Path Planning Algorithm

Goal

Based on the sensor to detect the surrounding environment, find the shortest path to the target. The path cost includes path distance and number of turns.

Algorithm

Use one of the famous shortest path algorithm: SPFA. A little bit difference is that our path planning algorithm use a more complex cost calculate function, which also takes the number of turns into account since it will cost a robot a lot of time to turn a turn.

Therefore, the cost function is below:

It’s easy to adjust the cost function by just changing the weight of different cost. For example, if the robot takes much longer time than walking straightly, then the cost function need to increase the weight of turning to make sure the shortest path found takes least turn.

Then, use SPFA to find a path which minimize the cost function defined above. The time complexity is O(size(position in the graph)).

Robot Simulation

After writing the code, we need to test our algorithm by simulate robot in program. Here is the way how we test our robot.

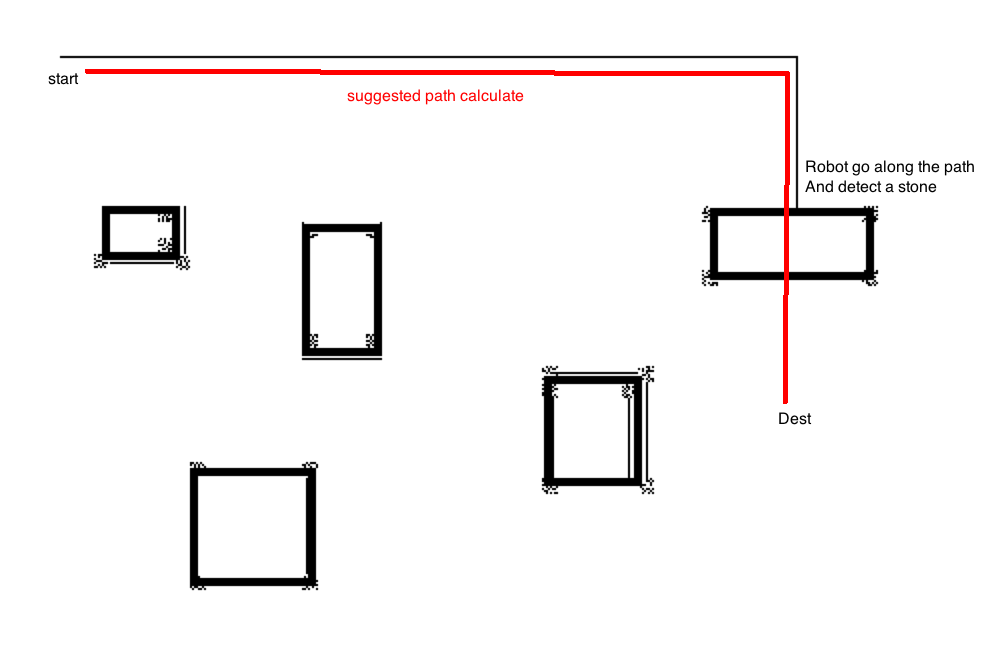
Assume the robot have two graphs in memory, one is the real graph which already contains some stones on it, another one is the detect graph that robot currently detect by using sensor, which is empty at first. Below is the testing flow:

Step1: call the path planning algorithm to get a path suggestion for the robot. Here we use the detect graph, not the real graph. Because we assume that there is no stone at first.

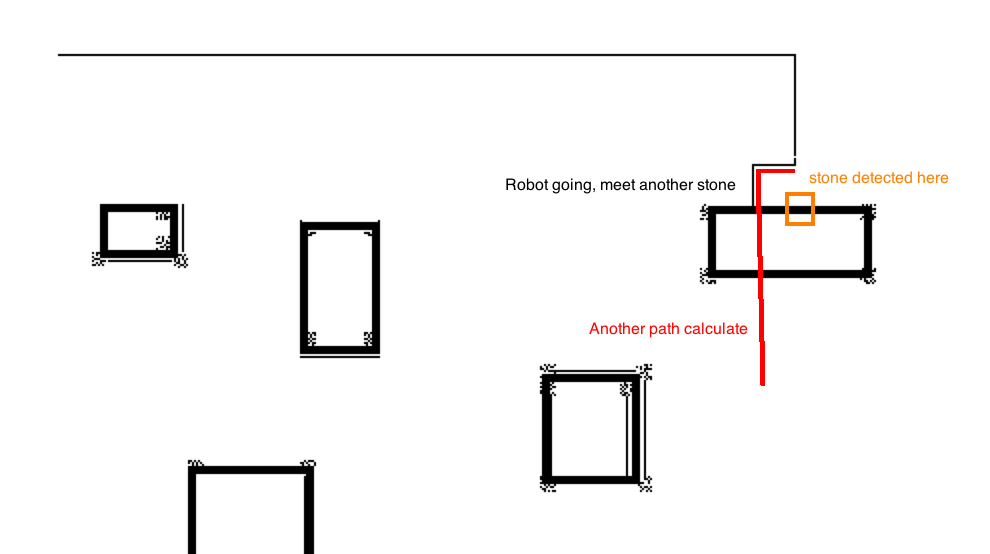
Step2: go along the suggested path until a stone is detected(we could know the stone in the real graph). If robot meet a stone, a stone flag will be put in the detect graph. Then robot will go back a few steps and go to Step1 again.

Simulation screenshot:

First step



Second step



third step

