MaP, CaP, RaLly! Hybrid Architecture for Planning and Control of UGV in Stochastic Environments

Nathaniel Snyder
Dept. of MAE
UCLA
natsnyder1@gmail.com

Brian Wang
NESL, Dept. of CS
UCLA
wangbri1@g.ucla.edu

Swapnil Sayan Saha
NESL, Dept. of ECE
UCLA
swapnilsayan@g.ucla.edu

1 Problem Statement

Real-time trajectory management in unmanned ground vehicles (UGV) in the presence of a large number of stochastic and dynamic obstacles entails computational and implementation complexities [1]. The computational elements within any UGV framework fulfill three main purposes: mapping, trajectory planning, and control / state estimation. Traditional approaches of mapping, planning and control of UGV in stochastic environments result in human engineered object-detector functions dealing with high-dimensional observation-space without probabilistic guarantees of fulfilling the end goal [1]. On one side, reactive navigation strategies, which infer the local costmap of obstacles from sensor information in real-time to continuously update waypoints, suffer from sub-optimal and understated trajectory projections. On the other hand, deliberate navigation strategies (pre-planned) are more likely to converge to global optima but are unable to handle dynamic deviations in a priori environmental belief states [2]. As a result, several reinforcement learning (RL) techniques (populated in [3]) have emerged and touted as promising solutions to the aforementioned challenges for trajectory planning and control in stochastic environments. However, end-to-end RL agents are difficult to train as they suffer from reward sparsity, tedious training phase, unpredictable / unsafe agent behavior, hyperparameter sensitivity, low generalizability and divergence in large and complex maps [1][4][5], constricting their superlative performance characteristics over the short and task-specific domain. As a result, we hypothesize **Ma**P, **Ca**P, **Ra**Lly, an end-to-end hybrid UGV controller using a **M**odel **P**redictive **C**ontrol (MPC) oracle, which exploits a LiDAR Simulataneous Localization and Mapping (SLAM) to plan, optimize and fulfill user-defined navigation goals, while using a **R**einforcement **L**earning agent to handle dynamic obstacles in the environment. MaP, CaP RaLly combines elements of control theory and imitation learning for dynamically avoid moving obstacles in an indoor warehouse environment, enabling the UGV to effectively and robustly navigate stochastic environments using RL under the guidance of a near-optimal MPC oracle.

2 Experimental Testbed and Algorithmic Implementation

Figure 1 shows the implemented experimental testbed and the UGV controller design. We have designed a full-stack Ackermann-drive UGV architecture equipped with LiDAR and camera and capable of trajectory planning and control in ROS-Gazebo framework. To achieve near-optimal path planning (unavailable for Ackermann-drive robots in ROS), a custom MPC trajectory optimizer node has been developed that tunes the output of classic path-planning algorithms (e.g. Dijisktra) to a near-optimal yet feasible path given the environmental and mechanical constraints. The path and desired controls are then fed into a custom error-tracking finite horizon LQR, which takes the vehicle to the goal point along the planned path in real-time. This framework is intended to bootstrap the initial phases of learning for the RL-agent (shown in purple), including data collection, training and deployment phases. UGV actions (continuous) include steering ($-\frac{\pi}{6}^c$ to $\frac{\pi}{6}^c$) and throttle (-0.25 m/s to 1.25 m/s), while the continuous observation space includes the current location of the car $\{(x_t, y_t, \theta_t), x, y \in \mathbb{R}, \theta \in [-\pi, \pi]\}$, LiDAR scan $L_v \in \mathbb{R}^{1080}$, goal point (x_f, y_f, θ_f), dynamics

error $\hat{e}$, waypoints from MPC oracle and priori belief states from SLAM (static boundaries) on a $4m \times 4m$ costmap. Various sets of these signals can be concatenated to represent the agent's state-space during training phase. A video demo of the framework is available at: `https://drive.google.com/file/d/1v34tkVi5bCUgJxQJNlpyA0adLCOj48J6/view?usp=sharing`

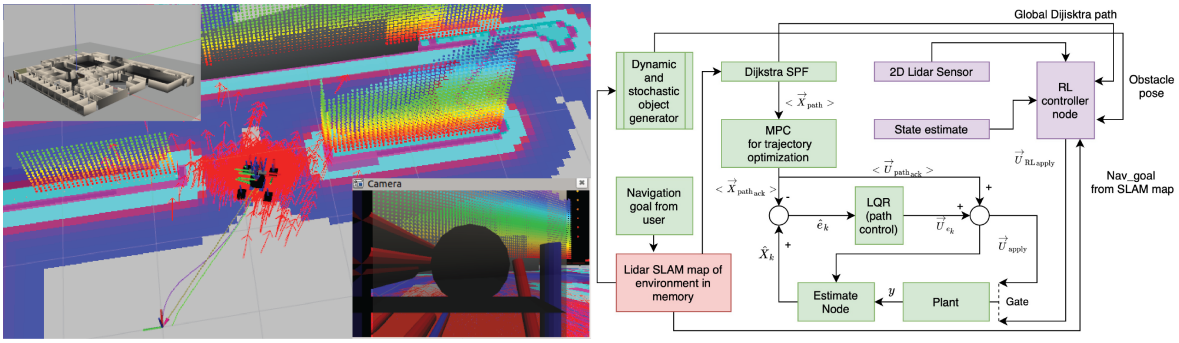

Figure 1: (Left) Implemented ROS-Gazebo testbed showing a UGV traversing through a warehouse; insets show the camera feed and warehouse map. (Right) Overview of the UGV control system (red: priori belief states from SLAM; green: deployment phase; purple: RL agent (during deployment).

For the RL node, we plan to evaluate the performance of Q-learning (state-of-the-art for collision avoidance [3]) on discretized state-action spaces or an actor-critic method for continuous spaces [1]. For training, we will adopt the imitation learning (IL) strategy from [6], teaching the agent how to follow projections. A stochastic online oracle [6][7] will be adopted to further explore the state space with agent in the loop. Afterwards, the agent will be trained in the presence of obstacles, with the objective of minimizing the L2 norm between MPC projection (near-optimal) and RL projection (safe). We allow oracles to intervene in case of sub-optimal choices. The online oracle may include the LQR tracker and an LQR obstacle avoider, as well as a user with a joystick. Lastly, we eliminate all bootstrappers, introduce a reward function and let the agent play itself. We postulate that the resulting controller, motivated from [4], should be robust to uncertainties in the environment while achieving near-optimal trajectory projections to the end goal.

3 Evaluation of Results

The metrics for results evaluation for each algorithm we implement include plots for the mean and variance of trajectory length, trajectory duration, obstacle clearance, average speed (as well as L2 norm of these quantities between optimal and agent projection) and goal completion ratio (a collision entails a failure), consistent with [4][6][8]. Probable baselines include purely reactive and deliberate controllers, as well as other formal, machine learning and RL methods populated in literature.

4 Related Work

Classical reactive and deliberate strategies for handling dynamic obstacles in real-time include traveling salesman /s hortest path variants, velocity-obstacle, probabilistic / tree-search methods, non-linear optimization, integer programming, artificial potential fields, receding horizon optimization, dynamic window and stochastic reachability [1][2][9][10]. Empirically, various RL techniques exploiting Q-learning [3][6] and actor-critic methods [1][2][4][8] have been shown to exhibit superior performance characteristics over formal methods [1][4][6][8] with smaller dependence on belief states, but are often unable to guarantee convergence to optimal trajectories in the long run. As a result, a small number of articles attempt to illustrate the benefits of coupling the ability of RL to handle uncertainties with the convergence guarantees of formal methods (e.g. MPC or PRM) in the context of intelligent transportation and robot locomotion, notably [4][8][9][11][12][13]. These approaches combine RL and classical methods for managing actions and making dynamic decisions to reach the end goal while at the same time putting constraints on action space in terms of safety, comfort and fuel economy.

References

[1]. A. Garg, H. L. Chiang, S. Sugaya, A. Faust and L. Tapia, *Comparison of Deep Reinforcement Learning Policies to Formal Methods for Moving Obstacle Avoidance,* 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Macau, China, 2019, pp. 3534-3541.

[2]. E. Meyer, H. Robinson, A. Rasheed and O. San, *Taming an Autonomous Surface Vehicle for Path Following and Collision Avoidance Using Deep Reinforcement Learning,* in IEEE Access, vol. 8, pp. 41466-41481, 2020.

[3]. M. S. Shim and P. Li, *Biologically inspired reinforcement learning for mobile robot collision avoidance,* 2017 International Joint Conference on Neural Networks (IJCNN), Anchorage, AK, 2017, pp. 3098-3105.

[4]. A. Faust et al., *PRM-RL: Long-range Robotic Navigation Tasks by Combining Reinforcement Learning and Sampling-Based Planning,* 2018 IEEE International Conference on Robotics and Automation (ICRA), Brisbane, QLD, 2018, pp. 5113-5120.

[5]. B. Lutjens, M. Everett and J. P. How, *Safe Reinforcement Learning With Model Uncertainty Estimates,* 2019 International Conference on Robotics and Automation (ICRA), Montreal, QC, Canada, 2019, pp. 8662-8668.

[6]. Y. Pan et al. *Agile Autonomous Driving using End-to-End Deep Imitation Learning.* Robotics: Science and Systems (RSS). Pittsburgh, PA, USA, 2018.

[7]. C-A Cheng et al. *Accelerating Imitation Learning with Predictive Models.* Proceedings of Machine Learning Research 89: 3187-3196 (2019).

[8]. A. Francis et al., *Long-Range Indoor Navigation With PRM-RL,* in IEEE Transactions on Robotics (2020).

[9]. C. Greatwood and A. G. Richards. *Reinforcement learning and model predictive control for robust embedded quadrotor guidance and control.* Autonomous Robots 43.7 (2019): 1681-1693.

[10]. S. R. Devaragudi and B. Chen. *MPC-Based Control of Autonomous Vehicles With Localized Path Planning for Obstacle Avoidance Under Uncertainties."*International Design Engineering Technical Conferences and Computers and Information in Engineering Conference. Vol. 59292. American Society of Mechanical Engineers, 2019.

[11]. N. K. Ure, M. U. Yavas, A. Alizadeh and C. Kurtulus, *Enhancing Situational Awareness and Performance of Adaptive Cruise Control through Model Predictive Control and Deep Reinforcement Learning,* 2019 IEEE Intelligent Vehicles Symposium (IV), Paris, France, 2019, pp. 626-631.

[12]. T. Tram, I. Batkovic, M. Ali and J. Sjöberg, *Learning When to Drive in Intersections by Combining Reinforcement Learning and Model Predictive Control,* 2019 IEEE Intelligent Transportation Systems Conference (ITSC), Auckland, New Zealand, 2019, pp. 3263-3268.

[13]. T. Koller, F. Berkenkamp, M. Turchetta and A. Krause, *Learning-Based Model Predictive Control for Safe Exploration*, 2018 IEEE Conference on Decision and Control (CDC), Miami Beach, FL, 2018, pp. 6059-6066.