# Articulated Swimming Creatures

Jie Tan[*]
Georgia Institute of Technology

Yuting Gu[†]

Greg Turk[‡]
Georgia Institute of Technology

C. Karen Liu[§]
Georgia Institute of Technology

## Abstract

We present a general approach to creating realistic swimming behavior for a given articulated creature body. The two main components of our method are creature/fluid simulation and the optimization of the creature motion parameters. We simulate two-way coupling between the fluid and the articulated body by solving a linear system that matches acceleration at fluid/solid boundaries and that also enforces fluid incompressibility. The swimming motion of a given creature is described as a set of periodic functions, one for each joint degree of freedom. We optimize over the space of these functions in order to find a motion that causes the creature to swim straight and stay within a given energy budget. Our creatures can perform path following by first training appropriate turning maneuvers through offline optimization and then selecting between these motions to track the given path. We present results for a clownfish, an eel, a sea turtle, a manta ray and a frog, and in each case the resulting motion is a good match to the real-world animals. We also demonstrate a plausible swimming gait for a fictional creature that has no real-world counterpart.

**CR Categories:** I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Animation; I.6.8 [Simulation and Modeling]: Types of Simulation—Animation.

**Keywords:** Swimming, articulated figures, fluid simulation, optimization

**Links:** DL PDF WEB VIDEO

## 1 Introduction

The oceans, lakes and rivers of our planet contain a wide variety of creatures that use swimming as their primary form of locomotion. There are an astonishing variety of body shapes and patterns of motion that are used by swimmers across the animal kingdom. Some of the many creature swimming patterns from nature include using thrust from a tail, moving an elongated body sinusoidally, using paddle-like motions of flippers, kicking with legs, and gentle bird-like flapping of fins. Our research goal is to develop a general platform for finding efficient swimming motion for a given creature body shape. There are a number of application areas that can benefit from realistic swimming simulation, including feature film animation [Stanton and Unkrich 2003], biological investigation of swimming mechanics [Kern and Koumoutsakos 2006; Shirgaonkar et al. 2008], locomotion of user-created creatures in video games [Hecker et al. 2008], and the invention of new modes of propulsion for underwater vehicles [Barrett et al. 2002].

[*]e-mail:jtan34@gatech.edu
[†]e-mail:windygyt@gmail.com
[‡]e-mail:turk@cc.gatech.edu
[§]e-mail:karenliu@cc.gatech.edu

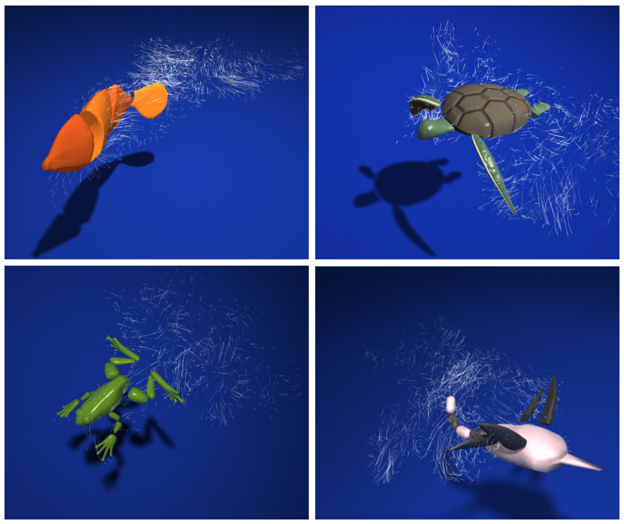

**Figure 1:** *Aquatic creatures with different shapes swim in a simulated fluid environment. The particle traces show the fluid flow near the swimmers. Our method provides a generic framework to discover natural swimming gaits and to simulate the swimming motion for a wide variety of animal bodies.*

Today, most scientific models for swimming motion are customized to specific species with predefined locomotion patterns [Shirgaonkar et al. 2008]. These models are highly accurate but are difficult to generalize to a variety of creatures. The existing 3D swimming animations, on the other hand, demonstrate a life-like underwater ecosystem with rich variety of creatures. However, their motions are typically animated manually or based on simplified physical models. Having a generic set of tools that can produce physically realistic aquatic motion for a wide array of creatures is challenging and has not been shown in previous work.

At the heart of synthesizing realistic aquatic locomotion lies the problems of simulation and control. Solving these two problems simultaneously under hydrodynamics presents some unique challenges. First, the relation between the movement of the aquatic animal and the forces exerted by surrounding fluid is extremely complex. Thus it is difficult to solve using an optimization approach. Any small changes in undulation or flapping gait can result in drastically different control strategies. In addition, the morphology of aquatic animals is astonishingly diverse and results in fundamentally different locomotion mechanisms. Designing control strategies based on ad-hoc observation or careful tuning of parameters would be extraordinarily difficult to generalize to the vast biodiversity found in nature.

This paper describes a complete system for controlling a wide variety of aquatic animals in a simulated fluid environment. Our goal is a system that balances between physical realism and generality. Given an aquatic animal that is represented by an articulated rigid body system, our system can automatically find the optimal locomotion in a hydrodynamically-coupled environment. Our system does not require any prior knowledge of the animal's behavior and minimizes the effort of manually tuning the physical and control parameters.

The system consists of two main components: simulating motion and optimizing control strategies. We simulate articulated rigid bodies submerged in invisid, incompressible fluid governed by the Navier-Stokes equations. The animal can exert torques to exercise each actuated joint. Through accurate two-way coupling of the rigid bodies and the fluid, the joint motion will lead to *some* locomotion in the fluid, but purposeful and balanced locomotion requires careful coordination and synchronization among those actuated joints. The second component provides an automatic way to discover joint motion that achieves a desired goal in locomotion (i.e. a joint motion that yields the fastest or the most energy efficient locomotion). We employ an optimization technique called Covariance Matrix Adaptation (CMA) to explore the domain of possible joint trajectories.

We evaluate our system by demonstrating optimized swimming gaits for a wider variety of aquatic animals and swimming strategies, including clownfish, eels, sea turtles, frogs, manta rays and some imaginary creatures. In addition, we compare the swimming motion in a Navier-Stokes fluid with motion in a simplified fluid. Our results show that these motions can differ dramatically depending on which fluid model is used.

## 2 Related Work

Prior research that has the most influence on our work include techniques from articulated figure control, approaches to solid-fluid coupling, and simulated creature locomotion. We will review the work in each of these sub-areas in turn.

### 2.1 Articulated Figure Control

Controlling a physically simulated articulated figure has been extensively studied in both computer animation and robotics. Hodgins et al. [1995] demonstrated that sophisticated biped controllers, such as gymnastic vaulting or tumbling, can be constructed based upon simple proportional-derivative (PD) controllers that track individual joints. This control framework has then been widely applied to physically simulate different types of human activities [Laszlo et al. 1996; Faloutsos et al. 2001; Zordan and Hodgins 2002; Yin et al. 2007; Sok et al. 2007]. As controllers become more complex, the domain of control parameters expands and tuning the parameters becomes less intuitive. Many researchers exploited optimization techniques or optimal control theory to improve the robustness of the controllers, as well as the quality of motions they produce. For example, the gains of feedback controllers can be optimized based on the dynamic state of balance in the reference trajectory [da Silva et al. 2008; Muico et al. 2009; Ye and Liu 2010]. Continuation methods can be used to design challenging controllers in an adaptive fashion [Yin et al. 2008]. Stochastic optimization algorithms, such as Covariance Matrix Adaptation (CMA) [Hansen and Kern 2004], have also been applied to search control parameters when the problem domain is highly discontinuous [Wu and Popović 2010; Wang et al. 2010; Mordatch et al. 2010]. Our method was inspired primarily by the success of applying CMA to optimal control problems demonstrated in the previous work. However, our work faces unique challenges because the controller is operated in a complex hydrodynamic environment with two-way coupling between the animal and the fluid.

### 2.2 Solid-Fluid Coupling

Many researchers have proposed various ways to simulate the two-way coupling between fluids and solids. Takahashi et al. [2002] presented a simple alternating two-way coupling method between fluids and solid objects. The velocities of the solid objects served as the boundary conditions for fluid motion while the pressure field solved from the Navier-Stokes equations was integrated at the solid surface to provide a net force and a net torque exerted on the solid objects. Arash et al. [2003] represented the solids by mass-spring models and fluids by marker particles. The interactions were calculated through the mutual forces between the marker particles and mass nodes at the interface. Carlson et al. [2004] proposed the rigid fluid method that treated solids as fluids at first and then projected the velocity field in the solid region onto a subspace satisfying the rigid constraints. Guendelman et al. [2005] made use of an alternating approach that is generalized to include octree and thin shells. They solved the pressure field for a second time by adding solid masses to the fluid grid density, which improves the pressure field. Klingner et al. [2006] used a tetrahedral mesh for accurate boundary discretization and extended the mass conservation (projection) step to include the dynamics of rigid body. This was extended to model the interaction between fluids and deformable bodies [Chentanez et al. 2006]. Batty et al. [2007] derived a fast variational approach that allowed sub-grid accuray using regular grids. Robinson et al. [2008] developed a generic and momentum conserving technique to couple fluids to rigid/deformable solids and thin shells. The coupled system is symmetric indefinite and solved using MINRES.

### 2.3 Simulated Swimmers

Although human motion has been the main focus in character animation, a number of researchers have achieved great realism in synthesizing the movement of other animals, such as worms [Miller 1989], fish [Tu and Terzopoulos 1994], birds [Wu and Popović 2003], dogs [Kry et al. 2009], and even imaginary creatures [Wampler and Popović 2009; Hecker et al. 2008].

Tu and Terzopoulos pioneered the animation of swimming fish using a a mass-spring system for the fish body and a simplified fluid model [Tu and Terzopoulos 1994; Terzopoulos et al. 1994; Grzeszczuk and Terzopoulos 1995]. They used simulated annealing and the simplex method to discover swimming gaits. Their simulation also incorporated vision sensors, motor controllers, and behavioral modeling of eating, escape, schooling and mating. The major difference between their paper and ours lies in the fluid model and the optimization technique. This early paper used a simplified fluid model while ours adopts a full Navier-Stokes solver and introduces a two-way coupling method between fluids and articulated figures in generalized coordinate.

Sims [1994] investigated the simulated evolution of creature locomotion. Sims' creatures were composed of blocks that are connected by articulated joints. He used genetic programming to evolve both the creature bodies and their controllers. In addition to walking and jumping behaviors, some of his creatures also learned to swim in a simplified fluid environment.

Wu and Popović [2003] used an articulated skeleton and deformable elements for feathers in order to animate the flight of birds. They used an optimization process to find the best wing beats in order to accurately follow a given path. Yang et al. [2004] used an articulated body representation, a simplified fluid model, and several layers of control to model human swimmers. Kwatra et al. [2009] used an articulated body representation and two-way coupling between the body and a fluid simulation to model human swimming. They used motion capture data of swimming motions as input to the swimmer control. Lentine et al. [2010] used an articulated skeleton with a deformable skin layer and two-way coupling to a fluid simulator to model figures that are moving in fluids. They optimized for certain styles of motion using objective functions designed for effort minimization and drag minimization/maximization. Their results also clearly demonstrated that using a full fluid simulator gives more realistic results than using a simplified fluid model.

In the field of computational fluid dynamics (CFD), there is a small but growing literature on the simulation of swimming creatures. These studies are typically focused on a single swimming style of one particular creature, and they usually make use of sophisticated fluid dynamics code, at a large cost in computational complexity, to generate more accurate and detailed fluid simulation. Often these studies are informed by laboratory studies of the creature in question, including flow data that has been gathered using methods such as particle image velocimetry [Grant 1997]. A good representative of such work is the investigation of Shiragaonkar et al. of the knifefish, which is a fish that propels itself using waves that travel along its elongated lower fin (gymnotiform swimming) [Shirgaonkar et al. 2008]. The simulator for this work used an immersed boundary method, and the simulations were performed on a 262 compute node Linux cluster. Another example of such a study is the work of Kern and Koumoutakos [2006] on the simulation of eels (anguilliform swimming). In this work, the fluid grid is matched to the eel body by using a cylindrical grid in most of the domain and a hemisphere-based grid for the head of the eel. They used the CMA technique [Hansen and Kern 2004] to optimize a five parameter motion model.

## 3 Coupling Articulated Figures with Fluids

We simulate fluids by solving the Navier-Stokes equations on a MAC grid and we simulate the articulated rigid body using generalized coordinates. We modify the projection step of the fluid solver to take into consideration the dynamics of the articulated figure.

### 3.1 Fluid Simulation

We simulate fluid using the inviscid, incompressible fluid equations (sometimes called the Euler equations):

$$\nabla \cdot \mathbf{u} = 0$$

$$\mathbf{u}_t = -(\mathbf{u} \cdot \nabla \mathbf{u}) - \frac{1}{\rho}\nabla p + \mathbf{f}$$

where $\mathbf{u} = (u, v, w)$ is the velocity of fluids, $p$ is the pressure, $\rho$ is the density and $\mathbf{f}$ accounts for the external body forces. We do not include a viscous term because such effects are negligable for the motion of the large animals in our examples. If we were studying swimming of millimeter sized creatures, however, incorporating viscous effects would be mandatory.

The standard way to solve the above equations on a MAC grid can be described in following two steps. First, we calculate an intermediate velocity field $\mathbf{u}^*$ by only considering the convection $\mathbf{u} \cdot \nabla \mathbf{u}$ and the body force $\mathbf{f}$:

$$\mathbf{u}^* = \mathrm{SL}(\mathbf{u}^n, \Delta t) + \Delta t \mathbf{f} \tag{1}$$

where $\mathbf{u}^n$ is the velocity at $n^{th}$ time step. We use the Semi-Lagrangian method [Stam 1999] to integrate the convection term and apply BFECC [Kim et al. 2007] to reduce the numerical dissipation.

Next, we solve the following Poisson equation with Neumann boundary conditions $\mathbf{u} \cdot \mathbf{n} = \mathbf{u}_{solid} \cdot \mathbf{n}$ at the solid boundary and Dirichlet boundary conditions $p = 0$ at the free surface. Then we project the intermediate velocity field to ensure the incompressibility condition.

$$\nabla^2 p = \frac{\rho}{\Delta t} \nabla \cdot \mathbf{u}^* \tag{2}$$

$$\mathbf{u}^{n+1} = \mathbf{u}^* - \frac{\Delta t}{\rho} \nabla p \tag{3}$$

In this work, we modify the second step ((2) and (3)) to take into account the interaction between the fluid and the articulated rigid bodies.

### 3.2 Articulated Rigid Body Simulation

In this section, we will describe the numerical techniques that we use to move the body parts of an articulated figure. Later, in Section 4, we will describe the optimization technique that we use to discover efficient swimming gaits.

The dynamic equations of an articulated rigid body in generalized coordinates can be expressed as follows.

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}) = \tau_{int} + \tau_{ext} \tag{4}$$

where $\mathbf{q}$, $\dot{\mathbf{q}}$ and $\ddot{\mathbf{q}}$ are vectors of positions, velocities and accelerations of joint degrees of freedom respectively. $\mathbf{M}(\mathbf{q})$ is the mass matrix and $\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})$ accounts for the Coriolis and Centrifugal force. $\tau_{int}$ and $\tau_{ext}$ are internal and external generalized forces.

Given the current state $\mathbf{q}^n$ and $\dot{\mathbf{q}}^n$, we can evaluate $\mathbf{M}$ and $\mathbf{C}$ of (4). For the external forces $\tau_{ext}$, we consider the fluid pressure force. We make use of the modified PD controller of Tan et al. [2011] in order to calculate the internal force $\tau_{int}$ that closely tracks a reference trajectory. Although the details of this method can be found in [Tan et al. 2011], we include an overview of this method below. The reference swimming trajectory is computed by an optimization process described in Section 4. Once we know both the external and internal forces, we can solve the acceleration $\ddot{\mathbf{q}}^n$ and advance to the next time step via explicit Euler integration.

**Modified Proportional-Derivative Controller** In computer animation, a PD servo (5) provides a simple framework to compute control forces for tracking a kinematic state of a joint trajectory:

$$\tau^n = -k_p(q^n - \bar{q}^n) - k_d \dot{q}^n \tag{5}$$

where $k_p$ and $k_d$ are the gain and damping coefficient. In general, high gain PD servos result in small simulation time steps in order to maintain stability.

The aquatic creatures in this work require high gain PD servos to track the desired swimming gait closely against strong fluid pressure. However, we cannot reduce the time step to accommodate stability due to the time-consuming fluid simulation. To achieve these two conflicting goals, large time steps and high gains, we modify the PD controller as follows. Instead of using the current state $q^n$ and $\dot{q}^n$ to compute the control force, we compute the control forces using the state at next time step $q^{n+1}$ and $\dot{q}^{n+1}$:

$$\tau^n = -k_p(q^{n+1} - \bar{q}^{n+1}) - k_d \dot{q}^{n+1} \tag{6}$$

Equation (6) can be linearized at $q^n$ and $\dot{q}^n$ as:

$$\tau^n = -k_p(q^n + \Delta t \dot{q}^n - \bar{q}^{n+1}) - k_d(\dot{q}^n + \Delta t \ddot{q}^n)$$

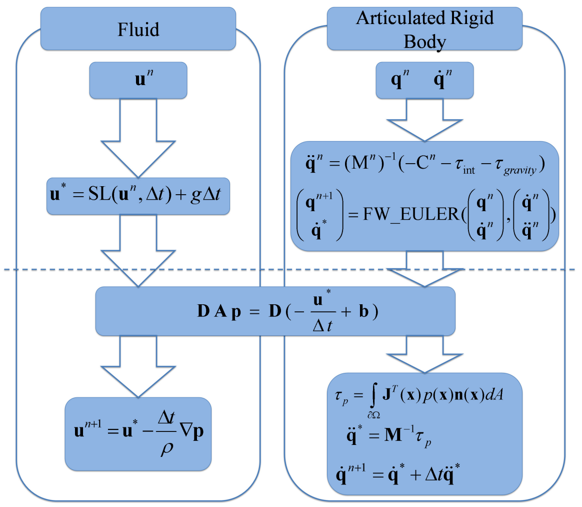
Fluid
Articulated Rigid Body
$\mathbf{u}^n$
$\mathbf{q}^n \quad \dot{\mathbf{q}}^n$
$\mathbf{u}^* = \mathrm{SL}(\mathbf{u}^n, \Delta t) + g\Delta t$
$\ddot{\mathbf{q}}^n = (\mathrm{M}^n)^{-1}(-\mathrm{C}^n - \tau_{\mathrm{int}} - \tau_{gravity})$
$\begin{pmatrix}\mathbf{q}^{n+1}\\ \dot{\mathbf{q}}^*\end{pmatrix} = \mathrm{FW\_EULER}(\begin{pmatrix}\mathbf{q}^n\\ \dot{\mathbf{q}}^n\end{pmatrix}, \begin{pmatrix}\dot{\mathbf{q}}^n\\ \ddot{\mathbf{q}}^n\end{pmatrix})$
$\mathbf{D}\,\mathbf{A}\,\mathbf{p} = \mathbf{D}(-\frac{\mathbf{u}^*}{\Delta t} + \mathbf{b})$
$\mathbf{u}^{n+1} = \mathbf{u}^* - \frac{\Delta t}{\rho}\nabla\mathbf{p}$
$\tau_p = \int_{\partial\Omega} \mathbf{J}^T(\mathbf{x})p(\mathbf{x})\mathbf{n}(\mathbf{x})dA$
$\ddot{\mathbf{q}}^* = \mathbf{M}^{-1}\tau_p$
$\dot{\mathbf{q}}^{n+1} = \dot{\mathbf{q}}^* + \Delta t\ddot{\mathbf{q}}^*$

**Figure 2:** *The computational steps for simultaneous coupling between fluids and articulated rigid bodies.*

Applying the modified PD controller to the articulated rigid body simulation with multiple degrees of freedom, we solve the acceleration as

$$\ddot{\mathbf{q}}^n = (\mathbf{M}+\mathbf{K}_d\Delta t)^{-1}(-\mathbf{C}-\mathbf{K}_p(\mathbf{q}^n+\dot{\mathbf{q}}^n\Delta t-\bar{\mathbf{q}}^{n+1})-\mathbf{K}_d\dot{\mathbf{q}}^n+\tau_{ext})$$

where both $\mathbf{K}_p$ and $\mathbf{K}_d$ are diagonal matrices that indicate the gains and damping coefficients.

## 3.3 Two-way Coupling Between Fluids and Articulated Rigid Bodies

The two-way coupling between the incompressible fluid and the articulated figures should satisfy following three conditions.

1. The normal velocity at the interface between the fluids and the articulated rigid bodies should agree with each other.
2. The motion of the articulated rigid body resulting from the fluid pressure force must be consistent with the Lagrangian equations of motion.
3. The fluid should be incompressible.

Two-way coupling is ensured by having the fluid exerting pressure forces on the rigid bodies, while at the same time the motion of the rigid bodies affects the pressure distribution of the fluid.

Our simultaneous two-way coupling technique is inspired by Klingner et al. [2006] since we both start from the acceleration at cell faces. Their method uses a tetrahedral mesh to represent the fluid, and their rigid bodies are in Cartesian space. Our simulator uses a regular MAC grid and we couple this fluid with articulated figures that are described in generalized coordinates. Similar to Klingner et al. [2006], we split the coupling into two steps (Figure 2). In the first step, the two systems are solved independently ignoring the pressure. The fluid solver calculates the intermediate velocity field $\mathbf{u}^*$ using (1). The articulated rigid body solver determines the acceleration $\ddot{\mathbf{q}}$ without external pressure forces and calculates the intermediate velocity $\dot{\mathbf{q}}^*$.

In the second step, we consider the motion of the two systems together so that they will satisfy all above three conditions. We first voxelize the body segments of the articulated figure (represented by water-tight polygon meshes) onto the MAC grid and we mark those cells inside the body segments as SOLID. For two-way coupling, we are particularly interested in the faces between a SOLID cell and a FLUID cell (defined as *coupled faces*). The velocity at a coupled face can be expressed in generalized coordinates by the Jacobian of the articulated rigid body and the joint velocity:

$$\mathbf{u}^*_{solid} = \mathbf{J}\dot{\mathbf{q}}^*$$

where $\mathbf{J}$ is the $3 \times m$ ($m$ is the number of degrees of freedom) Jacobian matrix

$$\mathbf{J} = \begin{pmatrix} \frac{\partial x}{\partial q_1} & \frac{\partial x}{\partial q_2} & \cdots & \frac{\partial x}{\partial q_m} \\ \frac{\partial y}{\partial q_1} & \frac{\partial y}{\partial q_2} & \cdots & \frac{\partial y}{\partial q_m} \\ \frac{\partial z}{\partial q_1} & \frac{\partial z}{\partial q_2} & \cdots & \frac{\partial z}{\partial q_m} \end{pmatrix}$$

Now consider the effect of the pressure field, which exerts forces and applies accelerations along the face normals $\mathbf{n}$. If a face is shared by two FLUID cells, the acceleration is $\frac{1}{\rho}\nabla\mathbf{p}\cdot\mathbf{n}$. If a face is shared by a FLUID cell and a SOLID cell (a coupled face), we need to take into account all the pressure values surrounding the articulated rigid body. We first construct a $k \times n$ selection matrix $\mathbf{S}$ to pick out of $\mathbf{p}$ the pressures at the coupled faces, where $k$ is the number of the coupled faces and $n$ is the number of FLUID cells. Thus the vector $\mathbf{Sp}$ constains all the pressure values surrounding the articulated rigid body. Each element $p_i$ of $\mathbf{Sp}$ contributes a pressure force $(\Delta x)^2 p_i \mathbf{n}_i$ to the articulated rigid body, which we transform to the generalized coordinate:

$$\tau_{p_i} = \mathbf{J}_i^T(\Delta x)^2 p_i \mathbf{n}_i$$

The total generalized force exerted by the fluid pressure on the articulated rigid body is

$$\tau_p = (\Delta x)^2\hat{\mathbf{J}}\mathbf{Sp}$$

where $\hat{\mathbf{J}} = [\mathbf{J}_1^T\mathbf{n}_1 \quad \ldots \quad \mathbf{J}_k^T\mathbf{n}_k]$. The pressure force results in the acceleration in generalized coordinates

$$\ddot{\mathbf{q}}_{\mathbf{p}} = \mathbf{M}^{-1}\tau_p$$

We transform the acceleration back to Cartesian space, and the magnitude of the acceleration at the coupled face is

$$a = \mathbf{n}^T(\mathbf{J}\ddot{\mathbf{q}}_{\mathbf{p}} + \dot{\mathbf{J}}\dot{\mathbf{q}}^*)$$

The second term $\dot{\mathbf{J}}\dot{\mathbf{q}}^*$ comes from the fact that the Jacobian matrix changes over time. Stacking the accelerations at the coupled faces into a vector, we have

$$\mathbf{a} = (\Delta x)^2\hat{\mathbf{J}}^T\mathbf{M}^{-1}\hat{\mathbf{J}}\mathbf{Sp} + \dot{\hat{\mathbf{J}}}^T\dot{\mathbf{q}}^* \tag{7}$$

where $\dot{\hat{\mathbf{J}}} = [\dot{\mathbf{J}}_1^T\mathbf{n}_1 \quad \ldots \quad \dot{\mathbf{J}}_k^T\mathbf{n}_k]$.

Since the velocity field should be divergence free at the beginning of the next time step,

$$\nabla\cdot\mathbf{u}^{n+1} = \nabla\cdot(\mathbf{u}^* + \Delta t\mathbf{a}) = 0 \tag{8}$$

the accelerations due to the pressure must satisfy the following equation.

$$\nabla\cdot\mathbf{a} = -\frac{1}{\Delta t}\nabla\cdot\mathbf{u}^*$$

Putting everything together, we reach the final linear system:

$$\mathbf{DAp} = \mathbf{D}(-\frac{\mathbf{u}^*}{\Delta t} + \mathbf{b}) \tag{9}$$

$$\mathbf{A} = \begin{cases} \frac{1}{\rho}\mathbf{G} & \text{faces shared by two FLUID cells} \\ (\Delta x)^2 \hat{\mathbf{J}}^T \mathbf{M}^{-1} \hat{\mathbf{J}} \mathbf{S} & \text{coupled faces} \end{cases}$$
$$\mathbf{b} = \begin{cases} \mathbf{0} & \text{faces shared by two FLUID cells} \\ -\dot{\hat{\mathbf{J}}}^T \dot{\mathbf{q}}^* & \text{coupled faces} \end{cases}$$

where $\mathbf{D}$ and $\mathbf{G}$ are the discretization of the divergence and gradient operators on a MAC grid.

We construct a system of linear equations (9) for the pressure field, which considers all of the three conditions to be satisfied by the coupled system. The fluid and solid velocity agrees at the interface (condition 1) because the velocity defined at the coupled faces are shared by the fluid and the articulated body. The movement of the articulated rigid body under the fluid pressure satisfies the equation of motion (condition 2) because (7) is derived from the dynamics (4). The fluid is incompressible (condition 3) because we enforce the divergence free condition by (8). The linear system is of the same size as the discretized Poisson equation (2) in a typical fluid simulation. The main difference is that the rows correpsonding to the cells adjacent to the SOLID cells have more non-zero entries. Furthermore, it is also symmetric positive definite, which allows the use of fast solvers such as the Preconditioned Conjugate Gradient method. After solving the pressure field, we project the velocity field to make it divergence free using (3) and update the articulated rigid body by considering the pressure forces.

## 4 Optimizaton of Swimming Gaits

Section 3 describes the two-way interaction between fluids and an articulated rigid body system. In particular, Section 3.2 describes how we move the body parts using torques and how we compute the torques for a given reference gait. In this section, we describe an algorithm to automatically design optimal controllers for an active articulated rigid body systems that is moving in a hydrodynamic environment. Our method generates physically realistic strokes based on the swimming efficiency of the stroke.

### 4.1 Swimming Gait Representation

Given the geometric and physical properties of an articulated rigid body system, we formulate an optimization to solve for the reference trajectory of PD controller at each actuated joint, $q_i$. We want to use a compact representation for the reference trajectory because incorporating a fluid simulation into the optimization is computational intensive. Because aquatic locomotion is typically cyclic, we parameterize the reference trajectory as periodic cycles in generalized coordinates.

$$q_i(t) = A_i \sin(\frac{2\pi t}{T_i} + \phi_i) + C_i$$

where $A_i, T_i, \phi_i$ and $C_i$ are the amplitude, period, phase and offset of a sine function. Using this parameterization, each reference trajectory $q_i(t)$ is parameterized by four values. In most cases we just optimize over two parameters, amplitude and phase, and leave the period and offset fixed.

### 4.2 Objective Function

The objective function in our optimization tries to balance between efficiency and energy expenditure of the swimming gait; the creature should move as fast as possible in the desired direction without using too much energy. Furthermore, the creature should try to avoid self-collisions and remain within the joint limits. In practice, the choice of objective function can vary by creatures, fluid conditions, or the user's application. Here we choose a simple objective function to find natural swimming motion:

$$E = -E_{distance} + w_1 E_{deviation} + w_2 E_{energy} + w_3 E_{collision} \quad (10)$$

where $E_{distance}$ measures the change of the creature's root position $\Delta\mathbf{p}$ along a specified direction $\mathbf{d}$ from time 0 to time $t_f$:

$$E_{distance} = \mathbf{d}^T(\Delta\mathbf{p})$$

$E_{deviation}$ measures the deviation from the specified direction and the initial orientation.

$$E_{deviation} = ||\Delta\mathbf{p} - \mathbf{d}^T(\Delta\mathbf{p})\mathbf{d}|| + ||\Delta\alpha||$$

where $\Delta\alpha$ stands for the change of root orientation in $t_f$, expressed using the exponential map. Since we're optimizing the gait of straight swimming, we penalize any orientation changes. We choose the weight $w_1 = 0.2$ for all the examples.

$E_{energy}$ penalizes the energy expenditure of the swimming gait. We calculate the work done by the actuated joints over the duration of the swimming gait:

$$E_{energy} = \int_0^{t_f} \sum_i \tau_i \dot{q}_i dt$$

Instead of penalizing energy expenditure linearly, we modulate $E_{energy}$ with a discontinuous function represented as the objective weight $w_2$. Instead of constantly trying to avoid using any energy, this modulation allows the creature to freely consume a certain amount of energy, while avoiding excessive use of torques.

$$w_2 = \begin{cases} 0 & \text{if } E_{energy} < E_{energyBound} \\ 1 & \text{otherwise} \end{cases}$$

where $E_{energyBound}$ is a user specified parameter.

$E_{collision}$ penalizes self-intersection. We detect self-intersection and calculate the overlapping volumes using a fast approximate method. We first voxelize the articulated rigid body using a fine grid (the typical grid resolution is about $100^3$). If a cell is inside a body link, we increment the counter for that cell. At each time step, we sum up all the cells with counter number larger than one and multiply by the cell volume to approximate the overlapping volume.

$$E_{collision} = \int_0^{t_f} V_{overlap} dt$$

where $w_3$ is chosen to be 500 for all the examples.

### 4.3 Optimization

Our objective function is discontinuous and prone to local minima due to sub-optimal swimming gaits, collision penalties, and the modulation of the energy penalty term. We perform gait optimization using Covariance Matrix Adaptation (CMA). CMA is based on evaluating the objective function for a given population of samples over the parameter space (in our case the joint trajectories). Some fraction of the best samples are then used to update the mean and a covariance matrix that determines the distribution of samples that are evaluated in the next generation. More details of the CMA method can be found in [Hansen and Kern 2004].

For each CMA sample, if it violates the user specified joint limits we simply discard it and select another sample. Because the joint limit test takes very little computation time, discarding infeasible samples at this stage is more "economical" than investing major computation effort on them but assigning them a near-zero weight at the end. Once a sample is accepted, we simulate the motion by applying the sampled swimming gait and evaluate the resulting motion using the objective function. To speed up CMA for solving such high-dimensional problems, we include two heuristics in our implementation for some examples. First, we utilize symmetry for some of our articulated figures: When a creature's body shape is symmetric, often its gait is also symmetric. In such cases, half of the optimization variables are enough to characterize the gait of the whole body because we mirror them to the other half of the body. This assumption is applied to reduce the required computational time, but it is not necessary. Second, for creatures that have more independent appendages, we separate the degrees of freedom in groups and progressively improve the solution by optimizing each group. For example, we assign the forelimbs and hindlimbs of a frog into two separate groups. During the optimization, we first search for the swimming gait for the hindlimbs while freezing the motions of the forelimbs. We then search for the swimming gait of the forelimbs with the optimal hindlimb motions that we already found.

## 5 Path Following

In addition to forward thrust, aquatic creatures also employ very efficient turning maneuvers, such as pitching up and down or turning left and right. The optimization technique described in Section 4 can be modified to learn various maneuvers. Once the aquatic creature builds a repertoire of swimming maneuvers, we can combine different maneuvers to achieve a high-level task such as path following.

First, we add another term to $E_{distance}$ in (10) to maximize the turning angle towards the desired direction:

$$E_{distance} = \mathbf{d}^T(\Delta\mathbf{p}) + \mathbf{r}^T(\Delta\alpha)$$

where $\mathbf{r}$ is the desired axis of rotation. We set the desired swimming direction $\mathbf{d}$ half way from the current facing direction towards the turning direction. We also change $E_{deviation}$ to penalize the undesired orientation changes.

$$E_{deviation} = ||\Delta\mathbf{p} - \mathbf{d}^T(\Delta\mathbf{p})\mathbf{d}|| + ||\Delta\alpha - \mathbf{r}^T(\Delta\alpha)\mathbf{r}||$$

We solve the above optimization using the CMA method in the same way as described in Section 4.3.

Once different maneuvers have been learned, we apply a simple heuristic to decide which maneuver to choose to follow the path. At the beginning of each cycle of the motion, we find the nearest point $\mathbf{p}$ on the path to the root of the articulated figure and transform $\mathbf{p}$ and its tangential direction $\mathbf{d}$ to the root coordinate system. We denote the transformed position and direction as $\hat{\mathbf{p}}$ and $\hat{\mathbf{d}}$. Without loss of generality, let's consider a one dimensional example. $\hat{p}_z$ is the z-component of $\hat{\mathbf{p}}$, which means the point is above or beneath the root of the articulated figure. Similarly, $\hat{d}_z$ indicates the path is going upwards or downwards relative to the root orientation. We choose the different maneuvers based on the following rules.

$$\text{Maneuver} = \begin{cases} \text{Go Straight} & \text{if } (\hat{p}_z \geq \epsilon \text{ and } \hat{d}_z \leq -\epsilon) \\ & \text{or } (\hat{p}_z \leq -\epsilon \text{ and } \hat{d}_z \geq \epsilon) \\ \text{Pitch Up} & \text{if } \hat{p}_z > \epsilon \text{ and } \hat{d}_z > \epsilon \\ \text{Pitch Down} & \text{if } \hat{p}_z < -\epsilon \text{ and } \hat{d}_z < -\epsilon \end{cases}$$

where $\epsilon$ is a small positive value to prevent the articulated figure from repeatedly choosing alternating turning maneuvers due to small deviations from the path. The first case indicates that the nearest point on the path is above/below the articulated figure while the direction of the path is going downwards/upwards. In other words, the articulated figure is swimming towards the path (Figure 3a). We choose the action "swim straight" in this situation. On the other hand, the second and third cases indicate the articulated figure is swimming away from the path (Figure 3b and 3c) and we choose "pitch up" or "pitch down" accordingly.

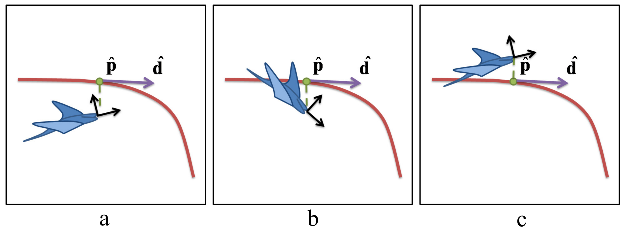

**Figure 3:** *Three different situations that determine if the creature chooses a "swim straight", "pitch up" or "pitch down" maneuver.*

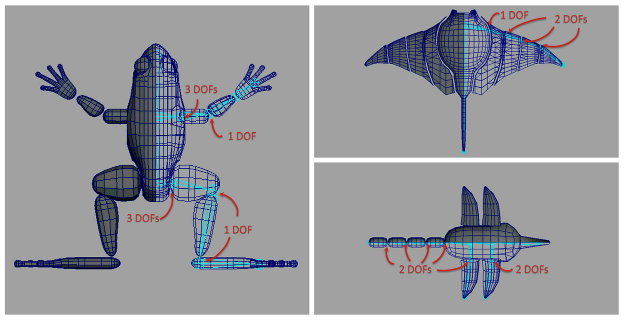

**Figure 4:** *The joint configurations of the frog, the manta ray and the alien.*

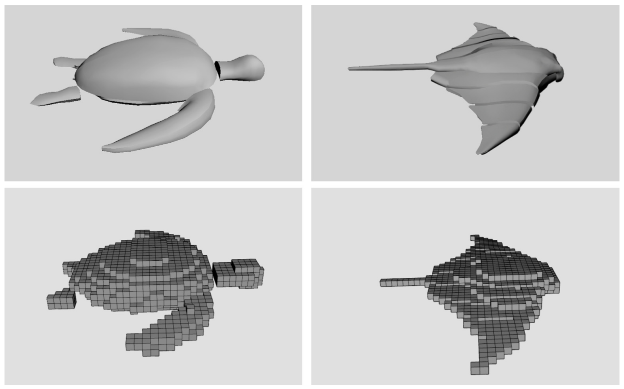

**Figure 5:** *The voxelized representations of the turtle and the manta ray. The input shapes of the articulated creatures are represented by water-tight polygon meshes. We voxelize these body shapes onto the simulation grid each time step to simulate the two-way coupling between the fluid and the creature.*

## 6 Implementation Details

We implemented our method using C++ and ran CMA on a cluster with a maximum of 100 iterations and with a population size of 16 for 2D and 31 for 3D examples. Each CMA sample evaluates the objective function by simulating two cycles of swimming motions.

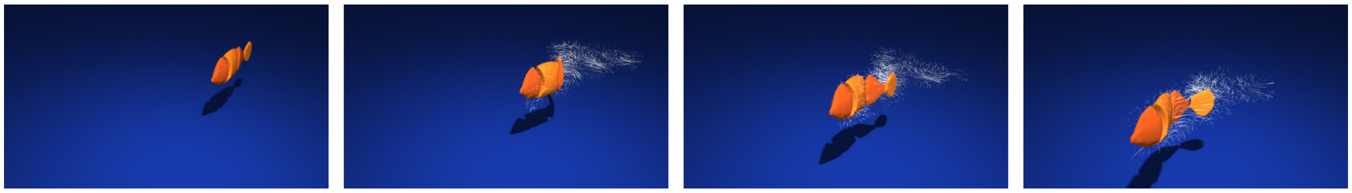

**Figure 6:** *A four-link clown fish swims. Carangiform swimmers like this flex the front of their body a little, with the majority of the motion near the tail. Note that this fish sheds two separate trails of vortices.*

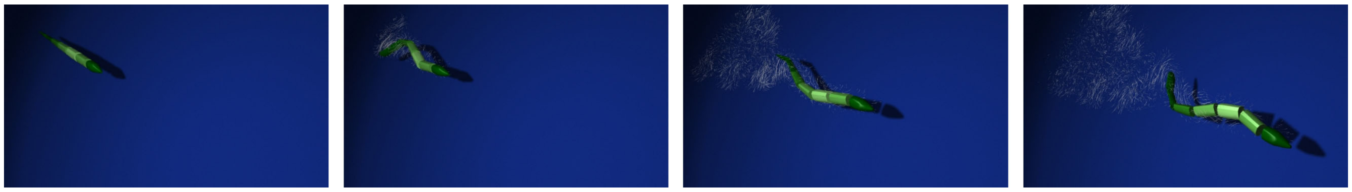

**Figure 7:** *A swimming seven-link eel. Anguiliform swimmers undulate their whole body as if a wave is travelling from head to tail, and shed two separate trails of vortices from the tail.*

The optimization took from several hours to two days, depending on the model and the grid resolution. After we found the swimming gait, we ran the swimming simulation on a 2.26GHz CPU with a single core. All of the data for our swimming examples are summarized in Table 1. In most of the cases, we use two optimization variables, amplitude and phase, for each degree of freedom. We set the period to one second and the offset to zero. When training the turning gaits, we included the offset in the optimization variables. For the accordian example, the degrees of freedom are interdependent and there is no phase shift among the different degrees of freedom. Thus one optimization variable is enough to characterize its motion. We also exploited the strong symmetry in geometry for some creatures, such as turtles and frogs, to halve the optimization dimensions. We illustrate the joint configurations for some creatures in Figure 4 and the voxelized representations of creatures in Figure 5.

In our implementation, we made three simplifications to reduce the simulation cost. 1) Instead of using a large computational domain to cover the whole space that the creature might swim to, we use a smaller domain that is about two to four times larger in each dimension than the creature's bounding box. This domain moves with the creature when the creature approaches a boundary. 2) At the boundary of the computional domain, we impose the Dirichlet boundary condition $p = 0$ so that the fluid outside the domain is free to flow in and vice versa. 3) Since the density of most aquatic creatures is similar to that of the fluid, we ignore the force of gravity in our simulator.

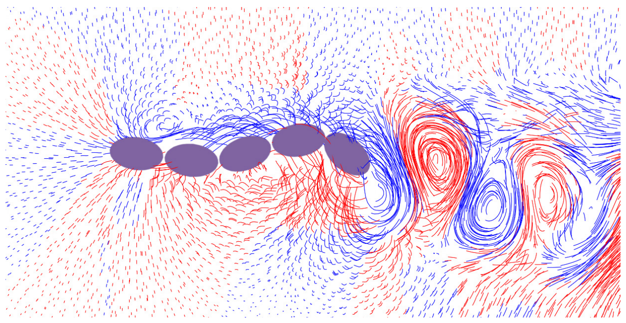

**Figure 8:** *A five-link eel swims in a 2D fluid environment. In contrast to the simulation in 3D, an eel swimming in 2D fluid sheds only one single vortex street. Red traces show the counter-clockwise vortices while blue traces show the clockwise vortices.*

| Examples | Num DOFs | Opt Dims | Sim Res | Sim Time |
|---|---|---|---|---|
| accordian | 10 | 1 | $120 \times 80$ | 1.37s |
| eel(2D) | 4 | 8 | $128 \times 64$ | 0.64s |
| turtle(2D) | 4 | 4 | $64 \times 64$ | 0.34s |
| fish | 3 | 6 | $64 \times 32 \times 32$ | 1.45s |
| eel(3D) | 6 | 12 | $64 \times 32 \times 32$ | 1.31s |
| manta-ray | 14 | 21 | $64 \times 32 \times 32$ | 10.92s |
| turtle(3D) | 10 | 10 | $64 \times 32 \times 32$ | 11.29s |
| frog | 18 | 18 | $96 \times 64 \times 48$ | 12.79s |
| alien | 16 | 24 | $96 \times 36 \times 24$ | 10.75s |

**Table 1:** *Parameters and performance of examples. Num DOFs is the number of degrees of freedom for the articulated rigid body. Opt Dims is the number of optimization variables. Sim Res is the grid resolution for the simulation and Sim Time is the average simulation time per frame.*

## 7 Results

In this section we describe the results of our swimming optimization method. Please see the accompanying video to observe the swimming animations. To visualize the fluid flow, we draw *particles traces*, which show the trajectory of massless particles inside the fluid in a short period of time (15 frames). We modulate the transparency of the particle traces in 3D examples according to the magnitude of the vorticity in order to focus attention on the visually interesting regions of the flow. In 2D examples, we colored the traces to indicate the directions of the vortices.

There are many body shapes and styles of locomotion for fish, and our first set of results investigates several of these. Figure 6 shows a four-segment model of a fish, modelled after the body shape of the clownfish. We used CMA to optimize for efficient forward motion, and snapshots of the resulting motion are given in the figure. Note that the forward body flexes just a little, with the majority of the motion near the tail, which is in good agreement for the style of motion known as the carangiform mode [Lindsey 1978]. Using

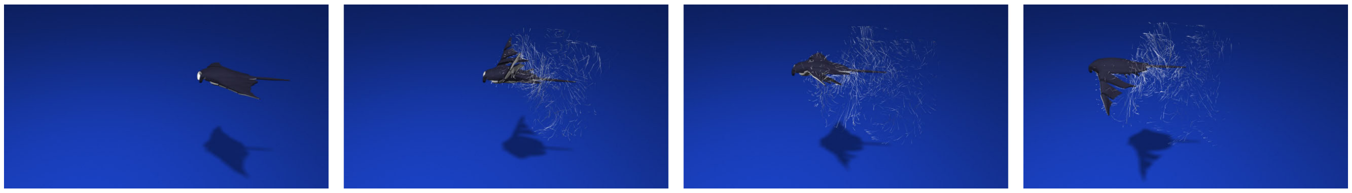

**Figure 9:** *A manta ray swimming forward. Rajiform swimmers swim by slow flapping strokes like a slow-motion version of a bird flapping its wings.*

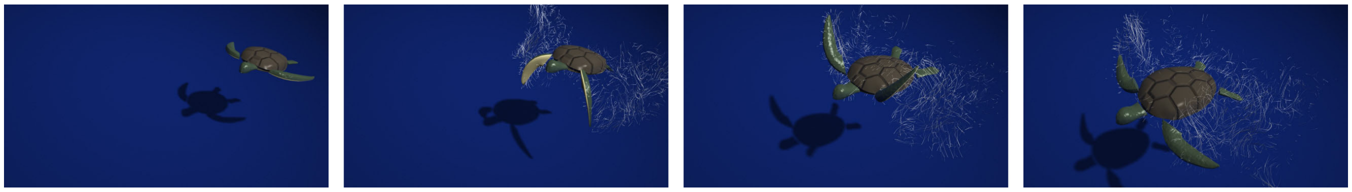

**Figure 11:** *A turtle swims in water with a flapping motion of its two front flippers.*

the same objective function, we optimized a seven-segment figure that was designed to mimic an eel body. Figure 7 shows that the resulting motion is that of a travelling wave along the body of the creature, as is typical of real eel swimming (anguiliform mode). Note that the wake of our eel has two separate trails of vortices that are shed from the tip of the tail, as has been observed in lab studies of eels [Tytell and Lauder 2004]. We show in Figure 8 that a different wake structure appears when an eel swims in a 2D fluid environment, that of a single vortex street. The difference of the wake structure between the 2D and 3D simulations agrees with Kern and Koumoutsakos's study of eels [Kern and Koumoutsakos 2006].

Our final example of fish motion is that of a manta ray. The manta has a body that is thin in the vertical direction and that has large pectoral fins that extend in the horizontal direction. It swims by slow flapping strokes of these wing-like pectoral fins (the rajiform swimming mode), somewhat like a slow-motion version of a bird flapping its wings. Although the manta ray does not seem to be a good candidate to be modelled as an articulated figure, we wanted to see how far the articulated models could be pushed. We modelled the ray's pectoral fins as four rows of thin plates that are connected

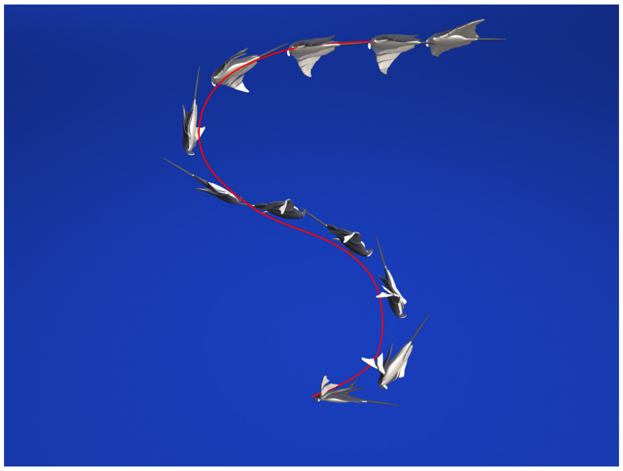

**Figure 10:** *A manta ray follows an S-shaped path by choosing maneuvers from "swimming straight", "pitch up" and "pitch down". The red curve is the path specified by the user.*

to one another near the leading edge of the fin. The resulting swimming motion from the optimization procedure exceeded our expectations, producing the same graceful flapping that these creatures use to swim (see Figure 9).

We tested our path following approach using the manta ray model. We used our optimization method to find efficient swimming for forward motion, an upward turn and a downward turn. We then gave the manta ray a vertical S-shaped path to follow using our path following controller. The simulated ray was able to follow the path quite closely, as the composite image in Figure 10 shows. Note that this path following motion was created with a single simulation, based on gait switching between the three learned basic motions. We also tested the path following algorithm using a simple 2D turtle model. We show that the turtle cannot swim straight without using the path following technique due to the accumulation of numerical errors. When the path following technique is applied, the turtle actively adjusts its swimming motions according to its position and orientation and successfully swims straight.

Figure 11 shows the motion of a sea turtle that was created using optimization. Adult sea turtles are underwater fliers, moving themselves forward with a flapping motion of their two front flippers that is called a *powerstroke* [Wyneken 1997]. Note that our turtle results show the characteristic rotating of the front flippers during the upward stroke. Figure 12 shows the results of our swimmer optimization for a model frog. As with real frogs, the large rear legs provide the forward thrust using a classic frog kick. Note that the frog uses its forelimbs with a small range of motion. We think this is because the contribution from the arms is small relative to the contribution from the legs. Based on our observation, some real frogs do not use their forelimbs much when swimming.

In the accompanying video, we also demonstrates that articulated figures can differ dramatically in their swimming motion depending on whether the simulated fluid is a simple model or a full Navier-Stokes (NS) solver. Our simple fluid simulator calculates the force as the square of the normal component of the velocity of a moving surface element. This simplified fluid model is identical to that in [Wu and Popović 2003; Lentine et al. 2010]. We show that swimming in different fluid models leads to different locomotions. Figure 14 shows a 2D swimmer that compresses and relaxes its body in an accordian-like manner moves through the water in the NS fluid but stay in one place in the simple fluid. We demonstrate that the gaits trained in different fluid models differ considerably. The swimming gait for a fish trained in NS fluid smoothly flaps its

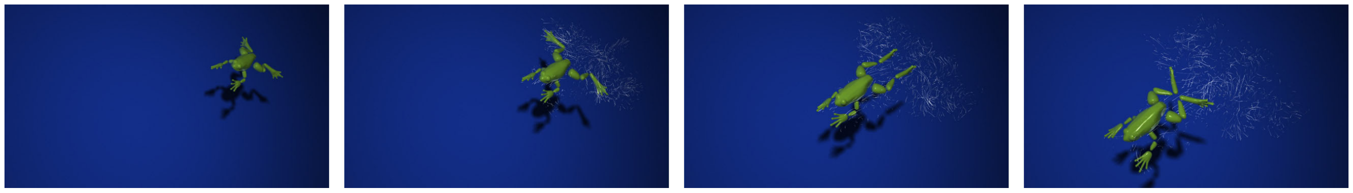

**Figure 12:** *A frog mainly relies on its large rear legs to provide forward thrust in the water.*

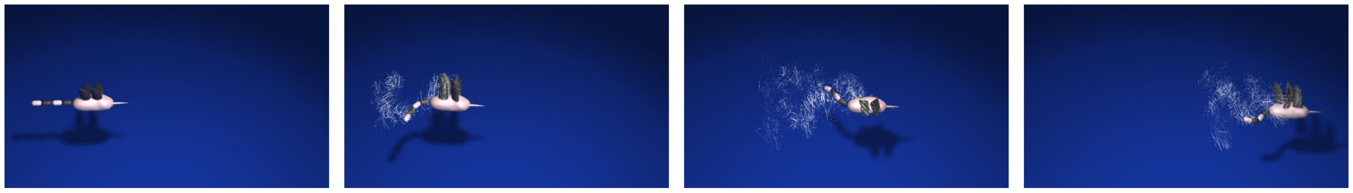

**Figure 13:** *An alien aquatic creature that swims in water by undulating its tails and flapping its wings. Note the two pairs of wings are slightly out of phase to mimic flapping motion of larger wings.*

tail to propel itself forward. When this same fish model is optimized using the simple fluid, the resulting motion is considerably different, gaining thrust mainly from bending at a sharp angle at the middle joint of the body. These differences in motion between a simple fluid and the NS fluid are in agreement with the findings of Lentine et al. [2010].

In order to test the generality of our method, we applied our swimming optimization to an articulated figure that has no counterpart in the real world (see Figure 13). This is the swimming version of the task of finding plausible walking motions for a user-created land creature [Hecker et al. 2008; Wampler and Popović 2009]. Our alien creature has two pairs of limbs on the trunk of its body, and in addition has a long and powerful tail. The motion that was found by our optimization combines a whip-like motion of the tail together with coordinated rowing from the pairs of limbs. Although there is no point of comparison in the real world for this creature's motion, the resulting swimming pattern looks entirely plausible.

Although our method requires little prior knowledge about the swimming gait of the creature, there are some parameters that users can change, including the energy bound, the period of the motion for each degree of freedom, and joint limits. This provides users the freedom to achieve different motion, agile or slow, by changing these parameters. In particular, we tuned the period of the swimming gait and the energy bound in our examples. We used a period of one second for all of the examples. We made this choice deliberately because choosing a longer period means longer optimization time (each CMA sample needs to simulate two cycles of swimming motions). However, we believe that including the period into the optimization will probably give more interesting results because different periods could make a big difference in the final swimming gait. We leave this as future work. To set the energy bounds, we began by trying out several energy bounds for an initial animal, the fish shown in the upper left of Figure 1. Once we were satisfied with the results, we then used this as our standard energy bound. For a new creature, we scaled this standard energy bound according to the mass of the new creature relative to the mass of the fish. Users can also change the weights in the objective function. In our examples, we set all these parameters by intuition without much tuning. The weights are reported at the end of each paragraph that introduces the different objectives in Section 4.2.

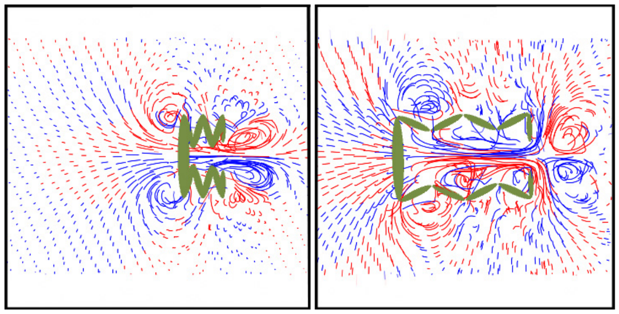

**Figure 14:** *An imaginary creature swims forward by compressing and relaxing its body in an accordion-like manner in a Navier-Stokes fluid model. The images are two snapshots in the animation sequence. This demonstrates that including the Navier-Stokes fluid model is necessary to capture certain swimming patterns, such as jet propulsion, because a simplified fluid model does not allow forward motion for such modes of locomotion.*

## 8 Limitations

Although we have successfully applied our method to various aquatic creatures with disparate body shapes and joint configurations, our approach does have limitations.

Our two-way coupling method needs to voxelize the articulated rigid body, and the accuracy for representing the articulated figure depends on the grid resolution. Thin features cannot be captured by the fluid simulator (Figure 5). We believe that incorporating adaptive grids [Losasso et al. 2004] or unstructured meshes [Brochu et al. 2010; Klingner et al. 2006] can dramatically increase the accuracy of the two-way coupling. Furthermore, the two-way coupling method is tailored for the interaction between fluids and articulated figures. Even though many aquatic creatures have a skeleton and can be represented well by articulated figures, there are exceptions such as jellyfish. Our framework for discovering the optimal swimming gaits and path following is still valid for soft-body creatures, but we would need an efficient two-way coupling mechanism to simulate these swimming motions. We leave this as future work.

We use the sine function to parameterize the joint space. There are quite a few motions that cannot be depicted by a single sine function, such as gliding. One possible way to improve this is to use a weighted sum of multiple sine functions with different amplitudes, phases and periods [Grzeszczuk and Terzopoulos 1995]. However, this would require more optimization variables and more computational resources to discover a swimming gait.

Our simulated swimmers seem to use more energy than the real creatures do because the simulated water is more viscous than real water. Even though we use the inviscous Navier-Stokes equation (Euler equation) to simulate the fluid, there is numerical viscosity. We chose to use relatively coarse grid, and thus incur large numerical viscosity, to keep the computational time tractable because CMA optimization needs to simulate the two-way coupling thousands of times. In addition, while the real aquatic creatures take advantage of their streamline shaped body to reduce the fluid drag, the simulated creatures are voxelized and the resultant stair-step shaped body is not particularly efficient inside the fluid.

## 9 Conclusion

We have demonstrated that our approach creates natural swimming behavior for a wide variety of animal bodies. For short-bodied fish and eels, our results show vortex trails that are in agreement with laboratory measurements and other published simulation results. For the other creatures, our optimized motions have the same overall appearance of the real-world animals, although lab data is not available. Our articulated body representation of creature anatomy is quite general, even allowing us to animate forms such as the manta ray that are not usually thought of as articulated figures.

There are a number of interesting avenues for future work. There are many ways this approach could be expanded to give more control to animators, including different path following strategies and higher-level behavior control. Our work has concentrated on continuous motion, but many animals have distinctly different movements for situations such as escaping a predator. It would be interesting to investigate these faster, intermittent motions. Swimming at the *surface* of the water could be studied, including motions such as a human swimmer doing the crawl or a whale jumping out of the water (breaching). Finally, taking a cue from the work of Karl Sims, it would be fascinating to simultaneously optimize for both swimming motion and body shape.

## Acknowledgements

We thank the anonymous reviewers for their helpful comments. We thank the members of the Aquatic Propulsion Lab for their help on this work. We also thank Yuting Ye for her help on articulated rigid body simulation and Ron Hutchins, Neil Bright and the Georgia Tech Office of Information Technology for providing us with computing cluster resources. We are grateful for an equipment donation from NVIDIA. This work was funded by NSF grants CCF-0811485 and IIS-1017014.

## References

ARASH, O. E., GENEVAUX, O., HABIBI, A., AND MICHEL DISCHLER, J. 2003. Simulating fluid-solid interaction. In *in Graphics Interface*, 31–38.

BARRETT, D., GROSENBAUGH, M., AND TRIANTAFYLLOU, M. 2002. The optimal control of a flexible hull robotic undersea vehicle propelled by an oscillating foil. In *Autonomous Underwater Vehicle Technology, 1996. AUV'96., Proceedings of the 1996 Symposium on*, IEEE, 1–9.

BATTY, C., BERTAILS, F., AND BRIDSON, R. 2007. A fast variational framework for accurate solid-fluid coupling. In *ACM SIGGRAPH 2007 papers*, ACM, New York, NY, USA, SIGGRAPH '07.

BROCHU, T., BATTY, C., AND BRIDSON, R. 2010. Matching fluid simulation elements to surface geometry and topology. In *ACM SIGGRAPH 2010 papers*, ACM, New York, NY, USA, SIGGRAPH '10, 47:1–47:9.

CARLSON, M., MUCHA, P. J., AND TURK, G. 2004. Rigid fluid: animating the interplay between rigid bodies and fluid. In *ACM SIGGRAPH 2004 Papers*, ACM, New York, NY, USA, SIGGRAPH '04, 377–384.

CHENTANEZ, N., GOKTEKIN, T. G., FELDMAN, B. E., AND O'BRIEN, J. F. 2006. Simultaneous coupling of fluids and deformable bodies. In *Proceedings of the 2006 ACM SIGGRAPH/Eurographics symposium on Computer animation*, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, SCA '06, 83–89.

DA SILVA, M., ABE, Y., AND POPOVIĆ, J. 2008. Interactive simulation of stylized human locomotion. In *ACM SIGGRAPH 2008 papers*, ACM, New York, NY, USA, SIGGRAPH '08, 82:1–82:10.

FALOUTSOS, P., VAN DE PANNE, M., AND TERZOPOULOS, D. 2001. Composable controllers for physics-based character animation. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, ACM, New York, NY, USA, SIGGRAPH '01, 251–260.

GRANT, I. 1997. Particle image velocimetry: a review. *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science 211*, 1, 55–76.

GRZESZCZUK, R., AND TERZOPOULOS, D. 1995. Automated learning of muscle-actuated locomotion through control abstraction. In *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, ACM, New York, NY, USA, SIGGRAPH '95, 63–70.

GUENDELMAN, E., SELLE, A., LOSASSO, F., AND FEDKIW, R. 2005. Coupling water and smoke to thin deformable and rigid shells. In *ACM SIGGRAPH 2005 Papers*, ACM, New York, NY, USA, SIGGRAPH '05, 973–981.

HANSEN, N., AND KERN, S. 2004. Evaluating the CMA evolution strategy on multimodal test functions. In *Parallel Problem Solving from Nature-PPSN VIII*, Springer, 282–291.

HECKER, C., RAABE, B., ENSLOW, R. W., DEWEESE, J., MAYNARD, J., AND VAN PROOIJEN, K. 2008. Real-time motion retargeting to highly varied user-created morphologies. In *ACM SIGGRAPH 2008 papers*, ACM, New York, NY, USA, SIGGRAPH '08, 27:1–27:11.

HODGINS, J. K., WOOTEN, W. L., BROGAN, D. C., AND O'BRIEN, J. F. 1995. Animating human athletics. In *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, ACM, New York, NY, USA, SIGGRAPH '95, 71–78.

KERN, S., AND KOUMOUTSAKOS, P. 2006. Simulations of optimized anguilliform swimming. *Journal of Experimental Biology 209*, 24, 4841.

KIM, B., LIU, Y., LLAMAS, I., AND ROSSIGNAC, J. 2007. Advections with significantly reduced dissipation and diffusion. *IEEE Transactions on Visualization and Computer Graphics 13*, 135–144.

KLINGNER, B. M., FELDMAN, B. E., CHENTANEZ, N., AND O'BRIEN, J. F. 2006. Fluid animation with dynamic meshes. In *ACM SIGGRAPH 2006 Papers*, ACM, New York, NY, USA, SIGGRAPH '06, 820–825.

KRY, P. G., REVERET, L., FAURE, F., AND M.-P.CANI. 2009. Modal locomotion: Animating virtual characters with natural vibrations. *Computer Graphics Forum (Eurographics) 28*, 2.

KWATRA, N., WOJTAN, C., CARLSON, M., ESSA, I., MUCHA, P., AND TURK, G. 2009. Fluid simulation with articulated bodies. *IEEE Transactions on Visualization and Computer Graphics 16*, 1, 70–80.

LASZLO, J., VAN DE PANNE, M., AND EUGENE, F. 1996. Limit cycle control and its application to the animation of balancing and walking. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, ACM, New York, NY, USA, SIGGRAPH '96, 155–162.

LENTINE, M., GRÉTARSSON, J., SCHROEDER, C., ROBINSON-MOSHER, A., AND FEDKIW, R. 2010. Creature Control in a Fluid Environment. *IEEE Transactions on Visualization and Computer Graphics 17*, 5, 682–693.

LINDSEY, C. 1978. Form, function, and locomotory habits in fish. *Fish physiology 7*, 1–100.

LOSASSO, F., GIBOU, F., AND FEDKIW, R. 2004. Simulating water and smoke with an octree data structure. In *ACM SIGGRAPH 2004 Papers*, ACM, New York, NY, USA, SIGGRAPH '04, 457–462.

MILLER, G. 1989. Goal-directed snake motion over uneven terrain. In *Computer Graphics '89*, 257–272.

MORDATCH, I., DE LASA, M., AND HERTZMANN, A. 2010. Robust physics-based locomotion using low-dimensional planning. In *ACM SIGGRAPH 2010 papers*, ACM, New York, NY, USA, SIGGRAPH '10, 71:1–71:8.

MUICO, U., LEE, Y., POPOVIĆ, J., AND POPOVIĆ, Z. 2009. Contact-aware nonlinear control of dynamic characters. In *ACM SIGGRAPH 2009 papers*, ACM, New York, NY, USA, SIGGRAPH '09, 81:1–81:9.

ROBINSON-MOSHER, A., SHINAR, T., GRETARSSON, J., SU, J., AND FEDKIW, R. 2008. Two-way coupling of fluids to rigid and deformable solids and shells. In *ACM SIGGRAPH 2008 papers*, ACM, New York, NY, USA, SIGGRAPH '08, 46:1–46:9.

SHIRGAONKAR, A., CURET, O., PATANKAR, N., AND MACIVER, M. 2008. The hydrodynamics of ribbon-fin propulsion during impulsive motion. *J. Exp. Biol 211*, 3490–3503.

SIMS, K. 1994. Evolving virtual creatures. In *Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, ACM, New York, NY, USA, SIGGRAPH '94, 15–22.

SOK, K. W., KIM, M., AND LEE, J. 2007. Simulating biped behaviors from human motion data. In *ACM SIGGRAPH 2007 papers*, ACM, New York, NY, USA, SIGGRAPH '07.

STAM, J. 1999. Stable fluids. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, SIGGRAPH '99, 121–128.

STANTON, A., AND UNKRICH, L. 2003. Finding Nemo [motion picture]. *United States: Walt Disney Pictures*.

TAKAHASHI, T., UEKI, H., KUNIMATSU, A., AND FUJII, H. 2002. The simulation of fluid-rigid body interaction. In *ACM SIGGRAPH 2002 conference abstracts and applications*, ACM, New York, NY, USA, SIGGRAPH '02, 266–266.

TAN, J., LIU, K., AND TURK, G. 2011. Stable proportional-derivative controllers. *Computer Graphics and Applications*.

TERZOPOULOS, D., TU, X., AND GRZESZCZUK, R. 1994. Artificial fishes: Autonomous locomotion, perception, behavior, and learning in a simulated physical world. *Artificial Life 1*, 4, 327–351.

TU, X., AND TERZOPOULOS, D. 1994. Artificial fishes: physics, locomotion, perception, behavior. In *Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, ACM, New York, NY, USA, SIGGRAPH '94, 43–50.

TYTELL, E., AND LAUDER, G. 2004. The hydrodynamics of eel swimming. I. Wake structure. *Journal of Experimental Biology 207*, 11, 1825–1841.

WAMPLER, K., AND POPOVIĆ, Z. 2009. Optimal gait and form for animal locomotion. In *ACM SIGGRAPH 2009 papers*, ACM, New York, NY, USA, SIGGRAPH '09, 60:1–60:8.

WANG, J. M., FLEET, D. J., AND HERTZMANN, A. 2010. Optimizing walking controllers for uncertain inputs and environments. In *ACM SIGGRAPH 2010 papers*, ACM, New York, NY, USA, SIGGRAPH '10, 73:1–73:8.

WU, J.-C., AND POPOVIĆ, Z. 2003. Realistic modeling of bird flight animations. In *ACM SIGGRAPH 2003 Papers*, ACM, New York, NY, USA, SIGGRAPH '03, 888–895.

WU, J.-C., AND POPOVIĆ, Z. 2010. Terrain-adaptive bipedal locomotion control. In *ACM SIGGRAPH 2010 papers*, ACM, New York, NY, USA, SIGGRAPH '10, 72:1–72:10.

WYNEKEN, J. 1997. Sea Turtle Locomotion: Mechanisms, Behavior, and Energetics. *The biology of sea turtles*, 165.

YANG, P.-F., LASZLO, J., AND SINGH, K. 2004. Layered dynamic control for interactive character swimming. In *Proceedings of the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animation*, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, SCA '04, 39–47.

YE, Y., AND LIU, C. K. 2010. Synthesis of responsive motion using a dynamic model. *Computer Graphics Forum (Eurographics) 29*, 2.

YIN, K., LOKEN, K., AND VAN DE PANNE, M. 2007. Simbicon: simple biped locomotion control. In *ACM SIGGRAPH 2007 papers*, ACM, New York, NY, USA, SIGGRAPH '07.

YIN, K., COROS, S., BEAUDOIN, P., AND VAN DE PANNE, M. 2008. Continuation methods for adapting simulated skills. In *ACM SIGGRAPH 2008 papers*, ACM, New York, NY, USA, SIGGRAPH '08, 81:1–81:7.

ZORDAN, V. B., AND HODGINS, J. K. 2002. Motion capture-driven simulations that hit and react. In *Proceedings of the 2002 ACM SIGGRAPH/Eurographics symposium on Computer animation*, ACM, New York, NY, USA, SCA '02, 89–96.