Autonomous Vehicle

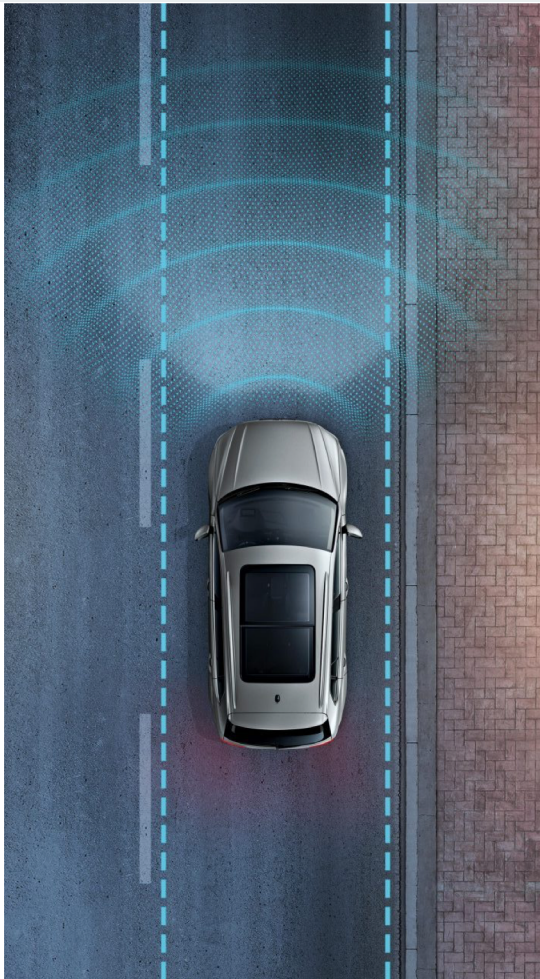
steering algorithm

Agenda

- Overview
- Phase 1 (Semester 4)
- RC Car
- Image recording
- Steerage
- Image-postprocessing
- Demo
- Further steps

Overview

- Big picture – autonomous steering algorithm
- Phase 1 – Creating training and test records which will be used to train the model
- Phase 2 – Implementing the autonomous steering algorithm
- Phase 3 – Bachelor thesis – finding the most suitable AI-framework for autonomous driving powered by a RaspberryPi



Phase 1

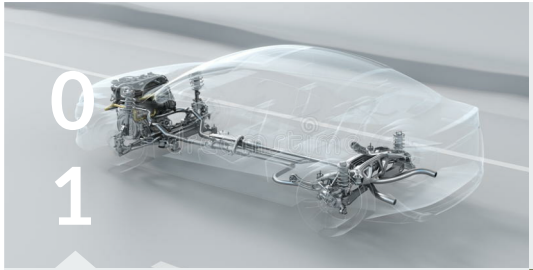


Image recording

Recording angle

Camera mount

03

Car

Steering

Speed

Steerage

Java client

Socket communication

Server management

Steering

PWM signal powers a servo motor via GPIO port on the RaspberryPi. Depending on the signal form the car changes it‘s steering angle.

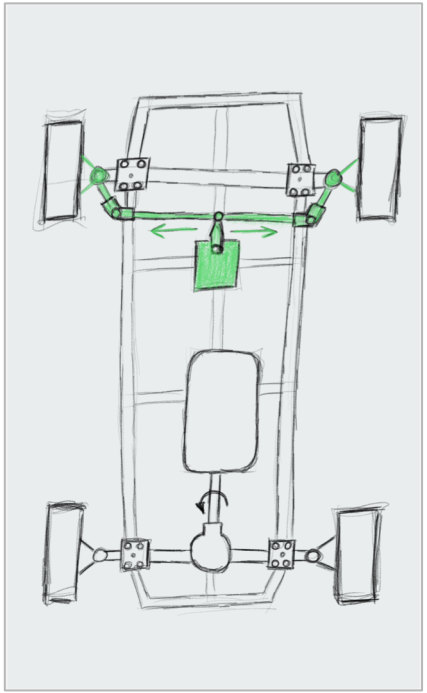
01 | Interpreting the steering command

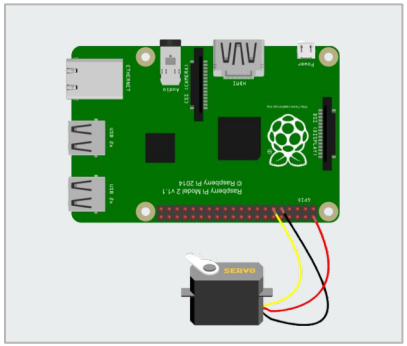
02 | Generating the PWM signal

03 | Sending signal via GPIO port to the servo

04 | Servo moves drag link

05 | Car changes it‘s direction





Speed

PWM signal powers a motor controller via GPIO port on the RaspberryPi. Depending on the signal form the car is accellerating or decellerating

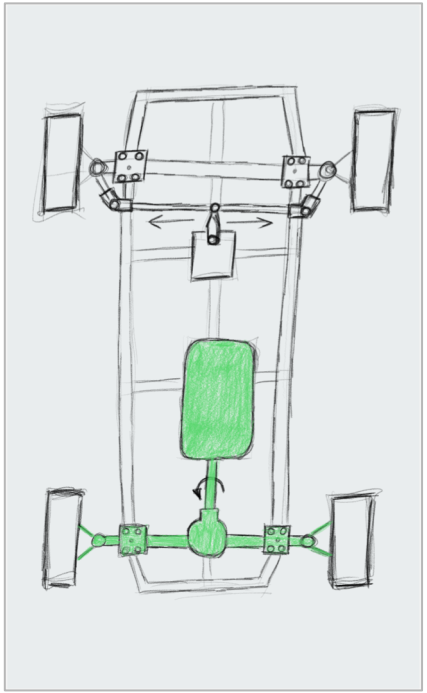
01 | Interpreting the speed command

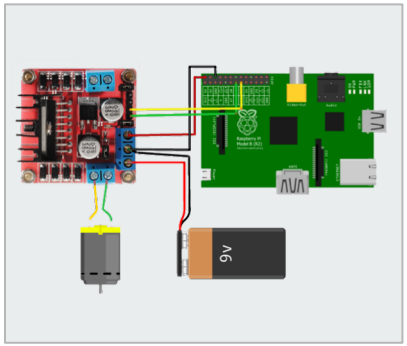
02 | Generating the PWM signal

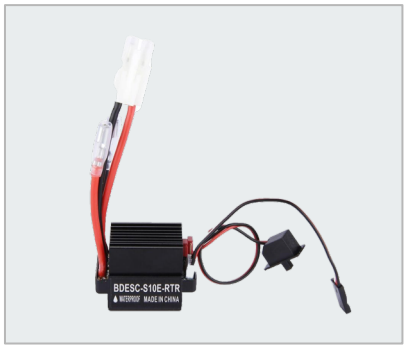
03 | Sending signal via GPIO to the motor controller

04 | Motor controller changes revolution speed

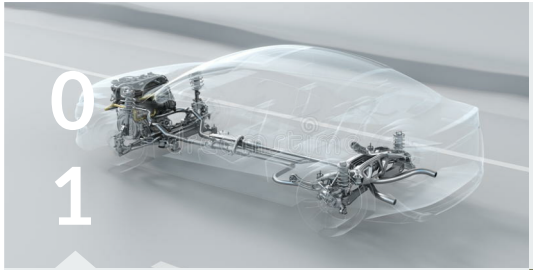
05 | Car is moving with appropriate speed







Phase 1



Car

Steering

Speed

Image recording

Recording angle

Camera mount

Steering

Java client

Socket communication

Server management

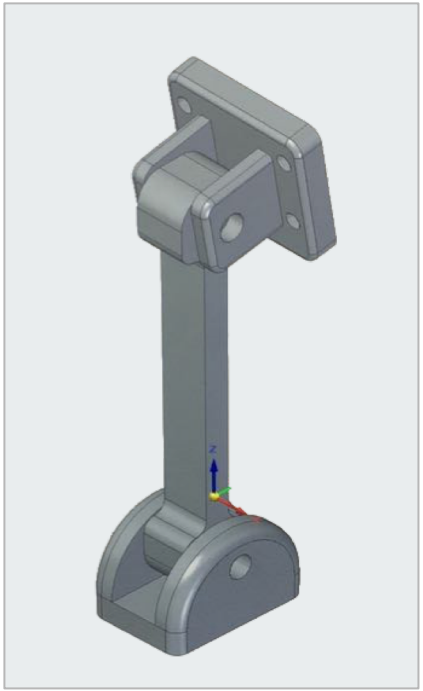
Camera

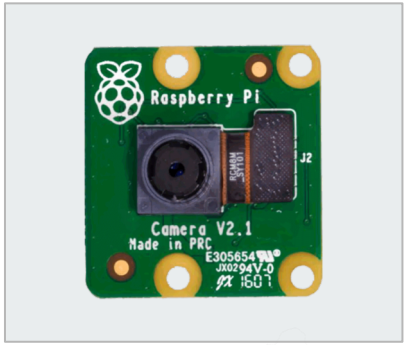
In order to be able to change the camera angle afterwards it is neccessary to design a camera mount which enables to do so.

01 | Camera position: Hood

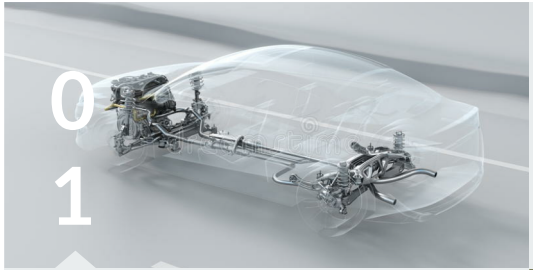
02 | Angle is adjustable in two axes

03 | 3D-print





Phase 1



Car

Steering

Speed

Camera recording

Recording angle

Camera mount

Steering

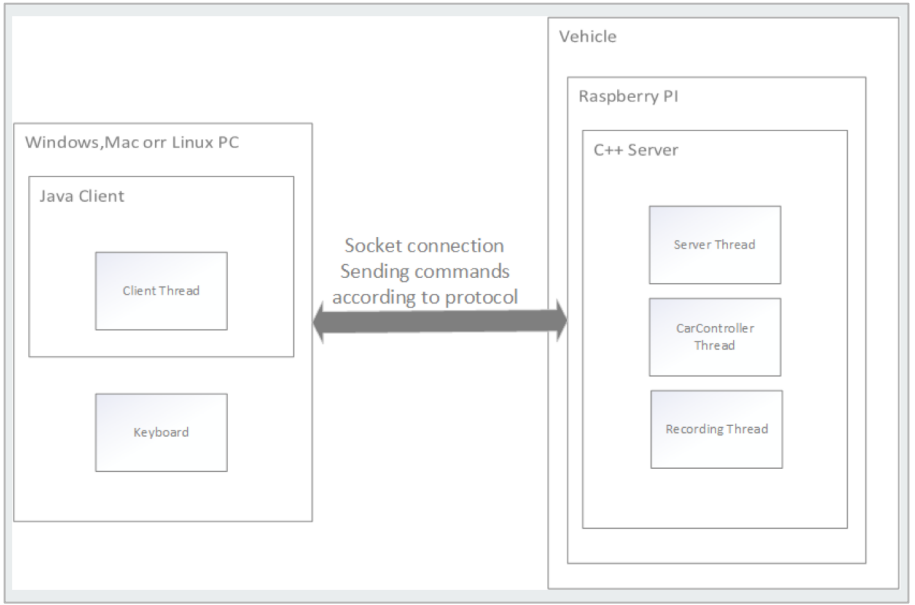
Java client

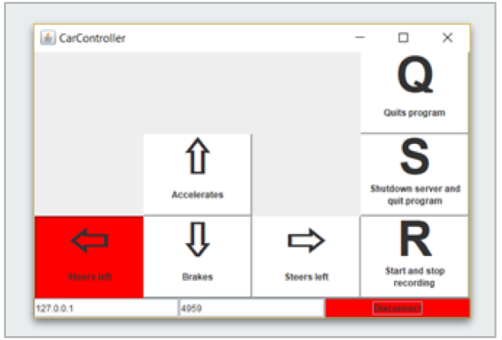
Socket communication

Server management

Network remote control

- C++ server on RaspberryPi
- Java client on Windows/MacOS/Linux
- Socket communication
- Sending commands from client to server via socket according to the communication protocol





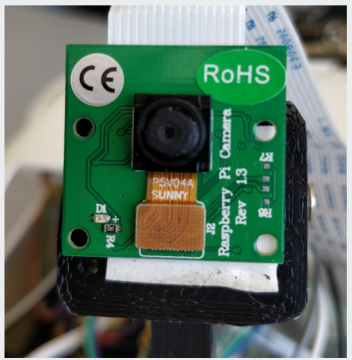
Communication protocol

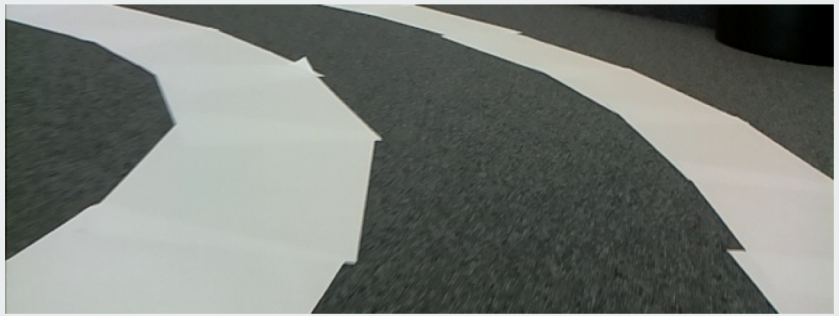
- Based on button pressed/released events
- Designed to easily add further commands
- Two types of steering mechanisms implemented

Taste	Zustand	Befehl	Aktion des Servers
Left (Keynr. 37)	pressed	left|pressed\0	Call CarController startLeft -> starts steering left
	released	left|released\0	Call CarController stopLeft -> stops steering left
Up (Keynr. 38)	pressed	forward|pressed\0	Call CarController forward -> move forward
	released	forward|released\0	Call CarController stopForward -> stop moving
Right (Keynr. 39)	pressed	right|pressed\0	Call CarController startRight -> starts steering right
	released	right|released\0	Call CarController stopRight -> stops steering right
.....	.....	.....	.....

Image recording

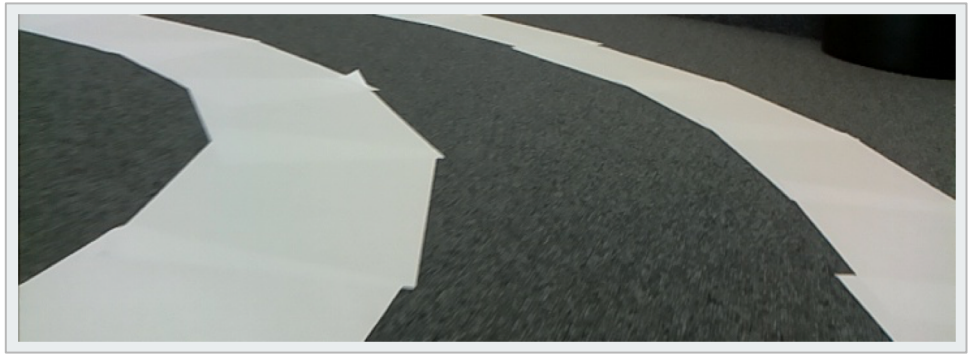
- Recording via PiCamera
- Saving the original images
- Extracting steering angle via shared memory
- Saving image according to steering angle in the appropriate folder (folder for each steering angle)

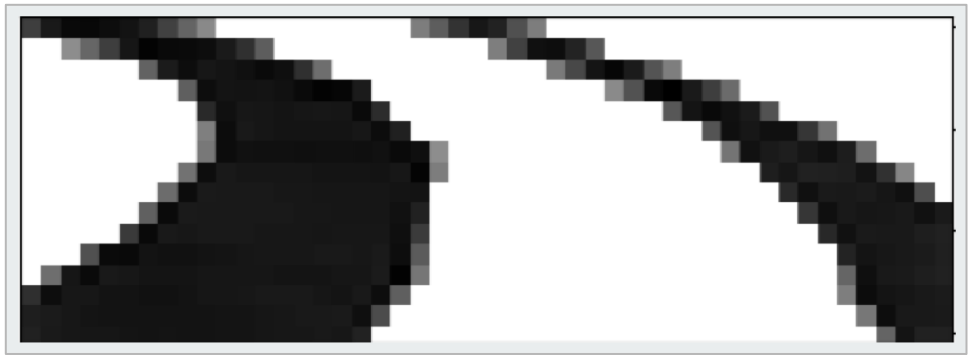




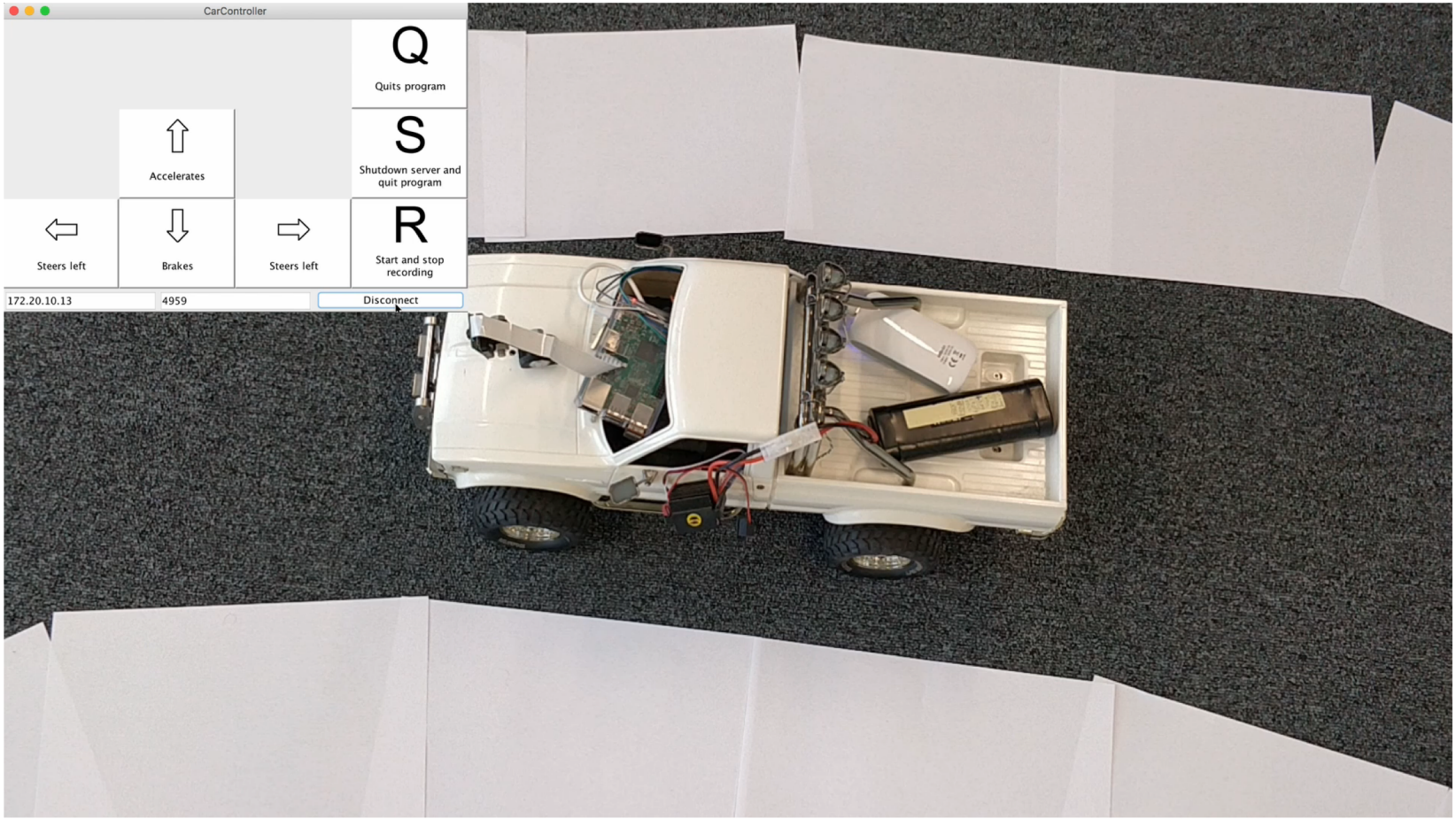
Postprocessing

- Original images saved
- Converting images into csv files
- Extracting the steering angle





Demo


CarController
Q
Quits program
S
Shutdown server and quit program
Accelerates
Steers left
Brakes
Steers left
R
Start and stop recording
172.20.10.13
4959
Disconnect

Further steps

- Driving the car in order to create as much training records as possible
- Autonomous steering
- Finding the most suitable model for this problem

Thank you for your attention