Praca Inżynierska

Politechnika Śląska

Szymon Ciemała

Październik 2021

Spis treści

Rozdział 1

Wstęp

1.1 Ogólnie

Praca przedstawia wykorzystanie bibliotek OpenCV, MediaPipe oraz metod uczenia maszynowego bibliteki SciKit Learn do stworzenia modułu dla języka Python, który umożliwi sterwowanie dowolnymi aplikacjami przy użyciu gestów oraz ruchu dłońmi.

1.2 Wykorzystane technologie

1.2.1 OpenCV

Biblioteka, dzieki której można wykorzystać obraz z kamery oraz wstępnie przetworzyć obraz, który zostanie wykorzystany przez bibliotekę MediaPipe.

1.2.2 MediaPipe

Biblioteka MediaPipe o otwartym źródle, udostępnia wieloplatformowe oraz konfigurowalne rozwiązania wykrzustujące uczenie maszynowe w dziedzienie rozpoznawania, segmentacji oraz klasyfikacji obiektów wizji komputerowej. Niektórymi z rozwiązań są:

- Rozpoznawanie twarzy
- Segmentacjia włosów oraz twarzy
- Rozpoznawnie oraz określanie rozmiarów obiektów trójwymiarowych na podstawie obrazu dwuwymiarowego.

Platformy/Języki programowania obsługiwane przez MediaPipe:

- Android
- IOS
- JavaScript
- Python
- C++
- Coral

???? Pozwoli na rozpoznaie dłoni oraz jej elementów charakterystycznych, takich jak nagdarstek, stawy oraz końcówki palców. ???

1.2.3 SciKit Learn

SciKit Learn to biblioteka, która oferuje różnego typu metody uczenia maszynowego. Biblioteka zawiera algorytmy klasyfikacji, regresjii oraz analizy skupień. Biblioteka pozwala na wykorzystanie metod klasyfikacji uczenia maszynowego. Co pozwli na rozpoznanie gestów dłoni.

Rozdział 2

Założenia projektowe

2.1 Budowa modułu

Moduł został napisany przy pomocy paradygmatu programowania obiektowego, co pozwala na przystępne wykorzystanie biblioteki w dowonych projektach wymagających osbługi gestów.

2.2 Dostęp

Całość projektu będzie dostępna na platformie GitHub wraz z możliwością pobrania przy pomocy programu pip ze zdlanego repozytorium PyPi, dzięki czemu pozowli to na sprawne i proste wykorzystanie modułu w dowolnym projekcie.

Rozdział 3

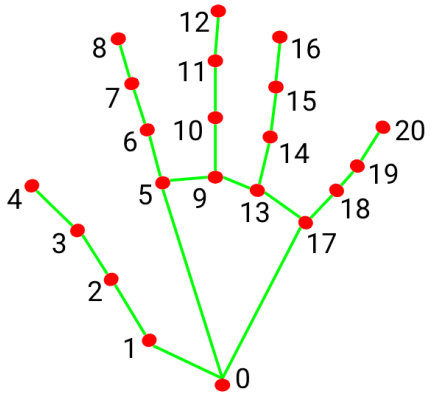
Część techniczna/praktyczna

3.1 Rozpoznawanie dłoni

Pierwszym elementem projektu jest rozpoznanie dłoni poprzez wyznacznie pozycji elementów charakterystycznych. Pozycja każdego z tych elementów jest względna według pozycji nadgarstka. ????

3.1.1 Elementy charakterystyczne

Model modułu MediaPipe pozwala na wyznaczenie pozycji 21 elementów charakterystycznych dłoni. Współrzędne X i Y są znomralizowane względem rozdzielczości obrazu kamery. Współrzędna X względem liczby pikseli w osi X, a współrzędna Y względem liczby pikseli w osi Y. Oś Z jest prostopadła do osi X i Y, z punktem początkowym w punkcie określającym pozycję nadgarstka. Współrzędna Z jest znormalizowana względem szerokości obrazu kamery, tak jak współrzędna X.



0. WRIST
1. THUMB_CMC
2. THUMB_MCP
3. THUMB_IP
4. THUMB_TIP
5. INDEX_FINGER_MCP
6. INDEX_FINGER_PIP
7. INDEX_FINGER_DIP
8. INDEX_FINGER_TIP
9. MIDDLE_FINGER_MCP
10. MIDDLE_FINGER_PIP
11. MIDDLE_FINGER_DIP
12. MIDDLE_FINGER_TIP
13. RING_FINGER_MCP
14. RING_FINGER_PIP
15. RING_FINGER_DIP
16. RING_FINGER_TIP
17. PINKY_MCP
18. PINKY_PIP
19. PINKY_DIP
20. PINKY_TIP

Rysunek 3.1: Elementy charakterystyczne dłoni

Pozycje nadgarstka, paliczków oraz stawów dłoni zostaną wykorzystane do obliczenia obrotu dłoni względem punkut 0 oraz do wytrenowania modeli uczenia maszynowego, których zadaniem będzie rozpoznawnie wybranych gestów.

3.1.2 OpenCV - przygotowanie obrazu z kamery

Poprawne działanie modelu MediaPipe wymaga odpowiedniego przygotowania obrazu kamery. Działanie kontrolerolera odbywa się poprzez główną metodę **main()**

Metoda **main()** jest główną funkcją, w której dokonywane są obliczenia oraz przeszktałcenia pozwalające na obliczenie obrotu dłonie, odległości między wybranymi palcami oraz na wykrycie gestu.

```
51          #Initiate camera
52          self.cap = cv2.VideoCapture(0)
```

W pierwszym kroku tworzymy instancję klasy **VideoCapture** biblioteki **OpenCV**, która pozwoli na odczytywanie obrazu kamery.

```
272     def main(self):
273         """
274         Main function that runs the core of the program.
275         """
276 
277         hand_type = None
278         if self.cap.isOpened():
279             ret, frame = self.cap.read()
280 
281             #BGR to RGB
282             image = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
283 
284             #Flip horizontal
285             image = cv2.flip(image, 1)
286 
287             #Set flag
288             image.flags.writeable = False
289 
290             #Detections
291             self.results = self.hands.process(image)
292 
293             #Set flag back to True
294             image.flags.writeable = True
295 
296             #RGB to BGR
297             image = cv2.cvtColor(image, cv2.COLOR_RGB2BGR)
```

tekst testowy

Rozdział 4

Możliwości wykorzystania

4.1 Kioski samoobsługowe

4.2 Obsługa komputera

4.3 Sterowanie Robotami Mobilnymi