IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2024, Yokohama, Japan. Cite as RA-L paper.

Interactive Joint Planning for Autonomous Vehicles

Yuxiao Chen[1], Sushant Veer[1], Peter Karkus[1], and Marco Pavone[2]

***Abstract*—In highly interactive driving scenarios, the actions of one agent greatly influence those of its neighbors. Planning safe motions for autonomous vehicles (AVs) in such interactive environments, therefore, requires reasoning about the impact of the ego's intended motion plan on nearby agents' behavior. Deep-learning-based models have recently achieved considerable success in trajectory prediction and many models in the literature allow for ego-conditioned prediction. However, leveraging ego-conditioned prediction remains challenging in downstream planning due to the complex nature of neural networks, limiting the planner structure to simple types, e.g., sampling-based planners. Despite the ability of gradient-based planning algorithms, such as model predictive control (MPC), to generate fine-grained high-quality motion plans, it is difficult for them to leverage ego-conditioned prediction due to their iterative nature and need for gradients. We present Interactive Joint Planning (IJP), which bridges MPC with learned prediction models in a computationally scalable manner to provide us with the best of both the worlds. In particular, IJP optimizes over the joint behavior of the ego and the surrounding agents and leverages deep-learned prediction models as prediction priors that the join trajectory optimization tries to stay close to. Furthermore, by leveraging free-end homotopy classes—a novel concept that we introduce in this paper—IJP efficiently searches over diverse motion plans. Closed-loop simulation results show that IJP significantly outperforms baselines without joint optimization or running sampling-based planning.**

***Index Terms*—Autonomous Vehicle Navigation, Machine Learning for Robot Control, Integrated Planning and Learning**

I. Introduction

A cornerstone for safe motion planning for autonomous vehicles is the ability to reason about interactions between the ego vehicle and other traffic agents, such as human-driven vehicles and pedestrians. A standard approach to deal with interactive scenarios is to leverage prediction models—heuristic [1] or data-driven [2], [3]—to generate predictions of the traffic agents' future motions and plan the ego motion accordingly. In particular, various deep-learned prediction models now represent the state of the art in prediction [4], [5], [6]. Modern deep learning prediction models widely use ego-conditioning, i.e., condition the prediction of adjacent agents' motion on the ego's future motion, to improve the prediction quality and capture the interaction between the ego and the agents. The resulting prediction is then consumed by a planner that aims to generate an ego motion plan that avoids collisions and makes progress towards the goal. Depending on how the prediction is consumed, there are two typical styles of planners: sampling-based planners and iterative planners. The former takes a bunch of ego motion samples, calls the prediction model to generate ego-conditioned predictions and searches for a motion plan [3], [7]. An iterative planner, on the other hand, iteratively refines the ego motion plan, with e.g. gradient [8] or Bayesian optimization [9]. While the latter may achieve finer granularity for the ego motion due to iterative refinement, it needs to evaluate the ego motion plan significantly more times than a sampling-based counterpart and the evaluation cannot be parallelized. As a result, the computational complexity prohibits the use of complex deep-learned ego-conditioned prediction models together with an iterative planner—when a prediction model is used, it is typically limited to simple analytical models [10]. In this paper, we propose a computationally tractable approach, called Interactive Joint Planning (IJP), which reasons about interactivity by combining deep-learned prediction models with iterative planners. IJP significantly outperforms other baselines yielding safer motion plans without sacrificing liveness and being overly conservative.

Contributions and paper organization. We propose IJP, which is a model predictive control (MPC)-based planner that is compatible with any (deep learned) prediction model. The two main novelties are (i) IJP jointly optimizes over the ego vehicle and the nearby agents' motion with collision avoidance constraints while penalizing deviation from the unconditioned predicted trajectories of the agents. The "planned" motion for the agents then serve as the ego-conditioned trajectory predictions for those agents and are integrated in the gradient-based planner. (ii) To remedy the local minimum issue of nonconvex optimization, we introduce the novel concept of free-end homotopy that allows us to efficiently explore a diverse range of motions. In particular, free-end homotopy is an extension of homotopy to trajectories that do not share the same end point. We empirically show that IJP significantly outperforms a baseline without joint optimization and is superior to a sampling-based planner baseline in both performance and computation complexity.

II. Related Works

Interactive Planning. Interactive / social-aware planning has been studied extensively in the literature. Some of the early approaches modeled the uncontrolled agents' behavior as Gaussian uncertainty [11] without consideration for the impact of ego behavior on nearby agents. Ignoring the ego's impact can lead to overly conservative motion plans as was famously shown in the freezing robot problem [12]. This led to

Manuscript received: June 19, 2023; Revised September 20, 2023; Accepted October 25, 2023.

This paper was recommended for publication by Editor Aniket Bera upon evaluation of the Associate Editor and Reviewers' comments.

[1]The authors are with NVIDIA Research, NVIDIA Corporation, 2788 San Tomas Expressway, Santa Clara, CA. {yuxiaoc,sveer,pkarkus}@nvidia.com

[2]Marco Pavone is with the Department of Aeronautics and Astronautics, Stanford University, and with NVIDIA Research. {pavone@stanford.edu, mpavone@nvidia.com}

Digital Object Identifier (DOI): see top of this page.

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2024, Yokohama, Japan. Cite as RA-L paper.

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2024, Yokohama, Japan. Cite as RA-L paper.

a plethora of research on navigating crowds while accounting for the reactivity of other agents, such as the joint optimization via Gaussian Process (GP) approach in [12], [13] and the reinforcement learning (RL) approach in [14].

Reactive Behavior Modeling. The crux of interactive planning is to properly model other agents' reactive behavior. Inverse reinforcement learning (IRL) [15] is an obvious choice, e.g., in [16], [17], and it is used subsequently in optimization over the ego motion [18], [19]. Another popular formulation leverages game theory, which assumes that every agent tries to maximize its own utility [20], [21]; however, the computational complexity of equilibrium solving remains a challenge and it is not straightforward to combine game theory with data-driven methods. Partially-observable Markov Decision Process (POMDP)-based methods were applied to interactive planning and inferring the hidden intention of surrounding agents [22], [23], but similar to the game-theoretic approaches, POMDPs also suffer from high computational complexity and they are typically hand-crafted, making them difficult to scale. Other analytical models such as Intelligent Driver Models (IDM) [24] and Probabilistic Graphic Models (PGM) [25] have also been applied to intention estimation and interactive planning, however, they are limited to simple scenarios, such as highway driving. The idea of joint optimization has been studied for conflict resolution [26], yet assumes knowledge about the other agent's cooperativeness.

Deep-Learned Prediction. The above-mentioned methods, though very different in nature, all make assumptions (e.g. rationality) about the surrounding agents' decision processes, and the planner then leverages the assumptions to make the planning problem tractable. In contrast, modern prediction methods are predominantly deep-learned phenomenological models [6], [27], [28], [4], [5], i.e., models trained with data to match the ground truth without a clear explanation of the decision process. While they achieve good prediction accuracy and are capable of ego-conditioned prediction, working with downstream interactive planner remains difficult, as pointed out previously. The expensive inference of ego-conditioned prediction under a large number of ego plans made it prohibitive to evaluate fine-grained ego plans. In [29] the authors use a linear system to represent the ego-conditioned prediction compactly, but the performance is limited by the simplicity of linear systems.

Homotopy Planning. Homotopy planning has been widely studied for motion planning of autonomous systems. To distinguish among homotopy classes of trajectories, [30] uses the relative lateral position (i.e., left or right) between two vehicles, [31], [32] partition the free space into sub-regions, [33], [34] construct homotopy-invariant words, while [35], [36] use a magnetic-field inspired approach. All these approaches require the start and end points of all candidate trajectories to coincide for homotopy classes to be well-defined—there exists no concept in the literature on homotopy that accommodates distinguishing trajectories that do not share the same end point. In this paper, we generalize homotopy to rigorously develop the notion of free-end homotopy which provides the same benefits as homotopy to motion planning, but for trajectories that *do not* share the same end point.

III. Free-end Homotopy

In this section, we will first provide the requisite background on homotopy and then extend it to develop the novel concept of free-end homotopy. Finally, we will discuss the application of free-end homotopy towards expediting our joint planner.

A. Background: Introduction to Homotopy

Homotopy helps identify trajectories that can be continuously deformed from one to another without colliding with an obstacle. More formally, homotopy is defined as follows:

Definition 1 (Homotopy [37]). *Let $\mathbf{x}_1:\mathbb{R}\to\mathcal{X}$ and $\mathbf{x}_2:\mathbb{R}\to\mathcal{X}$ be two continuous trajectories that share the same start and end points. Then, a continuous mapping $f:[0,1]\times\mathbb{R}\to\mathcal{X}$ is called a homotopy if it satisfies:*

- $f(0,\cdot)=\mathbf{x}_1(\cdot)$,
- $f(1,\cdot)=\mathbf{x}_2(\cdot)$.

If a homotopy f exists between $\mathbf{x}_1$ and $\mathbf{x}_2$, then the two trajectories are said to be homotopic.

It follows from Defintion 1 that if a continuous optimizer is initialized with a trajectory in a homotopy class, the optimizer will not explore trajectories in other homotopy classes. Therefore, homotopy classes serve as an effective tool to choose a minimal set of trajectory initializations of maximal diversity for efficient exploration of the planner's solution space. There are various approaches for verifying if two trajectories are homotopic, such as [30], [31], [35]. In this paper, we will use the magnetic-field homotopy introduced in [35], which is based on Ampere's law: $\oint_C \mathbf{B}\cdot d\mathbf{l}=\mu_0 I_{\text{enc}}$, which states that the line integral of magnetic field around a closed curve is equal to the product of the magnetic constant μ_0 and the current enclosed I_{enc}. Ampere's law establishes an equivalence condition among all closed curves that enclose the same current. Applying this to homotopy classes in motion planning, [35] lets obstacles carry current and calculates the Ampere circuit integral along the robot's trajectory. Trajectories that share the same magnetic-field integral with respect to all obstacles are homotopic.

In 2D space, all obstacles can be viewed as having genus (number of holes) 0 and the imaginary current can be set perpendicular to the X-Y plane crossing the center of the obstacle. Using the Biot-Savart law, it can be shown that the path integral along a curve around a point is proportional to the angular distance w.r.t. the point from the start to the end point of the curve. The angular distance is directional; is is negative / positive when the curve circles p counter-clockwise (CCW)/ clockwise (CW). With every CCW/CW circle, the angular distance increases / decreases by 2π, respectively. The angular distance provides two major benefits: (i) it is easy to compute and enforce as a constraint, and (ii) it can be easily extended to moving obstacles, as formalized next: Let $\mathbf{x}$ be the trajectory of the ego and $\mathbf{x}^\text{o}$ the trajectory of an obstacle. To calculate the angular distance $\Delta\theta$ that the ego sweeps w.r.t. an obstacle, we discretize the (X,Y) coordinates of the curves $\mathbf{x}$ and $\mathbf{x}^\text{o}$ into waypoints $\{(X_i,Y_i)\}_{i=1}^N$ and $\{(X_i^\text{o},Y_i^\text{o})\}_{i=1}^N$, respectively. Then

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2024, Yokohama, Japan. Cite as RA-L paper.

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2024, Yokohama, Japan. Cite as RA-L paper.

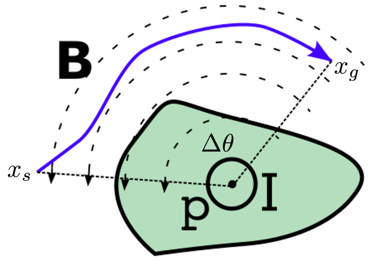

Fig. 1: Magnetic path integral in 2D. Obstacle is marked in green, imaginary current I goes through the obstacle's center p and generates a magnetic field $\mathbf{B}$, visualized with the dashed lines.

$\Delta\theta$ is computed as follows:

$$\Delta\theta(\mathbf{x},\mathbf{x}^{\mathrm{o}}) := \sum_{i=1}^{N-1} \arctan\frac{Y_{i+1}-Y_{i+1}^{\mathrm{o}}}{X_{i+1}-X_{i+1}^{\mathrm{o}}} - \arctan\frac{Y_i-Y_i^{\mathrm{o}}}{X_i-X_i^{\mathrm{o}}}. \quad (1)$$

In fact, similar definitions have been used in multimodal trajectory prediction [38] and robot navigation [39].

B. Free-end Homotopy

The motion planning problem for AV has a fixed starting point, but may not have a fixed end point. Homotopy, described in Definition 1, is not well-defined for two curves with different ending points, prohibiting the use of homotopy theory in this class of motion planning problems. To resolve this issue, we introduce *free-end homotopy*, an extension of homotopy, for trajectories that share the same initial point but may not share the same end point. The overarching objective is to develop an equivalence class of trajectories, which we call free-end homotopy classes, whose members execute the same relative motion with respect to other agents (e.g., overtake from left of agent 1 and stay behind agent 2) while being continuously transformable to any other member within the class. Free-end homotopy classes facilitate efficient planning by allowing us to downsample motion plan candidates to only those that belong to different free-end homotopy classes, i.e., ones with different relative motions with respect to obstacles.

Let $\mathbf{x}$ be the trajectory of the ego and $\mathbf{x}^{\mathrm{o}}$ be the trajectory of a particular obstacle. We begin by defining the *mode* $m : (\mathbf{x},\mathbf{x}^{\mathrm{o}}) \mapsto m(\mathbf{x},\mathbf{x}^{\mathrm{o}}) \in \mathbb{Z}$ of a trajectory with respect to a particular obstacle using the angular distances $\Delta\theta$:

$$m(\mathbf{x},\mathbf{x}^{\mathrm{o}}) := \begin{cases} -(k+1), & -(\hat{\theta}+k\pi) \le \Delta\theta(\mathbf{x},\mathbf{x}^{\mathrm{o}}) < -(\hat{\theta}+(k+1)\pi) \\ 0, & -\hat{\theta} \le \Delta\theta(\mathbf{x},\mathbf{x}^{\mathrm{o}}) < \hat{\theta} \\ k+1, & \hat{\theta}+k\pi \le \Delta\theta(\mathbf{x},\mathbf{x}^{\mathrm{o}}) < \hat{\theta}+(k+1)\pi \end{cases} \quad (2)$$

where $\hat{\theta} \in (0,\pi/2]$ determines the limits for mode 0. The positive modes result in counter-clockwise rotation of the ego relative to the obstacle; zero mode results in the ego staying either ahead or behind the obstacle, i.e., relatively stationary (S); and negative modes result in clockwise rotation of the ego relative to the obstacle. In practice, it is unusual to observe modes beyond $\{-1,0,1\}$; for the sake of convenience in exposition, we refer to these three classes as CW, S, and CCW, respectively. These three common modes are illustrated in Fig. 2.

If there are M obstacles in the scene, then the *mode vector* h for an ego trajectory $\mathbf{x}$ is defined as the cartesian product of the modes (2) with respect to each obstacle in the scene, i.e., $h(\mathbf{x},\{\mathbf{x}_i^{\mathrm{o}}\}_{i=1}^{M}) := (m(\mathbf{x},\mathbf{x}_1^{\mathrm{o}}),\cdots,m(\mathbf{x},\mathbf{x}_M^{\mathrm{o}}))$; Fig. 3 illustrates h with an example scene with two cars near the ego vehicle and three example trajectories. With this, we are now ready to define free-end homotopy.

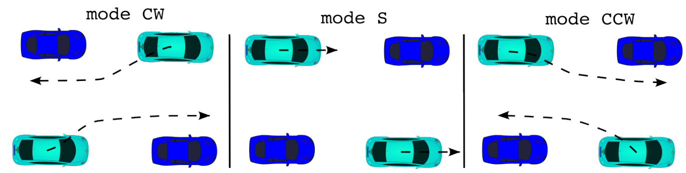

Fig. 2: Three homotopy classes: CW, S, and CCW

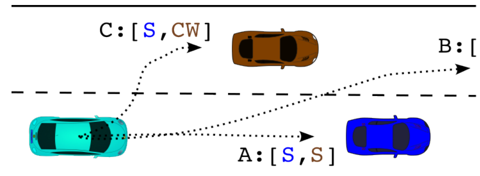

Fig. 3: Homotopy classes for two nearby objects where trajectory A is categorized as S for both objects; trajectory B is categorized as CW for the blue car and CCW for the brown car; and trajectory C is categorized as S for the blue car and CW for the brown car.

Definition 2 (Free-end Homotopy)**.** *Let* $\mathbf{x}_1 : \mathbb{R} \to \mathcal{X}$ *and* $\mathbf{x}_2 : \mathbb{R} \to \mathcal{X}$ *be two continuous trajectories that share the same start point, but do not necessarily share the same end point. Then, a continuous mapping* $f : [0,1]\times\mathbb{R} \to \mathcal{X}$ *is called a free-end homotopy if it satisfies the following criteria:*

- $f(0,\cdot) = \mathbf{x}_1(\cdot)$,
- $f(1,\cdot) = \mathbf{x}_2(\cdot)$, *and*
- *for all* $\lambda \in [0,1]$, *the mode vector* h_λ *for* $f(\lambda,\cdot)$ *are constant.*

If a free-end homotopy f *exists between* $\mathbf{x}_1$ *and* $\mathbf{x}_2$*, then the two trajectories are said to be free-end homotopic.*

Free-end homotopy is a generalization of the notion of homotopy. We make this clear in the next lemma which shows that all homotopic trajectories (with the same start and end points), are also free-end homotopic.

Lemma 1 (Free-end homotopy is a generalization of homotopy)**.** *Continuous trajectories with the same start and end points are homotopic,* if, and only if, *they are also free-end homotopic. Furthermore, the homotopy is also a free-end homotopy and vice-versa.*

Proof: Proof included in the full version [40]. ■

In the next lemma we show that the free-end homotopy relation defined above is, in fact, an equivalence relation.

Lemma 2 (Free-end homotopy is an equivalence relation)**.** *Free-end homotopy, as presented in Definition 1, is an equivalence relation.*

Proof: See full version [40] for proof. ■

This result ensures that all trajectories that are free-end homotopic can be continuously transformed from one to another while retaining the same mode vector. Hence, we can limit our planning to just one candidate per free-end homotopy class. However, it remains unresolved whether all trajectories with the same mode vector belong to the same free-end homotopy

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2024, Yokohama, Japan. Cite as RA-L paper.

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2024, Yokohama, Japan. Cite as RA-L paper.

class. Indeed, in the next theorem we will show that only one free-end homotopy class corresponds to one mode vector. This ensures that if we find a continuous trajectory with a particular mode vector, we can continuously transform it to *any* other trajectory with the same mode vector, since they all belong to the same free-end homotopy class. This facilitates faster planning by letting us run a continuous optimizer (as discussed in Section IV) on only one trajectory per mode vector.

Theorem 1 (Uniqueness of mode vector in a free-end homotopy class)**.** *Continuous trajectories with the same mode vector are free-end homotopic.*

Proof: Proof included in full version [40]. ■

C. Applying free-end homotopy classes in motion planning

The number of free-end homotopy classes grows exponentially with the number of objects in the scene and many of these classes are impractical; e.g., in Fig. 3, CCW relative to the blue vehicle will result in an off-road event. Therefore, to narrow down to a promising set of free-end homotopy classes, we use a trajectory sampler to generate N trajectory samples for the ego vehicle. Then we invoke a trajectory predictor to provide scene-centric trajectory predictions for all M objects in the scene. Together there are $N \times M$ free-end homotopy class candidates that are expressed as mode vectors, as described in Section III-B. Among these $N \times M$ mode vectors, many share the same mode vectors, that is, are free-end homotopic to others. Leveraging Theorem 1, we downsample to K trajectories from $N \times M$ by only retaining one trajectory per free-end homotopy class: the trajectory with the highest reward among all trajectories sharing the same mode vector. The reward function can be any scalar-valued function that measures the performance of the ego's trajectory amidst the objects' predictions. These K trajectories are used to initialize the gradient-based motion planner in two ways: (i) the nonlinear planning problem is linearized around the ego and the objects' trajectories to create an efficiently-solvable sequential quadratic program (SQP), and (ii) the free-end homotopy class of the trajectory is enforced as a constraint. The forthcoming section will discuss this in more detail.

IV. Interactive Joint Planning (IJP)

As discussed in the introduction, modern trajectory planners for autonomous vehicles rely on predictions for nearby agents, and ego-conditioning has been shown to improve the planning performance yet is expensive to run. The proposed IJP does not require ego-conditioned predictions, but replaces them with a joint optimization. Specifically, we invoke a prediction module to forecast the non-ego-conditioned future trajectories of the surrounding agents and pass them to the MPC planner. Intuitively, this prior supplies the optimizer with the non-ego agents' intent. The MPC planner then plans for both the ego vehicle and the surrounding agents to minimize the cost function, which we shall discuss in detail later, while enforcing collision avoidance constraint. In reality, the AV can only control its own motion, thus assuming control over surrounding agents without any limitation is obviously naive.

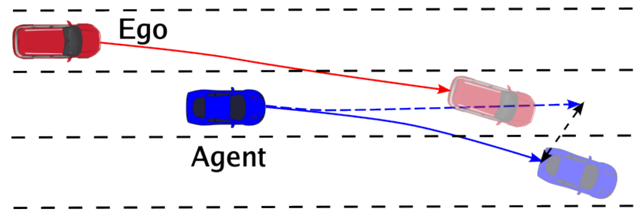

Fig. 4: Joint optimization as ego-conditioned prediction: solid trajectories: solution of the joint optimization; dashed line unconditioned predicted trajectory of the blue agent.

To remedy this, the cost contains two terms, a term that penalizes nearby agents' deviation from the predicted trajectories, and a term that penalizes their acceleration and jerk. These two terms are interpreted as the price for the ego to force nearby agents away from their nominal paths. The resulting "planned" trajectories of nearby agents can be viewed as the ego-conditioned prediction that roughly centers around the unconditioned trajectory prediction. The joint optimization as an ego-conditioned prediction is illustrated in Fig. 7 where the joint optimization chooses to let the ego (red) change lanes, and in response, the blue agent deviates slightly from the unconditioned prediction (dashed blue) to avoid a collision with the ego. Since the prediction model is only called once without ego-conditioning, the inference time decreases significantly. Moreover, the joint optimization provides finer granularity of behaviors compared to running ego-conditioned prediction on ego trajectory samples.

Next, we break down the key components of the joint MPC.

A. Dynamic models

We use a Dubin's car model for all vehicles and cyclists in the scene (including the ego vehicle).

$$x = \begin{bmatrix} X \\ Y \\ v \\ \psi \end{bmatrix}, u = \begin{bmatrix} \dot{v} \\ \dot{\psi} \end{bmatrix}, x^+ = \begin{bmatrix} X + v\cos(\psi)\Delta t \\ Y + v\sin(\psi)\Delta t \\ v + \dot{v}\Delta t \\ \psi + \dot{\psi}\Delta t \end{bmatrix}. \tag{3}$$

where X, Y are the longitudinal and lateral coordinates, v and $\dot{v}$ are the longitudinal velocity and acceleration, ψ and $\dot{\psi}$ are the heading angle and yaw rate. The pedestrians follow a double integrator model with the following dynamics:

$$x = \begin{bmatrix} X \\ Y \\ v_x \\ v_y \end{bmatrix}, u = \begin{bmatrix} \dot{v}_x \\ \dot{v}_y \end{bmatrix}, x^+ = \begin{bmatrix} X + v_x\Delta t \\ Y + v_y\Delta t \\ v_x + \dot{v}_x\Delta t \\ v_y + \dot{v}_y\Delta t \end{bmatrix}.$$

These dynamic models are linearized around an initial guess of x, u generated by a trajectory sampler as mentioned in Section III, which satisfies the nonlinear dynamic equations, and the linearized dynamic model takes the form $x^+ = Ax + Bu + C$. Furthermore, we impose dynamic constraints on the state and inputs of the agents. Specifically, for all vehicles,

$$\begin{aligned} &v \in [v^{\min}, v^{\max}], && |v\dot{\psi}| \leq a_y^{\max} \\ &|\dot{\psi}| \leq \frac{\delta^{\max}}{l}|v|, && \dot{v} \in [a_x^{\min}, a_x^{\max}] \end{aligned} \tag{4}$$

where $[v^{\min}, v^{\max}]$ is the velocity range, $a_y^{\max}$ is the maximum lateral acceleration, $a_x^{\min}$ and $a_x^{\max}$ are bounds for longitudinal

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2024, Yokohama, Japan. Cite as RA-L paper.

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2024, Yokohama, Japan. Cite as RA-L paper.

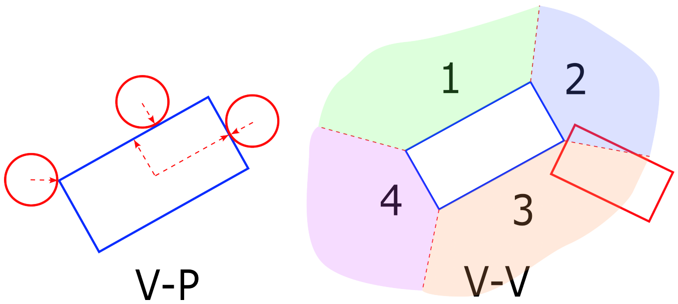

Fig. 5: Collision checks between vehicles and pedestrians (left), and two vehicles (right)

acceleration, $\delta^{\max}$ is the maximum steering angle and l is the axle distance. All pedestrians have bounds on velocity and acceleration: $||v|| \leq v_{\max}, \quad ||\dot{v}|| \leq \dot{v}_{\max}$.

B. Planning constraints

The main constraints enforced in the optimization are safety constraints, which consist of two parts, collision avoidance constraints and lane boundary constraints.

All vehicles are modeled as rectangles with varying size (including the ego) and the pedestrians are modeled as a circle with varying radius. The collision avoidance between the ego (rectangle) and pedestrians (circles) is encoded by checking the three cases where the maximum margin is achieved on the X axis, Y axis, and corners of the vehicle, as shown in Fig. 5. For two vehicles, we analytically calculate the 4 polytopic free spaces around one of the vehicles, as shown in Fig. 5, and enforce linear constraints that other vehicle's 4 corners and center point all lie in one of the free spaces. Then we do the same after reversing the role of the two vehicles.

Lane boundaries are represented as polylines (i.e., a sequence of lines connecting waypoints with headings). The lane boundary constraints are enforced by projecting the vehicle centers to the polylines and calculating the distance margins.

All of the above mentioned inequality constraints are differentiable with respect to (w.r.t.) the state of the ego vehicle and other agents on the road, hence, they are linearized and enforced as linear constraints in the MPC. To ensure feasibility, we add slack variables to collision avoidance constraints and lane boundary constraints.

In addition to the collision avoidance constraints, we also enforce the homotopy class constraint, which is computed as discussed in Section III. In simulation we observed that the MPC QP behaves similarly without the free-end homotopy constraint and initialization/linearization is sufficient to enforce the homotopy constraint.

C. Costs and MPC QP setup

Let $\mathbf{x}_\mathrm{e}$ be the future trajectory of the ego vehicle, $\mathbf{x}_\mathrm{o}$ be the future trajectory of nearby agents, and $\mathbf{x}^\mathrm{pred}$ be the unconditioned predicted trajectory of the nearby agents. The cost function, then, consists of 4 terms:

- Penalty on ego vehicle's tracking error w.r.t. the reference trajectory $\mathcal{J}_\mathrm{ref}(\mathbf{x}_\mathrm{e}, \mathbf{x}^\mathrm{ref})$
- Penalty on ego vehicle's acceleration and jerk $\mathcal{J}_\mathrm{u}(\mathbf{u}_\mathrm{e})$
- Penalty on nearby agents' deviation from their unconditioned trajectory prediction $\mathcal{J}_\mathrm{dev}(\mathbf{x}_\mathrm{o}, \mathbf{x}^\mathrm{pred})$
- Penalty on nearby agents' acceleration and jerk $\mathcal{J}_\mathrm{u}(\mathbf{u}_\mathrm{o})$

Everything put together, the MPC solves the following QP:

$$\min_{\mathbf{u}_\mathrm{e},\mathbf{u}_\mathrm{o},\mathbf{x}_\mathrm{e},\mathbf{x}_\mathrm{o}} \eta_e(\mathcal{J}_\mathrm{ref}(\mathbf{x}_\mathrm{e}, \mathbf{x}^\mathrm{ref}) + \mathcal{J}_\mathrm{u}(\mathbf{u}_\mathrm{e})) + \eta_o(\mathcal{J}_\mathrm{dev}(\mathbf{x}_\mathrm{o}, \mathbf{x}^\mathrm{pred}) + \mathcal{J}_\mathrm{u}(\mathbf{u}_\mathrm{o})) \quad (5)$$

$$\text{s.t. } \mathbf{x}_\mathrm{e}[0] = x_e^0,\ \mathbf{x}_\mathrm{o}[0] = x_o^0 \quad (6)$$

$$\forall t = 0, \ldots, T-1, i \in \{e, o_1, \ldots o_n\},$$

$$\mathbf{x}_i[t+1] = A_i[t]\mathbf{x}_i[t] + B_i[t]\mathbf{u}_i[t] + C_i[t] \quad (7)$$

$$\forall t = 1, \ldots, T, G_e^s[t]\,\mathbf{x}_\mathrm{e}[t] + G_o^s[t]\,\mathbf{x}_\mathrm{o}[t] \leq g^s[t], \quad (8)$$

where $\mathbf{x}_\mathrm{e}$ is the future state of the ego vehicle, $\mathbf{x}_{o_i}$ is agent i's future state, A, B, C are the matrices corresponding to the dynamic equality constraints, G_x^d, G_u^d, g^d are matrices corresponding to the input and state bounds, and G_e^s, G_o^s, g^s define the safety constraints, including collision avoidance, lane boundaries, and the homotopy constraint. The weighting constants η_e and η_o determine the distribution of emphasis on the ego vehicle and the agents. A large η_e leads to more selfish and intrusive behavior of the ego and a small η_e leads to more altruistic ego behavior.

D. Interactive Joint Planning

The IJP planner is summarized in Algorithm 1. The inputs are the reference trajectory for the ego vehicle given by some high-level planner, scene context $\mathbf{C}$, lane information $\mathbf{L}$, and the current state of the ego and surrounding agents.

First, IJP calls the trajectory prediction model, which is typically a neural network trained with traffic data, to generate predictions for the M surrounding agents from the scene context $\mathbf{C}$. IJP can work with any prediction model that generates dynamically feasible trajectories for the agents involved; for prediction models without a dynamic model, we perform inverse dynamics to retrieve the control inputs and re-propagate the trajectory prediction. It is preferred that the prediction is scene-centric, i.e., predicting joint trajectories for all agents involved. We use Agentformer [6] as our default predictor because it is scene-centric and is shown to work well with the downstream planner in [7].

Algorithm 1 IJP

procedure IJP($\mathbf{x}^\mathrm{ref}, \mathbf{C}, \mathbf{L}, x_e^0, x_o^0$)
  $\mathbf{x}^\mathrm{pred} \leftarrow$ TRAJ_PRED($\mathbf{C}$)
  $\{\mathbf{x}_{e,i}^\mathrm{sample}\}_{i=1}^N \leftarrow$ EGO_SAMPLING($x_e^0, \mathbf{L}$)
  $\{(\mathbf{x}_{e,k}, \mathbf{x}_{o,k}, h_k)\}_{k=1}^K \leftarrow$ HOM_SEL($\{\mathbf{x}_{e,i}^\mathrm{sample}\}_{i=1}^N, \mathbf{x}^\mathrm{pred}$)
  for $r = 1, \ldots, R$ **do**
    for $k = 1, \ldots, K$ **do**
      $\mathrm{QP}_k \leftarrow$ LINEARIZE($\mathbf{x}^\mathrm{ref}, \mathbf{x}_{e,k}, \mathbf{x}_{o,k}, h_k, x_e^0, x_o^0, \mathbf{L}$)
      $\mathbf{x}_{e,k}, \mathbf{x}_{o,k} \leftarrow$ SOLVE_QP(QP_k)
    end for
  end for
  return $\mathbf{x}_\mathrm{e}$ associated with the best homotopy class.
end procedure

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2024, Yokohama, Japan. Cite as RA-L paper.

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2024, Yokohama, Japan. Cite as RA-L paper.

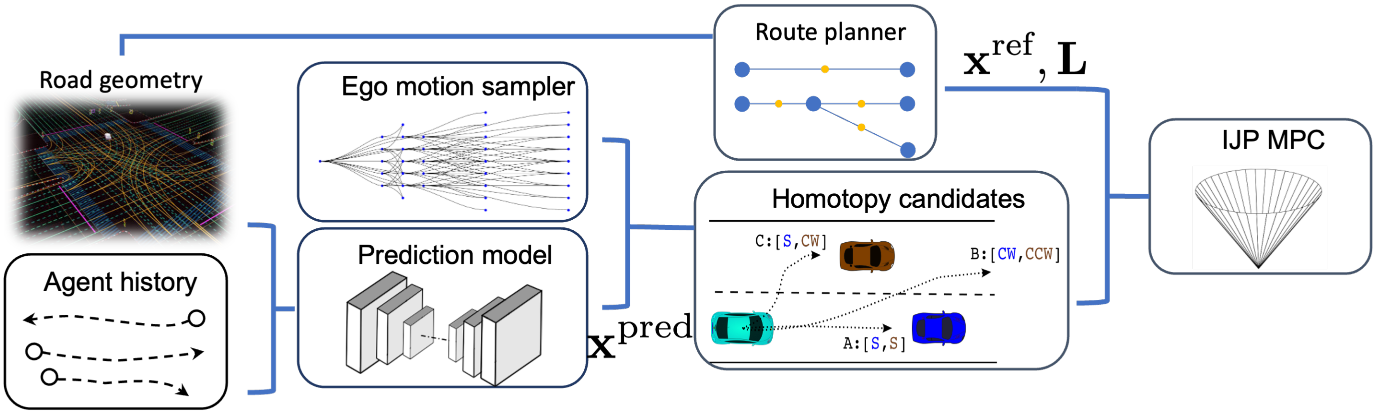

Fig. 6: Overview of IJP: the trajectory predictor takes in the lane graph and the agent history to predict the unconditioned prediction for the agents $\mathbf{x}^{\text{pred}}$; the route planner plans the desired route and generates the reference trajectory $\mathbf{x}^{\text{ref}}$ and distills the lane information (such as lane boundaries) $\mathbf{L}$; the trajectory sampler samples the ego trajectory samples, and together with $\mathbf{x}^{\text{pred}}$, the homotopy candidates are identified. Finally, IJP plans the ego motion via solving the joint model predictive control problem with SQP.

EGO_SAMPLING takes the ego state and lane information to generate ego trajectory samples with a spline sampler introduced in [7], which is then used to identify promising homotopies with the predicted trajectories of the surrounding agents in HOM_SEL. With the homotopies selected, IJP uses automatic differentiation to linearize the costs, constraints, and dynamics to formulate a quadratic program. JAX [41] is used for auto-differentiation, and thanks to its powerful parallelization functionality and Just-In-Time (JIT) compilation, the linearization can be done simultaneously for all homotopy classes. The generated QP is solved with the QP solver Forces Pro [42]. In a sequential quadratic programming (SQP) manner, the nonlinear trajectory optimization is linearized and solved as a QP for multiple rounds R, each round taking the solution from the last round as the updated linearization point. A proximal constraint is also added to limit the difference of solutions in between rounds to stabilize the SQP.

Remark 1. *When the trajectory prediction module outputs multimodal predictions of the surrounding agents, the criteria for selecting the optimal solution among the candidate homotopy classes should also take into account the likelihood of the prediction modes, however, we observed that the mode probabilities predicted by the prediction module are usually inaccurate and thus we ignore the mode probability in the final solution selection and simply choose the mode with the lowest cost. We shall investigate how to incorporate prediction likelihood in solution selection in future work.*

V. Simulation Setup and Results

In this section, through exhaustive closed-loop simulations we demonstrate that IJP significantly outperforms other baselines resulting in a safer drive without hurting progress.

A. Simulation Evaluation Setup

We conduct closed-loop simulation in nuPlan [43] to evaluate the proposed approach. The closed-loop planner consists of three modules, a trajectory predictor that generates the unconditioned trajectory prediction, a route planner that distills lane information and reference trajectory from the lane graph, and IJP that plans the trajectory, as shown in Fig. 6.

We use AgentFormer [6] as the trajectory predictor without ego-conditioning, which generates 4 samples of predicted future trajectories lasting for 3 seconds.

Route planner. The route planner takes the lane graph and the ego state as input, and performs a depth-first search to identify the optimal lane sequence. In nuplan simulation, no goal location is provided, instead the lane segments are labeled as ”on-route” or ”not on-route”. The route planner’s search criteria is to find the an on-route lane sequence (up to a certain depth) that balances (i) distance to the ego vehicle (ii) length of the lane plan and (iii) total curvature of the lane plan. With a lane sequence selected, the reference trajectory is generated by projecting the ego’s current position to the lane centerline and interpolating given the desired ego velocity.

To keep the QP complexity tractable, IJP only includes a subset of nearby agents in the joint optimization, denoted as EC agents (EC stands for ego conditioned); the rest of the agents are denoted as non-EC agents and IJP simply encodes collision avoidance constraints with their predicted trajectories. The assignment of EC and non-EC agents is based on their minimum distance to the ego vehicle along their predicted trajectories. When there are less agents than the prescribed number, the MPC QP is padded with dummy agents. To avoid frequent JIT compilation, the numbers of EC agents and non-EC agents are fixed so that the MPC QP maintains a fixed problem dimension. When the number of nearby obstacles is larger than the sum of the prescribed number of EC and non-EC agents, far-away agents are ignored

We compared the performance of IJP to two baselines: (i) non-EC MPC: IJP without joint optimization, which only plans the ego behavior and tries to avoid collision with the predicted trajectories of nearby agents, and (ii) TPP: a sampling-based planner using ego-conditioned prediction similar to TPP [7], but without multi-layer policy planning. [1]

[1] For fairness of comparison, the non-EC MPC considers a fixed number of non-EC agents, and the number is equal to the sum of EC agents and non-EC agents considered by IJP. The TPP planner considers all agents detected as the sampling-based algorithm does not require a fixed number of agents.

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2024, Yokohama, Japan. Cite as RA-L paper.

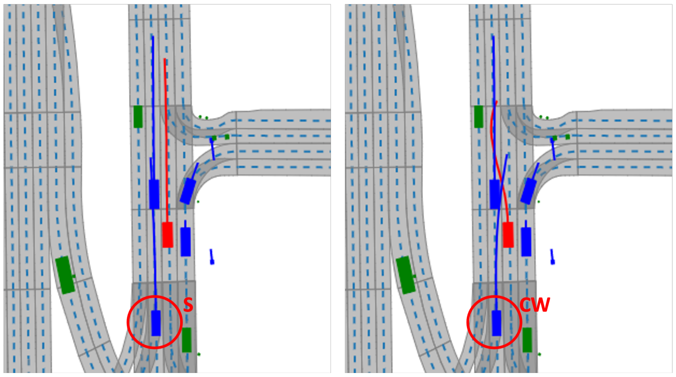

Fig. 7: Comparison of the solutions under two free-end homotopy classes: the ego vehicle is in red, the EC agents are in blue and the non-EC agents are in green. As the homotopy class w.r.t. the circled agent changes from S to CW, the ego's behavior changes from lane keeping to lane change, and the "predicted behavior" of the circled vehicle changes accordingly as a result of the joint optimization.

Planner	Drivable area compliance	Progress	No ego at fault collision	Final Score
IJP	**0.98**	0.86	**0.95**	**0.86**
TPP	0.86	0.87	0.89	0.71
non-EC MPC	0.94	**0.95**	0.66	0.62

TABLE I: Key metrics of in nuPlan simulation

B. Simulation Results

We first show an example snapshot from the nuPlan simulation of IJP in Fig. 7 to demonstrate IJP in action. The two plots in Fig. 7 are the MPC solutions under two free-end homotopy classes. The only difference between the two is the free-end homotopy class w.r.t. the circled vehicle: S (static) in the left case and CW (clockwise) in the right case. The blue curve is the solution of the EC agents' trajectories "planned" by IJP. In the right plot, as the ego (red) changes lanes, the trailing vehicle changes lane to the right to avoid collision with the ego, which is indeed similar to an ego-conditioned prediction.

Quantitatively, we ran closed-loop simulation of 50 scenes from nuPlan's Boston dataset which includes many interesting interactive scenarios with sophisticated road geometry. We compared key metrics such as collision rate and progress, all collected from the nuPlan simulator under IJP and the baselines, shown in Table I, where all metrics are from nuPlan. The simulation statistics show that IJP significantly outperformed the baselines in safety-related metrics such as collision rate and drivable area compliance, and did reasonably well in progress. In fact, upon inspection, we found that the few incidents where the ego was blamed for causing a collision were not correctly assessed. Those few accidents were caused by nearby agents not yielding when making a right turn or lane change (nuPlan label all frontal collision of the AV as at-fault despite right of way). We found no clear mistake made by IJP in the 50 scenes in the simulation. The key parameters of IJP can be found in Table III.

It is counter-intuitive that the non-EC MPC results in worse safety performance given that it fully "respects" the prediction. A likely explanation is that when near multiple agents, the prediction makes the motion planning problem infeasible (without slack), and when the prediction is of poor quality, the planner overreacts, causing the performance to deteriorate.

	prediction	Build time	Solve time
IJP	0.051s	0.150s	0.152s
non-EC MPC	0.051s	0.045s	0.007s
TPP	0.690s	-	0.006s

TABLE II: Computation time of IJP and the two baselines

Number of homotopy	Number of EC agents	Number of non-EC agents	Horizon	Time step
6	6	10	3s	0.15s

TABLE III: Key parameters of IJP

Table II shows the computation time of IJP and the baselines. Agentformer runs on an Nvidia 3090 GPU and the MPC QP runs on the CPU with Forces Pro QP solver. We separate the build and solve time of the MPC QP because the build process generates all MPC QP instances under different homotopy classes in parallel, while the solve time corresponds to solving one of the QP instances. Compared to TPP, while IJP takes longer to solve, it saves more time on the prediction phase as no EC prediction is needed. [2]

The computation time of IJP can be further improved in at least two ways: parallelizing the solving process of MPC QP under multiple homotopy classes, and utilizing the sparsity pattern in the QP. These improvements hold promise of achieving real-time planning with IJP.

VI. Conclusion and Discussion

We presented a new planning method, IJP, that can reason about the impact of the ego's actions on the behavior of other traffic agents by combining gradient-based joint planning for all agents with modern deep learning-based predictors. The key idea behind IJP is viewing joint optimization solutions as ego-conditioned predictions and penalizing deviations from the unconditional predictions to regularize the EC predictions. It should be pointed out that the EC predictions currently lack statistical grounding, i.e., no supervision is added in the prediction model training process to force the result of the subsequent joint optimization to match the ego-conditioned ground truth. The main missing piece is counterfactual traffic data, which is not available in general. The behavior of IJP largely depends on hyper-parameters such as η_e and η_o, and currently they are hand-tuned. Nonetheless, the closed-loop performance of IJP turned out significantly better than the baselines, and we believe the main reasons are the free-end homotopy that diversifies the search space, and the fine granular solution achieved by the joint optimization.

For future work, we will focus on providing a solid probabilistic grounding for the joint optimization solution viewed as ego-conditioned prediction by differentiating through the optimization and training the prediction-planning modules end-to-end.

References

[1] M. Liebner, M. Baumann, F. Klanner, and C. Stiller, "Driver intent inference at urban intersections using the intelligent driver model," in *2012 IEEE intelligent vehicles symposium*. IEEE, 2012, pp. 1162–1167.

[2] TPP without ego-conditioning has the same prediction time as IJP, but the final score dropped to 0.64 due to more safety violations.

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2024, Yokohama, Japan. Cite as RA-L paper.

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2024, Yokohama, Japan. Cite as RA-L paper.

[2] M. Bansal, A. Krizhevsky, and A. Ogale, "ChauffeurNet: Learning to drive by imitating the best and synthesizing the worst," 2019.
[3] A. Cui, S. Casas, A. Sadat, R. Liao, and R. Urtasun, "Lookout: Diverse multi-future prediction and planning for self-driving," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 16 107–16 116.
[4] T. Salzmann, B. Ivanovic, P. Chakravarty, and M. Pavone, "Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data," in *European Conference on Computer Vision*. Springer, 2020, pp. 683–700.
[5] Y. Chen, B. Ivanovic, and M. Pavone, "Scept: Scene-consistent, policy-based trajectory predictions for planning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 17 103–17 112.
[6] Y. Yuan, X. Weng, Y. Ou, and K. M. Kitani, "AgentFormer: Agent-aware transformers for socio-temporal multi-agent forecasting," in *IEEE Int. Conf. on Computer Vision*, 2021, pp. 9813–9823.
[7] Y. Chen, P. Karkus, B. Ivanovic, X. Weng, and M. Pavone, "Tree-structured policy planning with learned behavior models," *arXiv:1611.05763*, 2016.
[8] U. Rosolia, S. De Bruyne, and A. G. Alleyne, "Autonomous vehicle control: A nonconvex approach for obstacle avoidance," *IEEE Transactions on Control Systems Technology*, vol. 25, no. 2, pp. 469–484, 2016.
[9] A. Censi, D. Calisi, A. De Luca, and G. Oriolo, "A bayesian framework for optimal motion planning with uncertainty," in *2008 IEEE International Conference on Robotics and Automation*. IEEE, 2008, pp. 1798–1805.
[10] Y. Chen, U. Rosolia, W. Ubellacker, N. Csomay-Shanklin, and A. Ames, "Interactive multi-modal motion planning with branch model predictive control," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 5365–5372, 2022.
[11] T. Batz, K. Watson, and J. Beyerer, "Recognition of dangerous situations within a cooperative group of vehicles," in *2009 IEEE Intelligent Vehicles Symposium*. IEEE, 2009, pp. 907–912.
[12] P. Trautman and A. Krause, "Unfreezing the robot: Navigation in dense, interacting crowds," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2010, pp. 797–803.
[13] P. Trautman, J. Ma, R. M. Murray, and A. Krause, "Robot navigation in dense human crowds: the case for cooperation," in *2013 IEEE international conference on robotics and automation*. IEEE, 2013, pp. 2153–2160.
[14] C. Chen, Y. Liu, S. Kreiss, and A. Alahi, "Crowd-robot interaction: Crowd-aware robot navigation with attention-based deep reinforcement learning," in *2019 international conference on robotics and automation (ICRA)*. IEEE, 2019, pp. 6015–6022.
[15] B. D. Ziebart, A. Maas, J. A. Bagnell, and A. K. Dey, "Maximum entropy inverse reinforcement learning," in *Proc. AAAI Conf. on Artificial Intelligence*, 2008.
[16] H. Kretzschmar, M. Spies, C. Sprunk, and W. Burgard, "Socially compliant mobile robot navigation via inverse reinforcement learning," *Int. Journal of Robotics Research*, vol. 35, no. 11, pp. 1289–1307, 2016.
[17] M. Kuderer, S. Gulati, and W. Burgard, "Learning driving styles for autonomous vehicles from demonstration," in *Proc. IEEE Conf. on Robotics and Automation*, 2015.
[18] M. Pfeiffer, U. Schwesinger, H. Sommer, E. Galceran, and R. Siegwart, "Predicting actions to act predictably: Cooperative partial motion planning with maximum entropy models," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2016, pp. 2096–2101.
[19] D. Sadigh, S. Sastry, S. A. Seshia, and A. D. Dragan, "Planning for autonomous cars that leverage effects on human actions," in *Robotics: Science and Systems*, 2016.
[20] M. Bahram, A. Lawitzky, J. Friedrichs, M. Aeberhard, and D. Wollherr, "A game-theoretic approach to replanning-aware interactive scene prediction and planning," *IEEE Transactions on Vehicular Technology*, vol. 65, no. 6, pp. 3981–3992, 2015.
[21] N. Li, D. W. Oyler, M. Zhang, Y. Yildiz, I. Kolmanovsky, and A. R. Girard, "Game theoretic modeling of driver and vehicle interactions for verification and validation of autonomous vehicle control systems," *IEEE Transactions on control systems technology*, vol. 26, no. 5, pp. 1782–1797, 2017.
[22] J. Wei, J. M. Dolan, and B. Litkouhi, "Autonomous vehicle social behavior for highway entrance ramp management," in *IEEE Intelligent Vehicles Symposium*, 2013.
[23] C. Hubmann, M. Becker, D. Althoff, D. Lenz, and C. Stiller, "Decision making for autonomous driving considering interaction and uncertain prediction of surrounding vehicles," in *IEEE Intelligent Vehicles Symposium*, 2017.
[24] S. Hoermann, D. Stumper, and K. Dietmayer, "Probabilistic long-term prediction for autonomous vehicles," in *2017 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2017, pp. 237–243.
[25] C. Dong, J. M. Dolan, and B. Litkouhi, "Intention estimation for ramp merging control in autonomous driving," in *2017 IEEE intelligent vehicles symposium (IV)*. IEEE, 2017, pp. 1584–1589.
[26] M. Düring and P. Pascheka, "Cooperative decentralized decision making for conflict resolution among autonomous agents," in *2014 IEEE International Symposium on Innovations in Intelligent Systems and Applications (INISTA) Proceedings*. IEEE, 2014, pp. 154–161.
[27] S. Casas, C. Gulino, S. Suo, K. Luo, R. Liao, and R. Urtasun, "Implicit latent variable model for scene-consistent motion forecasting," in *European Conf. on Computer Vision*, 2020.
[28] N. Rhinehart, R. McAllister, K. Kitani, and S. Levine, "PRECOG: Prediction conditioned on goals in visual multi-agent settings," in *IEEE Int. Conf. on Computer Vision*, 2019.
[29] B. Ivanovic, A. Elhafsi, G. Rosman, A. Gaidon, and M. Pavone, "Mats: An interpretable trajectory forecasting representation for planning and control," *arXiv preprint arXiv:2009.07517*, 2020.
[30] B. Yi, P. Bender, F. Bonarens, and C. Stiller, "Model predictive trajectory planning for automated driving," *IEEE Transactions on Intelligent Vehicles*, vol. 4, no. 1, pp. 24–38, 2018.
[31] S. J. Anderson, S. B. Karumanchi, and K. Iagnemma, "Constraint-based planning and control for safe, semi-autonomous operation of vehicles," in *2012 IEEE intelligent vehicles symposium*. IEEE, 2012, pp. 383–388.
[32] D. Yi, M. A. Goodrich, and K. D. Seppi, "Homotopy-aware rrt: Toward human-robot topological path-planning," in *2016 11th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*. IEEE, 2016, pp. 279–286.
[33] S. Bhattacharya and R. Ghrist, "Path homotopy invariants and their application to optimal trajectory planning," *arXiv preprint arXiv:1710.02871*, 2017.
[34] S. Söntges and M. Althoff, "Computing possible driving corridors for automated vehicles," in *2017 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2017, pp. 160–166.
[35] S. Bhattacharya, M. Likhachev, and V. Kumar, "Topological constraints in search-based robot path planning," *Autonomous Robots*, vol. 33, no. 3, pp. 273–290, 2012.
[36] S. Bhattacharya, "Search-based path planning with homotopy class constraints," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 24, no. 1, 2010, pp. 1230–1237.
[37] A. Hatcher, *Algebraic Topology*. Cambridge Press, 2005.
[38] J. Roh, C. Mavrogiannis, R. Madan, D. Fox, and S. Srinivasa, "Multi-modal trajectory prediction via topological invariance for navigation at uncontrolled intersections," in *Conference on Robot Learning*. PMLR, 2021, pp. 2216–2227.
[39] C. Mavrogiannis, K. Balasubramanian, S. Poddar, A. Gandra, and S. S. Srinivasa, "Winding through: Crowd navigation via topological invariance," *IEEE Robotics and Automation Letters*, vol. 8, no. 1, pp. 121–128, 2022.
[40] Y. Chen, S. Veer, P. Karkus, and M. Pavone, "Interactive joint planning for autonomous vehicles," *arXiv preprint arXiv:2310.18301*, 2023.
[41] J. Bradbury, R. Frostig, P. Hawkins, M. J. Johnson, C. Leary, D. Maclaurin, G. Necula, A. Paszke, J. VanderPlas, S. Wanderman-Milne, and Q. Zhang, "JAX: composable transformations of Python+NumPy programs," 2018. [Online]. Available: http://github.com/google/jax
[42] A. Zanelli, A. Domahidi, J. Jerez, and M. Morari, "Forces nlp: an efficient implementation of interior-point... methods for multistage nonlinear nonconvex programs," *International Journal of Control*, pp. 1–17, 2017.
[43] H. Caesar, J. Kabzan, K. Tan, W. Fong, E. Wolff, A. Lang, L. Fletcher, O. Beijbom, and S. Omari, "nuplan: A closed-loop ml-based planning benchmark for autonomous vehicles," *arXiv preprint arXiv:2106.11810*, 2021.

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2024, Yokohama, Japan. Cite as RA-L paper.