Journal of Information System Research (JOSH)
Volume 6, No. 3, April 2025, pp 1715−1723
ISSN 2686-228X (media online)
https://ejurnal.seminar-id.com/index.php/josh/
DOI 10.47065/josh.v6i3.7008

Implementasi Unity-Gymnasium sebagai Alternatif Metode Reinforcement Learning dalam Pengembangan Environment Sliding Puzzle menggunakan Game Engine Unity

Agus Julpian Alwi*, Chandra Kusuma Dewa

Fakultas Teknologi Industri, Informatika, Universitas Islam Indonesia, Yogyakarta
Jalan Kaliurang Km. 14,5, Krawitan, Umbulmartani, Ngemplak, Sleman, Yogyakarta, Indonesia
Email: [1,*]agus.alwi@students.uii.ac.id, [2]chandra.kusuma@uii.ac.id
Email Penulis Korespondensi: agus.alwi@students.uii.ac.id
Submitted: **18/02/2025;** Accepted: **14/04/2025;** Published: **15/04/2025**

Abstrak−Artificial intelligence (AI) memiliki peran penting dalam industri game, penerapannya biasa ditemukan pada non-player character (NPC) hingga pada procedural content generation (PCG). Salah satu metode pengembangan AI yang populer adalah reinforcement learning (RL). Unity sebagai salah satu game engine yang paling mendominasi, memiliki framework RL-nya sendiri yang bernama Unity ML-Agents Toolkit. Unity dengan kemampuannya dalam mensimulasikan environment yang realistis, memungkinkan Unity ML-Agents Toolkit melakukan pengembangan serta pengujian RL agent dengan berbagai skenario yang kompleks. Namun, Unity ML-Agents Toolkit memiliki batasan, yaitu opsi algoritma yang sangat terbatas. Penelitian ini bertujuan untuk mengenalkan metode alternatif dalam menerapkan reinforcement learning pada game engine Unity, serta mengatasi batasan yang ada pada Unity ML-Agents Toolkit. Metode yang diusulkan adalah Unity-Gymnasium, yaitu mengintegrasikan Unity dengan Gymnasium, yang diuji dengan mengembangkan environment sliding puzzle. Hasil dari pernelitian ini menunjukkan bahwa metode Unity-Gymnasium dapat berfungsi dengan baik serta memungkinkan untuk mengakses total 38 algortima RL berbeda dari berbagai RL library yang kompatibel dengan Gymnasium seperti Stable Baseline 3, CleanRL, Tianshou, Ray Rllib, dan Dopamine, jumlah ini jauh lebih banyak jika dibandingkan dengan Unity ML-Agents Toolkit yang hanya memiliki lima opsi algoritma.

Kata Kunci: Reinforcement Learning; Algoritma; Unity; Gymnasium; Environment; Sliding Puzzle

Abstract−Artificial intelligence (AI) plays an important role in the gaming industry, its application commonly found in non-player character (NPC) and procedural content generation (PCG). One of the popular methods for developing AI is reinforcement learning (RL). Unity, as one of the most dominant game engines, has its own RL framework called Unity ML-Agents Toolkit. Unity with its capabilities to simulate realistic environments. allowing Unity ML-Agents Toolkit to develop and test RL agents in various complex scenarios. However, Unity ML-Agents Toolkit only has limited RL algorithms. This study aims to introduce an alternative method for implementing reinforcement learning in Unity, while addressing the Unity ML-Agents Toolkit's limitations. The proposed method is Unity-Gymnasium, which integrates Unity with Gymnasium, and tested by developing a sliding puzzle environment. The result of this study demonstrates that the Unity-Gymnasium method works well and allows access to total 38 different RL algorithms from various RL libraries that compatible with Gymnasium, such as Stable Baseline 3, CleanRL, Tianshou, Ray Rllib, and Dopamine, this number is significantly higher compared to Unity ML-Agents Toolkit which only offer five RL algorithm options.

Keywords: Reinforcement Learning; Algorithm; Unity; Gymnasium; Environment; Sliding Puzzle

1. PENDAHULUAN

Artificial intelligence (AI) memiliki peran penting dalam industri game, penerapannya yang biasa ditemukan yaitu pada non-player character (NPC) yang dapat bergerak atau bertingkah laku tanpa dikontrol secara langsung oleh player [1], gameplay balancing untuk menyesuaikan tingkat kesulitan permainan secara dinamis, hingga procedural content generation (PCG) untuk menciptakan konten yang bervariasi [2]. Selain itu, AI juga diterapkan dalam upaya mengumpulkan data (data mining) dari para player, data-data yang diperoleh untuk meningkatkan model monetisasi pada game yang dikembangkan [3].

Sejak tahun 1951 hingga 2020, berbagai metode dan algoritma telah dikembangkan dan digunakan untuk mengembangkan AI pada berbagai jenis game, mulai dari game dengan gameplay turn-based maupun real-time, game yang melibatkan satu player maupun lebih, hingga game yang bersifat kooperatif maupun kompetitif. Salah satu metode yang paling populer untuk mengembangkan AI pada game yaitu reinforcement learning (RL) [4].

Reinforcement learning yang merupakan salah satu sub-bidang dari machine learning adalah metode pengembangan AI yang terinspirasi dari cara manusia maupun hewan belajar berinteraksi dengan lingkungan [5]. Terdapat enam komponen yang ada pada reinforcement learning, yaitu agent, environment, action, state, reward, dan policy. Agent berlaku sebagai otak untuk mengambil keputusan, environment adalah segala hal yang dapat berinteraksi dengan agent, action adalah segala hal yang dapat dilakukan agent di environment, state adalah representasi dari keadaan atau situasi di environment, reward adalah penilaian yang diterima agent setelah melakukan action berdasarkan state yang dihasilkan, dan policy adalah ketetapan atau strategi yang dimiliki agent dalam bertingkah laku atau untuk mengambil keputusan. Proses reinforcement learning dilakukan dengan metode trial-and-error, agent memilih action yang akan dilakukan yang kemudian akan merubah state dari environment dan mendapatkan reward, proses ini dilakukan berulang-ulang hingga menghasilkan policy dan reward yang maksimal [5], [6].

Journal of Information System Research (JOSH)
Volume 6, No. 3, April 2025, pp 1715−1723
ISSN 2686-228X (media online)
https://ejurnal.seminar-id.com/index.php/josh/
DOI 10.47065/josh.v6i3.7008

Game engine adalah program yang menyediakan berbagai komponen siap pakai untuk mempermudah dan mempercepat proses pengembangan game [7], [8]. Komponen-komponen yang tersedia pada umumnya, yaitu rendering engine, animation module, audio module, dan network module. Rendering engine berfungsi untuk menampilkan gambar ke layar, animation module berfungsi untuk menjalankan animasi, audio module berfungsi untuk memutar musik latar ataupun efek suara, dan network module berfungsi untuk menghubungkan game ke internet sehingga game dapat dimainkan secara multiplayer [7]. Selain industri game, game engine juga digunakan di industri film, arsitektur, manufaktur, hingga artificial intelligence secara umum. Saat ini, pasar game engine didominasi oleh dua perusahaan, yaitu Unity Technology dengan Unity, dan Epic Games dengan Unreal Engine [8].

Unity memiliki RL framework-nya sendiri yang bernama Unity ML-Agents Toolkit atau biasa disingkat ML-Agents. ML-Agents adalah open-source project yang memungkinkan Unity environment berfungsi sebagai RL environment untuk mengembangkan AI. ML-Agents memiliki beberapa opsi algoritma yang bisa digunakan sesuai kebutuhan, diantaranya yaitu Proximal Policy Optimization (PPO), Soft Actor-Critic (SAC), dan Multi-Agent Posthumous Credit Assignment (MA-POCA), yang masing-masing algoritma tersebut dapat digunakan untuk melakukan training single-agent, cooperative multi-agent, dan competitive multi-agent. Selain itu, terdapat juga algoritma Generative Adversarial Imitation Learning (GAIL) dan Behavioral Cloning (BC) yang dapat digunakan untuk melakukan imitation training. Unity dengan kemampuannya dalam mensimulasikan environment yang realistis, memungkinkan ML-Agents digunakan dalam pengembangan dan pengujian terhadap RL agent di berbagai skenario yang kompleks [9]. Meskipun dengan kemampuannya tersebut, ML-Agents memiliki keterbatasan pada opsi algoritma yang bisa digunakan [10].

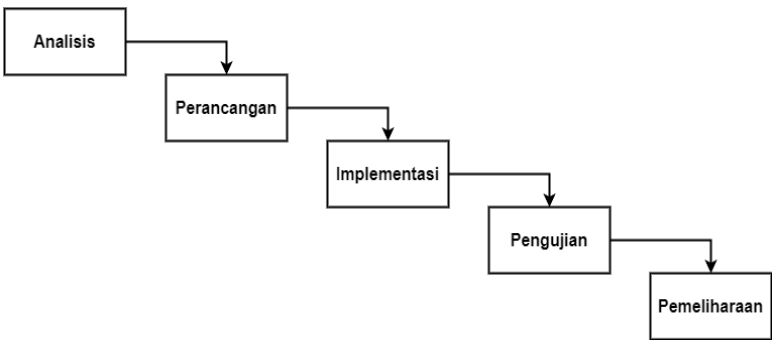
Penelitian ini bertujuan untuk mengenalkan metode alternatif dalam menerapkan reinforcement learning pada game engine Unity, serta mengatasi batasan yang ada pada ML-Agents. Metode yang diusulkan adalah adalah Unity-Gymnasium, yaitu menintegrasikan Unity dengan Gymnasium, metode ini diuji dengan mengembangkan environment sliding puzzle. Gymnasium adalah sebuah library berbasis OpenAI Gym yang menyediakan standarisasi API untuk RL environment yang kompatibel dengan berbagai RL library seperti Stable Baseline3, CleanRL, Tianshou, Ray Rllib, dan Dopamine [11], sehingga memungkinkan lebih banyak opsi algoritma RL yang bisa digunakan.

Sebuah penelitian serupa yang dilakukan oleh Abbas dan Hyun Soo Kang, mengintegrasikan Unity dengan program Python yang memungkainkan implementasi serta pengembangan algoritma reinforcement learning yang fleksibel. Metode tersebut diuji dengan menerapkan evalation-based control pada quadcopter drone [12]. Penelitian lainnya yang dilakukan oleh Yubing Mao dkk, juga mengintegrasikan Unity dengan program Python untuk menerapkan fungsi target-tracking pada autonomous underwater vehicle (AUV) [13]. Selain menggunakan game engine Unity, terdapat penelitian serupa lainnya yang menggunakan game engine Godot. Salah satunya yaitu penelitian yang dilakukan oleh Beeching dkk, mengembangkan RL framework yang mengintegrasikan Godot dengan RLlib dan Stable Baseline [14]. Penelitian lainnya yang dilakukan oleh Ranaweera dan Mahmoud, juga mengembangkan RL framework yang mengintegrasikan Godot dengan framework Python yang mereka kembangkan sendiri [15]. Selain itu, terdapat juga penelitian lainnya yang menggunakan Gymnasium, yaitu pada penelitian yang dilakukan oleh Malagón, Ceberio, dan Lozano, yang mengembangkan RL framework yang dinamai Craftium pada open-source game engine Minetest [16].

2. METODOLOGI PENELITIAN

2.1 Tahapan Penelitian

Waterfall adalah metodologi pengembangan perangkat lunak yang bersifat sistematis, dimana setiap tahapnya harus dikerjakan dengan teliti dan mendetail secara berurutan, hal ini memungkinkan untuk meminimalisir risiko serta memastikan kualitas dari perangkat lunak yang dikembangkan [17]. Meskipun bukan termasuk perangkat lunak, waterfall tetap dipilih sebagai metodologi untuk mengembangkan environment sliding puzzle karena memiliki alur kerja yang jelas.



Gambar 1. Tahapan pada metodologi waterfall

Journal of Information System Research (JOSH)
Volume 6, No. 3, April 2025, pp 1715−1723
ISSN 2686-228X (media online)
https://ejurnal.seminar-id.com/index.php/josh/
DOI 10.47065/josh.v6i3.7008

Gambar 1 menujukkan lima tahapan penelitian dalam metodologi waterfall yang harus dikerjakan secara berurutan, yaitu analisi, perancangan, implementasi, pengujian, dan pemeliharaan. Penjelasan dari setiap tahap dalam mengembangkan environement sliding puzzle adalah sebagai berikut.

a. Pada tahap analisis, dilakukan identifikasi kebutuhan fungsional dan kebutuhan non fungsional yang bertujuan memperoleh ekspektasi terhadap environment sliding puzzle yang akan dikembangkan.
b. Pada tahap perancangan, dilakukan perancangan terhadap environment sliding puzzle berdasarkan kebutuhan-kebuthan yang telah diidentifikasi pada tahap analisis. Rancangan terdiri dari arsitektur environment, desain state, action, dan reward function, serta desain visual sliding puzzle.
c. Pada tahap implementasi, environment sliding puzzle mulai dikembangkan berdasarakan hasil dari rancangan yang telah dibuat dengan memprogram dan mengintegrasikan komponen-komponen yang dibutuhkan, mulai dari algoritma program hingga elemen-elemen visual yang dibutuhkan.
d. Pada tahap pengujian, dilakukan evaluasi terhadap metode Unity-Gymnasium dan environment sliding puzzle yang telah dikembangkan. Hal ini bertujuan untuk memastikan metode dan environment berfungsi dengan baik serta sesuai dengan ekspektasi dan rancangan yang telah ditetapkan sebelumnya.
e. Pada tahap pemeliharaan, dilakukan perbaikan, improvisasi, maupun pembaharuan jika dibutuhkan. Hal bertujuan agar metode Unity-Gymnasium dan environment sliding puzzle tetap optimal dan relevan di masa depan.

2.2 Reinforcement Learning

Reinforcement Learning (RL) adalah salah satu metode pengembangan AI yang merupakan sub-bidang dari machine learning, metode ini terinspirasi dari cara manusia maupun hewan berinteraksi dengan lingkungan [5]. Terdapat enam komponen utama, yaitu sebagai berikut.

a. Agent yang berperan sebagai otak untuk mengambil keputusan.
b. Environment adalah segala hal selain agent yang berinteraksi dengannya.
c. Action adalah segala hal yang dapat dilakukan oleh agent di dalam environment.
d. State adalah representasi dari keadaan atau situasi yang ada di environment.
e. Reward adalah penilaian yang diterima oleh agent berdasarkan state yang dihasilkan setelah melakukan action.
f. Policy adalah ketetapan atau strategi yang dimiliki agent dalam mengambil keputusan.

Gambar 2. Proses Reinforcement Learning

Gambar 2 menunjukkan iterasi proses reinforcement learning dimana agent menerima data state dan reward dari environment, yang kemudian agent menentukan action yang akan dilakukan berdasarkan policy yang dimilikinya, setelah melakukan action, state di environment berubah berdasarkan action yang dilakukan, dan agent menerima data state dan reward terbaru. Proses ini terus dilakukan berulang kali menggunakan metode trial-and-error hingga menghasilkan reward yang maksimal dan policy yang diinginkan [5], [6].

2.3 Unity ML-Agents Toolkit

ML-Agents adalah framework RL yang bersifat open-source yang dimiliki oleh Unity. ML-Agents memungkinkan game atau unity environment berfungsi sebagai RL environment untuk mengembangkan AI dengan lima opsi algoritma RL yang bisa digunakan, yaitu PPO, SAC, MA-POCA, GAIL, dan BC. PPO, SAC, dan MA-POCA masing-masing mendukung metode training single-agent, cooperative multi-agent, dan competitive multi-agent. Sedangkan GAIL dan BC mendukung metode imitation training. Unity dengan kemampuannya untuk mensimulasikan environment yang realistis, memungkinkan ML-Agents melakukan training terhadapt RL agent di berbagai skenario yang kompleks [9].

2.4 Gymnasium

Gymnasium adalah sebuah library open-source yang dibuat berbasis OpenAI Gym yang menawarkan standarisasi API untuk RL environment yang kompatibel dengan berbagai RL library. Beberapa RL library yang kompatibel dengan Gymnasium, yaitu Stable Baseline3, CleanRL, Tianshou, Ray Rllib, dan Dopamine [11].

Tabel 1. Algoritma RL library yang mendukung Gymnasium

Library RL	Algoritma RL
Stable Baseline3	ARS, A2C, CrossQ, DDPG, DQN, HER, PPO, QR-DQN, RecurrentPPO, SAC, TD3, TQC, TRPO, dan Maskable PPO [18].
CleanLR	PPO, DQN, Categorical DQN, SAC, DDPG, TD3, PPG, RND, dan Qdagger [19].

Journal of Information System Research (JOSH)
Volume 6, No. 3, April 2025, pp 1715−1723
ISSN 2686-228X (media online)
https://ejurnal.seminar-id.com/index.php/josh/
DOI 10.47065/josh.v6i3.7008

Library RL	Algoritma RL
Tianshou	DQN, Double DQN, Dueling DQN, Branching DQN, Categorical DQN, Rainbow DQN, QR-DQN, IQN, FQF, PGP, NPG, A2C, TRPO, PPO, DDPG, TD3, SAC, REDQ, Discrete SAC, Imitation, BCQ, CQL, TD3B, Discrete BCQ, Discrete CQL, Discrete CRR, GAIL, PSRL, dan ICM [20].
Ray Rllib	PPO, DQN, Rainbow DQN, SAC, APPO, IMPALA, BC, dan MARWIL [21].
Dopamine	DQN, Categorical DQN, Rainbow DQN, IQN, SAC, dan PPO [22].

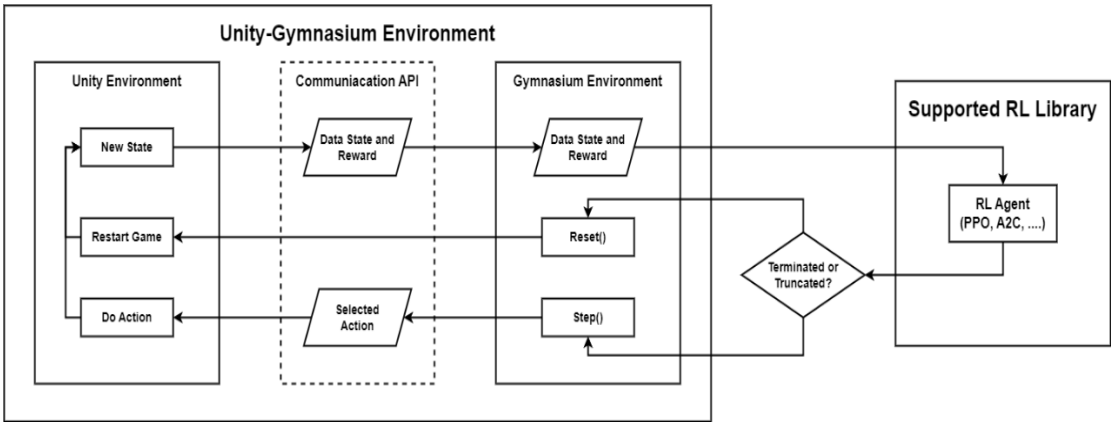
Tabel 1 menyajikan daftar algoritma RL yang tersedia pada masing-masing RL library yang kompatibel dengan Gymnasium. Masing-masing RL library memiliki beberapa algoritma RL yang sama satu sama lain, sehingga terdapat total 38 algoritma RL yang berbeda yang dapat diakses melalui Gymnasium. Sebuah environment Gymnasium (Env) utamanya memiliki komponen-komponen berikut.

a. observation space mendefinisikan struktur data state pada RL environment.
b. action space mendefinisikan action yang digunakan pada step() function.
c. step() function adalah fungsi yang mengeksekusi action di RL environment. Fungsi ini mengembalikan nilai observation (state), reward, terminated, truncated, dan info. Terminate adalah keadaan dimana RL agent berhasil ataupun gagal dalam mencapai goal yang telah ditentukan, sedangkan truncate adalah keadaan yang bersifat time-based, salah satu contohnya adalah jumlah maksimal step. Proses iterasi ini terus dilakukan berulang kali hingga training selesai atau dihentikan.
d. reset() function adalah fungsi yang digunakan untuk mengatur kembali state di RL environment. Fungsi ini mengembalikan nilai observation (state) dan info.

2.5 Unity-Gymnasium

Unity-Gymnasium adalah metode yang diusulkan dalam penelitian ini, yang diharapkan dapat mengatasi batasan yang ada pada ML-Agent, yaitu opsi algoritma RL yang sangat terbatas. Terdapat tiga komponen yang ada pada Unity-Gymnasium, yaitu communication API, unity environment, dan gymnasium environment sebagaimana yang diperlihatkan pada Gambar 2, yang penjelasan adalah sebagai berikut.

a. Communication API berperan sebagai perantara untuk mengirim dan menerima data antara unity environment dengan gymnasium environment.
b. Unity environment berfungsi sebagai RL environment yang berinteraksi dengan RL agent, serta memvisualisasikannya. Environment ini merupakan game atau simulasi tempat terjadinya perubahan state dan perolehan reward, serta tempat RL agent melakukan action.
c. Gymnasium environment berfungsi sebagai API agar unity environment dapat berfungsi sebagai RL environment yang kompatibel dengan berbagai RL library yang kompatibel dengan gymnasium. Gymnasium environment yang dimaksud dalam metode ini adalah custom gymnasium environment yang dibuat dengan membuat module turunan dari gymnasium.Env dengan menyesuaikan keempat komponen utamanya berdasarkan unity environment.



Gambar 3. Struktur dan alur Unity-Gymnasium

Gambar 3 menunjukkan proses yang terjadi pada setiap iterasi training yang dilakukan RL agent. Iterasi dapat berbentuk per-frame apabila game bersifat real-time, atau berbentuk per-turn apabila game bersifat turn-based. Pada setiap iterasinya, data state dan reward dikirim dari unity environement ke RL agent melalui gymnasium environment menggunakan communication API. Data yang telah diterima RL Agent diproses menggunakan algoritma yang dipilih, seperti PPO, A2C, SAC, DQN, dan sebagainya. Setelah data diproses, RL agent akan menentukan action selainjutnya dan mengirimkan perintah untuk melakukan action ke unity environment melalui step() function di gymnasium environment menggunakan communication API. Apabila environment telah memenuhi kondisi terminate atau truncate, maka perintah untuk mengulang kembali game dari

Journal of Information System Research (JOSH)
Volume 6, No. 3, April 2025, pp 1715−1723
ISSN 2686-228X (media online)
https://ejurnal.seminar-id.com/index.php/josh/
DOI 10.47065/josh.v6i3.7008

awal yang akan dikirimkan ke unity environment melalui reset() function di gymnasium environment pada iterasi berikutnya. Dengan mengintegrasikan unity environment dengan gymnasium environment menggunakan communication API, memungkinkan unity environment berfungsi sebagai RL environment yang dapat mengakses 38 algoritma RL dari berbagai RL library yang kompatibel dengan Gymnasium.

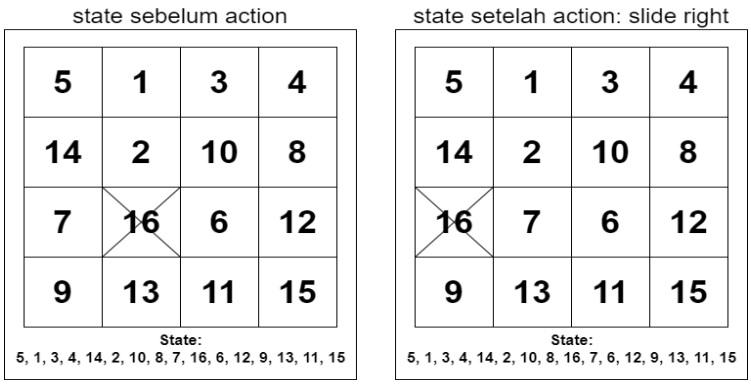
3. HASIL DAN PEMBAHASAN

3.1 Implementasi dan Pengujian Unity-Gymnasium

Metode Unity-Gymnasium dicoba diimplementasikan dan diuji dengan mengembangkan sliding puzzle environment. Proses pengembangan environment tersebut dilakukan menggunakan komputer dengan spesifikasi CPU AMD Rayzen 5 3600, GPU Nvidia RTX 3050 (VRAM 8GB), RAM 16GB (8GB x 2) 2666MHz, dan penyimpanan SSD Adata SX8200 (500GB). Adapun software dan library yang digunakan, yaitu bahasa pemrograman Python versi 3.10.12, game engine Unity 2021.3, library Gymnasium, library Peaceful Pie (https://github.com/hughperkins/peaceful-pie) sebagai communication API, library Stable Baseline3 yang digunakan untuk menguji environment, dan library Tensorboard untuk memvisualisasikan data training yang diperoleh saat pengujian.

3.2.1 Rancangan Environment Sliding Puzzle

Bagian ini membahas rancangan environment sliding puzzle yang meliputi dari desain visual sliding puzzle, state, dan action sebagaimana ditujukkan pada ditujukkan pada Gambar 3, serta rancangan reward function.



Gambar 3. Desain visual sliding puzzle environment.

Gambar 3 menunjukkan gambaran visual sliding puzzle environment. Dapat dilihat bahwa sliding puzzle memiliki layout berbentuk 2D grid, sehingga state dapat direpresentasikan sebagai 2D array. Namun, karena Stable Baseline3 tidak kompatibel dengan data berbentuk 2D array, maka state dikonversi dari 2D array menjadi 1D array. Index array merepresentasikan posisi tile, sedangkan nilai elemen pada index array merepresentasikan tile yang bersangkutan dan nilai elemen terbesar merepresentasikan tile kosong. Dapat dilihat juga bahwa cara action dilakukan adalah dengan menggeser satu tile ke arah tile kosong sesuai arah yang ditentukan action. Terdapat total empat discrete action, yaitu slide right, slide left, slide up, dan slide down.

Sliding puzzle environment ini memiliki beberapa aturan terkait action untuk megoptimisasi output yang dihasilkan akan reward fuction. Action dianggap valid jika menyebabkan perubahan pada posisi tile kosong. Sebaliknya, action dianggap tidak valid jika tidak menyebabkan perubahan pada posisi tile kosong, hal ini bisa terjadi ketika mencoba menggeser tile yang tidak ada, contohnya dengan melakukan action slide left ketika posisi tile kosong berada di ujung sebelah kanan, dimana pada posisi tersebut tidak ada tile yg berada di sebelah kanan tile kosong untuk digeser ke arah kiri, sehingga action tersebut dianggap tidak valid. Reward function yang dibuat dengan menggunakan formula berikut.

$$r = \begin{cases} P - 2D, if\ action\ valid \\ -1 \quad , if\ action\ invalid \\ +10 \quad , if\ puzzle\ solved \end{cases} \tag{1}$$

$$P = \frac{a}{b} \tag{2}$$

$$D(x,y) = \frac{\Sigma_{i,j}|x_i - x_j| + |y_i - y_j|}{max(x+y) - min(x+y) + max(x-y) - min(x-y)} \tag{3}$$

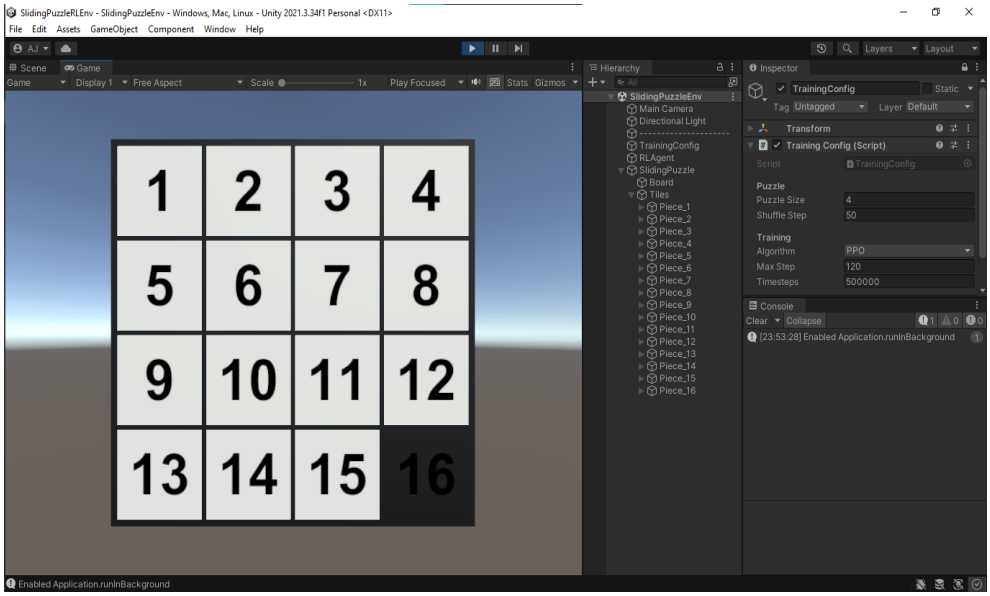
Formula tersebut dibuat dengan sedikit memodifikasi formula pada SPGym [23]. Modifikasi yang dilakukan yaitu dengan menambahkan progres penyelesaian puzzle dan mengakumulasikannya dengan dua kali nilai progres dalam manhattan distance. Penjelasan dari formula yang dibuat adalah sebagai berikut.

Journal of Information System Research (JOSH)
Volume 6, No. 3, April 2025, pp 1715−1723
ISSN 2686-228X (media online)
https://ejurnal.seminar-id.com/index.php/josh/
DOI 10.47065/josh.v6i3.7008

a. Nilai r merepresentasikan nilai reward yang diperoleh berdasarkan tiga kondisi setelah melakukan action. Pada Kondisi valid, nilai reward adalah -1 hingga 1. -1 merepresentasikan state terburuk, sedangkan 1 yang merepresentasikan state terbaik, yaitu puzzle berhasil diselesaikan.
b. P merepresentasikan progres penyelesaian puzzle dengan berdasarkan jumlah tile yang benar, nilai P diperoleh dari perbandingan a dengan b. a merepresentasikan jumlah tile yang berada pada posisi yang benar, sedangkan b merepresentasikan jumlah keseluruhan tile yang ada, sehingga nilai P yang bisa diperoleh adalah 0 hingga 1.
c. D merepresentasikan progres penyelesaian puzzle berdasarkan pada manhattan distance terkini, nilai D diperoleh berdasarkan perbandingan total manhattan distance terkini dari setiap tile dengan maksimal manhattan distance yang mungkin terjadi, sehingga nilai D yang bisa diperoleh adalah 0 hingga 1. (x, y) merepresentasikan ukuran grid puzzle, sedangkan i dan j merepresentasikan posisi tile puzzle.

3.2.2 Game Sliding Puzzle (Unity Environment)

Unity environment berupa game sliding puzzle tanpa adanya kendali dari player, hal ini bertujuan untuk memastikan tidak adanya interaksi antara manusia dengan environment yang dapat mengganggu proses training.



Gambar 4. Unity environment

Gambar 4 menujukkan unity environment yang berupa game sliding puzze. Bagian tab game sebelah kiri gambar merupakan sliding puzzle, dan bagian tab inspector sebelah kanan gambar merupakan konfigurasi training yang ada pada object TrainingConfig. Konfigurasi training mempengaruhi ukuran puzzle dan performa training Penjelasan dari setiap parameter dapat dilihat pada Tabel 2.

Tabel 2. Konfigurasi training

Parameter	Keterangan
Puzzle Size	Ukuran puzzle Size x Size.
Shuffle Step	Jumlah langkah pengacakan puzzle.
Algorithm	Opsi algoritma RL.
Max Step	Maksimum step per episode.
Timesteps	Total step untuk menyelesaikan training.

3.2.4 SlidingPuzzleEnv (Custom Gymnasium Environment)

Module custom gymnasium environment dibuat dengan nama SlidingPuzzleEnv, dengan komponen envrionment yang disesuaikan adalah sebagai berikut.

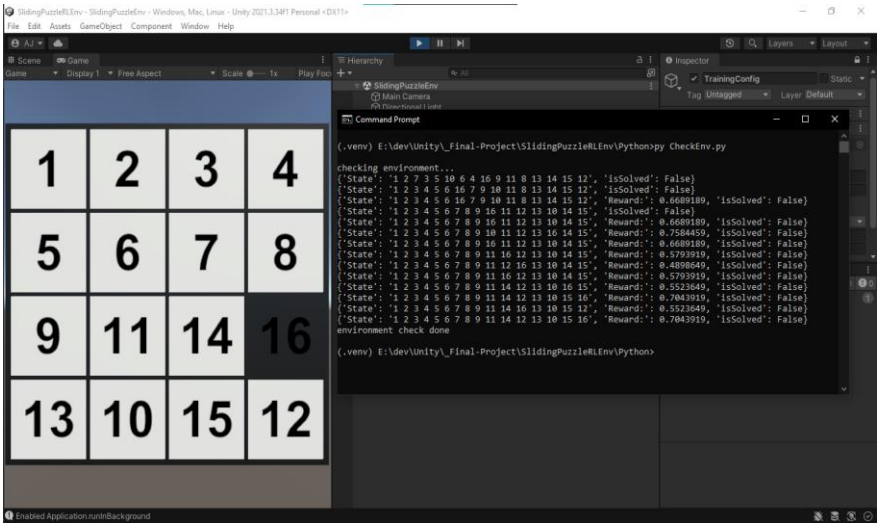
a. observation space bersturktur Box dengan nilai low adalah 1, nilai high serta shape adalah puzzle size dikali puzzle size, dan dengan dtype int32.
b. action space adalah discrete dengan nilai 4 yang merepresentasikan ada empat jenis action.
c. Pada step() function, kondisi terminate adalah true apabila puzzle berhasil diselesaikan, dan kondisi truncate adalah true apabila jumlah step terkini mencapai maksimal step.

3.2.5 Pengujian Environment

Pengujian terhadap environment sliding puzzle dilakukan dalam dua tahap, yaitu melakukan cek validasi terhadap SlidingPuzzleEnv dan melakukan percobaan training. Kedua tahap pegujian ini dilakukan dengan menggunakan RL library Stable Baseline3.

Journal of Information System Research (JOSH)
Volume 6, No. 3, April 2025, pp 1715−1723
ISSN 2686-228X (media online)
https://ejurnal.seminar-id.com/index.php/josh/
DOI 10.47065/josh.v6i3.7008

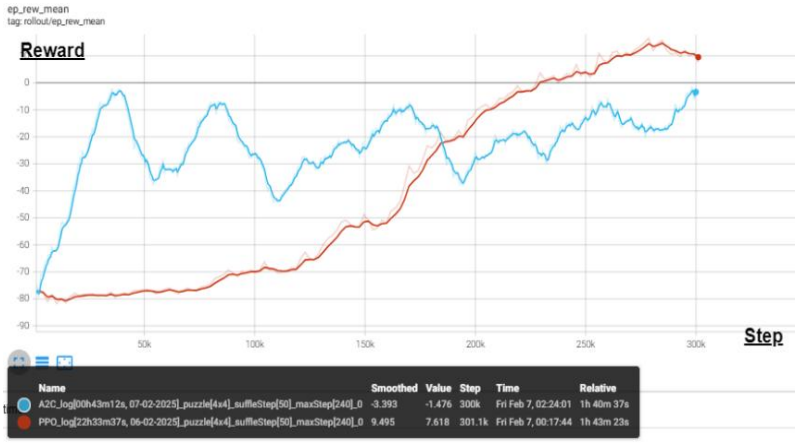
a. Cek validasi environment
Tahap pertama yaitu cek validasi SlidingPuzzleEnv menggunakan fungsi check_env() yang ada pada Stable Baseline3. Pengecekan ini bertujuan untuk memastikan bahwa SlidingPuzzleEnv sudah memenuhi standar dan struktur yang benar, sehingga environment dapat digunakan untuk melakukan training.



Gambar 5. Hasil cek validasi SlidingPuzzleEnv

Gambar 5 memperlihatkan cek validasi environment pada command promt dengan menjalankan script CheckEnv.py. Pada gambar tersebut tidak terlihat ada error muncul, hal ini menandakan bahwa SlidingPuzzleEnv sudah memenuhi syarat dan siap untuk digunakan. Selain itu, terlihat juga adanya log data state dan reward yang menandakan bahwa SlidingPuzzleEnv berhasil menerima data dari unity environment.

b. Percobaan Training
Tahap kedua yaitu percobaan training yang dilakukan sebanyak dua kali, dengan percobaan pertama dilakukan dengan menggunakan algoritma PPO, dan percobaan kedua dilakukan dengan menggunakan algoritma A2C. Konfigurasi training yang dipakai, yaitu puzzle size 4, shuffle step 50, max step 240, dan timestep 300,000.



Gambar 6. Grafik hasil percobaan training

Gambar 6 memperlihatkan hasil percobaan training yang dilakukan menggunakan algoritma A2C (grafik biru) dan PPO (grafik merah). Dapat dilihat bahwa kedua algoritma memiliki peforma training yang berbeda. Algoritma A2C terlihat mendapatkan performa yang tinggi dalam waktu singkat, namun peforma tersebut menjadi tidak stabil seiring berjalannya waktu, bahkan tidak ada peningkatan yang berarti pada akhir training. Sedangkan algortima PPO mendapatkan peningkatan performa secara perlahan namun stabil, bahkan mendapatkan hasil akhir yang lebih baik ketimbang algoritma A2C.

3.2 Temuan

Dari implementasi dan pengujian yang telah dilakukan, diketahui bahwa dengan metode Unity-Gymnasium, RL agent dapat berinteraksi dengan unity environment melalui gymnasium dengan menerima data state dan reward, mengirim perintah action, serta mendapatkan peningkatan performa dalam menyelesaikan sliding puzzle selama proses training berlangsung bahkan dengan menggunakan algoritma RL yang tidak tersedia pada Unity ML-Agents, yaitu A2C.

Journal of Information System Research (JOSH)
Volume 6, No. 3, April 2025, pp 1715−1723
ISSN 2686-228X (media online)
https://ejurnal.seminar-id.com/index.php/josh/
DOI 10.47065/josh.v6i3.7008

4. KESIMPULAN

Penelitian ini bertujuan mengenalkan metode alternatif dalam menerapkan reinforcement learning pada game engine Unity untuk mengatasi batasan yang ada pada Unity ML-Agents, yaitu opsi algoritma RL yang sangat terbatas. Metode yang diusulkan adalah Unity-Gymnasium, yaitu mengintegrasikan Unity dengan Gymnasium, metode ini memungkinkan unity environment berfungsi sebagai RL environment yang kompatibel dengan Stable Baseline3, CleanRL, Tianshou, Ray Rllib, dan Dopamine, sehingga memungkinkan untuk mengakses total 38 algoritma RL berbeda, jumlah ini jauh lebih banyak jika dibandingkan Unity ML-Agents yang hanya memiliki opsi lima algoritma. Metode Unity-Gymnasium ini diuji fungsionalitasnnya dengan mengembangkan sliding puzzle environment, yang kemudian environement tersebut diuji menggunakan Stable Baseline3. Hasil pengujian terhadap sliding puzzle environment menujukkan bahwa RL agent dapat berinteraksi dengan unity environment melalui gymnasium dengan menerima data state dan reward, mengirim perintah action, serta mendapatkan peningkatan performa dalam menyelesaikan sliding puzzle selama proses training berlangsung bahkan dengan menggunakan algoritma RL yang tidak tersedia pada Unity ML-Agents, yaitu A2C. Sehingga hasil dari penelitian ini menujukkan bahwa metode Unity-Gymnasium dapat berfungsi dengan baik dan dapat mengkases jauh lebih banyak algoritma RL dibandingkan dengan Unity ML-Agent. Namun, dalam penelitian ini, pengujian terhadap metode Unity-Gymnasium masih terbatas pada environment yang dan mekanisme permainan yang sederhana, percobaan training yang dilakukan masih pada metode single-agent, serta masih belum dicoba menggunakan algoritma RL selain A2C dan PPO. Selain itu, masih belum diketahui performa dari metode ini dibandingkan dengan Unity ML-Agents maupun metode pada penelitian yang serupa. Sehingga diperlukan pengujian lebih lanjut dengan menggunakan environment dan skenario yang lebih kompleks, mengujinya menggunakan algoritma selain A2C dan PPO, dan membandingkan performanya dengan Unity ML-Agents maupun metode lainnya.

REFERENCES

[1] M. Ranjitha, K. Nathan, dan L. Joseph, “Artificial Intelligence Algorithms and Techniques in the computation of Player-Adaptive Games,” dalam Journal of Physics: Conference Series, Institute of Physics Publishing, Jan 2020. doi: 10.1088/1742-6596/1427/1/012006.

[2] C. Hu, Y. Zhao, Z. Wang, H. Du, dan J. Liu, “Games for Artificial Intelligence Research: A Review and Perspectives,” IEEE Transactions on Artificial Intelligence, vol. 5, no. 12, hlm. 5949–5968, Des 2024, doi: 10.1109/TAI.2024.3410935.

[3] A. Filipović, “The Role Of Artificial Intelligence In Video Game Development,” KULTURA POLISA, vol. 20, no. 3, hlm. 50–67, Nov 2023, doi: 10.51738/kpolisa2023.20.3r.50f.

[4] Y. Lu dan W. Li, “Techniques and Paradigms in Modern Game AI Systems,” Algorithms, vol. 15, no. 8, hlm. 282, Agu 2022, doi: 10.3390/a15080282.

[5] P. Almeida, V. Carvalho, dan A. Simões, “Reinforcement Learning Applied to AI Bots in First-Person Shooters: A Systematic Review,” 1 Juli 2023, Multidisciplinary Digital Publishing Institute (MDPI). doi: 10.3390/a16070323.

[6] Z. Zhang, “Basic things about reinforcement learning,” Applied and Computational Engineering, vol. 6, no. 1, hlm. 199–203, Jun 2023, doi: 10.54254/2755-2721/6/20230788.

[7] A. Barczak dan H. Woźniak, “Comparative Study on Game Engines,” Studia Informatica, no. 23, hlm. 5–24, Des 2020, doi: 10.34739/si.2019.23.01.

[8] A. Jungherr dan D. B. Schlarb, “The Extended Reach of Game Engine Companies: How Companies Like Epic Games and Unity Technologies Provide Platforms for Extended Reality Applications and the Metaverse,” Social Media and Society, vol. 8, no. 2, Apr 2022, doi: 10.1177/20563051221107641.

[9] Kushagra, A. Sajjan, S. Jaiswal, H. Mishra, dan S. Singh, “Development and Evaluation of Autonomous Parking System Utilising Reinforcement Learning Agents Within Unity3D Environment,” International Journal For Multidisciplinary Research, vol. 6, no. 3, Jun 2024, doi: 10.36948/ijfmr.2024.v06i03.21878.

[10] Y. Savid, R. Mahmoudi, R. Maskeliūnas, dan R. Damaševičius, “Simulated Autonomous Driving Using Reinforcement Learning: A Comparative Study on Unity’s ML-Agents Framework,” Information (Switzerland), vol. 14, no. 5, Mei 2023, doi: 10.3390/info14050290.

[11] M. Towers dkk., “Gymnasium: A Standard Interface for Reinforcement Learning Environments,” ArXiv, Jul 2024, doi: 10.48550/arXiv.2407.17032.

[12] M. A. B. Abbass dan H.-S. Kang, “Drone Elevation Control Based on Python-Unity Integrated Framework for Reinforcement Learning Applications,” Drones, vol. 7, no. 4, hlm. 225, Mar 2023, doi: 10.3390/drones7040225.

[13] Y. Mao, F. Gao, Q. Zhang, dan Z. Yang, “An AUV Target-Tracking Method Combining Imitation Learning and Deep Reinforcement Learning,” J Mar Sci Eng, vol. 10, no. 3, Mar 2022, doi: 10.3390/jmse10030383.

[14] E. Beeching, J. Debangoye, O. Simonin, dan C. Wolf, “Godot Reinforcement Learning Agents,” ArXiv, Des 2021, doi: 10.48550/arXiv.2112.03636.

[15] M. Ranaweera dan Q. H. Mahmoud, “Deep Reinforcement Learning with Godot Game Engine,” Electronics (Switzerland), vol. 13, no. 5, Mar 2024, doi: 10.3390/electronics13050985.

[16] M. Malagón, J. Ceberio, dan J. A. Lozano, “Craftium: An Extensible Framework for Creating Reinforcement Learning Environments,” ArXiv, Jul 2024, doi: 10.48550/arXiv.2407.03969.

[17] Y. Akbar dan A. A. Albahy, “Implementasi Game 2D Edukasi Pengetahuan Islam untuk Remaja Menggunakan Unity,” Jurnal JTIK (Jurnal Teknologi Informasi dan Komunikasi), vol. 9, no. 1, hlm. 120–129, Nov 2024, doi: 10.35870/jtik.v9i1.3019.

[18] “RL Algorithms — Stable Baselines3 2.0.0a7 documentation.", Stable Baseline3. [Daring]. Tersedia: https://stable-baselines3.readthedocs.io/en/master/guide/algos.html. [Diakses: 6 Februari 2025].

Journal of Information System Research (JOSH)
Volume 6, No. 3, April 2025, pp 1715−1723
ISSN 2686-228X (media online)
https://ejurnal.seminar-id.com/index.php/josh/
DOI 10.47065/josh.v6i3.7008

[19] "Overview - CleanRL.", CleanRL. [Daring]. Tersedia: https://docs.cleanrl.dev/rl-algorithms/overview/. [Diakses: 6 Februari 2025].
[20] "Welcome to Tianshou! — Tianshou Documentation.", Tianshou. [Daring]. Tersedia: https://tianshou.org/en/stable/. [Diakses: 6 Februari 2025].
[21] "Algorithms — Ray 2.0.0.,", Ray Rllib. [Daring]. Tersedia: https://docs.ray.io/en/latest/rllib/rllib-algorithms.html. [Diakses: 6 Februari 2025].
[22] "Deepmind Github Repository", Google Deepmind. [Daring]. Tersedia: https://github.com/google/dopamine/. [Diakses: 6 Februari 2025].
[23] B. L. M. de Oliveira, M. L. da Luz, B. Brandão, L. G. B. Martins, T. W. de L. Soares, dan L. C. Melo, "Sliding Puzzles Gym: A Scalable Benchmark for State Representation in Visual Reinforcement Learning," ArXiv, Okt 2024, doi: 10.48550/arXiv.2410.14038.