STRESZCZENIE (JAN WALCZAK)

Zaangażowanie uczniów w szkołach średnich spada, a lekcje w standardowej formule mogą wydawać się nieatrakcyjne. Matematyka jest jednym z trudniejszych przedmiotów w szkole, którego uczniowie uczą się na każdym etapie edukacji. Rośnie więc potrzeba dywersyfikacji sposobów nauczania i urozmaicania zajęć, tak aby stawały się one ciekawsze.

W ramach pracy zaprojektowano i wykonano aplikację matematycznego escape roomu dla uczniów szkół średnich, która miałaby przedstawić im naukę matematyki jako coś interesującego i wartego ich zaangażowania. Projekt obejmuje opracowanie 13 zagadek, z których każda dotyczy innego działu matematyki i porusza zagadnienia obowiązujące w podstawie programowej szkół średnich. Implementacja została wykonana w Laboratorium Zanurzonej Wizualizacji Przestrzennej z wykorzystaniem technologii VR. Głęboka immersja i interaktywność przedstawionego rozwiązania mają pozwolić uczniom na efektywną i przyjemną naukę.

Słowa kluczowe: matematyczny escape room, Laboratorium Zanurzonej Wizualizacji Przestrzennej, CAVE, Unreal Engine, VR, szkoły średnie.

ABSTRACT (JAN WALCZAK)

Fewer and fewer students are engaged in school, and lessons in the standard format may seem unattractive to them. Mathematics is one of the most difficult subjects that students study at every stage of their education. Because of this, the demand for diversifying teaching methods is increasing so that learning becomes more interesting.

As part of an engineering project, a mathematical escape room application for high school students was designed and developed to present mathematics as something fresh and worth their effort. The project consists of 13 different puzzles, each of which introduces a different branch of mathematics and addresses topics covered in the curriculum. The project was implemented in the Immersive 3D Visualization Lab using VR technology. The deep immersion and interactivity of the solution are intended to enable students to study effectively and enjoyably.

Keywords: mathematical escape room, Immersive 3D Visualization Lab, CAVE, Unreal Engine, VR, middle school, high school.

SPIS TREŚCI

WYKAZ WAŻNIEJSZYCH OZNACZEŃ I SKRÓTÓW

Escape room – interaktywna gra, w której gracze rozwiązują zagadki, by „uciec” z zamkniętego pokoju lub przejść do kolejnego etapu.

Laboratorium Zanurzonej Wizualizacji Przestrzennej – specjalne środowisko lub przestrzeń, gdzie łączy się elementy rzeczywistości wirtualnej i fizycznej, często wykorzystywane do edukacji i interaktywnych doświadczeń.

Wirtualna rzeczywistość (VR) – technologia pozwalająca na zanurzenie się w cyfrowym świecie za pomocą gogli i kontrolerów.

1. WSTĘP I CEL PRACY (KONRAD CZARNECKI)

Współczesny rozwój technologii informatycznych i multimedialnych umożliwia tworzenie nowoczesnych, interaktywnych narzędzi wspomagających proces nauczania, które pozwalają uczniom nie tylko na bierne przyswajanie wiedzy, ale również na aktywne uczestnictwo w zajęciach oraz samodzielne rozwiązywanie problemów w angażującym środowisku. W szczególności dynamicznie rozwijające się technologie wirtualnej i rozszerzonej rzeczywistości znajdują coraz szersze zastosowanie nie tylko w przemyśle i rozrywce, lecz również w edukacji na różnych poziomach nauczania. Zastosowanie immersyjnych środowisk w nauczaniu matematyki może znacząco wpłynąć na wzrost zaangażowania uczniów oraz efektywność przyswajania materiału dydaktycznego.

Matematyka, jako jedna z podstawowych dziedzin nauki, często bywa postrzegana przez uczniów szkół średnich jako przedmiot trudny i mało atrakcyjny. Jednym ze sposobów przełamywania tego stereotypu jest wprowadzenie do procesu dydaktycznego elementów gier edukacyjnych, które poprzez zabawę i rywalizację angażują uczniów do aktywnego rozwiązywania problemów. W ostatnich latach coraz większym zainteresowaniem cieszą się tzw. escape roomy — gry logiczne, w których uczestnicy muszą w określonym czasie rozwiązać serię zagadek i łamigłówek, aby „wydostać się” z wirtualnego lub rzeczywistego pomieszczenia. Włączenie tego typu rozwiązań do nauczania matematyki stwarza szansę na połączenie nauki z zabawą oraz zastosowanie wiedzy teoretycznej w praktycznych sytuacjach problemowych.

Przedmiotem niniejszej pracy jest opracowanie i wykonanie aplikacji stanowiącej wirtualny pokój zagadek matematycznych (ang. *escape room*), przeznaczonej do działania w systemie jaskiń rzeczywistości wirtualnej dostępnych w Laboratorium Zanurzonej Wizualizacji Przestrzennej (LZWP). Laboratorium to wyposażone jest w nowoczesne rozwiązania umożliwiające projekcję obrazów w technologii VR na ścianach specjalnie przystosowanych pomieszczeń, dając użytkownikowi wrażenie całkowitego zanurzenia w trójwymiarowym środowisku. Aplikacja powinna umożliwiać uczestnikom interakcję z otoczeniem za pomocą kontrolerów ruchu wykorzystywanych w LZWP.

Celem głównym projektu jest stworzenie interaktywnej aplikacji edukacyjnej zawierającej 13 zagadek matematycznych, odpowiadających poszczególnym działom matematyki nauczanym w szkołach średnich. Działy te obejmują: liczby rzeczywiste, wyrażenia algebraiczne, równania i nierówności, układy równań, funkcje, ciągi, trygonometrię, planimetrię, geometrię analityczną na płaszczyźnie kartezjańskiej, stereometrię, kombinatorykę, rachunek prawdopodobieństwa i statystykę oraz optymalizację i rachunek różniczkowy.

W ramach realizacji projektu przewidziano następujące etapy prac:

1. Zapoznanie się z funkcjonowaniem systemu jaskiń rzeczywistości wirtualnej dostępnych w LZWP oraz metodami programowania aplikacji dla tego typu środowisk.
2. Opracowanie koncepcji i projektów zagadek matematycznych reprezentujących wymienione działy matematyki, z uwzględnieniem ich formy, poziomu trudności oraz możliwych metod interakcji użytkownika z aplikacją.
3. Konsultacja opracowanych projektów zagadek z dydaktykami matematyki w celu dostosowania ich treści oraz sposobu prezentacji do wymagań programowych i poziomu uczniów szkół średnich.

4. Implementacja aplikacji integrującej wszystkie zaprojektowane i zatwierdzone zagadki w ramach jednego spójnego środowiska wirtualnego escape roomu.

5. Przeprowadzenie testów funkcjonalnych, wydajnościowych oraz badań pilotażowych z udziałem grupy uczniów i dydaktyków w celu weryfikacji poprawności działania aplikacji oraz jej efektywności dydaktycznej.

Podział prac przedstawiono w tabeli 1.1.

Tabela 1.1: Podział prac

Imię i nazwisko	**Zakres prac**
Jan Walczak	Opracowanie i implementacja zadań: • 2 (Planimetria) • 7 (Liczby rzeczywiste i działania na zbiorach liczbowych) • 9 (Ciągi liczbowe) • 13 (Stereometria) Koordynacja prac Zarządzanie projektem na GitHubie Scalanie zagadek i integracja z systemem CAVE
Konrad Czarnecki	Opracowanie i implementacja zadań: • 3 (Nierówności) • 4 (Funkcje) • 5 (Geometria analityczna) • 8 (Znaki funkcji trygonometrycznych) Implementacja funkcji obracania pokoju o 90 stopni Scalanie zagadek i integracja z systemem CAVE
Andrii Demyshyn	Opracowanie i implementacja zadań: • 1 (Wzory skróconego mnożenia) • 6 (Kombinatoryka) • 10 (Prawdopodobieństwo) • 11 (Optymalizacja i rachunek różniczkowy) • 12 (Układy równań) Scalanie zagadek i integracja z systemem CAVE

Ostatecznym rezultatem pracy będzie kompletna, działająca aplikacja edukacyjna przeznaczona do uruchomienia w jaskiniach rzeczywistości wirtualnej, umożliwiająca użytkownikom rozwijanie kompetencji matematycznych w angażującej i nowoczesnej formie. Wnioski płynące z przeprowadzonych badań pilotażowych pozwolą na ocenę przydatności tego typu rozwiązań w praktyce dydaktycznej oraz wskazanie możliwości ich dalszego rozwoju i wykorzystania w nauczaniu przedmiotów ścisłych.

2. WPROWADZENIE DO DZIEDZINY (KONRAD CZARNECKI)

2.1. Rola technologii w edukacji

Rozwój technologii cyfrowych wywarł ogromny wpływ na niemal wszystkie obszary współczesnego życia, w tym również na edukację. Tradycyjne metody nauczania, oparte w dużej mierze na wykładzie i pracy z podręcznikiem, są coraz częściej wzbogacane lub nawet zastępowane przez nowoczesne narzędzia dydaktyczne, które wspomagają i uatrakcyjniają proces kształcenia. W szczególności technologie informacyjno-komunikacyjne oraz środowiska immersyjne, takie jak rzeczywistość wirtualna, rzeczywistość rozszerzona czy rzeczywistość mieszana, stają się istotnym elementem nowoczesnej edukacji.

Współczesne pokolenia uczniów i studentów od najmłodszych lat funkcjonują w środowisku przesyconym technologią i są przyzwyczajeni do interaktywnego oraz multimedialnego przekazu treści. Tradycyjny model nauczania nie zawsze jest w stanie zaspokoić potrzeby poznawcze młodych ludzi, co może prowadzić do spadku motywacji oraz trudności w przyswajaniu wiedzy. Odpowiedzią na to wyzwanie jest integracja narzędzi technologicznych z procesem dydaktycznym, co może przyczynić się do zwiększenia efektywności nauczania, ułatwienia zrozumienia trudnych zagadnień oraz podniesienia ogólnego poziomu zaangażowania uczniów.

Zastosowanie nowoczesnych technologii w edukacji nie tylko wpływa na sposób przekazywania wiedzy, ale również umożliwia tworzenie zupełnie nowych, interaktywnych form kształcenia. Szczególnie istotne znaczenie mają tutaj rozwiązania wykorzystujące elementy gamifikacji oraz immersyjnych doświadczeń edukacyjnych, które pozwalają uczniom wchodzić w bezpośrednią interakcję z materiałem dydaktycznym. W kontekście nauczania matematyki, będącej często postrzeganą jako przedmiot trudny i abstrakcyjny, technologie te mogą odegrać kluczową rolę w budowaniu pozytywnego nastawienia do nauki oraz lepszego zrozumienia prezentowanych treści.

2.1.1. Nowoczesne metody nauczania matematyki

Matematyka, jako dziedzina wymagająca myślenia analitycznego, logicznego rozumowania oraz umiejętności abstrakcyjnego operowania symbolami, od zawsze stawiała przed uczniami szczególne wyzwania. Z tego powodu nauczanie matematyki wymaga nieustannego poszukiwania skutecznych metod dydaktycznych, które nie tylko umożliwią efektywne przekazanie wiedzy, ale również wzbudzą w uczniach zainteresowanie i motywację do nauki.

Współczesne podejścia do nauczania matematyki coraz częściej odchodzą od modelu opartego wyłącznie na wykładzie i ćwiczeniach przy tablicy, na rzecz metod aktywizujących, w których uczniowie samodzielnie odkrywają zależności matematyczne, rozwiązują problemy oraz współpracują w grupie. Szczególne miejsce wśród nowoczesnych metod zajmują rozwiązania oparte na technologiach komputerowych — aplikacje i platformy edukacyjne, programy do wizualizacji danych matematycznych, a także symulacje i środowiska interaktywne.

Zastosowanie narzędzi interaktywnych pozwala uczniom na dynamiczne eksplorowanie pojęć matematycznych, eksperymentowanie z danymi i obserwowanie skutków wprowadzanych zmian w czasie rzeczywistym. Programy takie jak GeoGebra, Desmos, czy MATLAB wspierają nauczanie geometrii, analizy matematycznej i algebry w sposób znacznie bardziej przystępny i angażujący niż tradycyjne metody.

Również technologie immersyjne, takie jak VR, zyskują coraz większe znaczenie w edukacji matematycznej. Umożliwiają one prezentację skomplikowanych struktur geometrycznych w przestrzeni trójwymiarowej, co szczególnie sprzyja nauczaniu stereometrii czy geometrii analitycznej. Dzięki temu uczniowie mogą lepiej zrozumieć zależności przestrzenne oraz intuicyjnie postrzegać abstrakcyjne pojęcia matematyczne.

2.1.2. Gamifikacja w edukacji

Gamifikacja (ang. *gamification*) to zastosowanie mechanizmów znanych z gier komputerowych i planszowych w kontekście niezwiązanym bezpośrednio z grami, takim jak edukacja, zarządzanie czy marketing. W praktyce oznacza to wprowadzanie elementów, takich jak punkty, poziomy trudności, nagrody, rankingi czy wyzwania do tradycyjnych zadań edukacyjnych, co ma na celu zwiększenie motywacji, zaangażowania i satysfakcji uczestników procesu nauczania.

W edukacji gamifikacja znajduje szerokie zastosowanie, zwłaszcza w pracy z uczniami szkół podstawowych i średnich. Wprowadzenie grywalizacji do lekcji pozwala uczniom uczestniczyć w nauce w sposób bardziej aktywny i emocjonalnie zaangażowany. Zamiast biernego słuchania wykładu czy rozwiązywania zadań z podręcznika, uczniowie mogą wcielać się w bohaterów gier, zdobywać osiągnięcia i rywalizować z rówieśnikami w przyjazny sposób.

W kontekście nauczania matematyki, gamifikacja może znacząco ułatwić przyswajanie skomplikowanych treści. Zagadki logiczne, łamigłówki, quizy punktowane czy interaktywne escape roomy są przykładami narzędzi, które pozwalają uczniom na wykorzystanie wiedzy matematycznej w kontekście gry. Dzięki temu uczniowie nie tylko uczą się rozwiązywać konkretne typy zadań, ale także rozwijają umiejętności analitycznego myślenia, pracy zespołowej oraz podejmowania decyzji.

W szczególności w środowiskach immersyjnych, takich jak wirtualna rzeczywistość, gamifikacja osiąga nowy wymiar. Połączenie mechanizmów gry z wciągającym, interaktywnym środowiskiem pozwala użytkownikom na budowanie trwałych, pozytywnych skojarzeń z procesem nauki. Tego typu rozwiązania nie tylko zwiększają atrakcyjność zajęć, ale również mogą pozytywnie wpływać na wyniki edukacyjne i długofalowe postawy wobec nauki matematyki.

2.2. Edukacyjne zastosowania escape roomów

W ostatnich latach obserwuje się dynamiczny rozwój innowacyjnych metod dydaktycznych, które mają na celu zwiększenie zaangażowania uczniów oraz poprawę efektywności procesu nauczania. Jedną z takich metod są escape roomy, które pierwotnie funkcjonowały jako forma rozrywki, a z czasem zyskały również uznanie w środowisku edukacyjnym. Dzięki swojej interaktywnej i zespołowej formie, pokoje zagadek umożliwiają uczniom zdobywanie wiedzy oraz rozwijanie kompetencji miękkich w sposób atrakcyjny i emocjonujący.

Escape roomy w edukacji mogą przybierać różnorodne formy — od tradycyjnych wersji stacjonarnych, przez mobilne zestawy dydaktyczne, aż po aplikacje komputerowe i środowiska wirtualnej rzeczywistości. Ich celem jest nie tylko przekazywanie wiedzy merytorycznej, lecz także kształtowanie umiejętności pracy zespołowej, logicznego myślenia, zarządzania czasem oraz podejmowania decyzji pod presją. Dlatego coraz częściej wykorzystywane są one w szkołach, na uczelniach wyższych oraz podczas szkoleń i warsztatów.

W kontekście nauczania matematyki escape roomy stanowią wyjątkowo wartościowe narzędzie. Zagadki oparte na treściach matematycznych wymagają od uczestników nie tylko opanowania materiału, ale również zastosowania wiedzy w praktyce, co sprzyja utrwalaniu wiadomości oraz rozwijaniu umiejętności rozwiązywania problemów. Połączenie elementów gry z edukacją umożliwia uczniom naukę w przyjaznej atmosferze i sprzyja budowaniu pozytywnego nastawienia do przedmiotów ścisłych.

2.2.1. Historia i definicja escape roomu

Escape room, znany również jako pokój zagadek lub gra typu „ucieczka z pokoju”, to interaktywna forma rozrywki polegająca na rozwiązaniu serii łamigłówek i zadań logicznych w określonym czasie, aby wydostać się z zamkniętego pomieszczenia lub osiągnąć inny wyznaczony cel fabularny.

Pierwszy fizyczny escape room powstał w 2007 roku w Kioto w Japonii, a jego twórcą był Takao Kato, który postanowił przenieść ideę wirtualnej gry do świata rzeczywistego. Projekt szybko zyskał popularność, a w kolejnych latach podobne pokoje zaczęły powstawać w innych krajach azjatyckich, a następnie w Europie i Ameryce Północnej. Do Polski pierwsze escape roomy dotarły w 2014 roku i od tego czasu cieszą się dużym zainteresowaniem zarówno wśród młodzieży, jak i dorosłych.

Z czasem, obok komercyjnych pokoi zagadek, zaczęły pojawiać się również wersje edukacyjne, dostosowane do potrzeb szkół i uczelni. Edukacyjne escape roomy różnią się od wersji rozrywkowych tym, że ich głównym celem nie jest zabawa, lecz przekazanie wiedzy oraz rozwijanie określonych kompetencji. Zagadki w tego typu pokojach są projektowane w taki sposób, aby uczestnicy mogli przyswajać treści z wybranych dziedzin nauki podczas rozwiązywania interaktywnych zadań. Coraz częściej wykorzystywane są one także w środowiskach cyfrowych oraz w wirtualnej rzeczywistości, co dodatkowo zwiększa ich dostępność i atrakcyjność.

2.2.2. Escape roomy jako metoda aktywizacji uczniów

Jednym z największych wyzwań współczesnej edukacji jest utrzymanie wysokiego poziomu zaangażowania uczniów oraz motywowanie ich do aktywnego uczestnictwa w zajęciach. W tym kontekście escape roomy stanowią skuteczną metodę dydaktyczną, która pozwala na połączenie nauki z emocjonującą formą zabawy. Dzięki swojej interaktywnej strukturze i zespołowemu charakterowi, pokoje zagadek mobilizują uczestników do współpracy, komunikacji oraz kreatywnego rozwiązywania problemów.

Z punktu widzenia dydaktyki escape roomy wspierają rozwój kompetencji kluczowych, takich jak logiczne myślenie, umiejętność analizowania i syntetyzowania informacji, podejmowanie decyzji pod presją czasu oraz zarządzanie zadaniami w zespole. Dodatkowo umożliwiają uczniom zastosowanie zdobytej wiedzy teoretycznej w praktyce, co znacząco wpływa na trwałość zapamiętywania materiału i lepsze zrozumienie omawianych treści.

W szczególności istotne znaczenie mają escape roomy realizowane w środowisku rzeczywistości wirtualnej. Dzięki immersyjnemu charakterowi takich aplikacji uczestnicy mogą zanurzyć się w wirtualnym świecie, w którym zagadki matematyczne są częścią spójnej fabuły. Tego typu doświadczenie nie tylko zwiększa atrakcyjność nauki, ale również wzmacnia motywację wewnętrzną uczniów, którzy postrzegają naukę jako przygodę i wyzwanie, a nie obowiązek.

2.3. *Wirtualna rzeczywistość*

Współczesna edukacja coraz śmielej sięga po nowoczesne technologie, które pozwalają nie tylko wzbogacić proces nauczania, ale również znacząco zwiększyć zaangażowanie uczniów. Jednym z najbardziej dynamicznie rozwijających się obszarów w tym zakresie są technologie immersyjne, do których zalicza się wirtualną rzeczywistość, rzeczywistość rozszerzoną oraz rzeczywistość mieszaną. Ich wspólną cechą jest zdolność do tworzenia interaktywnych środowisk, w których użytkownik ma wrażenie pełnego zanurzenia w generowanym komputerowo świecie.

W kontekście edukacyjnym szczególnie istotne jest połączenie immersji z interaktywnością, czyli możliwością bezpośredniego wpływania na elementy wirtualnego środowiska. Interaktywne aplikacje VR sprzyjają aktywnej nauce poprzez angażowanie użytkownika w proces rozwiązywania problemów, podejmowania decyzji i wykonywania zadań w czasie rzeczywistym. Taka forma edukacji nie tylko zwiększa efektywność przyswajania wiedzy, ale również pozytywnie wpływa na motywację i postawy uczniów wobec nauki.

2.3.1. Wirtualna rzeczywistość i jej zastosowanie w nauce

Wirtualna rzeczywistość to technologia umożliwiająca tworzenie komputerowo generowanych środowisk trójwymiarowych, z którymi użytkownik może wchodzić w interakcję w czasie rzeczywistym. Dzięki zastosowaniu gogli VR, w szczególności jaskiń VR, a także kontrolerów ruchu możliwe jest odwzorowanie ruchów użytkownika w przestrzeni wirtualnej, co pozwala na pełne zanurzenie w symulowanym środowisku. Technologie VR znajdują zastosowanie nie tylko w rozrywce, ale również w przemyśle, medycynie, wojsku oraz, coraz częściej, w edukacji.

W środowisku edukacyjnym wirtualna rzeczywistość oferuje szerokie możliwości w zakresie tworzenia interaktywnych laboratoriów, symulacji zjawisk fizycznych, rekonstrukcji historycznych czy wirtualnych wycieczek. Uczniowie mogą dzięki niej eksplorować trudno dostępne miejsca, takie jak wnętrze ludzkiego organizmu, przestrzeń kosmiczną, odległe zakątki świata lub historyczne budowle, co znacząco wzbogaca tradycyjny proces dydaktyczny.

W przypadku dydaktyki matematyki VR pozwala na wizualizację skomplikowanych zagadnień geometrycznych, przestrzennych oraz analitycznych w atrakcyjnej i przystępnej formie. Uczniowie mogą w wirtualnym środowisku manipulować bryłami, obserwować zmiany funkcji w czasie rzeczywistym czy uczestniczyć w interaktywnych grach logicznych i escape roomach matematycznych. Dzięki temu abstrakcyjne treści stają się bardziej zrozumiałe i łatwiejsze do przyswojenia, co przekłada się na lepsze wyniki w nauce oraz pozytywnie wpływa na rozwój umiejętności analitycznego myślenia.

2.3.2. Wpływ interaktywności na zaangażowanie gracza

Interaktywność stanowi jeden z kluczowych elementów nowoczesnych metod dydaktycznych, w tym również aplikacji wykorzystujących technologię wirtualnej rzeczywistości. Oznacza ona możliwość aktywnego uczestniczenia ucznia w procesie dydaktycznym poprzez bezpośrednie oddziaływanie na środowisko edukacyjne, podejmowanie decyzji oraz realizację zadań w czasie rzeczywistym. W przeciwieństwie do pasywnego odbioru treści w tradycyjnych formach nauczania, interaktywne środowiska angażują uczniów na wielu płaszczyznach — poznawczej, emocjonalnej i motorycznej.

Z perspektywy pedagogicznej interaktywność sprzyja rozwijaniu kompetencji poznawczych, takich jak krytyczne myślenie, umiejętność analizy danych, wyciągania wniosków czy rozwiązywania problemów. Uczniowie uczestniczący w interaktywnych lekcjach częściej angażują się w zadania, są bardziej zmotywowani do podejmowania wyzwań i wykazują większą samodzielność w poszukiwaniu rozwiązań.

W przypadku zastosowań wirtualnej rzeczywistości, interaktywność przybiera różnorodne formy — od prostych gestów i ruchów wykonywanych za pomocą kontrolerów, przez manipulowanie obiektami w przestrzeni wirtualnej, aż po rozwiązywanie zagadek i wykonywanie eksperymentów. Szczególnie efektywne okazują się aplikacje edukacyjne, które łączą elementy gry z nauką, umożliwiając użytkownikom rywalizację, zdobywanie punktów, odblokowywanie kolejnych poziomów czy rozwiązywanie zagadek fabularnych.

Środowiska edukacyjne o wysokim stopniu interaktywności znacząco zwiększają motywację wewnętrzną oraz podnoszą poziom satysfakcji z nauki. Uczniowie mają również większą łatwość w przyswajaniu wiedzy oraz chętniej uczestniczą w zajęciach, co pozytywnie wpływa na ogólne efekty dydaktyczne.

W kontekście projektowanej aplikacji typu escape room dla jaskini rzeczywistości wirtualnej, wysoki poziom interaktywności będzie kluczowym elementem wpływającym na atrakcyjność i skuteczność dydaktyczną opracowanego rozwiązania. Możliwość bezpośredniego wpływania na otoczenie, rozwiązywania zagadek matematycznych w przestrzeni wirtualnej oraz współpracy z innymi uczestnikami w czasie rzeczywistym stworzy warunki sprzyjające aktywnej, angażującej nauce i rozwijaniu kompetencji.

3. ANALIZA ISTNIEJĄCYCH ROZWIĄZAŃ (JAN WALCZAK)

3.1. *„Gravity Sketch“*

3.1.1. Implementacja rozwiązania

Badanie, przeprowadzone na uczelni Tecnologico de Monterrey, miało charakter eksperymentalny i było przeprowadzone na studentach pierwszego roku [1]. Opierało się o zainstalowanie oprogramowania „Gravity Sketch“ na urządzeniu VR. Narzędzie służy w praktyce do trójwymiarowego projektowania różnych obiektów w wirtualnej przestrzeni. Aspektem, który wyróżnia oprogramowanie „Gravity Sketch“, jest umożliwienie rysowania obiektów opierających się o siatkę współrzędnych, a co za tym idzie, użyto go do wizualizacji wektorów w przestrzeni trójwymiarowej [1].

Możliwość samodzielnego rysowania wektorów powinna ułatwić studentom wizualizację omawianego zagadnienia przez responsywność narzędzia i możliwość zmiany perspektywy (przez ruch użytkownika).

3.1.2. Przeprowadzone badania

Głównym celem badania było sprawdzenie jak technologia VR pomaga w nauce abstrakcyjnych pojęć matematycznych, które wymagają od ucznia wyobraźni przestrzennej. Uczestnikami było 94 studentów, którzy zostali podzieleni na dwie grupy: kontrolną, która uczestniczyła w standardowych zajęciach i eksperymentalną, która brała udział w zajęciach z użyciem technologii VR [1].

Obie grupy rozwiązywały dwa testy: przed rozpoczęciem i po zakończeniu zajęć. Obejmowały one zagadnienia związane z wektorami i dotykały problemów powiązanych z ich wizualizacją, takich jak obliczanie kątów. Dodatkowo grupa eksperymentalna wypełniała ankietę zawierającą pytania o ich doznania przy podczas używania wirtualnej rzeczywistości [1].

3.1.3. Wyniki

W badaniu opracowanym przez Esmeraldę Campos, Irvinga Hidrogo i Genaro Zavala w [1] możemy przeczytać: *„We found that on those items in which the visualization was important, students in the experimental group, i.e., using VR, did better than those who did not use VR. We have evidence that VR can help students visualize angles and components that help them solve problems better“* (Stwierdziliśmy, że w zadaniach, w których wizualizacja odgrywała istotną rolę, uczniowie z grupy eksperymentalnej, czyli korzystający z VR, osiągali lepsze wyniki niż ci, którzy z VR nie korzystali. Mamy dowody na to, że VR może pomóc uczniom w wizualizowaniu kątów i składowych, co umożliwia im skuteczniejsze rozwiązywanie problemów). Warto zauważyć, że w pytaniach dotyczących wizualizacji, grupa, która korzystała z VR uzyskała znacznie lepszy przyrost wiedzy niż grupa kontrolna [1].

Można więc założyć, że zagadnienia wymagające od ucznia wyobraźni przestrzennej mogą wymagać dodatkowego wsparcia przez rozszerzenie standardowej formuły lekcji o aplikację opartą o technologię VR.

3.2. *Wirtualny pokój zagadek z zakresu matematyki*

3.2.1. Projekt i implementacja

Wirtualny pokój zagadek z zakresu matematyki został dwa lata temu opracowany w LZWP i opisany w „Zeszytach Naukowych Wydziału Elektroniki i Automatyki Politechniki Gdańskiej“ [2].

Opracowano innowacyjny projekt edukacyjny, polegający na stworzeniu matematycznego pokoju zagadek (escape room) na poziomie studiów inżynierskich. Jak możemy przeczytać we wskazanym artykule, celem pracy było stworzenie aplikacji na platformie Unity, która miałaby działać w Laboratorium Zanurzonej Wizualizacji Przestrzennej i uatrakcyjnić studentom naukę matematyki [2].

Escape room został podzielony na 4 pokoje, z czego pierwszy z nich był pokojem wprowadzającym, prowadził on do 3 kolejnych pokoi zawierających łącznie 13 zagadek. Pokoje były podzielone nie tylko ze względu na tematykę zagadnień, ale także ze względu na wystrój [2]. Podział ten wyglądał następująco:

1. Pokój w stylu nowoczesnym: zawierał zadania z zakresu m.in. wyznaczników macierzy (metoda Sarrusa), równań płaszczyzny oraz wykresów funkcji.
2. Pokój warsztatowy: skupiał się na ciągach liczbowych (np. ciągu Fibonacciego), liczbach zespolonych, schemacie Hornera oraz systemie binarnym.
3. Pokój w stylu egipskim: oferował zadania dotyczące działań na liczbach zespolonych oraz układów równań liniowych rozwiązywanych metodą Gaussa-Jordana.

3.2.2. Przeprowadzone badania

Przy użyciu wcześniej omawianego rozwiązania został przeprowadzony eksperyment obejmujący grupę 54 studentów. Badanie wykazało, że nauka poprzez rozwiązywanie zagadek w matematycznym escape roomie przynosi wymierne korzyści [2].

Badanie zostało rozszerzone i opisane w artykule „Educational values of a virtual escape room in mathematics“ [3]. Badanie polegało na podzieleniu studentów na 2 równe grupy, na których zostały zastosowane dwa oddzielne podejścia przeprowadzane w różnej kolejności. Pierwsze podejście polegało na uczestnictwie grupy w tradycyjnej lekcji matematyki, a drugie na przejściu matematycznego escape roomu w małych zespołach. Grupy brały udział w sesjach opartych o oba podejścia, z czego jedna najpierw uczestniczyła w tradycyjnej lekcji, a druga najpierw uczestniczyła w sesji w escape roomie.

Wyniki zaprezentowane w artykule wskazują, że studenci uczestniczący w sesjach odbywających się w wirtualnym escape roomie wykazują się większym skupieniem i lepszym samopoczuciem. Największy wzrost wiedzy odnotowano w grupie, która najpierw uczestniczyła w tradycyjnej lekcji matematyki, a po kilku dniach wzięła udział w zajęciach przeprowadzanych przy użyciu omawianej aplikacji [3].

3.3. *„Empiriusz 2.0“*

3.3.1. Wykorzystane narzędzie

„Empiriusz 2.0“ [4] to narzędzie opracowane przez wydawnictwo „Nowa Era“, które wykorzystuje technologię VR do wsparcia nauki w szkołach podstawowych i ponadpodstawowych. Platforma, która jest udostępniana razem z narzędziem, oferuje kilka aplikacji zatytułowanych kolejno:

- „Wirtualne laboratorium chemiczne“
- „Wirtualny atlas anatomiczny“
- „Geometria przestrzenna“
- „Magiczna komnata“
- „Pierwsza pomoc – 4 HELP VR“
- „Ziemia i Wszechświat PRO“

Jak deklaruje wydawnictwo – aplikacje są zgodne z podstawą programową, obowiązującą w szkołach podstawowych i ponadpodstawowych [4].

Urządzenia, na których działa platforma, to bezprzewodowe gogle VR i dwa kontrolery. Gracz wkracza do wirtualnej przestrzeni roboczej, a jego działania są na bieżąco obserwowane przez nauczyciela oraz pozostałych uczniów znajdujących się w klasie, dzięki wyświetlaniu obrazu z perspektywy gracza, np. na tablicy interaktywnej lub monitorze. Wydawnictwo dostarcza również materiały edukacyjne i karty pracy oraz oferuje szkolenia dla nauczycieli z obsługi platformy. Podkreślony jest również pozytywny charakter edukacyjny tego rozwiązania omawiany dalej w 3.3.3.

3.3.2. Aplikacja do nauki geometrii przestrzennej

Geometria przestrzenna jest jednym z tematów opracowanych na potrzeby platformy „Empiriusz 2.0“. Oferuje pomoc dydaktyczną przez rozszerzenie standardowej lekcji matematyki o interaktywną aplikację z zadaniami dla szkół podstawowych i ponadpodstawowych [5]. Na stronie internetowej wydawnictwa możemy przeczytać kilkukrotne wzmianki, o tym że: *„aplikacja zwiększa motywację młodych ludzi do nauki i ułatwia im przyswajanie zagadnień z zakresu geometrii“* [5]. Teza ta pokrywa się z celami tej pracy omówionymi w rozdziale numer 1 i jest dalej omawiana w 3.3.3.

Na rysunku 3.1 możemy zobaczyć zakres tematyki, który obejmuje aplikacja. Dla szkół ponadpodstawowych wydawnictwo przewiduje szerszą tematykę i więcej funkcjonalności, w skład których wchodzą m.in. możliwość nauki o siatkach brył przestrzennych, ich przekroju, kątach w bryłach i dostęp do interaktywnych wzorów.

	Geometria przestrzenna (podstawy)	Geometria przestrzenna (stereometria)
Sugerowany poziom nauczania	Szkoła podstawowa	Szkoła ponadpodstawowa
Rodzaje brył	• 4 graniastosłupy • 3 ostrosłupy • 3 bryły obrotowe	• 6 graniastosłupów • 3 ostrosłupy • 3 bryły obrotowe
Liczba zadań sprawdzających	25	27
Liczba brył z siatkami	7	9
Liczba brył z opcją *Przekroje*	–	7
Liczba brył z opcją *Kąt*	–	4
Interaktywne wzory	NIE	TAK

Rysunek 3.1: Skład zagadnień omawianych dla aplikacji o geometrii przestrzennej dostępnej na platformie „Empiriusz 2.0“ [5]

3.3.3. Wartość edukacyjna narzędzia „Empiriusz 2.0“

Atrakcyjność i pozytywny wpływ aplikacji VR, w kontekście zastosowania narzędzia „Empiriusz 2.0“ są podkreślane wielokrotnie przez wydawnictwo. Platforma ma oferować nowoczesne narzędzie dydaktyczne, uatrakcyjniające naukę [6]. Elementem, który wyróżnia to rozwiązanie jest ciągła obecność nauczyciela podczas lekcji, która odbywa się przy użyciu wirtualnej rzeczywistości. Może on komentować i omawiać poszczególne posunięcia ucznia, który wykonuje zadania. Dodatkowo nauczyciel jest wyposażony w materiały dydaktyczne, bezpośrednio powiązane z omawianymi zagadnieniami.

3.4. Wnioski z analizy dostępnych rozwiązań

Omawiane rozwiązania pokazują, że technologia VR może pozytywnie wpływać nie tylko na efekty edukacyjne, ale także na samopoczucie i nastrój uczniów.

Zgodnie z rozdziałem numer 1. celem tej pracy jest stworzenie narzędzia, które będzie wspierać proces nauczania, a nie go zastępować. Ma zwiększyć zainteresowanie matematyką wśród uczniów szkół średnich poprzez ich samodzielne uczestnictwo w angażującym środowisku.

4. PROJEKT SYSTEMU

4.1. Wymagania funkcjonalne i niefunkcjonalne (Konrad Czarnecki)

Przed przystąpieniem do implementacji aplikacji edukacyjnej w formie escape roomu w środowisku rzeczywistości wirtualnej konieczne było precyzyjne określenie wymagań funkcjonalnych i niefunkcjonalnych, jakie powinna spełniać projektowana aplikacja. Zdefiniowanie wymagań pozwala na kontrolowanie procesu realizacji projektu, zapewnia jego zgodność z oczekiwaniami użytkowników końcowych oraz umożliwia przeprowadzenie testów walidacyjnych w końcowej fazie prac.

4.1.1. Wymagania funkcjonalne

Wymagania funkcjonalne określają, jakie działania użytkownik może wykonać w aplikacji oraz jakie funkcje powinna ona realizować. W przypadku projektowanej aplikacji escape room VR wymagania te obejmują:

- Wyświetlanie wirtualnego środowiska escape roomu – Aplikacja powinna umożliwiać użytkownikowi poruszanie się po wirtualnym pokoju zagadek, obejmującym zestaw pomieszczeń związanych z różnymi działami matematyki.
- Realizacja 13 zagadek matematycznych – aplikacja musi zawierać 13 interaktywnych zagadek, po jednej dla każdego z wybranych działów matematyki: liczby rzeczywiste, wyrażenia algebraiczne, równania i nierówności, układy równań, funkcje, ciągi, trygonometria, planimetria, geometria analityczna, stereometria, kombinatoryka, rachunek prawdopodobieństwa i statystyka oraz optymalizacja i rachunek różniczkowy.
- Interaktywność zagadek – każda zagadka powinna wymagać od użytkownika aktywnej interakcji z otoczeniem wirtualnym, np. manipulowania obiektami, wpisywania odpowiedzi czy przestawiania elementów.
- System weryfikacji poprawności odpowiedzi – aplikacja powinna sprawdzać poprawność rozwiązań podawanych przez użytkownika oraz wyświetlać komunikaty informujące o sukcesie lub błędzie.
- Śledzenie postępu użytkownika – system powinien zapamiętywać, które zagadki zostały już rozwiązane, i odblokowywać dostęp do kolejnych pomieszczeń w ustalonej kolejności.
- Zakończenie rozgrywki – po rozwiązaniu wszystkich 13 zagadek aplikacja powinna wyświetlić ekran podsumowujący z informacją o ukończeniu escape roomu.
- Obsługa urządzeń VR i interfejsów w Laboratorium – aplikacja musi być kompatybilna z systemami śledzenia ruchu i projekcji wykorzystywanymi w Laboratorium Zanurzonej Wizualizacji Przestrzennej, w tym z systemami typu CAVE oraz kontrolerami ruchu.

4.1.2. Wymagania niefunkcjonalne

Wymagania niefunkcjonalne opisują cechy jakościowe systemu oraz warunki, jakie powinien spełniać, aby zapewnić poprawne działanie i pozytywne doświadczenia użytkowników. W kontekście projektowanej aplikacji escape room VR wymagania niefunkcjonalne obejmują:

- Kompatybilność z infrastrukturą LZWP – aplikacja musi działać poprawnie w warunkach technicznych Laboratorium Zanurzonej Wizualizacji Przestrzennej, współpracując z systemem projekcyjnym CAVE oraz wykorzystywanymi kontrolerami ruchu.
- Wydajność – aplikacja powinna działać płynnie, z minimalnym opóźnieniem renderowania obrazu i reakcji na ruchy użytkownika, zapewniając co najmniej 60 klatek na sekundę w środowisku VR.
- Jakość grafiki – wirtualne środowisko escape roomu powinno charakteryzować się realistyczną i spójną oprawą wizualną wspomagającą immersję.
- Łatwość rozbudowy i modyfikacji – struktura projektu powinna umożliwiać przyszłą rozbudowę aplikacji o nowe zagadki, pomieszczenia lub funkcjonalności bez konieczności przebudowy istniejącego kodu i modeli.
- Przenośność kodu i zasobów – wszystkie pliki źródłowe, modele oraz pliki graficzne powinny być przechowywane w repozytorium GitHub w sposób umożliwiający łatwe przenoszenie projektu pomiędzy stanowiskami roboczymi oraz serwerami laboratorium.
- Dokumentacja – projekt musi być opatrzony szczegółową dokumentacją techniczną i użytkową, opisującą strukturę aplikacji, sposób instalacji, obsługi oraz instrukcje dotyczące przyszłej rozbudowy.

4.2. Scenariusz gry (Konrad Czarnecki)

Po uruchomieniu aplikacji gracz przenosi się do pierwszego z 13 wirtualnych pomieszczeń. Każde pomieszczenie jest utrzymane w odmiennym stylu wizualnym i zawiera jedną interaktywną zagadkę matematyczną z przypisanego działu. Zagadki muszą być rozwiązywane w określonej kolejności, a przejście do kolejnego pokoju jest możliwe dopiero po poprawnym rozwiązaniu bieżącej zagadki. Pomijanie zagadek lub powrót do poprzednich pokoi nie są możliwe z poziomu użytkownika.

W każdym pokoju gracz ma możliwość poruszania się po przestrzeni VR, wchodzenia w interakcje z elementami zagadki oraz przechodzenia do kolejnego pomieszczenia po poprawnym rozwiązaniu zagadki.

Działanie każdej zagadki może wymagać różnych form interakcji — takich jak przeciąganie obiektów, wpisywanie wyników na wirtualnej klawiaturze, wskazywanie elementów przestrzeni, czy manipulowanie wirtualnymi narzędziami (np. suwakami i przyciskami). Po rozwiązaniu ostatniej, trzynastej zagadki użytkownik zostaje przeniesiony do wirtualnego pokoju podsumowań, gdzie gra informuje go o odniesionym sukcesie.

W przypadku przerwania rozgrywki przez użytkownika lub awarii systemu, aplikacja powinna zapewniać możliwość powrotu do gry lub całkowitego zamknięcia gry bez ryzyka utraty integralności danych systemu. Rozwiązanie to gwarantuje bezpieczeństwo użytkownika oraz stabilność aplikacji w środowisku CAVE.

Administrator systemu ma możliwość pomijania zagadek i bezpośredniego przeniesienia gracza do wybranego pokoju, co może być wykorzystane na życzenie użytkownika w związku z jego preferencjami w zakresie zagadek i dziedzin matematyki lub w przypadku awarii systemu. Funkcjonalność ta jest dostępna wyłącznie dla uprawnionych użytkowników i nie jest widoczna dla zwykłych graczy, by uniemożliwić przypadkowe pominięcie zagadek.

Tak skonstruowany scenariusz rozgrywki umożliwia nie tylko wprowadzenie użytkownika w świat wirtualnej rzeczywistości, ale także zapewnia dynamiczny, uporządkowany i logiczny przebieg interakcji. Dzięki systematycznemu rozwiązywaniu zagadek i przemieszczaniu się między pokojami, aplikacja utrzymuje wysoki poziom zaangażowania i motywacji użytkowników do ukończenia całej gry.

4.3. Model przypadków użycia (Andrii Demyshyn)

Model przypadków użycia opisuje sposób interakcji użytkowników z projektowaną aplikacją edukacyjną typu escape room w środowisku rzeczywistości wirtualnej. W systemie wyróżniono dwóch aktorów: gracza oraz administratora systemu.

Administrator systemu jest osobą nadzorującą przebieg rozgrywki z poziomu komputera sterującego, znajdującego się poza środowiskiem wirtualnym. Do zadań administratora należy uruchamianie i kończenie aplikacji, monitorowanie przebiegu gry oraz reagowanie na sytuacje awaryjne. Aby uniknąć jakichkolwiek problemów w aplikacji, administratorowi dodano możliwość pomijania zagadek w trakcie gry. Po otrzymaniu prośby lub wystąpieniu jakiegokolwiek błędu administrator może uznać poziom za zaliczony i przenieść gracza do następnej zagadki. To pozwala na zachowanie stabilności działania aplikacji w środowisku CAVE.

Gracz przebywa fizycznie w przestrzeni Laboratorium Zanurzonej Wizualizacji Przestrzennej i korzysta z aplikacji w środowisku VR. Podczas rozgrywki porusza się po wirtualnym pomieszczeniu, wchodzi w interakcję z obiektami poprzez ich naciskanie, przemieszczanie czy wybieranie. Gracz odczytuje informacje tekstowe i komunikaty systemowe wyświetlane w przestrzeni trójwymiarowej oraz rozwiązuje zagadki matematyczne. Także dla wygody użytkowania gracz za pomocą wybranego przycisku może obracać pokój i wszystkie obiekty w nim o 90 stopni, co pozwala dopasować pokój do wygodniejszego korzystania. Na każdym etapie system na bieżąco informuje gracza o poprawności wprowadzanych rozwiązań, wyświetlając odpowiednie komunikaty w przestrzeni wirtualnej. Interakcja z aplikacją odbywa się bez wykorzystania klasycznego interfejsu graficznego, a wszystkie działania realizowane są bezpośrednio w przestrzeni trójwymiarowej. Po poprawnym rozwiązaniu zagadki system automatycznie przechodzi do kolejnego etapu rozgrywki, aż do ukończenia wszystkich trzynastu zagadek.

Zaprojektowany model przypadków użycia odpowiada rzeczywistym funkcjonalnościom zaimplementowanym w aplikacji i zapewnia czytelny podział ról pomiędzy użytkownikiem końcowym a osobą nadzorującą system.

4.4. Projekt architektury systemu (Andrii Demyshyn)

Architektura projektowanej aplikacji została zaprojektowana z myślą o realizacji edukacyjnej gry typu escape room w środowisku rzeczywistości wirtualnej. System został zaimplementowany z wykorzystaniem silnika Unreal Engine i przystosowany do działania w środowisku CAVE, wykorzystywanym w Laboratorium Zanurzonej Wizualizacji Przestrzennej.

Struktura aplikacji opiera się na kilku głównych komponentach odpowiedzialnych za poszczególne aspekty działania systemu. Centralnym elementem architektury jest moduł zarządzania rozgrywką zaimplementowany jako obiekt typu Blueprint Actor MergeBP. MergeBP kontroluje aktualny etap gry, kolejność zagadek oraz postęp gracza. Moduł ten odpowiada również za przejścia pomiędzy kolejnymi etapami po poprawnym rozwiązaniu zagadek matematycznych.

Logika zagadek matematycznych została podzielona na trzynaście niezależnych poziomów gry, każdy z których odpowiada jednej zagadce i jednemu działowi matematyki. Każda zagadka posiada własny mechanizm interakcji oraz weryfikacji poprawności rozwiązania, co umożliwia ich łatwą modyfikację lub rozbudowę bez ingerencji w pozostałą część systemu. Takie podejście upraszcza również proces testowania poszczególnych etapów gry.

Istotnym elementem architektury aplikacji jest sposób realizacji środowiska gry. Cała rozgrywka odbywa się w jednej wspólnej przestrzeni wirtualnej, która dynamicznie zmienia się w zależności od aktualnego etapu gry. Poszczególne elementy pomieszczenia, obiekty zagadek oraz elementy interaktywne są ukrywane, usuwane, aktywowane lub teleportowane w odpowiednie miejsca po ukończeniu danego poziomu. Rozwiązanie to pozwala na zachowanie spójności środowiska oraz ograniczenie konieczności ładowania nowych scen.

Moduł interakcji odpowiada za obsługę kontrolerów ruchu oraz systemów śledzenia pozycji użytkownika w przestrzeni CAVE. Umożliwia on graczowi poruszanie się po wirtualnym pomieszczeniu, manipulowanie obiektami oraz wykonywanie akcji wymaganych do rozwiązania zagadek. Równolegle system obsługuje tryb administratora, który działa z poziomu komputera sterującego i umożliwia nadzorowanie przebiegu rozgrywki oraz ingerencję w jej przebieg w sytuacjach awaryjnych.

Zaprojektowana architektura systemu zapewnia stabilne działanie aplikacji w środowisku rzeczywistości wirtualnej oraz umożliwia jej dalszą rozbudowę, na przykład poprzez dodanie nowych zagadek matematycznych lub rozszerzenie scenariusza gry.

4.5. Projekt interfejsu użytkownika i środowiska gry (Andrii Demyshyn)

Projekt interfejsu użytkownika oraz środowiska gry został opracowany z myślą o specyfice rzeczywistości wirtualnej oraz warunkach pracy w systemie CAVE. Głównym założeniem było zapewnienie wysokiego poziomu immersji oraz intuicyjnej obsługi aplikacji przy jednoczesnym ograniczeniu elementów mogących rozpraszać uwagę gracza podczas rozwiązywania zagadek matematycznych.

W projektowanej aplikacji zrezygnowano z klasycznego, dwuwymiarowego interfejsu graficznego w postaci menu czy stałych elementów HUD. Wszystkie informacje przekazywane użytkownikowi, takie jak komunikaty systemowe, treści zadań czy informacja o poprawności rozwiązania, prezentowane są bezpośrednio w przestrzeni trójwymiarowej jako elementy świata gry. Takie rozwiązanie pozwala na zachowanie spójności wizualnej oraz zwiększa poczucie obecności w środowisku wirtualnym.

Środowisko gry zostało zaprojektowane jako jedna wspólna przestrzeń wirtualna, której wygląd i zawartość zmieniają się w zależności od aktualnego etapu rozgrywki. Poszczególne zagadki matematyczne realizowane są poprzez dynamiczne pojawianie się, ukrywanie lub modyfikowanie obiektów znajdujących się w pomieszczeniu. Dzięki temu możliwe było zachowanie jednolitej przestrzeni przy jednoczesnym wyraźnym rozróżnieniu kolejnych etapów gry.

Interakcja gracza z otoczeniem realizowana jest poprzez kontrolery ruchu, umożliwiające wybieranie, naciskanie oraz przemieszczanie obiektów. Dodatkowo w aplikacji zaimplementowano możliwość obrotu całego pomieszczenia wraz z obiektami o 90 stopni, co pozwala na dostosowanie orientacji przestrzeni do preferencji użytkownika. Rozwiązanie to zwiększa komfort użytkowania aplikacji i ułatwia rozwiązywanie zagadek.

Projekt interfejsu oraz środowiska gry został przetestowany w rzeczywistych warunkach Laboratorium Zanurzonej Wizualizacji Przestrzennej. Przeprowadzone testy pozwoliły na dopasowanie skali obiektów, czytelności komunikatów oraz sposobu interakcji do potrzeb użytkowników, zapewniając płynne i komfortowe korzystanie z aplikacji w środowisku VR.

5. TECHNOLOGIE I NARZĘDZIA (KONRAD CZARNECKI)

Realizacja projektu aplikacji edukacyjnej typu escape room w środowisku rzeczywistości wirtualnej wymagała doboru odpowiednich narzędzi oraz technologii, które umożliwiłyby stworzenie interaktywnej, atrakcyjnej wizualnie i funkcjonalnej aplikacji kompatybilnej z systemami dostępnymi w Laboratorium Zanurzonej Wizualizacji Przestrzennej.

5.1. Silnik gry

Do stworzenia aplikacji wirtualnego escape roomu zdecydowano się na wykorzystanie silnika Unreal Engine 5, który jest jednym z najpopularniejszych środowisk do tworzenia gier komputerowych oraz aplikacji wirtualnej rzeczywistości. Unreal Engine, rozwijany przez firmę Epic Games, umożliwia tworzenie zaawansowanych wizualnie, interaktywnych projektów 3D oraz VR dzięki nowoczesnemu systemowi renderowania, rozbudowanemu edytorowi oraz wsparciu dla technologii immersyjnych.

Silnik ten oferuje użytkownikom możliwość programowania logiki aplikacji zarówno w języku C++, jak i z wykorzystaniem wizualnego systemu skryptowego Blueprint, co znacząco przyspiesza proces tworzenia prototypów i ułatwia implementację interakcji w środowisku VR. Unreal Engine zapewnia również bogaty zestaw narzędzi do tworzenia animacji, efektów specjalnych, obsługi dźwięku oraz integracji z zewnętrznymi bibliotekami.

Podczas analizy możliwych rozwiązań rozważano również wykorzystanie silnika Unity, który podobnie jak Unreal Engine jest szeroko stosowany w branży gier i aplikacji VR. Unity charakteryzuje się dużą elastycznością, wsparciem dla wielu platform oraz dostępnością licznych wtyczek i rozszerzeń. W porównaniu z Unreal Engine, Unity posiada mniej rozbudowany natywny system graficzny oraz wymaga większego nakładu pracy przy tworzeniu zaawansowanych efektów wizualnych.

Ostatecznym argumentem przemawiającym za wyborem Unreal Engine była kwestia kompatybilności — Laboratorium Zanurzonej Wizualizacji Przestrzennej, w którym aplikacja miała zostać wdrożona, nie wspiera nowszych wersji Unity, natomiast Unreal Engine zapewniał pełną zgodność z istniejącą infrastrukturą sprzętową i programową laboratorium.

5.2. Laboratorium Zanurzonej Wizualizacji Przestrzennej

Laboratorium Zanurzonej Wizualizacji Przestrzennej (LZWP) to specjalistyczne środowisko badawczo-edukacyjne wyposażone w systemy rzeczywistości wirtualnej, umożliwiające tworzenie i testowanie aplikacji immersyjnych w warunkach kontrolowanych. W skład laboratorium wchodzą między innymi systemy typu CAVE, czyli pomieszczenia projekcyjne z ekranami ściennymi i podłogowymi, które otaczają użytkownika obrazem 3D wyświetlanym z kilku projektorów.

LZWP wyposażone jest również w systemy śledzenia pozycji i ruchu użytkownika oraz kontrolery umożliwiające interakcję z wirtualnym środowiskiem. Dzięki temu laboratorium stanowi doskonałe zaplecze do testowania aplikacji edukacyjnych VR oraz prowadzenia badań nad efektywnością i ergonomią rozwiązań immersyjnych.

5.3. Środowisko 3D

W procesie tworzenia aplikacji konieczne było opracowanie modeli 3D reprezentujących obiekty, elementy wystroju oraz interaktywne przedmioty pojawiające się w wirtualnym escape roomie. Zamiast korzystać z zewnętrznego oprogramowania do modelowania, wszystkie modele zostały utworzone bezpośrednio w Unreal Engine, wykorzystując jego wbudowane narzędzia do tworzenia i teksturowania obiektów 3D.

Silnik pozwala na generowanie szczegółowych modeli, nadawanie materiałów i tekstur oraz przygotowywanie elementów gotowych do interakcji w środowisku VR, co usprawniło proces integracji obiektów z aplikacją i zapewniło pełną kompatybilność z funkcjonalnościami gry.

5.4. System kontroli wersji

Dla zapewnienia bezpieczeństwa danych oraz efektywnego zarządzania projektem zastosowano system kontroli wersji Git wraz z usługą hostingową GitHub. GitHub umożliwia przechowywanie kodu źródłowego, modeli 3D, plików dźwiękowych oraz dokumentacji projektowej w repozytorium zdalnym z możliwością współdzielenia zasobów pomiędzy członkami zespołu.

Dzięki systemowi kontroli wersji możliwe było śledzenie historii zmian, zarządzanie gałęziami projektowymi oraz szybkie przywracanie poprzednich wersji w przypadku wystąpienia błędów. W projekcie GitHub pełnił również rolę platformy do przechowywania backupów.

6. ORGANIZACJA PRACY (JAN WALCZAK)

Zarządzanie zespołem i dobra organizacja pracy to dwa kluczowe aspekty podczas pracy grupowej. Metodyczne działanie pozwala usprawnić i uporządkować pracę projektową i komunikację w zespole [7]. Zespół, który implementował omawiane w tej pracy rozwiązanie, składał się z trzech osób: Jana Walczaka, Konrada Czarneckiego i Andriego Demyshyna.

6.1. System kontroli wersji

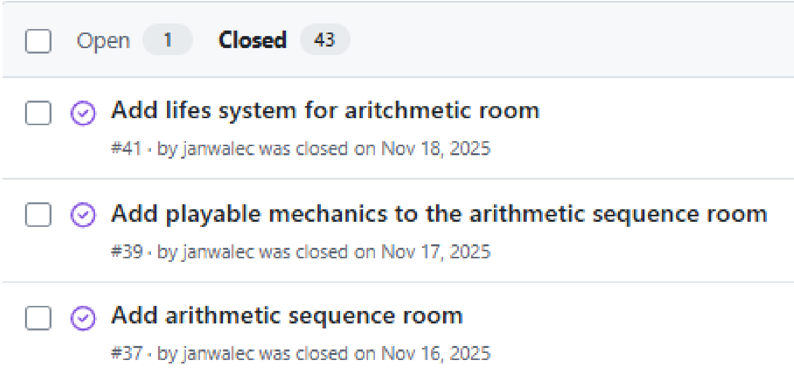
6.1.1. Zastosowane rozwiązanie

System kontroli wersji pomaga śledzić historię zmian dokonywanych przez uczestników projektu. Ułatwia zarządzanie procesem jego tworzenia, zapewnia integralność danych oraz gwarantuje, że dokonywane zmiany są na bieżąco zapisywane i udostępniane uczestnikom oraz użytkownikom [8].

Do wdrożenia systemu kontroli wersji w omawianym projekcie użyto internetowej platformy GitHub. Organizacja pracy sprowadza się do opracowania zasad dotyczących przepływu informacji, który ma na celu opisanie i wdrażanie zmian zachodzących w danym projekcie tak, aby ułatwić ich zrozumienie nie tylko deweloperom, ale także użytkownikom, którzy śledzą zmiany zachodzące w repozytorium [8].

6.1.2. Kontrola przepływu

Przepływ zaczyna się od zdefiniowania problemu, poprzez opisanie problemu lub funkcjonalności, których będzie dotyczyć implementowana zmiana. Na platformie GitHub odbywa się to poprzez tworzenie `Issues` (zagadnień, rys. 6.1). Pozwalają one również na planowanie i prowadzenie dyskusji na temat danego zagadnienia z innymi uczestnikami projektu lub użytkownikami [9]. Dodatkową zaletą jest możliwość bezpośredniego powiązania zmian dokonanych w kodzie z unikatowym numerem, który jest nadawany każdej takiej dyskusji przez omawianą platformę. W omawianym projekcie zdecydowano się na wprowadzanie zasad dotyczących formułowania tytułu i treści `Issue`. Tytuł powinien być napisany w języku angielskim, tak żeby każda osoba przeglądająca repozytorium mogła powiązać problem i dyskusję ze zmianą w aplikacji. Dodatkowo powinien być zwięzły i dotyczyć wyłącznie jednego zagadnienia. Opis i dyskusja mogą być napisane w języku polskim, tak żeby maksymalnie usprawnić komunikację zespołu. `Issue` może znajdować się w dwóch stanach: zamkniętym – czyli stan, w którym praca została zakończona i otwartym – czyli stan, w którym praca nad danym zagadnieniem nadal trwa.



Rysunek 6.1: Zrzut ekranu z platformy GitHub – tablica z przykładowymi Issues do projektu.

W repozytorium znajduje się tak zwana główna linia (ang. *main line of development*). Jest to liniowa historia zmian, które zostały wprowadzone do projektu. Deweloperzy mogą wprowadzać swoje zmiany, które są definiowane tym, w jaki sposób (w którym punkcie w historii zmian) odłączyły się od tej linii. W ten sposób można tworzyć rozgałęzienia (ang. *branches*) względem głównej linii [10]. Główna gałąź (ang. *main branch*), może umożliwić deweloperom zorientowanie się, czy zmiana, którą chcą wprowadzić, będzie mogła być połączona z aktualnie istniejącą wersją aplikacji. Ważnym elementem przepływu jest więc zdefiniowanie tego, w jaki sposób każda zmiana będzie dołączana do głównej gałęzi. Podczas implementacji projektu zdecydowano się na bezpośrednie powiązanie tworzonych gałęzi z `Issues`, tj. nazywanie ich tymi samymi tytułami oraz wiązanie ich przez linkowanie unikatowego identyfikatora nadawanego każdemu `Issue`.

6.1.3. Łączenie gałęzi

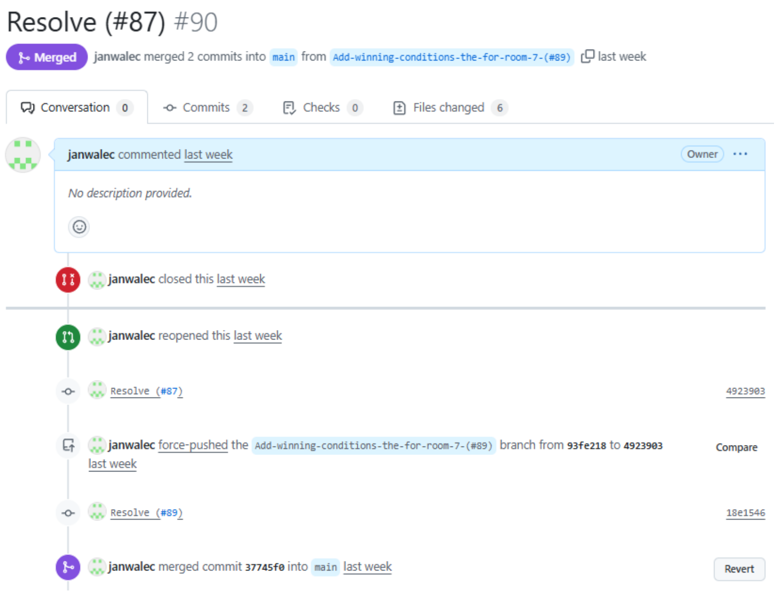
Domyślnie gałęzie mogą być wiązane bez żadnych ograniczeń. GitHub umożliwia tworzenie zbioru zasad (ang.*rulesets*) dotyczących wiązania gałęzi deweloperów z innymi gałęziami. Zasady te pozwalają na określenie, które grupy są upoważnione do jakich czynności związanych z daną gałęzią oraz definiują jakie zasady muszą spełnić, aby powiązać swoją gałąź, czyli wprowadzaną zmianę, do danej linii. [11]. W projekcie zastosowano następujące zasady dotyczące głównej linii:

- `Restrict deletions` – nie pozwalaj na usuwanie powiązanych referencji.
- `Require a pull request before merging` – powiąż z gałęzią przez „pull request".
- `Block force pushes` – nie pozwól na wymuszenie nadpisania zmian.

W celu spełnienia zasady `Require a pull request before merging` każda wprowadzana zmiana musi zostać przygotowana do połączenia w odpowiedni sposób. Odbywa się to przez mechanizm `Pull request` (rys. 6.2). Umożliwia on wgląd we wprowadzane modyfikacje innym uczestnikom projektu przed ich finalnym zatwierdzeniem i wdrożeniem [12]. W projekcie określono, że żeby zmiana została połączona z główną gałęzią, musi być zatwierdzona przez minimalnie jednego uczestnika projektu, niebędącego autorem. Dodatkowo, zmiana musi być możliwa do powiązania bez występowania konfliktów, czyli fragmentów plików, których system kontroli wersji nie mógł samodzielnie scalić ze zmianami [12].

Zmiany wprowadzane przez deweloperów składają się z serii migawek `Commit`. `Commit` zawiera wiadomość, unikatowy identyfikator oraz listę plików i zmian, które są z nim powiązane. Pozwala na wielokrotny zapis zmian podczas pracy w ramach jednej gałęzi. W projekcie zostało nałożone ograniczenie, wymuszające na uczestnikach scalanie migawek w jedność za pomocą mechanizmu `rebase` i `squash` [12].

Po poprawnym utworzeniu i scaleniu `Pull request` utworzona wcześniej gałąź może zostać usunięta. Jest to bezpieczne, ponieważ gałąź zostaje dołączona do głównej linii jako `Commit` i historia zachowuje swój liniowy charakter. Deweloper powinien oznaczyć `Issue` jako zamknięte. `Pull request` otrzyma status `Merged` (scalone) automatycznie.



Rysunek 6.2: Zrzut ekranu z platformy GitHub – tablica z przykładowym Pull Request

6.1.4. *Przechowywanie dużych plików – Git LFS*

Projekt był realizowany w środowisku Unreal Engine, przez co w projekcie znajdowało się dużo plików o dużym rozmiarze. Co do zasady Git nie jest przystosowany do przechowywania dużych plików (takich jak wideo, audio, grafiki), a jedynie małych plików binarnych takich jak tekst czy pliki z kodem.

W projekcie zastosowano mechanizm `Git LFS` czyli „Git Large File Storage“. Jest to system, który umożliwia tworzenie wskaźników kierujących do danego pliku, który jest umieszczany poza repozytorium (na zewnętrznym serwerze) [13]. Repozytorium wymaga wyspecyfikowania, które pliki powinny być przechowywane nie jako pliki binarne w repozytorium, a jako dowiązania. W tym celu należało zastosować odpowiednie filtry wskazujące na duże obiekty tworzone w Unreal Engine: rozszerzenia `*.uasset` oraz `*.umap`. W przypadku omawianej platformy zagwarantowany jest dysk o pojemności *10 GB* znajdujący się bezpośrednio po stronie platformy.

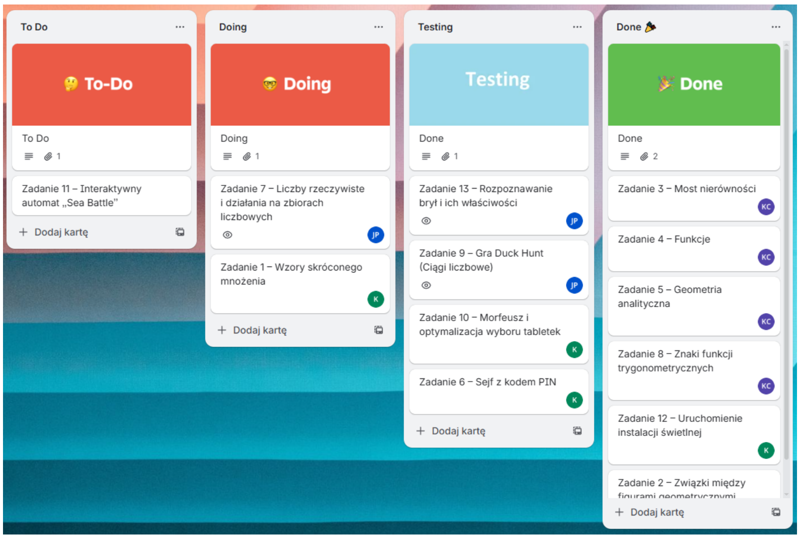
6.2. *Tablica Kanban*

Aby dobrze zarządzać projektem potrzebna jest jasna komunikacja i wyznaczanie celów już od najwcześniejszych etapów rozwoju projektu. Praktyki definiujące wymagania i przydzielające uczestników projektu do poszczególnych zadań znacząco zwiększają ich zaangażowanie i gwarantują, że aplikacja będzie spełniać potrzeby zdefiniowane przez interesariuszy [14].

Żeby zapewnić powyższe wymagania zdecydowano się na zastosowanie zwinnej metodyki Kanban. Jednym z jej głównych zadań jest wizualizacja działań podejmowanych w projekcie [15]. Metodyka jest często stosowana w formie fizycznej i występuje wtedy jako tablica podzielona na odpowiednie kolumny. Standardowy, trzykolumnowy podział prezentuje się następująco:

- ToDo – zadania do zrobienia,
- Doing – zadania, nad którymi trwa praca,
- Done – zadania zakończone.

Uczestnicy umieszczają samoprzylepne karteczki z krótko opisanymi zadaniami. W opisywanym projekcie zastosowano narzędzie internetowe, służące do wizualizacji wykonywanych zadań. Każdy uczestnik otrzymał przydział zadań według deklarowanych przez niego preferencji. Dodano dodatkową kolumnę „Testing“, w której były umieszczane zadania, które były w danej chwili testowane (rys. 6.3).



Rysunek 6.3: Zrzut ekranu z platformy Trello – wizualizacja tablicy Kanban

Kolejnym, charakterystycznym elementem metodyki Kanban jest kontrola przepływu [15]. Dzięki sekwencyjnemu przemieszczaniu karteczek z zadaniami na tablicy, zgodnie z zasadą, że można przesunąć tylko jedną karteczkę na raz (od lewej do prawej), uczestnicy mają możliwość śledzenia postępu prac. Aby kontrola przepływu działała sprawnie konieczna jest samodyscyplina uczestników w uaktualnianiu postępów wykonywanych zadań.

Kontrola przepływu wiąże się bezpośrednio z zasadą limitu WIP (ang. *Work In Progress*) określoną dla metodyki, która odróżnia ją od innych, zwinnych metodyk [15]. W przypadku opisanym w projekcie określała ona, nad iloma zagadkami na raz może pracować każdy uczestnik projektu. Zdecydowano się na ustalenie tego limitu na jedną zagadkę na raz. W takim przypadku każdy uczestnik projektu mógł umieścić tylko jedną karteczkę, przypisaną do niego, w kolumnie „Doing“.

7. WSTĘPNY PROJEKT ZAGADEK

W poniższym rozdziale znajduje się teoretyczny opis zagadek, który stanowi punkt wejściowy do implementacji projektu inżynierskiego. Stanowi on podstawę teoretyczną i nakreśla charakter pracy. Opisy poszczególnych zagadek zostały zredagowane na podstawie ogólnodostępnej podstawy programowej dla szkół średnich z przedmiotu matematyka. Jest to kluczowy aspekt całego projektu inżynierskiego, gdyż aplikacja końcowa ma być skierowana do uczniów szkół średnich, w związku z czym wymagany zakres wiedzy nie może wykraczać poza podstawę programową.

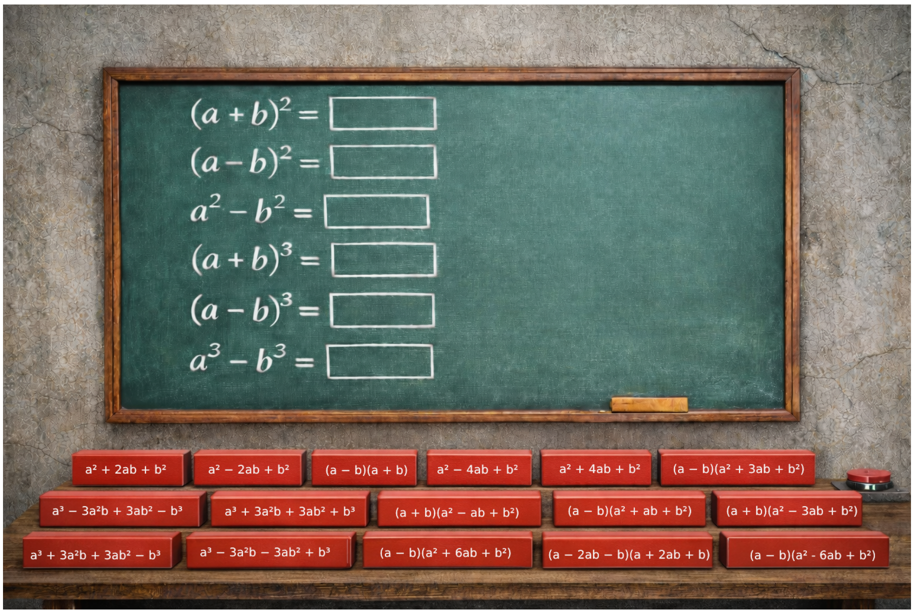
7.1. Zadanie 1 – Wzory skróconego mnożenia (Andrii Demyshyn)

7.1.1. Cel zadania

Celem zadania jest utrwalenie wiedzy z zakresu wzorów skróconego mnożenia oraz rozwijanie umiejętności rozpoznawania poprawnych zależności algebraicznych. Zadanie ma charakter wprowadzający i pozwala uczestnikowi na przypomnienie sobie podstawowych wzorów omawianych w programie nauczania matematyki na poziomie szkoły średniej. Dodatkowo zadanie wspiera rozwój logicznego myślenia oraz umiejętność analizy struktury wyrażeń algebraicznych.

7.1.2. Zasady działania

Zadanie polega na dopasowaniu początków wzorów skróconego mnożenia do ich poprawnych zakończeń. Uczestnik otrzymuje zestaw rozpoczętych wyrażeń algebraicznych, w których brakują prawe strony równań. Równocześnie dostępny jest zbiór możliwych zakończeń wzorów, wśród których znajdują się zarówno poprawne, jak i niepoprawne zapisy algebraiczne (rys. 7.1).



Rysunek 7.1: Przykładowy wygląd pierwszego zadania – wzory skróconego mnożenia

Celem użytkownika jest wybranie właściwych elementów i połączenie ich w taki sposób, aby utworzyć kompletne i matematycznie poprawne wzory skróconego mnożenia. Zadanie wymaga znajomości podstawowych własności działań algebraicznych oraz umiejętności odróżniania poprawnych wzorów od błędnych.

7.1.3. Zakończenie zadania

Zadanie uznaje się za zakończone w momencie poprawnego uzupełnienia wszystkich wzorów skróconego mnożenia. Prawidłowe rozwiązanie potwierdza, że uczestnik posiada wymaganą wiedzę teoretyczną oraz potrafi ją zastosować w praktyce poprzez rozpoznawanie i kompletowanie wyrażeń algebraicznych.

7.2. Zadanie 2 – Planimetria (Jan Walczak)

Po ukończeniu pierwszego zadania pojawia się tabela z trzema kolumnami. Dwie skrajne kolumny zawierają nazwy figur geometrycznych. Środkowa kolumna jest pusta i będzie uzupełniana przez gracza odpowiednimi symbolami.

7.2.1. Cel zadania

Celem gracza jest poprawne ułożenie relacji między figurami w kolejnych rzędach. Przykładowo, relacją taką jest to, że „każdy kwadrat jest rombem” lub „każdy kwadrat jest prostokątem”.

7.2.2. Zasady działania

Gracze przechodzą przez kolejne rzędy sekwencyjnie (rys. 7.2). Dopóki nie ułożą pierwszej relacji poprawnie, to nie mogą ułożyć kolejnej. Po poprawnym ułożeniu rzędu (wstawieniu odpowiedniego symbolu) podświetla się on na zielono, sygnalizując graczowi, że może przejść do kolejnego punktu. Symbole, które układają gracze są następujące:

- > to <
- > to nie <

Kwadrat	> to <	Prostokąt
Prostokąt	> to nie <	Kwadrat
Romb	[ ]	Kwadrat
...	...	...

Rysunek 7.2: Przykładowy diagram zależności między figurami geometrycznymi.

Gracze kierują się swoją wiedzą matematyczną oraz następującymi własnościami figur:

- liczba boków,
- długości boków,
- kąty (proste lub nie),
- cechy charakterystyczne danej figury.

7.2.3. Zakończenie zadania

Po poprawnym uzupełnieniu wszystkich relacji zadanie uznaje się za zakończone. Gracz może przejść do kolejnego zadania.

7.3. Zadanie 3 – Nierówności (Konrad Czarnecki)

Po wejściu do kolejnego pomieszczenia gracz widzi przed sobą most zawieszony nad przepaścią. Po drugiej stronie znajduje się zamknięte przejście, do którego gracz musi się dostać. Na bocznych ścianach wypisany jest układ nierówności: $2x - 5 < 9$; $x + 1 \geq 4$. Na płytkach mostu umieszczone są różne liczby z zakresu liczb całkowitych (np.: $1, 2, 3, 4, 5, 6, 8, 7, 10$).

7.3.1. Cel zadania

Gracz musi rozwiązać układ nierówności i wyznaczyć przedział, który spełnia oba warunki. Następnie powinien przeprowadzić niewielką figurkę przez most, odpowiednio nią poruszając, tak aby przesuwała się wyłącznie po płytkach z wartościami należącymi do tego przedziału. Poruszanie figurką odbywa się za pomocą czterech przycisków (w prawo, w lewo, do przodu i do tyłu) znajdujących się na jednej ze ścian. Rozwiązanie układu

1. Rozwiązanie pierwszej nierówności:
   - $2x - 5 < 9$
   - $2x < 14$
   - $x < 7$
2. Rozwiązanie drugiej nierówności:
   - $x + 1 \geq 4$
   - $x \geq 3$
3. Wspólny przedział:
   - $x \in [3;7)$

Gracz musi wybrać wyłącznie płytki z wartościami większymi lub równymi 3 i mniejszymi od 7, np. $3, 4, 5, 6$.

7.3.2. Zasady działania

- Gracz poruszając postacią, przechodzi przez kolejne płytki.
- Poprawna płytka – postać przechodzi dalej.
- Błędna płytka – płytka zapada się lub podświetla na czerwono, a postać wraca na początek mostu. Gracz może próbować dowolną liczbę razy, aż do skutecznego przejścia na drugą stronę.

7.3.3. Zakończenie zadania

Zadanie zostaje uznane za zakończone, gdy graczowi uda się przejść na drugą stronę mostu.

7.4. Zadanie 4 – Funkcje (Konrad Czarnecki)

Zadania 4 i 5 są realizowane w jednym pomieszczeniu. Gracz widzi dwie duże tablice, umieszczone na ścianach. Na obu naniesiona jest siatka układu współrzędnych. W układzie współrzędnych pojawiają się trzy punkty, np.:

- Punkt $A(1, 2)$

- Punkt $B(3, 4)$
- Punkt $C(5, 6)$

oraz wzór funkcji kwadratowej ze wszystkimi współczynnikami domyślnie ustawionymi na 0. Poniżej widoczne są elementy służące do sterowania wykresem funkcji. Dają one możliwość zmiany współczynników wylosowanego wzoru.

7.4.1. Cel zadania

Celem gracza jest dobranie odpowiednich współczynników funkcji krawdatowej za pomocą wirtualnego suwaka tak, aby przeszła ona przez wszystkie wyświetlone punkty.

7.4.2. Zasady działania

Podczas wybierania współczynników funkcji przez gracza, jej wykres na tablicy zmienia się na bieżąco zgodnie z ustawionym wzorem. Wykres funkcji powinien być losowany z wcześniej zdefiniowanej puli par typu funkcja – punkty, tak aby po każdorazowym uruchomieniu zadania gracz czuł, że ma przed sobą nowe wyzwanie.

7.4.3. Zakończenie zadania

Rozwiązanie jest sprawdzane na bieżąco i gdy gracz dobierze współczynniki funkcji kwadratowej poprawnie, tj. przetnie ona wszystkie wybrane punkty, zadanie zostaje uznane za rozwiązane. W takim przypadku tablica się blokuje i podświetla na zielono.

7.5. Zadanie 5 – Geometria analityczna (Konrad Czarnecki)

W układzie współrzędnych pojawiają się dwa wykresy funkcji liniowych. Są one losowane z pewnej określonej puli, podobnie jak w zadaniu 4.

7.5.1. Cel zadania

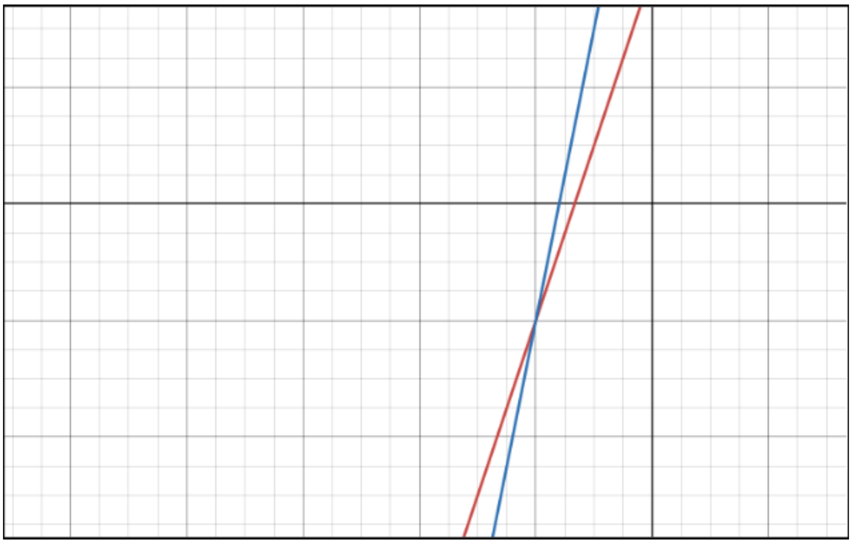
Zadaniem gracza jest znalezienie punktu przecięcia dwóch funkcji liniowych wyświetlanych w układzie współrzędnych. Określenie puli wyklucza możliwość prostych równoległych, które nie mają ze sobą żadnych punktów wspólnych lub w całości się pokrywają. W tych przypadkach zadanie byłoby niemożliwe do rozwiązania. Gracz będzie wpisywał swoją odpowiedź (współrzędne punktu przecięcia) na klawiaturze znajdującej się pod tablicą.

7.5.2. Zasady działania

Skala osi współrzędnych jest dobrana tak, by gracz nie mógł odczytać z niej rozwiązania; musi rozwiązać układ równań dysponując dwoma wzorami funkcji. Przykładowe dwa wykresy przecinających się funkcji (rys. 7.3):

- $y = 3x + 4$ (czerwony)
- $y = 5x + 8$ (niebieski)

Gracz dysponuje tymi wzorami i na ich podstawie musi rozwiązać układ równań. Wykresy powinny być dobrane tak, aby obie współrzędne punktu przecięcia były liczbami całkowitymi.



Rysunek 7.3: Przykładowy wygląd tablicy z przecinającymi się wykresami

7.5.3. Zakończenie zadania

Rozwiązanie jest sprawdzane na bieżąco i gdy gracz dobierze współrzędne punktu przecięcia funkcji poprawnie, zadanie zostaje uznane za rozwiązane. W takim przypadku tablica się blokuje i podświetla na zielono. Jeśli gracz wykonał wcześniej zadanie czwarte (znajdujące się w tym samym pokoju), przechodzi do następnego zadania.

7.6. Zadanie 6 – Kombinatoryka (Andrii Demyshyn)

7.6.1. Cel zadania

Gracz zostaje poinformowany, że musi wyliczyć, ile jest możliwych do ułożenia trzycyfrowych kodów do sejfu bez powtarzających się cyfr.

7.6.2. Zasady działania

Na środku pokoju znajduje się zamknięty sejf (rys. 7.4). Rozwiązaniem jest kod, a nie liczba – oznacza to, że początkową cyfrą w trzycyfrowym kodzie może być zero. Informacja ta powinna być jasno zakomunikowana graczowi, np. na tabliczce umieszczonej nad sejfem.

Rozwiązaniem zadania jest:

$$10 \cdot 9 \cdot 8 = 720$$

7.6.3. Zakończenie zadania

Po poprawnym wpisaniu kodu sejf automatycznie się otwiera, co utwierdza gracza w przekonaniu, że poprawnie wykonał zadanie. W środku znajdują się elementy potrzebne do wykonania następnego zadania.

Rysunek 7.4: Przykładowy wygląd sejfu występującego w zadaniu

7.7. *Zadanie 7 – Liczby rzeczywiste i działania na zbiorach liczbowych (Jan Walczak)*

W następnym zadaniu gracze znajdują skrzynię, na której powierzchni – z każdej strony – narysowane są liczby: *-3; 7; 12; 2; 5; 0; 10; 3,75;* √2.

7.7.1. Cel zadania

Skrzynia wyposażona jest w mechaniczny zamek z pięcioma polami na cyfry, obok których znajdują się symbole zbiorów:

- *N* – liczby naturalne,
- *Z* – liczby całkowite,
- *R* – liczby rzeczywiste,
- *Q* – liczby wymierne,
- *R\Q* – liczby niewymierne,

Gracz musi uważnie obejrzeć skrzynię i policzyć ile liczb, zapisanych na skrzyni, należy do jakiego zbioru. Po wpisaniu odpowiednich liczb, obok symboli zbiorów, skrzynia otwiera się, a gracz otrzymuje kilka symboli:

- zawieranie się zbiorów – ⊂,

- zbiór pusty – $\varnothing$,
- iloczyn zbiorów – $\cap$.

Tym samym gracz przechodzi do drugiego etapu zadania. W drugim etapie w pokoju ukazuje się plansza z symbolami zbiorów, takimi jak na skrzyni. Celem gracza jest ułożenie poprawnej relacji między nimi tj. zbiór liczb naturalnych zawiera się w zbiorze liczb całkowitych, zbiór liczb całkowitych zawiera się w zbiorze liczb rzeczywistych itd.

7.7.2. Zasady działania

W pierwszym etapie zadania gracz powinien policzyć, ile liczb ze skrzyni należy do danego zbioru i wpisać odpowiednią odpowiedź na kłódce. Przykładowo:

- *N* (naturalne): *7; 12; 2; 5; 10* – 5 liczb,
- *Z* (całkowite): *-3; 7; 12; 2; 5; 0; 10* – 7 liczb,
- *R* (rzeczywiste): wszystkie liczby – 9 liczb,
- *Q* (wymierne): *-3; 7; 12; 2; 5; 0; 10; 3,75* – 8 liczb,
- *R\Q* (niewymierne): $\sqrt{2}$ – 1 liczba.

Kombinacja do ustawienia na kłódce: *5, 7, 9, 8, 1*.

W drugim etapie gracz powinien ustawić posiadane symbole między literami symbolizującymi kolejne zbiory i odpowiednio je obrócić, tak aby powstała między nimi poprawna relacja tj.

- $N \subset Z \subset R$,
- $R \backslash Q \cap Q = \varnothing$,
- $R \backslash Q \cap R = R \backslash Q$.

7.7.3. Zakończenie zadania

Zadanie zostaje uznane za zakończone, gdy gracz poprawnie przejdzie przez oba etapy. Gracz nie może przejść do etapu drugiego, bez zakończenia etapu pierwszego – jest to wymuszone przez wymaganie otwarcia przez niego skrzyni.

7.8. Zadanie 8 – Znaki funkcji trygonometrycznych (Konrad Czarnecki)

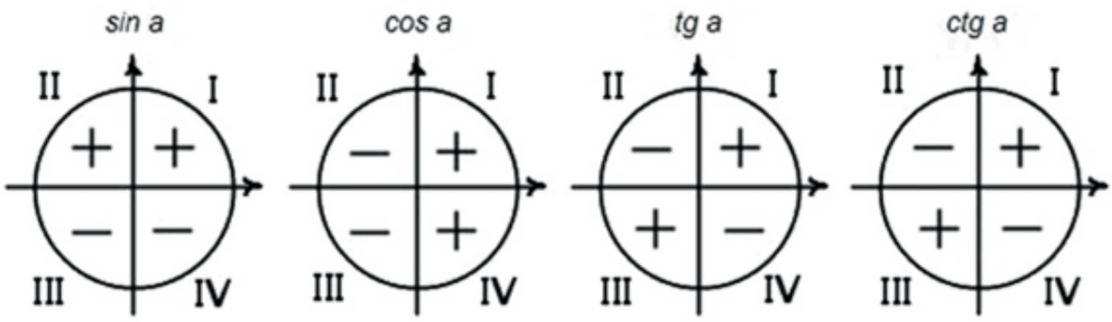
Po ukończeniu zadania ze zbiorami gracz kieruje się do ściany z czterema dużymi okręgami jednostkowymi, oznaczonymi nazwami funkcji:

- $sin(\alpha)$
- $cos(\alpha)$
- $tg(\alpha)$
- $ctg(\alpha)$

Każdy z okręgów podzielony jest na cztery ćwiartki, oznaczone jako I, II, III, IV. Przy każdej ćwiartce znajduje się puste miejsce, które gracz musi wypełnić odpowiednim znakiem (rys. 7.5):

- „+” (plus) — funkcja przyjmuje wartości dodatnie
- „−” (minus) — funkcja przyjmuje wartości ujemne

Gracz zmienia znak naciskając na niego. Domyślnie wszystkie znaki są ustawione na puste, dopiero po pierwszym naciśnięciu zmieniają się na znak „+”, po kolejnym na „−”, następnie znów na „+”, itd.



Rysunek 7.5: Ćwiartki układow jednostkowych ze znakami funkcji trygonometrycznych

7.8.1. Cel zadania

Celem gracza jest poprawne ustawienie znaków funkcji trygonometrycznych w każdej ćwiartce układu współrzędnych.

7.8.2. Zasady działania

Gracz modyfikuje znaki „+” i „−” w odpowiednich miejscach na planszy naciskając na nie. Może dowolnie poprawiać swój wybór, dopóki nie zatwierdzi odpowiedzi.

7.8.3. Zakończenie zadania

Po poprawnym ułożeniu wszystkich znaków ściana rozświetla się na zielono.

7.9. Zadanie 9 – Ciągi liczbowe (Jan Walczak)

Poniżej opisane zadanie będzie podobne do gry wyprodukowanej przez Nintendo na platformę Pegasus (rys. 7.6).

7.9.1. Cel zadania

Zadaniem gracza będzie odpowiednio szybko obliczyć kolejne wyrazy ciągu arytmetycznego z podanego wzoru. Będzie on wyposażony w wirtualny, laserowy pistolet, którym będzie musiał strzelać w odpowiednio oznaczone kaczki.

7.9.2. Zasady działania

Na jednej ze ścian pokoju pojawia się formuła ciągu arytmetycznego, np.: $a_n = 2n + 1$ wybranego z predefiniowanej puli. Gracz będzie musiał wybrać i strzelić do kaczki oznaczonej odpowiednią wartością kolejnych wyrazów ciągu. Gra przyspiesza (kaczki lecą coraz szybciej) razem z postępem w zadaniu. Aby ułatwić graczowi zadanie, na górze jednej ze ścian pokoju będzie podany nie tylko wzór ciągu, ale też aktualny numer wyrazu tego ciągu. Przykładowo, dla wzoru $a_n = 2n + 1$:

- $n = 1$ – gracz musi trafić w kaczkę z liczbą 3,
- $n = 2$ – gracz musi trafić w kaczkę z liczbą 5

7.9.3. Zakończenie zadania

Gracz będzie zdobywał punkty za każdy poprawnie wybrany wyraz ciągu. Gra zakończy się po upływie określonego czasu lub jeśli gracz pomyli się trzy razy.

Rysunek 7.6: gra Duck Hunt

7.10. Zadanie 10 – Prawdopodobieństwo (Andrii Demyshyn)

Gracze podchodzą do stołu ustawionego w rogu pokoju, gdzie siedzi postać Morfeusza. Na stole stoją dwa identyczne pojemniki, a obok leży 100 tabletek — 50 czerwonych i 50 niebieskich (rys. 7.7).

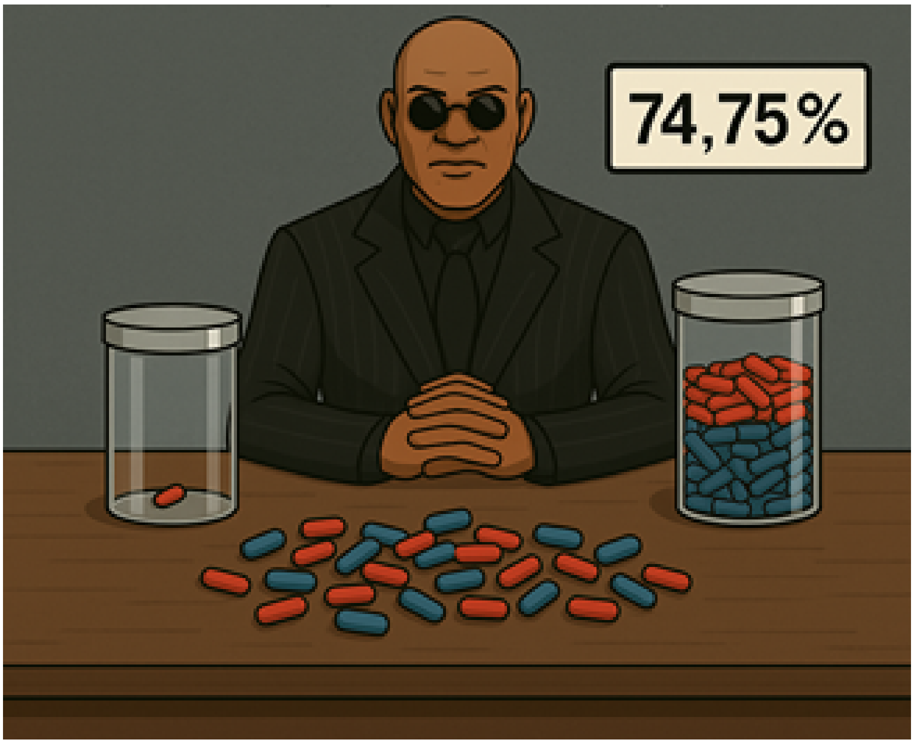
Pojawia się komunikat: „Pomóż Morfeuszowi zwiększyć jego szanse na powrót do rzeczywistości. Podziel tabletki między dwa pojemniki tak, aby miał jak największą szansę na wybranie czerwonej."

7.10.1. Cel zadania

Gracze muszą znaleźć optymalny układ, czyli: w pierwszym pojemniku umieścić jedną czerwoną tabletkę, w drugim pojemniku umieścić 49 czerwonych i 50 niebieskich tabletek (lub odwrotnie. Rozwiązanie działa niezależnie od tego, który pojemnik traktujemy jako „pierwszy", a który jako „drugi"). Taki układ osiągnie maksymalne prawdopodobieństwo wylosowania czerwonej tabletki – 74,75%.

7.10.2. Zasady działania

Gracze mogą: przeciągać tabletki do pojemników w dowolny sposób, testować różne rozkłady, sprawdzać procent szans na wygraną, który wyświetla się po każdym rozłożeniu.



Rysunek 7.7: Przykładowy wygląd w zadaniu

Po każdym rozkładzie system oblicza i wyświetla szansę na wygraną. Jeśli gracz nie osiągnie 74,75%, pojawia się komunikat: „Możesz to zrobić lepiej! Spróbuj jeszcze raz.” Jeśli gracz osiągnie 74,75% system gratuluje i zalicza zadanie.

7.10.3. Zakończenie zadania

Po poprawnym rozłożeniu tabletek i osiągnięciu maksymalnej szansy 74,75%, Morfeusz wstaje od stołu, uśmiecha się i mówi: „Dziękuję. Dzięki Wam mam szansę wrócić do rzeczywistego świata”. Po chwili Morfeusz znika, rozpuszczając się w powietrzu niczym hologram lub efekt teleportacji, symbolizujące jego powrót do rzeczywistości.

7.11. Zadanie 11 – Optymalizacja i rachunek różniczkowy (Andrii Demyshyn)

7.11.1. Cel zadania

Celem zadania jest utrwalenie wiedzy z zakresu rachunku różniczkowego, w szczególności podstawowych wzorów na pochodne funkcji elementarnych. Zadanie pozwala uczestnikowi na przypomnienie i zastosowanie poznanych zależności matematycznych obowiązujących w programie nauczania matematyki na poziomie szkoły średniej. Dodatkowo zadanie wspiera rozwój logicznego myślenia, koncentracji oraz umiejętności analizy i kojarzenia zależności pomiędzy funkcją a jej pochodną.

7.11.2. Zasady działania

Zadanie polega na dopasowaniu funkcji do odpowiadających im pochodnych. Uczestnik widzi panel z zamkniętymi tabliczkami, które po naciśnięciu odsłaniają wzory matematyczne z zakresu rachunku różniczkowego. Na tabliczkach znajdują się zarówno funkcje, jak i ich pochodne (rys. 7.8).

Uczestnik może jednocześnie odkryć maksymalnie dwie tabliczki. Jeżeli odkryte elementy przedstawiają funkcję oraz jej prawidłową pochodną, para zostaje zaliczona i usunięta z panelu. W przypadku błędnego dopasowania tabliczki po krótkim czasie odwracają się z powrotem do stanu początkowego. System na bieżąco weryfikuje poprawność wyborów użytkownika.

Rysunek 7.8: Przykładowy wygląd zadania – dopasowanie funkcji do pochodnych

Zadanie wymaga znajomości podstawowych wzorów na pochodne funkcji, takich jak x^n, $\sqrt{x}$, $\ln x$, e^x, a^x oraz $\frac{1}{x}$, a także umiejętności poprawnego rozpoznawania ich pochodnych.

7.11.3. Zakończenie zadania

Zadanie uznaje się za zakończone w momencie poprawnego dopasowania wszystkich funkcji do ich pochodnych. Prawidłowe wykonanie zadania potwierdza, że uczestnik posiada wymaganą wiedzę z zakresu rachunku różniczkowego oraz potrafi zastosować ją w praktyce poprzez analizę i rozpoznawanie zależności matematycznych.

7.12. Zadanie 12 – Układy równań (Andrii Demyshyn)

7.12.1. Cel zadania

Na środku pomieszczenia znajduje się stół z trzema przezroczystymi pojemnikami oraz kulkami w trzech kolorach: czerwonym, zielonym oraz niebieskim. Bezpośrednio nad stołem, na ścianie, umieszczona jest duża żarówka, która zapala się, gdy gracze prawidłowo uzupełnią pojemniki (rys. 7.9). Obok stołu znajduje się instrukcja techniczna opisująca zasady działania instalacji.

Zadanie składa się z dwóch etapów. W pierwszym etapie gracz, na podstawie podanych zależności matematycznych, wyznacza energię generowaną przez pojedynczą kulkę każdego koloru. W drugim etapie, znając już wartości energetyczne kulek, gracz dobiera kulki do pojemników w taki sposób, aby spełnić warunek uruchomienia żarówki.

Instrukcja techniczna wygląda następująco:

„Aby uruchomić instalację świetlną, należy załadować dokładnie 11 kulek, które muszą łącznie generować 48 jednostek energii. Czerwona kulka jest dwa razy bardziej wydajna energetycznie niż zielona. Suma energii jednej kulki czerwonej oraz dwóch kulek zielonych jest o 9 jednostek większa niż energia jednej kulki niebieskiej. Suma energii kulki czerwonej, zielonej i niebieskiej wynosi 12 jednostek".

7.12.2. *Zasady działania*

- Gracze wkładają kolorowe kulki do pojemników.
- System na bieżąco sprawdza poprawność ustawienia.
- W razie błędu wyświetlany jest komunikat: „Niepoprawne ustawienie. Spróbuj ponownie".

$$\begin{cases} x = 2y & x \quad \text{– energia generowana przez jedną kulkę czerwoną,} \\ x + 2y - z = 9 & y \quad \text{– energia generowana przez jedną kulkę zieloną,} \\ x + y + z = 12 & z \quad \text{– energia generowana przez jedną kulkę niebieską.} \end{cases}$$

- Energia jednej kulki czerwonej wynosi $x = 6$,
- energia jednej kulki zielonej wynosi $y = 3$,
- energia jednej kulki niebieskiej wynosi $z = 3$.

Po wyznaczeniu energii pojedynczych kulek gracz przechodzi do etapu doboru odpowiedniej konfiguracji kulek.

Rysunek 7.9: Przykładowy wygląd zadania

7.12.3. *Zakończenie zadania*

Po poprawnym ułożeniu żarówka zapala się nad stołem, sygnalizując zakończenie zadania.

7.13. *Zadanie 13 – Stereometria (Jan Walczak)*

Bryły przestrzenne mogą sprawić uczniom trudność, ponieważ wymagają przejścia od rysunku dwuwymiarowego tj. takiego na kartce papieru, do wyobrażenia sobie ich w przestrzeni trójwymiarowej.

7.13.1. Cel zadania

Celem zadania jest zapoznać uczniów z podstawowymi własnościami brył geometrycznych poprzez ich samodzielne odkrywanie.

7.13.2. Zasady działania

Gracz wchodzi do ciemnego pokoju i jest wyposażony w źródło światła np. pochodnię lub latarkę. Na środku pokoju znajduje się jedna z brył, wylosowanych z danej puli: prostopadłościan, ostrosłup, graniastosłup, kula lub sześcian. Na ścianie pokoju znajduje się panel z pytaniami np.:

- Jak nazywa się ta bryła?
- Jaki jest wzór na obliczenie jej pola powierzchni?
- Jaki jest wzór na obliczenie jej objętości?

Gracz obchodząc bryłę dookoła i rozświetlając ją latarką powinien móc udzielić odpowiedzi na te pytania.

7.13.3. Zakończenie zadania

Gdy gracz udziela poprawnych odpowiedzi na kolejne pytania, zostają one podświetlone na zielono, sygnalizując graczowi, że wykonał zadanie poprawnie. Jeżeli się pomyli, to losowana jest kolejna bryła, inna niż poprzednia. Gra kończy się w momencie, kiedy gracz udzieli poprawnych odpowiedzi na pytania dotyczące dwóch różnych brył.

8. IMPLEMENTACJA ZAGADEK

8.1. *Zadanie 1 – Wzory skróconego mnożenia (Andrii Demyshyn)*

8.1.1. Problemy i różnice w realizacji zadania w praktyce

Podczas implementacji zadania nie wystąpiły istotne problemy techniczne. Zastosowane rozwiązania w pełni odpowiadają założeniom projektowym, a mechanika przeciągania i dopasowywania elementów została zaimplementowana w sposób intuicyjny dla gracza.

W tym zadaniu zrealizowano zagadkę poświęconą wzorom skróconego mnożenia. Zadaniem gracza jest prawidłowe połączenie początków i końcówek wzorów skróconego mnożenia poprzez umieszczenie odpowiednich klocków w przeznaczonych dla nich miejscach.

Kiedy gracz pojawia się w pokoju, widzi przed sobą tablicę z początkami wzorów skróconego mnożenia do uzupełnienia, a także klocki z różnymi formułami, panel informacyjny oraz przycisk do sprawdzenia odpowiedzi.

8.1.2. Opis obiektów pokoju

Matematyczna tablica `BP_MathBoard`

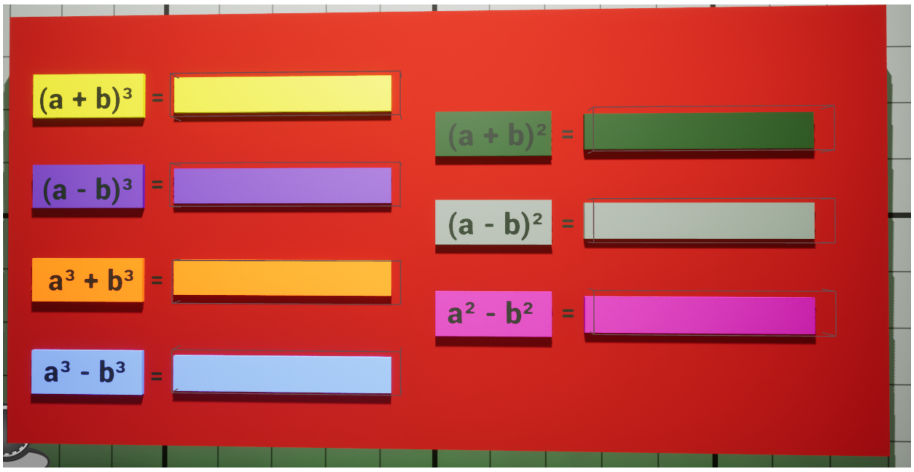
Głównym obiektem pokoju jest matematyczna tablica zrealizowana w postaci blueprintu aktora `BP_MathBoard` rys. 8.1. Tablica zawiera siedem wzorów do uzupełnienia:

1. **Kwadrat sumy**
$$(a+b)^2 = a^2 + 2ab + b^2$$
2. **Kwadrat różnicy**
$$(a-b)^2 = a^2 - 2ab + b^2$$
3. **Różnica kwadratów**
$$(a-b)(a+b) = a^2 - b^2$$
4. **Sześcian sumy**
$$(a+b)^3 = a^3 + 3a^2b + 3ab^2 + b^3$$
5. **Sześcian różnicy**
$$(a-b)^3 = a^3 - 3a^2b + 3ab^2 - b^3$$
6. **Rozkład sumy sześcianów**
$$a^3 + b^3 = (a+b)(a^2 - ab + b^2)$$
7. **Rozkład różnicy sześcianów**
$$a^3 - b^3 = (a-b)(a^2 + ab + b^2)$$

Naprzeciw każdego wzoru, po znaku „=”, znajduje się odpowiedni slot, w którym należy umieścić końcówkę wzoru. Każdy slot składa się z następujących elementów:

- wyróżnionego kolorem obszaru,
- komponentu `BoxCollision`,
- komponentu `Scene Component -- Point`,
- oczekiwanego identyfikatora `ExpectedID`,
- zmiennej typu bool `Slot[i]IsNoFull`, która informuje, czy slot jest zajęty (domyślnie ustawiona na `True`),
- własnego materiału slotu `Slot[i]HighLightMaterial`.

Każdemu slotowi odpowiada tylko jeden poprawny obiekt z formułą.



Rysunek 8.1: Wygląd BP_MathBoard

8.1.3. Zakończenie zadania

W obiekcie `BP_MathBoard` zaimplementowano również funkcję `AreAllSlotsFilled`, która sprawdza, czy wszystkie sloty zostały uzupełnione oraz czy zostały uzupełnione poprawnie.

Prostopadłościan z formułą `BP_FormulaBlock`

Wszystkie formuły są przedstawione jako interaktywne obiekty 3D w postaci `BP_FormulaBlock` typu Blueprint Actor. Gracz może podnosić i przenosić te obiekty po pokoju oraz umieszczać je w slotach tablicy.

Każdy obiekt `BP_FormulaBlock` składa się z:

- komponentu `StaticMesh`,
- komponentu `TextRender` z odpowiednią formułą,
- komponentu `BoxCollision`,
- identyfikatora `FormulaID`.

Występują zarówno poprawne obiekty `BP_FormulaBlock`, zawierające prawidłowe końcówki wzorów, jak i obiekty fałszywe (fikcyjne). Obiekty fałszywe wizualnie nie różnią się od poprawnych, lecz zawierają formuły oraz identyfikatory `FormulaID`, które nie pasują do żadnego wzoru. Zostały one dodane w celu zwiększenia poziomu trudności oraz zmylenia gracza. Formuły te są niemal poprawne, jednak zawierają drobne błędy, takie jak niewłaściwe znaki, współczynniki lub mieszanie typów wzorów.

Przycisk sprawdzenia `BP_CheckMath`

Na poziomie zaprojektowano przycisk `BP_CheckMath`, który umożliwia sprawdzenie poprawności uzupełnienia tablicy `BP_MathBoard`. Gracz decyduje, kiedy chce sprawdzić rozwiązanie, a po kliknięciu przycisku aktywowana jest funkcja `AreAllSlotsFilled`.

Panel informacyjny `BP_MathText`

Instrukcja zadania wyświetlana jest za pomocą blueprintu aktora `BP_MathText`, w którym umieszczono blueprintowy widget `W_MathText`. Ogólny wygląd pokoju algebraicznego, w którym realizowane jest zadanie, przedstawiono na rys. 8.2.

Rysunek 8.2: Wygląd pokoju algebraicznego

8.1.4. Przebieg zadania

Po rozpoczęciu zadania gracz znajduje się w pokoju i widzi na panelu informacyjnym `BP_MathText` komunikat:

> „Połącz odpowiednie początki i końce wzorów skróconego mnożenia. Przeciągnij klocki we właściwe miejsca.”

Jeżeli gracz podniesie obiekt z formułą i spróbuje umieścić go w slocie, w momencie wejścia obiektu w obszar `BoxCollision` slotu, obiekt tymczasowo przejmuje materiał slotu. Jeżeli w tym momencie gracz upuści obiekt, zostaje on automatycznie umieszczony w slocie. Rozwiązanie to zostało zastosowane w celu ułatwienia precyzyjnego umieszczania obiektów. Jeżeli gracz wyciągnie obiekt z obszaru kolizji slotu, jego materiał zostaje przywrócony do pierwotnego stanu.

W momencie umieszczenia obiektu w slocie tablicy `BP_MathBoard`, zmienna `Slot[i]IsNoFull` przyjmuje wartość `False`, co uniemożliwia umieszczenie innego obiektu w tym samym slocie do momentu usunięcia aktualnie umieszczonego klocka.

Gracz w dowolnym momencie może nacisnąć przycisk `BP_CheckMath`, który po interakcji odtwarza animację wciśnięcia oraz wywołuje funkcję `AreAllSlotsFilled`. Funkcja ta w pierwszej kolejności sprawdza, czy wszystkie sloty zostały uzupełnione. Jeżeli nie, do widgetu `W_MathText` wysyłany jest komunikat:

> „Nie wszystkie formuły są jeszcze uzupełnione. Połącz każdą część początkową z odpowiednim zakończeniem.”

Jeżeli wszystkie sloty są uzupełnione, funkcja sprawdza poprawność wzorów poprzez porównanie identyfikatora `FormulaID` obiektu z identyfikatorem `ExpectedID` slotu. W przypadku poprawnego dopasowania wyświetlany jest komunikat:

> „Świetnie! Poprawnie połączyłeś wzory skróconego mnożenia.”

Zadanie zostaje zaliczone, a gra przechodzi do kolejnego poziomu.

W przeciwnym przypadku wyświetlany jest komunikat:

> „Części formuł zostały połączone niepoprawnie. Spróbuj dobrać inne pary.”

Gracz może ponawiać próby do momentu poprawnego rozwiązania zadania.

8.2. Zadanie 2 – Planimetria (Jan Walczak)

8.2.1. Problemy i różnice w realizacji zadania w praktyce

Podczas implementacji zadania, zgodnie z projektem opisanym w podrozdziale 7.2, wystąpiło kilka problemów. Przede wszystkim zauważono, że odpowiedzi są zero-jedynkowe. Przykładowo: uczeń ma do uzupełniania relację typu „każdy kwadrat ... prostokątem“. Jeśli odpowie niepoprawnie, tj. zaznaczy odpowiedź „nie jest“, a zostanie od razu zapytany ponownie, o tę samą relację, to od razu wyklucza jedną z odpowiedzi. Tym samym zadanie zatraca swoją wartość edukacyjną – uczeń może rozwiązać całe zadanie, stosując jedynie metodę eliminacji.

Rozwiązanie tego problemu, które zostało zaimplementowane, to zdefiniowanie puli takich relacji i po udzieleniu przez ucznia odpowiedzi, każdorazowe losowanie relacji innej niż ta poprzednia. W ten sposób uniemożliwia się uczniowi stosowanie zasady eliminacji i wymusza na nim prawidłowe podejście.

Kolejnym problemem była prezentacja zadania. Zwykła tabela, którą uczeń miałby uzupełniać, mogłaby wydać mu się mało ciekawa. Tym samym postanowiono wizualnie usprawnić zagadkę. Na środku zostały umieszczone dwa przyciski:

- każdy
- nie każdy

Uczeń zostaje poinstruowany, że po obu ścianach pokoju zostaną wyświetlone różne figury geometryczne. Po lewej stronie, patrząc od przycisków – figury oznaczone kolorem czerwonym, po prawej stronie – figury oznaczone kolorem zielonym. Liczba figur jest stała. Każdorazowo zbiór wyświetlanych figur jest wybierany z określonej puli i wyświetlany w pseudolosowej konfiguracji – losowane jest ich położenie, obrót oraz rozmiar. Zadaniem ucznia jest uzupełnianie kolejnych relacji poprzez wybieranie odpowiednich przycisków. Relacja, wyświetlana na ścianie pokoju, przedstawia się jako: „zbiór figur reprezentowanych przykładami w kolorze czerwonym (po lewej stronie) zawiera się / nie zawiera się w zbiorze figur reprezentowanych przykładami w kolorze zielonym (po prawej stronie)“. Dodatkowo tekst jest odpowiednio pokolorowany tak, aby uczeń nie miał wątpliwości, że chodzi o figury z danego zbioru, które są wyświetlane tymże kolorem na ścianach.

8.2.2. Implementacja struktury danych przechowującej relację

Na początku pracy należało zdefiniować strukturę, przechowującą relacje, czyli pytania, na które uczeń będzie odpowiadał. Relacja taka zawiera następujące pola:

- `Every` – wartość logiczna

- `FigureA` – ciąg tekstowy, reprezentujący pierwszą figurę w relacji
- `FigureB` – ciąg tekstowy, reprezentujący drugą figurę w relacji

Wartość zmiennej `Every` odpowiada na pytanie, „czy każda figura typu pierwszego (`FigureA`) jest równocześnie figurą typu drugiego (`FigureB`)?". Struktura zawiera również funkcję słownikową, czyli taką, która tłumaczy wartości tekstowe na liczbowe.

8.2.3. Implementacja kontrolerów

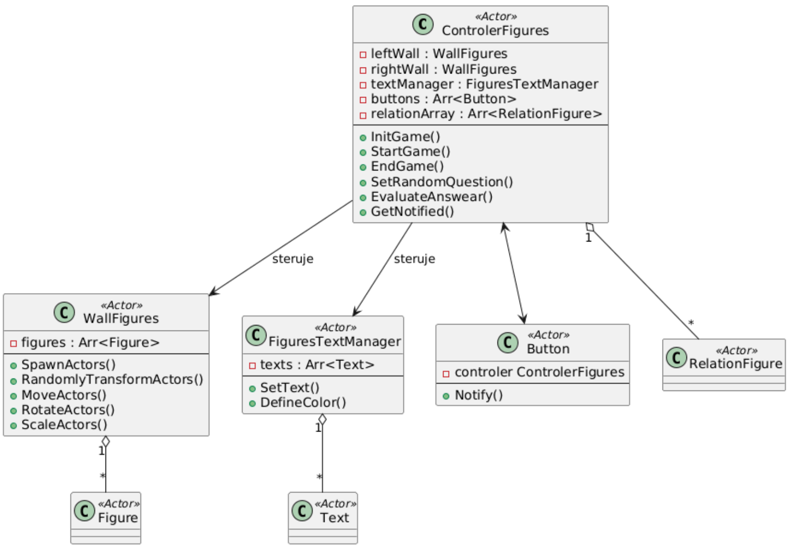
Aby zarządzać zagadką została zaimplementowana seria kontrolerów i menedżerów przedstawionych na rysunku 8.3.

- `ControlerFigures` (główny kontroler)
- `FiguresTextManager`
- `WallFigures`

`WallFigures` to najprostszy z kontrolerów, wykonuje polecenia głównego kontrolera. Jego zadaniem jest wyświetlanie żądanych figur i ich losowe ustawianie w świecie gry (obracanie, skalowanie, rozmieszczanie). W projekcie występują jego dwie instancje: kontroler lewy i prawy. Odpowiadają one za odpowiednie ściany, na których wyświetlane są figury.

`FiguresTextManager` jest odpowiedzialny za zarządzanie tekstem wyświetlanym na ekranie. Wykonuje polecenia głównego kontrolera. Jego zadaniem jest odpowiednie wyświetlanie i kolorowanie tekstu.

`ControlerFigures`, czyli kontroler główny, jest najważniejszym elementem zagadki. Zawiera referencję do pozostałych kontrolerów, zarządza obiektami wyświetlanymi na scenie i kontroluje przebieg zadania. Dodatkowo zawiera dwustronną referencję do przycisków (`Button`), które wysyłają do niego powiadomienia o tym, że zostały naciśnięte. Kontroler zawiera też predefiniowaną tablicę relacji typów figur opisanych wcześniej.



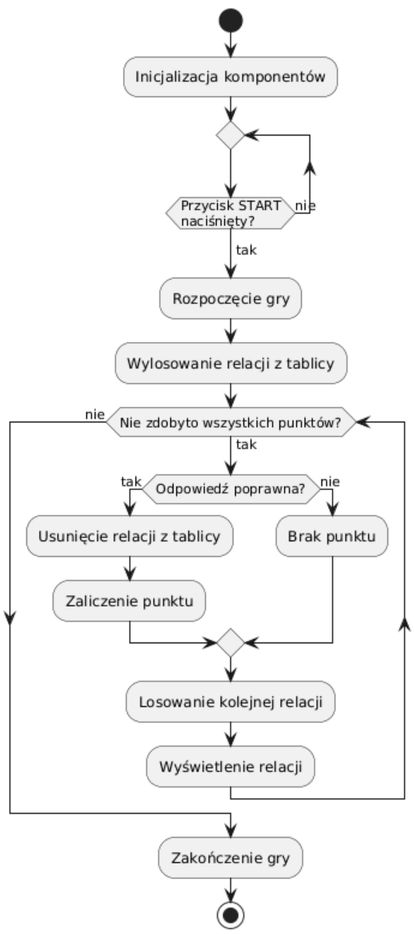
Rysunek 8.3: Diagram UML zawierający najważniejsze elementy kontrolerów dla zadania 2 – planimetria.

8.2.4. Przebieg zadania

Główny kontroler jest odpowiedzialny za inicjalizację zadania. Uruchamia on funkcję `InitBlueprint`, odpowiedzialną za wywołanie odpowiednich funkcji podrzędnych elementów, takich jak: ustawienie tekstu, inicjalizacja ścian i przycisków.

Kontroler oczekuje na otrzymanie powiadomienia od przycisku odpowiedzialnego za uruchomienie gry. Kiedy otrzyma powiadomienie, przesłane mu za pośrednictwem odpowiedniej funkcji, uruchamia grę, tj. wywołuje funkcje podrzędnych komponentów, ustawiając im odpowiednią widoczność w świecie gry.

Z predefiniowanej tablicy relacji zostaje wybrana jedna, która zostaje wyświetlona graczowi Główny kontroler czeka na powiadomienia od przycisków. Kiedy zostaje powiadomiony o tym, że jeden z nich został naciśnięty sprawdza poprawność odpowiedzi. Jeśli odpowiedź jest poprawna, usuwa relację z tablicy i losuje kolejną. Jeśli nie jest poprawna, losuje kolejną relację do wyświetlenia graczowi i nie zalicza punktu. Gra kończy się po poprawnym uzupełnieniu przez gracza wszystkich relacji. Przebieg całego zadania został zaprezentowany w postaci diagramu przepływu na rysunku 8.4.



Rysunek 8.4: Diagram przepływu dla zadania 2 – planimetria

8.3. *Zadanie 3 – Nierówności (Konrad Czarnecki)*

8.3.1. Problemy i różnice w realizacji zadania w praktyce

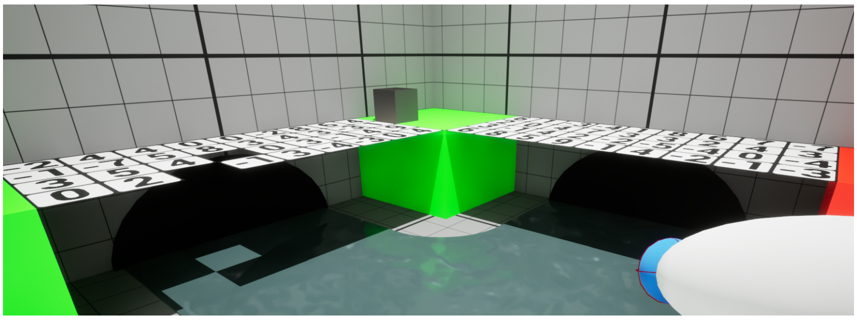
Podczas implementacji zadania, zgodnie z projektem opisanym w podrozdziale 7.3, ze względów estetycznych zmieniono postać, która miała być przeprowadzana przez most, na metaliczną kostkę, która zamiast przesuwania się po kolejnych kafelkach, obraca się wzdłuż swojej krawędzi, tym samym zmieniając swoją pozycję.

8.3.2. Główny kontroler

Zadaniem głównego kontrolera jest zarządzanie przebiegiem całego zadania. Porusza kostką w kierunkach wskazanych przez gracza, sprawdza poprawność ruchów, informuje o porażce lub zwycięstwie, zrzuca kafelki, które nie spełniają wylosowanej nierówności, jeśli gracz pokierował na nie kostkę, a także blokuje niedozwolone ruchy, takie jak opuszczenie mostu przez jego krawędź lub wejście w dziurę po kafelku, który wcześniej spadł – takie błędy świadczyłyby o przypadkowych, niezamierzonych akcjach, a nie o braku matematycznej wiedzy.

Kiedy gracz dotrze do punktu kontrolnego w połowie mostu, główny kontroler zapisuje tę informację i podświetla punkt kontrolny na zielono (rys. 8.5), aby w przypadku porażki w dalszej części mostu, kostka mogła zostać cofnięta do tego punktu, a nie do samego początku. Odblokowanie punktu kontrolnego uniemożliwia również cofnięcie się do pierwszej sekcji mostu.

Główny kontroler przechowuje także zmienną `real Position`, która określa aktualną pozycję kostki na moście względem kafelków, niezależnie od jej fizycznego położenia w świecie gry, które może być niewystarczająco precyzyjne. Zatem, gdy gracz przesunie kostkę o jeden kafelek w przód, wartość parametru y zmiennej `real Position` zostanie zwiększona o jeden. Cofanie zmniejszy wartość tego parametru o jeden, a ruchy w lewo i prawo odpowiednio zmienią wartość parametru x.



Rysunek 8.5: Punkt kontrolny w pokoju 3

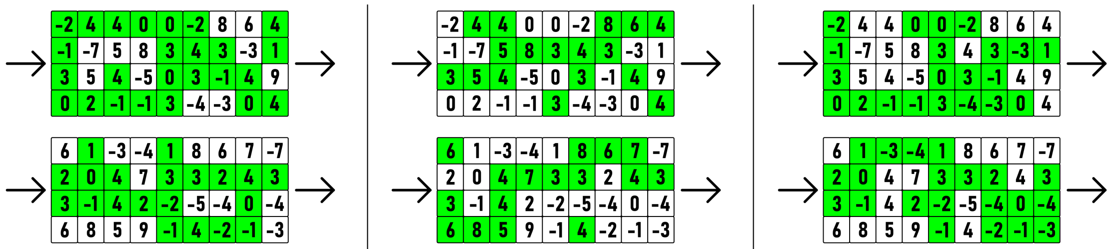
8.3.3. Wybór kafelków i układ nierówności

Gracz może przesuwać kostkę jedynie po kafelkach, które są oznaczone liczbami spełniającymi wybrany układ nierówności. W celu zwiększenia różnorodności rozgrywki, układ nierówności jest wybierany w sposób losowany ze zdefiniowanego wcześniej zbioru.

Lista układów nierówności oraz ich rozwiązań:

- $3x-4<11$; $x+1>-2$; rozwiązania całkowite: $-2, -1, 0, 1, 2, 3, 4$
- $2x-5>-1$; $x-3<6$; rozwiązania całkowite: $3, 4, 5, 6, 7, 8$
- $x+7>2$; $-x+1>-3$; rozwiązania całkowite: $-4, -3, -2, -1, 0, 1, 2, 3$

W związku z faktem, że zbiory liczb spełniające układ nierówności różnią się w zależności od układu, który zostanie wylosowany, a liczby na kafelkach zawsze pozostają te same, należało odpowiednio dobrać kafelki, tak aby w każdym z przypadków istniała droga przez obie sekcje mostu (rys. 8.6).



(a) Rozwiązanie dla $x \in [-2, 4]$ (b) Rozwiązanie dla $x \in [3, 8]$ (c) Rozwiązanie dla $x \in [-4, 3]$

Rysunek 8.6: Ścieżki przejścia przez obie sekcje mostu dla każdego z układów nierówności

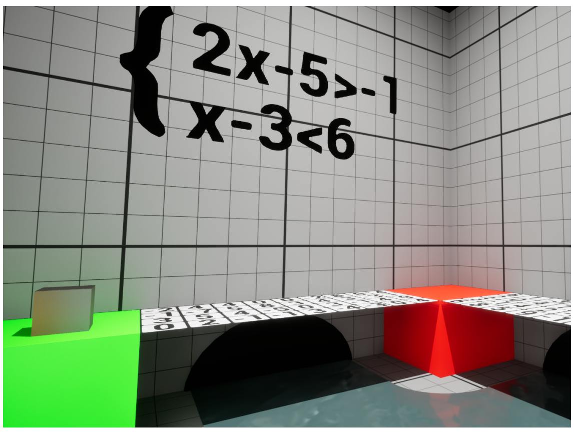
8.3.4. Przebieg zadania

Zadanie rozpoczyna się wraz z wywołaniem funkcji `Put On Tiles`, która ustawia kafelki z liczbami w odpowiednich pozycjach, budując z nich most, a także w losowy sposób wybiera układ nierówności, który gracz będzie musiał wykorzystać. Następnie główny kontroler wywołuje funkcję `Died`, która ustawia kostkę na początku mostu w pozycji wyjściowej. Most dzieli się na dwie sekcje oddzielone bezpiecznym punktem kontrolnym (rys. 8.7) – jeśli gracz przegra w drugiej sekcji, kostka zostanie cofnięta do punktu kontrolnego, a nie do samego początku.

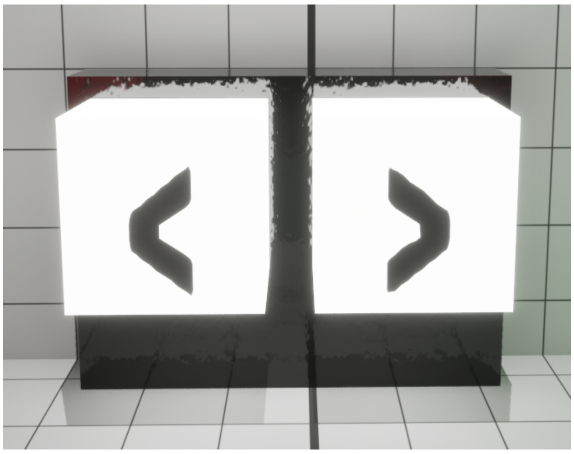
Po wstępnej konfiguracji rozpoczyna się właściwa część zagadki. Gracz dysponuje czterema przyciskami (rys. 8.8), odpowiadającymi czterem kierunkom (w przód, w tył, w lewo i w prawo), które po naciśnięciu wywołują funkcję `turn Side` z odpowiednim parametrem w głównym kontrolerze. Jego zadaniem jest przeprowadzenie kostki przez most, poruszając się jedynie po kafelkach, których liczby spełniają wylosowaną nierówność. Jeśli kostka stanie na kafelku, który nie spełnia nierówności, kafelek i kostka spadają w przepaść, a gracz musi zacząć od początku lub od punktu kontrolnego, w zależności, w której sekcji mostu popełnił błąd.

Niemożliwe jest wypadnięcie z mostu poprzez wyjście poza jego krawędzie lub wejście w dziurę po kafelku, który wcześniej spadł – gra uniemożliwia takie ruchy.

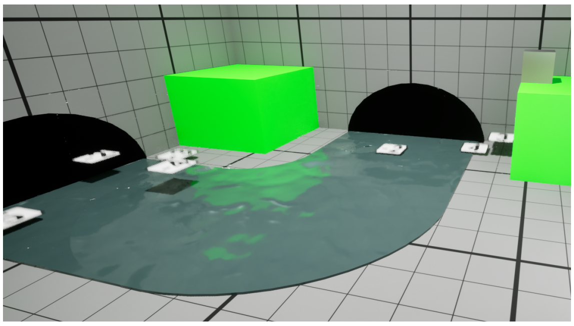
Po przejściu obu sekcji mostu, wszystkie pozostałe kafelki spadają do przepaści, a zagadka zostaje uznana za rozwiązaną (rys. 8.9).



Rysunek 8.7: Most podzielony na dwie sekcje w pokoju 3



Rysunek 8.8: Przyciski do poruszania kostką w pokoju 3



Rysunek 8.9: Rozwiązany pokój 3

8.4. Zadanie 4 – Funkcje (Konrad Czarnecki)

8.4.1. Problemy i różnice w realizacji zadania w praktyce

Jedyną istotną zmianą zadania względem opisu teoretycznego w podrozdziale 7.4 było rozdzielenie zagadki czwartej i piątej na dwa oddzielne pokoje. Pierwotnie oba zadania miały być realizowane w jednym pomieszczeniu, jednak aby udostępnić graczowi większą przestrzeń, zdecydowano się to zmienić.

8.4.2. Wybór dostępnych punktów

Zadanie składa się z trzech części. W każdej liczba punktów, które należy przeciąć wykresem funkcji zwiększa się o jeden, gracz otrzymuje dodatkowy suwak, pozwalający mu modyfikować następny współczynnik. Losowe punkty, które gracz ma za zadanie przeciąć, są wybierane z wcześniej zdefiniowanego zbioru. Zbiór ten został dobrany w taki sposób, aby współczynniki, które należy dobrać zawsze były liczbami całkowitymi, aby poziom trudności rósł wraz z kolejnymi częściami zadania, oraz by gracz musiał wykorzystywać wszystkie dostępne suwaki. Zatem części drugiej (z dwoma punktami do przecięcia i dwoma suwakami) nie da się rozwiązać, wykorzystując jedynie pierwszy suwak odpowiedzialny za współczynnik C. Analogicznie, część trzecia (z trzema punktami do przecięcia i trzema suwakami) wymaga wykorzystania wszystkich dostępnych suwaków.

Lista możliwych punktów do przecięcia w każdej części zadania:

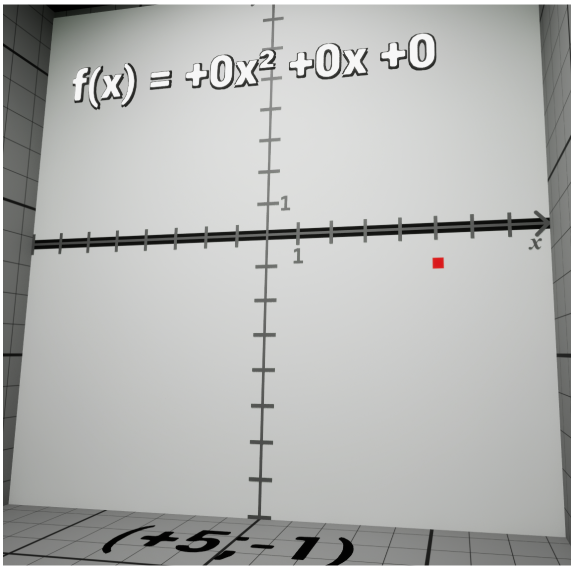
1. Część pierwsza (jeden punkt):
   - $P_1(4;3)$; rozwiązanie: $f(x) = 3$
   - $P_1(-3;5)$; rozwiązanie: $f(x) = 5$
   - $P_1(5;-1)$; rozwiązanie: $f(x) = -1$
   - $P_1(2;-4)$; rozwiązanie: $f(x) = -4$
   - $P_1(-3;2)$; rozwiązanie: $f(x) = 2$
2. Część druga (dwa punkty):
   - $P_1(-1;2)$; $P_2(0;5)$; rozwiązanie: $f(x) = 3x + 5$
   - $P_1(1;-2)$; $P_2(3;2)$; rozwiązanie: $f(x) = 2x - 4$
   - $P_1(-2;1)$; $P_2(-3;3)$; rozwiązanie: $f(x) = -2x - 3$
   - $P_1(-1;6)$; $P_2(1;-2)$; rozwiązanie: $f(x) = -4x + 2$
3. Część trzecia (trzy punkty):
   - $P_1(-1;5)$; $P_2(1;5)$; $P_3(0;2)$; rozwiązanie: $f(x) = 3x^2 + 0x + 2$
   - $P_1(-1;-2)$; $P_2(0;-2)$; $P_3(-2;4)$; rozwiązanie: $f(x) = 3x^2 + 3x - 2$
   - $P_1(0;2)$; $P_2(2;0)$; $P_3(-1;-3)$; rozwiązanie: $f(x) = -2x^2 + 3x + 2$
   - $P_1(-1;6)$; $P_2(-2;4)$; $P_3(-3;-2)$; rozwiązanie: $f(x) = -2x^2 - 4x + 4$

8.4.3. Przebieg zadania

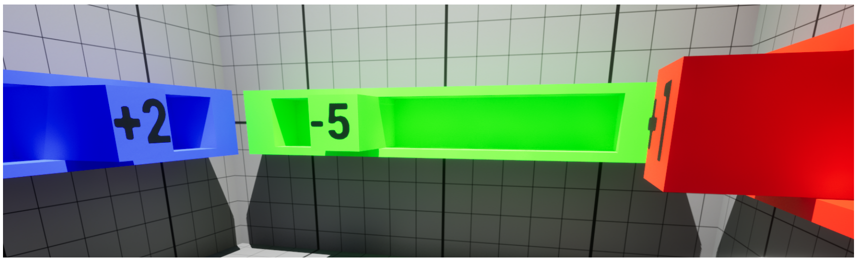
Zadanie rozpoczyna się zainicjalizowaniem i wstępną konfiguracją kanwy, na której wyświetlany będzie układ współrzędnych. Następnie główny kontroler wywołuje funkcję `Random Points`, która losowo wybiera współrzędne punktu, który gracz będzie musiał przeciąć funkcją, z wcześniej zdefiniowanego zbioru i wypisze je na podłodze pomieszczenia. Wzór funkcji $f(x) = Ax^2 + Bx + C$ (gdzie współczynniki A, B i C są domyślnie ustawione na 0) zostaje wypisany na ścianie, a jej wykres, jak i wybrany wcześniej punkt narysowany w kanwie (rys. 8.10).

W pierwszej części gracz ma do dyspozycji jeden suwak – odpowiadający za współczynnik C. Jego zadaniem jest przesunięcie wykresu funkcji w górę lub w dół tak, aby przeciął on wylosowany punkt. Po rozwiązaniu pierwszej części, główny kontroler ponownie wywołuje funkcję `Random Points`, tym razem losując współrzędne dwóch punktów, a gracz otrzymuje drugi suwak – odpowiadający za współczynnik B. Teraz jego zadaniem jest ustawienie odpowiednich wartości obu współczynników, tak aby wykres funkcji przeciął oba punkty jednocześnie. Ostatnia część jest analogiczna do poprzednich. `Random Points` losuje trzy punkty, a gracz dostaje do dyspozycji trzeci suwak – odpowiadający za współczynnik A (rys. 8.11) i musi dostosować wzór paraboli, przecinając wszystkie trzy punkty.

W momencie zakończenia wszystkich trzech części zadania, zagadka zostaje uznana za rozwiązaną.



Rysunek 8.10: Zainicjowany układ współrzędnych wraz z jednym punktem, który gracz musi przeciąć wykresem funkcji w pokoju 4



Rysunek 8.11: Wszystkie suwaki dostępne w ostatnim etapie w pokoju 4

8.5. Zadanie 5 – Geometria analityczna (Konrad Czarnecki)

8.5.1. Problemy i różnice w realizacji zadania w praktyce

Tak samo jak w przypadku zagadki 8.4 jedyną istotną zmianą zadania względem opisanego projektu w podrozdziale 7.5 było rozdzielenie zagadki czwartej i piątej na dwa oddzielne pokoje.

8.5.2. Kontroler przycisków

Przyciski wyświetlane po obu stronach kanwy dysponują własnym kontrolerem, który powiadamia kontroler główny za każdym razem, gdy któryś z nich zostanie naciśnięty. Kontroler ten obsługuje też zmiany wprowadzane przez gracza, np. jeśli gracz naciśnie przycisk odpowiadający współczynnikowi x punktu przecięcia, równemu 3, a następnie zmieni zadanie i wciśnie przycisk 4 dla współczynnika x, przycisk 3 zostanie automatycznie odznaczony. Analogicznie działa to dla przycisków współczynnika y. Liczba wyświetlana na przycisku wskazuje bezpośrednio wartość współczynnika punktu przecięcia – np. jeśli gracz chce wybrać współczynnik x równy 2, powinien wcisnąć przycisk oznaczony numerem 2.

W celu zachowania przestronności pomieszczenia, zdecydowano się zrezygnować z przycisków o wartościach ujemnych i zamiast nich wprowadzić przycisk oznaczony symbolem $\pm$, który po naciśnięciu zmienia znaki wszystkich innych przycisków w danej grupie (x lub y) na przeciwne.

8.5.3. Wybór dostępnych wzorów funkcji

Zadanie składa się z trzech części. W każdej części gracz musi wyliczyć punkt przecięcia wykresów funkcji i wcisnąć odpowiadające mu przyciski. W celu zwiększania poziomu trudności, w pierwszej części obie funkcje są funkcjami liniowymi, w drugiej jedna z nich jest funkcją liniową, a druga kwadratową, a w trzeciej obie funkcje są kwadratowe.

Losowe funkcje, których punkt przecięcia gracz ma za zadanie wyznaczyć, są wybierane z wcześniej zdefiniowanego zbioru. Zbiór ten został dobrany w taki sposób, aby funkcje zawsze przecinały się tylko w jednym miejscu (w przypadku funkcji kwadratowych drugi punkt przecięcia zawsze znajduje się w dużej odległości poza zakresem widoczności kanwy).

Lista możliwych wzorów przecinających się funkcji w każdej części zadania:

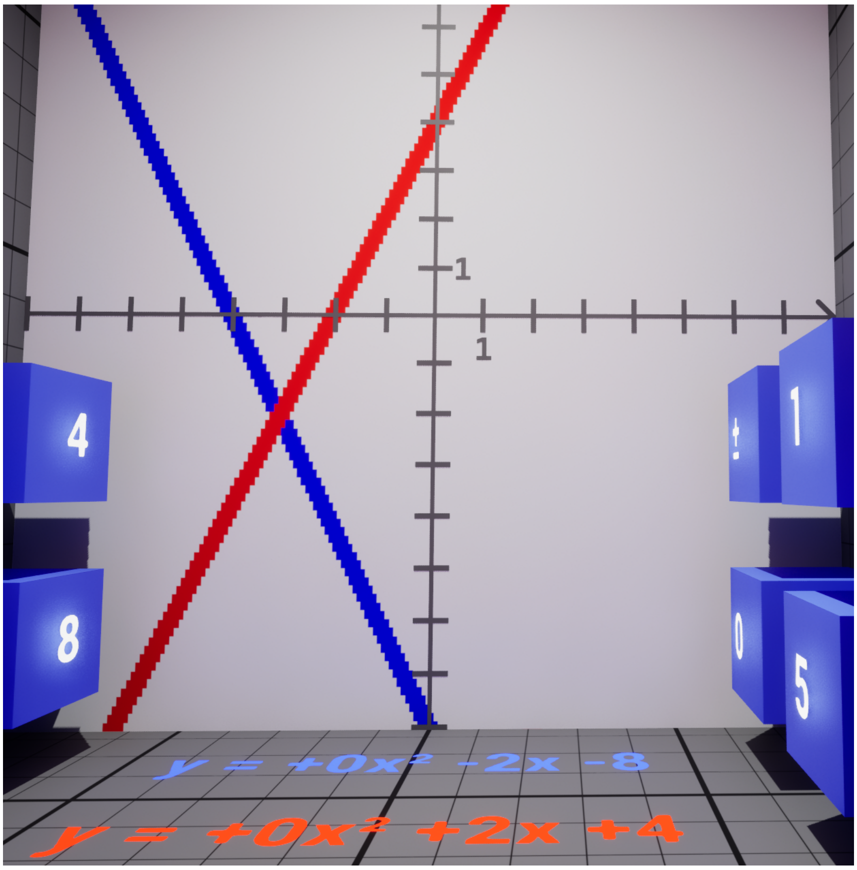
1. Część pierwsza (dwie funkcje liniowe):
   - $f_1(x) = 2x - 3$; $f_2(x) = -x + 6$; rozwiązanie: $P(3; 3)$
   - $f_1(x) = -2x - 8$; $f_2(x) = 2x + 4$; rozwiązanie: $P(-3; -2)$
   - $f_1(x) = -9x - 6$; $f_2(x) = 2x + 5$; rozwiązanie: $P(-1; 3)$
2. Część druga (jedna funkcja liniowa, jedna kwadratowa):
   - $f_1(x) = x^2 - 10x + 23$; $f_2(x) = 6x - 32$; rozwiązanie: $P(5; -2)$
   - $f_1(x) = x^2 + 10x + 29$; $f_2(x) = -6x - 26$; rozwiązanie: $P(-5; 4)$
   - $f_1(x) = x^2 - 6x + 14$; $f_2(x) = 8x - 19$; rozwiązanie: $P(3; 5)$
3. Część trzecia (dwie funkcje kwadratowe):
   - $f_1(x) = 2x^2 - 31x + 103$; $f_2(x) = x^2 - 10x + 23$; rozwiązanie: $P(5; -2)$
   - $f_1(x) = 2x^2 - 19x + 36$; $f_2(x) = x^2 - 4x + 10$; rozwiązanie: $P(2; 6)$
   - $f_1(x) = 2x^2 + 23x + 46$; $f_2(x) = x^2 + 6x + 4$; rozwiązanie: $P(-3; -5)$

8.5.4. Przebieg zadania

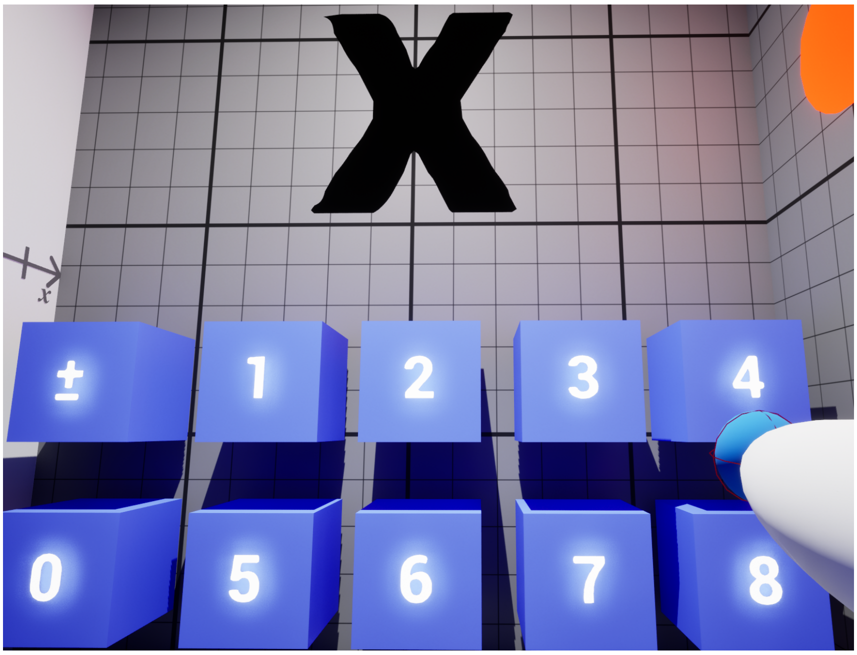
Podobnie jak w przypadku czwartej zagadki, zadanie rozpoczyna się zainicjalizowaniem i wstępną konfiguracją kanwy, na której wyświetlane będą układ współrzędnych oraz wzory funkcji. Następnie główny kontroler wywołuje funkcję `Random Points`, która losowo wybiera wzory dwóch przecinających się funkcji z wcześniej zdefiniowanego zbioru. Wzory funkcji zostają wypisane na podłodze, a ich wykresy narysowany w kanwie (rys. 8.12).

Gracz ma do dyspozycji rzędy przycisków po prawej i lewej stronie kanwy, których zadaniem jest odpowiednio modyfikacja współczynników x (rys. 8.13) i y. Za plecami gracza znajdują się trzy rozświetlone na czerwono lampy. Po rozwiązaniu każdej części zadania, jedna z lamp zmienia kolor na niebieski, a funkcja `Random Points` wybiera nowy zestaw funkcji.

W momencie zakończenia wszystkich trzech części zadania i rozświetlenia wszystkich lamp na niebiesko, zagadka zostaje uznana za rozwiązaną.



Rysunek 8.12: Zainicjowany układ współrzędnych wraz z przecinającymi się wykresami funkcji w pokoju 5



Rysunek 8.13: Przyciski odpowiedzialne za współrzędną X w pokoju 5

8.6. Zadanie 6 – Kombinatoryka (Andrii Demyshyn)

8.6.1. Różnice realizacji zadania w praktyce

W koncepcji teoretycznej sejf zawierał jedną, stałą zagadkę matematyczną. Takie rozwiązanie powodowało, że po jednorazowym rozwiązaniu zadania element ten tracił swoją funkcję edukacyjną oraz rozgrywkową. W związku z tym zdecydowano się na rozbudowanie pierwotnej koncepcji poprzez wprowadzenie wielu zagadek kombinatorycznych obsługiwanych przez jeden, uniwersalny mechanizm sejfu.

8.6.2. Zaimplementowane zagadki

Zaimplementowano dwanaście zagadek kombinatorycznych o zróżnicowanym poziomie trudności, z których każda posiada przypisany indeks. Podczas uruchomienia gry indeks zagadki jest losowo generowany z przedziału od 0 do 11, co sprawia, że gracz nie musi rozwiązywać wszystkich zagadek w jednej rozgrywce.

Następnie treść wybranej zagadki wyświetlana jest za pomocą blueprintu aktora `BP_RiddleSafe`, w którym umieszczony jest widget typu `WB_ZagadkaText`. Poprawna odpowiedź ustawiana jest jako zmienna `CurrentCode`.

1. **Zagadka 1:** „PIN to liczba wszystkich możliwych 3-cyfrowych kodów PIN, w których żadna cyfra się nie powtarza.” **Odpowiedź:** 720
2. **Zagadka 2:** „PIN to liczba wszystkich możliwych 4-cyfrowych kodów PIN, w których żadna cyfra się nie powtarza.” **Odpowiedź:** 5040
3. **Zagadka 3:** „PIN to liczba wszystkich możliwych 4-cyfrowych kodów PIN bez powtórzeń, gdzie pierwsza cyfra nie może być zerem.” **Odpowiedź:** 4536
4. **Zagadka 4:** „PIN to liczba wszystkich 4-cyfrowych kodów PIN, w których dokładnie jedna

cyfra występuje dwa razy, a pozostałe dwie cyfry są różne od niej i od siebie (np. 1213).” **Odpowiedź:** 4320

5. **Zagadka 5:** „PIN to liczba wszystkich 4-cyfrowych kodów PIN, które składają się z dokładnie dwóch różnych cyfr (obie muszą wystąpić co najmniej raz).” **Odpowiedź:** 630
6. **Zagadka 6:** „PIN to liczba wszystkich 4-cyfrowych kodów PIN będących palindromem (ABBA).” **Odpowiedź:** 100
7. **Zagadka 7:** „PIN to liczba wszystkich 6-cyfrowych kodów PIN będących palindromem (ABC-CBA).” **Odpowiedź:** 1000
8. **Zagadka 8:** „PIN to liczba wszystkich 3-cyfrowych kodów PIN, w których dokładnie jedna cyfra występuje dwa razy.” **Odpowiedź:** 270
9. **Zagadka 9:** „PIN to liczba wszystkich 4-cyfrowych kodów PIN, w których dokładnie dwie cyfry są parzyste.” **Odpowiedź:** 3750
10. **Zagadka 10:** „PIN to liczba wszystkich 4-cyfrowych kodów PIN, w których suma cyfr wynosi dokładnie 10.” **Odpowiedź:** 282
11. **Zagadka 11:** „PIN to liczba wszystkich 4-cyfrowych kodów PIN, w których cyfry są ustawione w ściśle rosnącym porządku.” **Odpowiedź:** 210
12. **Zagadka 12:** „PIN to liczba wszystkich 5-cyfrowych kodów PIN, w których cyfry są ustawione w ściśle malejącym porządku.” **Odpowiedź:** 252

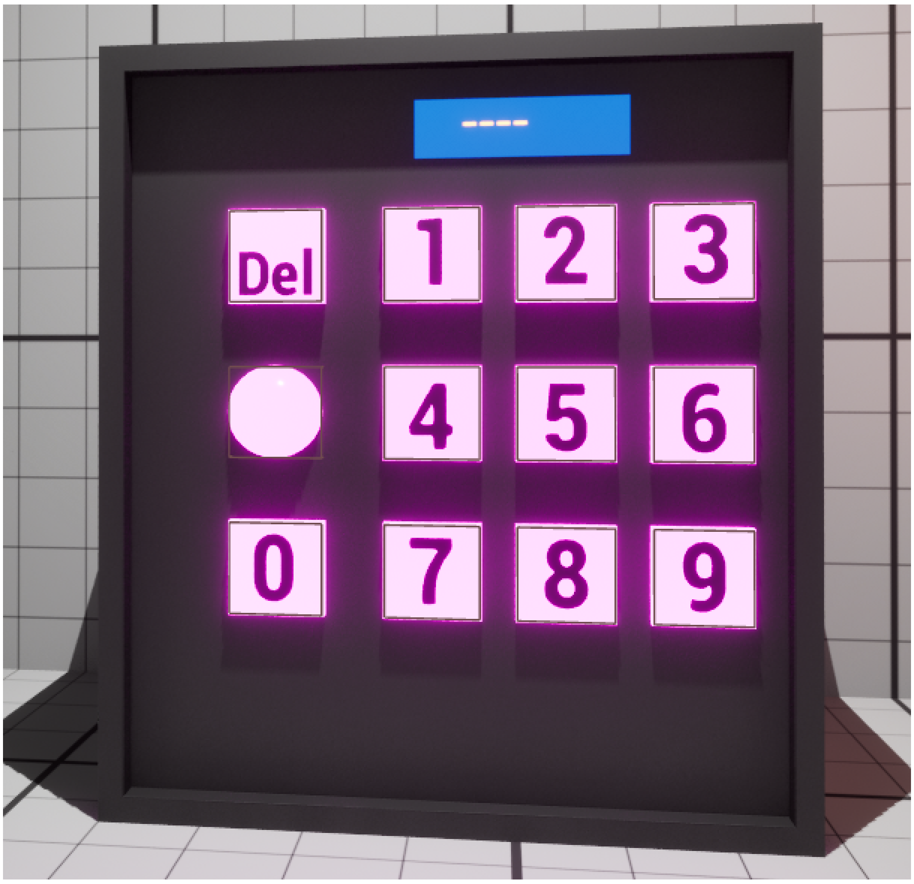
8.6.3. Sejf

Sejf został zrealizowany jako obiekt typu Blueprint Actor. Składa się on z komponentów `StaticMesh` tworzących obudowę sejfu oraz klawiaturę numeryczną, a także z wyświetlacza w postaci widgetu `WBP_SafeDisplay`, który prezentuje wprowadzane cyfry oraz komunikaty zwrotne.

Klawiatura sejfu zawiera (rys. 8.14):

- przyciski numeryczne od 0 do 9,
- przycisk `DEL`, umożliwiający usunięcie ostatnio wprowadzonej cyfry,
- przycisk zatwierdzenia wykonany w formie sferycznego obiektu 3D.

Gracz wchodzi w interakcję z sejfem poprzez naciskanie przycisków. Każdy przycisk sejfu składa się z komponentu wizualnego typu `StaticMesh` oraz komponentu `BoxCollision`. Po wciśnięciu przycisku odtwarzana jest animacja oraz wywoływane jest zdarzenie przekazujące informację o wciśniętym elemencie do głównej logiki sejfu.



Rysunek 8.14: Wygląd sejfu

8.6.4. Zakończenie zadania

Po wciśnięciu przycisku zatwierdzenia rozpoczyna się proces weryfikacji wprowadzonego kodu z wartością zmiennej `CurrentCode`. Jeżeli wartości są zgodne, sejf zostaje otwarty. W przypadku wprowadzenia błędnego kodu sejf pozostaje zamknięty, a na wyświetlaczu pojawia się komunikat „ERROR”. Gracz może ponowić próbę rozwiązania zagadki.

8.6.5. Podsumowanie

Zastosowanie jednego, rozszerzalnego mechanizmu sejfu umożliwiło implementację wielu zagadek kombinatorycznych w spójnym środowisku interaktywnym. Takie rozwiązanie zwiększa regrywalność gry oraz poprawia jej walory edukacyjne.

8.7. Zadanie 7 – Liczby rzeczywiste i działania na zbiorach liczbowych (Jan Walczak)

8.7.1. Problemy i różnice w realizacji zadania w praktyce

Podczas implementacji zadania, zgodnie z projektem opisanym w podrozdziale 7.7, należało wprowadzić względem niego kilka poprawek. Pierwotny mechanizm zakładał, że zamek umieszczony na skrzyni będzie wyposażony w pola, w które uczeń wpisywałby odpowiednie cyfry – liczby elementów danego podzbioru. W praktyce okazało się to niemożliwe. Maksymalna liczba jednocyfrowa to 9, tym samym zadanie jest ograniczone do wyświetlania maksymalnie 9 liczb rzeczywistych w pokoju.

Zamysł został zmieniony i zamiast kłódki postanowiono umieścić elektroniczny zamek, wyposażony w klawiaturę. W ten sposób otrzymano możliwość umieszczenia więcej niż 9 liczb w pokoju. Uczeń staje również przed dodatkowym wyzwaniem – wywnioskować kolejność wpisywanych liczb zgodnie z podpowiedzią, umieszczoną na ścianie (patrz rysunek 8.15).

Rysunek 8.15: Zrzut ekranu – skrzynia z zamkiem oraz podpowiedź na ścianie dla zadania 7.

Zmienione zostały również liczby, które są rozmieszczone w pokoju. Aktualnie pojawiają się liczby: *15; π; 4,5; √2; ¼; -3; 1; -7; ⅓ ; 9*. Tym samym podzbiory prezentują się następująco (kolejność zgodna z podpowiedzią na rysunku 8.15):

- *N* (naturalne): *15; 1; 9* – 3 liczby,
- *Z* (całkowite): *15; 1; 9; -3; -7* – 5 liczb,
- *R* (rzeczywiste): wszystkie liczby – 10 liczb,
- *Q* (wymierne): *15; 1; 9; -3; -7; 4,5; ¼; ⅓* – 8 liczb,
- *R\Q* (niewymierne): *π; √2* – 2 liczby.

Tym samym ciąg cyfr prowadzący do otwarcia sejfu to *351082*.

Ostatnią rzeczą, która została zmieniona jest koncepcja drugiej części zadania. Mechanizm obracania bloczków został uznany za niepotrzebny, ponieważ wybrane relacje są jednoznaczne – są czytane zawsze od lewej do prawej strony. Niepoprawny obrót bloczka od razu sugeruje błędną odpowiedź.

Spośród wszystkich relacji zostały wybrane dwie:

- $N \subset Z \subset R$,
- $R \backslash Q \cap Q = \varnothing$,

Zostały one wybrane przez swoją charakterystykę. Pierwsza z nich pokazuje, jak kolejne zbiory zawierają się w sobie nawzajem, a druga pokazuje rozłączność zbiorów liczb wymiernych i niewymiernych. Gracz ma do dyspozycji 6 bloczków:

- dwa bloczki z symbolem $\subset$,
- dwa bloczki z symbolem $\varnothing$,

- dwa bloczki z symbolem ∩

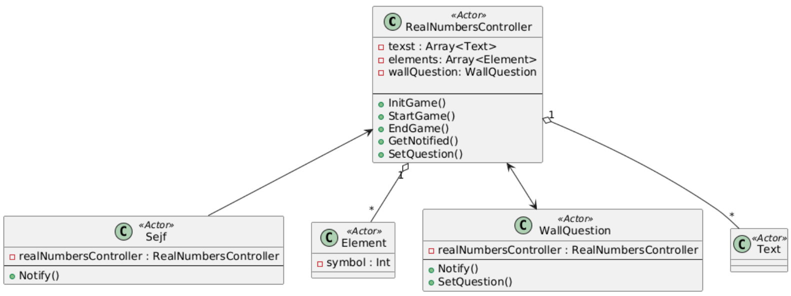
Bloczki mają mu służyć do uzupełnienia dwóch relacji wymienionych wyżej. Po ułożeniu relacji, poprawnie bądź nie, bloczki pozostają do dyspozycji gracza. W relacji elementami stałymi są „=“ i litery reprezentujące zbiory liczb (nie można ich przestawiać).

8.7.2. *Implementacja zadania*

Został zaimplementowany jeden, główny kontroler `RealNumbersController`. Zarządza on przebiegiem zadania, położeniem elementów na scenie oraz wyświetlaniem tekstu. Zawiera odpowiednie referencje do komponentów i odpowiada za poprawną inicjalizację. Relacje między nim, a pozostałymi komponentami zostały zaprezentowane na rysunku 8.16.

Komponent `Element` jest bloczkiem z odpowiednim symbolem relacji, `Sejf` jest skrzynią, która przetrzymuje bloczki, a `WallQuestion` to płaska tablica (ścianka), na której wyświetlane są pytania w drugiej części zadania.

`Sejf` zawiera jednostronną referencję do głównego kontrolera. Kontroler może zostać przez niego powiadomiony, że gracz poprawnie wpisał kod. W ten sposób kontroler może sterować dalszym przebiegiem zadania. `WallQuestion` zawiera dwustronną referencję do głównego kontrolera. Mechanizm powiadomień działa tutaj podobnie, ale kontroler może odpowiadać na otrzymywane powiadomienia, poprzez wywoływanie odpowiednich funkcji.



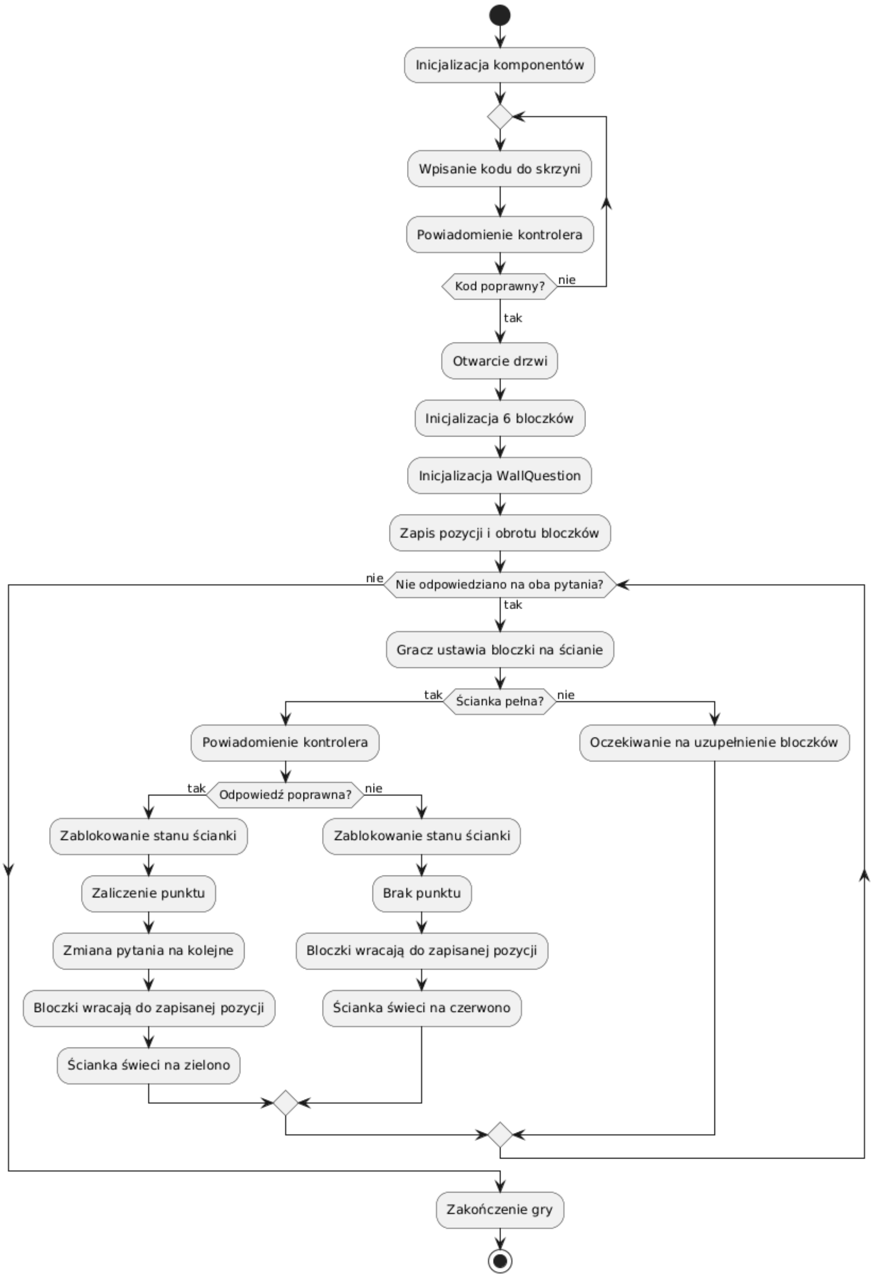
Rysunek 8.16: Diagram UML zawierający najważniejsze elementy dla zadania 7 – liczby rzeczywiste i działania na zbiorach liczbowych.

8.7.3. *Przebieg zadania*

Zadanie rozpoczyna się od pierwszej części – wpisania kodu do skrzyni. Kontroler inicjalizuje komponenty, a na ścianach i na skrzyni zostają wyświetlone liczby opisane w 8.7.1. Gracz powinien policzyć ile liczb, z jakiego zbioru, jest wyświetlanych w pokoju i poprawnie wpisać kod. W doborze kolejności wpisywanych liczb pomaga mu podpowiedź umieszczona na ścianie. Po poprawnym wpisaniu kodu, kontroler główny zostaje powiadomiony przez instancję skrzyni, że zadanie może przejść do następnego etapu. Drzwi się otwierają a kontroler inicjalizuje 6 bloczków, opisanych w 8.7.1 i ściankę z pytaniami `WallQuestion`. Zapisuje pozycję oraz obrót bloczków w świecie gry, tak aby możliwy był powrót do stanu początkowego.

Ścianka wykrywa kolizję z bloczkami, które gracz na niej umieszcza. W momencie kiedy zostaje uzupełniona w całości, tj. bloczki wypełniają miejsca na odpowiedzi, powiadamia kontroler o rozwiązaniu zadania. Kontroler decyduje, czy odpowiedź gracza jest poprawna. W przypadku gdy jest, to przesyła do `WallQuestion` powiadomienie o zmianie pytania na kolejne i zalicza punkt. Jeżeli odpowiedź nie jest poprawna, to również powiadamia o tym ściankę, ale nie zalicza punktu. W obu przypadkach bloczki wracają na wcześniej zapisaną pozycję. W zależności od poprawności udzielonej odpowiedzi, ścianka rozświetla się kolorem zielonym lub czerwonym, tak żeby gracz miał pewność, że udzielona odpowiedź jest poprawna. Podczas sprawdzania stanu ścianki, tj. ustawienia bloczków przez gracza, pytanie zostaje zablokowane, a gracz nie może modyfikować swojej odpowiedzi (dokładać lub zabierać bloczków).

Kiedy gracz odpowie na oba pytania poprawnie, poziom zostaje uznany za zaliczony. Przepływ został zaprezentowany w postaci diagramu na rysunku 8.17.



Rysunek 8.17: Diagram przepływu dla zadania 7 – liczby rzeczywiste i działania na zbiorach liczbowych.

8.7.4. Implementacja mechanizmu uzupełniania ścianki

Ścianka, czyli `WallQuestion`, wykrywa kolizję z bloczkami, czyli `Element`, w dwóch miejscach, w których będą one docelowo umieszczone. Kiedy gracz przytrzyma bloczek nad jednym z dwóch wskazanych miejsc, uruchamiana jest odpowiednia funkcja `StartCollision()`. W zależności od tego gdzie został umieszczony bloczek, sprawdzany jest aktualny stan ścianki – czy w danym miejscu gracz umieścił już bloczek. Jeśli tak, to ścianka nie pozwoli mu umieścić w tym miejscu kolejnego bloczka, dopóki nie wyjmie poprzedniego. Dzieje się tak za sprawą dwóch zmiennych wartości logicznych: `ActorHovered` i `ActorLockedIn`. Oznaczają one kolejno: stan, w którym gracz trzyma bloczek nad ścianką i stan, w którym gracz umieścił bloczek na ściance. W przypadku gdy `ActorLockedIn` jest ustawione na wartość `0`, to zapisywana jest referencja bloczku, który został właśnie umieszczony.

Dodatkowo informacja o tym, że gracz puścił bloczek, czyli zamierza go umieścić w danym miejscu, jest odpowiednio zapisywana w instancji bloczka. Kiedy bloczek znajduje się w ręce gracza otrzymuje sygnaturę `held`. Kiedy gracz go puści, sygnatura jest odbierana. Wyżej wymienione wartości logiczne są ustawiane dopiero wtedy, kiedy sygnatura najpierw istniała (w instancji bloczka), a potem została z niej usunięta.

W przypadku, gdy gracz chce usunąć bloczek ze ścianki, sytuacja jest analogiczna. Tym razem jednak, sprawdzamy czy bloczek, który zabiera gracz, jest tym znajdującym się na ściance, poprzez porównanie referencji.

8.8. Zadanie 8 – Znaki funkcji trygonometrycznych (Konrad Czarnecki)

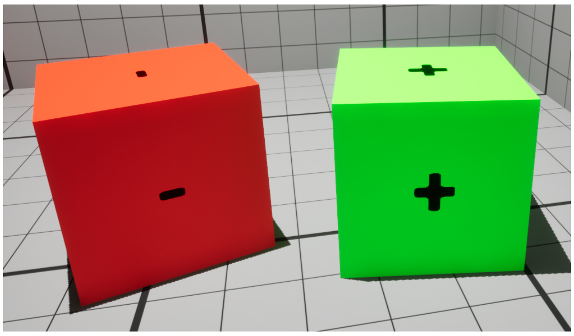
8.8.1. Problemy i różnice w realizacji zadania w praktyce

Podczas implementacji zadania zgodnie z opisem teoretycznym zawartym w podrozdziale 7.8 wprowadzono zmiany poprawiające estetykę i grywalność zagadki. Pierwotnie wszystkie cztery układy jednostkowe (oznaczone $sin(x)$, $cos(x)$, $tg(x)$ i $ctg(x)$) miały znajdować się na jednej ścianie, ale w celu uzyskania większej przejrzystości każdy z nich umieszczono na oddzielnej. Dodano ruchome sześciany w dwóch wariantach: zielone – oznaczone symbolami „+” oraz czerwone – oznaczone symbolami „–” (rys. 8.18). Przyciski do zmiany znaku funkcji umieszczone w czterech ćwiatrtkach układów jednostkowych zostały zastąpione lukami (rys. 8.19), które gracz musi wypełnić wyżej wspomniamymi sześcianami. Taka zmiana zwiększa poziom interakcji gracza z zagadką, co bezpośrednio przekłada się na wyższy poziom zaangażowania oraz lepszą czytelność mechanik zadania.

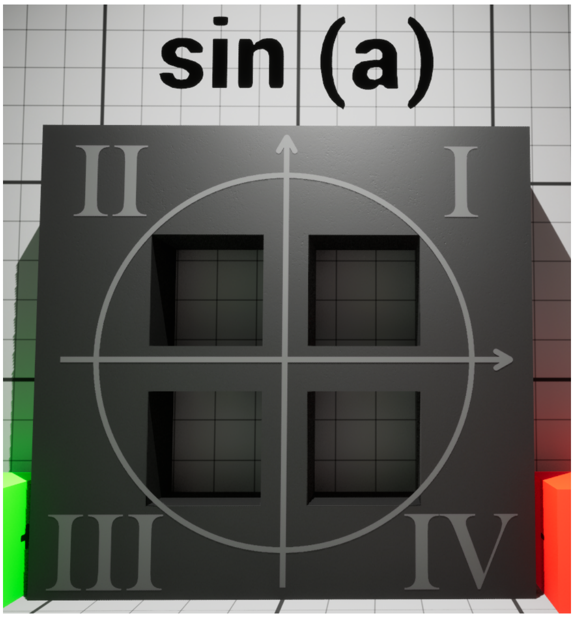
Ze względu na ograniczony rozmiar jaskini tylko jeden układ jednostkowy jest widoczny w danym czasie. Po wypełnieniu jego ćwiartek sześcianami opatrzonymi odpowiednimi symbolami, układ jednostkowy chowa się, a zza innej ściany płynnie wysuwa się następny – ich kolejność jest losowa. W związku z tym liczba sześcianów z symbolami dostępnych w jednym czasie w pomieszczeniu mogła zostać ograniczona do czterech – dwa oznaczone „+” i dwa oznaczone „–”. Pozwala to rozwiązać każdy poszczególny układ jednostkowy (rys. 7.5).

8.8.2. Implementacja sześcianów

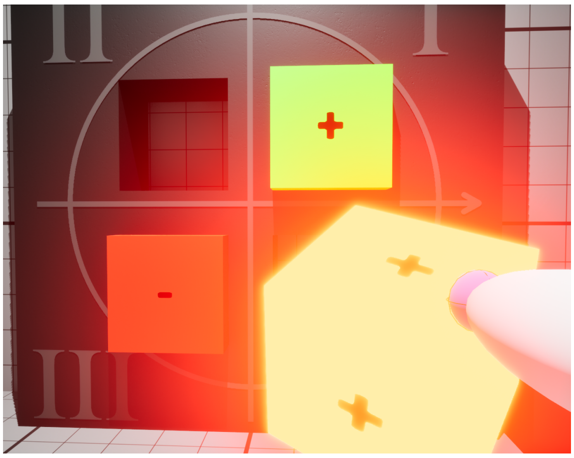
Sześciany dysponują własnym kontrolerem, który powiadamia kontroler główny za każdym razem, gdy któryś sześcian zostaje umieszczony w luce. Gracz może również wyjąć wcześniej umieszczony sześcian z luki i włożyć go w inną. Ponadto kontroler zapobiega umieszczeniu dwóch



Rysunek 8.18: Ruchome sześciany w pokoju 8



Rysunek 8.19: Układ jednostkowy w pokoju 8



Rysunek 8.20: Sześcian po wykryciu luki w pokoju 8

sześcianów w tej samej luce, a także wyrzuceniu ich poza obszar gry. W takim wypadku sześcian automatycznie pojawia się na środku pokoju, zapewniając ciągłość zagadki.

Kiedy gracz zbliży się do luki w układzie jednostkowym, trzymając sześcian, kolor sześcianu zmieni się na jaskrawo żółty, informując gracza, że luka została wykryta (rys. 8.20). Jeśli w takim momencie gracz puści sześcian, uruchomi się płynna animacja, w której ten automatycznie obróci się do odpowiedniego kąta i umieści w luce.

8.8.3. Przebieg zadania

Zadanie rozpoczyna się wywołaniem funkcji `Random Order` przez główny kontroler. Funkcja ta w losowy sposób ustala kolejność czterech układów jednostkowych funkcji trygonometrycznych, które gracz będzie musiał rozwiązać. Następnie wykonuje się funkcja `Reset Boxes Position`, która umieszcza ruchome sześciany w rogach pomieszczenia, również w losowy sposób.

Po tej wstępnej konfiguracji rozpoczyna się właściwa część zadania. Pierwszy układ jednostkowy wyłania się zza ściany, a gracz podnosi sześciany i umieszcza je w odpowiednich lukach. Za każdym razem wywoływana jest funkcja `Box Inside` informująca główny kontroler, że sześcian został umieszczony wewnątrz układu i sprawdzająca, czy położenie wszystkich sześcianów jest poprawne. Kiedy to nastąpi układ jednostkowy chowa się, nazwa funkcji trygonometrycznej nad nim rozświetla się na zielono, dając graczowi do zrozumienia, że element zagadki został rozwiązany poprawnie, sześciany cofają się do swoich pozycji startowych funkcją `Reset Boxes Position`, a następnie pojawia się kolejny układ jednostkowy do rozwiązania.

Kiedy gracz wypełni wszystkie układy poprawnie, zagadka zostaje uznana za rozwiązaną.

8.9. Zadanie 9 – Ciągi liczbowe (Jan Walczak)

8.9.1. Problemy i różnice w realizacji zadania w praktyce

W przeciwieństwie do zadań opisanych w podrozdziałach 8.2 i 8.7 w tym zadaniu nie pojawiło się dużo rozbieżności między projektem, a praktyczną implementacją.

Miejsce, w którym wprowadzono modyfikację, to warunki zakończenia gry. Zadanie, przez swój projekt, stawia bardziej na szybką analizę, niż na długie zastanawianie się nad odpowiedzią. Jest bezpośrednio inspirowane prawdziwą grą, w której największą wartością, wyróżniającą gracza, jest jego czas reakcji. Z tego powodu zdecydowano się na usunięcie limitu czasowego. Gra przyspiesza liniowo, razem z postępem, tj. liczbą zestrzelonych przez gracza kaczek.

Dodatkowo zwiększono liczbę żyć, tj. możliwości na pomyłkę gracza do 4 zamiast 3. Jest to spowodowane wprowadzeniem dodatkowej mechaniki gry – jeśli kaczka wyleci poza zasięg gracza (za ścianę), gracz traci życie. Gra kończy się jedynie w momencie utraty wszystkich żyć – co musi nastąpić, ponieważ prędkość kaczek będzie rosła tak długo, jak toczy się rozgrywka.

8.9.2. Implementacja pistoletu laserowego

Implementacja pistoletu laserowego była kluczowym punktem, potrzebnym do stworzenia całego poziomu.

Aby poprawnie zaimplementować taki laser potrzebne są trzy wartości:

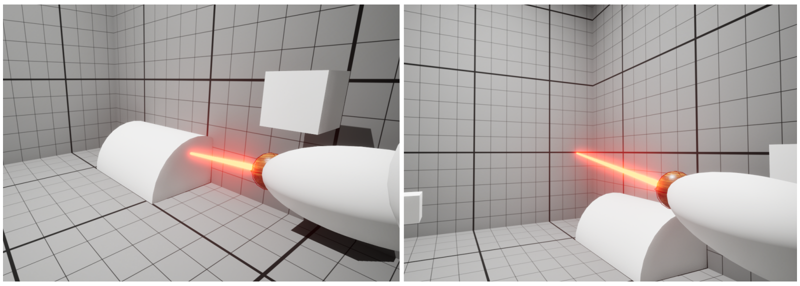
- punkt początkowy, od którego zaczynamy rysowanie lasera,
- punkt końcowy, do którego laser będzie rysowany,
- wektor kierunkowy, według którego laser będzie rysowany.

Za punkt początkowy został obrany koniec ręki gracza, tj. kontrolera, którym steruje. Z ustawienia kontrolera w świecie gry można również wyznaczyć wektor kierunkowy. Należy pobrać z kontrolera wartość tego wektora, przy użyciu systemowej funkcji Unreal Engine `Get Forward Vector`. Należy zapisać ten wektor, razem z punktem początkowym. Obliczenie punktu końcowego jest nieco trudniejsze. Polega na początkowym pomnożeniu wektora kierunkowego przez dużą liczbę (1000), a następnie na dodaniu wyniku tego działania do współrzędnych punktu początkowego. Tak obliczony punkt należało chwilowo zapisać jako punkt końcowy.

Następnie, przy użyciu systemowej funkcji `Line Trace by Channel`, należało wyznaczyć punkt, w którym linia, poprowadzona od punktu początkowego do końcowego, przecina się ze światem przedstawionym w grze. Funkcja ta zwraca wynik w postaci wartości typu struktury `Hit Result`, który można rozbić na składowe i pozyskać z niego `Impact Point`, czyli rzeczywisty punkt końcowy oraz `Distance`, czyli wartość zmiennoprzecinkową, reprezentującą długość wyznaczonej linii.

Dzięki obliczonym wartościom można narysować laser, przebiegający od kontrolera gracza, do pierwszego napotkanego obiektu w grze. W tym celu kontroler ma na stałe przypisany obiekt `Static Mesh`, z siatką statyczną w kształcie cylindra, który jest rozciągany zgodnie z obliczonymi wartościami. Nałożony jest na niego materiał, emitujący czerwone światło. Na końcu rozciągniętej siatki statycznej umieszczona jest niewidzialna kulka, z włączonym systemem kolizji. Dzięki niej można wykrywać, w jaki komponent na scenie aktualnie celuje gracz.

Na rysunku 8.21 widać, że laser faktycznie wykrywa otoczenie, w które celuje gracz i zmienia odpowiednio swoją długość. Zmianie ulega również kierunek, w którym celuje gracz.



Rysunek 8.21: Porównanie rysowania lasera w przypadku kontaktu z przykładowym elementem otoczenia i ścianą.

8.9.3. Implementacja zadania

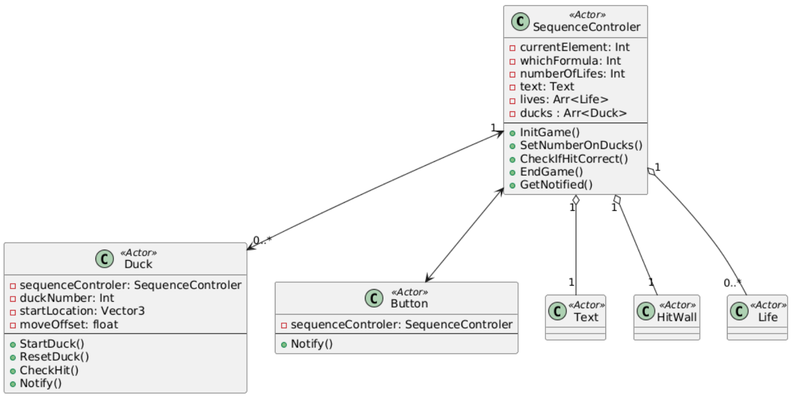
Zaimplementowany został jeden, główny kontroler `SequenceControler`. Jego zadaniem jest inicjalizacja gry, zarządzanie przebiegiem gry, wyświetlanie tekstu i zarządzanie instancjami kaczek. Kontroler wybiera pseudolosowo jeden z trzech dostępnych wzorów, na kolejny wyraz ciągu arytmetycznego, według którego będą obliczane kolejne elementy. Dostępne są wzory:

- $a_n = 3n + 4$
- $a_n = 2n + 3$
- $a_n = 5n + 7$

Komponent `Duck` implementuje funkcjonalności związane ze sterowaniem pojedynczej kaczki. Zawiera dwustronną referencję, do obiektu głównego kontrolera i przesyła powiadomienia, kiedy zostanie zestrzelony przez gracza, przy użyciu pistoletu laserowego (rys. 8.22). W instancji kaczki zapisywana jest pozycja początkowa, numer, który aktualnie jest na niej wyświetlany i wartość zmiennoprzecinkowa, czyli droga, którą pokona kaczka w każdej jednostce czasu w grze.

Gracz zostaje również zapoznany z zasadami działania gry. Kiedy jest gotowy do rozpoczęcia gry naciska przycisk „Start“. Przycisk również posiada dwustronną referencję do instancji kontrolera.

W zagadce występują jeszcze dwa komponenty: `HitWall` i `Life`. Główny kontroler zawiera tablicę referencji do 4 instancji obiektów `Life`. Reprezentują ilość pozostałych żyć gracza i nie pełnią w logice działania programu żadnej istotnej roli, poza tą wizualną. `HitWall` jest obiektem niewidocznym, znajdującym się poza zasięgiem gracza. Lecące kaczki będą wykrywać ten obiekt i sygnalizować to głównemu kontrolerowi zderzenie tak, jakby zostały trafione. Jeżeli kaczka uderzy w ten komponent, wyśle specjalny typ powiadomienia, mówiący o tym, że graczowi nie udało się jej zestrzelić. Jeśli kaczka zawierała poprawny wyraz ciągu to kontroler odczyta to jako błąd gracza i odejmie mu jedno życie. W takim przypadku, jeżeli gracz ma jeszcze pozostałe życia, wyświetli się kolejny wyraz ciągu, a kaczki zaczną lecieć od początku.



Rysunek 8.22: Diagram UML zawierający najważniejsze elementy dla zadania 9 – Ciągi liczbowe.

8.9.4. *Przebieg zadania*

Zadanie rozpoczyna się od inicjalizacji komponentów przez główny kontroler. Początkowo graczowi zostaje wyświetlona instrukcja do zadania. Kiedy gracz będzie gotowy może rozpocząć grę naciskając przycisk „Start“. Kontroler zostaje w takiej sytuacji powiadomiony przez instancję przycisku i rozpoczyna grę. W sposób pseudolosowy zostaje wybrany jeden z trzech wzorów na kolejny *n-ty* wyraz ciągu arytmetycznego. Kontroler ustawia również poprawny wyraz ciągu, początkowo $n = 0$, na jednej z trzech kaczek. Pozostałe kaczki otrzymują wyrazy większe lub mniejsze od poprawnego wyrazu, również w sposób pseudolosowy. Dzieje się to wedle zasady, że wybrane wyrazy mogą być odległe jedynie o 2 jednostki na osi współrzędnych od poprawnego wyrazu ($a_n \pm 2$). Przepływ tego mechanizmu można zobaczyć na rysunku 8.23.

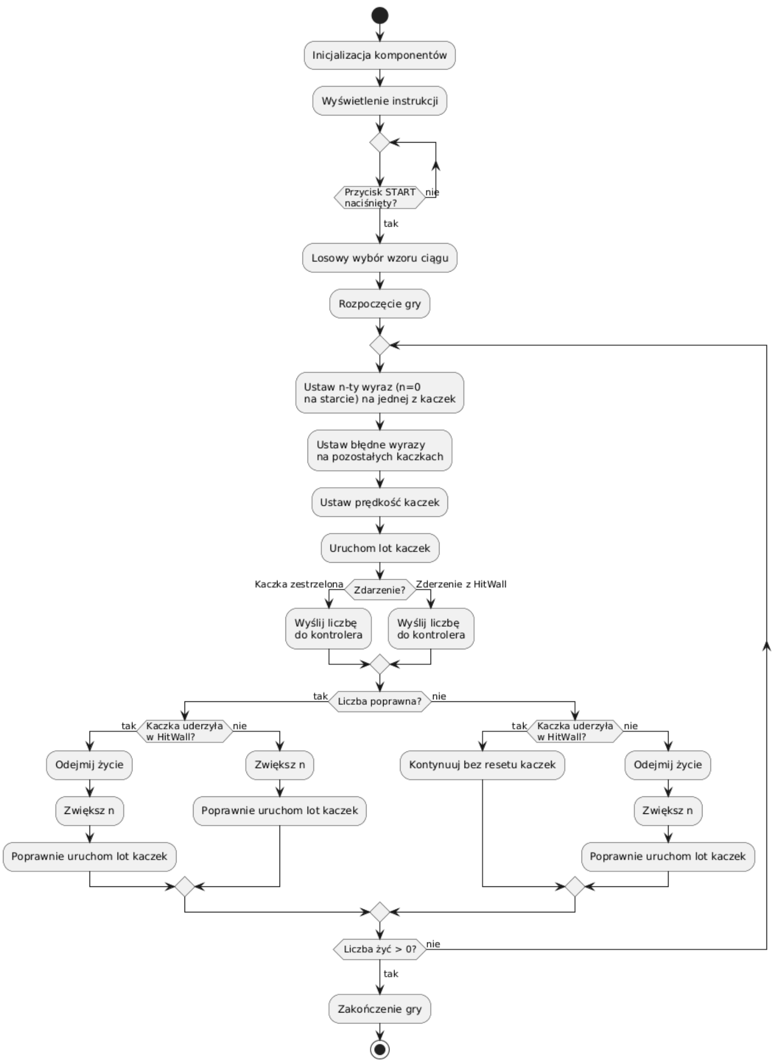
Kaczki pokonują stałą jednostkę odległości, w każdej jednostce czasu wykonywania się programu, wyrażoną za pomocą liczby zmiennoprzecinkowej. Jednostka odległości jest tym większa, im większy jest aktualny numer wyrazu ciągu arytmetycznego. Oznacza to, że będą przyspieszać razem z przebiegiem gry. Do tak obliczonej stałej wartości, dodawana jest pseudolosowa wartość zmiennoprzecinkowa wybrana pseudolosowo z przedziału $[0,5; 0,7]$.

Kaczki poruszają się od lewej do prawej strony ekranu, aż do zderzenia się z instancją obiektu `HitWall`. Kaczki mogą zostać zestrzelone przez gracza, zanim zderzą się z wyżej wymienionym obiektem. W obu przypadkach do kontrolera wysyłane jest powiadomienie odpowiedniego rodzaju: kaczka została zestrzelona lub kaczka zderzyła się z obiektem `HitWall`. W powiadomieniu przekazana jest również liczba całkowita, która znajdowała się na danej kaczce.

Kontroler po otrzymaniu w powiadomieniu liczby oraz informacji, czy dana kaczka została zestrzelona, podejmuje decyzję:

1. Jeśli liczba jest poprawnym wyrazem ciągu – zalicza punkt i ustawia ponownie kaczki. Zwiększa też numer wyrazu ciągu.
2. Jeśli liczba nie jest poprawna – odejmuje jedno życie, ale nie ustawia ponownie kaczek.
3. Jeśli kaczka, przechowująca poprawny wyraz ciągu, uderzy w obiekt `HitWall` (wyleci poza zasięg gracza) – odejmuje jedno życie i ponownie ustawia kaczki. Zwiększa też numer wyrazu ciągu.

Ponowne ustawienie kaczek polega na obliczeniu kolejnego wyrazu ciągu arytmetycznego, ustawieniu go na jednej z nich, ustawieniu dwóm pozostałym kaczkom błędnego wyrazu oraz na cofnięciu ich do stanu początkowego. Najgorszym scenariuszem dla gracza jest niepoprawne zestrzelenie dwóch z trzech kaczek i pozwolenie, aby kaczka zawierająca poprawny numer uderzyła w obiekt `HitWall`. W takim przypadku gracz straci aż 3 życia. Gra kończy się w momencie, gdy gracz straci wszystkie 4 życia.



Rysunek 8.23: Diagram przepływu dla zadania 9 – Ciągi liczbowe

8.10. Zadanie 10 – Prawdopodobieństwo (Andrii Demyshyn)

8.10.1. Praktyczna realizacja zadania

Praktyczna realizacja zadania nie różni się zasadniczo od tego, co zostało zaplanowane w projekcie, a niewielkie zmiany zostały opisane poniżej.

8.10.2. Design pokoju

Modele trójwymiarowe wykorzystane w projekcie zostały pobrane z platformy Sketchfab[1] na licencji Creative Commons i dostosowane do wymagań silnika Unreal Engine.

Początkowo planowano, że zagadka zostanie zrealizowana w stylistyce filmu „Matrix". W tym celu pobrano i dodano do projektu fotel oraz stół w odpowiednim stylu. Ze względu na brak bezpłatnego modelu postaci Morfeusza zdecydowano się na wykorzystanie podobnej postaci, której tekstury zostały zmodyfikowane tak, aby odpowiadały estetyce filmu. Efekt modyfikacji postaci przed i po zmianach przedstawiono na rys. 8.24.

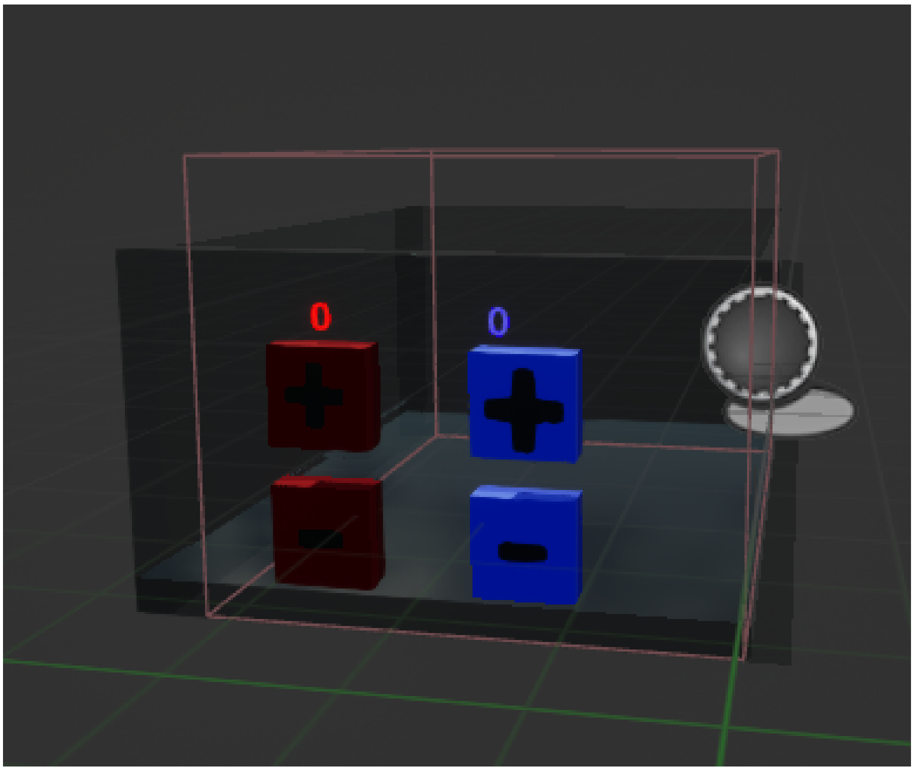
Rysunek 8.24: Postać przed i po modyfikacji

Zdecydowano się również na stylizację pokoju w estetyce „Matrixa". W tym celu stworzono materiał z neonowo-zielonymi symbolami na czarnym tle, który następnie zastosowano na obiektach ścian. Aby zwiększyć immersję gracza, dodano animację ruchu tekstury, dzięki czemu ściany sprawiają wrażenie dynamicznie poruszających się danych.

8.10.3. Obiekt sceny

Na scenie, oprócz stołu, krzesła oraz postaci Morfeusza, gracz widzi również tekst zagadki wyświetlany nad postacią, dwa pojemniki oraz zestaw 100 tabletek. Wygląd kontenerów oraz ich interfejs użytkownika przedstawiono na rys. 8.25.

[1] `https://sketchfab.com`



Rysunek 8.25: Wygląd kontenera

Tabletki – BP_Pil Tabletki zostały zaimplementowane jako obiekty typu Blueprint Actor BP_Pil. Każda tabletka posiada komponent wizualny StaticMesh, komponent kolizji typu BoxCollision, materiał w kolorze czerwonym lub niebieskim oraz odpowiedni identyfikator.

Przycisk sprawdzania – BP_CheckButton Na scenie znajduje się przycisk sprawdzania zaimplementowany jako obiekt typu Blueprint Actor BP_CheckButton. Składa się on z komponentu StaticMesh w formie sfery oraz komponentu kolizji typu BoxCollision.

Po spełnieniu warunków wstępnych obliczane jest prawdopodobieństwo wylosowania czerwonej pigułki według wzoru:

$$P = \left(\frac{R_A}{N_A} + \frac{R_B}{N_B} \right) \cdot \frac{1}{2} \cdot 100\%,$$

gdzie:

- R_A, R_B – liczba czerwonych pigułek w obu pojemnikach,
- N_A, N_B – liczba wszystkich pigułek w obu pojemnikach.

Jeżeli obliczone prawdopodobieństwo jest mniejsze niż 74,75%, gracz otrzymuje komunikat zwrotny i może ponowić próbę. W przeciwnym przypadku zadanie zostaje zaliczone.

8.11. Zadanie 11 – Optymalizacja i rachunek różniczkowy (Andrii Demyshyn)

8.11.1. Problemy i różnice w realizacji zadania w praktyce

W projekcie zaplanowano, że gracz będzie łączył początek i koniec formuły pochodnych, jednak podczas realizacji zdecydowano, że takie rozwiązanie będzie zbyt łatwe. W związku z tym podjęto decyzję o wykorzystaniu rzeczywistych funkcji oraz wyników ich pochodnych, które gracz będzie ze sobą łączył, aby podnieść poziom trudności zadania.

Przy takim podejściu konieczne było odpowiednie dobranie funkcji w taki sposób, aby zapobiec konfliktom, w których wynik pochodnej mógłby być jednocześnie początkiem innej funkcji. Przykładowo funkcja $\ln(x)$ ma pochodną $\frac{1}{x}$, a funkcja $\frac{1}{x}$ ma pochodną $-\frac{1}{x^2}$. W takim przypadku wyrażenie $\frac{1}{x}$ mogłoby występować zarówno jako początek, jak i jako koniec pary, co mogłoby prowadzić do konfliktów w rozgrywce.

Aby uniknąć takich sytuacji, wybrano następujące funkcje oraz ich pochodne:

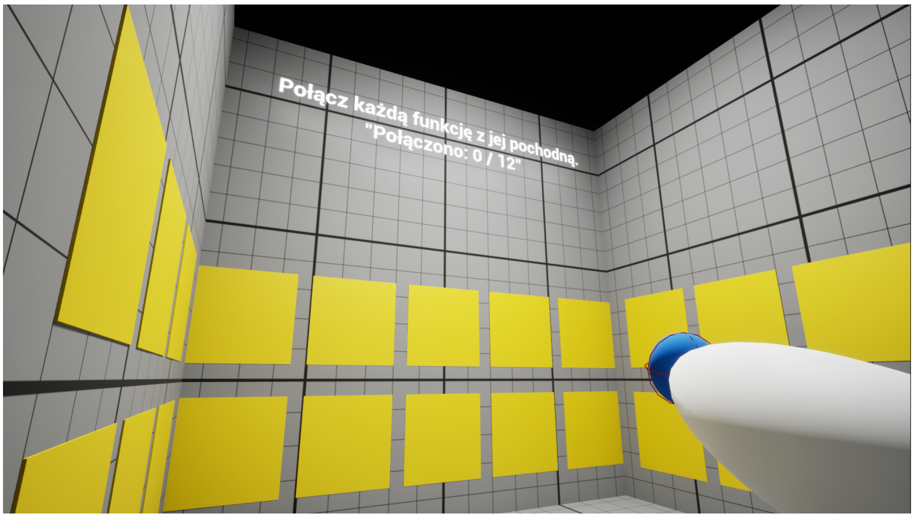
- $2x \to 2$
- $x^3 \to 3x^2$
- $\sqrt{x} \to \frac{1}{2\sqrt{x}}$
- $e^x \to e^x$
- $a^x \to a^x \cdot \ln(a)$
- $\log_a x \to \frac{1}{x \cdot \ln a}$
- $5x^2 \to 10x$
- $2^x \to 2^x \cdot \ln 2$
- $\ln(x^2) \to \frac{2}{x}$
- $\log_{10} x \to \frac{1}{x \cdot \ln 10}$
- $\frac{1}{3x} \to -\frac{1}{3x^2}$
- $\log_5\left(\frac{x}{3}\right) \to \frac{1}{x \cdot \ln 5}$

8.11.2. Przebieg zadania

W pokoju gracz widzi puste tabliczki oraz instrukcję „Połącz funkcje z jej pochodną”. Widoczny jest również licznik „Połączono: 0/12”, gdzie liczba 0 oznacza liczbę poprawnie połączonych par, a 12 jest wynikiem, który należy osiągnąć.

Po kliknięciu na tabliczkę obraca się ona, a gracz widzi wzór. Kolejne kliknięcia tej samej tabliczki nie wywołują reakcji. Po kliknięciu na inną tabliczkę, również ona zostaje odwrócona i wyświetlana jest kolejna formuła. Nie można otworzyć jednocześnie więcej niż dwóch tabliczek.

Jeżeli odkryte formuły stanowią funkcję oraz jej pochodną, tabliczki znikają, a licznik „Połączono” zostaje zwiększony. W przeciwnym przypadku tabliczki wracają do pozycji wyjściowej. Zadaniem gracza jest połączenie wszystkich funkcji z ich pochodnymi. Widok pokoju przedstawiono na rysunku 8.26.



Rysunek 8.26: Widok pokoju pochodnych

8.11.3. Opis pokoju

Tabliczka `BP_Card`

Wszystkie tabliczki zostały zrealizowane jako obiekty typu Blueprint Actor `BP_Card`. Każdy obiekt posiada komponent `StaticMesh`, odpowiedzialny za wygląd trójwymiarowy, komponent kolizji typu `BoxCollision` oraz komponent typu `Widget`, który odpowiada za wyświetlanie tekstu funkcji.

Obiekt zawiera również zmienną `PairID`, pełniącą rolę identyfikatora pary, oraz zmienną `FormulaText`, która przechowuje tekst formuły. Początkowo tabliczki umieszczone są w pokoju stroną, na której formuła nie jest widoczna.

Kiedy gracz wejdzie w obszar kolizji tabliczki, obiekt obraca się o 180 stopni, odsłaniając wzór, a interakcja z tym obiektem zostaje wyłączona.

Menedżer `BP_PuzzleManager`

W pokoju znajduje się obiekt typu Blueprint Actor `BP_PuzzleManager`. Obiekt ten składa się z komponentu widżetu typu Blueprint Widget `WBP_Status`, który jest wyświetlany nad tabliczkami i odpowiada za obsługę interakcji gracza.

Po kliknięciu pierwszej tabliczki menedżer zapisuje ją w zmiennej `FirstCard`, natomiast po kliknięciu drugiej zapisuje ją w zmiennej `SecondCard`. Następnie blokowana jest interakcja gracza z pozostałymi tabliczkami i porównywane są wartości zmiennej `PairID` obu obiektów.

Jeżeli identyfikatory są zgodne, wyświetlany jest tekst „Para", licznik „Połączono" zostaje zwiększony, a tabliczki znikają. W przeciwnym przypadku wyświetlany jest tekst „Nie para", a tabliczki wracają do pierwotnego położenia. Po zakończeniu sprawdzania interakcja gracza z tabliczkami zostaje ponownie włączona.

8.12. Zadanie 12 – Układy równań (Andrii Demyshyn)

8.12.1. Różnice realizacji zadania w praktyce

W tym zadaniu zrealizowano zagadkę matematyczną opartą na osiągnięciu zadanej wartości energetycznej niezbędnej do zapalenia żarówki. Zadaniem gracza jest umieszczenie odpowiedniej liczby kulek każdego koloru w pojemnikach w taki sposób, aby łączna wartość energetyczna kulek była równa wartości wskazanej nad żarówką.

W trakcie opracowywania zadania zdecydowano się na zmianę logiki zagadki. Początkowo planowano implementację rozwiązania posiadającego tylko jedno poprawne rozwiązanie, co nadawałoby grze charakter jednorazowy. We wcześniejszej koncepcji docelowa wartość energii była stała, a gracz musiał umieścić dokładnie 11 kulek w kontenerach. W toku realizacji podjęto decyzję o zwiększeniu różnorodności rozgrywki poprzez losowanie wartości docelowej z przedziału od 18 do 51. W obecnej wersji zagadki gracz sam decyduje o liczbie umieszczanych kulek, pod warunkiem że ich łączna wartość energetyczna odpowiada liczbie wyświetlanej nad żarówką. Dzięki tym zmianom zagadka stała się bardziej zróżnicowana i wielokrotnego użytku.

Po wejściu do pokoju gracz widzi lampę, trzy pojemniki na kulki, zestaw kolorowych kulek oraz panel informacyjny z instrukcją zadania. Nad lampą wyświetlana jest docelowa wartość energii, którą należy osiągnąć.

8.12.2. Opis obiektów sceny

Panel informacyjny `BP_RiddleText`

W pokoju umieszczono panel informacyjny w postaci blueprintu aktora `BP_RiddleText`, wewnątrz którego znajduje się blueprintowy widget `ZagadkaLamp`. W widżecie wyświetlana jest instrukcja zadania:

> „Aby uruchomić oświetlenie, należy włożyć do pojemników taką liczbę kulek, aby łącznie wygenerowana moc była dokładnie równa wartości wyświetlanej nad lampą. Jeśli moc będzie zbyt niska lub zbyt wysoka — instalacja nie zadziała. Pojemniki przyjmują tylko kulki w swoim kolorze.
>
> Parametry energetyczne kulek: Czerwona kulka ma o 3 jednostki energii więcej niż zielona. Niebieska kulka ma o 1 jednostkę energii więcej niż czerwona. Trzy zielone kulki mają o 2 jednostki energii mniej niż jedna niebieska.”

Kulki

W scenie zaprojektowano trzy typy kulek: czerwone, zielone oraz niebieskie. Zostały one zrealizowane za pomocą trzech blueprintów aktorów: `BR`, `BB` oraz `BG`, gdzie pierwsza litera jest skrótem od słowa „Ball”, a druga od pierwszej litery koloru.

Każdy obiekt kulki składa się z:

- komponentu `StaticMesh` w postaci sfery,
- komponentu `BoxCollision`,
- tagu `Ball`,
- tagu odpowiadającego kolorowi kulki (`Green`, `Blue`, `Red`).

Kulki są obiektami fizycznymi, które gracz może podnosić, przenosić oraz umieszczać w kontenerach.

Kontenery

W scenie zaimplementowano trzy kontenery w postaci obiektów typu Blueprint Actor: `BP_Container_Red`, `BP_Container_Blue` oraz `BP_Container_Green`.

Ściany każdego kontenera składają się z czterech komponentów typu `StaticMesh` z materiałem szklanym, dna wykonanego z komponentu `StaticMesh` w odpowiednim kolorze oraz komponentu `BoxCollision`.

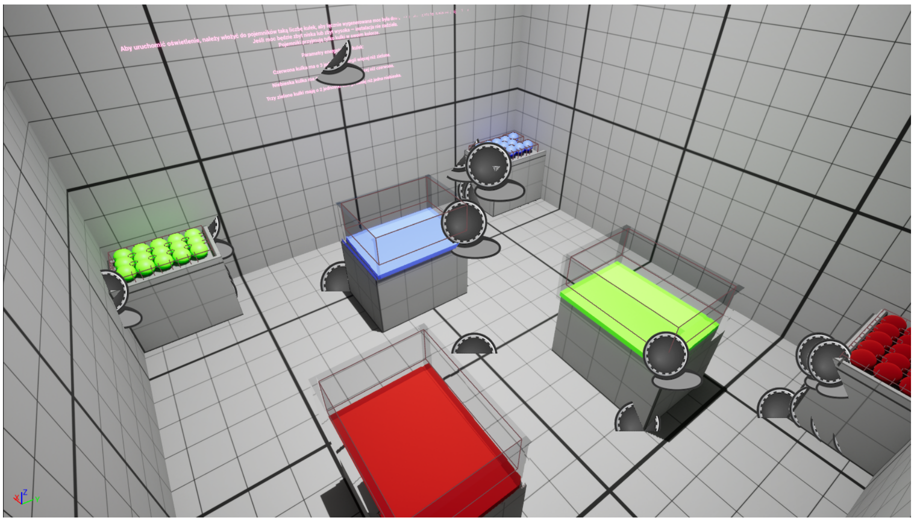
Kontenery przyjmują oraz przekazują do żarówki wartości energetyczne kulek odpowiadającego im koloru. Mechanizm ten został zrealizowany w taki sposób, że w momencie wejścia obiektu w obszar `BoxCollision` kontener sprawdza, czy obiekt posiada tag `Ball` oraz tag odpowiadający kolorowi kontenera. Jeżeli warunki są spełnione, kontener dodaje wartość energetyczną kulki do swojej łącznej wartości, która domyślnie wynosi 0. W przeciwnym przypadku nie jest wykonywana żadna akcja.

Jeżeli kulka opuszcza obszar kontenera, jej wartość energetyczna zostaje odjęta od sumy. Aktualne wartości energetyczne kontenerów są na bieżąco przekazywane do obiektu `BP_Lamp`. Ogólny wygląd pokoju wraz z rozmieszczeniem pojemników, kulek oraz żarówki przedstawiono na rys. 8.27.

Żarówka

Żarówka została zrealizowana jako blueprint aktora `BP_Lamp`. Obiekt składa się z komponentów:

- `StaticMesh`,
- `TextRender`,
- `PointLight`.



Rysunek 8.27: Wygląd pokoju z kulkami

8.12.3. Zakończenie zadania

Komponent `TextRender` wyświetla wartość energii, którą gracz musi osiągnąć. Wartość ta jest generowana losowo przy każdym uruchomieniu gry z przedziału od 18 do 51.

W obiekcie `BP_Lamp` zaimplementowano funkcję `ApplyTotal`, która porównuje sumę wartości energetycznych kulek przekazywanych przez kontenery z wygenerowaną wartością docelową. W przypadku zgodności wartości żarówka zmienia kolor, zapala się, a zadanie zostaje zaliczone. W przeciwnym przypadku lampa pozostaje wyłączona.

8.13. *Zadanie 13 – Stereometria (Jan Walczak)*

8.13.1. *Problemy i różnice w realizacji zadania w praktyce*

Zadanie nie odbiega za bardzo od wcześniej opracowanego opisu (podrozdz. 7.13). Zmieniono liczbę wyświetlanych brył geometrycznych. Według opisu graczowi miała być przedstawiana tylko jedna bryła. W praktyce takie zadanie byłoby bardzo krótkie. Zdefiniowano więc pulę złożoną z 5 brył:

1. kuli,
2. stożka,
3. walca,
4. sześcianu,
5. ostrosłupa prawidłowego czworokątnego.

Do każdej z wymienionych brył wybrano 3 pytania. Dotyczą one kolejno:

- rodzaju bryły,
- wzoru na obliczenie powierzchni bryły,
- wzoru na obliczenie objętości bryły.

Gracz musi odpowiedzieć na wszystkie 3 pytania prawidłowo. Jeśli się pomyli, ze zbioru wybierana jest inna bryła. Inna niż ta, o którą pytano go przed chwilą. W ten sposób uniknięto sytuacji, w której gracz odpowiadałby losowo i przechodził zadanie metodą eliminacji.

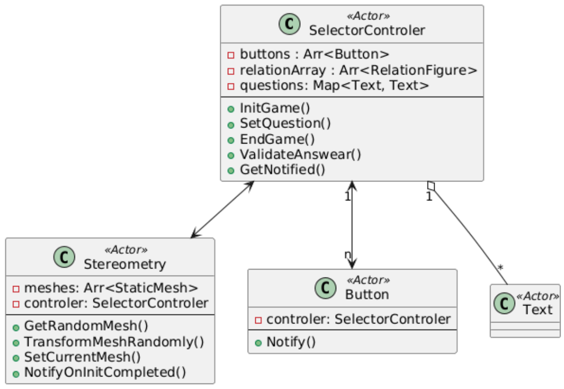
8.13.2. *Implementacja zadania*

W zadaniu został zaimplementowany jeden główny kontroler `SelectorControler`. Jego zadaniem jest inicjalizacja gry, zarządzanie przebiegiem gry, wyświetlanie tekstu i wybór odpowiednich pytań, wyświetlanych graczowi. Zawiera on dwustronną referencję do obiektu `Stereometry` oraz dwustronną referencję do trzech obiektów `Button` (takich, jak te opisane w podrozdziałach 8.2 i 8.9). Zawiera także predefiniowany zbiór pytań, dotyczących każdej z brył, reprezentowany przez mapę typu *pytanie:poprawna odpowiedź*. Powiązania kontrolera z pozostałymi komponentami w zadaniu zostały zaprezentowane na rysunku 8.28.

Obiekt `Stereometry` odpowiada za poprawne wyświetlanie brył. Zawiera 5 obiektów typu siatki statyczne, które wyświetla w świecie gry, wybierając bryłę odpowiadającą aktualnie zadawanemu pytaniu. Jest też odpowiedzialny za pseudolosowy mechanizm wyboru brył. Przed wyświetleniem obiekt `Stereometry` w sposób pseudolosowy obraca bryłę, zmienia jej rozmiar, oraz położenie w świecie gry. W tym przypadku nie było potrzeby implementowania mechanizmu powiadomień, ponieważ rodzaj komunikatów wysyłanych pomiędzy obiektami `Stereometry` i `SelectorControler` jest prosty i zawsze jednakowy (żądanie wyświetlenia nowej bryły). Dodatkowo, obiekt `Stereometry` zawiera referencję do kontrolera tylko dlatego, że jego inicjalizacja jest długa (prawdopodobnie przez wczytywanie siatek statycznych brył, z nałożoną teksturą odbijającą światło). Musi więc powiadomić kontroler główny, że zakończył swoją inicjalizację, aby ten mógł zacząć wykonywać na nim operacje, przez wywołanie odpowiednich funkcji.

8.13.3. *Przebieg zadania*

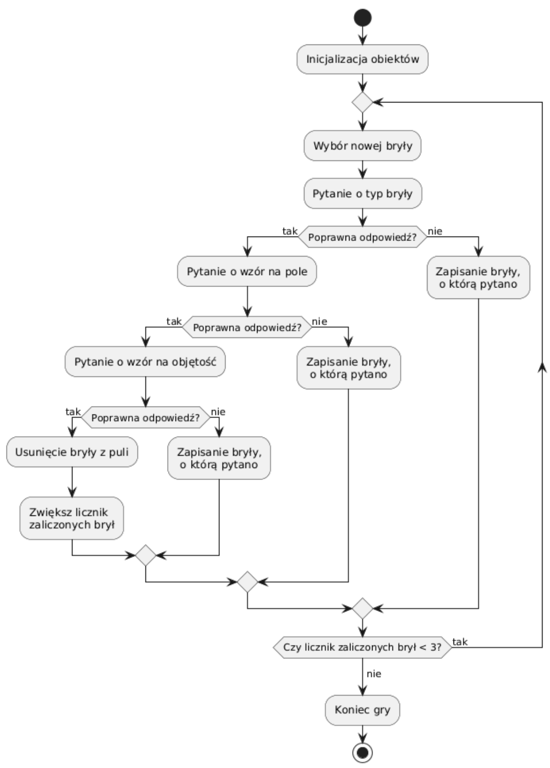
Kontroler inicjalizuje podrzędne mu obiekty i rozpoczyna grę. W sposób pseudolosowy wybierana jest jedna z 5 predefiniowanych brył, poprzez wywołanie odpowiedniej funkcji obiektu



Rysunek 8.28: Diagram UML zawierający najważniejsze elementy kontrolerów dla zadania 13 – stereometria.

`Stereometry`. Gracz ma do wyboru 3 przyciski, które zawierają odpowiedzi *A, B* lub *C*. Pytania są wyświetlane sekwencyjnie – najpierw zadawane jest pytanie o typ bryły, potem o wzór na obliczenie jej pola powierzchni, a na końcu o wzór na obliczanie jej objętości.

Jeśli gracz pomyli się podczas udzielania odpowiedzi, kontroler ponownie wywoła funkcję odpowiedzialną za wybór kolejnej bryły. W takim przypadku może się ona pojawić ponownie w zadaniu, ale nie od razu po udzieleniu błędnej odpowiedzi. Jeśli gracz odpowie na wszystkie 3 pytania prawidłowo, bryła zostaje uznana za zaliczoną i jest usuwana z puli dostępnych do wylosowania brył. Gra kończy się kiedy gracz poprawnie odpowie na pytania dotyczące 3 różnych brył. Przebieg całego zadania został zaprezentowany w postaci diagramu przepływu na rysunku 8.29



Rysunek 8.29: Diagram przepływu dla zadania 13 – stereometria

9. RAPORT Z TESTOWANIA (JAN WALCZAK)

9.1. *Przebieg testów*

Podczas implementacji aplikacji można wyróżnić trzy iteracje testowania, z których każda skupiała się na innej funkcjonalności. Głównym celem testów było sprawdzenie, jak zachowuje się przygotowana aplikacja w warunkach laboratoryjnych w LZWP.

Pierwsza iteracja obejmowała testy podstawowych mechanizmów dotyczących poruszania się gracza. Trwała ona dwa dni, podczas których odbyły się dwa spotkania w LZWP. Sprawdzono:

- śledzenie ruchu gracza w LZWP,
- poprawność wczytywania wejścia poprzez naciskanie odpowiednich przycisków na kontrolerze,
- podstawowe interakcje z otoczeniem, takie jak używanie wskaźnika laserowego oraz chwytanie wirtualnych przedmiotów.

Druga iteracja testów miała na celu sprawdzenie funkcjonalności wszystkich opracowanych zagadek. Skupiała się na identyfikacji miejsc, w których aplikacja generowała błędy w środowisku docelowym. Zidentyfikowane błędy były poprawiane poza LZWP. Iteracja ta obejmowała trzy dni spotkań w LZWP.

Ostatnia iteracja testów koncentrowała się na wielokrotnym budowaniu aplikacji oraz analizie różnic w działaniu pomiędzy wersją edytorską, testowaną w edytorze Unreal Engine, a wersją skompilowaną aplikacji (ang. *build*), która działała w CAVE w LZWP. Była to największa część testowania i zajęła dwa tygodnie.

9.2. *Napotkane problemy*

Najwięcej problemów napotkano podczas trzeciej iteracji testów. Okazało się, że skompilowana wersja aplikacji zachowywała się inaczej niż wersja edytorska. Wszelkie napotkane błędy wymagały wprowadzenia poprawek oraz ponownej kompilacji aplikacji, co było procesem czasochłonnym i znacząco wydłużyło tę część testowania.

Głównym problemem, którego nie można było wykryć na etapie testów w edytorze, było pozycjonowanie kamery gracza oraz odpowiednia synchronizacja pozycji kontrolera w LZWP. Różnice te wynikały z nieprawidłowego nałożenia przesunięcia (ang. *offset*) oraz skali kamery, która reprezentowała gracza w wirtualnej przestrzeni. W konsekwencji dochodziło do niepoprawnego wyświetlania wirtualnego pokoju w CAVE – jego podłoga częściowo nachodziła na ściany, a ruch kontrolera nie był poprawnie przekazywany do aplikacji, przez co gracz nie był w stanie skutecznie poruszać się po wirtualnym pomieszczeniu. Rozwiązaniem było eksperymentalne ustawienie przesunięcia i skali kamery, tak aby granice ścian pokoju były poprawnie wyświetlane.

Kolejnym problemem była synchronizacja obiektów generowanych w sposób pseudolosowy. Okazało się, że klaster komputerowy, na którym uruchamiana była aplikacja w LZWP, obliczał liczby pseudolosowe niezależnie na każdym komputerze. Skutkowało to desynchronizacją tych obiektów, a w konsekwencji ich niepoprawnym wyświetlaniem. Rozwiązaniem okazało się ustalenie wspólnego ziarna (ang. *seed*) dla wszystkich komputerów działających w klastrze.

Ostatnim istotnym problemem była desynchronizacja obiektów w grze, na które oddziaływała grawitacja. Problem ten miał podobną naturę do poprzedniego — położenie takich obiektów było obliczane niezależnie na każdym komputerze klastra, przez co nie było wyznaczane jednoznacznie. Rozwiązaniem było ustawienie w edytorze Unreal Engine symulacji fizyki w tryb deterministyczny.

9.3. Wykryte błędy

Oprócz wyżej wymienionych problemów pojawiły się również błędy, które nie były związane z działaniem aplikacji w LZWP, lecz wynikały bezpośrednio z błędów implementacyjnych.

Pierwszym wykrytym problemem była funkcja chwytania przedmiotów przez gracza. Kontroler, z którego korzysta użytkownik w LZWP, wyposażony jest w analogowy spust, który po naciśnięciu zwraca wartości zmiennoprzecinkowe. W aplikacji mechanizm chwytania przedmiotów został natomiast zaprojektowany z myślą o wejściu zero-jedynkowym, odpowiadającym wciśnięciu klawisza na klawiaturze. Błąd ten okazał się na tyle istotny, że uniemożliwiał poprawne chwytanie obiektów. Rozwiązaniem było zmodyfikowanie obsługi wejścia w taki sposób, aby przyjmowała wartości zmiennoprzecinkowe.

Drugim znaczącym błędem był sposób generowania figur w zadaniu opisanym w podrozdziale nr. 8.2. Figury były tworzone podczas inicjalizacji zadania w sposób nieprawidłowy. Obiekty należące do poszczególnych zadań były oznaczane odpowiednimi etykietami, co umożliwiało aplikacji rozpoznanie ich przynależności do konkretnego zadania. Generowane figury nie posiadały jednak takich etykiet, w wyniku czego nigdy nie znikały z wirtualnego pokoju. Kontroler odpowiedzialny za ich położenie zawierał referencje do instancji tych obiektów, przez co ukrywana była jedynie ich tekstura. Błąd ten okazał się szczególnie problematyczny, ponieważ niewidoczne figury wchodziły w kolizję z innymi obiektami znajdującymi się na scenie. Prowadziło to m.in. do niezamierzonego naciskania przycisków lub blokowania dostępu gracza do pozostałych elementów interaktywnych. Rozwiązaniem było dodanie odpowiednich etykiet do generowanych figur.

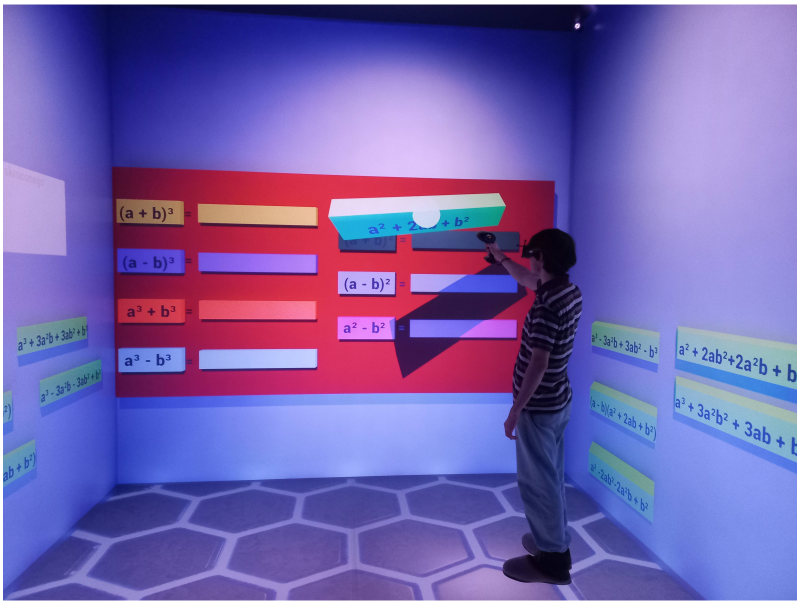
Ostatnim istotnym błędem było niespójne oznaczanie przedmiotów, które mogły być chwytane przez gracza. Funkcjonalność chwytania opierała się na przyczepianiu chwytanych elementów do kontrolera, co realizowano poprzez dołączanie całych obiektów reprezentowanych w aplikacji przez aktorów. Część elementów została jednak zaprojektowana w sposób nieprawidłowy, tzn. podczas chwytania przenoszona była jedynie ich tekstura, natomiast obiekt aktora pozostawał w miejscu, z którego został zabrany. W konsekwencji obiekty te mogły zostać złapane tylko jeden raz, po czym ich tekstura ulegała oddzieleniu od obiektu aktora. Prowadziło to do niespójnego zachowania aplikacji oraz utrudniało dalszą interakcję z otoczeniem. Rozwiązaniem było ujednolicenie systemu chwytania przedmiotów oraz oparcie go na przyczepianiu do ręki pełnych obiektów aktorów.

10. SPRAWOZDANIE Z DZIAŁANIA APLIKACJI (ANDRII DEMYSHYN)

W ramach realizacji projektu opracowano trzynaście interaktywnych zagadek matematycznych, z których każda reprezentuje inny dział matematyki nauczanej na poziomie szkoły średniej. Zagadki zostały zaprojektowane tak, aby nie tylko sprawdzać wiedzę użytkowników, ale również aktywizować ich poprzez wykorzystanie mechanik charakterystycznych dla gier logicznych i escape roomów. Dzięki zastosowaniu technologii wirtualnej rzeczywistości, gracze mogą wchodzić w bezpośrednią interakcję z otoczeniem i rozwiązywać zadania w formie angażujących, przestrzennych łamigłówek. Każde z zadań posiada unikalny scenariusz i zasady działania, dostosowane do specyfiki danego działu matematyki oraz do możliwości sprzętowych Laboratorium Zanurzonej Wizualizacji Przestrzennej. Poniżej przedstawiono szczegółowy opis wszystkich trzynastu zagadek, wraz z określeniem ich celu, przebiegu oraz warunków zakończenia.

10.1. Wzory skróconego mnożenia

Pierwsze zadanie wirtualnego escape roomu polega na rozpoznawaniu oraz uzupełnianiu wzorów skróconego mnożenia. Po pojawieniu się gracza w wirtualnym pomieszczeniu widoczna jest tablica zawierająca początki wybranych wzorów algebraicznych. Każda formuła przedstawiona jest w postaci lewej strony równania wraz ze znakiem „=”, po którym pozostawiono puste miejsce przeznaczone na brakującą część wyrażenia. W przestrzeni wirtualnego pokoju rozmieszczone są interaktywne bloki zawierające możliwe zakończenia wzorów. Wśród dostępnych elementów znajdują się zarówno poprawne zakończenia odpowiadające właściwym wzorom skróconego mnożenia, jak i bloki z nieprawidłowymi zapisami, które pełnią funkcję elementów rozpraszających. Zadaniem gracza jest przeciąganie wybranych bloków oraz umieszczanie ich w odpowiednich miejscach na tablicy, tak aby utworzyć kompletne i matematycznie poprawne formuły. Widok sceny zadania przedstawiono na rysunku 10.1.



Rysunek 10.1: Uzupełniania wzorów skróconego mnożenia

Po uzupełnieniu wszystkich pustych pól użytkownik ma możliwość uruchomienia przycisku sprawdzającego. Po jego użyciu system dokonuje weryfikacji poprawności skonstruowanych równań i wyświetla komunikat tekstowy informujący o wyniku zadania. Zagadkę uznaje się za rozwiązaną w momencie prawidłowego dopasowania wszystkich zakończeń do odpowiadających im początków wzorów. Szacowany czas rozwiązania zagadki wynosi od 5 do 7 minut.

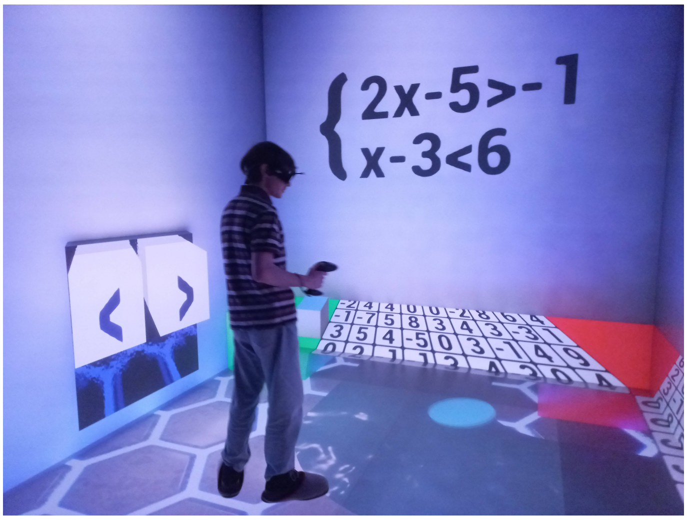
10.2. *Planimetria*

Po ukończeniu pierwszego zadania na ścianach pokoju pojawiają się figury geometryczne. Wśród nich znajdują się m.in. kwadraty, romby, prostokąty, równoległoboki, trapezy oraz różne typy trójkątów. Zadanie ma na celu sprawdzenie znajomości zależności między figurami oraz umiejętności rozpoznawania ich cech charakterystycznych. Na ścianie centralnej, pomiędzy prezentowanymi zbiorami figur, znajdują się przyciski służące do wyboru relacji. Gracze analizują właściwości figur i decydują, czy każda figura z czerwonego zbioru jest jednocześnie figurą należącą do zielonego zbioru. Do dyspozycji mają dwa przyciski: *Każdy* – oznaczający, że relacja jest prawdziwa, oraz *Nie każdy* – oznaczający, że relacja jest fałszywa. Przykładowy układ figur oraz wybór relacji pokazano na rys. 10.2. Po udzieleniu odpowiedzi system natychmiast weryfikuje poprawność wskazanej relacji. W przypadku błędnej odpowiedzi gracz otrzymuje komunikat zwrotny, a następnie wyświetlana jest kolejna relacja do oceny. Zadanie kończy się w momencie, gdy wszystkie relacje zostaną poprawnie oznaczone. Szacowany czas rozwiązania zagadki wynosi od 3 do 5 minut.

Rysunek 10.2: Zadanie planimetryczne – relacje między figurami

10.3. *Nierówności*

W trzecim zadaniu gracze trafiają na scenę z mostem zbudowanym z kamiennych płytek oraz metaliczną kostką umieszczoną na jego początku. Kostka musi przejść przez wiszący most, poruszając się po płytkach. Na każdej z płytek widnieje liczba, natomiast na ścianie wyświetlany jest układ dwóch nierówności. Widok mostu z kafelkami liczbowymi oraz układem nierówności przedstawiono na rys. 10.3.

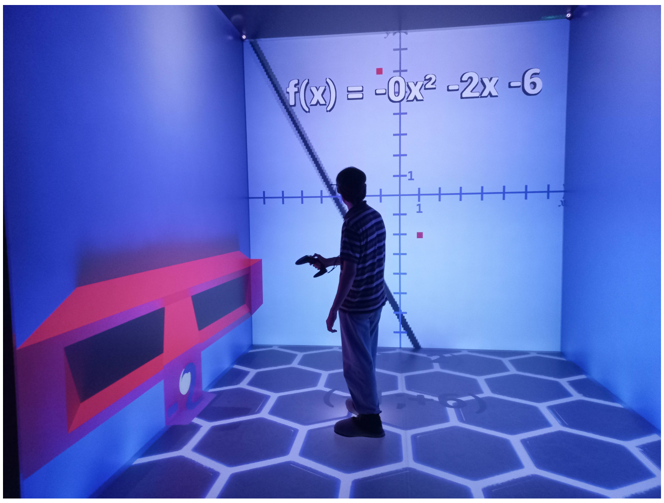


Rysunek 10.3: Zadanie z nierównościami – przejście przez most

Zadaniem uczestników jest rozwiązanie układu nierówności oraz wybór płytek z wartościami należącymi do przedziału spełniającego oba warunki. Gracze prowadzą kostkę, korzystając z przycisków strzałek, które przesuwają ją na wybraną płytkę, a błędny wybór powoduje zapadnięcie się płytki i konieczność powrotu na początek lub do punktu kontrolnego. Po poprawnym przejściu przez most zagadka zostaje uznana za rozwiązaną, a gracze uzyskują dostęp do kolejnego etapu gry. Szacowany czas rozwiązania zagadki wynosi od 5 do 7 minut.

10.4. Funkcje

W czwartym zadaniu gracze pracują z dużą interaktywną tablicą, na której wyświetlany jest układ współrzędnych. Ich celem jest dostosowanie parametrów funkcji w taki sposób, aby jej wykres przechodził przez wyznaczone punkty. Interaktywną tablicę z wykresem funkcji pokazano na rys. 10.4. Dzięki suwakom uczestnicy mogą dynamicznie zmieniać współczynniki funkcji, obserwując na bieżąco, jak zmienia się jej wykres. Zadanie polega na precyzyjnym dobraniu wartości parametrów funkcji tak, aby wykres przeciął wszystkie wskazane punkty. Szacowany czas rozwiązania zagadki wynosi od 4 do 6 minut.



Rysunek 10.4: Interaktywna tablica do modyfikacji wykresu funkcji

10.5. Geometria analityczna

W piątym zadaniu gracze ponownie pracują z interaktywną tablicą, na której tym razem pojawiają się dwa wykresy funkcji. Celem wyzwania jest wyznaczenie współrzędnych punktu przecięcia wykresów funkcji na płaszczyźnie kartezjańskiej. Uczestnicy mogą odczytać ten punkt bezpośrednio z wykresu lub wyznaczyć go na podstawie równań opisujących obie funkcje. Po ustaleniu wartości współrzędnych gracze wybierają je za pomocą dedykowanych przycisków. System weryfikuje poprawność wskazanych współrzędnych i odblokowuje przejście do kolejnego etapu gry. Przykład wyznaczania punktu przecięcia wykresów funkcji przedstawiono na rys. 10.5. Szacowany czas rozwiązania zagadki wynosi od 4 do 6 minut.

Rysunek 10.5: Interaktywna tablica do modyfikacji wykresu funkcji

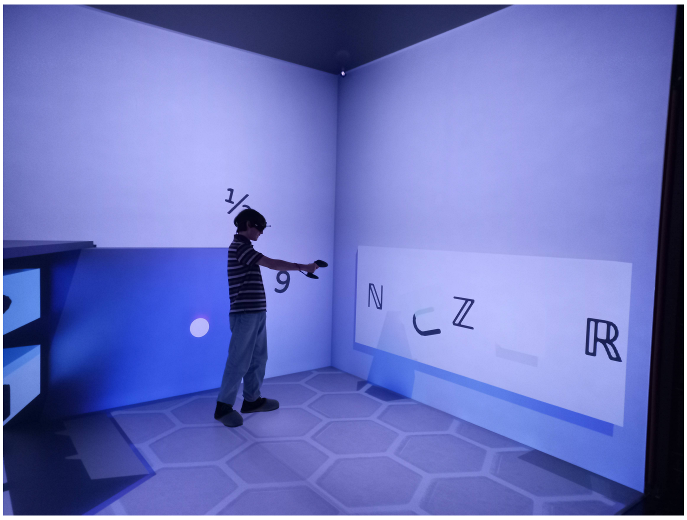
10.6. Kombinatoryka

W szóstym zadaniu gracze natrafiają na metalowy sejf wyposażony w podświetlany panel numeryczny. Na oddzielnym panelu informacyjnym wyświetlana jest zagadka z zakresu kombinatoryki dotycząca liczby możliwych ustawień kodu PIN. Uczestnicy muszą przypomnieć sobie zasady kombinatoryki i obliczyć liczbę możliwych kombinacji zgodnie z treścią wylosowanej zagadki. Po obliczeniu właściwej liczby gracze wprowadzają wynik za pomocą klawiatury sejfu. Widok sejfu z panelem numerycznym oraz treścią zagadki pokazano na rys. 10.6. Szacowany czas rozwiązania zagadki wynosi od 3 do 7 minut.

Rysunek 10.6: Sejf z zagadką kombinatoryczną

10.7. Liczby rzeczywiste i działania na zbiorach liczbowych

W siódmym zadaniu gracze pracują z elektroniczną skrzynią wyposażoną w klawiaturę numeryczną oraz elementy związane ze zbiorami liczbowymi. Zadanie składa się z dwóch etapów i ma na celu sprawdzenie znajomości podstawowych własności zbiorów liczb rzeczywistych oraz relacji między nimi. Pierwszy etap polega na analizie liczb wyświetlonych w pomieszczeniu oraz określeniu, ile z nich należy do poszczególnych zbiorów liczbowych, takich jak liczby naturalne, całkowite, wymierne, niewymierne oraz zbiór liczb rzeczywistych. Na podstawie uzyskanych wyników gracze ustalają kod otwierający elektroniczny zamek skrzyni, zgodnie z kolejnością wskazaną na panelu informacyjnym umieszczonym na ścianie. Po poprawnym wprowadzeniu kodu uruchamiany jest drugi etap zadania. Gracze otrzymują zestaw ruchomych bloczków z symbolami relacji zbiorów, takich jak zawieranie się zbiorów, część wspólna oraz zbiór pusty. Ich zadaniem jest uzupełnienie wyświetlonych relacji pomiędzy zbiorami liczbowymi poprzez umieszczenie odpowiednich symboli we właściwych miejscach. Drugi etap zadania przedstawiono na rys. 10.7.



Rysunek 10.7: Zadanie dotyczące zbiorów liczb rzeczywistych

Poprawne wykonanie obu etapów zadania skutkuje zaliczeniem zagadki oraz odblokowaniem przejścia do kolejnego etapu gry.Szacowany czas rozwiązania zagadki wynosi od 6 do 8 minut.

10.8. Znaki funkcji trygonometrycznych

W ósmym zadaniu gracze stają przed układem jednostkowym reprezentującym jedną z funkcji trygonometrycznych, takich jak sinus, cosinus, tangens lub cotangens. Każdy okrąg podzielony jest na cztery ćwiartki układu współrzędnych oznaczone numerami I–IV. Zadaniem uczestników jest uzupełnienie pustych pól w każdej ćwiartce odpowiednim znakiem „+” lub „−” za pomocą interaktywnych elementów, wskazujących, czy dana funkcja przyjmuje w tej ćwiartce wartości dodatnie czy ujemne. Poprawne przypisanie znaków wymaga znajomości własności funkcji trygonometrycznych w poszczególnych ćwiartkach. Układ jednostkowy z znakami funkcji trygonometrycznej pokazano na rys. 10.8. Szacowany czas rozwiązania zagadki wynosi od 3 do 5 minut.

Rysunek 10.8: Układ jednostkowy w zadaniu trygonometrycznym

10.9. Ciągi liczbowe

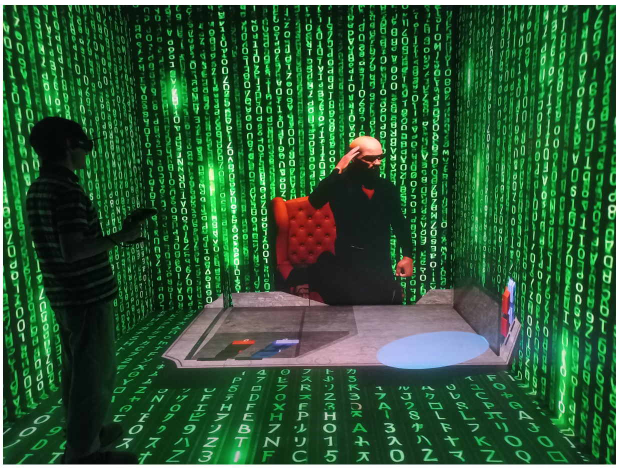
W dziewiątym zadaniu gracze przenoszą się na scenę stylizowaną na grę „Duck Hunt”, gdzie głównym wyzwaniem jest rozpoznawanie kolejnych wyrazów ciągów liczbowych. Na ekranie wyświetlana jest formuła ciągu arytmetycznego, a z góry lecą kaczki z różnymi wartościami liczbowymi. Zadaniem uczestników jest strzelanie laserem do tych kaczek, które zawierają poprawny wyraz podanego ciągu. Jeśli kaczka nie pasuje do ciągu, gracze powinni ją pozostawić. Scenę stylizowaną na grę „Duck Hunt” z wyrazami ciągu pokazano na rys. 10.9.

Rysunek 10.9: Rozpoznawanie wyrazów ciągu w dynamicznej rozgrywce

Poprawne wskazanie wyrazu ciągu skutkuje zaliczeniem punktu, natomiast błędne działanie prowadzi do utraty jednego z dostępnych żyć. Tempo gry rośnie wraz z postępem, zwiększając wymagania dotyczące refleksu i precyzji. Gra kończy się w momencie utraty wszystkich czterech żyć. Po zakończeniu zadania gracze uzyskują swój wynik i przechodzą do kolejnego etapu gry. Szacowany czas rozwiązania zagadki wynosi od 2 do 4 minut.

10.10. Prawdopodobieństwo

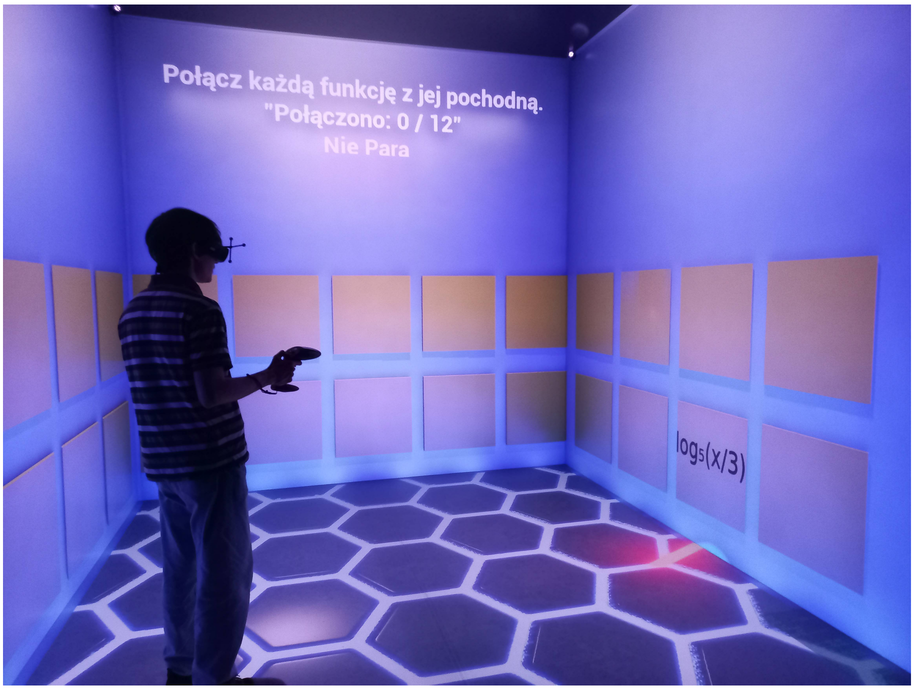
W tym zadaniu gracze stają przed wyzwaniem probabilistycznym, realizowanym w formie interaktywnej sceny z postacią Morfeusza, dwoma pojemnikami oraz zestawem 100 tabletek – po 50 czerwonych i niebieskich. Celem jest rozdzielenie tabletek między pojemniki w taki sposób, aby maksymalizować szansę wybrania czerwonej tabletki przez Morfeusza. Gracze mogą dowolnie przesuwać tabletki, testując różne układy i sprawdzając uzyskane prawdopodobieństwo za pomocą przycisku weryfikacji. Zadanie posiada rozwiązanie optymalne, które pozwala osiągnąć maksymalne możliwe prawdopodobieństwo sukcesu, wynoszące około 74,75%. Zadanie wymaga od uczestników zrozumienia i wykorzystania zasad prawdopodobieństwa do podejmowania decyzji. Końcowy stan zadania po poprawnym rozwiązaniu przedstawiono na rysunku 10.10. Szacowany czas rozwiązania zagadki wynosi od 5 do 7 minut.



Rysunek 10.10: Zadanie z zakresu rachunku prawdopodobieństwa

10.11. Optymalizacja i rachunek różniczkowy

W jedenastym zadaniu gracze mierzą się z zagadką z zakresu rachunku różniczkowego, polegającą na łączeniu funkcji z ich pochodnymi. W pomieszczeniu rozmieszczone są interaktywne tabliczki, które początkowo nie ujawniają swojej zawartości. Po ich aktywacji gracze odkrywają zapisane na nich wzory matematyczne. Widok pokoju przedstawiono na rys. 10.11.



Rysunek 10.11: Łączenie funkcji z odpowiadającymi im pochodnymi

Zadaniem uczestników jest odnalezienie par składających się z funkcji oraz odpowiadającej jej pochodnej. W danym momencie możliwe jest odkrycie maksymalnie dwóch tabliczek. Jeżeli wybrane wzory tworzą poprawną parę, tabliczki znikają z planszy, a licznik postępu zostaje zwiększony. W przypadku błędnego dopasowania tabliczki wracają do stanu początkowego. Zagadka zostaje uznana za rozwiązaną po poprawnym połączeniu wszystkich dostępnych par. Zadanie wymaga od graczy znajomości podstawowych reguł obliczania pochodnych oraz umiejętności kojarzenia zależności pomiędzy funkcją a jej pochodną. Szacowany czas rozwiązania zagadki wynosi od 6 do 8 minut.

10.12. Układy równań

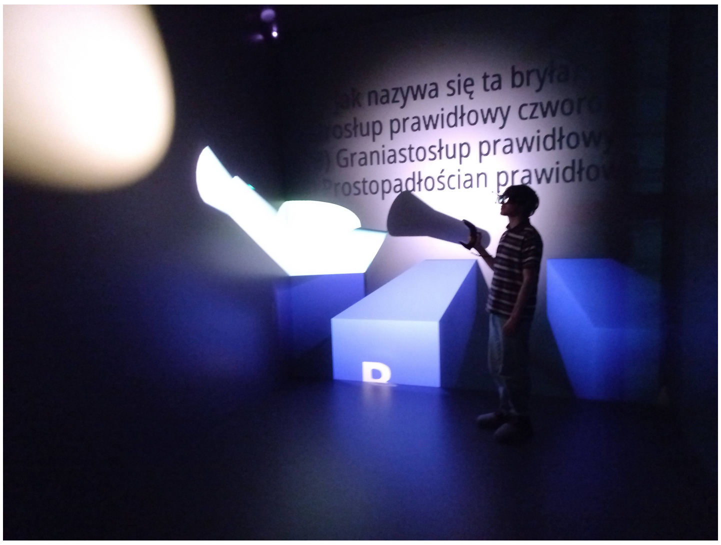
W dwunastym zadaniu gracze rozwiązują zagadkę opartą na układach równań, polegającą na dobraniu odpowiedniej liczby kulek o różnych wartościach energetycznych. W pomieszczeniu znajdują się trzy kolorowe pojemniki: czerwony, zielony oraz niebieski, z których każdy przyjmuje wyłącznie kulki odpowiadającego mu koloru. Zadanie z kulkami energetycznymi oraz żarówką przedstawiono na rys. 10.12.

Rysunek 10.12: Rozwiązywanie układów równań przy użyciu kulek

Zadaniem uczestników jest umieszczenie w pojemnikach takiej liczby kulek, aby łączna wartość generowanej energii była dokładnie równa wartości wyświetlanej nad żarówką. Przy doborze kulek należy uwzględnić zależności energetyczne pomiędzy poszczególnymi kolorami, co wymaga logicznego myślenia oraz poprawnego modelowania zależności w postaci układu równań. Poprawne dobranie liczby kulek powoduje zapalenie się żarówki nad stołem, sygnalizując rozwiązanie zagadki. W przypadku niepoprawnego ustawienia gracze mogą dowolnie modyfikować rozmieszczenie kulek i podejmować kolejne próby aż do uzyskania właściwego wyniku. Szacowany czas rozwiązania zagadki wynosi od 5 do 7 minut.

10.13. Stereometria

W trzynastym zadaniu gracze obserwują wyświetlaną w półmroku bryłę przestrzenną, prezentowaną w formie trójwymiarowego modelu. Wśród pojawiających się obiektów znajdują się m.in. kula, stożek, walec, sześcian oraz ostrosłup prawidłowy czworokątny. Przykład zadania pokazano na rys. 10.13.



Rysunek 10.13: Zadanie stereometryczne – rozpoznawanie brył

Zadaniem uczestników jest rozpoznanie rodzaju bryły oraz wskazanie poprawnych wzorów na obliczenie jej pola powierzchni i objętości, wybierając jedną z dostępnych odpowiedzi. Pytania zadawane są sekwencyjnie, a błędna odpowiedź powoduje zmianę wyświetlanej bryły. Zadanie zostaje zaliczone po poprawnym rozwiązaniu zestawu pytań dotyczących kilku różnych brył geometrycznych. Szacowany czas rozwiązania zagadki wynosi od 4 do 6 minut.

11. PODSUMOWANIE (ANDRII DEMYSHYN)

11.1. *Ocena realizacji celów*

Głównym celem pracy dyplomowej było opracowanie aplikacji edukacyjnej w formacie matematycznego escape roomu, funkcjonującej w środowisku wirtualnej rzeczywistości. Głównym zadaniem projektu było stworzenie interaktywnej gry VR, łączącej elementy nauki matematyki z mechaniką gry, a także jej dostosowanie do warunków pracy w systemie CAVE.

W ramach realizacji projektu udało się stworzyć środowisko wirtualne, w którym użytkownik kolejno rozwiązuje trzynaście zagadek matematycznych obejmujących różne dziedziny matematyki. Każda zagadka została zrealizowana w formie interaktywnej i wymaga aktywnej interakcji użytkownika z obiektami wirtualnego otoczenia. Aplikacja została zaimplementowana przy użyciu silnika Unreal Engine i przystosowana do działania w środowisku Laboratorium Zanurzonej Wizualizacji Przestrzennej.

Określone cele funkcjonalne, takie jak zapewnienie immersyjnego środowiska VR, obsługa interakcji użytkownika, weryfikacja poprawności rozwiązań oraz kontrola przebiegu rozgrywki, zostały osiągnięte. System umożliwia zarówno samodzielne przechodzenie gry przez użytkownika, jak i nadzór ze strony administratora, co zwiększa stabilność działania aplikacji oraz bezpieczeństwo jej użytkowania.

Zrealizowany projekt spełnia założenia edukacyjne, umożliwiając naukę matematyki w atrakcyjnej i angażującej formie. Dodatkowo podczas realizacji pracy osiągnięto cele osobiste autora, związane z nabyciem praktycznych umiejętności w zakresie projektowania aplikacji VR, pracy z silnikiem Unreal Engine oraz integracji systemu z infrastrukturą CAVE.

Podsumowując, można stwierdzić, że wszystkie główne cele pracy dyplomowej zostały zrealizowane zgodnie z przyjętymi założeniami, a otrzymany rezultat stanowi w pełni funkcjonalne i wartościowe rozwiązanie edukacyjne.

11.2. *Propozycje rozwoju projektu*

Aplikacja opracowana w ramach pracy dyplomowej może być w przyszłości rozbudowana i udoskonalona zarówno pod względem funkcjonalnym, jak i dydaktycznym. Zrealizowany prototyp matematycznego escape roomu w środowisku wirtualnej rzeczywistości stanowi solidną podstawę do dalszych usprawnień i rozwoju projektu.

W pierwszej kolejności można by zwiększyć różnorodność zagadek. Na przykład obecnie na poziomie „Układy równań” przy każdym uruchomieniu zmienia się tylko moc żarówki, ale układ trzech równań pozostaje ten sam. Można by dodać jeszcze kilka układów równań do zadania, ale należy również wziąć pod uwagę, aby można było osiągnąć każdą wygenerowaną moc żarówki. W poziomie „Prawdopodobieństwo” obecnie zawsze tworzy się 100 tabletek i 2 pojemniki. Można sprawić, aby generowano różną liczbę tabletek, na przykład od 26 do 200, oraz generować liczbę pojemników od 2 do 4. Podczas realizacji warto również zwrócić uwagę, aby liczba tabletek i pojemników była zależna od osiągnięcia celów i interesującej interakcji. Takie zmiany zmienności można dodać również do innych poziomów.

W aplikacji można dodać wybór poziomu trudności, na przykład tak, aby na początku przed graczem pojawiły się 3 przyciski wyboru trudności: łatwy, średni, wysoki. Każda zagadka mogłaby być trudniejsza, a po wybraniu poziomu trudności gracz przechodziłby do innego typu zagadki. Jako przykład weźmy zagadkę „Wzory skróconego mnożenia”. Dla łatwego poziomu trudności pozostawimy tę, która jest. Dla średniego poziomu trudności można zrealizować, aby zamiast całych prawych części gracz zbierał części np. „2ab”, „a^2” lub „ab^2”, a przy dopasowywaniu takich części między sobą pojawiałby się znak plus, który po naciśnięciu zmieniałby się na minus i odwrotnie. Dla wysokiego poziomu trudności można by stworzyć grę przypominającą logiczną grę „Piętnastka”. Na planszy rozmieszczone byłyby losowo małe kafelki, z których każdy zawierałby jeden symbol matematyczny lub wyrażenie algebraiczne. Zadaniem gracza w takim zadaniu jest również poprawne dokończenie siedmiu wzorów skróconego mnożenia na planszy. Początek każdego wzoru byłby podany w pierwszej kolumnie, a gracze musieliby przesuwać kafelki tak, aby w odpowiednim wierszu powstało pełne i poprawne wyrażenie algebraiczne. Podobnie, na przykład, w poziomie „Ciągi liczbowe” można zrobić tak, że aby przejść do następnego poziomu, trzeba osiągnąć określony wynik, a im wyższy poziom trudności tematu, tym wyższy wynik trzeba osiągnąć. Tak że w „Ciągi liczbowe” można w zależności od poziomu trudności zwiększać lub zmniejszać liczbę żyć. Takie zmiany dla poziomów trudności można wprowadzić w każdej zagadce.

Można by zrealizować całą aplikację w jednym stylu questowym, co nadałoby jej bardziej klimatyczny charakter. Na przykład zadania „Nierówności” i „Układy równań” można by zrealizować w stylu Indiany Jonesa. Gracz znajdowałby się w pokoju, w którym na ścianach napisane są nierówności, a gracz chodziłby po płytkach mostu, na których napisane są liczby, które pasują lub nie pasują do przedziału, co przypominałoby scenę z filmu „Indiana Jones – W drodze do Graala”. Zadanie „Układy równań” można zrealizować w stylu „Indiana Jones – Poszukiwacze Zaginionej Arki”. W pokoju byłyby rozrzucone różnokolorowe kamienie o różnej wadze, byłby piedestał, na którym napisane układy równań, a po ich rozwiązaniu gracz dowiadywałby się o wadze kamieni i artefakt na piedestale z liczbą jego wagi. Zadaniem gracza byłoby zebranie do worka kamieni o wadze odpowiadającej artefaktowi i zastąpienie ich na piedestale, tak jak w scenie otwierającej film. W takiej stylistyce można by zrealizować wszystkie zadania. Można również zrealizować wszystko w stylu filmu „Matrix” lub innego filmu, serialu, książki, gry. Albo stworzyć przejście w formie detektywa w jakimś laboratorium, gdzie coś się wydarzyło, a gracz musi dowiedzieć się, co się stało, gdzie po każdym etapie gracz otrzymywałby nowe szczegóły. Przy takiej realizacji quest byłby bardzo klimatyczny, co zwiększyłoby zainteresowanie i zachwyt z przejścia.

Jednym z możliwych kierunków byłoby również rozszerzenie bazy gier. Na przykład można by dać inne interpretacje gier w zależności od poziomu trudności lub wyboru gracza. Na przykład w zadaniu „Ciągi liczbowe” zamiast gry „DuckHunt”, w której gracz strzela do kaczek z liczbami, można by zrealizować również grę „BeatSaber”, w której gracz musiałby rozbijać kostki z liczbami mieczem świetlnym.

Można by również wprowadzić system rankingowy oraz limit czasowy, w którym gracze musieliby walczyć o osiągnięcie najlepszego wyniku.

Podsumowując, zaproponowane kierunki rozwoju pokazują, że stworzona aplikacja posiada duży potencjał rozbudowy i może w przyszłości stanowić kompleksowe narzędzie wspomagające nauczanie matematyki z wykorzystaniem technologii wirtualnej rzeczywistości.

11.3. Wnioski końcowe

W ramach niniejszej pracy dyplomowej opracowano i zrealizowano projekt edukacyjny w formie matematycznego escape roomu funkcjonującego w środowisku wirtualnej rzeczywistości. Głównym celem pracy było stworzenie interaktywnej aplikacji VR, łączącej naukę matematyki z elementami gry, oraz jej adaptacja do warunków pracy w systemie CAVE. Wyznaczone cele zostały osiągnięte, a zaprojektowana aplikacja stanowi w pełni funkcjonalne rozwiązanie edukacyjne.

W trakcie realizacji projektu opracowano wirtualne środowisko, w którym użytkownik rozwiązuje trzynaście zróżnicowanych zagadek matematycznych, wymagających aktywnej interakcji z obiektami świata wirtualnego. Zastosowane mechaniki pozwalają na angażujące i intuicyjne przechodzenie gry, a brak klasycznego interfejsu użytkownika sprzyja zwiększeniu immersji w środowisku VR.

Realizacja pracy umożliwiła autorom zdobycie praktycznego doświadczenia w zakresie projektowania aplikacji w silniku Unreal Engine, pracy z technologiami wirtualnej rzeczywistości oraz integracji systemu z infrastrukturą Laboratorium Zanurzonej Wizualizacji Przestrzennej. Projekt pozwolił również na pogłębienie wiedzy z zakresu projektowania interaktywnych systemów edukacyjnych.

Podsumowując, przeprowadzona praca potwierdza, że wykorzystanie technologii wirtualnej rzeczywistości w procesie nauczania matematyki może stanowić skuteczne i atrakcyjne narzędzie dydaktyczne. Opracowana aplikacja spełnia założenia pracy dyplomowej i może stanowić podstawę do dalszego rozwoju oraz zastosowań edukacyjnych.

WYKAZ LITERATURY

[1] E. Campos, I. Hidrogo i G. Zavala, "Impact of virtual reality use on the teaching and learning of vectors", *Frontiers in Education*, t. 7, 2022. URL: `https://www.frontiersin.org/articles/10.3389/feduc.2022.965640/full`

[2] R. Baziak, T. Daruk, K. Żyra, D. Żarek i J. Lebiedź, "Wirtualny pokój zagadek z zakresu matematyki", *Zeszyty Naukowe Wydziału Elektrotechniki i Automatyki Politechniki Gdańskiej*, I Konferencja Innowacje dydaktyczne w matematyce akademickiej DIAM 2024, Gdańsk, 24 czerwca 2024, 2024, s. 59–62. URL: `https://event.mostwiedzy.pl/event/56/attachments/449/945/ZNWEiAPG_76.pdf#page=60`

[3] D. Żarek i J. Lebiedź, "Educational values of a virtual escape room in mathematics", *Proceedings of the 33rd International Conference on Information Systems Development (ISD 2025)*, 3–5 September 2025, Belgrade, 2025. URL: `https://aisel.aisnet.org/isd2014/proceedings2025/education/5/`

[4] "Empiriusz 2.0", (**urldate** 2026-01-14). URL: `https://www.nowaera.pl/empiriusz`

[5] "Czas na geometrię w innym wymiarze!", (**urldate** 2026-01-14). URL: `https://www.nowaera.pl/wirtualne-laboratorium/geometria`

[6] M. Sołodki. "Okulary interaktywne dla uczniów. Jak działają i czy warto z nich korzystać podczas lekcji?", (**urldate** 2026-01-14). URL: `https://www.nowaera.pl/okulary-interaktywne-dla-uczniow-jak-dzialaja-i-czy-warto-z-nich-korzystac-podczas-lekcji`

[7] A. Koszlajda, *Zarządzanie projektami IT. Przewodnik po metodykach*. Helion, 2010.

[8] I. GitHub. "About Git", (**urldate** 2026-01-02). URL: `https://docs.github.com/en/get-started/using-github`

[9] I. GitHub. "About issues", (**urldate** 2026-01-02). URL: `https://docs.github.com/en/issues/tracking-your-work-with-issues/learning-about-issues/about-issues`

[10] G. community. "Branches in a Nutshell", (**urldate** 2026-01-02). URL: `https://git-scm.com/book/en/v2/Git-Branching-Branches-in-a-Nutshell`

[11] I. GitHub. "About rulesets", (**urldate** 2026-01-02). URL: `https://docs.github.com/en/repositories/configuring-branches-and-merges-in-your-repository/managing-rulesets/about-rulesets`

[12] I. GitHub. "Pull requests", (**urldate** 2026-01-02). URL: `https://docs.github.com/en/pull-requests/collaborating-with-pull-requests`

[13] G. community. "Git LFS", (**urldate** 2026-01-03). URL: `https://git-lfs.com/`

[14] P. Kruchten, *The Rational Unified Process: An Introduction*, 3 wyd. Addison-Wesley Professional, 2003.

[15] M. Roman. "Kanban (agile)", (**urldate** 2026-01-02). URL: `https://mfiles.pl/pl/index.php/Kanban_(agile)`

SPIS RYSUNKÓW

SPIS TABEL