Model-Free versus Model-Based Reinforcement Learning for Fixed-Wing UAV Attitude Control Under Varying Wind Conditions

David Olivares[1,2] [a], Pierre Fournier[1] [b], Pavan Vasishta[2] [c] and Julien Marzat[1] [d]

[1]*DTIS, ONERA, Université Paris-Saclay, 91123 Palaiseau, France*

[2]*AKKODIS Research, 78280 Guyancourt, France*

{*david.olivares, pierre.fournier, julien.marzat*}*@onera.fr, pavan.vasishta@akkodis.com*

Abstract: This paper evaluates and compares the performance of model-free and model-based reinforcement learning for the attitude control of fixed-wing unmanned aerial vehicles using PID as a reference point. The comparison focuses on their ability to handle varying flight dynamics and wind disturbances in a simulated environment. Our results show that the Temporal Difference Model Predictive Control agent outperforms both the PID controller and other model-free reinforcement learning methods in terms of tracking accuracy and robustness over different reference difficulties, particularly in nonlinear flight regimes. Furthermore, we introduce actuation fluctuation as a key metric to assess energy efficiency and actuator wear, and we test two different approaches from the literature: action variation penalty and conditioning for action policy smoothness. We also evaluate all control methods when subject to stochastic turbulence and gusts separately, so as to measure their effects on tracking performance, observe their limitations and outline their implications on the Markov decision process formalism.

1 INTRODUCTION

Robotic controllers must act according to their state and environment. In the context of fixed-wing unmanned aerial vehicles (FWUAV) flight, modeling the aircraft and its interactions with the environment is a challenging task for two reasons.

First, the aerodynamic forces exhibit different dynamic regimes that depend on the attitude of the FWUAV. On one hand, low attitude angles near the equilibrium state present relatively simple dynamics often approximated by formulating hypotheses on their linearity and independent coupling of the controlled axes. On the other hand, high attitude angles lead the FWUAV to encounter complex aerodynamics where lift and drag behave nonlinearly and also exhibit important cross-couplings between axes.

A second source of complexity lies in FWUAVs potentially evolving in disturbed environments. These disturbances are the result of unpredictable meteorological phenomena such as turbulence and wind gusts. As a result, effectively modeling FWUAVs' aerodynamics is often a tedious task required to design Control Theory (CT)-based controllers.

A solution could come from Reinforcement Learning (RL) to alleviate the burden of system modeling by using data to approximate these models either implicitly (model-free RL) or explicitly (model-based RL).

Model-free RL (MF-RL) control for attitude control of a quadrotor UAV is presented as outperforming the widely used PID in (Koch et al., 2019). Similar conclusions are drawn by (Song et al., 2023b) for acrobatic high-speed UAV racing. For FWUAVs, MF-RL was introduced as a robust control approach to deal with turbulence (Bøhn et al., 2019). As for model-based RL (MB-RL), it showed recent breakthroughs by outperforming MF-RL algorithms in highly complex dynamical environments such as games, locomotion and manipulation tasks (Hafner et al., 2023; Hansen et al., 2024). We identify a gap in literature for MB-RL applied to UAV flight: most recent works focus on MB-RL's sample efficiency for learning flying policies (Becker-Ehmck et al., 2020; Lambert et al., 2019) and do not evaluate the perfor-

mance of latest MB-RL algorithms. While not be-
ing the focus of our study, MB-RL often incorporates
concepts from Control Theory (CT) which can miti-
gate the blackbox aspect of MF-RL (by adding a plan-
ning step) and could be an additional benefit of this
class of methods. Our work aims to evaluate the ben-
efits of RL methods, in particular MB-RL versus MF-
RL, for attitude control of a FWUAV under varying
specific conditions (wind disturbances).

We focus on the attitude control problem, where
the controller has to track reference attitude angles for
roll and pitch. First, we choose state-of-the-art MF-
RL methods such as Proximal Policy Optimization
(PPO) and Soft Actor Critic (SAC). Second, we select
Temporal Difference Model Predictive Control (TD-
MPC), a MB-RL method that mixes concepts from
RL such as temporal difference learning and concepts
from CT for predictive planning. Finally, we include
a PID controller as a reference point, because of its
very wide use in UAV attitude control.

Our contributions are as follows:

- We propose the first application of a recent MB-
  RL method (TD-MPC) to FWUAV attitude con-
  trol under varying wind conditions.

- Our comparison provides insights for FWUAV
  practitioners on the current state-of-the-art RL al-
  gorithms with respect to the industry standard
  PID.

- We exhibit TD-MPC's superiority for pure refer-
  ence tracking in the case of nominal wind condi-
  tions (no turbulence and no gusts), most notably
  on hard attitude angle references.

- We improve this analysis with two additional
  studies:

  - A secondary metric for actuation fluctuation
    shows that RL, be it model-free or model-
    based, struggles to produce smooth action out-
    puts. We evaluate two methods from the litera-
    ture to deal with this issue.

  - In the presence of wind perturbations and when
    evaluated specifically on this aspect, the gains
    of RL methods appear more limited than sug-
    gested by previous work (Bøhn et al., 2019).

- For replicability and facilitating comparison
  among contributors, we release our code for the
  RL-compatible simulation framework[1] based on
  the open-source flight simulator JSBSim (Berndt,
  2004) and the control algorithms[2] presented in
  this work.

---

[1]https://github.com/Akkodis/FW-JSBGym
[2]https://github.com/Akkodis/FW-FlightControl

2 RELATED WORKS

Learning-based methods appear as a promising tool
to achieve nonlinear and perturbation resilient con-
trollers for UAVs. An example of learning with CT
is the field of adaptive control, which now includes
learning-based approaches where a neural network
learns a model of the problematic part of the dy-
namics (disturbances, modeling errors) and uses it as
a feed-forward component inside a classical control
controller (Shi et al., 2019; Doukhi and Lee, 2019;
Shi et al., 2021). These studies displayed encourag-
ing results for jointly using a learned model of the un-
known dynamics together with a mathematical model
of the known nominal dynamics.

2.1 Reinforcement Learning Control of UAVs

However, when the nominal dynamics are not given,
RL has proven to be a powerful choice for learn-
ing controllers without any prior knowledge of the
system. It has shown impressive results for deci-
sion making in games (Mnih et al., 2013; Schrit-
twieser et al., 2020) and in robotics (Hansen et al.,
2024; Hafner et al., 2023) such as dexterous robotic
hand manipulation (Andrychowicz et al., 2020) or
quadruped locomotion (Lee et al., 2020; Peng et al.,
2020) and can be classified into model-free and
model-based approaches. Since rotary-wing and
fixed-wing UAVs share similarities, methods applica-
ble to one may transfer effectively to the other.

2.1.1 Model-Free RL

In model-free RL (MF-RL), the agent learns to take
actions through trial and error without learning or us-
ing an explicit model of the environment's dynamics.
For rotary-wing UAVs, MF-RL has been applied as
a flight controller for waypoint navigation (Hwangbo
et al., 2017) and the recent work of (Koch et al., 2019)
has applied various RL algorithms in order to learn
attitude control policies outperforming their base-
line PID controller counterpart. A proof-of-concept
Deep Deterministic Policy Gradient (DDPG) agent
has demonstrated encouraging attitude control perfor-
mance despite exhibiting steady state errors (Tsour-
dos et al., 2019). Recent work has highlighted the
benefits of RL for trajectory tracking over classical
two-stage controllers (a high-level stage for trajectory
planning and a lower-level stage for attitude control)
(Song et al., 2023b). The authors concluded that the
cascading of two control loops splits the overall way-
point tracking objective into two sub-objectives lead-

ing to suboptimal solutions, whereas RL can optimize the higher-level waypoint objective directly.

For FWUAVs, experiments have been conducted in the context of high performance control in a combat environment of an F-16 fighter jet (De Marco et al., 2023). RL has also been studied for addressing specific problems on civil aircraft (Liu et al., 2021), where the authors used vision-based observations to achieve takeoff under crosswind conditions. A study (Bøhn et al., 2019) compared RL and PID attitude controllers under wind disturbances and both methods achieved similar performance, paving the way for the deployment of a RL control law in a real FWUAV (Bøhn et al., 2023). The former (Bøhn et al., 2019) demonstrated PPO's resilience to stochastic turbulence in simulation and the latter demonstrated SAC as a data-efficient algorithm for real-world deployment of the RL attitude controller. We use those two works as starting points and extend the analysis with a MB-RL method (TD-MPC) and under an additional disturbance source: wind gusts.

2.1.2 Model-Based RL

Model-based RL (MB-RL) is a branch of RL where the agent, through trial and error too, separately learns a model of the environment and an action planning strategy from this model. MB-RL is often more sample efficient, requiring fewer training episodes than MF-RL algorithms to converge. This observation was exploited in (Becker-Ehmck et al., 2020) where the authors trained a controller for waypoint navigation of a rotary-wing UAV, directly in the real world. (Lambert et al., 2019) present a sample efficient MB-RL method for hovering control using a "random-shooter" MPC for simulating candidate trajectories obtained with a learned-model of the dynamics. A higher-level trajectory tracking off-line MB-RL controller is presented by (Liang et al., 2018), leveraging data trajectories obtained on multiple UAVs. Current MB-RL literature for UAVs emphasizes MB-RL's sample efficiency for learning flying policies, yet does not assess the performance of the latest MB-RL algorithms. Moreover, we found no studies regarding MB-RL applied to FWUAV flight under varying wind conditions. With the present paper, we intend to extend the existing literature by evaluating the benefits of MF-RL versus MB-RL for this specific setting.

3 UAV MODEL

3.1 Kinematics

The studied FWUAV is the Skywalker x8 as in (Bøhn et al., 2019). The FWUAV is modeled as a rigid-body mass m with inertia matrix $\mathbf{I}$. According to (Beard and McLain, 2012), the following kinematic equations apply to the positions $\mathbf{p} = [x, y, z]^T$ and the orientation $\mathbf{q}$:

$$\dot{\mathbf{p}} = \mathbf{R}_b^n(\mathbf{q})\mathbf{v} \tag{1}$$

$$\dot{\mathbf{q}} = \frac{1}{2}\begin{bmatrix} 0 & -\omega^T \\ \omega & -\mathbf{S}(\omega) \end{bmatrix}\mathbf{q} \tag{2}$$

where $\mathbf{R}_b^n$ is the rotation matrix from the body frame $\{b\}$ attached to the FWUAV center of gravity to the inertial $\{n\}$ North-East-Down (NED) world frame, while $\mathbf{v} = [u, v, w]^T$ and $\omega = [p, q, r]^T$ are the linear and angular body velocities respectively and $\mathbf{S}$ is the skew-symmetric matrix.

The linear and angular velocities $\mathbf{v}$ and ω expressions are derived from the Newton-Euler equations of motion:

$$m\dot{\mathbf{v}} + \omega \times \mathbf{v} = \mathbf{R}_b^n(\mathbf{q})^T m\mathbf{g}^n + \mathbf{F}_{prop} + \mathbf{F}_{aero} \tag{3}$$

$$\mathbf{I}\dot{\omega} + \omega \times \mathbf{I}\omega = \mathbf{M}_{prop} + \mathbf{M}_{aero} \tag{4}$$

Where $\mathbf{g}^n$ is the force of gravity in the inertial frame. The other terms, $\mathbf{F}_{aero}$ and $\mathbf{M}_{aero}$ are the aerodynamical forces and moments while $\mathbf{F}_{prop}$ and $\mathbf{M}_{prop}$ are the propulsion forces and moments.

3.2 Aerodynamic Forces and Moments

3.2.1 Wind Disturbances

$\mathbf{v}_r^b$ is the velocity of the FWUAV with respect to the relative wind as:

$$\mathbf{v}_r^b = \mathbf{v}_g^b - \mathbf{v}_w^b \tag{5}$$

with $\mathbf{v}_g^b$ the velocity vector of the aircraft relative to the ground and $\mathbf{v}_w^b$ the wind speed in the body frame. The wind speed can be decomposed into two parts: a steady ambient wind component $\mathbf{v}_{w_s}^n$ expressed in the inertial frame and a stochastic process $\mathbf{v}_{w_g}^b$ representing wind turbulence expressed in the body frame.

$$\mathbf{v}_w^b = \mathbf{R}_n^b(\mathbf{q})\mathbf{v}_{w_s}^n + \mathbf{v}_{w_g}^b = \begin{bmatrix} u_r \\ v_r \\ w_r \end{bmatrix} \tag{6}$$

Similarly, the angular rate relative to the air comprises two terms: the nominal angular rate of the FWUAV ω^b and the additional turbulent angular rates ω_w^b:

$$\omega_r^b = \omega^b - \omega_w^b = \begin{bmatrix} p_r \\ q_r \\ r_r \end{bmatrix} \tag{7}$$

The stochastic processes modeling the additional turbulence correspond to white noise passed through filters given by the Dryden spectrum model (Beard and McLain, 2012; mil, 1980). The airspeed V_a, angle of attack α and sideslip angle β are defined as:

$$V_a = \sqrt{u_r^2 + v_r^2 + w_r^2} \qquad (8)$$

$$\alpha = \tan^{-1}\left(\frac{u_r}{w_r}\right) \ , \ \beta = \sin^{-1}\left(\frac{v_r}{V_a}\right) \qquad (9)$$

3.2.2 Aerodynamic Model

The flying wing Skywalker x8 is not equipped with a rudder. The available control surfaces only consist in aileron and elevator deflection angles, δ_a and δ_e, and a throttle command δ_t. The translational aerodynamic forces $\mathbf{F}_{aero}$ are the drag D, the lateral force Y and the lift L. They are expressed in the wind frame and can be expressed in the body frame as:

$$\mathbf{F}_{aero} = \mathbf{R}_w^b(\alpha, \beta) \begin{bmatrix} - & D \\ & Y \\ - & L \end{bmatrix} \qquad (10)$$

$$\begin{bmatrix} D \\ Y \\ L \end{bmatrix} = \frac{1}{2}\rho V_a^2 S \begin{bmatrix} C_D(\alpha, \beta, q_r, \delta_e) \\ C_Y(\beta, p_r, r_r, \delta_a) \\ C_L(\alpha, q_r, \delta_e) \end{bmatrix} \qquad (11)$$

Typically, these nonlinear forces are described by C_D, C_Y, C_L. They consist in a set of coefficients determined by computational fluid dynamics simulations and/or experimental wind tunnel data. The expressions of the aerodynamic moments $\mathbf{M}_{aero}$ can be found following the same logic:

$$\mathbf{M}_{aero} = \frac{1}{2}\rho V_a^2 S \begin{bmatrix} bC_l(\beta, p_r, r_r, \delta_a) \\ cC_m(\alpha, q_r, \delta_e) \\ bC_n(\beta, p_r, r_r, \delta_a) \end{bmatrix} \qquad (12)$$

where ρ is the air density, V_a the nominal airspeed, S the wing area, b the wingspan and c the aerodynamic chord. The aerodynamic coefficients and geometric parameters of the x8 FWUAV are taken from (Gryte et al., 2018).

3.2.3 Propulsion Forces and Moments

The propeller of the x8 FWUAV is located at the back of the airframe and aligned with the body frame's x-axis such that $\mathbf{F}_{prop} = [T_p, 0, 0]^T$. In the current simulation, a simplified model of the engine and the propeller is used, which provides a linear relation $T_p = C\delta_t$ between the thrust force T_p and the δ_t throttle command given in the $[0, 1]$ interval, with $C = 5.9$.

3.3 Actuator Dynamics and Constraints

Input commands in the simulator are normalized in the range $[-1, 1]$ for control surface deflections (δ_a, δ_e) and between $[0, 1]$ for the throttle δ_t. Following (Bøhn et al., 2019), the aileron and elevator dynamics are described by rate-limited and saturated second-order transfer functions with natural frequency $\omega_0 = 100$ and damping $\zeta = \frac{1}{\sqrt{2}}$. The control surface deflection angles and rates are limited to $\pm 30°$ and $\pm 200 °$/s. The throttle dynamics are modeled by a first-order transfer function, with gain $K = 1$ and time constant $T = 0.2$.

4 CONTROL METHODS

In this study the controller's task is to track desired states ϕ^d and θ^d for the attitude roll and pitch angles ϕ and pitch θ of the aforedescribed FWUAV when subject to various wind disturbances. Following (Bøhn et al., 2023), the airspeed remains controlled by a PI controller driving the throttle command δ_t to maintain a nominal airspeed, with the gains tuned as in (Bøhn et al., 2019): $k_{p_V} = 0.5$ and $k_{i_V} = 0.1$. This airspeed PI controller is common to all the following controllers. This choice was motivated by the fact that when certain references are generated, the desired pitch can contradict the desired airspeed, e.g. low desired airspeed and nose down desired pitch. In a case where one would control the desired altitude, the outer loop would give coherent desired states to the inner attitude control loops, but this is beyond the scope of the present work.

4.1 PID Control

PID control is a widely used controller for UAV attitude control because of its ease of use and is therefore considered as a reference point in the RL literature for UAV control (Bøhn et al., 2019; Bøhn et al., 2023; Koch et al., 2019; Lambert et al., 2019). We fine-tuned the PID gains given by (Bøhn et al., 2019) to better suit our simulated loops in nominal conditions i.e. with no wind disturbances, with the gains presented in Table 1.

Table 1: PID controller parameters.

Parameter	Value	Parameter	Value
$k_{p\phi}$	1.5	$k_{p\theta}$	-2
$k_{i\phi}$	0.1	$k_{i\theta}$	-0.3
$k_{d\phi}$	0.1	$k_{d\theta}$	-0.1

4.2 Reinforcement Learning (RL) Control

In RL, the control problem is formulated as that of finding an optimal action strategy in an environment modeled as a Markov Decision Process (MDP) (Sutton and Barto, 2018) comprised of:

- A set of actions $a \in A$, here the same as in the roll and pitch PID loops with the superscript c denoting a commanded position of the aileron and elevator, $a = (\delta_a^c, \delta_e^c)$.

- A set of states $s \in S$, based on (Bøhn et al., 2023):
  - the attitude angles: roll ϕ and pitch θ;
  - the airspeed V_a;
  - the angular rates $\omega_r^b = [p_r, q_r, w_r]^T$;
  - the angle of attack α and the angle sideslip β;
  - the roll and pitch errors:

$$e_\phi = \phi^d - \phi$$
$$e_\theta = \theta^d - \theta \qquad (13)$$

  - the roll and pitch integral errors:

$$I_{e_*} = I_{e_*} + e_* \cdot dt, * = \{\phi, \theta\}, dt = 0.01 \quad (14)$$

    where dt is the simulation period, also equal to the control rate. The integral error is reset at the beginning of each episode or target change.
  - The last action taken by the agent: $\delta_{a|_{t-1}}^c$ and $\delta_{e|_{t-1}}^c$.

- A transition function T, defining a distribution over the next states given a current state and action: $P(s_{t+1}|s_t, a_t) = T(s_t, a_t)$.

- A scalar reward function R_t, outputting a score, rewarding the agent for good actions and penalizing bad ones, that shapes the agent's task.

The reward function has been adapted from (Bøhn et al., 2019) as follows:

$$R_\phi = \text{clip}\left(\frac{|\phi - \phi^d|}{\zeta_\phi}, 0, c_\phi\right)$$
$$R_\theta = \text{clip}\left(\frac{|\theta - \theta^d|}{\zeta_\theta}, 0, c_\theta\right) \qquad (15)$$
$$R_t = -(R_\phi + R_\theta)$$
$$\zeta_\phi = 3.3, \zeta_\theta = 2.25, c_\phi = c_\theta = 0.5$$

where ζ_* are scaling coefficients to take into account the differences between roll and pitch error ranges, each reward component is clipped to 0.5 so the minimal total reward equals to -1.

RL algorithms aim to discover the best mapping π from a state to an action (control law), in the RL literature, referred as the optimal policy (π^*), and defined as the policy leading to trajectories $\tau = (s_0, a_0, s_1, a_1, ...)$ that maximize the expected discounted sum of accumulated rewards:

$$\pi^* = \arg\max_\pi \mathbb{E}_{\tau \sim \pi}\left[\sum_{t=0}^\infty \gamma^t R_t(s_t, a_t)\right] \qquad (16)$$

with γ being the discount factor.

This policy, often represented by an artificial neural network (ANN) with learnable parameters Θ, serves as a nonlinear controller.

4.2.1 Model-Free RL: PPO & SAC

We choose Proximal Policy Optimization (PPO) (Schulman et al., 2017) and Soft Actor-Critic (SAC) (Haarnoja et al., 2018) to represent the MF-RL end of the control spectrum because of their state-of-the-art performance for robotic applications with continuous action spaces. Moreover, SAC's choice was motivated by its similar policy update to the mixed method used in this work (TD-MPC).

4.2.2 Model-Based Method: TD-MPC

Temporal Difference Model Predictive Control (TD-MPC) (Hansen et al., 2022) is an MB-RL, mixed control method combining an explicit dynamics model and a terminal value function learned by using Temporal Difference learning in an RL setting. We use its latest implementation from (Hansen et al., 2024).

Trainable Models. TD-MPC learns a Task-Oriented Latent Dynamics (TOLD) model, comprised of several MLPs (Multi-Layer Perceptrons) with trainable parameters Θ. They aim at modeling certain functions, most notably: the agent's dynamics, the reward function R_Θ, the Q-value function Q_Θ and the policy. During training, a trajectory of prediction horizon H is sampled from a replay buffer in an off-policy RL fashion. Using this sampled trajectory, the TOLD model is trained by optimizing a loss function including: the reward prediction function, the dynamics model and a Bellman error component for Q_θ. The policy is learned by maximizing a similar objective to the SAC algorithm (Haarnoja et al., 2018).

Planning. TD-MPC uses a control-theory basis for planning, utilizing Model Predictive Path Integral (MPPI) (Williams et al., 2015). MPPI is a sampling-based model-predictive control (MPC) algorithm iteratively learning the parameters of a multivariate Gaussian distribution using importance sampling. First, TD-MPC simulates candidate trajectories Γ by sampling a mixture of trajectories of horizon H by querying both the Gaussian distribution and the

learned policy function π_Θ for actions and by recursively predicting next states using the learned dynamics model. One of TD-MPC's key idea is to use the learned Q-value function Q_Θ to give an estimation of the terminal value when computing a trajectory's total return ϕ_Γ. Here, Q_Θ gives an estimate of the value of the trajectory beyond the horizon limit, emulating infinite-horizon MPC.

$$\phi_\Gamma \triangleq \mathbb{E}_\Gamma[\underbrace{\gamma^H Q_\Theta(z_H, a_H)}_{\substack{\text{Terminal value} \\ \text{estimated by } Q_\Theta}} + \underbrace{\sum_{t=0}^{H-1} \gamma^t R_\Theta(z_t, a_t)}_{\substack{\text{Total return of the} \\ \text{candidate trajectory over} \\ \text{the time horizon } H}}] \quad (17)$$

As illustrated in Figure 1, after taking the trajectories with the best return, MPPI computes the action Gaussian parameters update by using importance sampling, see (Hansen et al., 2022). The action Gaussian is then sampled and the first action is sent to the actuators, as in receding-horizon MPC.
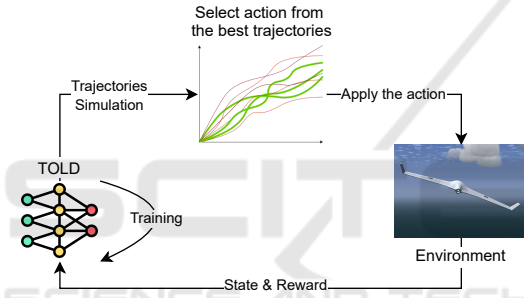

Figure 1: TD-MPC Control Block Diagram. 1) The trained TOLD model is used for simulating candidate trajectories and estimating their return. 2) Using importance sampling update an action Gaussian distribution and sample it. 3) Sample an action from the action distribution and apply it.

5 EXPERIMENTAL SETUP

5.1 Simulation

We conducted our experiments in the JSBSim flight simulator (Berndt, 2004) because of its ability to handle fixed-wing aircraft flight dynamics and to model all the studied wind disturbances: constant wind, wind gusts and stochastic turbulences. JSBSim was preferred over the PyFly simulator presented in (Bøhn et al., 2019) because of its wide use in the aerospace open-source and research communities and its built-in wind gust generation (Moallemi and Towhidnejad, 2016; Mathisen et al., 2021; De Marco et al., 2023).

We are interested in separating the performance of the studied control methods between linear and nonlinear regimes. Therefore, we introduce a classifica-

tion of attitude angle references in Table 2, with nominal and hard levels representing linear and nonlinear regions of the state space respectively.

Table 2: References classification.

Difficulty	Roll(°)	Pitch(°)
Nominal	$[-45, 45]$	$[-25, 25]$
Hard	$[-60, -45] \cup [45, 60]$	$[-30, -25] \cup [25, 30]$

5.2 Implementation Details

For the optimization algorithms, we used the CleanRL PPO & SAC implementations (Huang et al., 2022) with the parameters presented in Tables 3 & 4. We used the most recent implementation of TD-MPC, TD-MPC2 (Hansen et al., 2024) with the parameters specified in Table 5.

We trained the RL agents for 375 episodes of 2000 timesteps each, i.e. 20 seconds of flight time at 100 Hz. At the start of every episode, we initialize the FWUAV at 600 m above sea level, with a nominal speed of 17 m/s and roll, pitch, yaw $\{\phi, \theta, \psi\} = 0$. We compute the mean and standard error of the mean over 5 runs for each metric. Training was conducted on a RTX A6000 with 48GB of memory.

Table 3: PPO Parameters.

Parameter	Value	Parameter	Value
LR	3e-4	Entropy coef	1e-2
Parallel envs	6	Value fn coef	0.5
Rollout steps	2048	Total timesteps	750k
Discount factor γ	0.99	Minibatches	32
GAE λ	0.95	Batch size	12228
PPO clip	0.2		

Table 4: SAC Parameters.

Parameter	Value	Parameter	Value
Buffer size	1e6	Q-net LR	1e-3
Discount factor γ	0.99	π update freq	2
Q_{targ} smoothing τ	5e-3	Q_{targ} update freq	1
Batch size	256	Total timesteps	750k
Seed steps	5000	Noise clip	0.5
Policy LR	3e-4	Entropy reg	0.2

Table 5: TD-MPC Parameters.

Parameter	Value	Parameter	Value
Buffer size	1e6	Total timesteps	750k
Discount factor γ	0.99	Temporal coef	0.5
Target smoothing τ	0.01	MPPI Iterations	6
Batch size	256	MPPI Samples	512
Seed steps	10 000	MPPI Elites	64
LR	3e-4	MPPI π Trajs	24
Reward coef	0.1	Horizon	3
Value coef	0.1	Temperature	0.5
Consistency coef	20		

6 RESULTS AND DISCUSSIONS

Our first experiment aims at comparing RL agents in nominal, non-disturbed conditions and with respect to reference tracking only (Section 6.1.1). We also show that RL agents tend to converge towards policies that produce oscillating actions, and we evaluate two strategies to mitigate the issue (Section 6.1.2). Then we study the impact of turbulence and wind gusts separately on each algorithm performances (Section 6.2) and provide a deeper discussion of the results (Section6.3).

6.1 Nominal Wind Conditions

This section focuses on evaluating all 4 controllers under nominal wind conditions i.e. no wind, no turbulence and no wind gusts. Agents are trained and tested with these wind settings and across the two reference difficulty levels from Table 2.

6.1.1 Base RL Algorithms

The results from Table 6 confirm that PID is a strong baseline in nominal conditions, with only TD-MPC from RL algorithms performing slightly better. On the contrary, under more nonlinear dynamics forced by hard references, all RL methods clearly outperform PID, with a significant advantage of TD-MPC over PPO and SAC. Figure 4 illustrates TD-MPC's better hard reference tracking compared to PPO. Overall TD-MPC displays moderate to strong gains over all methods across both dynamics regimes.

6.1.2 RL Policies and High Action Oscillation

The RL literature usually focuses on performance metrics only (Koch et al., 2019; Hansen et al., 2022), but for true robotic applications, we practitioners must also make sure to minimize wear and tear as well as energy consumption. This can only be done by monitoring actuation solicitation and thus we incorporate in our evaluation an actuation fluctuation (Act Fluct) metric, following (Song et al., 2023a):

$$\xi(\pi) = \frac{1}{2} \sum_{j \in [a,e]} \frac{1}{E} \sum_{n=1}^{E} \left[\frac{1}{T} \sum_{t=1}^{T} |\delta_{j|_t} - \delta_{j|_{t-1}}| \right] \quad (18)$$

It incentivizes the minimization of the distance between two consecutive actuator positions (δ_a or δ_e).

Table 7 shows that high performances of RL methods come at the cost of strongly oscillating actions. In fact, RL can lead to bang-bang control, where the learned optimal policy consists in alternating abruptly between actions far from each other in the action

space (Seyde et al., 2021). To mitigate this issue, we evaluate two oscillation regulation methods from the literature.

Action Variation Penalty (AVP). Adding an AVP in the reward function is a common way of addressing the bang-bang policy problem. Hence, we modify the base reward of Equation 15 and propose adding a simple additional reward component R_δ, similar to (Bøhn et al., 2019) for penalizing abrupt changes of consecutive actions, forming the reward function denoted R_{AVP}:

$$R_\delta = \text{clip}\left(\frac{\frac{1}{2}\sum_{j \in [a,e]} |\delta^c_{j|_t} - \delta^c_{j|_{t-1}}|}{\zeta_\delta}, 0, c_\delta \right)$$

$$\delta^c_{a|_{t-1}} R_{AVP} = -(R_\phi + R_\theta + R_\delta)$$
$$\zeta_\phi = 3.3, \zeta_\theta = 2.25, \zeta_\delta = 2$$

$$(19)$$

We separately tuned the c_* clipping coefficients of the AVP reward function for each RL method, giving $c_\phi = c_\theta = 0.25$ and $c_\delta = 0.5$ for TD-MPC and SAC. However, such a tuning led PPO not converging to a successful flying policy, therefore we found an appropriate tuning for this method as $c_\phi = c_\theta = 0.45$ and $c_\delta = 0.1$.

Conditioning for Action Policy Smoothness (CAPS). Another candidate to mitigate this behavior is the Conditioning for Action Policy Smoothness (CAPS) loss (Mysore et al., 2021). CAPS is added to the policy loss and prevents the emergence of non-smooth action policies by applying temporal and spatial regularization:

$$L_{\text{CAPS}}(\pi_\Theta) = \lambda_{\text{TS}} \cdot L_{\text{TS}}(\pi_\Theta) + \lambda_{\text{SS}} \cdot L_{\text{SS}}(\pi_\Theta)$$
$$L_{\text{TS}}(\pi_\Theta) = \|\pi_\Theta(s_t) - \pi_\Theta(s_{t+1})\|_2$$
$$L_{\text{SS}}(\pi_\Theta) = \|\pi_\Theta(s_t) - \pi_\Theta(\hat{s}_t)\|_2, \quad \hat{s}_t \sim \mathcal{N}(s_t, 0.01)$$
$$(20)$$

The time-only version of the CAPS loss is used to enforce temporal smoothness (with $\lambda_{TS} = 0.05$) while leaving room for high reactivity in the state space.

Results. Table 7 exhibits the consistent improvement of the action fluctuation metric across all MF-RL agents with CAPS. For RMSE tracking performance, Table 7 confirms that TD-MPC remains better than PID and MF-RL agents.

As for action regulation methods, we observe that action regulation methods remain algorithm dependent. Table 7 shows that the AVP reward function has a strong effect on actuation fluctuation, reducing it by at least an order of magnitude for TD-MPC and SAC. Conversely, AVP for PPO shows a drastic increase in actuation fluctuation, which highlights the difficult task of tuning the reward function with the goal of combining a lower actuator fluctuating flying policy. In fact, one tuning of AVP can work for certain

Table 6: Nominal Atmosphere. Average of roll and pitch tracking RMSE, over all reference difficulty levels. Best scores are highlighted in blue.

Agent	Nominal Refs ($\times 10^{-2}$)	Hard Refs ($\times 10^{-2}$)
PID	4.20	20.10
TD-MPC	3.81 ± 0.10	9.10 ± 0.69
PPO	5.29 ± 0.12	15.78 ± 2.48
SAC	6.64 ± 0.83	11.15 ± 0.65

Table 7: Nominal Atmosphere + Actuation Regulation (through AVP or CAPS). Average of roll - pitch tracking errors and aileron - elevator action fluctuation, over all reference difficulty levels. **Bold** values are the best score among one agent type. Blue : best RMSE score across all agents. Red : best Actuator Fluctuation score across all agents.

Agent	Nominal Refs ($\times 10^{-2}$)		Hard Refs ($\times 10^{-2}$)	
	RMSE	Act Fluct	RMSE	Act Fluct
PID	4.20	0.07	20.10	0.17
TD-MPC	3.81 ± 0.10	4.37 ± 0.48	9.10 ± 0.69	6.02 ± 0.59
TD-MPC AVP	$\mathbf{3.58 \pm 0.01}$	$\mathbf{0.54 \pm 0.02}$	$\mathbf{8.09 \pm 0.03}$	$\mathbf{0.60 \pm 0.02}$
PPO	5.29 ± 1.20	1.98 ± 0.58	15.78 ± 2.48	5.54 ± 0.93
PPO CAPS	$\mathbf{5.05 \pm 1.50}$	$\mathbf{1.47 \pm 0.27}$	19.17 ± 4.68	$\mathbf{4.56 \pm 0.60}$
PPO AVP	5.78 ± 2.10	13.78 ± 3.16	$\mathbf{13.80 \pm 0.82}$	13.48 ± 2.50
SAC	6.64 ± 0.83	10.07 ± 1.12	$\mathbf{11.15 \pm 0.65}$	12.00 ± 0.71
SAC CAPS	6.79 ± 0.74	0.53 ± 0.27	12.28 ± 1.17	1.44 ± 0.55
SAC AVP	$\mathbf{6.54 \pm 0.75}$	$\mathbf{0.50 \pm 0.18}$	15.14 ± 3.39	$\mathbf{1.20 \pm 0.32}$

algorithms such as SAC and TD-MPC and give an underperforming PPO agent requiring a specific tuning. CAPS appears to be a better action regulation method for MF-RL methods because of its consistency with a single tuned parameter $\lambda_{TS} = 0.05$. As an example, Figure 5 outlines CAPSs' action smoothing for SAC through a trajectory plot. As a result, the CAPS method has been retained for the following ablations. Despite showing great results for MF-RL methods, CAPS is not directly applicable to TD-MPC because it only regularizes its learned policy prior, while TD-MPC never directly outputs an action by sampling this prior. Indeed, we found in additional experiments that applying CAPS led to learning a smooth action policy prior but did not guarantee smooth actions after the planning step.

6.2 Wind Disturbances

Aerial robotics present challenging environmental conditions which add a layer of complexity for control systems. This section focuses on studying the impact of such conditions on our four controllers of interest. Wind disturbances are divided into two sub-categories: stochastic turbulence modeled by probabilistic dynamics and wind gusts consisting in unpredictable and sudden changes in dynamics. Agents are tested and trained under stochastic turbulence in Section 6.2.1 and wind gusts in Section 6.2.2 separately.

The parameters for turbulence and gusts are drawn uniformly along the values reported in Table 8. This classification of constant winds is only used for evaluation. During training, constant wind magnitudes are uniformly sampled from the continuous interval $[0, 23]$ m/s. Constant wind and wind gusts directions with respect to the NED inertial frame are also sampled uniformly in the $[-1, 1]$ interval.

To better isolate the effects of each disturbance from the effects of hard attitude angle reference tracking, the agents were trained and tested only for the nominal attitude reference angle range from Table 2.

Table 8: Wind Disturbance Severity Levels (m/s). The classification for constant wind and gust magnitudes is taken from (Bøhn et al., 2019). "Turbulence W20" is a predefined parameter in JSBSim Dryden model turbulence classification (Berndt, 2004).

	Constant Wind	Turbulence W20	Gust
OFF	0	0	0
Light	7	7.6	7
Moderate	15	15.25	15
Severe	23	22.86	23

6.2.1 Stochastic Turbulence

As explained in Section 3.2.1, turbulence is simulated as a stochastic process applying linear and angular forces to the FWUAV. On one side, Figure 2 outlines SAC's underperforming across all turbulence levels. On the other side, TD-MPC shows good performance, reaching the same levels of performance as PID and PPO. We observe that the severity of wind disturbance dominates over the choice of algorithm

to predict the final performances. In contrast to Section 6.1.1, where TD-MPC showed a clear advantage on deterministic nonlinear dynamics for tracking high attitude reference angles, here its learned dynamics model seems here to have difficulty capturing the stochastic turbulent dynamics of this scenario, and can only resort to learning an "on-average" robust policy. To the best of our knowledge, TD-MPC has not been applied to environments with stochastic dynamics and adapting TD-MPC to stochastic environments remains an open problem.
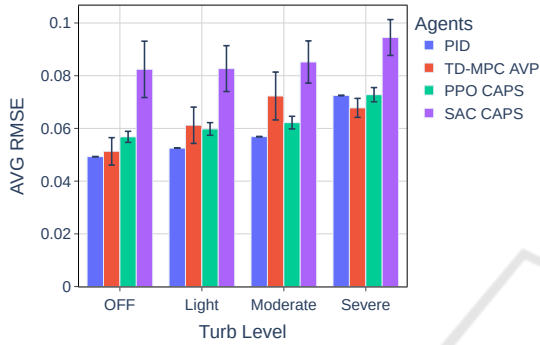

Figure 2: Stochastic Turbulences + Action Regulation. Average tracking RMSE on nominal reference difficulty.

6.2.2 Wind Gusts

In our simulated environment, wind gusts are triggered twice per episode at timesteps 500 and 1500. Gusts are parameterized with a startup duration (time for reaching maximum wind speed) of 0.25 s, a steady duration (time span where the wind stays at maximal magnitude) of 0.5 s, and an end duration (time for the gust to disappear) of 0.25 s. The startup and end transients are modeled as a smooth cosine function. The magnitude level of the gust is determined by its maximum wind speed and follows Table 8.

The results presented in Figure 3 point that SAC significantly underperforms relative to the other agents and TD-MPC slightly underperforms PPO and PID. To explain these observations, we emit the hypothesis that such a disturbance alters the MDP formalism into a Partially Observable Markov Decision Process (PO-MDP). In fact, an MDP is comprised of a transition function $P(s_{t+1}|s_t, a_t) = T(s_t, a_t)$ which does not exist at every point in time in the case of unpredictable gusts. From a formal point of view, while hard references simply lead to a more complex transition function, wind gusts actually make it impossible to rigorously define such a transition function. Wind information is hidden from the agent which cannot anticipate wind variations, thus conducing to state aliasing (McCallum, 1996). In fact, unpredictable

gusts lead to the following situation: for two separate experiments, for a given common state, taking the same action could lead to two different next states. Recurrent Neural Networks have been presented as a good baseline for solving PO-MDPs (Ni et al., 2022; Asri and Trischler, 2019) and could be applied to all the RL methods presented in the present article.

RL methods learn the transition function either explicitly in the case of model-based RL (TD-MPC) or implicitly in the case of model-free RL (PPO and SAC). Due to state aliasing, opposite updates of the function approximators occur at the onset of a wind gust. To explain PPO's better performance compared to its RL counterparts (SAC and TD-MPC), a second hypothesis could be PPO's clipping of the policy loss (Schulman et al., 2017). Motivated by limiting too large policy updates that could lead to a bad policy where it can be hard to recover from, this conservative policy update could protect the PPO agent from the contradictory updates when subject to wind gusts.
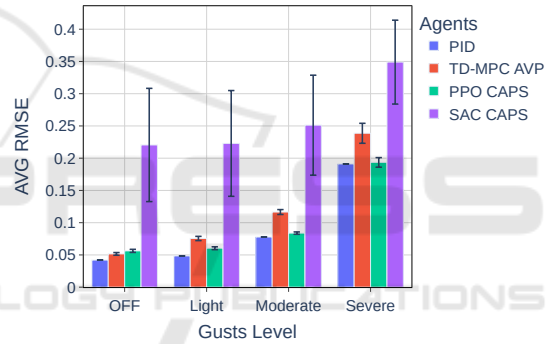

Figure 3: Wind Gusts + Action Regulation. Average tracking RMSE on nominal reference difficulty.

6.3 Discussions

TD-MPC's Superiority for Nonlinear Regimes. The superior results of TD-MPC under nonlinear dynamics are consistent with the reported superior performance of TD-MPC over other MF-RL methods for complex nonlinear manipulation and locomotion tasks (Hansen et al., 2022; Hansen et al., 2024). It solidifies the hypothesis that mixed control, combining learning an explicit model of the system dynamics and leveraging it for planning with terminal value estimation, can result in improved tracking performance for a wide range of attitude angles. It also suggests that learning an implicit representation of the dynamics limits the ability of MF-RL agents to deal with different dynamic regimes when trained simultaneously on nominal and hard regimes.

Wind Disturbances. We also evaluated the control methods under wind disturbances of different nature.
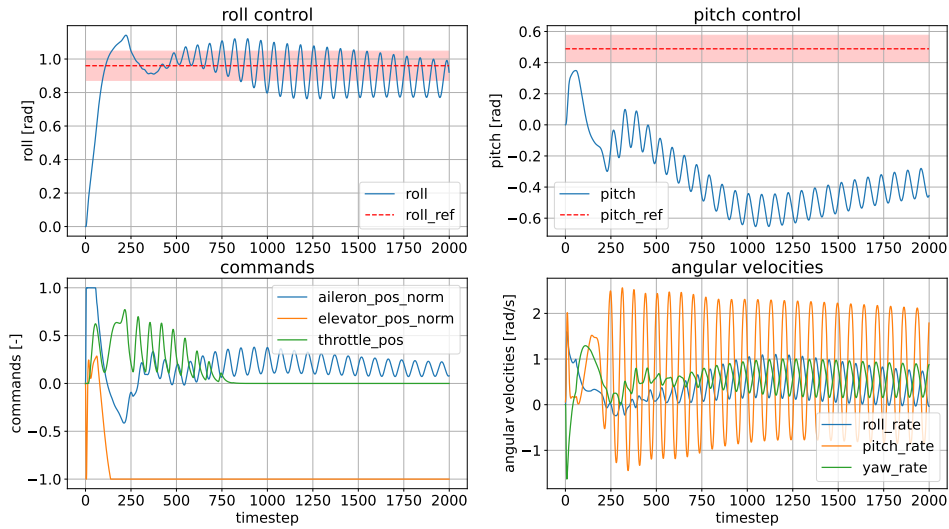
(a) PPO.
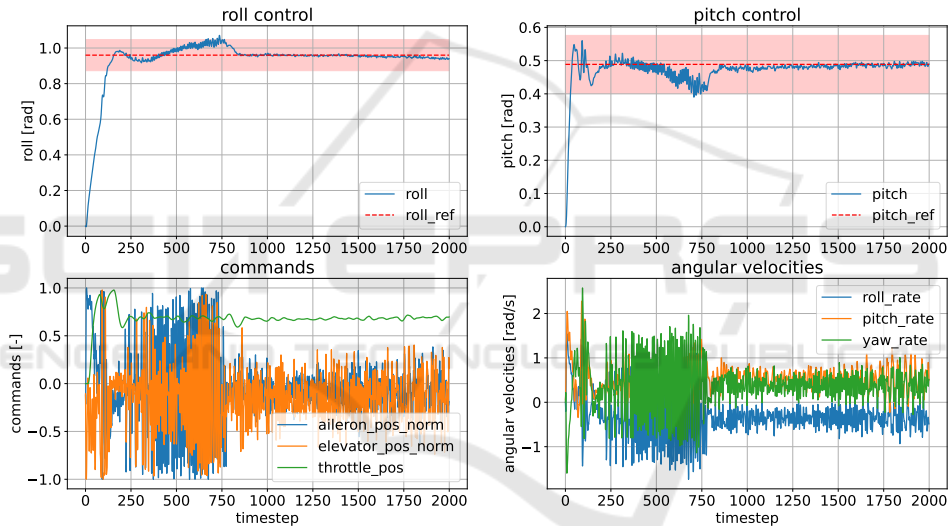
(b) TD-MPC.

Figure 4: TD-MPC's Superiority for Hard References: PPO vs TD-MPC. Red dashed lines are the references: Roll = 55°, Pitch = 28°. The red area around the reference line corresponds to ±5° error bounds.

Both studies revealed that with the exception of SAC underperforming, the severity of turbulences dominates over the choice of algorithm to predict final performances. As previously stated, hard references only consist in a harder-to-model nonlinear MDP transition function. We hypothesized that turbulence and gusts transform the MDP formalism into a non-stationary PO-MDP and we observe that not all perturbations affect with equal magnitude how well the MDP formalization fits the environment conditions. The turbulence study presented in Section 6.2.1 suggest that turbulence can be dealt with more simply than gusts because there exists a time-averaged MDP close to the true MDP that RL algorithms fall back into by de-

fault. However, gusts present a multi-modal problem (nominal mode and gust mode) by making it impossible to strictly define a transition function which could be detrimental in the learning phase as presented in Section 6.2.2. Therefore, we conclude that training agents with various of perturbations is not enough to ensure robustness and generalization and that different types of perturbations may require different types of approaches. The multi-model MDP framework could be investigated for this purpose (Steimle et al., 2021).
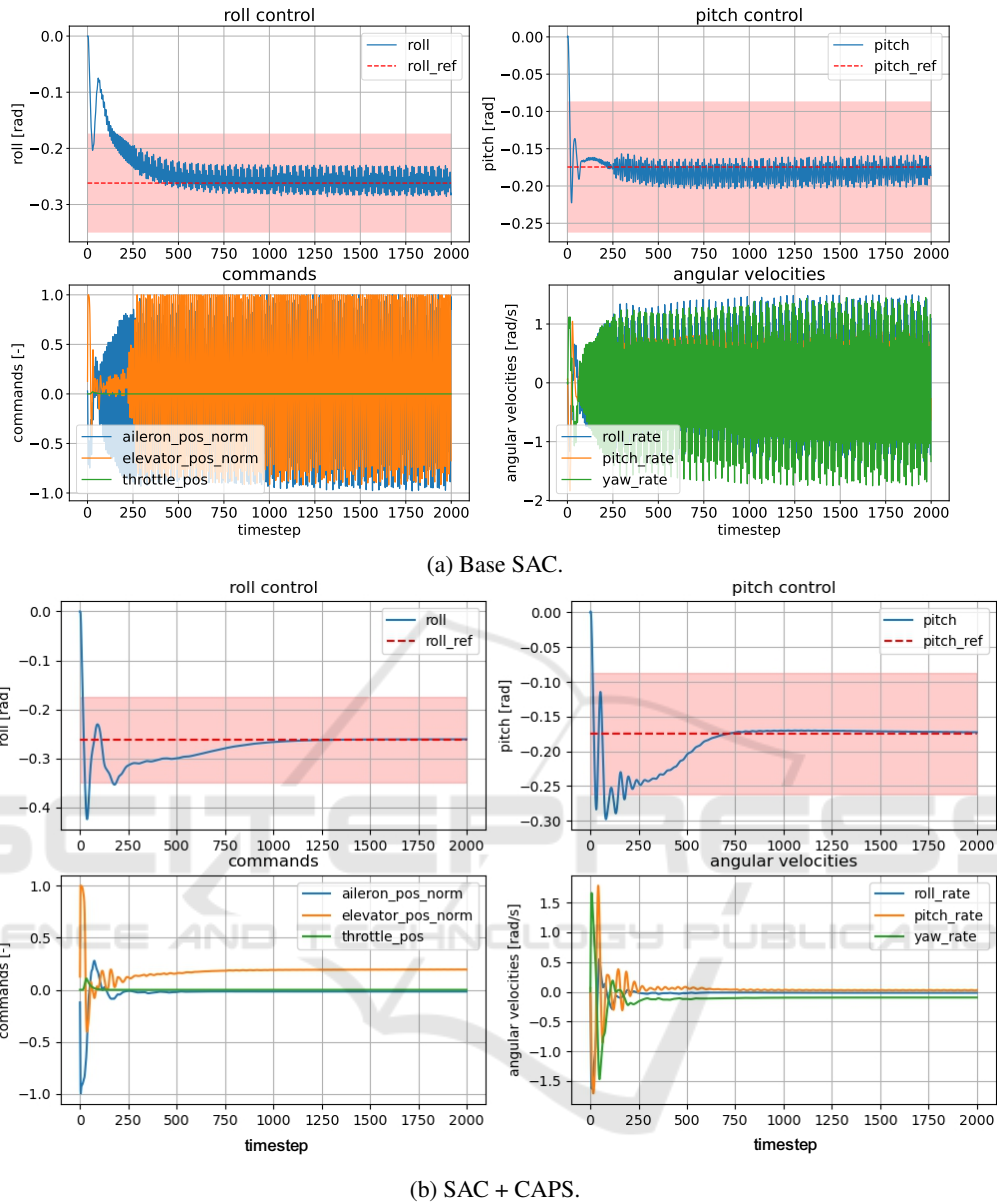
(a) Base SAC.

(b) SAC + CAPS.

Figure 5: Mitigating Highly Action Oscillating Policy: SAC vs SAC + CAPS. Red dashed lines are the references: Roll = $-15°$, Pitch = $-10°$. The red area around the reference line corresponds to $\pm5°$ error bounds.

7 CONCLUSIONS AND PERSPECTIVES

We proposed an in-depth study of different RL controllers with the aim of comparing model-free and model-based RL approaches. TD-MPC, an MB-RL method with algorithmic elements from CT and RL, yielded the best performance in nominal wind conditions. Its superiority especially shined for hard references and demonstrated its ability to perform well

for deterministic nonlinear dynamics across the entire state-space. We attribute such results to TD-MPC's learning of an explicit dynamics model jointly used with predictive planning. We evaluated these control methods under various wind perturbations and found that, aside from SAC underperforming, turbulence severity significantly impacts final performance more than the choice of algorithm.

We identified high actuation fluctuation as an important drawback of RL methods. Since this metric is is key in robotics, we applied and tested counter-

measures, namely: action variation reward penalty (AVP) for all methods and regularization of the actor network through CAPS for the MF-RL methods. As a result, we retained CAPS as action regulation method due to its consistency and ease of tuning across all MF-RL agents. Another path to smooth actions for TD-MPC (for which CAPS is not applicable) is MoDeM-v2 (Lancaster et al., 2023) which aims at learning safe policies by biasing the initial data distribution towards a desired behavior, in our case an action smooth controller.

We identify several future research directions: one could focus on experimenting with probabilistic models for TD-MPC in order to better capture the stochasticity of turbulence dynamics, thus formulating a more rigorous transition function that outputs a probability distribution. For both gusts and turbulence, RL algorithms with recurrent networks appear to be a good starting point (Ni et al., 2022; Asri and Trischler, 2019) to solve PO-MDPs. One could also use ideas from robust-RL to achieve disturbance resiliency of RL controllers (Hsu et al., 2024). Other directions include learning-based adaptive control as a field of potential mixed control methods (Shi et al., 2019; Doukhi and Lee, 2019), where a closed form of the nominal dynamics is used together with a feedforward component of the unknown, disturbance dynamics predicted by a learned model.

REFERENCES

(1980). Flying qualities of piloted airplanes, military specification. Technical report, MIL-F-8785C.

Andrychowicz, O. M., Baker, B., Chociej, M., Jozefowicz, R., McGrew, B., Pachocki, J., Petron, A., Plappert, M., Powell, G., Ray, A., et al. (2020). Learning dexterous in-hand manipulation. *The International Journal of Robotics Research*, 39(1):3–20.

Asri, L. E. and Trischler, A. (2019). A study of state aliasing in structured prediction with RNNs. *arXiv preprint arXiv:1906.09310*.

Beard, R. W. and McLain, T. W. (2012). *Small Unmanned Aircraft: Theory and Practice*. Princeton University Press.

Becker-Ehmck, P., Karl, M., Peters, J., and van der Smagt, P. (2020). Learning to fly via deep model-based reinforcement learning. *arXiv preprint arXiv:2003.08876*.

Berndt, J. (2004). JSBSim: An open source flight dynamics model in C++. In *AIAA Modeling and Simulation Technologies Conference and Exhibit*, page 4923.

Bøhn, E., Coates, E. M., Moe, S., and Johansen, T. A. (2019). Deep reinforcement learning attitude control of fixed-wing UAVs using proximal policy optimization. In *International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 523–533.

Bøhn, E., Coates, E. M., Reinhardt, D., and Johansen, T. A. (2023). Data-efficient deep reinforcement learning for attitude control of fixed-wing UAVs: Field experiments. *IEEE Transactions on Neural Networks and Learning Systems*.

De Marco, A., D'Onza, P. M., and Manfredi, S. (2023). A deep reinforcement learning control approach for high-performance aircraft. *Nonlinear Dynamics*, 111(18):17037–17077.

Doukhi, O. and Lee, D. J. (2019). Neural network-based robust adaptive certainty equivalent controller for quadrotor UAV with unknown disturbances. *International Journal of Control, Automation and Systems*, 17(9):2365–2374.

Gryte, K., Hann, R., Alam, M., Roháč, J., Johansen, T. A., and Fossen, T. I. (2018). Aerodynamic modeling of the Skywalker x8 fixed-wing unmanned aerial vehicle. In *International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 826–835.

Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S. (2018). Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International Conference on Machine Learning*, pages 1861–1870. PMLR.

Hafner, D., Pasukonis, J., Ba, J., and Lillicrap, T. (2023). Mastering diverse domains through world models. *arXiv preprint arXiv:2301.04104*.

Hansen, N., Su, H., and Wang, X. (2024). TD-MPC2: Scalable, robust world models for continuous control. In *International Conference on Learning Representations (ICLR)*.

Hansen, N., Wang, X., and Su, H. (2022). Temporal difference learning for model predictive control. In *International Conference on Machine Learning*.

Hsu, H.-L., Meng, H., Luo, S., Dong, J., Tarokh, V., and Pajic, M. (2024). REFORMA: Robust reinforcement learning via adaptive adversary for drones flying under disturbances. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*.

Huang, S., Dossa, R. F. J., Ye, C., Braga, J., Chakraborty, D., Mehta, K., and Araújo, J. G. (2022). CleanRL: High-quality single-file implementations of deep reinforcement learning algorithms. *Journal of Machine Learning Research*, 23(274):1–18.

Hwangbo, J., Sa, I., Siegwart, R., and Hutter, M. (2017). Control of a quadrotor with reinforcement learning. *IEEE Robotics and Automation Letters*, 2(4):2096–2103.

Koch, W., Mancuso, R., West, R., and Bestavros, A. (2019). Reinforcement learning for UAV attitude control. *ACM Transactions on Cyber-Physical Systems*, 3(2):1–21.

Lambert, N. O., Drew, D. S., Yaconelli, J., Levine, S., Calandra, R., and Pister, K. S. (2019). Low-level control of a quadrotor with deep model-based reinforcement learning. *IEEE Robotics and Automation Letters*, 4(4):4224–4230.

Lancaster, P., Hansen, N., Rajeswaran, A., and Kumar, V. (2023). MoDem-V2: Visuo-motor world mod-

els for real-world robot manipulation. *arXiv preprint arXiv:2309.14236*.

Lee, J., Hwangbo, J., Wellhausen, L., Koltun, V., and Hutter, M. (2020). Learning quadrupedal locomotion over challenging terrain. *Science robotics*, 5(47).

Liang, X., Zheng, M., and Zhang, F. (2018). A scalable model-based learning algorithm with application to uavs. *IEEE control systems letters*, 2(4):839–844.

Liu, F., Dai, S., and Zhao, Y. (2021). Learning to have a civil aircraft take off under crosswind conditions by reinforcement learning with multimodal data and preprocessing data. *Sensors*, 21(4):1386.

Mathisen, S., Gryte, K., Gros, S., and Johansen, T. A. (2021). Precision deep-stall landing of fixed-wing uavs using nonlinear model predictive control. *Journal of Intelligent & Robotic Systems*, 101:1–15.

McCallum, A. K. (1996). *Reinforcement learning with selective perception and hidden state*. University of Rochester.

Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., and Riedmiller, M. (2013). Playing Atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*.

Moallemi, M. and Towhidnejad, M. (2016). B-737 autopilot design and implementation for simulated flight management system. In *Proceedings of the 49th Annual Simulation Symposium*, pages 1–7.

Mysore, S., Mabsout, B., Mancuso, R., and Saenko, K. (2021). Regularizing action policies for smooth control with reinforcement learning. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1810–1816.

Ni, T., Eysenbach, B., and Salakhutdinov, R. (2022). Recurrent model-free RL can be a strong baseline for many pomdps. In *International Conference on Machine Learning*, pages 16691–16723. PMLR.

Peng, X. B., Coumans, E., Zhang, T., Lee, T.-W., Tan, J., and Levine, S. (2020). Learning agile robotic locomotion skills by imitating animals. *arXiv preprint arXiv:2004.00784*.

Schrittwieser, J., Antonoglou, I., Hubert, T., Simonyan, K., Sifre, L., Schmitt, S., Guez, A., Lockhart, E., Hassabis, D., Graepel, T., et al. (2020). Mastering Atari, Go, Chess and Shogi by planning with a learned model. *Nature*, 588(7839):604–609.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. (2017). Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Seyde, T., Gilitschenski, I., Schwarting, W., Stellato, B., Riedmiller, M., Wulfmeier, M., and Rus, D. (2021). Is bang-bang control all you need? solving continuous control with bernoulli policies. *Advances in Neural Information Processing Systems*, 34:27209–27221.

Shi, G., Azizzadenesheli, K., O'Connell, M., Chung, S.-J., and Yue, Y. (2021). Meta-adaptive nonlinear control: Theory and algorithms. *Advances in Neural Information Processing Systems*, 34:10013–10025.

Shi, G., Shi, X., O'Connell, M., Yu, R., Azizzadenesheli, K., Anandkumar, A., Yue, Y., and Chung, S.-J. (2019). Neural lander: Stable drone landing control using

learned dynamics. In *International Conference on Robotics and Automation*, pages 9784–9790.

Song, X., Duan, J., Wang, W., Li, S. E., Chen, C., Cheng, B., Zhang, B., Wei, J., and Wang, X. S. (2023a). LipsNet: a smooth and robust neural network with adaptive Lipschitz constant for high accuracy optimal control. In *International Conference on Machine Learning*, pages 32253–32272. PMLR.

Song, Y., Romero, A., Müller, M., Koltun, V., and Scaramuzza, D. (2023b). Reaching the limit in autonomous racing: Optimal control versus reinforcement learning. *Science Robotics*, 8(82).

Steimle, L. N., Kaufman, D. L., and Denton, B. T. (2021). Multi-model Markov decision processes. *IISE Transactions*, 53(10):1124–1139.

Sutton, R. S. and Barto, A. G. (2018). *Reinforcement learning: An introduction*. MIT press.

Tsourdos, A., Permana, I. A. D., Budiarti, D. H., Shin, H.-S., and Lee, C.-H. (2019). Developing flight control policy using deep deterministic policy gradient. In *IEEE International Conference on Aerospace Electronics and Remote Sensing Technology (ICARES)*.

Williams, G., Aldrich, A., and Theodorou, E. (2015). Model predictive path integral control using covariance variable importance sampling. *arXiv preprint arXiv:1509.01149*.