Learning-based Trajectory Tracking for Bird-inspired Flapping-Wing Robots

Jiaze Cai*, Vishnu Sangli*, Mintae Kim and Koushil Sreenath

***Abstract*— Bird-sized flapping-wing robots offer significant potential for agile flight in complex environments, but achieving agile and robust trajectory tracking remains a challenge due to the complex aerodynamics and highly nonlinear dynamics inherent in flapping-wing flight. In this work, a learning-based control approach is introduced to unlock the versatility and adaptiveness of flapping-wing flight. We propose a model-free reinforcement learning (RL)-based framework for a high degree-of-freedom (DoF) bird-inspired flapping-wing robot that allows for multimodal flight and agile trajectory tracking. Stability analysis was performed on the closed-loop system comprising of the flapping-wing system and the RL policy. Additionally, simulation results demonstrate that the RL-based controller can successfully learn complex wing trajectory patterns, achieve stable flight, switch between flight modes spontaneously, and track different trajectories under various aerodynamic conditions.**

I. Introduction

Flapping-wing Micro Aerial Vehicles (FMAVs) offer tremendous potential to achieve efficient and agile flight in complex, cluttered environments by mimicking the capabilities of their biological counterparts. These vehicles, capable of simultaneously generating lift, propulsion, and control forces, can perform maneuvers beyond the reach of conventional fixed-wing or rotary-wing aircrafts [1]–[3]. Compared to insect-inspired FMAVs, which excel in agility and hovering, bird-inspired FMAVs (commonly referred to as ornithopters) offer superior efficiency and robustness for sustained, long-distance flights. However, achieving robust, agile, and precise trajectory tracking control for bird-inspired FMAVs poses significant challenges. In particular, several key obstacles across various aspects of bird-inspired FMAV's design and operation hinder the development and performance of effective control systems for these vehicles.

A. Challenges for Control of Bird-inspired FMAVs

Modeling. The bird-inspired FMAVs exhibit highly nonlinear aerodynamics [2]. Birds and bird-inspired FMAVs typically operate within a Reynolds number range of 10^3 to 10^6, where the boundary layer is prone to laminar-turbulent transitions, significantly affecting the aerodynamic characteristics of flapping flight. These flow transitions are difficult to model accurately [4]. Additionally, unlike insect-scale or hummingbird-scale hoverable FMAVs (where wing inertia is often negligible), the larger wings of bird-inspired FMAVs introduce significant inertia effects. This necessitates the application of a multi-body dynamics model to capture the full complexity of the system's movements and interactions [5]. Furthermore, many of the current bird-inspired FMAVs feature flexible wings that couple with the aerodynamic forces acting on them [6]. The deformation of these wings during flight plays a critical role in determining their aerodynamic properties. This aeroelastic coupling, where aerodynamic forces and structural deformation interact, further complicates the modeling of FMAVs and presents a substantial challenge to the development of effective control strategies [4].

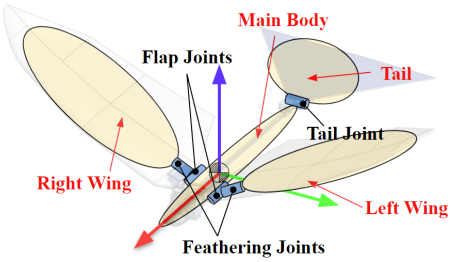


Fig. 1: The layout of the flapping-wing robot. In simulation, the flapping wing robot is modeled as 4 rigid ellipsoid bodies (yellow ellipses) for aerodynamic force computation mentioned in Section II-C with 5 revolute joints (blue cylinders). The bold red, green, and blue arrows represent the local xyz-frame of the vehicle.

Simulation. Given the current limitation on existing models in flapping-wing flight dynamics, aerodynamics, and aeroelastics, simulating the full flight dynamics is challenging. High-fidelity simulations are computationally intensive. For instance, Computational Fluid Dynamics - Computational Structure Dynamics (CFD-CSD) simulations usually take days to run. While the dynamics simulations for lower-fidelity models including Unsteady Vortex Latex Method (UVLM) and Unsteady Lifting Line Method are computationally feasible, they fail to account for viscous flow effects like leading-edge-vortex (LEV) and have a limited application range including the requirement of large wing aspect ratio and relatively small angle of attack for accurate results [7]–[10]. The lack of high-accuracy and low-computing-cost dynamical simulation environments for FMAVs is another challenge impeding the development of effective controllers for bird-inspired FMAVs.

Design. From the perspective of design, the limited weight budget arising from the nature of an aerial vehicle constrains the in-flight onboard computational power. This, in turn, limits the complexity of the controller despite the complex dynamics of the vehicle. In addition, the weight constraint also restricts the number of effective actuators in the design. Most of the recent bird-inspired FMAVs are driven by a link mechanism with a single motor with gearbox for flapping,

*Equal Contribution. The authors are with the Hybrid Robotics Group, University of California, Berkeley, CA 94720, United States.

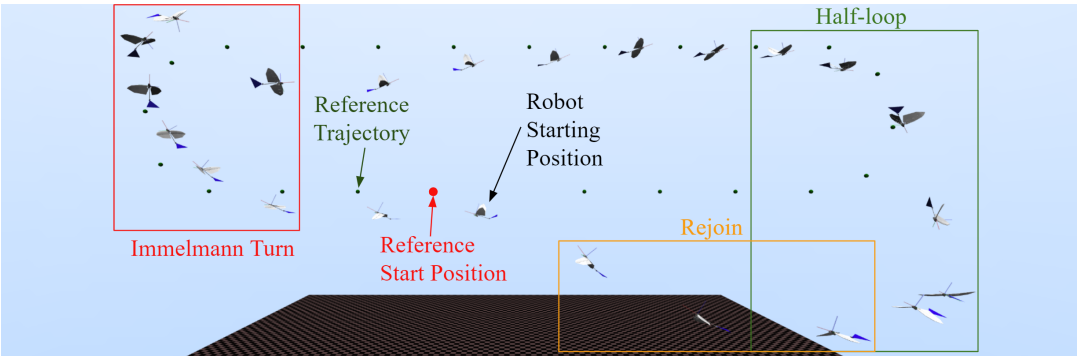


Fig. 2: The flapping-wing robot follows a loop trajectory generated from simulation. The robot performs an Immelmann turn (pitch up and roll back level), a half-loop maneuver (pitch down and roll back level), and a rejoin to the trajectory. The dark dots represent the reference points of the trajectory over time. Video results are posted at https://youtu.be/54Gcbvgfz7Q.

and controlling the vehicle by changing the motor speed and the angle of an actuated tail [11]. The limited control inputs restrict the agility and possible maneuvers of the FMAVs.

B. Related Work

FMAVs can be broadly categorized into two types: insect- or hummingbird-inspired FMAVs, which are optimized for near-hover flight, and bird-inspired FMAVs, which are primarily designed for forward flight [12]. Insect-sized FMAVs, due to their small size and lightweight structures, are typically more focused on hovering capabilities and agile maneuvers, often controlled with high-frequency wing flapping. In contrast, bird-inspired FMAVs are larger and more suitable for efficient forward flight, with the added complexity of managing the transition between various flight modes [13].

Traditional control approaches for these bird-inspired flapping-wing robots often rely on model-based methods with single-rigid-body dynamics, which are generally successful in limited environments but struggle with the nonlinear, highly dynamic aerodynamic forces and complex multi-degree-of-freedom movements inherent to flapping-wing flight [6]. These approaches include low-level attitude control systems designed for ornithopters, where proportional-integral-derivative (PID) techniques, and state feedback linearization are used to manage basic flight stabilization and orientation [6], [14].

Some progress has been made in optimization-based approaches for flight control of bird-sized FMAVs, as demonstrated in several studies [15]–[18]. Researchers have also explored model-based methods for trajectory tracking and generation, particularly in forward flight [19]–[23]. However, these systems usually result in limited accuracy due to the simplifications required to handle the complex dynamics of flapping-wing flight. Furthermore, the control mechanisms in such systems typically involve a small number of actuators, primarily focusing on the flapping frequency and the tail's pitch or yaw angles, which results in limited agility and maneuverability during flight.

In contrast, learning-based approaches have recently emerged to overcome the limitations of traditional control methods. Reinforcement learning (RL) has been successfully applied to various mobile platforms, including ground vehicles, drones, and legged robots, enabling robust and agile locomotion in highly dynamic environments [24]–[28]. However, its application in flapping-wing robots remains relatively underexplored. Some early studies have applied RL for specific tasks such as improving lift generation in butterfly-like MAVs or suppressing wing vibrations during wing trajectory tracking [29], [30]. Experimental work has also demonstrated the potential for RL in enhancing efficiency in lift generation and in modeling dynamic behavior in traditional control frameworks [31], [32]. RL-based control has particularly shown promise in enabling agile maneuvers, such as back-flips and escape behaviors, in insect-sized hoverable FMAVs [33]–[37]. Despite these advances, the use of RL in bird-sized FMAVs for forward flight remains an open area of research, with significant potential to unlock new levels of agility and performance.

C. Contribution

This paper presents a learning-based approach for trajectory tracking in flapping-wing robots using reinforcement learning. Our method leverages a simulation environment built in MuJoCo (Multi-Joint dynamics with Contact) [38] to model the robot's dynamics and aerodynamics, enabling us to train the RL policy under various conditions. We focus on developing a control framework that not only tracks predefined trajectories with high precision but also handles challenging maneuvers such as loops and Immelmann turns, as shown in Fig. 2. Furthermore, we explore the robustness of the proposed method in the presence of wind disturbances and varying aerodynamic conditions, demonstrating its versatility in real-world scenarios. The level of versatility in motions and the capability to manage high-DOF actuation achieved by this method are difficult to emulate using conventional approaches, as demonstrated in [6], [14]. In addition to introducing a novel control strategy, we demonstrate the stability of the controlled system and reveal its periodicity through a phase study.

II. Problem Description

A. Platform Overview

In this paper, we use a flapping-wing robot with a wingspan of 0.995m, a standard mean chord (average wing width)

of 0.17m, and a weight of 0.31kg, as shown in Fig. 1. A simplified model of the flapping-wing robot showing the joint and rigid bodies is shown in Fig. 1. Unlike many existing bird-inspired FMAVs, this robot features $n_j = 5$ individually controllable joints ($\mathbf{q}_j \in \mathbb{R}^5$) that independently actuate the wing and tail. Each wing is equipped with a flap joint (q_1 for the left, q_3 for the right) to control the flap angle, and a feathering joint (q_2 for the left, q_4 for the right) to control the wing pitch. Additionally, there is a dedicated joint q_5 to control the tail pitch. In total, the robot possesses 5 actuated DoFs. The robot also has a floating base $\mathbf{q}_b$ with 6 DoFs comprising translational positions $[q_x, q_y, q_z]^T$ and rotational positions $[q_\phi, q_\theta, q_\psi]^T$, which result in a total of $n = 11$ DoFs. Thus, the system has $n - n_j = 6$ degrees of underactuation. The full system's generalized coordinates $\mathbf{q}$ can be represented as $\mathbf{q} = [\mathbf{q}_b, \mathbf{q}_j]^T \in \mathbb{R}^n$.

B. Dynamical Model

Although the wings of a flapping-wing robot can be flexible, in this work, we assume all bodies in the robot are rigid for simplicity. The aerodynamic effect caused by the deformation of wings is handled by domain randomization mentioned in Table III. The full-order dynamical model of the flapping-wing robot system is derived with the Newton-Euler formulation:

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{G}(\mathbf{q}) = \begin{bmatrix} \mathbf{0}_6 \\ \boldsymbol{\tau} \end{bmatrix} + \mathbf{u}_{\text{aero}}(\mathbf{q}, \dot{\mathbf{q}}), \quad (1)$$

where $\mathbf{M}(\mathbf{q}) \in \mathbb{R}^{n\times n}$ is the mass matrix, $\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}) \in \mathbb{R}^{n\times n}$ represents the Coriolis and centrifugal forces, and $\mathbf{G}(\mathbf{q}) \in \mathbb{R}^n$ denotes the gravitational forces, $\boldsymbol{\tau} \in \mathbb{R}^{n_j}$ is the input joint torque, while $\mathbf{u}_{\text{aero}}(\mathbf{q}, \dot{\mathbf{q}}) \in \mathbb{R}^n$ represents the generalized aerodynamic forces. Note that the generalized aerodynamic force, $\mathbf{u}_{\text{aero}}$, can be calculated as

$$\mathbf{u}_{\text{aero}}(\mathbf{q}, \dot{\mathbf{q}}) = \sum_{i=1}^{m} \mathbf{J}_i^T(\mathbf{q}) \boldsymbol{F}_{\text{aero},i}(\mathbf{q}, \dot{\mathbf{q}}), \quad (2)$$

where $\mathbf{J}_i^T(\mathbf{q}) \in \mathbb{R}^{n\times 6}$ is the Jacobian transpose that maps Cartesian aerodynamics wrench $\boldsymbol{F}_{\text{aero},i}(\mathbf{q}, \dot{\mathbf{q}}) \in \mathbb{R}^6$ acting on each body, i, to the generalized coordinates, and m is the number of bodies considered in the fluid.

C. Simulation

MuJoCo is used as the physics simulator in this work, functioning as the environment for training and testing our control policy. In MuJoCo, the multi-body dynamics model mentioned in Section II-B is computed. MuJoCo also provides stateless fluid force models (i.e., the fluid force computation does not have its own dynamics with fluid states, but rather is only dependent on the state of the robot - a simplifying approximation that is made), which is used to model the Cartesian aerodynamic wrench $\boldsymbol{F}_{\text{aero}} = [\mathbf{f}_{\text{aero}}^T \ \boldsymbol{\tau}_{\text{aero}}^T]^T$ acting on each body i of the robot. MuJoCo provides two different fluid models: (1) a simplified Inertia Fluid Model that estimates the aerodynamic wrench based on its equivalent inertia box, and (2) a more elaborate Ellipsoid Fluid Model that accounts for the aerodynamic force and moment of 5 different aerodynamic effects on a projected ellipsoid [36]. For higher simulation accuracy, we model all lifting bodies, including wings and tail, with the more elaborate Ellipsoid Fluid Model and model the remaining bodies, i.e., the main body, with the Inertia Fluid Model considering the relatively small aerodynamic effect of the main body compared to the lifting bodies. This enables us to capture the most significant aerodynamic properties of the robot in MuJoCo with a relatively simple model.

TABLE I: Definition of variables used in the aerodynamics equations

Variable	Definition
$\mathbf{v}$	Velocity (vector) relative to airflow
$\mathbf{v}_{\parallel}$	Component of velocity parallel to surface
$\boldsymbol{\omega}$	Angular velocity (vector)
ρ	Fluid density
ν	Kinematic viscosity of the fluid
$\mathbf{m}_A$	Added mass (vector)
$\mathbf{I}_A$	Added moment of inertia (vector)
$C_{\text{D, blunt}}$	Drag coefficient for blunt body
$C_{\text{D, slender}}$	Drag coefficient for slender body
$C_{\text{D, angular}}$	Angular drag coefficient
C_K	Kutta lift coefficient
C_M	Magnus force coefficient
$A_{\mathbf{v}}^{\text{proj}}$	Projected area in the direction of velocity
$A_{\max}$	Maximum reference area
V	Volume of the body
$A_{\max}$	Maximum reference area
I_D	Reference moment of inertia for drag
r_V	Effective radius for viscous drag

The total aerodynamic force, $\mathbf{f}_{\text{aero}}$, and moment $\boldsymbol{\tau}_{\text{aero}}$ for a single rigid body in the fluid are calculated as follows: For Inertia Fluid Model,

$$\mathbf{f}_{\text{aero}} = \mathbf{f}_A + \mathbf{f}_V, \quad (3)$$

$$\boldsymbol{\tau}_{\text{aero}} = \boldsymbol{\tau}_A + \boldsymbol{\tau}_V, \quad (4)$$

and for Ellipsoid Fluid Model,

$$\mathbf{f}_{\text{aero}} = \mathbf{f}_A + \mathbf{f}_D + \mathbf{f}_M + \mathbf{f}_K + \mathbf{f}_V, \quad (5)$$

$$\boldsymbol{\tau}_{\text{aero}} = \boldsymbol{\tau}_A + \boldsymbol{\tau}_D + \boldsymbol{\tau}_V, \quad (6)$$

where subscripts A, D, M, K, and V represent added mass, viscous drag, Magnus lift, Kutta lift, and viscous resistance, respectively. The computation of each term is summarized as follows based on [36],

$$\begin{aligned}
\mathbf{f}_A &= -\mathbf{m}_A \circ \dot{\mathbf{v}} + (\mathbf{m}_A \circ \mathbf{v}) \times \boldsymbol{\omega}, \\
\boldsymbol{\tau}_A &= -\mathbf{I}_A \circ \dot{\boldsymbol{\omega}} + (\mathbf{m}_A \circ \mathbf{v}) \times \mathbf{v} + (\mathbf{I}_A \circ \boldsymbol{\omega}) \times \boldsymbol{\omega}, \\
\mathbf{f}_D &= -\rho \left[C_{D,\text{blunt}} A_{\text{proj}}^{\mathbf{v}} + C_{D,\text{slender}} (A_{\max} - A_{\text{proj}}^{\mathbf{v}}) \right] \|\mathbf{v}\| \mathbf{v}, \\
\boldsymbol{\tau}_D &= -\rho \left[C_{D,\text{angular}} I_D + C_{D,\text{slender}} (I_{\max} - I_D) \right] \boldsymbol{\omega}, \\
\mathbf{f}_M &= C_M \rho V \boldsymbol{\omega} \times \mathbf{v}, \\
\mathbf{f}_K &= C_K \rho A_{\text{proj}}^{\mathbf{v}} \|\mathbf{v}\| (\mathbf{v} \times \mathbf{v}_{\parallel}) \times \mathbf{v}, \\
\mathbf{f}_V &= -6\pi r_V \nu \mathbf{v}, \\
\boldsymbol{\tau}_V &= -8\pi r_V^3 \nu \boldsymbol{\omega},
\end{aligned} \quad (7)$$

where $\circ$ is defined as element-wise multiplication, and each of the terms used is defined in Table I. Note that in this work, the fluid coefficients $C_{D,\text{blunt}}$, $C_{D,\text{slender}}$, $C_{D,\text{angular}}$, C_K, and C_M are manually tuned to match the designed lift-to-drag

TABLE II: Aerodynamic coefficients used in this work.

Coefficient	$C_{\text{D, blunt}}$	$C_{\text{D, slender}}$	$C_{\text{D, angular}}$	C_K	C_M
Value	0.2	0.12	1.5	3.14	1

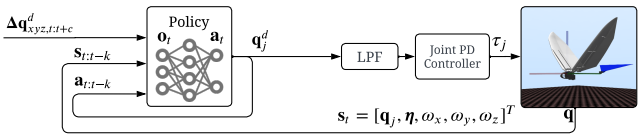


Fig. 3: The control diagram for flapping-wing robot trajectory tracking control. The variables used here are covered in Section III-C.

ratio of the FMAV at gliding, given that the horizontal-to-vertical distance covered by the FMAV is equal to the lift-to-drag ratio when gliding in steady state. The resultant coefficients are provided in Table II.

III. RL-Based Trajectory Tracking Control

A. Control Framework

Our goal is to develop an RL controller framework for a bird-inspired FMAV that can track target trajectories and achieve bird-like maneuverability. As seen in Fig. 3, we use an RL-trained policy that takes observations as input and outputs actions that are scaled and smoothed via a low-pass filter into target joint positions. These are then used by the low-level PD controller to compute motor torques. The policy operates at 50 Hz, while the low-level joint PD controller runs at the simulation frequency of 250 Hz.

One important aspect of our framework was ensuring energy-efficient behavior, since policies tend to prefer unrealistically high flapping frequencies when left unconstrained. Aside from the enforced energy penalties during training, the low-pass filter was given a 7Hz cutoff frequency to limit the viable flapping frequencies to 4-6Hz.

Due to the difficulty of adapting to a variety of arbitrary maneuvers as well as lack of a "safe state" in our state space, we found it difficult to train such a policy from scratch. Therefore, we use a curriculum-based training scheme involving 3 stages of increasing difficulty to train a controller: (1) constant forward flight, (2) climbing and diving at variable speeds, (3) turning and arbitrary maneuvers. Following this, Dynamics randomization was an additional stage to improve the robustness and adaptability of trained controllers, where we randomized parameters relating to rigid body dynamics, motor-level characteristics, and the aerodynamics and wind condition faced by the system.

B. Target Trajectories

We procedurally generate 3D position trajectories $\mathbf{Q}^d_{xyz}(t)$ that are defined by simple linear and circular paths. The policy is provided with a look-ahead buffer of the upcoming trajectory and is rewarded based on its position error against the current target position along the trajectory. These simple linear and circular trajectories were chosen over trajectories obtained through model-based optimization or Bézier curves [39] due to uncertainty in aerodynamic feasibility. The robot's flight path is not tightly constrained to this target trajectory to allow the policy to optimize its movement according to the aerodynamics of the system. Hence, this ensures that the learning is unconstrained and natural, with the resulting emergent behavior dictating its own orientation to fit the desired movement. With forward speed, Z-axis velocity, and global angular yaw rate, we can specify commands that combine the four basic flying skills: flying straight, diving, climbing, and turning. The duration of each command is fixed at 3 seconds, enabling the policy to follow arbitrary paths and become robust to skill transitions. Vertical loops were also modeled to simulate aerobatic maneuvers like back-flips and Immelmann turns.

C. State and Action Spaces

The controller outputs actions $\mathbf{a}_t \in \mathbb{R}^{n_j}$ (n_j is the number of controllable joints as mentioned in Section II-A) that specify the desired actuated joint positions $\mathbf{q}^d_j$, which are used by the low-level joint PD controller to compute the motor torque. The action space is centered in the nominal pose (wings and tail flat at 0 degrees) and is normalized to the respective joint limits.

The observation space of the policy, $\boldsymbol{o}_t \in \mathbb{R}^{490}$, consists of sensor-related observations and trajectory information. The sensor observations for each time frame include the orientation quaternion $\boldsymbol{\eta}$, local angular velocity p, q, r, joint motor positions $\mathbf{q}_j$, and a local x-velocity measurement relative to the wind $\mathbf{v}_{x,\text{air}}$. The latter sensor is modeled after a pitot tube and acts as the most plausible form for a physical flapping robot to receive onboard velocity readings. The policy receives a history of these sensor readings and past action outputs for the past 25 steps, corresponding to a window of 0.5s. This history allows the policy to infer the dynamics of the system. The policy also receives 5 trajectory points spaced evenly over 0.6s of the upcoming trajectory $\mathbf{Q}^d_{xyz}(t)$ in local-frame relative coordinates $\Delta \mathbf{q}^d_{xyz}$. Although shorter trajectory windows of 0.2s have also resulted in well-performing policies, larger windows are included to allow for smooth flapping behavior.

D. Rewards

The heuristics for our reward r for the policy are kept constant across all stages of training: to minimize position error relative to the target trajectory while preserving balance. The reward function has four parts

$$r = 0.5r_{\text{pos}} + 0.1r_{\boldsymbol{\Omega}} + 0.2r_{\phi,\theta} + 0.05r_{\text{energy}} \tag{8}$$

where r_{pos} is the position tracking term that minimizes body position error to the current desired position on the target trajectory. $r_{\boldsymbol{\Omega}}$ attempts to minimize the main body angular rates $\omega_x, \omega_y, \omega_z$ to ensure smoother and stable behavior. $r_{\phi,\theta}$ rewards a lower roll and pitch, motivating the robot to stay leveled. Although a bird's pitch and roll are involved in its flight movements, this reward ensures that the learned behavior varies its orientation only when needed. This is because a bird receives maximal horizontal thrust when oriented horizontally, and hence this reward encourages the policy to maximize its time in this orientation. r_{energy} is the

TABLE III: The range of dynamics randomization.

Parameters	Range
Joint Damping Ratio	[0.9, 1.1] Nms/rad
Link Mass & Link Inertia	$\pm 10\% \times$ default
Link CoM Position Offset	[-0.05, 0.05] m
Aerodynamic Coefficients	$30\% \times$ default
Aerodynamic Added Mass & Inertia	$10\% \times$ default
Wind x,y,z Velocity	[$\pm$2], [$\pm$2], [$\pm$1.5] m/s

energy term [40] that motivates energy-efficient behavior, i.e., gliding instead of flapping when possible, mimicking physical birds. This term also serves the purpose of discouraging high flapping frequencies so that the learned frequency resembles those of physical birds of the same wing area. The reward weights were manually tuned to ensure fast and stable learning.

E. Dynamics and Aerodynamic Force Randomization

We have incorporated the randomization of dynamic parameters in our training framework in order to better prepare policies for the sim-to-real gap. Similar to previous work [39], randomization was applied to rigid body mass, inertia, and center of mass.

Although the Mujoco Fluid model is capable of capturing salient aerodynamic effects using ellipsoids, it still has limited accuracy to simulate the real physics of the flapping-wing robot. This is due to three main reasons: (1) the hand-tuned fluid coefficients may not match the physical model, (2) it is difficult for the static (without fluid states) characteristic of the MoJoCo aerodynamic model to capture the dynamic unsteady aerodynamic force (with fluid states) unsteady aerodynamic force in highly dynamical motions, and (3) the deformation on the physical wing (which is not modeled) can cause variations in the resultant aerodynamic forces. To overcome this deviation between the simulation and the real world, a randomization on the aerodynamics is incorporated. This includes randomizing the five different fluid coefficients, the added mass $\mathbf{m}_A$, and the added inertia $\mathbf{I}_A$ mentioned in Table I on each body. Additional wind disturbances with randomized direction and magnitude were also implemented to improve the robustness and adaptability of the policy. The detailed range of dynamics randomization is shown in Table III. The initial position and velocity are also randomized to ensure that the policy can robustly recover and rejoin trajectories.

F. Episode Design

Each episode lasts a maximum of 30 s, corresponding to 1,500 control steps. A position error termination condition is enforced throughout all stages of control, terminating episodes if the robot deviates more than 3 meters from the target position. This condition accelerates early training and improves the precision of trajectory tracking while allowing the policy to optimize its own flight path. Additionally, an orientation termination condition is also applied during the first two stages of training to ensure that the robot's absolute roll $|q_\phi|$ and pitch $|q_\theta|$ never exceeded 90° when learning basic flight. This constraint is removed when training later stages that involve arbitrary paths and extreme maneuvers. We trained our policy with a multilayer perceptron (MLP) actor-critic architecture with Proximal Policy Optimization (PPO) [41] employing 256 parallel environments. Each training stage required approximately 8 million steps, with a full controller trained in about 30 million steps. The training was carried out on an Intel Core i7-11800H CPU and took around 5 hours to train the full controller.

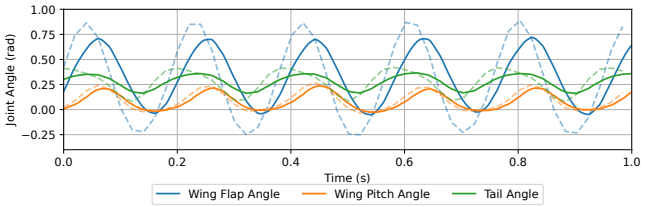


Fig. 4: Forward flapping behavior of the controller shown through time series of the wing pitch angle, wing flap angle, and tail angle in 1s. The dashed lines are target joint positions while the solid lines indicate the actual joint positions.

IV. Results

We now validate the performance of the policies trained for our control framework. Fig. 4 shows 1 second of flapping at a speed of 3.8 m/s. The wing flapping, wing pitch, and tail angles were found to have a fundamental frequency of 5.3 Hz with energies of 38.65%, 36.81%, and 38.24% respectively, indicating reasonable fit to a single frequency mode.

A. Stability Analysis of the Flight Controller

1) System Identification

To analyze the dynamics of the RL-based controller and validate the stability of the control system, a model derived from system identification using input-output pairs is introduced, as shown in Fig. 5. Following a method similar to [42], a low-dimensional linear system is extracted from the flapping-wing robot under the control of the RL-based controller, which is then employed to demonstrate its stability. The input u of the closed-loop system is the desired position $[q_x^d, q_y^d, q_z^d]^T \in \mathbb{R}^3$ while the output of the system is $\mathbf{y} = [\hat{q}_x, \hat{q}_y, \hat{q}_z]^T$. We develop a linear model that approximates the behavior of the closed-loop system governed by the model-free RL-based policy. The LTI system is obtained by fitting the input-output pairs, where the input of the closed system is determined by the input of the policy network. The fitted input-output dynamics of position is given by:

$$Y_x(s) = \frac{49.89s^2 + 164.9s + 26.27}{s^3 + 3.554s^2 + 6.438s + 2.809} \tag{9}$$

$$Y_y(s) = \frac{-0.09798s^2 - 10.07s - 24.67}{s^3 + 3.554s^2 + 6.438s + 2.809} \tag{10}$$

$$Y_z(s) = \frac{1.006s^2 + 1.020s + 3.836}{s^3 + 3.554s^2 + 6.438s + 2.809} \tag{11}$$

Note that the derived linear model's predicted result had a Mean Squared Error (MSE) as low as 5.629×10^{-5} against the actual system output.

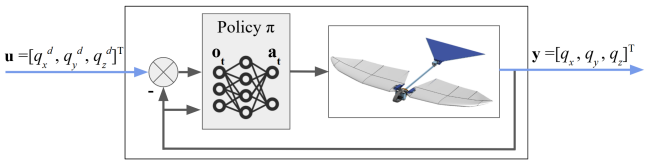
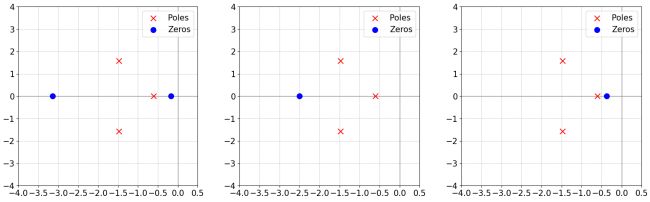


Fig. 5: System identification is performed on the closed-loop system. A low-dimensional system is derived from the high-dimensional, nonlinear dynamics of the flapping-wing robot, which is controlled by its RL policy. The input, $\mathbf{u}$, of this simplified system is the desired global position, while the output, $\mathbf{y}$, represents the robot's measured response, driven by the low-level RL policy.



(a) Pole-zero of q_x (b) Pole-zero of q_y (c) Pole-zero of q_z

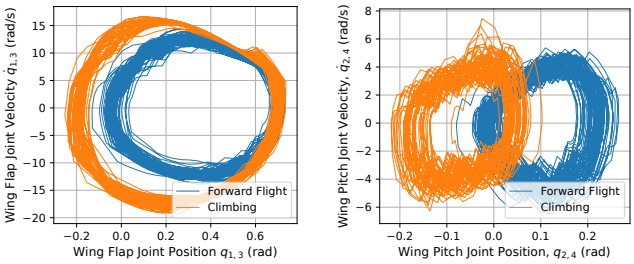
Fig. 6: Pole-zero plots for the closed-loop system's linear dynamics in three dimensions.

2) Stability

We assess the stability of the closed-loop system's linear dynamics by examining the pole positions of the above transfer functions. The plots of poles and zeros for all three dimensions are presented in Fig. 6. Based on the analysis, all identified linear systems exhibit Bounded Input Bounded Output (BIBO) stability, as all poles are located in the left-half plane (LHP). This shows that the input-output dynamics of the closed-loop system comprised of the nonlinear FMAV controlled by the RL policy is locally input-output stable. Additionally, all zeros are located in the LHP, confirming that the system is minimum phase in all dimensions. The result implies that the input-output relationship of our system does not exhibit non-minimum phase behavior, which can often be found in conventional fixed-wing aircraft, where tail-down effects precede tail-up behavior. Therefore, the choice of input-output plays a critical role in achieving a minimum phase, making the closed-loop system more controllable and stabilizable.

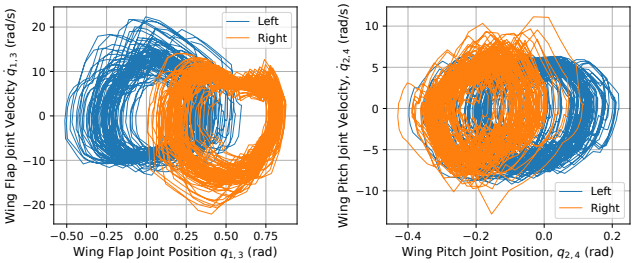
B. Phase portraits of Flying Tasks

To illustrate the system's periodicity under different tasks, we present phase portraits of the wing flap and wing pitch joints when undergoing forward flight, climbing, and turning. These portraits exhibit closed, consistent periodic orbits, indicating that the robot's flapping and pitching are stable and periodic throughout flight. Fig. 7 shows wing's pitch and flap angles when flying forward and climbing at $3\,\mathrm{m/s}$ forward velocity. We plot the phase of only the right wing as the robot's symmetry ensures identical behavior on the left. As shown in Fig. 7 (a), climbing flight tends to have a larger range of motion for flap joint position and velocity, which is consistent with the fact that robot consumes more



(a) Wing Flap joint (b) Wing Pitch joint

Fig. 7: Comparing phase portraits between cruising and climbing



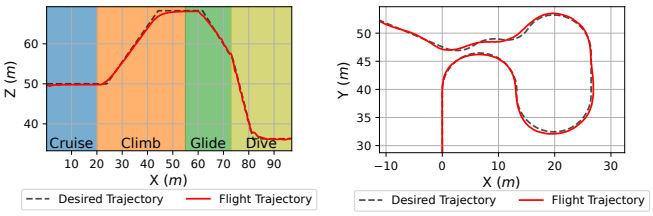
(a) Wing Flap joint (b) Wing Pitch joint

Fig. 8: Comparing left and right wing phase portraits when turning left

energy when climbing up. On the other hand, the average pitch joint angles in Fig. 7 (b) shift to negative values while retaining the same periodic motion. Note that the negative pitch angle is consistent with the body frame, where pitching up corresponds to more negative values. This implies that the policy tends to acquire a higher angle of attack to gather more lift for climbing. We then compare the flap and pitch joints between the left and right wings while gradually turning left, as seen in Fig. 8. The right wing is raised (higher flap joint position) and pitches up (more negative pitch joint position) to generate more lift on the right. This results in higher lift and thrust generation on the right wing, thereby turning left.

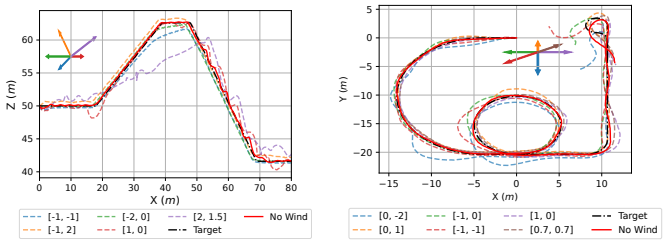
C. Trajectory Tracking Performance

A series of trajectories were generated to validate the performance of the trained policies. Simple trajectories demonstrating fundamental flight skills like cruising, climbing, gliding, diving, and turning are shown in Fig. 9. The target lookahead buffer from Section III-B allows the policies to anticipate command changes and accordingly optimize its flight path. We designed an aerobatic trajectory to demonstrate the versatility of the closed-loop RL controller on the FMAV, shown in Fig. 13. The corresponding snapshots of each maneuver in the same trajectory are shown in Fig. 12. These results show the RL controller's capability to track highly dynamic trajectories by utilizing various combinations of control inputs to manipulate its attitude. A few other examples of flapping-wing robot tracking trajectories are also shown in Fig. 2 and posted at https://youtu.be/54Gcbvgfz7Q. Note that there is a noticeable tracking error when we provide



(a) Longitudinal Flight Tasks (b) Lateral Flight tasks

Fig. 9: Example longitudinal and lateral trajectories to display the controller's tracking performance. (a) A series of cruising (blue), climbing (orange), gliding (green), diving (yellow) in 24 seconds with a 3.8m/s follow velocity. (b) A series of gradual and quick turns in 18 seconds with a $5.3\,\mathrm{m/s}$ follow velocity.



(a) Simple forward flight tasks. (b) Turning

Fig. 10: Effect of wind on trajectory following. The legend shows the 2D wind vectors in $\mathrm{m/s}$ in the corresponding plane for each respective trajectory. These vectors are also visualized in the plot.

a potentially dynamically infeasible target trajectory, but the FMAV is capable of rejoining the trajectory to some extent. Fig. 10 shows the controller's robustness to wind and aerodynamics randomization, illustrating each wind vector and fluid coefficient's influence on the flight path. This demonstrates that the FMAV is most sensitive to C_K, the Kutta lift coefficient, while still achieving a relatively high success rate even with other fluid coefficients randomized by up to a factor of 0.5. The high sensitivity to C_K can be caused by its dominant role in generating lift for bird-scale flapping flight.

V. Conclusion and Future Work

In this work, we proposed a novel reinforcement learning (RL)-based framework for trajectory tracking in bird-inspired flapping-wing robots. This developed control system demonstrates the ability to track complex 3D trajectories, perform agile maneuvers, and adapt to varying aerodynamic conditions in simulation. By leveraging MuJoCo's multi-body dynamic modeling and aerodynamic randomization, we ensured that the RL policy generalized well to diverse flight scenarios, including wind disturbances and different aerodynamics. Our stability analysis validated that the closed-loop system was both asymptotically stable and capable of maintaining stable, periodic joint action patterns.

It is worthwhile mentioning that the hardware platform of the flapping-wing robot is in the process of design. Our future

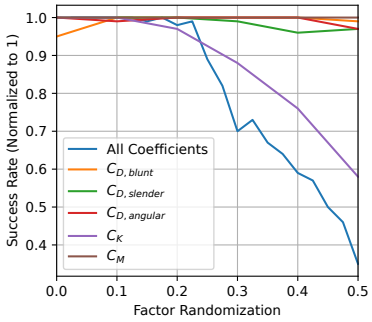


Fig. 11: Effect of randomizing specific fluid coefficients while others remain at the nominal values in Table II. The success rate is the fraction of 100 randomly sampled episodes that were within 3m at the end of the Fig. 10 (a) trajectory. All coefficients are uniformly randomized in the blue curve.

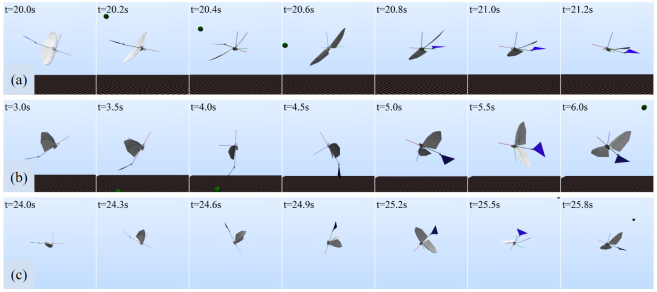


Fig. 12: Snapshots of three difference aerobatic maneuvers: (a) A sharp 180° turn maneuver in $1.2\,\mathrm{s}$. (b) A loop maneuver (back-flip) over $3\,\mathrm{s}$. (c) A roll-off-the-bottom maneuver (half loop pitching down followed by a roll back to level) in $1.8\,\mathrm{s}$.

work will involve experimental validation of the proposed RL policy once we design and build a flapping-wing robot.

Acknowledgement

This work is inspired from discussions on dynamics and control of FMAVs with A. Ramezani and B. Gupta from Northeastern University. This work is supported in part by National Science Foundation Grant CMMI-2140650 and in part by The Robotics and AI Institute. We thank T. Guo and Z. Li for developing preliminary RL policies related to this project. We also thank Q. Liao for the technical discussions and R. Zhang for proofreading. K. Sreenath has financial interests in the Robotics and AI Institute. He and the company may benefit from the commercialization of the results of this research.

References

[1] W. Shyy, M. Berg, and D. Ljungqvist, "Flapping and flexible wings for biological and micro air vehicles," *Progress in Aerospace Sciences*, vol. 35, no. 5, pp. 455–505, 1999.

[2] W. Shyy, H. Aono, S. Chimakurthi, P. Trizila, C.-K. Kang, C. Cesnik, and H. Liu, "Recent progress in flapping wing aerodynamics and aeroelasticity," *Progress in Aerospace Sciences*, vol. 46, no. 7, pp. 284–327, 2010.

[3] Y. Bayiz, M. Ghanaatpishe, H. Fathy, and B. Cheng, "Hovering efficiency comparison of rotary and flapping flight for rigid rectangular wings via dimensionless multi-objective optimization," *Bioinspiration & Biomimetics*, vol. 13, p. 046002, may 2018.

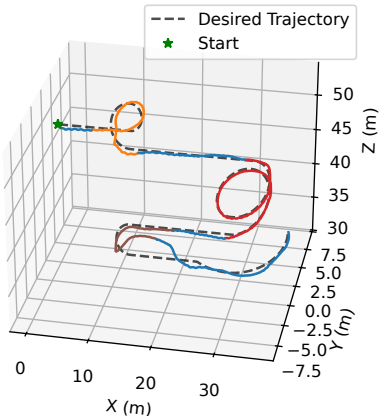


Fig. 13: An aerobatic trajectory consisting of a loop (orange), turns (red), and a roll-off-the-bottom maneuver (brown). The blue sections show cruising and recovery.

[4] H. E. Taha, M. R. Hajj, and A. H. Nayfeh, "Flight dynamics and control of flapping-wing mavs: a review," *Nonlinear Dynamics*, vol. 70, no. 2, pp. 907–939, 2012.

[5] M. Khosravi and A. B. Novinzadeh, "Comment on "modeling and simulation of nonlinear dynamics of flapping wing micro air vehicles"," *AIAA Journal*, vol. 57, no. 5, pp. 2195–2197, 2019.

[6] S. B. XUE Dong, ZHU Ziwen, "Key technologies of bird inspired flapping-wing micro aerial vehicles: Review," *Chinese Journal of Aeronautics*, vol. 45, no. 17, 2024.

[7] J. Murua, R. Palacios, and J. M. R. Graham, "Applications of the unsteady vortex-lattice method in aircraft aeroelasticity and flight dynamics," *Progress in Aerospace Sciences*, vol. 55, pp. 46–72, 2012.

[8] J. Boutet and G. Dimitriadis, "Unsteady lifting line theory using the wagner function for the aerodynamic and aeroelastic modeling of 3d wings," *Aerospace*, vol. 5, no. 3, 2018.

[9] C. Urban and R. K. Agarwal, "Validation and optimization of ptera software: An open-source unsteady flow simulator for flapping wings," in *AIAA SCITECH 2022 Forum*, p. 1967, 2022.

[10] E. Sihite, P. Ghanem, A. Salagame, and A. Ramezani, "Unsteady aerodynamic modeling of aerobat using lifting line theory and wagner's function," in *IROS 2022*, pp. 10493–10500, IEEE, 2022.

[11] S. Kim, M.-S. Kim, S. Kim, and J. Suk, "Design, fabrication, and flight test of articulated ornithopter," in *Proceedings of the 10th International Micro Air Vehicles Conference, Melbourne, Australia*, pp. 22–23, 2018.

[12] M. F. Bin Abas, A. S. Bin Mohd Rafie, H. Bin Yusoff, and K. A. Bin Ahmad, "Flapping wing micro-aerial-vehicle: Kinematics, membranes, and flapping mechanisms of ornithopter and insect flight," *Chinese Journal of Aeronautics*, vol. 29, no. 5, pp. 1159–1177, 2016.

[13] S. Liang, B. Song, and J. Xuan, "Active disturbance rejection attitude control for a bird-like flapping wing micro air vehicle during automatic landing," *IEEE Access*, vol. 8, pp. 171359–171372, 2020.

[14] J. Z. Torres, J. Davila, and R. Lozano, "Attitude and altitude control on board of an ornithopter," in *2016 ICUAS*, pp. 1124–1130, 2016.

[15] B. Gupta, Y. Shah, T. Liu, E. Sihite, and A. Ramezani, "Banking turn of high-dof dynamic morphing wing flight by shifting structure response using optimization," *arXiv preprint arXiv:2405.05490*, 2024.

[16] B. Gupta, A. Dhole, A. Salagame, X. Niu, Y. Xu, K. Venkatesh, P. Ghanem, I. Mandralis, E. Sihite, and A. Ramezani, "Bounding flight control of dynamic morphing wings," in *2024 IEEE AIM*, pp. 100–105, 2024.

[17] B. Zhu, Z. Zuo, L. Sun, Y. Zou, and K. Xia, "Model predictive control for a 3-dof flapping-wing unmanned aerial vehicle with control constraints," in *2018 3rd ICARM*, pp. 548–553, 2018.

[18] E. Sihite and A. Ramezani, "Enforcing nonholonomic constraints in aerobat, a roosting flapping wing model," in *2020 59th IEEE Conference on Decision and Control (CDC)*, pp. 5321–5327, 2020.

[19] C. Qian, R. Chen, P. Shen, Y. Fang, J. Yan, and T. Li, "Trajectory generation and tracking control for flapping wing robot three-dimensional flight," *IEEE/ASME Transactions on Mechatronics*, pp. 1–13, 2024.

[20] A. Ndoye, J. J. Castillo-Zamora, S. Samorah-Laki, R. Miot, E. Van Ruymbeke, and F. Ruffier, "Vector field aided trajectory tracking by a 10-gram flapping-wing micro aerial vehicle," in *ICRA 2023*, pp. 5379–5385, 2023.

[21] W. H. X. M. L. Zhang and Y. Zou, "Modeling and trajectory tracking control for flapping-wing micro aerial vehicles," *IEEE/CAA Journal of Automatica Sinica*, vol. 8, no. JAS-2020-0183, p. 148, 2021.

[22] J. Hoff, U. Syed, A. Ramezani, and S. Hutchinson, "Trajectory planning for a bat-like flapping wing robot," in *IROS 2019*, pp. 6800–6805, 2019.

[23] H. Li, H. Gao, Z. Geng, and Y. Yang, "Predictive control of trajectory tracking for flapping-wing aircraft based on linear active disturbance rejection," *Electronics*, vol. 13, no. 14, 2024.

[24] R. Zhang, J. Hou, G. Chen, Z. Li, J. Chen, and A. Knoll, "Residual policy learning facilitates efficient model-free autonomous racing," *IEEE Robotics and Automation Letters*, vol. 7, pp. 11625–11632, 2022.

[25] Y. Song, A. Romero, M. Müller, V. Koltun, and D. Scaramuzza, "Reaching the limit in autonomous racing: Optimal control versus reinforcement learning," *Science Robotics*, vol. 8, no. 82, p. eadg1462, 2023.

[26] R. Zhang, G. Chen, J. Hou, Z. Li, and A. Knoll, "Pipo: Policy optimization with permutation-invariant constraint for distributed multi-robot navigation," in *2022 IEEE International Conference on Multisensor Fusion and Integration*, pp. 1–7, 2022.

[27] Z. Li, X. B. Peng, P. Abbeel, S. Levine, G. Berseth, and K. Sreenath, "Reinforcement learning for versatile, dynamic, and robust bipedal locomotion control," *RSS 2023*, 2023.

[28] R. Zhang, D. Zhang, and M. W. Mueller, "Proxfly: Robust control for close proximity quadcopter flight via residual reinforcement learning," *arXiv preprint arXiv:2409.13193*, 2024.

[29] W. He, T. Meng, X. He, and C. Sun, "Iterative learning control for a flapping wing micro aerial vehicle under distributed disturbances," *IEEE Transactions on Cybernetics*, vol. 49, no. 4, pp. 1524–1535, 2019.

[30] M. Xiong, Z. Wei, Y. Yang, Q. Chen, and X. Liu, "Lift enhancement of a butterfly-like flapping wing vehicle by reinforcement learning algorithm," *Bioinspiration & Biomimetics*, vol. 18, p. 046010, may 2023.

[31] Y. E. Bayiz, S.-J. Hsu, A. N. Aguiles, Y. Shade-Alexander, and B. Cheng, "Experimental learning of a lift-maximizing central pattern generator for a flapping robotic wing," in *ICRA 2019*, pp. 1997–2003, 2019.

[32] J. Lee, S. Ryu, T. Kim, W. Kim, and H. J. Kim, "Learning-based path tracking control of a flapping-wing micro air vehicle," in *2018 IEEE/RSJ IROS*, pp. 7096–7102, 2018.

[33] Y. Song, L. Weng, and G. Lebby, "Human memory/learning inspired control method for flapping-wing micro air vehicles," *Journal of Bionic Engineering*, vol. 7, no. 2, pp. 127–133, 2010.

[34] F. Fei, Z. Tu, J. Zhang, and X. Deng, "Learning extreme hummingbird maneuvers on flapping wing robots," in *ICRA 2019*, pp. 109–115, 2019.

[35] Z. Tu, F. Fei, and X. Deng, "Bio-inspired rapid escape and tight body flip on an at-scale flapping wing hummingbird robot via reinforcement learning," *IEEE T-RO*, vol. 37, no. 5, pp. 1742–1751, 2021.

[36] R. Vaxenburg, I. Siwanowicz, J. Merel, A. A. Robie, C. Morrow, G. Novati, Z. Stefanidi, G. M. Card, M. B. Reiser, M. M. Botvinick, K. M. Branson, Y. Tassa, and S. C. Turaga, "Whole-body simulation of realistic fruit fly locomotion with deep reinforcement learning," *bioRxiv*, 2024.

[37] S. Hong, S. Kim, and D. You, "Control of a fly-mimicking flyer in complex flow using deep reinforcement learning," 2021.

[38] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," in *2012 IEEE/RSJ IROS*, pp. 5026–5033, IEEE, 2012.

[39] Y. Ji, Z. Li, Y. Sun, X. B. Peng, S. Levine, G. Berseth, and K. Sreenath, "Hierarchical reinforcement learning for precise soccer shooting skills using a quadrupedal robot," *2022 IEEE/RSJ IROS*, pp. 1479–1486, 2022.

[40] Z. Fu, A. Kumar, J. Malik, and D. Pathak, "Minimizing energy consumption leads to the emergence of gaits in legged robots," in *Conference on Robot Learning (CoRL)*, 2021.

[41] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[42] Z. Li, J. Zeng, A. Thirugnanam, and K. Sreenath, "Bridging model-based safety and model-free reinforcement learning through system identification of low dimensional linear models," in *RSS*, RSS, 2022.