Project 3

Topics:

- Particle Filter
- ICP- Iterative Closest Points
- Visual Odometry

3. **Visual odometry** (30%)

   The goal of this section is to implement a simple, monocular, visual odometry (VO) pipeline with the most essential features:
   - Initialization of 2D landmarks,
   - Extracting keypoint tracking between two frames,
   - pose estimation using established 2D↔ 2D correspondences,
   - Extracting R (rotation) and T (translation) from the essential matrix
   - Refine translation and rotations values by scale ratio

   You will use the knowledge acquired during in class, and can chose feature extraction methods (e.g KLT,SIFT,SURF,FAST).

   Technical issues:

   1. You are allowed to use prepared sub-functions for your implementations (e.g extraction R/T from the essential matrix).
   2. Dataset: Load KITTI visual odometery data:

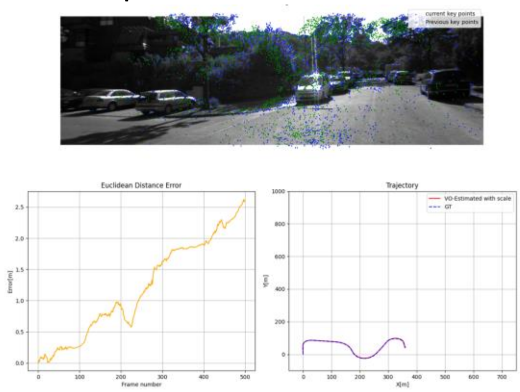
      Download odometry data set (grayscale, 22 GB)
      Download odometry data set (calibration files, 1 MB)
      Download odometry ground truth poses (4 MB)
      Download odometry development kit (1 MB)
      python tools

   3. The input (grayscale) include 22 scenarios (only 0-11 has ground truth). Select trajectory number 02 from the scenarios.
   4. Use camera calibration for intrinsic parameters.
   5. For monocular odometer use only camera 0. Estimate the motion only two pairs of images (t, t+1)

6. Calculate the scale factor as the ratio magnitude of GT (L2 norm) and estimated trajectory (magnitude GT is simulated odometer measurements).

Implement monocular visual odometry! (2D→2d).

a) Implement visual odometry on the scene. you should calibrate the algorithm to ensure a maximum Euclidian distance of 15 meters from the ground truth within the first 500 frames.
   I. Please describe your algorithm pipeline. (10%)
   II. Please add a figure showing the distance from the GT at frame 500: For example:



b) Analysis results
   a. Show an animation of the implementation, including images, features, ground truth and, error distance and estimated trajectory.(Image, features, GT and estimate trajectory) (5%)
   b. Analyze the results by comparing the estimated trajectory to the GT. Identify possible reasons for the drift observed in the results. (10%)
   c. Please suggest at least three mechanisms/ideas to improve the results (theoretically). (5%)

** some useful functions: (opencv-python package)
calcOpticalFlowPyrLK
goodFeaturesToTrack
recoverPose
SIFT_create
BFMatcher

Appendix

A. Please be honest, you may automatically lose points if you are caught copying including from the internet (code, results). The work is personal.

B. See instructions about the recorded data in the Appendix.

C. You are required to read the following paper for better understanding. Vision meets Robotics: The KITTI Dataset/ Andreas Geiger

D. The final grade is given according to the quality of your analyses, descriptions, conclusions, explanations, the form of the results (plot, graphs, animations), understandable code with comments and explanations. It is possible that the final performance and results will not be as perfect as you desired as this is real data and is part of the challenge of the autonomous driving field Feel free to suggest solutions that could improve your results if this is the case.

E. Your final package should contain the following folders:

- **Code** contains all functions + sub-functions
- **Results** stores the resulting figures ,movies, etc.
- Please save the package as zip file. The name of the should be your ID.

The report should be separated from the package, please use the attached format and read the comments therein.