Pareto Optimal Swimmers

Siddhartha Verma
Computational Science and Engineering Laboratory, ETH Zürich
Clausiusstrasse 33
Zürich CH-8092, Switzerland
sverma@ethz.ch

Panagiotis Hadjidoukas
Computational Science and Engineering Laboratory
ETH Zürich
Clausiusstrasse 33
Zürich CH-8092, Switzerland
chatzidp@ethz.ch

Philipp Wirth
Computational Science and Engineering Laboratory
ETH Zürich
Clausiusstrasse 33
Zürich CH-8092, Switzerland
pwirth@student.ethz.ch

Diego Rossinelli
Lucid Concepts AG
Technoparkstrasse 1
Zürich CH-8005, Switzerland
rossinelli.diego@gmail.com

Petros Koumoutsakos*†‡
Computational Science and Engineering Laboratory
ETH Zürich
Clausiusstrasse 33
Zürich CH-8092, Switzerland
petros@ethz.ch

ABSTRACT

A fundamental understanding of how various biological traits and features provide organisms with a competitive advantage can help us improve the design of a number of mechanical systems. Numerical optimization can play an invaluable role for this purpose, by allowing us to scrutinize the evolution of specific biological adaptations in nature. Importantly, the use of numerical optimization can help us overcome limiting constraints that restrict the evolutionary capability of biological species. We capitalize on these advantages by coupling high-fidelity simulations of self-propelled swimmers with evolutionary optimization algorithms, to examine peculiar swimming patterns observed in a number of fish species. More specifically, we investigate the intermittent form of locomotion referred to as 'burst-and-coast' swimming, which involves a few quick flicks of the fish's tail followed by a prolonged unpowered glide. This mode of swimming is believed to confer energetic benefits, in addition to several other advantages. We discover a range of intermittent-swimming patterns, the most efficient of which resembles the swimming behaviour observed in live fish. We also discover patterns which lead to a marked increase in swimming-speed, albeit with a significant increase in energy expenditure. Notably, the use of multi-objective optimization reveals locomotion patterns that strike the perfect balance between speed and efficiency, which can be invaluable for use in robotic applications. As an additional goal of the paper, we highlight the ease with which disparate codes can be coupled via the software framework used, without encumbering the user with the details of efficient parallelization and machine-learning based task-scheduling.

CCS CONCEPTS

• **Applied computing → Physical sciences and engineering**; *Computational biology*;

KEYWORDS

Task-based Parallelism; Multi-objective Optimization; Burst-and-coast Swimming; Energy-efficient Locomotion

ACM Reference format:
Siddhartha Verma, Panagiotis Hadjidoukas, Philipp Wirth, Diego Rossinelli, and Petros Koumoutsakos. 2017. Pareto Optimal Swimmers. In *Proceedings of PASC '17, Lugano, Switzerland, June 26-28, 2017,* 11 pages.
https://doi.org/10.1145/3093172.3093232

1 INTRODUCTION

Steady, continuous swimming for long duration is rarely observed in most fish species. Instead, individuals usually employ short bursts of activity followed by a brief inactive period, where inertia allows them to glide along for a certain distance [9]. This unsteady swimming pattern, referred to as 'burst-and-coast' swimming, has been hypothesized to yield energetic benefits [38, 39], and to bestow a competitive edge to species employing the technique. This sort of intermittent motion is not just limited to fish-swimming, but is found in a variety of animal species [27]. In addition to reducing energy expenditure, the inactive phase of the motion has been attributed with stabilizing the sensory field, enhancing the possibility of prey-detection, and diminishing the wake-signature to avoid alerting potential prey and predators [21, 40]. Unfortunately, all of these advantages are usually accompanied by a reduction in average speed of the organism [37].

*Corresponding Author: petros@ethz.ch
†Also with Radcliffe Institute of Advanced Study, Harvard University, MA, United States of America.
‡Also with Massachusetts Institute of Technology, MA, United States of America.

PASC '17, June 26-28, 2017, Lugano, Switzerland

ACM ISBN 978-1-4503-5062-4/17/06…$15.00
https://doi.org/10.1145/3093172.3093232

Early studies investigating the energetic benefits of intermittent swimming relied on simplified inviscid flow-dynamics [38, 39]. Certain approaches have even represented fish bodies using prolate spheres [3]. Videler & Weihs [38] analysed video of free swimming cod and saithe, and used simplified energetics models to surmise that the fish used burst-coast patterns that resulted in the most efficient locomotion. Müller et al. [23] and Videler et. al [37] used particle image velocimetry (PIV) to study flow-patterns generated during burst-coast swimming. McHenry & Lauder investigated the dependence of coasting behaviour on changes in the body-morphology [22]. Wu et al. [41] demonstrated the existence of two distinct tail-beat modes for swimming carp, which adopt burst-coast swimming for approximately 45% to 75% of their swimming time. Apart from various experimental investigations, burst-coast swimming has also been the subject of some numerical studies [2, 4], albeit using a-priori specified parameters for the burst-coast motion.

In order to discover the best burst-and-coast swimming approach, without having to resort to simplifying assumptions, we couple high-fidelity simulations of self-propelled swimmers with evolutionary-optimization algorithms [6, 18]. The use of multi-objective optimization allows us to consider two important, and potentially conflicting metrics, namely, the average speed, and the Cost of Transport (CoT). These two quantities dictate overall performance for the majority of motile organisms, and consequently, determine their evolutionary fitness to pass on advantageous characteristics to future generations. Analyzing flow-patterns generated by individuals that emerge as optimal solutions can provide invaluable insight regarding intermittent-swimming modes, which can help us improve the efficacy of underwater propulsion systems. Furthermore, we investigate the trade-off between speed and efficiency that fish experience when transitioning from the larval to the adult stage of their lives. This is essential for ascertaining the causal-effects that influence the evolution of the burst-coast pattern, as certain studies have shown that the preference for passive coasting increases with age (and hence, with the body size) [25].

The current work is distinct from previous numerical and analytical studies that have utilized optimization algorithms to examine self-propelled swimmers [8, 11, 19, 33, 35, 36], in that we study intermittent-locomotion modes. Moreover, we consider two disparate performance metrics simultaneously, both of which are critical for the survival of most organisms. From a computational viewpoint, the modular approach used for coupling standalone applications in this study, each of which is highly specialized for its own particular task, allows for nearly effortless, and extremely effective task-based parallelization of the optimization procedure. The parallelization is further enhanced by machine-learning based task-scheduling.

2 SIMULATION DETAILS

We use a software framework, 'MRAG-I2D' [30], to conduct simulations of self-propelled fish-like swimmers. The code is an open source framework for simulating two-dimensional, viscous, incompressible flows on multi-core architectures. The use of multi-resolution grids, capable of adapting automatically in both space and time, enables accurate simulation of physical systems, while keeping both computational cost and memory requirement low. The solver used in the present study is based on the remeshed vortex methods [20], and has been validated and used extensively for simulations of complex, deforming objects [10, 11, 35].

2.1 Governing equations

The velocity field in the simulations is governed by the incompressible Navier-Stokes equations:

$$\nabla \cdot \boldsymbol{u} = 0 \tag{1}$$

$$\frac{\partial \boldsymbol{u}}{\partial t} + (\boldsymbol{u} \cdot \nabla)\boldsymbol{u} = \frac{-\nabla P}{\rho} + \nu \nabla^2 \boldsymbol{u} \tag{2}$$

The interaction between fluid-flow and solid objects is achieved via Brinkman penalization [1], which leads to a modified form of the momentum equation:

$$\frac{\partial \boldsymbol{u}}{\partial t} + (\boldsymbol{u} \cdot \nabla)\boldsymbol{u} = \frac{-\nabla P}{\rho} + \nu \nabla^2 \boldsymbol{u} + \lambda \chi \left(\boldsymbol{u}_s - \boldsymbol{u}\right) \tag{3}$$

Here, λ is a penalty parameter, and χ is the characteristic function describing the distribution of the solid object on a the Cartesian fluid-grid. The symbol $\boldsymbol{u}_s$ in Eq. 3 denotes the pointwise velocity of the discretized solid, and accounts for translation, rotation, and deformation of the body. The vorticity form of the momentum equation is obtained by taking the curl of Eq. 3:

$$\frac{\partial \boldsymbol{\omega}}{\partial t} + (\boldsymbol{u} \cdot \nabla)\boldsymbol{\omega} = \nu \nabla^2 \boldsymbol{\omega} + \lambda \nabla \times \left(\chi \left(\boldsymbol{u}_s - \boldsymbol{u}\right)\right) \tag{4}$$

Further details regarding spatial discretization, fluid-solid interaction, and the time-splitting steps involved in solving Eq. 4 are described in refs. [10, 30].

2.2 Swimmer kinematics & Optimization parameters

The undulatory kinematics of the self-propelled swimmer is described by the following travelling wave:

$$y_{midline}(s,t) = f(t) \cdot \frac{4}{33}\left(s + 0.03125L\right) \cdot sin\left(2\pi\left(\frac{s}{L} - \frac{t}{T} + \phi\right)\right) \tag{5}$$

Here, s is the curvilinear parameter representing the backbone of the fish from the head to the tail tip, with $s \in [0, L]$. L is the length of the fish, T denotes the time-period of the travelling wave, and ϕ represents a phase shift. Further details regarding the description of the body-geometry may be found in [10].

The function $f(t)$ controls the envelope of the time-varying amplitude of the swimmer. It is defined as a piecewise continuous function, with intervals designated by the time points t_A, t_B, t_C and t_D. At t_A, the fish initiates a deceleration by reducing its undulatory movement. The swimmer ceases all motion at t_B, and glides in an unpowered state until t_C. At t_C, the swimmer restarts its motion until reaching full amplitude at t_D. For a swimmer performing the $i-th$ burst-and-coast cycle, the piecewise function $f(t)$ is computed as follows:

$$f(t) = \begin{cases} 1 & t_D^{(i-1)} \le t < t_A^{(i)} \\ 1 - 3\lambda_{coast}^2 + 2\lambda_{coast}^3 & t_A^{(i)} \le t < t_B^{(i)} \\ 0 & t_B^{(i)} \le t < t_C^{(i)} \\ 3\lambda_{burst}^2 - 2\lambda_{burst}^3 & t_C^{(i)} \le t < t_D^{(i)} \\ 1 & t_D^{(i)} \le t \end{cases} \tag{6}$$

Here, $\lambda_{coast}, \lambda_{burst} \in [0, 1]$ are ramp functions increasing linearly from 0 to 1 as t goes from t_A to t_B (t_C to t_D respectively). This particular definition of the piecewise function prevents discontinuous jerks and acceleration, thereby avoiding spurious numerical oscillations in the flow-field.

The controlling parameters for the optimizations are then:

$$t_{Decel} = t_B^{(i)} - t_A^{(i)} \quad \in [0.1, 1.5] \tag{7}$$

$$t_{Coast} = t_C^{(i)} - t_B^{(i)} \quad \in [0, 3] \tag{8}$$

$$t_{Accel} = t_D^{(i)} - t_C^{(i)} \quad \in [0.1, 1.5] \tag{9}$$

$$t_{Steady} = t_A^{(i+1)} - t_D^{(i)} \quad \in [0, 2] \tag{10}$$

This choice of controlling parameters allows a simple representation of burst-and-coast swimming, and keeps the dimension of the search-space to a reasonable size. The lower bounds, $t_{Decel} \geq 0.1$ and $t_{Accel} \geq 0.1$ prevent the swimmers from experiencing excessively large accelerations.

2.3 Performance metrics

The objective of the self-propelled swimmers is to discover the best mode of intermittent swimming, which maximizes the average speed, while at the same time minimizing the energy consumed for travelling a unit distance (also referred to as the Cost of Transport). Computing the Cost of Transport (or CoT) requires knowledge of the pressure- and viscosity-induced forces:

$$d\boldsymbol{F}_P = -P\boldsymbol{n}\, dS \tag{11}$$

$$d\boldsymbol{F}_v = 2\mu \boldsymbol{D} \cdot \boldsymbol{n}\, dS \tag{12}$$

Here, $\boldsymbol{D} = \frac{1}{2}\left(\nabla \boldsymbol{u} + \nabla \boldsymbol{u}^T\right)$ is the strain-rate tensor, μ is the dynamic viscosity of the fluid, $\boldsymbol{n}$ is the surface normal, and dS is the infinitesimal surface area. The pressure P is computed from a Poisson's equation obtained by taking the divergence of the penalized momentum equation (Eq. 3), and utilizing the incompressibility condition ($\nabla \cdot \boldsymbol{u} = 0$):

$$\nabla^2 P = -\rho\left(\nabla \boldsymbol{u}^T : \nabla \boldsymbol{u}\right) + \rho\lambda \nabla \cdot (\chi\,(\boldsymbol{u}_s - \boldsymbol{u})) \tag{13}$$

This Poisson's equation is solved via an extremely efficient tree-code algorithm, which is based on multipole expansions [12]. Moreover, pressure computations are carried out only at the surface nodes of the body, which reduces the computational cost significantly.

The thrust generated by the swimmers is determined as follows:

$$\text{Thrust} = \frac{1}{2\|\boldsymbol{u}\|} \iint (\boldsymbol{u} \cdot d\boldsymbol{F} + |\boldsymbol{u} \cdot d\boldsymbol{F}|) \tag{14}$$

where $d\boldsymbol{F} = d\boldsymbol{F}_P + d\boldsymbol{F}_v$. Using these quantities, the thrust- and deformation-power are computed as:

$$P_{Thrust} = \text{Thrust} \cdot \|\boldsymbol{u}\| \tag{15}$$

$$P_{Def} = -\iint \boldsymbol{u}_{Def} \cdot d\boldsymbol{F} \tag{16}$$

where $\boldsymbol{u}_{Def}$ represents the deformation-velocity of the fish body. The instantaneous swimming-efficiency is based on a modified form of the Froude efficiency proposed in ref. [34]:

$$\eta = \frac{P_{Thrust}}{P_{Thrust} + \max(P_{Def}, 0)} \tag{17}$$

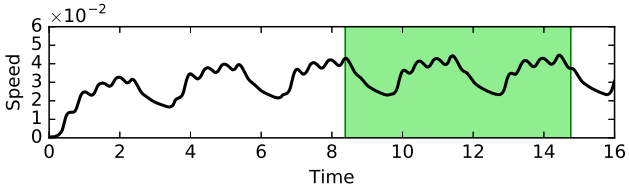


Figure 1: The instantaneous speed of a swimmer employing intermittent locomotion at $Re = 400$. The highlighted region depicts the averaging zone used for computing the fitness value. The averaging is done over the last two complete cycles to avoid unsteady transients towards the beginning of the simulation.

To compute both η and the Cost of Transport (CoT), we neglect negative values of P_{Def}, which corresponds to the fact that the rigid fish-body may not store energy internally:

$$CoT(t) = \frac{\int_{t-T_p}^{t} \max(P_{Def}, 0) dt}{\int_{t-T_p}^{t} \|\boldsymbol{u}\| dt} \tag{18}$$

This restriction yields a conservative estimate of potential savings in the CoT. To avoid the initial transient when the fish accelerates from rest, both the average speed and average CoT are determined over the final 2 burst-coast cycles in a simulation, as depicted in Fig. 1. These two quantities represent the 'fitness' values used for multi-objective optimization, which is described in greater detail in Section 3.

3 OPTIMIZATION ALGORITHMS

Evolutionary-optimization algorithms are designed to mimic the rules of evolution in biology, and involve the processes of mutation, reproduction, and selection. These strategies operate on a collection of individuals to iteratively improve the population with respect to a certain 'fitness' value. The fitness value represents the eligibility of a particular individual to pass on its characteristics to future generations, and in our case is determined via high-fidelity simulations of self-propelled swimmers.

We focus on multi-objective fitness functions that produce an output in $\mathbb{R}^2$, which can be ordered using the concept of Pareto dominance. A vector $\boldsymbol{u} \in \mathbb{R}^n$ dominates a vector $\boldsymbol{v} \in \mathbb{R}^n$ in terms of Pareto dominance if the following two conditions are met:

$$\forall i \in \{1, ..., n\}\ \boldsymbol{u}_i \leq \boldsymbol{v}_i \tag{19}$$

$$\exists j \in \{1, ..., n\}\ \boldsymbol{u}_j < \boldsymbol{v}_j \tag{20}$$

Any vector that is not dominated by any other vector in the set is called 'non-dominated'. Each fitness value is associated with a non-domination rank, which is set to 0 for non-dominated solutions. Further non-domination ranks are computed iteratively using the fitness values, after excluding individuals with lower ranks.

In the current work, we use two distinct algorithms, namely Multi-Objective Covariance Matrix Adaptation Evolution Strategy (MO-CMA-ES), and Non-dominated Sorting Genetic Algorithm-II (NSGA-II), to study intermittent-swimming characteristics. The two algorithms, which are described below, use the concept of Pareto

dominance to solve multi-objective optimization problems, by assigning each individual of the population a rank according to its level of non-dominance. Individuals with the same non-dominance ranks are ordered either using the S-metric (MO-CMA-ES), or the crowding distance parameter (NSGA-II). The algorithm parameters, such as population size and mutation probability were not varied in the current study, owing to the relatively high computational cost of individual function-evaluations.

3.1 MO-CMA-ES

The Covariance Matrix Adaptation Evolution Strategy (CMA-ES) [15] is a powerful evolutionary algorithm, which omits the concept of reproduction, and relies solely on mutation. The multi-objective variant of CMA-ES was developed, and compared to other widely-used multi-objective algorithms, by Igel et al. in [16]. The MO-CMA-ES algorithm [18] implemented in the 'Shark' library [17] was used for multi-objective optimization in the current work. An outline of the steps involved is shown in Algorithm 1, with further details described in the text below.

Algorithm 1: MO-CMA-ES

```
Initialize population P^(0) of size μ and set generation g = 0.
while No stopping criterion is met do
    for k = 1, ..., μ do
        Sample an offspring x̃_k ~ N(x_k^(g), σ_k^(g), C_k^(g)).
    end
    Choose the best μ individuals from a mixed population
    containing parents and offspring and store them in P^(g+1).
    for k = 1, ..., μ do
        Update success rate p̄_{succ,k}^(g+1) and global step size σ_k^(g+1).
        Update evolution path p_{k,c}^(g+1) and the covariance
        matrix C_k^(g+1).
    end
    g = g + 1
end
Output the current population.
```

The $k-th$ individual in the $g-th$ generation is described by a 5-tuple:

$$a_k^{(g)} = \left[\boldsymbol{x}_k^{(g)}, \bar{p}_{succ,k}^{(g)}, \sigma_k^{(g)}, p_{c,k}^{(g)}, C_k^{(g)}\right] \tag{21}$$

Here, $\boldsymbol{x}_k$ represents the vector of search-parameters, and $\bar{p}_{succ,k}$ denotes the average success rate of this individual. σ_k is the global step size, $p_{c,k}$ is the cumulative evolution path, and C_k is the covariance matrix. The average success rate $\bar{p}_{c,k}$ is a heuristic used to manage the global step size σ_k, and the covariance matrix C_k determines the most probable direction of the mutation. The covariance matrix is updated using the evolution path $p_{c,k}$, which keeps track of the mutation-direction of the previous few steps. The success rate of each individual is updated as follows:

$$\bar{p}_{succ,k}^{(g+1)} = (1-c_p)\bar{p}_{succ,k}^{(g)} + c_p p_{succ,k}^{(g)} \tag{22}$$

where c_p is a constant weighting factor, and $p_{succ,k}^{(g)}$ is equal to one if the offspring of the $k-th$ individual dominates its parent. The step size update then becomes:

$$\sigma_k^{(g+1)} = \sigma_k^{(g)} \cdot e^{\left(\frac{1}{d}\cdot\frac{\bar{p}_{succ,k}-p_{succ}^{target}}{1-p_{succ}^{target}}\right)} \tag{23}$$

Here, d is a damping parameter, and p_{succ}^{target} is the target success probability. Intuitively, the step size drops if the smoothed success probability $\bar{p}_{succ,k}$ is smaller than the target success probability p_{succ}^{target}, and grows otherwise. The update of the the covariance matrix is done differently depending on whether $\bar{p}_{succ}$ is smaller than some threshold $p_{thresh} \in [0,1]$ ([18]). This is necessary to prevent the eigenvalues of C_k from becoming too large. If $\bar{p}_{succ} < p_{thresh}$, the update reads as follows:

$$p_{c,k}^{(g+1)} = (1-c_c)p_{c,k}^{(g)} + \sqrt{c_c(2-c_c)}\left(\frac{x_k^{(g+1)}-x_k^{(g)}}{\sigma_k^{(g)}}\right) \tag{24}$$

$$C_k^{(g+1)} = (1-c_{cov})C_k^{(g)} + c_{cov}\cdot p_{c,k}^{(g+1)} p_{c,k}^{\mathsf{T}(g+1)} \tag{25}$$

otherwise, the update of the covariance matrix is performed as:

$$p_{c,k}^{(g+1)} = (1-c_c)p_{c,k} \tag{26}$$

$$C_k^{(g+1)} = (1-c_{cov})C_k^{(g)} \tag{27}$$

$$+ \; c_{cov}\cdot\left(p_{c,k}^{(g+1)} p_{c,k}^{\mathsf{T}(g+1)} + c_c(2-c_c)\cdot C_k^{(g)}\right) \tag{28}$$

3.2 NSGA-II

NSGA-II [6] improves on the original NSGA [31] by introducing elitism, the crowding distance parameter, and a fast non-dominated sorting algorithm. Elitism refers to the best individuals of a generation being carried over to the next generation. The crowding distance parameter is an estimate of the average side-length of the cuboid formed by an individual's two closest neighbours. Individuals with a higher crowding distance are preferred because they increase diversity in the solution.

A description of the NSGA-II procedure is provided in Algorithm 2. The k$-th$ individual in generation g is described by a vector $\boldsymbol{x}_k^{(g)}$ containing the search parameters. The crossover step

Algorithm 2: NSGA-II

```
Initialize population P^(0) of size μ and set generation g = 0.
while No stopping criterion is met do
    Generate an offspring population P̄^(g+1) from the parent
    population P^(g) using crossover (Algorithm 3) and
    mutation (Algorithm 4).
    Combine offspring P̄^(g+1) and parent population P^(g) into
    a mixed population and sort it according to
    non-domination rank and crowding distance.
    Select the best μ individuals for the next parent generation
    P^(g+1).
    g = g + 1
end
Output the current population.
```

of the algorithm is performed using Simulated Binary Crossover (SBX) (Algorithm 3), whereas Parameter-based Mutation (PBM - Algorithm 4) is used for the mutation step [5]. All the individuals are represented by real-valued, n-dimensional vectors, where n is the dimension of the search-space (i.e., the number of parameters being optimized). We define the parent individuals as $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ and the children as $\boldsymbol{c}_1$ and $\boldsymbol{c}_2$. Using the vectors $\boldsymbol{x}_{lower}$ and $\boldsymbol{x}_{upper}$ to denote the lower and upper boundaries of the search space, the crossover and mutation steps are performed as shown in Algorithms 3 and 4.

Algorithm 3: Simulated Binary Crossover

Inputs are the parent individuals $\boldsymbol{x}_1$ and $\boldsymbol{x}_2 \in \mathbb{R}^n$.
for $i \in \{1, ..., n\}$ **do**
  Compute

$$m = x_{1,i} - x_{lower,i}$$
$$M = x_{upper,i} - x_{2,i}$$

  Compute

$$\beta = \left(1 + \frac{2}{x_{2,i} - x_{1,i}} \cdot min\,(m, M)\right) \text{ and } \alpha = \left(2 - \beta^{-(\eta_c+1)}\right)$$

  Create a random number $u \in [0, 1]$

$$\beta_q = \begin{cases} (u\alpha)^{\frac{1}{\eta_c+1}} & if\ u \le \frac{1}{\alpha} \\ \left(\frac{1}{2-u\alpha}\right)^{\frac{1}{\eta_c+1}} & else \end{cases}$$

  Compute the children as

$$c_{1,i} = \frac{1}{2}\left[(x_{1,i} + x_{2,i}) - \beta_q |x_{2,i} - x_{1,i}|\right]$$
$$c_{2,i} = \frac{1}{2}\left[(x_{1,i} + x_{2,i}) + \beta_q |x_{2,i} - x_{1,i}|\right]$$

end
Output $\boldsymbol{c}_1$ and $\boldsymbol{c}_2$.

Algorithm 4: Parameter-based Mutation

Input is a child individual $\boldsymbol{c}$.
for $i \in 1, ..., n$ **do**
  Create a random number $u \in [0, 1]$ and compute

$$\delta = min\left((c_i - x_{lower,i}), (x_{upper,i} - c_i)\right)$$

  if $u \le \frac{1}{2}$ **then**
    $\delta_q = \left(2u + (1 - 2u)(1 - \delta)^{\eta_m+1}\right)$
  end
  else
    $\delta_q = 1 - (2(1 - u) + 2(u - \frac{1}{2})(1 - \delta)^{\eta_m+1})^{\frac{1}{\eta_m+1}}$
  end
  Compute

$$c_i = x_i + \delta_q(x_{upper,i} - x_{lower,i})$$

end
Output $\boldsymbol{c}$

4 IMPLEMENTATION IN A HPC FRAMEWORK

4.1 Simulation software

We performed the simulations of the self-propelled swimmers using an extended version of MRAG-I2D [30], an open source software framework[1] which implements various strategies to efficiently map the irregular computational workload of wavelet-adapted grids on multi-core nodes. The extensions to the code comprise of a novel algorithm to obtain the pointwise force distribution on solid objects (Eqs. 11 and 12) which is necessary for estimating the CoT, and an improved tree-code for implementing the Fast Multipole Method [12]. The parallelization of MRAG-I2D is based on both the Intel Threading Building Blocks library and OpenMP. The most computationally-intensive components of the solver, corresponding to 95% of the simulation time, are based on tree-codes which utilize multipole expansions [12], exploit loop and task-based parallelism by means of OpenMP, and have been explicitly vectorized using the Intel SPMD Program Compiler [28]. MRAG-I2D exhibits unprecedented performance in terms of time-to-solution for a series of benchmark problems, ranging from inviscid evolution of an elliptical vortex, flow past impulsively started cylinders for $Re = 40$ to 40000, and simulations of self-propelled anguilliform swimmers [30].

4.2 Task-parallel library

The parallel implementation of the optimization algorithms is based on the TORC task-parallel library [14]. TORC provides a programming and runtime environment similar to OpenMP tasks, but allows parallel programs to run on both shared and distributed memory systems. MPI applications run on the compute nodes with one or more workers, and can submit tasks for asynchronous execution from any nesting level of parallelism. The library exports a C/C++ and Fortran interface, provides transparent task and data management, and supports load balancing through intra- and inter-node work stealing, enhanced by a set of task distribution policies that can be applied by the programmer. The parallelization approach requires minimal modification of sequential algorithms, and is platform-agnostic.

TORC has already been used for the parallelization of algorithms for single objective stochastic optimization and uncertainty quantification, as part of the Π4U [13] framework[2]. It has been shown to achieve excellent parallel efficiency on up to 512 compute nodes of Piz Daint, even in cases where the simulation time exhibits significant variance depending on the input parameters. In the context of the current work, function-evaluations correspond to multi-core simulations performed by MRAG-I2D, using input parameters specified by the optimization algorithm. We utilize a single worker per compute node, launch MRAG-I2D with the fork-exec system calls, and perform data exchange through the local filesystem.

4.3 Task scheduling

The large variance of the MRAG-I2D simulation-time introduces load imbalance that can lead to poor hardware utilization. This

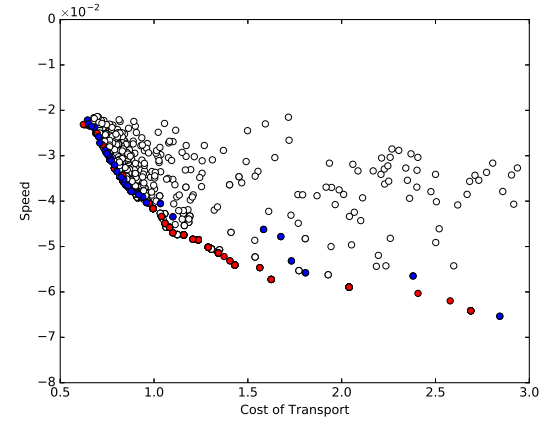
[1] https://github.com/rossinelli/2d-treecodes
[2] http://www.cse-lab.ethz.ch/software/Pi4U

can be avoided by setting the population size of the optimization algorithms to be larger than the number of available workers, and exploiting the task-stealing mechanism of the library. To further assist this approach, at every optimization step, function-evaluation tasks with larger expected runtime are scheduled first. In particular, we sort the tasks in descending order, distribute cyclically the first W of them to the W available workers, and insert the remaining to a single queue which is checked by idle workers for pending tasks.

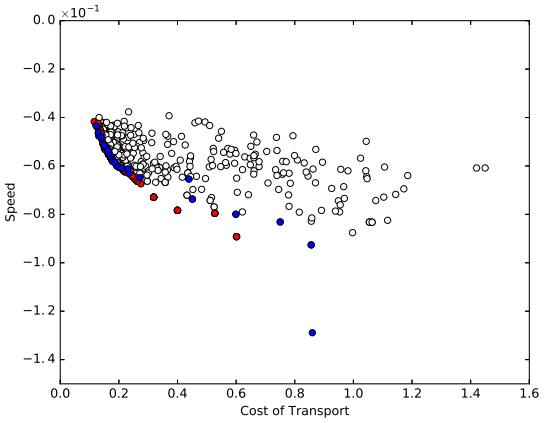
To estimate the expected runtime for each function evaluation of a new generation, we exploit the information available from all previous generations. More specifically, we design a fully connected artificial neural network, with one hidden layer consisting of 32 neurons and a non-saturating activation function. The training dataset consists of the simulation parameters (input) and the corresponding estimated runtimes (output). We optimize the mean squared error of the network using the L-BFGS optimization algorithm. The open source OpenANN library[3] was used for the implementation of the neural network.

5 RESULTS

The optimization studies were conducted at two different Reynolds numbers, Re = 400 and Re = 4000, in order to characterize differences in the swimming behaviour of fish larvae and adults. A single optimization campaign required several generations of swimmers (convergence was observed after approximately 20 generations), with each generation consisting of multiple distinct simulations running in parallel. Each simulation comprised a single individual of the population, and utilized 8 threads (spanning an entire compute node) to average a wall-clock time of approximately 3 hours. With the progression of the optimization, the population slowly converged to optimal solutions. The individuals that comprise the best possible solutions from the final generation constitute the 'Pareto front', as shown in Fig. 2. All of the individuals that lie on the Pareto front represent optimal solutions, albeit with different physical characteristics. The figure also depicts a variety of non-optimal individuals with different parameter values, that were obtained as part of the optimization process. The entire population shown in Figs. 2a and 2b is comprised of individuals obtained from both the MO-CMA-ES and the NSGA-II algorithms. A select few of these were chosen for further analysis, namely an efficient, a generalist, and a fast swimmer. The burst-coast characteristics and energetics-performance of these individuals are discussed in the following subsections.



(a)



(b)

Figure 2: Pareto front obtained from the optimization of burst-and-coast swimmers at (a) Re = 400, and (b) Re = 4000. The front obtained from the NSGA-II algorithm is shown in blue, whereas the front obtained using MO-CMA-ES is shown in red. The dominated individuals (non-optimal) are depicted as open circles.

5.1 Vorticity field and burst-coast characteristics

The flow-fields generated by an adult steady swimmer, and by three different adult individuals obtained as outcomes of the optimization procedure, are shown in Fig. 3. In a given time, the swimmers cover varying distances owing to differences in their average speed, which is a consequence of adopting different swimming patterns. There are notable differences in the wake-vortices, influenced by the particular intermittent modes selected by the different swimmers. From Fig. 3a, we observe that the steady swimmer generates regularly spaced wake vortices, owing to its sinusoidally varying body undulations. The efficient adult swimmer (Figs. 3b-3e, Movie 1) undergoes a long coasting period (t_{coast}), which results in minimal vortex-shedding into the surrounding flow-field. Upon resuming body-undulations, the swimmer sheds three distinct vortices (Fig. 3e), which resemble the trio of vortices generated during the initial asymmetric start-up. The 'generalist swimmer', i.e., the optimal individual that embodies a useful compromise between the two conflicting objectives (i.e., high speed and high energy efficiency), spends a measurable amount of time swimming steadily, and experiences relatively gentle accelerations. The resulting wake vortices resemble a combination of those generated by the steady (Fig. 3a) and the most efficient swimmers. The fast adult swimmer, on the other hand, utilizes extremely short speed-up and slow-down times,

[3] https://github.com/OpenANN/OpenANN

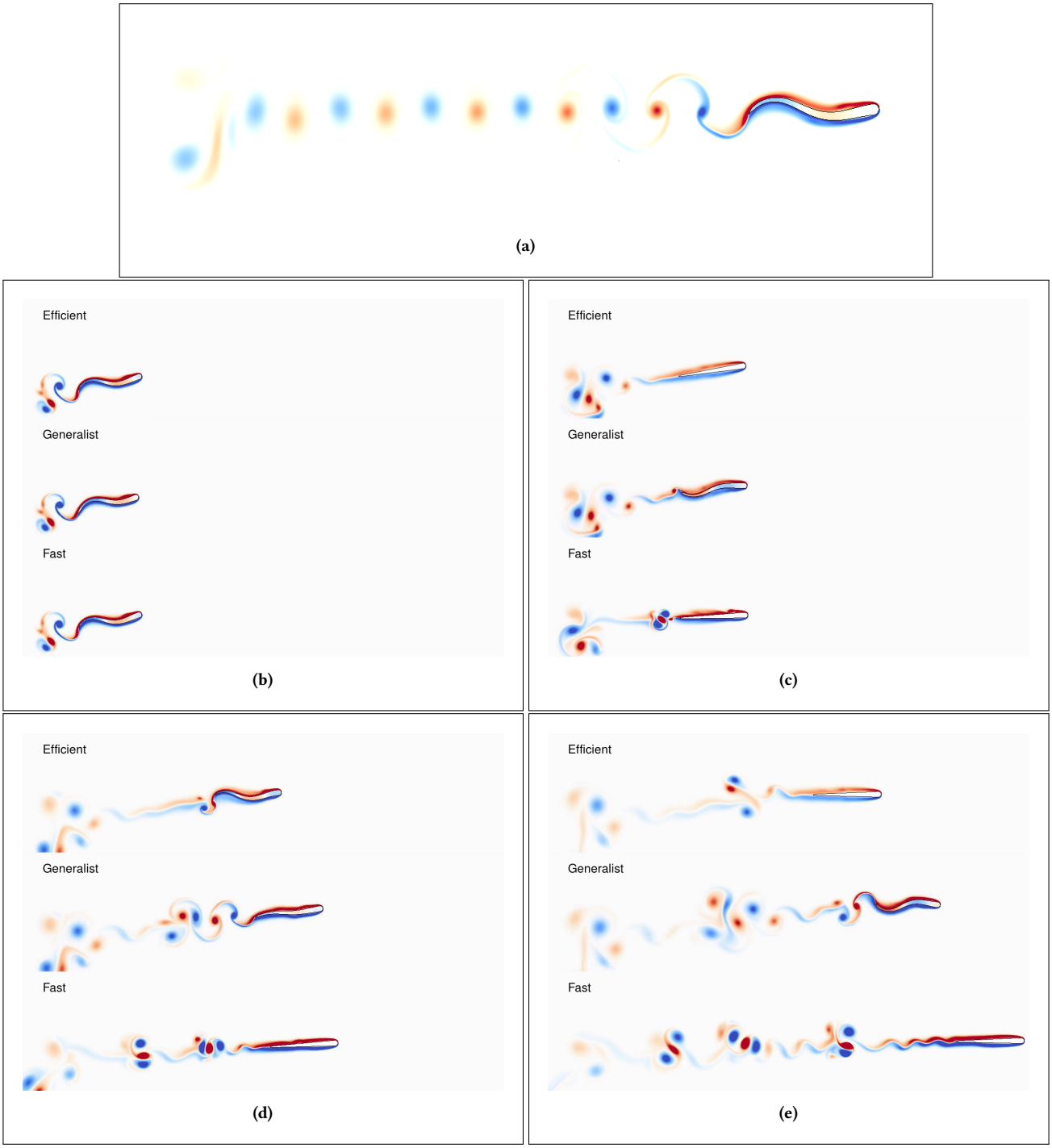


Figure 3: The vorticity field generated by (a) a steady adult swimmer; (b) the efficient, the generalist, and the fast swimmers at $t = 2.0$ (right before the initiation of burst-coast cycles); (c) at $t = 4.5$; (d) at $t = 7.0$; and (e) at $t = 9.5$. Movie 1 shows an animation of the flow-fields generated by the adult swimmers. Snapshots of the flow-fields generated by the larval swimmers are not shown here, but the relevant animation may be found in Movie 2.

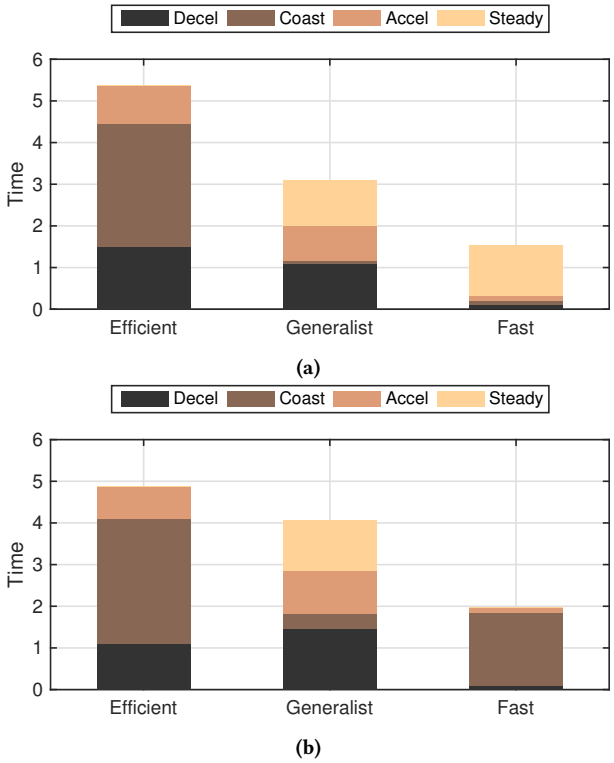


Figure 4: Breakdown of the burst-coast cycle time for the intermittent (a) larval swimmers (Re = 400), and (b) adult swimmers (Re = 4000).

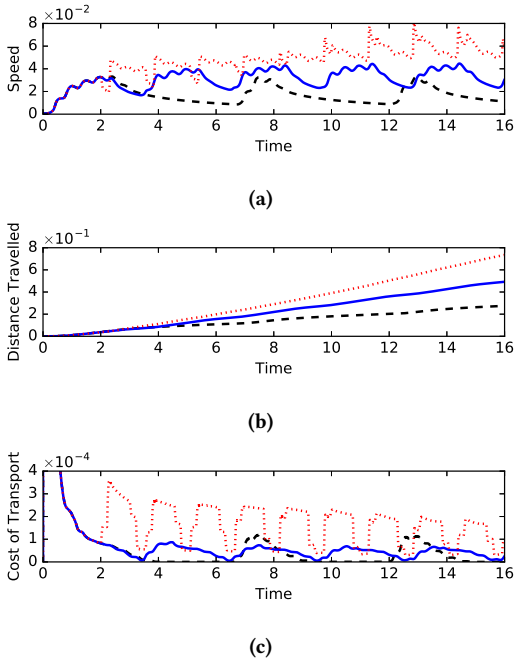


Figure 5: (a) Speed, (b) distance travelled, and (c) cost of transport for three different larval swimmers (Re = 400). The dashed black line corresponds to the efficient swimmer, the solid blue line to the generalist swimmer, and the dotted red line to the fast swimmer.

and correspondingly large accelerations. The relevant tail-beat motion allows the fast swimmer to rapidly accelerate packets of fluid backward, which in turn generates large forward thrust. Moreover, the acceleration phase is followed by a short coasting phase, which allows the adult to move out of the unsteady transient it generates, before initiating another acceleration cycle. The resulting strong vortices that emerge in the wake are visible in Figs. 3c-3e.

Analyzing the burst-coast cycle-time quantitatively in Fig. 4, we realize that the most efficient swimmers spend a negligible amount of time swimming with a steady gait, irrespective of the developmental stage they may be in (i.e., larva or adult). This correlates well with experimental observations, where fish executing burst-and-coast patterns are rarely observed to employ steady motion. We notice striking similarities between the burst-coast time-profiles for both the adult and larval efficient swimmers; there is an initial deceleration, followed by more than 50% of the time spent coasting, which is followed by an acceleration phase with almost no steady swimming in between. The generalist adult swimmer spends most of its time executing active motion, and coasts only for a short duration (Fig. 4). The burst-coast profile of the generalist larva is similar in nature, with the larva also adopting a short coasting stage.

The most notable differences appear in the burst-coast time-profiles of the fast adult and larval swimmers; while both of these adopt very short speed-up and slow-down times (t_{Accel} and t_{Decel}), giving rise to correspondingly large accelerations, the larva prefers to spend the majority of its time executing steady swimming, whereas the adult swimmer exhibits a strong preference for coasting. This is an important result, since the only difference in the two cases is the Reynolds number. The reluctance of the larva to undergo coasting may reflect the fact that it experiences large deceleration during coasting, due to high viscous drag, which is not the case for the adult swimmer. It is crucial that the optimizer is able to discern this key physical difference, which has a purely hydrodynamic origin. Furthermore, the propensity of both the fast larva and the adult to accelerate a significant mass of water in a short time is reminiscent of the strategy used by fish executing escape manoeuvres in life-threatening situations [7, 11, 24]. We also note that sudden acceleration/deceleration of fast individuals followed by short coasting periods is reminiscent of locomotion patterns observed in cephalopods, which have evolved specifically for high speed manoeuvring, albeit with the disadvantage of a high metabolic rate [32].

5.2 Performance metrics

The performance metrics of interest, namely, the speed, the distance traversed, and the Cost of Transport (CoT, Eq. 18) are shown in Fig. 5 for the larval swimmer, and in Fig. 6 for the adult swimmer. In both situations, we clearly observe the increasing trend in instantaneous speed and net distance traversed, as we move from the efficient, to the generalist, to the fast swimmer. For the sake of

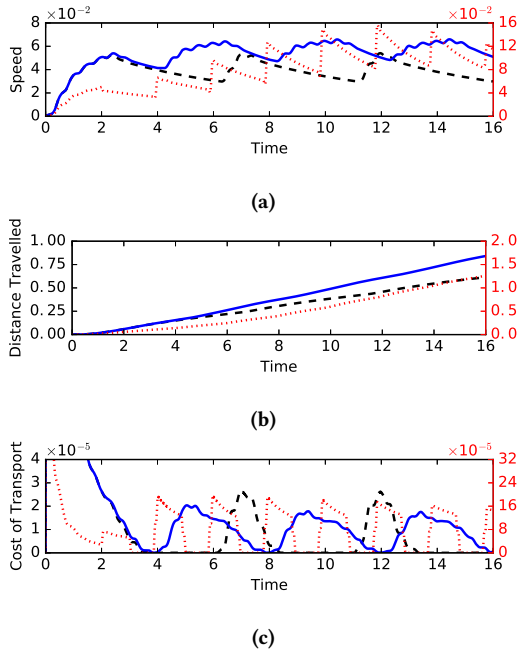


Figure 6: (a) Speed, (b) distance travelled, and (c) cost of transport for three different adult swimmers (Re = 4000). The dashed black line corresponds to the efficient swimmer, the solid blue line to the generalist swimmer, and the dotted red line to the fast swimmer. The secondary y-axes shown on the right hand side of each figure correspond to the fast adult swimmer.

clarity, the metrics for the fast adult swimmer have been plotted using a different y-range, shown on the right hand side of the graphs (Fig. 6). Importantly, the peak CoT values (Figs. 5c and 6c) show a marked increase in the case of the fast swimmers (approximately 2× for the larva, and 8× for the adult), but are comparable for the efficient and the generalist swimmers. The peak power output is usually constrained by the musculature of an organism, in addition to internal metabolic limitations, which may explain why the fast intermittent kinematics observed in Movie 1 is generally not encountered in nature [8].

The fitness values, computed by averaging the appropriate quantities over the last two burst-coast cycles, are listed in Tables 1 and 2 for the larval and adult swimmers, respectively. Towards the end of a swimming bout (marked by the termination time t = 16 for all cases), the fast larva experiences close to a 47% increase in average speed compared to a steadily-swimming larva (Table 1). However, this increase in speed is accompanied by a 170% increase in the amount of energy spent for covering a unit distance (i.e., the CoT). On the other hand, the burst-coast mode allows the most efficient larva to reduce energy-consumption by 38%, but with a reduction of 47% in average speed. Similarly, the generalist larva reduces energy consumption by 13% and average speed by 16%.

The benefits of burst-coast swimming are more pronounced in the case of the adult swimmer, as can be surmised by examining

Table 1: Fitness data for larval swimmers at Re = 400. The values have been normalized with respect to the data for the Steady swimmer. The un-normalized values are available in Fig. 7a.

Swimmer	CoT	Avg. Speed
Steady	1.00	1.00
Efficient	0.62	0.53
Generalist	0.87	0.84
Fast	2.70	1.47

Table 2: Fitness data for adult swimmers at Re = 4000. The values have been normalized with respect to the data for the Steady swimmer. The un-normalized values are available in Fig. 7b.

Swimmer	CoT	Avg. Speed
Steady	1.00	1.00
Efficient	0.48	0.63
Generalist	0.77	0.88
Fast	3.57	1.95

the data in Table 2. The most efficient adult reduces energy consumption by 52%, but with only a 37% drop in average speed. The generalist adult swimmer shows excellent performance gains, and reduces energy expenditure by almost a quarter, with merely a 12% reduction in average speed. The fast adult swimmer sees a 2× speed-up compared to a steady swimmer, but with an immense increase (257%) in CoT. Comparing the un-normalised values for average speed and the CoT in Fig. 7, we observe a positive correlation between the two quantities. Moreover, comparing absolute values for the larval and the adult swimmers, we notice that the adult is capable of attaining much higher speeds than the larva, at the expense of much lower CoT, which supports the hypothesis that rapid increase of body-length is a high priority for fish larvae, so as to escape the detrimental effects of high viscosity [26].

These results suggest that it is feasible to discover a diverse array of optimal locomotion techniques by combining high-fidelity simulations with multi-objective optimization techniques, especially when conflicting objectives may need to be considered simultaneously. Future investigations will build upon the current work by examining the optimal swimmers' motion from a hydrodynamic perspective, with the aim of better understanding the physics driving the evolution of intermittent locomotion in fish-like swimmers.

6 PERFORMANCE ANALYSIS

An extensive performance study of MRAG-I2D on multi-core CPUs is reported in [29], where speedups obtained by simulating various physical systems were measured on different multi-core architectures, using up to 16 cores. Figure 8 shows a strong scaling study of MRAG-I2D on a single compute node of Piz Daint. The input parameters correspond to the best fitness value obtained by the multi-objective optimization algorithm. We observe that the parallel efficiency is equal to 76% on 8 cores, which may be attributed to the relatively small problem size of the particular simulation case.

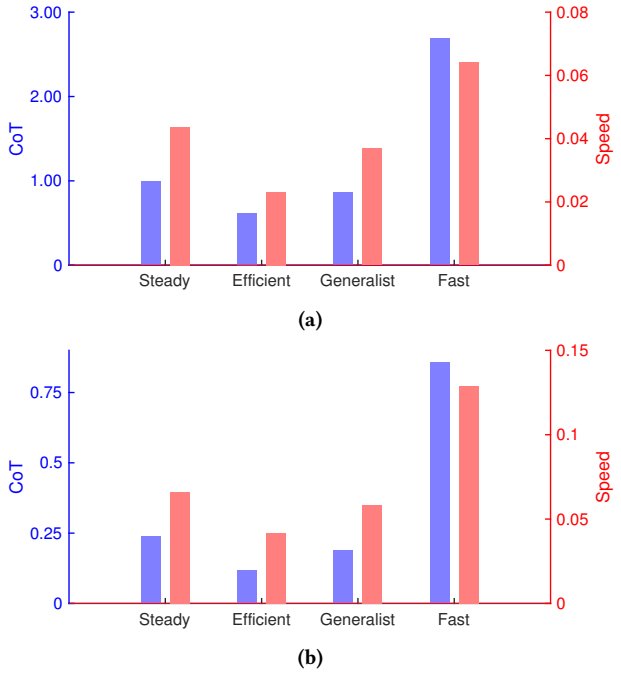


Figure 7: Absolute values of the CoT and average speed for the intermittent (a) larval swimmers at $Re = 400$, and (b) adults at $Re = 4000$.

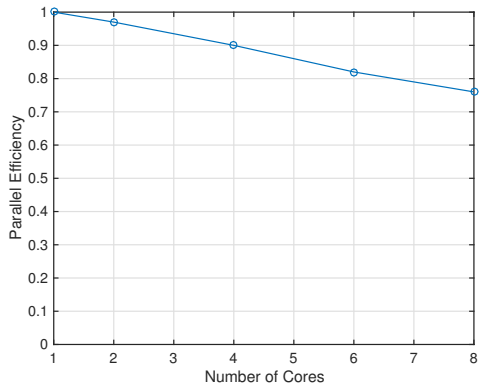


Figure 8: Strong scaling efficiency of MRAG-I2D on a single compute node of Piz Daint.

We run the task parallel optimization algorithms with a single worker per compute node. This allows us to avoid hardware contention between multiple running instances of MRAG-I2D, and to exploit the lower time-to-solution on 8 cores, thus minimizing the overall time required by the optimization algorithm.

To overcome load imbalance and maximize hardware utilization, we schedule tasks according to their predicted runtime, and require that the population size of the optimization algorithms be at least double the number of available workers (nodes). Figure 9a presents a strong scaling study for a single generation of the task parallel implementation of NSGA-II on up to 32 compute node of Piz Daint,

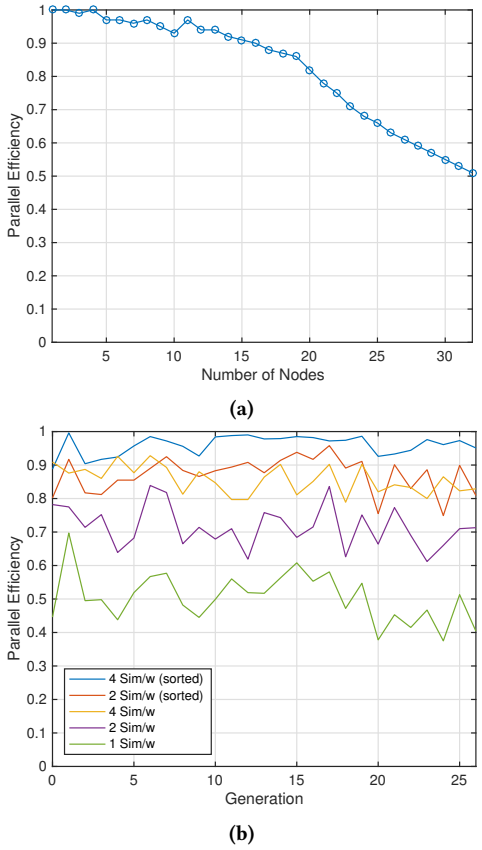


Figure 9: (a) Strong scaling study for a single generation of NSGA-II with population size 32 on up to 32 nodes of Piz Daint. (b) Parallel efficiency of NSGA-II on Piz Daint with and without the sorting algorithm implemented.

for a population size of 32 individuals. We observe that the parallel efficiency remains above 90% on up to 16 nodes (2 simulations per worker). It declines rapidly when more than 20 nodes are used, reaching 50% when only one simulation is performed on each node at every step of the algorithm.

Figure 9b shows the computed parallel efficiency for 27 generations of NSGA-II, when each worker executes 1, 2, and 4 simulations per generation on average (simulation per worker - Sim/w). If the number of tasks is equal to the number of workers (i.e. 1 Sim/w), the average parallel efficiency is only 50.3%. When task scheduling is based on sorted runtimes that have been predicted by the neural network, the efficiency increases from 71.6% to 87.0% and from 85.6% to 95.9% for 2 and 4 Sim/w, respectively. This behavior depends only on the Sim/w ratio, and is expected to hold for a much larger number of nodes than used for the present study. The performance achieved by the machine-learning based task-scheduling almost coincides ($< 0.2\%$ relative error) with the ideal case where the exact simulation times are known beforehand.

7 CONCLUSION

We have coupled task-parallel multi-objective optimization algorithms with high-fidelity simulations of self-propelled swimmers, to discover efficient intermittent locomotion modes in self-propelled swimmers. This approach has allowed us to discover a diverse range of motion patterns, each of which are optimal in their own right, but give rise to very different locomotion characteristics. The most efficient swimmers discovered by the optimization algorithms display prolonged coasting behaviour, which allows them to conserve energy. The fastest swimmers were able to generate large forward thrust by rapidly accelerating 'packets' of fluid opposite to their direction of motion. Intriguingly, the fast larva and the fast adult adopt markedly differently burst-coast behaviour, which may be ascribed to the increased viscous drag experienced by the larva during coasting. This indicates that the optimization algorithms are capable of correctly accounting for expected differences in the physics of the problem. The primary advantage of using such computational tools lies in the fact that it can lead to the discovery of patterns which may not generally occur in nature, but can be invaluable for use in robotic applications; optimal swimming gaits can be selected specifically to suit particular mission-requirements, e.g., to maximize energy conservation, minimize travel time, or utilize an optimal combination of both. The resulting increase in range, endurance, and average speed can greatly enhance the mission capability of robotic swimmers.

8 ACKNOWLEDGMENTS

This work was supported by the European Research Council Advanced Investigator Award (341117), and the SNSF Sinergia Award (CRSII3 147675). Computational resources were provided by the Swiss National Supercomputing Center (CSCS) under project ID 's658'.

REFERENCES

[1] P. Angot, C. H. Bruneau, and P. Fabrie. 1999. A penalization method to take into account obstacles in incompressible viscous flows. *Numer. Math.* 81 (1999), 497–520.
[2] M. Bergmann and A. Iollo. 2011. Modeling and simulation of fish-like swimming. *J. Comput. Phys.* 230 (2011), 329–348.
[3] R. W. Blake. 1983. Functional design and burst-and-coast swimming in fishes. *Canadian Journal of Zoology* 61, 11 (1983), 2491–2494.
[4] M-H Chung. 2009. On burst-and-coast swimming performance in fish-like locomotion. *Bioinspir. Biomim.* 4, 3 (2009), 036001.
[5] K. Deb and S. Agrawal. 1999. *A Niched-Penalty Approach for Constraint Handling in Genetic Algorithms.* Springer Vienna, Vienna, 235–243.
[6] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. 2000. A Fast and Elitist Multi-Objective Genetic Algorithm: NSGA-II. (2000).
[7] P. Domenici and R. Blake. 1997. The kinematics and performance of fish fast-start swimming. *J. Exp. Biol.* 200, 8 (1997), 1165–1178.
[8] C. Eloy. 2013. On the best design for undulatory swimming. *J. Fluid Mech.* 717 (25 002 2013), 48–89.
[9] L. A. Fuiman and P. W. Webb. 1988. Ontogeny of routine swimming activity and performance in zebra danios (Teleostei: Cyprinidae). *Anim. Behav.* 36, 1 (1988), 250–261.
[10] M. Gazzola, P. Chatelain, W. M. van Rees, and P. Koumoutsakos. 2011. Simulations of single and multiple swimmers with non-divergence free deforming geometries. *J. Comput. Phys.* 230 (2011), 7093–7114.
[11] M. Gazzola, W. M. van Rees, and P. Koumoutsakos. 2012. C-start: optimal start of larval fish. *J. Fluid Mech.* 698 (May 2012), 5–18.
[12] L. Greengard and V. Rokhlin. 1987. A fast algorithm for particle simulations. *J. Comput. Phys.* 73 (1987), 325–348.
[13] P. E. Hadjidoukas, P. Angelikopoulos, L. Kulakova, C. Papadimitriou, and P. Koumoutsakos. 2015. Exploiting Task-Based Parallelism in Bayesian Uncertainty Quantification. In *European Conference on Parallel Processing*. Springer Berlin Heidelberg, Berlin, Heidelberg, 532–544.
[14] P. E. Hadjidoukas, E. Lappas, and V. V. Dimakopoulos. 2012. A runtime library for platform-independent task parallelism. In *20th Euromicro International Conference on Parallel, Distributed and Network-based Processing*. IEEE, 229–236.
[15] N. Hansen and A. Ostermeier. 2001. Completely Derandomized Self-Adaptation in Evolution Strategies. *Evol. Comput.* 9, 2 (June 2001), 159–195.
[16] C. Igel, N. Hansen, and S. Roth. 2007. Covariance Matrix Adaptation for Multi-objective Optimization. *Evol. Comput.* 15, 1 (2007), 1–28.
[17] C. Igel, V. Heidrich-Meisner, and T. Glasmachers. 2008. Shark. *J. Mach. Learn. Res.* 9 (2008), 993–996.
[18] C. Igel, T. Suttorp, and N. Hansen. 2007. *Steady-State Selection and Efficient Covariance Matrix Update in the Multi-objective CMA-ES.* Springer Berlin Heidelberg, Berlin, Heidelberg, 171–185.
[19] S. Kern and P. Koumoutsakos. 2006. Simulations of optimized anguilliform swimming. *J. Exp. Biol.* 209 (2006), 4841–4857.
[20] P. Koumoutsakos and A. Leonard. 1995. High-resolution simulations of the flow around an impulsively started cylinder using vortex methods. *J. Fluid Mech.* 296 (1995), 1–38.
[21] D. L. Kramer and R. L. McLaughlin. 2001. The Behavioral Ecology of Intermittent Locomotion. *Amer. Zool.* 41, 2 (2001), 137–153.
[22] M. J. McHenry and G. V. Lauder. 2005. The mechanical scaling of coasting in zebrafish (Danio rerio). *J. Exp. Biol.* 208, 12 (2005), 2289–2301.
[23] U. K. Müller, E. J. Stamhuis, and J. J. Videler. 2000. Hydrodynamics of unsteady fish swimming and the effects of body size: comparing the flow fields of fish larvae and adults. *J. Exp. Biol.* 203, 2 (2000), 193–206.
[24] U. K. Müller, J. G. M. van den Boogaart, and J. L. van Leeuwen. 2008. Flow patterns of larval fish: undulatory swimming in the intermediate flow regime. *J. Exp. Biol.* 211, 2 (2008), 196–205.
[25] U. K. Müller, J. L. van Leeuwen, S. van Duin, and H. Liu. 2009. An Un-Momentous Start to Life: Can Hydrodynamics Explain Why Fish Larvae Change Swimming Style? *J. Biomech. Sci. Eng.* 4, 1 (2009), 37–53.
[26] U. K. Müller and J. J. Videler. 1996. Inertia as a 'safe harbour': do fish larvae increase length growth to escape viscous drag? *Rev. Fish Biol. Fish.* 6, 3 (1996), 353–360.
[27] P. Paoletti and L. Mahadevan. 2014. Intermittent locomotion as an optimal control strategy. *Proc. Royal Soc. London A: Math. Phys. Eng. Sci.* 470, 2164 (2014).
[28] M. Pharr and W. R Mark. 2012. ispc: A SPMD compiler for high-performance CPU programming. In *Innovative Parallel Computing (InPar), 2012.* IEEE, 1–13.
[29] D. Rossinelli, B. Hejazialhosseini, M. Bergdorf, and P. Koumoutsakos. 2011. Wavelet-adaptive solvers on multi-core architectures for the simulation of complex systems. *Concurr. Comput.* 23, 2 (2011), 172–186.
[30] D. Rossinelli, B. Hejazialhosseini, W. M. van Rees, M. Gazzola, M. Bergdorf, and P. Koumoutsakos. 2015. MRAG-I2D: Multi-resolution adapted grids for remeshed vortex methods on multicore architectures. *J. Comput. Phys.* 288 (2015), 1–18.
[31] N. Srinivas and K. Deb. 1994. Muiltiobjective Optimization Using Nondominated Sorting in Genetic Algorithms. *Evol. Comput.* 2, 3 (Sept. 1994), 221–248.
[32] A. R. Tanner, D. Fuchs, I. E. Winkelmann, M. T. P. Gilbert, M. Sabrina Pankey, A. M. Ribeiro, K. M. Kocot, K. M. Halanych, T. H. Oakley, R. R. da Fonseca, D. Pisani, and J. Vinther. 2017. Molecular clocks indicate turnover and diversification of modern coleoid cephalopods during the Mesozoic Marine Revolution. *Proc. R. Soc. Lond. B: Biol. Sci.* 284, 1850 (2017).
[33] G. Tokić and D. K. P. Yue. 2012. Optimal shape and motion of undulatory swimming organisms. *Proc. Roy. Soc. London B: Biol. Sci.* (2012).
[34] E. D. Tytell and G. V. Lauder. 2004. The hydrodynamics of eel swimming. *J. Exp. Biol.* 207 (2004), 1825–1841.
[35] W. M. van Rees, M. Gazzola, and P. Koumoutsakos. 2013. Optimal shapes for anguilliform swimmers at intermediate Reynolds numbers. *J. Fluid Mech.* 722 (5 2013), R3 1–12.
[36] W. M. van Rees, M. Gazzola, and P. Koumoutsakos. 2015. Optimal morphokinematics for undulatory swimmers at intermediate Reynolds numbers. *J. Fluid Mech.* 775 (25 007 2015), 178–188.
[37] J. J. Videler, E. J. Stamhuis, U. K. Müller, and L. A van Duren. 2002. The Scaling and Structure of Aquatic Animal Wakes. *Integr. Comp. Biol.* 42, 5 (2002), 988–996.
[38] J. J. Videler and D. Weihs. 1982. Energetic advantages of burst-and-coast swimming of fish at high speeds. *J. Exp. Biol.* 97, 1 (1982), 169–178.
[39] D. Weihs. 1974. Energetic advantages of burst swimming of fish. *J. Theor. Biol.* 48, 1 (1974), 215–229.
[40] S. Wieskotten, G. Dehnhardt, B. Mauck, L. Miersch, and W. Hanke. 2010. The impact of glide phases on the trackability of hydrodynamic trails in harbour seals (Phoca vitulina). *J. Exp. Biol.* 213, 21 (2010), 3734–3740.
[41] G. Wu, Y. Yang, and L. Zeng. 2007. Kinematics, hydrodynamics and energetic advantages of burst-and-coast swimming of koi carps (Cyprinus carpio koi). *J. Exp. Biol.* 210, 12 (2007), 2181–2191.