**Project Laboratory report**

Department of Telecommunications and Media Informatics

|  |  |
| --- | --- |
| Author: | **Fábián Füleki** |
| Neptun code: | **AEP0TG** |
| Specialization: | **Infocommunications - HIT** |
| E-mail address: | **ffabi1997@gmail.com** |
| Consultants: | **Róbert Moni,  Bálint Gyires-Tóth** |
| Their e-mail addresses: | **robertmoni@tmit.bme.hu, toth.b@tmit.bme.hu** |

**Subject:   
"AI Driver" - End-to-End Driver Activity Prediction**

**Task description**

In this work, I aim to create a self-driving system based on deep reinforcement learning. The input to the network shall be the video stream from a camera behind the windshield of the vehicle. The desired network output is the predicted steering wheel angle, braking force and acceleration for the next milliseconds. My final goal is to have an autonomous vehicle driving itself in a simulated environment. The further goal is to deploy the created agent on a real-world track.

**2018/2019** - **Spring**

# Starting point of the project laboratory, previous works on the subject

## Introduction

Self-driving is one of the most challenging problems of the automotive industry at the moment. Most of the car manufacturers and other technology companies are working on different types of advanced driver-assistance systems (ADAS). Every ADAS is aiming to lower the number of actions made by the driver, thus making the trip safer and less tedious. One key technology behind these systems is the neural network.

Neural networks are widely used for image classification, stock price prediction, or even for natural language processing. The field of self-driving contains a lot of subfields, such as lane detection, traffic sign recognition, collision detection, and last but not least pedestrian movement prediction. These tasks can be solved separately with a few dedicated subsystems with or without utilizing a neural network. In this work, my goal was to create a deep neural network for solving the task of lane following.

## Theoretical summaries

Before I introduce the different type of approaches that I have been experimenting with this semester, I would like to provide an overview of the topic of self-driving and deep learning as well.

## Autonomous vehicles

Automating road transportation — even partly — has a lot of advantages. The first factor is safety: Even the simplest collision warning system can reduce the number of accidents on the road. In the field of self-driving, some kind of categorical structure is need to keep tracks of the different technologies.

According to the SAE standard [1], six levels of automation can be differentiated by who is in control of the vehicle and who is monitoring the environment:

1. No automation: Full-time monitoring and execution are required by the human driver. A vehicle is on level 0 even if it is equipped with warning or intervention systems.
2. Driver assistance: The human driver is still completely in continuous control of the vehicle, but may be assisted by different systems for different tasks. In most cases, adaptive cruise control, lane keeping system and parking assistance are implemented.
3. Partial automation: In some cases, the onboard system can take control of the vehicle, but the human driver always has to monitor the environment and must take over if needed. Keeping the hands on the steering wheel and the eyes on the road are required. Dynamic driving is completely performed by the human driver.
4. Conditional automation: The autonomous system is in full control of the vehicle and can determine in which cases human intervention is needed. The driver has to be prepared to take over, mostly for the dynamic driving tasks, such as driving in a roundabout.
5. High automation: No human driver attention is ever needed for safety. The onboard system can manage to control the vehicle in most of the driving situations. If the human driver does not take control when needed, the vehicle can safely abort the trip.
6. Full automation: The vehicle can handle all of the driving situations as good as a human driver would.

There are different types of techniques to navigate a car. One of the simplest one is called lane-following, which utilizes infrared sensors placed above the ground. This is a perfect solution for guiding robots in a well-defined environment, i.e. a factory, although it cannot be used in a real-world environment on its own.

The most intuitive way of driving a car on public roads is using vision: human drivers are making decisions based only on their sight. The computer-based implementation of this approach is a forward-facing camera used as a sensor. The video feed is processed by the onboard computer using traditional image processing techniques or neural networks. In this semester my goal was to use a deep neural network for the driving action prediction. As it turned out, there is an interesting, state-of-the-art solution for the steering prediction based on reinforcement learning called World Models [2]. It has been published less than a year ago, and I aim to explore its capabilities.

Other types of sensors can be used for monitoring the environment. I only highlight their strengths and weaknesses, because I don’t plan to use any of them later in this work.

Radar can be used to determine the distance of an object from the vehicle. It can be used in any weather condition, at high speed, and has a long range as well. The drawback is, that radar cannot determine the shape of an object precisely.

LIDAR is another technology for distance measurement, it creates a 3D map of the environment using a laser. LIDAR has higher precision, higher resolution, but shorter range than radar. LIDAR’s biggest weakness is that it is relatively expensive and also it has to be cleaned quite often because dirt can pollute the sensor.

Ultrasonic sensors are mostly useful for parking assistance, but they can be used for blind spot monitoring and clearance detection at lane changing as well.

Adapting more than one type of sensors ensures that monitoring the environment can be achieved in all kind of conditions.

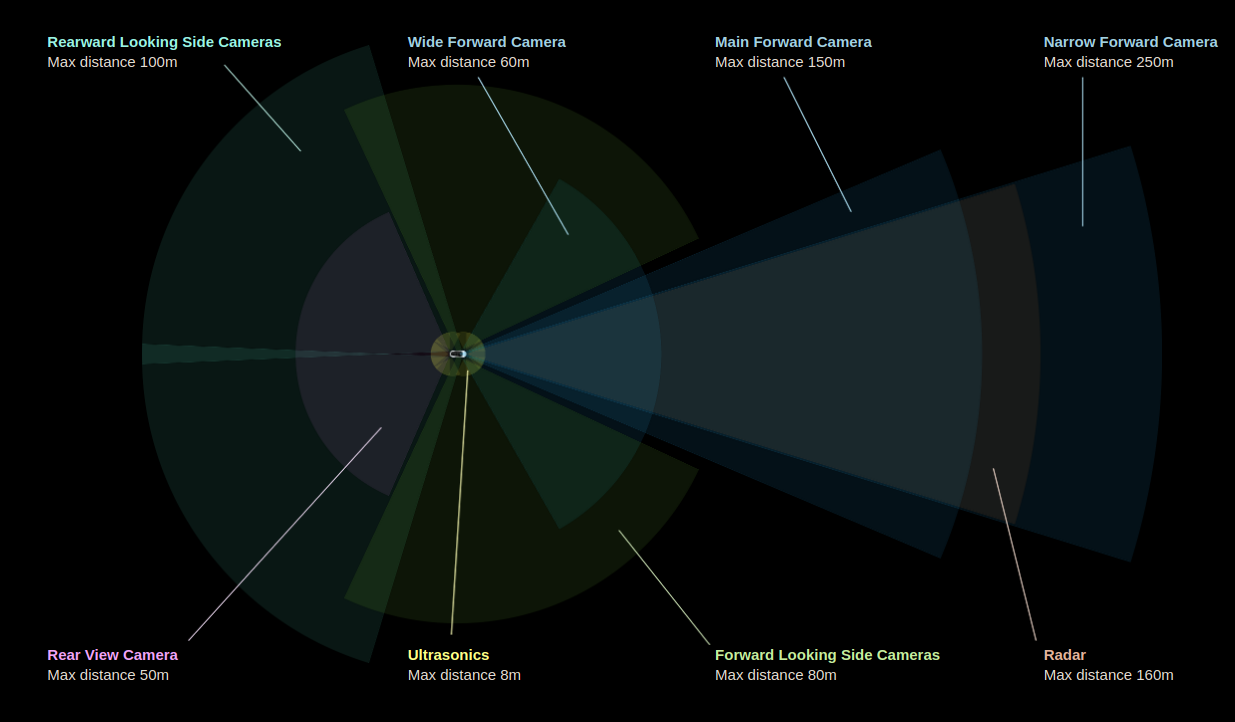


Figure 1.: A modern vehicle utilizes different types of technologies at the same time [3]

## Deep reinforcement learning

Some tasks, such as determining the species of an animal or identifying a voice sample are so complex problems, that they cannot be typically resolved using traditional techniques only. It would take too much work on the side of the developers and/or it would be exceedingly computational.

In the last couple of decades, neural networks have grown in popularity, and it started being a viable solution for many various kinds of problems. It has proved its capabilities in the fields of computer vision, natural language processing, voice recognition, and healthcare. Deep reinforcement learning is gaining performance as well, as DeepMind has shown with their AlphaGo[4] project.

It seems that maneuvering a car is a complex problem that could be solved by utilizing neural networks. Although deploying and testing self-driving systems in a real-world environment is easier said than done. Safety is the number one metric in the testing phase and at deployment as well. It goes without saying that no one should ever be harmed because of a runtime failure or a mistake made in the design of the system. At the moment, we cannot guarantee that a self-driving system purely based on neural networks will always perform as intended. There are a bunch of situations, where the system can fail and threaten the life of the passengers and/or the pedestrians.

A good workaround of these difficulties is creating a precisely simulated environment and utilizing reinforcement learning. The environment in most cases can be modeled by the Markov decision process (MDP). MDP consists of:

1. A set of possible states of the environment and the agent
2. A set of available actions in state
3. The probability that an action in a given state will lead to another given state
4. The reward acquired after the transition from one given state to another given state

The reinforcement learning process is the following in brief:

1. The agent takes an action
2. The environment gets changed
3. An observation and a reward are returned to the agent
4. The agent changes its policy in some way

In the environment, the agent can learn all the necessary features and it can be later adapted to the real world.

Training an agent in a virtual environment has a number of advantages and one significant weakness. One benefit is that the simulation can run at any speed, regardless of the laws of physics. The agent makes a decision and the environment can provide the observation immediately. The other advantage is that the simulation can be adjusted as needed, i.e. the difficulty of learning rare events can be solved effortlessly. The environment can generate and label a training set as well if required. The disadvantage of this technique is that the real world cannot be precisely and completely recreated, so domain adaptation has to be applied at the end of the training.

## Starting point of the project laboratory, previous works on the subject

In the matter of machine learning, the VITMAV45[5] course has showcased exceedingly well the theory behind deep learning for me, although I have never implemented a reinforcement learning agent before.

At the start of the semester, I did not receive any preceding work done on this topic.

# About the work done through the semester, achieved results

## Research

* 1. TODO

## 2.2 Summary

# Appendix

## Biblyography:

## Connecting documents and files:

[1] <https://www.sae.org/standards/content/j3016_201401>

[2] <https://arxiv.org/pdf/1803.10122.pdf>

[3] <https://www.tesla.com/en_EU/autopilot>

[4] <https://deepmind.com/research/alphago/>

[5] <http://smartlab.tmit.bme.hu/oktatas-deep-learning>