A Graph Representation for Autonomous Driving

Zerong Xi
Department of Computer Science
University of Central Florida
Orlando, FL
xizerong@gmail.com

Gita Sukthankar
Department of Computer Science
University of Central Florida
Orlando, FL
gitars@eecs.ucf.edu

Abstract

For human drivers, an important aspect of learning to drive is knowing how to pay attention to areas of the roadway that are critical for decision-making while simultaneously ignoring distractions. Similarly, the choice of roadway representation is critical for good performance of an autonomous driving system. An effective representation should be compact and permutation-invariant, while still representing complex vehicle interactions that govern driving decisions. This paper introduces the Graph Representation for Autonomous Driving (GRAD); GRAD generates a global scene representation using a space-time graph which incorporates the estimated future trajectories of other vehicles. We demonstrate that GRAD outperforms the best performing social attention representation on a simulated highway driving task in high traffic densities and also has a low computational complexity in both single and multi-agent settings.

1 Introduction

Attention is governed by both endogenous (top-down) and exogenous (stimulus-driven) cognitive processes that empower human drivers to selectively scrutinize the regions of the roadway most relevant to their planned trajectory [1]. In higher density traffic, stimulus-driven control becomes harder since there are more potential distractions on the road; thus the internal representation used by the agent must scale to larger numbers of objects by selectively focusing on the objects most likely to affect the driver's decision-making. One way that this can be achieved is by employing an egocentric foveal representation; however, this does not scale well to cooperative multi-agent driving systems. This paper introduces a new representation, GRAD (Graph Representation for Autonomous Driving) that possesses several key desiderata: 1) it is global, enabling a single representation to be shared across multiple agents; 2) it incorporates the future location of neighboring vehicles into the decision-making process; 3) it is more computationally efficient than attention mechanisms; 4) the same representation is applicable to both driving and trajectory prediction tasks.

Leurent and Mercat [2] identify roadway representation as a key problem for autonomous driving and note that most behavioral planning systems either employ a list of features for every vehicle or a spatial occupancy grid to represent traffic on the road. They propose a social attention model [2] which represents egocentric dependencies between the driver and surrounding vehicles; it handles varying length inputs and is permutation invariant. Their social attention technique uses a multi-headed attention model that focuses on vehicles that can potentially interfere with the planned route. This paper compares our proposed GRAD feature representation vs. social attention as an input to an RL agent that learns a highway driving policy for dense traffic scenarios using Proximal Policy Optimization (PPO) [3].

Our paper makes the following contributions:

Machine Learning for Autonomous Driving Workshop at the 36th Conference on Neural Information Processing Systems (NeurIPS 2022), New Orleans, USA.

1. We introduce GRAD (Graph Representation for Autonomous Driving): a global, general representation that can be shared across multiple agents and used for both driving and prediction tasks.
2. We demonstrate that GRAD outperforms the social attention model which is a top performer at driving in the *highway-env* simulation environment and possesses a comparable number of trainable parameters.
3. We present an analysis how modifying the space-time graph used by GRAD affects driving performance.

2 Related Work

Schwarting et al. [4] provide a comprehensive related work overview on autonomous driving that includes research on the myriad systems required to deploy an intelligent driving robot including integrated perception and planning, vehicle dynamics and control, autonomy, and safety verification; however our work is mainly relevant for learning systems that require a bird's-eye roadway representation.

Learning to Drive. Neural network architectures for highway driving have improved substantially since the initial deployment of ALVINN (Autonomous Land Vehicle In a Neural Network) on CMU's Navlab vehicle [5]. Many of the systems have leveraged demonstrations from human drivers using techniques such as behavioral cloning [6] and inverse reinforcement learning [7]. We demonstrate the usage of our representation as a feature extractor for a discrete control system using Proximal Policy Optimization. Saxena et al. [8] learn a continuous controller with PPO for driving in dense traffic; they use an occupancy grid representation parameterized by longitudinal field of view. In flavor their representation is very similar to the grid occupancy representation used by Leurent and Mercat [2] as a benchmark for their social attention model. Leurent and Mercat demonstrate that their social attention representation outperforms both a vanilla PPO implementation and a spatial grid plus convolutional neural network.

Motion Prediction. A key innovation of GRAD is the usage of a space-time graph that incorporates future estimations of neighboring vehicle positions or trajectories into the roadway representation. There are several competitions that explicitly test the problem of driving trajectory prediction, which requires modeling both the intent of the drivers (what they intend to do) and control uncertainty (how they intend to do it). The MultiPath [9] technique does this by factoring intent uncertainty into a distribution over trajectory anchors with normally distributed control uncertainty. MultiPath++ [10] (the leader in the Waymo Open Dataset Motion Prediction Challenge) improves on MultiPath by modifying the state representation in several ways including incorporating context awareness to represent relationships between vehicles. Similar to our work, LaneGCN [11] uses a graph to represent vehicle interactions for motion prediction on the Argoverse motion prediction benchmark; however much of their graph is devoted to lane interactions which are not used in our model.

3 Graph Representation for Autonomous Driving

3.1 Graph Transformer

As graphs are a flexible and powerful representation for numerous real-world datasets, including social networks, citation data, and biological structures, there have been significant research efforts dedicated to incorporating graph representations into neural networks [12, 13, 14, 15]. This paper is an attempt to apply graph neural networks to the autonomous driving domain.

In our proposed architecture, edge attributes play an important role in encoding the locality, which requires the graph neural network to be capable of exploiting edge attributes. Shi et al. [15] introduce a graph transformer operator where the edge attributes are utilized for both attention computation and feature aggregation; Fey and Lenssen [16] provide a neat implementation of the operator. Given a graph $(X, \mathcal{E})$ where X contains node features and $\mathcal{E}$ denotes edge features, the i-th node is updated as follows [16]

$$\mathbf{x}_i' = \mathbf{W}_1\mathbf{x}_i + \sum_{j \in \mathcal{N}(i)} \alpha_{i,j}(\mathbf{W}_2\mathbf{x}_j + \mathbf{W}_3\mathbf{e}_{i,j}) \tag{1}$$

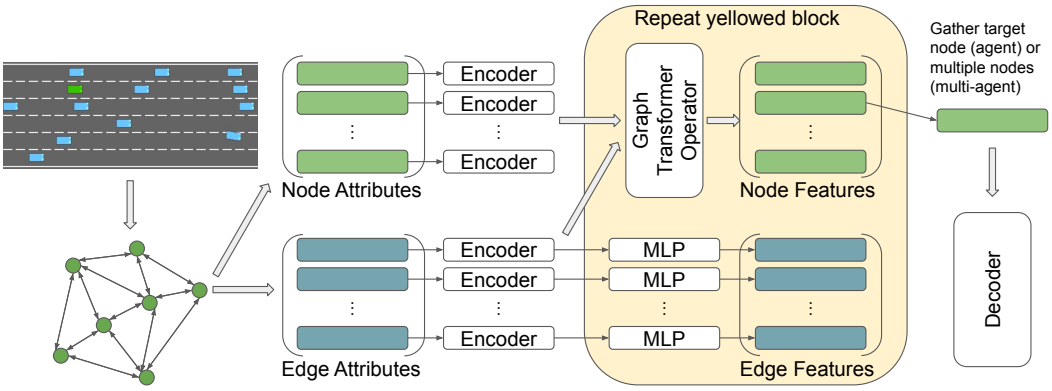


Figure 1: Global Graph Representation

where the attention coefficient $\alpha_{i,j}$ is

$$\alpha_{i,j} = \text{softmax}\left(\frac{(\mathbf{W}_4\mathbf{x}_i)(\mathbf{W}_5\mathbf{x}_j + \mathbf{W}_6\mathbf{e}_{i,j})}{\sqrt{d}}\right) \tag{2}$$

d (feature dimension) is used to rescale attention appropriately.

3.2 Global Graph Representation

The architecture of GRAD is depicted in Figure 1. A graph representation is well suited for selectively limiting attention since each agent perceives the interacting objects accurately via agent-centric attributes through direct links and the farther ones via the information flow over graph convolutional operators.

Given a road scene, an observation is a collection of road objects (i.e. vehicles if modeling vehicle interactions only), $X \in \mathbb{R}^{N \times d_x}$ where N is the number of objects and d_x is the dimensions of features. It contains the localization information for each of the objects, including coordinates, velocity, and acceleration. By applying a distance metric (as described in Section 3.3) on the localization information, a directed graph $(X, \mathcal{L})$ is constructed with objects as nodes, and links limited by either a maximum number of nearest neighbors or a fixed distance to connecting nodes. For each link connecting node i and node j, the edge attributes $\mathcal{E}_{i,j}$ are populated with relative information between the pair, including relative position, velocity, and acceleration. Note that objects are linked by a pair of directional links with different edge attributes. Therefore, from the viewpoint of object i, $\mathcal{E}_{i,j\in\mathcal{L}_i}$ contains the agent-centric information of a small surrounding sub-region.

Depending on whether the history is used, either MLP or sequence-based neural networks are applied to each node X_i and each edge $\mathcal{E}_{i,j}$ respectively. Then the encoded graph $(X', \mathcal{L}, \mathcal{E}')$ is passed through multiple blocks of graph transformer operators [15] which are capable of utilizing the edge attributes as shown in Section 3.1. While the operators commonly update only the nodes, the edge features are updated using a separate MLP within each block. The details of the block are shown in the yellow box in Figure 1. The output retains the original graph structure.

It worth noting that the architecture in Figure 1 is an "overkill" version which can be plugged into multi-agent settings or used directly as global scene representation. For the single agent configuration described in this paper, the computation can be reduced to be within the scope of the single target node (driving agent), which can be visualized as a pyramid over multiple graph convolutional layers. This only requires updating the target node representing the driving agent by making queries on the nodes directly linked to it in the last graph convolutional layer.

3.3 Space-Time Graph

In the first step of vanilla GRAD, a graph is built by linking the neighboring nodes based on their distance measurements, i.e. Euclidean distance on synchronous positions in the past and current time steps. To further concentrate direct attention on the objects that are highly likely to be involved in

Algorithm 1 Space-Time Graph

Require: coordinates $\mathbf{x}$, velocities $\mathbf{v}$, accelerations $\mathbf{a}$, horizon T, discount γ

function WAYPOINT($\mathbf{x_1}, \mathbf{v_1}, \mathbf{a_1}, \mathbf{x_2}, \mathbf{v_2}, \mathbf{a_2}, T, \gamma$)
  for $t = 0...T$ **do**
    ${\mathbf{x_1}}^t = \mathbf{x_1} + \mathbf{v_1} \cdot t + \mathbf{a_1} \cdot t^2/2$
    ${\mathbf{x_2}}^t = \mathbf{x_2} + \mathbf{v_2} \cdot t + \mathbf{a_2} \cdot t^2/2$
    $d^t = \|\mathbf{x_1} - \mathbf{x_2}\| \cdot \gamma^t$
  end for
  $d = \min_{1 \le t \le T} d^t$
  return d
end function

function TRAJECTORY($\mathbf{x_1}, \mathbf{v_1}, \mathbf{a_1}, \mathbf{x_2}, \mathbf{v_2}, \mathbf{a_2}, T, \gamma$)
  for $t = 0...T$ **do**
    ${\mathbf{x_1}}^t = \mathbf{x_1} + \mathbf{v_1} \cdot t + \mathbf{a_1} \cdot t^2/2$
    ${\mathbf{x_2}}^t = \mathbf{x_2} + \mathbf{v_2} \cdot t + \mathbf{a_2} \cdot t^2/2$
  end for
  for $t = 1...T$ **do**
    ${\mathbf{l_1}}^t = ({\mathbf{x_1}}^{t-1}, {\mathbf{x_1}}^t)$
    ${\mathbf{l_2}}^t = ({\mathbf{x_2}}^{t-1}, {\mathbf{x_2}}^t)$
  end for
  for $s = 1...T$ **do**
    for $t = 1...T$ **do**
      ▷ Euclidean distance between line segments
      $d^{s,t} = \|{\mathbf{l_1}}^s, {\mathbf{l_2}}^t\|_l$
    end for
  end for
  $d = \min_{1 \le s,t \le T} d^{s,t} \cdot \gamma^{|s-t|}$
  return d
end function

future interactions, we propose a Space-Time Graph which builds the graph not merely on the current localization but also the estimated future positions or trajectories.

In the simplest form, given the current localization $(\mathbf{x}_i, \mathbf{v}_i, \mathbf{a}_i)$ of node i, the future position at time step t is estimated with the motion equation

$$\mathbf{x}_i^t = \mathbf{x}_i + \mathbf{v}_i \cdot t + \mathbf{a}_i \cdot t^2/2 \tag{3}$$

and the estimated trajectory $\mathcal{T} = (\mathbf{x}_i^0, \mathbf{x}_i^1, ..., \mathbf{x}_i^T)$ where T is the time horizon.

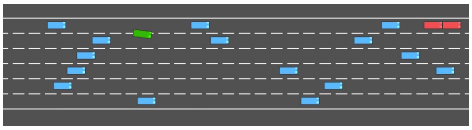
When working with positions, as shown as WAYPOINT function in Algorithm 1, the distance is the minimum measurement between a pair of coordinates sampled synchronously, that is

$$d_{i,j}^t = \|\mathbf{x}_i^t - \mathbf{x}_j^t\| \tag{4}$$

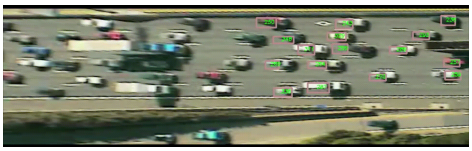
$$d_{i,j} = \min_{0 \le t \le T} d_{i,j}^t \cdot \gamma^t \tag{5}$$

where $\gamma \ge 1$ is a discount factor to reflect the higher importance of the nearer future. While being vulnerable to the change of motion status, it performs reasonably well with a limited time horizon. The position estimations can be further improved with more sophisticated prediction methods; for instance, the graph can be constructed cyclically and incrementally along with predictions for behavior prediction tasks.

Even without more accurate estimations, the robustness can be improved by measuring the distance between trajectories instead of positions, shown as the TRAJECTORY function in Algorithm 1. Given the trajectory $\mathcal{T} = (\mathbf{x}_i^0, \mathbf{x}_i^1, ..., \mathbf{x}_i^T)$, its representation is translated to line segments $\mathcal{T} = (\mathbf{l}_i^1, \mathbf{l}_i^2, ..., \mathbf{l}_i^T)$ where $\mathbf{l}_i^t = (\mathbf{x}_i^{t-1}, \mathbf{x}_i^t)$. The distance is the minimum measurement among pairwise



(a) Simulated driving scene in highway-env [17]



(b) Realistic driving scene in NGSIM dataset [18]

Figure 2: Comparison of simulated and realistic driving scene on highway.

line segments between the trajectories of a connecting pair of nodes, that is

$$d_{i,j}^{s,t} = \|\mathbf{l_i}^s, \mathbf{l_j}^t\|_l, s = 1...T, t = 1...T \tag{6}$$

$$d_{i,j} = \min_{0 \le s,t \le T} d_{i,j}^{s,t} \cdot \gamma^{|s-t|} \tag{7}$$

where $\gamma \geq 1$ is a discount factor to penalize the pairs of line segments which are expected to occur during more distant time frames. The hyperparameters used with GRAD are shown in Table 3.

4 Experiments

4.1 Task

We validate the proposed graph representation within a reinforcement learning framework performing a simulated highway driving task. GRAD is plugged into Proximal Policy Optimization (PPO) [3] as the feature extractor shared by both policy and value networks. Because of the plug-and-play property of GRAD, it can easily be paired up with other RL algorithms and other tasks such as behavior prediction. We have tested with DQN as well and observed consistent results to those shown in Section 4.3 for PPO. Our PPO implementation is described in Table 4.

Highway-env [17] is a simulated environment of highway driving scenarios for behavioral decision-making tasks. We use it to create new highway driving scenarios, but it is also possible to replay data from existing datasets such as NGSIM [18]. In sparse scenes with few vehicles, many algorithms perform comparably well, and both multi-headed attention and the GRAD representation degenerate to simply focusing attention on the few cars. Therefore, we conduct the experiments on a highway scenario where the agent drives forward on an one-way multi-lane road with heavy traffic density. The parameters for our *highway-env* experiments are given in Table 2, and Figure 2 shows the simulator.

The environment is configured to receive inputs at 2 frames per second within a maximum of 30 seconds, unless the agent crashes, which terminates the episode immediately. Therefore, the episode length is 60 at maximum. The scene is configured to contain 6 lanes of the same heading direction and 100 vehicles on road.

Observation The observations are the kinematics status of the agent and the surrounding vehicles, i.e. {coordinates, velocity, heading}. The maximum number of observed surrounding vehicles is 32, and their speed ranges from {15, 25}.

Action The action space is a discrete set {OneLaneLeft, Idle, OneLaneRight, SpeedLevelUp, SpeedLevelDown}, in which the speed levels are {20, 25, 30, 35, 40}. The actions are executed by a layer of speed and steering controller which is included in *highway-env* on top of the continuous low-level control.

Reward The reward consists of 1) a reward of 0.2 for surviving each time step, 2) a reward linearly mapped from the driving speed of (20, 40) to (0, 0.8). Therefore, the maximum reward is 1.0 per action step and is 60.0 per episode given that the maximum episode length is 60.

Each model is trained for 500k action steps, and evaluated for 24 episodes with deterministic policies every 10k training steps.

4.2 Baseline

The social attention (SA) technique [2] models agent interactions using a multi-headed attention mechanism. By default it uses the agent-centric coordinate system for surrounding vehicles, while

Table 1: Model Characteristics

Architecture	Social Attention	Graph Representation
Layer sizes	Encoder: [192, 192, 192] Attention: [192] Heads: 2 Decoder: [256, 256]	Encoder: [128, 128] Attention: [128, 128] Heads: 2 Decoder: [256, 256]
Number of parameters	∼770k	∼720k
Queries per node	All nodes	Linked nodes
Permutation invariant	Yes	Yes
Coordinate system	Agent-centric	Global

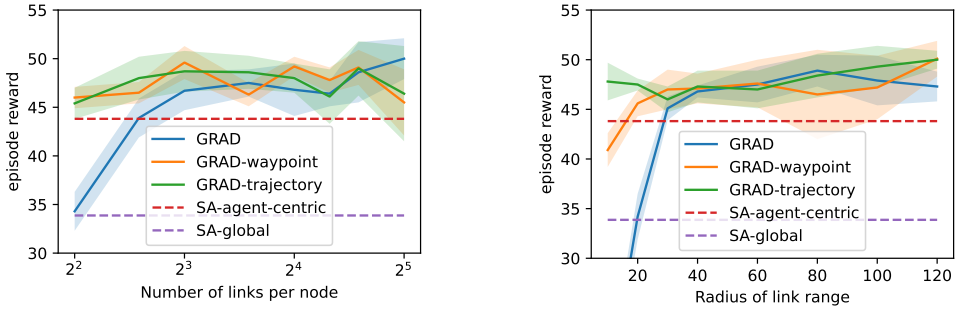


Figure 3: Episode rewards of best policy with limited number of links per node (left) or limited link range (right). Each data point is an average of 5 independent runs, and the shadows show the standard deviations. All versions of GRAD with a sufficient number of links outperform both types of social attention (SA), marked by the dotted lines.

representing the agent itself in global coordinates. Therefore, it has separate encoders for the agent and the other vehicles, and applies cross attention to allow the agent make queries on the surrounding vehicles.

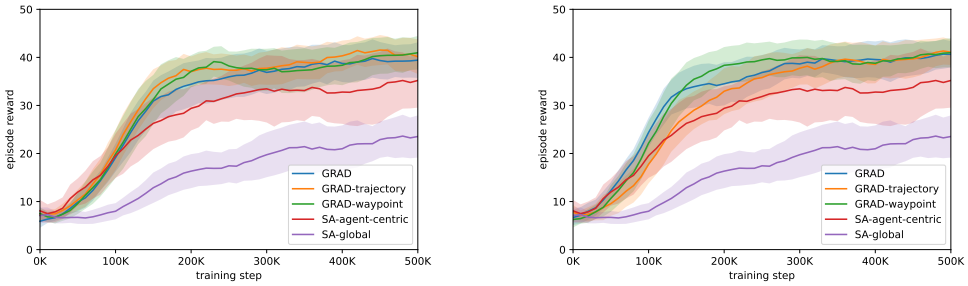
As our method works directly in the global coordinate system, which is easier to extend to multi-agent tasks without excessive additional computations, we compare GRAD to both the original SA (SA-agent-centric) and SA with the global coordinate system (SA-global).

To make a fair comparison, the neural networks in both our method and the baseline have the same number of layers and a similar number of parameters. The characteristics of the models are shown in Table 1.

4.3 Results

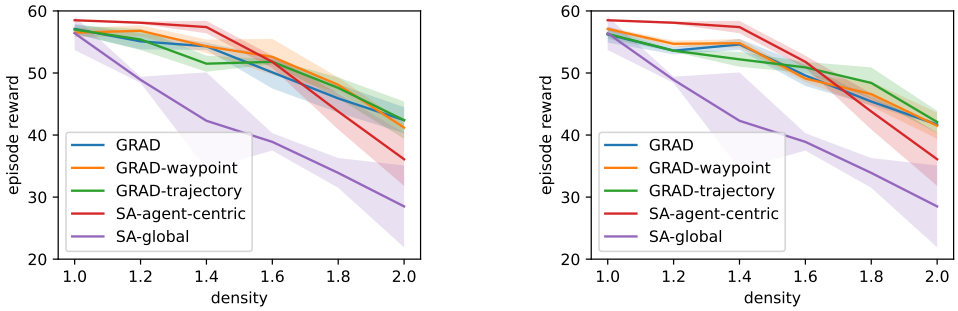
A multi-head attention mechanism can be viewed as equivalent to a version of the full graph GRAD without edge attributes where each of the nodes is allowed to make queries on all other nodes. However, we show that GRAD can achieve comparable performance with a sparser graph. Our experiments demonstrate that it outperforms SA consistently with sufficient links. GRAD has two advantages: 1) it utilizes a mixture of global and agent-centric coordinate systems, 2) it has two attention layers while SA has only one.

Sparse graph To demonstrate that the accurate agent-centric information of farther objects is unnecessary, we experiment over different levels of graph sparsity by limiting either the maximum number of links per node or the maximum distance of connecting nodes. As shown in Figure 3, GRAD achieves considerable performance with merely 8 links per node at maximum, or within a radius of 30 which is approximately the distance the agent travels in one second. Furthermore, GRAD with Space-Time Graph suffers much less from the harsher limitation on links as it concentrates the direct attention on most important objects. This difference in performance converges as this limitation is mitigated.



(a) Number of neighbors = 8 (b) Radius = 40

Figure 4: Episode rewards in evaluation over training steps. Each data point is an average of 5 independent runs and smoothed within a window size of 9. The shadows are the standard deviations of the runs.



(a) Number of neighbors = 8 (b) Radius = 40

Figure 5: Episode rewards of best policy trained and evaluated with various traffic densities. Each data point is an average of 5 independent runs, and the shadow shows the standard deviations. GRAD is less affected by denser traffic as shown by its gentle slope compared to SA.

Convergence speed A good representation needs to provide necessary information in a form that the model can easily use. As shown in Figure 4, GRAD and its variants generally converge in fewer than 60% training steps compared to SA. We believe that these differences in convergence speed and sample efficiency might be even more accentuated in prediction tasks.

Traffic density To study the expressive power of crowded scenes, we evaluate our method and the baseline in various simulated densities of traffic. Higher density traffic both makes the task more difficult and reduces the amount of episode rewards. For example, the agent has to slow down to wait for space to overtake in some states, which makes it impossible to reap the full speed reward. In Figure 5, we observe that GRAD shows greater robustness when faced with denser traffic; it can be seen that the slopes are much gentler than those of the baselines. This is critical for driving in the real-world where traffic density can be quite high.

5 Complexity Analysis

As the GRAD architecture operates directly in the global coordinate system, the representation remains symmetrical for all nodes regardless of the agent or the surrounding vehicles. Thus it's straightforward to extend to cooperative multi-agent tasks, such as multi-agent planning or behavior prediction, without excessive additional computations. GRAD has the exact same complexity for either single or multi-agent settings. As a comparison, agent-centric based methods [2, 10] must centralize the coordinates of all objects and execute the neural networks for each agent of interest individually.

Attention mechanisms [19] update each node using information aggregated across all nodes, which intrinsically has a computational complexity of $\mathcal{O}(n^2)$ for an input size of n, unless we merely allow the target node to make one-time queries in a single layer setting like social attention [2]. The empirical results show that this can only work well in agent-centric coordinate systems, which increases the complexity to $\mathcal{O}(n^2 \cdot m)$ for m agents in multi-agent settings.

As GRAD is capable of generating a global scene representation using efficient implementations such as k-d [20] or ball trees [21], the graph can be built within $\mathcal{O}(k \cdot n \log n)$ time where k is the maximum number of links per node. The Space-Time Graph does not necessarily easily fit into this framework, but it is still able to retain the same computational complexity by refinements on top of the Euclidean distance. The graph transformer operations require $\mathcal{O}(k)$ per node adding up to $\mathcal{O}(k \cdot n)$ for all nodes. Therefore, GRAD exhibits a computational complexity of $\mathcal{O}(k \cdot n \log n)$ for either single or multi-agent settings.

6 Limitations

This work is focused on modeling the interactions among vehicles. *Highway-env* [17] provides a relatively simple environment for concept validation that has the following limitations: 1) it includes no agent types other than vehicles and 2) road information is implicitly embedded in the coordinate system. However, it's important to have a comprehensive perception of the environment in the real world, including more complex roadway structures, traffic signals, and diverse agent types such as cyclists and pedestrians. It is straightforward to plugin GRAD into a larger architecture like Multipath++ [10] to model the agent interaction component. However, this obviously does not utilize the full potential of our graph representation.

Instead, there are two ways to integrate every object on the road into GRAD: 1) mark each category of object as a type of node and apply heterogeneous graph convolutional operators like [22]; 2) map various types of objects into the same embedding space.

To further compress the computation to handle a larger number of objects, GRAD can benefit from dimensionality reduction operators such as pooling. One possible solution is to gradually reduce the dimensions over graph convolutional layers, and to have area-level representations instead of node-level ones for the agents to make queries on. Assuming that only the agents are actively in motion, another possible solution is to apply graph convolutional operators on agents only, which effectively permits only the agent types to perform queries on all types of objects.

7 Conclusion

This paper introduces the GRAD representation for autonomous driving systems that rely on bird's-eye roadway information. GRAD is a global, permutation-invariant representation that retains the same computation complexity in single or multi-agent settings. A key innovation is GRAD's usage of a space-time graph that incorporates estimated future trajectories of neighboring vehicles. Each output node of GRAD functions as a high-level aggregation of local scene information. Our empirical results on using GRAD to learn highway driving policies with PPO show that GRAD is both efficient and robust to crowded traffic conditions.

References

[1] L. YC, L. JD, and B. LN, "Visual attention in driving: the effects of cognitive load and visual disruption," *Human Factors*, vol. 49, pp. 721–733, 2007.

[2] E. Leurent and J. Mercat, "Social Attention for Autonomous Decision-Making in Dense Traffic," Nov. 2019.

[3] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017.

[4] W. Schwarting, J. Alonso-Mora, and D. Rus, "Planning and decision-making for autonomous vehicles," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 1, no. 1, pp. 187–210, 2018.

[5] D. A. Pomerleau, "ALVINN: An autonomous land vehicle in a neural network," *Advances in Neural Information Processing Systems*, vol. 1, 1988.

[6] F. Codevilla, M. Müller, A. López, V. Koltun, and A. Dosovitskiy, "End-to-end driving via conditional imitation learning," in *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 4693–4700.

[7] D. Silver, J. A. Bagnell, and A. Stentz, "Learning autonomous driving styles and maneuvers from expert demonstration," in *Experimental Robotics*. Springer, 2013, pp. 371–386.

[8] D. M. Saxena, S. Bae, A. Nakhaei, K. Fujimura, and M. Likhachev, "Driving in dense traffic with model-free reinforcement learning," in *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 5385–5392.

[9] Y. Chai, B. Sapp, M. Bansal, and D. Anguelov, "MultiPath: Multiple Probabilistic Anchor Trajectory Hypotheses for Behavior Prediction," in *Proceedings of the Conference on Robot Learning*. PMLR, May 2020, pp. 86–99.

[10] B. Varadarajan, A. Hefny, A. Srivastava, K. S. Refaat, N. Nayakanti, A. Cornman, K. Chen, B. Douillard, C. P. Lam, D. Anguelov, and B. Sapp, "MultiPath++: Efficient Information Fusion and Trajectory Aggregation for Behavior Prediction," Dec. 2021.

[11] M. Liang, B. Yang, R. Hu, Y. Chen, R. Liao, S. Feng, and R. Urtasun, "Learning lane graph representations for motion forecasting," in *European Conference on Computer Vision*. Springer, 2020, pp. 541–556.

[12] T. N. Kipf and M. Welling, "Semi-Supervised Classification with Graph Convolutional Networks," in *International Conference on Learning Representations*. OpenReview.net, 2017.

[13] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph Attention Networks," in *International Conference on Learning Representations*, Feb. 2018.

[14] S. Brody, U. Alon, and E. Yahav, "How attentive are graph attention networks?" in *International Conference on Learning Representations*, 2022.

[15] Y. Shi, Z. Huang, S. Feng, H. Zhong, W. Wang, and Y. Sun, "Masked Label Prediction: Unified Message Passing Model for Semi-Supervised Classification," in *Proceedings of the International Joint Conference on Artificial Intelligence*, Aug. 2021, pp. 1548–1554.

[16] M. Fey and J. E. Lenssen, "Fast graph representation learning with PyTorch Geometric," in *ICLR Workshop on Representation Learning on Graphs and Manifolds*, 2019.

[17] E. Leurent, "An environment for autonomous driving decision-making," https://github.com/eleurent/highway-env, 2018.

[18] V. Alexiadis, J. Colyar, J. Halkias, R. Hranac, and G. McHale, "The Next Generation Simulation Program," *Institute of Transportation Engineers. ITE Journal*, vol. 74, no. 8, pp. 22–26, Aug. 2004.

[19] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30. Curran Associates, Inc., 2017.

[20] J. L. Bentley, "Multidimensional binary search trees used for associative searching," *Communications of the ACM*, vol. 18, no. 9, p. 509–517, Sep 1975.

[21] S. M. Omohundro, "Five balltree construction algorithms," International Computer Science Institute, Tech. Rep. TR-89-063, December 1989.

[22] Z. Hu, Y. Dong, K. Wang, and Y. Sun, *Heterogeneous Graph Transformer*. New York, NY, USA: Association for Computing Machinery, 2020, p. 2704–2710.

A Hyperparameters

Table 2: Highway-env

Parameter	Value
Number of visible vehicles	32
See vehicles behind	False
Action type	Discrete meta action
Speed levels	{20, 25, 30, 35, 40}
Episode length	60 = 2 fps $\times$ 30s
Number of road lanes	6
Number of vehicles	100
Traffic density	1.8
Rewarded speed range	[20, 40]
Reward for surviving	0.2
Maximum reward for speed	0.8

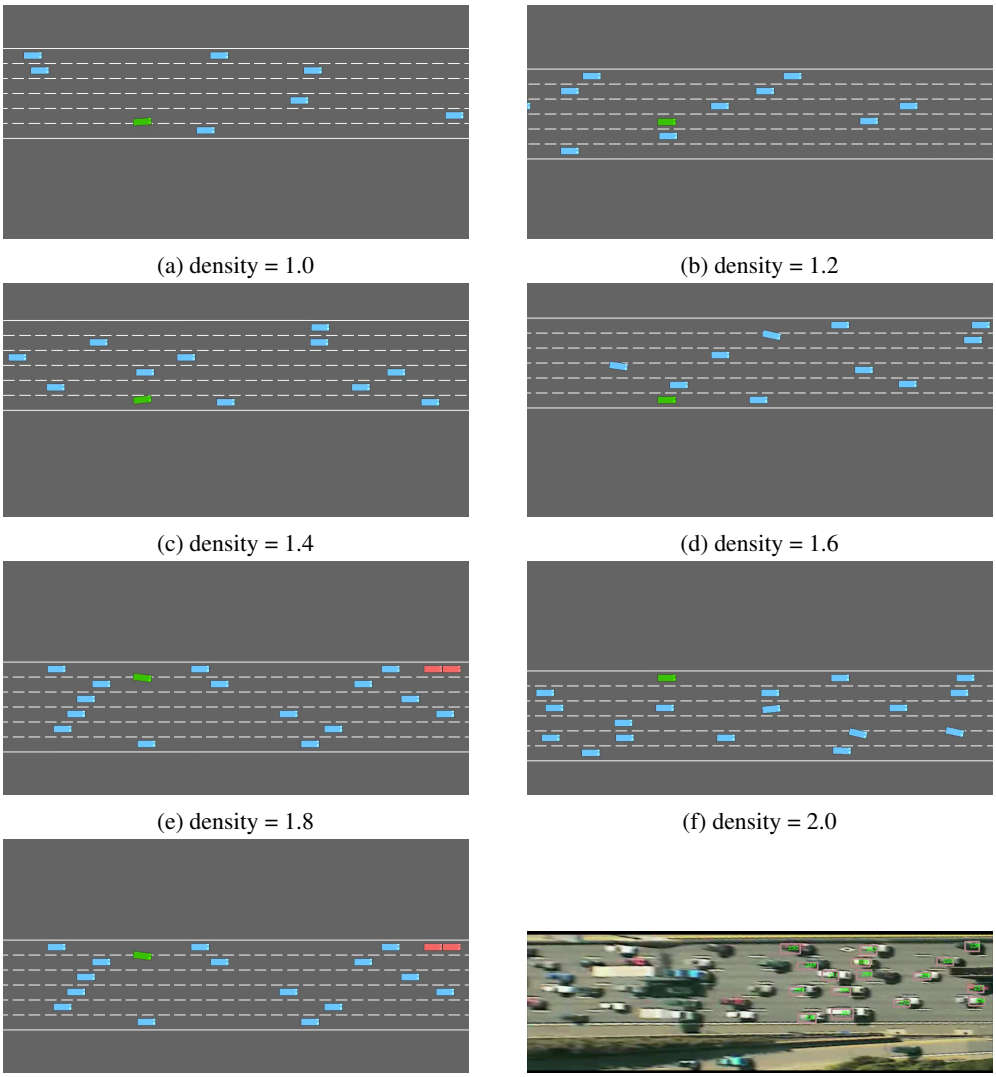
Table 3: Graph Representation for Autonomous Driving

Parameter	Value
Maximum number of links per node	8
Maximum distance of linked nodes	40
WAYPOINT-time horizon	5 seconds
WAYPOINT-sampling frequency	4 per second
WAYPOINT-γ	2.0
TRAJECTORY-time horizon	1 second
TRAJECTORY-γ	1.0

Table 4: Proximal Policy Optimization

Parameter	Value
Number of training steps	$500k$
Number of environments	12
Number of steps per update	512
Buffer size	6144
Number of epochs	10
Batch size	256
Learning rate	1e-3 $\rightarrow$ 5e-4 in 40% steps
Dimensions of policy networks	[256, 256]
Dimensions of value networks	[256, 256]
Discount factor	0.9

B Highway-env



(a) density = 1.0

(b) density = 1.2

(c) density = 1.4

(d) density = 1.6

(e) density = 1.8

(f) density = 2.0

(g) default density = 1.8

(h) realistic density in NGSIM dataset [18]

Figure 6: How the traffic appears under various density levels.