Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Інформаційних радіотехнологій та технічного захисту інформації
(повна назва)

Кафедра Медіаінженерії та інформаційних радіоелектронних систем
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)
(позначення документа)

Удосконалення та моделювання ігрових технологій засобами ігрового рушія Unreal Engine 4
(тема)

Виконав:
студент 2 курсу, групи *МІм-21-1*
Ілля ОРЛИК
(прізвище, ініціали)

Спеціальність
172 Телекомунікації та радіотехніка
(код і повна назва спеціальності)
Тип програми *освітньо-професійна*
(освітньо-професійна або освітньо-наукова)
Освітня програма *Медіаінженерія*
(повна назва освітньої програми)

Керівник *Костянтин КОЛІСНИК*
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ Володимир КАРТАШОВ
(підпис) (ім'я, прізвище)

2022 р.

Харківський національний університет радіоелектроніки

Факультет Інформаційних радіотехнологій та технічного захисту інформації
Кафедра Медіаінженерії та інформаційних радіоелектронних систем
Рівень вищої освіти перший (бакалаврський)
Спеціальність 172 Телекомунікації та радіотехніка
(код і повна назва)
Тип програми освітньо-професійна
(освітньо-професійна або освітньо-наукова)
Освітня програма "Медіаінженерія"

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«____»_____ 20 ___ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

Студентові *Орлику Іллі Вікторовичу*
(прізвище, ім'я, по батькові)

1. Тема роботи *Удосконалення та моделювання ігрових технологій засобами ігрового рушія Unreal Engine 4*
затверджена наказом по університету від «*24*» *10* *2022 р.* № *1384 Ст*
2. Термін подання студентом роботи *08.12.2022 р.*
3. Вихідні дані до проекту (роботи)
1. Створення механік гри
2. Створення зображень з використанням 3D графіки
3. Розробка дизайну та структури взаємодій ігрового рівня
4. Розробка гри в ігровому рушії Unreal Engine 4
5. Експорт гри в вигляді виконавчого файлу
4. Перелік питань, що потрібно опрацювати в роботі
ВСТУП
1. Аналіз існуючих методів та засобів розробки комп'ютерних ігор
2. Аналіз сучасних ігрових продуктів
3. Аналіз та розробка алгоритмів гри
4 проєктування та розробка гри Container 2048 Yard
ВИСНОВКИ
ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ
ДОДАТКИ
5. Перелік графічного матеріалу із зазначенням обов'язкових креслеників, схем, плакатів, комп'ютерних ілюстрацій:
1. Постановка задачі; 2. Актуальність роботи; 3. Розробка гри; 4. Програми для розробки комп'ютерних ігор; 5. Комп'ютерна графіка; 6. Програми для створення 3D графіки; 7. Аналіз сучасних ігрових продуктів; 8. Аналіз та розробка алгоритмів гри 2048; 10. Проєктування та розробка гри Container 2048 Yard; 11. Висновки.

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Аналіз існуючих методів та засобів розробки комп'ютерних ігор	24.10.22–28.11.22	
2	Аналіз сучасних ігрових продуктів	28.10.22–29.11.22	
3	Аналіз та розробка алгоритмів гри	15.11.22–30.11.22	
4	Проєктування та розробка гри Container 2048 Yard	30.11.22–07.12.22	
5	Графічна частина проекту	07.12.22–08.12.22	
6	Перевірка керівником проекту	07.12.22–08.12.22	
7	Перевірка на академічний плагіат	08.12.22–09.12.22	
8	Перевірка зав. кафедрою, рецензування	09.12.22–10.12.22	

Дата видачі завдання 24.10.2022 р.

Студент _____ Ілля ОРЛИК
(підпис) (ім'я, прізвище)

Керівник роботи _____ доц. Костянтин КОЛІСНИК
(підпис) (посада, ім'я, прізвище)

РЕФЕРАТ

Пояснювальна записка кваліфікаційної роботи має: 70 с., 48 рис., 1 табл., 33 джерела.

3D, ОБ'ЄКТ, РОЗРОБКА, КОМП’ЮТЕРНА ГРА, ІГРОВИЙ РУШІЙ, ПАКУВАННЯ, ІГРОВА ІНДУСТРІЯ, UNREAL ENGINE, АЛГОРИТМ

Об'єкт дослідження – створення та удосконалення комп’ютерних ігор.

Предмет дослідження – методи та засоби створення тривимірної гри на ігровому рушії Unreal Engine.

Мета кваліфікаційної роботи – розробка та удосконалення тривимірної гри на існуючому ігровому рушії Unreal Engine, створення її графічного наповнення.

Методи дослідження – теоретичний аналіз, числові розрахунки, математичне моделювання, статистична обробка даних.

У роботі було досліджено різні типи комп’ютерної графіки, проаналізовано вибір програм для створення графіки, а також ігрових рушіїв для розробки гри. Розглянуто процес створення та удосконалення гри на основі ігрового рушія Unreal Engine 4. Комп’ютерно гру було експортовано для платформи Windows (64-bit) у вигляді готового виконавчого файлу з розширенням “.exe”.

В результаті роботи створено тривимірну комп’ютерну гру на рушії Unreal Engine 4, що відповідає ігровим стандартам сучасної ігрової індустрії.

ABSTRACT

The explanatory note of the qualification work has: 70 pages, 48 figures, 1 table, 37 sources.

3D, OBJECT, DEVELOPMENT, COMPUTER GAME, GAME ENGINE, PACKAGING, GAME INDUSTRY, UNREAL ENGINE, ALGORITHM

The object of research is the creation and improvement of computer games.

The subject of research is the development of a three-dimensional game using the Unreal Engine game engine.

The purpose of the work is to create graphic content, develop and improve a three-dimensional game on the existing game engine.

Research methods – theoretical analysis, numerical calculations, mathematical modeling, statistical data processing.

The work explored different types of computer graphics, analyzed the choice of programs for creating graphics, as well as game engines for game development. The process of creating and improving the game based on the Unreal Engine 4 game engine was considered. The computer game was exported for the Windows platform (64-bit) in the form of a ready-made executable file with the extension '.exe'.

As a result of the work, a three-dimensional computer game was created using the Unreal Engine 4 engine, which meets the game standards of the modern game industry.

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ І СКОРОЧЕНЬ

ПЗ – програмне забезпечення;

3D – тривимірний опис об’єкта;

GPU – graphics processing unit – графічний процесор;

2D – двовимірний опис об’єкта;

WWW або WEB – World Wide Web – всесвітня мережа;

UE – Unreal Engine

ПК – персональний комп’ютер;

PS – Play Station;

iOS – iPhone operating system;

ЗМІСТ

ВСТУП

Комп'ютерні ігри - це складні програми, в яких елементи керування, що виконуються гравцем, призводять до змін у додатку, зображенні та звуці, які призначені для навчання або розваги людини. Вони призначені для різних вікових категорій, тому що кожна гра має своє призначення, тему і стиль.

З'явившись в 1940-х і 1960-х роках, комп'ютерні ігри не відразу стали популярними. Вони стали частиною людської культури тільки в 1970 році, коли ігрові консолі та домашні комп'ютери стали доступні широкому загалу.

Заняття комп’ютерними іграми має позитивний вплив на психічні процеси суб’єкта. Користь від комп’ютерної гри полягає у розвитку уваги, моторики, швидкості реакції, сприйняття, розвитку мислення, уяви, креативності.

Аналіз літературних джерел показав, що комп’ютерні ігри сприяють набуттю спеціальних знань та розвивають когнітивні навички, що входять у загальну структуру діяльності. Ігровий досвід розширює і поліпшує низку умінь і навичок, важливих для навчання, для професійної діяльності та соціальної взаємодії, тому задача створення нових і удосконалення уже існуючих ігор є актуальною.

Також остається актуальним те, що ігри можуть бути корисними інструментами навчання і соціалізації, допомагати дорослим і дітям у соціальному розвитку, збагачувати репертуар поведінки, розвивати соціальні навички, ситуаційну та рольову компетентність.

Основною метою дипломної роботи є розробка алгоритмів та вдосконалення механік й графічного навовненя гри за допомогою ігрового рушія Unreal Engine 4.

Для досягнення цієї цілі були поставлені такі задачі:

- проведення аналізу предметної області;
- обгрунтування вибору необхідного програмного забезпечення;
- розробка алгоритму та механік гри;

– створення 3D моделей для наповнення гри;
– розробка дизайну та структури взаємодій ігрового рівня;
– розробка гри в ігровому рушії Unreal Engine 4;
– експорт гри в вигляді виконавчого файлу.

1 АНАЛІЗ ІСНУЮЧИХ МЕТОДІВ ТА ЗАСОБІВ РОЗРОБКИ КОМП'ЮТЕРНИХ ІГОР

Розробка гри — комплексне завдання, яке торкається відразу кількох складних дисциплін. Зокрема, для створення гри необхідно мати ідею майбутньої гри, вибрати ігровий движок або написати його самостійно, а також необхідно підготувати алгоритм гри для розуміння, що необхідно реалізувати в грі. І тільки після підготовки можна приступати до написання механік гри та додавання до неї візуальної складової.

У цьому розділі будуть розглянуті теоретичні аспекти, необхідні для розуміння підходу до розробки комп'ютерних ігор.

1.1 Аналіз та обґрунтування вибору ігрового рушія

Термін «ігровий рушій» означає сукупність програмних засобів, які розробники використовують створення ПЗ [1].

У другій половині 1990-х і в першій половині 2000-х багато студій використовували самописні рушії, оскільки відомі технології – наприклад, idTech або Unreal Engine – були недоступні для невеликих команд через високу вартість ліцензування [2].

Найчастіше 3D-рушії або системи рендерингу в ігрових рушіях побудовані на графічному API, такому як Direct3D або OpenGL, що забезпечує програмну абстракцію GPU або відеокарти [1]. Низькорівневі бібліотеки, наприклад, DirectX, SDL і OpenAL, також використовуються в іграх, оскільки забезпечують апаратно-незалежний доступ до іншого апаратного забезпечення комп'ютера, такого як пристрої введення (миша, клавіатура та джойстик), мережні та звукові карти. До появи апаратно-прискорюваної 3D-графіки використовувалися програмні візуалізатори [3]. Програмний рендеринг все ще використовується в деяких інструментах моделювання для рендерингу зображень, для яких візуальна достовірність

важливіша за продуктивність (кількість кадрів на секунду) або коли апаратне забезпечення комп’ютера не задовольняє вимогам, наприклад, не підтримує шейдери.

1.1.1 Аналіз ігрового рушія «GameMaker Studio 2»

GameMaker Studio 2 — це зручний інструмент для розробників-початківців [4]. Усередині нього є власний магазин готових спрайтів, звукових доріжок, графічних карт, анімаційних об’єктів та ін.

Спочатку GameMaker Studio 2 було розроблено лише для 2D-ігор. Хоча 3D-графіка підтримується, але якість буде не найкраща [5].

На GameMaker Studio 2 можна моделювати ігри для різних популярних платформ, наприклад, для Windows, Linux, MacOS, Android, iOS, ігрових приставок і навіть просто для web.

Розробник з мінімальними навичками програмування зможе розібратися в цьому двигуні. Якщо є гарна ідея гри та мінімальні навички розробки ігор, то GameMaker Studio 2 модна використовувати для створення повноцінної 2D-гри, не написавши жодного рядка коду та жодного скрипту.

Як і будь-який інший інструмент для програмування чого-небудь, GameMaker Studio 2 має свої переваги і недоліки [6].

Серед переваг можна виділити:

1. Наявність графічного інтерфейсу, що дозволяє створювати ігри, не вдаючись до написання скриптів.
2. Кросплатформенність, яка відкриває можливість створювати ігри для різних операційних систем в одному інструменті.
3. Невисока ціна та безкоштовний тестовий період, під час якого є можливість спробувати цей інструмент та вирішити, чи варто за нього платити. Хоч і коштує він відносно недорого — 99 $ за довічну ліцензію.

4. Власна мова програмування, яка має багато спільного з іншими популярними мовами, що полегшує її вивчення.
5. Є можливість настроїти інтегрування із сервісом Steam.
6. Є можливість налаштувати інтегрування з багатьма інтернет-майданчиками для розповсюдження ігор.
7. Власний магазин готових рішень для створення ігор: спрайти, анімації, елементи та ін.
8. Детальна офіційна документація.
9. Велика спільнота, де завжди можна знайти відповідь на проблему зі створенням гри.

Серед недоліків можна виділити:

1. Погана якість 3D-ігор, тому для таких ігор цей двигун не підходить.
2. Ігри, зроблені на GameMaker Studio 2, неможливо перенести на інші двигуни.
3. Безкоштовна версія (Free) дозволяє створювати ігри лише під ігрову платформу Opera GX.games.

Проаналізувавши GameMaker Studio 2, можна зробити висновок, що це інструмент, який можна досить швидко освоїти і на якому є можливість почати створювати нескладні 2D-ігри. Але створити гарний 3D-проект на можливо.

1.1.2 Аналіз ігрового рушія «Unity»

Unity – це ігровий рушій для створення мобільних та комп’ютерних відеоігор [7]. Unity дозволяє адаптувати код під виконання на 28 платформах. Завдяки візуальному редактору дозволяє реалізувати творчі здібності дизайнерів, художників, програмістів, геймдевелоперів – початківців.

Окрім ігрової сфери застосовується в автомобільній, машинобудівній, авіакосмічній індустрії, при виробництві мультфільмів, кіно, будівельній галузі, промисловості, архітектурі [8].

Unity складається із візуального редактора, редактора коду, інструменту для написання скриптів – логіки поведінки об’єктів. Вона дозволяє вносити зміни, не залишаючи сцени, і в реальному часі оцінювати їх результат.

Містить інструменти для створення точних тривимірних копій реальних об’єктів та просторів (кімнат, будівель, відкритої місцевості).

Платформа надає готову фізичну модель взаємодії між об’єктами віртуальної сцени, чим позбавляє розробника від опрацювання поведінки кожного з елементів [9].

У Unity вбудований фізичний двигун. Він включає закони, правила взаємодії елементів сцени між собою, із навколишнім середовищем.

У бібліотеці багато пресетів із різними налаштуваннями фізики світу, які можна завантажувати у проект та змінювати.

Імітація фізичних явищ та об’єктів на основі частинок (атмосферні опади, вогонь, відображення) проводиться із задіянням розробки від Nvidia PhysX.

До складу фізичної сторони двигуна входить фізика твердих і м’яких тіл – тканин, волосся, диму та рідин. Система успадкування властивостей змусить дочірні предмети копіювати властивості та поведінку батьківських. Скрипти до кожного їх прикріплюються окремо [7].

Завдяки використанню C# програміст – початківець зможе писати скрипти – крихітні програми, шматки коду, що задають поведінкові реакції об’єктів на події, явища – створювати квести з розгалуженою структурою.

Інтерфейс з підтримкою Drag&Drop, що настроюється під розробника, спрощує роботу над сценами, їх налагодження і редагування без перемикання між вікнами.

Візуальний редактор працює з шейдерами, картами відбиття, проте останні не застосовуються безпосередньо до моделі, а прикріплюються до неї лише після призначення шейдера.

Останні можна як створювати з нуля, так і редагувати існуючі. Засіб для створення тривимірної анімації підтримує її імпорт із Blender, 3DsMax та інших 3D-редакторів.

Для оптимізації навантаження обладнання розробник передбачив опцію Level Of Detail. Вона знижує деталізацію та якість промальовування віддалених об’єктів, а при наближенні – промальовує їх повністю. Алгоритми Occlusion Culling не візуалізують об’єкти, розташовані поза полем зору камери, видаляють з пам’яті гравця, що залишається за спиною. Об’єкти за спиною видаляються з пам’яті [10].

Скомпільована гра для Windows запускається через виконавчий файл, всі ресурси (рівні, мультимедіа) зберігаються в окремих файлах, бібліотеки, що динамічно підключаються.

У Unity Asset Server є інструменти реалізації мультиплеєра. Система контролю версій із графічним поданням дозволить відкотити стан проекту, відстежувати зміни між версіями. Unity працює з освітленням: підтримує трасування променів – створює реалістичну картинку шляхом відстеження взаємодії світла з поверхнями. Працює з сучасними графічними технологіями: DirectX 12, VRWorks, Vulkan, LiquidVR [12].

Рушій має ряд позитивних та негативних сторін.

Переваги платформи:

1. Функціональний графічний редактор – передбачає створення локацій, моделей, розміщення об’єктів сцени з функцією тестування результатів у реальному часі.
2. Кросплатформове середовище розробки дозволяє адаптувати проект для будь-якої платформи шляхом внесення мінімальних змін. Мінімізує вартість проекту.
3. Інтегроване середовище розробки – програмний комплекс для роботи з рушієм, маючи базові знання програмування.

4. Підтримка плагінів – можливість встановлення компонентів, що розширюють можливості, що підвищують зручність, швидкість роботи із середовищем.
5. Модульність дозволяє конструювати пакети компонентів в одній ігровій сцені.
6. Просунута методика створення об’єктів: замість успадкування шляхом розміщення у дереві вони об’єднуються у функціональні блоки, що спрощує етап прототипування.
7. Використання високорівневої мови з рядом готових рішень спрощує підключення програмістів до розробки.
8. Якість підтримки – Unity сайт містить опис всіх функцій движка, приклади їх застосування з поясненнями. Служба підтримки оперативно відповідає на запитання.
9. Безкоштовна експлуатація для застосування в особистих та освітніх цілях.
10. Спільнота – просунуті користувачі завжди підкажуть, допоможуть порадою. На тематичних форумах та блогах повно інформації щодо Unity.

Недоліки платформи:

1. Проблеми розробки багатокомпонентних проектів через обмеження візуального редактора.
2. Складнощі в підключенні зовнішніх плагінів – через відсутність посилань на них у програмі модулі доведеться шукати, встановлювати та налаштовувати самому.
3. Проблематичне внесення змін до шаблонів екземплярів.
4. WebGL - редакція відрізняється низкою проблем зі швидкодією, працездатністю та стабільністю.
5. Швидкодія, в порівнянні з іншими двигунами, при розростанні проекту.

6. Великий розмір компілюваного файлу особливо для мобільних платформ.

Платформа розрахована на геймдевелоперів - початківців і невеликі компанії, яких функціональність простих продуктів на кшталт RPG Maker, Kodu Game Lab не влаштовує, а ресурсів придбати просунутий двигун немає. Unity застосовується для створення відеороликів, реалізації кросплатформових 2D-проєктів (підтримка 2D з'явилася нещодавно) та масштабних проектів на зразок The Elder Scrolls: Legends, Pillars Of Eternity. Також використовується для написання симуляцій.

1.1.3 Аналіз ігрового рушія «Unreal Engine»

Unreal Engine (UE) – це ігровий рушій, розроблений фахівцями Epic Games [13].

Рушій Unreal Engine активно застосовується для розробки простих казуальних ігор для смартфонів та планшетів, а також для створення повноцінних високобюджетних ігор, розрахованих на масову аудиторію. При цьому не потрібно самостійно писати код – система візуального створення скриптів Blueprints Visual Scripting значно спрощує завдання. Якщо розробник бажає прописати ігрову логіку вручну, він може використовувати мову програмування C++ [13].

5 квітня 2022 року Epic Games представила оновлений рушій Unreal Engine 5. Серед головних переваг якого – максимальний фотореалізм, збільшена продуктивність та новий інтерфейс.

«Unreal Engine» залишається популярним понад 20 років із кількох причин.

1. Широкий функціонал. У цьому рушії можна створити практично будь-яку гру від шутера до аркади з різними типами логіки.
2. Візуальне програмування. Працювати з UE зможуть навіть новачки: вбудована система візуального скриптингу позбавить необхідності

використання складних мов програмування. Але за умовчанням це підтримує і стандартний C++, звичний для розробників з великим досвідом.

3. Безкоштовна ліцензія. Особливо приємний бонус для багатьох розробників. У ліцензійній угоді ПЗ зазначено, що доки гра не окупить себе і не принесе $ 1 млн, двигуном можна користуватися безкоштовно. Далі доведеться сплачувати 5% від суми доходу.
4. Можливість створити крос-платформер. Багато користувачів володіють відразу кількома ігровими пристроями (ПК, консолями, смартфонами і т.д.), і рушій Unreal Engine дозволяє зробити додаток, який функціонуватиме на різних платформах (Android, Xbox, PS, Switch, iOS, Windows);
5. Велика база користувачів. За роки у цієї програми з’явилося багато користувачів, які готові поділитися своїм досвідом та напрацюваннями у тематичних спільнотах.

1.2 Огляд методів алгоритмізації

Основи алгоритмізації та програмування є фундаментальними основами теоретичної інформатики. Алгоритм – описана деякою мовою точна кінцева система правил, що визначає зміст і порядок дій над деякими об’єктами, суворе виконання яких дає рішення, поставленого завдання. Поняття алгоритму, яке є фундаментальним у математиці та інформатиці, з’явилось задовго до появи засобів обчислювальної техніки. Слово «алгоритм» з’явилося в середні віки, коли європейці познайомилися зі способами виконання арифметичних дій у десятковій системі числення [14].

Будь-який алгоритм існує не сам собою, а призначений для певного виконавця (людини, робота, комп’ютера, мови програмування тощо). Властивістю, що характеризує будь-якого виконавця, є те, що він вміє виконувати деякі команди [15]. Сукупність команд, які цей виконавець вміє

виконувати, називається системою команд виконавця. Алгоритм описується у командах виконавця, який його реалізовуватиме. Об’єкти, з яких виконавець може вчиняти дії, утворюють так зване середовище виконавця. Вихідні дані та результати будь-якого алгоритму завжди належать серед того виконавця, для якого призначений алгоритм.

Алгоритмом називається суворо певна послідовність дій, що визначають процес переходу від вихідних даних до результату.

Порядок дій вважається алгоритмом у тому випадку, якщо він має певні властивості [16]:

- Дискретність. Алгоритм повинен представляти процес розв’язання задачі як послідовність виконання простих дій (кроків, етапів). При цьому для виконання кожної дії алгоритму потрібен час.
- Детермінованість (однозначність). Кожна дія (крок, етап) має бути чіткою, однозначною, що виключає довільне тлумачення і не залишає місця для двозначності. Виконання алгоритму має, по суті, механічний характер і не вимагає додаткових вказівок.
- Зрозумілість. Алгоритм повинен включати лише команди, доступні виконавцю і входять у його систему команд.
- Результативність. Алгоритм повинен призводити до розв’язання задачі чи повідомлення, що завдання рішень немає за кінцеве число кроків.
- Кінцівка. Кожна окрема дія, як і весь алгоритм, повинні мати можливість реального виконання. Тому алгоритм має межу, тобто кінцевий.
- Масовість. Алгоритм розробляється у вигляді так, щоб його можна було застосовувати для класу завдань, що відрізняються тільки вихідними даними. При цьому вихідні дані вибираються з деякої області, яка називається областю застосування алгоритму.

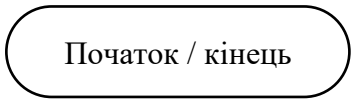
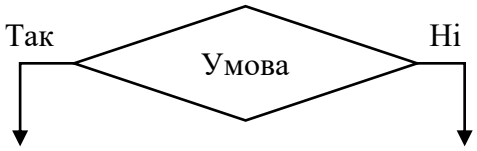
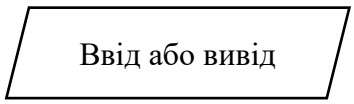
1.2.1 Огляд методів запису алгоритмів

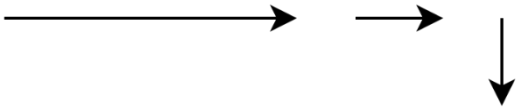
Існують різні способи запису алгоритмів – словесно-формульний, графічний, операторний (програма алгоритмічною мовою) [17]. Розглянемо їх:

а) словесно-формульний метод. Наприклад, потрібно вирішити квадратне рівняння $ax^2+bx+c=0$ у ділянці дійсних чисел. Математичною моделлю цього завдання є відома формула коренів квадратного рівняння: $y_{1,2}=-b^2-4*ac$;

б) графічний спосіб опису алгоритму інакше називають блок – схемою, у таблиці 1 наведені найбільш часто вживані блоки. У блок-схемах використовуються геометричні фігури, кожна з яких зображує якусь операцію чи дію, і навіть етап процесу розв’язання завдання. Кожна фігура називається блоком. Порядок виконання етапів вказується стрілками, що з’єднують блоки. Блоки необхідно розміщувати зверху донизу або зліва направо у порядку їх виконання [18].

Таблиця 1 - Найбільш часто вживані блоки

Позначення блоку	Виконувана функція
	Початок або Кінець алгоритму
	Обчислювані дії
	Перевірка умови: вибір одного із двох напрямків
	Введення або Виведення даних
	Організація циклічних процесів

	Напрямок ліній потоку – стрілки

в) Операторний метод (алгоритмічна мова). Алгоритм – це завдання виконавця. Виконавець виконує алгоритм, тобто робить те, що написано в алгоритмі. Якщо виконавець точно виконає те, що написано алгоритмі, він отримає результат.

Людина, автоматичний пристрій, комп'ютер – це різні виконавці алгоритмів. Для того щоб комп'ютер міг виконати алгоритм, його треба написати зрозумілою комп'ютеру мовою. Комп'ютер розуміє машинну мову. Наприклад, рівність x = y машинною мовою має вигляд: 111101110011110111110101 [19].

Людині важко писати і читати алгоритми машинною мовою, але людина легко може писати і читати звичайною мовою. Але не можна навчити комп'ютер розуміти звичайну мову.

Щоб людина і комп'ютер розуміли одне одного, розроблені спеціальні мови для записів алгоритмів – алгоритмічні мови. Найвідоміші алгоритмічні мови – Бейсік (Basic), Паскаль (Pascal), Фортран (Fortran), C, C#, C ++, Python.

Алгоритмічна мова відрізняється від машинної мови тим, що складається зі слів та символів, як природна мова. Алгоритмічна мова відрізняється від природної мови тим, що в ній мало основних слів (зазвичай 30-40) та дуже суворі правила складання речень. Основні слова алгоритмічної мови називають службовими словами. В алгоритмічних мовах використовують слова англійської абетки. Алгоритмічний мову легко розуміє і людина, і комп'ютер [20].

Алгоритм, записаний алгоритмічною мовою, – це програма для комп'ютера. Кожна пропозиція у програмі – це оператор.

Наприклад, можна написати програму розв'язання квадратного рівняння $ax^2+bx+c=0$ на комп'ютері. Алгоритмічною мовою C++ ця програма буде виглядати наступним чином:

```
#include <iostream>
#include <cmath>
using namespace std;
int main()
{
   double a;
   double b;
   double c;
   double x;
   cout << «Ведіть значення a: «;
   cin >> a;
   cout << « Ведіть значення b: «;
   cin >> b;
   cout << « Ведіть значення c: «;
   cin >> c;
   if((b*b – 4*a*c) >= 0) // Якщо дискримінант більший або дорівнює 0
   {
      x = ( -1*b + sqrt(b*b – 4*a*c) ) / (2 * a);
      cout << «Перший корінь дорівнює « << x << endl;
      x = ( -1*b – sqrt(b*b – 4*a*c) ) / (2 * a);
      cout << «Другий корінь дорівнює « << x << endl;
   }
   else
   {
          cout << « Дискримінант менше 0, корінні комплексні.» << endl;
   }
   return 0;
}
```

На алгоритмічній мові Python ця програма виглядатиме наступним чином:

```
import math

print(«Введіть коефіцієнти рівняння «)
print(«ax^2 + bx + c = 0:»)
a = float(input(«a = «))
b = float(input(«b = «))
c = float(input(«c = «))

discr = b ** 2 – 4 * a * c
print(«Дискримінант D = %.2f» % discr)
```

```
if discr > 0:
   x1 = (-b + math.sqrt(discr)) / (2 * a)
   x2 = (-b – math.sqrt(discr)) / (2 * a)
   print(«x1 = %.2f \nx2 = %.2f» % (x1, x2))
elif discr == 0:
   x = -b / (2 * a)
   print(«x = %.2f» % x)
else:
   print(«Корнів немає «)
```

1.2.2 Огляд типів алгоритмів

Алгоритми бувають лінійні, розгалужені та циклічні.

Лінійний алгоритм – це алгоритм, у якому дії виконуються лише один раз і суворо у тому порядку, в якому вони записані, приклад блок схеми лінійного алгоритму приведено на рисунку 1.1.

Розгалужуваний алгоритм – це алгоритм, у якому той чи інший дію виконується після аналізу умови. Процес аналізу умови та вибору однієї з гілок на блок-схемі показують за допомогою логічного блоку. Приклад структури, що розгалужується, показаний на рисунку 1.2.

Процес аналізу умови та вибору однієї з гілок на блок-схемі показують за допомогою логічного блоку. Логічний блок має один вхід і два виходи (гілка «так» і гілка «ні»).

У блок-схемах алгоритмів, що розгалужуються, завжди є логічний блок.

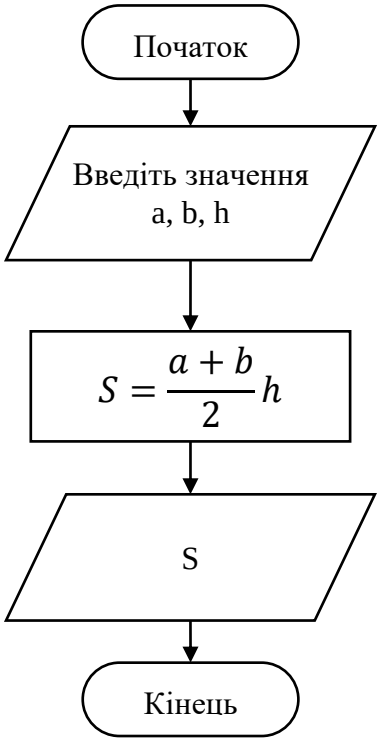


Рисунок 1.1 - Блок схема обчислення площі трапеції

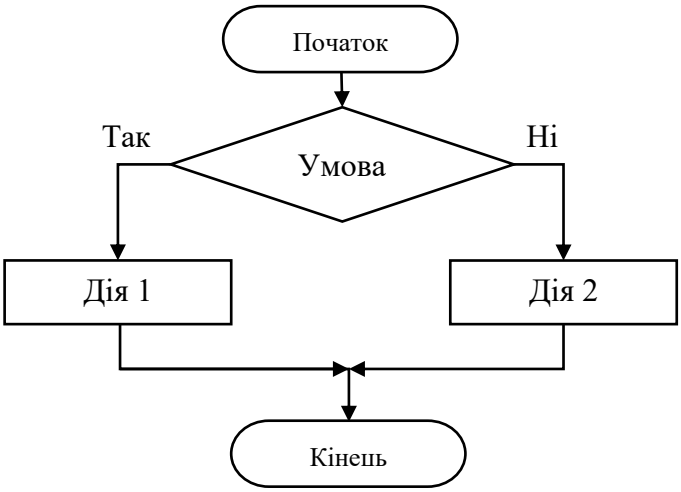


Рисунок 1.2 - Приклад структури, що розгалужується

На рисунку 1.3 наведено приклад блок-схеми розв’язання квадратного рівняння $ax^2+bx+c=0$.

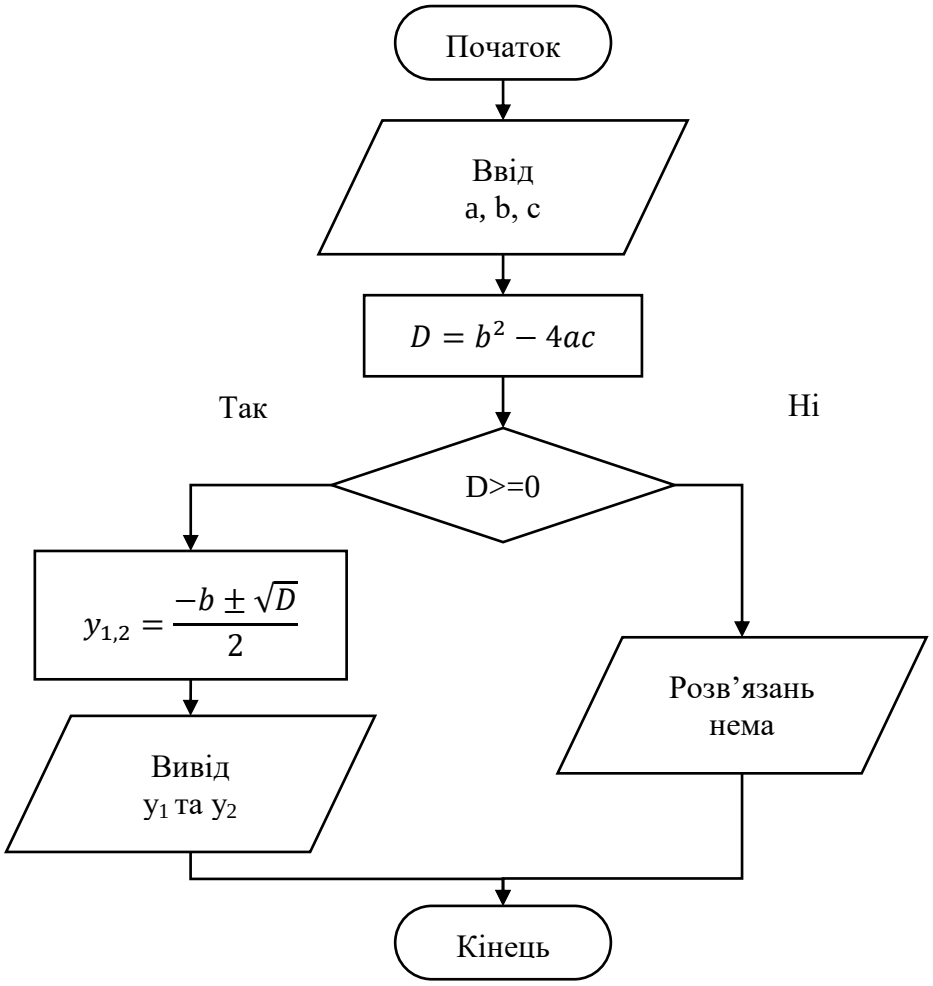


Рисунок 1.3 - Блок-схема розв’язання квадратного рівняння $ax^2+bx+c=0$.

Циклічний алгоритм. Цикл – це алгоритм, в якому група операторів виконується кілька разів поспіль. Блок-схема циклу обов’язково містить логічний блок, її структура показана на рис.

Виконується циклічний алгоритм так: спочатку перевіряється умова, якщо умова вірна (істина), то виконується тіло циклу (дії або група операторів) і далі змінюються значення параметра циклу і знову перевіряється умова і т.і. На якомусь кроці умова не виконається (брехня) і тоді відбувається вихід із циклу і продовжується виконання програми.

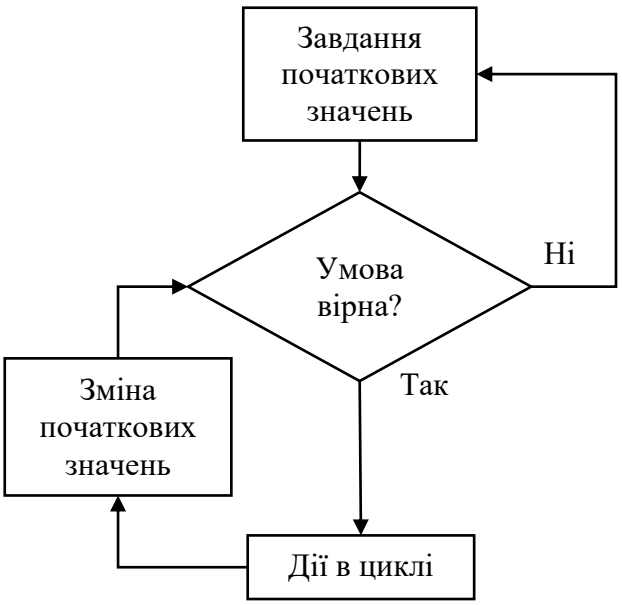


Рисунок 1.4 - Блок-схема циклу

Розрізняють три варіанти циклів:

- цикл із передумовою;
- цикл із постумовою;
- цикл із параметром.

Маючи ідею та алгоритм гри перед початком написання механік на ігровому рушії потрібно підготувати графічне наповнення з яким надалі будуть взаємодіяти користувачі.

1.3 Аналіз існуючих видів графіки у комп'ютерних іграх

Комп'ютерна графіка в сучасних іграх буває 2-х видів: 2D та 3D або двовимірна та тривимірна графіка [21]. Кожен із цих видів поділяється на власні види та типи, залежно від способу її реалізації.

1.3.1 Аналіз видів двомірної графіки та її типів

Історія графіки в іграх починається ще з 40-х років, коли були придумані перші гральні автомати, а як екран виступало полотно зі звичайних електричних лампочок. Це і графікою назвати складно, але все починалося саме так [22]. Потім йшов довгий розвиток графіки в іграх та поступове її вдосконалення. 2D-графіка в іграх вперше з'явилася вже у 80-х роках, але широкого поширення набула у другій половині 90-х.

На сьогоднішній день 2D-графіка досі застосовується у комп'ютерних іграх, проте не так активно, як 3D. Але кілька слів про двомірну графіку сказати потрібно [23].

Розрізняють такі види двовимірної графіки в іграх і не тільки:

Векторна графіка. Це набір з простих геометричних елементів, наприклад: точка, пряма, коло, прямокутник і т. д. В основному такий вид 2Д-графіки застосовується як іконки та логотипи для веб-сайтів. В іграх застосовується рідше. Відмінна риса такої графіки - це здатність до масштабування та деформації без втрати якості.

Растрова графіка. За основу такої графіки береться матриця пікселів, де кожен піксель має якесь значення чи комбінація значень кольору, світла, прозорості тощо. буд. Головний недолік такої графіки — це вища «вага», і навіть втрати якості при масштабуванні. Однак саме такий тип графіки найчастіше застосовується в іграх.

1.3.2 Аналіз видів тривимірної графіки

Тривимірна графіка – це сучасний напрямок комп'ютерної графіки. Він активно використовується починаючи від невеликих об'єктів на веб-сайтах до комп'ютерних ігор або кіно [24].

Моделювання. Це одна з найпопулярніших технологій створення 3Д-об'єктів, де кожен об'єкт описується великою кількістю вершин та гранями, що їх з'єднують. Будь-який об'єкт це геометрична фігура, що складається з

«сітки». Ця «сітка» деформується поки вона не набуде форми потрібного об'єкта [25].

Текстурування. На об'єкт потрібно накласти текстури, щоб він виглядав максимально реалістично. Сюди входить надання кольору, прозорості, матовості тощо.

Світло об'єкту. Щоб надати максимальну реалістичність об'єкту, потрібно щоб було правильно налаштовано світло на ньому. Для цього є точкові та глобальні джерела світла, які можна застосовувати залежно від ситуації.

Анімація об'єкта. Будь-якому об'єкту потрібна буде анімація. Для цього об'єкту створюють «скелет» та контролюють процес зміни його зовнішнього вигляду залежно від його пересування.

Композ - це процес при якому відбувається об'єднання кількох анімованих об'єктів в один кадр.

Симуляція частинок. Це система вільних точок у просторі, які можна використовувати на власний розсуд. З них можна зробити вогонь, воду, пісок, ефект вибухів чи чаклунства тощо.

Компанії, типу Pixar та Disney, використовують дороге ПЗ або ПЗ особистої розробки. Серед програм для 3D моделювання можна відзначити: 3DS Max, Maya, Zbrush, Mudbox, Blender та т.і. [26].

1.3.3 Аналіз програмного забезпечення для створення 3D графіки

Програма ZBrush від компанії Pixologic – це потужний професійний інструмент для створення та редагування тривимірної графіки. Програма спрямована на роботу з «цифровою глиною», з якої можна виліплювати об'єкти за допомогою різноманітних інструментів. Таке цифрове ліплення використовується для створення людей, тварин, та будь яких інших органічних об’єктів. ZBrush може використовуватися для твердотільного 3D-моделювання й має спеціальні інструменти для цього [27].

Набір спеціальних пензлів спрямований на досягнення максимальної реалістичності при створенні 3D моделей, а інструменти накладання текстур та візуалізації доповнюють функціонал програми. ZBrush практично не використовує можливості відеокарти, що суттєво впливає на швидкість роботи. Натомість основним ресурсом для програми є оперативна пам'ять комп'ютера, недолік якої може позначатися на продуктивності. ZBrush - це гнучкий пакет 3D графіки, який можна налаштувати відповідно до власних уподобань. Інтерфейс програми повністю налаштовується [28].

Blender — це безкоштовне програмне забезпечення для створення та редагування тривимірної графіки. Програма справляється з анімацією та реалістичними ландшафтами, але поступається у скульптингу персонажів.

На сьогоднішній день це повноцінний 3D-редактор, в якому користувача зустрічає інтерфейс з унікальною внутрішньою файловою системою. Оболонка програми може здатися незручною та незрозумілою, але після настроювання гарячих клавіш працювати в Blender стає просто та зручно. На офіційному сайті знаходяться у загальному доступі навчальні курси. Як мову програмування додаток використовує Python, володіючи яким є можливість створювати власні інструменти, редагувати інтерфейс й сам принцип роботи програми. Програма доступна на різних операційних системах [29].

Маya є найпоширенішою програмою для створення спецефектів, анімацій. Створює реалістичні картинки, підлаштовується під будь-якого користувача, проте дуже вимоглива до пристрою, з якого відбуватиметься робота. Маya має великий набір інструментів для анімації, текстурування та створення різноманітних спецефектів. Це редактор тривимірної графіки з реалізованим функціонало візуалізації готових моделей [30]. На офіційному сайті є ліцензія для студентів, яка видається на 3 роки безкоштовно.

3Ds Max — це програмне забезпечення для 3D-моделювання, анімації та рендерингу, створене та розроблене для ігор та візуалізації дизайну. Програмне забезпечення використовується для проєктування будівель,

інфраструктури та будівництва, а також для розробки продуктів та планування виробництва [31].

Крім того, 3Ds Max допомагає користувачам створювати масивні ігрові світи, деталізованих персонажів, налаштовувати оточення будівлі, створювати сцени, в яких багато людей, імітувати фізичні властивості рідин, таких як вода, олія та лава. Програма легка у вивченні на початкових етапах, навчльні матеріали є в загальному доступі [32]. Ліцензія для студентів видається на 3 роки.

Було проведено аналіз існуючих ігрових рушіїв в якості платформи для розробки гри для дипломного проєкту було обрано Unreal Engine. Тому що він має: зручний візуальний редактор із потужною функціональністю; націленість на масштабні проекти; завдяки візуальному програмуванню – «блюпринт» – прототипування не вимагає технічних навичок, а для розширення можливостей движка можна застосувати програмування на C++; гарна оптимізація; доступ до вихідного коду; багатий маркетплейс із безліччю ассетів під будь-які потреби; є безкоштовним та має велику кількість матеріалів, що спрощує початок роботи в ньому.

В ході роботи також було проведено аналіз існуючих видів графіки у комп'ютерних іграх який вказав на доцільність переходу до біль сучасних типів графіки в іграх, а саме тривимірної графіки.

2 АНАЛІЗ СУЧАСНИХ ІГРОВИХ ПРОДУКТІВ

Для удосконалення існуючої гри слід провести аналіз графічної частини та механик гри [33]. В результаті чого є можливість сформувати завдання на повторення механіки за допомогою візуального програмування Blueprint. Та після чого удосконалити графічне наповнення та додати нові механіки. Такий підхід наддасть можливість й в подальшому додавати нові механіки та режими гри.

2.1 Аналіз гри 2048

2048 - це захоплююча гра зі зміною плиток, в якій ми переміщуємо плитки, щоб комбінувати їх, прагнучи все більшого значення плитки.

Гра в 2048 грає на дошці 4 × 4. Кожна позиція на дошці може бути порожньою або утримувати плитку, і на кожній плитці буде номер.

Коли ми почнемо, на дошці будуть дві плитки у випадкових місцях, на кожній з яких або '2', або '4' - у кожної є незалежна 10% ймовірність того, що це '4', або інакше а одно '2'.

Ходи виконуються шляхом зсуву всіх плиток до одного краю - вгору, вниз, ліворуч або праворуч. При виконанні цього будь-які плитки з однаковим значенням, які знаходяться поруч один з одним і переміщуються разом, будуть об'єднані і в результаті отримають нову плитку, що дорівнює сумі двох попередніх. Приклад переміщення та об’єднання пліток представлено на рисунку 2.1.

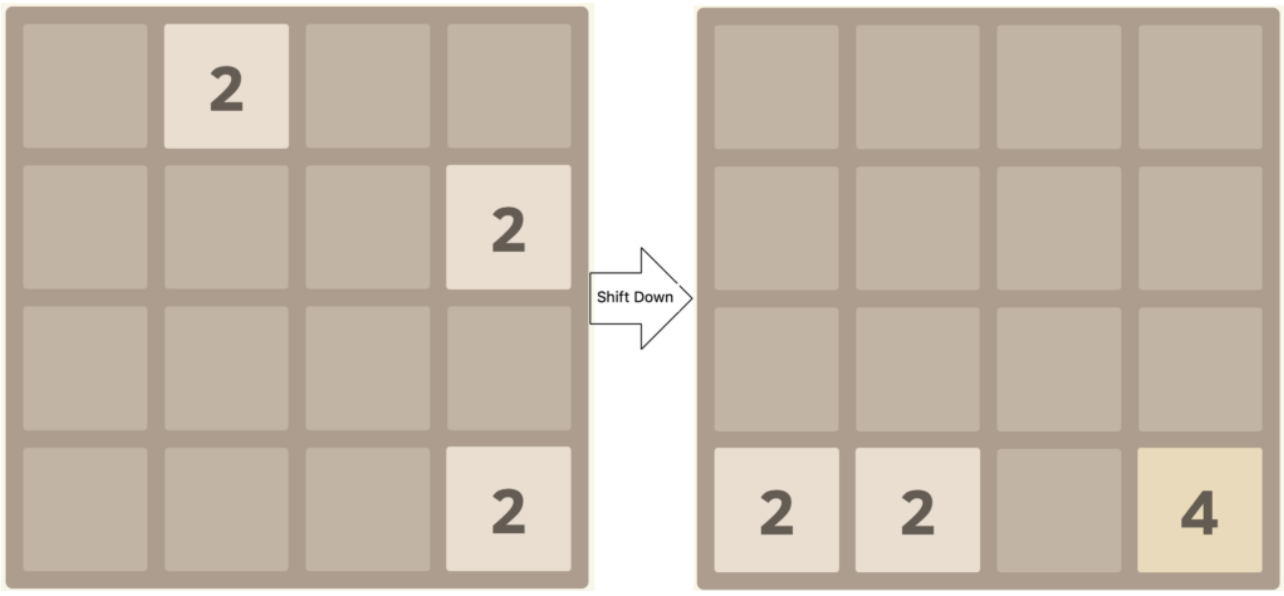


Рисунок 2.1 – Приклад переміщення пліток донизу [34]

Після того, як ми зробимо хід, на дошку буде вміщено нову плитку. Віна розміщується у випадковому місці і буде '2' або '4' так само, як і початкові плитки - '2' у 90% випадків та '4' у 10% випадків. Потім гра триває доти, доки не залишиться більше ходів.

Загалом мета гри - зібрати одну плитку зі значенням “2048”. Однак гра на цьому не закінчується, і ми можемо продовжувати грати, наскільки це можливо, прагнучи якомога більшого тайлу. Теоретично це плитка зі значенням '131072'.

2.2 Аналіз гри Age of 2048

Гра заснована на ідеї 2048 року. Age of 2048 - схожа гра з переробленим дизайном і замість числа ви граєте з елементами цивілізації. Мета гравця – побудувати місто та просунути цивілізацію вперед. Він починає з об'єднання різних блоків, щоб створити хатину племені, а потім об'єднайте їх, щоб створити плем'я, потім село, потім місто і т.і. Це гра з невеликою історією. Приклад гри Age of 2048 приведено на рисунку 2.2.

Рисунок 2.2 – Зовнішній вигляд гри Age of 2048 на мобільному пристрої [35]

Age of 2048 - схожа гра з переробленим дизайном і замість числа ви граєте з елементами цивілізації. Ваша мета – побудувати місто та просунути цивілізацію вперед. Почніть з об'єднання різних блоків, щоб створити хатину племені, а потім об'єднайте їх, щоб створити плем'я, потім село, потім місто та т.і.

2.3 Аналіз гри Kitty Cat Island

У цій грі ви граєте у класичний 2048 та допомагаєте кішкам ловити рибу. Щоразу, коли ви поєднуєте плитки під кішками, ви отримуєте рибу, яку можна з'їсти. Ваша мета - зловити якнайбільше спійманої риби. Приклад гри Kitty Cat Island приведено на рисунку 2.3.

Рисунок 2.3 – Алгоритм Kitty Cat Island на мобільному пристрої [36]

Гра має кілька додаткових функцій, які роблять її краще, ніж більшість з 2048 клонів. Наприклад, ви можете використовувати сітку для видалення непотрібних плиток, використовувати вудку, щоб позбутися плитки, що застрягла, і інструмент для перевороту, щоб поміняти місцями плитки з сусідньою плиткою. Ці інструменти допоможуть вам швидко досягти мети, якщо раніше ви мали проблеми з досягненням 2048 року.

До достоїнств гри 2048 можна віднести:

- унікальна реалізація ідеї розробників;
- найпростіше керування за допомогою свайпів;
- захоплюючий геймплей;
- геймплей є майже нескінченним.

Але гри 2048 має й недоліки пов’язані з наступним:

- дуже спрощена графіка;

- відсутній повноцінний музичний супровід;
- наявність реклами;
- досить складний ігровий процес.

Це призводить до обмежених можливостей як розробки нових механік так й для удосконалення графічної частини гри.

Тому розробка та удосконалення елементів гри та реалізація їх на ігровому рушії Unreal Engine 4 дасть можливість покращити графічну частину гри так й додати нові режими, які будуть потребувати від користувача більшої концентрації та швидкості прийняття рішень.

3 АНАЛІЗ ТА РОЗРОБКА АЛГОРИТМІВ ГРИ 2048

Як показав аналіз гри - гра починається з появи нового блоку в будь якому випадковому місці. Після чого гравець повинен вирішити в якому напрямку потрібно почати рухати блоки й дати команду комп'ютеру за допомогою клавіатури. В результаті чого: якщо це можливо гра переміщує його в обраному напрямку та додає ще один блок. Й послідовність цих дій повторюються поки не зникне можливість виконати переміщення блоків в будь якому з напрямків.

На рисунку 3.1 приведено розроблений загальний алгоритм гри.

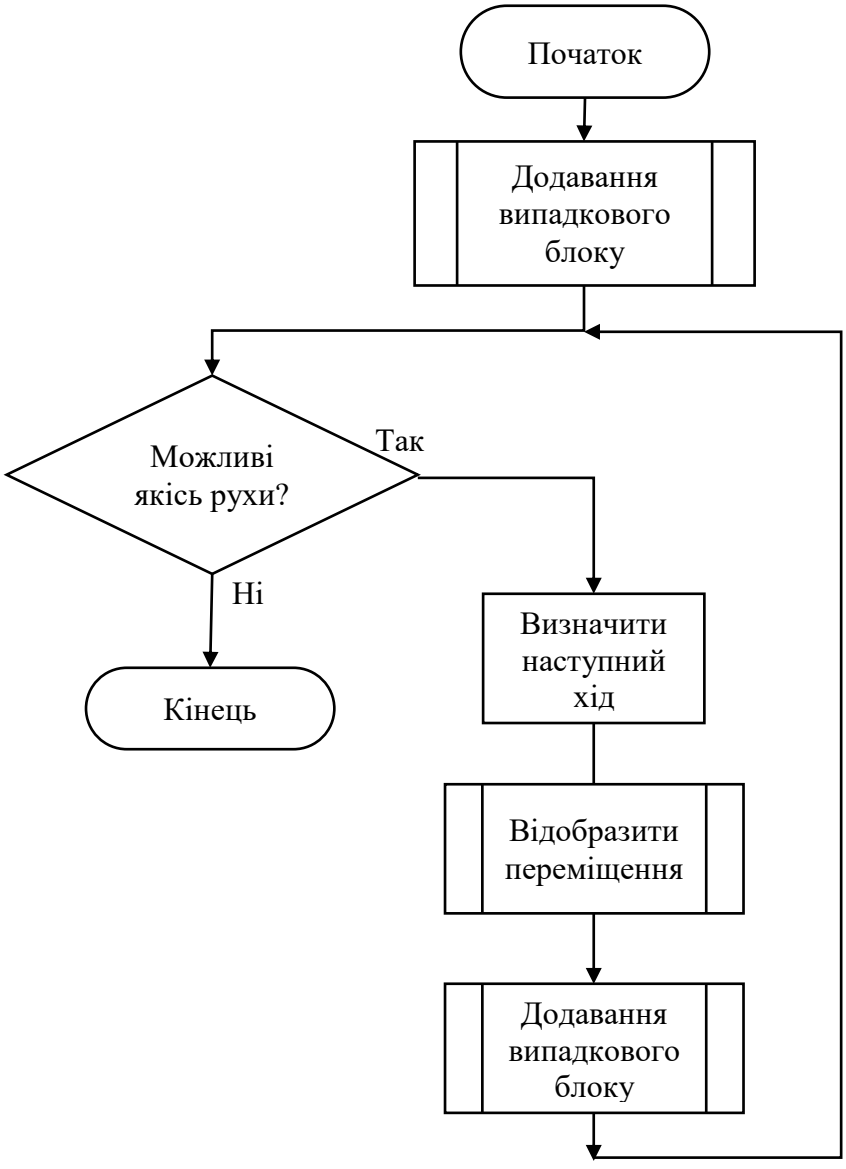


Рисунок 3.1 – Загальний алгоритм гри

При додаванні нового блоку повинно обиратись випадкове місце з загальної кількості вільних комірок. На рисунку 3.2 наведено розроблений алгоритм додавання випадкового блоку.

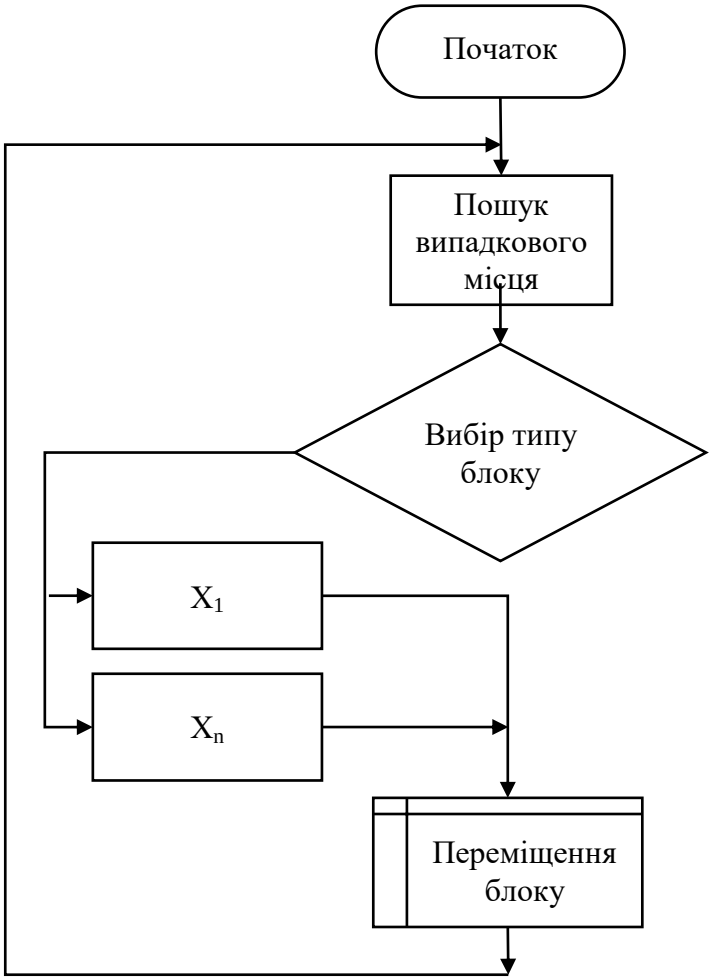


Рисунок 3.2 - Алгоритм додавання випадкового блоку

Кількість можливих місць появи блоку розраховується за формулою:

$$N = (A * B) - K \tag{3.1}$$

де A та B це кількість комірок по висоті й ширенні

K це кількість вже існуючих блоків.

В результаті чого ймовірність появи блоку в певній комірці розраховується за формулою:

$$P = {}^{1}/_{N} \quad (3.2)$$

Тобто при збільшенні кількості вже існуючих блоків ймовірність появи блоку в певній комірці збільшується.

Вибір типу блоку обирається видатковим чином й може керуватись за рахунок співвідношення між вірогідністю появи блоку «2» або «4».

Користувач за допомогою клавіатури має можливість вказати до якого краю потрібно перемістити усі блоки. На рисунку 3.3 приведено алгоритм підпрограми переміщення блоків.

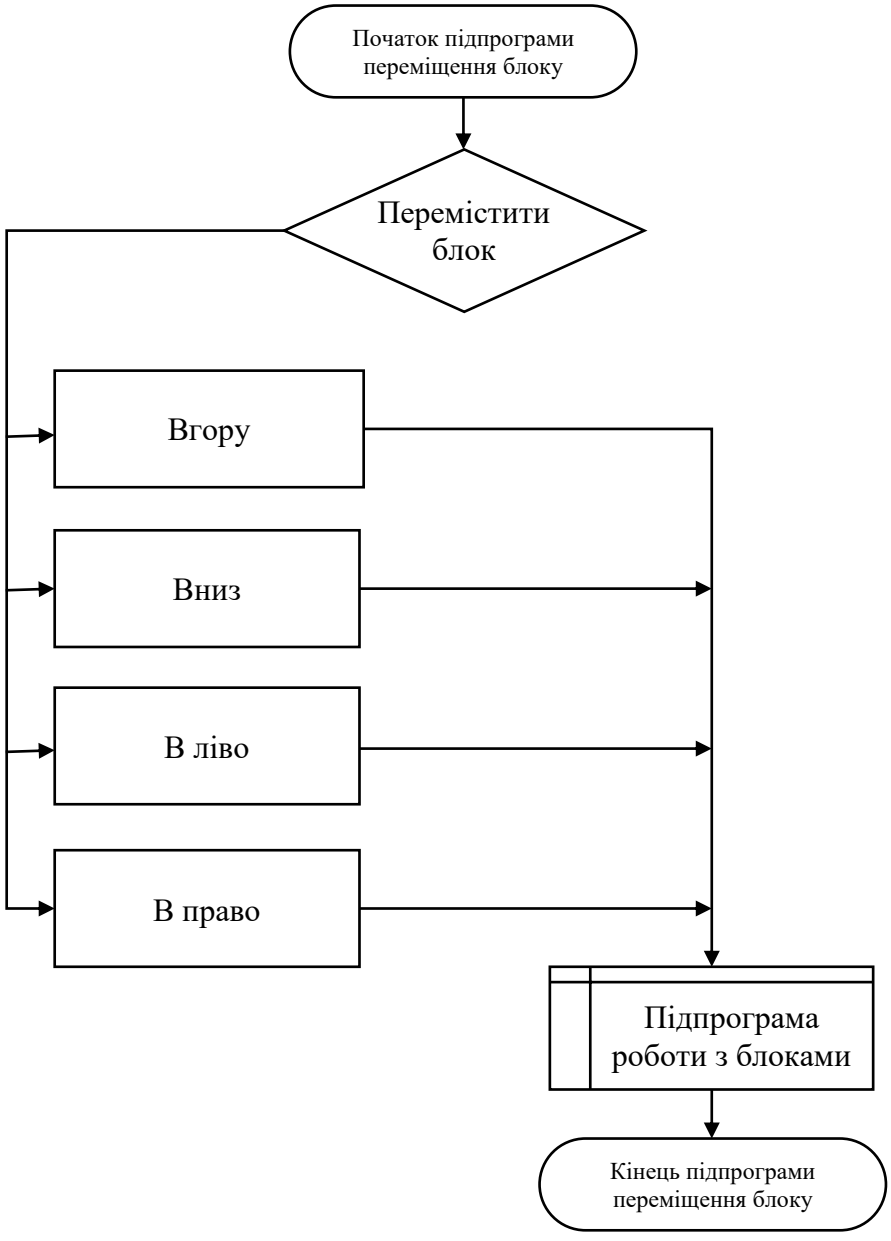


Рисунок 3.3 - Алгоритм підпрограми визначення напрямку переміщення блоків

Отримавши напрямок в якому вказано переносити усі блоки потрібно послідовно перенести блоки дотримуючись правильної послідовності перенесень. Послідовність взаємодії приведено на рисунку 3.4.

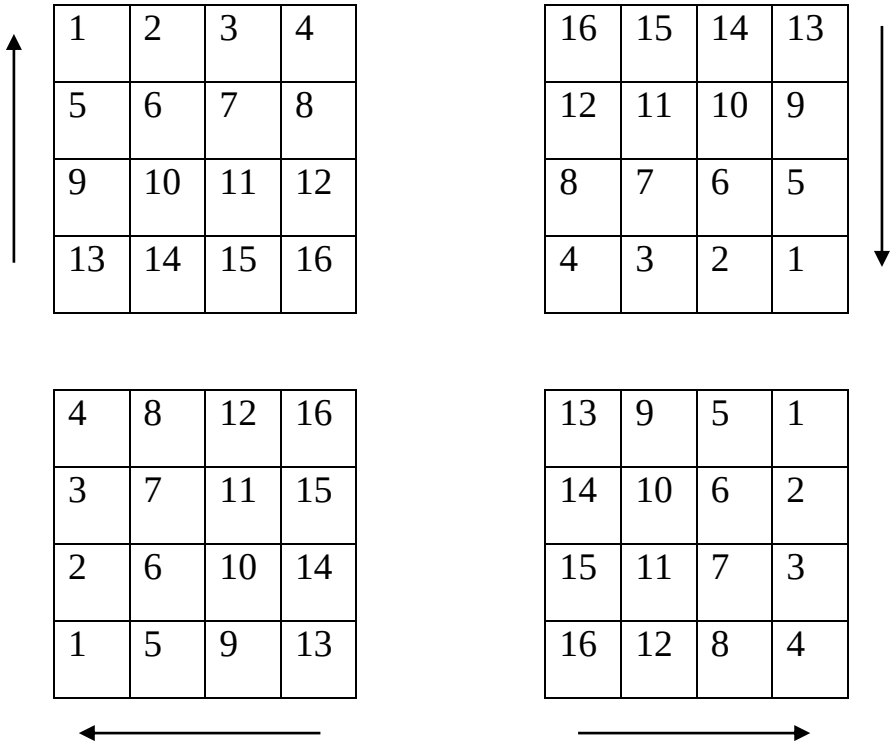


Рисунок 3.4 – Послідовність взаємодії з блоками
в залежності від напрямку переміщення

Наприклад якщо потрібно перенести блок розташований в комірці 9, то спочатку перевіряється чи можливо перенести його до комірки 5. Якщо можливо, то перевіряється можливість переміщення його до комірки 1. Якщо це можливо то зберігається інформація про переміщення блоку з комірки 9 до комірки 1. Якщо ні, то зберігається попередній вдалий результат переміщення, а саме з комірки 9 до комірки 5.

Таким чином незалежно від напрямку переміщення алгоритм прийняття рішення залишиться однаковим та універсальним для будь якого

напрямку та кількості комірок. На рисунку 3.5 приведено отриманий алгоритм прийняття рішення про переміщення блоків.

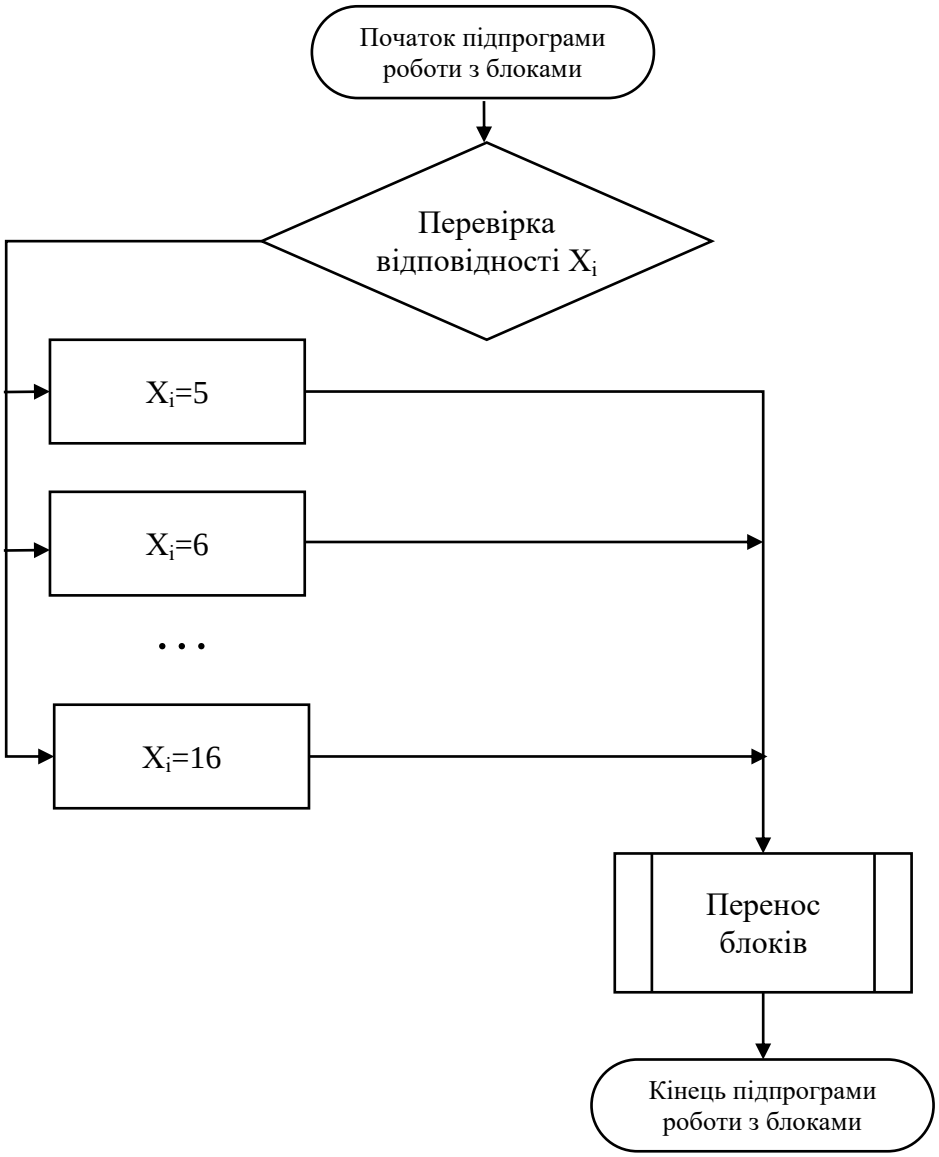


Рисунок 3.5 – Алгоритм переміщення блоків в потрібному напрямку

Можливість переміщення визначається за рахунок виконання умов:

– Чи існує блок в комірці куди буде переноситись блок?
– Чи однакові наминали блоків що перетинаються?
– Чи поєднувались блоки на цьому кроці?

В ході роботи було розроблено загальний алгоритм гри якій вирішує основні задачі. Алгоритм додавання випадкового блоку вирішує задачу розміщення блока у вільному місці та наданні йому числа. Підготовлено

алгоритми визначення напрямку та послідовності переміщень блоків. Визначені правила можливості переміщення та об’єднання блоків. Реалізація цих алгоритмів здасть логіку гри й її правила. Алгоритм є універсальним для вирішення поставленої задачі, та може працювати при зміні вхідних параметрів. А також, не залежить від візуальної частини, що дає можливість використовувати його при будь якому візуальному наповнені гри.

4 ПРОЄКТУВАННЯ ТА УДОСКОНАЛЕННЯ ГРИ CONTAINER 2048 YARD

4.1 Удосконалення графічного наповнення гри Container 2048 Yard

Для поліпшення графічної частини гри було використано [37] тривимірні моделі які були використані для створення меню та рівнів гри. Загалом було використано 48 тривимірних моделей для яких використовувалось 45 матеріалів текстур.

Для меню гри було підготовлено 3D сцену яка буде використовуватись як фон та середа для взаємодії є елементами меню які вбудовані в сцену. На рисунку 4.1 представлено загальний вид 3D сцени.

Рисунок 4.1 - Загальний вигляд сцени для меню гри

Для рівнів гри було розроблено 3D сцену яка буде використовуватись як фон та середа для взаємодії з механіками гри. На рисунку 4.2 представлено загальний вигляд 3D сцени.

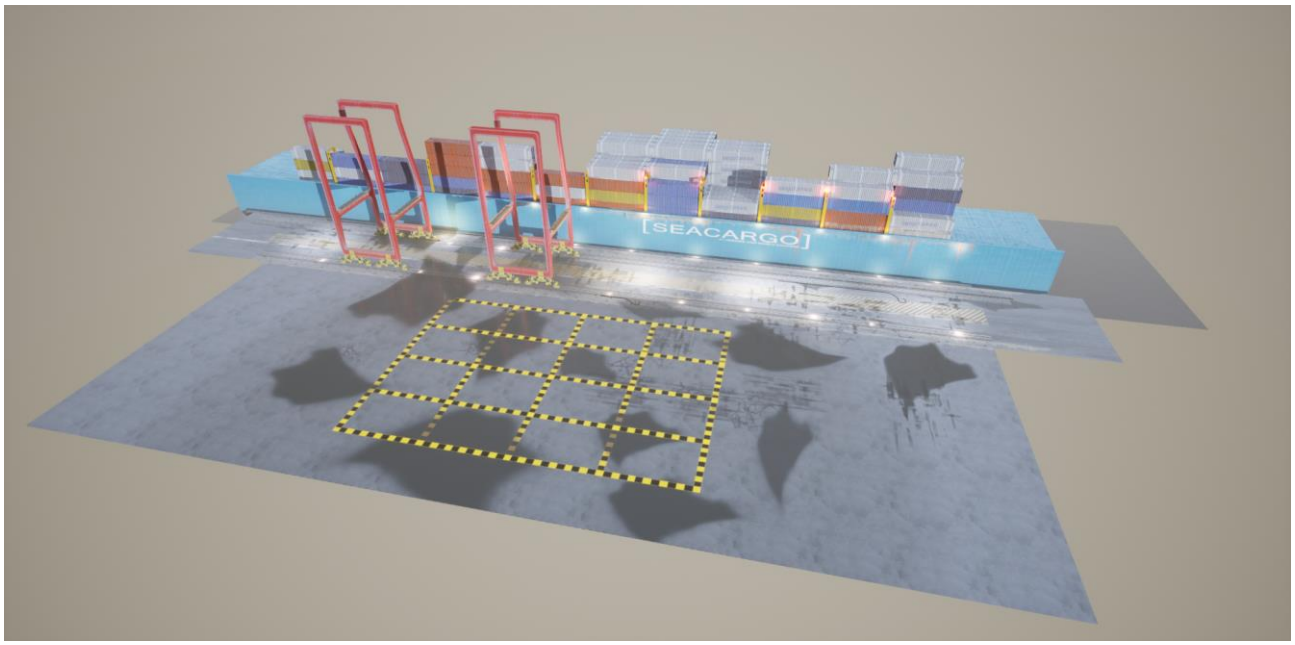


Рисунок 4.2 - Загальний вигляд сцени для рівнів гри

Для усіх рівнів налаштовано освітлення та шейдери для рейдеру зображення, в результаті чого рівні мають нічну атмосферу. Приклад вигляду результату рендеру зображення в грі приведено на рисунку 4.3.

Рисунок 4.3 – Приклад зображення з гри

Додаток в кожен блок вбудовано правила анімації переміщення блоків, Blueprint програми представлено на рисунку 4.4, їх анімації при надходженні вказівки на переміщення в нову точку простору.

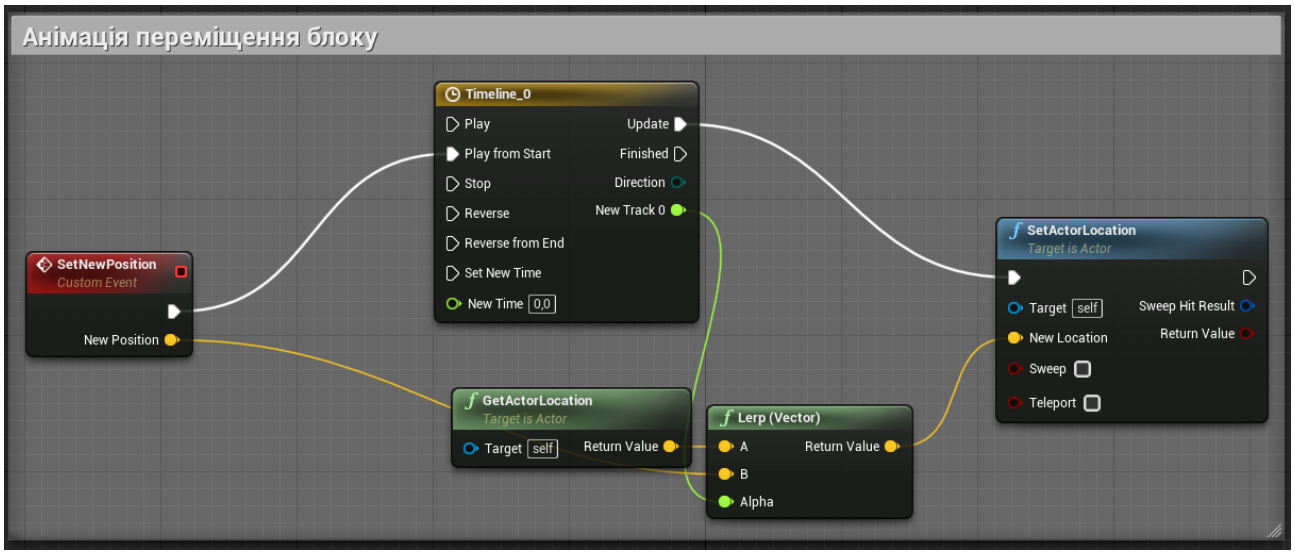


Рисунок 4.4 – Blueprint анімації переміщення блоків

Також поява нових блоків та зміна їх ціни була запрограмована за допомогою Blueprint наведеного на рисунку 4.5.

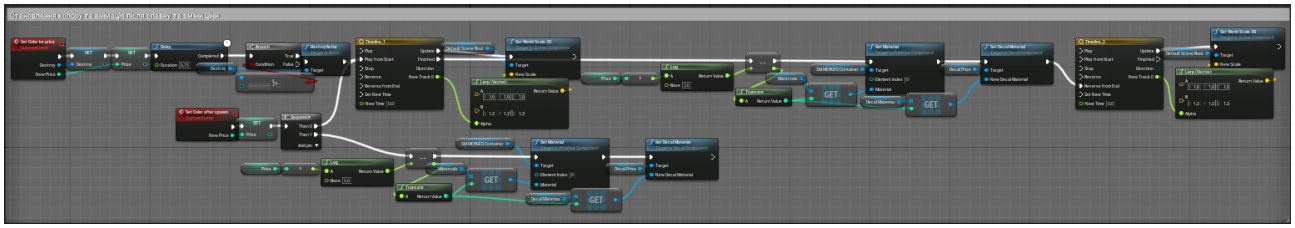


Рисунок 4.5 – Blueprint анімації появи та зміни “ціни” блоку

В результаті данні дії поліпшили візуальну частину гри.

4.2 Створення меню гри

При запуску гри камера демонструє 3D сцену й у користувача є можливість видрати порідний йому пункт меню з запропонованих, приведених на рисунку 4.6.

– New Game;
– Settings;
– Credits;
– Exit Game.

При наведенні показника миші на пункт меню колір тексту змінюється, а при натисканні камера плавно переміщаються до іншої зони та демонструє інші пукни меню.

Рисунок 4.6 – Вигляд основного меню гри

Після натискання на пункт меню «New Game» камера переміститься в іншу зону й з’явиться можливість обрати режим гри, або повернутись до попереднього меню. Вигляд меню вибору режиму гри приведено на рисунку 4.7.

Рисунок 4.7 – Меню вибору режиму гри

Алгоритм роботи меню реалізовано за допомогою Blueprint в якому при запуску гри користувачу надається можливість керувати покажчиком миші та відбувається прив’язка цільових точок переміщення та пунктів меню які може натиснути користувач. Це потрібно для зручності створення нових та редагування існуючих пунктів меню. Загальний вигляд Blueprint запуску гри приведено на рисунку 4.8.

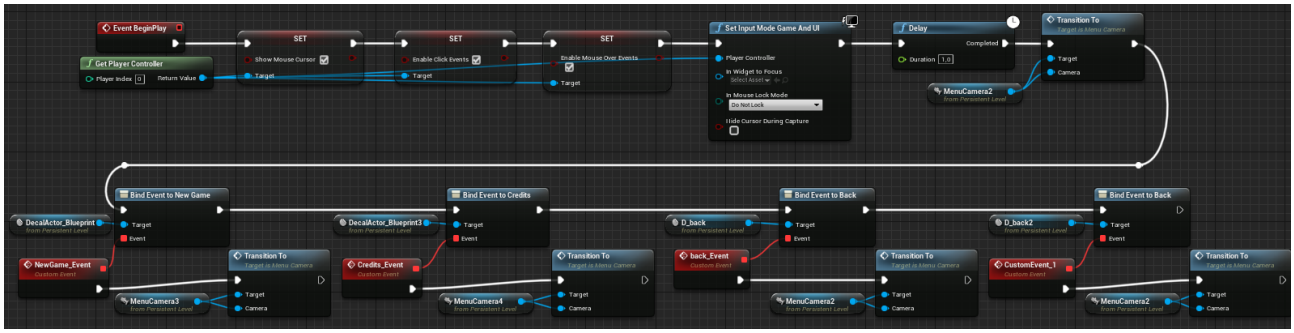


Рисунок 4.8 – Blueprint запуску гри

Реакція пунктів меню реалізовано за допомогою декількох команд Blueprint які однакові для будь якого меню, що дає можливість використовувати його для швидкого додавання нових розділів меню. Приклад реалізації цих команд для пункту меню «NewGame» приведено на рисунку 4.9.

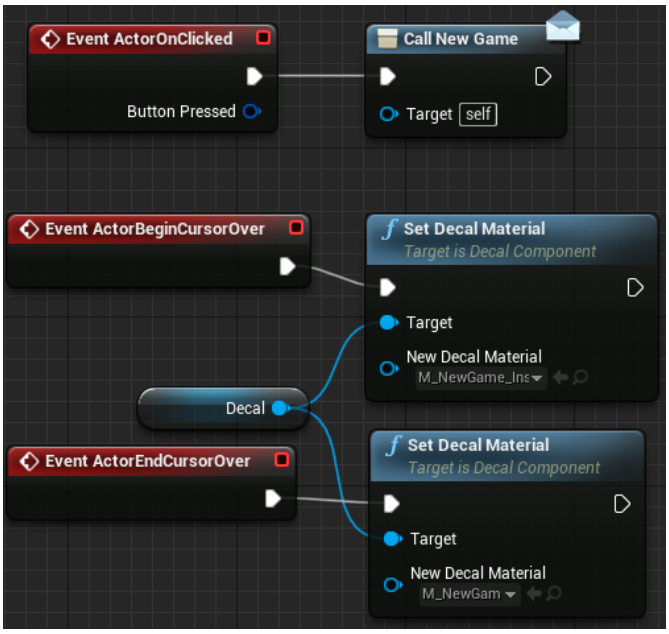


Рисунок 4.9 – Blueprint реалізації відгуків гри
на дії користувача з пунктами меню

При натисканні на будь який пункт меню камера анімовано переміщується до цільової точки прив'язка якої відбувалась при запуску гри. Реалізовано анімацію переміщення за допомогою Blueprint команди "Set View Target with Blend". Цей Blueprint універсальний й має вигляд представлений на рисунку 4.10.

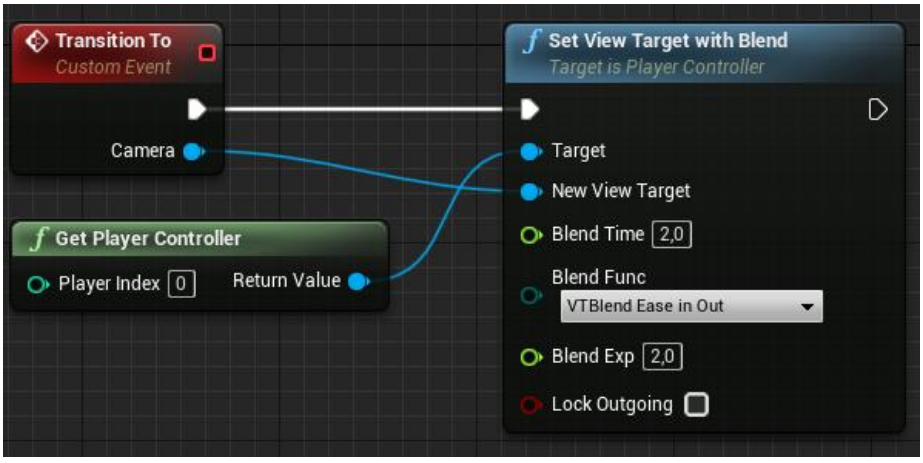


Рисунок 4.10 – Blueprint реалізації переміщення камери між розділами меню

Коли користувач вибирає один режим гри з запропонованих: Classic Game, Speed Game, DLC Game, завантажується один рівнів гри.

4.3 Реалізація алгоритму методами візуального програмування Blueprint

Classic Game реалізує основні механіки гри та є основою для інших режимів. Основний Blueprint гри представлено на рисунку 4.11. Blueprint складається з 11 основних блоків які відповідають за свої завдання:

- Початок гри/вихід з паузи;
- Виклик меню паузи;
- Перевірка програшу;
- Випадковий спавн у вільному місці;
- Встановлення “ціни” блоку;
- Курування з клавіатури;
- Виклик спавну нового блоку;
- Перевірка чи існує блок;
- Пошук точки переміщення;
- Переміщення та складання блоків;
- Виклик спавну нового блоку.

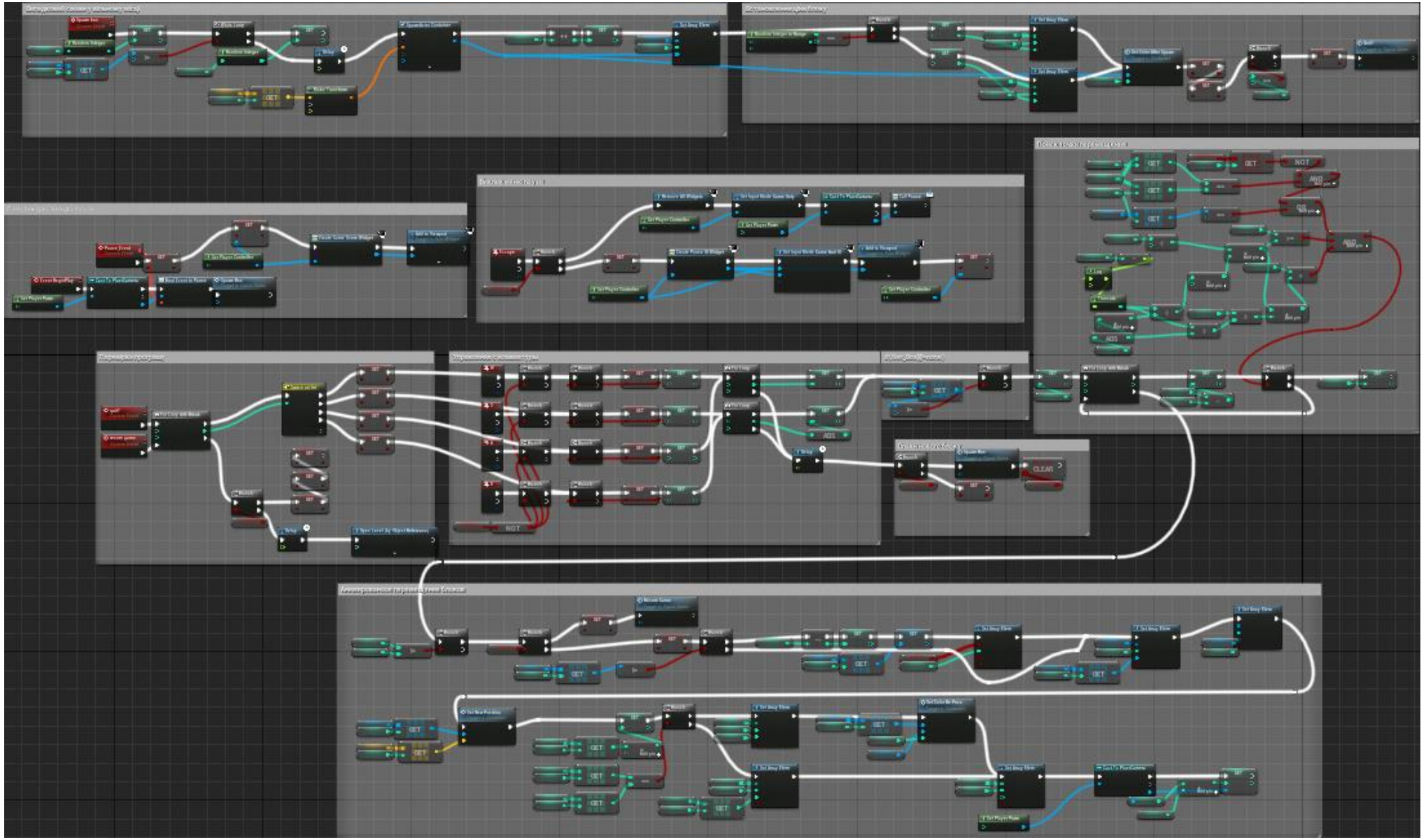


Рисунок 4.11 - Основний Blueprint режиму «Classic Game»

При завантаженні рівня гри виконується переключення системи керування та викликається спавн нового блоку. Для оптимізації цей блок також відповідає за вихід з меню паузи. Реалізація блоку Blueprint приведено на рисунку 4.12.

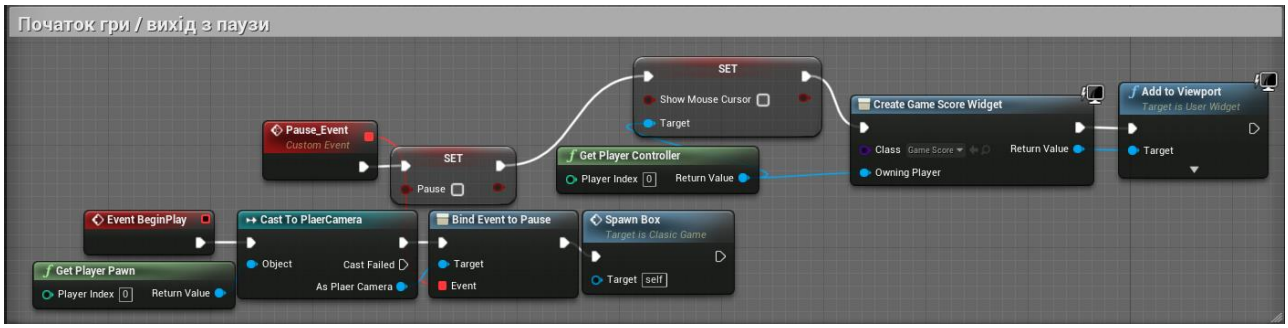


Рисунок 4.12 - Blueprint початку гри та виходу з паузи

За потреби користувач має можливість відкрити меню паузи щоб почати заново або вийти з гри в основне меню. Меню паузи викликається за допомогою клавіши Escape в результаті чого перемикається режим керування та на екран виводиться інтерфейс. Блок який відповідає за ці дії приведено на рисунку 4.13.

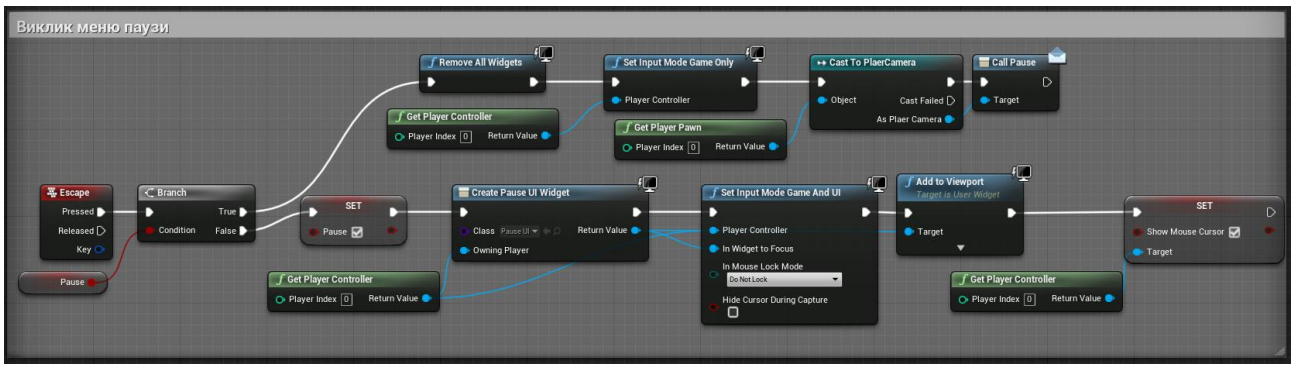


Рисунок 4.13 - Blueprint виклику меню паузи

Після того як рівень завантажився викликається блок якій відповідає за спавн блоків. В цьому блоці випадковим чином обирається одне з 16 можливих місць появи блоку, перевіряється чи вільне це місце, якщо вільне -

виконується його спавн та занесення даних в масив. Реалізація цих дій приведена на рисунку 4.14.

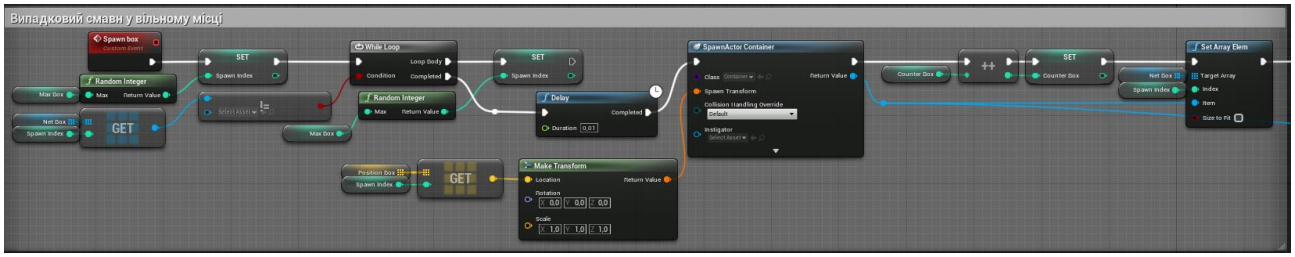


Рисунок 4.14 - Blueprint випадкового спавну у вільному місці

Після прийняття грою рішення про спавн нового блоку йому випадково надається “ціна” 2 або 4 з вірогідністю 75% та 25% відповідно. Після чого ці данні зберігаються та викликається анімація появи нового блоку. Перевіряється чи не був це 16 блок на рівні, якщо так то буде викликано перевірку на програш. Блок який відповідає за ці дії приведено на рисунку 4.15.

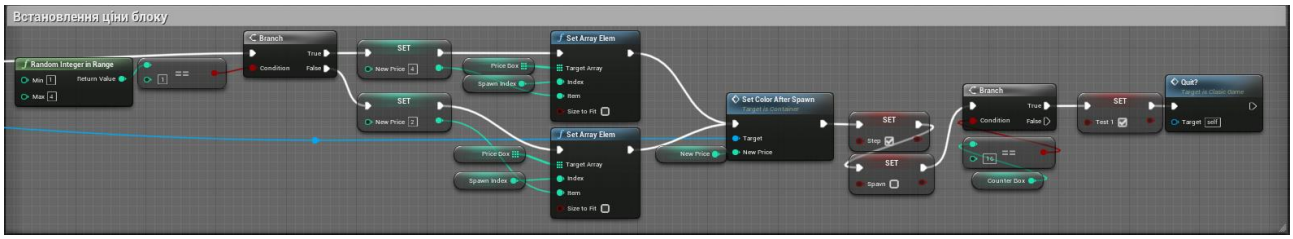


Рисунок 4.15 - Blueprint встановлення “ціни” блоку

У випадку якщо було створено шістнадцятий блок гра перевіряє чи існують варіанти подальших дій гравця. Виконується це перебором можливих дій гравця, якщо ні одна з чотирьох можливих дій не дасть результату гар буде вважатись програною. Blueprint перебору можливих дій приведено на рисунку 4.16.

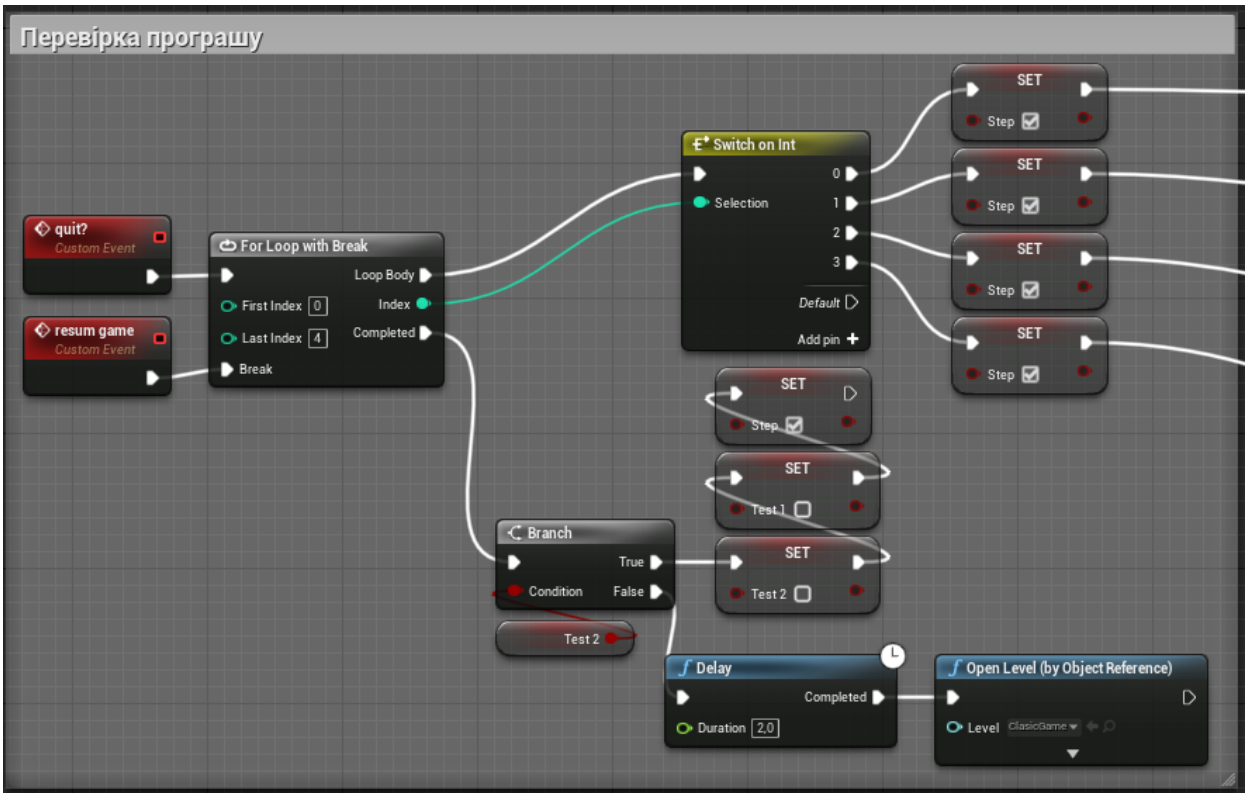


Рисунок 4.16 - Blueprint перевірки програшу

Керування грою виконується за допомогою клавіш WASD клавіатури. При натисканні обирається напрямок переміщення блоків в результаті чого виконується їх послідовне переміщення, якщо це можливо, відносно того краю до якого вони переміщуються, першим переміщується той блок що ближче до краю. Реалізація керування грою приведено на рисунку 4.17.

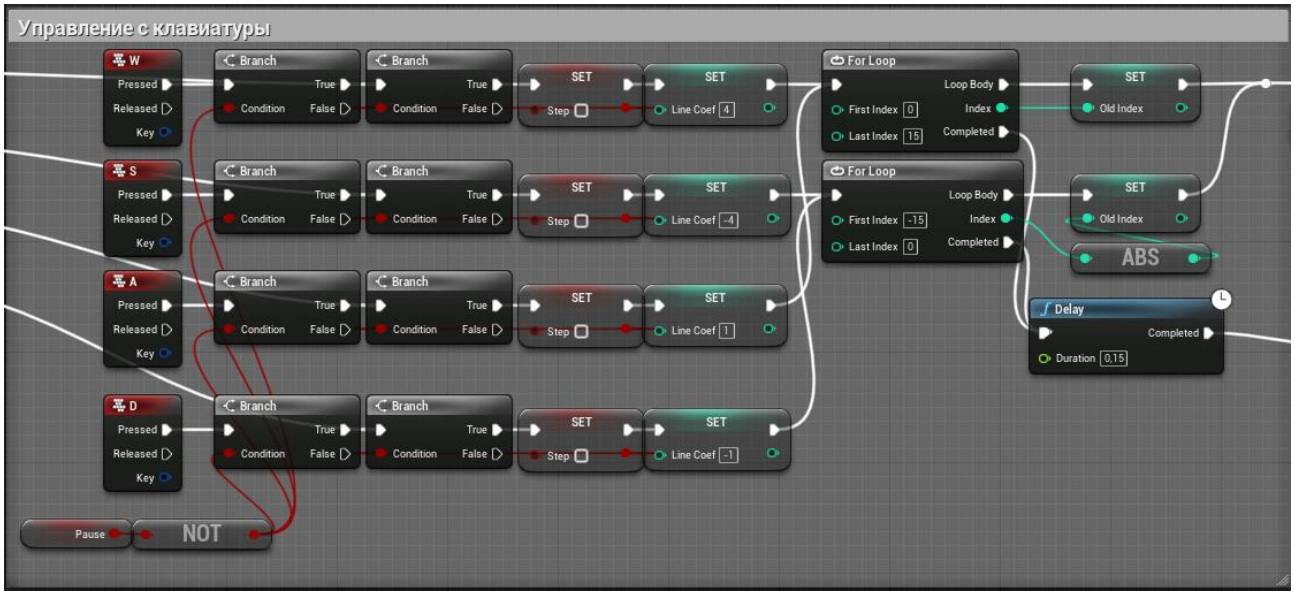


Рисунок 4.17 - Blueprint курування грою з клавіатури

Перед пошуком комірки куди переміщувати блок виконується перевірка чи є в комірці блок, якщо там нічого нема він пропускається. Реалізація перевірки приведена на рисунку 4.18.

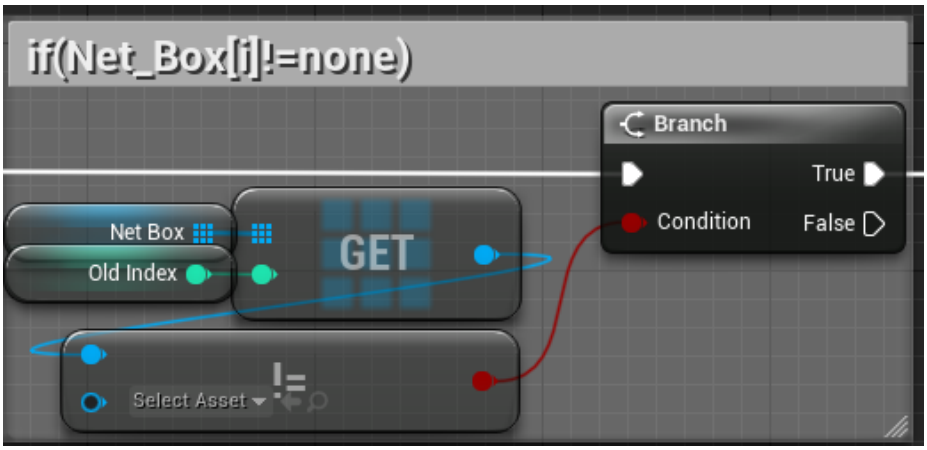


Рисунок 4.18 - Blueprint перевірки чи існує блок

Якщо блок існує гра шукає для нього комірку куди переміщувати враховуючи, що за один крок кожен блок одноразово має можливість об’єднатись з блоком того ж номіналу. Blueprint роботи з масивами даних та пошук точок переміщення приведено на рисунку 4.19.

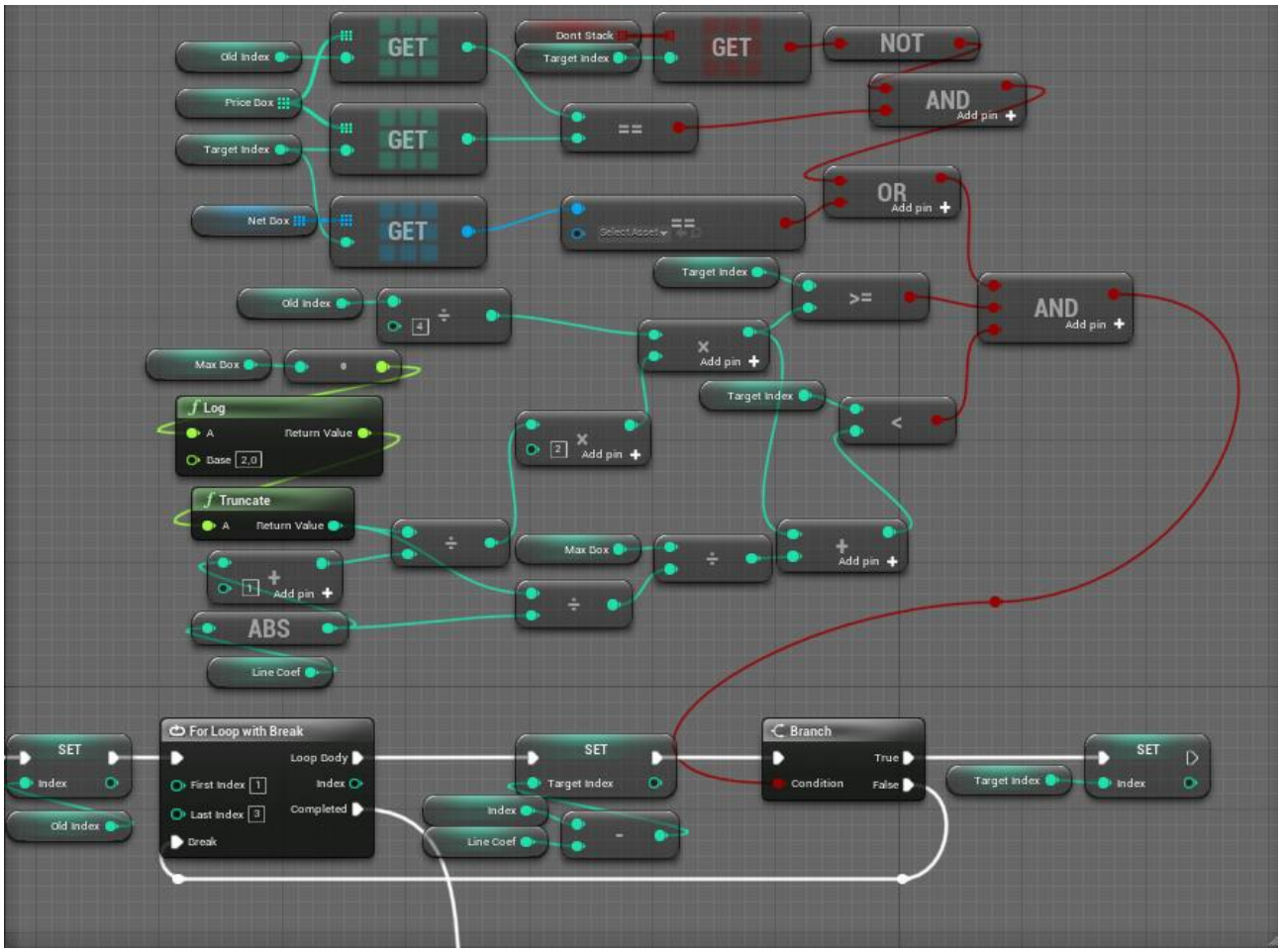


Рисунок 4.19 - Blueprint пошуку точки переміщення блоку

Коли було знайдено нову точку розташування блока виконуються перевірки, внесення змін в масиви даних, підрахунок об’єднань блоків та виклик анімації відмальовування змін, що були виконані в матрицях даних. Blueprint цих дій приведено на рисунку 4.20.

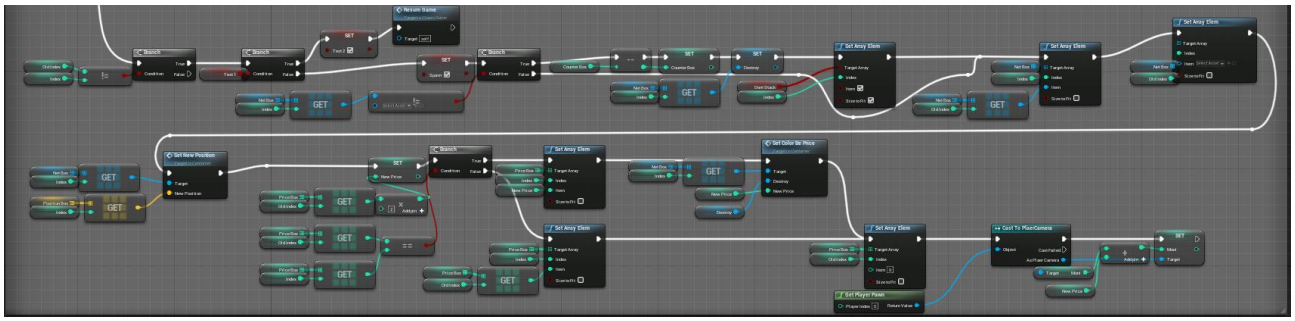


Рисунок 4.20 - Blueprint переміщення та складання блоків

Після виконання дій над усіма блоками виконується спавн нового блоку й цикл гри замикається й повторюється до програшу. Виклик спавну нового блоку приведено на рисунку 4.21.

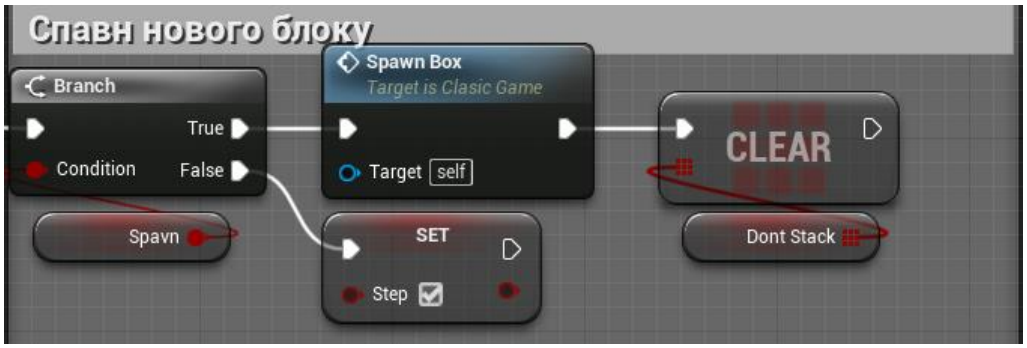


Рисунок 4.21 - Blueprint виклику спавну нового блоку

Завдяки приведеним вище Blueprint реалізується стандартний режим гри який стає основою до модифікації та додавання нових механік гри.

4.4 Створення режиму гри на швидкість

Speed Game базується на стандартному режимі гри, але для реаліззації механік гри на швидкість деякі блоки програми перероблені. Blueprint складається з 8 основних блоків, 5 з яких перероблено:

- початок гри/вихід з паузи;
- виклик меню паузи;
- випадковий спавн у вільному місці;
- встановлення “ціни” блоку;
- переміщення та складання блоків.

При завантаженні рівня «Speed Game» виконується переключення системи керування та активуються таймер спавну нового блоку. Для оптимізації цей блок також відповідає за вихід з меню паузи. Реалізація блоку Blueprint приведено на рисунку 4.22.

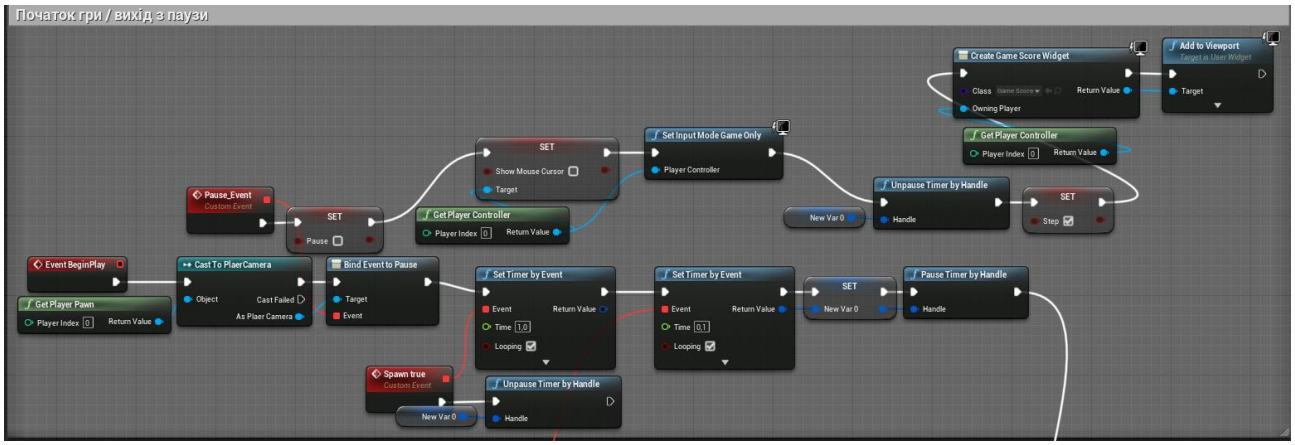


Рисунок 4.22 - Blueprint початку гри та виходу з паузи

Меню паузи викликається за допомогою клавіши Escape. В результаті чого перемикається режим керування, на екран виводиться інтерфейс та зупиняється таймер спавну нового блоку. Блок який відповідає за ці дії приведено на рисунку 4.23.

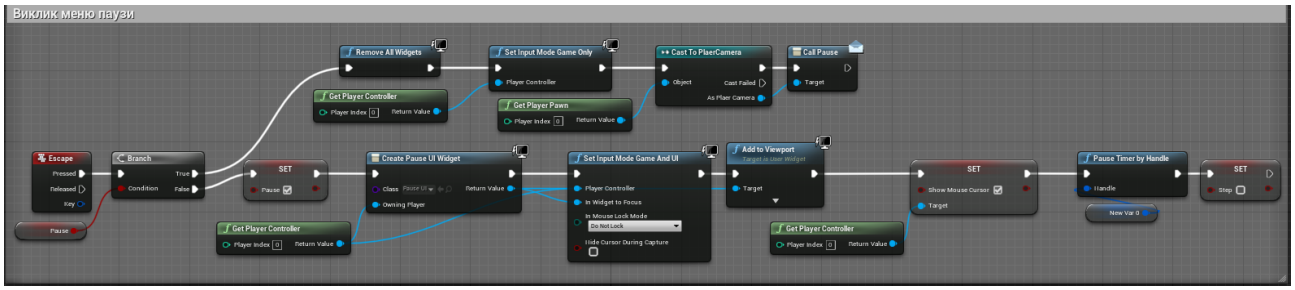


Рисунок 4.23 - Blueprint виклику меню паузи

Таймер викликає блок якій відповідає за спавн блоків. В цьому блоці випадковим чином обирається одне з 16 можливих місць появи блоку, перевіряється чи вільне це місце, якщо вільне - виконується його спавн та занесення даних в масив. Реалізація цих дій приведена на рисунку 4.24.

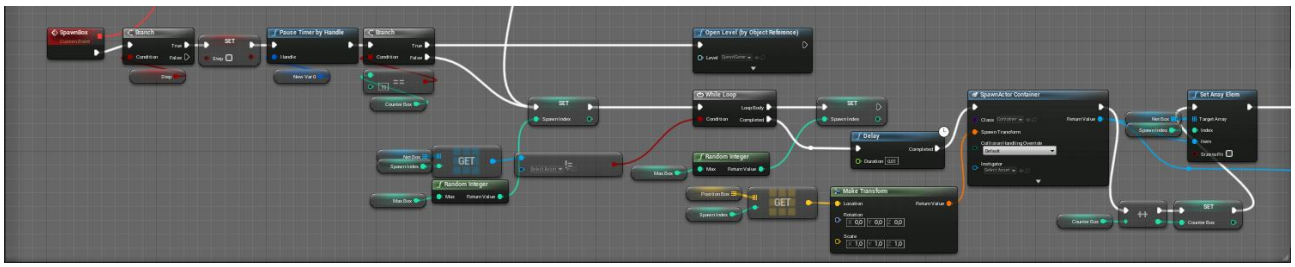


Рисунок 4.24 - Blueprint випадкового спавну у вільному місці

Після спрацьовування таймера та прийняття рішення про місце спавну нового блоку йому випадково надається “ціна” 2 або 4 з вірогідністю 75% та 25% відповідно. Після чого ці данні зберігаються та викликається анімація появи нового блоку. Блок який відповідає за ці дії приведено на рисунку 4.25.

На відміну від стандартного режиму на цьому етапі перевірка, чи не був це 16 блок на рівні не відбувається, через те що це перевіряється на попередньому кроці.

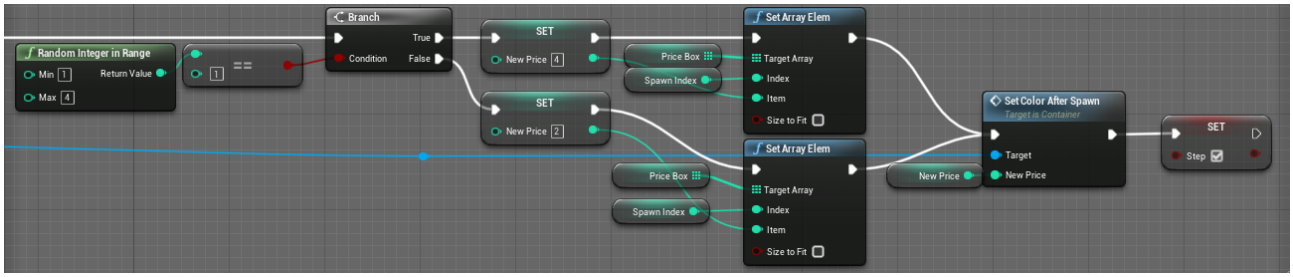


Рисунок 4.25 - Blueprint встановлення “ціни” блоку

Перевірки: внесення змін в масиви даних, підрахунок об’єднань блоків та виклик анімації відмальовування змін, що були виконані в матрицях даних. Blueprint цих дій приведений на рисунку 4.26 відрізняється від класичного режиму через відсутність деяких перевірок.

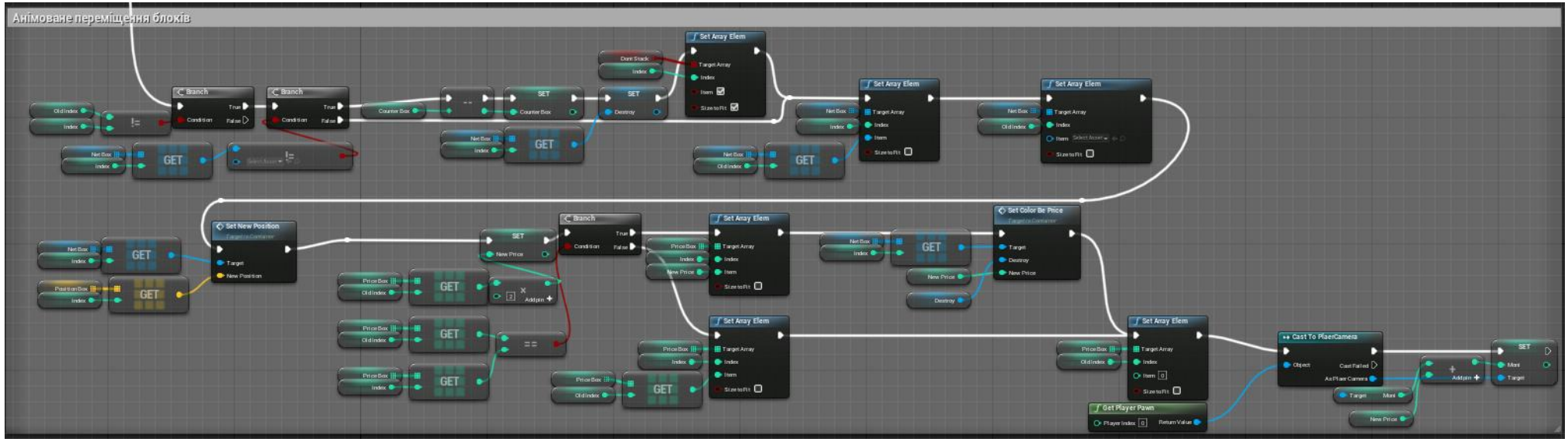


Рисунок 2.26 - Blueprint переміщення та складання блоків

Завдяки приведеним вище змінам в Blueprint реалізовано режим гри на швидкість який базується на механіках стандартного режиму гри.

4.5 Додавання до гри можливості видалення блоку з рівня

Для реалізації можливості видалення блоку з рівня реалізовані деякі нові механіки гри основані на стандартному режимі гри. Blueprint складається з 12 основних блоків. З яких перероблено 3 блоки, а також додано новий блок який відповідає за можливість видалення блоку з рівня. До вбудованих механік блоків додано нові можливості, які візуалізують наведення покажчику миші та безпосередньо видаляють блок з гри.

При завантаженні рівня гри налаштовуються зв’язки подій. Але блок паузи було відокремлено, хоча одразу після запуску гри цей Blueprint також виконується. Реалізація блоку Blueprint приведено на рисунку 4.27.

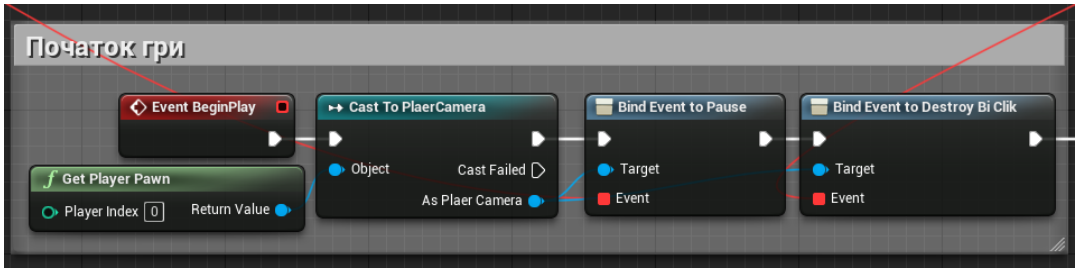


Рисунок 4.27 - Blueprint початку гри

При виході з паузи перемикається режим керування та вимикається візуалізація наведення покажчику миші на блок. Блок який відповідає за ці дії приведено на рисунку 4.28.

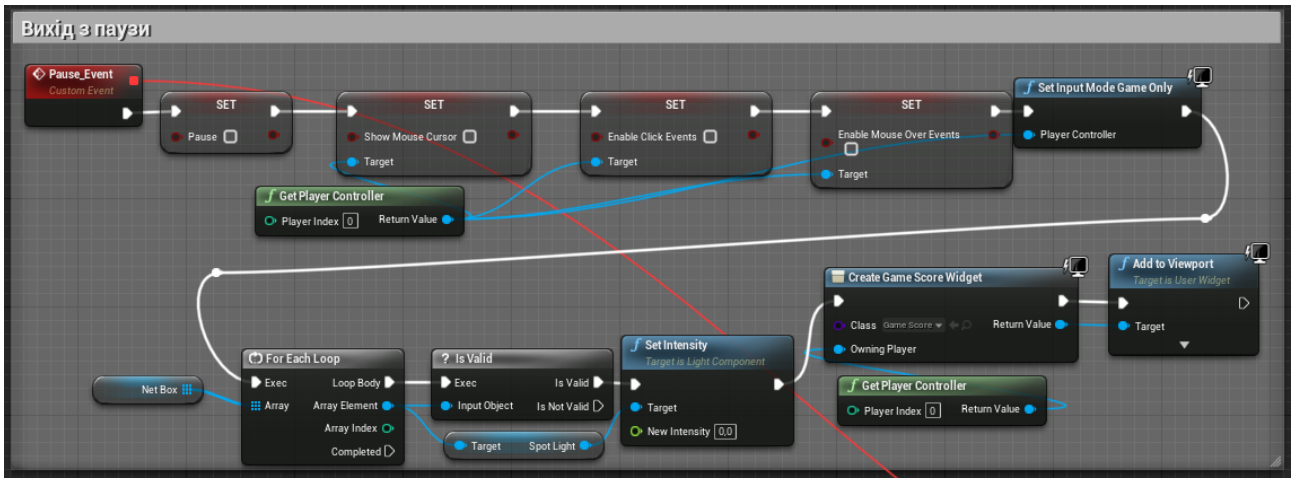


Рисунок 4.28 - Blueprint виходу з меню паузи

При виклику меню паузи перемикається режим керування та на екран виводиться інтерфейс. Додатково вмикається реакція гри на наведення покажчику миші на блоки за для візуалізації цієї події. Блок який відповідає за ці дії приведено на рисунку 4.29.

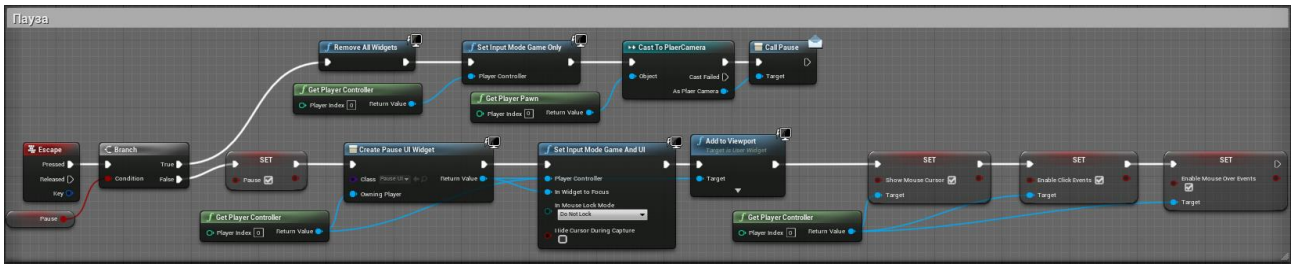


Рисунок 4.29 - Blueprint виклику меню паузи

Новий блок в грі відповідає за видалення даних про блок при натисканні на нього покажчиком миші та передання цьому блоку команди на видалення з гри. Реалізація блоку Blueprint приведено на рисунку 4.30.

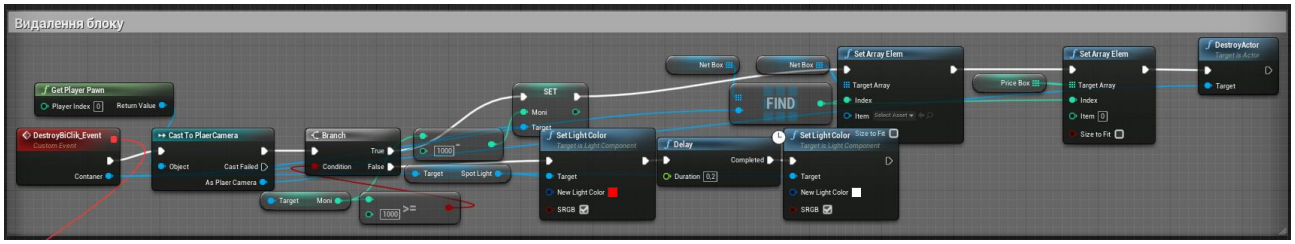


Рисунок 4.30 - Blueprint керування видалення блоку

До вбудованих в кожен блок команд додано нові механіки які відповідають за підсвічування блоку при наведені на нього покажчику миші, приведеного на рисунку 4.31, та видалення цього блоку з гри за потреби, показаного на рисунку 4.32.

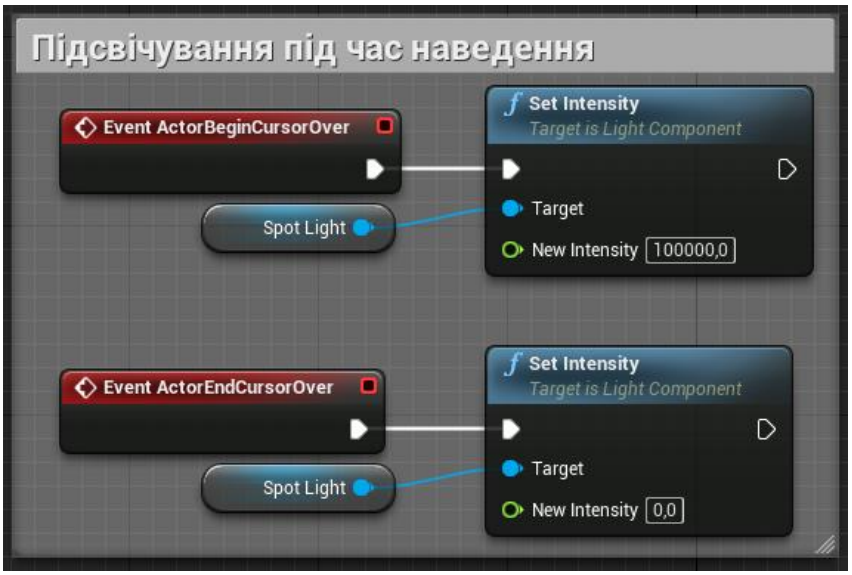


Рисунок 4.31 - Blueprint підсвічування блоку

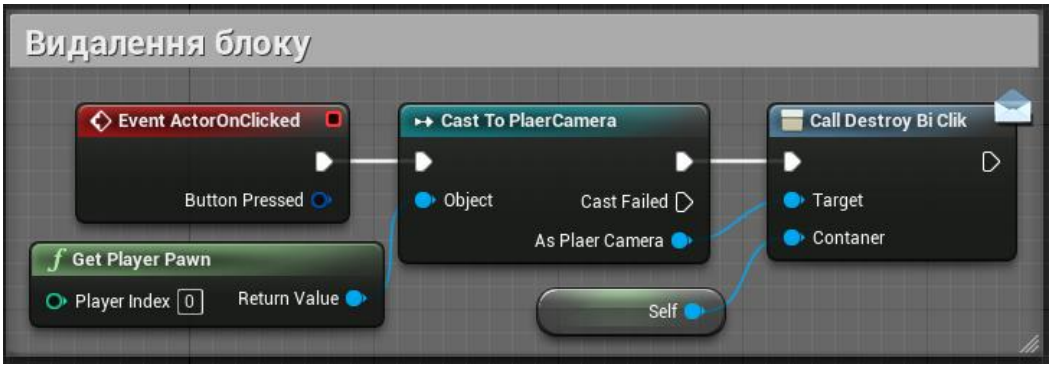


Рисунок 4.32 - Blueprint видалення блоку

Завдяки приведеним вище змінам та додаванням нових блоків в Blueprint реалізовано режим гри в якому є можливість видаляти блоки з гри залишаючи основні механіки стандартного режиму гри.

В результаті роботи була підготовлена графічна частина гри та розроблені основні механіки гри які можуть послужити в подальшому створені нових режимів гри. Це можливо, так як основа гри є універсальною та легко модифікуються й обгортається новими механіками.

ВИСНОВКИ

Було проведено аналіз існуючих ігрових рушіїв в якості платформи для розробки гри для дипломного проєкту було обрано Unreal Engine. Тому що він має: зручний візуальний редактор із потужною функціональністю; націленість на масштабні проекти; завдяки візуальному програмуванню – «блюпринтам» – прототипування не вимагає технічних навичок, а для розширення можливостей движка можна застосувати програмування на C++; гарна оптимізація; доступ до вихідного коду; багатий маркетплейс із безліччю ассетів під будь-які потреби; є безкоштовним та має велику кількість матеріалів, що спрощує початок роботи в ньому.

Розглянуто підходи до алгоритмізації, що буде потрібно при подальшому створенні механік гри засобами Blueprint.

Та було проведено аналіз існуючих видів графіки у комп'ютерних іграх який вказав на доцільність переходу до біль сучасних типів графіки в іграх, а саме тривимірної графіки.

До достоїнств гри 2048 можна віднести: унікальна реалізація ідеї розробників; найпростіше керування за допомогою свайпів; захоплюючий геймплей; геймплей є майже нескінченним. Але гри 2048 має й недоліки пов’язані з наступним: дуже спрощена графіка; відсутній повноцінний музичний супровід; наявність реклами; досить складний ігровий процес. Це призводить до обмежених можливостей як розробки нових механік так й для удосконалення графічної частини гри.

Тому розробка та удосконалення елементів гри та реалізація їх на ігровому рушії Unreal Engine 4 дасть можливість покращити графічну частину гри так й додати нові режими, які будуть потребувати від користувача більшої концентрації та швидкості прийняття рішень.

Розроблено алгоритм гри якій вирішує основні задачі:

- додавання випадкового блоку;

– визначення напрямку та послідовності переміщень блоків;

– визначені правила можливості переміщення та об’єднання блоків.

Алгоритм є універсальним для вирішення поставленої задачі, та може працювати при зміні вхідних параметрів. А також, не залежить від візуальної частини, що дає можливість використовувати його при будь якому візуальному наповнені гри.

В результаті роботи була підготовлена графічна частина гри та розроблені основні механіки гри на ігровому рушії Unreal Engine 4.

Для гри було створено:

- 48 тривимірних моделей;
- 45 матеріалів текстур.

На ігровому рушії було створено 4 різні рівні гри, серед них: меню гри; classic game; speed game; dlc game. Розроблені основні механіки гри можуть послужити в подальшому створені нових режимів гри. Це можливо, так як основа гри є універсальною та легко модифікуються й обгортається новими механіками.

ПЕРЕЛІК ПОСИЛАНЬ

1. Jacobson J., Lewis M. Game engine virtual reality with CaveUT //Computer. – 2005. – Т. 38. – №. 4. – С. 79-82.
2. A Comparative Example Between The Use Of Pca And Mds For Image Classification / Hernandez, W., Mendez, A., Flor-Unda, O., Camejo, I.M., Kolendovska, M.// IEEE International Symposium on Industrial Electronics, 29th IEEE International Symposium on Industrial Electronics, ISIE 2020; Delft; Netherlands; 17 June 2020 до 19 June 2020; Volume 2020-June, June 2020, № 9152565, Pages 1353-1358
3. Algorithm For Generating Refined Frequency Estimates In Atmospheric Radio Sounding Systems / Kartashov V., Hernandez W., Hernandez-Balbuena D., M. Kolendovska, Konovalenko O., Melnyk V.// IEEE International Symposium on Industrial Electronics, 29th IEEE International Symposium on Industrial Electronics, ISIE 2020; Delft; Netherlands; 17 June 2020 до 19 June 2020; Volume 2020-June, June 2020, № 9152562, Pages 79-82
4. Cossu S. M. Game Development With Game Maker Studio 2. – Apress, 2019.
5. Application of Fast Frequency Shift Measurement Method for INS in Navigation of Drones / D. Avalos-Gonzalez, D.H. Balbuena, V. Tyrsa, V.M. Kartashov, M. Kolendovska, S. Sheiko, O. Sergiyenko, V. Melnyk, F.N. Murrieta-Rico // IECON 2018 – 44th Annual Conference of the IEEE Industrial Electronics Society. – P. 3159–3164.
6. Avalos-Gonzalez, D., Sergiyenko, O., Hernandez-Balbuena, D., Tyrsa,V., Kartashov V.M., V.,Rivas-Lopes, M., Murrieta-Rico, F.N. Constraints definition and application optimization based on geometric analysis of the frequency measurement method by pulse coincidence// Measurement: Journal of the International Measurement Confederation (USA). 2018, V.126. P. 184-193.

7. Haas J. K. A history of the unity game engine //Diss. WORCESTER POLYTECHNIC INSTITUTE. – 2014. – T. 483. – C. 484.
8. Book “Control and Signal Processing Applications for Mobile and Aerial Robotic Systems”, Hardback - Advances in Computational Intelligence and Robotics English. Edited by Oleg Sergiyenko, Moises Rivas-Lopez, Wendy Flores-Fuentes, Julio Cesar Rodríguez-Quiñonez, Lars Lindner. Editorial IGI Global, Hershey, United States, January 2020, 340 páginas. ISBN10 152259924X, ISBN13 9781522599241
9. Cesar Sepulveda-Valdez ; Oleg Sergiyenko ; Vera Tyrsa ; Wendy Flores-Fuentes ; Julio César Rodríguez-Quiñonez ; Fabian Natanael Murrienta-Rico ; Jesús Elías Miranda-Vega ; Paolo Mercorelli ; Marina Kolendovska. "Geometric analysis of a laser scanner functioning based on dynamic triangulation," 2020 IEEE 29th International Symposium on Industrial Electronics (ISIE), Delft, Netherlands, 17-19 of June 2020, pp. 1398-1403, doi: 10.1109/ISIE45063.2020.9152268. https://ieeexplore.ieee.org/abstract/document/9152268
10. Cuauhtémoc Mariscal-García; Wendy Flores-Fuentes; Daniel Hernández-Balbuena; Julio C. Rodríguez-Quiñonez ; Oleg Sergiyenko. "Classification of Vehicle Images through Deep Neural Networks for Camera View Position Selection," 2020 IEEE 29th International Symposium on Industrial Electronics (ISIE), Delft, Netherlands, 17-19 of June 2020, pp. 1376-1380, doi: 10.1109/ISIE45063.2020.9152440. https://ieeexplore.ieee.org/abstract/document/9152440
11. Developing and Applying Optoelectronics in Machine Vision/ O. Sergiyenko, J.C. Rodriguez-Quiñonez, IGI Global, 2016; 341p.
12. Experimental estimation of direction finding to unmanned air vehicles algorithms efficiency by their acoustic emission, /Oleynikov, V., Zubkov, O., Kartashov, V., ...Sheiko, S., Babkin, S.//2019 IEEE International Scientific-Practical Conference: Problems of Infocommunications Science

and Technology, PIC S and T 2019 - Proceedings, 2019, стр. 175-178, 9061337

13. Qiu W., Yuille A. Unrealcv: Connecting computer vision to unreal engine //European Conference on Computer Vision. – Springer, Cham, 2016. – С. 909-916.
14. Features of acoustic noise of small unmanned aerial vehicles / Semenets, V.V., Kartashov, V.M., Leonidov, V.I. //Telecommunications and Radio Engineering (English translation of Elektrosvyaz and Radiotekhnika), 2020, 79(11), стр. 985-995
15. Geometric Analysis Of A Laser Scanner Functioning Based On Dynamic Triangulation /Sepulveda-Valdez, C., Sergiyenko, O., Tyrsa, V, Mercorelli, P., Kolendovska, M.// IEEE International Symposium on Industrial Electronics, 29th IEEE International Symposium on Industrial Electronics, ISIE 2020; Delft; Netherlands; 17 June 2020 до 19 June 2020; Volume 2020-June, June 2020, № 9152268, Pages 1398-1403

https://ieeexplore.ieee.org/abstract/document/9152255

https://ieeexplore.ieee.org/document/9161870

16. I. Y. A. Corpus, L.Lindner, O.Sergiyenko. "Transimpedance Amplifier for Laser Scanning System Range Extension," 2020 IEEE 29th International Symposium on Industrial Electronics (ISIE), Delft, Netherlands, 17-19 of June 2020, pp. 1421-1426, doi: 10.1109/ISIE45063.2020.9152487. https://ieeexplore.ieee.org/abstract/document/9152487
17. Ivanov, M., Sergiyenko, O., Mercorelli, P., Hernandez, W.c, Rodriguez Quinonez, J.C.d, Katashov V., Kolendovska, M., Iryna, T. Effective informational entropy reduction in multi-robot systems based on real-time TVS. IEEE International Symposium on Industrial Electronics, 2019-June,8781209, с. 1162-1167.
18. Jonathan J. Sanchez-Castro ; Julio C. Rodríguez-Quiñonez ; Luis R. Ramírez-Hernández ; Guillermo Galaviz ; Daniel Hernández-Balbuena ; Gabriel Trujillo-Hernández ; Wendy Flores-Fuentes ; Paolo Mercorelli ;

Wilmar Hernández-Perdomo ; Oleg Sergiyenko ; Félix Fernando González-Navarro. "A Lean Convolutional Neural Network for Vehicle Classification," 2020 IEEE 29th International Symposium on Industrial Electronics (ISIE), Delft, Netherlands, 17-19 of June 2020, pp. 1365-1369, doi: 10.1109/ISIE45063.2020.9152274. https://ieeexplore.ieee.org/abstract/document/9152274

19. Lindner, L., Sergiyenko, O., Rivas-López, M., (...), Gurko, A., Kartashov, V.M. Machine vision system for UAV navigation; IEEE, 2016 International Conference on Electrical Systems for Aircraft, Railway, Ship Propulsion and Road Vehicles and International Transportation Electrification Conference, ESARS-ITEC, 2016; pp.1–6. DOI: 10.1109/ESARS-ITEC.2016.7841356.
20. M. Ivanov, O. Sergiyenko, V. Tyrsa, P. Mercorelli, V. Kartashov, W. Hernandez, S. Sheiko, M. Kolendovska. Individual scans fusion in virtual knowledge base for navigation of mobile robotic group with 3D TVS // Proceedings of 44th Annual Conference of IEEE Industrial Electronics Society (IECON).. -2018. – Washington DC, USA. -S. 3187-3192. . ISBN 978-1-5090-6683-4/18/.
21. Murrieta-Rico, F.N., Petranovskii, V., Galvan, D.H., Sergiyenko, O., Yocupicio-Gaxiola, R.I., De Dios Sanchez-Lopez, J. Phase effect in frequency measurements of a quartz crystal using the pulse coincidence principle. 2020 IEEE 29th International Symposium on Industrial Electronics (ISIE), Delft, Netherlands, 17-19 of June 2020, pp. 185-190, 9152255, DOI: 10.1109/ISIE45063.2020.9152255
22. Oleksandr Sotnikov, Vladimir Kartashov, Oleksandr Tymochko, Oleg Sergiyenko, Vera Tyrsa, Paolo Mercorelli, Wendy Flores-Fuentes. Methods for Ensuring the Accuracy of Radiometric and Optoelectronic Navigation Systems of Flying Robots in a Developed Infrastructure. Chapter 16// Machine Vision and Navigation; Springer, Cham. pp.537–578. Editors: Sergiyenko, Oleg, Flores-Fuentes, Wendy, Mercorelli, Paolo. DOI: 10.1007/978-3-030-22587-2_16.

23. Optical detection of unmanned air vehicles on a video stream in a real-time/Kartashov, V., Oleynikov, V., Zubkov, O., Sheiko, S.// 2019 International Conference on Information and Telecommunication Technologies and Radio Electronics, UkrMiCo 2019 - Proceedings, 2019, 9165362/
24. Principles Of Construction And Assessment Of Technical Characteristics Of Multi-Frequency Atmospheric Sodar In The Humidity Measurement Mode / Kartashov, V.M., Sidorov, G.I., Sheiko, S.A., Kolendovskaya, M.M., Sergienko, O.Yu. // Telecommunications And Radio Engineering (English Translation Of Elektrosvyaz And Radiotekhnika), 2020, ISSN Print: 0040-2508, ISSN Online: 1943-6009, DOI: 10.1615/TelecomRadEng.v79.i4.50, p. 323-333/
25. Research Of The Uncertainty Of Measurement Frequencies And Definitions Of The Frequency Signal In The Waveguide With Respect To Power / Semenets, V.Zakharov, I. Serhiienko, M., Kartashov, V.M, , Kolendovska, M., Hernandez, W., Hipolito, J.I.N., , Tyrsa, V.// 45th Annual Conference of the IEEE Industrial Electronics Society, IECON 2019; Lisbon Congress CenterLisbon; Portugal; 14 October 2019 до 17 October 2019; CFP19IEC-ART; Код 155980, Volume 2019-October, October 2019, № 8927203, Pages 4674-4679
26. Spatial-Temporal Processing Of Acoustic Signals Of Unmanned Aerial Vehicles /Kartashov V.M., Oleinikov V.N., Zubkov O.V., Sheiko S.A., Kolendovska M.M.// Telecommunications And Radio Engineering (English Translation Of Elektrosvyaz And Radiotekhnika), 2020, ISSN Print: 0040-2508, ISSN Online: 1943-6009, DOI: 10.1615/Telecomradeng.v79.i9.40, p. 769-780
27. Stereoscopic Vision Systems In Machine Vision, Models, And Applications (Book Chapter)/ Ramírez-Hernández, L.R., Rodríguez-Quiñonez, J.C., Castro-Toscano, M.J., Kolendovska, M., Murrieta-Rico,

F.N.// Machine Vision And Navigation, 2019 Machine Vision and Navigation30 September 2019, Pages 241-265

28. StrelkovaT., KartashovV., Lytyuga A., Strelkov A. Theoretical Methods of Images Processing in Optoelectronic Systems. Chapter 16. // Biometrics: Concepts, Methodologies, Tools, and Applications; Oleg Sergiyenko and Julio C. Rodriguez-Quiñonez. (341p.), IGI Global, 2017; pp. 361-381. DOI: 10.4018/978-1-5225-0983-7.ch016.
29. StrelkovaT., KartashovV., Lytyuga A.,StrelkovA. Theoretical Methods of Images Processing in Optoelectronic Systems. Chapter 6// Developing and Applying Optoelectronics in Machine Vision; Oleg Sergiyenko and Julio C. Rodriguez-Quiñonez. (341p.) – USA, Herhey, IGI Global, 2016; pp.180-205.
30. Sytnik O., KartashovV. Methods and Algorithms for Technical Visionin Radar Introscopy. Chapter 13// Optoelectronics in Machine Vision-Based Theories and Applications. IGI Global, 2019; pp. 373-391.
31. The Use of Factorization and Multimode Parametric Spectra in Estimating Frequency and Spectral Parameters of Signal/Semenets, V., Kartashov, V., Sergiyenko, O., ...Rodriguez-Quinonez, J.C., Flores-Fuentes, W.//IEEE International Symposium on Industrial Electronics, 2020, 2020-June, p. 215-219
32. Unda, O.F., Hernandez, W., Vargas, O., Mendez, A., Sergiyenko, O., Tyrsa, V. Construction of a robotic platform of differential type for first-year students of electronic engineering, 2020 International Symposium on Power Electronics, Electrical Drives, Automation and Motion, SPEEDAM 2020, 24-26 de junio de 2020, Sorrento, Italia, pp. 538-543, 9161870, DOI: 10.1109/SPEEDAM48782.2020.9161870
33. Use of Acoustic Signature for Detection, Recognition and Direction Finding of Small Unmanned Aerial Vehicles/Kartashov, V., Oleynikov, V., Koryttsev, I., ...Babkin, S., Selieznov, I.//Proceedings - 15th International Conference on Advanced Trends in Radioelectronics,

Telecommunications and Computer Engineering, TCSET 2020, 2020, p. 377-380/

34. 2048 – Додатки в Google Play [Електронний ресурс] URL: https://play.google.com/store/apps/details?id=com.gabrielecirulli.app2048&hl=ua&pli=1 (дата звернення: 08.11.2022).
35. Age of 2048™: City Merge Games – Додатки в Google Play. [Електронний ресурс] URL: https://play.google.com/store/apps/details?id=com.soulgit.age2048&hl=ua (дата звернення: 08.11.2022).
36. Острів Кітті Кіт:номер злиття – Додатки в Google Play. [Електронний ресурс] URL: https://play.google.com/store/apps/details?id=com.fungry.kittycatisland&hl=ua (дата звернення: 08.11.2022).
37. Container Yard Environment Set in Environments - UE Marketplace. [Електронний ресурс]
URL: https://www.unrealengine.com/marketplace/en-US/product/container-yard-environment-set (дата звернення: 15.11.2022).