Akademia Górniczo-Hutnicza
im. Stanisława Staszica w Krakowie

Wydział Elektrotechniki, Automatyki, Informatyki i Elektroniki

Symulacja ruchu samochodów na I Obwodnicy Krakowa

Oskar Kocjan
Maciej Koch

Spis treści

1 Wprowadzenie

1.1 Opis problemu

W wielu miastach codziennie możemy spotkać się z dużym ruchem na drogach publicznych co niestety w przypadku większych miejscowości skutkuje korkami, częstymi wypadkami, wstrzymaniami komunikacji miejskiej. Innymi powodami wyżej wymienionych sytuacji mogą być również planowane przez urzędy miast plany rozbudowy infrastruktury czy też samych dróg.Przeprowadzanie skutecznych i realistycznych symulacji pozwala na lepsze rozwiązanie tych problemów.

1.2 Cel naszej symulacji

W naszym projekcie zajmiemy się I obwodnicą Krakowa. Głównym celem będzie przeprowadzenie takiej symulacji, aby w jak najbardziej realistyczny sposób zasymulować dynamikę ruchu miejskiego. Przedstawić ruch samochodów jednoosobowych oraz ciężarowych. Odzwierciedlimy faktyczny układ dróg z uwzględnieniem najważniejszych skrzyżowań, dróg jednokierunkowych, wielopasmowych oraz węzłów z sygnalizacją świetlną. W tym celu posłużymy się modelem Nagela-Schreckenberga - klasyczny sposób oparty na automatach komórkowych , który służy do opisu ruchu samochodów. Uwzględnimy specyfikę pojazdów oraz ich ruch w oparciu o automaty komórkowe. Określimy sąsiedztwo (w ujęciu Moore'a), przyśpieszenie, hamowanie, losową zmianę prędkości, regułę prawej ręki.

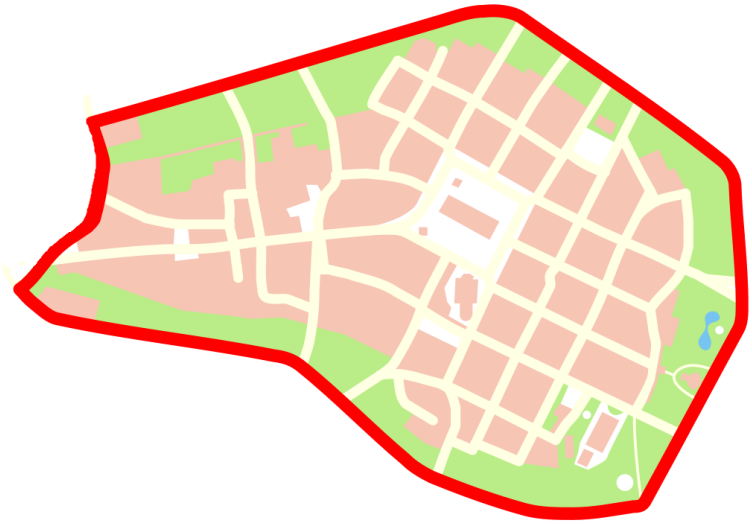
1.3 Podstawowy model Na-Sch

W 1992 r. Nagel i Schreckenberg przedstawili model - automat komórkowy opisujący ruch pojazdów. Założenia modelu to:

- podział ulicy na komórki długości 7.5m
- droga jest jednokierunkowa jedno pasowa
- komórka może być zajęta przez jeden pojazd lub wolna
- każdej komórce przypisana jest wartość prędkości
- ruch pojazdów w tym modelu opisują 4 pojęcia:
    1. przyśpieszanie - pojazd zwiększa swoja prędkość
    2. hamowanie - zmniejszenie szybkości w komórce poprzedzającej znajduje się pojazd
    3. hamowanie - na drodze bywają zdarzenia losowe np. wtargnięcie pieszego na jezdnię
    4. przesunięcie - o wykonaniu powyższych czynności następuje przesunięcie wszystkich pojazdów do następnej komórki i zwiększenie zmiennej czasu

1.4 I Obwodnica Krakowa

Mianem pierwszej obwodnicy określa się w Krakowie ciąg ulic otaczających Stare Miasto wzdłuż plant. Ruch na obszarze wewnątrz pierwszej obwodnicy jest wysoce ograniczony



W skład obwodnicy wchodzi niżej przedstawiony wykaz ulic (zgodnie z ruchem wskazówek zegara) :

- ul. św. Idziego (ograniczenie ruchu w jednym kierunku)
- ul. Podzamcze (ograniczenie ruchu w jednym kierunku)
- ul. F. Straszewskiego (ruch jednokierunkowy na odcinku ulicy)
- ul. Podwale (ograniczenie ruchu na odcinku ulicy i ruch jednokierunkowy na całości)
- ul. J. Dunajewskiego (ograniczenie ruchu i ruch jednokierunkowy)
- ul. Basztowa (ograniczenie ruchu na odcinku i ruch jednokierunkowy na całej ulicy)
- ul. Westerplatte
- ul. św. Gertrudy

2 Analiza modelu oraz modyfikacje podstawowego modelu Na-Sch

2.1 Automaty komórkowe

Automaty komórkowe (ang. cellular automata) mogą być prezentowane w postaci czwórki $A = (\alpha, S, N, f)$:

α - definiuje regularną, uporządkowaną siatkę złożoną z jednakowych komórek

S - skończony zbiór stanów, jaki może przyjąć komórka

N - zbiór sąsiadów

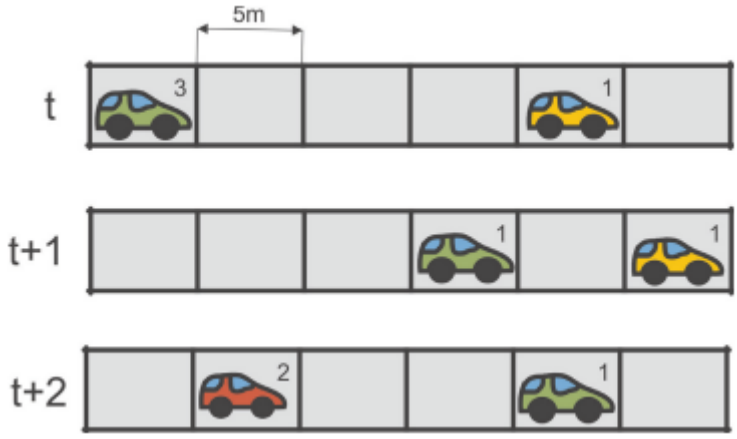
f - funkcja przejścia która definiuje ewolucje automatu w kolejnych jednostkach czasu

W modelu Nagel–Schreckenberga droga jest podzielona na komórki. Każda z nich może być w dwóch stanach tj. albo pełna (czyli zawiera jeden samochód) lub pusta (nie zawiera wcale). Każdy pojazd ma przypisaną prędkość od 0 do V_{max} ustalonego gdzie ta wartość jest liczbą całkowitą. Informuje ona o liczbie komórek, o które przesunie się pojazd w kolejnym kroku. Jest to model symulacyjny autostrady opracowany do badania warunków spontanicznego powstawania korków, jednak idealnie nada się on również do opisu ruchu miejskiego.W naszym projekcie ustaliliśmy trzy rodzaje prędkości, gdyż zakładamy, iż samochody nie przekraczają pewnej granicy tym bardziej na I obwodnicy, gdzie ruch jest i tak z powodów oczywistych utrudniony. Dodatkowo w naszym przypadku, komórki maja długość 5m.

2.2 Reguły poruszania pojazdów

W każdym dyskretnym kroku dla wszystkich pojazdów są wykonywane równolegle kolejno następujące reguły poruszania się:

- przyśpieszenie $v(t+1) \rightarrow min(v(t)+1, v_{max})$, gdzie $v(t)$ to prędkość aktualna pojazdu
- hamowanie $v(t+1) \rightarrow min(v(t), g(t)-1)$, gdzie $g(t)$ jest liczbą pustych komórek między pojazdami
- element losowy, prawdopodobieństwo p, że zajdzie $v(t+1) \rightarrow max(v(t)-1)$, jeżeli $v(t) \geqslant 1$
- ruch (zmiana położenia w czasie) $x(t+1) = x(t) + v(t)$



Rysunek 1: Ruch modelu Nagela-Schreckenberga na pasie ruchu kolejnych chwilach czasowych. W lewym górnym rogu każdej komórki można zauważyć aktualną prędkość pojazdu podaną w komórkach na chwilę czasową.

Sygnalizacja świetlna

W naszym projekcie chcemy zmodyfikować model poprzez dodanie dodatkowych parametrów w postaci sygnalizacji świetlnej co wpłynie na element ruchu:

jeżeli: $s(t) = 1$ i $x_i(t) < x_s$, to $v_i(t) \leftarrow min(v_i(t), x_s - x_i(t) - 1)$
gdzie:

$s(t)$ - oznacza sygnał wyświetlany dla pojazdów w chwili t
($s(t) = 1$, gdy wyświetlany jest sygnał czerwony)

x_s - jest numerem komórki, która znajduje się na modelowanym wlocie skrzyżowania, bezpośrednio za linia warunkowego zatrzymania

Reguła prawej ręki

Kolejną modyfikacją podstawowego modelu będzie dodanie reguły prawej ręki, która będzie działać w następujący sposób dla danego skrzyżowania:

jeżeli $(\ x(t) = p \in A\ ) \wedge (\ \forall\ z \in B\ )$, jeżeli $\exists\ z_n = 1 \Rightarrow v(t) = 0$
gdzie:

A - oznacza zbiór punktów (komórek) linii warunkowego zatrzymania na skrzyżowaniu w przypadku świateł, bądź ustąpienia pierwszeństwa.

B - oznacza zbiór punktów (komórka) *otoczenia* skrzyżowania, co w praktyce oznacza punkty skrzyżowania oraz punkty odcinka pasa ruchu na którym poruszają się samochody, którym być może będzie należało ustąpić pierwszeństwa. Każdy element tego zbioru może przyjmować wartość 0 lub 1 co oznacza kolejno iż punkt jest wolny lub zajęty.

3 Wybór technologii

Do stworzenia naszego projektu, wykorzystaliśmy następujące technologie:

- Środowisko PyCharm
- OpenStreetMap
- Oprogramowanie JOSM
- Github

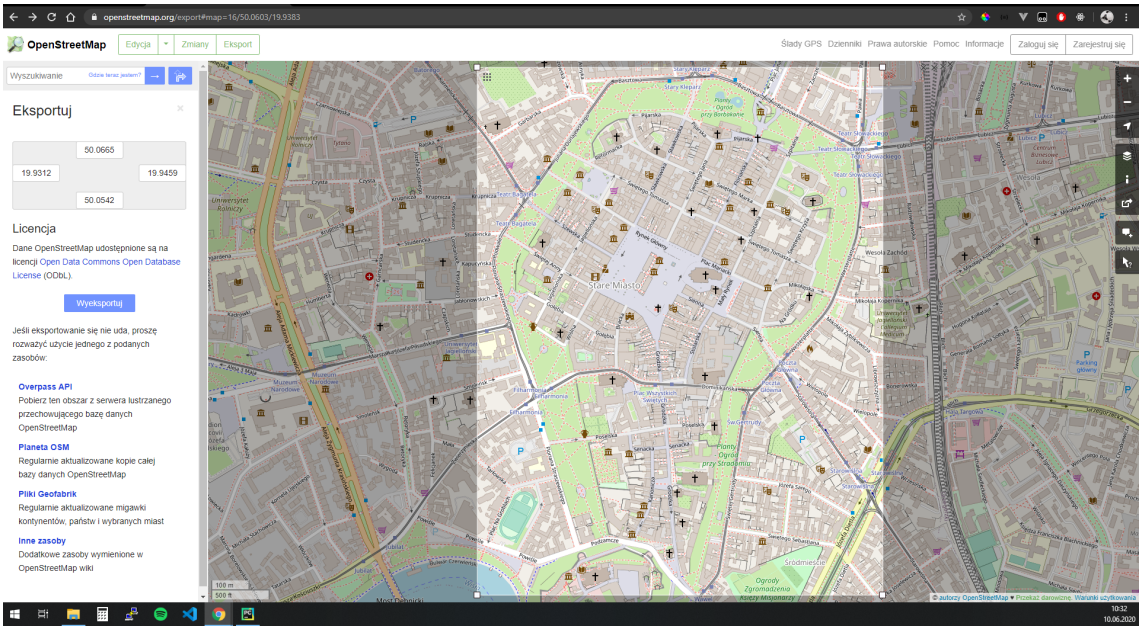
Zdecydowaliśmy się na wykorzystanie Pythona, gdyż jest prostym językiem, idealnie nadającym się do wykonywania różnych nieskomplikowanych matematycznie operacji. Istnieje wiele przydatnych bibliotek, które ułatwiają pracę. W naszym projekcie do samej wizualizacji użyliśmy PyGame, a do implementacji wszelkich zachowań oraz warunków ruchu wykorzystaliśmy m. in. os, json, threading, math, czy też time.

Dzięki stronie OpenStreetMap mogliśmy uzyskać rzeczywiste współrzędne interesujących nas punktów, a przy użyciu JOSM mogliśmy je odpowiednio edytować i eksportować do przyjaznego formatu *.json*. Całość była regularnie aktualizowana na zdalnym repozytorium na Githubi'e.

4 Symulacja zjawiska - implementacja

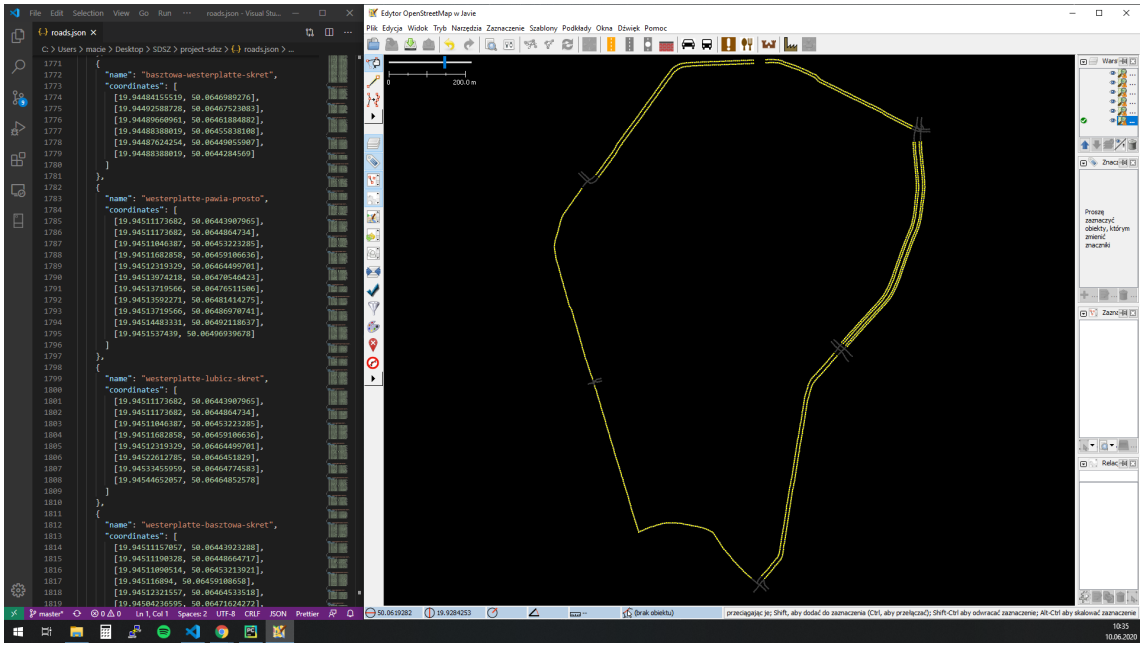
4.1 Przygotowanie siatki obwodnicy

Początek pracy rozpoczęliśmy od znalezienia i pobrania mapy w formacie *mapa.osm* ze strony OpenStreetMap.



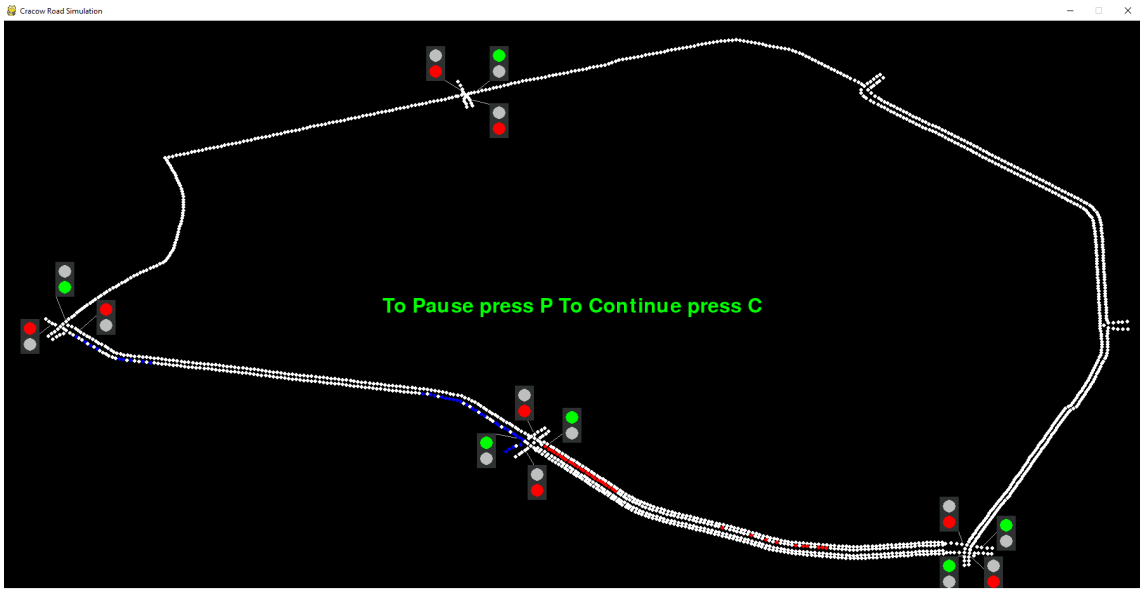
Rysunek 2: Strona internetowa OpenStreetMap

Następnie wykorzystaliśmy oprogramowanie JOSM, do którego zaimportowaliśmy mapę w formacie *mapa.osm* , a następnie nanieśliśmy punkty otrzymując w formacie *mapa.json* zbiór koordynatów. Współrzędne dzieki oprogramowaniu zostały ustawione w taki sposób, że każda komórka jest oddalona od siebie o 5m , gdyż tak założyliśmy w naszym modelu.



Rysunek 3: Docelowym plik *roads.js* oraz program JOSM

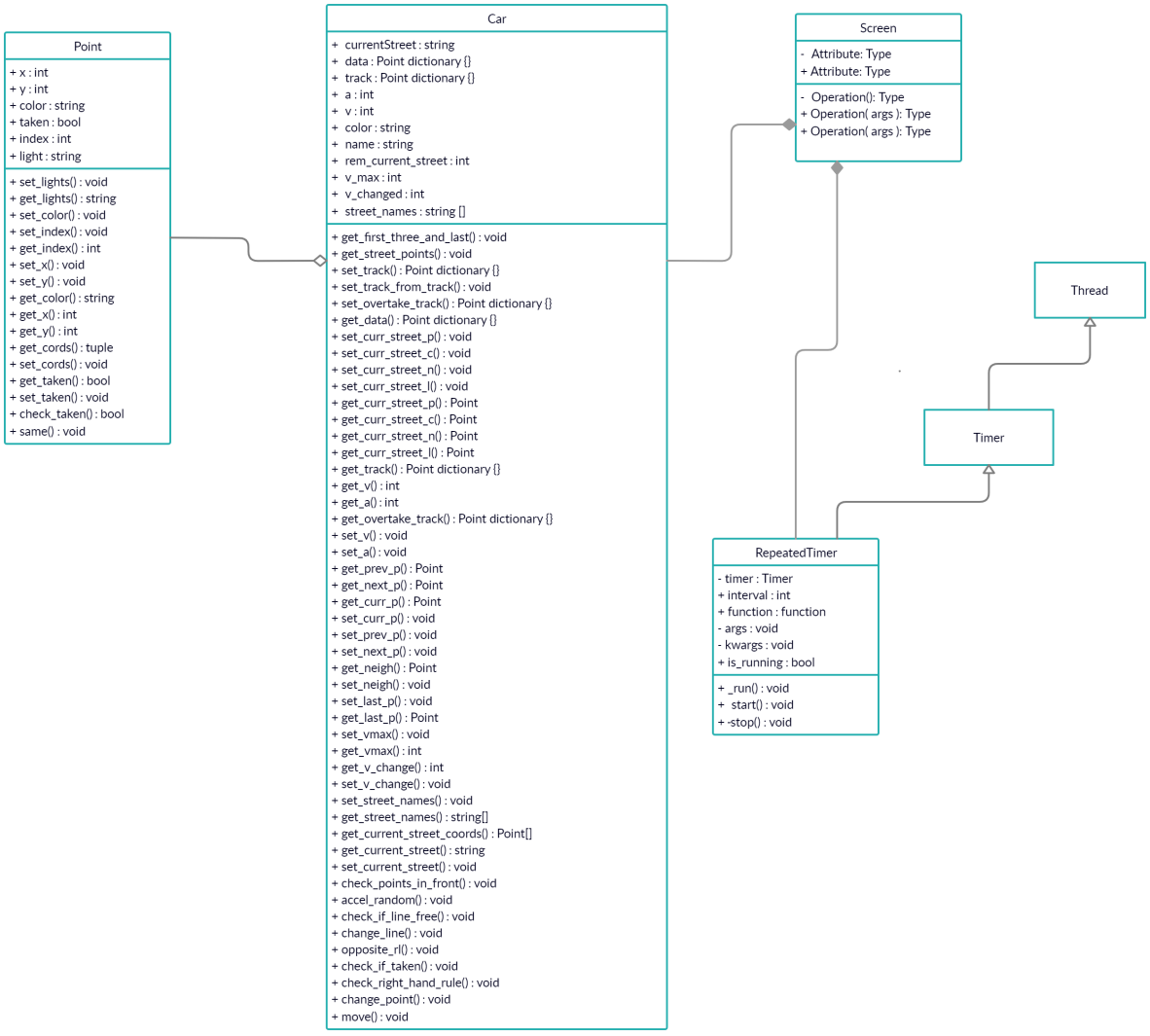
Po odpowiedniej konwersji punktow w pythonie oraz wykorzsytaniu PyGame wizualizacja modelu wygladała tak:



Rysunek 4: Wstępna wizualizacja modelu

4.2 Stworzenie logiki modelu

Poniżej został przedstawiony diagram UML klas naszego projektu, zasadniczo jak widać najwięcej się dzieje w obiektach klasy Car, gdzie zostały zaimplementowane wszystkie warunki dotyczące ruchu, klasa Point posłużyła do pracy z punktami i stworzeniem siatki z komórek, klasa Screen odpowiada jedynie za wizualizację, a jako ostatnia użyta klasa RepeatedTimer odpowiada za liczenie czasu i m. in. obsługę działania świateł. Czas niejako jest liczony w osobnym wątku, tak aby nie ingerować w sam ruch pojazdów na obwodnicy.



Rysunek 5: Diagram UML klas

4.3 Opis klas Point i Car

Pokrótce opiszemy najważniejsze pola i metody klas Point i Car.

Klasa Point

Pola:

- x i y - skonwertowane współrzędne
- color - kolor punktu
- taken - stan punktu 0 (wolne) oraz 1 (zajęte)
- index - unikatowy indeks punktu
- light - jeżeli punkt jest wlotem na skrzyżowanie otrzymuje kolor świateł

Metody:

- set_color() - ustawianie na stan 'red' bądź 'green'
- set_taken() - ustawianie stanu punktu
- get_cords() - pobieranie koordynatow w postaci krotki

Klasa Car

Pola:

- prev_p, curr_p, next_p - obiekty Point opisujące otoczenia i miejsce pobytu auta
- a, v, v_max - kolejno przyśpieszanie, prędkość chwilowa oraz prędkość max
- track, streets - zbiór Point'ów po których będzie jechał pojazd oraz jego wykaz ulic
- current_street - ulica na której obecnie jest pojazd
- v_changed - zmiana prędkości
- overtake_track - zmiana trasy podczas ewentualnego wyprzedzania
- track_end - flaga informująca o dojechaniu do celu

Metody:

- check_points_in_front() - metoda sprawdzająca tyle punktów przed przed sobą o ile w kolejnej iteracji miałaby się przemieścić pojazd, jeśli wykryje inny zwalnia zgodnie z przyjętym modelem.
- accel_random() - metoda odpowiedzialna za tzw. element losowy czyli prawdopodobieństwo na hamowanie

- check_if_line_free() - metoda sprawdzająca czy pas ruchu obok jest wolny na ewentualne wyprzedzanie
- change_lane() - metoda odpowiedzialna za zmianę pasa
- opposite_rl() - metoda elo elo
- check_right_hand_rule() - metoda symulująca regułę prawej ręki
- check_taken() - metoda pomocnicza do ww sprawdzająca możliwość skrętu w lewo
- change_point() - metoda odpowiadająca za przesuwanie pojazdu
- move() - metoda skupiająca wszystkie warunki ruchu i ostatecznie przesuwająca pojazd

4.4 Opis zasady działania algorytmu

- każdy pojazd zajmuje dokładnie jedną jedną komórkę o długości 5m (samochody jednoosobowe)
- każdy pojazd losuje trasę jaką pokona (wszystkie kombinacje tras zostały wygenerowane w osobnym pliku)
- każdy pojazd losuje prędkość maksymalną, z która będzie się poruszał z przedziału (30 - 1 komórka, 40 - 2 komórki, 50 - 3 komórki) gdyż zakładamy iż rzeczywisty ruch na obwodnicy nie przekracza pewnej granicy, zważywszy na ograniczenia.
- w momencie rozpoczęcia symulacji wszystkie samochody rozpoczynają swój ruch i poruszają się do przodu o zadaną ilość kratek w przeliczeniu na odpowiednią prędkość. W czasie trwania ruchu zachodzi sprawdzanie wszelkich warunków oraz wykonywane są zdarzenia losowe zgodnie z przyjętym modelem.
- każdy pojazd przy wjeździe na skrzyżowanie sprawdza czy istnieją na nim światła i jeśli są to odpowiednio się zachowuje. W naszym modelu zakładamy dwa rodzaje świateł czerwone i zielone. Zostały pominięte z strzałki warunkowe oraz światło żółte z powodów komplikacji implementacji
- kiedy pojazd dotrze do skrzyżowania docelowego zjeżdża z niego i zostaje usunięty ze zbioru pojazdów znajdujących się na obwodnicy. W przypadku mniejszej ilości pojazdów zostają losowo generowane kolejne samochody z pewnym zmiennym prawdopodobieństwem tak aby nie zaszła sytuacja, iż obwodnica jest pusta bądź praktycznie cała zapełniona.

4.5 Dane wejściowe i wyjściowe

4.5.1 Dane wejściowe

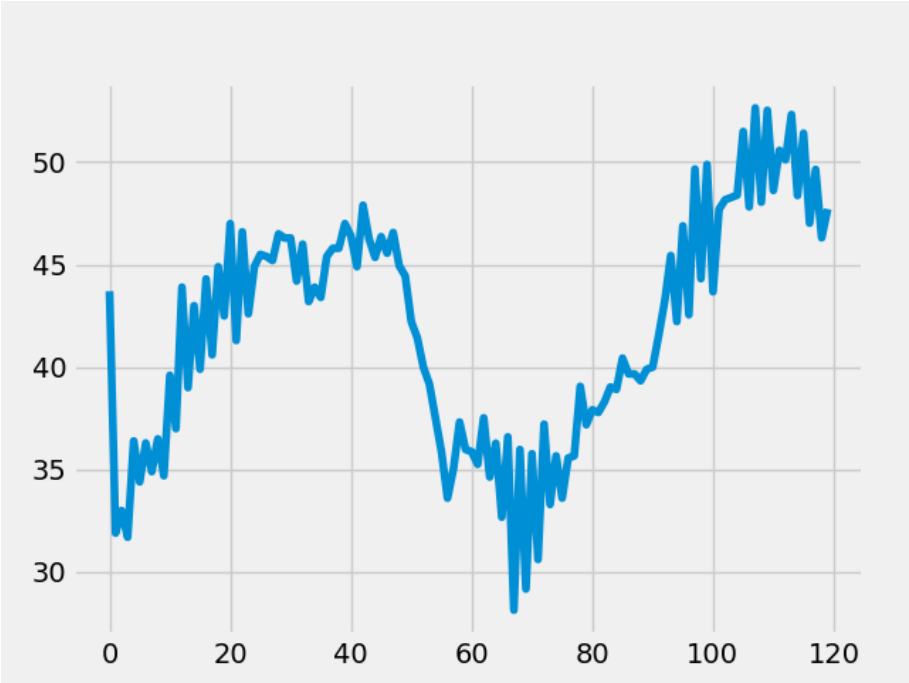
Przed rozpoczęciem symulacji możemy ustalić pewne dane wejściowe takie jak:

- początkowa liczba pojazdów na obwodnicy
- cykl sygnalizacji świetlnej każdego ze skrzyżowań
- czas trwania symulacji

4.5.2 Dane wyjściowe

W trakcie trwania symulacji otrzymujemy na bieżąco pewne informacje:

- prędkość średnią na obwodnicy
- Średnią prędkość maksymalna
- percetage dff
- ilość pojazdów na obwodnicy
- czas trwania czy tez numer iteracji
- powinna byc przepustowsc dorobic
- wykres średniej prędkości w czasie rzeczywistym



Rysunek 6: Wykres zależności prędkości średniej od czasu