МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ОДЕСЬКА ЮРИДИЧНА АКАДЕМІЯ»
ФАКУЛЬТЕТ КІБЕРБЕЗПЕКИ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Кафедра інформаційних технологій

КВАЛІФІКАЦІЙНА РОБОТА

на тему:

«СИМУЛЯТОР ОПЕРАТОРА НАДВОДНОГО ДРОНА»

Рівень «бакалавр»
Галузь знань 12 «Інформаційні технології»,
спеціальність 121 «Інженерія програмного забезпечення»
Виконав студент 4 курсу
Струк Нікіта Олександрович

Керівник к.т.н., доцент
Задерейко Олександр Владиславович

Рецензент к.пед.н., доцент
Логінова Наталія Іванівна

Одеса – 2024

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ “ОДЕСЬКА ЮРИДИЧНА АКАДЕМІЯ”

Факультет кібербезпеки та інформаційних технологій

Кафедра інформаційних технологій

Галузь знань 12 Інформаційні технології

Спеціальність 121 «Інженерія програмного забезпечення»

Назва освітньої програми «Інженерія програмного забезпечення»

ЗАТВЕРДЖУЮ
Завідувач кафедри інформаційних технологій
доцент Наталія Логінова _____
“____”_____ 2024 року

ЗАВДАННЯ
на кваліфікаційну (бакалаврську) роботу
здобувачу Струку Нікіті Олександровичу

1. Тема роботи: “ Симулятор оператора надводного дрона”
   керівник роботи: к.т.н, доцент Задерейко Олександр Владиславович
   затверджені наказом НУ «ОЮА» № 809-06 від 02 квітня 2024 року
2. Дата видачі завдання 15 вересня 2024 року
3. Строк подання здобувачем роботи 20 травня 2024 року

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів бакалаврської роботи	Строк виконання етапів роботи	Примітка
1	Огляд специфіки предметної області	16.09.2023 – 18.10.2023	
2	Аналіз вимог до симулятора	19.10.2023 – 24.11.2023	
3	Огляд симуляторів дронів	25.11.2023 – 24.12.2023	
4	Проєктування та розробка застосунку	25.12.2023 – 16.02.2024	
5	Тестування застосунку	17.02.2024 – 10.03.2024	
6	Аналіз результатів роботи	11.03.2024 – 28.03.2024	
7	Оформлення звітної документації	28.03.2024 – 19.05.2024	

Здобувач _____
(підпис) (ім’я та прізвище)

Керівник роботи _____
(підпис) (ім’я та прізвище)

АНОТАЦІЯ

Струк Н.О. Симулятор оператора надводного дрона. – Кваліфікаційна робота бакалавра за спеціальністю 121 «Інженерія програмного забезпечення».

Кваліфікаційна робота викладена на 99 сторінках, містить 3 розділи, 65 ілюстрацій, 3 таблиці, 17 використаних джерел.

Метою роботи є створення високоефективного, наближеного до реальності ігрового застосунку симулятора для спрощення підготовки, покрашення ефективності набуття нових та відточення наявних навичок пілотування надводних дронів,

Об’єкт дослідження – симулятори для операторів надводних дронів.

Предмет дослідження – засоби програмної розробки симулятора оператора надводного дрона засобами ігрового рушія Unreal Engine.

У кваліфікаційній роботі розроблено ігровий застосунок симулятора оператора надводного дрона шляхом використання рушія Unreal Engine з урахуванням специфіки роботи операторів надводних дронів; проаналізовано вимоги до ігрового застосунку; розглянуто існуючі симулятори дронів; виконано тестування розробленого застосунку.

Ключові слова: розробка ігрових застосунків, симулятор, безпілотний надводний апарат, C++, Unreal Engine, Blueprint.

ANNOTATION

Struk N. O. Simulator of a sea-surface drone operator. – Bachelor's qualification work in the specialty 121 "Software engineering".

The qualification work is set out on 99 pages, contains 3 chapters, 65 illustrations, 3 tables, 17 references.

The goal of the work is to develop a highly effective, realistic gaming application simulator to simplify training, improve the efficiency of acquiring new and sharpening existing skills of piloting surface drones.

The object of research is simulators for operators of surface drones.

The subject of the study is software development tools for a surface drone operator simulator using the Unreal Engine game engine.

In the qualification work, a game application for a surface drone operator simulator was developed using the Unreal Engine, taking into account the specifics of the work of surface drone operators; the requirements for the game application were analyzed; existing drone simulators were considered; the developed application was tested.

Keywords: game application development, simulator, unmanned surface vehicle, C++, Unreal Engine, Blueprint.

ПЕРЕЛІК СКОРОЧЕНЬ

БНА – безпілотний надводний апарат

ЗСУ – Збройні Сили України

МВС – Міністерство Внутрішніх Справ

НА – надводний апарат

ОС – операційна система

РЕБ – радіоелектронна боротьба

РЕР – радіоелектронна розвідка

ШІ – штучний інтелект

ЗМІСТ

ВСТУП

З початком повномасштабного війни в Україні одним із найбільших викликів була морська блокада російським чорноморським флотом портів. Через велику перевагу у кількості кораблів та у вогневій міці на боці росії можливі прямі зіткнення у водах Чорного моря були дуже ризикованими та неефективними у виконанні бойових завдань. Для цього були розробленні надводні безпілотні, керовані оператором апарати, які можуть виконувати різного роду задачі, а саме:

- знищення кораблів ворожого флоту;
- розвідка прибережних територій;
- доставка необхідного спорядження та провізії;
- розмінування;
- перевезення людей.

Разом з активним впровадженням новітніх технологій постала потреба у тренуванні особового складу практичного використання дронів у тих чи інших ситуаціях. Оскільки тренування керування дроном у реальному житті має велику кількість ризиків та значних витрат, найкращим рішенням є створення відповідного симулятора у цифровому просторі.

Тому вельми актуальними є роботи, присвячені розробкам симуляторів операторів надводного дрона, які мають на меті зниження витрат та зменшення ризиків при практичному навчанні майбутніх фахівців, а також створити міцну основу нових пілотів та покращити навички досвідчених.

Метою роботи є створення високоефективного, наближеного до реальності ігрового застосунку симулятора для спрощення підготовки, покрашення ефективності набуття нових та відточення наявних навичок пілотування надводних дронів.

Об’єкт дослідження – симулятори для операторів надводних дронів.

Предмет дослідження – засоби програмної розробки симулятора оператора надводного дрона засобами ігрового рушія Unreal Engine.

Відповідно до поставленої мети **завданнями** роботи є:

- аналіз сучасного стану індустрії дронів та виклики, з якими вона стикнулася;

– аналіз наявних симуляторів безпілотних надводних апаратів для з’ясування та уточнення функцій, структури, інтерфейсу, які матиме створюваний програмний продукт для якнайкращого задоволення вимог користувача;

– формування функціональних та нефункціональних вимог до створюваного програмного продукту;

– дизайн ігрового процесу з урахуванням можливостей дрона, а також проєктування перешкод та пасток і моделювання поведінки ворогів;

– розробка графічного інтерфейсу;

– розробка серверної частини з використанням ASP.NET

Застосунок буде забезпечувати широкий спектр можливостей застосування наявних моделей дронів у різних сценаріях, який зменшить ризики та витрати, а також спростить та покращить ефективність підготовки майбутніх пілотів безпілотних апаратів

Методи дослідження – аналітичні з використанням комп’ютерних технологій.

Практичне значення результатів дослідження: зменшення ризиків та витрат на підготовку пілотів, водночас покращення ефективності тренувань шляхом наближення ігрового функціоналу до реальних сценаріїв.

Публікації та апробація кваліфікаційної роботи:

1. Задерейко О. В., Толокнов А. А., Струк Н. О. Розробка ігрового застосунку «симулятор оператора надводного дрона». *Сучасні технології в енергетиці, електромеханіці, системах керування та машинобудуванні:* матер. VI Всеукр. наук.-практ. інтернет-конф. (м. Харків, 06-07 грудня 2023 р.) Харків: ННПІ УІПА, 2023. С. 11-12. URL: https://hdl.handle.net/11300/26945/

2. Трофименко О.Г., Струк Н.О. Специфіка тестування ігрових застосунків. *Українське суспільство та наука в умовах воєнного стану й євроінтеграційних перетворень: виклики, напрями відновлення і розвитку*: у 2 т. : матер. Всеукр. наук.-практ. конф. (м. Одеса, 20 травня 2023 р.) Одеса : Видавництво «Юридика», 2023. С.470-472. URL: https://dspace.onua.edu.ua/server/api/core/bitstreams/5b59bf6e-927b-4a81-9459-da85a6e6ff05/content

1 ОГЛЯД СПЕЦИФІКИ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Аналіз предметної області

1.1.1 Визначення пілотування надводних дронів та його особливості

Безпілотний надводний апарат (БНА) або надводний дрон (НД) – човен або корабель, який працює на поверхні води без екіпажу. БНА працюють із різними рівнями автономності, від дистанційного керування до повністю автономних надводних апаратів.

Залежно від специфіки задач безпілотні водні апарати поділяють на:

– спостережні – оснащенні камерами та датчиками для збору візуальних даних і даних про навколишнє середовище, часто використовуються для розвідки і моніторингу;

– картографічні та геодезичні – використовуються для створення детальних карт, проведення топографічних зйомок і збору геопросторових даних;

– екологічно-моніторингові – оснащені датчиками для вимірювання якості повітря і води, а також для виявлення забруднювачів, ці дрони допомагають у моніторингу навколишнього середовища і збереженні природи;

– вантажні – створені для транспортування товарів та матеріалів на короткі та середні відстані, ці дрони пропонують альтернативний засіб логістичного транспортування у віддалені або важкодоступні райони;

– охоронні та оборонні – використовуються для спостереження за периметром, патрулювання кордонів і в оборонних цілях, ці дрони підвищують безпеку, відстежуючи і реагуючи на потенційні загрози;

– науково-дослідні – оснащенні науковими приладами і датчиками, підтримують різні дослідницькі дисципліни, в тому числі екологію, метеорологію і океанографію, збираючи дані у віддаленому або небезпечному середовищі;

– підводні – оснащені камерами і датчиками для дослідження підводного середовища. Використовуються для морської археології, підводних інспекцій та розмінування деяких видів мін [3].

Ще наприкінці Другої світової війни дистанційно керовані надводні дрони використовувалися Військово-Морськими Силами Сполучених Штатів Америки для пошуку та розміновування водних ділянок. У двадцять першому столітті прогрес у системах керування та навігаційних технологіях призвів до появи нових дронів, якими оператор може керувати дистанційно з суші або сусіднього судна.

У контексті російсько-української війни БНА знайшли застосування у військовій сфері, а саме для:

- знищення військових кораблів;
- розвідки прибережних територій;
- транспортування необхідного спорядження та провізії;
- розмінування;
- перевезення людей на інший берег.

Таблиця 1.1. Сфери застосування безпілотних надводних апаратів

Сфера застосування	**Характеристика**
Знищення військових кораблів	Російський Чорноморський Флот значно переважає за кількістю кораблів та за вогневою міццю Український Чорноморський Флот, тому прямі зіткнення призвели б до великих втрат серед екіпажу кораблів та неефективності виконання бойових задач зі сторони України. Надводні дрони камікадзе, здатні наносити тяжкі ушкодження та знищувати навіть найбільші судна, оскільки вони носять у собі протикорабельну бомбу, яка детонує в таких випадках: автоматично при зіткненні носовою частиною (сигнал подається через датчики на носі) та через важкі ушкодження дрона, а також вручну, якщо цю функцію активує пілот.
Розвідка прибережних територій	Маючи як візуальне, так і радіолокаційне маскування, дрони здатні непомітно пропливати, передаючи через відеокамеру по одному видів зв’язку інформацію у реальному часі. Таким чином, дрони здатні проводити ефективно розвідку берегових територій. Можлива затримка поміж показом відео і реальними подіями, в залежності від відстані пульту керування до дрона.
Доставка необхідного спорядження та провізії	Маючи розмір невеличкого військового катера та високу швидкість пересування, БНА здатні перевозити необхідне спорядження та провізію до місця призначення за короткі терміни.
Розмінування	Слід зазначити, що для розмінування виділяють два типи дронів: розвідницькі та тральні. Перші дрони бувають підводними (використовують відеокамеру для пошуку мін) та надводними (використовують ехолокацію для пошуку мін). Другі, зануряючись під воду, можуть під’єднати заряд до міни або протаранити її для знешкодження, в залежності від виду дрона.

В усіх вище зазначених випадках пілотування безпілотних надводних апаратів передбачає керування і навігацію цими апаратами або за попередньо запрограмованими маршрутами, або за допомогою дистанційного керування. Частку кожного з типів надводних дронів на ринку наведено на рис. 1.1.

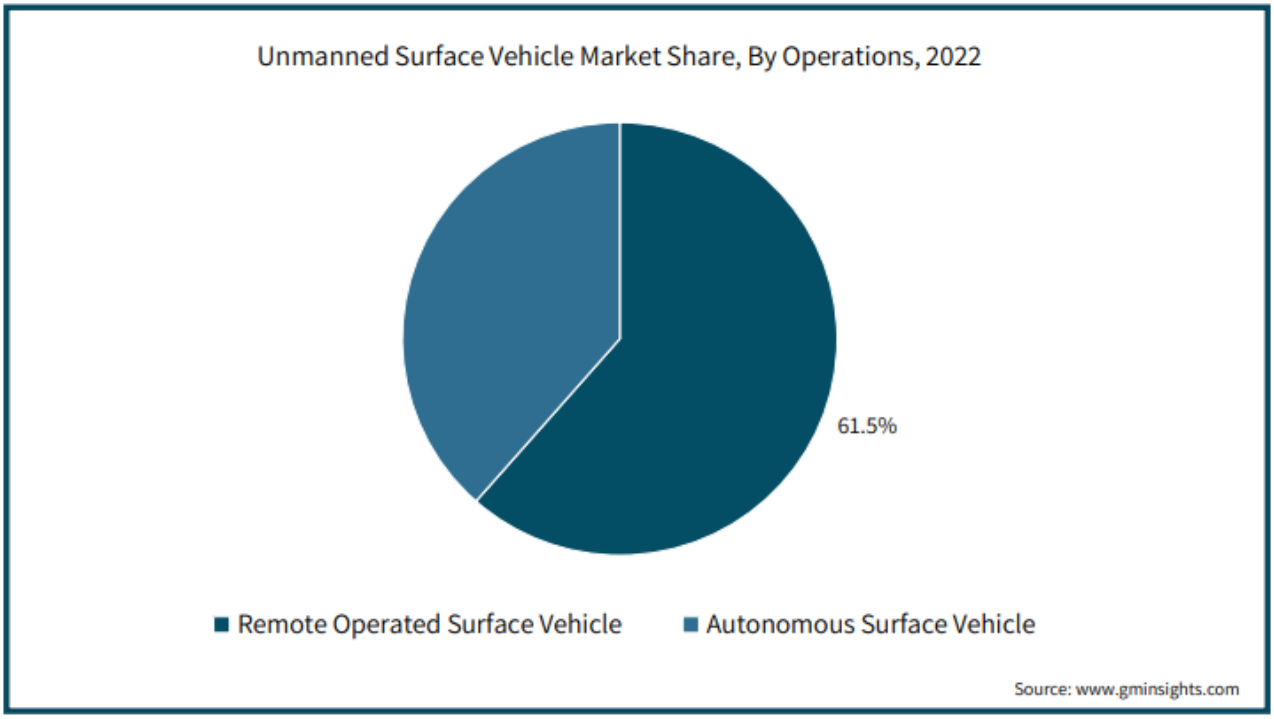

Рисунок 1.1 – Частка ринку безпілотних надводних транспортних засобів на 2022 рік: дистанційно-керовані БНА (61,5%); автономні БНА(38,5%) [1]

Безпілотні апарати оснащені передовими технологіями (датчиками, камерами і системами зв'язку), які дають змогу операторам контролювати їх і керувати ними на відстані. Це забезпечує безпечніші та ефективніші операції, оскільки люди-оператори не наражаються на ризики, пов'язані з перебуванням на борту судна в потенційно небезпечному середовищі.

1.1.2 Аналіз сучасного стану індустрії дронів та виклики, з якими вона зіштовхнулася

Індустрія безпілотних надводних апаратів продовжує активно розвиватися. Це можна побачити на рис. 1.2 [1].

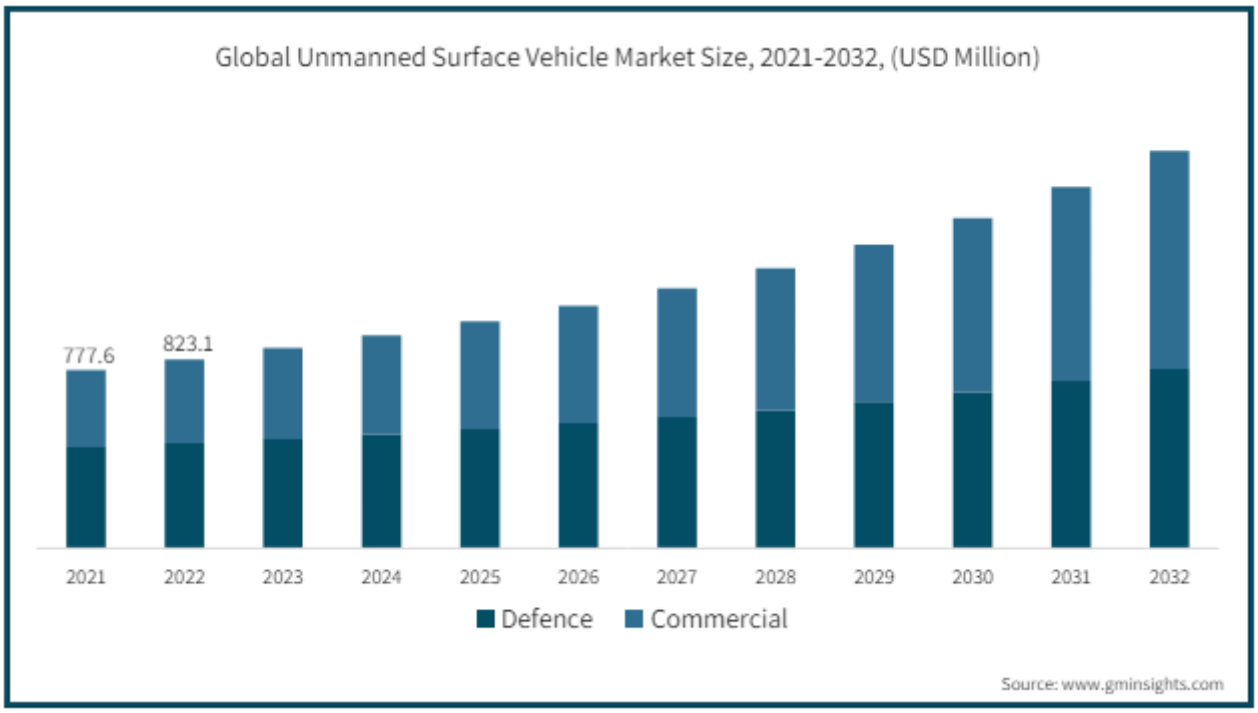

Рисунок 1.2 – Розмір світового ринку безпілотних надводних транспортних засобів, 2021-2032 рр., (млн. дол. США)

Наразі БНА мають широке застосування як у військовій сфері, в якості розвідників, тральників, транспортних кораблів та камікадзе, так і в цивільній сфері у дослідженнях водних ділянок, моніторингу навколишнього середовища та перевезення вантажів. Із кожним роком нові моделі дронів будуть замінювати людини у нових сферах життєдіяльності, яка є небезпечною, важкою для людини, чи вартісною при використанні людського ресурсу.

Однак нинішній стан індустрії безпілотників створює низку проблем, які потребують вирішення. Однією з основних проблем є неефективне, дороге чи ризикове тренування оператора. Кількість дронів можливо буде обмеженою або відсутня у тренувальних цілях, або втрата чи ушкодження дрона за непередбачуваних обставин буде вартісною у відновленні апарата.

Також можна зіштовхнутися з проблемою, коли кількість наявних фахових пілотів, що мають на меті тренувати керуванням дронами, буде значно меншою за потребу в них. Крім того, витрати на навчання з інструктором можуть буди значно більшими як для клієнта, так і для виробника, порівняно зі створенням ігрового

застосунку, який ознайомить зі значною частиною можливостей продукту та варіантами його використання. Щоправда, слід зазначити, що симулятор не може повністю замінити реалії через обмеженість функціоналу та сценаріїв застосунку.

Використання військових БНА також ставить обмеження у розробці практичного тренування керуванню апаратом за умов наближених до реальних, оскільки створення такого полігону є неефективним, через відсутність тренувальних цілей для дронів камікадзе, високий ризик пошкодити тренувальний дрон під час імітації відкриття вогню по ньому та неправдоподібність штучних сценаріїв до реальних. За умов війни створення такого полігону не є можливим через значну обмеженість як технічних ресурсів (дронів), так і людських (операторів), великий ризик влучання ракети або дрона-камікадзе, а також ймовірну диверсію.

Якщо брати до уваги зростаючі загрози морській безпеці, серед яких напади збройних сил чи піратів на кораблі, як оборонні так і цивільні, виробники стали розробляти все більше нових моделей з автономним режимом керування. Це сприяє зростанню світового ринку безпілотних надводних апаратів.

1.1.3 Роль симуляторів у покращенні керування надводними дронами

Симулятори відіграють важливу роль у вдосконалені підготовки операторів до керування безпілотних апаратів. Ігрові симуляції забезпечують безпечне та контрольоване віртуальне середовище, в якому пілоти можуть практикувати свої знання та вдосконалювати вміння керування БНА.

Застосування таких програмних рішень мають низку переваг, однією з них є відтворення реальних сценаріїв. Віртуальні симуляції можуть відтворювати різні умови навколишнього середовища, як-от: швидкість та напрям течії і вітру, штиль або шторм і навіть перешкоди, з якими можна зіткнутися під час реальних операцій. Також за допомогою симуляторів пілоти мають змогу розвинути необхідні навички маневрування апаратом та ухвалення обґрунтованих рішень у складних ситуаціях.

Ігрові симуляції можуть слугувати платформою для підготовки майбутніх фахівців із керування дронів до певних завдань або місій. Для прикладу, за умов

російсько-української війни важливо підготувати фахівців до виконання спецоперацій в умовах, максимально наближених до реальних: знищення того чи іншого корабля дроном камікадзе, розвідка прибережної території, транспортування екіпажу чи провізії, розмінування водних ділянок. Такий вид підготовки значно підвищить ефективність та результативність операторів під час реальних операцій.

Симулятори є економічно ефективним рішенням для навчання керуванню безпілотниками. Порівняно з використанням реальних дронів для тренувань, симулятори усувають ризик пошкодження дорогого обладнання під час тренувань. До того ж, симуляції можна легко налаштувати для різних рівнів підготовки, що дозволяє операторам розвиватися у власному темпі і зосередитися на конкретних сферах, які потребують вдосконалення. Гнучкість і доступність програмних рішень роблять симуляції привабливим варіантом для навчальних програм з керування безпілотниками.

На додачу до розвитку навичок, симулятори ї можуть підвищити загальну безпеку операцій з дронами. Забезпечуючи контрольоване середовище для відпрацювання операторами своїх навичок, симулятори мінімізують ризик нещасних випадків та інцидентів під час реального використання безпілотників. Оператори можуть випробувати різні надзвичайні ситуації у віртуальному середовищі і навчитися реагувати на них належним чином, не наражаючи себе та інших на небезпеку. Особливо цей акцент на безпеці є актуальним в оборонній сфері як у мирний, так і у воєнний час.

1.2 Огляд симуляторів керування дронами

Розглянемо два найпопулярніших симулятори безпілотних надводних апаратів у світі – Gazebo та The USV Simulator. Проаналізуємо переваги та недоліки цих популярних ігрових симуляцій.

Переваги Gazebo:

– налаштування погодних умов: застосунок може налаштовувати вітер, направлення течії, висоту хвиль, рівень туману та фазу дня;

– простий інтерфейс (рис. 1.3 – 1.4);

– реалістична фізика: рух хвиль, направлення течії та вітер, що впливають на рух дрона є наближеними до реальними;

– відкритий програмний код бібліотек: набір бібліотек з відкритим вихідним кодом містить все необхідне, наприклад, загальні математичні типи даних, ведення журналів, керування 3D-мережами та асинхронна передача повідомлень [4];

– можливість розширення функціоналу рушія: більшість бібліотек Gazebo пропонують інтерфейс плагінів, які підтримують використання користувацького коду під час виконання;

– окрема платформа для пошуку розширень: вебзастосунок Gazebo можна використовувати для пошуку нових розширень симулятора;

– немає обмеженості симулятора до певних видів дронів: нові види дронів можна створювати у редакторі або завантажувати з Інтернету.

Недоліки Gazebo:

– відсутність прямої підтримки від розробників: для вирішення проблем, пов’язаних з інструментами розробки, можна звернутися лише до документації або спільноти розробників та однодумців;

– можливі проблеми з інтеграцією до певного апаратного забезпечення: інтеграція симулятора на певні збірки апаратного забезпечення може становити технічні труднощі та вимагати додаткових зусиль з розробки для забезпечення безперебійної сумісності;

– обмеженість документації: документація являє собою посібник із текстовим та графічний матеріалом (зображеннями), що розглядає лише важливі аспекти використання застосунку. Відсутність секції помилок та вирішень проблем;

– складний процес встановлення застосунку: відсутній єдиний інсталятор для тої чи іншої ОС. Усі компоненти необхідно встановлювати та під’єднувати окремо.

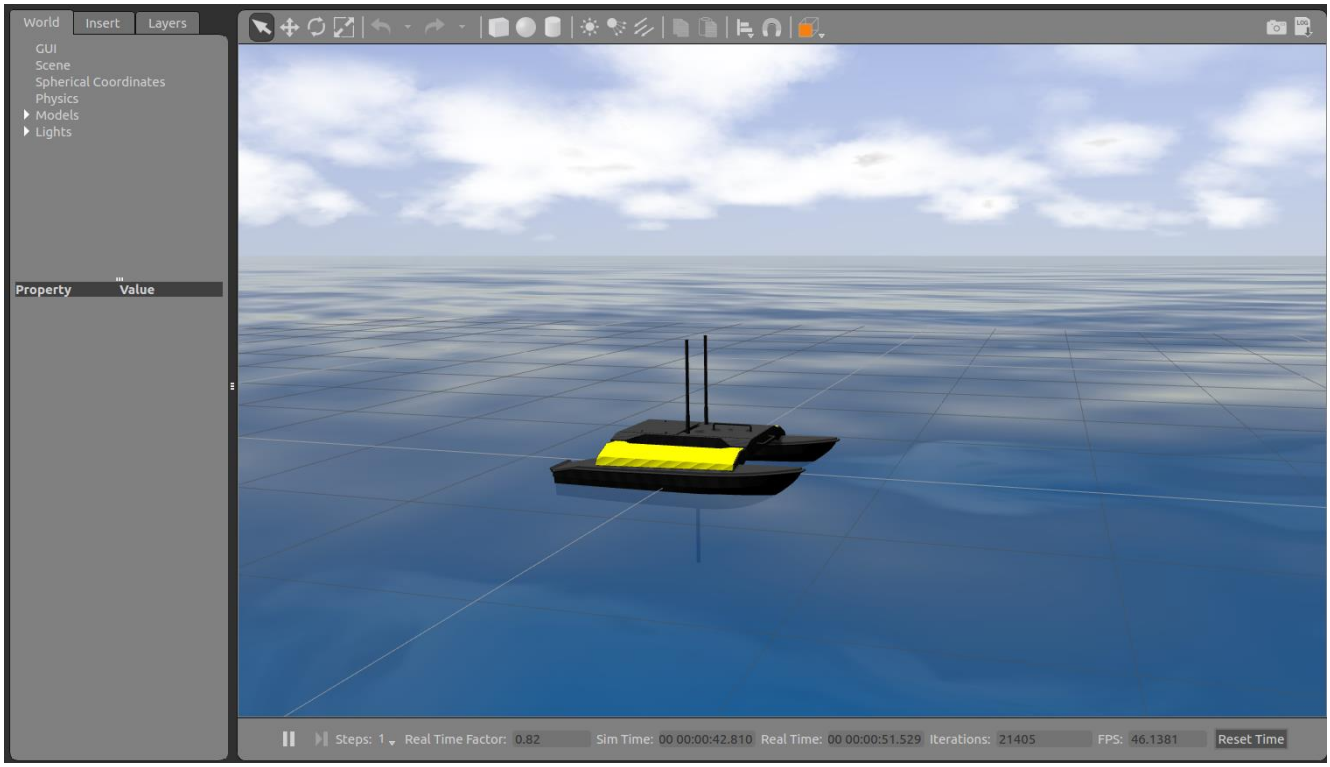

Рисунок 1.3 – Вигляд редактора Gazebo

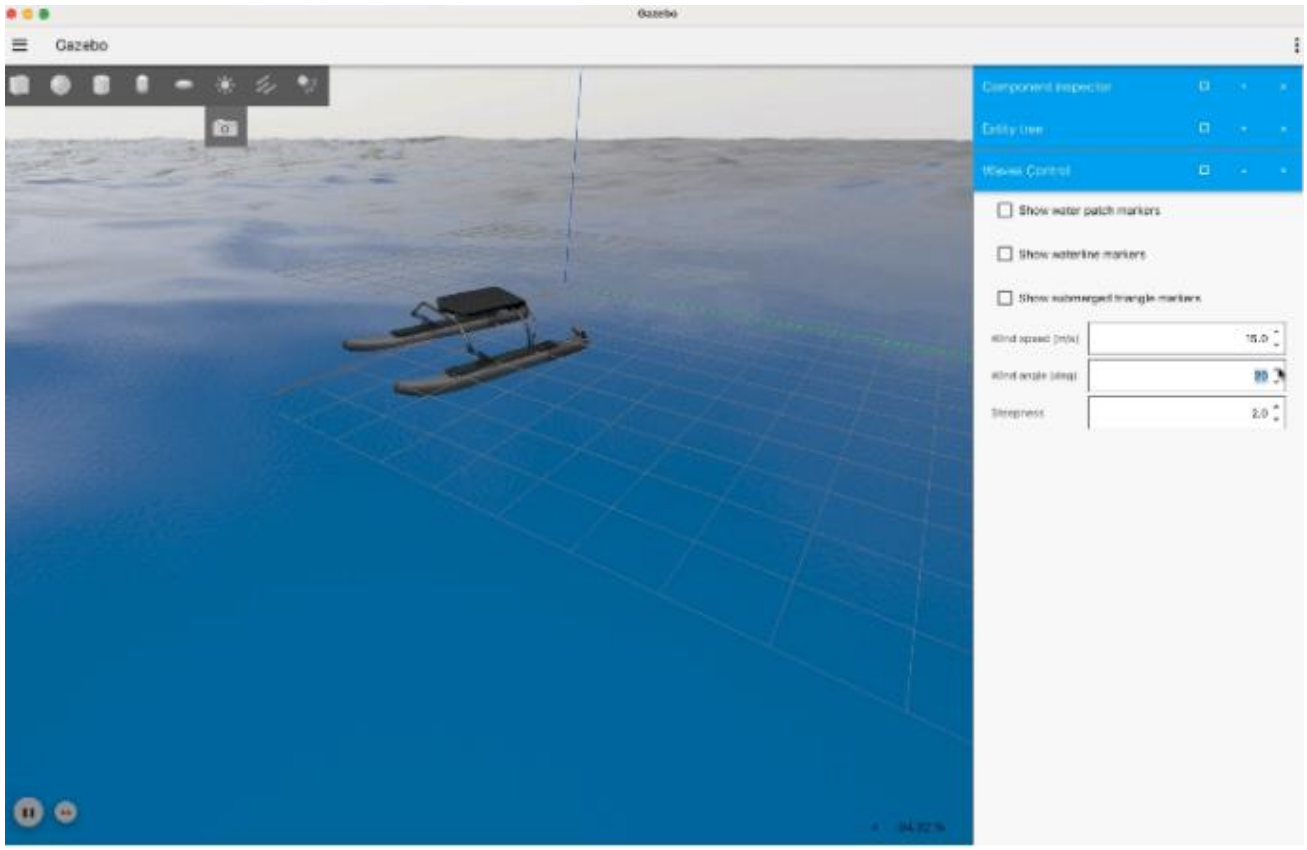

Рисунок 1.4 – Симуляція морських хвиль у Gazebo

Переваги The USV Simulator:

1. застосунок учня:

– відтворює функції та можливості реальної консолі оператора БНА;

– стеження за дроном можна проводити як від першої особи із імітацією 360-градусного огляду та електрооптичних систем відстеження з інфрачервоним зором. (від камери, що розташована на носі або на даху корабля) або від третьої особи (камера, що обертається навколо дрона);

– система моніторингу стану дрона, яка допомагає обслуговувати систему та відновлювати її після збоїв;

– звукова система, що імітує акустичний пристрій дальнього радіусу дії;

2. застосунок інструктора:

– операційна станція інструктора (IOS) дозволяє інструктору планувати та контролювати всі тренувальні вправи;

– створення нових або використання наявних сценаріїв. У симуляторі, можна виконувати такі місії:

- пожежогасіння палаючого судна;
- патрулювання водних ділянок;
- пошук тонучих кораблів та порятунок екіпажу;
- перехоплення суден-цілей;

– запис і відтворення для підтримки детального аналізу після виконання вправ;

– повністю інтегровані та вбудовані пакети для планування уроків, підготовки місій та керування;

– завантаження нових або використання наявних реалістичних моделей середовища. Симулятор має декілька створених образів реальних місць, для прикладу: прибережжя Буенос-Айрес;

– комп'ютерні згенеровані сили (Computer-Generated Forces) створюють імітацію морського трафіку, налаштовуються відповідно до уподобань інструктора;

– аналіз виконаної роботи для перевірки ефективності роботи оператора.

Різниця між застосунками учня та інструктора наведена у рис. 1.5.

3. наявність прямої підтримки від компанії;

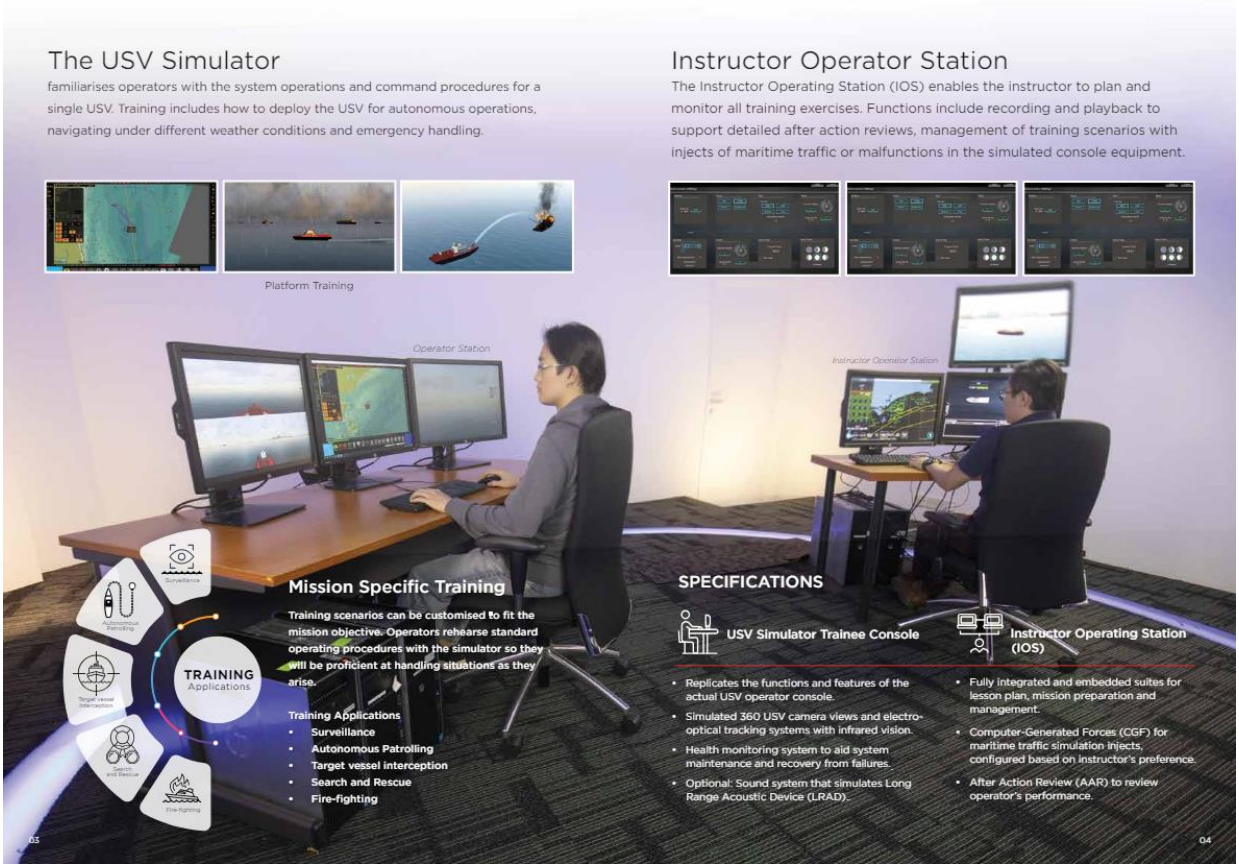

Рисунок 1.5 – Різниця між застосунками оператора та викладача

4. простий процес встановлення застосунку. При попередньому повідомленні компанії про бажання встановити застосунок, співробітники надішлють інсталятори для встановлення застосунків для викладача та учня.

Недоліки The USV Simulator:

1. обмеженість документації. Посібник із використання програмного забезпечення розглядає лише важливі аспекти використання застосунку;

2. обмежений вибір видів дронів. На даний момент часу доступний лише один вид БНА – пожежний рятувальний дрон, який можна побачити на рис. 1.6;

3. відсутність доступу до програмного коду. Неможливість перегляду програмного коду не дає змоги змінити симулятор або додати власні розширення.

Рисунок 1.6 – Кадр з The USV Simulator від компанії ST Engineering

Різниця між цими двома симуляторами:

– *доступ до коду*. Бібліотеки Gazebo знаходяться у публічному архіві. Код The USV Simulator не викладається у публічний доступ;

– *кількість типів безпілотників*. Редактор Gazebo має декілька заготовлених варіантів дронів, також можна створити або додати нові моделі дронів зі своєю унікальною механікою.The USV Simulator у розпорядженні має лише один дрон;

– *вид підготовки операторів*. Gazebo є одним застосунком, в якому створюються та тестуються сценарії. З цього слідує, що симулятор підходить для самостійного ознайомлення із тим чи іншим безпілотним апаратом. The USV Simulator розділений на два застосунки – учнівський та викладацький. Ідея полягає у тому, що викладач створює сценарії, налаштовує середовище тестування та аналізує отримані результати, у свою чергу оператор тестує наявний функціонал дрона;

– *підтримка від розробників*. Оскільки бібліотеки Gazebo є у відкритому доступі, тими, хто можуть дати ради стосовно вирішення конкретної проблеми

будуть розробники зі спільноти. У свою чергу, The USV Simulator має підтримку від компанії, що розробила симулятор;

– *цільова аудиторія*. Gazebo має широку цільову аудиторію, від розробників та дизайнерів роботів до викладачів робототехніки. Починаючи з 2016 року, його почали використовувати у всесвітньовідомих компаніях NASA та Toyota. Кожного року проводяться технічні конкурси зі створення роботів та дронів із використанням симулятора Gazebo [5]. The USV Simulator орієнтований на оборонні й безпекові відомства, а також військово-морські та морські навчальні заклади.

1.3 Аналіз вимог до симулятора

1.3.1 Нефункціональні вимоги

Провівши детальний аналіз симуляторів надводних дронів, визначивши їхні плюси, мінуси та зрозумівши різницю між ними, варто розробити нефункціональні вимоги до власного застосунку.

Основною цільовою аудиторію є Збройні Сили України та Міністерство Внутрішніх Справ, оскільки найбільша частина безпілотних надводних апаратів використовується саме у цих структурах. Через це головною мовою застосунку є українська.

Мінімальним вимогам до апаратного програмного забезпечення мають задовольняти не тільки функціонал ігрового рушія, а й додані до нього програмні розширення (плагіни). Вимоги до ігрового рушія можна знайти на сайті розробників ігрового рушія. Додаткові потреби в апаратних ресурсах можна визначити лише після завершення розробки проєкту.

Операційною системою, на яку орієнтується ігровий застосунок, є Windows 10, оскільки вона є найрозповсюдженішою серед військовослужбовців ЗСУ. Додатково можна створити версію застосунку для Windows 11. Версія симулятора на Windows 11 має працювати без будь-яких змін у поведінці чи продуктивності. Бажано, щоб симулятор було легко адаптувати до інших операційних систем.

Мінімальною частотою кадрів при виконанні ігрового процесу є 30 кадрів на секунду. Для відео така частота є теж задовільною. При нижчій частості гра виглядає неякісною, нецікавою та ресурсозатратною. Зміна частоти кадрів не має перевищувати визначений рівень, при якому погіршення продуктивності застосунку буде помічено гравцем, орієнтовна задовільна різниця у змінні частоти – 5 кадрів у секунду.

Користувач повинен мати можливість налаштовувати параметри симулятора на свій лад, а саме:

- якість відео та візуальних ефектів;
- рівень гучності гри та аудіо ефектів;
- керування безпілотником як клавіатурою та мишею, так і геймпадом;
- чутливість миші.

Окрім цього, гравцеві надано можливість налаштовувати ігровий процес:

- тип та кількість ворогів;
- тип та кількість пасток та перешкод;
- вид середовища;
- погодні умови;
- фаза дня;
- сценарії.

Симулятор має на меті навчати користувачів орієнтуватися у віртуальному просторі застосунку. Кращим рішенням для цього є додання тренувальних рівнів.

Користувальницький інтерфейс розроблений у спрощеному, але подібному до інструментів вимірювання, що є на пульті керування прототипу. Допускається функціональна можливість відімкнути інтерфейс для більш реалістичного вигляду з камери [6]. Сам інтерфейс має доносити суть того чи іншого функціоналу.

Вище зазначені нефункціональні вимоги до програмного застосунку організовані та описані у таблиці 1.2.

Таблиця 1.2 – Нефункціональні вимоги програмного застосунку

Нефункціональні вимоги	**Реалізація у проєкті**
Мінімальні вимоги до апаратного забезпечення	Мінімальні функціональні вимоги не мають перевищувати наступні показники щодо апаратного забезпечення: Процесор – 4 ядра, 2.5 GHz. Відеокарта із підтримкою DirectX11 або DirectX12 та 2 ГБ віртуальної пам'яті. Оперативна пам'ять – 8 гігабайт [7].
Сумісність	Застосунок має бути підтримуваним на Windows 10 або Windows 11. Версія симулятора на Windows 10 має працювати в Windows 11 без будь-яких змін у поведінці чи продуктивності та навпаки. Бажано, щоб симулятор було легко адаптувати до інших операційних систем.
Продуктивність	Симулятор повинен підтримувати стабільну частоту кадрів, 30 кадрів в секунду або вище, на різних апаратних конфігураціях, щоб забезпечити плавний ігровий процес. Зміна частоти кадрів при завантаженні додаткових об'єктів не має перевищувати 5 кадрів в секунду, тому що це буде помітно користувачеві та може вплинути на ігровий процес. Максимальний час завантаження рівня – 120 секунд. Максимальний час завантаження ролика – 30 секунд.
Налаштування застосунка	Симулятор має бути розширюваним, щоб відповідати різним рівням деталізації та складності моделей суден, середовищ та рівнів. Він повинен підтримувати можливість налаштовувати параметри симуляції, такі як погодні умови, час доби та характеристики ігрового сценарію. Також обов'язковими є налаштування застосунку: рівень гучності, якість графіки, розкладка клавіатури для ігрових можливостей, чутливість миші.
Навчання користування застосунком	Необхідно, щоб серед ігрових рівнів були присутні тренувальні місії для ознайомлення користувача із правилами використання застосунком.
Зручність використання	Інтерфейс користувача повинен бути інтуїтивно зрозумілим і зручним, дозволяючи користувачеві легко переміщатися по меню, налаштовувати параметри і контролювати симуляцію. Стиль користувальницького інтерфейс має бути простим та мінімалістичним, водночас доносити суть того чи іншого функціоналу. Ігровий застосунок повинен підтримувати декілька рівнів складності штучного інтелекту супротивників для різних рівнів підготовки пілотів.
Локалізація	Основною мовою застосунку має бути українська, оскільки цільовою аудиторією є ЗСУ та МВС ЗСУ..

1.3.2 Функціональні вимоги

Функціональні вимоги є важливою частиною на етапі аналізу та проєктування програмного забезпечення. Вони визначають певний функціонал або поведінку, які мають бути присутніми у застосунку для задоволення потреб користувача у вирішенні проблем.

Ігровий процес симулятора оператора надводних дронів буде поділяти на декілька етапів (табл. 1.3).

Таблиця 1.3 – Етапи ігрового процесу симулятора

Назва етапу	**Умови переходу на етап**	**Опис етапу**
Головне меню	Запустити застосунок	На цьому етапі оператор має змогу змінити загальні налаштування, що включають якість відео, графіки, аудіо, розкладку клавіатури для ігрового функціоналу, розпочати симуляцію, перейти до тренувальної місії, оглянути короткий опис застосунку та вийти із симулятора.
Тренувальний рівень	У головному меню обрати «Тренування», вибрати один із видів тренувань.	Завантажившись на рівень, пілот прослуховує короткий брифінг, після чого, розпочинається місія. Під час проходження місії, необхідно виконувати вказівки віртуального інструктора та на практичному досвіді вивчати відточувати навички пілотування у різних ситуаціях.
Сценарний рівень	У головному меню обрати «Розпочати симуляцію», обрати бажаний сценарій та параметри.	Завантажившись на рівень, пілот отримує завдання, яке можна виконати різними методами, для прикладу, розвідку прибережної території можна пройти місію не здійнявши тривогу.
Завершення гри	Успішно або провально пройдено тренувальний або сценарний рівень.	Після завершення ігрового рівня, гравець отримує результат у вигляді чи були виконані цілі гри.

З даних таблиці 1.3 слідує, що ігровий процес проходить наступним чином:

– оператор після завантаження застосунку переходить до головного меню. За бажанням, користувач може змінити загальні налаштування для застосунку, якщо вбачає у цьому потребу. Також, присутня можливість короткого ознайомлення із симулятором у текстовому форматі;

– пілот вибирає один із двох варіантів гри «Розпочати симуляцію» або «Тренувальні місії»:

- вибравши «Розпочати симуляцію», гравець має змінити параметри гри, а саме:
  - складність штучного інтелекту;
  - тип місцевості;
  - мінімальну та максимальну кількість берегової охорони (патрулів та вогневих позицій);

- наявність та кількість наземної, морської та повітряної техніки, що займається охороною території;
- наявність, кількість і дальність засобів радіоелектронної боротьби;
- кількість мін та бонів (загородження з плавучих плотів або колод, що захищають вхід у гавань порту від ворожих кораблів);

- вибравши «Тренувальні місії», гравець вибирає один із запропонованих рівнів, а саме:
  - вступна місія (ознайомлення не тільки з керуванням дроном, а й з методами їх використання);
  - стеження;
  - полювання.

Коли було вибрано «Розпочати симуляцію» та введено усі бажані налаштування гри, оператор завантажується на ігровий рівень. Після завершення цього процесу, гравець має слідкувати за вказівками та виконати назначену місію, паралельно засвоюючи та практикуючи навички керування дроном.

Після завершення місії та перегляду результату, гравець повертається до головного меню або розпочинає місію заново.

Якщо гравець натиснув на «Тренувальні місії», на вибір будуть запропоновані місії, що мають такі задачі:

– розвідка;
– знищення цілі;
– транспортування провізії або амуніції.

Далі гравець проходить коротке ознайомлення з місією та має виконати завдання. Методи виконання залежать від типу місії та виду дрона.

Після завершення місії та перегляду результату, гравець повертається до головного меню або розпочинає місію заново.

Оскільки проєкт розроблено у вигляді ігрового симулятора, для його реалізації вибрано такі інструментальні засоби:

– C++ в якості мови програмування;

– Unreal Engine як ігровий рушій;

– Blueprint як систему візуального скриптування;

– Unreal Test як фреймворк для тестування програмного забезпечення.

C++ широко застосовується у розробці застосунків, для яких основним параметром є оптимізація та ефективність виконання задач, серед таких програм – ігрові застосунки та симуляції.

Мова програмування C++ має такі переваги:

– висока оптимізація та ефективність рендерингу зображення. Оскільки симулятори часто включають складні обчислення та рендеринг якісного зображення у реальному часі, використання такої мови як C++, дозволяє досягти оптимальної продуктивності, гарантуючи, що симулятор зможе впоратися з обчисленнями необхідними для реалістичних симуляцій;

– наявність великої кількості фреймворків та бібліотек. C++ має велику спільноту та глибоку історію розвитку. Від випуску мови до сьогодення, було створено багато розширень, які допоможуть зменшити час розробки та необхідність винаходити раніше знайдені рішення проблем;

– кросплатформеність. C++ це портативна мова, код якої може бути скомпільований на різних платформах із мінімальними змінами.

У якості ігрового рушія вибрано Unreal Engine, тому що він має такі функціональні можливості:

– реалістична графіка. Unreal Engine відомий своїми передовими графічними можливостями, зокрема, високою точністю рендерингу, динамічним освітленням та реалістичній симуляції взаємодії об’єктів із фізичними явищами. Це буде корисно при побудові ігрових рівнів;

– вбудовані інструменти. Unreal Engine надає широкий спектр інтегрованих у рушій інструментів для створеннях складних симуляцій, у тому числі, систему візуального скриптування Blueprints, редактори анімацій, матеріалів, візуальних та аудіо ефектів. Ці програмні засоби пришвидшать процес розробки та спростять подальші доповнення до функціоналу ігрового застосунку. Також у рушії присутній

плагін (розширення) Water, що має готову симуляцію води, що допоможе у створенні рівнів;

– кросплатформна розробка. Unreal Engine підтримує розгортання на різних платформах, включаючи Windows, macOS, Linux, Xbox OS, Playstation OS, Nintendo Switch OS та мобільні пристрої, що робить її ідеальним вибором для розробки кросплатформних ігор та застосунків. Використовуючи C++ для розробки Unreal Engine, можна написати код один раз і розгортати його на різних платформах без втрати продуктивності та функціональності.

Як раніше було зазначено, Blueprint – це інтегрована система візуального скриптування у рушій Unreal Engine. Вона має такі переваги:

– візуальне представлення. Система Blueprints надає візуальний інтерфейс для написання сценаріїв, який дозволяє розробникам створювати та маніпулювати логікою, використовуючи систему на основі нодів (вузлів). Таке візуальне представлення полегшує розуміння і налагодження складних взаємодій в симуляторі, оскільки розробники можуть бачити потік логіки і даних лише поглянувши на вузли, їх зв’язки та характеристики;

– просте створення інтерфейсу. Редактор Unreal Engine надає можливість перетягувати віджети інтерфейсу безпосередньо на область перегляду, що спрощує додавання кнопок, повзунків, текстових полів та інших графічних елементів до інтерфейсу. Потім розробники можуть налаштовувати зовнішній вигляд і поведінку цих віджетів за допомогою Blueprints без написання коду;

– легке налагодження. Система Blueprints пропонує вбудовані інструменти налагодження, які дозволяють розробникам перевіряти змінні, встановлювати точки зупинки та проглядати логіку, щоб виявити та вирішити проблеми. Це полегшує усунення проблем під час тестування і гарантує, що симулятор поводиться так, як очікується, за різних умов;

– швидке створення прототипів. Система Blueprints дозволяє швидко створювати прототипи ігрових механік та взаємодій без необхідності писати великі обсяги коду. Особливо це може бути корисно для швидкого тестування різних аспектів симулятора та додання нових програмних можливостей;

– не потребує компіляції. На відміну від традиційного C++ коду, Blueprints не потребують компіляції, а це означає, що зміни можна вносити та тестувати в режимі реального часу в редакторі Unreal Engine. Цей швидкий цикл ітерацій прискорює процес тестування і дозволяє більш ефективно експериментувати з різними сценаріями та конфігураціями;

– інтеграція з C++: Хоча Blueprints в основному використовуються для написання високорівневого функціоналу (наприклад, створення взаємодії між графічним інтерфейсом та грою), їх також можна використовувати разом із C++ кодом для створення гібридних рішень.

Unreal Test – це надійний фреймворк для тестування, інтегрований у Unreal Engine, який має кілька особливостей для тестування ігрових застосунків:

– модульне та інтеграційне тестування. Unreal Test підтримує як модульне, так і інтеграційне тестування, яке надає можливість тестувати як окремі компоненти, так і взаємодію між різними під системами симулятора. Цей підхід до тестування допомагає переконатися, що кожна частина симулятора функціонує правильно як окремо, так і в поєднанні з іншими компонентами;

– тестування продуктивності. Unreal Test дозволяє проводити тестування продуктивності ігрового застосунку, оцінюючи його частоту кадрів та використання ресурсів за різних умовах. Це допомагає розробникам переконатися, що симулятор працює стабільно на окремих платформах та пристроях із різним апаратним забезпеченням;

– інтеграція з Unreal Engine. Unreal Test спеціально розроблений для тестування проєктів, створених за допомогою Unreal Engine, завдяки чому він легко інтегрується в середовище розробки. Ця тісна взаємодія спрощує процес тестування і дозволяє писати тести безпосередньо в редакторі Unreal Engine, використовуючи знайомі інструменти та робочі процеси;

– кросплатформне тестування. Unreal Test підтримує тестування на різних платформах, включаючи Windows, macOS, Linux, консолі та мобільні пристрої. Ця можливість значно зменшує кількість ресурсів, які необхідно витратити на

адаптацію тестів, також дає змогу дізнатися про надійність застосунку на тій іншій платформі чи пристрої.

Провівши детальний аналіз симуляторів операторів дронів, вимог до програмного застосунку, наявних інструментів розробки, можна дійти наступних висновків:

– функціонал симулятора повинен бути відповідним до завдання та мети роботи, тобто високоефективним, наближеним до реальності для спрощення підготовки, покрашення ефективності набуття нових та відточення наявних навичок пілотування дронів;

– симулятор має відповідати раніше зазначеним нефункціональним (табл. 1.2) та функціональним вимогам (табл. 1.3);

– у ході розробці, використовувати C++ у якості основного інструменту для створення елементів ігрового процесу, систему Blueprints як допоміжну.

Для того, щоб досягнути бажаного результату, необхідно спроєктувати та розробити такий функціонал симулятора:

– головне меню із усіма готовими опціями;

– меню створення ігрового рівня за вибраними параметрами налаштування та сценаріями;

– ігровий рівень, у якому ігрові об’єкти відповідають раніше зазначеним налаштуваннями та сценаріями;

– тренувальні рівні, із детальним аудіо та візуальним поясненнями можливостей симулятора;

– завершення рівня із загальною оцінкою проходження місії.

У наступному розділі буде детально розглянуто проєктування та розробку необхідних алгоритмів, структур та моделей, які необхідно реалізувати програмно та протестувати.

2 ПРОЄКТУВАННЯ СИМУЛЯТОРА

2.1 Дизайн ігрового процесу

2.1.1 Дизайн ігрових можливостей дрона

Кожна функція, розроблена для безпілотників, має на меті відтворити реальні характеристики апаратів та розширити можливості оператора ефективно діяти в різних сценаріях.

Краще за все, функціонал надводного дрона покаже функціональна модель. Така модель відображає функціонал усього або частини програмного забезпечення у вигляді ієрархічної моделі. Приклад такої моделі наведено на рис. 2.1 [8].

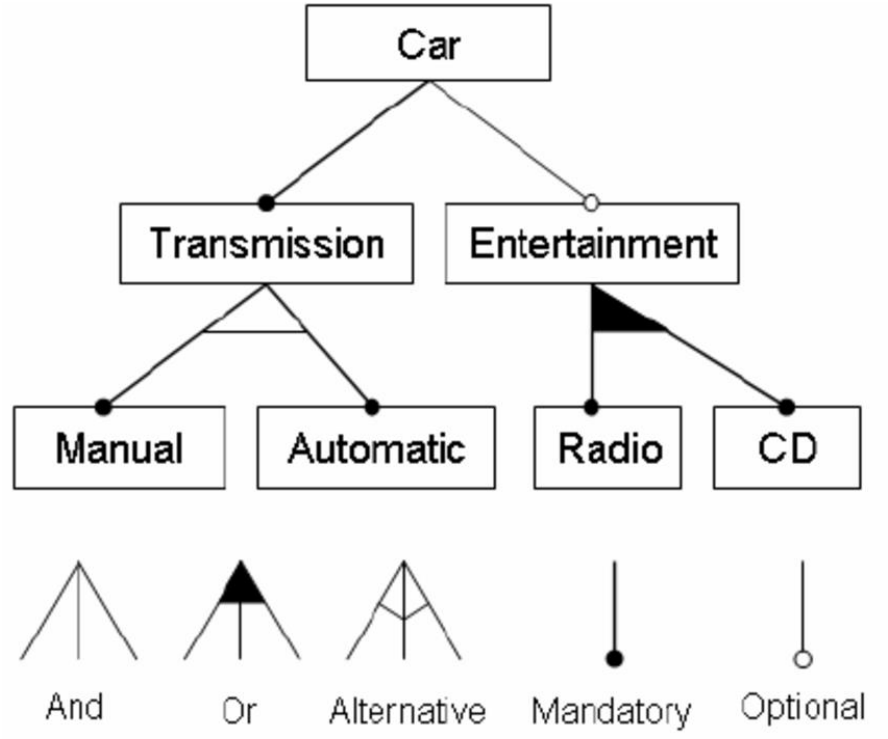

Рисунок 2.1 – Приклад функціональної моделі

Функціонал БНА зображено на рис. 2.2.

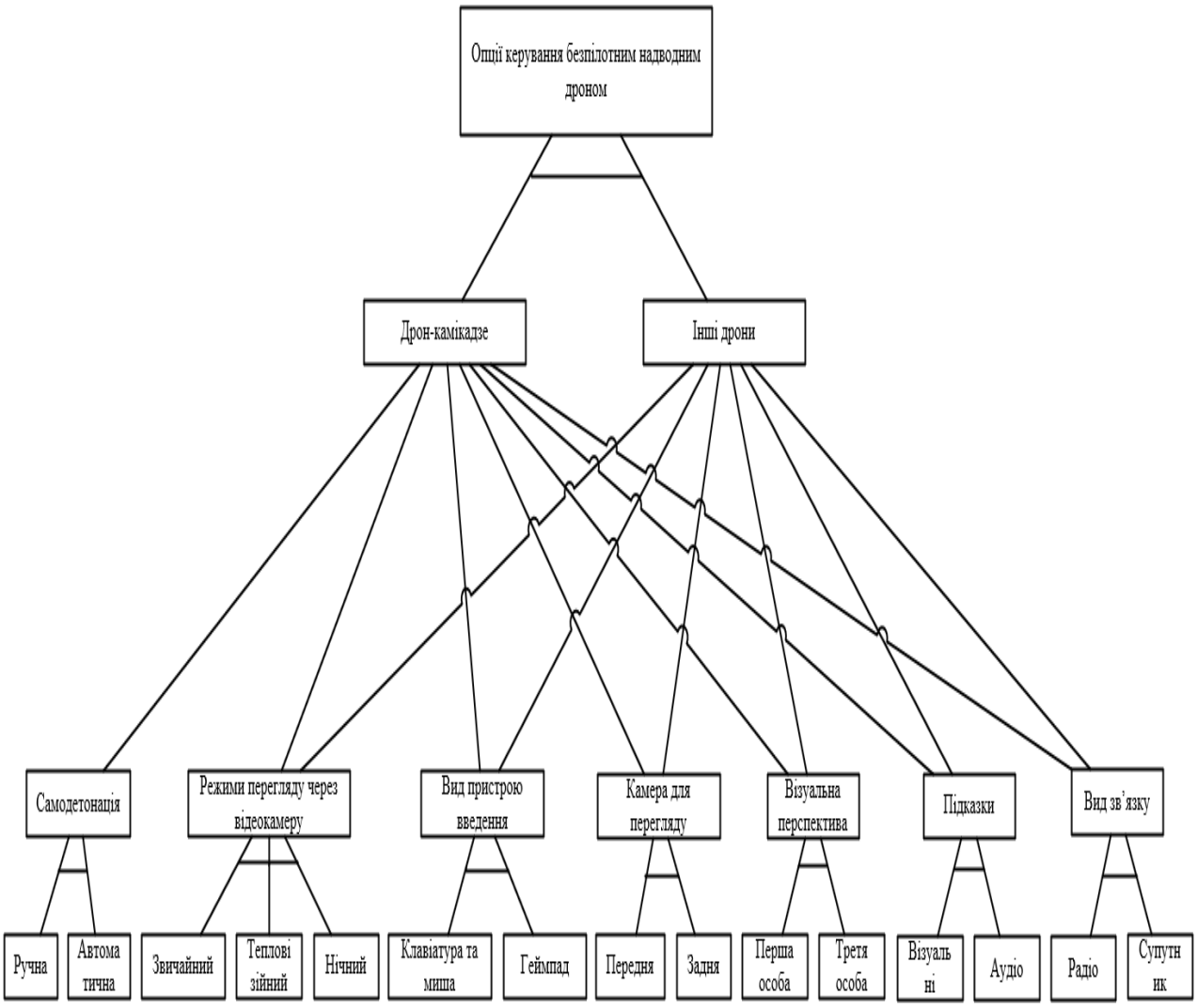

Рисунок 2.2 – Функціональна модель ігрових можливостей надводного дрона

Розглянемо кожну функцію надводного апарата. Симулятор підтримує декілька режимів перегляду через відеокамеру для покращення обізнаності щодо ситуації на ігровому рівні та оперативного прийняття необхідних рішень, а саме:

– *звичайний* – усталений режим перегляду, що забезпечує чітку видимість за нормальних умов освітлення;

– *тепловізійне бачення* дозволяє операторам виявляти теплові сигнали, що корисно для нічних операцій або крізь густий туман чи димові завіси;

– *нічне бачення* покращує видимість в умовах низької освітленості, що має вирішальне значення для нічних місій.

На додачу до різних режимів перегляду дрони будуть обладнані як передніми, так і задніми камерами, що сприятиме всебічному візуальному охопленню.

Оператор може перемикатися між цими апаратами, щоб отримати різні перспективи оточення.

Окрім цього, симулятор буде пропонувати перспективи перегляду як від першої, так і від третьої особи. Вид від першої особи забезпечує пряму видимість із камери дрона, що забезпечує спостереження із реального виду. Вид від третьої особи дає ширшу перспективу, корисну під час тренувань для покращення обізнаності у середовищі.

Також, у дрон інтегровані можливості для масштабування, що дозволяє операторам наближати та віддаляти зображення для детального огляду.

Керування дроном буде здійснюватися як за допомогою геймпаду, так і клавіатури та миші.

Дрон можна буде не тільки приводити у рух, але й зупиняти.

БНА підтримують декілька видів зв’язку, це зроблено для того, щоб у разі пошкодження антени, заглушення чи погіршення сигналу одного виду з’єднання, можна буде змінити на інший режим передачі інформації дрону. Оператори матимуть змогу перемикатися між радіо та супутниковим з'єднанням, щоб підтримувати контроль за різних умов навколишнього середовища та доступності сигналу. Радіозв’язок витрачає менше палива, щоправда його можна заглушити системою РЕБ. З іншого боку, супутниковий зв’язок неможливо перервати РЕБ, проте він витрачає значно більше палива.

Для тренувальних цілей у застосунку будуть присутні як візуальні, так і аудіо підказки, що допомагатимуть новачкові орієнтуватися у віртуальному просторі. За бажанням ці функції можна як ввімкнути, так і вимкнути.

Візуальною підказкою виступатиме значок цілі – невелика мітка з відстанню між ціллю та гравцем.

Аудіо підказками є:

- настанови віртуального помічника оператора;
- попередження про низький рівень палива;
- попередження про важкі ушкодження конкретної частини дрона.

Додатково, якщо гравець керує дроном-камікадзе, у нього будуть такі функціональні можливості:

– автоматичний самопідрив. Дрон автоматично вибухне при зіткненні із кораблем;

– ручний самопідрив. Оператор може детонувати дрон натиснувши на спеціальну кнопку або клавішу. Це дає змогу контролювати процес самоліквідації дрона під тактичні потреби;

– самознищення у разі фатальних пошкоджень. Якщо критично важливому компонентові дрона задіяно непоправної шкоди, дрон автоматично підірветься, не даючи ворогам можливість захопити БНА.

Щоб гравець мав змогу змінити розкладку клавіш, кнопок чи джойстиків, переглянути місію у деталях, призупинити чи завершити, процес симуляції, буде присутня функція виклику меню паузи.

Ігровий рушій Unreal Engine підтримує можливість контролювати персонажа як за допомогою миші та клавіатури, так і геймпадом. Нижче описано розкладку ігрових можливостей кожного із вище зазначених пристроїв введення інформації.

Розкладка клавіатури та миші

Гравець використовує мишу для огляду віртуального середовища, а клавіатуру для переміщення по ньому.

Список клавіш та кнопок із призначеними до них ігровими:

– W – переміщення уперед;

– S – переміщення назад;

– A – переміщення вліво;

– D – переміщення вправо;

– R – звичайний режим перегляду;

– T – тепловізійний режим перегляду;

– Y – нічний режим перегляду;

– 1 – вид від передньої камери;

– 2 – вид від задньої камери;

– 3 – вид перегляду від третього обличчя;
– ліва кнопка миші – зближення зображення;
– права кнопка миші – віддалення зображення;
– U – обрати з’єднання із дроном по радіо мережі;
– I – обрати з’єднання із дроном по супутниковому зв’язку;
– B – ввімкнути/вимкнути аудіо підказки;
– N – ввімкнути/вимкнути значок цілі;
– P – ручна детонація дрона-камікадзе;
– Esc (Escape) – ввімкнути/вимкнути меню паузи.

Розкладка геймпада

Геймпад, також відомий як контролер, є пристроєм введення інформації від користувача для взаємодії з віртуальним середовищем в ігровому застосунку. Зазвичай він складається з кнопок, тригерів, джойстиків, а іноді й клавіш керування (D-pad).

Оскільки ігровий застосунок підтримується на операційних системах Windows 10 та Windows 11, найкращим варіантом є геймпад від компанії Xbox. Серед багатьох варіантів такого пристрою було вибрано Xbox 360 Gamepad, оскільки він є найпопулярнішою моделлю для використання в ігрових цілях. На рис. 2.3 показана розкладка моделі геймпада Xbox 360 [9].

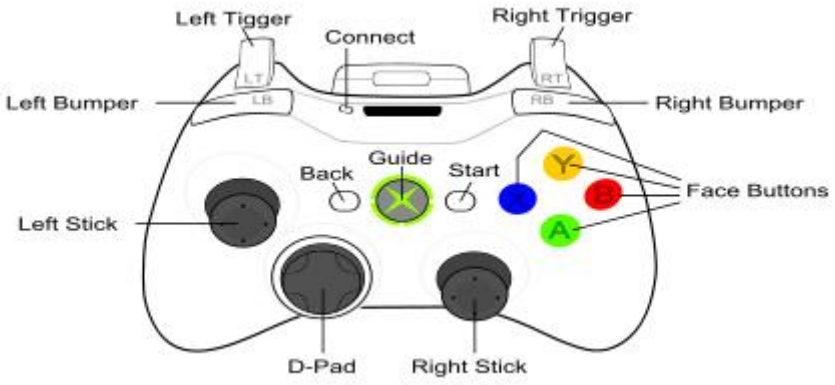

Рисунок 2.3 – Розкладка геймпаду Xbox 360

Керування геймпадом виглядає відміним від клавіатури та миші. Список стандартної розкладки для геймпаду:

– лівий джойстик та D-Pad – використовується для вибору напряму руху дрона;

– правий джойстик – огляд віртуального середовища;

– лівий тригер – газ (приведення дрон у рух);

– правий тригер – тормоз (зупинка дрона);

– лівий бампер – зближення зображення;

– правий бампер – віддалення зображення;

– кнопка Start – меню паузи;

– кнопка Y – зміна режиму перегляду через відеокамеру (перше натискання – звичайний вид, друге – тепловізійний вид, третє – нічний вид, четверте – до звичайного виду);

– кнопка B – зміна камери (перше натискання – передня камера, друге – задня);

– кнопка A – зміна виду від особи (перше натискання – вид від першої особи, друге – вид від третьої особи);

– кнопка X – зміна режиму з’єднання між дроном (перше натискання – по бездротовій мережі, друге – через супутник);

– кнопка Back – ручне знищення дрона камікадзе.

Візуальна та аудіо підказки, можна ввімкнути або вимкнути у меню паузи.

2.1.2 Дизайн ігрових можливостей перешкод та пасток

Перешкоди та пастки є невід’ємними компонентами симуляції, оскільки вони змінюють складність віртуального середовища під реальні сценарії, надаючи можливість операторові дрона застосовувати різні підходи та тактики пілотування. У цьому підрозділі описані типи перешкод та пасток, які вбудовані у симулятор, а також механізми їх взаємодії із безпілотником.

Є такі перешкоди:

– перешкоди навколишнього середовища – природні об'єкти (погодні умови), які можуть впливати на навігацію та роботу безпілотника;

– штучні перешкоди – міни, бони, ворожі судна, засоби РЕР та РЕБ.

Слід зазначити, що усі згадані перешкоди можна налаштовувати при створенні сценарного рівня.

Розглянемо детальніше кожну з категорій. Розпочнімо з перешкод навколишнього середовища.

Параметрами погодних умов є:

– фаза дня (ранок, день, вечір або ніч) – впливає на видимість через камеру дрона, необхідність вмикати тепловізійне чи нічне бачення, дальність видимості ворогів та використання ними прожекторів;

– атмосферні опади (відсутність опадів, дощ, буря) – впливає на видимість через камеру дрона та дальність видимості ворогів;

– видимість (відсутність туману, легкий туман, середній туман, сильний туман) – впливає на видимість через камеру дрона та дальність видимості ворогів;

– висота хвиль (штиль, низька, середня, велика) – впливає на маневрування та швидкість надводного апарата.

Далі розглянемо можливі штучні пастки.

У роботі буде використовуватися лише один вид мін – контактні, що спричиняють вибух при контакті із нею. Ці міні плавають на поверхні води. Було вибрано даний вид, оскільки, через велику швидкість, малий відсоток зануреного корпусу НА та його невеличкі розміри, лише такі міни здатні нанести шкоду дронові.

Бони або бонові загородження – плавучі загородження, які слугують для обмеження поширення будь-чого поверхнею води. Бонові загородження забезпечують ефективну локалізацію можливих зон розливу та переміщення нафти в акваторії портів, водосховищах, затонах, річках, у відкритому морі, а також використовуються для огородження нафтоналивних суден у процесі вантажних операцій, тим самим, забезпечуючи надійний захист від забруднення водних

акваторій [10]. У симуляції ці перешкоди будуть статичними та у разі зіткнення у них, дрон вибухне.

Усі військові кораблі матимуть стаціонарні кулемети, що наноситимуть значної шкоди дронові, у порівняні зі стрілецькою зброєю. Також, на борту можуть бути присутніми прожектори, які будуть вмикатися під час патрулювання та бойової тривоги, коли НА було помічено.

Засоби РЕР працюють так: якщо дрон входить у зону радіоелектронної розвідки, на екрані з’являється невеличкий значок у вигляді радару, подібного до того, який зображений на рис. 2.4. У оператора є обмежена кількість часу для того, щоб вийти із зони, поки не піднялася тривога [11].

Рисунок. 2.4 – Попередження про входження у зону системи РЕР

Засоби РЕБ має на меті погіршувати зв’язок між дроном та оператором із подальшим заглушенням сигналу. Якщо уявити зону дії РЕБ у вигляді кола, то воно матиме дві частини – внутрішня (мертва зона) та зовнішня (зона РЕБ), це можна побачити на рис. 2.5. Поки гравець знаходиться поза межами мертвої зони, він має змогу покинути зону РЕБ та продовжити місію, в іншому випадку, зв’язок із апаратом переривається та завдання є проваленим. Щоб уникнути ефекту системи РЕБ, гравцеві необхідно змінити вид зв’язку на супутниковий, щоправда ціною постійної значної витрати палива.

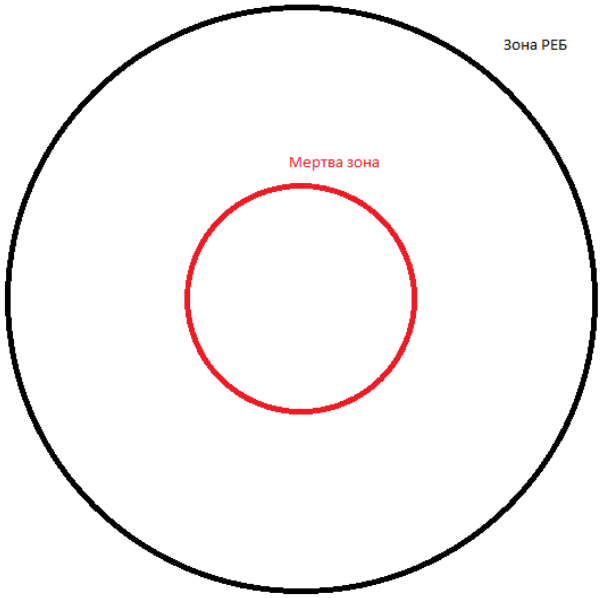

Рисунок. 2.5 – Спрощене відображення роботи системи РЕБ

2.1.3 Дизайн ігрових можливостей ворогів

Ворогами в симуляторі є люди чи військова техніка, яка керується штучним інтелектом. Їхньою головною метою є завадити гравцеві виконати місію. Вороги можуть як перебувати неподалік в цілі, так і патрулювати деякі частини ігрового рівня.

Вороги за типом пересування поділяються на:

– наземні – патрульні групи, вогневі позиції та броньовану машину піхоти;
– надводні – кораблі класів фрегат, корвет та катер;
– повітряні – гелікоптери.

Патрульні групи складаються з 2-5 людей, що патрулюють прибережну територію та мають малу дальність видимості й ураження. Вони озброєнні автоматами, що завдають певну шкоду дронові.

Вогневими позиціями може бути бункер або укриття з мішків із піском, посередині якого стоїть крупнокаліберний кулемет, що завдає середньої шкоди дронові. Дальність видимості є невеличкою, а ураження – середньою.

Броньовані машини піхоти в симуляторі виступають у ролі пересувних вогневих позицій, що мають на озброєнні крупнокаліберний кулемет, що завдає середньої шкоди дронові. Дальність видимості є невеличкою, а ураження – середньою. Швидкість пересування висока.

Кораблі класу фрегат виконують свої бойові місії у відкритому морі, тому бойові чергування такого роду кораблі будуть нести лише на рівнях із великим водним простором. Знаходячись у порту фрегати не нестимуть загрози. Оснащені протикорабельними кулеметами, що завдають велику шкоду дронові. Мають малу швидкість пересування, велику дальність видимості та ураження. Також, такий корабель може нести на собі вертоліт.

Кораблі класу корвет є кораблями ближньої морської зони, призначені для дозорної та конвойної служби, протичовної та протиповітряної оборони військово-морських баз та пунктів базування [12]. Вони будуть зустрічатися у місіях із прибережним типом місцевості, де будуть нести патрулювання, або портовим, де перебуватимуть у спокої. Оснащені протикорабельними кулеметами, що завдають велику шкоду дронові. Мають середню швидкість пересування, велику дальність видимості та ураження.

Кораблі класу катер є малими патрульними суднами для берегової або портової охорони. Зустрічаються катери лише на рівнях прибережного та портового типу місцевості, де нестимуть патрулювання. Оснащені крупнокаліберними кулеметами, що завдають середню шкоду дронові. Мають високу швидкість пересування, середню дальність видимості та ураження.

Вертоліт є повітряним противником, який може базуватися як на березі, так і на фрегаті. Зустрічається в усіх типах місцевості. Мають високу швидкість пересування, високу дальність видимості та середню дальність ураження. Оснащений крупнокаліберним кулеметом, що завдає середню шкоду дронові.

На початку місії, вороги не знають про наявність БНА у їхній місцевості, тому вони перебувають у стані спокою. Якщо на невеличку мить, гравець промайнув у полі зору РЕР або одного із ворогів, тоді противник переходить у стан зацікавленості. Через деякий проміжок часу, якщо гравець досі перебуває у полі

видимості противника, гравець викриє себе, тоді настає стан тривоги та усі патрульні сили перейдуть у бойовий режим. В інакшому випадку, вороги повернуться до стану спокою. Під час бойового режиму, всі будуть обізнані про місцезнаходження дрона, тому ті супротивники, що патрулюють, перейдуть якнайближче до дрона, щоб у нього поцілити. Якщо дрон покине переслідувачів, тоді стан ворогів зміниться на «на сторожі». Під час цього стану, сили противника патрулюватимуть територію, де останнього разу перебував дрон. Під час цього стану, якщо дрон не з’являвся у полі зору довгий проміжок часу, настає стан спокою, інакше – тривога.

Перед створенням рівня за сценарієм, у параметрах можна змінити рівень штучного інтелекту, який змінює швидкість реакції та точність ворогів. Рівні складності поділяються на:

- новачок;
- досвідчений;
- ветеран;
- легенда.

Також у параметрах можна змінити наявність того чи іншого виду противників, а також їхню чисельність. Максимальна кількість ворогів залежить від виду місцевості. Щоправда деякі із ворогів можуть перебувати лише у деяких видах місцевості, наприклад, наземні сили лише на прибережній та портовій місцевості.

2.2 Дизайн ігрових рівнів

2.2.1 Дизайн сценарних рівнів

Розробка сценарних рівнів у симуляторі спрямована на створення різноманітних, реалістичних середовищ, які спонукають операторів ефективно застосовувати свої навички в різних обставинах.

У функціональних умовах було коротко описано процедуру створення сценарних рівнів. Розглянемо більш докладно редактор рівнів та алгоритми, які допомагають у створенні таких рівнів.

Перейшовши до редактора рівнів, гравець має вибрати необхідні параметри в окремому розділі. Лише після визначення усіх необхідних опцій можна розпочати гру.

Налаштування у розділі «Параметри»:

– опції середовища. Необхідно обрати фазу дня, атмосферні опади, видимість та висоту хвиль:

- фаза дня:
  - ранок. Видимість ворогів дорівнює 75% від первинного значення рівня видимості ворогів. Видимість через камеру хороша;
  - день. Видимість ворогів дорівнює 100% від первинного значення рівня видимості. Видимість через камеру хороша;
  - вечір. Видимість ворогів дорівнює 50% від первинного значення рівня видимості. Видимість через камеру середня. Рекомендовано використати тепловізійне або нічне бачення в деяких моментах;
  - ніч. Видимість ворогів дорівнює 25% від первинного значення рівня видимості. Видимість через камеру низька. Є необхідність ввімкнути тепловізійне або нічне бачення для кращої орієнтації. Вороги використовують прожектори;
- атмосферні опади;
  - відсутність опадів. Не погіршує видимість як ворогів, так і оператора через камеру дрона;
  - дощ. Злегка погіршує вид через камеру дрона та рівень видимості у ворогів на 5% від значення, отриманого від вибору фази дня;
  - буря. Значно погіршує вид через камеру дрона та рівень видимості у ворогів на 10% від значення, отриманого від вибору фази дня;
- видимість:
  - відсутність туману. Не погіршує видимість як ворогів, так і оператора через камеру дрона;

- легкий туман. Злегка погіршує вид через камеру дрона та рівень видимості у ворогів на 2% від значення, отриманого від вибору виду атмосферних опадів;
- середній туман. Значно погіршує вид через камеру дрона та рівень видимості у ворогів на 4% від значення, отриманого від вибору виду атмосферних опадів;
- сильний туман. Сильно погіршує вид через камеру дрона та рівень видимості у ворогів на 6% від значення, отриманого від вибору виду атмосферних опадів;

- висота хвиль:
  - штиль. Не впливає на рух БНА;
  - низька. Задає незначних змін руху БНА;
  - середня. Задає помітних змін руху БНА. Зниження швидкості руху при зіткненні із зустрічною хвилею;
  - висока. Задає великих змін руху БНА. Зниження швидкості руху та направлення при зіткненні із зустрічною хвилею. Можливе нанесення ушкодження апаратові;
- тип місцевості:
  - морське прибережжя. Являє собою велику берегову лінію, яка омивається морем. Підходить для будь-яких сценаріїв;
  - суцільне море. Рівень повністю заповнений водою. Таку місцевість можна обрати лише із сценарієм «знищення цілі»;
  - порт. Місцевість являє собою великий порт та його околиці. Підходить лише для сценаріїв «знищення цілі» та «розвідка»;

– опції ворогів. Необхідно обрати складність штучного інтелекту та чисельність кожного виду ворогів:

- складність ШІ:
  - новачок. Рівень точності стрільби ворогів є низьким. Перебуваючи у стані зацікавленості противники переходять у режим тривоги після безперервного огляду гравця протягом 7

секунд із використанням PEP та 10 секунд шляхом огляду дрона. Якщо дрон зник із поля зору:

- час зміни із «зацікавленості» на «спокій» – 5 секунд;
- тривалість тривоги – 10 секунд;
- час зміни із «на сторожі» на «спокій» – 20 секунд;

▪ досвідчений. Рівень точності стрільби ворогів є середнім. Перебуваючи у стані зацікавленості противники переходять у режим тривоги після безперервного огляду гравця протягом 5 секунд із використанням PEP та 7 секунд шляхом огляду дрона. Якщо дрон зник із поля зору:

- час зміни із «зацікавленості» на «спокій» – 15 секунд;
- тривалість тривоги – 25 секунд;
- час зміни із «на сторожі» на «спокій» – 40 секунд;

▪ ветеран. Рівень точності стрільби ворогів є високим. Перебуваючи у стані зацікавленості противники переходять у режим тривоги після безперервного огляду гравця протягом 3 секунд із використанням PEP та 5 секунд шляхом огляду дрона.

- час зміни із «зацікавленості» на «спокій» – 30 секунд;
- тривалість тривоги – 40 секунд;
- час зміни із «на сторожі» на «спокій» – 60 секунд;

▪ легенда. Рівень точності стрільби ворогів є високим. Перебуваючи у стані зацікавленості противники переходять у режим тривоги після безперервного огляду гравця протягом 3 секунд із використанням PEP та 3 секунди шляхом огляду дрона.

- час зміни із «зацікавленості» на «спокій» – 60 секунд;
- тривалість тривоги – 60 секунд;
- час зміни із «на сторожі» на «спокій» – 75 секунд;

○ чисельність ворогів:

- патрульні групи. Від 0 до 10 на прибережжі та від 0 до 5 у портовій місцевості. Неможливо змінити кількість на суцільному морі;
- вогневі позиції. Від 0 до 2 на прибережжі та від 0 до 3 у портовій місцевості. Неможливо змінити кількість на суцільному морі;
- бронемашини піхоти. Від 0 до 5 на прибережжі та від 0 до 3 у портовій місцевості. Неможливо змінити кількість на суцільному морі;
- фрегати. Від 0 до 2 на прибережжі, у портовій місцевості та у суцільному морі;
- корвети. Від 0 до 3 на прибережжі та у портовій місцевості. Неможливо змінити кількість на суцільному морі;
- катери. Від 0 до 5 на прибережжі та у портовій місцевості. Неможливо змінити кількість на суцільному морі;
- гелікоптери. Від 0 до 2 на прибережжі, у портовій місцевості та у суцільному морі (в залежності від кількості фрегатів);

– опції перешкод та пасток. Необхідно обрати кількість тих чи інших видів перешкод та пасток:

- бони. Від 0 до 2 канатів у портовій місцевості. Неможливо змінити кількість на суцільному морі та прибережжі;
- міни. Від 0 до 10 на прибережжі та у портовій місцевості, а також від 0 до 15 у суцільному морі;
- РЕБ. Від 0 до 3 на прибережжі та від 0 до 2 у портовій місцевості. Неможливо змінити кількість на суцільному морі;
- РЕР. Від 0 до 4 на прибережжі та від 0 до 3 у портовій місцевості. Неможливо змінити кількість на суцільному морі;

– опції сценарію. Необхідно обрати один із трьох доступних сценаріїв. Слід зазначити, що в залежності від місії буде вибрано відповідний для виконання завдання дрон:

- розвідка. Для цієї місії знадобиться розвідувальний дрон. Основною ціллю цієї місії є пошук військових об’єктів на вибраній місцевості, їх маркування та подальша втеча. Маркування відбувається автоматично, через 3 секунди, після наведення камери на ціль. Об’єктами у цьому сценарії можуть виступати:
  - військовий штаб;
  - станція зв’язку;
  - нафтове сховище;
  - казарми;
  - військовий корабель;
- знищення цілі. Кращий у цій справі є дрон-камікадзе. Основною ціллю цієї місії є знищення військового корабля шляхом ручної або автоматичної детонації дрона. Кораблі, які необхідно знищити, матимуть клас ескадрильний міноносець, фрегат або корвет;
- транспортування провізії або амуніції. Транспортний дрон відіграє ключову роль у цьому завданні.

Щоб спростити процес створення ігрових рівнів, буде вручну створено три рівні, які мають три різні типи місцевості.

Далі виникає питання: як використати ці рівні для різних сценаріїв, створити ту кількість вибраних ігрових об’єктів, які були раніше зазначені, при тому, зберігши їхнє коректне місцезнаходження на рівні? Оптимальними вирішенням цієї проблеми є створення окремого класа спавнера. Його можна розмістити на будь-якому місті ігрового рівня та обрати об’єкт, який гравець бажає розмістити. Спавнери, що генерують певний об’єкт, зберігаються у відповідному масиві. Далі, ми ітеруємо крізь масив невключно до індекса заданого значення. Проходячи крізь кожен елемент масиву, ми генеруємо потрібний нам об’єкт. Програмний код цього рішення описано у додатку А.

Після вибору всіх необхідних пунктів оператор завантажується на рівень та бере контроль над своїм дроном. В подальшому гравець має виконати цілі сценарію.

При провальному чи успішному завершенні ігрового сценарію на екран виводиться загальна оцінка «Місію провалено» або «Місія виконана успішно».

2.2.2 Дизайн тренувальних рівнів

Тренувальні місії є заздалегідь заготовленими ігровими рівнями, що мають на меті ознайомити нових пілотів з експлуатаційними можливостями безпілотника та середовищем симуляції, а також для вдосконалення навичок більш досвідчених операторів. Загалом, у симуляторі присутні три навчальних сценарії:

- **Вступ**. На цьому тренувальному рівні гравець навчиться:
  - керуванню дроном;
  - орієнтацією на місцевості;
  - орієнтації у інтерфейсі статусу дрона;
  - використанню тепловізійного бачення при наявності туману та нічного бачення у нічну пору доби;
  - зміні камер наявних на дроні;
  - зміні виду від особи;
- **Стеження**. На цьому тренувальному рівні гравець навчиться:
  - ввімкненню/вимкненню аудіо та візуальних підказок для порівняння реального та доповненого інтерфейсу;
  - змінні виду зв’язку із радіо на супутниковий, при входженні у зону дії системи РЕБ;
  - як помічати цілі;
  - як втекти, якщо здійнялася тривога;
- **Полювання**. На цьому тренувальному рівні гравець навчиться:
  - маневруванню під час шквального вогню від того чи іншого виду противників;
  - використанню автоматичного режиму детонації;
  - використанню ручного режиму детонації.

Після завершення кожного рівня виводиться результат гри.

2.3 Дизайн користувальницького інтерфейсу

2.3.1 Головне меню

Із головного меню розпочинається ознайомлення користувача з ігровим застосунком. У першому розділі коротко описано ігровий процес симулятора. Далі буде розглянуто детально функціонал кожного з етапів ігрового процесу. Перш за все, необхідно розглянути програмні можливості головного меню та за яких умов їх можна буде використати. Кращим рішенням цієї проблеми є діаграми діяльності, вони показують потік від однієї діяльності до іншої в межах системи або процесу.

Діаграми діяльності опцій головного меню наведено на рис. 2.6. Оскільки рисунок має великий розмір, його було розділено на окремі частини для кожної з великих опцій, таких як налаштування відео (рис. 2.7), аудіо, (рис. 2.8), керування (рис. 2.9).

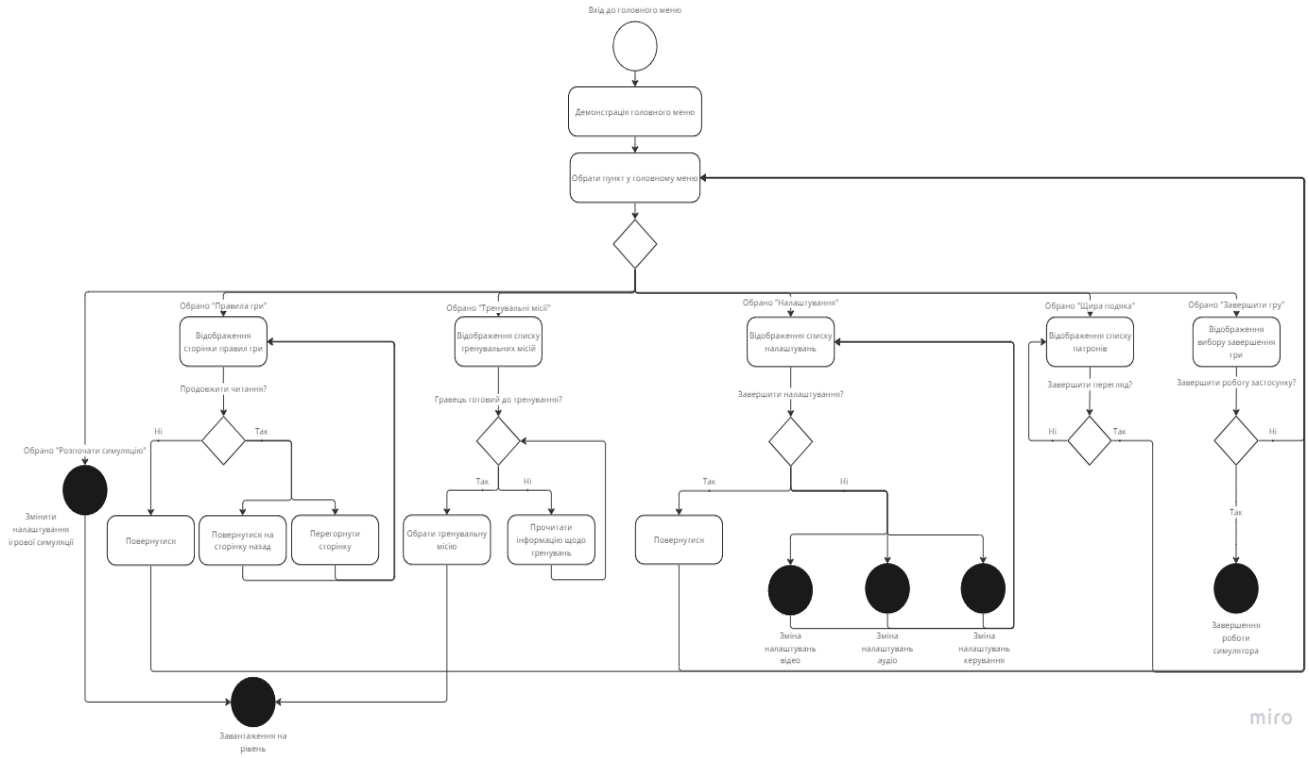

Рисунок 2.6 – Діаграма діяльності опцій головного меню

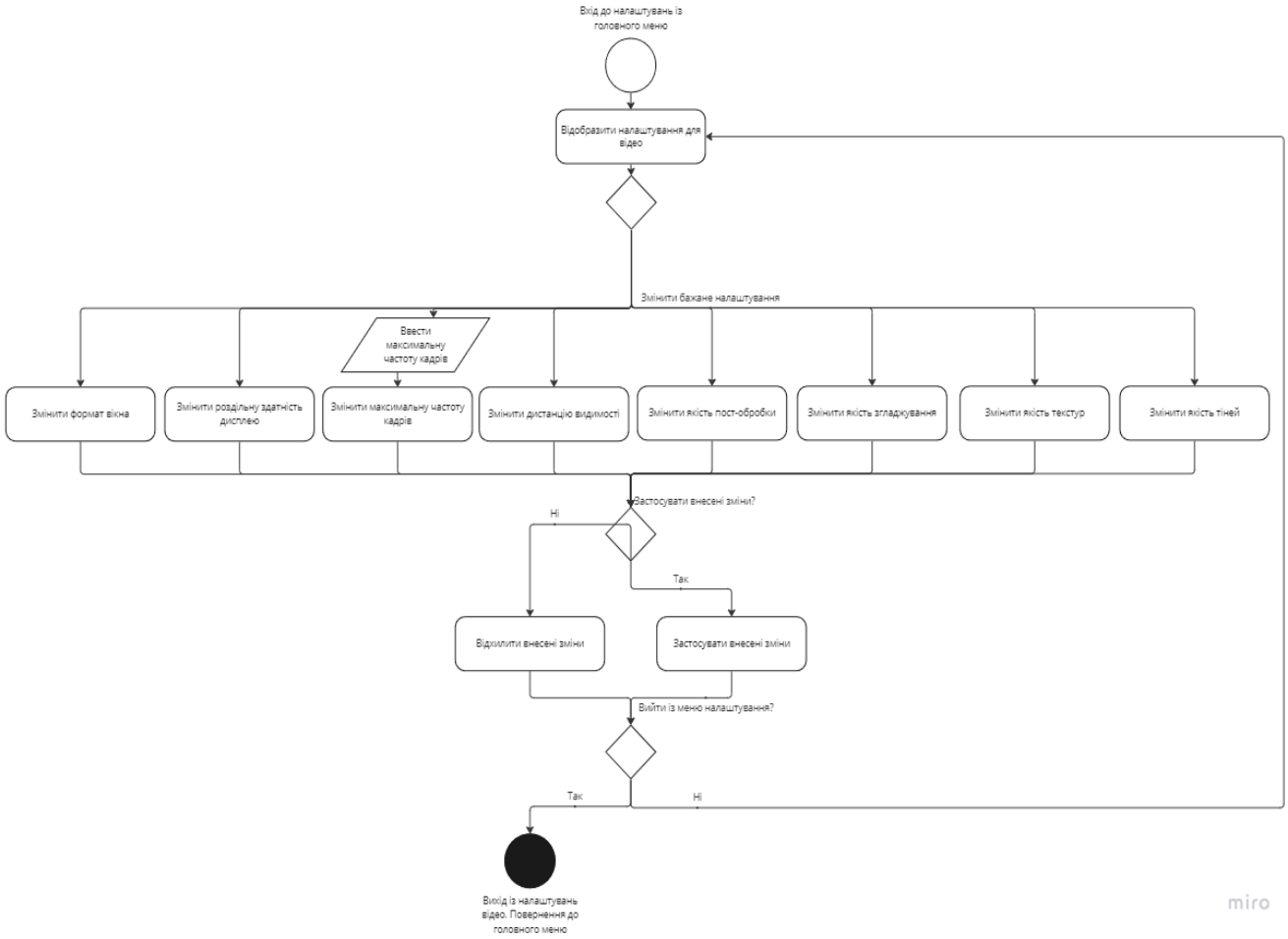

Рисунок 2.7 – Діаграма діяльності опцій налаштування відео

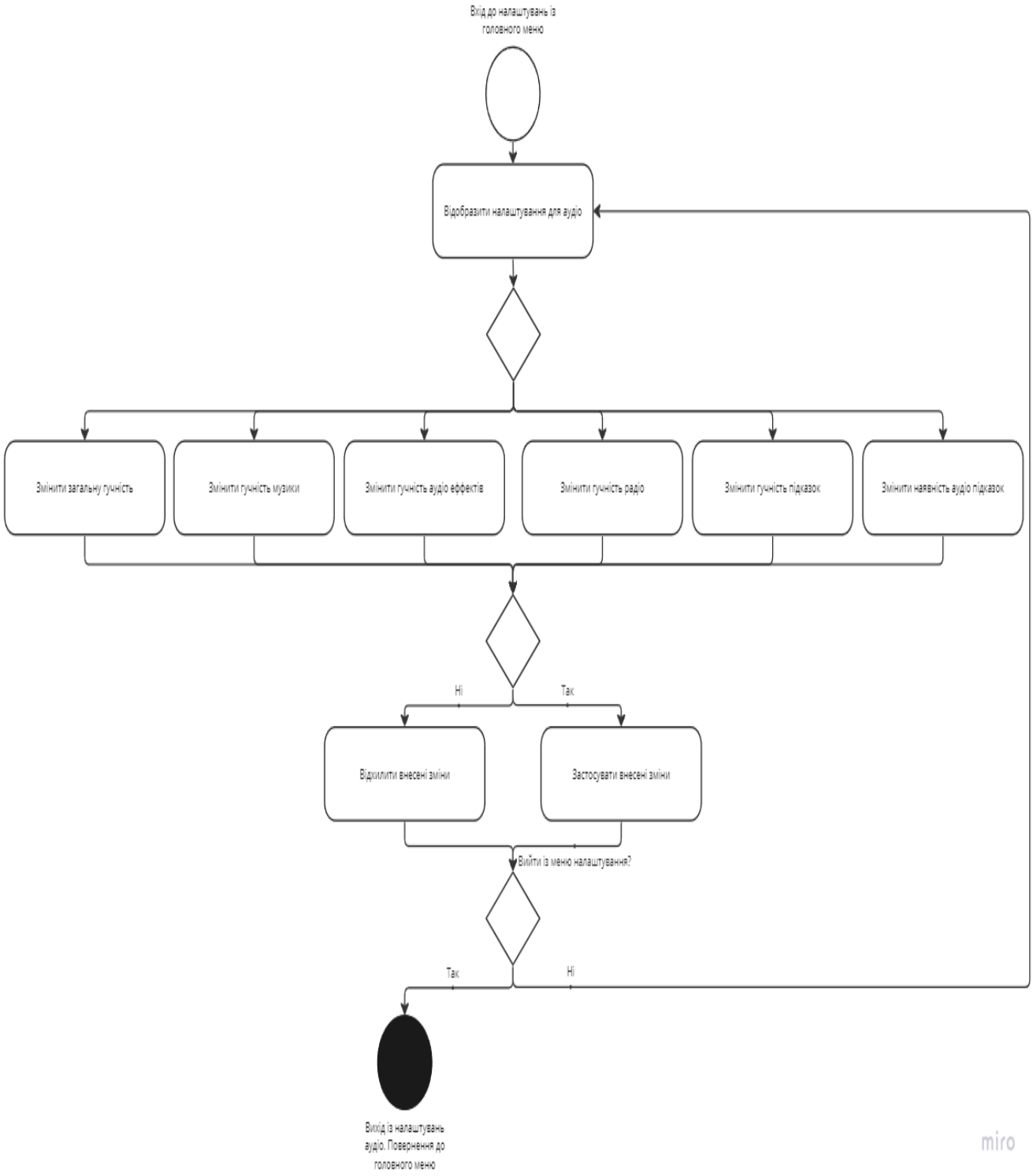

Рисунок 2.8 – Діаграма діяльності опцій налаштування аудіо

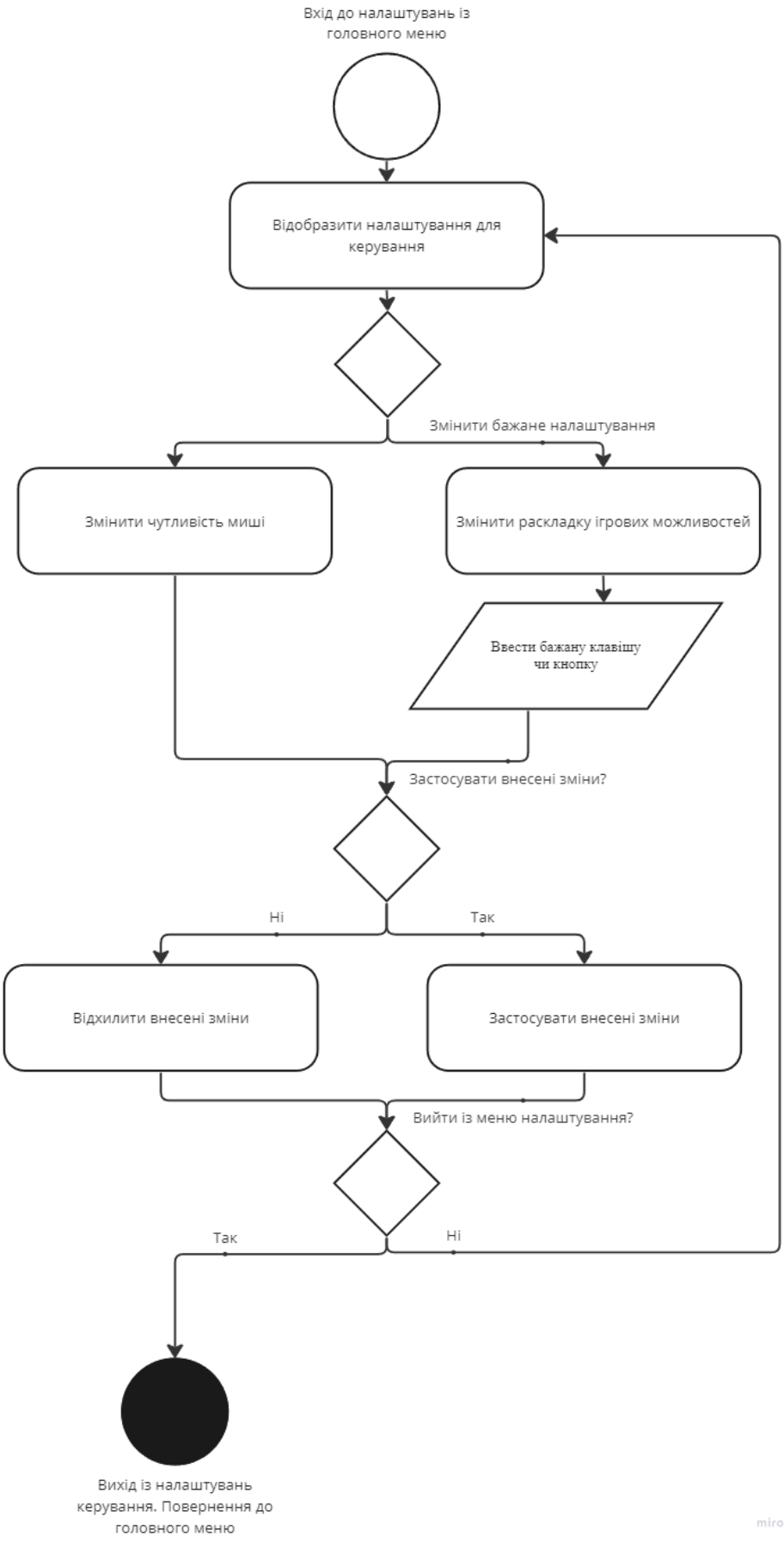

Рисунок 2.9 – Діаграма діяльності опцій налаштування керування

Якщо було вибрано «Розпочати симуляцію», переходимо до наступного етапу ігрового процесу – проходження ігрової симуляції за вибраним сценарієм та параметрами. Як і з головним меню, буде розглянуто програмний потік цього режиму гри та представимо як налаштування параметрів (рис. 2.10), так й ігровий процес самого режиму (рис. 2.11).

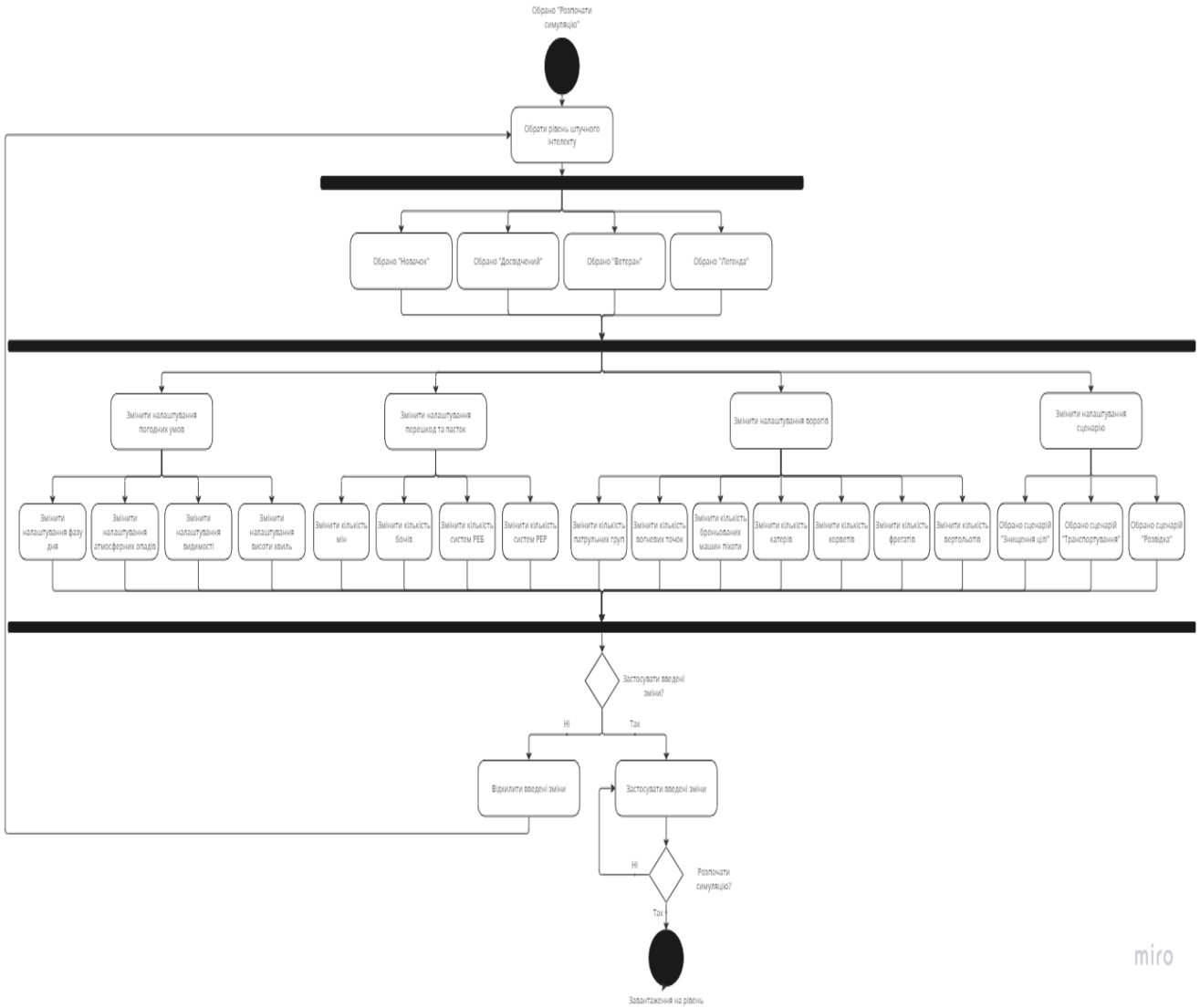

Рисунок 2.10 – Діаграма діяльності налаштування параметрів режиму гри за сценарієм та параметрами

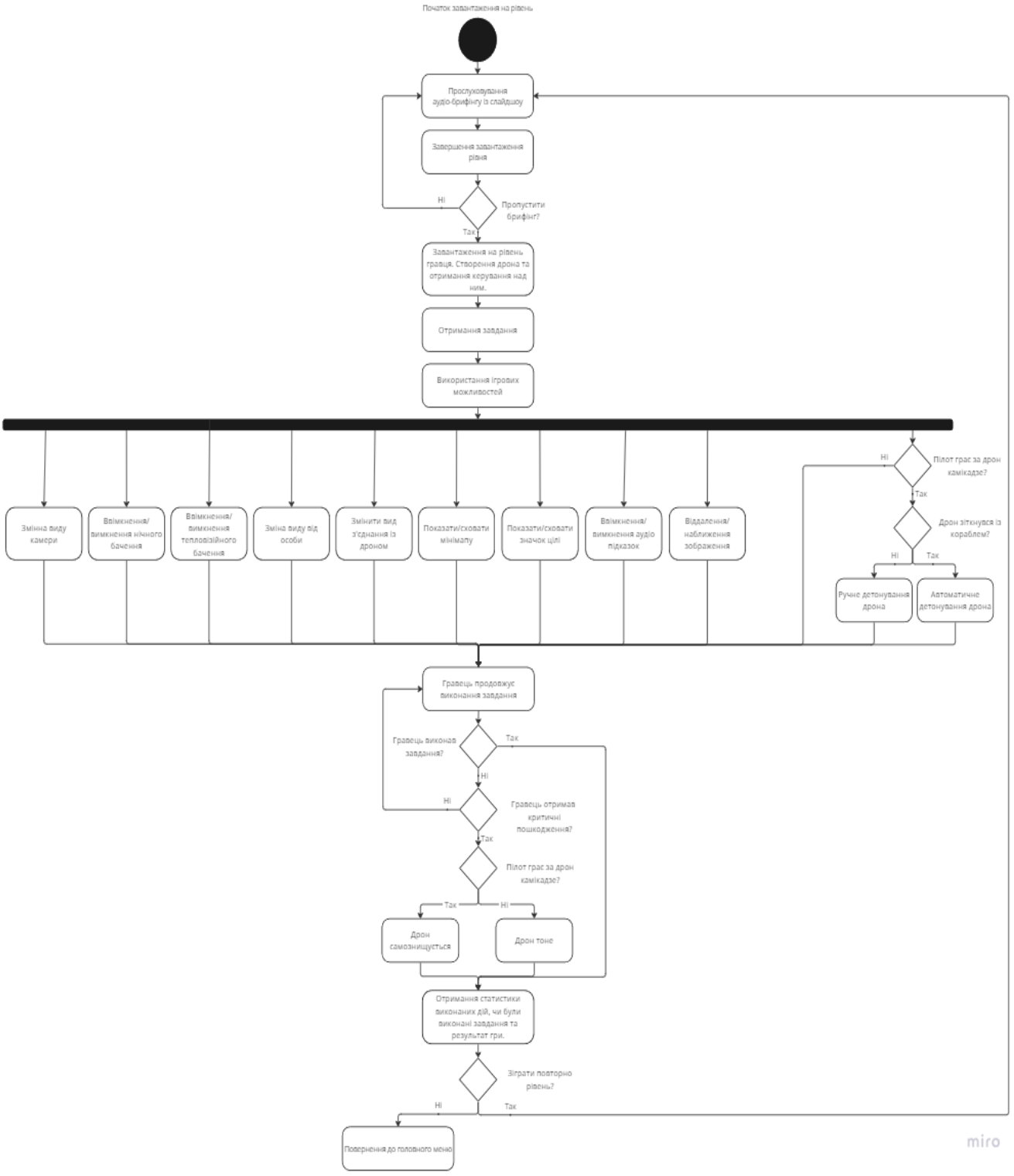

Рисунок 2.11 – Діаграма діяльності режиму гри за сценарієм та параметрами

У разі вибору «Тренувальні місії» програмний потік буде відміним від «Розпочати симуляцію», оскільки користувач має змогу лише обрати серед запропонованих тестовий рівень перед початком гри. Нижче, у діаграмі діяльності наведено програмний потік на тренувальних рівнях (рис. 2.12).

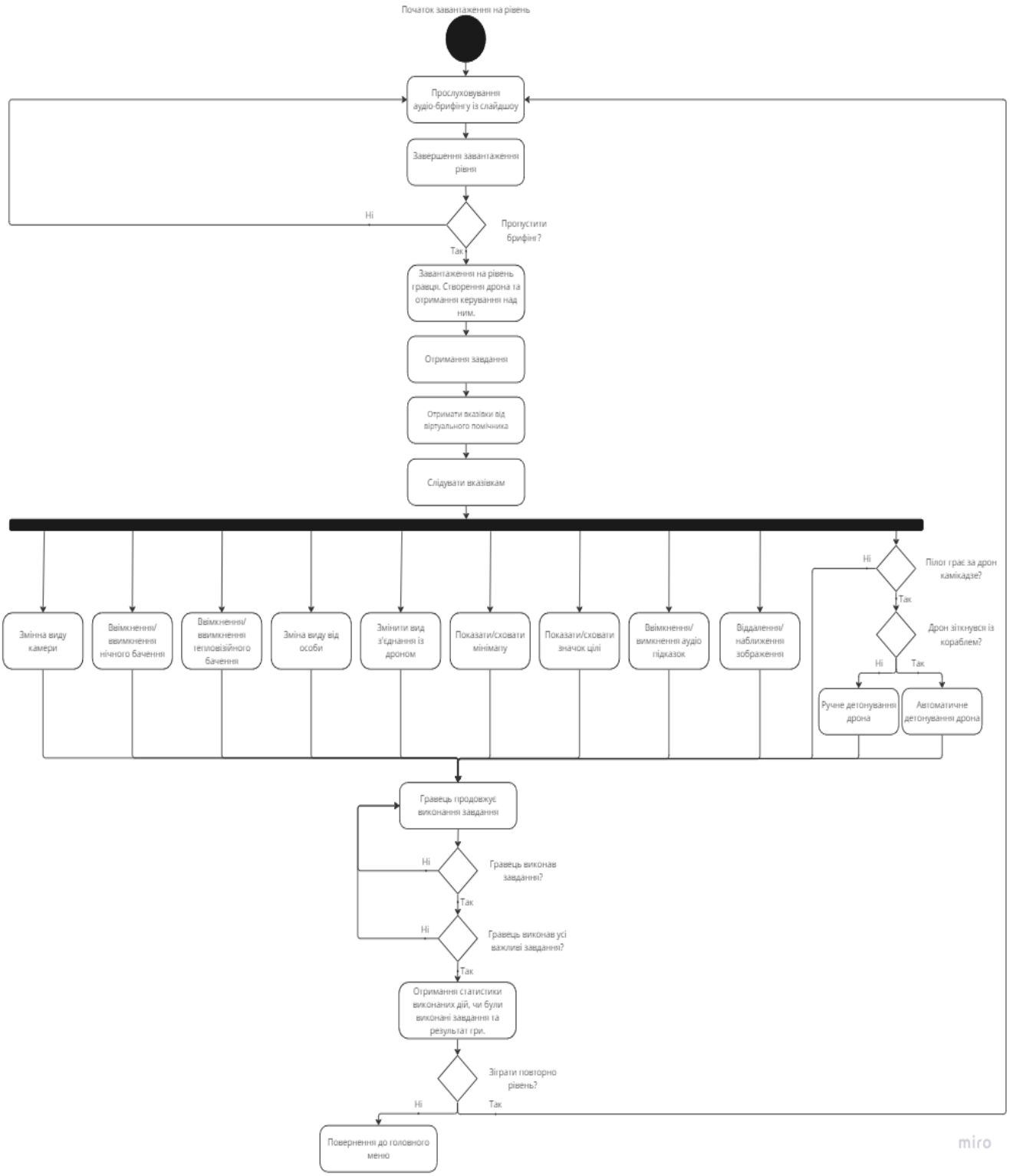

Рисунок 2.12 – Діаграма діяльності режиму гри «Тренувальні місії»

2.3.2 Меню паузи

Меню паузи є допоміжним меню, що викликається гравцем під час симуляції. Викликане меню зупиняє весь ігровий процес та надає операторові відкоригувати

налаштування, переглянути детальніше завдання або вийти у головне меню. Головний функціонал меню паузи включає в себе:

– відновити ігрову сесію. Продовжити розпочату симуляцію;

– налаштування. Налаштування є ідентичними до тих, що у головному меню;

– перезавантажити рівень. Дозволяє оператору перезапустити поточний сценарій з самого початку;

– повернутися до головного меню. Завершити сеанс та повернути користувача до головного меню, гарантуючи збереження змін перед виходом.

Для кращого розуміння, опишемо це меню у вигляді діаграми діяльності (рис. 2.13).

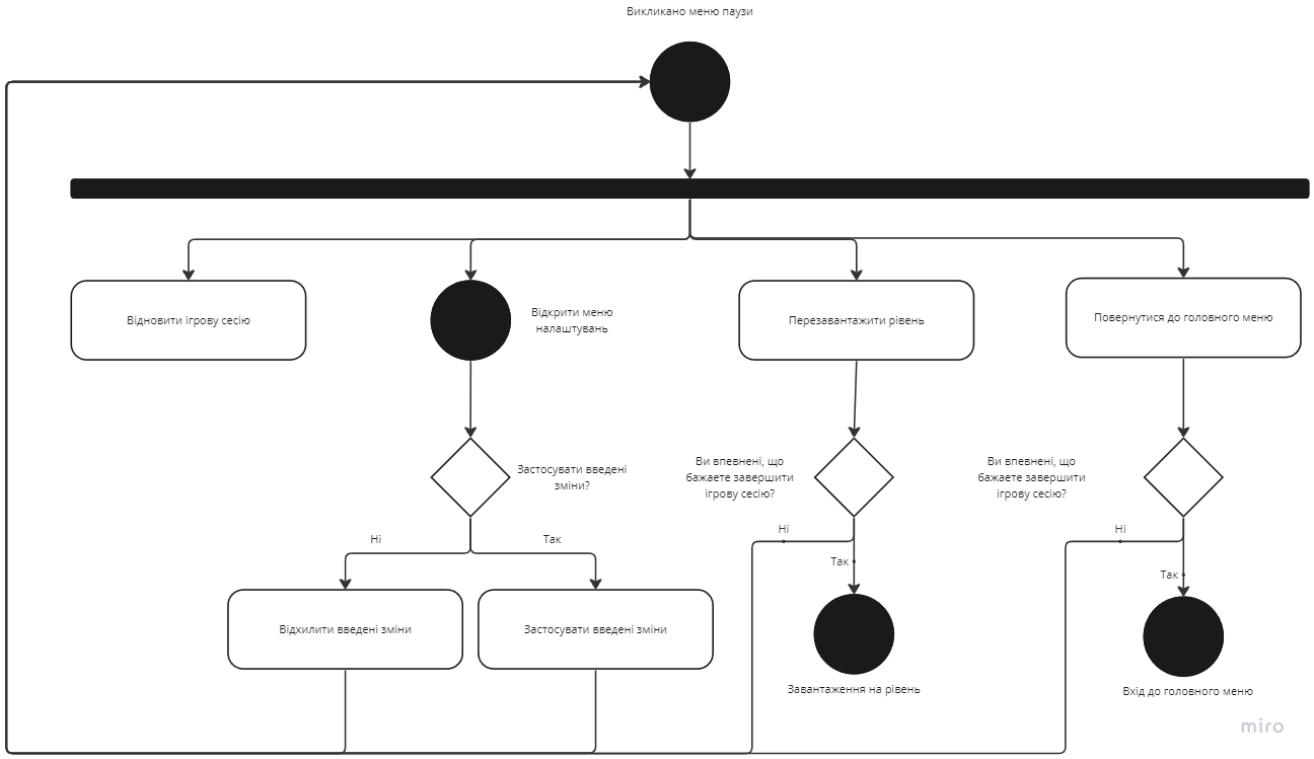

Рисунок 2.13 – Діаграма діяльності меню паузи

2.3.3 Інтерфейс дрона

Інтерфейс дрона є важливим компонентом симуляції, оскільки за допомогою нього користувач дізнається усе про стан НА, а також про потенційні небезпеки.

Оскільки симулятор імітує реальний апарат, то за основу слід взяти інтерфейс реальної моделі, з додаванням візуальних підказок для спрощення виявлення потенційних небезпек. Приклади такого інтерфейсу наведені на рис. 2.14 [13], 2.15 [14] та 2.16 [15].

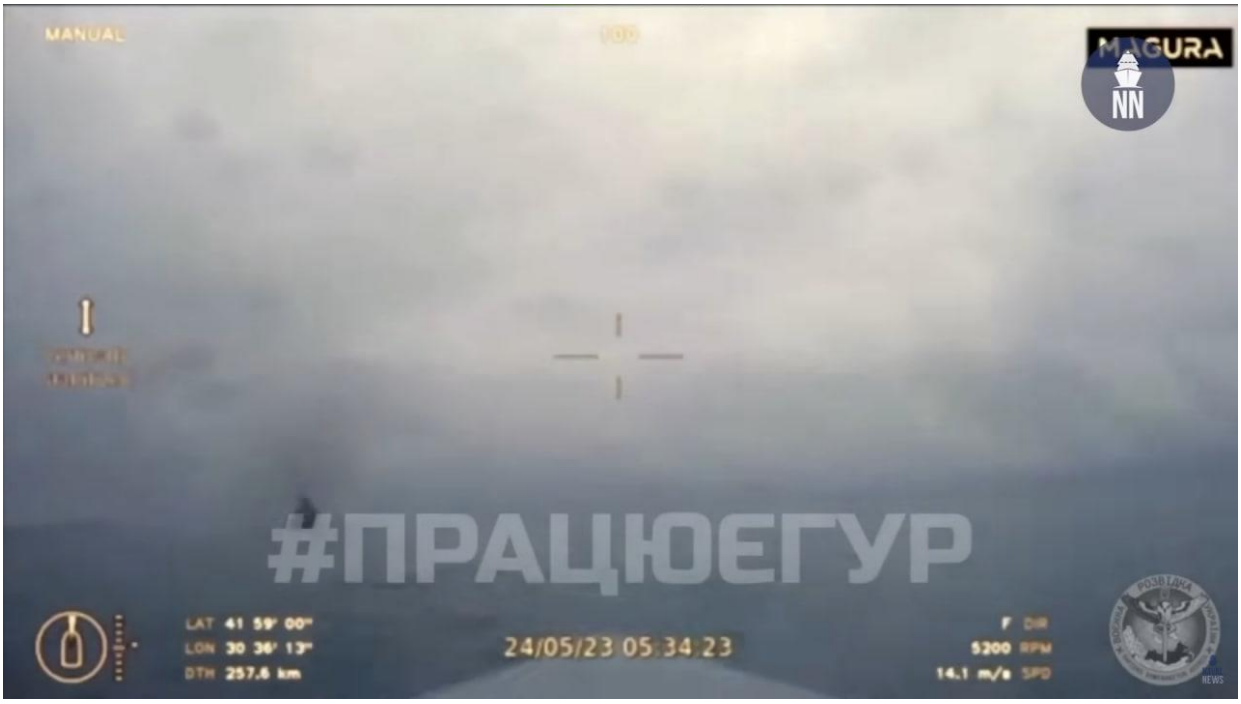

Рисунок 2.14 – Звичайне бачення з передньої камери Magura V5

Рисунок 2.15 – Тепловізійне бачення з передньої камери Magura V5

Рисунок 2.16 – Нічне бачення з передньої камери Magura V5

Як бачимо, зліва на фото присутні усі підказки стосовно тієї чи іншої функції, знизу відображається навігаційні, часові та дані про швидкість.

Для початку, проаналізуємо ліву панель. За фотографіями, у ній відображаються такі підказки (зверху униз):

– тип керування апаратом (ручний чи автопілот);
– режими бачення (нічний чи звичайний);
– тип стабілізації зображення.

Тип стабілізації не буде потрібним, оскільки у симуляторі не буде присутнім тремтіння камерою. Загалом такий набір цілком задовольняє, але на додачу до цих функцій, можна додати таку інформацію:

– яку камеру було вибрано (передню чи задню);
– тип з’єднання із дроном.

Також до режиму бачення можна додати окрему іконку тепловізійного бачення.

У нижні панелі є такі показники (зліва направо):

– направлення та кут нахилу камери;
– ширина та довгота, на якій знаходиться БНА (LAT та LON);
– пройдена відстань (DTH);
– поточні дата та час;
– направлення руху дрона (DIR);
– кількість обертів на хвилину (RPM);
– швидкість(SPD).

Серед усього зазначеного немає потреби додавати ширину, довготу та пройдену відстань, оскільки не треба створювати великий рівень, подібний до реального Чорного моря. Поточні дата та час є теж неважливими елементами, оскільки час у грі буде статичним, тобто фаза дня не зміниться після створення рівня. Краще замість дати та часу встановити таймер, щоб оператор знав кількість часу витраченого на місію. Значення кількості обертів у двигуні внутрішнього згоряння не несе корисної інформації для симулятора.

До показників на нижні панелі можна додати такі:

– рівень палива;
– попередження про входження у зону РЕР.

Додатково слід створити праву панель, де буде вказано відстань до цілі.

Напрям до цілі буде розміщено по середині екрану у вигляді невеличкого знаку локації (рис. 2.17), який буде пересуватися, залежно від місцезнаходження цілі. Для прикладу, якщо ціль перебуває від 90 до 180 градусів лівіше від напряму, куди дивиться гравець, тоді значок буде максимально зліва [16].

Рисунок 2.17 – Значок місцезнаходження цілі

Створивши детальний дизайн симулятора оператора надводного дрона, вдалося визначити:

– архітектуру застосунку, що складається з:

- головного меню;
- меню паузи;
- ігрового рівня, який може бути або тренувальним, або сценарним.

– потік ігрового процесу, як одні компоненти взаємодіють з іншими. Це стане у нагоді при:

– можливості надводного дрона. Окрім функціоналу вбудованого у реальну модель, було прийнято рішення додати й інші ігрові можливості для полегшення навчання нових кадрів;

– можливості ворогів. Це допоможе у підготовці фахівців, для розгляду різних сценаріїв;

– можливості пасток та перешкод;

– користувальницький інтерфейс. До нього відноситься не тільки головне та меню паузи, але й інтерфейс дрона.

Потік ігрового процесу описано у вигляді діаграм діяльностей, щоб краще зрозуміти, як будуть відбуватися ті чи інші процеси у симуляторі.

Надалі створений дизайн спросить розробку та тестування програмного застосунку.

3 РОЗРОБКА СИМУЛЯТОРА

3.1 Розробка графічного інтерфейсу

Перш за все необхідно підготувати основу для симулятора. Необхідно запустити ігровий рушій Unreal Engine, зліва вибрати тип проєкту Games. Рушій пропонує шаблон уже готових рішень для ігрових застосунків різних жанрів, щоправда, краще розпочати з чистого аркуша для більш гнучкого редагування та додавання нового коду у проєкт. Тому вибираємо порожній проєкт. Основною мовою програмування буде C++. Інші опції можна не чіпати, оскільки вони не стануть у нагоді (рис. 3.1).

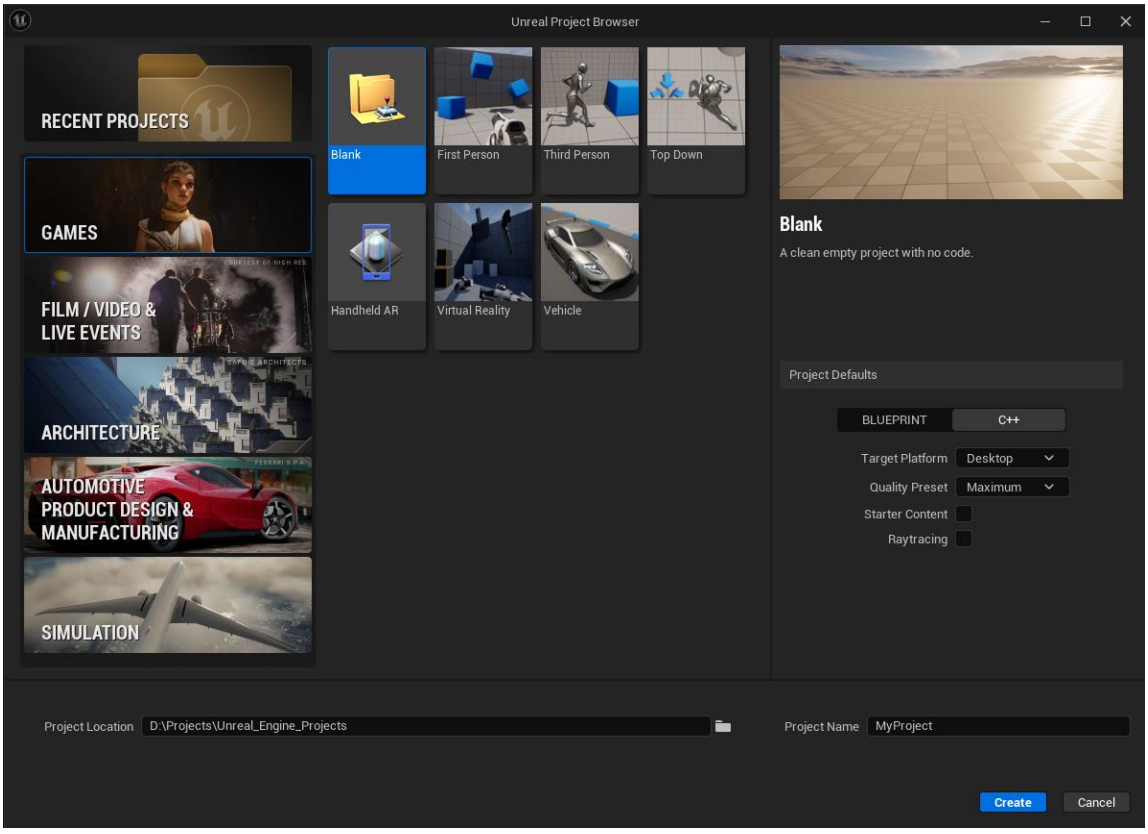

Рисунок 3.1 – Створення нового проєкту

Далі необхідно імпортувати усі необхідні графічні, відео та аудіо матеріали, що будуть використовуватися у роботі, у «Content Browser». Краще заздалегідь подбати про сортування файлів, що будуть використовуватися, для цього, створимо нову папку під назвою «Assests» у «Content Browser». У ній необхідно створити

папки «Characters» (зберігає Blueprint класи, 3D-моделі та текстури персонажів), «UI» (зберігає Blueprint класи графічного інтерфейсу та візуальні елементи, що присутні у ньому, для прикладу зображення), «Audio» (зберігає аудіо файли, які використовуються для створення діалогів та ефектів), «Maps» (зберігає ігрові рівні), «Animation» (зберігає анімаційні класи для персонажів), «VFX» (зберігає візуальні ефекти, а також текстури та анімації необхідні для їх створення), «Input» (зберігає класи дій на введення та розкладок контролю) «Level Design» (зберігає матеріали, які стосуються створення ігрового середовища.

Весь C++ код зберігається у папці C++ Classes, яка з'являється, як тільки створюється перший користувальницький клас.

Тепер треба підключити важливі розширення рушія (плагіни), щоб розпочати подальшу розробку.

У якості інтегрованого середовища розробки вибрано Visual Studio, оскільки є офіційна підтримка сумісної розробки за допомогою цих інструментів. Перейшовши до застосунку Epic Games Store, натиснемо зліва на розділ Unreal Engine, далі зверху – Marketplace. У пошуковій ділянці вводимо «Visual Studio Integration Tool», після чого необхідно забрати безкоштовний плагін, зображений на рис. 3.2.

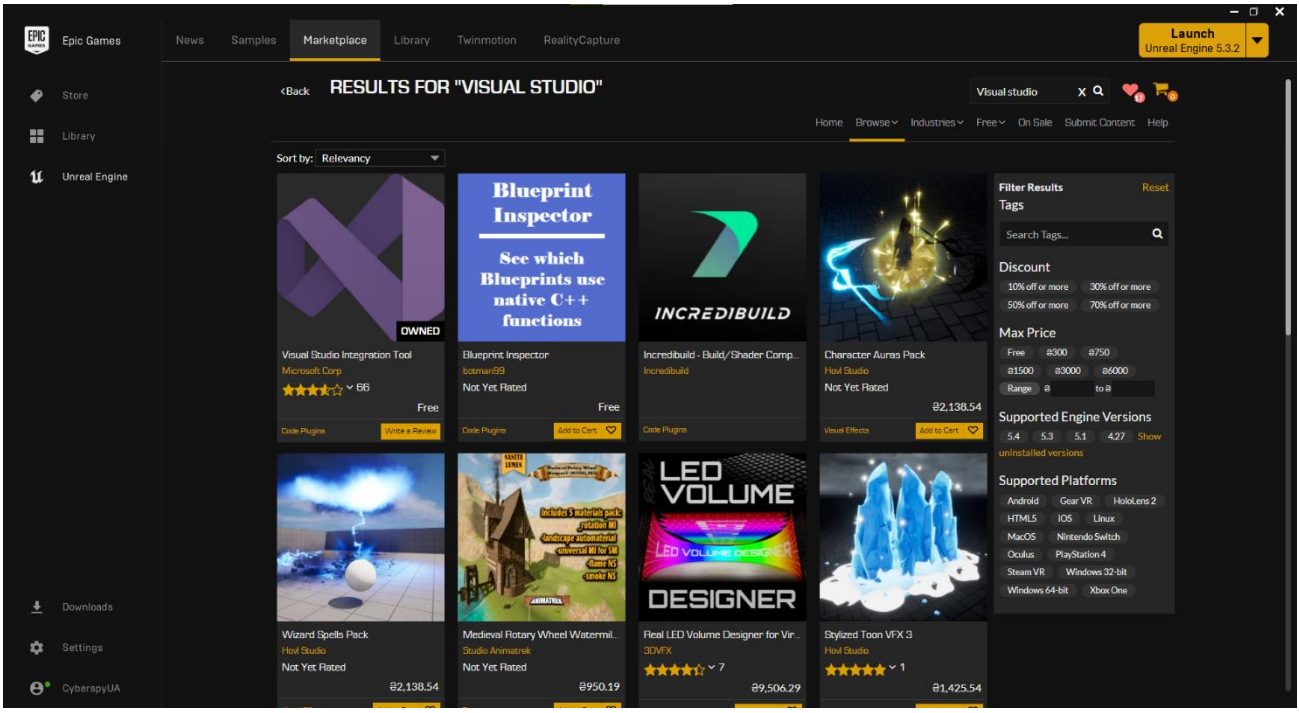

Рисунок 3.2 – Visual Studio Integration Tool у маркетплейсі

Після цього переходимо до вкладки Library, що є зверху, натискаємо на Visual Studio Integration Tool, клікаємо по «Download» та вибираємо версію ігрового рушія. Тепер ігровий рушій з'єднаний з інтегрованим середовищем Visual Studio. Щоб перевірити це, варто перейти до рушія та у верхній панелі натиснути на вкладку Tools, після чого на Open Visual Studio.

Unreal Engine має плагін, за допомогою якого можна створювати симуляцію води, що має назву «Water». У вікні рушія необхідно клікнути по Edit, далі – Plugins (рис 3.3). У пошуку вводимо «Water» та встановлюємо розширення (рис 3.4). Слід зазначити, щоб зміни набули чинності, треба перезавантажити рушій.

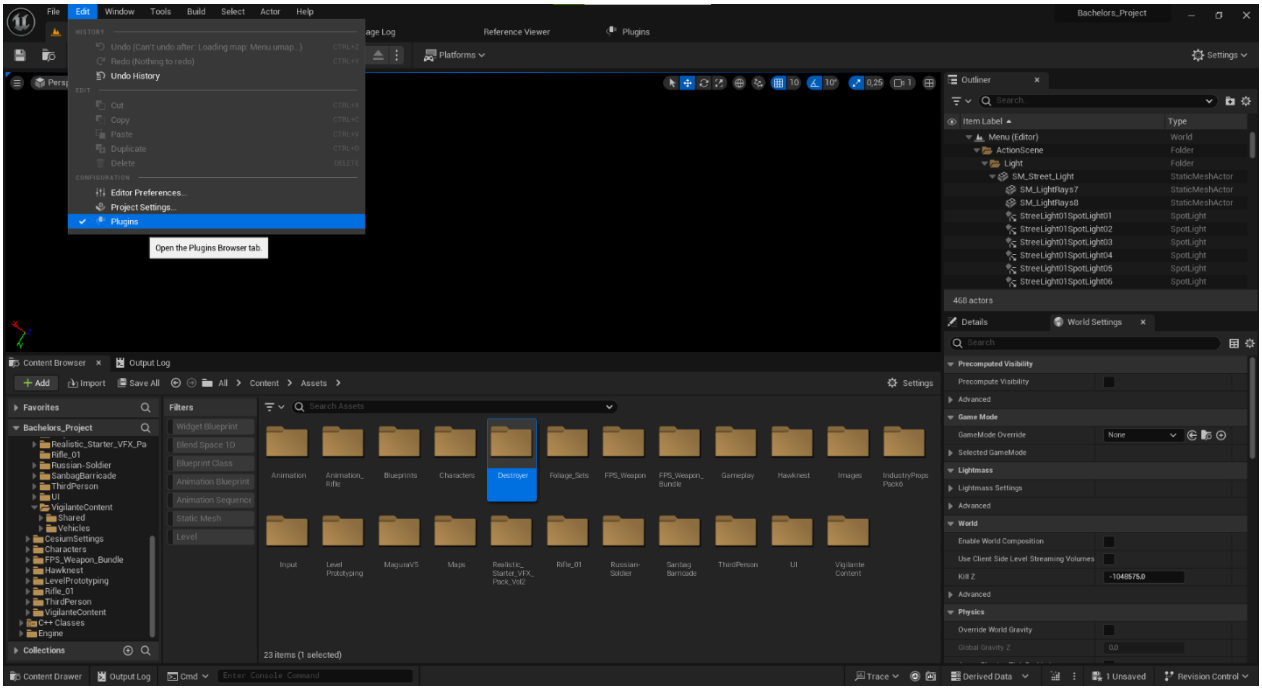

Рисунок. 3.3 – Розташування Plugins

Переходимо до наступного етапу – розробка графічного інтерфейсу. Слід зазначити, що головне меню буде знаходитися на окремому рівні, оскільки усі процеси в іграх створених на рушії Unreal Engine відбуваються на рівнях.

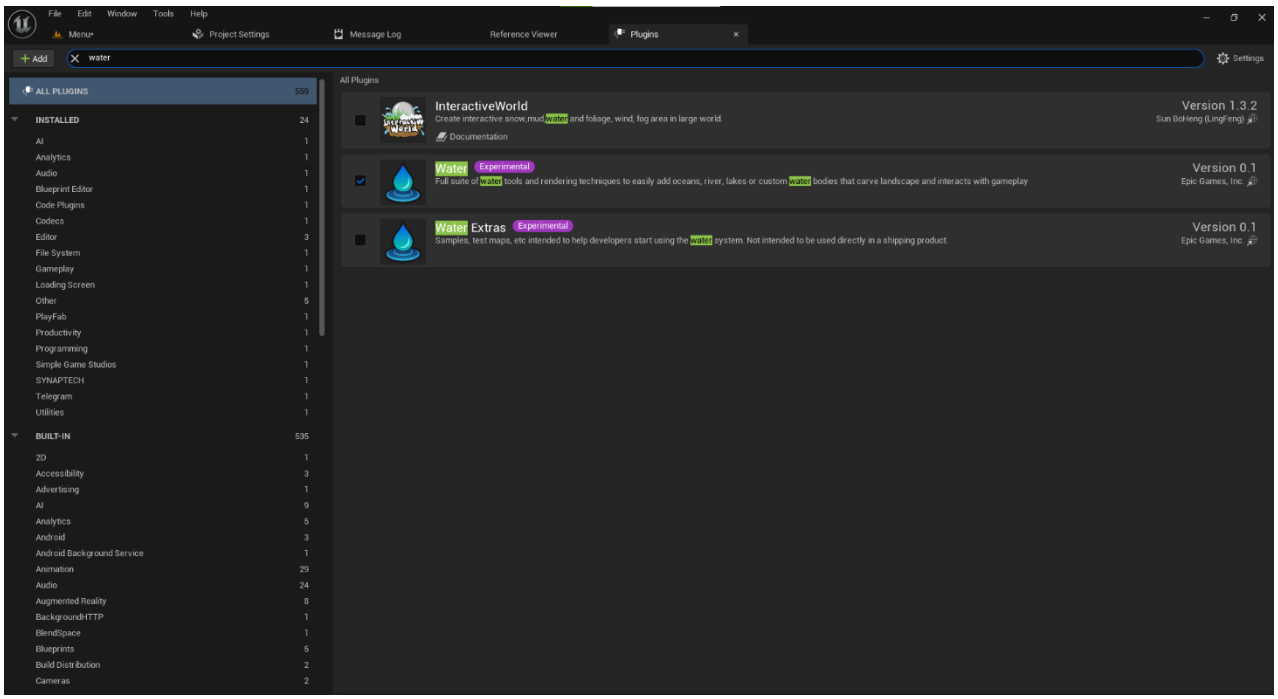

Рисунок. 3.4 – Розташування плагіну Water

У папці «Assets» створимо новий рівень, натиснувши правою кнопкою миші по пустому полю та вибравши «Level» в якості новоствореного ігрового елемента. Для короткого візуального ознайомлення гравця з подальшими викликами вирішено розробити головне меню у вигляді 3D-сцени, в якій при переході до іншого вікна налаштування, перспектива камери змінюється на інший ракурс.

Увесь графічний інтерфейс у Unreal Engine створюється у класах Blueprint widget. Створимо такий клас у папці UI та назвемо його «WBP_MainMenu». Після цього відкриємо цей віджет. Все, що стосується розробки графічної частини, відбувається у редакторі «Designer». Програмується логіка взаємодії між графічним інтерфейсом та його елементами або ігровими об’єктами у середовищі «Graph».

Редактор «Designer» пропонує готові варіанти елементів графічного дизайн. У нашому випадку краще створити кнопку окремим графічним віджетом, оскільки це надасть більше гнучкості у редагуванні стилю та логіки. Створимо новий віджет, назвемо його WBP_Button. Графічна частина показана на рис. 3.5.

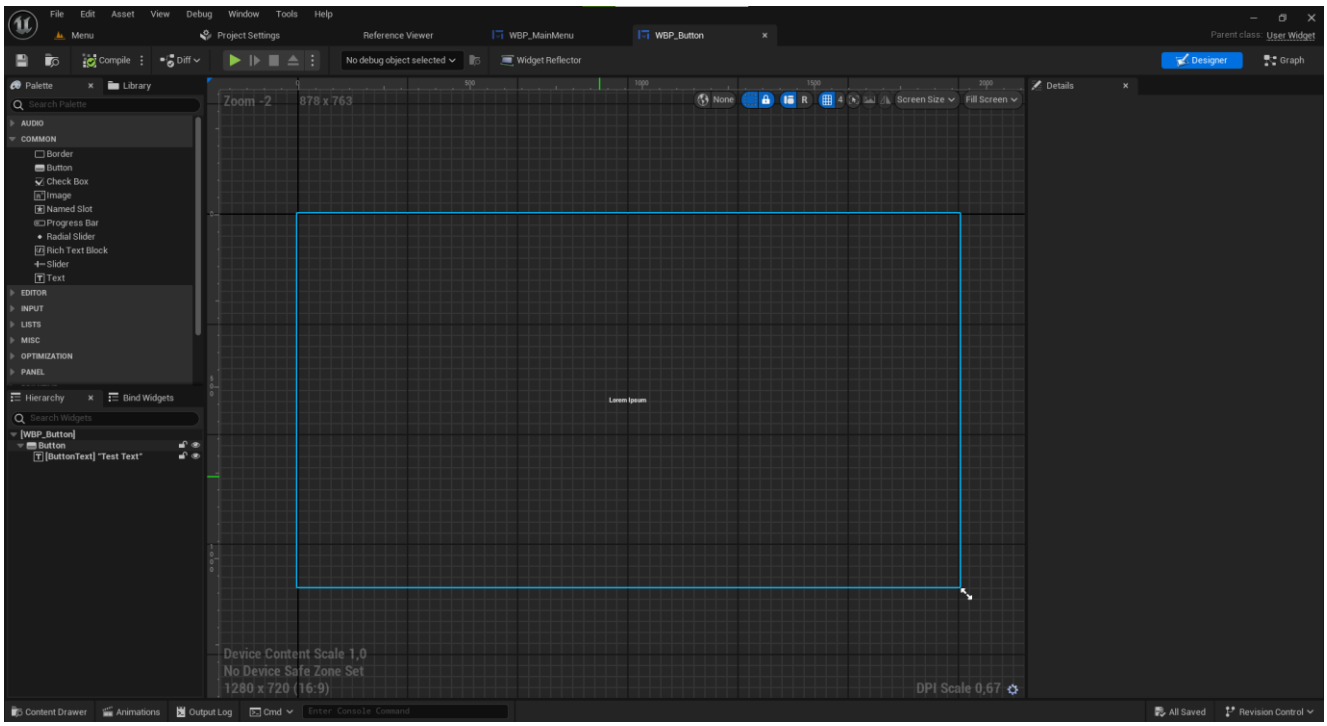

Рисунок 3.5 – Графічна частина віджета WBP_Button

Наступним кроком буде додання логіки ініціалізування кнопки:

– задання введеного у параметрах тексту;

– зміна кольору кнопки на визначений колір при наведені на неї;

– повернення до кольору нормального фону при прибиранні миші з кнопки;

– виклик визначеної користувачем події.

Логічна частина віджету зображена на рис. 3.6.

Повернемося до нашого віджету головного меню. Головним елементом у ньому та у подальших віджетів буде елемент Canvas Panel, що є умовним просторовим обмеженням, необхідним для адаптації графічного інтерфейсу. У Canvas Panel додамо Vertical Box, що відповідає за вертикальне розміщення елементів усередині цього елемента. Для того щоб зробити простір поміж кнопками, необхідно додати декілька Spacer поміж ними (рис. 3.7).

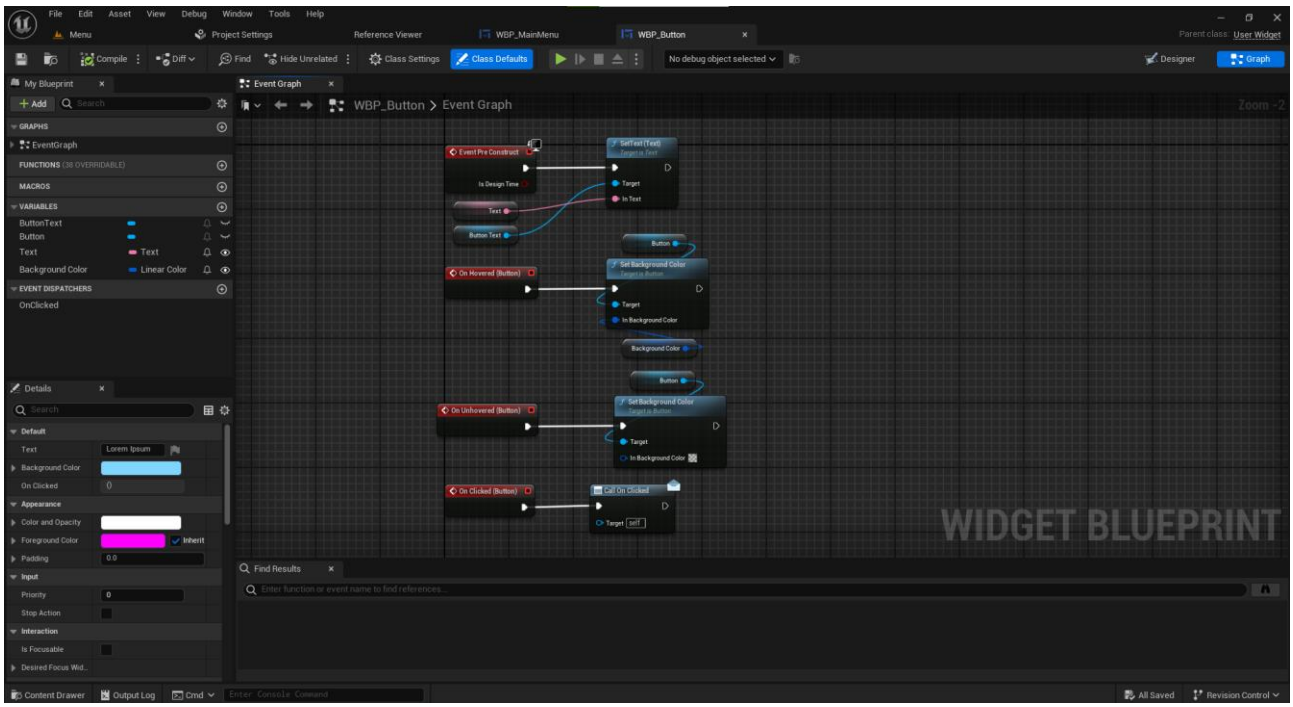

Рисунок. 3.6 – Логічна частина WBP_Button

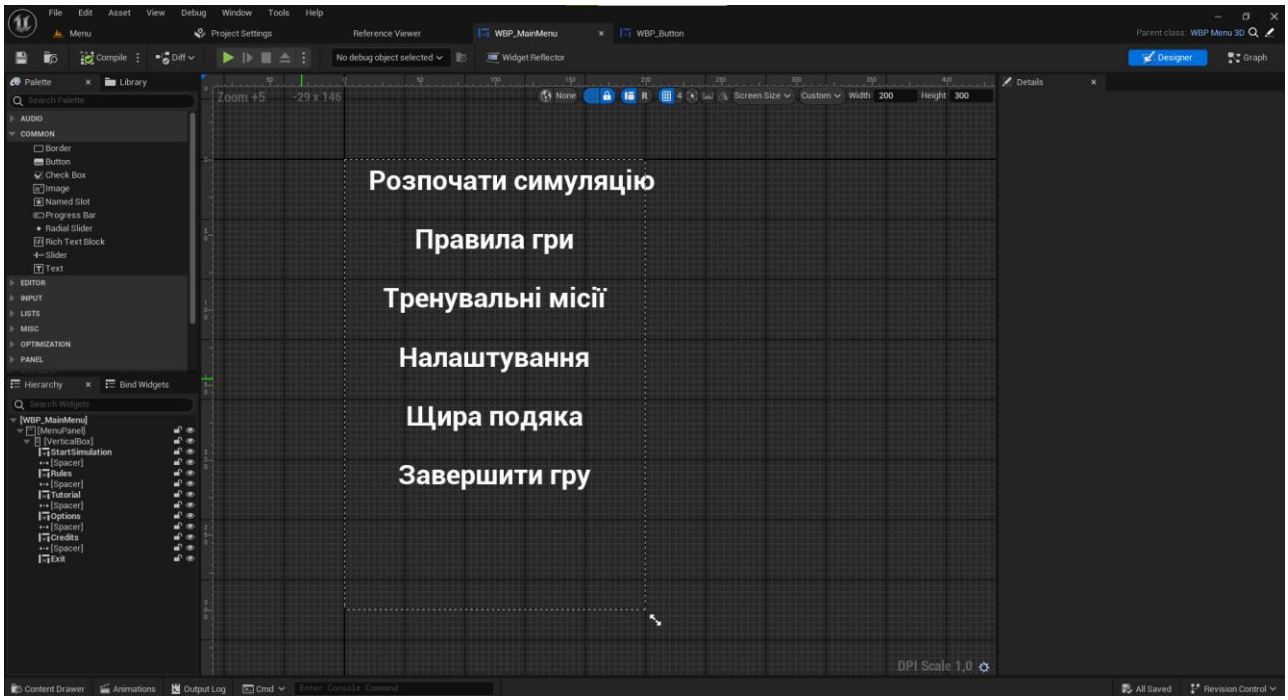

Рисунок 3.7 – Графічна частина WBP_MainMenu

Раніше зазначалося, що камера буде змінювати свою перспективу та положення для відображення сцени з різних ракурсів, тому необхіден клас Blueprint, який відповідатиме за цей перехід.

Створемо новий Blueprint клас та назвемо його «BP_Menu3D». Оскільки головне меню має декілька віджетів, до яких WBP_MainMenu звертається, необхідно створити у BP_Menu3D масив, що зберігатиме посилання на графічний віджет, до якого можна звернутися за індексом, який визначено при додані його у масив. Додатково необхідно перевірити, чи є наявним графічний віджет, до якого звернулися за індексом, якщо цього не зробити, застосунок аварійно завершить свою роботу. Також, слід взяти до уваги те, що хоч камера і робить переміщення у іншу позицію та змінює перспективу, щоправда колізія від попереднього віджета зберігається, тому при переході до нового меню, слід вимкнути її та ввімкнути колізію нового меню (рис. 3.8).

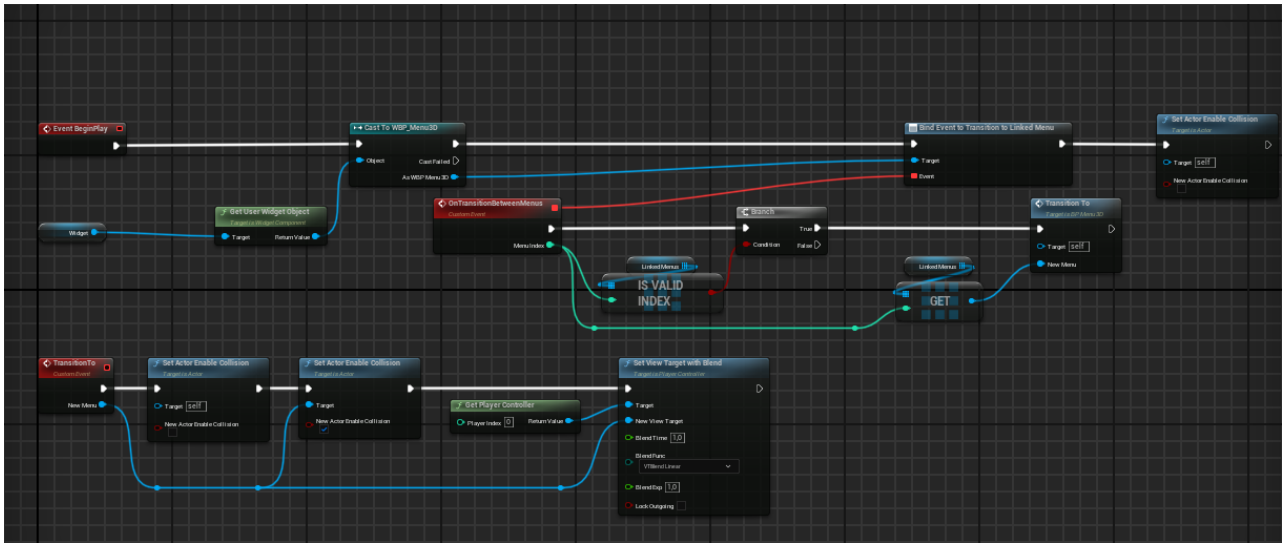

Рисунок 3.8 – Логічна частина BP_MainMenu3D

Далі створимо віджети з назвами:

– WBP_Difficulties. Викликається при натисканні на «Розпочати симуляцію», у ньому задаються налаштування складності штучного інтелекту;

– WBP_Info. Викликається при натисканні на «Правила гри». Відображає правила користування симулятором у текстовому форматі;

– WBP_Tutorial. Викликається при натисканні на «Тренувальні місії». Відображає список тренувальних рівнів та опису про них;

– WBP_Options. Викликається при натисканні на «Налаштування». Відображає налаштування відео, аудіо та керування;

– WBP_Credits. Викликається при натисканні на «Щира подяка». Виводить список людей, які давали раду під час розробки симулятора;

– WBP_Exit_Insurance. Викликається при натисканні на «Завершити гру». Питає користувача, чи він хоче завершити роботу застосунку.

Перший віджет WBP_AI_Properties відповідає за вибір складності штучного інтелекту та передачі його до редактора сценарних рівнів. Умовно цей віджет можна розділити на верхню (список рівнів складності ШІ та опис) та нижню (кнопки переходу між віджетами) (рис. 3.9).

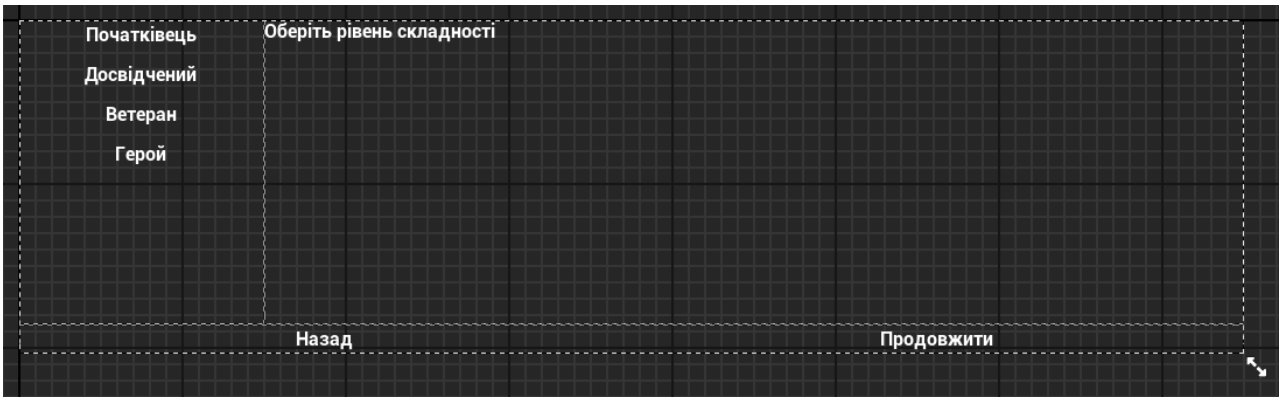

Рисунок 3.9 – Графічна частина WBP_AI_Properties

Складність ШІ можна зберегти у вигляді цілого числа. За кожен рівень відповідає певне число (1 – новачок, 2 – досвідчений, 3 – ветеран, 4 – легенда). У разі, якщо гравець не вибрав один із варіантів, застосунок не дасть змоги перейти на наступний етап. У системі Blueprint це виглядатиме так, як показано на рис. 3.10.

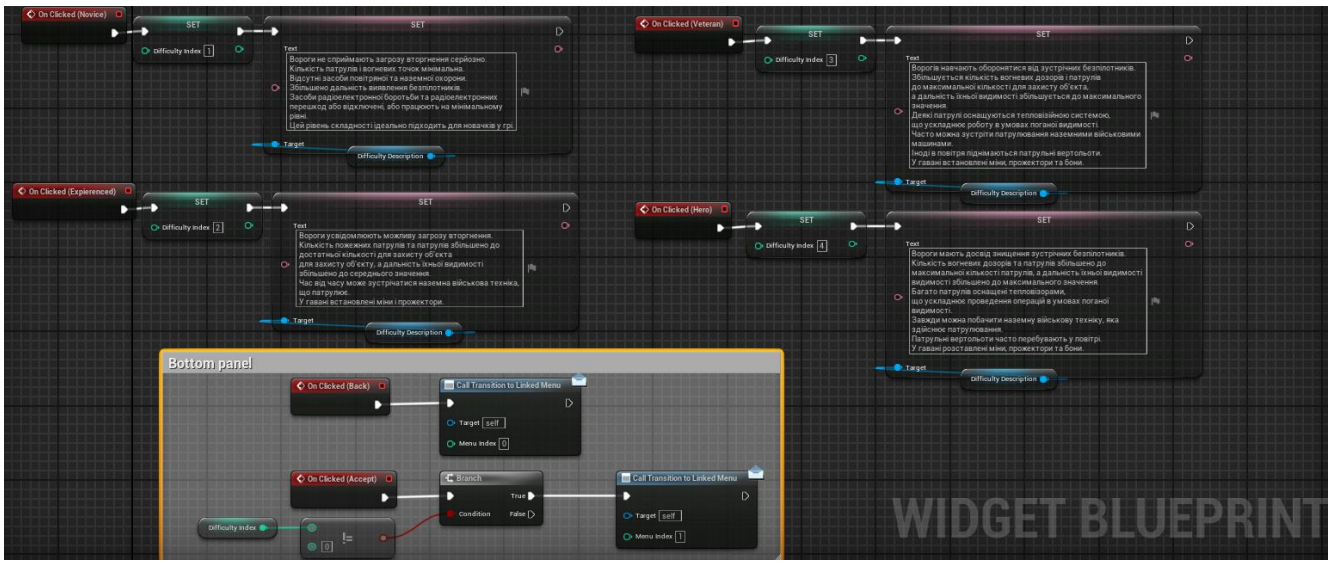

Рисунок 3.10 – Логічна частина WBP_AI_Properties

WBP_Info розділений на верхню частину, що містить інформацію про правила використання застосунку, та нижню, яка містить кнопки для переходу на попередню та наступну сторінки, а також повернення до головного меню. Віджет має вигляд, як на рис. 3.11.

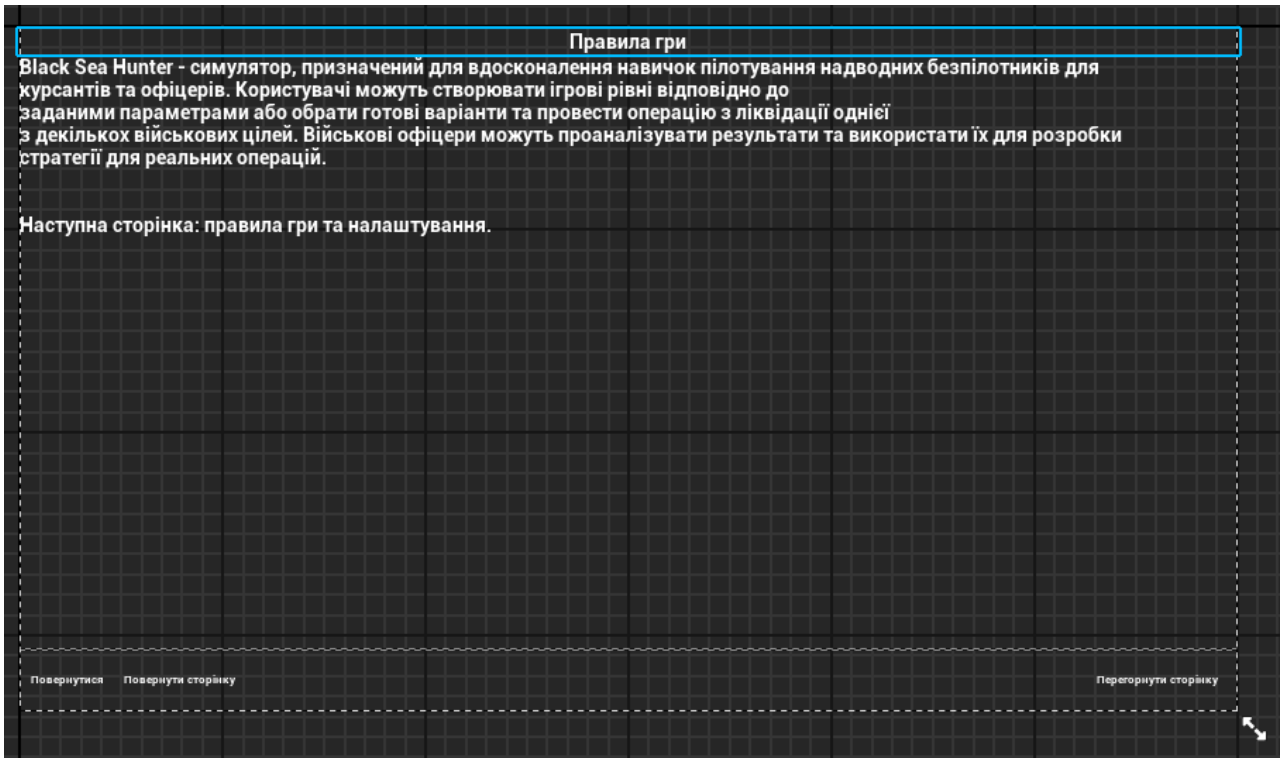

Рисунок 3.11 – Графічна частина WBP_Info

Номер сторінки можна зберегти у цілочисловій змінній. Використовуючи switch, можна за індексом сторінки змінювати текст, що є у верхній частині. Також необхідно встановити межу на максимальний та мінімальний індекс, щоб не було помилок, оскільки не знайшовши число у switch, жодних змін не буде (рис. 3.12).

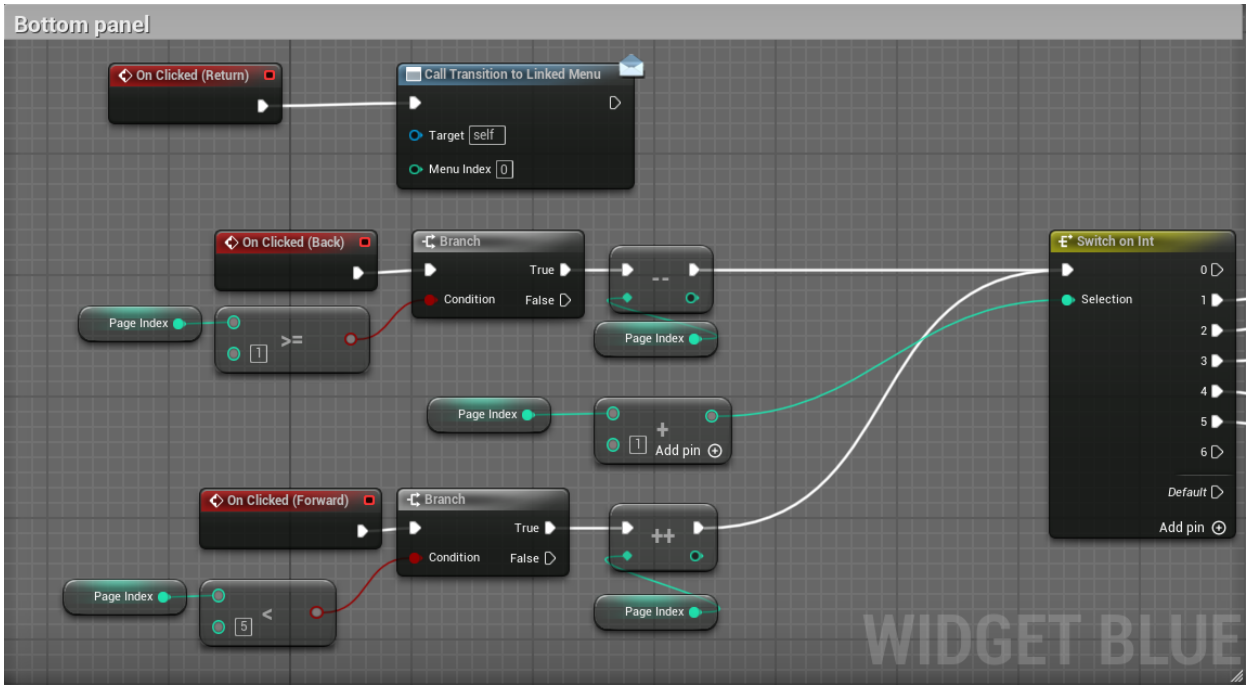

Рисунок 3.12 – Логічна частина нижньої панелі WBP_Info

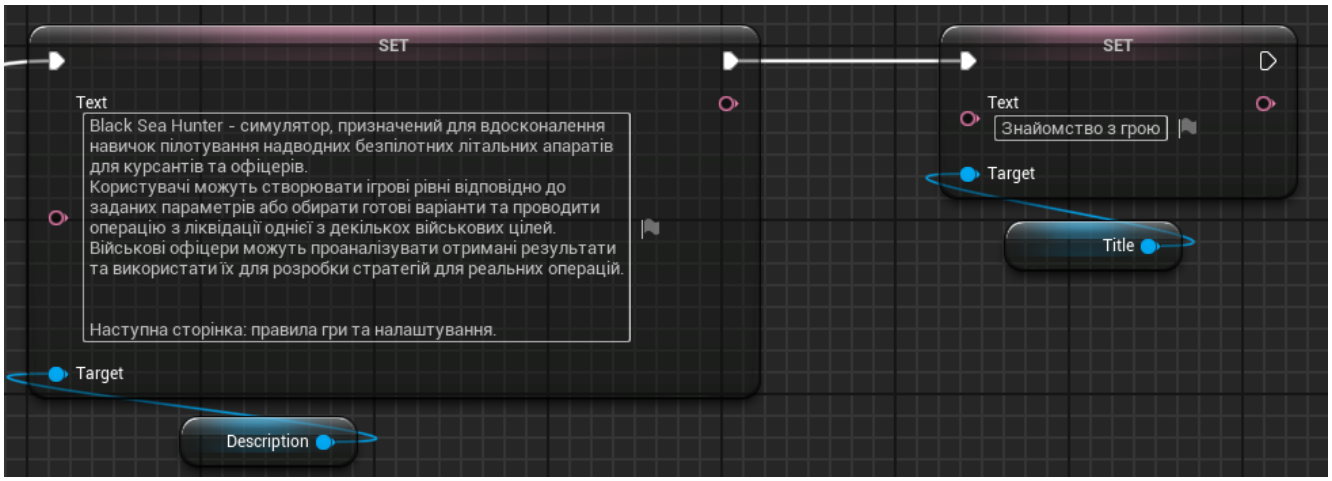

Рисунок 3.13 – Логічна частина одного із кейсів верхньої частини WBP_Info

WBP_Tutorial теж має верхню і нижню частини. Верхня частина є вертикальним списком тренувальних рівнів. Нижня частина – кнопка для

повернення назад та детальний опис кожного з тренувальних рівнів. Графічна частина має вигляд на рис. 3.14.

Рисунок 3.14 – Графічна частина WBP_Tutorial

Вибравши один із варіантів, гравцеві буде продемонстровано короткий опис місії. Нижня панель міститиме кнопки повернення назад та старт завантаження рівня. Скрипт цього функціоналу можна побачити на рис. 3.15. Скрипт щодо виведення окремого вікна з детальною інформацією про сценарій та місію зображено на рис 3.16. Функціонал кнопки «Повернутися» є на рис. 3.17.

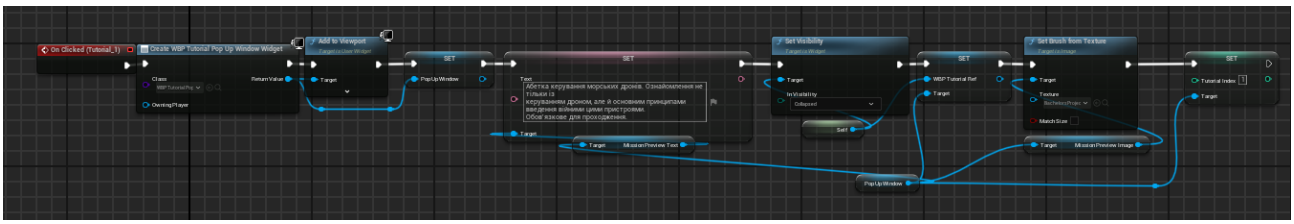

Рисунок 3.15 – Скрипт виведення вікна однієї із місій

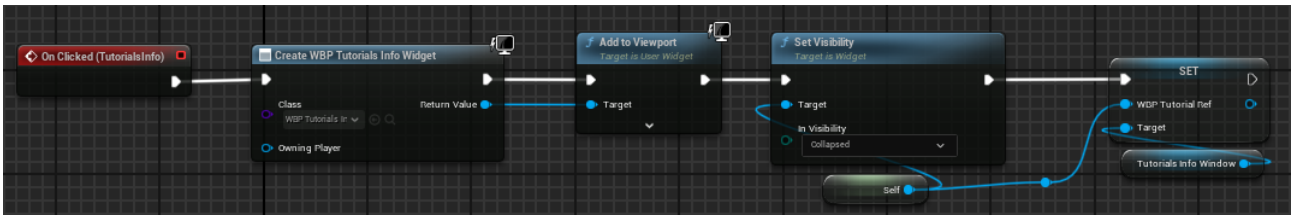

Рисунок 3.16 – Скрипт виведення короткої інформації про місію

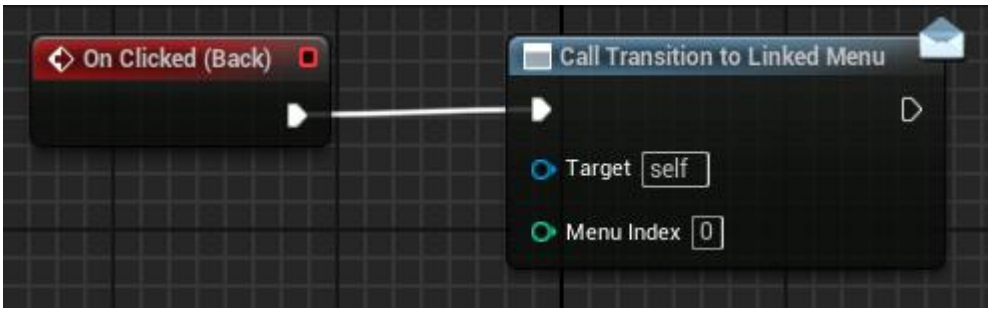

Рисунок 3.17 – Скрипт повернення на віджет назад

WBP_Options має аналогічну ієрархію елементів із WBP_Tutorial. Графічний вигляд віджета наведено на рис. 3.18.

Рисунок 3.18 – Графічна частина WBP_Options

Так само, як і з вікном, що виводить коротку інформацію про місію WBP_Tutorial віджета, маємо логіку на рис. 3.19.

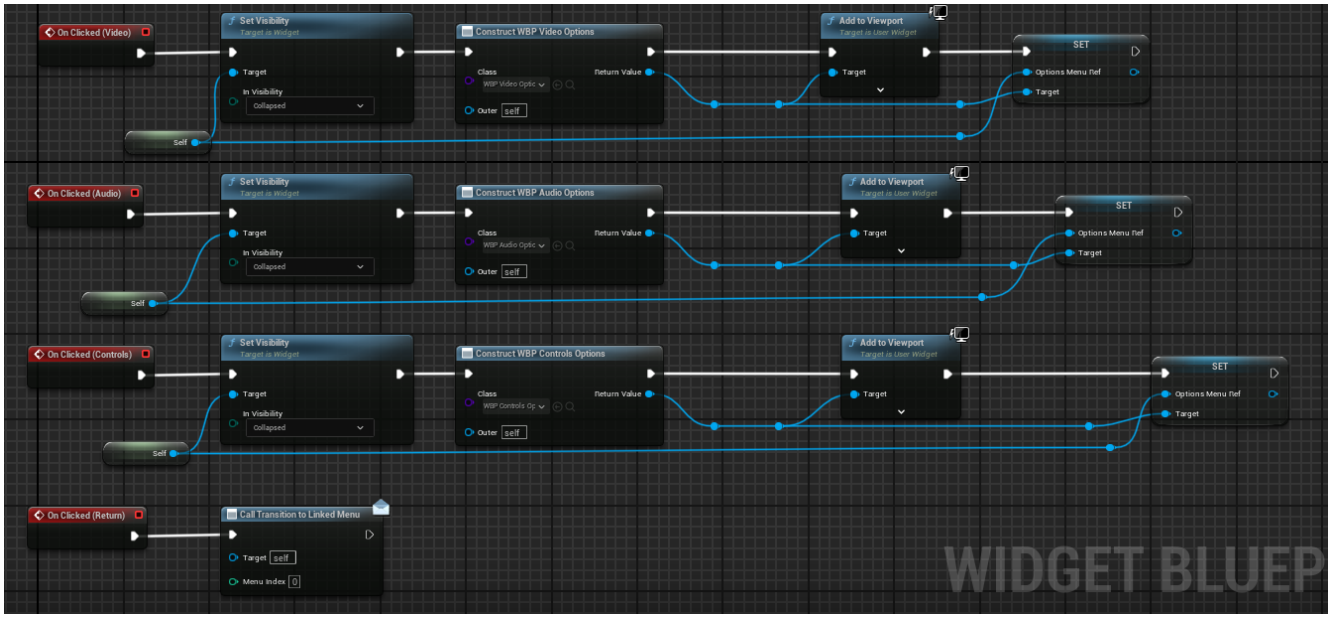

Рисунок 3.19 –Логічна частина WBP_Options

Розміщення графічних елементів у WBP_Credits таке саме, як у WBP_Options, тільки замість кнопок – текстове поле (рис 3.20).

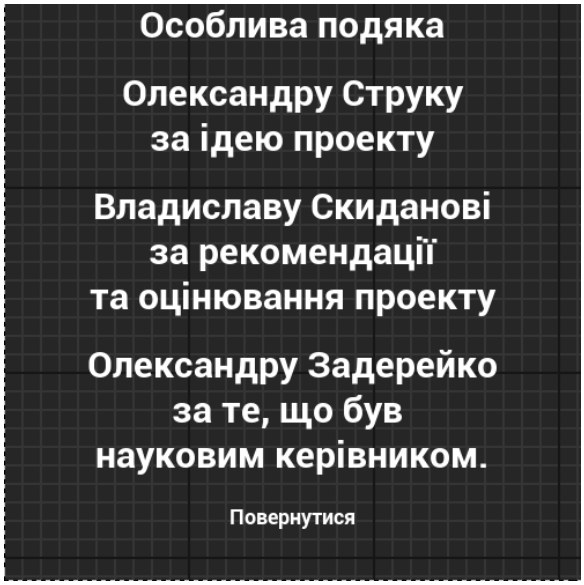

Рисунок 3.20 – Графічна частина WBP_Credits

Скрипт повернення назад має аналогію, як і у попередніх віджетах.

WBP_Exit_Insurance надає вибір гравцеві: повернутися до вікна головного меню, чи завершити роботу застосунку. Графічну частину можна побачити на рис. 3.21, а логічну – на рис 3.22.

Рисунок 3.21 – Графічна частина WBP_Exit_Insurance

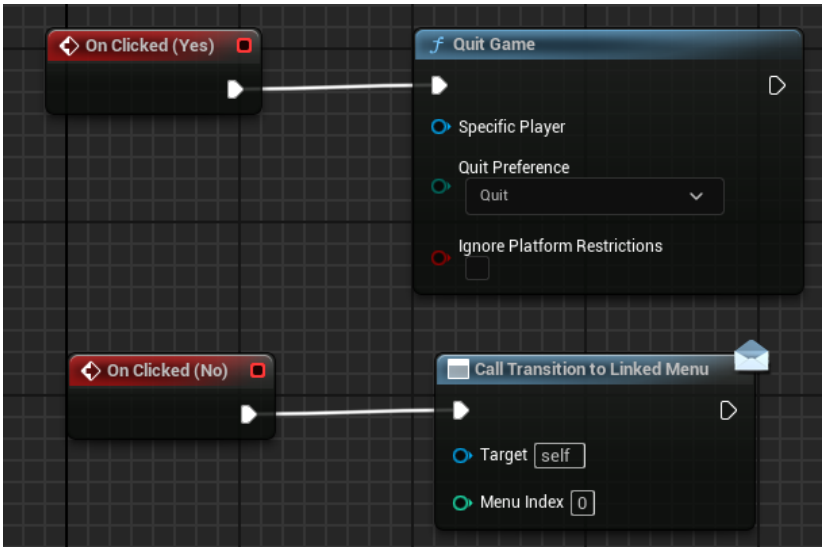

Рисунок 3.22 – Логічна частина WBP_Exit_Insurance

Коли гравець вибрав рівень складності у віджеті WBP_Difficulties, він переходить до редактора сценарного рівня, де може налаштувати параметри середовища, пасток, перешкод, сценарія та ворогів. Створимо новий віджет WBP_Level_Creator, який виступає у ролі редактора сценарних рівнів. Він

розподілятиметься на дві частини – верхню (де є усі налаштування) та нижню (панель переходу). У налаштуваннях середовища та місії запропоновано декілька варіантів вибору для кожного параметра, оскільки необхідно вибрати лише один, це можна зобразити у вигляді кнопок. У свою чергу, у налаштуваннях ворогів та пасток треба ввести ціле число, для цього можна використати поле введення. Візуальну частину віджета зображено на рис. 3.23.

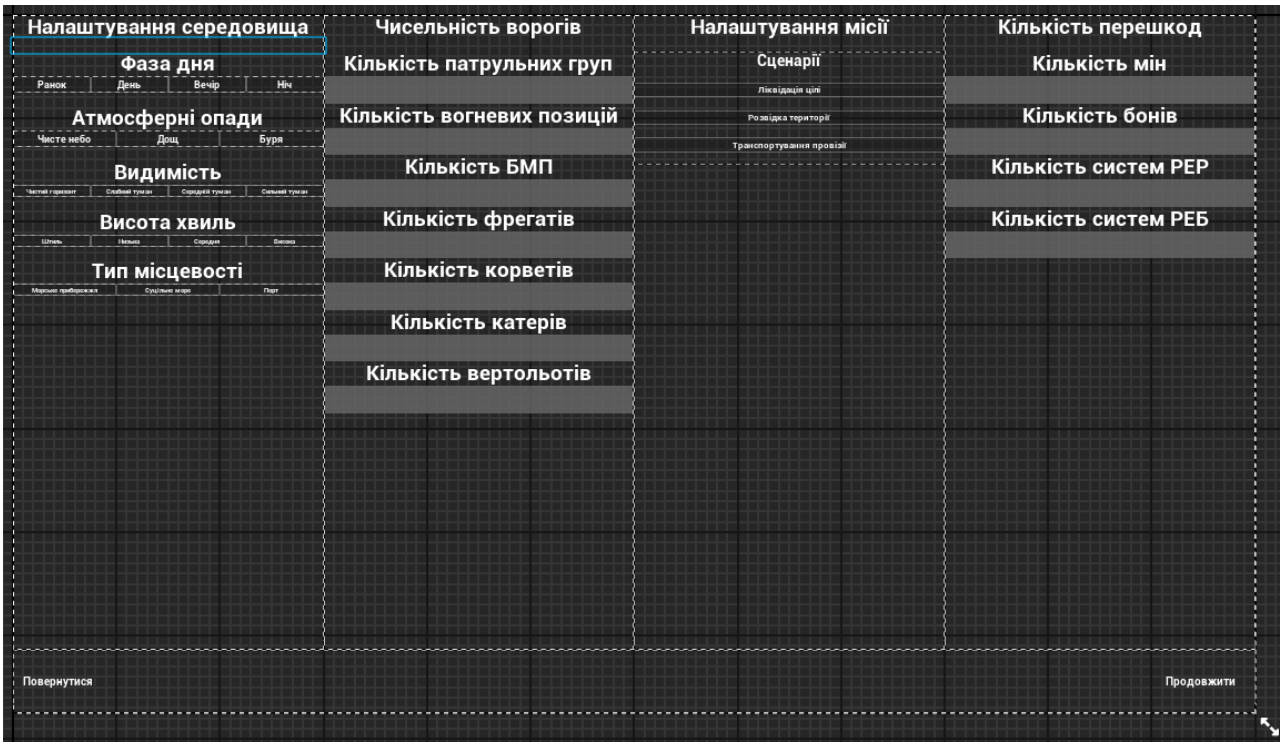

Рисунок 3.23 – Графічна частина WBP_Level_Designer

Після натискання на кнопку «Повернутися» гравець повертається до попереднього вікна. Перед тим як натиснути «Продовжити» та завантажитись на рівень, необхідно перевірити, чи всі зміни були застосовані та було вибрано тип місцевості і сценарій, а також кількість ворогів та пасток є у проміжку, визначеному в попередньому розділі. Далі можна побачити перевірку параметрів для створення рівня «прибережжя» (рис. 3.24), «суцільне море» (рис. 3.25) та «порт» (рис. 3.26).

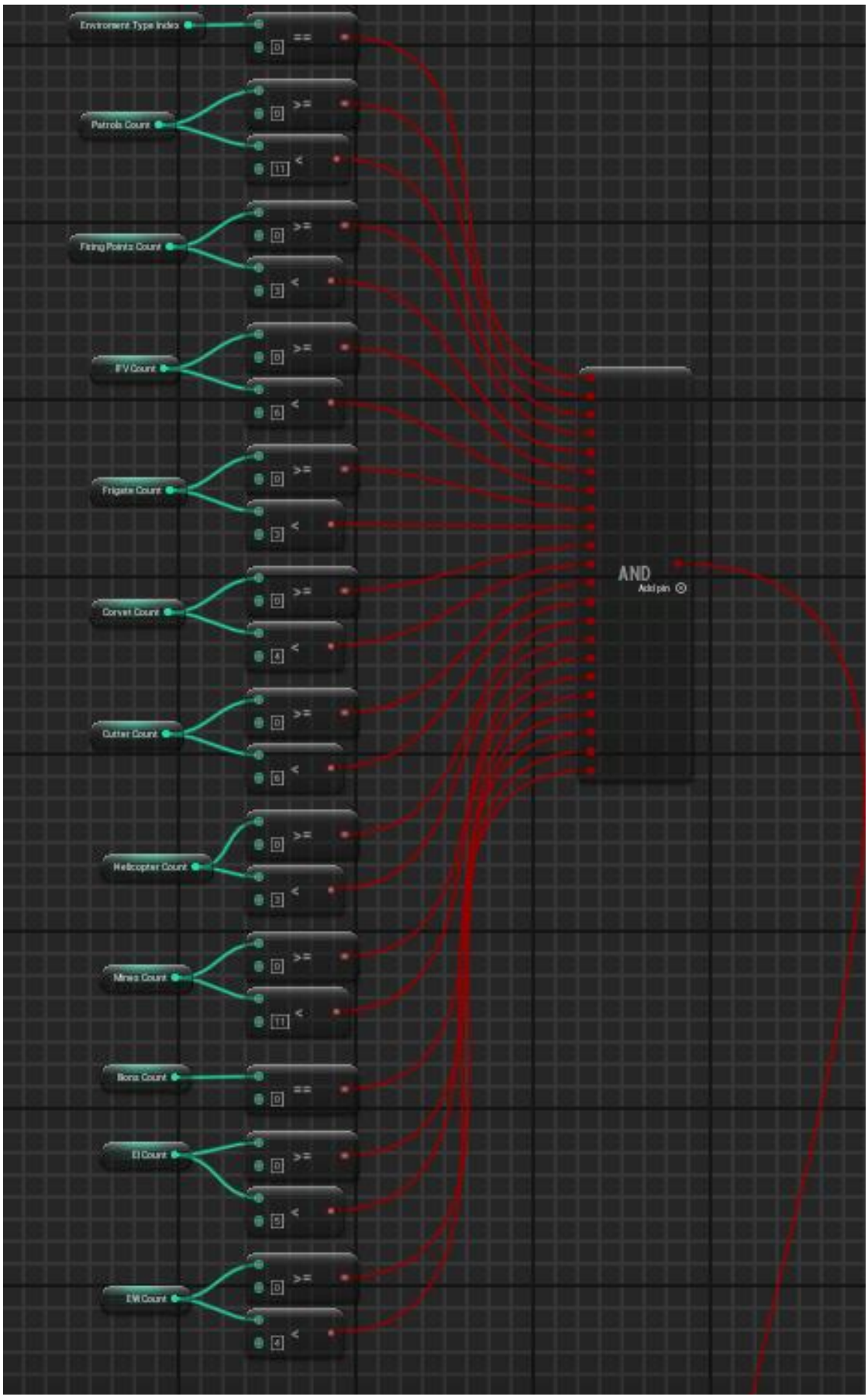

Рисунок 3.24 – Логічна частина WBP_Level_Designer. Перевірка змін значень параметрів для типу місцевості «прибережжя»

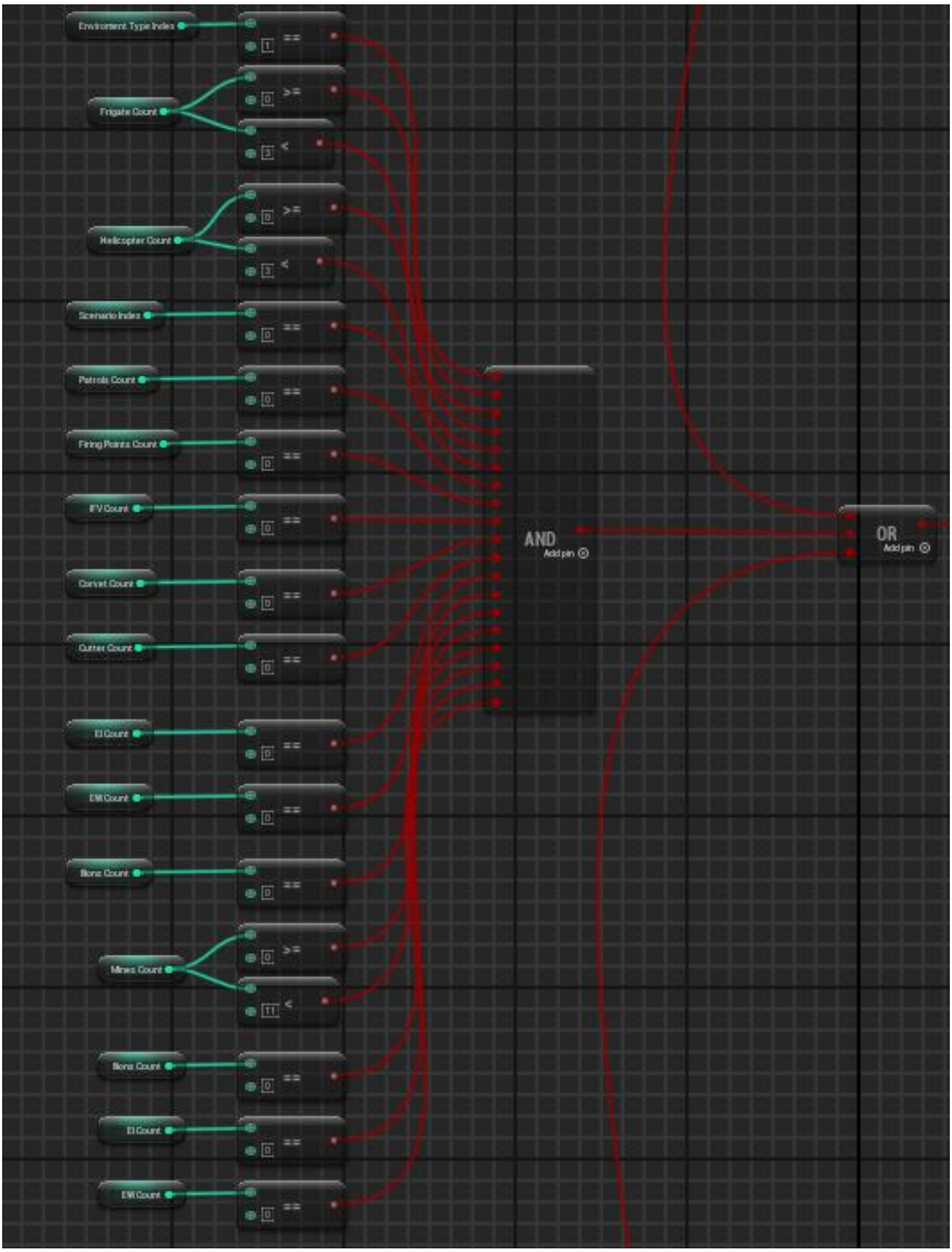

Рисунок 3.25 – Логічна частина WBP_Level_Designer. Перевірка змін значень параметрів для типу місцевості «суцільне море»

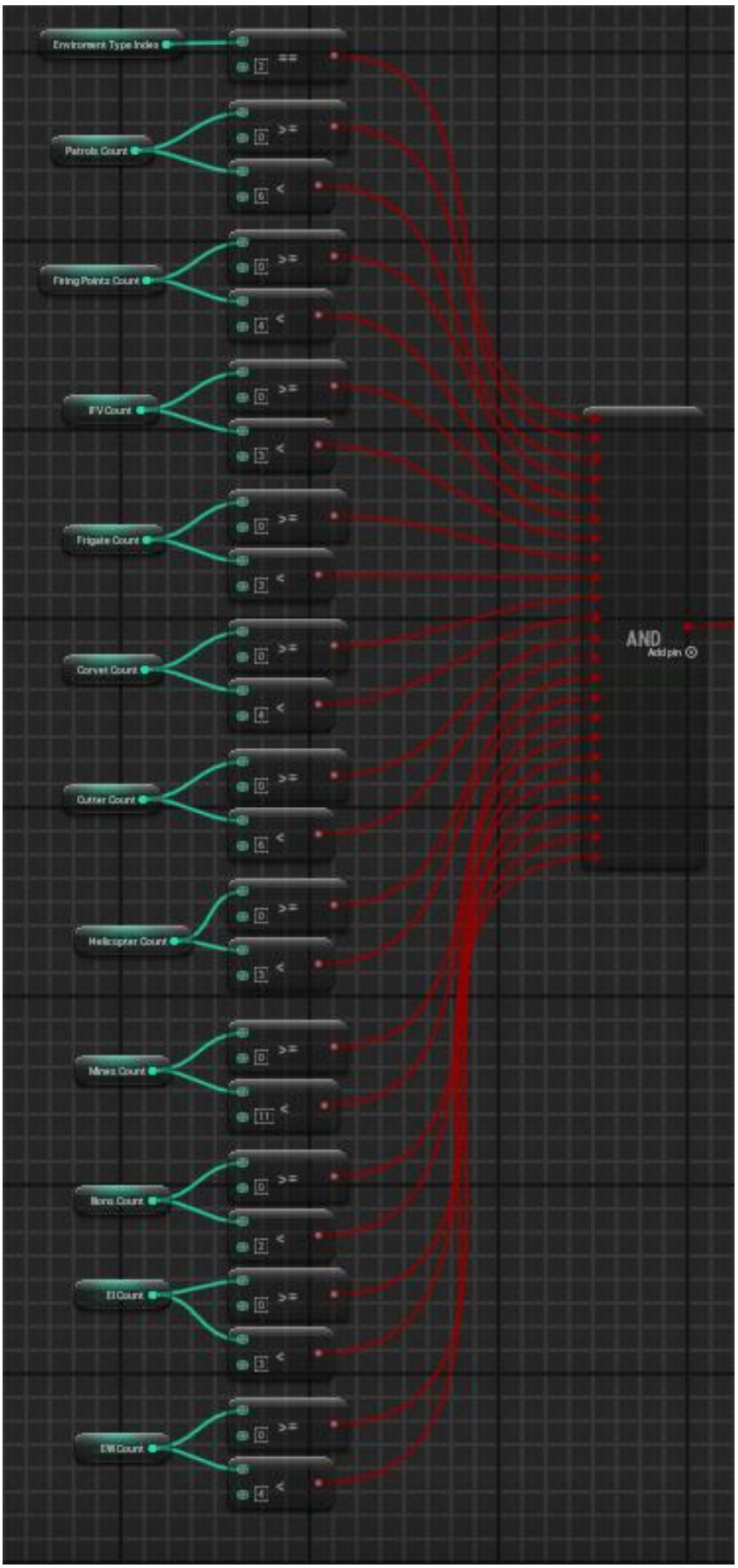

Рисунок 3.26 – Логічна частина WBP_Level_Designer. Перевірка змін значень параметрів для типу місцевості «порт»

Перед початком завантаження на рівень треба перевірити, чи є задовільними усі раніше зазначені умови та чи були внесені зміни. Якщо зміни не були внесені, то виводиться вікно, яке запитує користувача: «Застосувати внесені зміни?» (рис. 3.27). Якщо користувач погодився, то усі зміни набувають силу, інакше – повертаються до значення за умовчуванням (-1). Коли всі умови виконані, оператор завантажується на рівень із відповідним типом місцевості (рис. 3.28).

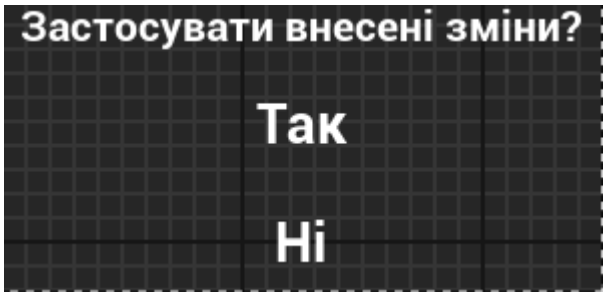

Рисунок 3.27 – Вікно підтвердження внесення змін

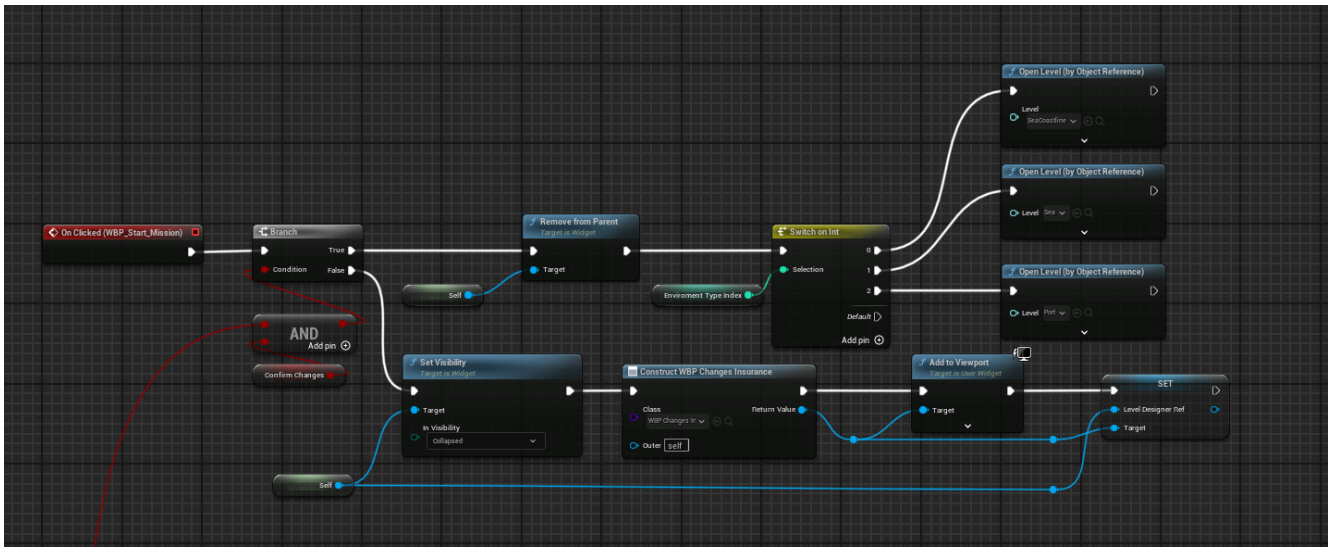

Рисунок 3.28 – Логічна частина WBP_Level_Designer. Основна частина події OnClicked «Продовжити»

Раніше було зазначено, що меню налаштувань надає змогу змінити опції якості відео (рис 3.29), аудіо (рис 3.30) та контролю (3.31), розробимо графічну та логічну частину для цих віджетів.

Рисунок 3.29 – Графічна частина вікна налаштувань відео

Рисунок 3.30 – Графічна частина вікна налаштувань аудіо

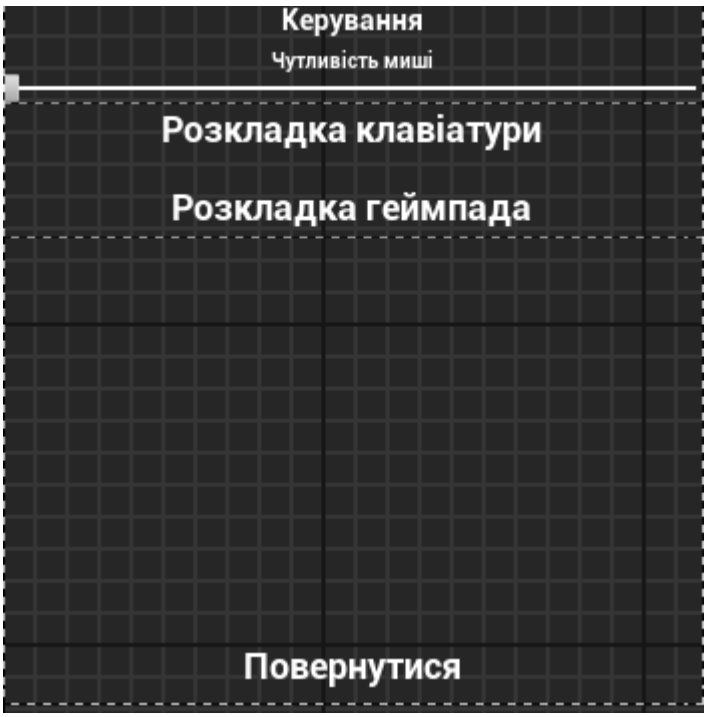

Рисунок 3.31 – Графічна частина вікна налаштувань контролю

Їхні логічні частини показані на рис. 3.32 – 3.39.

Рисунок 3.32 – Логічна частина вікна налаштувань відео. Готовий набір налаштувань

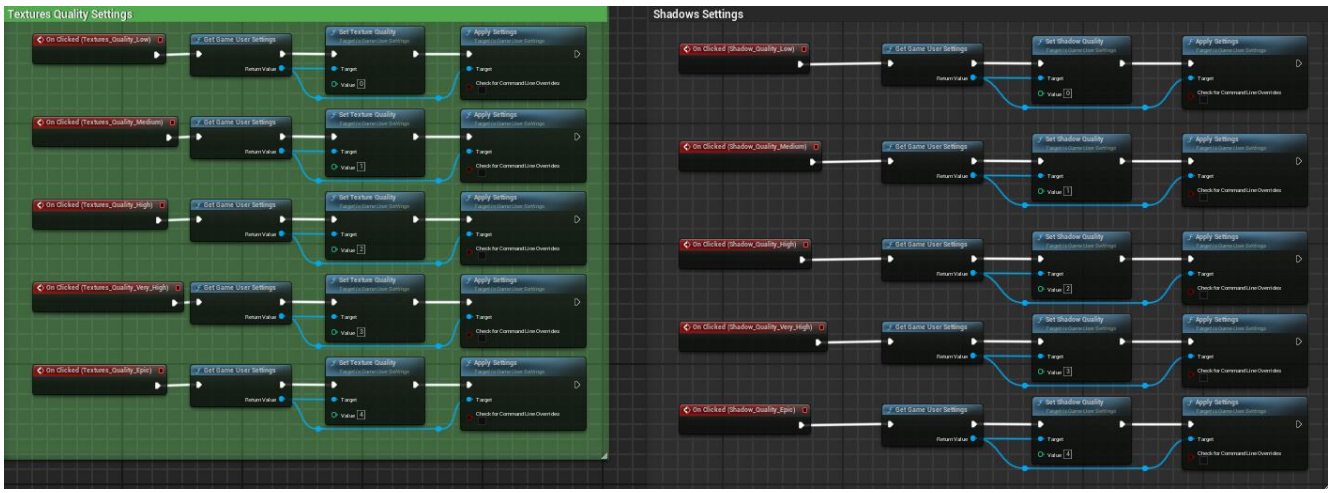

Рисунок 3.33 – Логічна частина вікна налаштувань відео. Варіанти налаштувань якості текстур та тіней

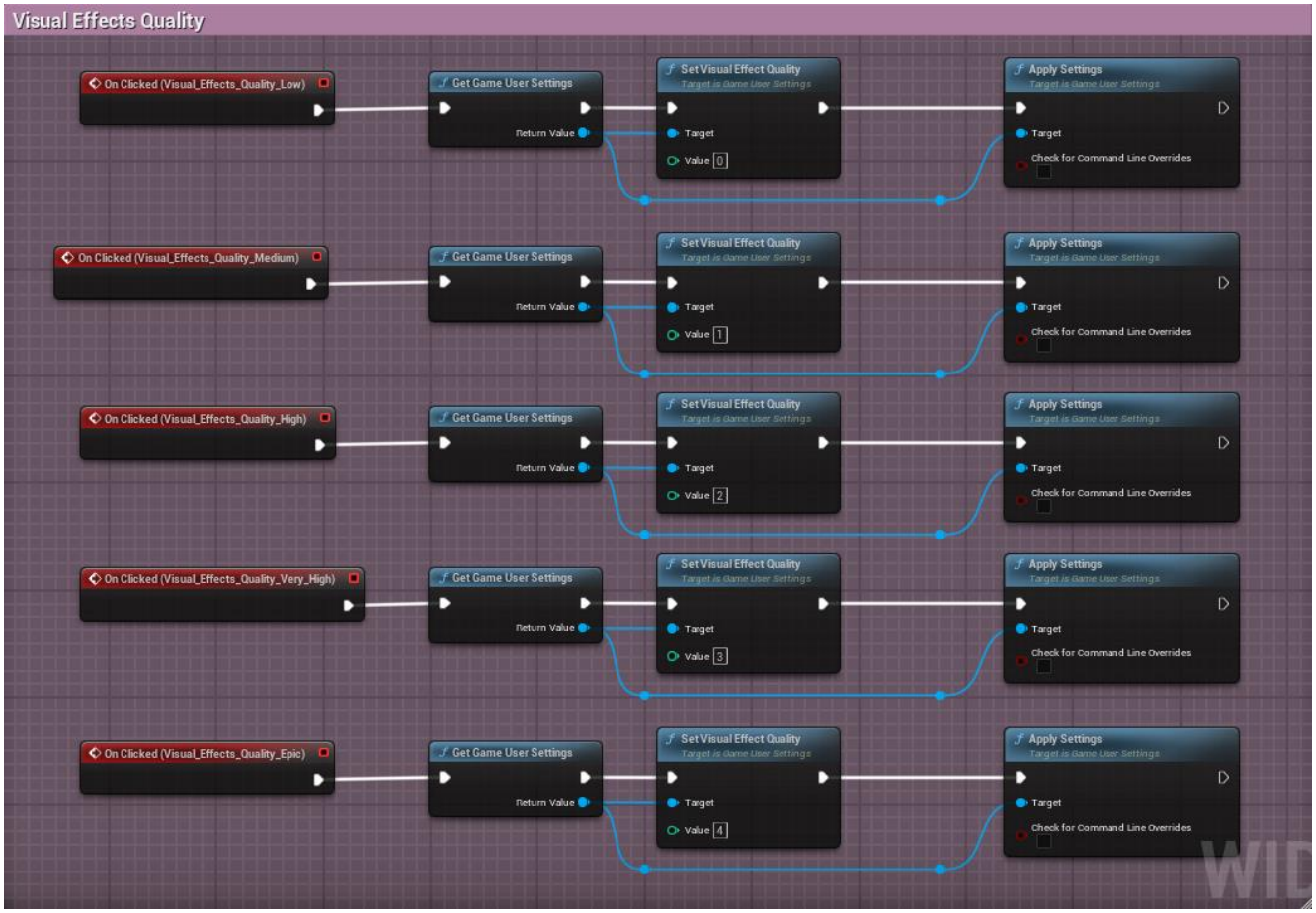

Рисунок 3.34 – Логічна частина вікна налаштувань відео. Варіанти налаштувань якості візуальних ефектів

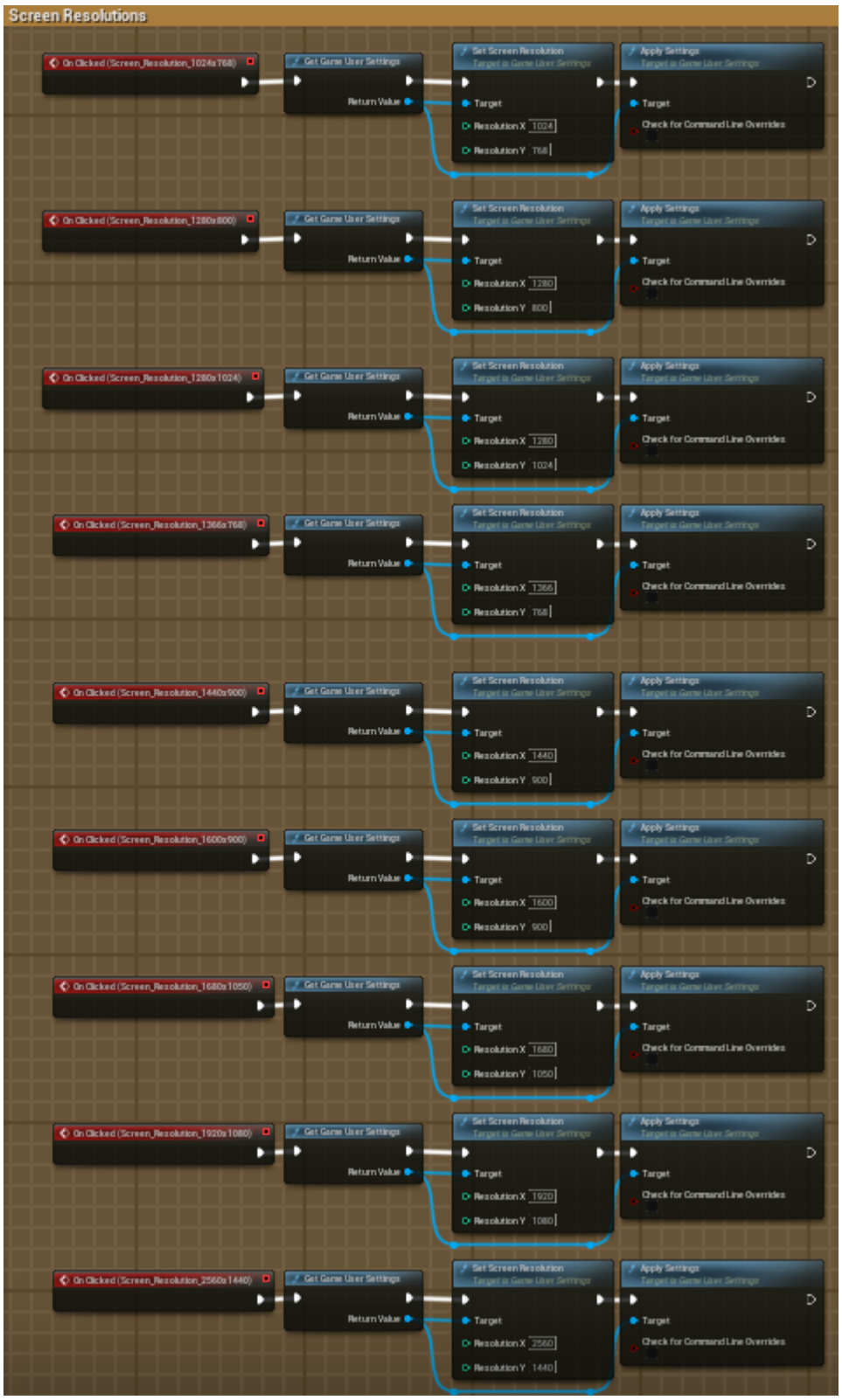

Рисунок 3.35 – Логічна частина вікна налаштувань відео. Варіанти налаштувань розміру вікна

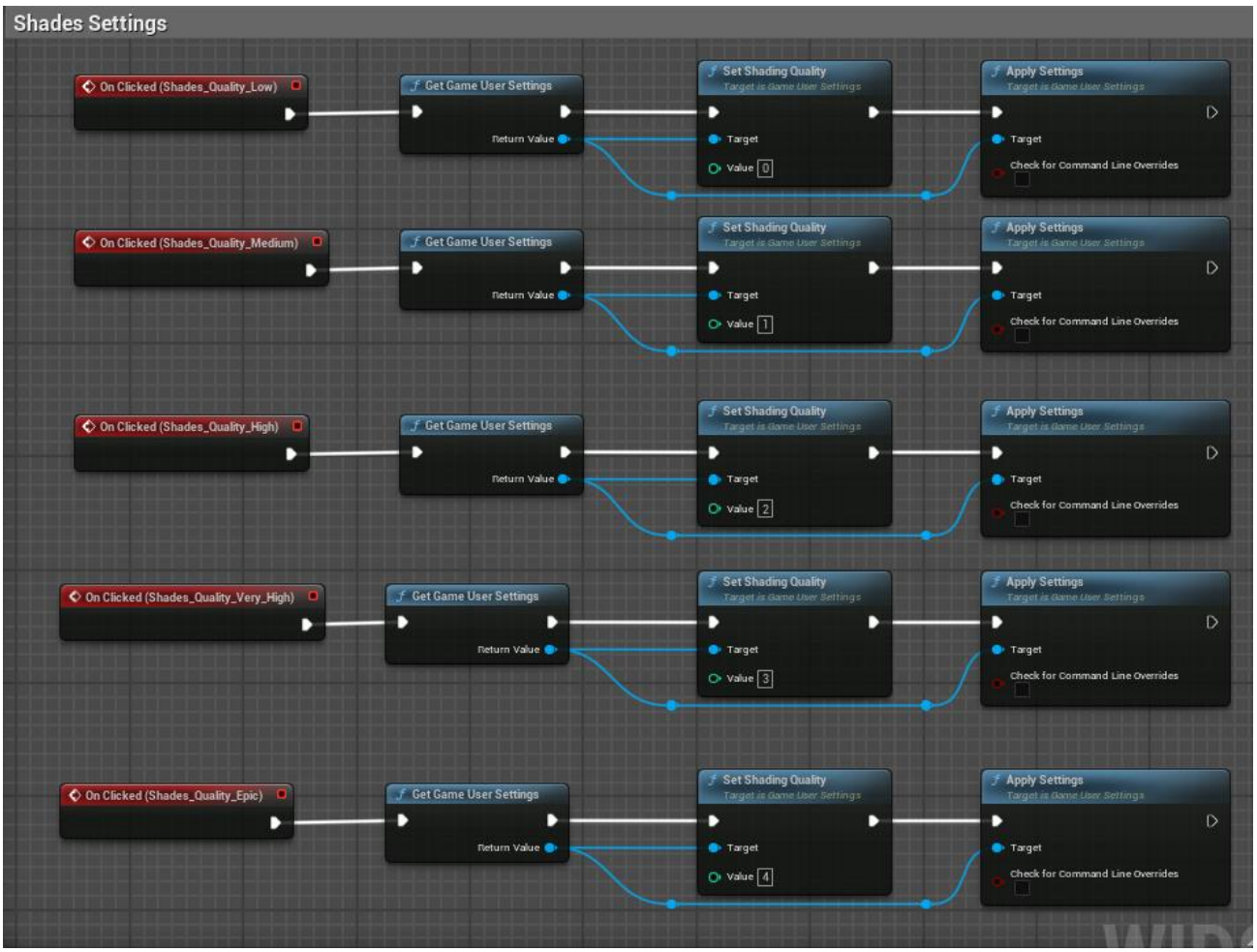

Рисунок 3.36 – Логічна частина вікна налаштувань відео. Варіанти налаштувань відтінків

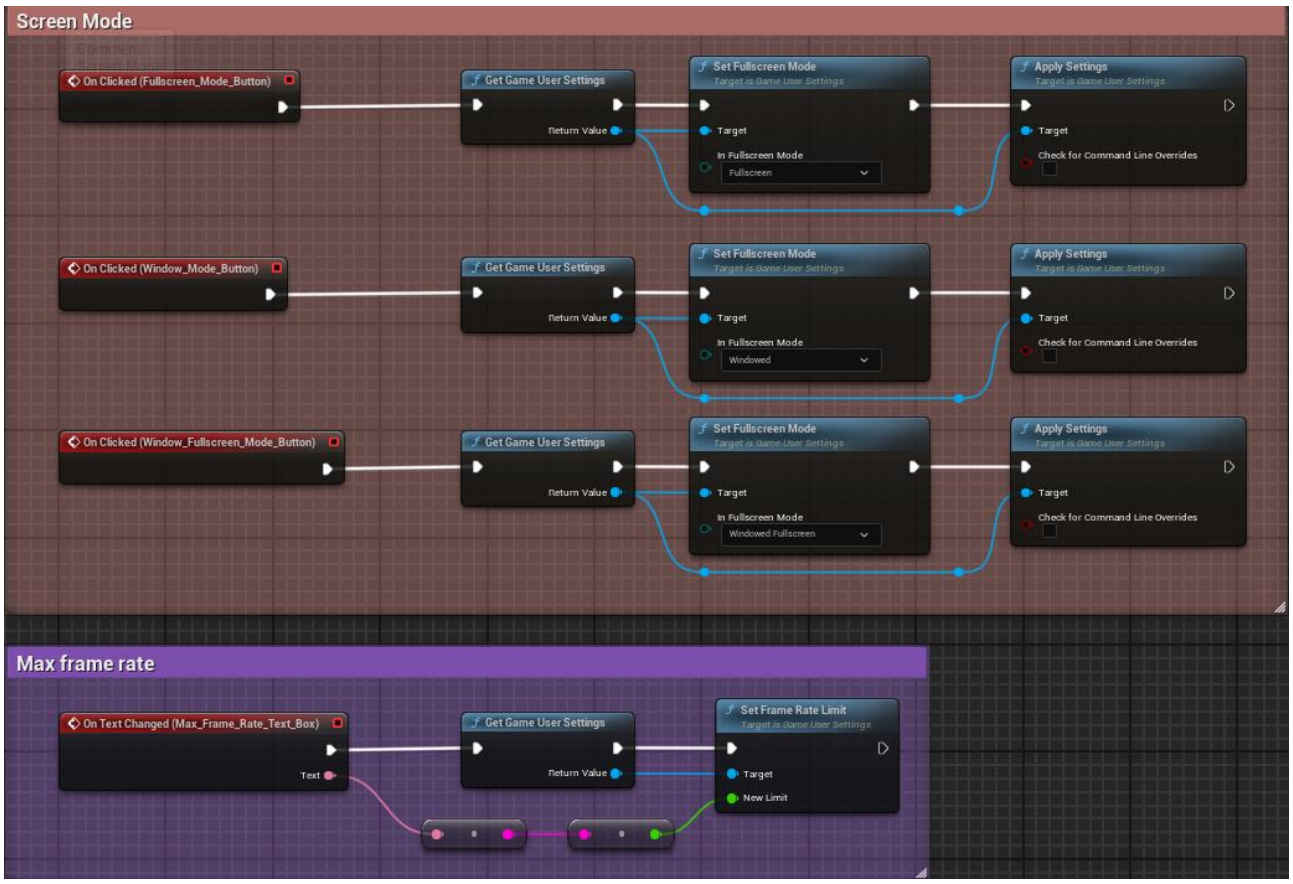

Рисунок 3.37 – Логічна частина вікна налаштувань відео. Варіанти налаштувань максимальної частоти кадрів та режиму вікна

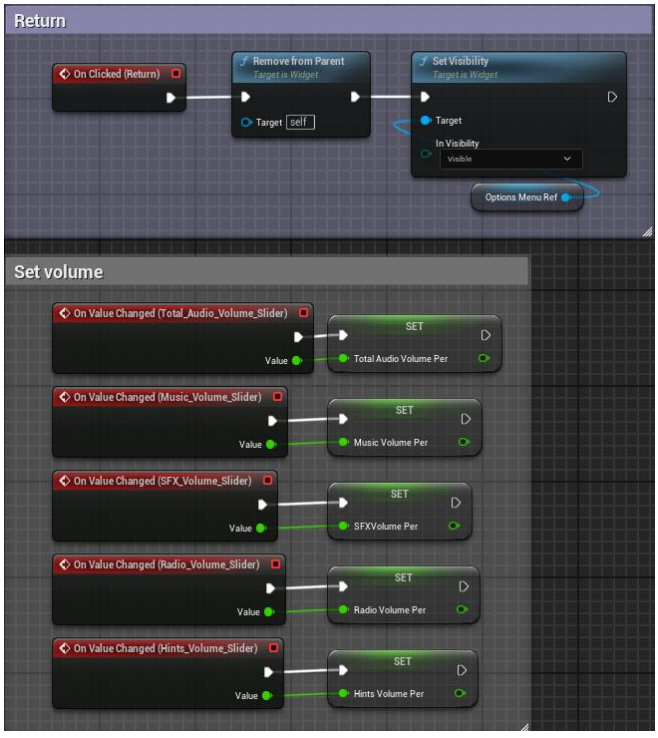

Рисунок 3.38 – Логічна частина вікна налаштувань аудіо

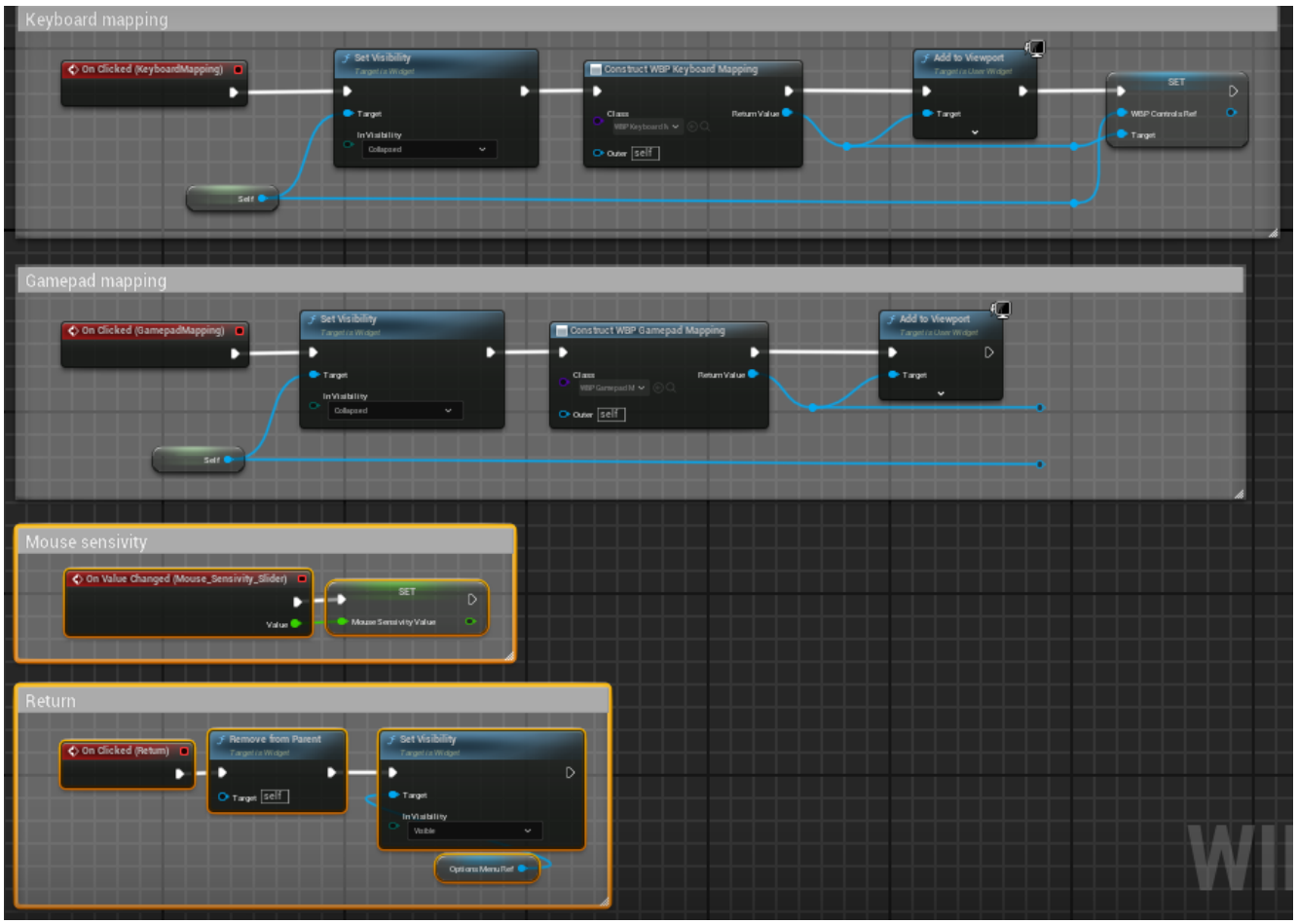

Рисунок 3.39 – Логічна частина вікна налаштувань

Головне меню повністю готове. Наступним кроком розробимо меню паузи та налаштуємо клавішу Escape на її виклик. Меню паузи майже ідентичне з головним меню, щоправда воно призупиняє ігрову сесію та надає можливість продовжити її, натиснувши на кнопку «Продовжити» (рис. 3.40).

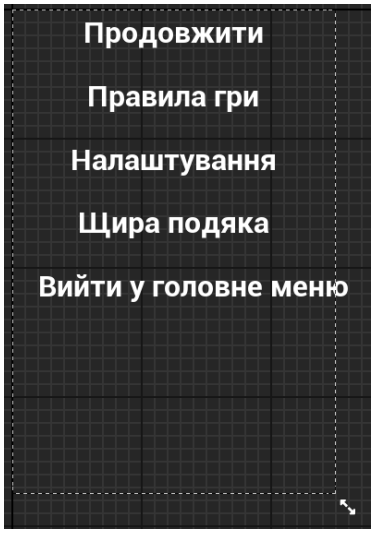

Рисунок 3.40 – Графічна частина меню паузи

Викликаючи «Продовжити», віджет згортається та паузи з гри прибираються. Натиснувши «Правила гри», оператор ознайомлюється з правилами гри, зазначеними у головному меню. Перейшовши до «Налаштування», гравець може змінити опції аудіо, відео та контролю. Клацнувши по «Щира подяка», з'явиться список людей, які тим чи іншим чином підтримали проєкт. Викликавши «Вийти у головне меню», виводиться вікно, яке просить підтвердження на завершення ігрової сесії, якщо користувач погодиться, то ігровий процес повернеться до головного меню.

У приватній секції додано новий приватний елемент PauseMenuWidget для зберігання екземпляра віджета меню паузи та PauseMenuClass для зберігання класу шаблона віджета WBP_PauseMenu. У конструкторі присвоїмо значення nullptr вказівнику. У публічній секції створимо дві функції SetupPlayerInputComponent (буде прив'язувати функціонал до певних клавіш чи їх комбінацій) та TogglePauseMenu(створює та додає віджет меню паузи у вікно перегляду, коли гру поставлено на паузу, і прибирає його, коли гру знято з паузи). Програмний код можна переглянути у Додатку А.

Тепер настав час створити графічний інтерфейс для надводного дрона. Спочатку треба створити віджет DroneUI, який буде містити компоненти, що відповідатимуть за візуальну частину дрона. Враховуючи вимоги до користувальницького інтерфейсу дрона, отримаємо в результаті (рис 3.41).

Щоб використати наявний HUD до гравця, необхідно додати вказівник до нього в приватній секції класу MaguraDrone, також ініціалізувати цей елемент у конструкторі. Режими нічного та тепловізійного бачення, а також тип зв'язку будуть змінюватися при натисканні на відповідну клавішу. У свою чергу рівень палива та дистанція до цілі оновлюються постійно.

Програмний код наведено у додатку А у розділі MaguraV5HUD.

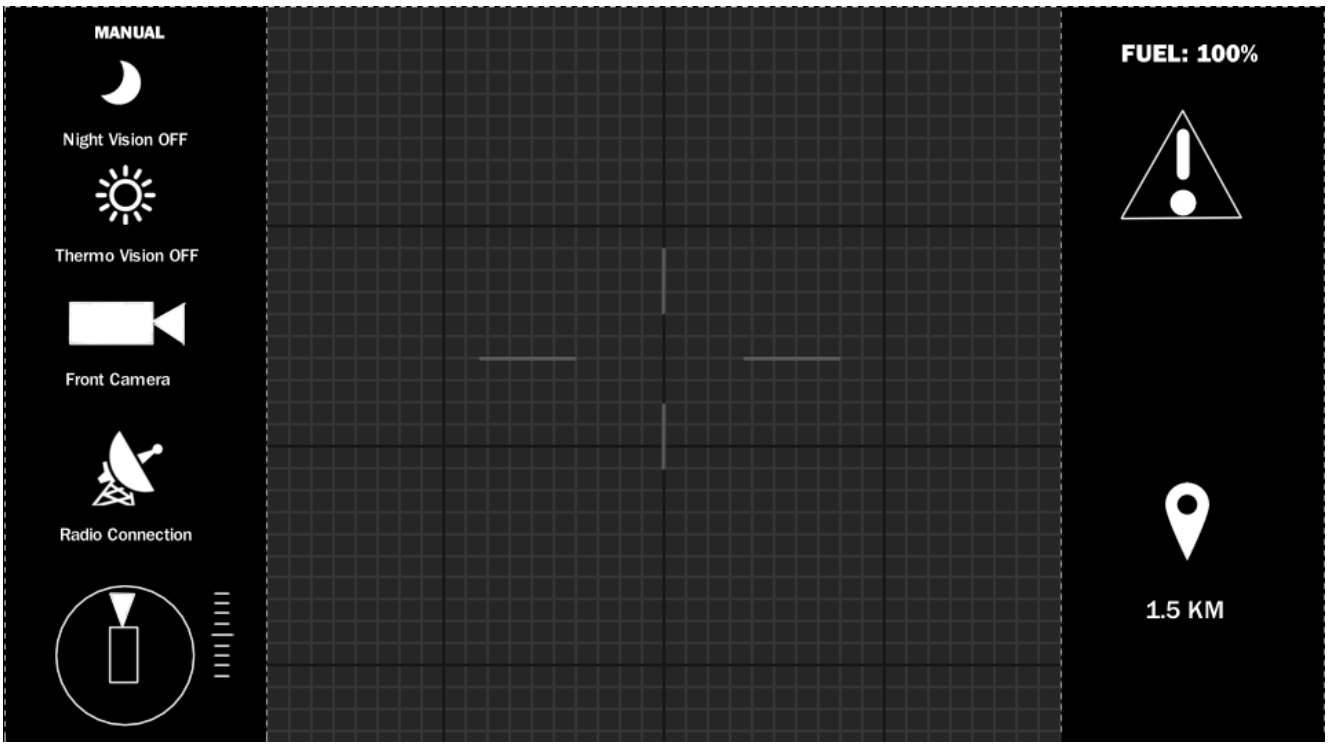

Рисунок 3.41 – Графічна частина WBP_DroneUI

3.2 Розробка режиму гри за вибраними сценарієм та параметрами

У попередньому підрозділі розроблено дизайнер рівнів, в який необхідно ввести дані налаштування ігрового середовища. Якщо для налаштування середовища та параметрів уже зроблений необхідний алгоритм, то тепер, слід розробити механізм використання даних кількості ворогів та пасток. Першочергово створимо класи EnemySpawner (спавнер ворогів) та ObstacleSpawner (спавнер пасток). Будь-які події, пов’язані зі створенням або модифікацією ігрового світу, мають контролюватися з класу типу «Ігровий режим», створимо один із таких для режиму «Ліквідація цілі», назвавши його LiquidationGameMode. Цей клас контролює загальною логікою відтворення, організовуючи, коли і як вороги та перешкоди з'являються в ігровому світі (додаток А).

Тепер на ігровому рівні з’являються вороги (рис. 3.42).

Рисунок 3.42 – Відтворенні вороги на рівні

Додавши механізм відтворення ворогів, необхідно додати й штучний інтелект, який буде керувати ними. Гравець керує своїм дроном за допомогою класу «Controller», у свою чергу ШІ керує персонажем за допомогою «AI controller». Контролер ШІ використовує дерево поведінки, щоб визначити логіку для різних станів ворога (в нашому випадку – «Спокій», «Зацікавлений», «На варті», «Тривога»). Blackboard використовується для зберігання даних для логіки штучного інтелекту, наприклад, чи патрулює ворог, чи дивиться на дрон, чи стріляє. ШІ-контролер відстежує стан ворога і відповідно оновлює значення на Blackboard. Наприклад, якщо ворог перебуває в стані "Спокій", на Blackboard з'являється інформація про те, що ворог повинен патрулювати.

Наш Blackboard містить ключі для зберігання різних станів та даних:

– IsPatrolling. Логічний ключ, який вказує на те, чи патрулює ворог;

– IsLookingAtDrone. Логічний ключ, який вказує, чи дивиться ворог на дрон;

– IsShooting. Булеве значення, що вказує на те, чи ворог стріляє;

– DroneLocation: Вектор, що вказує на місцезнаходження дрона.

Дерево поведінки має структуру з кореневим вузлом, що містить селектор, який оцінює поточний стан ворога.

Для кожного стану визначені різні послідовності:

– Патрулювання. Якщо ворог патрулює, він рухається по заздалегідь визначеному шляху;

– Спостереження за дроном. Якщо ворог зацікавлений у безпілотнику, він зупиняється і повертається обличчям до його місцезнаходження;

– Відкриття вогню: якщо ворог перебуває в стані тривоги, він стріляє в напрямку перебування дрона.

Для прикладу візьмемо програмування поведінки наземних противників:

– Патрулювання. Наземні вороги слідують за заздалегідь визначеним маршрутом патрулювання, коли вони перебувають у стані "Спокій". Це передбачає послідовне переміщення до різних точок патрулювання;

– Стан зацікавленості. Коли ворог перебуває в стані "Зацікавленість", він припиняє патрулювання і дивиться в бік дрона. Це досягається за рахунок повороту противника у напрямку руху дрона;

– Стан «На варті». У стані «На варті» противник патрулює найближчу прибережну зону. Це передбачає переміщення до заздалегідь визначених точок патрулювання поблизу берегової лінії;

– Стан «Тривога». Коли противник перебуває в стані "Тривога", він вступає в бій, відкриваючи вогонь у бік дрона. Кулі не обов’язково можуть летіти точно по напрямку.

Після програшу або перемоги гравцеві пропонується або розпочати місію заново, або вийти до головного меню. Програмний код наведено у додатку А.

3.3 Розробка режиму гри «Тренувальні рівні»

Упродовж усіх тренувальних місій гравця супроводжує аудіо-помічник, який надає вказівки по мірі їх виконання. Є багато варіантів реалізації такого інструктора, щоправда найпростіший, водночас практичний є виведення підказки у вигляді невеличкого вікна з текстом інструкції та фотографією кінцевого результату.

Створимо новий клас Guide, який унаслідується від AActor. Guide використовуватиме властивості класу UTexture2D для зберігання зображень для вікна підказок. На початку гри цей клас виводить перше повідомлення-підказку для знайомства гравця із середовищем. Інші підказки з’являються тільки після

завершення попереднього завдання. Як тільки завдання виконане, підказка зникає та гравець переходить до наступного завдання з відповідною підказкою.

3.4 Тестування та оцінка якості симулятора

Тестування ігор – це процес оцінки і виявлення багів, збоїв та інших помилок у відеоіграх перед тим, як вони будуть випущені для широкого загалу. Основна мета тестування ігор – виявити та повідомити про баги, збої та інші проблеми, які можуть негативно вплинути на ігровий досвід користувача [17].

Тестування симулятора проводилося для перевірки відповідності до нефункціональних та функціональних вимог, а також забезпечення стабільності роботи застосунку. Для досягнення цих цілей використовувалися такі методи тестування:

– модульне тестування. Перевірка окремих компонентів симулятора, включаючи головне меню, ігрові режими, а також окремі функції безпілотних надводних апаратів (БНА);

– інтеграційне тестування. Перевірка взаємодії між різними компонентами симулятора для забезпечення коректної роботи в комплексі;

– тестування продуктивності. Оцінка роботи симулятора за різних налаштувань графіки та навантаженнях для забезпечення стабільної частоти кадрів і швидкого завантаження рівнів.

Для прикладу, модульні тести для класу MaguraDrone є у новоствореному класі MaguraDroneTest, код якого наведено у додатку А. Результати тестування можна подивитися у додатку Б.

Модульне тестування проводилося для кожного з основних компонентів симулятора. Основні його результати:

– головне меню: перевірено функціональність всіх елементів меню, включаючи налаштування графіки, аудіо, керування, а також переходи до різних режимів гри. Усі елементи працюють коректно, зручність використання підтверджена;

– ігрові режими: тестування режимів "Тренувальні рівні" та "Сценарні рівні" продемонструвало їхню працездатність та відповідність заданим сценаріям. Інструкції та підказки для гравців зрозумілі та ефективні;

– функції БНА: усі режими перегляду, системи зв’язку, а також можливості дронів (зближення, віддалення, самознищення тощо) працюють без помилок.

Було також проведено інтеграційне тестування, яке тестувало взаємодію НА із HUD (лицьовим дисплеєм або графічним інтерфейсом надводного дрона). Код класу FMaguraIntegrationTest є у додатку А. Результати інтеграційного тестування можна подивитися у додатку Б.

Інтеграційне тестування показало, що взаємодія між різними компонентами симулятора відбувається коректно. Результатами є:

– синхронізація режимів гри та налаштувань: Зміни налаштувань у головному меню правильно відображаються в ігрових режимах;

– перехід між рівнями: завантаження тренувальних і сценарних рівнів відбувається без збоїв, всі налаштування зберігаються;

– взаємодія гравця з середовищем: усі об’єкти віртуального середовища коректно реагують на дії гравця.

Тестування продуктивності проводилося з метою оцінки програмного забезпечення на відповідність таким нефункціональним вимогам:

– підтримка стабільної частоти кадрів 30 FPS на середніх налаштуваннях графіки;

– час завантаження рівнів не перевищує 120 секунд, час завантаження відеороликів – 30 секунд.

У програмному коді тест на частоту кадрів має назву FProductionTest_FPS, а тест на час завантаження рівнів – FroductionTest_LevelLoadingTime. Їх імплементацію наведено у додатку А. Результати виконання тестування є у додатку Б.

Для задавання параметрів та умов усіх вище описаних тестувань необхідно створити файл FTestsRun. Програмний код є у додатку А.

На завершення слід зазначити, під час виконання програмного проєкту були виконані та протестовані як функціональні, так і нефункціональні вимоги.

Тестування показало, що розроблений симулятор відповідає всім заданим вимогам і є готовим до використання для навчання операторів надводних дронів. Застосунок демонструє стабільну роботу, високу продуктивність і зручність використання, що робить його ефективним інструментом для підготовки майбутніх операторів надводних дронів.

ВИСНОВКИ

Розробка ігрового застосунку-симулятора для навчання майбутніх фахівців з пілотування надводних дронів є успішно виконаною. Під час виконання було:

– проведено детальний аналіз предметної області, включаючи специфіку пілотування надводних дронів та аналіз сучасного стану індустрії;

– визначено нефункціональні та функціональні вимоги до симулятора;

– розроблено дизайн ігрових можливостей дрона, перешкод, пасток та ворогів;

– реалізовано симулятор з використанням сучасних технологій C++ та Unreal Engine, а також системи візуального скриптування Blueprint;

– виконано тестування застосунку та проведено оцінку його продуктивності.

Результати роботи отримані за допомогою:

– аналітичного підходу та комп'ютерних технологій, було визначено ключові аспекти та вимоги до симулятора;

– аналізу наявних симуляторів (Gazebo та The USV Simulator), розроблено власну модель, яка включає найбільш ефективні та релевантні функціональні можливості;

– розробки симулятора, що поєднала створення головного меню, режимів гри, тренувальних рівнів, а також реалізацію алгоритмів керування дроном та його взаємодії з навколишнім середовищем і ворогами;

– тестування застосунку проведено за допомогою фреймворка Unreal Test, що забезпечило високий рівень надійності та продуктивності.

При виконанні роботи досягнуто таких результатів:

– створений симулятор зменшує витрати та ризики при навчанні операторів, оскільки він моделює реальні сценарії;

– ігровий застосунок може надавати можливість військовим кадетам та офіцерам Збройних Сил України та Міністерства Внутрішніх Справ покращувати свої навички в різних сценаріях, наближених до реальних умов;

– симулятор може використовуватися як навчальний інструмент у військових академіях та навчальних центрах, що значно підвищуватиме рівень підготовки фахівців.

Поки програмний застосунок має дещо обмежений потенціал у розвитку, тому рекомендується:

– в подальшому вдосконалювати симулятор шляхом впровадження додаткових сценаріїв та більш складних умов пілотування;

– розширювати функціонал гри для підтримки різних моделей надводних апаратів та розробляти нові типи тренувальних місій;

– інтегрувати симулятор із реальними системами керування та моніторингу для підвищення його ефективності та реалістичності;

– провести порівняльний аналіз ефективності навчання операторів з використання симулятора та традиційних методів тренувань.

З цього слідує, що розробка ігрового застосунку-симулятора для оператора надводного дрона є значним внеском у підвищенні ефективності та безпеки підготовки фахівців, забезпечуючи їм можливість набувати практичних навичок у контрольованому та безпечному середовищі.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

3. Unmanned surface vehicle. Development. URL: https://en.wikipedia.org/wiki/Unmanned_surface_vehicle
4. gazebosim.org. OPEN SOURCE LIBRARIES URL: https://gazebosim.org/home
5. Wikipedia. Gazebo simulator. Competitions. URL: https://en.wikipedia.org/wiki/Gazebo_simulator
6. Задерейко О. В., Толокнов А. А., Струк Н. О. Розробка ігрового застосунку «симулятор оператора надводного дрона». *Сучасні технології в енергетиці, електромеханіці, системах керування та машинобудуванні:* матер. VI Всеукр. наук.-практ. інтернет-конф. (м. Харків, 06-07 грудня 2023 р.) Харків: ННІПІ УІПА, 2023. С. 11-12. URL: https://hdl.handle.net/11300/26945/
7. Catness gam studios. Unreal Engine 5 minimum requirements on Windows. URL: https://catnessgames.com/blog/unreal-engine-5-requirements/
8. Visualasation techniques to support derivation tasks in software product line development. URL: https://www.researchgate.net/publication/221500938_Visulasation_techniques_to_support_derivation_tasks_in_sofware_product_line_development
9. Xbox 360 Controller Input in C++ via XInput. URL: https://www.david-amador.com/2012/04/xbox-360-controller-input-in-c-via-xinput/
10. Бон (техніка). URL: https://uk.wikipedia.org/wiki/Бон_(техніка)
11. Iconduck. Alert sign. URL: https://iconduck.com/icons/146037/alert
12. Корвет. URL: https://uk.wikipedia.org/wiki/Корвет
13. Ukraine's Magura V5 USV – Kamikaze Drone Boat at DSEI 2023. URL: https://youtu.be/9-Zl2_qFjbs?si=dWgRtMFN1HPdb_Kk&t=160
14. Ukraine's Magura V5 USV – Kamikaze Drone Boat at DSEI 2023. URL: https://youtu.be/9-Zl2_qFjbs?si=op777sby0ovnbzJR&t=170
15. Ukraine's Magura V5 USV – Kamikaze Drone Boat at DSEI 2023. URL: https://youtu.be/9-Zl2_qFjbs?si=5lUwV-9ylrV1ozwi&t=163

16. Location Simple PNG Transparent Background. URL: https://www.freeiconspng.com/img/4242
17. Трофименко О.Г., Струк Н.О. Специфіка тестування ігрових застосунків. *Українське суспільство та наука в умовах воєнного стану й євроінтеграційних перетворень: виклики, напрями відновлення і розвитку*: у 2 т. : матер. Всеукр. наук.-практ. конф. (м. Одеса, 20 травня 2023 р.) Одеса : Видавництво «Юридика», 2023. С.470-472. URL: https://dspace.onua.edu.ua/server/api/core/bitstreams/5b59bf6e-927b-4a81-9459-da85a6e6ff05/content

ДОДАТКИ

Додаток А. Фрагменти програмного коду

MaguraV5HUD.h:

```
#pragma once

#include "CoreMinimal.h"
#include "SeaDroneHUD.h"
#include "MaguraV5HUD.generated.h"

/**
 *
 */
UCLASS()
class BACHELORS_PROJECT_API AMaguraV5HUD : public ASeaDroneHUD
{
	GENERATED_BODY()
public:
	AMaguraV5HUD();
	void BeginPlay() override;
	void Tick(float DeltaSeconds) override;
	virtual void DrawHUD() override;
    void ToggleNightVision(bool bEnabled);
    void ToggleThermoVision(bool bEnabled);
    void ToggleFrontCamera(bool bEnabled);
    void ToggleRadioConnection(bool bEnabled);
    void UpdateFuel(float FuelLevel);
    void UpdateDistance(float Distance);
    void ShowAlert(bool bShow);

private:
    UPROPERTY(EditDefaultsOnly, Category = "HUD")
    TSubclassOf<class UUserWidget> HUDWidgetClass;
    UPROPERTY()
    class UUserWidget* DroneHUDWidget;
    UPROPERTY()
    class UProgressBar* FuelBar;
    UPROPERTY()
    class UTextBlock* DistanceText;
    UPROPERTY()
    class UImage* NightVisionIcon;
    UPROPERTY()
    class UImage* ThermoVisionIcon;
    UPROPERTY()
    class UImage* FrontCameraIcon;
    UPROPERTY()
    class UImage* RadioConnectionIcon;
    UPROPERTY()
    class UImage* AlertIcon;
};
```

MaguraV5HUD.cpp

```
#include "MaguraV5HUD.h"
#include "Blueprint/UserWidget.h"
#include "Components/ProgressBar.h"
#include "Components/TextBlock.h"
#include "Components/Image.h"

AMaguraV5HUD::AMaguraV5HUD()
{
}

void AMaguraV5HUD::BeginPlay()
{
      Super::BeginPlay();
    if (HUDWidgetClass)
    {
        DroneHUDWidget = CreateWidget<UUserWidget>(GetWorld(), HUDWidgetClass);
        if (DroneHUDWidget)
        {
            DroneHUDWidget->AddToViewport();
            FuelBar = Cast<UProgressBar>(DroneHUDWidget-
>GetWidgetFromName(TEXT("FuelBar")));
            DistanceText = Cast<UTextBlock>(DroneHUDWidget-
>GetWidgetFromName(TEXT("DistanceText")));
            NightVisionIcon = Cast<UImage>(DroneHUDWidget-
>GetWidgetFromName(TEXT("NightVisionIcon")));
            ThermoVisionIcon = Cast<UImage>(DroneHUDWidget-
>GetWidgetFromName(TEXT("ThermoVisionIcon")));
            FrontCameraIcon = Cast<UImage>(DroneHUDWidget-
>GetWidgetFromName(TEXT("FrontCameraIcon")));
            RadioConnectionIcon = Cast<UImage>(DroneHUDWidget-
>GetWidgetFromName(TEXT("RadioConnectionIcon")));
            AlertIcon = Cast<UImage>(DroneHUDWidget-
>GetWidgetFromName(TEXT("AlertIcon")));
        }
    }
}

void AMaguraV5HUD::Tick(float DeltaSeconds)
{
      Super::Tick(DeltaSeconds);
}
void AMaguraV5HUD::DrawHUD()
{
    Super::DrawHUD();
}
void AMaguraV5HUD::ToggleNightVision(bool bEnabled)
{
    if (NightVisionIcon)
    {
        NightVisionIcon->SetVisibility(bEnabled ? ESlateVisibility::Visible :
ESlateVisibility::Hidden);
    }
}
```

Додаток Б. Результати роботи автоматизованих тестів

Результати модульних автоматизованих тестів функціоналу збережені у ModuleTest.log:

```
[2024.05.15-12.00.00:000][000]LogAutomationTest: Started test:
Game.MaguraDrone.BasicTests
[2024.05.15-12.00.00:100][001]LogAutomationTest: Test
'Game.MaguraDrone.BasicTests' started.

[2024.05.15-12.00.00:200][002]LogAutomationTest: Test description: Verify camera
switching, view modes, movement, and communication mode toggling.
[2024.05.15-12.00.00:300][003]LogAutomationTest:
[2024.05.15-12.00.00:400][004]LogAutomationTest: AMaguraDrone instance created.
[2024.05.15-12.00.00:500][005]LogAutomationTest:

[2024.05.15-12.00.00:600][006]LogAutomationTest: Test: Switch to Front Camera
[2024.05.15-12.00.00:700][007]LogAutomationTest:   Front Camera is active: PASSED

[2024.05.15-12.00.00:800][008]LogAutomationTest: Test: Switch to Rear Camera
[2024.05.15-12.00.00:900][009]LogAutomationTest:   Rear Camera is active: PASSED

[2024.05.15-12.00.01:000][010]LogAutomationTest: Test: Set Normal View
[2024.05.15-12.00.01:100][011]LogAutomationTest:   Thermal View is off: PASSED
[2024.05.15-12.00.01:200][012]LogAutomationTest:   Night View is off: PASSED

[2024.05.15-12.00.01:300][013]LogAutomationTest: Test: Set Thermal View
[2024.05.15-12.00.01:400][014]LogAutomationTest:   Thermal View is on: PASSED
[2024.05.15-12.00.01:500][015]LogAutomationTest:   Night View is off: PASSED

[2024.05.15-12.00.01:600][016]LogAutomationTest: Test: Set Night View
[2024.05.15-12.00.01:700][017]LogAutomationTest:   Thermal View is off: PASSED
[2024.05.15-12.00.01:800][018]LogAutomationTest:   Night View is on: PASSED

[2024.05.15-12.00.01:900][019]LogAutomationTest: Test: Move Forward
[2024.05.15-12.00.02:000][020]LogAutomationTest:   Drone is moving forward:
PASSED

[2024.05.15-12.00.02:100][021]LogAutomationTest: Test: Move Right
[2024.05.15-12.00.02:200][022]LogAutomationTest:   Drone is moving right: PASSED

[2024.05.15-12.00.02:300][023]LogAutomationTest: Test: Set Radio Communication
[2024.05.15-12.00.02:400][024]LogAutomationTest:   Radio Communication is active:
PASSED

[2024.05.15-12.00.02:500][025]LogAutomationTest: Test: Set Satellite
Communication
[2024.05.15-12.00.02:600][026]LogAutomationTest:   Radio Communication is
inactive: PASSED

[2024.05.15-12.00.02:700][027]LogAutomationTest: Test
'Game.MaguraDrone.BasicTests' completed successfully.
[2024.05.15-12.00.02:800][028]LogAutomationTest: Ended test:
Game.MaguraDrone.BasicTests
```

```
[2024.05.15-12.00.02:900][029]LogAutomationTest: Started test:
Game.MaguraDrone.HUDTests
[2024.05.15-12.00.03:000][030]LogAutomationTest: Test 'Game.MaguraDrone.HUDTests'
started.

[2024.05.15-12.00.03:100][031]LogAutomationTest: Test description: Verify HUD
updates for fuel level, distance to target, and alerts.
[2024.05.15-12.00.03:200][032]LogAutomationTest:
[2024.05.15-12.00.03:300][033]LogAutomationTest: AMaguraDrone instance created.
[2024.05.15-12.00.03:400][034]LogAutomationTest: APlayerController instance
created and possessed the drone.
[2024.05.15-12.00.03:500][035]LogAutomationTest:

[2024.05.15-12.00.03:600][036]LogAutomationTest: Test: Update Fuel Level
[2024.05.15-12.00.03:700][037]LogAutomationTest:   Fuel level updated: PASSED

[2024.05.15-12.00.03:800][038]LogAutomationTest: Test: Update Distance to Target
[2024.05.15-12.00.03:900][039]LogAutomationTest:   Distance to target updated:
PASSED

[2024.05.15-12.00.04:000][040]LogAutomationTest: Test: Show Alert
[2024.05.15-12.00.04:100][041]LogAutomationTest:   Alert is visible: PASSED

[2024.05.15-12.00.04:200][042]LogAutomationTest: Test 'Game.MaguraDrone.HUDTests'
completed successfully.
[2024.05.15-12.00.04:300][043]LogAutomationTest: Ended test:
Game.MaguraDrone.HUDTests
```