On a Connection between Differential Games, Optimal Control, and Energy-based Models for Multi-Agent Interactions

Christopher Diehl [1] **Tobias Klosek** [1] **Martin Krüger** [1] **Nils Murzyn** [2] **Torsten Bertram** [1]

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' preferences and goals. To address these challenges, we show 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 formulation, this work introduces 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 experiments using simulated mobile robot pedestrian interactions and real-world automated driving data provide empirical evidence that the game-theoretic layer improves the predictive performance of various neural network backbones.

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 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, challenges arise due to the additional verification requirements in safety-critical domains. Hence, Geiger & Straehle (2021) postulate 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.

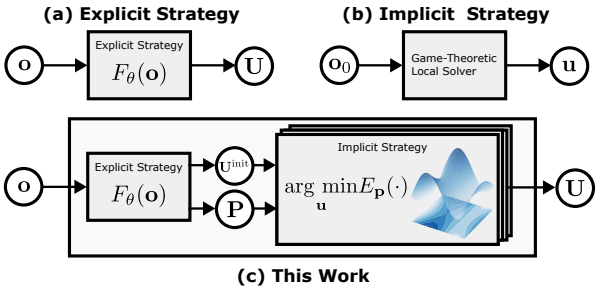

Figure 1. Strategy Representations. (a) Learned explicit strategy producing a multi-modal solution $\mathbf{U}$ based on observation $\mathbf{o}$. (b) Implicit game-theoretic strategy, which only considers the current observation $\mathbf{o}_0$ and hence does not account for the prior interaction evolution. Moreover, it has fixed game parameters and produces a local (uni-modal) solution $\mathbf{u}$. (c) This work infers 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 performs M game-theoretic energy minimizations in *parallel* in a learnable end-to-end framework.

Game-theoretic approaches, utilizing differential/dynamic games (Başar & Olsder, 1998), 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 (Le Cleac'h et al., 2022; Liu et al., 2023) based on the current observation $\mathbf{o}_0$, resulting in a single (uni-modal) joint strategy $\mathbf{u}$. While finding a suitable cost parametrization is non-trivial (Knox et al., 2023; Diehl et al., 2023) for the robot (e.g., an self-driving vehicle (SDV) in the open world), 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, such as goals and cost weights, necessary (Peters et al., 2021).

On the other hand, neural network-based approaches achieve state-of-the-art (SOTA) performance on motion forecast-

[1]Institute of Control Theory and Systems Engineering, TU Dortmund University, Dortmund, Germany [2]ZF Friedrichshafen AG, Artificial Intelligence Lab, Saarbrücken, Germany. Correspondence to: Christopher Diehl <christopher.diehl@tu-dortmund.de>.

Workshop on New Frontiers in Learning, Control, and Dynamical Systems at the International Conference on Machine Learning (ICML), Honolulu, Hawaii, USA, 2023. Copyright 2023 by the author(s).

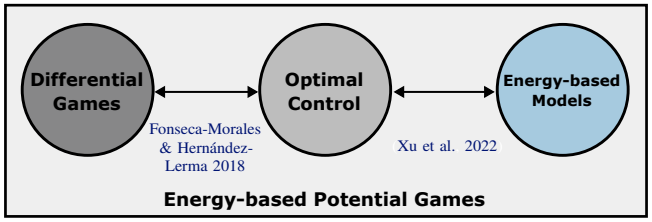

Figure 2. Energy-based Potential Games as a connection of different research areas.

ing benchmarks. Works like (Salzmann et al., 2020) and (Varadarajan et al., 2022) 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 the observed context $\mathbf{o}$. Although these models achieve impressive results, they are considered as low interpretable black-box models with limited controllability (Zablocki et al., 2022). *How can we leverage the benefits of both groups of approaches?*

Energy-based neural networks (LeCun et al., 2006), a type of generative models, provide an *implicit* mapping

$$\mathbf{u}^* = \arg\min_{\mathbf{u}} E_\theta(\mathbf{u}, \mathbf{o}). \tag{1}$$

Florence et al. (2022) demonstrate 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 (Monderer & Shapley, 1996). Hence, we combine *explicit strategies* for initialization and parameter inference with *implicit strategies* as shown in Fig. 1.

Contribution. This paper contributes the following: Theoretically, this work proposes *Energy-based Potential Game* (EPO) as a class of methods connecting differential games, optimal control, and energy-based models (EBMs), as visualized in Fig. 2. We further show how existing approaches can be unified under this framework. Application-wise, this work proposes a differentiable Energy-based Potential Game Layer (EPOL), which is combined with hierarchical neural network backbones in a novel system architecture. Third, we demonstrate that our practical implementation improves the performance of different SOTA neural network backbone architectures in simulated and real-world motion forecasting experiments.

2. Related Work

Game-Theoretic Planning. Game-theoretic motion planning approaches (Fridovich-Keil et al., 2020; Le Cleac'h et al., 2022; Liu et al., 2023), 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, Geiger & Straehle (2021) and Kavuncu et al. (2021) formulate the problem as potential game (Monderer & Shapley, 1996; Fonseca-Morales & Hernández-Lerma, 2018), enabling the solution of only a single optimal control problem (OCP). Different works use filtering techniques (Le Cleac'h et al., 2021) or inverse game-solvers (Peters et al., 2021) to learn game parameters. However, these methods have been primarily evaluated in simulation, and the learning objectives of Peters et al. (2021) assumed unimodal distributions. In contrast, our work utilizes SOTA neural networks to infer cost parameters and strategy initializations end-to-end. We further, provide empirical evidence using a interactive real-world driving dataset.

Data-driven Motion Forecasting. Motion forecasting approaches using neural networks currently represent the SOTA in benchmarks for various applications, such as human vehicle (Ettinger et al., 2021) or pedestrian prediction (Kothari et al., 2022). Most works focus on modeling interactions in the observation encoding part. For instance, the attention (Gao et al., 2020) or convolutional social pooling mechanism (Deo & Trivedi, 2018) are commonly employed. Zeng et al. (2020) and Luo et al. (2023) also utilize EBMs, with sampling instead of gradient-based optimization, like in this work. App. B provides a extension of the related motion forecasting applications. In general, our approach is complementary to the developments in motion forecasting, as it can be integrated on top of various network architectures as later shown in Section (5).

Differentiable Optimization for Machine Learning. Advances in differentiable optimization (Amos & Kolter, 2017; Pineda et al., 2022) enable our work allowing the combination of optimization problems with learning-based models such as neural networks. Geiger & Straehle (2021) use a concave maximization-based motion forecasting application, which is restrictive for general real-world scenarios. Additionally, the experiments are limited by the dataset size (max. 25 samples) and only involve two agents. The concurrent work of Liu et al. (2023) proposes a combination with a differentiable optimization planner evaluated on a simulated dataset and does not account for multi-modal demonstrations and predictions. Both works serve as proof of concepts and utilize simple network architectures with only two hidden layers. By contrast, we account for multi-modal behavior, evaluate on larger real-world datasets, and show that our game-theoretic layer can be easily applied to different SOTA neural networks. Moreover, both approaches draw no connection between EBMs and game theory.

Energy-based Model. The works of LeCun et al. (2006) and Song & Kingma (2021) provide reviews for EBMs. Belanger et al. (2017) identify three main paradigms for energy learning: (i) Conditional Density Estimation, (ii) Exact En-

ergy Minimization (iii) Unrolled Optimization. Models of the first group (i) use the probabilistic interpretation that low-energy regions have high probability. Approaches utilize maximum likelihood estimation (MLE) (Song & Kingma, 2021), noise contrastive (NC) divergence (Hinton, 2002), or NC estimation (Gutmann & Hyvärinen, 2012) learning objectives. Type (ii) methods solve the energy optimization problem exactly and differentiate by employing techniques such as the *implicit function theorem* (Amos & Kolter, 2017). Methods from type (iii), like (Belanger et al., 2017), approximate the solution with a finite number of gradient steps and backpropagate through the unrolled optimization. One closely related application is EBIOC (Xu et al., 2022), which proposes to use EBMs for inverse optimal control with a type (i) MLE learning. The application of this paper investigates type (iii) methods. Unlike EBIOC, we use SOTA neural network structures and provide deeper analysis in multi-agent scenarios and multi-modal solutions. Lastly, EBIOC draws no connections to game theory. However, EBIOC can also be viewed under the EPO formulation (Section 3.3) under specific assumptions.

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.

3. Energy-based Potential Games

This section describes the EPO framework. After introducing the game-theoretic background based on the works of Başar & Olsder (1998), Fonseca-Morales & Hernández-Lerma (2018) and Kavuncu et al. (2021) in Section 3.1, we will show how to connect the potential game with EBMs in Section 3.2 and discuss how different approaches can be unified under the EPO framework in Section 3.3.

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. $n_{u,i}$ and $n_{x,i}$ denote the dimension of the control and state of agent i. The overall state evolves according to a time-continuous differential equation with dynamics $f(\cdot)$:

$$\dot{\mathbf{x}}(t) = f(\mathbf{x}(t), \mathbf{u}(t), t). \tag{2}$$

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 $n_u = \sum_i n_{u,i}$ and $n_x = \sum_i n_{x,i}$. Assume each agent minimizes cost

$$C_i(\mathbf{x}_0, \mathbf{u}) = \int_0^T L_i(\mathbf{x}(t), \mathbf{u}(t), t)\mathrm{d}t + S_i(\mathbf{x}(T)) \tag{3}$$

with time horizon T, running cost L_i and terminal cost S_i of agent i. Costs are assumed to be conflicting rendering the game *noncooperative*. 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 simultaneously 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 open-loop strategy for all players *except* i. Then, $\mathbf{u}$ defines a *joint strategy* for all agents. Let $\mathbf{x}_0$ be the initial measured state. We can now characterize the differential game with notation: $\Gamma^T_{\mathbf{x}_0} := \left(T, \{\mathbf{u}_i\}_{i=1}^N, \{C_i\}_{i=1}^N, f\right)$.

Then, let us recall the following definition for NE from Başar & Olsder (1998, Chaper 6):

Definition 3.1. Given a differential game defined by all agents dynamics (2), and costs (3), a joint strategy $\mathbf{u}^* = (\mathbf{u}_1^*, \ldots, \mathbf{u}_n^*)$ is called an open-loop Nash equilibrium (OLNE) if, for every $i = 1, \ldots, N$,

$$C_i(\mathbf{u}^*) \leq C_i\left(\mathbf{u}_i, \mathbf{u}_{-i}^*\right) \ \forall \mathbf{u}_i, \tag{4}$$

where $(\mathbf{u}_i, \mathbf{u}_{-i}^*)$ 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 speaking, no agent is incentivized to unilaterally change its strategy, assuming that all other agents keep their strategy unchanged.

Potential Differential Games. Finding a NE involves solving *N-coupled* OCPs, which is non-trivial and computationally demanding (Geiger & Straehle, 2021; Kavuncu et al., 2021). However, according to Fonseca-Morales & Hernández-Lerma (2018) 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. Fonseca-Morales & Hernández-Lerma (2018)) A differential game $\Gamma^T_{x_0}$, 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 $\Gamma^T_{x_0}$.

Theorem 1 from Kavuncu et al. (2021) (see also App. C) implies, under the assumption of decoupled dynamics

$$\dot{\mathbf{x}}_i(t) = f(\mathbf{x}_i(t), \mathbf{u}_i(t), t) \quad \forall i, \tag{5}$$

[1]Open-loop strategies provide equivalence between strategy and control actions for all time instants (Başar & Olsder, 1998). 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 $[0, T]$.

that such an OCP[2] is given by:

$$\min_{\mathbf{u}(\cdot)} \int_0^T p(\mathbf{x}(t), \mathbf{u}(t), t)\mathrm{d}t + \bar{s}(\mathbf{x}(T)) \quad \text{subject to Equation (5).} \tag{6}$$

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 (Theorem 2 Kavuncu et al. (2021)): $C_{i,j}^{\text{pair}}(\mathbf{x}_i(t), \mathbf{x}_j(t)) = C_{j,i}^{\text{pair}}(\mathbf{x}_i(t), \mathbf{x}_j(t)) \, \forall \, i \neq j$. Intuitively speaking, two agents i and j care the same for common social norms. The potential functions are then given by

$$\begin{aligned} p(\cdot) &= \sum_{i=1}^{N} C_i^{\text{own}}(\mathbf{x}_i(t), \mathbf{u}_i(t)) + \sum_{1 \leq i < j}^{N} C_{i,j}^{\text{pair}}(\mathbf{x}_i(t), \mathbf{x}_j(t)) \\ \bar{s}(\cdot) &= \sum_{i=1}^{N} C_{i,T}^{\text{own}}(\mathbf{x}_i(T)). \end{aligned} \tag{7}$$

3.2. Connecting Potential Differential Games with Energy-based Models

While PDGs provide more tractable solutions to the original game, challenges still arise due to unknown game parameters, like preferences for tracking costs or common social norms. Hence, this work aims to infer the parameters online using function approximators, such as neural networks, based on an observed context $\mathbf{o}$ (e.g., agents' histories, map information, or raw-sensor data in robotics applications). We now demonstrate how to connect PDGs to EBMs, laying the foundation for unifying various existing applications (Section 3.3) and our practical solution in Section 4.

Direct Transcription. Due to its simplicity and resulting low number of optimization variables, we apply single-shooting, a direct transcription method (Betts, 2010), to transform the time-continuous formulation of (5) and (6) 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. Assume an approximation of the system dynamics (5) by an explicit integration scheme with $\mathbf{x}_{i,k+1} = f(\mathbf{x}_{i,k}, \mathbf{u}_{i,k})$. By applying single-shooting, the state $\mathbf{x}_i(t_k) := \mathbf{x}_{i,k}$ is obtained by integrating the system dynamics based on the controls $\mathbf{u}_{i,k}$ for $k = 0, \ldots, K-1$. Hence, states $\mathbf{x}_{i,k}$ of agent i are a function of the initial (measured) agent state $\mathbf{x}_{i,0}$ and the strategy $\mathbf{u}_i \in \mathbb{R}^{n_{u,i} \times K}$.

EPO Optimization Problem. Let us now formulate the solution of the resulting discrete-time OCP given by:

$$\mathbf{u}^* = \arg\min_{\mathbf{u}} \sum_{k=0}^{K-1} p(\mathbf{x}_k, \mathbf{u}_k) + \bar{s}(\mathbf{x}_K), \tag{8}$$

with discrete-time joint state $\mathbf{x}_k \in \mathbb{R}^{n_x}$ and control $\mathbf{u}_k \in \mathbb{R}^{n_u}$ at timestep k and joint strategy $\mathbf{u} \in \mathbb{R}^{n_u \times K}$. Now assume inference of the game parameters $\mathbf{p} = \phi_\theta(\mathbf{o})$ based on observations $\mathbf{o}$, and we can interpret the cost as an energy function, similar to Xu et al. (2022). Hence, the cost terms are now functions of the observations and also depend on some learnable parameters θ. That leads to the energy optimization problem

$$\mathbf{u}^* = \arg\min_{\mathbf{u}} \sum_{i=1}^{N} E_{\mathbf{p},i}^{\text{own}}(\mathbf{u}_i, \mathbf{o}) + \sum_{1 \leq i < j}^{N} E_{\mathbf{p},i,j}^{\text{pair}}(\mathbf{u}_i, \mathbf{u}_j, \mathbf{o}). \tag{9}$$

Here we combined Equations (7) and (8). Remember that states are functions of the strategy (sequence of controls) and the initial observation. Hence, state arguments are omitted. $E_{\mathbf{p},i}^{\text{own}}(\mathbf{u_i}, \mathbf{o})$ represent an agent specific energy, which can contain running and terminal costs, and $E_{\mathbf{p},i,j}^{\text{pair}}(\mathbf{u}_i, \mathbf{u}_j, \mathbf{o})$ an pairwise interaction energy, whereas both are summed over all K timesteps. The energies could depend on $\mathbf{o}$ in two ways, explicitly and implicitly, through the inferred parameters $\mathbf{p}$[3]. The interpretation as an energy, now allows to apply EBMs techniques for learning.

3.3. Discussion of Related Applications

This section revisits the literature and shows how existing applications from the field of multi-agent forecasting can be viewed under the EPO framework. Table 1 provides a comparison in terms of the energy structure, the method for solving the energy optimization problem (9), and the learning type (see Section 2 EBMs). These works provide additional empirical evidence that modeling real-world multi-agent interactions as PDG is promising. *Note that none of these approaches draw connections between PDGs, optimal control, and EBMs.*

DSDNet (Zeng et al., 2020) and JFP (Luo et al., 2023) use neural networks to approximate the energy and optimize

[2]The assumption of decoupled dynamics seems reasonable in interactive (robot) trajectory planning settings, as the coupling between agents mainly occurs due to the coupling of agents' cost functions, such as collision avoidance Kavuncu et al. (2021).

[3]The energies of our implementation (Section 4) depend on $\mathbf{o}$ through $\mathbf{p} = \phi_\theta(\mathbf{o})$. Moreover, energy features dependent on the states are functions of the initial state $\mathbf{x}_0$ extracted from $\mathbf{o}_0$, which induces another dependence on the observations.

Table 1. A comparison of EPO applications. MB: Model-based. GB: Gradient-based. CDE: Conditional Density Estimation. IFT: Implicit Function Theorem. UN: Unrolling. *Assuming the fully-connected graph of Luo et al. (2023). ⋄ TGL uses hard inequality constraints to force a forward movement of the agents, whereas the proposed EPO framework penalizes inequalities in the energy functions. ⋆ Assuming that the cost follows the PDG formulation.

	Energy Structure	Energy Optimization	Learning Type
Luo et al. (2023)*	NN	Sampling (Learn.)	CDE
Zeng et al. (2020)	Nonlin.+NN	Sampling (MB)	CDE
Geiger & Straehle (2021)⋄	Convex	GB	IFT
Xu et al. (2022)⋆	Nonlin./NN	GB	CDE
Section (4)	Nonlin.	GB	UN

via sampled future states. These future states can be generated by unrolling the dynamics with controls generated by model-based or learning-based sampling. Both works use the probabilistic interpretation of EBMs with a conditional density given by $p_\theta(\mathbf{u}|\mathbf{o}) = \frac{1}{Z} = \exp(-E_\theta(\mathbf{u}, \mathbf{o}))$ learned by MLE, with network parameters θ, and normalization constant Z, which is often intractable to compute in closed-form and needs to be approximated. In contrast, TGL (Geiger & Straehle (2021)) uses the implicit function theorem to learn the energy, which requires convergence to an optimal solution $\mathbf{u}^*$ (Pineda et al., 2022). The energy has the structure of a linear combination of features $E_\mathbf{w}(\mathbf{u}, \mathbf{o}) = \sum_j \mathbf{w}_j c_j(\mathbf{o}, \mathbf{u}) = \mathbf{w}^\intercal c(\mathbf{o}, \mathbf{u})$. $\mathbf{w}$ describes the vector of inferred cost/energy function weights, and $c_j(\cdot)$ the cost/energy features. 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). In contrast, Section 4 proposes a non-convex nonlinear gradient-based solution. Further, our approach learns by backpropagation through unrolled nonlinear optimization problems. Xu et al. (2022) does not make a potential game assumption in their multi-agent control experiments, nor do the authors draw connections between EBMs and PDGs. However, we also classified their approach under the EPO framework for completeness. Neural networks as approximations of the energy are more expressive and can overcome the design of features, which is necessary for domains like SDV (Naumann et al., 2020) to approximate human behavior. On the other hand, linear combinations of features allow to incorporate domain knowledge into the training process and provide a level of interpretability (Zablocki et al., 2022; Liu et al., 2023).

4. Practical Implementation

This section introduces a practical implementation that integrates a differentiable EPO formulation into the training process of neural networks for multi-agent forecasting.

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. 3. 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 future multi-modal 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 $m = 1, \dots, M$. $\mathbf{U}$ is obtained by M parallel gradient-based optimizations of energies defined by Equation (Eq.) (9). As minimizing the non-linear energy 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 parameter vectors $\mathbf{p}^m \in \mathbb{R}^{n_p}$ with dimension 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. Algorithm 1 provides a pseudocode of the the training and inference procedure.

Algorithm 1 Multi-Agent Forecasting

Require: observation $\mathbf{o} = \{\mathbf{h}, \mathbf{m}\}$, ground truth future joint trajectory $\mathbf{x}_{\text{GT}}$
$\mathbf{z} = \phi^{\text{glob}}(\phi^{\text{agent}}(\mathbf{h}), \phi^{\text{lane}}(\mathbf{m}))$ {observation encod.}
for $1 \leq i \leq N$ **do**
  $\mathbf{G}_i = \phi^{\text{goal}}(\mathbf{z}_i)$ {goal decod.}
  $\mathbf{W}_i^{\text{own}} = \phi^{\text{own}}(\mathbf{z}_i)$ {agent weight decod.}
  $\mathbf{U}_i^{\text{init}} = \phi^{\text{init}}(\mathbf{z}_i)$ {initial strategy decod.}
end for{in parallel for all agents i}
$\mathbf{W}^{\text{pair}} = \phi^{\text{pair}}(\mathbf{z}^{\text{all}})$ {interaction weight decod.}
for $1 \leq m \leq M$ **do**
  Parameterize energies (10), (11) with $\mathbf{g}^m$, $\mathbf{w}^m$
  Initialize optimization with $\mathbf{u}^m$
  Gradient-based minimization of (9)
end for{in parallel for all modes m}
$\mathbf{X} = f(\mathbf{x}_0, \mathbf{U})$ {unroll dynamics}
$\mathbf{PR} = \phi^{\text{prob}}(\mathbf{z}^{\text{prob}})$ {scene prob. decod.}
if Training **then**
  Update parameters θ based on $\nabla\mathcal{L}$ (12)
end if

4.1. Observation Encoding

Given observations $\mathbf{o}$, the first step is to encode agent-to-agent and agent-to-lane interactions. Inspired by Gao et al. (2020), this work uses 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 his-*

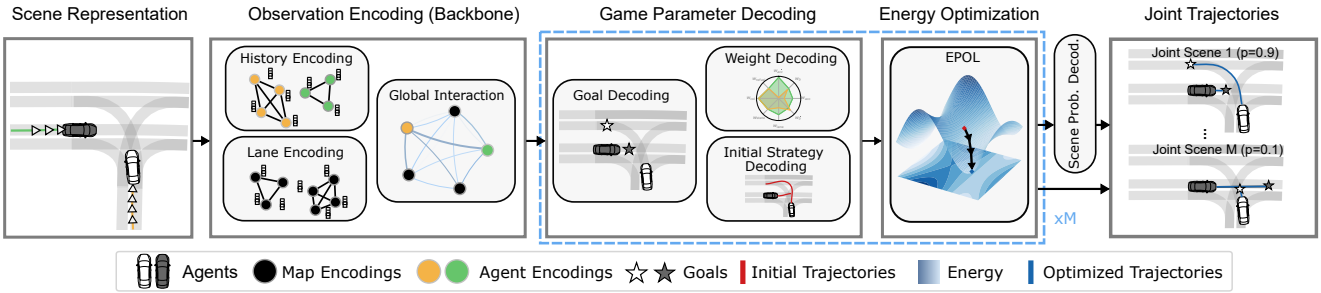

Figure 3. System Architecture of the implementation. Agent histories and lanes are encoded using a vectorized representation. The resulting features capture agent-to-agent and agent-to-lane interactions in a global interaction graph. These features are then passed through different decoders. To handle multi-modality in demonstrations, the decoders predict context-dependent goal positions, initial strategies, and weights for energy parameterization. Parallel energy optimization problems are solved in the *inner loop*, resulting in M joint strategies. The dotted blue box represents the parallelization of modes. Unrolling the dynamics generates scene-consistent future joint trajectories. Additionally, a probability decoder predicts scene probabilities. During training, the learnable parameters are updated in the *outer loop* optimization problem based on a multi-task loss.

tory ϕ^{hist} and *lane encoders* ϕ^{lane}, followed by a network to model high-level interactions in a global graph ϕ^{glob}. The result is an updated latent polyline feature vector $\mathbf{z}$. Section 5 and App. E.1 provide additional information.

4.2. Game Parameter Decoding

Let $\mathbf{z}_i$ be the updated feature of agent i after the global interaction graph extracted from $\mathbf{z}$. 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 multilayer perceptrons (MLP).

Goal Decoding. Human navigation is partially determined by goals (Wolbers & Hegarty, 2010). Hence, the *goal decoder* aims to provide a distribution $p_i^{\text{goal}}(\mathbf{g}_i|\mathbf{o})$ of future 2-D goal positions $\mathbf{g}_i \in \mathbb{R}^2$. In our SDV experiments, we follow (Zhao et al., 2021) and model $p_i^{\text{goal}}(\cdot)$ with a categorical distribution over G discrete goal locations to account for multi-modality over 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 $p_i^{\text{goal}}(\cdot)$. $\mathbf{G}_i$ is used to parameterize goal-related features[4] of energy (10).

Weight Decoding. The energy structure (10),(11) 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 $\mathbf{W}_i^{\text{own}}$ and pairwise weights $\mathbf{W}^{\text{pair}}$. These are predicted by two weight decoders. The *agent weight decoder* $\mathbf{W}_i^{\text{own}} = \phi^{\text{own}}(\mathbf{z}_i)$ predicts the weights $\mathbf{W}_i^{\text{own}}$, 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.

[4]Note that the proposed method is not restricted to energies using goal-related features. However, these features can improve the predictive performance, as shown later (Tab. 4).

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. (Donti et al., 2021). 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.

4.3. Energy-based Potential Game Layer

The energy-based potential game layer solves the M optimization problems, defined in Eq. (9) in parallel, using the predicted parameters $\mathbf{P}$ and initializations $\mathbf{U}^{\text{init}}$.

Energy Structure. The energies from Eq. (9) have the structure of a linear combination of weighted nonlinear vector-valued functions $c(\cdot)$ and $d(\cdot)$ given by

$$E_{\mathbf{p},i}^{\text{own}}(\mathbf{u}_i^m, \mathbf{o}) = \frac{1}{2}\left\|(\mathbf{w}_i^{\text{own},m})^\intercal c(\mathbf{u}_i^m, \mathbf{g}_i^m)\right\|^2, \quad (10)$$

$$E_{\mathbf{p},i,j}^{\text{pair}}(\mathbf{u}_i^m, \mathbf{u}_j^m, \mathbf{o}) = \frac{1}{2}\left\|(\mathbf{w}_{i,j}^{\text{pair},m})^\intercal d(\mathbf{u}_i^m, \mathbf{u}_j^m)\right\|^2, \quad (11)$$

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. The use of weighted features provides an additional layer of interpretability according to the definition (Zablocki et al., 2022). For instance, a visualization of feature weights provides further insights into the decision-making process. A high weight for reaching a goal lane could indicate a lane change. Remember from Section 3.2 that the future joint states $\mathbf{x}_i^m$ are a function of strategy $\mathbf{u}_i^m$, connected by the

explicit integration scheme of the dynamics. The approach uses an Euler-forward integration scheme with dynamically-extended differentiable unicycle dynamics (see App. E.2) to model pedestrians, mobile robots, or vehicles.

Differentiable Optimization. The structure of (10), (11) allows us to solve parallel optimizations using the differentiable Nonlinear Least Square solvers of Pineda et al. (2022). The implementation uses the second-order Levenberg–Marquardt method (J. Wright & Nocedal, 2006). Hence, we minimize Eq. (9) 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 (Pineda et al., 2022). 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*.

4.4. Scene Probability Decoding

The result of the M parallel optimizations are multi-modal joint strategies $\mathbf{U}$. Unrolling the dynamics leads to M multi-modal future joint state trajectories $\mathbf{X}$ and the goal of the scene probability decoder is to estimate probabilities for each future $\mathbf{PR} = \phi^{\text{prob}}(\mathbf{z}^{\text{prob}})$. The decoder takes as input the concatenation of $\mathbf{z}^{\text{all}}$ and joint trajectories $\mathbf{X}$, denoted by $\mathbf{z}^{\text{prob}}$ and outputs probabilities $\mathbf{PR}$ for the M futures.

4.5. Training Objectives

The implementation follows prior work (Ngiam et al., 2022; Zhao et al., 2021) and minimizes the multi-task loss

$$\mathcal{L} = \lambda_1 \mathcal{L}^{\text{imit}} + \lambda_2 \mathcal{L}^{\text{goal}} + \lambda_3 \mathcal{L}^{\text{prob}}. \tag{12}$$

with scaling factors $\lambda_1, \lambda_2, \lambda_3$ of the different loss terms. The imitation loss $\mathcal{L}^{\text{imit}}$ is a distance of the joint future closest to the ground truth. As $\mathcal{L}^{\text{imit}}$ induces imitation behavior, the energies/costs will be learned such that solving the optimal control problem with the learned energies/costs results in multi-agent imitation[5]. $\mathcal{L}^{\text{goal}}$ computes the negative log-likelihood based on the predicted goals $\mathbf{G}$ locations and $\mathcal{L}^{\text{prob}}$ similarly for the multi-modal future joint states $\mathbf{X}$. Further details are given in App. E.4.

5. Experimental Evaluation

The experiments presented below aim to answer the following research questions: *Q1* Is the methodology applicable to different motion forecasting backbones, and does it enhance the predictive performance? *Q2*: What are the most influential hyperparameters?

Evaluation Environments. The first dataset contains simulated multi-modal mobile robot pedestrian interaction (*RPI*), constructed based on the implementation of Peters et al. (2020). *exiD* is a real-world dataset of interactive scenarios, captured by drones at different locations of highway (Moers et al., 2022). The datasets contain 60338 (RPI) and 290735 (exiD) samples respectively. Methods are tasked to predict joint futures of $T = 4\,\text{s}$. Details are given in App. D.

Metrics. This work follows standard motion forecasting metrics (Nicholas Rhinehart, 2019; Ngiam et al., 2022; Luo et al., 2023). 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 futures, as Casas et al. (2020). The recent study of Weng et al. (2023) highlights the importance of these joint metrics. We further calculate the overlap rate *OR* of the most likely-joint prediction, which measures the scene consistency as described by Luo et al. (2023). When using marginal prediction methods, the 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* (Const. Vel.) is a kinematic baseline, achieving good results for predicting pedestrians (Schöller et al., 2020) or highway vehicles (Xu et al., 2022). The experiments also utilize the following SOTA architectures as baselines and observation encoding backbones. All methods utilize the lane encoders of Gao et al. (2020). *V-LSTM*: Ettinger et al. (2021) encode agent histories with an LSTM (Hochreiter & Schmidhuber, 1997) and a single-stage attention mechanism (Vaswani et al., 2017) directly models the interactions between agents and lanes. *HiVT-M*: Inspired by Zhou et al. (2022), this slightly modified baseline encodes the agent histories with transformers (Vaswani et al., 2017) and uses a two-stage attention mechanism. First, the map-to-agent interactions are modeled, and subsequently, the agent-to-agent interactions. *VIBES*: This unpublished baseline stands for Vectorized Interaction-based Scene Prediction and uses an LSTM for agent encoding and the previously described two-stage attention mechanism. V-LSTM, HiVT-M, and VIBES use a marginal loss formulation by minimizing the minADE for trajectory regression and classification loss similar to Zhao et al. (2021) to estimate probabilities. However, that could lead to inconsistencies in future trajectories as it approximates a *marginal distribution* over future locations *per actor*. `Backbone`+*SC*: To make a fair comparison, we introduce additional base-

[5]We can interpret cost learning as a type of multi-agent inverse reinforcement learning (RL) as Mehr et al. (2023), sometimes also called multi-agent inverse optimal control (Neumeyer et al., 2021). The forward pass is a type of multi-agent model-based RL, utilizing planning with learned cost (Moerland et al., 2023).

Table 2. Predictive performance of the different evaluated methods on the RPI test dataset. ADE, FDE, SADE and FDE are computed as the minimum over $M = 2$ predictions in $[\mathrm{m}]$. **Bold** numbers mark the best result and underlined numbers the second best of the group of approaches, which uses the same observation encoding backbone. Lower numbers are better.

	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**

lines that minimize a scene-consistent loss, consisting of the minSADE and the same scene probability loss as our approach ($\mathcal{L} = \mathcal{L}^{\text{imit}} + \mathcal{L}^{\text{prob}}$ from Eq. (12)). Moreover, these baselines predict control values like Cui et al. (2020). This approach approximates a *joint distribution* over future locations *per scene*. These approaches are closest to our implementation but, in contrast, do not use parameter decoding, nor the EPOL. Grid searches were performed to find the optimal hyperparameters for all baselines to ensure a fair comparison. *The hyperparameters for our method are the same across all backbones* (V-LSTM, VIBES, HiVT-M).

Energy Features and Dynamics. Both experiments use unicycle dynamics. Without loss of generality, agents' geometries are approximated by a circle of radius r_i. Hence $d(\cdot)$ in Eq. (11) is a Euclidean point-to-point distance, active when the circles overlap. In the RPI experiments, specific features in $c(\cdot)$ penalize deviations from goal locations, high controls and control derivations, velocities, as well as violations of state, control, and control derivation bounds. In the exiD experiments, agent-dependent features also penalize high distances to a reference line of the goal, and differences from a reference velocity, but not state or control bounds. App. D and E provide further details regarding the environments, network architectures, and implementation.

5.1. Does the EPOL improve the performance of different observation encoding backbones?

Consider the results in Tab. 2 and 3, showing the results on both datasets when the approach is applied to different observation encoding backbones. Our implementation consistently outperforms the baselines in all joint distance metrics (minSADE and minSFDE) across all backbones *without backbone-specific hyperparameter tuning*. Note that also the overlap rate decreases 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 of Weng et al. (2023). Further, Fig. 4 visualizes qualitative joint predictions in one interactive merging scenarios. Observe how our method produces scene-consistent predictions. For example, the yellow and red car perform a lane

Table 3. Predictive performance of the different evaluated methods on the exiD test dataset. The metrics and formatting is the same as in Tab. 2, but $M = 5$.

	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**
Const. Vel.	1.16	2.87	1.16	2.87	0.010

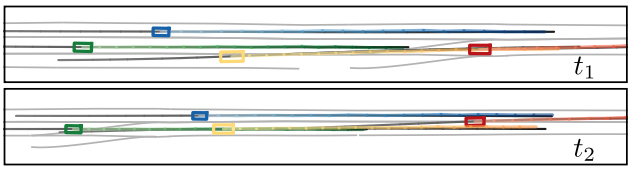

Figure 4. Qualitative results in a interactive exiD scenario for two time steps t_1, t_2. Agents and the most likely future joint trajectories are visualized in different colors. The saturation increases with the number of predicted steps. The groundtruth (history and future) is visualized with colors from dark grey to black. The map is visualized in light grey and the x-axis is about 250 m long.

change at high speeds. Our model predicts, the resulting interaction accurately. Additional multi-modal predictions are located in App. F.1. We conclude that our approach can be applied to different backbones and improves the predictions.

5.2. Ablation Study

The experiments identified that the most influential hyperparameter is the number of steps S during optimization. Fig. 5 visualizes the dependency. Observe how the approach gets reasonable small metrics with all configurations and hence could be used with different numbers of steps. However, while the distance between the closest optimized joint future and GT gets smaller with increasing optimization steps, the initialization gets slightly pushed away from GT. Hence, with more steps, the approach gets less dependent on the initialization. Huang et al. (2023) observe an similar effect.

Further ablations for energy features and learned initialization are given in Tab. 4. Turning off the goal-related features inhibits goal-reaching behavior, which is essential for modeling human behavior (Wolbers & Hegarty, 2010; Naumann et al., 2020). Hence, the performance declines in all metrics. When we turn off the learned initialization and initialize the controls for all agents with zeros, we also observe a

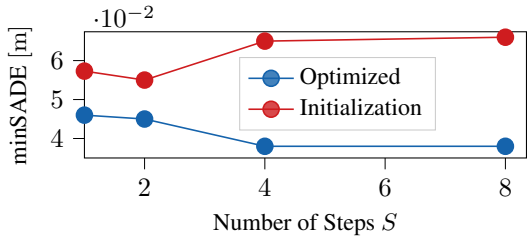

Figure 5. Predictive performance of the initial (red) and optimized strategy (blue) as a function of the number of optimization steps on the RPI validation dataset.

Table 4. Ablation study investigating the impact of learned initialization, and goal-related features on the exiD test dataset. All models use the V-LSTM backbone encoding. **Bold** numbers mark the best result and underlined numbers the second best of the group of approaches, using the same observation encoding backbone. Lower numbers are better.

	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

decline in performance. However, the decline is lower as unicycle dynamics with controls of zeros correspond to a constant velocity and constant turn rate movement, which is reasonable for highway scenarios. Tab. 3 demonstrate that a constant velocity movement is a straightforward yet competitive baseline in highway scenarios, subsequently enhanced through energy optimization. Nevertheless, it fails to attain the performance level exhibited by the learned initialization. The findings in Tab. 4 underline the significance of the algorithmic components in this study.

Figure 6. Mean runtime in [s] for different number of agents and optimizations steps averaged over 100 exiD samples with $M = 4$. The experiments for the number of modes use $S = 2$ and $N = 4$.

5.3. Limitations and Future Work.

As commonly reported in the literature, game-theoretic motion planning approaches suffer 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 Fig. 6), that effect is also present in our non-runtime optimized implementation. However, future work could apply decentralized optimization techniques similar to Williams et al. (2023), to further reduce the runtime. Without loss of generality, our implementation is limited by a fixed number of agents (see App. D) during optimization due to the requirement of fixed-size optimization variables (Pineda et al., 2022). Future work should overcome this issue and dynamically identify interacting agents, as not all agents constantly interact in a scene. That could be done utilizing the already existing attention mechanism, similar to Hazard et al. (2022), and would also address the first runtime limitation. Future work should also explore the generalization of our findings to more complex urban settings. This work considered applications for motion forecasting (open-loop). However, future work could also use our formulation for closed-loop control of one or more agents by executing the most likely trajectory like Peters et al. (2020). The proposed EPO framework further open opportunities for various future algorithms, which combine different types of energy structures, optimization, and differentiation techniques as indicated in Section 3.3.

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. Similiar to Finn et al. (2016), 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, particularly in the development of interpretable and scalable algorithms.

Acknowledgements

This work was supported by the Federal Ministry for Economic Affairs and Climate Action on the basis of a decision by the German Bundestag and the European Union in the Project *KISSaF - AI-based Situation Interpretation for Automated Driving.*

References

Amos, B. and Kolter, J. Z. 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*, pp. 136–145. PMLR, 06–11 Aug 2017.

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

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

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

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

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

Deo, N. and Trivedi, M. M. 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*, pp. 1468–1476. Computer Vision Foundation / IEEE Computer Society, 2018.

Diehl, C., Adamek, J., Krüger, M., Hoffmann, F., and Bertram, T. Differentiable constrained imitation learning for robot motion planning and control. art. arXiv:2210.11796, 2022.

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

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

Ettinger, S., Cheng, S., Caine, B., Liu, C., Zhao, H., Pradhan, S., Chai, Y., Sapp, B., Qi, C. R., Zhou, Y., Yang, Z., Chouard, A., Sun, P., Ngiam, J., Vasudevan, V., McCauley, A., Shlens, J., and Anguelov, D. 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)*, pp. 9710–9719, October 2021.

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

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

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

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

Gao, J., Sun, C., Zhao, H., Shen, Y., Anguelov, D., Li, C., and Schmid, C. 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.

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

Gu, T., Chen, G., Li, J., Lin, C., Rao, Y., Zhou, J., and Lu, J. Stochastic trajectory prediction via motion indeterminacy diffusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 17113–17122, June 2022.

Gupta, A., Johnson, J., Fei-Fei, L., Savarese, S., and Alahi, A. Social gan: Socially acceptable trajectories with generative adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

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

Hazard, C., Bhagat, A., Buddharaju, B. R., Liu, Z., Shao, Y., Lu, L., Omari, S., and Cui, H. 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*, pp. 2532–2535, June 2022.

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

Hochreiter, S. and Schmidhuber, J. Long short-term memory. *Neural computation*, 9:1735–80, 12 1997.

Huang, Z., Liu, H., Wu, J., and Lv, C. Differentiable integrated motion prediction and planning with learnable cost function for autonomous driving. art. arXiv:2207.10422, 2023.

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

Karkus, P., Ivanovic, B., Mannor, S., and Pavone, M. 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*, pp. 2170–2180. PMLR, 14–18 Dec 2023.

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

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

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

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

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

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

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

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

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

Luo, W., Park, C., Cornman, A., Sapp, B., and Anguelov, D. 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*, pp. 1457–1467. PMLR, 14–18 Dec 2023.

Mehr, N., Wang, M., Bhatt, M., and Schwager, M. Maximum-entropy multi-agent dynamic games: Forward and inverse solutions. *IEEE Transactions on Robotics*, 39 (3):1801–1815, 2023.

Moerland, T. M., Broekens, J., Plaat, A., and Jonker, C. M. Model-based reinforcement learning: A survey. *Foundations and Trends® in Machine Learning*, 16(1):1–118, 2023. ISSN 1935-8237.

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

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

Naumann, M., Sun, L., Zhan, W., and Tomizuka, M. 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)*, pp. 5481–5487, 2020.

Neumeyer, C., Oliehoek, F. A., and Gavrila, D. M. General-sum multi-agent continuous inverse optimal control. *IEEE Robotics and Automation Letters*, 6(2):3429–3436, 2021.

Ngiam, J., Vasudevan, V., Caine, B., Zhang, Z., Chiang, H. L., Ling, J., Roelofs, R., Bewley, A., Liu, C., Venugopal, A., Weiss, D. J., Sapp, B., Chen, Z., and Shlens, J. 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.

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

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

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

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

Qi, C. R., Su, H., Mo, K., and Guibas, L. J. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

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

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

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

Tang, C., Zhan, W., and Tomizuka, M. Interventional behavior prediction: Avoiding overly confident anticipation in interactive prediction. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 11409–11415, 2022.

Varadarajan, B., Hefny, A., Srivastava, A., Refaat, K. S., Nayakanti, N., Cornman, A., Chen, K., Douillard, B., Lam, C. P., Anguelov, D., and Sapp, B. Multipath++: Efficient information fusion and trajectory aggregation for behavior prediction. In *2022 International Conference on Robotics and Automation (ICRA)*, pp. 7814–7821, 2022.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. u., and Polosukhin, I. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

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

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

Wolbers, T. and Hegarty, M. What determines our navigational abilities? *Trends in Cognitive Sciences*, 14(3): 138–146, 2010.

Xiao, W., Wang, T.-H., Hasani, R., Chahine, M., Amini, A., Li, X., and Rus, D. Barriernet: Differentiable control barrier functions for learning of safe robot control. *IEEE Transactions on Robotics*, pp. 1–19, 2023.

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

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

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

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

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

Ziegler, J. and Stiller, C. Fast collision checking for intelligent vehicle motion planning. In *2010 IEEE Intelligent Vehicles Symposium*, pp. 518 – 522, 07 2010.

A. List of Abbreviations

Abbreviation	Description
ADE	average displacement error
CDE	conditional density estimation
Const. Vel.	Constant Velocity
EBM	Energy-based Model
EPOL	Energy-based Potential Game Layer
EPO	Energy-based Potential Game
Eq.	equation
FDE	final displacement error
GB	gradient-based
HiVT-M	Hierarchical Vector Transformer Modified
IFT	implicit function theorem
Learn.	learned
LSTM	long short-term memory
MG	model-based
MLE	maximum likelihood estimation
NC	noise contrastive
NE	Nash equilibrium
Nonlin.	nonlinear
OLNE	open-loop Nash equilibrium
OR	overlap rate
OCP	optimal control problem
PDG	potential differential game
RPI	robot pedestrian interaction
SADE	scene average displacement error
SDV	self-driving vehicle
SFDE	scene final displacement error
SOTA	state-of-the-art
UN	unrolling
VIBES	Vectorized Interaction-based Scene Prediction
V-LSTM	Vector-LSTM

B. Extended Related Work

Data-driven Motion Forecasting Deep learning motion forecasting approaches use different observation inputs. Cui et al. (2020) uses birds-eye-view images, which have a memory requirement and can lead to discretization errors. Gao et al. (2020) propose to use a vectorized environment representation instead, and Nicholas Rhinehart (2019) uses raw-sensor data. The approaches often utilize an encoder-decoder structure with convolutional neural networks (Deo & Trivedi, 2018), transformer (Ngiam et al., 2022), or graph neural networks (Casas et al., 2020) to model multi-agent interactions. In addition to deterministic models (Ngiam et al., 2022), various generative models, such as Generative Adversarial Networks (GANs) (Gupta et al., 2018) and Conditional Variational Autoencoder formulations (Salzmann et al., 2020), as well as Diffusion Models (Gu et al., 2022), is used. First, predicting goals in hierarchical approaches like (Zhao et al., 2021), further increases the predictive performance using domain knowledge of the map information. Motion forecasting models can also be conditioned on the control (Diehl et al., 2023) or future trajectory (Salzmann et al., 2020) of one agent. However, these conditional forecasts might lead to overly confident anticipation of how that agent may influence the predicted agents (Tang et al., 2022). To include domain knowledge such as system dynamics into the learning process, it is also common practice (Varadarajan et al., 2022; Cui et al., 2020; Salzmann et al., 2020) to first forecast the future control values of all agents and then to unroll a dynamics model to produce the future states.

Differentiable Optimization for Motion Planning Differentiable optimization has also been applied in motion planning for SDVs. Xiao et al. (2023) and Diehl et al. (2022) impose safety-constraints using differentiable control barrier functions or gradient-based optimization techniques in static environments. Karkus et al. (2023) and Huang et al. (2023) couple

a differentiable single-agent motion planning module with learning-based motion forecasting modules. In contrast, our work performs multi-agent joint optimizations in parallel, derived from a game-theoretic potential game formulation. Game-theoretic formulations can overcome overly conservative behavior when used for closed-loop control Liu et al. (2023).

C. Theorems

This section provides the full theorem of (Kavuncu et al., 2021):

Theorem C.1. *For a differential game* $\Gamma^T_{\mathbf{x}_0} := \left(T, \{\mathbf{u}_i\}_{i=1}^N, \{C_i\}_{i=1}^N, f\right)$*, if for each agent* i*, the running and terminal costs have the following structure* $L_i(\mathbf{x}(t), \mathbf{u}(t), t) = p(\mathbf{x}(t), \mathbf{u}(t), t) + c_i\left(\mathbf{x}_{-i}(t), \mathbf{u}_{-i}(t), t\right)$ *and*

$$S_i(\mathbf{x}(T)) = \bar{s}(\mathbf{x}(T)) + s_i\left(\mathbf{x}_{-i}(T)\right),$$

then, the open-loop control input $\mathbf{u}^* = (\mathbf{u}_1^*, \cdots, \mathbf{u}_N^*)$ *that minimizes the following*

$$\min_{u(\cdot)} \int_0^T p(\mathbf{x}(t), \mathbf{u}(t), t)dt + \bar{s}(\mathbf{x}(T))$$
$$\text{s.t. } \dot{x}_i(t) = f_i\left(\mathbf{x}_i(t), \mathbf{u}_i(t), t\right),$$

is an OLNE of the differential game $\Gamma^T_{\mathbf{x}_0}$*, i.e.,* $\Gamma^T_{x_0}$ *is a potential differential game.*

Proof: See (Kavuncu et al., 2021), with original proof provided by Fonseca-Morales & Hernández-Lerma (2018).

Here besides the potential functions p and $\bar{s}$, s_i and c_i are terms that are required to not depend on the state or control of agent i.

D. Datasets

D.1. RPI

The RPI dataset is a synthetic dataset of simulated mobile robot pedestrian interactions. Multi-modal demonstrations are generated by approximately solving a two-player differential game ($N = 2$) with the iterative linear-quadratic game implementation of Fridovich-Keil et al. (2020) based on different start and goal configurations. Fig. 7 D.1 provides an illustration for the dataset construction. The robot's initial positions (white circle) and goal locations (white star) are the same in all solved games. In contrast, the initial state (dark grey circle) and goal location (dark grey stars) of the pedestrian move on a circle, as illustrated on the left graphic in Fig. 7 D.1. As solving the game once leads to a unimodal local strategy, this work follows the implementation of Peters et al. (2020). It solves the PDG for a given initial configuration multiple times based on different initializations. Afterward the resulting strategies are clustered. The clustered strategies represent multi-modal strategies of the *main game*, and they are visualized in red and yellow in Figure D.1. The agents are tasked to reach a goal location given an initial start state while avoiding collisions and minimizing control efforts. The agents then execute the open-loop controls of the main game's initial strategies. After every time interval $\Delta t = 0.1$, the procedure of game-solving and clustering the results are repeated as long as the agents pass each other. The resulting strategies of the so-called *subgames* are visualized in green and blue on the right of Fig. 7 D.1. Based on the history (dotted red line) and the strategies of the subgame (blue and green), we then build a multi-modal demonstration for the dataset. Note that the main game and the corresponding subgames use the same cost function parametrizations, but the agents' preferences for collision avoidance differ between main games.

The resulting dataset is based on 20 main games and their corresponding subgame solutions. Here we draw collision cost parameters from a uniform distribution to enhance demonstration diversity. The resulting dataset contains 60338 samples, whereas we use 47822 ($\sim$80%) for training, 6228 for validation ($\sim$10%), and 6228 ($\sim$10%) for testing. The test set is constructed based on an unseen main game configuration. The goal is to predict $M = 2$ joint futures of $T = 4\,\mathrm{s}$ based on a history of $H = 1.8\,\mathrm{s}$ with a time interval of $\Delta t = 0.1$.

D.2. exiD

The exiD (Moers et al., 2022) dataset contains $19\,\mathrm{h}$ of real-world highly interactive highway data. Interactions between different type of vehicle classes (vehicles) are rich because the data was recorded by drones flying over seven locations

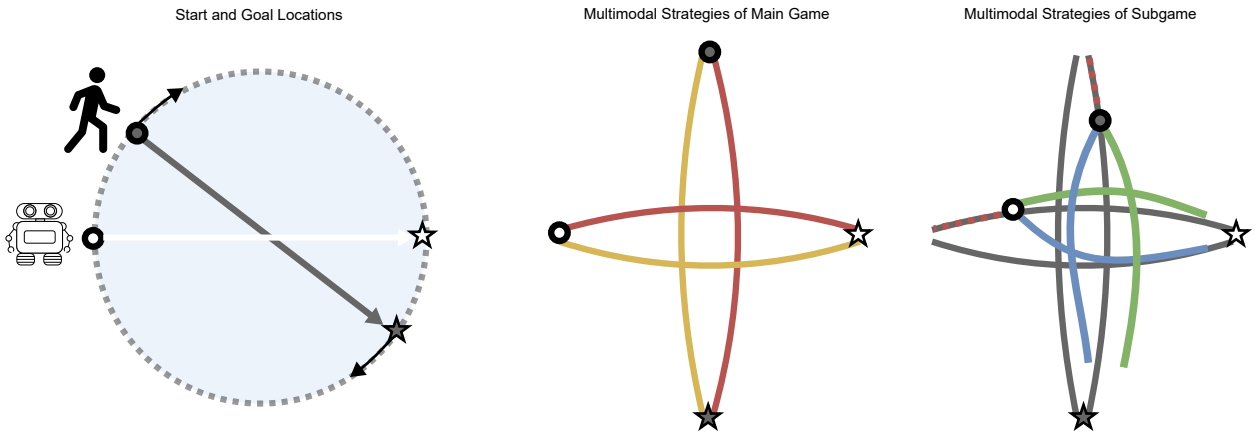

Figure 7. Dataset construction for RPI dataset. Left: First initial and goal states and game parameters are sampled. Middle: A *main game* is solved multiple times based on the sampled game configuration with subsequent result clustering. That leads to multimodal strategies (red and yellow). The agent moves according to the multimodal strategies of the main game. After a time step Δt, a *sub game* is solved. The results are multimodal strategies (blue and green) of the subgame. The histories (dotted red) and multi-modal strategies of the sub game build a demonstration for training and evaluation.

Figure 8. An exemplary highly-interactive scenario from the exiD dataset.

of German highway entries and exits. Highway entries and exits, designed with acceleration and deceleration lanes and high-speed limits, promote interactive lane changes due to high relative speeds between on-ramping and remaining road users. In addition, the most common cloverleaf interchange in Germany requires simultaneous observation of several other road users and gaps between them for safe entry or exit in a short time frame (Moers et al., 2022).

To further increase the interactivity, this work extracts scenarios with $N = 4$ agents in which at least one agent performs a lane change. We choose $N = 4$ as this resulted in the highest number of samples assuming a fixed number of agents. The recordings are then sampled with a frequency of $\Delta t = 0.2\,\mathrm{s}$. The different networks (see Section 5) are tasked to predict $M = 5$ joint futures of length $T = 4\,\mathrm{s}$ based on a history of $H = 1.8\,\mathrm{s}$. The resulting dataset contains 290735 samples, whereas we use 206592 ($\sim$72%) for training, 48745 for validation ($\sim$16%), and 35398 ($\sim$12%) for testing. To investigate the generalization capabilities of the different models, the test set contains *unseen scenarios from a different map* (map 0) than the training and validation scenarios. An exemplary scenario is visualized in Fig. 8 D.1.

E. Implementation Details

This section provides additional information for the used observation encoding backbones and the game parameter decoders.

E.1. Network Architectures

Lane Encoder. In all experiments, the lane encoder ϕ^{lane} uses a PointNet (Qi et al., 2017) like architecture as Gao et al. (2020) with three layers and a width of 64. The polylines are constructed based on vectors that contain a 2-D start and 2-D goal position in a fixed-global coordinate system. Agent polylines also include time step information and are encoded with different encoders depending on the used backbone.

Agent History Encoder. The V-LSTM (Vector-LSTM) (Ettinger et al., 2021) and VIBES (Vectorized Interaction-based Scene Prediction) backbones use an LSTM (Hochreiter & Schmidhuber, 1997) for agent history encoding with depth three and width 64. Our modified HiVT-M (Hierarchical Vector Transformer Modified) (Zhou et al., 2022) implementation uses a transformer (Vaswani et al., 2017) for the encoding of each agent individually. Note that this contrasts with the original implementation, where the encoding transformer already models local agent-to-agent and agent-to-lane interactions. We account for that in a modified global interaction graph as listed below. The transformer has a depth of three and a width of 64.

Global Interaction. The V-LSTM backbones update the polyline features in the global interaction graph with a single layer of attention (Vaswani et al., 2017) as described by Gao et al. (2020). The HiVT-M and VIBES models use a two-stage attention mechanism. First, one layer of attention between the map and agent polyline features, and afterwards a layer of attention between all updated agents features are applied. The global interaction graph has a width of 128.

Game Parameter and Initial Strategy Decoder. The agent weight, goal, and initial strategy decoders are implemented by a 3-layer MLP with a width of 64.

Goal Decoder. The goal decoder follows Zhao et al. (2021). It takes as input the concatenation of an agent feature $\mathbf{z}_i$ and $G = 60$ possible goal points, denoted by $\mathbf{z}^{\text{goal}}$. The goal points are extracted from the centerlines of the current and neighboring lanes. If there exists no neighboring lane, we take the lane boundaries. The decoder ϕ^{goal} then predicts the logits of a categorical distribution per agent $\mathbf{l}_i^{\text{goal}} = \phi^{\text{goal}}(\mathbf{z}^{\text{goal}})$. During training and evaluation; the method samples the M most-likely goals to receive goals $\mathbf{G}_i$ for all modes of a agent i. Probabilities for the goals per agents are computed by $\mathbf{PR}_i^{\text{goal}} = \text{softmax}(\mathbf{l}_i^{\text{goal}})$. The prediction of goals is made in parallel for all agents.

Scene Probability Decoder. The scene probability decoder also uses a 2-layer MLP with width $16 \times M$ and predicts logits $\mathbf{l}^{\text{prob}}$ for the M scene mode. The scene probabilities are derived by applying the softmax operations $\mathbf{PR} = \text{softmax}(\mathbf{l}^{\text{prob}})$.

The goal, agent weight and scene probability decoder use batch normalization. The interaction weight decoder, initial strategy decoder, and transformer agent encoder use layer normalization.

E.2. Dynamics

The discrete-time dynamically-extended unicycle dynamics Lavalle (2006, Chapter 13) are given by:

$$\begin{aligned} x_{k+1} &= x_k + v_k \cos(\theta_t)\,\Delta t \\ y_{k+1} &= y_k + v_k \sin(\theta_t)\,\Delta t \\ v_{k+1} &= v_k + a_k \Delta t \\ \theta_{k+1} &= \theta_k + \omega_k \Delta t \end{aligned} \tag{13}$$

x_k and y_k denote a 2-D position and θ_k the heading in fixed global coordinate system. v_k is the velocity, a_k the acceleration, ω_k the turnrate, δ_k the steering angle and Δt a time interval. Hence, $n_x = 4 \times N$ and $n_u = 2 \times N$.

E.3. Energy Features and Optimization

Energy Features. The energy function in the RPI experiment uses the following agent-dependent features: $c(\cdot) = [c_{\text{goal}}, c_{\text{vel}}, c_{\text{acc}}, c_{\text{velb}}, c_{\text{accb}}, c_{\text{turnr}}, c_{\text{accb}}, c_{\text{turnrb}}]$. In the RPI experiments, the goal is given and not predicted. The agent-dependent energy features in the exiD experiments are given by $c(\cdot) = [c_{\text{goal}}, c_{\text{lane}}, c_{\text{velref}}, c_{\text{vel}}, c_{\text{acc}}, c_{\text{jerk}}, c_{\text{steer}}, c_{\text{turnr}}, c_{\text{turnacc}}]$. c_{goal} is a terminal cost penalizing the position difference of the last state to the predicted goal. c_{lane} minimizes the distance of the state trajectory to the reference lane to which the predicted goal point belongs. Note that different goal points can be predicted for the modes. Hence different lanes can be selected to better model multi-modality. c_{velref} is the difference between the predicted and map-specific velocity limits. The other terms are running cost, evaluated for all timesteps and penalize high velocities (c_{vel}), accelerations (c_{acc}), jerks (c_{jerk}), as well as turn rates (c_{turnr}) and turn accelerations (c_{turnacc}). An index $\mathtt{b}$ marks a soft constraint implemented as a quadratic penalty, active when the constraint is violated. Hence a inequality constraint $g(z) \leq 0$ with optimization variable z is implemented by a feature $\max(0, g(z))$. The interaction feature $d(\cdot)$ is also implemented as such a quadratic penalty. We evaluate the collision avoidance features at every discrete time step in the RPI experiments. In both experiments, agent geometries are approximated by circles of radius r_i, which is accurate for the mobile robot and pedestrian but an over-approximation for vehicles and especially trucks in the highway exiD environment, where we use $r_i = L/2$. L is the length of a vehicle. Future work could also use more accurate vehicle approximations (e.g., multiple circles (Ziegler & Stiller, 2010)) to further evaluate collision avoidance at every time step to increase the predictive performance at a higher runtime and memory cost. In the RPI experiments, we set $r_i = 0.25\,\text{m}$.

Optimization. As the approach already predicts accurate initial strategies $\mathbf{U}^{\text{init}}$, our experiments only required a few optimization steps. Concretely, the results of Tab. 2 and 3 in the main paper use $s = 2$ optimization steps, rendering our approach real-time capable (see Fig. 6). Note while the approach also works, with a higher number of optimization steps (see Fig. 5), our experiments showed that fewer optimization steps lead to similiar results, with decreased runtime and memory requirements due to the predicted initialization. Both experiments use a stepsize of $\alpha = 0.3$. The experiments use a damping factor of $dp = 10$ in the Levenberg-Marquardt solver (Pineda et al., 2022).

E.4. Training Details

Loss Functions. The imitation loss in our experiments is the minSADE (Casas et al., 2020; Weng et al., 2023) given by:

$$\mathcal{L}^{\text{imit}} = \min_{m=1}^{M} \frac{1}{N} \sum_{i=1}^{N} \left\| \mathbf{x}_i^m - \mathbf{x}_{\text{GT}} \right\|^2 \tag{14}$$

It first calculates the average over all distances between agent trajectories $\mathbf{x}_i^m$ from agent i and mode m and the ground truth $\mathbf{x}_{\text{GT}}$. Then the minimum operator is applied to afterwards backpropagate the difference of the joint scene, which is closest to the ground truth. The second loss term $\mathcal{L}^{\text{goal}}$ computes the cross entropy CE for the goal locations averaged over all agents

$$\mathcal{L}^{\text{goal}} = \frac{1}{N} \sum_{i=1}^{N} \text{CE}\left(\mathbf{PR}_i^{\text{goal}}, \mathbf{g}_i^*\right), \tag{15}$$

whereas $\mathbf{g}_i^*$ is the goal location of the set of G possible goals closest to the ground truth goal location. Lastly, $\mathcal{L}^{\text{prob}}$ computes the cross entropy for the joint futures

$$\mathcal{L}^{\text{prob}} = \text{CE}\left(\mathbf{PR}, \mathbf{x}^*\right), \tag{16}$$

whereas $\mathbf{x}^*$ is the predicted joint 2-D position trajectory, which has the smallest distance (measured by minSADE) to the future ground truth joint 2-D position trajectory. We empirically set $\lambda_1 = 1, \lambda_2 = 0.1, \lambda_3 = 0.1$ in Eq. (12).

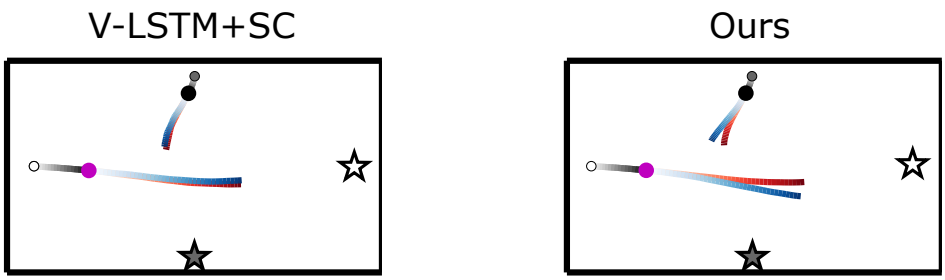

Figure 9. Qualitative comparison of the multi-modal ($M = 2$) joint predictions in the RPI environment. The start and end point of the pink agent are located on a circles with a radius of 3m . The start and endpoint of the black agent are visualized with a grey circle and star. The different modes are visualized in red and blue color. The start and endpoints are located on a circles with a radius of three meter.

All approaches are trained with batch size 32, using the Adam optimizer (Kingma & Ba, 2015). Our models in both experimental environments use a learning rate of 0.00005 across all backbones. Note that the evaluation favors the baselines, as we performed grid searches for their learning rates, whereas our approach uses the same learning rate across all backbones.

Training and evaluation was performed using an AMD Ryzen 9 5900X and a Nvidia RTX 3090.

F. Additional Experiments

F.1. Qualitative Results

This section provides extended qualitative results.

RPI. Fig. 9 F.1 visualizes an exemplary qualitative result of the RPI experiments. Both modes collapsed when using the V-LSTM+SC baseline (explicit strategy). In contrast, this work's implicit approach better models the multi-modality present in the demonstration. Since the dataset contains solutions of games solved with different collision-weight configurations, it can be seen that our proposed method accurately differentiates between different weightings of collisions, as can be seen in the trajectories. This finding aligns with these of Florence et al. (2022), which discovered that implicit models could better represent the multi-modality of demonstrations.

exiD. Fig. 10 F.1 visualizes multi-modal predictions in a highly interactive scenario, where one car (green) and one truck (yellow) merge onto the highway. The green car performs a double-lane change. Note how our model in mode three accurately predicts the future scene evolution and also outputs reasonable alternative futures. For example, in mode one, the green car performs a single lane change, whereas the blue and red cars are also predicted to change lanes. Another multi-modal prediction is visualized in Fig. 11 F.1. Observe again how the ground truth is accurately predicted in this interactive scenario (mode 5), whereas, for example, also other plausible futures are generated. For instance, the yellow vehicle stays longer on the acceleration lane in mode one, whereas in mode three, the green vehicle performs a lane change.

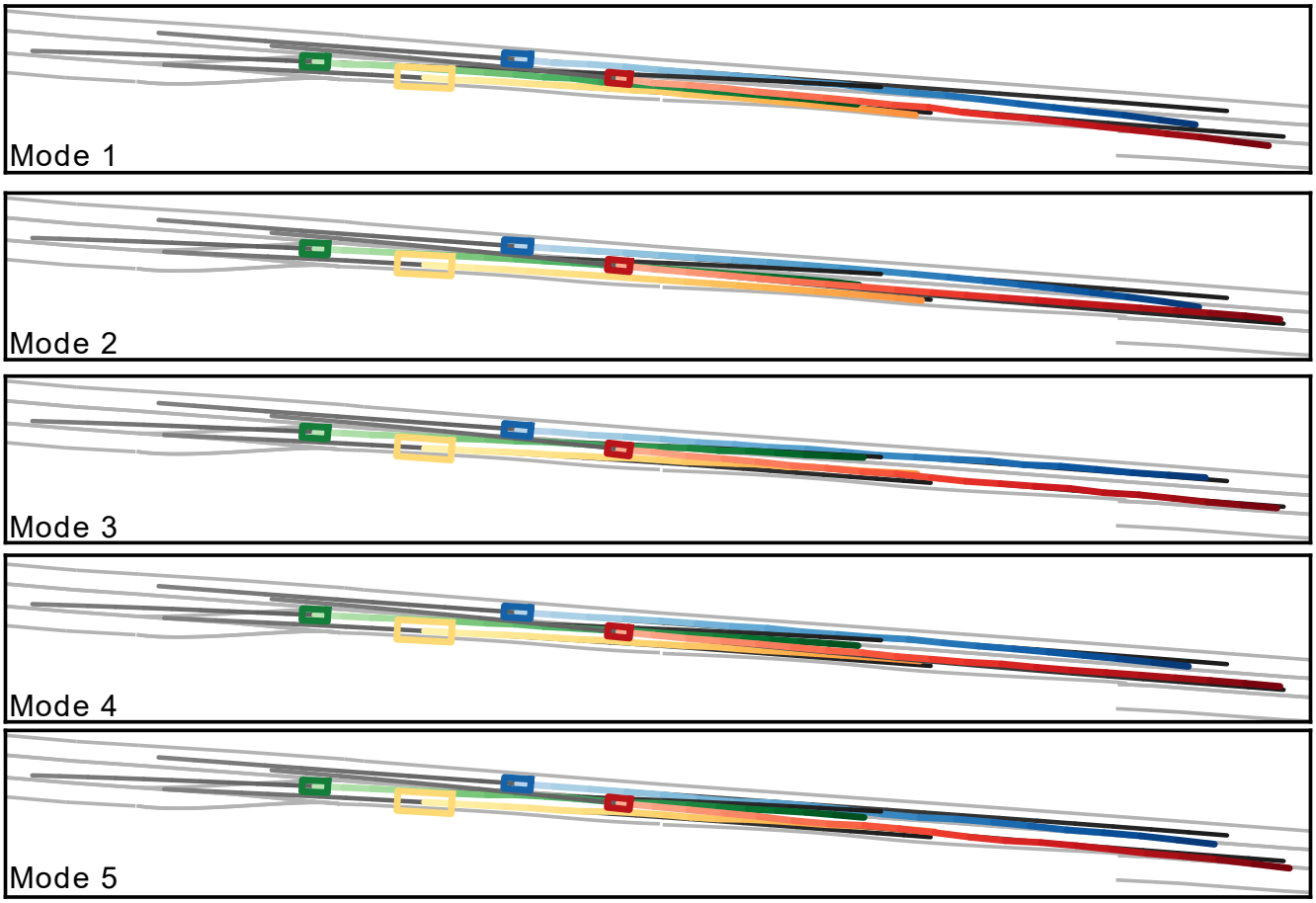

Figure 10. Multi-modal predictions in an interactive scenario, where the green and yellow perform on-ramp merges. The agent trajectories are visualized in different colors, whereas the color changes with an increasing number of predicted steps. The ground truth (history and future) is shown with colors from dark grey to black and the map in light grey.

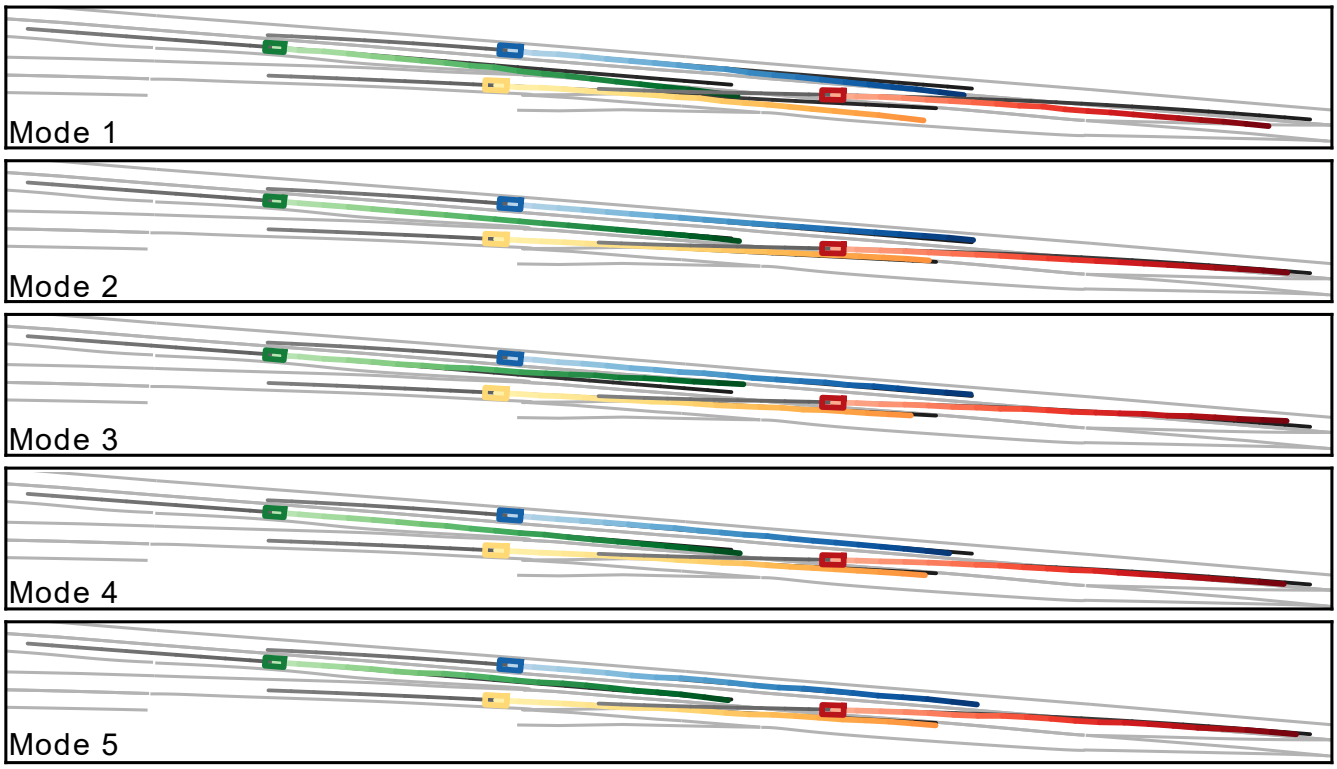

Figure 11. Multi-modal predictions in a interactive scenario, where the yellow and red agents perform lane changes. The agent trajectories are visualized in different colors, whereas the color changes with an increasing number of predicted steps. The ground truth (history and future) is shown with colors from dark grey to black and the map in light grey.