Supplementary Information for

Efficient collective swimming by harnessing vortices through deep reinforcement learning

Siddhartha Verma, Guido Novati, Petros Koumoutsakos

Petros Koumoutsakos.
E-mail: petros@ethz.ch

This PDF file includes:

Other supplementary materials for this manuscript include the following:

Movies S1 to S7

Supporting Information Text

Methods

Simulation details. The simulations presented here are based on the incompressible Navier-Stokes (NS) equations:

$$\nabla \cdot \mathbf{u} = 0$$
$$\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} = -\frac{\nabla P}{\rho_f} + \nu \nabla^2 \mathbf{u} + \lambda \chi (\mathbf{u}_s - \mathbf{u})$$

Each swimmer is represented on the computational grid via the characteristic function χ, and interacts with the fluid by means of the penalty[1] term $\lambda\chi(\mathbf{u}_s - \mathbf{u})$. Here, $\mathbf{u}_s$ denotes the swimmer's combined translational, rotational, and deformation velocity, whereas $\mathbf{u}$ and ν correspond to the fluid velocity and viscosity, respectively. P represents the pressure, and the fluid density is denoted by ρ_f.

The vorticity form of the NS equations was used for the two-dimensional simulations, with $\lambda = 1e6$. A wavelet adaptive grid[2] with an effective resolution of 4096^2 points was used to discretize a unit square domain. A lower effective resolution of 1024^2 points was used for the training-simulations to minimize computational cost. We have determined in previous tests that this resolution provides a reasonable balance between speed and accuracy.[3] The pressure-Poisson equation ($\nabla^2 P = -\rho_f \left(\nabla \mathbf{u}^T : \nabla \mathbf{u}\right) + \rho_f \lambda \nabla \cdot (\chi (\mathbf{u}_s - \mathbf{u}))$), necessary for estimating the distribution of flow-induced forces on the swimmers' bodies, was solved using the Fast Multipole Method with free-space boundary conditions.[3]

The three-dimensional simulations employed the pressure-projection method for solving the NS equations.[4] The simulations were parallelized via the CUBISM framework,[5] and used a uniform grid consisting of $2048 \times 1024 \times 256$ points in a domain of size $1 \times 0.5 \times 0.125$, with penalty parameter $\lambda = 1e5$. Further grid-refinement by $1.5\times$ in all three directions, and increasing the penalty parameter to $1e6$ resulted in no discernible change in the swimmer's speed. Thus, the lower grid resolution was selected to keep computational cost manageable. The CFL (Courant-Friedrichs-Lewy) number was constrained to be less than 0.1, resulting in approximately 2500 time steps per tail-beat period. The non-divergence-free deformation of the self-propelled swimmers was incorporated into the pressure-Poisson equation as follows:

$$\nabla^2 P = \frac{\rho_f}{\Delta t} \left(\nabla \cdot \mathbf{u}^\star - \chi \nabla \cdot \mathbf{u}_s\right), \tag{1}$$

where $\mathbf{u}^\star$ represents the intermediate velocity from the convection-diffusion-penalization fractional steps. Equation 1 was solved using a distributed Fast Fourier Transform library (AccFFT[6]). To prevent a periodic recycling of the outflow, the velocity field was smoothly truncated to zero as it approached the outflow boundary. We ensured that periodicity and velocity smoothing do not impact the results presented, by running simulations with a domain enlarged in all three spatial directions.

Flow-induced forces, and energetics variables. The pressure-induced and viscous forces acting on the swimmers are computed as follows:[3]

$$\mathbf{dF}_P = -P\mathbf{n}\, dS \tag{2}$$
$$\mathbf{dF}_\nu = 2\mu \mathbf{D} \cdot \mathbf{n}\, dS \tag{3}$$

Here, P represents the pressure acting on the swimmer's surface, $\mathbf{D} = \left(\nabla \mathbf{u} + \nabla \mathbf{u}^T\right)/2$ is the strain-rate tensor on the surface, and dS denotes the infinitesimal surface area. Since self-propelled swimmers generate zero net average thrust (and drag) during steady swimming, we determine the instantaneous thrust as follows:

$$\text{Thrust} = \frac{1}{2\|\mathbf{u}\|} \iint \left(\mathbf{u} \cdot \mathbf{dF} + |\mathbf{u} \cdot \mathbf{dF}|\right), \tag{4}$$

where $\mathbf{dF} = \mathbf{dF}_P + \mathbf{dF}_\nu$. Similarly, the instantaneous drag may be determined as:

$$\text{Drag} = \frac{1}{2\|\mathbf{u}\|} \iint \left(\mathbf{u} \cdot \mathbf{dF} - |\mathbf{u} \cdot \mathbf{dF}|\right) \tag{5}$$

Using these quantities, the thrust-, drag-, and deformation-power are computed as:

$$P_{Thrust} = \text{Thrust} \cdot \|\mathbf{u}\| \tag{6}$$
$$P_{Drag} = -\text{Drag} \cdot \|\mathbf{u}\| \tag{7}$$
$$P_{Def} = -\iint \mathbf{u}_{Def} \cdot \mathbf{dF} \tag{8}$$

where $\mathbf{u}_{Def}$ represents the deformation-velocity of the swimmer's body. The double-integrals in these equations represent surface-integration over the swimmer's body, and yield measurements for time-series analysis. On the other hand, only the integrand is evaluated when surface-distributions of thrust-, drag-, or deformation-power are required (as in main Figs. 4c to 4f).

The instantaneous swimming-efficiency is based on a modified form of the Froude efficiency proposed in ref.:[7]

$$\eta = \frac{P_{Thrust}}{P_{Thrust} + \max(P_{Def}, 0)} \tag{9}$$

To compute both η and the Cost of Transport (CoT), we neglect negative values of P_{Def}, which can result from beneficial interactions of the smart-swimmer with the leader's wake:

$$CoT(t) = \frac{\int_{t-T_p}^{t} \max(P_{Def}, 0) dt}{\int_{t-T_p}^{t} \|\mathbf{u}\| dt} \tag{10}$$

This restriction accounts for the fact that the elastically rigid swimmer may not store energy furnished by the flow, and yields a conservative estimate of potential savings in the CoT. We note that percentage-changes in P_{Def}, reported in the main text and the supplementary section, have been computed using this bounded value to avoid overstating any potential benefits.

Swimmer shape and kinematics. The Reynolds number of the self-propelled swimmers is computed as $Re = L^2/(\nu T_p)$. The body-geometry is based on a simplified model of a zebrafish.[8] The half-width of the 2D profile is described as follows:

$$w(s) = \begin{cases} \sqrt{2w_h s - s^2} & 0 \le s < s_b \\ w_h - (w_h - w_t)\left(\dfrac{s - s_b}{s_t - s_b}\right) & s_b \le s < s_t \\ w_t \dfrac{L - s}{L - s_t} & s_t \le s \le L \end{cases} \qquad [11]$$

where s is the arc-length along the midline of the geometry, $L = 0.1$ is the body length, $w_h = s_b = 0.04L$, $s_t = 0.95L$, and $w_t = 0.01L$. For 3D simulations, the geometry is comprised of elliptical cross sections, with the half-width $w(s)$ and half-height $h(s)$ described via cubic B-splines.[8] Six control-points define the half-width: $(s/L, w/L) = [(0.0, 0.0),\ (0.0, 0.089),\ (1/3, 0.017),\ (2/3, 0.016),\ (1.0, 0.013),\ (1.0, 0.0)]$; whereas eight control-points define the half-height: $(s/L, h/L) = [(0.0, 0.0),\ (0.0, 0.055),\ (0.2, 0.068),\ (0.4, 0.076),\ (0.6, 0.064),\ (0.8, 0.0072),\ (1.0, 0.11),\ (1.0, 0.0)]$. The length was set to $L = 0.2$, which keeps the grid-resolution, i.e., the number of points along the fish midline, comparable to the 2D simulations. Body-undulations for both 2D and 3D simulations were generated as a travelling-wave defining the curvature along the midline:

$$k(s, t) = A(s) \sin\left(\frac{2\pi t}{T_p} - \frac{2\pi s}{L}\right) \qquad [12]$$

Here $A(s)$ is the curvature amplitude and varies linearly from $A(0) = 0.82$ to $A(L) = 5.7$.

Reinforcement Learning. Reinforcement learning (RL)[9] is a process by which an agent (in this case, the smart-swimmer) learns to earn rewards through trial-and-error interaction with its environment. At each turn, the agent observes the state of the environment s_n and performs an action a_n, which influences both the transition to the next state s_{n+1} and the reward received r_{n+1}. The agent's goal is to learn the optimal control policy $a_n = \pi^*(s_n)$ which maximises the action value $Q^*(s_n, a_n)$, defined as the sum of discounted future rewards:

$$Q^*(s_n, a_n) = \max_\pi \mathbb{E}\left(r_{n+1} + \gamma r_{n+2} + \gamma^2 r_{n+3} + \ldots \,|\, a_m = \pi(s_m)\ \forall m \in [n+1, \mathcal{T}]\right) \qquad [13]$$

Here, $\mathcal{T}$ denotes the terminal state of a training-simulation, and the discount factor γ is set to 0.9. The optimal action-value function $Q^*(s_n, a_n)$ is a fixed point of the Bellman equation: $Q^*(s_n, a_n) = \mathbb{E}\left[r_{n+1} + \gamma \max_{a'} Q^*(s_{n+1}, a')\right]$.[10] We approximate $Q^*(s_n, a_n)$ using a neural network[11] with weights w_k, which are updated iteratively to minimize the temporal difference error:

$$\mathrm{TD}_{\mathrm{err}} = \mathbb{E}_{s_n, a_n, s_{n+1}}\left[r_{n+1} + \gamma Q(s_{n+1}, a'; \mathrm{w}_-) - Q(s_n, a_n; \mathrm{w}_k)\right] \qquad [14]$$

Here, w_- is a set of target weights, and a' is the best action in state s_{n+1} computed with the current weights ($a' = \arg\max_a Q(s_{n+1}, a; \mathrm{w}_k)$). The target weights w_- are updated towards the current weights as $\mathrm{w}_- \leftarrow (1 - \alpha)\mathrm{w}_- + \alpha \mathrm{w}_k$, where $\alpha = 10^{-4}$ is an under-relaxation factor used to stabilize the algorithm.[11]

States and actions. The six observed-state variables perceived by the learning agent include Δx, Δy, θ, the two most recent actions taken by the agent, and the current tail-beat 'stage' $\mathrm{mod}(t, T_p)/T_p$. The permissible range of the observed-state variables is limited to: $1 \le \Delta x/L \le 3$; $|\Delta y|/L \le 1$ (boundary depicted by R_{end} in SI Appendix Fig. S8); and $|\theta| \le \pi/2$. If the agent exceeds any of these thresholds, the training-simulation terminates and the agent receives a terminal reward $R_{end} = -1$.

The smart-swimmer (or agent) is capable of manoeuvering by actively manipulating the curvature-wave travelling down the body. This is accomplished by linearly superimposing a piecewise function on the baseline curvature $k(s, t)$ (equation 12):

$$k_{\mathrm{Agent}}(s, t) = k(s, t) + A(s) M(t, T_p, s, L) \qquad [15]$$

The curve $M(t, T_p, s, L)$ is composed of 3 distinct segments:

$$M(t, T_p, s, L) = \sum_{j=0}^{2} b_{n-j} \cdot m\left(\frac{t - t_{n-j}}{T_p} - \frac{s}{L}\right) \qquad [16]$$

The curve m is a clamped cubic spline with $m(0) = m'(0) = 0$, $m(1/2) = m'(1/2) = 0$, and $m(1/4) = 1$, $m'(1/4) = 0$. t_n represents the time-instance when action a_n is taken, whereas b_n represents the corresponding control-amplitude, which may take five discrete values: 0, ± 0.25, and ± 0.5.

Neural network architecture. One of the assumptions in RL is that the transition probability to a new state s_{n+1} is independent of the previous transitions, given s_n and a_n, i.e.,:

$$p(s_{n+1} \,|\, s_n, a_n) = p(s_{n+1} \,|\, s_n, a_n, \ldots, s_0, a_0) \qquad [17]$$

This assumption is invalidated whenever the agent has a limited perception of the environment. In most realistic cases the agent receives an observation o_n rather than the complete state of the environment s_n. Therefore, past observations carry information relevant for future transitions (i.e., $p(o_{n+1} \,|\, o_n, a_n) \ne p(o_{n+1} \,|\, o_n, a_n, \ldots, o_0, a_0)$), and should be taken into account in order to make optimal decisions. This operation can be approximated by a Recurrent Neural Network (RNN), which can learn to compute and remember important features in past observations. In this work we approximate the action-value function with a LSTM-RNN[12] composed of three layers of 24 fully connected LSTM cells each, and terminating in a linear layer (SI Appendix Fig. S3). The last layer computes a vector of action-values $\mathbf{q}_n = Q(o_n; y_{n-1}, \mathrm{w}_k)$ with one component $q_n^{(a)}$ for each possible action a available to the agent (y_{n-1} represents the activation of the network at the previous turn).

Training procedure. During training, both the leader and the follower (learning agent) start from rest. The leader swims steadily along a straight line, whereas the follower manoeuvers according to the actions supplied to it. Multiple independent simulations run simultaneously, with each of these sending the current observed-state o_n of the agent to a central processor, and in turn receiving the next action a_n to be performed. The central processor computes a_n using an ϵ-greedy policy (with ϵ gradually annealed from 1 to 0.1) from the most recently updated Q function. Once a training-simulation reaches a terminal state (e.g., the follower hits the boundary labelled R_{end} in SI Appendix Fig. S8), all the messages exchanged between the simulation and the central processor are appended to a training set of sequences $\mathcal{R}$. In the meantime, the network is continually updated by sampling B sequences from the set $\mathcal{R}$, according to algorithm 1. The batch gradient g is computed with back propagation through time (BPTT).[14] The network weights are then updated with the Adam

Algorithm 1: Asynchronous recurrent DQN algorithm.

```
initialize network $\mathrm{w}_0$ and target network $\mathrm{w}_- = \mathrm{w}_0$;
initialize set of transition sequences $\mathcal{R} = \emptyset$;
repeat
    $N \leftarrow 0$;
    sample batch of $B$ sequences from $\mathcal{R}$;
    for sequence $j \in [1, \dots, B]$ do
        $[\mathbf{q}_{j,0}, y_{j,0}] = Q(o_{j,0}; \emptyset, \mathrm{w}_k)$;
        for turns $n \in [0, \dots, \mathcal{T}_j - 1]$ do
            $[\mathbf{q}_{j,n+1}, y_{j,n+1}] = Q(o_{j,n+1}; y_{j,n}, \mathrm{w}_k)$;
            $[\tilde{\mathbf{q}}_{j,n+1}, \tilde{y}_{j,n+1}] = Q(o_{j,n+1}; y_{j,n}, \mathrm{w}_-)$;
            $a' = \arg\max_a \left[ q^{(a)}_{j,n+1} \right]$;
            if $s_{j,n+1}$ is terminal then
                $e_{j,n} = r_{j,n+1} - q^{(a_n)}_{j,n}$;
            else
                $e_{j,n} = r_{j,n+1} + \gamma \tilde{q}^{(a')}_{j,n+1} - q^{(a_n)}_{j,n}$;
            end
            $N \leftarrow N + 1$;
        end
    end
    perform BPTT: $g = \frac{1}{N} \sum_j \sum_n e_{j,n} \nabla_{\mathrm{w}} q^{(a_n)}_{j,n}$;
    update weights $\mathrm{w}_{k+1}$ by passing $g$ to the Adam algorithm[13];
    update target network: $\mathrm{w}_- \leftarrow (1-\alpha)\mathrm{w}_- + \alpha \mathrm{w}_{k+1}$;
    $k \leftarrow k + 1$;
until $Q(o, a; \mathrm{w}_k) = Q^*(o, a)$;
```

stochastic optimization algorithm.[13]

A total of 1200 forward simulations were used during the training procedure, which corresponds to approximately 46000 transitions (action-decisions) by the learning agent. To determine the convergence of network-fitting, we inspected the histogram distribution of the follower's preferred Δx position (similar to main Fig. 2e) during the final and the penultimate 10000 transitions. We observed that the distribution did not change noticeably towards the end of training, which indicates that the RL algorithm has arrived close to a local minimum. Running additional simulations would not alter the histogram distribution appreciably, and any incremental improvements would incur too large a computational cost to be justifiable.

Proportional-Integral feedback controller. The PI controller modulates the 3D follower's body-kinematics, which allows it to maintain a specific position (x_{tgt}, y_{tgt}, z_{tgt}) relative to the leader:

$$k(s,t) = \alpha(t) A(s) \left[\sin\left(\frac{2\pi t}{T_p} - \frac{2\pi s}{L} \right) + \beta(t) \right] \tag{18}$$

The factor $\alpha(t)$ modifies the undulation envelope, and controls the acceleration or deceleration of the follower based on its streamwise distance from the target position:

$$\alpha(t) = 1 + f_1 \left(\frac{x - x_{tgt}}{L} \right) \tag{19}$$

The term $\beta(t)$ adds a baseline curvature to the follower's midline to correct for lateral deviations:

$$\beta(t) = \frac{y_{tgt} - y}{L} \left(f_2 |\theta| + f_3 |\hat{\theta}| \right) \tag{20}$$

Here, θ represents the follower's yaw angle about the z-axis, and $\hat{\theta}$ is its exponential moving average: $\hat{\theta}_{t+1} = \frac{1-\Delta t}{T_p} \hat{\theta}_t + \frac{\Delta t}{T_p} \theta$. The swimmers' z-positions remain fixed at z_{tgt}, as out-of-plane motion is not permitted. The controller-coefficients were selected to have a minimal impact on regular swimming kinematics, which allows for a direct comparison of the follower's efficiency to that of the leader:

$$f_1 = 1 \tag{21}$$

$$f_2 = \max(0, 50\ \mathrm{sign}(\theta \cdot (y_{tgt} - y))) \tag{22}$$

$$f_3 = \max(0, 20\ \mathrm{sign}(\hat{\theta} \cdot (y_{tgt} - y))) \tag{23}$$

Supplementary Text, Figures, Tables, and Movies

Body-deformation during autonomous manoeuvres. The extent of body-bending that swimmers IS_η and IS_d undergo when manoeuvring is compared quantitatively in SI Appendix Fig. S1. A qualitative comparison was presented in main Fig. 2f. We observe that the body-deformation of IS_d is noticeably higher than that of a steady swimmer (with relative curvature 1), which implies a tendency to take aggressive turns. The deformation for swimmer IS_η is markedly lower, which plays an instrumental role in reducing the power required for undulating the body against flow-induced forces.

Comparison of four different swimmers. The performance metrics for four different swimmers are compared in SI Appendix Fig. S2. Interacting swimmer IS_d occasionally attains higher speed than IS_η (SI Appendix Fig. S2a), but at the cost of much higher energy expenditure (SI Appendix Fig. S2c and Table S1). Moreover, the speeds of solitary swimmers SS_η and SS_d are lower than those of either interacting swimmer (IS_η and IS_d), which suggests that wake-interactions may benefit a follower in some aspects, regardless of the goal being pursued. However, we stress that while interacting with a leader's wake appropriately may yield a benefit compared to the energy requirements of a steady solitary swimmer (e.g., IS_η - SI Appendix Fig. S2c), this may not be the case if the reinforcement learning reward does not account for energy usage. Both swimmers IS_d and SS_d have higher energetic costs of swimming compared to a steady solitary fish (Fig. S2c), which demonstrates that following a leader indiscriminately can indeed be disadvantageous. In SI Appendix Fig. S2d, P_{Def} attains negative values only for IS_η, which is indicative of maximum benefit extracted from flow-induced forces. Both IS_d and SS_d are capable of generating significantly higher thrust-power than IS_η, but suffer from larger deformation-power, and consequently, lower swimming-efficiency. Comparing the columns for IS_η and SS_η in Table S1, we note that interacting with a preceding wake has a measurable impact on swimming-performance; IS_η is approximately 32% more efficient than SS_η, spends 36% less energy per unit distance travelled, requires 29% less power for body-undulations, and generates 52% higher thrust-power. Wake-interactions may yield certain benefits even for the swimmer actively minimizing lateral displacement from the leader, primarily by increasing thrust-power, but at the penalty of higher energetic-cost for body-deformation, as can be surmised by comparing the data for IS_d and SS_d in SI Appendix Table S1. This observation further confirms that interacting with unsteady wakes may not prove to be beneficial overall, if the swimming-kinematics do not account for energetic considerations.

Uncovering underlying time-dependencies. While it is relatively straightforward to maintain a particular tandem formation via feedback control (when the follower strays too far to one side, a feedback controller can relay instructions to veer in the opposite direction), the same is not true for maximizing swimming-efficiency. It is difficult to formulate a simple set of a-priori rules for maximizing efficiency, especially in dynamically evolving conditions. This happens because: 1) the swimmer perceives only a limited representation of its environment (main Fig. 2a); and 2) there may be measurable delay between an action and its impact on the reward received over the long term. These traits make deep RL ideal for determining the optimal policy when maximizing swimming-efficiency, especially when augmented with recurrent neural networks (SI Appendix Fig. S3). These network architectures are adept at discovering and exploiting long-term time-dependencies. We remark that neither standard optimisation, nor optimal control[10] techniques are suitable for use in the current problem, both due to the need for adaptive control, and due to the unavailability of simplified sets of equations describing the system's response. Moreover, optimal-control algorithms evaluate multiple forward simulations at every decision-making step, which is decidedly impractical in the current study given the large computational cost of the forward Navier-Stokes simulations.

The advantage of using a Recurrent Neural Network (RNN). To illustrate the advantage of using a deep recurrent network, we compare the performance of a smart-swimmer trained to minimize lateral deviations (Δy) from a leader using two distinct neural network architectures: a Feedforward Neural Network (FNN) similar to the one used in our previous study;[15] and the more sophisticated deep Recurrent Neural Network (RNN) shown in Fig. S3. Using SI Appendix Fig. S4a, we observe that the FNN-trained smart-follower is unable to achieve its goal of maintaining $\Delta y = 0$ as rigorously as the RNN-trained follower, which clearly demonstrates the superior capability of the RNN. Moreover, in its attempt to maintain $\Delta y = 0$ rigorously, the RNN-trained swimmer executes severe turns (main Fig. 2f), which lead to an increase in its energy consumption (higher CoT in Fig. S4b). To explain the comparative energetic benefit observed by the FNN-trained swimmer (even though its reward does not account for energetic considerations), we note that it almost always settles close to the 'attractor point' $\Delta x = 2.2L$, where the head-motion is synchronised well with the wake flow. This leads to energetic gains for the FNN-based swimmer, although its primary objective of maintaining $\Delta y = 0$ is not achieved satisfactorily. We remark that similar migrations of a follower toward the favourable attractor point are observed, even when employing a feedback controller to attempt to hold position at an unfavourable location in the wake. We speculate that this may portend the existence of stability points throughout schooling formations, where minimal control-effort may yield large energetic gains.

Flow-interactions at the instant of minimum swimming-efficiency. The instant when swimmer IS_η attains the lowest efficiency during each half-period ($\eta_{min}(D)$ in main Fig. 3a) is examined in SI Appendix Fig. S5. The mean P_{Def} curve is mostly positive on both the lower and upper surfaces, with large positive peaks generated by interaction with the wake- and lifted-vortices. This increase in effort is not offset sufficiently by an increase in P_{Thrust}, resulting in low swimming-efficiency. Compared to the instance of maximum efficiency (main Fig. 4), increased effort is required in the head region, along with an increase in thrust-production by the tail section $s > 0.7L$.

Slight deviations impact performance. To examine the impact of small deviations in IS_η's trajectory on its performance, we compare two different time-instances (at the same tail-beat stage) in SI Appendix Fig. S6. At $t \approx 26.5$, IS_η deviates slightly to the left of its steady trajectory (Supplementary Movie S7), which throws it out of synchronization with the oncoming wake-vortices. The resulting reduction in efficiency at $t \approx 27.5$ indicates that even slight deviations are capable of impacting performance, and that there may be a measurable delay between actions and consequences. However, the smart-swimmer autonomously corrects for such deviations, and is able to quickly recover its optimal behaviour.

Correlation with the flow-field. The correlation-coefficient curve shown in main Fig. 2e, and the correlation map shown in SI Appendix Fig. S7, were computed as follows:

$$\rho(\mathbf{u}, \mathbf{u}_{\text{head}}) = \frac{\operatorname{cov}\left(\mathbf{u}(x,y), \mathbf{u}_{head}\right)}{\sigma_{\mathbf{u}(x,y)}\ \sigma_{\mathbf{u}_{\text{head}}}} = \frac{\sum_t \mathbf{u}(x,y,t)\cdot \mathbf{u}_{\text{head}}(t)}{\sqrt{\sum_t \|\mathbf{u}(x,y,t)\|^2}\sqrt{\sum_t \|\mathbf{u}_{\text{head}}(t)\|^2}} \qquad [24]$$

Here, $\mathbf{u}(x, y, t)$ was recorded in the wake of a solitary swimmer, whereas $\mathbf{u}_{\text{head}}(t)$ was recorded at the swimmer's head. Maxima in $\rho(\mathbf{u}, \mathbf{u}_{\text{head}})$ provide an estimate for the coordinates where a follower's head-movements would exhibit long-term synchronization with an undisturbed wake.

Limiting the exploration space. During training, the range of values that a smart-follower's states can take are constrained, as mentioned previously. This prevents excessive exploration of regions that involve no wake-interactions, and helps to minimize the computational cost of training-simulations. The limits of the bounding box (shown in SI Appendix Fig. S8) are kept sufficiently large to provide the follower ample room to swim clear of the unsteady wake, if it determines that interacting with the wake is unfavourable.

Power distribution in the presence/absence of a preceding wake. To determine the extent to which wake-induced interactions alter the distribution of P_{Def} and P_{Thrust}, both of which influence overall swimming-efficiency, we compare these quantities for IS_η and SS_η in SI Appendix Fig. S9. A similar comparison for IS_d and SS_d is shown in SI Appendix Fig. S10. For IS_η, a greater variation in P_{Def} and P_{Thrust} is observed (broad envelopes in SI Appendix Figs. S9a and S9b), compared to the solitary swimmer SS_η (SI Appendix Figs. S9c and S9d). This is caused by IS_η's interactions with the unsteady wake, which is absent for SS_η. The average P_{Def} for IS_η shows distinct negative troughs near the head ($s/L < 0.2$, SI Appendix Fig. S9a) and at $s/L = 0.6$. A lack of similar troughs for SS_η (SI Appendix Fig. S9c) implies that these benefits originate exclusively from wake-induced interactions. There is no apparent difference in drag for both IS_η and SS_η in the pressure-dominated region close to the head ($s \approx 0$). However, wake-induced interactions provide a pronounced increase in thrust-power generated by the midsection for IS_η (compare SI Appendix Figs. S9b and S9d, $0.2 < s/L < 0.4$). Among all of the four swimmers compared, only IS_η shows a distinct negative P_{Def} region close to the head ($s < 0.2L$), which further supports the occurrence of head-motion synchronization with flow-induced forces, when efficiency is maximized. Comparing the deformation- and thrust-power distribution for IS_d and SS_d in SI Appendix Fig. S10 provides additional evidence that wake-interactions have a marked impact on swimming-energetics.

Performance of IS_η with respect to an optimal solitary swimmer. A natural question (credited to one Referee) is whether solitary swimming may be preferred to swimming in the wake of a leader. The scenario of a solitary swimmer is an inherent part of the RL training procedure. There are no positional constraints imposed on the smart-follower during training, so it has the possibility to swim at a large lateral distance from the leader, free of the wake's influence and effectively as a solitary swimmer. If solitary swimming with optimal kinematics were preferable to interacting with the leader's wake, the RL algorithm would have converged to this swimming mode as the final strategy for IS_η, instead of preferring to harness the wake-vortices. We emphasize that RL cannot guarantee global minima, but during the training process we did not find solitary swimming as a preferred strategy, instead of the behaviour reported in the manuscript.

We note that optimal morphokinematics of solitary swimmers (albeit at $Re = 500$ and not $Re = 5000$ as studied herein) have been performed in our earlier work.[16] In principle one could train also an efficient solitary swimmer through Reinforcement Learning, but this will require changing the observed states. Finally one may remark that we could have used as baseline leader a swimmer that had been previously optimized. In this context, we have also conducted a parametric search to find the best steady-swimming kinematics for the present baseline fish model. The wake of optimal swimmers is not drastically different from the wake of the present swimmer, and it contains vortex rings that we believe the follower would have reacted to in similar fashion as to the present leader.

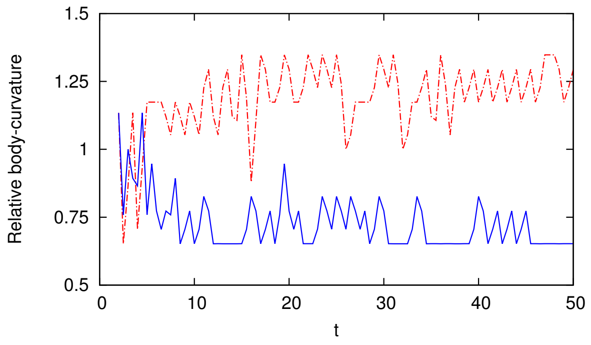
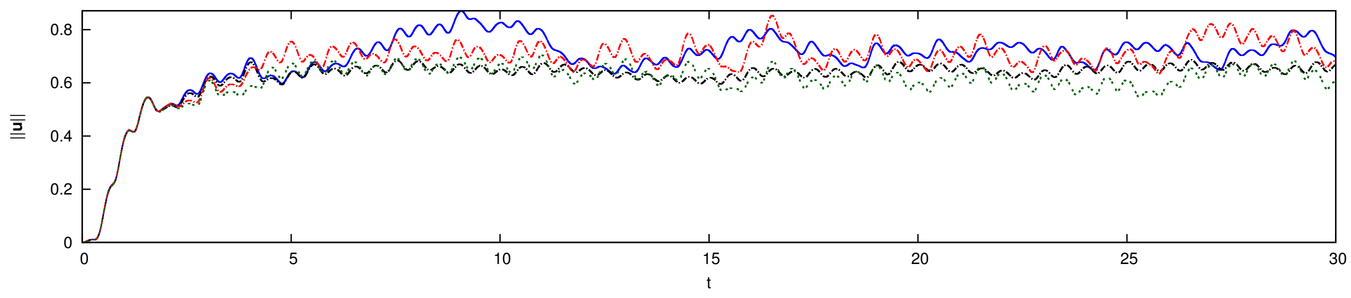
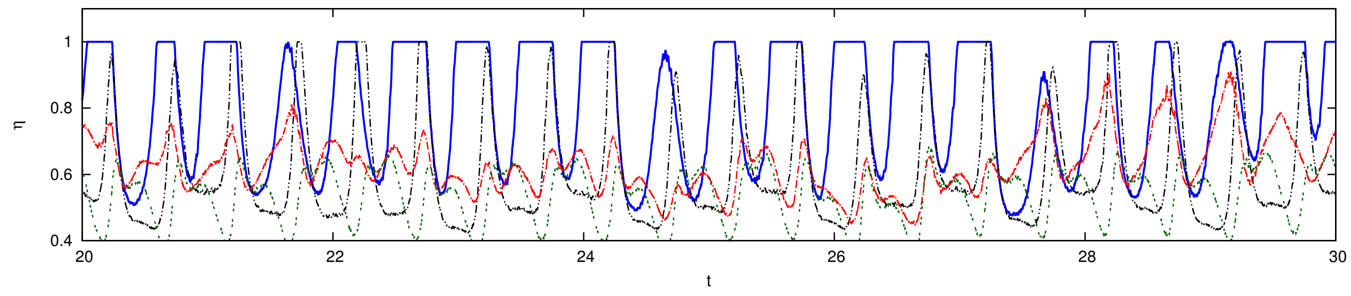


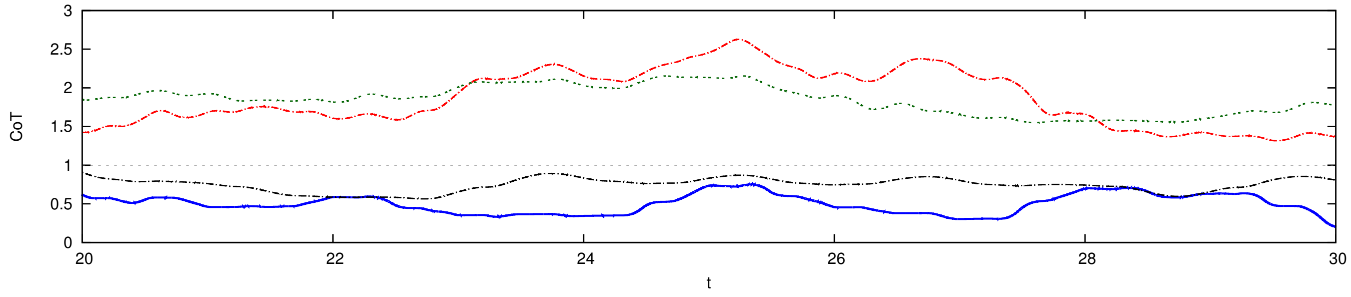
Fig. S1. Midline curvature. Severity of body-deformation for the swimmers IS_η (solid blue line) and IS_d (dash-dot red line), shown for 50 tail-beat periods starting from rest. The relative body-curvature is computed as $\Sigma_{i=1}^{6}|\kappa_i|$, normalized with the same metric for a solitary swimmer executing steady motion (κ_i represents the curvature at 6 control points along a swimmer's body).



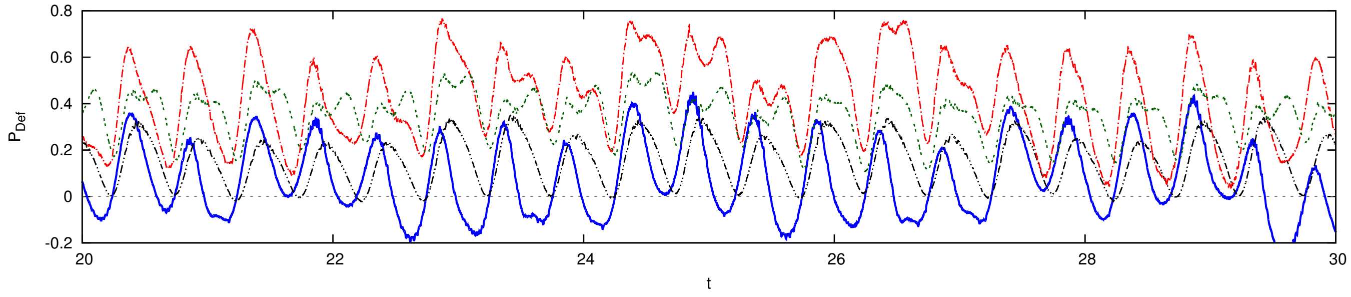
(a)



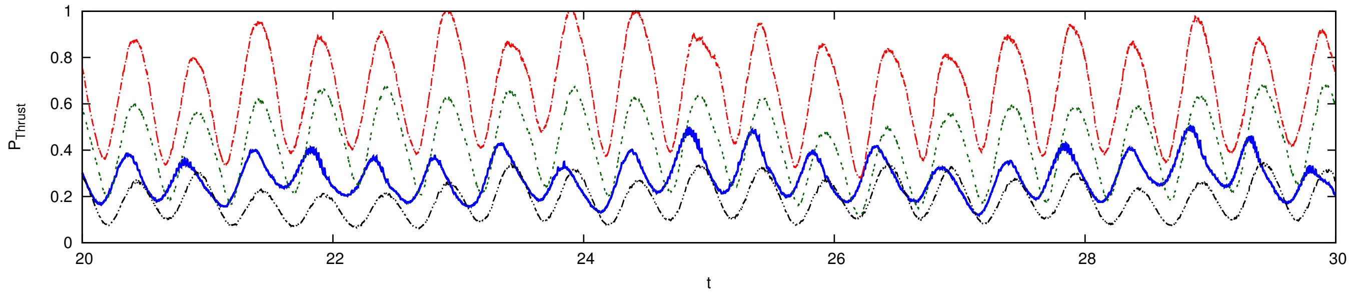
(b)



(c)



(d)



(e)

Fig. S2. Performance metrics for four different swimmers. Plots comparing (a) speed, (b) η, (c) CoT, (d) deformation-power , and (e) thrust-power for four different swimmers. The solid blue line corresponds to swimmer IS_η, the dash-double-dot black line to swimmer SS_η (a solitary swimmer executing actions identical to IS_η), the dash-dot red line to swimmer IS_d, and the double-dot green line to swimmer SS_d (a solitary swimmer executing actions identical to IS_d). The horizontal dashed line at $CoT = 1$ in (c) corresponds to a free-swimming solitary swimmer.

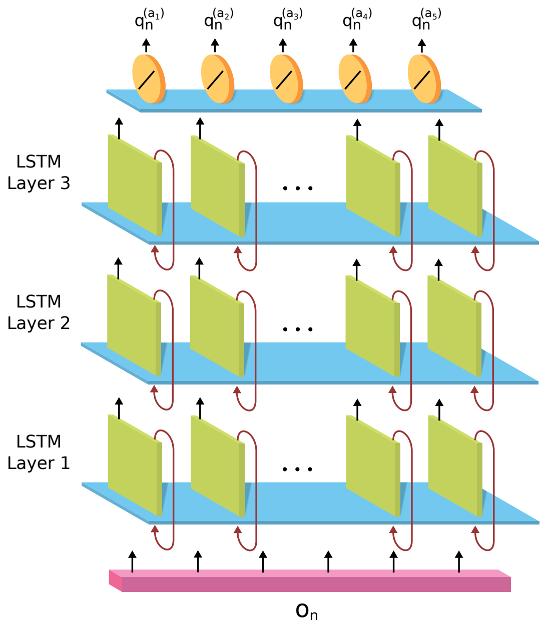
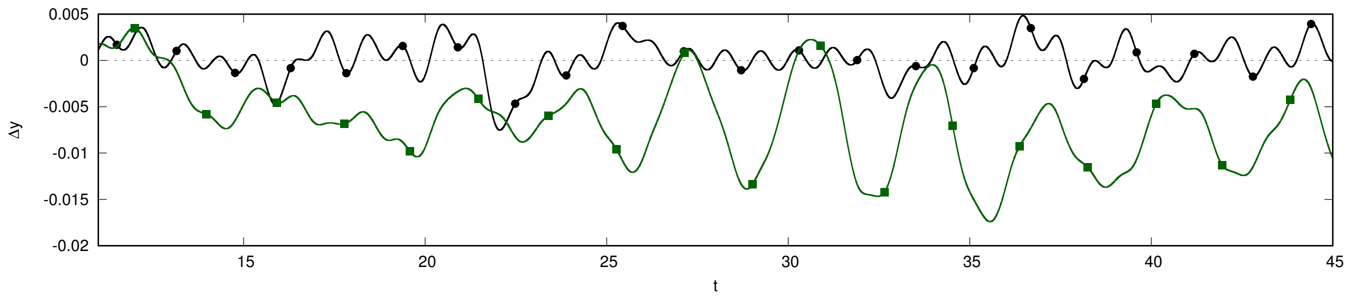
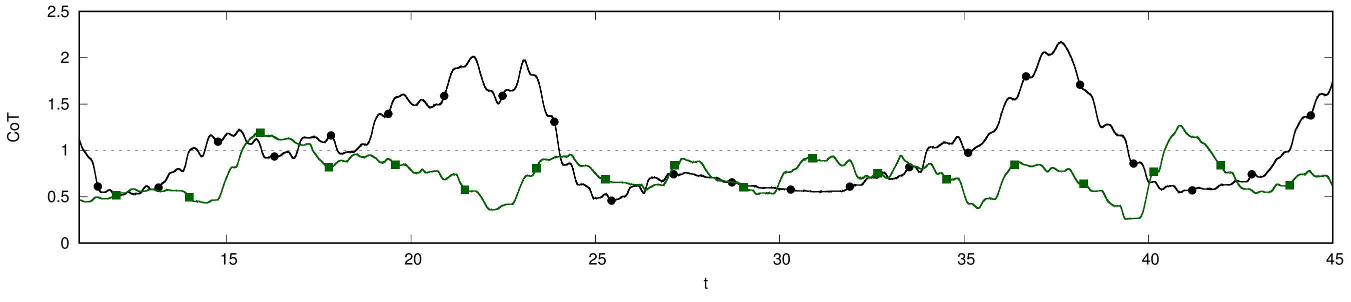


Fig. S3. Schematic of the Recurrent Neural Network (RNN). The RNN used in this study is composed of 3 LSTM layers, consisting of 24 cells (green blocks) each. The input layer (pink block) of the network comprises the 6 observed-state variables. The black arrows between different layers indicate all-to-all connections. The purple arrows indicate recurrent connections within each LSTM layer. The last layer consists of 5 output neurons (orange) with linear activation.

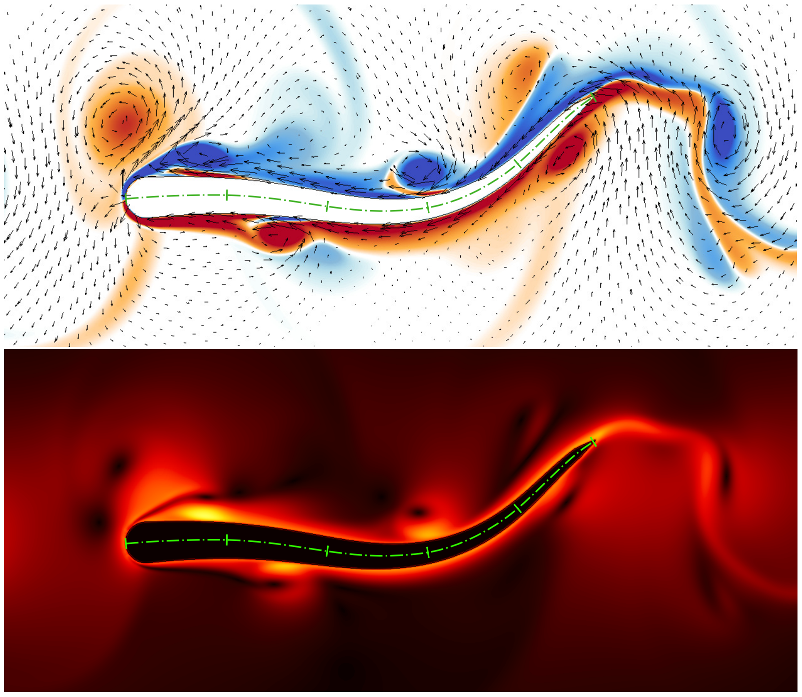


(a)

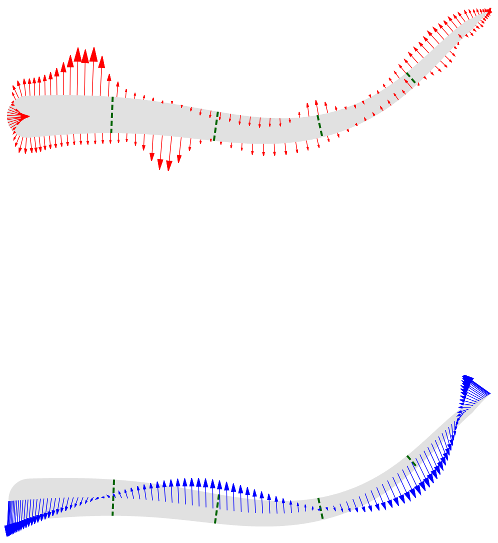


(b)

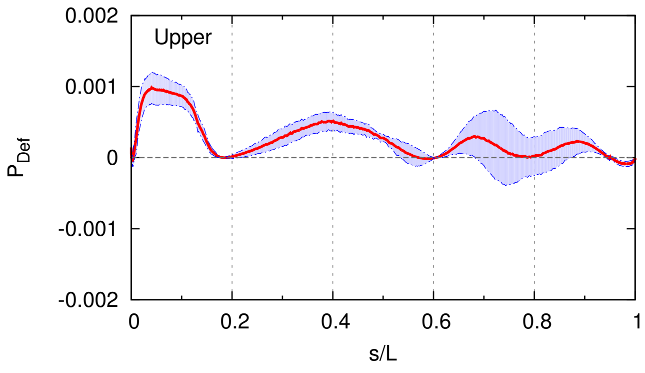
Fig. S4. Comparison of feedforward and recurrent neural networks. (a) Comparing the in-line following capability of smart-swimmers trained using a feedforward neural network (green line with square symbols), and the deep recurrent network shown in Fig. S3 (black line with circle symbols). The horizontal dashed line at $\Delta y = 0$ denotes the target specified for both smart-swimmers. (b) Cost of transport for the two swimmers. The horizontal dashed line at $CoT = 1$ corresponds to a free-swimming solitary swimmer.



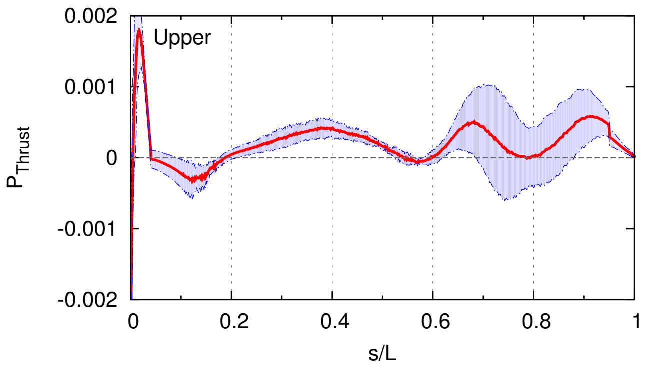
(a)



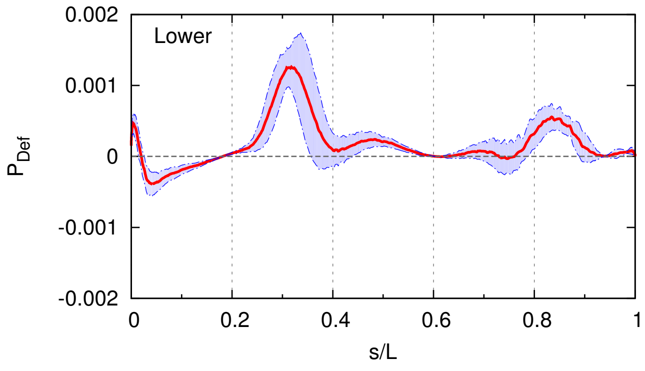
(b)



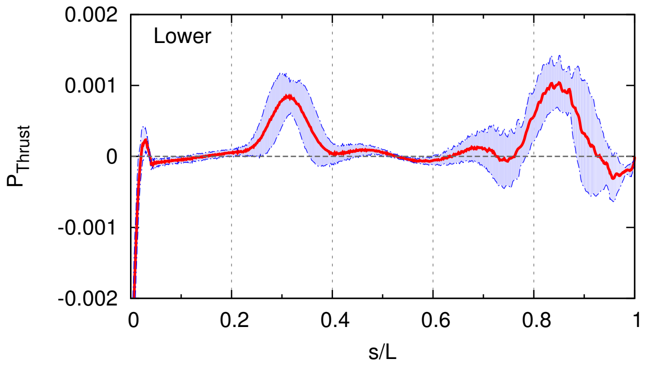
(c)



(d)

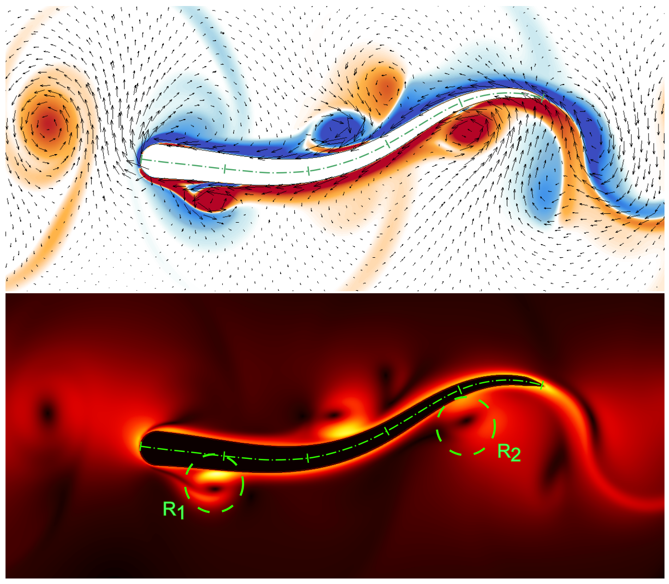


(e)

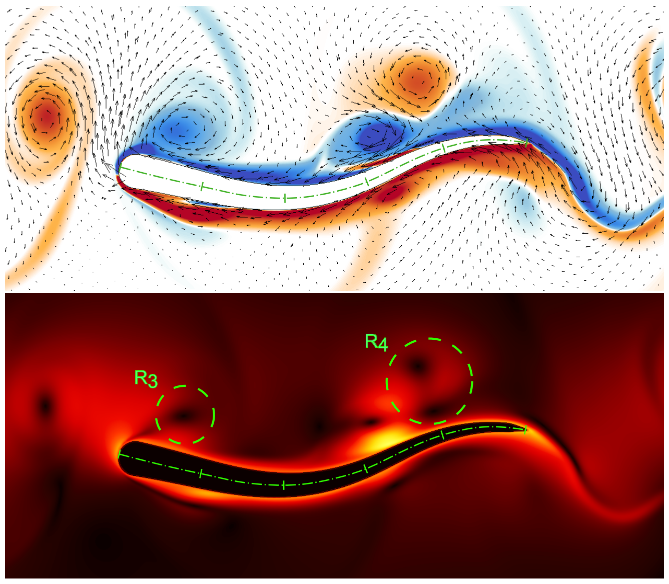


(f)

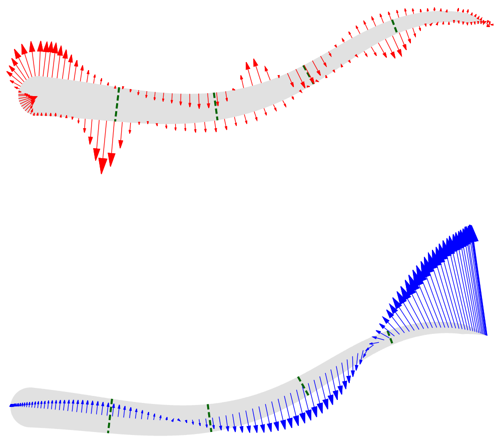
Fig. S5. Flow-field and flow-induced forces for IS_η, corresponding to minimum efficiency. (a) Vorticity field with the velocity vectors shown (top), and velocity magnitude (bottom) at $t = 26.87$ (point $\eta_{min}(D)$ in main Fig. 3). (b) Flow-induced force-vectors (top) and body-deformation velocity (bottom) at this instance. (c,d) Deformation-power and thrust-power acting on the upper (right lateral) surface of follower. The red line indicates the average over 10 different snapshots ranging from $t = 30.87$ to $t = 39.87$. The envelope denotes the standard deviation among the 10 snapshots. (e,f) Deformation-power and thrust-power on the lower (left lateral) surface of the fish.



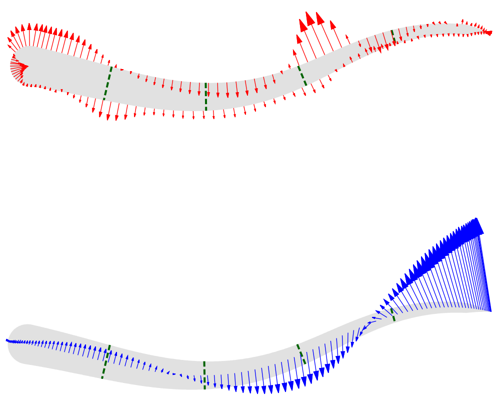
(a)



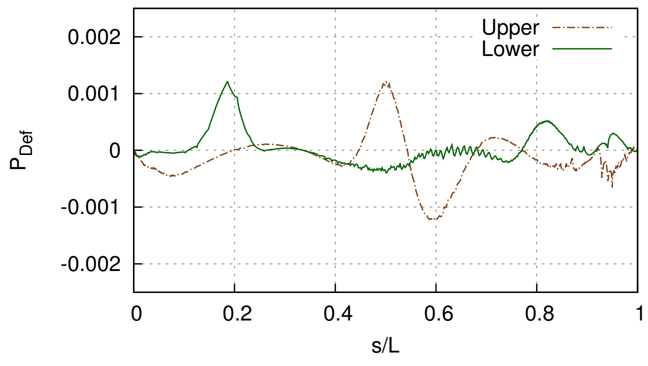
(b)



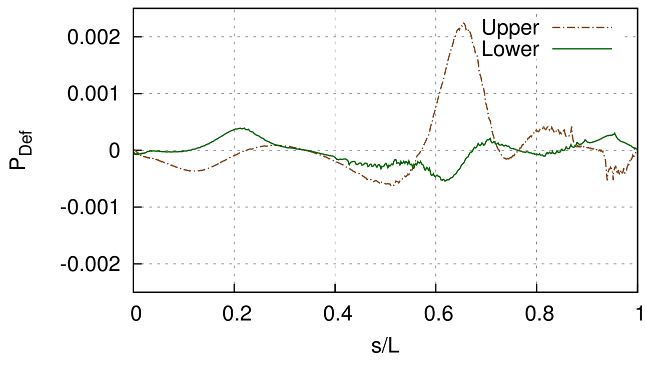
(c)



(d)



(e)



(f)

Fig. S6. Deviations impact performance. Comparison of two instances when a maximum in the swimming-efficiency is expected. The deformed shape and deformation-velocity for the two instances are similar, but differences in the flow-field influence efficiency. Panels on the left hand side of the page show data for IS_η at $t \approx 33.7$ ($\eta = 1$), whereas those on the right hand side correspond to $t \approx 27.7$ ($\eta = 0.86$). (a, b) Vorticity, velocity vectors, and velocity magnitude at the two time instances. A slight deviation in the follower's approach to the wake causes a noticeable change in the surrounding vortices, as well as in the velocity induced near the surface. The regions highlighting differences have been marked as R_1, R_2, R_3, and R_4. (c, d) A comparison of the surface force-vectors and body-deformation velocity. (e,f) There are notable differences in the distribution of P_{Def} on the upper and lower surfaces.

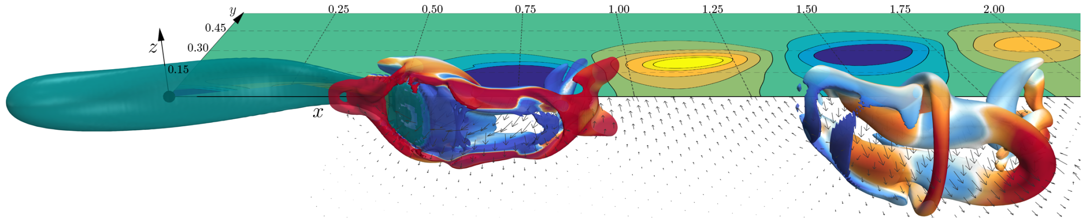


Fig. S7. Correlation map. The horizontal plane on the right side of the swimmer depicts the correlation-coefficient described by Equation 24. Areas of high correlation are denoted as yellow regions, whereas those of low correlation are shown in blue. The vortex rings shed are shown on the swimmer's left side, along with the velocity vectors on the left horizontal plane.

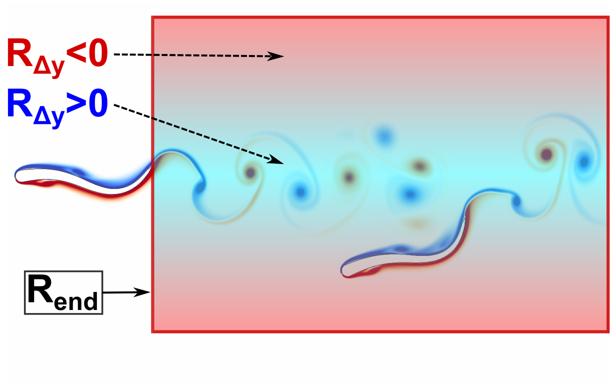


Fig. S8. Reward for IS_d. Visual representation of reward assigned to smart-swimmer IS_d, whose goal is to minimize its lateral displacement from the leader.

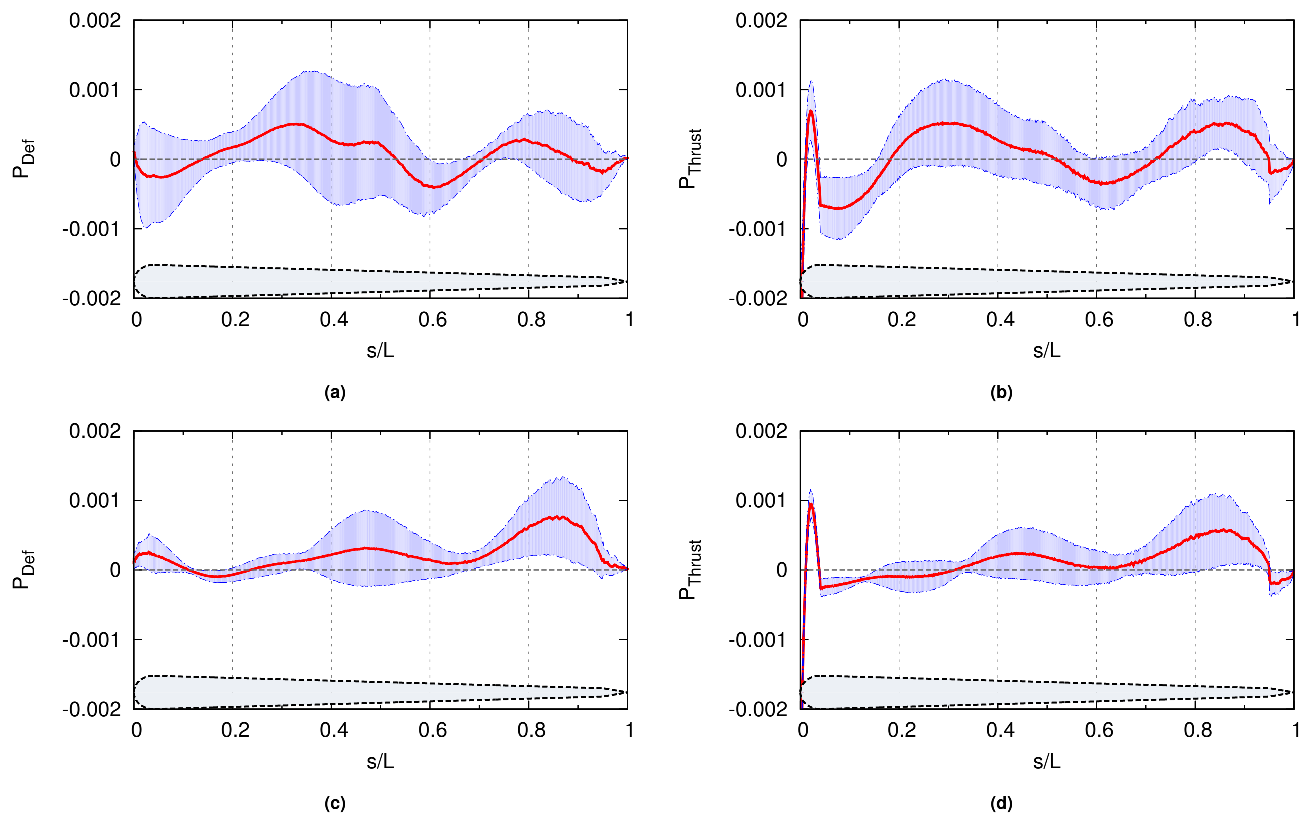


Fig. S9. Power distribution. Deformation-power and thrust-power distribution along the body of (a,b) swimmer IS_η, and (c,d) swimmer SS_η. The solid red line indicates the average over a single tail-beat period (from $t = 26$ to $t = 27$), whereas the envelope denotes the standard-deviation. The silhouettes at the bottom of each panel represent the fish body.

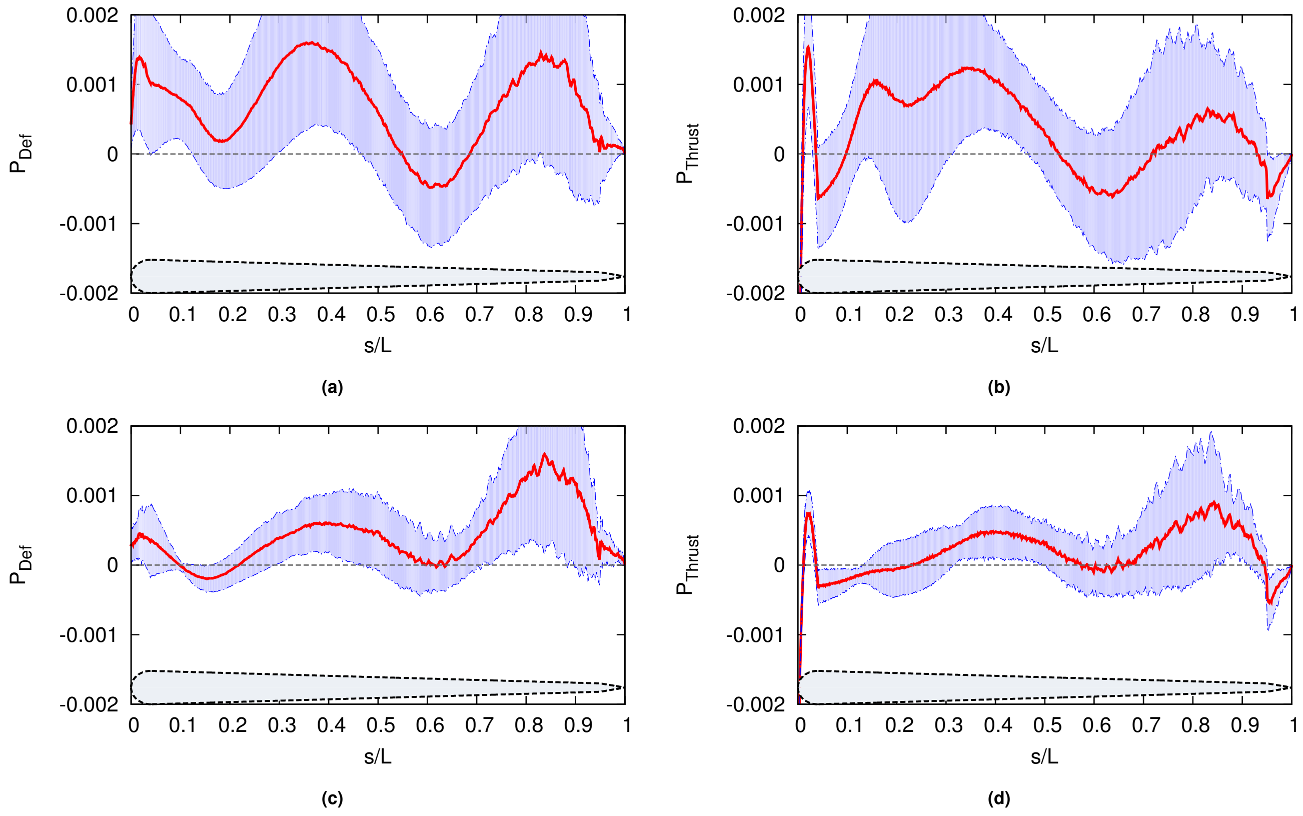


Fig. S10. Power distribution. Deformation-power and thrust-power distribution along the body of (a, b) swimmer IS_d, and (c, d) swimmer SS_d. The solid red line indicates the average over a single tail-beat period (from $t = 26$ to $t = 27$), whereas the envelope denotes the standard-deviation. The silhouettes at the bottom of each panel represent the fish body.

Table S1. Comparison of energetics metrics for the four swimmers. Averaged values computed for the data shown in SI Appendix Fig. S2. All the values shown have been normalized with respect to the corresponding value for IS_η.

	IS_η	SS_η	IS_d	SS_d
η	1.0	0.76	0.77	0.66
CoT	1.0	1.56	3.96	3.86
P_{Def}	1.0	1.41	3.90	3.28
P_{Thrust}	1.0	0.66	2.33	1.48

Movie S1. 3D simulation of three nonautonomous swimmers, in which the leader swims steadily, and the two followers maintain specified relative positions such that they interact favourably with the leader's wake. The flow-structures have been visualized using isosurfaces of the Q-criterion.[17]

Movie S2. 3D simulation of two nonautonomous swimmers, in which the leader swims steadily, and the follower maintains a specified relative position to interact favourably with the wake. The energetic-benefit for the follower is similar to that of each of the followers in Supplementary Movie S1.

Movie S3. 3D simulation of three nonautonomous swimmers, in which the leaders use a feedback controller to maintain formation abreast of each other, and the follower holds a specified position relative to the leaders. The energetic-benefit for the follower is double that of the followers in Supplementary Movies 1 and 2, as it now interacts profitably with wake-rings generated by both the leaders.

Movie S4. 2D simulation of a pair of swimmers, in which the leader swims steadily, and the follower (IS_η) takes autonomous decisions to interact favourably with the wake. The upper panel (labelled 'ω') shows the vorticity field generated by the swimmers, whereas the second panel (labelled 'v') shows the lateral flow-velocity. The smart-swimmer appears to synchronize the motion of its head with the lateral flow-velocity, which allows it to increase its swimming-efficiency. The lower panels show the energetics metrics, namely, the swimming efficiency η, the thrust-power P_{Thrust}, the deformation-power P_{Def}, and the Cost of Transport (CoT).

Movie S5. 2D simulation of a pair of swimmers, where the leader performs random actions, and the follower takes autonomous decisions to benefit from the flow-field. The smart-follower, which was trained with a steadily-swimming leader, is able to adapt to the erratic leader's behaviour without any further training. Remarkably, the follower chooses to interact deliberately with the wake in order to maximize its long-term swimming-efficiency, even though it has the option to swim clear of the unsteady flow-field.

Movie S6. A qualitative comparison between swimmer IS_η and a real fish following a leader. We observe that the motion of IS_η resembles that of the live follower quite well. The leader in the simulation executes random turns after every few tail-beat cycles, and the follower responds to changes in range and bearing, similarly to Supplementary Movie S4.

Movie S7. Detailed view of the flow-field around smart-swimmer IS_η. The top panel shows the vorticity field in colour and velocity vectors as black arrows. The middle panels show the swimming-efficiency and the deformation-power. The distribution of thrust-power and deformation-power along the swimmer's left- ('lower') and right-lateral ('upper') surfaces are shown in the lower panels, and depict how these quantities depend on wake-interactions.

References

[1] Coquerelle M, Cottet GH (2008) A vortex level set method for the two-way coupling of an incompressible fluid with colliding rigid bodies. *J. Comput. Phys.* 227:9121–9137.

[2] Rossinelli D, et al. (2015) MRAG-I2D: Multi-resolution adapted grids for remeshed vortex methods on multicore architectures. *J. Comput. Phys.* 288:1–18.

[3] Verma S, Abbati G, Novati G, Koumoutsakos P (2017) Computing the force distribution on the surface of complex, deforming geometries using vortex methods and brinkman penalization. *Int. J. Numer. Meth. Fluids* 85(8):484–501.

[4] Chorin AJ (1968) Numerical solution of the Navier-Stokes equations. *Math. Comp.* 22:745–762.

[5] Rossinelli D, et al. (2013) 11 pflop/s simulations of cloud cavitation collapse in *High Performance Computing, Networking, Storage and Analysis (SC), 2013 International Conference for.* (IEEE), pp. 1–13.

[6] Gholami A, Hill J, Malhotra D, Biros G (2015) AccFFT: A library for distributed-memory FFT on CPU and GPU architectures. *arXiv preprint arXiv:1506.07933.*

[7] Tytell ED, Lauder GV (2004) The hydrodynamics of eel swimming. *J. Exp. Biol.* 207:1825–1841.

[8] van Rees WM, Gazzola M, Koumoutsakos P (2013) Optimal shapes for anguilliform swimmers at intermediate reynolds numbers. *J. Fluid Mech.* 722:R3 1–12.

[9] Sutton RS, Barto AG (1998) *Reinforcement learning: An introduction.* (MIT press, Cambridge, MA, USA).

[10] Bertsekas DP (1995) *Dynamic programming and optimal control.* (Athena scientific Belmont, MA) Vol. 1.

[11] Mnih V, , et al. (2015) Human-level control through deep reinforcement learning. *Nature* 518:529–533.

[12] Hochreiter S, Schmidhuber J (1997) Long short-term memory. *Neural Comput.* 9:1735–1780.

[13] Kingma DP, Ba J (2014) Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980 [cs.LG].*

[14] Graves A, Schmidhuber J (2005) Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural Netw.* 18:602–610.

[15] Novati G, et al. (2017) Synchronisation through learning for two self-propelled swimmers. *Bioinspir. Biomim.* 12:036001.

[16] van Rees WM, Gazzola M, Koumoutsakos P (2015) Optimal morphokinematics for undulatory swimmers at intermediate Reynolds numbers. *J. Fluid Mech.* 775:178–188.

[17] Hunt JCR, Wray AA, Moin P (1988) Eddies, streams, and convergence zones in turbulent flows in *Studying Turbulence Using Numerical Simulation Databases, 2. Report CTR-S88.* pp. 193–208.