Journal of Artificial Intelligence and Engineering Applications

Website: https://ioinformatic.org/

15th June 2025. Vol. 4. No. 3; e-ISSN: 2808-4519

Implementation of Fuzzy Logic Algorithm to Improve NPC Decision-Making in 2D Adventure Games Using Unity

Kevin[1]*, Octara Pribadi[2], Hendri[3]

[1,2,3]*STMIK TIME, Medan*
Kevinmers17@gmail.com[1]*

Abstract

In 2D adventure games, Non-Playable Characters (NPCs) play a crucial role in creating a more immersive and interactive experience. However, static and non-adaptive NPC behavior may reduce the game quality. This study aims to enhance NPC artificial intelligence by implementing the Fuzzy Logic algorithm in decision-making processes. The input parameters include the distance between the NPC and the player, the player's health level, and the player's level, with four possible outputs: chase, evade, defend, and wait. A fuzzy rule base consisting of 27 rules was developed and implemented in a Unity-based game. Testing was conducted on various input combinations to evaluate the NPC's responses. Results indicate that NPCs respond more adaptively, such as evading when the player has high health and level at a close range, or waiting when the situation is unfavorable. This implementation improves interaction dynamics between NPCs and players, and adds strategic depth to the gameplay.

Keywords: *Fuzzy Logic, NPC, 2D Game, Unity, Decision-Making*

1. Introduction

The game industry continues to grow rapidly alongside technological advancements and the rising popularity of mobile devices and gaming platforms. 2D adventure games remain popular due to their mechanical simplicity and visual appeal [1]. However, developers often face the challenge of creating realistic artificial intelligence (AI) for Non-Playable Characters (NPCs) to ensure dynamic and engaging in-game interactions [2].

The Fuzzy Logic algorithm is a potential solution to enhance NPC decision-making in a more adaptive and realistic manner. This algorithm enables NPCs to respond more smoothly to environmental changes and player conditions, as fuzzy logic can handle uncertain or vague variables [3]. For instance, an NPC can adjust its level of aggression based on the player's health or proximity. Implementing fuzzy logic in games has proven effective in making NPCs more responsive, such as in the game Battle Fence, which uses fuzzy logic to control NPC attack and defense behavior [4].

Through the implementation of the Fuzzy Logic algorithm in a 2D adventure game built with Unity, NPCs are expected to make smarter and more dynamic decisions. This approach not only increases the gameplay challenge but also enriches the player experience with more lively and unpredictable interactions [3].

This study aims to improve gameplay quality by presenting more realistic and responsive NPCs, thereby enhancing player engagement and satisfaction. Additionally, this research contributes to the development of AI methods in the game industry, particularly in applying Fuzzy Logic in Unity-based games [5]. Therefore, this study is titled:
"Implementation of Fuzzy Logic Algorithm to Improve NPC Decision-Making in 2D Adventure Games Using Unity."

2. Research Methods

This research aims to develop an intelligent system for Non-Playable Characters (NPCs) in a 2D adventure game built with Unity using the Fuzzy Logic algorithm. This approach was chosen because it is capable of handling uncertainty and complex conditions, unlike conventional methods such as Finite State Machine (FSM), which are limited in terms of flexibility and adaptation to dynamic game environments.

The development process began with analyzing the needs and weaknesses of traditional NPC systems. A fuzzy model was then constructed using three main input parameters: player distance, player health, and player level (based on the number of spirit rings collected). The

system output consisted of four possible NPC actions: attack, evade, defend, and wait. These actions were determined using fuzzy rules formulated in the IF–THEN format.

The fuzzy algorithm implementation consisted of three main stages:

a. Fuzzification: Converts numerical input values into linguistic variables using trapezoidal membership functions.
b. Inference: Processes the fuzzy logic rules by applying the AND operator with the MIN-MAX method to determine degrees of confidence for each action.
c. Defuzzification: Applies the Mean of Maximum (MoM) method to produce a crisp output value that forms the basis of the NPC's final decision.

The system workflow is illustrated in Fig 1.

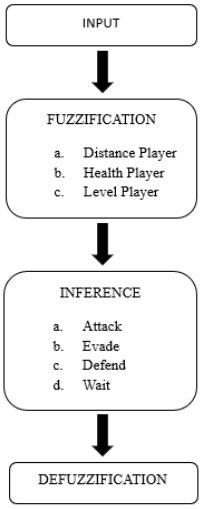

Fig. 1: NPC Decision-Making System Using Fuzzy Logic

Each stage in the diagram is described as follows:

a. Game Input: The system receives input from the game environment, including the distance between the NPC and the player, the player's health, and player level (represented by the number of collected spirit rings).
b. Fuzzification: Each input is transformed into fuzzy linguistic terms (e.g., Near, Medium, Far; Low, Medium, High; Beginner, Intermediate, Advanced) using trapezoidal membership functions.
c. Inference: The fuzzy rule base processes the combination of inputs to produce fuzzy outputs representing the degree of belief for each possible action.
d. Defuzzification: Fuzzy output values are converted into crisp values using the Mean of maximum (MoM) method. The action corresponding to the highest output value is selected as the final decision.
e. NPC Output: The NPC executes the selected action based on the defuzzified result, resulting in more realistic and adaptive behavior.

Through this approach, NPCs are expected to dynamically adjust their decisions in response to gameplay changes, leading to more challenging and immersive player experiences.

3. Results and Discussion

3.1. Game Interface

The developed game features an initial menu screen with three main options: Play, which starts the game; Settings, which allows the player to adjust background music volume; and Exit, which closes the game application.

Fig. 2: Game Main Menu Display

The gameplay consists of three maps, each with a unique environment designed to test the behavior of Non-Playable Characters (NPCs) under different conditions.

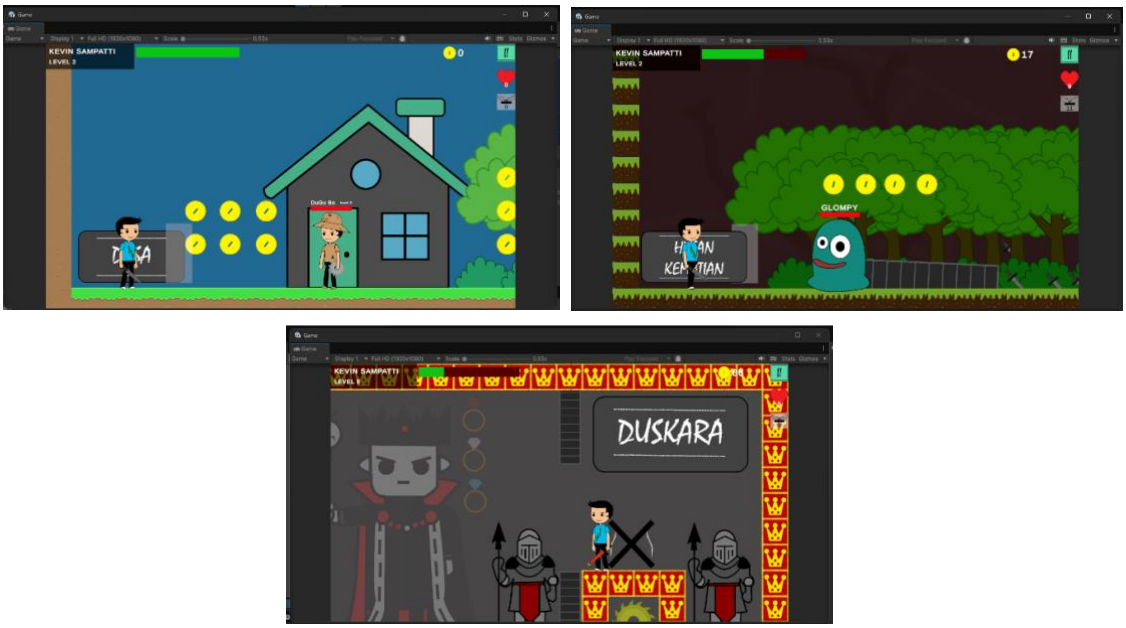

Fig. 3: Gameplay Map Design

3.2. Fuzzy Logic Testing

The NPC decision-making system was tested using various input combinations for distance, player health, and level. All output results were logged through Unity's debug console.

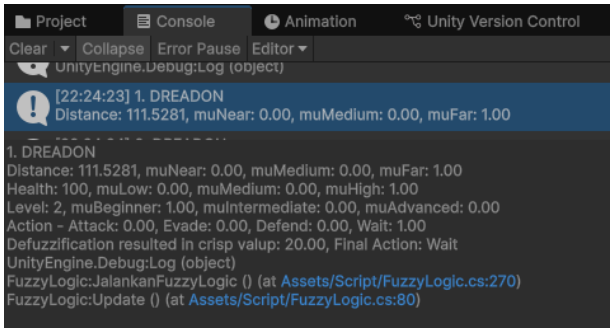

Fig. 4: Fuzzy Logic Testing Output

In the first test, a player distance of 111.5281 resulted in a full membership value ($\mu = 1$) in the Far category. The membership values for Near and Medium were both 0.

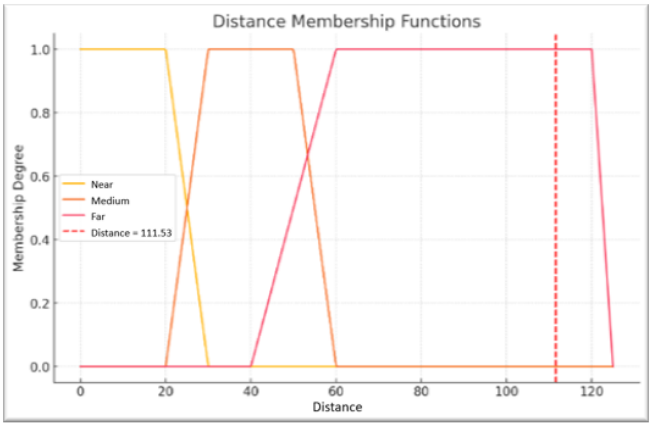

Fig. 5: Distance Fuzzification

A health value of 100 placed the player fully in the High health category ($\mu = 1$), with Low and Medium categories at 0.

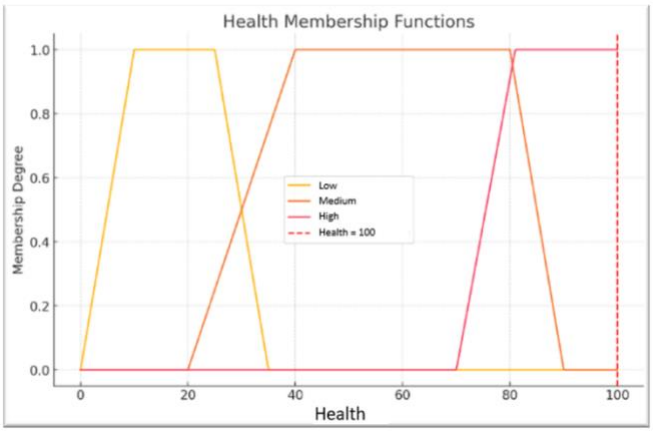

Fig. 6: Health Fuzzification

At level 2, the player was categorized as a Beginner, with a full membership value (μ = 1), while the Intermediate and Advanced categories held 0 values.

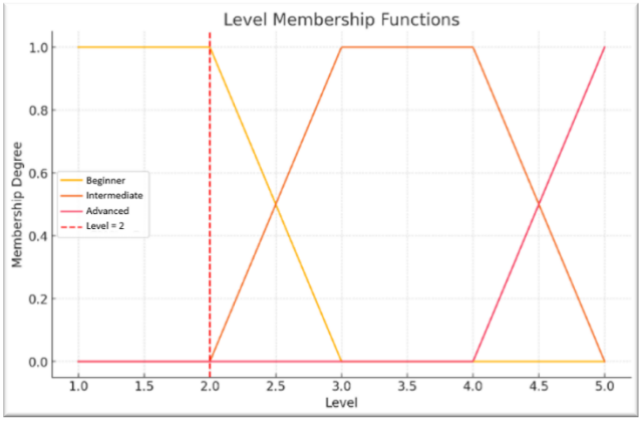

Fig. 7: Level Fuzzification

From the inference process, it was found that only the Wait action had a non-zero membership value (μ = 1), while the other actions (Attack, Evade, and Defend) were 0.

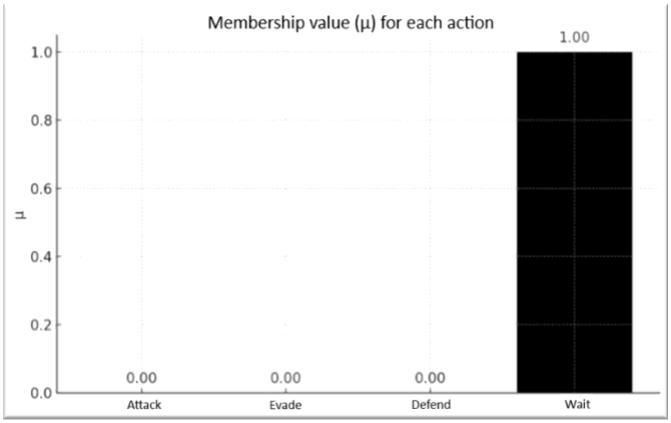

Fig. 8: Inference Process

In the final step, defuzzification was applied to convert the fuzzy output into a crisp value using the Mean of Maximum (MoM) method. This method calculates the average of all output values with the maximum degree of membership.

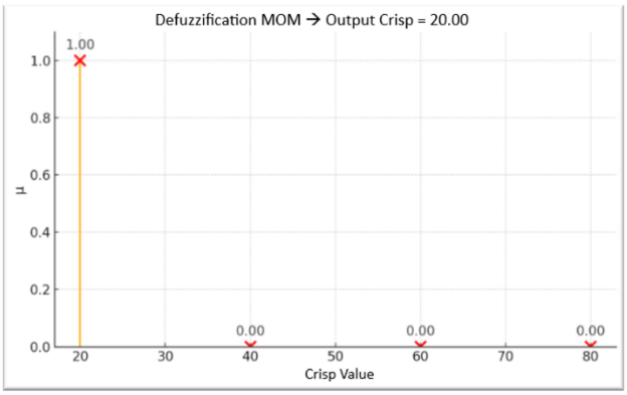

Fig. 9: Defuzzification Process

The following table shows the crisp values for each possible action based on the inference results:

Table 1: Crisp Value Results for Each Action

Action	Crisp Value	Membership (μ)
Attack	80	0
Evade	60	0
Defend	40	0
Wait	20	1

Additional tests were conducted using a wider range of input scenarios. Table 2 summarizes 15 trials using different combinations of player distance, health, and level to determine the most suitable NPC action.

Table 2: Extended Fuzzy Logic Test Results

No	Distance	μ_Near	μ_Medium	μ_Far	Health	μ_Low	μ_Med	μ_High	Level	μ_Begin	μ_Middle	μ_Advanced	Crisp	Valid Action (μ_highest)
1	102.67	0	0	1	100	0	0	1	2	1	0	0	20	**Wait (1)**
2	87.99	0	0	1	50	0	1	0	2	1	0	0	80	**Attack (1)**
3	10.22	1	0	0	50	0	1	0	2	1	0	0	80	**Attack (1)**
4	35.59	0	1	0	52.59	0	1	0	2	1	0	0	80	**Attack (1)**
5	51.68	0	0.83	0.58	52.59	0	1	0	3	0	1	0	40	**Defend (0.83)**
6	43.08	0	1	0.15	15	1	0	0	3	0	1	0	80	**Attack (1)**
7	21.44	0.86	0.14	0	15	1	0	0	3	0	1	0	80	**Attack (0.86)**
8	10.10	1	0	0	41.44	0	1	0	4	0	1	0	80	**Attack (1)**
9	43.71	0	1	0.19	93	0	0	1	4	0	1	0	60	**Evade (1)**
10	93.23	0	0	1	80	0	1	0.91	4	0	1	0	40	**Defend (1)**
11	20.53	0.95	0.05	0	81.12	0	0.89	1	5	0	1	0	40	**Defend (0.95)**
12	29.51	0.05	0.95	0	95.87	0	0	1	5	0	1	0	60	**Evade (0.95)**
13	55.71	0	0.43	0.79	45.84	0	1	0	5	0	1	0	20	**Wait (0.79)**
14	41.79	0	1	0.09	15	1	0	0	5	0	1	0	40	**Defend (1)**
15	21.42	0.86	0.14	0	15	1	0	0	5	0	1	0	80	**Attack (0.86)**

These results show that the fuzzy logic system successfully adapts NPC decisions based on different input conditions. For example, when the player is close and has high health, the NPC chooses to evade or defend; when the player is far and has low health, the NPC tends to attack. This dynamic behavior enhances gameplay and provides a more realistic and engaging experience for players.

4. Conclusion

This study aimed to develop artificial intelligence for Non-Playable Characters (NPCs) in a 2D adventure game using the Fuzzy Logic algorithm. Based on the design, implementation, and testing carried out in Unity, the system successfully generated NPC behavior that is more adaptive to gameplay conditions. By considering three input variables—distance, health, and level—the NPC was able to make more realistic decisions, such as attacking, defending, evading, or waiting. The testing results demonstrate that the fuzzy system can operate with high accuracy and efficiently in real-time.

The implementation of the fuzzy method not only enhances the quality of interaction between players and NPCs but also opens opportunities for developing more complex NPC intelligence. For future research, it is recommended that the system be expanded with more strategic behavior variations, tested in larger-scale game environments, and integrated with other AI algorithms such as machine learning. Furthermore, this fuzzy logic method has the potential to be applied in various other game genres, both 2D and 3D, to provide a more immersive gaming experience.

References

[1] I. Satrio, F. Santi Wahyuni, dan D. Rudhistiar, "PENERAPAN A* PATHFINDING DAN FSM (FINITE STATE MACHINE) PADA GAME 'LOST CIVILIZATION' BERBASIS ANDROID," Jurnal Mahasiswa Teknik Informatika, vol. 6, no. 2, hlm. 1192–1199, 2022.

[2] A. Qoiriah dan A. Haqi Annazili, "Implementasi Algoritma Fisher-Yates Shuffle Dan Fuzzy Tsukamoto Pada Game Petualangan Si Thole Berbasis Android Menggunakan Game Engine Unity," Journal of Informatics and Computer Science), vol. 1, no. 4, hlm. 188–199, 2020.

[3] R. Amalia, "GAME EDUKASI DAN CERITA INTERAKTIF SEJARAH KERAJAAN DI SUMATRA MENGGUNAKAN ALGORITMA FUZZY SUGENO UNTUK MENGATUR PERILAKU NPC," Jurnal Informatika dan Rekayasa Perangkat Lunak (JATIKA), vol. 1, no. 2, hlm. 192–202, 2020.

[4] I. Ilham Shagianto, G. Wahyu Wiriasto, D. Fikry Budiman, dan N. Made Seniari, "Aplikasi Game berbasis Andorid 2D dengan Logika Fuzzy pada NPC (Non-Player Character)," Journal of Electrical Engineering and Information technology, vol. 1, no. 1, hlm. 41–56, 2023.

[5] C. Giovani Simbolon, A. Tri Hanuranto, dan A. Novianti, "DESAIN DAN IMPLEMENTASI PROTOTIPE PENDETEKSI DINI KEBAKARAN GEDUNG MENGGUNAKAN ALGORITMA FUZZY LOGIC BERBASIS INTERNET OF THINGS (IOT)," e-Proceeding of Engineering, vol. 7, no. 2, hlm. 3532–3539, 2020.