Angad Sidhu
Team B Arcus
Logan Wan, Maitreya Naik, Clare Cui
ILR 6
Feb 2 2017

Individual Progress

Over the past few weeks we had started with many project management tasks like scheduling and prioritizing tasks. Since completing those I have focused on researching and implementing a more robust and effective integration of IMU measurements into our state estimation pipeline. The implementation is being built on the library GTSAM which implements iSAM2[1]. Our implementation of SLAM used GTSAM by adding incremental odometry estimates as factors in GTSAM's library. GTSAM would then use those factors to generate a state estimate based on all previously estimated factors. The addition of IMU measurements essentially adds both another estimate of position, velocity and orientation as factors into GTSAM. Because GTSAM keeps track of all previous factors in a Bayes net structure and constantly uses them to estimate position, the addition of a lot of factors can really have a large impact on performance. Therefore in order to reduce this impact, we can take a large amount of IMU data and integrate it before adding it as a factor[2]. Fortunately, GTSAM also provides a library which performs these calculations as well as calculates covariances and etc.

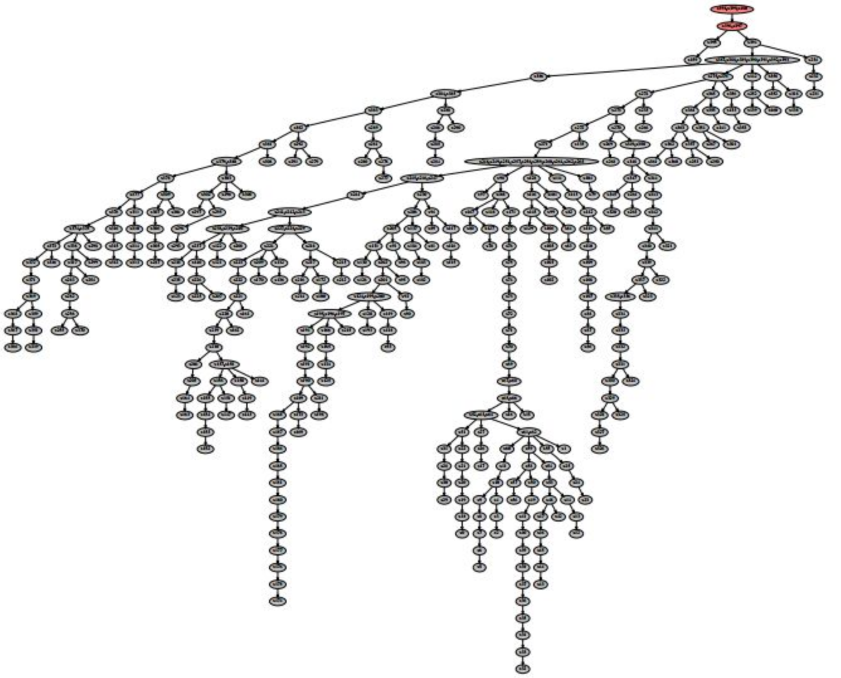

Figure 1. Represents an example of a factor graph(or tree) as generated by iSAM2. Referenced from [1].

I also helped everyone get our software (i.e. drivers and SLAM code) installed and running on everyone's computers. I also walked everyone through a high-level overview of the packages as the first step to prepare everyone for diving into the code. I also worked on creating a separate, self-contained repo for all of our ROS packages. This does violate ROS best practices in terms of code management however it is justified by the time it saves in our workflow. Because we have a lot of small packages (which was the structure of the BLAM package we selected) it can be cumbersome and error prone to commit and manage different branches for all the packages. By combining everything into a single repo, it simplifies the version management process significantly.

One problem we had last semester was a very low frame rate transfer of RGB images from our hex-rotor to our base station. This was being throttled by Wifi bandwidth and because the images were being transferred as full-size RGB images. I investigated various options in the ROS image_transport package which can help us encode our camera image stream as a video stream[3]. By using the Theora video encoder transport, we dropped bandwidth requirements from MB/s to a few kilobytes per second.

Challenges

I specifically have been struggling with implementation details and, more specifically, compiling and debugging issues. After following some example code provided by our advisor I've run into an issue while compiling the code base after integrating GTSAM. I'm still in the middle of debugging the precise issue but as of now all I know is that it is throwing an std::bad_alloc message. I have looked at my code that calls a GTSAM function and I am not totally sure why it is throwing this error yet. The steps that I am going to take to debug this issue will be to compile both GTSAM and Eigen in Debug configuration so that I can dig through the stack trace. This will aid in figuring out exactly what it is going wrong.

Teamwork

Maitreya and Clare have been looking into the flight controller replacement. This involved looking into how it would mechanically be mounted as well as how it would be electrically connected. Maitreya worked on updating the electrical design of the hex-rotor by routing power directly from the batteries to the PDB rather than behind the 3DR power unit of the Pixhawk. Clare has also worked on mounting the new GPS antenna and other CAD tasks related to the Delrin plates. Logan has been working on project management tasks like coming up with the schedule. He also has been working on CAD cleanup for the robot and completing some left over CAD tasks.

Plans

The beginning of this semester we focused on project management in rescheduling tasks. There is also a lot of unknown in some of the new tasks (like autonomy) that we have planned on picking up which requires research. I am going to investigate various algorithms and requirements that autonomy and exploration algorithms are going to require of us and our software stack. This will help inform decisions further when working on state estimation and map generation. I am also hoping to have IMU preintegration done within another week or two so that we can test the effect it has on our localization estimates. I'm also hoping to get everyone familiarized with the code so that they aren't blocked when working on their own individual tasks.

References

[1] iSAM2: Incremental Smoothing and Mapping Using the Bayes Tree (http://frc.ri.cmu.edu/~kaess/pub/Kaess12ijrr.pdf)
[2] IMU preintegration on manifold for efficient visual-inertial maximum-a-posteriori estimation (https://aragorn.library.gatech.edu/bitstream/handle/1853/55417/IMU%20Preintegration%20on%20Manifold%20for%20Efficient.pdf?sequence=1&isAllowed=y)
[3] ROS Image Transport Reference (http://wiki.ros.org/image_transport)