網媒所碩一　薛博文　R11944005

HW3: Visual Odometry

Procedures

1. Camera calibration
    - Capture several images manually.
    - Use openCV API to find the corners of the chessboard inside each image.

    $$cv.findChessboardCorners()$$

    - Save each pair of points, and use openCV API to help us calibrate the camera.

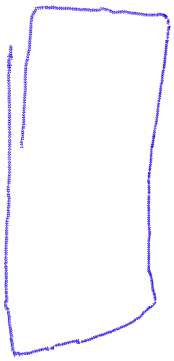
    $$cv.calibrateCamera()$$

2. Feature matching
    - Since we are making an online application, ORB is a better feature extractor. (10x faster than SIFT)
    - We find the keypoint descriptors for both images, and use BFMatcher to match the descriptors.

3. Pose from Epipolar Geometry
    - Pseudo code

```
R, t = np.eye(3), np.zeros((3, 1))

for i in range(1, len(self.frame_paths)):
    img1 = cv.imread(self.frame_paths[i-1])
    img2 = cv.imread(self.frame_paths[i])

    points1, points2 = self.get_orb_correspondeces(img1, img2)

    essential, mask = cv.findEssentialMat(points1, points2, self.K)
    points1 = mask(points1)
    points2 = mask(points2)

    R_relative, t_relative, mask, triangulated_pts = cv.recoverPose(
    essential, points1, points2, self.K, distanceThresh=800)

    triangulated_pts = triangulated_pts / triangulated_pts[-1, :]
    triangulated_pts = mask(triangulated_pts)
    points1 = mask(points1)
    points2 = mask(points2)

    scale = self.get_scale(triangulated_pts_prev, triangulated_pts,
    points2_prev, points1) if i > 1 else 1
    R = R_relative @ R
    t = scale * t_relative + R_relative @ t
    triangulated_pts_prev = triangulated_pts
    points2_prev = points2

    queue.put((R, t))
```

4. Result visualization



(a) trajectory

(b) descriptors

5. Plot the trajectory

   - Given the open3d camera object, intrinsic, extrinsic matrix, following open3d method can plot the camera pose.

     $$o3d.geometry.LineSet.create\_camera\_visualization()$$

   - With camera pose is calculate, we have $T_{world} = R^{-1}T_{cam}$, and we can draw the trajectory with

     $$o3d.geometry.LineSet()$$

6. Youtube link

   - https://youtu.be/FdZXe_120XU or click here

How to Run the code

1. Simply use 'python vo.py' to execute the code, the program will start running the visual odometry.
2. Environment

```
python==3.9.4
numpy==1.19.5
open3d==0.15.1
opencv_python==4.6.0.66
pandas==1.4.4
scipy==1.9.3
tqdm==4.36.1
matplotlib==3.6.2
```