EBS 221 HW 3

Benoit Rouchy and Kevin Oghalai

5/18/2025

The traveling salesman problem was solved in order to find good node traversal sequences. An example of a good solution is shown below in Figure 1.

Figure 1: Node traversal solution plotted for a field.

The actual sequence being shown here is: (1 13 23 3 5 25 15 17 27 7 9 29 19 21 31 11 10 30 20 18 28 8 6 26 16 14 24 4 2 22 12 32). This makes sense due to our definition of the problem. 1 is the start node and 33 is the end node. Nodes 2-11 are on the bottom of each row, 12-21 are on the top of each row, and 22-31 are in the middle of each row. For each row traversal, it either starts with the top or bottom row, then goes to the middle of the row, then travels to the opposite side of the row it started from. Additionally, there seems to be a general pattern in the order of which rows were visited. For example, rows 2, 4, 6, 8, then 10 were traveled in that order (although they are named row 3, 5, 7, 9, and 11 due to an offset caused by the starting node). On the way back to the start node, rows 9, 7, 5, 3, then 1 were traversed in that order.

Through testing, this did not always end up being the solution that was picked, but this was the most common, and the other options were still similar. As an example, some other solutions utilized 2 or 3 omega turns, but kept pi turns for most turns. To follow this path with a vehicle, low lookahead distances tended to perform well on this test. A lookahead distance of 0.4 times the turning radius was used for the final visualization. Figure 2 shows the simulated traversal.

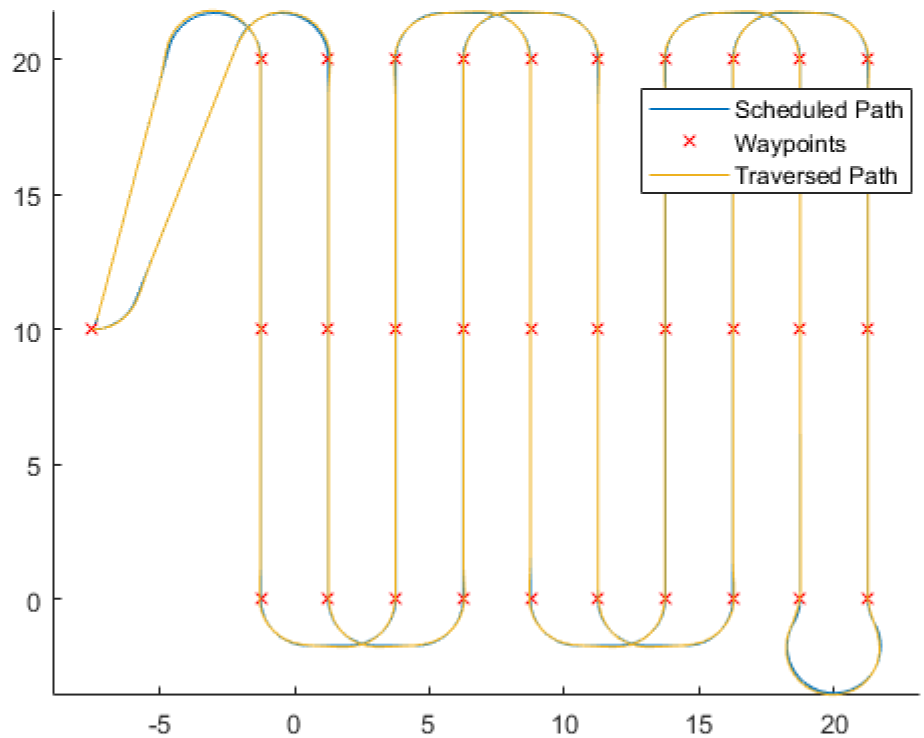

Figure 2: Field traversal according to node calculations. Both the desired and actual path are shown

This shows that in most cases, pi turns that skipped a single row ended up being more efficient than omega turns, except for a single omega turn used to go from row 10 to 9. This simulation is characterized by the simple row traversal pattern of skipping a single row. Figure 3 shows a plot of the cross track error when following a straight. The traversed path directly overlays the desired turn path, so it is difficult to see both lines.

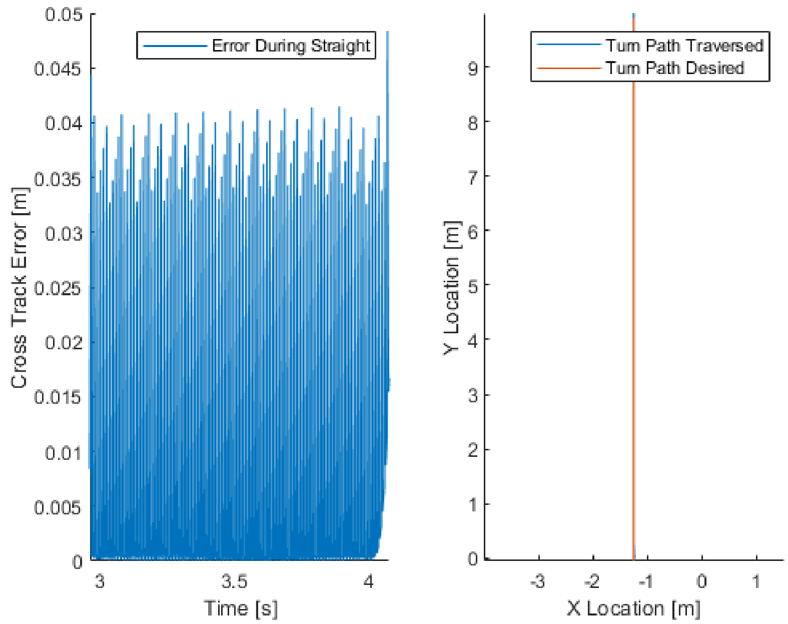

Figure 3: Cross track error for straight path.

The cross track error was low for the entire maneuver, staying under 0.05 m, with an RMS value of 0.0156 m. The cross track error is choppy because the discretization of the straight path, rather than high frequency trajectory error. The straight path required few sampling points for the vehicle to follow, thus leading to the discretization effect. The cross track error for the turning maneuver is shown in Figure 4.

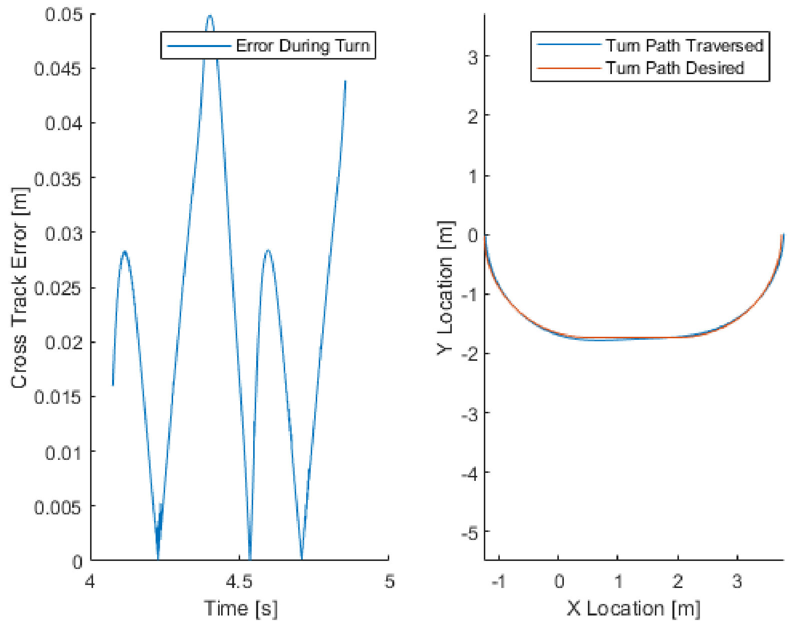

Figure 4: Cross track error for turning maneuver.

The cross track error here is still low, although the vehicle does visually deviate from the desired path. Once again, the cross track error stayed under 0.05 m, and the RMS error here was 0.0256 m.

The maximum steering angle was decreased to 30 degrees, even though the planning step thought it could perform steering angles of 60 degrees, resulting in Figure 5.

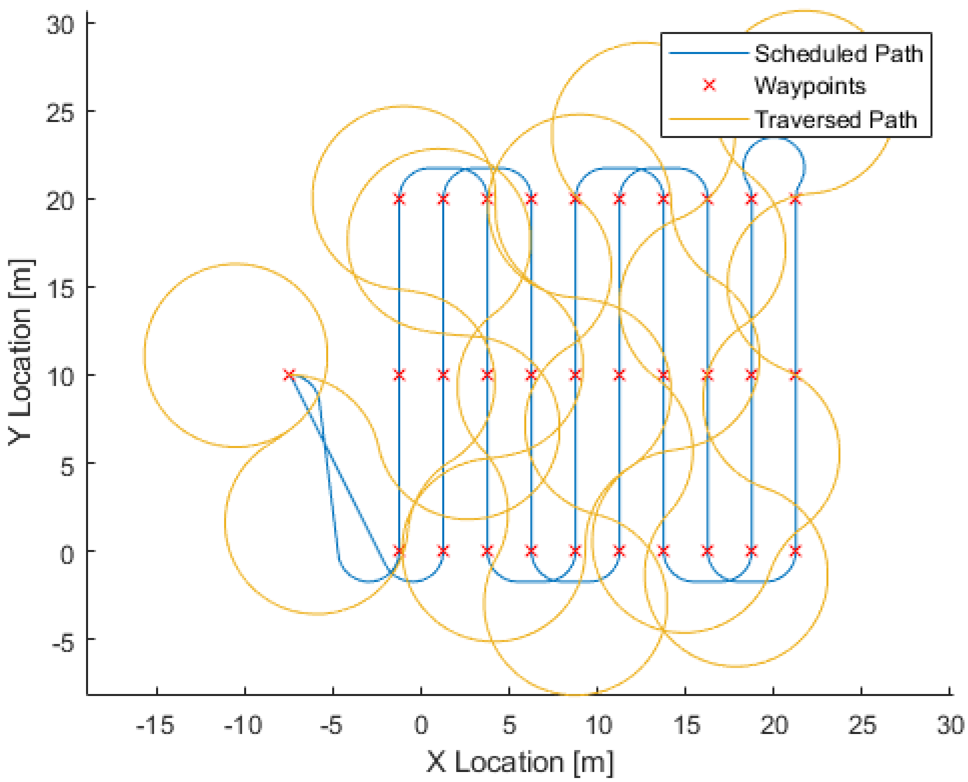

Figure 5: Planning was completed with 60 degree turns in mind, when the vehicle could only do 30 degree turns.

As might be expected, the vehicle is completely unable to follow proper trajectories, as it cannot do what it is asked to, leading to it going too far from the path, leading to chaotic movements.

The same optimization was carried out for a maximum turning angle of 30 degrees, but this time with planning that was aware of the steering limitation. In this case, the vehicle starts in a downward orientation to prevent it from cutting into the field when attempting to go from the start position to the first row. The look ahead distance was reduced to 0.2* the turning radius, as this seemed to reduce the cross track error. The following sequence shows the route: (1 4 24 14 18 28 8 3 23 13 17 27 7 11 31 21 19 29 9 5 25 15 20 30 10 6 26 16 12 22 2 32). Once again, the vehicle travels down the rows correctly(2-11 are the bottom node, 12-21 are on the top nodes, and 22-31 are the middle nodes), but in this case it skips over multiple rows, and has a less clear pattern. For example, it skips from row 3 to row 7, then back to row 2. Because the turning radius is so large, the field is not large enough for simple patterns like skipping single rows to be efficient. After running the program multiple times, it was observed that the sequences are not always the same, but the general pattern seems to be consistent. The optimized path and the path traversed by the robot are shown in Figure 6.

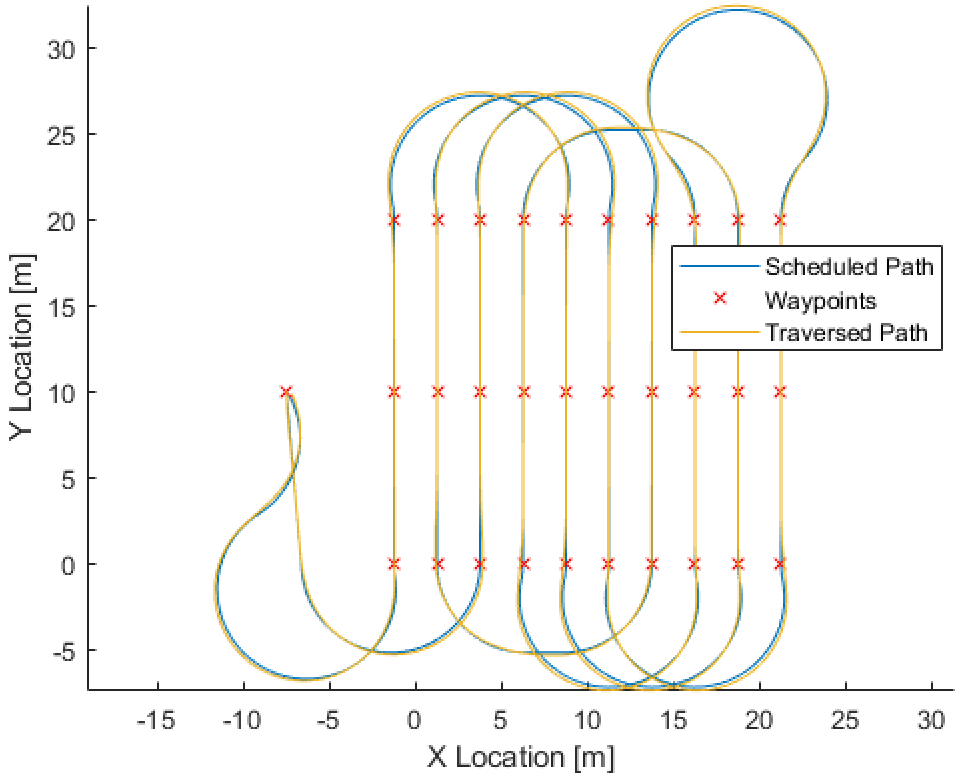

Figure 6: Field traversal with a maximum turning radius of 30 degrees.

In this case, more omega turns ended up being used, although pi turns were used for covering large gaps. These vehicle dynamics showcase the usefulness of performing optimization, rather than following simple patterns. A human driver would be unable to calculate that this is an efficient path and would likely be stuck with simple patterns, like skipping single rows, which require large, inefficient omega turns every time. By skipping 3 rows at a time, the omega turns ended up being smaller, even though the distance between the rows was actually larger. Shown in Figure 7 is the cross track error from a straight path. As before, the error is choppy, but stayed fairly low.

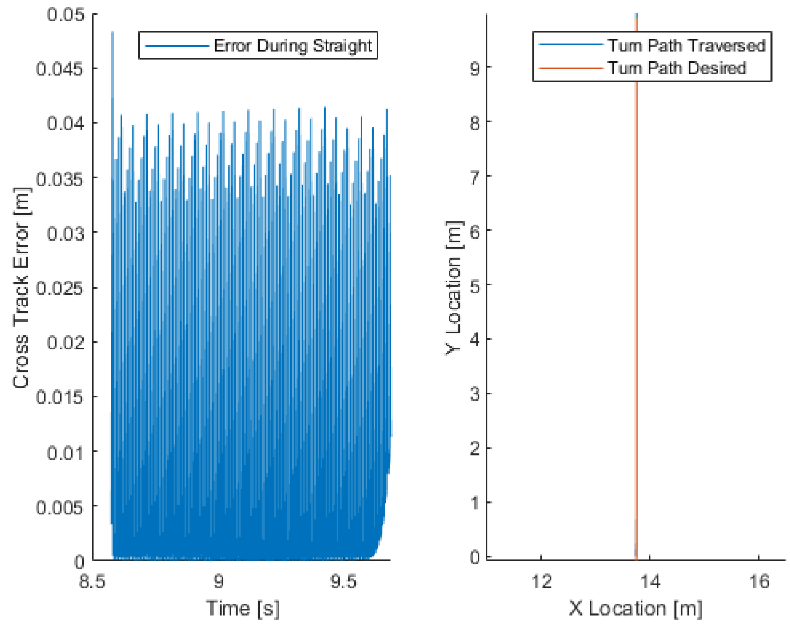

Figure 7: Cross track error for a straight path with a turning radius of 20 degrees.

The maximum cross track error ended up being less than 0.05, similar to the 60 degree turning radius. Once again, the traversed path lies almost directly under the desired path. The RMS error was 0.0156 m, roughly the same as before. The cross track error for the turn is shown in Figure 8.

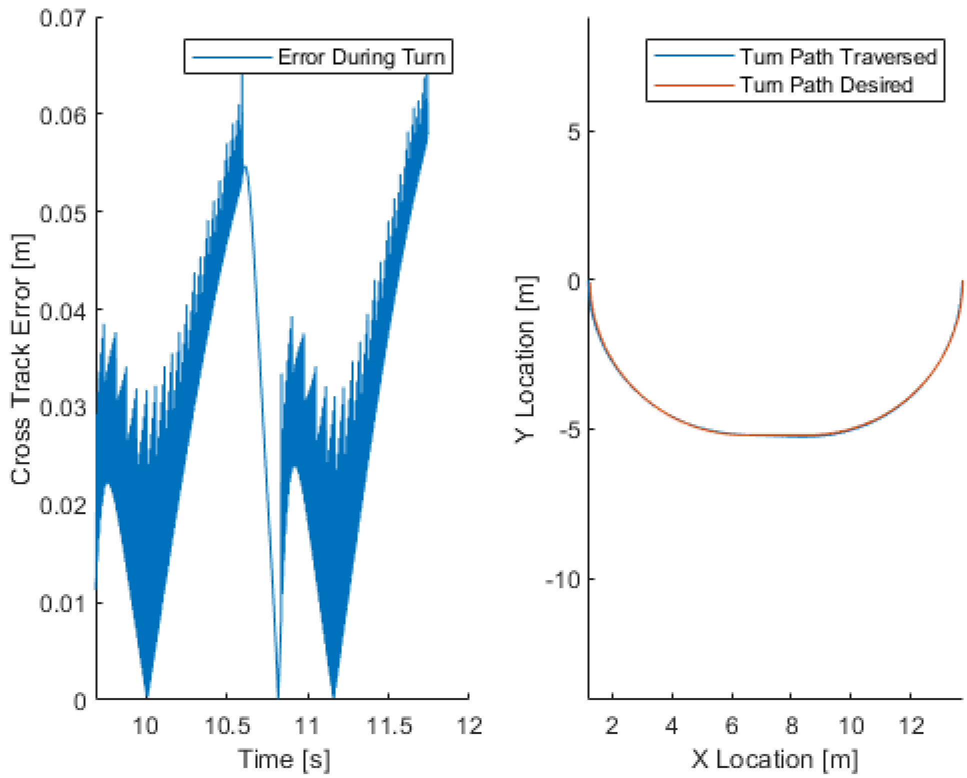

Figure 8: Cross track error for turning maneuver with a maximum turning radius of 30 degrees.

Here, the maximum error actually went up slightly to 0.07 m. This turn had larger discretization issues than the 60 degree turning radius, because the same number of points were used when interpolating the path, but the traversal distance of a turn increased by quite a bit. Even with this, the error stayed low, with an RMS value of 0.032 m.