Multimedia - rzeczywistość rozszerzona i wirtualna

RPGraphics - Dokumentacja projektu

Adam Howaniec 236497
Michał Macikowski 235099
Witek Marciniak 226194
Dawid Piechota 235851

Projekt
Dr inż. Marek Woda

Politechnika Wrocławska
Wydział Elektroniki

Spis treści

1 Wstęp

W ramach zajęć projektowych z przedmiotu *"Multimedia - rzeczywistość rozszerzona i wirtualna"* sporządzono w grupie czteroosobowej, o składzie widocznym na stronie tytułowej, aplikację z dziedziny rzeczywistości rozszerzonej.

Niniejszy dokument streszcza założenia projektu, metodę jego implementacji oraz rezultat końcowy.

2 Założenia projektowe

W ramach projektu podjęto się sporządzenia aplikacji z dziedziny Rzeczywistości Rozszerzonej, modyfikującej obraz pobierany z urządzenia wejściowego w postaci kamerki internetowej, celem wzbogacenia jego treści o elementy użyteczne w procesie prowadzenia tzw. gier fabularnych.

2.1 Funkcjonalności podstawowe

Na podstawowe funkcjonalności projektu składają się dwie główne kategorie:

1. Modyfikacja twarzy użytkownika
2. Modyfikacja otoczenia użytkownika
3. Wyświetlanie informacji towarzyszących

Modyfikacja twarzy użytkownika polegać ma na umożliwieniu zamiany twarzy na twarz "awatara" stanowiącego postać reprezentowaną przez gracza w grze fabularnej, lub jedną z postaci niezależnych reprezentowanych przez prowadzącego rozgrywkę.

Modyfikacja otoczenia umożliwić ma substytucję tła tak, aby ułatwić zbudowanie przekonującej atmosfery miejsca akcji opisywanej w narracji rozgrywki.

Przez informacje towarzyszące rozumiany jest tekst o dowolnej zdefiniowanej przez użytkownika treści, jak i zintegrowany interfejs użytkownika pozwalający na sterowanie aplikacją.

2.2 Funkcjonalności rozszerzone

W ramach funkcjonalności rozszerzonych przewidziano opcjonalne rozwinięcia powyższych kategorii o elementy takie jak:

1. Możliwość wgrania ruchomego tła (animacja, wideo).
2. Możliwość wgrania niestandardowej maski twarzy.

3. Nałożenia maski na inną wybraną część ciała. (N.p. element ubioru)

4. Zarejestrowanie zmodyfikowanego obrazu jako nowej wirtualnej kamery.

Funkcja ruchomego tła poszerza możliwości budowania atmosfery poprzez animacje takie jak ruch drzew na wietrze, czy woda kapiąca z sufitu jaskini.

Niestandardowa maska oraz możliwość maskowania innych części ciała poza twarzą, ułatwiają użytkownikowi dobranie wyglądu, który może zidentyfikować z odgrywaną postacią.

Zarejestrowanie obrazu jako kamery wirtualnej natomiast poprawia ogólne doznania użytkownika poprzez ułatwienie obsługi aplikacji i jej współpracę z aplikacjami trzecimi.

3 Realizacja

W niniejszej sekcji streszczony zostaje proces powstawania aplikacji oraz jej struktura, wraz z wykorzystanymi technologiami i narzędziami.

3.1 Technologie

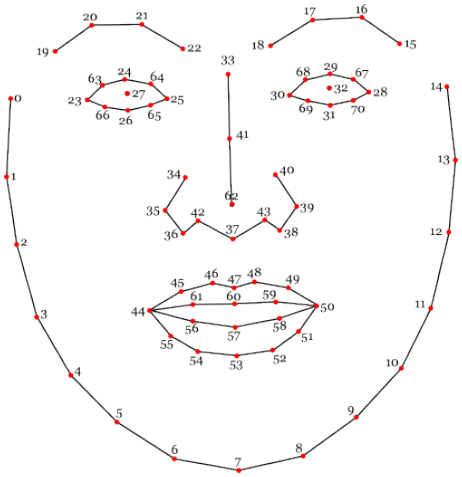
Do zrealizowania rdzenia aplikacji posłużono się środowiskiem uruchomieniowym `Node.js` oraz biblioteką `React`. Decyzja podyktowana została przystępnością rozwiązań, uprzednim doświadczeniem części członków grupy projektowej oraz wymaganiami zastosowanych rozwiązań.

Do implementacji kluczowych funkcjonalności posłużono się odpowiednio niniejszymi otwartymi technologiami:

3.1.1 Podmiana Twarzy - clmtrackr

Clmtrackr jest biblioteką JavaScript wykorzystującą uczenie maszynowe do wykrywania twarzy na zdjęciach i nagraniach wideo w czasie rzeczywistym. W realizacji aplikacji wykorzystano wstępnie wytrenowane modele załączone w biblotece - nauczone na zbiorze danych MUCT.

Clmtrackr wykrywa twarz opisując ją jako zestaw współrzędnych na danej grafice/klatce wideo. Dokładny model wizualizuje zamieszczona Rysunek 1. Na podstawie niniejszego opisu liczbowego możliwe jest dalsze przetwarzanie nabytej informacji - takie jak interpretacja (n.p. wykrywanie emocji), deformacja, czy wreszcie wykorzystana w projekcie substytucja, gdzie na zestaw współrzędnych nakładana jest grafika zastępcza.



Rysunek 1: Model współrzędnych opisujących twarz w clmtrackr

3.1.2 Podmiana tła - BodyPix

BodyPix jest kolejnym zastosowaniem uczenia maszynowego - w tym wypadku dedykowanym wykrywaniu oraz segmentacji sylwetki ludzkiej. BodyPix korzysta z biblioteki `TensorFlow.js` aby zidentyfikować na obrazie lub w klatce wideo model sylwetki wraz z podziałem na części ciała.

Wyróżnić można szerokie możliwości zastosowania tego zabiegu, takie jak zamazywanie twarzy czy wycinanie osób trzecich z ujęcia. W wypadku realizowanej aplikacji wykorzystano tą informację aby odseparować tło od osoby i umożliwić jego podmianę.

Rysunek 2: Przykład działania BodyPix

3.2 Implementacja

Skrypty i komponenty składające się na architekturę aplikacji podzielić można na trzy zasadnicze sekcje opisane poniżej. Opisy zawierają również powiązane próbki kodu.

3.2.1 Kontener główny

Dokonuje wstępnej weryfikacji oraz inicjalizacji aplikacji. Określa czy środowisko użytkownika spełnia niezbędne standardy, takie jak dostępność technologii WebGL czy obecność kamerki internetowej dostępnej dla aplikacji.

W wypadku wszelkiego rodzaju niepowodzenia użytkownik informowany jest o jego powodzie, a dalsze uruchamianie aplikacji jest przerywane.

```
let vid = document.getElementById('videoel');
setVid(vid);
navigator.getUserMedia = navigator.getUserMedia || navigator.
    ↪ webkitGetUserMedia || navigator.mozGetUserMedia ||
    ↪ navigator.msGetUserMedia;
window.URL = window.URL || window.webkitURL || window.msURL ||
    ↪ window.mozURL;
// check for camerasupport
if (navigator.mediaDevices) {
```

```
navigator.mediaDevices.getUserMedia({video : true}).then(
  ( stream ) => {
    let vid = document.getElementById('videoel');
    vid.srcObject = stream;
    vid.onloadedmetadata = () => {
        vid.play();
    }
  }
).catch((reason) => {
    console.log(reason);
});
} else if (navigator.getUserMedia) {
  navigator.getUserMedia({video : true});
} else {
  alert("This demo depends on getUserMedia, which your browser
      ↪ does not seem to support. :(");
}
```

Główny kontener inicjalizacje również wszelkie elementy HTML niezbędne do poprawnego działania dalszych modułów. Składają się na nie płótna (`canvas`), na których operują elementy przetwarzania obrazu czy obrazy masek clmtrackr (o czym więcej w sekcji 3.2.2)

```
<img id="skull" className="hidden" src="./src/components/
    ↪ Clmtrackr/Masks/skullmask.jpg" alt="skull"></img>
<img id="halfElf" className="hidden" src="./src/components/
    ↪ Clmtrackr/Masks/half-elf.jpg" alt="halfElf"></img>
<img id="elf" className="hidden" src="./src/components/Clmtrackr/
    ↪ Masks/elf.jpg" alt="elf"></img>
<img id="orc" className="hidden" src="./src/components/Clmtrackr/
    ↪ Masks/orc.jpg" alt="orc"></img>
```

Ostatnią z funkcji kontenera głównego jest zarządzanie nakładką interfejsu użytkownika umożliwiającą interakcje z funkcjami programu, oraz wyświetlanie wybranego przez użytkownika tekstu ponad generowanym obrazem wyjściowym. Interfejs użytkownika opisany jest szczegółowo w sekcji 4.3

```
/* Text */
useEffect(() => {
if (textValue !== '' ) {
  let txtSettings = {
      text: textValue,
      offsetX: 0,
      offsetY: 200,
      color: textColor,
      font: "32px Consolas",
  }
```

```
  if(!txtSettings.text) return;
  const txtCanvas = document.getElementById('text-canvas');
  let ctx = txtCanvas.getContext('2d');
  ctx.textAlign = "center";
  drawStroked(txtSettings)

  function drawStroked(settings) {
      ctx.font = settings.font;
      ctx.strokeStyle = 'black';
      ctx.lineWidth = 8;
      ctx.lineJoin="miter";
      ctx.miterLimit=2;
      ctx.strokeText(settings.text, txtCanvas.width/2 + settings.
          ↪ offsetX, txtCanvas.height/2 - settings.offsetY);
      ctx.fillStyle = settings.color;
      ctx.fillText(settings.text, txtCanvas.width/2 + settings.
          ↪ offsetX, txtCanvas.height/2 - settings.offsetY);
  }
}
}, [width, height, video, textValue, textColor]);
```

3.2.2 Przetwarzanie clmtrackr

Komponent przetwarzania clm odpowiada za wykrywanie na obrazie twarzy użytkownika oraz jej substytucję na wybraną maskę.

Po poprawnej inicjalizacji obrazu wideo z kamery i umieszczeniu go na elemecie `canvas`, inicjalizowany jest komponent typu `clm.tracker`

```
let ctrack = new clm.tracker({
  faceDetection: {
    useWebWorkers: false,
  },
});
ctrack.init();

ctrack.start(video);
```

Na tym etapie dokonywana jest początkowa analiza obrazu i detekcja twarzy. W przy każdej kolejnej klatce wideo pozycja twarzy aktualizowana jest na podstawie uprzednich wyliczeń, bez każdorazowej detekcji.

Warto nadmienić, że w przeciwieństwie do modułu BodyPix, możliwe jest tu wykrywanie twarzy pojedynczego użytkownika co stanowi dolne ograniczenie aplikacji. Nie odbiega to jednak od założeń projektowych.

Na obliczony zestaw współrzędnych nałożona zostaje następnie wybrana "maska" będąca odpowiednio opisaną teksturą w postaci elementu `<img>`, inicjalizowanego w kontenerze głównym.

```
function drawGridLoop() {
  // get position of face
  positions = ctrack.getCurrentPosition();
  overlayCC.clearRect(0, 0, 640, 480);
  if (positions) {
    // draw current grid
    ctrack.draw(overlay);
  }
  // check whether mask has converged
  var pn = ctrack.getConvergence();
  // console.log(pn);
  if (pn < 1000) {
    switchMasks();
    requestAnimationFrame(drawMaskLoop);
  } else {
    requestAnimationFrame(drawGridLoop);
  }
}

function switchMasks() {
  // get mask
  var maskname = Object.keys(masks)[currentMask];
  console.log(maskname);
  fd.load(document.getElementById(maskname), masks[maskname],
      ↪ pModel);
}

function drawMaskLoop() {
  // get position of face
  positions = ctrack.getCurrentPosition();
  overlayCC.clearRect(0, 0, 640, 480);
  if (positions) {
    // draw mask on top of face
    fd.draw(positions);
  }
  animationRequest = requestAnimationFrame(drawMaskLoop);
}
```

Ze względu na charakter zastosowania aplikacji przewiduje się, że maski stanowić będą głównie portrety lub inne przedstawienie postaci nie będące zdjęciami - co ukróciło próby zastosowania wyuczonego modelu detekcji do automatycznego mapowania masek. Sporządzenie dedykowanego modelu do wykrywania twarzy na ilustracjach jest wartą rozważenia opcją, lecz ryzykowną ze względu na skrajne rozbieżności w stylach rysunku poszczególnych artystów. Ostatecznie zdecydowano, że najbardziej bezpośrednie podejście ręcznego ma-

powania masek (stosowane również przez samych twórców biblioteki) będzie najbardziej stosowne, a sama aplikacja mogłaby być poszerzana o nowe maski czy przystosować UI do mapowania dla użytkownika w razie zapotrzebowania.

Proces ręcznego przygotowywania maski, pokazany na rysunku 3 odbywa się poprzez nałożenie odpowiednich punktów kluczowych biblioteki clmtracks na pożądany obraz a następnie wprowadzenie listy ich współrzędnych w pikselach do programu.

Rysunek 3: Proces ręcznego mapowania maski

Tekstura wraz z odpowiednią maską są przetwarzane przez załączoną w bibliotece projekcie klasę `faceDeformer` aby odpowiednio dopasować maskę do obecnej pozycji wykrytej twarzy. Końcowe rezultaty pokazuje rysunek 4.

Rysunek 4: Maska dopasowana do twarzy w czasie rzeczywistym

W ramach implementacji clmtrackr spożytkowano również możliwość defor-

macji wykrytej twarzy, którą umieszczono w interfejsie użytkownika jako alternatywną do nakładania maski opcję. Na rysunku 5 widać przykład użycia funkcji wraz z podmianą tła (o czym więcej w sekcji 3.2.3).

Rysunek 5: Przykład zastosowania deformacji twarzy

3.2.3 Przetwarzanie BodyPix

Drugim z modułów jest moduł odpowiadający za separację i podmianę tła, wykorzystujący bibliotekę BodyPix.

Podobnie jak opisany powyżej moduł clmtrackr, bodypix operuje na obrazie z kamery, który zostaje przetwożony a następnie przekierowany na wspólny wyjściowy element `canvas`.

Model BodyPix inicjalizowany jest z użyciem są niezbędnych informacji początkowych, takich jak źródłowe wideo czy parametry sieci neuronowej TensorFlow. Utworzono kilka presetów dostępnych do wyboru z panelu użytkownika, tak aby umożliwić wybór pomiędzy jakością a prędkością wedle potrzeb.

```
function getNeuralNetworkComplexity() {
    switch(props.architectureComplexity) {
        case "MobileNet basic": {
            return ({
                architecture: 'MobileNetV1',
                outputStride: 16,
                multiplier: 0.5,
                quantBytes: 1
            });
        }
        case "MobileNet standard": {
            return ({
                architecture: 'MobileNetV1',
                outputStride: 8,
```

```
                multiplier: 0.75,
                quantBytes: 2
            });
        }
        [...]
    }
}
```

Następnie w czasie rzeczywistym uruchamiana jest metoda wykorzystująca detekcję zawartą w bibliotece BodyPix aby odseparować postaci ludzkie od tła. Jako renderowane tło wykorzystać można zarówno grafikę statyczną jak i animowaną (tym samym osiągnięto rozszerzoną funkcjonalność projektu) - po czym wykryte sylwetki ludzkie nanoszone są na jej wierzch. Przykładowe rezultaty substytucji tła demonstruje Rysunek 6.

```
async function runBodySegments() {
    const net = await bodyPix.load(getNeuralNetworkComplexity());
    detection = setInterval(() => {
        if (!isCancelled) {
            detect(net);
            hueOffset = (hueOffset + 7) % 360;
        } else {
            clearInterval(detection);
        }
    }, consts.bpDelay);
}

runBodySegments();
```

Rysunek 6: Przykład podmiany tła

W ramach implementacji BodyPix udostępniono również szereg filtrów modyfikujących wyodrębnione sylwetki ludzkie pod względem koloru, czy rozmywających je. Przykład wykorzystania widoczny jest na rysunku 7.

Rysunek 7: Filtr kolorystyczny BodyPix

4 Instrukcja użytkownika

Sporządzona aplikacja jest aplikacją web'ową dostępną do lokalnego uruchomienia lub hostowania w postaci witryny. W ramach technicznej prezentacji dostępny jest do pobrania kod źródłowy umożliwiający utworzenie lokalnej instancji programu.

4.1 Instalacja i uruchomienie

Aby lokalnie uruchomić aplikację niezbędne jest zainstalowanie środowiska `Node.js` a następnie wydanie w folderze głównym plików aplikacji poleceń:

```
npm install
npm start
```

Od tego momentu aplikacja dostępna będzie z pomocą dowolnej przeglądarki pod adresem `localhost:3000`. Dla maksymalnej jakości doświadczeń, zaleca się korzystanie z Google Chrome, oraz uruchomienie aplikacji korzystając z wbudowanego w przeglądarkę trybu `app mode` aby zminimalizować interfejs użytkownika przeglądarki. W tym celu należy uruchomić aplikację a następnie z

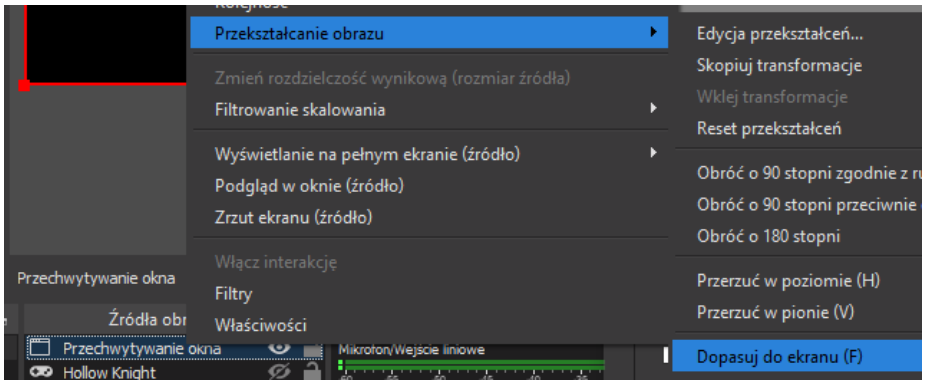
zakładki opcji Chrome (trzy kropki) wybrać ***Więcej narzędzi*** ⟩ ***Utwórz skrót*** i zaznaczyć opcję uruchamiania w oknie. Tak skonfigurowana aplikacja dostępna będzie każdorazowo z panelu aplikacji chrome, jak pokazano na rysunku 8.

Rysunek 8: Aplikacja łatwo dostępna z poziomu panelu Chrome

4.2 Aplikacje towarzyszące

Dla maksymalnej kompatybilności z aplikacjami zewnętrznymi zaleca się następnie przechwycić wygenerowany obraz za pomocą aplikacji OBS wykorzystując zintegrowaną funkcję kamery wirtualnej. Zainstalować i skonfigurować najnowszą wersję aplikacji zgodnie z instrukcjami wydawcy, a następnie w przechwycić w OBS okno aplikacji RPGraphics. Po aktywacji funkcji kamery wirtualnej, uzyskanego obrazu używać z dowolną aplikacją komunikacyjną.

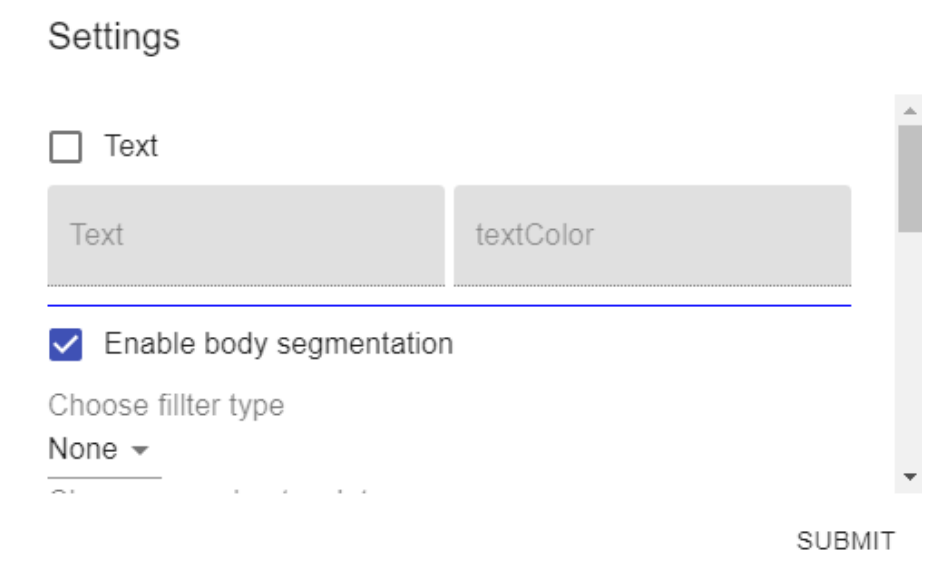
Ze względu na ograniczenia dotyczące rozmiaru renderowania RPGraphics (związane z wysoką złożonością operacji sieci neuronowych) zaleca się pozostawić okno aplikacji rozmiarów `800x450` i wykorzystać ustawienia OBS aby dopasować je do renderowanego obrazu kamery, jak pokazano na Rysunku 9.



Rysunek 9: Ustawienie skalowania obrazu aplikacji do kamery

4.3 Funkcje i użytkowanie

Po krótkiej inicjalizacji użytkownikowi ukaże się obraz z jego kamery internetowej. Jednokrotne kliknięcie w dowolne miejsce obrazu ukaże przycisk menu kontekstowego w prawym górnym rogu. Otwierając niniejszego menu użytkownik może dowolnie aktywować i konfigurować funkcje aplikacji. Wycinek menu umieszczono na rysunku 10, zaś poniże opisano dostępne funkcje.



Rysunek 10: Fragment interfejsu użytkownika

- **Tekst tytułowy:** Aplikacja umożliwia ustawienie tekstu tytułowego wi-

docznego w górnej części ekranu. Docelowo tekst ten służyć ma za informację określającą imię odgrywanej przez gracza postaci lub postaci opisywanej przez prowadzącego rozgrywkę. Za pomocą interfejsu można łatwo zdefiniować treść tekstu jak i jego kolor.

- **Segmentacja ciała:** Opcja główna aktywująca moduł wyodrębniający sylwetki ludzkie, odblokowująca możliwość użycia następujących sub-modułów.
  - **Filtry ciała**
  - **Tło statyczne**
  - **Tło wideo**

  Ponadto w niniejszej sekcji wybrać można predefiniowane ustawienia sieci neuronowych aby dobrać te o odpowiednich do potrzeb parametrach.

- **Deformacja Twarzy:** Aplikacja umożliwia zdeformowania zarejestrowanej twarzy użytkownika w szereg predefiniowanych sposobów, spośród których można dokonać wyboru za pomocą interfejsu użytkownika.

- **Maska twarzy:** Aplikacja umożliwia opcjonalne włączenie i wybranie maski nakładanej na twarz użytkownika, reprezentującej odgrywaną postać. Aplikacja dostarczana jest z zestawem masek, spośród których dokonać można wyboru za pomocą interfejsu.

5 Rezultaty i podsumowanie

Końcowa aplikacja łączy efekty działania obu omówionych komponentów na wspólnym elemencie `canvas`, wraz z interfejsem użytkownika służącym do zarządzania jej funkcjonalnościami takimi jak wybór masek twarzy.

Obraz poszerzony jest o wybrane tło, a twarz użytkownika jest z sukcesem wykrywana i podmieniana. Wszystkie podstawowe założenia projektowe zostały zrealizowane.

Założenia dotyczące funkcjonalności rozszerzonych zostały zrealizowane częściowo. Dodano możliwość podmiany tła na wideo oraz przedyskutowano kwestię umożliwienia użytkownikowi dostarczenia własnej maski twarzy do podmiany. Własne maski użytkowników okazały się jednak zadaniem dużo przekraczającym wymaganym nakładem pracy czas dostępny na realizację projektu. Podobnie podmiana elementów ciała na akcesoria ubioru/pancerza.

Kwestii rejestracji generowanego wyjścia jako wirtualnego urządzenia typu "kamerki internetowej", jako integralnej funkcjonalności samej aplikacji, nie udało się poruszyć w czasie pracy. Zadanie to jest jednak realizowane zewnętrznie.

Powodów dla, których funkcjonalności rozszerzone nie zostały w pełni zrealizowane szukać należy wśród kwestii obeznania z zastosowanymi technologiami. W pierwszej iteracji pracy nad projektem natknięto się na problemy techniczne

związane ze specyfiką działania zastosowanych bibliotek oraz braku możliwości ich współpracy z zastosowanymi komponentami wideo co przyczyniło się do konieczności refaktoryzacji i opóźniło dalsze prace. Dodatkowo część uczestników projektu na bieżąco zaznajamiała się ze stosowanym językiem i technologiami, co też rzutowało negatywnie na postęp prowadzenia prac.

Ostatecznie jednak, aplikacja została ukończona z całością funkcjonalności podstawowych i częścią rozszerzonych w związku z czym przedsięwzięcie uznaje się za sukces.