Wydział Matematyki i Nauk Informacyjnych Politechniki
Warszawskiej

Symulacja wyścigów samochodowych w 3D

Grafika Komputerowa I

Autor: **Maciej Grzeszczak**

v1.1

10 grudnia 2016r.

Spis treści

Tablica 1: Lista zmian

Data	Autor	Opis zmiany	Wersja
17.12.2016	Maciej Grzeszczak	Opis architektury oraz rozwinięcie opisu biznesowego	1.1
10.12.2016	Maciej Grzeszczak	Pierwsza wersja dokumentu	1.0

1. Specyfikacja

1.1. Opis biznesowy

Niniejszy program to aplikacja przeglądarkowa, wykorzystująca technologię WebGL w postaci biblioteki three.js. Będzie ona zoptymalizowana głównie pod przeglądarkę Mozilla Firefox.

Aplikacja jest symulacją wyścigów samochodowych w 3D. Umożliwia użytkownikowi prowadzenie pojazdu za pomocą odpowiednich klawiszy i ściganie się z samochodami sterowanymi przez komputer. Dany będzie jeden, z góry określony tor po którym będzie można jeździć, sam gracz będzie mógł również go opuścić i poruszać się po całej mapie. Samochód będzie przyspieszał, hamował i skręcał w zależności od wciśniętych klawiszy, będzie można również włączyć bieg wsteczny.

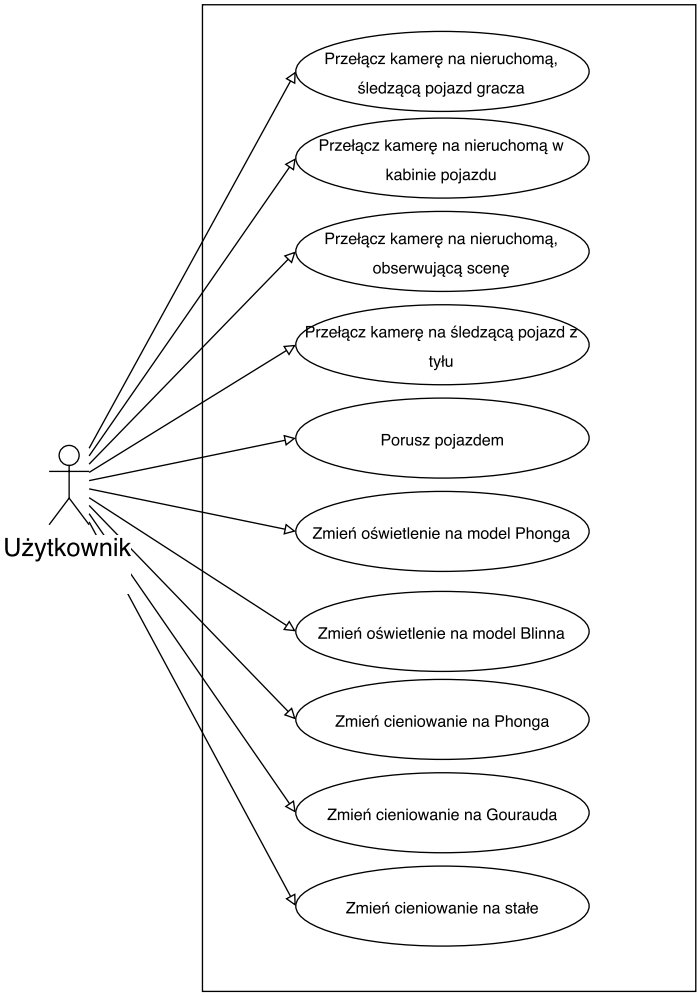
Aplikacja służy również jako pokaz różnych modeli oświetlenia oraz cieniowania. Gracz będzie mógł za pomocą odpowiednich przycisków zmienić obecny model oświetlenia na model Phonga lub Blinna, jak również wybrać jeden z trzech trybów cieniowania (Phonga, Gourauda, stałe).

Oprócz tego, dostępne będą cztery kamery, umożliwiające obserwowanie wyścigów z różnych perspektyw, między innymi ze środka kabiny oraz zza pojazdu, która będzie płynnie poruszać się z pojadem, wykorzystując do tego interpolację.

W planach jest również wprowadzenie większej ilości torów, możliwości ich wczytywania z tekstur, wprowadzenie nierówności terenu i dostosowywania się auta do nich, oraz użycie oddzielnych modeli do kół pojazdów, co umożliwiłoby ich kręcenie się oraz skręcanie.

1.2. Wymagania funkcjonalne

Poniższy rysunek w postaci diagramu UML przedstawia możliwe przypadki użycia systemu przez użytkownika:


Użytkownik
Przełącz kamerę na nieruchomą, śledzącą pojazd gracza
Przełącz kamerę na nieruchomą w kabinie pojazdu
Przełącz kamerę na nieruchomą, obserwującą scenę
Przełącz kamerę na śledzącą pojazd z tyłu
Porusz pojazdem
Zmień oświetlenie na model Phonga
Zmień oświetlenie na model Blinna
Zmień cieniowanie na Phonga
Zmień cieniowanie na Gourauda
Zmień cieniowanie na stałe

Tablica 2: Opisy przypadków użycia dla użytkownika

Aktor	Nazwa	Opis	Odpowiedź systemu
Użytkownik	Przełącz kamerę na nieruchomą, z kabiny pojazdu.	Zmiana położenia kamery na kabinę pojazdu, skierowaną na drogę przed pojazdem.	Natychmiastowa zmiana pozycji kamery.
	Przełącz kamerę na nieruchomą, obserwującą całą scenę.	Zmiana położenia kamery na pozycję umożliwiającą obserwowanie całej sceny z oddali.	Natychmiastowa zmiana pozycji kamery.
	Przełącz kamerę na nieruchomą, śledzącą pojazd gracza.	Zmiana położenia kamery na pozycję, która jest nieruchoma i śledzi pojazd gracza.	Natychmiastowa zmiana pozycji kamery.
	Przełącz kamerę na śledzącą pojazd z tyłu.	Zmiana położenia kamery na pozycję za pojazdem, która porusza się za pojazdem płynnie i wykorzystuje do tego interpolację	Natychmiastowa zmiana pozycji kamery.
	Porusz pojazdem.	Przemieszczenie się pojazdu pod wpływem wciśnięcia odpowiednich klawiszy.	Przemieszczenie się pojazdu.
	Zmień oświetlenie na model Phonga.	Zmiana obecnego modelu oświetlenia na model Phonga.	Natychmiastowa zmiana modelu oświetlenia na model Phonga.
	Zmień oświetlenie na model Blinna.	Zmiana obecnego modelu oświetlenia na model Blinna.	Natychmiastowa zmiana modelu oświetlenia na model Blinna.
	Zmień cieniowanie na Phonga.	Zmiana obecnego trybu cieniowania na cieniowanie Phonga.	Natychmiastowa zmiana trybu cieniowania na cieniowanie Phonga.
	Zmień cieniowanie na Gourauda.	Zmiana obecnego trybu cieniowania na cieniowanie Gourauda.	Natychmiastowa zmiana trybu cieniowania na cieniowanie Gourauda.
	Zmień cieniowanie na stałe.	Zmiana obecnego trybu cieniowania na stałe.	Natychmiastowa zmiana trybu cieniowania na cieniowanie stałe.

1.3. Wymagania niefunkcjonalne

Poniżej przykładowe wymagania niefunkcjonalne pogrupowane w poszczególne kategorie URPS.

Tablica 3: Lista wymagań niefunkcjonalnych

Obszar wymagań	Lp	Opis
Użyteczność	1	Aplikacja będzie działała na przeglądarce Mozilla Firefox dla każdej rozdzielczości powyżej 800x600.
Niezawodność	2	Aplikacja będzie dostępna 24/7 pod podanym adresem.
Wydajność	3	Aplikacja będzie utrzymywać minimalny poziom 15 FPS (klatek na sekundę).
Utrzymanie	4	Wraz z aplikacją zostaje dostarczona instrukcja użytkownika.

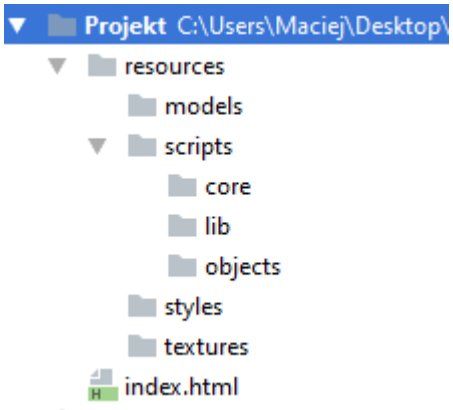
1.4. Harmonogram projektu

Implementacja projektu zostanie podzielona na dwie fazy:

1. **Faza tworzenia sceny (14 dni)** - stworzenie świata wraz z obiektami (pojazdami), implementacja poruszania się pojazdem oraz poruszania się i zmiany pozycji kamery.

2. **Faza implementacji poszczególnych modelów oświetlenia oraz cieniowania (7 dni)** - implementacja modeli oświetlenia Phonga i Blinna oraz cieniowań: stałego, Phonga i Gourauda.

1.5. Architektura rozwiązania



Powyższe zdjęcie przedstawia szkielet architektury projektu. Poniżej znajduje się opis kolejnych elementów.

1. **index.html** - główny i jedyny plik html, w którym zagnieżdżony będzie HTML5 Canvas.
2. **resources** - folder, w którym znajdują się wszystkie zasoby wykorzystywane przez aplikację:
   (a) models - folder z modelami 3D używanymi przez aplikację
   (b) scripts - folder ze skryptami Javascript
      i. core - folder zawierający pliki .js dotyczące szkieletu działania aplikacji, czyli pętli gry, liczenia fizyki, renderowania, obsługiwania interakcji użytkownika.
      ii. objects - folder w którym znajdują się pliki .js definiujące wszystkie obiekty wykorzystywane w aplikacji, między innymi pojazdu, kamery, mapy.
      iii. lib - folder w którym znajdują się pliki .js zewnętrznych bibliotek.
   (c) styles - folder z wszystkimi plikami .css
   (d) textures - folder z teksturami wykorzystywanymi przez aplikację

Oprócz tego projekt będzie oparty na wzorcu modułów, który pomaga w organizacji całości kodu oraz uzyskania tzw. 'loose coupling', czyli jak najmniejszej zależności pomiędzy poszczególnymi częściami kodu.