Car Navigation using C language and OpenGL

Anubrata Nath, B7TB1706

January 28, 2019

1 Summary of the code

- The code has been written in C language.
- map.dat has been used as source for the data points.
- When compiled and executed, the terminal prompts user to select the search parameter. The user can opt to search a particular intersection using name, id or coordinates of the intersection.
- The path from initial position to final destination through the pitstop is calculated using Djikstra's Algorithm.
- An OpenGL window is created displaying the map and animating the movement of an arrow from start to destination.

2 Features and improvements

2.1 Internal Improvements

- Custom made function has been employed for searching an intersection using name parameter. The search function works like a file search where it matches a string of characters from the beginning, or middle, or the end.
- Function usde for executing Djikstra's Algorithm is highly efficient as it does not calculate distance to every point on the map, but it calculates only the shortest distance to the destination.
- Custom function has been used to calculate distance, instead of hypot() function.
- Displaying of the intersection name and id has been optimised for user comfort.
- User can select to take a pitstop in the journey.

- Program shows error code and exits if the user puts wrong input.
- User can select the cruising speed for convenience.

2.2 Features added to OpenGL part

- Use of different colors in plotting the map to enable the user to differentiate between the intersections and roads.
- Characters can be printed out in the map using fonts function.
- The markers used for initial position and final destination has been distinguished from the other map points.
- The marker used to denote the car is an arrow, like we see in the popular navigation apps like Google maps.
- Different line width has been used for drawing the car marker to make it visible.
- The marker for the car does not get distorted due to zooming, panning, and rotating.
- The marker stays in first person view while the map rotates and translates, giving it a realistic look for convenient navigation.
- The path followed by the car is highlighted using different color and line width to distinguish easily from the rest of the roads.
- The section of the road the car is currently in, is highlighted in different color for better visibility.
- The names of intersections along the path are shown.
- The immediate next intersection is highligted in different color for better understanding of car's current location.
- Turn indicator shows which direction the car should turn next.
- Traffic lights has been simulated using srand() function. The car stops if it is red at an intersection and goes, if green.
- Traffic light is displayed at the intersection
- Estimated time of Arrival and Distance to turn is displayed.
- Map can be panned using 'W','A','S','D' keys
- The map can be zoomed in and out using 'Z' and 'X' keys respectively.
- The map can be rotated using 'Q' and 'E' keys respectively.
- Map can be recenterd using 'R' key.

- 'T' key enables the user to view the map in compass mode.
- Zooming in and out also changes the scales of the font for increasing visibility.
- The fonts rotate perfectly so that they can be read easily irrespective of map orientation.
- Roboto Condensed font has been used for clear visibilty and space efficiency.

3 Conclusion

This class was really fun to learning programming in C language and employ it for practical usage like car navigation. I enjoyed brainstorming and creating the custom function part the most. Creating the OpenGL features was a bit difficult for me due my limited understanding of the glfw library function syntax. Also, the glfw being so old, it was quite difficult to find resources online. Overall, this class taught me how to overcome obstacles while coding and find way around them if, they cannot be conquered.