Artificial Intelligence Battle City

Bing Lin
109550112
t109550112.cs09@nctu.edu.tw

Che-An Lin
109550088
lincha.cs09@nycu.edu.tw

Chia-Wen Chang
109550036
a650993@gmail.com

Abstract

Battle City is a multi-directional shooter video game for the Family Computer produced and published in 1985 by Namco. It is the first game that allows 2 players to dual at the same time. The player controls a tank and the goal is to eliminate a fixed number of the enemy tanks and prevent our castle form destroyed by enemy. We want to set an ai agent to let ai play the game by itself to see whether the performance is well.But how do we implement a way? In this project, we use various route-finding algorithms to help ai decides forwarding directions more wisely.

1. Introduction

Artificial intelligence (AI) has received increasing attention in recent years and has been applied in various real-world applications such as computer vision, autonomous driving, game playing and even **military**. With the help of AI, object recognition, navigation and tracking, and the defensing systems could work automatically with no need of human supervisions, thus reducing the efforts of soldiers and the associated training costs.

In this final project, we use Battle City, a popular tank game, to simulate the military systems. Our goal is to design an AI agent to find paths to the enemies and protect the castle before the limited number of lives is exhausted. Note that this is an even more challenging task compared to the previous route finding assignment, where the start and end point is fixed. Both the enemy tank and the player one would move as time goes. How to properly find the path to dynamic targets (enemies) is an important task in Battle City.

Several existing algorithms (including BFS, DFS, UCS and A*) could be applied to choose actions of tanks, while they may not choose the best action without considering proper heuristic functions. To overcome such challenge, we analyze different heuristic functions of A* and choose the one which lead to the best performance. We conduct extensive experiments and comparisons to find the best heuristic function and confirm it is effective for Battle City.

2. Related Work

1. https://github.com/munhouiani/AI-Battle-City
This is the base code of agent, we use this agent to test the search algorithms.
2. https://github.com/galaamn/battle-city
This is the game resource of pygame BATTLE CITY.
3. AI-HW2: Route Finding

3. Methodology

We know that finding the good path is an essential issue in the Battle City. Trying to alternate the algorithm in the ai agent to determine the direction in the next step is another good challenge. Hence, using different finding path way is our current goal.

In the beginning, We try to change initial setting, such as maximum number of enemies and default position in the scenario. Then, see how our change would determine the performance. In A* algorithm, we change the heuristic function by Manhattan-distance, Euclidean-distance and Chebyshev-distance to see whether different heuristic function will effect the agent. When we decide the direction which is gotten from find path algorithm, we need to use current direction to test in bullet avoidance. After the test, we can get our final direction to be executed in the next movement.

We record the points which are visited in the process. After an act is executed, we can know how many nodes we visit. When we know the number of visited nodes, we can analyze them.

4. Experiments

In the first experiment, to compare the performance of different algorithms, we take 10 test case of playing result of same game stage. The setting of environment is map of stage 2 with 8 enemies, which consists of two of each type of enemy. In addition, if the ai's winning rate is greater than 60%, we mark it as pass most case.

	Heuristic function	Average time(second)	Pass most case
DFS	-	- (fail in most case)	✗
BFS	-	44.17	✓
UCS	-	42.54	✓
A*	Manhattan	34.96	✓
	Euclidean	36.78	✓
	Chebyshev	34.13	✓

Table 1. Result of different algorithm after 10 test.

In second experiment, we want to compare the efficient of different algorithm, so that we took three places on the map as standard reference point.(The red dots on Figure 1.) After that, we record the number of points we have reached when we find the path from the initial point to the point marked by red dot.

	Heuristic function	Top middle	Top left	Top right
DFS	-	1806	783	1489
BFS	-	763	829	879
UCS	-	829	827	879
A*	Manhattan	258	85	527
	Euclidean	273	414	630
	Chebyshev	273	425	623

Table 2. Number of point have been visited in different algorithm.

In table 1, we can find out that A* with different heuristic function do not significantly affect playing result. Besides, although DFS can find a way to destination, it has bad performance in playing this game. We think it should because of that in the agent only takes the first few steps to decide next movement, so whether the algorithm find the best path would be an important factor affecting the result.

However, in this game both UCS and BFS should be able to find the best path, but their performamce is worse than A*, we can find answer of this question in table 2. In this agent, it continually perform calculations to determine the path and direction of movement, while BFS and UCS need to visit much more point than A* to find the path. That is to say, A* can find the path in shorter time and make more precise judgments to fire,move or dodge bullets. [1]

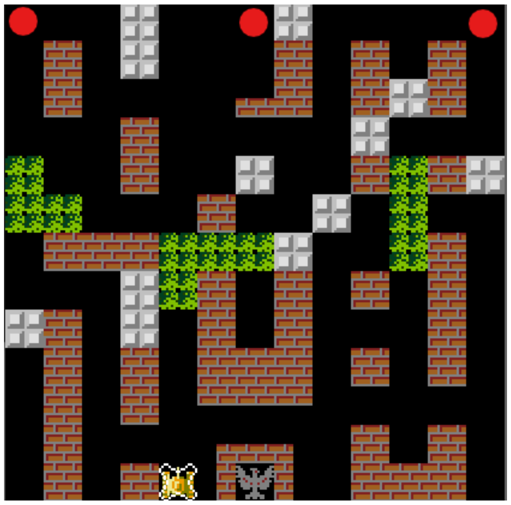


Figure 1. Location of test targets.

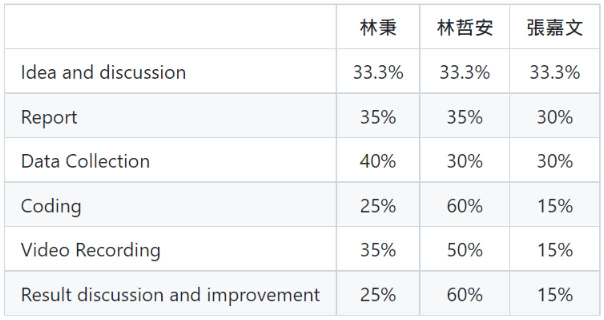
In table 2, we can find out that BFS and UCS have similar performance due to the fact that most of objects in this game environment are square, so that UCS won't visited much more point than BFS like previous homework.

Besides, the selection of heuristic function will greatly influence the efficient of A* algorithm. We consider that the main reason is because of tanks in Battle City can only move horizontally or vertically. The method of Manhattan-distance is to calculate the sum of the absolute differences of points Cartesian coordinates. In the other side, Euclidean distance and Chebyshev-distance's calculation will include the slash direction. Thus the distance of above two will have more deviation than Manhattan-distance.

In conclusion, when we use Manhattan-distance as heuristic function, We can get the closest prediction of the number of steps required by tank.

5. Github Link and Contribution

- `https://github.com/a650993/Artifitial-Intelligence-for-battle-city`



	林秉	林哲安	張嘉文
Idea and discussion	33.3%	33.3%	33.3%
Report	35%	35%	30%
Data Collection	40%	30%	30%
Coding	25%	60%	15%
Video Recording	35%	50%	15%
Result discussion and improvement	25%	60%	15%

References

[1] Shuo Xiong, Yiwen Tang, Long Zuo, and Hiroyuki Iida. Quantifying engagement of battle city with different ai strength. In *2019 8th International Congress on Advanced Applied Informatics (IIAI-AAI)*, pages 468–471, 2019. 2