IJAIT (International Journal of Applied Information Technology)

http://journals.telkomuniversity.ac.id/ijait/

Comparison Analysis of Dijkstra and A-Star Algorithms in NPC (Non-Playable Character) Movement on a Single-Player Game Case Study: Chaos Crossing Game

Dany Zaky Dhaifullah [a, *], Nelly Oktavia Adiwijaya [a], Priza Pandunata [a]

[a] *Dept. of Computer Science, Jember University, Indonesia*
192410103024@mail.unej.ac.id, nelly.oa@unej.ac.id, priza@unej.ac.id

ARTICLE INFO

Received May 31st, 2023
Revised July 13th, 2023
Accepted August 22nd, 2023
Available August 1st, 2024

Keywords
Artificial Intelligence, pathfinding, dijkstra algorithm, A-Star algorithm.

ABSTRACT

Artificial intelligence in a game plays a vital role in enhancing the player's gaming experience, especially in single-player games. NPCs are the primary means of interaction in single-player games, assisting and guiding players like interactions with other players. Chaos Crossing requires pathfinding technology for optimal NPC movement, allowing them to navigate the environment grid-based while avoiding static obstacles. The Dijkstra algorithm and the A-Star algorithm need to be compared because, based on previous research, the Dijkstra algorithm has proven effective for calculating the shortest distance to the destination point in a static environment based on a two-dimensional grid with characters moving in it, as well as the A-Star algorithm can avoid a static environment based on grid and is used to determine the shortest distance to the destination point in the character's movement. This quantitative research aims to find a solution that optimizes NPC movement by testing and comparing Dijkstra's and A-Star's algorithms in a static environment grid based on the game Chaos Crossing. The test results and comparative analysis show that the A-Star algorithm performs a faster route search with an average value of 36.37 seconds than Dijkstra's algorithm with an average matter of 20.76 seconds and utilizes memory more efficiently with an average value of 20.19 MB than Dijkstra's algorithm with a value 22.17 MB on average. However, Dijkstra's algorithm produces a slightly shorter track distance, with an average value of 42.26 units, compared to the A-Star algorithm, with an average value of 42.39 units.

Acknowledgement

We are grateful for the institutional support from the Department of Computer Science and the University of Jember.

* Corresponding author at:
Department of Computer Science, University of Jember
Jl. Kalimantan Tegalboto No.37, Sumbersari, Jember, 68121
Indonesia
E-mail address: 192410103024@mail.unej.ac.id

ORCID ID:
First Author: 0009-0000-7486-4993

https://doi.org/10.25124/ijait.v8i1.6053
Paper_reg_number IJAIT000080104 2024

1. Introduction

The game industry is growing rapidly, and one of the essential hopes of a good game is a game that can be more interesting, immersive, and innovative than other games [1]. This hope is because game technology continues to develop, one of which is artificial intelligence which allows players to compete against computers in games [2][3], so artificial intelligence in games is very influential in improving the playing experience [4], especially in single-player games [5]. A single-player game is a game that consists of only one player, so it only relies on a Non-Playable Character (NPC) to interact with other players [5].

In the case of this research is a game called Chaos Crossing, which has a top-down perspective on traffic edification. Game Chaos Crossing has a single-player game mode with a grid-based environment that contains the Main Character (MC) and game content. The problem with this game is that game content has two different types of objects: the NPC as a dynamic object and the environment as a static object. The NPC in the Chaos Crossing game aims to move from the starting point to the destination point, but this movement needs to be improved due to the static environment in the game. With a fixed environment, NPCs need pathfinding technology to move optimally by avoiding static environments to reach their destination [6][7].

Based on the description above, one way to solve the NPC problem is to determine which path to follow by applying the artificial intelligence pathfinding method [6][8][9]. The pathfinding method finds the shortest route in an environment by identifying pathways that can be passed and environmental elements that cannot [6]. Several algorithms can solve pathfinding problems with a grid-based static environment, namely the Dijkstra algorithm and the A-Star algorithm [9]–[12].

Dijkstra algorithm is a shortest path tracing algorithm that explores nodes sequentially, starting from the initial node and updating the shortest distance to neighboring nodes [12]–[15]. Dijkstra does not use heuristics and looks for the shortest path based on the actual cost from the initial node to the currently processed node [14], [16]. On the other hand, the A-Star algorithm is a variation of Dijkstra, which uses a heuristic function to estimate the remaining cost from the node being processed to the destination node [17][18]. With this heuristic function, A-Star can prioritize the vertices with the lowest estimated prices to find the shortest path in the graph with non-negative path costs [8][19][20].

Dijkstra and A-Star algorithms must be compared because, based on previous research, the Dijkstra algorithm has proven effective for calculating the shortest distance to the destination point in a two-dimensional grid-based static environment with characters moving [12]. On the other hand, the A-Star algorithm can avoid grid-based static environments and determines the shortest distance to the destination point in the character's movement [9]. Previous research has also proven that the A-Star algorithm has a higher speed in route search than Dijkstra [15]. However, it is essential to note that it is possible that the Dijkstra algorithm also has advantages over other comparison parameters, such as memory usage, which can be considered in selecting an algorithm to improve performance and compatibility when applied to a game. Therefore, a comparative analysis between the Dijkstra and A-Star algorithms is needed to determine the most optimal algorithm for NPC movement against a grid-based static environment in the Chaos Crossing game.

The Dijkstra algorithm and the A-Star algorithm have their advantages and disadvantages. Therefore, in this study, a comparative analysis will be carried out between the Dijkstra algorithm and the A-Star algorithm to obtain the optimal algorithm for applying NPC movement to a static environment in the Chaos Crossing game. This research aims to find a solution that optimizes NPC movement by testing and comparing Dijkstra and A-Star algorithms in a fixed environment grid based on the game Chaos Crossing.

2. Research Method

This research uses quantitative data, conducting a comparative analysis that aims to find the optimal algorithm for NPC movement in the Chaos Crossing game. To get the best results from the algorithm, it is necessary to test each parameter for comparison in finding the optimal algorithm for NPC movement in the Chaos Crossing game. The research process involves several stages, as shown in Figure 1.

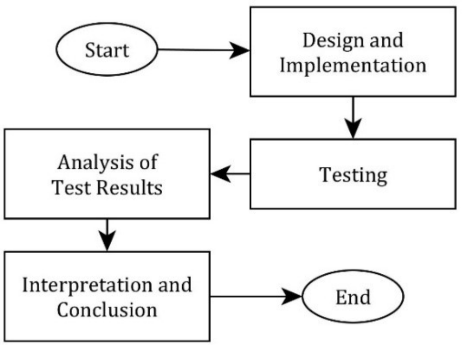

Figure 1 Research Methodology

2.1. Design and Implementation

Five different level designs were made, with a different static environment for each level design. The level design creates game obstacles, providing players with an exciting and challenging playing experience [21]. Next, three different starting and destination positions are placed on the NPC at each level design. The level designs are structured in such a way as to stay out of research case studies, adopting the same design style as in the Chaos Crossing game. Next, the Dijkstra and A-Star algorithms are implemented on the NPC against a static environment for each level design.

2.2. Testing

The testing process aims to obtain data based on comparative parameters. Testing was conducted using simulation techniques at five different level designs and three various NPC position points at each scenario. The data testing process uses the Dijkstra and A-Star algorithms implemented in the Chaos Crossing game using the Unity Engine. The types of variable data collected are listed in Table 1.

Table 1 Data Variable Types

No	Variable	Description	Data Type	Range
1	Route Finding Time	Processing time for searching the route from the starting point to the destination by the Dijkstra algorithm or A-Star algorithm in one iteration.	Float	0-1000 seconds

No	Variable	Description	Data Type	Range
2	Path Distance	In one iteration, the path distance is needed for the NPC to move from the starting point to the destination by the Dijkstra algorithm and A-Star algorithm.	Integer	0-1000 units
3	Memory Usage	The total memory usage used by Dijkstra and A-Star algorithms in one iteration.	Float	0-100 MB

The variable type of track distance uses units because the testing process is carried out using the Unity Engine. In the Unity Engine, the unit of coordinates used to represent an object's position is called a unit. This unit does not refer to a specific physical unit, such as meters, but a relative unit used in the Unity environment. In Unity Engine, data collection on the memory usage variable can be obtained using the 'System.GC.GetTotalMemory()' method. This method allows one to get the total amount of memory currently allocated by the running application. The test is carried out by implementing the Dijkstra and A-Star algorithms at five level designs and directly comparing the two algorithms at each implemented level design.

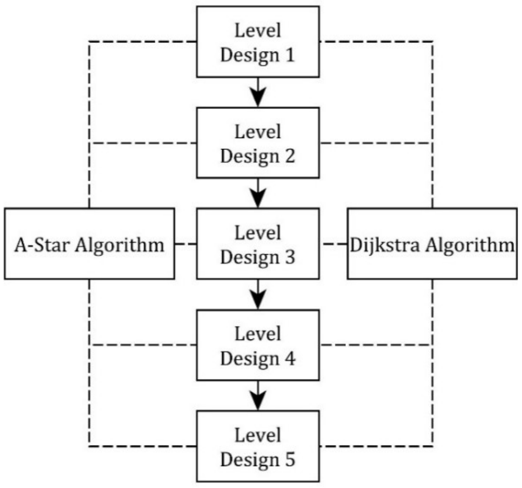

Figure 2 Test Scenario

There are five test scenarios, and each procedure is carried out by comparing the Dijkstra and A-Star algorithms implemented in three NPC movement positions to the level design. The flow of the test scenario can be seen in Figure 2. The case in the single-player game Chaos Crossing has a static environment that keeps the algorithm calculations fixed so that each design is tested only once.

2.3. Analysis of Test Results

The data analysis technique to produce a comparison between the Dijkstra algorithm and the A-Star algorithm is to determine the average result of the data collected based on the comparison parameters of each starting point to the destination at the level design using the formula in Equation 1.

$$Average = \frac{x_1 + x_2 + \ldots + x_n}{i} \qquad \text{Equation 1}$$

With the average results of the comparative parameter data for each level design, the efficiency and effectiveness of the Dijkstra and A-Star algorithms can be identified, and the best algorithm can be determined from the comparisons in each parameter based on the data collection results.

2.4. Interpretation and Conclusion

The interpretation was made to identify the advantages and disadvantages of Dijkstra and A-Star algorithms in the Chaos Crossing game. Conclusions are obtained from the results of analysis and interpretation.

3. Results and Discussion

3.1. Result of Design and Implementation

Based on the research method, the first step is to arrange five level designs and then place three starting points and destination points for each level design. The preparation results can be seen in Figure 3; the effects of placing the starting and destination points are listed in Table 3. To make it easier for visualization in testing and data analysis, each level's design is changed to a black-and-white format. The results of the level design conversion to black and white are shown in Figure 4. It is explained that there are two types of colors. The black color represents a static environment or a place the NPC cannot pass through, and the white color means where the NPC can pass.

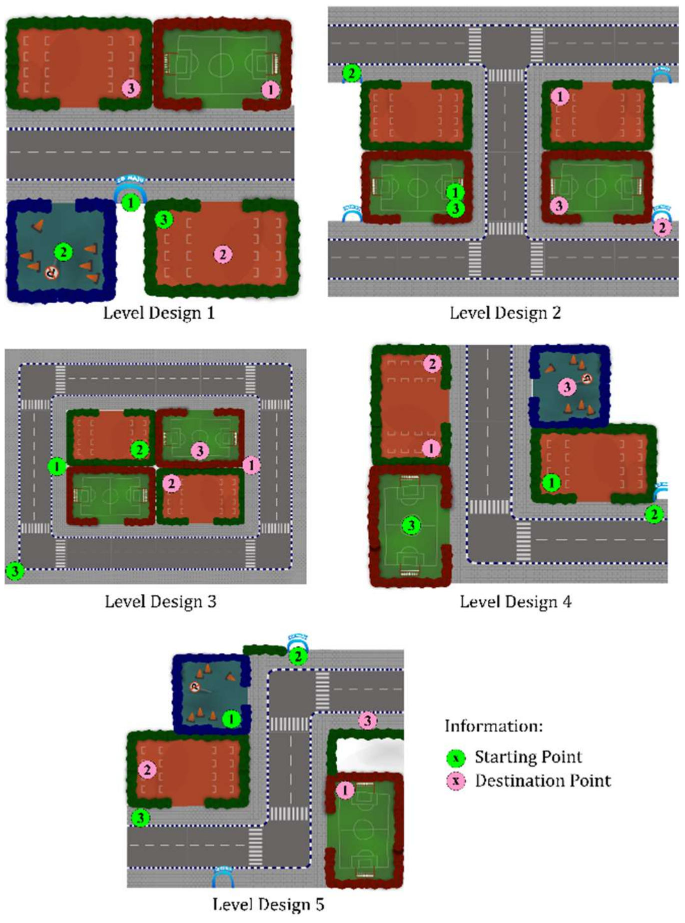

Figure 3 Results of Five Level Designs

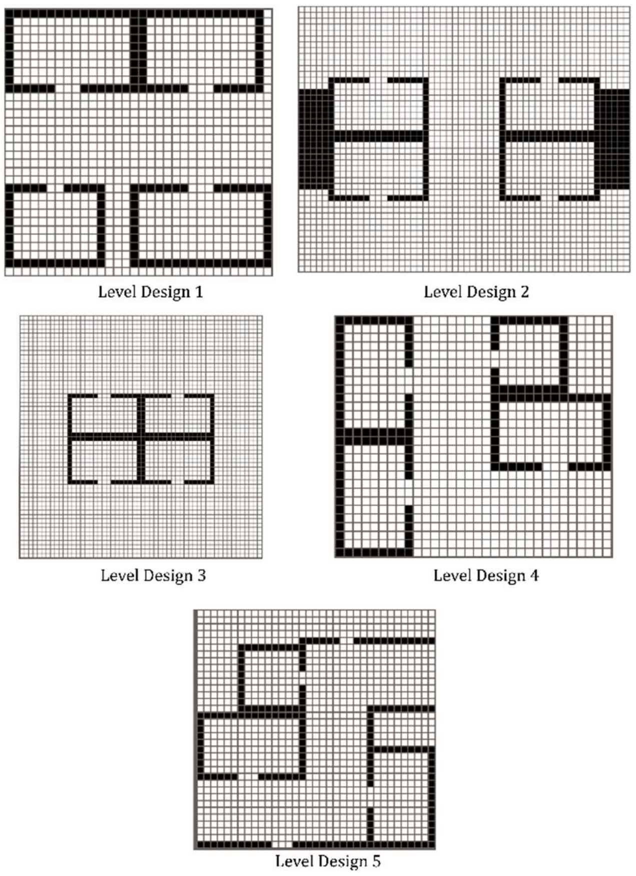

Figure 4 Five Level Designs in Black and White Format

The arrangement of Level Design 1, Level Design 2, and Level Design 3 use a reference to the extent of the level design area from the smallest to the largest because this approach allows for a gradual increase in complexity and challenge in the game. By starting with a smaller level design, players can understand the basics of the game and get through the initial levels more easily [21]. Meanwhile, Level Design 4 and Level Design 5 are arranged based on similarity or resemblance to the level design in the Chaos Crossing game to present a similar or comparable experience in terms of gameplay, level layout, or practical design elements. After the five-level designs were arranged, the Dijkstra and A-Star algorithms were implemented at each level design using the Unity Engine with the c-sharp programming language.

3.2. Result of Testing

The testing process is carried out by direct simulation techniques on the Unity Engine based on route search time, track distance, and memory usage at each design level. The testing process is carried out in stages from design level 1 to design level 5, according to predetermined test scenarios.

Table 2 The Results of Data Collection at each Level Design

Points (x,y)		Time (s)		Distance (unit)		Memory (MB)	
Start	Destination	Dijkstra	A-Star	Dijkstra	A-Star	Dijkstra	A-Star
			Level Design 1				
(13,8)	(28,24)	6.74	0.96	23	23	20.80	18.38
(6,4)	(22,4)	5.23	1.84	29	29	20.32	18.91
(17,8)	(13,24)	6.42	2.66	30	30	21.68	18.30
			Level Design 2				
(19,14)	(36,30)	29.71	9.93	54	54	23.56	22.80

Points (x,y)		Time (s)		Distance (unit)		Memory (MB)	
Start	Destination	Dijkstra	A-Star	Dijkstra	A-Star	Dijkstra	A-Star
(53,11)	(2,33)	35.86	8.24	66	67	22.93	22.51
(19,14)	(36,14)	13.89	5.92	39	39	20.86	20.43
Level Design 3							
(10,28)	(45,27)	56.43	13.08	55	55	22.73	18.83
(25,30)	(30,25)	64.84	30.11	75	74	23.26	18.47
(1,11)	(36,29)	52.50	22.99	64	64	22.66	19.93
Level Design 4							
(20,13)	(6,17)	4.31	1.82	28	29	21.28	18.25
(31,9)	(6,26)	6.25	1.58	35	35	22.37	18.91
(2,8)	(24,24)	6.38	1.49	31	31	21.33	19.87
Level Design 5							
(13,22)	(27,12)	5.97	2.30	28	28	22.53	22.31
(21,31)	(2,12)	9.77	4.13	39	39	22.88	22.13
(1,8)	(33,22)	7.16	2.07	39	40	23.37	22.81

Based on the data obtained in Table 2, it was found that the test results in the route search time on the Dijkstra algorithm obtained a minimum value of 4.31 seconds at design level 4 and a maximum value of 64.84 seconds at design level 3, while the a-star algorithm got the minimum value is 0.96 seconds at design level 1 and the maximum value is 64.84 seconds at design level 3. In the path distance, it is found that Dijkstra's algorithm obtains a minimum value of 23 units at design level 1 and a maximum value of 75 units at design level 3, while the a-star algorithm gets a minimum value of 23 units at design level 1 and a maximum value of 74 units at design level 3. In terms of memory usage, it is found that the Dijkstra algorithm obtains a minimum value of 20.32 MB at design level 1 and a maximum value of 23.37 MB at design level 5, while the A-Star algorithm gets a minimum value of 18.25 MB at design level 4 and a maximum value of 22.81 MB at design level 5.

3.3. Analysis of Test Results

After collecting data from the test results, the data is analyzed according to research data analysis techniques. The average and the comparison of algorithms are calculated based on the comparison parameters, namely route search time, track distance, and memory usage.

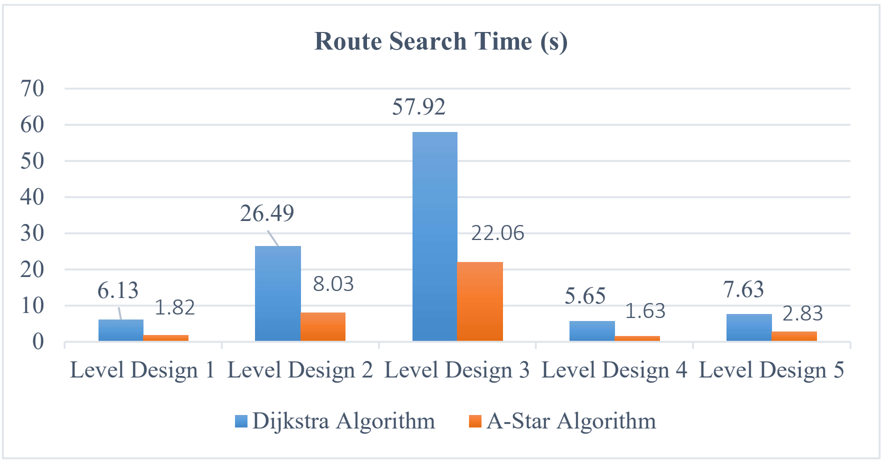

Figure 5 Comparison of the average Dijkstra Algorithm and A-Star based on Route Search Time

The results of the first analysis on route search time assume the smaller the average route search time for each level design, the faster the route search algorithm will be. Based on the results of calculating the average search time for each level design in Figure 5, it can be concluded that from level design 1 to level design 5, the A-Star algorithm is faster in searching routes than the Dijkstra algorithm. The same thing by previous research, the A-Star algorithm is faster than the Dijkstra algorithm based on search time [10][15].

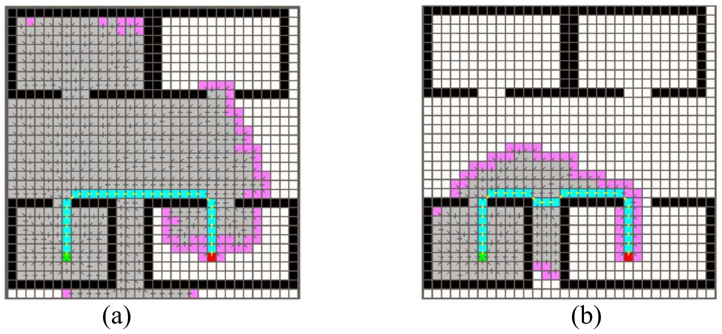

Figure 6 Visualization of Testing (a) Dijkstra and (b) A-Star Algorithm at the Level Design 1 point (6,4) to point (22,4)

The results of the visualization test between the Dijkstra algorithm and the A-Star algorithm in Figure 6 show that the A-Star algorithm has fewer explore points than the Dijkstra algorithm. This also indicates that the search time for the A-Star algorithm is shorter than the Dijkstra algorithm, as explained in Figure 6, with the grey color representing the algorithm's explore point.

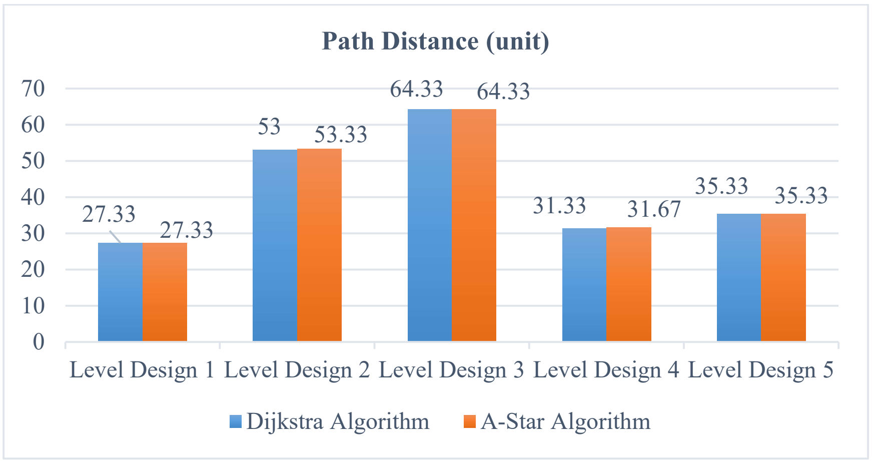

Figure 7 Comparison of the average Dijkstra Algorithm and A-Star based on Path Distance

The results of the second analysis on route search time assume the smaller the average path distance for each level design, the faster the NPC will move from the starting point to the destination. Based on the average calculation results based on the path distance for each level design in Figure 7, there are slightly different values at level design 2 and level design 4, so it can be concluded from the results of the other algorithms at level design 2 and level design 4 that the Dijkstra algorithm produces a shorter path distance so that the movement of the NPC in moving from the starting point to the destination is faster than the A-Star algorithm.

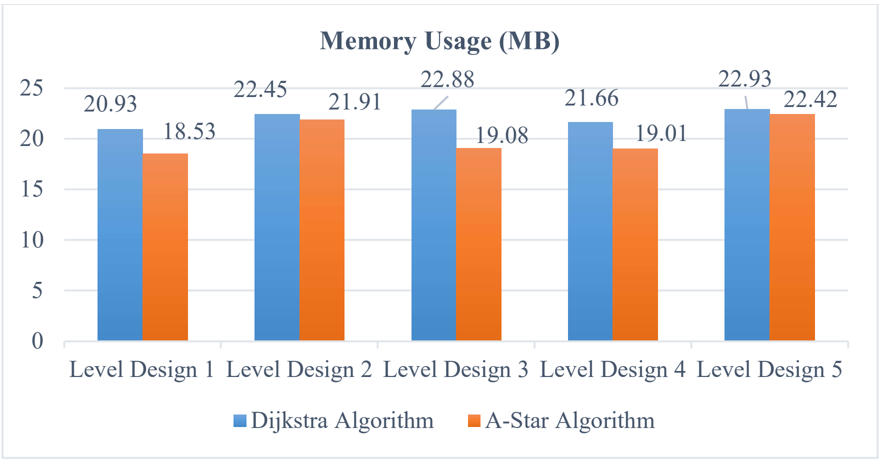

Figure 8 Comparison of the average Dijkstra Algorithm and A-Star based on Memory Usage

The results of the third analysis on route search time assume the smaller the average memory usage for each level design, the lighter the algorithm for searching routes in one iteration. Based on the average results for each level design in Figure 8, from level design 1 to level design 5, the A-Star algorithm is easier to search for routes in one iteration than the Dijkstra algorithm.

4. Conclusions

The pathfinding method with the Dijkstra algorithm and the A-Star algorithm can optimally solve the problem of NPC movement to the destination point without being hampered by the environment in the Chaos Crossing game. The test results and comparative analysis show that the A-Star algorithm performs a faster route search with an average value of 36.37 seconds than Dijkstra's algorithm with an average matter of 20.76 seconds and utilizes memory more efficiently with an average value of 20.19 MB than Dijkstra's algorithm with a value 22.17 MB on average. However, Dijkstra's algorithm produces a slightly shorter track distance, with an average value of 42.26 units, compared to the A-Star algorithm, with an average value of 42.39 units.

Algorithm comparison analysis can consider using other, more optimal algorithms in certain situations. Future research can also compare the effectiveness of the Dijkstra and A-Star algorithms in specific situations. Integrating other artificial intelligence technologies into developing NPCs, such as machine learning, is possible to enhance their ability to interact with the game environment and improve the player's playing experience.

Bibliography

[1] B. Xia, X. Ye, and A. O. M. Abuassba, “Recent Research on AI in Games,” *2020 Int. Wirel. Commun. Mob. Comput. IWCMC 2020*, no. June, pp. 505–510, 2020, doi: 10.1109/IWCMC48107.2020.9148327.

[2] A. Rafiq, T. Asmawaty Abdul Kadir, and S. Normaziah Ihsan, “A Review of Artificial Intelligence in Serious Game for Public Health,” *J. Phys. Conf. Ser.*, vol. 1830, no. 1, 2021, doi: 10.1088/1742-6596/1830/1/012001.

[3] S. A. Yakan, “Analysis of Development of Artificial Intelligence in the Game Industry,” vol. 2, no. 2, pp. 111–116, 2022.

[4] Y. Y. Dyulicheva and A. O. Glazieva, “Game based learning with artificial intelligence and immersive technologies: an overview,” *CEUR Workshop Proc.*, vol. 3077, pp. 146–159, 2022.

[5] T. T. Sujaka, K. A. Latif, S. Hadi, H. Hasbullah, and R. Hammad, “A* Pathfinding Applications in Two-Dimensional AI Video Games,” *Ina. Indones. J. Electr.*

Eletronics Eng., vol. 5, no. 1, pp. 25–29, 2022, doi: 10.26740/inajeee.v5n1.p25-29.

[6] P. Harsadi and S. Siswanti, "Penerapan Pathfinding Menggunakan Algoritma A* Pada Non Player Character (NPC) Di Game," *J. Ilm. SINUS*, vol. 17, no. 2, p. 39, 2019, doi: 10.30646/sinus.v17i2.423.

[7] Y. Sazaki, H. Satria, and M. Syahroyni, "Comparison of A∗ and dynamic pathfinding algorithm with dynamic pathfinding algorithm for NPC on car racing game," *Proceeding 2017 11th Int. Conf. Telecommun. Syst. Serv. Appl. TSSA 2017*, vol. 2018-Janua, pp. 1–6, 2018, doi: 10.1109/TSSA.2017.8272918.

[8] A. Candra, M. A. Budiman, and R. I. Pohan, "Application of A-Star Algorithm on Pathfinding Game," *J. Phys. Conf. Ser.*, vol. 1898, no. 1, 2021, doi: 10.1088/1742-6596/1898/1/012047.

[9] E. Agung, "Implementasi Metode Pathfinding dengan Algoritma A* pada Game Rogue-like menggunakan Unity," *Indones. J. Comput.*, vol. 1, no. 3, pp. 81–89, 2022, doi: 10.14710/jtk.v1i3.36700.

[10] R. N. Sarbini, I. Ahmad, R. O. Bura, and L. Simbolon, "Comparative Analysis of Pathfinding Artificial Intelligence Using Dijkstra and a* Algorithms Based on Rpg Maker Mv," *J. Ris. Inform.*, vol. 4, no. 3, pp. 283–290, 2022, doi: 10.34288/jri.v4i3.384.

[11] R. Shen, V. Gopalan, and S. Runkun, "A Comparative Review of 2 . 5D vs 3D Multiplayer Online Battle Arena Game Experience," 2022.

[12] M. S. Amin, P. Subarkah, R. Umma, and E. B. Prasetya, "Implementasi Algoritma Dijkstra pada Game Strategi RPG Berbasis Web dengan Framework Javascript P5," *J. IT CIDA*, vol. 8, no. 1, pp. 41–55, 2022.

[13] P. Jakhar, "International Journal of Research Publication and Reviews A Overview on Pathfinding Algorithm Between A * a nd Dijkstra ' s Algorithm for 2D Platformer Games," vol. 3, no. 11, pp. 91–94, 2022.

[14] R. A. Krisdiawan, A. Permana, E. Darmawan, F. Nugraha, and A. Kriswandiyanto, "Implementation Dijkstra's Algorithm for Non-Players Characters in the Game Dark Lumber," *J. Phys. Conf. Ser.*, vol. 1933, no. 1, 2021, doi: 10.1088/1742-6596/1933/1/012006.

[15] S. D. Handy Permana, K. B. Yogha Bintoro, B. Arifitama, and A. Syahputra, "Comparative Analysis of Pathfinding Algorithms A *, Dijkstra, and BFS on Maze Runner Game," *IJISTECH (International J. Inf. Syst. Technol.*, vol. 1, no. 2, p. 1, 2018, doi: 10.30645/ijistech.v1i2.7.

[16] J. Adler and B. F. Ramadhan, "Penerapan Algoritma Dijkstra Pada Game Learning Matematika Berbasis Android," *Komputika J. Sist. Komput.*, vol. 10, no. 2, pp. 173–181, 2021, doi: 10.34010/komputika.v10i2.4551.

[17] I. B. Gede Wahyu Antara Dalem, "Penerapan Algoritma A* (Star) Menggunakan Graph Untuk Menghitung Jarak Terpendek," *J. Resist. (Rekayasa Sist. Komputer)*, vol. 1, no. 1, pp. 41–47, 2018, doi: 10.31598/jurnalresistor.v1i1.253.

[18] M. M. Attoyibi, F. Emma Fikrisa, and A. N. Handayani, "The Implementation of A Star Algorithm (A*) In the Game Education About Numbers Introduction," vol. 242, no. Icovet 2018, pp. 234–239, 2019, doi: 10.2991/icovet-18.2019.57.

[19] A. W. R. Ramadhan and D. Udjulawa, "Perbandingan Algoritma Dijkstra dan Algoritma A Star pada permainan Pac-Man," *J. Algoritm.*, vol. 1, no. 1, pp. 12–20, 2020, doi: 10.35957/algoritme.v1i1.411.

[20] N. Kühl, M. Goutier, R. Hirt, and G. Satzger, "Machine learning in artificial intelligence: Towards a common understanding," *Proc. Annu. Hawaii Int. Conf. Syst. Sci.*, vol. 2019-Janua, pp. 5236–5245, 2019, doi: 10.24251/hicss.2019.630.

[21] T. Karlsson, J. Brusk, and H. Engström, "Level Design Processes and Challenges: A Cross Section of Game Development," *Games Cult.*, 2022, doi: 10.1177/15554120221139229.