Motor AI Technical Challenge

Introduction:

Please read this section carefully. As per your preference, you have decided to apply for the open position in the localization and sensor fusion team here at Motor AI. The challenge consists of one task which will be explained in detail below. We encourage you to submit your work even if you were unable to finish the task in the designated time or are not satisfied with the results. There are a few questions that should be answered with the challenges. You will also gain some points if you are just able to answer the questions, and have not been able to solve the problem. We just want to see how you approach the problem, the **results don't matter** to us. If you have any further doubts or difficulties, please do not hesitate to contact us.

Some important points:

- If any code is not self-developed but from a source such as StackOverflow, please include the link as a comment near the piece of code that you have copied.
- Please follow the standard coding guidelines for the respective programming language (including variable naming, code readability, understandability, project structure, code optimization, reusability, etc.).
- Prepare a concise but thorough report of your approach to each step in the task. Content to include in the report will be outlined in the task description but you can organize it as you like and add any extra information you would like.
- Create a GitHub private repository with your work and send the GitHub link as the final submission. Share the repository with username **yashmotorai**

Task - 1 LiDAR - RADAR Fusion

If you are working full-time, we ask you to just make a one-page report for this task if you don't find time to code the solution.

Radar and LiDAR both perceive the 3D environment as well as the positions of the actors within it. Unlike LiDAR, a Radar can also estimate the velocity of the perceived object. Since both sensors have their own disturbances, it becomes difficult to know the actual 3-D position if they are detecting a similar object. To create a credible understanding, we must fuse the output together and create a single trajectory of the actor.

Pre-Challenge Questions-:

1. Give some advantages and disadvantages of RADAR over LiDAR.

Task Description -:

In this task, you will track a single moving object which is measured by both Radar and Lidar. In the folder, we have given you sensor estimates from a public dataset. For simplicity, the moving object is represented by a single Lidar point, a single Radar point, and a single ground truth point at a single timestamp. A lidar point has the position of the object in the x and y direction, and a Radar point includes the position as well as the velocity in the x and y direction. Lastly, the ground truth data also contains position and velocity in the x and y directions. You can extract the data using this file. Please do not use ground truth data in any calculations. The data is provided only for the evaluation of your fusion output. You can tweak the parameters of your model, according to the evaluation.

Key points -:

1. All three Lidar, Radar, and ground truth estimates are provided in their respective bin files. When you load the bin file you will find a list of objects. There are 100 objects in each bin file. One Lidar object had three variables {timestamp, pose_x, pose_y}, one radar object has five variables {timestamp, pose_x, pose_y, vel_x, vel_y}, and one ground truth object also has five variables {timestamp, pose_x, pose_y, vel_x, vel_y}.
2. The objects in the Lidar file are synchronized sequentially with objects in Radar files, i.e the first object of Lidar corresponds to the first object of Radar. That

means you will find their timestamps are similar, as they were recorded at the same time. The same goes for the ground truth.

3. Please plot the output of the fusion using tools of your own choice like matplotlib, pandas, etc.
4. You can use a filtering method of your own choice. Just as a **hint** Bayesian filters are really famous for doing sensor fusion. We recommend you design the filter by yourself, but if you are short of time then you can directly use external libraries and solve the problem. In that case, you have to provide us a pseudo code of how the filter processes the data, the involved calculation, and how it generates the output.
5. Please code your solutions in C++ (recommended) or Python (less - recommended).
6. Lastly, we strongly suggest using the simplest filter possible. Please do not complicate it too much. We just want to know if you have an understanding of fusion techniques.

Post-Challenge Questions-:

1. Please provide the reasoning behind the filter you selected. Are there some other filters as well, which could have been selected?