Energy-based Potential Games for Joint Motion Forecasting and Control

Anonymous Author(s)
Affiliation
Address
email

Abstract: Game theory offers an interpretable mathematical framework for modeling multi-agent interactions. However, its applicability in real-world robotics applications is hindered by several challenges, such as unknown agents' game parameters. To address these challenges, we establish a connection between differential games, optimal control, and energy-based models and demonstrate how existing approaches can be unified under our proposed *Energy-based Potential Game* formulation. Building upon this, we introduce a new end-to-end learning application that combines neural networks for game-parameter inference with a differentiable game-theoretic optimization layer, acting as an inductive bias. The analysis provides empirical evidence that the game-theoretic layer provides a layer of interpretability and improves the predictive performance of various neural network backbones in two simulations, and based on a real-world public dataset.

1 Introduction

Modeling multi-agent interactions is essential for many robotics applications like motion forecasting and control. For instance, a mobile robot or a self-driving vehicle (SDV) has to interact with other pedestrians or human-driven vehicles to navigate safely toward its goal locations. Although data-driven approaches have made significant progress in multi-agent forecasting [1], challenges arise due to the additional verification requirements in safety-critical domains. Hence, [2] postulates the following critical objectives among others: (i) Integrating well-established principles like prior knowledge about multi-agent interactions to facilitate effective generalization, (ii) Ensuring interpretability of latent variables in models, enabling verification beyond mere testing of the final output.

Game-theoretic approaches [3] incorporate priors based on physics and rationality, such as system dynamics and agent preferences, into interaction modeling. Here, non-cooperative game-theoretic equilibria describe interactions, and solvers typically search for local equilibria [4, 5], resulting in a single (uni-modal) joint strategy $\mathbf{u}$). While finding a cost parametrization for the robot (e.g., an SDV in the open world) is non-trivial [6, 7], knowing the preferences and goals of all other agents is an unrealistic assumption. For example, the intents of human drivers are not directly observable. That makes online inference of game parameters necessary [8]. On the other hand, neural network-based approaches achieve state-of-the-art (SOTA) performance on motion forecasting benchmarks. Works like [11] and [12] employ *explicit strategies* by utilizing feed-forward neural networks with parameters θ to generate multi-modal joint strategies $\mathbf{U} = F_\theta(\mathbf{o})$ based on observation $\mathbf{o}$. Although these models achieve impressive results, they are considered low interpretable black-box models with limited controllability [13]. *How can we leverage the benefits of both groups of approaches?*

Energy-based neural networks [14], a type of generative models, provide an *implicit* mapping $\mathbf{u}^* = \arg\min_{\mathbf{u}} E_\theta(\mathbf{u}, \mathbf{o})$ and [15] demonstrates the advantageous properties of such implicit models in single-agent control experiments. This work shows how to parameterize the energy $E_\theta(\cdot)$ with a potential game formulation [16]. Hence, we combine *explicit strategies* for strategy initialization and game parameter inference with *implicit strategies* as shown in Fig. 1 (c).

Submitted to the 7th Conference on Robot Learning (CoRL 2023). Do not distribute.

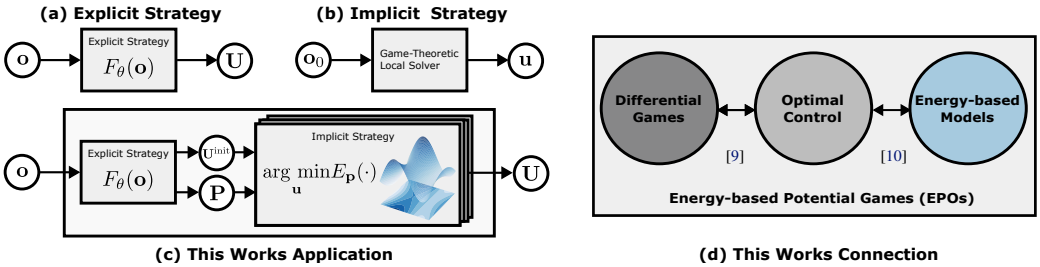

Figure 1: Strategy representations and the proposed connection. (a) Learned explicit strategy producing a multi-modal solution $\mathbf{U}$ based on observations $\mathbf{o}$. (b) Implicit game-theoretic strategy, which only considers the current observation $\mathbf{o}_0$, has fixed game parameters, and produces a local (uni-modal) solution $\mathbf{u}$. (c) We infer multi-modal strategy initializations $\mathbf{U}^{\text{init}}$ and game parameters $\mathbf{P} = \{\mathbf{p}^1, \dots, \mathbf{p}^M\}$ with an explicit strategy, then perform M game-theoretic energy minimizations in *parallel* in a learnable end-to-end framework. (d) EPOs as a connection of different research areas.

Contribution. This work proposes *Energy-based Potential Games* (EPOs) as a class of methods connecting differential games, optimal control, and energy-based models (EBMs) (Fig. 1 (d)). We further show how existing approaches can be unified under this framework. Further, this work proposes a differentiable *Energy-based Potential Game Layer* (EPOL), which is combined with hierarchical neural networks in a novel system architecture. Third, we demonstrate that this approach improves different neural networks in simulated and real-world motion forecasting experiments and is applicable for motion control of a single-agent.

2 Related Work

Game-Theoretic Planning. Game-theoretic motion planning methods [17, 4, 5], aiming to find Nash equilibria (NE), capture the interdependence about how one agent's action influences other agents' futures. However, these approaches typically involve computationally intensive coupled optimal control problems. Hence, [2] and [18] formulate the problem as a potential game [16], reducing the formulation to only a single optimal control problem (OCP). [19] and [8] use filtering techniques and inverse game solvers to learn parameters. However, these methods have been primarily evaluated in simulation, and the learning objectives of [8] assumed uni-modal distributions. In contrast, our work utilizes neural networks to infer multi-modal parameters and initializations and learns end-to-end.

Data-driven Motion Forecasting. Motion forecasting with neural networks currently represents the SOTA in benchmarks for human vehicle [20] or pedestrian prediction [21]. Most works focus on modeling interactions in the observation encoding part. For instance, the attention [12, 22, 23] or convolutional social pooling [24] mechanism is commonly employed. [25] and [26] use latent variable models to generate multi-modal joint predictions. In general, our approach is complementary to these developments, as it can be integrated on top of various network architectures (Sec. 5.1).

Energy-based Models. [14] and [27] provide reviews for EBMs. [28] identifies three main paradigms for energy learning: (i) Conditional Density Estimation (CDE), (ii) Exact Energy Minimization (iii) Unrolled Optimization (UN). Models of the first group (i) use the probabilistic interpretation that low-energy regions have high probability. Approaches utilize maximum likelihood estimation (MLE) [27], noise contrastive (NC) divergence [29], or NC estimation [30] learning objectives. Type (ii) methods solve the energy optimization problem exactly and differentiate using techniques such as the *implicit function theorem* (IFT) [31]. Methods from type (iii), like [28], approximate the solution with a finite number of gradient steps and backpropagate through the unrolled optimization.

Differentiable Optimization. Advances in differentiable optimization [31, 32] enable our work, to combine optimization problems with learning-based models such as neural networks. [33] introduces a differentiable architecture using a non-game-theoretic single-agent motion planner for SDVs. The concurrent work of [5] proposes a combination of a differentiable joint planner with a simple network architecture (multilayer perceptron (MLP)) and serves as a proof of concept. Further, [5] evaluates open-loop on a simulated dataset and does not account for multi-modality. In contrast, we produce multi-modal predictions, evaluate on a real-world dataset, and use different SOTA neural networks.

Discussion of Related Applications. Tab. 1 shows how existing applications from the field of multi-agent forecasting can be viewed under the EPO framework (Sec. 3) by comparing energy structures, the method for solving the energy optimization problem (6), and learning types. These works provide additional empirical evidence that modeling real-world multi-agent interactions as EPOs is promising. DSDNet [35] and JFP [34] use a neural network (NN) to approximate the energy and perform sampling-based (SB) optimization. The sampled future states can be generated by unrolling the dynamics (Eq. 1) with controls obtained from model-based (MB) or learning-based (LB) sampling. Both works use the probabilistic interpretation of EBMs with a conditional density given by $p_\theta(\mathbf{u}|\mathbf{o}) = \frac{1}{Z}\exp(-E_\theta(\mathbf{u},\mathbf{o}))$ and normalization constant Z, which is often intractable to compute in closed form. TGL [2] uses the IFT to learn the energy, which requires convergence to an optimal solution [32]. The energy is a linear combination of energy features $c_j(\cdot)$, given by $E_{\mathbf{w}}(\mathbf{u},\mathbf{o}) = \sum_j \mathbf{w}_j c_j(\mathbf{o},\mathbf{u})$, with w_j as an inferred weight. The approach uses gradient-based convex minimization (concave maximization), whereas the required convexity of c is restrictive for general real-world scenarios (e.g., curvy lanes). Additionally, the experiments are limited by the dataset size (25 samples) and only involve two agents. In contrast, we propose a non-convex nonlinear gradient-based solution with type (iii) learning (unrolling). Neural networks as energy approximations are more expressive and can overcome the design of features necessary for domains like SDV to represent human behavior [36]. On the other hand, linear combinations of features allow the incorporation of domain knowledge into the training process and provide a level of interpretability [13, 5]. EBIOC [10] proposes to use EBMs for inverse optimal control with type (i) learning. Unlike [10], we use SOTA neural network structures with a type (iii) formulation and provide deeper analysis in multi-agent scenarios and multi-modal solutions. Lastly, [10] does not fulfill the potential game assumption in their experiments, nor do the authors draw connections between EBMs and game theory. For completeness, we also classified their approach. To the author's best knowledge, besides the new connection of the three fields, this work's application is the first to combine nonlinear differentiable game-theoretic optimization with neural networks and successfully demonstrate its performance on considerably large real-world datasets.

Table 1: A comparison of EPO applications. *Assuming the fully-connected graph of [34], whereas the backbone [12] predicts controls. ⋄ [2] uses hard inequality constraints, whereas the EPO formulation penalizes inequalities in the energy functions. ⋆ Assuming the costs in [10] defines a potential game [18, Theorem 2].

	Energy Structure	Energy Optimization	Learn. Type
JFP [34]*	NN	SB (LB)	CDE
DSDNet [35]	Nonlin.+NN	SB (MB)	CDE
TGL [2]⋄	Convex	GB	IFT
EBIOC [10]⋆	Nonlin./NN	GB	CDE
Sec. 4 (Ours)	Nonlin.	GB	UN

3 Energy-based Potential Games

3.1 Background

Differential Games. Assume we have N agents, and $\mathbf{u}_i(t) \in \mathbb{R}^{n_{u,i}}$ represents the *control* vector, and $\mathbf{x}_i(t) \in \mathbb{R}^{n_{x,i}}$ the *state* vector for each agent $i, 1 \leq i \leq N$ at timestep t with dimensions $n_{u,i}$ and $n_{x,i}$. The joint state evolves according to a time-continuous differential equation:

$$\dot{\mathbf{x}}(t) = f(\mathbf{x}(t), \mathbf{u}(t), t), \tag{1}$$

with dynamics $f(\cdot)$ and starting at the initial state $\mathbf{x}(0) = \mathbf{x}_0$. $\mathbf{u}(t) = (\mathbf{u}_1(t), \cdots, \mathbf{u}_N(t)) \in \mathbb{R}^{n_u}$ and $\mathbf{x}(t) = (\mathbf{x}_1(t), \cdots, \mathbf{x}_N(t)) \in \mathbb{R}^{n_x}$ are the concatenated vectors of all agents controls and states at time t, with dimensions n_u and n_x. Assume each agent minimizes cost

$$C_i(\mathbf{x}_0, \mathbf{u}) = \int_0^T L_i(\mathbf{x}(t), \mathbf{u}(t), t)\mathrm{d}t + S_i(\mathbf{x}(T)) \tag{2}$$

with time horizon T, running cost L_i and terminal cost S_i of agent i. For instance, in robotics applications, the cost function C_i can be designed to encompass the agents' objectives of reaching a specified goal (encoded as S_i), while considering collision avoidance and minimizing control efforts (represented by L_i). $\mathbf{u}_i : [0, T] \times \mathbb{R}^{n_{x,i}} \to \mathbb{R}^{n_{u,i}}$ defines an *open-loop strategy*[1] and $\mathbf{u}_{-i}$ defines the

[1] Open-loop strategies provide equivalence between strategy and control actions for all time instants [3]. Hence, for clarity, we omitted to introduce a new variable for the strategy, and overloaded the notation for $\mathbf{u}_i$ such that it describes the controls of agent i in the time interval $[0, T]$.

open-loop strategy for all players *except* i. Then, $\mathbf{u}$ defines a *joint strategy* for all agents. We can now characterize the differential game with notation: $\Gamma_{\mathbf{x}_0}^T := \left(T, \{\mathbf{u}_i\}_{i=1}^N, \{C_i\}_{i=1}^N, f\right)$. Then, let us recall the definition for NE from [3, Chapter 6]:

Definition 3.1. Given a differential game defined by all agents dynamics (1), and costs (2), a joint strategy $\mathbf{u}^* = (\mathbf{u}_1^*, \ldots, \mathbf{u}_n^*)$ is called an open-loop Nash equilibrium (OLNE) if, for every $i = 1, \ldots, N$ the following conditions hold: $C_i(\mathbf{u}^*) \leq C_i\left(\mathbf{u}_i, \mathbf{u}_{-i}^*\right) \ \forall \mathbf{u}_i$,.
$(\mathbf{u}_i, \mathbf{u}_{-i}^*)$ is a shorthand for $\left(\mathbf{u}_1^*, \ldots, \mathbf{u}_{i-1}^*, \mathbf{u}_i, \mathbf{u}_{i+1}^*, \ldots, \mathbf{u}_N^*\right)$. Intuitively, no agent is incentivized to unilaterally change its strategy, assuming that other agents keep their strategy unchanged.

Potential Differential Games. Finding a NE involves solving *N-coupled* OCPs, which is non-trivial and computationally demanding [2, 18]. However, according to [9] there exists a class of games, namely *potential differential games* (PDGs), in which only the solution of a *single* OCP is required, and its solutions correspond to OLNE of the original game.

Definition 3.2. (cf. [9]) A differential game $\Gamma_{x_0}^T$, is called an open-loop PDG if there exists an OCP such that an open-loop optimal solution of this OCP is an OLNE for $\Gamma_{x_0}^T$.
[18, Theorem 1] (App. C.) implies that such an OCP is given by:

$$\min_{\mathbf{u}(\cdot)} \int_0^T p(\mathbf{x}(t), \mathbf{u}(t), t)\mathrm{d}t + \bar{s}(\mathbf{x}(T)) \quad \text{subject to } \dot{\mathbf{x}}_i(t) = f(\mathbf{x}_i(t), \mathbf{u}_i(t), t) \quad \forall i, \tag{3}$$

under the assumption of decoupled dynamics. Here, $p(\cdot)$ and $\bar{s}(\cdot)$ are so called *potential functions*. It is further shown that in the context of interactive game-theoretic trajectory planning the potential function cost terms of the agents have to be composed of two terms: (i) Cost terms $C_i^{\text{own}}(\mathbf{x}_i(t), \mathbf{u}_i(t))$ that only depend on the state and control of agent i (e.g., tracking costs or control input costs) and (ii) pair-wise coupling terms $C_{i,j}^{\text{pair}}(\mathbf{x}_i(t), \mathbf{x}_j(t))$ between agents i and j, which could encode some common social norms, such as collision avoidance. Further, the coupling terms have to fulfill the property [18, Theorem 2]: $C_{i,j}^{\text{pair}}(\mathbf{x}_i(t), \mathbf{x}_j(t)) = C_{j,i}^{\text{pair}}(\mathbf{x}_i(t), \mathbf{x}_j(t)) \, \forall \, i \neq j$. Intuitively speaking, two agents i and j care the same for common social norms. The potential functions are then given by

$$p(\cdot) = \sum_{i=1}^N C_i^{\text{own}}(\mathbf{x}_i(t), \mathbf{u}_i(t)) + \sum_{1 \leq i < j}^N C_{i,j}^{\text{pair}}(\mathbf{x}_i(t), \mathbf{x}_j(t)) \quad \text{and} \quad \bar{s}(\cdot) = \sum_{i=1}^N C_{i,T}^{\text{own}}(\mathbf{x}_i(T)). \tag{4}$$

3.2 Connecting Potential Differential Games and Energy-based Models

While PDGs provide more tractable solutions to the game, challenges still arise due to unknown game parameters, like preferences for tracking costs or common social norms. Hence, we aim to infer the parameters online using function approximators, such as neural networks, based on an observed context $\mathbf{o}$ (e.g., agent histories). As a solution, we now demonstrate how to connect PDGs to EBMs.

Direct Transcription. Due to its simplicity and resulting low number of optimization variables, we apply single-shooting, a direct transcription method [37], to transform the time-continuous formulation of (3) into a discrete-time OCP. Let the discretized time interval be $[0, T]$ with $0 = t_0 \leq t_1 \leq \cdots \leq t_k \leq \cdots \leq t_K = T$ and $k = 0, \ldots, K$. We assume a piecewise constant control $\mathbf{u}_i(t_k) := \mathbf{u}_{i,k} =$ constant for $t \in [t_k, t_k + \Delta t)$, where $\Delta t = t_{k+1} - t_k$ denotes the time interval. $\mathbf{x}_{i,k+1} = f(\mathbf{x}_{i,k}, \mathbf{u}_{i,k})$ is an approximation of the dynamics in Eq. (3) by an explicit integration scheme. Hence, the state $\mathbf{x}_i(t_k) := \mathbf{x}_{i,k}$ of agent i is obtained by integrating the dynamics based on the controls $\mathbf{u}_i$ and $\mathbf{x}_{i,k}$ is a function of the initial agent state $\mathbf{x}_{i,0}$ and the strategy $\mathbf{u}_i \in \mathbb{R}^{n_{u,i} \times K}$.

EPO Optimization Problem. The solution of the resulting discrete-time OCP is given by

$$\mathbf{u}^* = \arg\min_{\mathbf{u}} \sum_{k=0}^{K-1} p(\mathbf{x}_k, \mathbf{u}_k) + \bar{s}(\mathbf{x}_K), \tag{5}$$

with discrete-time joint state $\mathbf{x}_k \in \mathbb{R}^{n_x}$, control $\mathbf{u}_k \in \mathbb{R}^{n_u}$ at timestep k and joint strategy $\mathbf{u} \in \mathbb{R}^{n_u \times K}$. Assuming inference of the game parameters $\mathbf{p} = \phi_\theta(\mathbf{o})$ based on observations $\mathbf{o}$, and interpreting the cost as an energy function, similar to [10], the cost terms are now functions of the

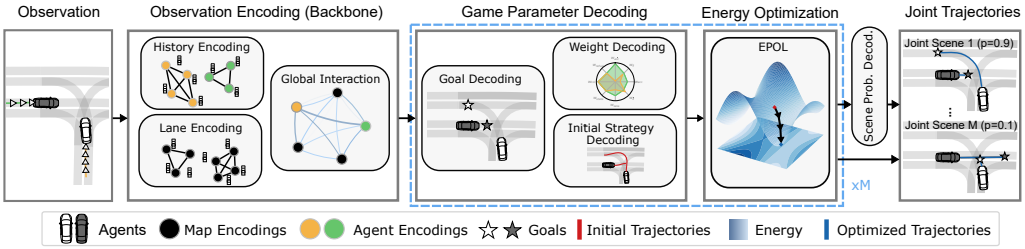

Figure 2: System Architecture. First, a vectorized observation representation is encoded. To handle multi-modality, the decoders predict context-dependent goal positions, initial strategies, and weights for energy parameterization. Parallel optimization problems are solved, resulting in M joint strategies, whereas the dotted blue box represents the parallelization of modes. Unrolling the dynamics generates scene-consistent future joint trajectories. Further, a probability decoder predicts scene probabilities.

observations depended on some learnable parameters θ. Thus, leading to the energy optimization

$$\mathbf{u}^* = \arg\min_{\mathbf{u}} \sum_{i=1}^{N} E^{\text{own}}_{\mathbf{p},i}(\mathbf{u}_i, \mathbf{o}) + \sum_{1 \leq i < j}^{N} E^{\text{pair}}_{\mathbf{p},i,j}(\mathbf{u}_i, \mathbf{u}_j, \mathbf{o}). \tag{6}$$

Here we combined Eq. (4) and (5). As the states are functions of the strategy and the initial observation, state arguments are omitted. $E^{\text{own}}_{\mathbf{p},i}(\cdot)$ is an agent specific energy, which can contain running and terminal costs, and $E^{\text{pair}}_{\mathbf{p},i,j}(\cdot)$ an pairwise interaction energy, whereas both are summed over all K timesteps. The energy interpretation, now allows to apply EBM learning techniques.

4 Practical Implementation

Problem Formulation. We assume access to an object-based representation of the world consisting of agent histories and (optional) map information as visualized for an SDV example in Fig. 2. Let an observation $\mathbf{o} = \{\mathbf{h}, \mathbf{m}\}$ be defined by a sequence of *all agents* historic 2-D positions (x, y), denoted by $\mathbf{h}$, with length H, and by an optional high-definition map $\mathbf{m}$. Our goal is to predict multi-modal future joint strategies $\mathbf{U} \in \mathbb{R}^{n_u \times K \times M}$ and the associate scene-consistent future joint states $\mathbf{X} \in \mathbb{R}^{n_x \times K \times M}$ of all agents and probabilities $\mathbf{PR} \in [0, 1]^M$ for all M joint futures.

General Approach. $\mathbf{U}$ represent M different joint strategies $\mathbf{u}^m \in \mathbb{R}^{n_u \times K}$ with modes $m = 1, \ldots, M$, obtained by M *parallel energy minimizations* (Eq. 6). As minimization with gradient-based solvers can induce problems with local optima, we propose to learn initial strategies $\mathbf{U}^{\text{init}} \in \mathbb{R}^{n_u \times K \times M}$ with a neural network consisting of M strategies $\mathbf{u}^{\text{init},m} \in \mathbb{R}^{n_u \times K}$. In addition, for every mode, the network predicts parameters $\mathbf{p}^m \in \mathbb{R}^{n_p}$, whereas $\mathbf{P} \in \mathbb{R}^{n_p \times M}$ describes the parameters of all modes. Concretely, $\mathbf{p}^m$ contains the weights $\mathbf{w}^m \in \mathbb{R}^{n_w}$ and goals $\mathbf{g}^m \in \mathbb{R}^{2 \times N}$ of all agents.

4.1 Observation Encoding and Decoding of Game Parameters and Probabilities

Observation Encoding. Given observations $\mathbf{o}$, the first step is to encode agent-to-agent and agent-to-lane interactions. Inspired by [22], we use different hierarchical graph neural network *backbones* for observation encoding. We first construct polylines $\mathcal{P}$ based on a vectorized environment representation of the agent histories and map elements. The resulting subgraphs are encoded with separate *agent history* ϕ^{hist} and *lane encoders* ϕ^{lane}, followed by a high-level global interactions graph. Sec. 5 and App. E.1 provide additional information. Let $\mathbf{z}_i$ be the updated feature of agent i extracted from $\mathbf{z}$ resulting after the global interaction graph. Next, we will describe the different decoders of the game parameters $\mathbf{P}$ and initial strategies $\mathbf{U}_{\text{init}}$ which are all implemented by MLPs.

Goal Decoding. As [38], we employ a *goal decoder* to model $p^{\text{goal}}_i(\mathbf{g}_i|\mathbf{o})$ with a categorical distribution over G 2-D goal locations, capturing agent intents (e.g., lane keeping vs. lane changing). We extract multiple goal positions per agent $\mathbf{G}_i \in \mathbb{R}^{2 \times M}$ by selecting the top M goals from $p^{\text{goal}}_i(\cdot)$.

Weight Decoding. The energy structure (7) follows a linear combination of features with multi-modal weights $\mathbf{W} \in \mathbb{R}^{n_w \times M}$, which is the concatenation of all agents time-invariant self-dependent weights $\mathbf{W}^{\text{own}}_i$ and pairwise weights $\mathbf{W}^{\text{pair}}$. These are predicted by two weight decoders. The *agent weight decoder* $\mathbf{W}^{\text{own}}_i = \phi^{\text{own}}(\mathbf{z}_i)$ predicts the weights $\mathbf{W}^{\text{own}}_i$, based on the agent features $\mathbf{z}_i$ from a *single*

agent i. The *interaction weight decoder* $\mathbf{W}^{\text{pair}} = \phi^{\text{pair}}(\mathbf{z}^{\text{all}})$ predicts all pairwise weights $\mathbf{W}^{\text{pair}}$ at once based on the input $\mathbf{z}^{\text{all}}$, which is the concatenation of the agent features $\mathbf{z}_i$ from *all* N agents.

Initial Strategy Decoding. It is important to note that gradient-based methods may not always converge to global or local optima. However, these methods can be highly effective when the solver is initialized close to an optimum [39]. Hence, we learn M initial joint strategies, denoted by $\mathbf{U}^{\text{init}}$. More concretely, the *initial strategy decoder* predicts $\mathbf{U}_i^{\text{init}} = \phi^{\text{init}}(\mathbf{z}_i)$. The parameters of the goal, agent weight, and strategy decoders are shared for all agents. Hence, the computation is parallelized.

Scene Probability Decoding. The M optimizations yield joint strategies $\mathbf{U}$. Unrolling the dynamics yields M future joint state trajectories $\mathbf{X}$. The *scene probability decoder* estimates probabilities for each future $\mathbf{PR} = \phi^{\text{prob}}(\mathbf{z}^{\text{prob}})$ using the concatenation of $\mathbf{z}^{\text{all}}$ and $\mathbf{X}$ as input, denoted by $\mathbf{z}^{\text{prob}}$.

4.2 Energy Optimization

Energy Structure. The EPOL solves the M optimization problems in parallel based on the predicted $\mathbf{P}$ and $\mathbf{U}^{\text{init}}$, whereas the energies in Eq. (6) have the structure of a linear combination of weighted nonlinear vector-valued functions $c(\cdot)$ and $d(\cdot)$ given by

$$E_{\mathbf{p},i}^{\text{own}}(\mathbf{u}_i^m, \mathbf{o}) = \frac{1}{2} \left\| (\mathbf{w}_i^{\text{own},m})^\intercal c(\mathbf{u}_i^m, \mathbf{g}_i^m) \right\|^2, \; E_{\mathbf{p},i,j}^{\text{pair}}(\mathbf{u}_i^m, \mathbf{u}_j^m, \mathbf{o}) = \frac{1}{2} \left\| (\mathbf{w}_{i,j}^{\text{pair},m})^\intercal d(\mathbf{u}_i^m, \mathbf{u}_j^m) \right\|^2, \tag{7}$$

which allows incorporating domain knowledge into the training process. $\mathbf{w}_i^{\text{own},m}$, $\mathbf{w}_{i,j}^{\text{pair},m}$, and $\mathbf{u}_i^m \in \mathbb{R}^{n_u \times K}$ describe the weight vectors and strategies of mode m and agent i. $c(\cdot)$ includes agent-dependent costs, which, for example, can induce goal-reaching behavior while minimizing control efforts. $d(\cdot)$ is a distance measure between two agent geometries.

Differentiable Optimization. The structure of (7) allows us to solve parallel optimizations using the differentiable Nonlinear Least Square solvers of [32]. The implementation uses the second-order Levenberg–Marquardt method [40]. Hence, we minimize Eq. (6) by iteratively taking S steps $\mathbf{u}_{s+1} = \mathbf{u}_s + \alpha \Delta \mathbf{u}$. s describes the iteration index with $s = 1, \ldots, S$ and α is a stepsize $0 < \alpha \leq 1$. $\Delta \mathbf{u}$ is found by linearizing the energy around the current joint strategy $\mathbf{u}$ and subsequently solving a linear system [32]. During training, we can then backpropagate gradients through the unrolled *inner loop energy minimization* based on a loss function of the *outer loop loss minimization*.

4.3 Training Objectives and Transfer to Control

Training Objectives. We follow prior work [23, 38] and minimize the multi-task loss $\mathcal{L} = \lambda_1 \mathcal{L}^{\text{imit}} + \lambda_2 \mathcal{L}^{\text{goal}} + \lambda_3 \mathcal{L}^{\text{prob}}$, with scaling factors $\lambda_1, \lambda_2, \lambda_3$. The imitation loss $\mathcal{L}^{\text{imit}}$ is a distance of the joint future closest to the ground truth. $\mathcal{L}^{\text{goal}}$ computes the negative log-likelihood (NLL) using the predicted goals $\mathbf{G}$ and $\mathcal{L}^{\text{prob}}$ the NLL for the future joint states $\mathbf{X}$. App. E.4 provides further details.

Transfer to Single-Agent Control. Assuming a controllable robot with index $i = 0$ in the scene, we adopt a similar approach as [41]. We extract the most likely mode index $m^* = \arg\max_m \mathbf{PR}^m$ and the robot's strategy $\mathbf{u}_{i=0}^{m^*}$. Using model-predictive control (MPC), we can execute the first control vector $\mathbf{u}_{i=0,k=0}^{m^*}$ or rely on an underlying tracking controller. The receding horizon MPC formulation and iterative solution also account for possible errors resulting from the open-loop assumption.

5 Experimental Evaluation

The experiments investigate the following research questions: *Q1*: Is the approach applicable to different motion forecasting backbones, and does it enhance the predictive performance? *Q2*: Can the method predict multi-modal joint futures in an interpretable manner?

Evaluation Environments. *RPI* is a dataset of simulated multi-modal mobile robot pedestrian interaction (RPI) constructed based on the implementation of [41]. *exiD* is a real-world dataset of interactive scenarios, captured by drones at different highway-ramp locations [42]. The implementation of [43] and the *CARLA* simulator [44] are used to construct a dataset and closed-loop simulations of interactive SDV left-turn scenarios. The datasets contain 60338 (RPI), 290735 (exiD), and 2959 (CARLA) samples, respectively. Details are given in App. D.

Metrics. We employ standard prediction metrics [45, 23, 34]. The *minADE* calculates the L_2 norm of a *single-agent trajectory* out of M predictions with the minimal distance to the groundtruth. The *minFDE* is similar to the minADE but only evaluated at the last timestep. The *minSADE* and *minSFDE* are the scene-level equivalents to minADE and minFDE, calculating the L_2 norm between *joint trajectories* and joint ground truth [26]. We further calculate the overlap rate *OR* as [34] of the most likely-joint prediction, which measures the scene consistency. When using marginal prediction methods, joint metrics (minSADE, minSFDE, OR) are computed by first ordering the single agent predictions according to their marginal probabilities and constructing a joint scenario accordingly.

Baselines. *Constant Velocity* (CV) is a kinematic baseline, achieving good results for predicting pedestrians [46] or highway vehicles [10]. We also use the following SOTA architectures as baselines and observation encoding backbones: *V-LSTM* [20], *HiVT-M*, a modified version of [47], and the unpublished model *VIBES*. For details, refer to App. E. These baselines employ a marginal loss formulation, minimizing the minADE for trajectory regression and using a classification loss similar to [38] to estimate probabilities. However, this formulation of a *marginal distribution per actor* may lead to inconsistencies in future trajectories. For a fair comparison, we introduce `Backbone`+*SC* baselines that predict control values like [48] and minimize a scene-consistent loss, consisting of minSADE and the same scene probability loss as our approach ($\mathcal{L} = \mathcal{L}^{\text{imit}} + \mathcal{L}^{\text{prob}}$ from Sec. 4.3). These baselines also approximate a *joint distribution* over future states *per scene*.

Energy Features and Dynamics. We use an explicit Euler-forward integration scheme of dynamics $f_i(\cdot)$ (dynamically-extended unicycles [49]). Agents' geometries are approximated by a circle of radius r_i. Hence, $d(\cdot)$ in Eq. (7) is a point-to-point distance, active when the circles overlap. In the RPI experiments, specific features in $c(\cdot)$ penalize deviations from a goal location, high controls and control derivations, velocities, as well as violations of state and control bounds. In the exiD and CARLA experiments, agent-dependent features also penalize high distances to a reference line of the goal, and differences from a reference velocity. App. E.3 provides additional information.

5.1 Results

Quantitative and Qualitative Evaluation. Tab. 2 and 3, summarize the results of the baselines and their extension with our approach on the exiD and RPI datasets. Our implementation consistently outperforms the baselines in all joint metrics (minSADE, minSFDE, OR) across all backbones (V-LSTM, VIBES, HiVT-M) *without backbone-specific hyperparameter tuning of our method* due to the game-theoretic inductive bias. Especially joint metrics are important, as they measure the scene consistency, which is also underlined in a recent study [50]. To answer Q1, we conclude that our approach can be applied to different backbones and improves predictions in joint metrics.

Table 2: Predictive performance comparison on the exiD test dataset. ADE, FDE, SADE and FDE are computed as the minimum over $M = 5$ predictions in $[\mathrm{m}]$. **Bold** marks the best result and underlined the second best of each group of approaches, which uses the same observation encoding backbone. Lower is better.

	Marginal ↓		Joint ↓		
Method	ADE	FDE	SADE	SFDE	OR
V-LSTM	1.25	3.64	1.98	3.90	0.031
+ SC	0.82	1.95	1.07	2.63	0.010
+ Ours	**0.80**	**1.89**	**0.99**	**2.37**	**0.008**
VIBES	1.35	**1.68**	1.93	2.93	0.021
+ SC	**0.73**	1.71	1.01	2.47	0.007
+ Ours	0.83	1.99	**0.99**	**2.40**	**0.006**
HiVT-M	1.83	2.02	2.39	2.86	0.013
+ SC	**0.78**	**1.90**	1.04	2.57	0.008
+ Ours	0.83	1.97	**1.03**	**2.48**	**0.007**
CV	1.16	2.87	1.16	2.87	0.010

Observe in Fig. 3 (a) how our approach predicts the correct joint future (mode 2) with a high probability and also outputs another reasonable alternative future. To answer Q2, consider the definition of interpretability in the context of SDV provided by [13]. Our approach allows us to visualize intermediate latent representations (feature weights). For example, the weight for reaching reference velocity w_{vref} is higher in mode 1 for the blue vehicle and lower for the red vehicle. That provides insights into the decision-making process (App. F.2). Note that the weights are normalized with respect to the maximum weight of the individual feature in the sequence. *Hence, weights of the same feature are comparable between the two modes, but weights of different features are not.* In Fig. 3 (b), we present a closed-loop control

result in another scenario, where the SDV accurately anticipates that the other vehicle will yield. Furthermore, Fig. 3 (c) visualizes the most likely joint predictions in an interactive merging scenario (lane change of yellow and red) at high speed. Our model accurately predicts the resulting interaction and produces scene-consistent predictions. Additional results can be found in App. F.

Ablation Study. This section ablates the influence of learning an initialization and using goal-related features as illustrated in Tab. 4. Turning off goal features inhibits goal-reaching behavior, which is important for modeling human behavior [36]. Hence, the performance declines in all metrics. When the learned initialization is turned off and agents' controls are initialized with zeros, we observe a decline in performance. These findings highlight the importance of the algorithmic components.

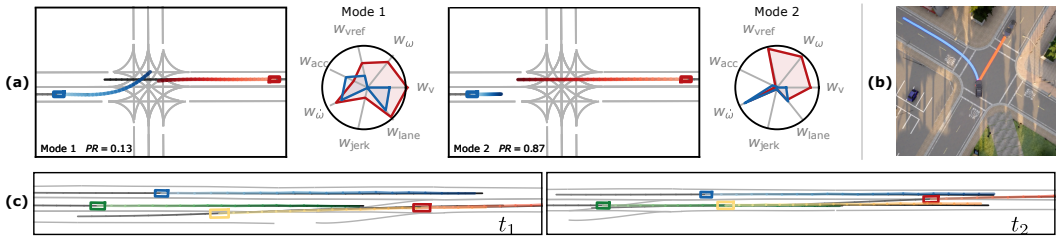

Figure 3: Qualitative joint predictions. (a) Open-loop (CARLA) for two modes and self-dependent weights $\mathbf{W}_i^{\text{own}}$ normalized w.r.t. the maximum weight in the sequence. (b) Closed-loop control (CARLA) of the SDV (blue) in another scenario. (c) Open-loop (exiD) for two time steps. (b) and (c) illustrate the most likely joint trajectories. The ground truth is visualized with colors from dark grey to black and the map in light grey. Agent trajectories have different colors.

Table 3: Predictive performance of different methods on the RPI test dataset. The metrics and formatting are the same as in Tab. 2, but $M = 2$.

	Marginal ↓		Joint ↓		
Method	ADE	FDE	SADE	SFDE	OR
V-LSTM	0.12	0.20	0.13	0.23	0.006
+ SC	0.04	0.11	0.04	0.11	0.001
+ Ours	**0.03**	**0.08**	**0.03**	**0.08**	**0.000**

Table 4: Ablation study investigating the impact of learned initialization, and goal-related features (exiD testset, VIBES backbone).

	Marginal ↓		Joint ↓		
Method	ADE	FDE	SADE	SFDE	OR
Ours (full)	**0.83**	**1.99**	**0.99**	**2.40**	**0.006**
No init	0.89	2.04	1.01	2.47	0.007
No goal	0.89	2.13	1.10	2.64	0.009

5.2 Limitations

As commonly reported in the literature, game-theoretic motion planning suffers from increased runtime, especially with an increasing number of agents. While our implementation scales well with the number of modes (nearly constant runtime) due to parallelization (see App. E.4), that effect is also present in our non-runtime optimized application. Here, future work could apply decentralized optimizations similar to [51], to further reduce the runtime. Without loss of generality, our implementation is limited by a fixed number of agents due to the requirement of fixed-size optimization variables of [32]. Future work should dynamically identify interacting agents, considering that not all agents are constantly engaged in a scene. That could be done utilizing the already existing attention mechanisms, similar to [52], or heuristics, similiar to [34] and would also address the works runtime limitation. The proposed EPO further opens opportunities for various future algorithms, combining different types of energy structures, optimization, and differentiation techniques, which could also overcome scalability limitations induced by the use of energy features in our application. While this study and [2, 34, 35] provide empirical evidence, the assumption of pairwise equal preferences for collision avoidance is restrictive and could be relaxed in future work at the cost of higher runtime.

6 Conclusions

This work presented a connection between differential games, optimal control, and EBMs. Based on these findings, we developed a practical implementation that improves the performance of various neural networks in scene-consistent motion forecasting experiments and is applicable for motion control. Similar to [53], we hope that by highlighting the connection between these fields, researchers in these three communities will be able to recognize and utilize transferable concepts across domains.

References

[1] I. Teeti, S. Khan, A. Shahbaz, A. Bradley, and F. Cuzzolin. Vision-based intention and trajectory prediction in autonomous vehicles: A survey. In L. D. Raedt, editor, *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*, pages 5630–5637. International Joint Conferences on Artificial Intelligence Organization, 7 2022. Survey Track.

[2] P. Geiger and C.-N. Straehle. Learning game-theoretic models of multiagent trajectories using implicit layers. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(6):4950–4958, May 2021.

[3] T. Başar and G. J. Olsder. *3. Noncooperative Finite Games: N-Person Nonzero-Sum, Dynamic Noncooperative Game Theory, 2nd Edition*, pages 77–160. 1998.

[4] S. Le Cleac'h, M. Schwager, and Z. Manchester. Algames: a fast augmented lagrangian solver for constrained dynamic games. *Autonomous Robots*, 46, 01 2022.

[5] X. Liu, L. Peters, and J. Alonso-Mora. Learning to play trajectory games against opponents with unknown objectives. art. arXiv:2211.13779, 2023.

[6] W. B. Knox, A. Allievi, H. Banzhaf, F. Schmitt, and P. Stone. Reward (mis)design for autonomous driving. *Artificial Intelligence*, 316:103829, 2023. ISSN 0004-3702.

[7] C. Diehl, T. S. Sievernich, M. Krüger, F. Hoffmann, and T. Bertram. Uncertainty-aware model-based offline reinforcement learning for automated driving. *IEEE Robotics and Automation Letters*, 8(2):1167–1174, 2023.

[8] L. Peters, D. Fridovich-Keil, V. Royo, C. Tomlin, and C. Stachniss. Inferring objectives in continuous dynamic games from noise-corrupted partial state observations. In *Robotics: Science and Systems XVII*, 2021.

[9] A. Fonseca-Morales and O. Hernández-Lerma. Potential Differential Games. *Dynamic Games and Applications*, 8(2):254–279, June 2018.

[10] Y. Xu, J. Xie, T. Zhao, C. Baker, Y. Zhao, and Y. N. Wu. Energy-based continuous inverse optimal control. *IEEE Transactions on Neural Networks and Learning Systems*, pages 1–15, 2022.

[11] T. Salzmann, B. Ivanovic, P. Chakravarty, and M. Pavone. Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data. In A. Vedaldi, H. Bischof, T. Brox, and J.-M. Frahm, editors, *Computer Vision – ECCV 2020*, pages 683–700, Cham, 2020. Springer International Publishing.

[12] 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. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 7814–7821, 2022.

[13] É. Zablocki et al. Explainability of vision-based autonomous driving systems: Review and challenges. *Interntional Journal Computer Vision*, 2022.

[14] Y. LeCun, S. Chopra, R. Hadsell, M. Ranzato, and F. Huang. A tutorial on energy-based learning. *Predicting structured data*, 2006.

[15] P. Florence, C. Lynch, A. Zeng, O. A. Ramirez, A. Wahid, L. Downs, A. Wong, J. Lee, I. Mordatch, and J. Tompson. Implicit behavioral cloning. In *Proceedings of the 5th Conference on Robot Learning*, volume 164 of *Proceedings of Machine Learning Research*, pages 158–168. PMLR, 08–11 Nov 2022.

[16] D. Monderer and L. S. Shapley. Potential games. *Games and Economic Behavior*, 14(1): 124–143, 1996. ISSN 0899-8256.

[17] D. Fridovich-Keil, E. Ratner, L. Peters, A. D. Dragan, and C. J. Tomlin. Efficient iterative linear-quadratic approximations for nonlinear multi-player general-sum differential games. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1475–1481, 2020.

[18] T. Kavuncu, A. Yaraneri, and N. Mehr. Potential ilqr: A potential-minimizing controller for planning multi-agent interactive trajectories. In *Robotics: Science and Systems XVII*, 07 2021.

[19] S. Le Cleac'h, M. Schwager, and Z. Manchester. Lucidgames: Online unscented inverse dynamic games for adaptive trajectory prediction and planning. *IEEE Robotics and Automation Letters*, 6(3):5485–5492, 2021.

[20] S. Ettinger, S. Cheng, B. Caine, C. Liu, H. Zhao, S. Pradhan, Y. Chai, B. Sapp, C. R. Qi, Y. Zhou, Z. Yang, A. Chouard, P. Sun, J. Ngiam, V. Vasudevan, A. McCauley, J. Shlens, and D. Anguelov. Large scale interactive motion forecasting for autonomous driving: The waymo open motion dataset. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9710–9719, October 2021.

[21] P. Kothari, S. Kreiss, and A. Alahi. Human trajectory forecasting in crowds: A deep learning perspective. *IEEE Transactions on Intelligent Transportation Systems*, 23(7):7386–7400, 2022.

[22] J. Gao, C. Sun, H. Zhao, Y. Shen, D. Anguelov, C. Li, and C. Schmid. Vectornet: Encoding hd maps and agent dynamics from vectorized representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[23] J. Ngiam, V. Vasudevan, B. Caine, Z. Zhang, H. L. Chiang, J. Ling, R. Roelofs, A. Bewley, C. Liu, A. Venugopal, D. J. Weiss, B. Sapp, Z. Chen, and J. Shlens. Scene transformer: A unified architecture for predicting future trajectories of multiple agents. In *The Tenth International Conference on Learning Representations, ICLR 2022, April 25-29, 2022*, 2022.

[24] N. Deo and M. M. Trivedi. Convolutional social pooling for vehicle trajectory prediction. In *2018 IEEE Conference on Computer Vision and Pattern Recognition Workshops, CVPR Workshops 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 1468–1476. Computer Vision Foundation / IEEE Computer Society, 2018.

[25] C. Tang and R. R. Salakhutdinov. Multiple futures prediction. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[26] S. Casas, C. Gulino, S. Suo, K. Luo, R. Liao, and R. Urtasun. Implicit latent variable model for scene-consistent motion forecasting. In *Computer Vision – ECCV 2020*, pages 624–641, Cham, 2020. Springer International Publishing.

[27] Y. Song and D. P. Kingma. How to train your energy-based models. art. arXiv:2101.03288, 2021.

[28] D. Belanger, B. Yang, and A. McCallum. End-to-end learning for structured prediction energy networks. In *International Conference on Machine Learning*, 2017.

[29] G. E. Hinton. Training products of experts by minimizing contrastive divergence. *Neural Computation*, 14(8):1771–1800, 2002.

[30] M. U. Gutmann and A. Hyvärinen. Noise-contrastive estimation of unnormalized statistical models, with applications to natural image statistics. *Journal of Machine Learning Research*, 13(11):307–361, 2012.

[31] B. Amos and J. Z. Kolter. OptNet: Differentiable optimization as a layer in neural networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 136–145. PMLR, 06–11 Aug 2017.

[32] L. Pineda, T. Fan, M. Monge, S. Venkataraman, P. Sodhi, R. T. Q. Chen, J. Ortiz, D. DeTone, A. Wang, S. Anderson, J. Dong, B. Amos, and M. Mukadam. Theseus: A library for differentiable nonlinear optimization. In *Advances in Neural Information Processing Systems*, volume 35, pages 3801–3818. Curran Associates, Inc., 2022.

[33] P. Karkus, B. Ivanovic, S. Mannor, and M. Pavone. Diffstack: A differentiable and modular control stack for autonomous vehicles. In *Proceedings of The 6th Conference on Robot Learning*, volume 205 of *Proceedings of Machine Learning Research*, pages 2170–2180. PMLR, 14–18 Dec 2023.

[34] W. Luo, C. Park, A. Cornman, B. Sapp, and D. Anguelov. Jfp: Joint future prediction with interactive multi-agent modeling for autonomous driving. In *Proceedings of The 6th Conference on Robot Learning*, volume 205 of *Proceedings of Machine Learning Research*, pages 1457–1467. PMLR, 14–18 Dec 2023.

[35] W. Zeng, S. Wang, R. Liao, Y. Chen, B. Yang, and R. Urtasun. Dsdnet: Deep structured self-driving network. In *Computer Vision – ECCV 2020*, pages 156–172, Cham, 2020. Springer International Publishing.

[36] M. Naumann, L. Sun, W. Zhan, and M. Tomizuka. Analyzing the suitability of cost functions for explaining and imitating human driving behavior based on inverse reinforcement learning. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5481–5487, 2020.

[37] J. T. Betts. *Practical Methods for Optimal Control and Estimation Using Nonlinear Programming, Second Edition*. Society for Industrial and Applied Mathematics, second edition, 2010.

[38] H. Zhao, J. Gao, T. Lan, C. Sun, B. Sapp, B. Varadarajan, Y. Shen, Y. Shen, Y. Chai, C. Schmid, C. Li, and D. Anguelov. Tnt: Target-driven trajectory prediction. In *Proceedings of the 2020 Conference on Robot Learning*, volume 155 of *Proceedings of Machine Learning Research*, pages 895–904. PMLR, 16–18 Nov 2021.

[39] P. L. Donti, D. Rolnick, and J. Z. Kolter. DC3: A learning method for optimization with hard constraints. In *International Conference on Learning Representations (ICLR)*, 2021.

[40] S. J. Wright and J. Nocedal. *Numerical Optimization: Least-Squares Problems*, pages 245–269. Springer New York, New York, NY, 2006.

[41] L. Peters, D. Fridovich-Keil, C. J. Tomlin, and Z. N. Sunberg. Inference-based strategy alignment for general-sum differential games. In *Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems*, AAMAS '20, page 1037–1045. International Foundation for Autonomous Agents and Multiagent Systems, 2020.

[42] T. Moers, L. Vater, R. Krajewski, J. Bock, A. Zlocki, and L. Eckstein. The exid dataset: A real-world trajectory dataset of highly interactive highway scenarios in germany. In *2022 IEEE Intelligent Vehicles Symposium (IV)*, pages 958–964, 2022.

[43] N. Rhinehart, J. He, C. Packer, M. A. Wright, R. McAllister, J. E. Gonzalez, and S. Levine. Contingencies from observations: Tractable contingency planning with learned behavior models. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 13663–13669, 2021.

[44] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun. CARLA: An open urban driving simulator. In *Proceedings of the 1st Annual Conference on Robot Learning*, volume 78 of *Proceedings of Machine Learning Research*, pages 1–16. PMLR, 13–15 Nov 2017.

[45] K. M. K. S. L. Nicholas Rhinehart, Rowan McAllister. Precog: Prediction conditioned on goals in visual multi-agent settings. In *Proceedings of (ICCV) International Conference on Computer Vision*, pages 2821 – 2830, October 2019.

[46] C. Schöller, V. Aravantinos, F. Lay, and A. Knoll. What the constant velocity model can teach us about pedestrian motion prediction. *IEEE Robotics and Automation Letters*, 5(2):1696–1703, 2020.

[47] Z. Zhou, L. Ye, J. Wang, K. Wu, and K. Lu. Hivt: Hierarchical vector transformer for multi-agent motion prediction. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8813–8823, 2022.

[48] H. Cui, T. Nguyen, F.-C. Chou, T.-H. Lin, J. Schneider, D. Bradley, and N. Djuric. Deep kinematic models for kinematically feasible vehicle trajectory predictions. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 10563–10569, 2020.

[49] S. M. Lavalle. *Planning Algorithms*. Cambridge University Press, 2006.

[50] E. Weng, H. Hoshino, D. Ramanan, and K. Kitani. Joint metrics matter: A better standard for trajectory forecasting. art. arXiv:2305.06292, 2023.

[51] Z. Williams, J. Chen, and N. Mehr. Distributed potential ilqr: Scalable game-theoretic trajectory planning for multi-agent interactions. art. arXiv:2303.04842, 2023.

[52] C. Hazard, A. Bhagat, B. R. Buddharaju, Z. Liu, Y. Shao, L. Lu, S. Omari, and H. Cui. Importance is in your attention: Agent importance prediction for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 2532–2535, June 2022.

[53] C. Finn, P. Christiano, P. Abbeel, and S. Levine. A connection between generative adversarial networks, inverse reinforcement learning, and energy-based models. art. Neural Information Processing Systems, Workshop on Adversarial Training, 2016.