Individual Lab Report #08

Progress Review #9

October 15th, 2020

Alex Withers

Pit Navigator

Team G: The Pit Crew: Awadhut Thube, Justin Morris, Alex Withers

Individual Progress

Pit Navigator

Inputs and Outputs for RASM

During the first week, I analyzed the file structure from RASM to determine the usefulness of the code. I quickly realized that RASM is not Moonranger's only code and is only a small part of their whole. While we are only interested in their triangular meshing algorithm and their SLAM algorithm, there was a lot more to their code. I have shown a simplified version of their files in Figure 1.

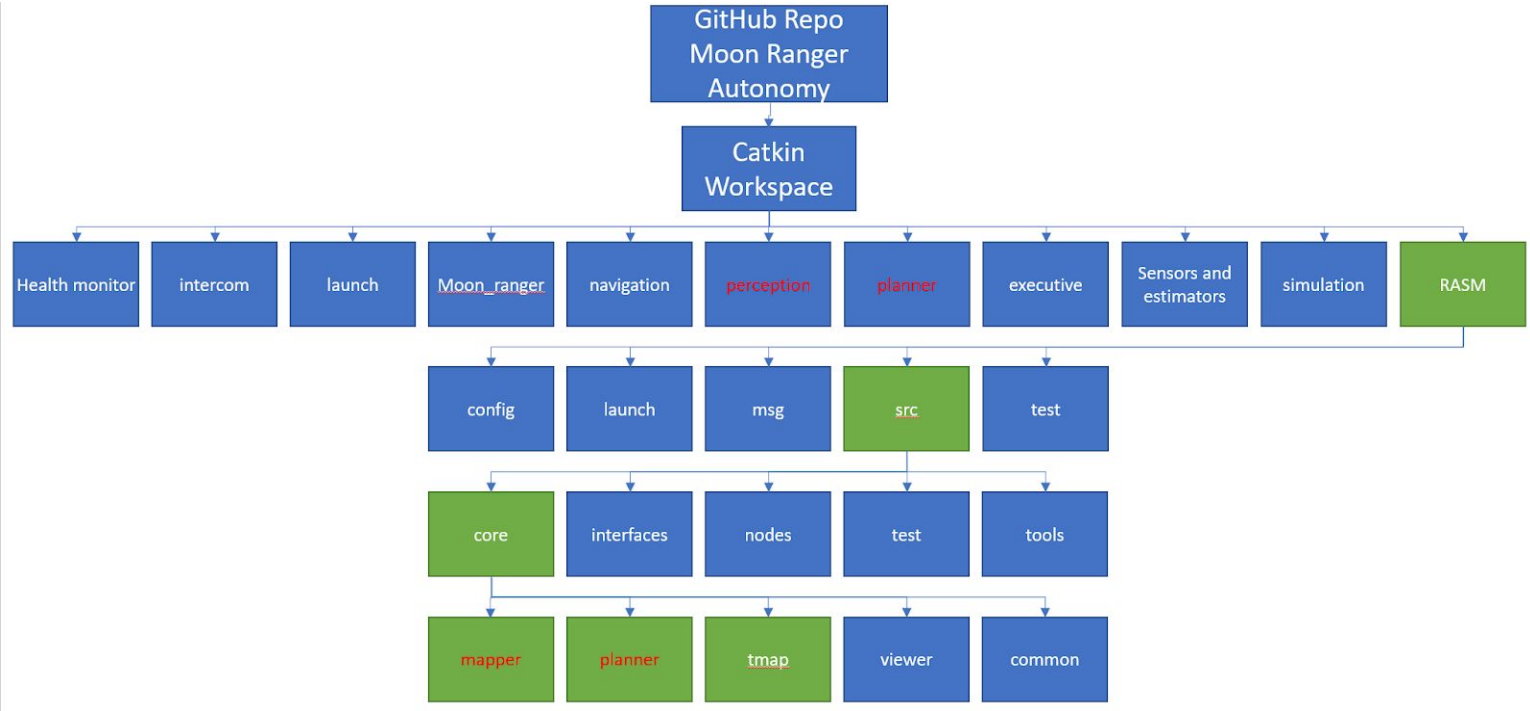

Figure 1: Moon Ranger Autonomy file structure with RASM expanded with green boxes. The boxes with red text seem to have duplicate logic.

My initial predictions, that the mapper class was important and the planner was redundant to us, proved to be false and true respectively, the mapper class was a visualization tool for the mesh representation of the environment and concatenating it over time. The Tmap class which was linked within the mapper class was a wrapper for the triangle library, which is a commonly used triangulation method found in many libraries that deal with point clouds. Ultimately we only cared about the triangle library but can find it elsewhere in easier to access libraries. I did find it to be true that the planner class was redundant, and it planned in 3 dimensions, x,y,z rather

than the x,y structure that we have been using. This means that it would take a more significant amount of work than we can provide to make it compatible. Additionally the boxes with red text inf figure 1 seem to have duplicate logic, covering the same functions. Mapper and perception both try to use stereo imagery and map it. The planner files want to plan through the map, but neither is clear as to which is acutally being used. This code clearly needs a trimming and rework like ours did to make it readable and understandable.

Pit Edge Creep Algorithm

I gave the algorithm that brings the rover closer to the pit edge a small upgrade this PR. I timed how long the rover is creeping forward before it detects the pit and use that calculated time to send the reverse commands to end up in the same position after the rover took its pictures at the pit. I also created and added a simulated IMU to the rover. The IMU has a resolution of 0.01 radians and publishes every millisecond. The IMU is critical to detecting unsafe slopes when the robot is tilted in any way as it allows the rover to see the terrain in the world frame, which will tell us what the true slope of an area is. I added the IMU within the Webots world simulation file, under the pioneer3at model's expansions. The IMU will share the same coordinates as base_link so that it is in the geometrical center of the rover. Lastly, I lent a hand to Awadhut and Justin during the integration of the codes.

Challenges

Pit Navigator

Inputs and Outputs for RASM

After downloading all of Moonranger's autonomy code and following their instructions to run it, I realized that it is set up for a later version of Ubuntu than our simulation and we are not prepared to upgrade. So when Justin set up a meeting with David Wettergreen, I attended and asked about how we might integrate with Moonranger code. David said that they don't even plan on using RASM or anything that they have currently for their flight code. He said that we would be better off just using the triangle library independent of RASM or Moonranger code and that it is contained within scipy, which we have easy access to.

We had also wanted to use their SLAM algorithm, ORB-SLAM2, which maps and localizes the rover from stereo imagery. Our buddy in Pit Ranger, Jordan was having a lot of trouble with getting it to properly localize with arcs and can't figure out how to use it with the IMU. It has become clear to me that we would add value if we took another approach to see if we could find another open-source algorithm that would fit with our system better. All this is piled on top of the fact that neither Moon ranger code nor Pit ranger code is complete and does not have the same timelines that we do. In general, finding the inputs and outputs for Moon ranger code seems to be out of scope for our project due to the reasons above. This means that we should find our own triangulation algorithm for creeping to the pit and a different SLAM algorithm.

Teamwork

Pit Navigator

Individual	Main	Sub	Description
Alex W.	Determine Input and output hooks for RASM	Add IMU and transforms, Quality of life bits within the simulation	I was to determine how we could use rasm to our advantage, and add the imu to simulation.
Awadhut T.	Create Mesh from the point cloud	Get Simulation Computer and set up remote access	Awadhut played a major role in creating a mesh from the point cloud after we determined that RASM was too much work. He also set up and is running the simulation computer at his house.
Justin M.	Read point cloud and determine a way to evaluate slope	Get Simulation Computer and set up remote access	Justin worked on getting information from the mesh that was created and colored the mesh based on the normals of the triangles. He also helped Awadhut set up the remote access for the computer.

Plans

Pit Navigator

Map Updates and Rover code

In preparation for the next PR, I will help with both deliverables. I will find a way to edit the map of the pit to match what we see in the mesh while running the brinksmanship code. I will also assist in setting up the code for use on the rover and have it plan and navigate to a few points in an apartment. I suspect that our motors will begin to cause us trouble for this task and I expect that the motor encoders will be difficult to work with as well.