Learning-based NMPC Framework for Car Racing Cinematography Using Fixed-Wing UAV

Dev Soni, Amith Manoharan, Prakrit Tyagi and P.B. Sujit

***Abstract*—A learning-based nonlinear model predictive control (L-NMPC) scheme is designed for the iterative task of filming a race-car using a gimbaled camera mounted on a fixed-wing autonomous aerial vehicle (AAV). The controller is capable of avoiding the environmental obstacles that block the path of the AAV. It also ensures that the car always lies in the field of view (FOV) of the camera while satisfying the control and state constraints. The controller is able to learn from the previous iterations and improve the tracking performance with the help of reinforcement learning (RL). Simulation results are given to demonstrate the efficacy of the proposed learning-based control scheme.**

***Index Terms*—Iterative learning control, reinforcement learning, model predictive control, online learning, target tracking.**

I. Introduction

Iterative learning control (ILC) is the control strategy that allows the controller to learn from the previous iteration and increase the future performance of the system while rejecting periodic disturbances. ILC can be applied to systems that repeat certain tasks over time, and each repetition is known as an "iteration" or a "trial" where the system starts at the same initial conditions. ILC is exhaustively studied in literature [1]–[3].

ILC follows a model-free control strategy which inherently has limitations on constraint handling. Combining ILC with a model-based approach such as nonlinear model predictive control (NMPC) can compensate for the unavailability of the accurate model of the system or unmeasured disturbances and increases the performance of the system while respecting the imposed control and state constraints. The concept of integrating the ILC with MPC has been explored in [4], where the author proposed a control technique, called batch-MPC (BMPC) for a time-varying MIMO system. The effectiveness of the approach was shown through experimental results. In [5], Lee and Lee proved that the tracking error of the BMPC converges to zero as the number of iterations increases. In [6], a nonlinear model predictive controller based on iterative learning control was proposed, where the NMPC is designed for disturbance rejection and the ILC to minimize the errors occurring at each iteration. The authors proved that the steady-state tracking error converges to zero as the number of iterations goes to infinity. Artificial intelligence techniques such as model-free reinforcement learning (RL) are considered as one of the best options to learn from the previous iterations [7]. The authors proposed an approach where RL was used to learn the length of the prediction horizon of MPC and shows the increase in resulting performance. In the works [8] and [9], authors used the RL technique to tune the weights of the MPC objective function.

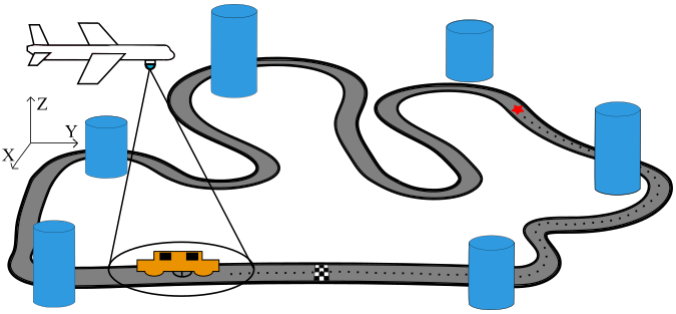


Fig. 1: Abstract representation of a race track showing a car filmed by an AAV.

In this paper, we are interested in the repetitive task of filming an autonomous robot-race-car with the use of a gimbaled fixed-wing autonomous aerial vehicle (AAV), as illustrated in Fig. 1. Competitions involving robots are becoming increasingly familiar with hobbyists, students, and researchers for demonstrating cutting-edge technology or entertainment. The races are conducted in indoor and outdoor settings with varying degrees of difficulty. We are interested in the outdoor races where the tracks are situated in large metropolitan cities. This kind of environment makes the aerial videography of the race very difficult. The cityscape cluttered with buildings and other infrastructures create obstructions for the free movement of the AAV and, in some cases, occludes the camera field of view (FOV). This problem can be modeled as a target-tracking problem with obstacle avoidance, where the objective is to minimize the tracking errors while respecting the imposed physical constraints.

We assume that the car and the AAV start at the same initial conditions for all-iterations (laps of the race), and the car follows the same trajectory for all iterations. The control, state, and gimbal parameters are constrained to mimic a realistic

Dev Soni is an undergraduate student in the Department of Mechatronics Engineering at ITM Vocational University, Gujarat, India 2018btechmdevs@itmvu.in.

Amith Manoharan is with the Department of Electronics and Communication Engineering, Indraprastha Institute of Information Technology Delhi, New Delhi, India amithm@iiitd.ac.in.

Prakrit Tyagi is a research assistant at the Department of Electrical and Computer Science at Indian Institute of Science Education and Research Bhopal, Bhopal, India tyagiprakrit@gmail.com.

P.B. Sujit is with the Department of Electrical Engineering and Computer Science at Indian Institute of Science Education and Research Bhopal, Bhopal, India sujit@iiserb.ac.in.

scenario. We propose the use of NMPC for control since it can easily handle state and input constraints. Tuning the weights for a multi-objective cost function is a tedious task. Hence, the RL technique is used to learn the weights of the components of the cost function. The combined RL-NMPC scheme, which we refer to as learning-based NMPC (L-NMPC) from here onward, results in precise tracking of the car due to the perfect balance of the weights in the objective function.

Previous studies on the learning based MPC used offline training to tune the MPC parameters [8], [9]. Also, the RL was used to learn a constant set of weights, which then will be used for the entire duration of the actual mission. We propose to use time-varying weights, which will give the perfect balance for the cost function, and we learn these weights on the fly, i.e., the training takes place online in the actual mission, and after each iteration, the performance of the controller increases and the weights converges to the optimal value over time. We identify this advancement as the main contribution of this paper.

The rest of the paper is organized as follows. The related works are given in Section II. The L-NMPC scheme is explained in Section III. Simulation results are presented in Section IV. Finally, the conclusions are given in Section V.

II. Related Work

Since the last decade, AAVs have been used for a wide range of applications, with aerial videography being the most popular. Despite being a topic of interest, the research on filming a moving object, considering realistic constraints and obstacle avoidance, is inadequate. In [10], Quintero and Hespanha designed optimization-based strategies using game theory for a pair of fixed-wing AAVs and shows that the visibility can be improved by using a gimbaled camera. The author assumed that the AAVs can only fly at a fixed altitude in an obstacle-free environment. Optimization-based methods such as the greedy method and informed tree search were explored with the presence of obstacles and visibility constraints by Theodorakopoulos and Lacroix [11]. However, the approach is restricted to 2D environments only. Vision-based air-to-ground target tracking problem was solved with stochastically optimized guidance law in urban environments while maximizing the navigation accuracy in [12]. Altan and Hacıoglu [13], and Mali et al. [14] proposed a model predictive control strategy for a target tracking AAV in an obstacle free environment. The approach presented in [14] was advanced in [15] by including obstacles in the environment, visibility constraints due to the limited field of view of the camera, and visibility obstruction due to the obstacles. These constraints increase the complexity of the multi-objective optimization problem and make the tuning of NMPC weights a tedious task. We extend [15] by combining a reinforcement learning approach to auto-tune the weights to get the optimal performance. We use the repetitive nature of the race to train the RL-agent (AAV) online, and the control system performance increases over time and converges to the optimal value.

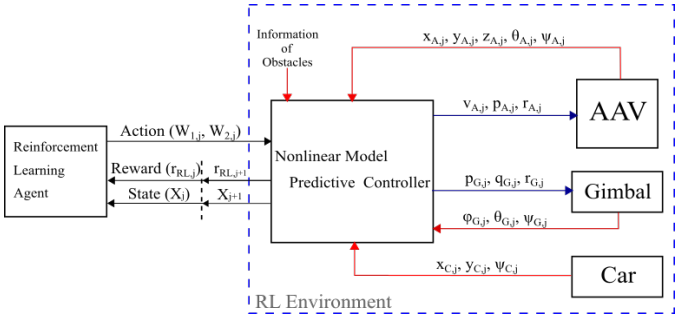


Fig. 2: Block diagram of the proposed RL-based Learning-NMPC framework.

III. Learning-Based NMPC

A block diagram showing the proposed L-NMPC scheme is given in Fig. 2. The state parameters of the AAV, gimbal, and the car are given to the NMPC block as feedback. Computed control commands from the NMPC block are applied to the AAV-gimbal system. The weights required in the NMPC cost function are calculated by the RL block. The two core components of the proposed framework, the NMPC and the RL are explained in detail in the following sections.

A. Nonlinear Model Predictive Control

NMPC uses a mathematical model of the system to compute the optimal control actions over a finite prediction horizon (N) by solving a constrained optimization problem (COP). A nonlinear cost function is minimized while satisfying the imposed constraints. Although the NMPC solves the COP for the entire horizon, only the first step of the optimized control action is applied to the system. After that, the initial time is shifted one step ahead, and the process is repeated using the current state values. Due to this behavior NMPC is also known as receding horizon control.

1) System kinematics: We use a discrete-time model for representing the motion of the gimbaled AAV. The state vector of the AAV is represented by $X_A \in \mathbb{R}^5$ and the evolution of the states over the time step j is written as

$$X_{A,j+1} = X_{A,j} + f(X_{A,j}, U_{A,j})T_s, \tag{1}$$

where T_s is the sampling time, $X_{A,j} = [x_{A,j}, y_{A,j}, z_{A,j}, \theta_{A,j}, \psi_{A,j}]^T$ and $U_{A,j} = [v_{A,j}, p_{A,j}, r_{A,j}]^T$ are the state and control vector of the AAV. Omitting the time subscript j for readability, the function $f(X_A, U_A)$ is defined as:

$$f(X_A, U_A) = \begin{bmatrix} v_A \cos\psi_A \cos\theta_A \\ v_A \sin\psi_A \cos\theta_A \\ v_A \sin\theta_A \\ p_A \\ r_A \end{bmatrix}, \tag{2}$$

where (x_A, y_A, z_A) is the location coordinates of the AAV in Cartesian frame, θ_A and ψ_A are pitch and heading (yaw) angles, respectively. v_A represents the linear velocity of the AAV, and p_A and r_A represent the pitch and yaw rates, respectively.

A 3-DoF gimbal is attached to the AAV with the assumption that the centre of the camera co-inside with the centre of gravity of the AAV. The kinematics of the gimbal is defined as

$$X_{G,j+1} = X_{G,j} + f(X_{G,j}, U_{G,j})T_s, \tag{3}$$

where $X_G = [\phi_G, \theta_G, \psi_G]^T$ and $U_G = [p_G, q_G, r_G]^T$. The function $f(X_G, U_G)$ is defined as:

$$f(X_G, U_G) = \begin{bmatrix} p_G \\ q_G \\ r_G \end{bmatrix}, \tag{4}$$

where ϕ_G, θ_G, ψ_G are the roll, pitch, and yaw angles of the gimbal, and p_G, q_G, r_G are the corresponding angular rates.

We assume that the race track is flat and the car moves only in a 2D plane. We also assume that the linear velocity v_C, angular velocity r_C, and heading angle ψ_C along with the location coordinates of the car (x_C, y_C) are available to the AAV. A standard Kalman filter can be used if the car states are not directly available. The kinematic model of the car is given as

$$X_{C,j+1} = X_{C,j} + f(X_{C,j}, U_{C,j})T_s, \tag{5}$$

where $X_C = [x_C, y_C, \psi_C]^T$ and $U_C = [v_C, r_C]^T$. The function $f(X_C, U_C)$ is given by

$$f(X_C, U_C) = \begin{bmatrix} v_C \cos\psi_C \\ v_C \sin\psi_C \\ r_C \end{bmatrix}. \tag{6}$$

The states of AAV, gimbal, and car is combined to formulate a joint state vector given by

$$X = \begin{bmatrix} X_A \\ X_G \\ X_C \end{bmatrix}. \tag{7}$$

2) Cost function: NMPC is used to determine the optimal control inputs for the AAV and the gimbal such that the AAV can film the car without physically colliding with obstacles and always keep the car within the camera FOV.

The cost function is defined as:

$$\min_{U_j} J(X_j, U_j) = \sum_{i=j}^{j+N} W_{1,i} f_1(X_i) + W_{2,i} f_2(X_i), \tag{8}$$

subject to:

$$X_{j+1} = f(X_j, U_j), \tag{9}$$
$$X_j \in [X^-, X^+], \tag{10}$$
$$U_j \in [U^-, U^+], \tag{11}$$
$$-\big[D_{o,k} - (R_A + R_{o,k})\big] \leq 0, \quad k = 1, 2 \ldots, n_o, \tag{12}$$

where $U_j = [v_{A,j}, p_{A,j}, r_{A,j}, p_{G,j}, q_{G,j}, r_{G,j}]^T$ is the combined control vector of the AAV and the gimbal, W_1 and W_2 are the weighting coefficients, X^-, X^+ are the lower and upper bounds on the states, and U^-, U^+ are the lower and upper bounds of the control inputs, respectively. The constraint (12) is added to incorporate obstacle avoidance capability for the AAV. The parameter $D_{o,k}$ is the euclidean distance between the center of the projection of the AAV on the XY-plane and center of the projection of the k^{th} obstacle on the XY-plane given by

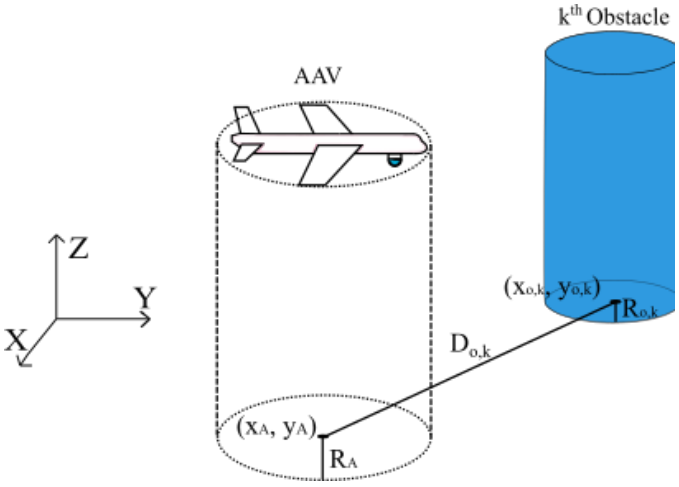


Fig. 3: Obstacle avoidance of the AAV.

$$D_{o,k} = \sqrt{(x_A - x_{o,k})^2 + (y_A - y_{o,k})^2}, \; k = 1, 2, \ldots, n_o, \tag{13}$$

where n_o is the number of obstacles present, R_A is the radius of the AAV, $R_{o,k}$ is the radius of the k^{th} obstacle, and $x_{o,k}, y_{o,k}$ are the coordinates of the center of the k^{th} obstacle on the XY-plane as shown in Fig. 3.

The components of the cost function, $f_1(X)$ and $f_2(X)$ are defined as follows.

$f_1(X)$: for accurately tracking the car, it is desired that the horizontal projection of the AAV stays aligned with the x, y coordinates of the car. Hence, f_1 is defined as the distance between the AAV and the car in the XY-plane:

$$f_1(X) = \sqrt{(x_A - x_C)^2 + (y_A - y_C)^2}, \tag{14}$$

It might not be possible to make this function zero for all instances of time because of the difference in the speed of AAV and the car. Since the AAV does not have hover capability, it will start circling on top of the car to minimize the distance in the XY-plane.

$f_2(X)$: this cost component makes sure that the car is within the FOV (Field Of View) of the camera, which is modeled as an ellipse.

$$f_2(X) = \alpha(x_C - x_F)^2 + \beta(x_C - x_F)(y_C - y_F) + \gamma(y_C - y_F)^2 - 1, \tag{15}$$

where (x_F, y_F) is the center of the FOV calculated as follows:

$$x_F = a + x_A + z_A \tan(\theta_G - VFOV/2), \tag{16}$$
$$y_F = b + y_A + z_A \tan(\phi_G - HFOV/2), \tag{17}$$

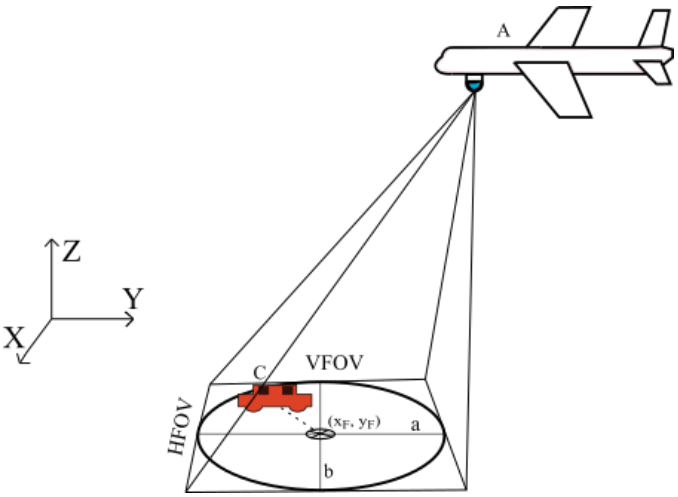


Fig. 4: Illustration showing the FOV ellipse of the gimbaled camera, the race-car and the AAV.

where a and b are the major and minor axis of the FOV ellipse, defined as

$$a = \frac{1}{2}\Big(z_A \tan\left(\theta_G + VFOV/2\right) - z_A \tan\left(\theta_G - VFOV/2\right)\Big), \quad (18)$$

$$b = \frac{1}{2}\Big(z_A \tan\left(\phi_G + HFOV/2\right) - z_A \tan\left(\phi_G - HFOV/2\right)\Big). \quad (19)$$

The parameters $VFOV$ and $HFOV$ are the vertical and horizontal field of view of the camera, which is available in the camera specifications. The coefficients α, β, and γ are defined as:

$$\alpha = \frac{\cos^2\psi_G}{a^2} + \frac{\sin^2\psi_G}{b^2}, \quad (20)$$

$$\beta = 2\cos\psi_G \sin\psi_G \left(\frac{1}{a^2} - \frac{1}{b^2}\right), \quad (21)$$

$$\gamma = \frac{\sin^2\psi_G}{a^2} + \frac{\cos^2\psi_G}{b^2}. \quad (22)$$

The cost component (15) gives a negative value when the car is within the FOV and a positive value if the car is outside of the FOV. The aforementioned parameters are illustrated in Fig. 4.

B. Reinforcement Learning

RL is a nature-inspired, experience-based learning method with three main components: the state, action, and reward. The RL agent makes a selection among the possible actions, observes the state and reward obtained for the corresponding action, and makes future actions to maximize the reward [16]. The selection of actions has two modes, exploration and exploitation. In exploration, the agent searches for actions that will give a higher reward, and in exploitation, the same action is used to increase the sum of rewards. Striking a balance between these two modes is the key to increasing the performance of the agent.

In this work, we used the constant alpha Q-learning algorithm for learning the weights for each iteration (lap) of the race. Q-update rule for the constant alpha is given by [16] as:

$$Q^{new}(s_t, a_{RL,t}) = Q(s_t, a_{RL,t}) + \alpha_{RL}\Big(r_{RL,t} + \gamma_{RL} \max_{a_{RL}} Q(s_{t+1}, a_{RL}) - Q(s_t, a_{RL,t})\Big), \quad (23)$$

where Q^{new} is the updated Q-value of the state-action pair $(s_t, a_{RL,t})$ while α_{RL} and γ_{RL} are the learning rate and the discount factor, respectively. $Q(s_t, a_{RL,t})$ is the old Q-value, and $\max_{a_{RL}} Q(s_{t+1}, a_{RL})$ represents the estimate of the optimal future value.

We consider each lap of the race as an RL episode. The state space s is same as the NMPC state vector X. The action space a_{RL} is defined as the NMPC weight set (W_1, W_2) where $W_1, W_2 \in [1, 100]$. This makes the size of action space equal to 100×100. Reward is defined as the reciprocal of the tracking error given by

$$r_{RL} = \frac{1}{\text{error}} = \frac{1}{\sqrt{(x_F - x_C)^2 + (y_F - y_C)^2}}, \quad (24)$$

The reward function gives a higher reward when the distance between the car and the center of the FOV is less.

Action selection is governed by the ε-greedy method where epsilon decay is defined as exponential decay formula given as:

$$\varepsilon = \varepsilon_{min} + (\varepsilon_{max} - \varepsilon_{min}) \times e^{-\lambda \times episode}. \quad (25)$$

where λ is the epsilon decay rate while ε_{min} and ε_{max} are the maximum and minimum values of ε.

IV. Results and Discussion

The performance of L-NMPC for filming the race-car was evaluated through numerous simulations with race tracks of different configurations and the selected results are analyzed in this section. The NMPC algorithm was implemented using the CasADi [17] framework, and a custom reinforcement learning environment was created using the OpenAI Gym [18] library.

A. Simulation setup

1) NMPC and track setup: The sampling time for the NMPC is selected as T_s = 0.2 s and the length of the prediction horizon N = 15 for all simulations. The car moves with a constant speed of v_C = 12 m/s, and the track is surrounded by ten cylindrical obstacles with a radius of 50 m and a height of 120 m. The initial positions of the car and the AAV are selected as (x_C, y_C) = (100,150) and (x_A, y_A, z_A) = (99,150,80),

AAV constraints			Gimbal constraints		
variable	min	max	variable	min	max
$z_A(m)$	75	150	$\phi_G(rad)$	$-\pi/6$	$\pi/6$
$\theta_A(rad)$	-0.2618	0.2618	$\theta_G(rad)$	$-\pi/6$	$\pi/6$
$v_A(m/s)$	14	30	$\psi_G(rad)$	$-\pi/2$	$\pi/2$
$p_A(rad/s)$	$-\pi/30$	$\pi/30$	$p_G(rad/s)$	$-\pi/30$	$\pi/30$
$r_A(rad/s)$	$-\pi/21$	$\pi/21$	$q_G(rad/s)$	$-\pi/30$	$\pi/30$
			$r_G(rad/s)$	$-\pi/30$	$\pi/30$

TABLE I: Constraints on the system.

respectively. The bounds on the states and controls are given in Table I.

2) RL setup: In order to balance the exploration and exploitation, ε-greedy method with ε_{min} = 0.03, ε_{max} = 1.0, and ε-decay rate λ = 0.008 were used. Learning rate was selected as α = 0.95 and discount factor γ = 0.85. The RL parameters were selected in a way that the agent can learn with a low number of episodes where each episode has 2000 NMPC steps. The simulations consisted of 500 episodes of training, which were carried out on an Intel i7 workstation with 16 GB of RAM.

B. Simulation results

At the beginning of the race, the RL agent explores some random actions (weight set), and they are passed to the NMPC cost function. The NMPC finds the optimal control inputs for the AAV-gimbal system with the received weight set and the error values. Here, the NMPC framework is considered as the RL environment, which gives the reward to the RL agent. The agent updates the Q-table with the weights for the corresponding NMPC step, and the iterative process is continued to get the best actions.

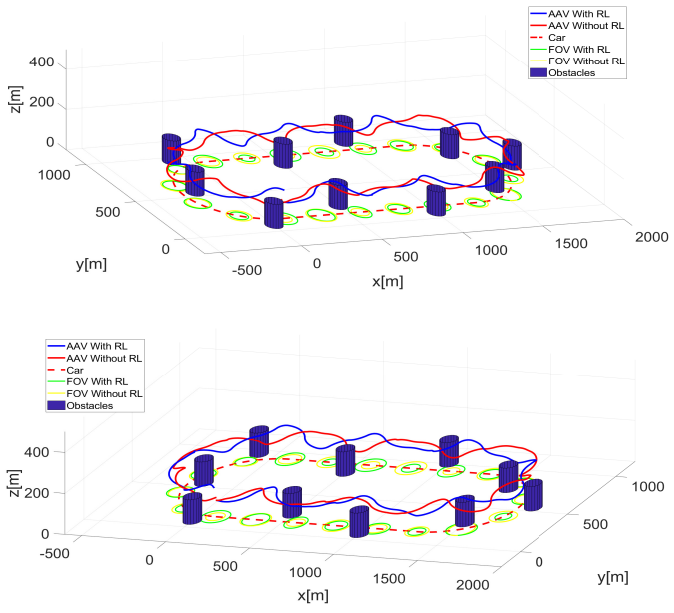
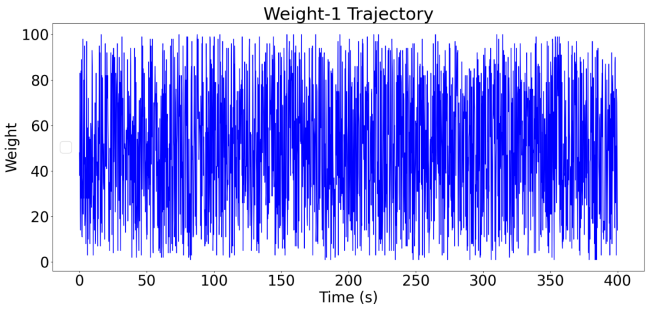
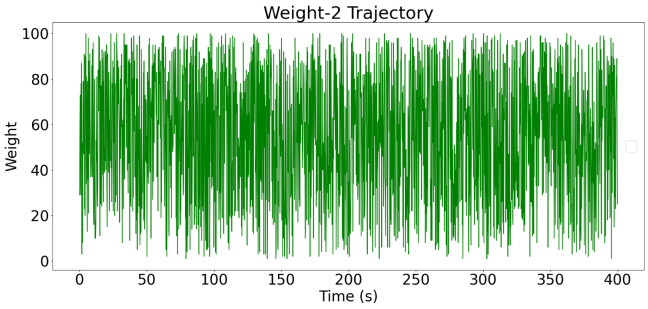


Fig. 5: Different views of the AAV tracking the car. The trajectories of the RL-trained agent (blue) and the untrained agent (red) are shown. The FOVs of the agents at intermittent time steps are also given.



(a)



(b)

Fig. 6: Weights for the NMPC cost function obtained through the learning process: (a) W_1 which regulates the term $f_1(X)$ defined in (14), (b) W_2 which regulates the term $f_2(X)$ defined in (15).

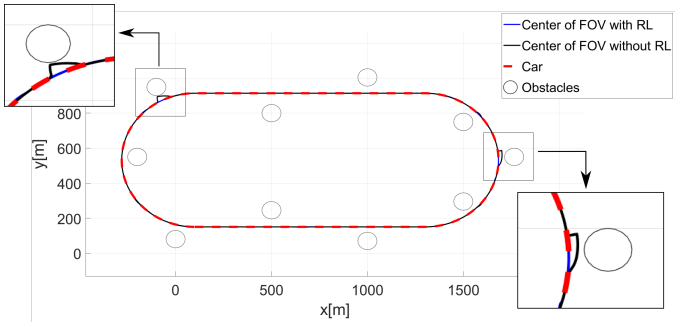


Fig. 7: Top view of the track showing the trajectories of the center of FOV of RL-trained and untrained agents and the car trajectory.

Fig. 5 shows the optimal trajectory of the AAV obtained using the proposed L-NMPC scheme. It is also compared with an agent using a standard NMPC scheme with constant pre-defined weights ($W1 = 1, W2 = 1$). It can be seen that the RL-trained AAV was able to track the car while avoiding collision with the obstacles. The performance of the trained AAV is better than the untrained AAV, as shown in Fig. 7. The error between the trajectory of the car and the center of FOV of the AAV is lower for the RL agent. The untrained agent intermittently lost track of the car while the proposed scheme delivered precise tracking. The weights obtained from

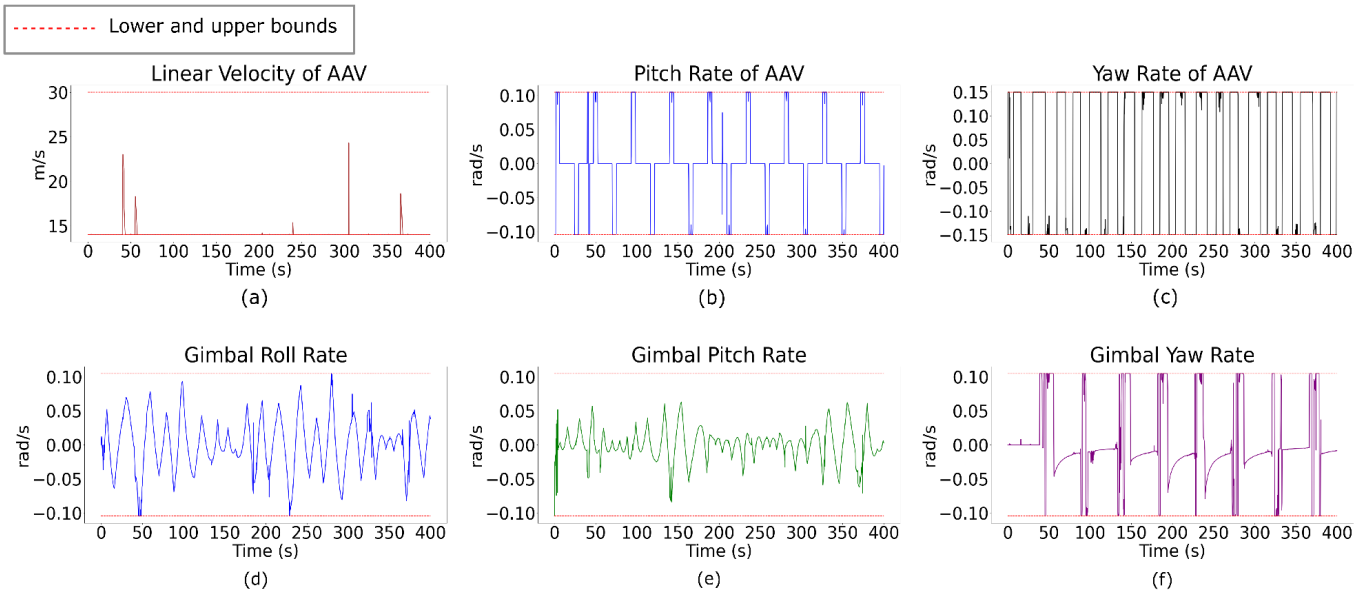


Fig. 8: Control actions of L-NMPC: (a) Linear velocity of AAV, (b) Pitch rate of AAV, (c) Yaw rate of AAV, (d) Roll rate of gimbal, (e) Pitch rate of gimbal, (f) Yaw rate of gimbal.

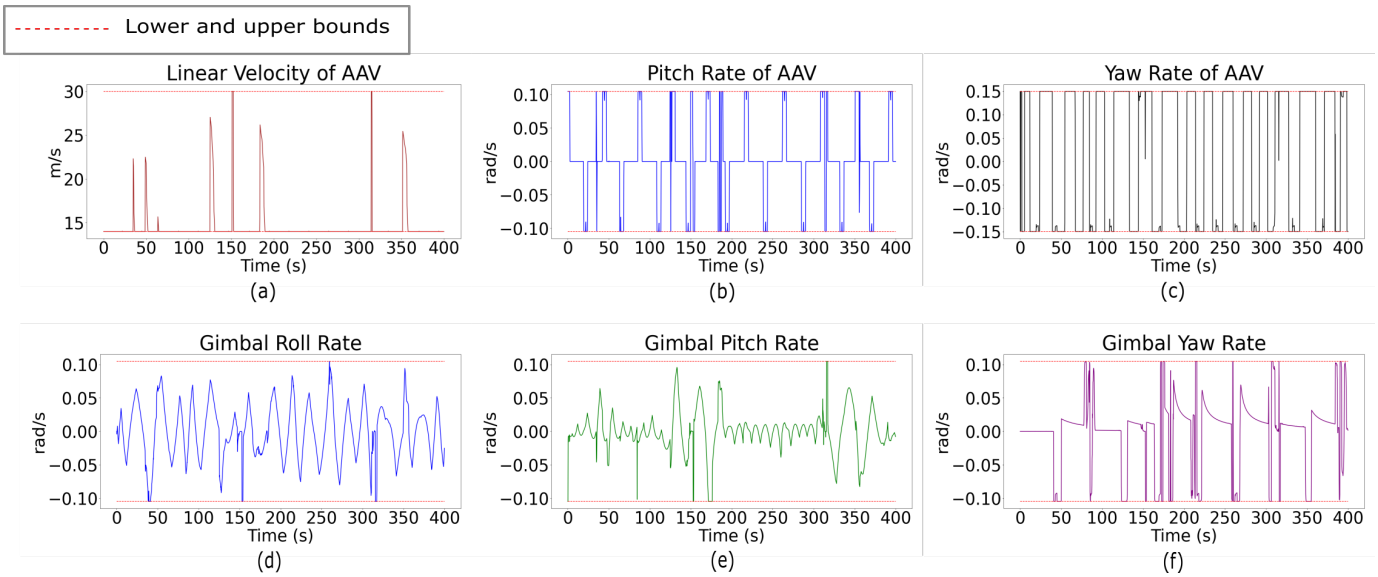


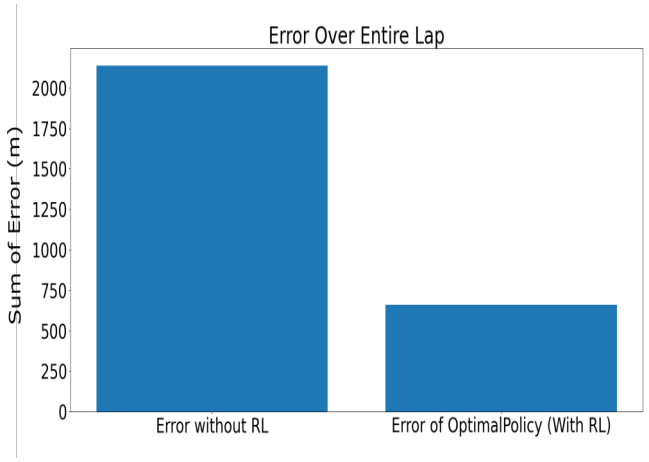
Fig. 9: Control actions of untrained NMPC: (a) Linear velocity of AAV, (b) Pitch rate of AAV, (c) Yaw rate of AAV, (d) Roll rate of gimbal, (e) Pitch rate of gimbal, (f) Yaw rate of gimbal.

the optimal RL episode is shown in Fig. 6. It is to be noted that the rapid switching of weights do not put any strain on the physical system since they are not actuating signals.

The control inputs for the AAVs computed by the L-NMPC and the standard NMPC are shown in Figures 8 and 9 respectively. The number of spikes in the linear velocity is lower for the RL-trained AAV compared to the untrained one, as shown in Fig. 8(a) and Fig. 9(a), respectively. This helps the trained AAV to film the car with more precision while roaming smoothly in the urban environment. A similar trend is visible in the other control inputs of the AAV and the gimbal as well.

The error between the car trajectory and the center of FOV of the trained and untrained AAVs are shown in Fig. 10(a). Intermittent spikes of large error can be seen for the untrained AAV, meaning that the tracking failed on multiple occasions. The RL-trained AAV showed excellent performance in keeping the tracking error to a minimum. The sum of errors for an entire lap of the race is given in Fig. 10(b). It can be seen that the L-NMPC method provided a 70% reduction in the overall lap error compared to the standard.

(a)



(b)

Fig. 10: Error between the center of FOV and the car trajectory for the trained and untrained agents. (a) Instantaneous error. (b) Total error.

V. Conclusions

In this paper, we presented a Learning-NMPC strategy for filming a robot-car using a gimbaled AAV, where the AAV increases the tracking performance iteratively using the Q-learning RL algorithm. The time-varying weights for each time step are learned to balance the NMPC cost components, resulting in an optimal balance between the multiple NMPC objectives. The proposed control scheme was found to be effective in minimizing the overall tracking errors in the race. The L-NMPC strategy was also able to handle the state and control constraints of the AAV-gimbal system and provided obstacle avoidance capability. Simulation results showed that the L-NMPC framework could increase the filming accuracy by 70% compared to a standard NMPC scheme.

Since a race usually involves a number of cars, a natural extension of this framework would be to consider tracking multiple race-cars with multi-AAVs.

References

[1] J. H. Lee and K. S. Lee, "Iterative learning control applied to batch processes: An overview," *Control Engineering Practice*, vol. 15, no. 10, pp. 1306–1318, 2007.

[2] Y. Wang, F. Gao, and F. J. Doyle III, "Survey on iterative learning control, repetitive control, and run-to-run control," *Journal of Process Control*, vol. 19, no. 10, pp. 1589–1600, 2009.

[3] C.-Y. Lin, L. Sun, and M. Tomizuka, "Robust principal component analysis for iterative learning control of precision motion systems with non-repetitive disturbances," in *American Control Conference (ACC)*. IEEE, 2015, pp. 2819–2824.

[4] K. S. Lee, I.-S. Chin, H. J. Lee, and J. H. Lee, "Model predictive control technique combined with iterative learning for batch processes," *AIChE Journal*, vol. 45, no. 10, pp. 2175–2187, 1999.

[5] K. S. Lee and J. H. Lee, "Convergence of constrained model-based predictive control for batch processes," *IEEE Transactions on Automatic Control*, vol. 45, no. 10, pp. 1928–1932, 2000.

[6] X. Liu and X. Kong, "Nonlinear fuzzy model predictive iterative learning control for drum-type boiler–turbine system," *Journal of Process Control*, vol. 23, no. 8, pp. 1023–1040, 2013.

[7] E. Bøhn, S. Gros, S. Moe, and T. A. Johansen, "Reinforcement learning of the prediction horizon in model predictive control," *IFAC-PapersOnLine*, vol. 54, no. 6, pp. 314–320, 2021.

[8] M. Mehndiratta, E. Camci, and E. Kayacan, "Automated tuning of nonlinear model predictive controller by reinforcement learning," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 3016–3021.

[9] K. M. Cabral, S. R. B. dos Santos, S. N. Givigi, and C. L. Nascimento, "Design of model predictive control via learning automata for a single uav load transportation," in *Annual IEEE International Systems Conference (SysCon)*, 2017, pp. 1–7.

[10] S. A. Quintero and J. P. Hespanha, "Vision-based target tracking with a small uav: Optimization-based control strategies," *Control Engineering Practice*, vol. 32, pp. 28–42, 2014.

[11] P. Theodorakopoulos and S. Lacroix, "Uav target tracking using an adversarial iterative prediction," in *IEEE International Conference on Robotics and Automation*, 2009, pp. 2866–2871.

[12] Y. Watanabe and P. Fabiani, "Optimal guidance design for uav visual target tracking in an urban environment," *IFAC Proceedings Volumes*, vol. 43, no. 15, pp. 69–74, 2010.

[13] A. Altan and R. Hacıoğlu, "Model predictive control of three-axis gimbal system mounted on uav for real-time target tracking under external disturbances," *Mechanical Systems and Signal Processing*, vol. 138, p. 106548, 2020.

[14] P. Mali, A. K. Singh, M. Krishnal, and P. B. Sujit, "Model predictive control for target tracking in 3d with a downward facing camera equipped fixed wing aerial vehicle," in *16th International Conference on Automation Science and Engineering (CASE)*. IEEE, 2020, pp. 165–172.

[15] P. Tyagi, Y. Kumar, and P. B. Sujit, "Nmpc-based uav 3d target tracking in the presence of obstacles and visibility constraints," in *International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2021, pp. 858–867.

[16] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[17] J. A. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "Casadi: a software framework for nonlinear optimization and optimal control," *Mathematical Programming Computation*, vol. 11, no. 1, pp. 1–36, 2019.

[18] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, "Openai gym," *arXiv preprint arXiv:1606.01540*, 2016.