Cooperative Planning for Autonomous Lane Merging

Shray Bansal[1], Akansel Cosgun[2] and Alireza Nakhaei[2]

***Abstract*— Autonomously performed lane merging as collaborative planning in a scenario with human drivers. Showed that planning to co-operate with the human significantly outperforms maximizing selfish reward in this domain.**

I. INTRODUCTION

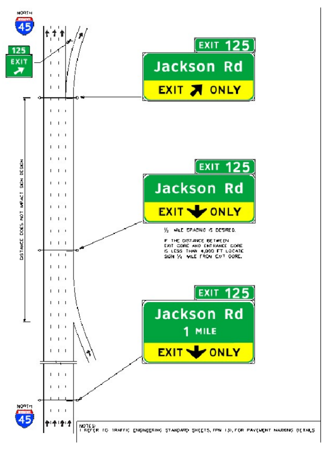
Merging into another lane is easy in isolation, however, traffic density and road design make the situation complex. For example, in Figure 1a there is only half a mile between the entry and the exit. This creates a scenario where cars exiting and entering have to negotiate with each other to safely navigate this lane merge in the time available. Humans do so by inferring intentions, driving styles, etc. of those around them and conditioning their behavior on that. This is in contrast to scenarios like lane-following or intersection navigation where one could get away with treating other cars like obstacles to avoid instead of agents to interact with.

Prior work in autonomous driving disregards the complexity of these agents, assuming them to be constant velocity obstacles, or controlled by simple rules. We take inspiration from recent work by Sadigh et al. [1] which grants agency to the other (human) driver and take into account their goals and actions. However, their self-interested Autonomous Vehicle (AV) only maximizes it's own reward, they assume also that the AV is able to choose its actions first. This works well in scenarios where the aims AV to influence the human's actions like making them slow down by moving in front of them but can be uncomfortable or unsafe in real-world situations. On the other hand, it is not uncommon for human drivers to slow down and let others into their lane. Behavioral economics also supports the argument that people are influenced by notions of fairness and reciprocity when making decisions, even at the expense of their own self-interest.

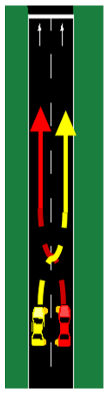
Our goal is to successfully navigate the Double Lane merging scenario autonomously in the presence of human drivers. Here, two cars start in adjacent lanes and must merge into each others' lanes in a limited road length (Figure 1b). Our hypothesis is that *an AV which considers the goals of other agents in addition to it's own selfish reward should benefit the human driver without adversely affecting itself.*

We simulated Double Lane merging and measured the performance of the AV with different levels of cooperation in the presence of people. Our analysis shows that having a more balanced reward, when planning for the AV, leads to better human performance. We also found that it positively affected the AV's performance as well.

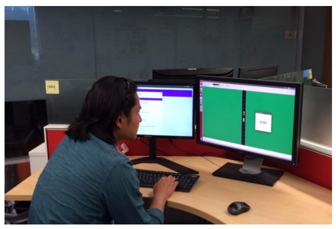
[1] Georgia Institute of Technology. `sbansal34@gatech.edu`
[2] Honda Research Institute (HRI), Mountain View, CA, USA. `{acosgun, anakhaei}@hri.com`



(a) Challenging Highway



(b) Problem Setup



(c) User controls the human car.

Fig. 1: (a) Highway in San Diego where the entry and exit are only 0.5 mile apart. (b) Double Lane Merge Scenario: where two cars start side-by-side with and have to merge into each other's lanes on a road of limited length.

II. PROBLEM

The goal is for the autonomous agent to safely move to it's target lane as quickly as possible in the presence of a human driver. We use the scenario of Figure 1b where either car changing lanes independent of the other can lead to a collision. So the challenge here is to take actions based upon the predicted behavior of the other car. We limit this investigation to the two car case, with one human (H) and the other robot (R) car.

Our world is a fully-observed dynamical system where the state x contains the position and speed of both cars. The state at the next time-step of the system is determined by applying the control for both human (u_H) and robot (u_R) according to the deterministic transition function T,

$$x_{t+1} = T(x_t, u_R, u_H). \tag{1}$$

The controls u_H, u_R are discrete and each agent has five available actions. (1) *accelerate* action increases the speed, while (2) *decelerate* decreases it by a constant factor, (3) *stay* maintains it, (4) *turn-left*, and (5) *turn-right* introduce a positive, and negative lateral component to the velocity while maintaining the speed.

III. APPROACH

Our approach involves planning for a finite time horizon to determine the best actions to execute at the present moment in time. The goal of our robot car is to choose a set of actions $\boldsymbol{u_R} = \{u_R^t\}_{t=0}^N$ that maximize a given reward function r_R over the fixed length of time N. The reward for the robot taking a set of actions, $\boldsymbol{u_R}$, starting at state x is given by

$$R_R(x, \boldsymbol{u_R}, \boldsymbol{u_H}) = \sum_{t=0}^{N-1} r_R(T(x_t, u_H^{(t)}, u_R^{(t)})). \tag{2}$$

Here, $\boldsymbol{u_H}$ are the actions to be taken by the human in this time horizon. So the optimal set of actions $\boldsymbol{u_R^*}$ from the initial state x_0 is,

$$\boldsymbol{u_R^*}(x_0) = \underset{\boldsymbol{u_R}}{\arg\max}\, R_R(x_0, \boldsymbol{u_R}, \boldsymbol{u_H}) \tag{3}$$

Ideally, $\boldsymbol{u_H}$ are *future* actions, which are unknown even to the human, as they depend upon the actions of the robot. Let us instead take $\boldsymbol{u_H}$ to be the human's current plan. Which means that to compute the optimal plan for the robot we need to estimate the current plan of the human. Let us assume that the human acts rationally to attain their goal, i.e. for a reward function R_H the human picks a plan, $\boldsymbol{u_H^*}$, to maximize it

$$\boldsymbol{u_H^*}(x_0) = \underset{\boldsymbol{u_H}}{\arg\max}\, R_R(x_0, \boldsymbol{u_H}, \tilde{\boldsymbol{u_R}}) \tag{4}$$

Here, $\tilde{\boldsymbol{u_R}}$ is the plan that the human expects the robot to follow. The actions that the human takes will then depend upon the predicted actions of the robot and the predicted robot actions depend upon the actions it predicts the human to take and so on. It was argued that this would lead to infinite regress in Sadigh et. al. [1] and so they assumed that the predicted human had access to the robot's plan and optimized its actions accordingly, i.e. $\tilde{\boldsymbol{u_R}} = \boldsymbol{u_R}$ in Equation 4. This means that in their model, the robot is aware of its influence on the human planning but the human believes that the robot's plan is fixed. This has the consequence of making the predicted human passive to the robot's plans if R_R is biased towards the robot's goals. Now, this can lead to two different situations, if the actual human complies with this assumption, the interaction would probably be uncomfortable for the human and if she does not, then it leads to the robot's prediction of the human's model becoming inaccurate which leads to conflicting or suboptimal plans for the two agents. For example, in the double merge scenario if the robot is rewarded for merging lanes quickly it may decide that the optimal plan for it is to start merging lanes immediately, forcing the human to slow down. However, if the human complied this forceful behavior is bound to make the human uncomfortable, if however, the human does not slow down then it leads to a kind of deadlock situation where both agents are trying to get ahead of each other or a collision because each believes that the other will act according to their wishes.

We take a different approach by assuming this two-agent game to be collaborative. This is due to the non zero-sum nature of this multi-agent gameIn it, to determine the optimal set of actions for the robot u_R^* we will maximize the joint reward $R_J = \alpha R_R + (1-\alpha)R_H$. with respect to the joint action space (u_R^*, u_H^*). α here is the co-operation factor which determines the relative importance of each reward in the collaboration. For example, $\alpha = 1$ is an aggressive robot with no consideration for the human and $\alpha = 0$ is the reverse, while $\alpha = 0.5$ equally considers the goal of both agents. We do not assume apriori knowledge of α and plan to study its effect with the user study. So the optimal plans for human and robot under this model can be found by

$$\boldsymbol{u_R^*}(x_0), \boldsymbol{u_H^*}(x_0) = \underset{\boldsymbol{u_R},\boldsymbol{u_H}}{\arg\max}\, R_J(x_0, \boldsymbol{u_R}, \boldsymbol{u_H}). \tag{5}$$

Here, the optimal human plan $\boldsymbol{u_H^*}(x_0)$ is only computed for the planning and we execute the first action from $\boldsymbol{u_R^*}(x_0)$. This planning is performed by an online search procedure described in the next subsection which is repeated after every time step in a discrete time simulation.

A. Finite Horizon A^* Search

The optimization from Equation 5 is implemented as a limited depth online a-star search. The nodes in this search are the states, where a state x contains both human and robot, $x = ((y_R, l_R, sl_R, v_R), (y_H, l_H, sl_H, v_H))$. Here, y, l, sl, v are the lateral position, the lane, sub-lane position, and velocity respectively, and the subscripts H and R refer to the human and robot cars. The search starts at the current state of the system and then simulates taking actions and the search is either terminated when all the unpruned states in the tree (of limited depth) are exhausted or if the time budget runs out. The time budget is set to be the length of a time-step in the simulation to keep the planning real-time. The action is chosen by taking the first action in the plan that led to the node with the highest accumulated reward. Rewards are defined on states as explained earlier and so the accumulated reward at a node is simply the sum of the rewards at each of the states leading to the node in the plan. In case there are multiple maximal reward nodes at the end of the search we randomly choose one of them and use it to control the robot car. Algorithm 1 gives pseudo code for our algorithm. In case we have to end the search prematurely, due to the time constraints, we pick the state with the highest reward, breaking ties randomly.

B. Reward

The goal of the robot is to reach its target lane (l_{goal}) while avoiding collisions and preferring to drive in the middle of the lane. We designed a simple reward to elicit this behavior,

$$R = \begin{cases} -10, & \text{if collision} \\ \delta e^{-|sl|} + (1-\delta), & \text{if } l = l_{goal} \\ 0, & \text{otherwise} \end{cases} \tag{6}$$

Here, a small constant $\delta = 0.3$ is used to make the agent favor the middle of the lane after it reaches the target

```
Data: state $x$
Result: $u_R^*$
1  start timer ;
2  initialize visited, front, rewards, as empty ;
3  f := heuristic(0) ;
4  push (f, state) to front ;
5  while front is not empty do
6      break if timer reached ;
7      (f, current) := pop node in front with highest value ;
8      break if f < maxReward ;
9      for each neighbor of current do
10         reward := get-reward(neighbor) ;
11         reward-total = reward + rewards[current] ;
12         continue if neighbor in visited and has less
             reward ;
13         if neighor is terminal state then
14             add neighbor to visited ;
15             update max-reward ;
16             continue ;
17         end
18         future-reward := reward-total + heuristic(t) ;
19         add (future-reward, neighbor) to front ;
20     end
21 end
22 return first action leading to node with maximal reward
     ;
```

Algorithm 1: Search(state)

```
Data: t $x$
Result: $R$
return t * max-reward ;
```

Algorithm 2: Heuristic

lane. The same reward was used for both R_R and R_H with different l_{goal} for each.

C. BASELINE

D. Implementation Details

The search was implemented in Python. For the simulation we used the open source Simulation of Urban MObility (SUMO) [2] software and its in-built visualizer to render the cars. This was a discrete-time simulation with a time-step of 0.2 seconds, which also served as the time-limit for the search algorithm to return an action. The cars were allowed a maximum lateral speed of $30m/s$ and a longitudinal speed of $3m/s$. With a lane width of $20m$ *check*, this meant that it took over six seconds to move from the center of one to the center of another even with empty lanes. Due to this, the time horizon of the search was kept at 6 seconds for interesting plans to develop. However, with a $0.2s$ time-step this would mean searching a tree of depth 30 and branching factor 5^2 which is infeasible in $0.2s$. To make this search faster we increased the time-step for the search to $1s$, which is equivalent to repeating each action for five time-steps. This, combined with our choice of data-structures enabled us to typically finish the search in less than $0.2s$. We plan to release our code.

IV. USER STUDY

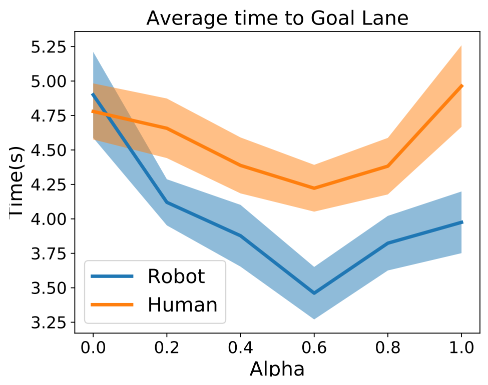
We conducted a user study involving 21 participants to test the performance of our approach with real humans as well as study the effect of the cooperation factor (α). There 4 female and 17 male participants, 19 of them had a valid driver's license, 20 were between 20 to 40 years of age, while 1 was over 40. Each participant was given a set of instructions explaining the simulation environment, the controls of the car, as well as their goal. Their goal was to safely navigate their car to the goal lane in the presence of another agent. After which we asked them some biographical questions and let them interact with the simulation environment with an agent that remained at a constant speed. This agent did not react to them, they were informed though that the agents in the practice environment may act differently from the real setting. After the participants felt comfortable controlling the car in simulation they were asked to perform 18 trials each taking about $30s$. They were presented with a small questionnaire after every trial which asked them about their interaction with the other agent. We asked them if they felt that the other agent was considerate of their goal and if they felt it acted safely. They were given three choices: "yes", "no" and "don't know". In case there was a collision they were also asked who was at fault.

This was a within-subject design where we sampled a cooperation factor uniformly at random from $\{0.0, 0.2, 0.4, 0.6, 0.8, 1.0\}$ and a road length from $\{100, 200\}m$. The subjects were informed prior what the road length would be but were not informed about the agent's reward or even if the agent would change. Other variations we made to the task we made were randomly sampling the color of the agent's car, starting lanes. This varied the stimulus for the human so that we did not lose their interest and thus, attention. We also sampled the starting speed of the agent car from a normal distribution with a mean of $15m/s$ and a standard deviation of $5m/s$ *check*.

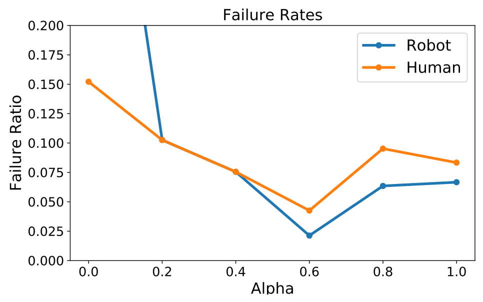
V. EXPERIMENTS AND ANALYSIS

We used the SUMO simulator [2] and the in-built visualizer to render the cars, the simulation time-step is 0.2 seconds. Our user-study involved 20 participants each performing 18 trials. They were instructed to safely drive the car to the goal lane with an AV that communicated its goal with blinkers. The cooperation factor (α) and road lengths were randomly sampled from $\{0.0, 0.2, 0.4, 0.6, 0.8, 1.0\}$, and$\{100, 200\}m$ respectively. We studied the effect of α, having around 50 trials for each, in Figure 2.

Figure 2a shows the average time it takes for a car to merge into it's goal lane for each α. Lower is better here and the minima is near the middle and is high at either end, implying the higher effectiveness of the *Fair* AV (FAV) with $\alpha = 0.6$. We compared the FAV to the Selfish AV (SAV) with $\alpha = 1.0$



(a)



(b)

Fig. 2: Results from the user study.

using an unpaired one-sided t-test and found that the merge-time was significantly lower for both the human ($p < 0.05$) and the AV ($p < 0.05$) in the case of the fairer vehicle. The SAV is equivalent to the agent proposed by Sadigh et al. [1] for our scenario. Since the FAV is more considerate towards the human's goal it improves the humans' ability to reach the goal. However, the FAV fairs better even for the robot, we attribute this to its possession of a more accurate model of the user's behavior. During planning, the SAV assumes that the human actively plans for the AV's goal and not her own. However, this is false and probably leads to it modifying its plan often.In Figure 2a, we also observe that the human is slower at lane merging. This can be explained by the fact that the robot's reward favors the fastest lane merge plan but the human has only wishes to merge before the road ends and is not incentivized to do it as soon as possible.

When calculating the average times above we only considered the successful trials, i.e. trials where the car was able to successfully merge into it's goal lane before the end of the road was reached. In Figure 2b we plot the Failure Rate, which is the ratio of unsuccessful trials to the total trials for each alpha. Again, we found that the FAV outperformed all others, including the SAV, which means that both human and robot were able to reach their goal lanes more often when the AV had a fair reward function.

VI. Related Work

Tie it to Robotics and HRI by explaining how the dynamics of the game are different. Summarize some collaborative robotics. Usually the robot here has the role of the helper, so in this case the goal is shared, although, it may not be fully observed. They model the human's behavior and structure the tasks, learning is sometimes applied. We use a very simple model for the human, which is not learned, but learning work is complementary to ours and we could replace our simple reward function with a learned one for better modeling of the human - use this to lead to IRL and LfD. Example and explain it. So most of this work is collaborative, however, in our case the two-agent game is not fully-cooperative as each agent posses their own reward which might conflict with the other agent. Find some HRI/ HRC work which is not fully-cooperative (shared reward).

Shortcoming of our work is the scalability. Reinforcement learning can solve it but faces issues with exploration and perhaps requiring a large training size in case of model-free RL. Also, this is only the case for single-agent RL which gets even harder in the presence of other agents. Also, the way to train multi-agent RL might be self-play but that may not be a good approximation for a human agent (Why?). Recent work has approached this problem by RL and their claim was that it is infeasible to solve the problem with planning, which is certainly true if we plan with low-level actions like steering angles. However, to simplify the learning process they use an option graph and put a lot of structure on the problem and it is not clear if after these steps planning would remain infeasible. It is difficult to compare with them since they proprietary software and have seemingly only tested their approach in self-play.

The fact that we do not know the reward of the human leads us to Inverse Reinforcement Learning. It would be useful and complementary if we could infer the rewards of the human actor. However, there is little work in multi-agent IRL and less so in cases that are not completely cooperative or competitive. Also, we believe that different people do not act in a similar manner when interacting with agents and so may not share the same reward function. This makes the already ill-posed IRL problem even harder.

VII. Conclusion

We formulated a planning framework with different co-operation levels and used it to control an autonomous car interacting with a human in simulation. We observed that in this mixed-autonomy set-up collaboration outperformed selfishness. In the future we would like to study this effect in other human-robot interaction domains.

References

[1] Sadigh, Dorsa, et al. ”Planning for Autonomous Cars that Leverage Effects on Human Actions.” Robotics: Science and Systems. 2016.
[2] Krajzewicz, Daniel, et al. ”Recent development and applications of SUMO-Simulation of Urban MObility.” IJOASM 2012.