****

**Projektowanie interfejsów multimedialnych**

**Aplikacja rozpoznawania twarzy wyzwalana za pomocą mowy**

**Jakub Kuśnierz**

**w61674**

**Informatyka, II semestr studiów magisterskich**

1. **Cel projektu**

Celem projektu było stworzenie aplikacji na urządzenia z mobile z systemem android posiadającymi mikrofon oraz kamerę. Aplikacja ma za zadanie rozpoznawać twarz na kamerze w czasie rzeczywistym oraz obrysowywać ją na ekranie urządzenia. Dodatkowo została ona wyposażona w sterowanie głosowe umożliwiające wykonanie oraz zapisanie zdjęcia za pomocą wydania komendy głosowej w języku polskim oraz angielskim. Pobocznym celem, który udało się zrealizować poprzez stworzenie aplikacji było zaznajomienie z technologiami tworzenia aplikacji na urządzenia mobilne i bibliotekami służącymi przetwarzaniu obrazu oraz mowy. Taka wiedza może przynieść wymierne korzyści w czasie gdy urządzenia mobilne zdobywają coraz większy udział w rynku, a komunikacja różnymi metodami z odbiorcą pozwala dotrzeć do większej liczby osób zainteresowanych korzystaniem z aplikacji.

1. **Wykorzystane technologie**

Systemem operacyjnym na który zdecydowałem się zaprojektować aplikację został Android. Na jego korzyść przemówiła popularność, największy udział w rynku globalnym oraz przede wszystkim posiadanie urządzenia z tym systemem. Wersją systemu którą wybrałem początkowo było APK 27 czyli android w wersji 8.1 o nazwie *Oreo.* Podczas rozwijania aplikacji zdecydowałem się na obniżenie do APK 25 czyli wersji 7.1.2 o nazwie *Nougat.* Zmianawynikała z konieczności przejścia w tworzeniu aplikacji z emulatora dostępnego w Android studio na rzeczywisty telefon spowodowana brakiem możliwości udostępnienia mikrofonu w komputerze do emulatora.

Systemem rozpoznawania obrazu wykorzystanym przeze mnie było OpenCV w wersji 3.4.3

Jest to wolne oprogramowanie, dostępne na wielu platformach umożliwiające przetwarzanie obrazu w czasie rzeczywistym. Biblioteka została stworzona w języku C, jednak może być używana w sposób na systemie android.

Do rozpoznawania mowy wykorzystałem wbudowany w system android mechanizm przetwarzający mowę na tekst dostarczany przez firmę google. Potrafi on poprawnie przetworzyć mowę zarówno w języku angielskim jak i polskim.

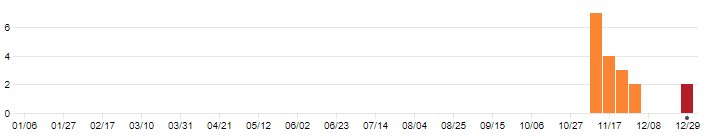
Językiem programowania używanym w projekcie była Java w wersji JDK 1.8. Narzędziem wspomagającym budowanie projektu był gradle. Środowiskiem programistycznym było Android Studio w wersji 3.0.1 wraz z wbudowanym emulatorem urządzeń mobilnych.

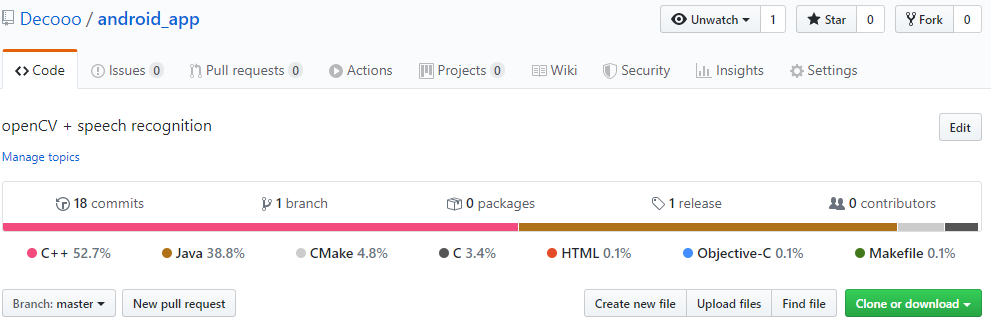
1. **Wersjonowanie projektu**

Do tworzenia projektu wykorzystany został system kontroli wersji git. Repozytorium zostało utworzone na portalu GitHub na początku tworzenia projektu. Zawiera ono cały kod źródłowy aplikacji wraz z historią zmian dokonywanych podczas tworzenia, dokumentujących cały przebieg powstawania projektu.

Repozytorium dostępne jest pod poniższym adresem:

[**https://github.com/Decooo/android\_app**](https://github.com/Decooo/android_app)

 Poniżej zostały pokazane zdjęcia pokazujące wykres historycznych commitów dodawanych czasie trwania projektu. Można z nich zauważyć, że główny okres powstawania samej aplikacji przypadł na listopad 2019, w którym zmiany były oddawane w sposób regularny na przestrzeni 4 tygodni. Drugim okresem jest czas przełomu roku 2019 i 2020 gdy powstawała dokumentacja oraz została wydana wersja release 1.0.0.



1. **Działanie aplikacji**
2. **Przyznanie pozwoleń**

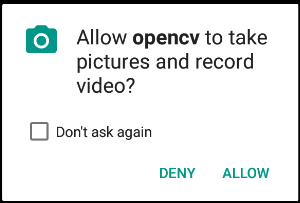
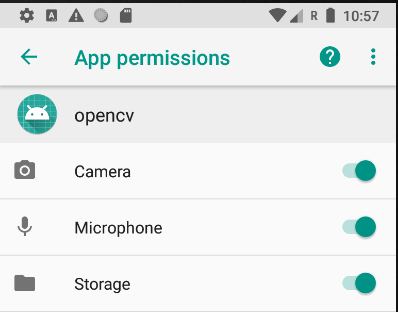
Pierwszą rzeczą jaką użytkownik musi zrobić po uruchomieniu aplikacji to wyrażenie zgody na dostęp do trzech funkcjonalności bez których aplikacja nie może działać poprawnie. Są to:

- Kamera – do przechwytywania obrazu i rozpoznawania twarzy

- Mikrofon – do obsługi za pomocą mowy

- Pamięć – do zapisywania zdjęć w pamięci telefonu.

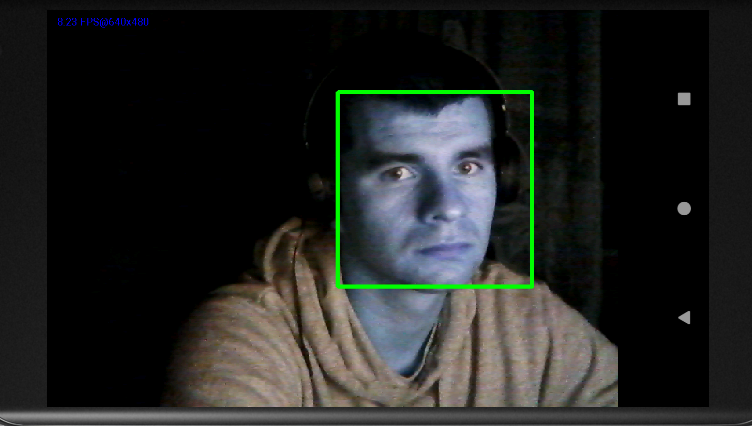
Obowiązkowym uprawnieniem, bez którego aplikacja nie włączy się jest kamera. Pozostałe dwa są opcjonalne. Niemniej jednak zalecane jest wyrażenie zgody, gdyż bez nich nie otrzymamy możliwości pełnego wykorzystania funkcjonalności dostarczanych przez aplikację.

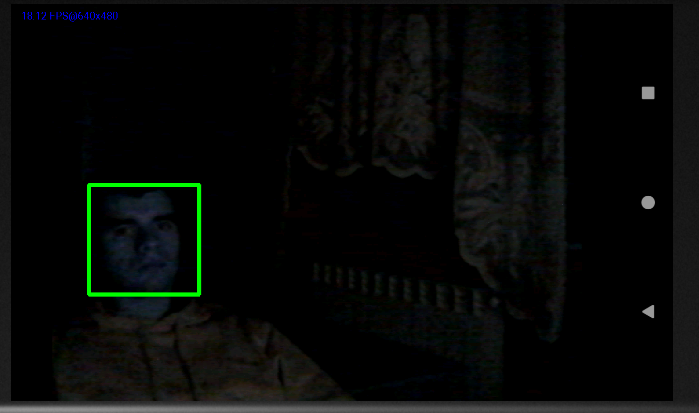


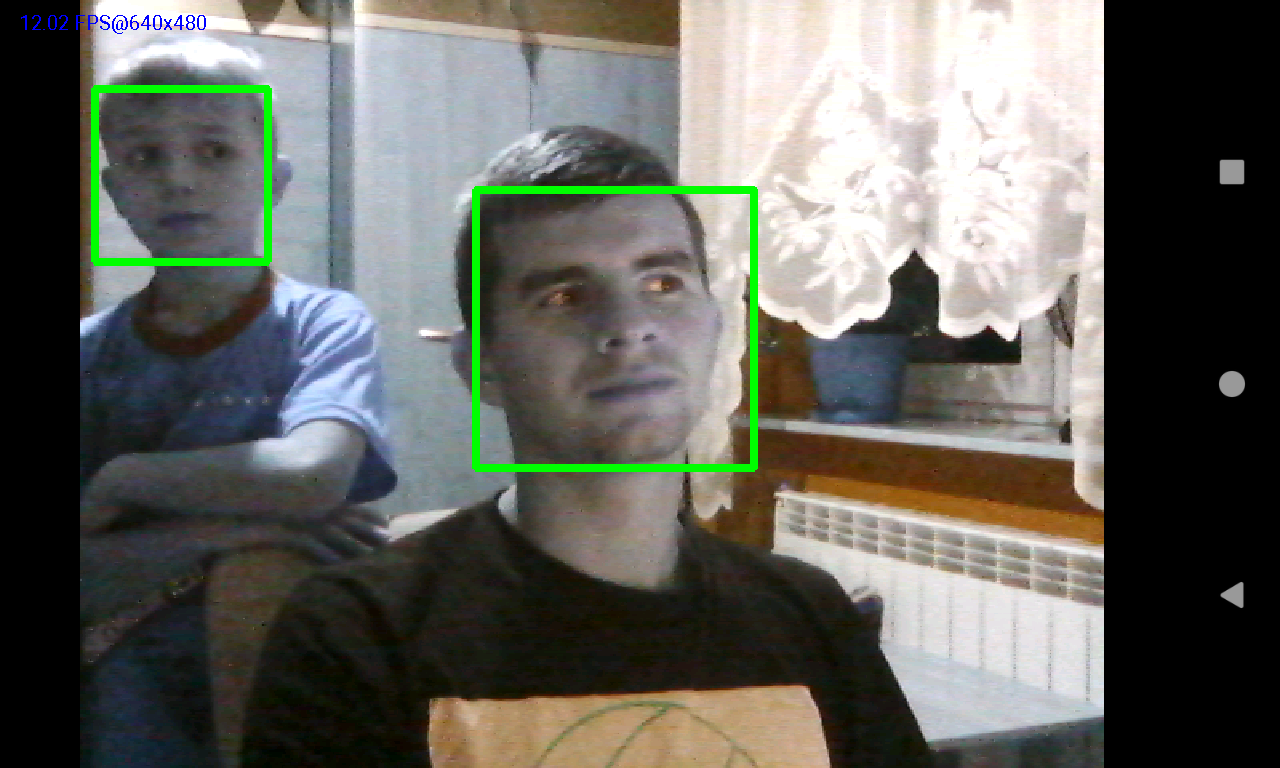
1. **Przyznanie pozwoleń**

Po przyznaniu wymaganych uprawnień, użytkownikowi zostaje wyświetlone okno w którym zostaję odpalona kamera obsługująca tylny aparat urządzenia. Kamera rozpoznaje w czasie rzeczywistym twarz na obrazie i obrysowywuje je zielonym kwadratem. Aby twarz została rozpoznana musi by widoczna dla kamery w sposób umożliwiający dopasowanie do wzorca. Twarz która jest ucięta, zasłonięta lub widoczna tylko z profilu jest trudniejsza, a czasami niemożliwa do rozpoznania z uwagi na brak cech charakterystycznych. Jako wzorzec twarzy został wykorzystany plik, który jest udostępniany w źródłach biblioteki o nazwie *haarcascade\_frontalface\_default.xml.* Jest to plik w formacie XML zawierający ponad 33 000 linii.W fazie testów wykorzystywałem również drugi znacznie mniejszy udostępniany wzorzec o nazwie *frontalface.xml,* jednak jego skuteczność rozpoznawania była na bardzo niskim poziomie. Aplikacja radzi sobie również z wykrywaniem wielu twarzy równocześnie. W lewym górnym rogu kamery zostaje wyświetlana aktualna liczba FPSów.

Poniżej kilka przykładów z działania rozpoznawania twarzy:







1. **Rozpoznawanie mowy**

Funkcja rozpoznawania mowy również działa w czasie rzeczywistym. Umożliwia ona zrobienie zdjęcia z trudnej pozycji, gdy nie możemy spojrzeć w ekran telefonu aby nacisnąć przycisk wykonujący zdjęcie. Działa na zasadzie serwisu uruchomionego w tle aplikacji. Nasłuchuję wszystkich dźwięków, następnie je przetwarzając i analizując. Po wykryciu prawidłowego wzorca wykonuje zdjęcie, które zostaję zapisane do pamięci lokalnej telefonu. Dźwięki przetwarzane i rozpoznawane są w językach polskim i angielskim. Aby zdjęcie zostało wykonane system musi rozpoznać obecnie zaprogramowane poniższe wzorce:

- *Zrób zdjęcie –* w języku polskim

- *Take photo* – w języku angielskim

Użytkownik o działaniu systemu rozpoznawania zostaje informowany komunikatami w formie anndroidowego „*Toast”* czyli małej wiadomości na ekranie podobnej do podpowiedzi lub wyskakującego powiadomienia wyświetlanej jako popup. Do celów testowych uruchomiony został również dźwięk „piknięcia” informujący nas o każdej próbie przetworzenia wyłapanego tekstu.

Komunikaty jakie zostają wyświetlone na ekranie:

- „*Aktywowano rozpoznawanie mowy*” – przy starcie usługi

- „*Wyłączono rozpoznawanie mowy*” – po wyłączeniu usługi

- „*Tekst nie rozpoznany! Spróbuj ponownie*” – gdy przetworzony tekst nie pasuję do obsługiwanych wzorców

- „*Zrobiono zdjęcie*” – gdy rozpoznano wzorzec i wykonano zdjęcie

Przykładowe wyświetlenie komunikatu widoczne jest poniżej.

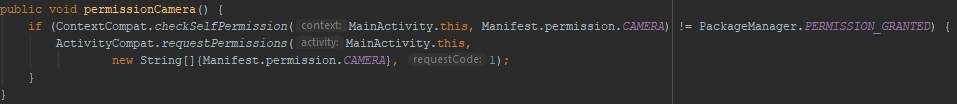
1. **Zapisywanie zdjęcia**

Zapisywanie zdjęcia w pamięci telefonu możliwe jest tylko i wyłącznie w momencie gdy zostało wydane pozwolenie na dostęp do pamięci telefonu przez aplikacje. Zapisywanie zostaje wyzwalane w momencie wykonania zdjęcia, czyli po kliknięciu w ekran lub wydaniu odpowiedniego komunikatu głosowego. Zdjęcia zapisywane są w formacie JPG w folderze, który dostępny jest pod następującą ścieżką w pamięci telefonu */storage/emulated/0/Pictures/openCV/pictureFileName,* gdzie *pictureFileName* to nazwazdjęcia tworzona na podstawie daty i czasu wykonania zdjęcia według poniższego schematu:



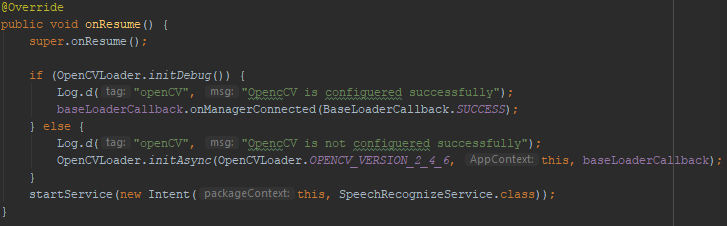
1. **Przykłady kodu**
2. **Uprawnienia dla kamery**

Poniższy fragment kodu odpowiada za sprawdzenie czy aplikacja ma nadane uprawnienia do wykorzystania kamery. W przypadku gdy ich nie ma wyświetla okienko z zapytaniem czy użytkownik wyraża zgodę na wykorzystanie kamery.



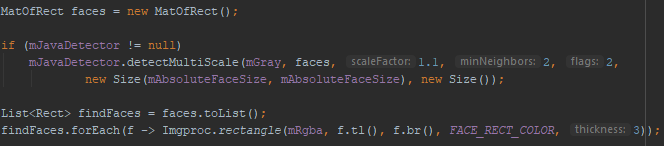
1. **Wznawianie aplikacji**

Poniższy kod odpowiada za wznowienia działania aplikacji. Jest to przeciążona metoda z klasy *Activity*, która odpowiada za aktywności w androidzie. W metodzie najpierw wywołana zostaje ta sama metoda z klasy którą dziedziczymy za pomocą słowa kluczowego *super.* Następnie zostają wczytane biblioteki OpenCV. W zależności czy aplikacja jest w trybie debugowania odbywanie się to w sposób synchroniczny lub asynchroniczny. Wybrana metoda inicjalizacji bibliotek zostaje zapisywana do logów aplikacji na poziomie debug z odpowiednim komunikatem. Na koniec metody zostaje wywołana metoda odpowiadająca za uruchomienie intencji uruchamiająca serwis przetwarzania mowy.



1. **Wykrycie i obrysowanie twarzy**

Kod widoczny poniżej odpowiedzialny jest za wywołanie funkcji detekcji twarzy. Wykorzystywany jest do tego obiekt o nazwie *mJavaDetector*, który jest klasyfikatorem kaskadowym zainicjalizowanym wcześniej. Zwraca on listę prostokątów (zawsze są to kwadraty), która następnie w pętli nanoszona jest na obraz w czasie rzeczywistym .



1. **Podsumowanie**

Podczas projektu powstała aplikacja na urządzenia z systemem android łącząca ze sobą wykorzystanie dwóch interfejsów komunikacji z użytkownikiem czyli przetwarzania obrazu oraz mowy. Ciekawym jej zastosowaniem jest możliwość wykonania zdjęcia za pomocą komunikatu głosowego możliwego do wykorzystania w sytuacji gdy chcemy wykonać zdjęcie w trudnej pozycji lub trudno dostępnym miejscu i nie mamy możliwości naciśnięcia przycisku. Taką funkcjonalność można wykorzystać również jako samowyzwalacz bez konieczności ustawiania czasu po jakim wykona się zdjęcie. Możemy po prostu wydać komunikat głosowy gdy będziemy gotowi. Funkcjonalność rozpoznawania twarzy nie jest niczym nowym. Jest ona dostępna w większości wbudowanych aparatów. Niemniej jednak zdobyte informację podczas tworzenia aplikacji o rozpoznawaniu twarzy można wykorzystać w momencie implementacji aplikacji do rozszerzonej rzeczywistości, która wykona bardziej skomplikowane operacje po wykryciu twarzy.

Biblioteka, która zastosowana do wykrywania twarzy jest bardzo popularna na rynku w przetwarzaniu obrazu. Jednak zastosowane w niej podejście bazujące na kaskadowych klasyfikatorach tworzonych za pomocą przygotowanych wcześniej wzorców wydaję się mało przyszłościowym rozwiązaniem z uwagi na liczne ograniczenia. Znacznie bardziej wykorzystywane będą teraz rozwiązania wykorzystujące uczenie maszynowe i sztuczną inteligencje, które znacznie podnoszą skuteczność rozpoznawania. Z kolei system przetwarzania mowy, którego użyłem wydaję się doskonałym rozwiązaniem do wykorzystania w kolejnych latach z uwagi na stojącego za nim potentata na rynku jakim jest firma Google oferująca obecnie najlepszy, wciąż rozwijany słownik jakim jest tłumacz google. Największym zagrożeniem dla tego systemu jest polityka firmy, która może ograniczyć możliwość bezpłatnego korzystania.

Niedoskonałością, która zauważyłem podczas tworzenia aplikacji, a która może okazać się bardzo cenna w przyszłości jest dużo gorsze działanie biblioteki do wykrywania twarzy na nowych telefonach komórkowych od emulatora działającego niemal bezbłędnie na laptopie. Jako przyczynę takiego stanu uznałem jakość kamery a co za tym idzie rozdzielczość. W telefonach posiadających aparaty rzędu kilkunastu megapikseli obraz nie jest poprawnie konwertowany przez co klasyfikator ma problemu z dopasowaniem go do wzorca twarzy. Na laptopie gdzie jakość wbudowanej kamery nie jest najlepsza problem ten nie występuję.