First, I would like to thank you for proceeding with step two of your hiring process. The organization of the report is as follows: In the first section, I will describe the procedure to estimate the car's trajectory using visual data. Later, I will answer the questions that were asked alongside the task.

Visual Odometry:

In general, odometry uses sensors to estimate the change in ego-position over time. Visual odometry estimates the relative ego motion from images. Due to relative motion estimation, odometry is only precise locally and thus drifts over time due to error accumulation. The methods that aim to perform visual odometry can be characterized into ***indirect*** and ***direct*** methods. The indirect approaches extract image features and then estimate relative pose from the correspondences between the consecutive image frames. On the other hand, direct techniques skip the feature detection and extraction steps and consider the entire image for relative pose estimation. Both have their pros and cons. For instance, direct approaches are usually more accurate, while indirect methods are much faster and thus satisfy real-time requirements.

In this work, I used the indirect paradigm to solve the challenge. Hence, the first step is extracting features from image pairs and establishing the correspondences between them. The second step uses those correspondences and two-view epipolar geometry to recover the camera pose. The following points outline the steps I followed to estimate the camera's trajectory:

- Iterate over the entire image sequence; at each time, step *t*, load the images I[*t-1*] and I[*t*] from the directory and perform the following procedure:

    1. **Feature Extraction:** Extract features from the image pair. I have used ORB features here, but these could also be other features, for instance, SIFT, SURF, BRIEF, etc. The maximum number of features I extract from an image is 2500. The next step is to compute descriptors of all extracted features. I have used a built-in OpenCV function that performs both steps and returns the key points and their descriptors for each image.

    2. **Feature Matching:** The next step matches the feature descriptors of the two images. For this, I have used the FLANN library from OpenCV and ran the k-nearest neighbors algorithm to find the k-best matches for each key point. Here, I set the value of k to 2. The reason for finding

multiple matches for a single key point will become apparent in the next step.

3. **Filter Matches:** This step rejects the ambiguous matches and retains only the well-discriminated matches using the well-known *Lowe's ratio test*. It computes the ratio distance between the two nearest matches of a considered key point and declares it a good match if the value is below the specified threshold. This step eliminates all false positives and retains only the key points with good matches.

4. **Calculate the Essential Matrix E:** The essential matrix E is a 3x3 matrix that encodes epipolar geometry. We can compute this matrix using the N-correspondences between the two images obtained in step 3. The OpenCV function uses Nister's five-point algorithm to estimate E. This matrix can be further decomposed to obtain the rotation matrix ***R*** and the translation vector ***t***. Please note that the essential matrix has 5 DoF (3 for rotation ***R***, 2 for translation direction ***t***). In other words, the *global scale* (or length of the translation vector) cannot be determined using *monocular visual odometry*.

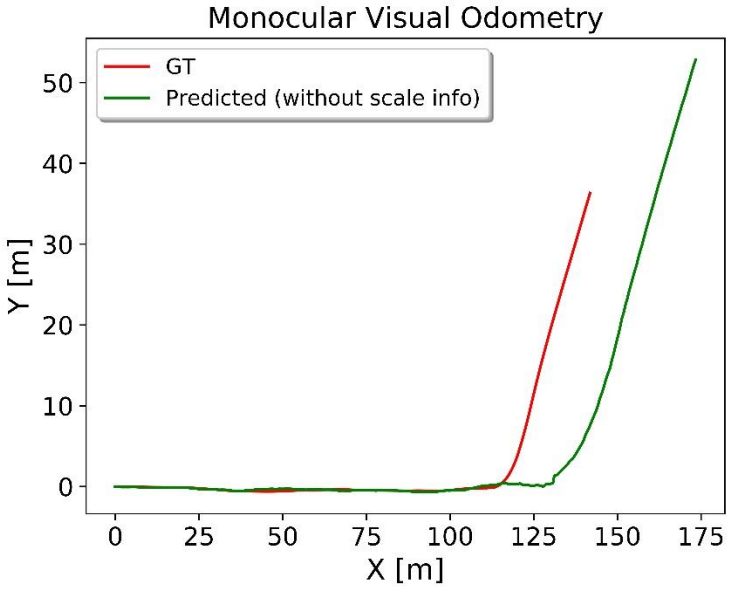
5. **Decompose E to obtain Pose (R, t):** This step decomposes E to obtain the relative pose between the two images. Again, I have used an OpenCV function for this step. It recovers the relative rotation and the translation vector (direction) using the estimated matrix E and the corresponding image points. Behind the scenes, it also runs the chirality check to return a physically plausible combination of ***R*** and ***t***.

6. Set the current image I[t] to previous image I[t-1]. Load the next image as the current image I[t] and repeat steps **1-5**.

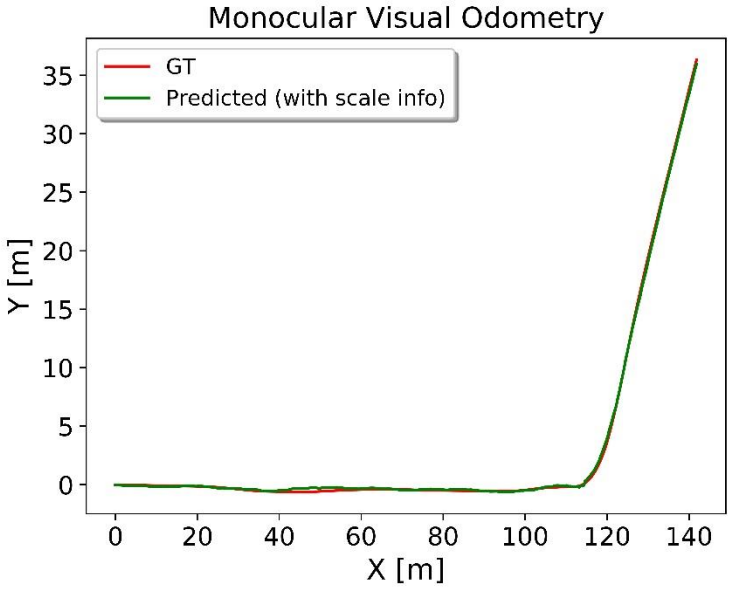
After iterating over the complete image sequence, we obtain the relative poses (the goal of VO) between all the images.

Remarks: As mentioned in the task description, not to use the ground-truth information for any computation, I have set the scale equal to 1 and plotted the ground-truth position and estimated trajectory utilizing the approach. However, we can incorporate scale information from other sources like scene knowledge or sensors. In this case, the ground-truth data is available; therefore, I utilized it to

include the absolute relative distance between the two images. In this way, we can align the predicted and ground-truth trajectory. So below, you can see the plots for both cases. Another thing, in my code, the rotation matrix is a 3x3 matrix, and I did not convert it to Euler angles (Yaw, Pitch, and Roll). I assumed that it was not necessary for this challenge. Otherwise, it is a straightforward step to transform ***R*** to standard Euler angles.



Estimated Trajectory – Without Scale Info: The used approach was able to estimate the correct shape of the trajectory.



Estimated Trajectory – With Scale Info: The estimated trajectory perfectly aligns with the ground-truth trajectory.

Question/ Answers:

1. **What is meant by the localization of a self-driving car?**
   Localization, in general, is the ability of an entity (a robot, a car) to locate itself in an environment. It precisely refers to answering the question ***where am I?*** In the context of self-driving vehicles, localization is locating a car on the road. This is a fundamental prerequisite for intelligent behavior. For instance, the car must be aware of its current location in an environment to navigate to a goal location.

2. **Please give a few applications of visual odometry.**

   a. Self-driving cars and robots.
   b. Space exploration.
   c. Sparse local mapping (but it is not the primary goal of VO, it's only a byproduct).
   d. Augmented Reality.

3. **Please provide your views on visual odometry complementing localization solutions. If you think it is not necessary, please also provide a reason for that.**
   Various sensors/techniques (for instance, wheel odometry, GPS, inertial navigation system, laser sensors etc.) can be used to localize an agent in an environment. However, each method has its weakness. ***Wheel odometry*** is a low-cost and straightforward technique to estimate the relative pose of an agent. However, it suffers from position drift due to wheel slippage and thus accumulates a significant error over time, making the method unsuitable for providing accurate localization over a longer time. The ***inertial navigation system (INS)*** uses accelerometers and gyroscopes to estimate the relative pose of a moving vehicle, but it is also prone to accumulate long-term drift errors. On the other hand, a very ***precise INS system*** is expensive, which makes it less attractive for commercial products. ***GPS*** can provide absolute position without accumulating errors, but it only works in places with a clear sky view. Moreover, its error is typically in the order of meters, which is not an attractive solution for applications requiring precise localization, such as self-driving cars. ***Differential GPS*** can provide centimeter-level accuracy, but it is also an expensive technique. ***Visual Odometry (VO)*** estimates the relative pose of an

agent by using images from a camera(s). VO is relatively more accurate and can overcome most of the drawbacks of other sensors. For instance, wheel slippage does not affect it. In contrast to GPS, VO can work in GPS-denied environments. Moreover, the technique can be implemented with a consumer-graded camera compared to expensive INS and differential GPS systems. Due to all the issues mentioned above, VO is a viable alternative to localization and can be fused with other sensors to complement a localization solution.

4. **If you feel visual odometry is necessary, where does it helps the most in the localization of self-driving car?**
   VO has a wide range of applications in robotics and the automotive industry. In general, VO is unsuitable for providing global localization (localizing an agent given a map of the environment). However, it is good at estimating local relative motion, which is often essential for self-driving because we want to know how the car is moving locally. VO has been applied to various driver assistance systems, such as vision-based braking systems. Moreover, it is also considered a cost-effective solution compared to a highly accurate but expensive LIDAR system. Thus, VO is beneficial for many applications related to autonomous cars.

5. **Is the camera odometry enough or do we need any other measurements too for reaching an accurate localization?**
   Short Answer: No. We need to fuse multiple sensors to achieve accurate localization.
   As I mentioned, each technique has its drawbacks, and one should not rely on a single modality for highly critical applications like self-driving cars. For instance, if the vehicle drives into a tunnel, the scene becomes darker, and extracting features from images and performing visual odometry would be hard. We need to switch to another sensor for localization in such a scenario. Thus, scene illumination and lighting conditions pose a severe challenge for camera-based systems. Moreover, we cannot recover the scale of the scene using a monocular camera. Hence external source such as scene knowledge or some other sensor modality is needed to estimate the scale. To summarize, I think a good system should utilize multiple sensor modalities to reach accurate localization performance while avoiding the drawbacks of individual sensor modalities.

6. **If you have been given measurements from Lidar odometry and camera odometry. Will you use them both, or will you use either one of them? Regardless of your first answer, which one will you trust more, and Why?**
   Short Answer: Both. It relates to the reasoning of the previous answer.
   A Lidar system is generally more accurate and precise than a camera-based system since lasers are not affected by a change in illumination, shadows, etc. However, a Lidar system also has limitations when foggy or rainy weather occurs. While such conditions may also impair the camera's performance, advanced camera-based algorithms can cope with these conditions and do not become completely blind in such scenarios. So, in my opinion, combining sensors is the most effective way to advance solutions for autonomous driving.