ZESPOŁOWE PRZEDSIĘWZIĘCIE INŻYNIERSKIE:

„Opracowanie aplikacji do monitorowania zachowania kierowcy podczas jazdy na podstawie danych z czujników dostępnych w telefonie komórkowym”

RAPORT KOŃCOWY

Berezowski Wojciech, 243113

Kolarczyk Robert, 244821

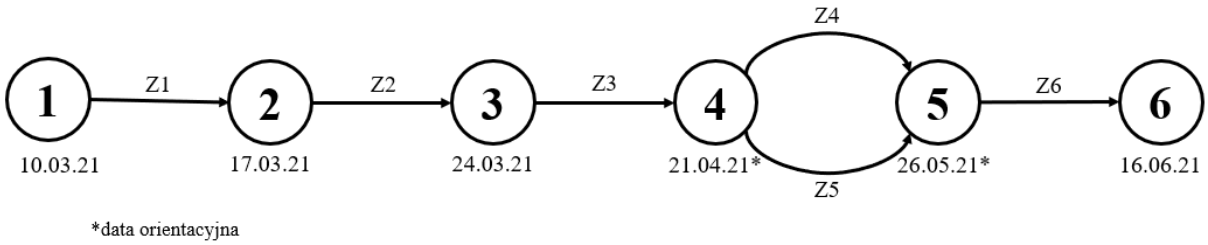
Opiekun:
Dr hab. inż. Krzysztof Brzostowski

WROCŁAW 2021

Spis treści:

1. Harmonogram prac.

Początkowym etapem realizacji projektu było wykonanie harmonogramu prac obejmujących zadania do wykonania w czasie całego semestru. Hipotetyczną strukturę projektu przedstawiono na rysunku 1. Lista zadań do wykonania wraz z ich spodziewanymi efektami przedstawiono w tabeli 1.



Rys. 1. Hipotetyczna struktura projektu.

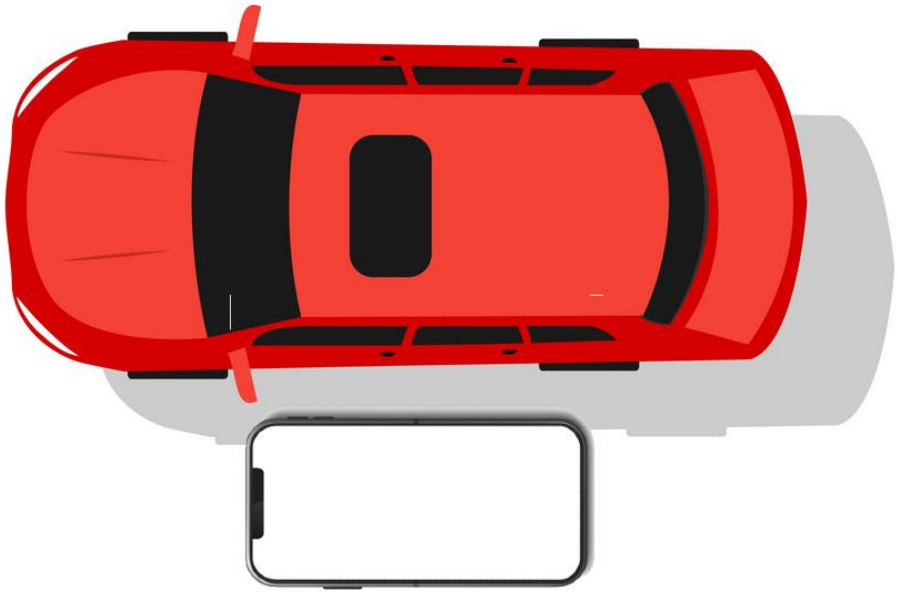
Tabela 1. Plan zadań i ich efekty

Nr zadania	Zadanie	Efekt	Wykonawcy	Termin wykonania
1	Zapoznanie się z podstawami przetwarzania danych pomiarowych z czujników dostępnych w nowoczesnych telefonach komórkowych;	Dokumentacja-pisemna część raportu końcowego	Berezowski Wojciech 243113 Kolarczyk Robert 244821 Parzonka Anna 246802	17.03.21
2	Zaplanowanie i wykonanie eksperymentów w celu pozyskania danych pomiarowych z wykorzystaniem gotowej aplikacji zbierającej dane;	Dokumentacja-pisemna część raportu końcowego		24.03.21
3	Opracowanie algorytmów przetwarzania zgromadzonych danych na potrzeby monitorowania zachowania kierowcy w samochodzie;	Przygotowany kod Dokumentacja-pisemna część raportu końcowego		21.04.21*
4	Opracowanie wyników przetwarzania zebranych danych pomiarowych;	Przygotowany kod Dokumentacja-pisemna część raportu końcowego		26.05.21*
5	Implementacja aplikacji do monitorowania zachowania kierowcy w samochodzie wykorzystującej opracowane algorytmy przetwarzania danych;	Przygotowany kod		26.05.21*
6	Przygotowanie dokumentacji powykonawczej i instrukcji użytkownika dla opracowanej aplikacji.	Raport końcowy w formie pisemnej		16.06.21

** data orientacyjna*

2. Geneza pracy.

W dzisiejszych czasach smartfony odgrywają jedną z głównych ról w naszym codziennym życiu oraz funkcjonowaniu. Telefon komórkowy nie jest już kojarzony jedynie z możliwością komunikacji z innymi osobami, lecz z szeregiem innych zadań, które są z jego pomocą realizowane. Ich powszechność sprawia, że z biegiem czasu urządzenia te spełniają coraz to więcej funkcji, co tylko zwiększa ich atrakcyjność dla potencjalnego użytkownika. Smartfony znajdują swoje zastosowanie w ogromie różnych dziedzin życia, np.: bankowość, rozrywka, aktywność fizyczna, źródło wiedzy, czy kalendarz, ale również swoje zastosowanie znajdują wśród kierowców. Można je wykorzystywać jako nawigacje, ale także w celu monitorowania jazdy oraz oceny jej poprawności. Aby zwiększyć bezpieczeństwo podczas prowadzenia pojazdu często umieszcza się telefony w specjalnych uchwytach, znajdujących się w widocznym, łatwo dostępnym miejscu. My jednak będziemy musieli zrezygnować z takiego położenia, aby umożliwić sensorom poprawne zebranie danych. Na rysunku 1. pokazano w jaki sposób kierowca powinien umieścić telefon w samochodzie, aby mógł używać stworzonej przez nas aplikacji.



Rys. 2. Telefon komórkowy umieszczony w odpowiedni sposób.

3. Cel pracy.

Celem projektu jest stworzenie aplikacji monitorującej kierowców pod względem agresywności ich jazdy. Wykorzystuje ona dane z czujników wbudowanych w smartfon. Aby poprawnie zebrać te dane, należy umieścić smartfon tak jak na rysunku 2.

4. Zakres prac.

Aby wykonać projekt z realizacją postawionego celu konieczne będzie wykonanie następujących prac:

1. Zdefiniowanie zachowania kierowcy.
2. Wybór aplikacji zbierającej dane z czujników.
3. Pobranie danych zebranych przez czujniki z telefonu.
4. Wstępne przetwarzanie danych.
5. Ekstrakcja cech z zebranych danych.
6. Uczenie modelu klasyfikatora.
7. Ocena jakości stworzonego modelu.
8. Stworzenie aplikacji monitorującej zachowania kierowcy.

5. Zdefiniowanie zachowania kierowcy.

Wykorzystując telefon komórkowy możliwe jest sklasyfikowanie manewrów wykonywanych przez kierowcę na agresywne i spokojne. Manewry agresywne charakteryzują się nagłymi przyśpieszeniami lub hamowaniami. Natomiast manewry spokojne charakteryzują się płynnymi zmianami prędkości.

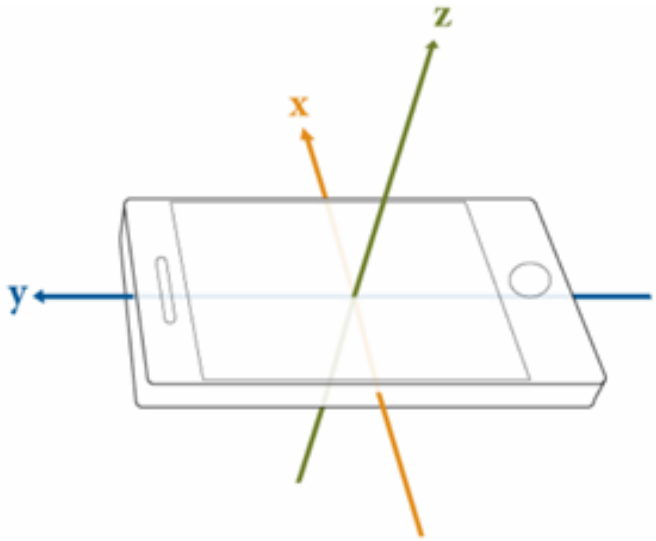
Dodatkowo, wykorzystując smartfon, można określić inne zachowania kierowcy które nie były przedmiotem projektu, np. czy kierowca trzyma telefon w dłoni i skupia na nim uwagę podczas jazdy, czy może jest on zamocowany w uchwycie bądź schowku. Można również monitorować długość, szerokość i wysokość geograficzną, a także kierunek i prędkość poruszania się pojazdu.

6. Czujniki w smartfonach.

Używanie czujników w smartfonie jest jedną z lepszych metod na zbieranie informacji o kierowcy, ponieważ telefon jest zdecydowanie tańszy oraz łatwiej dostępny niż specjalistyczny sprzęt do monitorowania ruchu drogowego. Na korzyść tego rozwiązania przemawia również fakt, że duża część społeczeństwa posiada smartfony. Spis czujników znajdujących się we współczesnych telefonach przedstawiono w tabeli 2. W ramach projektu wykorzystano dane uzyskane z akcelerometru, którego osie orientacji przedstawiono na rysunku 3. Osie te umożliwiają analizowanie uzyskanych danych.

Tabela 2. Czujniki znajdujące się w smartfonach.

CZUJNIK	OPIS
Akcelerometr (Acceleration)	Sensor mierzący przyspieszenie urządzenia. W telefonach służy do ustalania orientacji telefonu, wykrywając ruch w danym kierunku.
Żyroskop (Orientation, Angular Velocity)	Służy zarówno do pomiaru prędkości kątowej jak i położenia urządzenia. W urządzeniach mobilnych odgrywa również rolę w ustalaniu orientacji telefonu, podobnie jak akcelerometr. Dodaje informacje o obrocie i skręcie urządzenia.
Magnetometr (Magnetic Field)	Pomiar siły pola magnetycznego. W smartfonie używany do ustalania orientacji urządzenia w stosunku do magnetycznej północy planety. Sprawuje funkcje cyfrowego kompasu.
GPS (Position)	Pomiar takich wartości jak: długość i szerokość geograficzna, prędkość, wysokość nad poziomem morza. W telefonach używany do nawigacji satelitarnej.



Rys. 3. Osie orientacji akcelerometru w odniesieniu do typowego smartfonu z Androidem.

7. Przegląd aplikacji do zbierania danych z czujników.

W celu uzyskania danych z czujników telefonu konieczne było wykorzystanie aplikacji zbierającej dane z takich sensorów. W tym celu dokonano przeglądu aplikacji dostępnych na rynku. Zestawienie analizowanych aplikacji przedstawiono w tabeli 3 wraz z podkreśleniem spełnienia poszczególnych kryteriów takich jak: obsługa akcelerometru, możliwość zapisu danych do pliku, brak kosztów eksploatacji aplikacji, możliwość instalacji na systemach operacyjnych: iOS oraz Android. Każde kryterium posiada taką samą wagę, ponieważ wszystkie kryteria muszą zostać spełnione, aby możliwe było wykorzystanie danej aplikacji w projekcie.

Tabela 3. Zestawienie analizowanych aplikacji zbierających dane z czujników wraz z kryteriami

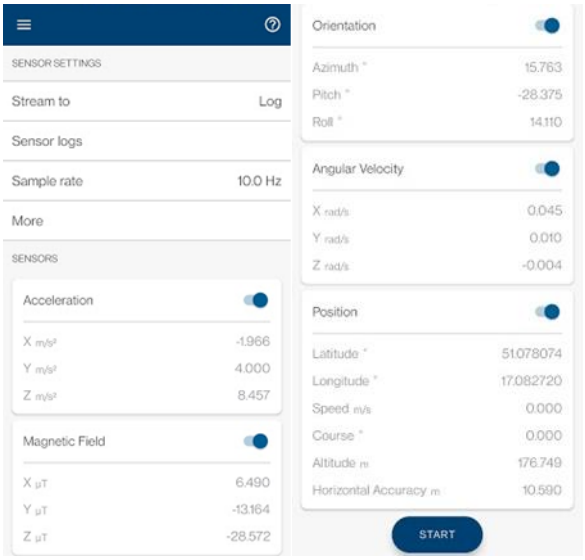
Aplikacja	Akcelerometr	Zapis danych	Brak kosztów	iOS/Android	Suma
Sensory i Czujniki	+	-	+	+	3
Sensor Box	+	+	-	-	2
Sensor Test	+	-	+	+	3
SKC	+	-	+	-	1
MATLAB	+	+	+	+	**4**

Po wnikliwej analizie zalet poszczególnych pozycji oraz konsultacji z opiekunem przedsięwzięcia zdecydowano o wykorzystaniu aplikacji mobilnej środowiska MATLAB, przedstawionej na rysunku 4. Aplikacja na smartfony pozwala na korzystanie z wielu czujników, których spis przedstawiono w tabeli 4 wraz z informacjami jakie można uzyskać z ich zastosowaniem. Dużą zaletą tej aplikacji jest możliwość zbierania danych ze wszystkich czujników jednocześnie, a samo użytkowanie jest dla studentów darmowe wykorzystując licencję edukacyjną.

Tabela 4. Czujniki wykorzystywane przez aplikację MATLAB.

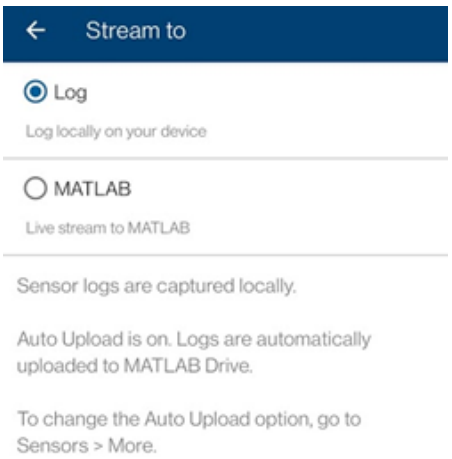
CZUJNIK	UZYSKANA INFORMACJA
Akcelerometr (Acceleration)	Przyśpieszenie [m/s^2]
Żyroskop (Orientation, Angular Velocity)	Prędkość kątowa [rad/s] Orientacja urządzenia [°].
Magnetometr (Magnetic Field)	Siła pola magnetycznego [μT]
GPS (Position)	Długość i szerokość geograficzna [°] Prędkość [m/s] Wysokość nad poziomem morza [m]

W ramach projektu wykorzystano akcelerometr, jednakże możliwość zbierania danych ze wszystkich dostępnych czujników jednocześnie umożliwiłaby w przyszłości rozbudowę aplikacji.

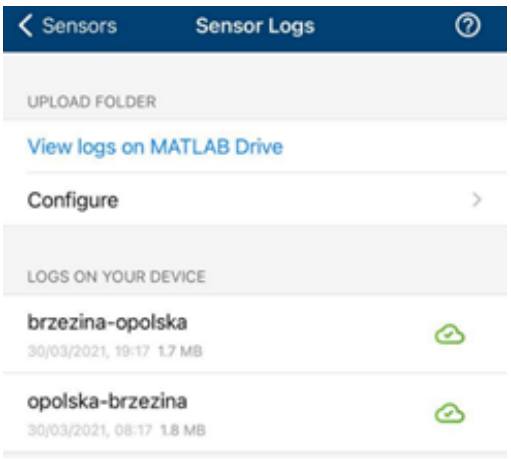


Rys. 4. Ekran startowy aplikacji MATLAB, zawierający listę dostępnych czujników.

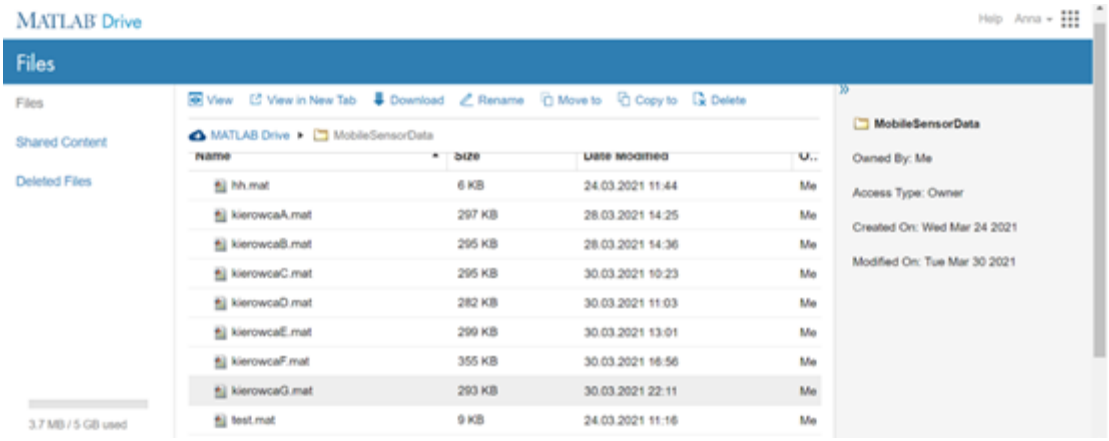
Aplikacja MATLAB pozwala na logowanie danych w dwojaki sposób, co przedstawiono na rysunku 5: możemy zapisywać pliki z danymi bezpośrednio w pamięci urządzenia lub wysyłać dane do środowiska MATLAB. Program pozwala również na wysyłanie plików z zebranymi danymi od razu do chmury MathWorks, do której mamy dostęp np. w przeglądarce, co przedstawiono na rysunkach 6-7. Proces gromadzenia i zapisu danych jest bardzo szybki. Kolejną ważną zaletą aplikacji MATLAB jest możliwość zbierania danych w tle, dzięki czemu użytkownik może korzystać z innych aplikacji np. nawigacji, wykonywania połączeń czy kontroli odtwarzanej muzyki. Aplikacja została uruchomiona zarówno na smartfonach z systemem iOS i Android, które posiadają takie czujniki jak: akcelerometr, żyroskop, magnetometr, GPS. Narzędzie obsługuje wszystkie wymienione czujniki, w tym akcelerometr, z którego zdecydowaliśmy się korzystać.



Rys. 5. Możliwość wyboru trybu zapisywania danych.



Rys. 6. Szybki dostęp do plików z danymi oraz ich stan wysłania.



Rys 7. Dostęp do plików z danymi na komputerze.

8. Plan oraz przeprowadzenie eksperymentu.

Opracowano podstawowe założenia przeprowadzenia eksperymentów: czas eksperymentu: 300 s (5 min), w chwili „0" rozpoczynamy pomiar naciskając przycisk „START" w dole ekranu oraz uruchamiamy aplikację w trybie zbierającym dane. Następnie aplikacja przez 300 s. zbiera dane z wybranych czujników po czym zatrzymujemy tryb zbierania danych przyciskiem „STOP". Kierowca wykonuje kilka 5-minutowych pomiarów.

W ramach przeprowadzenia eksperymentu wykonano 7 pomiarów: Kierowca A, Kierowca B, Kierowca C, Kierowca D, Kierowca E, Kierowca F oraz Kierowca G. Pliki wyeksportowane z aplikacji mają postać surowych danych, co przedstawiono na rysunku 8. Dane te można również w łatwy sposób przekonwertować na inny format przez prostą komendę wpisaną w wiersz poleceń w MATLABIE:

writetimetable(data.Acceleration, 'Acceleration.csv').

Acceleration

3590x3 timetable

	Timestamp	1 X	2 Y	3 Z
1	30-Mar-2021 13:01...	2.2702	0.1891	9.8337
2	30-Mar-2021 13:01...	1.8597	-0.4004	9.3584
3	30-Mar-2021 13:01...	1.8138	-0.4489	9.5922
4	30-Mar-2021 13:01...	1.9494	-0.1040	9.4982
5	30-Mar-2021 13:01...	2.0030	0.1406	9.7078
6	30-Mar-2021 13:01...	1.9031	-0.2777	9.4362
7	30-Mar-2021 13:01...	1.7678	-0.4573	10.0196
8	30-Mar-2021 13:01...	1.6115	-0.8282	9.5353
9	30-Mar-2021 13:01...	1.9124	-0.7839	9.2557
10	30-Mar-2021 13:01...	2.1432	-0.5902	10.0561

Rys. 8. Przykładowe dane z akcelerometru dla Kierowcy F w programie MATLAB.

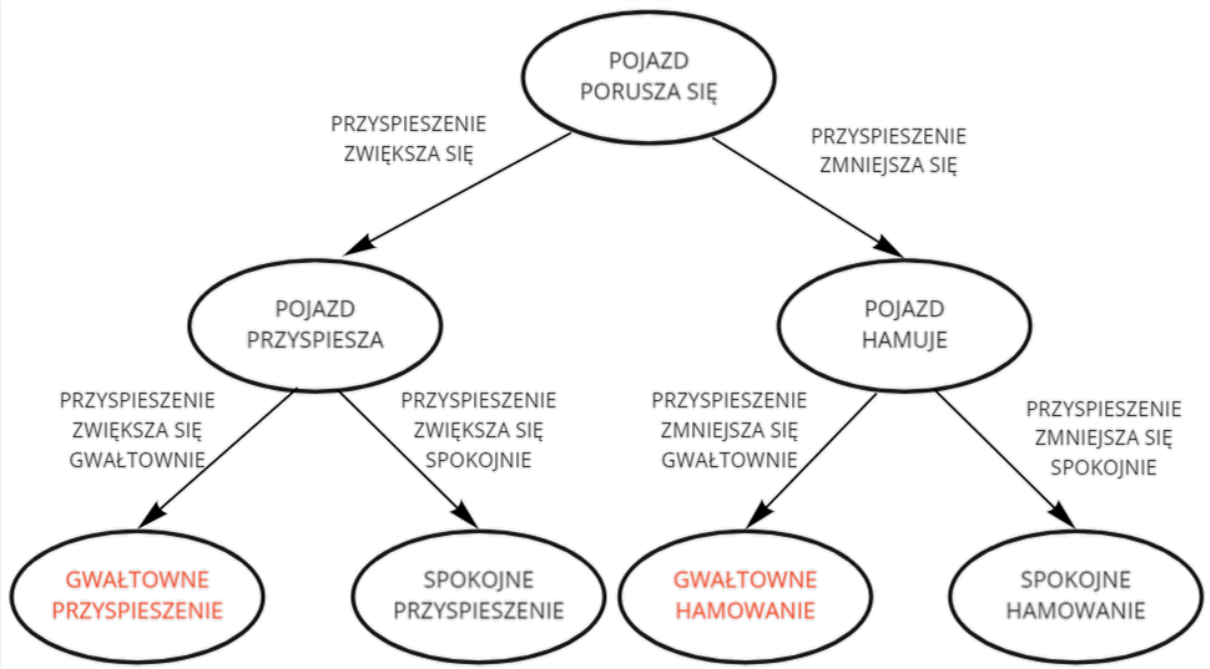
Z danych zamieszczonych na powyższym rysunku wywnioskowano, że w trakcie pomiarów telefon leżał relatywnie płasko, czyli zgodnie z zaleceniami, ponieważ wartości prędkości wzdłuż osi Z są zbliżone do wartości przyspieszenia grawitacyjnego naszej planety. Wartości dodatnie oraz ujemne uzyskane w ramach danych wskazują na kierunek przemieszczania się telefonu po danej osi, zgodnie lub przeciwne do założonego przez czujnik kierunku (rysunek 3), np. dla osi Y wartości dodatnie świadczą o przesunięciu telefonu do przodu, natomiast ujemne w tył.

9. Opis działania algorytmu.

Dane potrzebne do klasyfikacji manewrów wykonywanych przez kierowcę rejestrowane są za pomocą aplikacji MATLAB, zgodnie z płaszczyznami w których akcelerometr zbiera dane, które opisano we wcześniejszej części niniejszej pracy i przedstawiono na rysunku 3. Zdecydowano, że do klasyfikacji manewrów przyspieszania i hamowania zbędne będą dane z akcelerometru w płaszczyźnie X i Z.

Stworzona aplikacja będzie działać zgodnie z założonym algorytmem postępowania:

1. Przetworzenie danych w Matlabie i Pythonie.
2. Ekstrakcja cech z przetworzonych danych.
3. Klasyfikacja manewrów jako **agresywne** bądź **spokojne** realizowana zgodnie z grafem przedstawionym na rysunku 9.
4. Wyświetlenie raportu informującego o rezultatach klasyfikacji w postaci statystyk dotyczących jazdy.



Rys. 9. Graf przedstawiający klasyfikację manewrów na spokojne oraz agresywne.

10. Wstępne przetwarzanie danych.

Dane, których fragment został zaprezentowany na rysunku 10., wykorzystywane do oceny agresywności jazdy kierowcy są pozyskiwane z wykorzystaniem środowiska Matlab. W pierwszej kolejności należało załadować uzyskane pliki z telefonu. W celu łatwiejszego przekształcania danych eksportowano wszystkie rezultaty z programu MATLAB do pliku csv.

Zauważono, że uzyskane dane nie są odpowiednio przygotowane do dalszej analizy, ponieważ sensory nie zbierają danych w regularnych odstępach czasu – na każdą sekundę monitorowania nie zawsze przypada tyle samo rekordów z danymi. Dane rzeczywiste, czyli nasze surowe dane, są często niekompletne. Pre-processing miał na celu przygotować dane, aby móc zbudować algorytm przystosowany do każdego przypadku, przykładowo, gdy każdy kierowca jedzie inną drogą. W tym celu wykonano przetworzenie rekordów. Na tym etapie dane zostały poddane procesowi agregacji i ponownemu próbkowaniu z wykorzystaniem interpolacji liniowej, przy użyciu wbudowanej funkcji *synchronize* w środowisku MATLAB. Odszumianie polega na usunięciu z pomiarów danych przypadkowych, czyli mocno odległych od średnich (przeciętych) wartości - usuwa pomyłki pomiarowe. Można powiedzieć, że jest to wyselekcjonowanie poprawnych danych ze strumienia przypadkowych danych. Wykonano również standaryzację oraz normalizację uzyskanych danych w celu zwiększenia trafności modelu.

Standaryzacja to odejmowanie od wartości zmiennej średniej ze zbioru wszystkich zmiennych i dzielenie przez odchylenie standardowe z tego samego zbioru zmiennych co przedstawia następujący wzór:

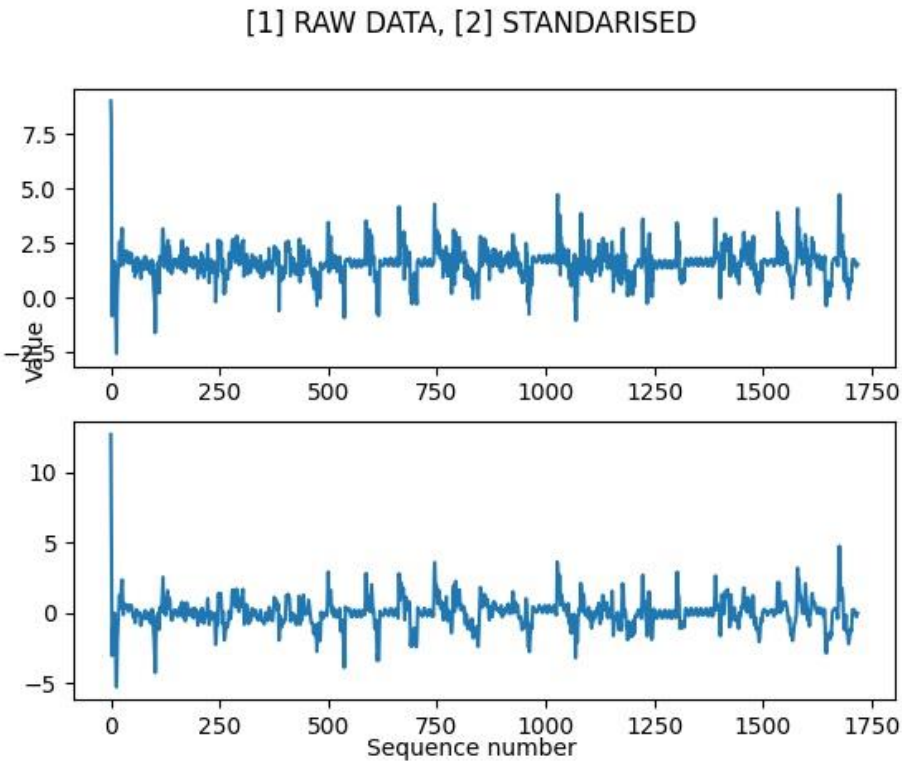
$$x_{stand} = \frac{x - \overline{x}}{\sigma}$$

Przyjmuje się, że poddane standaryzacji wartości powinny znajdować się w zbiorze (-3, 3). Standaryzacja lepiej sprawdza się w bardziej ogólnych przypadkach. Dzięki standaryzacji średnie otrzymane z różnych źródeł mogą być między sobą porównywane. Standaryzowane zmienne mają te same średnie i odchylenia standardowe – mówią czy i jak bardzo jednostka odstawała od pozostałych w próbie oraz czy było to odchylenie na + czy –.

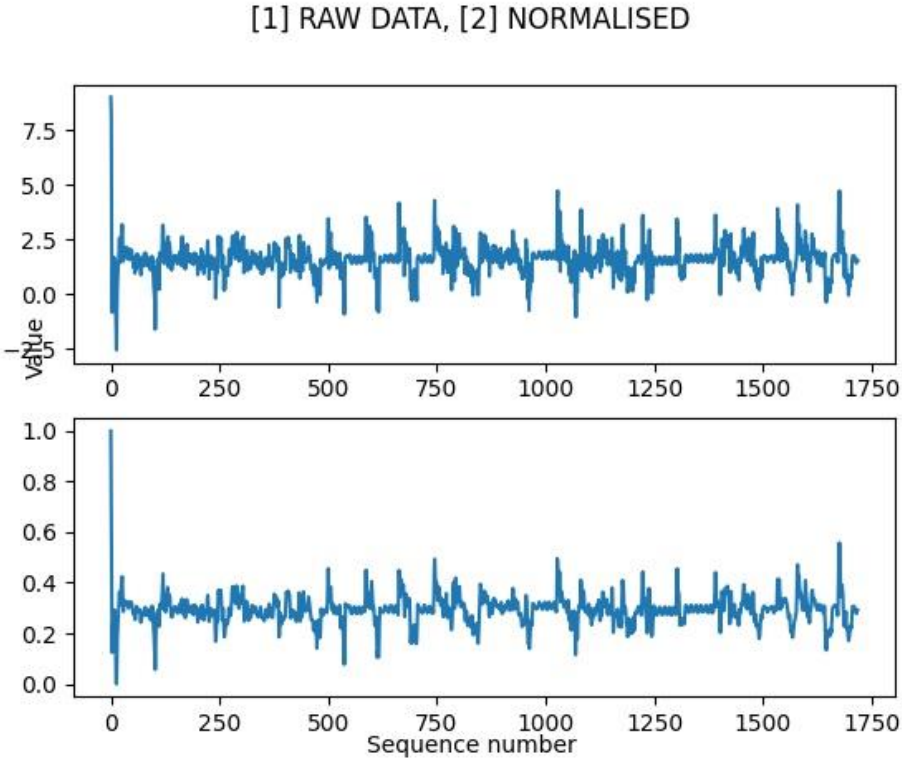
Normalizacja natomiast to odejmowanie wartości zmiennej od minimalnej wartości z danego zbioru zmiennych i dzielenie przez różnice między maksymalną a minimalną wartością z tego samego zbioru zmiennych, co przedstawia następujący wzór:

$$x_{norm} = \frac{x - \min(x)}{\max(x) - \min(x)}$$

Przyjmuje się, że poddane normalizacji wartości powinny znajdować się w zbiorze (0, 1). Normalizacja jako metoda skalowania sprawdza się najlepiej, kiedy zbiory zmiennych mają rozkład normalny i jest uznawana za dobre narzędzie tylko w specyficznych sytuacjach. Wyniki standaryzacji oraz normalizacji przedstawiono na wykresach 1 – 2, natomiast kod w środowisku Python wykorzystany w tym celu znajduje się w załączniku 1. Po analizie wyników zdecydowaliśmy o wykorzystaniu standaryzacji.



Wykres 1. Wynik standaryzacji danych.



Wykres 2. Wynik normalizacji danych.

Działania te były motywowane faktem, iż akcelerometr jest sensorem bardzo wrażliwym na różnego rodzaju zakłócenia. W literaturze stosuje się procesy wstępnego przygotowania danych w celu np. odfiltrowywanie wibracji samochodu powodowanej jazdą lub zmiany odnotowywane przez różnego rodzaju nierówności. Na zaszumienie danych z akcelerometru ma również wpływ oddziaływanie grawitacyjne (*Hemminki et al., 2013; Shin et al., 2015*). W celu poprawienia jakości takich danych używa się m.in filtrów dolnoprzepustowych (*Lu et al., 2018*).

	Timestamp	1 Y
1	13-Jun-2021 19:02:...	9.0189
2	13-Jun-2021 19:02:...	8.2641
3	13-Jun-2021 19:02:...	2.0880
4	13-Jun-2021 19:02:...	-0.8329
5	13-Jun-2021 19:02:...	0.1946
6	13-Jun-2021 19:03:...	1.1493
7	13-Jun-2021 19:03:...	0.9039
8	13-Jun-2021 19:03:...	1.5553
9	13-Jun-2021 19:03:...	1.6731
10	13-Jun-2021 19:03:...	1.1940
11	13-Jun-2021 19:03:...	1.4961
12	13-Jun-2021 19:03:...	-0.6052
13	13-Jun-2021 19:03:...	-1.4249
14	13-Jun-2021 19:03:...	-2.5581
15	13-Jun-2021 19:03:...	-0.6278

Rys.10. Dane zebrane w programie Matlab.

11. Ekstrakcja cech.

Aby w sposób poprawny sklasyfikować manewry konieczne jest stworzenie cech, które określą czy dane zachowanie kierowcy było agresywne czy spokojne. W celu stworzenia cech zmieniono strukturę danych. Dane uzyskane z wykorzystaniem akcelerometru uzyskane w kolejnych sekundach tworzą następujący zbiór:

$$z(0), z(1), \dots, z(n-1), z(n), z(n+1), \dots, z(M)$$

Z tak otrzymanego zbioru wyznaczono średnią arytmetyczną:

$$\bar{Z} = \frac{z(0) + \ \dots + \ z(M)}{M+1}$$

11.1. Detekcja manewru.

W przypadku manewru hamowania oraz przyśpieszenia, cechą je opisującą będzie znaczne odchylenie wartości przyspieszenia od średniej dla całego czasu trwania doświadczenia. W tym celu detekcji manewrów hamowania oraz przyśpieszenia wyznaczono ze zbioru danych wartości odbiegające od wartości średniej o 0,1:

$$z(n) < \bar{Z} - 0{,}1$$

oraz:

$$z(n) > \bar{Z} + 0{,}1$$

Następnie dane podzielono na 3-elementowe zestawy danych, na podstawie których ustalano, czy detekcja nastąpiła dla hamowania czy dla przyśpieszenia. Określono to na podstawie różnicy pierwszego i trzeciego wyniku 3-elementowego zbioru. Hamowanie wykazano, gdy:

$$z(n) > z(n+2),$$

natomiast przyśpieszenie, gdy:

$$z(n) < z(n+2)$$

11.2. Klasyfikacja manewru.

W celu klasyfikacji manewru na agresywny i spokojny ustalono wartość progową P po przekroczeniu, której manewr zostaje sklasyfikowany jako agresywny w sytuacji, gdy wartość bezwzględna z różnicy pierwszej i ostatniej wartości danej grupy jest większa od danego progu granicznego (przypisując mu wartość 1):

$$|z(n+2) - z(n)| > P$$

W przeciwnym przypadku klasyfikuje się manewr jako manewr spokojny (przypisując mu wartość 0).

Przykład klasyfikacji manewru hamowania został przedstawiony w tabeli 5, natomiast przykład klasyfikacji manewru przyśpieszenia w tabeli 6. Algorytm przedstawiono również w postaci schematu blokowego, który został przedstawiony w załączniku 1. Szczegóły implementacyjne uwzględniające kod stworzonych funkcji uwzględniono w załączniku 2 do niniejszego raportu.

Tabela 5. Klasyfikacja manewru hamowania.

z(n)	z(n+1)	z(n+2)	|z(n+2)-z(n)|	Czy agresywny?
3.80	3.80	3.69	0.11	0
3.65	2.96	2.00	1.65	1
2.05	1.77	-0.12	2.17	1

Tabela 6. Klasyfikacja manewru przyśpieszenia.

z(n)	z(n+1)	z(n+2)	|z(n+2)-z(n)|	Czy agresywny?
2.92	3.13	3.41	0.48	0
0.95	1.31	3.19	2.24	1
1.73	1.94	2.05	0.32	0

12. Podsumowanie.

12.1. Wyniki z programu

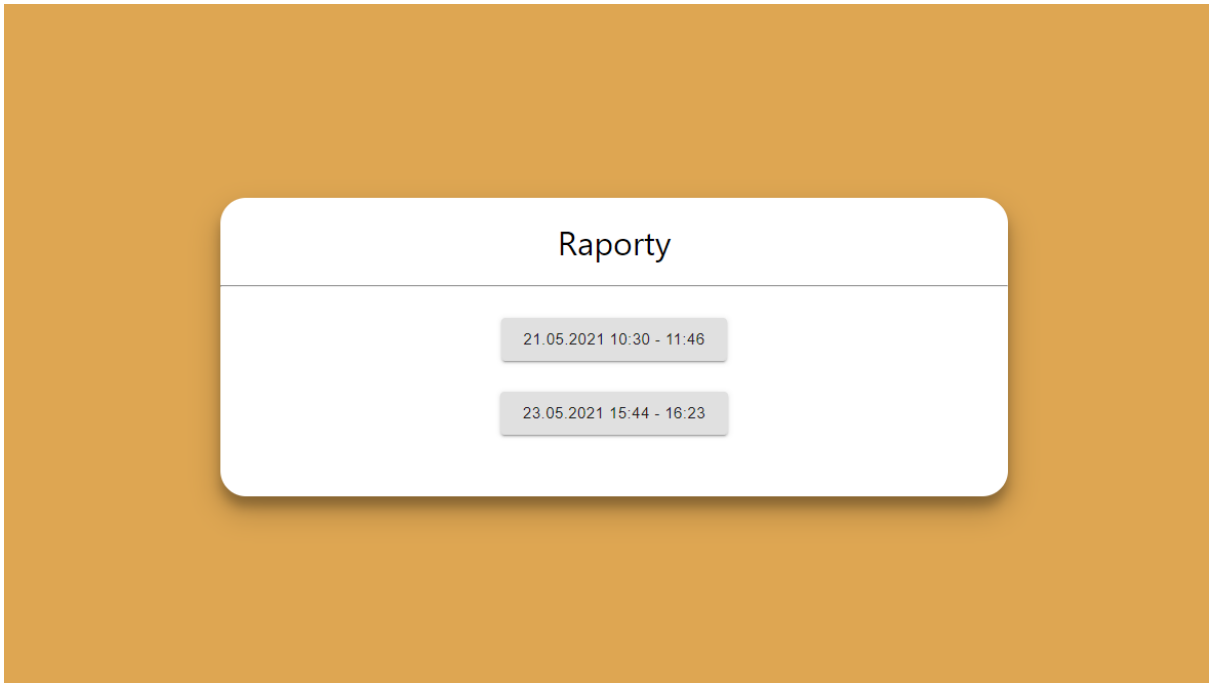
Nasz program zwraca statystyki dotyczące jazdy, które zdecydowaliśmy się przedstawić w formie tabeli 7. Do statystyk zdecydowaliśmy się dołączyć również dane o prędkości pochodzące z czujnika GPS. Wyniki pochodzą z przykładu pierwszego.

Tabela 7. Statystki dotyczące jazdy z przykładu 1..

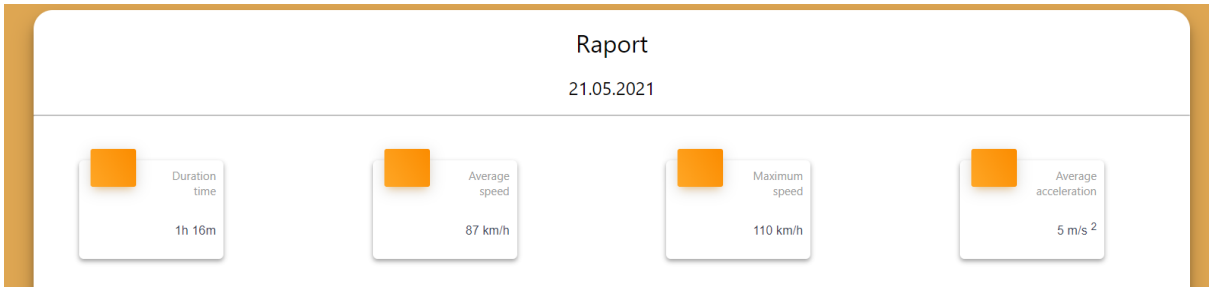
Informacja	**Wartość**
Data przejazdu	2021-06-13
Czas rozpoczęcia	19:02:55
Czas zakończenia	19:31:33
Czas przejazdu	0:28:38
Liczba wszystkich wykrytych manewrów	473
Liczba wykrytych manewrów hamowania	247
Liczba agresywnych hamowań	61
Liczba spokojnych hamowań	186
Liczba wykrytych manewrów przyspieszania	226
Liczba agresywnych przyspieszeń	64
Liczba spokojnych przyspieszeń	162
Średnia prędkość	9.35 m/s
Maksymalna prędkość	20.34 m/s
Średnie przyspieszenie	1.6 m/s^2
Maksymalne przyspieszenie	9.02 m/s^2

12.2. Widoki systemu webowego

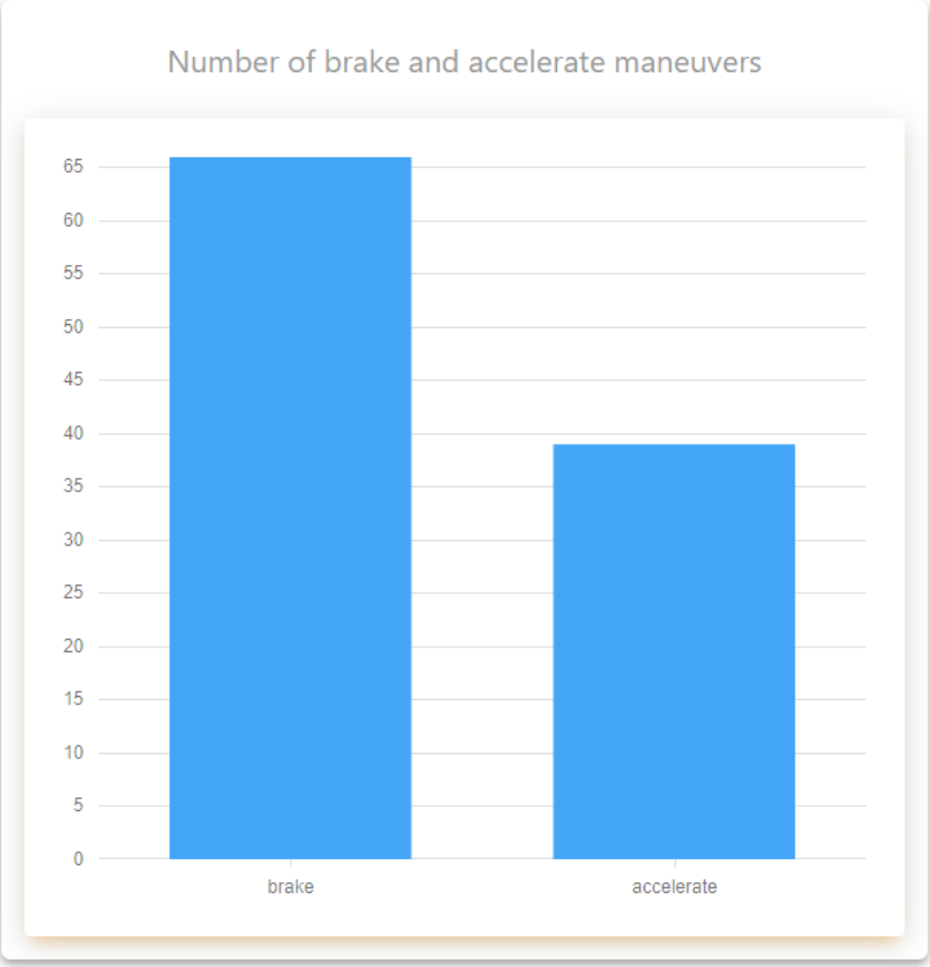
Zdecydowaliśmy się również zaimplementować prosty system webowy, który służy jako graficzne przedstawienie wyników zwracanych przez program. System posiada ekran główny (przedstawiony na rysunku 11.) na którym użytkownik ma możliwość wyboru raportu dotyczącego konkretnej jazdy. Po kliknięciu w konkretny raport wyświetla się widok szczegółowy, na którym w jasny i minimalistyczny sposób są zaprezentowane statystyki dotyczące jazdy w formie kafelków i wykresów. Zostały one przedstawione na rysunkach 12-14.



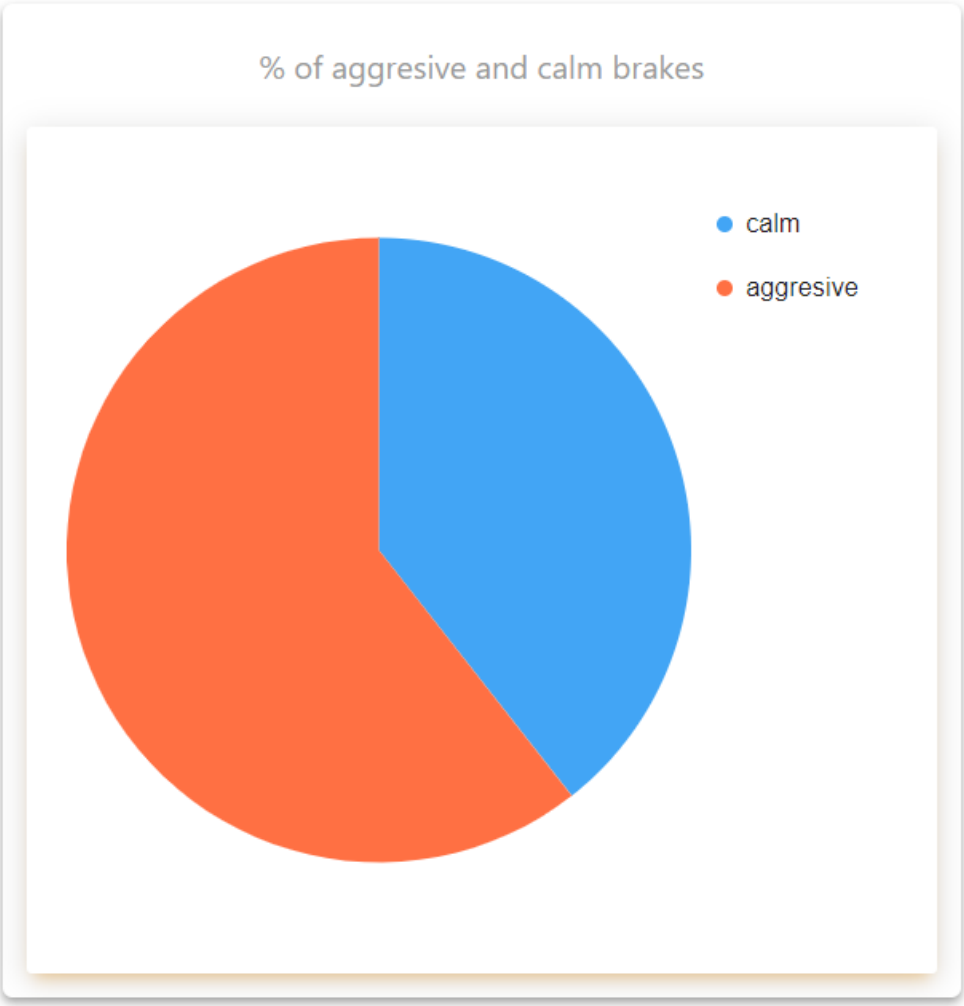
Rys.11. Strona główna systemu



Rys.12. Kafelki z informacjami o jeździe



Rys.13. Wykres porównujący liczbę wykrytych manewrów



Rys.14.Liczba manewrów sklasyfikowanych jako agresywne bądź spokojne

ZAŁĄCZNIK 1

W załączniku uwzględniono fragmenty kodów wykorzystywanych do realizacji poszczególnych etapów aplikacji. Kod przedstawiony na rysunkach Z1 – Z3 został wykorzystany w ramach wstępnego przetwarzania danych. Rysunki Z4 oraz Z5 przedstawiają kod stworzony w ramach ekstrakcji cech. Rysunki Z6 – Z7 przedstawiają fragmenty kodu potrzebnego do uczenia modelu klasyfikacji, natomiast rysunek Z8 przedstawia kod wykorzystany podczas oceny jakości modelu.

```
clc
clear
load('example1.mat')
acc = synchronize(Acceleration(:, 2:end-1),'regular', 'linear', 'TimeStep',seconds(1));
speed = Position(:, 4);
writetimetable(acc, 'acceleration.csv')
writetimetable(speed, 'speed.csv')
```

Rys. Z1. Kod realizujący załadowanie plików .mat do środowiska oraz przetworzenie danych tak aby rekordy były co 1 s.

```
def run_matlab_in_python():
    eng = matlab.engine.start_matlab()
    eng.writing(nargout=0)
    eng.quit()
```

Rys. Z2. Funkcja uruchamia kod napisany w Matlabie w środowisku Python.

```
import matplotlib.pyplot as plt
import pandas as pd
from sklearn.preprocessing import StandardScaler, MinMaxScaler
import statistics
import numpy as np
from scipy.signal import savgol_filter
import matlab.engine
import datetime
```

Rys. Z3. Importowanie potrzebnych bibliotek w środowisku Python.

```python
def excel_to_python():
    acceleration = pd.read_csv('acceleration.csv')
    speed = pd.read_csv('speed.csv')
    return acceleration, speed
```

Rys. Z4. Funkcja excel_to_python pobierająca z pliku .csv dane

```python
def standarisation(dataframe):
    sc = StandardScaler()
    scaled_df = sc.fit_transform(dataframe)
    standarised_df = pd.DataFrame(scaled_df, columns=dataframe.columns)
    return standarised_df
```

Rys. Z5. Kod w środowisku Python wykorzystany do standaryzacji danych

```python
mean_Y = statistics.mean(dataframe.iloc[:, 0].values)
lower_bound = mean_Y - 0.1
upper_bound = mean_Y + 0.1

aggro_breakpoint = P
```

Rys. Z6. Kod realizujący przedział cech przyśpieszenia oraz wartości granicznej.

```
temp = []
if_current_lower_than = False
for i in range(len(dataframe.index) - 1):
    current = None
    if len(temp) == 3:
        X.append(temp)
        difference = temp[-1] - temp[0]
        if abs(difference) > aggro_breakpoint:
            y.append(1)
        else:
            y.append(0)
        temp = []
        if_current_lower_than = False
    if dataframe.iloc[i, 0] < lower_bound or dataframe.iloc[i, 0] > upper_bound:
        current = dataframe.iloc[i, 0]
        if len(temp) == 1:
            if temp[0] > current:
                if_current_lower_than = True
                temp.append(dataframe.iloc[i, 0])
            else:
                temp = [dataframe.iloc[i, 0]]
                continue
        elif len(temp) > 1:
            if temp[-1] > current and if_current_lower_than is True:
                temp.append(dataframe.iloc[i, 0])
            else:
                temp = []
                if_current_lower_than = False
                temp.append(current)
        elif len(temp) == 0:
            temp.append(dataframe.iloc[i, 0])

return np.array(X), np.array(y)
```

Rys. Z7. Kod realizujący ocenę agresywności zmian wartości hamowania w grupach 3-sekundowych.

ZAŁĄCZNIK 2

Schemat blokowy algorytmu ekstrakcji cech: