MTRX3760 – Lab 3 Report

SID: 500490778
Tutorial: Lab 3, Thursday 3pm

Functionality:

Output:

```
Evaluating Vector approach using test points from ExampleObservations_Small.txt and
ExampleNotObserved_Small.txt

Loading the file of observations... ExampleObservations_Small.txt

PopulateForEval loaded 1848 observed test points

Loading the file of test points that don't correspond to observations...
ExampleNotObserved_Small.txt

PopulateForEval loaded 912 unobserved test points

Measuring runtime...

Total time to test all points: 17.9977 ms

Evaluating Hash table-based approach using test points from ExampleObservations_Small.txt and
ExampleNotObserved_Small.txt

Loading the file of observations... ExampleObservations_Small.txt

PopulateForEval loaded 924 observed test points

Loading the file of test points that don't correspond to observations...
ExampleNotObserved_Small.txt

PopulateForEval loaded 912 unobserved test points

Measuring runtime...

Total time to test all points: 0.142834 ms
```

Understanding:

- The hash table approach is considerably faster than the 2D vector data structure, taking 0.14ms compared to 18.00ms. It is faster as hash tables use a hash function that takes 1-n searches to find a random key, depending on the number of collisions which in this case is 0 as the number of buckets was chosen to be greater then the number of keys. While an unsorted 2D vector structure will look through an average of half the values to find the key (n searchs), or all of them if the key has no match

- The vector table uses 8 bytes of memory for every location (two ints), so approximately 16kB. The hash table uses 16 bytes for every location (two doubles, <key, hash>) so uses approximately 32kB, however it would have been possible to store values as two ints with more manipulation.
- Vector table would use approx 8Mb. Hash table approx 16Mb. However vector table random search increases linearly to size while hash table increases logarimically.

Code Quality: Git History

b78f508 (HEAD -> master, origin/master, origin/HEAD) formated and added header comments

742e0fc finished functions for hash, tests are all positive :)

420d5b9 finished hash approach, implementing into addOcc... functinos

028009f working on hash table key creation

01d6f78 temporary fix on include error, finished vector class methods

563a2d6 include path issues suck

915667d created hash files, creating hash class

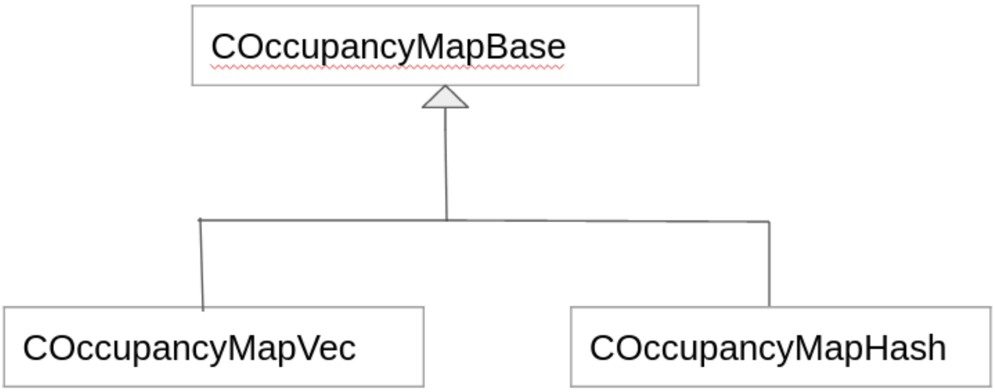
9406978 finished virtual functions, error with pure virtual

e9edaa7 writing functions for virtual methods given

e4f571b added initial files from canvas

ae68e6a Initial commit

Design: UML Diagram


COccupancyMapBase
COccupancyMapVec
COccupancyMapHash