# 1. Methods

**Algorithm 1.0.1:** COLLECT DATA()

**comment:** Fill RolloutBuffer with samples obtained by current model

**procedure** COLLECTDATA()
$num\_steps \leftarrow 0$
$RolloutBuffer.\text{RESET}()$
$Env.\text{RESET}()$
**while** $RolloutBuffer\text{NOTFULL}()$

$\quad$ **do** $\begin{cases} obs \leftarrow Env.\text{GETOBSERVATION}() \\ action \leftarrow Model.\text{GETACTION}(obs) \\ reward \leftarrow Env.\text{STEP}(action) \\ num\_steps \leftarrow num\_steps + 1 \\ \text{ADDTOROLLOUTBUFFER}(obs, action, reward) \\ \textbf{if } Env.\text{ISFINISHED}() \\ \quad \textbf{then } Env.\text{RESET}() \end{cases}$

$\quad$ **return** $(num\_steps)$

**Algorithm 1.0.2:** TRAIN MODEL()

**comment:** Sample from replay buffer and update the model based on the loss

**procedure** TRAINMODEL()
$amount\_of\_batches \leftarrow \frac{rollout\_buffer\_size}{batch\_size}$
**for** $i \leftarrow 0$ **to** $n\_epochs$

$\quad$ **do** $\begin{cases} RolloutBuffer.\text{SHUFFLE}() \\ \textbf{for } i \leftarrow m \textbf{ to } amount\_of\_batches \\ \quad \textbf{do} \begin{cases} batch \leftarrow RolloutBuffer.\text{GETBATCH}(m) \\ loss \leftarrow \text{COMPUTELOSS}(batch) \\ Model.\text{BACKPROPAGATE}(loss) \\ Optimizer.\text{STEP}() \end{cases} \end{cases}$
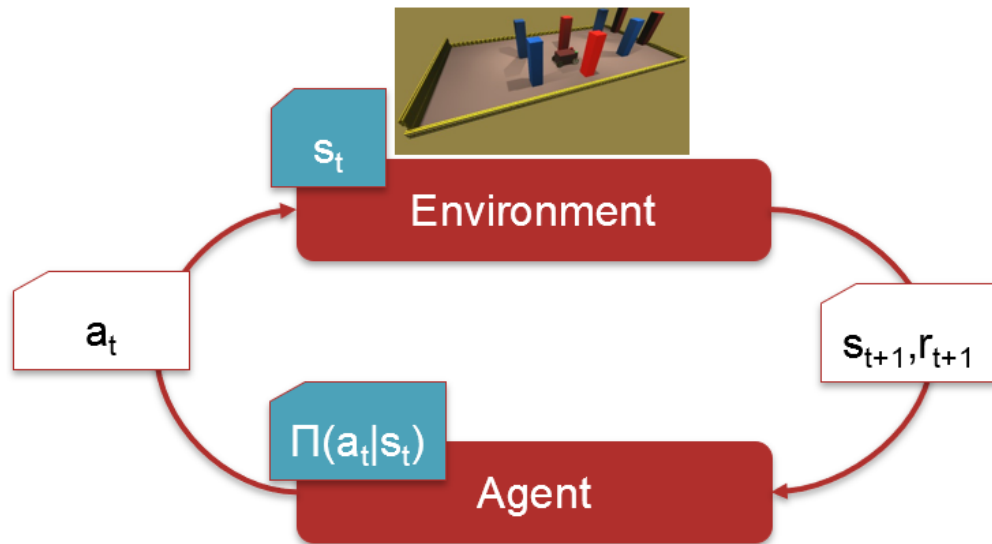
Georg Schneeberger
3707914

Abbildung 1.1.: Loop in Collect Data

**Algorithm 1.0.3:** Evaluate Model()

**comment:** Evaluate Model on tracks of all difficulties

**procedure** Episode()
while $Env.$IsFinished$() == False$
 **do** $\begin{cases} obs \leftarrow Env.\text{GetObservation}() \\ action \leftarrow Model.\text{GetAction}(obs) \end{cases}$
if $Env.$FinishedSuccessfully$()$
 **then** $success \leftarrow 1$
 **else** $success \leftarrow 0$
**return** $(success)$

**procedure** EvalModelTrack$(n\_episodes, difficulty)$
$successful\_episodes \leftarrow 0$
**for** $episodes \leftarrow 0$ **to** $n\_episodes$
 **do** $\begin{cases} Env.\text{Reset}(difficulty) \\ successful\_episodes \leftarrow successful\_episodes + \text{Episode}() \end{cases}$
$success\_rate \leftarrow \frac{successful\_episodes}{n\_eval\_episodes}$
**return** $(success\_rate)$

**procedure** EvalModel()
$easy\_success\_rate = \text{EvalModelTrack}(n\_episodes = num\_evals\_per\_difficulty,$
$difficulty ='' easy'')$
$medium\_success\_rate = \text{EvalModelTrack}(n\_episodes = num\_evals\_per\_difficulty,$
$difficulty ='' medium'')$
$hard\_success\_rate = \text{EvalModelTrack}(n\_episodes = num\_evals\_per\_difficulty,$
$difficulty ='' hard'')$

$total\_success\_rate = (easy\_success\_rate + medium\_success\_rate + hard\_success\_rate)/3$
**if** $total\_success\_rate > best\_success\_rate$
 **then** $\begin{cases} best\_success\_rate \leftarrow total\_success\_rate \\ Model.\text{SaveToFile}() \end{cases}$

**Algorithm 1.0.4:** TRAINNETWORK()

**comment:** Main Training Algorithm

**main**
$Env \leftarrow$ ENVIRONMENT(*environment_parameters*)
$RolloutBuffer \leftarrow$ ROLLOUTBUFFER(*rollout_buffer_size*)
$Model \leftarrow$ MODEL(*model_parameters*)
$Optimizer \leftarrow$ OPTIMIZER(*optimizer_parameters*)
$best\_success\_rate \leftarrow 0$
$num\_timesteps \leftarrow 0$
$iteration \leftarrow 0$
**while** $num\_timesteps < total\_timesteps$

$\textbf{do} \begin{cases} num\_steps \leftarrow \text{COLLECTDATA}() \\ num\_timesteps \leftarrow num\_timesteps + num\_steps \\ iteration \leftarrow iteration + 1 \\ \text{TRAINMODEL}() \\ \textbf{if } iteration \ \% \ log\_interval = 0 \\ \quad \textbf{then} \begin{cases} success\_rate \leftarrow \text{EVALMODEL}() \\ \text{LOGTOTENSORBOARD}(key ='' eval/success\_rate'', value = success\_rate, \\ step = num\_timesteps) \end{cases} \end{cases}$

Georg Schneeberger
3707914

# 2. Isolierte Ergebnisse

### 2.0.1. reward funktionen

Previous sections introduced the composite reward function. The composite reward function consists of a weighted sum of individual reward functions. The individual reward functions are designed to encourage the agent to learn the desired behaviour. The goal is to achieve an agent that completes the parcour without collisions, this is encapsulated in the event reward function. However the event reward function is a very spare signal, which makes it hard for the agent to learn. The other individual reward functions are designed to not be sparse, however some of these functions are not enough to guide the agent to the desired behaviour alone, such as the orientation reward. It is important to find appropriate weights of the individual reward functions for the composite reward function. We are conducting experiments to analyse the usefullness of the individual reward functions. We analyse if the agent is capable of learning the behaviour encouraged by the reward function.

see 2.1

Note: it also happened, that the agent turned around and drove backwards (towards the next goal) when given the distance reward only TODO run these experiments multiple times to see if the results are consistent

–> this shows the distance reward alone might not be the best approach to finding the best policy since it does not encourage driving forward. –> the agent policy got stuck (turning around and then driving backwards) –> from that point the agent cannot conttinue to improve since the agent looks in the wrong direction (important parts are not in field of view)

–> distance reward in der derzeitigen Form ist nicht ausreichend –> im isolierten easy Training lernt er immer mehr reward zu bekommen –> anstatt ins (letzte) Tor zu fahren dreht er das Hinterteil zum Tor

Tabelle 2.1.: Training runs with different reward functions alone (all coefficients 0 except the one of the reward function)

| function name | encouraged behaviour | learned behaviour | expected behaviour learned? |
|---|---|---|---|
| event reward | agent drives through the parcour without collisions | agent turns on the spot continuously | no |
| distance reward | agent drives towards the next goal | agent drives towards the next goal | yes |
| orientation reward | agent turns towards closest goal | agent turns around on the spot continuously | no |
| velocity reward | full speed ahead (no turning) | full speed ahead (no turning) | yes |

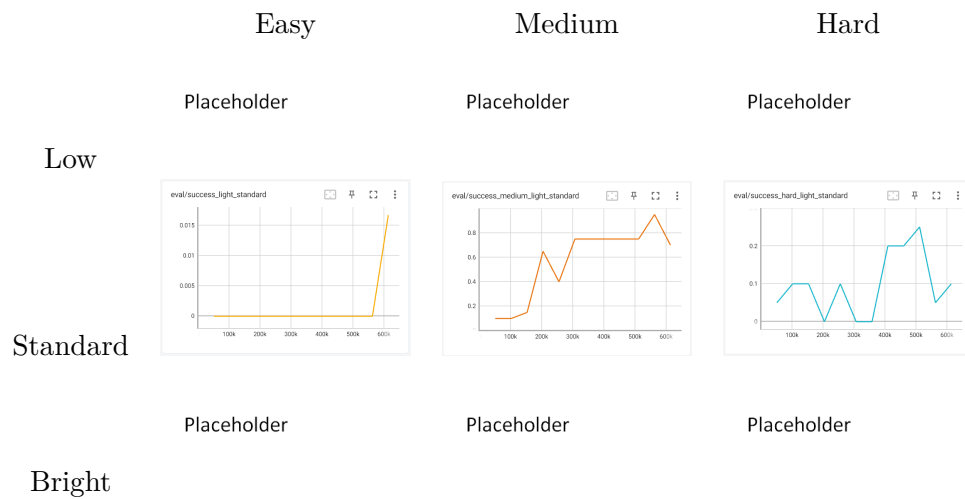| | Easy | Medium | Hard |
|---|---|---|---|
| Low | Placeholder | Placeholder | Placeholder |
| Standard | eval/success_light_standard | eval/success_medium_light_standard | eval/success_hard_light_standard |
| Bright | Placeholder | Placeholder | Placeholder |

Abbildung 2.1.: Figures of isolated training results for their specific setting

Placeholder

Abbildung 2.2.: easy Difficulty Success Rates for training on easy standard only

### 2.0.2. isoliertes Training

#### 2.0.2.1. light settings

The light setting does not play a significant role in the agent's performance?

### 2.0.3. generalization across difficulty settings

The graph shows that the agents are generally able to generalise to lower difficulty settings 2.5 .

### 2.0.4. easy standard isolated training

Agent learned to drive forward through the goals but turn around in front of the last one. This gives the agent more orientation reward but does not finish the parcour.

see isolated/easy-standard/ gifs

Placeholder

Abbildung 2.3.: Medium Difficulty Success Rates for training on medium standard only
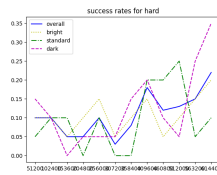
Abbildung 2.4.: Hard Difficulty Success Rates for training on hard standard only
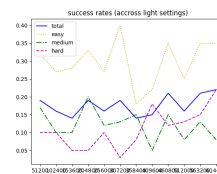


Abbildung 2.5.: total success rates for difficulties trained on easy, medium and hard only (standard light)
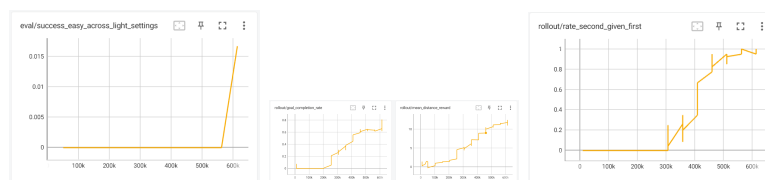


Abbildung 2.6.: easy standard

# 3. Evaluation and Experimentation

## 3.1. Evaluation Metrics

During the training and the final experiments, the agents are primarily evaluated based on the success rate, as well as the average completion time and the collision rate. Success rate is defined as the percentage of episodes in which the agent successfully completes the parcour. These metrics, which were previously used by [1], measure the agent behaviour's most important properties.

Additional metrics will be monitored during the training process to identify weak-points and erroneous behaviour of the agent. Monitoring the training process can provide insights into the agent's behaviour and aid in selecting appropriate hyperparameters. TensorBoard will be used to visualize the monitored metrics **??**. These metrics may include be the average cumulative reward, the average number of passed goals, the average distance travelled, the average amount of collisions, the average game duration and the average speed of the agent.

## 3.2. Experiments

The proposed experiments build on the scenarios from [1]. These experiments included three parcours with different difficulty levels 3.1 and conducted 3 different experiments with each trained agent. The first experiment used the same settings as the simulation environment. The second experiment was conducted under different lighting settings. The third one changed the motor power of the agent's two front wheels. All experiments primarily used the success rate to evaluate the agent's performance, the success rate is the proportion of successfully completed parcours.

The first two experiments will be used in this thesis as well, using the same experiments allows for an easy comparison to the previous research. The experiment with varying motor power will be omitted, since it is not related to the research goals of this thesis.

### 3.2.1. Research Question 1 - Is it possible to train an autonomous driving agent consisting of a convolutional neural network with end-to-end reinforcement learning to reliably solve the parcours of all difficulty levels?

The experiments under minimal changes will be used to judge if the agent is able to reliably solve all parcours. 3.1 shows three parcours of different difficulty levels, there are further variations of these parcours that change the positioning and colour of the obstacles. The parcours are evaluated using the success rate, if the agent is able to reliably solve all parcours the agent and training implementation can be considered successful.
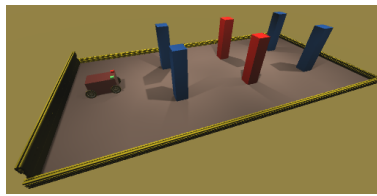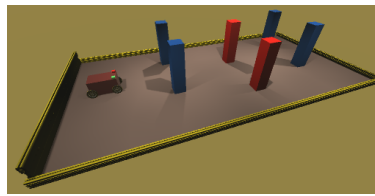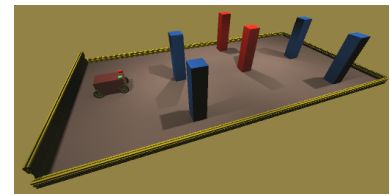
| (a) Easy | (b) Medium | (c) Hard |

Abbildung 3.1.: Evaluation Tracks of different difficulties

### 3.2.2. Research Question 2 - Is it possible to use an end-to-end trained CNN to make the agent robust to changing light conditions?

The evaluation parcours will be used with varying light settings to evaluate the agent's robustness towards changing light conditions. The agent's performance will be measured using the success rate, if the agent performs similarly across all light settings, the agent can be considered robust to changing light conditions.

### 3.2.3. Research Question 3 - Is it possible to use a neural network that can be transfered to a physical robot?

To answer question 3, the developed agent will be evaluated on the JetBot's processing unit. There will be no physical experiments with the JetBot due to time constraints. Instead a replay of an evaluation parcour is generated in Python. The replay is then processed on the JetBot to measure its processing capabilities. The replay will consist of input-output pairs and metadata, such as processing times. If the JetBot is able to reproduce the behaviour from the replay at sufficient speed, the agent can be considered transferable to the JetBot.
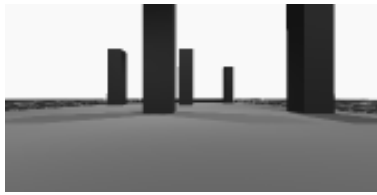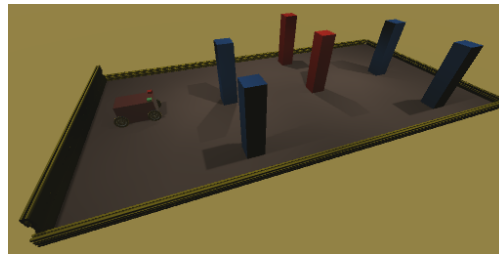
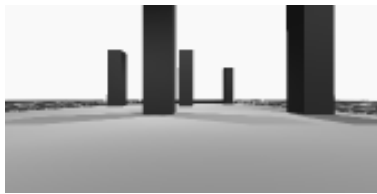(a) Standard Lighting Agent POV
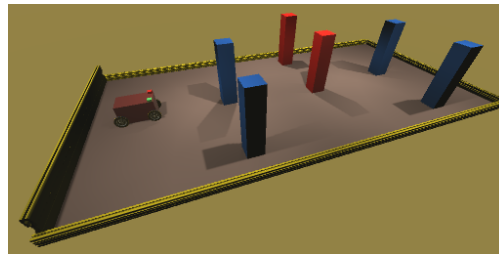


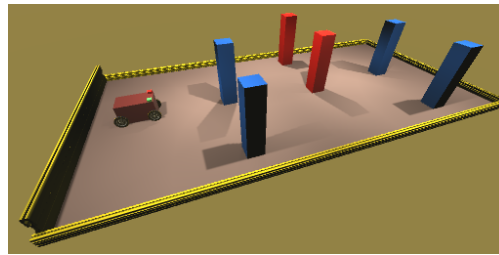(b) Standard Lighting Arena
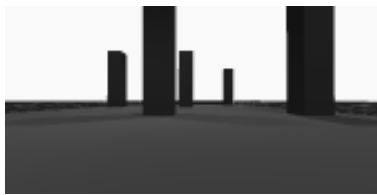


(c) Reduced Lighting Agent POV



(d) Reduced Lighting Arena



(e) Increased Lighting Agent POV



(f) Increased Lighting Arena

Abbildung 3.2.: Agent Camera POV and Arena Screenshots of the different light settings

# 4. Motivation

The increasing utilization of artificial intelligence in academia and industry have lead to massive efficiency improvements for all kinds of tasks. The development of autonomous vehicles promises to greatly reduce the number of traffic accidents and transportation cost [2]. As a result, researchers and private enterprises from all over the globe are making progress towards fully autonomous driving agents and integrating them in commercial vehicles, many companies started to integrate adaptive cruise control and lane centering assistance [3]. Due to the recent developments in artificial intelligence and the very high complexity of the task of autonomous driving, artificial intelligence often plays a big role in these systems [4].

Predictions for the future of autonomous driving have been very optimistic and although huge progress has been made, the task of fully autonomous driving is still far from being solved [5]. This thesis aims at contributing to the research in this field by applying reinforcement learning to autonomous driving agents in a simulated environment. This work builds upon the work of [1] and will use the same task and evaluation metrics. This thesis focusses on improving the agent's resiliency to changing light conditions by training a convolutional neural network end-to-end using reinforcement learning.

# 5. Research Goals

The goal of this thesis is to contribute in the domain of autonomous driving by investigating the use of reinforcement learning to train an autonomous driving agent that is resilient to changes in light conditions. The agent is evaluated on simulated parcours that consist of a series of goals indicated by two blocks, a parcour is successfully completed if the agents drives through all goals without collisions. This thesis builds upon previous work at the ScaDS.AI [1] and uses the same parcour and task specifications. The agent from previous work was not able to reliably complete parcours under changing light conditions, motivating the research goals of this thesis.

The self-driving agent is trained using reinforcement learning in a simulated environment, the training process will include changing light conditions and possibly data augmentation to help the agent generalize. Parcours of different difficulties and lighting settings are used to evaluate the agent's reliability and generalisation capabilities. The most important evaluation metric is the success rate. A parcour is considered a success when the autonomous driving agent passes all goals without any collisions.

## 5.1. Question 1 - Is it possible to train an autonomous driving agent consisting of a convolutional neural network with end-to-end reinforcement learning to reliably solve the parcours of all difficulty levels?

The previous work [1] showed that it is possible to train an agent using reinforcement learning to solve the evaluation parcours, however the trained agents were not successful in reliably traversing the parcours of higher difficulty levels. Furthermore this work will implement the agent in a fundamentally different way, the agents developed in previous work utilized an extensive preprocessing pipeline to extract the relevant information from the camera images whereas the agents in this thesis will use a convolutional neural network to learn and extract the relevant information themselves.

Due to these differences in implementation and as a prerequisite for question 2 and 3, it is first important to investigate if it is possible to train an agent to reliably solve the parcours of all difficulty levels. This raises question 1: Is it possible to train an autonomous driving agent consisting of a
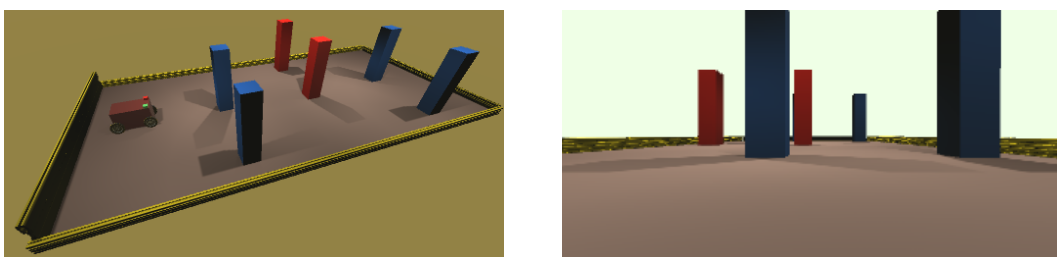


Abbildung 5.1.: Example image of a parcour and the agent's camera

convolutional neural network with end-to-end reinforcement learning to reliably solve the parcours of all difficulty levels?

The question will be answered by training agents that have been developed based on related work and analyzing their performance on the evaluation parcours. The evaluation parcours consist of different difficulty levels, the agent's success rate will be primarily used to answer the question.

## 5.2. Question 2 - Is it possible to use an end-to-end trained CNN to make the agent robust to changing light conditions?

While question 1 simply investigates if it is possible to train an agent to reliably solve the parcours of all difficulty levels, question 2 investigates if it is possible to train an agent that is robust to changing light conditions in addition to being capable of solving parcours of all difficulty levels. The performance of agents from previous work [1] declined massively under changing light conditions. This raises question 2 - Is it possible to use an end-to-end trained CNN to make the agent robust to changing light conditions?

The question will be answered by training agents that have been specifically designed to be robust to changing light conditions, the agents will be trained using reinforcement learning in a simulated environment with changing light conditions and possibly further data augmentation to help the agent generalize and learn. The agents will be evaluated on the evaluation parcours used in question 1 with changing light conditions. Similarly the success rate will be primarily used to evaluate and compare the agent's performance. The difference in performance for different light conditions will be used to answer the question, if the performance is similar for all light conditions the agent is considered robust to changing light conditions.

## 5.3. Question 3 - Is it possible to use a neural network that can be transfered to a physical robot?

One goal of the ongoing research at the ScaDS.AI is to build real life robots for demonstration and research purposes [6], the robots are based on the NVIDIA JetBot platform. The robots are equiped with a camera, wheels and a small computer. A future goal is to transfer a trained agent onto these robots, however the limited processing power of these robots might not be sufficient for more complex agents that utilize neural networks. This raises question 3 - Is it possible to use a neural network that can be transfered to a physical robot?

The question will be answered by investigating the processing power required to run the preprocessing steps and neural networks used in the agents that are developed in this thesis. This will be evaluated empirically by creating replays of the agents in simulations and running these replays on the physical robots. If the robots are able to reproduce the behaviour from the replays, the agents can be considered transferable to the robots.

# Literatur

[1] Maximilian Schaller. „Train an Agent to Drive a Vehicle in a Simulated Environment Using Reinforcement Learning". Magisterarb. Universität Leipzig, 2023.

[2] Johannes Deichmann u. a. *Autonomous driving's future: Convenient and connected.* 2023. URL: `https://www.mckinsey.com/industries/automotive-and-assembly/our-insights/autonomous-drivings-future-convenient-and-connected` (besucht am 09. 12. 2023).

[3] Mike Monticello. *Ford's BlueCruise Remains CR's Top-Rated Active Driving Assistance System.* 2023. URL: `https://www.consumerreports.org/cars/car-safety/active-driving-assistance-systems-review-a2103632203/` (besucht am 24. 10. 2023).

[4] B Ravi Kiran u. a. *Deep Reinforcement Learning for Autonomous Driving: A Survey.* 2021. arXiv: `2002.00444 [cs.LG]`.

[5] Collimator. *The State of Autonomous Vehicles: Seeking Mainstream Adoption.* 2023. URL: `https://www.collimator.ai/post/the-state-of-autonomous-vehicles-in-2023` (besucht am 16. 02. 2023).

[6] Merlin Flach. „Methods to Cross the simulation-to-reality gap". Bachelor's Thesis. Universität Leipzig, 2023.