IEEE Copyright Notice

Learned Fourier Bases for Deep Set Feature Extractors in Automotive Reinforcement Learning

Maximilian Schier[1], Christoph Reinders[1] and Bodo Rosenhahn[1]

***Abstract*— Neural networks in the automotive sector commonly have to process varying number of objects per observation. Deep Sets feature extractors have shown great success on problems in reinforcement learning with dynamic observations, achieving state-of-the-art performance on common tasks like highway driving. However, recent work has shown that neural networks suffer from a spectral bias, fitting to low frequencies of scalar features used in such representations, like velocities and distances. We introduce a novel set feature extractor combining learned Fourier features with the Deep Sets architecture. The proposed architecture reduces this spectral bias allowing more sample-efficient training and better performance by learning a more detailed representation. Extensive experiments are conducted on three different environments, including two novel environments featuring a large number of objects in challenging random scenarios. Our method outperforms state-of-the-art approaches and exceeds the performance of existing Deep Sets architectures on these challenging new tasks.**

I. Introduction

Reinforcement Learning has enjoyed growing success in many contexts, including automotive [7], [8], [26]. Recent work has primarily focused on suitable representations for scenes with a variable number of objects or measurements, usually referred to as dynamic scenes. For such environments, set and graph representations have shown enormous potential, often outperforming static flat representations as well as occupancy grids or other spatial representations [6], [7]. Set architectures like Deep Sets [28] are simpler than most graph neural networks (GNNs), but have been found to perform as well as graphs on some automotive problems [8]. Their simplicity is a key advantage as previous studies have found the computational overhead of GNNs to be excessively large with full connectivity [8], [6], [22]. In this work, we focus on improving the Deep Sets architecture for automotive RL. Previous work from the computer vision community has shown a spectral bias in neural networks towards low frequency components and suggested Fourier bases as a remedy [24], [23]. Applying the same architectural changes to continuous robot control problems has improved performance for static representations in reinforcement learning [1], [27]. Set-based representations usually encode the scene as a set of statically sized feature vectors, containing scalar features such as distances or velocities. The Deep Sets architectures used in previous research in the automotive community [7], [8] use multilayer perceptrons to encode the objects composed of such features, thus exhibiting the same spectral bias. In this work, we propose a novel feature extractor for dynamic object representations combining the Deep Sets architecture with a suitable learned Fourier basis to reduce bias and improve the locality and, thus, attention of the Deep Sets encoder. We perform extensive experiments on the well known *highway-env* [13] environment as well as a challenging novel environments covering precise continuous vehicle control while parking and navigating a tightly winding road. To summarize, our main contributions are:

- We present a novel feature extractor architecture for dynamic scenes combining Deep Sets and learnable Fourier bases.
- In addition, we present novel benchmark environments for precise continuous vehicle control based on high-level object detections.
- Our proposed approach outperforms existing Deep Sets architectures, as well as static flat and bird's-eye CNN encodings.
- Upon acceptance, we publish our implementation of our agent, all comparative agents, and the novel environments at `https://github.com/m-schier/LFF-DS`.

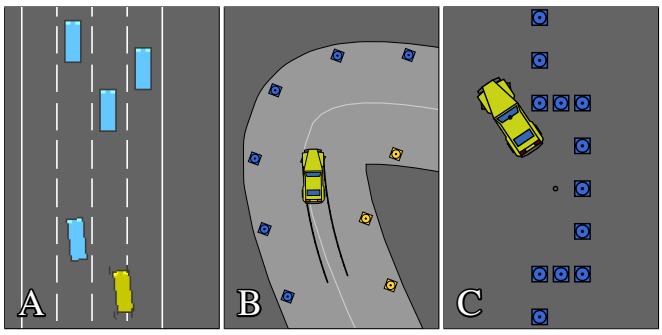


Fig. 1: Reinforcement learning on variable-sized high-level object perceptions is still a difficult task. Our proposed architecture using a Deep Sets feature extractor on a learned Fourier bases significantly improves sampling efficiency, feature representation, and final performance. We evaluate our method on three tasks: an established highway scenario (A) and two novel simulation environments for navigating a winding country road (B) and reverse parking (C).

II. Related Work

Dynamic scene representations have seen some popularity in automotive reinforcement learning. Most commonly used are graphs and sets. Early work applied the Deep Sets architecture [28] in the automotive context to control high-level discrete lane change actions in a highway scenario [7].

[1]Institute for Information Processing, L3S - Leibniz University Hannover, {schier,reinders,rosenhahn}@tnt.uni-hannover.de

Hügle et. al. also analyze graph neural networks (GNNs) for the same problem. The performance of set- and graph-based methods was found to be comparable on the highway scenario [8]. With the former methods using deep Q-networks (DQNs), Hart and Knoll have suggested using proximal policy optimization on graphs for this problem [6]. GNNs have also been proposed in the context of intersection navigation to control longitudinal dynamics [22]. Klimke *et al.* [10], [11] have applied GNNs to cooperative intersection management and discovered that applying transformations to specific features, such as the inverse, may improve performance.

Another popular scene representation which is static but may still encode a variable number of scene objects are occupancy grids. 2D occupancy grids have been applied to highway lane change decisions [25] and 1D path segment grids to intersection navigation [12].

Various automotive simulation environments have been used by previous work. The SUMO simulator [16] can simulate complicated urban traffic flow between various traffic participants. Lateral vehicle control is limited to discrete lane change actions. The *highway-env* [13] simulator allows continuous control in a highway scenario with other traffic participants. Works on autonomous racing [26], [9] have used commercial video games with high graphics and vehicle physics fidelity but no publicly accessible interfaces. TORCS is a popular open-source 3D racing simulator [17], [20] which can provide camera observations, though not photo-realistic, or track curvature. CARLA [4] is a 3D urban traffic simulator focusing on decision making based on low-level sensor data. Finally, Trackmania is another commercial simulator for which a public interface exists [19], but the problem setting is purely virtual.

III. Approach

In this section, the problem formulation using a Markov Decision Process is given and the concepts of learned Fourier features and Deep Sets are introduced. Finally, our novel architecture combining the advantages of the previous architectures is introduced.

A. Markov Decision Process

The problem of vehicle control is formulated as a Markov-Decision Process. The agent must control the vehicle behavior in accordance with its partial observation of the environment. At each discrete time step of the simulation, the agent chooses an action based on the current observation and receives a reward based on the environment state and selected action. Such an MDP is commonly described by the tuple $(\mathcal{S}, \mathcal{A}, \mathcal{T}, r, \gamma)$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ the action space, $\mathcal{T} : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$ the transition function, $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ the reward function, and γ the discount factor which reduces future rewards. An agent's policy π determines the probability of selecting an action given the current states, $\pi : \mathcal{S} \times \mathcal{A} \to \{0, 1\}$. The expected total reward G starting from some initial state is the decaying sum of the reward of current time step and all future time steps

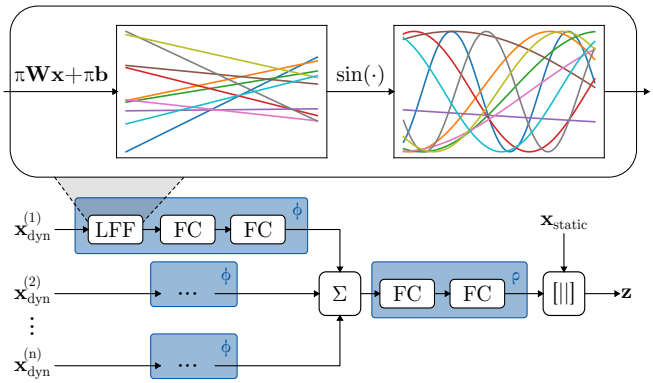


Fig. 2: Our proposed novel feature extractor combines Deep Sets and learned Fourier features (LFF) enabling efficient training on dynamic scene representations. A variable number of object observations $\mathbf{x}^{(1)}_{\text{dyn}}, \ldots, \mathbf{x}^{(n)}_{\text{dyn}}$ are encoded using the shared item encoder ϕ. Critically, we propose a learned Fourier feature as the first layer in the item encoder to reduce spectral bias through learnable frequencies. The feature-wise summation of all encoded items is transformed by the multilayer perceptron ρ and concatenated with static features $\mathbf{x}_{\text{static}}$. The output $\mathbf{z}$ is used by the policy- or Q-network.

t: $G = \sum_t \mathrm{E}[\gamma^t r(s_t, a_t)]$, where each a_t is chosen by π. If selecting an arbitrary action a in the first time step this reward is also called the Q-value $Q(s, a)$ of action a from the current state s. The objective of the agent is maximizing G, which is accomplished by selecting the action with largest Q-value. To determine the Q-value $Q^*(s, a)$ following an optimal policy for current state s, let us assume that the Q-values $Q^*(s', a')$ of the optimal policy for the next timestep are known with $s' = \mathcal{T}(s, a)$, $a' \in \mathcal{A}$. The optimal Q-value for the current state s and some action a is then given by:

$$Q^*(s, a) = r(s, a) + \gamma \max_{a' \in \mathcal{A}} Q^*(s', a'). \tag{1}$$

In deep Q-Learning, a neural network Q_θ is used to approximate the optimal policy Q^*, where θ are learnable parameters. These are iteratively trained using the mean squared Bellman error loss:

$$\mathcal{L} = (r + \gamma \max_{a' \in \mathcal{A}} Q_\theta(s', a') - Q_\theta(s, a))^2 \tag{2}$$

which minimizes the difference between predicted Q-value and the sum of immediate reward and discounted future predicted reward. The best action according to the learned policy π_θ can then be determined for a state s by $\operatorname{argmax}_{a \in \mathcal{A}} Q_\theta(s, a)$.

B. Proposed Architecture

To extract features from a dynamic scene with a set representation, Deep Sets [28] are commonly used. The Deep Sets architecture allows permutation-invariant encoding of a dynamically sized set of statically sized vectors. Deep Sets uses a permutation-invariant pooling operation. In the context of this work, we will use feature-wise summation. Given the two neural networks $\phi : \mathbb{R}^u \to \mathbb{R}^v$ and $\rho : \mathbb{R}^v \to \mathbb{R}^w$, the

Environment	Realistic setting	View	Pure Python	Open source	Physics quality	Object perceptions	Random tracks
TORCS [17]	✓	3D	✗	✓	+	✗	✗
Trackmania (tmrl) [19]	✗	3D	✗	✗	o	✗	✗
WRC6 [9]	✓	3D	✗	✗	++	✗	✗
Car-Racing [2]	✓	2D	✓	✓	o	✗	✓
highway-env (Racetrack) [13]	✓	2D	✓	✓	o	✓	✗
Racecar Gym [3]	✓	3D	✓	✓	o	✗	✗
Ours	✓	2D	✓	✓	o	✓	✓

TABLE I: Qualitative comparison of our new driving environment with various road driving, racing, and rally environments used in previous work. Our environment is fully open source and purely written in python code making it easy to integrate. It provides object detections and offers a sufficient level of vehicle physics where the vehicle can skid, understeer, etc. For training, any number of random road tracks can be generated.

forward pass of Deep Sets on a set $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots\}$, $\mathbf{x}_i \in \mathbb{R}^u$ is then given as:

$$\mathrm{DS}(\mathbf{X}) = \rho\left(\sum_{\mathbf{x}\in\mathbf{X}} \phi(\mathbf{x})\right). \tag{3}$$

The networks ϕ and ρ have previously been commonly implemented as multilayer perceptrons (MLP) [28], [7], [8].

Neural networks, especially the MLP architecture as used in Deep Sets, are known to bias towards learning low frequencies of a low dimensional input [24], [27], [14]. Therefore, learned Fourier features were proposed, which transforms or *lifts* the input into a different harmonics basis. This work uses the learned Fourier feature (LFF) definition introduced by [27], which proposed using a learned Fourier basis to improve performance on continuous control tasks with statically sized input features. The LFF is a network layer transforming an input vector $\mathbf{x} \in \mathbb{R}^d$ into the output domain $\mathbb{R}^o$. Its forward pass is defined as:

$$\mathrm{LFF}(\mathbf{x})_j = \sin\left(\sum_{i=1}^{d} \pi \mathbf{W}_{j,i}\mathbf{x}_i + \pi \mathbf{b}_j\right). \tag{4}$$

The trainable parameters $\mathbf{W} \in \mathbb{R}^{o\times d}$ and $\mathbf{b} \in \mathbb{R}^o$ are initialized as follows:

$$\mathbf{W}_{j,i} \sim \mathcal{N}(0, b_{\mathrm{cut}}/d),\ \mathbf{b}_j \sim \mathcal{U}(-1, 1), \tag{5}$$

where b_{cut} is the tunable cutoff frequency.

We propose to integrate a learned Fourier features layer as part of the item encoder $\phi(\cdot)$ of the Deep Sets feature extractor by substituting the first fully connected (FC) layer. Thus, we extent the advantages of learned Fourier features from statically sized observation spaces to dynamically sized ones. Our proposed architecture is shown in Figure 2. This modification allows the item encoder $\phi(\cdot)$ to more accurately estimate high frequency components of the scalar observation features, e.g. small changes in distance compared to the maximum range of distances. Such a hypothesis is well founded as it is known that replacing the first FC layer of an MLP with an LFF reduces this spectral bias [27], [14], [24]. The Deep Sets item encoder $\phi(\cdot)$ must learn a transformation that allows reconstructing the input distribution as accurately as required for the problem from the feature-wise aggregation. We postulate that the LFF helps the item encoder to more accurately activate on ranges of the input space, thus reducing cross-talk from different encoded items on the same feature channels. Effectively, the encoding using an LFF aids the Deep Sets architecture to learn a representation comparable to a histogram of varying granularity. We will analyze the learned representation later in the experiments.

IV. Environments

We evaluate our proposed method on multiple RL environments representing different automotive problems, which are briefly presented in this section.

A. Highway Driving

The well established *highway-env* [13] environment has been used in previous works as a simulator for highway traffic. The objective is navigating a 4-lane highway through traffic as quickly as possible, overtaking other vehicles. Static input is $\mathbf{x}_{\mathrm{static}} = (x, y, v_x, v_y)$ with (x, y) as the cartesian position of the ego vehicle and (v_x, v_y) the velocity vector in the global coordinate frame. Dynamic features per other vehicle are $\mathbf{x}_{\mathrm{dyn}} = (\Delta x, \Delta y, \Delta v_x, \Delta v_y)$ with $(\Delta x, \Delta y)$ as the relative position to the ego vehicle in the global frame and $(\Delta v_x, \Delta v_y)$ as the relative velocity vector to the ego vehicle in the global frame. The reward $r(s, a)$ is given by:

$$r(s, a) = -\mathbb{1}_{\mathrm{fail}} + 0.4 \cdot \frac{v_x}{v_{\max}} \tag{6}$$

with v_x as the current velocity in forward direction of the road, $v_{\max}$ the maximum speed, and $\mathbb{1}_{\mathrm{fail}}$ indicating collision with another vehicle or the road boundary.

B. Country Road Driving

We present a novel 2D top-down environment that simulates driving on a winding road with challenging surface conditions. The environment features randomly generated road tracks. Exemplary generated layouts are shown in Figure 3. The road boundary can be perceived by the agent only through a set of boundary markers, see Fig. 1B. The objective of the agent is travelling a maximum distance along the road within a specified time (in these experiments 60 s) while not leaving the road area. A qualitative comparison with existing environments is given in Table I. Our motivation for implementing a new 2D environment is the lack of easy-to-use environments with dynamically sized object

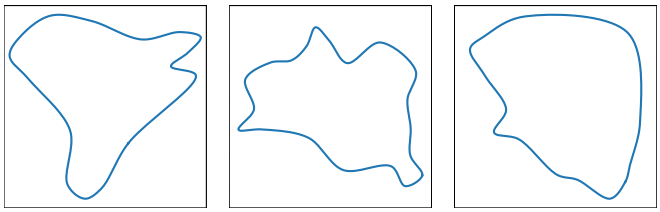


Fig. 3: Our simulator environment can generate random road tracks during training. Tracks are approximately $500\,\mathrm{m}$ long. Three examples are shown.

observations. In addition, our environment generates random scenarios making the task more challenging and encouraging the learning of generalizing behavior. The vehicle physics are based on a dynamic bicycle model with a maximum traction limit according to the circle of forces, thus the vehicle may understeer, oversteer, and skid if excessively braking, accelerating, and steering. As such the control task is more complex than using a kinematic model. The environment is purely written in Python and requires no external simulators. It can thus be easily used and still achieves high frame rates of 200-300 fps on a modern computer using a single CPU core. The random track generation allows for good generalization to arbitrary road layouts.

Static input is $\mathbf{x}_{\text{static}} = (v_x, v_y, \omega, \beta)$ with the ego vehicle's velocity vector (v_x, v_y) in vehicle reference frame, steering angle β, and angular yaw velocity ω. Dynamic observations are high-level detections of the boundary traffic cones $\mathbf{x}_{\text{dyn}} = (x, y, \mathbb{1}_{\text{left}}, \mathbb{1}_{\text{right}})$, where (x, y) is the position in the ego vehicle local reference frame and $\mathbb{1}_{\text{left}}$ and $\mathbb{1}_{\text{right}}$ indicate that the traffic cone color matches that of the left-side or right-side boundary cone, respectively.

The reward function is designed to punish collisions with the track boundary objects ($\mathbb{1}_{\text{collision}}$) and leaving the track with the center of mass ($\mathbb{1}_{\text{fail}}$), which also terminates the episodes. Additionally, the agent receives positive reward for the scalar projection of its velocity vector $\mathbf{v}_{\text{ego}}$ onto the forward track direction $\mathbf{n}_{\text{track}}$ to encourage fast driving:

$$r(s,a) = -\mathbb{1}_{\text{fail}} - 0.2 \cdot \mathbb{1}_{\text{collision}} + 0.01 \cdot \mathbf{n}_{\text{track}} \cdot \mathbf{v}_{\text{ego}}. \quad (7)$$

The coefficients of the reward terms were determined empirically. Previous work on rally driving has included terms for the distance to the road center line [9], [17]. We omit such terms to keep prior knowledge minimal.

C. Parking

Additionally, we also present a new parallel parking problem in our environment. The ego vehicle starts in a random pose in front of a parallel parking spot and must precisely park backwards. The parking spot is marked by traffic cones in our test, but these object perceptions can stand in for any objects that might realistically define a parking spot, such as points on an outline. Episodes are truncated after $15\,\mathrm{s}$ unless a collision with the boundary happens, in which case the episode terminates. The environment is perceived like $\mathbf{x}_{\text{dyn}}$ on *Country Road*. The ego state is $\mathbf{x}_{\text{static}} = (v_x, \beta)$ perceiving longitudinal velocity v_x and steering angle β. This is sufficient as a kinematic model is used to simulate the vehicle motion at the low speeds of parking.

The reward for this problem encourages quickly aligning the vehicle with the target parking pose, which is positioned in the center of the spot in direction of the road. Let (x, y, α) be the pose of the vehicle and (u, v, κ) the target pose. The reward r is given by:

$$r(s,a) = -\mathbb{1}_{\text{collision}} + c \cdot r_{\text{pose}}(s,a), \text{ with} \quad (8)$$

$$r_{\text{pose}}(s,a) = \max\left(0, 1 - \frac{|x-u|}{x_{\max}}\right) \quad (9)$$

$$\cdot \max\left(0, 1 - \frac{|y-v|}{y_{\max}}\right) \cdot \max\left(0, \cos(\alpha - \kappa)\right).$$

We empirically set $c = 0.05$, $x_{\max} = 40\,\mathrm{m}$, $y_{\max} = 5\,\mathrm{m}$. This objective jointly rewards all aspects of the pose while also punishing collisions with the boundary through $\mathbb{1}_{\text{collision}}$. Since a good pose is continuously rewarded, the agent is encouraged to arrive quickly.

D. Actions

Action spaces on all environments are continuous. The agent controls the steering angle on all environments, in our environment steering rate is limited to $60\,^\circ/\mathrm{s}$, on *Highway* any steering angle can immediately be applied. Longitudinal motion on *Highway* and *Parking* is controlled through a single continuous action in range [-1, 1] from maximum deceleration to maximum acceleration. On *Country Road* two continuous actions in range [0, 1] control the throttle and brake independently. The agent interacts with all environments at a rate of $10\,\mathrm{s}^{-1}$, which is considered to be a good trade-off between performance and training efficiency in challenging state-of-the-art applications [26].

V. Experiments

In this section, we briefly describe the training setup and evaluate the performance of LFF-DS in comparison to regular Deep Sets and other popular representations. We analyze the influence of different Fourier features, the robustness to hyperparameter changes and we visualize the finer apprehension of object positions through our suggested architecture.

A. Experimental Setup

We use a Soft Actor-Critic (SAC) [5] agent for all experiments, which is the selected architecture for most of the state-of-the-art methods in continuous action automotive RL [26], [21]. We train for $1\,000\,000$ environment steps, and carry out an equal amount of gradient steps. We set the buffer size to $1\,000\,000$, learning rate to $3 \cdot 10^{-4}$, and discount factor $\gamma = 0.95$. As suggested by previous work [20] Ornstein-Uhlenbeck noise [15] is applied to the actions on the *Country Road* task where the agent controls throttle and brake independently as we have found it to improve the training for all tested methods. The parameters of the noise are set to $\mu = 0$, $\sigma = 0.2$, $\theta = 0.15$, $\Delta t = 0.1$. Ten agents are trained per environment and configuration. For

Agent	Highway		Country Road		Parking	
	Reward ↑	CI95	Reward ↑	CI95	Reward ↑	CI95
CNN	40.74	[29.36, 42.82]	76.89	[38.66, 80.97]	1.84	[-0.82, 5.17]
Flat	12.32	[11.31, 13.37]	23.31	[16.16, 25.77]	5.45	[5.25, 5.68]
DS	**46.91**	[45.23, 48.61]	<u>83.08</u>	[81.88, 83.79]	<u>5.83</u>	[5.62, 6.09]
LFF-DS (**ours**)	<u>46.62</u>	[45.71, 48.50]	**88.72**	[85.62, 91.49]	**6.46**	[6.38, 6.53]

TABLE II: Interquartile mean of reward over ten training runs. Best results are highlighted in bold, second best are underlined. Deep Sets based architectures (DS, LFF-DS) outperform CNNs and flat encodings. Our proposed learned Fourier features item encoder (LFF-DS) further enhances performance significantly on the *Country Road* and *Parking* scenarios.

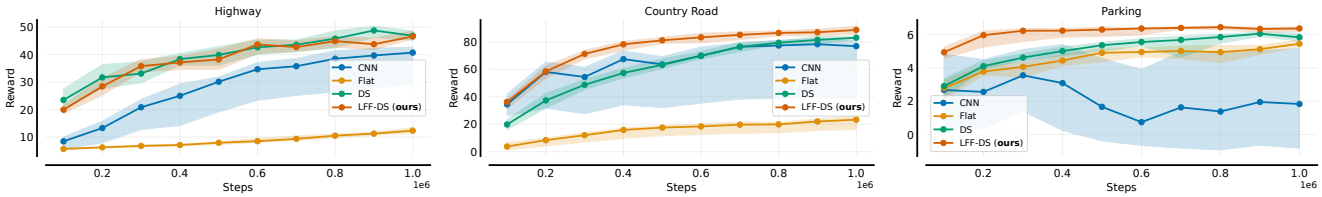


Fig. 4: Interquartile mean of reward on all environments during training as well as 95% confidence intervals as shaded region. Our proposed architecture LFF-DS achieves significantly higher reward during most points in training compared to regular Deep Sets on *Country Road* and *Parking*. On *highway*, differences between both are present but not statistically significant with $p < 0.05$. CNN and flat encoding perform significantly worse than LFF-DS at allmost all time steps.

evaluation, we report the interquartile mean (IQM) and estimate confidence intervals (CI) using stratified bootstrapping with 50 000 iterations according to best practice in the RL community [1]. We compare the following methods:

- A regular Deep Sets feature extractor, serving as an ablation baseline (*DS*)
- A CNN feature extractor operating on an occupancy grid encoding of the set items (*CNN*)
- An MLP feature extractor operating on a flattened version of the set items with a presence marker per item (*Flat*)
- Our proposed architecture integrating an LFF with the item encoder of a Deep Sets feature extractor (*LFF-DS*)

We evaluate all agents on three environments described in Section IV. All features are normalized by division with the maximum observable values. *LFF-DS* and *DS* share the same architecture, with just the first layer being an FC instead of an LFF layer, thus the number of parameters is equal. The exact configurations of all architectures are provided in Table III. For LFF-DS we set $b_{\text{cut}} = 30$ on *Country Road* and $b_{\text{cut}} = 3$ on *Highway*. For the CNN, we base the feature extractor for *Highway* on previous work performing an architecture search on three-lane highway lane change decisions [7]. On *Country Road*, the standard CNN feature extractor for the ALE benchmark is used [18], as the representation is comparable in width and height. The Deep Sets and flat architectures were determined using a random search with 20 tries. We only carried out an architecture search on the *Country Road* and *Highway* environment. On the *Parking* environment, we use the same architecture and hyperparameters as on *Country Road*. This is motivated by the same dynamic scene features being used in both environments, thus an architecture that generalizes well should also perform well on *Parking*.

Country Road and *Parking*

LFF-DS	DS
ϕ : LFF(64), FC(32), FC(128) ρ : FC(128), FC(128)	ϕ : FC(64), FC(32), FC(128) ρ : FC(128), FC(128)
Flat	**CNN**
FC(512) FC(256)	Conv2D(32, 8×8, 4×4) Conv2D(64, 4×4, 2×2) Conv2D(64, 3×3, 1×1) Flatten(), FC(256)

Highway

LFF-DS	DS
ϕ : LFF(128), FC(64), FC(256) ρ : FC(256), FC(256)	ϕ : FC(128), FC(64), FC(256) ρ : FC(256), FC(256)
Flat	**CNN**
FC(256) FC(256)	Conv2D(32, 1×7, 1×3) Conv2D(16, 1×7, 1×3) Conv2D(16, 2×3, 1×2) Flatten(), FC(256)

TABLE III: Overview of the network architectures. Learned Fourier features (LFF) and fully connected layers (FC) with respective output units. Conv2D with output units, kernel size, and stride, all using zero padding. All layers but LFF use ReLU activations.

B. Comparison with State-of-the-Art

We train all agents on the individual environments using ten seeds. During training, we evaluate the performance at time steps of 100 000. The results are shown in Table II. In final performance, LFF-DS achieves a 5.64 points (6.8 %) higher reward on *Country Road* and 0.63 points (10.8%) higher reward on *Parking* compared to the next best feature extractor architecture, regular Deep Sets (DS). These results are significant with $p < 0.05$. On *Highway*, LFF-DS and

DS achieve similarly good results with a reward of 46.62 and 46.91, respectively. We attribute similar performance to the low number of objects in this scenario and the lack of need for very accurate observations to solve this problem. On *Highway* and *Country Road*, the performance of the CNN and flat encoding is significantly lower than both LFF-DS and DS confirming the advantage of Deep Sets based feature encoders for dynamically sized scenes found in previous work [7]. On *Parking*, our new LFF-DS architecture is significantly better than all methods including the flat encoding, which regular Deep Sets do not outperform significantly. Interestingly, the CNN performs worse than flat encoding on *Parking*, whereas it performs much better than flat encoding on *Highway* and *Country Road*. We attribute this to the sensitivity of CNNs on the grid size depending on the problem, even if input representation remains unchanged, and the sensitivity of the flat encoding to an advantageous stable sorting order depending on the problem.

In the next experiment, we analyze the performance over training time. The results are presented in Figure 4. Our proposed LFF-DS architecture performs significantly better than the compared architectures during most parts of the training, with the exception of the *Highway* environment, where performance is comparable to regular Deep Sets. On the *Country Road* environment, our proposed architecture outperforms the regular Deep Sets encoder and a flat encoding at all time steps and the CNN at all time steps after 200 000. These results are significant with $p < 0.05$. On the *Parking* environment, our proposed architecture achieves significantly higher reward than the compared methods at all points in training but 900 000 steps. Our proposed method is very sampling efficient, exceeding the best performance of all other methods after only 200 000 steps on *Parking*.

On *Country Road*, the individual objectives of the reward function, driving quickly and safe, are evaluated during training, see Figure 5. From 200 000 training steps onwards LFF-DS exceeds all other tested architectures in both objectives. LFF-DS has a significantly higher mean velocity than DS during all steps of the training. Comparing failure rates - the rate of leaving the road - shows that LFF-DS is much more sampling efficient than DS for determining a safe policy, failing less often at 200 000 and 400 000 steps than DS at 400 000 and 700 000 steps respectively. We present a video of the final performance of all agents on *Country Road*[1].

Furthermore, on *Parking*, we compare the accuracy of the final vehicle pose between DS and our proposed feature extractor, LFF-DS, in Table IV. As shown in previous results (Table II), LFF-DS achieved a higher reward than DS on this problem. The position error in both lateral and longitudinal direction is significantly lower with 0.14 m and 0.08 m compared to 0.76 m and 0.41 m, in relative terms 82% better and 80% better. The improvement is significant with $p < 0.05$. The heading error is insignificantly larger for LFF-DS, while the reward, which combines the individual objectives, is significantly better. Thus, using our new feature extractor has improved the accuracy but also slightly shifted the balancing of the individual terms of the multi-objective optimization that is the compromise between longitudinal, lateral, and heading error.

[1] https://www.tnt.uni-hannover.de/en/staff/schier/LFF-DS/CountryRoad.mp4

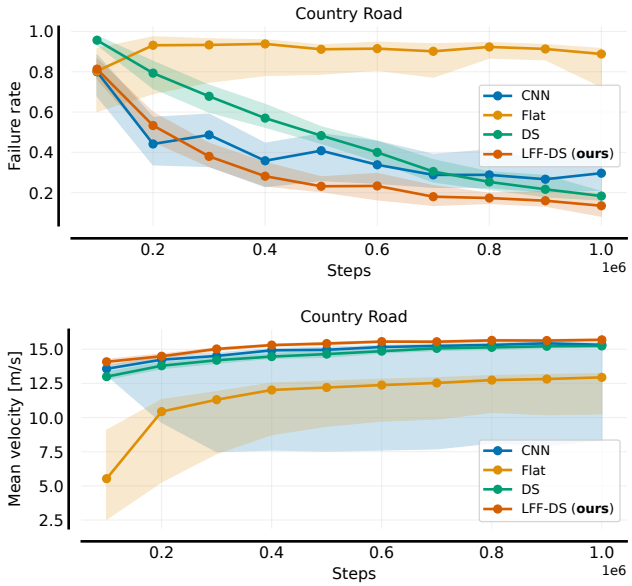


Fig. 5: Interquartile mean of failure rate and mean vehicle velocity over ten training runs on *Country Road*. Shaded areas show 95% confidence interval. Our proposed agent performs better by driving faster than the competitors, while failing at less episodes through leaving the road.

Metric	DS		LFF-DS (**ours**)	
$\Delta x \downarrow$ [m]	0.76	[0.56, 1.03]	**0.14**	[0.08, 0.20]
$\Delta y \downarrow$ [m]	0.41	[0.27, 0.71]	**0.08**	[0.06, 0.11]
$\Delta\omega \downarrow$ [°]	3.81	[2.59, 4.97]	4.84	[3.34, 6.18]
$r_{\text{pose}} \uparrow$	0.875	[0.820,0.904]	**0.972**	[0.964,0.977]

TABLE IV: Quality of final vehicle pose on the *Parking* problem using our proposed architecture and the baseline Deep Sets feature extractor. 95% confidence interval reported in brackets, significant improvements in bold. Our architecture achieves a significantly lower longitudinal (Δx) and lateral (Δy) position error as well as better final reward r_{pose}. Heading error $\Delta\omega$ is not significantly larger.

C. Analyses

Our proposed feature extractor introduces one additional hyperparameter b_{cut} compared to regular Deep Sets, which governs the sensitivity to different parts of the spectrum of each encoded item. In the first experiment, the influence and robustness of the hyperparameter b_{cut} are evaluated. We train on the *Country Road* and *Parking* scenario using a wide range of values in steps of roughly half powers of ten. The results are shown in Figure 6. First, on both environments values ranging from 1 to 30 all produce comparable results to the best performing, which we consider a fairly broad

Agent	Highway		Country Road		Parking	
	Reward ↑	CI95	Reward ↑	CI95	Reward ↑	CI95
DS	**46.91**	[45.23, 48.61]	83.08	[81.88, 83.79]	5.83	[5.62, 6.09]
Positional+DS	46.55	[46.20, 46.90]	<u>88.03</u>	[86.34, 89.65]	<u>6.43</u>	[6.14, 6.48]
SIREN+DS	10.49	[8.68, 11.53]	79.13	[77.42, 81.86]	6.19	[5.75, 6.30]
LFF-DS (**ours**)	<u>46.62</u>	[45.71, 48.50]	**88.72**	[85.62, 91.49]	**6.46**	[6.38, 6.53]

TABLE V: Analysis of different Fourier features integrated with Deep Sets extractors. Best results are highlighted in bold, second best underlined. SIREN [23] item encoders combined with Deep Sets perform significantly worse than LFF-DS on all tested problems. Static positional encodings [24] combined with Deep Sets perform mostly better than regular Deep Sets, but worse than our proposed approach. We attribute this to the fixed axis alignment of positional encodings.

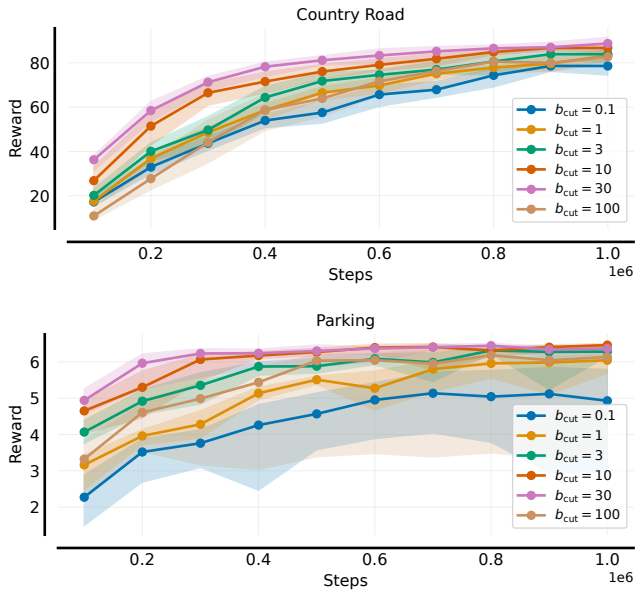
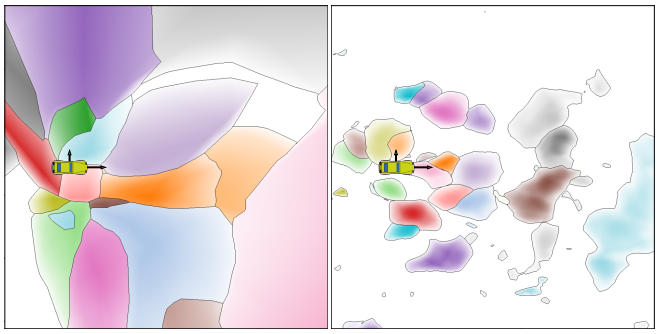


Fig. 6: Influence of the b_{cut} hyperparameter of the learned Fourier features on the performance on *Country Road* and *Parking*. The performance is stable regarding the choice of b_{cut} within a reasonable range. Furthermore, the performance after 10% of steps is a good indicator for final performance, which is useful for hyperparameter optimization.



Deep Sets LFF-DS (**ours**)

Fig. 7: Activation of the item encoder $\phi(\cdot)$ by object position of left-side road boundary on the *Country Road* scenario. Only first 20 features shown, color indicates index of feature with largest activation, intensity magnitude. Vehicle reference frame shown by red arrow (forward) and green arrow (left). Our proposed architecture learns a more local activation allowing more precise reasoning about the observed input distribution after the feature-wise aggregation of all objects in the Deep Sets architecture.

range for stable training, especially considering that the grid size for the CNN has a similar effect for spatial features and its choice must be made carefully. In addition, the order of performance achieved by choice of b_{cut} after 10% of steps is identical to the order at the end of training. This is a useful observation, since it indicates that to carry out a hyperparameter search on b_{cut} it is sufficient to train for at most 10% of the final desired steps.

Next, the choice of Fourier feature for the item encoder is examined. Our proposed method is compared to two other reasonable integrations of Fourier features into Deep Sets. As alternatives, we implement a SIREN network [23] as the item encoder and an architecture were the LFF layer of our proposed method is replaced with a positional encoding [24]. For SIREN, the entire item encoder $\phi(\cdot)$ is build from SIREN layers matching the dimensions of LFF-DS, as the authors suggest building entire networks out of SIREN layers. Results are given in Table V. LFF-DS performs significantly better than SIREN+DS on all tested environments, with SIREN+DS achieving especially low reward on *Highway*. The positional encoding, which is not learnable, performs slightly worse than LFF-DS on all environments. This could be attributed to the positional encodings being strictly axis aligned, whereas LFF-DS learns an affine combination of input features. On *Highway* LFF-DS, DS and Positional+DS perform nearly equally well.

Finally, our claim that the LFF-DS feature extractor allows for a more fine-grained policy to be learned is examined. A fine grid of objects in the vehicle reference frame is processed using item encoders $\phi(\cdot)$ from both LFF-DS and DS trained on *Country Road*. At each encoded position, the index and magnitude of the output feature with highest activation are visualized, see Figure 7. Only the first 20 features are shown for clarity and comprehensibility. Features on DS are activated for much larger subspaces in the input space of objects compared to LFF-DS. Thus, indeed the integration of Fourier features makes activations more local on the feature space and prevents undesired cross-talk between large regions of the sensory range.

VI. Conclusions

In this work, we presented a novel model architecture for reinforcement learning on dynamic scene representations in the form of sets. By integrating Fourier features with Deep Sets we proposed a feature extractor which can both process a dynamically sized observation while maintaining locality of observations, thus reducing feature cross-talk and improving training convergence and performance. Our architecture significantly improves the performance on common automotive tasks. In the future, we want to apply our method to vehicles with joints, such as trucks or cars with trailers. To further improve performance, Fourier features could also be applied to the static parts of the representations, which we omitted here as it was previously suggested [27].

Acknowledgment

This work was supported by the Federal Ministry of Education and Research (BMBF), Germany under the project LeibnizKILabor (grant no. 01DD20003) and the AI service center KISSKI (grant no. 01IS22093C), the Center for Digital Innovations (ZDIN) and the Deutsche Forschungsgemeinschaft (DFG) under Germany's Excellence Strategy within the Cluster of Excellence PhoenixD (EXC 2122).

References

[1] Rishabh Agarwal, Max Schwarzer, Pablo Samuel Castro, Aaron C. Courville, and Marc G. Bellemare. Deep reinforcement learning at the edge of the statistical precipice. In *NeurIPS*, 2021.

[2] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

[3] Axel Brunnbauer and Luigi Berducci. racecar_gym. `https://github.com/axelbr/racecar_gym`, 2018.

[4] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. Carla: An open urban driving simulator. In *Conference on robot learning*, pages 1–16. PMLR, 2017.

[5] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pages 1861–1870. PMLR, 2018.

[6] Patrick Hart and Alois Knoll. Graph neural networks and reinforcement learning for behavior generation in semantic environments. In *2020 IEEE Intelligent Vehicles Symposium (IV)*, pages 1589–1594. IEEE, 2020.

[7] Maria Huegle, Gabriel Kalweit, Branka Mirchevska, Moritz Werling, and Joschka Boedecker. Dynamic input for deep reinforcement learning in autonomous driving. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7566–7573. IEEE, 2019.

[8] Maria Huegle, Gabriel Kalweit, Moritz Werling, and Joschka Boedecker. Dynamic interaction-aware scene understanding for reinforcement learning in autonomous driving. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4329–4335. IEEE, 2020.

[9] Maximilian Jaritz, Raoul De Charette, Marin Toromanoff, Etienne Perot, and Fawzi Nashashibi. End-to-end race driving with deep reinforcement learning. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2070–2075. IEEE, 2018.

[10] Marvin Klimke, Jasper Gerigk, Benjamin Völz, and Michael Buchholz. An enhanced graph representation for machine learning based automatic intersection management. *2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC)*, pages 523–530, 2022.

[11] Marvin Klimke, Benjamin Völz, and Michael Buchholz. Cooperative behavior planning for automated driving using graph neural networks. *2022 IEEE Intelligent Vehicles Symposium (IV)*, pages 167–174, 2022.

[12] Karl Kurzer, Philip Schörner, Alexander Albers, Hauke Thomsen, Karam Daaboul, and J Marius Zöllner. Generalizing decision making for automated driving with an invariant environment representation using deep reinforcement learning. In *2021 IEEE Intelligent Vehicles Symposium (IV)*, pages 994–1000. IEEE, 2021.

[13] Edouard Leurent. An environment for autonomous driving decision-making. `https://github.com/eleurent/highway-env`, 2018.

[14] Alexander Li and Deepak Pathak. Functional regularization for reinforcement learning via learned fourier features. *Advances in Neural Information Processing Systems*, 34:19046–19055, 2021.

[15] Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016.

[16] Pablo Alvarez Lopez, Michael Behrisch, Laura Bieker-Walz, Jakob Erdmann, Yun-Pang Flötteröd, Robert Hilbrich, Leonhard Lücken, Johannes Rummel, Peter Wagner, and Evamarie Wießner. Microscopic traffic simulation using sumo. In *2018 21st international conference on intelligent transportation systems (ITSC)*, pages 2575–2582. IEEE, 2018.

[17] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pages 1928–1937. PMLR, 2016.

[18] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

[19] Simon Ramstedt, Yann Bouteiller, and Edouard Geze. Tmrl. `https://github.com/trackmania-rl/tmrl`, 2021.

[20] Adrian Remonda, Sarah Krebs, Eduardo Enrique Veas, Granit Luzhnica, and Roman Kern. Formula rl: Deep reinforcement learning for autonomous racing using telemetry data. In *Workshop on Scaling-Up Reinforcement Learning: SURL*, 2019.

[21] Kasra Rezaee, Peyman Yadmellat, and Simon Chamorro. Motion planning for autonomous vehicles in the presence of uncertainty using reinforcement learning. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3506–3511. IEEE, 2021.

[22] Maximilian Schier, Christoph Reinders, and Bodo Rosenhahn. Deep reinforcement learning for autonomous driving using high-level heterogeneous graph representations. In *International Conference on Robotics and Automation (ICRA)*, 2023.

[23] Vincent Sitzmann, Julien Martel, Alexander Bergman, David Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. *Advances in Neural Information Processing Systems*, 33:7462–7473, 2020.

[24] Matthew Tancik, Pratul Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. *Advances in Neural Information Processing Systems*, 33:7537–7547, 2020.

[25] Peter Wolf, Karl Kurzer, Tobias Wingert, Florian Kuhnt, and J Marius Zollner. Adaptive behavior generation for autonomous driving using deep reinforcement learning with compact semantic states. In *2018 IEEE Intelligent Vehicles Symposium (IV)*, pages 993–1000. IEEE, 2018.

[26] Peter R Wurman, Samuel Barrett, Kenta Kawamoto, James MacGlashan, Kaushik Subramanian, Thomas J Walsh, Roberto Capobianco, Alisa Devlic, Franziska Eckert, Florian Fuchs, et al. Outracing champion gran turismo drivers with deep reinforcement learning. *Nature*, 602(7896):223–228, 2022.

[27] Ge Yang, Anurag Ajay, and Pulkit Agrawal. Overcoming the spectral bias of neural value approximation. In *International Conference on Learning Representations*, 2022.

[28] Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Russ R Salakhutdinov, and Alexander J Smola. Deep sets. *Advances in neural information processing systems*, 30, 2017.