UDACITY

Return to "C++" in the classroom

DISCUSS ON STUDENT HUB

Build an OpenStreetMap Route Planner

REVIEW

CODE REVIEW 5

HISTORY

Meets Specifications

Congratulations on passing this project, this was quite a feat! You evinced the capability to write efficient code to solve a non-trivial problem such as path planning utilizing the A* algorithm. You can definitely be very proud of yourself for this amazing work! Feel free to share this project on your personal GitHub page with a ReadMe of yours to show the world your results. If you want to dig a bit deeper into the topics covered in this chapter, you can also have a look at the following resources:

- A* Algorithm: http://theory.stanford.edu/~amitp/GameProgramming/AStarComparison.html
- IO2D: https://kazakov.life/2018/06/07/io2d-demo-maps/
- C++ foundations:
  - https://www.youtube.com/playlist?list=PLlrATfBNZ98dudnM48yfGUldqGD0S4FFb
  - https://www.youtube.com/playlist?list=PLAE85DE8440AA6B83&app=desktop

I wish you best of luck for the rest of this course and most importantly fun with the other projects to come!

If you liked this review please also feel free to leave a rating for it :)

Compiling and Testing

The project code must compile without errors using `cmake` and `make`.
Your code compiles properly, well done!

```
[ 47%] Building CXX object CMakeFiles/OSM_A_star_search.dir/src/route_planner.cpp.o
[ 52%] Linking CXX executable OSM A star search
[ 52%] Built target OSM_A_star_search
Scanning dependencies of target gtest
[ 57%] Building CXX object thirdparty/googletest/googlemock/gtest/CMakeFiles/gtest.di
r/src/gtest-all.cc.o
[ 61%] Linking CXX static library ../../../../lib/libgtest.a
[ 61%] Built target gtest
Scanning dependencies of target gtest main
[ 66%] Building CXX object thirdparty/googletest/googlemock/gtest/CMakeFiles/gtest_ma
in.dir/src/gtest_main.cc.o
[ 71%] Linking CXX static library ../../../../lib/libgtest main.a
[ 71%] Built target gtest_main
Scanning dependencies of target test
[ 76%] Building CXX object CMakeFiles/test.dir/test/utest_rp_a_star_search.cpp.o
[ 80%] Linking CXX executable test
[ 80%] Built target test
Scanning dependencies of target gmock
[ 85%] Building CXX object thirdparty/googletest/googlemock/CMakeFiles/gmock.dir/src/
gmock-all.cc.o
[ 90%] Linking CXX static library ../../../lib/libgmock.a
[ 90%] Built target gmock
Scanning dependencies of target gmock main
[ 95%] Building CXX object thirdparty/googletest/googlemock/CMakeFiles/gmock_main.dir
/src/gmock_main.cc.o
[100%] Linking CXX static library ../../../lib/libgmock main.a
[100%] Built target gmock_main
```

Code must pass tests that are built with the `./test` executable from the `build` directory of the project. See the project submission instructions for more details on how to run the tests.

All tests pass immediately, amazing!

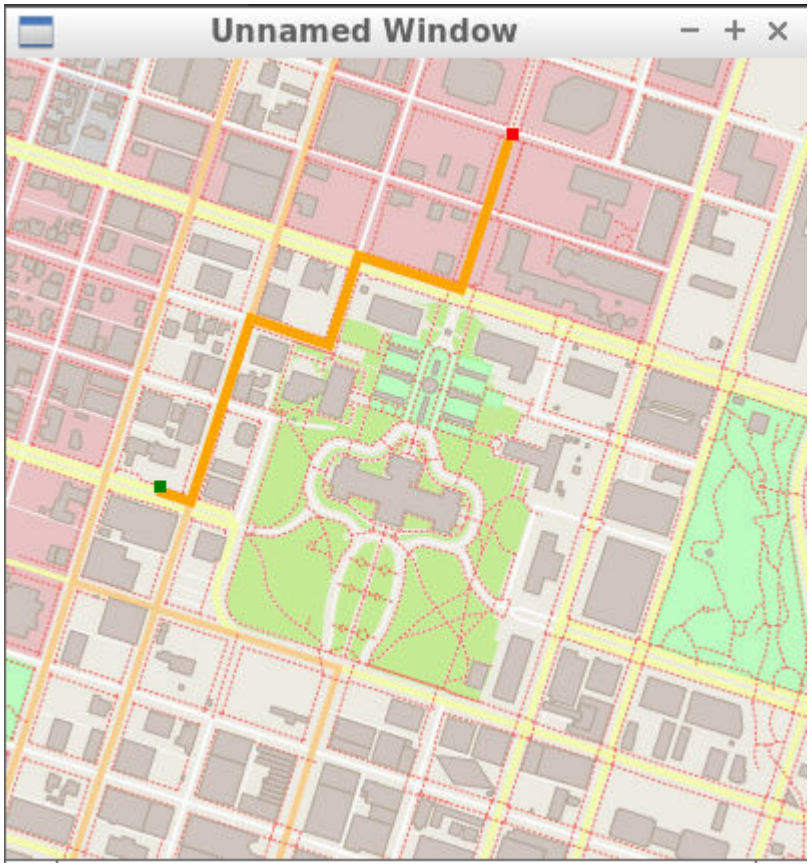
```
[==========] Running 4 tests from 1 test case.
[----------] Global test environment set-up.
[----------] 4 tests from RoutePlannerTest
[ RUN      ] RoutePlannerTest.TestCalculateHValue
[       OK ] RoutePlannerTest.TestCalculateHValue (147 ms)
[ RUN      ] RoutePlannerTest.TestAddNeighbors
[       OK ] RoutePlannerTest.TestAddNeighbors (121 ms)
[ RUN      ] RoutePlannerTest.TestConstructFinalPath
[       OK ] RoutePlannerTest.TestConstructFinalPath (141 ms)
[ RUN      ] RoutePlannerTest.TestAStarSearch
[       OK ] RoutePlannerTest.TestAStarSearch (139 ms)
[----------] 4 tests from RoutePlannerTest (548 ms total)

[----------] Global test environment tear-down
[==========] 4 tests from 1 test case ran. (548 ms total)
[  PASSED  ] 4 tests.
```

User Input

A user running the project should be able to input values between 0 and 100 for the start x, start y, end x, and end y coordinates of the search, and the project should find a path between the points.

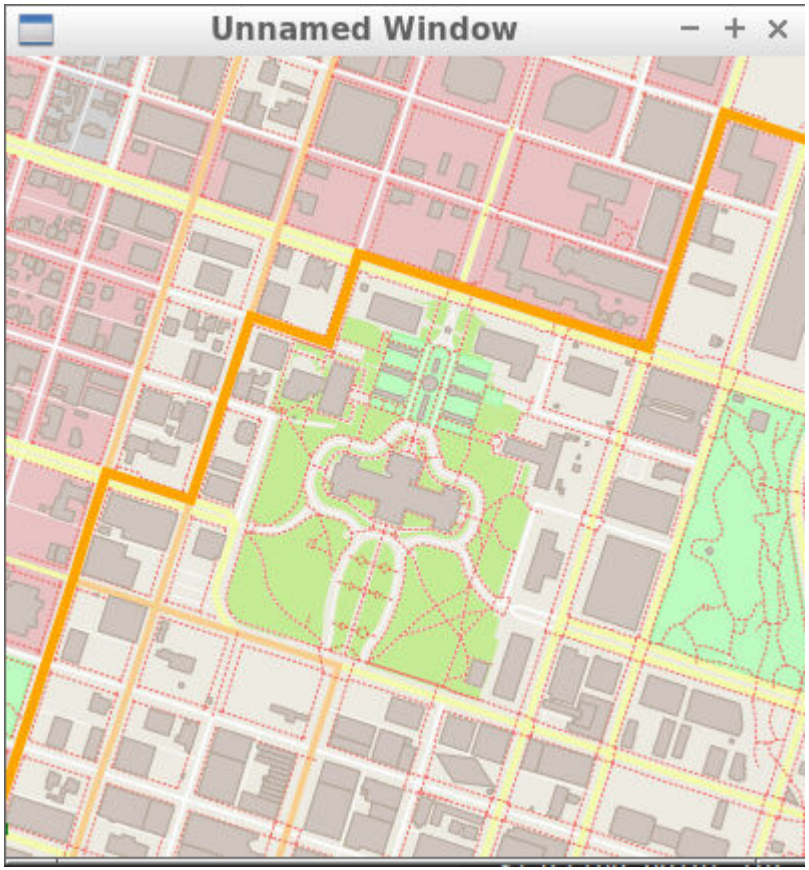
Your application works like a charm with valid user input, amazing!



The coordinate (0, 0) should roughly correspond with the lower left corner of the map, and (100, 100) with the upper right.

Note that for some inputs, the nodes might be slightly off the edges of the map, and this is fine.

Correct mapping is also preserved, awesome!



Code Efficiency

Your code does not need to sacrifice comprehension, stability, or robustness for speed. However, you should maintain good and efficient coding practices when writing your functions.

Here are some things to avoid. This is not a complete list, but there are a few examples of inefficiencies.

- Running the exact same calculation repeatedly when you can run it once, store the value and then reuse the value later.
- Loops that run too many times.
- Creating unnecessarily complex data structures when simpler structures work equivalently.
- Unnecessary control flow checks.

Your code is very readable and does not sacrifice performance in any way, way to go!

DOWNLOAD PROJECT

5 CODE REVIEW COMMENTS ›

RETURN TO PATH

Rate this review