HW 2

1st Chengkai Wu
Xidian University
Xi'an, China
ckwu1201@163.com

***Abstract*—In this report, I compare the operation efficiency of A* algorithm with different heuristic functions, with and without Tie Breaker. It turns out that**

***Index Terms*—A* algorithm; Heuristic Function; Tie Breaker**

I. A* Algorithm

Algorithm 1: A*

Data: Grid Map $\mathbf{M}$, Openlist $\mathbf{O}$, Closedlist $\mathbf{C}$
Input: Start $\mathbf{S}$, Goal $\mathbf{G}$
Output: Path $\mathbf{P}$

Initialize:$\mathbf{O} \leftarrow \varnothing, \mathbf{C} \leftarrow \varnothing$
$\mathbf{O}.insert(f(\mathbf{S}), \mathbf{S})$
while $\mathbf{O} \neq \varnothing$ **do**
  $N_{current} \leftarrow *\mathbf{O}.begin()$
  $\mathbf{O}.erase(f(N_{current}), N_{current})$
  $\mathbf{C}.insert(f(N_{current}), N_{current})$
  if $N_{current} == \mathbf{G}$ **then**
    break
  end
  $N \leftarrow getSucc(N_{current}, \mathbf{M})$
  for N_i *in* N **do**
    if $N_i \notin \mathbf{O} \cup \boldsymbol{C}$ **then**
      $N_i.g \leftarrow g(N_{current}, N_i)$
      $N_i.h \leftarrow h(N_i)$
      $\mathbf{O}.insert(f(N_i), N_i)$
    else
      if $N_i \in \mathbf{O}$ & $\mathbf{g(N_{current}, N_i) < N_i.g}$ **then**
        $\mathbf{O}.erase(f(N_i), N_i)$
        $N_i.g \leftarrow g(N_{current}, N_i)$
        $\mathbf{O}.insert(f(N_i), N_i)$
      end
    end
  end
end
if $N_{current} == \mathbf{G}$ **then**
  $\mathbf{P} \leftarrow \mathbf{getPath(N_{current}, S)}$
end

II. Heuristic Function

A heuristic function is an estimated distance from a node N(x,y,z) to the goal G(x,y,z), which can be added to the accumulated cost to estimate the length of the path going through N, and thus improve the efficiency of search. The following is three different types of heuristic function.

A. Manhattan

$$D_M = \sum_{i=1}^{3} |N_i - G_i| \tag{1}$$

B. Euclidean

$$D_E = \sqrt{\sum_{i=1}^{3} (N_i - G_i)^2} \tag{2}$$

C. Diagonal

$$D_D = \sqrt{\sum_{i=1}^{3} (N_i - G_i)^2} \tag{3}$$

III. Tie Breaker

IV. EXPERIMENTAL RESULTS

In order to generate the same experimental environment, I set the random seed as 1 when generating random map. What's more, I write a new node to publish a goal so that the searcher will receive the same goal. For each method, the program will run 20 times and then output the average of the results. Table I shows the results of different methods.

V. Problems I meet

In order to generate the same experimental environment, I set the random seed as 1 when generating random map. What's more, I write a new node to publish a goal so that the searcher will receive the same goal. For each method, the program will run 20 times and then output the average of the results. Table I shows the results of different methods. In order to generate the same experimental environment, I set the random seed as 1 when generating random map. What's more, I write a new node to publish a goal so that the searcher will receive the same goal. For each method, the program will run 20 times and then output the average of the results. Table I shows the results of different methods. In order to generate the same experimental environment, I set the random seed as 1 when generating random map. What's more, I write a new node to publish a goal so that the searcher will receive the same goal. For each method, the program will run 20 times and then output the average of the results. Table I shows the

TABLE I
RESULT

Method	Running Time(ms)	Length(m)	Visited Nodes
A* Manhattan	0.263700	6.726164	54
A* Euclidean	✓	×	×
A* Diagonal	1.552156	6.550500	725
A* Manhattan with tie breaker	✓	×	×
A* Euclidean with tie breaker	✓	×	×
A* Diagonal with tie breaker	✓	×	×

results of different methods. In order to generate the same experimental environment, I set the random seed as 1 when generating random map. What's more, I write a new node to publish a goal so that the searcher will receive the same goal. For each method, the program will run 20 times and then output the average of the results. Table I shows the results of different methods. In order to generate the same experimental environment, I set the random seed as 1 when generating random map. What's more, I write a new node to publish a goal so that the searcher will receive the same goal. For each method, the program will run 20 times and then output the average of the results. Table I shows the results of different methods. In order to generate the same experimental environment, I set the random seed as 1 when generating random map. What's more, I write a new node to publish a goal so that the searcher will receive the same goal. For each method, the program will run 20 times and then output the average of the results. Table I shows the results of different methods.

In order to generate the same experimental environment, I set the random seed as 1 when generating random map. What's more, I write a new node to publish a goal so that the searcher will receive the same goal. For each method, the program will run 20 times and then output the average of the results. Table I shows the results of different methods.