Coordinated Multi-Agent Motion Planning via Imitation Learning

Andrei Ivanovic[1] Rowan McAllister[2] Ashkan Mirzaei[1] Igor Gilitschenski[1]

***Abstract*— As autonomous vehicles (AVs) become increasingly adopted, many opportunities to share information and communicate with one another will arise. This ability to communicate reduces future uncertainties and allows for collaboration in downstream tasks, such as planning, ensuring increased roadside safety. Currently, many motion planners treat other AVs as standalone human agents and plan to avoid their futures without exploring the potential for collaboration and better-informed motion planning. Towards this end, we present a method for coordinated multi-agent motion planning between two or more AVs that search over a distribution of expert future trajectories to jointly plan paths. We evaluate our model on a didactic, illustrative dataset to experimentally verify its performance, with future plans to use more realistic perception data.**

I. INTRODUCTION

Recent years have seen large advancements in autonomous driving and within the last decade autonomous vehicles (AVs) are already being adopted worldwide. AVs operate using a set of software henceforth referred to as an "autonomy stack". This autonomy stack consists of several different submodules that address the main tasks the AV needs to perform to move in the world. The stack is most commonly comprised of four components: perception (perceiving the vehicle's surroundings), prediction (forecasting potential future events), planning (devising motion plans for the vehicle to follow to reach a desired goal), and control (executing those motion plans).

Motion planning is a crucial component of modern autonomous driving systems, creating safe trajectories for a vehicle to follow in an uncertain and ever-changing environment [1]. To generate these motion plans, autonomous vehicles must first reason about the surrounding environment, as well as predict the future motions of other agents present in the scene. To do so, current motion planners treat other vehicles as standalone sentient agents, predicting their futures and planning paths that avoid them. As AVs are increasingly deployed in the real world, however, scenes with multiple AVs present will gradually become commonplace. In these scenarios, each AV will execute the aforementioned autonomy stack, predicting the future trajectories of other AVs and human-driven vehicles and developing motion plans based on these predictions. However, if both AVs could trustfully communicate, the ability share information arises, enabling the synthesis of coordinated motion plans with reduced uncertainty by removing the need to forecast other AVs' futures. Currently, the vast majority of state-of-the-art motion planning algorithms neglect coordination in such scenarios, instead treating each vehicle uniformly without distinguishing human agents from AVs.

[1]Andrei Ivanovic, Ashkan Mirzaei, and Igor Gilitschenski are with the University of Toronto. a.ivanovic@mail.utoronto.ca, {ashkan,gilitschenski}@cs.toronto.edu.

[2]Rowan McAllister is with the Toyota Research Institute. rowan.mcallister@tri.global.

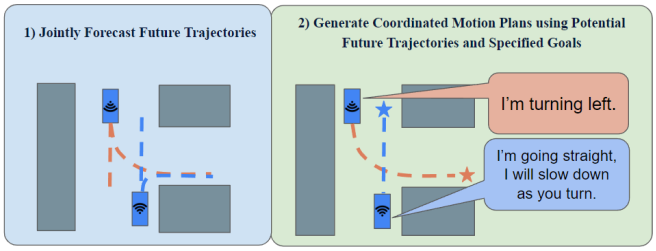

Fig. 1. A visual overview of our method operating in a three-way intersection. Both vehicles are autonomous with their trajectories denoted by dashed lines and desired goals represented as stars.

To this end, this paper presents a context-aware, data-driven, multi-agent coordinated motion planning algorithm that leverages vehicle-to-vehicle (V2V) communication. To achieve coordinated motion planning, our method leverages two key components: (1) a generative autoregressive flow that determines possible future ego-vehicle trajectories by learning to imitate expert driving behavior, and (2) a coordinated motion planner that searches over AVs' joint latent behavior space to generate plans for each AV to achieve their respective goals. An example of our proposed pipeline is visualized in Fig. 1, where two AVs are approaching a three-way intersection in opposing lanes. As the AVs near each other, we assume temporary centralized control of both vehicles. Our trajectory forecaster then predicts the potential future trajectories of the two AVs. Our coordinated motion planner then searches these trajectories to determine a joint motion plan that will deliver each AV to their goal in a safe manner. An example trajectory forecast for two vehicles is shown in Fig. 2. Fig. 3 shows an example of a joint motion plan for two vehicles.

Contributions. The key contribution of this paper is a coordinated multi-agent motion planning algorithm that searches over a distribution of potential future trajectories to jointly plan paths for multiple AVs. This algorithm's performance was experimentally verified on a didactic dataset.

II. RELATED WORK

Multi-Agent Trajectory Forecasting. Data-driven methods have been heavily used to forecast interactions between various types of agents (e.g., vehicles, pedestrians) [2], [3], [4], [5], [6]. Broadly, most approaches apply Imitation Learning (IL) [7] to mimic previously observed expert in-

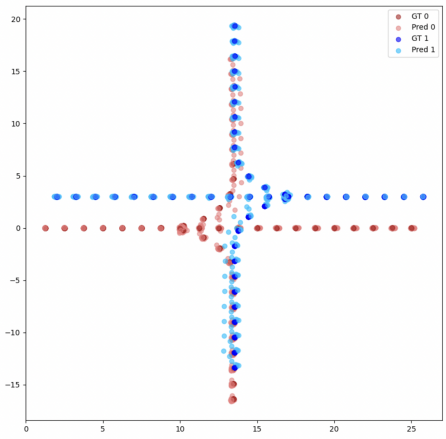

Fig. 2. Our traffic model's prediction results on an illustrative two-car dataset. The two cars can move in one of three directions, and are shown in red (moving rightwards) and blue (moving leftwards), with dark red and dark blue denoting the ground truth futures of each car, respectively.

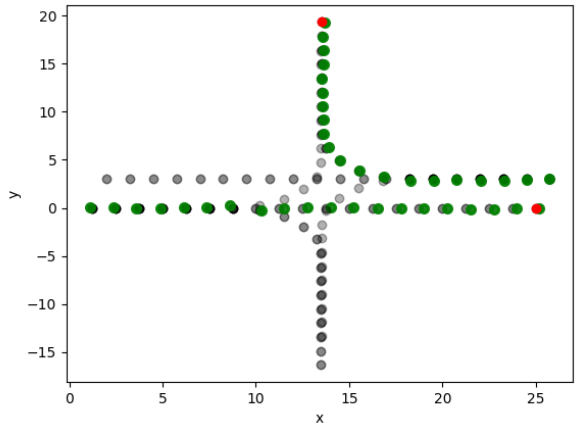

Fig. 3. An example of a joint optimal path (in green) found by our optimizer that delivers both AVs to their goal destination (in red) using expert-like trajectories (in gray).

teractions to generalize and predict multi-agent behaviours in new scenarios. Our method is similar to methods such as Behaviour Cloning (BC) [8] whereby visuomotor policies are learned through expert data. However, we differ from BC through Deep Imitative Models (DIM) [9], a method we extend, and their use of spatial information inherent in the waypoints outputted by an A* planner.

A majority of methods used for human trajectory forecasting fall under two types of approaches: deterministic regression models and generative models. Several works detailed in [10] are deterministic methods used for agent trajectory forecasting, however, forecasting is inherently non-deterministic due to the future's uncertainty and existence of many possible outcomes. Therefore, probabilistic methods are better suited to deal with these uncertainties.

As of late, generative, probabilistic approaches to multi-agent trajectory forecasting and planning have emerged as state-of-the-art [10]. These methods have transitioned to predicting a distribution of possible future trajectories as opposed to only providing the single best. This shift has proven useful for downstream tasks such as planning because of the access to more information that the motion planner can use to make safer decisions, something our method relies heavily upon. Our method is most similar to works using Conditional Variational Autoencoders (CVAE) [11] to encode multimodality using recurrent backbone models [12], [13], [6], [14], [15].

Prediction for Planning and Control. Many of the methods previously discussed are not developed with downstream tasks such as planning and control in mind. Recently, several works have designed their predictions' output representation to be cognizant of downstream motion planning. One such work is the Neural Motion Planner (NMP) [16] which outputs a temporal cost volume to be used in conjunction with a sampling-based motion planner to evaluate trajectory samples online. Another approach is MATS [17], whose predictions consider agent-agent interactions as well as the ego-vehicle's future.

Given different output representations, trajectory forecasting can be incorporated in planning in several different ways. Some works optimize a plan in position space so as to avoid other agents' forecasted futures [18], [19] while others project their predictions into the ego-vehicle's control space and subsequently optimize its controls (e.g., [17]). However, our method which extends DIM, differs significantly because our planning is done *through* our predictor by searching its learned joint latent space during optimization, rather than simply using its trajectory forecasts.

Methods Extended. Our method is most related to R2P2 [20], DIM [9], and PRECOG [13]. R2P2 proposes a method that forecasts ego-vehicle trajectories as a probability distribution over expert-like paths, conditioned on visual features. In addition to a novel objective function, R2P2 also proposes a trajectory forecaster which learns a bijection that transforms a Gaussian "base" distribution to a distribution of expert-like trajectories such that when a sample from the Gaussian is "pushed forward" through the bijection, it yields an expert path sample. We illustrate R2P2's forecasting factor graph in the left of Fig. 4. Using scene observations and the ego-vehicle's past states ϕ, R2P2 forecasts the vehicle's future states. For R2P2, the AV's intent Z^r is unknown and is represented with a circular node.

DIM [9] builds on R2P2's trajectory forecasting model by combining the advantages of imitation learning and goal-based planning. DIM devises future ego-motion plans to desired goal locations by searching over a restricted space of "human-like" future trajectories. Using R2P2's trajectory prediction model, DIM is able to search over the latent base distribution to find an expert-like trajectory that best achieves the desired task. The paper also presents numerous different goal formulations. As can be seen in Fig. 4, DIM expands upon R2P2 with the ability to plan for and execute ego-vehicle motion plans. Accordingly, Z^r is now referred to as z^r and represented with a square, shaded node, where shading denotes that the variable is observed.

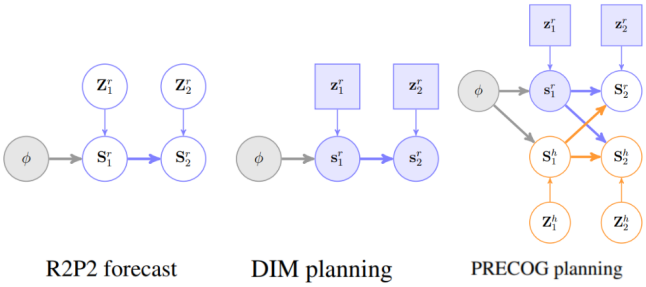

Fig. 4. Factor graphs for R2P2 [20], DIM [9], and PRECOG [13]. As shown, the forecasts and plans are conditioned on the scene context, ϕ (explained further in Section III). Z denotes a driver's intent and S denotes the future system ("h" superscript denotes human and "r" superscript denotes robot). Shaded nodes represent variables that are observed and square nodes denote a decision made by an AV. The subscripts denote different timesteps.

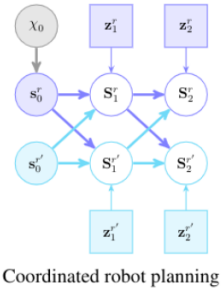

Fig. 5. A factor graph showcasing this paper's main differences from the three prior works in Fig. 4. Specifically, our method extends PRECOG [13] by (1) incorporating knowledge of other agents' intent, and (2) optimizing for a joint motion plan using additional latent variables.

Lastly, PRECOG [13] expands upon the prior two works by proposing a prediction model able to forecast future interactions between a variable number of agents present in a scene. Specifically, the model generates conditional forecasts in which the model reasons how agents will react to knowledge of the ego-vehicle's goal. As an example, if our ego-vehicle plans to stop at an upcoming intersection we can assume with a higher probability that any vehicle tailing the ego-vehicle will subsequently stop. Shown on the right of Fig. 4, we can see that PRECOG extends DIM by incorporating interactions with other agents, treating the state of the robot (denoted with an "r" superscript) and state of the human (denoted with an "h" superscript) as a joint state. Notably, if the other agent was a robot, the factor graph would remain as-is because the other robot's intent is assumed unknown.

Our method extends [13] by increasing the dimensionality of the latent space N-fold in order to jointly model all N AVs present in a scene and their interactions, as opposed to only considering the ego-vehicle. Furthermore, the AVs are easily controlled and delivered to their goals as a result of our optimization process.

III. PROBLEM FORMULATION

Nomenclature. Throughout the rest of this paper, a "plan" denotes a sequence of states an AV can follow to reach its goal. A "goal" is taken to mean an AV's desired future state. The term "forecasting" refers to the inference of the potential future states of various agents present in a scene [13]. Finally, "scene context" refers to visual features or observations such as LIDAR or past vehicle states.

We aim to generate a set of non-colliding motion plans $M^{1:N} \in \mathbb{R}^{N\times T\times D}$ that deliver N autonomous agents $A^{1:N}$ to their respective goals $G^{1:N}$ within T timesteps. D refers to the dimension of the trajectory data, which in our case is two. An agent's goal $G^i \in \mathbb{R}^D$ is its desired final state. Since we are primarily concerned with AVs in this work, each agent A^i is a vehicle and its D-dimensional state $\mathbf{S}_t^i \in \mathbb{R}^D$ consists of the AV's x, y coordinates at time t. We additionally assume that each agent receives high-dimensional observations of the scene χ (e.g., LIDAR or RGB) and its τ previous states, $\mathbf{S}^i_{-\tau:0} \in \mathbb{R}^{\tau\times D}$. Altogether, these components comprise an agent's overall scene context, ϕ. Finally, each AV also possesses the ability to communicate and share information with the other AVs present in the scene, allowing for coordinated motion planning.

IV. MULTI-AGENT COORDINATED PLANNING

Our method assumes temporary, centralized control over two or more AVs to execute a coordinated motion plan. At a high-level, our approach is a two-step process. First, a flow-based traffic model encodes the prior history of agents and any scene context to produce an informative joint latent space that captures potential future human-like behaviors for all AVs. The resulting joint latent space is then searched by our motion planner to determine safe, human-like motion plans for all AVs conditioned on their respective goals. Our work's contribution is best illustrated in Fig. 5, extending PRECOG by introducing additional latent variables that enable optimizing the joint plan. We can also see that now the other agent's intent is known due to communication between AVs.

A. Flow-Based Generative Trajectory Forecasting

The first component of our approach is a trajectory prediction model that encodes a particular scene χ and its history $\mathbf{S}^i_{-\tau:0}$ into a joint latent space $\mathbf{Z}$. Since this joint latent space will be used later in planning, each latent vector should directly correspond to an output trajectory. Accordingly, we apply normalizing flows to learn a mapping between latent vectors and future trajectories. A normalizing flow [21] is a method by which a complex distribution (e.g., over future trajectories) is synthesized by transforming a simple probability density (e.g., a Gaussian) through a series of invertible, bijective functions. Intuitively, the simple density "flows" through the invertible mappings to achieve a (normalized) general probability distribution. Formally,

$$\mathbf{Z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}); \qquad \mathbf{E} = g_\pi(\mathbf{Z}; \phi), \tag{1}$$

where $\phi = \{\chi, \mathbf{S}^i_{-\tau:0}\}$, $\mathbf{E}$ is a distribution of expert-like trajectories, and g_π is the bijection transforming the joint latent space $\mathbf{Z}$ to a complex distribution $\mathbf{E}$. The function g_π

is learned as a result of training our flow-based prediction model.

In this work, we extend the flow-based Deep Imitative Model (DIM) architecture [20], [9] for our traffic model. While the original method only modeled a single agent, we increase its latent dimensionality N-fold in order to jointly model all AVs and their interactions.

DIM uses a differentiable and invertible simulator g_π as the learnable bijection between two distributions: (1) a Gaussian joint latent base distribution $\mathbf{Z}$, and (2) a distribution of path trajectories $\mathbf{E}$, which are expert trajectories in our case. The simulator g_π is then used to "push forward" samples from distribution (1) to elements of distribution (2). The simulator g_π is an autoregressive map representing a discrete, stochastic dynamical system as follows:

$$\mathbf{E}_t = g_\pi(\mathbf{Z_{1:t}})\big|_t = \mu_\pi(\mathbf{E}_{1:t-1}, \phi) + \sigma_\pi(\mathbf{E}_{1:t-1}, \phi)\mathbf{Z}_t, \quad (2)$$

where $\mu_\pi(\mathbf{E}_{1:t-1}, \phi)$ represents a "Verlet" step, as in [20], and is defined as:

$$\mu_\pi(\mathbf{E}_{1:t-1}, \phi) = 2\mathbf{E}_{t-1} - \mathbf{E}_{t-2} + m_\pi(\mathbf{E}_{1:t-1}, \phi). \quad (3)$$

The two quantities m_π and σ_π are learned as a result of training our traffic model, enabling it to accurately generate multimodal distributions resembling expert trajectories.

We additionally use the MobileNetV2 [22] Convolutional Neural Network (CNN) architecture to encode the feature maps χ, which condition our flow-based predictor. The past states of each AV, $\mathbf{S}^a_{-\tau:0}$, are encoded using a Gated Recurrent Unit (GRU) [23].

To train the model, for each data point (ϕ, y) we maximize the log-likelihood of the ground truth future y given the scene context ϕ. Formally,

$$\max_\pi \ \log q_0\left(g_\pi^{-1}(y;\phi)\right) - \log\left|\det J_{g_\pi}\left(g_\pi^{-1}(y;\phi)\right)\right|, \quad (4)$$

where q_0 is the Gaussian base distribution, $g_\pi^{-1}(y;\phi)$ is the inverse flow from the path distribution to the base distribution (evaluated at y), and $J_{g_\pi}\left(g_\pi^{-1}(y;\phi)\right)$ is the Jacobian of g_π (evaluated at $g_\pi^{-1}(y;\phi)$). The full model architecture is shown in Fig. 6 with additional implementation details in Appendix A.

B. Coordinated Multi-Agent Motion Planning

With a trajectory forecasting model in hand, our coordinated planner can search for human-like trajectories that guide each AV to its desired goal. To do so, the planner minimizes a non-convex cost function with decision variable z by gradient descent. Formally, it aims to solve

$$\min_z \frac{1}{N}\sum_{i=1}^{N} \|g_\pi(z)^i_T - y^i_T\|_2^2, \quad (5)$$

where y^i_T denotes the final state of the ground truth expert trajectory and z is a latent vector in the joint latent base distribution, that, when pushed forward, results in expert-like trajectories for each AV. An optimal z^* results in each AV reaching their goals without any unsafe interactions between each other.

At the start of the optimization process, to avoid local minima, the latent vector z is chosen by sampling different initial base distribution vectors to determine which results in the lowest final displacement error between the goal and the last state of each of the vectors when pushed forward to the expert trajectory distribution. Section V-E will further discuss the necessity of this initialization scheme. The conditioning base vector is then optimized using gradient descent to converge to a satisfactory z, yielding expert trajectories that deliver each AV to its goal.

V. EXPERIMENTS

A. Simple Illustrative Dataset

We evaluate our method's performance on a simple, didactic dataset. The dataset is trimodal, representing an intersection where a vehicle can either turn left, right, or continue straight. Two autonomous cars are placed in the scene, facing each other in two different lanes. Each car had the option to turn right or left into single-lane streets, or continue straight on their respective lanes. Expert, ground-truth trajectories were generated for each of the three modes with constant velocities. Scene context ϕ consisted of each car's states for the previous $\tau = 5$ time steps as well as a semantic birds-eye-view observation $\chi \in \mathbb{R}^{100\times100\times3}$ of the intersection, where roads are represented as white rectangles, and the surrounding ground is green. This semantic observation is shown in the background of Fig. 7. The single-car dataset consisted of three cases: the car turns right, left, or continues straight. Each of these cases were duplicated 10000 times and combined to create training, validation, and test datasets of size 30000 each. For the two-car dataset, five cases were made for each valid combination of trajectories that each of the two AVs could take: (1) straight and straight, (2) straight and right, (3) left and left, (4) right and straight, and (5) right and right. A training, validation, and test dataset were made by replicating each case 6000 times and combining them to have datasets with 30000 examples total. Noise was applied to each of the trajectories to introduce variance into the dataset. Goal points were selected to be the final point of each vehicles' respective ground truth trajectories.

B. Prediction Evaluation

The trajectory prediction model was trained using this trimodal dataset for 250 epochs and evaluated using a negative log-likelihood (NLL) loss. Our loss values were compared against a theoretical lower-bound of -120 for the single-vehicle case and -240 for the two-vehicle case. After training, our model achieved an NLL of -116 for the single-car case and an NLL of -231 for the two-car case, as expected. Trajectory forecasts were also visually verified for both the single-car and two-car case. An example of the two-car results is shown on Fig. 2.

C. Contextual Awareness Evaluation

We evaluated our model's ability to appropriately incorporate scene context by adding different intersection types into

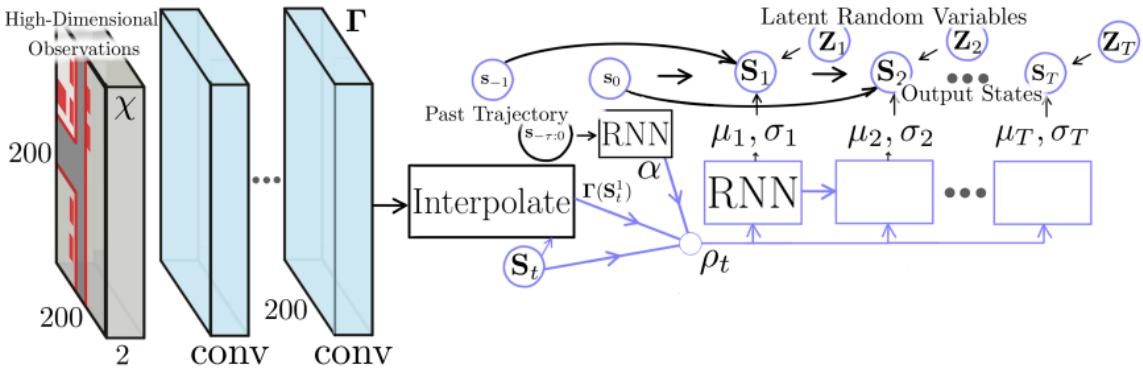

Fig. 6. Our method's architecture diagram (building upon that of DIM [9], reproduced with author's permission) of m_π and σ_π which are used to learn expert agent dynamics. The inputs to this model are scene context, $\phi = \{\chi, \mathbf{S}^i_{-\tau:0}\}$, and latent noise $\mathbf{Z}^{1:T}$. The outputs of the model are trajectories, $\mathbf{S}^{1:T}$. Most importantly, the traffic model can be used to search for expert-like trajectories that reach all AVs' respective goals.

our didactic dataset. Two more road geometries, a left-turn-only and right-turn-only road, were added in addition to the trimodal intersection. In this evaluation, we are looking for trajectory forecasts that account for each input intersection type (e.g., only turning left on the left-turn intersection). As can be seen in Fig. 7, our model is indeed able to factor in scene context and produces predictions appropriate for the input scenario. We additionally evaluated our model's capability to match dataset statistics by varying the amount of different trajectory types in the dataset. In particular, for the trimodal intersection, the car would go straight half of the time and turn left or right a quarter of the time. These same proportions between different future possibilities was maintained in our trajectory forecasts, displaying a similar $50\%/25\%/25\%$ direction split.

D. Motion Planning Evaluation

Using the prior model trained on the trimodal dataset, our motion planner was experimentally verified by selecting one of the single-car and two-car modes and placing goals at the end of each car's trajectory. Gradient descent was performed for 1000 iterations using the mean squared error between our path's final state and the goal state as our loss function. Fig. 8 visualizes this optimization process. For each case, the final, optimal path was visually shown to reach the desired goal by using an expert-like trajectory. An example of an optimal path reaching each car's goal for the two-car example is shown in Fig. 3. Additional results are shown in Appendix B.

E. Planning - Forecasting Accuracy Trade-Off

During motion planning evaluation, there was an interesting pattern that emerged. When using a prior model that was better trained on the trimodal dataset, the three modes were more "rigidly" separated in the joint latent base distribution distribution. As a result, when optimizng the MSELoss dropped significantly at one point, but has difficulties searching the different modes until then. A visual comparison of the optimization process between a better trained prior and less trained prior is shown in Fig. 8. Therefore, it seems that a better-trained prior is better for open-loop prediction but worse for optimization and a worse-trained prior is worse for open-loop prediction but better for optimization. To remedy this, a better prior was used without this consequence by initially sampling different conditioning base vectors, z, and using the one with the lowest MSELoss, allowing the optimizer to start its search "closer" to the optimal path.

VI. CONCLUSION

In this work, we present a method for coordinated multi-agent planning with promising performance on a didactic test dataset. This method has the potential to be deployed alongside the increasing adoption of AVs to leverage their abilities to communicate and share information, creating safer roadside interactions.

Future Work. Firstly, we will upgrade from using our didactic trimodal dataset to more realistic perception data. Additionally, there is also discussion as to expanding the nature of how perception data is shared between the vehicles. Currently, the ability to share visual observations is not leveraged and methods like V2VNet [24] which create better world representations by having the different AVs fill in each others' gaps can be incorporated or built on.

REFERENCES

[1] B. Paden, M. Čáp, S. Z. Yong, D. Yershov, and E. Frazzoli, "A survey of motion planning and control techniques for self-driving urban vehicles," *IEEE Transactions on Intelligent Vehicles*, vol. 1, no. 1, pp. 33–55, 2016.

[2] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese, "Social LSTM: Human trajectory prediction in crowded spaces," in *IEEE Conf. on Computer Vision and Pattern Recognition*, 2016.

[3] A. Gupta, J. Johnson, F. Li, S. Savarese, and A. Alahi, "Social GAN: Socially acceptable trajectories with generative adversarial networks," in *IEEE Conf. on Computer Vision and Pattern Recognition*, 2018.

[4] W.-C. Ma, D.-A. Huang, N. Lee, and K. M. Kitani, "Forecasting interactive dynamics of pedestrians with fictitious play," in *IEEE Conf. on Computer Vision and Pattern Recognition*, 2017.

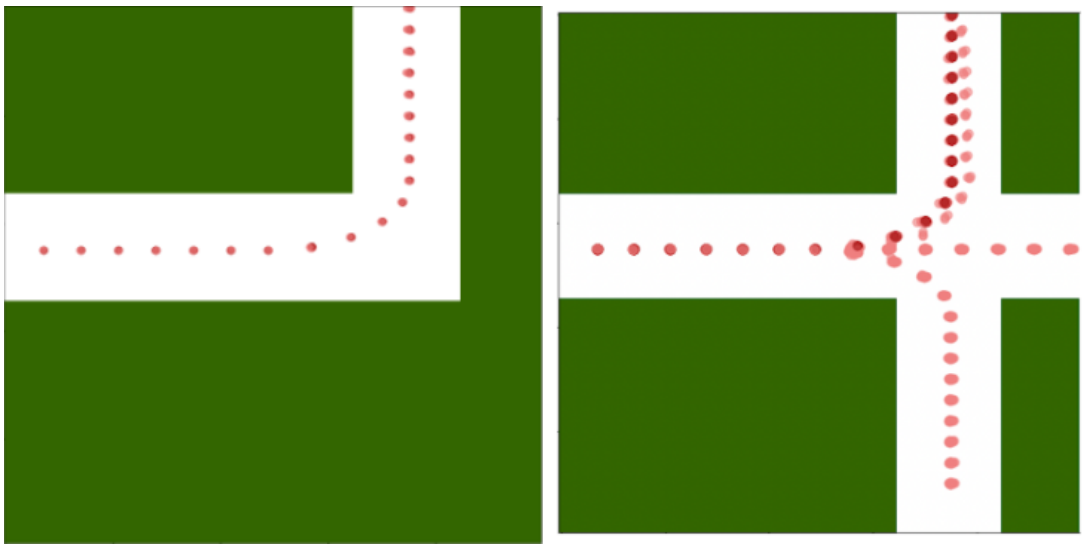

Fig. 7. A side-by-side comparison of two different intersection types present in our didactic dataset, and our model's ability to produce accurate trajectory forecasts that are conditioned on its visual observations.

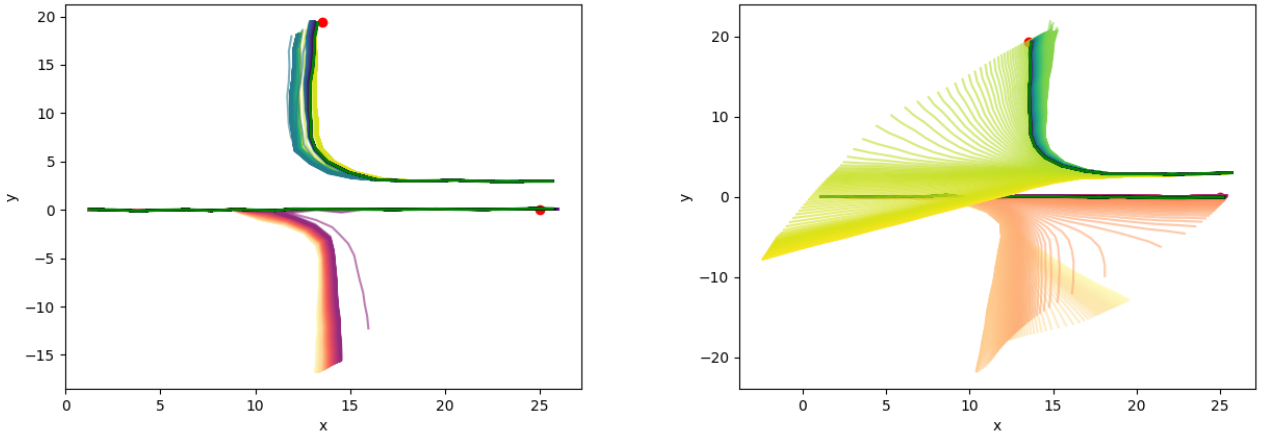

Fig. 8. A side-by-side comparison of the optimization process between a better trained prior (trained for 238 epochs, left) and a worse trained prior (trained for 10 epochs, right). Lighter paths are ones generated early in the optimization process for each car, and darker colours represent later ones. The goal points for each AV are shown in red, with a final optimal path in green. As can be seen, the left process has difficulties searching different modes and "snaps" to an answer, one that is worse than the right final optimal path.

[5] N. Deo and M. M. Trivedi, "Multi-modal trajectory prediction of surrounding vehicles with maneuver based lstms," in *IEEE Intelligent Vehicles Symposium*, 2018.

[6] N. Lee, W. Choi, P. Vernaza, C. B. Choy, P. H. S. Torr, and M. Chandraker, "DESIRE: distant future prediction in dynamic scenes with interacting agents," in *IEEE Conf. on Computer Vision and Pattern Recognition*, 2017.

[7] T. Osa, J. Pajarinen, G. Neumann, J. A. Bagnell, P. Abbeel, and J. Peters, "An algorithmic perspective on imitation learning," *Foundations and Trends in Robotics*, vol. 7, no. 1-2, pp. 1–179, 2018.

[8] D. A. Pomerleau, "Alvinn: An autonomous land vehicle in a neural network," in *Conf. on Neural Information Processing Systems*, 1989.

[9] N. Rhinehart, R. McAllister, and S. Levine, "Deep imitative models for flexible inference, planning, and control," in *Int. Conf. on Learning Representations*, 2020.

[10] A. Rudenko, L. Palmieri, M. Herman, K. M. Kitani, D. M. Gavrila, and K. O. Arras, "Human motion trajectory prediction: A survey," *Int. Journal of Robotics Research*, vol. 39, no. 8, pp. 895–935, 2020.

[11] K. Sohn, H. Lee, and X. Yan, "Learning structured output representation using deep conditional generative models," in *Conf. on Neural Information Processing Systems*, 2015.

[12] B. Ivanovic, E. Schmerling, K. Leung, and M. Pavone, "Generative modeling of multimodal multi-human behavior," in *IEEE/RSJ Int. Conf. on Intelligent Robots & Systems*, 2018.

[13] N. Rhinehart, R. McAllister, K. Kitani, and S. Levine, "PRECOG: Prediction conditioned on goals in visual multi-agent settings," in *IEEE Int. Conf. on Computer Vision*, 2019.

[14] E. Schmerling, K. Leung, W. Vollprecht, and M. Pavone, "Multimodal probabilistic model-based planning for human-robot interaction," in *IEEE/RSJ Int. Conf. on Intelligent Robots & Systems*, 2018.

[15] T. Salzmann, B. Ivanovic, P. Chakravarty, and M. Pavone, "Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data," in *European Conf. on Computer Vision*, 2020.

[16] W. Zeng, W. Luo, S. Suo, A. Sadat, B. Yang, S. Casas, and R. Urtasun, "End-to-end interpretable neural motion planner," in *IEEE Conf. on Computer Vision and Pattern Recognition*, 2019.

[17] B. Ivanovic, A. Elhafsi, G. Rosman, A. Gaidon, and M. Pavone, "MATS: An interpretable trajectory forecasting representation for planning and control," in *Conf. on Robot Learning*, 2020.

[18] J. Ziegler, P. Bender, T. Dang, and C. Stiller, "Trajectory planning

for bertha - a local, continuous method," in *IEEE Intelligent Vehicles Symposium*, 2014.

[19] C. Liu, S. Lee, S. Varnhagen, and H. E. Tseng, "Path planning for autonomous vehicles using model predictive control," in *IEEE Intelligent Vehicles Symposium*, 2017.

[20] N. Rhinehart, K. M. Kitani, and P. Vernaza, "R2P2: A reparameterized pushforward policy for diverse, precise generative path forecasting," in *European Conf. on Computer Vision*, 2018.

[21] D. J. Rezende and S. Mohamed, "Variational inference with normalizing flows," in *Int. Conf. on Machine Learning*, 2015.

[22] T. Salzmann, B. Ivanovic, P. Chakravarty, and M. Pavone, "MobileNetV2: Inverted residuals and linear bottlenecks," in *IEEE Conf. on Computer Vision and Pattern Recognition*, 2018.

[23] K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," in *Proc. of Conf. on Empirical Methods in Natural Language Processing*, 2014, pp. 1724–1734.

[24] T.-H. Wang, S. Manivasagam, M. Liang, B. Yang, W. Zeng, J. Tu, and R. Urtasun, "V2VNet: Vehicle-to-vehicle communication for joint perception and prediction," in *European Conf. on Computer Vision*, 2020.

APPENDIX A - ARCHITECTURAL DETAILS AND TRAINING HYPERPARAMETERS

The architecture of our trajectory forecaster (in Fig. 6) is shown in Fig. 9. Extends the architecture of [9] by increasing the dimensionality N-fold.

Component	Input [dimensionality]	Layer or Operation	Output [dimensionality]	Details
Static featurization of context: $\phi = \{\chi, \mathbf{s}_{-\tau:0}^{1:A}\}$.				
MapFeat	$\chi\ [H, W, 2]$	2D Convolution	${}^{1}\chi\ [H, W, 32]$	3×3 stride 1, ReLu
MapFeat	${}^{i-1}\chi\ [H, W, 32]$	2D Convolution	${}^{i}\chi\ [H, W, 32]$	3×3 stride 1, ReLu, $i \in [2, \ldots, 8]$
MapFeat	${}^{8}\chi\ [H, W, 32]$	2D Convolution	$\mathbf{\Gamma}\ [H, W, 8]$	3×3 stride 1, ReLu
PastRNN	$\mathbf{s}_{-\tau:0}\ [\tau + 1, D]$	RNN	$\alpha\ [32]$	GRU across time dimension
Dynamic generation via loop: for $t \in \{0, \ldots, T-1\}$.				
MapFeat	$\mathbf{s}_t\ [D]$	Interpolate	$\gamma_t = \mathbf{\Gamma}(\mathbf{s}_t)\ [8]$	Differentiable interpolation
JointFeat	$\gamma_t, \mathbf{s}_0, {}^{2}\eta, \alpha, \boldsymbol{\lambda}$	$\gamma_t \oplus \mathbf{s}_0 \oplus {}^{2}\eta \oplus \alpha \oplus \boldsymbol{\lambda}$	$\rho_t\ [D + 50 + 32 + 1]$	Concatenate ($\oplus$)
FutureRNN	$\rho_t\ [D + 50 + 32 + 1]$	RNN	${}^{1}\rho_t\ [50]$	GRU
FutureMLP	${}^{1}\rho_t\ [50]$	Affine (FC)	${}^{2}\rho_t\ [200]$	Tanh activation
FutureMLP	${}^{2}\rho_t [200]$	Affine (FC)	$m_t\ [D],\ \xi_t\ [D, D]$	Identity activation
MatrixExp	$\xi_t\ [D, D]$	$\mathrm{expm}(\xi_t + \xi_t^{a,\mathrm{transpose}})$	$\boldsymbol{\sigma}_t\ [D, D]$	Differentiable Matrix Exponential Rhinehart et al. (2018)
VerletStep	$\mathbf{s}_t, \mathbf{s}_{t-1}, m_t, \boldsymbol{\sigma}_t, \mathbf{z}_t$	$2\mathbf{s}_t - \mathbf{s}_{t-1} + m_t + \boldsymbol{\sigma}_t \mathbf{z}_t$	$\mathbf{s}_{t+1}\ [D]$	

Fig. 9. Implementation details (reproduced with author's permission [9]) for the trajectory forecaster's two components: scene context featurizer and trajectory generator. For our didactic dataset, $H = W = 100$.

Relevant Training Hyperparameters								
Optimizer	Learning Rate	Weight Decay	Epochs	Batch Size	Clip Gradients	Number of Cars	State Dimension	Output Shape
AdamW	1.00E-03	0	250	64	True (to 1.0)	1 or 2	2	(20, 2)

Fig. 10. Relevant training hyperparameters.

APPENDIX B - ADDITIONAL PLANNING RESULTS

The results of our coordinated motion planner on the left-left and right-right case are shown in Fig. 11 and the results from the straight-straight and right-straight case are shown in Fig. 12. For each of the pair of figures, the left one depicts the joint optimization process for both vehicles across 1000 steps. The optimal path at the end of the process is in green, delivering each to their respective goals in red. Each optimal path is overlaid onto our trimodal dataset in the right image of each figure. This demonstrates that each path follows expert-like trajectories to reach each vehicles' goal.

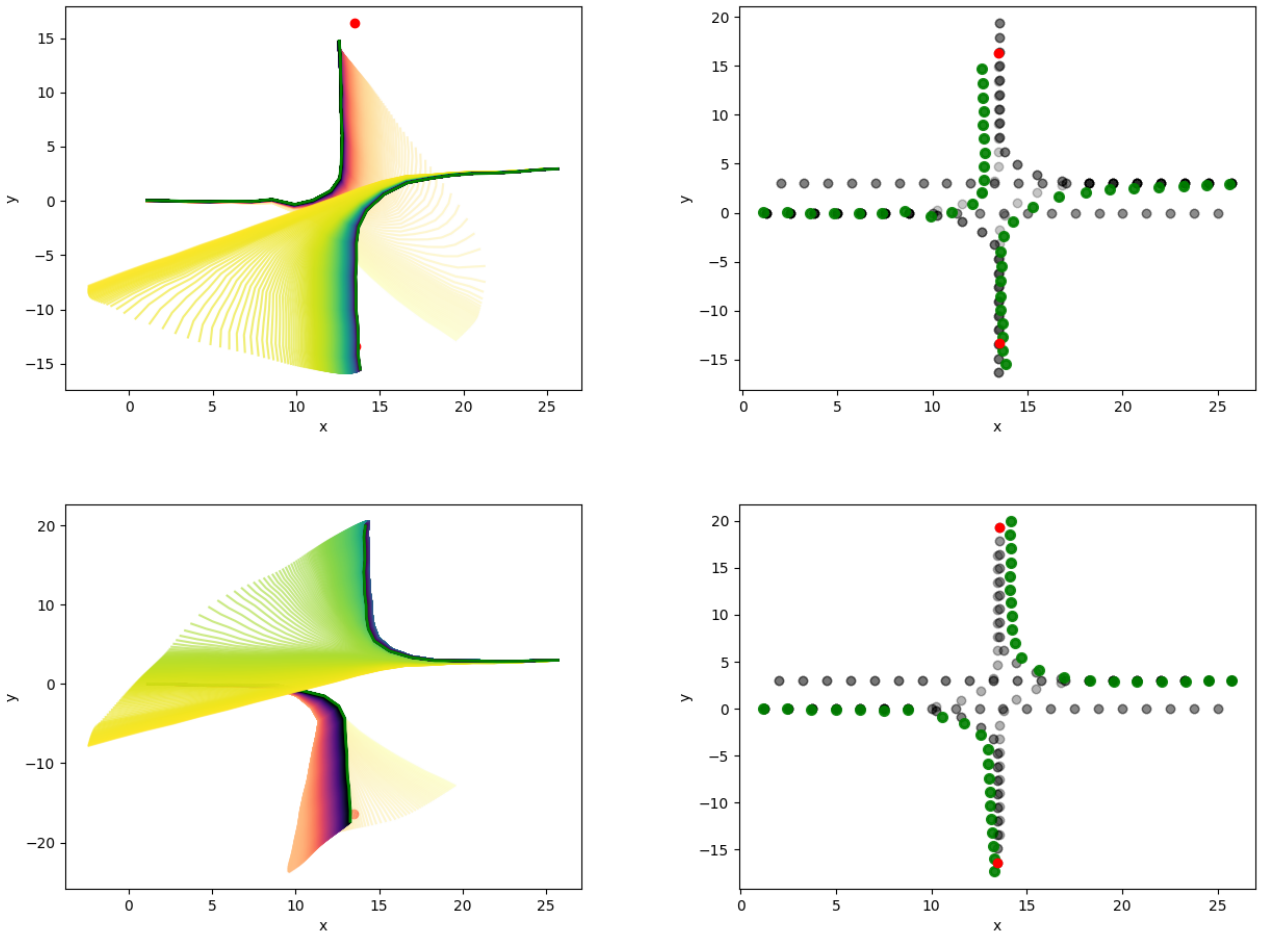

Fig. 11. Additional coordinated motion planning results with the optimization process shown for each case on the left and optimal path (green) overlaid onto our trimodal dataset on the right. The red points denote each vehicle's goal.

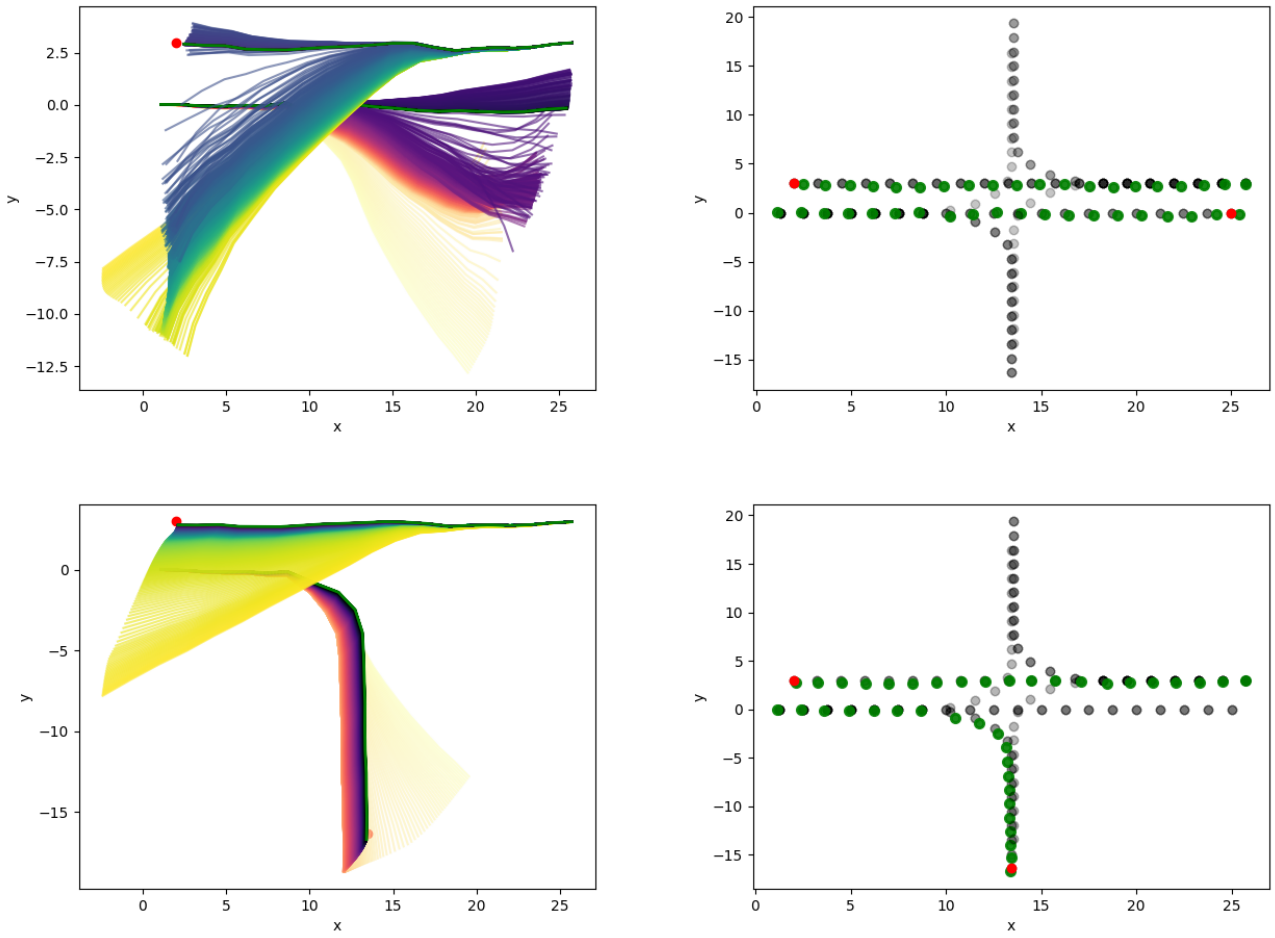

Fig. 12. Additional coordinated motion planning results with the optimization process shown for each case on the left and optimal path (green) overlaid onto our trimodal dataset on the right. The red points denote each vehicle's goal.