Predictive Reachability for Safe Navigation among Human Agents

Luke Shimanuki

***Abstract*—Safe control in the presence of other human agents is a challenging but critical requirement for autonomous vehicles. On one hand, reachability-based control schemes provide formally verifiable safety certificates, but force overly-conservative robot behavior in order to avoid all possible behaviors of the other agents. On the other, approaches that attempt to predict the other agents' behaviors then plan around them grant the robot greater flexibility, but the safety of such policies is difficult to quantifiably measure outside of real-world testing. We present a framework integrating these two approaches: we leverage reachability analysis to guarantee safety, but when computing the reachable sets we restrict the behaviors of the other agents to a predicted set of policies. Because the resulting controller is safe as long as the other agents' behaviors are represented in the predictions, the fraction of situations in which it is unsafe can be evaluated (or optimized in a learning setting) *offline* on a dataset of observed trajectories. We implement and evaluate a couple of simple prediction models under this framework on the inD driving dataset [6], demonstrating significantly better safety bounds at long time horizons than state-of-the-art prediction and planning systems.**

I. Introduction

The goal of autonomous navigation is generally to reach some desired location without colliding with any obstacles. Unfortunately, this gets complicated when the obstacles can move, especially if such movements are based on partially observable processes (e.g. the internal state of the mental processes of the driver of another vehicle).

Reachability analysis has been demonstrated to provide policies that are provably safe regardless of the behavior of the other agents [2, 22, 11, 4, 20]. However, these policies must be overly conservative, and oftentimes the problem is unsatisfiable, since in crowded environments another agent can nearly always force a collision. In order to behave in a natural way, the robot must be able to reason about what the other agents are likely to actually do, rather than everything they could possibly do.

The standard approach is to create some kind of model of how the other agents move, predict where they're going to go, and then use a time-dependent motion planner to avoid where they are expected to be. Unfortunately, there has been no demonstrated way to provide sufficient safety guarantees for these methods aside from extensive real-world testing. This is because offline evaluation based on recorded behaviors does not capture how other agents react to the robot's actions, and simulators do not necessarily accurately reflect real-world human behavior.

We propose a different paradigm which can be evaluated offline and still provide safety guarantees. There are two problems that make predicting trajectories ill-formed: unobservability and interactivity. To handle interactivity, we will predict *policies* rather than fixed trajectories. To handle unobservability, we will predict *sets* of policies rather than single policies (note that we could instead predict distributions over policies, but at planning time, we'd have to draw a risk cutoff at some point anyways, so we prefer to do that at the prediction step for convenience). We know that the correct joint policy over all agents will avoid collisions, so one evaluation metric would be the number of situations in an observed training dataset where a collision would be possible given that every agent follows a policy from the predicted set. And since we need the predictions to actually match reality, the other evaluation metric would be the number of situations in the training dataset where an agent's actually observed behavior deviates from the policy set.

To illustrate with some trivial examples, consider the policy set where every agent is stationary. Then the collision rate would be zero, since stationary agents do not collide. But the deviation would be high, because nearly all the time, there will be a non-stationary agent in the data. In the other extreme, consider the policy set where every agent is allowed any possible movement. Then the collision rate would be high, because a collision would always be achievable by some assignment of policies. But the deviation would be low, because anything that was actually observed fits within the policy set. An ideal policy set is then one that is narrow enough to prevent collisions but broad enough to contain every actual observation. In Section III we prove that a robot following a policy from such a set will collide at most as frequently as the policy set either allows a collision or fails to capture an actual human behavior.

To motivate this approach, we construct a simple policy set in a manner similar to reachability analysis that can be proven to never allow collisions, and empirically captures nearly all behaviors observed in the inD dataset [6]. A controller can then ensure safety by running a conventional controller optimizing some objective, checking whether the chosen action deviates from the policy set, and if it does deviate, falling back to a safe stopping policy. The stopping policy is guaranteed to be safe because the policy set is constructed to ensure that any action can be followed by the stopping policy without colliding. The controller will only need to resort to the fallback stopping policy as often as the policy set fails to contain the observed behavior, so the low empirical deviation from observed behaviors demonstrates that this policy set allows for non-conservative robot behavior. The inD dataset [6] was

chosen due to its availability, size, complexity of traffic (large unsignalized intersections), and precision of ground truth labels. We emphasize that the fraction of behaviors that are captured by this policy set is easily evaluated offline, and translates to online safety guarantees as long as the recorded dataset is representative of the real world.

II. Background

A. Motion Forecasting

A key prerequisite for safe planning is having some notion of the other agents' future behavior. By far the most common form this takes is predicting trajectories or distributions over trajectories for each agent, which are then evaluated by measuring how closely the predicted trajectories match those observed in real-world recordings using some distance metric. There is a wide range of research in this area, but most recent state-of-the-art results employ large neural networks [23, 17], many explicitly modeling intermediate variables such as intent [9, 37] and interactions [37, 10]. These are trained via standard learning algorithms from recorded trajectories. However, as the prediction horizon increases, the error increases dramatically. This is unavoidable, as it reflects the actual variance in the real world, since the effect of multimodal outcomes becomes more prominent at longer horizons. Hence, these tools often only provide sufficient safety guarantees at low horizons (e.g. often evaluated up to only 2-3 seconds), before the multimodal complexities take effect.

Our work gives up on attempting to predict specific trajectories and instead predicts sets of policies, which can capture this variance, as well as the interaction between different agents. This allows for results that scale well to much longer horizons.

B. Planning under Uncertainty

Approaches for planning in the presence of other agents typically falls into one of three camps. The first characterizes the domain as a single agent avoiding collision with other obstacles with motion governed by some partially observable process. A common formalism for describing such a domain is a partially observable Markov decision process (POMDP). There is a wide body of research on solving these problems to varying degrees of generality [5, 7, 26, 8, 25, 14, 35] Unfortunately, these approaches tend to scale poorly with dimension or complexity of the interactions between agents, as the problem is in general PSPACE-complete.

The second treats the problem as a multi-agent game theory problem, taking the assumption that each agent is maximizing some known utility function and then either estimating the optimal or equilibrium joint policy [28, 13, 3, 19] or ordering the agents using heuristics and then allowing each to select its optimal move given the moves of the prior agents [29, 21].

The third and final category is learning from real-world recordings. Typically these approaches do not explicitly model the unobservable structure of the problem, instead relying on the learning process to infer these kinds of reasoning from the data. The different methods for this kind of approach are highly diverse and take direction from many different subfields of machine learning, but some common patterns include inverse reinforcement learning [16, 1, 38] and end-to-end planning methods [34, 33, 36, 27].

Our work draws from all three of these paradigms, allowing each agent to make its decisions by using heuristics on a highly structured POMDP, inferring common motion patterns from real-world data, and then providing safety guarantees based on a game theoretic analysis.

C. Formally Verified Controllers

Analytical proofs of safety following synthesis [24, 18] or formal verification [31, 12] techniques have met with some success, but unfortunately they require known models of the world, and so do not scale up to more complex systems involving interactions between agents. In the absence of knowing exactly how the world will evolve, we are left with considering all possible ways it could potentially evolve. Reachability-based approaches, including open-loop planning around possible future states of other agents [2, 22] and closed-loop control that always ensures that there exists a policy that will avoid collision regardless of the behavior of other agents [11, 4, 20] do provide safety guarantees in these domains. However, these result in overly-conservative behavior in crowded settings, since the other agents can nearly always force a collision regardless of evasive robot behavior.

Our work follows a similar path as the closed-loop control approaches, but it instead restricts the behavior it expects from the other agents to allow for less conservative robot behavior. This is safe under the assumption that there exists some set of social rules limiting driving behavior, and the validity of such a set of rules can be empirically verified offline. However, this does come at the cost of not guaranteeing safety in the rare instance of another agent intentionally trying to force a collision.

D. Evaluating Self-driving Systems

It remains an open question how best to evaluate a self-driving system to provide formal safety guarantees, as it has proven to be an extremely difficult objective to measure [30].

The alternative to formal safety certificates is empirically evaluating safety on a large number of potential driving situations. There are three common methods of generating the obstacle motions for other agents for such an evaluation. The first, and most ideal, is real-world testing [32]. With sufficient on-road time, this provides a good picture of how safe the planning system is. However, it is slow and costly, in terms of equipment, manpower, and risk of something going wrong. Evaluating via testing in the wild is not an option for most labs and even many companies researching this problem, and it goes without saying that it would be especially impractical for any learning system to directly optimize over this metric. If we can't test in the real world, another option is simulation [5, 14, 35]. However, this is inherently circular because the simulated motions of other agents must be derived from some computational process that does not necessarily reflect reality. A planner that achieves safety in this simulated domain has

only demonstrated that it is able to predict and respond to the process governing the simulation, rather than real-world behaviors, and so we must then verify the accuracy of the simulator itself, which brings us back to the original problem.

The last option is to record real driving behavior, but then allow the planner to change what the robot does offline [34, 15]. While this seems to be the best option available to the average researcher, it is limited by its inability to capture how other agents will respond to different robot actions. Houston et al. [15] find that 85% of situations requiring intervention (e.g. collision) under this evaluation paradigm are at least in-part due to the non-reactivity of the other agent motions. It is impossible to compute in how many of those a collision would have actually occured without being able to predict the agent motion in the counterfactual case, that is, what the other agent would have done had the robot done something else. Hence, up until now to the best of our knowledge, real-world testing has been the only way to provide guaranteed bounds for the performance of a self-driving system without excessively conservative behavior, with the offline options often used in its place for cost reasons, but failing to provide the same guarantees.

Our work provides an analytical proof of safety for a particular model of the world which can then be compared against real-world data offline to provide quantitative safety guarantees for the domain in which the data was generated.

III. Problem Definition

We start by defining a highly structured partially observable deterministic decision process (deterministic POMDP). Let O denote the observable state space for a single agent, including the pose, momentum, shape, and anything else that defines how it can move and interact with other agents, and let Π denote the unobservable state space including information such as where it wants to go or how it plans to get there. Suppose that each agent's behavior is fully described by a policy $\pi : H \to A$, where H is the joint observable state space history $O^{n\tau}$ (where n is the number of agents and τ is the number of observed timesteps), and A is the action space for a single agent, typically actuator settings for the duration of a timestep. Then the unobservable state space of an agent captures some space of such policies, that is $\Pi \subseteq H \to A$. In order to retain the Markov property, we restrict τ to be upper bounded by some constant (the results in this paper require just 3), and so the full state space of the POMDP is $(O^\tau \times \Pi)^n$. Let us further suppose that the transition model for a single agent $M : O \times A \to O$ is fully known and independent of the state and actions of other agents, and that there is a unique action that leads from any given state o_1 to o_2. For situations where it is not unambiguous, we specify $h^{(i)}$ as the history h but with agent i treated as the ego-agent, that is, the one that is being controlled.

We can now informally define a navigation problem.

Problem 1 (Navigation Problem)**.** *Given the current history $h \in H$ and a "typical" goal state $g \in O$ (describing just the robot), find a policy $\pi \in \Pi$ minimizing the number of fixed-time situations in which the goal cannot be reached or the robot collides with another agent, assuming each other agent is following a "typical" policy.*

Ideally, a solution would be evaluated based on some combination of how frequently it reaches the goal and how frequently it avoids collision. Unfortunately, neither of these can be measured for arbitrary policies offline, because we cannot record what policies other agents are following, and so don't have a notion of what a "typical" policy would be. Hence, a true evaluation would have to take place in the wild, which, unless the policy is actually safe, has high risk for human injury and property damage.

The data we do have access to is a set of observed trajectories of agents within a scene over a period of time. While this does not provide the full policies they are following (since if you changed what one of the agents did, the other agents' actions would also change), it does specify what action their policy returned given a very specific input. Therefore, in order to take advantage of what we do have access to, we instead propose an alternative problem.

Definition 1 (deviant)**.** *A set of policies $P \subset \Pi$ is deviant from a history $h \in H$ over a given time period (τ_1, τ_2) if for any timestep $\tau \in (\tau_1, \tau_2)$, any agent i took an action $a \in A$ during that time period for which there does not exist a policy $\pi \in P$ such that $\pi(h_\tau^{(i)}) = a$, where $h_\tau^{(i)}$ denotes the truncated history up to timestep τ.*

Definition 2 (collidable)**.** *A set of policies $P \subset \Pi$ is collidable over a history $h^{(i)} \in H$ and a given time period (τ_1, τ_2) if there exists an assignment of policies from P to each agent such that simulating forward starting at $h_{\tau_1}^{(i)}$ for $\tau_2 - \tau_1$ timesteps results in agent i colliding with some other agent.*

Problem 2 (Offline Navigation Problem)**.** *Given the full history of a scene $h \in H$, find a set of policies $P \subset \Pi$ minimizing the fraction of time periods for which P is deviant or collidable over h.*

This may seem like a strange objective, but it turns out that solving Problem 2 also solves Problem 1, when the dataset of recorded trajectories is representative, that is, contains all "typical" goals and policies. This is precisely because we know that people tend to reach their goals without colliding with each other. To spell it out, consider the 2 metrics in Problem 1. First, we observe that if the goal is something an agent in the dataset would have had as a goal, then that agent would likely be recorded as reaching the goal, and so either P contains a policy that leads to the goal or it is deviant. Second, we upper bound the fraction of situations in which a collision is possible by following a policy in P.

Theorem 1. *The total risk of collision R of following a policy in $P \subset \Pi$, which upper bounds the fraction of situations in which a collision is possible by following a policy in P, is given by $C + D$, where C is the fraction of situations in which P is collidable, and D is the fraction of situations in which P is deviant.*

Proof: Consider a situation for which following a policy in P leads to collision. If every other agent is also following a policy in P, then by definition this is a situation where P is collidable. If some agent is not following a policy in P, then by definition this is a situation where P is deviant. Then the number of such situations is upper bounded by the total number of situations in which P is collidable and those in which P is deviant, hence $R = C + D$. ■

Short aside: we can think of the fraction of situations in which P is not deviant as the recall (since it measures how many actual behaviors are represented), and the fraction of situations in which P is not collidable as the precision (since they are measuring how well the model excludes bad outputs). Therefore we could consider measuring a variant of the F_1 score. However, the harmonic mean in this case doesn't mean much intuitively, whereas the sum upper bounds the risk of collision, so the sum turns out to be a nicer metric to optimize for.

Now representing sets of policies is fairly difficult, and moreso checking whether any contained policy allows collision, especially since the set is usually infinitely large. While under certain assumptions, this can be approximated by sampling a large number of policies and verifying that none of them allow collision, this is not foolproof, and it is trivial to construct policy sets for which policies that lead to collision always exist but are rarely sampled (e.g. suppose there are 10 agents and each is allowed to teleport up to 1000 meters – this policy set is always collidable, but the odds that a pose sampled for one agent would be in collision with that for another agent is small because of the large sample space).

Instead, we will construct a policy set that by definition is never collidable. Then, this policy set will be evaluated solely based on how frequently it is deviant.

IV. Collision-free Policy Set

We now construct a set, first in very general terms that can be applied to any navigation domain with any set of priors, then with specific details for a 2D space with minimal priors.

A. A Parameterized Policy Set

We define a space P_{ϕ,π_0} of policy sets parameterized by 2 parameters. The first parameter is a claiming map $\phi : H \times O \times T \times \mathbb{R}^d \to \mathbb{R}$ indicating the degree to which a given agent (with observable state O) claims a given point in task space for a particular timestep (T is the set of timesteps). If $\phi(h, i, \tau, x) > \phi(h, j, \tau, x)$ for each other agent j, then agent i is considered to claim x at time τ (and hence no other agent should occupy that space-time). From this scoring function we can derive a partitioning of $T \times \mathbb{R}^d$ into disjoint sets, denoted $\psi(h^{(i)})$ for each agent i. An example of a partitioning resulting from a specific claiming map is illustrated in Figures 1 and 2, although we note that the results in this section equally apply to any claiming map. The second parameter is a default policy $\pi_0 \in \Pi$. Each agent will ensure that it is always able to safely begin following π_0 and that each other agent can safely begin following π_0 at any timestep.

The resulting policy set is then simple to construct. $\pi \in P_{\phi,\pi_0}$ iff the following conditions hold, starting from some history h_τ:

- Simulating forward the robot agent i following π starting at state $h_\tau^{(i)}$ for a single timestep does not collide with any point $x \notin \psi(h_{\tau-2}^{(i)})$.
- Simulating forward the robot agent i following π starting at state $h_\tau^{(i)}$ for a single timestep and then executing π_0 for $\eta - 1$ timesteps does not collide with any point $x \notin \psi(h_{\tau-1}^{(i)})$.
- Simulating forward another agent j following π starting at state $h_\tau^{(j)}$ for a single timestep does not collide with any point $x \in \psi(h_{\tau-2}^{(i)})$.
- Simulating forward another agent j following π starting at state $h_\tau^{(j)}$ for a single timestep and then executing π_0 for $\eta - 1$ timesteps does not collide with any point $x \in \psi(h_{\tau-1}^{(i)})$.

Here we use η to denote the time horizon (e.g. 3 seconds). Informally, this is basically saying that the robot is allowed to move anywhere in space-time that it claimed both 2 timesteps ago and 1 timestep ago (the intersection of these regions), as long as it ensures that it can execute the default policy afterwards without leaving the region it claimed 1 timestep ago, and that every other agent is allowed to do the same but with the complement of the robot's claimed region. The reason why the policy set is assymetric is because we are measuring only how often the robot in question is in collision. If our objective was to minimize all collisions regardless of which agents were involved, then we could restrict every agent to their specific claimed region. However, measuring only collisions involving the robot makes for a fairer comparison with other methods because that is what is usually measured in the literature. We restrict the policies to act based on information at least one timestep old because neither humans nor computers can actually act on new information immediately. If we set a timestep to be 80ms long, then this will allow for a reasonable amount of time for a computer to react (2 frames for a camera-based system operating at 25Hz). Examples of policies that are contained and not contained in this set for a particular initial state and claiming map for a one-dimensional space are shown in Figure 3.

Now we show that this policy set does not allow for collisions.

Theorem 2. *P_{ϕ,π_0} is not collidable for any history $h^{(i)}$*

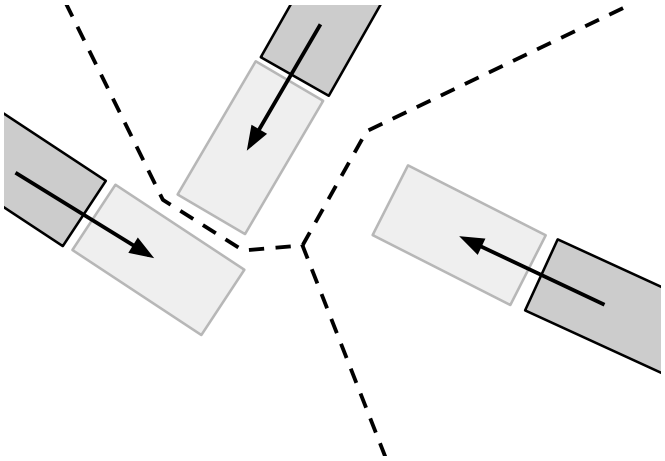

Fig. 1. An example partitioning in a 2D space at a particular time. Each agent's state at time $\tau + j$ given the default policy (depicted as a trajectory following the dotted line), along with the corresponding partitioning. Each agent's claimed region is the region of space that is closer to it's footprint at time $\tau + j$ than that of any other agent. The states at time τ are in dark gray, the states at time $\tau + j$ are in light gray, and the dotted lines divide the space at time $\tau + j$ into disjoint regions.

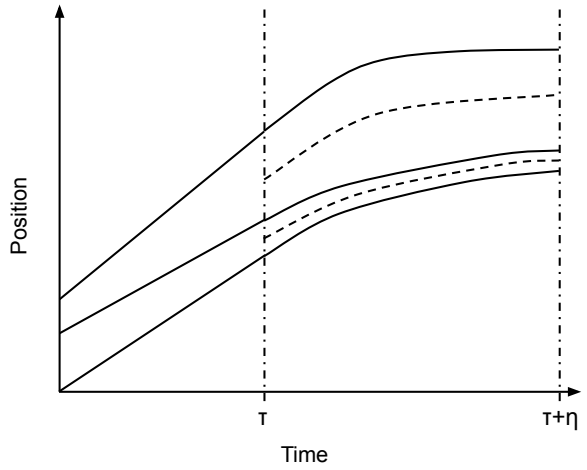

Fig. 2. An example partitioning in a 1D space over time, where each agent is a point robot. The solid curves are the trajectories resulting from the stopping policy for each agent. The dashed curves divide the space-time into the regions claimed by each agent.

and time period $(\tau, \tau+\eta)$ if agent i following the default policy starting at time τ would not lead to collision.

Proof: Suppose the robot agent i can execute a policy in P_{ϕ,π_0} for a single timestep starting at τ_1, where $\tau \leq \tau_1 \leq \tau + \eta$. Then each other agent is either also executing such a policy or the default policy; this is guaranteed to not be in collision because agent i avoids each other agent j's $\psi(h^{(j)}_{\tau_1-1})$, including any possible default trajectory. Suppose agent i cannot execute such a policy. Then it can execute the default policy, which is guaranteed to not collide with such a policy for any other agent because each other agent avoids $\psi(h^{(i)}_{\tau_1-1})$, which contains agent i's default trajectory. Agent i's default trajectory starting at τ_1 cannot collide with another agent j's default trajectory also starting at τ_1 because at time $\tau_1 - 1$ both agents selected trajectories such that an ensuing default trajectory is contained in $\psi(h^{(i)}_{\tau_1-2})$ and its complement, respectively, which are disjoint. If agent i initiates a default trajectory at τ_1 and some agent j initiates a default trajectory before τ_1, then at τ_1-1 agent i chose a trajectory that can be followed by a default trajectory at time τ while staying within $\psi(h^{(i)}_{\tau_1-2})$, which cannot intersect with any other agent's default trajectory starting at $\tau_1 - 1$. If no such trajectory could be found, then agent i would have executed the default policy at τ_1-1, and by induction we can see that the only situation where agent i's default trajectory can collide with any other agent's default trajectory is if those were the initial conditions (i.e. the preceding actions were not drawn from policies in P_{ϕ,π_0}). ■

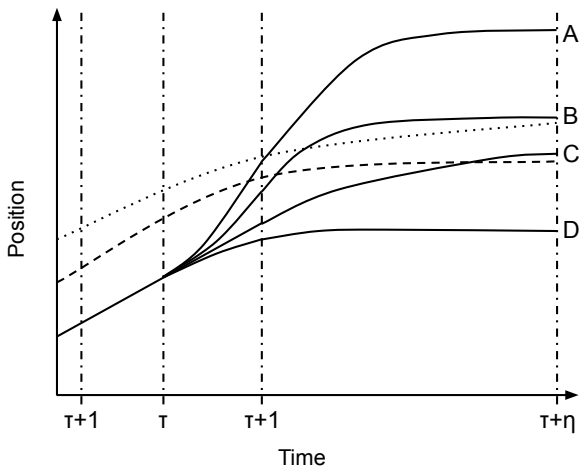

Fig. 3. We are currently trying to decide controller settings for timestep τ based on observations up to time $\tau-1$ in a one-dimensional domain. Suppose the dotted line represents the upper border of $\psi(h^{(i)}_{\tau-1})$ and the dashed line represents the upper border of $\psi(h^{(i)}_{\tau-2})$. Up to time τ is drawn the historical trajectory in addition to the expected trajectory for timestep $\tau - 1$, since the action for that timestep was already chosen at the previous timestep. Then for timestep τ there are the trajectories produced by four example policies. Each of these is followed by the default policy (depicted here as a stopping policy) from time $\tau+1$ to $\tau+\eta$. Policy A is not contained in the policy set because it exceeds $\psi(h^{(i)}_{\tau-2})$ during timestep τ. Policy B does not exceed $\psi(h^{(i)}_{\tau-1})$ or $\psi(h^{(i)}_{\tau-2})$ during timestep τ, but the default policy in the following η timesteps does exceed $\psi(h^{(i)}_{\tau-1})$, so Policy B is also not contained in the set. Policy C does not exceed $\psi(h^{(i)}_{\tau-1})$ or $\psi(h^{(i)}_{\tau-2})$ during timestep τ, nor does the subsequent default policy exceed $\psi(h^{(i)}_{\tau-1})$, so Policy C is contained in the set, even though the subsequent default policy does exceed $\psi(h^{(i)}_{\tau-2})$. Policy D does not exceed $\psi(h^{(i)}_{\tau-1})$ or $\psi(h^{(i)}_{\tau-2})$ during timestep τ or in the subsequent default policy, so it is also contained in the set.

By extension we conclude that P_{ϕ,π_0} is not collidable for any history h and time period $(\tau, \tau+\eta)$ if it is not deviant from h over time period $(\tau, \tau+1)$ (since if the default policy starting from τ leads to collision, P_{ϕ,π_0} contains no valid policies for

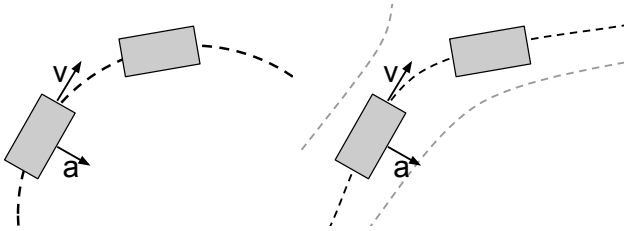

Fig. 4. The default policies π_k (left) and π_{nn} (right). In π_k, the trajectory maintains a constant longitudinal deceleration and angular velocity, following a circular path based on the centripetal acceleration at the current state. In π_{nn}, the trajectory also maintains a constant longitudinal deceleration, but the path is chosen based on the closest observed trajectory in the training set.

the colliding agents, and so whatever action that was observed would be classified as deviant). Therefore, the overall risk of P_{ϕ,π_0} is its rate of deviation.

B. Parameter Details

Suppose simulating an agent i forward following π_0 starting at state $h_\tau^{(i)}$ for j timesteps results in agent i being centered at point $y_0 \in \mathbb{R}^d$. Then we define $\phi(h, i, \tau + j, x)$ as the negative distance from x to the footprint of agent i at pose y_0. Example partitionings resulting from this scoring function are depicted in Figures 1 and 2. Informally, this scoring function says that an agent is allowed to move anywhere near its stopping trajectory, so long as another agent is not expected to be even closer.

For the default policy π_0, the simplest option is a stopping policy. We implement two potential settings for π_0, both illustrated in Figure 4. The first, π_k, is the kinematic stopping policy, in which the agent will start moving along its current kinematic curve and slow down with a fixed deceleration until it stops. The second, π_{nn}, is a learned nearest neighbor model, which predicts an agent to slow down with a fixed deceleration following the same curve as the agent in the training dataset (i.e. observations from the same location) with the closest starting pose. π_{nn} is expected to perform slightly better than π_k over longer horizons because it implicitly captures lane geometry, since the agents in the training set were presumably also following their lanes. We note that the behaviors permitted by P_{ϕ,π_k} and $P_{\phi,\pi_{nn}}$ are not limited to those described by π_k and π_{nn}, respectively. π_k and π_{nn} merely provide reference trajectories from which we can infer the degree to which a given agent has a claim on a particular region of space-time. Hence, the resulting policy sets can generalize to a broader set of policies so long as no other agent has a stronger claim on the regions of space-time that would be occupied.

Note that the results in Section IV-A hold regardless of our choice of π_0 and ϕ. The end goal would be to choose these parameters to minimize deviation, which will be empirically measured later. We expect that explicitly reasoning about vehicle dynamics, environment geometry and semantics (e.g. lanes), and interactions between agents would lead to better performance. Nevertheless, even with our overly simplistic models, we are still able to observe deviation rates far below the reported collision rates of state-of-the-art planning systems.

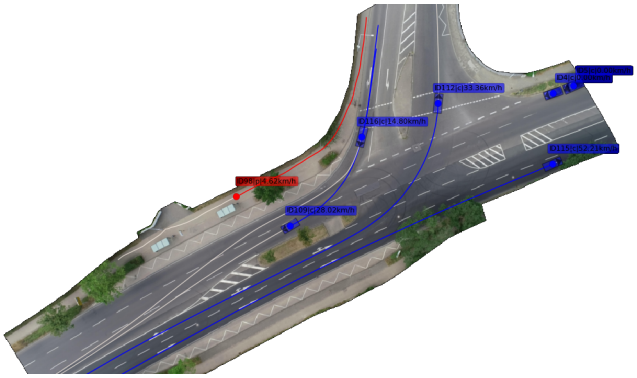

Fig. 5. An example intersection in the inD dataset [6] with recorded agent trajectories. Rendered by the inD Dataset Python Tools.

V. Empirical Results

We evaluate our method on the inD dataset [6] for every vehicle (cars and trucks, ignoring bicycles and pedestrians) that moves at least 5 meters (mostly ignoring parked vehicles). We remove segments within 5 seconds of collisions in the ground truth, assuming that those were due to slight labeling error, as well as at the end of trajectories so we are not predicting beyond what has been recorded. A small portion of the data from each location is set aside as the training set. An example intersection present in the dataset is depicted in Figure 5.

While we can't directly evaluate deviation and collidability for existing methods, we can still compare the risk bound of our method to what has been reported in the literature. These comparisons come with the caveat that they were evaluated on different datasets, and so the comparisons might not be completely fair. In particular, Zeng et al. [34] evaluate their proposed model as well as a number of baselines on an internally collected proprietary dataset, and Houston et al. [15] evaluate their proposed model on an internally collected dataset which has been made publicly available. The inD dataset [6] was collected at relatively complex unsignaled intersections, so we believe it to be at least as hard as other datasets since the rate of difficult interactions is fairly high. Furthermore, the methodology used by prior work does not take into account how others will react to the robot. Unfortunately, the vast majority of work in this area demonstrates safety primarily in simulation, and so quantitative safety metrics from real-world data are fairly sparse.

- A_K: The policy set defined in Section IV using $\pi_0 = \pi_k$ evaluated on the inD dataset [6].
- A_{NN}: The policy set defined in Section IV using $\pi_0 = \pi_{nn}$ evaluated on the inD dataset [6].
- Augmented Imitation Learning (AIL): A ResNet-50-based motion forecasting model trained to imitate expert behavior, with the training data augmented with synthetic perterbations, trained on 1000 hours of demonstration [15]. Evaluated on an openly available dataset collected by Lyft [15]. Their results are reported in terms of collisions per 1000 miles, so we will estimate the collision

rate for each time horizon assuming an average velocity of 10 m/s. Additionally, they attempt to manually classify interventions as due to nonreactivity (other agents colliding with the robot) vs actual robot misbehavior (robot colliding with other agents). We list the overall collision rate (mostly other agents colliding into the robot) under AIL[1], and only those caused by the robot under AIL[2]. It is unclear how many of the instances of other agents colliding into the robot would have actually occurred if the robot were operating in that situation.

- End-to-End Neural Network (NN): An end-to-end interpretable neural motion planner which produces intermediate representations of predicted future trajectories [34]. Evaluated on an internal Uber dataset.
- Ego-Motion (EM): A 4-layer MLP to predict future trajectory based solely on personal history [34]. Evaluated on an internal Uber dataset.
- Imitation Learning (IL): An imitation learning network to predict future trajectory of self based on personal history (encoded in same way as Ego-Motion) and the same backbone as End-to-End NN [34]. Evaluated on an internal Uber dataset.
- Adaptive Cruise Control (ACC): Following leading vehicle while staying in the center of the lane [34]. Evaluated on an internal Uber dataset.
- Manual Cost (MC): Same planning mechanism as NN but with a manually generated cost map based on simple heuristics [34]. Evaluated on an internal Uber dataset.

	Collision Rate (%)				
	1s	2s	3s	5s	10s
A_K	.00415	.0179	.0371	.114	.276
A_{NN}	.00747	.0363	.0826	.158	.314
AIL[1]	1.60	3.21	4.81	8.01	16.0
AIL[2]	.0287	.0573	.0860	.143	.287
NN	.01	.04	.78		
EM	.01	.54	1.81		
IL	.01	.55	1.72		
ACC	.12	.53	2.39		
MC	.02	.22	2.21		

VI. Conclusion and Future Work

We have presented a compromise between the absolute safety provided by reachability approaches and the non-conservative behavior allowed by systems relying on predictions. As we can see, even a very rudimentary model with few priors around motion models and rules of the road can achieve very low risk of collision, beating existing approaches by a significant margin and scaling well with horizon.

Our approach has some convenient properties. One nice side effect is that it reports exactly when it doesn't know what's going on, and so the robot can resort to some kind of fallback mechanism if necessary. It also optimizes for always allowing for the very reasonable fallback of immediately braking. We see that its risk seems to scale better with the time horizon than other approaches, although this is likely more due to limitations in ability to fairly evaluate these other approaches. By far its greatest strength is that its safety is empirically testable offline without making assumptions about how agents' actions would change in response to the robot's actions.

An interesting direction for future work would be to learn a policy set minimizing deviation on an observed training dataset. Because the metric can be evaluated offline, it becomes a supervised learning problem, which will likely be more data efficient than reinforcement learning approaches.

Additionally, designing controllers that explicitly optimize some objective subject to the constraints imposed by the policy set could lead to better performance in risky situations, rather than always falling back to the stopping policy.

References

[1] Pieter Abbeel, Dmitri Dolgov, Andrew Ng, and Sebastian Thrun. Apprenticeship learning for motion planning with application to parking lot navigation. pages 1083–1090, 09 2008. doi: 10.1109/IROS.2008.4651222.

[2] Matthias Althoff and John Dolan. Online verification of automated road vehicles using reachability analysis. *Robotics, IEEE Transactions on*, 30:903–918, 08 2014. doi: 10.1109/TRO.2014.2312453.

[3] Mohammad Bahram, Andreas Lawitzky, Jasper Friedrichs, Michael Aeberhard, and Dirk Wollherr. A game-theoretic approach to replanning-aware interactive scene prediction and planning. *IEEE Transactions on Vehicular Technology*, 65:1–1, 01 2015. doi: 10.1109/TVT.2015.2508009.

[4] Andrea Bajcsy, Somil Bansal, Eli Bronstein, Varun Tolani, and Claire Tomlin. An efficient reachability-based framework for provably safe autonomous navigation in unknown environments. pages 1758–1765, 12 2019. doi: 10.1109/CDC40024.2019.9030133.

[5] Tirthankar Bandyopadhyay, Kok Sung Won, Emilio Frazzoli, David Hsu, Wee Sun Lee, and Daniela Rus. Intention-aware motion planning. In *Workshop on the Algorithmic Foundations of Robotics*, 2012.

[6] Julian Bock, Robert Krajewski, Tobias Moers, Steffen Runde, Lennart Vater, and Lutz Eckstein. The ind dataset: A drone dataset of naturalistic road user trajectories at german intersections. 2019.

[7] Sebastian Brechtel, Tobias Gindele, and Rüdiger Dillmann. Probabilistic decision-making under uncertainty for autonomous driving using continuous pomdps. 10 2014. doi: 10.1109/ITSC.2014.6957722.

[8] Adam Bry and Nicholas Roy. Rapidly-exploring random belief trees for motion planning under uncertainty. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 723–730. IEEE, 2011.

[9] Sergio Casas, Wenjie Luo, and Raquel Urtasun. IntentNet: Learning to Predict Intention from Raw Sensor Data. In Aude Billard, Anca Dragan, Jan Peters, and Jun Morimoto, editors, *Proceedings of The 2nd Conference on Robot Learning*, volume 87 of *Proceedings of Machine Learning Research*, pages 947–956. PMLR, 29–31 Oct 2018.

[10] Sergio Casas, Cole Gulino, Simon Suo, Katie Luo, Renjie Liao, and Raquel Urtasun. Implicit latent variable model for scene-consistent motion forecasting, 2020.

[11] Mo Chen and Claire Tomlin. Hamilton–jacobi reachability: Some recent theoretical advances and applications in unmanned airspace management. *Annual Review of Control, Robotics, and Autonomous Systems*, 1:333–358, 05 2018. doi: 10.1146/annurev-control-060117-104941.

[12] Jonathan DeCastro, Javier Alonso-Mora, Lucas Liebenwein, Wilko Schwarting, Cristian-Ioan Vasile, Sertac Karaman, and Daniela Rus. Compositional and contract-based verification for autonomous driving on road networks. 12 2017.

[13] Michael During and Patrick Pascheka. Cooperative decentralized decision making for conflict resolution among autonomous agents. pages 154–161, 06 2014. ISBN 978-1-4799-3020-3. doi: 10.1109/INISTA.2014.6873612.

[14] Jason Hardy and Mark Campbell. Contingency planning over probabilistic hybrid obstacle predictions for autonomous road vehicles. In *IEEE International Conference on Intelligent Robots and Systems*, 2010.

[15] John Houston, Guido Zuidhof, Luca Bergamini, Yawei Ye, Long Chen, Ashesh Jain, Sammy Omari, Vladimir Iglovikov, and Peter Ondruska. One thousand and one hours: Self-driving motion prediction dataset. In *Conference on Robot Learning*, 2020.

[16] Sandy Huang, David Held, Pieter Abbeel, and Anca Dragan. Enabling Robots to Communicate Their Objectives. *Robotics: Science and Systems XIII*, Jul 2017. doi: 10.15607/rss.2017.xiii.059.

[17] Michael Janner, Igor Mordatch, and Sergey Levine. γ-models: Generative temporal difference learning for infinite-horizon prediction. In *Advances in Neural Information Processing Systems*, 2020.

[18] Eric Kim, Murat Arcak, and Sanjit Seshia. Compositional controller synthesis for vehicular traffic networks. pages 6165–6171, 12 2015. doi: 10.1109/CDC.2015.7403189.

[19] David Lenz, Tobias Kessler, and Alois Knoll. Tactical cooperative planning for autonomous highway driving using monte-carlo tree search. pages 447–453, 06 2016. doi: 10.1109/IVS.2016.7535424.

[20] Karen Leung, Edward Schmerling, Mengxuan Zhang, Mo Chen, John Talbot, J Gerdes, and Marco Pavone. On infusing reachability-based safety assurance within planning frameworks for human–robot vehicle interactions. *The International Journal of Robotics Research*, 39:027836492095079, 08 2020. doi: 10.1177/0278364920950795.

[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*, 26(5):1782–1797, 2018. doi: 10.1109/TCST.2017.2723574.

[22] Stefan Liu, Hendrik Roehm, Christian Heinzemann, Ingo Lütkebohle, Jens Oehlerking, and Matthias Althoff. Provably safe motion of mobile robots in human environments. 09 2017. doi: 10.1109/IROS.2017.8202313.

[23] W. Luo, B. Yang, and R. Urtasun. Fast and furious: Real time end-to-end 3d detection, tracking and motion forecasting with a single convolutional net. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3569–3577, 2018. doi: 10.1109/CVPR.2018.00376.

[24] Petter Nilsson, Omar Hussien, Ayca Balkan, Yuxiao Chen, Aaron Ames, Jessy Grizzle, Necmiye Ozay, Huei Peng, and Paulo Tabuada. Correct-by-construction adaptive cruise control: Two approaches. *IEEE Transactions on Control Systems Technology*, 24:1–14, 12 2015. doi: 10.1109/TCST.2015.2501351.

[25] R. Platt, R. Tedrake, L. Kaelbling, and T. Lozano-Perez. Belief space planning assuming maximum likelihood observations. In *Proceedings of Robotics: Science and Systems*, Zaragoza, Spain, June 2010. doi: 10.15607/RSS.2010.VI.037.

[26] Sam Prentice and Nicholas Roy. The belief roadmap: Efficient planning in linear pomdps by factoring the covariance. In *Proceedings of the 13th International Symposium of Robotics Research (ISRR)*, Hiroshima, Japan, November 2007.

[27] Stephane Ross, Geoffrey Gordon, and J. Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. *Journal of Machine Learning Research - Proceedings Track*, 15, 11 2010.

[28] Dorsa Sadigh, Shankar Sastry, Sanjit Seshia, and Anca Dragan. Planning for autonomous cars that leverage effects on human actions. 06 2016. doi: 10.15607/RSS.2016.XII.029.

[29] Wilko Schwarting and Patrick Pascheka. Recursive conflict resolution for cooperative motion planning in dynamic highway traffic. pages 1039–1044, 10 2014. doi: 10.1109/ITSC.2014.6957825.

[30] Wilko Schwarting, Javier Alonso-Mora, and Daniela Rus. Planning and Decision-Making for Autonomous Vehicles. *Annual Review of Control, Robotics, and Autonomous Systems*, 1(1):187–210, 2018. doi: 10.1146/annurev-control-060117-105157.

[31] Bastian Schürmann, Daniel Heß, Jan Eilbrecht, Olaf Stursberg, Frank Köster, and Matthias Althoff. Ensuring drivability of planned motions using formal methods. 10 2018.

[32] Waymo. Waymo Safety Report. 2020.

[33] H. Xu, Y. Gao, F. Yu, and T. Darrell. End-to-end learning of driving models from large-scale video datasets. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3530–3538, 2017. doi: 10.1109/CVPR.2017.376.

[34] Wenyuan Zeng, Wenjie Luo, Simon Suo, Abbas Sadat, Bin Yang, Sergio Casas, and Raquel Urtasun. End-to-end interpretable neural motion planner. pages 8652–8661, 06 2019. doi: 10.1109/CVPR.2019.00886.

[35] Wei Zhan, Changliu Liu, Ching-Yao Chan, and

Masayoshi Tomizuka. A non-conservatively defensive strategy for urban autonomous driving. In *IEEE 19th International Conference on Intelligent Transportation Systems*, 2016.

[36] Jiakai Zhang and Kyunghyun Cho. Query-efficient imitation learning for end-to-end autonomous driving. 05 2016.

[37] Hang Zhao, Jiyang Gao, Tian Lan, Chen Sun, Benjamin Sapp, Balakrishnan Varadarajan, Yue Shen, Yi Shen, Yuning Chai, Cordelia Schmid, Congcong Li, and Dragomir Anguelov. Tnt: Target-driven trajectory prediction, 2020.

[38] Brian D. Ziebart, Andrew Maas, J. Andrew Bagnell, and Anind K. Dey. Maximum entropy inverse reinforcement learning. In *Proc. AAAI*, pages 1433–1438, 2008.