Revealing principles of autonomous thermal soaring in windy conditions using vulture-inspired deep reinforcement-learning

Yoav Flato,[1,2,3] Roi Harel,[4,5,6,7] Aviv Tamar,[8] Ran Nathan,[7] and Tsevi Beatus[1,2,3,a)]

[1)] *Rachel and Selim Benin School of Computer Science and Engineering, The Hebrew University of Jerusalem, Jerusalem, Israel 9190401*

[2)] *Department of Ecology, Evolution, and Behavior, Alexander Silberman Institute of Life Sciences, The Hebrew University of Jerusalem, Jerusalem, Israel 9190401*

[3)] *Grass Center of Bioengineering, The Hebrew University of Jerusalem, Jerusalem, Israel 9190401*

[4)] *Department for the Ecology of Animal Societies, Max Planck Institute of Animal Behavior, Konstanz, Germany 78467*

[5)] *Department of Biology, University of Konstanz, Konstanz, Germany 78457*

[6)] *Centre for the Advanced Study of Collective Behaviour, University of Konstanz, Konstanz, Germany 78457*

[7)] *Movement Ecology Lab, Department of Ecology, Evolution, and Behavior, Alexander Silberman Institute of Life Sciences, The Hebrew University of Jerusalem, Jerusalem, Israel 9190401*

[8)] *Department of Electrical and Computer Engineering, Technion, Haifa, Isarel 3200003*

(*Electronic mail: tsevi.beatus@mail.huji.ac.il)

Thermal soaring, a technique used by birds and gliders to utilize updrafts of hot air, is an appealing model-problem for studying motion control and how it is learned by animals and engineered autonomous systems. Thermal soaring has rich dynamics and nontrivial constraints, yet it uses few control parameters and is becoming experimentally accessible. Following recent developments in applying reinforcement learning methods for training deep neural-network (deep-RL) models to soar autonomously both in simulation and real gliders, here we develop a simulation-based deep-RL system to study the learning process of thermal soaring. We find that this process has learning bottlenecks, we define a new efficiency metric and use it to characterize learning robustness, we compare the learned policy to data from soaring vultures, and find that the neurons of the trained network divide into function clusters that evolve during learning. These results pose thermal soaring as a rich yet tractable model-problem for the learning of motion control.

I. INTRODUCTION

Vultures and other large birds soar by using thermal updrafts, which are localized warm air columns that rise due to their buoyancy. Thermal soaring enables these birds, as well as human glider pilots, to fly for long distances with high energetic efficiency and under challenging wind conditions[1–5]. These impressive capabilities make bird-like thermal soaring a model for imitation in unmanned aerial vehicles (UAVs)[6]. More broadly, thermal soaring is an appealing model-problem for studying motion control, and how motion control is learned by animals and by autonomous engineered systems. The main reasons are that thermal soaring has rich dynamics and nontrivial constraints, yet it uses relatively few control parameters compared, for example, with cursorial animals and robots. Additionally, thermal soaring is relatively simple to model owing to the lack of contact forces and the availability of reliable and simple quasi-steady aerodynamic force models[7]. Finally, thermal soaring is becoming more experimentally accessible: on the one hand, data on the flight kinematics of soaring birds is becoming available[1] and, in parallel, implementing thermal soaring in small-scale radio-controlled gliders with onboard AI-based control has become affordable at a relatively low budget[8–10] .

Indeed, recent pioneering studies have achieved autonomous thermal soaring in simulated environments and in unmanned gliders using various methods. Rule-based algorithms estimate the position of the thermal's center and rely on a set of pre-programmed rules to achieve thermal soaring[11–17]. Although these algorithms are pragmatic, effective, and thoroughly comprehended, they do not explore innovative solutions and, therefore, have constraints in advancing our understanding of motion control as a learning challenge.

This limitation can be alleviated by using machine learning (ML) and, particularly, reinforcement learning (RL) methods[18–21]. In RL, an agent operates in an environment to maximize a specific goal function, referred to as a *reward*. Reward maximization is done via a learning process and, as the agent learns, it acquires a policy that determines its behavior. The policy controls the agent's *actions* based on a set of available observations of the environment, which is known as the agent's *state*. This setting enables the agent to explore the environment and discover its own strategy. Thus, in an ideal learning process, the agent can find an optimal policy whose complexity is limited only by the agent's computational architecture. The agent's policy may even exceed the performance of policies based on human intuition or expertise[22].

The first application of RL to thermal soaring was presented by Reddy *et al.*[8,23], who trained a rule-based RL agent using the SARSA algorithm operating in an environment free of significant horizontal winds. The agent is represented by a lookup table, which defines the glider's discretized action (bank angle) as a function of its discretized current state (vertical velocity and bank angle). This system achieved successful thermal soaring both in simulations and on a real glider. Although the policy of rule-based RL can be easily interpreted, it is likely

[a)] http://www.beatus-lab.org

that richer ML architectures may achieve better gliding performance[24], especially under more challenging environments, for example, with thermals that are drifted by horizontal winds. This particular condition is probably the norm rather than the exception in thermal soaring[1,4].

Currently, the most relevant ML architecture for such tasks is deep neural networks (NN). Combined with state-of-the-art RL algorithms (deep-RL), such as actor-critic and policy-gradient[25–27], these methods have led to numerous successful applications in complicated motion control and planning tasks[18–21]. In particular, deep-RL methods have been successfully implemented in autonomous thermal soaring both in simulated environments and, more impressively, in real gliders. Novati *et al.* showed that deep-RL can robustly solve a gliding control problem of a simulated elliptical object, achieving gliding and landing in a complex flow environment[28]. Notter *et al.* implemented thermal soaring both in simulation and in a real glider by using deep-RL with an LSTM (long short-term memory) network architecture, which is a recurrent-NN which implements memory of previous states and actions[9,10]. Moreover, Notter *et al.* used deep-RL to treat the thermal soaring problems in two levels: first, locating and exploiting an individual thermal under horizontal wind conditions[10] and, second, solving the decision-making strategy in cross-country soaring. Such strategy should balance two, often conflicting objectives: locally, it should exploit thermal updrafts to gain altitude and, globally, it should race between designated way-points[9,29]. The exploitation-exploration problem in thermal soaring was also addressed by Cui *et al.*, who solved it in a simulated environment using deep-RL and combining new energy considerations[30]

The success of deep-RL models in handling thermal soaring opens the way for using such models to address basic questions related to the learning of motion control. These include, for example: what is the structure of the problem in terms of its bottlenecks – sub-problems that must be solved sequentially to achieve the final behavioral goal[31]? What sensors and actuators are crucial for successful learning? How robust is the acquired policy? Can we dissect the agent's neural network – a computational object that is typically treated as a 'black box' – to gain insight into its function[32–34]? And, how does a learned policy compare with the thermal soaring behavior of birds in the wild?

Here, we address these questions by developing and studying a simulation-based deep-RL solution for thermal soaring under challenging horizontal winds. Analyzing the reward dynamics revealed that this problem has at least two bottlenecks that can be alleviated by specific modulations of the reward function. The robustness of the RL policy and of the learning process were characterized by evaluating the agent's performance in unencountered environmental conditions and under noise, as well as by comparing various state and action representations and different learning dynamics. We show, for example, that curriculum learning, in which the horizontal wind speed gradually increases during the learning process is crucial for achieving a policy that can handle different wind speeds. Sensing the ambient wind and controlling the angle-of-attack are also crucial. Additionally, we analyzed the activation patterns of the agent's NN 'brain' and revealed distinct neuronal patterns associated with specific behavioral modes that evolve during the learning process, *i.e.,* with the agent's 'age'. And, finally, we identified similarities between the learned policy and the soaring technique and learning dynamics observed in free-ranging vultures. Overall, we believe that these results both contribute for improving deep-RL implementations in thermal soaring, and more broadly, pose thermal soaring as an rich yet tractable model-problem for complex motion control.

II. RESULTS

A. Thermal soaring under horizontal wind

Using deep-RL, we trained a NN-based agent in a simulated environment to locate and soar in thermal updraft (Fig. 1 and Methods Section). The environment included one thermal that drifted horizontally at speed u (Figs. 1c,d and Methods Section). The chosen parameters of the system, referred to as the 'nominal' configuration, are as follows. The agent's state at a given time step consisted of six parameters $\{V, v_z, \sigma, \alpha, \theta, u\}$ (Figs. 1b,e,f), with overall speed V, climbing rate v_z, bank angle σ, angle of attack α, angle with respect to the wind θ, and wind speed u. Additionally, the state included a memory buffer of these parameters for the 8 previous 1 sec time steps. The agent's action was $\{\Delta\sigma, \Delta\alpha\}$, which are the controlled modulations in σ and α, respectively. The reward per each simulation step was a sum of two components: the agent's climb rate v_z and a penalty P_{center} equal to $-d/50$, where d is the agent's horizontal distance from the thermal center in meters. The thermal center is not part of the agent's state. Assigning such penalty is only feasible for training in simulations, as ground truth for the updraft center is hardly available in a real-world scenario. Yet, training with this penalty may still be indirectly advantageous in these cases (see Discussion). Additionally, the agent was penalized in two cases. First, if the agent went unstable, defined as spinning 360° in pitch, the simulation was stopped and the agent was penalized by P_{stab}, equal to -1 per each second left until the end of the run. Second, if the agent crashed ($z = 0$), the simulation was stopped and the agent was penalized by -1000. We used the DDPG algorithm where both actor and critic NNs included two fully connected hidden layers with 200 neurons each (Methods and Appendix C). Sample thermalling trajectory is shown in Fig. 1g.

We found that using curriculum learning was crucial for achieving successful soaring under horizontal winds. To this end, we gradually increased u during the learning process. First, the agent was learning on environments with u randomly selected within the range 0–2 m/sec. Then, the range of u was increased to 0–4 m/sec and, finally, to 0–6 m/sec. Each of these three steps consisted of $3–5 \cdot 10^6$ sec of simulated flight.

Agents with the nominal configuration achieved stable thermal soaring under horizontal winds of u=0–6 m/sec (Fig. 2 and Videos 1, 2). Fig. 2a shows representative trajectories of the same agent, soaring under u=1, 3 and 5 m/sec. As agents were, on average, placed initially close to, but not within, the thermal, the trajectories include a short searching phase of 5–20 sec that consisted of a circling motion. Once the agent located the thermal, it started

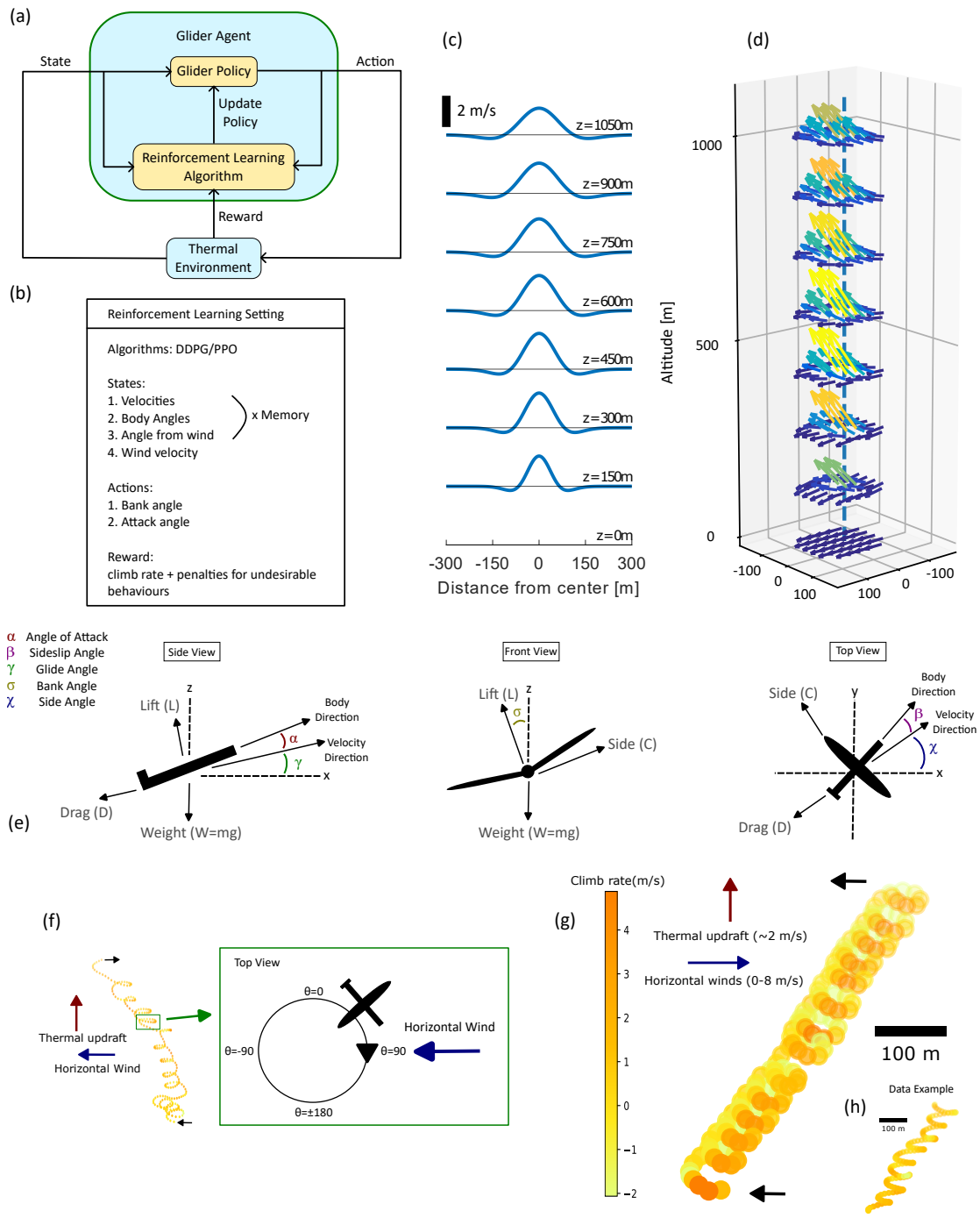

FIG. 1. **Reinforcement learning system for simulation-based thermal soaring.** **(a)** Overall system layout. **(b)** The state, action, reward and algorithm settings of our RL system. **(c)** Thermal updraft distribution as a function of distance from the thermal center r, for different heights. The thermal model parameters are w^*=5, z^*=2000 (Methods Section IV B). **(d)** The velocity field of a thermal with horizontal wind. Each arrow represents wind velocity direction and magnitude at a given (x,y,z) position. Arrows are color-coded by the magnitude of v_z. The horizontal wind is u=3 m/sec and thermal parameters identical to the thermal in (c). **(e)** The glider was modeled as a point mass with lift force L, drag force D, side force C and weight W. The three views shows the angles that define the glider's orientation: the angle-of-attack α; the sideslip angle β between the body axis and the velocity; the glide angle γ; the bank (roll) angle σ and the side angle χ of the velocity with respect to the horizon. **(f)** The angle from wind, θ during circling motion. Left: A trajectory of a thermalling vulture taken from data[1]. Right: a scheme of one circle within the thermalling motion that demonstrates the values of θ (the angle between the glider velocity and the wind u in the xy plane) along the circle. θ=0 represents motion against the wind (headwind) and $\theta = \pm 180^\circ$ represents motion in the wind direction (tailwind). θ is defines such that it is increasing during the circling. Here, for example, the glider is rotating clockwise, hence when it goes from headwind to tailwind, θ increases from 0 to 180°. For counter clockwise rotations, θ is mirrored such that it still increases when transitioning from headwind to tailwind. **(g)** A representative flight trajectory of the nominal agent configuration, which shows climbing thermalling motion under horizontal wind (blue arrow). The agent's climb rate is color coded along the trajectory. **(h)** A representative flight trajectory of a free-ranging vulture during thermal soaring[1]. Scale bars are 100 m.

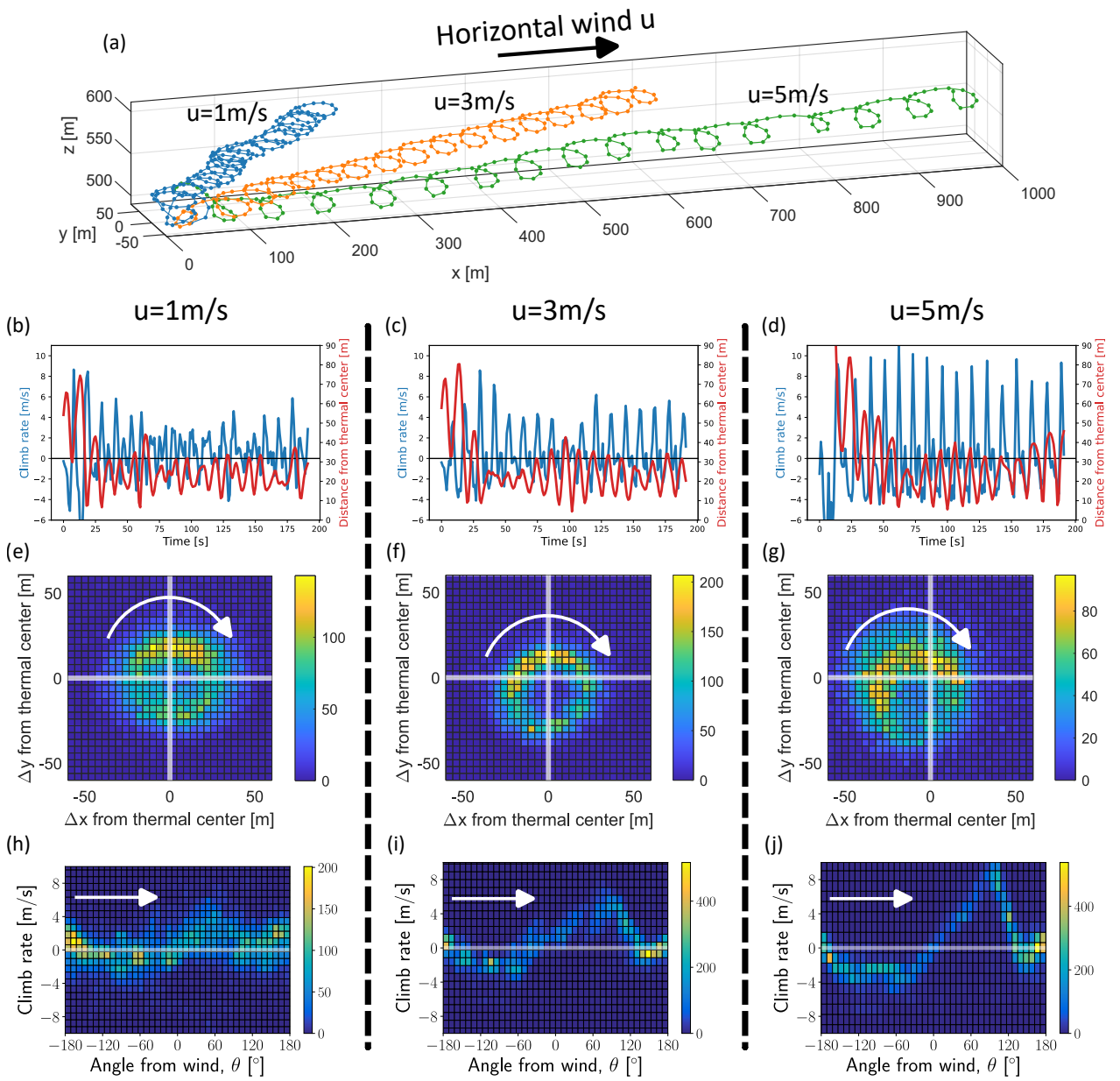

FIG. 2. **Thermal soaring of the nominal agent under various horizontal wind speeds (a)** Representative 3D soaring trajectories in winds of 1 m/sec (blue), 3 m/sec (orange) and 5 m/sec (green). The wind is in the $+x$ direction. **(b-d)** The climb rate v_z and the agent's distance from the (moving) thermal center as a function of time, for the same trajectories in (a). The left column is for u=1 m/sec, middle column is for u=3 m/sec, and the right column is for u=5 m/sec. **(e-g)** histograms of the agent's xy position with respect to the the moving thermal center. Each histogram for a given value u was calculated for 100 runs with random initial conditions. The white arrows indicate the direction of circling. **(h-i)** 2D histograms of the climb rate v_z and the angle from wind, θ. The histograms were calculated for the same 100 runs per each u. The white arrows indicate the direction of circling.

thermalling: circling the thermal center and simultaneously tracking it downwind. To track the drifting thermal, the agent decreased its turning radius every time its trajectory is aligned with the wind, *i.e.*, at angle of $\theta \approx \pm 180°$ with respect to the wind (in the $+x$ direction). This motion is most evident in the u=5 m/sec trajectory, where the trajectory becomes almost parallel to the x axis. The resulting trajectories are trochoid in the xy plane, which manifests a thermalling technique similar to the techniques of soaring vultures[1] and professional human gliders[4,5]. Each learning session resulted in an agent that consistently rotated either clockwise or anti clockwise; the agent in Fig. 2 is of the clockwise type. During thermalling, the agent keeps itself in a range of 10–50 m from the thermal center, with an average distance of ~25 m (Figs. 2b-g). The agent's climb rate v_z consists of two contributions: the updraft velocity of the thermal (peak velocity ~2 m/sec), and the horizontal wind velocity, which the agent learned to convert to vertical velocity when flying against the wind.

To understand the agent's thermalling policy, we tested the agent on 100 random initial conditions for each of the three horizontal wind values u=1, 3 and 5 m/sec. Figs. 2e-g show two-dimensional (2D) histograms of the agent's position with respect to the moving thermal center, calculated

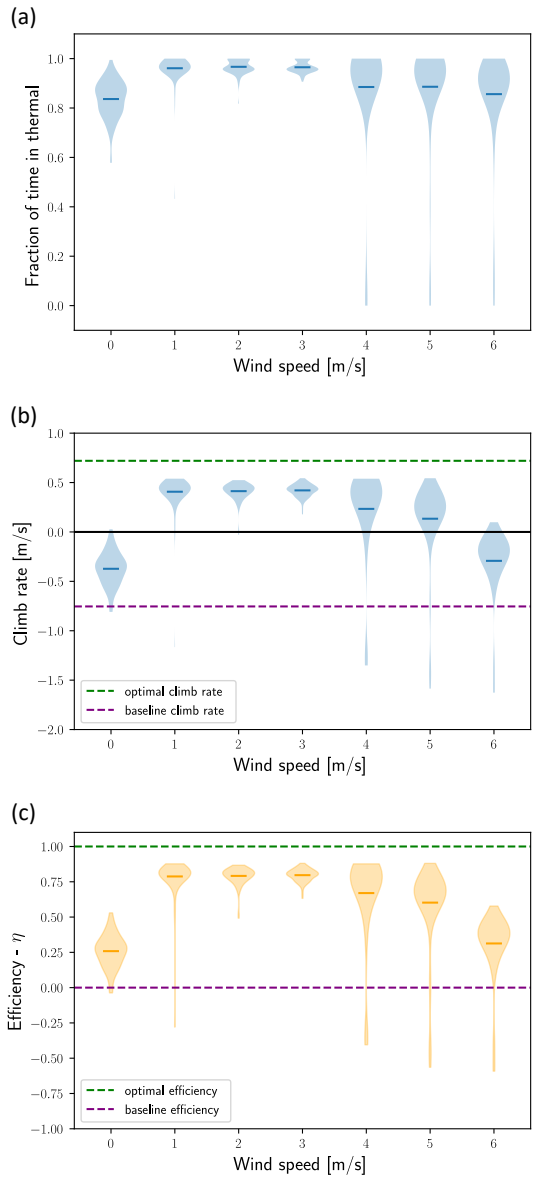

FIG. 3. **Thermal soaring performance of the nominal agent at different horizontal wind speeds.** Agent's performance was analyzed over 100 runs with random initial conditions for each value of the horizontal wind speed u. The performance measures are presented in violin plot which show the distribution over all runs with the mean marked by a horizontal line. For reference, each plot shows also the optimal and baseline climb rates (Appendix B). **(a)** The time the agent spent within the thermal as a fraction of its full run of 200 sec. **(b)** The distribution of average climb rates for each u. **(c)** The distribution of the thermal soaring efficiency η for each u (Eq. 1).

for each u across all 100 runs. The histograms show that the agent learned to move around the thermal center in a radius of ~25 m. Interestingly, for $u \geq 3$ m/sec, the center of the circling motion does not align with the thermal center, but the peaks of the 2D distributions show that along its circular trajectory, the agent spends more time closer to the thermal center. In addition, in this part of the trajectory, the agent chases the moving thermal ($\theta \approx \pm 180°$, tail wind). Analyzing v_z across the trajectories, shows that the agent gains most of its vertical velocity at angles of $\theta \approx 90°$ with respect to the wind (Figs. 2h-j). For $\theta < 90°$ the agent moves against the wind, and for $\theta > 90°$ it moves along the wind. For u=5 m/sec we see a clear signature of the agent's vertical acceleration as it rotates in θ, through the slope of $v_z(\theta)$. The vertical acceleration is highest between $\theta = -45°$ and 90° when the agent mostly flies against the wind, showing it exploits the wind to increase v_z. As u increased, the variability of the agent's σ and α increased, indicating a higher control load. For example, the standard deviation of σ increased from 6° in u=0 to 12° in u=5 m/sec.

To quantify the agent's soaring performance, we first calculate the fraction of its flight time spent in the thermal, *i.e.*, 80 m from its center, where the thermal updraft is still positive. Fig. 3a shows that under horizontal winds of 0–6 m/sec the agent spends most of its flight time thermalling. To quantify soaring performance in terms of the agent's mean climb rate v_z^{agent}, we define a soaring efficiency metric , η, which uses two limits for the climb rate: an upper bound v_z^{optimal} and a baseline v_z^{baseline}:

$$\eta = \frac{v_z^{\text{agent}} - v_z^{\text{baseline}}}{v_z^{\text{optimal}} - v_z^{\text{baseline}}}. \tag{1}$$

The upper bound, v_z^{optimal}, was estimated for a thermal with the parameters used in our environment: w^*=5 m/sec and z^*=2000m (Methods Section IV B and Eq. 3). It was derived from the equations of motion (Eq. 2) by characterizing their steady state under constant (σ, α) in the wind frame-of-reference. The steady state radius was then used to calculate the v_z upper bound in a circular trajectory within the thermal. Under these settings, the upper bound for the climb rate is v_z^{optimal}=0.72 m/sec (see Appendix B for the full derivation)The second limit of the climb rate $v_z^{\text{baseline}} = -0.75$ m/sec (*i.e.*, a sink), is the maximum climb rate that can be achieved with no thermal updraft. This climb rate is obtained in a straight descending flight with σ=0 and α=6°. Naturally, for gliders, $v_z^{\text{baseline}} < 0$, hence, for example, any positive v_z^{agent} indicates positive soaring efficiency. Overall, η represents the fraction of the updraft exploited by the agent. An agent with η=1 uses all the thermal updraft; an agent with η=0 can be gliding straight at v_z^{baseline}; and an agent with η<0 is doing worse than v_z^{baseline}.

Fig. 3b shows the distributions of the agent's mean v_z value for different horizontal winds, compared with v_z^{baseline} and v_z^{optimal}, and Fig. 3c plots the corresponding η distributions. The agent achieved η>0 across the entire range of horizontal wind speeds and performed best in the mid-range u=1–3 m/sec, with $\eta \approx 0.8$. The highest average v_z was 0.42 m/sec obtained at u=3 m/sec. For this u, when omitting the first 20 sec of each trajectory, the mean v_z reaches 0.54 m/sec (maximum of 0.67 m/sec), the mean η reaches 0.88 (maximum of 0.96).

B. Identifying learning bottlenecks via reward shaping

To reveal the underlying structure of the thermal soaring problem and break it into meaningful sub-problems, we tested the nominal agent in a slightly more challenging protocol, in which u was randomly selected between

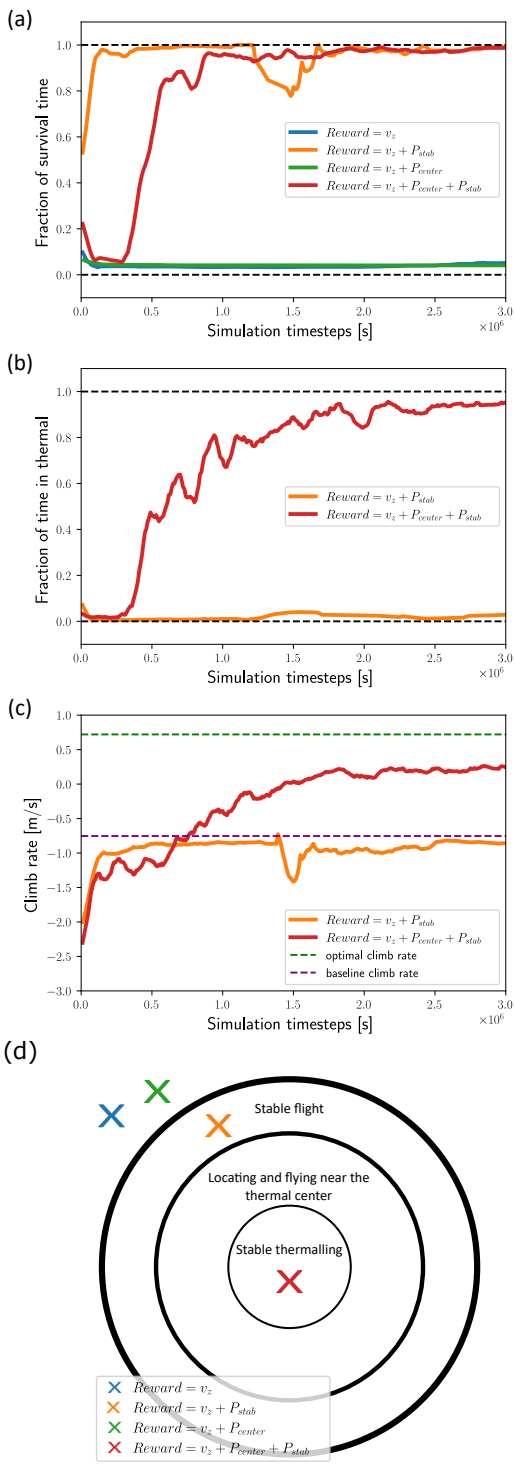

FIG. 4. **Learning bottleneck. (a-c)** Gliding performance during the learning process as a function of simulation time in seconds. The four colors represents agents with different reward functions (legend). **(a)** The fraction of stable flight time *vs.* learning time for different reward agents. **(b)** The fraction of time spent in the thermal area *vs.* learning times for different rewards **for which agents that survived a significant time**. **(c)** Climb rate *vs.* learning times for different reward functions, in which agents that survived for a significant duration. **(d)** Learning bottlenecks are represented as concentric circles of flight technique that should be learned to achieve stable thermalling (inner circle). The × symbols indicate the stages reached by the agents in (a-c) with the same color coding. The transition into the outermost circle represents the first learning bottleneck related to stabilization. The transition to the next inner circle corresponds to the second bottleneck: locating and flying near the thermal. The final transition into the innermost circle of "stable thermalling" was achieved via a smooth process facilitated by the v_z reward.

2.5–3.5 m/sec, without curriculum learning. We identified that in our system the task of efficient thermal soaring is too difficult for learning when the reward is simply the climb rate v_z. As shown Fig. 4a, an agent with this reward did not manage to stabilize itself and crashed in a very early stage of the simulation. The first sub-problem of thermal soaring is, therefore, achieving stable flight. Hence, we changed the reward function by adding the penalty P_{stab}, defined above (Section II A), which penalizes the agent when it gets unstable. Agents that learned with the reward $v_z + P_{\text{stab}}$ achieved stable flight (Fig. 4a) but did not manage to find the thermal and to spend a significant time in its vicinity (Fig. 4b). This reveals the second sub-problem of thermal soaring – staying close to the thermal center. Thus, we defined another penalty term, P_{center}, proportional to the agent's distance from the thermal center (Section II A). Agents learning with the combined reward $v_z + P_{\text{stab}} + P_{\text{center}}$ managed to fly both stably and near the thermal center, and eventually learned to circle the center to soar efficiently (Fig. 4c). Interestingly, agents with the reward $v_z + P_{\text{center}}$ did not overcome the stabilization bottleneck (Fig. 4a).

In summary, these results indicate that the problem of thermal soaring, as modeled here, consists of at least two sub-problems that can be manifested as learning bottlenecks (Fig. 4d). The first bottleneck identified is achieving stable flight, and the second is locating and flying near the thermal center. Using reward shaping assisted the agent to sequentially funnel its flight technique through those bottlenecks and achieve efficient soaring. Interestingly, these gliding principles, which are known by human glider pilots, were 'rediscovered' by the current RL system using a neural network with only 400 inter-neurons.

C. Probing the state and action representations

As part of the state optimization process, we tested agents with several state representations, whose performance shed light on the importance of specific attributes of the nominal state representation: the horizontal wind speed u and the memory size of previous states.

Horizontal wind speed. First, we compared the nominal agent to an agent whose state representation excluded u, including only $\{V, v_z, \sigma, \alpha, \theta\}$. Both agents' memory included their current state and the state in the previous 8 time steps, and they were trained in the nominal curriculum learning protocol (section II A). Each agent was evaluated on a range of horizontal winds u=0–6 for 100 runs on each value of u. Fig. 5a shows that the agent whose state did not include u was significantly less efficient than the nominal agent which had u in its state. Additionally, we found that agents without knowledge of u were able to learn to soar under one specific horizontal wind $u_0 \leq 8$ m/sec, when trained in a curriculum learning protocol which gradually increased in 1 m/sec steps. However, these agents performed poorly on horizontal winds weaker than u_0, indicating that this configuration was over-fitted for a specific horizontal wind. Together these results demonstrate that u is a crucial state parameters, and hint that an agent without explicit knowledge of u would have to infer it from other information (see the Discussion Section).

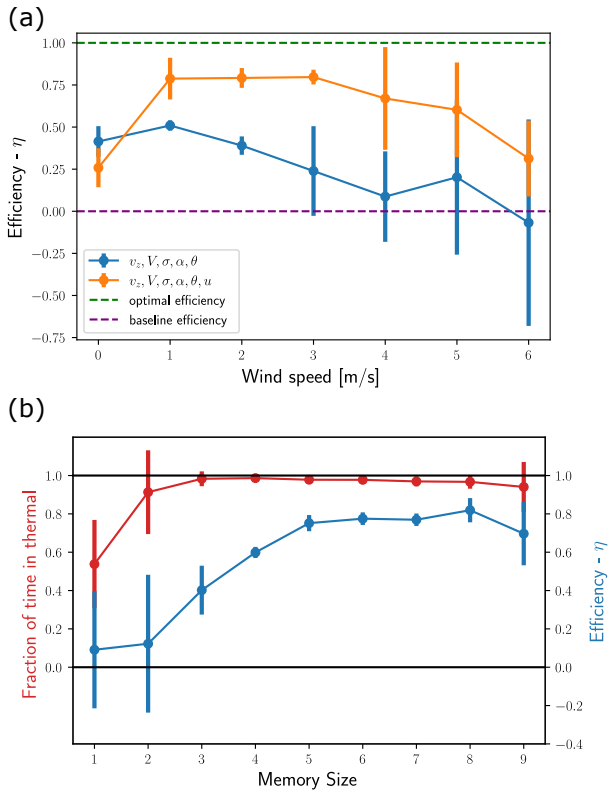

FIG. 5. **State representation.** **(a)** The horizontal wind u is a crucial state variable. The thermalling efficiency η was calculated for two agents under the same learning protocol. One agent (orange) had u included in its state, and the other agent (blue) did not. The thermalling efficiency is plots as the mean η on 100 runs in each horizontal wind speed u from 0 to 6 m/sec. For reference, the baseline and optimal values of η are plotted as well. **(b)** Agent performance as as function of its memory buffer size. The efficiency (blue) and fraction of time spend within the thermal (red) are plotted for the agents trained under u=1−3 m/sec and tested on u=2 m/sec. These results show that a memory buffer of 3 sec is sufficient for flying within the thermal, but efficient soaring requires a buffer of at least 5 sec.

Memory size. Next, we compared the nominal agent to a series of agents whose state representations included memory buffers of different sizes, from 1 to 9 previous time steps (nominal is 8), in addition to the current state. These agents were trained under horizontal winds of 1−3 m/sec and evaluated on u=2 m/sec. Fig. 5b shows that for staying mostly in the vicinity of the thermal it is sufficient to have memory of ≥2 time steps. However, for exploiting the thermal and soaring efficiently it is required to have memory of ≥5 time steps, as measured by η. Interestingly, inferring the center position of a thermal updraft from local updraft observations is a non-Markovian problem. A previous observability analysis of thermal soaring[29], has also shown that a history of ≥2 time steps is sufficient for this inference.

Actions. Our model allows for three control parameters: $\Delta\sigma$, $\Delta\alpha$, and $\Delta\beta$, which represent modulations the the bank angle, angle-of-attack and sideslip angle, respectively. While the bank angle is crucial for thermalling[4,8,9,35], it is insufficient for stabilizing in strong horizontal winds. As taught in basic gliding courses[4], our system consistently showed that adding $\Delta\alpha$ was sufficient for achieving efficient thermalling, whereas further adding $\Delta\beta$ resulted in slow convergence. Therefore, the actions of the nominal agent we set to $\Delta\sigma, \Delta\alpha$.

D. Robustness of the nominal agent

To mimic several aspects of the real world, we characterized the robustness of the nominal solution with respect to changes in the parameters of the thermal and to sensor noise.

Parameters of the thermal. The nominal agent was trained on environments with random horizontal wind and initial condition but with fixed thermal parameters, w^* and z^* (Methods Section IV B). To evaluate the agent's ability to generalize its technique for different thermal parameters, we tested its performance in a range of w^* and z^* values without further training (Fig. 6a). For each combination of w^* and z^* we ran 20 simulations with horizontal wind of u=3 m/sec and random initial conditions. The agent performance was evaluated by the efficiency metric η and by calculating its average v_z and optimal v_z under each condition. Fig. 6a shows that for the weaker thermals with w^*=3 m/sec the agent performed quite poorly, but it managed to generalize for all the tested values of w^*≥5 m/sec and all z^*, where its efficiency was >70%.

Sensor noise. We modeled sensor noise by adding Gaussian noise to the state variables $\{v_z, V, \sigma, \alpha, \theta, u\}$, one variable at a time. Fig. 6b shows the average climb rate of the nominal agent under different noise levels. The noise levels were normalized with respect to the standard deviation of the sensor readout, as a typical scale for its variation. Overall, the climb rate reduced with the noise level, with different performance degradation for noise in different sensors. For example, the most sensitive sensor was θ (angle with respect to u): relative noise of ~0.65 degraded the agent's performance down to the baseline climb rate, equivalent to gliding with no thermal. The least sensitive sensor was V, for which a similar performance degradation was met for relative noise of ~1.7.

Testing the performance under sensor noise in u (Fig. 6c), shows that the agent's performance degraded to the baseline climb rate at a Gaussian noise with standard deviation of ~7 m/sec, greater than the horizontal wind speed u=3 m/sec used in this trial. This result hints that the agent averages the readouts of u across its memory buffer. Finally, we tested the agent's robustness to environmental noise in u, which simulated a horizontal wind with gusts that change every 20sec and have a Gaussian distribution. The agent performance monotonically degraded to the baseline at gusts with standard deviation of ~3 m/sec, where the mean u was 3 m/sec.

E. Opening the black box: analyzing the neural network's functional modes

The highly complex and opaque nature of NNs makes its difficult for us to understand the resulting gliding policy. To address this challenge, we analyzed the NN activity by clustering the neural activation values in the hidden layers, to obtain clusters with similar activation patterns (Meth-

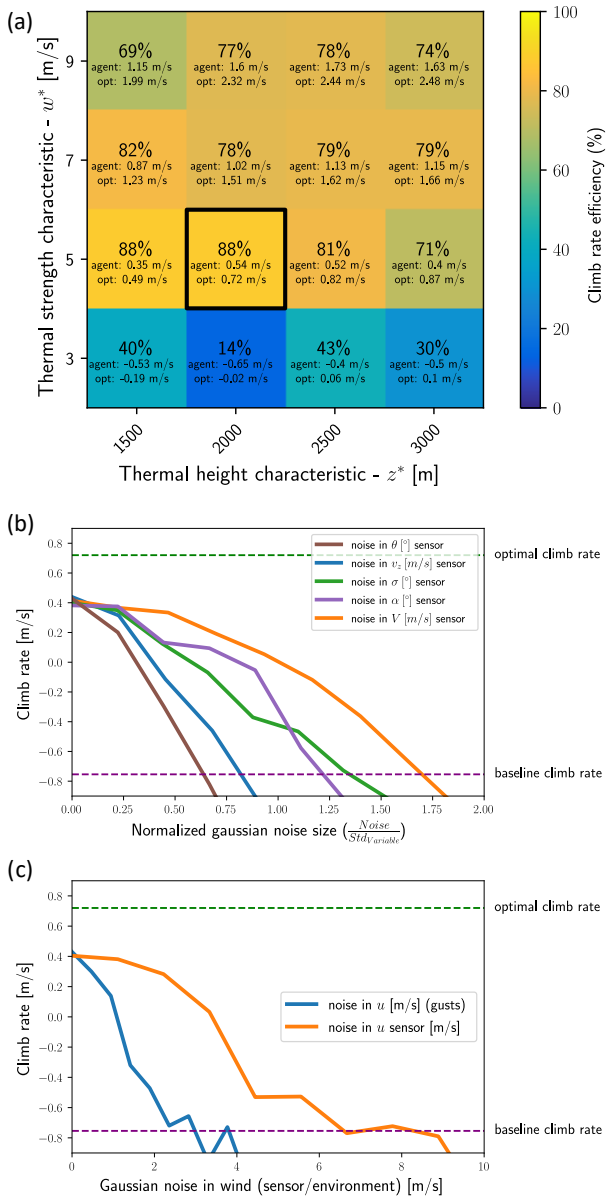

FIG. 6. **(a,b)** Climb rate performance degrade with sensor noise and wind gusts. **(a)** The average climb rate of the nominal agent under different types of sensor noise and different normalized noise levels. The noise level in each sensor was normalized by the standard deviation of the measured variable. The standard deviations were: std(V)=2.3 m/sec, std(v_z)=2.6 m/sec, std(θ)=116.4°, std(σ)=11.9°, std(α)=5.9°. **(b)** The orange line describes the agent's average climb rate as a function of noise level in its u sensor, as defined in the main text. The blue line shows the average climb rate as a function of noise in the *environmental* value of u, which represents physics gusts that change over a 20 sec timescale. **c)** Climb rate performance in different thermal parameters. To quantify the system's generalization, the nominal agent was tested on a 15 combinations of thermal parameters (z^*, w^*) different than the values on which it was trained (dark square). For each combination of (z^*, w^*) we averaged over 20 runs in random initial conditions. The plot shows the mean efficiency η in percent (color coded), the mean v_z and v_z^{optimal}.

ods Section IV D). We analyzed a nominal agent that was trained in a range of horizontal winds 1−2 m/sec, whose performance during the learning process is shown in Fig. 7a. We selected three copies of the agent during the learning process and labeled them 'young', 'intermediate' and 'expert'. Representative trajectories of these three stages are show in Fig. 7b. Clustering the NN activation patterns revealed 4 distinct clusters. At any given time within a flight trajectory we associated the internal NN activation pattern with its closest cluster and obtained a labeling of each timepoint according to its cluster (color coded in Fig. 7b).

Markedly, this cluster labeling can be associated with specific stages of the circling flight in the thermal. Labeling is mostly associated with the angle of the glider with respect to the horizontal wind, which is equivalent to its phase in the circling motion (Fig. 1f). Furthermore, this association becomes more apparent with increasing agent's 'age': Fig. 7c, which plots how cluster labeling is distributed by θ, shows that as the agent 'matures', the distribution of clusters becomes more correlated with θ. For example, in the 'expert' agent, one cluster (red) is correlated with flight against the wind in the range θ=−100° to 90°, a second cluster (purple) is correlated with the values from θ=90° to ±180°, and a third cluster (blue), aligns with the range θ=±180° to −100°. A fourth cluster (green) appeared for a brief time in the beginning of each trajectory and is associated with a short thermal searching phase. The changes in the clustering patterns over the agent's 'lifetime' is reminiscent of the differences between the thermal soaring of young and adult vultures, where the climb rate of young vultures is independent of θ and the (significantly higher) climb rate of adult vultures depends on θ[1]. To rule out the possibility that this functional association of the clusters with θ is a trivial outcome of having θ as a state variable, we performed the same analysis on an agent that did not have access to θ in its state (Appendix D). While this agent did not perform as well as the nominal agent, it did exhibit neural activation clustering with similar distribution in θ.

F. Comparing the RL agent to soaring vultures data

Finally, to compare the performance of our RL agent to the motion of free-ranging soaring vultures that was measured by Harel *et al.*[1], we selected 243 characteristic thermalling trajectories with consistent motion of ≥100 sec each, encompassing a total of 9.4 hours, that were taken during a narrow time period (August-September). The magnitude and direction of the horizontal wind, as well as the bank angle, were estimated from the data. To adjust the parameters of the thermals (z^*, w^*) to be used during learning, we applied the calculation of v_z^{optimal} (Methods Section IV E and Appendix B), while assuming that vulture climb in an optimal v_z. Then, we chose the values of (z^*, w^*) that correspond to the vultures' mean v_z and thermalling radius. Using these parameters, we trained a nominal agent with vulture-like mass m=7.75 kg and wing area S=0.87 m^2 and tested it over 100 random trials.

Comparison shows conspicuous similarity between the vulture and agent thermalling trajectories. The distributions of circling radii (Figs. 8a,b) of the vultures and agent

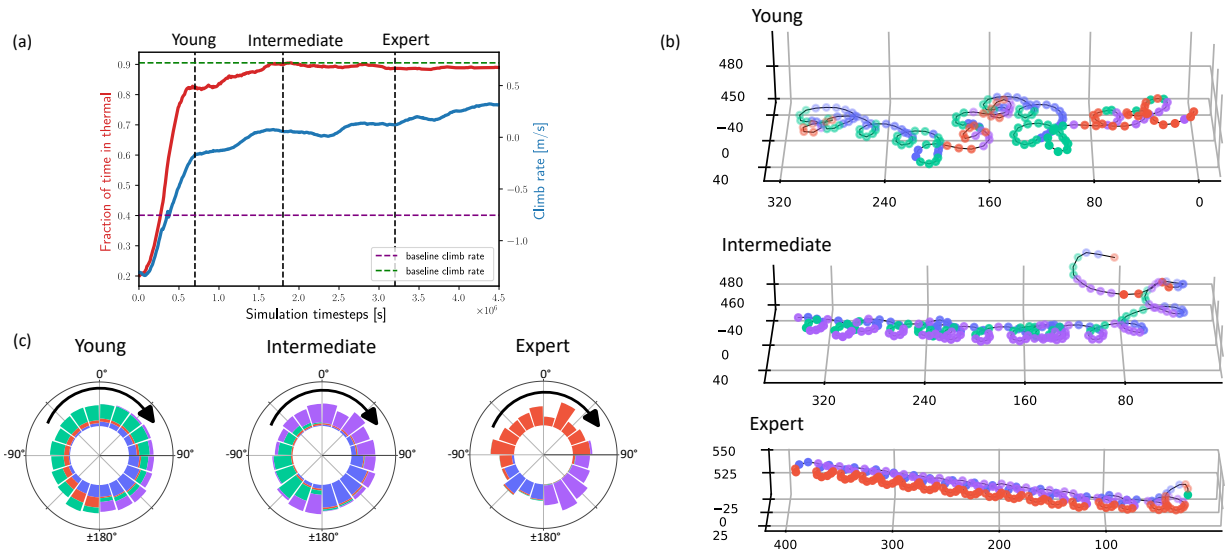

FIG. 7. **NN activation clusters through learning process.** **(a)** The evolution of agent performance during the learning process. Dashed vertical lines mark the times at which the "young", "intermediate" and "expert" agents were selected for further analysis. **(b)** Representative trajectories of the three agents colored the active neural cluster at each timestep. **(c)** Distribution of the neural activation clusters as a function θ for each of the three agent. Color coding is identical to (b).

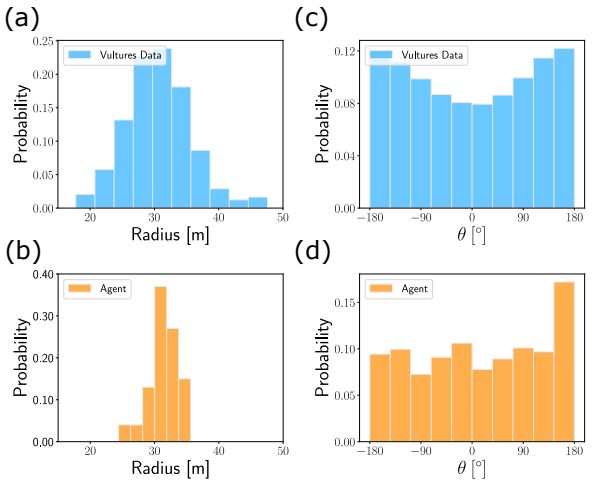

FIG. 8. **Comparing the RL agent and thermalling vultures.** **(a,b)** distributions of the thermalling radii for the vultures (a) and agent (b). **(c,d)** distributions of, θ, the angle with respect to the horizontal wind for the vultures (c) and agent (d).

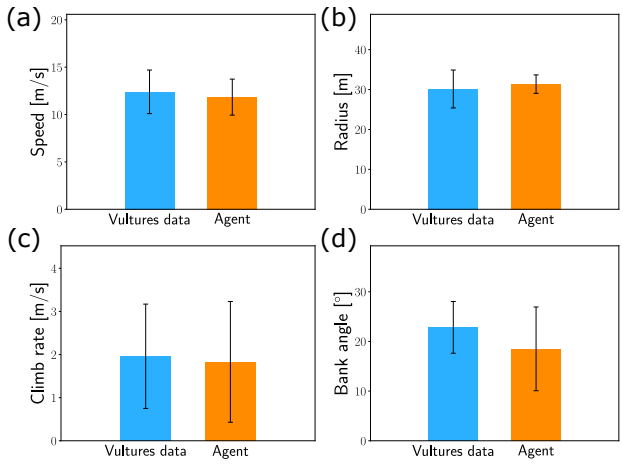

FIG. 9. **Comparing the mean trajectory parameters of the RL agent and thermalling vultures.** Each bar represents the mean value of a kinematic parameter, with the whiskers indicating its standard deviation. **(a)** Speed V. **(b)** Thermalling radius. **(c)** Climb rate v_z. **(d)** Bank angle σ.

have similar average of ~30 m, where the agent's distribution is narrower, probably owing to the synthetic simulation conditions. The vultures' distribution of θ is more structured that for the agent, but both peak around 180°, when flying along the horizontal wind (Figs. 8c,d). Fig. 9 shows that the mean values of the speed V, climb rate v_z and bank angle σ of both vultures and agent are comparable.

III. DISCUSSION

We developed a simulation-based deep-RL system that learned to perform efficient thermal soaring under strong horizontal wind, and we used this system to address questions about the learning of motion control. Modulating the reward function revealed an underlying structure of the thermal soaring problem in its current formulation. This problem consists of at least two bottlenecks, or subproblems, that must be solved sequentially to achieve efficient soaring: the first bottleneck is achieving stable flight and the second is flying at the vicinity of the thermal center. While using a reward that penalizes the agent based on its distance from the thermal center is applicable only in simulations and not during training in the real-world, such a penalty may be of some value. Pre-training real-world agents in simulation with such a penalty may improve their ability to estimate the thermal position based on their available state information, thereby bridging some of the 'sim-to-real' gap.

Identifying the learning bottlenecks and mitigating them using reward shaping enabled us to train a small NN to soar in relatively strong winds. The method of reward shaping may be applicable for finding the underlying structure of other complex problems in RL. Identifying learning bottlenecks and addressing them sequentially during learning can, therefore, improve the solution strategy and convergence time for problems that are too difficult for a direct solution based on a single final goal. This is similar to learning to crawl before learning to walk and run. Additionally, we showed that curriculum learning was crucial for learning to soar under different horizontal wind speeds. The principle of gradually learning to handle more difficult conditions may be applicable for other motion control problems.

Using a new efficiency metric, η (Eq. 1, Figs. 3, 5a and 6a), which quantifies how well the thermal is exploited by the agent, we characterized the robustness of the acquired policy with respect to environmental conditions, sensor noise and different state-action representations. The control over the environment, state, action and reward offered by such RL system, may be helpful in gaining insights into the limiting factors of complex behaviors such as thermal soaring. This methodology is widely applied for non-NN, models[36] and has recently been applied for deep learning and RL models of the sensory system of bats[34,37]. Here, for example, by testing various action-representations we found that controlling the glider's bank angle alone is insufficient for stable flight under strong horizontal winds, and that the angle-of-attack must be controlled as well. Testing different state representations highlighted the importance of the wind speed u and angle θ for efficient soaring. These results raise the questions of whether and how these parameters are measured or estimated by soaring birds, and how they can be estimated by UAVs. If not measured directly, can these parameters be observable by a given flying system, *e.g.*, by using a body-mounted airspeed sensor combined with a calculation that exploits body rotation[29,38]. characterizing the agent's robustness to sensor noise and wind gusts may provide physical limits to the relative importance of different sensory information in soaring birds. Further, comparing the flight trajectories of the RL-agent performance to the trajectories of soaring vultures together with the efficiency metric, learning bottlenecks and the evolution of the agents technique with its 'age', may shed light on the constraints and learning process of soaring birds. These concepts may be tested by long-term motion tracking of individual birds combined with accurate measurement of the local environment.

Finally, we analyzed the neural activation patterns of the agent's NN and found it divides into functional clusters. During the learning process, this clustering became more distinct, and in the 'expert' agent the clusters were associated with specific phases of θ during the circling motion around the thermal center. Further analyzing these clusters and their evolution during the learning process may inform us about the inner workings of the NN and may be used to understand the agent's policy by extracting explicit rules, for example, by describing each cluster as a decision tree[39–42]. In summary, the application of deep-RL method as presented here may contribute to further improving autonomous UAV gliding systems and to our understanding of motion control learning.

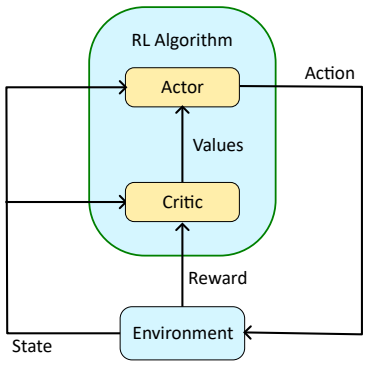

FIG. 10. **Actor-critic RL architecture.** The actor acts in the environment and the critic learns the environment, by assigning a 'value' to each state (or state-action pair). Critic values are used to update the actor. Together, the RL algorithm operates to obtain maximum cumulative reward.

IV. METHODS

A. Thermal soaring as an RL problem

Reinforcement learning is a type of machine learning in which an agent learns to take actions in an environment to maximize a cumulative reward signal (Figs. 1 and Fig. 10). The agent observes its state in the environment and chooses an action based on a policy $\pi : S \times A \rightarrow p$, where S is the state space and A is the action space. The policy π maps state-action pairs to the probability $p \in [0,1]$ of taking a particular action in a given state. The policy can be represented in various ways and here we represent it as a NN. The agent receives a reward based on the state resulting from the selected action. The goal of RL is to learn a policy that maximizes the expected cumulative reward. In our thermal soaring environment, the agent is a simulated glider with vulture-like parameters. The agent's kinematic parameters, such as speed and direction, represent its state, while the glider's control parameters, which are modulations of its aerodynamics angles, represent the actions. The reward signal can be designed to incentivize the agent to gain altitude, which in our simulated environment can be achieved by thermal soaring. Through trial and error, the goal of the RL algorithm is to shape the policy π to optimize the agent's behavior in the thermal soaring environment based on the reward received.

B. Simulation Model

In our RL framework, the environment is a simplified 3 degrees-of-freedom simulation model of a small glider in the presence of wind, used as an approximation for flight dynamics under moderate wind speeds. It is based on the simplified model of Beeler *et al.*[7], in which the glider is described as a point-mass aircraft in an environment with a spatially varying wind. The glider's parameters, such as mass, wingspan and aerodynamic coefficients are detailed in Appendix A. The glider model is described in Fig. 1 and its equations of motion are:

$$
\begin{aligned}
\dot{z} &= V\sin\gamma \\
\dot{x} &= V\cos\chi\cos\gamma \\
\dot{y} &= V\sin\chi\cos\gamma \\
\dot{V} &= -\frac{1}{m}D(V,\alpha,\beta) - g\sin\gamma \\
\dot{\gamma} &= \frac{1}{mV}\Big(L(V,\alpha)\cos\sigma + C(V,\beta)\sin\sigma\Big) - \frac{g}{V}\cos\gamma \\
\dot{\chi} &= \frac{1}{mV\cos\gamma}\Big(L(V,\alpha)\sin\sigma - C(V,\beta)\cos\sigma\Big).
\end{aligned} \tag{2}
$$

The dynamic variables are the (x,y,z) coordinates of the glider's center-of-mass, its velocity V, glide (pitch) angle γ, and side (yaw) angle χ. The glider's two control parameters are its bank (roll) angle σ and angle-of-attack α. The model supports a third control parameter – the sideslip angle β – which, as we verified, is not essential for obtaining robust thermalling. The glider mass is m and gravity acceleration is g. The lift force L, drag force D, and side force C, are calculated by a simplified aerodynamic model based on V, α, and β, which include the effect of the local air motion due to thermals and horizontal wind. Detailed description of the aerodynamic model is given in Appendix A.

The environment's atmospheric model consists of a sum of a horizontal wind and thermal updraft. Horizontal wind is approximated as uniform wind velocity u in the $+x$ direction. This direction is practically randomized with respect to the glider-agent, given that the agent has no prior information about its heading and its initial heading is randomized. Soaring birds and human glider pilots obviously use more information about the environment[43]. The magnitude of the horizontal wind is constant per each simulation run. We assume that u has no z dependence despite its known logarithmic profile $u(z)$ in open landscapes, because gliders and vultures mostly soar at relatively high elevations where the boundary-layer resistance (that shapes the logarithmic profile) is weak and $u(z)$ is nearly uniform. The shape of the updraft velocity vertical profile $w(z)$ is determined by Gedeon's model[44], and its amplitude $w_{\text{core}}(z)$ is given by Lenschow's model[45]. The combined vertical updraft velocity, as described by Bencatel is given in cylindrical coordinates (r,z),[46] is:

$$
\begin{aligned}
w(r,z) &= w_{\text{core}}(z)\, e^{-\left(\frac{r}{R(z)}\right)^2}\left[1-\left(\frac{r}{R(z)}\right)^2\right] \\
w_{\text{core}}(z) &= w^*\left(\frac{z}{z^*}\right)^{1/3}\left(1-1.1\frac{z}{z^*}\right) \\
R(z) &= 0.08\left(\frac{z}{z^*}\right)^{1/3}\left(1-\frac{1}{4}\frac{z}{z^*}\right)z^*,
\end{aligned} \tag{3}
$$

where w^*=5m/sec is the characteristic updraft amplitude and z^*=2,000m is the characteristic thermal altitude. In the resulting radial updraft profile (Figs. 1c,d), the thermal radius increases with height z and its velocity peak is surrounded by a ring of downwards velocity. For example, the peak updraft velocity at z=500 m is w_{core}=2.3 m/sec. To combine the horizontal wind and updraft velocity we use the chimney model[46,47], in which thermal motion may include both horizontal motion and leaning of its centerline. The results presented here use uniform horizontal motion without leaning. We achieved successful RL also with the full chimney model. The overall atmospheric model is, therefore:

$$
\mathbf{v}(x,y,z,t) = \begin{pmatrix} u \\ 0 \\ w\big(r(t),z\big) \end{pmatrix}, \tag{4}
$$

with $r(t) = \sqrt{(x-ut)^2+y^2}$ (Fig. 1d).

C. Deep-RL model

The deep-RL agent consists of a neural network that controls the simulated glider in an environment with a single thermal. Each simulation run spans 200 sec of flight, divided into 1 sec intervals. In each interval, the agent obtains its state and calculates a steering action. The action is implemented in the simulated dynamics of the next 1 sec interval. In a real glider system, this mode of operation would rely on a nested feedback loop that provides orientation control via the aircraft control surfaces. After a full simulation run, the agent gets a reward based on its cumulative performance. The state, action and reward are described below.

State. The agent's state consists of the following parameters:

- Glider's speed V.
- Climb rate v_z.
- Bank angle σ.
- Angle of attack α.
- The magnitude of the horizontal wind u.
- The angle θ between the glider's velocity vector and u (Fig. 1f).

Each state entry was normalized to the range $[-1,1]$. In gliders, these parameters can be measured by standard sensors, and can probably be sensed by vultures and other soaring birds. Additionally, the agent's state contains a short history of the state parameters, as vultures were found to integrate short-term information on their soaring performance from their current and recent experiences[48]. The length of this history is a hyper-parameter of the model. To characterize the robustness of our agents to noise, we tested some of the trained agents with sensor noise, where we added Gaussian noise to their state data (Section II D). To further examine the effect of the state on performance, we tested additional state representations, which included the glider's distance from the thermal center, and the difference of the vertical wind between the two wings.

Actions. The agent's two actions are[7,9]:

- Bank angle change, $\Delta\sigma$. In each 1 sec interval, the agent can change σ by $\Delta\sigma \in [-15°, 15°]$, and σ is limited to the range $[-50°, 50°]$.
- Angle of attack change, $\Delta\alpha$. In each time step the agent can change α by $\Delta\alpha \in [-10°, 10°]$, and α is limited to $[-30°, 30°]$.

We also tested agents that can modulate their sideslip angle and wingspan, and found that these control parameters are not crucial for the current task.

Reward. In analogy to vultures and gliders, the basic reward in our model was the glider's climb rate v_z. To overcome the bottlenecks that we identified in the learning process (section II B), we employed two additional factors in the reward, which can be considered as 'reward shaping'[49]. The first factor is a penalty for instability, P_{stab}: when the agent spins out of control, the simulation is stopped and the agent is penalized proportionally to the remaining simulation time. The second reward shaping factor is a penalty proportional to the glider's horizontal distance from the thermal center, termed P_{center}. Importantly, because this penalty is not part of the state, the agent has no direct information on its distance from the thermal center. This penalty helps in shaping the agent's behavior through the reward, by encouraging the agent to fly closer to the thermal center (Section II B).

Training algorithm. We used the DDPG (Deep Deterministic Policy Gradient) algorithm, which is a state-of-the-art actor-critic policy gradient method designed for continuous control problems[26] (Fig. 10). Actor-critic systems are composed of two neural networks: the actor network and the critic network. The actor determines the agent's action based on the current state, while the critic estimates the value or Q-functions associated with a given state and action. Both the value function and Q-function represent the expected cumulative future reward: the value function's estimate relies on a given state, while the Q-function's estimate relies on a given state and action. The critic network learns to estimate one of these functions by observing trajectories of (state, action, reward) generated by the actor in the environment. Most of the time, the critic's estimate is utilized to calculate the temporal difference (TD) error or advantage function. The TD error quantifies the discrepancy between the current estimate of the state-value and the discounted value estimate of the subsequent state, along with the actual reward obtained by the actor. The advantage function is the expectation of the TD error with respect to the next state, which also quantifies the quality of a certain action given a certain state. The critic's value function serves the dual purpose of updating the critic's value estimation and providing feedback to the actor regarding the quality of its chosen action. In the case of DDPG, the critic network specifically calculates the Q-function rather than the value function. In addition to DDPG, we tested the PPO (Proximal Policy Optimization) algorithm[25], which is another actor-critic policy gradient method. PPO also achieved successful learning, though in longer times.

We use the code implementation of the stable-baselines3 library[50], which includes various deep-RL algorithms based on the PyTorch library. We optimized the learning process by performing a hyper-parameter optimization[51] for the learning rate, learning algorithm (DDPG/PPO), number of NN layers, and layer size (Appendix C).

Overall, in the learning process, we begin with an initial neural network policy that maps states to actions. This policy is updated using a critic neural network through the policy gradient actor-critic method, which is executed in the thermal environment and uses the rewards obtained from the environment.

In each simulation run during the learning process, the glider was initialized at a random orientation, random horizontal position in (x,y), height of z=500m, velocity of 15 m/sec, and gliding angle of $-5°$. The thermal center was set to $(x,y)=(0,0)$, and the horizontal wind velocity u in the $+x$ direction was set to a random value between $u_{\min}$ and $u_{\max}$, which are the wind speed hyper-parameters. The thermal was, therefore, drifting at velocity u while keeping a vertical structure. These random initialization improved the agent's generalization capacity. On top of this scheme, we used a curriculum learning approach, where the maximum horizontal wind velocity, $u_{\max}$, was gradually increased during the learning process.

D. Interpreting and analyzing the deep-RL agents

To understand the agent's policy and performance during and after the learning process, we employed both kinematic analyses of the flight trajectories, and analyzed the NN itself by clustering the neural activation patterns in the network's hidden layers.

The activation of a neuron in the network is the output of its nonlinear activation function (ReLU), which, in our case, is a non-negative number. At each timestep, we represent the activation of the entire network by concatenating the activations of all neurons in the NN hidden layers into one vector. Prior work has shown that analyzing the hidden layers is useful for interpreting NN in both physical-ecological[34] and RL problems[33]. To perform the clustering, we used k-means clustering [52] on the concatenated activation vectors. To determine the number of clusters k, we used the knee locator method[53] based on the sum-of-squared-error (SSE)[54], which finds the k value at which the SSE curve starts to straighten up. We later show that specific clusters can be assigned to distinct flight modes.

E. Comparing the RL agent to soaring vultures data

We compared our RL model to vulture flight trajectories collected by Harel *et al.*[1]. The dataset consists of over 4 million GPS location samples, taken at 1Hz rate of 20 vultures of different ages collected over ~3 years. Among these locations, there were almost 1 million samples from thermal soaring in over 6,000 thermal updrafts, with each thermal soaring trajectory lasting more than 100 sec. Selection of thermal trajectories is described in Section II F.

To estimate the bank angle from the data, we consider the forces in the radial and vertical axes:

$$\begin{aligned} ma_r &= L\sin(\sigma)\cos(\gamma) - D\sin(\sigma)\sin(\gamma) \\ mg &= L\cos(\sigma)\cos(\gamma) - D\cos(\sigma)\sin(\gamma), \end{aligned} \tag{5}$$

were a_r is the radial acceleration. Since $\sin(\gamma) = v_z/V \approx 0.1$ and $D < L$, we neglect the drag components in eq. 5, and by dividing the remaining terms we estimate $\sigma \approx \arctan(a_r/g)$. We obtain σ in each timestep by calculating a_r from the data, and then averaging σ over each trajectory. To validate the estimation the average σ we tested this calculation on the simulation results, where σ is known. On average, our estimation was accurate within 7%.

To approximate the wind velocity in the data, we calculated the average velocity in the xy plane during an entire

thermalling trajectory. This approximation neglects the effect of thermal leaning[46], yet it provides a wind velocity distribution similar to the distribution measured on the same dataset using the RAMS atmospheric model[1]. we use the estimated wind velocity to choose the data trajectories for comparison with the agent's trajectories. Estimation of the local thermalling radius is described in Methods Section IV F.

F. Estimating thermalling radius from trajectories

We calculated the local thermalling radius of a soaring-flight trajectory of either the RL-agents or vultures. First, we projected the 3D trajectory points onto the 2D *xy* plain. The, we found the points where the trajectory crosses through $\theta = \pm 180°$ (flying with the wind, type A points), and $\theta = \pm 0$ (flying against the wind, type B points). We sorted these points according to their time-label and used them to form interlacing triangles of points ABA and BAB. For the altitude in each triangle is our estimate for the local thermalling diameter: in an ABA triangle we use the altitude from the B vertex, and in an BAB triangle we use the altitude from an A vertex.

ACKNOWLEDGEMENT

This study was supported by the Center for Interdisciplinary Data Science Research (CIDR) of the Hebrew University of Jerusalem, Israel (joint grant to TB and RN). The vulture study was funded by the US-Israel Binational Science Foundation (BSF 255⁄2008) and the special BSF Multiplier Grant Award from the Rosalinde and Arthur Gilbert Foundation (to RN and Wayne Getz). We also acknowledge financial support from the Adelina and Massimo Della Pergola Chair of Life Sciences and the Minerva Center for Movement Ecology (to RN). Finally, we thank Walter Nesser for his ecological insight and for taking us paragliding.

REFERENCES

[1] R. Harel, N. Horvitz, and R. Nathan, "Adult vultures outperform juveniles in challenging thermal soaring conditions," Scientific reports **6**, 1–8 (2016).

[2] E. Nourani and N. M. Yamaguchi, "The effects of atmospheric currents on the migratory behavior of soaring birds: a review," Ornithological Science **16**, 5–15 (2017).

[3] H. J. Williams, E. Shepard, M. D. Holton, P. A. E. Alarcón, R. Wilson, and S. A. Lambertucci, "Physical limits of flight performance in the heaviest soaring bird," Proceedings of the National Academy of Sciences **117**, 17884–17890 (2020).

[4] H. Reichmann, *Cross-Country Soaring* (homson Publications, Santa Monica, CA, 1978).

[5] Z. Ákos, M. Nagy, and T. Vicsek, "Comparing bird and human soaring strategies," Proceedings of the National Academy of Sciences **105**, 4139–4143 (2008).

[6] Z. Ákos, M. Nagy, S. Leven, and T. Vicsek, "Thermal soaring flight of birds and unmanned aerial vehicles," Bioinspiration & biomimetics **5**, 045003 (2010).

[7] S. C. Beeler, D. D. Moerder, and D. E. Cox, "A flight dynamics model for a small glider in ambient winds," Tech. Rep. (2003).

[8] G. Reddy, J. Wong-Ng, A. Celani, T. J. Sejnowski, and M. Vergassola, "Glider soaring via reinforcement learning in the field," Nature **562**, 236–239 (2018).

[9] S. Notter, F. Schimpf, G. Müller, and W. Fichter, "Hierarchical reinforcement learning approach for autonomous cross-country soaring," Journal of Guidance, Control, and Dynamics **46**, 114–126 (2023).

[10] S. Notter, C. Gall, G. Müller, A. Ahmad, and W. Fichter, "Deep reinforcement learning approach for integrated updraft mapping and exploitation," Journal of Guidance, Control, and Dynamics **46**, 1997–2004 (2023).

[11] M. Allen, "Autonomous soaring for improved endurance of a small uninhabitated air vehicle," in *43rd AIAA Aerospace Sciences Meeting and Exhibit* (2005) p. 1025.

[12] M. J. Allen and V. Lin, "Guidance and control of an autonomous soaring uav," Tech. Rep. (2007).

[13] D. Edwards, "Implementation details and flight test results of an autonomous soaring controller," in *AIAA Guidance, Navigation and Control Conference and Exhibit* (2008) p. 7244.

[14] K. Andersson, I. Kaminer, V. Dobrokhodov, and V. Cichella, "Thermal centering control for autonomous soaring; stability analysis and flight test results," Journal of Guidance, Control, and Dynamics **35**, 963–975 (2012).

[15] N. T. Depenbusch, J. J. Bird, and J. W. Langelaan, "The autosoar autonomous soaring aircraft part 2: Hardware implementation and flight results," Journal of Field Robotics **35**, 435–458 (2018).

[16] N. T. Depenbusch, J. J. Bird, and J. W. Langelaan, "The autosoar autonomous soaring aircraft, part 1: Autonomy algorithms," Journal of Field Robotics **35**, 868–889 (2018).

[17] J. Rosales, L. Sun, L. R. G. Carrillo, and A. Gross, "Evaluation of an autonomous soaring algorithm for survey missions in simulation environment," IEEE Transactions on Aerospace and Electronic Systems (2024).

[18] S. Guo, X. Zhang, Y. Zheng, and Y. Du, "An autonomous path planning model for unmanned ships based on deep reinforcement learning," Sensors **20**, 426 (2020).

[19] S. Aradi, "Survey of deep reinforcement learning for motion planning of autonomous vehicles," IEEE Transactions on Intelligent Transportation Systems **23**, 740–759 (2020).

[20] Y. Li, "Deep reinforcement learning: Opportunities and challenges," arXiv preprint arXiv:2202.11296 (2022).

[21] E. Bøhn, E. M. Coates, S. Moe, and T. A. Johansen, "Deep reinforcement learning attitude control of fixed-wing uavs using proximal policy optimization," in *2019 International Conference on Unmanned Aircraft Systems (ICUAS)* (IEEE, 2019) pp. 523–533.

[22] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, *et al.*, "Mastering the game of go without human knowledge," nature **550**, 354–359 (2017).

[23] G. Reddy, A. Celani, T. J. Sejnowski, and M. Vergassola, "Learning to soar in turbulent environments," Proceedings of the National Academy of Sciences **113**, E4877–E4884 (2016).

[24] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, *et al.*, "Human-level control through deep reinforcement learning," nature **518**, 529–533 (2015).

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

[26] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," arXiv preprint arXiv:1509.02971 (2015).

[27] K. Arulkumaran, M. P. Deisenroth, M. Brundage, and A. A. Bharath, "Deep reinforcement learning: A brief survey," IEEE Signal Processing Magazine **34**, 26–38 (2017).

[28] G. Novati, L. Mahadevan, and P. Koumoutsakos, "Controlled gliding and perching through deep-reinforcement-learning," Physical Review Fluids **4**, 093902 (2019).

[29] S. Notter, P. Gross, P. Schrapel, and W. Fichter, "Multiple thermal updraft estimation and observability analysis," Journal of Guidance, Control, and Dynamics **43**, 490–503 (2020).

[30] Y. Cui, D. Yan, and Z. Wan, "Study on the glider soaring strategy in random location thermal updraft via reinforcement learning," Aerospace **10**, 834 (2023).

[31] G. Reddy, "A reinforcement-based mechanism for discontinuous learning," Proceedings of the National Academy of Sciences **119**, e2215352119 (2022).

[32] A. Alharin, T.-N. Doan, and M. Sartipi, "Reinforcement learning interpretation methods: A survey," IEEE Access **8**, 171058–171077 (2020).

[33] T. Zahavy, N. Ben-Zrihem, and S. Mannor, "Graying the black box: Understanding dqns," in *International conference on machine learning*

(PMLR, 2016) pp. 1899–1908.
[34]A. Goldshtein, S. Akrish, R. Giryes, and Y. Yovel, "An artificial neural network explains how bats might use vision for navigation," Communications biology **5**, 1–8 (2022).
[35]H. J. Williams, O. Duriez, M. D. Holton, G. Dell'Omo, R. P. Wilson, and E. L. Shepard, "Vultures respond to challenges of near-ground thermal soaring by varying bank angle," Journal of Experimental Biology **221**, jeb174995 (2018).
[36]M. A. Lewis, W. F. Fagan, M. Auger-Methe, J. Frair, J. M. Fryxell, C. Gros, E. Gurarie, S. D. Healy, and J. A. Merkle, "Learning and animal movement," Frontiers in Ecology and Evolution **9**, 681704 (2021).
[37]A. Goldshtein, M. Handel, O. Eitan, A. Bonstein, T. Shaler, S. Collet, S. Greif, R. A. Medellín, Y. Emek, A. Korman, *et al.*, "Reinforcement learning enables resource partitioning in foraging bats," Current Biology **30**, 4096–4102 (2020).
[38]B. Cellini, B. Boyacioğlu, and F. Van Breugel, "Empirical individual state observability," in *2023 62nd IEEE Conference on Decision and Control (CDC)* (IEEE, 2023) pp. 8450–8456.
[39]A. Silva, M. Gombolay, T. Killian, I. Jimenez, and S.-H. Son, "Optimization methods for interpretable differentiable decision trees applied to reinforcement learning," in *International conference on artificial intelligence and statistics* (PMLR, 2020) pp. 1855–1865.
[40]Y. Coppens, K. Efthymiadis, T. Lenaerts, A. Nowé, T. Miller, R. Weber, and D. Magazzeni, "Distilling deep reinforcement learning policies in soft decision trees," in *Proceedings of the IJCAI 2019 workshop on explainable artificial intelligence* (2019) pp. 1–6.
[41]G. Liu, O. Schulte, W. Zhu, and Q. Li, "Toward interpretable deep reinforcement learning with linear model u-trees," in *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2018, Dublin, Ireland, September 10–14, 2018, Proceedings, Part II 18* (Springer, 2019) pp. 414–429.
[42]M. Vasic, A. Petrovic, K. Wang, M. Nikolic, R. Singh, and S. Khurshid, "Moët: Interpretable and verifiable reinforcement learning via mixture of expert trees," arXiv preprint arXiv:1906.06717 (2019).
[43]R. Harel, O. Spiegel, W. M. Getz, and R. Nathan, "Social foraging and individual consistency in following behaviour: testing the information centre hypothesis in free-ranging vultures," Proceedings of the Royal Society B: Biological Sciences **284**, 20162654 (2017).
[44]J. Gedeon, "Dynamic analysis of dolphin-style thermal cross-country flight: Part ii," Technical Soaring **3**, 17–34 (1973).
[45]D. Lenschow and P. Stephens, "The role of thermals in the convective boundary layer," Boundary-Layer Meteorology **19**, 509–532 (1980).
[46]R. Bencatel, J. T. de Sousa, and A. Girard, "Atmospheric flow field models applicable for aircraft endurance extension," Progress in Aerospace Sciences **61**, 1–25 (2013).
[47]M. Allen, "Updraft model for development of autonomous soaring uninhabited air vehicles," in *44th AIAA Aerospace Sciences Meeting and Exhibit* (2006) p. 1510.
[48]R. Harel and R. Nathan, "The characteristic time-scale of perceived information for decision-making: Departure from thermal columns in soaring birds," Functional ecology **32**, 2065–2072 (2018).
[49]A. D. Laud, *Theory and application of reward shaping in reinforcement learning* (University of Illinois at Urbana-Champaign, 2004).
[50]A. Raffin, A. Hill, A. Gleave, A. Kanervisto, M. Ernestus, and N. Dormann, "Stable-baselines3: Reliable reinforcement learning implementations," The Journal of Machine Learning Research **22**, 12348–12355 (2021).
[51]S. Falkner, A. Klein, and F. Hutter, "Bohb: Robust and efficient hyperparameter optimization at scale," in *International Conference on Machine Learning* (PMLR, 2018) pp. 1437–1446.
[52]J. A. Hartigan and M. A. Wong, "Algorithm as 136: A k-means clustering algorithm," Journal of the royal statistical society. series c (applied statistics) **28**, 100–108 (1979).
[53]V. Satopaa, J. Albrecht, D. Irwin, and B. Raghavan, "Finding a" kneedle" in a haystack: Detecting knee points in system behavior," in *2011 31st international conference on distributed computing systems workshops* (IEEE, 2011) pp. 166–171.
[54]R. Nainggolan, R. Perangin-angin, E. Simarmata, and A. F. Tarigan, "Improved the performance of the k-means cluster using the sum of squared error (sse) optimized by using the elbow method," in *Journal of Physics: Conference Series*, Vol. 1361 (IOP Publishing, 2019) p. 012015.

Appendices and Supplementary Information

Revealing principles of autonomous thermal soaring in windy conditions using vulture-inspired deep reinforcement-learning

Yoav Flato, Roi Harel, Aviv Tamar, Ran Nathan, and Tsevi Beatus

Appendix A: Aerodynamic force model and glider parameters

Following Beeler *et al.*[7], the environment is based on a 3 degrees-of-freedom (DOF) simulation of a point-mass glider governed by the equations of motion in Eq. 2. The aerodynamic forces: lift L, drag D and side force C, are:

$$L(V,\alpha) = \frac{1}{2}\rho S C_L(\alpha) V^2 \tag{A1}$$

$$D(V,\alpha,\beta) = \frac{1}{2}\rho S C_D(\alpha,\beta) V^2 \tag{A2}$$

$$C(V,\beta) = \frac{1}{2}\rho S C_C(\beta) V^2, \tag{A3}$$

where the forces are functions of the velocity V, the aerodynamic coefficients $C_L(\alpha), C_D(\alpha,\beta), C_C(\beta)$, the wing surface area S, the air density ρ. The force coefficients are functions of the angle-of-attack α and the sideslip angle β, and depend also on the glider's mass, wingspan and wing aspect ratio (Table A.1). Following Beeler *et al.*[7], To account for the effects of wind on the glider, the full aerodynamic model calculates the forces in the wind frame-of-reference and then transforms these forces to the world frame-of-reference for solving the equations of motion.

Symbol	Name	Value	Units
AR	aspect ratio	16	–
l	wingspan	2.5	m
S	wing area	0.39	m^2
m	mass	5	kg

TABLE A.1. **Glider parameters.** The glider parameters used in the simulation are based on Beeler *et al.*[7].

Appendix B: An upper bound for the thermal climb rate

One way to evaluate a policy of an agent in our system, would be to compare the policy climb rate performance (mean v_z) in a given environment to the maximum climb rate possible under the same conditions. Hence, we estimated an upper bound for the climb rate, as described below.

We assume that an optimal trajectory has constant rotation radius in the wind frame of reference and, therefore, constant control parameters (σ,α,β) in that frame. Note that to keep these control parameters constant in the wind frame of reference (under any wind, horizontal or vertical), an agent would need to modulate them in the body frame of reference. Further, we assume β=0 as used in our nominal agent.

First, for each pair of (σ,α) we analytically solve the equations of motion (section IV B) in steady state and find the values of V_{ss}, γ_{ss} and R_{ss}, where the latter is the trajectory radius. Second, we define, d, the distance between the center of the trajectory and the thermal center, and use numerical optimization to find (σ,α,d) that result in maximum climb rate v_z in the world frame of reference. These two stages are detailed below.

In steady state of eq. (2) the velocity and glide angle are constant:

$$\begin{aligned}\dot{V} &= 0 = -\frac{1}{m}D(V_{ss},\alpha,\beta) - g\sin\gamma_{ss}\\ \dot{\gamma} &= 0 = \frac{1}{mV}\Big(L(V_{ss},\alpha)\cos\sigma + C(V_{ss},\beta)\sin\sigma\Big) - \frac{g}{V_{ss}}\cos\gamma_{ss}\end{aligned} \tag{B1}$$

Assuming β=0, we get:

$$\begin{aligned}D(V_{ss},\alpha,0) &= -mg\sin\gamma_{ss}\\ L(V_{ss},\alpha)\cos\sigma &= mg\cos\gamma_{ss}\end{aligned} \tag{B2}$$

By dividing these two equations and rearranging, we obtain an expression for the steady state glide (pitch) angle:

$$\gamma_{ss}(\sigma,\alpha) = \arctan\left(-\frac{C_D(\alpha,0)}{C_L(\alpha)cos(\sigma)}\right). \tag{B3}$$

We then substitute the definition of $D(V,\alpha,\beta)$ from eq. (A3) into eq. (B2) to get V_{ss}:

$$V_{ss}(\sigma,\alpha) = \sqrt{-\frac{2mg\sin(\gamma_{ss}(\alpha,\sigma))}{C_D(\alpha,0)S\rho}}. \tag{B4}$$

In this steady state of circular motion in the wind frame of reference, the radial force due to the tilted lift vector is equal to the radial acceleration: $L(V_{ss},\alpha)\sin(\sigma) = mV_{ss}^2/R_{ss}$. To find R_{ss} from this equation, we substitute $L(V_{ss},\alpha)$ from eq. (A3):

$$R_{ss}(\sigma,\alpha) = \frac{2m}{C_L(\alpha)S\rho\sin(\sigma)}. \tag{B5}$$

Note that because $L{\sim}V^2$, then R_{ss} is independent of V_{ss}. The steady state climb rate in the wind frame is:

$$v_z^{\text{wind}}(\sigma,\alpha) = V_{ss}(\sigma,\alpha)\sin(\gamma_{ss}(\sigma,\alpha)) \tag{B6}$$

Next, for each combination of (σ,α,d), we calculate the average climb rate v_z in the world frame of reference. This climb rate consists of two terms:

$$v_z(\sigma,\alpha,d) = v_z^{\text{wind}}(\sigma,\alpha) + v_z^{\text{thermal}}(\sigma,\alpha,d). \tag{B7}$$

The first term is given in eq. (B6) at steady state, and the second term is the vertical velocity of the wind frame itself due to the thermal updraft averaged across the agent's trajectory. Each pair of control parameters (σ, α) defines a value of R_{ss} (eq. (B5)). Together with d, and relying on the radial symmetry of the thermal, we numerically calculate the mean thermal updraft along one rotation period $v_z^{\text{thermal}}(\sigma, \alpha, d)$:

$$v_z^{\text{thermal}}(\sigma, \alpha, d) = \frac{1}{T} \int_0^T w(r, z) dt, \tag{B8}$$

where T is the rotation period, $w(r, z)$ is the thermal updraft velocity (eq. (3)), $r = r(d, R_{ss}, t)$ is the glider distance from the thermal center, and z is the glider altitude. We assume constant altitude of 500m because we characterize the optimal updraft at this altitude and because the thermal updraft profile does not change much with altitude (Figs. 1c,d).

Finally, using numerical optimization on (σ, α, d), we find that under our simplifying assumptions the upper bound the the climb rate in the world frame of reference is obtained for $\alpha \approx 13.3°$, $\sigma \approx 31.5°$, $d = 0$ (*i.e.*, flying directly around the thermal center) and $R_{ss} = 28.2$ m. The upper bound is:

$$v_z^{\text{optimal}} = 0.72\,\text{m/sec} \tag{B9}$$

Appendix C: Hyper-parameters search

We used a Bayesian method for hyper-parameter search[51] to optimize the following hyper-parameters of our RL system: the learning algorithm (DDPG[26] or PPO[25]), learning rate, and the NN architecture (hidden layer size and number of hidden layers) for both actor and critic. Each combination of hyper-parameters, was tested on a model problem of thermal soaring under u=1 m/sec given 10^6 time steps (~5,000 thermals) to converge.

The optimization results are summarized in Fig. C.1. The chosen algorithm was DDPG with learning rate of 10^{-3}. The actor NN architecture was: 2 hidden layers of 200 neurons each and ReLU activation, and tanh activation for the output layer. The critic NN architecture was the same, without an activation function on its (scalar) output. In this parameter scan, the PPO algorithm achieved similar results but with larger networks of 500 neurons in 2 hidden layers.

Appendix D: Clustering analysis of agents with a reduced state representation

To verify that the activity distribution of the NN clusters in θ was not a trivial outcome of having θ as an implicit state variable, we trained an agent whose state did not include θ. Instead, the state included $\{V, v_z, \sigma\}$ and the agent was trained on one value of u=1 m/sec. The agent with this reduced state-representation achieved stable soaring and exhibited similar clustering characteristics that evolved to be θ-dependent with the agent's 'age' (Fig. D.1).

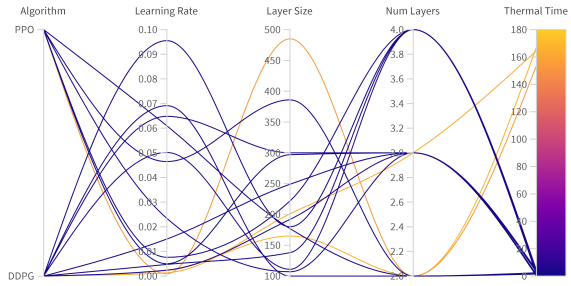

FIG. C.1. **Bayesian Hyper-Parameter Search Results.** This scheme includes of five columns: four for each hyperparameter and the rightmost one for the agent's performance in terms of its average thermal time. Each continuous line represents a single run of 10^6 timesteps, indicating the values of the four hyperparameters and gliding performance. The lines are color-coded by gliding performance.

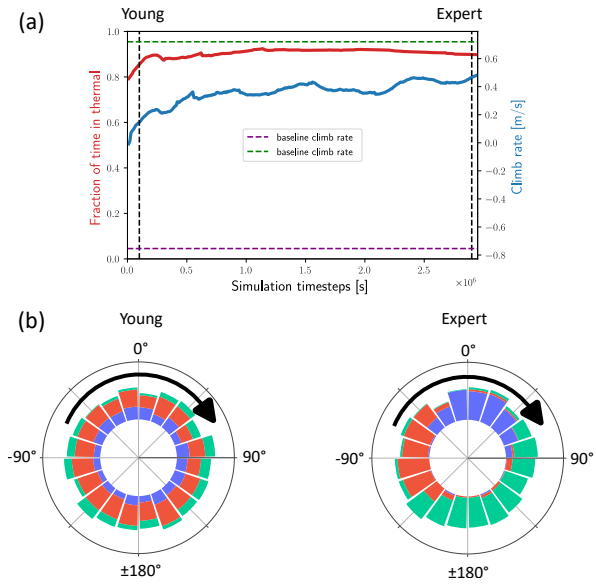

FIG. D.1. **NN neural clusters during the learning process of reduced-state agent.** **(a)** Performances values during the learning process. The high performance values observed in the beginning of the run are attributed to the fact that this learning process started from pre-trained agents. **(b)** Distribution of the clusters of each agent as a function of θ.

Appendix E: Supplementary material

Supplementary videos. Representative trajectories of the nominal agent soaring in a thermal under horizontal winds of 2.4 m/sec (Video 1) and 3 m/sec (Video 2).