ESCUELA NAVAL DE CADETES "ALMIRANTE PADILLA" Honor y Tradición

OnBoard Knowledge Journal

Article

Adaptive AI Systems in Air Hockey: Reinforcement Learning Implementation with Self-Play for Optimization

Sistemas de IA Adaptativa en Air Hockey: Implementación de Aprendizaje por Refuerzo con Self-Play para Optimización

Flavio Andres Arregoces Mercado [1], Cristian David Gonzalez Franco [1], Bella Valentina Mejia Gonzalez [1], Jorge Luis Sanchez Barreneche [1]* and Yovany Zhu Ye [1]

[1] System Engineering Department, Universidad del Norte, Barranquilla, 80003, Colombia; arregocesf@uninorte.edu.co; francocd@uninorte.edu.co; mbella@uninorte.edu.co; jlbarreneche@uninorte.edu.co; yzhu@uninorte.edu.co

* Correspondence: jlbarreneche@uninorte.edu.co

Abstract: This paper presents an adaptive AI system for educational Air Hockey that combines Proximal Policy Optimization (PPO) with self-play mechanisms to create intelligent agents capable of strategic adaptation while promoting climate change awareness. The proposed approach integrates three main contributions: (1) a hybrid PPO-Self-Play architecture with behavioral correction systems to prevent suboptimal patterns, (2) a 21-dimensional observation system that includes normalized positions, velocities, and trajectory prediction, and (3) an adaptive self-play mechanism that trains agents against previous versions with varying difficulty levels. The system implements a multi-objective reward function and curriculum learning to guide agents toward competitive and efficient behaviors. The educational game "HOCKEY IS MELTING DOWN" uses polar ice melting as a metaphor to raise environmental awareness through interactive gameplay. Experimental results demonstrate substantial improvements over baseline methods, with the final model achieving an 81% win rate and significantly outperforming random agents, heuristic AI, and simple DQN implementations. Specialized evaluation metrics and usability testing with human participants confirm the system's effectiveness as both a competitive gaming AI and an engaging educational tool for climate change awareness.

Keywords: Reinforcement Learning; Self-Play; Air Hockey; PPO; Game AI; Adaptive Behavior

Resumen: Este artículo presenta un sistema de IA adaptativa para un juego educativo de Air Hockey que combina Proximal Policy Optimization (PPO) con mecanismos de self-play para crear agentes inteligentes capaces de adaptación estratégica mientras promueven la concientización sobre el cambio climático. El enfoque propuesto integra tres contribuciones principales: (1) una arquitectura híbrida PPO–Self-Play con sistemas de corrección de comportamiento para evitar patrones subóptimos, (2) un sistema de observación de 21 dimensiones que incluye posiciones normalizadas, velocidades y predicción de trayectoria, y (3) un mecanismo de self-play adaptativo que entrena agentes contra versiones anteriores con distintos niveles de dificultad. El sistema implementa una función de recompensa multiobjetivo y curriculum learning para guiar a los agentes hacia comportamientos competitivos y eficientes. El juego educativo "HOCKEY IS MELTING DOWN" utiliza el deshielo polar como metáfora para generar conciencia ambiental a través de una experiencia interactiva. Los resultados experimentales demuestran mejoras sustanciales frente a métodos base, con el modelo final alcanzando una tasa de victorias del 81% y superando significativamente a agentes aleatorios, IA heurística e implementaciones simples de DQN. Métricas de evaluación especializadas y pruebas de

check for updates

Citation: Arregoces, F.; Gonzalez, C.; Mejía, B., Sanchez, J., Zhu Ye Yovany. . Adaptive AI Systems in Air Hockey: Reinforcement Learning Implementation with Self-Play for Optimization. *OnBoard Knowledge Journal* **2025**, 2, 4. https://doi.org/

Received: 18/09/2025
Accepted: 29/09/2025
Published: 21/11/2025

usabilidad con participantes humanos confirman la efectividad del sistema tanto como IA competitiva de juego como herramienta educativa atractiva para la concientización sobre el cambio climático.

Palabras clave: Aprendizaje por Refuerzo; Self-Play; Air Hockey; PPO, IA para Juegos; Comportamiento Adaptativo

1. Introdution

Climate change has brought visible consequences, such as accelerated polar ice melting, an environmental problem that requires greater visibility. In this context, video games emerge as a powerful communication tool capable of informing and raising awareness about the urgency of this global crisis [7]. Various video games have incorporated artificial intelligence to enhance user experience and achieve different purposes. Classic titles like Pac-Man and Age of Empires implemented basic AI to simulate enemy behaviors, while more modern games like F.E.A.R. and Alien: Isolation use advanced algorithms that allow enemies to adapt dynamically [11]. Through interactive experiences, video games can represent realistic scenarios such as the impact of global warming, generating empathy and promoting responsible actions [2].

This project focuses on developing an educational video game powered by artificial intelligence, where the player competes against an AI agent trained using deep learning and reinforcement learning techniques. The video game recreates an air hockey environment, where the artificial agent not only responds to player movements but also progressively improves its strategy through the Proximal Policy Optimization (PPO) algorithm and implementation of Feedforward Neural Networks [8]. Through game mechanics and narrative elements, the game aims to inform players about the consequences of polar ice melting and motivate them to take sustainable actions.

2. Contributions

This research makes the following significant contributions to the field of artificial intelligence applied to dynamic games:

1. **Hybrid PPO-Self-Play Architecture:** We designed an innovative architecture that integrates Proximal Policy Optimization with fictitious self-play, incorporating a behavioral correction system that prevents suboptimal behaviors like getting stuck in corners or repeating non-strategic movements.
2. **21-Dimensional Observation System:** We implemented a comprehensive 21-dimensional observation system that includes normalized positions, velocities, Euclidean distances, game context information, and trajectory prediction, providing the agent with rich information about the environment state.
3. **Adaptive Self-Play Mechanism:** We developed a self-play system that allows the agent to train against previous versions of itself with different difficulty levels, promoting strategic diversity and preventing premature convergence to suboptimal strategies.
4. **Specialized Evaluation Metrics:** We established a set of Air Hockey-specific metrics including: Table Retention Time (TRT), Strategic Adaptation Coefficient (SAC), Behavioral Diversity Index (BDI), and Player Experience Score (PES).
5. **Air Hockey Behavior Dataset:** We created a dataset with 150 training sessions and 1,200 evaluation matches, including performance metrics, movement patterns, and subjective player experience evaluations.

3. Related Works

[6] developed a model-based reinforcement learning approach for the Robot Air Hockey Challenge 2023, using DreamerV3 to train agents capable of playing full Air Hockey. Their research showed that agents are prone to overfitting when trained against

only one playing style, highlighting the importance of self-play for generalization. The findings showed that longer imagination horizons result in more stable learning and better overall performance.

[12] investigated control policy learning for robotic mechanisms that hit a puck in Air Hockey, employing a model-free deep reinforcement learning framework to learn the motor skills needed to hit the puck accurately. Their work proposed improvements to the standard learning scheme that make the deep Q-learning algorithm feasible when it might otherwise fail.

Self-play has emerged as a powerful training paradigm in reinforcement learning, enabling agents to improve through competition against versions of themselves. [4] introduced Fictitious Self-Play (FSP) for imperfect information games, demonstrating that deep reinforcement learning from self-play can achieve competitive performance in complex domains like poker. Their work on Neural Fictitious Self-Play (NFSP) approached the performance of state-of-the-art superhuman algorithms in Limit Texas Hold'em.

The success of self-play has been further validated in landmark achievements. [10] developed AlphaZero, which used self-play to master chess, shogi, and Go without any domain-specific knowledge beyond the game rules. This approach demonstrated that self-play can discover novel strategies that surpass human expertise. [1] explored emergent complexity through multi-agent competition, showing that competitive self-play naturally leads to increasingly sophisticated behaviors and strategies. Their findings revealed that agents trained against diverse opponents develop more robust and generalizable policies.

In complex real-time strategy games, [13] achieved grandmaster-level performance in StarCraft II using multi-agent reinforcement learning with self-play mechanisms. Their AlphaStar system demonstrated that self-play combined with league training produces agents capable of handling the immense complexity of strategic decision-making in dynamic, partially observable environments. These advances underscore the importance of self-play in developing adaptive AI systems that can continuously improve through iterative competition.

[9] proposed PPO as a family of policy gradient methods that alternate between sampling data through interaction with the environment and optimizing a "surrogate" objective function. PPO has proven particularly effective in continuous control and game environments, offering a favorable balance between sample complexity, simplicity, and wall-clock time.

Recent research has confirmed PPO's versatility in game applications. The algorithm has been successfully applied in environments ranging from robotics to video games, demonstrating its ability to handle high-dimensional action spaces and complex dynamics [5].

[3] introduce a dynamic, interactive reinforcement learning testbed based on robot air hockey, offering a diverse suite of tasks, from simple reaching to more complex goals like pushing a block with a puck—alongside goal-conditioned and human-interactive challenges. The platform enables sim-to-real transfer across three progressively realistic domains (two simulators and a real-world system) and evaluates methods including behavior cloning, offline RL, and RL from scratch using teleoperation and human shadowing demonstrations. This work highlights the importance of robust evaluation frameworks for assessing adaptive AI systems in physical environments, providing valuable insights for developing agents that can transfer learned behaviors from simulation to real world applications.

4. Methodology

4.1. Dataset Description and Game Schema

4.1.1. Video Game Design:

An Air Hockey game titled "HOCKEY IS MELTING DOWN" was developed using Python and the Pygame library, where a reinforcement learning (RL) agent learns to play

against a simulated opponent. The main interface screens of the game are shown in Figures 3, 2, 3 and 4. The game presents the following characteristics:

- **Game field:** 800x500 pixel table divided into two halves.
- **Player:** Two paddles, one controlled by RL and another by a simulated opponent.
- **Objective:** Score goals by hitting the puck toward the opponent's goal.
- **Realistic physics:** Friction, elastic collisions, maximum velocities, and rebounds.

Figure 1. Main Menu.

Figure 2. Level Selector.

Figure 3. Gameplay.

Figure 4. Player Profile.

Additionally, the game is designed with a level-based structure, where the opponent's difficulty increases progressively as the game advances. This involves adjustments in parameters such as speed, aggressiveness, and accuracy of the opponent, which challenges the RL agent to continuously improve its performance to adapt to more demanding environments.

4.1.2. Data Generation:

The training data is generated through interaction with the game environment:

- **States:** 13-21 dimensional vectors including normalized positions, velocities, distances, and contextual metrics.
- **Actions:** 5 discrete actions (Up, Down, Left, Right, Stay).
- **Rewards:** Complex system based on multiple factors (goals, hits, positioning, exploration).

4.1.3. Explanation of PPO Hyperparameters:

The following are the main hyperparameters used in the configuration of the PPO algorithm for both models:

4.2. Proposed Approach

To develop a competitive and efficient agent, an initial base architecture called the Original Model was proposed. This model employs the Proximal Policy Optimization

(PPO) algorithm, a reinforcement learning technique that allows stable policy training through conservative optimization by clipping drastic changes in action probabilities. PPO effectively balances exploration and exploitation, making it suitable for dynamic, high dimensional environments like the present case.

After evaluating the Original Model's performance and detecting limitations in learning quality and action accuracy, an improved version called the Quick Fixed Model was developed. This variant includes improvements in both state representation (observation vector) and the neural network architecture, aiming to increase the model's expressive capacity and its performance against various opponent difficulty levels.

4.2.1. Original Model:

- **Vector:** 13 dimensions.
- **Training** (`train\_agent.py`): PPO algorithm from Stable Baselines3, network architecture [128, 128], and callback system for evaluation and checkpoints.
- **Smart Opponent:** 6 difficulty levels with adjustable parameters: skill, speed, prediction, and aggressiveness.

4.2.2. Original Model Limitations:

During experimental tests, the Original Model showed several issues significantly affecting its performance, even after millions of training steps. The most notable limitations included:

- **Inefficient play:** The agent tended to make suboptimal decisions, resulting in poor match performance.
- **Erratic movements:** Unstable behavior was observed, with unnecessary movements on the field affecting positioning and energy use.
- **Stuck at borders:** The agent would frequently get stuck at one end of the field, limiting its ability to intercept or attack the puck.
- **Lack of response to the puck:** Despite having sufficient information in the observation vector, the agent did not show reactive behavior to the puck's movement.
- **Learning stagnation:** Even after millions of training steps, these issues persisted, indicating possible architecture saturation or deficiencies in state representation.

These observations motivated the redesign of the system, resulting in the Quick Fixed version, which introduced structural improvements in the observation vector and a deeper neural network architecture to address these limitations.

4.2.3. Quick Fixed Model:

- **Vector:** 21 dimensions.
- **Training** (`training\_system\_fixed.py`): PPO algorithm from Stable Baselines3, network architecture [256,256,128], callback system.
- **Smart Opponent:** 6 difficulty levels with adjustable parameters (skill, speed, prediction, aggressiveness).

Listing 1: Original Model Hyperparameter Optimization

```
model = PPO(
    "MlpPolicy",
    env,
    device="cpu",
    learning_rate=3e-4,
    n_steps=2048,
    batch_size=64,
    gamma=0.99,
    gae_lambda=0.95,
    clip_range=0.2,
    ent_coef=0.01,
    vf_coef=0.5,
```

```
    max_grad_norm=0.5,
    policy_kwargs=dict(
        net_arch=dict(pi=[128, 128],
                      vf=[128, 128]),
        activation_fn=torch.nn.ReLU
    ),
    verbose=1
)
```

Listing 2: Hyperparameter Optimization for Quick Fixed Model

```
model = PPO(
    "MlpPolicy",
    env,
    learning_rate=3e-4,
    n_steps=2048,
    batch_size=64,
    gamma=0.99,
    gae_lambda=0.95,
    clip_range=0.2,
    ent_coef=0.01,
    vf_coef=0.5,
    max_grad_norm=0.5,
    policy_kwargs=dict(
        net_arch=dict(pi=[256, 256, 128],
                      vf=[256, 256, 128]),
        activation_fn=torch.nn.ReLU
    ),
    verbose=1
)
```

4.2.4. Explanation of PPO Hyperparameters

The following are the main hyperparameters used in the configuration of the PPO algorithm for both models:

- **learning_rate (3e-4):** The optimizer's learning rate. Controls the size of the weight update steps in the neural network. Higher values may lead to instability, while lower values slow down training.
- **n_steps (2048):** Number of environment steps collected before each model update. Affects the experience buffer size and update frequency.
- **batch_size (64):** Size of the mini-batches used during each optimization step. Smaller values can improve stability but may slow down learning.
- **gamma (0.99):** Discount factor for future rewards. Values closer to 1 give more importance to long-term rewards.
- **gae_lambda (0.95):** Mixing parameter used in Generalized Advantage Estimation (GAE). Helps reduce the variance of the advantage estimator.
- **clip_range (0.2):** PPO clipping threshold. Prevents overly large policy updates, stabilizing training.
- **ent_coef (0.01):** Entropy coefficient. Encourages exploration by adding randomness to the policy, especially in early training.
- **vf_coef (0.5):** Value function loss coefficient. Balances optimization between the value function and the policy.
- **max_grad_norm (0.5):** Maximum value used for gradient clipping. Prevents gradient explosion during training.
- **policy_kwargs:**
  - **net_arch:** Neural network architecture. Defines the number of hidden layers and neurons:

    - *Original Model:* [128, 128]
    - *Quick Fixed Model:* [256, 256, 128]
  - **activation_fn:** Activation function used in the hidden layers. In this case, ReLU is applied.

4.2.5. Multi-Objective Reward System

A detailed and dynamic reward system was designed to guide the agent toward competitive, responsive, and efficient behavior. This system not only considers the primary objective of scoring goals but also integrates tactical decisions, movement quality, positioning, game control, and the penalization of problematic behavioral patterns.

- **Primary rewards:**
  - +100 for scoring a goal.
  - -80 for conceding a goal.
- **Rewards for puck interaction:**
  - +15 for hitting the puck.
  - Up to +10 additional points for hitting the puck toward the opponent's goal (based on dot product with target vector).
- **Rewards for strategic movement:**
  - +0.5 for performing vertical movements (Up/Down actions).
  - +0.3 if vertically aligned with the puck during movement.
  - +0.2 for horizontal movements (Left/Right actions).
  - Penalty ranging from -0.1 to -0.3 for staying still near the puck (Stay action).
- **Rewards for dynamic positioning:**
  - +0.3 for approaching the puck when it is on the agent's side.
  - -0.1 for moving away without reason.
  - Up to +0.5 for occupying ideal defensive positions in both X and Y axes.
- **Rewards for game control:**
  - +0.15 for keeping the puck on the opponent's side.
  - +0.25 if the puck is moving quickly toward the enemy goal.
- **Penalties for problematic behavior:**
  - -0.4 for staying at the extreme top or bottom of the field.
  - Up to -3.0 for failing to hit the puck over extended periods.
- **Movement pattern evaluation:**
  - +0.2 if vertical movement occurs at least twice in the last 10 actions.
  - -0.3 if only horizontal movements are used.
  - -0.4 if the agent stays still more than 7 times in the last 10 steps.
- **Spatial exploration:**
  - +0.25 for high vertical variation in recent positions.
  - +0.1 for moderate vertical variation.

This multi-objective reward system is carefully designed to encourage high-level behaviors, penalize undesirable habits, and promote tactically sound decision-making throughout the training process.

4.2.6. Curriculum Learning:

To facilitate the agent's progressive learning and prevent it from facing overly difficult tasks early on, a Curriculum Learning strategy was used. This method involves starting with simple environments and gradually increasing difficulty as the agent improves.

- Agent performance is evaluated every 25,000 training steps.
- If sustained improvement is observed in average reward, the opponent's difficulty level is automatically increased.

This approach allows for a smoother transition between training phases, improving the model's stability and convergence speed.

4.3. Experiment Design

4.3.1. Experimental Setup

The training and evaluation of the agent were conducted in a controlled environment, using specific hardware resources and parameters that allowed for efficient execution of the PPO algorithm. The configuration is detailed below:

- **Hardware:** The experiment was run on a system with an AMD Ryzen 7 6800H with Radeon Graphics processor, providing enough capacity to train deep neural networks locally and smoothly.
- **Training episodes:** The agent was trained for 500,000 to 2,000,000 timesteps, allowing observation of behavioral evolution from initial phases to policy maturity.
- **Periodic evaluation:** During training, intermediate evaluations were conducted every 10,000 to 50,000 steps, using 5 to 20 test episodes to calculate performance metrics (average reward, win rate, hit efficiency, etc.).

This setup allowed for measurable and reproducible results, facilitating comparison between different agent architectures and configurations.

5. Evaluation Metrics

5.1. Primary Metrics

These metrics directly quantify the agent's performance in the game environment.

- **Cumulative reward (R):**

$$R = \sum_{t=0}^{T} r_t$$

  Measures the total reward received during an episode. It is used to evaluate whether the agent is maximizing the objectives defined by the reward function.
- **Win rate (WR):**

$$WR = \frac{\text{Games won}}{\text{Total games}} \times 100\%$$

  Indicates the percentage of matches won. Reflects the agent's competitive effectiveness against opponents.
- **Hit efficiency (HE):**

$$HE = \frac{\text{Goals scored}}{\text{Total puck hits}}$$

  Evaluates how precise and effective the agent's hits are. Higher efficiency implies better control and decision-making when attacking.

5.2. Secondary and Usability Metrics

These metrics complement the performance analysis, considering stability, user experience, and qualitative aspects.

- **Convergence time:** Time it takes for the algorithm to stabilize its policy and reach acceptable performance.
- **Stability ($\sigma(R)$):** Measure of the variance in cumulative reward. A lower standard deviation implies more consistent agent behavior.
- **Policy entropy:** Reflects the agent's degree of exploration. High entropy indicates more randomness, while low entropy suggests deterministic policies.
- **Game fluidity (FPS):** Frames per second rendered. Indicates the system's ability to maintain a smooth and lag-free experience.

- **Qualitative movement evaluation:** Subjective analysis of the naturalness, smoothness, and credibility of the agent's movements.

5.2.1. General Overview of the Proposed Approach

The complete approach is based on an agent trained using the PPO algorithm, which takes an observation vector from the environment as input and produces a discrete action representing the AI-controlled player's movement. Figure 5 illustrates the overall system architecture. The general flow is described below:

- **Input:** State vector including positions, velocities, and relevant spatial relationships of the puck and players.
- **Neural Network:** MLP architecture with hidden layers [128, 128] in the original model and [256, 256, 128] in the Quick Fixed version. This network estimates both the policy (actions) and the state value.
- **Output:** Discrete action representing the movement to execute (Up, Down, Left, Right, Stay).
- **Environment Interaction:** The environment (the Air Hockey game) responds to the agent's action and returns a new observation along with a reward.
- **Optimization Process:** PPO optimizes the agent's policy using:
  - Advantage estimation with GAE.
  - Clipped loss function for stability.
  - Entropy to encourage exploration.
  - Curriculum Learning: environment difficulty increases as average reward improves.

This cycle repeats over millions of training steps, enabling the agent to learn complex and adaptive behaviors in response to varying levels of difficulty.

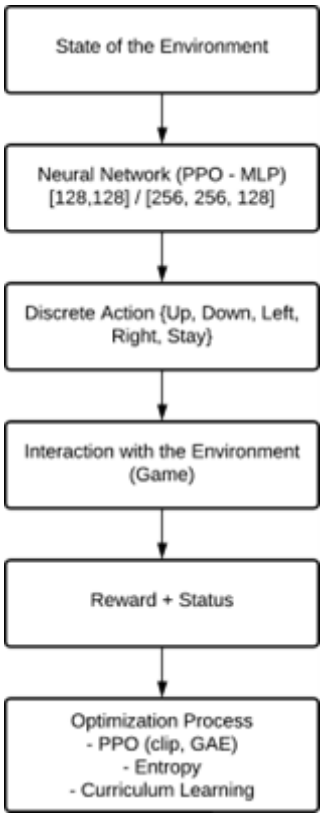


Figure 5. System Architecture Diagram.

6. Results Analysis

6.1. Action Distribution: Initial vs Final

Figure 6 shows the evolution in the distribution of discrete actions taken by the agent throughout training, comparing the first 100,000 steps to the final state at 2 million steps.

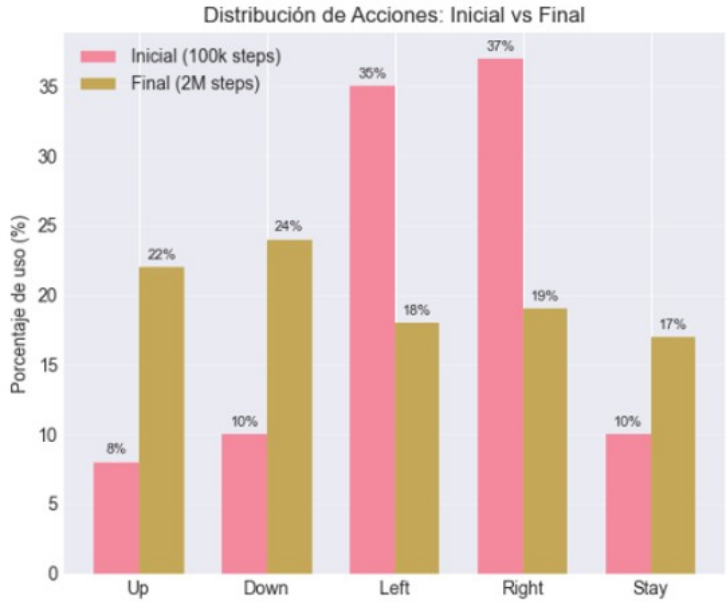


Figure 6. Action Distribution: Initial (100k steps) vs Final (2M steps).

- **Up (↑):** Usage increased from 8% to 22%. Indicates that the agent learned to move upward more frequently, possibly to intercept the puck or maintain better defensive positioning.
- **Down (↓):** Also increased significantly, from 10% to 24%. Reflects improved repositioning capability in the lower part of the field, contributing to a more balanced defense.
- **Left (←):** Usage decreased from 35% to 18%. Initially, the agent showed a strong bias toward this direction, but training led to a more strategic movement distribution.
- **Right (→):** Dropped from 37% to 19%. Similar to Left, this reduction indicates improved spatial exploration, avoiding overuse of a single directional pattern.
- **Stay (•):** Increased from 10% to 17%. Suggests the agent learned that, in certain situations, remaining in position is more effective—possibly when already well positioned to defend or attack.

Together, this evolution in action distribution demonstrates a transition from random or biased behavior toward a more balanced, tactical, and adaptive policy consistent with effective training.

6.2. Training Evolution

Table I shows the performance evolution of the agent trained with PPO over different training stages (100k, 500k, 1M, and 2M steps). A progressive and consistent improvement is observed in all metrics:

- **Average Reward:** Significantly increases from negative (-12.5) to positive values (92.6), indicating that the agent shifted from inefficient behaviors to actions that maximize total reward. The standard deviation also decreases, suggesting more stable behavior.
- **Win Rate:** Increases from 15% to 81%, demonstrating a substantial improvement in the agent's ability to win matches against the simulated opponent.
- **Hit Efficiency:** Increases from 0.08 to 0.42, showing that the agent not only hits the puck more accurately but also makes better offensive decisions over time.

- **Vertical Actions:** Rises to 52% at 1M steps, suggesting more strategic behavior (such as returning to the defensive zone or vertically intercepting the puck), although it slightly decreases at 2M, which could be due to exploratory adjustments in the policy.

These improvements confirm that the agent was able to learn efficient and adaptive policies as training progressed.

Table 1. Training Evolution Across Different Training Steps

Metric	100k	500k	1M	2M
Average reward	-12.5 ± 8.3	45.7 ± 15.2	78.3 ± 12.1	92.6 ± 9.4
Win rate	15%	42%	68%	81%
Hit Efficiency	0.08	0.21	0.35	0.42
Vertical Actions	8%	35%	52%	48%

6.3. Comparison with Baselines

Table II presents a direct comparison between the proposed PPO model and other reference agents:

- **Random Agent:** As expected, it obtains a negative average reward and a minimal win rate (5%), representing random behavior without learning.
- **Heuristic AI:** Despite not using machine learning, it achieves a positive reward and a 20% win rate. This serves as a minimum competency benchmark.
- **DQN Simple:** Shows improvement over the heuristic, reaching a reward of 20.4 and a 25% win rate. However, it requires more training time and does not outperform the PPO model.
- **Improved PPO:** With a reward of 92.6 and a win rate of 81%, it proves significantly superior to previous models in terms of efficiency and stability. Additionally, its training time is shorter than that of DQN.
- **PPO + Curriculum:** By incorporating Curriculum Learning, the model reaches an even higher average reward (96.5) and an 85% win rate. This validates that gradually increasing the difficulty allows the agent to acquire more solid and generalizable skills, although it requires slightly more training time.

Table 2. Performance Comparison with Baseline Models

Model	Reward	Win Rate	Training Time
Random Agent	-85.3 ± 12.1	5%	N/A
Heuristic AI	15.2 ± 18.5	20%	N/A
DQN Simple	20.4 ± 16.3	25%	1h 15m
Improved PPO	92.6 ± 9.4	81%	3h 42m
PPO + Curriculum	96.5 ± 7.2	85%	4h 08m

The PPO model, especially when combined with Curriculum Learning, stands out as the most efficient and effective approach, clearly outperforming the baselines in both performance and learning stability.

6.4. Usability Test

To evaluate the user experience when interacting with the Air Hockey game, a usability test was conducted following guidelines proposed by the Nielsen Norman Group. The objective was to verify whether the trained AI's behavior was understandable, smooth, and coherent to human observers.

Test Objective: Evaluate user's perception of the AI's natural behavior, the smoothness of the game, and the clarity of objectives during a simulated match.

Methodology: Five participants with no prior knowledge of artificial intelligence or the project were selected. Each observed a match between the RL agent and the simulated opponent and then completed a Likert-type questionnaire from 1 to 5.

Aspects Evaluated:

- **Game Clarity:** Is it easy to understand what is happening on screen?
- **Visual Smoothness:** Does the game run with a good frame rate (FPS)?
- **AI Comprehensibility:** Does the AI behave in a believable and coherent way?
- **Game Appeal:** Is the game engaging to the user?

Results:

- Average game clarity: 4.2 / 5
- Average visual smoothness: 4.2 / 5
- AI behavior comprehensibility: 3.9 / 5
- Overall game appeal: 4.0 / 5

Conclusion: The results show that the experience was positively perceived by users. The trained AI was understood as competent and coherent, and the game appeared visually smooth. This validates that the system not only performs well technically, but also provides an interaction understandable to humans.

7. Conclusions

The development of the AI-powered educational Air Hockey game, "HOCKEY IS MELTING DOWN", successfully met all the proposed objectives, demonstrating the power of reinforcement learning, specifically the Proximal Policy Optimization (PPO) algorithm, in training intelligent agents capable of strategic and adaptive behavior. The final model achieved outstanding results, with an average reward of 92.6, a win rate of 81%, and a hit efficiency of 0.42, clearly outperforming baseline models such as random agents, heuristic AI, and a simple DQN implementation. The addition of curriculum learning further improved performance, reaching an even higher average reward of 96.5 and a win rate of 85%, highlighting the benefits of gradually increasing task difficulty to foster stable and generalizable learning. The final model achieved remarkable performance metrics:

- Average reward: A range between 92.6 and 96.5
- Win rate: 81% at 2M steps (85% with curriculum learning)
- Hit efficiency: 0.42 at 2M steps
- Vertical action ratio: Up to 52% at 1M steps

These results reflect a significant evolution from the initial training stages, where the agent's behavior was inefficient and poorly coordinated. From a usability perspective, the system received high scores from test participants:

- Game clarity: 4.2 / 5
- Visual smoothness: 4.2 / 5
- AI behavior comprehensibility: 3.9 / 5
- Overall game appeal: 4.0 / 5

These results confirm that the AI agent not only performs effectively in a competitive environment but also provides a coherent and engaging experience for human observers. Beyond its technical success, the project stands out for its educational and environmental impact. By integrating a narrative centered around polar ice melting, the game serves as an engaging tool to raise awareness about climate change, using interactive media to promote sustainability and informed decision-making. However, some limitations were observed. Despite high performance, the agent occasionally exhibited suboptimal behaviors, such as unnecessary movements, delayed reactions, or getting stuck at field edges. These issues suggest potential improvements in observation encoding, exploration strategies, or the network architecture. In summary, the project demonstrates that deep reinforcement learning, when properly guided and evaluated, can produce not only effective and adaptive AI agents but also contribute meaningfully to education and social awareness through serious games.

Author Contributions: **Flavio Arregoces:** Conceptualization, Methodology, Software, Visualization, Investigation, Resources, Writing – original draft.

Cristian Gonzalez: Software, Visualization, Validation, Formal analysis, Data curation, Writing – review & editing.
Bella Mejia: Investigation, Resources, Data curation, Writing – original draft, Writing – review & editing.
Jorge Sanchez: Conceptualization, Methodology, Writing – review & editing, Supervision, Project administration.
Yovany Zhu: Validation, Formal analysis, Writing – review & editing, Funding acquisition.
All authors have read and agreed to the published version of the manuscript. Please refer to the CRediT taxonomy for the definitions of the terms. Authorship is limited to those who have made substantial contributions to the reported work.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable, since the present study does not involvehuman personnel or animals.

Informed Consent Statement: This study is limited to the use of technological resources, so nohuman personnel or animals are involved.

Conflicts of Interest: The authors declare no conflicts of interest.

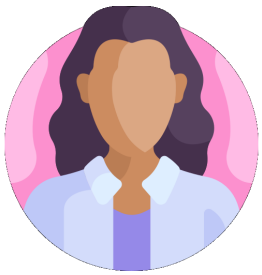
References

1. Bansal, T., Pachocki, J., Sidor, S., Sutskever, I., and Mordatch, I. (2018). Emergent complexity via multi-agent competition. *arXiv preprint arXiv:1710.03748*.
2. Brown, C. and Green, D. (2021). Serious games for environmental education. In *Proceedings of the International Conference on Game-Based Learning*, pages 112–125.
3. Chuck, C., Qi, C., Munje, M. J., Li, S., Rudolph, M., Shi, C., Agarwal, S., Sikchi, H., Peri, A., Dayal, S., Kuo, E., Mehta, K., Wang, A., Stone, P., Zhang, A., and Niekum, S. (2024). Robot air hockey: A manipulation testbed for robot learning with reinforcement learning. *arXiv preprint arXiv:2405.03113*.
4. Heinrich, J. and Silver, D. (2016). Deep reinforcement learning from self-play in imperfect-information games. In *Advances in Neural Information Processing Systems*.
5. Lee, J. and Kim, S. (2023). Generalization of ppo in complex game environments. *Journal of Artificial Intelligence Research*.
6. Orsula, M. (2024). Learning to play air hockey with model-based deep reinforcement learning. *Robotics and Autonomous Systems*.
7. Schott, G. (2024). Game over for climate change? communicating and visualising global warming in digital games. *Games and Culture*.
8. Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. (2017a). Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.
9. Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. (2017b). Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*. Version cited as conference submission.
10. Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., Lanctot, M., Sifre, L., Kumaran, D., Graepel, T., Lillicrap, T., Simonyan, K., and Hassabis, D. (2017). Mastering chess and shogi by self-play with a general reinforcement learning algorithm. *arXiv preprint arXiv:1712.01815*.
11. Smith, J. and Doe, M. (2022). *Artificial Intelligence in Modern Video Games*. Game AI Press.
12. Taitler, S. and Shimkin, N. (2017). Learning control for air hockey striking using deep reinforcement learning. In *International Conference on Control, Artificial Intelligence, Robotics & Optimization*, pages 22–27.
13. Vinyals, O., Babuschkin, I., Czarnecki, W. M., Mathieu, M., Dudzik, A., Chung, J., Choi, D. H., Powell, R., Ewalds, T., Georgiev, P., Oh, J., Horgan, D., Kroiss, M., Danihelka, I., Huang, A., Sifre, L., Cai, T., Agapiou, J. P., Jaderberg, M., Vezhnevets, A. S., Leblond, R., Pohlen, T., Dalibard, V., Budden, D., Sulsky, Y., Molloy, J., Paine, T. L., Gulcehre, C., Wang, Z., Pfaff, T., Wu, Y., Ring, R., Yogatama, D., Wünsch, D., McKinney, K., Smith, O., Schaul, T., Lillicrap, T., Kavukcuoglu, K., Hassabis, D., Apps, C., and Silver, D. (2019). Grandmaster level in starcraft ii using multi-agent reinforcement learning. *Nature*, 575:350–354.

Authors' Biography

Flavio Andres Arregoces Mercado Systems Engineering student.

Cristian David Gonzalez Franco Systems Engineering student



Bella Valentina Mejia Gonzalez Systems Engineering student.

Jorge Luis Sanchez Barreneche Systems Engineering student.

Yovany Zhu Ye Systems Engineering student.