

Karol Czub  
Adam Dębaczak

Informatyka Semestr VI  
Podstawy Teleinformatyki

Rozpoznawanie twarzy i śledzenie ruchu

# Spis treści

<b>1. Opis ogólny aplikacji.</b>	<b>3</b>
1.1. Menu główne aplikacji:	4
1.2. Identyfikacja osób:	5
1.3. Dodawanie nowej osoby:	6
<b>2. Uzasadnienie wyboru tematu.</b>	<b>7</b>
<b>3. Podział prac między członków zespołu.</b>	<b>8</b>
<b>4. Funkcjonalności oferowane przez aplikację.</b>	<b>8</b>
4.1. Identyfikacja osób.	9
4.2. Dodawanie nowej osoby.	10
4.3. Wyświetlanie listy rozpoznawanych osób.	10
4.4. Usuwanie rozpoznawanej osoby z bazy.	11
4.5. Ustanowienie połączenia z kamerą przy użyciu IP.	11
4.6. Wyświetlanie instrukcji dla użytkownika.	11
4.7. Wyświetlanie komunikatów o błędach.	12
<b>5. Użyte technologie.</b>	<b>12</b>
5.1. Python	12
5.2. OpenCV	13
5.3. Numpy	13
5.4. Tensorflow	13
5.5. Tkinter	13
5.6. JSON	14
<b>6. Architektura rozwiązania.</b>	<b>14</b>
<b>7. Problemy podczas implementacji aplikacji oraz testy.</b>	<b>14</b>
7.1 Brak GUI w OpenCV	15
7.2. Wykrywanie twarzy za pomocą kaskad Haara i Local Binary Patterns.	15
7.3. Identyfikacja osób za pomocą gotowych algorytmów biblioteki OpenCV.	16
7.4 Testy.	18
7.4.1. Nakrycie głowy.	18
7.4.2. Okulary przeciwsłoneczne.	19
7.4.3. Zasłonięte usta przez wysoki szal.	22
7.4.4. Uszy zasłonięte przez zimową czapkę lub długie włosy.	23
<b>8. Instrukcja użytkowania.</b>	<b>24</b>
8.1. Pola i przyciski interfejsu:	25
8.2. Tryb identyfikacji osób.	28
8.3. Tryb dodawania nowej osoby.	29
<b>9. Bibliografia.</b>	<b>31</b>

## 1. Opis ogólny aplikacji.

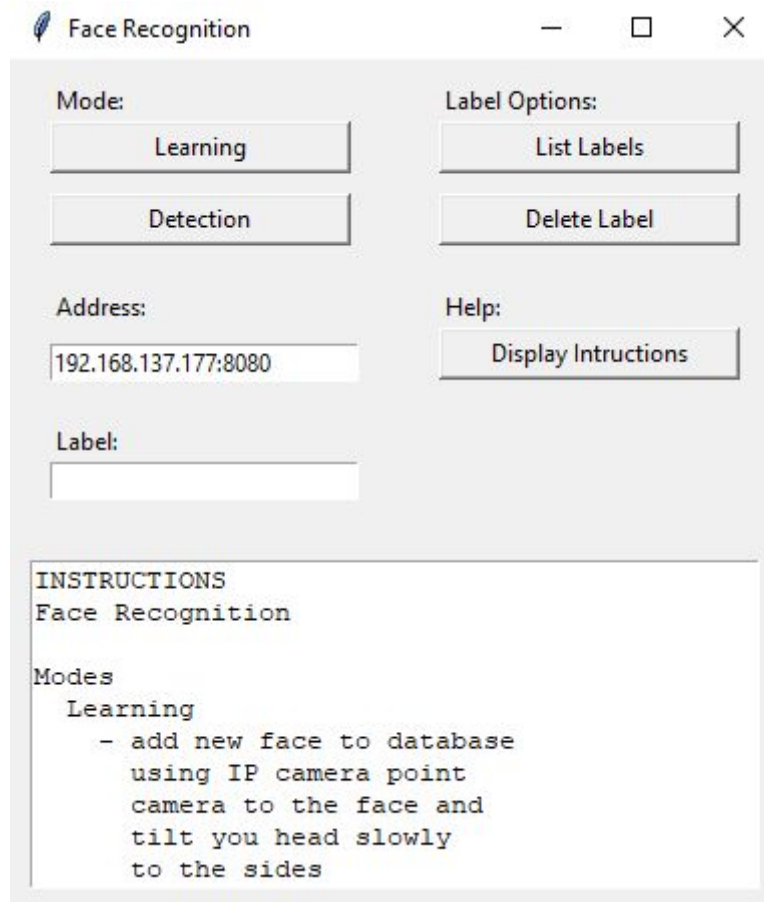
Aplikacja, napisana na systemy operacyjne Windows, do której prawidłowego działania potrzebna jest możliwość podłączenia do komputera kamery IP. Aplikacja ta służy do identyfikacji osób, które znajdują się w polu widzenia kamery. Zarówno interfejs jak i komunikaty wyświetlane użytkownikowi są w języku angielskim, ze względu na popularność języka. Sama aplikacja udostępnia trzy główne funkcjonalności:

- **Dodawania nowej osoby**, która służy do uczenia się twarzy pojedynczej osoby, znajdującej się przed kamerą. Wprowadza się imię i nazwisko takiej osoby, a następnie po pojawieniu się podglądu wykrywanej twarzy, osoba ta powinna obracać się pod różnym kątem przed kamerą, zmieniając mimikę twarzy, tak by zapewnić jak najszerszy zbiór obrazów w różnych orientacjach, co skutkuje większym prawdopodobieństwem wykrycia i prawidłowego rozpoznania tej osoby w przyszłości, niezależnie od jej wyrazu twarzy. W dowolnym momencie użytkownik może zakończyć naukę. Wtedy twarz, wraz z przypisanym do niej imieniem i nazwiskiem, zostanie dopisana do bazy wyuczonych wzorców, którą aplikacja tworzy w folderze swojej lokalizacji, w postaci pliku tekstowego facerec\_128D.
- **Identyfikacji osób**, w której otwierany jest podgląd widoku z kamery. Twarze wszystkich widocznych na nim osób oznaczone są kwadratem, a dodatkowo te, które zostaną rozpoznane jako wcześniej nauczone, zostaną opisane nazwą, podaną przy nauce danej twarzy jako imię i nazwisko.
- **Usuwanie danych konkretnej osoby** z bazy danych tak, by nie było możliwe dalsze rozpoznawanie tej osoby w trybie identyfikacji osób opisanym powyżej.

W celu poprawnego działania, kamera powinna znajdować się w tej samej sieci co komputer obsługujący aplikację. W kolejnym kroku, po podaniu w polu tekstowym prawidłowego adresu IP oraz portu następuje połączenie.

## Ekran aplikacji:

### 1.1. Menu główne aplikacji:



The screenshot shows the main menu of a 'Face Recognition' application. The window has a title bar with the text 'Face Recognition' and standard window controls (minimize, maximize, close). The interface is divided into several sections:

- Mode:** Two buttons, 'Learning' and 'Detection', are stacked vertically.
- Label Options:** Two buttons, 'List Labels' and 'Delete Label', are stacked vertically.
- Address:** A text input field containing the IP address '192.168.137.177:8080'.
- Help:** A button labeled 'Display Intructions' (note the spelling error).
- Label:** An empty text input field.
- INSTRUCTIONS:** A text area at the bottom containing the following text:

```
INSTRUCTIONS
Face Recognition

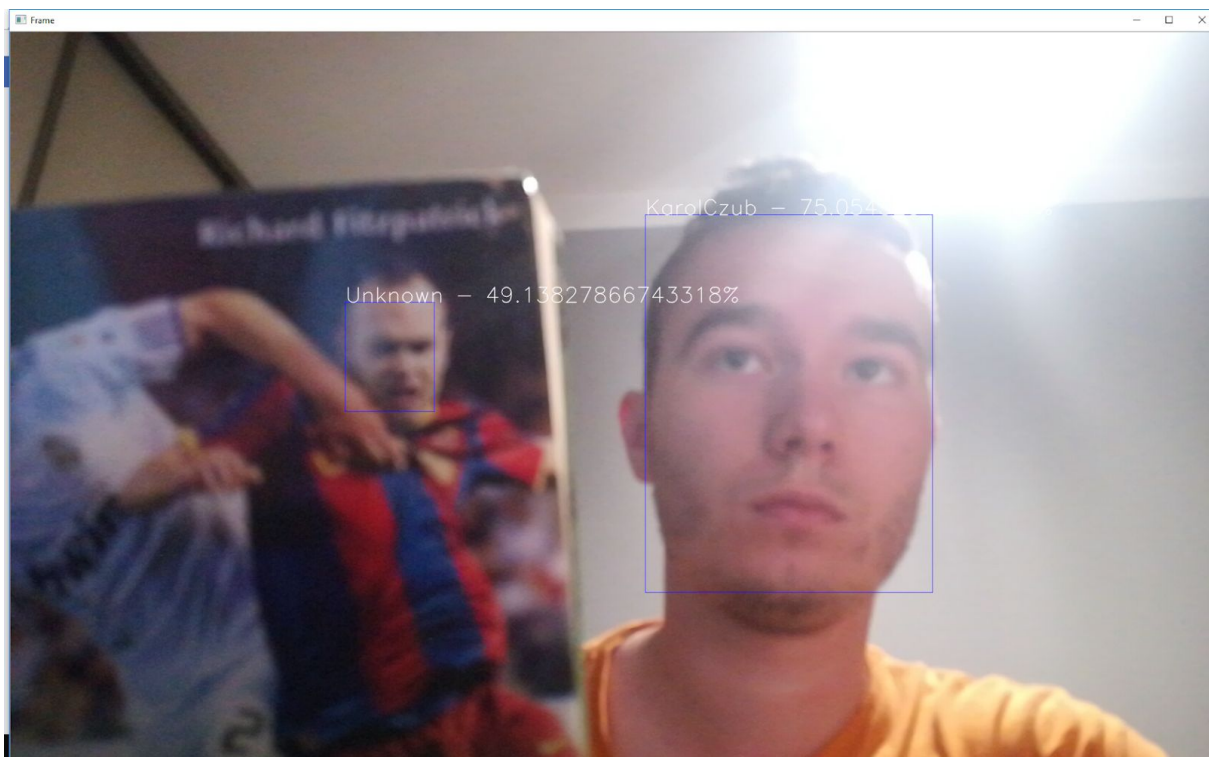
Modes
  Learning
    - add new face to database
      using IP camera point
      camera to the face and
      tilt you head slowly
      to the sides
```

Jest to Menu główne aplikacji, które umożliwia:

- Wybranie odpowiedniego trybu pracy: Learning, czyli tryb dodawania nowej osoby lub Detection, czyli tryb identyfikacji osób
- Wpisanie adresu IP kamery, której aplikacja będzie używać (pole Address).
- Sprawdzenie listy osób, które aplikacja rozpoznaje (przycisk List Labels)
- Usunięcie danych konkretnej osoby z bazy osób, które aplikacja rozpoznaje (przycisk Delete Label)
- Wybranie nazwy nowej osoby, która zostanie dodana lub wskazanie nazwy osoby, której dane mają zostać usunięte (pole Label)

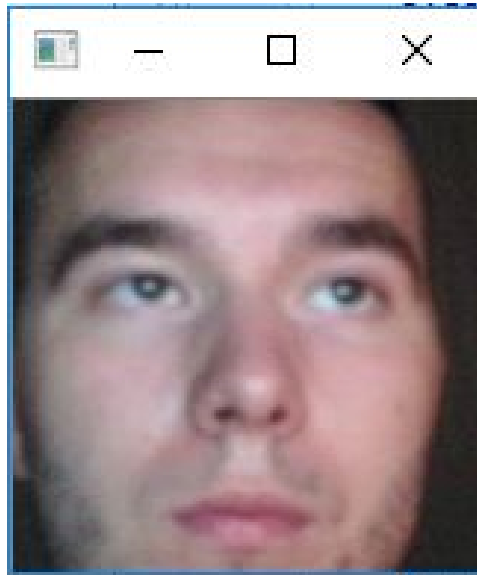
- Wyświetlenie instrukcji odnośnie użytkowania aplikacji (przycisk Display Instructions), która jest automatycznie wyświetlana na samym początku lecz może przestać się wyświetlać na przykład w przypadku wystąpienia komunikatu o błędzie
- Komunikację z użytkownikiem za pomocą pola tekstowego na dole ekranu aplikacji. Służy do wyświetlania instrukcji, komunikatów o błędach oraz listy osób zapisanych w bazie

## 1.2. Identyfikacja osób:



Ekran podglądu z kamery IP, wraz z oznaczonymi niebieskimi prostokątami twarzami osób, które aplikacja wykrywa. Jeżeli dana osoba została rozpoznana, to nad górną krawędzią kwadratu widnieć będzie jej nazwa, która została podana przy dodawaniu jej do bazy. W przeciwnym wypadku widnieć będzie napis "Unknown". Oprócz tego obok etykiety wyświetla się procentowa wartość dopasowania, do najbardziej zbliżonego ze wzorców. Wartość powyżej 70% oznacza dopasowanie do danego w bazie wzorca.

### 1.3. Dodawanie nowej osoby:



Ekran podglądu wycinku z kamery, zawierającego wykrywaną twarz, dzięki czemu użytkownik ma pewność, że jest prawidłowo wykrywany, a pokazywane do kamery profile twarzy lub zmieniana mimika zdążyła zostać zarejestrowana. Od tego jak dokładne i różnorodne oblicza swojej twarzy użytkownik pozwoli zapamiętać aplikacji, zależy skuteczność identyfikacji osób w przyszłości.

## 2. Uzasadnienie wyboru tematu.

Temat ten został przez nas wybrany z powodu zainteresowania uczeniem maszynowym. Działanie sztucznej inteligencji i jej możliwości w postaci między innymi identyfikowania osób i obiektów są nie tylko ciekawym zagadnieniem, ale też czymś, na czego rozumienie i umiejętność implementacji jest coraz większe zapotrzebowanie. Dodatkowo kierunek ten się cały czas rozwija, ponieważ możliwości sztucznej inteligencji i powszechność jej wykorzystania w wielu dziedzinach rosną z dnia na dzień, więc programistom zajmującym się uczeniem maszynowym raczej nie grożą problemy ze znalezieniem ciekawej pracy. Uznaliśmy, że wiedza którą trzeba przyswoić do implementacji tej aplikacji - technologie, metody i algorytmy używane do rozpoznawania obiektów przez wykorzystanie uczenia maszynowego, jak i sama implementacja będą ważnym elementem naszego rozwoju w tej dziedzinie. Dodatkowo uznaliśmy, że branie udziału w takim projekcie pozwoli nam sprawdzić czy kierunek ten jest rzeczywiście tym, w czym chcielibyśmy pracować w przyszłości. Znaczącym czynnikiem było również nabyte wcześniej doświadczenie oraz podstawowa znajomość technik z dziedziny przetwarzania obrazu.

Projekt okazał się całościowym przeglądem technik używanych w dziedzinach takich jak: Face Detection, Face Recognition, Motion Tracking oraz ich wad, zalet i ogólnych możliwości. Przydatną wartością jest znajomość ograniczeń poszczególnych metod jak na przykład odporność na zmiany oświetlenia czy efektywność wykrywania względem odległości od urządzenia rejestrującego obraz. Mając te kwestie na uwadze można odpowiednio dobrać metodę detekcji dla danego zadania, jest to szczególnie istotne, gdyż żaden algorytm nie cechuje się absolutną elastycznością i odpornością na wszelkie zakłócenia. Uczenie maszynowe jest pomocne ze względu na swoją uniwersalność (pod warunkiem odpowiedniego wyszkolenia), jednak wymaga stworzenia, bądź znalezienia odpowiedniej wielkości zbioru uczącego, a samo szkolenie modelu jest czasochłonne.

### 3. Podział prac między członków zespołu.

Zadanie	Wykonawcy
Organizacja sprzętowa	Adam Dębczak
Konfiguracja bezprzewodowego przechwytywania obrazu z kamery IP	Karol Czub
Implementacja wykrywania twarzy	Adam Dębczak, Karol Czub
Implementacja rozpoznawania twarzy za pomocą metod udostępnianych przez bibliotekę OpenCV	Karol Czub
Implementacja rozpoznawania twarzy przy użyciu TensorFlow	Adam Dębczak, Karol Czub
Implementacja dodawania nowej osoby do zbioru twarzy nauczonych	Adam Dębczak, Karol Czub
Implementacja usuwania istniejącej osoby ze zbioru twarzy nauczonych	Adam Dębczak, Karol Czub
Testy podstawowych rozwiązań z dziedziny Motion Tracking	Adam Dębczak
Interfejs aplikacji	Adam Dębczak

### 4. Funkcjonalności oferowane przez aplikację.

W aplikacji zostało zaimplementowano siedem głównych funkcjonalności, w których skład wchodzi: Identyfikacja osób, Dodawanie nowej osoby, Wyświetlanie listy rozpoznawanych osób, Usuwanie rozpoznawanej osoby z bazy, Ustanowienie połączenia z kamerą przy użyciu IP, Wyświetlanie instrukcji dla użytkownika oraz Wyświetlanie komunikatów o błędach.



#### 4.1. Identyfikacja osób.

Aplikacja otwiera okno, w którym wyświetla podgląd ekranu z kamery. Następnie za pomocą algorytmu MTCNN wykorzystującym sieci neuronowe, znajduje wszystkie twarze jakie występują na tymże podglądzie, zarówno od frontu jak i z profilu, które zaznacza na ekranie podglądu niebieskim prostokątem. Kolejnym krokiem jest porównanie każdej z wykrytych twarzy z twarzami zapisanymi w bazie danych. Aby to zrobić, pobierane są dane wszystkich osób z pliku facerec\_128D.txt, który znajduje się w głównym folderze aplikacji. Dane te przechowywane są w formacie JSON i składają się z etykiet użytkowników, podawanych przy wprowadzaniu kolejnych osób do bazy danych oraz ze stu dwudziestoośmio wymiarowego wektora dla frontu i każdego z profili twarzy, w którym przetrzymywane są odległości pomiędzy cechami twarzy. Wszystkie twarze wykryte podczas identyfikacji osób zostają przeskalowane w jednakowy sposób, co jest znaczące przy wprowadzaniu nowej osoby, dzięki temu niwelowana jest różnica odległości osoby od kamery. Następnie każda z nich również zostaje przedstawiona w postaci opisanego wyżej wektora, a w kolejnym kroku dochodzi do porównania wektorów z tymi, które są zapisane w bazie. Proces weryfikacji działa w następujący sposób - sprawdzane są różnice odległości cech twarzy wykrytej i tych z bazy danych. Wybierana jest wtedy z pliku twarz najlepiej dopasowana i jeżeli zapamiętany model pokrywa się z wykrytym w minimum siedemdziesięciu procentach, to aplikacja uznaje, że jest to ta sama osoba i podpisuje dany kwadrat etykietą tej osoby, nad górną krawędzią obramowania. W przeciwnym wypadku zamiast etykiety wpisane zostaje "Unknown". Do tego oprócz etykiety wyświetlana jest procentowo wartość, z jaką dana twarz pokrywa się z najlepiej dopasowanym modelem z bazy.

## **4.2. Dodawanie nowej osoby.**

Aplikacja łączy się z kamerą i rozpoczyna wykrywanie twarzy. Po znalezieniu jakiegś na obrazie przechwyconym z kamery, uruchamiane jest okno, na którym wyświetla się podgląd nagrywanej twarzy. Użytkownik powinien pokazać się zarówno od frontu jak i z obu profili, gdyż aplikacja zapisuje osobny wektor odległości między cechami dla każdego z profili i frontu twarzy. Jest to niezmiernie ważne, aby uzyskać w przyszłości dobre wyniki przy identyfikacji osób. Równie istotne jest, aby podczas dodawania nowej osoby do bazy, w polu widzenia kamery znajdowała się tylko jedna osoba. W przypadku, gdy więcej niż jedna twarz będzie wykrywana podczas tego trybu, aplikacja może "przeskakiwać" pomiędzy każdą z nich i zapisane dane będą mieszanką cech każdej z twarzy, co w efekcie najprawdopodobniej doprowadzi do nierozpoznawania żadnej z nich. Kiedy użytkownik zakończy naukę nowej osoby, zestaw wektorów opisujących jej twarz zostaje przypisany do etykiety podanej przy rozpoczynaniu dodawania, następnie całość zostanie spakowana do formatu JSON i zapisana do pliku facerec\_128D.txt, znajdującego się w folderze głównym aplikacji.

## **4.3. Wyświetlanie listy rozpoznawanych osób.**

Użytkownik ma możliwość wyświetlenia listy rozpoznawanych przez aplikację osób bezpośrednio z menu głównego. Działa to poprzez wczytanie pliku będącego bazą wszystkich zapamiętanych twarzy o nazwie facerec\_128D.txt, znajdującego się w folderze głównym aplikacji, a następnie wyodrębnienie z niego samych etykiet, które zostały wpisane przez użytkownika przy dodawaniu konkretnych osób. Funkcjonalność ta przydaje się zarówno do upewnienia się, czy dana osoba jest w bazie danych aplikacji jak i do usuwania, gdyż wymaga ono wpisania etykiety osoby do usunięcia z bazy.

#### **4.4. Usuwanie rozpoznawanej osoby z bazy.**

Aplikacja umożliwia użytkownikowi usunięcie danych dotyczących konkretnej osoby z bazy. Opcja ta jest dostępna z menu głównego aplikacji i wystarczy do odpowiedniego pola tekstowego wpisać etykietę osoby, którą chce się usunąć, a następnie kliknąć Delete Label. Po naciśnięciu przycisku, aplikacja pobiera wszystkie obiekty JSON z pliku facerec\_128D.txt, znajdującego się w folderze głównym, a następnie rozpakowuje je, sprawdzając etykietę osoby, której dane są przetrzymywane wewnątrz każdego z obiektów. Po znalezieniu etykiety, odpowiadającej wpisanej przez użytkownika nazwie, obiekt taki jest usuwany, a plik ten jest nadpisywany.

#### **4.5. Ustanowienie połączenia z kamerą przy użyciu IP.**

Po podaniu przez użytkownika adresu IP kamery w formacie x.x.x.x:y, gdzie każdy x odpowiada liczbie z przedziału od 0 do 255, a y jest maksymalnie pięciocyfrowym numerem portu oraz wybraniu trybu pracy (Detection lub Learning), aplikacja łączy się z podanym adresem i zaczyna przechwytywanie obrazu. Funkcjonalność ta jest udostępniona przez biblioteka OpenCV poleceniem VideoCapture. W przypadku gdy obraz do przechwytywania nie został wykryty na podanym adresie, wtedy aplikacja wyświetla użytkownikowi komunikat o błędzie związanym z niepoprawnym adresem IP.

#### **4.6. Wyświetlanie instrukcji dla użytkownika.**

Aplikacja informuje użytkownika o możliwych do wybrania funkcjonalnościach i sposobie ich użycia. Jest to zrealizowany za pomocą pola tekstowego w menu głównym aplikacji. Po uruchomieniu aplikacji wyświetla się w nim komunikat, który opisuje ogólnie działanie każdego z trybów pracy oraz znaczenie pól, które użytkownik musi wypełnić (jak na przykład adres IP kamery). Oprócz tego po wybraniu danego trybu pracy instrukcja zostaje zmieniona na taką, która opisuje dokładnie tryb wybrany przez użytkownika i mówi w jaki sposób z niego korzystać. Po powrocie do menu głównego jest dostępny również przycisk Display Instructions, który umożliwia przywrócenie instrukcji głównej, która wyświetlała się na samym początku po uruchomieniu aplikacji.

#### **4.7. Wyświetlanie komunikatów o błędach.**

W przypadku wystąpienia jakiegoś błędu, aplikacja informuje użytkownika o jego specyfice za pomocą pojawiającego się nowego okna wiadomości. Przykładowo, wpisanie niepoprawnego IP kamery, z którego aplikacja nie jest w stanie pobrać obrazu skutkuje wyświetleniem okna popup z komunikatem "Wrong address, Camera connection error".

### **5. Użyte technologie.**

Językiem programowania w jakim została zaimplementowana aplikacja jest Python. Dodatkowo wykorzystane zostały takie technologie i biblioteki jak :

- OpenCV
- Numpy
- Tensorflow
- Tkinter
- JSON

#### **5.1. Python**

Python to język programowania o przejrzystej składni i braku jawnego typowania, który ułatwia szybkie prototypowanie kodu. Jest przydatny do zastosowań naukowych, ponieważ umożliwia skupienie się na rozwiązywaniu problemów, a nie analizie składni i struktur danych oraz innych aspektach. Największą jego zaletą w tym przypadku jest szeroka gama bibliotek naukowych umożliwiających analizę i przetwarzanie obrazów oraz implementację sieci neuronowych.

## 5.2. OpenCV

OpenCV to prawdopodobnie najpopularniejsza otwarta biblioteka do przetwarzania obrazów. Zawiera metody do przetwarzania obrazu takie jak nakładanie filtrów, operacje morfologiczne, progowanie, wykrywanie konturów i ich analiza oraz wiele innych. Jest kluczowym narzędziem w projekcie uwzględniającym pracę z obrazami. Jest napisana w języku C, ale posiada interfejsy umożliwiające jej użycie w językach takich jak Python, Java, C++. W naszym przypadku głównym wykorzystaniem było przechwytywanie obrazu z kamery, początkowo z urządzeń fizycznie podłączonych do komputera, a później poprzez adres IP oraz podstawowych przekształceń wykonywanych na obrazach (jak np, skalowanie). Biblioteka udostępnia dużo różnorodnych funkcji bezpośrednio związanych z tematyką projektu (Face Detection, Motion Tracking, Face Recognition), jednak po wstępnym przetestowaniu ich skuteczność okazywała się niezadowalająca w porównaniu do zastosowanych później algorytmów uczenia maszynowego.

## 5.3. Numpy

NumPy jest biblioteką będącą wsparciem dla wszelkiego rodzaju obliczeń. Z naszej perspektywy najbardziej użyteczną spośród dostępnych funkcji były rozszerzone względem podstawowej implementacji (w Pythonie) typy dostępnych tablic i macierzy N-wymiarowych oraz udostępniane przez bibliotekę wysokopoziomowe przekształcenia. Ta funkcjonalność znajduje zastosowanie przede wszystkim w procesie przetwarzania obrazów podczas wykonywania własnych przekształceń. Użycie Numpy przyspiesza również proces tworzenia gotowych tablic ze zdefiniowaną funkcyjnie zawartością.

## 5.4. Tensorflow

Otwarty framework rozwijany przez Google, służący do tworzenia rozwiązań opartych o uczenie maszynowe i głębokie sieci neuronowe. Oferuje możliwość uczenia przy użyciu GPU, co znacznie przyspiesza proces. Może być wykorzystany jako backend dla wysokopoziomowych bibliotek do wygodnego budowania sieci, jak na przykład Keras.

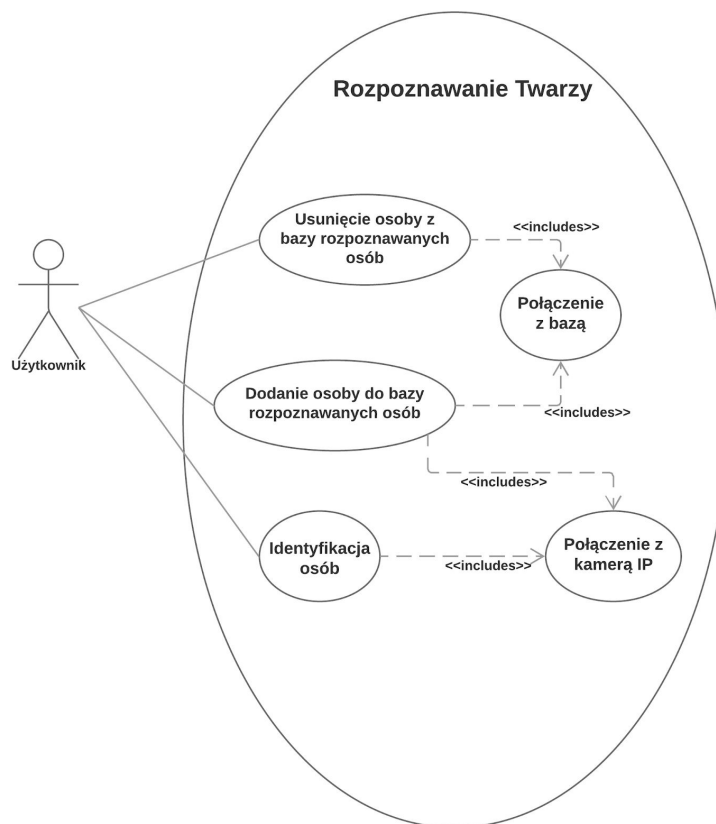
## 5.5. Tkinter

Wbudowana biblioteka języka python umożliwiająca tworzenie graficznego interfejsu użytkownika w szybki i przejrzysty sposób. Jest oparta na bibliotece Tk.

## 5.6. JSON

JavaScript Object Notation - format reprezentacji danych. Cechuje go hierarchiczna struktura oraz zastosowanie par klucz-wartość. Łatwy do wykorzystania przy użyciu słowników z języka Python oraz dedykowanej biblioteki. W projekcie jest wykorzystywany do przechowywania wektorów cech wyznaczonych dla danej twarzy.

## 6. Architektura rozwiązania.



## 7. Problemy podczas implementacji aplikacji oraz testy.

Podczas implementacji aplikacji wystąpiły trzy główne problemy, z którymi musieliśmy sobie poradzić. Były to: brak GUI w bibliotece OpenCV, Wykrywanie twarzy za pomocą kaskad Haara i Local Binary Patterns oraz Identyfikacja osób za pomocą gotowych algorytmów biblioteki OpenCV.

## 7.1 Brak GUI w OpenCV

Z powodu braku możliwości utworzenia GUI przy użyciu OpenCV oraz wygody dla użytkownika jaką niosą ze sobą wykorzystanie guzików i pól tekstowych (w przeciwieństwie do stosowania jedynie zdefiniowanych wcześniej przycisków klawiatury, bez wbudowanej w interfejs instrukcji oraz mapy klawiszy), zdecydowaliśmy się na użycie osobnej biblioteki w celu stworzenia osobnego GUI, które umożliwi ustawianie parametrów, dostęp do instrukcji oraz ingerencję w zbiór zapisanych osób. Zwiększa to wygodę użytkownika, gdyż w klasycznym przypadku w celu zmiany parametrów algorytmu należy dokonać ich bezpośrednio w kodzie źródłowym.

Minusem jest wykorzystanie kilku odrębnych okien, które pojawiają się po uaktywnieniu poszczególnych funkcji, jednakże mechanizm używany przez OpenCV do wyświetlania obrazu nie daje możliwości na zabudowanie w oknie innej biblioteki.

## 7.2. Wykrywanie twarzy za pomocą kaskad Haara i Local Binary Patterns.

Początkowo zdecydowaliśmy się na użycie gotowego pliku XML z kaskadą Haara i wprowadzenie go jako klasyfikatora do biblioteki OpenCV, która z jego pomocą miała sobie poradzić z wykrywaniem twarzy osób na przechwytywanym przez kamerę obrazie. Niestety efekt nie był do końca zadowalający. Aplikacja nie miała problemu z wykrywaniem nieruchomych twarzy, lecz przy ruchomym obrazie z kamery potrafiła momentalnie zaprzestać detekcji oraz występowały znaczne problemy przy rozpoznawaniu twarzy, która nie jest zorientowana frontalnie do kamery (nawet delikatne odchylenie zaburzało wykrywanie obiektu). Nawet łącząc kaskady dotyczące frontu i profili twarzy, biblioteka nie do końca dawała sobie radę z ciągłym i stabilnym wykrywaniem twarzy, w związku z tym trzeba było znaleźć inne rozwiązanie.

Podjęta została kolejna próba z wykorzystaniem biblioteki OpenCV, jednak tym razem zamiast kaskady Haara podany został XML zawierający Local Binary Patterns (LBP). Niestety skutek był bardzo podobny, aplikacja co jakiś czas gubiła twarz lub znajdowała ją w miejscu na obrazie, gdzie nie było żadnej.

Algorytmy zawarte w bibliotece OpenCV przy różnych parametrach wciąż nie dawały zadowalających efektów, skuteczność wykrywania była niska, a w przypadkach bardziej czułych ustawień pojawiały się wyniki fałszywie pozytywne.

W związku z powyższymi problemami, zdecydowaliśmy się nie wykorzystywać wbudowanego w bibliotekę OpenCV do wykrywania twarzy na obrazie z kamery. W toku dalszych poszukiwań doszliśmy do rozwiązania końcowego jakim jest wykorzystanie wielozadaniowych, konwolucyjnych sieci neuronowych (MTCNN), z implementacją opartą na bibliotece Tensorflow, które uczą się rozpoznawać twarz z predefiniowanych modeli (link do ich pobrania znajduje się w instrukcji użytkowania). Dzięki temu uzyskaliśmy płynne i stabilne wykrywanie twarzy w aplikacji z dużą odpornością na zmiany orientacji twarzy oraz elastycznością w kwestii mimiki.

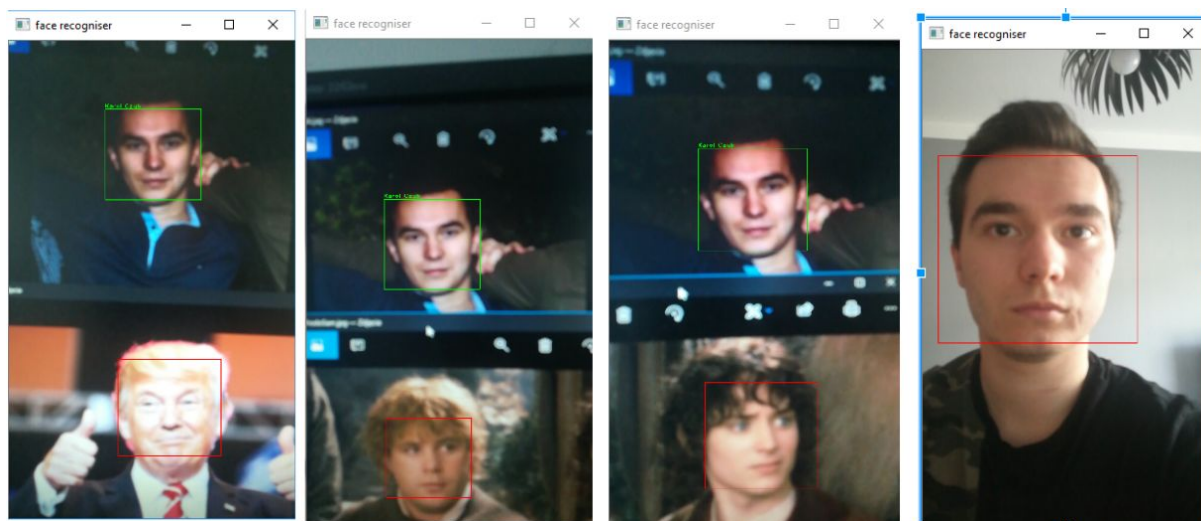
### **7.3. Identyfikacja osób za pomocą gotowych algorytmów biblioteki OpenCV.**

Po uporaniu się z problemem wykrywania twarzy, nadeszła pora na nauczanie aplikacji rozpoznawania, do kogo dana twarz należy. Początkowo zdecydowaliśmy się w tym aspekcie dać kolejną szansę bibliotece OpenCV, która oferuje gotowe algorytmy do nauki rozpoznawania twarzy takie jak:

- Local Binary Patterns Histograms (LBPH)
- Eigenfaces
- Fisherfaces



Stworzony został zbiór treningowy zawierający zdjęcia danej osoby i odpowiednio oznaczony etykietą, którą algorytm miał przypisywać do danej twarzy. Najlepiej ze wszystkich algorytmów sprawdzał się LBPH, a efekty jego pracy były następujące:



W trzech pierwszych przypadkach od lewej algorytm miał do rozpoznania zdjęcia osoby, która znajduje się w bazie (Karol Czub) oraz wykrycie twarzy znanej osoby, której w bazie nie posiada i nie przypisanie jej żadnej etykiety. Z tym zadaniem jak widać poradził sobie prawidłowo, jednak ostatni zrzut ekranu pochodzi z obrazu z kamery na żywo, z czym algorytm już nie potrafił sobie poradzić w żaden sposób.

W związku z tym po raz kolejny gotowe rozwiązanie, oferowane przez bibliotekę OpenCV zostało odrzucone i ostatecznie zastosowany został po raz kolejny Tensorflow, wraz z wytrenowanym wcześniej modelem (również do pobrania z linku w instrukcji użytkownika), który pozwala na rozpoznawanie cech wykrywanej twarzy i przekonwertowanie ich na stu dwudziestoośmio wymiarowy wektora (Dokładne działanie w punkcie 4.1. Identyfikacja osób).

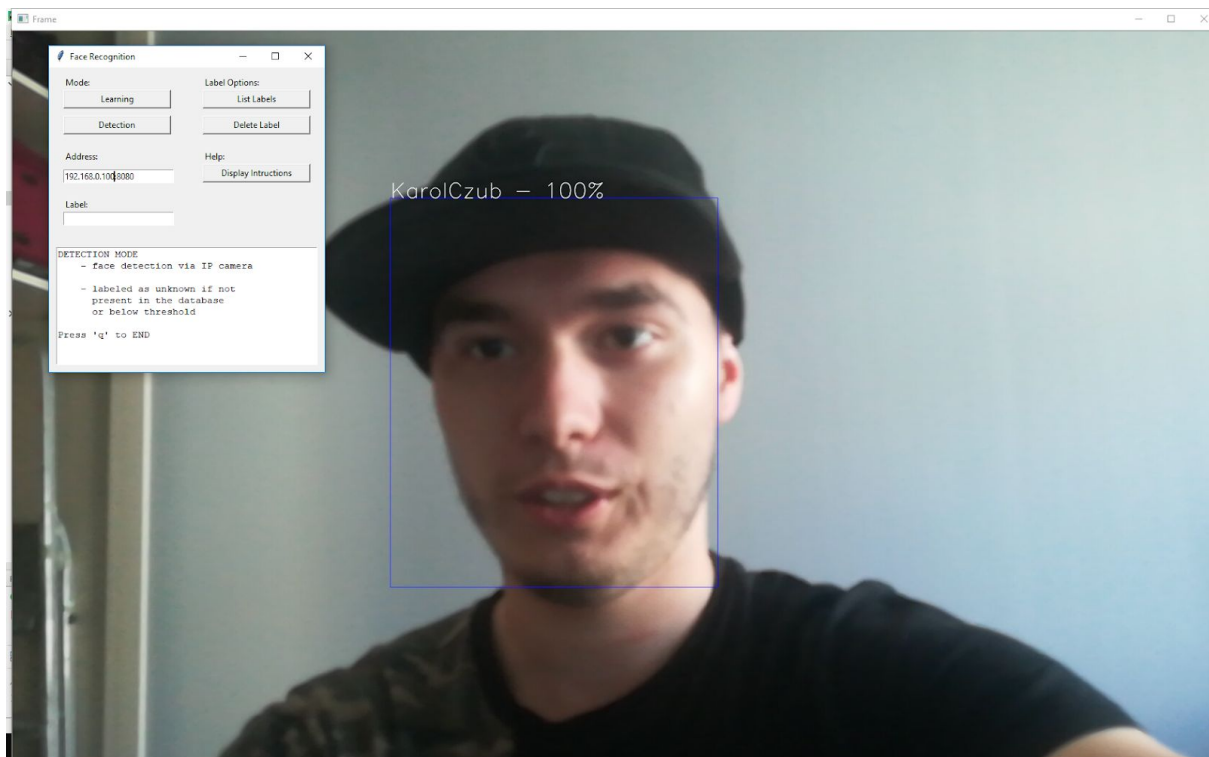
Równie ważną kwestią prócz samej skuteczności jest również wydajność algorytmów, w przypadku problemów, w których występują obrazy statyczne nie jest to tak znaczące jak, gdy przechwytywany jest dynamiczny obraz z kamery, a przetwarzanie i wykrywanie trzeba wykonać w czasie rzeczywistym. W tym aspekcie algorytmy uczenia maszynowego mają lepszą wydajność (pomimo długiego czasu nauczania).

## 7.4 Testy.

Przy sprawdzaniu działania aplikacji, został wykonany szereg testów mających sprawdzić jej dokładność i umiejętność radzenia sobie w różnych przypadkach. Tutaj przedstawię kilka z nich:

### 7.4.1. Nakrycie głowy.

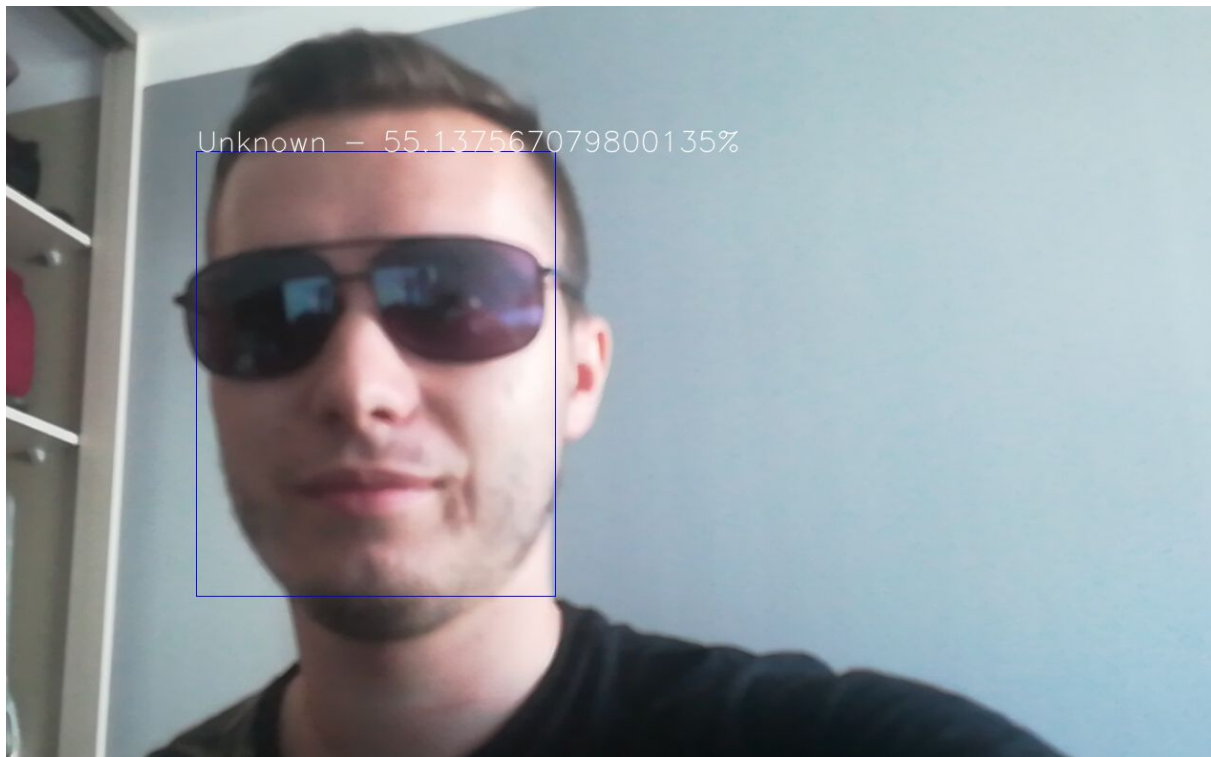
Pierwszą rzeczą jaka została przetestowana było sprawdzenie, jak poradzi sobie aplikacja w przypadku kiedy normalnie rozpoznawana osoba założy czapkę z daszkiem, której nie miała na sobie podczas dodawania swojej twarzy do bazy. Efekt był następujący:



Jak widać aplikacja nie zwraca uwagi na przysłonięcie czoła przez czapkę i ze stuprocentową pewnością była w stanie stwierdzić, że przed kamerą znajduje się ta sama osoba co w bazie. Wniosek jest taki, że do bycia rozpoznanym przez aplikację nie trzeba zdejmować, jeżeli mamy ją aktualnie na sobie.

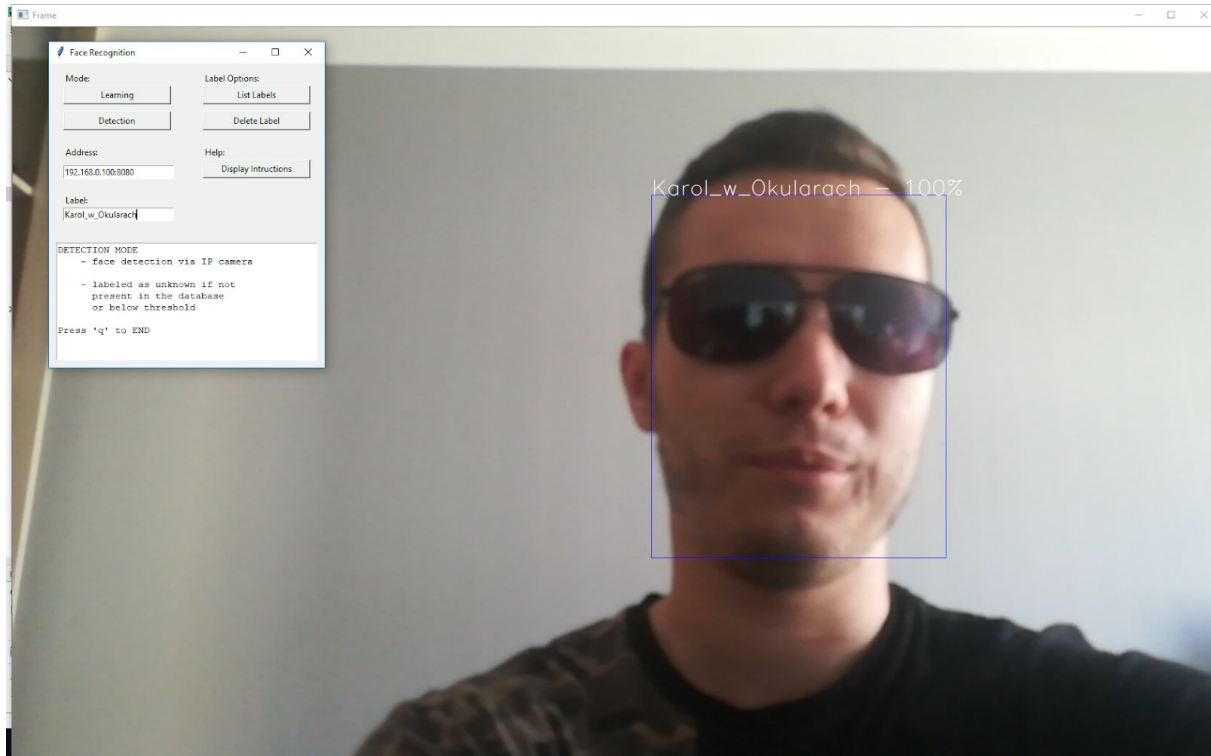
#### 7.4.2. Okulary przeciwsłoneczne.

Kolejną ważną kwestią, która została poddana testom jest założenie przez osobę normalnie rozpoznawaną okularów przeciwsłonecznych, których nie miała na sobie podczas dodawania swojej twarzy do bazy. Efekt był następujący:



Jak widać aplikacja nie jest w stanie sobie poradzić w przypadku, kiedy osoba rozpoznawana bez okularów przeciwsłonecznych osoba, zdecyduje się na ich założenie. Dzieje się tak ze względu na słabą widoczność oczu i całkowitą zmianę barwy w miejscu, w którym powinny one występować. W związku z tym algorytm nie jest w stanie dopasować twarzy do żadnego z nauczonych wzorców i nie rozpoznaje jej. Oczywiście w przypadku zwykłych, przezroczystych okularów takie problemy nie występują, w związku z czym osoby, które na przykład czasem noszą okulary, a czasem soczewki nie muszą martwić się o bycie niewykrytym przez aplikację.

W przypadku jeżeli podczas dodawania nowej osoby będzie ona miała na sobie okulary przeciwsłoneczne, wtedy efekt przy rozpoznawaniu będzie następujący:



Jak widać aplikacja nie ma problemu z rozpoznaniem osoby w okularach przeciwsłonecznych, jeżeli jest nauczona wyglądu danej osoby w nich. W związku z tym jeżeli chcemy mieć pewność, że dana osoba zostanie rozpoznana zarówno w okularach jak i bez nich, to należy dodać osobno dwie etykiety dla tej osoby: jedna, w przypadku kiedy jest ona bez okularów, a druga w przypadku, kiedy ma ona je na sobie.

Jako dodatkową ciekawostkę można dodać fakt, że aplikacja po nauczaniu jej wyglądu danej osoby w okularach przeciwsłonecznych nie ma większego problemu w przypadku, kiedy podczas trybu identyfikacji osób założy inne okulary przeciwsłoneczne niż miała na sobie podczas dodawania.



Podczas dodawania etykiety "Karol\_w\_Okularach", posiadałem na sobie czarne okulary z poprzednich dwóch zdjęć. Jak widać zmienił się więc zarówno kształt okularów, kolor oprawek, a nawet kolor szkła. Mimo to aplikacji udało się mnie rozpoznać. Zmiany te co prawda obniżyły pewność aplikacji co do dopasowania odpowiedniej etykiety z bazy do niecałych 94 procent, ale jest to i tak na tyle dokładne dopasowanie, że aplikacja rozpoznaje tę samą osobę, którą ma w bazie.



#### 7.4.3. Zasłonięte usta przez wysoki szal.

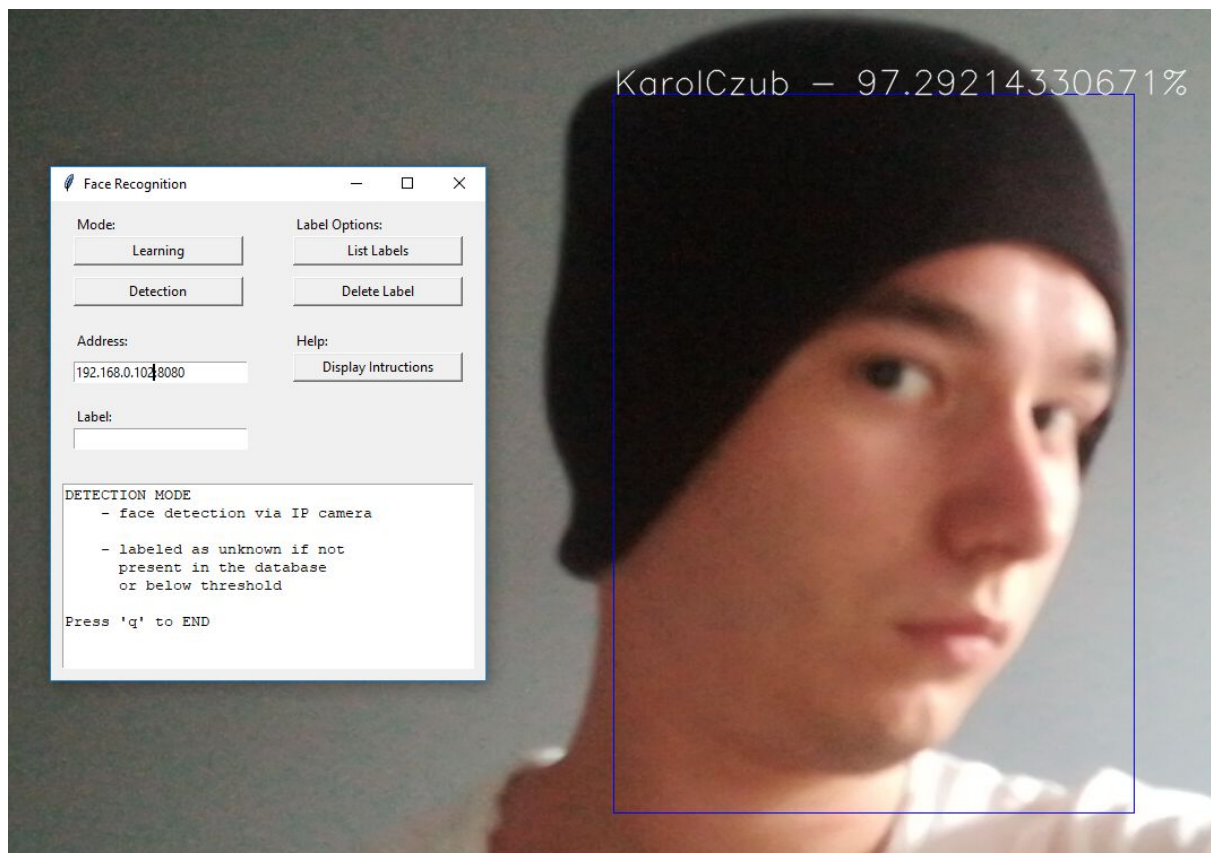
Następną rzeczą, która została poddana testom jest posiadanie zakrytych ust przez rozpoznawaną normalnie osobę. Przykładowo może się to zdarzyć jako skutek założonego szala, którego ta osoba nie miała na sobie podczas dodawania swojej twarzy do bazy. Efekt był następujący:



Jak widać aplikacja jest w stanie rozpoznać taką osobę poprawnie, jednak dopasowanie do wzorca, a zarazem pewność co do rozpoznania osoby znacznie maleje (ze 100 do 82% jak widać na załączonym obrazku). Co więcej, jeżeli szal będzie podniesiony zbyt wysoko i zasłoni nie tylko usta, ale również nos, wówczas nie będzie szansy na zidentyfikowanie takiej osoby przez aplikację.

#### 7.4.4. Uszy zasłonięte przez zimową czapkę lub długie włosy.

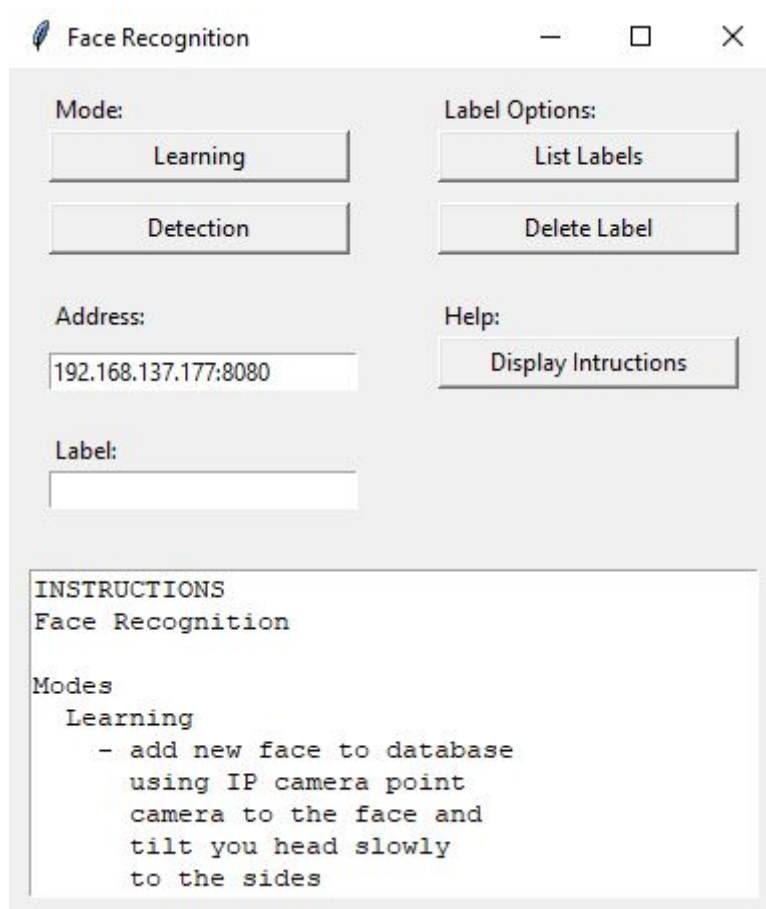
Ostatnia z przetestowanych sytuacji dotyczy się przypadku, w którym normalnie rozpoznawana osoba ma zasłonięte uszy podczas trybu identyfikacji osób, co nie miało miejsca podczas dodawania tej osoby do bazy danych. Przykładem takiej sytuacji jest założenie czapki zimowej lub rozpuszczenie długich włosów, w sposób zasłaniający uszy. Efekt był następujący:



Jak widać aplikacja nie miała większego problemu z rozpoznaniem osoby, która ma zakryte uszy, nawet pomimo ustawienia się do kamery z profilu (gdzie powinny być one bardziej widoczne). Spadek pewności aplikacji co do rozpoznania danej osoby był nieznaczny (ponad 97% pewności), w związku z tym można stwierdzić, że nie trzeba specjalnie odsłaniać uszu na potrzeby użytkowania aplikacji.

## 8. Instrukcja użytkowania.

Aby uruchomić aplikację, wymagany jest Python, wraz z zainstalowanymi bibliotekami: OpenCV, Numpy, Tensorflow, Tkinter. Kiedy mamy już tak skonfigurowany interpreter, należy pobrać aplikację z githuba: <https://github.com/Nagerrack/PT-Face-Detection-Tracking/tree/master/Face%20Recognition> oraz modele, niezbędne do prawidłowego rozpoznawania twarzy z dysku google: <https://drive.google.com/file/d/0Bx4sNrhhBr3TDRMMUN3aGtHZzg/view> . Modele po pobraniu, należy umieścić w folderze models, który to znajduje się w głównym katalogu projektu. Następnie wystarczy jedynie uruchomić plik GUI.py, aby aplikacja uruchomiła się i wyświetliła interfejs użytkownika:





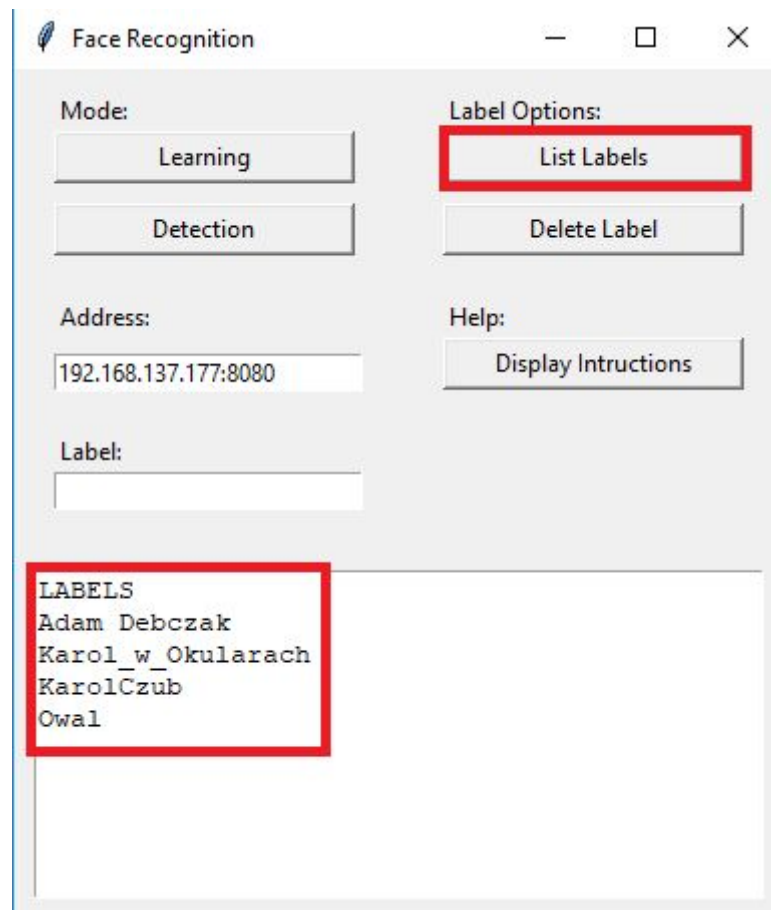
### 8.1. Pola i przyciski interfejsu:

- Pole Address jest miejscem, w którym użytkownik musi wprowadzić IP wraz z portem, na którym jest połączony z kamerą IP. Jest to absolutnie niezbędne i bez tego aplikacja nie pozwoli na działanie w żadnym trybie pracy. Adres wraz z portem wpisujemy według formatu: 0-255.0-255.0-255.0-255:port. Przykładowy, poprawnie wprowadzony adres kamery IP, to: 192.168.0.4:8080
- Pole Label jest niezbędne jedynie w dwóch przypadkach: w przypadku wybrania trybu Learning lub kliknięcia przycisku Delete Label, których opis można znaleźć poniżej. Aplikacja wymaga, aby osoba, która będzie dodana do bazy była opisana przy pomocy niepustej etykiety lub w przypadku usuwania rekordu z bazy, by określić etykietę przeznaczoną do usunięcia ze zbioru.

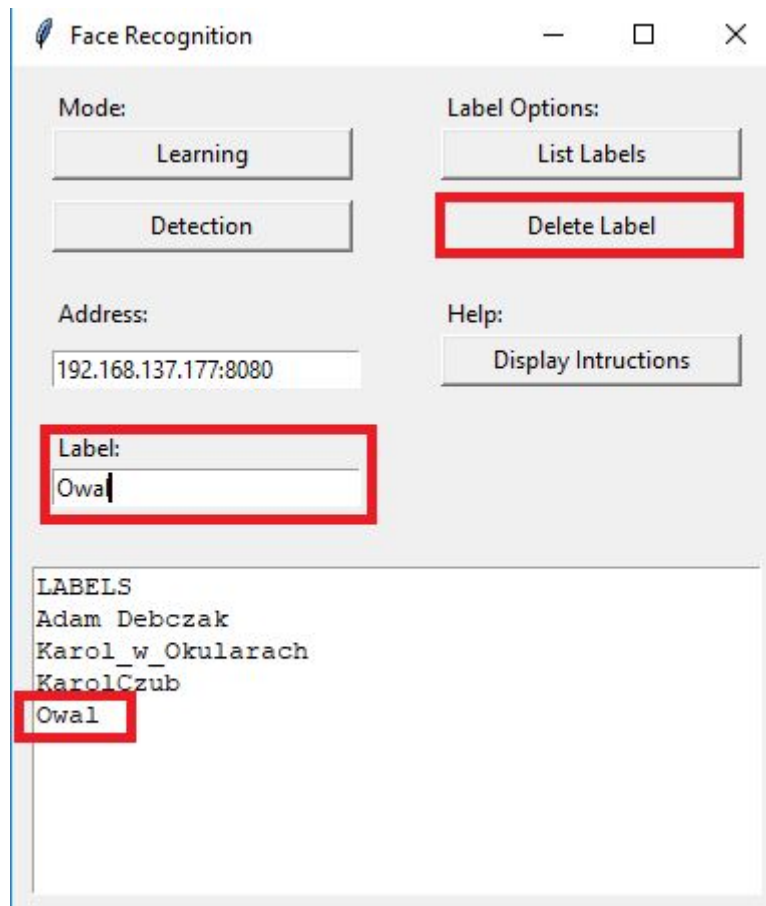
Kiedy już wiemy, które pole musimy wypełnić w jakim przypadku, możemy wybrać tryb pracy aplikacji:

- Przycisk Learning służy do uruchomienia trybu dodawania nowej osoby do bazy danych. Do jego działania należy wpisać adres IP w polu Address oraz etykietę dodawanej osoby, która będzie się przy niej wyświetlać w polu Label.
- Przycisk Detection służy do uruchomienia trybu identyfikacji osób, które znajdują się w polu widzenia kamery. Do jego działania należy wpisać adres IP kamery, z którą aplikacja ma się połączyć.

Pozostałe pola i przyciski jakie znajdują się w interfejsie aplikacji to:



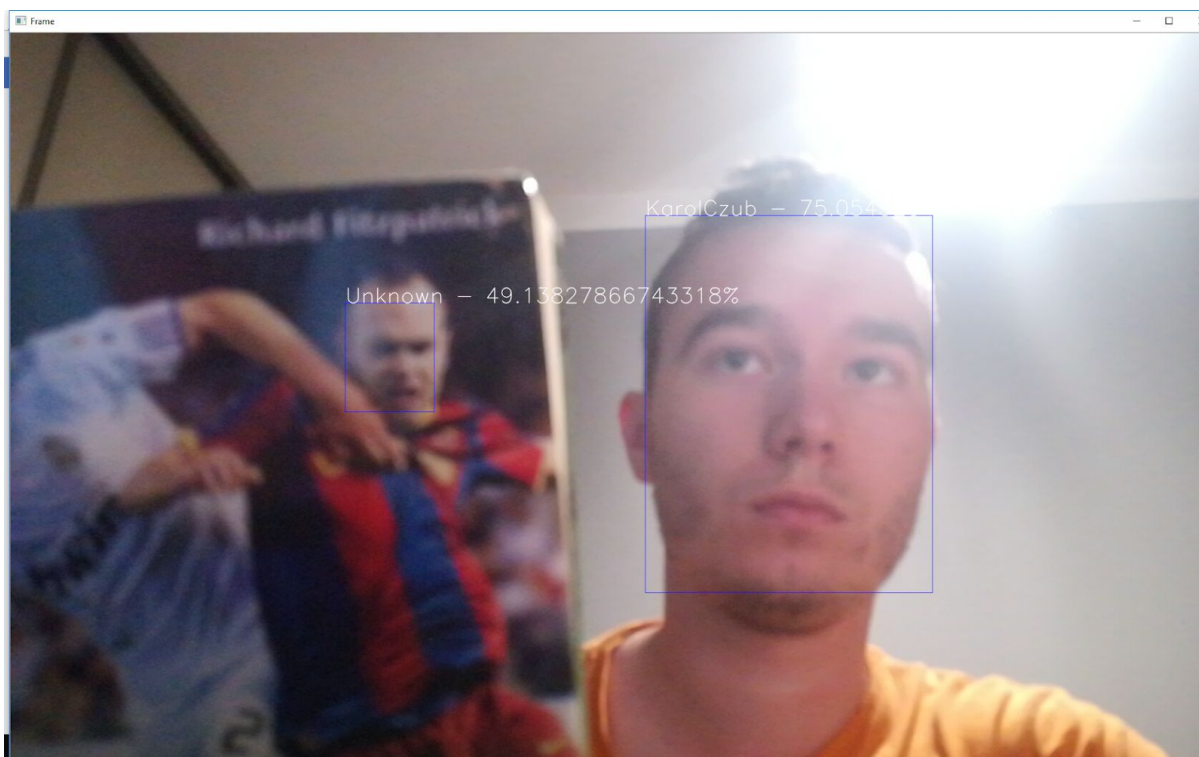
- Przycisk List Labels służy do wyświetlenia listy etykiet, do których zapisane są w bazie modele rozpoznawanych twarzy. Po naciśnięciu ukazują się one użytkownikowi w polu tekstowym, znajdującym się w dolnej części ekranu.



- Przycisk Delete Label służy do usuwania z bazy danych modelu twarzy, przypisanego do wskazanej etykiety. Do jego działania należy przed naciśnięciem przycisku, wpisać w polu Label etykietę osoby, której model chcemy usunąć. Jeżeli nie pamięta się etykiety danej osoby lub nie jest się pewnym jej pisowni, można ją sprawdzić wywołując listę etykiet przyciskiem List Labels. Po usunięciu, można jeszcze raz wywołać listę osób, w celu zweryfikowania, czy została ona zredukowana o wskazaną pozycję.
- Przycisk Display Instructions służy do wyświetlenia instrukcji użytkownika aplikacji w polu tekstowym, w dolnej części aplikacji. Jest to w momentach, kiedy użytkownik wywoła np. listę etykiet i następnie chce wrócić do widoku instrukcji.
- Pole tekstowe w dolnej części interfejsu, służące do wyświetlania instrukcji dla użytkownika lub listy etykiet, do których zapisane są w bazie modele rozpoznawanych twarzy

## 8.2. Tryb identyfikacji osób.

Uruchamiany za pomocą przycisku Detection. Do jego uruchomienia wymagany jest poprawnie wpisany w pole Address, adres IP oraz port kamery.



Po naciśnięciu przycisku, zostanie otwarte nowe okno, na którym wyświetlony zostanie podgląd widoku z kamery. Na obrazie, w czasie rzeczywistym zaznaczone zostają niebieskim prostokątem, wszystkie wykryte przez aplikację twarze. Następnie aplikacja przyporządkowuje twarzom stosowną etykietę, która wyświetlona zostaje nad górną krawędzią prostokąta. Jeżeli stwierdziła zgodność, z najlepiej dopasowanym modelem, wynoszącą więcej niż 70% (liczba po prawej stronie etykiety, poprzedzona myślnikiem), to nadawana jest etykieta danego modelu. Jeżeli mniej, to przyporządkowana zostaje etykieta "Unknown".

Rzeczy, które mają wpływ na poprawne rozpoznawanie osób i na które użytkownik powinien zwrócić uwagę, to:

- jakość obrazu przechwytywanego przez kamerę. Co oczywiste - im lepszy obraz, tym łatwiej będzie o poprawne rozpoznanie osoby na nim. Dotyczy to nie tylko rozdzielczości, ale również stabilności obrazu, gdyż poruszanie kamerą doprowadzi do jego rozmazania.

- oświetlenie. Tak jak w przypadku zamieszczonego wyżej obrazka, złe światło, które świeciło w kamerę osłabiło pewność rozpoznania do 75%. Warto zauważyć, że aplikacja dalej rozpoznała twarz, gdyż pewność dalej przekraczała próg 70%, jednak jest to sygnał, że coś utrudnia poprawną identyfikację osób.
- ekspozycja twarzy. Jeżeli twarz zostanie przysłonięta w znaczący sposób, może to wpłynąć na rozpoznawanie danej osoby przez aplikację. Program radzi sobie z niektórymi zmianami, takimi jak zapuszczenie zarostu przez osobę identyfikowaną, czy posiadania przez nią nakrycia głowy (np. czapka z daszkiem, przysłaniająca czoło), jednak założenie okularów przeciwsłonecznych lub poddanie się operacji plastycznej powiększania ust, może doprowadzić do problemów z identyfikacją takiej osoby.

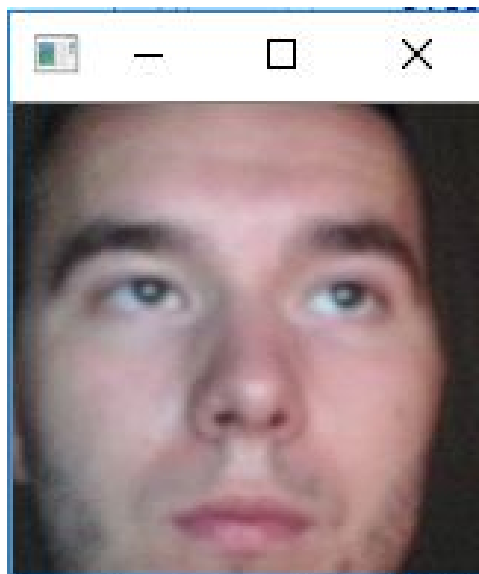
Tryb identyfikacji osób można wyłączyć poprzez naciśnięcie klawisza 'q' na klawiaturze lub zamknięcie okna (podglądu widoku z kamery) krzyżykiem w rogu ekranu.

### **8.3. Tryb dodawania nowej osoby.**

Uruchamiany za pomocą przycisku Learning. Do jego uruchomienia wymagane jest wcześniejsze wprowadzenie etykiety osoby, która zostanie przypisana do zapisanego w bazie danych modelu twarzy. Oprócz tego należy poprawnie wpisać w pole Address, adres IP oraz port kamery.

Po poprawnym uzupełnieniu wymienionych wyżej pól oraz naciśnięciu przycisku Learning, pole tekstowe w dolnej części interfejsu poinformuje nas, że uruchomiony został tryb dodawania nowej osoby i przybliży sposób jego użycia. Nowe okno uruchomi się dopiero w momencie, w którym aplikacja wykryje twarz, na obrazie przechwyconym za pomocą kamery. Jeżeli nie chce się ono uruchomić, a aplikacja nie pokazuje żadnego błędu związanego z istniejącą już etykietą lub złym adresem IP kamery, to należy upewnić się, że jesteśmy dobrze widoczni przez kamerę.

Po poprawnym wykonaniu powyższych czynności, powinien wyświetlić się w nowym oknie podgląd przechwytywanej twarzy. Wygląda on następująco:



Następnie należy powoli obracać głowę w taki sposób, aby pokazać twarz przed kamerą zarówno od frontu jak i z każdego profilu. Wskazane jest również, aby wolno zmieniać mimikę swojej twarzy, podczas gdy aplikacja się jej uczy oraz zaprezentować swoją twarz pod pewnym kątem zarówno od dołu jak i od góry. Wszystkie te operacje nie są obowiązkowe, ale pozytywnie wpłyną na identyfikację takiej osoby w przyszłości. Jeżeli, którąś z tych rzeczy wykonamy za szybko, to może ona nie zostać zarejestrowana przez aplikację, w związku z tym należy się upewnić, że jest ona widoczna na podglądzie i dopiero wtedy przejść dalej. Proces uczenia się naszej twarzy kończymy za pomocą klawisza 'q' na klawiaturze.

Należy pamiętać, że warunki oraz ekspozycja naszej twarzy, w momencie dodawania jej do bazy danych ma kluczowy wpływ na jej identyfikację w przyszłości. Jeżeli będzie ona słabo oświetlona lub częściowo przysłonięta, to mogą wystąpić problemy z rozpoznaniem jej w przyszłości. W takim wypadku należy usunąć twarz z naszej bazy za pomocą przycisku Delete Label oraz wprowadzić ją poprawnie, na nowo.

## 9. Bibliografia.

- OpenCV kaskada Haara  
[https://docs.opencv.org/3.3.0/d7/d8b/tutorial\\_py\\_face\\_detection.html](https://docs.opencv.org/3.3.0/d7/d8b/tutorial_py_face_detection.html)
- Kaskada Haara vs Histogram LBP  
<https://www.superdatascience.com/opencv-face-detection/>
- Konwolucyjne sieci neuronowe w wykrywaniu i rozpoznawaniu twarzy  
<http://cs231n.stanford.edu/reports/2017/pdfs/222.pdf>
- Klasyfikacja emocji  
[https://github.com/oarriaga/face\\_classification](https://github.com/oarriaga/face_classification)
- Kaskada konwolucyjnych sieci neuronowych  
[https://kpzhang93.github.io/MTCNN\\_face\\_detection\\_alignment/](https://kpzhang93.github.io/MTCNN_face_detection_alignment/)
- Przegląd metod segmentacji obrazów  
<https://blog.athelas.com/a-brief-history-of-cnns-in-image-segmentation-from-r-cnn-to-mask-r-cnn-34ea83205de4>
- Motion tracking  
<https://www.pyimagesearch.com/2015/05/25/basic-motion-detection-and-tracking-with-python-and-opencv/>
- Tkinter dokumentacja  
<https://docs.python.org/2/library/tkinter.html>
- OpenCV dokumentacja  
<https://docs.opencv.org/3.4.1/>
- Keras dokumentacja  
<https://keras.io/>