

CZECH TECHNICAL UNIVERSITY IN PRAGUE
FACULTY OF ELECTRICAL ENGINEERING



BACHELOR THESIS

Control of Road Vehicles Using Machine
Learning Methods

Prague, 2011

Aleš Franěk

Prohlášení

Prohlašuji, že jsem svou bakalářskou práci vypracoval samostatně a použil jsem pouze podklady (literaturu, projekty, SW atd.) uvedené v příloženém seznamu.

V Praze dne 27.5.2011


.....
Podpis

Acknowledgements

First and foremost I thank to RNDr. Lukáš Chrpa, Ph.D. for leading me during this project. Furthermore I want to thank Ing. Jiří Vokřínek and Ing. Antonín Komenda for valuable advices at the beginning of the project as well as Pavel Janovský and Martin Schaefer for constructive cooperation not only during this project. I am also very thankful to my family for their support.

Abstrakt

Tato bakalářská práce je součástí projektu Highway2. Dává si za cíl nasimulovat vozidla, které jsou vybavena umělou inteligencí a dokáží postupně zdokonalovat svoje jízdní schopnosti. Za tímto účelem byla použita metoda posilovaného učení zvaná Q-learning. K vytvoření prostředí dálnice a autonomních vozidel byla použita multiagentní simulace.

Po úvodu o autonomních vozidlech a multiagentních systémech je popsána struktura simulace a následně pak i vlastní učící algoritmus. V závěru práce jsou popsány metody použité ostatními členy projektu Highway2 a všechny navržené algoritmy jsou následně porovnány v sérii testů.

Abstract

This thesis is a part of the Highway2 project. It aims to create cars with artificial intelligence which spontaneously develop their skill of driving. For this purpose is used method called Q-learning which is a sort of reinforcement learning. It uses a multi-agent system to simulate the highway and the autonomous vehicles.

After introduction about driverless cars and multi-agent systems is described actual structure of the simulation and the machine learning algorithm. At the end of the thesis are described methods which were used by other members of the Highway2 project and all the algorithms are compared in several tests.

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Student: Aleš Franěk
Studijní program: Elektrotechnika a informatika (bakalářský), strukturovaný
Obor: Kybernetika a měření
Název tématu: Řízení silničních vozidel pomocí metod strojového učení

Pokyny pro vypracování:

1. Nastudujte tematiku řízení vozidel pomocí strojového učení.
2. Seznamte se s agentní platformou a simulátorem.
3. Navrhněte metody pro řízení agentů vozidel pomocí strojového učení.
4. Implementujte a validujte navržené metody pomocí simulace.

Seznam odborné literatury:


- [1] Vokřínek, J.; Jiránek, J.; Hodík, J.: Agent-based Highway Traffic Simulation, Praha.
- [2] Sutton, R.S.; Barto, A.G.: Reinforcement Learning: An Introduction, Cambridge, MA, USA, 1988.
- [3] Watkins, Ch.J.C.H.; Dayan, P.: Q-learning, Boston, USA, 1992.
- [4] Alpaydin, E.: Introduction to machine learning, Cambridge, MA, USA, 2004.

Vedoucí bakalářské práce: Mgr. Lukáš Chrpa, Ph.D.

Platnost zadání: do konce zimního semestru 2011/2012


prof. Ing. Vladimír Mařík, DrSc.
vedoucí katedry




prof. Ing. Boris Šimák, CSc.
děkan

V Praze dne 2. 2. 2011

BACHELOR PROJECT ASSIGNMENT

Student: Aleš Franěk

Study programme: Electrical Engineering and Information Technology

Specialisation: Cybernetics and Measurement

Title of Bachelor Project: Control of Road Vehicles Using Machine Learning Methods

Guidelines:


1. Work up knowledge of control of vehicles using machine learning methods.
2. Get to know the agent platform and simulator.
3. Design methods for control the agent vehicles using machine learning techniques.
4. Implement and validate design methods by simulation.

Bibliography/Sources:

- [1] Vokřínek, J.; Jiránek, J.; Hodík, J.: Agent-based Highway Traffic Simulation, Praha.
- [2] Sutton, R.S.; Barto, A.G.: Reinforcement Learning: An Introduction, Cambridge, MA, USA, 1988.
- [3] Watkins, Ch.J.C.H.; Dayan, P.: Q-learning, Boston, USA, 1992.
- [4] Alpaydin, E.: Introduction to machine learning, Cambridge, MA, USA, 2004.

Bachelor Project Supervisor: Mgr. Lukáš Chrpa, Ph.D.

Valid until: the end of the winter semester of academic year 2011/2012


prof. Ing. Vladimír Mařík, DrSc.
Head of Department




prof. Ing. Boris Šimák, CSc.
Dean

Prague, February 2, 2011

Content

LIST OF FIGURES	IX
LIST OF TABLES	XI
1 INTRODUCTION	1
2 MULTI-AGENT SYSTEMS	3
3 HIGHWAY SIMULATION	5
3.1 Alite.....	5
3.2 Event-based simulation	5
3.3 Entities.....	6
3.4 Environment.....	7
3.4.1 Storages	7
3.4.2 Actuators.....	8
3.4.3 Sensors.....	9
4 MACHINE LEARNING	11
4.1 Supervised learning	11
4.2 Reinforcement learning	11
4.2.1 Adaptive dynamic learning	12
4.2.2 Temporal difference learning.....	13
4.2.3 Active learning agent	13
4.2.4 Q-learning.....	14
5 SIMULATION	15
5.1 Machine learning drive.....	15
5.1.1 State space	15
5.1.2 Optimization	17
5.2 Noncooperative drive.....	20
5.3 Cooperative drive	21

6	EXPERIMENTS	23
6.1	Results of tests	25
7	CONCLUSION	27
	REFERENCES	29
	APPENDIX A	31
	Test results	31
	APPENDIX B	35
	Source code	35

List of figures

FIGURE 3.1: DIAGRAM OF THE SIMULATION BLOCKS [5]	6
FIGURE 3.2: DIAGRAM OF ACTUATOR AND SENSOR LAYERS [6]	8
FIGURE 5.1: VISIBILITY ZONES OF THE AGENT	16
FIGURE 5.2: HISTOGRAM OF ITERATIONS OF THE STATES	18
FIGURE 5.3: PROGRESS OF UTILITY VALUES	19
FIGURE 5.4: PROGRESS OF UTILITY VALUES WHEN SOME OF THE ACTIONS WERE INITIALY RESTRICTED	19
FIGURE 5.5 CONFLICTING SITUATION	20
FIGURE 5.6 NONCOOPERATIVE PLANNING DIAGRAM [6]	21
FIGURE 5.7 COOPERATIVE PLANNING DIAGRAM [5]	21
FIGURE 5.8: CHANGED LANES IN TEST IV	24
FIGURE 5.9: AVERAGE VELOCITY IN TEST IV	24

List of tables

TABLE 1: TABLE OF ENTITIES	23
TABLE 2: RESULTS OF TEST I	31
TABLE 3: RESULTS OF TEST II	32
TABLE 4: RESULTS OF TEST III	32
TABLE 5: AVERAGE VELOCITY IN TEST IV	33
TABLE 6: NUMBER OF CHANGINGS OF LANE.....	33

1 Introduction

The technology is with the development more and more becoming an inherent part of people's everyday lives and the transportation is no exception. The transport industry experienced an extensive progress in the last 50 years however the way how we drive our automobiles did not change at all.

There is a set of driver assistance systems embedded in the most of the modern vehicles including ABS¹, EBD² or ESC³. Several cars are equipped with more advanced assistance systems like adaptive cruise control, lane departure warning system or intelligent parking assist, nevertheless there is no autonomous driving system in common use so far.

Not only that with these autonomous systems there is no need of human assistance during the drive but moreover they come with many other possible benefits. The most significant one is the improvement of safety on the road. An automated system can rapidly decrease a reaction time by about two seconds when car is driven by a human. [1] Another benefit would be efficient utilization of highway space. Due to the incomparably shorter decision-making time can be the distance between cars reduced and achieve up to five times better utilization. [1]

First attempts to construct a driverless car date to the year 1977. [2] In the year 2004 Darpa⁴ founded a competition for driverless cars called The Darpa Grand Challenge where vehicles went across the desert and after that they went through

¹ ABS (from german Antilockiersystem) is a system that prevents car's wheels from blocking and thus from losing the adhesion during the breaking.

² EBD (Electronic Brakeforce Distribution) is a system that intelligently distributes the breaking power among the wheels.

³ ESC (Electronic Stability Control) is an extension of ABS for minimizing skids and improving control of the vehicle.

⁴ DARPA (Defense Advanced Research Projects Agency), more information on URL <http://www.darpa.mil/>.

an urban environment. [3] The last milestone achieved VisLab⁵ in the year 2010 when their two autonomous electric vans successfully went 13 000 km from Italy to Shanghai, China. [4]

This thesis is a part of the project Highway2 at the Department of Cybernetics, Faculty of Electrical Engineering at the Czech Technical University in Prague. Along with cooperative [5] and noncooperative [6] control it aims to observe advantages and disadvantages of these different approaches for a possible future use.

The simulation uses multi-agent system which is presented in Chapter 2. The structure of the simulation is described in Chapter 3. In Chapter 4 and Chapter 5 is defined used machine learning algorithm along with cooperative and noncooperative control. All methods are then compared in Chapter 6.

⁵ More information on URL <http://vislab.it/>.

2 Multi-agent systems

In the last years agent technology is becoming a widely used segment of computer science. An agent is the software representation of an object from the real world with its own behavior and a certain degree of autonomy. The multi-agent system is then a system with multiple agents set into particular environment.

Multi-agent systems are used everywhere where a general description of the problem is too complicated or even impossible to describe sufficiently. In these cases is the problem description distributed into simpler function units, agents. Each multi-agent system has an environment which is the world where agents operate.

There are three basic types of intelligent agents according to their behavior. [7] The first, the simpler one, is the reactive agent. A reactive agent reacts on the changes of the environment's state by completing an action to achieve its goals.

The second type and more sophisticated type of intelligent agents is the deliberative agent. A deliberative agent also reacts on changes in the environment to achieve its goals however it understands their purpose and is able to estimate the consequences of the actions. The behavior of the deliberative agent is similar to the thinking entities in the real life.

Another agent's feature is the social communication. A social agent can communicate with the other agents and thus can get information about their plans and goals.

A hybrid agent is then a combination of any of the agents above.

For their simplicity of design and their complexity of description of the real life problems are multi-agent systems used in a wide spectrum of branches, for instance economics, logistics, biology, simulation of physical and social phenomena and also transportation.

3 Highway simulation

The highway simulation was designed and implemented in collaboration with students Pavel Janovský, Martin Schaefer and Karel Jalovec led by Ing. Jiří Vokřínek and Ing. Antonín Komenda.

3.1 Alite

„Alite is a software toolkit helping with particular implementation steps during construction of multi-agent simulations and multi-agent systems in general.“ [8]

The simulation uses the Alite software toolkit developed at the Department of Cybernetics, Faculty of Electrical Engineering at the Czech Technical University in Prague. It provides a set of basic software tools for constructing multi-agent event-based simulations however it does not include any advanced pre-designed framework. [8]

API divides simulation universe into the several units by their function, described bellow.

3.2 Event-based simulation

It is a discrete-time simulation where the behavior of the world is represented by a series of events. Each event has its own timestamp of its execution. These events are aligned in a queue according to their timestamp and executed by the event processor one after another. When an event is called it is send to its handler where a command defined by event's type is executed and the time of the simulation is set at event's timestamp. After that the loop continues with the next event in the queue until the event for stopping the simulation is called. [8]

Time jumps after various steps in this type of simulation and the state of the world changes only when an event is executed. Event-based simulations are used in cases when continuous description of the world would be very difficult or more likely unnecessary.

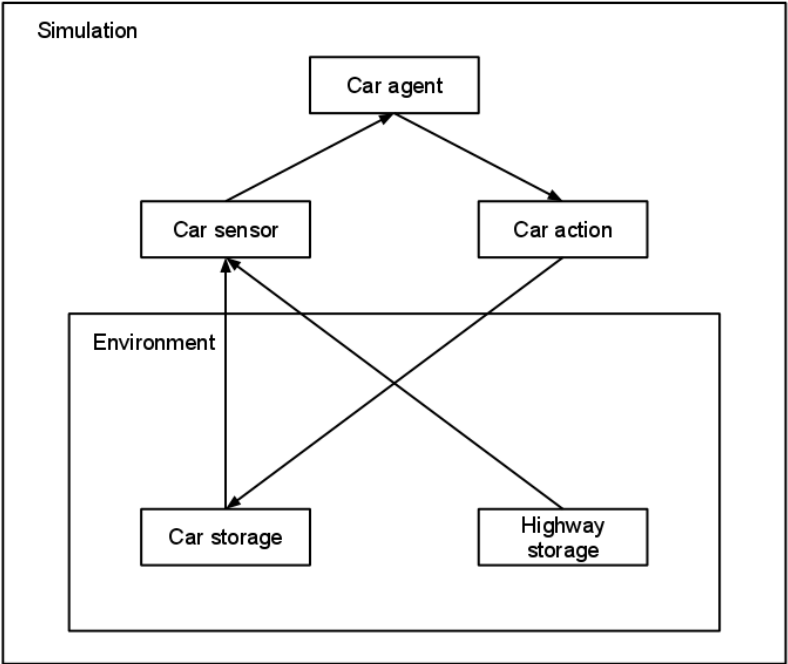


Figure 3.1: Diagram of the simulation blocks [5]

3.3 Entities

Entity is a general title for all objects in the simulation. It can be both agents and embodiments in the simulation or parts of the other entities. [8] Entities which represent physical objects are called simulation entities, thus agents are not simulation entities.

In the highway simulation is each car represented by the agent and by one simulation entity. An agent is the thinking of the car which creates a plan of the drive. A simulation entity contains all features of the particular vehicle such as length, width, possible acceleration and deceleration, maximal velocity as well as the state of the vehicle like current lane, position within the highway segment, velocity or information if the car is crashed.

3.4 Environment

Environment consists of three different function blocks, storages, actuators and sensors.

3.4.1 Storages

Storages hold all the information about entities in the environment. Elements are grouped according to their element type. [8] There are two storages in this simulation, highway storage and car storage.

The highway storage holds information about the highway. The highway consists of the highway segments. Each segment has information about the next and previous segment, the number of its lanes and the absolute position of the segment in the world. The highway can be created either straight or it can be defined by the Bézier curve.⁶

In the car storage is data about simulation entities. Their features have been already described in Chapter 3.3. The car storage also contains a loop which periodically fires events that update position of cars. This period was set on 100 miliseconds. Important fact is that the car storage does not change the states of the cars itself and only sends events to agents' handlers. This makes the simulation decentralized.

⁶ Bézier curve is a parametric curve. More details about Bézier curve can be find on URL <http://mathworld.wolfram.com/BezierCurve.html>.

3.4.2 Actuators

Entities cannot directly communicate with each other nor change any data in the storages. For this purpose are in the simulation actuators (see Figure 3.1). Actuators along with sensors can be organized in layers to increase the modularity of whole simulation. There are three different layers ordered according to their degree of complexity (see Figure 3.2).

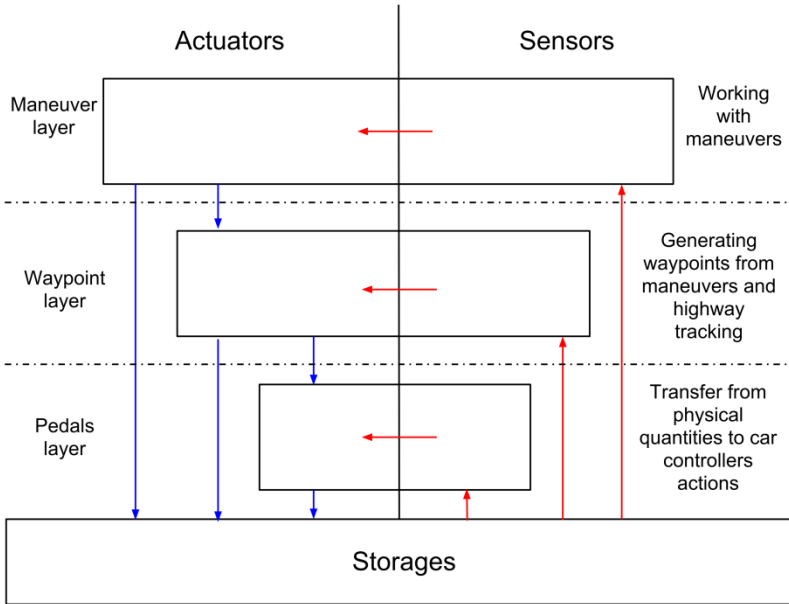


Figure 3.2: Diagram of actuator and sensor layers [6]

Maneuver layer

The agent operates only with the highest layer, maneuver layer. Agent's actions were reduced to a set of maneuvers to simplify agent's planning process. With maneuver actuator is car agent able to perform one of five defined maneuvers. They are maneuvers

for straight drive, acceleration, deceleration and changing the lane to the left and to the right.

All the maneuvers have their predefined time. Their length depends on the current speed of the car. Each agent can perform one maneuver at the time so it can either change its speed or change its lane or do not change anything (by performing a straight maneuver).

Waypoint layer

The waypoint layer is there in order to maintain agent's decisions of choosing maneuver independent on the highway profile. The waypoint layer transforms a maneuver into a set of waypoints. These waypoints are situated along the maneuver trajectory whereas maneuvers for changing lanes follow Bézier curves which represent movement of the vehicle better than straight abscissa. Each waypoint has information about its position on the highway and expected velocity of the vehicle at that point which can be used as a value for controllers of the car.

Pedals layer

The last layer, pedals layer, transforms desired direction of the car and velocity into the values for the accelerator, the brake and the steering wheel. This layer is supposed to adjust the values for different car types with different handling characteristics.

3.4.3 Sensors

The sensors are organized into the three layers like the actuators. Again, the agent uses only the sensor from the highest layer, however sensors have a different structure than actuators. Unlike actuators each sensor extends functionality of the lower one.

The pedals sensor provides information about position of the car on the highway, the waypoint sensor adds information

Chapter 3: Highway simulation

about state of the car and the maneuver sensor adds information about highway situation and the surrounding other cars.

4 Machine learning

Machine learning is a part of artificial intelligence important for the multi-agent systems which can be used on many different areas of our lives. Machine learning extends the functionality of the agent in the world and enables to improve its behavior by several various methods.

4.1 Supervised learning

Supervised learning is part of machine learning where the knowledge is obtained from already known examples with a given right output. The goal of supervised learning is to generalize properties of the objects from a training set. After the agent is taught knowledge is applied on a testing set of inputs to verify the outcome.

Supervised learning is dealing with issues when a set of training data is too small or when demanded function is too complex. Since the situation on the highway is an extremely complex function with a high sensitivity to the noise on the outcome, which could have fatal consequences, supervised learning was not used in this project. Also, we probably would not be able to provide a sufficient training set of data.

4.2 Reinforcement learning

Previous supervised learning needed a set of labeled examples in order to generalize the utility function. However as a matter of fact we do not dispose of such set in most of the cases. Reinforcement learning takes a different approach. It uses feedback function to detect if the actions of the agent led to the right outcome or not. This feedback is called reward (negative reward is sometimes called punishment).

The reward can be given not only in terminal state but in any state and the reward can be estimated at any value by various factors. The agent can be an active part of the environment or it can be a passive learner and just observe the world. The agent can also start learning with already some knowledge of utility of some actions. [9]

Basic model of reinforcement learning agent consists of [7]:

- Finite set of possible states S
- Set of actions A
- Reward function $R: S \times A \rightarrow \mathbb{R}$ (agent gets the reward $r \in \mathbb{R}$ for executing the action $a \in A$ in the state $s \in S$)
- Probability function $T: S \times A \rightarrow \Pi(S)$, where $\Pi(S)$ is a distribution of probabilities
- Strategy $\pi: S \rightarrow A$, which assigns an action in each state

4.2.1 Adaptive dynamic learning

Adaptive dynamic learning uses known probability function to estimate the utility value in the state by known reward in this state and weighted utility values of all states an agent can reach. Utilities are computed by following equation

$$U(s) = R(s) + \sum_{s'} M(s, s') U(s'), \quad (4.1)$$

where $R(s)$ is the reward in the state s , $M(s, s')$ is probability that the agent will get from the state s to the state s' by one action and $U(s)$ and $U(s')$ are utilities of particular states. [9]

In this case is the agent passive and it does not include any information about actions.

4.2.2 Temporal difference learning

Temporal difference learning does not use the probability function instead of it observes the transitions between states. It reinforce influence of direct neighbour states by using equation

$$U(s) = U(s) + \alpha(R(s) + U(s) - U(s')), \quad (4.2)$$

where parameter α is called the learning rate. [9] Even if the temporal difference learning and adaptive dynamic learning use different approach the equilibrium that they will eventually find will be the same. [9]

Since the temporal difference learning does not use the probability function, it can be used even in the unknown environment.

4.2.3 Active learning agent

Active learning agent is a part of the environment so except utilities of the states must it observe the effect of the actions and the utilities in possible following states. The extended equation (4.1) will be

$$U(s) = R(s) + \max_a(\sum_{s'} M(s, a, s')U(s')), \quad (4.3)$$

where $M(s, a, s')$ is probability that the agent will get from state s to state s' by taking the action a . [9] Next difference from previous algorithms is that the agent can choose which action is going to take. That is why only the action with the highest utility value affects the change of utility in state s . All other actions with worse utility will probably get the agent to the worse state, thus they are irrelevant.

4.2.4 Q-learning

Q-learning uses action-value function where utility values are assigned to actions for each state. These values are called Q-values. There is a direct connection between Q-values and utility values:

$$U(s) = \max_a Q(s, a), \quad (4.4)$$

where $Q(s, a)$ is the Q-value for action a going from state s . (8)

Extended equation for adaptive dynamic learning is

$$Q(s, a) = R(s) + \max_{a'} (\sum_{s'} M(s, a, s') Q(s', a')). \quad (4.5)$$

However this function still requires a table of probabilities. Against it the extended equation for temporal difference learning will be

$$Q(s, a) = Q(s, a) + \alpha(R(s) + \gamma \max_{a'} Q(s', a') - Q(s, a)), \quad (4.6)$$

where γ is the discount factor, does not requires any probability at all. [10] The discount factor γ weights the effect of the next state's utility and also limits possible utility of a state in interval

$$\left(\frac{\min_s R(s)}{1-\gamma}, \frac{\max_s R(s)}{1-\gamma} \right). \quad (4.7)$$

5 Simulation

5.1 Machine learning drive

Since there was no initial description of the behavior of the environment or any known probability function, the action-value approach was used in this project. Major priority was stressed on safety, thus a collision of vehicles was evaluated with the highest negative reward. On the other hand the goal is to go as fast as possible and at the same time not to exceed the speed limit so a positive reward was given every time the car is moving and with the linear dependence on its velocity.

5.1.1 State space

A proper identification of the state and its parameters appears to be crucial for finding desired results. When a state has too little parameters learning cannot include all the effects in the environment and the utility values will not converge to their equilibrium. Vice versa if a state is defined with too many parameters time of learning process grows exponentially and learning will most likely lose its ability to generalize. There was used a state space with 19 discrete parameters in this simulation, that means there are 2^{19} possible states.

As seen on Figure 5.1 a car has around it several zones. The first one, safety zone, detects the closest vehicles around the car. There are six different mutual positions of the cars (on the left, in the same lane or on the right and in front of or behind the car). Each position is one parameter of the state space.

Analogically works detection of surrounding vehicles in the visibility zone. Cars which have been already detected in the safety zone are cast out from the set of cars in the visibility

zone thus only cars from annulus affect the next parameters of the state space.

Previous parameters tell information about mutual position of the vehicles but they do not say anything about velocities which are for making a right decision equally important. In general it does not matter if there is a slower car ahead in the next lane or a faster car behind in the very same lane. Either way changing the lane will be equally risky. Because of that these two cases could be reduced into one there are four parameters to detect possible risks, one for each side of the car.

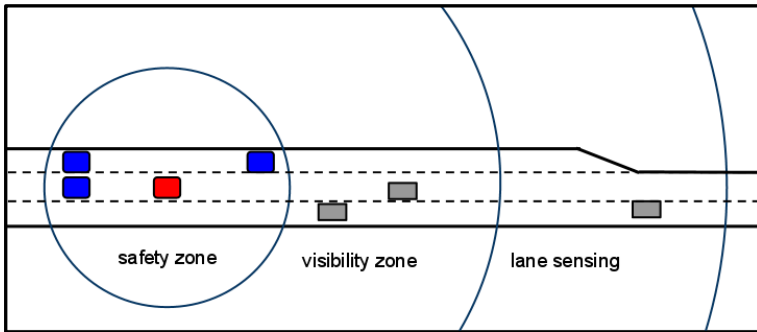


Figure 5.1: Visibility zones of the agent

In the real world and also in the simulation does not the highway always consist of the same number of lanes. When current lane of the car is going to end car must react to it as well as the other cars must react to narrowing of the road. Therefore the state space must contain information about lanes ahead as well.

The velocity of the car is also a significant parameter. The agent must know its velocity to decide if it should accelerate or it has already a satisfactory speed. Another case when knowledge of the velocity is crucial is when a car is not moving.

A deceleration maneuver or any other maneuver except acceleration will not have any effect on changing state of the car. By a binary combination of two values from the state space can be achieved detection of four different levels of the velocity from zero to the maximal allowed speed of the road.

The last parameter indicates if the car is crashed. Crashed car does not contribute to evaluating of the states however it is essential to keep this parameter. All the states where car is already crashed typically get a minimal possible utility value (see Equation 4.7) for all actions. Thus when a car gets crashed the last action before the accident instantly gets a negative evaluation (see Equation 4.6).

5.1.2 Optimization

The first attempt of the implementation held the algorithm by the definition and begun with blank Q-values table and the radiuses of the zones were estimated on reasonable absolute values. When a state reached enough iteration the action with the lowest utility value was restricted since it was evaluated as the least beneficial. However the results were not satisfying at all and there was still an inconsiderable number of collisions.

After a series of changes showed significant improvements. First of all it is advantageous to limit the state space and restrict all the downright wrong actions. In this case was restricted to perform maneuver of changing lanes when a car was in the last lane and there was no lane to go in the particular direction.

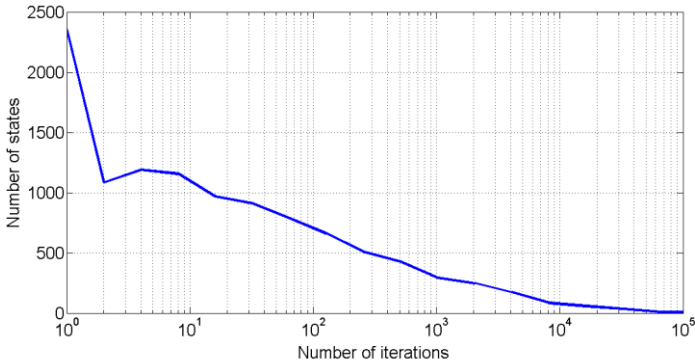


Figure 5.2: Histogram of iterations of the states

On Figure 5.3 and Figure 5.4 is shown the progress of the utility values with and without the initial restriction. Both are for the same, most elementary state; one car on a straight road with one lane. Notice that the results may appear almost the same however learning without initial restrictions took approximately 150 times more simulation time. This is happening because agents are often unnecessarily making wrong decisions and they crash before they can visit any other states.

To visit all states in a sufficient degree appears to be essential for Q-learning function. As seen on Figure 5.2 most of the states do not even reach enough iteration for converge their utility-action values. The sufficient number of iteration was set by the tests on at least hundreds or more likely thousands of visits. Even though the most of the states does not satisfy this condition agents are in this phase very well learned because these states are very unlikely to happen. In time when only 11% of the states reached the desired number of iterations made these states 97% of all states ever visited.

It also appears to be helpful when agents learn at the beginning from elementary simple states where right actions are obvious and get proper evaluation in them. After that when

agent visits not so frequent more complicated state it immediately gets an undistorted utility value of the next state. This approach significantly accelerates the time of learning.

Another enhancement was to change the radiuses of the visibility zones from static to dynamic with a dependence on the velocity. It is obvious that a car which is stuck in the traffic jam affects cars in a different range than a car which is going the maximal speed.

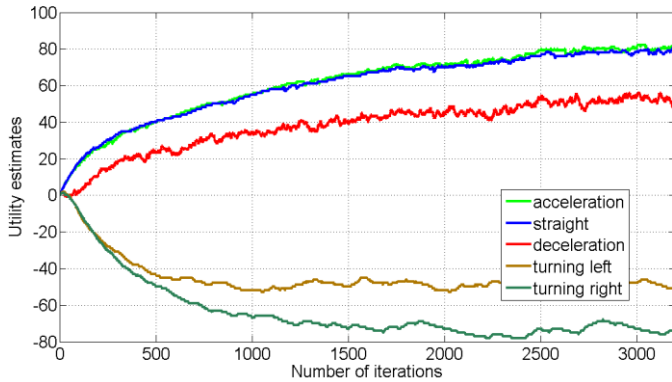


Figure 5.3: Progress of utility values

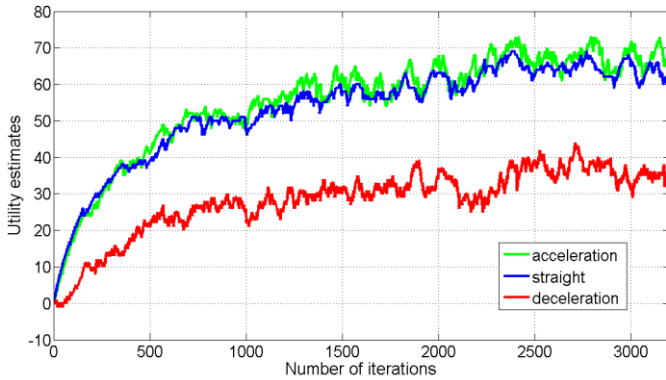


Figure 5.4: Progress of utility values when some of the actions were initially restricted

Even if agents were taught well for all of the cases there are still a few situations where they will still fail. One example for all is shown on Figure 5.5. Since the state of the car is affected only by the closest lanes around the car, agents cannot avoid the collision in this particular situation. Because situations like this are very unlikely to happen and also that preventing all these situations would enormously extend the state space were these situations not considered in the simulation.

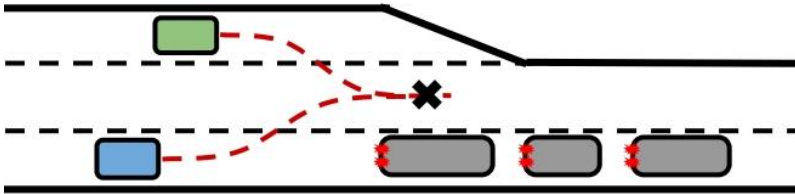


Figure 5.5 Conflicting situation

5.2 Noncooperative drive

The noncooperative agent [6] uses situation on the highway similar to the state from machine learning to estimate behavior of the other cars. Since it does not use discrete values it has much more information about its state, it is expected to have better result in the simulation.

The agent uses to knowledge about environment to predict all possible reasonable maneuvers of the other cars and chooses among the nonconflicting ones to create its own plan (see Figure 5.6).

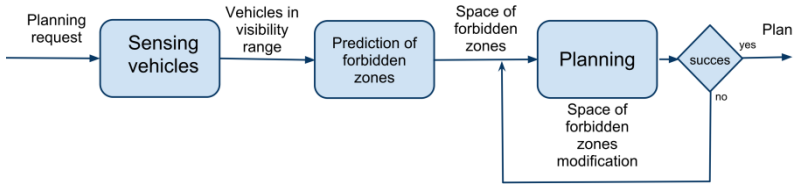


Figure 5.6 Noncooperative planning diagram [6]

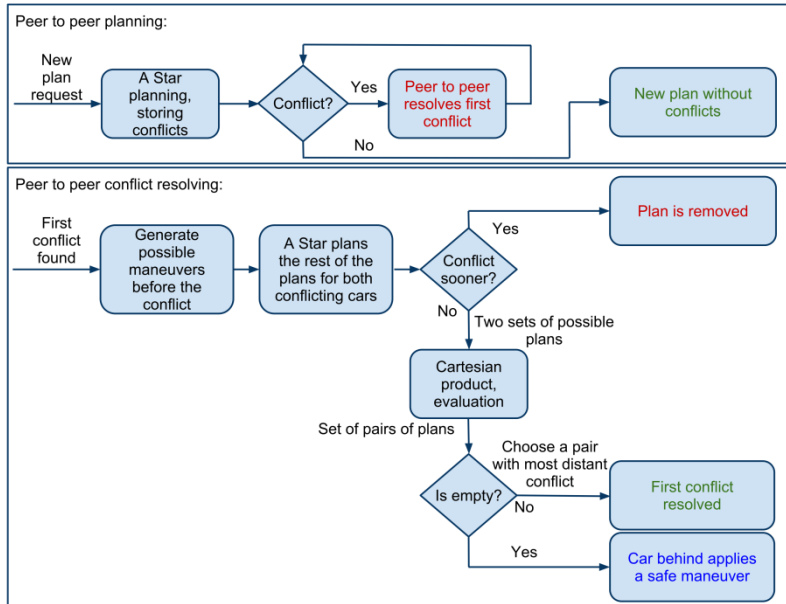


Figure 5.7 Cooperative planning diagram [5]

5.3 Cooperative drive

In cooperative simulation [5] an agent communicates with all the other agents of the cars on the highway. Each agent creates its own plan without consideration for demands of the other cars at the beginning. Then all plans are compared and expected collisions are found. Each collision is solved by recreating new plans

one after another (see Figure 5.7). For collision avoidance and estimating new plans uses a cooperative agent A* algorithm.⁷

Cooperative agents know about all plans of the other cars so they can estimate exact state of whole environment in any moment in the future. Because of that are cooperative agents able to make plans in order of tens of maneuvers unlike machine learning and noncooperative agents which plan one maneuver ahead.

⁷ A* is an algorithm used to find an optimal path in graphs using the heuristic function.

6 Experiments

Several different scenarios were designed to show characteristics of implemented algorithms. In all tests was used the same set of vehicles (see Table 1). Each result is an average of results got from several independent runs. Each of the runs lasted 10 minutes in the simulation time.

Entity	Maximal speed [km/h]	Length [m]	Width [m]	Percentage [%]
Car	130	4,0	2,0	62,5
Van	100	6,0	2,3	12,5
Bus	90	10,0	2,5	12,5
TIR	80	13,0	2,5	12,5

Table 1: Table of entities

Test I

Scenario: Entities are created at the beginning of the highway in the interval of 6 seconds. Length of the highway is 1500 meters and it has two lanes.

Test II

Scenario: Entities are created at the beginning of the highway in the interval of 6 seconds. Length of the highway is 1500 meters. After 1000 meters the highway narrows from three lanes to two.

Test III

Scenario: Entities are created at the beginning of the highway in the interval of 6 seconds. Length of the highway is 1500 meters. After 1000 meters the highway narrows from two lanes to one.

Test IV

Scenario: The highway in the test IV has the same shape as the highway in the test II and only the interval of creating entities varies. This aims to test behavior of agents in a different density of traffic.

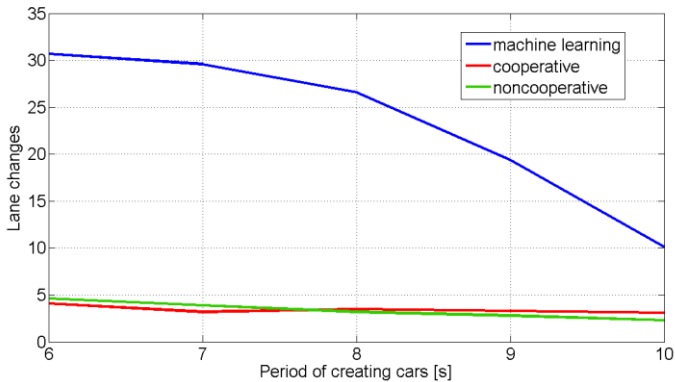


Figure 6.1: Changed lanes in test IV

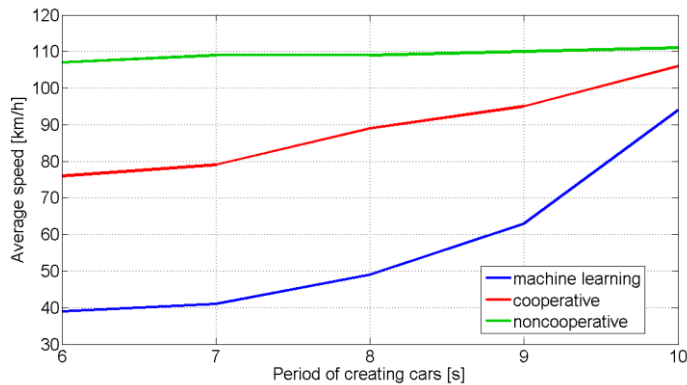


Figure 6.2: Average velocity in test IV

6.1 Results of tests

The tables of detailed results from each test are in Appendix A. We can dedicate some qualities of each algorithm from the results.

The machine learning agent achieve in general the worst results, it has the highest collision rate of 0,53 collisions per test and the lowest average speed among all agents. It is understandable because the agent operates with much smaller state space than the other two. Notice that average state which caused the collision was 80,6 times visited. This is negligible small number since the most frequent states have tens of thousands iterations (see Figure 5.2). States need at least several hundreds of visits so they can estimate utility values for each action. Before that the agent randomly picks from possible actions.

On the other hand the noncooperative drive reached the best results. It has the highest average speed and did not cause any accident. This is surprising because it was expected that cooperative drive will be better since it operates with more information. When we look behind the numbers we find out that noncooperative simulation deals with an issue. In the noncooperative simulation agent maximizes its velocity as long as it save for it and it does not interact with the other agents. This has an impact when the highway narrows and stopped cars stay in the ending lane. Passing cars in the next lane are going so fast that the agent cannot safely change lane and is stucked. So the noncooperative algorithm is effective in most of the cases however it cannot deal with narrowing of the highway very well.

In Figure 6.1 and Figure 6.2 you can see graphs of the results from the test IV. The first one shows how many times a car changed its lane. There is a significant difference between machine learning and the other two algorithms. This phenomenon happens because both cooperative and noncooperative algorithms use preferred maneuvers so agent rather perform straight maneuver than

changing the lane. Machine learning agent chooses actions randomly and does not prefer any maneuver at the beginning of learning.

The second figure shows the average velocity depending on traffic density. The results demonstrate that the machine learning agent converge to the results of the other two algorithms when the traffic on the highway is small. With the increasing traffic density loses the machine learning agent ability of solving the conflicts. This happens because there are exponentially more possibilities of potential states with the higher density. That means that there is a higher chance that the agent will get into the state with very few iterations.

One of the factors why the noncooperative and the cooperative agents reached better results it because they assume that all of the other cars use the same algorithms of driving. The machine learning agent was on the other hand taught with literally all possible kinds of behavior of the other cars so it should be able to drive among the cars with a different control.

7 Conclusion

The autonomous agent vehicles with artificial intelligence were implemented and verified in a series of tests. It appeared which parameters of the state space are crucial for the right functioning and the consecution how to act during the learning process. In general, the results are significantly better when an agent learns from the elementary situations and then with gradually increasing complexity of the problems.

Even though the results of the tests were not as good as the results of other methods they were fairly satisfactory and they met the requirements.

The already implemented structure is ready to be developed in the future work. One of the main issues that I was dealing with was that the agents did not visit all possible states enough times. Since the state space grows exponentially with the number of parameters a more sophisticated state indication is suggested for the future work.

References

1. GOLDSMITH, Theodore. AZINET LLC Publishing. *Advanced Vehicle Control Systems (AVCS) - The Real Automobile*. [Online] December 1998. [Cited: May 24, 2011.] <http://www.azinet.com/articles/real98.htm>.
2. Tech-FAQ by TopBits. *Driverless Car*. [Online] [Cited: May 24, 2011.] <http://www.tech-faq.com/driverless-car.html>.
3. Defense Advanced Research Projects Agency. *Welcome*. [Online] [Cited: May 24, 2011.] <http://archive.darpa.mil/grandchallenge/>.
4. Intercontinental Challenge. [Online] VisLab. [Cited: May 24, 2011.] <http://viac.vislab.it/>.
5. JANOVSKEÝ, Pavel. *Cooperative Collision Avoidance of Road Vehicles*. Department of Control Engineering, Faculty of Electrical Engineering, CTU in Prague. Prague : s.n., 2010. Bachelor Thesis.
6. SCHAEFER, Martin. *Noncooperative Collision Avoidance of Road Vehicles*. Department of Control Engineering, Faculty of Electrical Engineering, CTU in Prague. Prague : s.n., 2010. Bachelor Thesis.
7. KUBÍK, Aleš. *Intelligentní agenty : tvorba aplikačního software na bázi multiagentových systémů*. Brno : Computer Press, 2004. p. 280. ISBN 80-251-0323-4.
8. Alite - Wiki - Redmine. [Online] [Cited: May 24, 2011.] <http://merle.felk.cvut.cz/redmine/projects/alite/wiki>.
9. RUSSELL, Stuart J.; NORVIG, Peter. *Artificial intelligence : a modern approach*. 1st ed. Upper Saddle River : Prentice Hall, 1995. p. 932. ISBN 0-13-103805-2.
10. —. *Artificial intelligence : a modern approach*. 3rd ed. Upper Saddle River : Prentice Hall, 2010. p. 1132. ISBN 978-0-13-207148-2.

Appendix A

Test results

	Noncooperative agent	Cooperative agent	Machine learning agent
Cars finished	92,8	78,6	70,4
Cars crashed	0	0	0,4
Average speed	112,2	84,6	68,18
Straight	66	1209	468,2
Acceleration	1487,8	1208	1225
Deceleration	51,2	250,8	431,8
Turn left	12,2	0,4	75,2
Turn right	16,6	0	71,8
Previous visits of state before crash	-	-	16,5

Table 2: Results of test I

Appendix A

	Noncooperative agent	Cooperative agent	Machine learning agent
Cars finished	86	79,2	45,2
Cars crashed	0	0	1,2
Average speed	108	77	39,98
Straight	268,2	1164,2	533,6
Acceleration	1440,4	1534,4	1084,4
Deceleration	533,2	384,2	1060,6
Turn left	3,8	0	117,2
Turn right	41	40,4	132,2
Previous visits of state before crash	-	-	102

Table 3: Results of test II

	Noncooperative agent	Cooperative agent	Machine learning agent
Cars finished	67,6	48,2	43,4
Cars crashed	0	0,4	0
Average speed	79,6	62	49,46
Straight	1149	657,8	266
Acceleration	1085,6	1265,4	898,8
Deceleration	1372,6	401	403
Turn left	5,4	0,6	83
Turn right	28,2	34,4	104,2
Previous visits of state before crash	-	-	-

Table 4: Results of test III

Appendix A

Average velocity [km/h]			
Time [s]	Noncooperative agent	Cooperative agent	Machine learning agent
6	107	76	39
7	109	79	41
8	109	89	49
9	110	95	63
10	111	106	94

Table 5: Average velocity in test IV

Changed lanes			
Time [s]	Noncooperative agent	Cooperative agent	Machine learning agent
6	4,1	4,6	30,7
7	3,2	3,9	29,6
8	3,5	3,2	26,6
9	3,3	2,8	19,4
10	3,1	2,3	10,1

Table 6: Number of changings of lane

Appendix B

Source code

Source code of the project can be found on the enclosed CD.

Source code and files closely related to this thesis are placed in following folders:

- `cz/agents/highway2/planner/machineLearning`
- `cz/agents/highway2/planner/plan`
- `recourses/tables/machine learning`