Czech Technical University in Prague
Faculty of Electrical Engineering
Department of Cybernetics

Bachelor's Thesis

Collision Avoidance on General Road Network

David Kubeša

Supervisor: Ing. Martin Schaefer

Study Programme: Open Informatics

Field of Study: Computer and Informatics Science

May 20, 2015

České vysoké učení technické v Praze
Fakulta elektrotechnická

Katedra kybernetiky

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Student: David K u b e š a

Studijní program: Otevřená informatika (bakalářský)

Obor: Informatika a počítačové vědy

Název tématu: Algoritmy pro vyhýbání se kolizím silničních vozidel

Pokyny pro vypracování:

1. Nastudujte metody pro vyhýbání se vozidel na jednosměrných silnicích, hlavně ideu SafeDistance [3] metody.
2. Zobecněte tuto metodu na celou silniční síť, včetně křižovatek bez světel a silničních pravidel.
3. Vytvořte nové metody pro ovládání aut v simulaci.
4. Vytvořte ukázkové scénáře za využití této metody.

Seznam odborné literatury:
[1] Russell, S., & Norvig, P. (2010). Artificial Intelligence: A Modern Approach (p. 1132). doi:10.1017/S0269888900007724
[2] Schaefer, M. Collision Avoidance of Highway Traffic, 2014. Master's Thesis. Czech Technical University in Prague, Czech Republic
[3] Schaefer, M. Noncooperative collision avoidance of road vehicles, 2011. Bachelor's Thesis. Czech Technical University in Prague, Czech Republic
[4] Lalish, E., & Morgansen, K. A. (2012). Distributed reactive collision avoidance. Autonomous Robots, 32, 207–226. doi:10.1007/s10514-011-9267-7

Vedoucí bakalářské práce: Ing. Martin Schaefer

Platnost zadání: do konce letního semestru 2015/2016

L.S.

doc. Dr. Ing. Jan Kybic
vedoucí katedry

prof. Ing. Pavel Ripka, CSc.
děkan

V Praze dne 14. 1. 2015

Czech Technical University in Prague
Faculty of Electrical Engineering

Department of Cybernetics

BACHELOR PROJECT ASSIGNMENT

Student:	David K u b e š a
Study programme:	Open Informatics
Specialisation:	Computer and Information Science
Title of Bachelor Project:	Collision Avoidance on General Road Network

Guidelines:

1. Research collision avoidance methods for one-way roads, focus on SafeDistance [3] method idea.
2. Discuss generalization of the idea to general road network including junctions without priorities and traffic lights.
3. Design and implement a method to control cars in a simulation.
4. Provide an experimental evaluation of the proposed method.

Bibliography/Sources:

[1] Russell, S., & Norvig, P. (2010). Artificial Intelligence: A Modern Approach (p. 1132). doi:10.1017/S0269888900007724
[2] Schaefer, M. Collision Avoidance of Highway Traffic, 2014. Master's Thesis. Czech Technical University in Prague, Czech Republic
[3] Schaefer, M. Noncooperative collision avoidance of road vehicles, 2011. Bachelor's Thesis. Czech Technical University in Prague, Czech Republic
[4] Lalish, E., & Morgansen, K. A. (2012). Distributed reactive collision avoidance. Autonomous Robots, 32, 207–226. doi:10.1007/s10514-011-9267-7

Bachelor Project Supervisor: Ing. Martin Schaefer

Valid until: the end of the summer semester of academic year 2015/2016

L.S.

doc. Dr. Ing. Jan Kybic
Head of Department

prof. Ing. Pavel Ripka, CSc.
Dean

Prague, January 14, 2015

Poděkování

Rád bych hlavně poděkoval svému vedoucímu práce Ing. Martinu Schaeferovi za trpělivost a pomoc při psaní této práce. Poděkování patří také mým rodičům a Zuzaně Vozárové za podporu při mém studiu.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

V Praze dne 20. května 2015 ....
Podpis autora práce

Abstract

Autonomous Cruise Control (ACC) systems are widely used in automotive industry to maintain safe longitudinal distance between vehicles. The concept of maintaining safe distance is well applicable on straight roads, especially highways. We discuss the generalization of the concept on the general road network (e.g., junctions without traffic lights). We propose a generalization of the Safe-distance method – implementation of the ACC in the Agent-Drive simulation platform. The generalization is based on the idea of application of the ACC control on a virtual traffic situation. A real world traffic situation is transformed to the virtual situation, that is then maintainable by the prior Safe-distance method. The proof-of-concept experiments are presented. These experiments show non-collisional nature of proposed method and its ability to work in realtime. This work promises interesting future work and proposes several directions of our research intentions.

Abstrakt

Adaptivní tempomaty (Autonomous Cruise Control - ACC) jsou již široce rozšířené v automobilovém průmyslu. Umožnují udržovat vzdálenost mezi za sebou jedoucími vozidly. Tento koncept se již dobře osvědčil na dálnicích nebo rovných silnicích. V této práci probíráme rozšíření tohoto principu na celou silniční síť (bez světelné signalizace). Předkládáme rozšíření Safe-distance metody – implementaci ACC na simulační platformně Agent-Drive. Rozšíření je založeno na nápadu použít ACC i na jiné situace než pro které bylo vytvořeno pomocí transformace reálných situací do takových situací, které původní Safe-distance zvládne vyřešit. Možnosti metody ukazujeme na experimentech, které ukazují bezkolizní charakter této metody. Další hlavní výhodou této metody je, že pracuje v reálném čase. Práce ukazuje další možné směry výzkumu na toto téma.

Keywords

autonomous vehicles, multi-agent simulation, non-cooperative planning, collision avoidance

Contents

List of Figures

Chapter 1

Introduction

The car transportation is a dangerous mode of transportation. In 2011, more than 30,000 people died on the roads of the European Union [3]. There is a lot of concern how to make this mode of transportation safer. There is an ongoing development of passive and also active safety components in cars. Since most of the collisions are caused by a human factor these systems have a goal to ease the driving for a driver.

Advanced driver assistance systems (ADAS) are examples of an active safety system. Assistance systems vary in the level of autonomy. Some systems offer a detection and warning of a dangerous situation, e.g, when driver needs to slow down to avoid a collision with a vehicle ahead. A more advanced system can even take control of the car in case of emergency situation. Fully autonomous – driver-less cars are already tested on public roads. Automation can bring more safety and even more efficiency. Automated cars achieve significant efficiency boost in sense of usage of road infrastructure.

The integration of assistance systems or autonomous vehicles is a gradual process. The traffic is heterogeneous even in the sense of autonomy. We are interested in coordination techniques that are applicable in such heterogeneous setting. Particularly, we build our method on principle of an existing driver assistance system. We propose a concept of generalized autonomous cruise control.

In the Chapter 2 we will focus on the description of different approaches to collision avoidance. In the Chapter 3 we will specify the representation of the road network. In the next Chapter 4 we describe the basic principle of our method's algorithm. In the Chapter 5 we will describe the architecture of a simulator used for evaluation of our method. Exact implementation of our method can be found in Chapter 6. Evaluation of this method and the concrete scenarios are described in Chapter 7. The conclusion and future enhancements are summarized in Chapter 8.

Chapter 2

Background

In this section we will define what is an agent and what is the multi-agent system. This is especially for the problem specification. The difference between Reactive agents and Planning agents is also described in this chapter.

2.1 Agent Definition

Jacques Ferber in [4] defines an Agent as:

> "*An agent can be a physical or virtual entity that can act, perceive its environment (in a partial way) and communicate with others, is autonomous and has skills to achieve its goals and tendencies. It is in a multi-agent system (MAS) that contains an environment, objects and agents (the agents being the only ones to act), relations between all the entities, a set of operations that can be performed by the entities and the changes of the universe in time and due to these actions.*"

2.2 Multi-agent system example

We describe an example of multi-agent system bellow for better imagination how multi-agent system can look like.

Imagine a game of Blackjack. Every player of this game tries to win this game and every player has to know the game rules - the Environment in order to achieve this goal. If the player is represented as an agent, it needs to know the player's budget to be able to truly represent the player. The agent also need to interact with another agents, for example with the croupier or the another agent to be able to maximize the chances of win. The collection of all these agents represents the multi-agent system.

2.3 Reactive agents vs Planning agents

We use division between reactive agents and planning agents and the terminology used in [5]. These agents represent two main approaches in collision avoidance. Reactive agents only react to the behaviour of other vehicles. The reactive agent (Generalised Safe-distance Agent) is proposed in this thesis. Planning agents use negotiation between vehicles for planning. The coordination between plans can be done before or after agents create their plans to achieve non-collisional plans. If coordination is done after planning, plans need to be merged to be non-collisional. When coordination is done before planning, some social law need to be used. As [5] stated, "A social law is a generally accepted convention that each agent has to follow". The example of a planning agent that coordinate plans after agent's plans creation is an agent based on Asynchronous Decentralized Prioritized Planning (ADPP) algorithm ([6]). The ADPP agent that is used for the same problem as the Generalised Safe-distance Agent is presented in [7].

The main drawback of planning algorithms is the time required for the communication between vehicles. Cooperative planning also usually needs more resources than reactive planning. However planning agents create more optimised plans in general.

Chapter 3

Problem specification

The Environment - the road network for the multi-agent system and the problem of a multi-agent coordination is described in this section.

3.1 Road network

We define the environment – the road network. The road network is a graph structure. Junctions are nodes and roads connecting junctions are edges. There are optionally more lanes in one edge and every lane is directed. All the components have coordinates and dimensions in 2D. Vehicles move freely in the 2D space, but respect the underlying road network.

3.2 Multi-agent coordination

Every vehicle is driven by a related agent. Every agent has a specific destination and knows a sequence of edges of road network to get there. Agent is able to generate and follow a path towards its destination. The problem to solve is a collision avoidance. We need the agent to follow its sequence of edges while avoiding collisions with other vehicles. The agent is able to adjust vehicle's speed and also adjust trajectory, e.g., change lanes. The control is online, i.e., the agent is repeatedly sensing vehicle's state and considering the next control inputs. The sensing is considered to be perfect (i.e., agent knows positions and velocity vectors of others in a limited radius).

We assume junctions without any traffic rules (e.g., priority to the right, priority of the main road) nor traffic lights nor signs. The agents form a multi-agent system without any particular communication between agents. There is also no negotiation with other agents, agents only react on behaviour of other agents. This system is decentralized.

Chapter 4

Solution method

Our approach is based on the principle of Autonomous Cruise Control system (ACC). We describe the principle in Section 4.1. Considering the functionality of the ACC system, we propose a mechanism using the ACC system as its core component for collision avoidance.

4.1 Autonomous Cruise Control functionality

The ACC is an automotive feature that allows vehicle to adapt the vehicle's speed. A radar system attached to the front of the vehicle is used to detect whether slower moving vehicles are in the ACC vehicle's path. If a slower moving vehicle is detected, the ACC system will slow the vehicle down. If the system detects that the forward vehicle is no longer in the ACC vehicle's path, the ACC system will accelerate the vehicle back to its set cruise control speed. We use the ACC definition as it is described in [8].

4.2 Collision avoidance in junctions

This subsection is about a collision avoidance in junctions. This collision avoidance algorithm is an extension to the regular ACC - it allows vehicles not only to control vehicles speed but it also allows vehicles to pass junctions safely. We call it Generalized ACC.

Autonomous Cruise Control system can maintain the safe distances between vehicles, but can't solve junctions like in Figure 4.1. Interesting thought is to transform situation shown in Figure 4.1 to the situation in the Figure 4.2 which ACC can solve.

Basic idea is to trick the ACC, that vehicles that are heading to the junction from another roads are in front of him in some specific distance to the junction. This would force this system to make safe distance between my car and the vehicle which is also heading to the junction. This would allow them to pass junction safely. The main problem in here is how to virtually put another vehicle before me. For this purpose, location and velocity of another vehicle and precise road map is required. From the location of another vehicle we can calculate the vehicles distance to the junction using the road map. As we know this distance, we can compare it to the operating vehicle's distance. If it is closer to the junction we can try to set it as a vehicle before me. We can try to set multiple vehicles and take in account only the closest one. Autonomous Cruise Control system guaranties safe distance between me and the vehicle before me, so if the Generalised ACC is activated far enough to the junction, all vehicles will create safe-distance between them and with safety reserve big enough it allows them to pass the junction safely.

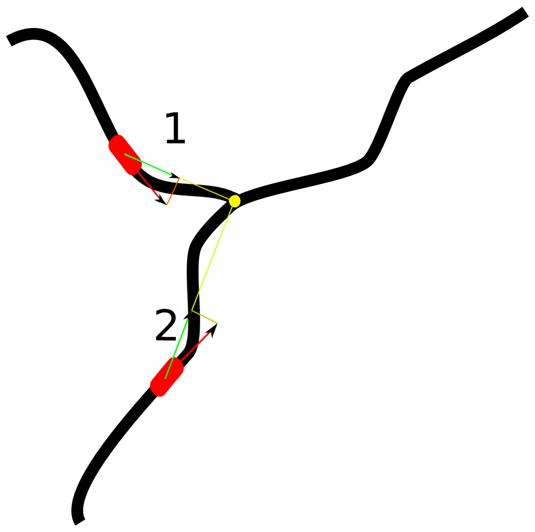


Figure 4.1: Simple situation, two cars are heading to the junction. Car 1 is a bit closer to the junction than car 2.

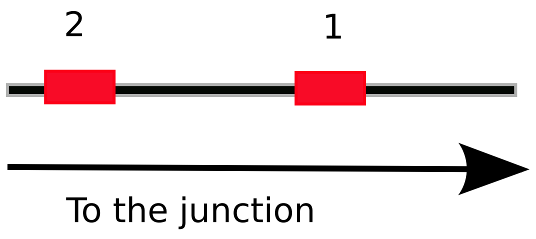


Figure 4.2: Translated situation in Figure 4.1 to the situation where these two cars are on the same road. The order, the distance to the junction and the velocity of each vehicle are preserved.

4.3 Extensions

The algorithm proposed in Section 4.2 works fine, but it allows only one vehicle in the junction. To allow more vehicles in the junction it is needed to know vehicles plans. If we have this knowledge, vehicles can only care about other vehicles which crosses their path in the junction. For example if two vehicles are only passing the junction in the opposite direction it is no need for them to avoid themselves and they can pass junction simultaneously. From vehicles plans it is possible to determinate if they collide. Exact implementation of this principle can be found in Section 6.2.

Chapter 5

Design and Architecture

In this section the Alite toolkit and the AgentDrive project is described. Alite is a base of our AgentDrive project. AgentDrive platform consists of Simulators and Modules.

5.1 Alite

The AgentDrive project is developed using Alite toolkit. Alite toolikit is according to the project wiki[1]:

> *"Alite is a software toolkit helping with particular implementation steps during construction of multi-agent simulations and multi-agent systems in general. The goals of the toolkit are to provide highly modular, variable, and open set of functionalities defined by clear and simple API. The toolkit does not serve as a pre-designed framework for one complex purpose, it rather associates number of highly refined functional elements, which can be variably combined and extended into a wide spectrum of possible systems."*

The Alite simulation is an event based simulation. This means that simulation is driven by an event queue. Every event is handled by this queue so this simulation runs only in one thread. Every simulation event is added to the queue for example when simulator sends updated vehicles it is added to the queue and this event is caught by the AgentDrive and processed. The simulation environment is created by creators and every entity that is in this world is saved in the storage. Every entity in the environment has a sensor and actuator. The Sensor is for sensing the environment state and the actuator is for making desired actions of the entity. More about sensors and actuators is in Section 7.2. The structure of the Alite simulator is also described in the Figure 5.1 bellow. More information about the Alite can be found in [1].

[1]`http://merle.felk.cvut.cz/redmine/projects/alite`

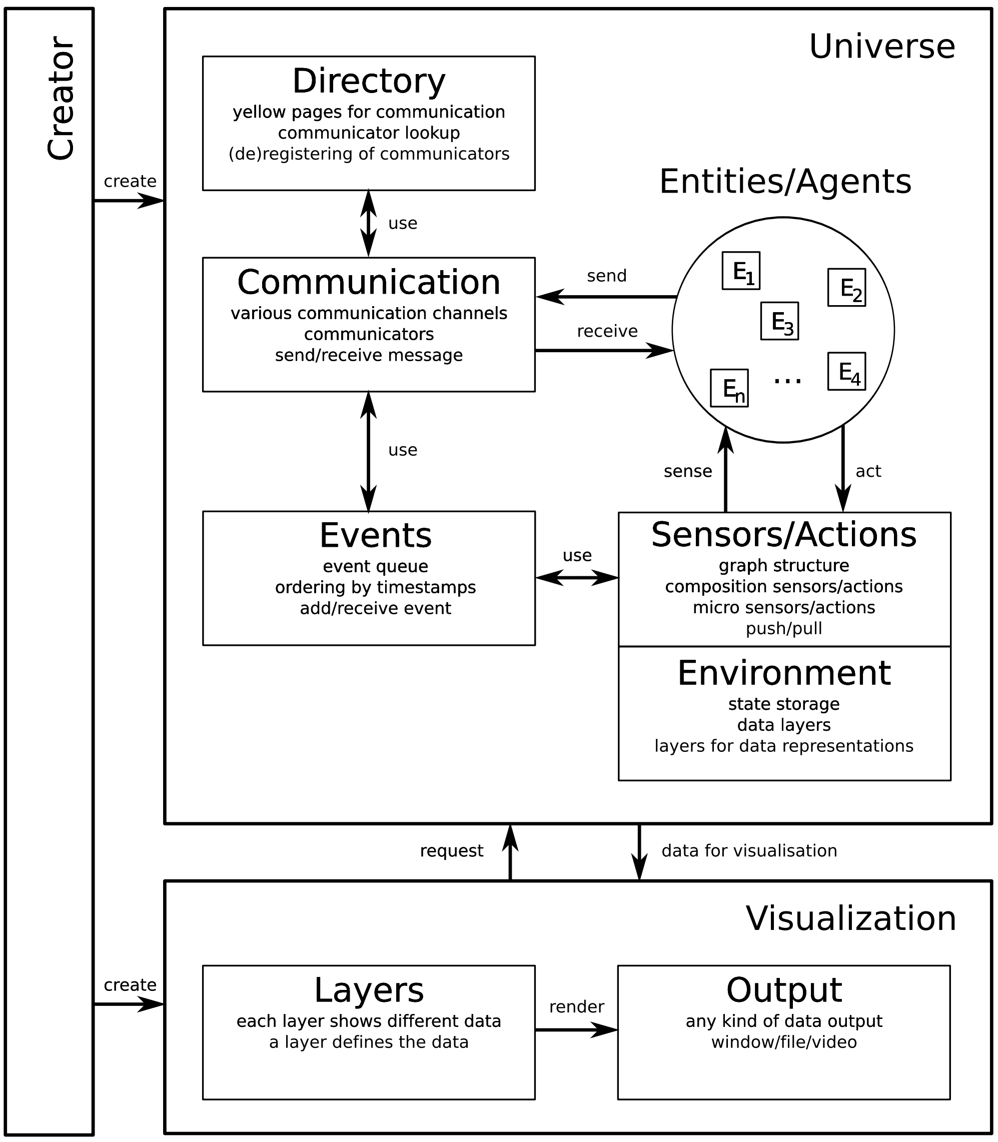


Figure 5.1: Alite schema from [1].

5.2 AgentDrive

Our AgentDrive project is developed using the Alite toolkit. In our AgentDrive platform, every vehicle is controlled by an agent. We work on the module called Highway, the name is a bit misleading because it was extended to the whole road network recently. In our case the entities in the simulation are the vehicles which are stored in the Highway storage. Every vehicle has its sensor and actuator.

5.2.1 Sensor

The vehicle's sensor is vehicle's object that has an access to the Environment. In this case to the Highway Storage. The most basic function of the sensor is to get the location and velocity vector of the vehicle. Another function is for example to get all neighbour vehicles. Additional function is to obtain planned routes of vehicles nearby using the vehicle to vehicle (V2V) communication.

5.2.2 Actuator

The vehicle itself does not have an access to the Environment and the actuator is a tool for sending information from the vehicle to the Environment. The main actuator's function is informing the Environment about creation of the new plan of the vehicle so the simulator can simulate this plan.

5.2.3 Vehicles

Every vehicle has its actuator and sensor but vehicles differ in several parameters. Vehicles can have different parameters e.g. maximum speed, acceleration. Every vehicle is driven by a related Agent. An agent uses the sensor to obtain information from the Environment and the actuator to send information to the Environment.

5.2.4 Simulator controller

The Simulation controller is based on the creator provided by Alite. It contains the local simulator and initializes the simulation itself.

5.3 Simulators

There are two main simulators used to test our method. First is the Local simulator and second is the Simulator Lite. The simulators can be external - not part of the environment and can run separately.

Local simulator Local simulator is used for perfect execution of plans. It just evaluate them and send back updated position speed after specified delay. There is no physical model implemented. This simulator is used for initial experiments and for debugging.

Simulator Lite Another simulator we use is called Simulator Lite. This simulator uses basic physics model. It takes into the account the actual velocity, the acceleration and the capabilities of the selected vehicle. This simulator allows to control the method outputs and it is used for checking if the plans are executable in the psychical word. This simulator has a limit of 70 vehicles correctly simulated. This is because of the limited communication channel between the simulator and the AgentDrive.

The simulation allows to run multiple simulators at the same time. Simulators divide simulated vehicles so every simulator simulates only a subset of vehicles in the simulation.

5.4 Simulation schema

Vehicle's agent uses data from vehicle's sensor for its reasoning. When the new plan is created, the vehicle's agent uses the actuator to let the Environment know about the plan creation. The Environment waits till all plans from all reasoning agents are collected and then it will send them to the simulator. The simulator simulates the plans and sends back radar data (updated vehicles locations and speeds) to the Environment. The Local simulator is implemented in the environment. When the simulator sends radar data back to the environment, Highway Storage updates the agents current positions and then let agents know about that their locations and velocity has been updated. More can be seen in Figure 5.2.

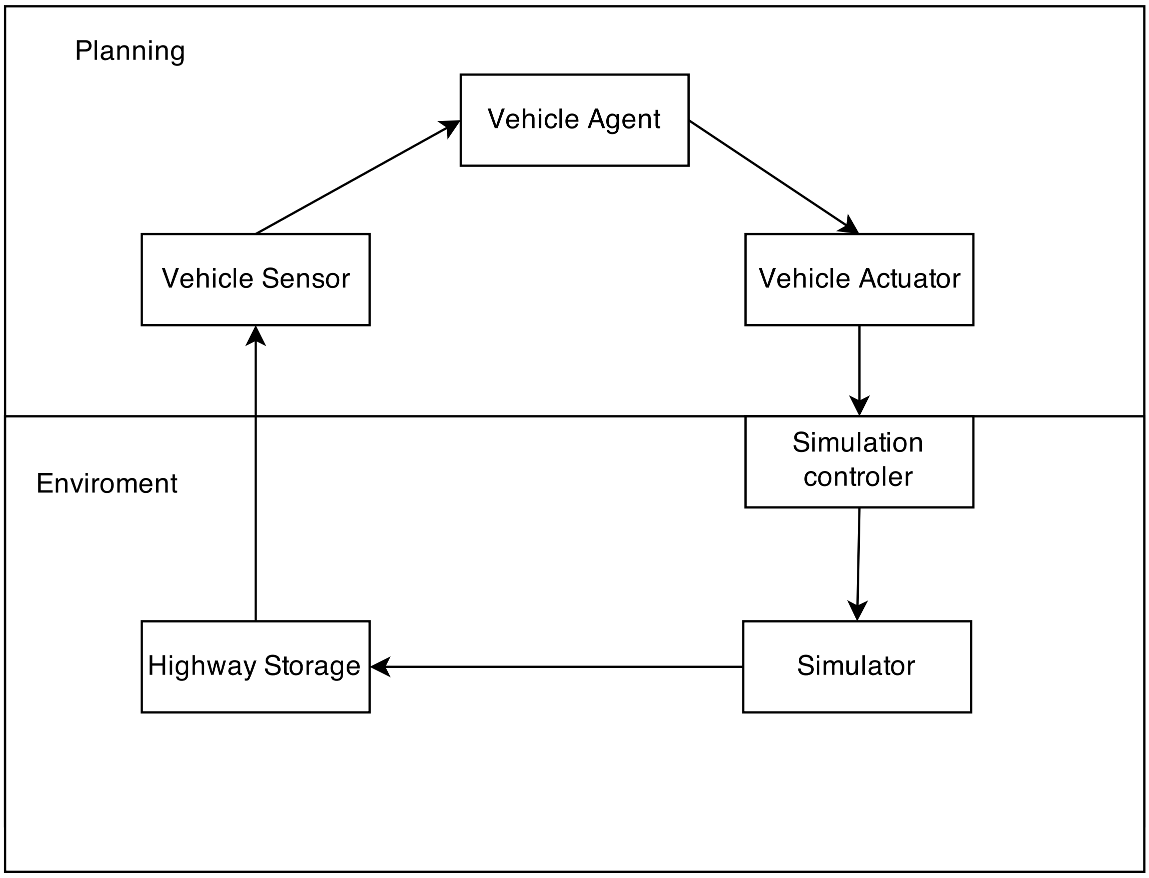


Figure 5.2: The Highway schema. This figure shows how the simulation works. Vehicle's Agent informs the Environment using the actuator about a creation of a new plan. The simulator simulates when all plans from all vehicles are gathered. Simulator send updated vehicles locations and speeds to the Highway Storage and Highway Storage notifies vehicle's sensor about the update.

5.5 Three layer architecture

The Highway architecture consists of these three layers:

1. Manoeuvre layer
2. Waypoint layer
3. Pedal layer.

The most basic layer is the Pedal layer. The Pedal layer is extended by the Waypoint layer that is extended by the Manoeuvre layer. Higher level needs the lower layer to work, but the lower layer does not need the higher level to work. Agents can plan on the different layers.

5.5.1 Manoeuvre layer

The Manoeuvre layer is on the top of a pyramid of layers. The manoeuvre is an object with these predefined parameters:

1. duration. Duration of the manoeuvre stores the time for which is this manoeuvre planned.
2. lane, position and velocity in.
3. lane, position and velocity out
4. acceleration

The speed is in metres per second. A vehicle is represented by manoeuvre(s). Position of a manoeuvre is calculated relatively to the operating vehicle. So for the operating vehicle the position in is always an axis centre. The set of manoeuvres represents a vehicle plan. Manoeuvres in plan correspond to each other - end position and velocity of a manoeuvre is the same as the beginning position and velocity of the next manoeuvre. When the plan is created, it is translated to the waypoint layer.

Type of manoeuvres

1. Straight manoeuvre. This manoeuvre has zero acceleration and does not switch lane.
2. Acceleration manoeuvre. This manoeuvre is the same as the Straight manoeuvre, but only differs in acceleration from which is the velocity out calculated. Acceleration is an constant for every vehicle.
3. Deceleration manoeuvre. This manoeuvre is the same as the Straight manoeuvre, but only differs in acceleration from which is the velocity out calculated. Deceleration constant is a minus number in this case. Deceleration is an constant for every vehicle.
4. Lane left manoeuvre. This manoeuvre is for switching lane. In this case, to the left lane. This manoeuvre has zero acceleration.
5. Lane right manoeuvre. This manoeuvre is for switching lane. In this case, to the right lane. This manoeuvre has zero acceleration.

5.5.2 Waypoint layer

The Waypoint layer consists of the waypoint actions. The waypoint action is an object with:

- car id (number)
- time stamp (long)
- position (3D position in the map)
- speed (number)

How many waypoint actions ahead are calculated wp_{count} is determined by this equation:

$$wp_{count} = (actualSpeed \cdot 2) + 1$$

Positions in the waypoint actions are generated according to the shape of the road. Speed is calculated from minimal speed. Minimal speed s_m is a minimum of all speeds calculated as:

$$s_m = \frac{1}{angle} \cdot 6$$

where the angle is an angle between my position and the waypoint position. The number six is there for scaling the speed to the usual vehicles speed. When the angle is less than 0.4 (around 20 degrees) the speed is set as the maximum speed. Minimal speed is 2 metres per second. The speed of all waypoints actions is scaled from the actual speed to the minimal speed regarding the number of planned waypoint actions.

When the manoeuvre is translated, the minimal speed is set as manoeuvre's speed when manoeuvre's speed is lower than the minimal speed.

5.5.3 Pedal layer

The Pedal layer is represented by a simulator. The simulator reads waypoint actions and try to evaluate them. This layer represents the physical layer. The actual control of the vehicle using pedals and a steering wheel. This layer simulates the driver itself.

Chapter 6

Implementation Details

The Safe-distance method proposed in [2] is used in this thesis as the Autonomous Cruise Control.

The Safe-distance method as the name suggests is designed to keep safe distances between vehicles. This method was designed for highways and has one key ability that the ACC does not have. It is the ability to switch lanes in order to solve the situation. Not only to decrease and increase speed.

6.1 Implementation: Safe-distance method

In this section the original Safe-distance method is described.

The Safe-distance method requires predicted manoeuvres of nearby vehicles to work. To be able to predict manoeuvres of vehicles nearby we need to know these parameters:

- Position and speed of an agent's controlled vehicle.
- Position and speed of an vehicles nearby in the sensors visibility range. From that positions it is possible to find out on which road and lane vehicle is located. The Kd-Tree structure [9] is used to obtain nearest neighbouring lane of the vehicle position.
- Visual inputs from vehicles. For example if the breaking lights are visible or a vehicle is in the turning lane and etc.

6.1.1 Prediction

In this subsection we will describe the possible manoeuvre prediction of the vehicles nearby.

When the vehicle nearby has visible tail break lights we will predict the Deceleration manoeuvre. When the corner lights are visible we will predict the Lane left or the Lane right manoeuvre. By using these lights we can only predict the first manoeuvre in the vehicle's plan. So the rest of the plan is filled by Straight manoeuvres. If we don't have any light signals from the vehicles nearby we can only predict sequence of the Straight manoeuvres. Possible prediction are visible in the Figure 6.1 bellow:

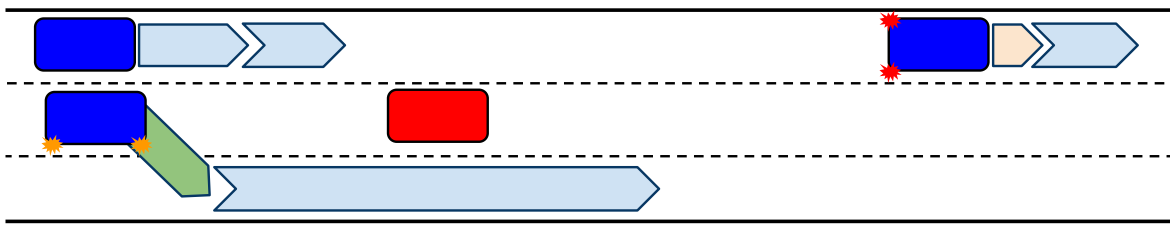


Figure 6.1: Red car is an agent controlled vehicle. The predicted plans for the vehicles nearby are shown in this figure. Source [2]

6.1.2 Considered vehicles

In this subsection we will show on the example which cars need to be considered for this method.

For the situation when the vehicle is on the road that contains only one lane in one direction it is only necessary to know the predicted manoeuvres of the nearest car ahead. Predicted manoeuvres of the car behind can be omitted because this car can adjust it's speed to the our vehicle.

This method is reactive because all agents react on what they see and they don't communicate with each other. For example we can take a situation in which there are three cars on the straight road. They keep safe distances between each other so they can safely decrease their speed if necessary. The first car can decrease the speed without caring about cars behind. This is because the second car will have enough time to react. The same principle can be applied to the third car. Situation changes when there are multiple lanes. It is necessary to know the position of the nearest car behind and nearest car ahead in the next lane for the safe lane changing.

Let's summarize which cars nearby are necessary to be considered for the Safe-distance method. We need to know the predicted manoeuvres of the car ahead, the car in left and right lane ahead and cars left and right behind if these lanes exist. These predicted manoeuvres are stored in the car's State space. The State space is basically an object in which all the nearest vehicles predicted manoeuvres in all listed directions are saved. Example situation is shown in Figure 6.2. Considered vehicles are highlighted.

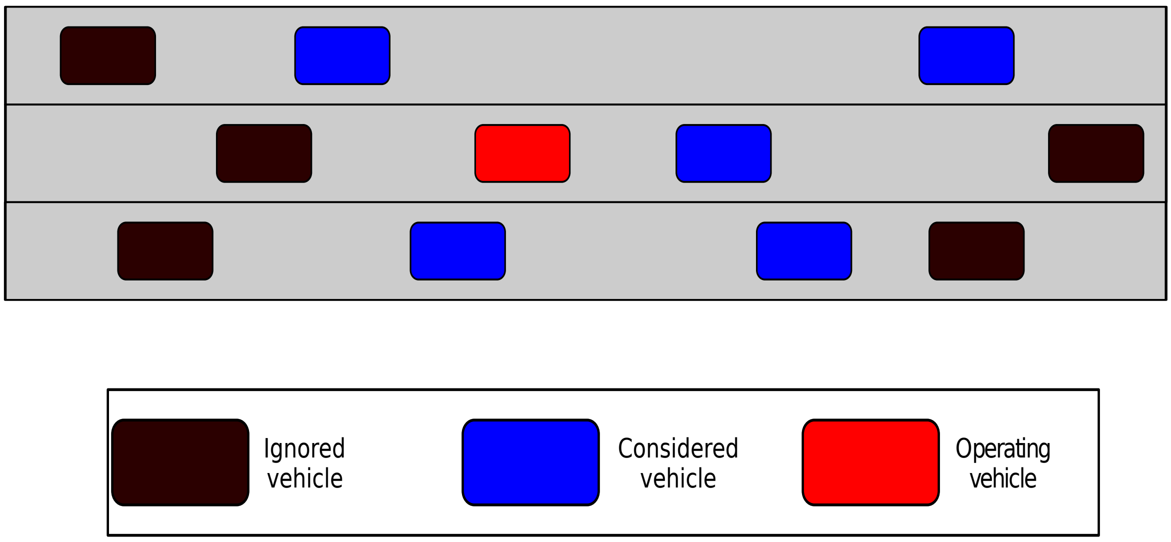


Figure 6.2: Considered vehicles

6.1.3 Planning loop

To be able to plan safe manoeuvres we need to define what is the safe manoeuvre. Safe manoeuvre is that manoeuvre which does not collide with the road structure (More in Section 6.1.4) or another vehicles. (More in Section 6.1.5). Simple plan for example can consist of two manoeuvres.

1. Decrease an agent's car speed.
2. Switch to the next lane to avoid a collision with an object on the road.

Since this method is reactive we can have only limited knowledge what are the intentions of the vehicles in the State space as mentioned in 6.1.2. In order to achieve save planning we need to constantly update our plans to the situation changes on the road.

This method is designed to in every moment every vehicle has a possibility to do a safe manoeuvre if the starting situation was safe. In the Figure 6.3 the control loop is described. From the list of possible manoeuvres one manoeuvre is taken and it is checked if collides with the road structure. If collides, another manoeuvre is tested. If not, it is checked if collides with another vehicles. Also if collides, another manoeuvre is checked. If does not collide, it is executed.

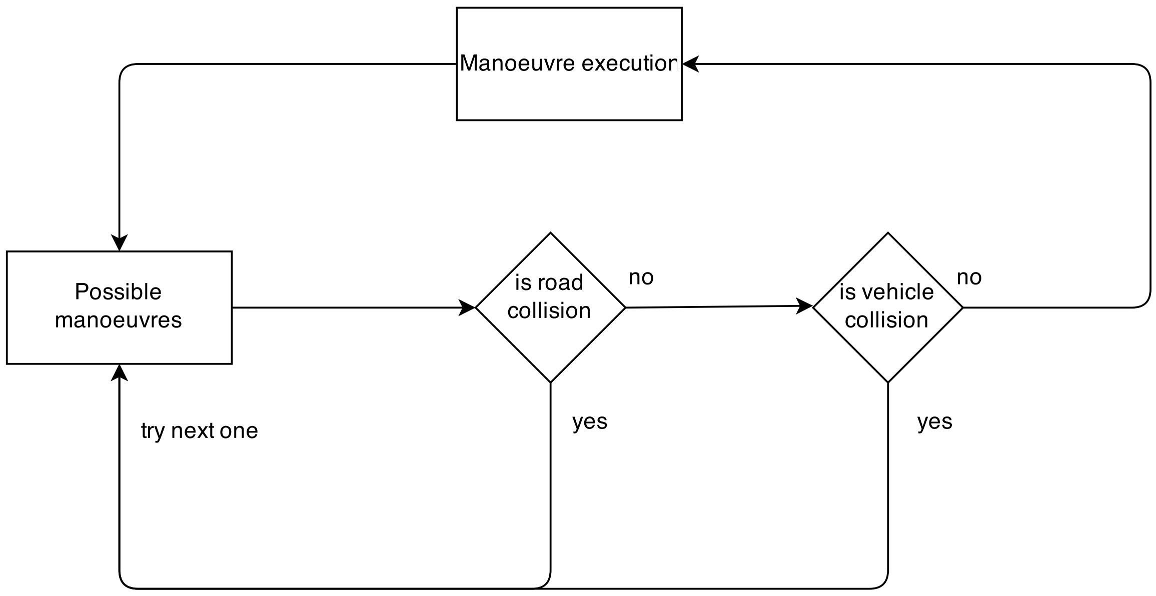


Figure 6.3: Planning loop scheme. Source [2]

Possible manoeuvres are sorted in this order to achieve maximal possible speed:

1. If the last manoeuvre started switching lanes, try to finish the lane switch.
2. Lane right manoeuvre.
3. Acceleration manoeuvre.
4. Straight manoeuvre.
5. Lane left manoeuvre.
6. Deceleration manoeuvre.

6.1.4 Detection of collision with road structure

There are two main conditions that need to be satisfied in order not to collide with the road structure.

1. The destination lane of the manoeuvre must be on the lane that exists.
2. If the lane is ending, it must be possible to stop the vehicle or switch the lane before the end of the lane.

The manoeuvre is safe if from its ending state it is possible to perform any safe manoeuvre. Safe manoeuvres are marked green and unsafe manoeuvres are marked red in this Figure 6.4.

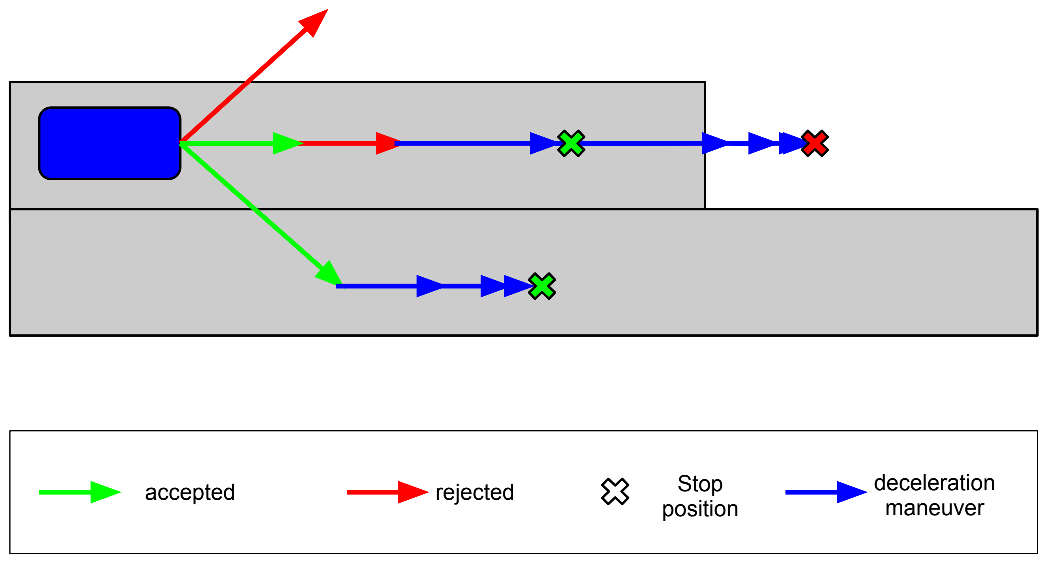


Figure 6.4: This figure shows comparison of safe manoeuvres marked green and unsafe manoeuvres marked red. this figure is from [2].

6.1.5 Vehicle collision detection

The vehicle avoidance of this method is based on maintaining safe distance between vehicles. Every manoeuvre is checked if the vehicle in the end of a manoeuvre has safe distance to the considered vehicles (6.1.2). The safe distance need to be maintained in order not to endanger surrounding vehicles. If the calculated minimal safe distance is greater than the real distance manoeuvre is not safe.

Safe distance calculation is a crucial part of the original Safe-distance method introduced in [2]. This calculation is also used in our Generalized Safe-distance method (6.2). The Safe-distance method checks the worst situation that can occur. The safe distance is calculated from predicted manoeuvre and our manoeuvre candidate. For the safe distance calculation between vehicles manoeuvres representation of vehicles is used (5.5.1). Operating vehicle is represented by the planned manoeuvre and the next vehicle is represented by the predicted manoeuvre. Minimal safe distance is calculated between these two manoeuvres. Every manoeuvre has a constant acceleration. Safe distance is derived from an idea that vehicles cannot collide if they have the same speed. Manoeuvres velocities are explained in Figure 6.5.

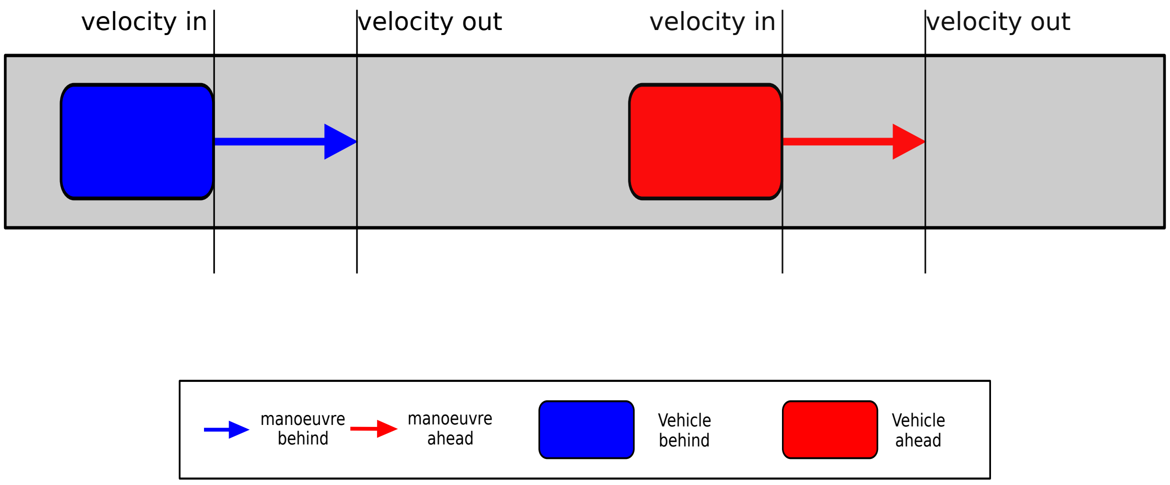


Figure 6.5: Vehicles manoeuvres velocities explained.

Calculation of minimal safe distance d_s is explained bellow:

$$v_{AOut} = v_{BIn}$$

$$v_{AIn} + a \cdot T_M = v_{AOut}$$

$$T_M = \frac{v_{AOut} - v_{AIn}}{a}$$

$$d_s = v_{AIn} \cdot T_M + \frac{1}{2} \cdot a \cdot T_M^2$$

$$d_s = v_{AIn} \cdot \frac{v_{AOut} - v_{AIn}}{a} + \frac{1}{2} \cdot a \cdot (\frac{v_{AOut} - v_{AIn}}{a})^2$$

$$d_s = \frac{2v_{AIn}(v_{AOut} - v_{AIn}) + (v_{AOut} - v_{AIn})^2}{2a}$$

$$d_s = \frac{2v_{AIn}v_{AOut} - 2v_{AIn}^2 + v_{AOut}^2 - 2v_{AOut}v_{AIn} + v_{AIn}^2}{2a}$$

$$d_s = \frac{v_{AOut}^2 - v_{AIn}^2}{2a}$$

$$d_s = \frac{v_{BIn}^2 - v_{AIn}^2}{2a}$$

v_{AIn} ... velocity in of the manoeuvre behind.
v_{AOut} ... velocity out of the manoeuvre behind
v_{BIn} ... velocity in of the manoeuvre ahead.
a ... acceleration constant of a manoeuvre.
T_M ... duration of manoeuvre.

To this calculated minimal safe distance we add safety reserve. It is because minimal safe distance is calculated to the one specific location. To respect that vehicles are not only points the safety reserve is added.

Straight manoeuvres check. When the manoeuvre is straight safe distance need to be calculated only to the vehicle ahead (Why explained in 6.1.2). The situation is shown in the Figure 6.6.

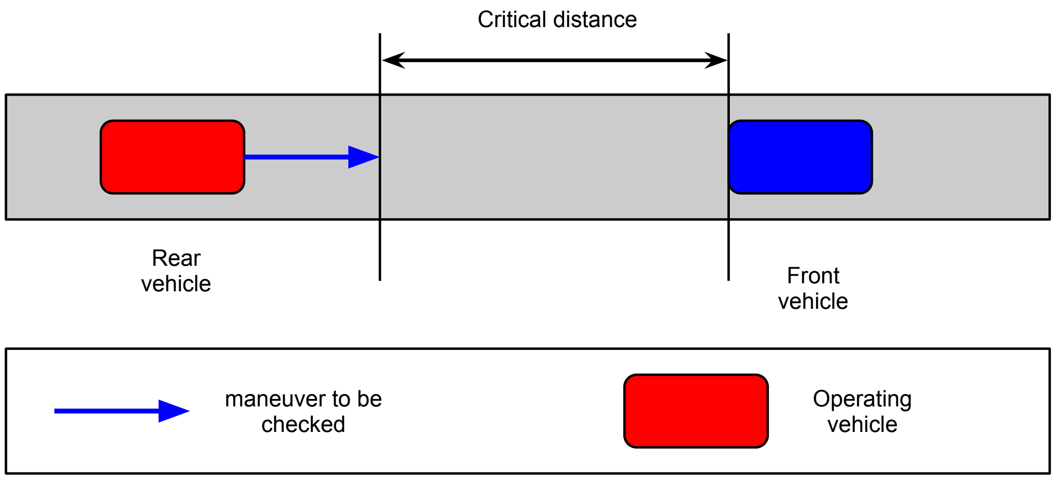


Figure 6.6: Straight manoeuvre check. Source [2]

Lane changing manoeuvres check. When the manoeuvre is one of the lane changing manoeuvres safe distance need to be calculated to the both most close vehicles in the target lane (Why explained in 6.1.2). The situation is shown in the Figure 6.7.

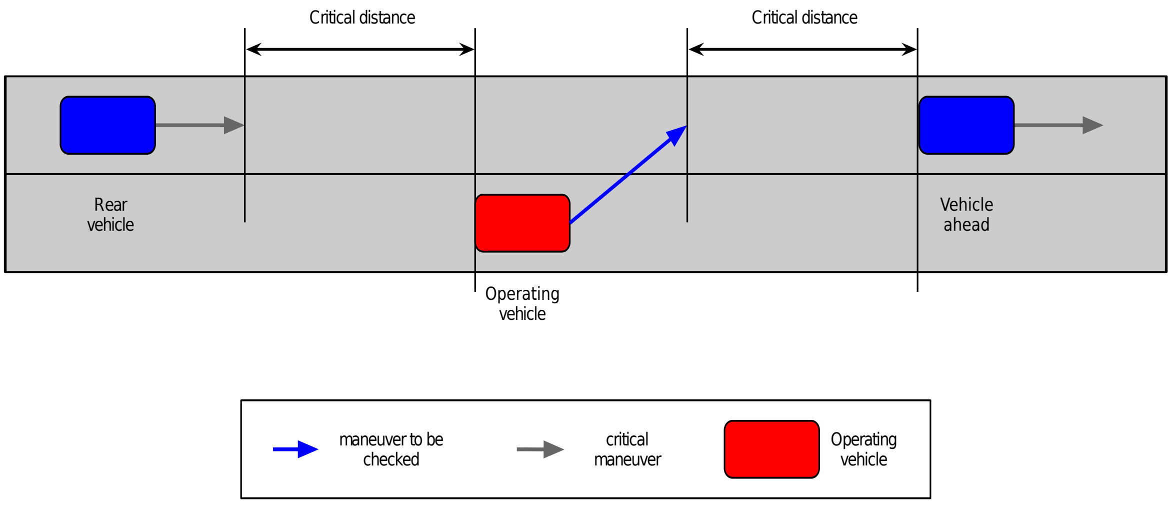


Figure 6.7: Lane changing manoeuvre situation. Edited image from [2]

.

6.2 Generalised Safe-distance method

In this section we will describe the Safe-distance method generalized for the whole road network. We call it Generalized Safe-distance method. The Generalised method uses original Safe-distance method as a base and extends its ability to the whole road network. The basic idea behind is to use Safe-distance method as a ACC mentioned in 4.2. We will now describe how Generalised Safe-distance Agent (GSD Agent) works.

The GSD Agent is build as an extension of the Route agent. We now describe how the basic Route Agent work in order to be able to describe GSD Agent itself.

Route Agent The Route Agent is an agent that operates on the waypoint layer (5.5.2). The Route Agent has a Route Navigator that has vehicle's routes as a set of edges. The Route Navigator also holds the most closer waypoint on the vehicles lane. The Route Navigator is also responsible for lane switching and switching between edges. The Route Agent plans waypoints according to the Route network and vehicle's route. Distance between vehicles is calculated as a distance between two most close road waypoints to the vehicles minus safety reserve.

The Route Agent is used to translate manoeuvres from the GSD Agent. The GSD Agent only plans in short term and it is not aware of the shape of the road network, this is why the Route Agent is a base of the GSD Agent. The Route Agent takes in account the shape of the road, so manoeuvres speed can be overwritten by a lower speed when for example vehicle is approaching the 80 degrees curve. Lane switching manoeuvres are never overwritten however Route Agent can switch lanes and edges when it is necessary (a junction, a funnel on the highway).

GSD Agent When the vehicle is approaching a junction, special Junction mode is enabled. The distance is calculated from the maximum allowed speed on incoming lanes (roads) to the junction so the distance varies junction to junction. In this mode lane changing manoeuvres are disabled. Lane switching is now handled by Route Agent. This switch is however always safe because this can happen only when the vehicle is near the junction and our method guaranties safe lane switching near the junction because the switch is always made at the last possible moment. When the vehicle is in Junction mode it scans not only vehicles on the same road but also vehicles heading to the same junction and vehicles leaving the junction in the same direction as an operating vehicle.

When a vehicle approaching the junction is found, its distance to the junction is calculated.The distance to the junction of the operating vehicle is compared to the other vehicles. If it is greater, this vehicle's predicted manoeuvres is set as a vehicle ahead in the Safe-distances state. If there is already a manoeuvre in the Safe-distance state, the closer one is used.

If the vehicle is leaving the junction at the same edge as it is in my plan only the distance between vehicles is calculated and is set as a vehicle ahead in the Safe-distances states if it is the closest one. So the operating vehicle can create also the safe distance to the vehicle that is already leaving the junction. This is needed in the situation for example where the vehicle stops behind the junction for some reason.

As proposed in Section 4.3, there are situations in which there can be more vehicles in the junction. Operating vehicle does not need to consider vehicles that does not cross its path. In this case, the sensor of the operating vehicle is used to obtain long-term plans of vehicles nearby. This is an additional sensor's feature that uses the V2V communication. From the plans of vehicles nearby it is possible to determine if the plans collide in the junction. If the plans do not collide the vehicle is ignored. If they collide, we calculate approximated place of collision using knowledge of the road network. We create a line segment from the last waypoint before the junction and the first waypoint after the junction for each vehicle. Cross of these line segments is an approximate point of collision. We use this point for calculation to the junction rather than junction centre because it is a better approximation of the place of collision as it can be seen in in Figure 6.8. Example junction can be visible in Figure 6.9.

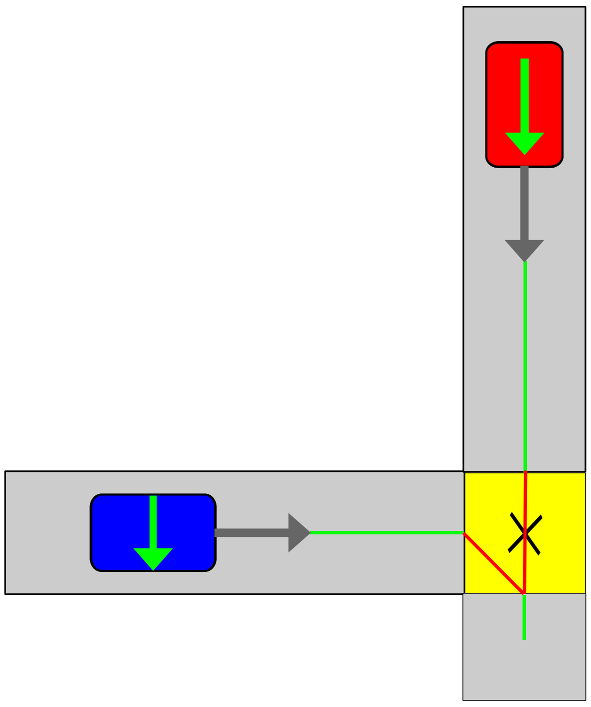


Figure 6.8: Junction centre vs estimated collision point. This Figure shows that the location of the junction centre and the point of collision differers. So for the better estimation of the point of collision we use the intersection of the line segments between the last waipoint before the junction and the first waipoint after the junction. This is still only the approximation, but it is more accurate than the junction centre.

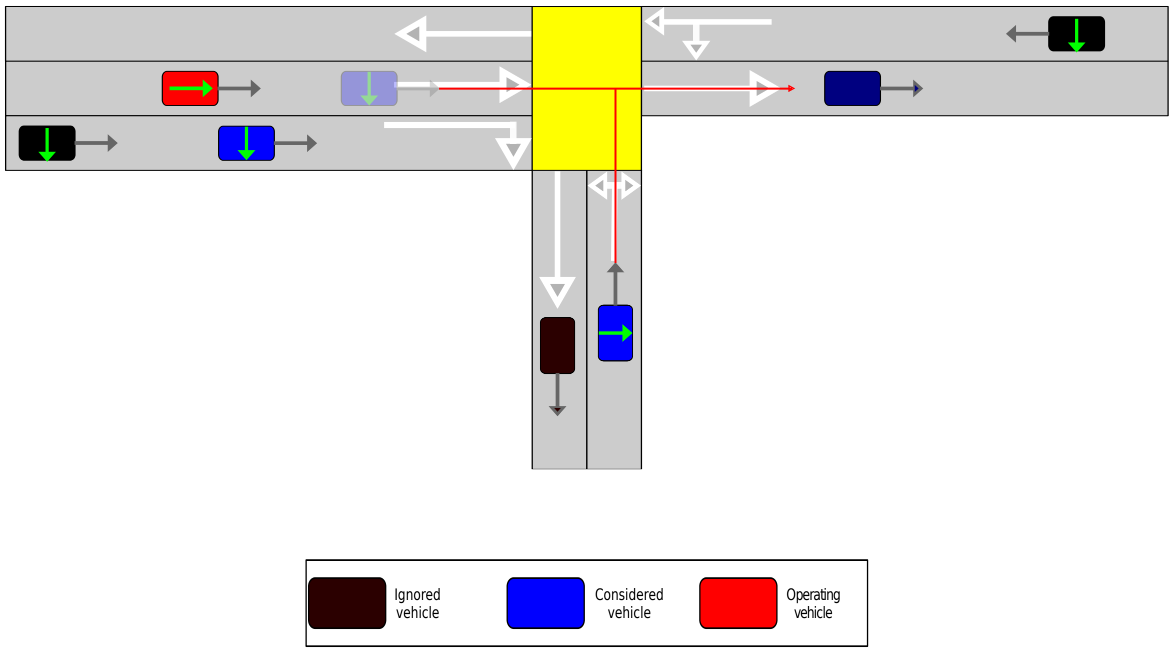


Figure 6.9: GSD Agent reasoning example in T junction. Operating vehicle (the red one) needs to consider two vehicles in this situation. The vehicle in the lane next to him and the closest vehicle ahead of him that is also approaching the junction. In this case it is the blue one at the bottom. Position of this vehicle is transformed to the transparent position visible in the Figure. Operating vehicle now can adjust its speed to this vehicle and can safely pass the junction afterwards. The dark blue vehicle is not considered by the operating vehicle, but the blue vehicle on the bottom must consider this vehicle as a vehicle ahead.

Chapter 7

Evaluation

We test our method to show that is non-collisional in our testing environment. We also like to show, that it requires only low CPU power for calculations. We used the Simulator Lite for the evaluation of the method. We tested our method mostly on the deterministic scenarios, so it was sufficient to run the simulation only once on them. We also included three non-deterministic scenarios. They were non-deterministic because vehicles went on random routes. However first two converges to the same result and the third one was a proof of a concept scenario on the real road network taken from the OpenStreetMap[1].

The tested scenarios were picked to represent the most common scenarios on the road network. In all scenarios these values were measured:

- number of collisions
- the total average speed.
- the average speed of every route.
- the time needed for calculations of all plans. (However this varies by the hardware used).
- time needed for every vehicle to pass the scenario.
- the distance of all vehicles to the junction centre.
- Number of vehicles travelling through the junction per second.

Some of these values were plotted to the graphs using Matlab to visualize the results for the better understanding.

7.1 Scenarios description

There are four main scenarios presented in this section. First one is the Highway (7.2). Second one is the T junction (7.3) Third one is the full X junction (7.4) and the fourth is

[1]`https://www.openstreetmap.org/`

the scenario from the real road structure (7.5). There was one situation recorded from all four scenarios. These videos can be found on YouTube[2] or on the attached CD. Last but not least there is a simple scenario for better understanding how proposed method works bellow. The vehicles are removed from the simulation when they reach approximately 1100 metres. Every lane is 600 metres long so the distance to the junction is 600 metres.

Simple testing scenario This scenario is the X junction. 5 vehicles are heading from each of all directions. Some of them turns, some of them go straight. As it can be seen from the Figure 7.2, all vehicles sorted themselves on the road to the junction and passed the junction safely. The reaction between the vehicles is well visible as all vehicles were forced to adjust their speed to the fourth vehicle which was turning right. It can be clearly seen from which distance this method was activated. The screenshot from this scenario is in Figure 7.1.

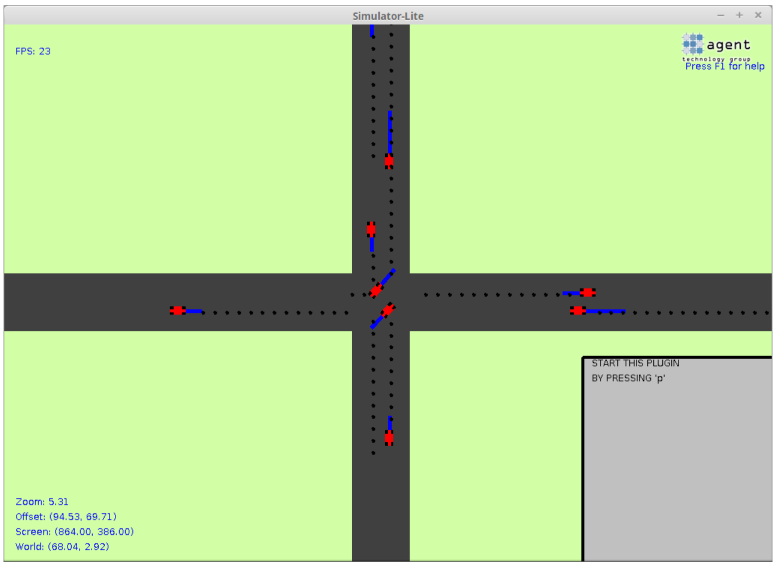


Figure 7.1: Simple testing junction, There are five vehicles in every direction in distance of 600 metres to the junction. The black points are the waipoints and these blue rectangles are the velocity vectors.

[2]https://www.youtube.com/playlist?list=PLw4P2lxgt3ma3bv3zdn28HZU8MDV-yCrI

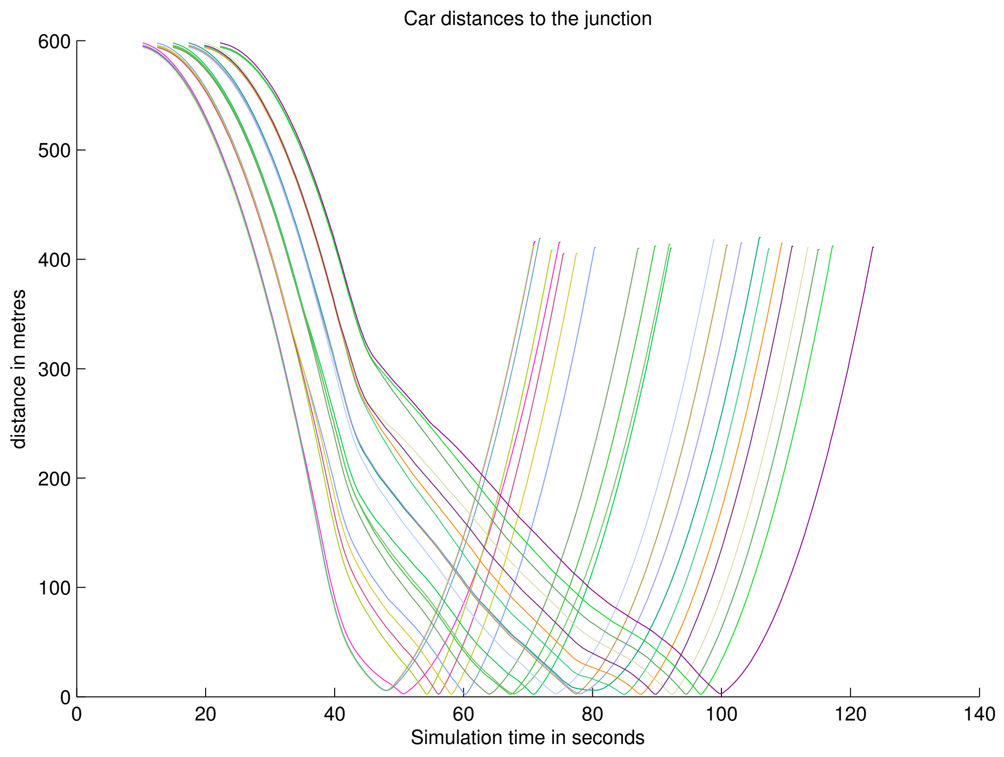


Figure 7.2: This figure shows every vehicles distances to the junction in the selected time. Vehicles go straight or turn right or left. The Junction mode was activated 400 metres from the junction. Safety distance reserve is 10 metres. All vehicles started with zero velocity. Turning right or left requires quite big speed reduction because the angle which vehicles need to pass is 90 degrees. As it can be seen in the figure, first three vehicles were able to pass junction simultaneously as their routes did not collide. However fourth vehicle needed to slow down to pass the junction afterwards so every vehicle behind this one needed to slow down to reflect this. More simultaneous passing the junction and the creating of safe distances between the vehicles is well visible in this figure. Number of collisions is 0.

7.2 Highway

This section consists of three scenarios. First is a simple straight road with one lane (7.2.1). Second is a simple highway with two lanes (7.2.2) and third is a highway with 2 lanes merging into one (7.2.3). Vehicles are generated and added on the start as soon as it is possible. The purpose of these experiments is to get maximum capacity of these roads and average speeds. 80 vehicles were send to every lane.

7.2.1 Road with one lane

Number of vehicles at road is visible in Figure (7.3).

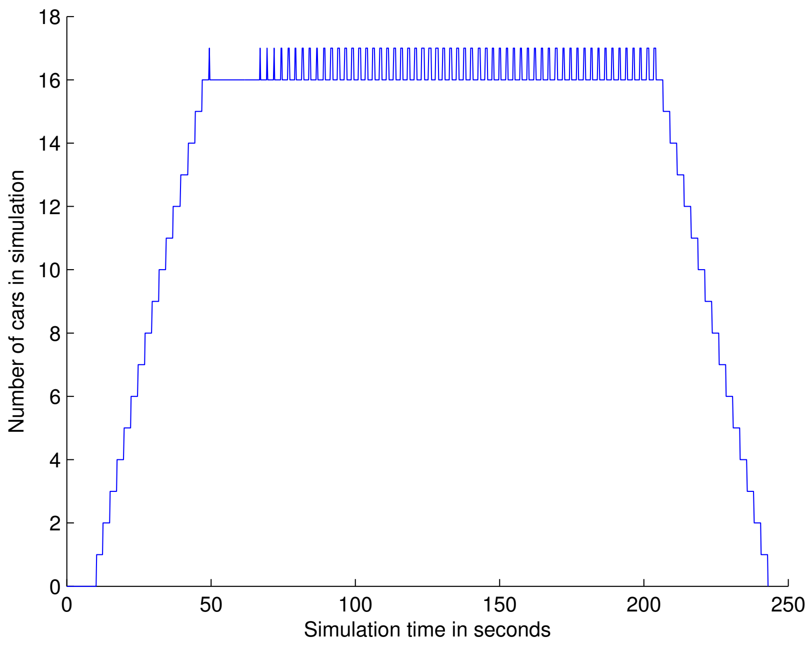


Figure 7.3: Number of vehicles in simulation in time. In this case on the road with one lane.

The maximum capacity of the road is around 16 vehicles. The average speed was 23.62 metres per second. Number of collisions is 0.

7.2.2 Highway with two lanes

Number of vehicles at road is visible in Figure 7.4.

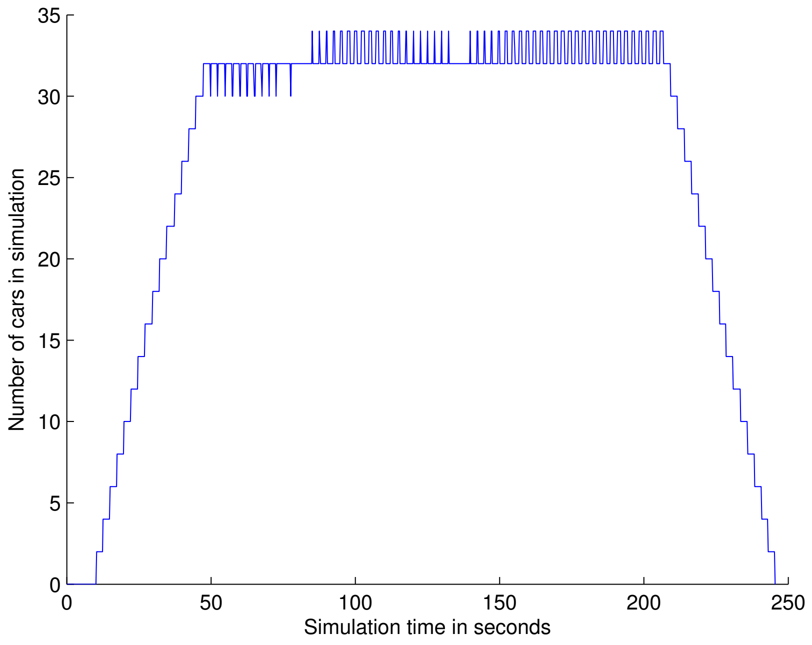


Figure 7.4: Number of vehicles in simulation in time. In this case on the highway with two lanes.

The maximum capacity of the highway is around 33 vehicles. The average speed was 23.45 metres per second. The number of vehicles was nearly two times the number of vehicles on the road with one lane. The average speed was a bit lower. This happens because vehicles try to be in the right lane. Number of collisions is 0.

7.2.3 Funnel

This situation consists of the straight highway with a funnel in the middle there two lanes merge into one. First scenario is with the Junction mode enabled from the beginning (600 metres before the junction), in the second one the Junction mode is enabled 400 metres before the junction. We compare these two results and the results from the road with one lane and the highway with two lanes. Third one is the scenario when the Junction mode is also enabled 400 metres before the junction but the number of incoming vehicles is doubled. We compare these results to the second one.

7.2.3.1 Junction mode is enabled from the beginning

Number of vehicles at road is visible in the Figure 7.5.

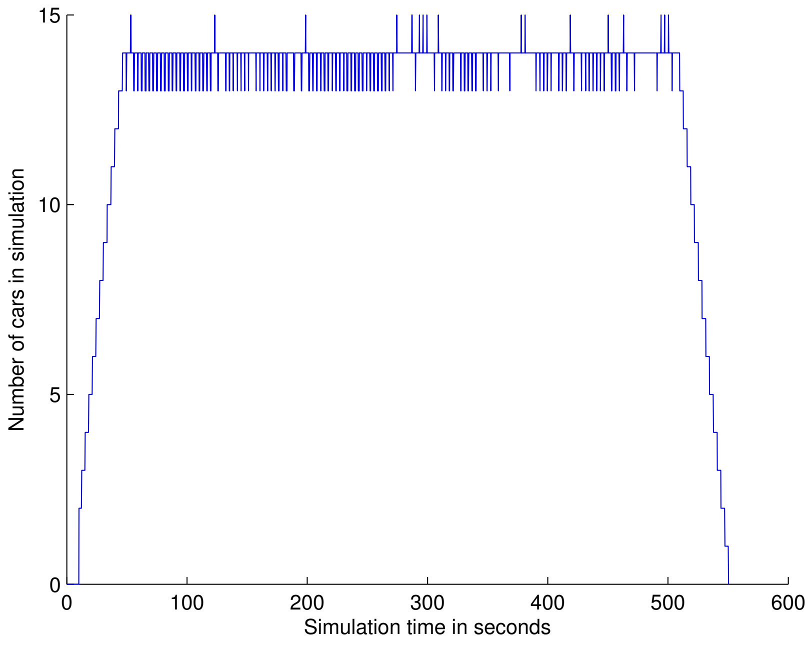


Figure 7.5: Number of vehicles in simulation in time. In this case on the highway with a funnel.

The maximum capacity of this highway with funnel is around 14 vehicles. The average speed was 21.43 metres per second. When the Junction mode was activated from the start, the results were similar as the road with one lane. Number of collisions is 0.

7.2.3.2 Junction mode is enabled 400 metres to the funnel

Number of vehicles at road is visible in the Figure 7.6.

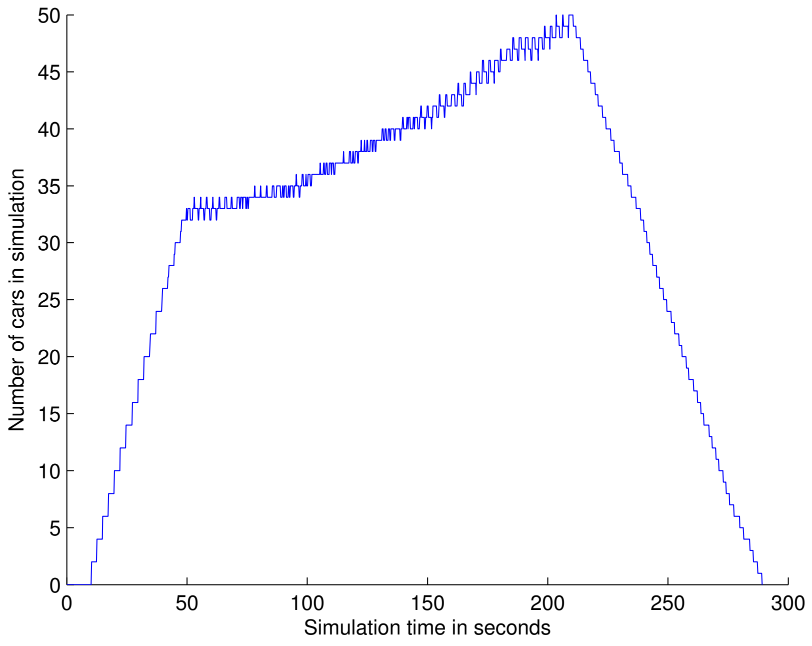


Figure 7.6: Number of vehicles in simulation in time. In this case on the highway with a funnel.

As the Junction mode was activated only 400 metres to the junction and not from the beginning, vehicles got some initial speed before they start to prepare for the funnel. This allowed them to sort themselves quicker than when the mode was activated from the beginning. This is why the simulation time was two times shorter. The average speed refers to the fact that vehicles were forced to decrease their speed in order to be able to merge. This was not the case than the mode was activated from beginning because vehicles were inserted already sorted and prepared for the merge. This is why average speed was 12.34 metres per second. Number of collisions is 0. The average speed was decreasing over time. This can be well seen in the Figure 7.7 that shows how much time was needed for passing the scenario. If the simulation would run a bit longer (e.g. more vehicle would be send into the simulation) a traffic jam would be created from the start to the 200 metres. This can be seen in 7.2.3.3.

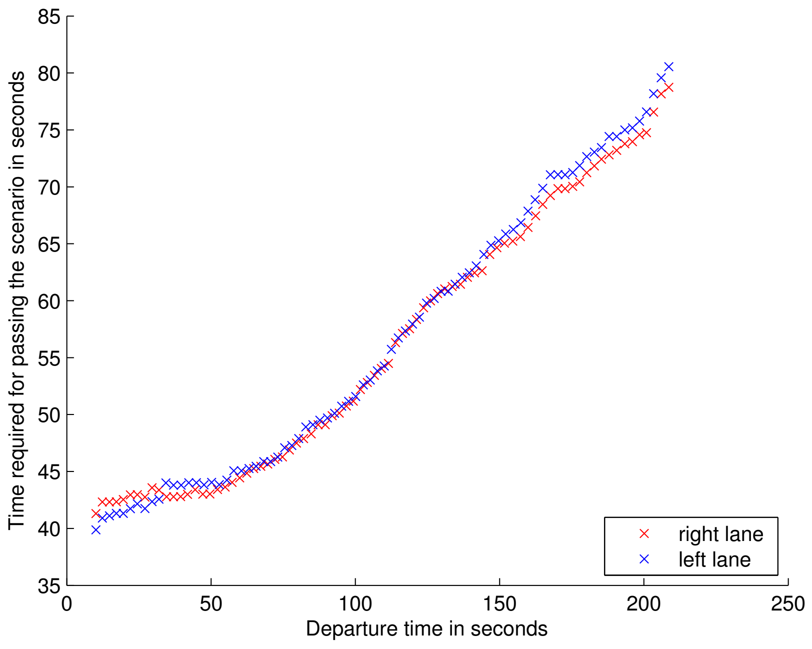


Figure 7.7: Time needed for passing the scenario during the simulation.

7.2.3.3 Junction mode is enabled 400 metres to the funnel with two times more vehicles

Number of vehicles at road is visible in the Figure 7.18.

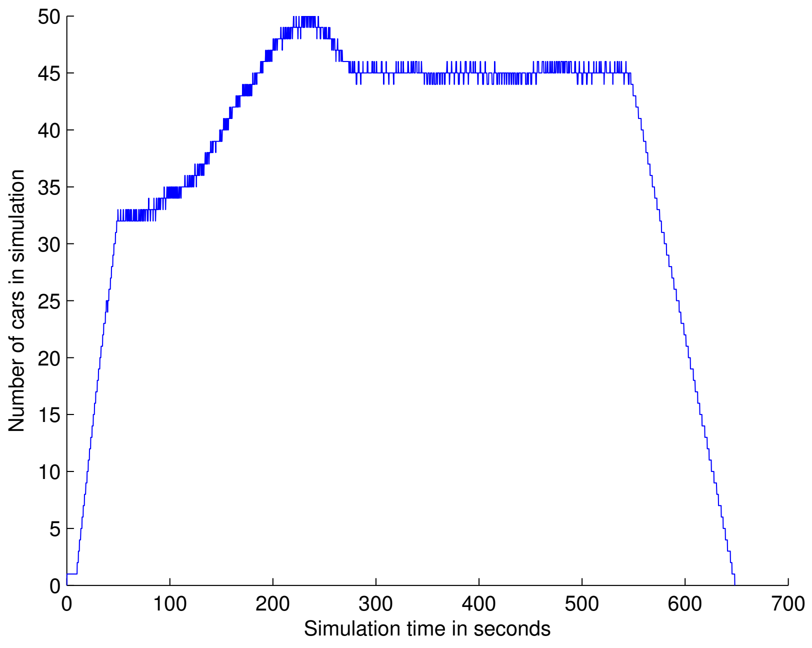


Figure 7.8: Number of vehicles in simulation in time. In this case on the highway with a funnel.

This shows the situation when 320 vehicles were send from start. In this scenario number of vehicles in simulation stabilised on 45 vehicles. Average speed was 14.52 metres per second. In the Figure 7.19 is visible the increase of time needed to pass the scenario until the time needed stabilised. This Figure shows the creation of traffic jam before the place where the Junction mode was activated. It is well visible on the screenshot in the Figure 7.20. The number of vehicles sent to the simulation exceeded capabilities of proposed method. Number of collisions is 0. The video from this scenario is on YouTube[3] and on the attached CD.

[3]`https://www.youtube.com/watch?v=ZTCtWeQTrPQ&index=2&list=PLw4P2lxgt3ma3bv3zdn28HZU8MDV-yCrI`

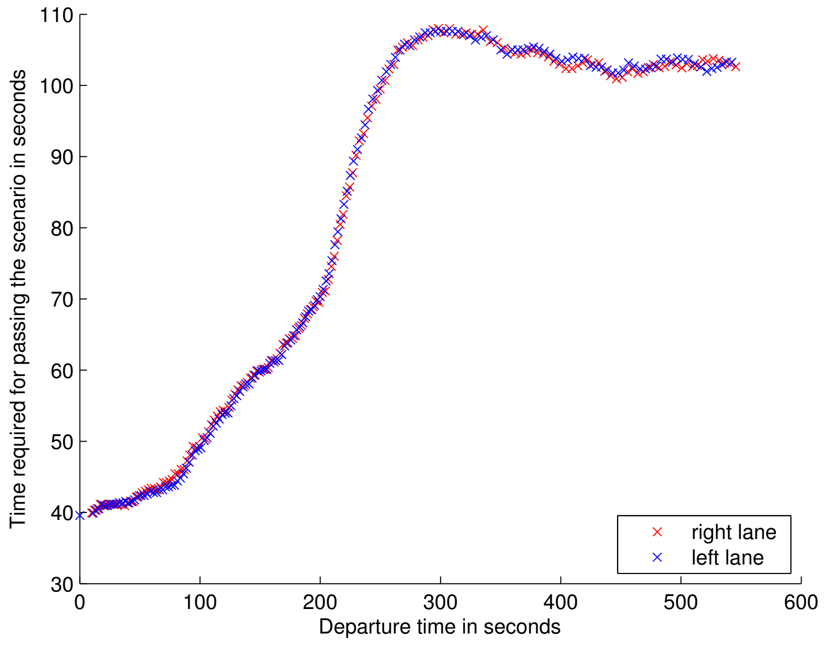
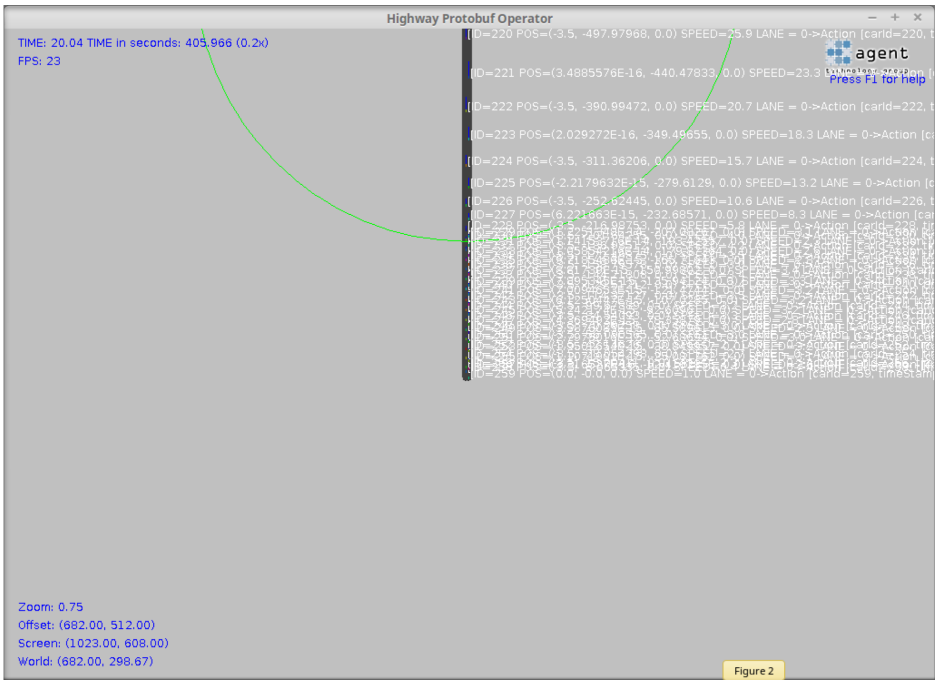


Figure 7.9: Time needed for passing the scenario during the simulation. It is visible that there is no difference between the lanes.



```
TIME: 20.04 TIME in seconds: 405.966 (0.2x)
FPS: 23
```

```
[ID=220 POS=(-3.5, -497.97968, 0.0) SPEED=25.9 LANE = 0->Action [carId=220, t
[ID=221 POS=(3.4885576E-16, -440.47833, 0.0) SPEED=23.3
[ID=222 POS=(-3.5, -390.99472, 0.0) SPEED=20.7 LANE = 0->Action [carId=222, t
[ID=223 POS=(2.029272E-16, -349.49655, 0.0) SPEED=18.3 LANE = 0->Action [ca
[ID=224 POS=(-3.5, -311.36206, 0.0) SPEED=15.7 LANE = 0->Action [carId=224, t
[ID=225 POS=(-2.2179632E-15, -279.6129, 0.0) SPEED=13.2 LANE = 0->Action [c
[ID=226 POS=(-3.5, -252.62445, 0.0) SPEED=10.6 LANE = 0->Action [carId=226, t
[ID=227 POS=(6.221663E-15, -232.68571, 0.0) SPEED=8.3 LANE = 0->Action [car
[ID=228 POS=(-3.5, -216.98753, 0.0) SPEED=5.8 LANE = 0->Action [carId=228, tir
[illegible]
[ID=259 POS=(0.0, -0.0, 0.0) SPEED=1.0 LANE = 0->Action [carId=259, timeStam
```

```
Zoom: 0.75
Offset: (682.00, 512.00)
Screen: (1023.00, 608.00)
World: (682.00, 298.67)
```

Figure 7.10: Traffic jam. The green line shows the point where the Junction mode was activated. Before this point vehicles move slowly.

7.3 T junction

This section consists of three scenarios. First one is the funnel simulation similar to the situation before (7.3.1). The only difference is that angle between merging lanes is now 90 degrees. Second one is the situation when the traffic on the main road and side road are simulated (7.3.2). Third is the scenario where vehicles comes from all directions with same probability (7.3.3). Junction mode is now always enabled 400 metres before the junction.

7.3.1 Funnel

Number of vehicles at road is visible in the Figure 7.11.

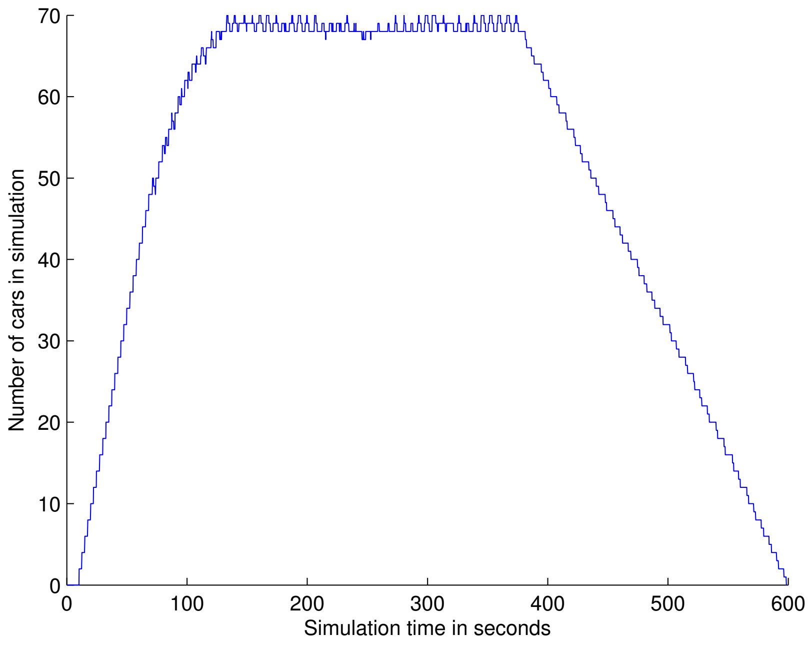


Figure 7.11: Number of vehicles in simulation in time. In this case on the T junction representing a funnel.

This shows the situation when 80 vehicles were send from the bottom lane and 80 vehicles were send from the left lane. All vehicles were heading to the right road. In this scenario number of vehicles in simulation exceeded simulator's limit. Average speed was 6.39 metres per second. In the Figure 7.12 is visible the increase of time needed to pass the scenario until the time needed stabilised. The low average speed in comparison to the similar scenarios in Subsection 7.2.3 was because turning right requires a dramatical speed reduction. Because this method is reactive, vehicles heading from right to left needed to slow down to merge with vehicles coming from the bottom lane. Number of collisions is 0.

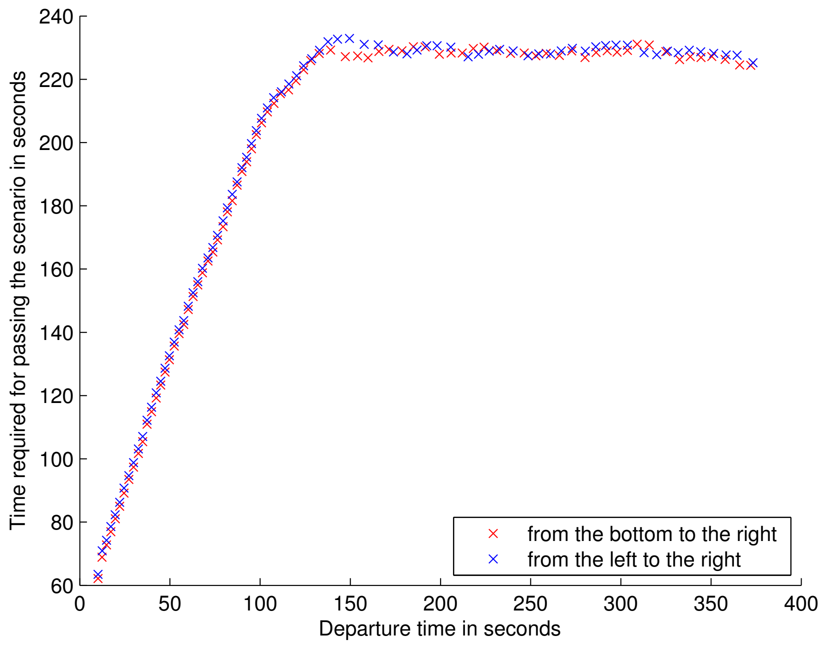


Figure 7.12: Time needed for passing the scenario during the simulation. It is visible that there is no difference between the roads.

7.3.2 More frequent and less frequent road

Number of vehicles at road is visible in the Figure 7.13.

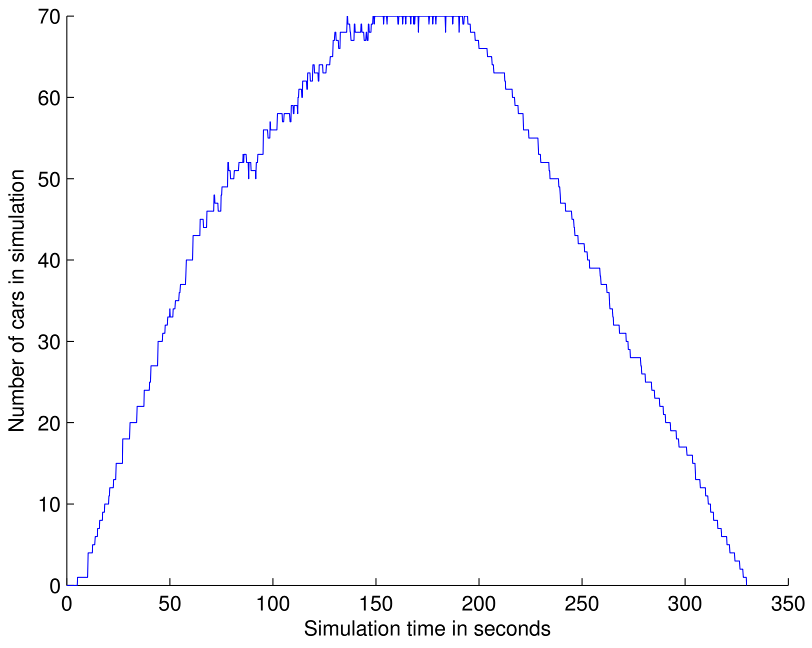


Figure 7.13: Number of vehicles in simulation in time. In this case on the T junction.

This shows the situation when 50 vehicles were send from left to right and another 50 vehicles from right to left. 5 vehicles went from left to bottom and 5 vehicles went from right to bottom. 10 vehicles went from the bottom to the right and 10 vehicles went from the bottom to left. In this scenario number of vehicles in simulation exceeded simulators limit. Average speed was 10.34 metres per second. In the Figure 7.14 is visible the difference of the frequent road and the less frequent side road. The difference is that vehicles from the side road and vehicles that turns slow down vehicles on the frequent road. Number of collisions is 0.

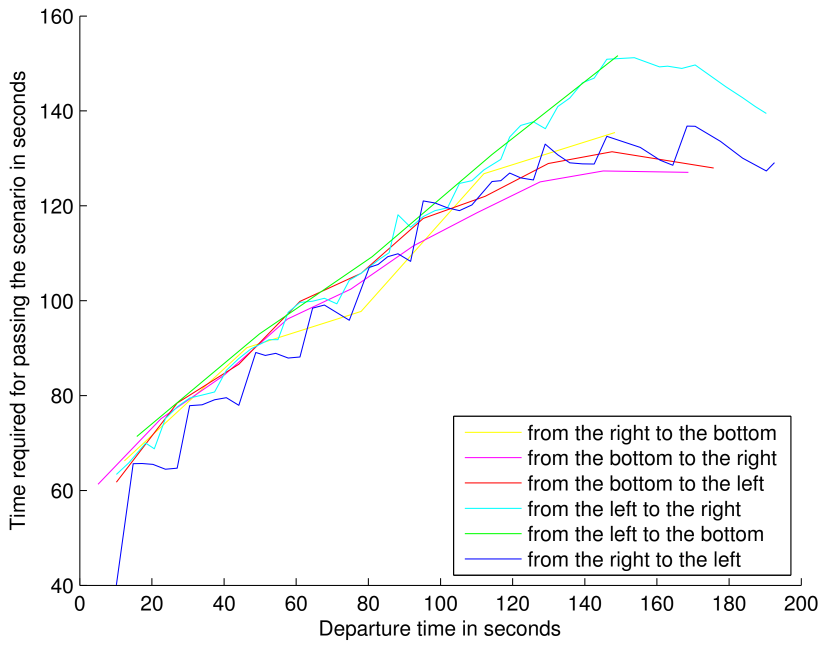


Figure 7.14: Time needed for passing the scenario during the simulation. Points are connected for better visibility.

7.3.3 All roads with the same vehicles frequency

Number of vehicles at road is visible in Figure 7.15.

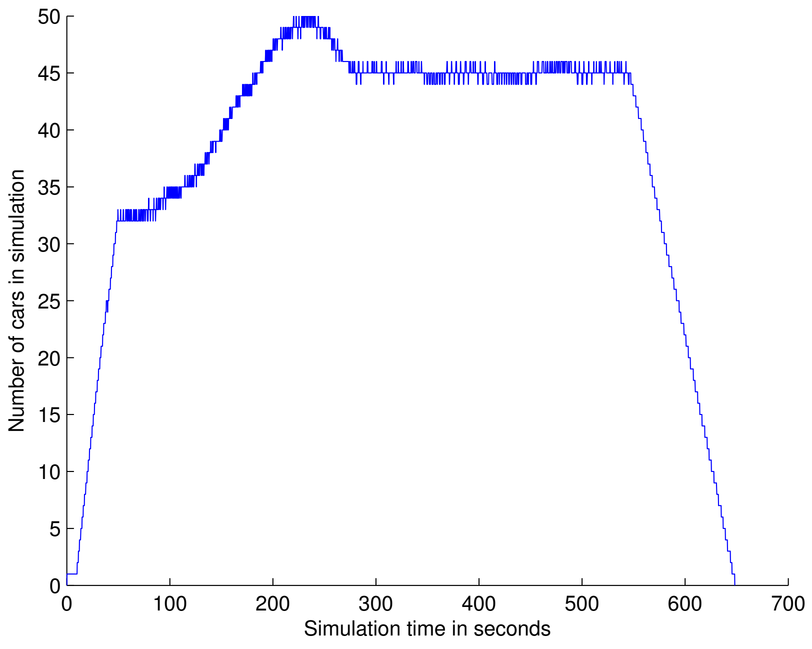


Figure 7.15: Number of vehicles in simulation in time. In this case on the T junction.

Average speed was 14.55. All routes had similar average speed. Number of collisions is 0. The video from this scenario is on YouTube[4] and on the attached CD.

7.4 X junction

This section consists of three scenarios. First one shows how it is easy to turn left on the frequent road when this method is activated (7.4.1). The Second scenario shows vehicles not turning just passing the X junction from different directions (7.4.2) and the Third scenario shows the situation when vehicles come from every direction with the same probability and they can turn or go straight (7.4.3).

[4]`https://www.youtube.com/watch?v=bOT2xqPyWrA&index=4&list=PLw4P2lxgt3ma3bv3zdn28HZU8MDV-yCrI`

7.4.1 Turning left on the frequent road

Number of vehicles at road is visible in the Figure 7.21.

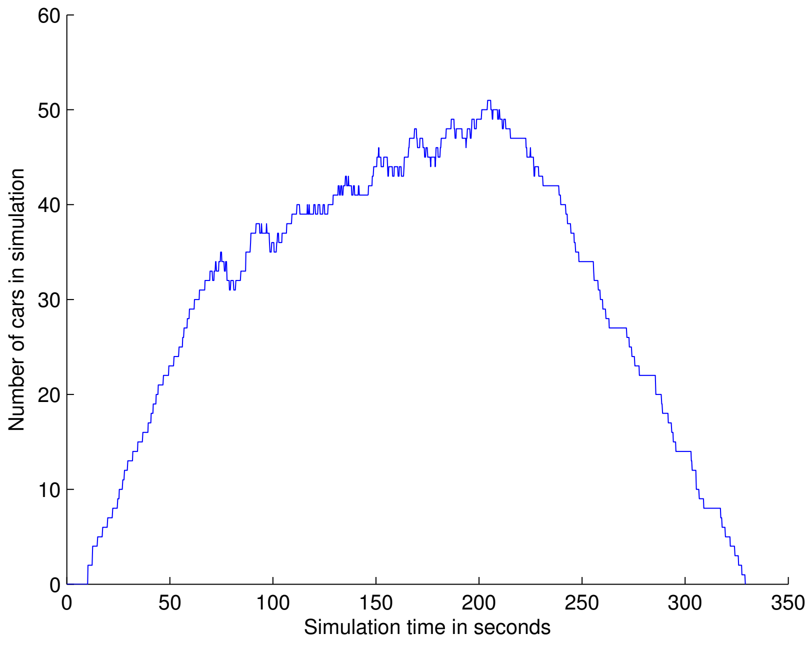


Figure 7.16: Number of vehicles in simulation in time. In this case on the X junction.

The road from left to right represents frequent road. On such road it is nearly impossible to turn left in the real word. This shows the situation when 80 vehicles were send from left to right and 10 vehicles went from right to the top and 10 vehicles went from right to the bottom. Average speed on the frequent road was 11.52 m/s. Average speed of vehicles turning left was 12.61 m/s and for turning right it was 12.61 m/s. The results show nearly no difference between turning left or right from the frequent road.

The difference is visible between vehicles on the frequent road and vehicles trying to turn left or right in Figure 7.17. Vehicles turning right had no need to avoid cars coming from left. Number of collisions is 0.

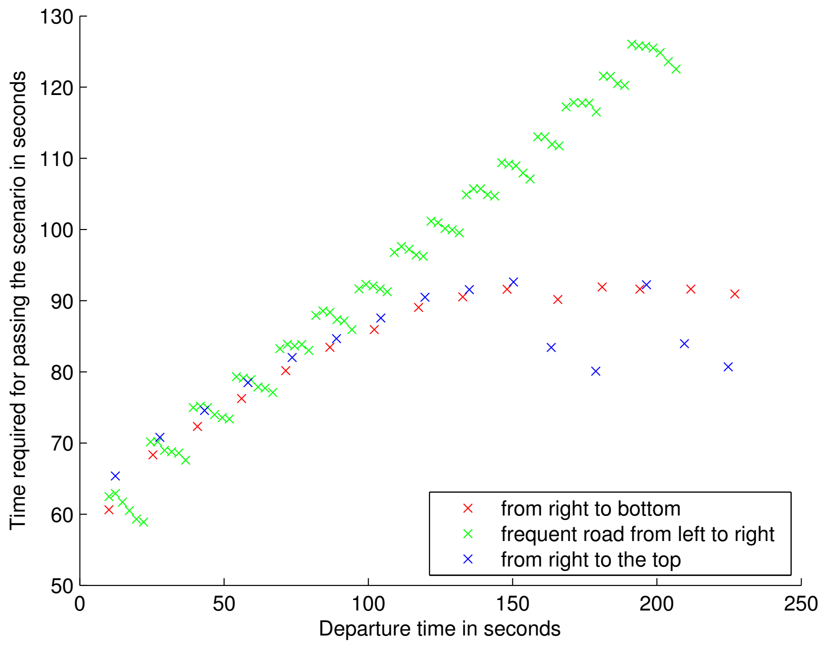


Figure 7.17: Time needed for passing the scenario during the simulation.

7.4.2 Simple cross

Number of vehicles at road is visible in the Figure 7.18.

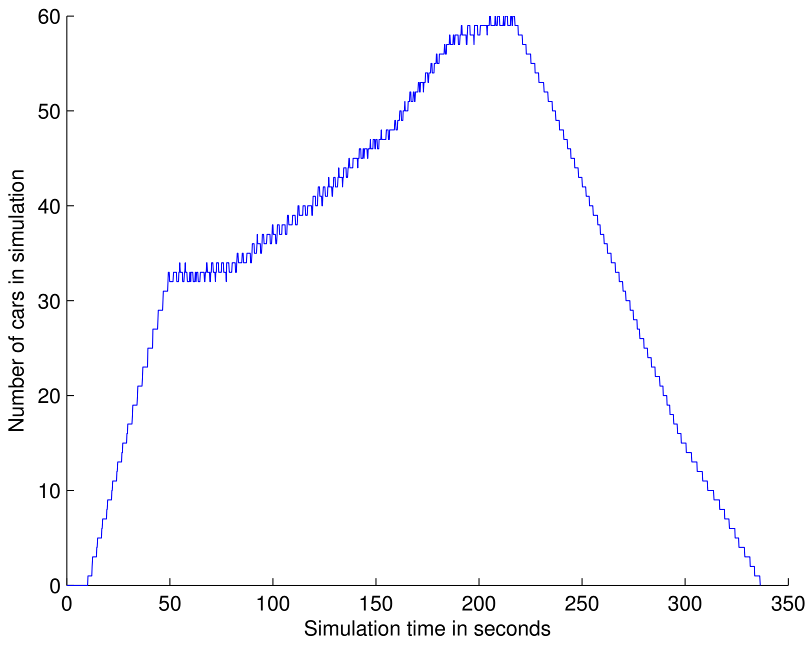


Figure 7.18: Number of vehicles in simulation in time. In this case on the X junction.

This shows the situation when 160 vehicles were send from the bottom the the up and 160 vehicles were send from the right to the left. In this scenario number of vehicles in simulation reached maximum on 60 vehicles. Average speed was 15.59 metres per second. In the Figure 7.19 is visible the increase of time needed to pass the scenario. This Figure shows the creation of traffic jam before the place where the junction mode was activated. It is well visible on the screen shot in the Figure 7.20. The number of vehicles sent to the simulation exceeded capabilities of proposed method. Number of collisions is 0.

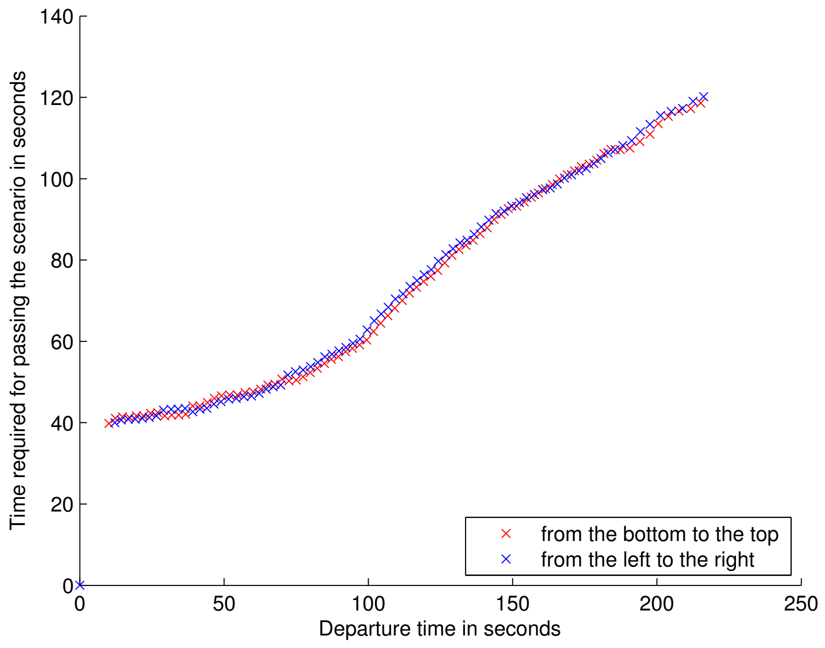
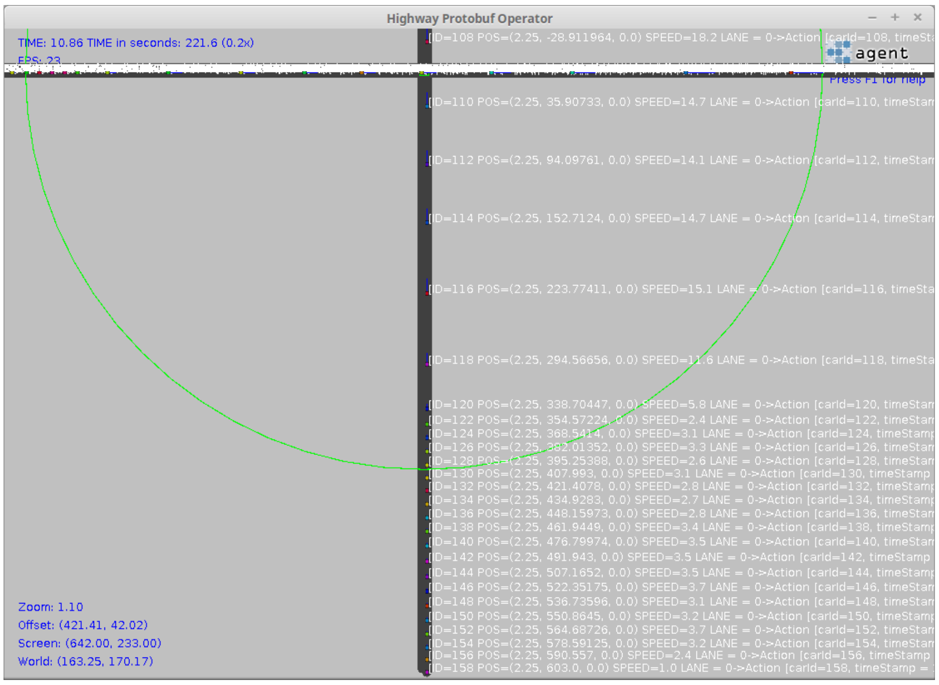


Figure 7.19: Arrival times



```
TIME: 10.86 TIME in seconds: 221.6 (0.2x)
FPS: 23

Zoom: 1.10
Offset: (421.41, 42.02)
Screen: (642.00, 233.00)
World: (163.25, 170.17)
```

```
[ID=108 POS=(2.25, -28.911964, 0.0) SPEED=18.2 LANE = 0->Action [carId=108, timeSta
[ID=110 POS=(2.25, 35.90733, 0.0) SPEED=14.7 LANE = 0->Action [carId=110, timeStam
[ID=112 POS=(2.25, 94.09761, 0.0) SPEED=14.1 LANE = 0->Action [carId=112, timeStam
[ID=114 POS=(2.25, 152.7124, 0.0) SPEED=14.7 LANE = 0->Action [carId=114, timeStam
[ID=116 POS=(2.25, 223.77411, 0.0) SPEED=15.1 LANE = 0->Action [carId=116, timeSta
[ID=118 POS=(2.25, 294.56656, 0.0) SPEED=11.6 LANE = 0->Action [carId=118, timeSta
[ID=120 POS=(2.25, 338.70447, 0.0) SPEED=5.8 LANE = 0->Action [carId=120, timeStam
[ID=122 POS=(2.25, 354.57224, 0.0) SPEED=2.4 LANE = 0->Action [carId=122, timeStam
[ID=124 POS=(2.25, 368.5414, 0.0) SPEED=3.1 LANE = 0->Action [carId=124, timeStamp
[ID=126 POS=(2.25, 382.01352, 0.0) SPEED=3.3 LANE = 0->Action [carId=126, timeStam
[ID=128 POS=(2.25, 395.25388, 0.0) SPEED=2.6 LANE = 0->Action [carId=128, timeStam
[ID=130 POS=(2.25, 407.993, 0.0) SPEED=3.1 LANE = 0->Action [carId=130, timeStamp
[ID=132 POS=(2.25, 421.4078, 0.0) SPEED=2.8 LANE = 0->Action [carId=132, timeStamp
[ID=134 POS=(2.25, 434.9283, 0.0) SPEED=2.7 LANE = 0->Action [carId=134, timeStamp
[ID=136 POS=(2.25, 448.15973, 0.0) SPEED=2.8 LANE = 0->Action [carId=136, timeStam
[ID=138 POS=(2.25, 461.9449, 0.0) SPEED=3.4 LANE = 0->Action [carId=138, timeStamp
[ID=140 POS=(2.25, 476.79974, 0.0) SPEED=3.5 LANE = 0->Action [carId=140, timeStam
[ID=142 POS=(2.25, 491.943, 0.0) SPEED=3.5 LANE = 0->Action [carId=142, timeStamp
[ID=144 POS=(2.25, 507.1652, 0.0) SPEED=3.5 LANE = 0->Action [carId=144, timeStamp
[ID=146 POS=(2.25, 522.35175, 0.0) SPEED=3.7 LANE = 0->Action [carId=146, timeStam
[ID=148 POS=(2.25, 536.73596, 0.0) SPEED=3.1 LANE = 0->Action [carId=148, timeStam
[ID=150 POS=(2.25, 550.8645, 0.0) SPEED=3.2 LANE = 0->Action [carId=150, timeStamp
[ID=152 POS=(2.25, 564.68726, 0.0) SPEED=3.7 LANE = 0->Action [carId=152, timeStam
[ID=154 POS=(2.25, 578.59125, 0.0) SPEED=3.2 LANE = 0->Action [carId=154, timeStam
[ID=156 POS=(2.25, 590.557, 0.0) SPEED=2.4 LANE = 0->Action [carId=156, timeStamp
[ID=158 POS=(2.25, 603.0, 0.0) SPEED=1.0 LANE = 0->Action [carId=158, timeStamp =
```

Figure 7.20: Traffic jam. The green line shows the point where the Junction mode was activated.

7.4.3 All roads with the same vehicles frequency

Number of vehicles at road is visible in the Figure 7.21.

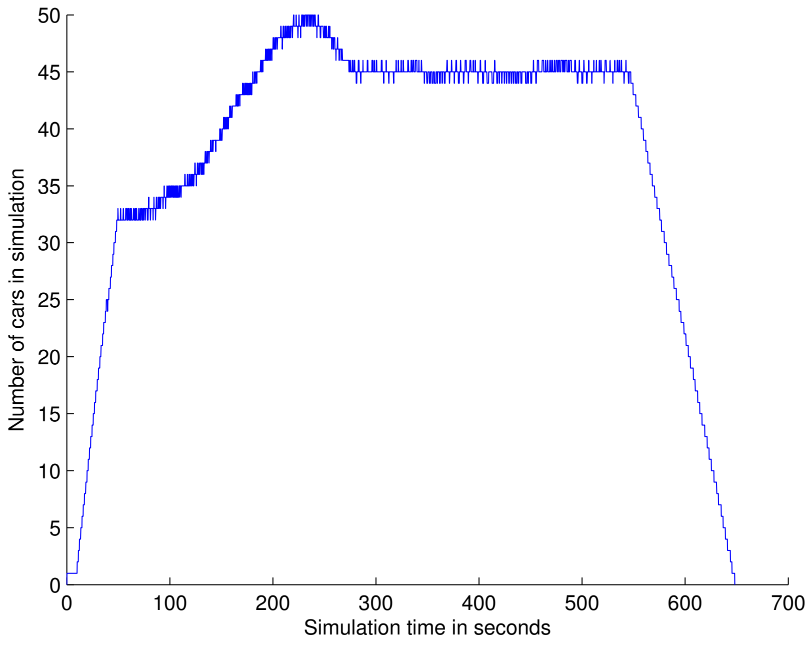


Figure 7.21: Number of vehicles in simulation in time. In this case on the X junction.

Average speed was 14.52. Situation was similar as the same situation on T junction. All vehicles had similar speed. Average speed decreased over time. Number of collisions is 0. The video from this scenario is on YouTube[5] and on the attached CD.

7.5 City map

The map was obtained from OpenStreetMap[6]. It was converted to our structure using the sumo tools[7]. The video from this scenario is on YouTube[8] and on the attached CD. This scenario is shown just to prove that this method also works on the random road network. Screenshots from this scenario can be visible bellow in Figures 7.22, 7.23 and 7.24. The black points are the waipoints and these blue rectangles are the velocity vectors. The road network of this scenario is from a part of Prague called “Košíře”. In this scenario there were several collisions in several runs, particularly because the injection of new vehicles to the real road network is not ideal (vehicles were inserted with a wrong direction) and because some

[5]`https://www.youtube.com/watch?v=krkAydTD2mQ&index=3&list=PLw4P2lxgt3ma3bv3zdn28HZU8MDV-yCrI`

[6]`https://www.openstreetmap.org/`

[7]`http://sumo.dlr.de/wiki/Main_Page`

[8]`https://www.youtube.com/watch?v=vgwAatGeneralisedSafe-distancNYDUg&index=1&list=PLw4P2lxgt3ma3bv3zdn28HZU8MDV-yCrI`

junctions were not big enough to handle more vehicles passing the junction (this is however hardly detectable in our converted road network representation). Also there is a problem that vehicle in the junction is visible as a vehicle leaving the junction for other vehicles. This is because in our representation there are no waypoints in the junctions so it is hard to detect if the vehicle is still in the junction. If for some reason vehicle get stuck in the junction, it is visible as the vehicle that is on the lane leaving the junction. In the run visible on the CD and on the screenshots there were 8 collisions.

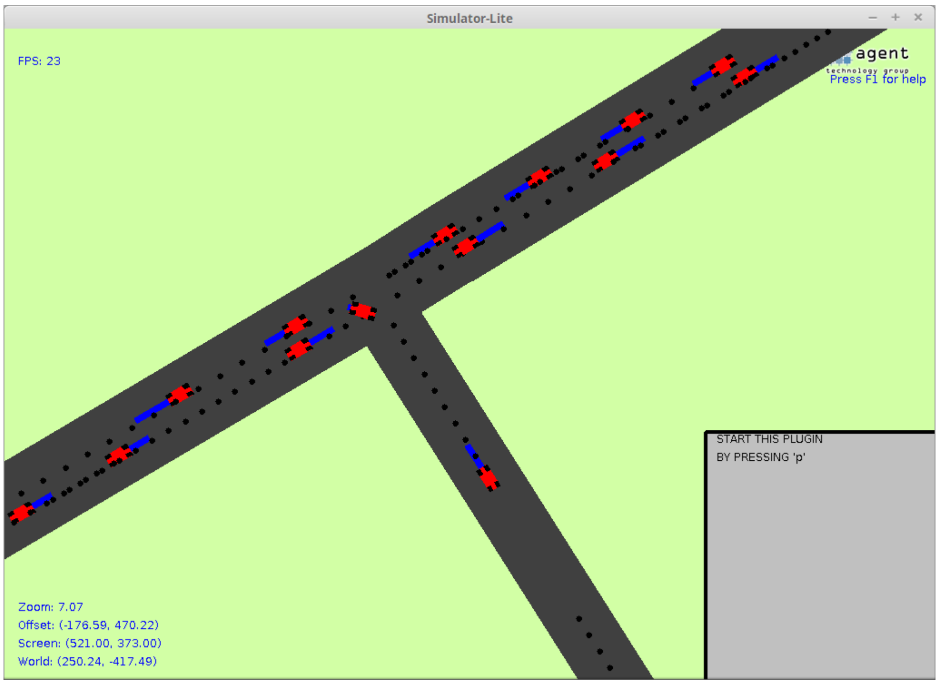


Figure 7.22: Screenshot from the simulation of the city map.

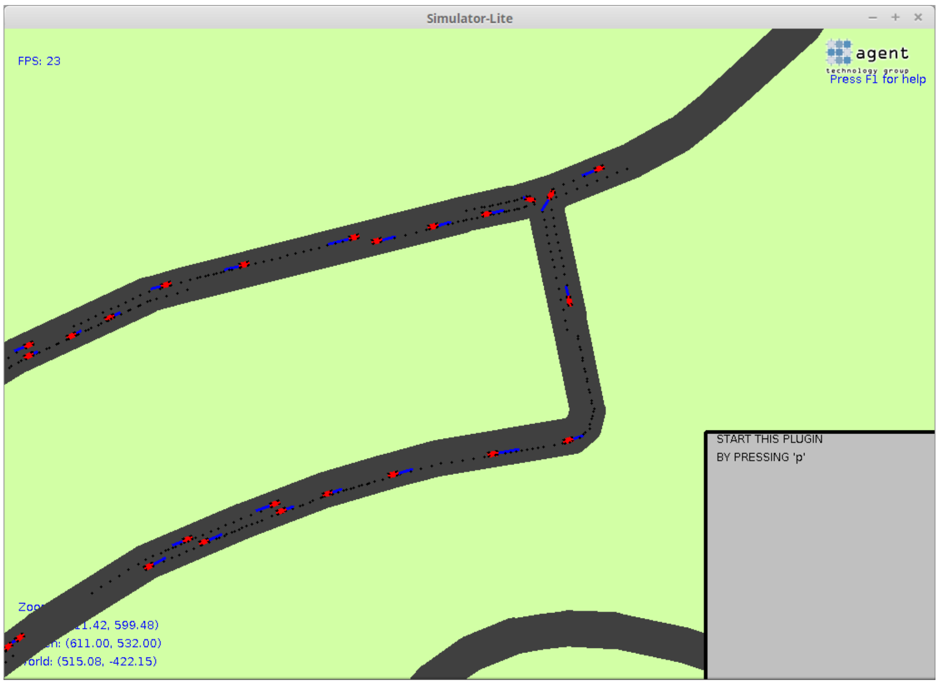


Figure 7.23: Screenshot from the simulation of the city map

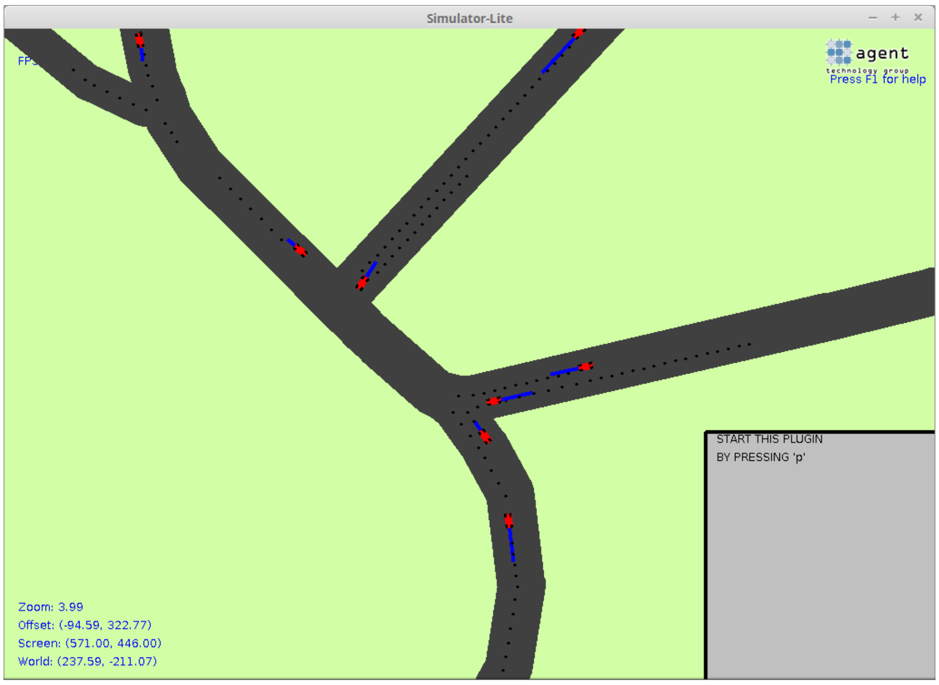


Figure 7.24: Screenshot from the simulation of the city map

Chapter 8

Conclusion

Our proposed method proved its ability to control vehicles autonomously with zero collisions. This method is non-cooperative so it does not require negotiations between vehicles which can be costly in terms of time. This method also proved itself as lightweight in terms of CPU power. This allows calculations to be made at realtime. However our method requires exact data about vehicles nearby that are not easy to obtain. Also the exact map with exact junction shapes is required for this method.

Advantages of this method are:

1. It is a reactive method, so the calculations are fast. For all scenarios in Chapter 7 it never exceeded 50 milliseconds for calculating plans for 70 vehicles.
2. Using existing system – the ACC as a base.
3. Easy implementation.
4. Simple solution.
5. Number of collisions in all scenarios presented in Chapter 7 is 0 except the city map (caused by map data).

Disadvantages of this method are:

1. It allows only one vehicle in the junction when the others vehicles routes are unknown.
2. Ignores traffic rules. It can be improved as it is proposed in Section 8.1.

This method is based on the Safe-distance method proposed in [2], it extends it allowing the original Safe-distance method to be able to navigate through junctions. Original Safe-distance method was only designed for highways. The main problem is to get vehicles location data, this method requires location sensor in every vehicle providing its location to work property. This method can also work without it, but with limited abilities.

Contribution Our contribution was proposing the Generalised Safe-distance method and implementing the GSD Agent into the AgentDrive. Original Safe-distance method has been studied and modified for the new data structure of the road network. Implementing the Road Agent was needed for the GSD Agent. Also some work on the Highway module had to be accomplished in order to be able to simulate the proposed method. The experiments has been made in order to prove our proposed method functionality. This method is based on the Safe-distance method proposed in [2]. The base principle used in this thesis is described in [10].

8.1 Future Work

There are several discretions for future improvements:

- Allowing the road network with traffic rules. It would require to prioritize the vehicles on the main road. This would allow vehicles from the side road to merge when there is an available space on the main road.
- Implement dealing with obstacles on the road. Implement obstacles from the real word (e.g. pedestrian crossing the road, a bicycle, work on the road) into the simulation and teach agents to react on them. This would require more realistic input data from sensors.
- Implement algorithm for overtaking slower vehicles using the opposite lane on the road.
- Implement heterogeneous environment where vehicles are controlled by different agents. This would allow to simulate more accurately the interaction between autonomous and non-autonomous vehicles.

Bibliography

[1] A. Komenda, J. Vokřínek, M. Čáp, and M. Pěchouček, "Developing multiagent algorithms for tactical missions using simulation," *IEEE intelligent systems*, vol. 28, no. 1, pp. 42–49, 2013.

[2] M. Schaefer, "Noncooperative collision avoidance of road vehicles," bachelor thesis, České vysoké učení technické v Praze. Vypočetní a informační centrum., 2011.

[3] European commission, "Road safety," http://ec.europa.eu/transport/road˙safety/specialist/statistics/index˙en.htm, 2011, [Online; accessed 19-April-2015].

[4] J. Ferber, *Multi-agent systems: an introduction to distributed artificial intelligence.* Addison-Wesley Reading, 1999, vol. 1.

[5] M. de Weerdt and B. Clement, "Introduction to planning in multiagent systems," *Multiagent and Grid Systems*, vol. 5, no. 4, pp. 345–355, 2009.

[6] M. Čáp, P. Novák, A. Kleiner, and M. Selecký, "Prioritized planning algorithms for trajectory coordination of multiple mobile robots," *CoRR*, vol. abs/1409.2399, 2014. [Online]. Available: http://arxiv.org/abs/1409.2399

[7] M. Vavřinec and M. Schaefer, "Prioritized planning for road vehicles coordination," in *Proceedings of 19th International Student Conference on Electrical Engineering POSTER 2015.*

[8] R. Kumar and R. Pathak, "Adaptive cruise control–towards a safer driving experience," *Int. J. Sci. Eng. Res*, vol. 3, no. 8, 2012.

[9] J. L. Bentley, "Multidimensional binary search trees used for associative searching," *Commun. ACM*, vol. 18, no. 9, pp. 509–517, Sep. 1975. [Online]. Available: http://doi.acm.org/10.1145/361002.361007

[10] D. Kubeša, "Autonomous cruise control for general collision avoidance of road vehicles," in *Proceedings of 19th International Student Conference on Electrical Engineering POSTER 2015.*

Chapter 9

Contents of attached CD

CD

- README.TXT .... File with contents of attached CD
- source .... folder with the source code. The GSD Agent is located in Highway/src/main/java/cz/agents/highway/agent
- text .... folder with the text of the bachaelor thesis
- video .... folder with proof of concept videos
  - citymap.mp4 .... demostration on the real road network
  - funnel.mp4 .... demostration of the funnel on the Highway
  - Tjunction.mp4 .... demostration of the T junction
  - Xjunction.mp4 .... demostration of the X junction