Autonomous Vehicles: Learning and Simulation for Safe and Robust Behavior

Philipp Hildebrandt*
Hasso-Plattner Institute
University of Potsdam
Germany
philipp.hildebrandt@student.hpi.uni-potsdam.de

Finn Kaiser*
Hasso-Plattner Institute
University of Potsdam
Germany
finn.kaiser@student.hpi.uni-potsdam.de

Selina Raschack*
Hasso-Plattner Institute
University of Potsdam
Germany
selina.raschack@student.hpi.uni-potsdam.de

Christian M. Adriano†
Hasso-Plattner Institute
University of Potsdam
Germany
christian.adriano@hpi.de

Holger Giese‡
Hasso-Plattner Institute
University of Potsdam
Germany
holger.giese@hpi.de

ABSTRACT

In 2023 automated vehicles are deployed across multiple markets and in multiple cities across the world. This implies that different seasonal and geographical conditions, traffic rules, or types of vehicles have to be handled by these automated systems. Meanwhile, if we expect the public and regulators to trust the automated vehicle driving platforms, we need to provide approaches that can adapt to the complex tasks of automated driving in a robust and safe way. We studied this problem from the perspective of underspecification, which is one of the underlying challenges in designing and implementing a model driven approach for automated vehicle driving. We combined state-of-the-art proposals for a driving environment, a simulator, and a driving agent and build a flexible architecture for agent independent evaluation. We further built a modular set of scores to measure safety from a user-centered perspective. Finally, we modeled and compared the relationship of selected training inputs and safety score outcomes to address cases of underspecification. We developed and implemented (1) a method to evaluate models for automated vehicle driving independent from the models specific architecture, (2) a modular set of scores to evaluate safety in an agent's driving, and evaluated (3) a set of parameters to influence a model's behaviour with a Bayesian Multi-level Model approach. We were able to show underspecification when training in a simulation and we found patterns in how certain parameters influence the level of underspecification to support thriving towards a holistic approach to driving safety.

1 INTRODUCTION

"Ultimately, moving people and goods autonomously will have as much of an impact on our daily lives, economies, and societies as the invention of the car itself." [1], concludes Waymo CEO Dmitri Dolgov in his *WIRED* article "Road Robots Are Coming to the Rescue". In 2023 automated vehicles are deployed across multiple markets, e. g. delivery services, passenger transportation, or trucking, in multiple cities of for instance the U.S., England, or China. This implies that different seasonal and geographical conditions, traffic rules, or types of vehicles have to be handled by these automated systems.

Hence, the engineering challenge is to provide solutions that can adapt to the complex tasks of automated driving in a robust and safe way. Multiple technological challenges have been identified and assessed in the last couple of years. In his article "Failing to Distinguish between a Tractor Trailer and the Bright White Sky" [2] James Bridle analyses the case of Joshua Brown and his death in his self-driving Tesla Model S. While the driver had ignored safety regulations of being attentive to the traffic at all times during his ride, the sensors of the vehicle have failed to distinguish a white truck from the bright sky. To investigate observed higher error rates of machine learning models for certain demographic groups than others, in their work "Predictive Inequity in Object Detection" [3] Wilson et al. focused on the predictive performance on pedestrians with different skin tones and investigated to what extend different factors influence such a predictive bias.

Various approaches investigated the highlighted and other challenges in automated driving. In May 2023, in her article "Probabilistic AI that knows how well it's working" [4] Rachel Paiste summarized new advances in the area of model explainability and how to use these insights in a feedback loop to improve a model continuously. *Distributionally robust optimization* (DRO) [17] addresses sparsity in the training data and shifts the focus to a model's worst-case training loss rather than it's average training loss. *Domain-invariant learning* approaches the task for models to handle distribution shifts which is central to research on domain generalization, robust optimization, or fairness. [5] "Diversify and Disambiguate: out-of-distribution robustness via disagreement" by Lee et al., on the other hand, proposes a two-stage learning approach in which a set of

*All authors contributed equally to this research. The list of authors is sorted alphabetically.
†Advised on the research questions, wrote the introduction, the state-of-the-art, and the threats to validity.
‡Advised on the conceptual definitions and the framing of the research problem.
[1]Dolgov (2023): "Road Robots Are Coming to the Rescue".
[2]Bridle (2019): "Failing to Distinguish between a Tractor Trailer and the Bright White Sky".
[3]Wilson et al. (2019): "Predictive Inequity in Object Detection".
[4]Paiste (2023): "Probabilistic AI that knows how well it's working". Referenced paper: Alexander K. Lew, George Matheos, Tan Zhi-Xuan, Matin Ghavamizadeh, Nishad Gothoskar, Stuart Russell, Vikash K. Mansinghka: "SMCP3: Sequential Monte Carlo with Probabilistic Program Proposals". *Proceedings of The 26th International Conference on Artificial Intelligence and Statistics*, PMLR 206:7061-7088, 2023.

models is trained that incorporate existing underspecification in the data. Training in a simulation, e. g. a *Digital Twin (DT)*, is used to equip a model with as much driving experience as possible before deploying it into real world environments.

Nonetheless, the *state-of-the-art* is lacking in terms of addressing safety in automated driving in its entirety, as research is mainly focusing on the model training perspective. To help mitigating the gaps between existing solutions and technical challenges in the domain of *Automated Vehicle Driving (AV)*, the industry and governments formulated frameworks and collected use cases to define, train and evaluate models for automated driving, e. g. SAE's *six levels of driving automation* [12] or NHTSA's pre-crash typology [5].

Our *insight* was to tackle these challenges from the perspective of underspecification. To be able to include human-centered aspects to the question "What does safety mean?", we shifted from a model training perspective to a model evaluation perspective. In formulating scores related to driving safety rather than training accuracy the performance of an AV agent can be evaluated with a set of metrics the model had no chance to explicitly optimize for.

The problem that we investigate is the phenomenon of underspecification, which is one of the underlying challenges in designing and implementing a model driven approach for AV. Especially, if the problem of underspecification in model-based approaches for safe AV agents is significant. Building on existing technical solutions in this domain, this comprises (1) the aspect to be able to simulate (parts) of a real world sufficient well enough, (2) investigating if the phenomenon of underspecification is present in the domain of AV, and (3) in which ways a potential performance gap between agents trained in simulation compared to agents trained in the real world can be increased or decreased.

Our Approach combines (1) building a DT to simulate the aspect that AV agents are usually trained and evaluated in a lab environment before being deployed into the real world, (2) formulating metrics to measure driving safety related aspects, and (3) formulating possible configuration parameters for training inputs to address cases of underspecification. For the first part, we combine state-of-the-art proposals for a driving environment, namely the CARLA simulator which also provides an expert agent, the DriveGAN simulator, and a driving agent from the CARLA Leaderboard 1.0 challenge, specifically *Trajectory-guided control prediction (TCP)*. We leverage tools from *Design Thinking* to build a set a of scores to measure safety from a user-centered perspective for the second part. We then modeled and compared the relationship of selected training inputs and safety score outcomes for the third part.

Our contributions comprise (1) a method to evaluate models for AV independent from the models specific architecture, (2) a modular set of scores to evaluate safety in an agent's driving, and (3) a set of parameters to influence the models behavior. The *implications* of our results are promising in a sense that we were able to clearly show underspecification when training in a simulation and we found patterns in how certain parameters influence the level of underspecification to support thriving towards a holistic approach to driving safety.

[5]U.S. Department of Transportation. National Highway Traffic Safety Administration (2007):
Pre-Crash Scenario Typology for Crash Avoidance Research.

We structured the rest of this report as follows. In Section 2 we highlight state-of-the-art approaches to underspecification. In Section 3 we explain the preliminaries that are the foundations of our approach and summarize our research questions in Section 4. Our approach is then detailed in Section 5, followed by the results and discussion of the models for the AV agent in Section 6. We review the threats to validity in Section 7, while in Section 8 we position our contributions with respect to other related work. Finally, in Section 9, we summarize our contributions and future work.

2 STATE OF THE ART

The phenomenon of underspecification highlights poor and unpredictable behavior in machine learning models after they are deployed in real world domains. One of the sources for this is that models will include all available information if it results in better performance during training. Therefore, over-parameterized neural networks can be highly accurate *on average* on a test set while generalizing poorly on some of the groups.

Distributionally robust optimization (DRO) [17] focuses on minimizing a models worst-case training loss over a set of pre-defined groups. To address spurious correlations multiple approaches were investigated by the research community to leverage on the biases those models have. One direction is training a pair of models, where the first model's purpose is to support and improve the learning of the second one. In "Learning from Failure" by Nam et al. [15] examples, when the spurious correlation does not hold, are identified by such a biased helper model and then those examples are upweighted during the training of the second model. This approach was then enhanced and iterated on independently by multiple researching teams, e. g. in [14] or [5].

Underspecification is not limited to model training and how models make use of spurious correlations. Underspecification within a dataset results in the challenge that the observed data can be explained equally well by multiple hypotheses. "Diversify and Disambiguate: out-of-distribution robustness via disagreement" by Lee et al. proposes a two-stage learning approach in which a set of models that incorporate the underspecification in the data is trained. In supporting a different hypothesis for the same observations all of these models make different predictions when given the same inputs. During the disambiguate step one of the identified functions is selected with minimal supervision. [6]

While the highlighted approaches address underspecification from a model training perspective, in this project we investigated underspecification from a model evaluation perspective. In formulating scores related to driving safety rather than training accuracy the performance of an AV agent can be evaluated with a set of metrics the model had no chance to explicitly optimize for. This allows a differentiated view on underspecification: (1) we can analyze which safety aspects are more at risk of underspecification and (2) we can evaluate which hyperparameters favor certain aspects of safety with the help of *Multilevel Models (MLM)*. Ultimately, this knowledge will enable us to choose parameters for a training pipeline with respect to individual safety constraints. A suitable approach to address the underspecification will stay a case-by-case matter depending on the architecture of the driving agent, the used training data, or other factors of the learning approach.

3 PRELIMINARIES

In this section the project background is introduced, including the introduction of different software components we build our approach on. A summary of the general process can be found in Appendix A.1 and a summary of the engineering process in Appendix A.2.

3.1 Scenario

The domain we are working in is Automated Vehicle Driving (AV), specifically aspects of robustness and safety in AV.

The SAE International has established five levels of driving automation [12]. Levels three and above denote automated driving, wherein the AV system assumes full sustained control of dynamic driving tasks, instead of only assisting a human driver.

- In *Conditional Driving Automation (Level 3)* the AV system completes driving within the given *operational design domain* and can rely on a human as fallback.
- In *High Driving Automation (Level 4)* the AV system completes driving within the given *operational design domain* without human supervision.
- In *Full Driving Automation (Level 5)* the AV system completes driving in any domain and without human supervision.

Operating at level four or higher implies that the AV system handles unexpected situations and situations with uncertainty, without human intervention. These scenarios occur due to the inability to anticipate every potential situation during system design. Additionally, the system is constrained by the information provided by available sensors, introducing an element of uncertainty. The tasks of an AV system include responding to such situations, evading both stationary and moving obstacles and recovering from unsafe states.

Modern approaches for designing AV systems include tools and techniques from machine learning. This usually includes (1) formulating a model that comprises the different tasks of the AV system, (2) training it in a supervised or unsupervised approach, (3) deploying the model to the real world - in the context of AV to a vehicle that is to be automated, and (4) setting up a feedback loop to further improve the model after deployment. The complexity of AV tasks introduce the risk for underspecification during (1), (2) and (4). (2) is usually done in a simulation first which further introduces risks related to transitioning from simulation to the real world (*Sim2Real*) [6].

3.2 CARLA

In our setup CARLA will substitute as the "real world" driving environment. CARLA stands for "Car Learning to Act" and is an open simulator for urban driving. To support the training and evaluation of AV agents the simulator provides multiple urban maps with moving and stationary obstacles, an autopilot that can be used as an expert agent during *Imitation Learning (IL)*, or the activation of different weather conditions. [8] The autopilot is a machine learning model that uses restricted information like a segmented birds eye view of the map. This simplifies the driving scenario and enables the model to reach great levels of accuracy compared to models using only real sensors. Therefore, in multiple projects the predictions of the CARLA autopilot are used as ground truth for IL.

To collect data from the agent under test, CARLA provides different sensors that can be attached to a vehicle in the simulation. During this project we used the following sensors to measure driving safety related aspects during an agent's evaluation:

Collision Detector detects collisions between the actor it is attached to and something in the world. With this sensor we collected data about the normal impulse of a collision and the other actors, the agent collided with. These measurements are reflected in our collision-based safety scores.

Lane Invasion Detector detects when the actor it is attached to crosses a lane marking. With this sensor we collected data about the lane markings the agent crossed and whether the lane crossing is permitted w.r.t. traffic rules. These measurements are reflected in our traffic rules-based safety scores.

Obstacle Detector detects when the actor it is attached to has an obstacle ahead. The sensor can be configured with different hit radius and which type of objects to consider. With this sensor we collected data about "near misses", which means when the agent did not collide with the obstacle but in an emergency the distance would be too short to brake safely. We configured it with a hit radius of 0.5 and considered any type of obstacle, stationary and moving. These measurements are reflected in our traffic rules-based safety scores.

IMU Sensor provides measurements for an accelerometer, gyroscope, or compass of the actor it is attached to. With this sensor we collected data about the agents linear acceleration in m/s^2. These measurements are reflected in our subjective feeling-based safety scores.

The CARLA Leaderboard is an annually challenge in AV in which agents are tasked to reach a target destination following a predefined route without traffic infractions. The challenges includes traffic situations modeled after the traffic scenarios collected in the NHTSA pre-crash typology.

In this project we used the evaluation script from the CARLA Leaderboard 1.0 challenges [7]. The main metric is the *driving score*, which is a composed score based on the average distance driven of the total route length in percent (route completion) and the average infraction penalty coefficient in [0, 1] for multiple infractions (infraction penalties), e. g. collisions with pedestrians or other vehicles, red light infractions and others.

The CARLA Leaderboard evaluation is done in two towns, in 15 scenarios, each being repeated twice. Figure 1 shows the routes of the first scenario for each of the towns including the routes' waypoints.

3.3 DriveGAN

DriveGAN is a simulator based on a deep neural network architecture introduced by Kim et al. in 2021 [13]. Figure 2 shows the general

[6] Sim2Real problems have been addressed in multiple research, e. g.: Kadian et al.: "Sim2Real Predictivity: Does Evaluation in Simulation Predict Real-World Performance?". In: *IEEE Robotics and Automation Letters* 5, 4, 6670–6677, 2020; Balaji et al. (2019): "DeepRacer: Educational Autonomous Racing Platform for Experimentation with Sim2Real Reinforcement Learning".

[7] CARLA Leaderboard 1.0 challenges were hold annually from 2019 - 2022 and are based on CARLA 0.9.10.1.

Figure 1: CARLA Leaderboard evaluation towns: (l.t.r.) Town 02 and Town 05.

architecture of the DriveGAN simulator. As the name suggests, it is trained as a *Generative Adversial Network (GAN)* to produce realistic new frames given an initial frame and a row of actions. In this way, the simulator can generate sequences of artificial frames starting either from a real frame or a random initialization which is achieved by combining an auto-encoder architecture with a Sequence Model called *Dynamics Engine*.

Training a custom simulator based on the DriveGAN architecture can be done in two stages:

(1) The auto-encoder learns to transform driving images into a latent space and back into the visual space. This latent space consists of a theme vector, which includes information about the general setting of a frame like time of day, lighting or weather conditions, and a spatial vector, which contains information about the spatial scene of a frame like the path of the road, buildings, other cars or pedestrians.
(2) The weights of the auto-encoder are frozen and the dynamics engine learns to predict the next frame given a certain action. The dynamics engine operates only in the latent feature space, which means it gets a theme and spatial vector and learns to predict the next theme and spatial vector pair, which can in turn be decoded into an image.

While the auto-encoder uses a classical feature pyramid using convolutional layers, the dynamics engine combines *Long Short-Term Memory (LSTM)* and convolutional layers. Extending this approach to multiple views or sensors can produce multimodal inputs that are required by different existing AV agents.

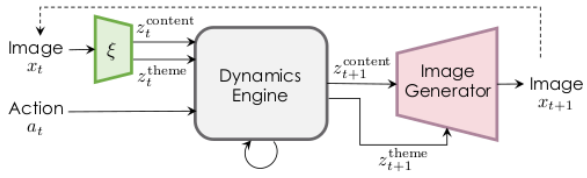

Figure 2: The general process of the DriveGAN simulator. Figure taken from [13].

3.4 Digital Twin

In the white paper "Digital Twin: Manufacturing Excellence through Virtual Factory Replication" Grieves introduces the concept of a Digital Twin (DT). The concept contains three main parts: "a) physical products in Real Space, b) virtual products in Virtual Space, and c) the connections of data and information that ties the virtual and real products together." [9, p. 1]

Since then the term has been adopted to many different domains and it has been revised to account for specific aspects of projects in those domains. In their review "Digital twin-based sustainable intelligent manufacturing: a review" He and Bai highlight a selection of expanded definitions on this term. [10, pp.4-5]

We will use the term DT in such an expanded manner, as we did not connect a real world agent and a real world environment with a virtual one in our project. We rather take the CARLA simulator as our "real world" and our DT based on the DriveGAN architecture simulates some of the inputs for a driving agent based on CARLA outputs.

3.5 TCP Agent

In their paper "Trajectory-guided control prediction for end-to-end autonomous driving: A simple yet strong baseline" [19] Wu et al. proposed an architecture that combines the benefits of predicting a next action and predicting a series of waypoints. Figure 3 shows the general architecture of the TCP model. The AV agent from the paper uses a current frame from a wide angle front mounted camera as well as some auxiliary information, like the current driving speed, and predicts a next control action for the automated vehicle. In the first step, the image is encoded into a latent feature space using a fine-tuned version of ResNet-34. Then, two main branches are utilized to predict an action from this latent feature space. While the first branch focuses on predicting a series of next actions directly, the second one predicts a series of next waypoints for the agent which in turn can be converted to actions using a PID controller. This trajectory branch is used to guide the action prediction in order to achieve higher quality action predictions. The next actions of both branches are then weighted and combined. The weight depends on the current state of the agent, taking into account situations in which one or the other method is known to usually be better.

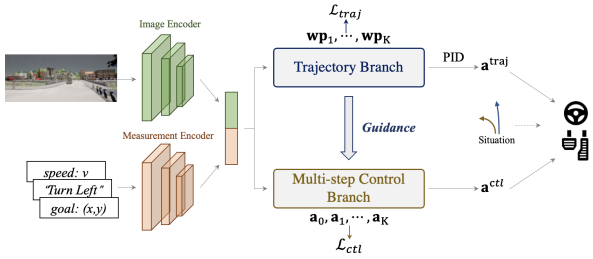

Figure 3: The general architecture of Trajectory-guided control prediction (TCP) models. Figure taken from [19].

4 RESEARCH PROBLEM

As discussed in Section 2, one of the challenges that have to be resolved when designing and implementing a model driven approach, is the phenomenon of underspecification.

In this project we want to investigate if the problem of underspecification in model-based approaches for safe AV agents is significant.

We think this is an interesting research problem, since in automated driving often multiple valid solutions to reach a destination exist. Previous research in this area [1] [11] has shown, that defining measurements for safety or robustness in AV is difficult.

4.1 Research Questions

Based on our general process as described in Appendix A.1 we want to investigate the formulated research problem by investigating the following research questions:

(1) Do parameters for our DT architecture and do CARLA test routes exist, with which agents that were trained with images from a DT score the same or better on unseen AV test routes from CARLA compared to agents which were trained with images from CARLA?
(2) Are there patterns or groups in the configuration of parameters for our DT and test routes from CARLA, with which the agent trained with images from the DT scores worse than the agent trained on the images from the CARLA simulator?

The first question addresses the aspect to be able to simulate (parts) of a real world sufficiently well enough. If this would not be the case then there would be no reason to use a simulator in the first place. The second question investigates if the phenomenon of underspecification is present in the domain of AV. These parameter sets can then be investigated further to see in which ways changing them may increase or shrink the performance gap between agents trained on a DT compared to agents trained in the real world.

5 APPROACH

Our approach combines (1) building a DT to simulate the aspect that AV agents are usually trained and evaluated in a lab environment before being deployed into the real world, (2) formulating metrics to measure driving safety related aspects, and (3) formulating possible configuration parameters for training inputs to address cases of underspecification. For this third part, we modeled and compared the relationship of selected training inputs and safety score outcomes.

5.1 Architecture Overview

Our approach focuses on evaluating AV agents rather than proposing a new one. Therefore, we build on state-of-the-art proposals for a driving environment, namely the CARLA simulator, the DriveGAN simulator, and a driving agent from the CARLA Leaderboard 1.0 challenge, specifically TCP. Figure 4 shows how we combined those different components into one pipeline.

Expert Agent The expert agent as described in Section 3.2 steers our car during the data collection. Its driving actions are used as ground truth labels for the agent training while the sensor readings for the image frames and additional meta information, like orientation, speed, and location information, are used as inputs for the TCP model.

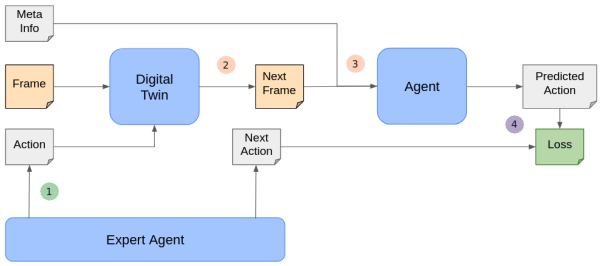

Figure 4: Architecture Overview: (1) the expert agent generates a ground truth action a_{gt-t-1} for a frame at time t-1, (2) based on the frame f_{t-1} and selected action a_{t-1} the DT predicts the current image frame f_t, (3) the agent takes the predicted current image frame f_t and optional meta data to predict the next action a_{t+1}, and (4) the predicted next action a_{t+1} is evaluated, based on the next ground truth action a_{gt-t+1} selected by the expert agent.

DriveGAN We used the DriveGAN simulator as described in Section 3.3 as the basis for our DT. We trained it using the data collected with the TCP data collection script and modified it to work with the action format used by TCP. This way, we can generate synthesized frames based on the real TCP dataset.

TCP As our agent we used TCP as described in Section 3.5. Since we modified DriveGAN to produce data in the format TCP expects, our agents could be trained with only minor modifications. This was important since we wanted to evaluate existing agents with minimal modifications.

We chose TCP as the architecture for driving agents because it needs visual input from only one camera. This allows us to build a DT with minimal complexity. If the agent needed multimodal inputs, our DT would need to be modified in order to simulate all needed sensors.

5.2 Safety Scores

We started with the question "What does safety mean?" which can be approached from different perspectives.

In the industry, multiple standards have been developed to define and implement safety systems. *Safety Integrity Levels (SIL)* measure a safety systems performance in terms of its probability of failure on demand. Therefore, the concept of functional safety was developed. This term describes "the safety system that is dependent on the correct functioning of the logic solver, sensors, and final elements to achieve a desired risk reduction level. Functional Safety is achieved when every safety function is successfully carried out and the process risk is reduced to the desired level" [8]. With respect to AV this can include monitoring a vehicle's engine and controls and incorporating safe actions in case of the detection of a malfunction.

[8] https://blog.msasafety.com/what-safety-integrity-level-means/, lastly accessed on October 30th, 2023.

The CARLA Leaderboard evaluates AV agents based on the "Pre-Crash Scenario Typology for Crash Avoidance Research" from 2007 by the U.S. Department of Transportation. Therefore, their perspective focuses on safety aspects related to the physical integrity of human actors and to traffic rules. This is measured in the different infraction penalties that modify an agents driving score.

For us it was important to enhance the human-centered perspective. Choosing the people sitting in the vehicle as a target group, additionally to aspects mentioned before, safety can be expressed in subjective scores as well, e. g. driving smoothness or the interpretability of an AV agents actions. The full set of aspects we came up with is added to the Appendix A.3.

5.3 Route Subsets

For evaluation we used the evaluation towns, Town 02 and Town 05, from the CARLA Leaderboard 1.0 challenges. The routes themselves are encoded as a list of waypoints, which allowed us to split the routes into subsets.

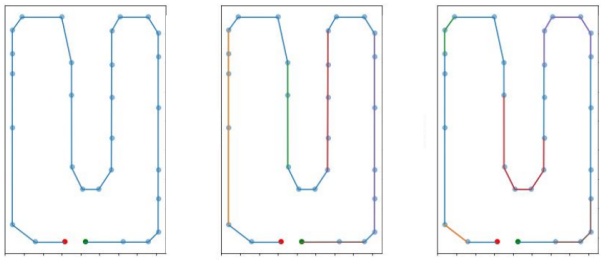

Figure 5: l.t.r.) Outline of CARLA Leaderboard Town 02, Town 02 straight routes, and Town 02 curved routes.

We created subsets of only straight and curved parts of the routes with curves including intersections. Figure 5 shows the first full route of Town 02 and the straight and curved route subsets we created for this town. We created subroutes from both of the routes shown in Figure 1 and we reused the scenario's weather condition during evaluation.

During evaluation the subroutes, together with the full evaluation town routes, are used to investigate our first research question.

5.4 Groups of Configuration Parameters

There are different ways to manipulate the training process of an agent based on the training input configuration. These groups of configuration parameters (*levers*) can be clustered as follows:

Direct Input Parameter influence the training process directly:

- Number of training iterations
- Time or epochs spent on DT vs. CARLA
- Sample Rate
- Latent Space Dimensionality
- Noise or other type of degradation of images

Meta Parameter influence the training process indirectly:

- Behavioral patterns of the expert agent, e.g. risky or patient driving style
- Alignment of the conditions in training and during evaluation, e. g. weather conditions or car placements on the maps
- different DriveGAN theme vectors

During evaluation the following parameters are used to investigate our second research question:

Mix Percentage is a continuous parameter and configures the time during training the agent spent on the DT and on CARLA:

- 0% - Agent trained fully on CARLA images
- 50% - Agent trained on CARLA and DT equally
- 100% - Agent trained fully on DT images

Forecast is a categorical parameter and configures the number of frames the DT predicts:

- 0 - for images from CARLA
- 16 - for DT images, predicted 16 frames in the future w.r.t. a given image
- 32 - for DT images, predicted 32 frames in the future w.r.t. a given image

Time is a categorical parameter and configures the lighting conditions w.r.t. to the time of the day:

- day - agent is trained only on daytime images
- night - agent is trained only on nighttime images
- all - agent is trained on a randomly mixed set of images (day- and nighttime)

This resulted in a total of 15 different configurations, in the rest of the report referred to as *experiment types*.

6 RESULTS

In this section the project results are presented and discussed. The results are grouped by different metrics based on agent training, the introduced safety scores, and the route complexity.

6.1 Training Condition Evaluation

This section focuses on the results by comparing them to different agents from the CARLA Leaderboard. In the first part we evaluate the results from our experiment types using the loss from TCP. In the second part we evaluate them using the scores from the CARLA Leaderboard.

6.1.1 TCP Loss. We used the combined loss, which is a weighted sum of the different loss components. We evaluated our agents on the test set with images generated by the DT and also the test set from CARLA. The results of this evaluation can be found in Table 1. As expected, the agents, that were trained on "real world" data from CARLA perform the best on the CARLA test set while the agents completely trained on the DT perform better than the ones trained on CARLA on the DT test set. This shows that our training with the DT works and there likely is a distribution shift in the data. While the agents completely trained on DT data have an about 30% higher test loss, the agents trained in the mixed condition perform only slightly worse on the CARLA test set. During training on the CARLA dataset, daytime training has an advantage over nighttime training which is reversed when training on DT data.

Portion DT	Forecast Steps	Time Filter	Loss DT	Loss CARLA
0%	0	All	2.1551	0.7144
		Day	2.2158	0.7329
		Night	2.2303	0.8056
50%	32	All	1.1885	0.7536
		Day	1.3457	0.7924
		Night	1.2501	0.8220
	16	All	1.0738	0.7346
		Day	1.2580	0.7657
		Night	1.3583	0.9785
100%	32	All	0.9744	0.9297
		Day	1.3460	1.1305
		Night	1.1339	0.9916
	16	All	0.9713	0.9433
		Day	1.3803	1.0452
		Night	1.1339	1.0240

Table 1: Comparison based on the different experiment types using the test loss of TCP.

6.1.2 CARLA Leaderboard Scores. In Table 2 we can observe a massive drop of route completion when we start training using the DT. Using the mixed training dataset cuts route completion in half. Training on DT data only, halves the scores a second time when training only on daytime and is reduced by almost three-fourths in the other conditions. On all the daytime data, training fully on the DT with 32 steps forecast achieves 12.78% of the route completion from the baseline. The same pattern applies to the composed score, as shown by the score penalties when training on the DT which are similar or higher. This big difference of score degradation compared to the TCP test loss clearly shows underspecification in this training setup. While training on the DT only increased the loss on the CARLA test set by 30%, the CARLA Leaderboard score based on actual driving in the simulator decreases by up to more than 80% when fully training on the DT.

Portion DT	Forecast Steps	Time Filter	Route Completion	Penalty	Score Composed
0%	0	All	82.84	0.6376	52.81
		Day	76.17	0.4855	36.98
		Night	74.21	0.5496	40.78
50%	32	All	50.05	0.4662	23.33
		Day	55.95	0.2789	15.6
		Night	46.29	0.6691	30.97
	16	All	54.2	0.4913	26.62
		Day	40.82	0.5098	20.81
		Night	46.99	0.6045	28.4
100%	32	All	10.59	0.7071	7.48
		Day	23.75	0.4214	10
		Night	12.92	0.7763	10.02
	16	All	12.95	0.771	9.98
		Day	20.02	0.5235	10.48
		Night	12.73	0.7328	9.32

Table 2: Comparison based on the different experiment types using the evaluation scores from CARLA Leaderboard.

6.2 Safety Scores

This section covers the evaluation of the different experiments using our proposed safety scores. This highlights how sensitive individual safety aspects are to underspecification.

6.2.1 Collision-based Scores. When looking at the number of collisions per distance driven as shown in Table 3, we can also observe the phenomenon of underspecification. While there are already more collisions when mixing DT data with CARLA data, it further increases drastically when training fully on the DT. We can also see that agents collide more frequently with stationary obstacles than with other vehicles once we incorporate the DT training data. The average collision impulse however decreases for agents trained with the DT. Since the maximum impulse stays similar for all training conditions this does not mean that agents trained on the DT are more careful. Instead, many collisions with low impulse lower the mean impulse.

Portion DT	Forecast Steps	Time Filter	Collisions / Km	Mean Collision Impulse	Interaction with
0%	0	All	0.72	8494.93	vehicle
		Day	1.33	7830.77	vehicle
		Night	1.66	7154.9	vehicle
50%	32	All	3.45	3061.1	static
		Day	5.11	3395.38	static
		Night	4.09	4407.44	static
	16	All	3.25	4321.04	static
		Day	4.16	2940.7	static
		Night	3.04	5612.49	static; vehicle
100%	32	All	38.42	1099.18	static
		Day	16.13	2600.02	static
		Night	26.64	802.71	static
	16	All	25.11	1012.24	static
		Day	11.89	3020.88	static
		Night	11.53	1061.67	static

Table 3: Comparison based on the different experiment types using the number of collisions relative to the distance driven in kilometers, the mean impulse of these collisions and the most frequent type of object the actor collided with.

6.2.2 Traffic Rules-based Scores - Brake Path Shortage. Mixing DT data with CARLA data does not seem to cause the agent to come closer to other objects more frequently as shown in Table 4. Only when trained fully on the DT data the number of near misses increased. Still, the increase is not as large as with many of the other scores. Furthermore, the mean distance the agent approaches to other objects during near misses does not substantially change in our different experiment types. This means that this safety aspect is comparably resistant to underspecification.

6.2.3 Traffic Rules-based Scores - Lane Infractions. We can clearly observe underspecification in the number of lane infractions per kilometer driven in Table 5. The more we mix in frames from the DT into the training set, the more lane infractions happen. Increasing the number of forecast steps further increases the number of lane

Portion DT	Forecast Steps	Time Filter	Near Misses / Km	Mean Brake Path Shortage
0%	0	All	0.043	1.704
		Day	0.204	0.748
		Night	0.342	2.462
50%	32	All	0.034	1.153
		Day	0.628	1.237
		Night	0.321	1.881
	16	All	0.248	0.942
		Day	0.124	0.977
		Night	0.401	1.3
100%	32	All	1.53	1.208
		Day	2.151	1.055
		Night	1.055	0.427
	16	All	0.401	0.396
		Day	1.475	1.438
		Night	0.563	0.385

Table 4: Comparison based on the different experiment types using the number of near misses to the distance driven in kilometers, and the mean distance the actor drive too close to the other object in meters.

infractions. Using nighttime images only, training with images fromCARLA increases the number of lane infractions, while there is no such clear relationship with the DT images. Furthermore, bad lighting conditions make it more difficult for the agent to recognize details like lane markings on CARLA images and they often seem to be missing on DT images independent of the daytime.

Portion DT	Forecast Steps	Time Filter	Lane Infractions / Km
0%	0	All	5.46
		Day	7.95
		Night	10.24
50%	32	All	23.83
		Day	22.88
		Night	23.1
	16	All	20.67
		Day	22.56
		Night	19.63
100%	32	All	52.19
		Day	43.43
		Night	22.95
	16	All	21.37
		Day	34.84
		Night	39.09

Table 5: Comparison based on the different experiment types using the number of lane infractions relative to the distance driven in kilometers.

6.2.4 Subjective Feeling-based Scores. We implemented two scores to measure some aspects of the subjective driving feeling, a stop-and-go score and an emergency brake score. In Table 6 we see that agents fully trained on the DT drive more steadily which is captured in the stop-and-go score. On the one hand, this could be interpreted as driving more confidently, on the other hand, it could also be due to the fact that these agents miss important details around the road, as mentioned when discussing the lane score results. These additional details may cause agents trained on CARLA data to accelerate or slow down. For all of our experiment types no emergency brakes happened and the small deviations from one are due to measurement and detection errors. We still think, this score captures an important aspect of driving smoothness and should stay included in future experiments.

Portion DT	Forecast Steps	Time Filter	Mean Stop and Go Score	Mean Emergency Brake Score
0%	0	All	0.491	0.999
		Day	0.548	0.998
		Night	0.531	0.995
50%	32	All	0.561	0.996
		Day	0.585	0.998
		Night	0.613	0.993
	16	All	0.586	0.997
		Day	0.619	0.996
		Night	0.6	0.997
100%	32	All	0.755	0.999
		Day	0.718	0.998
		Night	0.744	0.999
	16	All	0.76	1
		Day	0.693	0.998
		Night	0.761	0.998

Table 6: Comparison based on the different experiment types using the mean stop-and-go score and mean emergency brake score.

6.2.5 Availability. The availability of the agent is measured in the distance driven. This is proportional to the route completion. In Table 7 we can see that the distance driven decreases the more we train on DT data. Surprisingly, training only on daytime images results in more availability when training fully on the DT. This shows that the availability is also clearly affected by underspecification.

6.3 Route Subsets

As described in Section 5.3 we created subroutes from the CARLA Leaderboard evaluation towns. Table 8 summarizes the aggregated results for curved driving and Table 9 for straight driving for the implemented safety scores. Throughout our different safety scores we can see that curved driving seems to be a more difficult challenge for our agents which also seems to be more susceptible to underspecification. We do not see a general trend of more lane infractions or collisions when testing on the straight route subsets, while this relationship is clearly visible when testing on the curved route subsets. This relationship is also visualized in Figure 6. We showed before that training with the DT tends to result in an improved stop-and-go score. This improvement is greater when testing on the straight subsets. While the distance driven decreases in both

Portion DT	Forecast Steps	Time Filter	Distance Driven in Meter
0%	0	All	47035.40
		Day	44149.84
		Night	40904.93
50%	32	All	29534.58
		Day	33433.08
		Night	24925.62
	16	All	32217.63
		Day	24239.45
		Night	24954.85
100%	32	All	5882.09
		Day	12087.49
		Night	7580.30
	16	All	7487.09
		Day	10848.24
		Night	7110.56

Table 7: Comparison based on the different experiment types using the availability as mean distance driven in meters.

cases, the decrease is more rapid on the curved subsets. Only the relative number of near misses seems to be lower on the curved route subsets. In general this helps us understand that underspecification, especially in cases when the agent was fully trained on the DT without ever seeing images from CARLA, seems to be a bigger problem the more complicated the test scenario is.

%DT	FS	Time Filter	Lane Score / Km	Coll. / Km	Near Misses / Km	Stop & Go Score	Distance Driven
0%	0	A	11.45	0.87	0.00	0.466	1503.82
		D	13.12	1.18	0.71	0.441	1702.66
		N	14.65	2.14	1.43	0.440	1695.91
50%	32	A	36.62	9.91	2.78	0.500	1406.68
		D	34.13	9.06	1.49	0.465	1573.36
		N	68.07	17.25	1.40	0.418	1212.28
	16	A	46.77	8.85	0.46	0.494	1625.40
		D	41.37	4.66	0.55	0.527	1349.52
		N	45.96	12.77	0.49	0.490	1520.86
100%	32	A	66.04	45.35	5.16	0.726	344.88
		D	71.44	26.67	6.38	0.604	557.80
		N	52.30	11.90	4.52	0.587	336.14
	16	A	68.78	20.49	0.00	0.700	501.65
		D	51.13	13.79	0.36	0.631	828.84
		N	59.90	34.27	0.00	0.605	336.93

Table 8: Comparison based on the different experiment types using selected safety and availability scores on curved route subsets.

6.4 Model Visualization

Having discussed the different results based on the TCP loss, the CARLA Leaderboard score and our safety scores, we now try to understand why different agents behave differently. We will look into the embedding space of the models behind the agents and use

%DT	FS	Time Filter	Lane Score / Km	Coll. / Km	Near Misses / Km	Stop & Go Score	Distance Driven
0%	0	A	8.93	2.29	0.87	0.492	1740.16
		D	15.44	3.60	0.92	0.482	1697.70
		N	5.82	2.35	0.92	0.469	1702.82
50%	32	A	12.26	1.71	1.35	0.565	1559.21
		D	9.38	3.48	1.04	0.601	1531.44
		N	7.93	2.54	1.27	0.601	1448.95
	16	A	0.00	0.00	0.00	0.622	1770.68
		D	8.36	1.35	1.01	0.595	1551.59
		N	7.39	1.54	0.92	0.579	1630.83
100%	32	A	0.00	1.92	1.28	0.799	549.14
		D	8.16	1.26	0.94	0.710	1098.91
		N	0.00	2.58	1.55	0.762	562.95
	16	A	8.71	3.15	1.24	0.720	834.47
		D	4.78	2.17	1.30	0.684	906.12
		N	22.63	3.39	1.70	0.718	627.13

Table 9: Comparison based on the different experiment types using selected safety and availability scores on straight route subsets

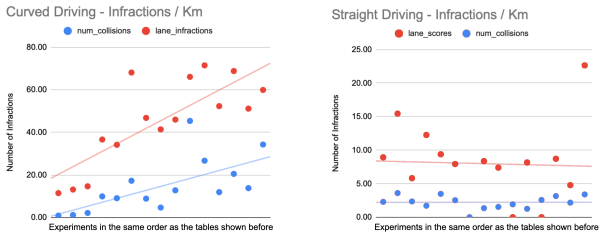

Figure 6: Trends in lane infractions and number of collisions when evaluated on curved (left) and straight (right) subroutes only.

Integrated Gradients (IG) [9] to indicate which parts of an image the model focuses on.

6.4.1 Image Focus. Integrated gradients propagates the gradients from one output score through the model back to the input image. This lets us visualize the gradients in the input space. Higher gradients mean that this part from the input image had a big influence on the model output. We can see that an agent trained fully on CARLA images clearly focuses on important aspects of the image like the edges of the street, the curb and lane markings. An agent fully trained on DT images also focuses on big objects like the bright white curb but misses important details like lane markings. Since the DT looses small details when generating images, this agent could never learn to focus on the details the agent trained on CARLA images focuses on.

6.4.2 Image Embeddings. TCP uses a pretrained ResNet-34 to extract information from the visual inputs. This network is then fine-tuned together with the different prediction branches during the training of TCP. We visualized the embeddings of the test images

[9] Mukund Sundararajan, Ankur Taly, Qiqi Yan (2017): "Axiomatic Attribution for Deep Networks".

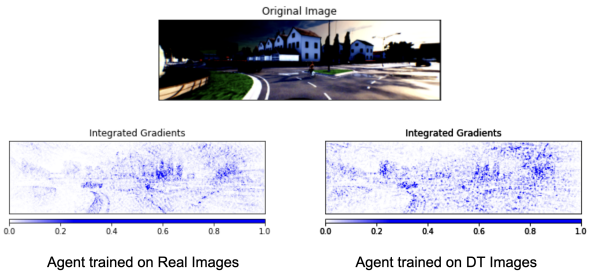

Figure 7: Model Interpretability: the input image given to a model (top), the IG of a model trained on real world images (left) and the IG of a model trained on synthesized images (right).

by performing a *Principal Component Analysis (PCA)* on two components as shown in Figure 8. The first thing to highlight is, that daytime and nighttime images produce two separate clusters. In order to simplify the rest of the discussion, we will focus only on the nighttime images, but the same analysis can be done considering only daytime images or images from both groups.

In the top row, on the right, we can see that models trained fully on images from CARLA map images from CARLA to a different cluster in the embedding space than images from the DT. Models trained on both types of images map them closer together and models fully trained on DT images only produce one cluster with images from CARLA and the DT. While this might look like the distribution shift would be minimal for models fully trained on the DT, when sifting images from CARLA, we can see in the second row, that the same series of images gets mapped to two different locations in the embedding space. Combined with the insights from the previous section we come to a different conclusion: a model fully trained on DT images never learns to recognize fine details and only extracts more general features. Since the images from CARLA and the DT share those general features, they get mapped to the same cluster. The more we mix in CARLA images to the training set, the more the model learns to focus on smaller details in the images. This also helps distinguishing the embeddings of CARLA images and DT images, since the details are only present in CARLA images.

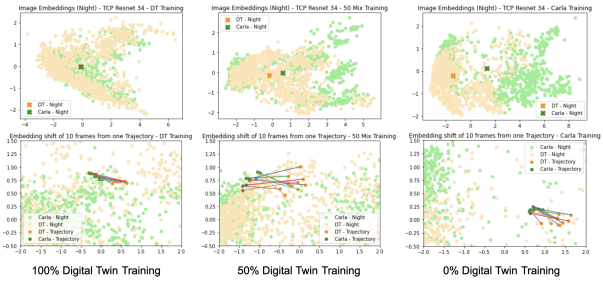

Figure 8: Visualization of the embeddings of the input images: (l.t.r.) general trends for mappings when trained with 100%, 50% and 0% mix percentage.

6.5 Model Comparison

The different experiment types discussed so far, result in different models for the AV agent. When comparing theses models it is important to highlight that the data for the different safety scores is hierarchical: based on the type of experiment we can cluster the observations w.r.t. the evaluated parameters. This also suggests that we can use estimates from the different experiments as a prior. We therefore used the following Bayesian Multi-Level-Models (Bayesian MLM):

$$\begin{aligned} \textit{safety score } y \sim 1 + \beta_1 \texttt{mix percentage} + \beta_2 \texttt{forecast} + \beta_3 \texttt{time} \\ +(1 + \beta_1 \texttt{mix percentage} + \beta_2 \texttt{forecast} + \beta_3 \texttt{time} \\ \texttt{| experiment type}) \end{aligned} \tag{1}$$

This results in eight different models for each safety score y. Therefore we limited the evaluation based on safety scores to the following five scores:

- Distance driven
- Lane infractions
- Collisions per driven distance
- Brake path shortage
- Mean stop-and-go score

For the comparison of the resulting 40 models we decided to estimate the pointwise out-of-sample predictive accuracy via *Leave-one-out Cross Validation using Pareto-smoothed importance sampling (PSIS-LOO)* and to apply a model-based criteria to address model optimism and the risk of overfitting. Since our data is hierarchical, and therefore generally singular [10], we used the *Widely Applicable Information Criterion (WAIC)*. [18]

We analyzed the chosen safety scores using one Multilevel Model each. Each time we chose the respective model as a compromize between optimal PSIS-LOO and WAIC scores and a maximum number of explained parameters.

We can use the estimated coefficients for our evaluated hyperparameters in order to analyze the influence of them on underspecification. High absolute estimates mean that tuning a certain hyperparameter is a good predictor of the tested safety score. This means that varying this parameter creates models that perform similar comparing the test loss as shown in Section 6.1.1 but different test performance in the simulator w.r.t. the evaluated safety score. The output for the following scores can be found in A.4.

6.5.1 Distance driven. For the distance driven or the availability we find two main influences using the MLM. Both mixing in DT data and only training with nighttime images decreases the total distance an agent is able to drive on the test routes with the mix percentage being the most influential factor.

6.5.2 Lane infractions. The model confirms our findings from Section 6.2.3 that a higher number of forecast steps increases the number of lane infractions. While adding DT data to the training set alone has no clear effect on the number of lane infractions, autoregressively producing more frames with the DT adds about one lane

[10] Sumio Watanabe. "Asymptotic Equivalence of Bayes Cross Validation and Widely Applicable Information Criterion in Singular Learning Theory"". In: Journal of Physics Conference Series, 233, No.012014. (2010)

infraction per two kilometers when doubling the forecasted frames from 16 to 32.

6.5.3 Collisions per driven distance. Our MLM for the collision score confirms the increase of collisions caused by training with DT data. It further hints to a slight advantage of training only at nighttime although training only at daytime also has a negative coefficient. Using 50% DT data adds roughly one collision per kilometer.

6.5.4 Conclusion. Using MLMs for the analysis of our individual runs instead of only comparing aggregated scores allowed us to have a second perspective on our results. While we could confirm certain findings this way, there were no significant coefficients in our models regarding the brake path shortage or the mean stop-and-go score. While the first part of this fits with our analysis of the aggregated obstacle scores, the clear trend we were able to see with the stop and go score was not visible with our MLMs.

7 THREATS TO VALIDITY

In this section we discuss possible threats to the validity of the approach and the results presented so far. This comprises aspects of generalization, confounders, construction validity, and validity of the drawn conclusions.

7.1 Digital Twin Selection

We chose a deep learning based DT to simulate our driving environment for training. This has multiple reasons: First this system would be able to learn from real world driving images and actions and then act as a DT for the real world. This allows us to make claims that can be in parts applied to driving and learning scenarios of agents that are supposed to drive in the real world. Secondly, we chose a system that worked fundamentally different from CARLA, a rule based simulator, which acted as our real world. We did that under the assumption that every simulator that could be used to train an AV agent for the real world would also have a very different internal logic from the real world. Although these assumptions made sense for our project, this also means that our findings are specific to our choice of DT.

7.2 Agent Selection

Our results are evaluated with only one architecture for AV agents, therefore, generalizing them to other architectures is limited. We chose a simple agent from the perspective of inputs and outputs in order to start from a basic architecture and not specify too much for a very special agent. Furthermore, using cameras as the main sensors becomes more and more popular even in commercial applications. However, other sensors, like LiDAR or Radar, are possible and common to be used by AV agents. While one could try to generalize certain findings like the loss of detail with multiple timesteps generated into the future to other data types, new and other findings are possible and likely once this analysis is extended to other sensor types.

7.3 Architecture

During the design of the setup of how our different components work together we had to make multiple design choices. Some were predetermined by the choice of our components, others we had to decide for ourselves. There are two main decisions we want to cover here:

(1) We started every trajectory generated with the DT by initializing it with one real image from CARLA. This has two reasons: Firstly, while Nvidia demonstrated that DriveGAN is theoretically able to start from arbitrary initialization, we were not able to train the DT to a quality where this would have been possible using the computational resources available to us. Secondly, TCP learns by imitation learning which fundamentally needs some ground truth actions to imitate. Generating completely new trajectories outside of CARLA does not allow us to run the expert agent which means we would need to change the way TCP trains. Since we did not want to come up with a custom approach for an agent but rather evaluate existing ones we chose to not go in this direction.
(2) We only generated new images for given trajectories but reused the measurements like speed or the labels given by the expert agent. The longer the trajectories generated this way the higher the risk of a mismatch of the meta information from the real trajectory and the newly generated images. While a modified version of DriveGAN might be able to generate basic meta information like speed and orientation, generating new expert labels would again be very difficult and imply that the DT would be an expert agent itself since it can predict the next frame and next optimal action given a certain frame.

7.4 Evaluation

We had to pick out certain hyperparameters and specific values for our experiments. While this means that some aspects were out of scope for this project, it also means that we only have a very limited number of specific values for every parameters we tested. While we found some relationships between the hyperparameters and the safety scores, a clearer and more fine-grained analysis would be needed with more different values for every parameter like the forecast steps, or the mix percentage.

8 RELATED WORK

In this project we investigated the phenomenon of underspecification in the domain AV. As described in Section 3.1 and Section 5 our approach is an enhancement of multiple other findings in this domain and the phenomenon of underspecification in machine learning approaches in general. This section highlights a couple of domain-related and domain-independent topics we build our research on.

8.1 Domain-related Topics

Modern approaches in designing AV systems make use of the rapid advances in deep machine learning tools. Multiple architectures for a driving agent's model have been proposed by the research community, each of them tackling and solving different aspects of automated driving.

One of the common challenges in model fitting is to train on diverse and representative data. With respect to safety in AV this

implies that unsafe states and unsafe state handling has to be learned from observations which usually are rare. In their paper "Learning from All Vehicles" (LAV) [2] Chen et al. designed and implemented an approach that leverages on how humans transfer learnings they made from observations only rather than their own experiences. The agent is trained on observations based on the agents ego-perspective and based on partial observations reflecting other vehicles' experiences. In their approach, they use a set of supervisory tasks to learn an intermediate representation of the environment and enable the agent to become independent from its ego-centered viewpoint.

While imitation learning based on RGB-camera data has been explored in multiple research projects, in their paper "TransFuser: Imitation with Transformer-Based Sensor Fusion for Autonomous Driving" (TransFuse) [4] Chitta et al. proposed a data fusion approach based on self-attentions to exploit the complementary advantages of a perspective view (ego view) provided by RGB-based sensor data and a *bird's eye view* provided by LiDAR-based sensor data.

In contrast to the imitation learning strategy many AV approaches follow, "Learning to drive from a world on rails" (WoR) [1] by Chen et al. constructs a tabular value function by discretizing the current state and chosen action of the agent. By doing so they allow the agent to train the agent, represented by a deep learning model, to train on prerecorded trajectories, but instead of learning to predict a certain ideal action, the agent can observe the reward of all possible discretized actions. This way they are able to reduce the amount of training data needed and achieve state of the art performance at the time of the publication.

With respect to testing, different approaches have been developed to focus on finding situations that test on an agents safety and robustness and to automate the generation of test scenarios. In their paper "Doppelgänger Test Generation for Revealing Bugs in Autonomous Driving Software" [11] Huai et al. proposed a test generation approach based on genetic algorithms to discover bug-revealing violations executed by an agent. This implies, that the test scenarios are constructed in a way that (i) it is possible to attribute whether the agent under test is responsible for the outcome of a test case and (ii) in case of a discovered bug, what type of bug it is. In filling the test scenarios with multiple agents which are doppelgangers of the agent under test it is possible to explore a scenario from different perspectives and identify bugs in a model if at least one agent makes a mistake.

8.2 Domain-independent Topics

The challenge of ensuring that AI systems pursue goals that match human values or interests is known as the so-called *Alignment Problem*. In their paper "The Alignment Problem from a Deep Learning Perspective" Ngo, Chan, and Mindermann formulated an argument how different aspects of deep learning approaches make these approaches vulnerable to situationally-aware reward hacking, misaligned internally-represented goals, up to power-seeking behaviors. [16] In the context of safety and robustness for AV, e. g. agents may opt for reward hacking opportunities if safe behavior is rewarded with a bias, as was observed by Chen et al.[1].

The phenomenon of underspecification has been identified as the underlying cause for poor performance or unexpected behaviours of machine learning models when deployed in real world domains by D'Amour et al. in their work "Underspecification Presents Challenges for Credibility in Modern Machine Learning" [7]. They define a solution to a problem as underspecified if multiple distinct solutions that solve the problem equivalently exist and make two main claims about the role of underspecification in modern machine learning. Their research is independent from the domain of AV and it formulates the baseline for our work.

9 CONCLUSION AND FUTURE WORK

In this section we answer our research questions and summarize the findings of our work. Afterwards we give an outlook on potential future directions.

9.1 Conclusion

We were able to clearly show underspecification when training AV agents in a simulation. While certain parameter configurations slightly decrease the test loss, they can have catastrophic consequences in our real world environment. In simple scenarios like straight driving, agents that trained using a mix of real and DT data perform equally as well as the agents only trained on CARLA data. While the route completion is slightly lower for the agents trained on the DT, the safety scores are even slightly better in some cases. Even some of the agents trained with only DT data perform acceptable in those scenarios. Based on this, we can answer our first research question with yes. However once we test on more complex scenarios like curved or intersection driving or the complete evaluation routes, agents using the DT score worse on one or multiple evaluation metrices. In general we found the following patterns:

- Increasing the percentage of DT data results in more collisions and lane infractions per kilometer while decreasing the distance driven
- Increasing the forecast steps in the DT leads to more lane infractions
- Decreasing the variation in the time of day in the training set decreases the distance driven with only nighttime training decreasing it the most
- Decreasing the variation in the time of day in the training set also affects validation loss and CARLA Leaderboard score, but depending on the configuration of the other parameters, either only daytime or only nighttime training is better than the other option

This means that we found patterns in how certain parameters influence underspecification. However it is important to highlight that the percentage to which the DT was used was the main determining factor for underspecification in our evaluation. In order to focus more on the other parameters we suggest multiple next steps in Section 9.2 and Section 7.

9.2 Future Work

9.2.1 Underspecification. In order to analyze underspecification as a general problem in AV, our approach needs to be expanded to different agents and training approaches. One could either come

up with variations of the architecture for different agents and their requirements or try to design a new more powerful DT that is able to simulate the driving environment in a more global way instead of only simulating local sensor readings. This goal of this analysis would be to find common components in the training setup of the different agents which are mainly responsible for underspecification. These insights can be used to estimate hyperparameters for new agents to minimize underspecification and aim for safe driving on deployment.

9.2.2 Improved Digital Twin. We found that the amount of DT data used was the main determining factor for the amount of underspecification. In order to find out more about other parameters it might be a goal to build a DT that is even better at simulating the real world. While more training time and training data could already produce an improved model in our setup, there are multiple ideas to improve the DT conceptually:

- In order to take advantage of the decomposition into spatial and theme vectors more variety in the weather and content conditions might help the model to learn better and more interpretable feature vectors.
- We found that the loss of details in DT images is a key factor that lets agents trained on them perform poorly in the real test environment. Using a pretrained agent on real data and a method similar to the IG visualization as a mask for a pixelwise loss during training could help to guide the DT towards producing more details important for the driving agent.
- Instead of trying to take advantage of the decomposition of the latent vector one could also go into the direction of simplifying the latent vector in order to maximize image quality. Since the decomposition of the latent embedding is an additional constraint on the DT that is not necessary in our analysis, removing the constraint could help the model to focus on producing high quality images.
- Our approach focused on building a DT that simulates the camera sensors for a specific agent. In order to build a DT that can simulate an environment for arbitrary agents, building a new DT that creates its internal representation based on global information like BEV maps and 3D pointclouds of the scenery rather than local sensor readings could help building a general DT of the driving environment. From this multiple branches could be used to extract local sensor readings. This could also be easily extended to multiple agents.

9.2.3 Contrastive Learning. In contrastive learning, expanding the training dataset by augmenting existing data is a valid approach [3]. Alternatively, using a DT as proposed in our research offers a different approach to data augmentation. To implement this idea, images should be generated from CARLA, with corresponding actions serving as labels. The DT can subsequently produce multiple images with distinct theme vectors while maintaining the same spatial vector. These images may serve as positives for additionally training an AV agent. However, the impact of this approach on agent performance remains uncertain so far.

9.2.4 Improved Expert Agent. During our experiments we observed multiple opportunities for improvement in the IL setting itself, which is used for training by many agents. During data collection the expert agent does not make any major mistakes. Since all cars are steered by an instance of the expert agent, this means that the training data neither contains situations in which the agent needs to recover from an unwanted situation nor situations where other cars behave in an unsafe manner and the agent needs to deal with the other cars. While this includes major problems like crashed cars, also simple behavior like taking over another car that stopped for some reason is included in this. By training the expert to perform complicated driving behaviors with other non expert drivers agents learning from this new expert would be more robust.

9.2.5 Causality. Using *Reinforcement Learning (RL)* within a car simulator presents a valuable opportunity for examining causality in the context of causal RL. The simulator serves as an ideal structural causal model, as all its elements are predetermined by the underlying physics engine. The sensor data perceived by the agent can be regarded as the known variables. When an action is taken by the agent, the environment changes, leading to observable outcomes. As a first step, a collection of scenarios and observations must be established, similar to the scenarios and observations used in the computation of our safety scores. Causal discovery and causal inference are particularly worth looking into [20].

9.2.6 Multiple Agents. Expanding our scenario to multiple agents involves significant complexity. As common in multi agent RL, stationarity is not given anymore, as each agent's actions impact the environment for others [21]. Challenges include:

- Achieving optimal traffic flow through collaborative actions of multiple cars, as in fully cooperative multi agent RL [21].
- Exploring centralization and decentralization approaches for negotiating right of way at intersections, including considerations of introducing a traffic manager instead of peer-to-peer communication.
- Addressing equilibrium outcomes, where traffic flow remains the same but with varying agent priorities.
- Managing the storage and transmission of information. Determining the relevant information to share, optimizing communication, and enhancing robustness against malicious actors.
- Scalability of the solutions.

A APPENDIX

A.1 General Process

The general process of this project consists of the following steps:

(1) Training a DT to simulate a real-world driving environment
(2) Training an agent based on observations either coming from the "real world" or the DT
(3) Evaluate the agents performance based on availability and safety related aspects

Figure 9 visualises this process.

A.2 Engineering Summary

From an engineering perspective we performed the following steps:

- Configured, ordered and set up a new workstation for our experiments

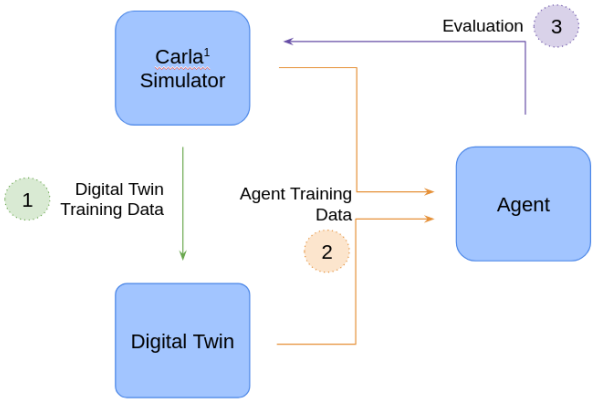

Figure 9: The general process includes (1) train data generation, (2) agent training (3) agent evaluation.

Figure 10: Idea Blossom that captures our notions of safety in AV.

- Setup the CARLA simulator and collected driving data using the requirements from TCP
- Expanded the code from DriveGAN to support the custom aspect ratio needed for TCP, use the action format of TCP, built a custom dataloader and a new script to generate data in the right format for TCP. This way it could work as our DT.
- Trained the DT in two stages
- Trained multiple instances of the TCP agent, including many test instances and 15 final agents for evaluation
- Built custom evaluation script to measure agent performance w.r.t test datasets
- Explored evaluation scenarios and manually extracted sub test routes
- Designed, implemented, tested and refined safety scores and combined them with the evaluation code for the CARLA Leaderboard
- Compare the underlying statistical models of the different agents w.r.t. to model
- Analyzed the relationship between different hyperparameters and safety scores using MLMs which we compared agents w.r.t. to model complexity, the risk overfitting and consequently the risk of underspecification

A.3 Safety Scores

For us, it was important to enhance the human-centered perspective of safety in AV. Choosing the people sitting within the vehicle as a target group, we started with the question "What does safety mean?" and collected possible safety scores within an *idea blossom* as shown in Figure 10. Table 10 summarizes the different ides, how to measure them and whether or not we implemented them.

In the rest of this section the implemented scores are detailed. For all scores the following notations apply:

- E donates the total number of the different experiment types
- S donates the total number of scenarios for a given experiment type
- R donates the total number of repetitions for a given scenario
- *length* donates the total length of the route
- *completion* donates the percentage of achieved route completion

A.3.1 Collision-based Scores. We implemented the following scores:

- Ratio of collisions occurred per driven distance unit
- Ratio of the weighted collision impulses per occurred collision, weighted by the total number of occurred collisions

Equation (2) show the calculation of collisions occurred per driven distance unit. C donates the total number of collisions.

$$\frac{\sum_{i=1}^{E}\sum_{j=0}^{S}\sum_{k=0}^{R}\left(C_{i,j,k}\cdot \text{length}_{i,j,k}\cdot \text{completion}_{i,j,k}\right)}{\sum_{x=1}^{E}\sum_{y=0}^{S}\sum_{z=0}^{R}\left(\text{length}_{x,y,z}\cdot \text{completion}_{x,y,z}\right)} \tag{2}$$

Equation (3) show the calculation of weighted collision impulses per occurred collisions. C donates the total number of collisions and impulse donates the impulse intensity. Depending on whether we were interested in the min, max, or mean collision impulses, we used the measurements for the min, max, or mean impulses respectively.

$$\frac{\sum_{i=1}^{E}\sum_{j=0}^{S}\sum_{k=0}^{R}\left(C_{i,j,k}\cdot \text{impulse}_{i,j,k}\right)}{\sum_{x=1}^{E}\sum_{y=0}^{S}\sum_{z=0}^{R}C_{x,y,z}} \tag{3}$$

A.3.2 Traffic Rules-based. We implemented the following scores:

- Lane infractions per driven distance unit
- Brake path shortage per obstacle

Equation (4) show the calculation of lane infractions occurred per driven distance unit. L donates the total number of infractions.

$$\frac{\sum_{i=1}^{E}\sum_{j=0}^{S}\sum_{k=0}^{R}\left(L_{i,j,k}\cdot \text{length}_{i,j,k}\cdot \text{completion}_{i,j,k}\right)}{\sum_{x=1}^{E}\sum_{y=0}^{S}\sum_{z=0}^{R}\left(\text{length}_{x,y,z}\cdot \text{completion}_{x,y,z}\right)} \tag{4}$$

Equation (5) show the calculation of brake path shortages occurred per driven distance unit. O donates the total number of

Idea	How to measure	Type of score	Implemented	Comment
No one get's hurt	Count agents collision with other agents, dynamic or stationary objects; count collisions of living beings inside the vehicle with vehicle internals; measure collision intensity	Collision-based	Partially	Only collisions of agent with the environment implemented
No erratic movements	acceleration, angular velocity, orientation of the wheels physical force on human driver; compare with smooth trajectory; pattern definitions	Subjective Feeling-based	Yes	
Subjective feeling	Survey	Subjective Feeling-based	No	
Detect mechanical defects (engine, controls, brake system)	Regular feedback loop to agent, detect certain patterns in data		No	
Be adaptive to different set of traffic rules (e. g. speed, geographic region)	Context-based meta learning (e. g. CARE)	Traffic Rules-based	Partially	lane infraction-based scores implemented
Handle the unexpected properly (unexpected obstacles, emergency vehicles, inattentive pedestrians)	Weight based or frequency in train data; count and penalize collisions	(Collision-based)	Partially	collision-based aspects implemented
Safety by learning in a simulation			no	
Human interpretability	Modular score calculation		no	
Giving safety guarantees	Safety integrity levels		no	

Table 10: Full set of safety scores, how to measure them, and whether we implemented them.

obstacles such a shortage occurred with.

$$\frac{\sum_{i=1}^{E}\sum_{j=0}^{S}\sum_{k=0}^{R}\left(O_{i,j,k}\cdot \text{length}_{i,j,k}\cdot \text{completion}_{i,j,k}\right)}{\sum_{x=1}^{E}\sum_{y=0}^{S}\sum_{z=0}^{R}\left(\text{length}_{x,y,z}\cdot \text{completion}_{x,y,z}\right)} \tag{5}$$

A.3.3 Subjective Feeling-based Scores. We implemented the following scores:

- Ratio of the weighted stop-and-go movements per driven distance unit

Equation (6) show the calculation of weighted stop-and-go movements occurred per driven distance unit. A stop-and-go movement is a spike in acceleration changes during driving. M donates the total number of spikes. Parameters for the algorithm to count these spikes are: peak height, distance between peaks, and peak prominence.

$$\frac{\sum_{i=1}^{E}\sum_{j=0}^{S}\sum_{k=0}^{R}\left(M_{i,j,k}\cdot \text{length}_{i,j,k}\cdot \text{completion}_{i,j,k}\right)}{\sum_{x=1}^{E}\sum_{y=0}^{S}\sum_{z=0}^{R}\left(\text{length}_{x,y,z}\cdot \text{completion}_{x,y,z}\right)} \tag{6}$$

During evaluating we counted small peaks for the stop-and-go behaviour and big peaks as emergency braking.

A.4 Model Comparison

Output of R for the chosen MLMs. Non zero centered confidence intervals are highlighted.

```
Population-Level Effects:
              Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
Intercept         0.01      0.01    -0.02     0.04 1.00     2157     1665
forecast          0.01      0.02    -0.04     0.06 1.01      936      612
mix_percentage    0.02      0.02    -0.03     0.07 1.00     1184      934
timeday          -0.01      0.02    -0.04     0.02 1.00     1408     1596
timenight        -0.02      0.02    -0.05     0.01 1.00     1198     1316
```

Figure 11: Model Comparison: Analysis of the influence of the different input parameters on the number of collisions per distance driven using an MLM in R.

```
Population-Level Effects:
              Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
Intercept         0.02      0.01    -0.01     0.04 1.00     3418     2739
forecast          0.04      0.04    -0.03     0.11 1.00     2098     1968
mix_percentage    0.00      0.04    -0.07     0.07 1.00     2110     1888
```

Figure 12: Model Comparison: Analysis of the influence of the different input parameters on the number of lane infractions per distance driven using an MLM in R.

REFERENCES

[1] Dian Chen, Vladlen Koltun, and Philipp Krähenbühl. 2021. Learning to drive from a world on rails. In *Proceedings of the 2021 IEEE/CVF International Conference on Computer Vision*. ICCV, Montreal, QC, Canada, 15590–15599.

Population-Level Effects:

	Estimate	Est.Error	l-95% CI	u-95% CI	Rhat	Bulk_ESS	Tail_ESS
Intercept	961.59	64.31	833.34	1089.78	1.00	1914	1348
forecast	67.99	110.52	-163.47	290.94	1.00	1155	1162
mix_percentage	-804.02	115.45	-1024.05	-569.46	1.00	1116	963
timeday	-8.80	81.14	-181.18	156.38	1.00	1056	861
timenight	-87.03	80.21	-245.93	59.71	1.00	1338	677

Figure 13: Model Comparison: Analysis of the influence of the different input parameters on the distance driven using an MLM in R.

Population-Level Effects:

	Estimate	Est.Error	l-95% CI	u-95% CI	Rhat	Bulk_ESS	Tail_ESS
Intercept	0.53	0.02	0.49	0.57	1.00	2284	1476
forecast	0.02	0.04	-0.05	0.10	1.01	675	440
mix_percentage	0.24	0.04	0.16	0.32	1.00	698	822

Figure 14: Model Comparison: Analysis of the influence of the different input parameters on the stop and go score using an MLM in R.

[2] Dian Chen and Philipp Krähenbühl. 2022. Learning from all vehicles. In *Proceedings of the 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. CVPR, New Orleans, LA, USA, 17222–17231.

[3] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A Simple Framework for Contrastive Learning of Visual Representations. arXiv:2002.05709 [cs.LG]

[4] Kashyap Chitta, Aditya Prakash, Bernhard Jaeger, Zehao Yu, Katrin Renz, and Andreas Geiger. 2022. TransFuser: Imitation with Transformer-Based Sensor Fusion for Autonomous Driving. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 44 (2022), 1–18.

[5] Elliot Creager, Jörn-Henrik Jacobsen, and Richard Zemel. 2021. Environment Inference for Invariant Learning. *Proceedings of the 38th International Conference on Machine Learning* 139 (2021), 2189–2200.

[6] Elliot Creager, Jörn-Henrik Jacobsen, and Richard Zemel. 2023. Diversify and Disambiguate: out-of-distribution robustness via disagreement. In *Proceedings of the 11th International Conference on Learning Representations*. ICLR, Kigali, Rwanda, 19 pages.

[7] Alexander D'Amour, Katherine Heller, Dan Moldovan, Ben Adlam, Babak Alipanahi, Alex Beutel, Christina Chen, Jonathan Deaton, Jacob Eisenstein, Matthew Hoffman, Farhad Hormozdiari, Neil Houlsby, Shaobo Hou, Ghassen Jerfel, Alan Karthikesalingam, Mario Lucic, Yian Ma, Cory McLean, Diana Mincu, and D. Sculley. 2020. Underspecification Presents Challenges for Credibility in Modern Machine Learning. *The Journal of Machine Learning Research* 23 (11 2020), 10237—-10297.

[8] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. 2017. CARLA: An Open Urban Driving Simulator. In *Proceedings of the 1st Annual Conference on Robot Learning*. CoRL, Mountain View, California, USA, 1–16.

[9] Michael Grieves. 2014. Digital Twin: Manufacturing Excellence through Virtual Factory Replication. , 7 pages.

[10] Bin He and Kai-Jian Bai. 2020. Digital twin-based sustainable intelligent manufacturing: a review. *Advances in Manufacturing* 9 (05 2020), 1–21.

[11] Yuqi Huai, Yuntianyi Chen, Sumaya Almanee, Tuan Ngo, Xiang Liao, Ziwen Wan, Qi Alfred Chen, and Joshua Garcia. 2023. Doppelgänger Test Generation for Revealing Bugs in Autonomous Driving Software. In *2023 IEEE/ACM 45th International Conference on Software Engineering*. ICSE, Melbourne, Australia, 2591–2603.

[12] SAE International. 2021. Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles J3016_202104. , 30–32 pages.

[13] Seung Wook Kim, Jonah Philion, Antonio Torralba, and Sanja Fidler. 2021. Drivegan: Towards a controllable high-quality neural simulation. In *Proceedings of the 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. CVPR, Nashville, TN, USA, 5820–5829.

[14] Evan Zheran Liu, Behzad Haghgoo, Annie S. Chen, Aditi Raghunathan, Pang Wei Koh, Shiori Sagawa, Percy Liang, and Chelsea Finn. 2021. Just Train Twice: Improving Group Robustness without Training Group Information. *Proceedings of the 38th International Conference on Machine Learning* 139 (07 2021), 6781–6792.

[15] Junhyun Nam, Hyuntak Cha, Sungsoo Ahn, Jaeho Lee, and Jinwoo Shin. 2020. Learning from Failure: Training Debiased Classifier from Biased Classifier. In *Proceedings of the 34th Conference on Neural Information Processing Systems*. NeurIPS, Vancouver, BC, Canada, 20673—-20684.

[16] Richard Ngo. 2022. The alignment problem from a deep learning perspective. , 21 pages.

[17] Shiori Sagawa, Pang Wei Koh, Tatsunori B. Hashimoto, and Percy Liang. 2020. Distributionally Robust Neural Networks for Group Shifts: On the Importance of Regularization for Worst-Case Generalization. In *Proceedings of the 8th International Conference on Learning Representations*. ICLR, Addis Ababa, Ethiopia, 18 pages.

[18] Aki Vehtari, Andrew Gelman, and Jonah Gabry. 2016. Practical Bayesian model evaluation using leave-one-out cross-validation and WAIC. , 28 pages.

[19] Penghao Wu, Xiaosong Jia, Li Chen, Junchi Yan, Hongyang Li, and Yu Qiao. 2022. Trajectory-guided control prediction for end-to-end autonomous driving: A simple yet strong baseline. In *Proceedings of the 36th Conference on Neural Information Processing Systems*. NeurIPS, New Orleans, LA, USA, 13 pages.

[20] Yan Zeng, Ruichu Cai, Fuchun Sun, Libo Huang, and Zhifeng Hao. 2023. A Survey on Causal Reinforcement Learning. arXiv:2302.05209 [cs.AI]

[21] Kaiqing Zhang, Zhuoran Yang, and Tamer Başar. 2021. Multi-Agent Reinforcement Learning: A Selective Overview of Theories and Algorithms. arXiv:1911.10635 [cs.LG]