Trajectory Generation Problem

Problem

Given a set of points representing cones detected by the perception, generate the middle line of those cones. The middle line found will be used as a trajectory to move the car between the cones, this implies that the trajectory does not need to be a perfect middle line but as good as it permits the car to stay in between the cones with a good threshold.

Rev. 2
Add more edge cases to the first implementation and add an asynchronous simulation of the perception. The perception will send cone data every frame on a zeroMQ topic, the simulation should be able to handle this case and keep computing the optimal trajectory in an efficient way.
To handle this second revision you should first understand the basics concepts of ZeroMQ (https://zeromq.org/), than implement some simple iterations.
We also add to the problem the fact that the start line of a circuit is delimited by 4 equal orange cones, placed in a rectangle pattern in line with the other cones.

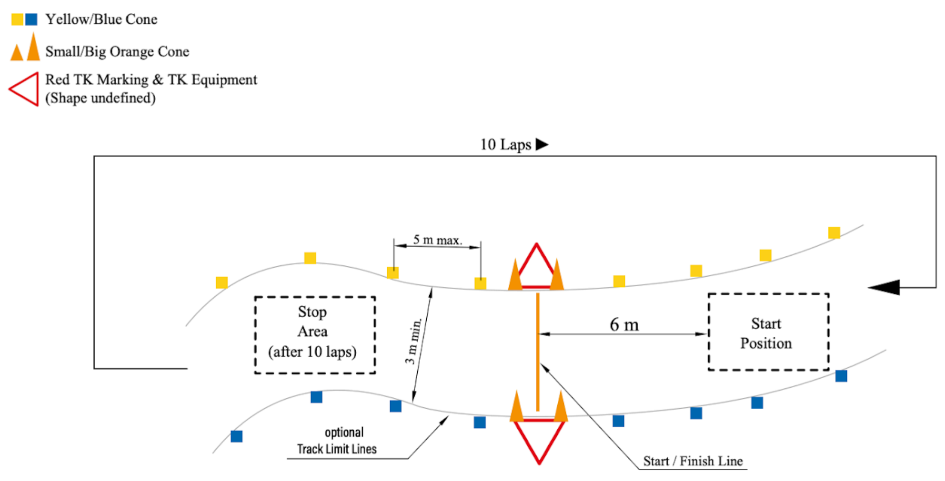

Yellow/Blue Cone
Small/Big Orange Cone
Red TK Marking & TK Equipment (Shape undefined)
10 Laps
5 m max.
Stop Area (after 10 laps)
3 m min.
6 m
Start Position
optional Track Limit Lines
Start / Finish Line

Figure 4: Trackdrive according to D8.1

Tasks

1. Analyze the possible Edge Cases in the perception data generation (some provided below).
   Analyze case of U turn
2. Create a dataset generator that can simulate all the edge cases.
   Make the generator real time, generate test points real time and send via zeroMQ
3. Start with ideal data and generate the middle line between the cones
   Add async data processing using zeroMQ topics
4. Find a solution that can generate an optimal trajectory for the problem
   I would suggest you to take a look at Clothoids

(https://github.com/ebertolazzi/Clothoids), as a tool to handle curves and data points. Those can be useful in particular when you implement dynamic/asynchronous data flow.

In the process document all the edge cases and the program produced so that everybody can run the code and interpret the results.

Code

Produce two programs:

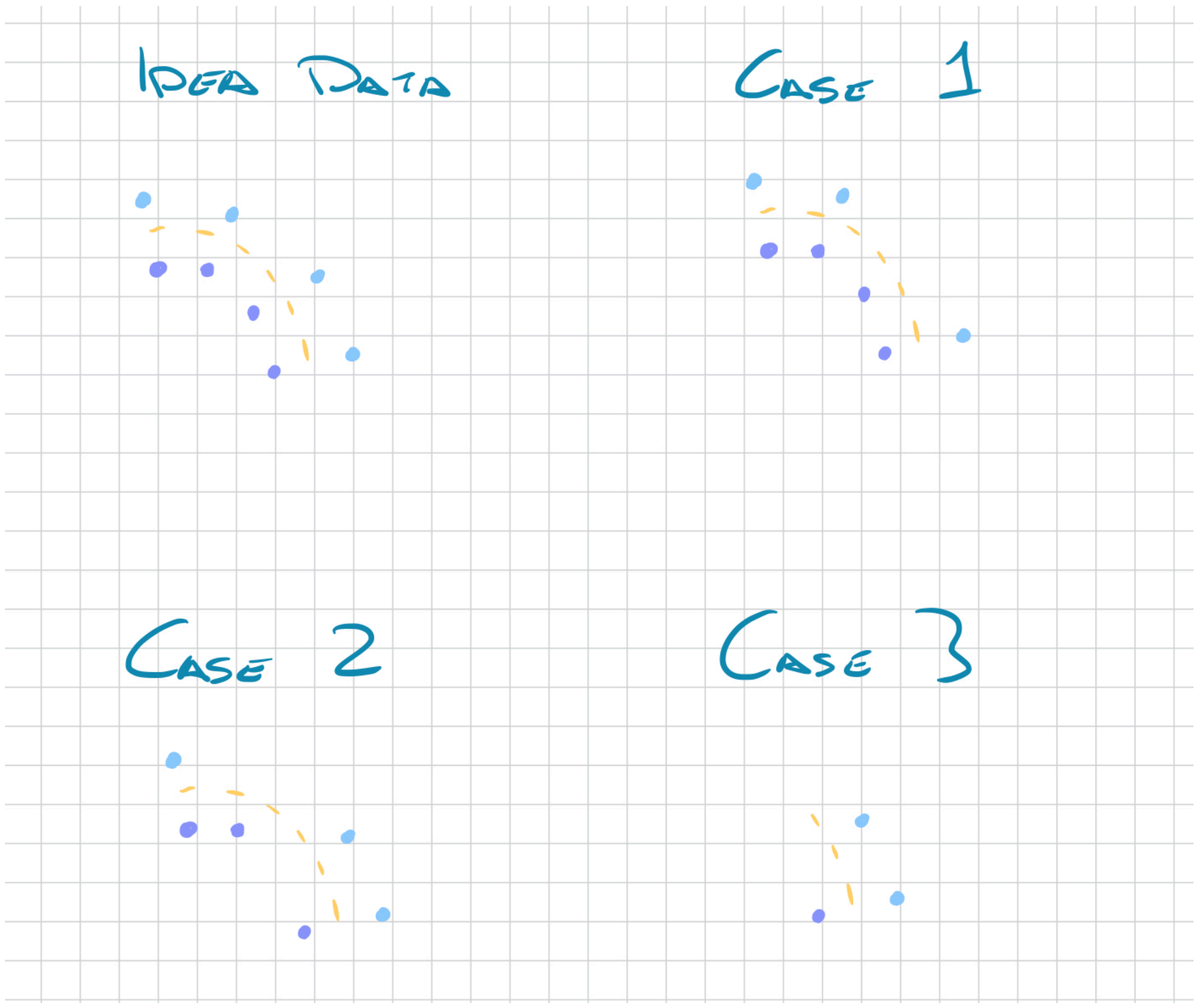
- Data generator: with some parameters in input, produce a file with the generated data. This script can be in python.
  Should be able to asynchronously generate data and send new data to problem solver. We can assume for the moment independent data between two transmission, such that the trajectories generate can be completely unrelated and disjunct.
- Problem solver: a program that takes in input a set of cones from the generator can produce a trajectory. Some tests can be made in Python but the final solution has to be made in C++.
  Continue the solution in Python to keep it simple and use fancy visualizations, we will move to C++ when the algorithm is strong enough.

Improvement Ideas

- Estimate curve radius based on cone layout and distance difference between left and right.

Examples

Edge cases


Ideal Data
Case 1
Case 2
Case 3