



POLITECHNIKA KRAKOWSKA IM. TADEUSZA KOŚCIUSZKI

PROJEKT ZREALIZOWANY W RAMACH
INFORMATYCZNEGO OBOZU PRZETRWANIA

„IntelligentEye”

Autorzy:

Mateusz CHRZĄSZCZ
Anna CZARNOTA
Mateusz GAWEL
Paweł GÓRALIK
Bartosz GÓRECKI
Mateusz GRUSZKA
Bartłomiej GRZEBINOĞA
Michał ŁOZA
Łukasz OBSZYŃSKI

Opiekun:

dr inż. Piotr KOWALSKI

2012-09-20

Spis treści

1	Założenia projektu	2
1.1	Technika wykonania	2
1.2	Analiza obrazu	2
2	Wstęp teoretyczny	2
2.1	Język JAVA	2
2.2	Rozpoznawanie obrazu	2
3	Rozpoznanie twarzy	2
3.1	Wybór zestawu cech	3
3.2	Tworzenie klasyfikatora kaskadowego	3
4	Praktyczna realizacja zadania.	4
5	Wnioski i uwagi.	4

1 Założenia projektu

Celem projektu Inteligenteye jest wykonanie programu wykrywającego podejrzone zachowania na podstawie analizy obrazu z kamery wideo.

1.1 Technika wykonania

Projekt będzie stworzony w języku JAVA. Do rozpoznawania obrazu będzie użyta biblioteka OpenCV, opakowana za pomocą JavaCV.

i cos z reszty zespolow

1.2 Analiza obrazu

Obraz otrzymywany z kamery komputerowej będzie analizowany pod kątem przede wszystkim rozpoznawania twarzy. Następnie zostanie przeszukana baza danych zawierająca portrety osób podejrzanych. W momencie wychwycenia podobieństwa program poinformuje użytkownika o niebezpieczeństwie. W miarę możliwości program będzie rozpoznawał również wszelkiego rodzaju podejrzone pakiety i przedmioty a także oznaki wskazujące na nerwowość i zaniepokojenie obserwowanych osób.

2 Wstęp teoretyczny

2.1 Język JAVA

Java jest obiektowym językiem programowania, powstałym w roku 1995. Poprzez standardowe jak i rozszerzone biblioteki wkracza w różnorodne rejony zastosowań takie jak np. karty inteligentne i elektronika, systemy zarządzania bazami danych, obsługa multimedialnych, internet, grafika 3D, kryptografia, itd. Co więcej JAVA jest niespotykanie bezpiecznym środowiskiem i umożliwia w znaczny sposób kontrolę i sterowanie bezpieczeństwem. Zdecydowanie różni się od innych języków trzeciej generacji tym, że jest językiem interpretowanym a nie kompilowanym. Oznacza to, że powstały w wyniku kompilacji kod wynikowy nie jest programem jaki można niezależnie uruchomić lecz stanowi tzw. Beta-kod, który jest interpretowany przez Maszynę Wirtualną (JavaVM) pracującą w określonym środowisku. Ze względu na kod nie istotne jest na jakim sprzęcie będzie uruchamiana aplikacja. Ważna jest tylko Maszyna Wirtualna. Jest to niezwykle ciekawy pomysł umożliwiający odcięcie się od wszystkich poziomów sprzętowo-programowych będących poniżej Maszyny Wirtualnej. Koncepcja ta jest powszechna również w samym języku JAVA, dzięki czemu poprzez stworzenie abstrakcyjnych klas i metod podstawowe biblioteki Javy nie muszą być nieustannie rozbudowywane.

2.2 Rozpoznawanie obrazu

Elementami składowymi kompletnego rozpoznawania obrazów są trzy poziomy: niskiego, średniego i wysokiego poziomu. Przetwarzanie niskiego poziomu obejmuje odbiór obrazu, przetwarzanie wstępne, oraz poprawę jakości obrazu (jak np. eliminacja zakłóceń, zmiana kontrastu czy filtracja). Przetwarzanie średniego poziomu polega na segmentacji i wydzielaniu obiektów obrazu. Wysoki poziom odpowiada za klasyfikację, rozpoznanie i interpretację analizowanej sceny.

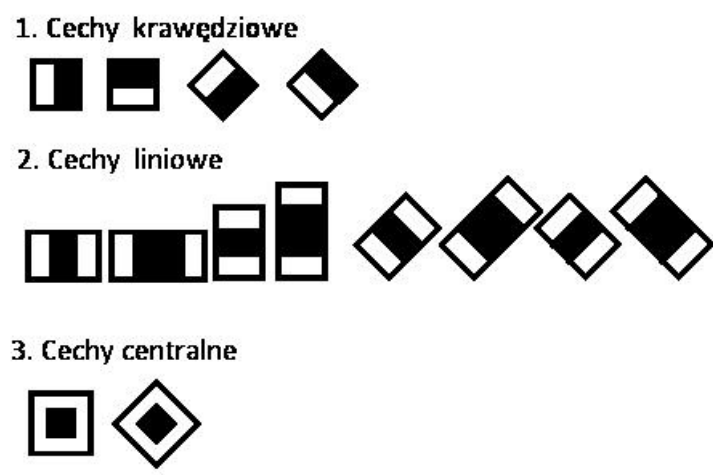
3 Rozpoznanie twarzy

Rozpoznanie twarzy zostało zrealizowane za pomocą otwartej biblioteki OpenCV opakowanej przez JavaCV w celu umożliwienia jej użycia przez język Java. Udostępnia ona szereg narzędzi

ułatwiających rozpoznanie obrazu i jego dalszą obróbkę. Aby algorytmy poprawnie rozpoznawały twarze, został wygenerowany plik posiadający listę cech charakterystycznych dla twarzy. Generacja pliku polega na podaniu zestawu zdjęć "pozytywnych", czyli zawierających tylko twarze z frontu i profilu, zdjęć "negatywnych" ukazujących jedynie tła i środowisko w których twarze nie występują, oraz zwykłych zdjęć ludzi w zwykłym otoczeniu. Aby rozpoznać cechy typowe dla twarzy stosowane są algorytmy Haara i Viola-Jones'a (rozwiniętym przez Rainera Lienhart'a oraz Johena Maydt'a).

3.1 Wybór zestawu cech

W metodzie Viola-Jenes cechy traktowane są jako kombinacja dwóch, trzech lub czterech prostokątów. Ilość możliwych kombinacji połączenia prostokątów była bardzo duża, dlatego przy określonych założeniach została wyselekcjonowana grupa podstawowych cech. Następnie udoskonalony zestaw cech jeszcze lepiej identyfikujący poszukiwane obiekty został zaproponowany w metodzie Lienhart-Maydt. Grupa ta została podzielona na cechy krawędziowe, liniowe oraz centralnie otoczone. Właśnie z tego zestawu cech korzysta funkcja z OpenCV. Zasada działania cech polega na odnajdywaniu obszarów, na których różnica pomiędzy sumą pikseli regionów ograniczonych czarnymi i białymi prostokątami znajduje się powyżej pewnego progu. Na podstawie tej wartości algorytm może zidentyfikować czy badany obszar jest poszukiwanym obiektem czy jego tłem.



Rysunek 1: Zestaw cech zaproponowany przez Lienhart'a i Maydt'a

Dla przykładu, prostokątne regiony oczu mają dużo mniejszą intensywność niż obszar czoła, ten zaś dużo większą od obszaru włosów

3.2 Tworzenie klasyfikatora kaskadowego

Odbyna się z wykorzystaniem algorytmu uczącego AdBoost. Wykorzystuje on tzw. wzmocnienie adaptacyjne (ang. adaptive boosting). Jego zadaniem jest wygenerowanie silnego klasyfikatora z kaskady słabych. W przypadku obrazów, na których wykryty ma zostać obiekt wykorzystuje się technikę przesuwne okna. Z tak zeskanowanego obrazu generowany jest zbiór okien, które poddawane są weryfikacji przez klasyfikatory. Na podstawie informacji z tych okien słabe klasyfikatory uczą się odrzucać obszary, na których na pewno nie znajduje się poszukiwany obiekt. W każdej pętli algorytm skupia się na źle wykrytych obszarach przez poprzednie klasyfikatory. Nadaje im wyższe wagi (gdyż w następnej iteracji istnieje większe prawdopodobieństwo prawidłowego wykrycia obiektu). Wykorzystywanie tych wag przez następne klasyfikatory pozwala na zawężenie przestrzeni poszukiwań do coraz mniejszego obszaru. Tak skonstruowany silny klasyfikator umożliwia szybkie wykrywanie poszukiwanych obiektów.

4 Praktyczna realizacja zadania.

opis

5 Wnioski i uwagi.

wnioski.

Spis rysunków

1	Cechy rozpoznawane przez OpenCV	3
---	---	---