ILR08: Progress Review 11

Maitreya Naik

Team B: Arcus

Teammates: Logan Wan, Clare Cui, Angad Sidhu

ILR10

Apr. 6, 2016

1. Individual Progress

Since the last progress review, I was responsible for fixing a bug encountered in the motion manager for our quadcopter simulator.

1.1. Bug Tracking

Our quadcopter simulator setup had a bug that randomly presented itself where the drone would stop tracking the intended trajectory and deviate in a vertical direction.

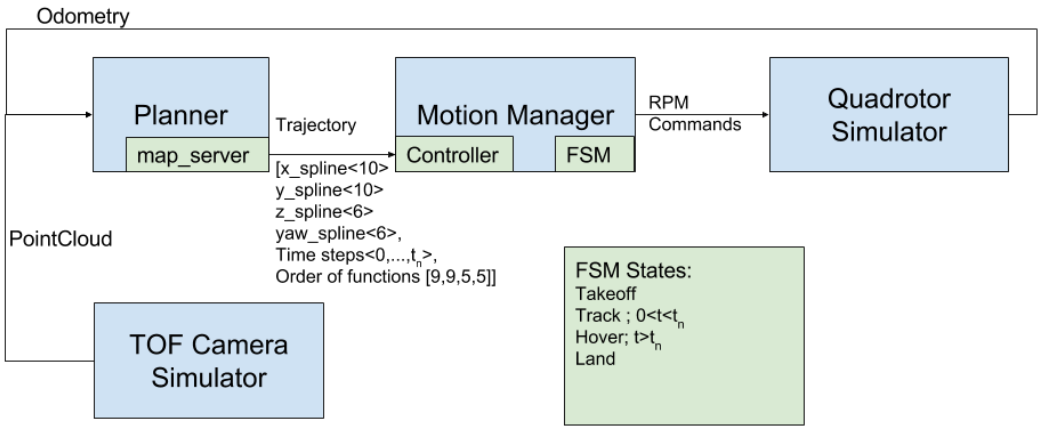

Figure 1: High-level motion planning architecture

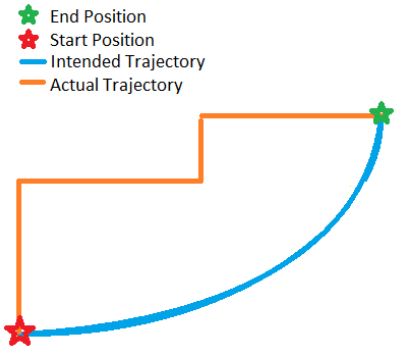

Figure 2: Bug representation. The quadcopter moves in horizontal and vertical motions

Initially, it was hypothesized that the bug was a side-effect of a time-based trajectory tracking, in which case a lag in the qaudcopter tracking the trajectory would lead to a deviating displacement vector as shown in Figure 3

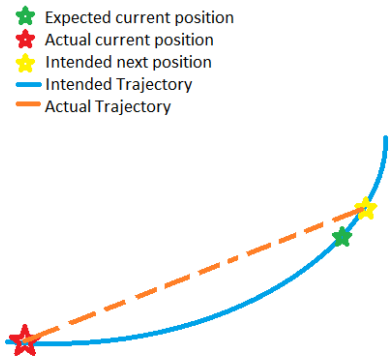

Figure 3: Hypothesized reason for bug

However, it was found that the bug was appearing because of a race condition in the case when the planner finished tracking the trajectory and shifted to “Hover” state before a new trajectory was received. This existence in the “Hover” state was in the case that the allotted planning time for tracking the trajectory was too high and the drone reached the end of the trajectory before this time. Since our trajectory is a queue of splines, when the drone is in “Hover” mode, the faulty trajectory generator is still creating vertical and horizontal movement splines. These splines get appended to our trajectory but are not visible until the quadcopter shifts into “Tracking” mode.

The timing of this race condition is depicted in Figure 4 (courtesy Angad Singh Sidhu). Reducing the planning time allotted to the planner led to this issue being circumvented. However, the issue of the erroneous lines being appended would still occur if the drone shifts into the “Hover” state. However, according to our supervisor and team, this workaround is sufficient for now so that more focus can be put on the obstacle avoidance part of the pipeline.

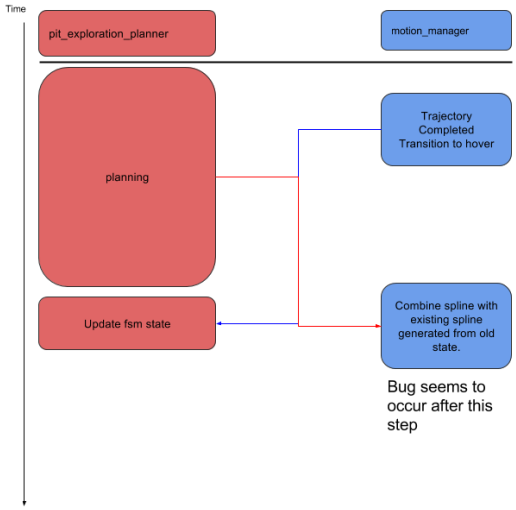

Figure 4: Race condition leading to erroneous vertical and horizontal movements being appended (courtesy: Angad Sidhu)

2. Challenges

The biggest challenge after getting a high-level understanding of the current motion planning architecture (Figure 1), was to dive into the code and try tracking down the exact cause for the bug. The code base consists of more than 15 packages along with ROS actionlib, which uses a Client and Server instantiation based on action messages. Leading to more virtual code place-holders and convoluted function calls.

Another challenge we faced was in the bringup for the small quadcopter to test the algorithm in the real-world with a motion capture system. For this purpose, we needed the assistance of Moses Bangura [1], Postdoc, RASL [2] as the quadcopter is based on the ODroid [3] Linux system. This system is more complicated to bring-up and he has the experience of the full procedure for the bring-up.

3. Teamwork

Clare and Logan finished the occupancy grid population with camera data.

Logan continued valuable project management.

Angad led the bug tracking with me and was key to pin-pointing the bug

4. Plan

I will be switching focus to the waypoint navigation and obstacle avoidance side of things from the trajectory generation. Angad will be working to bring-up the small ODroid [3] based quadcopter with Moses Bangura [1] in the RASL [2]. Clare and Logan will be finishing up on the camera data integration into the occupancy grid map and test the code on the full system by manually carrying around the system with the LiDAR and camera, using the motion capture system for state-estimates.

References

[1] "Moses Bangura," [Online]. Available: http://www.ri.cmu.edu/person.html?person_id=4504.

[2] C. Robust Adaptive Systems Lab, "Robust Adaptive Systems Lab, CMU," [Online]. Available: http://rasl.ri.cmu.edu/.

[3] H. C. Ltd, "Hard Kernel ODroid," [Online]. Available: http://www.hardkernel.com/main/main.php.