POLITECHNIKA WROCŁAWSKA
WYDZIAŁ ELEKTRONIKI

KIERUNEK: INFORMATYKA
SPECJALNOŚĆ: GRAFIKA I SYSTEMY MULTIMEDIALNE

PRACA DYPLOMOWA INŻYNIERSKA

Interactive 3D visualization of geographical data based on open sources using web technologies.

Interaktywna wizualizacja 3D danych geograficznych z otwartych źródeł z wykorzystaniem technologii webowych.

AUTOR:
Damian Koper

PROWADZĄCY PRACĘ:
Dr inż. Marek Woda

OCENA PRACY:

WROCŁAW, 2020

Spis treści

Spis rysunków

Spis tabel

Spis listingów

Skróty

GIS (ang. *Geographic Information System*)
API (ang. *Application Programming Interface*)

Rozdział 1

Wstęp

Rzeczywistość otaczająca człowieka i jej aspekty są bardzo złożonym zagadnieniem. Człowiek w procesie jej poznawania może postawić się w różnych punktach odniesienia. Może obserwować rzeczywistość w skali wszechświata badając i poszerzając wiedzę na temat galaktyk oraz innych ciał niebieskich, gdzie Ziemia jest pomijalnie małym elementem. Może również obserwować świat w skali makro i mikroskopowej skupiając się na organizmach zamieszkujących i strukturach budujących planetę, schodząc również na poziom atomów i kwarków.

Większość obserwacji nie może być dokonana bezpośrednio przez człowieka. Nie może on bowiem objąć wzrokiem całek galaktyki, albo dostrzec poszczególnych atomów. Obrazowanie takich zjawisk musi być zaprezentowane w formie przystępnej dla człowieka wizualizacji zbudowanej z uwzględnieniem konkretnych aspektów danego przypadku.

Dobrze zbudowana wizualizacja danych, jaką jest chociażby prosty wykres punktowy, pozwala na analizę danych w lepszym stopniu i łatwiejsze wyciągnięcie wniosków. Dobrze skonstruowana wizualizacja, w przypadku prezentacji jej większemu gronu odbiorców, pozwala również na skuteczniejsze zainteresowanie grupy tematem oraz pomóc w opowiedzeniu historii, a co za tym idzie, na wyciągnięcie przez odbiorców właściwych wniosków [5].

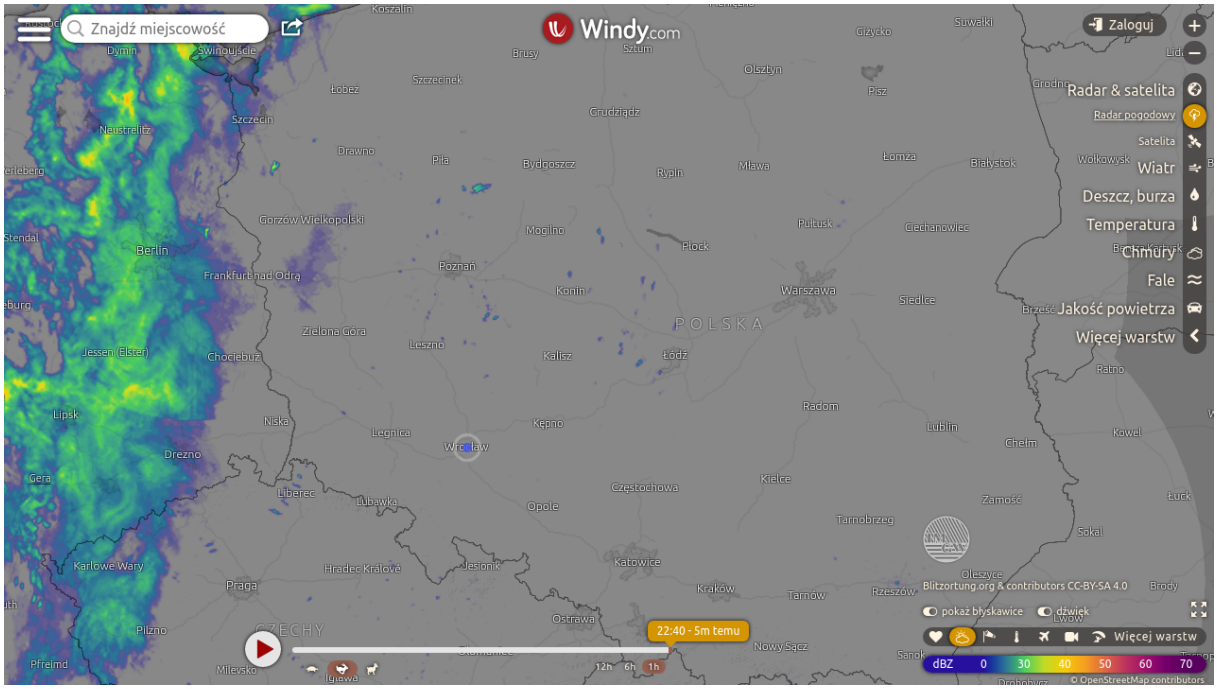
1.1. Istota rzeczy

Jednym z obszarów, w którym wizualizacje pełnią istotną rolę są reprezentacje zjawisk geograficznych oraz tych w bliskim sąsiedztwie Ziemi. Prezentowane dane mogą być związane z działalnością człowieka, bądź z obiektami i zjawiskami fizycznymi, którymi planeta się cechuje.

System, który zajmuje się wprowadzaniem, analizowaniem i wizualizacją danych geograficznych jest nazywa się Systemem informacji geograficznej (ang. geographic information system, **GIS**). Może on wyświetlać informacje z wielu źródeł ujęte w warstwy, które wyświetlane razem w różnych kombinacjach mogą nadawać danym różnego kontekstu. Każda wyświetlana informacja jest ściśle powiązana z pozycją na powierzchni Ziemi. [4, Rozdział 1.6].

Wizualizacje danych mogą tyczyć się dowolnych zjawisk. GIS może obrazować podział terytorialny państw świata, jak i położenie obiektów kosmicznych w bliskim sąsiedztwie Ziemi. Korzystając z aktualizowanych na bieżąco źródeł danych wizualizacje mogą obrazować zjawiska zmienne, pokazywać stan obecny, przeszły (Rysunek 1.1), jak i prognozować przyszłość.

Ważnym czynnikiem odbiorze wizualizacji, jest jej przystępność dla użytkownika. Profesjonalne, skomplikowane systemy nierzadko cechują się złożonym interfejsem użytkownika. Duża liczba opcji pomaga łatwo uzyskać pożądane dane przez profesjonalnego użytkownika, ale odstraszyć może niezagłębionego w temat odbiorcę. Przystępność odbioru wiąże się również z szybkością uzyskania dostępu do samej platformy obsługującej wizualizację. Alternatywą



Rys. 1.1: Widok wizualizacji dwuwymiarowej na stronie *windy.com* wyświetlający informacje pogodowe na dwuwymiarowej mapie

dla instalowanych aplikacji desktopowych jest przeglądarka internetowa. Tworzy ona środowisko, które może być uruchomione na wielu systemach operacyjnych, również na urządzeniach mobilnych, a zaimplementowane wspomaganie sprzętowe generowania grafiki i interfejsy takie jak *HTML5 Canvas*[2] i *WebGL*[3] czynią ją potężnym narzędziem do wydajnego wyświetlania złożonych grafik. Aplikacje webowe oczywiście nie będą nigdy dorównywać profesjonalnym aplikacjom dedykowanym konkretnej platformie, jednak stanowią ich dobrą i ogólnodostępną alternatywę.

Innym kryterium definiującym wizualizację jest jej interaktywność. Definiuje ono w jakim stopniu użytkownik może dostosować wyświetlany widok, zarządzać warstwami, sterować położeniem kamery, czy też wyszukiwać informacje. Dwuwymiarowy widok mapy (Rysunek 1.1) pozwala jednoznacznie odnieść informacje z różnych warstw do konkretnego miejsca na planecie. Z kolej widok trójwymiarowy (Rysunek 1.2) pozwala na obserwację sceny z różnych perspektyw, pokazuje kulistość Ziemi i redukuje efekty zniekształcenia danych związany z techniką rzutowania sfery na płaszczyznę. Przy kamerze skierowanej prostopadle do płaszczyzny powierzchni, oraz w bliskim powiększeniu widok taki jest porównywalny do widoku dwuwymiarowego. Czynniki te zdaniem autora pracy czynią taką wizualizację bardziej atrakcyjną dla ogólnego odbiorcy. Oczywiście wybór techniki wizualizacji zawsze zależy od konkretnego przypadku, jak i od oczekiwanej wydajności, gdyż złożoność generowania grafiki w przypadku wizualizacji trójwymiarowych jest z reguły większa.

1.1.1. Podejścia do tworzenia wizualizacji

Zadaniem twórcy wizualizacji jest zebranie i przetworzenie danych na formę grafiki 2D lub 3D. Od używanego systemu informacji przestrzennej zależy w jaki sposób definiowana jest wizualizacja i skutkiem tego, jaki poziom wiedzy i umiejętności z danej dziedziny jest potrzebny do jej stworzenia. System w definicji wizualizacji opiera się na swoich założeniach. Aplikacje uruchamiane bezpośrednio w środowisku systemu operacyjnego mogą być wyposażone w rozbudowane kreatory i edytory, które zaspokajają wymagania użytkowników. Pozwalają skupić

Rys. 1.2: Widok wizualizacji trójwymiarowej na stronie *earth.google.com*

się na zagadnieniach domenowych, na poprawności i dokładności wizualizacji zamiast na aspektach generowania grafiki.

W środowisku przeglądarki internetowej do tworzenia wizualizacji nie stosuje się zwykle rozbudowanych edytorów graficznych i formularzy. Biblioteki wyświetlające dane geoprzestrzenne konfigurowalne są zwykle z poziomu języka JavaScript. Przykładem takiej biblioteki jest Cesium[1]. Potrafi ona generować wizualizacje 2D i 3D różnego rodzaju danych, a jej konfiguracja następuje poprzez jej API, które dostarcza, ale też ogranicza jej możliwości (listing 1.1).

Listing 1.1: Konfiguracja podstawowej wizualizacji w bibliotece Cesium. Źródło: `https://cesium.com/docs/tutorials/getting-started/`

```
<script>
    Cesium.Ion.defaultAccessToken = 'your_access_token';
    var viewer = new Cesium.Viewer('cesiumContainer', {
        terrainProvider: Cesium.createWorldTerrain()
    });

    var tileset = viewer.scene.primitives.add(
        new Cesium.Cesium3DTileset({
            url: Cesium.IonResource.fromAssetId(your_asset_id)
        })
    );
    viewer.zoomTo(tileset);
</script>
```

Jeszcze innym podejściem, możliwym do zastosowania w przypadku aplikacji i desktopowych, i webowych, jest dostarczenie twórcy tylko podstawowych abstrakcji (najczęściej interfejsów programistycznych) wizualizacji takich jak sterowanie kamerą, przekazywanie zdarzeń pochodzących od odbiorcy, czy interfejs służący do generowania obiektów na scenie 2D lub 3D. Podejście daje to najwięcej możliwości, ale z drugiej strony wymaga posiadania największej wiedzy o funkcjonowaniu dostarczonych interfejsów.

W każdym wypadku istotnym czynnikiem ułatwiającym tworzenie wizualizacji jest dostarczona przez narzędzia i biblioteki interaktywna dokumentacja. Powinna ona dobrze opisywać dostarczone rozwiązania ze strony praktycznej i przez swoją interaktywność ułatwiać poruszanie się po niej użytkownikowi.

1.2. Cel i zawartość pracy

Rozdział 2

Wymagania

Ze względu na możliwy podział funkcjonalności projektu na wiele typów, zdefiniowano następujące pojęcia:

1. Silnik - zbiór komponentów odpowiedzialnych za definicję i wyświetlenie wizualizacji.
2. Wizualizacja - konfigurowalny widok przedstawiający obiekty, których położenie zdefiniowano za pomocą współrzędnych geograficznych, na powierzchni sfery.
3. Aplikacja - uruchomiona w przeglądarce użytkownika strona umożliwiająca wybór i wyświetlenie wizualizacji.

Silnik dostarcza komponenty i interfejs programistyczny, dzięki którym można definiować, wyświetlać i zarządzać wizualizacją. Pozwala także na zdefiniowane wielu niezależnych wizualizacji. Z tego powodu można wyróżnić dwa typy użytkowników:

1. Twórcę wizualizacji,
2. Odbiorcę wizualizacji.

Wymagania aplikacji zostały zdefiniowane z podziałem na typ użytkownika. Struktura danych definiująca renderowany obraz, zwana dalej będzie sceną.

2.1. Twórca wizualizacji

2.1.1. Wymagania funkcjonalne

1. Twórca może zdefiniować metadane wizualizacji określone przez interfejs Silnika.
2. Twórca może zdefiniować statyczną scenę określając położenie obiektów na sferze z wykorzystaniem długości i szerokości geograficznej.
3. Twórca do definicji sceny może wykorzystać interfejs tworzenia obiektów dostarczony przez aplikację lub załadować obiekty, materiały i tekstury z zewnętrznego źródła.
4. Twórca może zagnieżdżać sceny predefiniowane w silniku, oraz sceny wcześniej stworzonych przez siebie.
5. Twórca może parametryzować sceny w celu określonej ich modyfikacji w procesie zagnieżdżania.
6. Twórca może określić parametry początkowe obserwatora, dynamikę i zakres jego ruchów:
    1. położenie,
    2. prędkość i przyspieszenie ruchu,
    3. ograniczenie przybliżenia,
    4. ograniczenie pozycji.

7. Twórca może zdefiniować wygląd i funkcjonalność panelu kontrolnego. Panel ten służyć będzie do zmiany parametrów wizualizacji i obsługiwany będzie przez odbiorcę.
8. Twórca, poprzez interfejs programistyczny dostarczony przez silnik, może aktualizować scenę w dowolnym momencie, określonym przez siebie w definicji wizualizacji.
9. Twórca może definiować zachowania, które będą odpowiedzią na zdarzenia związane z poruszaniem się po scenie generowane przez odbiorcę.

2.1.2. Wymagania niefunkcjonalne

1. Silnik powinien definiować i w sposób jasny przekazywać potencjalnemu twórcy akceptowalną strukturę danych, plików i katalogów, określającą jedną wizualizację.
2. Włączenie zdefiniowanej wizualizacji do ich zbioru w aplikacji powinno ustanowione być tylko w jednym miejscu poprzez prosty interfejs.
3. Dane wizualizacji muszą być ładowane asynchronicznie. Dane źródłowe definiujące scenę mogą być przetwarzane po stronie odbiorcy lub być przetworzone wcześniej i pobrane.

2.2. Odbiorca wizualizacji

2.2.1. Wymagania funkcjonalne

1. Odbiorca może zobaczyć dane dostępnych wizualizacji.
2. Odbiorca może wyświetlić wybraną wizualizację.
3. Odbiorca może poruszać się po wizualizacji, zmieniając położenia kamery, używając myszki lub klawiatury.
4. Odbiorca może zobaczyć orientację kamery relatywnie do kierunku północnego i ją zresetować.
5. Odbiorca może wyświetlić lub ukryć panel sterujący wizualizacją dostarczony przez twórcę.
6. Odbiorca może ustawić poziom szczegółowości grafiki. Ustawienia te przekazywane są twórcy i mogą, ale nie muszą, być respektowane.

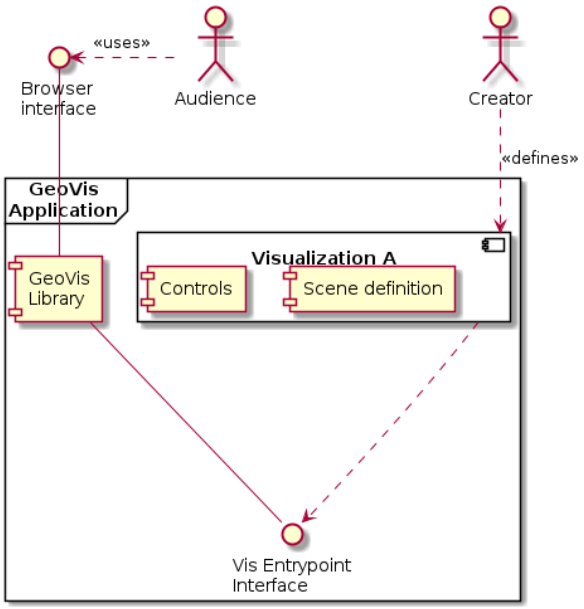
2.2.2. Wymagania niefunkcjonalne

1. Każda akcja użytkownika związana ze sterowaniem kamerą może zostać wykonana używając myszki lub równolegle klawiatury.

2.3. Aplikacja

2.3.1. Wymagania niefunkcjonalne

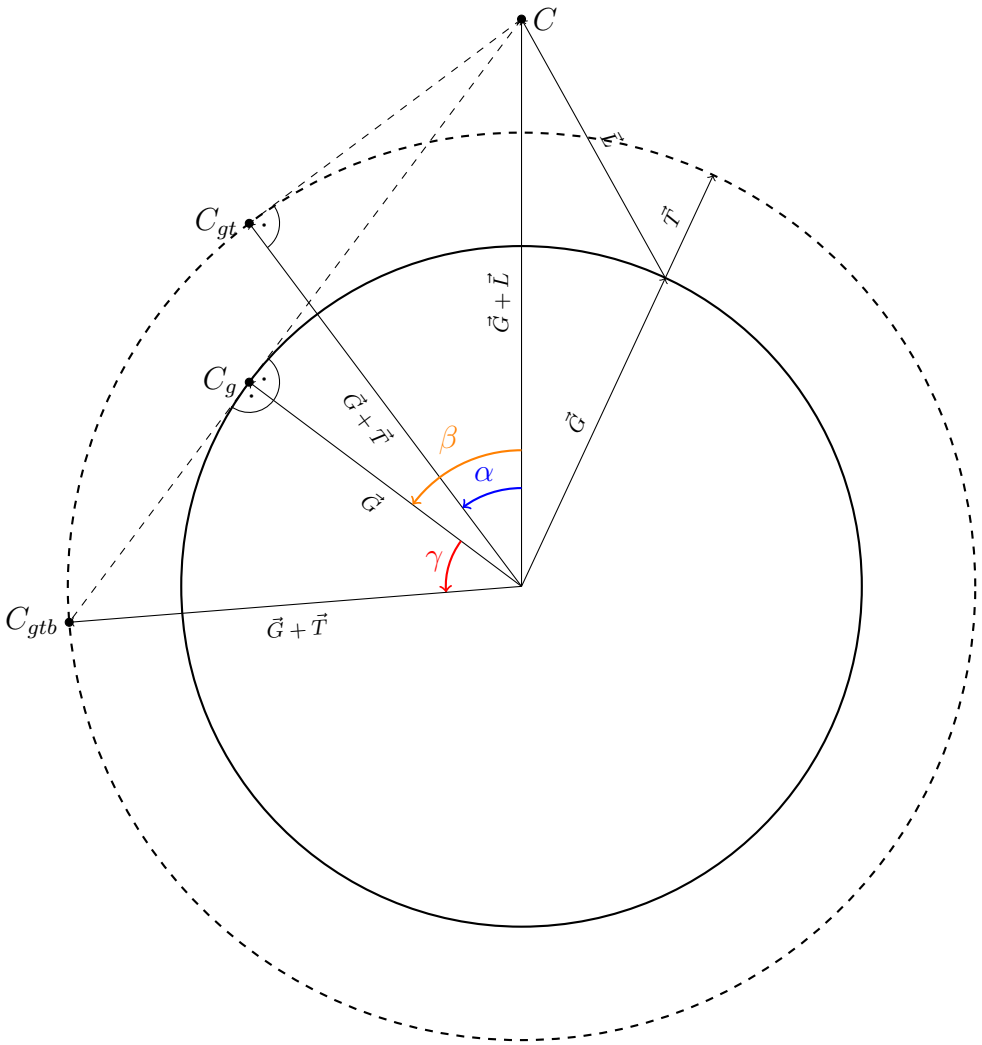
1. Aplikacja powinna być stroną typu *Single Page Application*.
2. Jeśli to możliwe aplikacja powinna wykorzystywać sprzętową akceleracje obliczeń graficznych.
3. Aplikacja powinna ustawiać i obsługiwać adres URL w przeglądarce definiujący wyświetlaną wizualizację.



Rys. 3.1: Schemat ogólny komponentów aplikacji.

Rozdział 3

Architektura aplikacji



Literatura

[1] Cesium. https://github.com/CesiumGS/cesium/. Na dzień: 2020-09-04.

[2] Html living standard - the canvas element. https://html.spec.whatwg.org/multipage/canvas.html. Na dzień: 2020-09-03.

[3] Webgl 2.0 specification. https://www.khronos.org/registry/webgl/specs/latest/2.0/. Na dzień: 2020-09-03.

[4] D. Dorrell, J. Henderson, T. Lindley, and G. Connor. *Introduction to Human Geography (2nd Edition)*. GALILEO, University System of Georgia, 2019.

[5] C. N. Knaflic. *Storytelling with Data*. Wiley John&Sons Inc, 2015.