Programowanie Równoległe i Rozproszone

Projekt: Morphing

Patryk Bołoz, Marcin Cisek

Informatyka II st. Niestacjonarne, 1 rok

1. **Wstęp**

Celem projektu było stworzenie aplikacji mobilnej na urządzenia z systemem Android, która umożliwia użytkownikowi stworzenie animacji płynnej transformacji dwóch wejściowych zdjęć, wykorzystując technikę morphingu.

1. **Środowisko implementacyjne**

W projekcie wykorzystano język programowania Kotlin. Działa on na maszynie wirtualnej Javy i jest zaprojektowany z myślą o interoperacyjności z tym językiem. Środowiskiem uruchomieniowym aplikacji był Android Studio. Został on wybrany ze względu na wsparcie języka dla tej platformy przez producenta - wsparcie dla Kotlina jest standardową funkcjonalnoścą IDE.

1. **Morphing**

**3.1. Wstęp**

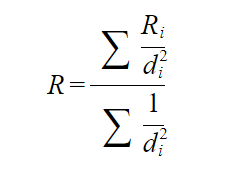
Morphing jest techniką płynnej transformacji jednego obrazu w drugi. Jest on szeroko wykorzystywany już od lat dziewiędziesiątych ubiegłego wieku jako efekt w animacjach oraz filmach - najczęściej używa się go do przedstawiania twarzy jednej osoby zmieniającej się w inną w sposób płynny.

Aby wykorzystać technikę morphingu, należy zaznaczyć punkty charakterystyczne (takie jak oczy, nos, usta), które będą sobie odpowiadały na obu obrazach wejściowych. Istnieją również bardziej skomplikowane algorytmy (w pełni automatyczne), które pozwalają na automatyczne wykrywanie twarzy.

**3.2. Algorytm**

Zadaniem algorytmu jest wygenerowaniu kilku obrazów wynikowych, które są transformacją pierwszego obrazu w drugi. Ilość obrazów pośrednich zostaje zaimplementowana na początku programu. Każdy wygenerowany obraz posiada własny współczynnik lambda z zakresu od 0 do 1. Jest on uzależniony od ilości końcowych obrazów (czyli od płynności przejścia animacji) i opisuje stopień zawartości obrazów wejściowych w każdym obrazie wyjściowym. Proces nie jest w pełni automatyczny, ponieważ aplikacja wymaga od użytkownika wybrania trzech punktów charakterystycznych dla każego obrazu wejściowego (w przypadku naszej aplikacji punktami są oczy oraz usta).

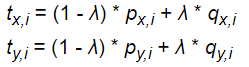
Kolor obrazu wynikowego to średnia kolorów RGB miejsc charakterystycznych obu obrazów wejściowych. Obliczenie poszczególnych kolorów opisuje poniższy wzór:



gdzie:

Ri - kolor współrzędnych i-tego punktu kontrolnego  
 di - odległośc i-tego punktu kontrolnego

Jeśli wzpołrzędne punktów charakterystycznych oznaczymy jako px oraz py, to punkty na obrazie wynikowym obliczymy jako:



Rys.2 Wzór na współrzędne punktów obrazu wynikowego.

1. **Działanie programu**

Głowne okno programu zawiera trzy przyciski, z których każdy odpowiada danej funkcjonalności aplikacji. Aby wybrać obrazy wejściowe, należy nacisnąć button “Library”. Użytkownik zostanie przeniesiony do galerii zdjęć, posortowanych w kolejności ostatniej modyfikacji/dodania zdjęcia do pamięci urządzenia. Następnie użytkownik zaznacza dwa zdjęcia i naciska przycisk “Done”, który przekierowuje go do menu głównego.

Kolejnym etapem jest wciśnięcie buttona “Choose points”. Jeśli do tej pory użytkownik nie wybrał żadnego zdjęcia, następuje walidacja i na ekranie zostaje wyświetlony komunikat “Please select photos”. W przeciwnym wypadku wyświetlony zostanie widok z pierwszym wybranym zdjęciem. Rolą użytkownika jest wskazanie na ekranie trzech charakterystycznych punktów (oczy i usta). Po wykonaniu tej czynności na ekranie pojawi się drugie wybrane zdjęcie (schemat wskazywania punktów jest ten sam). Po poprawnym wybraniu punktów uytkownik zostaje przekierowany do widoku menu głównego.

Ostatnim etapem jest wciśnięcie przycisku “Start”, który uruchamia funkcję *getNewImage(),* odpowiedzialną za stworzenie zdjęcia wyjściowego.

1. **Wnioski**

Stworzona aplikacja moze być przykładem, jak duże korzyści wynikają ze zrównoleglenia obliczeń (zwłaszcza w grafice). Aplikację przetestowano na dwóch urządzeniach. Pierwszy posiadajał procesor Qualcom Snapdragon 410 (z czterema rdzeniami) - wynik obliczeń to ponad 30 sekund. Dla drugiego urządzenia z procesorem Qualcom Snapdragon 435 (z ośmioma rdzeniami) czas wyniósł około 20 sekund.