Race Car Driver Model: Improved Generalization for Behavioral Cloning with Image-Based Feature Sets

Renée Schwietert

Submitted in partial fulfillment of the requirements for the degree of Master of Science in Robotics at Delft University of Technology, to be defended publicly on the 26th of November, 2024

Supervisors:
Arkady Zgonnikov
Jens Kober

External Supervisors:
Siwei Ju
Peter van Vliet

Graduation committee:
Arkady Zgonnikov
Jens Kober
Julian Kooij

TU Delft

Race Car Driver Model: Improved Generalization for Behavioral Cloning with Image-Based Feature Sets

Renée Schwietert

***Abstract*—This thesis explores enhancing track generalization in motorsport driver models through image-based feature sets, drawing inspiration from autonomous driving applications in urban settings. Traditional motorsport models often rely on numeric features, which excel on known tracks but face limitations when adapting to new, unseen environments. To address this, I introduce a CNN-based model that integrates bird's-eye-view images with vehicle states and path-planning data, allowing a more holistic perception of track layouts and surroundings. Through open-loop evaluations on unseen tracks, the proposed model demonstrates superior generalization, achieving significantly lower RMSE compared to boundary point-based models, with improvements observed across steering, braking, and acceleration actions. Additionally, I apply novelty detection using Mahalanobis Distance to isolate Out-of-Distribution(OoD) scenarios, providing a precise measure of the generalization gap. This work establishes a baseline for image feature design in motorsport driver modeling, emphasizing the role of spatial and contextual information in achieving adaptable and high-performance autonomous racing agents.**

I. Introduction

Motorsport racing captivates audiences worldwide, drawing attention not only for its thrilling speed and precision but also for its broader impact on technological advancements and engineering excellence. Beyond the spectacle, motorsports serve as a proving ground for pioneering innovations in vehicle performance, safety, and sustainability.

Developing high-performance vehicles and refining driver skills require substantial resources. Physical trials, though effective, are costly and time-consuming. As a solution, simulations have become central to motorsports, enabling engineers to explore a range of scenarios and adjustments in a controlled environment. While simulations often include human drivers interacting directly within the virtual environment, these driver-in-the-loop setups still require significant time and resources. Driver models offer a more efficient alternative by replicating human-like decision-making and driving behavior, allowing simulations to closely mirror real-world scenarios without the need for constant human involvement.

In motorsports, the focus in creating and evaluating these models has been on optimizing performance on individual tracks, with limited attention given to generalization [Fuchs et al., 2021], [Braghin et al., 2008]. Track generalization—the ability of a model to perform well across a variety of tracks—is a key challenge that has not been fully addressed in motorsport simulations. The concept of track generalization is critical in motorsport applications, where small differences in track layout can have a substantial impact on the vehicle's optimal path and overall performance [Gustafsson, 2008]. Achieving robust generalization is not only crucial for improving performance but also significantly enhances the efficiency of training the models. In practice, it is often impractical or inefficient to train or explore on all available tracks, especially when new tracks are introduced. An effective generalization strategy can relieve this, allowing models to perform well on unseen tracks without requiring extensive retraining. This capability is particularly valuable in scenarios where no prior data from a new track is available, enabling the agent to drive and navigate the track autonomously and effectively from the outset.

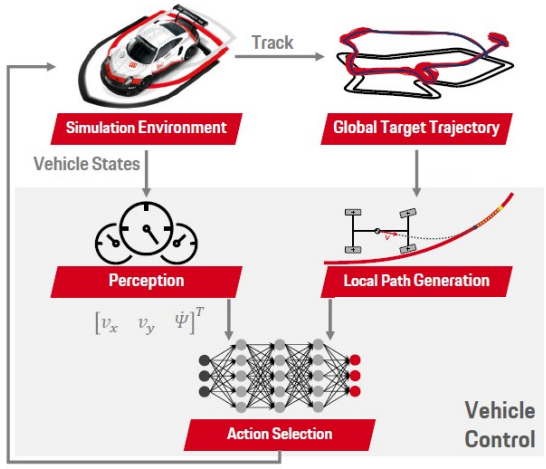

Fig. 1. Behavioral Cloning race driver model from [Löckel, 2022a] incorporating numerical perception features, vehicle state features and path planning features from the expert trajectory to select actions with a feedforward neural network.

An approach to race car driver modeling that incorporates human factors, Behavioral Cloning (BC) [Löckel, 2022a], [Löckel, 2022b] (Figure 1), has shown to be able to generalize to unknown tracks to some extent. However, in the current method, adaptation needs to be done through post processing and it is limited in cases that it can handle due to the covariate shift [Ross and Bagnell, 2010]. To further enhance BC in race car driver modeling to address the challenge of generalization, most approaches in related fields revolve around diversifying data or applying specific training methods, including: **Procedural Track Generation for Training** [Behrens, 2020] which diversifies the training set by exposing the model to algorithmically created synthetic tracks [Li et al., 2020]. While this method creates diverse scenarios, the synthetic tracks often lack the nuanced dynamics and environmental realism of real-world tracks. This disconnect can lead to models that overfit to procedural patterns (systematic biases) rather than generalizable principles of driving. Moreover, the

computational expense of generating high-fidelity tracks can outweigh the benefits for real-time applications, especially in resource-constrained settings. **Curriculum Learning** introduces progressively complex tracks during training to help the model build foundational skills before tackling more difficult scenarios [Bengio et al., 2009]. However, this method struggles to prepare models for entirely novel scenarios outside the curriculum. **Data Augmentation** applies transformations such as flipping, scaling, or noise injection to expand the training data [Volpi et al., 2018]. Data augmentation relies on existing data as a foundation. While transformations add diversity, they do not create fundamentally new scenarios. This means the model may still struggle with completely novel track layouts or conditions, which are common in motorsport.

Models used in motorsport simulations [Ganesh et al., 2016], [Remonda et al., 2022], predominantly rely on numeric perception features, such as rangefinders and other distance measurements in combination with vehicle state and path-planning features. **Multi-Modal Input Features**, particularly the combining of image-based representations with different inputs, have shown to enhance generalizability in urban driving applications [Xiao et al., 2020], [Hwang et al., 2024]. By capturing a holistic view of the track and combining this with other inputs, image-based features can enable driver models to learn patterns directly from the visual context. Despite the success of generalization with multi-modal image-based methods in urban driving, the motorsport domain has yet to fully embrace this approach.

In this thesis, I aim to study the effects of image-based feature sets on track generalization in motorsport driver modeling, building on the successful approaches used in urban autonomous driving. By leveraging CNNs for feature extraction and combining them with traditional numeric inputs, I find that this model not only performs well on known tracks but also generalizes effectively to new, unseen tracks. This approach provides a more comprehensive understanding of the track environment, enabling the model to make more informed decisions based on the visual context of the track, rather than relying solely on numeric features. Ultimately, this work contributes to driver modeling in motorsport simulations by demonstrating the benefits of image-based feature sets for enhancing track generalization.

II. Methodology

My objective is to study the effects on track generalization of the integration of image-based features in a driver model. The method involves creating an image feature-set and a feature extractor, such that spatial and contextual information about the surroundings can be extracted, learned from and transferred to enable high performance on an unknown track. In this section I introduce a previous driver model perception method that serves as a comparison to the new method; explain the design methodology of the images; show the integration into the network architecture and present the evaluation method.

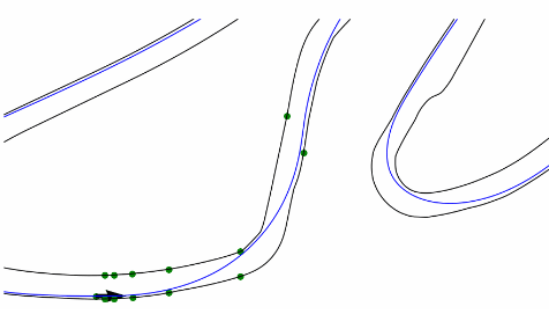

Fig. 2. Boundary point perception features visualized on race track. Points mark [5, 10, 20, 40, 80, 160, 320, 640] distances on left and right track boundary. Arrow indicated the heading angle of the car. The raceline of the expert is visualized in blue.

A. Previous Driver Modeling Method

This works builds on previous research done regarding a Behavioural Cloning agent used as initialisation for a Reinforcement Learning agent [Ju et al., 2023]. The model consists of a feed-forward neural network mapping a feature set to actions.

1) Feature set: The feature set used is made up of three types: vehicle state features, path-planning features and perception features:

$$[v,\ a_x,\ a_y,\ \beta_F,\ \beta_R,\ r_F,\ r_R,\ \alpha_{\text{position}},\ c_{\text{poly}},\ \alpha_{\text{offset}},\ d_{\text{offset}}]$$

Vehicle states provide a comprehensive representation of the vehicle's dynamics, allowing for precise modeling of the forces and motions that affect its stability and handling.

- The **vehicle states** include absolute velocity (v), longitudinal acceleration (a_x), lateral acceleration (a_y), and the average slip ratios (r_F, r_R) and slip angles (β_F, β_R) for the front and rear tires.
- The **path planning features** consist of the coefficients of a polynomial (c_{poly}) that describe the local path, along with angular and distance offsets (α_{offset}, d_{offset}) calculated within a specified preview time. These are derived from the reference trajectory for each lap.
- **Boundary Points** (BPs) are used as perception features. BPs are typically defined as 32 relative distances from the car's center of gravity to the track's boundary on each side. They are defined as the x, y distances [5, 10, 20, 40, 80, 160, 320, 640] on the left and right boundary.

The BPs provide essential information about the car's spatial position within the track in a simplified manner, representing only a few critical points rather than the entire track. This makes BPs highly efficient for the model by reducing computational complexity. However, the limitation of this approach is that it sacrifices finer-grained details about the track's curvature or environmental context. Next to this, BPs may face challenges in generalizing across tracks with varying layouts, especially in regions with sharp curves or significant changes in track width, potentially reducing the model's ability to adapt to new tracks.

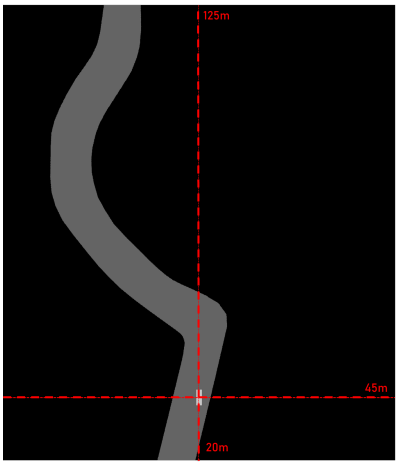

Fig. 3. Final image feature design with 125m forward visibility, 45m horizontal visibility, 20 m backward visibility and 2.5 pixels per meter resolution. Designed to specifically consider the trade-off between speed and accuracy. The car is shown in white.

2) Actions: The action $\mathbf{a}_t$ consists of three components: braking, accelerating, and steering, represented as $\mathbf{a}_t = [g_t, b_t, \delta_t]^T$, where g_t refers to the accelerator pedal actuation, b_t refers to the brake pedal actuation, and δ_t represents the steering wheel angle.

B. Image Design Methodology

Race car drivers' decision-making is influenced by various sensory inputs, including visual, auditory, and haptic feedback. Visual perception plays a central role, providing crucial information about the track layout, vehicle position and upcoming turns. This visual data allows drivers to anticipate necessary maneuvers, such as braking, steering, and accelerating, in real-time. In addition to sensory inputs, drivers rely on prior knowledge of the track and vehicle characteristics, as well as gathered knowledge while driving, like the current grip level and vehicle performance. In my model, the integration of images brings an overall spatial overview of the track ahead, next to and behind the agent. These images allow the model to capture spatial and contextual information, improving decision-making by giving a more complete representation of the driving environment.

The image feature is designed to capture the information most relevant for the agent at the current timestep. This information consists of the current scenario, relevant upcoming scenarios and relevant previous scenarios. These can be broken down into 4 design parameters (forward visibility, horizontal visibility, backward visibility and resolution) with their respective criteria.

Forward Visibility

- Braking distance: the agent needs at least the braking distance to the next turn in view to make correct decisions on where to brake
- Turn anticipation: for the sharpness of turns and upcoming track changes (e.g. whether another turn follows or a straight)
- Gradual state changes: seeing further ahead may mean encountering more similar states over time, reducing abrupt differences

Horizontal Visibility

- Track boundaries: the agent needs to see both edges of the track to maintain proper positioning within the lane and avoid going off the track.
- Lateral adjustments: with a wider view, the agent can make smoother, more informed lateral adjustments, especially during cornering, where understanding track width is crucial for precision.
- Avoiding mistakes: seeing the full width of the track reduces the risk of mistakes like understeering or oversteering, as the agent is always aware of its boundaries.

Backward Visibility

- Previous trajectory: the agent needs to see a portion of where it came from to help it maintain continuity in decision-making, particularly in recovery scenarios (e.g., after a corner or a mistake).
- Acceleration decisions: by seeing the immediate past, the agent can better time its acceleration out of corners, understanding whether it has fully exited a turn or if it's still within the corner's trajectory.
- Recovery from mistakes: the view behind the agent helps it adjust after unexpected events (e.g., minor oversteer) by giving context to recent changes in its state.

Resolution

The resolution of the image affects the level of detail that the agent can perceive and process.

- Track detail precision: higher resolution allows the agent to discern fine details of the track, such as exact curvature, or subtle changes in track width, which are essential for making precise control decisions.
- Feature detection: a higher resolution aids in better detection of track boundaries and curvature.
- Relevance of detail: the level of resolution should capture only the most relevant features (e.g., curvature, track boundaries) without overloading the agent with unnecessary details that don't significantly impact decision-making.
- Temporal coherence: a consistent resolution helps the agent maintain coherent state transitions, reducing confusion that may arise from varying levels of detail in successive frames.

These four parameters can be incorporated into an image from a birds-eye-view perspective. Next to the required visibility needed in the image, the criteria in order to ensure the practical implementation of the feature set needs to be considered. The simulation-time as well as the training time need to stay within reasonable bounds in order for the image feature set to be a valid solution for this use-case. Multiple methods of generating birds-eye-view images were created

and tested. As explained in Appendix A and B, the final design the result of the best trade-off for speed, simplicity, informativity and accuracy (on train and test data). Initial generalization trials were done to determine the values of the design parameters and alternative methods (involving active zooming and altering resolution to velocity to match human peripheral vision) were created and evaluated. The final design (Figure 3) is a grey-scale image oriented at the heading angle of the car. The car is placed with 125m forward visibility, 45m horizontal visibility (on either side), 20m backward visibility and 2.5 pixels per meter resolution. This results in an input size of (362, 225, 1).

C. Model Architecture

The architecture of the CNN model, as shown in Figure 4, is designed to process a combination of image features from the track and state/path features. The CNN was chosen based on tests done with AlexNet [Krizhevsky et al., 2017] as used in [Djuric et al., 2020], Nature CNN [Mnih et al., 2015] (for ease of future implementation in the RL agent described in Section II-A), and Nature CNN with an additional layer. Architecture comparisons are provided in Appendix D. Further hyperparameter tuning as well as testing resulted in the best generalization performance with the following model. The model begins with a series of convolutional layers, which are responsible for extracting spatial features from the images. Each convolutional layer is followed by a max-pooling layer to reduce spatial dimensionality while preserving the most important features. The architecture includes three convolutional layers: the first layer applies 32 filters with a kernel size of 5x5, followed by max-pooling with a 2x2 kernel. The second and third convolutional layers each use 64 filters with smaller kernel sizes (3x3), and max-pooling is applied after the second layer. Batch normalization is applied after each convolutional layer to improve training stability, and the value for L2 regularization (tuned to 0.005) is used to prevent overfitting. ReLU activation is used for all layers. After the convolutional blocks, the image features are flattened and concatenated with the state and path planning features before being passed into two fully connected (dense) layers, each with 64 units and no L2 regularization (the use of L2 regularization here was shown to have no generalization benefits during tuning). The model is trained over 150 epochs with a learning rate of 0.00002.

The CNN architecture is detailed in Table I, which lists the layers, filter sizes, kernel sizes, strides, regularization techniques, and activation functions used throughout the network.

D. Evaluation Method: Open-Loop Evaluation of Generalization and Novelty Detection on Unseen Tracks

The primary objective of this thesis is to analyze the generalization performance of an agent with image-based perception, by assessing the gap between training loss and test loss on new tracks (the generalization gap). A key challenge to properly analyze this is to understand when the agent is truly generalizing and when the agent is encountering segments of a new track that theoretically fall within the training distribution.

1) Open-Loop Evaluation: Initially, I evaluate the agent's performance using an open-loop evaluation. This method involves running the model in a simulated environment where the agent is exposed to predefined expert lap data and expected to make predictions at each timestep, without influencing future states of the simulation. This allows for a controlled analysis of the model's decision-making at each point on the track.

Open-loop evaluation is beneficial for focusing on prediction accuracy without the complexity of closed-loop interactions, where the agent's predictions influence the next state of the environment. The goal of this phase is to assess how well the agent can generalize based on unseen track layouts by examining the prediction errors throughout the track.

The open-loop evaluation is particularly valuable for benchmarking the proposed approach against the previous method (described in Section II-A), where prediction accuracy at each point on the track is compared directly to the old approach. This provides insight into whether the new method performs better or worse in terms of error reduction.

2) Novelty Detection for Further Evaluation: While an open-loop comparison of the full lap on a new track can provide decent overall insights, it does not provide detailed insights into specific regions of the track where the methods truly encounter a new scenario. A straight segment on a new track may, in theory, be familiar to the agent, for example after having encountered straight segments in the training data. To further evaluate the methods, I introduce novelty detection to identify regions of the track that are novel to the agents, to reach a more precise calculation of the generalization gap.

I employ Mahalanobis Distance (MD) [McLachlan, 1999] to quantify the novelty of track segments.

Given the high-dimensional feature embeddings $\mathbf{z}_i \in \mathbb{R}^{2304}$ extracted from the outputs of agent's feature extractor (CNN), I reduce the dimensionality to m principal components using Principal Component Analysis (PCA) [Wold et al., 1987] to ensure computational efficiency and stability.

TABLE I
ARCHITECTURE OF THE CNN MODEL. BATCH NORMALIZATION IS APPLIED AFTER EACH CONVOLUTIONAL LAYER, AND PADDING WAS USED FOR THE CONVOLUTIONAL LAYERS. THE MODEL WAS TRAINED FOR 150 EPOCHS WITH A LEARNING RATE OF 0.00002.

Layer Type	Filter/Units	Kernel Size	Strides	Regularization	Activation
Conv2D	32	(5, 5)	(4, 4)	L2 (0.005)	ReLU
MaxPooling2D	-	(2, 2)	(2, 2)	-	-
Conv2D	64	(3, 3)	(2, 2)	L2 (0.005)	ReLU
MaxPooling2D	-	(2, 2)	(2, 2)	-	-
Conv2D	64	(3, 3)	(1, 1)	L2 (0.005)	ReLU
Dense Layer 1	64	-	-	L2 -	ReLU
Dense Layer 2	64	-	-	-	ReLU

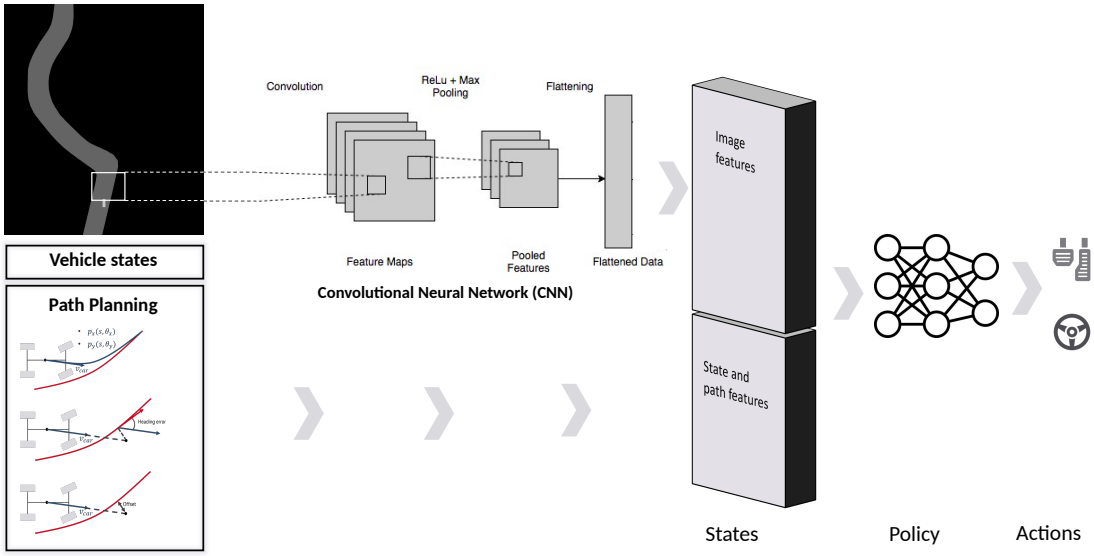

Fig. 4. Model architecture with CNN feature extractor for images as perception features. Extracted features are concatenated with vehicle state and path-planning features and fed into dense layers.

The reduced feature embeddings $\mathbf{z}_i \in \mathbb{R}^m$, in combination with the state and path-planning features, are then used to compute Mahalanobis Distance as a measure of how far a given feature vector is from the distribution of the training data.

The Mahalanobis Distance for a given feature embedding $\mathbf{z}_i$ with respect to the distribution of the training embeddings is calculated as:

$$\mathrm{MD}(\mathbf{z}_i) = (\mathbf{z}_i - \boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1} (\mathbf{z}_i - \boldsymbol{\mu}) \tag{1}$$

where $\boldsymbol{\mu}$ is the mean of the training feature embeddings, and $\boldsymbol{\Sigma}$ is the shared covariance matrix. The Mahalanobis Distance provides a measure of how far a new feature embedding $\mathbf{z}_i$ is from the center of the training distribution. The same method is applied to find novel timesteps for the old agent by taking the boundary points, state and path-planning features.

III. Results

To have an general overview of the performance, both agents were tested in open-loop on 12 laps for two unseen tracks: track C and track D (Figure 6). Both agents were trained on 50 expert laps on track B and 50 on track A (Figure 5).

A. Full Track Generalization Gap

The generalization gap results highlight the differences in adaptability between the image-based and boundary point (BP) agents on track C and D. As can be seen in Figure 7 and Figure 8, the image agent showed a smaller generalization gap (0.014 on track C and 0.023 on track D) compared to the BP agent (0.071 on track C and 0.097 on track D). These results suggest that the image-based feature set allows for better generalization across unseen tracks, demonstrating a reduced performance drop from training to testing environments. This indicates that image features can capture a more versatile understanding of track layout, thereby enhancing the agent's ability to adapt to new tracks more effectively than traditional boundary point features.

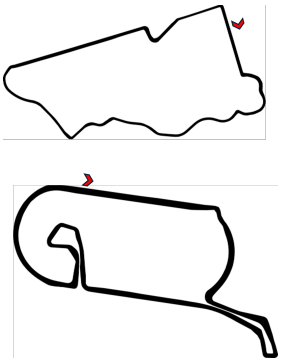

Fig. 5. Track A (above) and track B (below) for training the agent. Arrow indicating start and driving direction.

B. Open-Loop Timeseries Analysis

The timeseries of the action traces of each agent of a lap on track C and track D are shown in Figure 9 and Figure 10 respectively.

The overall RMSE results of the 12 laps on track C (Table 2) and 12 laps on track D (Table 3) show a clear higher performance for the image agent. Overall, the image agent has a loss of 2.3 times lower than the BP agent in steering, 1.5 times lower in braking and 2.5 times lower in accelerating. All together, this leads to a 66% lower loss in decision-making for the image agent in comparison to the BP agent. On the training data the image agent performs the same as the BP agent in terms of steering loss, and slightly better in braking accelerating (Table IV).

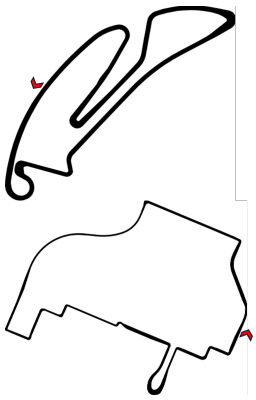

Fig. 6. Track C(above) and track D (below) for testing the agent. Arrow indicating start and driving driection.

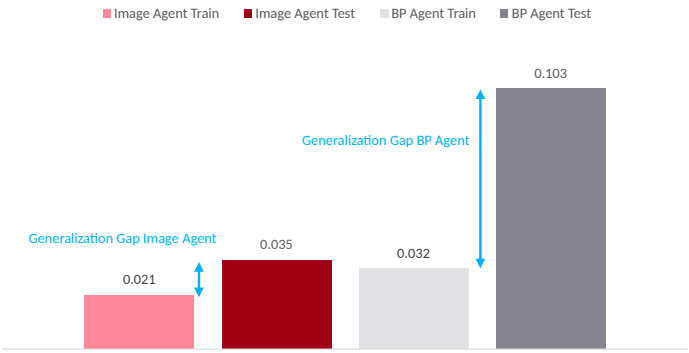

Fig. 7. Generalization gap of the RMSE on train data track A and B of the image agent and the boundary point agent when tested on track C.

1) Understeering of the BP Agent: On track C, the Boundary Point (BP) agent understeers in the first and third turn (visible in Figure 9), the issue likely stems from the BP agent's difficulty to predict and react appropriately to rapid changes in curvature. The boundary points focus on specific discrete distances, and these do not always provide sufficient granularity for fine-tuned lateral adjustments. This becomes a critical limitation in sections where maintaining a precise trajectory is necessary, such as in tight turns. Since boundary points represent the track in a simplified manner, the agent may not fully understand the sharpness of an upcoming turn until it is too late, leading to incorrect steering adjustments, in this case understeering.

TABLE II
RMSE BP AGENT AND IMAGE AGENT ON TRACK C

Model	RMSE actions		
	Steering	***Braking***	***Accelerating***
BP Agent	0.098	0.062	0.14
Image Agent	0.035	0.049	0.009

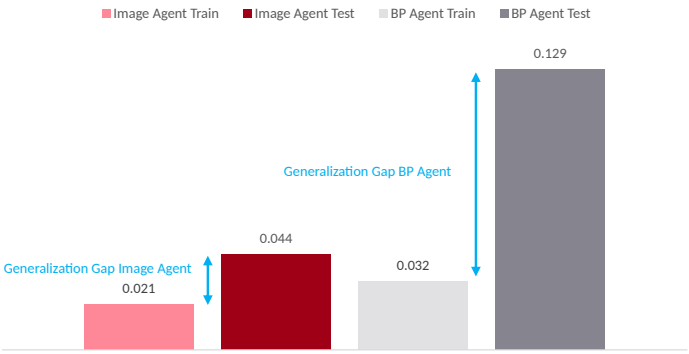

Fig. 8. Generalization gap of the RMSE on train data track A and B of the image agent and the boundary point agent when tested on track D.

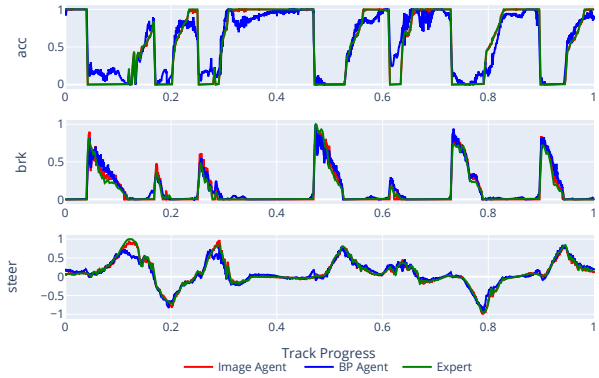

Fig. 9. Comparison of image agent and BP agent on a lap on track C. 'acc' refers to the throttle percentage, 'brk' refers to the brake pressure and 'steer' refers to the steering angle.

Furthermore, this issue is compounded by the nature of the BP agent's reliance on a smaller field of view compared to the image agent, which uses a broader set of visual information. The image agent has a wider 'awareness' of the track's full width and curvature, enabling more precise decision-making, which is especially important in complex cornering situations. This broader context allows the image agent to better predict and execute steering adjustments, reducing the likelihood of understeering.

2) Braking (Timing and Max Brake Pressure): The braking behavior observed on track C and D shows distinct differences between the BP and image agents, particularly in terms of braking timing. The BP agent often brakes too late (82% of the cases late, 18% early), missing the optimal braking points by **0.06 seconds** on average, whereas the IM agent, while better, is more prone to braking too early (83% of the case early, 17% late), **0.02 seconds** on average. Braking at the correct time is critical in motorsports, especially when the agent will be used in practice in a closed-loop simulation (includes feedback from the vehicle model and environment), because an error at the braking point can cascade, leading to a domino effect. In these cases the agent can enter corners at higher-than-expected

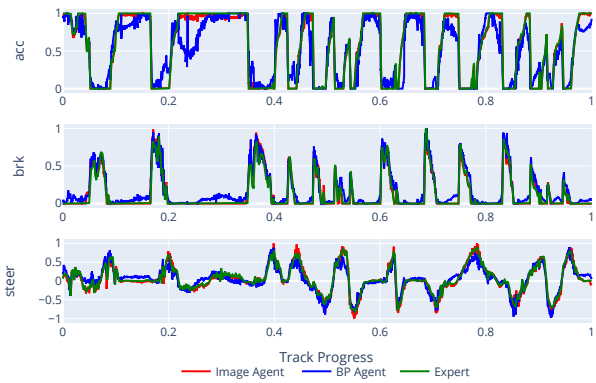

Fig. 10. Comparison of image agent and BP agent on a lap on track D. 'acc' refers to the throttle percentage, 'brk' refers to the brake pressure and 'steer' refers to the steering angle.

TABLE III
RMSE BP AGENT AND IMAGE AGENT ON TRACK D

Model	**RMSE actions**		
	Steering	***Braking***	***Accelerating***
BP agent	0.12	0.068	0.17
Image agent	0.043	0.060	0.018

speeds and struggle to stick to the plan and performance.

The BP agent's late braking is likely tied to the same issue of underrepresenting critical track features. Because the BP agent depends heavily on state-based features and boundary points, it may not fully recognize when to brake based on track curvature and width. This results in a reactive approach to braking, where the agent only responds once it is too late. In contrast, the image agent, which has a broader forward view and more detailed contextual information from the images, can anticipate braking points better, though it tends to be cautious and sometimes brakes earlier than necessary.

Both agents also exhibit occasional difficulties in reaching maximum brake pressure. While this happens less frequently, both models fail to reach the full braking capacity in around **15%** of the braking zones. The primary reason for this could be tied to an over-reliance on deceleration cues from the states instead of focusing on the direct braking actions.

TABLE IV
RMSE BP AGENT AND IMAGE AGENT ON TRAIN TRACKS

Model	**RMSE actions**		
	Steering	***Braking***	***Accelerating***
BP agent	0.032	0.026	0.041
Image agent	0.032	0.022	0.014

This subtle issue is more apparent in scenarios where braking is not just about reducing speed but also about maintaining control and maximizing tire grip, which requires hitting the maximum brake pressure precisely.

3) Acceleration Timing and Max Throttle Pressure: The timing of acceleration is another area where the BP agent struggles. On both the track C and D, the BP agent frequently accelerates prematurely, on average by **0.85 seconds** in **92%** of acceleration zones. Premature throttle application in a closed-loop setting can lead to significant performance drawbacks, as it may cause the vehicle to become unstable when exiting turns and can lead to overshooting the intended speed.

This behavior likely stems from the agent's over-reliance on state-based cues, such as velocity or slip ratio, which signal that the vehicle is ready to accelerate, rather than using track layout and trajectory to anticipate the optimal moment for throttle application. The BP agent appears to misinterpret these cues, causing it to initiate acceleration too early in anticipation of the need to gain speed.

This premature acceleration means that the agent overshoots the ideal throttle points, which can disrupt the vehicle's balance, especially when transitioning from cornering to straight-line driving. This can lead to reduced control and responsiveness, as the car may exceed the intended speed too quickly and require correction. In a closed-loop context, this early application of throttle can destabilize the vehicle's trajectory and increase the risk of understeering or overshooting turns, ultimately impacting its ability to maintain competitive performance on the track.

On the other hand, the image agent is able to more accurately time its acceleration. The image agent's broader awareness of the track allows it to anticipate when to get on the throttle, aligning its acceleration much more closely with the optimal points, early **95%** of the time by only **0.01 seconds**. The image agent also consistently reaches and keeps maximum throttle pressure more effectively.

4) Error Analysis Using Gates: In order to evaluate the agent's performance across different driving scenarios, I define three distinct types of 'gates', which correspond to different driving actions:

- **Cornering gate**: a section of the track where the expert driver is entering or in the middle of a cornering maneuver.
- **Braking gate**: a region where the expert driver applies significant braking force to decelerate the vehicle. This is detected when the brake pressure exceeds a minimal predefined threshold, indicating an active braking maneuver.
- **Drive gate**: a section where the expert driver is accelerating, marked by throttle input that surpasses a defined threshold.

Figure 8 shows the Kernel Density Estimates (KDEs) of action errors across three driving gates: Cornering, Brake, and Drive. The KDEs visualize the distribution of errors between predicted and expert actions, comparing the BP and

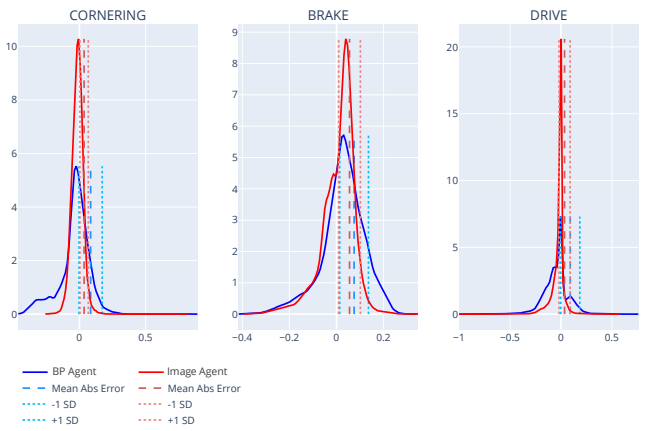

Fig. 11. Error KDEs of the BP agent and image agent of actions in specific gates. In 'cornering' only the steering error is taken, in 'brake' the braking error and in 'drive' the acceleration error.

image agent models. Each plot demonstrates how the errors are distributed around zero (where zero indicates perfect prediction) and provides a visual representation of how well each agent performs in different driving scenarios.

Cornering gate: In this gate, the image agent demonstrates a tighter distribution of errors around zero, with a lower mean absolute error of 0.04, indicating better accuracy in cornering compared to the BP model, which has a broader spread and a higher mean absolute error of 0.08. The standard deviation lines further emphasize the variability in predictions, with the image agent showing less deviation than the BP agent.

Brake Gate: For braking, both agents show similar mean absolute errors (0.07 for BP and 0.06 for image). However, the KDE shows that the image agent's errors are more tightly concentrated around zero, while BP has a slightly broader spread. The standard deviation reveals that the image model is more consistent in predicting braking actions with a lower variability compared to the BP model.

Drive Gate: The image model outperforms the BP model again, with a significantly lower mean absolute error of 0.01 compared to 0.09 for BP. The KDE demonstrates that image agent's predictions are clustered much closer to zero, meaning its driving acceleration is closely aligned with the expert data. In contrast, the BP model exhibits a broader error distribution, indicating less accurate predictions.

In summary, across all gates, the image agent consistently exhibits lower errors and less variability, particularly in acceleration (drive) and cornering. This demonstrates that image has a better overall understanding of the track and is more capable of replicating expert actions.

Gate	Image Agent		BP Agent	
	Mean Error	Std Error	Mean Error	Std Error
Cornering	0.035	0.032	0.084	0.088
Brake	0.055	0.046	0.074	0.061
Drive	0.011	0.016	0.091	0.094

C. Novelty Detection and True Generalization Gap

1) Novelty Detection Using Mahalanobis Distance: The Mahalanobis distance (Equation 1) was calculated separately for both the image-based and boundary point (BP) agents to assess the portions of each test track that resemble the training distribution, thereby identifying truly novel segments. On track C (Figure 12), 59.5% of the track for the image agent was classified as in-distribution, whereas only 13.8% was in-distribution for the BP agent. On track D (Figure 13), the in-distribution coverage was 24.5% for the image agent and a mere 1.2% for the BP agent. These percentages indicate that the image agent recognizes a larger portion of the test tracks as similar to its training data, likely due to the richer contextual information provided by image-based features, which aids in generalizing across similar track sections. Both agents have significantly less coverage of track D, these larger amount of timesteps that are out-of-distribution provide clarification on the higher mean error on track D in comparison to track C in Table II and Table III for both agents.

By isolating Out-of-Distribution (OoD) samples—those identified as novel by Mahalanobis distance—I can calculate the true generalization gap. For the image agent, the gap between in-distribution and OoD samples on the test track is determined as the difference between the in-distribution RMSE (0.021) and the OoD RMSE (0.079), yielding a gap of 0.058. In contrast, the BP agent shows a significantly larger true generalization gap, with an in-distribution RMSE of 0.032 and an OoD RMSE of 0.26, resulting in a gap of 0.228. This analysis provides a refined understanding of each agent's adaptability to genuinely unfamiliar track sections, offering insight into their generalization capabilities on unseen tracks.

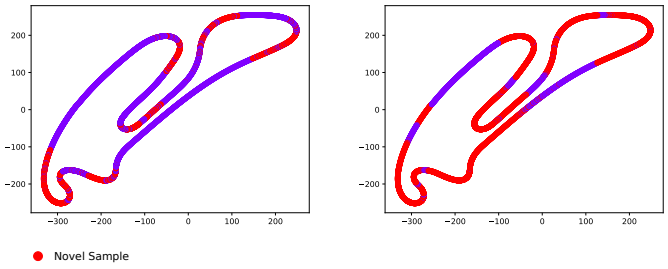

Fig. 12. Novel samples (or timestamps) on track C for each agent. The left image are the samples for the image agent and the right for the BP agent. The regions that are classified as novel are Out-of-Distribution (OoD) determined by the Mahalanobis Distance to the training data, whereas the other samples are In-Distribution (ID).

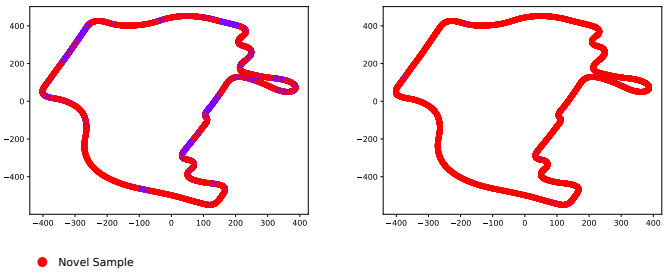

Fig. 13. Novel samples (or timestamps) on track D for each agent. The left image are the samples for the image agent and the right for the BP agent. The regions that are classified as novel are Out-of-Distribution (OoD) determined by the Mahalanobis Distance to the training data, whereas the other samples are In-Distribution (ID).

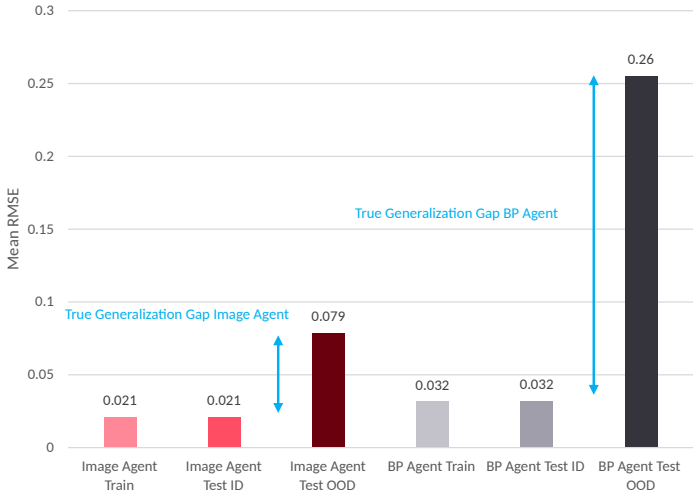

Fig. 14. True generalization gap for RMSE for the image and BP agents based on samples on the test track placed in (ID) and out of the training distribution (OoD).

2) Focused Attention and Error Analysis: The novel samples detected for the image-based agent on track C provide a natural segmentation of the track, as shown in Figure 15. These segments, defined by regions where the agent encounters novel inputs, allow for a more focused analysis to understand the causes of the generalization gap. By examining the agent's perception in each segment, I can identify specific areas where the model struggles to generalize. To gain further insights, Grad-CAM analysis [Selvaraju et al., 2017] was applied to these novel samples.

Using Grad-CAM, I identified an aggregated location of the highest-impact pixels (focus point), representing where the agent focuses its attention. By averaging the locations of the focus points across samples, I derived an average distance to the focus point for each segment, as well as a variance measure indicating the spread of focus within the region. To further investigate the role of these focus points, an additional agent was trained with track C included in the training data. Figure 16 shows the focus points and its spread of each agent. The focus distances—measured as the track distance between the car and the high-impact focus point—differed significantly

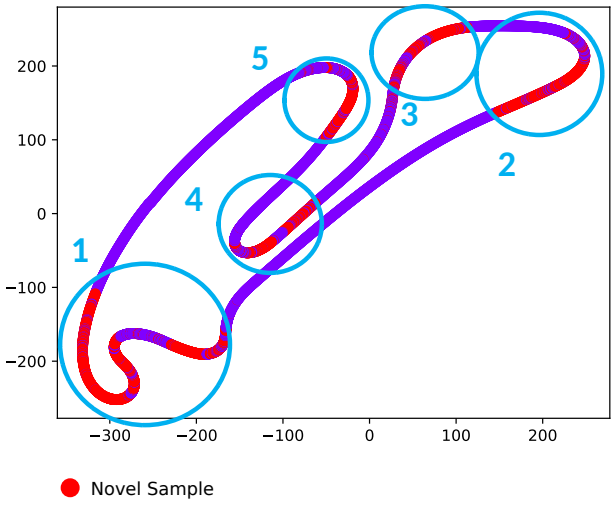

Fig. 15. Five identified segments based on novel samples on track C for the image agent.

between the agent trained with and without track C, revealing variations in perception across segments. In contrast, on the in-distribution parts of the track, the focus distances between the trained and untrained agents only differ by a maximum of 2 meters, indicating a much closer alignment in familiar sections. Furthermore, it was shown earlier that the agent has the highest average error in braking. The images in Figure 16 show timestep in the first turn of track C when the expert is at max brake pressure. The agent is braking at 0.88 times the expert brake pressure in the left image and exactly the expert's brake pressure on the right. The images show a difference of 30 m in focus distance in this specific case. On track C, the first turn is responsible for the majority of cases the image agent was unable to reach max brake pressure on this track. This trend is apparent in other regions where the image agent doesn't brake hard enough (in the case of these samples, they are out of distribution), on average this distance is approximately 28m. Overshooting the brake-pressure also happens in this turn, where the difference in focus distance is approximately 21m. On brake points where the agent manages to reach the right brake-pressure without overshooting, the difference in focus distance is approximately 7m. This large difference indicates a lack of attention to features farther ahead in higher braking error regions.

Table V summarizes the focus distances for each novel segment from Figure 15. These results indicate that the trained agent exhibits greater consistency in its focus across segments, with generally lower variance in focus distance, particularly in regions 1 and 3. The untrained agent, however, shows a wider spread in focus distance, suggesting that the inclusion of track C in training helps the agent develop a more stable and accurate perception of key areas on the track.

TABLE V
AVERAGE FOCUS POINT DISTANCES (FPD) AND FOCUS POINT STANDARD DEVIATION PER NOVEL SEGMENT FOR AGENT TRAINED ON TRACK C IN ADDTION TO TRACK A AND B (IMAGE AGENT TRAINED) AND AN AGENT ONLY TRAINED ON TRACK A AND B (IMAGE AGENT UNTRAINED)

Segment	Image Agent Untrained		Image Agent Trained	
	Avg. FPD	Avg. FP Std	Avg. FPD	Avg. FP Std
1	34	10.6	44	5.4
2	68	15.6	86	6.8
3	55	8.3	72	3.7
4	44	9.2	69	4.9
5	39	9.8	65	4.4

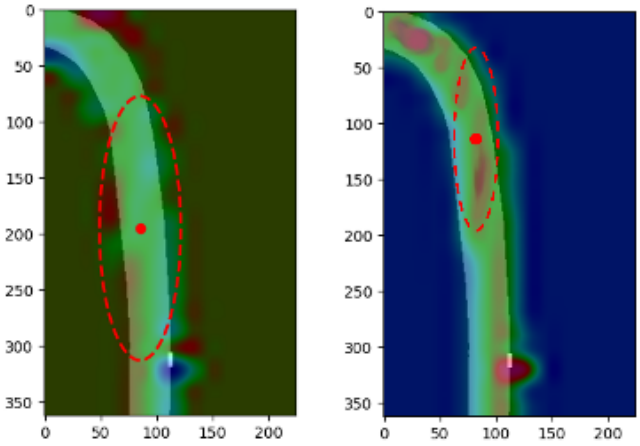

Fig. 16. Untrained (left) and trained (right) agents and their difference in gradients for the same timestep in turn 1. In this heatmap, the color scale represents the importance of different regions in the image: green areas indicate moderate impact and blue areas indicate little to no impact regions, while red areas highlight the pixels with the most influence on the agent's decision-making. The focus point is shown in red and the variance of gradients in x and y as a red ellipse. The focus distance of the right image is 30m of track distance ahead of the left image.

IV. DISCUSSION

The findings of this thesis demonstrate that incorporating image-based features has a positive impact on the generalizability of a race driver agent. Unlike previous approaches that relied on numeric state representations, which, although simple, often lack the detail necessary for precision on new or unseen data, image features provide a richer source of spatial awareness and contextual detail. This enhanced perception allows the agent to capture a more nuanced understanding of track conditions, leading to improved adaptability when transitioning to new environments.

One limitation of purely numeric features is their tendency to lead agents toward reactive behaviors, relying on state representations that may only trigger responses after certain thresholds are reached. This could limit the agent's ability to anticipate changes, resulting in delayed reactions rather than proactive adjustments. In contrast, image-based features offer a broader field of perception, allowing for more anticipatory decision-making that can initiate behavior changes before the agent encounters critical areas on the track.

This research suggests that shifting the focus from pure performance optimization to prioritizing generalization can have long-term benefits, especially in high-performance environments like motorsport. Improved generalization capabilities could contribute to more stable closed-loop performance, reducing the likelihood of compounding errors that can emerge when the agent encounters unfamiliar conditions. By minimizing the performance gap between in-distribution and out-of-distribution scenarios, image-based agents are better equipped to handle the variability and unpredictability of racing, thereby taking a step closer to integrating into the iterative feedback loops of real-world motorsport engineering. Additionally, if a model successfully closes the generalization gap, this could provide insights into areas where the driver may face difficulty on a new track, effectively acting as a tool for identifying and addressing challenging sections.

However, achieving a truly generalizable agent—one capable of maintaining comparable RMSE in both in-distribution and out-of-distribution cases—remains a challenge. The current image-based approach, while effective, still lacks certain informational cues that could lead to even better decision-making. The resolution of the image is currently set to 0.4 meters per pixel, which impacts the precision of both the car's location and the track borders, introducing a degree of imprecision. In a closed-loop scenario, this could lead to localization errors, potentially resulting in collisions or off-track events. Increasing the resolution could enhance precision but would also raise simulation time and computational cost. Further closed-loop analysis will be necessary to evaluate the trade-offs between resolution and stability in the agent's behavior, aiming to identify an optimal balance where precision is sufficient without compromising real-time performance. Furthermore, a time-based image feature as described in Appendix A could provide the right amount of precision when needed (sharp, slow corners), while reducing the resolution at fast paced (straights) regions. The input-size could remain the same, however the additional calculations could still slow down simulation. These features could also potentially be very track and lap specific and therefore prone to overfitting. Extensive tuning and testing would be needed to validate this method. Another clear limitation of the image agent is its inability to reach the maximum brake pressure in 15% of the test cases. Braking is essential and needs to be done at the right amount in order to approach curves effectively. The focus area of the agent is shows a larger spread and is focused closer towards the agent in regions of error. This is also apparent specifically in braking. Additional information highlighting regions of the image the model can expect valuable information could be beneficial.

This work establishes a baseline in image feature design for race car driver agents, setting the foundation for further exploration of more advanced perception methods. Future research could examine alternative theories in image-based

perception, such as the role of peripheral vision and the impact of velocity on visual focus and attention. Additionally, incorporating multi-agent dynamics and adapting to specific track conditions could provide valuable insights, potentially leading to agents that are not only generalizable but also capable of sophisticated interactions in complex racing environments.

This work also highlights the effectiveness of incorporating image-based features that, even with a simple design, capture broad spatial and contextual information beneficial for fast-paced simulation environments. This design simplicity allows the model to focus on relevant cues without unnecessary detail. By demonstrating the feasibility and advantages of image-based approaches in motorsport, this effectively opens a door to plenty of possibilities. In urban driver modeling, image features have enabled advanced capabilities such as scene understanding, semantic segmentation, and adaptive decision-making in highly variable environments (summarized in [Tampuu et al., 2020]). In regards to further improvement in generalization, concepts such as attention mechanisms, domain adaptation, and multi-scale feature extraction, long utilized in urban settings to tackle the unpredictability of real-world driving, can be adapted and refined for motorsport's challenges.

By accelerating and improving the efficiency of the innovation process in motorsport through better generalization, this research reduces the reliance on extensive data collection and training for every new track. This enables faster development cycles, more resource-efficient experimentation, and a streamlined path to advancing both competitive performance and the underlying technologies that drive progress in the automotive and mobility sectors.

V. Conclusion and Recommendations

In this thesis, I aimed to study the effects on track generalization in motorsport driver modeling of image-based feature sets with a behavioral cloning agent.

The use of images as perception features enabled the agent to generalize more effectively to unseen tracks. In comparison to boundary point-based perception, the image-based agent demonstrated a 71% reduction in RMSE on in steering, braking, and acceleration on out of distribution test data. This result confirms that image-based features offer a more robust and adaptable representation of the track's spatial and contextual information, crucial for effective generalization.

The remaining gap on both test tracks in generalization for the image agent can be partially attributed to certain segments of track C where the decisions made by the agent showed worse performance, particularly in braking. Test track D was overall more difficult to tackle by the agent due to the larger number of samples in truly novel regions.

To further enhance the performance of autonomous agents in racing, future work could explore:

- Improving forward-looking attention: the analysis of the focus point suggests that agents trained on unfamiliar tracks tend to focus too close to the vehicle, resulting in poor anticipation of upcoming track features. Future work could include refining the agent's attention mechanisms to extend its forward-looking capabilities, particularly in unfamiliar or novel track regions. This could be achieved by enhancing the feature extraction process to prioritize long-range track visibility by pretraining the CNN on an auxiliary task such as apex location prediction. Another method could be to use a training method such as the one described in [Akhauri et al., 2021], where saliency, gradient and edge maps (including GradCAM) generated by a pretrained model were used to train a second model to ensure focus on the important regions of the image.
- Incorporating additional information into image features: enhancing image inputs by adding information such as brake points or the projected trajectory ahead could provide more context for the agent, enabling better decision-making. Peripheral vision could also serve as inspiration, where resolution and focus vary with velocity, allowing the agent to adapt its perception based on its speed and driving conditions.
- Considering future information on other agents: future research could explore incorporating data on the presence and actions of other agents on the track, enabling the model to account for dynamic interactions. This would be especially beneficial for multi-agent racing scenarios, where anticipating the behavior of other vehicles becomes critical.
- Expanding the track dataset: including a wider variety of tracks with more complex features in the training dataset could help test the robustness of the agent in even more novel environments. This would provide a broader range of scenarios and help the agent adapt to diverse track layouts.
- Integrating closed-loop evaluations: closed-loop evaluations could be used to assess how the agent handles long-term decision-making and corrective actions in real-time racing scenarios. This would give insights into how well the model performs under conditions where its actions influence future states of the environment, simulating real racing conditions more accurately.

These enhancements would not only improve the agent's adaptability and anticipation of track features but also open the door to more sophisticated behavior in complex, multi-agent racing environments.

References

Shivam Akhauri, Laura Zheng, Tom Goldstein, and Ming Lin. Improving generalization of transfer learning across domains using spatio-temporal features in autonomous driving. *arXiv preprint arXiv:2103.08116*, 2021.

Fabian Behrens. Procedural race track generation for domain randomization. 2020.

Y. Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Journal of the American Podiatry Association*, volume 60, page 6, 06 2009. doi: 10.1145/1553374.1553380.

F. Braghin, F. Cheli, S. Melzi, and E. Sabbioni. Race driver model. *Computers & Structures*, 86(13):1503–1516, 2008. ISSN

0045-7949. doi: https://doi.org/10.1016/j.compstruc.2007.04.028. Structural Optimization.

Nemanja Djuric, Vladan Radosavljevic, Henggang Cui, Thi Nguyen, Fang-Chieh Chou, Tsung-Han Lin, Nitin Singh, and Jeff Schneider. Uncertainty-aware short-term motion prediction of traffic actors for autonomous driving. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 2095–2104, 2020.

Florian Fuchs, Yunlong Song, Elia Kaufmann, Davide Scaramuzza, and Peter Duerr. Super-Human Performance in Gran Turismo Sport Using Deep Reinforcement Learning. *IEEE Robotics and Automation Letters*, 6(3):4257–4264, July 2021. ISSN 2377-3766, 2377-3774. doi: 10.1109/LRA.2021.3064284. arXiv:2008.07971 [cs].

Adithya Ganesh, Joe Charalel, Matthew Das Sarma, and Nancy Xu. Deep reinforcement learning for simulated autonomous driving, 2016.

Thomas Gustafsson. Computing the ideal racing line using optimal control, 2008.

Jyh-Jing Hwang, Runsheng Xu, Hubert Lin, Wei-Chih Hung, Jingwei Ji, Kristy Choi, Di Huang, Tong He, Paul Covington, Benjamin Sapp, et al. Emma: End-to-end multimodal model for autonomous driving. *arXiv preprint arXiv:2410.23262*, 2024.

Siwei Ju, Peter van Vliet, Oleg Arenz, and Jan Peters. Digital twin of a driver-in-the-loop race car simulation with contextual reinforcement learning. *IEEE Robotics and Automation Letters*, 8 (7):4107–4114, 2023. doi: 10.1109/LRA.2023.3279618.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2017.

Quanyi Li, Zhenghao Peng, Qihang Zhang, Chunxiao Liu, and Bolei Zhou. Improving the generalization of end-to-end driving through procedural generation. *arXiv preprint arXiv:2012.13681*, 2020.

Stefan Alexander Löckel. Machine learning for modeling and analyzing of race car drivers. 2022a.

Stefan Alexander Löckel. *Machine Learning for Modeling and Analyzing of Race Car Drivers*. PhD thesis, Technische Universität Darmstadt, Darmstadt, 2022b.

Goeffrey J McLachlan. Mahalanobis distance. *Resonance*, 4(6):20–26, 1999.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, February 2015. ISSN 1476-4687. doi: 10.1038/nature14236. URL https://doi.org/10.1038/nature14236.

Adrian Remonda, Sarah Krebs, Eduardo Veas, Granit Luzhnica, and Roman Kern. Formula RL: Deep Reinforcement Learning for Autonomous Racing using Telemetry Data, June 2022. arXiv:2104.11106 [cs].

Stephane Ross and Drew Bagnell. Efficient reductions for imitation learning. In Yee Whye Teh and Mike Titterington, editors, *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pages 661–668, Chia Laguna Resort, Sardinia, Italy, 13–15 May 2010. PMLR. URL https://proceedings.mlr.press/v9/ross10a.html.

Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pages 618–626, 2017.

Ardi Tampuu, Maksym Semikin, Naveed Muhammad, Dmytro Fishman, and Tambet Matiisen. A Survey of End-to-End Driving: Architectures and Training Methods. *arXiv e-prints*, art. arXiv:2003.06404, March 2020. doi: 10.48550/arXiv.2003.06404.

Riccardo Volpi, Hongseok Namkoong, Ozan Sener, John Duchi, Vittorio Murino, and Silvio Savarese. Generalizing to Unseen Domains via Adversarial Data Augmentation. *arXiv e-prints*, art. arXiv:1805.12018, May 2018. doi: 10.48550/arXiv.1805.12018.

Svante Wold, Kim Esbensen, and Paul Geladi. Principal component analysis. *Chemometrics and intelligent laboratory systems*, 2(1-3): 37–52, 1987.

Yi Xiao, Felipe Codevilla, Akhil Gurram, Onay Urfalioglu, and Antonio M López. Multimodal end-to-end autonomous driving. *IEEE Transactions on Intelligent Transportation Systems*, 23(1): 537–547, 2020.

APPENDIX

APPENDIX A: DECISION CRITERIA FOR INITIAL IMAGE DESIGN

When determining the optimal image design for training my model, several practical and problem-specific criteria had to be addressed. The practical criteria were shaped by the simulation environment and available computational resources. The model needed to work efficiently with a single GPU and a 32-core CPU, ensuring a manageable training time, inference speed, and computational complexity. In simulation, the goal was to run as close to real-time (100Hz) as possible, requiring fast processing for effective deployment.

From a problem-solving perspective, the images needed to fulfill the following criteria:

1) **Sufficient Surrounding Information**: The model must perceive enough of the track environment to make informed decisions.
2) **Clear Vehicle Localization**: It was critical for the car's position to be distinctly represented in the images.
3) **Focus on Upcoming Scenarios**: The images must capture upcoming sections of the track central to decision-making, such as turns or straights.
4) **Track Segments Impacting Current Decisions**: Any segment that might influence the agent's next actions needed to be visible, such as the following turn or straight section.

Initial Trials

Given these criteria, I initially experimented with a global track map (as seen in Figure 17), using the Nature CNN architecture. This architecture is commonly applied in reinforcement learning tasks, in continuation of my findings from my thesis, my method would need to be implementable for the Reinforcement Learning agent described in [Ju et al., 2023] (other CNN architectures were additionally tested, this information can be found in Appendix 3).

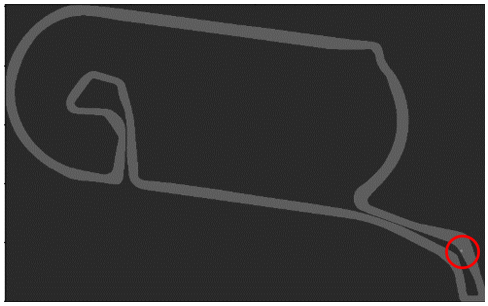

Fig. 17. Global track map image feature. Location of the car is circled in red.

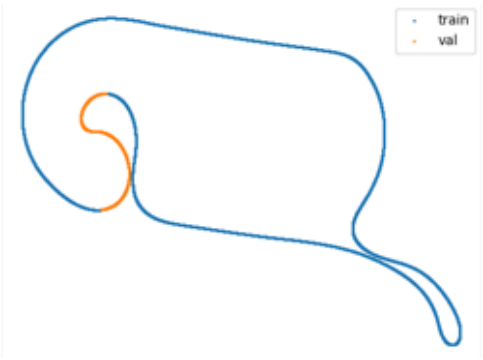

Fig. 18. Locations of training and validation split of samples on track B for initial generalization tests.

However, while the approach of a global map captured the full track, it did not meet real-time performance goals. With a full map resolution of 1 pixel per meter, each prediction took around 0.1 seconds, which is ten times slower than real-time simulation requirements (0.01s per prediction). Moreover, at this resolution, the distinction between the car and the track was insufficient.

Refinement: Focused Image Regions

Realizing the limitations of the full map, I shifted focus to smaller regions of the track, where critical decision-making occurs. Four key parameters were tested to strike a balance between simulation time and the necessary visibility for decision-making:

- **Forward Visibility**: How far ahead the car can see (measured in meters).
- **Horizontal Visibility**: The lateral view to the left and right of the car (measured in meters).
- **Backward Visibility**: The view behind the car, to understand rear dynamics (measured in meters).
- **Resolution**: The image's spatial granularity, in pixels per meter.

Through multiple tests, I determined that a reasonable simulation time of 0.06 seconds per prediction could be achieved with the following parameter values:

- **Forward Visibility**: 200 meters
- **Horizontal Visibility**: 120 meters
- **Backward Visibility**: 50 meters
- **Resolution**: 0.1 meters per pixel (full resolution)

Training and Validation Tests

Once these values were determined, I conducted training and validation tests using a Nature CNN on track data. The train-validation split was set to 85% and 15%, respectively, across 50 laps of data (train and validation segment shown in Figure 18). The primary focus was to determine whether the chosen parameters led to a generalizable model or caused overfitting to the training data.

The resulting figures (Figure 19 and Figure 20) illustrate the relationship for both forward and horizontal visibility to the model's training and validation losses at the best epoch. A clear cut-off point for both forward and horizontal visibility was observed. Beyond certain visibility thresholds, the additional information led to diminishing returns and a larger gap between training loss and validation loss. This gap indicates that adding too much visibility may cause the model to overfit to the training data, reducing its ability to generalize to unseen track segments. For backward visibility a similar pattern emerged with a cut-off at 20m. The resolution analysis left with a choice of a low loss for training and validation (0.00604 training, 0.0084 in validation) at a resolution of 0.4m per pixel or a slightly higher resolution (0.3m per pixel) with a lower loss (0.00601, 0.0083) at the cost of an increase in inference time of 0.008s. The final decision was taken in favour of faster simulation and therefore 0.4m per pixel.

Final Design Choices

Based on these tests, I refined the final design to the following parameters:

- **Forward Visibility**: 125 meters directly in front of the agent.
- **Horizontal Visibility**: 90 meters total (45 meters to each side of the agent).
- **Backward Visibility**: 20m behind the agent
- **Resolution**: 0.4m per pixel

The final input size into the model is therefore: 362 x 225. A single channel was chosen where the values are normalized to: car: 1, track: 0.5, off-track: 0.

These values balanced the need for accurate decision-making with the constraints of simulation time and computational power. The cut-off points illustrated in Figure 19 and Figure 20 demonstrate that further increasing forward or horizontal visibility would have resulted in diminishing returns, leading to a larger gap between training and validation loss. Therefore, these values were used as a compromise to avoid overfitting while maintaining sufficient visibility for the agent to make informed decisions.

Appendix B: Exploration of Other Image-Based Feature Variants

This section provides an overview of the different image-based feature variants considered during the development of

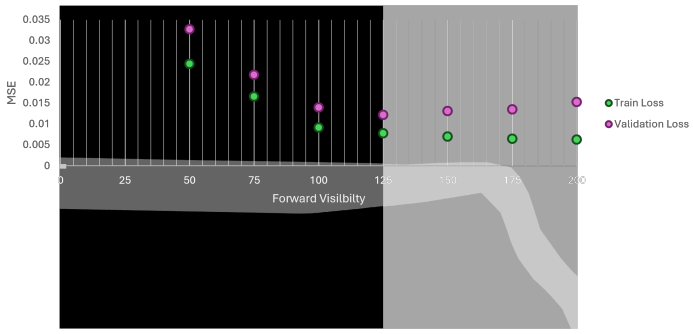

Fig. 19. Train loss and validation loss for agents with different forward visibility. Cut-off for design parameter decision can be seen where image turns to grey-scale, validation loss diverges from 125m.

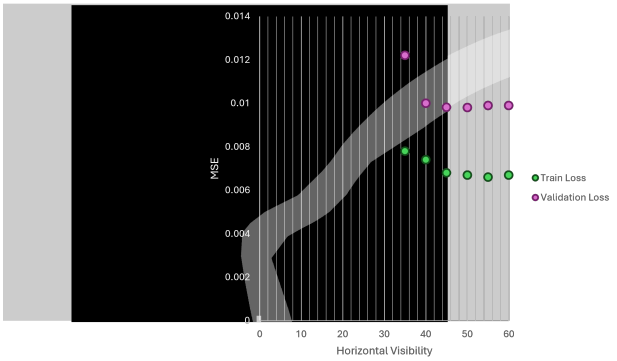

Fig. 20. Train loss and validation loss for agents with different horizontal visibility. Cut-off for design parameter decision can be seen where image turns to grey-scale on the right, validation loss diverges from 45m.

the behavioral cloning model for race car driver behavior. The focus is on the trade-offs between local and global information, time complexity, and smoothness penalties applied to the images.

Key Concepts: Local, Global Information, Time & Smoothness

Local information refers to the immediate surroundings of the car, primarily critical for close-up maneuvers such as cornering, and capturing the car's behavior with respect to nearby track boundaries.

Global information refers to the lookahead distance or forward visibility that gives the agent an understanding of the upcoming sections of the track, crucial for high-speed decision-making and long-term planning.

Time refers to the total computational cost of training and inference, which is critical for offline simulations and deployment. The creation time of each image feature, combined with the training time, provides a measure of the computational load imposed by each feature variant.

Smoothness measures the consistency of the image transitions from one frame to the next, as assessed by optic flow. A smooth image transition reflects natural, less erratic visual inputs, which can help the model generalize better. A smoothness penalty is applied to variants where distortions occur to keep specific elements (like track borders) in focus, which might otherwise disrupt training.

The following feature variants were explored, each tackling the balance between local and global information, time complexity, and smoothness differently. The main performance conclusions are based on time complexity, local and global information, and the effects of smoothness.

IM_ALA (Agent-Lookahead)

In this variant, the car always maintains a fixed lookahead distance ahead of it, corresponding to its braking distance at maximum speed. This design ensures that the agent has sufficient forward visibility to plan maneuvers but might miss portions of upcoming corners due to the fixed width of the image. The resolution is consistent, and the car's position is fixed in the frame.

Strengths: Captures local information reliably without losing details. It doesn't suffer from significant image distortions.

Weaknesses: Some critical information about corners might be missed because the image is always aligned with the heading angle of the car.

Time: IM_ALA is the least computationally expensive, as both the creation and training times are minimal.

Smoothness: High, as the image is not dynamically adjusted.

IM_TLA (Track-Lookahead)

Two variants of this method are used: **TLA_time** and **TLA_distance**.

TLA_time (Time-Based Lookahead): This variant captures the car's future and past positions, measured in time (seconds) ahead and behind. The lookahead distance changes based on the car's velocity, mimicking how humans focus further ahead at higher speeds. This results in the resolution adjusting based on speed: at lower velocities, the resolution improves, providing more local detail, while at higher velocities, the focus shifts further down the track.

Strengths: Dynamic resolution based on time offers an advantage for low-speed maneuvers, providing clearer local detail. It balances local and global information well, particularly in time-critical decision-making.

Weaknesses: Some information is cropped off because the method doesn't track the borders of the track, focusing instead on maintaining the future point within the frame.

Time: TLA_time has moderate computational cost, more than IM_ALA but still less than the other variants due to the temporal nature of its adjustments.

Smoothness: High, due to its relation to velocity, allowing for more natural transitions at different speeds.

TLA_distance (Distance-Based Lookahead)

In this variant, the lookahead is based on a fixed track distance, not time. The advantage here is that it adjusts slightly based on the curvature of the track, meaning it provides more visibility around sharp turns than IM_ALA. However, like TLA_time, some local information may be cropped off

because the method doesn't prioritize tracking track borders but rather keeps the desired future point within the image.

Strengths: Ties the lookahead more closely to the curvature of the track, making it more effective in anticipating sharp turns.

Weaknesses: The dynamic adjustment may lead to cropping similar to TLA_time. Cropped sections can lead to missing important track boundaries in certain scenarios.

Time: Similar to TLA_time, but slightly more computationally expensive because it involves additional processing linked to track curvature.

Smoothness: Moderate, linked to curvature but still more stable than IM_FZoom.

IM_FZoom (Focused Zoom)

This variant is designed to ensure that all track borders between the car and a predefined focus point (e.g., 125 meters ahead) are always visible in the image. The car is positioned towards the bottom of the image, and the focus point is at the top, ensuring complete coverage of the track between them. The primary trade-off in this method is the frequent image distortions necessary to maintain this level of coverage, which negatively impacts smoothness.

Strengths: Excellent global information capture since it tracks the entire segment between the car and the focus point. Ideal for scenarios where it is crucial to have all track borders visible.

Weaknesses: High computational cost due to frequent distortions. The smoothness is significantly affected by these adjustments.

Time: IM_FZoom is the most computationally expensive, due to both the creation time (as the entire track between the car and the focus point must be included) and the distortions that lead to increased processing time.

Smoothness: Poor, as frequent adjustments and distortions are necessary to maintain the focus on track borders.

Analysis Methodology

The plots presented in Figure 21 and Figure 22 aim to compare the performance of different feature variants based on time, local information, global information, and smoothness. The test was done on track B and all parameter tuning in this analysis was done specifically for this track. The input into the network would ideally be the same in order to avoid further architecture hyperparameter tuning per variant. The input size for this analysis chosen was based on the results of (APPENDIX A): (362, 225). The following parameters were used for each feature variant in the analysis:

Parameters for each variant:

- **Forward visibility:**
  - *IM_ALA (Agent Lookahead)*: 125m directly in front of the agent (see Appendix A for further reasoning).
  - *TLA_distance (Track Lookahead Distance)*: 125m of track distance measured along the expert trajectory.
  - *IM_FZoom (Focused Zoom)*: Focus point at 125m of track distance ahead (along expert trajectory).
  - *TLA_time (Track Lookahead Time)*: Based on time rather than distance. The forward visibility is determined by the average braking duration (2.9 seconds) plus a small margin (1.5 seconds for handling slow turns (determined after tuning)), which totals 4.4 seconds. The distance from the agent to the location of the expert 4.4 seconds ahead is taken per image.
- **Horizontal visibility:**
  - *IM_ALA (Agent Lookahead)*: 45m on either side (see Appendix A for further reasoning).
  - *TLA_distance (Track Lookahead Distance)*: is dependant on the scenario. 45m on straights. Affected by the curvature of the track and what distance is shown in the horizontal direction in that scenario.
  - *IM_FZoom (Focused Zoom)*: Scenario dependent. Zoom keeps all of the track in the frame so this depends on the track curvature.
  - *TLA_time (Track Lookahead Time)*: Similar to TLA_distance and IM_FZoom. The width varies with the time taken by the expert to pass a certain segment.
- **Backward visibility:**
  - *IM_ALA (Agent Lookahead)*: 20m behind the agent (see Appendix A for further reasoning).
  - *TLA_distance (Track Lookahead Distance)*: 20m behind the agent in distance on the expert trajectory.
  - *IM_FZoom (Focused Zoom)*: 20m behind the agent in distance on the expert trajectory.
  - *TLA_time (Track Lookahead Time)*: 0.5 seconds behind the agent. Keeps the 20m on straights with a small margin of 0.2 seconds (tuned during designing the images).
- **Resolution:**
  - *IM_ALA (Agent Lookahead)*: 0.4 meters per pixel (see Appendix A for further reasoning)
  - *TLA_distance (Track Lookahead Distance)*: varies between 0.4 meters per pixel at top speed (straights) and max resolution (0.1 m per pixels) at complex curves.
  - *IM_FZoom (Focused Zoom)*: varies between 0.4 meters per pixel at top speed (straights) and max resolution (0.1 m per pixels) at complex curves.
  - *TLA_time (Track Lookahead Time)*: varies between 0.4 meters per pixel at top speed (straights) and max resolution (0.1 m per pixels) at slow segments.

Time calculation: Time in the plots represents a combination of inference time (the time it takes to process each sample during simulation) and training time (time per epoch during model training). This combination provides an overall measure of computational efficiency for each feature variant.

Local information calculation: Local information is calculated by measuring how much of the immediate track around the agent is captured within a 20m radius. For each variant, the number of track borders (points where the track meets its boundary) falling inside this radius during a lap was

counted. Variants with more local detail will show higher local information values.

Global information calculation: Global information represents how much of the track ahead is captured in the image. For each feature variant, the total number of track borders visible in the image was counted. Variants with wider coverage of the track ahead (including borders and curves) yield higher global information values.

Smoothness adjustment:

Both local and global information are adjusted based on the smoothness of the image transitions. This was measured using the Farneback optic flow algorithm, which computes pixel displacements between consecutive images by approximating local neighborhoods with quadratic polynomials. The optic flow field $\mathbf{f}(x, y) = (u(x, y), v(x, y))$ represents the flow vector at each pixel (x, y), where $u(x, y)$ and $v(x, y)$ denote the horizontal and vertical displacements, respectively. The magnitude of the flow is calculated as:

$$|\mathbf{f}| = \sqrt{u(x, y)^2 + v(x, y)^2}$$

Larger magnitudes indicate more abrupt changes between consecutive frames, meaning less smooth transitions. To penalize this, the average optic flow magnitude $|\bar{\mathbf{f}}|$ was computed for each method, and the penalty was applied based on the formula:

$$\text{Adjusted information} = \text{Information} \times \left(1 - \frac{|\bar{\mathbf{f}}|}{\text{max flow}}\right)$$

Variants with larger average optic flow magnitudes, such as IM_FZoom, were penalized more, reducing their overall local and global information scores. This penalty reflects the fact that smoother transitions are preferred in image-based decision-making, where abrupt visual changes may mislead the model.

Conclusion: trade-offs in time, local & global information

The main trade-offs between these feature variants, as illustrated in the plots, revolve around the balance between local information, global information, time complexity, and smoothness. IM_ALA stands out as a well-rounded variant that balances local detail with minimal cropping but faces challenges in global visibility, particularly around corners.

The TLA variants (TLA_time and TLA_distance) dynamically adjust the lookahead based on speed and track curvature. While these methods provide flexibility, they come with the drawback of occasionally cropping parts of the track that lie outside their scope (the cropping is dependent on keeping the track distance in the image, not every trackborder specifically). Despite this, their behavior closely mimics human driving, where the focus shifts depending on speed and maneuvers, making them strong contenders for tasks that require generalization across various tracks.

IM_FZoom does well in providing comprehensive global information, as it ensures that all track borders between the car and the focus point are included. However, this comes at the cost of smoothness due to frequent image distortions and high computational complexity, making it less feasible for real-time applications.

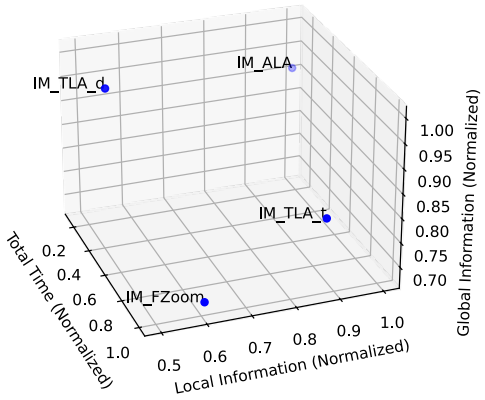

Fig. 21. 3D plot showing the parameters taken into account for the trade-off to determine the image feature design: global information, local information and total time.

From a time complexity perspective, IM_ALA is the least computationally expensive, making it optimal for fast offline simulations and real-time applications. In contrast, IM_FZoom incurs the highest computational cost due to its extensive coverage of track borders, making it more suitable for scenarios where detailed global information is critical and computational resources are less constrained.

Finally, the velocity effect plays a crucial role in the TLA variants. TLA_time adjusts its resolution based on speed, providing more local detail at slower speeds. Meanwhile, TLA_distance uses track curvature to adjust the lookahead, maintaining focus on relevant track sections. Both methods offer dynamic behavior, but at the cost of occasionally cropping information, as they do not track borders as comprehensively as IM_FZoom.

The appendix plots illustrate these trade-offs in detail, focusing on the interplay between time, local, and global information, as well as the smoothness penalty applied to each variant. In conclusion, IM_ALA provides the best balance between all factors for the purposes of this thesis, offering sufficient local and global coverage with minimal time complexity, making it the preferred method for real-time simulation in racecar driver modeling.

APPENDIX C: ADDITIONAL RESULTS

This section provides an overview of the additional results on the training and testing data.

C.1: Performance Results BP agent & image agent on training data

The training performance of both feature sets, BP agent and image agent, was measured using a combination of learning curves and open-loop evaluation across multiple laps.

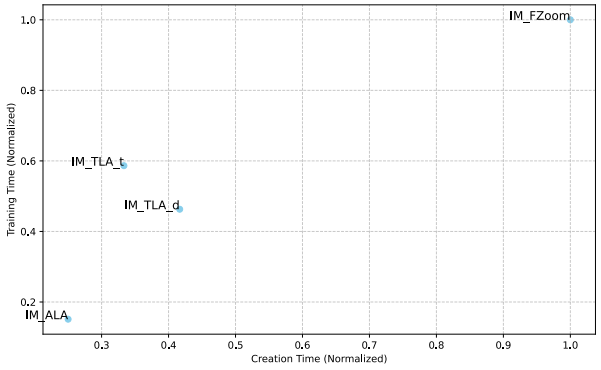

Fig. 22. Training time and creation time comparison for considered feature designs

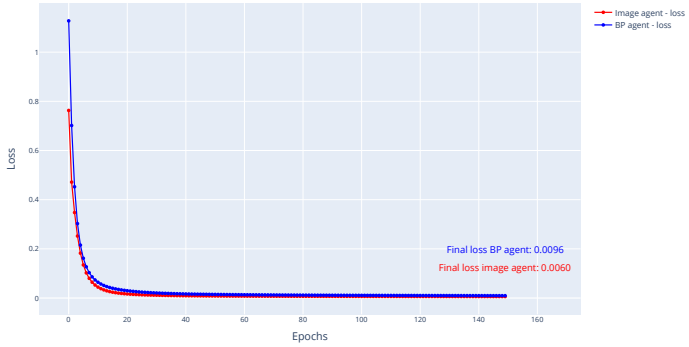

Fig. 23. Learning curve image agent and BP agent on training data

1) Learning Curves: The learning curves for both BP agent and image agent show the reduction in loss across training epochs. Figure 23 demonstrates that both models successfully converge, with image agent reaching a slightly lower loss at the end of training, indicating indicating better fitting to the training data compared to BP agent. Specifically, the final loss for image agent is 0.0060, whereas BP agent ends with a loss of 0.0096. While both agents show a similar trend in learning, the rate at which BP agent decreases early on indicates that it may adapt faster initially, but ultimately fails to achieve the lower loss of image agent.

This behavior could be attributed to the different architectures and regularization techniques employed in the models. For instance, image agent, which incorporates more advanced feature processing (e.g., higher-order feature extraction), might be better at capturing finer distinctions between track states, leading to a more refined convergence.

2) Open-Loop Evaluation: For open-loop evaluation, both models were tested on a single lap on track B, as shown in Figure 24. The predicted actions for steering, braking, and acceleration are overlaid against the expert data to assess how well the models handle real-time decision-making without feedback correction (i.e., open-loop).

The image agent model shows a closer alignment with expert actions in the critical regions of the track, particularly during braking and mid-corner transitions, where accurate

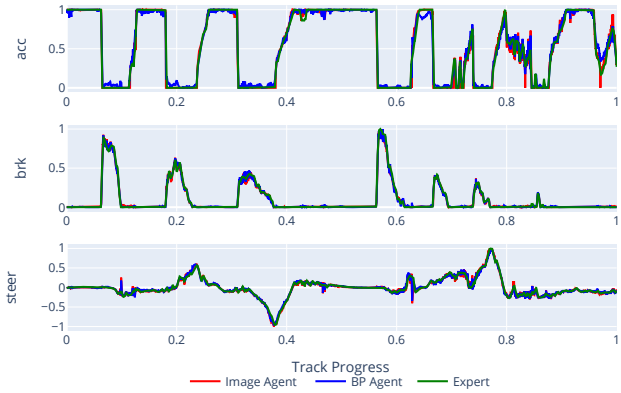

Fig. 24. Actions of the image agent, BP agent and expert in an open-loop simulation for a lap on track B

predictions are essential for maintaining stability and optimal driving trajectories. In contrast, BP agent exhibits greater deviations, particularly in the mid-corner sections, where its braking and steering predictions lag behind or overshoot the expert data, leading to potential understeer or oversteer scenarios.

However, both models demonstrate some level of difficulty with acceleration transitions, particularly in sharp turns and exit points. While image agent is generally more consistent.

3) RMSE Analysis Across Gates: Table VI provides a detailed breakdown of the RMSE for steering, braking, and acceleration across various 'gates' or track segments on track B, which represent specific driving scenarios like cornering, mid-corner, and entry/exit transitions.

- **Steering:** Both the BP agent and the image agent show similar performance across most gates, but the image agent consistently performs slightly better. For example, in the cornering gate, image agent achieves an RMSE of 0.0320, compared to BP agent's 0.0412. This suggests that image agent has better control and prediction accuracy during sharp turns, likely due to its enhanced handling of complex track geometries.
- **Braking:** There is a more substantial difference in braking performance. In gates like 'brake' and 'entry' , the image agent outperforms the BP agent significantly, with RMSE values of 0.0327 and 0.0313, respectively, versus 0.0308 and 0.0316 for the BP agent. This indicates that the image agent is better at predicting when and how much braking force to apply, which is critical in minimizing braking distances and improving corner entry performance.
- **Acceleration:** For acceleration, image agent generally outperforms the BP agent in most gates, showing lower RMSE values in crucial segments such as 'entry' and 'cornering' . For example, in the 'entry' gate, image agent has an RMSE of 0.0095 compared to the BP agent's 0.0244. Similarly, in the 'cornering' gate, image agent achieves an RMSE of 0.0315, while the BP agent

TABLE VI
EVALUATION TRACK B FOR 12 LAPS IN SPECIFIED GATES

Model	Gate	RMSE		
		Steering	Braking	Accelerating
BP agent	Full lap	0.0321	0.0263	0.0409
	Brake	0.0331	0.0308	0.0363
	Drive	0.0321	0.0082	0.0441
	Entry	0.0341	0.0316	0.0244
	Midcorner	0.0412	0.0148	0.0550
	Exit	0.0451	0.0091	0.0569
	Cornering	0.0412	0.0218	0.0464
Image agent	Full lap	0.0316	0.0223	0.0337
	Brake	0.0321	0.0327	0.0234
	Drive	0.0298	0.0083	0.0398
	Entry	0.0331	0.0313	0.0095
	Midcorner	0.0409	0.0138	0.0388
	Exit	0.0385	0.0076	0.0531
	Cornering	0.0320	0.0167	0.0315

TABLE VII
EVALUATION TRACK C FOR 12 LAPS ON SPECIFIED GATES

Model	Gate	RMSE		
		Steering	Braking	Accelerating
BP agent	Full lap	0.098	0.062	0.129
	Brake	0.080	0.104	0.156
	Drive	0.093	0.019	0.115
	Entry	0.065	0.115	0.131
	Midcorner	0.204	0.050	0.199
	Exit	0.109	0.023	0.136
	Cornering	0.138	0.095	0.162
Image agent	Full lap	0.037	0.046	0.042
	Brake	0.042	0.079	0.041
	Drive	0.033	0.009	0.047
	Entry	0.041	0.084	0.018
	Midcorner	0.063	0.024	0.062
	Exit	0.050	0.007	0.050
	Cornering	0.051	0.067	0.041

lags behind with 0.0464. This indicates that the image agent provides more consistent and accurate predictions during acceleration transitions, particularly in challenging sections of the track.

In conclusion, the image agent generally outperforms the BP agent, particularly in critical driving actions like steering, braking, and acceleration, which are essential for maintaining smooth and safe driving trajectories. The results suggest that the design choices in image agent allow it to handle complex driving scenarios better, especially when it comes to accurately predicting actions during cornering and braking. The BP agent, while competitive in some acceleration transitions, is ultimately surpassed by image agent in overall performance.

C.2: Additional Performance Results BP agent & Image agent on testing data

In Table VII, I provide an additional performance comparison between the BP agent (consisting of Boundary Points, vehicle states, and path planning features) and image agent (consisting of images, vehicle states, and path planning features), tested over 12 laps on track C. The Root Mean Square Error (RMSE) was measured for steering, braking, and acceleration actions across various driving gates, including the full lap, 'brake' , 'drive', 'entry' , 'midcorner', 'exit', and 'cornering' . The results demonstrate that image agent consistently achieves lower RMSE across all gates, particularly excelling in 'brake' and 'drive', indicating superior generalization and decision-making capabilities compared to the BP agent

APPENDIX D: ARCHITECTURAL COMPARISONS OF CNN MODELS

Architectural Options Explored

In the initial stages of model selection, three CNN architectures were evaluated for their ability to generalize to unseen tracks: **AlexNet**, **Nature CNN**, and **Nature CNN with an Additional Layer**.

- **AlexNet**:
  - Designed for image classification tasks, AlexNet features deep convolutional layers capable of extracting complex features. However, its computational demands and propensity for overfitting made it less suitable for this task.
  - *Strengths:* Robust feature extraction for high-dimensional inputs.
  - *Weaknesses:* Inefficient for reinforcement learning (needs to be future-proof), overfits easily, and has high computational costs.
- **Nature CNN**:
  - A lightweight architecture commonly used in reinforcement learning tasks, Nature CNN focuses on simplicity and efficiency, making it well-suited for generalization tasks in motorsport simulations.
  - *Strengths:* Balances computational efficiency with generalization performance.
  - *Weaknesses:* Limited capacity for highly complex image features compared to deeper architectures.
- **Nature CNN with an additional layer**:
  - By adding an extra convolutional or dense layer, this variant enhances the capacity to model nuanced spatial and contextual information. However, the added complexity increases the risk of overfitting.
  - *Strengths:* Improved feature extraction and capacity for complex tasks.
  - *Weaknesses:* Higher computational demands and greater risk of overfitting, particularly with limited training data.

TABLE VIII
TRAINING LOSS, VALIDATION LOSS, AND GENERALIZATION GAP FOR EACH ARCHITECTURE (LOSS IN MSE)

Architecture	Training Loss	Validation Loss
AlexNet	0.00062	0.0105
Nature CNN	0.00082	**0.0034**
Nature CNN (+ 1)	0.00078	0.0093

Results and Selection

As can be seen in Table VIII, testing revealed that **AlexNet**, while achieving the lowest training loss (0.0062), indicated significant overfitting to the training data. Its computational inefficiency and lack of suitability for reinforcement learning further reduced its viability.

Nature CNN, by contrast, demonstrated the best overall balance between generalization and efficiency, with a training loss of 0.0082 and a validation loss of 0.0034. This resulted in the smallest generalization gap, reflecting robust performance on unseen tracks. Its lightweight architecture and scalability for reinforcement learning made it the optimal choice.

Nature CNN with an Additional Layer showed slightly improved training loss (0.0078), but the increased complexity introduced a higher validation loss (0.0093), making it less efficient for generalization tasks.

Conclusion

Based on these results, the **Nature CNN** was selected as the final architecture for its ability to generalize effectively while maintaining computational efficiency. This architecture balances simplicity with robust performance, making it well-suited for real-time simulations and reinforcement learning extensions in motorsport driver modeling.

Effects of MaxPooling and Batch Normalization

To evaluate the impact of architectural enhancements, variations of the Nature CNN were tested with and without **MaxPooling** and **Batch Normalization (BatchNorm)**. These techniques are commonly used to improve feature extraction, stabilization during training, and generalization performance.

a) MaxPooling: MaxPooling layers reduce the spatial dimensions of feature maps by selecting the maximum value within a region. This operation emphasizes dominant features, reduces computational load, and introduces invariance to small translations in the input. However, excessive pooling can lead to the loss of fine-grained details, which are critical in motorsport simulations where precise spatial information is required.

b) Batch Normalization: BatchNorm normalizes the inputs to each layer by adjusting the mean and variance, reducing internal covariate shift. This improves training stability, allows for higher learning rates, and serves as an implicit regularizer, mitigating overfitting. By normalizing intermediate feature distributions, BatchNorm accelerates convergence and enhances generalization.

TABLE IX
VALIDATION LOSS FOR NATURE CNN VARIANTS WITH MAXPOOLING AND BATCHNORM (LOSS IN MSE)

Variant	MaxPooling	BatchNorm	Val Loss
Baseline	None	None	0.0034
MaxPooling Only	Yes	None	0.003
BatchNorm Only	None	Yes	0.0021
MaxPooling + BatchNorm	Yes	Yes	**0.0019**

Results and Selection

Testing revealed the following trends (as seen in Table IX):

- **MaxPooling Only:** Adding MaxPooling improved validation loss from 0.0034 (baseline) to 0.003 by emphasizing dominant features and reducing computational load. However, the lack of stabilization led to less efficient generalization compared to BatchNorm.
- **BatchNorm Only:** BatchNorm significantly reduced validation loss to 0.0021 by normalizing intermediate activations, improving training stability and generalization.
- **MaxPooling + BatchNorm:** Combining MaxPooling and BatchNorm achieved the best validation loss (0.0019). This configuration benefited from efficient feature extraction and training stabilization, minimizing overfitting while retaining fine-grained spatial details.

Conclusion

The combination of **MaxPooling** and **BatchNorm** was selected as the final configuration for the Nature CNN. This setup ensures robust feature extraction and training stabilization, achieving optimal generalization for motorsport simulations while maintaining computational efficiency.