ELEC 5660: Introduction to Aerial Robotics
Project 3: Phase 2

Assigned: 16-04-2023 Due: 11:59 PM 10-05-2023

1 Project Work

You already implement the extended Kalman filter to fuse IMU with absolute pose measurements from PnP in Project 3 Phase 1. In Project 3 Phase 2, you need to extend the EKF to an Augmented State EKF, to fuse not only absolute pose measurements from PnP, but also relative pose measurements from the stereo visual odometry. The final filter should be able to account for failures in any one of the measurements, i.e., both the absolute and relative pose measurement may be unavailable sometimes.

Note:

(1) This project is rather difficult. Please start as early as possible.

(2) This is an individual project, so you should finish it all by yourself.

(3) Please read the comments of the code carefully. **You are strongly recommended to follow the predefined structure / variables / functions** to implement the Augmented State EKF. If bugs occur in your submitted code, we will deduct your points.

You will be provided with a ROS package named `aug_ekf`, which is the skeleton code for the filter. You need to use it to implement the Augmented State EKF to fuse the IMU, PnP-based pose measurements and VO-based relative pose measurements.

Two bags are given for testing your code:

(a) A large bag *bag_aug_ekf.bag* contains IMU data, images from a forward-looking stereo camera and images from a downward-looking monocular camera. We recommend you can use this bag, for which you need to integrate the PnP, visual odometry and EKF together to form a complete system.

(b) A small bag *imu_pnp_vo.bag* contains IMU data, absolute pose from the PnP and relative pose from the VO that are calculated using bag (a). This bag is provided for easier testing at the beginning, since the measurements from PnP and VO are handy. Also, for students who does not complete the previous PnP or VO projects, you can have the correct measurements to finish this project. Note that the absolute pose from PnP is already expressed in the world frame and the relative pose from VO is expressed in the body frame of the key frame (the relative pose of the current body frame relative to the body frame of the key frame). You may also plot the */tag_detector/odom_ref* and */vo/Odometry* to visualize the sequence of pose in the world frame estimated by PnP and VO respectively.

2 Tutorial

This project is based on the previous one, where you implemented the EKF to fuse IMU and PnP pose measurements. In this project, we go one step further to fuse relative pose from visual odometry. The estimated states need to be published using the *nav_msgs::Path* message. A path is shown in Fig. 1 for your reference.

Be careful about the frame transformation. We suggest you to transform the relative pose measurement into the body frame (aligned with the IMU), the pose from the PnP into the world frame (Forward-Left-Up and origin at the marker's origin) in your previous projects before you fuse them. In the previous project of PnP, the pose you obtained directly from solving PnP is the pose of the origin of the marker in the camera frame. The rotation from the tag to the world $\mathbf{R}_{wt}$ is:

$$\mathbf{R}_{wt} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & -1 \end{bmatrix}$$

Meanwhile the transform from the downward looking camera to the body $\mathbf{T}_{bcd}$ is:

$$\mathbf{T}_{bcd} = \begin{bmatrix} 1 & 0 & 0 & 0.07 \\ 0 & -1 & 0 & -0.02 \\ 0 & 0 & -1 & 0.01 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

If you are using the small bag, then the pose from PnP is already transformed.

2.1 Sensor Fusion Model

As you have learned from the example of the Lecture, to fuse the relative pose, a copy of the past pose associated with the keyframe of the visual odometry should be maintained. The copied state and the original state together form the augmented state. Based on it, you need to slightly change the process model of IMU and measurements model of absolute pose in accordance with the new augmented state. Then you need to implement the measurements model of relative pose, calculate its Jacobian matrices and follow the formulas of EKF to fuse the relative pose.

2.2 Dealing with Out-of-Order Measurements

When fusing multiple measurements, it is possible that the measurements arrive out-of-order to the filter, that is, a measurement that corresponds to an earlier state arrives after the measurement that corresponds to a later state. For example, an absolute pose corresponding to t_1 arrives after a relative pose corresponding t_2, but $t_1 < t_2$. There are also similar cases between IMU & PnP and IMU & VO measurements. To deal with the out-of-order data, we store the incoming measurements in ascending order of time. Each measurement is attached with a time stamp and the mean and covariance matrix of the associated state. When a new measurement arrives, it is searched along the container where the new measurement should be inserted. Then, using the inserted measurements and other measurements after it, a few predict/update steps are performed to recalculate the subsequent means and covariance matrices.

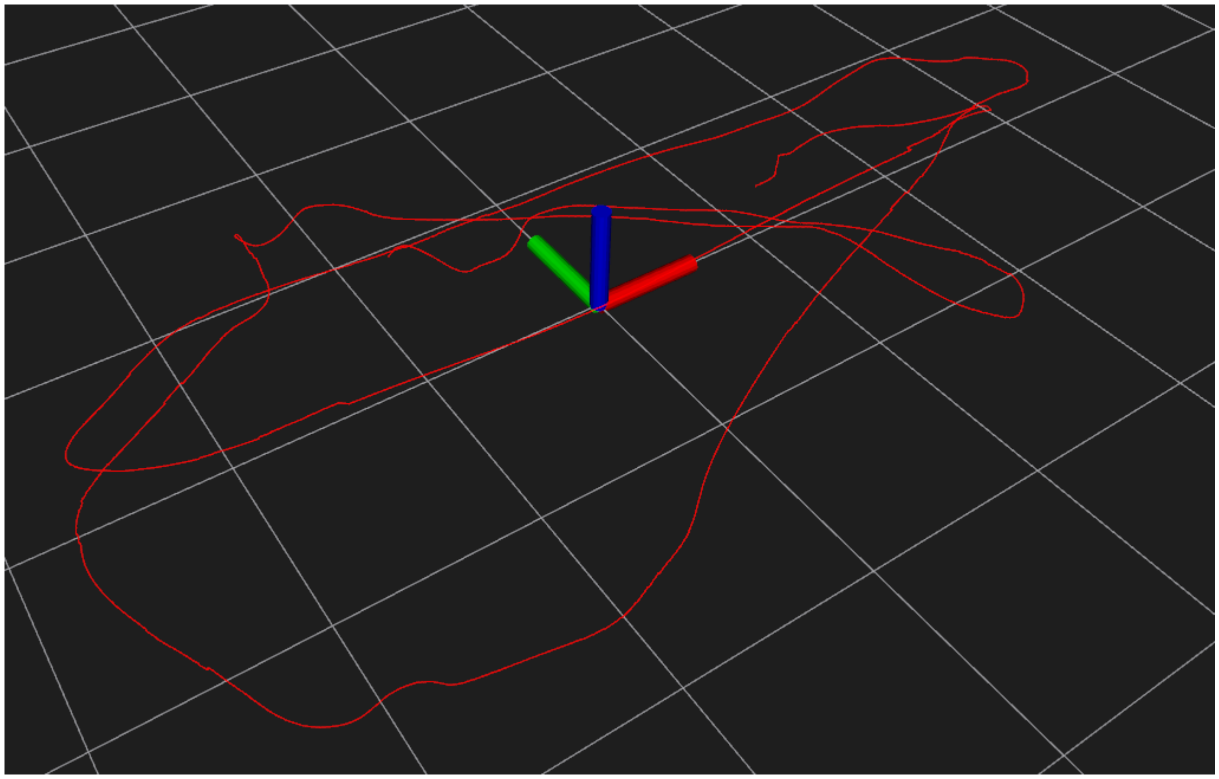

Figure 1: The estimated path of the given bag using the filter.

2.3 Fusing Relative Pose

When fusing relative pose measurements, it is essential to have a correct copy of the past pose and the covariance matrix of the augmented state. In particular, when the keyframe changes, the copied state should also change according to the timestamp of the keyframe. Similar to dealing with disordered measurements, we need to search along the container for the state that corresponds to the keyframe and change the copied part of the state as well as the covariance matrix. Then we insert the relative pose measurement and do predict/update (starting from the state whose copied part is just changed) to recalculate the subsequent means and covariance matrices.

3 Other Implementation Information

3.1 config files and launch files

Two new config folders are provided for `stereo_vo_estimator` and `tag_detector`. Please put the files inside into the corresponding folder and use them with your previous code to run the bag provided. You can use the `augekf.launch` to run the code and you may modify the launch file according to your configuration.

3.2 Pipeline

$$Callback\ functions \rightarrow processNewStates \begin{cases} insertNewState \\ initFilter \\ repropagate \\ removeOldState \\ publishFusedOdom \end{cases}$$

Also, please read the comments in the code carefully before you start.

4 Submission

When you complete the tasks you could submit your code and documents to canvas before **May 10th, 2023 23:59:00**. The title of your submission should be "`proj3phase2_YOUR-NAMES`".

Please cite the paper, GitHub repo, or code url if you use or reference the code online. Please keep academic integrity. **Plagiarism** is not tolerated in this course.

Your submission should contain:

1. A **maximum 2-page** document including:
   (a) Figures plotted by **rqt_plot** and **rviz**.
   (b) Descriptions about your implementation.
   (c) Any other things we should be aware of.
2. **All files** you need to run your code except for the bag (including the **tag_detector** and **stereo_vo** package you used).