Project 3: Visual Inertial SLAM

Collaboration in the sense of discussion is allowed, however, the assignment is individual and the work you do should be entirely your own. See the collaboration and academic integrity statement here: https://natanaso.github.io/ece276a. *Books, websites, and papers may be consulted but not copied from. It is absolutely forbidden to copy or even look at anyone else's code.* ***Please acknowledge in writing people you discuss the project with.***

Submission

You should submit the following files on **Gradescope** by the deadline shown at the top right corner.

1. **Programming assignment**: upload all code you have written for the project (do not include the provided datasets) and a README file with a clear, concise description of the main file and how to run your code.
2. **Report**: upload your report in pdf format. You are encouraged but not required to use an IEEE conference template[1] for your report.

Problems

In square brackets are the points assigned to each problem.

1. Implement visual-inertial simultaneous localization and mapping (SLAM) using an Extended Kalman Filter in *Python*. You are provided with synchronized measurements from an inertial measurement unit (IMU) and a stereo camera as well as the intrinsic camera calibration and the extrinsic calibration between the two sensors, specifying the transformation from the IMU to the left camera frame. The data includes:
   - **IMU Measurements**: linear velocity $\mathbf{v}_t \in \mathbb{R}^3$ and angular velocity $\boldsymbol{\omega}_t \in \mathbb{R}^3$ measured in the body frame of the IMU.
   - **Time Stamps**: time stamps τ_t in UNIX standard seconds-since-the-epoch January 1, 1970.
   - **Intrinsic Calibration**: stereo baseline b (in meters) and camera calibration matrix:
   $$K = \begin{bmatrix} fs_u & 0 & c_u \\ 0 & fs_v & c_v \\ 0 & 0 & 1 \end{bmatrix}.$$
   - **Extrinsic Calibration**: the transformation ${}_IT_C \in SE(3)$ from the left camera to the IMU frame.

   Implement an EKF prediction step based on $SE(3)$ kinematics with IMU measurements and an EKF update step based on the stereo camera observation model with feature observations to perform localization and mapping. In detail, you should complete the following tasks:

   (a) [15 pts] **IMU-based Localization via EKF Prediction**: Implement the EKF prediction step based on the $SE(3)$ kinematics and the linear and angular velocity measuremetns to estimate the pose $T_t \in SE(3)$ of the IMU over time t.

   (b) **Feature detection and matching**: We are going to prepare the feature tracks that will be used for landmark mapping. The goal is to obtain the pixel coordinates $\mathbf{z}_t \in \mathbb{R}^{4\times M}$ of detected visual features with correspondences between the left and the right camera frames, and then track the detected features in the left image over time (see Fig. 1). The landmarks i that are not observable at time t have a measurement of
   $$\mathbf{z}_{t,i} = \begin{bmatrix} -1 & -1 & -1 & -1 \end{bmatrix}^\top.$$
   There are many features in the dataset and you can think of ways to keep the computational complexity manageable. You have two options, and you need to choose **one**:

[1] https://www.ieee.org/conferences_events/conferences/publishing/templates.html

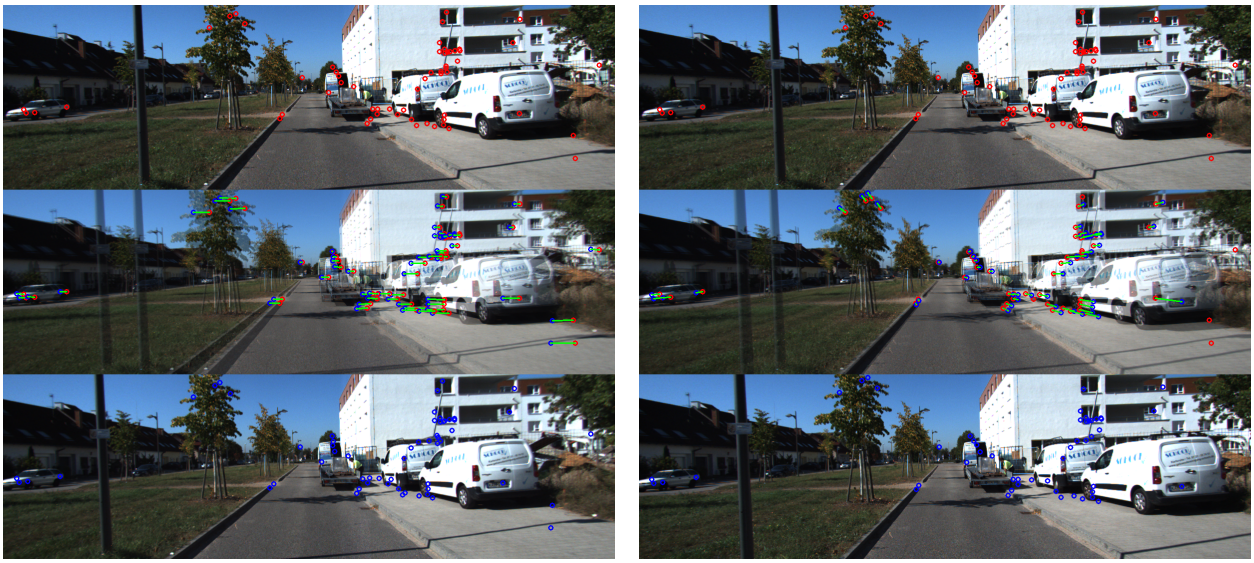

Figure 1: Visual features matched across the left-right camera frames (left) and across time (right).

- [0 pts] Use dataset *10.npz*, which includes features already. In this case, you do not have to do anything else for part (b) and can proceed to parts (c) and (d) directly.
- Use dataset *03.npz*, which does not provide features. In this case, you need to perform feature detection and tracking on your own and you will receive **extra credit**.
  i. Download the left and right colored images from the link[2]. The zip file *03.zip* is about 2 GB and contains left images (00) and right images (01).
  ii. [5 pts] Detect features in the left image using the Harris corner detector covered in the lecture. You need to implement this on your own and the OpenCV function *cornerHarris* is *not allowed*.
  iii. [5 pts] Perform stereo matching using the features detected on the left image. Use optical flow to track the features in the left image to the right image to find correspondences. You are allowed to use the OpenCV function *calcOpticalFlowPyrLK*. You are also allowed to refer to *image_processor.cpp*[3], which is the feature tracking part implemented in C++. However, you are *not* allowed to directly copy from any other Python repositories on the internet. Performing RANSAC is optional.
  iv. [5 pts] Perform temporal feature tracking using the features detected in the left image to the left image in the next time instance, using optical flow. You are allowed to use the OpenCV function *calcOpticalFlowPyrLK*.

(c) [15 pts] **Landmark Mapping via EKF Update**: assume that the predicted IMU trajectory from part (a) above is correct and focus on estimating the landmark positions. In detail, you should implement an EKF with the unknown landmark positions $\mathbf{m} \in \mathbb{R}^{3\times M}$ as a state and perform EKF update steps after every visual observation $\mathbf{z}_t$ in order to keep track of the mean and covariance of $\mathbf{m}$. Note that we are assuming that the landmarks are static so it is not necessary to implement a prediction step. Moreover, since the sensor does not move sufficiently along the z-axis, the estimation of the z coordinate of the landmarks will not be very good. You can assume that the z coordinates for all landmarks are 0 and focus only on estimating their xy coordinates.

(d) [20 pts] **Visual-Inertial SLAM**: combine the IMU prediction step from part (a) with the landmark update step from part (c) and implement an IMU update step based on the stereo camera observation model to obtain a complete visual-inertial SLAM algorithm.

[2] https://drive.google.com/file/d/1l3RlzuFIYIsPEoUrRVZmRBSpEZcDcAZW/view?usp=sharing
[3] https://github.com/KumarRobotics/msckf_vio/blob/master/src/image_processor.cpp

2. Write a project report describing your approach to the visual-inertial SLAM problem. Your report should include the following sections:
   - [2 pts] **Introduction**: discuss why the problem is important and present a brief overview of your approach
   - [8 pts] **Problem Formulation**: state the problem you are trying to solve in mathematical terms. This section should be short and clear and should rigorously define the quantities you are interested in.
   - [20 pts] **Technical Approach**: describe your approach to visual-inertial localization and mapping.
   - [20 pts] **Results**: present your results, and discuss them – what worked, what did not, and why. Make sure your results include plots clearly showing the estimated robot trajectory as well as the estimated 2-D positions of the visual features. If you have videos do include them in the zip file and refer to them in your report!