Interpretable Goal-based Prediction and Planning for Autonomous Driving

Interpretable Goal-based Prediction and Planning for Autonomous Driving

Stefano V. Albrecht*†, Cillian Brewitt*†, John Wilhelm*†, Balint Gyevnar*†,
Francisco Eiras*‡, Mihai Dobre*, Subramanian Ramamoorthy*†
*Five AI Ltd., UK, {firstname.lastname}@five.ai
†School of Informatics, University of Edinburgh, UK
‡Department of Engineering Science, University of Oxford, UK

***Abstract*— We propose an integrated prediction and planning system for autonomous driving which uses rational inverse planning to recognise the goals of other vehicles. Goal recognition informs a Monte Carlo Tree Search (MCTS) algorithm to plan optimal maneuvers for the ego vehicle. Inverse planning and MCTS utilise a shared set of defined maneuvers and macro actions to construct plans which are explainable by means of *rationality* principles. Evaluation in simulations of urban driving scenarios demonstrate the system's ability to robustly recognise the goals of other vehicles, enabling our vehicle to exploit non-trivial opportunities to significantly reduce driving times. In each scenario, we extract intuitive explanations for the predictions which justify the system's decisions.**

I. Introduction

The ability to predict the intentions and driving trajectories of other vehicles is a key problem for autonomous driving [1]. This problem is significantly complicated by the need to make fast and accurate predictions based on limited observation data which originate from coupled multi-agent interactions.

To make prediction tractable in such conditions, a standard approach in autonomous driving research is to assume that vehicles use one of a finite number of distinct high-level maneuvers, such as lane-follow, lane-change, turn, stop, etc. [2], [3], [4], [5], [6], [7]. A classifier of some type is used to detect a vehicle's current executed maneuver based on its observed driving trajectory. The limitation in such methods is that they only detect the *current* maneuver of other vehicles, hence planners using such predictions are effectively limited to the timescales of the detected maneuvers. An alternative approach is to specify a finite set of possible *goals* for each other vehicle (such as road exit points) and to plan a full trajectory to each goal from the vehicle's observed local state [8], [9], [10]. While this approach can generate longer-term predictions, a limitation is that the generated trajectories must be matched relatively closely by a vehicle in order to yield high-confidence predictions of the vehicle's goals.

Recent methods based on deep learning have shown promising results for trajectory prediction in autonomous driving [11], [12], [13], [14], [15], [16], [17]. Prediction models are trained on large datasets that are becoming available through data gathering campaigns involving sensorised vehicles (e.g. video, lidar, radar). Reliable prediction over several second horizons remains a hard problem, in part due to the difficulties in capturing the coupled evolution of traffic. In our view, one of the most significant limitations of this class of methods (though see recent progress [18]) is the difficulty in extracting interpretable predictions in a form that is amenable to efficient integration with planning methods that effectively represent multi-dimensional and hierarchical task objectives.

Our starting point is that in order to predict the future maneuvers of a vehicle, we must reason about *why* – that is, to what end – the vehicle performed its past maneuvers, which will yield clues as to its intended goal [19]. Knowing the goals of other vehicles enables prediction of their future maneuvers and trajectories, which facilitates planning over extended timescales. We show in our work (illustrated in Figure 2) how such reasoning can help to address the problem of overly-conservative autonomous driving [20]. Further, to the extent that our predictions are structured around the interpretation of observed trajectories in terms of high-level maneuvers, the goal recognition process lends itself to *intuitive interpretation* for the purposes of system analysis and debugging, at a level of detail suggested in Figure 2. As we develop towards making our autonomous systems more trustworthy [21], these notions of interpretation and the ability to justify (explain) the system's decisions are key [22].

To this end, we propose *Interpretable Goal-based Prediction and Planning* (IGP2) which leverages the computational advantages of using a finite space of maneuvers, but extends the approach to planning and prediction of *sequences* (i.e., plans) of maneuvers. We achieve this via a novel integration of rational inverse planning [23], [24] to recognise the goals of other vehicles, with Monte Carlo Tree Search (MCTS) [25] to plan optimal maneuvers for the ego vehicle. Inverse planning and MCTS utilise a shared set of defined maneuvers to construct plans which are explainable by means of *rationality* principles, i.e. plans are optimal with respect to given metrics. We evaluate IGP2 in simulations of diverse urban driving scenarios, showing that (1) the system robustly recognises the goals of other vehicles, even if significant parts of a vehicle's trajectory are occluded, (2) goal recognition enables our vehicle to exploit opportunities to improve driving efficiency as measured by driving time compared to other prediction baselines, and (3) we are able to extract intuitive explanations for the predictions to justify the system's decisions.

In summary, our contributions are:

- A method for goal recognition and multi-modal trajectory prediction via rational inverse planning.

S.A. is supported by a Royal Society Industry Fellowship. C.B., J.W., B.G. were interns at Five AI with partial financial support from the Royal Society and UKRI. **IGP2 code:** `https://github.com/uoe-agents/IGP2`

- Integration of goal recognition with MCTS planning to generate optimised plans for the ego vehicle.
- Evaluation in simulated urban driving scenarios showing accurate goal recognition, improved driving efficiency, and ability to interpret the predictions and ego plans.

II. Preliminaries and Problem Definition

Let $\mathcal{I}$ be the set of vehicles in the local neighbourhood of the ego vehicle (including itself). At time t, each vehicle $i \in \mathcal{I}$ is in a local state $s_t^i \in \mathcal{S}^i$, receives a local observation $o_t^i \in \mathcal{O}^i$, and can choose an action $a_t^i \in \mathcal{A}^i$. We write $s_t \in \mathcal{S} = \times_i \mathcal{S}^i$ for the joint state and $s_{a:b}$ for the tuple $(s_a, ..., s_b)$, and similarly for $o_t \in \mathcal{O}, a_t \in \mathcal{A}$. Observations depend on the joint state via $p(o_t^i|s_t)$, and actions depend on the observations via $p(a_t^i|o_{1:t}^i)$. In our system, a local state contains a vehicle's pose, velocity, and acceleration (we use the terms velocity and speed interchangeably); an observation contains the poses and velocities of nearby vehicles; and an action controls the vehicle's steering and acceleration. The probability of a sequence of joint states $s_{1:n}$ is given by

$$p(s_{1:n}) = \prod_{t=1}^{n-1} \int_{\mathcal{O}} \int_{\mathcal{A}} p(o_t|s_t) p(a_t|o_{1:t}) p(s_{t+1}|s_t, a_t)\, do_t\, da_t \tag{1}$$

where $p(s_{t+1}|s_t, a_t)$ defines the joint vehicle dynamics, and we assume independent local observations and actions, $p(o_t|s_t) = \prod_i p(o_t^i|s_t)$ and $p(a_t|o_{1:t}) = \prod_i p(a_t^i|o_{1:t}^i)$. Vehicles react to other vehicles via their observations $o_{1:n}^i$.

We define the planning problem as finding an optimal policy π^* which selects the actions for the ego vehicle, ε, to achieve a specified goal, G^ε, while optimising the driving trajectory via a defined reward function. Here, a policy is a function $\pi : (\mathcal{O}^\varepsilon)^* \mapsto \mathcal{A}^\varepsilon$ which maps an observation sequence $o_{1:n}^\varepsilon$ to an action a_t^ε. (State filtering [26] is outside the scope of this work.) A goal can be any subset of local states, $G^\varepsilon \subset \mathcal{S}^\varepsilon$. In this paper, we focus on goals that specify target locations and "stopping goals" which specify a target location and zero velocity. Formally, define

$$\Omega_n = \left\{ s_{1:n} \;\middle|\; s_n^\varepsilon \in G^\varepsilon \wedge \forall m < n : s_m^\varepsilon \notin G^\varepsilon \right\} \tag{2}$$

where $s_n^\varepsilon \in G^\varepsilon$ means that s_n^ε satisfies G^ε. The second condition in (2) ensures that $\sum_{n=1}^{\infty} \int_{\Omega_n} p(s_{1:n}) ds_{1:n} \leq 1$ for any policy π, which is needed for soundness of the sum in (3). The problem is to find π^* such that

$$\pi^* \in \arg\max_{\pi} \sum_{n=1}^{\infty} \int_{\Omega_n} p(s_{1:n}) R^\varepsilon(s_{1:n})\, ds_{1:n} \tag{3}$$

where $R^i(s_{1:n})$ is vehicle i's reward for $s_{1:n}$. We define R^i as a weighted sum of reward elements based on trajectory execution time, longitudinal and lateral jerk, path curvature, and safety distance to leading vehicle.

III. IGP2: Interpretable Goal-based Prediction and Planning

Our general approach relies on two assumptions: (1) each vehicle seeks to reach some (unknown) goal from a set of possible goals, and (2) each vehicle follows a plan generated from a finite library of defined maneuvers.

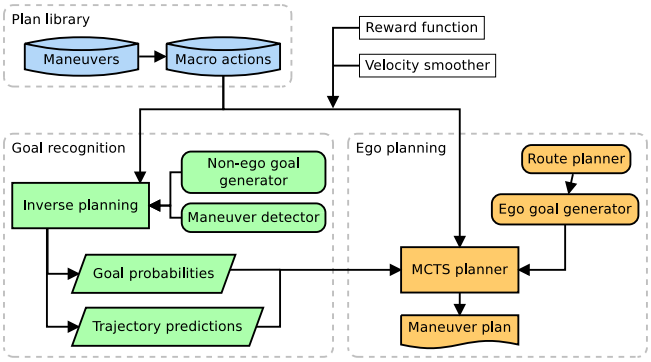


Fig. 1: IGP2 system overview.

Figure 1 provides an overview of the components in our proposed IGP2 system. At a high level, IGP2 approximates the optimal ego policy π^* as follows: For each non-ego vehicle, generate its possible goals and inversely plan for that vehicle to each goal. The resulting goal probabilities and predicted trajectories for each non-ego vehicle inform the simulation process of a Monte Carlo Tree Search (MCTS) algorithm, to generate an optimal maneuver plan for the ego vehicle toward its current goal. In order to keep the required search depth in inverse planning and MCTS shallow (and thus efficient), both plan over a shared set of macro actions which flexibly concatenate maneuvers using context information. We detail these components in the following sections.

A. Maneuvers

We assume that at any time, each vehicle is executing one of the following maneuvers: *lane-follow, lane-change-left/right, turn-left/right, give-way, stop.* Each maneuver ω specifies applicability and termination conditions. For example, lane-change-left is only applicable if there is a lane in same driving direction to the left of the vehicle, and terminates once the vehicle has reached the new lane and its orientation is aligned with the lane. Some maneuvers have free parameters, e.g. *follow-lane* has a parameter to specify when to terminate.

If applicable, a maneuver specifies a local trajectory $\hat{s}_{1:n}^i$ to be followed by the vehicle, which includes a reference path in the global coordinate frame and target velocities along the path. For convenience in exposition, we assume that $\hat{s}^i$ uses the same representation and indexing as s^i, but in general this does not have to be the case (for example, $\hat{s}$ may be indexed by longitudinal position rather than time, which can be interpolated to time indices). In our system, the reference path is generated via a Bezier spline function fitted to a set of points extracted from the road topology, and target velocities are set using domain heuristics similar to [27].

B. Macro Actions

Macro actions specify common sequences of maneuvers and automatically set the free parameters (if any) in maneuvers based on context information such as road layout. Table I specifies the macro actions used in our system. The applicability condition of a macro action is given by the applicability

Macro action:	Additional applicability condition:	Maneuver sequence (maneuver parameters in brackets):
Continue	—	*lane-follow* (end of visible lane)
Continue next exit	Must be in roundabout and not in outer-lane	*lane-follow* (next exit point)
Change left/right	There is a lane to the left/right	*lane-follow* (until target lane clear), *lane-change-left/right*
Exit left/right	Exit point on same lane ahead of car and in correct direction	*lane-follow* (exit point), *give-way* (relevant lanes), *turn-left/right*
Stop	There is a stopping goal ahead of the car on the current lane	*lane-follow* (close to stopping point), *stop*

TABLE I: Macro actions used in our system. Each macro action concatenates one or more maneuvers and automatically sets their parameters.

condition of the first maneuver in the macro action as well as optional additional conditions. The termination condition of a macro action is given by the termination condition of the last maneuver in the macro action.

C. Velocity Smoothing

To obtain a feasible trajectory across maneuvers for vehicle i, we define a velocity smoothing operation which optimises the target velocities in a given trajectory $\hat{s}^i_{1:n}$. Let $\hat{x}_t$ be the longitudinal position on the reference path at $\hat{s}^i_t$ and $\hat{v}_t$ its target velocity, for $1 \leq t \leq n$. We define $\kappa : x \to v$ as the piecewise linear interpolation of target velocities between points $\hat{x}_t$. Given the time elapsed between two time steps, Δt; the maximum velocity and acceleration, v_{max}/a_{max}; and setting $x_1 = \hat{x}_1, v_1 = \hat{v}_1$, we define velocity smoothing as

$$\begin{aligned} \min_{x_{2:n}, v_{2:n}} \quad & \sum_{t=1}^{n} ||v_t - \kappa(x_t)||_2 + \lambda \sum_{t=1}^{n-1} ||v_{t+1} - v_t||_2 \\ & \text{s.t. } x_{t+1} = x_t + v_t \Delta t \\ & 0 < v_t < v_{\max},\ v_t \leq \kappa(x_t) \\ & |v_{t+1} - v_t| < a_{\max} \Delta t \end{aligned} \tag{4}$$

where $\lambda > 0$ is the weight given to the acceleration part of the optimisation objective. Eq. (4) is a nonlinear non-convex optimisation problem which can be solved, e.g., using a primal-dual interior point method (we use IPOPT [28]). From the solution of the problem, $(x_{2:n}, v_{2:n})$, we interpolate to obtain the achievable velocities at the original points $\hat{x}_t$.

D. Goal Recognition

We assume that each non-ego vehicle i seeks to reach one of a finite number of possible goals $G^i \in \mathcal{G}^i$, using plans constructed from our defined macro actions. We use the framework of rational inverse planning [23], [24] to compute a Bayesian posterior distribution over i's goals at time t

$$p(G^i | s_{1:t}) \propto L(s_{1:t} | G^i) p(G^i) \tag{5}$$

where $L(s_{1:t}|G^i)$ is the likelihood of i's observed trajectory assuming its goal is G^i, and $p(G^i)$ specifies the prior probability of G^i.

The likelihood is a function of the reward difference between two plans: the reward $\hat{r}$ of the optimal trajectory from i's initial observed state s^i_1 to goal G^i after velocity smoothing, and the reward $\bar{r}$ of the trajectory which follows the observed trajectory until time t and then continues optimally to goal G^i, with smoothing applied only to the trajectory after t. The likelihood is defined as

$$L(s_{1:t}|G^i) = \exp(\beta(\bar{r} - \hat{r})) \tag{6}$$

where β is a scaling parameter (we use $\beta = 1$). This likelihood definition assumes that vehicles drive approximately *rationally* (i.e., optimally) to achieve their goals while allowing for some deviation. If a goal is infeasible, we set its probability to zero.

Algorithm 1 shows the pseudo code for our goal recognition algorithm, with further details in below subsections.

1) **Goal Generation:** A heuristic function is used to generate a set of possible goals $\mathcal{G}^i$ for vehicle i based on its location and context information such as road layout. In our system, we include goals for the visible end of the current road and connecting roads (bounded by the ego vehicle's view region). In addition to such static goals, it is also possible to add dynamic goals which depend on current traffic. For example, in the dense merging scenario shown in Figure 2d, stopping goals are dynamically added to model a vehicle's intention to allow the ego vehicle to merge in front.

2) **Maneuver Detection:** Maneuver detection is used to detect the current executed maneuver of a vehicle (at time t), allowing inverse planning to complete the maneuver before planning onward. We assume a module which computes probabilities over current maneuvers, $p(\omega^i)$, for each vehicle i. One option is Bayesian changepoint detection (e.g. [29]). The details of maneuver detection are outside the scope of our paper and in our experiments we use a simulated detector (cf. Sec IV-B). As different current maneuvers may hint at different goals, we perform inverse planning for each possible current maneuver for which $p(\omega^i) > 0$. Thus, each current maneuver produces its associated posterior probabilities over goals, denoted by $p(G^i \mid s_{1:t}, \omega^i)$.

3) **Inverse Planning:** Inverse planning is done using A* search [30] over macro actions. A* starts after completing the current maneuver ω^i which produces the initial trajectory $\hat{s}_{1:\tau}$. Each search node q corresponds to a state $s \in \mathcal{S}$, with initial node at state $\hat{s}_\tau$, and macro actions are filtered by their applicability conditions applied to s. A* chooses the next macro action leading to a node q' which has lowest estimated total cost[1] to goal G^i, given by $f(q') = l(q') + h(q')$. The cost $l(q')$ to reach node q' is given by the driving time from i's location in the initial search node to its location in q', following the trajectories returned by the macro actions leading to q'. A* uses the assumption that all other vehicles not planned for use a constant-velocity lane-following model after their observed trajectories. We do not check for collisions during inverse planning. The cost heuristic $h(q')$ to estimate remaining cost from q' to goal G^i is given by the driving time from i's location in q' to goal via straight line at speed

[1]Here we use the term "cost" in keeping with standard A* terminology and to differentiate from the reward function defined in Sec. II.

Algorithm 1 Goal recognition algorithm

Input: vehicle i, current maneuver ω^i, observations $s_{1:t}$
Returns: goal probabilities $p(G^i|s_{1:t}, \omega^i)$

Generate possible goals $G^i \in \mathcal{G}^i$ from state s_t^i
Set prior probabilities $p(G^i)$ (e.g. uniform)
for all $G^i \in \mathcal{G}^i$ **do**
  $\hat{s}^i_{1:n} \leftarrow$ A*(ω^i) from $\hat{s}^i_1 = s^i_1$ to G^i
  Apply velocity smoothing to $\hat{s}^i_{1:n}$
  $\hat{r} \leftarrow$ reward $R^i(\hat{s}^i_{1:n})$
  $\bar{s}^i_{1:m} \leftarrow$ A*(ω^i) from $\bar{s}^i_t$ to G^i, with $\bar{s}^i_{1:t} = s^i_{1:t}$
  Apply velocity smoothing to $\bar{s}^i_{t+1:m}$
  $\bar{r} \leftarrow$ reward $R^i(\bar{s}^i_{1:m})$
  $L(s_{1:t}|G^i, \omega^i) \leftarrow \exp(\beta(\bar{r} - \hat{r}))$
Return $p(G^i|s_{1:t}, \omega^i) \propto L(s_{1:t}|G^i, \omega^i)\, p(G^i)$

limit. This definition of $h(q')$ is admissible as per A* theory, which ensures that the search returns an optimal plan. After the optimal plan is found, we extract the complete trajectory $\hat{s}^i_{1:n}$ from the maneuvers in the plan and initial segment $\hat{s}_{1:\tau}$.

4) **Trajectory Prediction:** Our system predicts multiple plausible trajectories for a given vehicle and goal. This is required because there are situations in which different trajectories may be (near-)optimal but may lead to different predictions which could require different behaviour on the part of the ego vehicle. We run A* search for a fixed amount of time and let it compute a set of plans with associated rewards (up to some fixed number of plans). Any time A* search finds a node that reaches the goal, the corresponding plan is added to the set of plans. Given a set of smoothed trajectories $\{\hat{s}^{i,k}_{1:n}|\omega^i, G^i\}_{k=1..K}$ to goal G^i with initial maneuver ω^i and associated reward $r_k = R^i(\hat{s}^{i,k}_{1:n})$, we compute a distribution over the trajectories via a Boltzmann distribution

$$p(\hat{s}^{i,k}_{1:n}) \propto \exp(\gamma\, r_k) \tag{7}$$

where γ is a scaling parameter (we use $\gamma = 1$). Similar to Eq. (6), Eq. (7) encodes the assumption that trajectories which are closer to optimal are more likely.

E. Ego Vehicle Planning

To compute an optimal plan for the ego vehicle, we use the goal probabilities and predicted trajectories to inform a Monte Carlo Tree Search (MCTS) algorithm [25] (see Algorithm 2).

The algorithm performs a number of closed-loop simulations $\hat{s}_{t:n}$, starting in the current state $\hat{s}_t = s_t$ down to some fixed search depth or until a goal state is reached. At the start of each simulation, for each non-ego vehicle, we first sample a current maneuver, then goal, and then trajectory for the vehicle using the associated probabilities (cf. Section III-D). Each node q in the search tree corresponds to a state $s \in \mathcal{S}$ and macro actions are filtered by their applicability conditions applied to s. After selecting a macro action μ using some exploration technique (we use UCB1 [31]), the state in the current search node is forward-simulated based on the trajectory generated by the macro action μ and the sampled trajectories of non-ego vehicles, resulting in a partial trajectory $\hat{s}_{\tau:\iota}$ and new search node q' with state $\hat{s}_\iota$. Forward-simulation

Algorithm 2 Monte Carlo Tree Search algorithm

Returns: optimal maneuver for ego vehicle ε in state s_t
Perform K simulations:

Search node $q.s \leftarrow s_t$ ($root$ node)
Search depth $d \leftarrow 0$
for all $i \in \mathcal{I} \setminus \{\varepsilon\}$ **do**
  Sample current maneuver $\omega^i \sim p(\omega^i)$
  Sample goal $G^i \sim p(G^i \,|\, s_{1:t}, \omega^i)$
  Sample trajectory $\hat{s}^i_{1:n} \in \{\hat{s}^{i,k}_{1:n} \,|\, \omega^i, G^i\}$ with $p(\hat{s}^{i,k}_{1:n})$
while $d < d_{max}$ **do**
  Select macro action μ for ε applicable in $q.s$
  $\hat{s}_{\tau:\iota} \leftarrow$ Simulate μ until it terminates, with non-ego vehicles following their sampled trajectories $\hat{s}^i_{1:n}$
  $r \leftarrow \emptyset$
  if ego vehicle collides during $\hat{s}_{\tau:\iota}$ **then**
    $r \leftarrow r_{coll}$
  else if $\hat{s}^\varepsilon_\iota$ achieves ego goal G^ε **then**
    $r \leftarrow R^\varepsilon(\hat{s}_{t:n})$
  else if $d = d_{max} - 1$ **then**
    $r \leftarrow r_{term}$
  if $r \neq \emptyset$ **then**
    Use (8) to backprop r along search branches (q, μ, q') that generated the simulation
    Start next simulation
  $q'.s = \hat{s}_\iota$; $q \leftarrow q'$; $d \leftarrow d + 1$
Return maneuver for ε in s_t, $\mu \in \arg\max_\mu Q(root, \mu)$

of trajectories uses a combination of proportional control and adaptive cruise control (based on IDM [32]) to control a vehicle's acceleration and steering. Termination conditions of maneuvers are monitored in each time step based on the vehicle's observations. Collision checking is performed on $\hat{s}_{\tau:\iota}$ to check whether the ego vehicle collided, in which case we set the reward to $r \leftarrow r_{coll}$ which is back-propagated using (8), where r_{coll} is a method parameter. Otherwise, if the new state $\hat{s}_\iota$ achieves the ego goal G^ε, we compute the reward for back-propagation as $r \leftarrow R^\varepsilon(\hat{s}_{t:n})$. If the search reached its maximum depth d_{max} without colliding or achieving the goal, we set $r \leftarrow r_{term}$ which can be a constant or based on heuristic reward estimates similar to A* search.

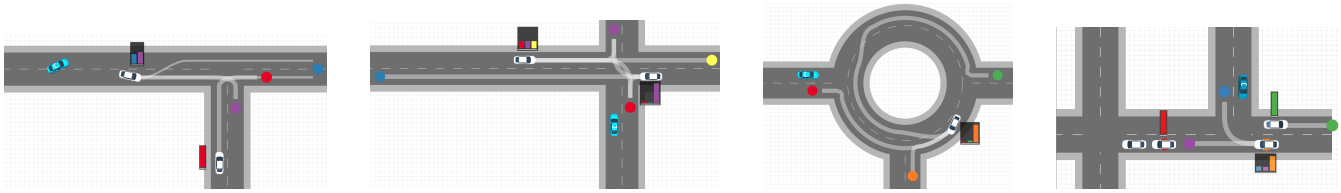
The reward r is back-propagated through search branches (q, μ, q') that generated the simulation, using a 1-step off-policy update function (similar to Q-learning [33])

$$Q(q,\mu) \leftarrow Q(q,\mu) + \begin{cases} \delta^{-1}[r - Q(q,\mu)] \text{ if } q \text{ leaf node, else} \\ \delta^{-1}[\max_{\mu'} Q(q',\mu') - Q(q,\mu)] \end{cases} \tag{8}$$

where δ is the number of times that macro action μ has been selected in q. After the simulations are completed, the algorithm selects the best macro action for execution in s_t from the root node, $\arg\max_\mu Q(root, \mu)$.

IV. Evaluation

We evaluate IGP2 in simulations of diverse urban driving scenarios, showing that: (1) our inverse planning method robustly recognises the goals of non-ego vehicles; (2) goal recognition leads to improved driving efficiency measured by driving time; and (3) intuitive explanations for the predictions



(a) Scenario 1 (S1) (b) Scenario 2 (S2) (c) Scenario 3 (S3) (d) Scenario 4 (S4)

Fig. 2: **IGP2 in 4 test scenarios.** Ego vehicle shown in blue. Bar plots show goal probabilities for non-ego vehicles. For each goal, up to two of the most probable predicted trajectories to goal are shown with thickness proportional to probability. (a) **S1**: Ego's goal is blue goal. Vehicle V_1 is on the ego's road, V_1 changes from left to right lane, biasing the ego prediction towards the belief that V_1 will exit, since a lane change would be irrational if V_1's goal was to go east. As exiting will require a significant slowdown, the ego decides to switch lanes to avoid being slowed down too. (b) **S2**: Ego's goal is blue goal. Vehicle V_1 is approaching the junction from the east and vehicle V_2 from the west. As V_1 approaches the junction, slows down and waits to take a turn, the ego's belief that V_1 will turn right increases significantly, since it would be irrational to stop if the goal was to turn left or go straight. Since the ego recognised V_1's goal is to go north, it predicts that V_1 will wait until V_2 has passed, giving the ego an opportunity to enter the road. (c) **S3**: Ego's goal is green goal. As V_1 changes from the inside to the outside lane of the roundabout and decreases its speed, it significantly biases the ego prediction towards the belief that V_1 will take the south exit since that is the rational course of action for that goal. This encourages the ego to enter the roundabout while V_1 is still in roundabout. (d) **S4**: Ego's goal is purple goal. With two vehicles stopped at the junction at a traffic light, vehicle V_1 is approaching them from behind, and vehicle V_2 is crossing in the opposite direction. When V_1 reaches zero velocity, the goal generation function adds a stopping goal (orange) for V_1 in its current position, shifting the goal distribution towards it since stopping is not rational for the north/west goals. The interpretation is that V_1 wants the ego to merge in front of V_1, which the ego then does.

can be extracted to justify the system's decisions. (Video showing IGP2 in action: `https://www.five.ai/igp2`.)

A. Scenarios

We use two sets of scenario instances. For in-depth analysis of goal recognition and planning, we use four defined local interaction scenarios shown in Figure 2. For each of these scenarios, we generate 100 instances with randomly offset initial longitudinal positions ($\sim [-10, +10]$ meters) and initial speed sampled from range $[5, 10]$ m/s for each vehicle including ego vehicle. Here the ego vehicle observes the whole scenario. To further assess IGP2's ability to complete full routes with random traffic, we use two random town layouts shown in Figure 3. Each town spans an area of 0.16 square kilometers and consists of roads, crossings, and roundabouts with 2–4 lanes each. Each junction has one defined priority road. The ego vehicle's observation radius in towns is 50 meters. Non-ego vehicles are spawned within 25 meters outside the ego observation radius, with random road, lane, speed, and goal. The total number of non-ego vehicles within the ego radius and spawning radius is kept at 8 to maintain a consistent medium-to-high level of traffic. In each town we generate 10 instances by choosing random routes for the ego vehicle to complete. The ego vehicle's goal is continually updated to be the outermost point on the route within the ego observation radius. In all simulations, the non-ego vehicles use manual heuristics to select from the maneuvers in Section III-A to reach their goals. All vehicles use independent proportional controllers for acceleration and steering, and IDM [32] for automatic distance-keeping. Vehicle motion is simulated using a kinematic bicycle model.

B. Algorithms & Parameters

We compare the following algorithms in scenarios S1–S4. **IGP2:** full system using goal recognition and MCTS. **IGP2-MAP:** like IGP2, but MCTS uses only the most probable goal and trajectory for each vehicle. **CVel:** MCTS without goal recognition, replaced by constant-velocity lane-following prediction after completion of current maneuver. **CVel-Avg:** like CVel, but uses velocity averaged over the past 2 seconds. **Cons:** like CVel, but using a conservative *give-way* maneuver which always waits until all oncoming vehicles on priority lanes have passed. In the town scenarios we focus on IGP2 and Cons, and additionally compare to **SH-CVel** which works similarly to MPDM [5]: it simulates each macro action followed by a default *Continue* macro action, using CVel prediction for non-ego vehicles, then choosing the macro action with maximum estimated reward. (SH stands for "short horizon" as the search depth is effectively limited to 1.)

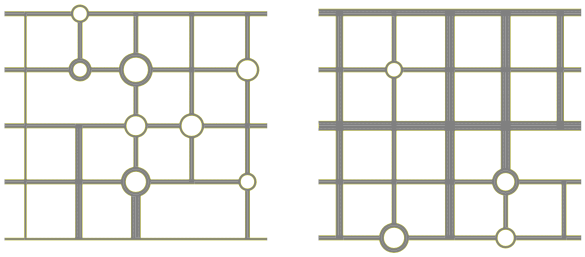


Fig. 3: Town 1 and Town 2 layouts.

We simulate noisy maneuver detection (cf. Sec. III-D.2) by giving 0.9 probability to the current executed maneuver of the non-ego vehicle and the rest uniformly to other maneuvers. Prior probabilities over non-ego goals are uniform. A* computes up to two predicted trajectories for each non-ego vehicle and goal. MCTS is run at a frequency of 1 Hz, performs $K = 30$ simulations with a maximum search depth of $d_{max} = 5$, and uses $r_{coll} = r_{term} = -1$. We set $\lambda = 10$ for velocity smoothing (cf. Eq. (4)).

C. Results

1) Goal probabilities: Figure 4 shows the average probability over time assigned to the true goal in scenarios S1–S4. In all tested scenario instances we observe that the

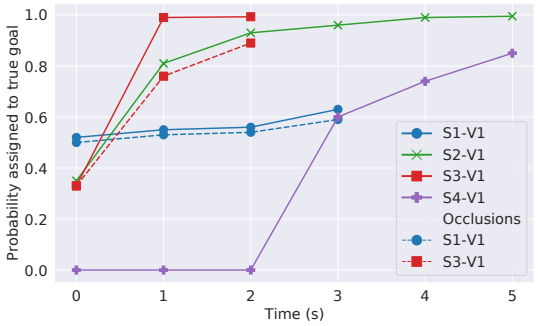


Fig. 4: Average probability given to true goal of selected vehicles in scenarios S1–S4. Note: lines for S1/S3 are shorter than indicated in Tab. II since possible vehicle goals change after exit points are reached and we only show lines for initial possible goals.

probability increases with growing evidence and at different rates depending on random scenario initialisation. Snapshots of goal probabilities (shown as bar plots) associated with the non-ego's most probable current maneuver can be seen in Figure 2. We also tested the method's robustness to missing segments in the observed trajectory of a vehicle. In scenarios S1 and S3 we removed the entire *lane-change* maneuver from the observed trajectory (but keeping the short lane-follow segment before the lane change). To deal with occlusion, we applied A* search before the beginning of each missing segment to reach the beginning of the next observed segment, thereby "filling the gaps" in the trajectory. Afterwards we applied velocity smoothing to the reconstructed trajectory. The results are shown as dashed lines in Figure 4, showing that even under significant occlusion the method is able to correctly recognise a vehicle's goal.

2) Driving times: Table II shows the average driving times required of each algorithm in scenarios S1–S4. Goal recognition enabled IGP2 and IGP2-MAP to reduce their driving times. **(S1)** All algorithms change lanes to avoid being slowed down by V1, leading to same driving times, however IGP2 and IGP2-MAP initiate the lane change before all other algorithms by recognising V1's intended goal. **(S2)** Cons waits for V_1 to clear the lane, which in turn must wait for V_2 to pass. IGP2 and IGP2-MAP anticipate this behaviour, allowing them to enter the road earlier. CVel and CVel-Avg wait for V_1 to reach near-zero velocity. **(S3)** IGP2 and IGP2-MAP are able to enter early as they recognise V_1's goal to exit the roundabout, while CVel, CVel-Avg, and Cons wait for V_1 to exit. **(S4)** Cons waits until V_1 decides to close the gap after which the ego can enter the road. IGP2 and IGP2-MAP recognise V_1's goal and merge in front.

IGP2-MAP achieved shorter driving times than IGP2 on some scenario instances (such as S3 and S4). This is because IGP2-MAP commits to the most-likely goal and trajectory of other vehicles, while IGP2 also considers residual uncertainty about goals and trajectories which may lead MCTS to select more cautious actions in some situations. The limitation of IGP2-MAP can be seen when simulating unexpected (irrational) behaviours in other vehicles. To test this, we compared IGP2 and IGP2-MAP on instances from S3 and S4 which were modified such that V1, after slowing down,

	S1	S2	S3	S4
IGP2	$5.97 \pm .02$	$7.24 \pm .05$	$8.54 \pm .05$	$10.83 \pm .03$
IGP2-MAP	$5.99 \pm .02$	$7.23 \pm .05$	$8.36 \pm .06$	$10.40 \pm .03$
CVel	$6.04 \pm .03$	$9.80 \pm .17$	$10.49 \pm .09$	$12.83 \pm .03$
CVel-Avg	$6.01 \pm .02$	$11.31 \pm .17$	$10.49 \pm .09$	$13.59 \pm .02$
Cons	$6.01 \pm .02$	$12.89 \pm .03$	$10.90 \pm .04$	$16.78 \pm .02$

TABLE II: Average driving time (seconds) required to complete scenario instances from S1–S4, with standard error.

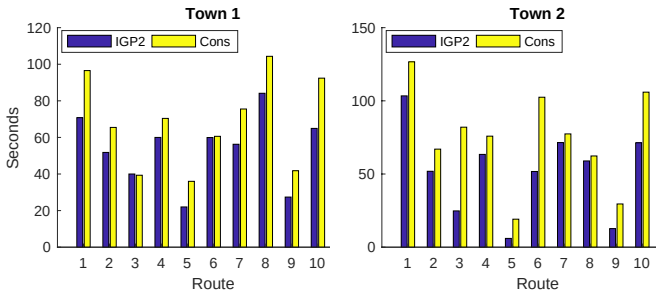


Fig. 5: Driving times (seconds) of IGP2 and Cons for 10 routes in Town 1 and Town 2.

suddenly accelerates and continues straight (rather than exiting as in S3, or stopping as in S4). In these cases we observed a 2-3% collision rate for IGP2-MAP (in all collisions, V1 collided into the ego) while IGP2 produced no collisions. These results show that IGP2 exhibits safer driving than IGP2-MAP by accounting for uncertainty over goals and trajectories.

Figure 5 shows the driving times of IGP2 and Cons for the routes in the two towns. Both algorithms completed all of the routes. Goal recognition allowed IGP2 to reduce its driving times substantially by exploiting multiple opportunities for proactive lane changes and road/junction entries. In contrast, Cons exhibited more conservative driving and often waited considerably longer at junctions or before taking a turn until traffic cleared up. SH-CVel was unable to complete any of the given routes, as its short planning horizon often caused it to take a wrong turn (thus failing the instance).

3) Interpretability: We are able to extract intuitive explanations for the predictions and decisions made by IGP2. The explanations are given in the caption of Figure 2.

V. Conclusion

We proposed an autonomous driving system, IGP2, which integrates planning and prediction over extended horizons by reasoning about the goals of other vehicles via rational inverse planning. Evaluation in diverse urban driving scenarios showed that IGP2 robustly recognises the goals of non-ego vehicles, resulting in improved driving efficiency while allowing for intuitive interpretations of the predictions to explain the system's decisions. IGP2 is general in that it uses relatively standard planning techniques that could be replaced with other techniques (e.g. POMDP-based planners [34]), and the general principles underlying our approach could be applied to other domains in which mobile robots interact with other robots/humans. Important future directions include goal recognition in the presence of occluded objects which can be seen by the non-ego vehicle but not the ego vehicle, and accounting for human irrational biases [35], [36].

REFERENCES

[1] W. Schwarting, J. Alonso-Mora, and D. Rus, "Planning and decision-making for autonomous vehicles," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 1, pp. 187–210, 2018.

[2] C. Dong, J. M. Dolan, and B. Litkouhi, "Smooth behavioral estimation for ramp merging control in autonomous driving," in *IEEE Intelligent Vehicles Symposium*. IEEE, 2018, pp. 1692–1697.

[3] C. Hubmann, J. Schulz, M. Becker, D. Althoff, and C. Stiller, "Automated driving in uncertain environments: Planning with interaction and uncertain maneuver prediction," *IEEE Transactions on Intelligent Vehicles*, vol. 3, no. 1, pp. 5–17, 2018.

[4] B. Zhou, W. Schwarting, D. Rus, and J. Alonso-Mora, "Joint multi-policy behavior estimation and receding-horizon trajectory planning for automated urban driving," in *IEEE International Conference on Robotics and Automation*. IEEE, 2018, pp. 2388–2394.

[5] E. Galceran, A. Cunningham, R. Eustice, and E. Olson, "Multipolicy decision-making for autonomous driving via changepoint-based behavior prediction: Theory and experiment," *Autonomous Robots*, vol. 41, no. 6, pp. 1367–1382, 2017.

[6] 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 (IV)*. IEEE, 2017, pp. 1671–1678.

[7] W. Song, G. Xiong, and H. Chen, "Intention-aware autonomous driving decision-making in an uncontrolled intersection," *Mathematical Problems in Engineering*, vol. 2016, 2016.

[8] B. D. Ziebart, N. Ratliff, G. Gallagher, C. Mertz, K. Peterson, J. A. Bagnell, M. Hebert, A. K. Dey, and S. Srinivasa, "Planning-based prediction for pedestrians," in *Intelligent Robots and Systems*, 2009, pp. 3931–3936.

[9] J. Hardy and M. Campbell, "Contingency planning over probabilistic obstacle predictions for autonomous road vehicles," *IEEE Transactions on Robotics*, vol. 29, no. 4, pp. 913–929, 2013.

[10] T. Bandyopadhyay, K. S. Won, E. Frazzoli, D. Hsu, W. S. Lee, and D. Rus, "Intention-aware motion planning," in *Algorithmic Foundations of Robotics X*. Springer, 2013, pp. 475–491.

[11] H. Zhao, J. Gao, T. Lan, C. Sun, B. Sapp, B. Varadarajan, Y. Shen, Y. Shen, Y. Chai, C. Schmid, C. Li, and D. Anguelov, "TNT: Target-driven trajectory prediction," in *Conference on Robot Learning*, 2020.

[12] N. Rhinehart, R. McAllister, K. Kitani, and S. Levine, "PRECOG: prediction conditioned on goals in visual multi-agent settings," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 2821–2830.

[13] Y. Chai, B. Sappm, M. Bansal, and D. Anguelov, "MultiPath: multiple probabilistic anchor trajectory hypotheses for behavior prediction," in *Conference on Robot Learning*, 2019.

[14] Y. Xu, T. Zhao, C. Baker, Y. Zhao, and Y. N. Wu, "Learning trajectory prediction with continuous inverse optimal control via Langevin sampling of energy-based models," *arXiv preprint arXiv:1904.05453*, 2019.

[15] S. Casas, W. Luo, and R. Urtasun, "IntentNet: learning to predict intention from raw sensor data," in *Conference on Robot Learning*, 2018, pp. 947–956.

[16] N. Lee, W. Choi, P. Vernaza, C. B. Choy, P. H. Torr, and M. Chandraker, "DESIRE: distant future prediction in dynamic scenes with interacting agents," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 336–345.

[17] M. Wulfmeier, D. Z. Wang, and I. Posner, "Watch this: Scalable cost-function learning for path planning in urban environments," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2016, pp. 2089–2095.

[18] A. Sadat, S. Casas, M. Ren, X. Wu, P. Dhawan, and R. Urtasun, "Perceive, predict, and plan: Safe motion planning through interpretable semantic representations," in *European Conference on Computer Vision*. Springer, 2020, pp. 414–430.

[19] S. V. Albrecht and P. Stone, "Autonomous agents modelling other agents: A comprehensive survey and open problems," *Artificial Intelligence*, vol. 258, pp. 66–95, 2018.

[20] J. Stewart, "Why people keep rear-ending self-driving cars," WIRED magazine, 2020, https://www.wired.com/story/self-driving-car-crashes-rear-endings-why-charts-statistics/ (Accessed: 2020-10-31).

[21] P. Koopman, R. Hierons, S. Khastgir, J. Clark, M. Fisher, R. Alexander, K. Eder, P. Thomas, G. Barrett, P. H. Torr, A. Blake, S. Ramamoorthy, and J. McDermid, "Certification of highly automated vehicles for use on UK roads: Creating an industry-wide framework for safety," 2019, FiveAI.

[22] M. Gadd, D. De Martini, L. Marchegiani, P. Newman, and L. Kunze, "Sense–Assess–eXplain (SAX): Building trust in autonomous vehicles in challenging real-world driving scenarios," in *2020 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2020, pp. 150–155.

[23] M. Ramírez and H. Geffner, "Probabilistic plan recognition using off-the-shelf classical planners," in *24th AAAI Conference on Artificial Intelligence*, 2010, pp. 1121–1126.

[24] C. Baker, R. Saxe, and J. Tenenbaum, "Action understanding as inverse planning," *Cognition*, vol. 113, no. 3, pp. 329–349, 2009.

[25] C. Browne, E. Powley, D. Whitehouse, S. Lucas, P. Cowling, P. Rohlfshagen, S. Tavener, D. Perez, S. Samothrakis, and S. Colton, "A survey of Monte Carlo tree search methods," *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 4, no. 1, pp. 1–43, 2012.

[26] S. V. Albrecht and S. Ramamoorthy, "Exploiting causality for selective belief filtering in dynamic Bayesian networks," *Journal of Artificial Intelligence Research*, vol. 55, pp. 1135–1178, 2016.

[27] C. Gámez Serna and Y. Ruichek, "Dynamic speed adaptation for path tracking based on curvature information and speed limits," *Sensors*, vol. 17, no. 1383, 2017.

[28] A. Wächter and L. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Mathematical Programming*, vol. 106, no. 1, pp. 25–57, 2006.

[29] S. Niekum, S. Osentoski, C. Atkeson, and A. Barto, "Online Bayesian changepoint detection for articulated motion models," in *IEEE International Conference on Robotics and Automation*. IEEE, 2015.

[30] P. Hart, N. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," in *IEEE Transactions on Systems Science and Cybernetics*, vol. 4, July 1968, pp. 100–107.

[31] P. Auer, N. Cesa-Bianchi, and P. Fischer, "Finite-time analysis of the multiarmed bandit problem," *Machine Learning*, vol. 47, no. 2-3, pp. 235–256, 2002.

[32] M. Treiber, A. Hennecke, and D. Helbing, "Congested traffic states in empirical observations and microscopic simulations," *Physical Review E*, vol. 62, no. 2, p. 1805, 2000.

[33] C. Watkins and P. Dayan, "Q-learning," *Machine Learning*, vol. 8, no. 3-4, pp. 279–292, 1992.

[34] A. Somani, N. Ye, D. Hsu, and W. S. Lee, "DESPOT: online POMDP planning with regularization," in *Advances in Neural Information Processing Systems*, 2013, pp. 1772–1780.

[35] M. Kwon, E. Biyik, A. Talati, K. Bhasin, D. P. Losey, and D. Sadigh, "When humans aren't optimal: Robots that collaborate with risk-aware humans," in *Proceedings of the ACM/IEEE International Conference on Human-Robot Interaction*, 2020.

[36] Y. Hu, L. Sun, and M. Tomizuka, "Generic prediction architecture considering both rational and irrational driving behaviors," in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*. IEEE, 2019, pp. 3539–3546.