Augmented Reinforcement Learning with Efficient Social-based Motion Prediction for Autonomous Decision-Making

Rodrigo Gutiérrez-Moreno[1], Carlos Gómez-Huelamo[1], Rafael Barea[1], Elena López-Guillén[1], Felipe Arango[1] and Luis M. Bergasa[1]

***Abstract*— This paper presents an approach that improves the efficiency and generalization capabilities of Reinforcement Learning-based autonomous vehicles operating in urban driving scenarios. The proposed method introduces an Efficient Social-based Motion Prediction module, which predicts the future positions of vehicles within the scenario. These predictions serve as input to a Reinforcement Learning-based Decision-Making module, responsible for executing high-level actions. The Proximal Policy Optimization algorithm is employed to develop our approach. We conduct experiments in an unsignalized T-intersection scenario using the SMARTS framework, comparing our approach with and without the proposed state representation, as well as against various baseline methods. Through this study, we demonstrate that our approach achieves performance improvements, particularly in scenarios involving high velocities. Our code and qualitative results are available at `https://github.com/Cram3r95/argo2goalmp`.**

I. Introduction

The increasing popularity of autonomous vehicles (AVs) has brought with it significant challenges in ensuring safe and effective decision-making, particularly in complex urban driving scenarios [1]. Reinforcement learning (RL) techniques have emerged as a promising solution to address these challenges [2]. They enable AVs to learn from their interactions with the driving environment, without relying on pre-defined rules. However, RL-based approaches still face a number of limitations that can hinder their development, including issues related to state representation.

A key challenge in RL-based AVs is the development of effective state representations that can account for the complexity of urban driving scenarios. Unlike in simpler environments, state representations for urban driving must be able to incorporate a wide range of information, including dynamic features of traffic flows and interactions among different agents. Finding ways to effectively encode this information and develop accurate state representations is essential to enabling RL-based AVs to generalize to various scenarios and make effective driving decisions. Distilling predictive information from scene representations can aid in the development of effective decision-making policies for AVs. By better understanding the potential consequences of different driving actions, RL-based AVs can make more informed decisions that lead to safer and more efficient driving behaviour.

This paper proposes an approach to enhance the efficiency and generalization of RL-based AVs in urban driving scenarios. Specifically, we introduce the use of a Motion Prediction (MP) module to obtain the future positions of the ego-vehicle and the surrounding vehicles (adversaries) in the scenario. These predictions are the input to an RL-based decision-making module that executes high-level actions. Our approach is developed using the Proximal Policy Optimization (PPO) algorithm [3]. We carry out an evaluation in the unsignalized T-intersection scenario shown in Fig. 1 with and without the proposed state representation and provide a comparison with some baseline methods.

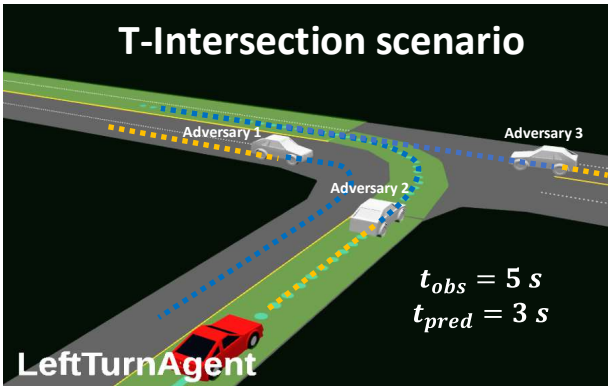

Fig. 1. Simulation environment. A visualization of the ego-vehicle driving in the T-intersection scenario. The past positions of the adversaries (**yellow**) and the predicted trajectories (**blue**) are represented in the scenario.

A. Related Works

In recent years, RL has emerged as a promising approach for developing decision-making policies for AVs [4], outperforming ruled-based approaches that usually cannot solve complex situations [5]. However, a large number of interactions with the environment are required to obtain the desired policy. This is why other approaches such as imitation learning [6] and inverse RL [7], based on human experts' behaviours are also used in the literature.

This work is focused on a critical issue for RL-based AVs, which is the state representation problem. Traditional state representations often focus on low-dimensional features such as distance to obstacles, lane positions, and vehicle velocities [8]. However, these representations may not be

*This work has been supported from the Spanish PID2021-126623OB-I00 project, funded by MICIN/AEI and FEDER, and TED2021-130131A-I00, PDC2022-133470-I00 projects, funded by MICIN/AEI and the European Union NextGenerationEU/PRTR.

[1]R. Gutiérrez-Moreno, C. Gómez-Huelamo, R. Barea, E. López-Guillén, F. Arango and L.M. Bergasa are with the Electronics Departament, University of Alcalá (UAH), Spain.{rodrigo.gutierrez, carlos.gomezh, rafael.barea, elena.lopezg, juanfelipe.arango, luism.bergasa}@uah.es

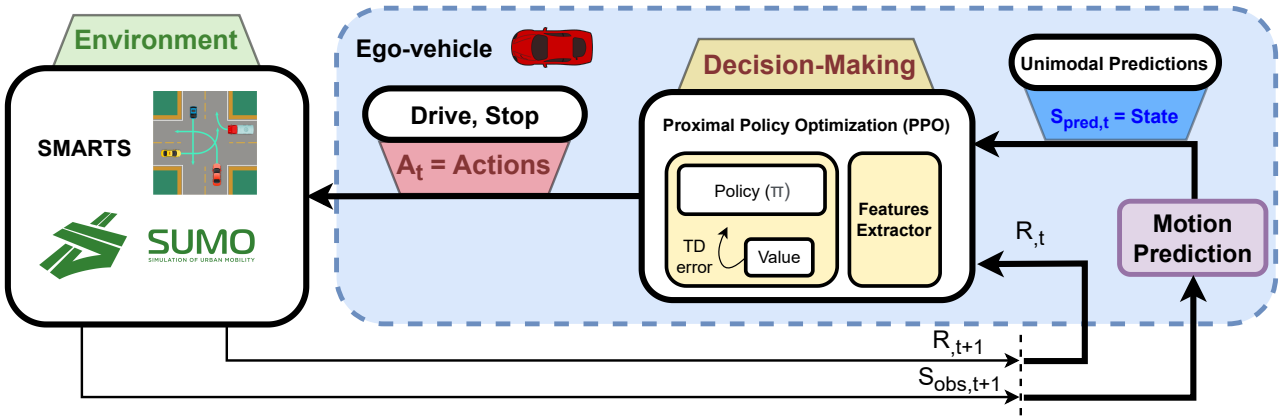

Fig. 2. An overview of the Augmented Reinforcement Learning with Efficient Social-based Motion Prediction for Autonomous Decision-Making. The observations (both position and ID, so, trackers) of the vehicles in the scenario are obtained from the simulator. The MP module estimates the future positions of these vehicles, taking into account the most plausible score of a multi-modal prediction. The decision-making module selects high-level actions based on this information. These actions are executed by the simulator, which provides a new state to the framework.

sufficient to capture the complex interactions among different agents and road structures in urban driving scenarios. To address the state representation problem, some methods have been proposed that use higher-dimensional or learned representations, such as convolutional neural networks [9] and recurrent neural networks [10]; other methods have been proposed to use more detailed representations, such as Bird-Eye-View images [11], image augmentation [12] or occupancy grids [13]. These methods have shown promising results in improving the generalization and robustness of the decision-making approaches. Recently, transformer-based approaches have gained increasing attention for their ability to capture long-term dependencies and interactions among different entities in sequential data. In the context of AVs, transformers have been used to reduce the computational load in end-to-end approaches [14] and anticipate future states with prediction-aware planning [15].

B. Contribution

The objective of this study is to illustrate the efficacy of employing a low-dimensional state representation in conjunction with an MP method. We aim to prove that the proposed framework can lead to good performance in urban scenarios. More specifically, we present the following contributions:

- The augmentation of RL techniques with MP to improve state representation. By predicting vehicle trajectories, we can better capture the complex interactions between different agents and road structures in urban driving scenarios.
- Higher explainability than end-to-end methods. Intermediate states are accessible in our approach. This can help to understand the decisions made.
- We provide a comparison with baseline methods in a standard scenario. We demonstrate that our approach leads to some improvements in performance, particularly in scenarios with high velocities.

II. Augmented Reinforcement Learning with Motion Prediction

The RL framework proposed in this work, which executes high-level decisions to solve urban driving scenarios, is represented in Fig. 2. The past observations of the position of adversaries are obtained from the environment. This information is provided to the MP module, which estimates future positions. The PPO algorithm takes these predictions and generates the decision-making output.

We propose two different learning processes: supervised learning for the MP module and an RL approach for the decision-making module. These two modules are trained separately, which allows access to the information of the predictions that feed the decision-making module.

A. Efficient Social-based Motion Prediction

Predicting the future behaviour of traffic agents [16] around the ego-vehicle is one of the key unsolved challenges in reaching full self-driving autonomy. In that sense, an Autonomous Driving Stack (ADS) can be hierarchically broken down into the following tasks: (i) perception, responsible for identifying what is around the vehicle, then tracking and predicting what will happen next, (ii) planning and decision-making, deciding what the ADS is going to do in the near future and (iii) control, that sends the corresponding low-level commands (brake, throttle and steering angle) to the vehicle.

This prediction must be multi-modal, which means that given the past motion of a particular vehicle and its surrounding scene, there may exist more than one possible future behaviour (also known as modes). Therefore, MP models need to cover the different choices a driver could make (i.e. going straight or turning, accelerations or slowing down) as a possible trajectory in the immediate future or as a probability distribution of the agent's future location. In other words, when an ADS attempts to make a specific action (e.g. left turn), it must consider the future motion of the other vehicles, since its own future actions (also known as decision-making

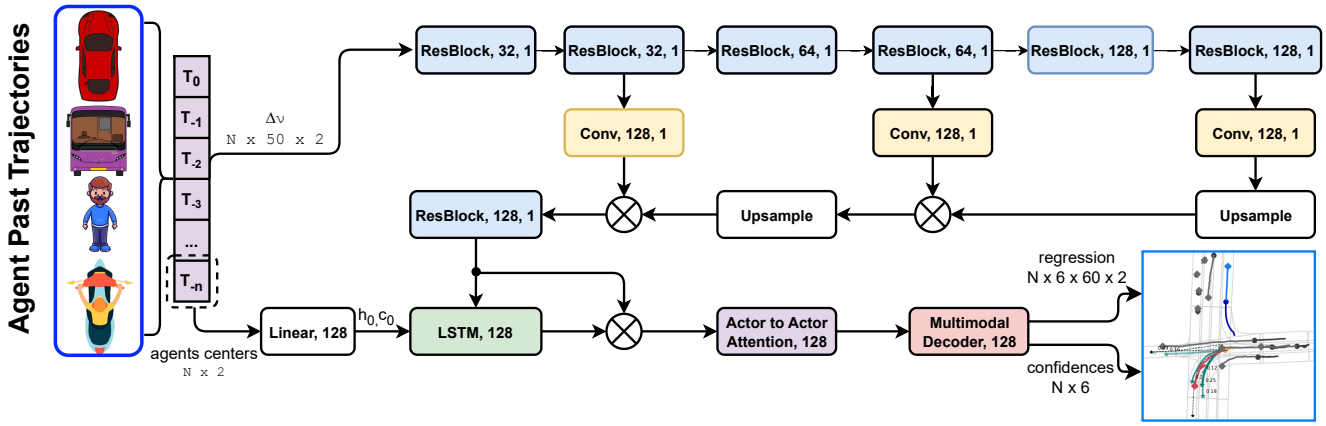

Fig. 3. Overview of our Efficient Social-based Motion Prediction. The main inputs are the relative displacements and centers (last observations) of the agents in the ego-vehicle frame. The relative displacements and centers are encoded through a sequence of Residual, Convolutional Blocks, LSTM and Attention modules. Finally, a multi-modal decoder based on residual blocks is used to predict *K* final future trajectories (modes) and their confidence scores.

or behaviour planning) depends on all possible manoeuvres of the other agents of the scene for safe driving.

Since our proposed pipeline (Fig. 2) is multi-stage to provide a more interpretable framework, we follow the principles of the Argoverse 2 Motion Forecasting dataset [16] to train our prediction model. In our case, we build an efficient model solely based on past trajectories (motion history, $obs_{len} = 50$) and agents interactions, taking into account the corresponding traffic rules, not requiring fully-annotated HD map information, to predict $obs_{len} = 60$ future steps.

B. Preprocessing

The SMARTS [17] framework provides only the positions of the agents in the timestamp *t*. Nevertheless, in order to predict the future $pred_{len}$ trajectories of the agents, we require their corresponding obs_{len} trackers over a certain set of observations. Most vehicle prediction datasets [16] aims to predict the future behaviour of a target agent assuming the surrounding agents have been detected and tracked (so, monitored over time) and the map information is also provided. In that sense, since SMARTS provide the agents in the same order for consecutive timestamps (that is, agent 5, unless it disappears from the scene, will be agent 5 again in the next frame), we are able to compute a FIFO (*First Input First Output*) for each agent, not requiring data association [18] to perform this task.

On top of that, as proposed by multiple methods [19], [20], we consider only the vehicles that are observable at *t=0*, handling those agents that are not observed over the full sequence spectrum (observation length = obs_{len} + prediction length = $pred_{len}$) by concatenating a binary flag b_i^t that indicates if the agent is padded or not. In particular, we filter the static elements and track fragments scored by Argoverse 2 to get only the most relevant traffic agents, reducing the number of agents to be considered in complex traffic scenarios. Furthermore, to make the model translation and rotation invariant, the coordinate system in our model is Bird-eye-view (BEV) centered on a given target agent at $t = 0$, and we use the orientation from the target location given in the same timestamp as the positive x-axis. Note that this representation will benefit the model to have a common representation to enhance the generalization of the model and prevent overfitting. Once the scene has been translated and rotated, instead of using absolute 2D-BEV (*xy* plane), the input for the agent *i* is a series of relative displacements:

$$\Delta \boldsymbol{\nu}_i^t = \boldsymbol{\nu}_i^t - \boldsymbol{\nu}_i^{t-1} \tag{1}$$

Where $\boldsymbol{\nu}_i^t$ represents the state vector (in this case, *xy* position of the agent *i* at timestamp *t*).

1) Efficient social encoding: Since our model focuses on an efficient encoding of social information, we base our model on the ActorNet backbone proposed by [19], as observed in Fig. 3. While both CNNs and RNNs can be used for temporal data, ActorNet uses a 1D CNN to process the trajectory input for its effectiveness in extracting multi-scale features and efficiency in parallel computing. The output is a temporal feature map, whose element at $t = 0$ is used as the actor feature. The network has 3 groups/scales of 1D convolutions. Each group consists of 2 residual blocks, with the stride of the first block as 2. Then, a Feature Pyramid Network (FPN) [21] is used to fuse the multi-scale features, and apply another residual block to obtain the output tensor.

On top of that, in a similar way to [22], the agents' centers (observations at $t = 0$) are encoded through a linear layer to initialize the hidden and cell vector of the LSTM layer, which processes the previously latent motion history. After that, a social attention block is used to compute the most representative agents' interaction.

2) Multi-modal decoder: Taking the final actor features after motion history and agents interaction, a multi-modal prediction header outputs the final motion forecasting. For each agent, it predicts K possible future trajectories and their confidence scores. The header has two branches, a regression branch to predict the trajectory of each mode and a classification branch to predict the confidence score of each mode.

For the m-th actor, a residual block and a linear layer in the regression branch to regress the K sequences of BEV coordinates are obtained:

$$O_{m,\text{reg}} = \{(\mathbf{p}^k_{m,1}, \mathbf{p}^k_{m,2}, ..., \mathbf{p}^k_{m,T})\}_{k\in[0,K-1]} \quad (2)$$

where $O_{m,\text{reg}}$ is the whole set of regressions and $\mathbf{p}^k_{m,i}$ is the predicted m-th actor's BEV coordinates of the k-th mode at the i-th time step.

On the other hand, for the classification branch, a Multi-Layer Perceptron (MLP) to $\mathbf{p}^k_{m,T} - \mathbf{p}_{m,0}$ to get K distance embeddings is applied. Finally, each distance embedding is concatenated with the actor feature, applying a residual block and a linear layer to output K confidence scores, $O_{m,\text{cls}} = (c_{m,0}, c_{m,1}, ..., c_{m,K-1})$.

In this particular work, we take the most plausible future trajectory for each agent (both the adversaries and the ego-vehicle) in the following timestamps: *t=0*, *t=10*, *t=20* and *t=30*, which correspond to the current position and the predicted position of the corresponding agent 1, 2 and 3 seconds in the future respectively. Even though we train our prediction model following the principles of Argoverse 2 (5s and 6s of observation and prediction respectively), given the velocities and traffic density of the experiments run in the SMARTS simulator (see Section III), we believe that predicting 3s in the future is enough for this purpose to evaluate the high-level actions of decision-making layer preventing overfitting.

C. Reinforcement Learning-based Decision Making

A Markov Decision Process (MDP) is a discrete-time stochastic control process that provides a mathematical framework for modelling decision-making environments. An MDP is a tuple (S, A, P, R) in which S is a set of states named state space, A is a set of actions named action space, P is the probability function and R is a reward function. An algorithm with a given state $s \in S$ takes an action $a \in A$ transitioning to s' with a probability $P(s, a, s')$, and getting a reward $R(s, a, s')$ as shown in Fig. 2. This algorithm iterates through this loop to learn a desired behaviour.

The goal in an MDP is to find a good policy for the decision-making system. The objective is to find the optimal policy $\pi*(s)$, that maximizes the cumulative function of the future reward.

We represent the driving scenario as an MDP to develop our decision-making module. We consider the output of the MP module as an input to this module. The state space, action space, and reward functions are defined in this section.

1) State Space: The state is defined by the predicted trajectories of the ego-vehicle and the five closest vehicles in the scenario.

$$s_t = (K^{ego}_t, K^1_t, ..., K^5_t) \quad (3)$$

where $K^i_t = (x^i_{t_0}, y^i_{t_0}, x^i_{t_1}, y^i_{t_1}, x^i_{t_2}, y^i_{t_2}, x^i_{t_3}, y^i_{t_3})$ contains the future estimations of the positions of the vehicles across a future horizon of three seconds. A representation of a state vector is shown in Fig 4, where the vehicles' predicted positions are represented.

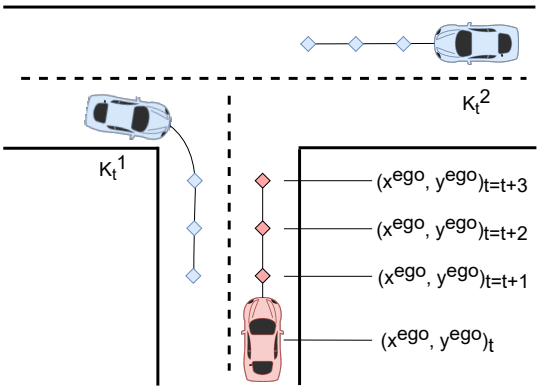

Fig. 4. The ego-vehicle (red) and the adversaries (blue) predicted positions in the next three seconds are represented.

2) Action Space: We propose a discrete action space formed by two actions. A low-level controller implemented by the simulator is in charge of performing smooth driving based on these actions. These actions are focused on the ego-vehicle velocity. The first action aims to reach a desired predefined velocity and the second action reduces the velocity until the vehicle stops. The action space is defined as:

$$a = (Drive, Stop) \quad (4)$$

3) Reward Function: The reward function is defined in terms of success or failure. A negative reward is given when there is a collision and a positive reward is given when the vehicle reaches the success point, situated at the end of the scenario.

$$r = k_v * v_{ego} + \begin{cases} 1 & if \quad success \\ -1 & if \quad collision \end{cases} \quad (5)$$

As shown in eq. 5, we add one more factor to the reward function to encourage the ego-vehicle to move. We propose a cumulative reward based on its longitudinal velocity. We use a constant k_v, small enough to ensure that the reward per episode is bounded between -1 and 1.

Our approach for the RL implementation (Fig. 5) builds upon our previous research [23], where we demonstrated that incorporating a feature extractor module to a PPO algorithm yields improved metrics and faster convergence.

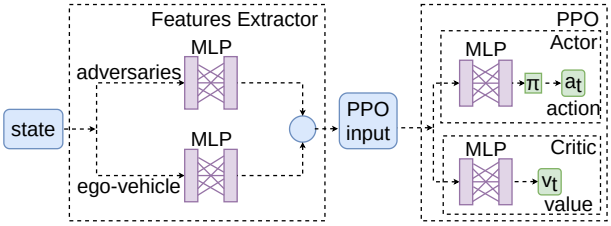

Fig. 5. The neural network architecture consists of two fully connected layers followed by the concatenation of both adversaries and ego vehicle features. The resulting concatenated features are then passed through an actor-critic structure, which comprises two layers, each containing 128 neurons.

In this implementation, we introduce separate feature extractors for adversaries and the ego-vehicle, which are

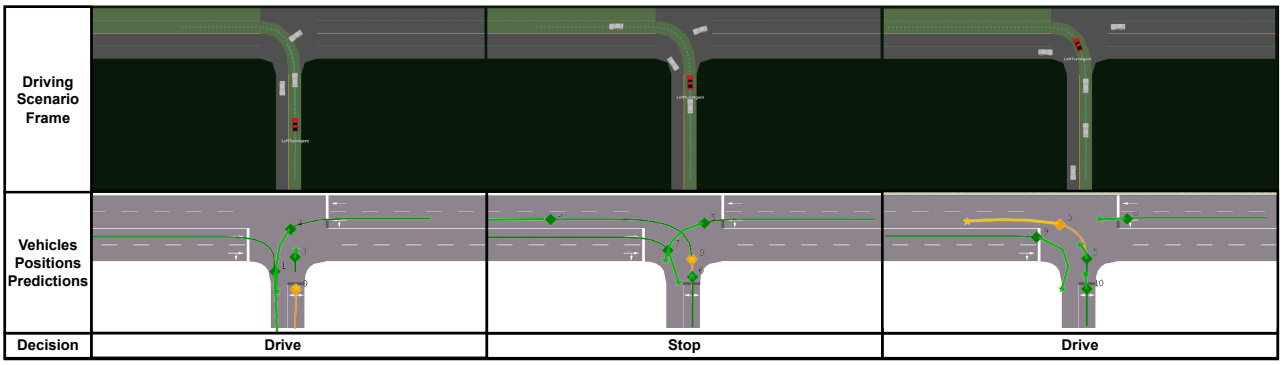

Fig. 6. Simulation overview of our system behaviour. The red car follows the green path, and each image represents a different frame of the simulation. We also show the predicted positions below each image and the actions taken by the decision-making module.

then concatenated into the input for the PPO algorithm. This algorithm consists of two models: the Actor, responsible for selecting an action based on the policy, and the Critic, which estimates the value function.

III. Experiments

A. Driving Scenario

To validate the performance of our approach, an intersection scenario is implemented in SMARTS, which is a SUMO [24] based simulation platform for research on autonomous driving.

The scenario is an urban unsignalized T-intersection. The objective is to execute a left turn maneuver in the absence of traffic signal protection, allowing the continuous flow of traffic. Fig. 1 illustrates the drivable area (highlighted in green) where the ego-vehicle can navigate to reach the target location. Simulations are reset under three conditions: 1) the ego-vehicle successfully reaches the target, 2) the episodic step surpasses the maximum time steps limit, and 3) the ego-vehicle collides or deviates from the drivable route.

We define different scenario configurations to test the performance of the proposed framework. First, the regular T-intersection scenario which is defined in SMARTS, where a random number of vehicles between [5-10] are spawned every minute, and the maximum velocity of these vehicles is 14 km/h. Then, we propose different configurations increasing the maximum velocity of the adversaries to 30, 60, and 90 km/h.

B. Evaluation Metrics

In decision-making, the success rate serves as a direct measure of the effectiveness of the RL agent in accomplishing the designated task. Besides, the average time of the episode is a common metric used in the literature. These metrics are defined as:

- $success\ [\%] = n_{success}/n_{success}$
- $t_e[s] = \sum t_n / n_{episodes}$

where the number of episodes n_e is 100 and simulation time is measured in seconds.

C. Results

To evaluate the performance of our approach we first present a comparison with the existing methods for decision-making in the literature and then an ablation study is conducted.

This first study compares the proposed approach with other existing methods for decision-making. The baseline methods used for comparison are Data-regularized Q-learning (DrQ) [12], Soft Actor-Critic (SAC) [25], and PPO. These methods have different features and serve as reference points for evaluating the proposed approach. The results presented in Table I demonstrate a higher success rate of our proposal.

TABLE I. A comparison of the proposed framework against the existing baselines in the T-intersection scenario. The success rate S[%] and the average episode time t_e are presented.

Metric	Ours	PPO	SAC	DrQ
S[%]	**80**	70	68	78
t_e(s)	22.3	36.4	19.2	18.2

Two ablative studies are carried out to see how the use of MP in the state representation can improve the performance of the framework. The first approach is to use just the position of the vehicles as the input to the decision-making module and the second approach is to use the locations over the past five seconds. We test the three approaches under the previously introduced configurations with different adversaries' velocities, from 15km/h to 90km/h. To correctly evaluate the performance of the decision-making system we propose different metrics that aim to provide a better comprehension of the behaviour. We believe that the success rate is still a good indicator, but we slightly modify the average time, only considering the successful episodes to calculate this metric. In addition, we include a new relevant metric: the average ego-vehicle velocity when a collision takes place v_c.

The results presented in Table II show that the use of the predicted positions in the state vector avoids more collisions as the velocities increase. Besides, the average collision velocity and the average time to complete the scenario are lower for the proposed approach.

TABLE II. An ablation study comparing three state representations with the different scenario configurations: Current positions, Past positions, and Future positions. The success rate S[%], the episode time t_e in these successful episodes, and the average velocity of collision v_c are presented.

	Metric	15 km/h	30 km/h	60 km/h	90 km/h
Future	S [%]	80	78	78	77
	t_e (s)	22.3	23.3	23.4	23.3
	v_c (km/h)	4.9	5.1	5.6	5.6
Current	S [%]	77	73	70	70
	t_e (s)	25.1	23.4	23.4	23.3
	v_c (km/h)	5.1	5.5	6.1	6.2
Past	S [%]	78	75	72	71
	t_e (s)	24.2	23.3	23.2	23.1
	v_c (km/h)	4.9	5.2	5.9	6.0

Finally, an overview of the behaviour of our system is shown in Fig. 6. The ego-vehicle in red follows the trajectory defined in green. Each image represents a different frame of the simulation and the respective predictions of the positions are displayed below. Besides, the action executed by the decision-making module for each frame is shown.

IV. Conclusions and Future Works

The proposed method incorporates an Efficient Social-based MP module, which predicts the future positions of vehicles within the scenario. These predictions improve a Reinforcement Learning-based Decision Making module. The results of the study demonstrate that our approach achieves significant performance improvements, particularly in scenarios involving high velocities.

This research opens up several potential directions for future work. The proposed approach can be evaluated in a broader range of urban driving scenarios to assess its robustness and scalability, up-to-pair with the difficulty of the Argoverse 2 dataset scenarios (especially in terms of intersections or lane change behaviours at high speed) where multi-modal predictions with higher prediction horizons will be required. Furthermore, the Reinforcement Learning-based Decision Making module can be enhanced by exploring advanced algorithms.

References

[1] E. Yurtsever, J. Lambert, A. Carballo, and K. Takeda, "A survey of autonomous driving: Common practices and emerging technologies," *CoRR*, vol. abs/1906.05113, 2019.

[2] B. R. Kiran, I. Sobh, V. Talpaert, P. Mannion, A. A. A. Sallab, S. K. Yogamani, and P. Pérez, "Deep reinforcement learning for autonomous driving: A survey," *CoRR*, vol. abs/2002.00444, 2020.

[3] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *CoRR*, vol. abs/1707.06347, 2017.

[4] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, pp. 529–533, Feb. 2015.

[5] L. Zhu, F. R. Yu, Y. Wang, B. Ning, and T. Tang, "Big data analytics in intelligent transportation systems: A survey," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 1, pp. 383–398, 2019.

[6] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis, "Mastering the game of Go with deep neural networks and tree search," *Nature*, vol. 529, pp. 484–489, jan 2016.

[7] S. Ross, G. J. Gordon, and J. A. Bagnell, "No-regret reductions for imitation learning and structured prediction," *CoRR*, vol. abs/1011.0686, 2010.

[8] R. Gutiérrez-Moreno, R. Barea, E. López-Guillén, F. Arango, N. Abdeselam, and L. M. Bergasa, "Hybrid decision making for autonomous driving in complex urban scenarios," in *2023 IEEE Intelligent Vehicles Symposium (IV)*, 2023.

[9] C. Hoel, K. Wolff, and L. Laine, "Automated speed and lane change decision making using deep reinforcement learning," *CoRR*, vol. abs/1803.10056, 2018.

[10] T. Tram, A. Jansson, R. Grönberg, M. Ali, and J. Sjöberg, "Learning negotiating behavior between cars in intersections using deep q-learning," *CoRR*, vol. abs/1810.10469, 2018.

[11] Z. Zhang, A. Liniger, D. Dai, F. Yu, and L. V. Gool, "End-to-end urban driving by imitating a reinforcement learning coach," 2021.

[12] I. Kostrikov, D. Yarats, and R. Fergus, "Image augmentation is all you need: Regularizing deep reinforcement learning from pixels," 2021.

[13] M. Moghadam and G. H. Elkaim, "A hierarchical architecture for sequential decision-making in autonomous driving using deep reinforcement learning," 2019.

[14] G. Li, Y. Qiu, Y. Yang, Z. Li, S. Li, W. Chu, P. Green, and S. E. Li, "Lane change strategies for autonomous vehicles: A deep reinforcement learning approach based on transformer," *IEEE Transactions on Intelligent Vehicles*, vol. 8, no. 3, pp. 2197–2211, 2023.

[15] R. Valiente, M. Razzaghpour, B. Toghi, G. Shah, and Y. P. Fallah, "Prediction-aware and reinforcement learning based altruistic cooperative driving," 2022.

[16] B. Wilson, W. Qi, T. Agarwal, J. Lambert, J. Singh, S. Khandelwal, B. Pan, R. Kumar, A. Hartnett, J. K. Pontes, *et al.*, "Argoverse 2: Next generation datasets for self-driving perception and forecasting," *arXiv preprint arXiv:2301.00493*, 2023.

[17] M. Zhou, J. Luo, J. Villella, Y. Yang, D. Rusu, J. Miao, W. Zhang, M. Alban, I. Fadakar, Z. Chen, A. C. Huang, Y. Wen, K. Hassanzadeh, D. Graves, D. Chen, Z. Zhu, N. Nguyen, M. Elsayed, K. Shao, S. Ahilan, B. Zhang, J. Wu, Z. Fu, K. Rezaee, P. Yadmellat, M. Rohani, N. P. Nieves, Y. Ni, S. Banijamali, A. C. Rivers, Z. Tian, D. Palenicek, H. bou Ammar, H. Zhang, W. Liu, J. Hao, and J. Wang, "Smarts: Scalable multi-agent reinforcement learning training school for autonomous driving," 11 2020.

[18] H. W. Kuhn, "The hungarian method for the assignment problem," *Naval research logistics quarterly*, vol. 2, no. 1-2, pp. 83–97, 1955.

[19] M. Liang, B. Yang, R. Hu, Y. Chen, R. Liao, S. Feng, and R. Urtasun, "Learning lane graph representations for motion forecasting," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*, pp. 541–556, Springer, 2020.

[20] C. Gómez-Huélamo, M. V. Conde, R. Barea, and L. M. Bergasa, "Improving multi-agent motion prediction with heuristic goals and motion refinement," in *CVF Conference on Computer Vision and Pattern Recognition*, IEEE, 2023.

[21] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2117–2125, 2017.

[22] M. Wang, X. Zhu, C. Yu, W. Li, Y. Ma, R. Jin, X. Ren, D. Ren, M. Wang, and W. Yang, "Ganet: Goal area network for motion forecasting," *arXiv preprint arXiv:2209.09723*, 2022.

[23] R. Gutiérrez-Moreno, R. Barea, E. López-Guillén, J. Araluce, and L. M. Bergasa, "Reinforcement learning-based autonomous driving at intersections in carla simulator," *Sensors*, vol. 22, no. 21, 2022.

[24] M. Behrisch, L. Bieker, J. Erdmann, and D. Krajzewicz, "Sumo - simulation of urban mobility: An overview," in *in SIMUL 2011, The Third International Conference on Advances in System Simulation*, pp. 63–68, 2011.

[25] T. Haarnoja, A. Zhou, K. Hartikainen, G. Tucker, S. Ha, J. Tan, V. Kumar, H. Zhu, A. Gupta, P. Abbeel, and S. Levine, "Soft actor-critic algorithms and applications," 2019.