DOI: 10.25140/2411-5363-2024-4(38)-163-171
УДК 004.89

Віталій Леонідович Левківський[1], Галина Вікторівна Марчук[2], Олександр Сергійович Москалик[3]

[1]доктор філософії з інженерії програмного забезпечення, доцент кафедри комп'ютерних наук
Державний університет «Житомирська політехніка» (Житомир, Україна)
E-mail: levkivskyy@ztu.edu.ua. **ORCID:** https://orcid.org/0000-0002-1643-0895. **ResearcherID:** GYU-9377-2022

[2]старший викладач кафедри комп'ютерних наук
Державний університет «Житомирська політехніка» (Житомир, Україна)
E-mail: pzs_mgv@ztu.edu.ua. **ORCID:** https://orcid.org/0000-0003-2954-1057. **ResearcherID:** AAD-7514-2022

[3]магістрант кафедри комп'ютерних наук
Державний університет «Житомирська політехніка» (Житомир, Україна)
E-mail: alexundercover191@gmail.com. **ORCID:** https://orcid.org/0009-0005-6555-1545

МОДЕЛЬ ІГРОВОГО ШТУЧНОГО ІНТЕЛЕКТУ ДЛЯ НЕКЕРОВАНОГО ГРАВЦЕМ ПЕРСОНАЖУ

Штучний інтелект у відеоіграх відіграє ключову роль у формуванні ігрового процесу, а його створення завжди було комплексним та складним завданням. Навчання некерованих персонажів є ключовим завданням для отримання захопливих та реалістичних ігрових середовищ. Основною метою навчання NPC було забезпечення здатності до ефективного переслідування рухомої цілі. Результати експерименту підтвердили, що розроблена модель успішно навчається та приймає оптимальні рішення в умовах ігрового середовища. Дослідження довело ефективність використання методів машинного навчання у створенні ігрового штучного інтелекту. Перспективними напрямами подальших досліджень є інтеграція методів машинного навчання для надання неігровим персонажам додаткових здібностей, які сприятимуть більш глибокому зануренню гравців у ігровий процес.

Ключові слова: *штучний інтелект; гра; некерований гравцем персонаж; машинне навчання; модель; навчання з підкріпленням; NPC.*

Рис.: 9. Бібл.: 11.

Актуальність теми дослідження. Завдяки досягненням у сфері машинного навчання, розробники відеоігор отримали потужні інструменти для створення більш інтерактивних та реалістичних ігрових світів. Від розробки алгоритмів для штучного інтелекту некерованих гравцем персонажів (Non Playable Characters – NPC) до застосування технологій масштабування зображень та аналізу поведінки гравців – машинне навчання знайшло широке застосування у сучасних іграх, суттєво спростивши та прискоривши процес розробки. На сьогодні машинне навчання - це революційна технологія, яка швидко розвивається та охоплює всі сфери розробки додатків. Ігровий штучний інтелект створений за допомогою методів машинного навчання дозволяє створити різноманітні та цікаві взаємодії гравця із NPC.

Постановка проблеми. Відеоігри зазнають стрімкої еволюції, а штучний інтелект стає рушійною силою цього прогресу. Завдяки машинному навчанню розробники отримують доступ до нових інструментів, що дозволяють творити захопливі та динамічні ігрові світи. Штучний інтелект дає можливість розширювати межі ігрового дизайну, створюючи NPC з глибоким характером, реалістичною поведінкою та непередбачуваними реакціями. Це робить взаємодію з ними більш природною та захопливою, а ігровий процес – більш динамічним. Однак основною проблемою на сьогодні є обмежений досвід використання навчання з підкріпленням для створення інтелектуальних неігрових об'єктів.

Аналіз останніх досліджень і публікацій. Нині методи машинного навчання застосовуються в багатьох процесах розробки відеоігор. Автор статті [1] описує, як можна використати методи машинного навчання для генерації рівнів, керування NPC, зокрема, для генерації потрібних анімацій персонажів і розстановки світла на локації, що значно спрощує роботу розробникам та підвищує якість гри. Одним із перспективних

© В. Л. Левківський, Г. В. Марчук, О. С. Москалик, 2024

напрямів використання методів машинного навчання у сфері відеоігор є розробка моделі штучного інтелекту для некерованих гравцем персонажів. У статті [2] автор розглядає методи машинного навчання і підходи до створення штучного інтелекту у грі. Також він проводить експеримент зі створення моделі машинного навчання для керування NPC у грі про виживання, а в кінці наводить оцінку ефективності створеної моделі та порівнює її з традиційним способом створення штучного інтелекту у грі. Експеримент показав, що створена модель за допомогою методів машинного навчання має підвищену ефективність у порівнянні з класичною моделлю. Автори робіт [3; 4] демонструють успішну інтеграцію методів машинного навчання у відеоігри. Було розглянуто методи та способи їх застосувань, визначення нейронної мережі і те, як вона працює та вирішує проблему отримання та використання наборів даних для навчання моделі. У роботах описано повний цикл розробки - від планування можливостей моделі до її інтеграції у гру та збір даних про ефективність моделей.

У статті [5] проведено дослідження різних алгоритмів машинного навчання, включаючи традиційні методи, для створення інтелектуальних агентів для гри в шахи. За результатами аналізу було встановлено, що найкраще для цього підходить метод глибинного навчання. У роботі використано штучний інтелект AlphaZero.

Стаття [6] присвячена дослідженню методів машинного навчання з вчителем. Автор статті детально описує процес машинного навчання, приділяючи основну увагу навчанню із вчителем та розглядаючи такі алгоритми підходу, як дерево рішень, лінійну регресію, наївний Баєс та логістичну регресію. У статті [7] описано підхід самонавчання (англ. Self-play) який також можна застосувати у відеоіграх. Цей метод дозволяє пришвидшити та оптимізувати навчання моделі, оскільки вона тренується сама із собою.

Мета статті. Метою роботи є дослідження можливості застосування методу навчання з підкріпленням для створення моделі ігрового штучного інтелекту некерованого гравцем персонажу, який може самонавчатись та переслідувати ігрових персонажів.

Виклад основного матеріалу. Перші алгоритми та моделі нейронних мереж з’явились у 1950-х роках. Артур Семюел створив програму на основі алгоритмів, які здатні самонавчатись. Семюел дав визначення терміну «машинне навчання». Згідно з цим визначенням, машинне навчання це «область досліджень розробки машин, які не є заздалегідь запрограмованим» [8]. Машинне навчання є розділом штучного інтелекту, його мета – передбачити результат та ухвалювати рішення на основі вхідних даних. Навчання з підкріпленням почало активно розвиватись протягом останніх років. У моделях машинного навчання існує два типи параметрів – параметрами моделі та гіперпараметри. Параметри моделі можуть бути ініціалізовані та оновлені в процесі навчання. Гіперпараметри (рис. 1) повинні бути ініціалізовані перед навчанням моделі, оскільки вони визначають архітектуру моделі машинного навчання [9].

У процесі розробки ігор активно використовується машинне навчання для автоматизації рутинних завдань, наприклад для створення локацій, генерації текстур, моделей тощо. Особливої популярності набуває ігровий штучний інтелект для некерованих гравцем персонажів. Машинне навчання стає дедалі потужнішим інструментом для розробників ігор, що прагнуть створити більш реалістичних та захопливих неігрових персонажів. Замість того, щоб вручну кодувати кожен аспект поведінки NPC, машинне навчання дозволяє їм «вчитися» у потрібному середовищі, що робить їх більш гнучкими та адаптивними до ігрового середовища.

```
behaviors:
  Basic:
    trainer_type: ppo
    hyperparameters:
      batch_size: 32
      buffer_size: 256
      learning_rate: 0.0003
      beta: 0.005
      epsilon: 0.2
      lambd: 0.95
      num_epoch: 3
      learning_rate_schedule: linear
    network_settings:
      normalize: false
      hidden_units: 20
      num_layers: 1
      vis_encode_type: simple
    reward_signals:
      extrinsic:
        gamma: 0.9
        strength: 1.0
    keep_checkpoints: 5
    max_steps: 500000
    time_horizon: 3
    summary_freq: 2000
```

Рис. 1. Гіперпарметри інструментарію ML-Agents

Джерело: розроблено авторами.

NVIDIA ACE For Games – це спеціальний сервіс для створення моделей штучного інтелекту, який має на меті перетворити ігри, наділивши неігрових персонажів інтелектом за допомогою взаємодії з природною мовою на основі штучного інтелекту [10]. Іншим прикладом використання машинного навчання є DLSS (deep learning super sampling) є технологія покращення зображення, розроблена NVIDIA для комп'ютерних ігор. За допомогою алгоритмів глибинного навчання DLSS може масштабувати зображення, збільшуючи роздільну здатність і збільшуючи частоту кадрів та зменшуючи навантаження на відеокарту.

Створення штучного інтелекту для некерованого гравцем персонажу.

Для навчання моделі машинного навчання та впровадження її у гру можна використовувати рушій Unity, який має модуль ML-Agents. За допомогою його можна проводити навчання моделі методом навчання з підкріпленням на основі PyTorch. Unity Machine Learning Agents Toolkit – це проєкт із відкритим вихідним кодом, який дозволяє використовувати ігри та симуляції як середовище для навчання інтелектуальних агентів [11].

Навчання моделі починають з простого завдання і поступово його ускладнюють. Воно складається з епізодів, де агент робить кроки з метою отримати нагороду. Спочатку агент діє випадково, досліджуючи навколишнє середовище та формуючи початкові зв'язки. Його мета – навчитися виконувати завдання, яке полягає у досягненні динамічної цілі. Для цього він буде використовувати підкріплення, отримуючи нагороди за кожен успішний крок. Як показано на рисунках 2 і 3, агент, що має набір скриптів, більшість з яких надаються інструментарієм ML-Agents, може навчатися на основі досвіду.

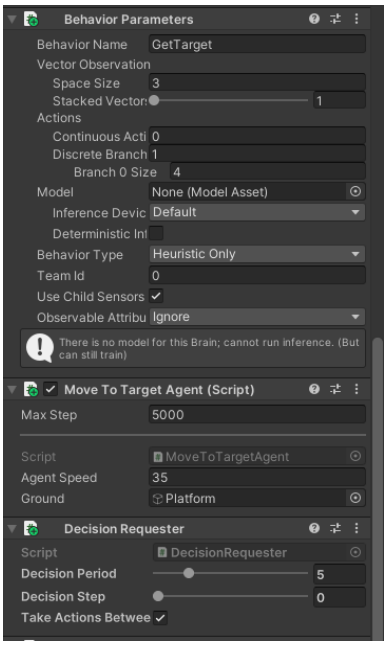

Рис. 2. Компоненти додані до агента
Джерело: розроблено авторами.

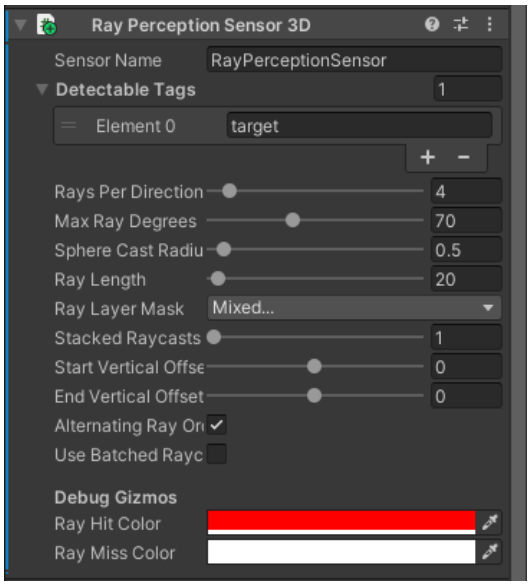

Рис. 3. Сенсор для розпізнавання об'єктів
Джерело: розроблено авторами.

Одним із головних компонентів є Behavior Parameters (рис. 2), де можна налаштувати назву поведінки, кількість значень для спостереження, тип поведінки (проєктування нейронного мозку, або ручне управління) та тип (цілі числа, або числа з плаваючою комою) і кількість дій агента. Компонент, який зображений на рисунку 3 (Ray Perception Sensor 3D) потрібен для того, аби агент «спостерігав» і з часом асоціював об'єкти.

Наступним кроком потрібно дати можливість агенту пересуватись та отримувати нагороду або штрафи за поведінку. Для цього було створено скрипт MoveToTargetAgent (рис. 4).

```
public override void CollectObservations(VectorSensor sensor)
{
    sensor.AddObservation(transform.localPosition);
}

16 references
public override void OnActionReceived(ActionBuffers actions)
{
    var dirToGo = Vector3.zero;
    var rotateDir = Vector3.zero;

    var action = actions.DiscreteActions[0];

    switch (action)
    {
        case 1:
            dirToGo = transform.forward * 1f;
            break;
        case 2:
            dirToGo = transform.forward * -1f;
            break;
        case 3:
            rotateDir = transform.up * 1f;
            break;
        case 4:
            rotateDir = transform.up * -1f;
            break;
        case 5:
            dirToGo = transform.right * -0.75f;
            break;
        case 6:
            dirToGo = transform.right * 0.75f;
            break;
    }
    transform.Rotate(rotateDir, Time.fixedDeltaTime * 200f);
    m_AgentRb.AddForce(dirToGo * agentSpeed * Time.deltaTime,
        ForceMode.VelocityChange);
    AddReward(-1f / MaxStep);
}
```

Рис. 4. Скрипт MoveToTargetAgent
Джерело: розроблено авторами.

Навчання ML-Agents – це ітеративний процес, на початку якого він випадковим чином отримує дані про свою позицію і починає рух. Залежно від того, чи досягнуто ціль він отримує нагороду або штраф. Після виконання дії процес знову повторюється. Також для того щоб модель була здатна адаптуватись, потрібно кожного епізоду випадковим чином розставляти ціль, перепони та самого агента (рис. 5).

```
1 reference
private Vector3 GetRandomPosition()
{
    return new Vector3(Random.Range(-8, 8), 0f, Random.Range(-8, 0));
}

Unity Message | 0 references
public void OnTriggerEnter(Collider other)
{
    if (other.CompareTag("target"))
    {
        SetReward(5f);
        EndEpisode();
    }
}

Unity Message | 0 references
private void Update()
{
    if (transform.position.y < -1)
    {
        SetReward(-1f);
        EndEpisode();
    }
}
```

Рис. 5. Отримання нагороди

Джерело: розроблено авторами.

```
behaviors:
  GetTarget:
    trainer_type: ppo
    hyperparameters:
      batch_size: 1024
      buffer_size: 10240
      learning_rate: 0.0003
      beta: 0.005
      epsilon: 0.2
      lambd: 0.95
      num_epoch: 3
      learning_rate_schedule: linear
    network_settings:
      normalize: false
      hidden_units: 256
      num_layers: 1
      vis_encode_type: simple
    reward_signals:
      extrinsic:
        gamma: 0.99
        strength: 1.0
    keep_checkpoints: 5
    max_steps: 2000000
    time_horizon: 64
    summary_freq: 10000
```

Рис. 6. Налаштований конфігураційний файл гіперпараметрів

Джерело: розроблено авторами.

На початку тренування агент діє випадково, поступово він набуває досвіду. На рис. 7 можна спостерігати, що на перших ітераціях він має від’ємну нагороду (Mean reward), а далі вона стає позитивною.

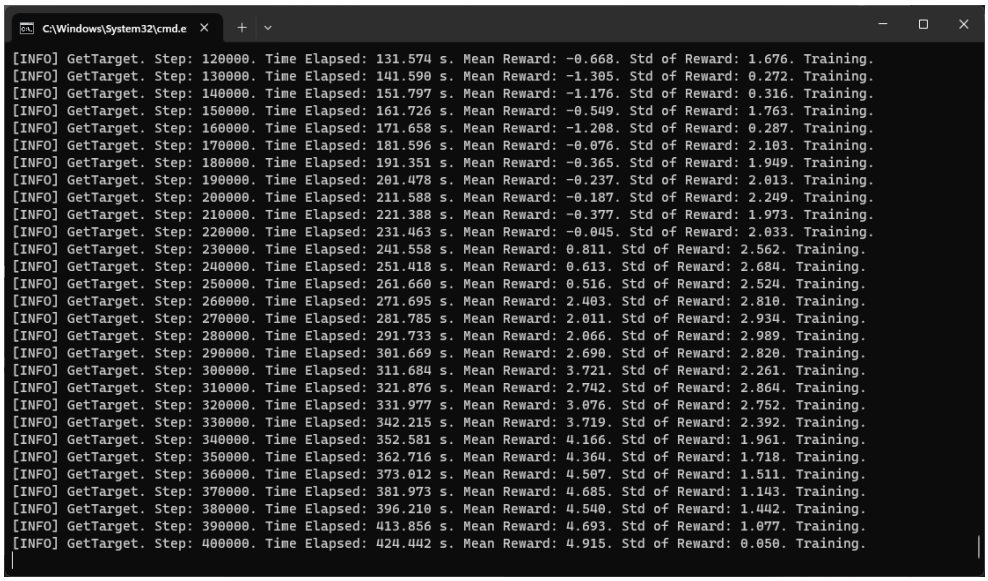

Рис. 7. Результати тренування ML-Agents

Джерело: розроблено авторами.

Після закінчення навчання було отримано модель у форматі «.onnx», яку можна інтегрувати у систему управління поведінкою агента.

Для аналізу процесу навчання представлено візуалізацію у вигляді графіків: кумулятивна винагорода, довжина епізоду. На рисунку 8 продемонстровано процес накопичення сумарної нагороди. Як видно з графіка, агент навчився слідувати за ціллю і поступово отримував винагороду з кожним проміжком кроків. Ця крива винагород доводить ефективність алгоритму навчання.

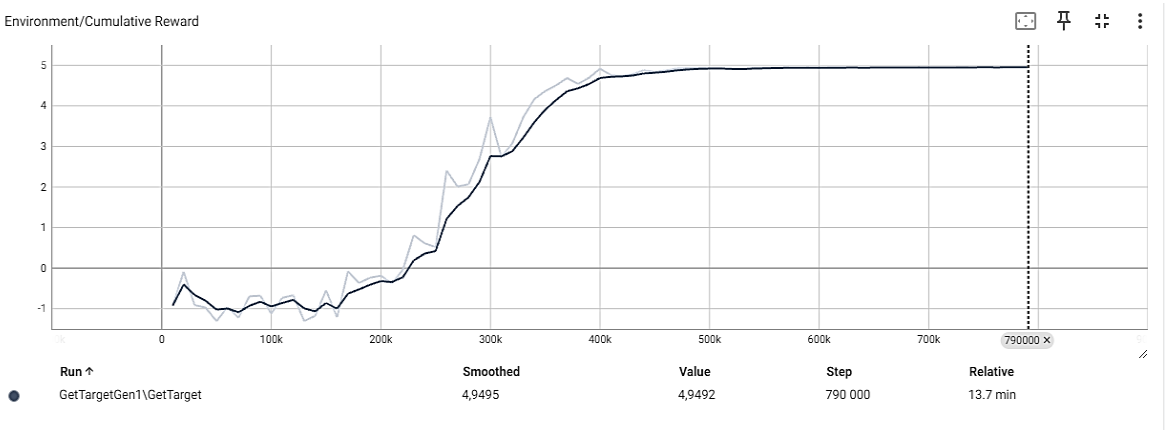

Рис. 8. Графік кумулятивної винагороди

Джерело: розроблено авторами.

На рис. 9 можна побачити, що на початку навчання агент здійснює значно більше дій для досягнення мети, що свідчить про його недостатню навченість. Однак зі збільшенням кількості епізодів, агент поступово вчиться ефективніше виконувати завдання, що призводить до скорочення довжини епізоду досягнення цілі. Це означає, що агент стає дедалі більш компетентним у досягненні поставленої мети.

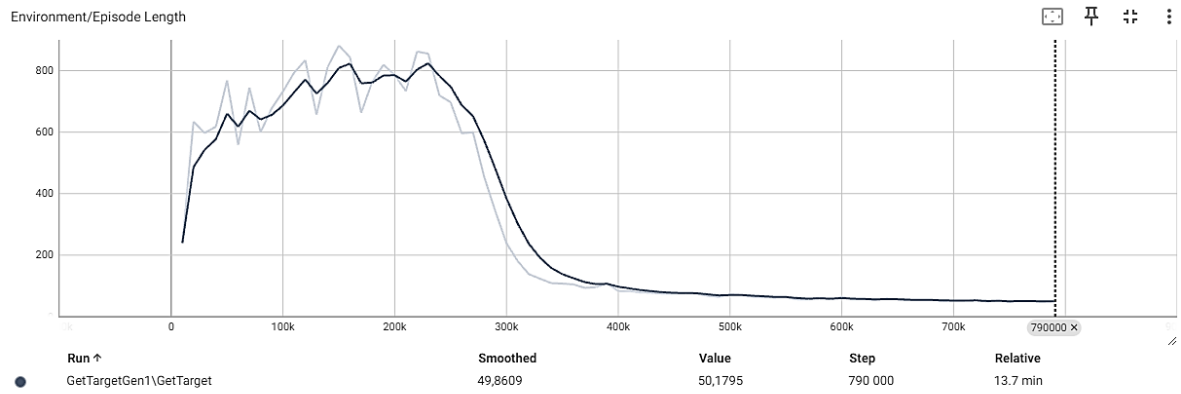

Рис. 9. Графік довжини епізоду

Джерело: розроблено авторами.

Висновки. З розвитком технологій штучного інтелекту відкриваються ширші можливості створення інтелектуальних персонажів для комп'ютерних ігор, з більш реалістичною поведінкою, що сприяє глибшому зануренню в гру.

У роботі було проведено експеримент з навчанням некерованого гравцем персонажа. Для цього було обрано метод навчання з підкріпленням. Персонаж навчається шляхом правильних спроб та помилок, отримуючи винагороду або покарання. Головною метою навчання NPC було набуття досвіду переслідування рухомої цілі. Результати експерименту показали, що модель здатна на успішне навчання та ухвалення оптимальних рішень в ігровому середовищі.

Дослідження показало, що машинне навчання є потужним інструментом для розробки ефективного ігрового штучного інтелекту. Навчання з підкріплення відкриває нові можливості для розробки більш динамічних та непередбачуваних ігрових сценаріїв. Перспективним напрямком подальших досліджень вважаємо застосування методів машинного навчання для можливості надання неігровим персонажам емоцій, можливості вступати в діалог тощо.

Список використаних джерел

1. Сеніва, К. Р. Способи використання нейронних мереж та машинного навчання в комп'ютерних іграх / К. Р. Сеніва // Вісник Хмельницького національного університету. – 2021. – № 2 (295). – С. 97-100. DOI: https://www.doi.org/10.31891/2307-5732-2021-295-2-97-100.

2. Шиманський, В. Аналіз методів машинного навчання для керування неігровими персонажами у грі виживання / В. Шиманський, В. Задерецький // XII Міжнародна науково-практична конференція «New integrations of modern education in universities», 05-08 грудня 2023 р., Амстердам, Нідерланди. – 2023. – С. 379-382. URL: https://isg-konf.com/new-integrations-of-modern-education-in-universities.

3. Almeida Rocha, D. Simulating Human Behaviour in Games using Machine Learning / D. de Almeida Rocha, J. Cesar Duarte // Proceedings of 18th Brazilian Symposium on Computer Games and Digital Entertainment (SBGames), Rio de Janeiro, Brazil. – 2019. – Pp. 514-523. DOI: https://doi.org/10.1109/SBGames.2019.00030.

4. Geisler, B. Integrated Machine Learning For Behavior Modeling in Video Games / B. Geisler // Radical Entertainment. – 2014. – Pp. 1-9.

5. Марчук, Г. Дослідження методів штучного інтелекту для створення інтелектуальних ігрових агентів / Г. Марчук, О. Коротун, В. Левківський, М. Українець // Технічні науки та технології. – 2024. – № 3 (37). – С. 122-131. DOI: https://doi.org/10.25140/2411-5363-2024-3(37)-122-131.

6. Nasteski, V. An overview of the supervised machine learning methods / V. Nasteski // HORIZONS.B. – 2017. – Vol. 4. – Pp. 51-62. DOI: https://doi.org/10.20544/HORIZONS.B.04.1.17.P05.

7. Plaat, A. Learning to Play / A. Plaat // Springer, Cham. – 2020. – P. 330. DOI: https://doi.org/10.1007/978-3-030-59238-7.

8. Wiederhold, G. Arthur Samuel: Pioneer in Machine Learning / G. Wiederhold, J. McCarthy // IBM Journal of Research and Development. – 1992. – Vol. 36, № 3. – Pp. 329-331. DOI: https://doi.org/10.1147/rd.363.0329.

9. Yang, L. On hyperparameter optimization of machine learning algorithms: Theory and practice / L. Yang, A. Shami // Neurocomputing. – 2020. – Vol. 415. – Pp. 295-316. DOI: https://doi.org/10.1016/j.neucom.2020.07.061

10. Burnes, A. Introducing NVIDIA ACE For Games - Spark Life Into Virtual Characters With Generative AI [Electronic resource] / A. Burnes // NVIDIA. – 2023. – Access mode: https://www.nvidia.com/en-us/geforce/news/nvidia-ace-for-games-generative-ai-npcs.

11. Almón-Manzano, L. Reinforcement Learning in Agents' Training: Unity ML-Agents / L. Almón-Manzano, R. Pastor-Vargas, J.M.C. Troncoso // Bio-inspired Systems and Applications: from Robotics to Ambient Intelligence. IWINAC 2022. Lecture Notes in Computer Science, Springer, Cham. – 2022. – Vol. 13259. – Pp. 391-400. DOI: https://doi.org/10.1007/978-3-031-06527-9_39.

References

1. Seniva, K.R. (2021). Sposoby vykorystannia neironnykh merezh ta mashynnoho navchannia v kompiuternykh ihrakh [Ways to use Neural Networks and Machine Learning in Computer Games]. *Visnyk Khmelnytskoho natsionalnoho universytetu – Herald of Khmelnytskyi national university, 2(295),* 97-100. https://www.doi.org/10.31891/2307-5732-2021-295-2-97-100.

2. Shimanskyi, V., Zaderetskyi, V. (2023). Analiz metodiv mashynnoho navchannia dlia keruvannia neihrovymy personazhamy u hri vyzhyvannia [Analysis of machine learning methods for controlling non-playable characters in a survival game]. *XII Mizhnarodna naukovo-praktychna konferentsiia «New integrations of modern education in universities», Amsterdam, Niderlandy – Proceedings of the XII International Scientific and Practical Conference. Amsterdam, Netherlands.* (pp. 379-382). https://isg-konf.com/new-integrations-of-modern-education-in-universities.

3. Almeida Rocha, D., Cesar Duarte, J. (2019). Simulating Human Behaviour in Games using Machine Learning. *Proceedings of 18th Brazilian Symposium on Computer Games and Digital Entertainment (SBGames), Rio de Janeiro, Brazil,* 514–523. DOI: https://doi.org/10.1109/SBGames.2019.00030.

4. Geisler, B. (2014). Integrated Machine Learning for Behavior Modeling in Video Games. *Radical Entertainment,* 1-9.

5. Marchuk, G., Korotun, O., Levkivskyi, V., Ukrainets, M. (2024). Doslidzhennia metodiv shtuchnoho intelektu dlia stvorennia intelektualnykh ihrovykh ahentiv [Research of artificial intelligence methods for creating intelligent game agents]. *Tekhnichni nauky ta tekhnolohii - Technical Sciences and Technologies,* 3 (37), 122–131. https://doi.org/10.25140/2411-5363-2024-3(37)-122-131.

6. Nasteski, V. (2017). An overview of the supervised machine learning methods. *HORIZONS.B,* 4, 51-62. DOI: https://doi.org/10.20544/HORIZONS.B.04.1.17.P05.

7. Plaat, A. (2020). Learning to Play. *Springer, Cham,* 330. DOI: https://doi.org/10.1007/978-3-030-59238-7.

8. Wiederhold, G., McCarthy, J. (1992). Arthur Samuel: Pioneer in Machine Learning. *IBM Journal of Research and Development,* 36(3), 329–331. DOI: https://doi.org/10.1147/rd.363.0329.

9. Yang, L., Shami, A. (2020). On hyperparameter optimization of machine learning algorithms: Theory and practice. *Neurocomputing,* 415, 295–316. DOI: https://doi.org/10.1016/j.neucom.2020.07.061

10. Burnes, A. (2023). Introducing NVIDIA ACE For Games - Spark Life Into Virtual Characters With Generative AI. *NVIDIA.* https://www.nvidia.com/en-us/geforce/news/nvidia-ace-for-games-generative-ai-npcs.

11. Almón-Manzano, L., Pastor-Vargas, R., Troncoso, J.M.C. (2022). Deep Reinforcement Learning in Agents' Training: Unity ML-Agents. Bio-inspired Systems and Applications: from Robotics to Ambient Intelligence. IWINAC 2022. Lecture Notes in Computer Science. *Springer, Cham,* 13259, 391–400. DOI: https://doi.org/10.1007/978-3-031-06527-9_39

Отримано 04.12.2024

TECHNICAL SCIENCES AND TECHNOLOGIES

UDC 004.89

Vitalii Levkivskyi[1], Galyna Marchuk[2], Oleksandr Moskalyk[3]

[1]PhD in Software Engineering, Associate Professor of the Department of Computer Sciences
Zhytomyr Polytechnic State University (Zhytomyr, Ukraine)
E-mail: levkivskyy@ztu.edu.ua. **ORCID:** https://orcid.org/0000-0002-1643-0895. **ResearcherID:** GYU-9377-2022

[2]Senior Lecturer of the Department of Computer Sciences
Zhytomyr Polytechnic State University (Zhytomyr, Ukraine)
E-mail: pzs_mgv@ztu.edu.ua. **ORCID:** https://orcid.org/0000-0003-2954-1057. **ResearcherID:** AAD-7514-2022

[3]Master's student of the Department of Computer Sciences
Zhytomyr Polytechnic State University (Zhytomyr, Ukraine)
E-mail: alexundercover191@gmail.com. **ORCID:** https://orcid.org/0009-0005-6555-1545

GAME ARTIFICIAL INTELLIGENCE MODEL FOR NON-PLAYABLE CHARACTERS

Artificial intelligence in video games plays a crucial role in shaping gameplay, and its creation has always been a complex and challenging task. Developing effective game artificial intelligence requires understanding and knowledge of many aspects, including algorithms, rules, and strategies that a created artificial intelligence will follow. Applying machine learning methods to create an artificial intelligence model opens up many opportunities to accelerate and optimize game development processes and improve the behavior of non-playable characters, and consequently, the overall player experience. One type of machine learning is reinforcement learning, which involves receiving rewards for correct behavior or penalties for mistakes. In this process, a non-playable character gains experience interacting with the external environment.

The aim of the work is to explore the possibilities of using the reinforcement learning method to develop a game artificial intelligence model for a non-playable character capable of self-learning and pursuing other game characters. Training non-playable characters is a key task for creating exciting and realistic game environments. The primary goal of NPC training was to ensure the ability to effectively pursue a moving target. The experimental results confirmed that the developed model successfully learns and makes optimal decisions in a game environment. The study proved the effectiveness of using machine learning methods in creating game artificial intelligence. In particular, reinforcement learning opens up new horizons for developing їхadaptive and unpredictable scenarios in games. Promising areas for further research include integrating machine learning methods to provide non-playable characters with additional capabilities that will contribute to a deeper player immersion in the gameplay.

Keywords: *Artificial Intelligence; Game; Non-Playable Character; Machine Learning; Model; Reinforcement Learning; NPC.*
Figures: 9. References: 11.

Левківський В. Л., Марчук Г. В., Москалик О. С. Модель ігрового штучного інтелекту для некерованого гравцем персонажу. *Технічні науки та технології.* 2024. № 4(38). С. 163-171.