Proceduralne generowanie planet w czasie rzeczywistym
w dynamicznie zmieniającej się skali

Dokumentacja techniczna

MATEUSZ CHECHLIŃSKI

Wersja 1.0

27 listopada 2014

Modyfikacje dokumentu			
Wersja	Data	Autor	Opis
1.0	27 listopada 2014	Mateusz Chechliński	Iteracja nr 1.

Spis treści

1. Słownik pojęć

Shader - krótki program komputerowy, który oblicza właściwości pikseli oraz wierzchołków.

Uniform - globalna zmienna GLSL zadeklarowana ze ze słowem kluczowym „uniform". Wykorzystywana w roli atrybutu przekazywanego do shadera.

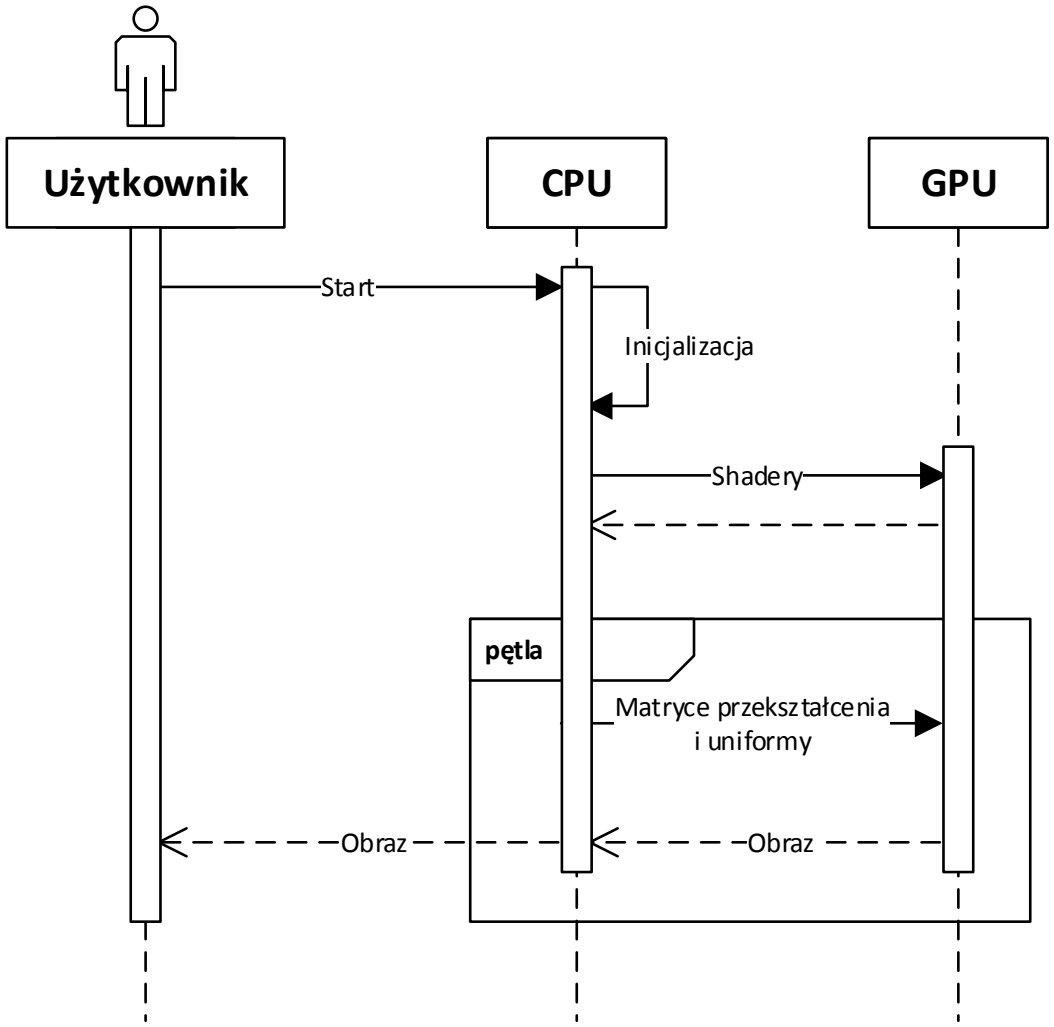
Renderowanie - (od ang. *rendering*), analiza modelu danej sceny oraz utworzenie na jej podstawie dwuwymiarowego obrazu wyjściowego w formie statycznej lub animacji.

2. Wstęp

Niniejszy dokument stanowi dokumentację techniczną aplikacji do proceduralnego generowania planet w czasie rzeczywistym w dynamicznie zmieniającej się skali.

3. Iteracja nr 1

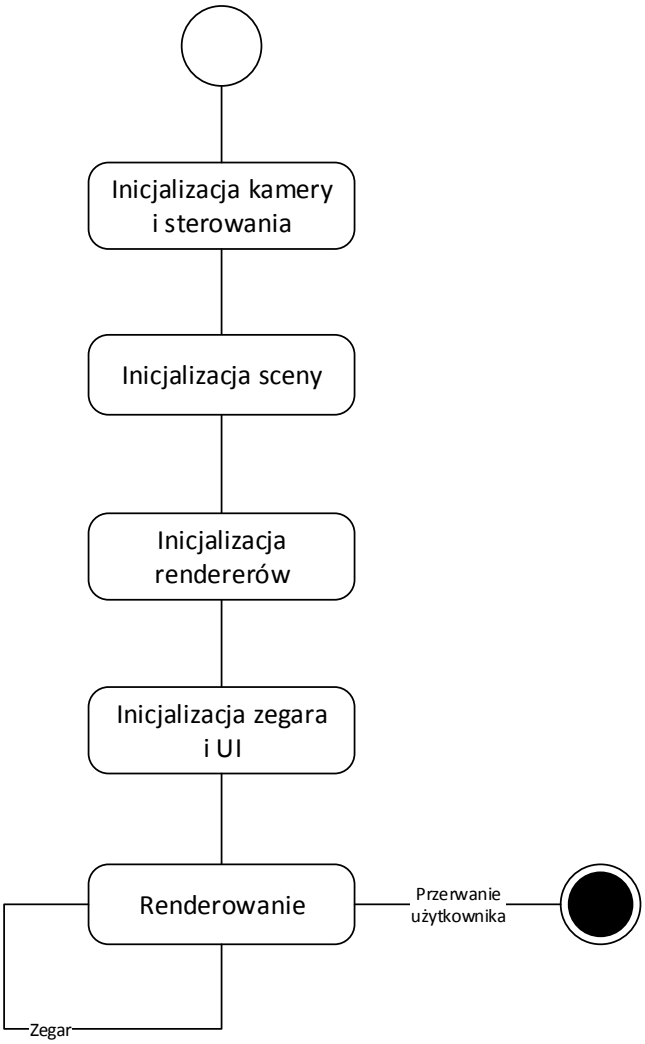
3.1 CPU - GPU



Rys. 1 - Schemat komunikacji CPU - GPU

Po uruchomieniu aplikacji (otwarciu strony w przeglądarce) wykonana zostanie inicjalizacja po stronie CPU. Potem załadowane do GPU zostaną shadery. Wówczas rozpoczyna się pętla, w której CPU przekazuje do GPU matryce przekształceń oraz uniformy, a GPU renderuje na tej podstawie obraz, który następnie jest prezentowany użytkownikowi.

3.2 Inicjalizacja



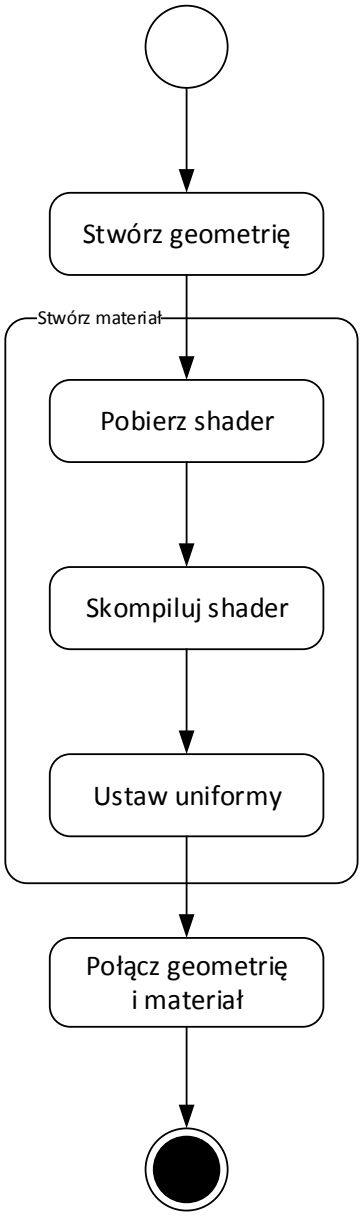
Rys. 2 - Inicjalizacja aplikacji

Powyższy schemat przedstawia kolejność, w jakiej inicjalizowane są poszczególne elementy systemu.

W tej części, spore zastosowanie ma biblioteka Three.js, która pozwala całą inicjalizację zawrzeć w zaledwie kilku liniach kodu.

Po zakończeniu inicjalizacja, aplikacja przechodzi do pętli renderującej, która przerwana może zostać tylko poprzez zakończenie pracy aplikacji.

3.3 Generowanie planet i gwiazd



Rys. 3 - Schemat generowania planet i gwiazd

Powyższy schemat przedstawia kolejne kroki związane z wygenerowaniem planety lub gwiazdy. W pierwszej kolejności tworzona jest geometria (sfera). Następnie pobierane i ładowane do GPU są shadery (Vertex i Fragment Shader), po czym ustalane są wartości uniformów (co do reguły, ich wartość nie będzie się zmieniać - poza uniformem czasu).

Następnie tak utworzone dane są łączone i tworzą planetę lub gwiazdę.

3.4 Metody

3.4.1 PlanetManager

Metoda	Opis
startTime()	Oblicza Time Uniform
planetTypes	Słownik trzymający dane o dostępnych typach planet
updateAnimatedUniforms(material, animatedUniforms)	Metoda odświeżająca uniformy oparte na funkcjach
createMaterial(data)	Tworzy materiał na podstawie danych. Kopiuje materiał zawierający shadery oraz zapełnia ich uniformy.
createPlanet(name)	Tworzy planetę wybranego typu (lub gwiazdę)
update()	Odświeża planety

3.4.2 Resouces

Metoda	Opis
getShaderMaterial(name)	Pobiera oba shadery razem
getTexture(name)	Pobiera plik tekstury

3.4.3 StaticLoader

Metoda	Opis
ajaxGet(path)	Wysyła zapytania AJAX
loadShader(name)	Pobiera wskazany shader