Bio-Inspired Rapid Escape and Tight Body Flip on an At-Scale Flapping Wing Hummingbird Robot Via Reinforcement Learning

Zhan Tu, Fan Fei, and Xinyan Deng

Abstract—**Insects and hummingbirds are capable of acrobatic maneuvers such as rapid turns and tight 360° body flips. It is challenging for bio-inspired flapping wing micro aerial vehicles to achieve animal-like performance during such maneuvers due to their limitation in mechanism sophistication and flight control. Besides being significantly underactuated compared to their natural counterparts, flight dynamics is highly nonlinear with rapidly changing, unsteady aerodynamics which remains largely unknown during aggressive maneuvers. As a result, conventional model-based control methods are inadequate to address such maneuvers effectively due to the lack of control references and aerodynamic models. In addition, during acrobatic maneuvers such as body flips, conventional control methods with underlying stabilization mechanisms would temporarily contradict the maneuvering requirements when the vehicle undergoes a full body flip including turning upside-down. In this article, reinforcement learning (RL) has been used to complement model-based control to enable animal-like maneuverability. The learned control policy serves in two different ways to either aid or even completely takeover the conventional stabilization controller in certain cases. We experimentally demonstrate animal-like maneuverability on an at-scale, dual-motor actuated flapping wing hummingbird robot. Two test cases have been performed to demonstrate the effectiveness of such integrated control methods: 1) a rapid escape maneuver recorded from hummingbirds, 2) a tight 360° body flip inspired by houseflies. By leveraging RL, the hummingbird robot demonstrated a shorter completion time in escape maneuvers compared to the traditional control-based method. It also performed 360° body flip successfully within only one wingspan vertical displacement.**

Index Terms—**Aggressive maneuvers, biologically-inspired robots, flapping wing, reinforcement learning (RL).**

Manuscript received November 23, 2020; revised February 15, 2021; accepted February 28, 2021. This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors. This article was recommended for publication by Editor M. Yim upon evaluation of the reviewers' comments. This paper was recommended for publication by Associate Editor Y. Shen and Editor M. Yim upon evaluation of the reviewers' comments. *(Corresponding author: Xinyan Deng.)*

Zhan Tu is with the School of Mechanical Engineering, Purdue University, West Lafayette, IN 47907 USA (e-mail: tu17@purdue.edu).

Fan Fei is with the Amazon.com, Inc, Seattle, WA 98109 USA, and also with the School of Mechanical Engineering, Purdue University, W Lafayette, IN 47907 USA (e-mail: ffnc1020@gmail.com).

Xinyan Deng is with the School of Mechanical Engineering, Purdue University, West Lafayette, IN 47907 USA (e-mail: xdeng@purdue.edu).

This article has supplementary material provided by the authors and color versions of one or more figures available at https://doi.org/10.1109/ TRO.2021.3064882.

Digital Object Identifier 10.1109/TRO.2021.3064882

I. Introduction

THROUGH millions of years of adaptation, insects and hummingbirds have evolved with superior flight capabilities including various aggressive maneuvers [1]–[4]. For example, as shown in Fig. 1, flies can perform nearly drift-free 360° flips within a minimum footprint (translational drift); hummingbirds can perform a rapid escape maneuver with 180° yaw turn in 0.25 seconds when subject to visual threats. These near-maximal acrobatic maneuvers in tight spaces epitomize the maneuverability extremes of animal fliers. Such extraordinary maneuverability poses great challenges for bio-inspired flapping wing micro aerial vehicles (FWMAVs) to even come close to the performance metrics. Meanwhile, benefit from flapping flight, FWMAVs are favorable to hover and maneuver in compact spaces within a small footprint [5]–[8]. In this study, we aim to push the performance boundaries of such robots through aggressive maneuvers shown in Fig. 1.

To date, a number of bio-inspired FWMAVs have demonstrated stable hovering flight [9]–[17]. Few attempts were aimed to challenge animal-like aggressive maneuvers [11], [15], [16]. In [11], an adaptive controller combined with an iterative learning method is designed for an 80 mg, 3 cm wingspan FWMAV to perform vertical landing. In [15], the authors propose to use control torque coupling to achieve fruitfly-inspired backed turns on a 28.2 g, 33 cm wingspan FWMAV. Work [16] is a preliminary study for this article.

To attempt maneuvers shown in Fig. 1, it usually requires more effort on flight control design other than stabilizing flight [11], [16]. Moreover, the robots may still require considerably more time and longer travel distance [9], [15], [16] compared to their natural counterparts, if not losing stability totally. The maneuverability gap is due to several challenges facing FWMAVs in flight control:

1) Dynamics uncertainty: Flapping flight is highly nonlinear, coupled dynamics which is further complicated by its unique inherent dynamic mechanisms and unsteady aerodynamics [5]–[7], [18]. As a result, flight dynamics varies significantly under different flight modes and many remain unknown besides the hovering scenario.
2) Actuation limitation: Compared to flying creatures' highly intricate and powerful wing-thorax actuation systems, the number of actuators and their power density of the robots is very limited [19], [20], resulting in severely inadequate control effort in agile maneuvering.

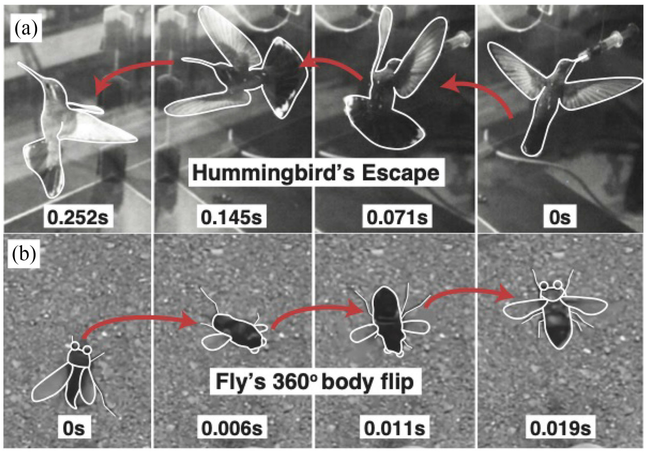


Fig. 1. (a) Sequences of a hummingbird's escape maneuver. (b) Sequences of a housefly's backflip. The details are shown in the attached video.

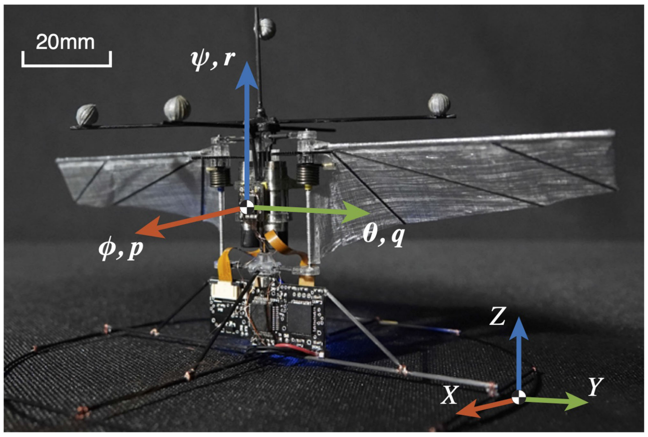


Fig. 2. Illustration of the test robot. Platform details are presented in [21].

3) Lack of control references in maneuvering: Conventional flight control relies on explicit references to pilot the vehicle. However, in aggressive maneuvering, it is infeasible to accurately establish certain trajectories to track [11], [15], [16]. In fact, flying animals also rarely perform the same trajectory each time during rapid escapes or body flips. Most importantly, during certain maneuvers such as body flips, conventional flight control, which is pursuing stability, becomes ineffective as its control effort contradicted vehicle maneuvering goal when it turned upside-down.

To address the above control challenges, in this work, we propose to incorporate reinforcement learning (RL) into conventional model-based flight control. We focus on different integration approaches of RL. In particular, two integration methods are proposed, i.e., the learned control policy is used to either aid or fully takeover the conventional model-based stabilization controller to address different aggressive maneuvers. The test platform is a dual-motor actuated hummingbird robot shown in Fig. 2. The details of the test platform can be found in [21]. On the test robot, the optimal control policy can be learned and in place of the traditional reference tracking methods in maneuvering because the process from low-level actuation commands to the robot states can be modeled as a Markov decision process. To demonstrate the effectiveness of such an integrated control effort, we implemented it on the proposed robot to challenge the aggressive maneuvers shown in Fig. 1: rapid escape and tight 360° body flip. Both simulation and experiments show that the proposed hybrid strategy not only can work with instantaneous uncontrollable scenarios effectively, but also demonstrate flight performance that resembles their natural counterparts. By leveraging RL, the hummingbird robots, which actually lack the elaborate actuation systems seen in flying animals, demonstrate comparable flight performance with natural fliers in escape maneuvers and can also successfully perform 360° body flip within only one wingspan vertical travel. This is the first time that such maneuvers are achieved on a flapping wing robot with only two actuators under such tight spatiotemporal constraints.

We note here that part of the preliminary result was initially presented in a conference version [16]. As the first step of integrating RL in flight control, this preliminary study proves that RL can be fused with a traditional controller for flight control. The effectiveness of such an initial design highly depends on whether the traditional controller operates properly. Meanwhile, the reward function design of the initial setup relies on confirmed references, similar to tracking control. Therefore, such initial results may not able to explicitly demonstrate the advantages of RL in maneuver control. Improved upon it, this article investigated aggressive maneuvering scenarios in which the traditional controller became completely ineffective, e.g., 360° flipping. In this case, a new RL integration method is proposed to fully takeover the flight control. A new design method of the reward function is presented as well, which avoids using biological data or a particular trajectory for control reference. For escape maneuver, a performance comparison between RL-based and traditional control-based methods have been conducted as a systematic extension to our preliminary study [16].

The rest of the article is organized as follows. Section II summarizes the robot design and the baseline flight control for stabilization. Section III proposes two different ways for RL integration, aiming to challenge different animal-like acrobatic maneuvers on the proposed hummingbird robot. For each maneuver, a particular reward function is designed to incentivize the robot's behavior in training. Section IV presents the experimental validation and the evaluation of the training results. Finally, Section V concludes this article.

II. Robot Design and Stabilization Control

A. *Robot Design*

In this section, we introduce the test platform in this study-a hummingbird-inspired flapping wing robot shown in Fig. 2. Such a robot equips two independent controlled wings to produce thrust and control torques. Accordingly, two dc motors are implemented to modulate wing kinematics for full 6-Degree of Freedom control [21]. Key parameters of the robot and its natural counterpart, i.e., a magnificent hummingbird [3], are listed in Table I.

TABLE I
PARAMETER COMPARISON OF THE ROBOT AND A REAL HUMMINGBIRD

Parameters	Robot	Hummingbird
Wing length (R_w)	70mm	77mm
Wingbeat frequency (f)	34Hz	32Hz
Wing aspect-ratio ($\mathcal{R}$)	6.6	7.9
2nd moment of wing area ($\hat{r}_2$)	0.53	0.49
Total weight (m)	12g	7.1g
x-axis moments of inertia (J_{xx})	4238.13gmm^2	2962gmm^2
y-axis moments of inertia (J_{yy})	3970.16gmm^2	2119gmm^2
z-axis moments of inertia (J_{zz})	2440.95gmm^2	2741gmm^2

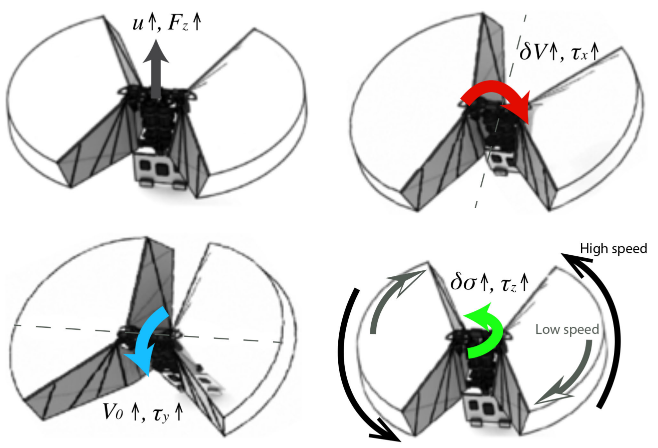

$u\uparrow, F_z\uparrow$
$\delta V\uparrow, \tau_x\uparrow$
High speed
$\delta\sigma\uparrow, \tau_z\uparrow$
Low speed
$V_0\uparrow, \tau_y\uparrow$

Fig. 3. Illustration of the control inputs and control force/torque generation. The control force/torque rises with the increase of their respective control input.

As shown in Table I, the proposed robot is designed to morphologically close to the real magnificent hummingbird except for the moments of inertia about x and y axes. Such differences affect the robot's maneuverability, resulting in its unique behavior compared to its natural counterparts in the same flight tasks, which is detailed in Section IV.

B. Stabilization Control

Stable flight is the basis for acrobatic maneuvers of FWMAVs. In this section, we summarize the stabilization control of the robot.

First, the control inputs should be defined. As introduced in [21], the pilot command of the robot is given by a sinusoidal input voltage for each motor. Then, the forced response of their respective wings performs a near sinusoidal trajectory accordingly. As shown in Fig. 3, by leveraging the decoupled wing kinematics design, we define four parameters as control inputs:

1) u-nominal input voltage of the two motors, which determines the overall thrust F_z generation for altitude control;
2) δV-differential input voltage between two motors, which generates roll torque τ_x for roll angle ϕ control;
3) V_0-nominal voltage bias of the two motors, which generates pitch torque τ_y for pitch angle θ control;
4) $\delta\sigma$-split-cycle parameter, which controls the half-stroke differential drag forces of the wings for yaw torque τ_z generation and yaw angle ψ control.

In addition, for controller design, we model the vehicle as a rigid body

$$\begin{aligned} \dot{\mathbf{P}} &= \mathbf{v}, & m\ddot{\mathbf{P}} &= \mathbf{R}\mathbf{F}^b + m\mathbf{g}, \\ \dot{\mathbf{R}} &= \mathbf{R}\hat{\omega}, & \mathbf{J}\dot{\boldsymbol{\omega}}^b &= \boldsymbol{\tau}^b - \boldsymbol{\omega}^b \times \mathbf{J}\boldsymbol{\omega}^b \end{aligned} \tag{1}$$

where $\mathbf{P} = [x, y, z]^T$ and $\mathbf{v} = [\dot{x}, \dot{y}, \dot{z}]^T$ are the position and velocity vectors of the vehicle in the inertial frame; m is the total mass; $\mathbf{g}$ is the gravity acceleration vector; $\mathbf{R} = [\hat{i}, \hat{j}, \hat{k}]$ is the rotation matrix; $\hat{\bullet}$ denotes the skew-symmetric matrix mapping from $\hat{\mathbf{a}}\mathbf{b}$ to $\mathbf{a} \times \mathbf{b}$; $\bullet^b$ represents the body frame vector, including the 3-axis control force $\mathbf{F}^b$, the vehicle angular velocity $\boldsymbol{\omega}^b = [p, q, r]$, and the 3-axis control torque $\boldsymbol{\tau}^b$; $\mathbf{J}$ is the inertia matrix of the vehicle. $\mathbf{F}^b$ and $\boldsymbol{\tau}^b$ are generated by the motor excitation modulations, which is approximated with linear fitting in implementation [21].

Flight control of such vehicles is already a challenging task due to the severe underactuated body dynamics (using two motors for 6-axis control), modeling uncertainty, and time-varying aerodynamics. In this article, we use the same controller as proposed in the preliminary study [16] to stabilize the vehicle since its effectiveness is already demonstrated. It has been designed with parameter adaptation and robust control terms to deal with the control challenges mentioned above. A detailed formulation of the controller can be found in [16].

III. RL-ENABLED AEROBATIC MANEUVERS

For stable hovering, the averaged dynamics model of FWMAVs is relatively accurate for model-based control. By comparison, for aggressive maneuvers, the approximation error becomes unmanageable [11], [16]. In particular scenarios, such as flip maneuvers, conventional model-based control even lose its effectiveness as mentioned above. To enable aggressive maneuvers of FWMAVs, we propose to integrate the stabilization control and RL trained maneuver policy. Through the training, the learned maneuver policy can adapt to and generate appropriate control action in a wide flight envelope.

In this section, we first introduce the RL basics and different RL integration methods for particular aggressive maneuvers. Then, we present two kinds of reward function design that incentivize their respective maneuvers. Finally, we introduce the training and implementation process.

A. Problem Setup

RL is a specific area of machine learning. The key mechanism in RL is performing optimal actions to maximize a certain reward. In a typical RL problem setup, there is an agent that keeps interacting with an environment. This interaction can be formulated as a Markov decision process, which consists of environment state space $\mathcal{S}$, action space $\mathcal{A}$, state transition model $\mathrm{p}(s_{t+1}|s_t, a_t)$, reward model $\mathrm{r}(s_t, a_t)$, and discount factor γ, wherein t represents time step. An agent policy function $\pi : \mathcal{S} \to \mathcal{A}$ is accordingly defined to take the current environment state and return an action. π is a policy network $\pi_\nu(a_t|s_t)$ which is parameterized by ν.

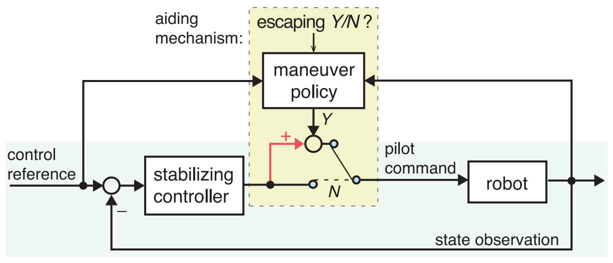


Fig. 4. Block diagram of the escape maneuver control.

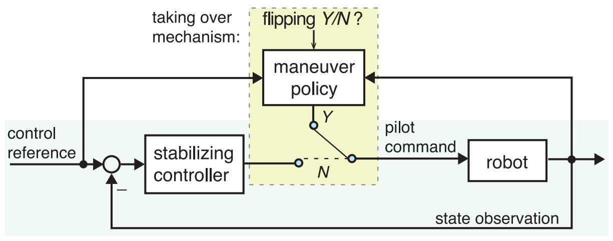


Fig. 5. Block diagram of the flip maneuver control.

The agent starts from an initial state distribution $\mathrm{p}(s_1)$. At each time step, the environment state changes according to the agent's action. Meanwhile, agent action results in a reward/penalty response from the environment. The goal of the learning process is to obtain the optimal policy parameter ν, i.e., how actions affect the state, such that it can maximize the cumulative rewards. The expected discounted return of a finite-horizon RL problem is

$$J = \mathbb{E}_{a_i \sim \pi_\nu}[\mathcal{R}] = \mathbb{E}_{a_i \sim \pi_\nu} \sum_{i=0}^{T} \gamma^{i-1} \mathrm{r}(s_i, a_i) \tag{2}$$

where T is the horizon, i is the discrete step and γ is the discount factor. In this study, each component is defined by

1) Agent: The proposed hummingbird robot;
2) Environment: A constraint flight test environment;
3) States: $s = [\mathbf{P}, \mathbf{v}, \mathbf{R}, \boldsymbol{\omega}^b] \in \mathcal{S}$;
4) Action: The additional control commands, wherein action $a = [\Delta u, \Delta\delta V, \Delta V_0, \Delta\delta\sigma] \in \mathcal{A}$;
5) State transition dynamics: The closed-loop vehicle dynamics;
6) Reward function: A customized function that incentivize the robot to perform a certain maneuver.

Based on the above problem setup, we propose two targeted architectures that integrate RL in the conventional control system, aiming to improve the maneuverability of the robot in different flight tasks.

1) In those maneuvers that the conventional stabilization controller can handle, the RL-based method can aid the original stabilization control law to further improve robot's maneuverability. In this article, we use a hummingbird-inspired rapid escape maneuver as an example. To address this case, the integration approach of the RL-based method and the original control system is summarized in Fig. 4. During the escape maneuvering, the control command is integrated by the controller and the trained policy. When the robot finished maneuver, e.g., reached a safe distance with proper orientation, the stabilizing controller will takeover the flight control.

2) In some other maneuvers that the conventional stabilization controller **cannot** handle, RL-based control method can be used to fully takeover the flight control during maneuvering. In this article, we use a 360° body flip maneuver for demonstration. To tackle this maneuver, another integration approach of the RL-based method and the control system is illustrated in Fig. 5 originally. During the body flipping, the trained policy pilots the robot. As a 360° turn finished, the control command switches back to the stabilizing controller for hovering. Regarding the switch logic, RL would be able to figure it out with long-term training. Nevertheless, given a determined switch condition can obviously save training time and shrink the searching range.

B. Reward Function

Reward function design is one of the most important steps in RL application. A well-designed reward function could stipulate the agent's specific task and incentivize the agent to follow an optimal way to accomplish the task. This section presents two different reward functions that address two representative maneuvers, respectively, rapid escape and 360° body flip. Such designs also correspond to two different integration ways of the RL-based control mentioned above.

1) Rapid Escape: The detailed hummingbird's escape maneuver illustrated in Fig. 1 has been studied in [3]: The tested hummingbird performed a rapid retreat about three wing length (≈ -0.21 m) with a 180° heading turn. It can be completed within 0.3 s (about 9–10 wingbeats). Based on the above information, the control reference is clear. We choose architecture.1) introduced in the last section to incorporate RL policy, treating this maneuver exactly like a tracking control problem. Based on the known control reference, we can define the switching logic of the control method. Particularly, when x<-$0.21m$, $|\dot{x}|$<$1m/s$, and yaw angle ψ>$180°$, the control priority switches from hybrid control to the conventional stabilization controller.

With such control reference, the reward for the escape maneuver is designed by

$$\mathrm{r}_t = (f_t + \varepsilon_f)^{-1} - \hat{i} \cdot \hat{I} + (\hat{k} \cdot \hat{K})^{-} + \lambda_x (x_T - x)^{-} \tag{3}$$

where f_t is a cost function which is composed by stability cost and control cost presented in [16], ε_f is a small number for numerical stability, $\hat{I} = [1, 0, 0]^T$ and $\hat{K} = [0, 0, 1]^T$ are unit vectors in the inertial frame. λ_x is a positive scaling factor. $\bullet^{-}$ means that the function only takes negative value. In training, the first term will minimize control cost. The second and third terms will penalize the incorrect pose of the robot. The last term will ensure the vehicle converges to the target x_T. These terms incentivize the agent to learn a policy that enables it to perform a hummingbird-like escape as fast as it can to avoid penalization and the cumulative negative rewards.

2) Tight 360° Body Flip: To perform the flip maneuver, the robot initially hovers at a certain position $\mathbf{P}_0 = [0, 0, z_0]^{\mathrm{T}}$. When a flip motion is triggered, the robot initiates a 360° body turn

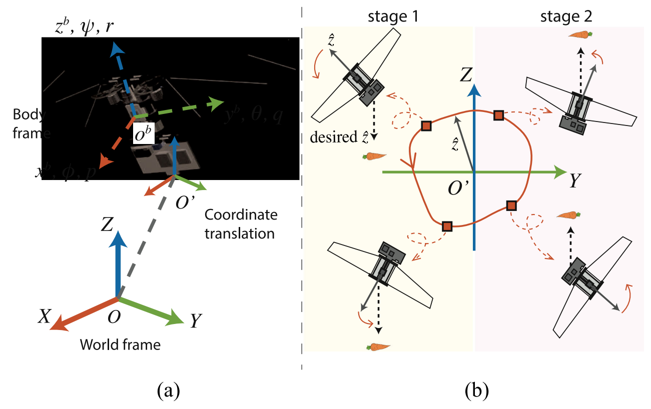


Fig. 6. (a) Coordinate definition. (b) Illustration of "carrot-and-stick" method for the reward function design. In (b), a typical successful "carrot guided" 360° side flip and the corresponding projected trajectory is shown.

and then stabilizes at $\mathbf{P}_0$ again. As shown in Fig. 6, a successful 360° rollover can be described by the trajectory of body z-axis projection crossing all four quadrants of coordinate $ZO'Y$ (side flip) or $ZO'X$ (back flip). The coordinate system attached to the tail of the vehicle is translated from the world frame. Fig. 6(b) illustrates a successful side flip maneuver.

Unlike the escape maneuver, flip maneuver does not have a determined control reference. For optimization purposes, it is pointless to set a reward function to enforce the robot to follow an exact maneuver pattern in this task, e.g., tracking a preset trajectory or performing a determined back or side flip. Different from the "tracking" idea in traditional control, a straightforward solution of reward function design is presented here, which is aiming to tap the potential of the robot maneuverability and exploring the platform-unique optimal behavior.

A complete flip maneuver can be intuitively split into two steps: first, tilting upside down, then, turning back to the head-up state. Therefore, we propose a "carrot-and-stick" design to describe the whole process. As shown in Fig. 6(b), the first "carrot" is set to the bottom of the robot; after the robot flips 180° vertically and successfully reached the first "carrot," the second "carrot" appears at the robot's original head area to lead the robot to turn back. In order to get the reward in the second stage, at the end of the first stage, the vehicle will learn to maintain a certain rotational speed to avoid mission failure.

From Fig. 6(b), at the beginning of flipping, a well-designed reward function should incentivize the robot to perform an upside-down motion. Mathematically, the reward should keep increasing when the body z-axis projection in the inertia frame $\hat{z} = \hat{k} \cdot \hat{K}$ is approaching -1 (representing the robot heading downwards). So the reward model on this stage is designed as $\mathrm{r}_1 = K_1/[1 + (1 + \hat{z})^2]$, where K_1 is a positive gain. Once the robot finished the upside-down step, the reward function will move on to award the $\hat{z}$ approaching 1 to recover the attitude of the robot. Thus, the second term of our reward model is defined oppositely to r_1, which is $\bar{\mathrm{r}}_2 = K_2/[1 + (1 - \hat{z})^2]$. Like r_1, K_2 is a positive gain as well. Note, in our particular setup, when $\hat{z} > 0.8$, the angular speed $p \in [-2.79, 2.79]$rad/s, and $q \in [-2.79, 2.79]$rad/s, the control command switches back to the stabilization controller. Such a "carrot-and-stick" configuration successfully guides a basic flip maneuver.

Besides the "carrot-and-stick" mechanism, the above setup may generate some undesirable behavior. For example, the robot will learn to stay in stage 1 and keep oscillating to collect cumulative rewards. To filter out these undesirable behaviors, a linear growing stability reward r_s and a position tracking reward $\mathrm{r}_p = \lambda_{\mathbf{P}}||\mathbf{e_P}|| + \lambda_{\mathbf{v}}||\mathbf{v}||$ have also been added to r_2 to further promote a flip-drift-recovery process, where $\lambda_{\mathbf{P}}$ and $\lambda_{\mathbf{v}}$ are positive gains. On the other hand, a constant reward term r_s is given in the recovery stage which leads the robot to perform flip as quickly as possible for gaining more cumulative reward. Meanwhile, these additional terms prevent large positional drift, assisting the model-based control when the robot turns back to normal flight. The lumped $\mathrm{r}_2 = \bar{\mathrm{r}}_2 + \mathrm{r}_s + \mathrm{r}_p$. The total reward is given by

$$\mathrm{r}_t = (1 - \lambda)\mathrm{r}_1 + \lambda \mathrm{r}_2 \tag{4}$$

where λ is a binary gain to indicate whether the robot is in the first or second stage in the flip maneuver.

In order to achieve animal-like performance where the available space for flipping is tuned to be tightly constrained. Eventually, we only allow one wingspan length of translational drift in all X, Y, Z directions. During the training, if the robot travels out of bounds, the current rollout will be terminated to cut the cumulative reward, thus penalizing the corresponding action. By maximizing the reward, the policy will learn to minimize the vehicle's position drift.

C. Training

The RL policy training is conducted in a high fidelity simulation tool that is developed with full system dynamics including flapping flight aerodynamics, wing-actuation thorax dynamics, and vehicle body dynamics. In the simulator, we use a quasi-steady model to approximate the aerodynamic force. The effectiveness of such a method has already been validated and widely used by many flapping-wing platforms [6], [21], [22]. For the proposed robot, the modeled result has been fitted linearly. The accuracy of such results has been experimentally validated [21], [23]. In order to transfer to the real robot platform, the sensing error, communication delay, and arbitrary noise are also introduced in the simulator to emulate the real flight condition. All physical parameters and signal characteristics in the simulation come from the system identification of the robot. Details of the simulator setup are presented in [23].

In our design, both the state transition model and reward function are deterministic functions. Therefore, a deep deterministic policy gradient (DDPG) method is chosen as the training algorithm due to its robust performance on such robot platforms with continuous action spaces [24]. DDPG is an off-policy algorithm constructed with actor-critic architecture that updates a policy function (actor) and an action-value function (critic) by batch sampling from a replay pool of tuples $(s_t, a_t, \mathrm{r}_t, s_{t+1})$ [24]. The function approximators are fully connected multilayer perceptrons. The actor and critic networks are configured with 32×32 and 64×64 hidden units, respectively. Hidden and

TABLE II
RANDOMNESS IN TRAINING

Noise/Uncertainty	Distribution	Randomness bound
total mass	Uniform	±0.3g
3-axis body inertia	Uniform	± 50g.mm^2
initial position	Uniform	±0.1m
initial velocity	Uniform	±0.1m/s
initial attitude	Uniform	±11.4°
initial angular velocity	Uniform	±5.7°/s
actuation noise	Gaussian	±0.2V
position feedback	Gaussian	±0.05m
velocity feedback	Gaussian	±0.05m/s
attitude feedback	Gaussian	±2°
angular velocity feedback	Gaussian	±5.7°/s

output activation functions in both networks are hyperbolic tangent functions. Such a small-scale network design reduces computational load substantially. In particular, it costs 58 additions and 1728 multiplications in a typical control cycle. For the proposed robot, which equips an STM32 onboard processor with a 32-bit ARM-based Cortex-M4 core, it costs about $<$1 ms to complete the calculation under the working condition of single-precision, 72 MHz computation frequency, and Floating Point Unit (FPU) on. Note, in this case, FPU is necessary. As FPU turns off, the processing time would $\geq$5.5 ms that could not match the control loop frequency. In this work, we have not tested different network structures. The proposed structure is versatile for the test maneuvers.

During the training, vehicle dynamics is updating at 10KHz. Considering the consistency between real and simulated flight control of the robot, the simulated control frequency is downsampled to 500 Hz which is the same as the onboard control. The horizon of each rollout has 1000 maximum samples which correspond to 2 seconds. Each epoch is set to a maximum of 10 000 samples, i.e., $\leq$ 20 seconds or 10 rollouts at least. All training sessions presented in this article were done on a personal computer with an i7-8700 CPU. The operating system is Ubuntu 16.04. The training algorithm is deployed by using $rllab$ benchmark [25]. An escape maneuver training typically lasted about 75-90 hours. A body flip maneuver training typically lasted about 65-80 hours. During the implementation, the maneuvering policy is triggered when the vehicle is within a specified state space which covers p(s_1). When the maneuver is finished and the robot reaches the state space near the defined target p(s_T), the maneuver policy will turn off.

In order to acquire a sim-to-real portable solution, we use the dynamics randomization approach in the training process [26]. Randomness was injected into the physical parameters of the simulated vehicle, such as mass, inertia, actuation noise, initial conditions, and sensing noise to imitate the errors that highly possible occurs in the system. The particular noise distribution is listed in Table II. Furthermore, the termination condition is set to stop an episode early to improve the training efficiency and prevent the useless action state from appearing in the replay pool. By setting termination conditions, we intend to penalize the meaningless behavior with a huge position drift or fully destabilized flight. Constantly compressed termination space can also incentivize robots to maneuver in tight spaces, such as drift-free 360° somersaults. Successful maneuvers wherein the robot resumes a hovering flight will keep collecting positive rewards and maximizing the total return.

IV. Experimental Result and Discussion

In this section, we conducted flight tests to evaluate the performance of the proposed RL-based motion control on both simulated and real hummingbird robots. Both two aggressive maneuvers as discussed in Section III are tested. In real-world experiments, a Vicon motion capturing system is used to provide indoor position feedback. Attitude feedback is from an onboard inertial sensor with onboard sensor fusion. The sensor fusion method is similar as [27] with delay calibration and a different Kalman filter design. By leveraging the accuracy of the periodical Vicon feedback, the sensing drift can be corrected. The stabilization control and trained policy are tested in the simulation first. Both of them are then port to the onboard processor for real world tests. Same as the simulation test, onboard control frequency is 500 Hz and the lumped pilot command is limited by a predefined saturation range, i.e., $u \in [9\ V, 18V]$, $\delta V \in [-3\ V, 3V]$, $V_0 \in [-3.5\ V, 3.5V]$, and $\delta\sigma \in [-0.15, 0.15]$. The power consumption is obtained by the multiplication of the motor input voltage and the current.

A. Hummingbird-Like Rapid Escape

The key parameters of the studied magnificent hummingbird [3] are shown in Table I. Compared to the proposed robot, it shows similar wing morphology and flapping frequency f in escape maneuver despite differences in total mass and body inertia. With RL aiding, the hybrid control policy manifests a behavior that shares some similarities to that observed on hummingbirds as detailed in our previous work [16]. With limited wing actuation and degrees of freedoms, and larger body mass and inertia, the FWMAV completes the escape maneuver within 20 wingbeats, compared to the magnificent hummingbird's $\approx$10 wingbeats performance. Such a result shows that an engineered flapping wing vehicle can produce animal-like extreme maneuvers even with limited actuation.

As shown in [16], the overall motions for both magnificent hummingbird and FWMAV follow the same pattern, i.e., backward and pointing body horizontal toward the direction of escape, yaw turn, and pointing body upwards again. A similar movement is also previously observed on hawkmoth [2]. However given their different abilities to generate yaw torque, the magnificent hummingbird was able to complete 180° yaw turn in just 4 wingbeats, whereas the robot is generating yaw torque during the entire maneuver. The hummingbird can use the tail as a damper to generate additional control effort in x direction to decelerate, whereas the robot cannot. Such differences in the actuation of these two systems are also reflected in the power consumption. As shown in Fig. 8(b), for the proposed robot, the average power in the entire maneuver stage and hover stage are 10.14 and 5.32 w, respectively. The stroke-averaged burst power surges to almost 19.38 w, about 3.64 times that of the hovering flight. This ratio is not as aggressive as the magnificent

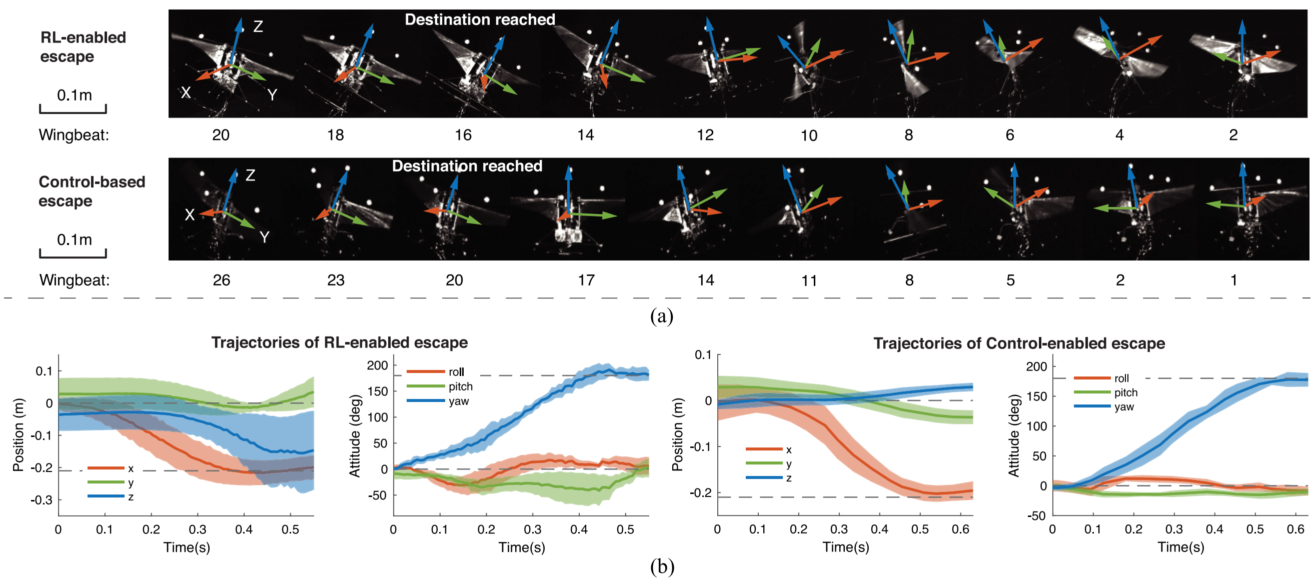


Fig. 7. (a) Sample of escape maneuver sequences achieved by the RL-enabled method and traditional control-based method. (b) Flight trajectories. Trajectories are averaged over four trials for each case. The grey dashed lines denote control references. The shaded area indicates their respective standard deviation.

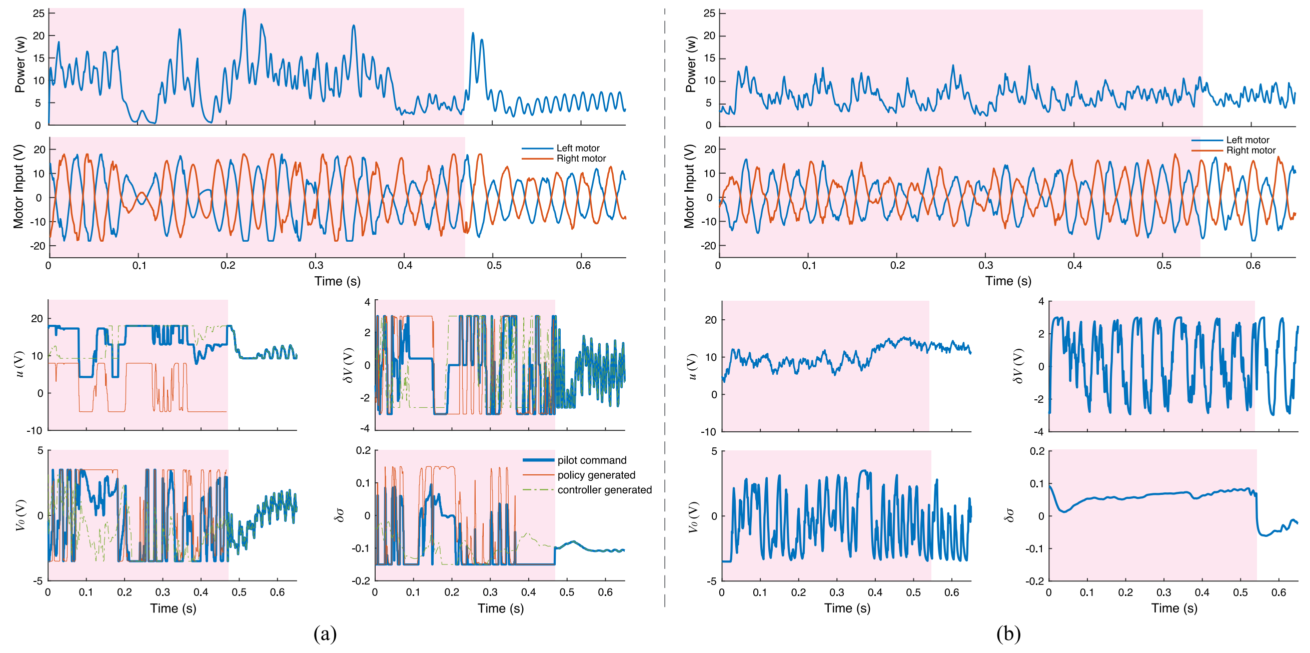


Fig. 8. (a) Energy consumption, motor input, and pilot command of a typical RL-based escape maneuver. (b). Energy consumption, motor input, and pilot command of a typical control-based escape maneuver. The shaded area represents the maneuvering period. For (a), during maneuvering, the pilot command is a hybrid input - the integration of the learned policy and the stability controller. As the maneuver is finished, the flight control is handled by the stabilization controller solely. The pilot command is bounded by a predefined saturation range.

hummingbird whose peak power was almost 5 times of the hovering flight as introduced in [3].

In order to demonstrate the advantages of RL, we compared the RL-assisted control scheme with traditional control methods in the same task. The stabilization controller introduced in Section II-B is the candidate for comparison test, which has already demonstrated accurate trajectory tracking performance [16]. We set the exact same $\mathbf{P}_T$, and ψ_T as the control references for control-based escape maneuver. As illustrated in Fig. 7, the escape actions achieved by the traditional control-based method show less attitude and height drift than those of RL-enabled flights. On the other hand, the RL-assisted maneuver reduces the completion time of 3 wingbeats on average (about 0.088 s) compared to using the conventional control. For such a maneuver that only requires about 0.5 s, the proposed method saves about 18% time cost. As shown in Fig. 8, the two methods

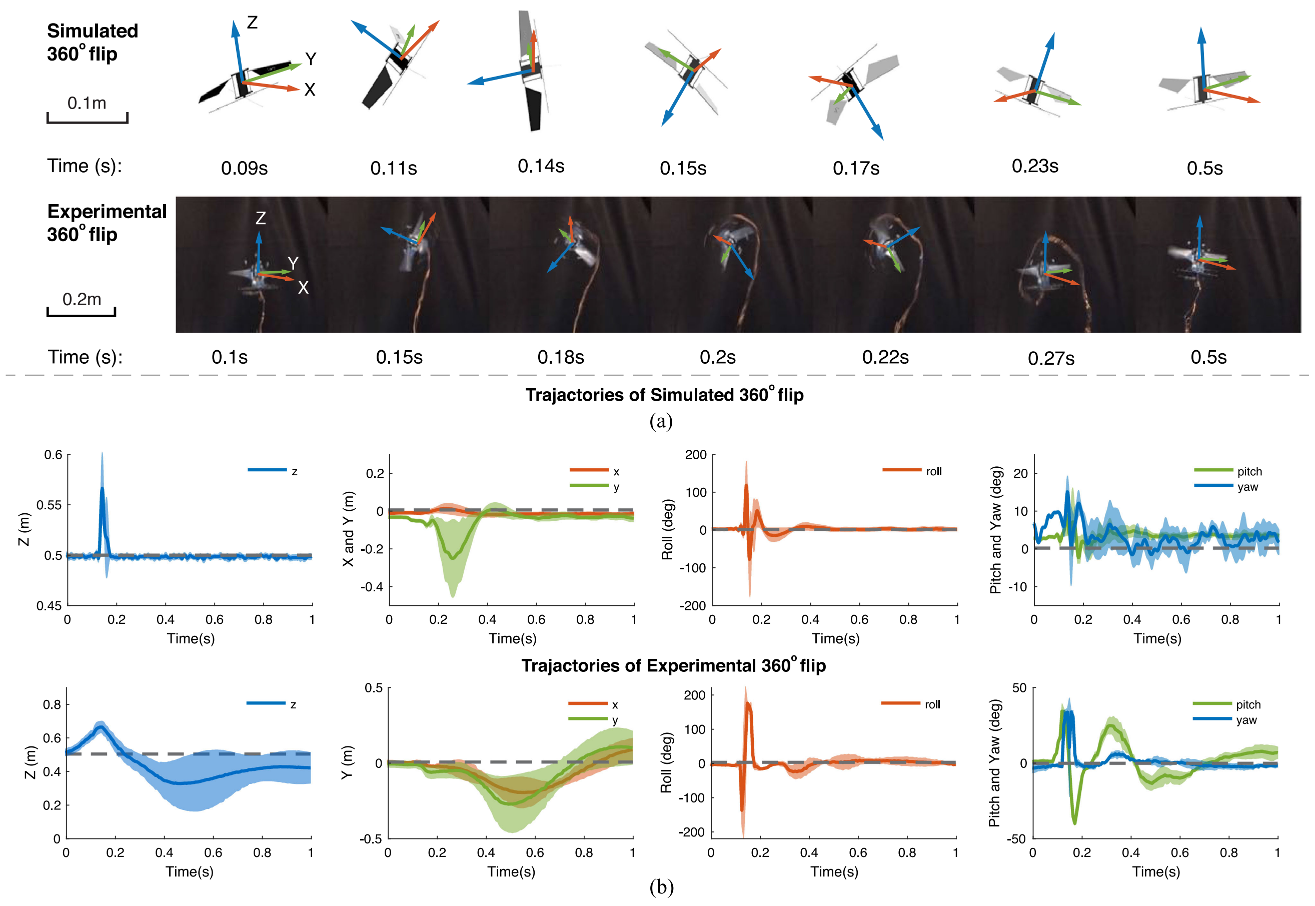


Fig. 9. (a) Flip maneuver sequence in both simulation and experiment. (b). Flight trajectories of the robot in both simulated and experimental flip maneuver. Trajectories are averaged over five trials. The dashed lines denote control references. The shaded area indicates their respective standard deviation.

perform two different control patterns. During the maneuvering, the learned policy performs more aggressive attitude control in order to boost the escaping speed, e.g., the learned policy will keep pushing pitch and yaw torque to the upper bound. This is obviously against the design of the stabilization controller. Thus, it has, to some extent, explored the extreme maneuverability of the proposed robot. On the other hand, though the RL-based method reaches the safe distance faster, its performance fluctuates. By comparison, the control-based method shows overall consistent performance. Such consistency is also reflected in power consumption as shown in Fig. 8(b).

B. *Tight 360° Body Flip*

Besides such extreme maneuvers that traditional control-based methods can handle but with performance drop, in this section, we use the proposed RL-based method to challenge the scenario that such traditional methods become ineffective. Particularly, we challenge animal-like tight 360° body flip on the hummingbird robot. In such a scenario, the stabilization controller introduced in Section II-B cannot work properly as normal because the control effort contradicts the control target when the robot flipped upside-down.

After training, the learned 360° body flip maneuver of the robot converges to the side flip motion as shown in Fig. 9, which is a bit out of our expectation since the front/back flip is more observed on the animals. Theoretically, the flip along the longitudinal direction (dorsoventral) is easier for flapping wing platform due to the FCF and FCT effects. Perhaps due to the larger torque generation along the roll axis, the manifested flip maneuver trends to body lateral direction [23].

As shown in Fig. 9, to finish such a tight flip, the robot learned to first generate an upward acceleration. After gaining some momentum, the left wing's amplitude increases to enable the roll turn. When the robot is upside down, the right wing's amplitude increases to decelerate body rotation. To reduce the altitude drop, the maneuver policy is optimized to sacrifice a few X-Y tracking performance. Eventually, the robot will return to the upright position with small overshoots, which results in a small lateral position drift. Once return to the normal flight, such a small lateral tracking error can be corrected by the model-based flight controller quickly. Due to certain trim conditions, the robot always tends to rotate counterclockwise.

We have conducted a total of 8 experimental flight trials, 6 of which completed the task, 2 of them performed upside-down flip but lost stability during the recovery step. The optimized policy can complete the flip maneuver in about 0.15 s, and within

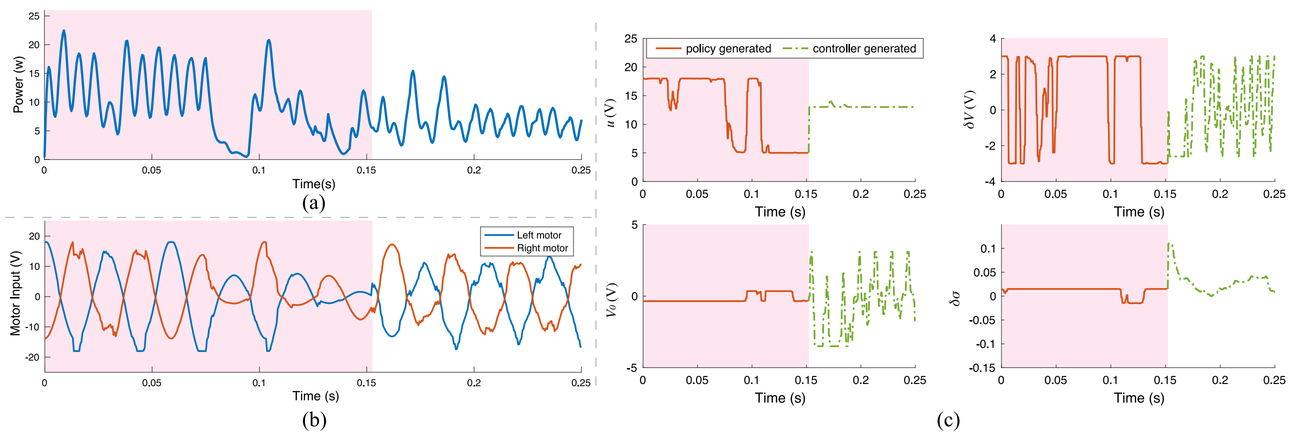


Fig. 10. (a) An energy consumption sample of a typical 360° side flip maneuver. (b). The motor voltage input corresponds to (a). (c). The pilot command corresponds to (a), which is composed of the learned policy and stabilization controller. The shaded area represents the maneuvering period, during which time the learned policy completely takeover the flight control. As the maneuver finished, just the stabilization controller handles the flight control. The pilot command is bounded by a predefined saturation range.

vertical travel of approximately one wingspan ($\approx$0.17 m). It completes within such tight spatiotemporal constraints, which is similar to the behaviors observed on real animals. A sample of energy consumption and the corresponding pilot command is shown in Fig. 10. The average power in the entire maneuver stage and hover stage is 9.278 and 6.179 w, respectively. As shown in Fig. 10(c), during the body flip, the pilot command is completely from the learned policy. Once the body flip complete, the pilot command switches to stabilization controller generated inputs. Note, compared to the real system, the simulated robot showed better ascending capability and almost drift-free altitude performance due to the discrepancy between the modeled and actual z-direction damping. Such a discrepancy affects altitude control performance.

V. Conclusion

In this article, we presented the successful attempts of using reinforcement learning-enabled coupled with conventional flight control strategies for bio-inspired flapping wing robots to achieve animal-like acrobatic maneuvers. During aggressive maneuvers, the model-free reinforcement learning policy is trained in simulation to assist the robot to achieve the desired performance in maneuvering. We experimentally demonstrated that such an AI-based control strategy can achieve animal-like maneuverability on an at-scale hummingbird robot equipped with just two actuators. Two acrobatic maneuvers have been demonstrated to show the performance of the proposed control strategy: 1) A rapid escape maneuver that was observed in real hummingbirds, 2) a nearly drift-free 360° body flip. The successes in both of these extreme maneuvering cases show the promise of using reinforcement learning in bio-inspired robots to cope with model uncertainty, unmodeled dynamics, and demanding flight performance with severely underacted systems. We showed that although such robots cannot replicate the elaborate actuation in animals, through a combined method of reinforcement learning and conventional controls, they can successfully achieve comparably flight performance as their natural counterparts.

References

[1] S. Dalton and J. Kings, *Borne on the Wind: The Extraordinary World of Insects in Flight*. London, U. K.: Chatto and Windus, Jan. 1975.
[2] B. Cheng, X. Deng, and T. L. Hedrick, “The mechanics and control of pitching manoeuvres in a freely flying hawkmoth (manduca sexta),” *J. Exp. Biol.*, vol. 214, no. 24, pp. 4092–4106, 2011.
[3] B. Cheng *et al.*, “Flight mechanics and control of escape manoeuvres in hummingbirds ii aerodynamic force production, flight control and performance limitations,” *J. Exp. Biol.*, vol. 219, no. 22, pp. 3532–3543, 2016.
[4] P. Liu, S. P. Sane, J.-M. Mongeau, J. Zhao, and B. Cheng, “Flies land upside down on a ceiling using rapid visually mediated rotational maneuvers,” *Sci. Adv.*, vol. 5, no. 10, 2019, Art. no. eaax 1877.
[5] C. Ellington, “The aerodynamics of insect flight ii morphological parameters,” *Phil. Trans. Roy. Soc. Lond. B*, vol. 305, pp. 17–40, 1984.
[6] M. H. Dickinson, F.-O. Lehmann, and S. P. Sane, “Wing rotation and the aerodynamic basis of insect flight,” *Sci.*, vol. 284, no. 5422, pp. 1954–1960, 1999.
[7] S. P. Sane, “The aerodynamics of insect flight,” *J. Exp. Biol.*, vol. 206, no. 23, pp. 4191–4208, 2003.
[8] W. Shyy, Y. Lian, J. Tang, D. Viieru, and H. Liu, *Aerodynamics of Low Reynolds Number Flyers*. Cambridge, U.K.: Cambridge Univ. Press, 2007, vol. 22.
[9] M. Keennon, K. Klingebiel, and H. Won, “Development of the nano hummingbird: A tailless flapping wing micro air vehicle,” in *Proc. 50th AIAA Aerospace Sci. Meeting Including New Horizons Forum Aerosp. Expo.*, no. 588, 2012, pp. 1–24.
[10] K. Y. Ma, P. Chirarattananon, S. B. Fuller, and R. J. Wood, “Controlled flight of a biologically inspired, insect-scale robot,” *Sci.*, vol. 340, no. 6132, pp. 603–607, 2013.
[11] P. Chirarattananon, K. Y. Ma, and R. J. Wood, “Perching with a robotic insect using adaptive tracking control and iterative learning control,” *The Int. J. Robot. Res.*, vol. 35, no. 10, pp. 1185–1206, 2016.
[12] D. Coleman, M. Benedict, V. Hirishikeshaven, and I. Chopra, “Development of a Robotic Hummingbird Capable of Controlled Hover,” *J. Amer. Helicopter Soc.*, vol. 62, no. 3, pp. 1–9, 2017.
[13] A. Roshanbin, H. Altartouri, M. Karasek, and A. Preumont, “Colibri: A hovering flapping twin-wing robot,” *Int. J. Micro Air Veh.*, vol. 9, no. 4, pp. 270–282, 2017.
[14] H. V. Phan, T. Kang, and H. C. Park, “Design and stable flight of a 21 g insect-like tailless flapping wing micro air vehicle with angular rates feedback control,” *Bioinspiration Biomimetics*, vol. 12, no. 3, 2017, Art. no. 0 36006.

[15] M. Karasek, F. T. Muijres, C. De Wagter, B. D. Remes, and G. C. de Croon, "A tailless aerial robotic flapper reveals that flies use torque coupling in rapid banked turns," *Science*, vol. 361, no. 6407, pp. 1089–1094, 2018.

[16] F. Fei, Z. Tu, J. Zhang, and X. Deng, "Learning extreme hummingbird maneuvers on flapping wing robots," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2019, pp. 109–115.

[17] S. B. Fuller, "Four wings: An insect-sized aerial robot with steering ability and payload capacity for autonomy," *IEEE Robot. Automat. Lett.*, vol. 4, no. 2, pp. 570–577, Apr. 2019.

[18] T. L. Hedrick, B. Cheng, and X. Deng, "Wingbeat time and the scaling of passive rotational damping in flapping flight," *Science*, vol. 324, no. 5924, pp. 252–255, 2009.

[19] D. Campolo, M. Azhar, G.-K. Lau, and M. Sitti, "Can dc motors directly drive flapping wings at high frequency and large wing strokes?" *IEEE/ASME Trans. Mechatronics*, vol. 19, no. 1, pp. 109–120, Feb. 2014.

[20] L. Hines, "Design and control of a flapping flight micro aerial vehicle," Ph.D. dissertation, Carnegie Mellon University, Pittsburgh, PA, USA, Aug. 2014.

[21] Z. Tu, F. Fei, J. Zhang, and X. Deng, "An at-scale tailless flapping-wing hummingbird robot. i. design, optimization, and experimental validation," *IEEE Trans. Robot.*, vol. 36, no. 5, pp. 1511–1525, Oct. 2020.

[22] J. P. Whitney and R. J. Wood, "Aeromechanics of passive rotation in flapping flight," *J. Fluid Mechanics*, vol. 660, pp. 197–220, 2010.

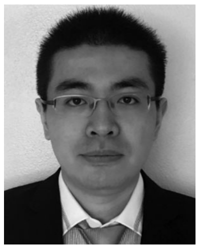
[23] F. Fei, Z. Tu, Y. Yang, J. Zhang, and X. Deng, "Flappy hummingbird: An open source dynamic simulation of flapping wing robots and animals," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2019, pp. 9223–9229.

[24] T. P. Lillicrap *et al.*, "Continuous control with deep reinforcement learning," 2015, *arXiv:1509.02971*.

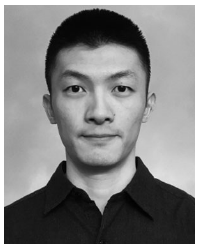
[25] Y. Duan, X. Chen, R. Houthooft, J. Schulman, and P. Abbeel, "Benchmarking deep reinforcement learning for continuous control," in Proc. *Int. Conf. Mach. Learn.*, 2016, pp. 1329–1338.

[26] X. B. Peng, M. Andrychowicz, W. Zaremba, and P. Abbeel, "Sim-to-real transfer of robotic control with dynamics randomization," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2018, pp. 1–8.

[27] S. Armanini, M. Karasek, G. De Croon, and C. De Visser, "Onboard/offboard sensor fusion for high-fidelity flapping-wing robot flight data," *J. Guidance Control Dyn.*, vol. 40, no. 8, pp. 2121–2132, 2017.



Zhan Tu received the B.S. degree in optoelectronics engineering from Beihang University, Beijing, China, in 2009, the M.S. degree in computer science and technology from Tsinghua University, Beijing, in 2012, and the Ph.D. degree in mechanical engineering from Purdue University, W Lafayette, IN, USA, in 2019.



Fan Fei received the B.S. degree from the Nanjing University of Aeronautics and Astronautics, Nanjing, Jiangsu, China, in 2011, the M.S. and Ph.D. degrees from Purdue University, W Lafayette, IN, USA, in 2014 and 2019 respectively, all in mechanical engineering.

Xinyan Deng received the B.S. degree in automation from Tianjin University, Tianjin, China, in 1995 and the Ph.D. degree in mechanical engineering from the University of California, Berkeley, CA, USA, in 2004.

She was an Assistant Professor with the Department of Mechanical Engineering, University of Delaware, Newark, DE, USA, from 2004 to 2009. She is currently an Associate Professor with the School of Mechanical Engineering, Purdue University, West Lafayette, IN, USA. Her research interests include bio-inspired robots, micro air vehicles, experimental fluid mechanics, dynamics, and control.