Politechnika Warszawska

WYDZIAŁ ELEKTRYCZNY

Instytut Elektrotechniki Teoretycznej
i Systemów Informacyjno-Pomiarowych
Zakład Elektrotechniki Teoretycznej
i Informatyki Stosowanej

Praca dyplomowa inżynierska

na kierunku Informatyka
w specjalności Inżynieria oprogramowania

Rozpoznawanie gestów na przykładzie gry papier, kamień, nożyce

Adrian Szewczyk
nr albumu 279074

promotor
mgr inż. Marek Wdowiak

WARSZAWA 2018

Rozpoznawanie gestów na przykładzie gry papier, kamień, nożyce

Streszczenie

W pracy zostało przedstawione podejście do problemu rozpoznawania gestów przez wszystkie jego etapy. Na początku zostało pokazane zabranie i przetwarzanie danych w celu uzyskania jak najlepszego zbioru uczącego jak i testowego. Potem zostały opisane etapy filtracji, segmentacji dłoni. Po uzyskaniu danych treningowych zostały stworzone klasyfikatory. Pierwszym podejściem było stworzenie klasyfikatora KNN, a następnie stworzenie klasyfikatora bazującego na sieciach neuronowych. Końcowym etapy pracy to porównanie wyników uzyskanych przez wymienione klasyfikatory na przykładzie gry papier, kamień, nożyce.

Słowa kluczowe: rozpoznawanie gestów, klasyfikacja, TensorFlow, sieci neuronowe

ENG Rozpoznawanie gestów na przykładzie gry papier, kamień, nożyce

Abstract

Keywords: ENG rozpoznawanie gestów, klasyfikacja, python, sieci neuronowe

WARSZAWA, 1 lutego 2018

POLITECHNIKA WARSZAWSKA
WYDZIAŁ ELEKTRYCZNY

OŚWIADCZENIE

Świadom odpowiedzialności prawnej oświadczam, że niniejsza praca dyplomowa inżynierska pt. Rozpoznawanie gestów na przykładzie gry papier, kamień, nożyce:

- została napisana przeze mnie samodzielnie,
- nie narusza niczyich praw autorskich,
- nie zawiera treści uzyskanych w sposób niezgodny z obowiązującymi przepisami.

Oświadczam, że przedłożona do obrony praca dyplomowa nie była wcześniej podstawą postępowania związanego z uzyskaniem dyplomu lub tytułu zawodowego w uczelni wyższej. Jestem świadom, że praca zawiera również rezultaty stanowiące własności intelektualne Politechniki Warszawskiej, które nie mogą być udostępniane innym osobom i instytucjom bez zgody Władz Wydziału Elektrycznego.

Oświadczam ponadto, że niniejsza wersja pracy jest identyczna z załączoną wersją elektroniczną.

Adrian Szewczyk.....

Spis treści

Rozdział 1

Wstęp

1.1 Wprowadzenie

Gesty są jednym z najważniejszych elementów komunikacja pomiędzy ludźmi. Niewerbowana mowa może przekazać zdecydowanie więcej informacji niż tylko słowa wypowiedziane przez nadawcę. Gesty mogę, również zastąpić słowa i być wykorzystywane jako zamiennik na słyszalne słowa. Przykładem tego jest język migowy, który pozwala na wzajemną komunikację ludzi głuchych . Oprócz samej mowy gesty, mają też inne zastosowania. Przykładem takim może być gra papier, kamień, nożyce. W której każdy z graczy pokazuje jeden z trzech dostępnym gestów. I w zależności od kombinacji gestów pokazanych przez graczy określany jest zwycięzca.

Klasyfikacja gestów jest powszechnym problem przetwarzania obrazów. Jest to temat obszerny i coraz lepiej zbadanym problem nauki. Klasyfikacja gestów ma szerokie zastosowanie. Jedną zalet rozpoznawania gestów jest możliwość sterowania naszym smartfonem za pomocą ruchu rąk. Po wykonaniu gestu a następnie po rozpoznaniu go przez oprogramowanie telefonu wykonywana jest określona akcja. Innym wykorzystaniem klasyfikacji gestów jest sterowanie gier, podając jako przykład konsolę Nintendo Wii.

1.2 Cel pracy

W swojej pracy dyplomowej będę chciał przedstawić sposób podejścia do problemu klasyfikacji gestów, a następnie zaimplementować działające rozwiązanie i wykorzystać go w praktyce, na przykładzie gry papier, kamień, nożyce. Stworzona gra pozwoli na automatyczne rozpoznawanie gestów, liczenie punktów i byciem arbitrem w grze. Wszystkim tym zajmie się specjalnie napisany system, który będę chciał zaprezentować w tej pracy dyplomowej.

Rozdział 2

Przetwarzanie obrazów

2.1 Cyfrowe przetwarzanie obrazów binarnych

Cyfrowe przetwarzanie obrazów binarnych jest to dziedzina przetwarzania obrazów cyfrowych zajmująca się algorytmami obróbki obrazów binarnych.

Rysunek 2.1: **Przykładowy obraz binarny**

Obrazy binarne składają się z pikseli, które mogą przyjmować jedynie dwie wartości. Piksele takie możmy oznaczać różnymi symboli, np. 0/1, false/true czy też (0,0,0)/(255,255,255). Podczas interpretacji obrazu jedne z pikseli można traktować jako tło, inne zaś jako część obiektu. Dzięki czemu jesteśmy wstanie rozróżnić interesującą nas część obrazu od pozostałej części.

2.2 OpenCV

OpenCV (Open Source Computer Vision Library) jest to otwartoźródłowa biblioteka wykorzystywana przy rozpoznawaniu obrazów oraz przy uczeniu maszynowym. Została napisana w języku C przez programistów z firmy Intel w 1999 roku.

W późniejszym czasie kolejne części biblioteki były także pisane w języku C++. OpenCV umożliwia pisanie kodów nie tylko w językach C i C++ ale również mamy możliwość wykorzystywania tej biblioteki w językach Python, Java czy też Matlab. Co więcej zostały udostępnione nakładki do języków takich jak C#, Perl, Ruby czy Haskell aby móc również wykorzystywać zalety biblioteki.

OpenCV udostępnia zaawansowaną funkcjonalność w szeroko rozumianym pojęciu rozpoznawaniu obrazów:

- przetwarzania obrazów
- klasyfikowaniu wzorców
- dokładnych pomiarów obrazów

Jako główne zalety OpenCV możemy wyróżnić, że jest on darmowy oraz otwartoźródłowy. Zawiera bogaty zakres funkcjonalności, a kod biblioteki został napisany w sposób zoptymalizowanym, tak aby operacje wymagające dużej mocy obliczeniowej czy działające w czasie rzeczywistym mogły wykonywać się możliwie jak najszybciej.

OpenCV znalazło zastosowanie w wielu dziedzinach naszego codziennego życia:

- medycyna
- robotyka
- samochody autonomiczne
- systemy antywłamaniowe
- systemy zabezpieczające
- rozpoznawanie gestów
- segmentacja obiektów
- wykrywanie ruchu
- rozszerzona rzeczywistość
- rozpoznawanie obiektów

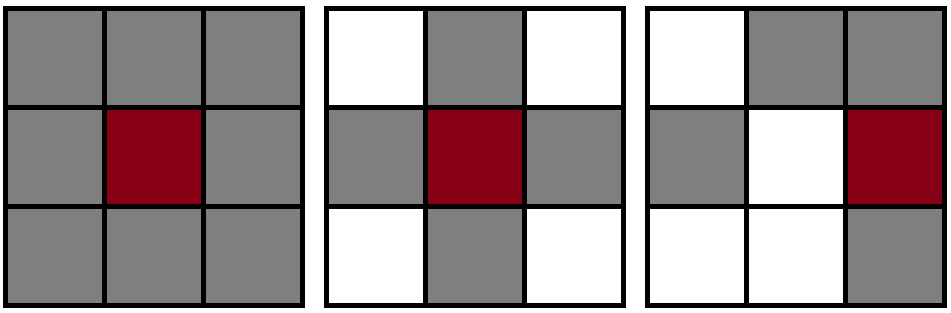
2.3 Operacje morfologiczne

Operacje morfologiczne to podstawowe operacje przetwarzania obrazów. Pozwalają na złożone czynności związane z analizą kształtu poszczególnych

elementów obrazu oraz położeniem względem siebie. W wyniku operacji struktura obiektu na obrazie zostaje zmieniona, w celu osiągnięcia określonych rezultatów. Operacje morfologiczne najczęściej stosuje się dla obrazów binarnych, dla których są to operacji podstawowe oraz jedne z najważniejszych. Dzięki nim jesteśmy wstanie wyszczególnić interesujące nas części czy też przefiltrować nasz obraz. Najczęściej wykorzystywane operacje morfologiczne to: erozja, dylatacja, otwarcie oraz zamknięcie. Wymienione operacje można ze sobą łączyć, tworząc zaawansowane systemy analizy.

Operacje morfologiczne modyfikują wartość pikseli biorąc pod uwagę wartości pikseli ich otaczających. Liczbę punktów otoczenia określa tzw. element strukturalny, który definiuje wartości i ich rozmieszenie w otoczeniu. Szablon strukturalny(element strukturalny) posiada jeden wyróżniony punkt, nazywany punktem centralnym.

Najczęściej stosowanymi elementami strukturalnymi są kwadrat o boku o nieparzystej liczbie pikseli, które wszystkie przyjmują wartość równą 1 oraz element strukturalny, który aproksymuje swoim kształtem koło.



(a) (b) (c)

Rysunek 2.2: Przykładowe elementy strukturalne, gdzie punkt kolorze czerwonym jest punktem centralnym

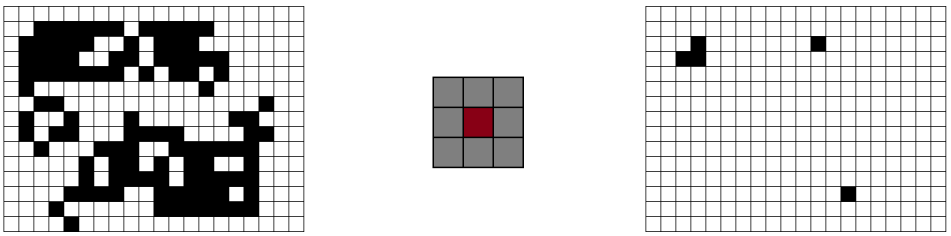
Etapy przekształceń morfologicznych:

1. Szablon strukturalny jest przesuwany po całym obiekcie, tak aby punkt centralny szablonu był analizowanym pikselem
2. Następuje porównanie otoczenia analizowanego piksela z elementem strukturalnym
3. W zależności od stosowanej operacji morfologicznej wartość analizowanego piksela zmienia się lub też pozostaje bez zmian

Erozja:
Jest to operacja zwężania i zmniejszania poprzez usunięcie pikseli granicznych. Usuwa wszystkie mniejsze obiekty, które możemy zinterpretować jako szumy.

Etapy operacji erozji:

1. Element strukturalny przesuwany jest iteracyjnie po całym obiekcie, tak aby punkt centralny szablonu był analizowanym pikselem
2. Porównuje się otoczenie analizowanego piksela z elementem strukturalnym
3. Jeżeli przynajmniej jeden piksel z otoczenia objętego przez szablon strukturalny ma wartość równą 0, to punkt centralny przyjmuje wartość 0. Jeżeli zaś taki przypadek nie wystąpił piksel centralny zachowuje swoją poprzednią wartość



(a) Obraz początkowy (b) Element struturalny, punkt czerwony jest punktem centralnym (c) Obraz wyjściowy

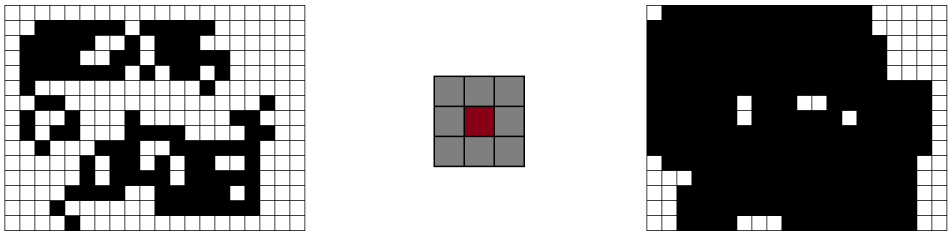
Rysunek 2.3: Operacja erozji na obrazie binarnym

Dylatacja:
Jest to operacja rozszerzania i zwiększania. Pozwala na wypełnienie dziur binarnych w obiektach. Jeżeli dwa obiekty są położone blisko siebie, może nastąpić złączenie w jeden obiekt.

Etapy operacji erozji:

1. Element strukturalny przesuwany jest iteracyjnie po całym obiekcie, tak aby punkt centralny szablonu był analizowanym pikselem
2. Porównuje się otoczenie analizowanego piksela z elementem strukturalnym

3. Jeżeli przynajmniej jeden piksel z otoczenia objętego przez szablon strukturalny ma wartość równą 1, to punkt centralny przyjmuje również wartość 1. Jeżeli zaś wszystkie piksele z otoczenia piksela centralnego określonego przez element strukturalny mają wartość 0 to wówczas piksel centralny przyjmuje wartość 0.



(a) Obraz początkowy (b) Element struturalny, punkt czerwony jest punktem centralnym (c) Obraz wyjściowy

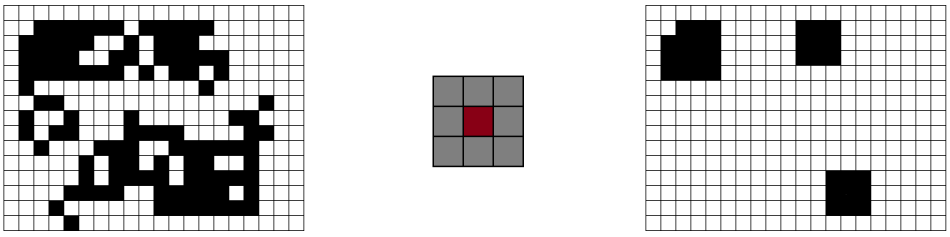
Rysunek 2.4: Operacja dylatacji na obrazie binarnym

Otwarcie:
Operacja otwarcia morfologicznego jest definiowana przez połączenie metod erozji oraz dylatacji. Metoda ta wygładza obiekt, usuwa niechciane szumy, a w dodatku nie modyfikuje znacząco wielkości obiektów tak jak to jest w przypadku zastosowania operacji erozji czy dylatacji.

Etapy operacji otwarcia:

1. Wykonywana jest operacja erozji na zadanym obrazie

2. Wykonywana jest operacja dylatacji na obraz, który jest wynikiem erozji z punktu 1.



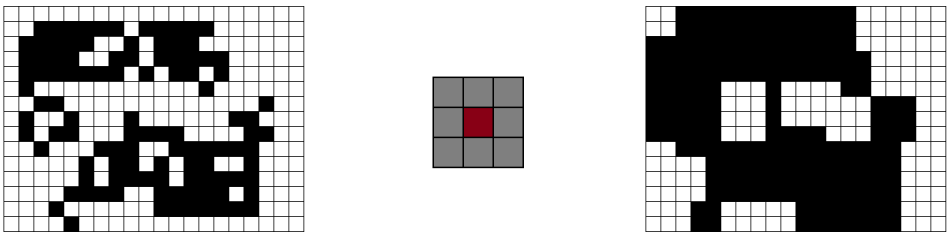
(a) Obraz początkowy (b) Element struturalny, punkt czerwony jest punktem centralnym (c) Obraz wyjściowy

Rysunek 2.5: Operacja otwarcia na obrazie binarnym

Zamknięcie:
Operacja zamknięcia łączy w sobie połączenie dwóch metod dylatacji oraz erozji, analogicznie jak było w przypadku operacji zamknięcia, jednakże w odwrotnej kolejności. W rezultacie uzyskujemy połączenie obiektów o zbliżonych odległościach, wypełnienie dziur w obiektach, jednakże końcowy kształt obiektu zostaje w dużym stopniu zmieniony w stosunku do kształtu przed zastosowaniem operacji zamknięcia.

Etapy operacji otwarcia:

1. Wykonywana jest operacja dylatacji na zadanym obrazie
2. Wykonywana jest operacja erozji na obraz, który jest wynikiem erozji z punktu 1.



(a) Obraz początkowy (b) Element struturalny, punkt czerwony jest punktem centralnym (c) Obraz wyjściowy

Rysunek 2.6: Operacja zamknięcia na obrazie binarnym

2.4 Filtry ruchu

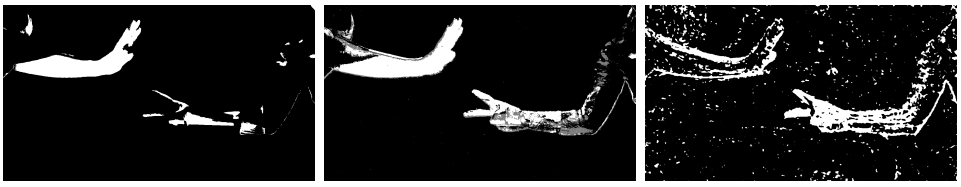
Filtry ruchu są wykorzystywane do generowania maski binarnej zawierającej poruszające się obiekty. Tak zwany z ang. background subtraction wykorzystuje różnicę z dwóch różnych klatek z pliku wideo. Odejmowanie dwóch klatek pozwala nam prześledzić, które z części obrazów zmieniły się. Różnica dwóch pikseli da nam wartość niezerową, tylko wtedy jeżeli z analizowanych klatkach wartość odpowiadających pikseli różni się, co interpretujemy, że ta część obrazu poruszyła się.

(a) Tło (b) Zdjęcie (c) Różnica zdjęcia i tła

Rysunek 2.7: Przykładowe metody background subtractor z biblioteki OpenCV

OpenCV w swojej bibliotece posiada wiele gotowych rozwiązać do wykrywania obiektów w ruchu. Jako jedne z najczęściej wykorzystywanych metod możemy wyróżnić następujące:

- BackgroundSubtractorMOG
- BackgroundSubtractorMOG2
- BackgroundSubtractorGMG



(a) BackgroundSubtractorMOG (b) BackgroundSubtractorMOG2 (c) BackgroundSubtractorGMG

Rysunek 2.8: Przykładowe metody background subtractor z biblioteki OpenCV

W celu polepszenia efektów, należy pozbyć się niedokładnej segmentacji oraz możliwych szumów, które mogą wystąpić podczas analizy. Są one

spowodowane różnymi odbiciami, cieniami czy innymi niezauważalnymi ruchami. Wówczas należy wykorzystać przedstawione wcześniej operacje morfologiczne, które poprawią nam wykonaną detekcję.

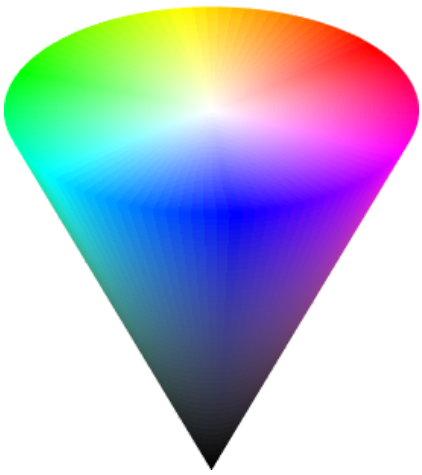
2.5 Detekcja kolorów

Celem detekcji kolorów jest wygenerowanie obrazu binarnego, w którym piksele binarne o wartości prawdy oznaczają piksele, które na zadanym obrazie są w określonym przedziale kolorów. Jeżeli dany piksel nie należy do określonego przedziału, wówczas obraz binarny w tym punkcie otrzymuje wartość fałszu. Detekcja taka pozwala na wyłuskania tylko obiektów o określonym koloru.

Detekcja jest bardzo wrażliwa na różnego rodzaju szumy. Częstymi powodem występowanie ich jest zmienne oświetlenie. Obiekt przy różnym świetle możemy odbierać kolorystycznie całkowicie inaczej. W przypadku, kiedy analizujemy dynamiczny ruch, wówczas dynamika również może wpływać na kolorystykę danego obiektu. Te wszystkie czynniki, sprawiają, że detekcja określonego koloru nie jest zadaniem takim prostym.

Aby polepszyć efekt końcowy detekcji obiektów o określonym kolorze, należy analogicznie jak w przypadku detekcji ruchu, wykorzystać odpowiednie operacje morfologiczne.

Przy detekcji kolorów tradycyjny model rgb (red, green, blue), najczęściej nie jest wystarczająco dobrym modelem. W zadaniach takich zdecydowanie lepiej sprawuje się model HSV.



Rysunek 2.9: **Model HSV**

HSV (ang. Hue Saturation Value) jest to model opisu przestrzeni barw, wprowadzony przez Alveya Raya Smitha w 1978 roku.

Model składa się z następujących części: barwa, nasycenie, wartość. Geometrycznie możemy go przedstawić jako stożek. Wszystkie barwy wywodzą się ze światła białego, czyli ze środka stożka. Składowa H, oznacza barwę w postaci kąta od 0 do 360 stopni. Składowa S czyli nasycenie, geometrycznie to odległość od środka na promieniu podstawy. Decyduje o bliskości naszego koloru do koloru białego. Ostatnia składowa V, to wartość oznaczająca wysokość na stożku, co możemy traktować jako jasność naszego koloru. Czym mniejsza jej wartość tym kolor ciemniejszy.

Rozdział 3

Sztuczna inteligencja

3.1 Sztuczna inteligencja, uczenie maszynowe, głębokie uczenie

3.2 Sieci neuronowe

3.2.1 Uczenie sieci neuronowych

3.3 Klasyfikatory

Klasyfikacja to wyznaczanie klasy decyzyjnej do której należy nowy, nieznany dotąd obiekt. Metody klasyfikacji możemy podzielić na dwie kategorie. Pierwsza z nich jest klasyfikacją nadzorowaną, druga zaś to klasyfikacja nienadzorowana.

Klasyfikacja nadzorowana w momencie uczenia klasyfikatora tzn. podawania zbioru danych treningowych, musi w sobie zawierać etykietę(atrybut decyzyjny), która oznacza do jakiej klasy należy ten konkretny przypadek. Zaś w przypadku klasyfikacji nienadzorowanej atrybutu decyzyjnego nie istnieje.

3.3.1 Klasyfikator KNN

Klasyfikator KNN jest przykładem klasyfikacji nadzorowanej. Zbiór treningowy zawiera w sobie cechy dla klasyfikatora czyli atrybuty oraz atrybut decyzyjny, który określa przynależność do konkretnej klasy. Elementami wejścia dla stworzenia klasyfikatora jest zbiór uczący. Wyjściem jest klasyfikator, który wykorzystujemy do określania atrybutów decyzyjnych nieznanych wcześniej obiektów.

Po utworzeniu klasyfikatora bazującego na zbiorze uczącym, szacowanie atrybutu decyzyjnego odbywa się w następujących krokach:

1. Ustalamy wartość k
2. Znajdujemy k obiektów treningowych, najbliższych naszemu obiektowi
3. Analizowany obiekt należy do klasy najliczniejszej w znalezionym zbiorze z podpunktu 2

Ocenianie odległości pomiędzy dwoma obiektami polega na umieszczaniu obiektów w przestrzeni d-wymiarowej, gdzie d opisuje ilość atrybutów dla obiektów. Metryka miary może odbywać się w różny sposób. Najczęściej jest to miara euklidesowa, jednakże możemy również wykorzystać inne miary tj. Manhattana czy Minkowskiego.

Po znalezieniu k-najbliższych sąsiadów, atrybut decyzyjny przyjmuje wartość od najbardziej licznej klasy w wyznaczonym zbiorze.

Rysunek:

3.3.2 Dobór i ocena cech

3.4 Python

Jest to interpretowany, obiektowy język wysokiego poziomu. Używany jest w szerokiej gamie aplikacjach. Jest językiem ogólnego przeznaczenia, co oznacza, że może zostać wykorzystany praktycznie do wszystkiego. Jest rozwijany jako projekt otwartoźródłowy. Pojawił się w roku 1991, zaprojektowany przez holenderskiego programistę Guido van Rossum.

Zalety używania Pythona:

- prosta, czytelna, klarowna składnia, zmniejszająca ilość potrzebnych linijek kodu w programach
- dynamicznie zarządza pamięcią oraz typy danych
- nie wymusza stylu programowania
- jest wieloplatformowy
- posiada bogaty zbiór różnego rodzaju bibliotek
- łatwy do nauczenia, nawet dla osób zaczynających programować
- szybkość działania w stosunku do innych języków kryptowych

Jeśli mielibyśmy dyskutować o wadach językach Python to ciężko jednoznacznie wskazać takie. Na pewno część z programistów może uznać zalety dynamicznego zarządzania typami jako wadę, ponieważ w niektórych przypadkach

może spowodować błędy trudniejsze do znalezienia. Działania Pythona jest również wolniejsze w stosunku do języków takich jak C czy C++. Za jedną z wad na pewno można uznać sposób programowania obiektowego jakie odbywa się w Pythonie. W szczególności nie ma enkapsulacji, istnieją metody które symulują takie działa, jednakże w stosunku do innych języków obiektowych czytelność kodu zdecydowanie jest zmniejszona.

Język mimo swoich lat ciągle zyskuje na popularności. Ostatnio wszedł do pierwszej trójki najbardziej popularnych języków programowania według TIOBE (stan na grudzień 2018), wyprzedzając między innymi język C++, a ulegając jedynie językowi Java oraz C. Wzrost języka Python na pewno można łączyć ze wzrostem popularności uczenia maszynowego i głębokich sieci, gdzie język Python jest jednym z najlepszych, jak nie najlepszym językiem w tych dziedzinach. Bardzo bogata biblioteka oraz przyjemna składnie sprawia, że język Python jest częściej wykorzystywany.

Jako ciekawostkę można powiedzieć, że nazwa języka Python wzięła się nie od zwierzęcia, lecz od słynnego Brytyjskiego serialu "Monty Python's Flying Circus".

3.5 TensorFlow

Jest to biblioteka programistyczna wykorzystywana w uczeniu maszynowym i głębokich sieciach neuronowych. Została wydana jako otwarte oprogramowanie przez Google Brain Team w dniu 9 listopada 2015.

Umożliwia pisanie programów m.in. w językach takich jak Python czy C. Jest dostępny na 64-bitowych systemach operacyjnych: Windows, Linux, macOS oraz na platformach mobilnych: Android oraz iOS. Zaś w maju 2017 został wydany TensorFlow Lite jako dedykowane rozwiązanie specjalnie dla użytkowników Androida.

Olbrzymią zaletą TensorFlow jest to, że reprezentuje on paradygmat Dataflow, w którym program ma postać grafu skierowanego modelującego przepływ danych pomiędzy niezależnymi operacjami w węzłach. W pierwszym kroku należy zdefiniować model, a następnie stworzyć tzw. TensorFlow session, która pozwala na uruchomienie programu. Takie podejście do pisania programów posiada ogromną zaletą jaką jest możliwość efektywnego programowania równoległego oraz rozproszonego.

TensorFlow daje możliwość nie tylko wykorzystania CPU, ale równie dobrze korzystania z GPU. Wszystko to powoduje pełne wykorzystanie mocy obliczeniowej komputerów przy obliczeniach równoległych.

TensorFlow oprócz niskopoziomowych struktur posiada również moduły wyższego poziomu. Do modułów niskiego poziomu możemy odwoływać się

poprzez warstwę API, która umożliwia łatwy do używania interfejs przeznaczony do modelów głębokiego uczenia. Zaś nad warstwą API znajduje się warstwa wysokiego poziomu, przykładem jej jest Keras.

3.6 Keras

Otwartoźródłowa biblioteka programistyczna napisana w języku Python wydana w dniu 27 marca 2015 napisana przez pierwotnego autora François Chollet.

Jej głównym przeznaczeniem jest to, aby w jak najprostszy i jak najszybszy sposób umożliwić pracę w głębokich sieciach neuronowych. Co więcej, biblioteka została zaprojektowana w sposób przyjemny do użytkowania, skupiona na modularności oraz rozszerzalności. Zaś od 2017 roku TensorFlow wspiera w swojej bibliotece Kerasa. Dzięki czemu Keras jest wysokopoziomową warstwą tworzenia modelów sieci neuronowych i jej uczenia.

Rozdział 4

Gra papier, kamień, nożyce

Jest to klasyczna gra towarzyska, w tradycyjnym wariancie przeznaczona dla dwóch graczy. W każdej rundzie każdy z graczy pokazuje gest przez siebie wybrany z trzech dostępnych. Wszystkie rundy odbywają się na ustalony wcześniej sygnał, a pokazywane gesty powinny być jak najbardziej zsynchronizowane. Gra kończy się wtedy, jeżeli jeden z graczy osiągnie określoną wcześniej ilość zwycięstw.

Trzy gesty dozwolone w grze:

- Papier
- Kamień
- Nożyce

(a) Gest papieru (b) Gest kamienia (c) Gest nożyc

Rysunek 4.1: BackgroundSubtractorKNN na gestach z gry papier, kamier, nożyce

Zwycięzca określany jest w zależności od kombinacji gestów pokazanych przez graczy. Jeżeli oba gracze pokazali ten sam gest, wówczas następuje remis.

W przypadku kombinacji:

- papier-kamień - zwycięzcą zostaje gracz, który pokazał gest papieru

- papier-nożyce - zwycięzcą zostaje gracz, który pokazał gest nożyc
- kamień-nożyce - zwycięzcą zostaje gracz, który pokazał gest kamienia

Zachowując takie zasady, każdy z gestów ma takie samo prawdopodobieństwo zwycięstwa. W grze liczy się w dużej mierze szczęście, ale także umiejętność przewidywania gestów przeciwnika.

Rozdział 5

Rozpoznawanie gestów na przykładzie gry papier, kamień, nożyce

5.1 Aktywizacja danych

Pierwszym etapem mojej pracy była aktywizacja danych potrzebnych do uczenia maszynowego. Zbieranie danych polegało na nagrywaniu plików wideo na których dana osoba wykonywała określony gest. W swojej pracy dyplomowej zakładam, że każdy wykonywany gest, jest pokazywany w niebieskiej rękawiczce. To założenie będę wykorzystywał w całej swojej pracy inżynierskiej.

Rysunek 5.1: Rękawiczki wykorzystywane w pracy inżynierskiej

Osoba podczas nagrywania wykonuje ciąg określonych gestów, w dowolnym odstępie czasie. Jest ona ustawiona prostopadle do kamery, co pozwala, aby gest odbywał się poprzez wyciągnięcie ręki oraz jego pokazanie. Na nagraniu zostaje zarejestrowany wykonywany gest, tak aby jedyną widoczna częścią ciała była prawa dłoń oraz ewentualnie cześć przedramienia.

Rysunek 5.2: Przykładowa klatka z nagrania

Do nagrywania plików wideo używałem dwóch kamer dedykowanych w laptopach:

- USB2.0 UVC HD Webcam
- HD Webcam

Obie kamery pozwoliły na zadowalające efekty zbierania danych. Jakość nagrań przy dostatecznie dobrym oświetleniu była bardzo zadowalające, pozwalająca bez problem rozpoznawać wykonywany gest. Również liczba klatek na sekundę pozwoliła, aby dynamiczny ruch był płynny i łatwy do prawidłowego wysegmentowania.

W celu zebrania jak najszerszego zakres danych, zbieranie gestów nie ograniczyło się do jednej osoby, jednakże do pięciu różnych osób:

- trzech kobiet
- dwóch mężczyzn

Zbieranie danych na różnych osobach pozwala na zmniejszenie wpływu charakterystycznych cech dla danej osoby:

- długości dłoni
- szerokości dłoni
- ułożenia palców
- nachylenia dłoni

Tak zebrane danych dają ogólny pogląd na różnorodność cech danego gestu. Sprawia, że uczenie maszynowe jest skuteczniejsze i daje oczekiwane rezultaty.

Po zgromadzeniu całego materiału wideo, przeszedłem do etapu wyznaczania numerów klatek nagrań na których był moment wykonanego gestu. Informacje o numerach klatek gestów będę wykorzystywał w późniejszym etapie, przy segmentacji dłoni z wykonanym gestem.

W mojej pracy inżynierskiej wykorzystywałem zbiór danych składający się z:

- 350 próbek gestu papieru
- 350 próbek gestu kamienia
- 350 próbek gestu nożyc

Aby zapewnić prawidłowe rozpoznawanie, należy zgromadzić jak największą liczbę próbek gestów, którą będzie w późniejszych etapach wykorzystywana przy uczeniu oraz weryfikacji klasyfikatorów.

5.2 Detekcja ruchu

Po zebraniu danych zająłem się ich przetwarzaniem. Pierwszym krokiem było wysegmentowanie obiektów, które znajdowały się w ruchu. Aby to osiągnąć należało wygenerować maskę binarną, którą rozdziera elementy obrazu, która są w ruchu od tych, które są elementami statycznymi. Do tego wykorzystałem tzw. background subtraction czyli tłumacząc z angielskiego „odejmowanie tła".

W swojej pracy inżynierskiej w analizie wziąłem pod uwagę następujące dostępne funkcje w bibliotece OpenCV:

- BackgroundSubtractorMOG
- BackgroundSubtractorMOG2
- BackgroundSubtractorGMG
- BackgroundSubtractorKNN
- BackgroundSubtractorGSOC
- BackgroundSubtractorLSBP

(a) BackgroundSubtractorKNN (b) BackgroundSubtractorGSOC (c) BackgroundSubtractorLSBP

Rysunek 5.3: Przykładowe obrazy biorące udział w wyborze odpowiedniej funkcji usuwającej tło z biblioteki OpenCV

Po przeproawdzeniu licznych testów wywnioskowałem, że najlepszą metodą przy mojej pracy będzie funkcja z biblioteki OpenCV BackgroundSubtractorKNN. Uważam, że na przetestowanych próbkach segmentacja ruchu metodą tą dawała najlepsze rezultaty w porównaniu z innymi przetestowanymi metodami. Parametry funkcji przyjęły następującą końcową wartość:

1. history: 200
2. dist2Threshold: 8
3. detectShadows: false

(a) Znak papieru (b) Gest kamienia (c) Gest nożyc

Rysunek 5.4: Uzyskane rezultaty metodą BackgroundSubtractorKNN z biblioteki OpenCV

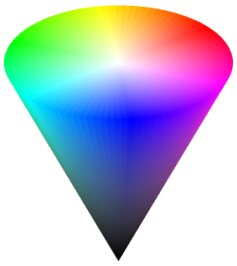
Można zauważyć, że wykonywany ruch dłonią jest bardzo widoczny, jednakże dodatkowo zawarta jest duża ilość szumu. Pozbycie się elementów dodatkowych, nieinteresujących nas części obiektów przedstawię w kolejnych podrozdziałach o detekcji kolorów oraz w podrozdziale o filtracji szumu.

5.3 Detekcja kolorów

Równolegle do segmentacji obiektów w ruchu, zająłem się detekcją obiektów o określonym kolorze. W swojej pracy inżynierskiej założyłem, że każdy

wykonywany gest jest pokazywany w niebieskiej rękawiczce, w celu łatwiejszej segmentacji dłoni. Nie użyłem tradycyjnego koloru skóry ze względu na fakt, że oprócz detekcji dłoni spowodowałby także segmentację innych części ręki, jak również pozostałych części ciała, takich jak twarz czy klatka piersiowa. Wszystko to mogłoby spowodować mniej precyzyjne rozpoznawanie gestu.

Wybrałem kolor niebieski ze względu na to, że jest to kolor dobrze rozróżniający się od innych, na który nie wpływa aż tak znacząco moc oświetlenia. Jako, że każdy gest wykonywany jest w niebieskiej rękawiczce, mogłem wykorzystać ten fakt i przy pomocy modelu HSV określić zakres kątów barwy w tym modelu.



Rysunek 5.5: **Model HSV**

Na powyżej przedstawionym modelu HSV, można łatwo zaobserwować, że kolor niebieski znajduje się w pewnym przedziale składowej H modelu HSV. To pozwala na zdecydowanie łatwiejsze określenie zakresu niż w przypadku modelu RGB. Chciałem tak dobrać przedział kolorów w modelu HSV, aby był możliwie jak najszerszy w celu dobrej detekcji dłoni. Zakres ten zawierał dodatkowo kolory zbliżone do niebieskiego oraz większość z jego odcieni. To pozwoliło na zwiększoną liczbę wykrytych pikseli, a w tym lepiej wysegmentowaną dłoń. Jednakże wszystko to również niesie za sobą różnego rodzaju szumy, które nie są związane z obiektem dłoni.

(a) Znak papieru (b) Gest kamienia (c) Gest nożyc

Rysunek 5.6: Rezultaty detekcji dłoni w niebieskiej rękawiczce

Zakres jaki wykorzystałem w swojej pracy inżynierskiej jest następujący:

1. od HSV (98,50,20)
2. do HSV (130,255,255)

Jak widzimy występuje znacząca liczba szumu, jednakże jeżeli połączymy detekcję koloru niebieskiego z detekcją ruchu, która została opisana w poprzednim podrozdziale, może pozbyć się w większości tych niechcianych elementów.

(a) Znak papieru (b) Gest kamienia (c) Gest nożyc

Rysunek 5.7: Rezultaty detekcji dłoni w niebieskiej rękawiczce oraz wykrywania ruchu

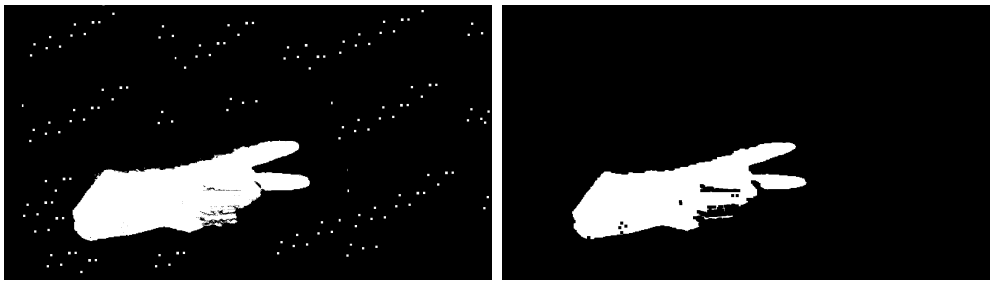
Jak widzimy większość z niepożądanych szumów została usunięta za sprawą połączania dwóch masek binarnych. Możemy też zaobserwować, że nie wszystko zostało prawidłowo przefiltrowane, mogły też pozostać tzw. dziury binarne czyli części obiektów, które nie zostały zakwalifikowane jako części obiektu, a nimi faktyczne są.

Wszystko to powoduje, że konieczne są kolejne filtracje szumów. W tym przypadku wykorzystałem operacje morfologiczne na obrazach binarnych, co opiszę w następnym podrozdziale.

5.4 Filtracja szumów

Połączenie iloczynowe maski binarnej uzyskanej z segmentacji obiektów w ruchu oraz maski binarnej z obiektami o określonym kolorze niesie za sobą możliwe niedokładności, których dla polepszenia efektowności rozpoznawania gestów należy zminimalizować. Odbywa się to poprzez zastosowanie operacji morfologicznych na obrazie binarnym.

Dodatkowe elementy:
Pierwszym problemem po wygenerowaniu maski binarnej można zostać uznane wszystkie dodatkowe elementy, które nie powinny być zakwalifikowane jako część dłoni. Segmentacja elementów obrazu w ruchu, może nieść ze sobą różnego rodzaju szum spowodowany tym, że inne obiekty, różne od naszej dłoni też mogą się poruszać, co w wyniku generacji maski binarnej zostaną też wzięte pod uwagę. Kolejnym problem są także wszystkie te pikseli, które są koloru niebieskiego lub też są zbliżone do niego, a więc przy wyłuskiwaniu wszystkich pikseli o kolorze rękawiczki, czyli w kolorze niebieskim, zostaną uwzględnione jako obiekt.



(a) Maska binarna przed operacją morfologiczną erozji

(b) Maska binarna po operacji morfologicznej erozji

Rysunek 5.8: **Zastosowanie operacji morfologicznej erozji z elementem strukturalnym kwadratu o boku 8 pikseli**

Zastosowanie operacji morfologicznej erozji na masce binarnej powoduje usunięcie elementów niechcianych, które są traktowane jako dodatkowy, niepotrzebny szum. W swojej pracy inżynierskiej wykorzystałem w pierwszym kroku filtracji masek binarnych operację morfologiczną erozji z bardzo małym elementem strukturalnym, bo z kwadratem o boku 2. Różniejsza operacja erozji, po wykonaniu operacji zamknięcia odbywała się już z większym wielkością elementem strukturalnym: kwadratem o boku 8. Wykorzystanie takiej operacji dało zadowalające efekty w mojej ocenie, które pozwolą w późniejszym etapie pracy na dokładniejsze rozpoznawanie gestów.

Ubytku w dłoni:
Oprócz pozbycia się dodatkowych elementów na obrazie binarnym należy także, wypełnić luki które mogły się pojawić podczas segmentacji obiektów w ruchu oraz przy uzyskiwaniu maski binarnej z obiektami o określonym kolorze. Są one spowodowane tym, że pewne części z obrazów podczas tych operacji zostały prawidło zakwalifikowane. Zbyt mocne lub też zbyt słabe

światło czy też pojawiające się cienie mogły wpłynąć na kolor niebieskiej rękawiczki, a przez to niedokładnie ją wykryć. Również gest ręki mógł być bardzo powolny, a przez to nie zostanie uznana pełna dłoń jako element w ruchu przez operacje segmentacji ruchu. Zarówno reż dynamiczny ruch ręku w połączeniu z występujących w otoczeniu światłem wpływa na odbierany kolor rękawiczki. Należy też dodatkowo wziąć pod uwagę dziury w obrazie binarnym, który mogły wyniknąć z operacji morfologicznej erozji, która została zastosowana jako pierwsza.

(a) Maska binarna przed operacją morfologiczną dylatacji

(b) Maska binarna po operacji morfologicznej dylatacji

Rysunek 5.9: **Zastosowanie operacji morfologicznej dylatacji z elementem strukturalnym kwadratu o boku 11 pikseli**

Zastosowanie operacji morfologicznej dylatacji na masce binarnej powoduje wypełnienie dziur, które mogły wystąpić w masce binarnej. W swojej pracy inżynierskiej do pozbycia się różnego rodzaju dziur w maskach binarnych użyłem operacji morfologicznej dylatacji z elementem strukturalnym kwadratu o boku 11.

(a) Gest kamienia przed filtracją

(b) Gest kamienia po filtracji

Rysunek 5.10: **Przykładowe maski binarne po zastosowaniu operacji morfologicznych erozji oraz dylatacji z elementami strukturalnymi kwadratu o boku 8 dla erozji oraz kwadratu o boku 11 dla operacji dylatacji**

W swojej pracy inżynierskiej zastosowałem następujące operacje do przefiltrowania masek binarnych:

1. operację morfologiczną erozji na masce binarnej z elementem strukturalnym kwadratu o boku 4
2. operację morfologiczną zamknięcia z elementem strukturalnym kwadratu o boku 8 na wyjściowej masce binarnej uzyskanej w podpunkcie nr 1
3. operację morfologiczną dylatacji z elementem strukturalnym kwadratu o boku 11 na wyjściowej masce binarnej uzyskanej w podpunkcie nr 2
4. operację morfologiczną dylatacji z elementem strukturalnym kwadratu o boku 11 na wyjściowej masce binarnej uzyskanej w podpunkcie nr 3

(a) Gest papieru (b) Gest kamienia (c) Gest nożyc

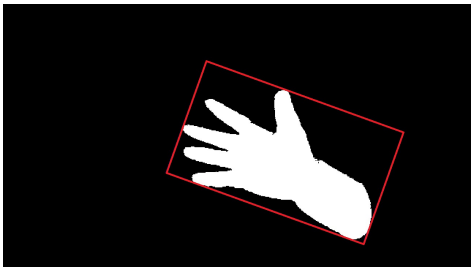
Rysunek 5.11: **Przykłady efektów wykonanej filtracji, która została powyżej opisana**

Cała filtracja w mojej ocenie bardzo dobre rezultaty, które znacząco polepszyły prawidłową segmentację dłoni, a przez to w późniejszych krokach dokładniejsze rozpoznawanie dłoni.

5.5 Znajdowanie obiektów

Ostatnim etapem przetwarzania obrazów w mojej pracy inżynierskiej było wydzielenie z maski binarnej prostokąta, zawierającego w sobie całą wysegmentową wcześniej dłoń, bez ogólnego tła. Dzięki czemu jesteśmy wstanie analizować jedynie tą część maski binarnej, która zawiera w sobie całą dłoń, bez zbędnych innych elementów.

Znajdowanie to polegało na wyszukaniu prostokąta na całej masce binarnej, która będzie miała największe pole z ze wszystkich znalezionych konturów na obiektach z maski binarnej.



Rysunek 5.12: **Zaznaczenie na czerwono prostokąta o największym polu, który zawiera w sobie całościowo obiekt**

Po znalezieniu takiego prostokąta następowało przycięcie maski do odpowiednich rozmiarów zgodnie z znalezionym prostokątem o największym polu. Prostokąt ten jest przyległy przez obiekt binarny z każdej swojej strony, co oznacza, że dokładnie przycięcie nie jest możliwe, bez zmniejsza znalezionego obiektu.

Rysunek 5.13: **Obcięcie zgodnie ze znalezionym prostokątem**

Etapy przycięcia maski binarnej:

1. Przypisz do obecnie znalezione największego pola prostokąta wartość 0
2. Znajdź wszystkie kontury obiektów na masce binarnej
3. Dla każdego konturu wykonaj:
   (a) Oblicz współrzędne lewego górnego oraz prawego dolnego wierzchołka prostokąta, który będzie otaczał analizowany kontur
   (b) Oblicz pole powierzchni tak utworzonego prostokąta
   (c) Jeżeli pole powierzchni utworzonego prostokąta jest większe niż pole powierzchni obecnie znalezionego największego prostokąta, to traktur tą powierzchnię jako największą i dodatkowo zapamiętaj współrzędne prostokąta

4. Przytnij maskę binarną o zapamiętanych wierzchołkach największego znalezionego prostokąta

Efekt końcowy wysegmentowania dłoni zawiera w sobie obiekt binarny reprezentujący dłoń, którą jest wykonywany gest. Zostanie on wykorzystany przez klasyfikatory opisany w następnych podrozdziałach klasyfikatora KNN oraz klasyfikatora opartego o sieci neuronowe.

(a) Gest papieru (b) Gest kamienia (c) Gest nożyc

Rysunek 5.14: **Końcowa segmentacja dłoni**

5.6 Dobór i ocena cech

Po wysegmentowaniu gestów dla uczenia maszynowego nadszedł czas, aby zgromadzony materiał opisać dobranymi cechami. Zbiór danych to wysegmentowana dłoń. Zdecydowałem się, że gesty będę opisywał w następujący sposób.

- Stosunek szerokości do długości
- Pole powierzchni znalezionego obiektu dłoni
- Pole powierzchni otoczki wypukłej utworzonej na znalezionej dłoni
- Stosunek półosi małej do półosi wielkiej elipsy opisanej na znalezionej dłoni
- Pole powierzchni otoczki wypukłej utworzonej elipsy na znalezionej dłoni

Przy ocenie jakości dobranych cech w pierwszej kolejności wykorzystałem odchylenie standardowe. Dzięki czemu miałem porównanie jak dana cecha wpływa na określony gest.

Kolejnym etapie było skuteczności wybranej cechy przy rozpoznawaniu konkretnych dwóch gestów. Do tego wykorzystałem wyt ‘odę...

Po wyżej wymienionej ocenie ustaliłem następujące cechy dla danego gestu:

Ważnym etapem dla każdej cechy była jej normalizacja, aby nie powodowała większego wpływy od innych

5.7 Rozpoznawanie gestów przy pomocy klasyfikatora KNN

5.8 Rozpoznawanie gestów przy pomocy sieci neuronowej

5.8.1 Sieć neuronowa dla cech

5.8.2 Sieć neuronowa obrazów binarnych

Rozdział 6

Podsumowanie

Bibliografia