NEAT: Neural Attention Fields for End-to-End Autonomous Driving

Kashyap Chitta*[1,2] Aditya Prakash*[1] Andreas Geiger[1,2]
[1]Max Planck Institute for Intelligent Systems, Tübingen [2]University of Tübingen
{firstname.lastname}@tue.mpg.de

Abstract

Efficient reasoning about the semantic, spatial, and temporal structure of a scene is a crucial prerequisite for autonomous driving. We present NEural ATtention fields (NEAT), a novel representation that enables such reasoning for end-to-end imitation learning models. NEAT is a continuous function which maps locations in Bird's Eye View (BEV) scene coordinates to waypoints and semantics, using intermediate attention maps to iteratively compress high-dimensional 2D image features into a compact representation. This allows our model to selectively attend to relevant regions in the input while ignoring information irrelevant to the driving task, effectively associating the images with the BEV representation. In a new evaluation setting involving adverse environmental conditions and challenging scenarios, NEAT outperforms several strong baselines and achieves driving scores on par with the privileged CARLA expert used to generate its training data. Furthermore, visualizing the attention maps for models with NEAT intermediate representations provides improved interpretability.

1. Introduction

Navigating large dynamic scenes for autonomous driving requires a meaningful representation of both the spatial and temporal aspects of the scene. Imitation Learning (IL) by behavior cloning has emerged as a promising approach for this task [5, 10, 16, 53, 78]. Given a dataset of expert trajectories, a behavior cloning agent is trained through supervised learning, where the goal is to predict the actions of the expert given some sensory input regarding the scene [48]. To account for the complex spatial and temporal scene structure encountered in autonomous driving, the training objectives used in IL-based driving agents have evolved by incorporating auxiliary tasks. Pioneering methods, such as CILRS [16], use a simple self-supervised auxiliary training objective of predicting the ego-vehicle velocity. Since then, more complex training signals aiming to reconstruct the scene have become common, e.g. image auto-encoding [47], 2D semantic segmentation [29], Bird's Eye View (BEV) semantic segmentation [37], 2D semantic prediction [27], and BEV semantic prediction [58]. Performing an auxiliary task such as BEV semantic prediction, which requires the model to output the BEV semantic segmentation of the scene at both the observed and future time-steps, incorporates spatiotemporal structure into the intermediate representations learned by the agent. This has been shown to lead to more interpretable and robust models [58]. However, so far, this has only been possible with expensive LiDAR and HD map-based network inputs which can be easily projected into the BEV coordinate frame.

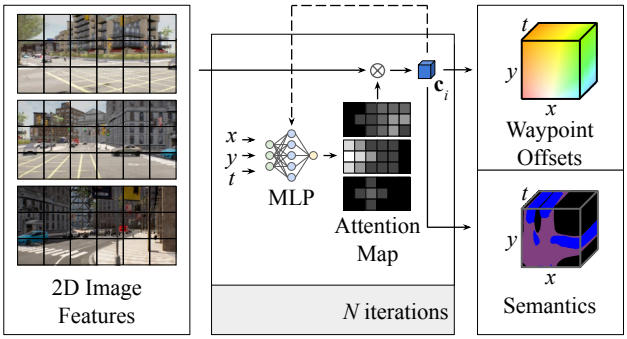


Figure 1: **Neural Attention Fields.** We use an MLP to iteratively compress the high-dimensional input into a compact low-dimensional representation $\mathbf{c}_i$ based on the BEV query location (x, y, t). Our model outputs waypoint offsets and auxiliary semantics from $\mathbf{c}_i$ continuously and with a low memory footprint. Training for both tasks jointly leads to improved driving performance on CARLA.

The key challenge impeding BEV semantic prediction from camera inputs is one of association: given a BEV spatiotemporal query location (x, y, t) in the scene (e.g. 2 meters in front of the vehicle, 5 meters to the right, and 2 seconds into the future), it is difficult to identify which image pixels to associate to this location, as this requires reasoning about 3D geometry, scene motion, ego-motion, and intention, as well as interactions between scene elements. In this paper, we propose **NEural ATtention fields (NEAT)**, a

*indicates equal contribution

flexible and efficient feature representation designed to address this challenge. Inspired by implicit shape representations [39, 50], NEAT represents large dynamic scenes with a fixed memory footprint using a multi-layer perceptron (MLP) query function. The core idea is to learn a function from any query location (x, y, t) to an attention map for features obtained by encoding the input images. NEAT compresses the high-dimensional image features into a compact low-dimensional representation relevant to the query location (x, y, t), and provides interpretable attention maps as part of this process, without attention supervision [79]. As shown in Fig. 1, the output of this learned MLP can be used for dense prediction in space and time. Our end-to-end approach predicts waypoint offsets to solve the main trajectory planning task (described in detail in Section 3), and uses BEV semantic prediction as an auxiliary task.

Using NEAT intermediate representations, we train several autonomous driving models for the CARLA driving simulator [20]. We consider a more challenging evaluation setting than existing work based on the new CARLA Leaderboard [1] with CARLA version 0.9.10, involving the presence of multiple evaluation towns, new environmental conditions, and challenging pre-crash traffic scenarios. We outperform several strong baselines and match the privileged expert's performance on our internal evaluation routes. On the secret routes of the CARLA Leaderboard, NEAT obtains competitive driving scores while incurring significantly fewer infractions than existing methods.

Contributions: (1) We propose an architecture combining our novel NEAT feature representation with an implicit decoder [39] for joint trajectory planning and BEV semantic prediction in autonomous vehicles. (2) We design a challenging new evaluation setting in CARLA consisting of 6 towns and 42 environmental conditions and conduct a detailed empirical analysis to demonstrate the driving performance of NEAT. (3) We visualize attention maps and semantic scene interpolations from our interpretable model, yielding insights into the learned driving behavior. Our code is available at https://github.com/autonomousvision/neat.

2. Related Work

Implicit Scene Representations: The geometric deep learning community has pioneered the idea of using neural implicit representations of scene geometry. These methods represent surfaces as the boundary of a neural classifier [12, 13, 35, 39, 59] or zero-level set of a signed distance field regression function [36, 40, 50, 62, 63, 77]. They have been applied for representing object texture [44, 45, 64], dynamics [43] and lighting properties [41, 46, 61]. Recently, there has been progress in applying these representations to compose objects from primitives [11, 18, 19, 24], and to represent larger scenes, both static [8, 31, 51] and dynamic [21, 33, 34, 74]. These methods obtain high-resolution scene representations while remaining compact, due to the constant memory footprint of the neural function approximator. While NEAT is motivated by the same property, we use the compactness of neural approximators to learn better intermediate features for the downstream driving task.

End-to-End Autonomous Driving: Learning-based autonomous driving is an active research area [30, 65]. IL for driving has advanced significantly [5, 15, 42, 53, 72, 78] and is currently employed in several state-of-the-art approaches, some of which predict waypoints [7, 10, 23], whereas others directly predict vehicular control [4, 6, 16, 29, 47, 54, 75, 80]. While other learning-based driving methods such as affordances [60, 76] and Reinforcement Learning [9, 66, 70] could also benefit from a NEAT-based encoder, in this work, we apply NEAT to improve IL-based autonomous driving.

BEV Semantics for Driving: A top-down view of a street scene is powerful for learning the driving task since it contains information regarding the 3D scene layout, objects do not occlude each other, and it represents an orthographic projection of the physical 3D space which is better correlated with vehicle kinematics than the projective 2D image domain. LBC [10] exploits this representation in a teacher-student approach. A teacher that learns to drive given BEV semantic inputs is used to supervise a student aiming to perform the same task from images only. By doing so, LBC achieves state-of-the-art performance on the previous CARLA version 0.9.6, showcasing the benefits of the BEV representation. NEAT differs from LBC by directly learning in BEV space, unlike the LBC student model which learns a classical image-to-trajectory mapping.

Other works deal with BEV scenes, e.g., obtaining BEV projections [2, 81] or BEV semantic predictions [26, 28, 38, 49, 56] from images, but do not use these predictions for driving. More recently, LSS [52] and OGMs [37] demonstrated joint BEV semantic reconstruction and driving from camera inputs. Both methods involve explicit projection based on camera intrinsics, unlike our learned attention-based feature association. They only predict semantics for static scenes, while our model includes a time component, performing prediction up to a fixed horizon. Moreover, unlike us, they only evaluate using offline metrics which are known to not necessarily correlate well with actual downstream driving performance [14]. Another related work is P3 [58] which jointly performs BEV semantic prediction and driving. In comparison to P3 which uses expensive LiDAR and HD map inputs, we focus on image modalities.

3. Method

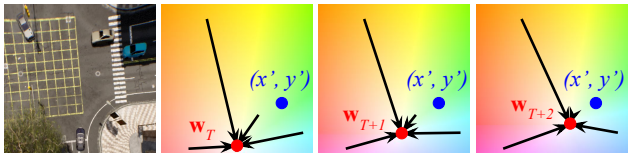
A common approach to learning the driving task from expert demonstrations is end-to-end trajectory planning, which uses *waypoints* $\mathbf{w}_t$ as outputs. A waypoint is defined

as the position of the vehicle in the expert demonstration at time-step t, in a BEV projection of the vehicle's local coordinate system. The coordinate axes are fixed such that the vehicle is located at $(x, y) = (0, 0)$ at the current time-step $t = T$, and the front of the vehicle is aligned along the positive y-axis. Waypoints from a sequence of future time-steps $t = T + 1, ..., T + Z$ form a trajectory that can be used to control the vehicle, where Z is a fixed prediction horizon.

As our agent drives through the scene, we collect sensor data into a fixed-length buffer of T time-steps, $\mathcal{X} = \{\mathbf{x}_{s,t}\}_{s=1:S,t=1:T}$ where each $\mathbf{x}_{s,t}$ comes from one of S sensors. The final frame in the buffer is always the current time-step ($t = T$). In practice, the S sensors are RGB cameras, the standard input modality in existing work on CARLA [10]. By default, we use $S = 3$ cameras, one oriented forward and the others 60 degrees to the left and right. After cropping these camera images to remove radial distortion, these $S = 3$ images together provide a full 180° view of the scene in front of the vehicle. While NEAT can be applied with different buffer sizes, we focus in our experiments on the setting where the input is a single frame ($T = 1$), as several studies indicate that using historical observations can be detrimental to the driving task [69, 73].

In addition to waypoints, we use BEV semantic prediction as an auxiliary task to improve driving performance. Unlike waypoints which are small in number (e.g. $Z = 4$) and can be predicted discretely, BEV semantic prediction is a dense prediction task, aiming to predict semantic labels at any spatiotemporal query location (x, y, t) bounded to some spatial range and the time interval $1 \leq t \leq T + Z$. Predicting both observed ($1 \leq t < T$) and future ($T < t \leq T + Z$) semantics provides a holistic understanding of the scene dynamics. Dynamics prediction from a single input frame is possible since the orientation and position of vehicles encodes information regarding their motion [68].

The coordinate system used for BEV semantic prediction is the same as the one used for waypoints. Thus, if we frame waypoint prediction as a dense prediction task, it can be solved simultaneously with BEV semantic prediction using the proposed NEAT as a shared representation. Therefore, we propose a **dense offset prediction** task to locate waypoints as visualized in Fig. 2 using a standard optical flow color wheel [3]. The goal is to learn the field of 2-dimensional offset vectors $\mathbf{o}$ from query locations (x, y, t) to the waypoint $\mathbf{w}_t$ (e.g. $\mathbf{o} = (0, 0)$ when $(x, y) = \mathbf{w}_T$ and $t = T$). In certain situations, future waypoints along different trajectories are plausible (e.g. taking a left or right turn at an intersection), thus it is important to adapt $\mathbf{o}$ based on the driver intention. We do this by using provided *target locations* (x', y') as inputs. Target locations are GPS coordinates provided by a navigational system along the route to be followed. They are transformed to the same coordinate system as the waypoints before being used as inputs. These



(a) Scene BEV (b) t = T (c) t = T + 1 (d) t = T + 2

Figure 2: **Dense offset prediction.** We visualize the target location (x', y') (blue dot), waypoint $\mathbf{w}_t$ (red dot) and waypoint offsets $\mathbf{o}$ (arrows) for a scene at three time instants. The offsets $\mathbf{o}$ represent the 2D vector from any query location (x, y) to the waypoint $\mathbf{w}_t$ at time t and thus implicitly represent the waypoint. The arrows illustrate $\mathbf{o}$ for four different query locations (x, y). We also show a color coding based visualization of the dense flow field learned by our model, representing $\mathbf{o}$ from any (x, y) location in the scene.

target locations are sparse and can be hundreds of meters apart. In Fig. 2, the target location to the right of the intersection helps the model decide to turn right rather than proceeding straight. We choose target locations as the method for specifying driver intention as they are the default intention signal in the CARLA simulator since version 0.9.9. In summary, the goal of dense offset prediction is to output $\mathbf{o}$ for any 5-dimensional query point $\mathbf{p} = (x, y, t, x', y')$.

3.1. Architecture

As illustrated in Fig. 3, our architecture consists of three neural networks that are jointly trained for the BEV semantic prediction and dense offset prediction tasks: an encoder e_θ, neural attention field a_ϕ, and decoder d_ψ. In the following, we go over each of the three components in detail.

Encoder: Our encoder e_θ takes as inputs the sensor data buffer $\mathcal{X}$ and a scalar v, which is the vehicle velocity at the current time-step T. Formally, it is denoted as

$$e_\theta : \mathbb{R}^{S\times T\times W\times H\times 3} \times \mathbb{R} \rightarrow \mathbb{R}^{(S*T*P)\times C} \quad (1)$$

where θ denotes the encoder parameters. Each image $\mathbf{x}_{s,t} \in \mathbb{R}^{W\times H\times 3}$ is processed by a ResNet [25] to provide a grid of features from the penultimate layer of size $\mathbb{R}^{P\times C}$, where P is the number of spatial features per image and C is the feature dimensionality. For the 256×256 pixel input resolution we consider, we obtain $P = 64$ patches from the default ResNet architecture. These features are further processed by a transformer [67]. The goal of the transformer is to integrate features globally, adding contextual cues to each patch with its self-attention mechanism. This enables interactions between features across different images and over a large spatial range. Note that the transformer can be removed from our encoder without changing the output dimensionality, but we include it since it provides an improvement as per our ablation study. Before being input to the transformer, each patch feature is combined (through addition)

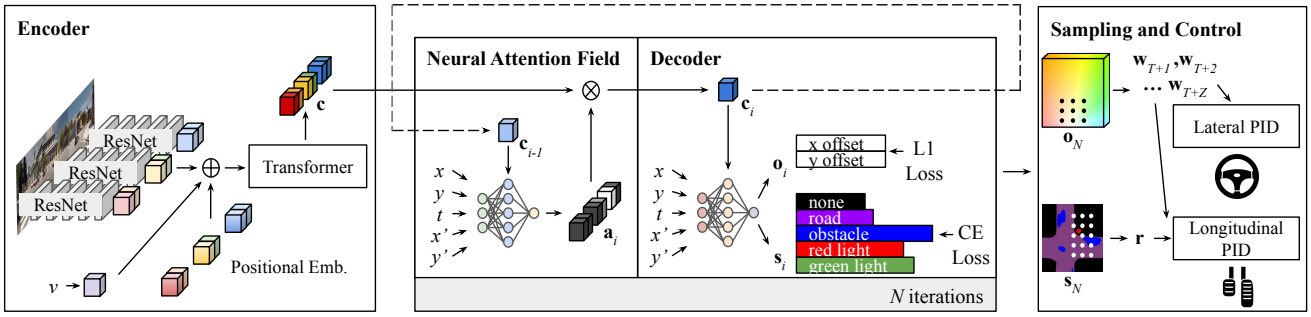


Figure 3: **Model Overview.** In the encoder, image patch features, velocity features, and a learned positional embedding are summed and fed into a transformer. We illustrate this with 2 features per image, though our model uses 64 in practice. NEAT recurrently updates an attention map $\mathbf{a}_i$ for the encoded features $\mathbf{c}$ for N iterations. The inputs to NEAT are a query point $\mathbf{p} = (x, y, t, x', y')$ and feature $\mathbf{c}_i$. For the initial iteration, $\mathbf{c}_0$ is set to the mean of $\mathbf{c}$. The dotted arrow shows the recursion of features between subsequent iterations. In each iteration, the decoder predicts the waypoint offset $\mathbf{o}_i$ and the semantic class $\mathbf{s}_i$ for any given query $\mathbf{p}$, which are supervised using loss functions. At test time, we sample predictions from grids on $\mathbf{o}_N$ and $\mathbf{s}_N$ to obtain a waypoint for each time-step $\mathbf{w}_t$ and red light indicator $\mathbf{r}$, which are used by PID controllers for driving.

with (1) a velocity feature obtained by linearly projecting v to $\mathbb{R}^C$, and broadcasting to all patches of all sensors at all time-steps, as well as (2) a learned positional embedding, which is a trainable parameter of size $(S * T * P) \times C$. The transformer outputs patch features $\mathbf{c} \in \mathbb{R}^{(S*T*P)\times C}$.

Neural Attention Field: While the transformer aggregates features globally, it is not informed by the query and target location. Therefore, we introduce NEAT (Fig. 1), which identifies the patch features from the encoder relevant for making predictions regarding any query point in the scene $\mathbf{p} = (x, y, t, x', y')$. It introduces a bottleneck in the network and improves interpretability (Fig. 6). Its operation can be formally described as

$$a_\phi : \mathbb{R}^5 \times \mathbb{R}^C \rightarrow \mathbb{R}^{S*T*P} \tag{2}$$

Note that the target location (x', y') input to NEAT is omitted in Fig. 1 for clarity. While NEAT could in principle directly take as inputs $\mathbf{p}$ and $\mathbf{c}$, this would be inefficient due to the high dimensionality of $\mathbf{c} \in \mathbb{R}^{(S*T*P)\times C}$. We instead use a simple iterative attention process with N iterations. At iteration i, the output $\mathbf{a}_i \in \mathbb{R}^{S*T*P}$ of NEAT is used to obtain a feature $\mathbf{c}_i \in \mathbb{R}^C$ specific to the query point $\mathbf{p}$ through a softmax-scaled dot product between $\mathbf{a}_i$ and $\mathbf{c}$:

$$\mathbf{c}_i = \text{softmax}(\mathbf{a}_i)^\top \cdot \mathbf{c} \tag{3}$$

The feature $\mathbf{c}_i$ is used as the input of a_ϕ along with $\mathbf{p}$ at the next attention iteration, implementing a recurrent attention loop (see Fig. 3). Note that the dimensionality of $\mathbf{c}_i$ is significantly smaller than that of the transformer output $\mathbf{c}$, as $\mathbf{c}_i$ aggregates information (via Eq. (3)) across sensors S, time-steps T and patches P. For the initial iteration, $\mathbf{c}_0$ is set to the mean of $\mathbf{c}$ (equivalent to assuming a uniform initial attention). We implement a_ϕ as a fully-connected MLP with 5 ResNet blocks of 128 hidden units each, conditioned on $\mathbf{c}_i$ using conditional batch normalization [17,22] (details in supplementary). We share the weights of a_ϕ across all iterations which works well in practice.

Decoder: The final network in our model is the decoder:

$$d_\psi : \mathbb{R}^5 \times \mathbb{R}^C \rightarrow \mathbb{R}^K \times \mathbb{R}^2 \tag{4}$$

It is an MLP with a similar structure to a_ϕ, but differing in terms of its output layers. Given $\mathbf{p}$ and $\mathbf{c}_i$, the decoder predicts the semantic class $\mathbf{s}_i \in \mathbb{R}^K$ (where K is the number of classes) and waypoint offset $\mathbf{o}_i \in \mathbb{R}^2$ at each of the N attention iterations. While the outputs decoded at intermediate iterations ($i < N$) are not used at test time, we supervise these predictions during training to ease optimization.

3.2. Training

Sampling: An important consideration is the choice of query samples $\mathbf{p}$ during training, and how to acquire ground truth labels for these points. Among the 5 dimensions of $\mathbf{p}$, x' and y' are fixed for any $\mathcal{X}$, but x, y, and t can all be varied to access different positions in the scene. Note that in the CARLA simulator, the ground truth waypoint is only available at discrete time-steps, and the ground truth semantic class only at discrete (x, y, t) locations. However, this is not an issue for NEAT as we can supervise our model using arbitrarily sparse observations in the space-time volume. We consider $K = 5$ semantic classes by default: none, road, obstacle (person or vehicle), red light, and green light. The location and state of the traffic light affecting the ego-vehicle are provided by CARLA. We use this to set the semantic label for points within a fixed radius of the traffic light pole to the red light or green light class, similar to [10]. In our work, we focus on the simulated setting where this information is readily available, though BEV semantic annotations

of objects (obstacles, red lights, and green lights) can also be obtained for real scenes using projection. The only remaining labels required by NEAT (for the road class) can be obtained by fitting the ground plane to LiDAR sweeps in a real dataset or more directly from localized HD maps.

We acquire these BEV semantic annotations from CARLA up to a fixed prediction horizon Z after the current time-step T and register them to the coordinate frame of the ego-vehicle at $t = T$. Z is a hyper-parameter that can be used to modulate the difficulty of the prediction task. From the aligned BEV semantic images, we only consider points approximately in the field-of-view of our camera sensors. We use a range of 50 meters in front of the vehicle and 25 meters to either side (detailed sensor configurations are provided in the supplementary).

Since semantic class labels are typically heavily imbalanced, simply using all observations for training (or sampling a random subset) would lead to several semantic classes being under-represented in the training distribution. We use a class-balancing sampling heuristic during training to counter this. To sample M points for K semantic classes, we first group all the points from all the $T + Z$ time-steps in the sequence into bins based on their semantic label. We then attempt to randomly draw $\frac{M}{K}$ points from each bin, starting with the class having the least number of available points. If we are unable to sample enough points from any class, this difference is instead sampled from the next bin, always prioritizing classes with fewer total points.

We obtain the offset ground truth for each of these M sampled points by collecting the ground truth waypoints for the $T + Z$ time-steps around each frame. The offset label for each of the M sampled points is calculated as its difference from the ground truth waypoint at the corresponding time-step $\mathbf{w}_t$. Being a regression task, we find that offset prediction does not benefit as much from a specialized sampling strategy. Therefore, we use the same M points for supervising the offsets even though they are sampled based on semantic class imbalance, improving training efficiency.

Loss: For each of the M sampled points, the decoder makes predictions $\mathbf{s}_i$ and $\mathbf{o}_i$ at each of the N attention iterations. The encoder, NEAT and decoder are trained jointly with a loss function applied to these MN predictions:

$$\mathcal{L} = \frac{1}{MN} \sum_{i}^{N} \gamma_i \sum_{j}^{M} ||\mathbf{o}_j^* - \mathbf{o}_{i,j}||_1 + \lambda \mathcal{L}_{CE}(\mathbf{s}_j^*, \mathbf{s}_{i,j}) \quad (5)$$

where $||.||_1$ is the L_1 distance between the true offset $\mathbf{o}^*$ and predicted offset $\mathbf{o}_i$, $\mathcal{L}_{CE}$ is the cross-entropy between the true semantic class $\mathbf{s}^*$ and predicted class $\mathbf{s}_i$, λ is a weight between the semantic and offset terms, and γ_i is used to down-weight predictions made at earlier iterations ($i < N$). These intermediate losses improve performance, as we show in our experiments.

3.3. Controller

To drive the vehicle at test time, we generate a red light indicator and waypoints from our trained model; and convert them into steer, throttle, and brake values. For the red light indicator, we uniformly sample a sparse grid of $U \times V$ points in (x, y) at the current time-step $t = T$, in the area 50 meters to the front and 25 meters to the right side of the vehicle. We append the target location (x', y') to these grid samples to obtain 5-dimensional queries that can be used as NEAT inputs. From the semantic prediction obtained for these points at the final attention iteration $\mathbf{s}_N$, we set the red light indicator $\mathbf{r}$ as 0 if none of the points belongs to the red light class, and 1 otherwise. In our ablation study, we find this indicator to be important for performance.

To generate waypoints, we sample a uniformly spaced grid of $G \times G$ points in a square region of side A meters centered at the ego-vehicle at each of the future time-steps $t = T + 1, \ldots, T + Z$. Note that predicting waypoints with a single query point ($G = 1$) is possible, but we use a grid for robustness. After encoding the sensor data and performing N attention iterations, we obtain $\mathbf{o}_N$ for each of the G^2 query points at each of the Z future time-steps. We offset the (x, y) location coordinates of each query point towards the waypoint by adding $\mathbf{o}_N$, effectively obtaining the waypoint prediction for that sample, i.e. $(\mathbf{p}[0], \mathbf{p}[1])$ += $\mathbf{o}_N$. After this offset operation, we average all G^2 waypoint predictions at each future time instant, yielding the final waypoint predictions $\mathbf{w}_t$. To obtain the throttle and brake values, we compute the vectors between waypoints of consecutive time-steps and input the magnitude of these vectors to a longitudinal PID controller along with the red light indicator. The relative orientation of the waypoints is input to a lateral PID controller for turns. Please refer to the supplementary material for further details on both controllers.

4. Experiments

In this section, we describe our experimental setting, showcase the **driving performance** of NEAT in comparison to several baselines, present an **ablation study** to highlight the importance of different components of our architecture, and show the interpretability of our approach through **visualizations** obtained from our trained model.

Task: We consider the task of navigation along pre-defined routes in CARLA version 0.9.10 [20]. A route is defined by a sequence of sparse GPS locations (target locations). The agent needs to complete the route while coping with background dynamic agents (pedestrians, cyclists, vehicles) and following traffic rules. We tackle a new challenge in CARLA 0.9.10: each of our routes may contain several predefined dangerous **scenarios** (e.g. unprotected turns, other vehicles running red lights, pedestrians emerging from occluded regions to cross the road).

Routes: For training data generation, we store data using an expert agent along routes from the 8 publicly available towns in CARLA, randomly spawning scenarios at several locations along each route. We evaluate NEAT on the official CARLA Leaderboard [1], which consists of 100 secret routes with unknown environmental conditions. We additionally conduct an internal evaluation consisting of 42 routes from 6 different CARLA towns (Town01-Town06). Each route has a unique environmental condition combining one of 7 weather conditions (Clear, Cloudy, Wet, MidRain, WetCloudy, HardRain, SoftRain) with one of 6 daylight conditions (Night, Twilight, Dawn, Morning, Noon, Sunset). Additional details regarding our training and evaluation routes are provided in the supplementary. Note that in this new evaluation setting, the multi-lane road layouts, distant traffic lights, high density of background agents, diverse daylight conditions, and new metrics which strongly penalize infractions (described below) make navigation more challenging, leading to reduced scores compared to previous CARLA benchmarks [16, 20].

Metrics: We report the official metrics of the CARLA Leaderboard, **Route Completion (RC)**, **Infraction Score (IS)**[1] and **Driving Score (DS)**. For a given route, RC is the percentage of the route distance completed by the agent before it deviates from the route or gets blocked. IS is a cumulative multiplicative penalty for every collision, lane infraction, red light violation, and stop sign violation. Please refer to the supplementary material for additional details regarding the penalty applied for each kind of infraction. Finally, DS is computed as the RC weighted by the IS for that route. After calculating all metrics per route, we report the mean performance over all 42 routes. We perform our internal evaluation three times for each model and report the mean and standard deviation for all metrics.

Baselines: We compare our approach against several recent methods. **CILRS** [16] learns to directly predict vehicle controls (as opposed to waypoints) from visual features while being conditioned on a discrete navigational command (follow lane, change lane left/right, turn left/right). It is a widely used baseline for the old CARLA version 0.8.4, which we adapted to the latest CARLA version. **LBC** [10] is a knowledge distillation approach where a teacher model with access to ground truth BEV semantic segmentation maps is first trained using expert supervision to predict waypoints, followed by an image-based student model which is trained using supervision from the teacher. It is the state-of-the-art approach on CARLA version 0.9.6. We train LBC on our dataset using the latest codebase provided by the authors for CARLA version 0.9.10. **AIM** [55] is an improved version of CILRS, where a GRU decoder regresses waypoints. To assess the effects of different forms of auxiliary supervision, we create 3 multi-task variants of AIM (**AIM-MT**). Each variant adds a different auxiliary task during training: (1) 2D semantic segmentation using a deconvolutional decoder, (2) BEV semantic segmentation using a spatial broadcast decoder [71], and (3) both 2D depth estimation and 2D semantic segmentation as in [32]. We also replace the CILRS backbone of Visual Abstractions [4] with AIM, to obtain **AIM-VA**. This approach generates 2D segmentation maps from its inputs which are then fed into the AIM model for driving. Finally, we report results for the privileged **Expert** used for generating our training data.

Implementation: By default, NEAT's transformer uses $L = 2$ layers with 4 parallel attention heads. Unless otherwise specified, we use $T = 1$, $Z = 4$, $P = 64$, $C = 512$, $N = 2$, $M = 64$, $U = 16$, $V = 32$, $G = 3$ and $A = 2.5$. We use a weight of $\lambda = 0.1$ on the L_1 loss, set $\gamma_i = 0.1$ for the intermediate iterations ($i < N$), and set $\gamma_N = 1$. For a fair comparison, we choose the best performing encoders for each model among ResNet-18, ResNet-34, and ResNet-50 (NEAT uses ResNet-34). Moreover, we chose the best out of two different camera configurations ($S = 1$ and $S = 3$) for each model, using a late fusion strategy for combining sensors in the baselines when we set $S = 3$. Additional details are provided in the supplementary.

4.1. Driving Performance

Our results are presented in Table 1. Table 1a focuses on our internal evaluation routes, and Table 1b on our submissions to the CARLA Leaderboard. Note that we could not submit all the baselines from Table 1a or obtain statistics for multiple evaluations of each model on the Leaderboard due to the limited monthly evaluation budget (200 hours).

Importance of Conditioning: We observe that in both evaluation settings, CILRS and LBC perform poorly. However, a major improvement can be obtained with a different conditioning signal. CILRS uses discrete navigational commands for conditioning, and LBC uses target locations represented in image space. By using target locations in BEV space and predicting waypoints, AIM and NEAT can more easily adapt their predictions to a change in driver intention, thereby achieving better scores. We show this adaptation of predictions for NEAT in Fig. 4, by predicting semantics and waypoint offsets for different target locations x' and time steps t. The waypoint offset formulation of NEAT introduces a bias that leads to smooth trajectories between consecutive waypoints (red lines in $\mathbf{o}_N$) towards the provided target location in blue.

AIM-MT and Expert: We observe that AIM-MT is a strong baseline that becomes progressively better with denser forms of auxiliary supervision. The final variant which incorporates dense supervision of both 2D depth and

[1]The Leaderboard refers to this as infraction penalty. We use the terminology 'score' since it is a multiplier for which higher values are better.

Method	Aux. Sup.	RC ↑	IS ↑	DS ↑
CILRS [16]	Velocity	35.46±0.41	0.66±0.02	22.97±0.90
LBC [10]	BEV Sem	61.35±2.26	0.57±0.02	29.07±0.67
AIM [55]	None	70.04±2.31	0.73±0.03	51.25±0.17
AIM-MT	2D Sem	80.21±3.55	0.74±0.02	57.95±2.76
	BEV Sem	77.93±3.06	0.78±0.01	60.62±2.33
	Depth+2D Sem	**80.81±2.47**	0.80±0.01	64.86±2.52
AIM-VA	2D Sem	75.40±1.53	0.79±0.02	60.94±0.79
NEAT	BEV Sem	79.17±3.25	**0.82±0.01**	**65.10±1.75**
Expert	N/A	86.05±2.58	0.76±0.01	62.69±2.36

(a) **CARLA 42 Routes.** We show the mean and standard deviation over 3 evaluations for each model. NEAT obtains the best driving score, on par with (and sometimes even outperforming) the expert used for data collection.

#	Method	Aux. Sup.	RC ↑	IS ↑	DS ↑
1	WOR [9]	2D Sem	57.65	0.56	31.37
2	MaRLn [66]	2D Sem+Aff	46.97	0.52	24.98
3	NEAT (Ours)	BEV Sem	41.71	0.65	21.83
4	AIM-MT	Depth+2D Sem	67.02	0.39	19.38
5	TransFuser [55]	None	51.82	0.42	16.93
6	LBC [10]	BEV Sem	17.54	0.73	8.94
7	CILRS [16]	Velocity	14.40	0.55	5.37

(b) **CARLA Leaderboard.** Among all publicly visible entries (accessed in July 2021), NEAT obtains the third-best DS. Of the top 3 methods, NEAT has the highest IS, indicating safer driving on unseen routes.

Table 1: **Quantitative Evaluation on CARLA.** We report the RC, IS and DS over our 42 internal evaluation routes (Table 1a) and 100 secret routes on the evaluation server [1] (Table 1b). We abbreviate semantics with "Sem" and affordances with "Aff".

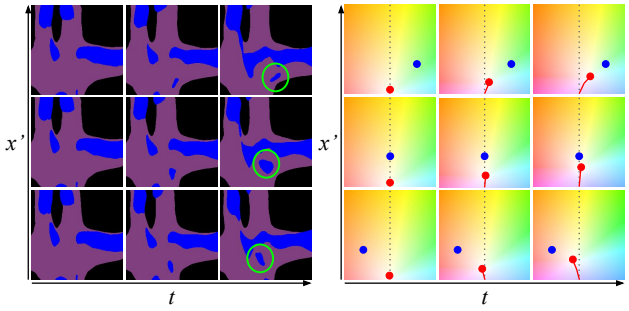


Figure 4: **NEAT Visualization.** We show $\mathbf{s}_N$ (left) and $\mathbf{o}_N$ (right) as we interpolate x' and t for the scene in Fig. 1. The green circles highlight the different predicted ego-vehicle positions. On the right, we show the predicted trajectory and waypoint $\mathbf{w}_t$ in red. Note how the model adapts its prediction to the target location (x', y') (shown in blue).

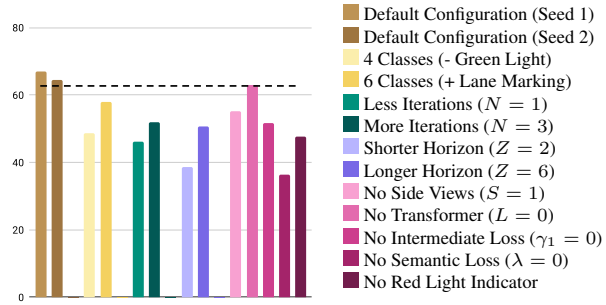


Figure 5: **Ablation Study.** We show the mean DS over our 42 evaluation routes for different NEAT model configurations. Expert performance is shown as a dotted line.

2D semantics achieves similar performance to NEAT on our 42 internal evaluation routes but does not generalize as well to the unseen routes of the Leaderboard (Table 1b). Interestingly, in some cases, AIM-MT and NEAT match or even exceed the performance of the privileged expert in Table 1a. Though our expert is an improved version of the one used in [10], it still incurs some infractions due to its reliance on relatively simple heuristics and driving rules.

Leaderboard Results: While NEAT is not the best performing method in terms of DS, it has the safest driving behavior among the top three methods on the Leaderboard, as evidenced by its higher IS. WOR [9] is concurrent work that supervises the driving task with a Q function obtained using dynamic programming, and MaRLn is an extension of the Reinforcement Learning (RL) method presented in [66]. WOR and MaRLn require 1M and 20M training frames respectively. In comparison, our training dataset only has 130k frames, and can potentially be improved through orthogonal techniques such as DAgger [54, 57].

4.2. Ablation Study

In Fig. 5, we compare multiple variants of NEAT, varying the following parameters: training seed, semantic class count (K), attention iterations (N), prediction horizon (Z), input sensor count (S), transformer layers (L), and loss weights (γ_1, λ). While a detailed analysis regarding each factor is provided in the supplementary, we focus here on four variants in particular: Firstly, we observe that different random training seeds of NEAT achieve similar performance, which is a desirable property not seen in all end-to-end driving models [4]. Second, as observed by [4], 2D semantic models (such as AIM-VA and AIM-MT) rely heavily on lane marking annotations for strong performance. We observe that these are not needed by NEAT for which the default configuration with 5 classes outperforms the variant that includes lane markings with 6 classes. Third, in the shorter horizon variant ($Z = 2$) with only 2 predicted waypoints, we observe that the output waypoints do not deviate sharply enough from the vertical axis for the PID controller to perform certain maneuvers. It is also likely that the additional supervision provided by having a horizon of $Z = 4$ in our default configuration has a positive effect on performance. Fourth, the gain of the default NEAT model

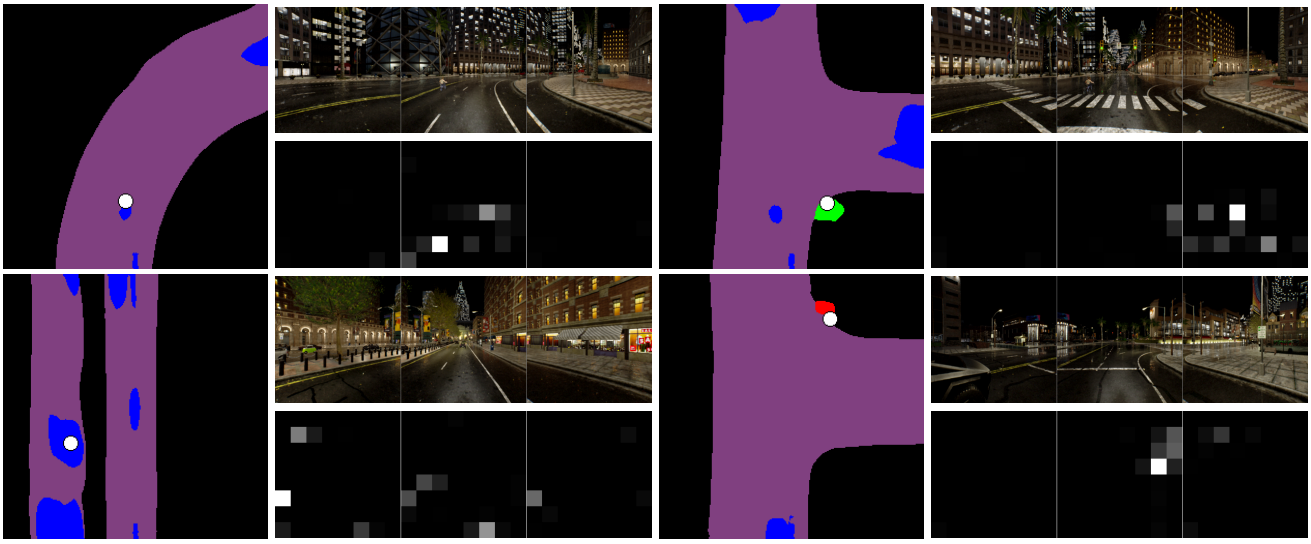


Figure 6: **Attention Maps.** We visualize the semantics $\mathbf{s}_N$ for 4 frames of a driving sequence (legend: none, road, obstacle, red light, green light). We highlight one particular (x, y) location as a white circle on each $\mathbf{s}_N$, for which we visualize the input and corresponding attention map $\mathbf{a}_n$. NEAT consistently attends to the region corresponding to the object of interest (from top left to bottom right: bicyclist, green light, vehicle and red light). Best viewed on screen, zoom in for details.

compared to its version without the semantic loss ($\lambda = 0$) is 30%, showing the benefit of performing BEV semantic prediction and trajectory planning jointly.

Runtime: To analyze the runtime overhead of NEAT's offset prediction task, we now create a hybrid version of AIM and NEAT. This model directly regresses waypoints from NEAT's encoder features $\mathbf{c}_0$ using a GRU decoder (like AIM) instead of predicting offsets. We still use a semantic decoder at train time supervised with only the cross-entropy term of Eq. (5). At test time, the average runtime per frame of the hybrid model (with the semantics head discarded) is 15.92 ms on a 3080 GPU. In comparison, the default NEAT model takes 30.37 ms, i.e., both approaches are real-time even with un-optimized code. Without the compute-intensive red light indicator, NEAT's runtime is only 18.60 ms. Importantly, NEAT (DS = 65.10) significantly outperforms the AIM-NEAT hybrid model (DS = 33.63). This shows that NEAT's attention maps and location-specific features lead to improved waypoint predictions.

4.3. Visualizations

Our supplementary video[2] contains qualitative examples of NEAT's driving capabilities. For the first route in the video, we visualize attention maps $\mathbf{a}_N$ for different locations on the route in Fig. 6. For each frame in the video, we randomly sample BEV (x, y) locations and pass them through the trained NEAT model until one of the locations corresponds to the class obstacle, red light, or green light. Four such frames are shown in Fig. 6. We observe a common trend in the attention maps: NEAT focuses on the image corresponding to the object of interest, albeit sometimes at a slightly different location in the image. This can be attributed to the fact that NEAT's attention maps are over learned image features that capture information aggregated over larger receptive fields. To quantitatively evaluate this property, we extract the 32×32 image patch which NEAT assigns the highest attention weight for one random (x, y) location in each scene of our validation set and analyze its ground truth 2D semantic segmentation labels. The semantic class predicted by NEAT for (x, y) is present in the 2D patch in 79.67% of the scenes.

[2] https://www.youtube.com/watch?v=gtO-ghjKkRs

5. Conclusion

In this work, we take a step towards interpretable, high-performance, end-to-end autonomous driving with our novel NEAT feature representation. Our approach tackles the challenging problem of joint BEV semantic prediction and vehicle trajectory planning from camera inputs and drives with the highest safety among state-of-the-art methods on the CARLA simulator. NEAT is generic and flexible in terms of both input modalities and output task/supervision and we plan to combine it with orthogonal ideas (e.g., DAgger, RL) in the future.

Acknowledgements: This work is supported by the BMBF through the Tübingen AI Center (FKZ: 01IS18039B) and the BMWi in the project KI Delta Learning (project number: 19A19013O). We thank the International Max Planck Research School for Intelligent Systems (IMPRS-IS) for supporting Kashyap Chitta. The authors also thank Micha Schilling for his help in re-implementing AIM-VA.

References

[1] Carla autonomous driving leaderboard. https://leaderboard.carla.org/, 2020. 2, 6, 7

[2] Ammar Abbas and Andrew Zisserman. A geometric approach to obtain a bird's eye view from an image. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV) Workshops*, 2019. 2

[3] Simon Baker, Daniel Scharstein, J. Lewis, Stefan Roth, Michael Black, and Richard Szeliski. A database and evaluation methodology for optical flow. *International Journal of Computer Vision (IJCV)*, 92:1–31, 2011. 3

[4] Aseem Behl, Kashyap Chitta, Aditya Prakash, Eshed Ohn-Bar, and Andreas Geiger. Label efficient visual abstractions for autonomous driving. In *Proc. IEEE International Conf. on Intelligent Robots and Systems (IROS)*, 2020. 2, 6, 7

[5] Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D. Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, Xin Zhang, Jake Zhao, and Karol Zieba. End to end learning for self-driving cars. *arXiv.org*, 1604.07316, 2016. 1, 2

[6] Andreas Bühler, Adrien Gaidon, Andrei Cramariuc, Rares Ambrus, Guy Rosman, and Wolfram Burgard. Driving Through Ghosts: Behavioral Cloning with False Positives. In *Proc. IEEE International Conf. on Intelligent Robots and Systems (IROS)*, 2020. 2

[7] Sergio Casas, Abbas Sadat, and Raquel Urtasun. Mp3: A unified model to map, perceive, predict and plan. *arXiv.org*, 2101.06806, 2021. 2

[8] Rohan Chabra, Jan Eric Lenssen, Eddy Ilg, Tanner Schmidt, Julian Straub, Steven Lovegrove, and Richard Newcombe. Deep local shapes: Learning local sdf priors for detailed 3d reconstruction. In *Proc. of the European Conf. on Computer Vision (ECCV)*, 2020. 2

[9] Dian Chen, Vladlen Koltun, and Philipp Krähenbühl. Learning to drive from a world on rails. *arXiv.org*, 2105.00636, 2021. 2, 7

[10] Dian Chen, Brady Zhou, Vladlen Koltun, and Philipp Krähenbühl. Learning by cheating. In *Proc. Conf. on Robot Learning (CoRL)*, 2019. 1, 2, 3, 4, 6, 7

[11] Zhiqin Chen, Andrea Tagliasacchi, and Hao Zhang. Bsp-net: Generating compact meshes via binary space partitioning. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2

[12] Zhiqin Chen and Hao Zhang. Learning implicit fields for generative shape modeling. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019. 2

[13] Julian Chibane, Thiemo Alldieck, and Gerard Pons-Moll. Implicit functions in feature space for 3d shape reconstruction and completion. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2

[14] Felipe Codevilla, Antonio M. Lopez, Vladlen Koltun, and Alexey Dosovitskiy. On offline evaluation of vision-based driving models. In *Proc. of the European Conf. on Computer Vision (ECCV)*, 2018. 2

[15] Felipe Codevilla, Matthias Miiller, Antonio López, Vladlen Koltun, and Alexey Dosovitskiy. End-to-end driving via conditional imitation learning. In *Proc. IEEE International Conf. on Robotics and Automation (ICRA)*, 2018. 2

[16] Felipe Codevilla, Eder Santana, Antonio M. López, and Adrien Gaidon. Exploring the limitations of behavior cloning for autonomous driving. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*, 2019. 1, 2, 6, 7

[17] Harm de Vries, Florian Strub, Jérémie Mary, Hugo Larochelle, Olivier Pietquin, and Aaron C. Courville. Modulating early visual processing by language. In *Advances in Neural Information Processing Systems (NIPS)*, 2017. 4

[18] Boyang Deng, Kyle Genova, Soroosh Yazdani, Sofien Bouaziz, Geoffrey Hinton, and Andrea Tagliasacchi. Cvxnet: Learnable convex decomposition. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2

[19] Boyang Deng, JP Lewis, Timothy Jeruzalski, Gerard Pons-Moll, Geoffrey Hinton, Mohammad Norouzi, and Andrea Tagliasacchi. Neural articulated shape approximation. In *Proc. of the European Conf. on Computer Vision (ECCV)*, 2020. 2

[20] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. CARLA: An open urban driving simulator. In *Proc. Conf. on Robot Learning (CoRL)*, 2017. 2, 5, 6

[21] Yilun Du, Yinan Zhang, Hong-Xing Yu, Joshua B. Tenenbaum, and Jiajun Wu. Neural radiance flow for 4D view synthesis and video processing. *arXiv.org*, 2012.09790, 2020. 2

[22] Vincent Dumoulin, Ishmael Belghazi, Ben Poole, Olivier Mastropietro, Alex Lamb, Martin Arjovsky, and Aaron Courville. Adversarially learned inference. In *Proc. of the International Conf. on Learning Representations (ICLR)*, 2017. 4

[23] Angelos Filos, Panagiotis Tigas, Rowan McAllister, Nicholas Rhinehart, Sergey Levine, and Yarin Gal. Can autonomous vehicles identify, recover from, and adapt to distribution shifts? In *Proc. of the International Conf. on Machine learning (ICML)*, 2020. 2

[24] Kyle Genova, Forrester Cole, Daniel Vlasic, Aaron Sarna, William T Freeman, and Thomas Funkhouser. Learning shape templates with structured implicit functions. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*, 2019. 2

[25] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2016. 3

[26] Noureldin Hendy, Cooper Sloan, Feng Tian, Pengfei Duan, Nick Charchut, Yuesong Xie, Chuang Wang, and James Philbin. Fishing net: Future inference of semantic heatmaps in grids. *arXiv.org*, 2006.09917, 2020. 2

[27] Anthony Hu, Fergal Cotter, Nikhil Mohan, Corina Gurau, and Alex Kendall. Probabilistic future prediction for video scene understanding. In *Proc. of the European Conf. on Computer Vision (ECCV)*, 2020. 1

[28] Anthony Hu, Zak Murez, Nikhil Mohan, Sofía Dudas, Jeff Hawke, Vijay Badrinarayanan, Roberto Cipolla, and

Alex Kendall. FIERY: future instance prediction in bird's-eye view from surround monocular cameras. *arXiv.org*, 2104.10490, 2021. 2

[29] Zhiyu Huang, Chen Lv, Yang Xing, and Jingda Wu. Multi-modal sensor fusion-based deep neural network for end-to-end autonomous driving with scene understanding. *IEEE Sensors Journal*, 2020. 1, 2

[30] Joel Janai, Fatma Güney, Aseem Behl, and Andreas Geiger. *Computer Vision for Autonomous Vehicles: Problems, Datasets and State of the Art*, volume 12. Foundations and Trends in Computer Graphics and Vision, 2020. 2

[31] Chiyu Jiang, Avneesh Sud, Ameesh Makadia, Jingwei Huang, Matthias Nießner, and Thomas Funkhouser. Local implicit grid representations for 3d scenes. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2

[32] Peilun Li, Xiaodan Liang, Daoyuan Jia, and Eric P. Xing. Semantic-aware grad-gan for virtual-to-real urban scene adaption. *arXiv.org*, 1801.01726, 2018. 6

[33] Tianye Li, Mira Slavcheva, Michael Zollhoefer, Simon Green, Christoph Lassner, Changil Kim, Tanner Schmidt, Steven Lovegrove, Michael Goesele, and Zhaoyang Lv. Neural 3d video synthesis. *arXiv.org*, 2103.02597, 2021. 2

[34] Zhengqi Li, Simon Niklaus, Noah Snavely, and Oliver Wang. Neural scene flow fields for space-time view synthesis of dynamic scenes. *arXiv.org*, 2011.13084, 2020. 2

[35] Shichen Liu, Shunsuke Saito, Weikai Chen, and Hao Li. Learning to infer implicit surfaces without 3d supervision. In *Advances in Neural Information Processing Systems (NIPS)*, 2019. 2

[36] Shaohui Liu, Yinda Zhang, Songyou Peng, Boxin Shi, Marc Pollefeys, and Zhaopeng Cui. DIST: Rendering deep implicit signed distance function with differentiable sphere tracing. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2

[37] Abdelhak Loukkal, Yves Grandvalet, Tom Drummond, and You Li. Driving among Flatmobiles: Bird-Eye-View occupancy grids from a monocular camera for holistic trajectory planning. *arXiv.org*, 2008.04047, 2020. 1, 2

[38] Kaustubh Mani, Swapnil Daga, Shubhika Garg, N. Sai Shankar, Krishna Murthy Jatavallabhula, and K. Madhava Krishna. MonoLayout: Amodal scene layout from a single image. In *Proc. of the IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2020. 2

[39] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019. 2

[40] Mateusz Michalkiewicz, Jhony K Pontes, Dominic Jack, Mahsa Baktashmotlagh, and Anders Eriksson. Implicit surface representations as layers in neural networks. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*, 2019. 2

[41] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. NeRF: Representing scenes as neural radiance fields for view synthesis. In *Proc. of the European Conf. on Computer Vision (ECCV)*, 2020. 2

[42] Matthias Müller, Alexey Dosovitskiy, Bernard Ghanem, and Vladlen Koltun. Driving policy transfer via modularity and abstraction. In *Proc. Conf. on Robot Learning (CoRL)*, 2018. 2

[43] Michael Niemeyer, Lars Mescheder, Michael Oechsle, and Andreas Geiger. Occupancy flow: 4d reconstruction by learning particle dynamics. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*, 2019. 2

[44] Michael Niemeyer, Lars Mescheder, Michael Oechsle, and Andreas Geiger. Differentiable volumetric rendering: Learning implicit 3d representations without 3d supervision. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2

[45] Michael Oechsle, Lars Mescheder, Michael Niemeyer, Thilo Strauss, and Andreas Geiger. Texture fields: Learning texture representations in function space. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*, 2019. 2

[46] Michael Oechsle, Michael Niemeyer, Christian Reiser, Lars Mescheder, Thilo Strauss, and Andreas Geiger. Learning implicit surface light fields. In *Proc. of the International Conf. on 3D Vision (3DV)*, 2020. 2

[47] Eshed Ohn-Bar, Aditya Prakash, Aseem Behl, Kashyap Chitta, and Andreas Geiger. Learning situational driving. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020. 1, 2

[48] T. Osa, J. Pajarinen, G. Neumann, J. A. Bagnell, P. Abbeel, and J. Peters. *An Algorithmic Perspective on Imitation Learning*. 2018. 1

[49] Bowen Pan, Jiankai Sun, Ho Yin Tiga Leung, Alex Andonian, and Bolei Zhou. Cross-view semantic segmentation for sensing surroundings. *IEEE Robotics and Automation Letters (RA-L)*, 2020. 2

[50] Jeong Joon Park, Peter Florence, Julian Straub, Richard A. Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019. 2

[51] Songyou Peng, Michael Niemeyer, Lars Mescheder, Marc Pollefeys, and Andreas Geiger. Convolutional occupancy networks. In *Proc. of the European Conf. on Computer Vision (ECCV)*, 2020. 2

[52] Jonah Philion and Sanja Fidler. Lift, splat, shoot: Encoding images from arbitrary camera rigs by implicitly unprojecting to 3d. In *Proc. of the European Conf. on Computer Vision (ECCV)*, 2020. 2

[53] Dean Pomerleau. ALVINN: an autonomous land vehicle in a neural network. In *Advances in Neural Information Processing Systems (NIPS)*, 1988. 1, 2

[54] Aditya Prakash, Aseem Behl, Eshed Ohn-Bar, Kashyap Chitta, and Andreas Geiger. Exploring data aggregation in policy learning for vision-based urban autonomous driving. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2, 7

[55] Aditya Prakash, Kashyap Chitta, and Andreas Geiger. Multi-modal fusion transformer for end-to-end autonomous driv-

ing. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2021. 6, 7

[56] Thomas Roddick and Roberto Cipolla. Predicting semantic map representations from images using pyramid occupancy networks. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2

[57] Stéphane Ross, Geoffrey J. Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Conference on Artificial Intelligence and Statistics (AISTATS)*, 2011. 7

[58] Abbas Sadat, Sergio Casas, Mengye Ren, Xinyu Wu, Pranaab Dhawan, and Raquel Urtasun. Perceive, predict, and plan: Safe motion planning through interpretable semantic representations. In *Proc. of the European Conf. on Computer Vision (ECCV)*, 2020. 1, 2

[59] Shunsuke Saito, Zeng Huang, Ryota Natsume, Shigeo Morishima, Angjoo Kanazawa, and Hao Li. Pifu: Pixel-aligned implicit function for high-resolution clothed human digitization. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*, 2019. 2

[60] Axel Sauer, Nikolay Savinov, and Andreas Geiger. Conditional affordance learning for driving in urban environments. In *Proc. Conf. on Robot Learning (CoRL)*, 2018. 2

[61] Katja Schwarz, Yiyi Liao, Michael Niemeyer, and Andreas Geiger. Graf: Generative radiance fields for 3d-aware image synthesis. In *Advances in Neural Information Processing Systems (NIPS)*, 2020. 2

[62] Vincent Sitzmann, Eric R. Chan, Richard Tucker, Noah Snavely, and Gordon Wetzstein. Metasdf: Meta-learning signed distance functions. In *Advances in Neural Information Processing Systems (NIPS)*, 2020. 2

[63] Vincent Sitzmann, Julien N.P. Martel, Alexander W. Bergman, David B. Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. In *Advances in Neural Information Processing Systems (NIPS)*, 2020. 2

[64] Vincent Sitzmann, Michael Zollhöfer, and Gordon Wetzstein. Scene representation networks: Continuous 3d-structure-aware neural scene representations. In *Advances in Neural Information Processing Systems (NIPS)*, 2019. 2

[65] Ardi Tampuu, Maksym Semikin, Naveed Muhammad, Dmytro Fishman, and Tambet Matiisen. A survey of end-to-end driving: Architectures and training methods. *arXiv.org*, 2003.06404, 2020. 2

[66] Marin Toromanoff, Emilie Wirbel, and Fabien Moutarde. End-to-end model-free reinforcement learning for urban driving using implicit affordances. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2, 7

[67] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems (NIPS)*, pages 5998–6008, 2017. 3

[68] J. Walker, A. Gupta, and M. Hebert. Dense optical flow prediction from a static image. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*, 2015. 3

[69] Dequan Wang, Coline Devin, Qi-Zhi Cai, Philipp Krähenbühl, and Trevor Darrell. Monocular plan view networks for autonomous driving. In *Proc. IEEE International Conf. on Intelligent Robots and Systems (IROS)*, 2019. 3

[70] Jingke Wang, Yue Wang, Dongkun Zhang, Yezhou Yang, and Rong Xiong. Learning hierarchical behavior and motion planning for autonomous driving. *arXiv.org*, 2005.03863, 2020. 2

[71] Nicholas Watters, Loïc Matthey, Christopher P. Burgess, and Alexander Lerchner. Spatial broadcast decoder: A simple architecture for learning disentangled representations in vaes. In *Proc. of the International Conf. on Learning Representations (ICLR) Workshops*, 2019. 6

[72] Bob Wei, Mengye Ren, Wenyuan Zeng, Ming Liang, Bin Yang, and Raquel Urtasun. Perceive, attend, and drive: Learning spatial attention for safe self-driving. *arXiv.org*, 2011.01153, 2020. 2

[73] Chuan Wen, Jierui Lin, Trevor Darrell, Dinesh Jayaraman, and Yang Gao. Fighting copycat agents in behavioral cloning from observation histories. In *Advances in Neural Information Processing Systems (NIPS)*, 2020. 3

[74] Wenqi Xian, Jia-Bin Huang, Johannes Kopf, and Changil Kim. Space-time neural irradiance fields for free-viewpoint video. *arXiv.org*, 2011.12950, 2020. 2

[75] Yi Xiao, Felipe Codevilla, Akhil Gurram, Onay Urfalioglu, and Antonio M. López. Multimodal end-to-end autonomous driving. *IEEE Trans. on Intelligent Transportation Systems (TITS)*, 2020. 2

[76] Yi Xiao, Felipe Codevilla, Christopher Pal, and Antonio M. López. Action-Based Representation Learning for Autonomous Driving. In *Proc. Conf. on Robot Learning (CoRL)*, 2020. 2

[77] Qiangeng Xu, Weiyue Wang, Duygu Ceylan, Radomír Mech, and Ulrich Neumann. DISN: deep implicit surface network for high-quality single-view 3d reconstruction. In *Advances in Neural Information Processing Systems (NIPS)*, 2019. 2

[78] Wenyuan Zeng, Wenjie Luo, Simon Suo, Abbas Sadat, Bin Yang, Sergio Casas, and Raquel Urtasun. End-to-end interpretable neural motion planner. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019. 1, 2

[79] Ruohan Zhang, Zhuode Liu, Luxin Zhang, Jake A. Whritner, Karl S. Muller, Mary M. Hayhoe, and Dana H. Ballard. Agil: Learning attention from human for visuomotor tasks. In *Proc. of the European Conf. on Computer Vision (ECCV)*, 2018. 2

[80] Albert Zhao, Tong He, Yitao Liang, Haibin Huang, Guy Van den Broeck, and Stefano Soatto. Sam: Squeeze-and-mimic networks for conditional visual driving policy learning. In *Proc. Conf. on Robot Learning (CoRL)*, 2020. 2

[81] Xinge Zhu, Zhichao Yin, Jianping Shi, Hongsheng Li, and Dahua Lin. Generative adversarial frontal view to bird view synthesis. In *Proc. of the International Conf. on 3D Vision (3DV)*, 2018. 2