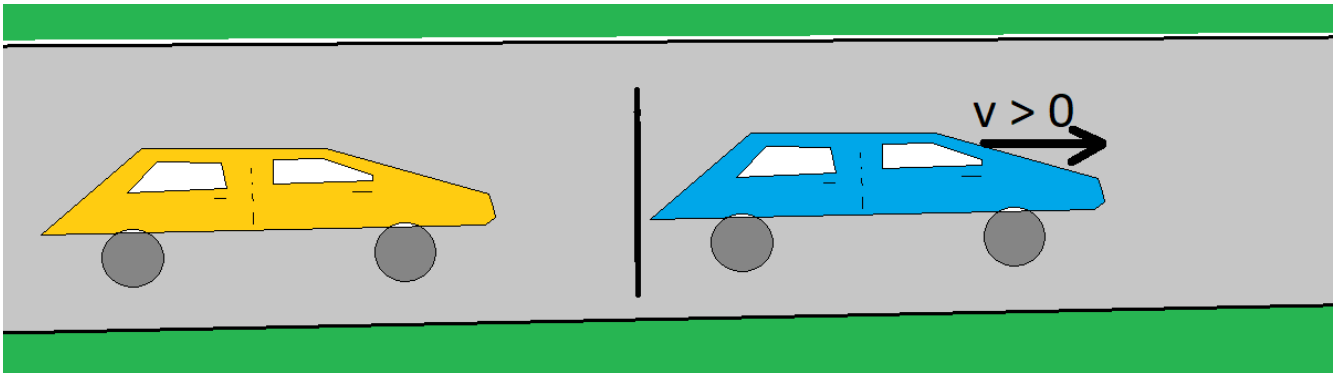
Pomiar czasu ruszania na podstawie nagrania z kamery

Autor: Robert Czwartosz

27.11.2019r.

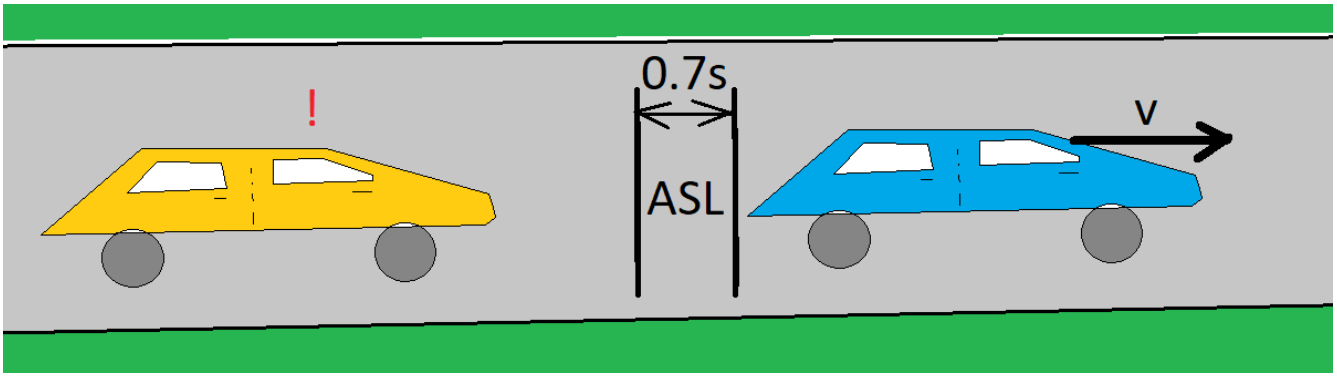
Wprowadzenie

Celem projektu było napisanie oprogramowania mierzącego czas ruszania samochodów. Przed pomiarem samochody stoją w oczekiwaniu na zielone światło. Jeśli pojawi się zielone światło, to pierwszy z nich rusza. Jeśli pierwszy ruszy, to zaczyna się odliczanie czasu po którym ruszy drugi samochód.



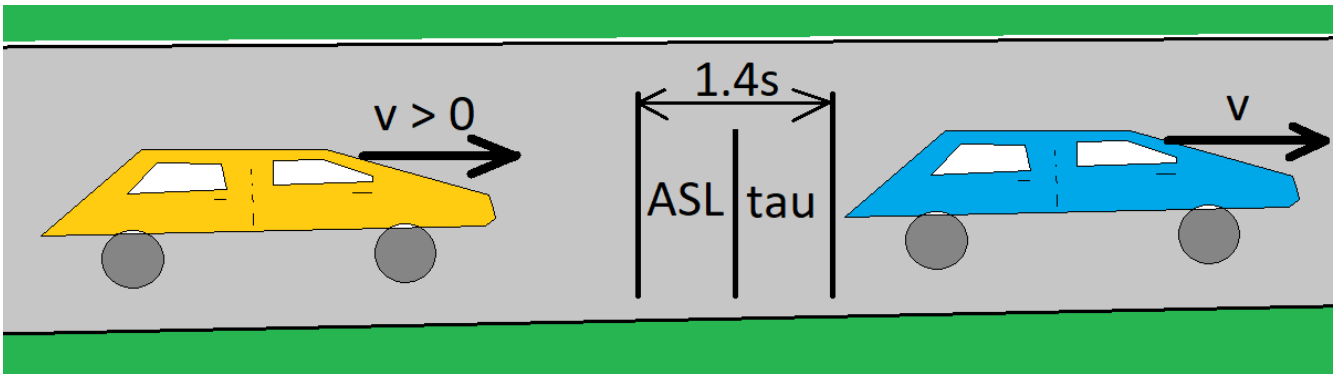
Rysunek 1: Pierwszy samochód rusza

Po czasie **ASL** – *action step length* kierowca żółtego samochodu zauważa, że niebieski samochód ruszył [4].



Rysunek 2: Kierowca drugiego samochodu widzi ruszenie pierwszego

Po czasie **tau** żółty samochód rusza [4]. Czas **tau** jest czasem potrzebnym do bezpiecznego ruszenia przy zachowaniu odpowiedniej odległości.



Rysunek 3: Drugi samochód rusza

Program mierzy sumę czasów **ASL** i **tau**. Skrypt **czasRuszania.py** został napisany na podstawie programów([1], [2], [3]) znalezionych w internecie.

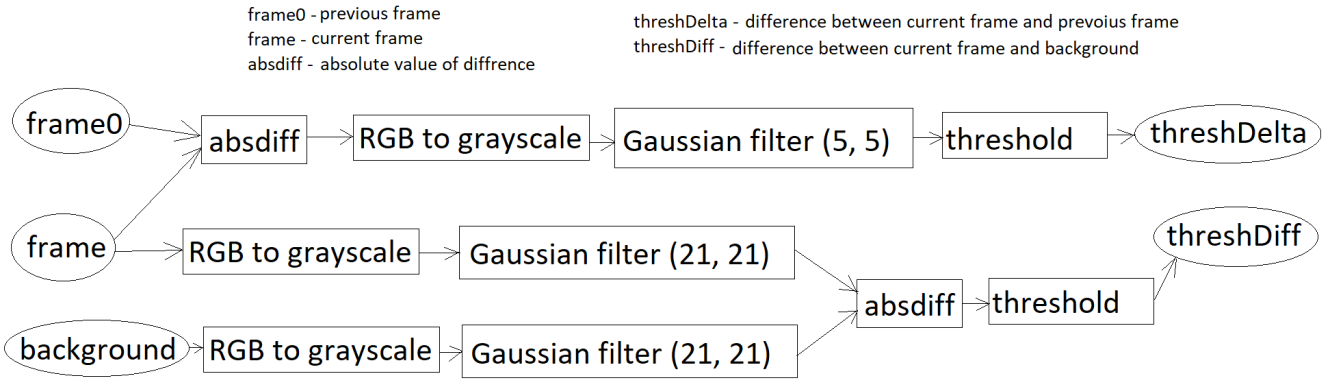
Sposób obliczania czasów ruszania z nagrania – działanie skryptu czasRuszania.py

W celu sprawdzenia obecności pojazdów na odpowiednich miejscach, jest obliczana wartość bezwzględna różnicy pomiędzy aktualną klatką, a tłem. Tło jest zmieniane, tak aby dopasować się do pory dnia. Aby wykryć ruch pojazdów, obliczana jest wartość bezwzględna różnicy pomiędzy aktualną klatką, a poprzednią.

Rysunek 4: Pierwszy samochód rusza

Rysunek 5: Drugi samochód rusza

Do filtracji obrazów są używane filtry uśredniające (gaussowskie) z maskami o wymiarach 5x5 oraz 21x21. Filtry uśredniające są używane przed lub po zamianie współrzędnych RGB na skalę szarości. Po filtracji obrazów z szumów następuje progowanie.

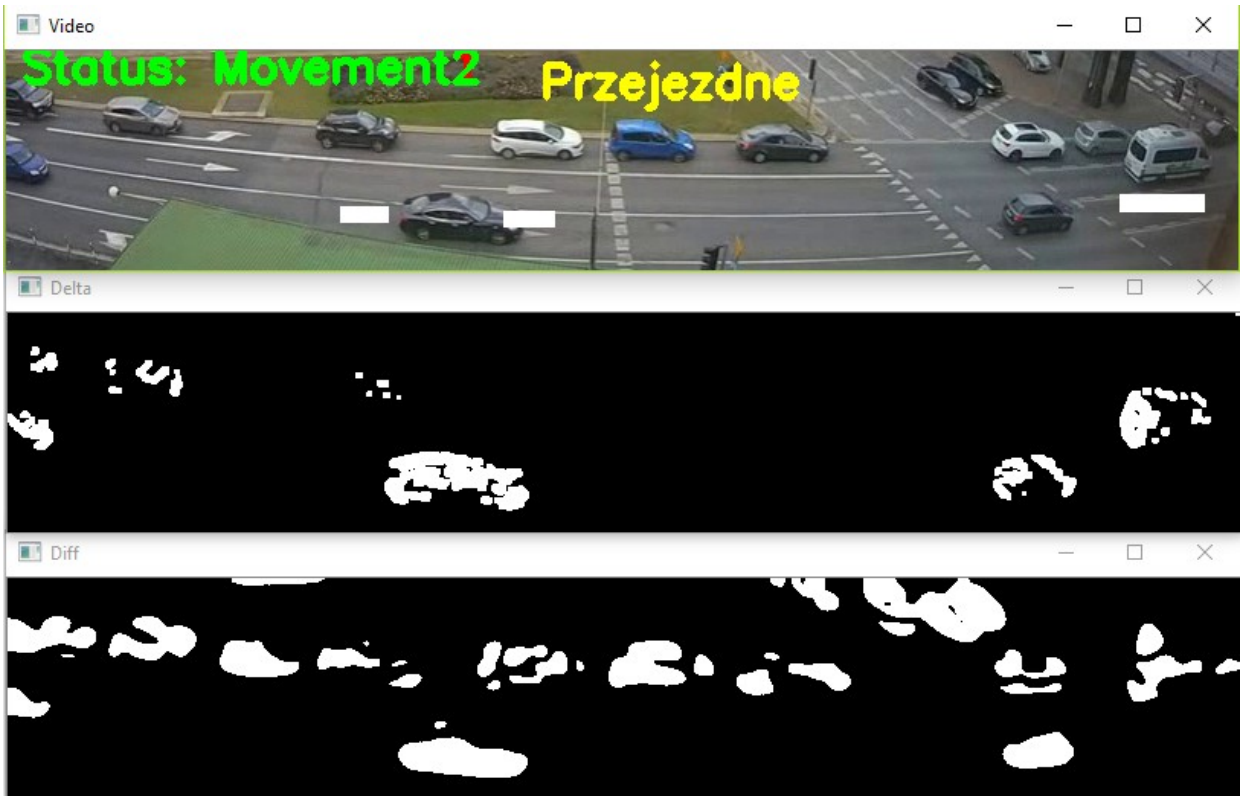


Rysunek 6: Metoda filtrowania obrazów

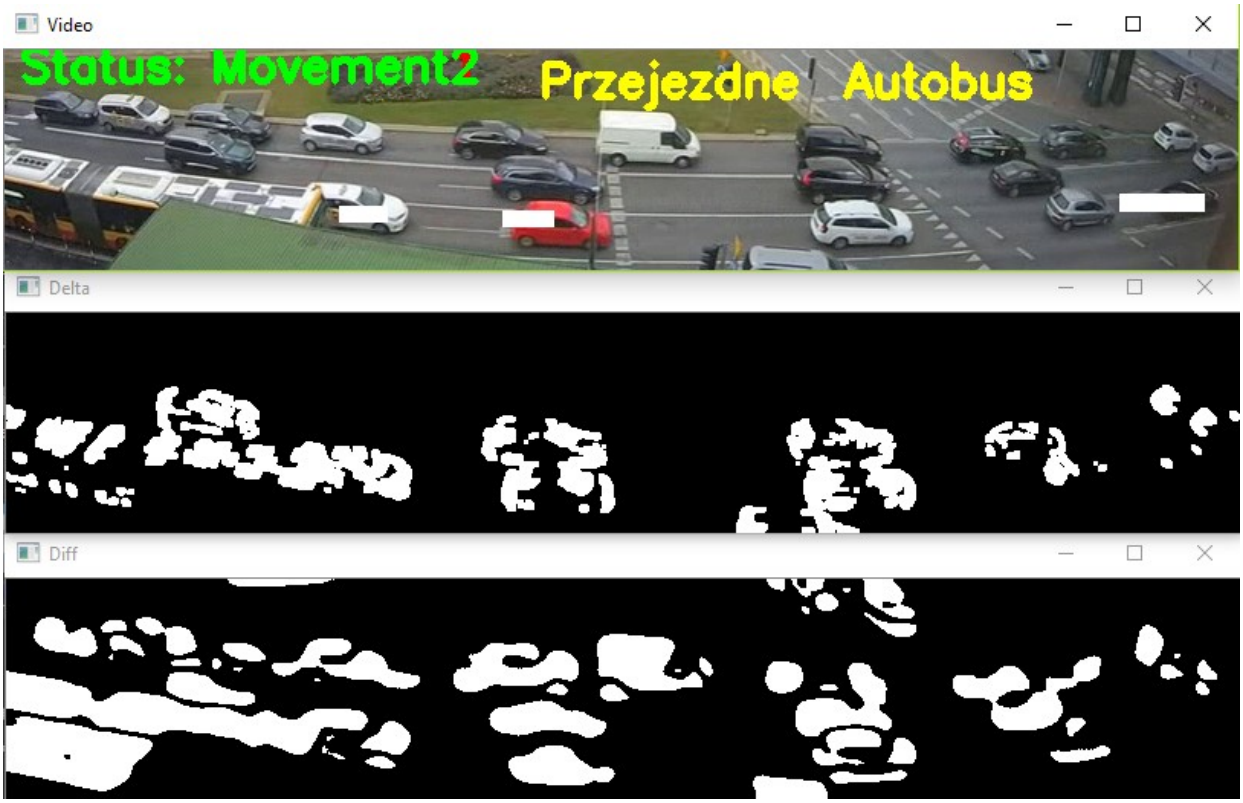
Jeśli pojazdy są na miejscach nr 1 i 2 przez co najmniej 2 sekundy, to rozpoczyna się oczekiwanie na ruch pojazdu nr 1. Po ruszeniu pojazdu nr 1 rozpoczyna się oczekiwanie na ruszenie pojazdu nr 2. Gdy pojazd nr 2 ruszy, to jest obliczany czas ruszenia pojazdu nr 2 od momentu ruszenia pojazdu nr 1. Ten czas jest traktowany jako pomiar. Dodatkowo jeśli pojazd nr 1 rusza w momencie gdy jest korek, to pomiar jest odrzucany. Korek jest wtedy gdy miejsce nr 3 jest zajęte. Położenia i rozmiary miejsc nr 1, 2 i 3 ustala użytkownik wpisując je w pliku konfiguracyjnym.

Rysunek 7: Rozmieszczenie miejsc nr 1, 2 i 3

Ten sposób obliczania czasów jest narażony na różnego rodzaju zakłócenia np.: autobus zajmujący dwa miejsca jednocześnie, przesłanianie przez pojazdy z sąsiedniego pasa ruchu itp. Dlatego dodano funkcję wykrywania pojazdów długich takich jak autobus lub ciężarówka. Pojazdy długie są wykrywane tylko w okolicy pasa na którym wykonywany jest pomiar. Samo wykrywanie działa w oparciu o detekcję „blobów”, czyli grup połączonych pikseli o tym samym kolorze [3]. Kryterium tej detekcji był odpowiednio duży obszar zajmowany przez „blob”.



Rysunek 8: Jeden samochód zajmuje dwa miejsca, przez co ruch jest wykryty w dwóch miejscach jednocześnie



Rysunek 9: Autobus jest wykryty poprzez detekcję "blobów"

Rysunek 10: Dla tego obrazu autobus został "niezauważony" przez detektor

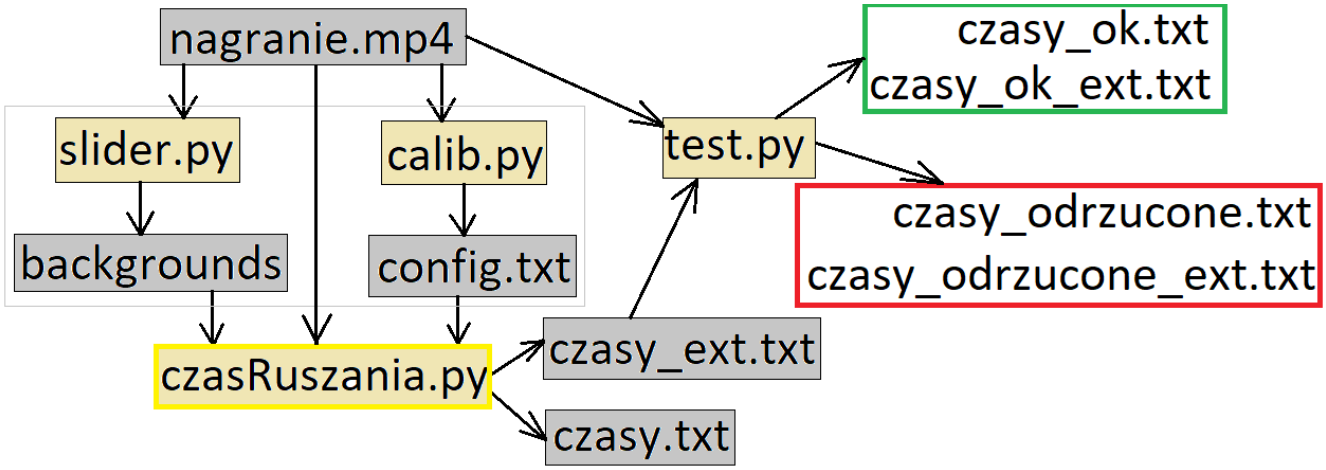
Ta metoda dobrze sprawdza się do wykrywania długich pojazdów, ale nie odrzuca wszystkich błędnych pomiarów. Dlatego po wykonaniu skryptu, przeglądano wszystkie fragmenty filmów pomiędzy czasem ruszenia pojazdu nr 1, a czasem ruszenia pojazdu nr 2. Aby uprościć przeglądanie napisano odpowiedni skrypt. Użytkownik ma możliwość przeglądania fragmentów po kolei oraz akceptowania lub odrzucania w zależności czy wystąpiły pewne nieprawidłowości np.: przesłonięcie przez pojazd z sąsiedniego pasa wpłynęło na pomiar.

Sposób postępowania

Najpierw nagranie jest przeglądane za pomocą programu **slider.py** w celu wydobycia odpowiedniej liczby teł. Tła są wydobywane średnio co pół godziny w dzień i co 1.5h w nocy. Następnie wykonywana jest kalibracja, czyli ułożenie miejsc na odpowiednich pozycjach i określenie ich wymiarów. W tej czynności skrypt **calib.py** pełni funkcję pomocniczą.

Po przygotowaniu teł i kalibracji uruchamiany jest skrypt **czasRuszania.py**. Ten skrypt mierzy czasy ruszania i odrzuca pomiary wykonane w złych warunkach np.: przesłonięcie przez autobus lub korek. Nie zawsze program rozpoznaje czy pomiar wykonany był w odpowiednich warunkach. Zatem należy przejrzeć momenty wykonywania pomiarów. Zmierzone czasy są zapisywane do pliku **czasy.txt**. Natomiast czasy wraz z momentami rozpoczęcia i zakończenia pomiaru są zapisywane do pliku **czasy_ext.txt**.

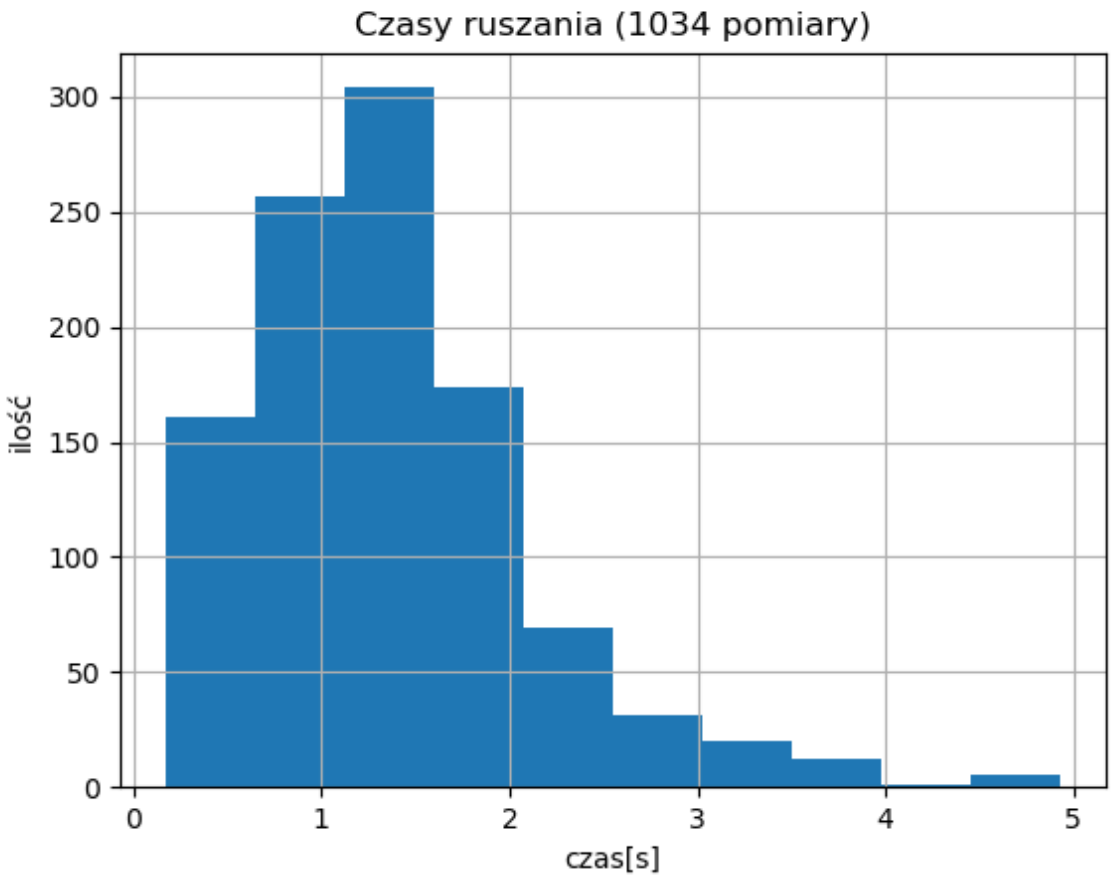
Skrypt **test.py** pomaga użytkownikowi przeglądać klatka po klatce momenty wykonania pomiarów. Skrypt korzysta z plików **czasy_ext.txt** oraz **nagranie.mp4**. Jeśli pomiar był wykonany w odpowiednich warunkach użytkownik może zatwierdzić go klawiszem “A”, lub w przeciwnym razie odrzucić klawiszem “O”. Po przejrzeniu wszystkich pomiarów, wszystkie pomiary wykonane w odpowiednich warunkach są zapisywane do plików **czasy_ok.txt** oraz **czasy_ok_ext.txt**. Analogicznie postępuje się z odrzuconymi pomiarami.



Rysunek 11: Sposób postępowania prowadzący do otrzymania pomiarów czasów ruszania

Wyniki

Na podstawie ok. 150h nagrań, wykonano 1034 pomiary. Średni czas ruszania wynosi 1.36s. Wyniki przedstawiono na histogramie na rysunku nr 12.



Rysunek 12: Histogram czasów ruszania samochodów ze świateł

Bibliografia

[1] https://www.pyimagesearch.com/2015/05/25/basic-motion-detection-and-tracking-with-python-and-opencv – wykrywanie obecności

[2] https://gist.github.com/pknowledge/623515e8ab35f1771ca2186630a13d14 – wykrywanie ruchu

[3] https://www.learnopencv.com/blob-detection-using-opencv-python-c/ – wykrywanie „blobów”

[4] https://sumo.dlr.de/docs/Car-Following-Models.html – wyjaśnienie parametrów ASL i tau w symulatorze SUMO