Dokumentacja struktury wewnętrznej

Who’s there

Spis treści

[Główne moduły 3](#_Toc168512217)

[Face recognition 3](#_Toc168512218)

[Tkinter 3](#_Toc168512219)

[OpenCV 3](#_Toc168512220)

[YAML 3](#_Toc168512221)

[Struktura programu 3](#_Toc168512222)

[Przechowywanie danych 3](#_Toc168512223)

[Folder „Photos” 3](#_Toc168512224)

[Plik „known\_faces.yml” 4](#_Toc168512225)

[Pliki programów 4](#_Toc168512226)

[Main.py 4](#_Toc168512227)

[\_\_main\_\_ 4](#_Toc168512228)

[callback() 4](#_Toc168512229)

[grantAccess() 4](#_Toc168512230)

[face\_confidence() 4](#_Toc168512231)

[Take\_Photo.py 4](#_Toc168512232)

[\_\_main\_\_ 4](#_Toc168512233)

[callback() 4](#_Toc168512234)

[save\_photo() 4](#_Toc168512235)

[delete\_photos() 5](#_Toc168512236)

[show\_widget() 5](#_Toc168512237)

[get\_existing\_names() 5](#_Toc168512238)

[count\_users() 5](#_Toc168512239)

[Trening.py 6](#_Toc168512240)

# Główne moduły

## Face recognition

Moduł face recognition dostarcza podstawowe narzędzia do identyfikacji i rozpoznawania twarzy na grafikach i zdjęciach

## Tkinter

Tkinter jest modułem oferującym szeroka gamę elementów graficznych, na bazie których skonstruowano wszystkie elementy GUI programu

## OpenCV

Moduł OpenCV posłużył za sterowniki kamery komputera oraz przetwarzanie surowego obrazu poprzez dodawanie odpowiednich elementów graficznych, zapis i konwersje do odpowiedniego formatu.

## YAML

Moduł udostępniające funkcje związane z tworzeniem, edycją oraz zapisem plików .yml. W tym projekcie stosowane są one jako format zapisu cech charakterystycznych uzyskanych z zdjęć wprowadzonych użytkowników. Dane z pliku .yml są odczytywane przez główny program rozpoznający tważe.

# Struktura programu

Program składa się z 3 plików wykonywalnych napisanych w języku python, pliku yml do przechowywania twarzy w sposób gotowy do odczytu oraz folderu „Photos” do przechowywania wszystkich zdjęć wprowadzonych przez użytkowników.

Pliki wykonywalne programu:

* main.py – program rozpoznający twarze na bazie danych zawartych w pliku yml. Nie wprowadza żadnych zmian zarówno do pliku yml oraz bazy zdjęć w folderze „Photos”
* take\_photo.py – Program do dodawania zdjęć do folderu „Photos”, kontroluje aktualną ilość użytkowników i ilość zdjęć im przypisaną. Udostępnia on opcję usunięcia wszystkich zdjęć przypisanych do danej nazwy. Nie ma bezpośredniego wpływu na plik yml.
* trening.py – Przeprowadza konwersje grafik w folderze „Photos” na plik yml. Nie wprowadza zmian do folderu, a stary plik .yml zostaje całkowicie nadpisany nowym.

# Przechowywanie danych

## Folder „Photos”

Przechowuje surowe zdjęcia w formacie .jpg uzyskane wskutek działania programu take\_photo.py. Każde zdjęcia posiada unikatową nazwę w formacie : <NazwaUżytkownika\_numer.jpg>, gdzie NazwaUżytkownika to dowolny ciąg znaków podany podczas zapisu zdjęcia, a numer to cyfra z zakresu 0-9 dodawana automatycznie przez program w celu rozróżnienia zdjęć i kontrolowania ich ilości. W przypadku przekroczenia dozwolonego indeksu nadpisywane są zawsze w kolejności od zdjęcia z najniższym numerem. Do jednego użytkownika może być przypisane 10 zdjęć w jednym momencie. Dodanie zdjęć do folderu „Photos” nie skutkuje natychmiastowym dodaniem użytkownika do bazy zatwierdzonych przez program main.py. W celu zatwierdzenia zmian w folderze trzeba zaktualizować plik .yml.

## Plik „known\_faces.yml”

Plik generowany na bazie zdjęć z folderu „Photos”. Aktualizowany jest ręcznie poprzez program trening.py. Zawiera dane na temat użytkowników w postaci gotowej do odczytu przez program do rozpoznawania twarzy main.py. Występuje on tylko w pojedynczym egzemplarzu niezależnie od ilości użytkowników.

# Pliki programów

## Main.py

### \_\_main\_\_

### callback()

Parametry: Brak

Zwraca: Brak

Zamyka okno w kontrolowany sposób wyłączając kluczowe opcje i niszczy okno

### grantAccess()

Parametry: Brak

Zwraca: Brak

Otwiera okno programu MsPaint gdy użytkownik uzyska dostęp

### face\_confidence()

Parametry:

* face\_distance: float – Odległość twarzy od kamery, gdzie twarz znajdująca się bliżej gwarantuje większe podobieństwo do zdjęcia w bazie danych
* face\_match\_threshold: float – Próg dopasowania. Jest to wartość graniczna, powyżej której dopasowanie twarzy jest uznawane za niedostateczne

Zwraca:

* Typ ‘string’ – Procentowa pewność dopasowania twarzy w formacie tekstowym

## Take\_Photo.py

### \_\_main\_\_

Generuje GUI programu i uruchamia podgląd z kamery. Podczas całego działania programu aktualizuje licznik zdjęć aktualną wartością

### callback()

Parametry: Brak

Zwraca: Brak

Zamyka okno w kontrolowany sposób wyłączając kluczowe opcje i niszczy okno

### save\_photo()

Parametry:

* name: str – nazwa użytkownika
* photo\_counting: [int, int] – licznik zdjęć
* max\_index: int – maksymalny dozwolony indeks zdjęcia
* first\_index: int – minimalny dozwolony indeks zdjęcia
* photo\_dir: str – ścieżka do zapisu zdjęcia
* frame\_to\_save: None – surowa klatka z podglądu kamery
* widget: tk.Frame – ramka do elementu wyświetlającego informację o zapisie
* window: tk.Tk – okno programu
* label: tk.Label – element trzymający tekst do wyświetlania informacji o zapisie
* button: tk.Button – przycisk aktywujący tą funkcję

Zwraca: Brak

Zapisuje zdjęcie pod aktualną nazwą z następującym indeksem. Jeżeli wprowadzany nowy użytkownik wyświetla zapytanie o dodaniu nowego użytkownika. Pomija zapis zdjęcia gdy twarz nie zostanie wykryta. Aktualizuje tablice zliczającą zdjęcia i wywołuje funkcję do wyświetlania odpowiedniego komunikatu

### delete\_photos()

Parametry:

* name: str – nazwa użytkownika
* photo\_counting: [int, int] – licznik zdjęć
* first\_index: int - minimalny dozwolony indeks zdjęcia
* photo\_dir: str - ścieżka do usunięcia zdjęcia

Zwraca: Brak

Usuwa wszystkie zdjęcia przypisane do danego użytkownika i resetuje licznik. Przed dokonaniem operacji wymaga dodatkowego potwierdzenia

### show\_widget()

Parametry:

* widget: tk.Frame – ramka do elementu wyświetlającego informację
* label: tk.Label -– element trzymający tekst do wyświetlania informacji
* window: tk.Tk – okno programu
* message: str – informacja do wyświetlenia
* color: str – kolor widgetu

Zwraca: Brak

Wyświetla informacje określoną w parametrach i zatrzymuje działanie programu na 1s

### get\_existing\_names()

Parametry:

* photo\_path: str – ścieżka do folderu ze zdjęciami

Zwraca:

* Krotkę pustą lub z nazwami istniejących użytkowników

Zwraca wszystkich użytkowników istniejących w folderze z zdjęciami

### count\_users()

Parametry:

* name: str – nazwa użytkownika
* photo\_dir: str – ścieżka do folderu ze zdjęciami
* max\_index: int - maksymalny dozwolony indeks zdjęcia
* first\_index: int - minimalny dozwolony indeks zdjęcia

Zwraca:

* tabele 2 elementową opisującą ilość zdjęć dla danego użytkownika oraz następny indeks

Zlicza ilość zdjęć przypisanych do danego użytkownika i zwraca indeks następnego zdjęcia do zapisu i ilość wszystkich zdjęć do niego przypisanych. Automatycznie uwzględnia limity podane w parametrach

## Trening.py

### train()

Parametry: Brak

Zwraca: Brak

Służy do trenowania modelu rozpoznawania twarzy na podstawie zdjęć znajdujących się w katalogu "Photos". Tworzy listę nazw użytkowników oraz ich zakodowanych twarzy, a następnie zapisuje te informacje do pliku "known\_faces.yml". W przypadku powodzenia wyświetli komunikat "Photos saved successfully!". W przeciwnym wypadku "Photos are too blurry, please repeat the series!"

### colorLoop()

Parametry: Brak

Zwraca:

* str – Kod koloru w formacie szesnastkowym

Oblicza dynamicznie zmieniający się kolor przycisku

### ButtonUpdate()

Parametry: Brak

Zwraca: Brak

Aktualizuje kolor przycisku co 50 milisekund