Progress Review 12

Tae-Hyung Kim (tk1)

Team B / Auto Pirates

Teammates: Tushar Chugh, Shiyu Dong, William Seto,
Bikramjot Hanzra

ILR #11

April 13, 2016

1. Individual Progress

1. Robot Localization Package test

I've been studying on robot localization package to test previous rosbag data stored in previous field test. Robot localization package provides nonlinear estimation through sensor fusion of an arbitrary number of sensors like IMU, GPS, Camera, and Odometry like the following figure 1.

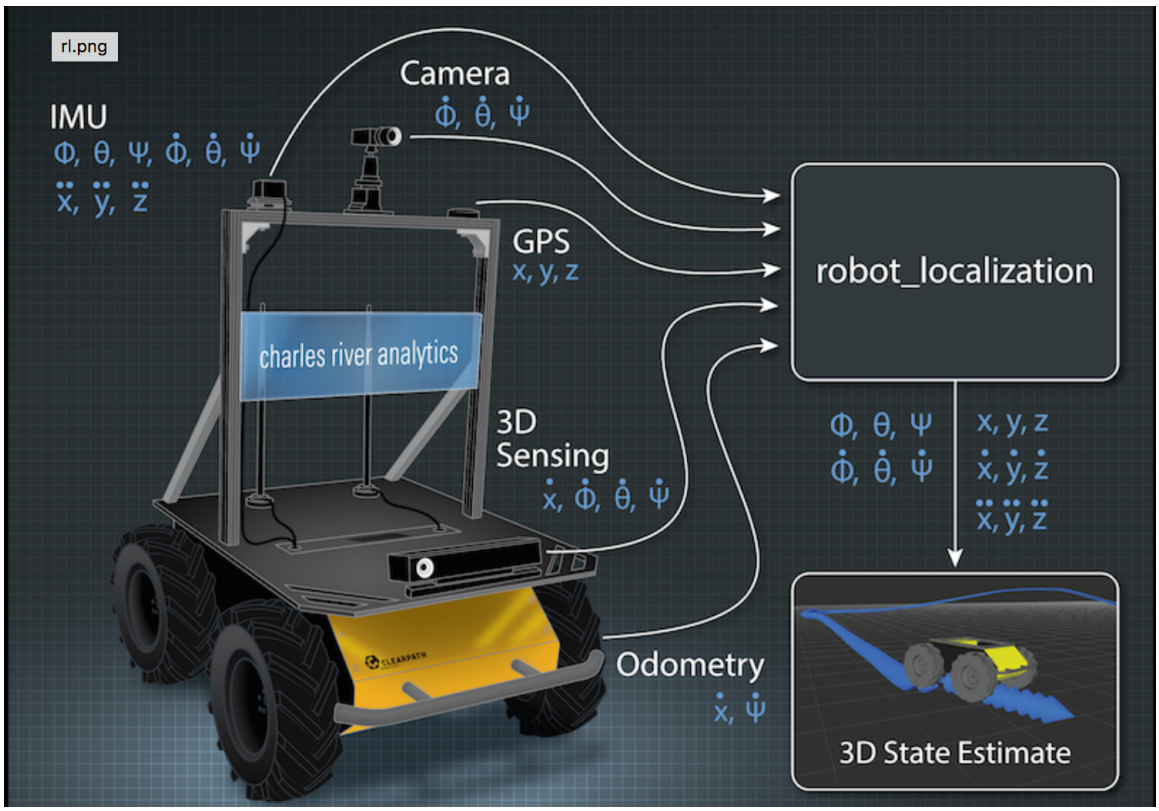

Figure 1. Structure of robot_localization package

The robot_localization package is composed by state estimation nodes, which are ekf_localization_node and ukf_localization_node. The robot_localization package provides navsat_transform_node, which provides the GPS integration into robot's localization.

For installing robot_localization package, I used installation from source like the following commands

```
$ source /opt/ros/indigo/setup.bash
$ cd catkin_ws/src
$ git clone —recursive https://github.com/cra-ros-pkg/robot_localization —branch jade-devel
$ cd ..
$ catkin_make -DCMAKE_BUILD_TYPE=Release
$ source catkin_ws/devel/setup.bash
```

There are some standard parameters that I should check like the followings.

1. frame parameters: These parameters define the operating mode for robot_localization. Specifically, REP-105 specifies following map, odom, and base_link.

- map_frame: The coordinate frame called map is a world fixed frame where Z-axis points upwards. This map_frame is not continuous, so the robot can jump its position in a discrete way. In our system, map_frame is used by the occupancy grid map of the river.

- odom_frame: The coordinate frame called odom is a world-fixed frame but the robot can drift over time. The robot's pose in odom_frame is continuous unlike the map_frame.

- base_link_frame: The base_link frame is attached to the mobile robot base.

- world_frame: If fusing global absolute position data (GPS or position update from landmark observations), we should set world_frame to map_frame value. The default values for world_frame parameter should be value of odom_frame.

The relation between frames are like the following.

map —> odom —> base_link

This shows that map frame is the parent of odom and odom is the parent of base_link frame.

2. odomN, twistN, imuN, poseN parameter: Users need to define these parameters using published messages. For our case, we have /imu/data /navsat/odom. So I can define the parameter like the followings.

```
<param name="imu0" value="/imu/data"/>
<param name="odom0" value="/navsat/odom"/>
```

The index for each parameter should be sequentially so I define it from 0-index.

3. odomN_config, twistN_config, imuN_config, poseN_config
The values in this parameter determine what variables should be fused into the final state estimate. The parameters type is like the below.

```
<rosparam param="imu0_config">[X                , Y                , Z                ,
                               roll             , pitch            , yaw              ,
                               X velocity       , Y velocity       , Z velocity       ,
                               roll velocity    , pitch velocity   , yaw velocity     ,
                               X acceleration   , Y acceleration   , Z acceleration]
</rosparam>
```

In my setting, I defined it like the following.

```
<rosparam param="imu0_config">[false            , false            , false            ,
                               false            , false            , false            ,
                               true             , true             , true             ,
                               true             , true             , true             ,
                               true             , true             , true             ]
</rosparam>
```

After setting the these parameters, the robot_localization package made filtered localization data of the robot through Extended Kalman Filter (EKF) or Unscented Kalman Filter (UKF).

2. Challenges

In this work, I couldn't understand the relationship of each frame. However, in the mobile robot class, frame conversion part was covered. So it helped me to understand the relationship of frames in the robot_localization package.

3. Teamwork

1) Shiyu: Shiyu worked on display velocity and add routes for goal and integrate GUI with planner

2) Tushar: Tushar worked on improving the waypoint skipping functionality, and adding the stopping condition for when the boat reaches the goal.

3) William: William worked on the elliptical obstacle cost inflation.

4) Bikram: Bikram worked on adding synthesized voice messages to the user in the GUI.

4. Future Plans

1) Supporting the SVE

As SVE schedule is fixed, I'll support and help to the team mates. There are some minor issues that should be resolved in SVE like navigation of waypoints and avoiding obstacle more securely. And then I'll review and elaborate our code so that the code could be more neat and tidy.