

Gliwice, 06.03.2018 r.

Projekt z Grafiki Komputerowej

Dokument projektowy

Tytuł projektu:
Gra 2D typu Tower Defence

Prowadzący:
dr Ewa Lach

Skład sekcji:
Karol Marszałek
Błażej Moska
Kamil Janas

1. Treść zadania

Gra typu Tower Defence 2D zrealizowana z wykorzystaniem języka C++ oraz bibliotek graficznych umożliwiających tworzenie gier komputerowych w tym języku. Gra opiera się na popularnym modelu gier typu Tower Defence polegającym na obronieniu pewnego kluczowego punktu na mapie, na przykład zamku, przed nadchodzącymi przeciwnikami. Przeciwnicy poruszają się po ściśle określonej ścieżce na mapie, na pozostałych polach użytkownik może ustawiać wieże eliminujące przeciwników w drodze do zamku. Rozgrywka kończy się wygraną gracza jeżeli pokona on wszystkich przeciwników lub jego porażką jeżeli przeciwnicy zniszczą zamek.

2. Analiza zadania:

- 1. Podstawy teoretyczne problemu:**
- 2. Wykorzystane zagadnienia grafiki komputerowej:**
 1. Przekształcenia afiniczne
 2. Shadery
 3. Wykrywanie kolizji
- 3. Wykorzystane biblioteki i narzędzia programistyczne:**
 1. Visual Studio jako rozbudowane narzędzie programistyczne zostanie wykorzystane w celu implementacji projektu z wykorzystaniem języka C++.
 2. Biblioteka SFML - biblioteka graficzna do języka C++ ułatwiająca korzystanie z mechanizmów do tworzenia grafiki komputerowej w wykorzystywanym środowisku programistycznym. SFML wprowadza podejście obiektowe do programowania efektów graficznych co pozwala na lepszą integrację warstwy graficznej programu z resztą programu napisaną z wykorzystaniem podejścia programowania zorientowanego obiektowo. Dodatkowo biblioteka SFML zapewnia multiplatformowość tworzonych z jej wykorzystaniem programów. Jest ona zorientowana w celu optymalizacji grafiki dwuwymiarowej, w związku z czym nie wspiera ona mechanizmów do tworzenia grafiki trójwymiarowej. Jest to ograniczenie, które nie wpływa na specyfikację projektu ze względu na wybór implementacji grafiki dwuwymiarowej. Biblioteka SFML nie jest przystosowana do prostego tworzenia graficznego interfejsu użytkownika w przeciwieństwie do, na przykład, biblioteki QT, jednak jej możliwości pozwalają na tworzenie intuicyjnego interfejsu gry komputerowej.
- 4. Algorytmy, struktury danych, ograniczenia specyfikacji**
 - 1. Struktury danych:**
 1. Klasa opisująca mapę zrealizowana wewnętrznie w postaci tablicy jednowymiarowej.
 2. Klasa umożliwiająca przechowywanie historii wyników gracza zapisywanych na dysku twardym użytkownika, wczytywana do aplikacji przy jej uruchomieniu do listy.
 3. Klasa przechowująca konfigurację klas graficznych, w tym zbiór tekstur wykorzystywanych przez grę w tablicy.
 4. Klasa przechowująca informacje o znajdujących się na mapie przeciwnikach implementowana z wykorzystaniem tablicy jednowymiarowej.
 5. Klasa przechowująca informacje o znajdujących się na mapie wieżach ustawionych przez użytkownika w postaci tablicy jednowymiarowej.
- 5. Algorytmy:**
 1. Algorytm wyboru celu wieży na mapie na podstawie jej zasięgu oraz położenia przeciwników na mapie.
 2. Algorytm wyznaczania trajektorii pocisku.
 3. Algorytm wyznaczania trasy przejścia przeciwników po mapie.

6. Ograniczenia specyfikacji:

1. Rozmiar planszy - minimalnie: 250x250 px, maksymalnie 500x500 px.
2. Ograniczenie grafiki do grafiki dwuwymiarowej.
3. Gra przeznaczona jest dla jednego użytkownika bez rozróżnienia użytkowników lokalnych.

3. Plan pracy

1. Stworzenie dokumentacji. - 1 dzień deadline: 23/03/2018
2. Przygotowanie środowiska programistycznego oraz repozytorium. - 1 dzień
data ukończenia: 23/03/2018
3. Stworzenie projektu odpowiedzialnego za grafikę oraz działającej pustej aplikacji graficznej. - 1 dzień deadline: 24/03/2018
4. Stworzenie klasy przechowującej globalne ustawienia grafik, między innymi ścieżki tekstur. - 1 dzień **data ukończenia: 24/03/2018**
5. **Stworzenie zestawu klas do obsługi mapy gry - kamień milowy**
data ukończenia: 01/04/2018 :
 1. Stworzenie klasy implementującej mapę gry w formie tablicy. - 1 dzień
 2. Stworzenie klasy rysującej mapę na podstawie tablicy. - 1 dzień
 3. Integracja klas wymienionych w podpunktach 1), 2). - 1 dzień
 4. Implementacja reguł ścieżek przechodzenia planszy przez przeciwników w przypadku rozgałęzienia ścieżek. - 2 dni
 5. Przetestowanie działania mapy. - 1 dzień
6. **Stworzenie zestawu klas do obsługi wież oraz przeciwników na mapie. - kamień milowy, data ukończenia: 10/04/2018 :**
 1. Obsługa wież na planszy - umieszczenie, wykrywanie kolizji. - 1 dzień
 2. Obsługa przeciwników na planszy - rysowanie przeciwników oraz przejście po planszy zgodnie z regułami zdefiniowanymi na planszy. - 2 dni
 3. Wykrywanie kolizji pomiędzy przeciwnikami oraz odczytanie reguł przejścia po planszy w przypadku rozgałęzień ścieżek. - 2 dni
 4. Przetestowanie działania stworzonych funkcjonalności oraz ich integracji z wcześniejszymi etapami projektu. - 2 dni
7. **Stworzenie zestawu klas odpowiedzialnych za interakcje wież oraz przeciwników (wyznaczanie celu oraz trajektorii pocisku) - kamień milowy, data ukończenia: 26/04/2018 :**
 1. Implementacja algorytmu wyboru celu dla wieży. - 1 dzień
 2. Narysowanie pocisków na mapie. - 1 dzień
 3. Implementacja algorytmu obliczającego trajektorie pocisków. - 2 dni
 4. Implementacja kolizji pocisku oraz przeciwnika. - 1 dzień
 5. Implementacja usuwania pokonanych przeciwników z mapy. - 1 dzień
 6. Przetestowanie działania stworzonych funkcjonalności oraz ich integracji z wcześniejszymi etapami projektu. - 2 dni
8. Stworzenie zestawu klas zarządzających pojedynczą rozgrywką (rozpoznanie końca gry oraz warunków początkowych). - 1 dzień **data ukończenia: 08/05/2018**
9. Obsługa statystyk gracza. - 1 dzień **data ukończenia: 13/05/2018**
10. Przetestowanie działania programu. - 2 dni **data ukończenia: 15/05/2018 – finalna wersja beta**

4. Wstępny podział pracy między osoby w zespole:

1. Stworzenie dokumentacji - Karol Marszałek, Kamil Janas, Błażej Moska
2. Punkty 2, 3 - Karol Marszałek
3. Punkt 4 - Kamil Janas
4. Punkt 5, 1) Karol Marszałek; 2) Kamil Janas; 3,4) Karol Marszałek
5. Punkt 6, 1) Kamil Janas, 2) Błażej Moska, 3) Karol Marszałek
6. Punkt 7, 1) Błażej Moska, 2) Karol Marszałek, 3) Kamil Janas, Karol Marszałek,
7. 4) Błażej Moska, 5) Karol Marszałek, Błażej Moska
8. Punkt 8, Karol Marszałek
9. Punkt 9, Błażej Moska
10. Testowanie działania kolejnych kamieni milowych oraz finalne testowanie programu :
Karol Marszałek, Błażej Moska, Kamil Janas.

5. Link do repozytorium

<https://github.com/karmar1995/GK.git>