CMPT310 - Assignment 1 Report

1) By adjusting the turn cost, the behavior of some search methods does in fact change. The exceptions to this are depth-first and breadth-first search, as neither is affected by the turn cost. This is due to the turn cost being added to the path cost, and both depth-first and breadth-first are uninformed searches that don't make use of the path cost or a heuristic in their decision-making, so they will function essentially the same no matter the turn cost.

   Uniform Cost search on the other hand uses the path cost to determine its path and is an uninformed search. This means that it will choose its next node to be one of a lower path cost, and since the turn cost is calculated in terms of the direction the agent is facing (it's previous move) and the direction it will move, this causes the UCS to favor nodes in the same direction as the agent is facing since they will have a lower cost than the ones that require turns. This results in the search to continue deep down the path the agent is facing, almost replicating depth-first search in behavior.

   Greedy search, as an informed search, uses a heuristic in its decision-making as $f(n) = h(n)$, in our case the Manhattan distance. Since the Manhattan distance only counts the absolute distance from a point to the goal, its behavior is not altered by the modification to the turn value. The turn value affects the path cost, but changing it will cause no change in behavior as the Manhattan distance is independent of the turn cost, so similar to depth-first and breadth-first searches, there will be no changes to the behavior of the search.

   A* star search will see a slight change, though not as drastic as the Uniform cost search. Since A* search is an informed search algorithm that makes use of both a heuristic and the path cost, in $f(n) = g(n) + h(n)$, or the sum of the path cost and the Manhattan distance, it will be affected by altering the turn cost multiplier as it will directly affect the value of $g(n)$. This will again cause nodes that require a lot of turns to reach to have higher $f(n)$ values, and thus will be placed lower on the priority queues, leading to the A* search favoring searching in a straight direction first. That being said, it will still return the optimal path within these conditions, but in its behavior of searching, will prioritize straighter routes as opposed to turn filled routes.

2) To achieve the condition of vertical movement costing twice that of horizontal movement, my implementation would be done all within the path_cost function defined in the VaccuumPlanning class in the xy_vaccuum_search.py. To achieve this, I would take a similar approach to calculating the rotational cost. When calculating the path_cost, I would check what the direction the vacuum is about to move is, through the action variable, and if the action was to move either UP or DOWN, then the path cost would be 2 points instead of 1. This would be added before and outside of the conditional check for the turn cost. There is also a heuristic used, ie. the Manhattan distance of a node, which could be affected. Since it now costs double to move in the vertical directions, I would alter the findManhattanDist function by doubling the y-value difference. To clarify, I am not sure if this would be required within the criteria of the assignment, but for the sake of this report, I will continue with the assumption that the Manhattan distance will also be affected by the double vertical

cost. Thus, while I used abs(x2 - x1) + abs(y2 - y1) to find the Manhattan distances, I would likely change this by making it abs(x2 - x1) + 2 * abs(y2 - y1), doubling the cost of the vertical distance.

Depth-first search and breadth-first search will see no operational/ behavioral changes due to this modification, as they did not use the path cost nor a heuristic in their decision-making in the first place, but they will still accumulate twice the path cost when moving in a vertical direction. Uniform Cost search will be affected since it uses our best first search implementation, as the f value stored in the node is calculated using our path cost, this will lead to a change in behavior. With this change, even if a node is closer using the regular path cost, it may have a lower priority on the priority queue used for the frontier since if the node is farther away vertically than horizontally, the value of its distance will be proportionately higher than it would have been if the horizontal and vertical costs were 1:1. This leads to the uniform cost search to favor moving horizontally as it pops nodes with lower costs from the frontier first since it is an uninformed search method.

Greedy search will also be affected by this, since it is an informed search, it uses the heuristic in f(n) = h(n), in this case, the h(n) being the Manhattan distance, to expand towards nodes who are the shortest distance from the goal. In this case, the change in behavior will be similar to UCS, as the heuristic will see nodes with a relatively greater horizontal distance as being further, even if it would otherwise not be. This will lead to the distance of goals that are positioned vertically to the vacuum to be deemed further, so when the greedy best first search is examining nodes, it will also prioritize nodes that are horizontally positioned, even if they are relatively the same distance as vertically placed nodes, possibly leading to a less optimal solution than expected.

A* search makes use of both the path cost and the heuristic, as the evaluation function used is f(n) = g(n) + h(n), being the sum of the path cost and the Manhattan distance. In this sense, since f(n) makes use of both, and both are affected by the double value of vertical movement, A* search will be the most affected, as both the path cost and the Manhattan distance will cumulatively have a much higher cost for nodes located vertically from the vacuum. Since A* is an informed search as well, it will have access to all the nodes, and will more likely tend to move towards the horizontals as this cost difference caused by the vertical multipliers will be even more influential than in greedy search, as it only made use of h(n).

The two ways I can think of implementing this is with either the Manhattan distance included or excluded from the vertical distance modification. If we choose to exclude the Manhattan distance, then Greedy will instead not be affected by this change, since it does not make use of the path cost in its decision-making, this is beneficial if we want to keep the Greedy search more accurate to its original behavior. The other method includes Manhattan distance, which is what this report assumed was the case, in which more search results are affected, but in this sense, the comparison between them would be more "consistent" as both the Manhattan distance and the path cost will be affected, however, the downside is that A* search is disproportionately affected by this change since it makes use of both the path cost and the heuristic.