Deep Occupancy-Predictive Representations for Autonomous Driving

Eivind Meyer, Lars Frederik Peiss, and Matthias Althoff

Abstract—Manually specifying features that capture the diversity in traffic environments is impractical. Consequently, learning-based agents cannot realize their full potential as neural motion planners for autonomous vehicles. Instead, this work proposes to learn which features are task-relevant. Given its immediate relevance to motion planning, our proposed architecture encodes the probabilistic occupancy map as a proxy for obtaining pre-trained state representations of the environment. By leveraging a map-aware traffic graph formulation, our agent-centric encoder generalizes to arbitrary road networks and traffic situations. We show that our approach significantly improves the downstream performance of a reinforcement learning agent operating in urban traffic environments.

I. INTRODUCTION

Human drivers inherently possess an ability to react to new situations. This is in stark contrast to the narrow operational domains of current reinforcement learning (RL) approaches for self-driving, due to the prevalence of ad hoc feature vectors associated with poor generalization [1], [2]. In particular, two domain-specific characteristics of autonomous driving render the systematic design of relevant and comprehensive state representations difficult: First, the variable number and lack of a canonical ordering of other traffic participants is incompatible with fixed-sized feature vectors. Second, the diversity in road networks in terms of geospatial topology complicates specifying a universal map representation [3].

By adopting graph neural networks (GNNs) as RL policies, recent works have outperformed traditional approaches relying on fixed-sized feature vectors. However, these were confined to homogeneous road network geometries such as highways [4], [5] or roundabouts [6], simplifying the learning problem. Recently, GNN architectures that unify traffic and infrastructure have been proposed for the related task of vehicle trajectory prediction [7]–[12]. However, directly adopting heterogeneous GNNs as policy networks is challenging, as current state-of-the-art RL algorithms cannot be reliably trained in complex environments [13]–[15].

To mitigate the challenging nature of the learning task, we instead formulate a representation objective and design a state representation model detached from the RL training loop. As opposed to letting the agent directly infer control signals from a multi-modal graph representation, we use spatio-temporal occupancy map prediction, as depicted in Fig. 1, as a learning proxy for environment understanding. As illustrated in Fig. 2, we specifically develop a GNN-based encoder-decoder model whose intermediate latent states serve as low-dimensional, pre-trained state representations. Notably, the flexible nature of our encoder allows arbitrary road network topologies and traffic environments to be captured by the learned representations. To alleviate the lossy nature of compressive graph encoding, we propose a novel occupancy prediction framework that constrains the decoding space in accordance with a priori known physical priors for vehicle motion. We implement our approach using CommonRoad-Geometric (crgeo) [16], a PyTorch-based framework offering a standardized graph extraction pipeline for traffic scenarios. Our source code is available at https://github.com/CommonRoad/crgeo-learning.

II. RELATED WORK

We first introduce the learning frameworks used by our approach alongside related applications to motion planning.

A. State representation learning (SRL)

By learning encoded representations of the surroundings, SRL methods enhance the performance of RL agents operating in high-dimensional, complex environments [17]–[19]. An agent-centric approach is generally preferred, so that the learned representations are aligned with the planning context [20]. Further desired requirements for representations are they enable predicting the future world state [21]–[25] (as opposed to merely reconstructing the present) and that they are low-dimensional [26], [27]. To mitigate the trade-off between dimensionality reduction and expressiveness, SRL can be supported by incorporating knowledge about the world as representation priors [18], [28], [29]. Imposing structural...
constraints on the representations, e.g. by enforcing correspondence to physically plausible world states, improves their generalization and downstream effectiveness [30], [31].

B. Graph neural networks (GNNs)

As the graph-compatible counterpart of traditional encoder architectures, GNNs present a framework for applying SRL to traffic environments. Within the context of the widely adopted message passing paradigm [32], GNNs compute neighborhood-aware hidden representations via the permutation-invariant aggregation of edge messages, i.e., neural encodings transmitted from a node to its (outgoing) neighbors. This facilitates the propagation of task-relevant information flow across the graph, which, depending on the learning problem, can be summarized on the graph level via readout operations [33]. As necessitated by the multi-modal traffic graph formulation assumed in this work, GNNs are also extendable to heterogeneous graph inputs [34], [35].

C. Applications to motion planning

Autoencoder-based representation models [36] have been used in a multitude of existing works for learning latent states based on, e.g., rasterized bird’s eye view images [37]–[41] or on-board sensor data [42]. However, they do not leverage the structural biases [43] induced by the road network topology. In line with our approach, [44] uses a GNN-based encoder to learn structurally-aware state representations, but in the context of RL-based robotic manipulators.

Road occupancy as a standalone prediction task is widely covered in existing works [45]–[49]. With the objective of encoding traffic scenes similar to ours (albeit not in the context of motion planning), encoder architectures for learning representations of occupancy maps have been proposed [50]–[52]. Using graphical or otherwise spatially-aware encoders similar to ours, recent works such as [53]–[57] predict occupancy grids [58] as an intermediate learning target for guiding the training of neural motion planners. However, these approaches do not provide global, low-dimensional representations appropriate for decoupled RL agents. In contrast to our work, they also suffer from the lossy nature of grid-wise occupancy discretization [59].

III. METHODOLOGY

Next, we outline the details of our approach.

A. Definitions

1) Heterogeneous traffic graph: As originally proposed in [60], we model road networks as atomic, interconnected road segments (i.e., lanelets). We formalize the dynamic traffic environment at the current time step by the heterogeneous graph tuple $G = (V, E, X_V, X_E)$, where $V = (V_V, V_L)$ indexes the vehicle (V) and lanelet (L) nodes, $E = (E_{v2l}, E_{l2l})$ defines the corresponding vehicle-to-lanelet (V2L) and lanelet-to-lanelet (L2L) edges, and $X_V = (X_v, X_L)$ and $X_E = (X_{v2l}, X_{l2l})$ contain node and edge-level graph features, respectively. Here, the time-dependent V2L edges relate to the physical presence of a vehicle on a given lanelet, whereas the static L2L edges are implied by the road network topology. Our approach incorporates the default graph features provided by crgaeo [16], e.g. velocity (V) and vehicle-lanelet heading difference (V2L).

2) Planning context: The state of the ego vehicle is chosen as the tuple $(p_{ego}, v_{ego})$, consisting of its x-y center position and longitudinal speed. We further denote its length as $\lambda_{ego}$. Next, we let the reference path $\Gamma_{ego} : [0, \zeta_{ego}] \rightarrow \mathbb{R}^2$ of length $\zeta_{ego}$ be parameterized by arclength $s$, and impose the natural constraint that $\Gamma_{ego}(0) = p_{ego}$. As illustrated in Fig. 3, we assume that $\Gamma_{ego}$ follows the centerline of a connected, traffic-compliant sequence of lanelets. The corresponding sequence of lanelet node indices in $V_L$ is denoted by $R_{ego}$. Further, we let $s_{j}^{start}$ and $s_{j}^{end}$ denote the start and endpoint coordinates of the $j$th element in $R_{ego}$, as defined within the arclength-parameterized coordinate frame of the centerlines. Also, we let $d_j$ denote lanelet length, and let $d_j^{prior}$ be the aggregated length of the path segments preceding $j$. Finally, we let the spatial context matrix $C_{ego}$ contain the row vectors $c_j = [s_{j}^{start}, s_{j}^{end}, d_j, d_j^{prior}]$.

B. Occupancy as representation objective

As occupancy explicitly expresses drivable and non-drivable space, it can be considered as the foundational environment characteristic in a motion planning context [61].
We consider occupancy solely in the longitudinal direction, as this is more relevant than the lateral direction and simplifies our modelling assumptions. For a given spatio-temporal coordinate vector \([s, t] \in \mathbb{R}^2\), future path occupancy on \(\Gamma_{ego}\) can be formalized as \(o : [0, \zeta_{ego}] \times \mathbb{R} \rightarrow \{0, 1\}\). As shown in Fig. 4, \(o(s, t)\) is derived from a path projection of the vehicles that overlap with the road surface at time \(t\).

C. Encoder architecture

Our encoding pipeline is formalized as

\[
z_{ego} = \text{ENCODER}(\mathcal{G}, R_{ego}, C_{ego}),
\]

with the low-dimensional representations \(z_{ego} \in \mathbb{R}^Z\) being the final output of the encoder. The GNN-based encoder is designed to facilitate the probabilistic propagation of traffic participants across the given lanelet network, which is used as a neural infrastructure for social message passing. Next, we outline the encoding steps. In general, we use \(\Theta_C\) to denote trainable, nonlinear functions, \(\Sigma\) to denote a permutation-invariant aggregation operation, and assume the activation function \(\rho\) to be applied after each step.

1) Vehicle-to-lanelet: Unlike the vehicle-to-vehicle paradigm proposed in e.g. [5], we capture v2v interaction effects in a map-agnostic fashion by first embedding the vehicles onto the lanelet graph. Letting vehicle and lanelet nodes be indexed by \(i\) and \(j\), respectively, we compute the initial hidden lanelet states \(h_j^{(0)} \in \mathbb{R}^H\) as

\[
h_j^{(0)} = \Theta_k(x_j) + \sum_{(i,j) \in \mathcal{E}_{vl}} \Theta_{vl}([x_i, x_j, x_{i\rightarrow j}]),
\]

where the node and edge features \((x_i, x_j)\) and \(x_{i\rightarrow j}\) denote the corresponding row vectors in \(X_V\) and \(X_E\), respectively.

2) Lanelet-to-lanelet: Next, a total of \(L\) successive l2l message passing layers are used to facilitate the propagation of information flow across the lanelet network. With the superscript \(l\) denoting the layer index, we recursively update the hidden lanelet states according to the update equation

\[
h_j^{(l+1)} = h_j^{(l)} + \sum_{(j',j) \in \mathcal{E}_{l2l}} \Theta_{l2l}([h_{j'}^{(l)}, h_j^{(l)}, x_{j'\rightarrow j}]),
\]

3) Ego-attentional readout operation: Then, an attentional readout layer [62] is used to compute graph-level hidden states \(h_{ego} \in \mathbb{R}^H\). To obtain ego-centered representations, the aggregation of \(h_j^{(L)}\) is weighted by attention scores \(\alpha_{ego}\) based on the ego vehicle’s spatial context \(C_{ego}\), i.e.,

\[
\alpha_{ego} = \text{softmax}(\Theta_C(C_{ego})),
\]

\[
h_{ego} = \sum_{j \in R_{ego}} \alpha_{ego,j} h_j^{(L)}.
\]

4) Downscaling layer: Finally, a downscaling MLP layer is applied to obtain the final latent states \(z_{ego} = \Theta_{\alpha}(h_{ego})\), intended to be used as state observations for the RL agent.

D. Decoder architecture

Our proposed decoder maps \(z_{ego}\) to a continuous parameterization of the probabilistic occupancy map according to

\[
\hat{\delta}(s, t) = \text{DECODER}(z_{ego}, s, t),
\]

\[
o(s, t) \sim \text{Bernoulli}(\hat{\delta}(s, t)).
\]

However, as an abstraction of something tangible (i.e., the presence of vehicles), predicting \(o(s, t)\) in an unconstrained fashion using, e.g., a regular MLP network, might lead to overparameterized predictions that are inconsistent with the data [63]. The large hypothesis space is especially problematic given the low-dimensional nature of \(z_{ego}\) and the resulting information bottleneck. Instead, our novel decoding architecture prevents nonsensical predictions of the occupancy map by inferring it from a decoded set of time-evolving probability distributions referred to as virtual vehicles. This enforces temporal and spatial consistency on the output space and streamlines the learning task. By exploiting the physical priors of our application domain, our method further guarantees that the decoded occupancy maps conform to plausible limits for e.g. vehicle length and velocity.

1) Virtual vehicles: We parameterize the probabilistic occupancy map \(\hat{\delta}(s, t)\) via recurrently decoded virtual vehicles. As outlined in Section III-C, our fixed-sized intermediate representations \(z_{ego}\) are computed by aggregating the elements in \(\mathcal{G}\). Due to the resulting node-level data association loss, it is not viable to reconstruct vehicle instances in a way comparable to related trajectory prediction works [7]–[12]. Without assuming an association between decoded and actual vehicles, our virtual vehicle formulation instead enables a differentiable and permutation-invariant parameterization of the joint probabilistic occupancy map \(\hat{\delta}(s, t)\). This translates the learning problem to the graph-level (i.e., global) domain, yielding a feasible training target.
2) Formal definition: Formally, we let the state of a decoded virtual vehicle $q$ of length $\lambda_q \in \mathbb{R}$ be defined by the tuple $(I_q, p_q)$, where $I_q : \mathbb{R} \rightarrow \{0, 1\}$ is an existence indicator at time $t$, and $p_q : \mathbb{R} \rightarrow \mathbb{R}$ models its time-dependent longitudinal center position on $\Gamma_{ego}$. Further, we let $o_q : [0, \zeta_{ego}] \times \mathbb{R} \rightarrow \{0, 1\}$ return its path occupancy

$$o_q(s, t) = \begin{cases} 1 & \text{if } I_q(t) = 1 \land |s - p_q(t)| < \frac{\lambda_q}{2}, \\ 0 & \text{otherwise}. \end{cases}$$

This expression differs from the vehicle occupancy $o$ considered in Fig. 4, as virtual vehicles serve as non-deterministic, atomic proxies for modelling future occupancy flow.

3) Stochastic formulation: We let $f_p : \mathbb{R} \times \mathbb{R} \rightarrow [0, 1]$ denote the time-varying probability density function (PDF) for $p_q$. As motivated in [64], the Fokker-Planck [65] equation can be used for modelling microscopic traffic flow as a stochastic process. In this framework, $p_q$ is described by a partial differential equation influenced by stochastic forces, addressing the accumulation of uncertainty over time with regards to the vehicle position. As the model training requires a differentiable inference procedure, we use a simplified behavior model with a tractable solution. Specifically, we assume linear drift and diffusion terms given by $\eta_{p,v}$ and $\eta_{p,d}$, as well as a positional offset $\eta_{p,o}$ corresponding to the initial vehicle position. Using ";" to separate the distributions' input space from their given parameterizations, this results in the time-evolving Gaussian solution [65]

$$f_p(s; t, \hat{\eta}_p) = \frac{1}{\sqrt{4\pi\eta_{p,d}t}} \exp \left( \frac{(s - \hat{\eta}_{p,o} - \hat{\eta}_{p,v}t)^2}{4\eta_{p,d}t} \right),$$

$$F_p(s; t, \hat{\eta}_p) = \frac{1}{2} \erf \left( \frac{\hat{\eta}_{p,v}t + \hat{\eta}_{p,o} - s}{2\sqrt{\eta_{p,d}t}} \right),$$

with $F_p(s; t, \hat{\eta}_p) = \int_{-\infty}^{s} f_p(s'; t, \hat{\eta}_p) \, ds'$ denoting the cumulative distribution function. Further, we let $f_T : \mathbb{R} \rightarrow [0, 1]$ denote the predicted existence probability for $q$ at a given $t$. Delayed appearances or disappearances of vehicles on $\Gamma_{ego}$ due to, e.g., lane changes cannot be captured by a constant existence probability $f_T(t) = \hat{\eta}_{T,o}$. By also introducing the temporal offset parameter $\hat{\eta}_{T,v}$, we instead use the time-dependent parameterization given by

$$m_i(t; \hat{\eta}_{T,v}) = \sigma(\tau_T(t - \hat{\eta}_{T,v} + \tau_C)), \quad m_r(t; \hat{\eta}_{T,v}) = \sigma(\tau_T(t - \hat{\eta}_{T,v} + \tau_C)), \quad m(t; \hat{\eta}_{T,v}) = m_i(t; \hat{\eta}_{T,v}) \cdot m_r(t; \hat{\eta}_{T,v}),$$

$$f_T(t; \hat{\eta}_{T,o}, \hat{\eta}_{T,v}) = \hat{\eta}_{T,o} \cdot m(t; \hat{\eta}_{T,v}),$$

with $\sigma$ denoting the sigmoid function, and $\tau_C$ and $\tau_R$ being fixed model hyperparameters. As visualized in Fig. 5, the temporal masking effect produced by the left and right shifting mechanism $m$ provides the modelling flexibility for $f_T$ to predict vehicles leaving or entering $\Gamma_{ego}$ at future time instances. Given the vector parameterization $\eta^{(q)} = \eta^{(q)}_\lambda, \eta^{(q)}_{T,o}, \eta^{(q)}_{T,v}, \eta^{(q)}_{p,v}, \eta^{(q)}_{p,d}, \eta^{(q)}_{p,o}$, where $\eta^{(q)}_\lambda$ is the decoded length, and $\eta^{(q)}_{T,o}$ and $\eta^{(q)}_{T,v}$ contain the respective coefficients for $f_T$ and $f_p$, we model $q$ stochastically as

$$P(I_q(t) = 1) = f_T(t; \hat{\eta}_{T,0}^{(q)}),$$

$$P(p_q(t) = s) = f_p(s; t, \hat{\eta}_{p,0}^{(q)}).$$

Next, we derive the unimodal occupancy map $\hat{o}_q(s, t)$, referred to as the probabilistic occupancy footprint of a single virtual vehicle $q$. We assume that $I_q(t)$ and $p_q(t)$ are independent random variables and use the shorthand notation $s_\pm = s \pm \eta^{(q)}_\lambda / 2$ to denote the occupancy bounds for a given $s$. As is illustrated in Fig. 6, applying (3) and (6) results in

$$\hat{o}_q(s, t) = P(o_q(s, t) = 1) = P(I_q(t) = 1)(s - p_q(t)) < \frac{\eta^{(q)}_\lambda}{2}$$

$$= f_T(t; \hat{\eta}_{T,0}^{(q)}) \int_{s_-}^{s_+} f_p(s'; t, \hat{\eta}_{p,0}^{(q)}) \, ds'.$$

$$= f_T(t; \hat{\eta}_{T,0}^{(q)})(F_p(s_+; t, \hat{\eta}_{p,0}^{(q)}) - F_p(s_-; t, \hat{\eta}_{p,0}^{(q)})).$$

![Fig. 5: Existence probability $f_T(t)$ for two virtual vehicles with identical baseline existence probability $\hat{\eta}_{T,0} = 0.3$ and different temporal offset parameters ($\hat{\eta}_{T,0}^{(1)}, \hat{\eta}_{T,0}^{(2)}$).](image)

E. Joint occupancy probability

We next derive an expression for the predicted joint occupancy $\hat{o}(s, t)$. To facilitate tractable inference, we assume the virtual vehicles to be independent. For a set of virtual vehicles $Q = \{q_1, ..., q_N\}$, the probability of at least one vehicle occupying $\Gamma_{ego}$ at the location $[s, t]$ is given by

$$\hat{o}(s, t) = 1 - \prod_{q \in Q} (1 - \hat{o}_q(s, t)), \quad \text{as exemplified in Fig. 7.}$$

Given a sufficiently large $N$, neither the independence assumption nor the simplified linear motion model assumed in (4) pose significant restrictions on the modeling capacity of our decoder, as (8) allows complex behavior patterns and the corresponding occupancy maps to be modeled via superimposed virtual vehicles.
functions so as to conform to \( \eta \). We transform the raw outputs of \( \Theta \) dataset and introduce the numerical RL experiments that are otherwise intractable, we approximate them numerically with being predominantly unoccupied. As the spatial integrals are introduced to avoid the dependence on segment length and order. Minimizing the binary cross-entropy, we then define respective path segments, the normalization factor \( 1 \) is the considered time horizon and \( \delta_t \) is a discount factor for reflecting the diminishing significance of future occupancy. With \( |\Omega| \) denoting the arclength of the respective path segment, the normalization factor \( 1/|\Omega| \) is introduced to avoid the dependence on segment length and to counteract the class imbalance caused by road surfaces being predominantly unoccupied. As the spatial integrals are otherwise intractable, we approximate them numerically with resolution \( R_t \) as illustrated in Fig. 8a. Similarly, we discretize the temporal integral over \( T_D \) time steps.

IV. Numerical experiments

We next describe the collection of our simulated traffic dataset and introduce the numerical RL experiments that evaluate the effectiveness of our learned state representations.

A. Dataset

Using the OpenStreetMap [67] API, our dataset is comprised of a diverse set of urban locations sampled from within the Munich metropolitan region. The collected road networks are then populated with vehicles using the traffic simulator SUMO [68]. In total, the dataset contains 1000 simulated scenarios, jointly amounting to 28 hours of traffic.

B. Reinforcement learning agent

For our experiments, we extract \( z_{ego} \) as inputs for a PPO-based [69] RL agent together with the current ego velocity \( v_{ego} \). At each time step, the encoder is conditioned on the navigation context generated by a high-level route planner. The resulting reference routes span across multiple lanelets and include heterogeneous map structures such as intersections. We intentionally limit the agent to longitudinal acceleration control along the reference path to focus on the effect of the learned representations. Given the weighting coefficients \( w \in \mathbb{R}^4 \), the reward \( r \) is defined as

\[
 r = w \cdot [r_{path}, r_{collision}, r_{speed}, r_{\delta}],
\]

where \( r_{path} \) is a dense path progression reward, \( r_{collision} \) is a sparse penalty imposed on collisions, \( r_{speed} \) is a linear over-speeding penalty, and \( r_{\delta} \) penalizes expected occupancy conflict from

\[
 r_{\delta} = \int_{t=0}^{T} \delta_t \left( \hat{o}(s, t) + \lambda_{ego} \right) ds dt,
\]

with \( \hat{s}(t) = v_{ego} t \) being a constant-velocity extrapolation of the ego position and \( \delta_r \in [0, 1] \) being a discount factor.

C. Baseline approaches

We compare our PPO agent’s performance against multiple baselines trained with identical reward configurations:

1) V2V: A vehicle-to-vehicle (i.e., not map-aware) GNN policy network as proposed in [5].
2) V2L: A direct adoption of our ENCODER as policy network (i.e., without pre-training \( z_{ego} \)).
3) Naive: The pre-trained representations resulting from using a feed-forward network MLP (256, 128) for decoding \( \hat{o}(s, t) \) from \( [z_{ego}, s, t] \) without architectural constraints. This naively assumes occupancy to be an independent property for each spatio-temporal coordinate vector \([s, t]\).
D. Implementation and training

The end-to-end training of the representation model was conducted on an NVIDIA A100 Tensor Core GPU for 48 hours using the Adam optimizer [70] with the hyperparameters listed in Table I. Subsets of the collected dataset were used for training (90%) and testing (10%). To aid generalization, random planning contexts for the encoder were resampled for each mini-batch during training. The PPO agents were implemented using the Stable-Baselines 3 framework [71] and trained for $10^6$ steps on simulated replays of the scenarios, with start and goal positions for the ego vehicle being randomly sampled for each episode.

TABLE I: Selected hyperparameters for our experiments. We refer to our online available implementation for further details.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Encoding layers ($\Theta_e, \Theta_{e2l}, \Theta_{l2l}, \Theta_C, \Theta_z$)</td>
<td>Linear</td>
</tr>
<tr>
<td>Encoding dimensions $(H, Z)$</td>
<td>256, 32</td>
</tr>
<tr>
<td>Number of l2l layers ($L$)</td>
<td>4</td>
</tr>
<tr>
<td>Aggregation function ($\Sigma$)</td>
<td>max</td>
</tr>
<tr>
<td>Activation function ($\rho$)</td>
<td>tanh</td>
</tr>
<tr>
<td>Reference path length ($\ell_{ego}$)</td>
<td>45 m</td>
</tr>
<tr>
<td>Decoding layers ($\Theta_d, \Theta_q$)</td>
<td>LSTM(256), Linear</td>
</tr>
<tr>
<td>Number of virtual vehicles ($N$)</td>
<td>12</td>
</tr>
<tr>
<td>Temporal masking constants ($\tau_R, \tau_C$)</td>
<td>6.0, 0.7</td>
</tr>
<tr>
<td>Decoding horizon ($T, T_D$)</td>
<td>2.4 s, 60</td>
</tr>
<tr>
<td>Spatial integration method</td>
<td>Trapezoidal($R_f = 40$)</td>
</tr>
<tr>
<td>Temporal occupancy discount factor ($\delta_t$)</td>
<td>0.99</td>
</tr>
</tbody>
</table>

V. RESULTS

As is evident from the results reported in Table II, our proposed model achieves better decoding performance than the unconstrained baseline. This indicates that our approach mitigates the effects of the information bottleneck caused by the encoding pipeline. Specifically, it is likely that the simpler hypothesis space streamlines the training process. Further, a qualitative assessment of the decoded probability maps shown in Fig. 9 suggests that our model is able to accurately predict complex environments to a degree where the intermediate encodings $z_{ego}$ will enable intelligent planning decisions for the agent.

TABLE II: Empirical evaluation results on the test dataset.

<table>
<thead>
<tr>
<th>Model</th>
<th>Decoding loss ($\ell$)</th>
<th>RL Performance</th>
<th>Goal reach (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ours</td>
<td>1.045</td>
<td>Ours</td>
<td>72.9</td>
</tr>
<tr>
<td>Naive</td>
<td>1.210</td>
<td>V2V</td>
<td>39.9</td>
</tr>
<tr>
<td></td>
<td></td>
<td>V2L</td>
<td>49.0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Naive</td>
<td>54.0</td>
</tr>
</tbody>
</table>

The effectiveness of our representations is confirmed by the results of the conducted RL experiments. The improved success rate of the representation-enhanced agent indicates that the pre-trained representations simplify its motion planning task. Effectively, using them as state observations frees the agent from the responsibility of modeling its own surroundings, allowing it to concentrate its learning capacity on the lower-level control aspects of motion planning. As traffic modeling is a complex endeavor that is more easily

![Fig. 9: Decoded occupancy maps $\hat{o}(s,t)$ visualized together with color encodings of the corresponding latent representations $z_{ego}$, which are extracted as state observations for the RL agent.](image)

(b) The rapid intensity decay of the leftmost probability mode suggests an accurate representation of B’s likely departure from the ego route.

tackled in a supervised setting, it is thus unsurprising that the simplification of the RL task improves the final performance.

VI. CONCLUSION

We present a novel encoder-decoder architecture for learning latent state representations that enhance the performance of RL-based motion planning in heterogeneous driving environments. Our approach recurrently decodes and aggregates virtual vehicles to predict the spatio-temporal occupancy map. This ensures that the representation space is constrained to physically plausible predictions, which further enhances the model’s interpretability and reduces its black-box nature. By using a heterogeneous GNN encoder to compress the traffic surroundings, our approach offers a significant benefit compared to previous works; as opposed to feature vectors narrowly tailored to specific traffic settings, our approach naturally extends to arbitrary road networks.

ACKNOWLEDGEMENTS

This research was funded by the German Research Foundation grant AL 1185/7-1 and the German Federal Ministry for Digital and Transport through the project KoSi.
REFERENCES


[7] Y. Li, B. Liang, B. Yang, R. Hu, Y. Chen, R. Liao, S. Feng, and R. Urta-


[12] X. Mo, Z. Huang, Y. Xing, and C. Lv, “Multi-agent trajectory pre-


