nature machine intelligence

Article

https://doi.org/10.1038/s42256-024-00893-9

Wing-strain-based flight control of flapping-wing drones through reinforcement learning

Received: 4 February 2024

Accepted: 5 August 2024

Published online: 20 September 2024

Check for updates

A list of authors and their affiliations appears at the end of the paper

Although drone technology has advanced rapidly, replicating the dynamic control and wind-sensing abilities of biological flight is still beyond reach. Biological studies reveal that insect wings are equipped with mechanoreceptors known as campaniform sensilla, which detect complex aerodynamic loads critical for flight agility. By leveraging robotic experiments designed to mimic these biological systems, we confirm that wing strain provides crucial information about the drone's attitude angle, as well as the direction and velocity of the wind. We introduce a wing-strain-based flight controller that employs the aerodynamic forces exerted on a flapping drone's wings to deduce vital flight data such as attitude and airflow without accelerometers and gyroscopic sensors. The present work spans five key experiments: initial validation of the wing strain sensor system for state information provision, control in a single degree of freedom movement environment with changing winds, control in a two degrees of freedom movement environment for gravitational attitude adjustment, a test for position control in windy conditions and a demonstration of precise flight path manipulation in a windless condition using only wing strain sensors. We have successfully demonstrated control of a flapping drone in various environments using only wing strain sensors, with the aid of a reinforcement-learning-driven flight controller. The demonstrated adaptability to environmental shifts will be beneficial across varied applications, from gust resistance to wind-assisted flight for autonomous flying robots.

Understanding the aerodynamic mechanism[1–3] of biological flight has led to remarkable progress in the system design of flapping-wing micro aerial vehicles (FWMAVs) such as RoboBee[4], DelFly[5], KUBeetle[6] and Purdue Hummingbird[7]. Although these FWMAVs offer high agility and energy efficiency at a small scale over quadcopters[8] and fixed-wing drones[9], they encounter difficulties in terms of flight control and lightweight design[10]. Specifically, controlling them in windy conditions is challenging due to their complex dynamics and small size, which limit the generation of sufficient thrust force and moments[11]. To control FWMAVs, sensors and feedback systems combining inertial measurement units (IMUs), vision cameras and proportional-integral-derivative controllers are commonly utilized[12,13]. However, they are not sufficient for controlling FWMAVs for several reasons. First, IMUs are typically located on the bodies, requiring body movement for disturbance detection, which can often cause delayed stabilization of the robot[14]. Second, existing sensors are inadequate to handle the flexible bodies of flapping-wing drones due to their rigidity. Third, model-based controllers with limited

e-mail: jskoh@ajou.ac.kr; sy84han@ajou.ac.kr; dskang@ajou.ac.kr

tuning parameters struggle to manage the complex aerodynamics of flexible flapping wings[15,16].

Flying insects (for example, dragonflies, hawk moths and hoverflies) possess a high level of flight ability, capable of manoeuvres like hovering, turning and gliding even in unsteady aerodynamic environments[17–22]. Notably, some species like the globe skimmer dragonfly traverse thousands of kilometres, from India to East Africa[23], by utilizing fast, high-altitude airstreams and navigating in seasonally favourable directions[23–27]. The superior sensing and control mechanism are attributed to active stabilization supported by complex sensory systems including mechanosensory, vision and olfaction[28–30]. Among these sensory organs, the campaniform sensilla, which are strain-detecting mechanosensors distributed on the wings, enable immediate, proprioceptive flight control[31,32]. The wings of flying insects are subject to combined inertial–aerodynamic (aeroelastic) loads because of various aerodynamic phenomena, including airflow stagnation and separation and nonlinear phenomena such as vortex growth and shedding[33,34]. Moreover, recent research claims that the campaniform sensilla detect the Coriolis force and function as inertial sensors based on electrophysiology data[35–37] and numerical simulations[38,39]. From an engineering perspective, the direct measurement of wing load is a notable advantage for flight control because the unsteadiness in aerodynamics makes it hard for conventional control systems to accurately estimate the wing load in real time[40]. Therefore, inspired by the control systems of these flying insects, attempts have been made to develop controllers that can sensitively respond to changes in airflow by measuring the wing deformation of aircraft[14]. However, there have been no successful cases of flapping drones because the aerodynamics between easily deformable wings and airflow are extremely complex[41,42], and developing sensing and control algorithms for stable flight and manipulation has been difficult, even with supercomputers[43].

In this Article, we present a wing-strain-based flight controller, which we term fly-by-feel. Given the computational and complexity challenges associated with modelling aerodynamics, we employed the powerful and cutting-edge method of reinforcement learning (RL). Utilizing wing strain sensors, our goal is to assess the feasibility and reliability of such a system in autonomously determining flight paths. Our experimental approach is structured into five distinct phases. (1) Validation as a state informative sensor system: The initial phase of the research focuses on verifying the wing strain sensor's ability to provide useful and reliable state information for drone control. This foundational step is crucial in confirming the sensory system's detection capabilities. (2) Single degree of freedom (1 DOF) flight path control experiment: This stage evaluates how effectively the drone can adapt to changing wind speeds and directions and how well it can manage one-dimensional (1D) flight paths using data from strain sensors. (3) Two degrees of freedom (2 DOF) attitude control experiment: This experiment tests the drone's ability to identify and maintain optimal pitch angles through gravity-based attitude adjustment, which is vital for precise posture control and flight path modification. (4) Position control in a windy environment: This phase tests the drone's capability to maintain its position in a wind-influenced environment solely using wing strain sensor data. Success in this experiment would underline the potential of strain sensors to manage complex aerodynamic conditions without additional sensory inputs. (5) Wing-strain-based flight path control in a calm environment: The final experimental step involves verifying the drone's ability to control its flight path in a windless environment using only the information provided by wing strain sensors, which would demonstrate the precision of location and attitude adjustments possible with this system. Each stage of this study meticulously verifies different aspects of the system, building upon each other to substantiate the utility and effectiveness of an autonomous flight control system based on strain sensor data. Through this series of empirical tests, we aim to implement the biological hypothesis that wing strain sensors are useful for immediate flight control in response to airflow changes into robotic experiments and explore the potential of wing strain-based control systems in real flight situations.

Main fly-by-feel control incorporating wing strain RL

Inspired by insects' agile flight abilities, the fly-by-feel control system shown in Fig. 1 enables rapid adaptation to complex airflow conditions and immediate response to sudden fluctuations. The campaniform sensilla are sensory organs distributed on the wing base that sense the strain on the wing[44] (Fig. 1a, middle, and Supplementary Fig. 1). The spike rate of neurons depends on the flapping motion of the wing, as shown in Fig. 1a, left, and the campaniform sensilla can detect the rotational rate during flight[37]. To develop an efficient autonomous flight system by understanding the role of the campaniform sensilla in flying insects, ultrasensitive crack-based strain sensors[45–47] were attached to the wing bases of commercial wing-flapping drones inspired by the sensory system of a flying insect (Fig. 1a, right).

The crack-based strain sensors attached to the wing base collect aeroelastic information in various states (Fig. 1b). After analysing wing deformation with a 1D convolutional neural network (CNN) model, we designed an experimental setup in complex airflow to study control of aircraft attitudes using external flow information (Fig. 1c). The measured wing strain data is transmitted to a RL-based controller in real time, and the RL agent is trained to find the optimal policy guaranteeing the highest reward via a RL algorithm. As a result, the RL-based flight controller can recognize different states and control the thrust and direction of the drone.

The crack-based strain sensors, comprising metal layers deposited on a polyimide substrate, are very thin (~7.5 µm) and lightweight (~3 mg) and thus have little effect on the structural deformation of the wing (Fig. 1d, right). The inset scanning electron microscope image shows that channelled nanoscale cracks on the metal surface are regularly aligned at intervals of 3 µm, and when the sensor is mechanically deformed, the gap between cracks widens and causes a large change in electrical resistance. Due to nanoscale cracks on the metal layer surfaces, the crack-based sensor has a higher gauge factor (~30,000) and signal-to-noise ratio than conventional strain gauges (Supplementary Fig. 2). The flapping-wing drone uses two motors located on the body (motor 1) and tail (motor 2) to control its motion (Fig. 1d, left). The attachment of crack-based strain sensors to a drone's flapping wings allows for the precise and stable measurement of trajectory changes and the aerodynamic forces exerted on the wing over a long time (Supplementary Fig. 3 and Supplementary Video 1).

Validation as state-informative sensor system

To analyse how accurately the wing deformation signal contains information about changes in wind direction and speed, we designed an experiment in which the orientation of the drone was systematically controlled, allowing us to create scenarios with wind blowing from multiple directions and varying in speed. In the experiment, the relative wind direction was changed by controlling the drone's orientation. The drone rotated according to different combinations of yaw, pitch and roll angles, resulting in a total of 62 different wind vectors (Fig. 2a and Supplementary Video 2). Also, the speed of the wind varied in three steps, resulting in a total of 186 cases of blowing wind (Fig. 2b and Supplementary Table 1). The strain sensors attached to the wing base measured the strain, which was affected by the complex aerodynamic loads on the flapping wings. Not only does the sensor signal change with flapping-wing power (Supplementary Fig. 4), but it also varies with the direction and speed of the blowing wind (Supplementary Fig. 5). In this way, various information from the environment is encoded into the sensor signals. To decode the information, we used a temporal 1D CNN and classified the direction and speed of the blowing wind. The raw sensor signals are given as input to the model, and the model predicts the wind direction and speed as outputs (Supplementary Fig. 6).

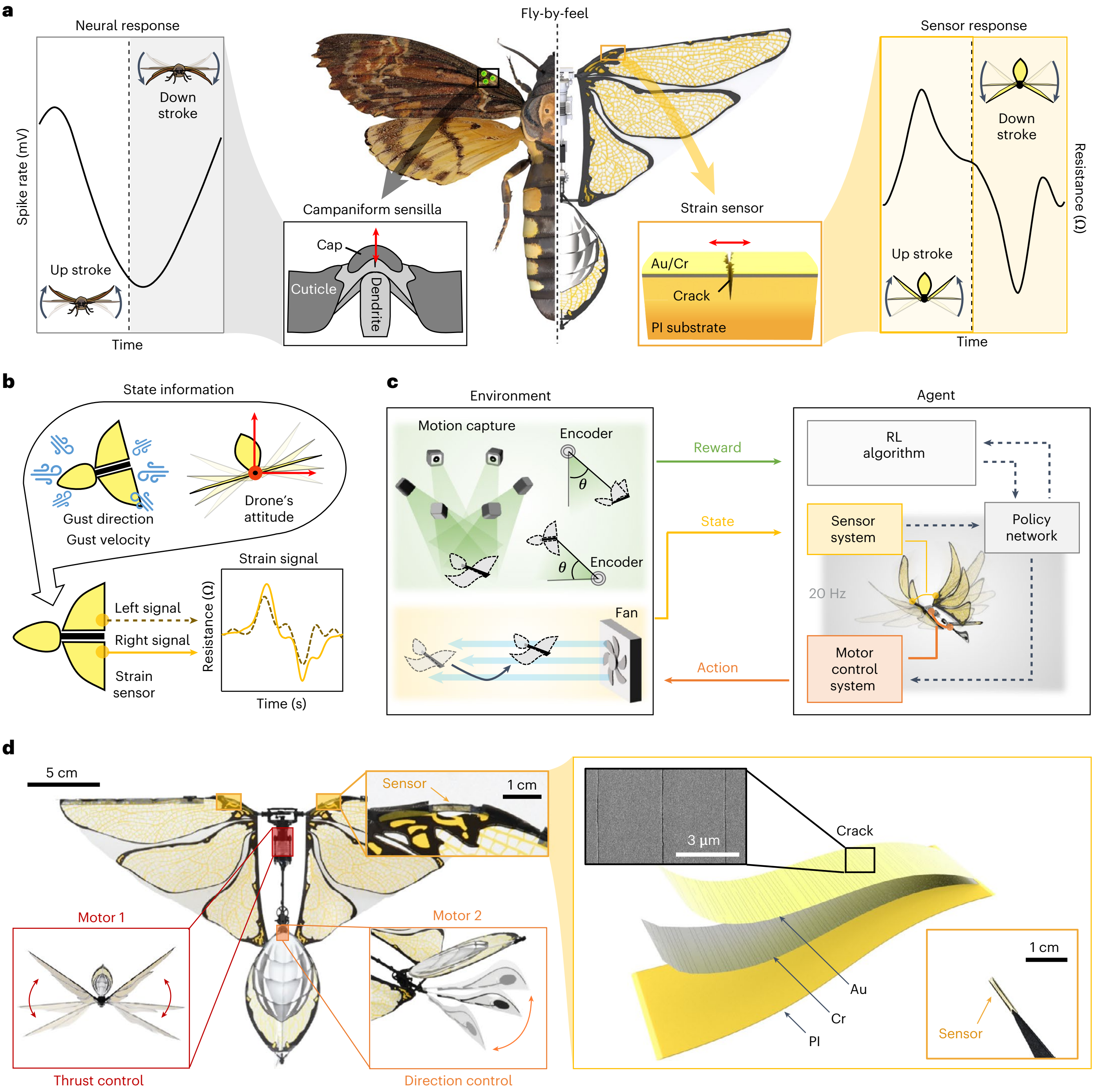


Fig. 1 | Fly-by-feel-based flight control strategy. a, Mechanism for detecting wing deformation in death's head hawk moth and flapping drone. Left, Death's head hawk moth: spike rate change via campaniform sensilla in response to wing movement. Right, Flapping drone: resistance change of crack-based sensor according to wing flapping. **b**, The state information that can be derived from the wing deformation measured by the crack-based strain sensors in the fly-by-feel system. **c**, The RL-based flight control system using wing strain information. **d**, Photograph of a flapping drone equipped with crack-based strain sensors on the wing bases and two motors on the body and tail for thrust and direction control, respectively. Inset: Channelled nanoscale cracks on the metal surface are regularly aligned at intervals of 3 μm. PI, polyimide.

After training, we verified the feasibility of state recognition using only strain information based on the calculated error from trained regression (Fig. 2c) and a confusion matrix from the trained classification model (Fig. 2d). Out of a total of 12,416 validation cases, the mean absolute value of theta error (θ_{error}) was estimated to be 29°. Specifically, 76% of the angle errors fell inside the range 0–33°. In addition, the mean value of the speed error (s_{error}) was calculated as 26%, and 63.5% of the speed errors fell within the range of 0–30%. For the classification model, the mean area under the curve was approximately 0.99 (Supplementary Fig. 6c), and the model classified the wind direction and speed with a mean accuracy of 80% (Fig. 2d), which means the model can predict the direction and speed of the blowing wind with about 80% of accuracy from 186 cases. In addition, the prediction accuracy is displayed as a colour map in spherical coordinates (Fig. 2e). As shown in the colour map, the prediction accuracy was highest in the front, lower-left and lower-right regions, and the accuracy was the lowest in the upper-rear

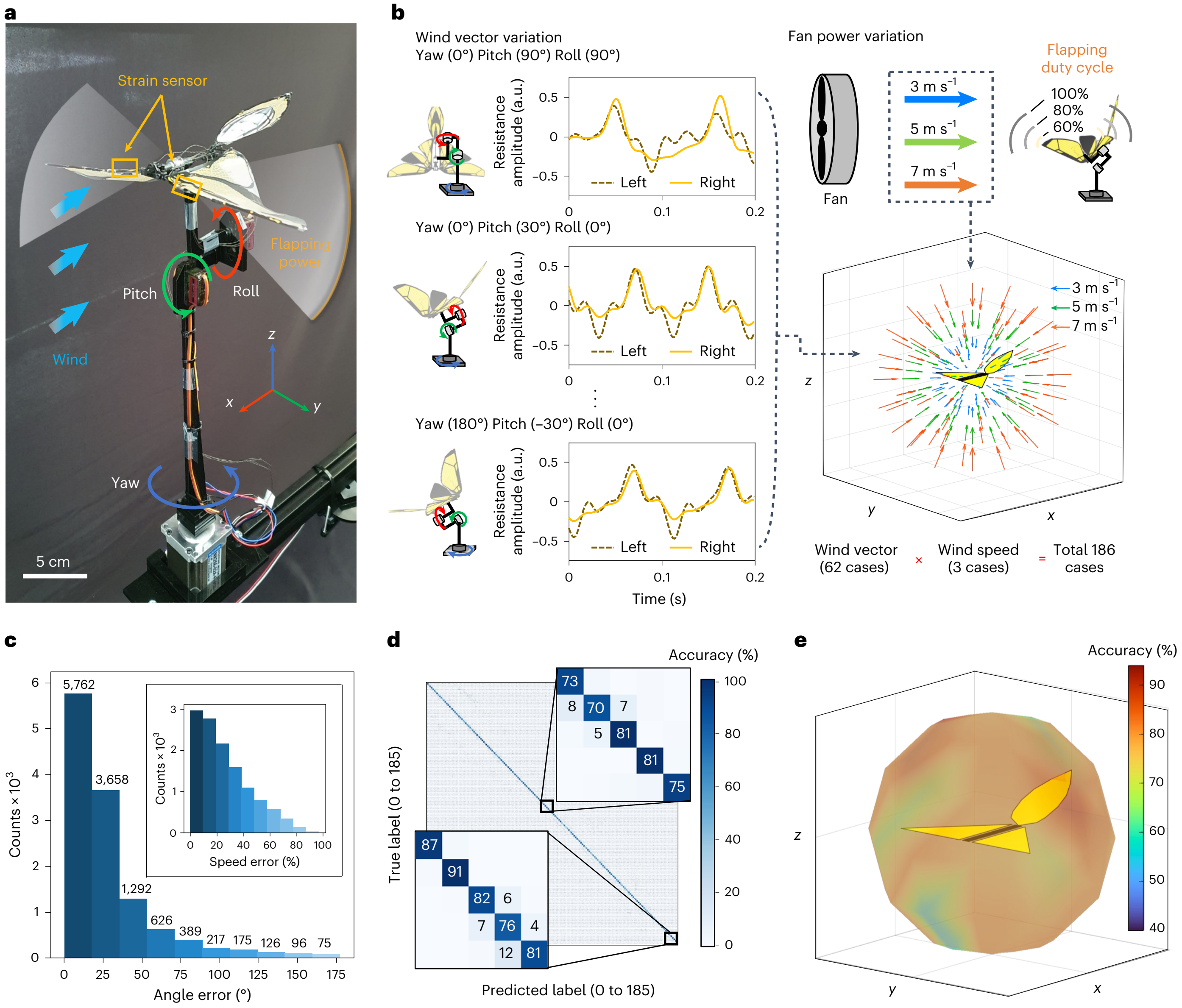


Fig. 2 | Aerodynamic feature extraction from wing strain data. a, Experimental design for changing the wind direction and speed. **b**, Wing strain data (186 sets) were acquired by changing the wind direction and speed. **c**, Histogram according to the error range, angle error and speed error. **d**, Confusion matrix from the test dataset. **e**, Visualization of the classification accuracy as a colour map in spherical coordinates. Red-coloured range indicates well-classified trained model.

region (Supplementary Fig. 7). This is because the blowing wind from the front, left and right regions made the greatest mechanical deformation at the wing base. These results confirm that the drone can determine the wind direction and speed for various airflows, especially when the front, left and right sides are windward, by using only the strain information collected by the two sensors on the wing base.

Control in a 1 DOF movement environment

Next, we designed an experimental setup that limited the drone's flight path to a single DOF to confirm its ability to utilize strain data for real-time adaptation to varying wind direction and speed, employing a RL controller (soft actor-critic (SAC) algorithm; described in detail in Methods). The drone's body was anchored to a rotational encoder for simplified circular motion and rotated clockwise in the x–y plane as the flapping power increased. In this setup, the direction and speed of the apparent wind acting on each wing changed according to the motor encoder's rotation angle (α). At the initial position, the wind blew from the rear of the drone at a speed of 3 m s^{-1}, which is not enough to move the drone but enough to assist it after it starts to fly. When thrusting force was generated by the flapping motion, a torsion spring dragged the drone back to the initial position. After the drone overcame the drag and passed over 90° of rotational angle (α), it had to overcome a faster wind speed (5 m s^{-1}) that made it more difficult for the drone to reach the target position. The sensor signal at the wing base was used as state information in the RL algorithm. The motor encoder at the centre measured the rotational angle change, which was used only to calculate the reward during training (Fig. 3a and Supplementary Fig. 8a). The angle of the target position was 180°, and Fig. 3b shows images from one training episode (Supplementary Video 3). The power increased, then quickly decreased and finally stabilized as the angle fluctuated near 180°, as shown in Fig. 3c. Figure 3d shows that the drone reacted sensitively to the state change. As the episode advanced, the drone chose another strategy and slowly increased and decreased the motor power to approach the target position more smoothly.

We next compared the performance of the trained drone in these experiments with that of various untrained drones. There are three

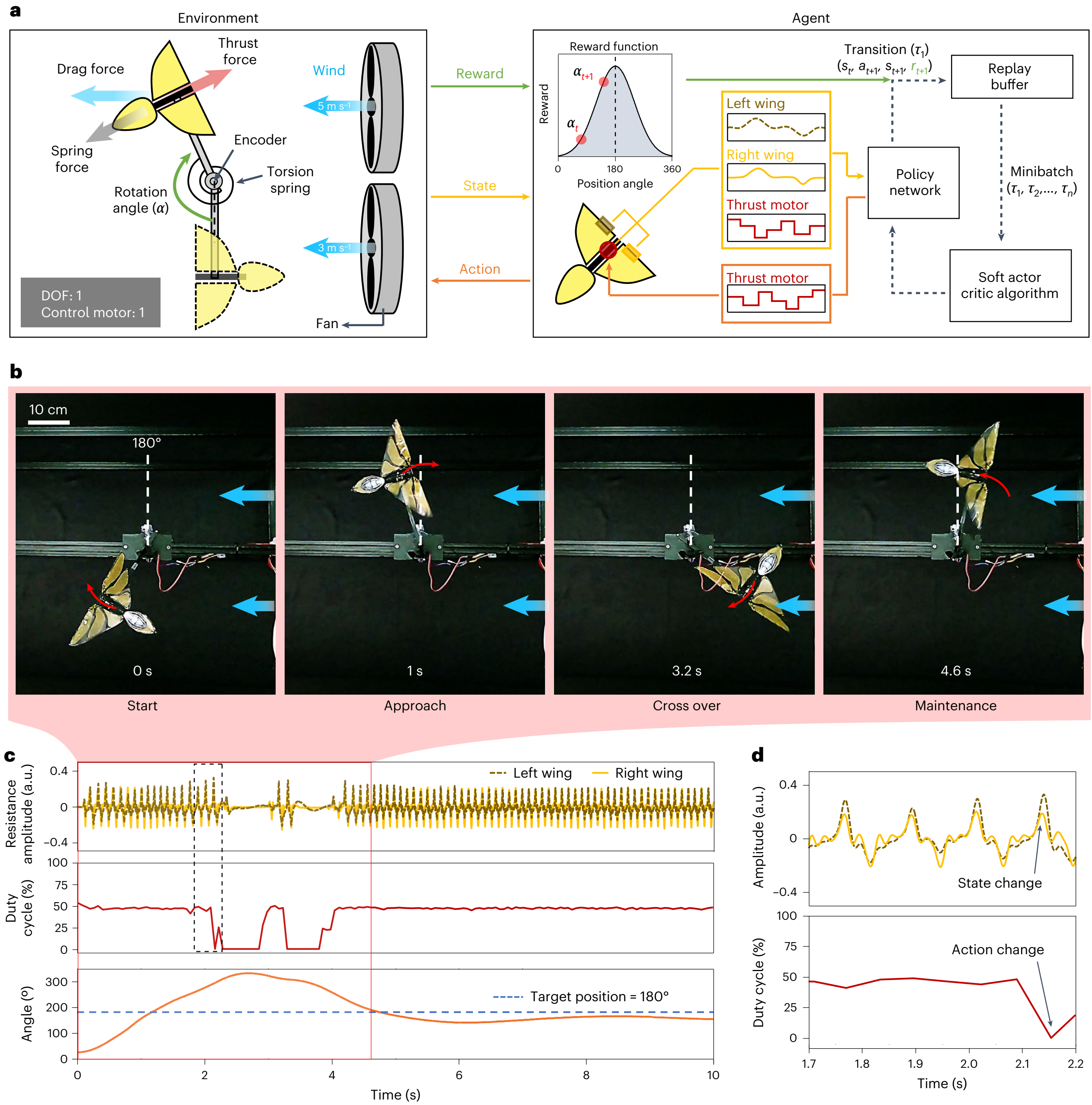


Fig. 3 | Position control by sensing the wind direction and speed, which vary depending on position. **a**, Schematic illustration of the environment and agent in the designed experiment with 1 DOF. Two sensor signals and thrust motor power were used as input, and the rotation angle of the encoder was used as a variable of reward function. **b**, Snapshots of an example episode in which a drone controls the motor power to maintain a target position. **c**, Strain sensor signal (top), motor power (middle) and drone angle (bottom). The range in the red box shows the signals in the snapshots in **b**. After the drone passes over the target, it stops flapping to reach the target position. **d**, Magnified view of the black dotted box in **c**. Trained drone reacts with agility as sensor signals change.

types of untrained drones: flapping with minimum, maximum and random motor power. Compared to the untrained cases, the RL-based drone adapted to the environment and outperformed on accumulated reward (score) as the episodes increased (Supplementary Fig. 8b,c). Furthermore, the drone maintained the target position at a different angle (270°) and even returned to the target position when there were dragging-backward disturbances (Extended Data Fig. 1). When the target position is changed to 270°, the only thing that has changed is the reward function. These results show that the RL-controlled drone can feel the direction of the wind by the strain information and control its flapping frequency in a 1 DOF environment to maintain the target position where the highest reward is given.

Control in a 2 DOF movement environment

In the experimental environment, the drone had to find the optimal pitch angle and maintain balance to increase the rotational angle, which

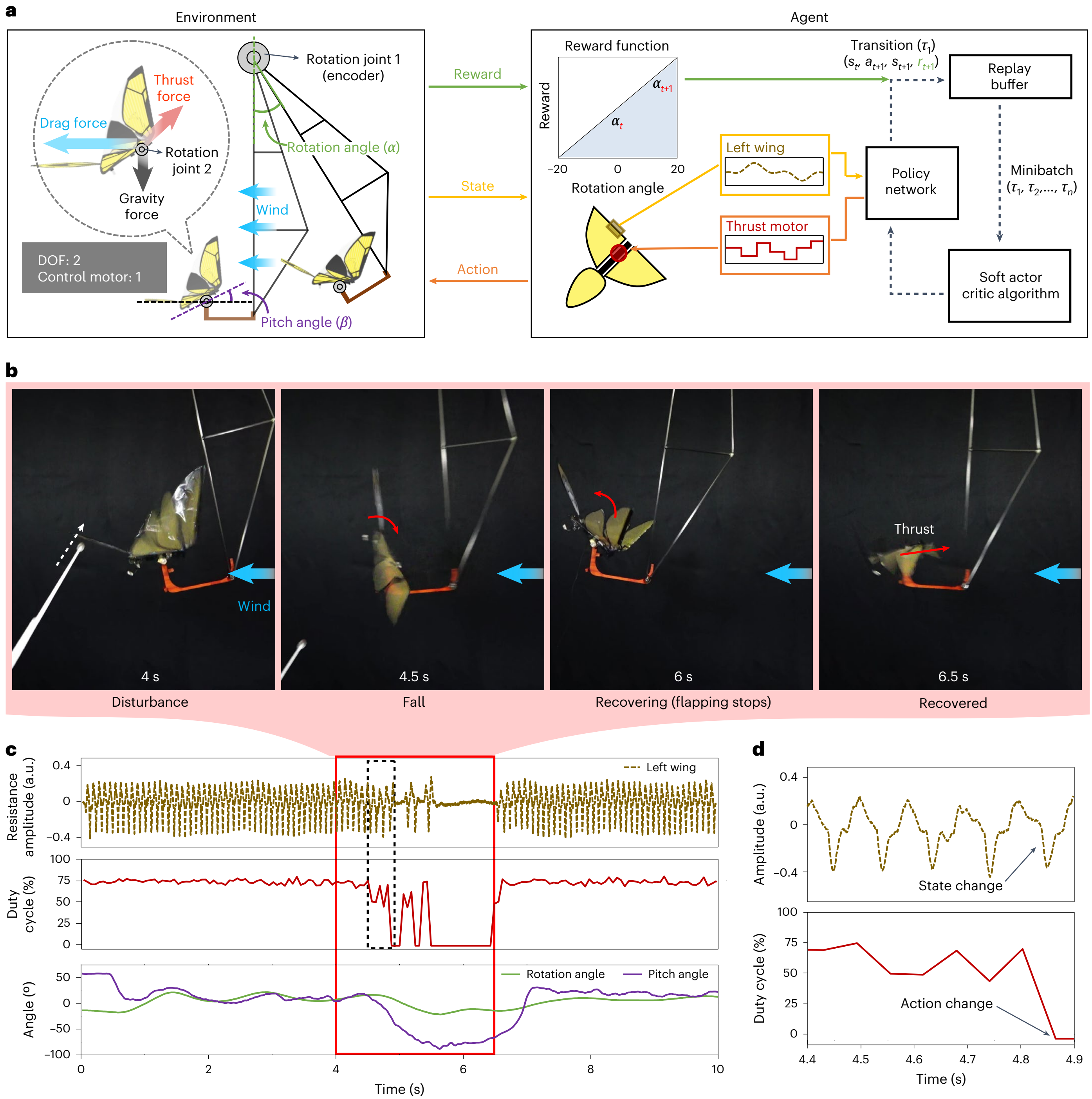


Fig. 4 | RL in a 2 DOF environment. a, Schematic illustration of the environment and agent in the designed experiment with 2 DOF. One sensor signal was used as input, and the rotation angle of the encoder was used as a variable of reward function. **b**, Snapshots of a representative episode. The drone controls the motor power despite the induced disturbance. **c**, Sensor signal (top), motor power (middle) and rotational and pitch angles of the drone (bottom). The range in the red box shows the signal in the snapshots in **b**. As the trained drone recognizes the falling state, it stops flapping to recover the proper pitch angle. **d**, Magnified view of the black dotted box in **c**.

resulted in a higher reward. The body of the drone was anchored to a rotational encoder (rotation joint 1) by a triangle-shaped rod, and the centre of the body frame was attached to a pin joint (rotation joint 2) so that the drone's pitch angle (*β*) varied from −30° to 60°. Each joint moved separately as the drone's pitch angle changed, regardless of the rotation angle (*α*) (Fig. 4a and Supplementary Fig. 9a). An apparent wind caused a drag force, pushing the drone backward and creating counterclockwise momentum in rotation joint 2. Because the reward increased proportionately with the rotation angle, the drone received a higher reward when rotation joint 1 rotated counterclockwise. Therefore, the drone had to precisely control the horizontal pitch angle to obtain the highest score during the episode. The drone aimed to simply fly forward through the apparent wind; however, the drone had difficulty maintaining the optimal pitch angle as the rotational angle increased.

In the experiment, the drone had to detect its own pitch angle based on the data collected by the wing strain sensor. As the drone is constrained to move clockwise or counterclockwise around the centre of the motor encoder, only one crack-based sensor attached to the left side of the wing base was enough to acquire state information in the environment. The falling state is defined as the drone failing to maintain balance and the pitch angle decreasing to less than 0°. In this state, because the direction of the blowing wind is perpendicular to the flapping direction, a flapping motion does not assist in increasing the rotation angle (α) or pitch angle (β). The only way to restore from this state is to stop flapping and wait for the ambient wind to recover the pitch angle horizontal with the direction of the wind. Figure 4b shows this recovery process of the trained drone during a disturbance (Supplementary Video 4). As the drone balanced at the optimal pitch angle, the tail of the drone was lifted intentionally to induce the falling state. The drone recognized the falling state transition within 0.1 s and decreased its motor power to zero to recover the optimal pitch angle (Fig. 4c and Supplementary Fig. 9b). And accordingly, the amplitude of the sensor signals also decreased in response to the decreased motor power (Fig. 4d). Regardless of the flapping motion, the drone recognized the pitch angle with only one crack-based sensor and quickly recovered to reach a higher rotational angle.

The RL-agent-controlled drone's performance improved with training (Supplementary Figs. 9d,e). The trained drone recognized the subtle change in state well and successfully flew forward through the apparent wind to get the highest reward during a given episode (Extended Data Figs. 2a–d and Supplementary Video 5). Moreover, even if disturbances were intentionally induced, the drone maintained its balance and controlled the motor power with agility (Extended Data Figs. 2e–h and Supplementary Video 6). These results show that the RL-based drone can recognize the forward range (90°) of the blowing wind and learn to get the highest reward during a given episode by maintaining the optimal pitch angle in the designed environment.

Position control in a windy environment

Flapping-wing drones typically use IMU and optical flow sensors for three-dimensional (3D) position and attitude control[48,49]. In contrast, we indirectly measure 3D space by using two strain sensors to determine x, y, z positions and speed from 1D strain data. We tested flight control by training a drone to navigate towards a target in unsteady airflow, where it earns higher rewards by closely reaching the target. The drone relies solely on airflow variability to locate the target, necessitating the decoding of aeroelastic information from wing deformation in complex airflow conditions. We used both flapping wings and a tail fin for control in 3D space.

Inside the wind tunnel, we set up an experimental apparatus with motion capture cameras to record position data and a jig to protect the drone from collision damage during repetitive training (Fig. 5a). The drone's position information, measured by the motion capture cameras, was not used as a control input but was used as a variable in the reward function to provide a reward for problem training (Supplementary Fig. 10). The airflow inside the wind tunnel was analysed using an optical flow estimation method, and the complex asymmetric flow was visualized through colour maps representing the wind direction and speed (Supplementary Fig. 11 and Supplementary Video 7).

Figure 5b shows a short period (between 7.7 and 9.4 s) during training as the drone flies to the middle target position (Supplementary Video 8). The drone's motion is categorized into four types: clockwise rotation, counterclockwise rotation, acceleration and deceleration—with its x-axis movement driven by motor 1's thrust changes and y-axis rotation controlled by motor 2's directional adjustments. Figure 5c shows the trajectory of the drone over time in 3D space, facilitating a comparative analysis with the change in the output of the motors. The trained RL-based controller adjusts the output of the motors based on the wing deformation measured by the strain sensors to determine the optimal path with the highest reward (Fig. 5d).

Extended Data Fig. 3 demonstrates that the trained drone substantially outperformed the untrained one in maintaining proximity to the target, demonstrating effective control. The trained drone consistently stayed near the target, showing a reduced standard deviation in its average position and increased training efficiency, as evidenced by decreasing entropy and increasing scores. This confirms the effectiveness of our control strategy using only two strain sensors for position and attitude control in a flapping-wing drone, without relying on vision sensors or IMUs.

Following this, Extended Data Fig. 4 further validates the autonomous flight control by displaying cases where the drone was trained to reach alternative target positions and comparing these results with those of untrained drones. Target positions on both the left and right sides of the drone were tested to evaluate position control capabilities. The drone successfully detected and approached the target in both scenarios, as documented in Supplementary Videos 9 and 10. The observed performance discrepancy between the two positions likely stems from variations in the flow field and the drone's hardware limitations.

Flight path control in a windless environment

To verify the system's robust controllability, we conducted a free flight demonstration in a windless environment, relying solely on two strain sensors to manage various manoeuvres. Unlike previous tasks that maintained position or attitude by detecting apparent wind, this challenge involves identifying the aircraft's speed, direction, rotation and attitude by sensing induced wind and acceleration, requiring delicate sensing and control.

We conducted experiments in an enough space (width, 6 m; depth, 8 m; height, 4 m) for drone's natural flight motion (Fig. 6a left). For reliable training, we designed the drone launcher to ensure the same start conditions such as take-off speed (Supplementary Fig. 12). In addition, very thin and lightweight wires (40 milligram per metre) were used to minimize the tensile force of the tether applied to the drone. By utilizing 12 sets of motion capture cameras, we were able to adjust the reward based on the drone's trajectory (Extended Data Fig. 5). Our first objective was to manage simple left and right motion during flight to verify the reliability of trajectory control (Extended Data Fig. 6). The RL agent enhanced its navigation skills by learning from human demonstrations, managing to fly in both zigzag and circular patterns with the aid of a Replay buffer filled with human-controlled episodes. Additionally, by adeptly adjusting the directional motor's power, the agent improved drone control, achieving higher scores and reduced entropy, a notable improvement over the 30% success rate observed with manual trajectory control (Supplementary Fig. 13).

In addition to controlling the directional motor, the thrust motor power was also trained for adjustment during free flight. Our objective was to train the RL agent to maintain altitude higher or lower than the takeoff altitude. Remarkably, the RL agent successfully orchestrated the thrust motor power and directional motor power to sustain appropriate altitude, achieving higher scores as the episodes progressed (Extended Data Fig. 7).

Furthermore, we compared the odometry and ground truth of the trajectory (Fig. 6c,e) to confirm that the sensor could predict trajectory in the windless environment. In previous research, the accuracy of odometry has been improved by combining an airflow sensor that can detect the wind speed and direction with a IMU-based attitude controllable multirotor[50,51]. Because wing deformation also carries information about wind speed and direction, we were able to reconstruct the drone's driving path. We trained a drone control model using 0.5 s sensor data inputs, separating episodes into training and validation sets at a 9:1 ratio. The model featured a dropout layer to prevent overfitting. Over five sessions, we observed reduced mean squared error and and recorded the root mean squared error for zigzag, circular and s-curve motions to verify performance and repeatability (Extended Data Fig. 8).

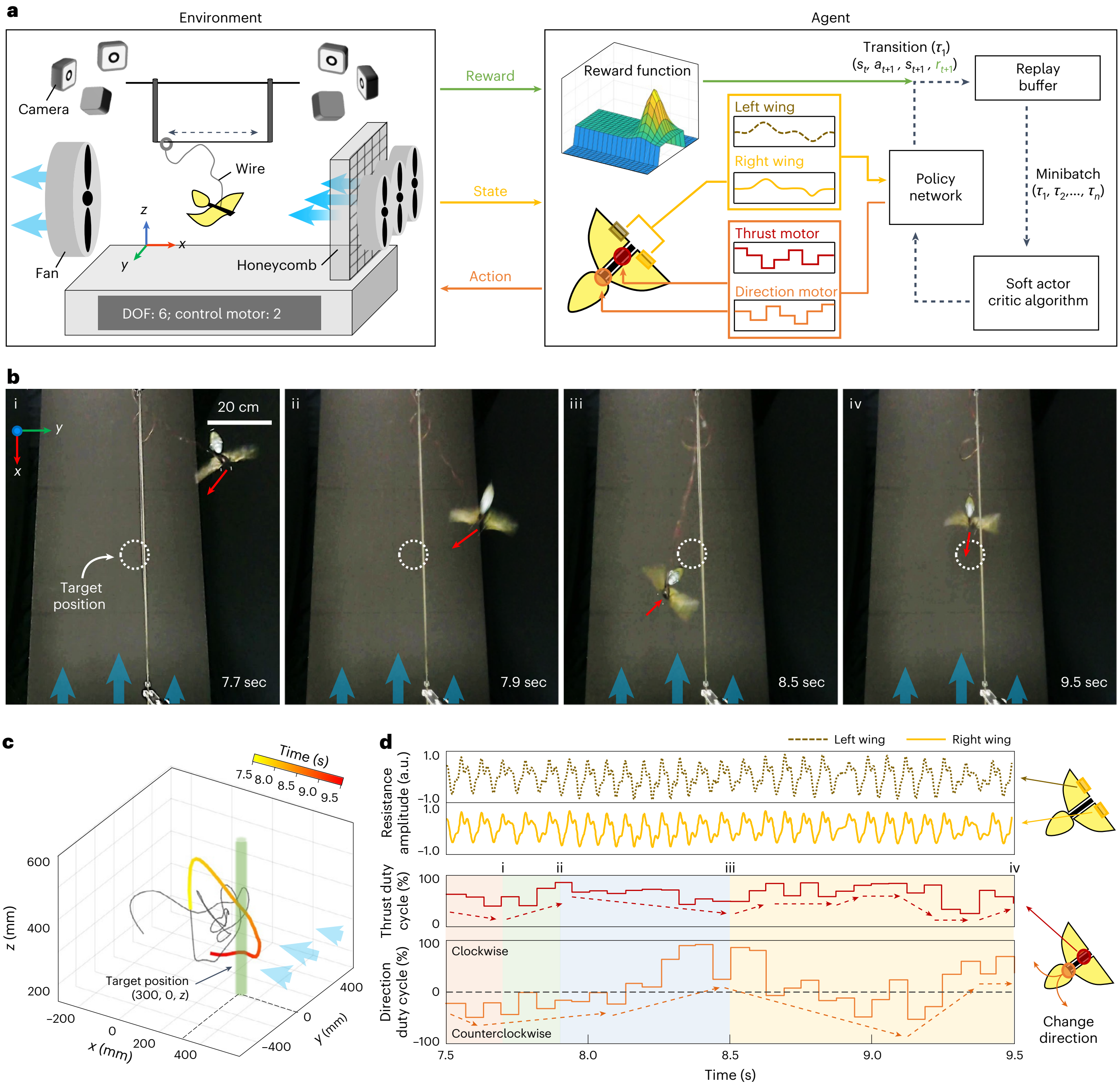


Fig. 5 | Position control of a flapping-wing drone with the proposed fly-by-feel system. a, Wind tunnel experiment setup to train a tethered flapping-wing drone, and the training system configuration. **b**, Representative images of a flapping drone flying to the target position based only on information obtained by the strain sensors attached to the wings. **c**, Trajectory of the drone in *x*–*y*–*z* space. The black line is the flight trajectory before 7.5 s. **d**, Changes in wing strain and the output of motors 7.5–9.5 s during the target positioning flight.

Discussion

Inspired by the sensory mechanisms of flying insects, particularly their ability to sense aeroelastic loads via strain-measuring organs known as campaniform sensilla, we developed a wing-strain-based flight control system. The bio-inspired design and engineering foundation of the proposed control system demonstrated that wing strain information effectively encodes the surrounding environment and flying orientation during flight, even in noisy conditions resulting from high vibration of flapping. We attached ultrasensitive, lightweight crack-based strain sensors to the wings of a drone and measured aeroelastic loads during flight. Although the sensor is lightweight (3 mg for 1 mm width × 5 mm length) and flexible with a thickness of 7.5 μm and a Young's modulus of 2.5 GPa (ref. 52), further miniaturization is necessary for application in subgram drones with wing weights of 3 mg. In terms of miniaturization, crack sensors are advantageous due to their simple structure, which involves metal deposited on a thin plastic film, and can be efficiently fabricated using photolithography or laser ablation. We maintained solely deploying strain sensors within our system to evaluate the effectiveness of these sensors in controlling the flapping-wing drone. Our experimental approach was carefully organized into five distinct phases, aimed at rigorously testing our hypothesis and exploring the role of wing strain sensors in controlling a flapping-wing drone. Each

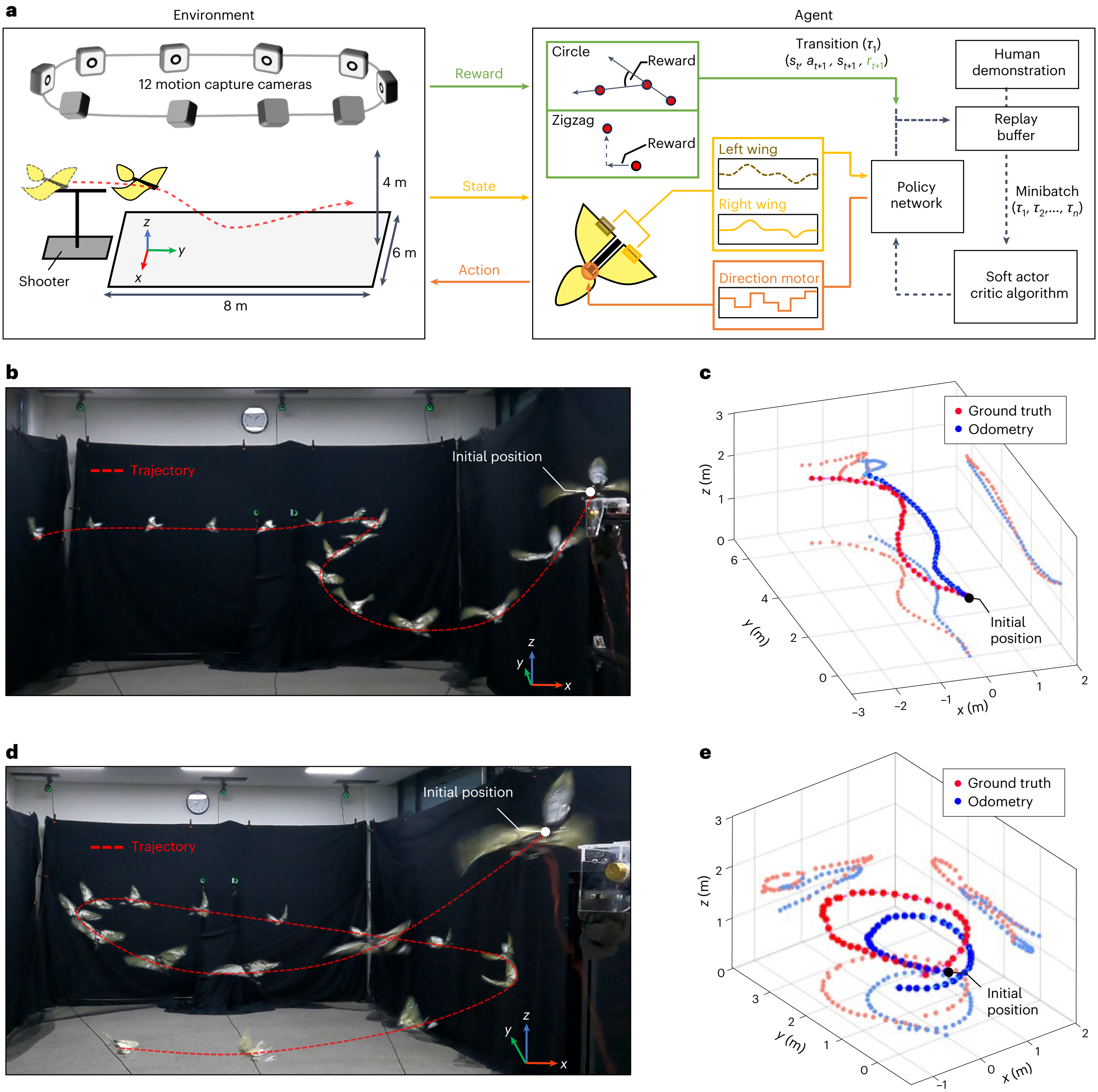


Fig. 6 | Flight path control of a flapping-wing drone in windless environment. **a**, Windless environmental setup designed for training a flapping-wing drone, and the corresponding training system configuration. **b**, Sequential snapshots from a video illustrating the flight of a flapping-wing drone in a zigzag trajectory, relying solely on information acquired through strain sensors attached to its wings. **c**, Comparison between odometry and ground truth trajectory of the drone in zigzag motion. **d**, Sequential snapshots from a video showing a flapping drone executing a circular trajectory, guided exclusively by information obtained from strain sensors attached to its wings. **e**, Comparison between odometry and ground truth trajectory of the drone in circular motion.

phase provided results that reinforced the viability and ingenuity of utilizing wing strain sensors for autonomous drone flight control. These sensors offer strong real-time feedback for system modifications across different atmospheric conditions and enable advanced control strategies without relying on traditional sensors such as IMUs or vision systems. This highlights their potential to improve drone responsiveness and agility in challenging environments.

Consequently, the use of just two strain sensors proved adequate for the drone to effectively track wind cues from multiple directions. The strain sensors at each wing's base comprehensively captured various features such as the wind speed and direction and the drone's attitude and flapping speed. In the robot experiments, following only the cues given from the wind, flight tasks such as positioning, balancing and navigating were performed without employing other flight control sensors such as gyro sensors, accelerometers and vision sensors, which were necessary in previous drones[53].

Developing a wing-strain-based flight control model is very challenging. Natural flyers independently control their wing stroke amplitude, pitch angle amplitude and phase lag to accomplish various manoeuvres[54]. This kinematics generates and sheds vortices into the flow and causes the wing shape to vary. Even simplified computational models require substantial computational resources and often take several days for each simulation to converge[55]. Our study demonstrated that a bio-inspired fly-by-feel control system, utilizing strain sensors and end-to-end RL, directly maps sensory inputs to control outputs. This minimizes manual intervention and enhances accuracy in complex environments. Offering a unique navigation method, this system operates independently of IMU, Global Positioning System or visual cues and adapts to aerodynamic shifts where conventional systems may be compromised.

The campaniform sensilla are hypothesized to play an important role in sensing gravity in flying insects. Although no dedicated gravity sensing organ has been observed in insects, it is possible that the deformation of the wings could vary based on their orientation relative to gravity. However, to accurately differentiate this deformation from that caused by wind and acceleration, multiple sensors may be required to detect complex spatiotemporal deformations of the wing, such as twisting or buckling. In future research, we intend to verify whether attaching multiple sensors to the drone wings enables gravity detection and advanced reactive autonomous flight, such as hovering and wind-assisted flight, without the use of accelerometers and gyro sensors. Concurrently, we will explore the integration of gyroscopes and accelerometers to further enhance control performance. This will involve researching optimization strategies to effectively combine these sensors, aiming to achieve higher precision and stability in the drone's flight control while minimizing sensor redundancy and optimizing data utilization.

Overall, we expect the aerobatic fly-by-feel flight control system to be applied in many scenarios in the near future.

Methods

Method overview

We aim to train RL agents that control a flapping drone. The drone must perform various tasks within an airflow, including maintaining a specific target position and executing different types of turns in desired directions. To train such agents, we employed the most straightforward method: directly collecting experience in real-world interactions and then training the agents with the data.

In detail, we first embedded bio-inspired crack-based sensors into a commercial flapping-wing robot. These sensors measure the deformation of the wing base for every time step, allowing agents to perceive the surrounding environment. This system can be viewed as a partially observable Markov decision process (POMDP). To address partial observability, we extracted the state of the robot using a 1D CNN. The reward is calculated using a motion capture camera and motor encoder. Subsequently, any off-the-shelf RL algorithm can be applied to modify the policy of the agents to maximize the reward. In the following sections, we provide details about the chosen algorithm, the sensor fabrication process and the experimental setup.

RL algorithm

Drone control as a POMDP. We define the system as a POMDP. A POMDP includes a set of states S, a set of actions A, a reward function $r(s,a)$, the probability distribution $s' \sim P(s'|s,a)$ of the next state s' given the current state and action, a set of observations Ω, a set of conditional observation probabilities O and a discount factor $\gamma \in [0,1)$. For every time step, the agent samples an action $a \in A$ using policy $a \sim \pi(a|s_{\text{agent}})$. Here, because the agent cannot access the true state s of the environment, it internally constructs s_{agent}, a belief state about the true state s using history of the observations. When the agent executes an action, the environment makes the state transition, and the agent receives the next observation o'. The goal of RL is to find a policy $\pi(a|s_{\text{agent}})$ that maximizes the expected sum of the discounted reward for a given POMDP $\equiv (S, A, r, P, \Omega, O, \gamma)$. Because we parametrize $\pi(a|s_{\text{agent}})$ through a neural network with parameters θ, we denote the policy as π_θ. Therefore, the goal of RL is to find θ that satisfies the following:

$$\theta = \underset{\theta}{\operatorname{argmax}}\ \mathbb{E}_{\pi_\theta}\left(\sum_t \gamma^t r_t\right)$$

In our experimentation, we discretized the time into 0.05 s intervals. At each step, the agent observes o_t from the signal received by the sensors attached to the drone's wing. Then, the agent constructs the agent state s_t based on the most recent 32 observations, which is equivalent to 1.6 s of time:

$$s_t = (o_t, o_{t-1}, \ldots, o_{t-31}) \in \mathbb{R}^{32}.$$

The agent determines the action according to $a_t \sim \pi_\theta(a_t|s_t)$. Here, note that a_t is the torque of the motor for flapping the wing of a drone, so the action is continuous, such that $a_t \in [0,1]$. When a_t is equal to 1, the motor output is maximized, and when it is 0, the motor output is minimized. In all experiments, the discount factor γ is set to 0.98 empirically. Through the experiments, the drone executes an action and then the environment make the transition, so that we can gather the transition $\tau_t = (o_t, a_t, r_{t+1}, o_{t+1})$ for every time step. These experiences are then collected in the replay buffer, and the RL algorithm samples minibatches from the replay buffer to update the policy. Each minibatch of data consists of 64 transitions. After a certain number of updates, we obtain a new parameter θ' and send it directly to the neural network to replace the previous parameter. This process is repeated until the algorithm converges.

RL algorithm. We use a SAC[56] algorithm as the off-the-shelf policy gradient algorithm. The SAC algorithm offers a high sample efficiency because it reuses the gathered data multiple times, which is a distinctive characteristic from widely used on-policy algorithms[57–59]. High sample efficiency is essential because we are gathering data from the real-world robot and the amount of data is limited.

Additionally, the SAC algorithm is known for its robustness because it is trained with a maximum entropy objective[60]. $\mathcal{H}$ of the policy π as following:

$$\mathcal{H}\left(\pi\left(\bullet, |, \mathrm{s}\right)\right) = \mathbb{E}_{a_t \sim \pi}\left(-\log \pi\left(a_t, |, s_t\right)\right)$$

Because $\mathcal{H}$ measures the randomness of the policy, maximizing $\mathcal{H}$ indicates that the algorithm is in favour of stochastic policy. This ensures that the algorithm finds the policy as random as possible while achieving high rewards. More concretely, SAC aims to maximize the sum of both the reward and the entropy along the trajectory:

$$\theta_{sac} = \underset{\theta}{\operatorname{argmax}}\ \mathbb{E}_{\pi_\theta}\left(\sum_t \gamma^t [r_t + \alpha \mathcal{H}(\pi(\bullet|s_t))]\right)$$

Note that a temperature parameter α is introduced to control the relative importance of the entropy term against the reward term. With the objective given, we update the following parameters: ψ for the soft state value function V_ψ, ϕ for the soft action value function Q_ϕ and θ for the policy π_θ. Here, $V_\psi(s)$ is the expected sum of reward and entropy given state s, and $Q_\phi(s,a)$ is the expected sum of reward and entropy given state s and action a. Both functions assume the future is sampled using policy π. We first update ψ to minimize the squared residual error

$$J_V(\psi) = \mathbb{E}_{s_t \sim D}\left[\frac{1}{2}\left(V_\psi(s_t) - \mathbb{E}_{a_t \sim \pi_\theta}[Q_\phi(s_t, a_t) - \log \pi_\theta(a_t|s_t)]\right)^2\right]$$

where D refers to the distribution of states and actions in the replay buffer. We then update ϕ to minimize the soft Bellman residual $J_Q(\phi)$ as follows:

$$J_Q(\phi) = \mathbb{E}_{(s_t, a_t) \sim D}\left[\frac{1}{2}\left(Q_\phi(s_t, a_t) - \hat{Q}(s_t, a_t)\right)^2\right],$$

where $\hat{Q}(s_t, a_t) = r(s_t, a_t) + \gamma \mathbb{E}_{s_{t+1}}[V_{\bar{\psi}}(s_{t+1})]$.

Here, $\bar{\psi}$ denotes the exponential moving average of ψ, introduced to stabilize the learning process. Finally, we update θ to minimize the Kullback–Leibler divergence between the current policy and the target density defined by the Q function as follows:

$$\mathbb{E}_{s_t \sim D}\left[D_{\mathrm{KL}}\left(\pi_\theta\left(s_t\right) \| \frac{\exp(Q_\phi\left(s_t, a_t\right))}{Z_\phi(s_t)}\right)\right]$$

$Z_\phi(s_t)$ is a partition function that normalizes the distribution $\exp(Q_\phi\,(s_t, a_t))$. More details of the algorithm can be found in the original paper[56].

Neural network architecture and training details. We use a modified version of SAC algorithm that works only with π_θ and Q_ϕ (ref. 61). The value of α can be trained concurrently by setting the target entropy explicitly. We use a 1D CNN architecture to properly encode residing patterns from sensor signals across the time domain. In detail, $s_t \in \mathbb{R}^{32}$ is fed into the neural network π_θ, and then the two consecutive 1D CNN layers and a max pooling layer are applied in the time dimension. The 1D CNN layers employ 32 filters with width 5, and the pooling layer uses a window with width 2. The output tensor is flattened into a 1D vector and embedded into a 256-dimensional vector through a linear layer. Finally, the network outputs mean μ and variance σ^2 of the Gaussian distribution. We sample final actions from the Gaussian distribution. The action value function Q_ϕ basically has the same structure but with an additional concatenation layer added right before the final output layer. We embed the action into a 64-dimensional vector and concatenate it with the flattened output of the CNN layers. π_θ and Q_ϕ have separate network parameters. A nonlinear ReLU function[62] is applied to the outputs of all convolutional and linear layers. Other detailed hyperparameters are listed in Supplementary Table 2.

Equipping the sensors on the flapping-wing drone

The crack-based sensor was fabricated the same way as in previous studies[45,63]. After covering the shadow mask on a 7.5 µm polyimide film substrate (3022-5 Kapton thin film, Chemplexo), 50 nm chrome (Cr) and 20 nm gold (Au) thin films were deposited at a rate of 0.5 Å s^{-1} at 1.2 × 10^{-5} mTorr using a thermal evaporator (DDHT-SB015, Dae Dong Hitech). The deposited substrate was cut at 5 mm intervals and stretched 2% using a material tester (3342 UTM, Instron) to generate nanoscale cracks on the surface, as this facilitated a high gauge factor of ~30,000 at 2% strain. To measure the wind deformation, we equipped the fabricated crack-based sensors on the wings of a flapping drone (MetaFly, XTIM). The measured weight and wingspan of the drone are 10 g and 13.5 cm, respectively. The polydimethylsiloxane dry transfer method[64] was used to accurately attach the sensors to the thin wing base, which had a width of 1 mm, and a commercial strain gauge adhesive (CN adhesive, Tokyo Sokki Kenkyujo) was used for strong and conformal bonding. Conductive epoxy (CW2400, Chemtronics) was used for electrical connection between the sensor and wires, and commercial epoxy adhesive (EE-05, Axia) was applied to reduce the strain on the wiring interface. We used modified strip theory based on blade elemental analysis to calculate the lift and trust force using the wing strain (Supplementary Fig. 14).

Signal processing

We acquired the strain sensor signal using data acqusition equipment (DEWESoft SIRIUS). We used a bandpass filter over the range 0.1–55 Hz and normalized the signal into −1–1 to increase the RL agent learning efficiency. All data from the DAQ system and motion capture camera module (PrimeX 13, Optitrack) were transmitted to the desktop computer via serial communication and postprocessed through the Python language (v.3.7). For the signal-to-noise ratio experiment, we acquired the signal using LabVIEW equipment (Supplementary Fig. 2).

Wind prediction experiment

Simulating 186 wind cases. The flapping-wing drone was anchored to a three DOF motor system with three motors bound along the x, y and z axes. As shown in Fig. 4a, wind was generated by an electric fan and the drone rotated with various combinations of yaw, pitch and roll angles (Supplementary Figs. 7–12). With the system, 62 cases of wind directions and three wind speeds (3.5 and 7 m s^{-1}) were simulated, resulting in 186 cases of wind vectors (Fig. 4b). While the drone flapped its wings at 60%, 80% and 100% duty cycles, these variations did not affect the number of classification labels.

Neural network. For classifying the 186 wind cases, we used a neural network composed of two 1D convolutional layers followed by two fully connected layers. Each convolutional layer included five filters with size 50, yielding a total of 64 filters. The ReLU[62] activation function and a max pooling layer were applied to the output of each convolutional layer (Supplementary Fig. 6b). For the fully connected layers, dropout was applied to prevent overfitting. The final output was transformed to the shape of a probability distribution with the softmax function, and cross-entropy was used as the loss function. The model received 0.2 s of the sensor signal as input. For regression, there was just the difference in the number of nodes in fully connected layers and final output layers (Supplementary Fig. 6d).

Calculating prediction errors. We used the inverse trigonometric function of cosine to determine the angle by dividing the dot product by the product of the magnitudes of each vector (**a** is the prediction vector, and **b** is the ground vector):

$$\theta_{\mathrm{error}} = \cos^{-1}\left(\frac{a \bullet b}{|a|\,|b|}\right)(\mathbf{a} = (x_1 i, y_1 j, z_1 k)\,,\ \mathbf{b} = (x_2 i, y_2 j, z_2 k))$$

On the other hand, we represented the speed error as a percentage, which is calculated by dividing the difference between the predicted vector magnitude and the ground vector magnitude by the ground vector magnitude:

$$s_{\mathrm{error}} = \left(\left|\frac{|a| - |b|}{|b|}\right|\right) \times 100\,(\mathbf{a} = (x_1 i, y_1 j, z_1 k)\,, \mathbf{b} = (x_2 i, y_2 j, z_2 k))$$

Experimental setup

Control in a 1 DOF movement environment experiment. The drone was fixed at the end of a 20-cm-long stick, and the other end of the stick was fixed with a rotary joint (which was also an encoder). As a result, the drone moved in a circular motion when flapping its wings. The movement range of the drone was 0° to 350°. When the drone was not flapping its wings, a torsion spring allowed the drone to return to its original position. Constant wind in the −Y direction was generated by the two fans at speeds of 5 m s^{-1} (upward) and 3 m s^{-1} (downward). We used a Gaussian distribution function with 180 as the median for the reward function, and accordingly, the drone received the highest score when it maintained its position at 180°.

Control in a 2 DOF movement environment experiment. The drone was anchored at a pin joint so that the drone's pitch angle varied from −30° to 60°. The pitch angle of the drone was measured using Kinovea. A triangle-shaped rod was attached to the upper encoder, with 5 m s^{-1} of wind blew against the drone. In the system, the drag force of the wind pushed the drone back, resulting in the pitch angle turning counterclockwise, which allowed the drone to return to its original state. During training, a reward was given proportional with the rotational angle that increases when the drone moves forward.

Position control in a windy environment. The drone was tethered with a nylon cord to prevent collisions and ensure that it returned to its initial position after each learning episode. The drone had 2 DOF of control with two motors and flew with six DOF movement. To generate airflow in the wind tunnel, a large air circulator (Shinil Electronics) was placed at the outlet, and three small air circulators (Unimax) were placed at the inlet. An asymmetric and complex airflow was formed by controlling the output of the air circulators. To analyse the airflow inside the wind tunnel, smoke was generated behind the air circulator using a smoke machine, and the turbulence flow was visualized with the smoke converted into linear motion by passing through a honeycomb lattice (Supplementary Fig. 11). Due to the limited DOF control of the drone, the target position was given as x and y coordinates. We used a two-dimensional multivariate probability distribution function as the reward function and gave the target location as the mean.

Flight path control experiments. In a confined space measuring approximately 6 m in width, 8 m in depth and 4 m in height, we positioned a total of 12 motion capture cameras and five passive markers attached to the drone. To facilitate the free flight of drones in this space, a drone launcher was developed (Supplementary Fig. 12). This launcher comprises rail frames, wheels and a motor ensuring consistent start conditions such as take-off velocity. At the beginning of each episode, the drone advances at a speed of 1 m s^{-1} until it is halted by a mechanical brake. Leveraging the law of the inertia, the mounted drone takes off from the stage, flapping its wings. During training, the drone was picked up and remounted to the launcher after each episode, and curtains and nets were hung around the walls to prevent collisions and potential damage to the drone. For actuation and data acquisition from the drone, a total of eight copper wires (American Wire Gauge 40; diameter, 96.5 μm; length, 8 m) were utilized. Specifically, two wires were allocated for the thrusting motor, two for the directional motor and four for the two strain sensors attached to the drone's wings. As depicted in Fig. 6, the actual dimensions of the drone's flying space are 4 m × 6 m × 2 m (width × length × height). However, because the tethers are centrally attached to the floor, the operational space is effectively reduced to 2 m × 3 m × 2 m (width × length × height), which is the area where the drone and tethers interact. This configuration minimizes the tether length, consequently reducing the tensile force exerted on the drone and preventing loss of thrust, thereby enhancing flight efficiency. With this arrangement, the maximum length of each tether is approximately 4.12 m. Given that the mass of the tether is 40 mg m^{-1}, and with eight wires connected, the total additional weight from the tethers amounts to 1.32 g, which is only 13.2% of the drone's total weight. This confirms that the mass of the tethers is minimal and does not substantially impact the drone's performance.

The reward function is designed based on each trajectory of the drone. For leftward flight, the reward is proportionally given according to the displacement in the negative direction of the x axis and the positive direction of the y axis. This means the drone receives a larger reward when moving to the left-front direction (Extended Data Fig. 7a). Conversely, in the case of rightward flight, the reward is proportional to the positive displacement along the x axis and the positive direction of the y axis (Extended Data Fig. 7b). For a mixed trajectory involving both leftward and rightward flight, such as the zigzag trajectory, the reward is adjusted based on the displacement along the y axis (Extended Data Fig. 7c). Additionally, we increased the input size as the RL agent needed to consider more earlier standpoints. The reward is initially proportional to the displacement in the negative direction of the x axis and changes in the opposite direction when the drone passes a certain point along the y axis. We focused solely on the displacement in the x axis to induce a dramatic directional change in the drone. Due to inertia, a substantial force is required to alter the drone's direction, making this strategy suitable for achieving our objective.

Next, we trained the RL agent to control the drone to fly in a circular motion by adjusting the reward function (Extended Data Fig. 7d). The reward is determined proportionally by the dot product of two vectors extracted from sequential positions. Considering the displacement of the drone from its viewpoint, global vectors are rotated according to the local coordinate system of the drone itself. Because our experiment was conducted only in the real world and not in simulation, we fixed the thrust power of the drone to 80% of the duty cycle to reduce the time required for the RL agent to learn. In this environment, the RL agent is tasked with controlling the angle of the directional motor in the drone's tail.

We also trained the RL agent to control the altitude of the drone, allowing it to fly at a higher or lower altitude than the starting point of the drone launcher. The reward function designates that the drone receives a higher reward when flying further away from the starting point while maintaining the target altitudes of 1.6 and 0.8 m, respectively. The RL agent had to manage both thrust motor power and directional motor power to uphold the desired altitude in this environment.

Reporting summary

Further information on research design is available in the Nature Portfolio Reporting Summary linked to this article.

Data availability

Datasets used in machine learning are available in public repositories. Training and test datasets of wind direction prediction are available via figshare at https://doi.org/10.6084/m9.figshare.26185967 (ref. 65). Datasets of wing strain information with position data for path trajectory prediction are available via figshare at https://doi.org/10.6084/m9.figshare.26190473 (ref. 66). Source data are provided with this paper.

Code availability

Codes for the RL used in this study are available via Zenodo at https://doi.org/10.5281/zenodo.12683619 (ref. 67).

References

1. Ansari, S., Żbikowski, R. & Knowles, K. Aerodynamic modelling of insect-like flapping flight for micro air vehicles. *Prog. Aerosp. Sci.* **42**, 129–172 (2006).
2. Ma, K. Y., Chirarattananon, P., Fuller, S. B. & Wood, R. J. Controlled flight of a biologically inspired, insect-scale robot. *Science* **340**, 603–607 (2013).
3. Chen, Y. et al. Controlled flight of a microrobot powered by soft artificial muscles. *Nature* **575**, 324–329 (2019).
4. Jafferis, N. T., Helbling, E. F., Karpelson, M. & Wood, R. J. Untethered flight of an insect-sized flapping-wing microscale aerial vehicle. *Nature* **570**, 491–495 (2019).
5. De Croon, G., De Clercq, K., Ruijsink, R., Remes, B. & De Wagter, C. Design, aerodynamics, and vision-based control of the DelFly. *Int. J. Micro Air Veh.* **1**, 71–97 (2009).
6. Phan, H. V., Kang, T. & Park, H. C. Design and stable flight of a 21 g insect-like tailless flapping wing micro air vehicle with angular rates feedback control. *Bioinspir. Biomim.* **12**, 036006 (2017).
7. Zhang, J., Fei, F., Tu, Z. & Deng, X. Design optimization and system integration of robotic hummingbird. In *Proc. IEEE International Conference on Robotics and Automation* 5422–5428 (IEEE, 2017).
8. Mellinger, D. & Kumar, V. Minimum snap trajectory generation and control for quadrotors. In *Proc. IEEE International Conference on Robotics and Automation* 2520–2525 (IEEE, 2011).
9. Elijah, T., Jamisola, R. S., Tjiparuro, Z. & Namoshe, M. A review on control and maneuvering of cooperative fixed-wing drones. *Int. J. Dyn. Control* **9**, 1332–1349 (2021).
10. de Croon, G. Flapping-wing drones show off their skills. *Sci. Robot.* **5**, eabd0233 (2020).
11. Chirarattananon, P. et al. Dynamics and flight control of a flapping-wing robotic insect in the presence of wind gusts. *Interface Focus* **7**, 20160080 (2017).

12. Baek, S. S., Bermudez, F. L. G. & Fearing, R. S. Flight control for target seeking by 13 gram ornithopter. In *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems* 2674–2681 (IEEE, 2011).
13. Nguyen, Q.-V. & Chan, W. L. Development and flight performance of a biologically-inspired tailless flapping-wing micro air vehicle with wing stroke plane modulation. *Bioinspir. Biomim.* **14**, 016015 (2018).
14. Shin, H. S. et al. Bio-inspired large-area soft sensing skins to measure UAV wing deformation in flight. *Adv. Funct. Mater.* **31**, 2100679 (2021).
15. Tong, S., Weiping, Z., Jiawang, M. & Zihao, C. Research progress on control of bioinspired flapping-wing micro air vehicles. In *Proc. IEEE International Conference on Unmanned Systems* 842–847 (IEEE, 2019).
16. Woolf, P. J. *Chemical Process Dynamics and Controls* (Univ. Michigan Engineering Controls Group, 2009).
17. Walker, S. M., Thomas, A. L. & Taylor, G. K. Deformable wing kinematics in free-flying hoverflies. *J. R. Soc. Interface* **7**, 131–142 (2010).
18. Somps, C. & Luttges, M. Dragonfly flight: novel uses of unsteady separated flows. *Science* **228**, 1326–1329 (1985).
19. Wang, H., Zeng, L., Liu, H. & Yin, C. Measuring wing kinematics, flight trajectory and body attitude during forward flight and turning maneuvers in dragonflies. *J. Exp. Biol.* **206**, 745–757 (2003).
20. Springthorpe, D., Fernández, M. J. & Hedrick, T. L. Neuromuscular control of free-flight yaw turns in the hawkmoth Manduca sexta. *J. Exp. Biol.* **215**, 1766–1774 (2012).
21. Salami, E., Ward, T. A., Montazer, E. & Ghazali, N. N. N. A review of aerodynamic studies on dragonfly flight. *Proc. Inst. Mech. Eng. C* **233**, 6519–6537 (2019).
22. Mischiati, M. et al. Internal models direct dragonfly interception steering. *Nature* **517**, 333–338 (2015).
23. Hedlund, J. S. et al. Unraveling the world's longest non-stop migration: the Indian Ocean crossing of the globe skimmer dragonfly. *Front. Ecol. Evol.* **9**, 525 (2021).
24. Gao, B. et al. Adaptive strategies of high-flying migratory hoverflies in response to wind currents. *Proc. R. Soc. B* **287**, 20200406 (2020).
25. Cardé, R. T. Animal migration: seasonal reversals of migrant moths. *Curr. Biol.* **18**, R1007–R1009 (2008).
26. Menz, M. H. et al. Individual tracking reveals long-distance flight-path control in a nocturnally migrating moth. *Science* **377**, 764–768 (2022).
27. Hu, G. et al. Mass seasonal bioflows of high-flying insect migrants. *Science* **354**, 1584–1587 (2016).
28. Frye, M. A. & Dickinson, M. H. Motor output reflects the linear superposition of visual and olfactory inputs in Drosophila. *J. Exp. Biol.* **207**, 123–131 (2004).
29. Sherman, A. & Dickinson, M. H. Summation of visual and mechanosensory feedback in Drosophila flight control. *J. Exp. Biol.* **207**, 133–142 (2004).
30. Tuthill, J. C. & Wilson, R. I. Mechanosensation and adaptive motor control in insects. *Curr. Biol.* **26**, R1022–R1038 (2016).
31. Dickerson, B. H., Fox, J. L. & Sponberg, S. Functional diversity from generic encoding in insect campaniform sensilla. *Curr. Opin. Physiol.* **19**, 194–203 (2021).
32. Taylor, G. K. & Krapp, H. G. Sensory systems and flight stability: what do insects measure and why? *Adv. in Insect Phys.* **34**, 231–316 (2007).
33. Combes, S. & Daniel, T. Flexural stiffness in insect wings II. Spatial distribution and dynamic wing bending. *J. Exp. Biol.* **206**, 2989–2997 (2003).
34. Bomphrey, R. J. & Godoy-Diana, R. Insect and insect-inspired aerodynamics: unsteadiness, structural mechanics and flight control. *Curr. Opin. Insect. Sci.* **30**, 26–32 (2018).
35. Dickerson, B. H., Aldworth, Z. N. & Daniel, T. L. Control of moth flight posture is mediated by wing mechanosensory feedback. *J. Exp. Biol.* **217**, 2301–2308 (2014).
36. Dickinson, M. H. Comparison of encoding properties of campaniform sensilla on the fly wing. *J. Exp. Biol.* **151**, 245–261 (1990).
37. Pratt, B., Deora, T., Mohren, T. & Daniel, T. Neural evidence supports a dual sensory-motor role for insect wings. *Proc. R. Soc. B* **284**, 20170969 (2017).
38. Fabian, J. et al. Systematic characterization of wing mechanosensors that monitor airflow and wing deformations. *iScience* **25**, 104150 (2022).
39. Hinson, B. T. & Morgansen, K. A. Gyroscopic sensing in the wings of the hawkmoth Manduca sexta: the role of sensor location and directional sensitivity. *Bioinsp. Biomim.* **10**, 056013 (2015).
40. Taylor, G. in *Flow Phenomena in Nature*, Vol. 1 (ed. Liebe, R.) 155–166 (WIT, 2007).
41. Sane, S. P. The aerodynamics of insect flight. *J. Exp. Biol.* **206**, 4191–4208 (2003).
42. Chin, D. D. & Lentink, D. Flapping wing aerodynamics: from insects to vertebrates. *J. Exp. Biol.* **219**, 920–932 (2016).
43. Breuer, K. Flight of the RoboBee. *Nature News & Views* https://www.nature.com/articles/d41586-019-01964-3 (2019).
44. Aiello, B. R. et al. Spatial distribution of campaniform sensilla mechanosensors on wings: form, function, and phylogeny. *Curr. Opin. Insect Sci.* **48**, 8–17 (2021).
45. Kang, D. et al. Ultrasensitive mechanical crack-based sensor inspired by the spider sensory system. *Nature* **516**, 222–226 (2014).
46. Lee, E. et al. Effect of metal thickness on the sensitivity of crack-based sensors. *Sensors* **18**, 2872 (2018).
47. Kim, T. et al. Polyimide encapsulation of spider-inspired crack-based sensors for durability improvement. *Appl. Sci.* **8**, 367 (2018).
48. Yang, R., Zhang, W., Mou, J., Zhang, B. & Zhang, Y. Attitude estimation algorithm of flapping-wing micro air vehicle based on extended Kalman filter. In *Proc. International Conference on Autonomous Unmanned Systems* 1432–1443 (Springer, 2022).
49. Bloesch, M. et al. Fusion of optical flow and inertial measurements for robust egomotion estimation. In *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems* 3102–3107 (IEEE, 2014).
50. Tagliabue, A. & How, J. P. Airflow-inertial odometry for resilient state estimation on multirotors. In *Proc. IEEE International Conference on Robotics and Automation* 5736–5743 (IEEE, 2021).
51. Tagliabue, A. et al. Touch the wind: simultaneous airflow, drag and interaction sensing on a multirotor. In *Proc. IEEE*/RSJ *International Conference on Intelligent Robots and Systems (IROS)* 1645–1652 (IEEE, 2020).
52. Kim, T. et al. Ultra-stable and tough bioinspired crack-based tactile sensor for small legged robots. *npj Flex. Electron.* **7**, 22 (2023).
53. Pan, E., Liang, X. & Xu, W. Development of vision stabilizing system for a large-scale flapping-wing robotic bird. *IEEE Sens. J.* **20**, 8017–8028 (2020).
54. Shyy, W. et al. Recent progress in flapping wing aerodynamics and aeroelasticity. *Prog. Aerosp. Sci.* **46**, 284–327 (2010).
55. Schwab, R., Reade, J. & Jankauski, M. Quasi three-dimensional deformable blade element and unsteady vortex lattice reduced-order modeling of fluid–structure interaction in flapping wings. *Phys. Fluids* **34**, 121903 (2022).
56. Haarnoja, T., Zhou, A., Abbeel, P. & Levine, S. Soft actor-critic: off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *Proc. 35th International Conference on Machine Learning* (eds Dy, J. & Krause, A.) 1861–1870 (PMLR, 2018).
57. Mnih, V. et al. Asynchronous methods for deep reinforcement learning. In *Proc. 33rd International Conference on Machine Learning* (eds Balcan, M. F. & Weinberger, K. Q.) 1928–1937 (PMLR, 2016).

58. Espeholt, L. et al. IMPALA: scalable distributed deep-RL with importance weighted actor-learner architectures. In *Proc. 35th International Conference on Machine Learning* (eds Dy, J. & Krause, A.) 1407–1416 (PMLR, 2018).
59. Schulman, J., Wolski, F., Dhariwal, P., Radford, A. & Klimov, O. Proximal policy optimization algorithms. Preprint at https://arXiv.org/1707.06347 (2017).
60. Ziebart, B. D. *Modeling Purposeful Adaptive Behavior with the Principle of Maximum Causal Entropy* (Carnegie Mellon Univ., 2010).
61. Haarnoja, T. et al. Soft actor-critic algorithms and applications. Preprint at https://arXiv.org/1812.05905 (2018).
62. Agarap, A. F. Deep learning using rectified linear units (relu). Preprint at https://arXiv.org/1803.08375 (2018).
63. Roh, Y. et al. Vital signal sensing and manipulation of a microscale organ with a multifunctional soft gripper. *Sci. Robot.* **6**, eabi6774 (2021).
64. Sundar, V. C. et al. Elastomeric transistor stamps: reversible probing of charge transport in organic crystals. *Science* **303**, 1644–1646 (2004).
65. Kim, T., Hong, I., Im, S. & Rho, S. Datasets of wing strain sensor and surrounding wind vectors. *figshare* https://doi.org/10.6084/m9.figshare.26185967 (2024).
66. Kim, T., Hong, I., Im, S. & Rho, S. Datasets of wing strain and position data for path trajectory prediction. *figshare* https://doi.org/10.6084/m9.figshare.26190473 (2024).
67. Rho, S., Im, S. & Kim, T. Fly-by-Feel: wing strain-based flight control of flapping-wing drones through reinforcement learning. *Zenodo* https://doi.org/10.5281/zenodo.12683619 (2024).

Acknowledgements

S.I. thanks M. Karásek for providing detailed information about the prior research on the flapping drone (Delfly). S.I. also thanks A.M. Yarger for advice on current research trends regarding campaniform sensilla on the dragonfly's wing and for sharing related prior research. D.K., S.H. and J.-S.K. acknowledge financial support from the Ajou University research fund. This work was supported by National Research Foundation of Korea grants funded by the Korea government (grant nos. 2021R1A6A3A01087289, 2021R1C1C1011872, 2022R1A2C2093100, RS-2023-00277110, RS-2024-00411660) and Korea Environment Industry & Technology Institute through the Digital Infrastructure Building Project for Monitoring, Surveying and Evaluating the Environmental Health Program, funded by the Korea Ministry of Environment (grant no. 2021003330009).

Author contributions

T.K., I.H., S.I., S.R., J.-S.K., S.H. and D.K. led the development of the concepts, designed the experiments, interpreted the results and wrote the paper. T.K., I.H. and S.I. designed the experimental setups, and S.R., T.K. and S.I. developed codes for the RL algorithm system. I.H. and M.K. provided strain sensors on the flapping drone with support from Y.R., C.K., J.P., D. Lim, D. Lee and S.L. with fabrication of the sensors. Jingoo Lee designed the embedded circuit system, and I.B. built the motion capture tracking system of the flapping drone. J.C. and Y.-M.C. built the free flight experiments, and S.I. analysed the odometry of the drone in free flight with support from M.R.H. and S.K. In addition, Junho Lee checked the RL results, and S.S. contributed to the progress of the experiments. U.K. provided the kinematics ideas for implementing the flapping motion of the drone.

Competing interests

The authors declare no competing interests.

Additional information

Extended data is available for this paper at https://doi.org/10.1038/s42256-024-00893-9.

Supplementary information The online version contains supplementary material available at https://doi.org/10.1038/s42256-024-00893-9.

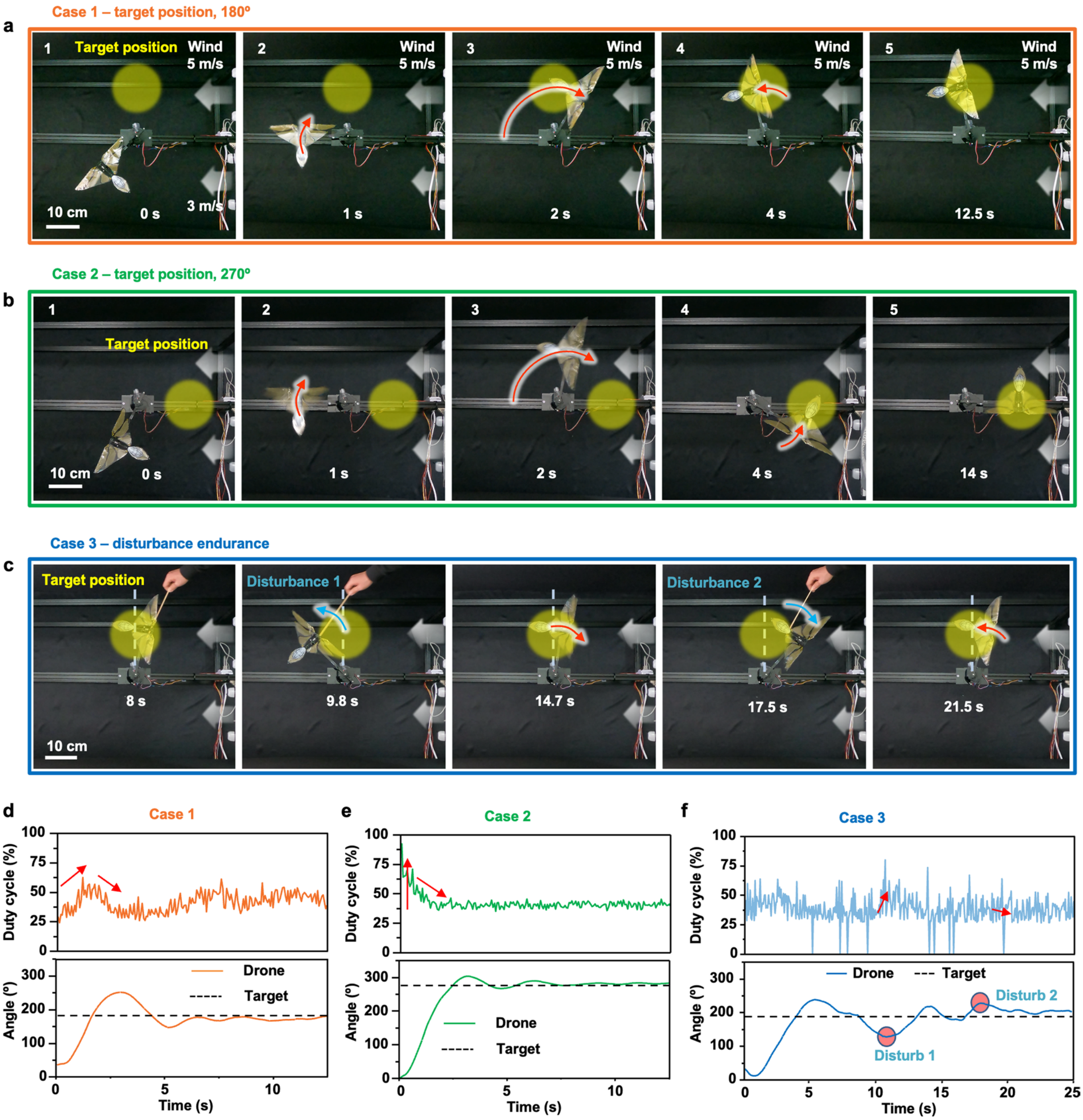
Correspondence and requests for materials should be addressed to Je-sung Koh, Seungyong Han or Daeshik Kang.

Peer review information *Nature Machine Intelligence* thanks the anonymous reviewers for their contribution to the peer review of this work.

Reprints and permissions information is available at www.nature.com/reprints.

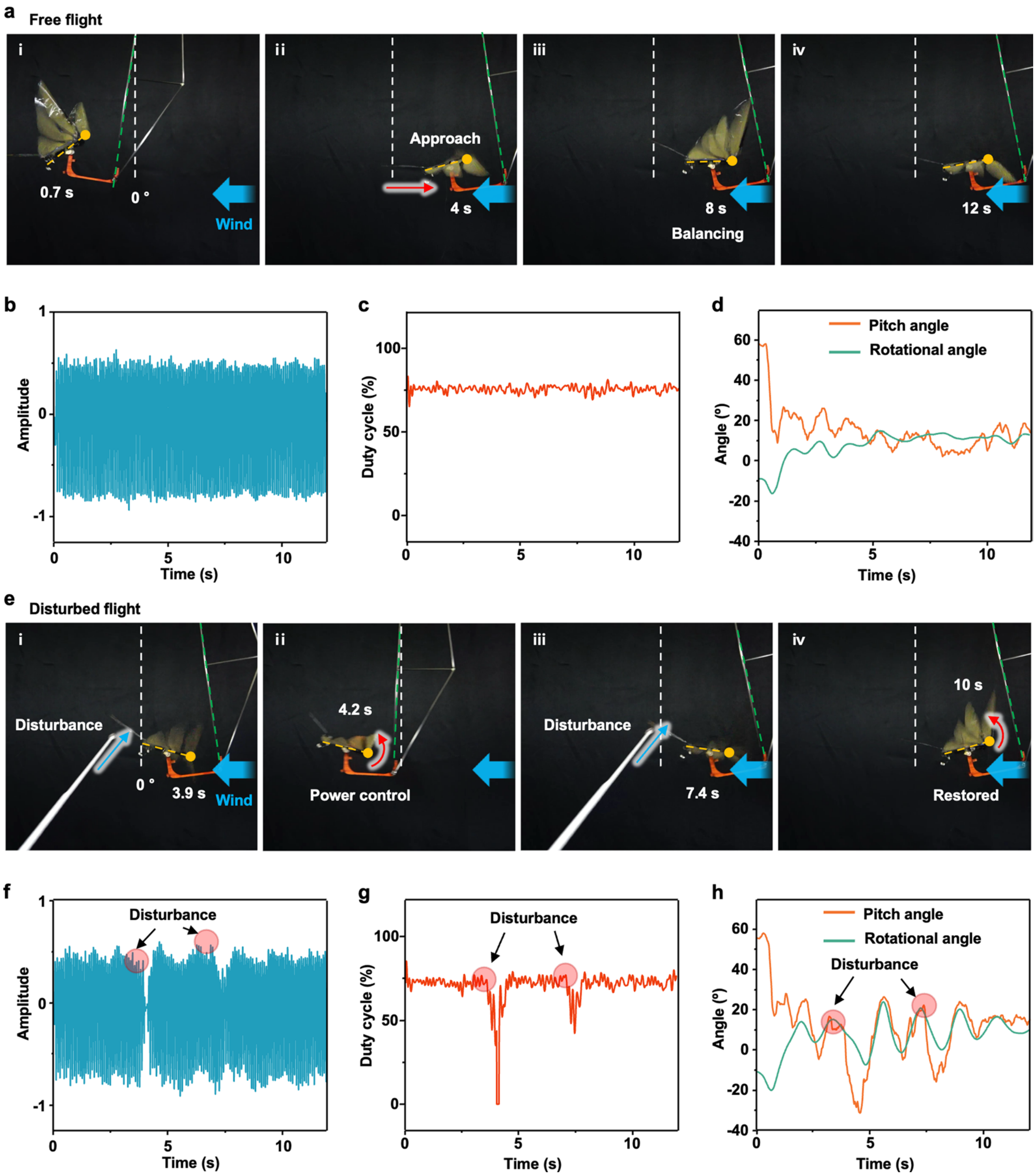
Taewi Kim[1,5], Insic Hong[1,5], Sunghoon Im[1,5], Seungeun Rho[2,5], Minho Kim[1], Yeonwook Roh[1], Changhwan Kim[1], Jieun Park[1], Daseul Lim[1], Doohoe Lee[1], Seunggon Lee[1], Jingoo Lee[1], Inryeol Back[1], Junggwang Cho[1], Myung Rae Hong[1], Sanghun Kang[1], Joonho Lee[3], Sungchul Seo[4], Uikyum Kim[1], Young-Man Choi[1], Je-sung Koh[1] ✉, Seungyong Han[1] ✉ & Daeshik Kang[1] ✉

[1]Department of Mechanical Engineering, Ajou University, Suwon, Korea. [2]Kakaobrain, Seongnam, Korea. [3]Robotic Systems Lab, ETH-Zürich, Zürich, Switzerland. [4]Department of Nanochemical, Biological and Environmental Engineering, Seokyeong University, Seoul, Korea. [5]These authors contributed equally: Taewi Kim, Insic Hong, Sunghoon Im, Seungeun Rho. ✉e-mail: jskoh@ajou.ac.kr; sy84han@ajou.ac.kr; dskang@ajou.ac.kr

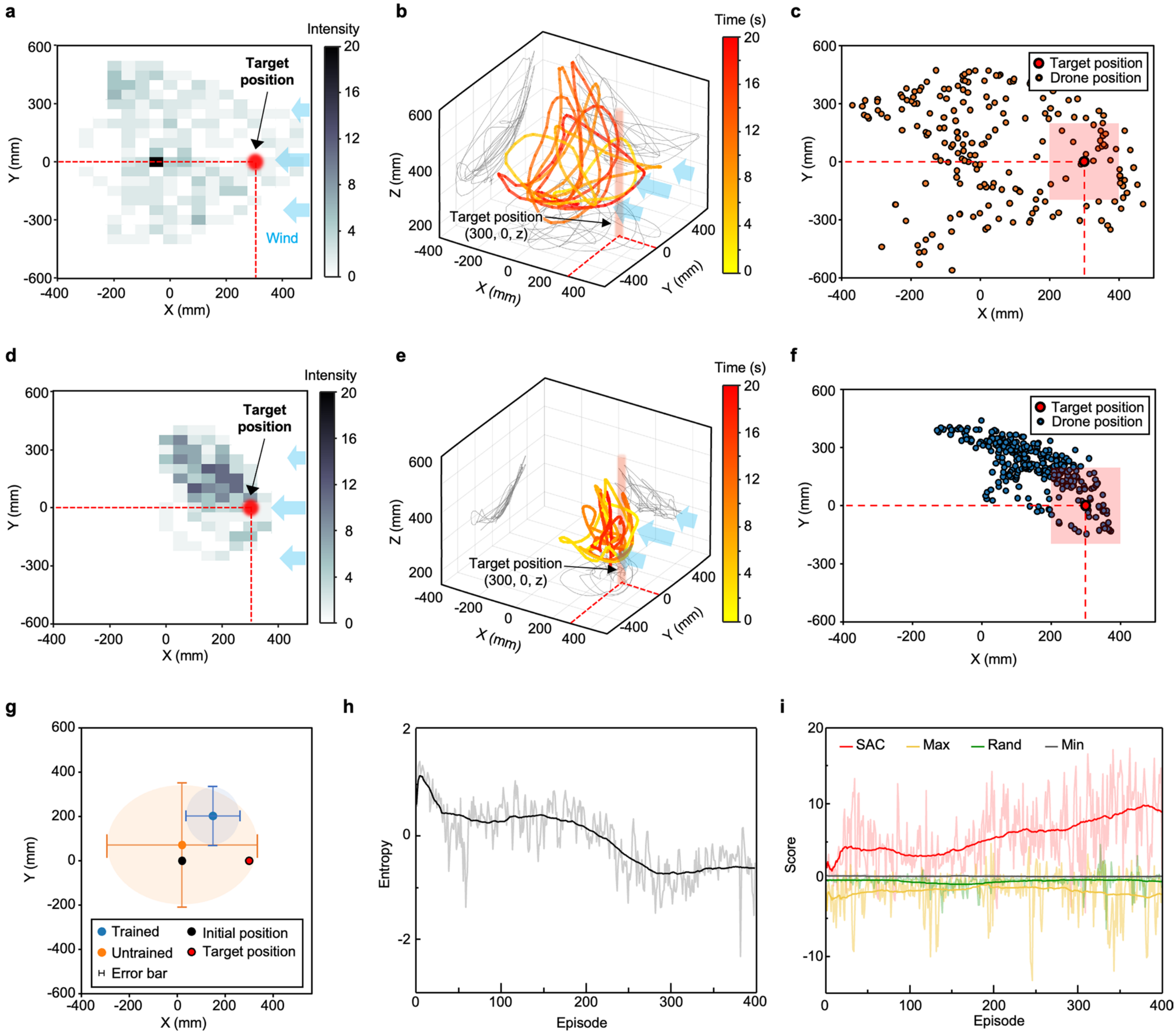


Extended Data Fig. 1 | Additional examples of the 1 DOF system. **a**, Snapshots of the episode when the target angle is 180°. It moves quickly to reach target position and reduces its motor power when it passes over the target. **b**, The case when the target position angle is 270°. When the target position is changed, the drone could recognize the position based on the sensor information. **c**, The case when the researcher disturbs the flight by pushing the drone backward and forward. Even if there is an intentional disturbance, a trained drone controls its motor power to maintains the target position. **d**, Motor power control and corresponding angle change for case 1. **e**, Motor power control and corresponding angle change for case 2. **f**, Motor power control and corresponding angle change for case 3.

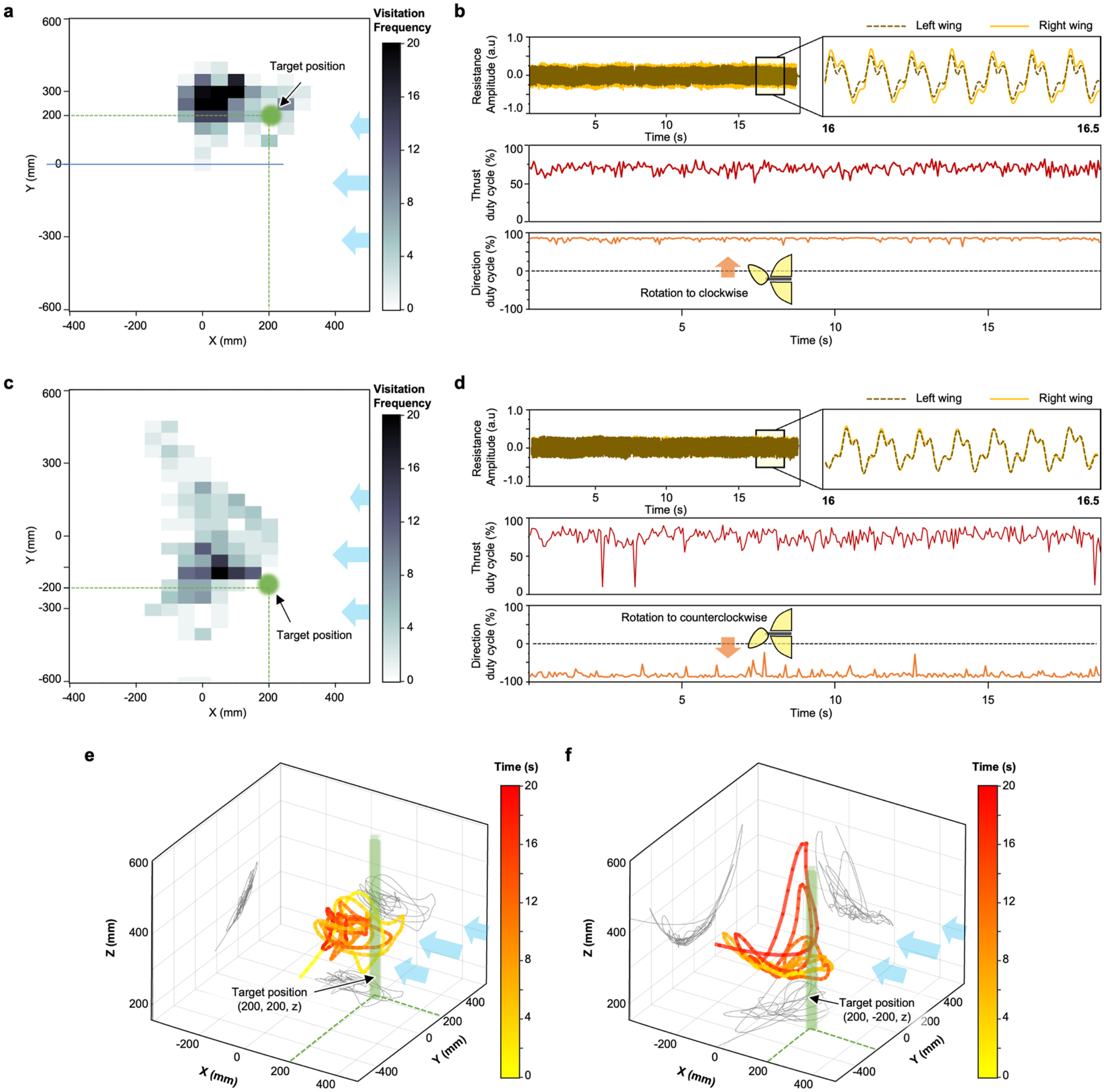
https://doi.org/10.1038/s42256-024-00893-9



Extended Data Fig. 2 | Extended episode for optimal flight learning. **a**, Snapshots of free flight in the 2 DOF system. The trained drone balanced to maintain proper pitch angle during flying. **b**, Signal amplitude of the crack-based sensor attached on the left wing's base. **c**, Change in the drone's motor power. The drone fine-tunes the power to sustain the optimal pitch angle (β). **d**, Changes in the rotational angle (α) and pitch angle (β). During flight, drone maintains optimal pitch angle and so that could get the highest rotation angle (approximately 10°). **e**, Snapshots of a disturbed flight in the 2 DOF system. Even if there is an intentional disturbance, a trained drone reduces its motor power to avoid falling state. **f**, Variation in the signal amplitude, synchronized with (e). The perturbation events are marked with red circles. **g**, Power of the motor during the disturbed flight. After the perturbation events, the drone immediately reduces its power to maintain balance. **h**, Rotational angle (α) and pitch angle (β) during the disturbed flight. The drone recovers the optimal angles quickly after the perturbation events.

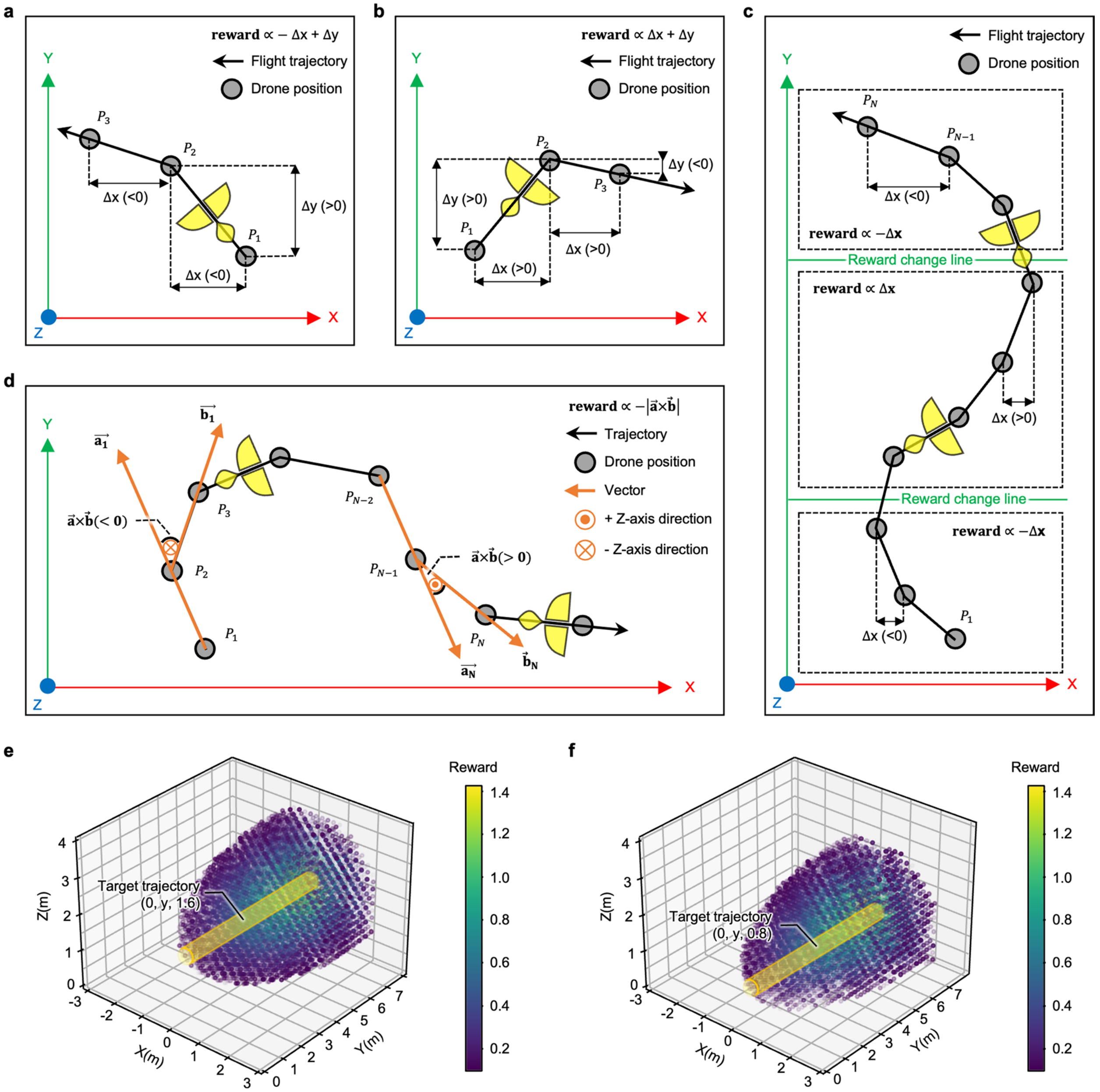


Extended Data Fig. 3 | Comparison of an untrained and trained drone. **a** and **b**, Plane and space trajectories of the untrained drone heading toward middle target position. **c**, Frequency distribution of untrained drone within a radius of 20 ~ 30 cm from the target point. **d** and **e**, Plane and space trajectories of the trained drone heading toward middle target position. **f**, Frequency distribution of trained drone within a radius of 20 ~ 30 cm from the target point. **g**, Comparison of mean value of the position from untrained and trained drone. **h** and **i**, Changes in the entropy and score of the artificial neural network with the number of training epochs.

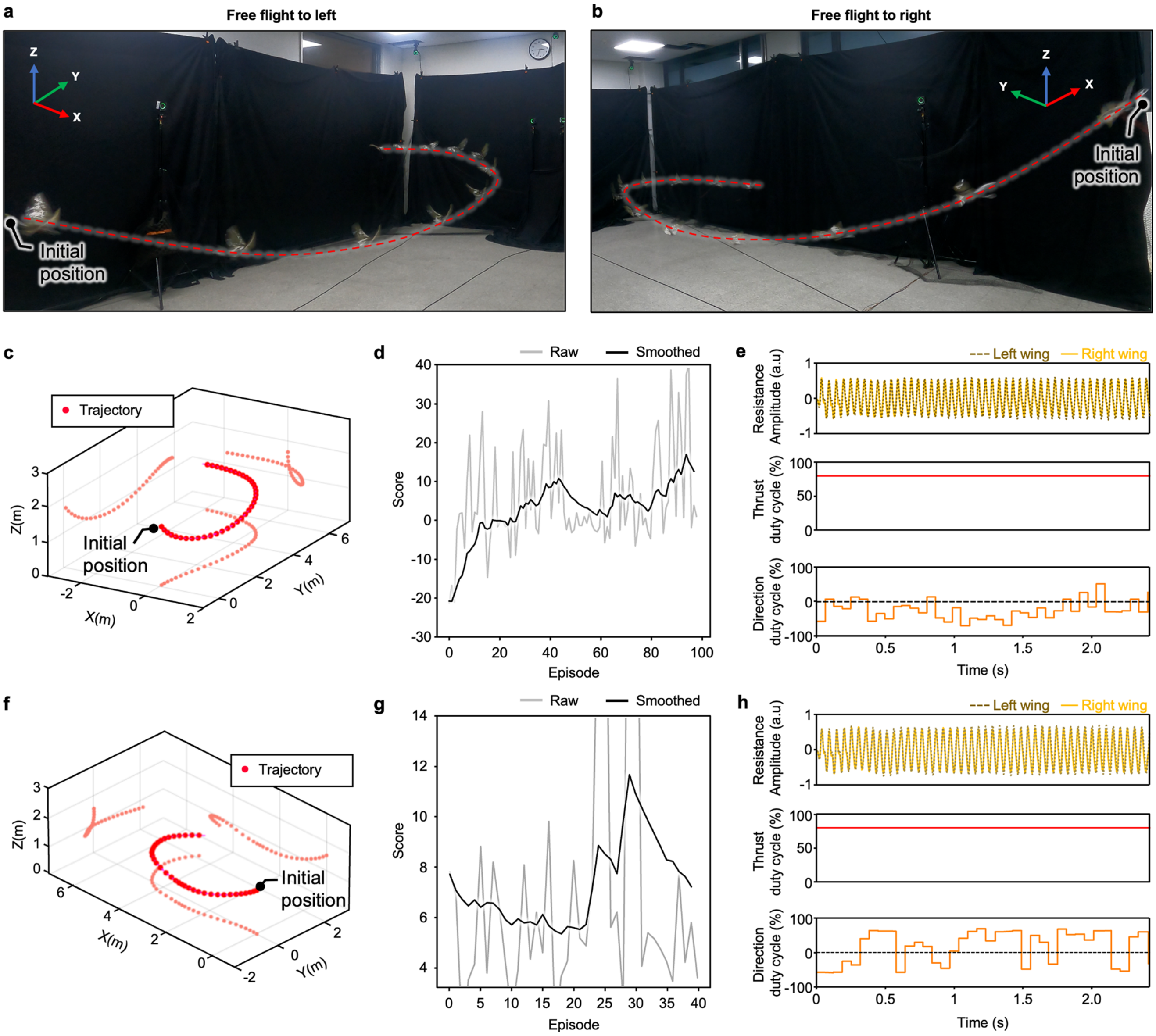


Extended Data Fig. 4 | Position control of a flapping-wing drone when trained to reach different target positions. a and **b**, Trajectory and change in the wing deformation and the output of motors of a trained drone heading toward the left target position. **c** and **d**, Trajectory and change in the wing deformation and the output of motors of a trained drone heading toward the right target position 3 (fast airflow). **e** and **f**, 3D trajectories of the trained drone heading toward target position (e: left target position, f: right target position).

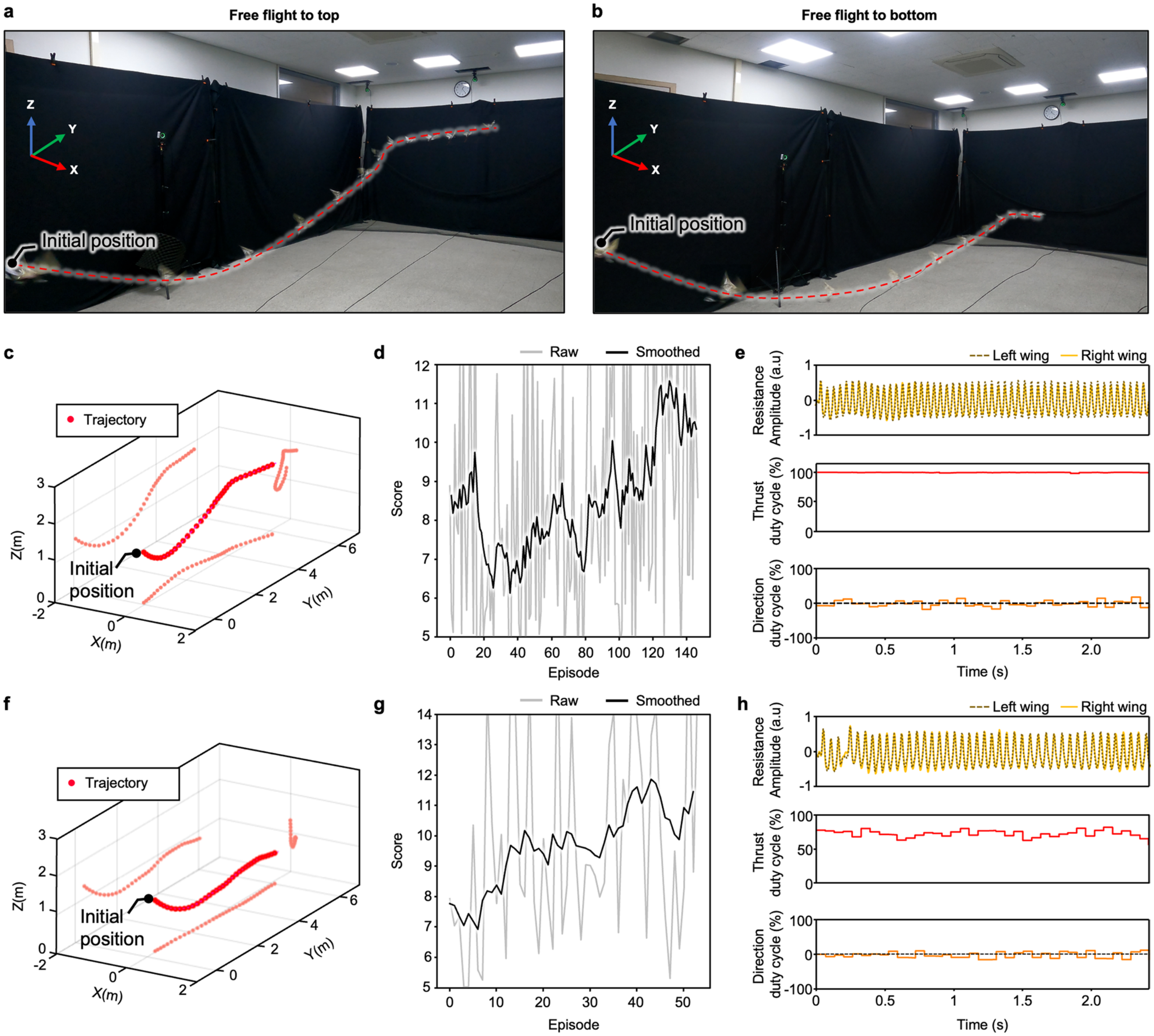
https://doi.org/10.1038/s42256-024-00893-9



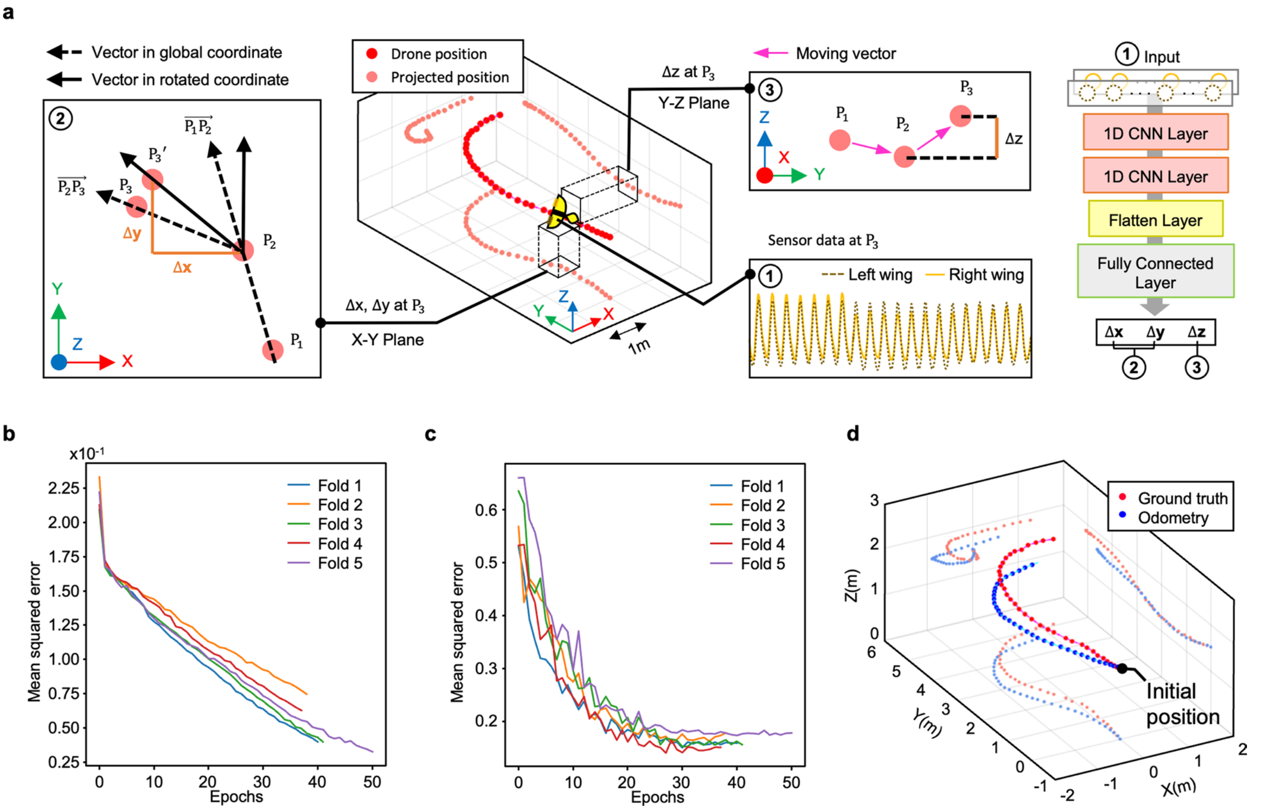
Extended Data Fig. 5 | Reward functions for flight trajectory control experiments. a, Reward function designated for leftward flight. **b**, Reward function designated for rightward flight. **c**, Reward function designated for zigzag trajectory of flight. **d**, Reward function designated for circular motion flight. **e**, Reward function designated for higher altitude trajectory of flight. **f**, Reward function designated for lower altitude trajectory of flight.



Extended Data Fig. 6 | Results of left and rightward flight control. **a**, Sequential snapshots from a video depicting the drone's leftward flight. **b**, Sequential snapshots from a video displaying drone's rightward flight. **c**, Trajectory of the drone during the leftward flight. **d**, Scores observed as episode progressed during the training of leftward flight. **e**, Strain sensor signals during leftward flight and corresponding duty cycle of thrust motor power and directional motor power. **f**, Trajectory of the drone during rightward flight. **g**, Scores recorded as the episode progressed during the training of rightward flight. **h**, Strain sensor signals corresponding to the rightward flight, along with the associated duty cycle of thrust motor power and directional motor power.



Extended Data Fig. 7 | Results of higher and lower altitude flight control. **a**, Sequential snapshots from a video illustrating flight at higher altitudes. **b**, Sequential snapshots from a video displaying drone's flight at lower altitudes. **c**, Trajectory of the drone during the higher altitude flight. **d**, Scores recorded as episode progressed during the training of higher altitude flight. **e**, Strain sensor signals during the higher altitude flight, along with the corresponding duty cycle of thrust motor power and directional motor power. **f**, Trajectory of the drone during lower altitude flight. **g**, Scores recorded as the episodes progressed during the training of the lower height intended flight. **h**, Strain sensor signals corresponding to the lower height flight, along with the associated duty cycle of thrust motor power and directional motor power.



Extended Data Fig. 8 | Methods of yielding odometry and results. **a**, Labelling procedure for comparing odometry and ground truth trajectories. **b**, Comparison of mean squared error between 5 folds of different training datasets. **c**, Comparison of mean squared error between 5 folds of different validation datasets. **d**, Representative result of comparison between odometry and ground truth trajectories from additional experiment.

nature portfolio

Corresponding author(s): Prof. Daeshik Kang

Last updated by author(s): Jul 12, 2024

Reporting Summary

Nature Portfolio wishes to improve the reproducibility of the work that we publish. This form provides structure for consistency and transparency in reporting. For further information on Nature Portfolio policies, see our Editorial Policies and the Editorial Policy Checklist.

Statistics

For all statistical analyses, confirm that the following items are present in the figure legend, table legend, main text, or Methods section.

n/a	Confirmed	
☐	☒	The exact sample size (*n*) for each experimental group/condition, given as a discrete number and unit of measurement
☐	☒	A statement on whether measurements were taken from distinct samples or whether the same sample was measured repeatedly
☐	☒	The statistical test(s) used AND whether they are one- or two-sided *Only common tests should be described solely by name; describe more complex techniques in the Methods section.*
☒	☐	A description of all covariates tested
☐	☒	A description of any assumptions or corrections, such as tests of normality and adjustment for multiple comparisons
☒	☐	A full description of the statistical parameters including central tendency (e.g. means) or other basic estimates (e.g. regression coefficient) AND variation (e.g. standard deviation) or associated estimates of uncertainty (e.g. confidence intervals)
☒	☐	For null hypothesis testing, the test statistic (e.g. *F*, *t*, *r*) with confidence intervals, effect sizes, degrees of freedom and *P* value noted *Give P values as exact values whenever suitable.*
☒	☐	For Bayesian analysis, information on the choice of priors and Markov chain Monte Carlo settings
☒	☐	For hierarchical and complex designs, identification of the appropriate level for tests and full reporting of outcomes
☒	☐	Estimates of effect sizes (e.g. Cohen's *d*, Pearson's *r*), indicating how they were calculated

Our web collection on statistics for biologists contains articles on many of the points above.

Software and code

Policy information about availability of computer code

Data collection	DEWESoft SIRIUS, PrimeX 13, Labview, 3342 UTM
Data analysis	MATLAB R2020a, Python(version 3.7), Kinovea

For manuscripts utilizing custom algorithms or software that are central to the research but not yet described in published literature, software must be made available to editors and reviewers. We strongly encourage code deposition in a community repository (e.g. GitHub). See the Nature Portfolio guidelines for submitting code & software for further information.

Data

Policy information about availability of data

All manuscripts must include a data availability statement. This statement should provide the following information, where applicable:

- Accession codes, unique identifiers, or web links for publicly available datasets
- A description of any restrictions on data availability
- For clinical datasets or third party data, please ensure that the statement adheres to our policy

Datasets used in machine learning are available in public respiratory. Training and Test datasets of wind direction prediction is available at http://doi.org/10.6084/m9.figshare.26185967. Datasets of wing strain information with position data for path trajectory prediction is available at http://doi.org/10.6084/m9.figshare.26190473.

Research involving human participants, their data, or biological material

Policy information about studies with human participants or human data. See also policy information about sex, gender (identity/presentation), and sexual orientation and race, ethnicity and racism.

Reporting on sex and gender	No human participants or human data.
Reporting on race, ethnicity, or other socially relevant groupings	No human participants or human data.
Population characteristics	No human participants or human data.
Recruitment	No human participants or human data.
Ethics oversight	No human participants or human data.

Note that full information on the approval of the study protocol must also be provided in the manuscript.

Field-specific reporting

Please select the one below that is the best fit for your research. If you are not sure, read the appropriate sections before making your selection.

☒ Life sciences ☐ Behavioural & social sciences ☐ Ecological, evolutionary & environmental sciences

For a reference copy of the document with all sections, see nature.com/documents/nr-reporting-summary-flat.pdf

Life sciences study design

All studies must disclose on these points even when the disclosure is negative.

Sample size	We acquired data for machine learning and reinforcement learning, and determined the sample size based on the validation accuracy.
Data exclusions	There is no data excluded.
Replication	We confirmed the reproducibility of our study.
Randomization	We randomized training datasets in machine learning, and randomized actions in reinforcement learning.
Blinding	Blinding is not relevant to our study, because our study dose not include clinical experiments.

Reporting for specific materials, systems and methods

We require information from authors about some types of materials, experimental systems and methods used in many studies. Here, indicate whether each material, system or method listed is relevant to your study. If you are not sure if a list item applies to your research, read the appropriate section before selecting a response.

Materials & experimental systems

n/a	Involved in the study	
☒	☐	Antibodies
☒	☐	Eukaryotic cell lines
☒	☐	Palaeontology and archaeology
☒	☐	Animals and other organisms
☒	☐	Clinical data
☒	☐	Dual use research of concern
☒	☐	Plants

Methods

n/a	Involved in the study	
☒	☐	ChIP-seq
☒	☐	Flow cytometry
☒	☐	MRI-based neuroimaging

Plants

Seed stocks	No plants
Novel plant genotypes	No plants
Authentication	No plants