16-662 Robot Autonomy: Final Project Report
Symbiotic Localization

Group Members:

1. I-Chen Jwo (ijwo)
2. Pratibha Tripathi (pratibht)
3. Pulkit Goyal (pulkitg),
4. Daniel Arnett (darnett)
5. Akshat Agarwal (akshata)

Project Mentors: Eugene Fang, Red Whittaker

MOTIVATION

The goal for our project is to develop a symbiotic localization system and deploy it on a pair of planetary rovers, enabling them to be localized with much greater accuracy than if they were operating alone. This is motivated by the fact that rovers deployed in pairs can enable vastly superior technologies and scientific experimentation than a single rover, at the low cost of a small increment in rocket payload and development cost.

Planetary explorations to date have been solo acts. Rover pairs can cover more terrain, take more risks, accomplish more planned goals, localize more accurately, image one another and provide third-person views never before possible with solitary robots. The two rovers can be equipped with different scientific equipment for conducting a variety of tests, and can enable mission control to take greater risk, explore more areas and potentially serve as backups for the other.

Rovers are traditionally localized using data from wheel encoders and inertial measurement units. The data is put through a filter (like the Extended Kalman Filter, or EKF), and used to derive an estimate of the location of the robot with respect to its starting location. This process of dead-reckoning is highly prone to accumulation of drift over time, leading to incorrect localization which is obviously undesirable. This problem is severely compounded in a long-term autonomous deployment situation, like that of planetary rovers - since they basically do not get any ground truth estimate of their position at any point during their mission.

Localization accuracy for multiple rovers is improved over dead-reckoning by using each rover as a landmark for the other rovers. Since the errors due to drift accumulation in both rovers' localization are random, knowing the other rover's relative bearing acts as a sort of **loop closure**.

Formally, the problem of multi-rover co-localization using relative observations can be defined as follows. Given a team of multiple rovers, each with the capability of estimating its own state (position and orientation) using proprioceptive sensors (encoders and inertial measurement units) and the ability to observe other rovers using exteroceptive sensors (VIVE tracker, camera, range sensor, etc.), determine the set of states for all rovers over time that maximize the probability of obtaining the set of exteroceptive observations.

OBJECTIVE

We will work on two planetary rovers (Auto Krawler 1 and 2, respectively) developed in the Planetary Robotics Lab. Both are equipped with wheel encoders and an IMU. They have also been retrofit with the HTC Vive VR Tracker and 2 Base Stations, each. They have an onboard Odroid computer which runs ROS.

The ROS architecture for controlling a single rover by giving it waypoints has been provided to us. They can also be controlled manually using a PlayStation 3 wireless controller.

Fig 1: Rovers - Platform used

Our objective is to improve localization of the 2 planetary rovers using their relative bearing measurements with a HTC Vive Tracker and a camera. This will involve:

1. **HTC Vive Tracker Pipeline**: Using the HTC Vive Trackers and Base Stations to get the relative pose for the 2nd rover
2. **Vision Pipeline**: Tracking one rover with a pan-tilt camera onboard another, and using that to get a relative bearing for localization
3. **System Integration:** Filtering these measurements with wheel odometry for robust collaborative localization.

KEY CHALLENGES

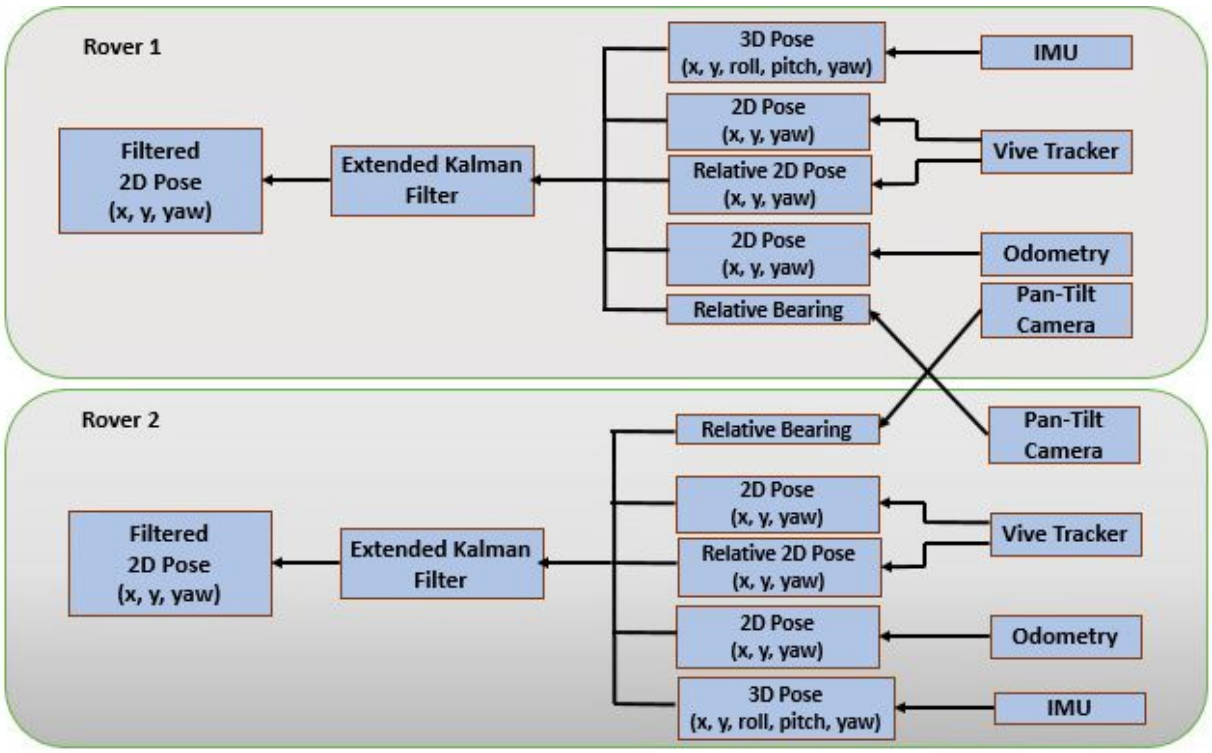
Getting both Vive Trackers working robustly with fluid base station interchange

   a. Each rover is being retrofit with 2 Base Stations, one facing forward and the other backward. Since the field of view of each base station is approx. 110^{o}, this leaves a blind zone in between, when the two rovers are roughly side-by-side. In this interval, when the vive tracker is not functional, the rovers will be localized using wheel encoder and IMU data. When the rover enters the view of one base station of one base station after having exited the view of the other station a while back, it is important that there be a robust snap back of the localization to correct drift accumulated in the blind zone. This is quite complex to implement robustly.

2. Detecting the other rover in camera feed and controlling the pan-tilt camera to keep it in sight
   a. To further increase robustness of localization and provide redundancy in localization methods (as backups in case of malfunction/unexpected contingencies), the rovers will be equipped with a pan-tilt camera. Each rover will learn to identify the position of the other rover in its camera feed, and use optical flow along with PID control to control the pitch, roll and yaw of the camera gimbal such that the other rover is kept firmly in sight.
   b. Using monocular visual localization methods, we can estimate the relative bearing of the other rover via the camera, and use that as another source of data for the localization filter to obtain an even more accurate estimate of the other rovers' position.

3. Integrating both relative pose measurements with wheel odometry
4. Constraining paths to maintain rover line-of-sight within 6m

SYSTEM ARCHITECTURE



Components and their utility in our system:

1) **Rovers:** These are serving as platform for our project.
    a) Input: From all sensors.
    b) Output: Execution of the path

    For path planning in these rovers we are using the teb_local_planner package of the ROS.
2) **HTC Vive Tracker :** We are using these to improve the localization of rovers with respect to each other.
    a) Input: IR of the other rovers vive base
    b) Output: Localizing/Pose correction w.r.t other robot's pose
3) **Intel RealSense:** It is used to get the dense point cloud of the environment. Ultimately they will be used to segment the obstacle and place that in our local cost map. It will be finally used in obstacle avoidance while path planning.
    a) Input: Environment visual details
    b) Output: Point cloud of environment finally converted to obstacles in map.

    For converting the point cloud to obstacles in occupancy grid, we are going to use Elevation mapping package from ROS. We had to make considerable amount of changes to read the data from appropriate topic and publish the data in right format on right topic. We have just completed the basic integration of this package. It'll require considerable amount of modification in the base package to make it useful for our use case.

4) **Pan-tilt camera:** For visual perception of rovers. This is extended goal as they don't impact the primary functionality of the system.
    a) Input: Perceptual information from surroundings
    b) Output: Images of other rover.

    We are still facing integration issues in the camera. We are in touch with the DJI team for the same.

WORK COMPLETED

Major Work completed in various areas is as follows:

1) Platform:

We have integrated the path planning algorithm - Time-Elastic-band Path planner (teb) for both the rovers.The initial trajectory generated by a global planner is optimized during runtime w.r.t. minimizing the trajectory execution time. We are able to do this on the basis of just the wheel odometry.

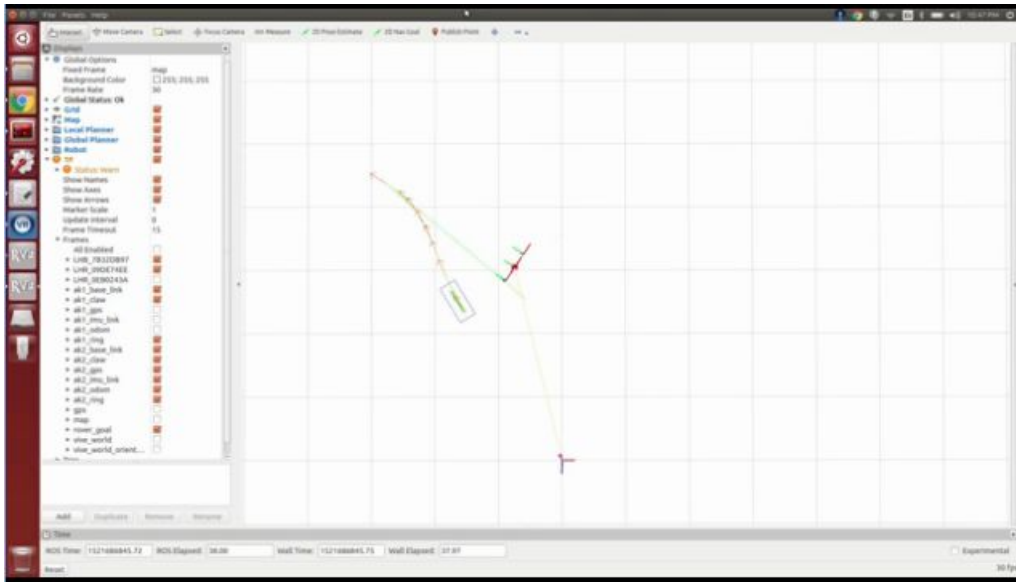


Fig 2: Teb local path planner

2) **Vive Tracker:**

We have integrated the vive trackers with both the robots on one side. If we plan to use the leapfrog approach, we need to have vive trackers on both side for both rovers. We need to mount and test vive trackers on both sides of both the rovers. Even the when both the rovers are parallel to each other the vive tracker will lose its signal and the rover will be moving only on the basis of the wheel odometry. So, we are planning to test other approaches as well, the ones suggested during our presentation.

Image Courtesy: https://www.vive.com/us/vive-tracker/

Fig 3: Vive Tracker

3) **Intel RealSense:**

We are able to get the point cloud from Intel RealSense on ROS Node. We are working on integrating the obstacles on occupancy grid or costmap, using Elevation Mapping ROS stack. This method is beneficial to reduce the uncertainty in the map around the robot.

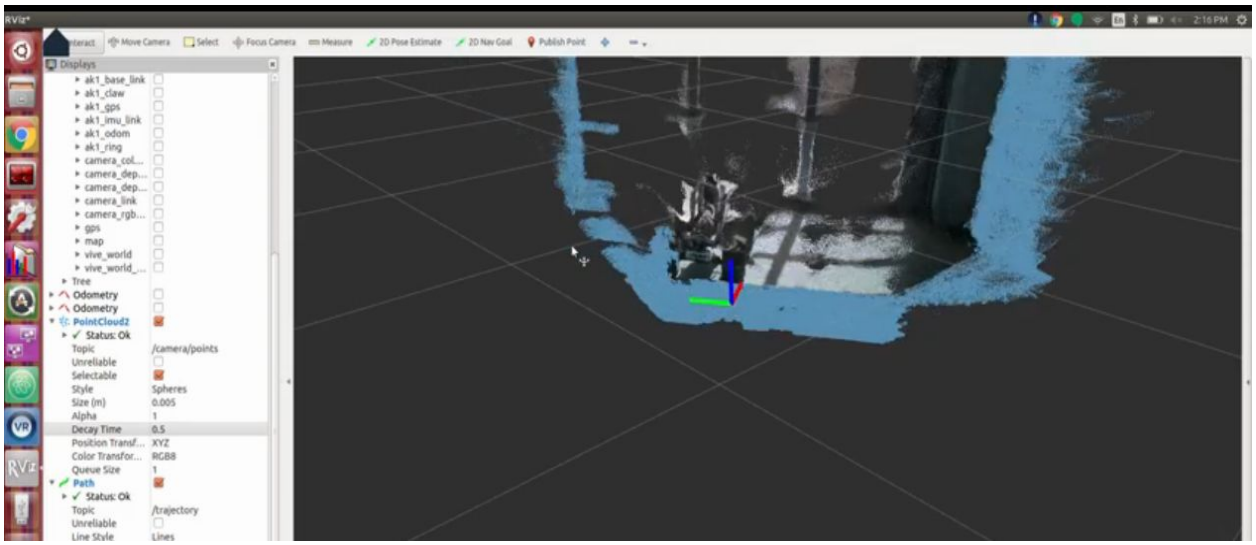


Fig 4: Point cloud from Intel RealSense

4) **Pan-tilt Camera:**

We are having some issues with integrating the pan tilt camera. We have contacted DJI for their support as it is a problem that when they release a new hardware they stop the support for the previous hardware. We are trying to fix that, but this doesn't hinders with the basic intended functionality our system, so this is a secondary goal for us.

Image Courtesy: https://www.dji.com/zenmuse-x3

Fig 5: Pan-tilt camera

WORK TO BE DONE

We have broken the whole project in some major tasks as given below (Fig 6). The number of days taken for the tasks are higher than they might take, keeping the buffer for other project deadlines in running in parallel.

We have done some primary work on the planner, getting point cloud from RealSense and Vive tracker, as explained in detail above.

Further plan of work is as follows:

a) We have to detect obstacle using RealSense and integrate them in costmap.
b) Test the path planner with this new cost map.
c) We have to mount another set of vive trackers on both the rovers. We have to test the vive trackers.
d) There were some suggestions for using different maneuvering options apart from leapfrogging. We have to test those configurations.
e) Integrating pan-tilt camera on the rovers this is a stretch goal as this is not required for the primary functionality of our system.
f) Then complete integration and testing of the system.

Fig 6: Gantt Chart of Project Major Timelines

Task Name	Duration	Start	Finish
Plan to a given destination	6d	03/27/18	04/03/18
Detect obstacles by Realsense	9d	04/04/18	04/16/18
Use Vive Trackers to improve localization	9d	04/06/18	04/18/18
Integrate the Realsense on the rover	7d	04/12/18	04/20/18
Integrate the pan-tilt camera	9d	04/12/18	04/24/18
Leapfrogging to the given destination	4d	04/19/18	04/24/18
Detect obstacles dynamically	9d	04/17/18	04/27/18
Integration	5d	04/24/18	04/30/18
Testing	5d	04/24/18	04/30/18
Final Presentation	1d	05/01/18	05/01/18

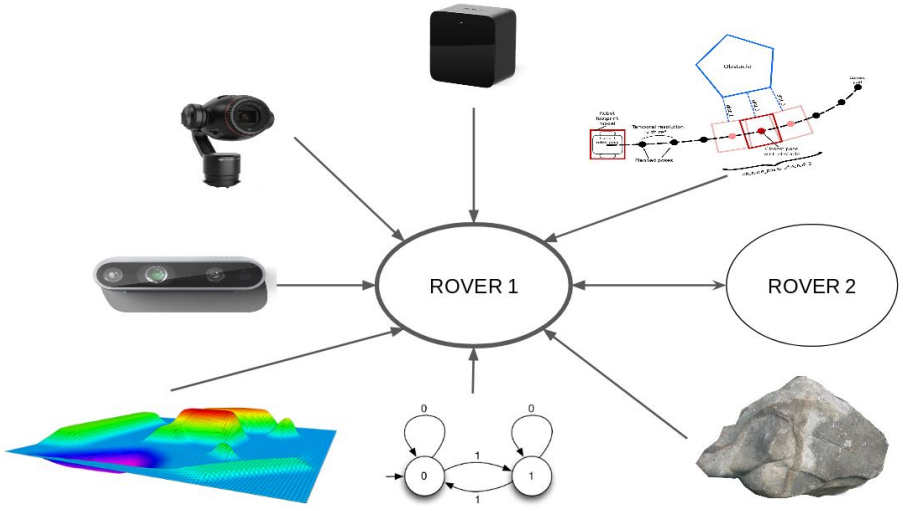


Fig 7: Components to be integrated on both rovers