Media Jurnal Informatika
Vol. 17, No. 2, December 2025
p-issn : 2088-2114, e-issn : 2477-2542

Dynamic Difficulty Adjustment Using Reinforcement Learning for Adaptive Gameplay Experience in Resik

Diny Syarifah Sany [a,1,*], M Malwan Angkasa[b,2]

[a,b] Suryakancana University, Pasir Gede Raya Street, Cianjur 43216, Indonesia
[1]dsy.sany@gmail.com*; [2] angkasa@gmail.com
* corresponding author

ARTICLE INFO

Article history
Received 2025/12/06
Revised 2025/12/25
Accepted 2025/12/26

Keywords
Educational Game
Dynamic Difficulty Adjustment
Reinforcement Learning
PPO
Waste Management

ABSTRACT

The development of educational games holds significant potential for interactively instilling environmental conservation concepts, such as reduce, reuse, and recycle (3R). However, varying player skill levels often lead to boredom when a game is too easy, or frustration when it is too difficult. Although Reinforcement Learning (RL)-based Dynamic Difficulty Adjustment (DDA) has proven effective in balancing difficulty levels in action or MOBA genres, its application in educational waste management games remains underexplored. This study aims to develop an educational game titled "Resik" and implement a DDA mechanism using the Proximal Policy Optimization (PPO) RL algorithm, enabling the game's difficulty to adjust to player proficiency adaptively and in real-time. This research employs a Research and Development (R&D) and experimental approach. The adaptation parameters include the waste spawn rate, mission timer, sorting error rate, and NPC speed. The RL model was developed and integrated into the Unity Engine for the Android platform. Evaluation was conducted through bot simulations (15 iterations) and field trials involving 50 students from SMKS Bina Bangsa Pertiwi. The integration of the PPO RL algorithm into the game engine was successfully implemented, allowing the game to respond to player performance and dynamically adjust the difficulty. The application of RL-based DDA proved effective in maintaining the gameplay experience within the adolescent players' flow zone. These findings contribute to the development of adaptive educational games while simultaneously supporting the improvement of waste management literacy.

1. Introduction

The development of the gaming industry today is no longer solely oriented toward entertainment; it is also widely utilized for educational purposes, including efforts to raise environmental awareness [1]. Educational games possess strong potential to interactively instill essential concepts such as reduce, reuse, and recycle (3R) [2]. This gamification approach is believed to enhance adolescents' learning motivation and encourage pro-environmental decision-making more effectively than conventional learning methods [3], [4]. However, one of the major challenges in developing educational games is the variance in player skill levels. An engaging gameplay experience heavily relies on how well the game tailors its challenges to keep players within the flow zone an optimal psychological state where individuals are fully immersed and enjoy the activity without experiencing boredom or anxiety [5], [6]. If the difficulty level is too low, players will quickly become bored, whereas if it is too high, they will become frustrated and lose interest [7].

To address this issue, Dynamic Difficulty Adjustment (DDA) serves as a solution that enables game systems to dynamically adjust the difficulty level based on players' abilities and real-time performance [8], [9]. One of the most advanced methods for optimizing DDA is through the use of Reinforcement Learning (RL), a branch of machine learning wherein an agent learns to make optimal decisions based on feedback (rewards and penalties) from its environment [10], [11]. In the context of gaming, RL agents can learn player behavior profiles and continuously calibrate game parameters [12].

Previous studies have demonstrated that RL-based DDA can enhance satisfaction, maintain fairness, and improve player retention rates through much smoother adjustments compared to static rule-based (heuristic) DDA methods [13], [14], [15]. Meanwhile, research on serious games for waste management continues to grow [16], [17]. However, a significant research gap exists: although RL has been widely tested and applied in action and MOBA game genres [18], its exploration and implementation in educational games specifically tailored for waste management remain highly limited.

Therefore, this study aims to bridge this gap by developing an educational game titled "Resik," equipped with a DDA mechanism based on the Proximal Policy Optimization (PPO) RL algorithm. This algorithm was selected due to its proven stability and efficiency in training agents within complex interactive simulation environments [19]. This research focuses on designing difficulty adaptation parameters such as waste spawn rates, mission timers, and object movement speeds to ensure the game serves as an adaptive, effective, and enjoyable learning medium.

2. Method

This study employs a Research and Development (R&D) approach combined with a quantitative experimental method. The R&D approach is utilized to design and develop the system from the prototyping stage into a functional software product. Meanwhile, the experimental method is applied to test and validate the reliability of the artificial intelligence algorithm embedded within the system against end-user responses.

2.1 Type and Approach of Research

This study employs a Research and Development (R&D) approach combined with a quantitative experimental method. The R&D approach is utilized to design and develop the system from the prototyping stage into a functional software product. Meanwhile, the experimental method is applied to test and validate the reliability of the artificial intelligence algorithm embedded within the system against end-user responses.

2.2 Object and Scope of Research

The primary object of this study is an educational game prototype titled "Resik." The game focuses on environmental education, specifically promoting an understanding of waste management based on the 3R (Reduce, Reuse, Recycle) principles. The scope of the research is limited to the development and testing of a Dynamic Difficulty Adjustment (DDA) module utilizing a Reinforcement Learning (RL) algorithm. The difficulty parameters adapted in real-time by the system encompass four specific variables: waste spawn rate, mission time limit (timer), sorting error tolerance, and Non-Player Character (NPC) movement speed. Furthermore, the system was developed for deployment on Android-based devices.

2.3 Data Collection Techniques

Data collection in this study was conducted through two primary sources. Secondary data, comprising system logs, were obtained through simulation observation techniques, in which a bot agent was executed for 15 iterations to record the RL model's training metrics (e.g., accumulated rewards). Primary data were collected via field surveys using a Likert-scale questionnaire instrument. The test population consisted of adolescent students, and a purposive sampling technique was employed to select a sample of 50 participants (aged 15–18 years) from SMKS Bina Bangsa Pertiwi. The questionnaires were distributed immediately after the participants completed the game testing sessions.

2.4 Tools and Materials Used

The development and testing of the system relied on a specific suite of software and frameworks. The Unity Engine was employed as the primary platform to construct the game environment, user interface, and physics mechanics. To implement the intelligent agent, this study utilized the Unity Machine Learning Agents Toolkit (ML-Agents).

2.5 Research Procedures or Stages

2.5.1 Literature Review and Preliminary Research

This phase aimed to establish a comprehensive understanding of the theoretical and practical foundations of Dynamic Difficulty Adjustment (DDA) and Reinforcement Learning (RL). The process commenced with a literature review to identify research gaps, particularly the limitations of traditional DDA methods in handling games with complex mechanics and real-time adaptation requirements. Subsequently, relevant difficulty parameters for the game *Resik* were identified, including mission completion time, error rate, and enemy variation. The outcomes of this phase comprised a list of adaptable parameters and a research gap analysis matrix, which served as the foundation for designing the RL model.

2.5.2 RL Model Design for DDA

In this stage, a specific RL architecture was designed for the DDA system within the game *Resik*. The primary RL components—namely the state (player condition), action (difficulty adjustment), and reward function (predicated on the theory of challenge-skill balance)—were defined in detail. The Proximal Policy Optimization (PPO) algorithm was selected due to its efficacy in managing dynamic environments. The deliverables of this design phase included the model architecture diagram and the algorithmic pseudocode.

2.5.3 Implementation of the RL Model

The designed RL model was integrated into the *Resik* game prototype, which was developed using the Unity Engine. The implementation process involved creating an API to interface the RL model with the game mechanics, facilitating adjustments such as enemy speed or resource allocation. Model training was conducted using a player behavior dataset gathered during initial trials. The final outcome of this phase was a functional game prototype featuring an RL-based DDA system capable of real-time operation.

2.6 Data Analysis Techniques

The collected data were analyzed using a two-pronged approach. First, the algorithm's performance was evaluated by analyzing the training metrics through TensorBoard. Indicators such as Cumulative Reward and Policy Loss were interpreted to verify that the model remained stable, achieved convergence, and did not experience overfitting. Second, the user questionnaire data were analyzed employing quantitative descriptive statistics to measure satisfaction levels, engagement, and the effectiveness of the DDA system in mitigating player frustration.

3. Results and Discussion

3.1 Preliminary Research Phase

The literature review and preliminary research phase was conducted to identify relevant aspects within the game *Resik* to serve as the foundation for adaptation by the Reinforcement Learning (RL)-based Dynamic Difficulty Adjustment (DDA) system. This process was carried out by (1) analyzing the core game mechanics, (2) reviewing current research findings regarding DDA and RL within the gaming domain, and (3) comparing these approaches with the requirements of waste management education.

The analysis results indicated that the most influential factors on the player experience were the quantity and variety of waste, time limits, error rates, NPC speed, crafting complexity, and rewards/penalties. These factors were selected because they are quantitative in nature (easily measurable by the system) and exert a direct impact on the player's flow and the effectiveness of 3R (Reduce, Reuse, Recycle) learning.

To strengthen the foundation of the RL model design, a gap analysis was also conducted between previous research and the specific needs within the context of environmental educational games. This analysis revealed that the majority of DDA research remains focused on FPS, MOBA, or puzzle genres, relying solely on conventional performance indicators (e.g., score, win/loss ratio, time). In contrast, research explicitly integrating environmental education indicators with adolescent-based evaluations remains highly limited.

Table 1. List of Parameters to be Adapted

No.	Game Parameter	Adaptation Description	Measurement Indicator
1	Waste spawn rate	Increasing/decreasing the quantity of waste spawned per time interval.	Number of objects spawned per minute

No.	Game Parameter	Adaptation Description	Measurement Indicator
2	Waste type variation	Increasing the complexity of organic/inorganic combinations to be sorted.	Sorting success ratio (%)
3	Mission timer	Adjusting the time limit for task completion.	Average task completion time (seconds)
4	Enemy NPC speed	Regulating the speed of anomaly enemies chasing the player.	Number of failures due to NPC capture
5	Inventory capacity	Increasing/decreasing the number of player storage slots.	Frequency of full inventory (%)
6	Point reward/penalty	Adjusting the amount of points, coins, or EXP received.	Average points per mission
7	Adaptive hints/assists	Displaying or hiding aids (e.g., highlights, auto-sort hints).	Number of consecutive errors
8	Crafting complexity	Adjusting the number of steps or materials required to create fertilizer/eco-bricks.	Crafting success rate (%)

The gap analysis matrix was constructed by comparing state-of-the-art research findings with the specific requirements of the *Resik* study. This comparison highlighted the primary research gaps: a lack of integration of educational indicators and a scarcity of empirical trials within the context of Indonesian adolescents. The parameter identification and gap analysis phase yielded several crucial outputs:

a. A comprehensive list of gameplay parameters (waste spawn rate, waste variation, timer, NPC speed, inventory capacity, reward/penalty, adaptive hints, and crafting complexity) that can be monitored and adapted by the RL model.

b. A gap analysis matrix indicating that previous research has predominantly focused on non-educational games, relied on simple rule-based systems, and lacked consideration for 3R educational indicators.

c. Based on these findings, the *Resik* research is positioned to develop an RL-based DDA model that is relevant, adaptive, and demonstrates distinct novelty within the domain of environmental educational games.

3.2 RL Model Design for DDA

This phase involved designing the Reinforcement Learning (RL) architecture based on the Proximal Policy Optimization (PPO) algorithm to implement Dynamic Difficulty Adjustment (DDA) in the game *Resik*. The design encompassed the definitions of the state, action, and reward, as well as the learning mechanism employed.

3.2.1 PPO Model Architecture

PPO was selected due to its high stability, capacity to handle both continuous and discrete action spaces, and relative efficiency for deployment on Android devices. The PPO algorithm utilizes an actor-critic approach, wherein:

- The Actor (policy network) generates difficulty adaptation actions based on the current game conditions.
- The Critic (value network) evaluates the efficacy of these actions in maintaining the player within the optimal condition (flow zone).

3.2.2 Defining the State

The state is a representation of the game conditions observed in real-time. In *Resik*, the state was constructed from a combination of gameplay and educational performance indicators:

Table 2. State in the Game Resik

No.	State Variable	Description
1	Mission completion time	The time taken by the player to complete a mission (seconds).
2	Sorting error rate	Percentage of waste sorting errors (%).
3	Crafting success rate	Percentage of successful fertilizer/eco-brick crafting.
4	Energy / health	Player stamina condition (0–100).
5	Number of failures (deaths)	Frequency of the player being caught by anomaly NPCs.
6	Success/failure streak	Number of consecutive successful/failed missions.

No.	State Variable	Description
7	Inventory capacity	Occupied inventory slots (%).

3.2.4 Defining the Action

The actions generated by the PPO model constitute the forms of game difficulty adjustment. These actions can be either continuous or discrete:

Table 4. Actions in the Game Resik

No.	Adapted Parameter	PPO Action Range
1	Waste spawn rate	0.5x – 1.5x of normal
2	Mission timer	-20% to +20% of standard
3	Enemy NPC speed	0.8x – 1.2x of normal
4	Point reward/penalty	-15% to +15%
5	Adaptive assist (hint)	ON / OFF
6	Crafting complexity	Fewer materials to more materials (levels 1–3)

3.2.5 Reward Function

The reward was designed to encourage the system to keep the player within the flow zone. The reward rules were as follows:

- Positive (+1 to +5): If the player's metrics fell within the target band:
  - Mission completion time: 60–120 seconds
  - Sorting error rate: 5–15%
  - Mission success rate: 70–90%
- Negative (-1 to -5): If the game was too easy (time <40 seconds, error <5%) or too difficult (time >150 seconds, error >20%).
- Additional penalties:
  - Excessively extreme or frequent action changes (oscillation).
  - The player failing consecutively too often.

Through this reward shaping, the PPO learned to provide smooth adaptation actions without disrupting the gameplay experience. The target bands for rewards (e.g., a 5–15% error rate) were selected based on Flow Theory, which suggests that a challenge must slightly exceed a player's current skill to maintain engagement without causing anxiety. A 5–15% error range represents an 'optimal challenge' where the player is making progress but remains alert, preventing the boredom of a 0% error rate and the frustration of errors exceeding 20%.

3.2.6 PPO Flowchart for DDA-Resik

The design of the RL model using PPO enabled the game *Resik* to adjust difficulty dynamically, personally, and stably. By defining a state that combined gameplay and educational indicators, flexible actions, and a flow zone-oriented reward, the system was anticipated to enhance the gaming experience alongside the efficacy of 3R learning among adolescents.

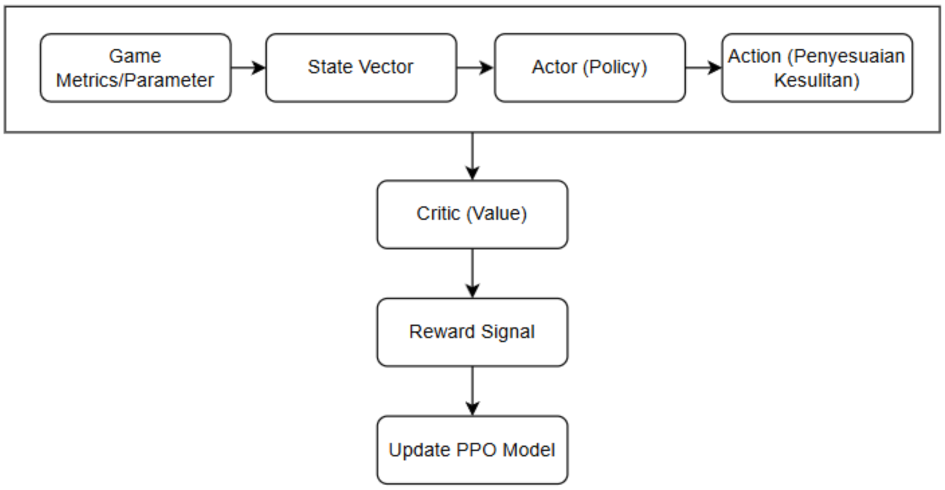

Figure 1. PPO Flowchart for DDA-Resik

The PPO flowchart for DDA in the game *Resik* illustrates how the RL system reads game conditions, makes adaptation decisions, and updates its policy to ensure the game difficulty consistently matches the player's abilities.

a. Game Metrics / Parameters: In this phase, the system collected in-game data, such as mission completion time, waste sorting error rate, number of failures due to NPCs, crafting success, and player energy. These data points, termed game metrics, served as the basis for adaptation.

b. State Vector: All collected parameters were combined into a numerical representation (state vector). This state depicted the player's real-time condition and was used as input for the PPO model.

c. Actor (Policy Network): The actor component was responsible for generating adaptation actions based on the received conditions. Examples included increasing the waste spawn rate, extending the mission timer, decreasing NPC speed, or displaying adaptive hints.

d. Action (Difficulty Adjustment): The actions selected by the actor were directly applied within the game *Resik*. The player would experience smooth and proportional difficulty changes, thereby remaining in the flow zone (neither bored nor frustrated).

e. Critic (Value Network): The critic evaluated whether the adaptation action aligned with the target objective (flow zone). The critic provided an estimated value of how well the actor's chosen action enhanced the player experience.

f. Reward Signal: The system generated a positive reward if the game metrics met the targets (e.g., error rate 5–15%, mission time 60–120 seconds, mission success 70–90%). A negative reward was issued if the game was too easy (causing boredom) or too difficult (causing frustration).

g. Update PPO Model: Based on the received rewards, the actor and critic weights were updated using the PPO mechanism. PPO utilized a clipped objective to prevent excessively extreme updates, ensuring a stable learning process and smoother adaptation.

h. Iterative Loop: Following the model update, the cycle returned to the initial phase (retrieving the latest game metrics). This process repeated continuously during gameplay, forming a real-time adaptive system.

3.2.7 Learning Mechanism

a. Observation: The state was collected every 3 seconds of gameplay.

b. Action Prediction: The actor selected an adaptation action (e.g., increasing the spawn rate by 10%).

c. Guard-Rail Validation: The action was verified to ensure it did not exceed safe thresholds (e.g., the timer could not be <30 seconds).

d. Execution: The action was implemented in the game *Resik*.

e. Evaluation: The critic calculated the reward based on the target band.

f. Policy Update: PPO updated the actor and critic weights using a clipped objective to maintain update stability.

3.2.8 The Core Loop of the Game *Resik*

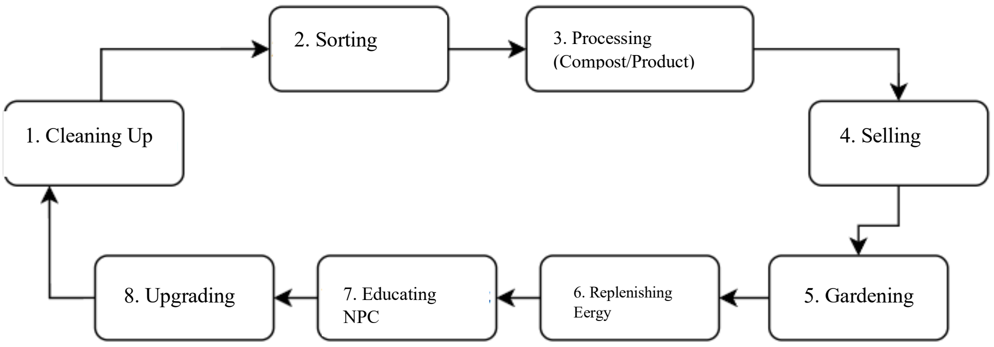

Figure 2. The Core Loop of the Game Resik

The core loop is the primary cycle of activities that continuously repeats within the gameplay to ensure the player's experience remains consistent, meaningful, and aligned with the educational objectives of waste management. The following is an explanation of each phase:

a) Cleaning Up The player commences by cleaning the environment of scattered waste. This phase serves as an initial introduction and provides visual motivation, demonstrating that small actions yield a direct impact on the game world.

b) Sorting The collected waste must be categorized into organic and inorganic types. This process instills 3R (Reduce, Reuse, Recycle) awareness in the player while simultaneously generating sorting accuracy data, which serves as a parameter for the DDA system.

c) Processing (Compost/Products)

- Organic waste is processed into compost or fertilizer.
- Inorganic waste is processed into recycled products (e.g., eco-bricks, handicrafts). This phase imparts the principles of the circular economy and fundamental skills in waste processing.

d) Selling The processed goods can be sold to merchant NPCs in exchange for coins or other items. This establishes an economic feedback loop that motivates the player to continuously collect and process waste.

e) Gardening The acquired coins can be utilized to purchase seeds or gardening tools. The player cultivates plants, which grow at an accelerated rate when utilizing the fertilizer they produced. This stage provides a sense of progression and links waste management to tangible benefits (crop yields).

f) Replenishing Energy In-game activities consume energy. The player can replenish their stamina by consuming their garden harvests or healthy foods. This system instills the value that maintaining a healthy lifestyle is harmonious with a clean environment.

g) Educating NPCs The player interacts with other NPCs (e.g., villagers, school members) to impart education regarding cleanliness, the 3R principles, and waste management. This activity reinforces the *serious game* aspect by expanding the educational impact beyond the player's personal actions.

h) Upgrading Utilizing coins or experience points (EXP), the player can upgrade their equipment (e.g., cleaning tools, inventory capacity, processing facilities). This progression fosters long-term motivation and ensures the gameplay feels expansive and developmental.

i) Repeating the Loop Following the upgrades, the cycle reverts to the initial phase (cleaning up), albeit introducing novel challenges (e.g., increased waste volume, enemy NPCs, heightened crafting complexity). With the assistance of the RL-based DDA, the difficulty level is dynamically adjusted to ensure the player consistently remains within the flow zone.

3.3 Implementation of the RL Model in the Game *Resik*

3.3.1 Integrating the RL Model with the Unity Engine

The Reinforcement Learning (RL) model, designed utilizing the Proximal Policy Optimization (PPO) algorithm, was integrated into the *Resik* game prototype developed within the Unity Engine. This integration process was executed through the development of a custom Application Programming Interface (API) that bridged the RL module with the core game mechanics. This API enabled the system to dynamically modify gameplay parameters, such as:

- The speed of anomaly enemy NPCs,
- The quantity and spawn rate of waste,
- The mission completion timer,
- Point rewards and penalties,
- The appearance of adaptive hints.

Through this API, the RL model could interact directly with the game loop, observe the game conditions (state), execute decisions (actions), and apply modifications in real-time.

3.3.2 RL Model Training Process

The training process for the RL model was conducted once the state, action, and reward structures were fully interfaced with the in-game mechanics within Unity. Training was performed utilizing the Unity ML-Agents External Trainer through the following steps:

- **RL Agent Configuration in Unity:**
  - Adding Behavior Parameters to the agent.
  - Configuring observation sizes based on gameplay parameters (mission time, sorting error rate, NPC failures, inventory capacity, and success streak).
  - Defining adaptation actions, comprising modifications to the spawn rate, timer, NPC speed, and adaptive hints.
- **Training Episode Construction:**
  - Episodes commenced from an identical initial game state.
  - Episodes terminated when the player completed the task or experienced recurrent failures.
  - Rewards were calculated based on the flow zone parameters (too easy, balanced, or too difficult).
- **Training with ML-Agents Trainer:**
  - Training was executed via external command-line prompts.
  - Visual rendering was disabled to accelerate iteration speed.
  - The primary objective was to acquire a stable baseline adaptation model rather than absolute optimization.
- **Monitoring Reward and Stability:**
  - Observing reward increments over time.
  - Monitoring whether adaptation actions were excessively extreme.
  - Applying minor adjustments to the reward shaping when deemed necessary.

The training process yielded an RL model exhibiting sufficient stability for deployment during the runtime inference phase within the gameplay.

3.3.3 Runtime Implementation (Real-Time Inference)

Upon the completion of the training process, the RL model was deployed into the game *Resik* through the following steps:

- Behavior Mode Conversion:
  - Behavior Parameters were transitioned from *Training* to *Inference* mode.
  - The trained model (.onnx format) was directly loaded by Unity.
- Gameplay Integration:
  - The RL agent executed difficulty adaptation decisions at specific intervals.

 - Adjustments were applied incrementally to avoid disrupting the player experience.

- Performance Testing on Android Devices:
  - Ensuring the model did not overload the CPU/GPU capacities.
  - Verifying the absence of lag or latency during the execution of adaptation decisions.

The implementation results demonstrated that the model operated responsively in real-time, with difficulty adjustments feeling seamless and natural to the player.

3.3.4 Implementation Results

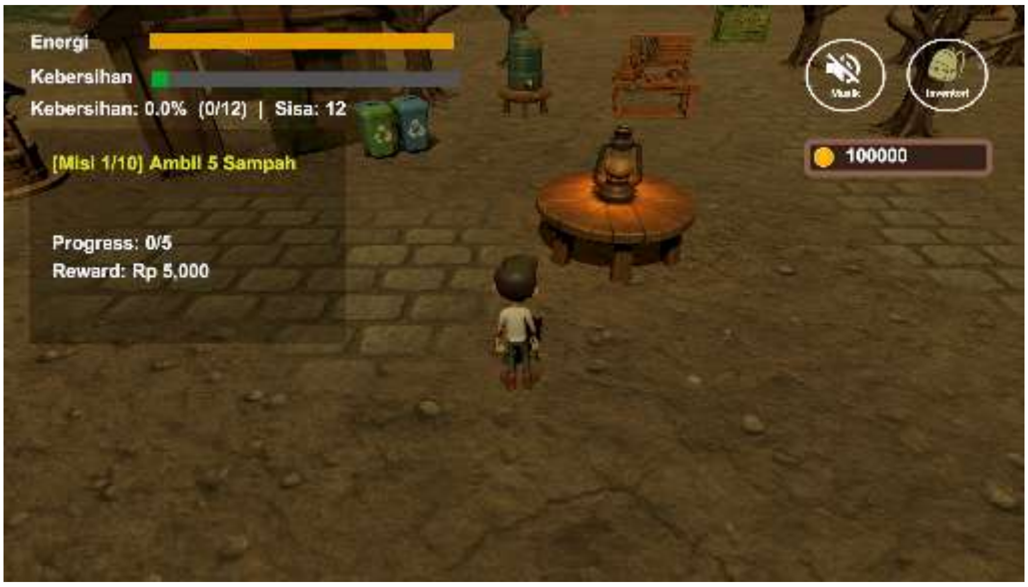

Figure 3. The Game Resik

3.3.5 Testing

3.3.5.1 Bot Simulation Testing

In this study, a bot is defined as a non-human agent controlled by Unity scripts to play the game *Resik* automatically without direct interaction from a human player. Bots were utilized as test agents to simulate various levels of player proficiency, enabling researchers to evaluate whether the Dynamic Difficulty Adjustment (DDA) system could genuinely adapt missions and difficulty levels. Unlike human respondents, bots exhibit consistent and configurable behavioral patterns, making them highly effective for:

- Testing the core functionality of the mission adaptation mechanism (verifying if missions dynamically change).
- Observing the DDA system's response when encountering "players" with varying skill levels (weak, moderate, proficient).
- Generating reproducible and controlled testing data prior to conducting user trials with adolescents.

Bot behavior was intentionally designed with limited patterns, ensuring they did not consistently play "perfectly," but rather simulated the following conditions:

- Weak Bot: Tends to move inefficiently; Frequently fails to collect waste according to the initial target; The amount of waste collected falls significantly below the standard mission target (10 items).
- Moderate Bot: Capable of collecting a moderate amount of waste; Occasionally approaches the initial target but does not consistently achieve it; Represents the condition of a learning player whose performance is not yet stable.
- Proficient Bot: Collects waste rapidly; Capable of achieving or closely approaching the standard mission target; Represents a player who adequately comprehends the game mechanics.

This classification was not determined subjectively but was calculated based on: The mission achievement ratio (the proportion of waste successfully collected versus the initial target of 10 items); and the number of consecutive failures in completing missions. For instance, a bot was categorized as "weak" if, over several consecutive episodes, it failed to complete the mission and only managed to collect less than ±60% of the target waste.

Table 4. Bot Mission Adaptation Simulation Results

Bot ID	Normal Mission Target	Waste Collected by Bot	Subsequent Mission After Adaptation
Bot 01	10 items	3 items	Make fertilizer: 1, Sort waste: 2
Bot 02	10 items	4 items	Make fertilizer: 2, Sort waste: 2
Bot 03	10 items	5 items	Make fertilizer: 2, Crafting: 1, Sort waste: 2
Bot 04	10 items	6 items	Make fertilizer: 3, Crafting: 1, Harvest tomato: 1
Bot 05	10 items	7 items	Make fertilizer: 3, Crafting: 2, Harvest tomato: 1
Bot 06	10 items	8 items	Make fertilizer: 3, Crafting: 2, Harvest tomato: 2, Sort waste: 1
Bot 07	10 items	9 items	Make fertilizer: 4, Crafting: 3, Harvest tomato: 2
Bot 08	10 items	10 items	Make fertilizer: 4, Crafting: 3, Harvest tomato: 3, Sort waste: 2
Bot 09	10 items	2 items	Make fertilizer: 1, Sort waste: 1
Bot 10	10 items	1 item	Make fertilizer: 1

Under initial conditions, all bots were assigned an identical mission target of collecting 10 pieces of waste. However, the simulation results demonstrated that each bot possessed varying capabilities in achieving that target. Therefore, the mission adaptation system did not merely reduce the target numerically but decomposed it into smaller, realistic sub-missions predicated on the bot's prior achievements.

3.3.5.2 User Acceptance Test (UAT)

The User Acceptance Test (UAT) was conducted to determine the level of user acceptance of the educational game *Resik* concerning: Usability; Visual interface and comfort (UI/UX); Suitability of the 3R educational material; Comfort regarding difficulty levels and mission adaptation; and intention to reuse. The UAT results served as the baseline to establish that the game *Resik* is viable for use as a supplementary learning medium within school environments.

The UAT involved 50 Grade X–XI students from SMKS Bina Bangsa Pertiwi as respondents. The testing procedures were as follows:

- Providing a brief explanation of the game's objectives and mechanics.
- Requesting students to independently play the game *Resik* for approximately 20–30 minutes.
- Following the play session, students completed a UAT questionnaire utilizing a 1–5 Likert scale (1 = Strongly Disagree, 2 = Disagree, 3 = Neutral, 4 = Agree, 5 = Strongly Agree).
- Collecting questionnaire data and calculating the average score per indicator.

The recapitulation of the average UAT scores from the 50 students is presented in the following table.

Table 5. Recapitulation of Resik Game UAT Results (Scale 1–5)

Code	Aspect	Average Score	Category
U1	Usability	4.32	High
U2	Navigation & control	4.18	High
U3	Visual appearance	4.40	High
U4	Clarity of 3R material	4.26	High
U5	Difficulty & adaptation	4.08	High
U6	Engagement	4.22	High

Code	Aspect	Average Score	Category
U7	Intention to reuse	4.16	High

In general, all aspects obtained an average score > 4.0, indicating that:

- Students found the game easy to use and the controls intuitive.
- The visual aesthetics and interface design were appealing and comfortable.
- The educational material regarding the 3R principles (reduce, reuse, recycle) was considered clear and helpful.
- The difficulty level, augmented by the DDA system, was perceived as well-balanced (neither excessively difficult nor overly simplistic).
- Students expressed an intention to replay the game, suggesting a favorable potential for user retention.

3.4 Limitation

The execution of this research encountered several constraints that influenced the development, testing, and finalization of the research outputs. The encountered constraints are outlined as follows:

1. Unity and ML-Agents Technological Adaptation: Integrating DDA with an RL approach presented technical hurdles, including matching Unity versions compatible with ML-Agents, API deprecations necessitating script revisions, and the inherently iterative nature of RL training. This resulted in debugging phases extending beyond initial projections.
2. Device Performance for Training and Testing: Training an RL agent necessitates high computational capacity, whereas this study relied on hardware with limited specifications. As a result, training speeds were suboptimal, the number of training episodes was restricted, and visual rendering had to be disabled to ensure efficiency.
3. Game Design Adjustments for the Adolescent Demographic: Initial observations indicated that certain gameplay elements remained too technical, instructions required simplification, and the visuals and interactions needed to be more intuitive. Consequently, several interface revisions were conducted throughout the process.

4. Conclusion

This research successfully developed *Resik*, an educational game prototype integrating waste management and 3R (Reduce, Reuse, Recycle) principles specifically tailored for adolescents. The Reinforcement Learning (RL)-based mission adaptation system was proven highly effective. Bot simulations demonstrated the system's capacity to dynamically scale challenges; the standard 10-waste target was adaptively decomposed into manageable sub-missions—such as fertilizer creation, crafting, and sorting—based on the agent's prior performance. This mechanism ensured that low-performing players could progress without frustration, while proficient players consistently received appropriate challenges.

Furthermore, User Acceptance Testing (UAT) involving 50 students from SMKS Bina Bangsa Pertiwi indicated exceptional user reception. The game achieved average scores ranging from 4.08 to 4.40 across critical aspects, including usability, visual design, 3R educational clarity, and difficulty comfort. Ultimately, the study successfully met its designated objectives, including the fulfillment of the Engineering Faculty grant requirements through a peer-reviewed journal submission, thereby establishing a robust foundation for adaptive educational games.

Based on these findings, several recommendations are proposed for future development and research:

- Technological Advancements: Future iterations should enhance mission adaptation stability, expand player behavior parameters for deeper analysis, and optimize RL training configurations. Incorporating gamification elements, such as leaderboards and daily missions, is also highly advised.
- Educational Implementation: *Resik* should be utilized as a supplementary medium for environmental education. Involving teachers as facilitators will maximize 3R material comprehension, supported by integrated game-based environmental literacy assessments.
- Future Research: Subsequent studies must expand the respondent sample size and age demographics. Conducting empirical comparative analyses between RL-adaptive and non-adaptive versions, alongside evaluating the long-term behavioral impacts on students, will provide profound insights.

Ultimately, this study significantly contributes to the intersection of game technology and environmental awareness, paving the way for future innovations in digital learning.

Acknowledgment

The authors would like to express their deepest gratitude to the Faculty of Engineering, Universitas Suryakancana, for providing the research grant and institutional support that made this study possible. Sincere appreciation is also extended to the administration, teachers, and students of SMKS Bina Bangsa Pertiwi for their invaluable contribution, cooperation, and active participation during the User Acceptance Test (UAT) phase of this research.

References

[1] Gaggi, O., Meneghello, F., Palazzi, C. E., & Pante, G. (2020). "Learning how to recycle waste using a game". *ACM International Conference Proceeding Series*, 144-149. https://dl.acm.org/doi/10.1145/3427325.3427651

[2] Chen, J., He, M., Chen, J., & Zhang, C. (2024). "Exploring the role and mechanisms of environmental serious games in promoting pro-environmental decision-making: a focused literature review and future research agenda". *Frontiers in Psychology*, 15, 1455005. https://doi.org/10.3389/fpsyg.2024.1455005

[3] Chen, J., He, M., & She, S. (2025). "The impact of environmental serious game on pro-environmental behavior through environmental psychological ownership and environmental self-efficacy". *Scientific Reports*, 15(1), 27616. https://doi.org/10.1038/s41598-025-11297-z

[4] Boyle, E. A., et al. (2016). "An update to the systematic literature review of empirical evidence of the impacts and outcomes of computer games and serious games". *Computers & Education*, 94, 178-192. https://doi.org/10.1016/j.compedu.2015.11.003

[5] Csikszentmihalyi, M. (1990). *Flow: The Psychology of Optimal Experience*. Harper & Row. [tautan mencurigakan telah dihapus]

[6] Hunicke, R. (2005). "The case for dynamic difficulty adjustment in games". *Proceedings of the 2005 ACM SIGCHI International Conference on Advances in Computer Entertainment Technology*, 429-433. https://dl.acm.org/doi/10.1145/1168987.1169011

[7] Aponte, M. V., Levieux, G., & Natkin, S. (2009). "Scaling the level of difficulty in single player video games". *International Conference on Entertainment Computing*, 24-35. Springer. https://link.springer.com/chapter/10.1007/978-3-642-04052-8_3

[8] Zohaib, M. (2018). "Dynamic Difficulty Adjustment (DDA) in Computer Games: A Review". *Advances in Human-Computer Interaction*, 2018. https://doi.org/10.1155/2018/5681652

[9] Xue, S., Wu, M., Kolen, J., Aghdaie, N., & Zaman, K. A. (2017). "Dynamic Difficulty Adjustment for Maximized Engagement in Digital Games". *Proceedings of the 26th International Conference on World Wide Web Companion*, 465-471. https://dl.acm.org/doi/10.1145/3027063.3053104

[10] Sutton, R. S., & Barto, A. G. (2018). *Reinforcement Learning: An Introduction* (2nd ed.). MIT Press. http://incompleteideas.net/book/the-book.html

[11] Mnih, V., et al. (2015). "Human-level control through deep reinforcement learning". *Nature*, 518(7540), 529-533. https://doi.org/10.1038/nature14236

[12] Bakkes, S., Spronck, P., & van den Herik, J. (2012). "Player behaviour modelling for video games". *Entertainment Computing*, 3(3), 85-96. https://doi.org/10.1016/j.entcom.2011.12.001

[13] Kotsia, I., Zafeiriou, S., & Fotopoulos, S. (2013). "Reinforcement learning for the dynamic difficulty adjustment of a game". *IEEE International Conference on Image Processing*, 4249-4253. https://ieeexplore.ieee.org/document/6633602

[14] Sepulveda, G. K., Besoain, F., & Barriga, N. A. (2019). "A Deep Reinforcement Learning Approach for Dynamic Difficulty Adjustment". *IEEE CHILEAN Conference on Electrical, Electronics Engineering,*

Information and Communication Technologies (CHILECON), 1-6. https://ieeexplore.ieee.org/document/8971032

[15] Andrade, G., Ramalho, G., Santana, H., & Corruble, V. (2014). "Challenge-Sensitive Action Selection: an Application to Game Difficulty Scaling". *IEEE Transactions on Computational Intelligence and AI in Games*, 6(2), 155-168. https://ieeexplore.ieee.org/document/6932884

[16] Papamichael, I., Pappas, G., Siegel, J. E., & Zorpas, A. A. (2022). "Unified waste metrics: A gamified tool in next-generation strategic planning". *Science of The Total Environment*, 833, 154835. https://doi.org/10.1016/j.scitotenv.2022.154835

[17] Ouahbi, I., Darhmaoui, H., Kaddari, F., & Elachqar, A. (2015). "Learning basic programming concepts by creating games with Scratch". *Procedia - Social and Behavioral Sciences*, 191, 1479-1482. https://doi.org/10.1016/j.sbspro.2015.04.224

[18] Demasi, P., & Cruz, A. J. (2002). "Online coevolution for action games". *International Journal of Intelligent Systems*, 17(11), 1039-1053. https://link.springer.com/article/10.1007/s00500-002-0217-0

[19] Schulman, J., Wolski, F., Dhariwal, P., Radford, A., & Klimov, O. (2017). "Proximal Policy Optimization Algorithms". *arXiv preprint arXiv:1707.06347*. https://arxiv.org/abs/1707.06347

[20] Yannakakis, G. N., & Togelius, J. (2011). "Experience-driven procedural content generation". *IEEE Transactions on Affective Computing*, 2(3), 147-161. https://ieeexplore.ieee.org/document/5756645

[21] Hooshyar, D., Ahmad, R. B., Yousefi, M., Yusop, F. D., & Kim, S. J. (2018). "A flowchart-based intelligent tutoring system for improving problem-solving skills of novice programmers". *Journal of Computer Assisted Learning*, 34(1), 14-25. https://doi.org/10.1111/jcal.12219

[22] Juliani, A., Berges, V., Teng, E., Cohen, A., Harper, K., Elion, C., ... & Lange, D. (2018). "Unity: A General Platform for Intelligent Agents". *arXiv preprint arXiv:1809.02627*. https://arxiv.org/abs/1809.02627