**Rozpoznawanie człowieka metodami biometrii**

**Sprawozdanie 3**

Michalik Piotr

Spis treści

[Wstęp 3](#_Toc514958954)

[Wykrywanie Twarzy 3](#_Toc514958955)

[Cechy Twarzy 5](#_Toc514958956)

[Weryfikacja 7](#_Toc514958957)

[Wyniki 7](#_Toc514958958)

[Wnioski 9](#_Toc514958959)

[Źródła 9](#_Toc514958960)

# Wstęp

W poniższym dokumencie przedstawię środki użyte do realizacji drugiego projektu na przedmiot ‘Wykrywanie człowieka metodami biometrii’. Po kolei zostaną opisane i przedstawione metody zaimplementowane w tym projekcie. Później nastąpi części właściwa, to znaczy ocena i opis zaimplementowanych przeze mnie algorytmów. Na samym końcu zamieszczone zostanie krótkie podsumowanie etapu. Dla każdej opisanej metody zostaną zamieszczone screenshoty prezentujące działanie poszczególnej funkcjonalności.

W tym sprawozdaniu opisane, zaprezentowane i ocenione zostaną dwa własne algorytmy: algorytm wykrywania twarzy oraz algorytm służący do ekstrakcji cech oraz ich weryfikacji. Celem projektu jest weryfikacja osób na podstawie nagrań zdjęć ich twarzy.

# Wykrywanie Twarzy

## Teoria

## Głównym zadaniem jakie trzeba wykonać w procesie rozpoznawania twarzy jest samo znalezienie twarzy na zdjęciu. Nie jest to jednak proste zadanie ze względu na kilka problemów. Po pierwsze twarze nie zawsze są tych samych rozmiarów, nie zawsze są w centrum zdjęcia oraz nie zawsze mają ten sam odcień skóry. Kolejnym poważnym problemem jest tło. Może być jaśniejsze lub ciemniejsze od twarzy. Może być w tym samym kolorze albo może zawierać wiele obiektów przypominających ludzką twarz. Dlatego w swoim algorytmie zaproponowałem połączenie działań powszechnie znanych, jak i tych które eksperymentalnie uznałem za przydatne.

## Algorytm wykrywania twarzy:

1. Dla każdego pixela, wszystkie składowe koloru przyjmują wartość równą:
2. Dla całego obrazu (z wyłączeniem marginesu obejmującego 15% wymiarów zdjęcia) ustalamy treshold który jest średnim kolorem obrazu. Następnie każdy pixel którego wartość jest mniejsza niż 120% wartości treshold zamieniamy na pixel czarny.
3. Na zdjęcie nakładamy filtr Gaussa (używamy maski 5x5)
4. W celu zwiększenia kontrastu między potencjalną twarzą a tłem, rozciągamy kolory. Wszystkie pixele których wartość jest mniejsza od 120% wartości treshold zmieniamy na 10% ich dotychczasowej wartości. Pixele których wartość jest większa od 120% treshold zmieniamy wartość na 150% ich dotychczasowej wartości.
5. Binaryzujemy zdjęcie używając 160% wartości treshold jako wyznacznika granicy miedzy kolorami.
6. Obliczamy pionowy i poziomy histogram uzyskanego zdjęcia. Następnie wygładzamy histogramy metodą Moving Average. Z uzyskanych w ten sposób histogramów wyznaczamy najjaśniejszy punkt na zdjęciu (potencjalny środek twarzy).
7. Uzywając algorytmu Flood Fill, wypełniamy kolorem czerwonych uzyskane zdjęcie z miejsca oznaczonego jako potencjalny środek twarzy.
8. Ponownie obliczamy pionowy i poziomy histogram uzyskanego zdjęcia. Jednak tym razem nie badamy czarnej barwy, a barwę czerwoną. Następnie wygładzamy histogramy metodą Moving Average. Z uzyskanych w ten sposób histogramów wyznaczamy najciemniejszy punkt na zdjęciu (poprawiony środek twarzy) oraz skrajne wartości które kolejno wyznaczają granice poziome i pionowe znalezionej twarzy.

Na podstawie uzyskanych danych wycinamy prostokątny fragment z oryginalnego zdjęcia który naszym zdaniem zawiera twarz. Następnym krokiem będzie ekstrakcja cech.

## Implementacja

Ponieważ zaproponowany algorytm jest kompozycją własną, został on zaimplementowany dosłownie zgodnie z powyższym opisem.

## Screenshot



Rysunek - Proces wykrywania twarzy

# Cechy Twarzy

## Teoria

Mając już znalezioną twarz możemy przystąpić do ekstrakcji cech. Postanowiłem na początku kodować twarz za pomocą 4 cech charakterystycznych:

* Stosunek szerokości twarzy do długości twarzy
* Stosunek rozstawu między oczami i ustami do długości twarzy
* Stosunek odległości między ustami a dolną częścią twarzy do długości twarzy
* Stosunek odległości między oczami a górną częścią twarzy do długości twarzy

Dzięki temu że każda cecha jest odniesiona do długości twarzy, która z kolei jest niezależna od rozmiaru zdjęcia, nasze cechy są uniwersalne. Każda z cech jest liczbą rzeczywistą z przedziału [0;1].

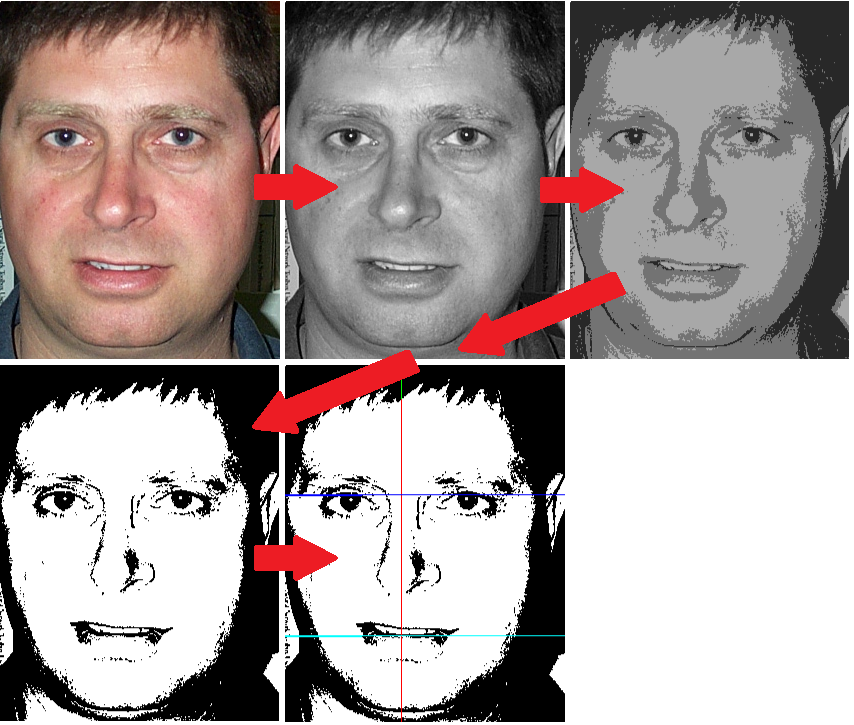
Algorytm ekstrakcji cech:

1. Zamieniamy zdjęcie do skali szarości.
2. Za pomocą algorytmu KNN obliczamy 3 najliczniejsze kolory i najliczniejszy z nich traktujemy jako kolor skóry.
3. Binaryzujemy zdjęcie gdzie granicą między kolorami jest 70% wartości koloru skóry.
4. Wyznaczamy pionowy i poziomy histogram który wygładzamy metodą Moving Averages.
5. Wyznaczamy oś twarzy szukając minimum w poziomym histogramie.
6. Wyznaczamy wysokość twarzy szukając na poziomym histogramie pierwszej i ostatniej gwałtownej zmiany wartości koloru (na odcinku 5% histogramu zmiana wartości o 35% różnicy między maksymalną a minimalna wartością).
7. Wyznaczamy szerokość twarzy szukając na pionowym histogramie pierwszej i ostatniej gwałtownej zmiany wartości koloru (na odcinku 5% histogramu zmiana wartości o 35% różnicy między maksymalną a minimalna wartością).
8. Wyznaczamy pozycję oczu. Na pionowym histogramie, na odcinku należącym już do wysokości twarzy szukamy pierwszego lokalnego maximum poszukując od góry.
9. Wyznaczamy pozycję ust. Na pionowym histogramie, na odcinku należącym już do wysokości twarzy szukamy pierwszego lokalnego maximum poszukując od dołu.

## Implementacja

Ponieważ zaproponowany algorytm jest kompozycją własną, został on zaimplementowany dosłownie zgodnie z powyższym opisem.

## Screenshot

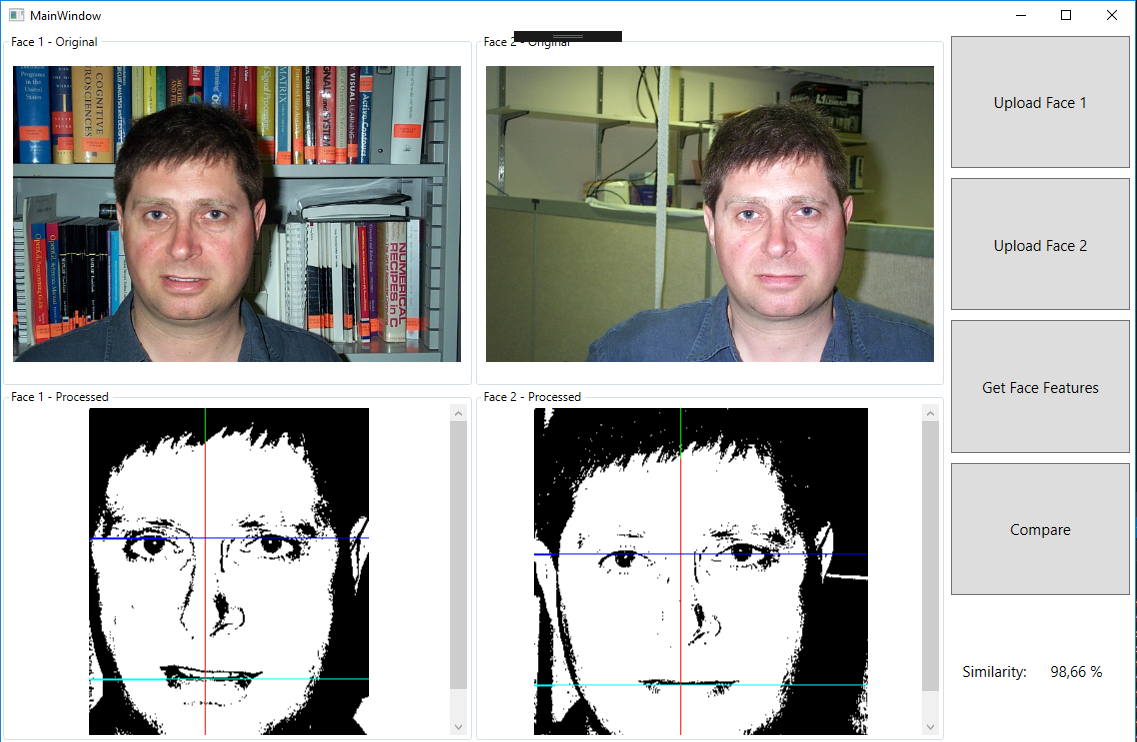


Rysunek - Ekstrakcja cech

# Weryfikacja

Posiadając dwa zdjęcia, potrafimy dla nich wyznaczyć dwa wektory cech która każda ma wartości od 0 do 1. W ten sposób weryfikacja jest bardzo prosta, gdyż dla każdej pary cech obliczamy różnice pomiędzy nimi i sumujemy wszystkie cechy. Najbardziej różne zdjęcia będą miały sumę różnic równą długości wektora cech, a najbardziej podobne zdjęcia będą miały sumę różnic równą 0. Dzięki temu łatwo obliczamy procentowe podobieństwo pomiędzy dwoma zdjęciami.

## ScreenShoty



Rysunek - Interface aplikacji oraz proces weryfikacji

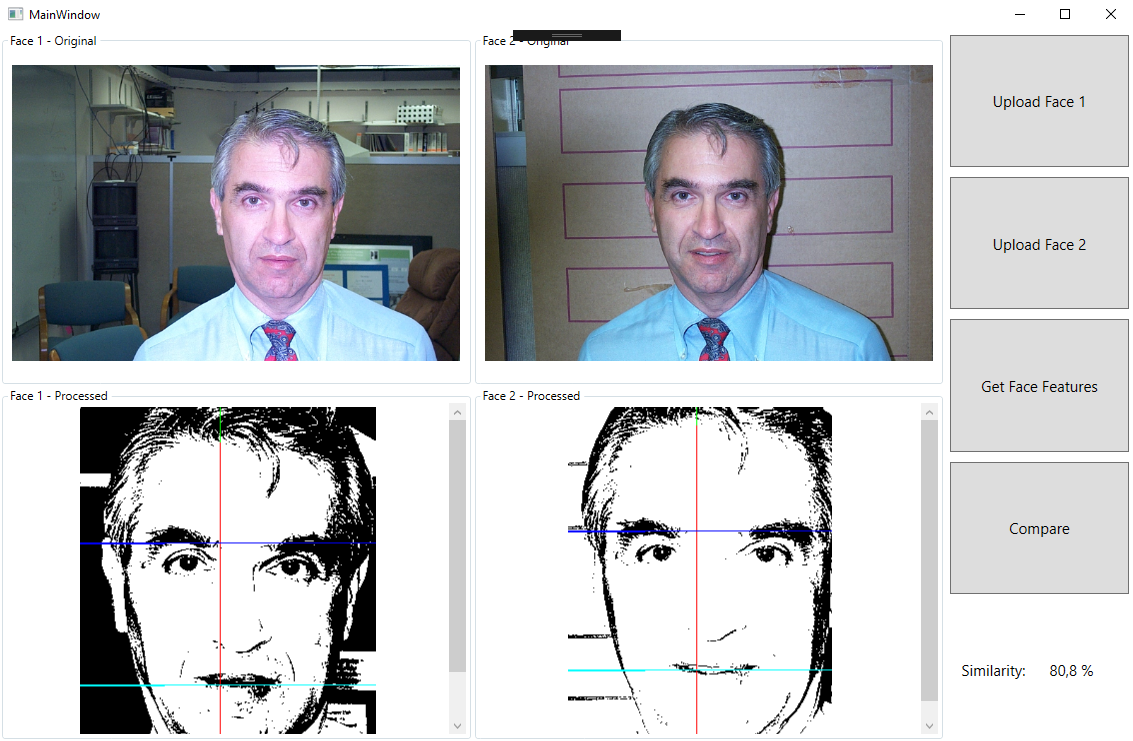
# Wyniki

Niestety, zaproponowane przeze mnie rozwiązanie nie sprawdza się w zastosowaniu ogólnym. Pomimo usilnych starań aplikacja jest niesamowicie wrażliwa na tło. Obiekty w tle zakłócają nie tylko samą analizę za pomocą histogramów, jak również wykrywanie twarzy w ogóle. W sytuacjach gdy tło jest jaśniejsze od twarzy, gdy tło zawiera np. białą kartkę papieru lub gdy twarz w połowie jest zakryta cieniem, algorytm prawie na pewno się pomyli.

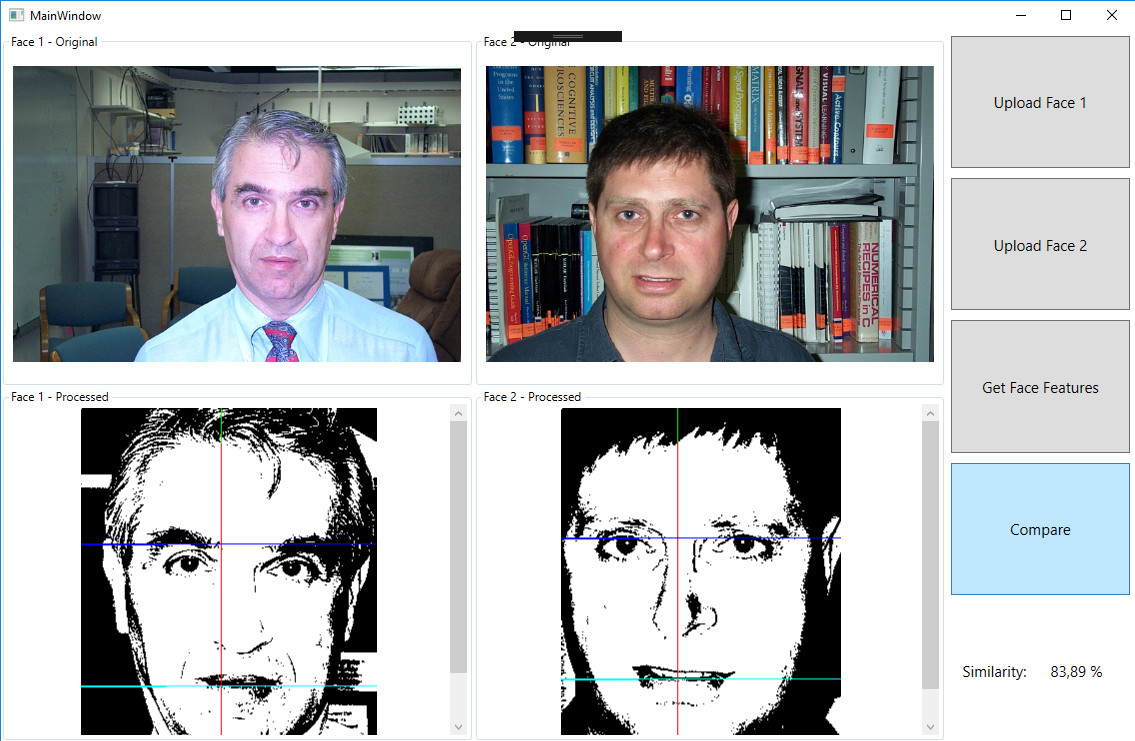
W sytuacji gdy twarz zostanie poprawnie wykryta, nadal problem jej analiza histogramu. U osób o krzaczastych brwiach i małych oczach, to brwi zostaną potraktowane jako oczy. Gdy osoba posiada ledwo widoczne usta, za to posiada ciemną bródkę, histogram usta zlokalizuje w miejscu brody.

To wszystko wpływa na to że weryfikacja zaproponowanym przeze mnie rozwiązaniem jest nieskuteczna.

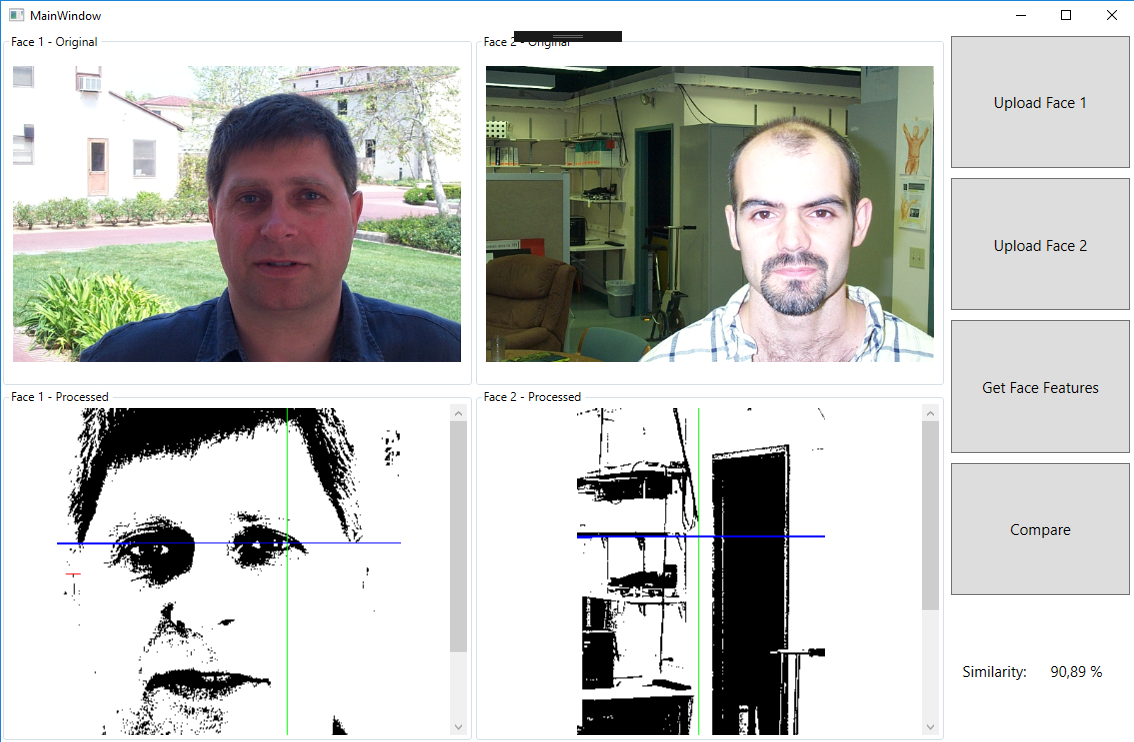
## ScreenShoty



Rysunek - Poprawna ekstrakcja, poprawna weryfikacja



Rysunek - Poprawna ekstrakcja, niepoprawna weryfikacja



Rysunek - nie poprawna ekstrakcja, nie poprawna weryfikacja

# Wnioski

Metoda rozpoznawania twarzy w oparciu o analizę histogramów jest skuteczna, ale tylko pod warunkiem czystości tła. Zdjęcie musi być dobrej jakość i bez zakłóceń.

Być może lepszy pre-processing zdjęcia poprawiłby wyniki pracy.

Najlepszym rozwiązaniem była by zmiana koncepcji na wykrywanie twarzy poszukując przygotowanych przykładów geometrii twarzy ( poszukiwanie owalnych kształtów ) oraz poszukując koloru skóry (podobnie jak przy ekranach green-screen), gdyż ta metoda jest mniej podatna na zakłócenia.

# Źródła

* „Wybrane zagadnienia biometrii” – K. Ślot
* Praca licencjacka – A. Wrońska, P. Luśtyk
* Praca licencjacka – A. Bednarz, T.Felski