Zach Forster

CS 335

A-Star

Although this was technically a short assignment, it was probably the most difficult short assignment that we have received thus far. At first, I wondered why we weren't given any pseudocode for the algorithm, as we usually are. It wasn't until a few minutes later that I remembered that the algorithm was nothing more than the best-first search backtracking framework that we had already written. Once I realized this, setting everything up became quite simple. I had to make minimal changes to the code that you had given us (adding some getters to the TerrainMap class for the height and the width was about all I did).

The only part of the assignment which did not go so smoothly for me (hence the first sentence of this paper), was avoiding the squares which had already been checked. Because of the out-of-order nature of a best-first search, I found it very difficult to make sure that every child was fully up to date on which squares had already been checked. Sometimes, a child would be created and added to the queue, but wouldn't be pulled out of the queue until many other squares around it which had previously been empty had already been checked. This persisted even after I had switched to one shared array to mark the positions which had already been checked. I realized that the list of viable children (to be pulled from with getNextChild()) had to be created/updated each time getNextChild() was called. Therefore, I removed my isFeasible() method and put that logic directly into the getNextChild() method. This was probably the only non-standard thing I did. Everything else was a repeat of the graph coloring assignment.