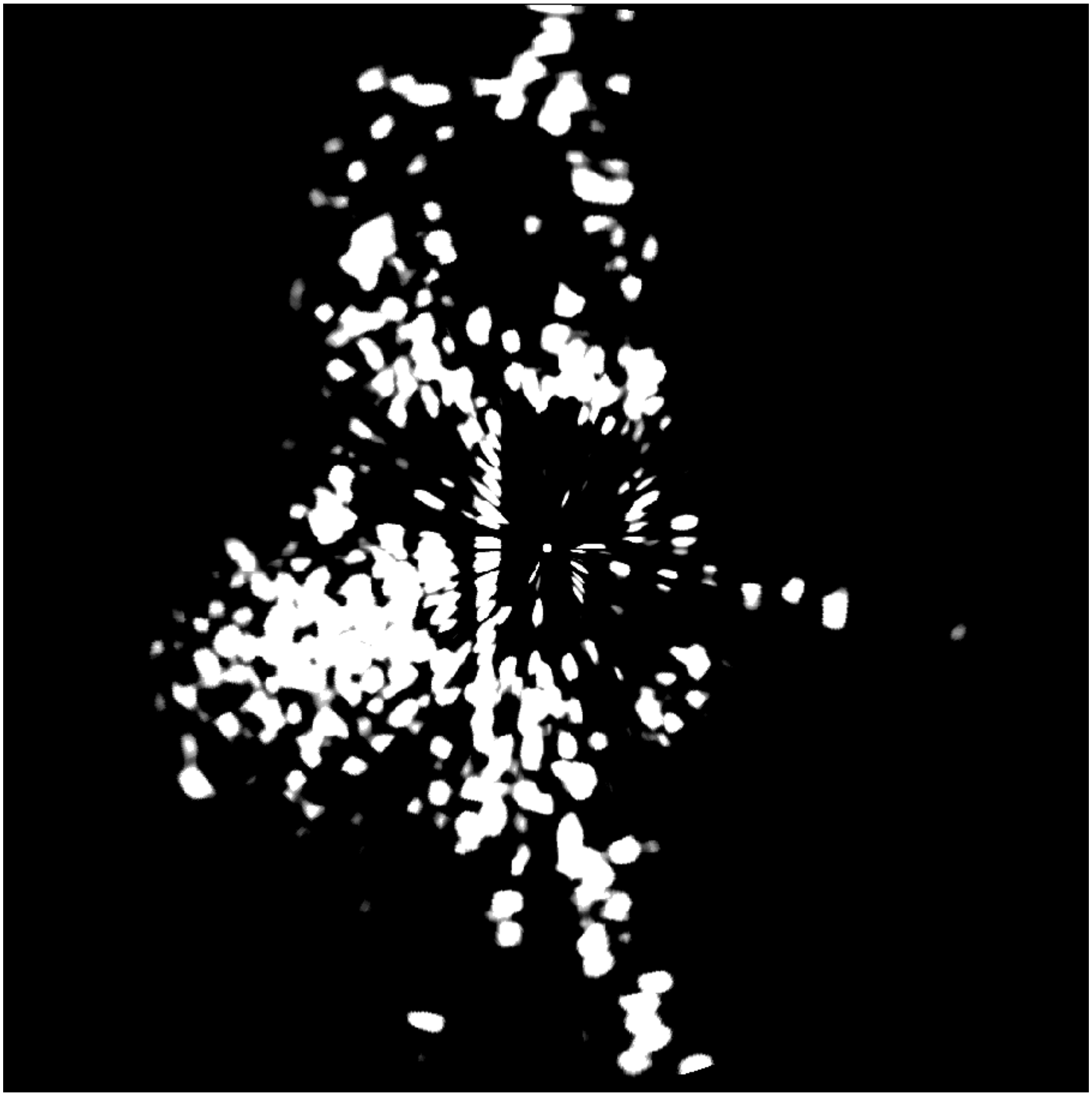


Figure 1: Raw data from the RADAR

In order to ameliorate this issue, I applied morphological operations on the image to connect the small obstacles belonging to the same obstacle. I tried a combination of dilation, erosion, opening and closing operations using rectangular kernels of sizes 3x3, 5x5, 7x7 and so on. After applying the morphological operations, I was able to combine the separated obstacles as shown in Figure 2.

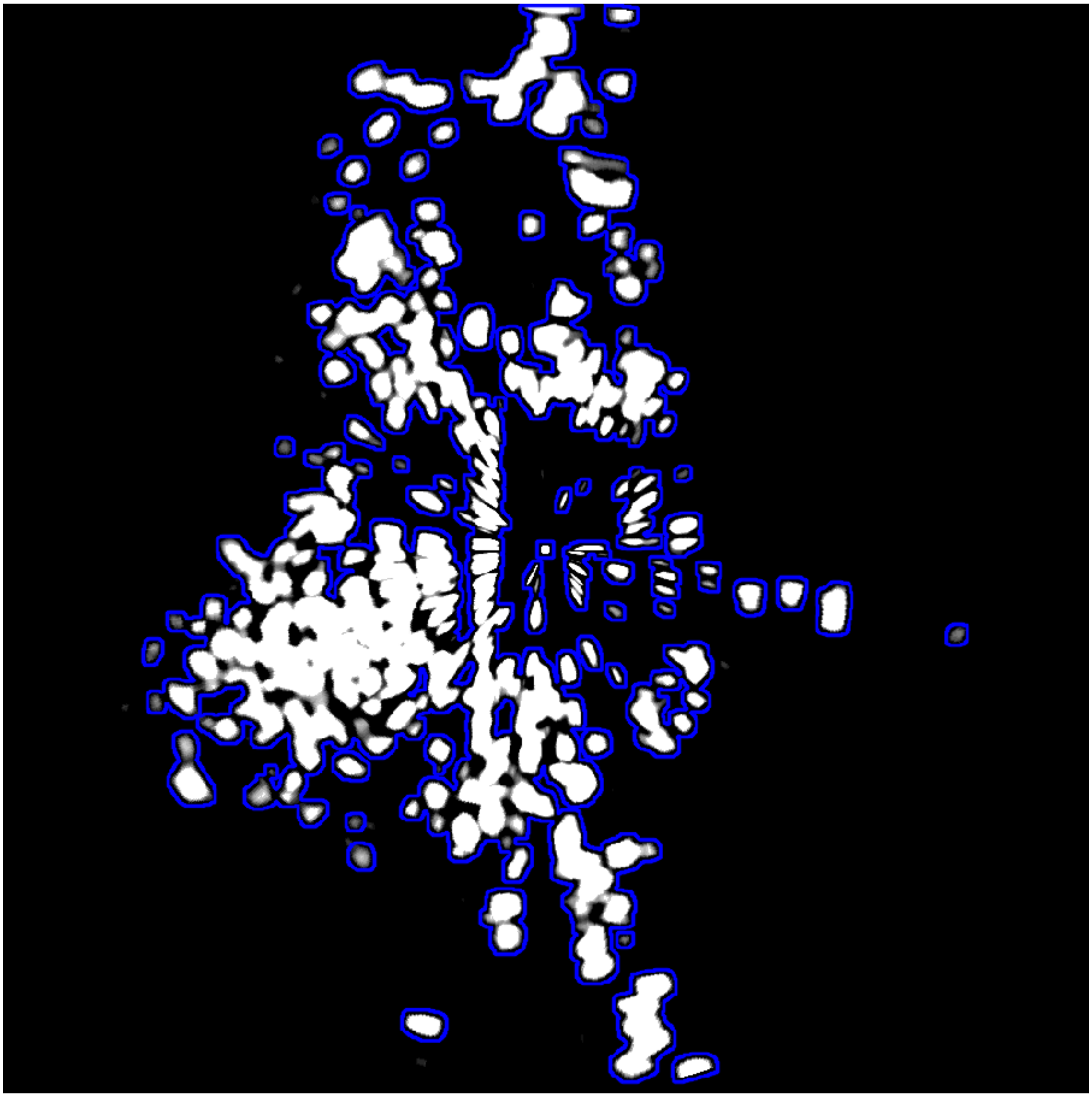


Figure 2: Data from the RADAR after processing

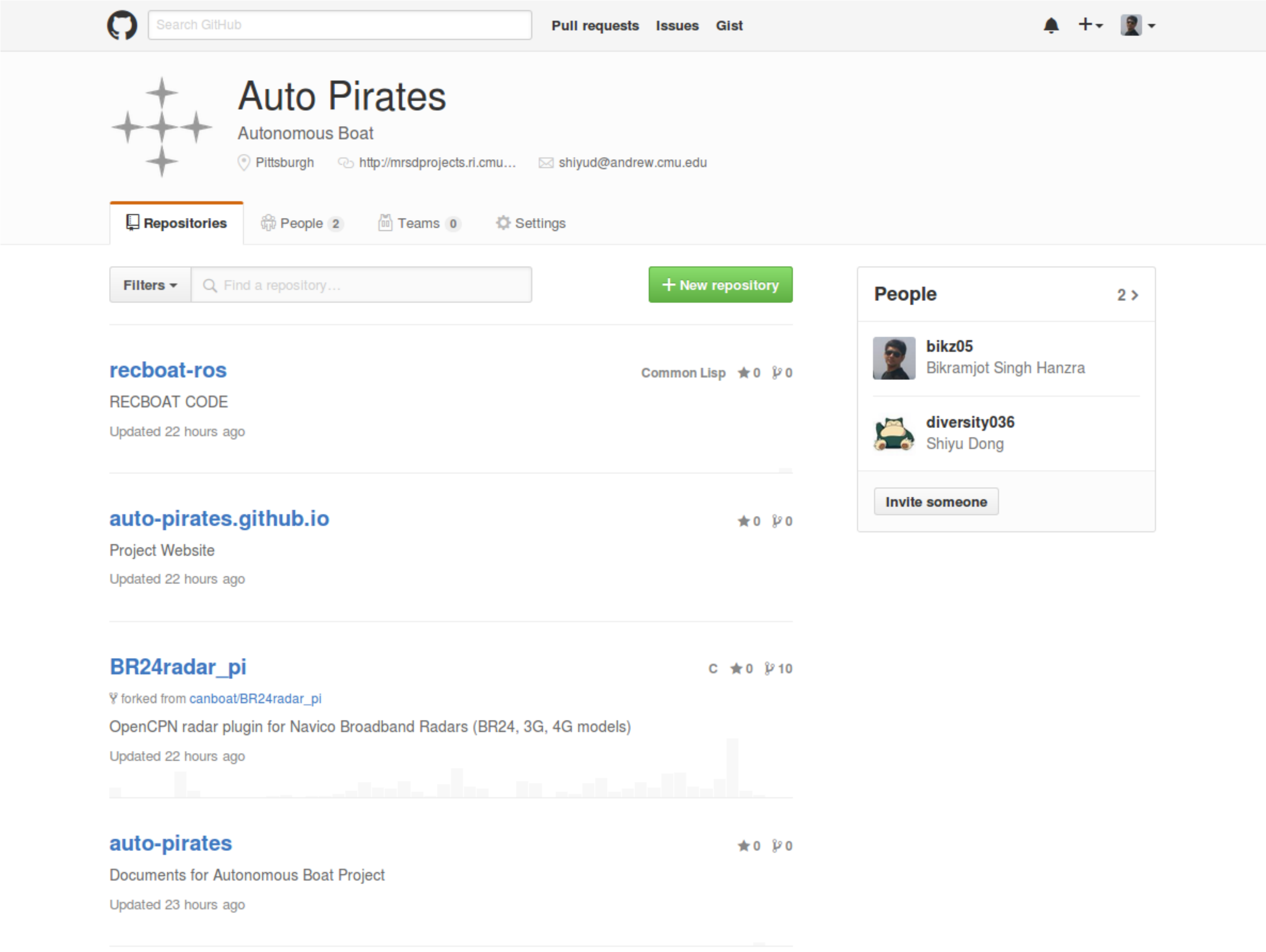
The blue outline shows the contour detected for each obstacles. After this step of contour extraction, I was able to get the coordinates of the edges of each obstacles. The vector below shows how each contour is stored in the memory –

```
 New Object Detected --> [497, 72] [498, 71] [499, 71] [500, 72] [500, 74] [499, 75]
[499, 76] [497, 78] [496, 78] [495, 79] [495, 80] [493, 82] [493, 83] [492, 84] [491,
84] [491, 91] [488, 94] [488, 95] [487, 96] [486, 95] [486, 85] [485, 85] [484, 84]
```

[484, 82] [483, 82] [482, 81] [481, 81] [480, 80] [480, 79] [483, 76] [485, 76] [486, 75] [489, 75] [490, 74] [493, 74] [495, 72]

After getting the shape of each object, I calculated the centroid and area of each obstacles.

We have written a lot of source code till now and it was important we host the code at one single location. So, I created a GitHub organization account (`https://github.com/auto-pirates`) and made four repositories.



Auto Pirates

Autonomous Boat

Pittsburgh http://mrsdprojects.ri.cmu… shiyud@andrew.cmu.edu

**recboat-ros** Common Lisp ★0 0

RECBOAT CODE

Updated 22 hours ago

**auto-pirates.github.io** ★0 0

Project Website

Updated 22 hours ago

**BR24radar_pi** C ★0 10

forked from canboat/BR24radar_pi

OpenCPN radar plugin for Navico Broadband Radars (BR24, 3G, 4G models)

Updated 22 hours ago

**auto-pirates** ★0 0

Documents for Autonomous Boat Project

Updated 23 hours ago

People 2 >

bikz05 Bikramjot Singh Hanzra

diversity036 Shiyu Dong

Figure 3: GitHub Organizational Account Homepage

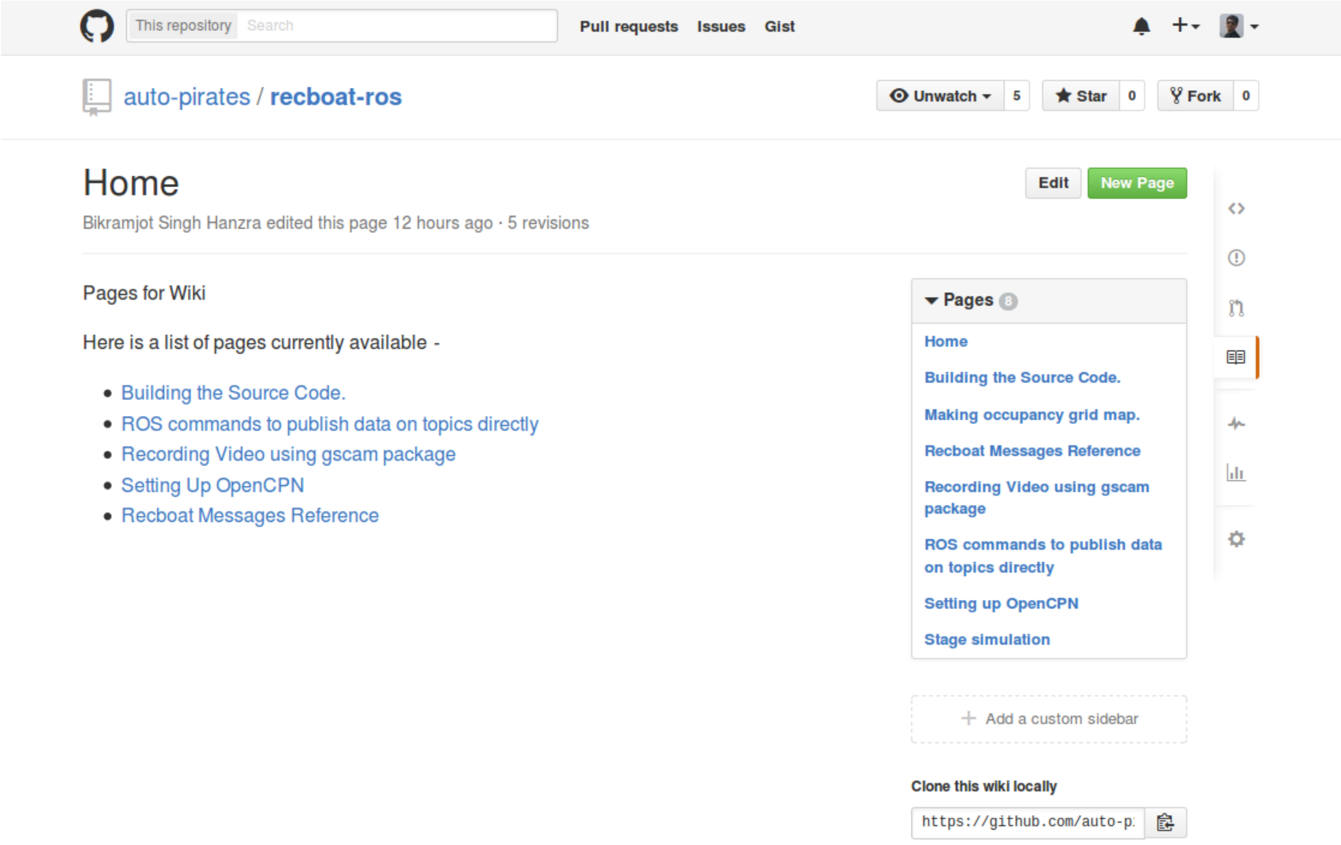
The four repositories as shown in Figure 3 are –

- `recboat-ros` – ROS WORKSPACE that contains all the packages written by the team members
  Source Code at – `https://github.com/auto-pirates/recboat-ros.git`
- `auto-pirates.github.io` – Initialized the GitHub Pages repository. Currently the reposi-

tory is empty.
Source Code at – `https://github.com/auto-pirates/auto-pirates.github.io.git`

- `BR24radar_pi` – Modified forked repository of OpenCPN plugin `BR24radar_pi`.
  Source Code at – `https://github.com/auto-pirates/BR24radar_pi.git`
- `auto-pirates` – Repository for Projects Documents like slides, ILRs etc.
  Source Code at – `https://github.com/auto-pirates/auto-pirates.git`

I also wrote Wiki pages which contains instructions about how to use the packages. Setup instructions are also included in the Wiki. Figure 4 shows the screenshot of the Wiki page.



This repository
Search
Pull requests
Issues
Gist
auto-pirates / recboat-ros
Unwatch 5
Star 0
Fork 0
Edit
New Page
Pages 8
Home
Building the Source Code.
Making occupancy grid map.
Recboat Messages Reference
Recording Video using gscam package
ROS commands to publish data on topics directly
Setting up OpenCPN
Stage simulation
+ Add a custom sidebar
Clone this wiki locally
https://github.com/auto-p

Home

Bikramjot Singh Hanzra edited this page 12 hours ago · 5 revisions

Pages for Wiki

Here is a list of pages currently available -

- Building the Source Code.
- ROS commands to publish data on topics directly
- Recording Video using gscam package
- Setting Up OpenCPN
- Recboat Messages Reference

Figure 4: Screenshot of Project Wiki

## 2 Challenges

The biggest setback that we faced this week was that our field trial today was cancelled due to the aberrant weather conditions. We have now shifted the field trial to Tuesday because tomorrow is Saturday and unfortunately, the driver is not available on weekends. The weather on Monday is also not expected to be favorable. This has certainly pushed our schedule by four days.

While writing the `pirate_filter`, I did not have much difficulty because I am pretty comfortable with the `OpenCV` library. Although, I did not know how to run OpenCV code in a ROS node but I was able to learn this during the ROS Assignment.

We faced some issues while compiling all the packages in the `recboat` workspace. Although, I was able to compile the packages ultimately but I still don't feel comfortable in writing `CMakeLists.txt` files. I need to put some effort into learning how to write a `CMakeLists.txt` file.

## 3 Teamwork

The work done by the rest of team is as under –

- Tushar Chugh – Tushar worked on generating the grid occupancy map and worked on the PCB assignment with Shiyu.
- Shiyu Dong – Shiyu worked on the PCB assignment with Tushar and wrote the launch file for recording raw camera frames.
- Tae-Hyung Kim – Kim worked on generating the map of the river in the `Stage` simulator. He also worked on simulating a TurtleBot in `Stage` to test the path planning algorithm he is working upon.
- William Seto – William worked on visualization of the radar data using `matplotlib` library in Python.

## 4 Work Overview for Coming Week

In the next week, I will be concentrating on –

- Logging useful RADAR data collected during the field test.
- Continue working on code for removing noise from RADAR data.
- Currently, we are sending commands to the boat using `rostopic pub` which is both cumbersome and time consuming. To make this easier, I am planning to write a few launch files.

## References

[1] `OpenCV` Official Website
`http://opencv.org/`