Mini-project: A visual odometry pipeline!

Juan Bermeo[1] **and Tom Paye**[2]

[1]**juandiego.bermeoortiz@uzh.ch, LEGI: 20-754-951**
[2]**Address of second author**

MILESTONES

1. **Intialization/Bootstrapping Module**

   (a) Iterate over k frames from (b) to (d) until we find one that satisfies the key-point rule of thumb.

   (b) Identify features in both frames with one of the following techniques:
   - Corner detection: Harris
   - **Bonus**: FastCorners
   - **Bonus**:Sift Features
   - **Bonus**: Lift

   (c) Match the keypoints between frames to find keypoint correspondences between different views (between frames or between multiple cameras).
   - Choose a type of distance/similarity metric
   - Choose to do matching with either pairwise comparison or KLT

   (d) Estimate the point cloud of the landmarks using
   - Estimate relative pose of the next frame by calculating the fundamental matrix.
     - Use 8 point ransac (estimateFundamentalMatrix)
     - **Bonus**: 5 point ransac
     - **Bonus**: 1 point ransac + 5 point ransac
   - Triangulate the points to get the 3D point cloud.
   - Refine the estimated pose with Bundle Correction.

2. **Continous VO Module**

   (a) Create functions that iterates over all subsequent states, always using the keypoints in frame i and compare/match them to those of frame $i+1$ to estimate the pose of frame $i+1$.
   - If number of matched keypoints has not dropped below a given threshold, then use P3P Ransac to estimate pose.
   - If the number has dropped below the threshold, run bootstrapping again and get a new pointcloud of keypoints

   (b) Persist the the estimated poses at each step.

   (c) Choose error metrics and create function that calculates global and local error in pose estimation:
   - Figure out how to obtain the ground truth of poses at the local and global level
   - Create function that computes the metric for both

   (d) **Bonus:** Implement a back-end to correct global drift.
   - Bundle-Adjustment for the past k-frames
   - Pose-graph optimization

3. **Visualizing results**:

   (a) Create interface that runs the pipeline

- Display current image, plot’s current and past features and the distance between them (plots matched features).
- Plot trajectory and landmarks of the last 20 frames.
- Plot # of tracked landmarks in the last 20 frames
- Plots full trajectory
- **Bonus:** Plots trajectory over sattelite map

(b) As the script runs, stitch the resulting matched features into a video (screencast)

IMPLEMENTATION: OVERVIEW

How we expanded or deviated from outlined pipeline
List of things delivered

IMPLEMENTATION: SUCCINCT RECOUNT OF PROGRESS

Setting of milestones
Achieving milestones
Interesting problems encountered and their solution

IMPLEMENTATION: RESULTS

Qualitative evaluation of screencasts

Qualitative evaluation of our VO system

1. Interesting things that went right
2. Interesting things that went right
3. What might explain or be the solution to the things that went wrong
4. Behaviour of additional things we implemented Note that having implemented an additional feature which degrades the quality of your VO pipeline will be accepted and valued as long as i) the implemented feature is properly motivated and described, and ii) an analysis showing the effect of your additional feature is provided in the report. If the additional feature degrades the quality, provide both a screencast with additional feature enabled and additional feature disabled.