Progress Review 12

Individual lab report – 11 | April 11, 2016

TEAM DAEDALUS

Submitted By:

Mohak Bhardwaj

Team Members:
Dorothy Kirlew
Richa Varma
Shivam Gautam

Pranav Maheshwari

1. Individual Progress

The following responsibilities were taken up by me:

- Fully functioning local planner
- Integration of local planner with global planner and rendering engine.
- Full SVE demos with multiple platforms inside parking lot

FULLY FUNCTIONING LOCAL PLANNER

The local planner is responsible for returning an optimal path given a query of a start pos and end pose of the robot. Before the last progress review, the planning algorithm of the local planner was implemented in C++ and ROS services had been setup by me in order to communicate with the global planner and rendering engine. The local planner was integrated with the rendering engine and paths could be visualized on a simple map in RVIZ with no obstacles and random start and end poses.

The major task left for the completion of the local planner was to make it work with the map of our simulation parking lot. This required correct parsing of the environment costmap from the environment file and integrating collision check along the nonholonomic path of the cars. The state-space Astar planner implemented was initially failing as collision check was not working properly especially in the reverse direction. One of the reasons for this was the mismatch in the location of origin used by me and the rendering engine. After this was corrected, the paths obtained were still erroneous and crossing obstacles. After extensive debugging, it was found that there was a fundamental difference in the way the modulo or remainder operator is implemented in C++ and python. Due to this, during testing the planner would often fail to find a feasible path to the goal location even though its motion primitives allowed it to.

Another major aspect of the planner that had to be addressed was the motion primitives to be used. The planner was taking an extraordinarily large amount of time to return feasible solutions. One of the reasons for this was the resolution and heuristics being used. The resolution for discretization or raster size being used was 0.025m per cell to be consistent with global planner. This lead to a costmap of 1920x1700 dimensions. In the state-space planner, any two poses lying in the same voxel are considered to be the same in the discrete environment. Also, it does not make sense to allow for motion primitives that allow for motions much larger than the voxel dimensions since that would involve more number of collision checks and increase processing time. This lead to all of the successors to be added to the closed list and the planner would exhaust all options. Therefore, the resolution of the raster was increased to 0.1m. Additionally, the costmap was edited by softening the obstacles so that the effect of the obstacle can be felt at a greater distance. One of the reasons for this was that the planner would get stuck in local minima due to the heuristic and concave obstacles (parking spots). Figure 1 shows the map of the final simulation used to derive the costmap for the local planner. The original map image was processed in MATLAB by dilating the obstacles and then using a Gaussian filter to soften the obstacles.

The resulting image was downsampled and converted to a .txt to be fed into the motion planner. Pranav assisted me in the above debugging process of the local planner.

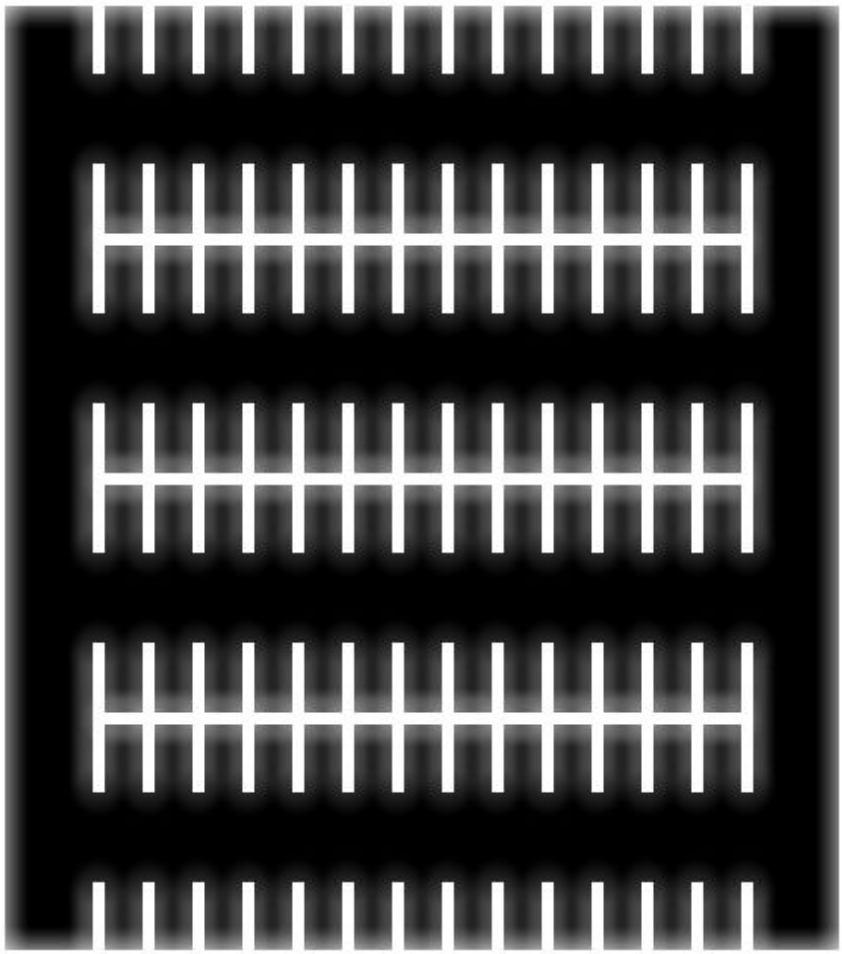

Figure 1: Final downsampled and inflated costmap

An additional cost was added as a soft constraint in the local planner for closeness to the obstacles based on the costmap with an associated weight. Following this, the motion primitives were edited to add additional motion primitives allowing for different curvatures and distances moved. As a general policy, the motion primitives were added to allow larger distances covered with large radii of turning and smaller distances with small radii of turning for sharper turns. This would allow the cars to navigate successfully in the parking lot. The weights on the path cost, obstacle cost and heuristic were also tuned. The effect of all this was a couple of sleepless nights and a state-space planner that successfully returns paths with an average time of 0.1 seconds. The paths returned remain at a reasonable distance from obstacles and enter the parking spots by backing into them. Figure 2 shows a screenshot of the solution rate from local planner.

```
Publish car # 19
[ INFO] [1460594480.588177304]: Received path query from rendering engine for [2
.500000, 2.000000] to [23.000000, 25.000000]
23.601 26.7097
Forming path
[ INFO] [1460594480.716697741]: Path returned in 0.128397307 seconds
```

Figure 2: Console screenshot showing successful output of the local planner

The underlying graph structure of the planner encoded in the lattice was formalized by dealing with discrete configurations in terms of node ids for collision checking. Additionally, multi-threaded spinning functionality was added to the planner to deal with a huge amount of queries from the global planner and rendering engine very fast.

INTEGRATION OF LOCAL PLANNER WITH GLOBAL PLANNER AND RENDERING ENGINE

The final local planner was integrated successfully with the global planner and rendering engine in two steps. First, the local planner was integrated with the rendering engine which gave it random goal locations. The obstacle free paths were visualized for different cars and can be seen in Figure 3. Later on, when the global planner was fully functional as well, all three components were integrated and spots were passed to the local planner according to the current heuristics implemented by the global planner which include distance from exit, queue size, time of day and state of parking spots. These heuristics are coupled with the dynamic cost of the actual path given by the state-space planner to decide on the final best spot. Figure 4 shows the visualization of the paths taken by the cars after integration of all three aspects. The rendering engine in figure 4 has been updated with models of different cars for aesthetic purposes.

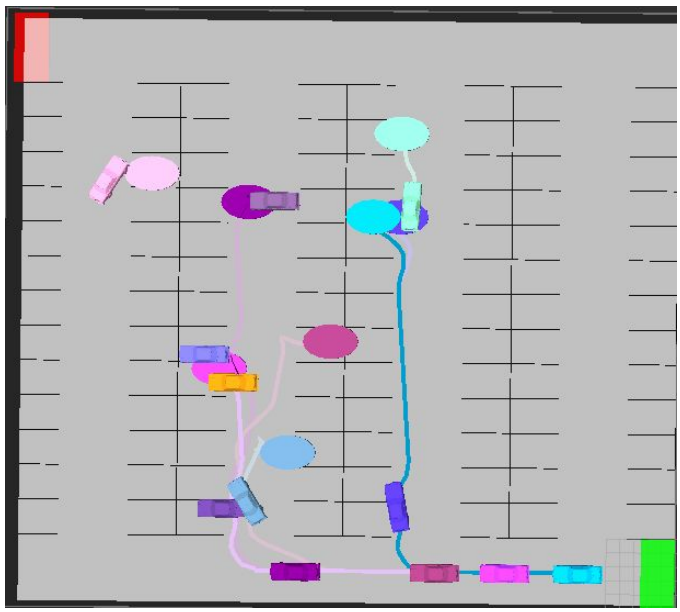

Figure 3: Paths visualized for random goal positions in final parking lot

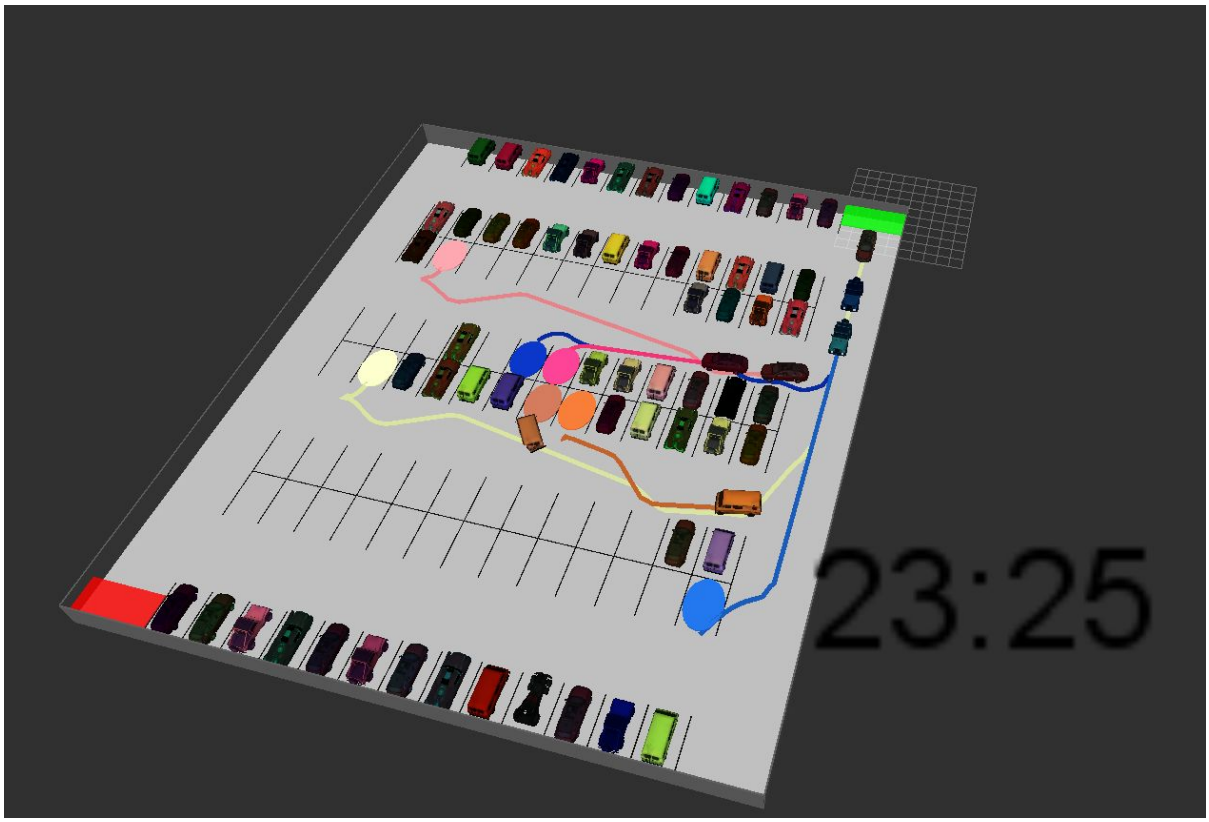

Figure 4: Paths visualized after integration of all components of simulation

(Contributors: Mohak, Shivam and Pranav)

SVE DEMOS WITH MULTIPLE PLATFORMS

I worked along with Shivam and Pranav on readying our SVE demos. Our SVE consists of two demos:

Demo 1

Demo 1 involves the physical platforms navigating through the parking lot and parking collaboratively. It showcases the entire pipeline from the app to communication, planning and navigation working seamlessly on two platforms. We were facing a few issues with the old platform related to WiFi card and bluetooth initially. After the new WiFi card was installed, there was still problems regarding absence of antennas because of which we were not able to SSH into the platform. These problems were fixed for the short time period by using one antenna from the other platform and the demo was run successfully.

Demo 2

Demo 2 involves the simulation of the planning aspect of the parking lot. The simulation requires all three components to work with each other from a single launch files. Since our different planners had been integrated and working together properly, this part of the demo went by smoothly.

2. Challenges

1. Debugging the local planner was a very involved process and required hours of intense work. One of the different reasons for this were the different nuances and layers of functionality in the local planner that need to be working perfectly right.
2. One of the custom functors for maintaining an STL set of poses was not functioning properly and thus a proper treatment of the discrete part of the environment had to be done by maintaining node ids.
3. Many hardware issue with the platforms required excessive amount of debugging and patience. The old platform took a long time to be ready and could not be tested beforehand.

3. Teamwork

The other members of my team were working on the following subsystems:

1. Shivam: He worked with Pranav and me on getting the SVE demos ready. He also worked on the global planner and in the integration of simulation environment with Pranav and me.
2. Pranav: He worked on the navigation of the platforms and getting the SVE demos ready with me and Shivam. He also completed different aspects of the rendering engine and setting up services to communicate the state of the parking lot to global and local planners. He also worked with me on the debugging of the local planner.
3. Dorothy: She worked with Richa on setting up the hardware and software of the old platform and on one of our reach goals which involves planning for different cars simultaneously navigating in the parking lot using a modified Astar approach.
4. Richa: She worked with Dorothy on setting up the hardware and software of the old platform on one of our reach goals which involves planning for different cars simultaneously navigating in the parking lot using a modified Astar approach.

4. Plans

Since our new platform has arrived, we will be working on developing all necessary functionality on that platform and keep our old platform as a backup. I will be working on improving the performance of the local planner by adding tuning the different weights of the cost and formalizing the process of weighting parking spots dynamically. Pranav will be working on completing the functionality of the rendering engine and Shivam will tune the different parameters of the global planner so that we can showcase the policies we want. Pranav, Shivam and I will also be working on ensuring the robustness of the physical platform and and full system integration. Richa and Dorothy will be working on the complete our reach requirement to showcase different cars parking simultaneously in the parking lot.