ECEn 631 Visual Odometry

Objectives:

- Learn to detect, describe, and match feature points.
- Learn feature matching and feature tracking.
- Learn to compute the essential matrix from images taken by one camera from multiple views.
- Learn to estimate camera pose from matched feature points.
- Learn visual odometry and its limitations.

Instructions:

- This is a 2D-2D Visual Odometry assignment. No 3D information is needed.
- There are two image sequences for this assignment. Camera intrinsic parameters and fixed scale factors are provided.
- The first sequence (VO Practice Sequence) is for practice and confirmation of the correct functioning of your code. Ground truth for this sequence is included.
- The second sequence (BYU Hallway Sequence) is for testing the accuracy of your VO algorithm.
- Feature matching with skipping frames can be performed to increase the baseline to increase accuracy. For example, you can choose to match Frame 0 to 2 and Frame 2 to 4, etc.
- Any accuracy improving methods discussed in published technical papers can be selected and implemented to improve your algorithm's performance.
- Include your result, images, and discussion for all tasks in one PDF file.
- Submit your PDF file and source code file(s) in one zip file without the folder or directory.
- Use your first name and last name (e.g., justinsmith.zip) as the file name.
- Login to myBYU and submit your work through BYU Learning Suite online submission.

Preparations:

- Download and study the VO tutorial paper (two parts) posted on BYU Learning Suite.
- Download the practice package (VO Practice Sequence.zip) from BYU Learning Suite that includes
  - 702 images of the practice sequence in a folder called "VO Practice Sequence".
  - Camera intrinsic parameters (VO Practice Camera Parameters.txt). Distortion coefficients are not available.
  - Ground truth plot (VO Practice Sequence Ground Truth.png).
  - Frame by frame R and T (VO Practice Sequence R and T.txt).
- Download the test package (BYU Hallway Sequence.zip) from BYU Learning Suite that includes
  - 2401 images of the test sequence in a folder called "BYU Hallway Sequence".
  - Camera intrinsic parameters (BYU Hallway Camera Parameters.txt). Distortion coefficients are not available.

Task 1: Practice Sequence 40 points

- Use all 702 images in VO Practice Sequence for this task.
- Select your preferred OpenCV feature detection, description, and matching methods.
- Write a program to detect and match features between two frames.
- Use matched feature points and the provided camera intrinsic parameters to estimate camera pose for each frame. You can use your code from the Structure from Motion assignment for this step.
- Concatenate your camera poses to perform visual odometry. Refer to Visual Odometry lecture slides.
- Calculation of a relative scale factor from frame to frame is not required but encouraged (refer this to the VO Tutorial paper).
- A fixed scale factor of 1.0 is to be used for all frame to frame (no skipping frames) camera pose estimations of this sequence if relative scale factor is not calculated.
- The ground truth of this sequence is provided in two files. This ground truth is obtained without skipping frames for matching.
- The first one (VO Practice Sequence Ground Truth.png) is the plot of the vehicle path and distance traveled in meters. The horizontal axis represents travel in the x direction and the vertical axis represents travel in the z direction.
- The second one is a text file (VO Practice Sequence Camera Pose.txt) that has the accumulated camera pose C_k represented as R and t for each frame. This ground truth text file contains 701 lines because no skipping frames and each line has 12 numbers. As discussed in the lecture slides, the transformation between two matched frames is represented as $T_k = \begin{bmatrix} R_k & t_k \\ 0 & 1 \end{bmatrix}$, which contains 9 parameters for rotation ($R_k \in R^{3x3}$) and 3 parameters for translation ($t_k \in R^{3x1}$). These 12 numbers are arranged as R_{11} R_{12} R_{13} T_x R_{21} R_{22} R_{23} T_y R_{31} R_{32} R_{33} T_z or in a matrix form as

$$\begin{bmatrix} R_{11} & R_{12} & R_{13} & T_x \\ R_{21} & R_{22} & R_{23} & T_y \\ R_{31} & R_{32} & R_{33} & T_z \end{bmatrix}$$ without the last row of 0 0 0 1 in T_k.

- Run your program and generate a text file that contains 701 lines of your accumulated camera pose represented as *R* and *t* for each frame and arrange your rotation and translation parameters in the same order shown above for automatic grading.
- You need to "concatenate" your transformations T_k to get the camera pose for the k^{th} frame as $C_k = C_{k-1}T_k$ and plot the camera position along the entire path (x in horizontal and z in vertical). You may need to take the inverse of T_k to concatenate camera poses as $C_k = C_{k-1}\,T_k^{-1}$ if your plot is flipped both horizontally and vertically.
- Use and implement the best drift reduction method you can find in published technical papers to get the best accuracy of your VO algorithm.
- Submit your camera pose estimation plot and explanation of your algorithm (what drift reduction method you use, how many frames you skip for matching, etc.) and your observation in a PDF file.
- Submit your PDF, text file, and code in a zip file.

Task 2: Rotation and Translation between Frames 60 points

- Refer to the instruction for Task 1.
- Use all 2241 images in BYU Hallway Sequence for this task.
- Run you code for Task 1 but use the BYU Hallway Sequence images and camera intrinsic parameters.
- Use a fixed scale factor of 0.8 for this sequence for automatic grading.
- Use and implement the best drift reduction method you can find in published technical papers to get the best accuracy of your VO algorithm.
- If you decide to skip frame or frames to increase the baseline, then your text file of *R* and *t* will have fewer lines than 2240. For example, if you skip every other frame (match Frame 0 to 2 and Frame 2 to 4 and so on), then you will only have 1120 lines in your text file.
- Submit your camera pose estimation plot and the accumulated camera pose text file.
- Explain of your algorithm (what drift reduction method you used, how many frames you skipped for matching, etc.) and include your observation in a PDF file.
- Submit your PDF, text file, and code in a zip file.