

# Porównanie symulatorów ruchu miejskiego

Zuzanna Brzezińska, Agnieszka Szynalik

March 2023

## 1 Analiza problemu i dziedziny

### 1.1 Modele i systemy

Modele dzielimy ze względu na ich szczegółowość - wyróżniamy modele mikro, mezo i makroskopowe.

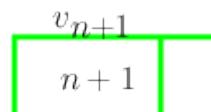
- **Modele mikroskopowe** symulują każdy pojedynczy pojazd. Ich logika zawiera algorytmy opisujące jak pojazdy poruszają się oraz jak wchodzą w interakcje z innymi obiektami. Symulacja uwzględnia pozycję, prędkość przyspieszenie i wiele innych parametrów dla każdego pojedynczego pojazdu. Takie modele pozwalają również na symulowanie innych cech otoczenia pojazdów, takich jak sygnalizacja świetlna czy piesi.
- **Modele mezoskopowe** - stanowią niejako model pośredni między mikro i makroskopowym. Symulacja jest obliczana dla grup pojazdów, jednak pojazdy mogą mieć indywidualne cechy.
- **Modele makroskopowe** - symulują przepływ ruchu. Uwzględniają średnią prędkość oraz gęstość strumienia pojazdów, bez symulowania pojedynczych obiektów.

System może być dyskretny oraz ciągły. W systemie dyskretnym update stanu odbywa się w określonych odstępach czasowych, a zmiany mają charakter dyskretny, a w systemie ciągłym zmiany zachodzą w sposób ciągły.

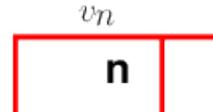
Model Car-Following - opisuje jak pojazd podąża za innym pojazdem za pomocą równań różniczkowych. Parametrami wejściowymi w tym modelu są prędkość danego pojazdu, prędkość pojazdu przed nim oraz odległość między tymi dwoma pojazdami. Na ich podstawie obliczane są położenia oraz prędkości każdego z samochodów. Przykładami takich modeli są np. model Gippsa czy model Wiedemanna.

---

## Direction of traffic

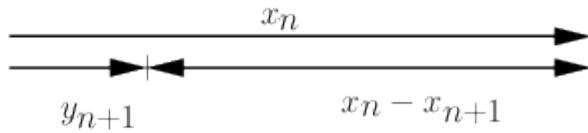


**Follower**



**Leader**

---



Rysunek 1: Car Following Model

Model Lane-Changing - symuluje zachowanie pojazdów zmieniających pas ruchu w celu uniknięcia przeszkód lub optymalizacji czasu podróży. Bierze pod uwagę takie czynniki, jak szerokość pasa ruchu, natężenie ruchu oraz prędkość i rozmiar innych pojazdów.

Model Intersection - opisuje zachowanie pojazdów i pieszych na skrzyżowaniach. Uwzględnia czas sygnalizacji świetlnej, skręty i przejścia dla pieszych.

## 1.2 Przegląd istniejących symulatorów ruchu drogowego

Nazwa	Model	System	Parametry	Cechy
MOVSIM	mikro	dyskretny i ciągły	wielopasmowość, sygnalizacja świetlna, możliwość używania różnych modeli (IDM, Gipps, Krauss, etc)	XML-based configuration, GUI, csv output
SUMO (Simulation for Urban MObility)	mikro (model Kraussa)	ciągły	wielopasmowość, sygnalizacja świetlna, piesi i rowery, VSL (Variable Speed Limit), Rerouter	model ruchu może być zdefiniowany przez użytkownika, xml-based output
AIMSUN (Advanced Interactive Microscopic Simulator for Urban and Non-Urban Networks)	mikro (oparty na Gipps)	ciągły	pojazdy mogą zmieniać wybraną trasę na podstawie natężenia ruchu, wielopasmowość, przejścia dla pieszych, VSL (VMS), różne typy dróg	GUI, animowany output 2D i 3D
Vissim	mikro (model Wiedemann)	ciągły	dużo rodzajów pojazdów	COM (Component Object Model) programming interface - umożliwia użytkownikowi implementację symulacji przy użyciu różnych języków programowania

SMARTS (Scalable Microscopic Adaptive Road Traffic Simulator)	mikro (IDM model)	dyskretny	planowanie tras pojazdów, sygnalizacja świetlna, transport publiczny, generowanie ruchu, blokowanie dróg	OSM, możliwość wizualizacji wyników
MATSim (Multi-Agent Transport Simulation)	mezo (Krauss model)	dyskretny	transport publiczny, planowanie tras pojazdów, sygnalizacja świetlna	plik konfiguracyjny XML, paczki rozszerzające zakres funkcjonalności, graficzne wyświetlanie wyników
Visum	makro	dyskretny / ciągły	różne modele (Wiedemann, Krauss, etc), użytkowanie gruntów, sygnalizacja świetlna, transport publiczny, czynniki środowiskowe	integracja GIS, analiza wydajności, optymalizacja, prognozowanie
TRANSIMS (Transportation Analysis and Simulation)	mikro (Nagel-Schreckenberg model)	dyskretny	syteza populacji, parametryzacja pojazdów, sygalizacja świetlna, transport publiczny, pogoda	wydajny, import map z różnych źródeł, wiele bibliotek, optymalizacja i prognozowanie
GAMA (Generic Architecture for Modular Agents)	mezo / mikro / makro	dyskretny	różne modele (IDM, Gipps, Krauss), pluginy, sganizacja, wielopasmowość	mapa w formacie shapfile, wykresy

## 2 Analiza i wybór narzędzi

### 2.1 SMARTS

#### 2.1.1 Konfiguracja danych wejściowych

- Import mapy OpenStreetMap z możliwością pobrania jej w symulatorze.
- Wczytanie pliku XML definiującego plan trasy dla dowolnej liczby pojazdów
- Definiowanie miejsc na mapie, w których generowane są pojazdy.
- Definiowanie celu na mapie, do którego zmierzają pojazdy.
- Dodawanie oraz usuwanie sygnalizacji świetlnej.
- Blokowanie wybranych dróg.

#### 2.1.2 Konfiguracja parametrów symulacji

- Liczba pojazdów prywatnych oraz publicznych, tj. tramwaje ibusy.
- Możliwość zmiany trasy pojazdów w trakcie trwania symulacji.
- Dystans pomiędzy pojazdami.
- Sposób kontroli sygnalizacji świetlnej.
- Algorytm generowania tras.
- Ruch prawostronny lub lewostronny.

#### 2.1.3 Wyniki symulacji

Symulator zapewnia podgląd statystyk w trakcie trwania symulacji:

- Średnia prędkość w zdefiniowanych oknach na mapie.
- Informacje na temat pojazdów: typ pojazdu, profil kierowcy, długość drogi do celu.
- Średnia prędkość na poszczególnych drogach.

Na koniec symulacji generowane są pliki wynikowe:

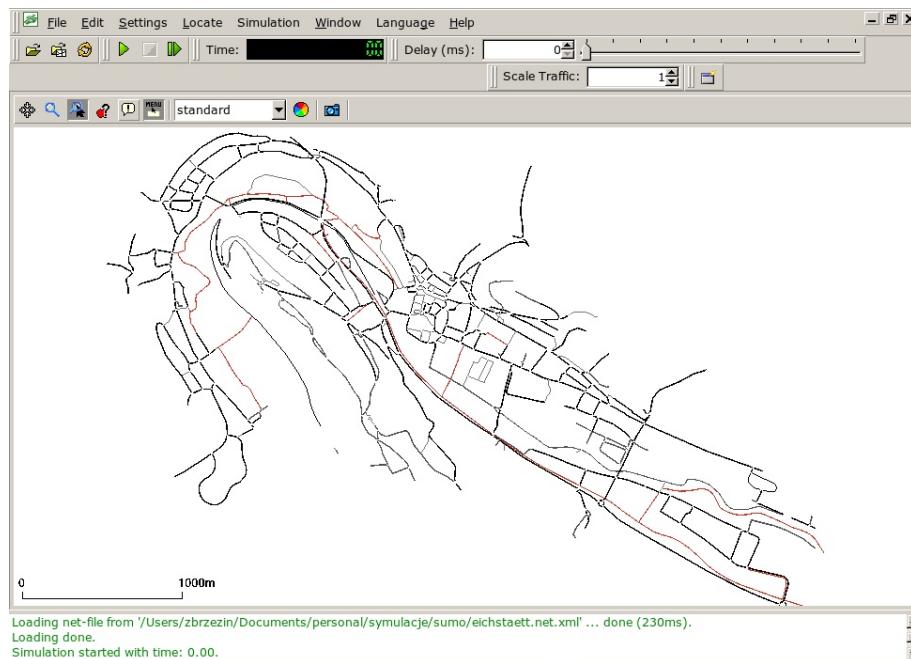
- Początkowe trasy wewnętrznie wygenerowanych losowych pojazdów. (id pojazdu, typ pojazdu, czas startu, profil kierowcy, lista node'ów)
- Czas podróży pojazdów, które dotarły do celu. (id pojazdu, czas podróży)
- Ogólne informacje, tj. czas trwania symulacji, średnia prędkość pojazdów. (timestamp, czas trwania symulacji, średnia prędkość pojazdów)
- Pozycje pojazdów w czasie. (id pojazdu, typ pojazdu, timestamp, długość i szerokość geograficzna)

## 2.2 SUMO

### 2.2.1 Konfiguracja danych wejściowych

**Open street map** - jest możliwość przetworzenia pliku .osm na plik .net.xml za pomocą polecenia netconvert

- Typemaps - są to rozszerzenia które dodają brakujące informacje, takie jak infrastruktura czy zasady panujące na określonej drodze. Przykładowe typemaps to:
  - osmNetconvertUrbanDe (dodaje typowe dla miast ograniczenia prędkości)
  - osmNetconvertPedestrians (dodaje chodniki)
  - osmNetconvertBicycle (dodaje ścieżki rowerowe)
- Opcje (flagi):
  - geometry.remove - upraszcza sieć bez zmiany topologii
  - ramps.guess - dodaje pasy rozbiegowe
  - lefthand - ustawia ruch lewostronny

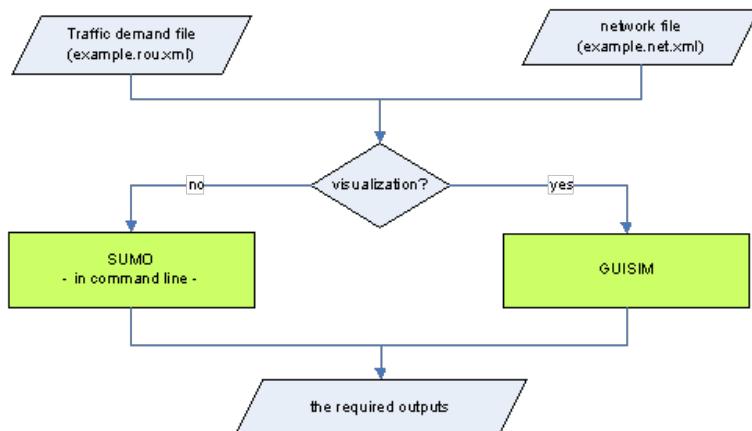


Rysunek 2: Mapa OSM zimportowana do SUMO

**GUI** - przy użyciu narzędzia NETEDIT jesteśmy w stanie tworzyć sieci i definiować ich własności oraz dodawać pojazdy wraz z ich cechami. Netedit posiada dwa główne tryby:

- Network - w ramach tego trybu możemy tworzyć nody i modyfikować połączenia między nimi, w tym:
  - definiować liczbę pasów
  - blokować wybrane drogi
  - dodawać i usuwać sygnalizację świetlną
  - tworzyć przejścia dla pieszych
  - tworzyć TAZs (Traffic Analysis Zones), czyli grafy skierowane, działające jak sieci przepływowe
- Demand - dzięki temu trybowi tworzymy pojazdy oraz definiujemy ich zachowania - wyznaczamy ich sposób poruszania się, czyli na przykład:
  - Route (trasa) - sekwencja odcinków dróg, które muszą zostać pokonane przez pojazd, aby dotrzeć z punktu A do punktu B.
  - Trip - ruch pojazdu z jednego miejsca do drugiego określony przez krawędź początkową, krawędź docelową i czas odjazdu
  - Flow - trip dotyczący wielu pojazdów

### 2.2.2 Konfiguracja symulacji



Rysunek 3: Sposoby przeprowadzania symulacji w SUMO

**Command line** - symulacja przeprowadzana jest przy użyciu komendy `sumo -c <filename>.sumocfg`, z argumentem oznaczającym plik konfiguracyjny danej symulacji.

**GUI** - umożliwia obserwację przeprowadzania symulacji w czasie rzeczywistym, symulacja może być włączana, stopowana i uruchamiana

### 2.2.3 Wyniki symulacji

Dostępne pliki wynikowe:

- Dotyczące pojazdów
  - *raw vehicle positions dump* - pozycje wszystkich pojazdów razem z prędkościami w kolejnych timestampach
  - *emission* - zanieczyszczenia powstałe podczas jazdy dla każdego samochodu
  - *FCK Output* - Floating car data - nazwy, pozycje, kąty i typy wszystkich pojazdów
  - *Trip information* - zagregowane dane dotyczące podróży każdego auta
- Dotyczące pasów/krawędzi
- Dotyczące skrzyżowań
- Dotyczące światel

## 2.3 Movsim

### 2.3.1 Konfiguracja danych wejściowych

Konfigurację sieci zawieramy w pliku Opendrive, jednak nie wszystkie cechy charakterystyczne dla tego typu pliku są zaimplementowane w symulatorze Movsim. Możemy w nim definiować takie rzeczy jak:

- drogi <road>
- pasy ruchu <lane>
- sygnalizację świetlną i znaki <signals>
- limity prędkości
- tory <railroads>

### 2.3.2 Symulacja

Do uruchomienia symulacji niezbędne są dwa pliki:

- `projectName.xprj` - ogólna specyfikacja projektu, używająca xml
- `projectName.xodr`: -specyfikacja sieci w formacie Opendrive Dodatkowo można dodać plik `projectName.properties` (w przypadku jego braku używany jest plik defaultowy).

Plik specyfikujący symulację umożliwia dodawanie układów pojazdów (`TrafficComposition`), źródeł ich wyjazdu (`TrafficSource`) oraz definiowanie outputu (`OutputConfiguration`). Dostępne jest również kilka typów modelu car-following do wyboru, min Newella, Kraussa czy Gippsa.

Symulację możemy przeprowadzić na dwa sposoby - używając Movsim Viewer oraz Movsim Core. Pierwsza wersja zapewnia GUI z możliwością blokowania i wznowiania symulacji, podglądem aktualnego stanu dla całej symulacji (liczba pojazdów, ich średnia prędkość, liczba stojących pojazdów), jak i stanu pojedynczego pojazdu (pozycja, prędkość, przyspieszenie, cel). Dostępne są również logi postaci:

```
476266 [Thread-41] DEBUG (RoadSegment.java:  
    updateVehiclePositionsAndSpeeds:964) — Vehicle [  
        id=1074, label=IDM1, length=6, frontPosition  
        =997.001, frontPositionOld=997.001, speed=0,  
        accModel=1.2, acc=0, accOld=0, vehNumber=-1, lane  
        =1, brakeLightOn=false ]
```

Z kolei Movsim Core nie zawiera strony wizualnej, jedynie logi pojawiające się w konsoli.

Wyniki symulacji zapisywane są do plików CSV, których zawartość różni się od typu outputu zdefiniowanego w pliku symulacji:

- `FloatingCarOutput` - po jednym dla każdego pojazdu, zawiera min następujące informacje:
  - położenie -  $x[m]$
  - prędkość -  $v[m/s]$
  - przyspieszenie
  - kąt pod jakim położony jest pojazd  $slope[rad]$
  - odległość od celu  $distToTL[m]$
  - id poprzedzającego pojazdu - `frontVehID`



Rysunek 4: Symulacja za pomocą Movsim Viewer

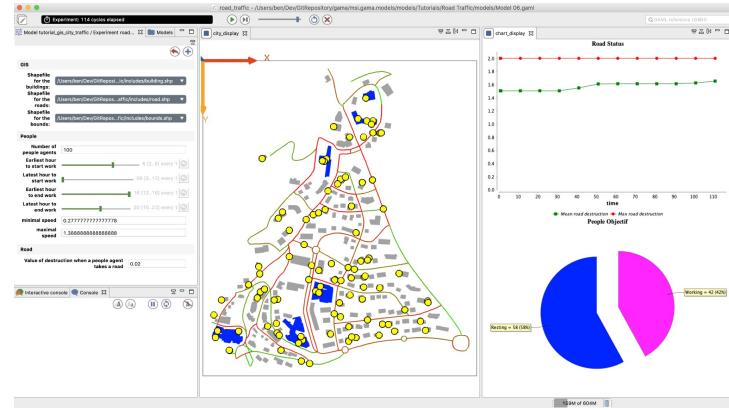
## 2.4 GAMA

### 2.4.1 Konfiguracja

- Import danych w formacie GIS podzielonych na warstwy, np. budynki i drogi.
- Konwersja pliku OpenStreetMap do pliku shapefile.
- Możliwość skorzystania z gotowych pluginów. Jednak brakuje w nich niektórych funkcjonalności, więc wymagałyby ręcznej konfiguracji modelu w celu uzyskania odpowiednich statystyk.
- Dodanie agentów zdefiniowanych przez użytkownika.
- Zamodelowanie sposobu zachowania wprowadzonych agentów.
- Ręczne ustalenie zasad działania ekseprymantu.

### 2.4.2 Wyniki symulacji

- Metryki wyświetlane na wykresach w trakcie trwania symulacji - zdefiniowane przez użytkownika w modelu, np. średnia prędkość.



Rysunek 5: Widok panelu symulacji w GAMA

### 3 Określenie celu i zakresu prac

Celem naszej pracy jest porównanie symulatorów ruchu drogowego: SMARTS oraz SUMO. Wybrane przez nas aplikacje zaliczmay do modeli mikroskopowych. Pozwalają one na symulację tych samch konstrukcji drogowych oraz wspierają funkcjonalności, tj. sygnalizacja świetlna oraz wielopasmowość. Umożliwiają także skorzystanie z tego samego typu modelu car following - IDM.

Biorąc pod uwagę możliwości, jakie dostarczają przeanalizowane przez nas rozwiązania, a przede wszystkim dane wyjściowe, zdefiniowany został zakres metryk. Między innymi porównywać będziemy *Floating Car Data/ Output*, na podstawie którego wyciągnąć można następujące informacje:

- Średnia prędkość wszystkich pojazdów na przestrzeni całej symulacji
- Liczba pojazdów w czasie na wybranym obszarze
- Liczba samochodów o średniej prędkości dla różnych przedziałów
- Liczba zatrzymanych samochodów (?)

### 4 Przeprowadzenie symulacji

Z racji, że oba symulatory wspierają import pliku OpenStreetMap, rozważanym przez nas obszarem będzie dzielnica Krakowa - Krowodrza. Mapę uzyskałyśmy z serwisu OpenStreetMap, a następnie wgrałyśmy do symulatorów (w przypadku SUMO uprzednio konieczne było przekonwertowanie jej poleceniem netconvert)



Rysunek 6: Widok mapy Krowodrzy w programie Netedit

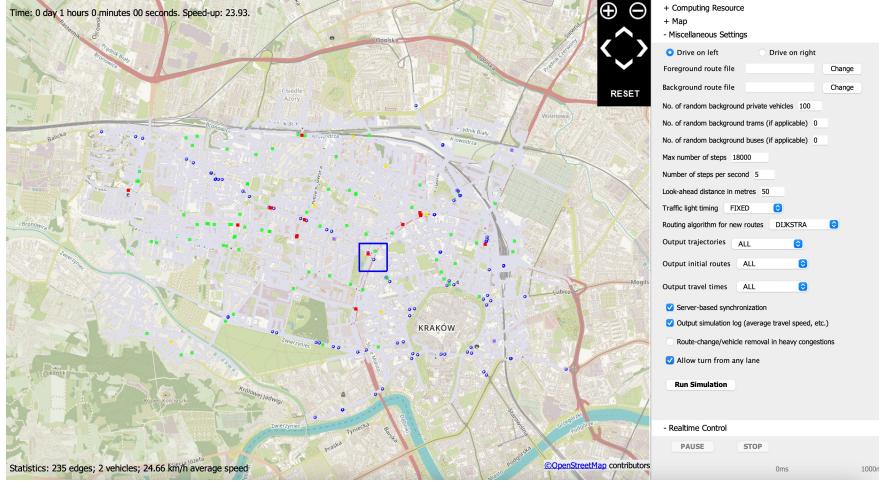
Symulować będziemy ruch na przestrzeni jednej godziny, gdzie na sekundę wykować będziemy 5 kroków. Ważnym aspektem, który należy взять pod uwagę, jest konfiguracja wejściowa. Chcemy, aby symulacja przeprowadzona była w tych samych warunkach. W idealnym scenariuszu, możliwe byłoby zdefiniowanie liczby samochodów, dla których przypisane zostaną: początek trasy, koniec trasy oraz czas wyruszenia. Jednakże symulator SMARTS pozwalał jedynie na wyznaczenie dokładnej trasy pojazdów, co oznacza, że miałyby ją odgórnie narzuconą. Alternatywnym podejściem jest stworzenie punktów startowych oraz docelowych. Ma to jednak swoje wady, np. brak kontroli nad celem podróży poszczególnych pojazdów.

Ze względu na ograniczenia, jakie narzuca przede wszystkim symulator SMARTS, zdecydowałyśmy się na scenariusz, w którym wybrana liczba pojazdów będzie poruszać się ruchem prawostronnym w sposób losowy na przestrzeni całej mapy. Dodatkowo uwzględnimy także rowerzystów.

#### 4.1 SMARTS

Odpowiednie parametry symulacji ustawione zostały w GUI aplikacji. Dostosowana została między innymi liczba kroków tak, aby symulacja trwała godzinę czasu rzeczywistego. W celu wygenerowania odpowiedniego ruchu w trakcie jednej godziny, dostosowano także parametr odpowiadający za liczbę pojazdów prywatnych. Wygenerowane w ten sposób pojazdy zaliczyć możemy do samochodów oraz rowerów. Warto także wspomnieć, że w przypadku dużego ruchu,

pojazdy mają możliwość zmiany trasy.



Rysunek 7: Widok mapy Krowodrzy w programie SMARTS wraz z przykładowymi ustawieniami

## 4.2 SUMO

Do przeprowadzenia symulacji w programie SUMO niezbędne było uzyskanie trzech plików wejściowych:

- Network (o rozszerzeniu .net.xml) - zawierającego przekonwertowaną mapę Krowodrzy
- Route/Demand (.rou.xml) - zawierającego informacje o pojazdach i ich sposobach poruszania się
- Config (.sumocfg) - zawierającego ścieżki do obu poprzednich plików

Do przeprowadzenia symulacji zgodnie z naszymi założeniami kluczowy był plik Route, który posiada następującą strukturę:

```
<routes>
  <vType/>
  ...
  <flow/>
</routes>
```

Tag vType pozwala definiować pojazdy poruszające się w ramach symulacji. W naszym przypadku są to:

- Samochody osobowe

```
<vType id="car" length="5.00" maxSpeed="70.00"
      speedFactor="normc(1.00,0.20,0.20,2.00)" vClass="passenger"
      accel="2.6" decel="4.5" sigma="0.2" carFollowModel="IDM"/>
```

- Rowery

```
<vType id="bike" length="1.80" maxSpeed="7.50"
speedFactor="normc(1.00,0.50,0.20,2.00)" vClass="bicycle" width="0.80"
accel="0.8" decel="1.5" sigma="0.5" carFollowModel="IDM"/>
```

- Tramwaje

```
<vType id="tram" length="40.00" maxSpeed="30.00"
speedFactor="normc(1.00,0.10,0.20,2.00)" vClass="rail_urban"
accel="0.8" decel="0.5" sigma="0.1" carFollowModel="IDM"/>
```

We wszystkich pojazdach parametr carFollowModel został ustawiony na wartość "IDM", ponieważ taki model jest używany w symulatorze SMARTS. Długości i maksymalne prędkości pojazdów zostały dopasowane, aby odzwierciedlały rzeczywiste wartości.

Tag flow został skonstruowany w następujący sposób:

```
<flow id="f_4" begin="0.00" from="-126376304#1" to="365151260#3"
via="277156308" end="86400.00" vehsPerHour="1800.00"/>
```

## 5 Porównanie wyników

### 5.1 SUMO

#### 5.1.1 Liczba wygenerowanych pojazdów

- Wyniki dla 50 pojazów:

- Samochody: 460
- Rowery: 33

- Wyniki dla 100 pojazów:

- Samochody: 892
- Rowery: 53

- Wyniki dla 200 pojazów:

- Samochody: 1487
- Rowery: 90

- Wyniki dla 500 pojazów:

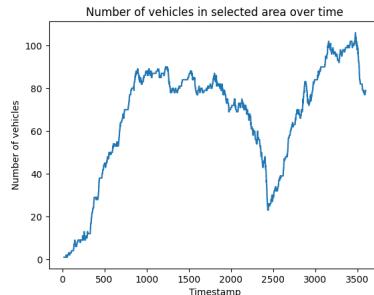
- Samochody: 2947
- Rowery: 180

### 5.1.2 Analiza średniej prędkości na przestrzeni całej symulacji

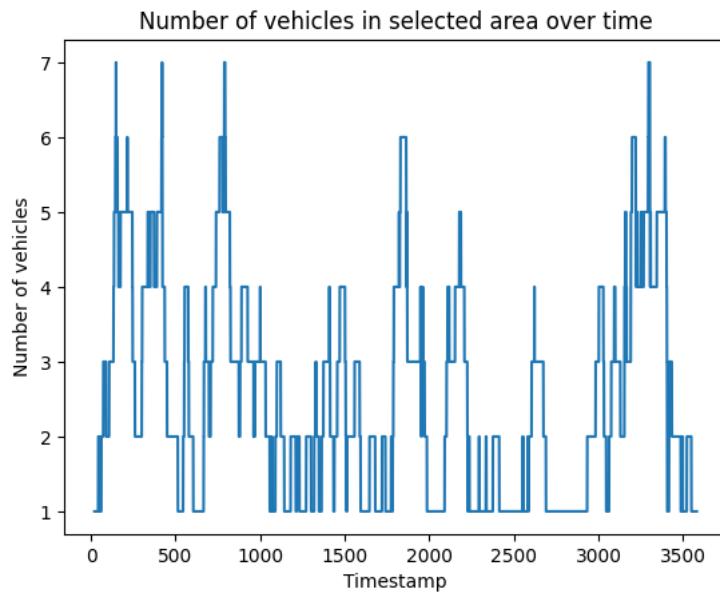
- Wyniki dla 50 pojazów:
  - Samochody: 28.162557311390902
  - Rowery: 16.009694210046515
- Wyniki dla 100 pojazów:
  - Samochody: 26.963830956127094
  - Rowery: 15.429064271935468
- Wyniki dla 200 pojazów: 25.04540030786246
  - Samochody: 22.705233257083794
  - Rowery: 14.454949230887962
- Wyniki dla 500 pojazów:
  - Samochody: 19.755821037458027
  - Rowery: 13.883045841419813

### 5.1.3 Analiza liczby pojazdów na wybranym obszarze

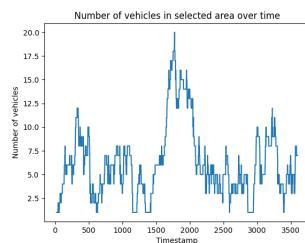
- Wyniki dla 50 pojazów:
- Wyniki dla 100 pojazów:
- Wyniki dla 200 pojazów:
- Wyniki dla 500 pojazów:



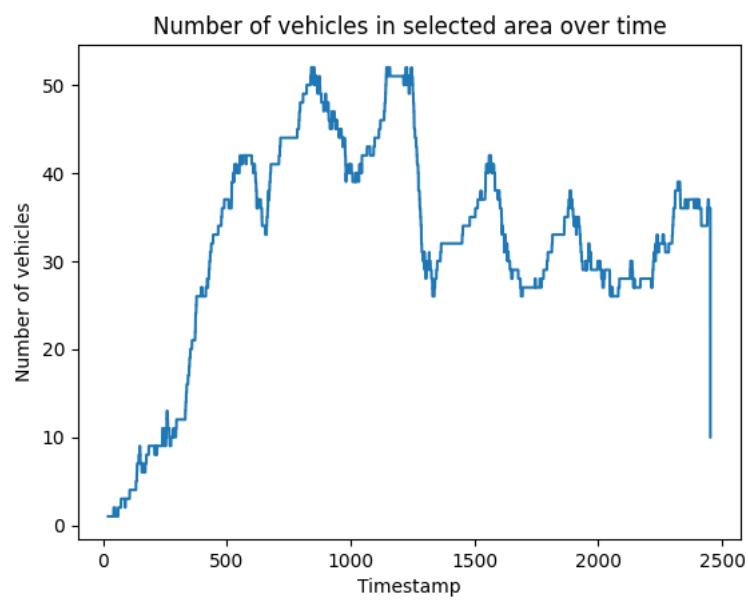
Rysunek 11: Wykres liczby pojazdów na obszarze [50.06870, 50.071445, 19.900597, 19.90811] w czasie dla symulacji w SUMO, gdzie liczba pojazdów wynosi 500



Rysunek 8: Wykres liczby pojazdów na obszarze [50.06870, 50.071445, 19.900597, 19.90811] w czasie dla symulacji w SUMO, gdzie liczba pojazdów wynosi 50



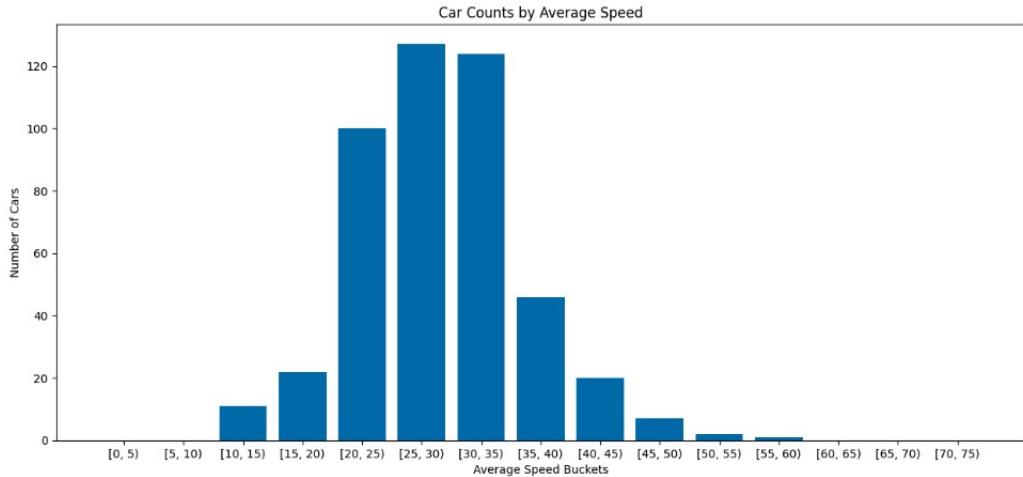
Rysunek 9: Wykres liczby pojazdów na obszarze [50.06870, 50.071445, 19.900597, 19.90811] w czasie dla symulacji w SUMO, gdzie liczba pojazdów wynosi 100



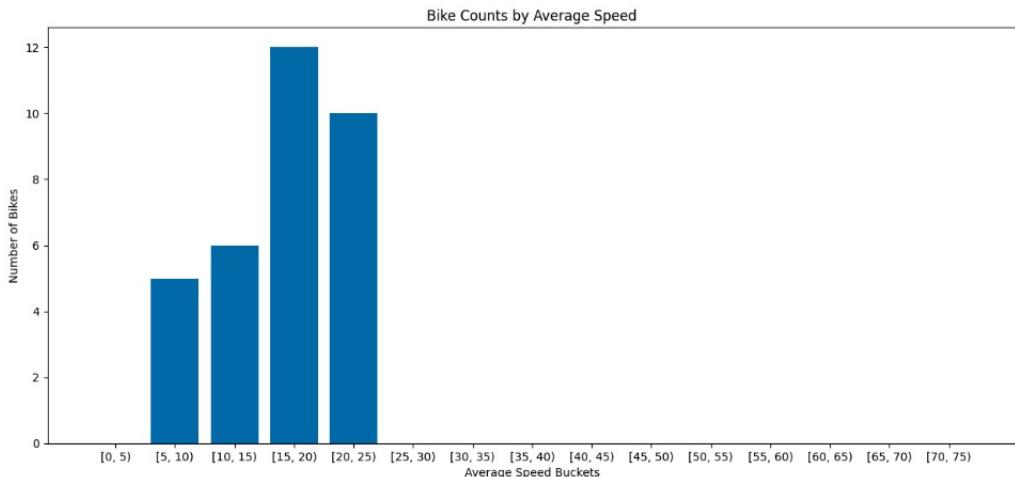
Rysunek 10: Wykres liczby pojazdów na obszarze [50.06870, 50.071445, 19.900597, 19.90811] w czasie dla symulacji w SUMO, gdzie liczba pojazdów wynosi 200

#### 5.1.4 Analiza średniej prędkości dla różnych przedziałów

- Wyniki dla 50 pojazdów:

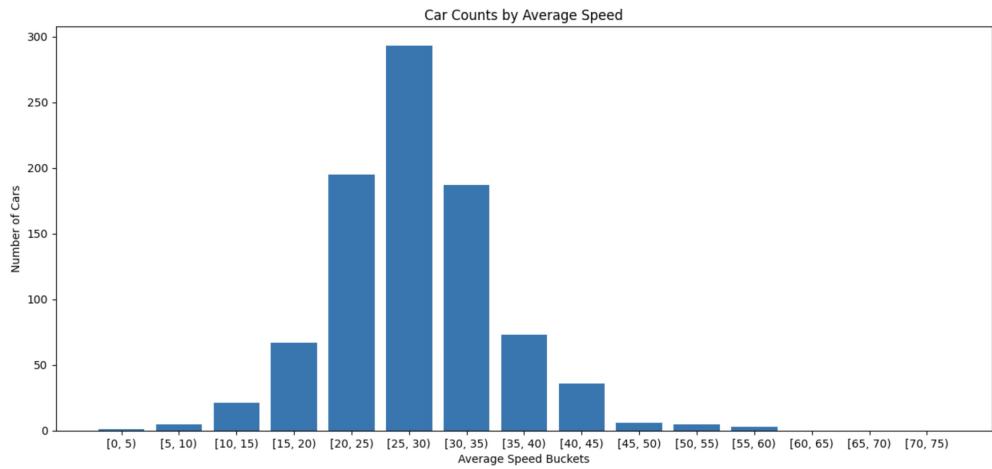


Rysunek 12: Histogramy średniej prędkości samochodów dla symulacji w SUMO, gdzie liczba pojazdów wynosi 50

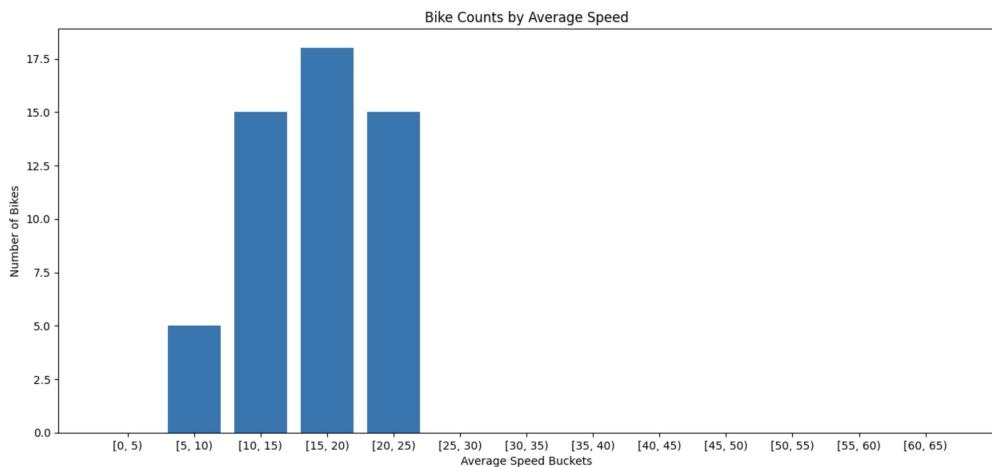


Rysunek 13: Histogramy średniej prędkości rowerów dla symulacji w SUMO, gdzie liczba pojazdów wynosi 50

- Wyniki dla 100 pojazdów:

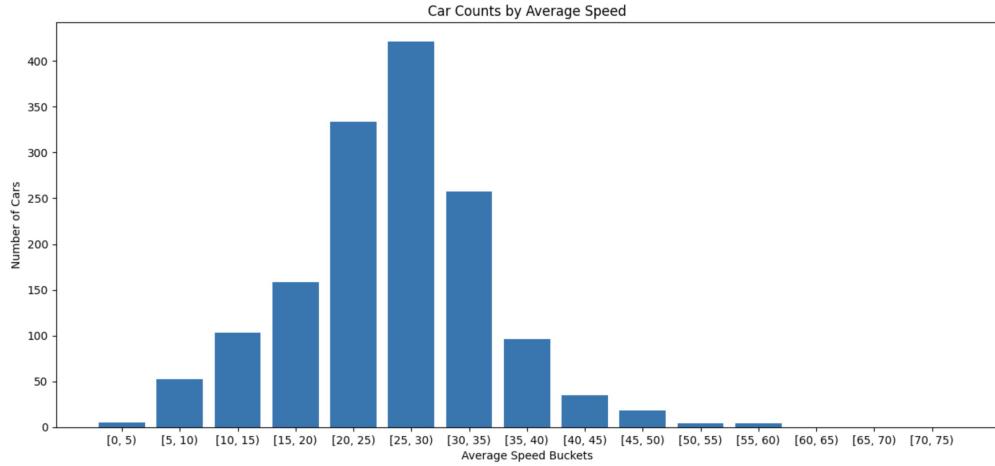


Rysunek 14: Histogramy średniej prędkości samochodów dla symulacji w SUMO, gdzie liczba pojazdów wynosi 100

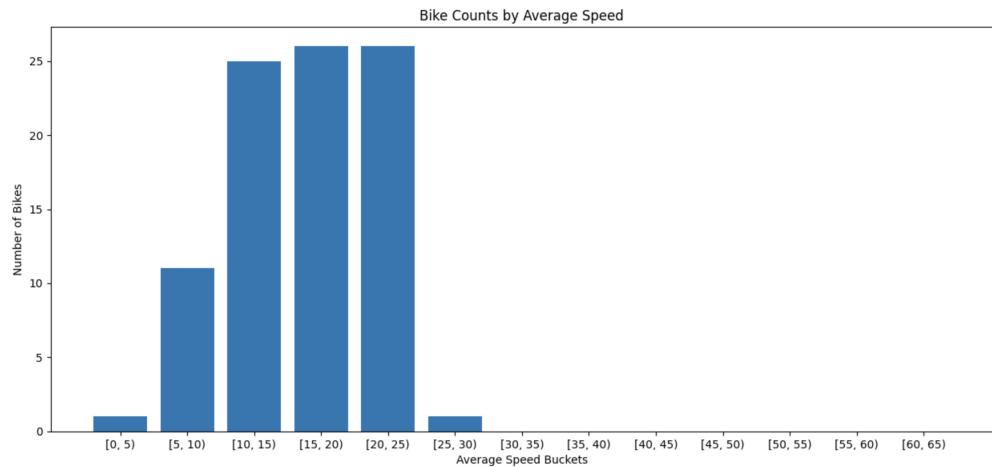


Rysunek 15: Histogramy średniej prędkości rowerów dla symulacji w SUMO, gdzie liczba pojazdów wynosi 100

- Wyniki dla 200 pojazdów:



Rysunek 16: Histogramy średniej prędkości samochodów dla symulacji w SUMO, gdzie liczba pojazdów wynosi 200

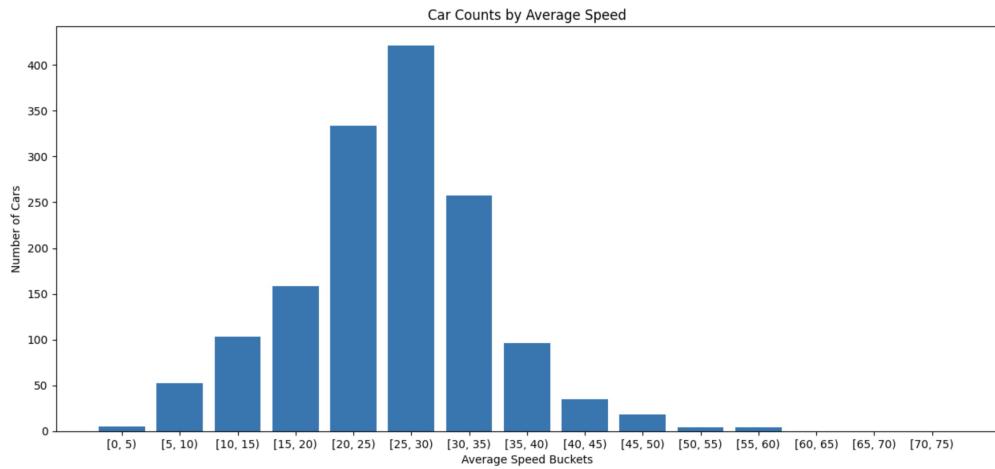


Rysunek 17: Histogramy średniej prędkości rowerów dla symulacji w SUMO, gdzie liczba pojazdów wynosi 200

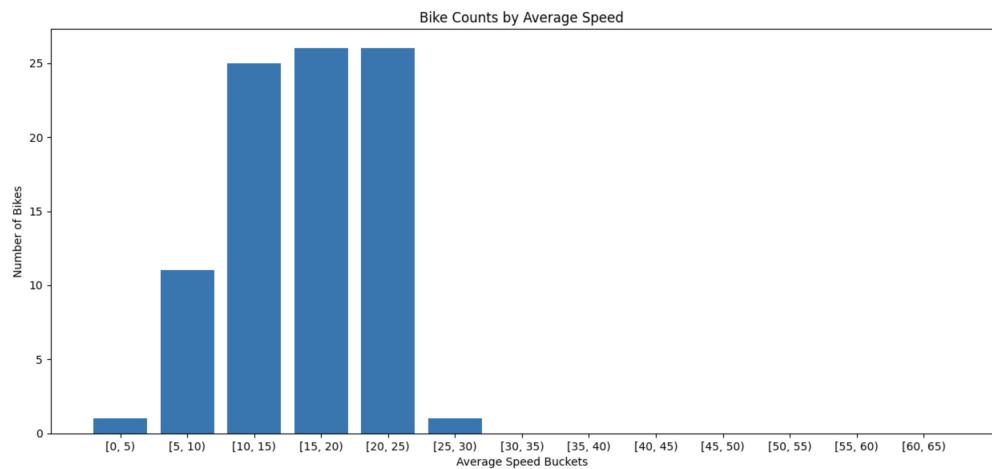
- Wyniki dla 500 pojazdów:

### 5.1.5 Analiza średniej prędkości dla różnych typów pojazdów

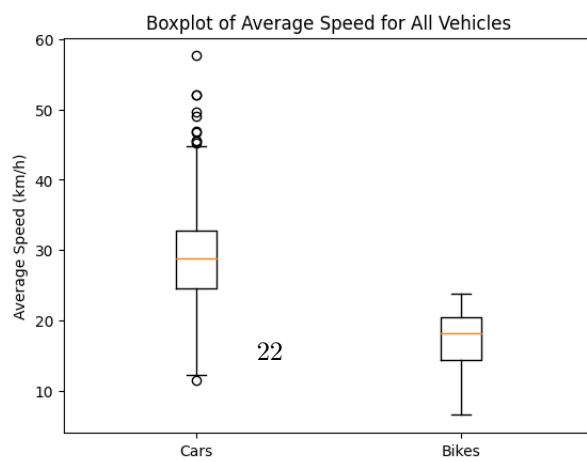
- Wyniki dla 50 pojazdów:



Rysunek 18: Histogramy średniej prędkości samochodów dla symulacji w SUMO, gdzie liczba pojazdów wynosi 500

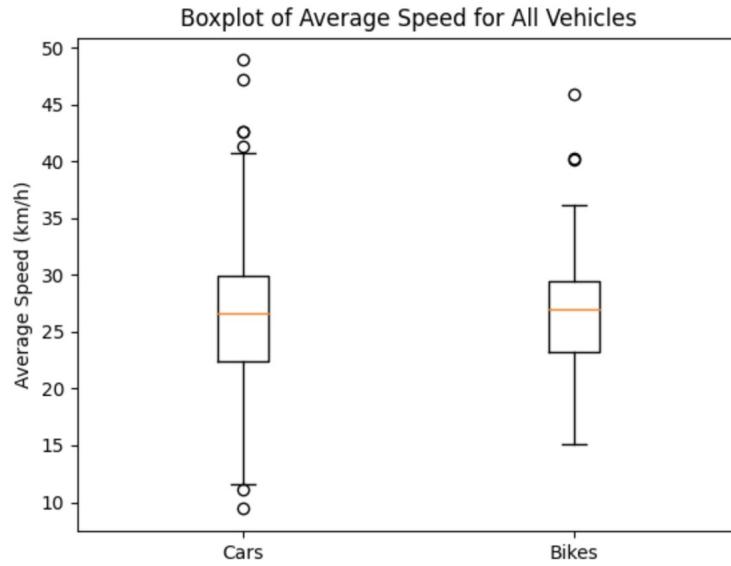


Rysunek 19: Histogramy średniej prędkości rowerów dla symulacji w SUMO, gdzie liczba pojazdów wynosi 500



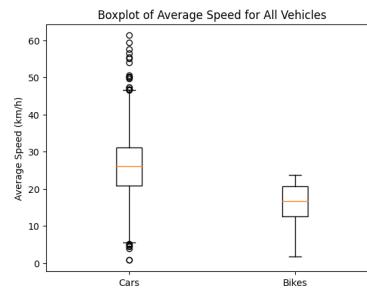
Rysunek 20: Wykres pudełkowy średniej prędkości wybranych typów pojazdów dla symulacji w SUMO, gdzie liczba pojazdów wynosi 50

- Wyniki dla 100 pojazdów:



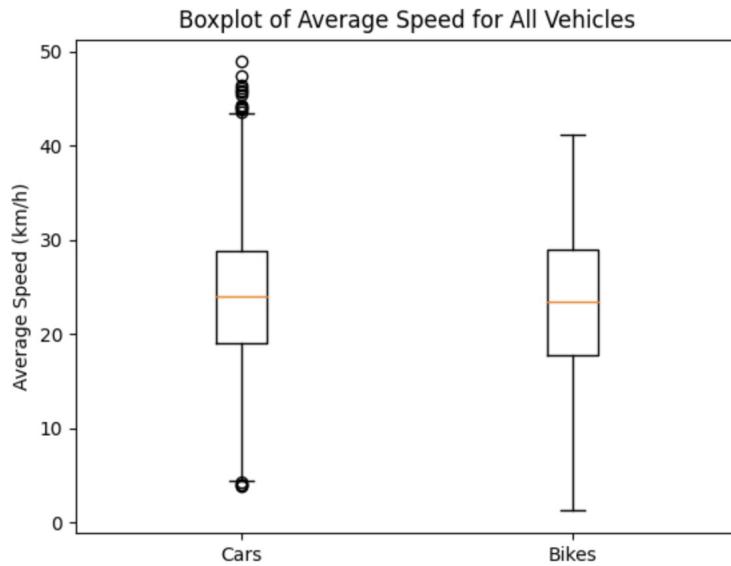
Rysunek 21: Wykres pudełkowy średniej prędkości wybranych typów pojazdów dla symulacji w SUMO, gdzie liczba pojazdów wynosi 100

- Wyniki dla 200 pojazdów:



Rysunek 22: Wykres pudełkowy średniej prędkości wybranych typów pojazdów dla symulacji w SUMO, gdzie liczba pojazdów wynosi 200

- Wyniki dla 500 pojazdów:



Rysunek 23: Wykres pudelkowy średniej prędkości wybranych typów pojazdów dla symulacji w SMARTS, gdzie liczba pojazdów wynosi 500

## 5.2 SMARTS

### 5.2.1 Liczba wygenerowanych pojazdów

- Wyniki dla 50 pojazów:

- Samochody: 458
- Rowery: 27

- Wyniki dla 100 pojazów:

- Samochody: 877
- Rowery: 52

- Wyniki dla 200 pojazów:

- Samochody: 1782
- Rowery: 77

- Wyniki dla 500 pojazów:

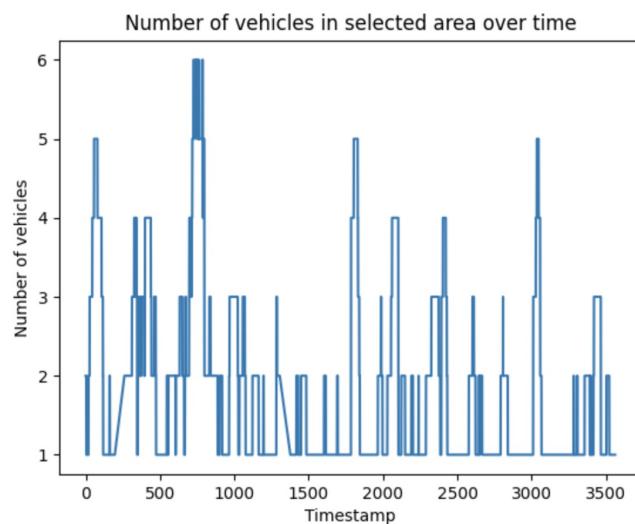
- Samochody: 3714
- Rowery: 201

### 5.2.2 Analiza średniej prędkości na przestrzeni całej symulacji

- Wyniki dla 50 pojazów: 25.91238583605483
- Wyniki dla 100 pojazów: 25.639722827730413
- Wyniki dla 200 pojazów: 25.04540030786246
- Wyniki dla 500 pojazów: 21.12746199911813

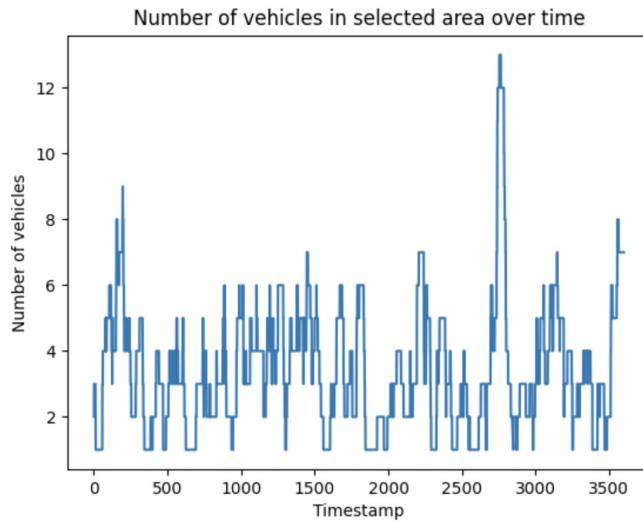
### 5.2.3 Analiza liczby pojazdów na wybranym obszarze

- Wyniki dla 50 pojazów:



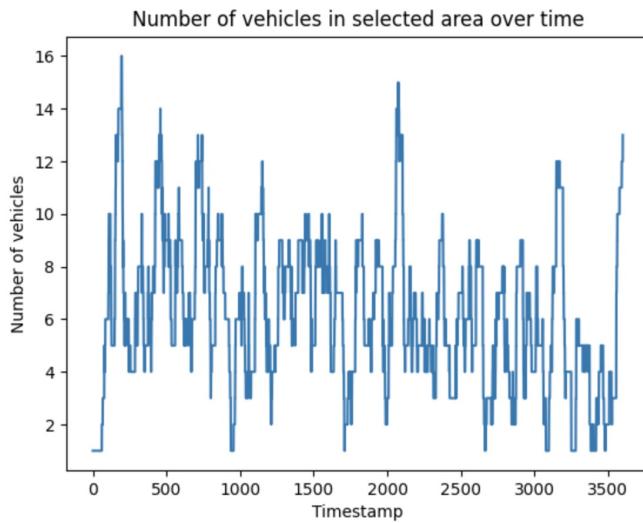
Rysunek 24: Wykres liczby pojazdów na obszarze [50.06870, 50.071445, 19.900597, 19.90811] w czasie dla symulacji w SMARTS, gdzie liczba pojazdów wynosi 50

- Wyniki dla 100 pojazów:



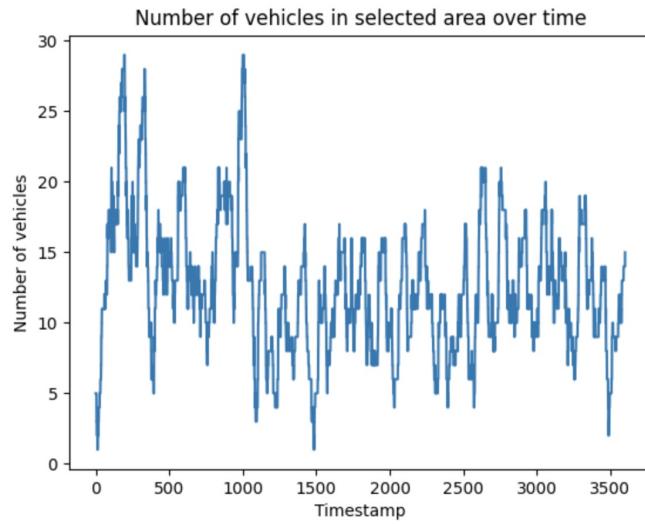
Rysunek 25: Wykres liczby pojazdów na obszarze  $[50.06870, 50.071445, 19.900597, 19.90811]$  w czasie dla symulacji w SMARTS, gdzie liczba pojazdów wynosi 100

- Wyniki dla 200 pojazdów:



Rysunek 26: Wykres liczby pojazdów na obszarze  $[50.06870, 50.071445, 19.900597, 19.90811]$  w czasie dla symulacji w SMARTS, gdzie liczba pojazdów wynosi 200

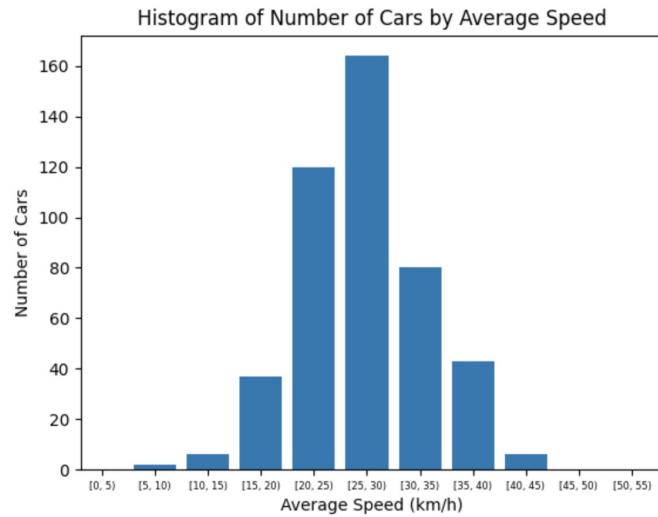
- Wyniki dla 500 pojazdów:



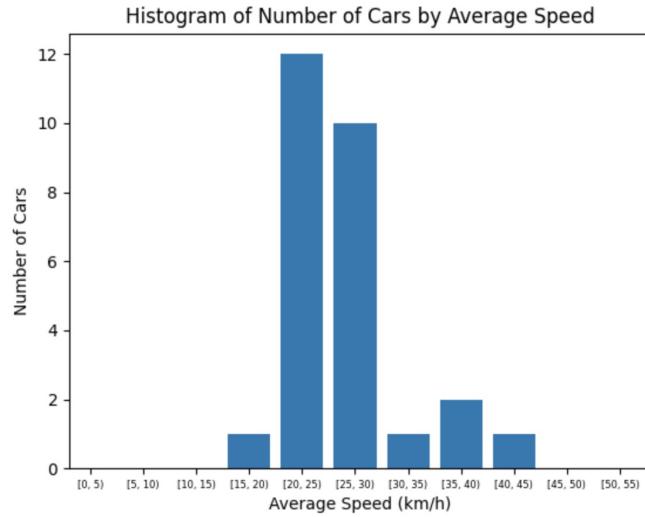
Rysunek 27: Wykres liczby pojazdów na obszarze [50.06870, 50.071445, 19.900597, 19.90811] w czasie dla symulacji w SMARTS, gdzie liczba pojazdów wynosi 500

#### 5.2.4 Analiza średniej prędkości dla różnych przedziałów

- Wyniki dla 50 pojazdów:

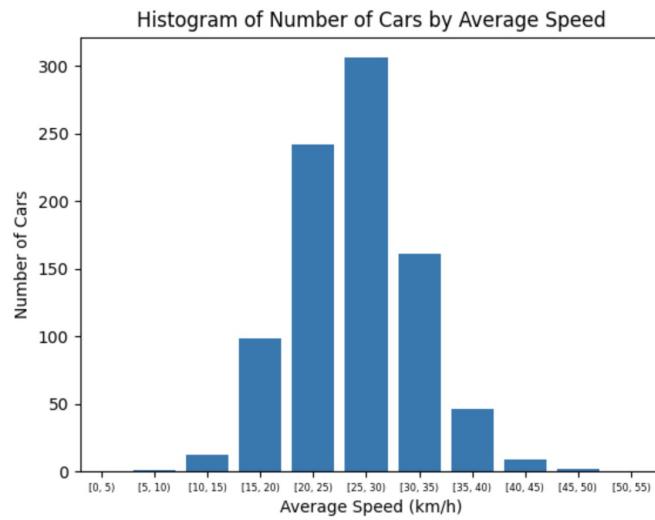


Rysunek 28: Histogram średniej prędkości wszystkich samochodów dla symulacji w SMARTS, gdzie liczba pojazdów wynosi 50

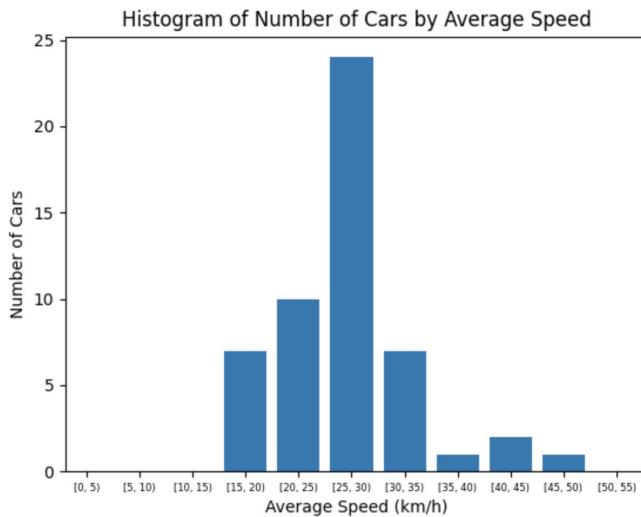


Rysunek 29: Histogram średniej prędkości wszystkich rowerów dla symulacji w SMARTS, gdzie liczba pojazdów wynosi 50

- Wyniki dla 100 pojazdów:

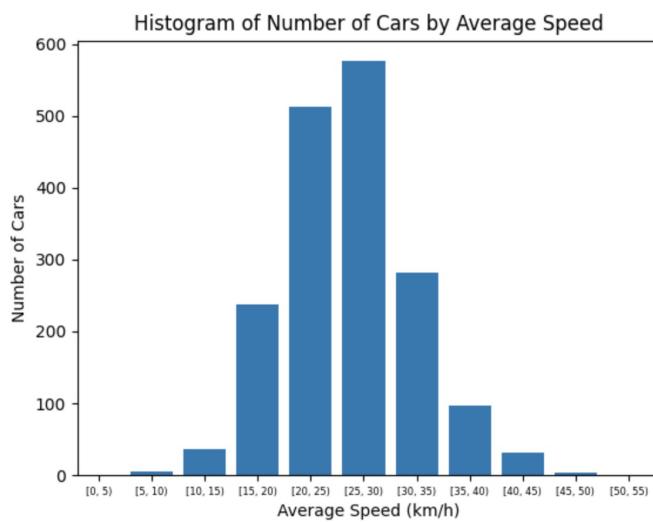


Rysunek 30: Histogram średniej prędkości wszystkich samochodów dla symulacji w SMARTS, gdzie liczba pojazdów wynosi 100

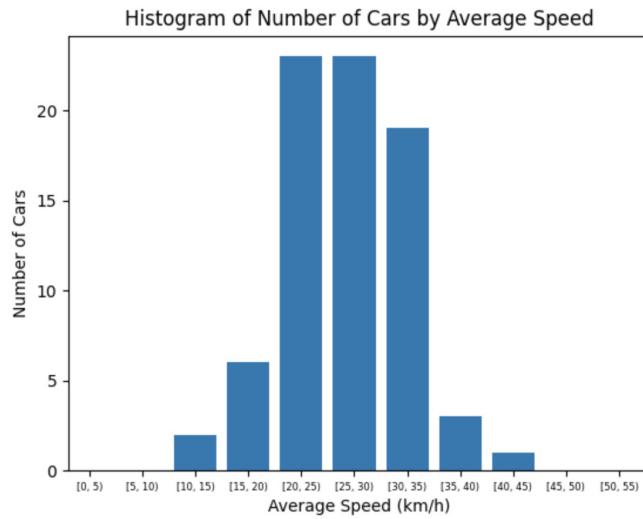


Rysunek 31: Histogram średniej prędkości wszystkich rowerów dla symulacji w SMARTS, gdzie liczba pojazdów wynosi 100

- Wyniki dla 200 pojazdów:

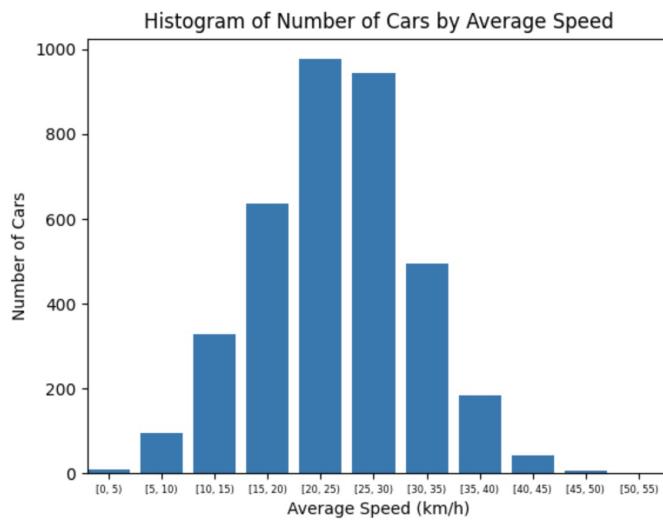


Rysunek 32: Histogram średniej prędkości wszystkich samochodów dla symulacji w SMARTS, gdzie liczba pojazdów wynosi 200

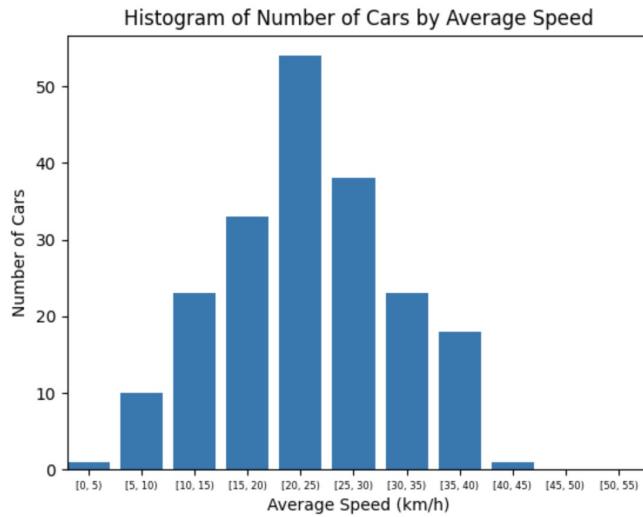


Rysunek 33: Histogram średniej prędkości wszystkich rowerów dla symulacji w SMARTS, gdzie liczba pojazdów wynosi 200

- Wyniki dla 500 pojazdów:



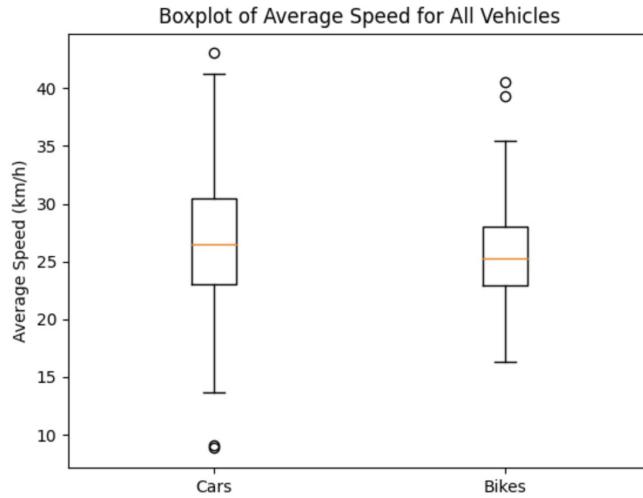
Rysunek 34: Histogram średniej prędkości wszystkich samochodów dla symulacji w SMARTS, gdzie liczba pojazdów wynosi 500



Rysunek 35: Histogram średniej prędkości wszystkich rowerów dla symulacji w SMARTS, gdzie liczba pojazdów wynosi 500

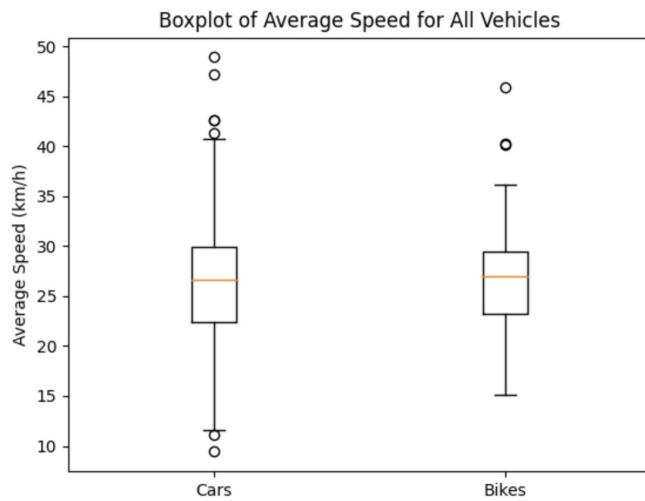
#### 5.2.5 Analiza średniej prędkości dla różnych typów pojazdów

- Wyniki dla 50 pojazdów:



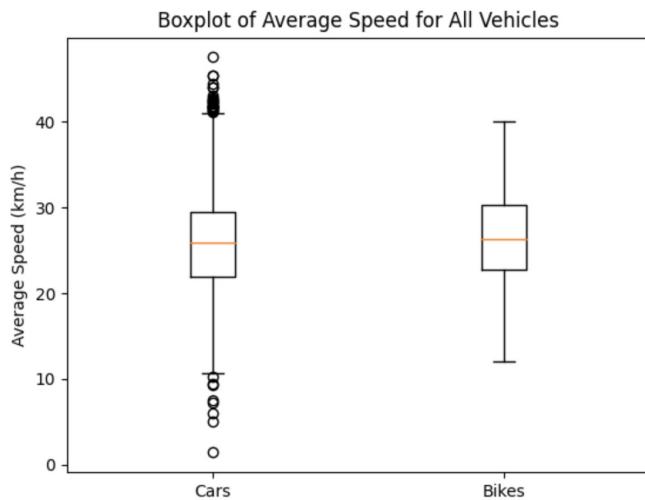
Rysunek 36: Wykres pułapkowy średniej prędkości wybranych typów pojazdów dla symulacji w SMARTS, gdzie liczba pojazdów wynosi 50

- Wyniki dla 100 pojazdów:



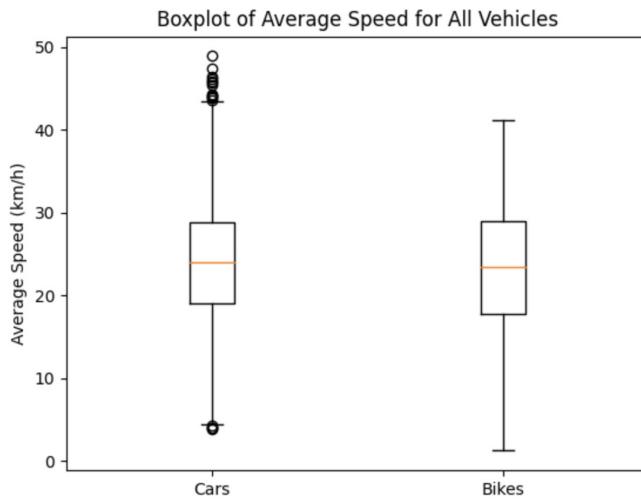
Rysunek 37: Wykres pudełkowy średniej prędkości wybranych typów pojazdów dla symulacji w SMARTS, gdzie liczba pojazdów wynosi 100

- Wyniki dla 200 pojazdów:



Rysunek 38: Wykres pudełkowy średniej prędkości wybranych typów pojazdów dla symulacji w SMARTS, gdzie liczba pojazdów wynosi 200

- Wyniki dla 500 pojazdów:



Rysunek 39: Wykres pułapkowy średniej prędkości wybranych typów pojazdów dla symulacji w SMARTS, gdzie liczba pojazdów wynosi 500

## 6 Wnioski

## 7 Bibliografia

- Simulation Approaches in Transportation Analysis 2005 - Jaume Barcelo, Jordi Casas
- CAR-FOLLOWING MODELS. COMPARISON BETWEEN MODELS USED BY VISSIM AND AIMSUN - Ionuț-Sorin MITROI, Ana-Maria CIOBÎCĂ, Mihaela POPA
- Car Following Models - Lecture Notes in Transportation Systems Engineering - Prof. Tom V. Mathew
- A Comparative Study of Urban Road Traffic Simulators - Mustapha Sadallah, Abdeslam El Fergougui and Abdelbaki Elbelrhiti Elalaoui 2016
- SMARTS: Scalable Microscopic Adaptive Road Traffic Simulator - Kotagiri Ramamohanarao, Hairuo Xie, Lars Kulik, Shanika Karunasekera, Egemen Tanin, Rui Zhang, Eman Bin Khunayn
- Traffic simulation with the GAMA platform - Patrick Taillandier