Realistic Biomechanical Simulation and Control of Human Swimming

WEIGUANG SI
University of California, Los Angeles
SUNG-HEE LEE
Korea Advanced Institute of Science and Technology
EFTYCHIOS SIFAKIS
University of Wisconsin, Madison
and
DEMETRI TERZOPOULOS
University of California, Los Angeles

We address the challenging problem of controlling a complex biomechanical model of the human body to synthesize realistic swimming animation. Our human model includes all of the relevant articular bones and muscles, including 103 bones (163 articular degrees of freedom) plus a total of 823 muscle actuators embedded in a finite element model of the musculotendinous soft tissues of the body that produces realistic deformations. To coordinate the numerous muscle actuators in order to produce natural swimming movements, we develop a biomimetically motivated motor control system based on Central Pattern Generators (CPGs), which learns to produce activation signals that drive the numerous muscle actuators.

Categories and Subject Descriptors: I.3.7 [**Computer Graphics**]: Three-Dimensional Graphics and Realism—*Animation*

General Terms: Algorithms, Theory

Additional Key Words and Phrases: Biomechanical human simulation, CPG biomechanical control, human swimming

During a three-month stay in Korea, W. Si was supported in part by the Global Frontier R&D Program of the National Research Foundation of Korea (2012M3A6A3055690). E. Sifakis was supported in part by National Science Foundation grants IIS-1253598, CNS-1218432, and IIS-1407282. D. Terzopoulos acknowledges the support of an Okawa Foundation Research grant for "Realistic Human Simulation."

Authors' addresses: W. Si (corresponding author), Neuromuscular Biomechanics Laboratory, Stanford University, Stanford, CA 94305; email: forswg@gmail.com; S.-H. Lee, Korea Advanced Institute of Science and Technology, 373-1 Guseong-dong, Yuseong-gu, Daejeon, South Korea; E. Sifakis, University of Wisconsism, Madison, Madison, WI 53706; D. Terzopoulos, University of California, Los Angeles, CA 90095.

DOI: http://dx.doi.org/10.1145/2626346

ACM Reference Format:
Weiguang Si, Sung-Hee Lee, Eftychios Sifakis, and Demetri Terzopoulos. 2014. Realistic biomechanical simulation and control of human swimming. ACM Trans. Graph. 34, 1. Article 10 (November 2014) 15 pages.
DOI: http://dx.doi.org/10.1145/2626346

1. INTRODUCTION

The simulation of human motion is of interest in computer graphics, robotics, biomechanics, control theory, and other disciplines. Among the many approaches proposed to synthesize human movement, efforts that involve modeling the detailed anatomical structure and biomechanical characteristics of the human body, in conjunction with the design of motion controllers ideally capable of adapting to the body's environment, have progressed steadily. Despite the progress, it remains a grand challenge to achieve anatomically detailed simulation of human motion with impeccable realism.

To synthesize realistic, anatomically detailed human animations in a physics-based manner, we must inevitably construct a comprehensive human model with synthetic hard (bone) and soft (flesh) tissues properly coupled and simulated, and we must also design sophisticated motor controllers in order for such a biomechanical model to produce natural, lifelike human motions in its environment. In the work reported herein, we are especially interested in aquatic environments for several reasons: On the one hand, the dynamically rich physical interaction of the human body with water provides a fertile proving ground that confronts a biomechanical human simulation/control system with interesting and difficult motor control problems. On the other hand, the aquatic environment is somewhat forgiving in that it has a stabilizing effect, which leads to nonetheless interesting control scenarios that serve as good starting points for designing more sophisticated human motor controllers suitable for terrestrial environments. There are many elegant human motions possible in the aquatic environment that deserve study from the perspective of simulation and control, such as swimming for locomotion, artistic synchronized swimming, water polo, diving, etc.

1.1 Multiphysics Simulation Framework

We introduce a multiphysics simulation framework for realistic swimming within which we develop a detailed biomechanical model of the human body and a biomimetically motivated controller for synthesizing various swimming motions (Figure 1). From

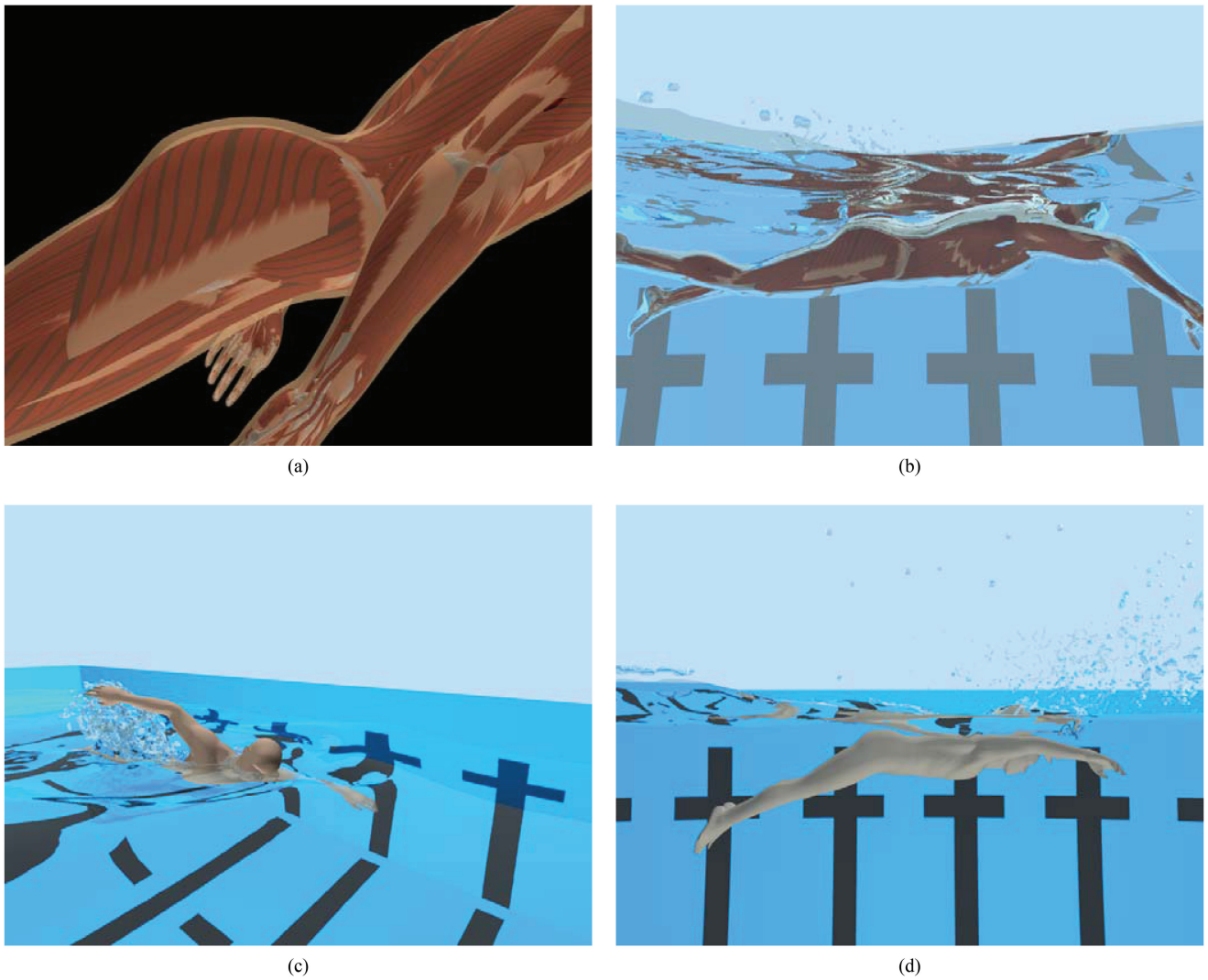

Fig. 1. A biomechanically simulated/controlled human swimmer. (a) Closeup view of the biomechanical model rendered with transparent skin to reveal the muscle geometries. (b) Biomechanical model immersed in simulated water. The autonomously controlled biomechanical model simulates swimming in crawl (c) and butterfly (d) strokes.

the biomechanics perspective, our human model includes all of the relevant articular bones and muscles, including 103 rigid bones plus a total of 823 muscle actuators, modeled as Hill-type, uniaxial contractile Musculotendinous Actuators (MAs). We employ multi-rigid-body dynamics to simulate the articulated musculoskeletal motions. To simulate the dynamic deformation of flesh and muscles, we employ a lattice-based discretization of quasi-incompressible elasticity augmented with active contractile muscle terms. To simulate the physics of the water environment in which the biomechanically simulated body floats, we employ an Eulerian (Navier-Stokes) fluid simulation on a MAC grid and use a particle-level-set method to track the surface of the water. Thus, our multiphysics simulator encompases rigid-body, deformable, and fluid regimes.

We deal with the coupling between bone and flesh as well as that between flesh and water in an interleaved manner, which has several advantages over tight two-way couplings, as tightly coupling the articulated rigid bodies and deformable solid and surrounding fluid would be challenging and costly. Interleaved coupling makes our simulation framework much more flexible and allows for the reuse and improvement of the individual simulation components.

1.2 Controlling the Biomechanical Human Model

A primary focus of this article is the challenging problem of controlling the biomechancial human model. In particular, we develop a *locomotion controller* that produces realistic swimming; that is, we present a successful approach to controlling the numerous muscle actuators in order to synthesize naturally repetitive body motions that enable our human model to produce self-propelled movement in the simulated fluid environment.

We develop a biomimetic motor control system based on *Central Pattern Generators (CPGs)*, which produces activation signals that drive the many Hill-type MAs. CPGs are biological neural networks capable of producing rhythmic outputs even in isolation from motor and sensory feedback. They offer important advantages, such as

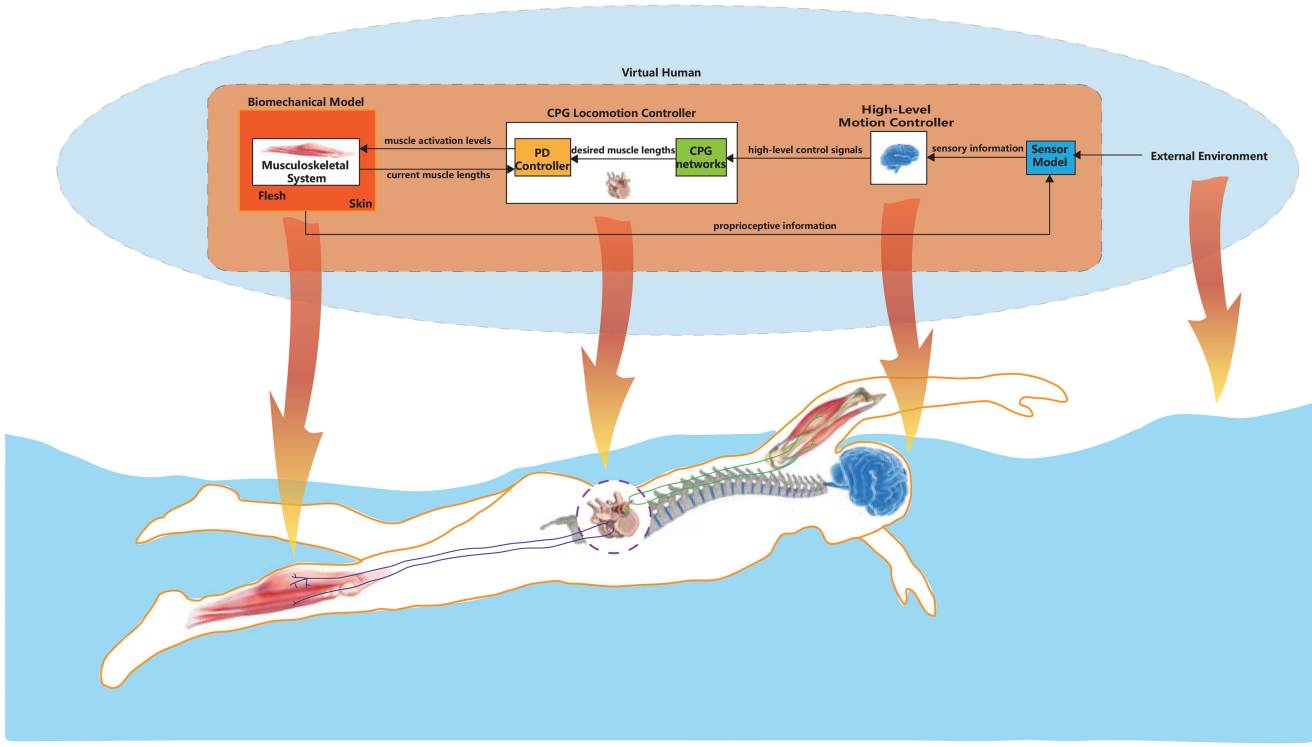

Fig. 2. Overview of our biomimetic human swimming simulation and control framework.

producing stable rhythmic motor patterns that are easily modulated, which are very desirable in biomechanical control.

To control the virtual swimmer's body, we design CPG networks that produce muscle activation signals to induce muscle contraction forces. These enable the human model to swim in various ways. Each CPG unit associated with a muscle actuator is modeled as a nonlinear dynamical oscillator with good stability and convergence properties. The easy modulation property implies that only a few parameters (such as amplitude, frequency, and phase) need be adjusted in order to achieve different swimming tasks.

1.3 Overview

Figure 2 illustrates the overall biomimetic structure of our human swimming simulation and control framework. Our autonomous virtual human comprises the biomechanical body model with its skeletal, active muscular, and passive soft-tissue components, and a brain model with a perception center that encompasses proprioception as well as the sensing of visual targets in the environment. The motor center of the brain has a low-level CPG locomotion controller (emulating biological CPG networks in the spinal cord) and one that produces higher-level motor signals such as swimming style, speed, turn direction/sharpness, etc., taking the perceptual information into account. Given these motor signals as inputs, the CPG networks automatically synthesize the desired muscle length signals online, from which a proportional/derivative (PD) control mechanism produces the associated muscle activation levels. The activation lavels innervate the muscles whose contractions actuate the biomechanical body. Our multiphysics simulation framework simulates the biomechanical human model along with the aquatic environment in which it is situated, as well as their physical interaction.

The remainder of this article is organized as follows: Section 2 reviews relevant research in the graphics, robotics, and biomechanics literature. Section 3 presents our multiphysics simulation framework and, along with Appendix A, details all the simulation components and the dynamic couplings among them. Section 4 develops our CPG-based locomotion controller that works within our simulation framework to produce natural swimming motions. Section 5 reports our experiment results. Within our simulation framework, our complex yet appropriately controlled human model demonstrates coordinated swimming tasks. Section 6 discusses the limitations of our work and, along with Appendix B, compares alternative approaches for the key components of our simulation and control framework. Section 7 presents conclusions and proposes avenues for future work.

2. RELATED WORK

Our work builds upon relevant technical advances in computer graphics, robotics, and biomechanics to model the biomechanical characteristics of the human body and to emulate its motor control mechanisms, as well as to simulate the continuum mechanics of the relevant solids and fluids.

2.1 Biomechanical Human Modeling

In graphics, researchers have traditionally used joint torques to drive articulated skeletal animation [Hodgins et al. 1995; Faloutsos et al. 2001], in contrast to facial animation where muscle actuators have been used for over two decades to synthesize expressions [Lee et al. 1995]. As a means of improving realism, skeletal-muscle-driven motion generation is receiving growing attention and researchers have been developing increasingly sophisticated biomechanical

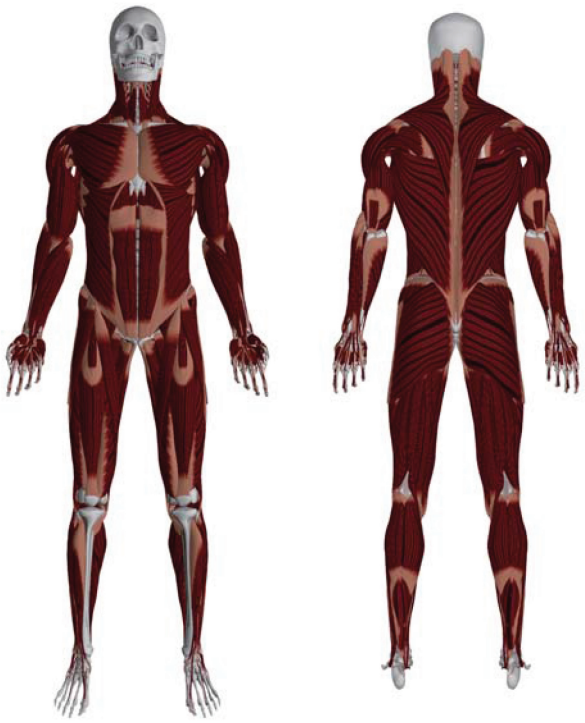

Fig. 3. Rendering of the musculoskeletal model.

models of individual body parts actuated by muscles, such as the arm [Albrecht et al. 2003; Tsang et al. 2005; Sueda et al. 2008], leg [Komura et al. 2000; Dong et al. 2002; Wang et al. 2012], neck [Lee and Terzopoulos 2006], trunk [Zordan et al. 2006], and of the entire body [Nakamura et al. 2005]. The closest precedent to the biomechanical human model that we have developed for the work reported herein is the upper-body musculoskeletal model reported in Lee et al. [2009], which employed a one-way coupling between flesh and bones. Our new model is a full-body comprehensive human model with two-way flesh-bone coupling.

2.2 Underwater Motion Simulation

Early work on simulating the underwater movements of aquatic creatures adopted rather simple solid and fluid models [Tu and Terzopoulos 1994; Yang et al. 2004]. As the simulation techniques for solids and fluids advance, researchers have used increasingly sophisticated fluid models and solid-fluid coupling techniques. In their work on the interaction of multiphase flow (specifically water including air bubbles) with animated bodies, Mihalef et al. [2008] produced a 7-second animation clip that demonstrates a human figure making swimming-like body movement while floating in simulated water. Kwatra et al. [2010] and Tan et al. [2011] used a simplified articulated body representation and two-way coupling between the body and a fluid simulation to model creatures locomoting in fluids. Lentine et al. [2011] employed articulated skeletons with a deformable skin layer and two-way coupling to a fluid simulator to model figures moving in fluids. We too employ an articulated (human) skeleton, but also include nonrigid simulated flesh, and use two-way coupling between the deformable skin and water to synthesize natural human aquatic motion.

2.3 Underwater Motion Control

Motion control in underwater creatures was pioneered by Tu and Terzopoulos [1994]. Grzeszczuk and Terzopoulos [1995] achieved optimal parameters for underwater gait behavior in rather simple creatures through spatial-temporal optimization methods. Tan et al. [2011] proposed a covariance-matrix-adaptation-based optimization to create realistic swimming behavior for a given articulated creature body. However, achieving sophisticated human swimming styles through spatial-temporal optimization is a huge challenge, as one must define a tailored objective function for each style. Other methods have therefore been developed to create gait motions for more complex systems such as humans. Yang et al. [2004] developed a layered strategy for human swimming control in which each control layer is procedurally modeled and empirically tuned to create physics-based swimming motion in real time. Kwatra et al. [2010] developed a swimming controller that computes the necessary joint torques to follow captured human motions that mimic swimming.

We develop a CPG-based locomotion controller that, after learning a few parameters, automatically generates muscle contraction signals that enable the human model to perform swimming motions. Our controller is able to achieve more complex tasks, such as changing speed, turning, style transition, etc. CPGs are neural circuits found in both invertebrate and vertebrate animals that can produce rhythmic patterns of neural activity without receiving rhythmic inputs. Research in biology and robotics has shown that animal locomotion is in large part based on CPGs [MacKay-Lyons 2002; Ijspeert 2008]. CPG models have already been successfully applied to robotic control. Ijspeert et al. [2007] built an amphibious salamander robot controlled by CPG models, and Righetti and Ijspeert [2006] developed a programmable CPG for the online generation of periodic signals to control bipedal locomotion in a simulated robot. Taga [1995] constructed a human locomotion controller based on CPGs and Hase et al. [2003] optimized this controller for 3D musculoskeletal models without activation dynamics. The aforementioned efforts employ CPGs to generate desired joint angle signals, whereas we use CPGs to generate the desired muscle contractions. In our case, muscle contraction control has several advantages over joint angle control, among them easy computation of the activation levels needed to drive the contractile muscle actuators using a simple feedback scheme, which makes it very suitable for controlling our biomechanical human model.

3. SIMULATION AND COUPLING

Our multiphysics simulation framework for realistic human swimming comprises three mutually coupled specialized component simulators—an articulated multibody simulator for the skeleton, a (Lagrangian) deformable solid simulator for the flesh and muscles, and a (Eulerian) fluid simulator for the water. In this section, we will detail how we employ these simulators in an interleaved manner to animate swimming and related underwater motions using a biomechanical human model.

3.1 Overview

For our purposes in this article, we have developed a comprehensive biomechanical human model (Figures 3, 4, and 5) with 103 rigid bones (comprising 163 articular degrees of freedom), including the vertebrae and ribs, that is actuated by 823 muscles modeled as piecewise uniaxial, Hill-type musculotendinous actuators. The skeleton is simulated as an articulated, multibody dynamical system. The

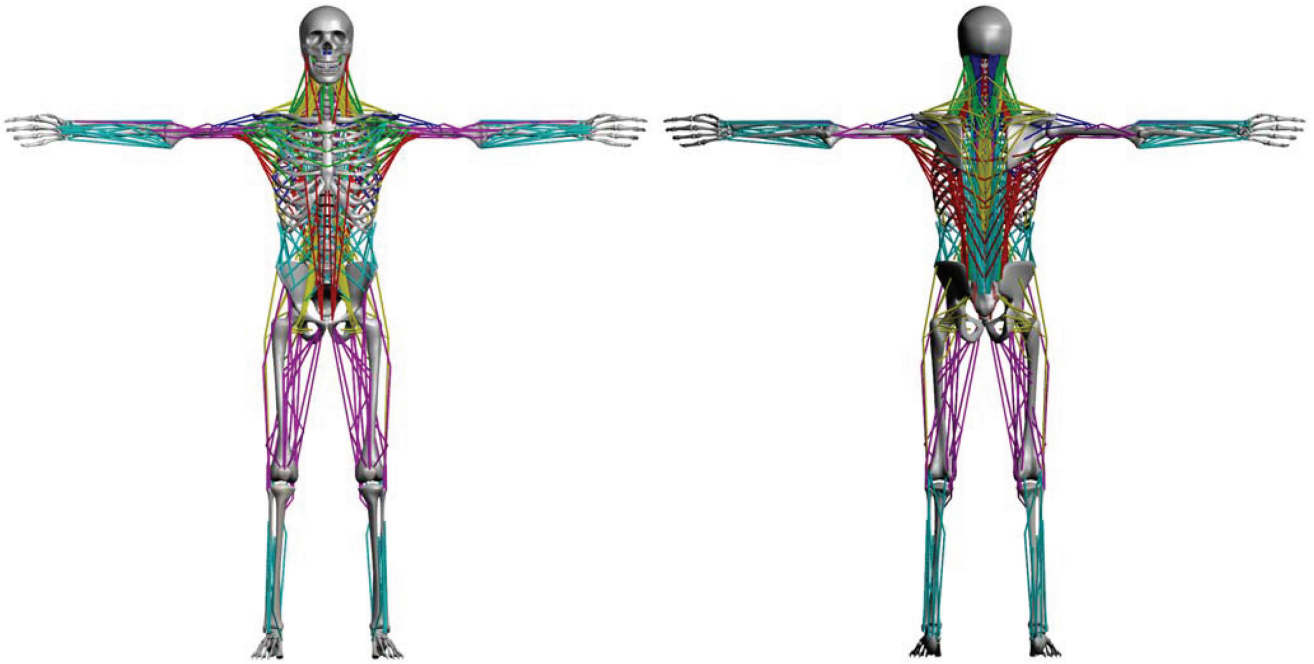

Fig. 4. The 823 Hill-type musculotendinous actuators (MAs).

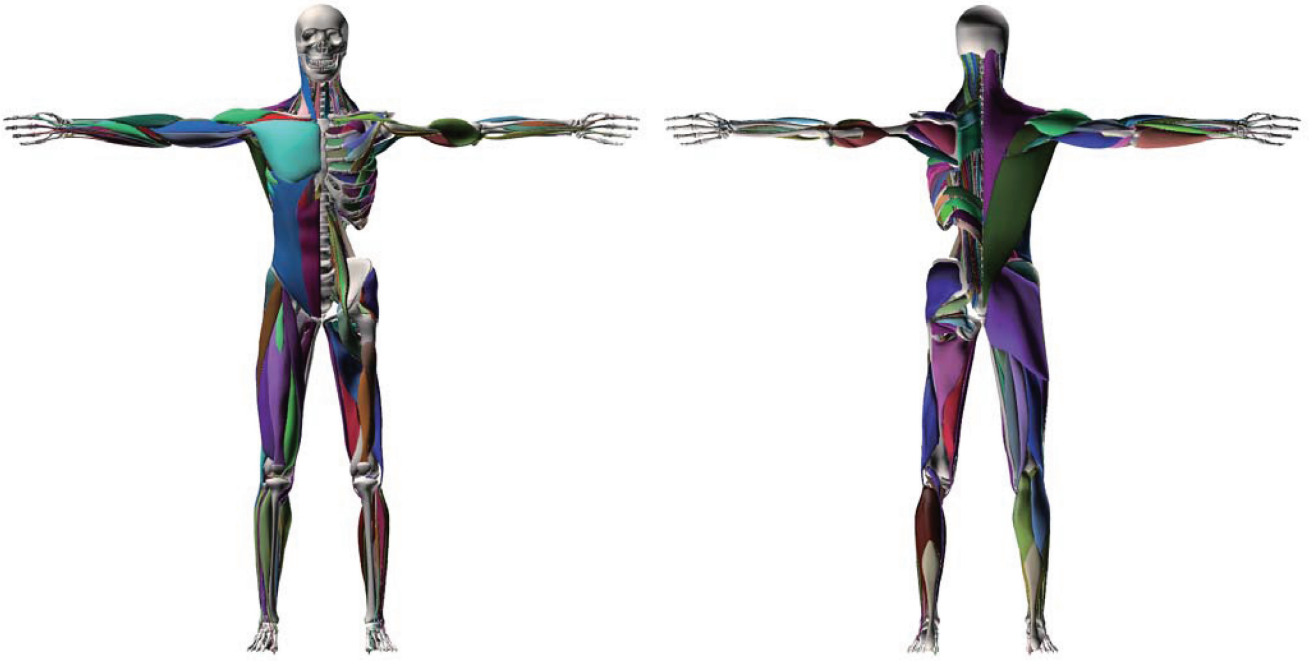

Fig. 5. Muscle geometries; superficial muscles on the left side of the body are not shown so as to reveal the deeper muscles beneath them.

deformable 3D muscle and passive flesh simulation is accomplished by a lattice-based discretization of quasi-incompressible elastic material augmented with active muscle terms. The inertial properties of the skeleton are approximated from the dense volumetric physical parameters of the soft-tissue elements, that is, each bone's inertial tensor is augmented by the inertial parameters of its associated soft tissues. The natural dynamics of the simulated human are induced by muscle forces generated by the contractile actuators. The surrounding water in which the biomechanical human model floats is simulated according to the Navier-Stokes equations using an Eulerian fluid solver. Our simulation framework implements the natural dynamic couplings between the flesh and

skeleton, as well as between the deformable skin surface of the virtual human and the surrounding water, in an interleaved manner.

3.2 Simulation Components

The Navier-Stokes equations for the water are simulated using an Eulerian method on a MAC grid and the water surface is tracked using the particle-level-set method, in accordance with Enright et al. [2002] and Foster and Fedkiw [2001].

The force generating characteristic of the MA is governed by a linearized Hill-type muscle model. Assuming the length of the tendon is constant, we model a muscle force as the sum of forces from a contractile element (CE) and a parallel element (PE). The PE force accounts for the passive elasticity of a muscle while the CE represents the active muscle force that is controlled by the motor neurons. Additional details can be found in Lee et al. [2009].

The low-level control inputs of our biomechanical human model comprise the activation levels of each muscle (Section 4 describes how these muscle activation levels are determined). The activated muscles generate forces that drive the skeletal simulation. Given the contractile muscle forces plus the external forces from the flesh simulation, we simulate the skeleton using the Articulated Body Method [Featherstone 1987] to compute the forward dynamics in conjunction with a backward Euler time-integration scheme as in Lee et al. [2009]. For the purpose of simulating the dynamic deformation of the flesh and muscles, we employ a lattice-based discretization of quasi-incompressible elasticity [Patterson et al. 2012] augmented with active muscle terms. This approach avoids the need for multiple meshes conforming to individual muscles and its regular structure offers significant opportunities for performance optimizations. Appendix A provides additional implementation details.

3.3 Coupling Framework

We demonstrate our overall multiphysics coupling framework in Figure 6. In Figure 6(a), circled numbers tag the simulated components and interfaces that are involved in our couplings: ① denotes the bones, ② denotes the flesh-bone interface, ③ denotes the muscles, ④ denotes the passive flesh, ⑤ denotes the skin-fluid interface, and ⑥ denotes the fluid. The following five steps, which are illustrated in Figures 6(b)–(f), are repeated in every coupling cycle.

(1) The fluid forces are computed on the immersed skin surface (Figure 6(b)).
(2) Given the fluid forces and the attachment spring forces from the bones, the flesh simulation is advanced to equilibrium, which also transfers the external forces acting on the skin surface to the bones (Figure 6(c)), where the flesh-bone and skin-fluid gaps are exaggerated for clarity and the white region inside the flesh is hollow).
(3) Given the muscle forces and attachment spring forces from the flesh, the skeleton simulation is then advanced to the next timestep (Figure 6(d)), where the dashed lines indicate the bone positions from the previous timestep (Figure 6(c) to illustrate the movement of the bones).
(4) In the new bone configuration, the flesh simulation is again advanced to equilibrium, subject to the fluid forces and new attachment spring forces from the bones (Figure 6(e)).
(5) Finally, given the new skin surface, the fluid simulation is advanced to the next timestep (Figure 6(f)).

The next two sections further detail the flesh-bone coupling and the skin-fluid coupling.

3.4 Flesh-Bone Coupling

The deformable flesh tissue is coupled to the rigid articulated skeleton via a network of spring constraints, as has been previously demonstrated in Lee et al. [2009] and McAdams et al. [2011]. From the viewpoint of the volumetric flesh simulation, such spring attachments serve as soft constraints. They also serve in computing the aggregate force and torque that the deformable flesh exerts on each bone. In our framework, we further leverage this network of soft constraints to transfer to the bones the external forces applied to the skin surface, in a fashion that respects the deformable flesh which intervenes between the bones and the points of application of the external forces. After computing the distribution of external forces on the skin, originating from any sources including fluid forces or collisions, we solve for the quasistatic equilibrium shape of the deformable flesh. Once the steady state configuration has been computed, the tension of the attachment springs is used to calculate how the skin-applied forces have been distributed to the bone-flesh interface. From balance of force properties, we have strong guarantees that the aggregate force is applied by the attachment springs to the bones (at equilibrium), independent of the material parameters of the soft tissue or the stiffness of the attachment springs. Of course, different material parameters may have an effect on how broadly a surface force gets spread out from the point of application. This quasistatic process makes the force transfer from the flesh to the bones occur instantaneously, which eliminates a potential lag while ensuring that external forces acting on the skin will realistically influence the articulated dynamics of the skeleton.

3.5 Skin-Fluid Coupling

The traditional method for coupling fluids and solids is for the solid to prescribe velocity boundary conditions on the fluid and for the fluid to provide force boundary conditions on the solid [Benson 1992]. Accordingly, we also use the velocity of the human body model skin surface to enforce the Neumann boundary condition along the surface by making the normal component of the fluid velocity equal to that of the skin's velocity. To calculate the force of the fluid on the body, we would ideally integrate over the skin surface the pressure computed by the fluid solver. For incompressible flow, however, the pressure (that serves as a penalty term in the Navier-Stokes computation) is both stiff and noisy, hence more or less unreliable, as discussed in Fedkiw [2002].[1] As a consequence, instead of demanding a higher degree of accuracy in the pressure computation from our underlying fluid simulation engine, we opt for a computation of fluid-to-solid forces based on fluid velocities, which are generally more accurate and temporally coherent. We use the relative velocity of the human skin with respect to the fluid to compute the hydrodynamic force and we construct a new level-set representation of the water to compute the buoyancy force. These forces due to the water acting on the body are computed at each triangle of the skin surface and applied to the skin as external forces.

To compute the hydrodynamic force on each triangle of the skin surface, we employ a simplified hydrodynamic force model similar to those found in Tu and Terzopoulos [1994], Yang et al. [2004], and Lentine et al. [2011]:

$$\mathbf{f} = \min\left[0, -\rho A\,(\mathbf{n}\cdot\mathbf{v})\right](\mathbf{n}\cdot\mathbf{v})\,\mathbf{n}, \tag{1}$$

[1] While the velocity field is a primary state variable and limited in its temporal variation due to momentum conservation, the pressure field is a byproduct of the projection of velocities into a divergence-free field, and may exhibit notably higher temporal variance than the fluid velocities.

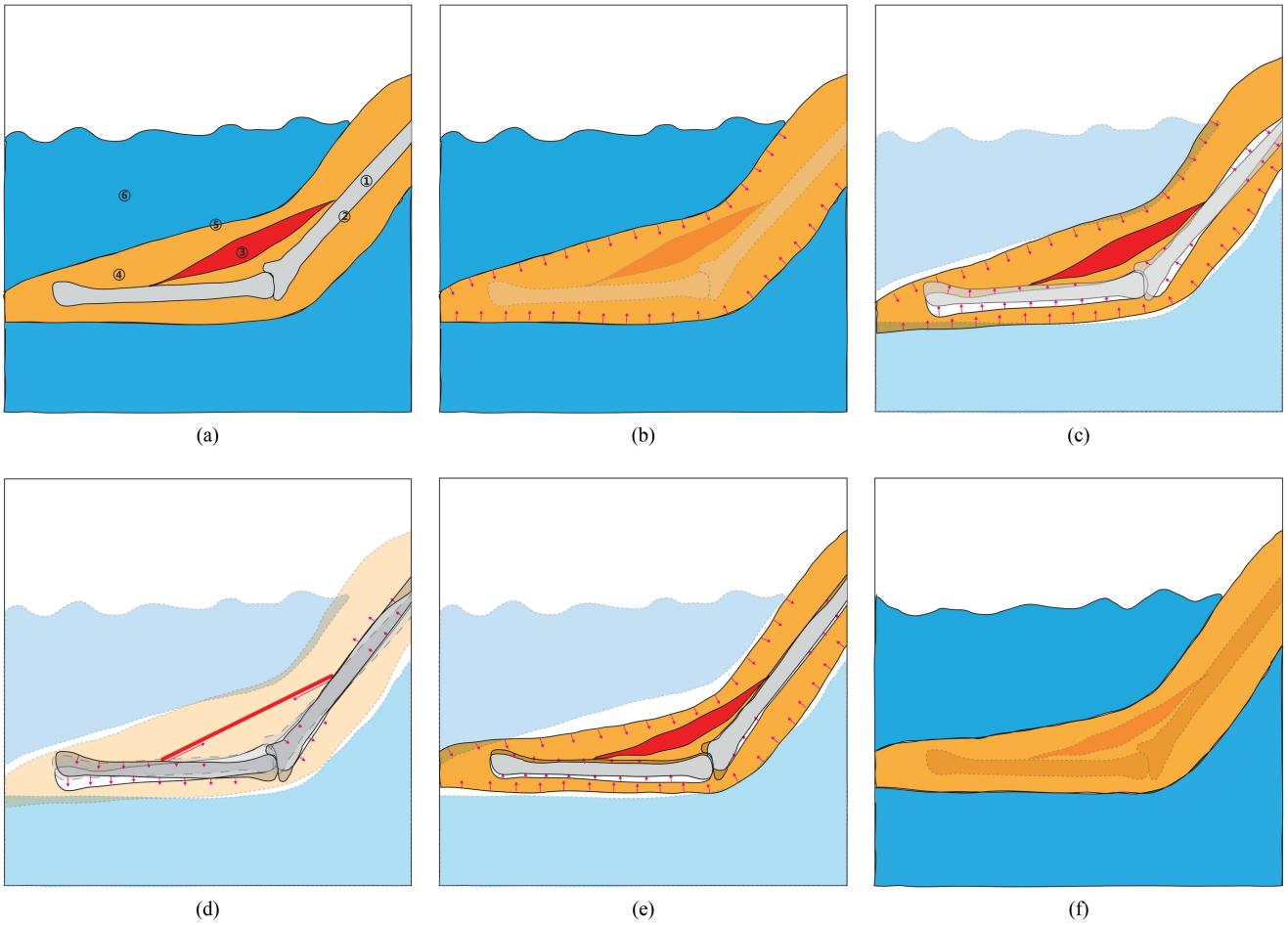

Fig. 6. Overview of our multiphysics coupling framework.

where ρ is the density of the water, A is the area of the triangle, $\mathbf{n}$ is its normal, and $\mathbf{v}$ is its velocity relative to the water. To enforce the boundary conditions in the fluid solver, we must make the normal component of the fluid velocity equal to that of the solid's velocity, so we cannot use the fluid velocity on the boundary cell to compute the relative velocity (as its normal component will be approximately zero). Instead, we accumulate velocities of the fluid in neighboring cells around the boundary cell in which the skin triangle lies and employ the mean local fluid velocity to compute the relative velocity $\mathbf{v}$.

The total buoyancy force acting on the floating body equals the weight of water displaced by the body. For underwater motion with the body wholly immersed, the buoyancy approximately cancels out the gravity force, since the average density of the human body approximately equals the density of water. However, this is not the case for swimming where the human body is often only partially immersed. It is therefore important to compute buoyancy correctly in order to simulate realistic dynamic trunk motions, especially for the butterfly swimming style. We can represent the buoyancy as $\mathbf{B} = -\rho \mathbf{g} V$, where $\mathbf{g}$ is the gravitational acceleration and V is the volume of water displaced by the body. We may rewrite this as

$$\mathbf{B} = \rho \mathbf{g} \int_S h(\mathbf{n} \cdot \hat{\mathbf{j}}) dA, \tag{2}$$

where S is the immersed surface of the body model, $\mathbf{n}$ is the normal of the area element, $\hat{\mathbf{j}}$ is the upward unit vector, and h denotes the distance from the water surface to the area element. Thus, the force on each triangle is $\rho h A n_y \mathbf{g}$, where n_y is the y component of the normal.

The main problem is how to compute h. A simple way is using $h = y_0 - y_A$, assuming that the water surface is at a constant height y_0, where y_A is the y coordinate of the triangle center. Unfortunately, this will cause problems in the simulation, since the error can become very large when there are significant waves on the surface of the water. Even worse, the error will propagate back and forth in the interleaved two-way coupling, causing an oscillation in the motion of the floating body model. We tackle this problem by constructing a *Water PseudoSurface* (WPS) at each timestep, from which we derive h. The portion of the human body that is below this WPS is treated as the submerged part. Constructing the WPS is a minimal surface problem: We set Dirichlet boundary conditions on the human skin surface, assigning a negative Dirichlet value for skin regions that are immersed, and a positive one for areas of skin that are not in contact with water. We perform a harmonic interpolation between these values to reconstruct a zero isocontour of the level-set function that will extend the water inside the swimmer's body. Once this WPS has been reconstructed, we approximate the immersion depth by projecting the closest-surface-point vector ($-\phi \nabla \phi$, derived from the reconstructed level set) along the vertical direction.

Another benefit of the WPS is that we can use it for the purposes of rendering. Generally, the fluid and solid surfaces are not tightly coupled because of the limit in the fluid simulation resolution, so there is a noticeable gap between the water and the human body. However, since the WPS eliminates the part that is submerged, we can exploit it for rendering. The rendering results shown in Figure 1 are obtained using the WPS.

4. CPG LOCOMOTION CONTROL

The control of biomechanically simulated human swimming is a challenging problem. Swimming motions have several distinctive styles, such as butterfly and crawl, each requiring the coordinated rhythmic movement of multiple body parts.

CPGs are biological neural networks capable of generating coordinated patterns of rhythmic activity. Applied to biomechanical locomotion control problems, CPG models offer important advantages. Each individual MA in our biomechanical model has its own activation input. A CPG controls the temporally varying length of each MA and a PD feedback loop synthesizes the associated muscle activation signal. The CPG produces the desired rhythmic motor control signal, which remains stable and smoothly varying, even for abrupt changes in the control parameters. The CPG's inherent stability readily restores the biomechanical system's normal rhythmic action even after perturbative transients. Furthermore, CPG models typically involve only a few parameters that modulate their rhythmic outputs. Hence, a properly implemented CPG-based approach reduces the dimensionality of the motor control problem such that higher-level locomotion controllers need to produce only task-oriented control signals rather than an unwieldy set of low-level MA activation signals.

Our high-level swimming controller, which functions by modulating the CPG oscillators, can be simplified by grouping muscles. As illustrated in Figure 7, we divide the muscles into 10 groups—left trunk, right trunk, medial trunk, left neck, right neck, medial neck, left arm, right arm, left leg, right leg. All the muscles in each group share the same frequency and initial phase. We determined empirically that these 10 muscle groups afford adequate control over the limbs, trunk, and head to produce the swimming strokes demonstrated in Section 5, and turns can be induced by simply decreasing the activation amplitudes on one side of the body relative to their counterparts on the other. Using a larger number of muscle groups would afford finer control over body movement, albeit with increased high-level controller complexity.

Our control architecture (whose details are presented in the following sections) results in an easy-to-use biomechanical swimming controller with nontrivial functionality, such as changing speed, turning, and transitioning between swimming styles.

4.1 Generating the Desired Muscle Lengths

We use Virtual-Swim [2007] as a reference for CPG learning. For each swimming style, we manually select around 20 joint angle key poses. As CPG learning needs to use both the first and second-order derivatives of the signals (see Gams et al. [2009]), we want the muscle length data to be doubly differentiable. We first use cubic B-splines to least-squares fit the joint angle training data. From the desired kinematic skeleton motion, we determine the desired muscle length over time between the two attachment points of each muscle. We then fit B-splines to the desired muscle lengths, from whose coefficients we can easily compute the first- and second-order derivatives.

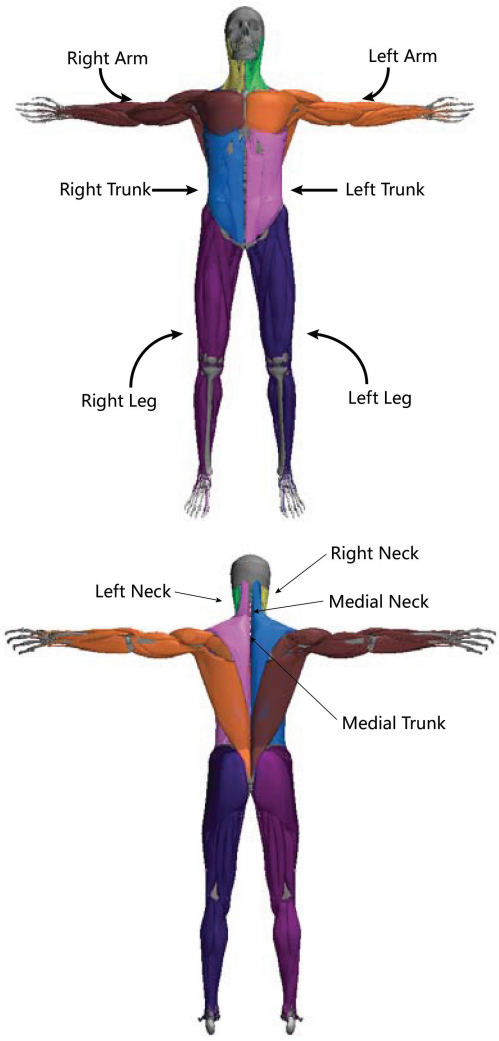

Fig. 7. Muscle grouping for CPG control. Each muscle group is shown in a different color. The medial muscle groups of the trunk and neck are less visible as they include deeper muscles.

4.2 CPG Learning

Following Gams et al. [2009], we use a group of nonlinear differential equations to model each CPG unit. The following dynamical system specifies the attractor landscape of a 1-DOF signal trajectory y oscillating around an attractor g:

$$\dot{z} = \Omega\left(\alpha_z\left(\beta_z\left(g - y\right) - z\right) + \frac{\Sigma_{i=1}^{N}\Psi_i w_i r}{\Sigma_{i=1}^{N}\Psi_i}\right) \tag{3}$$

$$\dot{y} = \Omega z, \tag{4}$$

where the $\Psi_i = \exp\left(h\left(\cos\left(\Phi - c_i\right) - 1\right)\right)$ are N Gaussian-like periodic kernel functions whose width is determined by h (in all our simulations, we set $N = 25$ and $h = 2.5N$, and c_i are equally spaced between 0 and 2π in N steps). Here, y is the generated signal, whose phase is Φ, and z is an intermediate variable that describes the first-order derivative of y. The fundamental (lowest nonzero) frequency of the input signals is Ω. Since swimming is a periodic motion, we can specify Ω as $2\pi/T$, where T is the

period of a swimming cycle. The positive constants α_z and β_z are set to $\alpha_z = 8$ and $\beta_z = 2$ for all our simulations. The amplitude control parameter is r, which we set to $r = 1$. The preceding model encapsulates several desirable properties in a single set of differential equations, such as the reproduction of the trajectories, easy modulation, and robustness against perturbations.

We use Incremental Locally Weighted Regression (ILWR) [Schaal and Atkeson 1997] to learn the weights w_i in (3) as in Gams et al. [2009]. The CPG control model allows easy modulation of the signals. Changing the parameter g modulates the baseline of the rhythmic movement. This smoothly shifts the oscillation without modifying the signal shape. Modifying Ω and r changes the frequency and the amplitude of the oscillations, respectively. Since the dynamical system is of second order, even an abrupt change in the parameters yields smooth variations in y. Although the length trajectories of different muscles may share the same frequency, the amplitudes and baseline may vary significantly. To enhance learnability, we normalize and center each muscle length trajectory to bracket the signal between -0.5 and 0.5. For convenience, we also scale the period of the input signals to 1 sec, and then use r, g, and Ω to modulate the learned signals. In the learning process, we simply set $r = 1$, $g = 0$, and $\Omega = 2\pi$.

After learning the parameters, the desired muscle lengths can be generated by numerically integrating (3) and (4) using the fourth-order Runge-Kutta method. Φ is updated as $\Phi = \Phi + \Omega\, dt$, where dt is the timestep.

Additional details about our CPG learning method are provided in Si [2013].

4.3 Muscle Control

After CPG synthesis of the desired muscle lengths, we use a first-order damping approach to compute the activation level

$$a = K_e(l - l_d) + K_d(\dot{l} - \dot{l}_d) \tag{5}$$

for each muscle, where l is the current muscle length, l_d is the desired muscle length, and K_e and K_d are elastic and damping coefficients, respectively. In our experiments, we simply set $K_e = 5/l_0$ and $K_d = 0.005/l_0$, where l_0 is the rest length of the muscle. The desired muscle length $\dot{l}_d$ is synthesized as y according to (4). As muscle activation levels range between 0 and 1, we clamp the computed activation a between 0 and 1. These generated activation levels drive the Hill-type MAs that exert forces on the skeleton and also serve as inputs to the deformable flesh simulation.

4.4 High-Level Motion Control

Our CPG-based motion controller is easy to use. After having learned several types of locomotion modes, it can easily perform them in any desired frequency, switch among modes, or achieve some desired pose. It can also control motions for different muscle groups separately; for instance, one arm can maintain some desired pose while the remaining body parts carry out a locomotion pattern.

The frequency of the movement is controlled by Ω, its phase by Φ, and its amplitude by r. A static pose can be achieved by setting $r = 0$. Not updating Φ maintains the current pose. To transition from one motion to another, we simply switch the parameters (w_i, r, and g) of the CPG units. We can do this abruptly since, per (3) and (4), this will merely cause abrupt changes in the second derivative of the desired muscle length signal $\dot{z}$. Because Ω directly influences $\dot{y}$, so long as Ω is continuous, the desired muscle length signals will be C^1-smooth. This nice property yields natural motion transitions (which can be seen in the accompanying video). See Section 5.2 for more about motion modulation.

5. EXPERIMENTS AND RESULTS

In this section, we present experimental results produced using our simulation and control framework for various swimming styles as well as changing orientation in the water environment. We refer the reader to the animations in our accompanying video.

On a 2.8GHz Intel i7 CPU with 4GB of RAM, the running times of our swimming simulator range from 3 to 10 minutes per frame with a 192 fps frame rate, depending on how many steps the adaptive timestepping fluid and deformable solid simulators execute per frame. The overhead for stepping the controller is negligible compared to the cost of the physics simulation.

For all the experiments presented, we set the water density to $1000\ \mathrm{kg/m^3}$ and the Young's modulus of the human flesh to $5 \times 10^5\ \mathrm{N/m^2}$ based on the results reported in Agache et al. [1980]. The average density of our human model is $980\ \mathrm{kg/m^3}$.

5.1 Swimming Styles

We trained our CPG control system on two different swimming styles—butterfly and crawl—which Figures 8 and 9 illustrate. The accompanying demo video shows the simulation results for these two swimming styles and compares them with video footage of a real human swimmer.

5.2 Swimming Motion Modulation

In addition to generating coordinated swimming motion, our CPG controller can also achieve more complex tasks by modulating a few high-level parameters. Swimming speed can be modulated by scaling the fundamental frequency Ω of each muscle group by the same amount. In order to generate a natural transition, we can change the frequency gradually to the desired frequency. In the accompanying video, we show the simulation result of increasing the speed of the butterfly stroke by doubling the fundamental frequency.

Swimming style transitions are accomplished by switching the parameters (w_i, r, and g) of the CPG units from one motion to a different motion. The accompanying video shows a simulation result in which the swimmer transitions from butterfly to crawl strokes.

To produce a left turn, the g of left neck and left trunk muscles are decreased, the g of the right neck and right trunk muscles are increased, and the r for all the neck and trunk muscles is decreased. Right turns are produced by doing the opposite. To execute sharp turns, we can also keep one arm straight by switching the CPG parameters of that arm muscle group to a static pose ($r = 0$). Figure 10 shows an animation sequence of the virtual swimmer making a 90-degree right turn.[2]

5.3 Anatomically Detailed Simulation

As a result of our comprehensive biomechanical human modeling, we can also demonstrate the detailed, anatomically accurate animation of the swimmer's body. In Figures 1(a)–(b) and in the accompanying video, we reveal the deformation of the swimmer's muscles by rendering the skin translucently. Figures 11 shows two closeup frames from the butterfly swimming simulation shown in the video to demonstrate the bulging of the thigh muscles as they contract to rotate the bones meeting at the knee.

[2]This is similar to the turn demonstrated by the (real) swimmer just after 1:22 in the YouTube video at the following url: `https://www.youtube.com/watch?v=YLT7YEwUCwI`.

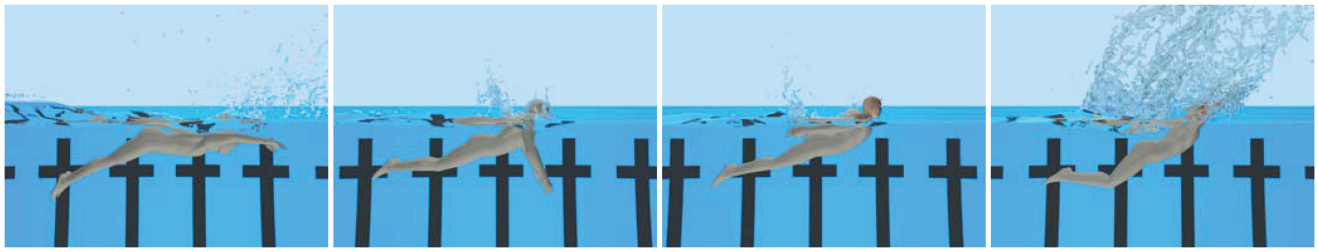

Fig. 8. Butterfly swimming sequence.

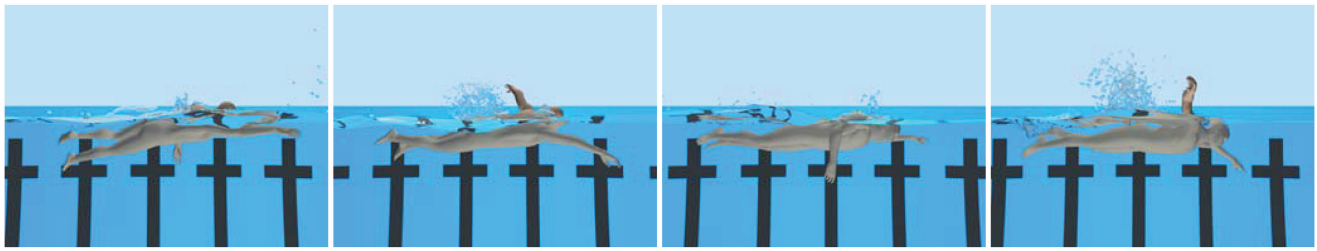

Fig. 9. Crawl swimming sequence.

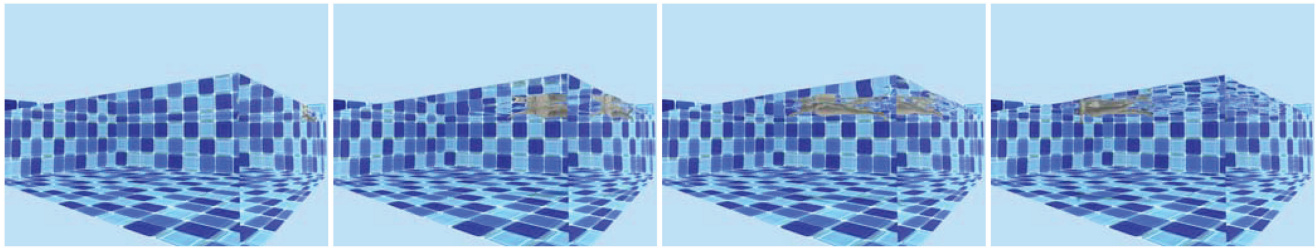

Fig. 10. Making a 90-degree right turn.

6. DISCUSSION

Our interleaved approach to coupling fluid, flesh, and skeletal components provides us flexibility and versatility in constructing a simulation and control framework from different algorithmic building blocks and simulation algorithms. However, this interleaved coupling reflects a conscious compromise in traits such as stability, accuracy, and performance potential. A fully coupled system with deformable, fluid, and rigid components would, in theory, enable implicit integration techniques that would achieve stable simulation while tolerating larger timesteps. In contrast, we take the most time step-restrictive of the phases involved (generally the fluid) and use it to dictate the timestep for the interleaved simulation cycle. As control techniques mature and the value of easy testing of modular simulation components becomes less pronounced, a closer look at a tightly coupled multiphysics/control system would certainly be appropriate.

In our biomechanical body model, forces due to the Musculotendinous Actuators (MAs) plus the volumetric flesh simulation affect the mechanical response of the skeleton. As the body pose causes stretching or compression of the flesh, reactive flesh forces act on the bones, which serves a similar purpose as the passive components of the MAs. Additionally, since our flesh simulation incorporates a contractile component controlled by the MAs, it will also transmit active flesh forces to the bones. Ideally, our volumetric simulation would capture the entirety of forces due to flesh elasticity and muscle contraction; however, producing accurate muscle forces exclusively from the flesh simulation would require a high degree of modeling accuracy, including detailed geometric and material descriptions for tendons and connective tissues. Fortunately, this is unnecessary for synthesizing natural looking flesh deformations, where deep muscle force accuracy is not a crucial factor. Conversely, while the MAs cannot model volumetric flesh deformation, they produce biomechanically faithful and stable muscle forces and torques. Our synergistic approach includes both MA and volumetric force simulations, enabling each to compensate for the limitations in the other. In particular, the MAs contribute their actuation forces to anatomically accurate bone attachment points, which compensates for the actively contractile flesh forces that are spread broadly over the bones in the absence of tendon models in the volumetric flesh simulation. On the other hand, the transfer to the bones of external forces acting on the skin relies on the volumetric flesh simulation. Overall, our combined simulation system serves as a hybrid approximation whose parameters are adapted to produce a realistic biomechanical simulation of the musculotendinous soft tissues and skeletal substructure of the human body.

Appendix B provides a comparative discussion, based on additional experiments reported therein, of more conventional alternatives to the main simulation and control components of our framework.

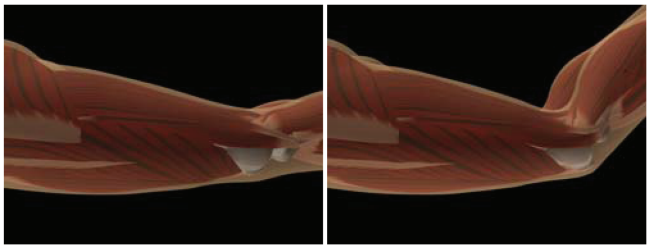

Fig. 11. Contraction and bulging of the thigh muscles (from the butterfly swimming simulation).

7. CONCLUSION AND FUTURE WORK

The main contributions of our reported research are as follows.

—We have introduced *a multiphysics simulation and control framework*, interleaving an articulated multibody simulator, a Lagrangian deformable solid simulator, and an Eulerian fluid simulator, within whose scope is the realistic animation of a sophisticated autonomous human model that is capable of controlled swimming.

—We have developed *a comprehensive biomechanical model of the human body*, which includes 103 rigid bones (comprising 163 articular degrees of freedom) simulated as an articulated, multibody dynamical system that is driven by 823 contractile muscles, modeled using piecewise uniaxial Hill-type musculotendinous units, plus a muscle and passive flesh simulation via an efficient volumetric finite element model of quasi-incompressible elastic material augmented with active (contractile) muscle terms, as well as the appropriate two-way coupling between the articulated skeleton and deformable flesh.

—With regard to *the control of the biomechanical human model* such that it produces complex coordinated locomotion, we developed a Central Pattern Generator (CPG)-based controller that generates muscle activation signals to induce appropriately coordinated muscle contractions, governed by a perceptive, higher-level, task-oriented motion controller.

Contemplating how people learn to swim, we are inspired to further investigate this topic in our future work. Humans learn to swim by first learning the movement of the limbs, perhaps by mimicking swimming demonstrations. This corresponds to the supervised learning process of our CPG system. After attaining command of the kinematic pattern of a swimming style, one can improve one's swimming skill through practice. This can be treated as an optimization process. Similarly, we can try to optimize the learned parameters of our CPG system in order to improve our biomechanical swimmer's efficiency. Generally speaking, CPG models offer a good substrate for automated learning and optimization algorithms. Studying how the swimmer responds to perturbations will be another interesting research direction. In particular, we can potentially simulate how a human should perform swimming in a torrential flow.

In the aquatic environment we do not deal with balance, and losing balance does not cause serious problems for underwater motion control in a calm water environment, since buoyancy approximately cancels gravity and humans can efficiently control their limbs to generate proper drag forces, thus making their motions controllable. Under large perturbations, however, we are forced to confront balance in order to produce controllable motion. Balance is also a very troublesome issue when controlling terrestrial motion. It will not suffice to simply apply our CPG controller to walking and running motions as we would need to develop a more sophisticated feedback scheme to handle the balance problem. This is another interesting avenue for future work. Real-world motion may be a superimposition of locomotion and voluntary movements (e.g., waving hands while walking). Combining our CPG controller with other controllers such as the neuromuscular controller developed in Lee and Terzopoulos [2006] may be a viable approach to dealing with a broader variety of motor tasks.

Energy efficiency, which is an important principle for human motion, is not considered in our swimming controller. Since global spatio-temporal optimization would be very computationally expensive for our complex simulation, it is challenging to apply the energy efficiency principle directly. A possible solution and avenue for future work would be to compute an energy-efficient controller for a simplified system and then refine it for use in our simulation framework.

APPENDIXES

A. DEFORMABLE FLESH MODEL

The elastic flesh and musculature serves as an intermediary between the fluid environment and the articulated skeleton. The shape and deformation of the flesh volume is determined by the dynamics of the articulated skeleton and the hydrodynamic forces acting on the flesh surface. Naturally, the exact tissue behavior is also dependent on the geometric layout and material properties of the heterogeneous array of tissue components that constitute the flesh. Some of these material traits are encoded as static distributions of scalar (e.g., elastic moduli) or vector (e.g., muscle fiber orientations) quantities; other material properties, such as the muscle activations, are time-varying signals that are provided as input to the flesh simulation along with the skeletal dynamics.

We capture the physical behavior of the human swimmer's soft tissue and musculature via numerical simulation of a discrete volumetric model. In designing this discrete representation, we commit to certain simplifying assuptions and modeling approximations to strike a reasonable balance between computational complexity, geometric resolution, biomechanical accuracy, and robustness of the simulation. First, we do not seperately model the skin as a distinct simulation component; for the purposes of fluid-flesh interaction, the contact surface is simply the boundary of the flesh volume and not a separate 2D skin layer. The entirety of the space between the skin and bones is modeled as an elastic continuum; no air-filled cavities or fluid volumes are explicitly simulated as such, although we are free to modulate the elastic properties (e.g., stiffness or compressibility) of such areas to reflect their macroscopic behavior. In addition, the entire flesh volume is assumed to deform as a connected continuum; that is, we do not allow slip or separation in the interior of the flesh volume. Note that connective tissue typically limits the extent of such motions, but there are parts of anatomy where true sliding or separation is possible in the real human body.

A.1 Lattice Representation

We use a lattice-based representation (in essence, a lattice deformer) to capture the shape of the deforming flesh volume. This discrete model is simply created by superimposing a cubic lattice (we use a lattice size of 10mm) on a 3D model of the human body, and we discard all cells that do not intersect the flesh volume (i.e., cells that are outside the body, or wholly within solid bones). Of course, the lattice representation thus created does not accurately capture the geometry of the flesh volume, but provides only a "cubed"

approximation. Despite this, we construct the discrete governing equations so as to compensate for this geometric discrepancy. We discretize the elasticity equations following the methodology of Patterson et al. [2012], which captures the fact that lattice elements on the boundary of the flesh volume are only fractionally covered by elastic material. The jagged boundary of the lattice-derived simulation volume also differs from the actual skin surface where fluid forces are to be applied; we compensate for this by embedding a high-resolution skin surface mesh within the cubic lattice and distributing the forces acting on the skin surface into the volumetric lattice by scaling with the appropriate embedding weights, as discussed in Zhu et al. [2010]. Finally, since the contact surface between the flesh and bones is not resolved in the lattice-derived mesh, we use stiff zero rest-length springs to elastically attach points sampled on bone surfaces to embedded locations in the flesh simulation lattice, as detailed by Lee et al. [2009] and McAdams et al. [2011].

We shall further discuss these modeling traits after presenting the material model for the elastic flesh volume.

A.2 Flesh Constitutive Model

Due to the lattice-based nature of our discretization, the exact shape of the active muscles is not fully captured in our elastic flesh deformer. Although it is possible to replicate the approach of Patterson et al. [2012] and adapt a quadrature scheme to capture the localized presence of an active muscle within a lattice cell, we found it adequate to average the effect of the muscle with respect to each lattice cell that it intersects, an approach similar to what Lee et al. [2009] employed in their tetrahedral discretization. Specifically, for a given cell of our lattice deformer, we compute the fractional coverage $d_m \in (0, 1]$ by the volume of muscle m that it contains. In other words, a cell that is fully inside the volume of muscle m would have $d_m = 1$, while a muscle that covers only 25% of the cell in question would yield $d_m = 0.25$. We refer to these volume fractions as *muscle densities*, which are used to modulate the mechanical effect that a given muscle has within each cell. We similarly define the muscle fiber orientations $\mathbf{f}_m$ on a per-cell basis to be the averaged orientation of the muscle fiber field within the lattice element fraction covered by muscle m. In practice, we compute both d_m and $\mathbf{f}_m$ by Monte-Carlo integration, and the cost of this preprocessing step is sustained only at the time of model creation.

We model the material response of the elastic flesh as a background isotropic substrate augmented by an additional response due to the presence of muscles. Thus, our constitutive model is defined as a weighted average of the constitutive models for passive flesh (Ψ_p) and contractile muscles (Ψ_m), using the previously computed muscle densities d_m, as

$$\Psi(\mathbf{F}) = \Psi_p(\mathbf{F}) + \sum_m d_m \Psi_m(\mathbf{F}) + \Psi_v(\mathbf{F}), \tag{6}$$

which also includes a volume conservation term Ψ_v that forces the flesh volume to remain nearly incompressible.

The passive flesh is modeled as an isotropic, quasi-incompressible Mooney-Rivlin material [Bonet and Wood 1997], leading to the following formula for its strain energy density Ψ_p in (6):

$$\Psi_p = \mu_{10}(\operatorname{tr}\hat{\mathbf{C}} - 3) + \frac{1}{2}\mu_{01}\left[\operatorname{tr}^2\hat{\mathbf{C}} - \hat{\mathbf{C}} : \hat{\mathbf{C}} - 6\right], \tag{7}$$

where $\hat{\mathbf{C}} = \hat{\mathbf{F}}^T\hat{\mathbf{F}}$ is the deviatoric Cauchy strain tensor, $\hat{\mathbf{F}} = J^{-1/3}\mathbf{F}$ is the deviatoric component of the deformation gradient, and $J = \det\mathbf{F}$ is the local volume change ratio.[3] We use the values $\mu_{01} = 0.06$ MPa and $\mu_{10} = 0.02$ MPa for the moduli of elasticity.

Each muscle that intersects a given lattice cell supplements the cell's strain energy density in (6) by the scaled contribution $d_m\Psi_m(\mathbf{F})$. The term $\Psi_m(\mathbf{F})$ is in fact only dependent on the along-fiber elongation or contraction, computed as

$$\lambda_m = \|\hat{\mathbf{F}}\mathbf{f}_m\|. \tag{8}$$

Following the formulation in Blemker and Delp [2005], we define $\Psi_m(\lambda_m)$ indirectly via its derivative

$$\frac{\partial\Psi_m(\lambda_m)}{\lambda_m} = \frac{\sigma_{\max}}{\lambda_{\text{opt}}} f_{\text{tot}}(\lambda_m), \tag{9}$$

where $\sigma_{\max} = 0.3$ MPa is the peak isometric stress of skeletal muscle, $\lambda_{\text{opt}} = 1.4$ is the optimal fiber contraction ratio for force generation, and f_{tot} is the normalized force-length function for the passive and active components. We define f_{tot} in accordance with a standard Hill-type muscle model [Zajac 1988].

Both active (muscles) and passive (tendon, collagen, fat) components of flesh are primarily composed of water and, consequently, tissue deformation is largely incompressible. This is of particular importance in our model for reproducing muscle bulging behaviors.[4] Specifically, the volume conservation term in (6) is

$$\Psi_v(\mathbf{F}) = \frac{\kappa}{2}(J-1)^2, \tag{10}$$

with the *bulk modulus* κ set to 100 MPa in our model. As discussed by Patterson et al. [2012], this stiff energy term, which exceeds the stiffness of the nonvolumetric elastic tissue response by more than two orders of magnitude, could severely hinder an efficient numerical solution by slowing down the convergence of iterative equilibrium solvers. We follow the mixed formulation proposed in their work, rewriting the constitutive model (6) as

$$\hat{\Psi}(\mathbf{F}, p) = \Psi_0(\mathbf{F}) + \alpha p(J-1) - \frac{\alpha^2 p^2}{2\kappa}, \tag{11}$$

where $\Psi_0(\mathbf{F}) = \Psi_p(\mathbf{F}) + \sum_m d_m\Psi_m(\mathbf{F})$ is the *deviatoric* component of the strain energy excluding response due to volume change. This new strain energy introduces an auxiliary "pressure" variable p that in the limit of true incompressibility ($\kappa \to \infty$) becomes a Lagrange multiplier for the volume preservation constraint $J = 1$. The strain energy (11) has the remarkable property that a *saddle point* of $\hat{\Psi}$ can occur at a configuration $(\mathbf{x}^*, p^*)$ iff $\mathbf{x}^*$ is a critical point (typically, a stable minimum) of the strain energy (6). However, (11) can remain extremely well conditioned, even for highly incompressible materials, by tuning the free parameter α (whose user-specified value does not affect the location of its saddle point or the respective critical point of (6)). We found that a value of $\alpha \approx \sqrt{\kappa(\mu_{01} + \mu_{10})}/h$ leads to convergence behavior comparable to compressible materials, even when κ is set to the high value of

[3] The operators "det", "tr", and ":" denote the determinant, trace ($\operatorname{tr}\mathbf{A} = \sum_i A_{ii}$), and double contraction ($\mathbf{A} : \mathbf{B} = \sum_{i,j} A_{ij}B_{ij}$), respectively.

[4] Volume preservation in real human tissue is a rather "global" effect, since the displacement of blood volume and intercellular water is very much possible due to both pathological factors (e.g., swelling, circulatory anomalies) as well as mechanical means (external pressure, body posture, etc.). In our approach we do not aim to resolve such complex, often viscoelastic effects, and settle for a hyperelastic quasi-incompressible material response, where *local* volume preservation is enforced by means of a penalty term that discourages volume change.

100 MPa. Of course, we must employ an iterative method capable of solving indefinite systems (we use MINRES in our implementation), since the strain energy (11) is nonconvex by design (i.e., it has an indefinite Hessian). We refer the reader to Patterson et al. [2012] for the discretization and numerical solution details.

A.3 Skeletal Attachment Constraints

Similar to the approach of Lee et al. [2009], we employ elastic, zero rest-length springs to anchor the deformable flesh to attachment locations that are uniformly sampled on the surface of every bone; in this way, the regular lattice need not strictly conform to the exact surface shape of the skeletal bones. The soft nature of these constraints also provides a degree of tolerance against modeling or articulation inaccuracies that might lead to (limited) bone intersection during motion, or extreme nonphysical compression of flesh around joints in tight contact. Additionally, the attachment springs provide the mechanism by which external forces acting on the skin surface propagate to the underlying skeletal bones. Technically, each discrete attachment location contributes a term $\Psi^{(i)}$ to the (deviatoric) strain energy Ψ_0 in (11), defined as

$$\Psi^{(i)} = \frac{\beta^{(i)}}{2} \|\mathbf{W}^{(i)}\mathbf{x} - t^{(i)}\|_2^2, \tag{12}$$

where $\beta^{(i)}$ is the stiffness of the spring constraint (set proportional to the contact area attributed to this attachment point), $\mathbf{W}^{(i)}$ is a weighted embedding (typically a trilinear interpolation map from the lattice degrees of freedom) of a flesh anchor inside the deforming lattice, and $t^{(i)}$ is the respective kinematic target on the bone surface.

B. COMPARING ALTERNATIVE APPROACHES

Guided by biological and physical first principles, our framework has embraced:

(1) CPG-based locomotion control, since the CPG is a biologically principled low-level motor control mechanism;
(2) muscle-based actuation, as contractile muscles are the biologically principled skeletal actuation mechanism;
(3) volumetric soft-tissue simulation as the biologically principled fleshing approach; and
(4) detailed physical simulation of the environment, particularly Navier-Stokes simulation of water, and its interaction with the swimmer's body.

In this appendix, we report on experiments aimed at assessing the importance of these simulation/control components of our framework relative to more conventional approaches in computer animation.

B.1 CPG Control vs. Splines

Spline-based animation methods have traditionally been more familiar to graphics practitioners than CPG-based animation control. In fact, as discussed in Section 4.1, we initially use cubic B-splines to approximate the CPG training data. As a simple alternative to the CPG dynamical model, our continuous spline approximations may be repeated in time to produce periodic muscle signals to drive our virtual swimmer. The accompanying video includes a comparison of our CPG-controlled swimming against spline-controlled swimming. Although the results look qualitatively similar for any particular swimming stroke in steady state, the spline technique is noticeably choppier than the CPG technique due to discontinuities in the derivatives of the periodic spline functions across cycles, whereas the muscle control signals generated by our CPGs are always C^1-smooth. Moreover, to switch from one swimming stroke to another, the spline-based controller would have to transition carefully between numerous periodic spline functions, one per muscle. By contrast, our CPG muscle controllers can effect smooth transitions and control swimming speed by simply switching and/or modifying the values of a few parameters.

B.2 Muscle Actuation vs. Joint Torques

In human animation, joint-torque actuation methods have traditionally been more familiar to graphics practitioners than muscle-based actuation. Since skeletal muscle forces, through (bone) moment arms, eventually produce torques at rotational joints in the skeleton (but see Lee and Terzopoulos [2008]), we can in principle achieve similar animation results through equivalent joint-torque-driven simulation. The accompanying video includes a comparison of our muscle-actuated simulation against both inverse-dynamic (ID) and PD joint-torque-actuated simulation. In the case of swimming, we obtained plausible results using joint-torque actuation, but it was necessary to set high gains for the PD joint-torque controllers accompanied with an order-of-magnitude smaller numerical timestep compared to the muscle-based approach. Moreover, a further advantage of the latter is that modifying the parameters of contractile muscles situated in anatomically accurate positions is the natural way to create nuanced biological motion patterns, including pathological ones [Wang et al. 2012], as well as of naturally effecting realistic flesh deformations [Lee et al. 2009].

B.3 Flesh Simulation vs. Procedural Skinning

For fleshing human bodies, procedural skinning techniques have traditionally been more familiar to graphics practitioners than volumetric soft-tissue simulation. The accompanying video includes a comparison of our deformable flesh simulation against a state-of-the-art dual-quaternion skinning method [Kavan et al. 2008] with bounded biharmonic weights [Jacobson et al. 2011]. The volumetric flesh simulation and procedural skinning result in similar swimming performances. From some but not all viewpoints, the skin deformation appears plausible as the body articulates, but cannot adequately synthesize anatomically detailed deformations, such as the muscle bulging effects demonstrated in Figure 11.

B.4 Fluid Simulation vs. Velocity Fields

Procedural velocity field techniques have traditionally been easier for graphics practitioners to use than detailed physics-based fluid simulation (e.g., Tu and Terzopoulos [1994]). The accompanying video includes a comparison of our Navier-Stokes water simulation approach against the use of a static, zero-velocity water field, employing the same flesh-water force coupling method for both. With the same amount of muscle effort, the virtual swimmer swims significantly faster in the simulated fluid environment compared to the zero-velocity field. Moreover, fluid simulation provides realistic wave, splash, and other effects that are entirely absent with the latter.

B.5 A Comparison of Swimming Performances

Figure 12 presents a quantitative comparison of the performance of our virtual swimmer in the experimental scenarios described in the previous sections of this appendix, by plotting the distance traveled by the swimmer's pelvis over time. The table in Figure 13 indicates the associated average swimming speeds. In the figure and table, *CPG muscle control* refers to our experimental setting developed in

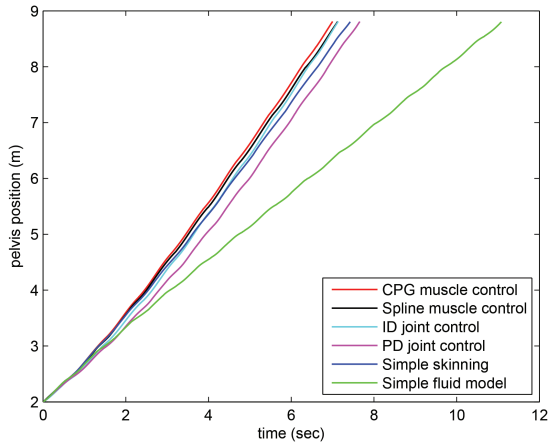

Fig. 12. Swimming performance in the experimental scenarios.

Experiment	Average Swimming Speed (m/s)
CPG muscle control	0.973
Spline muscle control	0.956
ID joint control	0.954
PD joint control	0.889
Simple skinning	0.916
Simple fluid model	0.614

Fig. 13. The virtual swimmer's average speed in the experimental scenarios.

the main text of this article, that is, using CPG locomotion control to synthesize muscle-length signals for the muscle-driven biomechanical body simulation with simulated flesh situated in simulated water. Under this same simulation scenario, *spline muscle control* refers to the use of B-splines to synthesize the muscle-length signals (Section B.1), *ID joint control* refers to inverse-dynamics controlled joint-torque-driven simulation (Section B.2), and *PD joint control* refers to PD-controlled joint-torque driven simulation (Section B.2). *Simple skinning* refers to using the dual-quaternion skinning approach (Section B.3). *Simple fluid model* refers to using a zero-velocity field (Section B.4).

The figure and table reveal that the virtual swimmer swims most efficiently in our original experimental scenario. The swimmer can achieve similar swimming performances in the other experimental settings, except when the simulated fluid model is replaced by a zero-velocity field, which results in significantly lower efficiency.

ACKNOWLEDGMENTS

The review summary from the SIGGRAPH program committee prompted the addition of Appendix A and Appendix B with the experiments reported therein.

REFERENCES

P. Agache, C. Monneur, J. Leveque, and J. Rigal. 1980. Mechanical properties and Young's modulus of human skin in vivo. *Archiv. Dermatol. Res.* 269, 3, 221–232.

I. Albrecht, J. Haber, and H.-P. Seidel. 2003. Construction and animation of anatomically based human hand models. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA'03).* 98–109.

D. J. Benson. 1992. Computational methods in Lagrangian and Eulerian hydrocodes. *Comput. Methods Appl. Mech. Engin.* 99, 2–3, 235–394.

S. S. Blemker and S. L. Delp. 2005. Three-dimensional representation of complex muscle architectures and geometries. *Ann. Biomed. Engin.* 33, 5, 661–673.

J. Bonet and R. Wood. 1997. *Nonlinear Continuum Mechanics for Finite Element Analysis*. Cambridge University Press, Cambridge, UK.

F. Dong, G. J. Clapworthy, M. A. Krokos, and J. Yao. 2002. An anatomy-based approach to human muscle modeling and deformation. *IEEE Trans. Visual. Comput. Graph.* 8, 2, 154–170.

D. Enright, S. Marschner, and R. Fedkiw. 2002. Animation and rendering of complex water surfaces. *ACM Trans. Graph.* 21, 3, 736–744.

P. Faloutsos, M. Van De Panne, and D. Terzopoulos. 2001. Composable controllers for physics-based character animation. In *Proceedings of the 28th Annual ACM Conference on Computer Graphics and Interactive Techniques (SIGGRAPH'01).* 251–260.

R. Featherstone. 1987. *Robot Dynamics Algorithms*. Kluwer Academic, Boston.

R. P. Fedkiw. 2002. Coupling an eulerian fluid calculation to a lagrangian solid calculation with the ghost fluid method. *J. Comput. Phys.* 175, 1, 200–224.

N. Foster and R. Fedkiw. 2001. Practical animation of liquids. In *Proceedings of the 28th Annual ACM Conference on Computer Graphics and Interactive Techniques (SIGGRAPH'01).* 23–30.

A. Gams, A. J. Ijspeert, S. Schaal, and J. Lenarcic. 2009. On-line learning and modulation of periodic movements with nonlinear dynamical systems. *Auton. Robots* 27, 1, 3–23.

R. Grzeszczuk and D. Terzopoulos. 1995. Automated learning of muscle-actuated locomotion through control abstraction. In *Proceedings of the 22nd Annual ACM Conference on Computer Graphics and Interactive Techniques (SIGGRAPH'95).* 63–70.

K. Hase, K. Miyashita, S. Ok, and Y. Arakawa. 2003. Human gait simulation with a neuromusculoskeletal model and evolutionary computation. *J. Vis. Comput. Anim.* 14, 2, 73–92.

J. K. Hodgins, W. L. Wooten, D. C. Brogan, and J. F. O'Brien. 1995. Animating human athletics. In *Proceedings of the 22nd Annual ACM Conference on Computer Graphics and Interactive Techniques (SIGGRAPH'95).* 71–78.

A. J. Ijspeert. 2008. Central pattern generators for locomotion control in animals and robots: A review. *Neural Netw.* 21, 4, 642–653.

A. J. Ijspeert, A. Crespi, D. Ryczko, and J.-M. Cabelguen. 2007. From swimming to walking with a salamander robot driven by a spinal cord model. *Sci.* 315, 5817, 1416–1420.

A. Jacobson, I. Baran, J. Popovic, and O. Sorkine. 2011. Bounded biharmonic weights for real-time deformation. *ACM Trans. Graph.* 30, 4, 78:1–8.

L. Kavan, S. Collins, J. Zara, and C. O'Sullivan. 2008. Geometric skinning with approximate dual quaternion blending. *ACM Trans. Graph.* 27, 4, 105.

T. Komura, Y. Shinagawa, and T. L. Kunii. 2000. Creating and retargetting motion by the musculoskeletal human body model. *Vis. Comput.* 16, 5, 254–270.

N. Kwatra, C. Wojtan, M. Carlson, I. Essa, P. J. Mucha, and G. Turk. 2010. Fluid simulation with articulated bodies. *IEEE Trans. Visual. Comput. Graph.* 16, 70–80.

S.-H. Lee, E. Sifakis, and D. Terzopoulos. 2009. Comprehensive biomechanical modeling and simulation of the upper body. *ACM Trans. Graph.* 28, 4, 99:1–17.

S.-H. Lee and D. Terzopoulos. 2006. Heads up! Biomechanical modeling and neuromuscular control of the neck. *ACM Trans. Graph.* 25, 3, 1188–1198.

S.-H. Lee and D. Terzopoulos. 2008. Spline joints for multibody dynamics. *ACM Trans. Graph.* 27, 3, 22:1–8.

Y. Lee, D. Terzopoulos, and K. Waters. 1995. Realistic modeling for facial animation. In *Proceedings of the 22nd Annual ACM Conference on Computer Graphics and Interactive Techniques (SIGGRAPH'95).* 55–62.

M. Lentine, J. T. Gretarsson, C. Schroeder, A. Robinson-Mosher, and R. Fedkiw. 2011. Creature control in a fluid environment. *IEEE Trans. Visual. Comput. Graph.* 17, 5, 682–693.

M. MacKay-Lyons. 2002. Central pattern generation of locomotion: A review of the evidence. *Phys. Therapy* 82, 1, 69–83.

A. McAdams, Y. Zhu, A. Selle, M. Empey, R. Tamstorf, J. Teran, and E. Sifakis. 2011. Efficient elasticity for character skinning with contact and collisions. *ACM Trans. Graph.* 30, 4, 37:1–12.

S. Mihalef, M. Kadioglu, D. Sussman, V. Metaxas, and V. Hurmusiadis. 2008. Interaction of multiphase flow with animated models. *Graph. Models* 70, 3, 33–42.

Y. Nakamura, K. Yamane, Y. Fujita, and I. Suzuki. 2005. Somatosensory computation for man machine interface from motion-capture data and musculoskeletal human model. *IEEE Trans. Robot.* 21, 1, 58–66.

T. Patterson, N. Mitchell, and E. Sifakis. 2012. Simulation of complex nonlinear elastic bodies using lattice deformers. *ACM Trans. Graph.* 31, 6, 197:1–10.

L. Righetti and A. J. Ijspeert. 2006. Programmable central pattern generators: An application to biped locomotion control. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA'06).* 1585–1590.

S. Schaal and C. G. Atkeson. 1997. Constructive incremental learning from only local information. *Neural Comput.* 10, 2047–2084.

W. Si. 2013. Realistic simulation and control of human swimming and underwater movement. PhD thesis, Computer Science Department, University of California, Los Angeles. http://www.cs.ucla.edu/∼dt/theses/si-thesis.pdf.

S. Sueda, A. Kaufman, and D. K. Pai. 2008. Musculotendon simulation for hand animation. *ACM Trans. Graph.* 27, 3, 83:1–8.

G. Taga. 1995. A model of the neuro-musculo-skeletal system for human locomotion. *Biol. Cybernet.* 73, 2, 113–121.

J. Tan, Y. Gu, G. Turk, and C. K. Liu. 2011. Articulated swimming creatures. *ACM Trans. Graph.* 30, 4, 58:1–12.

W. Tsang, K. Singh, and E. Fiume. 2005. Helping hand: An anatomically accurate inverse dynamics solution for unconstrained hand motion. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA'05).* 319–328.

X. Tu and D. Terzopoulos. 1994. Artificial fishes: Physics, locomotion, perception, behavior. In *Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH'94).* 43–50.

Virtual-Swim. 2007. Aquatic animation for analysis and education. http://www.virtual-swim.com/.

J. M. Wang, S. R. Hamner, S. L. Delp, and V. Koltun. 2012. Optimizing locomotion controllers using biologically-based actuators and objectives. *ACM Trans. Graph.* 31, 4, 25:1–11.

P.-F. Yang, J. Laszlo, and K. Singh. 2004. Layered dynamic control for interactive character swimming. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA'04).* 39–47.

F. E. Zajac. 1988. Muscle and tendon: Properties, models, scaling, and application to biomechanics and motor control. *Critical Rev. Biomed. Engin.* 17, 4, 359–411.

Y. Zhu, E. Sifakis, J. Teran, and A. Brandt. 2010. An efficient multigrid method for the simulation of high-resolution elastic solids. *ACM Trans. Graph.* 29, 2, 16:1–18.

V. B. Zordan, B. Celly, B. Chiu, and P. C. Dilorenzo. 2006. Breathe easy: Model and control of human respiration for computer animation. *Graph. Models* 68, 113–132.

Received March 2014; accepted May 2014