# EBS 289K

## Homework 4

ID: 916686000

Name: Boyan Cai

**Description:**

The whole model generates a path first and maneuver the robot to follow the path. The program has 9 major steps:

1. Set points with 10 rows, a start and a end point
2. Compute cost between points
3. Get the order when passing points based on GA algorithm
4. Generate path based on the points order
5. Following path, start from a start point
6. Find the nearest point to locate robot itself
7. Find a leash point for the pure pursuit controller as a target point
8. Target point as an input for the Ackermann system
9. Go to step 5 and check reach the end or not, if not, keep running

**Problem encountered:**

GA cannot guarantee a doable path for our Ackermann system. It is likely that the GA method fails but sometimes not. If bad paths shown, please just run my program once again. There are lots of way to do solve this problem. One way is to add points between a pair of upper and down points in the middle. Set costs between upper and down points to infinite. But set costs to zero between middle points to its headland points. In this experiments, bad paths may still be generated.

Here are figures show those two situations. In the right part of figure 1, there is a missing path. Besides, for path in the top right, its turning ratio is inapplicable. Bad paths appear sometimes because of the randomness of GA algorithms. For figure 2, the path is practical for our robot. The path can be followed and the whole area would be covered.

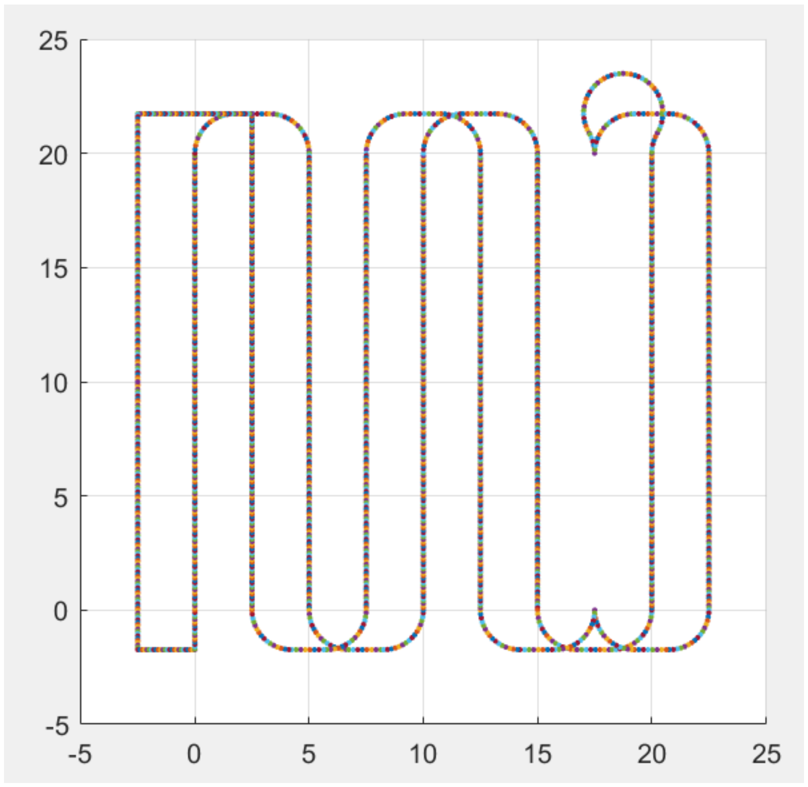

Fig. 1 Bad path generated by GA methods

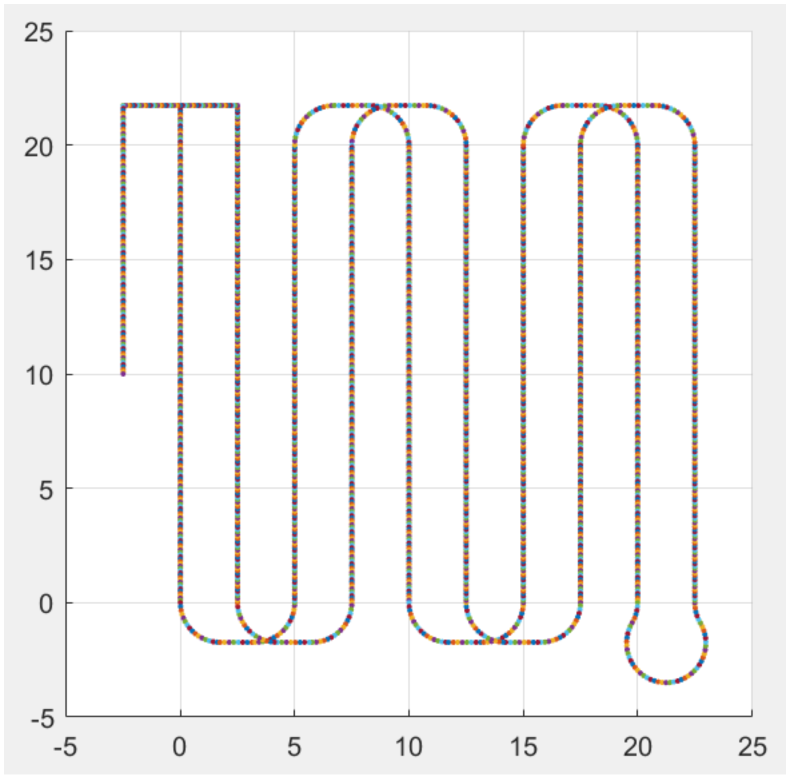

Fig. 2 Good path generated by GA methods

**GA algorithms:**

Here is the figure demonstrate the insider variables of GA.

The top left subplot shows the points need to be covered by robots.

The top right subplot shows the symmetric cost matrix used as the input of GA algorithms.

The bottom left subplot shows distances between points.

The bottom right subplot shows the solution scores of each iteration.

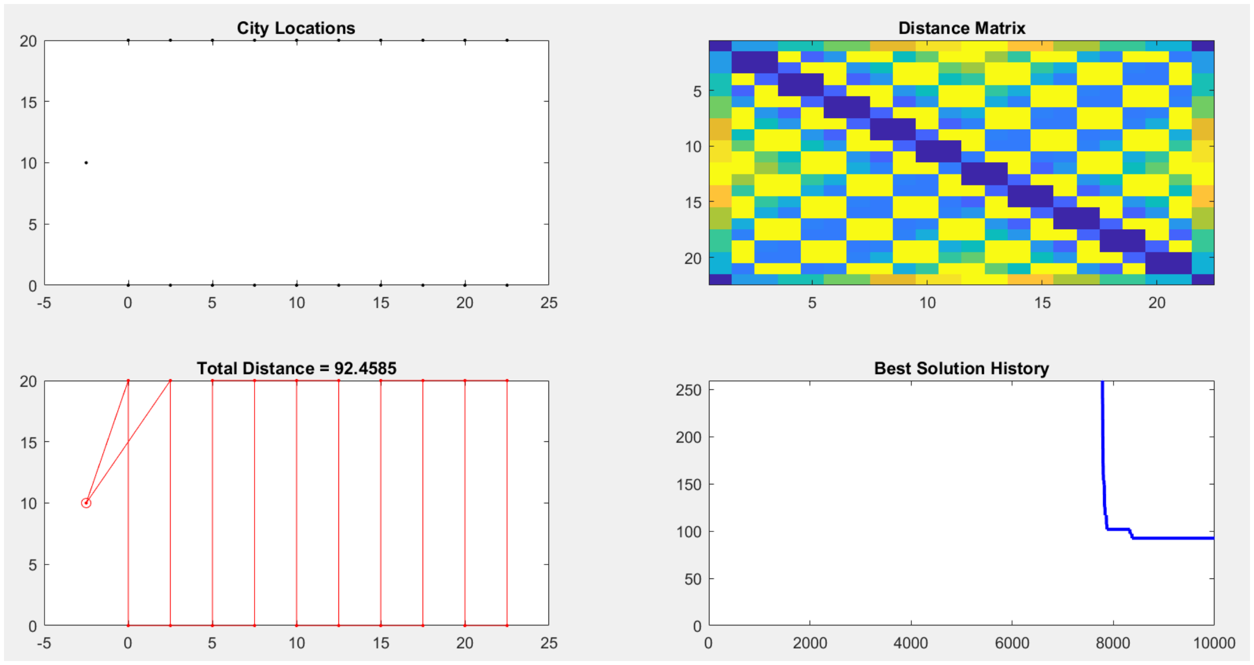

Fig. 3 Variables of GA

**Part 1: implementation of running in 10 rows and 60-degree steering angle.**

The robot can follow the path and cover the whole area almost perfectly expect when it enters the headland. The imperfection is due to the rectangle path for the start and end path. Besides, W is less than the minimum turning radius. The robot would run out a little bit. Figure 4 and 5 show the implementation.

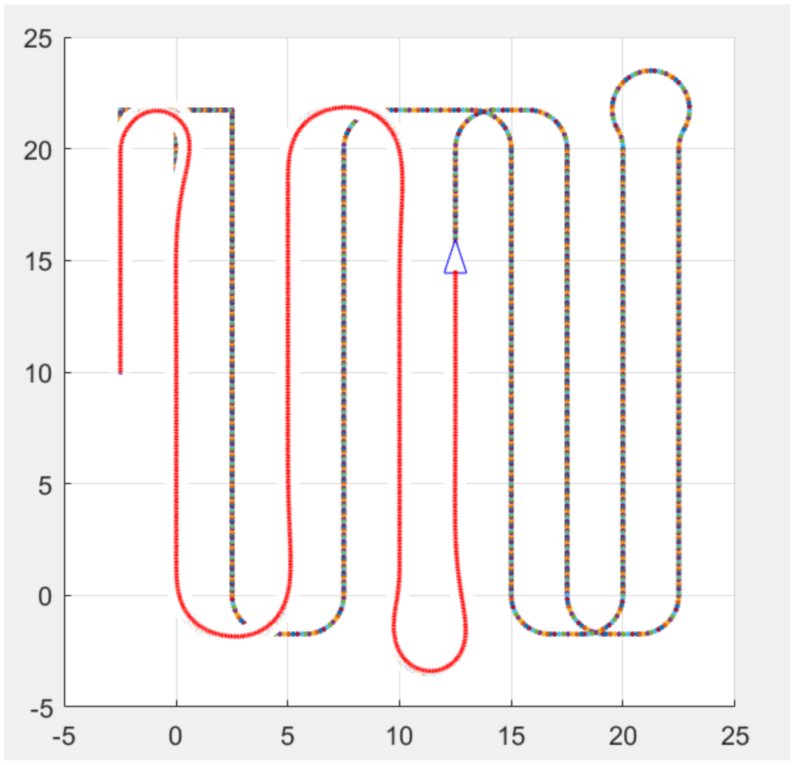

Fig. 4 steering angle 60: Running

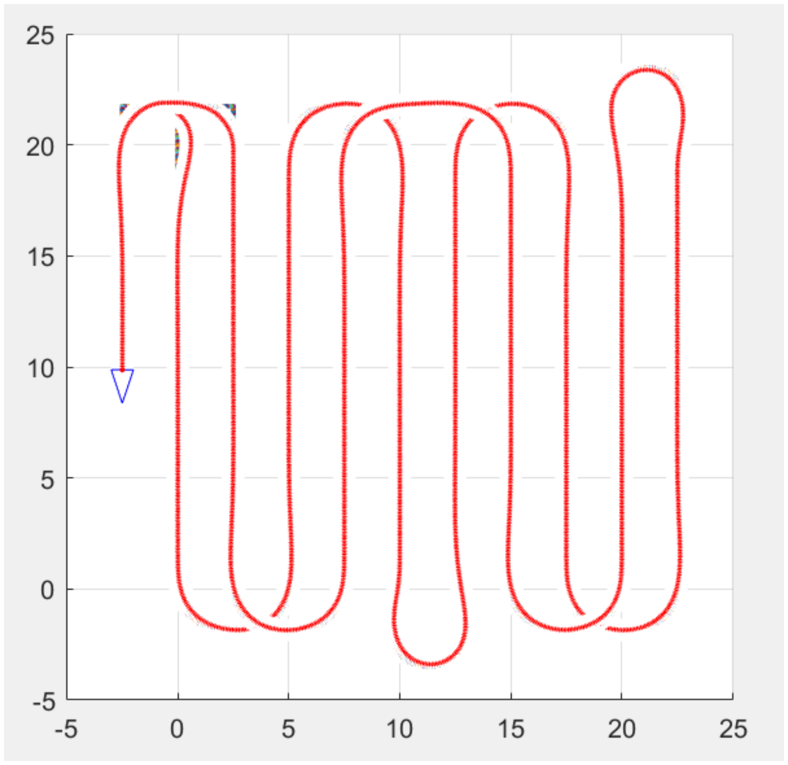

Fig. 5 steering angle 60: finished

**Part 2: Repeat for max. steering of ±30 degree if the steering angle is ±30 degree when path planning**

When following, the robot encountered a harder problem in the start and end part because of the worse steering ability. But the rest of the path can be followed perfectly.

If we set the path steering angle as ±30 degree, the path goes like below:

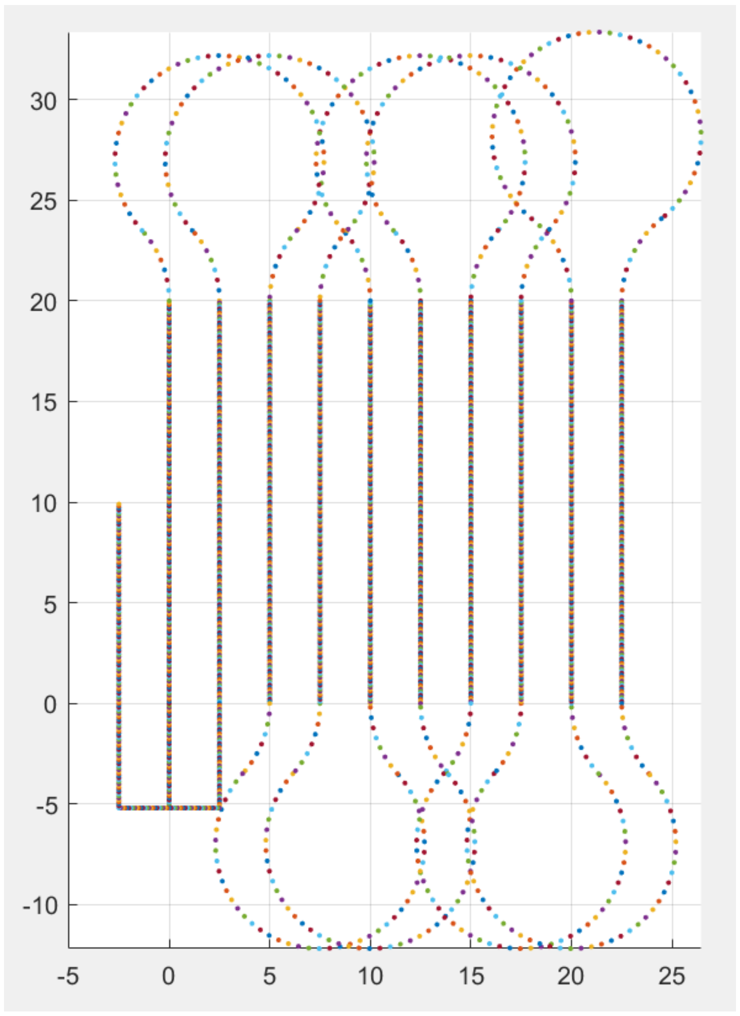

Fig. 6 path with steering angle 30

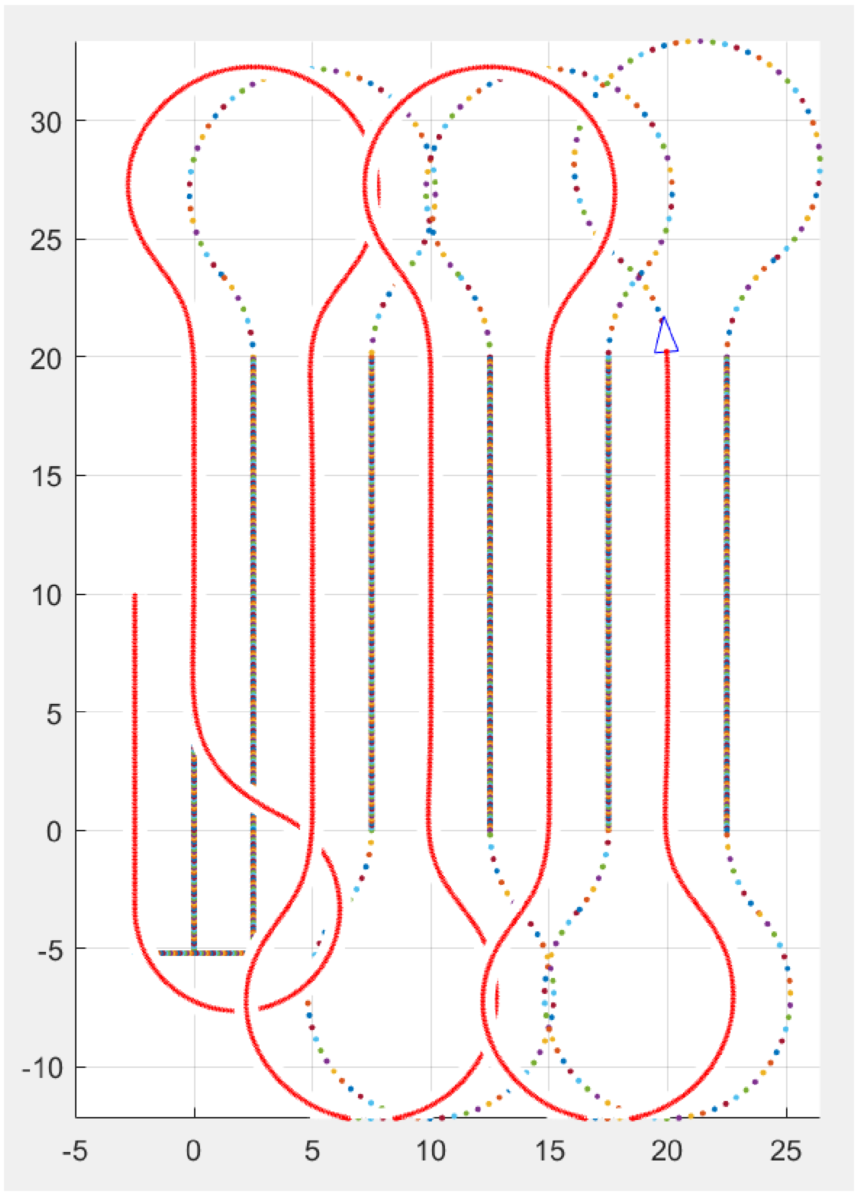

Fig. 7 steering angle 30: Running

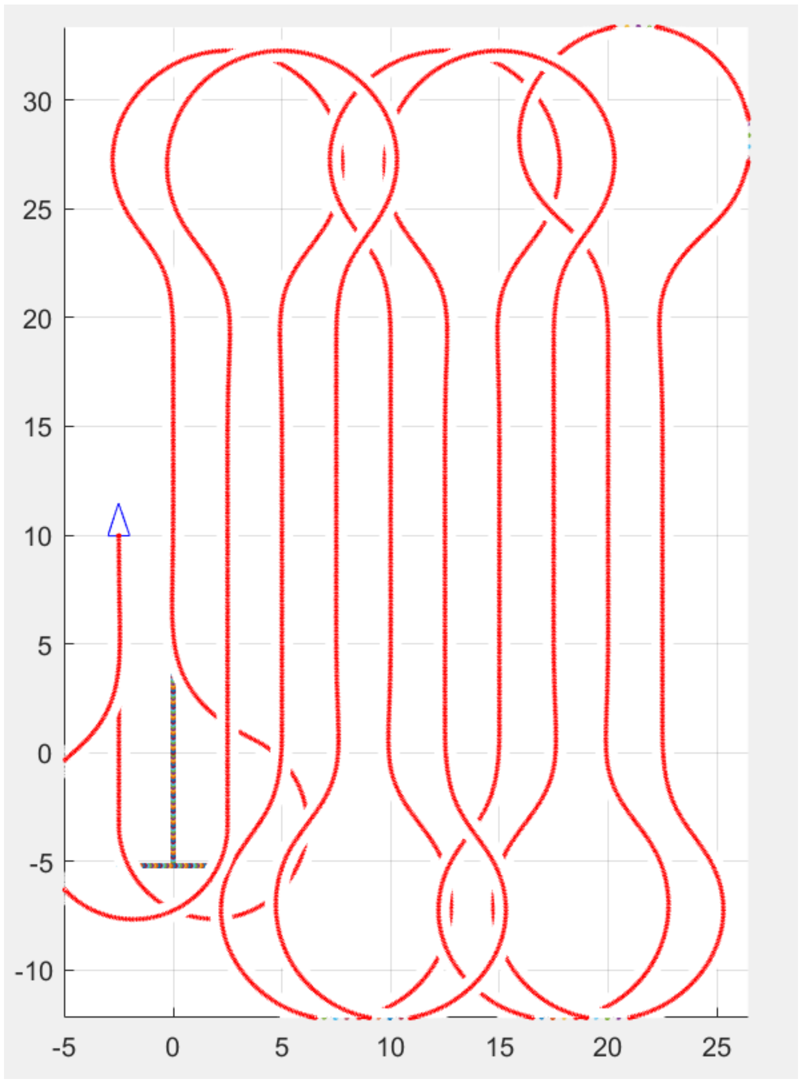

Fig. 8 steering angle 30: finished

**Part 3: Repeat for max. steering of ±30 degree if the steering angle is ±60 degree when path planning**

Figure 9 shows the path we generated with expected steering angle ±60 degree. Figure 10 shows the actual path of robot with steering angle ±30 degree. The robot goes out of the expected path a lot because of the steering radius is bigger than the radius expected when path planning.

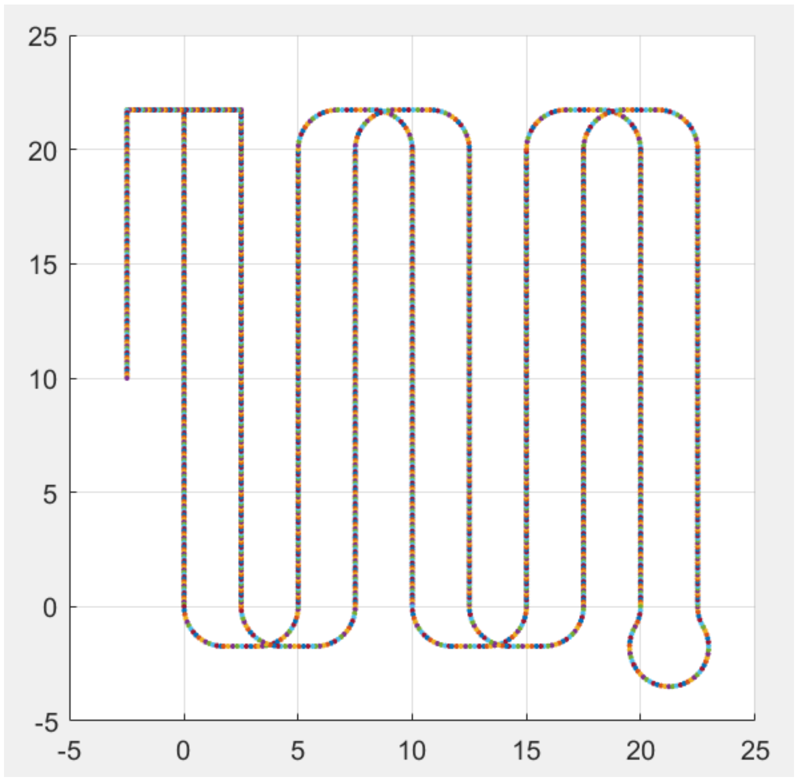

Fig. 9 the path with steering angle 60

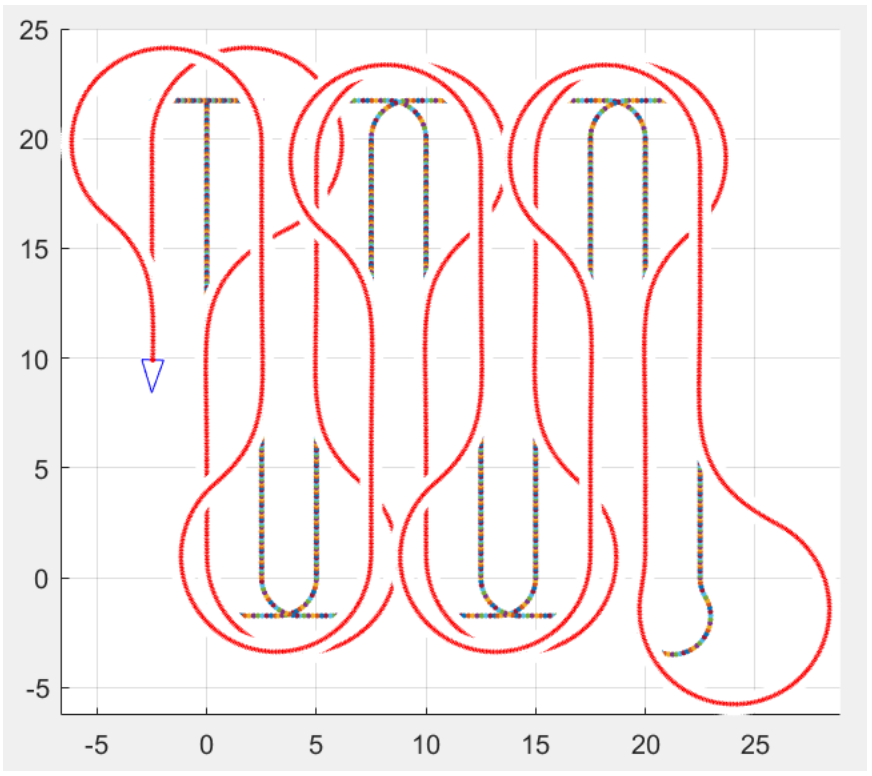

Fig. 10 following the path with steering angle 30