Deep learning of biomimetic sensorimotor control for biomechanical human animation

Article *in* ACM Transactions on Graphics · July 2018

DOI: 10.1145/3197517.3201305

5 authors, including:

Masaki Nakada
University of California, Los Angeles
9 PUBLICATIONS **13** CITATIONS
SEE PROFILE

Tao Zhou
UCLA
21 PUBLICATIONS **117** CITATIONS
SEE PROFILE

Deep Learning of Biomimetic Sensorimotor Control for Biomechanical Human Animation

MASAKI NAKADA, TAO ZHOU, HONGLIN CHEN, TOMER WEISS, and DEMETRI TERZOPOULOS,
University of California, Los Angeles, USA

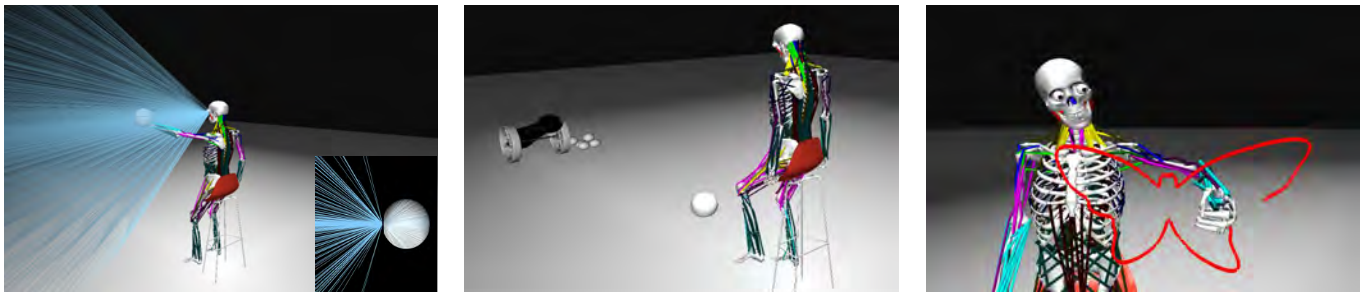


Fig. 1. Our anatomically accurate biomechanical human model features a biomimetic sensorimotor system that incorporates 20 trained deep neural networks, 10 devoted to visual perception and 10 to motor control. Its motor subsystem comprises 5 neuromuscular controllers that actuate the cervicocephalic and four limb musculoskeletal complexes. Via ray tracing (blue rays), its virtual eyes (b) visually sample the environment (a) to compute the irradiance at RGB photoreceptors nonuniformly distributed on their foveated retinas. Its sensory subsystem performs visual analysis that drives the motor subsystem to actuate appropriate eye, head, and limb movements. Our virtual human can visually foveate and track a moving ball, and intercept it with arms (a) and legs (c), as well as write and draw with its finger (d). The rendered appearance of the numerous Hill-type skeletal muscles is unimportant in our work; they are depicted as lines, color coded to distinguish the different muscle groups, with brightness proportional to each contractile muscle's efferent neural activation.

We introduce a biomimetic framework for human sensorimotor control, which features a biomechanically simulated human musculoskeletal model actuated by numerous muscles, with eyes whose retinas have nonuniformly distributed photoreceptors. The virtual human's sensorimotor control system comprises 20 trained deep neural networks (DNNs), half constituting the neuromuscular motor subsystem, while the other half compose the visual sensory subsystem. Directly from the photoreceptor responses, 2 vision DNNs drive eye and head movements, while 8 vision DNNs extract visual information required to direct arm and leg actions. Ten DNNs achieve neuromuscular control—2 DNNs control the 216 neck muscles that actuate the cervicocephalic musculoskeletal complex to produce natural head movements, and 2 DNNs control each limb; i.e., the 29 muscles of each arm and 39 muscles of each leg. By synthesizing its own training data, our virtual human automatically learns efficient, online, active visuomotor control of its eyes, head, and limbs in order to perform nontrivial tasks involving the foveation and visual pursuit of target objects coupled with visually-guided limb-reaching actions to intercept the moving targets, as well as to carry out drawing and writing tasks.

Authors' address: Masaki Nakada, nakada@cs.ucla.edu; Tao Zhou, taozhou@cs.ucla.edu; Honglin Chen, chenhonglin@g.ucla.edu; Tomer Weiss, tweiss@cs.ucla.edu; Demetri Terzopoulos, dt@cs.ucla.edu, Computer Science Department, University of California, Los Angeles, Los Angeles, CA, 90095, USA.

0730-0301/2018/8-ART56 $15.00
https://doi.org/10.1145/3197517.3201305

CCS Concepts: • **Computing methodologies** → **Computer graphics**; **Animation**; **Bio-inspired approaches**; **Neural networks**; **Artificial life**; **Physical simulation**;

Additional Key Words and Phrases: Biomechanical human animation; Sensorimotor control; Biomimetic perception; Deep neural network learning.

ACM Reference Format:
Masaki Nakada, Tao Zhou, Honglin Chen, Tomer Weiss, and Demetri Terzopoulos. 2018. Deep Learning of Biomimetic Sensorimotor Control for Biomechanical Human Animation. *ACM Trans. Graph.* 37, 4, Article 56 (August 2018), 15 pages. https://doi.org/10.1145/3197517.3201305

1 INTRODUCTION

The modeling of graphical characters based on human anatomy is attracting much attention in computer animation. Increasingly accurate biomechanical modeling of the human body should, in principle, yield more realistic human animation. However, it remains a daunting challenge to control complex biomechanical human musculoskeletal models with over two hundred bones and on the order of a thousand contractile muscle actuators. This paper reports on a major stride toward achieving this goal. We take a neuromuscular control approach based on machine learning using Deep Neural Networks (DNNs). Beyond pure motor control, we confront the difficult, less explored problem of sensorimotor control, in which processed sensory information acquired through active perception drives appropriate motor responses that support natural behavior.

More specifically, the main research problems that we tackle in this paper are as follows:

- **Biomimetic motor control:** The human nervous system generates the required muscle activations seemingly instantaneously, and biological motor control mechanisms appear incompatible with any online control approach that would rely on solving inverse kinematics, inverse dynamics, or minimal effort optimization problems in order to compute the required muscle activations needed to produce desired movements. Rather, the brain is capable of learning muscle controls from experience. Therefore, we explore a neuromuscular control approach based on machine learning, particularly DNNs and deep learning.
- **Biomimetic sensory control:** Since our objective is realistic virtual humans, we revisit the fundamentals of human vision, taking a strongly biomimetic approach to computer vision for virtual humans. We create an anatomically consistent 3D eye model, which is capable of eye movements, with a retina populated by photoreceptors in a naturally nonuniform, foveated pattern. Furthermore, we pursue a visual DNN approach to producing natural eye movements and to inferring basic visual information from the retinal photoreceptor responses.
- **Biomimetic sensorimotor control:** We combine the above motor and sensory control models to achieve an integrated, biomimetic sensorimotor control system capable of performing several nontrivial tasks. These include not only well-coordinated eye-head movements characteristic of natural active vision, but also natural limb motions to reach static and kinetic visual targets as well as to write and draw by hand.

Our overarching contribution is a novel biomimetic framework for human sensorimotor control, which is illustrated in Fig. 1. The human sensorimotor control model that we develop is unprecedented, both in its use of a sophisticated biomechanical human simulation incorporating numerous contractile muscle actuators, as well as in its use of modern machine learning methodologies to achieve online motor control of realistic musculoskeletal systems and perform online visual processing for active, foveated perception. We accomplish this using a modular set of DNNs, all of which are automatically trained offline with training data synthesized by the biomechanical model itself, a strategy that we regard a contribution of interest to the machine learning research community as well.

The remainder of the paper is organized as follows: Section 2 reviews relevant prior work. Section 3 reviews our biomechanical human musculoskeletal model. Section 4 overviews our novel sensorimotor control framework and system for this model. Section 5 develops the motor subsystem and Section 6 develops the sensory (currently purely visual) subsystem, while Section 7 describes the sensorimotor simulation cycle. Section 8 presents our experiments and results with the virtual human model. Section 9 concludes the paper with a summary of our work and current plans for future work.

2 RELATED WORK

The following two sections review prior work relevant to the primary topics of this paper—neuromuscular and sensorimotor control of an anatomically accurate biomechanical human musculoskeletal model.

2.1 Neuromuscular and Visuomotor Control

With their pioneering "NeuroAnimator", Grzeszczuk et al. [1998] were the first to apply artificial neural networks and backpropagation learning in computer graphics, demonstrating the learning of neural network emulators of physics-based models, as well as how these emulators can support fast reinforcement learning of motion controllers based on backpropagation through time, a policy gradient method utilizing recursive neural networks. Lee and Terzopoulos [2006] were the first to employ neural networks as neuromuscular controllers, specifically for their detailed biomechanical model of the human cervicocephalic system. Their work demonstrated that neuromuscular learning approaches are the most biomimetic and most promising in tackling the high-dimensional problems of muscle-actuated biomechanical motor control. Unfortunately, the shallow neural networks that could practically be trained at that time proved inadequate at coping with the higher dimensionality and greater quantities of training data required to learn to control full-body musculoskeletal models. To tackle a more challenging generalization of the cervicocephalic control problem, Nakada and Terzopoulos [2015] introduced the use of deep learning techniques [Goodfellow et al. 2016].

The work reported in the present paper was inspired not just by the aforementioned efforts, but also by the impressive visuomotor "EyeCatch" model of Yeo et al. [2012]. EyeCatch is a non-learning-based visuomotor system in a simple, non-physics-based humanoid character. By contrast, we demonstrate dramatically more complex sensorimotor control realized using a set of trained DNNs in a comprehensive, anatomically-accurate, muscle-actuated biomechanical human model.

Furthermore, the EyeCatch model and the more primitive visuomotor model of Lee and Terzopoulos [2006] made direct use of the 3D spatial positions of virtual visual targets, without any significant biologically-inspired visual processing. By contrast, we build upon the pioneering "Animat Vision" work on foveated, active computer vision for animated characters [Terzopoulos and Rabie 1995], which demonstrated vision-guided bipedal locomotion and navigation albeit in purely kinematic human characters [Rabie and Terzopoulos 2000]. We introduce a substantially more biomimetic active vision model based on the foveated pattern of cone photoreceptor placement in biological retinas [Schwartz 1977].

2.2 Biomechanical Human Modeling and Animation

Human modeling and animation has been a topic of interest since the early days of computer graphics. As the literature is now vast, we focus here on the most relevant work that precedes our accomplishment of neurosensorimotor control of a detailed biomechanical human musculoskeletal model.

Aside from the domain of biomechanical human facial animation, in which muscle actuators have been employed for decades with increasingly realistic results, e.g., [Terzopoulos and Waters 1990], [Lee et al. 1995], [Kähler et al. 2002], [Sifakis et al. 2005], [Ichim et al. 2017], the simulation and control of articulated anthropomorphic

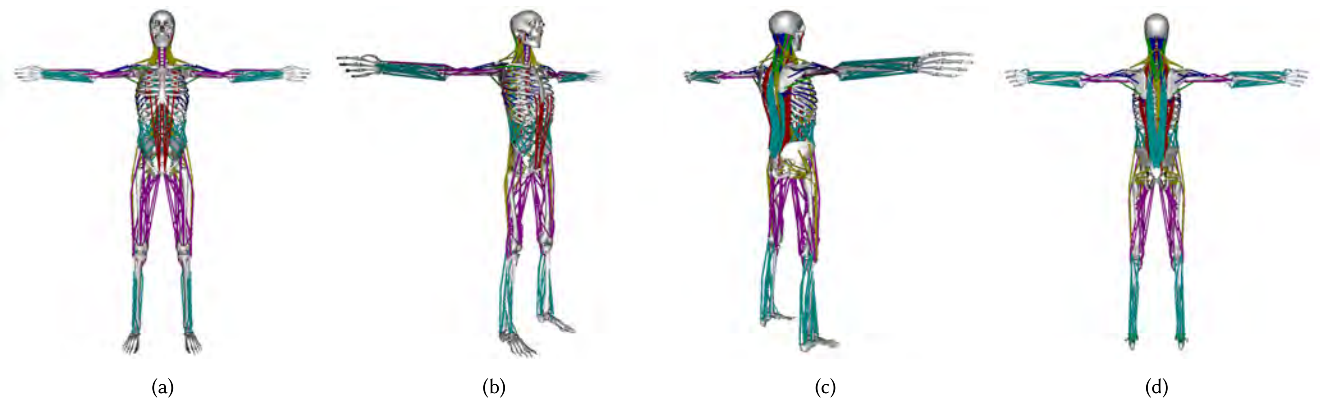


Fig. 2. The biomechanical human musculoskeletal model, showing the skeletal system with its 193 bones and 823 Hill-type muscle actuators.

models has been heavily researched, although mostly via "stick figures" actuated by joint torque motors, which was the state-of-the-art in physics-based character animation two decades ago; e.g., [Hodgins et al. 1995], [Faloutsos et al. 2001]. Unfortunately, these are poor models of the human musculoskeletal system that inadequately address the mystery of its highly robust neuromuscular control.

Anatomically and biomechanically accurate musculoskeletal simulation—early examples of which are the cervicocephalic model of Lee and Terzopoulos [2006] and hand model of Sueda et al. [2008] (see also [Sachdeva et al. 2015])—is now rapidly superseding the earlier modeling mindset, having finally overcome the reluctance of the animation research community to embrace the daunting complexities of strongly biomimetic approaches. Researchers have been pursuing the construction of detailed musculoskeletal models from noninvasive 3D medical images [Fan et al. 2014], body surface scans [Kadleček et al. 2016], and other modalities. Controlling musculoskeletal models remains a challenging problem, but recent papers have reported progress on the important problem of locomotion synthesis [Geijtenbeek et al. 2013; Lee et al. 2014; Wang et al. 2012], most recently by adopting machine-learning-based control methods [Holden et al. 2017; Liu and Hodgins 2017; Peng et al. 2017]. Cruz Ruiz et al. [2017] present a comprehensive survey on muscle-based control for character animation.

Most pertinent to our present work, Lee et al. [2009] created a comprehensive biomechanical human musculoskeletal model, and the full-body version of this model was employed by Si et al. [2014] to animate human swimming in simulated water with a neural CPG control system that synthesizes sustained rhythmic muscle activations suitable for locomotion. We utilize the same biomechanical model, but instead tackle the important open problem of transient, task-based sensorimotor control, particularly for observing and reaching out to stationary or moving visual targets (through an entirely different approach than in earlier work [Huang et al. 2010]).

3 BIOMECHANICAL MUSCULOSKELETAL MODEL

Fig. 2 shows the musculoskeletal system of our anatomically accurate biomechanical human model. It includes all of the relevant articular bones and muscles—193 bones (hand and foot bones included) plus a total of 823 muscle actuators. (To mitigate computational cost, we exclude the finite element musculotendinous soft-tissue simulation that produces realistic flesh deformations.) Each skeletal muscle is modeled as a Hill-type uniaxial contractile actuator that applies forces to the bones at its points of insertion and attachment. The human model is numerically simulated as a force-driven articulated multi-body system; refer to [Lee et al. 2009] for the details.

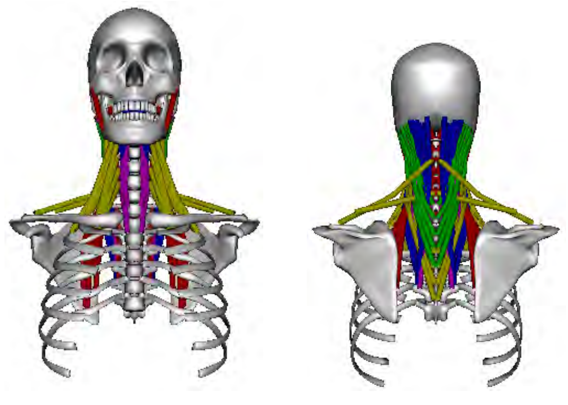
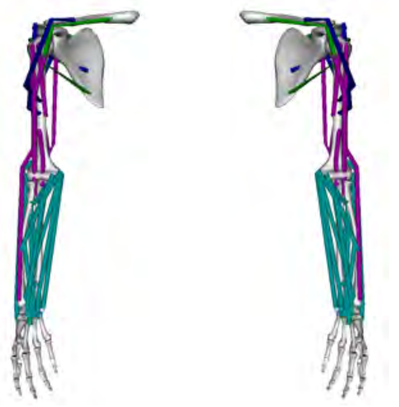


Fig. 3. The biomechanical cervicocephalic musculoskeletal model showing the 216 neck muscles, for some of which the displayed vertebra below C1 plus the ribs, clavicles, and scapulae serve as root attachments.

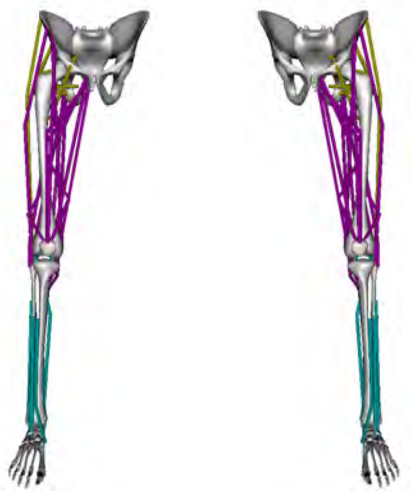
Given our human model, the challenge in neuromuscular motor control is to generate efferent activation signals for its muscles necessary to carry out various motor tasks. For the purposes of our research to date, we mitigate complexity by placing our virtual human in a seated position, immobilizing the pelvis as well as the lumbar and thoracic spinal column vertebra and other bones of the torso, leaving the cervical column, arms, and legs free to articulate.

Cervicocephalic Musculoskeletal Complex: Fig. 3 shows in greater detail the neck-head musculoskeletal complex, which is rooted at the thoracic vertebra (T1), with its seven vertabrae, C7 through C1 (atlas), progressing up the cervical spine to the skull, which is an "end-effector" of substantial mass. A total of 216 short, intermediate,



(a) Right arm (b) Left arm

Fig. 4. The musculoskeletal anatomy of the arms, showing the 29 arm muscles, for some of which the clavicle and scapula serve as root attachments.



(a) Right leg (b) Left leg

Fig. 5. The musculoskeletal anatomy of the legs, showing the 39 leg muscles, for some of which the pelvis serves as a root attachment.

and long Hill-type uniaxial muscle actuators arranged in deep, intermediate, and superficial layers, respectively, actuate the seven 3-degree-of-freedom joints of the cervicocephalic musculoskeletal complex. Each cervical joint incorporates damped rotational springs to approximate the passive elasticity of the intervertebral discs and ligaments.

Arm Musculoskeletal Complexes: Fig. 4 shows the musculoskeletal model of the bilaterally symmetric arms, which are rooted at the clavicle and scapula, and include the humerus, ulnar, and radius bones, plus the bones of the wrist and hand. In each arm, a total of 29 Hill-type uniaxial muscles actuate the shoulder, elbow, and wrist joints. To mitigate complexity, the bones of the hand are actuated kinematically.

Leg Musculoskeletal Complexes: Fig. 5 shows the musculoskeletal model of the bilaterally symmetric legs, which are rooted at the pelvis and include the femur, patella, tibia, and fibula bones, plus the bones of the ankle and foot. In each leg, a total of 39 Hill-type uniaxial muscles actuate the hip, knee, and ankle joints. To mitigate complexity, the bones of the foot are actuated kinematically.

4 SENSORIMOTOR CONTROL FRAMEWORK

Thus, of the 823 muscle actuators in the complete musculoskeletal model, $216 + (2 \times 29) + (2 \times 39) = 352$ muscle actuators must be controlled by specifying for each of them a time-varying, efferent activation signal $a(t)$. In our sensorimotor control framework, all of these activation signals are generated by trained DNNs.

Fig. 6 presents an overview of the sensorimotor system, showing its sensory and motor subsystems. The figure caption describes the information flow and the functions of its 20 DNNs.

The two subsequent sections develop first the motor subsystem, which includes its 5 neuromuscular motor controllers (each comprising 2 DNNs) (Fig. 6f,gL,gR,hL,hR), then the sensory subsystem, which includes the eyes (Fig. 6eL,eR) and their retinas (Fig. 6bL,bR), along with 10 vision DNNs (Fig. 6cL,cR,dL,dR).

5 MOTOR SUBSYSTEM

The cervicocephalic and limb musculoskeletal complexes of our biomechanical human model are actuated by groups of skeletal muscles driven by neuromuscular motor controllers that generate efferent activation signals for the muscles. The motor subsystem includes a total of 5 neuromuscular motor controllers, one for the cervicocephalic complex, two for the arm complexes, and two for the leg complexes. Each generates the muscle activations $\boldsymbol{a}$ to actuate its associated musculoskeletal complex in a controlled manner.

Fig. 7 shows the neuromuscular motor controller architecture, which is comprised of a voluntary controller and a reflex controller (c.f. [Lee and Terzopoulos 2006]), both of which are trained DNNs that produce muscle activation adjustment (Δ) signals. The voluntary controller produces signals $\Delta\boldsymbol{a}_v$, which induce the desired actuation of the associated musculoskeletal complex, while the reflex controller produces muscle-stabilizing signals $\Delta\boldsymbol{a}_r$. Thus, the output of the neuromuscular motor controller is given by

$$\boldsymbol{a}(t + \Delta t) = \boldsymbol{a}(t) + (\Delta\boldsymbol{a}_v(t) + \Delta\boldsymbol{a}_r(t)) . \tag{1}$$

In Fig. 7, note that the muscle activation signal $\boldsymbol{a}$ feedback loop makes the neuromuscular controllers recurrent neural networks.

Through systematic experiments, reported in Appendix A, we identified a DNN architecture that works well for all 10 motor DNNs; specifically, rectangularly-shaped, fully-connected networks with six 300-unit-wide hidden layers.[1]

[1] The DNNs employ rectified linear units (i.e., the ReLU activation function). The total number of weights in the DNNs ranges from 11.5–24.2 million. The initial weights are sampled from the zero-mean normal distribution with standard deviation $\sqrt{2/fan\_in}$, where *fan_in* is the number of input units in the weight tensor [He et al. 2015]. To train the DNNs, we apply the mean-squared-error loss function and the Adaptive Moment Estimation (Adam) stochastic optimizer [Kingma and Ba 2014] with learning rate $\eta = 10^{-6}$, step size $\alpha = 10^{-3}$, forgetting factors $\beta_1 = 0.9$ for the gradients and $\beta_2 = 0.999$ for their second moments, and avoid overfitting using the early stopping condition of negligible improvement for 10 successive epochs. Each epoch requires less than 10 seconds of computation time.

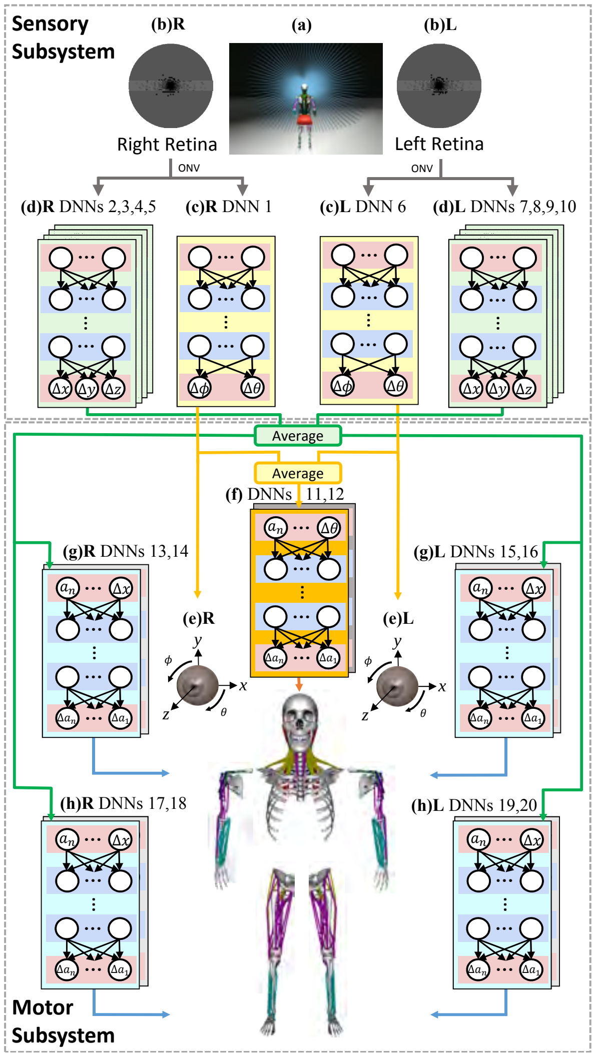


Fig. 6. Sensorimotor system architecture, showing the modular neural network controllers in the sensory subsystem (top) and motor subsystem (bottom), including a total of 20 Deep Neural Networks (DNNs), numbered 1–20, of four types (colored yellow, green, orange, and blue).
• Sensory Subsystem: (a) Each retinal photoreceptor casts a ray into the virtual world to compute the irradiance captured by the photoreceptor. (b) The arrangement of the 3,600 photoreceptors (black dots) on the left (b)L and right (b)R foveated retinas. Each retina outputs a 10,800-dimensional Optic Nerve Vector (ONV). There are 10 vision DNNs. The two (yellow) foveation DNNs (c) (1,6) input the ONV and output left-eye (c)L and right-eye (c)R angular discrepancies that drive the movements of the eyes (e) to foveate visual targets. (d) The other eight (green) vision DNNs—(d)L (7,8,9,10) for the left eye and (d)R (2,3,4,5) for the right eye—also input the ONV and output the limb-to-target visual discrepancy estimates.
• Motor Subsystem: There are 10 motor DNNs. The (orange) cervicocephalic neuromuscular motor controller (f) (DNNs 11,12) inputs the average of the left (c)L and right (c)R foveation DNN responses and outputs activations to the neck muscle group. The four (blue) limb neuromuscular motor controllers (g),(h) (DNNs 13–20) of the limb musculoskeletal complexes input the average of the left (d)L and right (d)R limb vision DNN responses and output activations to the respective arm and leg muscle groups.

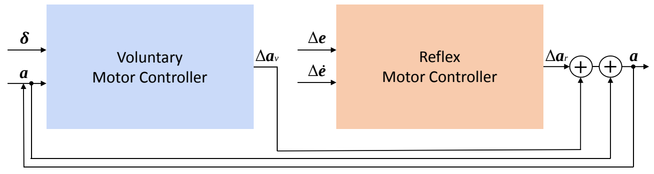


Fig. 7. Neuromuscular motor controller architecture. The voluntary controller inputs a target discrepancy $\boldsymbol{\delta}$ and, recurrently, the muscle activations $\boldsymbol{a}$. The reflex controller inputs the changes in muscle strains $\boldsymbol{e}$ and strain rates $\dot{\boldsymbol{e}}$. Both controllers output muscle activation adjustments.

In the next two sections, we present the details of the 5 voluntary motor DNNs and the 5 reflex motor DNNs, and describe the training data synthesis regimens for each type of network.

5.1 Voluntary Motor DNNs (11,13,15,17,19)[2]

The function of the cervicocephalic voluntary motor DNN (Fig. 8) is to generate efferent activation signals to the neck muscles in order to balance the mass of the head in gravity atop the flexible cervical column while actuating realistic head movements to achieve target head poses. The function of the 4 limb voluntary motor DNNs (Fig. 9), is to generate efferent activation signals to the muscles of the four limbs in order to execute controlled arm and leg movements, such as extending a limb to reach a target.

The architecture of all five voluntary motor DNNs is identical, except for the sizes of the input and output layers. Their input layers include units that represent the (angular or linear) components of the discrepancy ($\boldsymbol{\delta}$ in Fig. 7) between the value of some relevant feature of the virtual human's state, such as head orientation or hand/foot position, and the target value of that feature, as well as units that represent the current activations, $a_i(t)$, for $1 \leq i \leq n$, of each of the n muscles in the associated musculoskeletal complex. The 6 hidden layers have 300 units each. The output layer consists of units that encode the adjustments $\Delta a_i(t)$, for $1 \leq i \leq n$, to the muscle activations, which then contribute additively as $\boldsymbol{a}_v$ to updating the muscle activations according to Equation 1.

The following sections provide additional details about each of the DNNs and how they are trained. Appendix B explains in more detail how we use the biomechanical musculoskeletal model itself to synthesize training data.

5.1.1 *Cervicocephalic Voluntary Motor DNN (11).*

As shown in Fig. 8, the input layer of the cervicocephalic voluntary motor DNN has 218 units, comprising 2 units for the target head angular discrepancies, $\Delta\theta$ and $\Delta\phi$, and 216 units for the neck muscle activations a_i, for $1 \leq i \leq 216$. The output layer has 216 units that provide the muscle activation adjustments Δa_i.

To train the DNN, we use our biomechanical human musculoskeletal model to synthesize training data, as follows: Specifying a target orientation for the head yields angular discrepancies, $\Delta\theta$ and $\Delta\phi$, between the current head orientation and the target head orientation. With these angular discrepancies serving as the target control input, we compute inverse kinematics followed by inverse dynamics with minimal muscle effort optimization applied to the biomechanical

[2]The numbers in parentheses here and elsewhere refer to the numbered DNNs in Fig. 6.

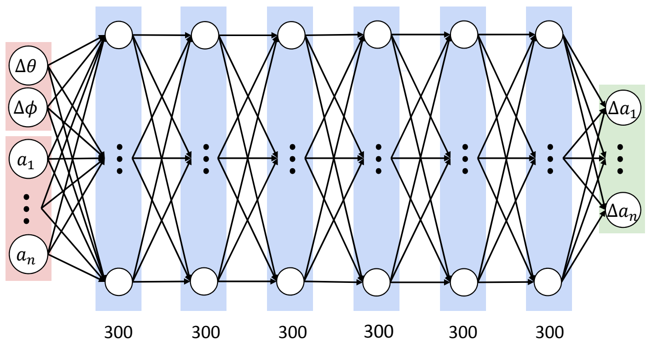


Fig. 8. Voluntary motor DNN for the cervicocephalic complex.

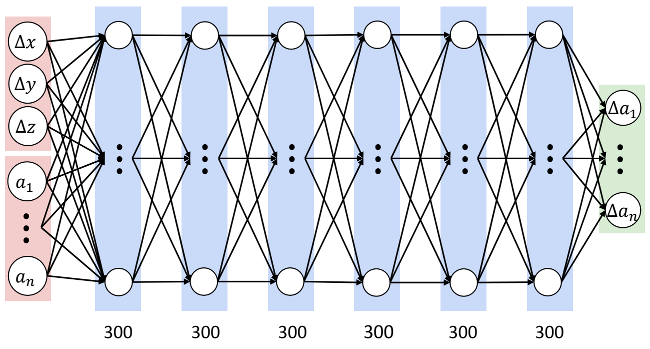


Fig. 9. Voluntary motor DNN for a limb (arm or leg) complex.

cervicocephalic model, as described in Appendix B. This determines muscle activation adjustments, which serve as the desired output of the cervicocephalic voluntary motor DNN. Hence, the input/output training pair consists of an input comprising the concatenation of the desired angle discrepancies, $\Delta\theta$ and $\Delta\phi$, and the current muscle activations a_i, for $1 \leq i \leq 216$, along with an associated output comprising the desired muscle activation adjustments Δa_i, for $1 \leq i \leq 216$.

This *offline* training data synthesis process takes 0.8 seconds of computational time to solve for 0.02 simulation seconds on a 2.4 GHz Intel Core i5 CPU with 8GB of RAM. Repeatedly specifying random target orientations and actuating the cervicocephalic musculoskeletal subsystem to achieve them, we generated a training dataset of 1M (million) input-output pairs.

The backpropagation DNN training process converged to a small error in 2,000 epochs. Fig. 10 graphs the progress of the training process. After the DNN is trained, it serves as the *online* cervicocephalic voluntary motor controller.

5.1.2 Limb Voluntary Motor DNNs (13,15,17,19). As shown in Fig. 9, the input layer of the limb voluntary motor DNN has 3 units that represent Δx, Δy, and Δz, the estimated linear discrepancies between the position of the end effector and a specified target position, as well as units that represent the current activations of the 26 arm muscles, a_i, for $1 \leq i \leq 26$, or of the 36 leg muscles, a_i, for $1 \leq i \leq 36$. The output layer consists of units that encode the muscle activation adjustments Δa_i.

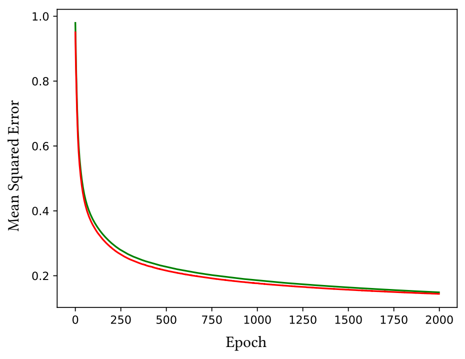


Fig. 10. Progress of the backpropagation training of the cervicocephalic voluntary motor DNN on the training (green) and validation (red) datasets.

To train the DNN, we employ our biomechanical human musculoskeletal model to synthesize training data as was described in the previous section: We specify a target position, which determines the linear discrepancies, Δx, Δy, and Δz, between the position of the end effector (hand or foot) and the target. Given these discrepancies and the current muscle activations as the input, the desired output of the network is the muscle activation adjustments, which are similarly computed by solving appropriate inverse kinematics followed by inverse dynamics and minimal muscle effort optimization problems for the limb musculoskeletal complex (see Appendix B). Repeatedly specifying a series of random target positions and actuating the limb to reach them, we generated a large training dataset of 1M input-output pairs. Four such training datasets were generated, one for each limb. After the DNNs are trained, they serve *online* as limb voluntary motor controllers.

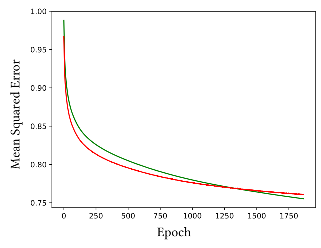
Fig. 11a graphs the progress of the backpropagation training process for the left arm voluntary motor DNN (15). For its training dataset, it converged to a small error after 1,865 epochs before triggering the early stopping condition to avoid overfitting. Fig. 11b graphs the progress of the backpropagation training process for the right arm voluntary motor DNN (13). It converged to a small error after 1,777 epochs before triggering the early stopping condition to avoid overfitting.

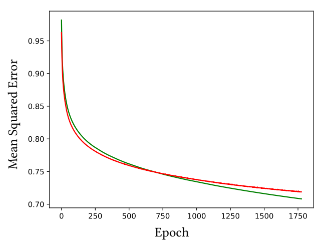
Fig. 11c and Fig. 11d graph the progress of the backpropagation training process for the left leg and right leg voluntary motor DNNs, respectively. Convergence to a small error was achieved after 2,000 epochs for the left leg DNN (19) and after 1,850 epochs for the right leg DNN (17) before triggering the early stopping condition to avoid overfitting.

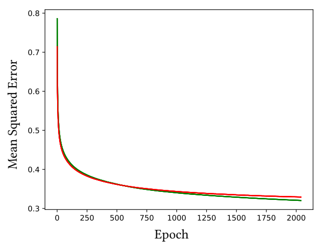
5.2 Reflex Motor DNNs (12,14,16,18,20)

The reflex motor controller of Lee and Terzopoulos [2006] computed the relevant low-level dynamics equations online, which is not biologically plausible. By contrast, our reflex motor controllers are motor DNNs that have learned the desired control functionality from training data synthesized offline.

Fig. 12 shows the architecture of the reflex motor DNNs. It is identical for the 5 reflex motor controllers, except for the sizes of the input and output layers, which are determined by the number


(a) Left arm


(b) Right arm


(c) Left leg

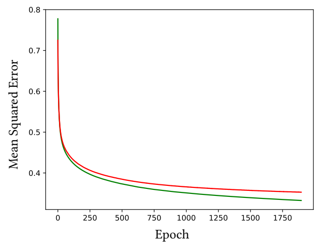

(d) Right leg

Fig. 11. Progress of the backpropagation training of the limb voluntary motor DNNs on the training (green) and validation (red) datasets.

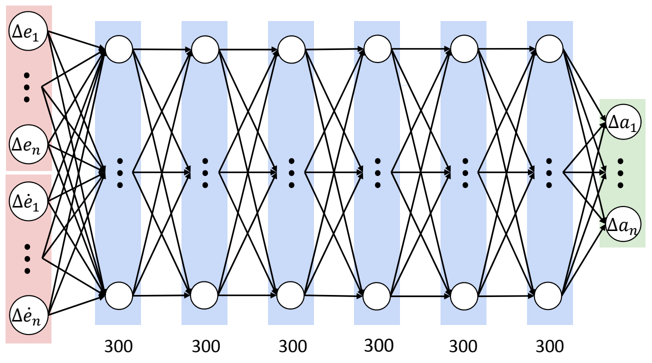


Fig. 12. Architecture of the reflex motor DNNs. The networks for the cervicocephalic and limb musculoskeletal complexes are identical except for the size of the input and output layers, which are determined by the number of muscles.

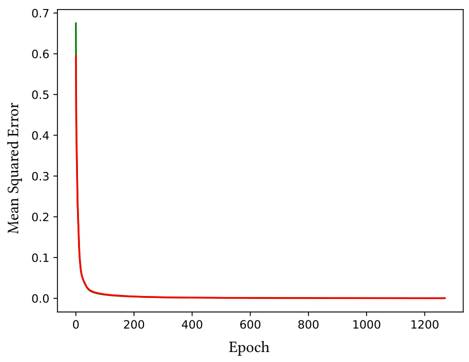


Fig. 13. Progress of the backpropagation training of the right leg reflex motor DNN on the training (green) and validation (red) datasets.

of muscles n in the cervicocephalic, arm, and leg musculoskeletal complexes. The input layer consists of $2n$ units, comprising units that represent the change in muscle strain Δe_i and in strain rate $\Delta \dot{e}_i$, for $1 \leq i \leq n$. Like the voluntary motor DNNs, the networks have 6 hidden layers with 300 units each. The output layer consists of n units providing muscle activation adjustments Δa_i, for $1 \leq i \leq n$, which then contribute additively as $\boldsymbol{a}_r$ to updating the muscle activations according to Equation 1.

To train the reflex motor DNNs, we employ the same training methods as for the voluntary motor DNNs. We again use our biomechanical human musculoskeletal model to synthesize training data. The process is similar to the voluntary motor controller training process, and the training dataset for the reflex controller is generated as described in Appendix B.

After the DNNs are trained, they serve as *online* reflex motor controllers. In general, the backpropagation DNN training processes converged to a small error in the range of 1,200 to 1,600 epochs. Fig. 13 graphs the progress of the training process for the right leg reflex DNN. The graphs for the other 4 reflex DNNs are similar.

6 SENSORY SUBSYSTEM

We now turn to the sensory subsystem (Fig. 6), which includes the eye and retinal models, as well as 10 trained vision DNNs that perform visual processing.

6.1 Eye and Retina Models

6.1.1 Eye Model. We modeled the eyes in accordance with human physiological data.[3] As shown in Fig. 6e, we model the virtual eye as a sphere of 12mm radius that can be rotated with respect to its center around its vertical y axis by a horizontal angle of θ and around its horizontal x axis by a vertical angle of ϕ. The eyes are in their neutral positions, looking straight ahead, when $\theta = \phi = 0°$. We currently model the eye as an ideal pinhole camera with aperture (optical center) at the center of the pupil and with horizontal and vertical fields of view of 167.5°.

We compute the irradiance at any point on the spherical retinal surface at the back of the eye using conventional ray tracing. Sample rays from the positions of photoreceptors on the retinal surface are cast through the pinhole and out into the 3D virtual world, where they recursively intersect with the visible surfaces of virtual objects and, in accordance with the Phong local illumination model, combine with shadow rays to light sources. The RGB values returned by these rays determine the irradiance impinging upon the retinal photoreceptors. Fig. 14 illustrates the retinal "imaging" process.

6.1.2 Placement of the Photoreceptors. To emulate biomimetic foveated vision, we procedurally position the photoreceptors on

[3]The transverse size of an average eye is 24.2 mm and its sagittal size is 23.7 mm. The approximate field of view of an individual eye is 100 degrees to temporal, 45 degrees to nasal, 30 degrees to superior, and 70 degrees to inferior. The combined field of view of the two eyes is approximately 200 degrees horizontally and 135 degrees vertically.

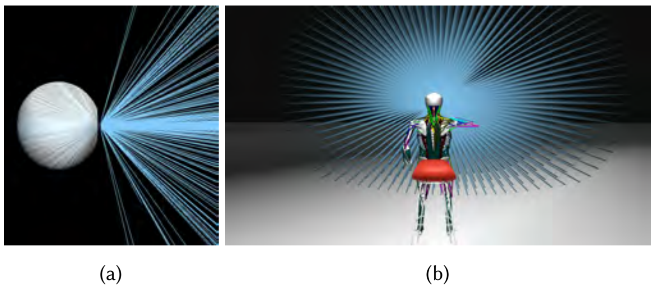


Fig. 14. (a) Rays cast from the positions of photoreceptors on the retina through the pinhole aperture and out into the scene by the ray tracing procedure that computes the irradiance responses of the photoreceptors. (b) Rays cast from both eyes as the seated virtual human looks forward.

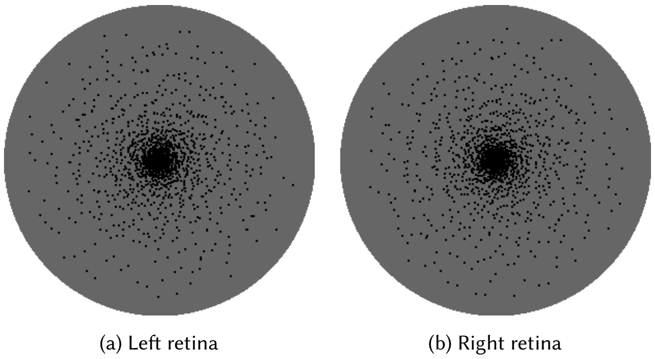


Fig. 15. Positions of the photoreceptors (black dots) on the retinas according to the noisy log-polar model.

the hemispherical retina according to a noisy log-polar distribution, which has greater biological fidelity compared to earlier foveated vision models [Rabie and Terzopoulos 2000]. On each retina, we include 3,600 photoreceptors situated at

$$\boldsymbol{d}_k = e^{\rho_j} \begin{bmatrix} \cos\alpha_i \\ \sin\alpha_i \end{bmatrix} + \begin{bmatrix} \mathcal{N}(\mu,\sigma^2) \\ \mathcal{N}(\mu,\sigma^2) \end{bmatrix}, \quad \text{for } 1 \leq k \leq 3{,}600, \qquad (2)$$

where $0 < \rho_j \leq 40$, incremented in steps of 1, and $0 \leq \alpha_i < 360°$, incremented in 4° steps, and where $\mathcal{N}$ denotes additive IID Gaussian noise. We set mean $\mu = 0$ and variance $\sigma^2 = 0.0025$, which places the photoreceptors in slightly different positions on the two retinas. Fig. 15 illustrates the placement of the photoreceptors on the left and right retinas. Other placement patterns are readily implementable, including more elaborate procedural models [Deering 2005] or photoreceptor distributions empirically measured from biological eyes, all of which differ dramatically from the uniformly-sampled rectangular images common in computer graphics and vision.

6.1.3 Optic Nerve Vectors. The foveated retinal RGB "image" captured by each eye is output for further processing down the visual pathway, not as a 2D array of pixels, but as a 1D vector of length $3{,}600 \times 3 = 10{,}800$, which we call the Optic Nerve Vector (ONV). The raw sensory information encoded in the ONV feeds the vision DNNs that directly control eye movements and feed the neuromuscular motor controller networks that orchestrate neck-actuated head motions and the actions of the limbs.

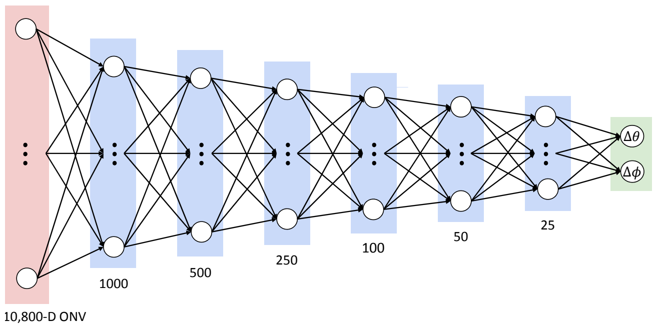


Fig. 16. The feedforward vision DNN architecture. The DNN shown produces saccadic eye movements for visual target foveation.

6.2 Vision DNNs (1–10)

The sensory subsystem includes two types of fully-connected feed-forward DNNs that interpret the sensory information provided by the 10,800-dimensional ONV. The first type controls the eye and head movements, the latter via the cervicocephalic neuromuscular motor controller. The second type estimates arm-to-target 3D discrepancies, Δx, Δy, and Δz, that drive limb actions via the limb neuromuscular motor controllers. Again, through a systematic set of experiments (see our companion paper [Nakada et al. 2018]), we identified a common DNN architecture that works well for all 10 vision DNNs—tapered networks with six hidden layers (Fig. 16).

6.2.1 Foveation DNNs (1,6). Along with the eyes, the left and right foveation DNNs constitute the oculomotor subsystem. The first role of these DNNs is to produce voluntary changes in gaze direction by driving saccadic eye movements to foveate visible objects of interest, thereby observing them with maximum visual acuity. This is illustrated in Fig. 17 for a white sphere in motion that enters the field of view from the lower right, stimulating some peripheral photoreceptors at the upper left of the retina. The maximum speed of saccadic eye movements is 900 degrees/sec, and the eye almost instantly rotates to foveate the visual target.

To aid foveation, fixation, and visual tracking, eye movements induce compensatory head movements, albeit much more sluggish ones due to the considerable mass of the head. Hence, the second role of the foveation DNNs is to control head movements, by driving the cervicocephalic neuromuscular voluntary motor DNN (11) (Fig. 6f) with the average of their two outputs.

As shown in Fig. 16, the input layer to this tapered, fully-connected DNN has 10,800 units in accordance with the dimensionality of the ONV, the output layer has 2 units representing eye rotation adjustments, $\Delta\theta$ and $\Delta\phi$, and there are 6 hidden layers with unit counts as indicated in the figure. The other details of the DNN are per the specifications in Footnote 1.

We use our human model to train the network, as follows: We present a white sphere within the visual field. The responses of the photoreceptors in the retinas of each eye are computed by ray tracing the 3D scene, and they are output as the RGB components of

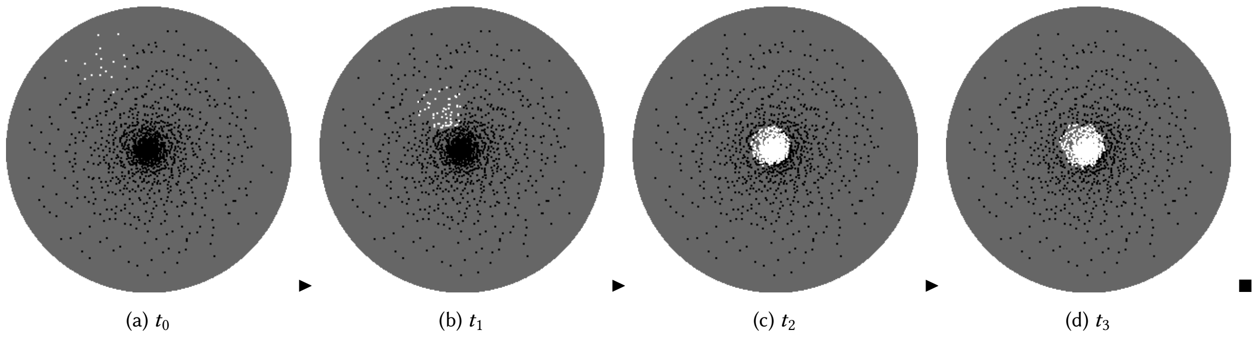


Fig. 17. Time sequence (a)–(d) of photoreceptor responses in the left retina during a saccadic eye movement that foveates and tracks a moving white sphere. At time t_0 the sphere becomes visible in the periphery, at t_1 the eye movement is bringing the sphere toward the fovea, and the moving sphere is being fixated in the fovea at times t_2 and t_3.

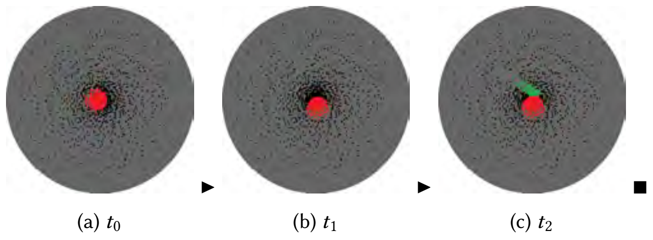


Fig. 18. Photoreceptor responses during an arm-reaching motion toward a moving target sphere. The photoreceptors are simultaneously stimulated by the fixated red sphere and by the green arm entering the eye's field of view from the lower right (upper left on the retina).

the respective ONV. Given its ONV input, the desired output of the network is the angular discrepancies, $\Delta\theta$ and $\Delta\phi$, between the actual gaze directions of the eyes and the known gaze directions that would foveate the sphere. Repeatedly positioning the sphere at random locations in the visual field, we generated a training dataset of 1M input-output pairs. The backpropagation DNN training process converged to a small error after 80 epochs before triggering the early stopping condition to avoid overfitting.

6.2.2 Limb Vision DNNs (2,3,4,5 & 7,8,9,10). The role of the left and right limb (arm and leg) vision DNNs is to estimate the separation in 3D space between the position of the end effector (hand or foot) and the position of a visual target, thus driving the associated limb neuromuscular motor controller to extend the limb to touch the target. This is illustrated in Fig. 18 for a fixated red sphere and a green arm that enters the eye's field of view from the lower right, stimulating peripheral photoreceptors at the upper left of the retina.

The architecture of the limb vision DNN is identical to the foveation DNN in Fig. 16, except for the size of the output layer, which has 3 units, Δx, Δy, and Δz, the estimated discrepancies between the 3D positions of the end effector and visual target.

Again, we use our biomechanical human musculoskeletal model to train the four limb vision DNNs, as follows: A red sphere is presented in the visual field and the trained foveation DNNs are allowed to foveate the sphere. Then, a limb (arm or leg) is extended toward the sphere. Again, the responses of the photoreceptors in the retinas of the eyes are computed by ray tracing the 3D scene, and they are output as the RGB components of the respective ONV. Given the ONV input, the desired output of the network is the discrepancies, Δx, Δy, and Δz, between the known 3D positions of the end effector and visual target. Repeatedly placing the sphere at random positions in the visual field and randomly articulating the limb to reach for it in space, we generated a training dataset of 1M input-output pairs. The backpropagation DNN training process converged to a small error after 388 epochs, which triggered the early stopping condition to avoid overfitting. As expected, due to the greater complexity of this task, the training is significantly slower than for the foveation DNN.

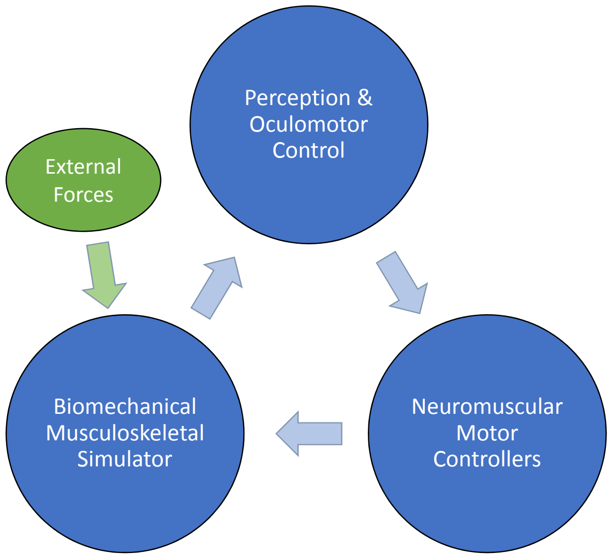


Fig. 19. The sensorimotor simulation cycle.

7 THE SENSORIMOTOR SIMULATION CYCLE

Given the components of Fig. 6, the diagram in Fig. 19 illustrates the embodied, sensorimotor simulation process. Each iteration proceeds as follows: The ONVs output by the eyes are processed by the vision

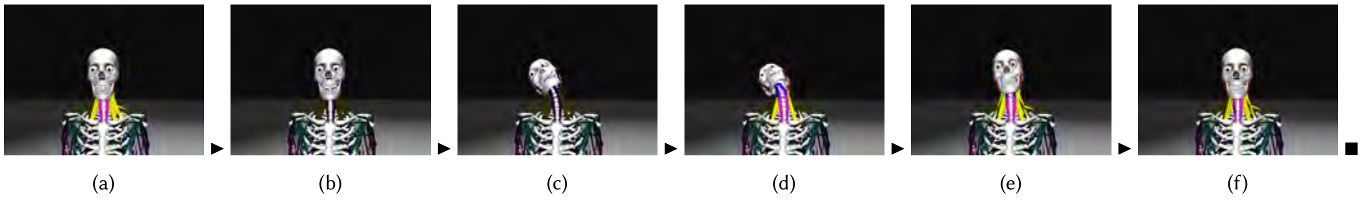


Fig. 20. Sequence of frames rendered from a simulation in which the cervicocephalic neuromuscular controller is deactivated (b) and reactivated (d).

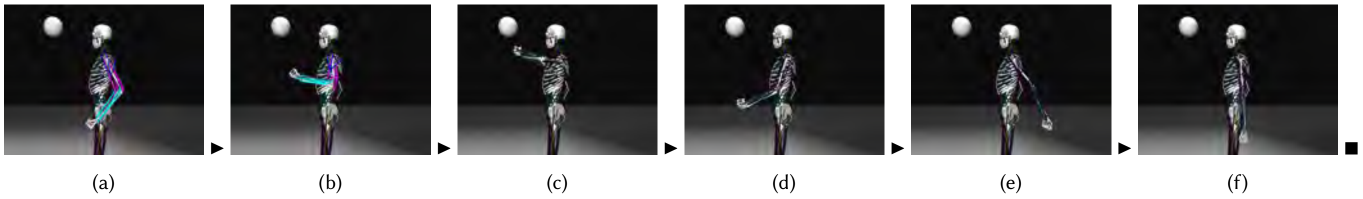


Fig. 21. Sequence of frames rendered from a simulation in which the left arm neuromuscular motor controller is activated (a) to reach toward the sphere and deactivated (c) to release the arm, which swings down in gravity (d–e) before coming to rest (f).

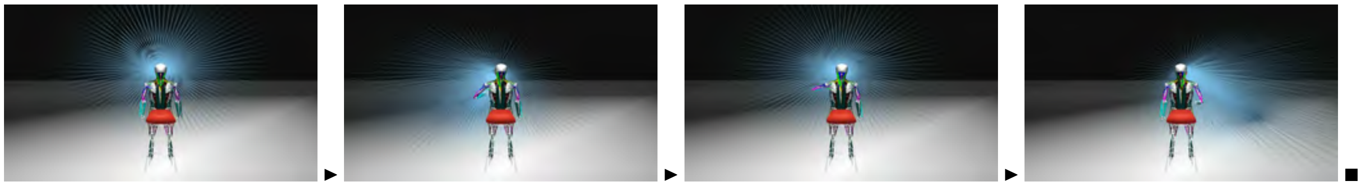


Fig. 22. Sequence of frames rendered from a simulation of the virtual human sitting on a stool and actively executing arm-reaching movements. The blue lines show the rays from the photoreceptors which sample the scene. The sphere is perceived by the eyes and vision DNNs, foveated and tracked through eye movements in conjunction with muscle-actuated head movements driven by the cervicocephalic neuromuscular motor controller and reactive reaching movements driven by the arm neuromuscular motor controllers.

DNNs, and incremental eye rotations that effect saccadic foveation and visual tracking of moving targets are driven by the foveation DNNs. The vision DNNs simultaneously feed the neuromuscular motor controllers, which produce muscle activation signals for the cervicocephalic and limb musculoskeletal complexes. The biomechanical simulator advances the state of the musculoskeletal system through time, thus incrementally actuating the five complexes subject to external forces, such as gravity. The three main simulation components can run asynchronously.

8 EXPERIMENTS AND RESULTS

All the DNNs in our sensorimotor control system were implemented and trained using the Theano library [Bergstra et al. 2010] running on an NVIDIA Titan X GPU installed in a Ubuntu 16.04 system with a 3.2 GHz Intel Core i7 CPU.

8.1 Neuromuscular Controller Activation/Deactivation

Fig. 20 shows a sequence of frames rendered from the simulation of the cervicocephalic musculoskeletal complex with its trained neuromuscular motor controller, which is deactivated (all output muscle activations become 0.0) and then reactivated. Upon controller deactivation, the head naturally droops backward due to gravity. After controller reactivation, the neck muscles lift the head back up, balance it atop the cervical column, and make controlled head movements.

Similarly, Fig. 21 shows a sequence of frames rendered from a simulation of the left arm musculoskeletal complex in which its trained neuromuscular motor controller is deactivated and reactivated, demonstrating fully dynamic control of the arm.

8.2 Perception Demonstration

Fig. 22 shows a sequence of frames rendered from a simulation of our virtual human sitting on a stool and actively executing arm-reaching movements towards a target sphere. As shown from a side view in Fig. 1a,b, the blue lines indicate the rays cast from the retinal photoreceptor positions to sample the 3D scene and compute the irradiance at the photoreceptors. First, the sphere is perceived by the eyes, then processed by the vision DNNs, foveated and tracked through eye movements in conjunction with muscle-actuated head movements controlled by the cervicocephalic neuromuscular motor controller. The muscle-actuated reaching movements are actively controlled by the arm neuromuscular motor controllers.

8.3 Intercepting an Incoming Ball

Fig. 23 shows a sequence of frames rendered from a simulation that demonstrates the active sensorimotor system in action. A cannon

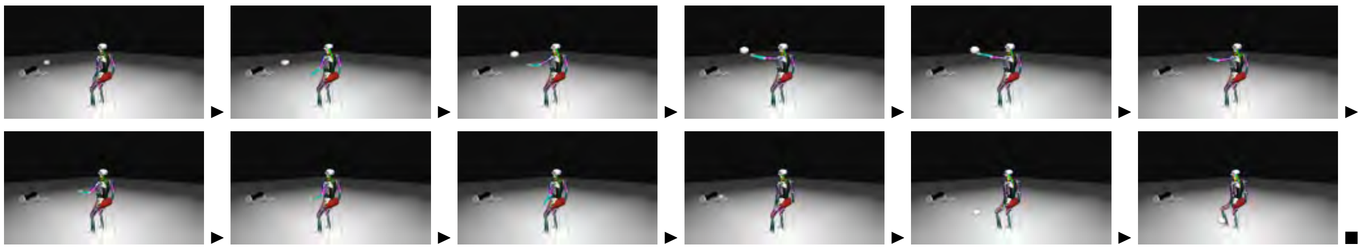


Fig. 23. Sequence of frames rendered from a simulation of the biomechanical virtual human sitting on a stool and actively executing arm and leg reaching movements to intercept balls shot by a cannon. The balls are perceived by the eyes, processed by the vision DNNs, foveated and smoothly pursued through eye movements in conjunction with muscle-actuated head movements controlled by the cervicocephalic neuromuscular motor controller, and the muscle-actuated reaching movements are actively controlled by the arm and leg neuromuscular motor controllers. The red lines indicate the eye viewing directions.

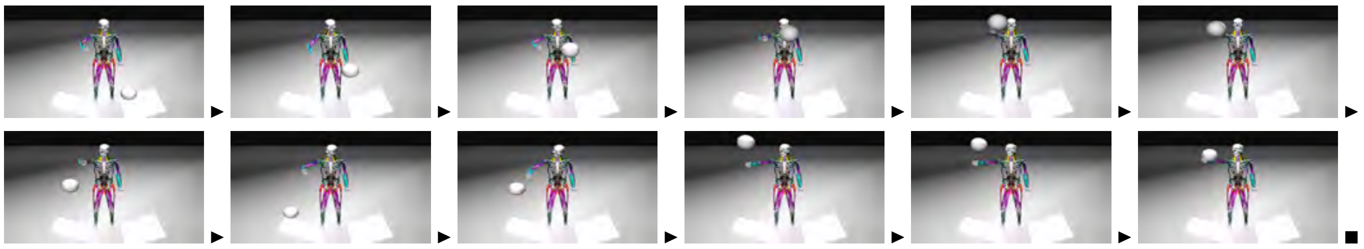


Fig. 25. Sequence of frames rendered from a simulation of the biomechanical virtual human sitting on a stool and actively executing arm-reaching movements to make contact with balls bouncing on multiple sloped planes. The balls are perceived by the eyes, processed by the vision DNNs, foveated and smoothly pursued through eye movements in conjunction with muscle-actuated head movements controlled by the cervicocephalic neuromuscular motor controller. In this sequence, the muscle-actuated active reaching motion is controlled by the right arm neuromuscular motor controller.

Fig. 24. Sequence of frames from a ray tracing-rendered animation of the biomechanical virtual human in the scenario shown in Fig. 23, with a skin covering the musculoskeletal body model.

shoots balls at the virtual human, which actively perceives the balls on its foveated retinas. The ONVs from the eyes are processed by the vision DNNs to enable foveation and visual pursuit of the incoming balls, coupled with cooperative cervicocephalic motor controller driven head motions that pursue the gaze direction. Simultaneously fed by the vision DNNs, the limb neuromuscular motor controllers effectively control the arms and legs to intercept the incoming balls. Thus, our biomechanical virtual human successfully controls itself online to carry out this nontrivial dynamic sensorimotor task.

Fig. 24 shows a frame sequence from a higher quality, ray tracing-rendered animation of our biomechanical virtual human with a skin surface covering its musculoskeletal body model. In the same scenario as in Fig. 23, the virtual human uses its limbs to intercept and deflect balls shot at it by the cannon.

8.4 Intercepting a Bouncing Ball

Fig. 25 shows a sequence of frames from an animation of our biomechanical virtual human actively performing reaching actions to intercept an approaching ball bouncing on multiple sloped planes. The eyes, in conjunction with the head, visually pursue the target ball and the right arm also naturally tracks the ball until it comes within reach. The head, eye, and arm movements are automatically induced by the instantaneous position of the visual target, which is visually estimated by the vision DNNs from the ONV outputs of the eyes. The eyes make fast saccadic movements to quickly fixate the moving visual target and the pursue it, while the head actively orients itself in a natural, cooperative manner, albeit more sluggishly due to its substantial mass. Although the ball is initially out of reach, our virtual human automatically generates lifelike preparatory arm movements toward the approaching ball.

8.5 Drawing a Shape

Fig. 26 shows a sequence of frames from an animation of our biomechanical virtual human drawing the shape of a butterfly with its finger. The butterfly shape was provided as a predefined trajectory in space. Entirely through the neuromuscular control of its arm muscles, our human musculoskeletal model is able to follow the specified trajectory with its finger. Our demonstration video also includes a simulation of our virtual human using its finger to write “SIGGRAPH 2018”.

8.6 Robustness of the Sensorimotor System

To investigate the robustness of the sensorimotor system of our biomechanical human musculoskeletal model, we performed experiments to determine the extent to which its neuromuscular motor

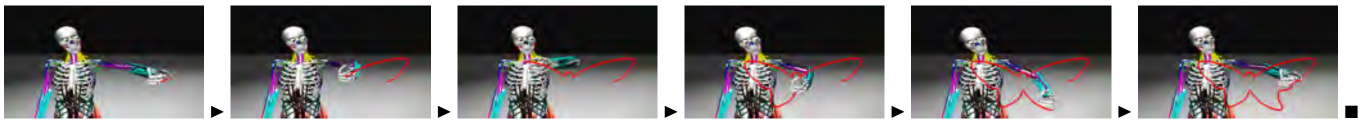


Fig. 26. Sequence of frames rendered from a simulation of our virtual human actively drawing a butterfly with its finger. Controlled head movements are driven by the cervicocephalic neuromuscular motor controller while the arm-reaching and drawing movements are controlled by the left arm neuromuscular motor controller.

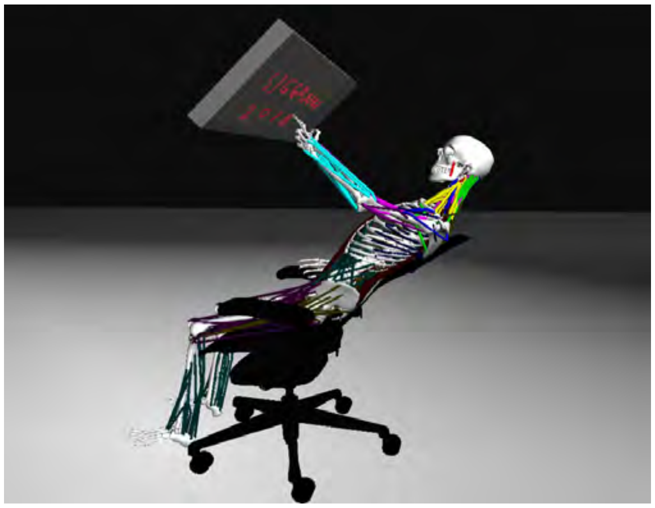


Fig. 27. A frame rendered from a simulation of our virtual human reclining on an office chair and writing with a finger.

controllers are capable of generalizing to situations for which they were not specifically trained.

First, by applying a continuous sinusoidal displacement function we kinematically perturbed the 3D position of the stool vertically and horizontally. Our virtual human attempted to intercept visual targets in a scenario similar to that of Fig. 23. For perturbations of amplitude 10 cm and frequency 3.75 Hz, we discovered that the sensorimotor system was still able to function robustly, whereas for an amplitude of 70 cm and frequency of 6.0 Hz, it often failed to intercept the moving balls with its arms and legs.

Second, we laid the biomechanical virtual human in a supine position and discovered that, without any additional training on the dramatically different relative orientation of gravity, its neuromuscular controllers were sufficiently robust to enable it to reach up and touch with its hand a target sphere placed in various positions over its upper body.

Finally, Fig. 27 shows a frame from an animation of our virtual human writing "SIGGRAPH 2018" while seated in an office chair and continuously reclining down and up, again without any retraining.

9 CONCLUSION

Our sensorimotor control framework is unprecedented in several significant ways, as we will now discuss.

First, it features an anatomically accurate, biomechanically simulated human musculoskeletal model that is actuated by numerous contractile muscles.[4] Second, it employs a biomimetic vision system, including foveated retinas in human-like eyes capable of realistic eye movements. Ray tracing serves in computing the irradiance captured by a biologically-inspired nonuniform distribution of photoreceptors on the retina. Third, through deep learning, its 5 recurrent neuromuscular motor controllers, each of which employ 2 fully-connected deep neural networks and muscle proprioceptive feedback, learn to control the cervicocephalic, arm, and leg musculoskeletal complexes. Fourth, also through deep learning, our virtual human learns to perceive using 10 deep feedforward neural networks fed by the optic nerve outputs of its eyes.

Another novel feature of our sensorimotor approach is the fact the 20 DNNs are trained with large quantities of training data synthesized by the biomechanical human musculoskeletal model itself. In particular, it is implausible that biological neuromuscular motor control mechanisms can perform IK, ID, and minimal muscle effort optimization (MO) computations per se. In our approach, such computations are performed (incrementally) *only* to synthesize training data. Subsequently, backpropagation DNN learning from the synthesized data amounts to nonlinear regression in a high-dimensional space. Finally, the properly-trained DNNs yield biomimetic neuromuscular motor controllers with the required input-output behaviors. It is important to distinguish between these three aspects of our approach, as well as to understand that once the DNNs are trained offline, the training data can be discarded; i.e., training data need *not* be stored or cached for the trained DNNs to operate online. Moreover, the trained neuromuscular motor controllers are very efficient; e.g., our voluntary motor DNNs generate muscle activations in less than a millisecond, whereas the conventional IK-ID-MO approach consumes 120–230 msec per timestep.

We demonstrated the robust performance of our innovative approach in performing nontrivial sensorimotor tasks, such as generating natural limb-reaching actions so as to make contact with moving visual targets, or drawing and writing with a finger. These tasks require simultaneous eye movement control for saccadic foveation and smooth pursuit of visual targets in conjunction with lifelike dynamic head motion control as well as visually-guided dynamic limb control, all hampered by the downward pull of gravity.

Our work should inspire others to pursue increasingly realistic, embodied computational models of human perception and motion, which should eventually impact not only the visual effects and game industries, but also robotics and possibly even the brain sciences.

[4]In our experience [Lee et al. 2009; Lee and Terzopoulos 2006; Nakada and Terzopoulos 2015; Si et al. 2014], biomimetic muscle-actuated control is inherently more stable and robust than robotic torque-actuated control.

9.1 Future Work

From the genesis of this project, we have believed that a deep-learning-based, strongly biomimetic sensorimotor approach is a promising direction for advanced human animation research. Yet it was by no means clear from the outset that our approach would prove to be so remarkably successful for a foveated vision model in conjunction with an anatomically accurate, biomechanically-simulated musculoskeletal model. Prior sensorimotor systems work very differently from ours and, in our opinion, pale in comparison. However, our work to date inevitably has some obvious limitations that we plan to address in our future work.

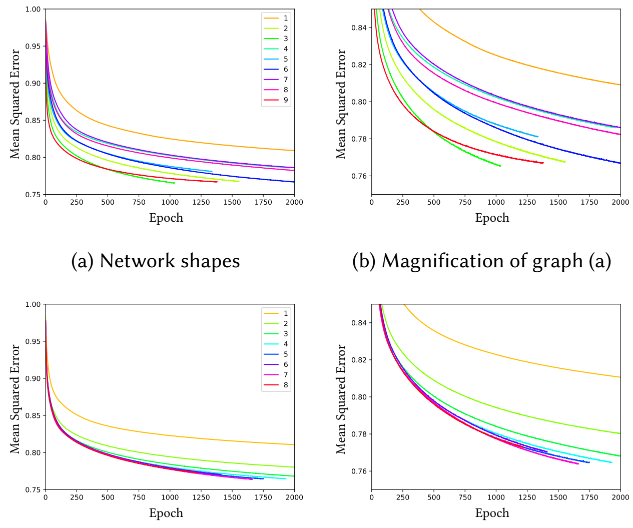
After the DNNs in our prototype sensorimotor system are trained offline, in advance, they remain unchanged as they perform their online sensorimotor control tasks. To address this unnatural shortcoming, we plan to incorporate an online, continual, deep reinforcement learning scheme into our model.

Our current eye model is an ideal pinhole camera. We plan to create a more realistic eye model that not only has a finite-aperture pupil that dilates and constricts to accommodate to the incoming light intensity, but also includes cornea and lens components to refract light rays and is capable of adjusting the optical depth of field through active, ciliary muscle control of the lens deformation. Furthermore, our current eye model is a purely kinematic rotating sphere. We plan to implement a biomechanical eye model (see, e.g., [Wei et al. 2010]) with the typical 7.5 gram mass of a human eyeball, actuated by extraocular muscles, not just the 4 rectus muscles to induce most of the θ, ϕ movement of our current kinematic eyeball, but also the 2 oblique muscles to induce torsion movements around the gaze direction. In this context, the nonlinear pulse-step controller of Lesmana and Pai [2011] for their robotic camera platform system may become relevant, as would their VOR and OKR based gaze stabilization work [Lesmana et al. 2014].

Although our work has demonstrated that in order to perform certain sensorimotor tasks that are crucial first steps in sensorimotor control, it suffices to map rather directly from photoreceptor responses to control outputs, it remains to introduce additional stages of visual processing to enable the execution of more complex tasks. To this end, we will want to increase the number of retinal photoreceptors, experiment with different nonuniform photoreceptor distributions, and (perhaps automatically) construct 2D retino-cortical maps from the 1D retinal ONV outputs.

Our current human model lacks a vestibular system emulating that of the human inner ear and related neural circuits. We plan to develop one that utilizes trained neural controllers that appropriately couple head orientation to eye movement through a Vestibulo-Ocular Reflex (VOR). Also, our current sensory subsystem lacks an Opto-Kinetic Reflex (OKR), which we plan to develop as well, again with trained neural networks.

Ultimately, our goal is to free up our entire biomechanical musculoskeletal model, include additional neuromuscular motor controllers to control its entire spine and torso, and enable it to stand upright, balance in gravity, locomote, and perform various active vision tasks, including attending to and recognizing objects. This is a challenging proposition, but we are excited by the seemingly endless possibilities in this research direction.



(a) Network shapes (b) Magnification of graph (a)

(c) Network depths (d) Magnification of graph (c)

Fig. 28. Training progress of the left arm voluntary motor DNN in experiments with various network shapes (a),(b) and depths (c),(d).

ACKNOWLEDGMENTS

We are indebted to Sung-Hee Lee for creating the biomechanical human musculoskeletal model that we have employed in our work. We thank Gestina Yassa for her voice over of our demonstration video, Zhili Chen for synthesizing handwriting trajectories, and the anonymous reviewers for their constructive comments. We gratefully acknowledge financial support from VoxelCloud, Inc., and a 5-year Scholarship for Long-Term Study Abroad to MN from the Japan Student Organization (JASSO).

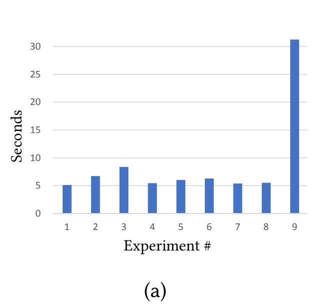
A EXPERIMENTS WITH THE ARM MOTOR DNN

A.1 Network shape:

We conducted training experiments with the left arm voluntary motor controller using 9 different network architectures of various shapes. The network shapes are as follows, where the leftmost number, rightmost number, and intermediate numbers, indicate the number of units in the input layer, output layer, and hidden layers, respectively:

(1) 32 | 100 | 100 | 100 | 100 | 100 | 29
(2) 32 | 300 | 300 | 300 | 300 | 300 | 29
(3) 32 | 500 | 500 | 500 | 500 | 500 | 29
(4) 32 | 300 | 250 | 200 | 150 | 100 | 29
(5) 32 | 500 | 400 | 300 | 200 | 100 | 29
(6) 32 | 100 | 200 | 300 | 400 | 500 | 29
(7) 32 | 100 | 200 | 300 | 200 | 100 | 29
(8) 32 | 300 | 200 | 100 | 200 | 300 | 29
(9) 32 | 3000 | 3000 | 29

This list encompasses wide-shallow and deep network architectures with rectangular, triangular, and diamond shapes. Fig. 28a,b graph the mean squared error as a function of the number of epochs during the training process on the validation datasets for each of the above



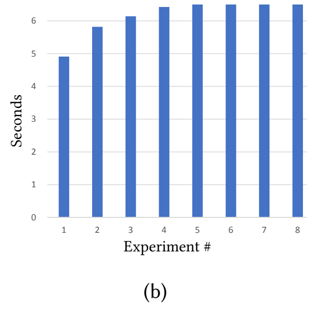


Fig. 29. Average time per epoch for the left arm voluntary motor DNN experiments with various (a) network shapes and (b) network depths.

listed network architectures, by number.[5] All the training processes converged; however, the convergence speed and stability varied by architecture.

The rectangular networks (1,2,3) required fewer epochs to converge as the number of units in each hidden layer increased. The network with 100 units in each hidden layer did not converge well enough to make it useful as a motor controller. Those with 300 and 500 units in each hidden layer converged very well, especially network (3) with 500 units in each layer, which showed one of the best performances. The triangular and the diamond shaped architectures (4,5,6,7,8) converged with average speed; the learning progress was stable but slow. The wide-shallow network (9) was efficient in the sense that it required fewer epochs.

Fig. 29a graphs the average computational cost for each epoch. As expected, the required time increases with an increasing number of weights, the wide-shallow network requiring 6 to 8 times more computation per epoch than deep-narrow ones.

A.2 Network depth:

The next experiment evaluated networks of the same architecture, but different depths, as follows:

(1) 32 | 1800 | 29
(2) 32 | 900 | 900 | 29
(3) 32 | 600 | 600 | 600 | 29
(4) 32 | 450 | 450 | 450 | 450 | 29
(5) 32 | 360 | 360 | 360 | 360 | 360 | 29
(6) 32 | 300 | 300 | 300 | 300 | 300 | 300 | 29
(7) 32 | 257 | 257 | 257 | 257 | 257 | 257 | 257 | 29
(8) 32 | 225 | 225 | 225 | 225 | 225 | 225 | 225 | 225 | 29

To evaluate the effect of network depth independently of the total number of hidden units in the network, each of the above networks includes approximately 1,800 hidden units. Fig. 28c,d graph the mean squared error as a function of the number of epochs of the training process run on the validation dataset. Fewer epochs were required as the network depth increases, until the number of hidden layers reaches 6. However, the required number of epochs increased for 7 hidden layers. It then decreases for 8 hidden layers.

Fig. 29b graphs the average computational cost of each epoch. Interestingly, the required time per epoch increases until the number of hidden layers reaches 6, but the required times with more than 5 hidden layers are nearly the same. The network with 7 hidden layers actually consumed slightly less time than the one with 6 hidden layers, and the time increased for the network with 8 hidden layers.

Thus, this experiment shows that the number of hidden layers affects the regression abilities of the voluntary motor DNN as well the computational time consumed for each training epoch, albeit much less substantially so beyond 6 hidden layers.

[5] 90% of our 1M-item synthesized datasets were used for training the DNNs while the remaining 10% were reserved for validation of the trained DNNs.

B MOTOR CONTROLLER TRAINING DATA SYNTHESIS

The DNNs that implement the voluntary and reflex neuromuscular motor controllers in our sensorimotor system are trained offline in a supervised manner. We synthesize training data offline using our biomechanical human musculoskeletal model simulator. Once they are trained, the DNNs can quickly produce the required muscle activation signals online.

Given the current state of the musculoskeletal system, the current muscle activations, and a desired target state, we compute an appropriate adjustment to each muscle activation so as to drive the musculoskeletal system closer to the target state.

For example, if a limb end effector is in some given position and orientation and we want it to achieve a new pose, first we compute the desired changes of the skeletal joint angles $\boldsymbol{q}$ by solving an inverse kinematics problem. Second, we compute the desired accelerations for the joints to achieve the desired motion in a given time step. Third, we compute the required joint torques by solving the inverse dynamics problem for each joint, subject to external forces such as gravity. Finally, we apply a muscle optimization technique to compute the minimal muscle activations that can generate the desired torque for each joint; i.e, a minimal-effort objective.

Voluntary Motor Controller. For the purposes of training the voluntary motor DNN, the input is the concatenation of the desired movement vector $\Delta\boldsymbol{p} = \boldsymbol{p}^d - \boldsymbol{p}$ between the current pose $\boldsymbol{p}$ and desired pose $\boldsymbol{p}^d$, as well as the current muscle activations $\boldsymbol{a}$, while the desired output of the network in response to this input is the adjustment to the voluntary activations $\Delta\boldsymbol{a}_v = \boldsymbol{a}^d - \boldsymbol{a}$, where $\boldsymbol{a}^d$ is the desired voluntary muscle activations, which is to be added to the neuromuscular controller output per Equation 1. Hence, a single training pair for the voluntary network is as follows:

$$\text{Training input-output pair} \begin{cases} \text{Input:} & [\Delta\boldsymbol{p}, \boldsymbol{a}]; \\ \text{Output:} & \Delta\boldsymbol{a}_v. \end{cases} \tag{3}$$

Reflex Motor Controller. The reflex muscle activation signals are

$$\boldsymbol{a}_r = k_p(\boldsymbol{e}^d - \boldsymbol{e}) + k_d \,\mathrm{sat}_m(\dot{\boldsymbol{e}}^d - \dot{\boldsymbol{e}}), \tag{4}$$

where $\boldsymbol{e}$ and $\dot{\boldsymbol{e}}$ are the current muscle strain and strain rates, and $\boldsymbol{e}^d$ and $\dot{\boldsymbol{e}}^d$ are the desired strain and strain rates, respectively, and where the saturation function

$$\mathrm{sat}_m(x) = \begin{cases} x & |x| < m; \\ m\,\mathrm{sgn}(x) & \text{otherwise,} \end{cases} \tag{5}$$

avoids instability from overly large derivative feedback. We set the proportional gain $k_p = 8.0$, the derivative gain $k_d = 0.05$, and $m = 2.0$, and compute the desired stain and strain rates by the setpoint method [Lee et al. 2009]. Therefore, for the purposes of training the reflex DNN controllers, the input is the concatenation of the desired adjustment of the strain $\Delta\boldsymbol{e} = \boldsymbol{e}^d - \boldsymbol{e}$ and strain rate $\Delta\dot{\boldsymbol{e}} = \dot{\boldsymbol{e}}^d - \dot{\boldsymbol{e}}$, while the desired output of the network in response to the input is the reflex activations adjustment $\Delta\boldsymbol{a}_r$, which is to be added to the neuromuscular controller output per Equation 1. Hence, a single training pair for the reflex network is as follows:

$$\text{Training input-output pair} \begin{cases} \text{Input:} & [\Delta\boldsymbol{e}, \Delta\dot{\boldsymbol{e}}]; \\ \text{Output:} & \Delta\boldsymbol{a}_r. \end{cases} \tag{6}$$

We iteratively update the joint angles in a gradient descent manner such that the difference between the current pose $\boldsymbol{q}$ and target pose $\boldsymbol{q}^*$ is minimized. The controller determines the required accelerations to reach the target at each time step h using the following PD approach:

$$\ddot{\boldsymbol{q}}^* = k_p(\boldsymbol{q}^* - \boldsymbol{q}) + k_d(\dot{\boldsymbol{q}}^* - \dot{\boldsymbol{q}}), \tag{7}$$

with proportional gain $k_p = 2(1-\gamma)/h^2$, where γ is the error reduction rate, and derivative gain $k_d = 2/h$. We set $h = \gamma = 0.1$.

We use the hybrid recursive dynamics algorithm due to Featherstone [2014], which makes it possible to compute the desired accelerations for acceleration-specified joints and the desired torques for torque-specified joints, as follows: First, we set the muscle-driven joints as acceleration-specified joints, while the passive joints remain torque-specified joints. We compute the desired accelerations for the muscle-driven joints and run the hybrid dynamics algorithm to compute the resulting accelerations for the passive joints. Second, we advance the system to the next time step using a first-order implicit Euler method, and use the hybrid dynamics algorithm to compute torques for the muscle-driven joints, thus obtaining the desired torques for the muscle-driven joints and accelerations for the passive joints.

After the desired torques are obtained, an optimization problem for agonist and antagonist muscles is solved to compute the desired minimal-effort muscle activation levels.

Additional details about the above numerical techniques are found in [Lee et al. 2009; Lee and Terzopoulos 2006].

REFERENCES

J. Bergstra, O. Breuleux, F. Bastien, P. Lamblin, R. Pascanu, G. Desjardins, J. Turian, D. Warde-Farley, and Y. Bengio. 2010. Theano: A CPU and GPU math compiler in Python. In *Proc. 9th Python in Science Conference*. Austin, TX, 1–7.

A.L. Cruz Ruiz, C. Pontonnier, N. Pronost, and G. Dumont. 2017. Muscle-based control for character animation. *Computer Graphics Forum* 36, 6 (2017), 122–147.

M. F. Deering. 2005. A photon accurate model of the human eye. *ACM Transactions on Graphics* 24, 3 (2005), 649–658.

P. Faloutsos, M. van de Panne, and D. Terzopoulos. 2001. Composable controllers for physics-based character animation. In *Proc. 28th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '01)*. Los Angeles, CA, 251–260.

Y. Fan, J. Litven, and D.K. Pai. 2014. Active volumetric musculoskeletal systems. *ACM Transactions on Graphics* 33, 4 (2014), 152.

R. Featherstone. 2014. *Rigid Body Dynamics Algorithms*. Springer, New York, NY.

T. Geijtenbeek, M. Van De Panne, and A.F. Van Der Stappen. 2013. Flexible muscle-based locomotion for bipedal creatures. *ACM Transactions on Graphics* 32, 6 (2013), 206.

I. Goodfellow, Y. Bengio, and A. Courville. 2016. *Deep Learning*. MIT Press, Cambridge, MA.

R. Grzeszczuk, D. Terzopoulos, and G. Hinton. 1998. NeuroAnimator: Fast neural network emulation and control of physics-based models. In *Computer Graphics Proceedings, Annual Conference Series*. Orlando, FL, 9–20. Proc. ACM SIGGRAPH 98.

K. He, X. Zhang, S. Ren, and J. Sun. 2015. Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification. In *Proc. IEEE International Conference on Computer Vision*. Santiago, Chile, 1026–1034.

J.K. Hodgins, W.L. Wooten, D.C. Brogan, and J.F. O'Brien. 1995. Animating human athletics. In *Proc. ACM SIGGRAPH '95 Conference*. Los Angeles, CA, 71–78.

D. Holden, T. Komura, and J. Saito. 2017. Phase-functioned neural networks for character control. *ACM Transactions on Graphics* 36, 4 (2017), 42.

W. Huang, M. Kapadia, and D. Terzopoulos. 2010. Full-body hybrid motor control for reaching. In *Motion in Games (Lecture Notes in Computer Science, Vol. 6459)*. Springer-Verlag, Berlin, 36–47.

A.-E. Ichim, P. Kadleček, L. Kavan, and M. Pauly. 2017. Phace: Physics-based face modeling and animation. *ACM Transactions on Graphics* 36, 4 (2017), 153.

P. Kadleček, A.-E. Ichim, T. Liu, J. Křivánek, and L. Kavan. 2016. Reconstructing personalized anatomical models for physics-based body animation. *ACM Transactions on Graphics* 35, 6 (2016), 213.

K. Kähler, J. Haber, H. Yamauchi, and H.-P. Seidel. 2002. Head shop: Generating animated head models with anatomical structure. In *Proc. 2002 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. San Antonio, TX, 55–63.

D. Kingma and J. Ba. 2014. *Adam: A method for stochastic optimization.* Technical Report. arXiv preprint arXiv:1412.6980.

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

S.-H. Lee and D. Terzopoulos. 2006. Heads Up! Biomechanical modeling and neuromuscular control of the neck. *ACM Transactions on Graphics* 23, 212 (2006), 1188–1198. Proc. ACM SIGGRAPH 2006.

Y. Lee, M.S. Park, T. Kwon, and J. Lee. 2014. Locomotion control for many-muscle humanoids. *ACM Transactions on Graphics* 33, 6 (2014), 218.

Y. Lee, D. Terzopoulos, and K. Waters. 1995. Realistic modeling for facial animation. In *Computer Graphics Proceedings, Annual Conference Series (Proc. ACM SIGGRAPH 95)*. Los Angleles, CA, 55–62.

M. Lesmana, A. Landgren, P.-E. Forssén, and D.K. Pai. 2014. Active gaze stabilization. In *Proc. Indian Conference on Computer Vision, Graphics, and Image Processing*. Bangalore, India, Article 81, 8 pages.

M. Lesmana and D.K. Pai. 2011. A biologically inspired controller for fast eye movements. In *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, Shanghai, China, 3670–3675.

L. Liu and J. Hodgins. 2017. Learning to schedule control fragments for physics-based characters using deep Q-learning. *ACM Transactions on Graphics* 36, 3 (2017), 29.

M. Nakada, H. Chen, and D. Terzopoulos. 2018. Deep learning of biomimetic visual perception for virtual humans. In *Proc. ACM Symposium on Applied Perception (SAP '18)*. Vancouver, BC, 1–8.

M. Nakada and D. Terzopoulos. 2015. Deep learning of neuromuscular control for biomechanical human animation. In *Advances in Visual Computing (Lecture Notes in Computer Science, Vol. 9474)*. Springer, Berlin, 339–348. Proc. *International Symposium on Visual Computing*, Las Vegas, NV, December 2015.

X.B. Peng, G. Berseth, K. Yin, and M. Van De Panne. 2017. Deeploco: Dynamic locomotion skills using hierarchical deep reinforcement learning. *ACM Transactions on Graphics* 36, 4 (2017), 41.

T.F. Rabie and D. Terzopoulos. 2000. Active perception in virtual humans. In *Proc. Vision Interface 2000*. Montreal, Canada, 16–22.

P. Sachdeva, S. Sueda, S. Bradley, M. Fain, and D.K. Pai. 2015. Biomechanical simulation and control of hands and tendinous systems. *ACM Transactions on Graphics* 34, 4 (2015), 42.

E.L. Schwartz. 1977. Spatial mapping in the primate sensory projection: Analytic structure and relevance to perception. *Biological Cybernetics* 25, 4 (1977), 181–194.

W. Si, S.-H. Lee, E. Sifakis, and D. Terzopoulos. 2014. Realistic biomechanical simulation and control of human swimming. *ACM Transactions on Graphics* 34, 1, Article 10 (Nov. 2014), 15 pages.

E. Sifakis, I. Neverov, and R. Fedkiw. 2005. Automatic determination of facial muscle activations from sparse motion capture marker data. *ACM Transactions on Graphics* 1, 212 (2005), 417–425.

S. Sueda, A. Kaufman, and D.K. Pai. 2008. Musculotendon simulation for hand animation. *ACM Transactions on Graphics* 27, 3 (Aug. 2008), 83.

D. Terzopoulos and T.F. Rabie. 1995. Animat vision: Active vision with artificial animals. In *Proc. Fifth International Conference on Computer Vision (ICCV '95)*. Cambridge, MA, 840–845.

D. Terzopoulos and K. Waters. 1990. Physically-based facial modelling, analysis, and animation. *Computer Animation and Virtual Worlds* 1, 2 (1990), 73–80.

J.M. Wang, S.R. Hamner, S.L. Delp, and V. Koltun. 2012. Optimizing locomotion controllers using biologically-based actuators and objectives. *ACM Transactions on Graphics* 31, 4, Article 25 (2012), 11 pages.

Q. Wei, S. Sueda, and D.K. Pai. 2010. Biomechanical simulation of human eye movement. In *Biomedical Simulation (Lecture Notes in Computer Science)*, Vol. 5958. Springer-Verlag, Berlin, 108–118.

S.H. Yeo, M. Lesmana, D.R. Neog, and D.K. Pai. 2012. Eyecatch: Simulating visuomotor coordination for object interception. *ACM Transactions on Graphics* 31, 4 (2012), 1–10.