



PROGRESS REVIEW #3


INDIVIDUAL LAB REPORT [ILR04]

NOVEMBER 13, 2015

TUSHAR CHUGH

TEAM B-AUTOPIRATES

SHIYU DONG, BIKRAMJOT HANZRA, TAE-HYUNG KIM, WILLIAM SETO



PROGRESS REVIEW #3

INDIVIDUAL PROGRESS

This week I worked on creating occupancy grid map and designing the PCB.

Path Planning

a. Occupancy Grid Map

I created an Occupancy Grid Map which is compatible with the SBPL library. To create the map I did following steps:

- I created the polygon representing the region where would be having the test runs of our boat (SVE). This was done on 'my maps' of google which allows us to add layers on the top of the map. Figure 1, shows the layer of polygon with gray shade on the top of the map. The length of this polygon (rectangle in this case) is 2.3 mi or 3.701km. The breadth of the rectangle is 0.517 mi or 0.832km. The actual distance which the boat can travel in the river (by keeping its position in the center of the river) is around 3 mi (which is more than what we have in our requirements for the test run). I also recorded the GPS coordinates of end-points of the rectangle.

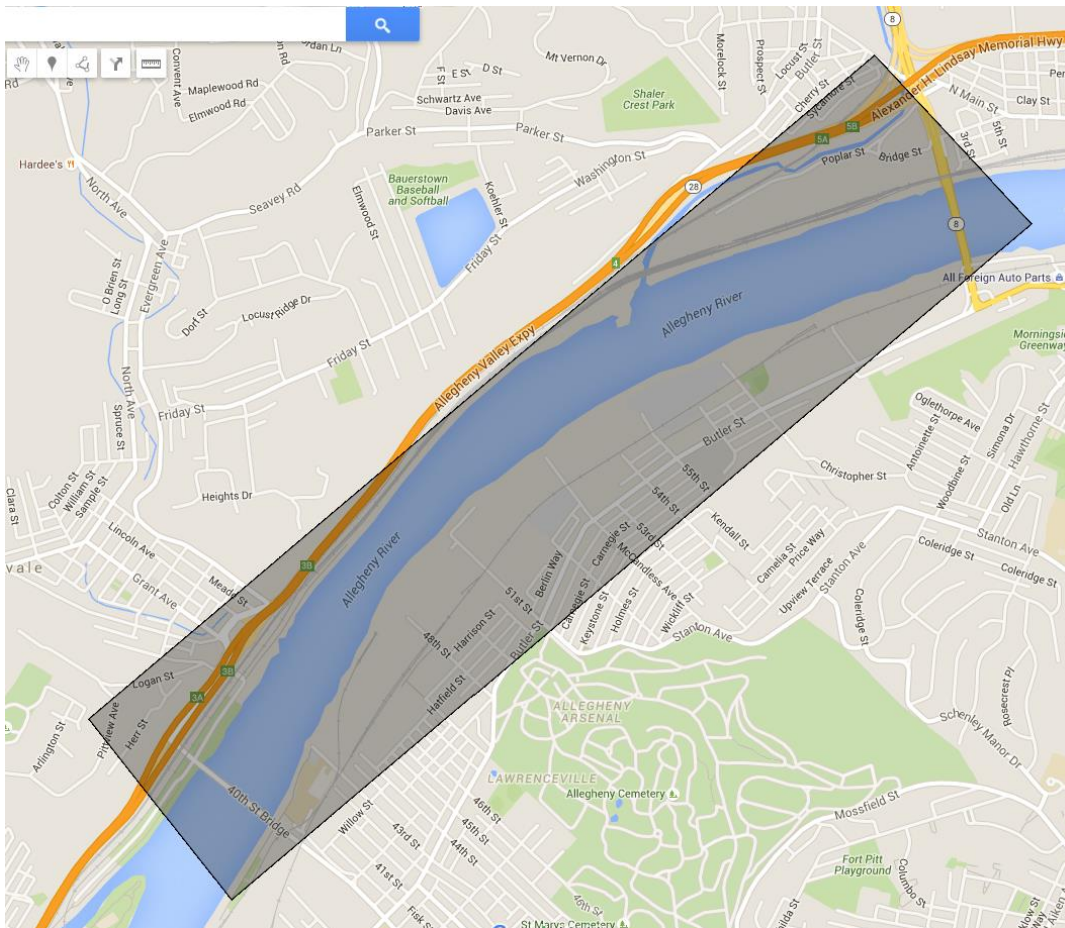


Figure1: Layer of polygon added on the Google Map

- I downloaded the above image from google maps and converted into a black and white image and saved that as a bmp file. Figure 2 shows the converted image. It is to be noted that in a bmp file '1' denotes white and '0' denoted black 'absence of the color'.



Figure2: Image of the river converted into a binary format (bmp)

- I converted this bmp file into occupancy grid format of resolution 4 meters by iterating through the pixels and assigning the corresponding values to the grid. After processing, number of rows times four ($\text{rows} \times 4$) is the width of the image (in meters) and number of columns times four ($\text{column} \times 4$) is the length of the image (in meters). This was done in MATLAB and the result is shown in Figure 3. Here 1 represents obstacle (white in image) and 0 obstacle clear path (black in image).



Figure3: Visualization of the occupancy grip map

- Finally, I converted the matrix into .cfg file which can be given as an input to the executable of the SBPL library. In addition to the matrix, the cfg files contains dimensionsdiscretization (cells), Obsthresh, cost inscribed thresh, Cost possibly circumscribed thresh, Cellsize (meters), nominalvel (mpersecs), start(meters,rads), end(meters,rads). I tested the file by passing it to the executable with random start and end location and the planner was able to find the path.

PCB Designing

I worked with Shiyu to design the PCB for our device 'AlertPirates'. The device will function to alert us whenever the obstacle is reported to be near from the perception code. The device will indicate when obstacles are near to the boat using red LED and a sound alarm. If there are no obstacles in the range of boat then it will show the status with green LED. This circuit is built using ATMEGA 328 microcontroller with Arduino bootloader which will receive serial commands from ROS (ROS topic will publish obstacle presence for Arduino).

I created the Schematic for the circuit, selected the right parts for the PCB and ordered the components. Figure 4 shows the final PCB layout which was submitted for printing.

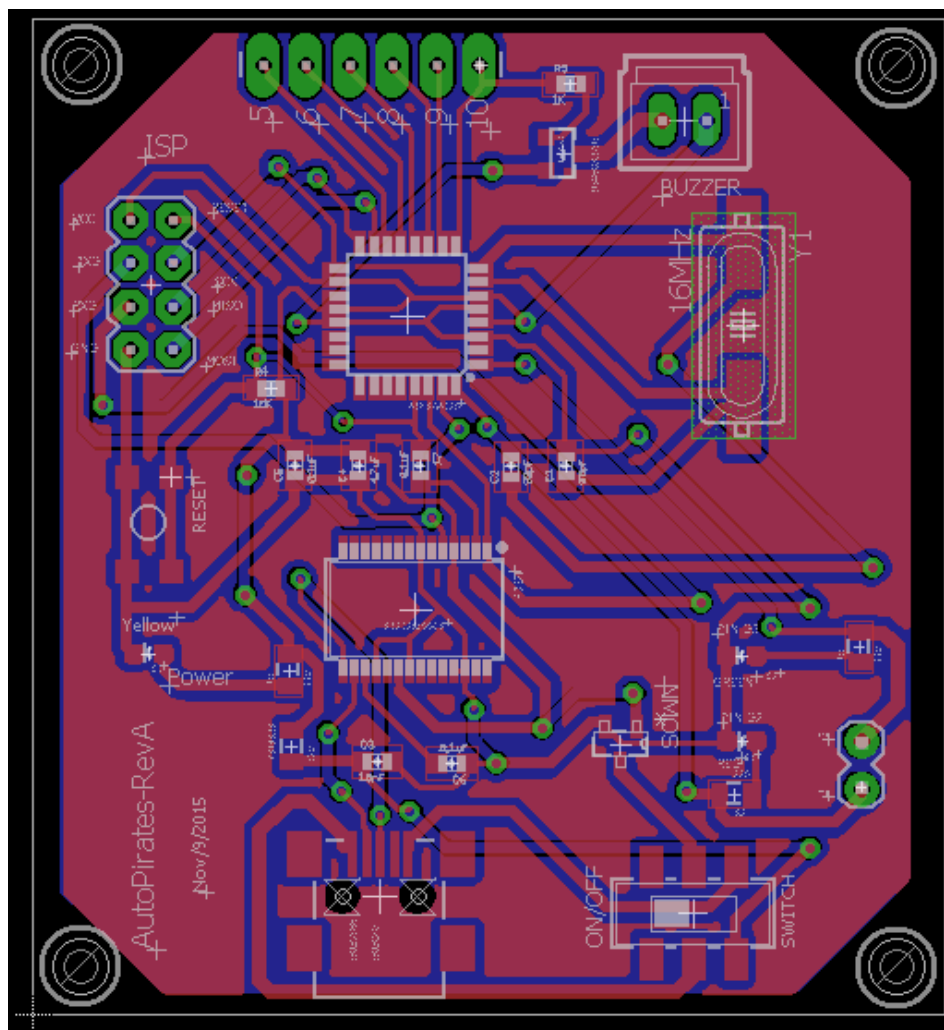


Figure4: PCB layout of AlertPirate device

CHALLENGES

a. Laying the rectangle layer on Google Maps

Google maps only has the option of creating the line but not the rectangle. So, getting the perfect rectangle is difficult. I had to try several times to get the desired result.

b. Resolution of the Image

Sometimes, when I take the screenshot/download the image from the google map, the image is of low resolution (less number of pixels) which makes harder/infeasible to create an Occupancy Grid Map of the desired resolution.

TEAM WORK

a. **Shiyu Dong:** Shiyu worked with me to create the PCB Layout and also interfaced camera on the boat such that we can record the video while the boat is running. This video is our ground truth for testing.

b. **Bikram Hanzra:** Bikram worked on filtering the radar data by creating the blobs around the obstacles. He also contributed on the perception code and create wiki page of the project.

c. **Tae-Hyung Kim:** Tae-Hyung worked on ROS STAGE simulator in which he explored how to simulate path planning using turtle boat. He also imported the Occupancy Grid Map (same one which I had generated) on STAGE.

d. **William Seto:** William was instrumental in driving the progress of the perception front. He worked on collecting the raw data from radar and also he wrote code to visualizing the data in the proper format. William also played the role of Field Manager and created the test plan for our first field test.

FUTURE WORK

a. Field Test

- Write ROS publisher to send command (waypoint, status, etc) to the low-level controller
- Go for the field test of Monday and collect the required data
- Document the results from Field Test

b. Path Planning

- Show visualization of the planned path on the Occupancy grid map by running the planner from the SBPL library

c. PCB

- We have received all the components of the PCB. This week I would be soldering the components on the PCB, will burn Arduino bootloader and will make the PCB functional.
- If time permits, I will write the interface to communicate this device with ROS through serial protocol.