Neural Network for Dynamic Human Motion Prediction

Mohammad Bataineh, Timothy Marler, Karim Abdel-Malek, and Jasbir Arora
Virtual Soldier Research Program – Center for Computer-Aided Design, The University of Iowa, Iowa City, IA, USA

Email addresses: bataineh.moe@gmail.com (Mohammad Bataineh), tmarler@engineering.uiowa.edu (Timothy Marler), amalek@engineering.uiowa.edu (Karim Abdel-Malek), jasbir-arora@uiowa.edu (Jasbir Arora)

Corresponding author: Mohammad Bataineh will handle correspondence at all stages of refereeing and publication, also post-publication. (Tel: +1-319-331-5454; Email: bataineh.moe@gmail.com; Address: 330 S. Madison Street. Iowa City, IA 52242. USA).

Abstract

Digital human models (DHMs) are critical for improved designs, injury prevention, and a better understanding of human behavior. Although many capabilities in the field are maturing, there are still opportunities for improvement, especially in motion prediction. Thus, this work investigates the use of an artificial neural network (ANN), specifically a general regression neural network (GRNN), to provide real-time computation of DHM motion prediction, where the underlying optimization problems are large and computationally complex. In initial experimentation, a GRNN is used successfully to simulate walking and jumping on a box while using physics-based human simulations as training data. Compared to direct computational simulations of dynamic motion, use of GRNN reduces the calculation time for each predicted motion from 1-40 minutes to a fraction of a second with no noticeable reduction in accuracy. This work lays the foundation for studying the effects of changes to training regiments on human performance.

Keywords: digital human modeling, artificial neural networks, motion prediction, general regression neural network.

1. Introduction

The use of digital humans is becoming more prevalent with the upstream design of any product or process that involves human interaction or human systems integration. In order to enhance the design process as effectively as possible, computationally fast human simulation and analysis become critical. The faster one can simulate and obtain feedback concerning human performance, the faster one can evaluate and refine designs. A cornerstone of human performance is simulated motion, which often requires dynamic analysis (consideration of forces, acceleration, and inertia, not just kinematics). However, accurate dynamic simulation and analysis can be computationally demanding, depending on the task being simulated. Therefore, this work presents the use of an artificial neural network (ANN) to provide fast motion simulation.

Whether the motion tasks are simulated using data-based methods (Chaffin, 2002; Moeslund, Hilton, & Krüger, 2006), which depend on motion capture systems to track, record, and reproduce human motion during various tasks, or using physics-based methods (Xiang, Chung, et al., 2010), which primarily entail using optimization to predict motion, techniques for capturing one's history and the consequent strategy for completing the task continue to be a challenge. Why do different people with similar capabilities and size, for instance, enter the same vehicle in different ways? Although various human modeling methods can capture the nuances of one's motion or the cause and effect demonstrated with changes in parameters, few methods offer the ability to capture one's *strategy* in approaching a task without significant input from the user. Hence, there is a need for an algorithm that produces real-time motion prediction and can incorporate a history of experience.

Motion-capture-based methods are limited in terms of their ability to produce different motions that correspond to changes in the task parameters, because the underlying algorithm depends on prerecorded data that cannot be changed due to the change in the task conditions. In addition, these methods do not incorporate dynamics; they do not capture effects of loads and inertia. Alternatively, physics-based methods like predictive dynamics (PD), which is an optimization-based motion prediction algorithm (Xiang, Chung, et al., 2010), tend to be more flexible in showing the effects of changes in task parameters, especially with respect to dynamics. In addition, these methods are predictive; they predict human performance with minimal dependence on prerecorded data. Computational speed, however, can be a limiting factor with PD, when real-time performance feedback is needed. Depending on the task being simulated and its settings, PD can require up to 40 minutes (the PD algorithm is run on a Windows 7 computer with an Intel® Core™ i3 processor and 8 GB of RAM), even with small changes in the configuration. A variety of techniques are being explored to address this challenge, and the use of an ANN is especially promising.

An ANN can be used for real-time motion prediction and can be integrated with the physics-based motion simulation method, PD, in order to improve the computational speed when predicting a motion task. An ANN also provides a platform for incorporating alternative sources for real-time motion prediction, such as motion capture data, if desired. Unlike other simulation tools, ANNs are capable of providing acceptable simulations for a problem without the need for complex time-consuming algorithms. This work 1) demonstrates the feasibility and advantages of using ANNs for direct motion prediction, and 2) presents the use of one of the ANN types as an appropriate network for successful simulation of such problems.

An ANN is a mathematical model for predicting system output, inspired by the structure and function of human biological neural networks. Compared to other simulation and statistical tools, ANN is fast and produces relatively accurate and acceptable simulations for complex systems. ANNs entail two steps or processes. First they are *trained* using some form of pre-existing data. Essentially, optimization is used to set model parameters. After its training process is completed, it is then *run* and provides relatively fast output given various input conditions. ANNs can be powerful tools for generalizing, which means providing accurate and acceptable results for all inputs conditions, many practical problems (Coit, Jackson, & Smith, 1998; Twomey & Smith, 1998), and hence have been used successfully in many digital human model (DHM)-related problems (Bataineh, 2015; Bataineh & Marler, 2013; Bataineh, Marler, & Abdel-Malek, 2013; Bu, Okamoto, & Tsuji, 2009; Kang, Kim, Park, & Kim, 2007; Li, Li, & Song, 2007; Zhang, Horváth, Molenbroek, & Snijders, 2010; Zhao, Zheng, & Wen, 2010). Motion-related applications include, but are not limited to, robotics and controller system motion, motion analysis, reconstruction of dynamic objects, and time-series dynamic prediction and classification. In general, the main use of ANNs has been focused on human model posture prediction (Jung & Park, 1994; Zhang, et al., 2010) and motion prediction of robotics and dynamics systems (Frank, Davey, & Hunt, 2001; Stakem & AlRegib, 2008). One approach proposed the use of multiple ANNs in controlling a robot manipulator (Y. H. Kim & Lewis, 1999). The system was evaluated successfully on a two-link robot manipulator, demonstrating the ANN’s ability to handle the nonlinear unknown parameters in the system manipulator. Moreover, the feedforward-backpropagation network, which was trained by gaits of various people, is used to recognize humans automatically (Yoo, Hwang, Moon, & Nixon, 2008). In other motion-related applications, ANNs have been used as an indirect source that led to providing improved motion predictions. Lung tumor motion during respiration was predicted in advance using ANNs (Isaksson, Jalden, & Murphy, 2005). Most of the preceding scholars use feedforward-backpropagation ANNs with single or few outputs to preserve accuracy.

So far, ANNs have been applied only to very specific scenarios in DHM problems and have not been developed for robust use with complex problems like whole-body dynamic motion prediction. In general, ANNs have been used to solve confined systems with a relatively small number of inputs and outputs. Most applications have involved feedforward-backpropagation networks, which have memory and accuracy issues when used with a large number of inputs and outputs. Thus, this work explores using other types of ANNs for relatively large and complex human-modeling problems.

The overarching hypothesis is that if designed/selected properly, ANNs can in fact be used to simulate human motion quickly and accurately, despite a relatively large number of outputs. We contend that a radial-basis network (RBN) is most appropriate, because it has the smallest number of parameters to be set when simulating a problem with a large number of outputs. In addition, it provides a global solution when optimizing the network parameter values (during the training process). Specifically, we propose using a general regression neural network (GRNN), which is a type of RBN. The work integrates GRNN with PD to increase the computational speed of PD with minimal detriment to accuracy. This is shown using two different simulated tasks with a large number of outputs (on the order of hundreds), both of which run in under one second. Eventually, PD can be replaced by GRNN, which is trained to provide a standalone instant motion simulation. The next section describes the necessary parameters (inputs and outputs) of the PD simulated tasks, as well as details of the underlying ANN architecture.

2. Methods

2.1. Digital human model and physics-based motion prediction

As a foundation for the proposed method, this section summarizes the digital human model as well as PD. This work capitalizes on and adds to a foundation of virtual human modeling capabilities, housed within a human model called Santos (Abdel-Malek, et al., 2006; Abdel-Malek, et al., 2007). Santos, as shown in Fig. 1, is a highly realistic, biomechanical computer-based human that predicts, among other things, static posture, dynamic motion, joint strength, and development of fatigue. Such capabilities can be used to predict and assess human function, providing task performance measures and ergonomic analysis. Thus, in a virtual world, Santos can help design and analyze various products and processes. In addition, Santos can help study and evaluate various restrictions and impediments, such as fatigue, reduced range of motion, environmental obstacles, etc.

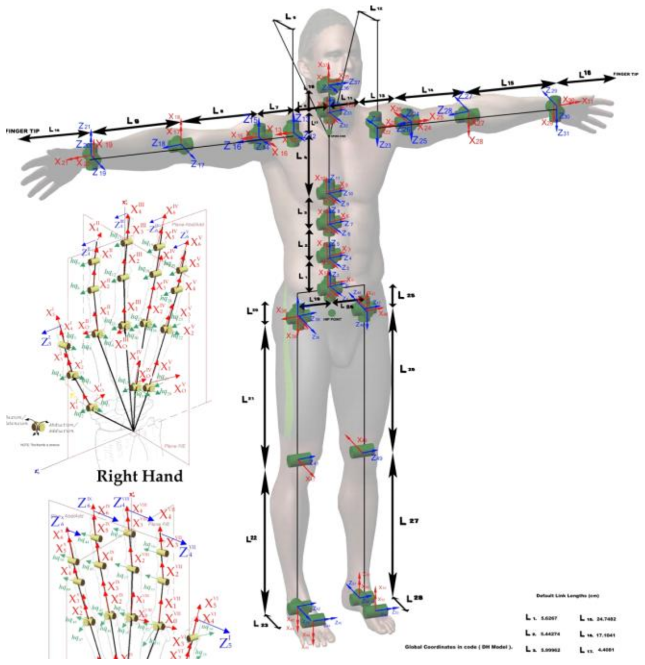
Right Hand
FINGER TIP
FINGER TIP
Default Link Lengths (cm)
Global Coordinates in code (DH Model)

Fig. 1. Schematic of the skeleton model for the virtual human model Santos.

A key aspect of any virtual human is the ability to simulate human posture and motion realistically and quickly while considering external and internal loads/forces. With respect to motion, there are traditionally two types of dynamics problems that need to be addressed. In the first problem, called forward dynamics, the external forces and torques on the system are known and the motion of the system is desired. The problem is solved by integrating the governing equations of motion forward in time using a numerical algorithm. In the second problem, called inverse dynamics, the motion of the system is known (i.e., from motion capture), and the forces and torques causing the motions are calculated using the equations of motion. Both of these problems can be solved using traditional multi-body dynamics software. The problem of *predictive dynamics* arises when one wants to simulate the human motion for any task. In this problem, both the joint torques and the motion of the joint are unknown. Therefore, the problem becomes more difficult to solve. With the PD approach, the joint angles (one for each degree of freedom, or DOF) essentially provide design variables that are determined through optimization. The objective function(s) is one or more human performance measure, such as energy consumption, discomfort, and joint displacement. Including the dynamic equations of motion as constraints then ensures that the laws of physics are satisfied.

The specifics of PD (Xiang, Arora, Rahmatalla, & Abdel-Malek, 2009; Xiang, Chung, et al., 2010) are summarized as follows. In general, predicting dynamic human motion is approached as an optimization problem (Arora, 2004), as shown in Equation 1. This formulation provides the context for the discussion of inputs and outputs used with the proposed neural network. Joint angle profiles over time are represented as B-spines, and the control points, which dictate the shape of the profile curve, serve as design variables in an optimization problem. The problem entails determining design variables $\boldsymbol{q}$, which represent the control points (i.e., joint angle profiles) of all body DOFs, in order to minimize the objective function, $f(\boldsymbol{q})$, subject to the physical equality ($h_i(\boldsymbol{q})$) and inequality ($g_j(\boldsymbol{q})$) constraints. The control points ($\boldsymbol{q}$) form B-splines for all DOFs that simulate the motion of the DHM. Having more control points in a B-spline leads to more accurate motion simulation but increases the number of design variables and can thus increase computational complexity. $\boldsymbol{q}_{MoCap}$ is a vector that represents the reference motion provided by motion capture.

$$\begin{array}{ll} \text{Find:} & \boldsymbol{q} \text{ (control points for 55-DOFs)} \\ \text{Minimize:} & f(\boldsymbol{q}) = f(\boldsymbol{q} - \boldsymbol{q}_{MoCap}) + f(\sum_1^{DOFs} joint\ torque) \\ \text{Subject to:} & h_i(\boldsymbol{q}) = 0,\ i = 1, \ldots, m \\ & g_j(\boldsymbol{q}) \leq 0,\ j = 1, \ldots, k \\ & \text{Equations of motion, including reaction forces} \end{array} \quad (1)$$

The objective function $f(\boldsymbol{q})$ can vary slightly depending on the task being simulated, but it generally includes two components. First, the difference between the predicted motion $\boldsymbol{q}$ and a *seed* motion based on experimental motion capture is minimized ($f(\boldsymbol{q} - \boldsymbol{q}_{MoCap})$). This component helps represent one's overall strategy when performing a task, as opposed to kinematic and dynamic nuances. The second component includes the tendency to minimize the joint torque being used to complete a task and thus minimize the energy being used ($f(\sum_1^{DOFs} joint\ torque)$).

The constraints, which represent $h_i(\boldsymbol{q})$ and $g_j(\boldsymbol{q})$, include the following: contact points (between the avatar and the environment), joint-angle limits that represent one's range of motion (Marler et al., 2008), torque limits that represent one's strength limits, restriction on the zero moment point (responsible for balance), ground reaction forces, and equations of motion.

Since its development, PD has been used to simulate different motion tasks and scenarios (J. H. Kim, et al., 2008; J. H. Kim, Xiang, Yang, Arora, & Abdel-Malek, 2010; Kwon, et al., 2014; Xiang, Arora, & Abdel-Malek, 2010, 2012; Xiang, Arora, Rahmatalla, et al., 2010). The inputs for each simulation take two forms: avatar-based and task-based. Avatar-based inputs include anthropometric parameters (i.e., skeletal link lengths, mass for body segments, etc.), joint ROMs, torque limits, which represent strength limits, and loads applied to the avatar as a result of external factors. Task-based inputs include parameters that define the characteristics of a task (i.e., step size in a walking task, box height in a jumping-on-the-box task, etc.). Each newly developed PD task/simulation is validated using motion capture and force plates (Rahmatalla, Xiang, Smith, Meusch, & Bhatt, 2011).

Creating a large number of different task conditions (thousands) is difficult, because it can be time consuming. The running time for each case, even with minor condition changes, can take minutes to hours in order to complete and produce the simulation. The long running time is due to the large size of the PD problem with respect to the number of outputs. The number of outputs in a PD task is approximately 500-700 outputs (i.e., the outputs represent $\boldsymbol{q}$ for all 55 DOFs, which are the design variables), depending on the number of control points in each task, which can slow down the optimization. In addition, this process cannot be automated completely, because the simulations require some post-processing before considering the case acceptable and usable. Hence, the time-cost issue leads to a PD problem with a limited number of simulations available to be used by pattern recognition tools like ANN to be trained to provide real-time PD simulations. Therefore, a carefully selected type of ANN that is capable of being trained well with a limited number of simulations needs to be employed to provide the most acceptable simulation results. The following section illustrates the architecture of the proposed ANN for successful real-time simulation of a PD problem that has a limited number of training cases.

2.2. General regression neural network (GRNN)

This work proposes the use of a GRNN for simulating motion prediction in DHM. Among the types of RBN, which is known to be powerful when simulating large-scale and complex problems like motion, GRNN has the fewest parameters to be set among RBN types for successful use in DHM applications with relatively large problems (Bataineh, 2012). ANNs consist of three main parts (Fig. 2): (1) an input layer, (2) a hidden layer, and (3) an output layer. The input and output layers consist of the system's inputs and outputs, respectively. The hidden layer represents the core of the ANN and consists of units called neurons. Inside the hidden neurons, which are also called the basis functions, the main mathematical calculations occur while processing the inputs and providing the proper outputs.

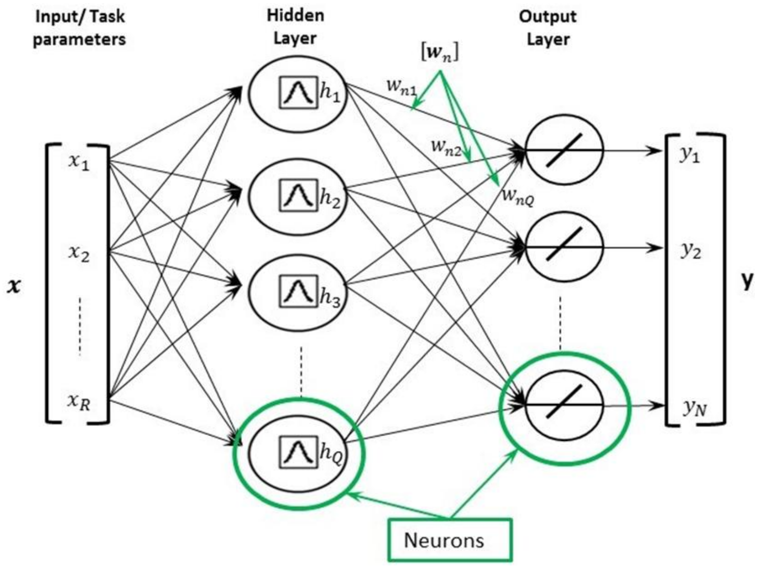

Fig. 2. General regression neural network architecture.

The network input is represented by $\boldsymbol{x} = [x_1, x_2, \dots, x_R]$ and provides the input for each neuron in the hidden layer. R is the number of inputs, Q is the number of training cases as well as the hidden neurons $[1, 2, \dots, Q]$, and N is the number of outputs. The hidden neuron in the ANN receives input(s) from the input layer, completes a mathematical transformation/calculation, and sends the result to the neuron(s) of the output layer.

Inside each hidden neuron, there is a radial transfer function that produces outputs depending on the provided input (Wasserman, 1993). Once the ith hidden neuron receives the input $\boldsymbol{x}$, the Euclidian norm of the difference between $\boldsymbol{x}$ and the hidden neuron's center $\boldsymbol{U}_i^I$ is calculated to produce A_i (Equation 2). Then, the value A_i is multiplied by the bias constant B to provide a_i, which is called the radial distance (Equation 3). The radial function output h_i is then calculated as a function of a_i, (Equation 4).

$$A_i = \sum_{j=1}^{R} \left\| u_{ij}^{I} - x_j \right\| \tag{2}$$

$$a_i = A_i * B \tag{3}$$

$$h_i = rad(a_i) \tag{4}$$

Each output neuron receives the hidden neuron's output h_i as its input. The output neuron essentially combines the output of all hidden neurons in a weighted sum to provide the final network output (Equation 5). The vector $\boldsymbol{W}_k^O$ in the k^{th} output neuron represents the weight associated with that neuron necessary to provide the proper value for the k^{th} output y_k. The output layer has *N* neurons, where *N* is the number of outputs $[y_1, y_2, \dots, y_N]$.

$$y_k = \frac{\sum_{q=1}^{Q} h_q \, . \, w_{kq}^{o}}{\sum_{q=1}^{Q} h_q} \tag{5}$$

The training process in a GRNN involves determining the most appropriate value of each hidden neuron center $\boldsymbol{U}_i^I$ and each output weight vector $\boldsymbol{W}_k^O$. The training cases are used as the hidden neuron centers, where the number of hidden neurons equals the number of training cases Q. The weighting vector is set as the outputs of the training cases. More details regarding the GRNN architecture are provided in the literature (Specht, 1991).

2.3. GRNN for Motion Simulation

The crux of the proposed method entails using an ANN to approximate dynamic motion prediction. In essence, an ANN is used to create a meta-model or hyper response surface of the PD computational model. Combining these two elements provides two modes of use.

First, given that PD entails running a gradient-based optimization model, and ANN can be used to provide an initial guess for the optimization problem. With this mode, PD is run first, regardless of the computational demands, and is used to create a library of base simulations. These simulations are then used to train an ANN. With subsequent PD simulations, the ANN is used to provide an initial guess based on simulation parameters (i.e., avatar anthropometry, loads applied to the avatar from equipment, etc.). This initial guess then helps increase the speed of the optimization problem and thus the simulation. On the other hand, even with an appropriate (i.e., close to optimal) initial guess that is provided from the ANN, the complex nonconvex optimization in the PD algorithm is not always guaranteed to run faster.

We contend that the results of the ANN can actually be used directly as a final solution in and of itself, and subsequent results prove this hypothesis. This process in inherently faster than running PD, training an ANN, and then running PD again. Alternatively, PD is run off line to produce training data, and the ANN is run to produce final results. Task characteristics (i.e. walking speed, jumping height, etc.) and avatar characteristics provide the input for a GRNN. Joint-angle profiles, joint-torque profiles, and ground reaction forces are the output (see Equation 1). These parameters are first used to train a GRNN (one for each task). Then, conceivably, any set of input values can be used to run the GRNN and extract associated output values (simulation results). This in turn provides a new and relatively fast method for human motion prediction. Details regarding the inputs and outputs for the tasks of walking and jumping on a box are provided in the subsequent section.

3. Results

The use of GRNN to simulate dynamic human motion quickly is applied and tested on two tasks: walking forward with a backpack, and jumping up on a box. The approximate results provided by the GRNN are compared to, and verified with those provided from the source predictive-dynamics models. The work of developing/running the training cases, training the network, and then executing/running the trained network is completed on a Windows 7 computer with an Intel® Core™ 2 processor and 8 GB of RAM. Since there is no optimization procedures in the GRNN training process, it is trained within approximately 2-5 seconds, where that includes setting the network parameters to their appropriate values (as indicated in Section 2.2). With the presented tasks, the run time for PD ranges between 1 and 40 minutes. The network-predicted (approximated) motion outputs, which are all produced in a fraction of a second, are presented, evaluated, and compared with those exact (true) outputs from PD. After the network is trained with cases produced from PD, the network can simulate motions instantly for new conditions (i.e., test cases) that had never been used to train the network. To evaluate the network results of the test cases, all the produced motions from the GRNN and PD are compared visually and objectively.

3.1. Example 1: Walking

The first simulated task is walking forward with a backpack (Xiang, et al., 2009). The inputs for the task include: walking velocity, backpack weight, four lower-body link lengths (spine to hip, hip to knee, knee to ankle, and ankle to football), and three body joint ROMs. ROM is specified as upper and lower limits for the hip, knee, and ankle, each at flexion and extension. These specific joint ROMs are used (note that many others are stipulated as detailed by Marler et al., 2008), because changing their limits has significant effects on the resulting motion, whereas other ROMs have a relatively small effect on the task. Table 1 shows a typical range of values, for each input parameter, that reflect various scales of human anthropometric data. The training cases are formed as various combinations of these values, although not all permutations are used simply to reduce complexity. Velocity represents the speed of walking for Santos and is measured in m/sec. Value 1 and Value 2, respectively, typical minimum and maximum speeds for a person of average height. Each training case (set of input values) represents a point on a training grid. The consequent trained ANN can then be run using any set of inputs. Input values that do not mimic any training case represent what is called a *test case* (i.e., off-grid point).

Table 1
Input parameters for training the walking-forward task.

Input parameter	Value 1	Value 2	Value 3
Velocity (m/s)	0.8	--	1.6
Backpack weight (N)	0	175	315
Link1 (Spine to Hip) (cm)	7.8	8.8	9.8
Link2 (Hip to Knee) (cm)	43.5	44.5	45.6
Link3 (Knee to Ankle) (cm)	39.5	42.4	45.4
Link4 (Ankle to Football) (cm)	11.3	11.7	12.1
Joint1 (Hip)- lower limit (degrees)	-123.3	-105	-90
Joint1 (Hip)- upper limit (degrees)	8.7	5	2
Joint2 (Knee)- lower limit (degrees)	5	10	20
Joint2 (Knee)- upper limit (degrees)	149.7	130	110

Joint3 (Ankle)- lower limit (degrees)	7.3	15	20
Joint3 (Ankle)- upper limit	71.6	60	50

Thus, for the first training case, the velocity is set to Value 1, the backpack weight is set to Value 1, the group of avatar inputs correspond to the link lengths (the four inputs for body segments' lengths) are set to Value 1, and the group of avatar inputs correspond to the joint limits (the six inputs for upper and lower joint limits) are set to Value 1. The next training case is created by keeping the backpack weight, the velocity, and the group of inputs correspond to the link lengths fixed at Value 1, and moving the group of avatar joint limits to Value 2. Then, the avatar joint limits are set to Value 3 to create the third training case. The same process is applied to create the next 3 training cases by setting velocity and backpack weight to Value 1, link lengths to Value 2, and using joint limits iteratively from Value 1 to Value 3. The following 3 training cases are created using the previous settings, except the link lengths are set to Value 3. By the end of this step, nine training cases have been produced so far. Next, the velocity is set to Value 1, the backpack weight is set to Value 2, and the previous steps are repeated to create the next new nine training cases. Then, another nine training cases are created using Value 3 for the backpack weight. Thus far, 27 training cases have been produced after all the input values are used with fixed velocity at Value 1. Setting the velocity to Value 2 and repeating all the aforementioned procedures produces another set of 27 cases. Finally, after a manual post-processing of the resulting 54 training cases, two training cases are removed because they are subjectively unacceptable due to visually odd motions. This results in 52 training cases in total.

The network's outputs in a PD task include control points for Santos's joint-angle profiles for all 55 DOFs, as well as values for joint torques at specified time steps for certain specified DOFs. With the walking task, each joint profile consists of six control points that represent joint values at different times over the task. Hence, there are 330 outputs representing the joint angle profiles. Joint torques are considered for the six lower-joint DOFs (three for the hip, one for the knee, and two for the ankle), because these are the most highly articulated DOFs during the walking task. Assuming symmetry, the joint torques are evaluated at ten time steps during the walking task. This results in 60 additional output values. Thus, to summarize, the task is defined with 12 inputs and 390 outputs in total, and there are 52 training cases.

Once the ANN has been trained, the ANN model for the walking task is tested using the conditions shown in Table 2. Each of these testing cases is analyzed by comparing the predicted network outputs and the exact PD results. The test cases include two off-grid points. Note that these cases are not used to train the network, so they help evaluate the general performance of the network prediction for off-grid (i.e., new unseen) test cases. These cases are chosen to cover different input combinations and with input values that are completely different from the values in the training cases (Table 1). The results of the test case are evaluated based on subjective motion, and on objective evaluation of joint angle profiles and joint torques.

Table 2
Input variables for two test cases.

Input parameter	Case 1	Case 2
Velocity (m/s)	1.4	0.9
Backpack weight (N)	220	63
Link1 (Spine to Hip) (cm)	9	8
Link2 (Hip to Knee) (cm)	44	43
Link3 (Knee to Ankle) (cm)	41.4	45

Link4 (Ankle to Football) (cm)	12	11.3
Joint1 (Hip)- lower limit (degrees)	-98.6	-111
Joint1 (Hip)- upper limit (degrees)	2.2	6.5
Joint2 (Knee)- lower limit (degrees)	8	16
Joint2 (Knee)- upper limit (degrees)	146	127.2
Joint3 (Ankle)- lower limit (degrees)	12	7
Joint3 (Ankle)- upper limit	57	70

As a basic subjective analysis, the visual motion results for Case 1 and Case 2 are presented in Fig. 3. For this and all following cases, snapshots are taken at three different time frames of the total task time. The visual results for Case 1 show accurate network prediction of all joint profiles when compared to PD results; the motion produced from the network is visually comparable to that from the PD. With Case 2, Santos's back is almost straight, which is a reflection of the input backpack weight of just 63 N. The relatively short step size reflects the relatively low walking velocity. Generally speaking, the visual subjective results are all realistic.

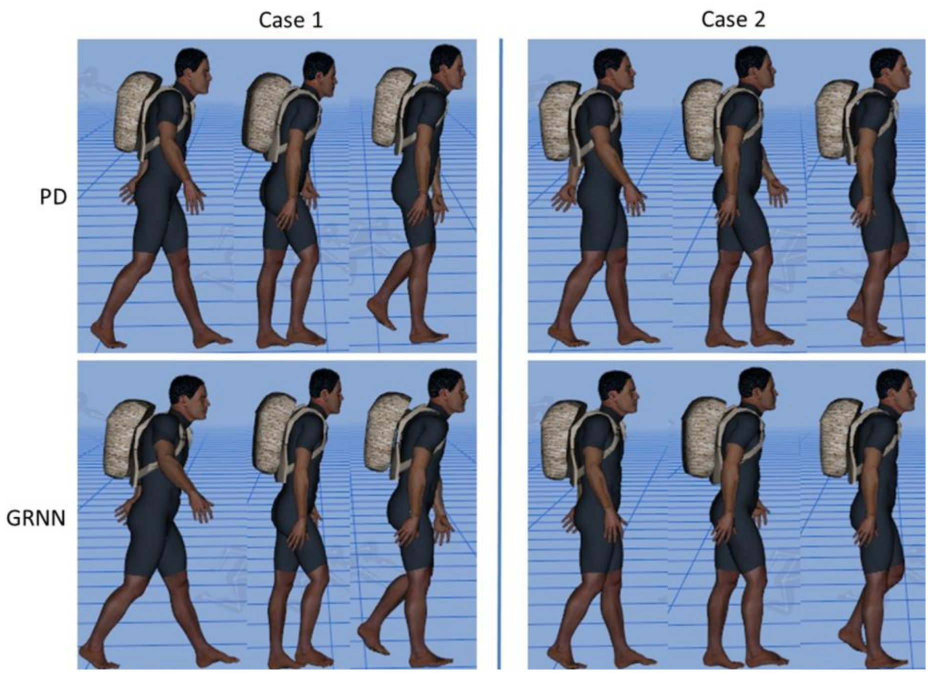

Fig. 3. Visual walking task results for test Case 1 and Case 2.

As mentioned earlier, 330 of the network's outputs represent joint angle profiles for the 55 DOFs. Statistical comparison is performed on these outputs for both testing cases. The mean-absolute error (MAE) values are calculated for joint angle profiles between the predicted values from the GRNN and the exact ones from the PD. The MAE are 0.029 and 0.033 for Case 1 and Case 2, respectively, which are relatively small, thus indicating agreement between the ANN results and the physics-based results (using predictive dynamics).

As another objective test for the results, adjusted R-square values are calculated between the predicted GRNN results and the exact PD ones (Fig. 4). In both cases, accurate results are achieved; the R-squared values are above 0.99, and the visual results show minimal discrepancies. The slight increase in R-squared for Case 1 is simply the result of the neural-network hypersurface providing slightly more accurate results for different points (sets of inputs). Interestingly, the scatter plot for Case 1 appears to suggest a decrease in accuracy.

However, the MAE result for this case provides the evidence for that accuracy level that matches the resulting R-squared value. Both results confirm the high correlation between the predicted and exact results. In addition, given that there are 330 data points, the number of points that visually deviate from the line in Case 1 is relatively small and numerically insignificant. In other words, in Case 1, the network produces different solutions for some of the joint angles that have minimal effect on the total motion behavior.

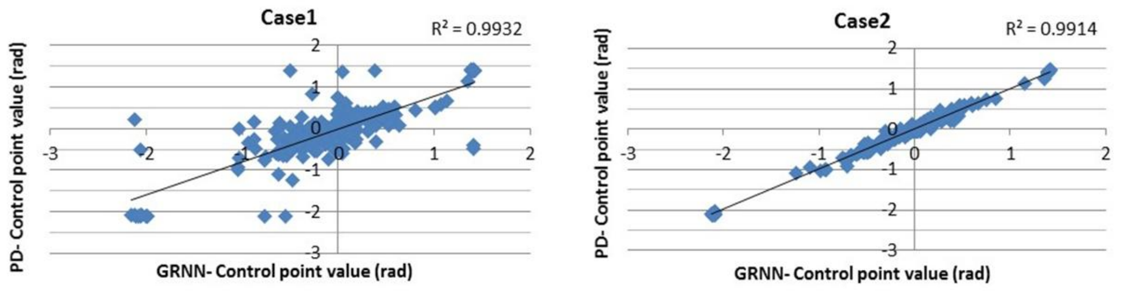

Fig. 4. Statistical plots and adjusted R-square values for the produced joint angle profiles in test Case 1 and Case 2 in the task of walking forward.

As with joint angle profiles, the MAEs are calculated for the produced joint-torque profiles. The results are 10.01 for Case 1 and 9.16 for Case 2. The adjusted R-square values are also plotted and calculated (see Fig. 5). Recall that 60 of the output values relate to torque profiles for specified DOFs. The values are approximately 0.96 and 0.88 for Cases 1 and 2, respectively. These calculated values are relatively high and generally acceptable. They are slightly different from those in Fig. 4, in part because the scale of the output is different, which can affect the accuracy of results produced from the network. Although inputs are already scaled, future work might include investigating the effects of normalizing output values during the training process.

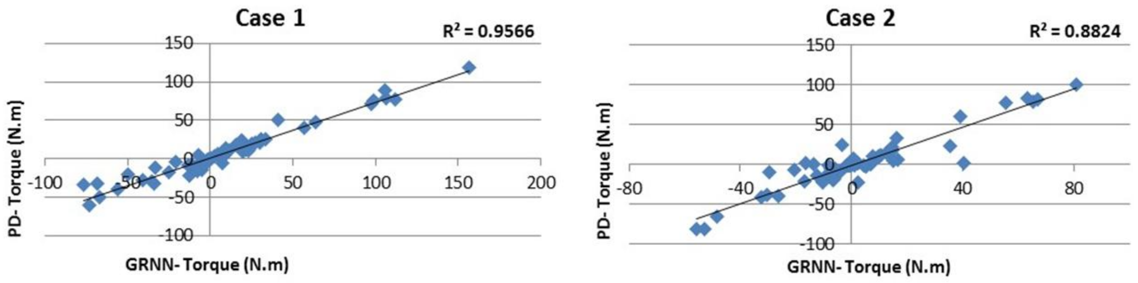

Fig. 5. Statistical plots and adjusted R-square values for the produced joint torque profiles in test Case 1 and Case 2 in the task of walking forward.

On the other hand, these MAEs and R-square results are not quite as accurate as those for the joint angles, because each joint has a large range of torque values compared to the joint angle values. Moreover, there are many ranges for torque values at different joints, while the joint angles were all measured in radian (rad) and fell between +6.28 rad to -6.28 rad. Therefore, the network can predict the joint angle values, which are consistent for all DOFs, with less error compared to the variety joint torque range of values.

3.2. Example 2: Jumping on a Box

The second example task for evaluating GRNN performance is jumping up on a box (Fig. 6). This is a relatively complex task that has many constraints (i.e., feet and hands should be located at specific places) and requires highly accurate results. Box height is the primary task-based input, because it has the greatest effect on the resulting motion profile. When training the GRNN, the box height ranges between 0.5 meter and 1 meter. Avatar-based inputs include four link lengths: spine-hip, hip-knee, knee-ankle, and ankle-football. Minimum and maximum values for these inputs are shown in Table 3. In order to generate training cases, first input values are set at their minima, and box height is increased in increments of 5 cm with all other parameters fixed. Then, input values are set at their maxima, and box height is decreased in increments of 5 cm. This results in 22 training cases.

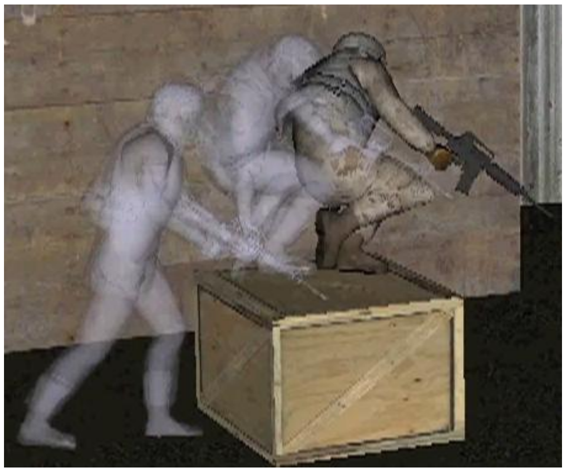

Fig. 6. Task of jumping up on a box.

Table 3
Input parameters and training values for the task of jumping up on a box.

Input parameter	Minimum	Maximum
Box height (cm)	50	100
Link1 (Spine to Hip) (cm)	7.8	9
Link2 (Hip to Knee) (cm)	38	43
Link3 (Knee to Ankle) (cm)	39	39
Link4 (Ankle to Football) (cm)	9	12

The jumping task has two types of outputs: joint splines and ground reaction forces (GRF) for both feet. The total number of outputs is 370, with 330 outputs for joint splines and 40 outputs for GRFs.

As with the walking example, two off-grid testing cases (Table 4) are evaluated and analyzed visually and statistically. The visual results for Case 1 and Case 2 are compared in Fig. 7: the left and right parts in each case represent motion segments over the task time for PD and GRNN, respectively. In Case 1, even though there were some differences for both motions due to the recording procedures of the snapshots, the motions had the same behavior over the task time, including the height of the feet and the hand positions. In Case 2, Santos's hands and feet from the network's predicted motion were also at the exact locations that the PD results provided.

Table 4
Input variables for two test cases.

Input parameter	Case 1	Case 2
Box height (cm)	0.68	0.92
Link1 (Spine to Hip) (cm)	8.2	8.8
Link2 (Hip to Knee) (cm)	40	42
Link3 (Knee to Ankle) (cm)	39	39
Link4 (Ankle to Football) (cm)	10	11

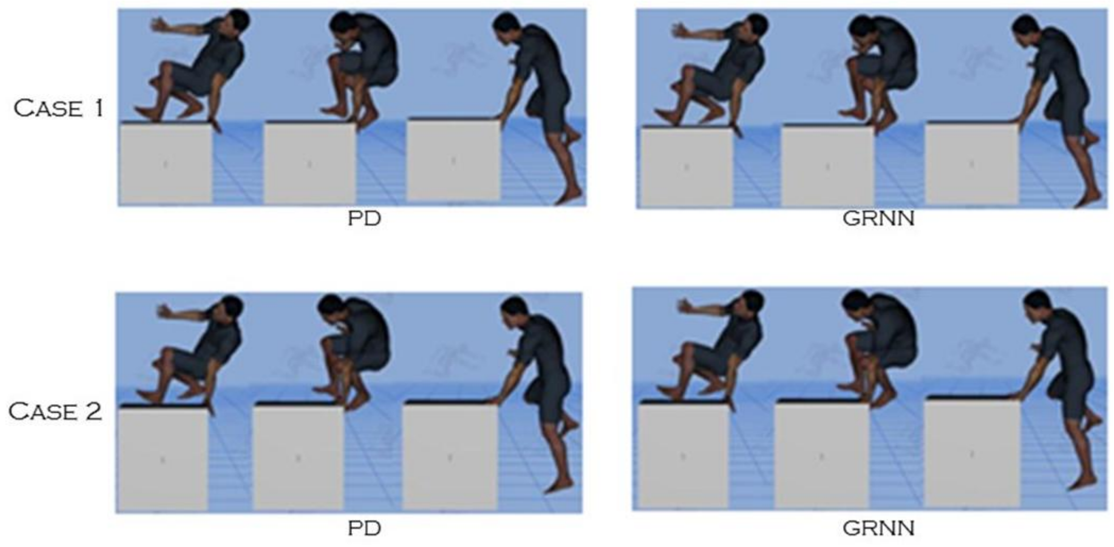

Fig. 7. Visual results for test Case 1 and Case 2 in the task of jumping up on a box.

With respect to statistical evaluation, comparison between predicted GRNN outputs and exact PD ones is performed for both testing cases. The MAE values are calculated, and the results show small values with 0.017 and 0.015 for Case 1 and Case 2, respectively.

In terms of adjusted R-square values, the results are shown in Fig. 8. The R-square values for angle profiles are approximately 1 for both testing cases, which suggests high accuracy of motion prediction produced from the GRNN.

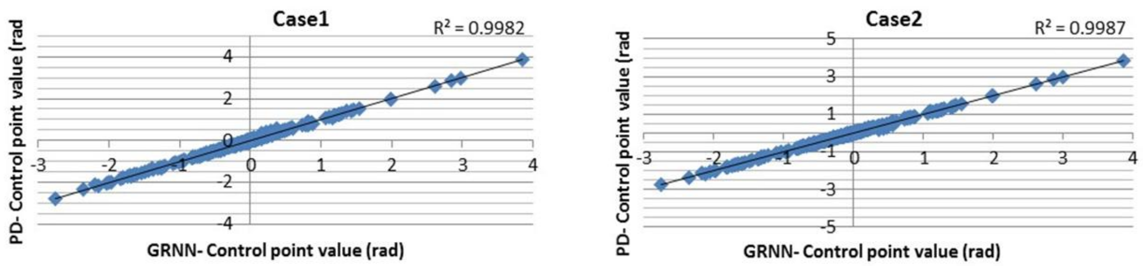

Fig. 8. Statistical plots and adjusted R-square values for the produced joint angle profiles in test Case 1 and Case 2 in the task of jumping up on a box.

In terms of the GRNN prediction results for the GRFs, the results comparison with those provided from the PD show high R-square values for both Case 1 and Case 2 (Fig. 9). In addition, the MAE results for the GRF are calculated and the results are 0.38 and 0.53 for Case 1 and Case 2, respectively. In general, the statistical results suggest acceptable network-predicted solutions for the GRFs, and such results are enhanced by the relatively small simulation errors.

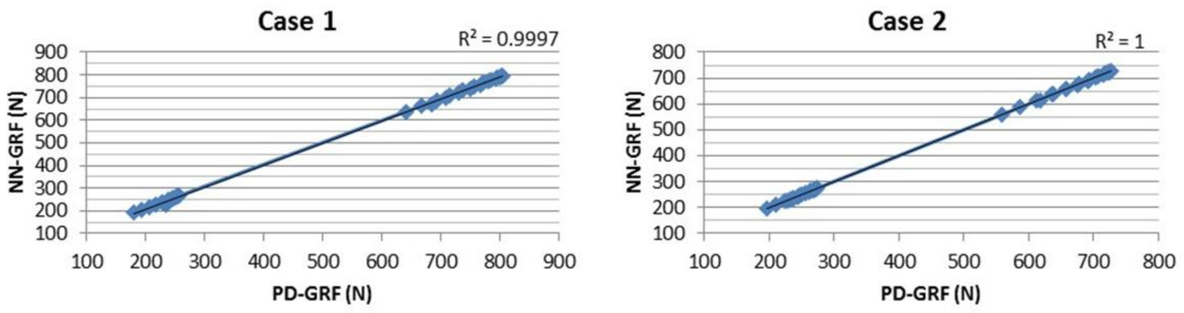

Fig. 9. Statistical plots and adjusted R-square values for the produced ground reaction forces in test Case 1 and Case 2 in the task of jumping up on a box.

4. Discussion

This work has demonstrated the successful use of ANNs for simulating dynamic human motion in real time, based on a library of physics-based simulations. The use of an ANN is shown to provide real-time motion prediction for a full-body DHM, using various tasks. The GRNN, in particular, is applied successfully as a powerful tool that can be trained quickly and without any memory problems, regardless of the size of the problem. This provides a new variant of physics-based human simulation with increased computational speed, which in turn allows one to conduct trade-off analyses more efficiently for evaluating products and processes form a human system integration perspective.

As with any regression analysis or meta-model, the approximate solution is expected to deviate at least some minimal amount from the source model. We show that this deviations is acceptable when evaluated both objectively and subjectively. As is often the case with digital human modeling, although subjective results may seem identical when compared, quantitative results may differ slightly. Nonetheless, the ANN model performed well when compared to the predictive-dynamics source model.

With the use of ANNs for motion prediction, the main issue that was successfully addressed is the computational time of producing PD outputs. The time is decreased from minutes to a fraction of a second. Even with a task that requires highly accurate results like the jumping on a box one, the simulation accuracy, as demonstrated, was preserved. Accuracy in such task is critical, especially where the hands and feet should be in contact with the box at some point during the task, and network results showed visual satisfaction of such constraints.

Conceptually, the proposed process reflects how people actually learn and perform tasks. One’s reaction to specific scenarios is an interpolation, in part, of previously experienced scenarios. To be sure, rational thought and cognitive extrapolation play a role. However, the proposed method demonstrates a model for incorporating a learned history into task simulation. In fact, it provides a platform with which one can study how various experiences can affect performance and how variations in one’s learning set can alter a simulation. This has practical applications to the study and design of training regiments and education.

This initial investigation into the use of GRNN for human modeling has presented some opportunities for future work. First, as is the case with most ANN applications, work is needed to determine the optimal number of training cases for a task. In order to ensure contact constraints are satisfied, future work should include adding constraints to the network construction. In addition, we propose training the network to predict joint-center locations instead of joint angles, in order to produce more accurate results. Predicting joint angles with even small errors can sometimes produce inaccurate results, whereas predicting joint centers with small error should still provide acceptable results. Although the network is trained using simulations (predictive dynamics) that are presumably validated, and although this work presents the *feasibility* of the proposed method based on subjective validation, predictions from the actual ANN should be validated directly and objectively. Finally, as suggested with respect to joint torques, further work is needed to investigate the benefits of normalizing outputs during the training process.

Acknowledgments
This work is funded by the Office of Naval Research (ONR). The authors would also like to thank the team at The University of Iowa's Virtual Soldier Research (VSR) Program.

References

Abdel-Malek, K., Arora, J., Yang, J., Marler, T., Beck, S., Swan, C., Frey-Law, L., Mathai, A., Murphy, C., & Rahmatalla, S. (2006). Santos: A physics-based digital human simulation environment. In *Proceedings of the Human Factors and Ergonomics Society Annual Meeting* (Vol. 50, pp. 2279-2283): SAGE Publications.

Abdel-Malek, K., Yang, J., Kim, J. H., Marler, T., Beck, S., Swan, C., Frey-Law, L., Mathai, A., Murphy, C., & Rahmatallah, S. (2007). Development of the virtual-human SantosTM. In *Digital Human Modeling* (pp. 490-499): Springer.

Arora, J. S. (2004). Intriduction to Optimum Design. 2nd Edition, Elsevier Academic Press, Amsterdam.

Bataineh, M. (2012). Artificial neural network for studying human performance. *M.S. thesis, University of Iowa*.

Bataineh, M. (2015). New neural network for real-time human dynamic motion prediction. *Ph.D. thesis, University of Iowa*.

Bataineh, M., & Marler, T. (2013). Neural networks for performance-measure selection. In *2nd International Digital Human Modeling Symposium*. Ann Arbor, MI-USA.

Bataineh, M., Marler, T., & Abdel-Malek, K. (2013). Artificial neural network-based prediction of human posture. In *Digital Human Modeling and Applications in Health, Safety, Ergonomics, and Risk Management. Human Body Modeling and Ergonomics* (pp. 305): Springer.

Bu, N., Okamoto, M., & Tsuji, T. (2009). A hybrid motion classification approach for EMG-based human–robot interfaces using bayesian and neural networks. *Robotics, IEEE Transactions on, 25*, 502-511.

Chaffin, D. B. (2002). On simulating human reach motions for ergonomics analyses. *Human Factors and Ergonomics in Manufacturing & Service Industries, 12*, 235-247.

Coit, D. W., Jackson, B. T., & Smith, A. E. (1998). Static neural network process models: considerations and case studies. *International Journal of Production Research, 36*, 2953-2967.

Frank, R. J., Davey, N., & Hunt, S. P. (2001). Time series prediction and neural networks. *Journal of Intelligent and Robotic Systems, 31*, 91.

Isaksson, M., Jalden, J., & Murphy, M. J. (2005). On using an adaptive neural network to predict lung tumor motion during respiration for radiotherapy applications. *Medical physics, 32*, 3801.

Jung, E. S., & Park, S. (1994). Prediction of human reach posture using a neural network for ergonomic man models. *Computers & industrial engineering, 27*, 369-372.

Kang, B., Kim, B., Park, S., & Kim, H. (2007). Modeling of artificial neural network for the prediction of the multi-joint stiffness in dynamic condition. In *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on* (pp. 1840-1845): IEEE.

Kim, J. H., Xiang, Y., Bhatt, R., Yang, J., Chung, H.-J., Patrick, A., Arora, J. S., & Abdel-Malek, K. (2008). Efficient ZMP formulation and effective whole-body motion generation for a human-like mechanism. In *ASME 2008 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference* (pp. 1073-1084): American Society of Mechanical Engineers.

Kim, J. H., Xiang, Y., Yang, J., Arora, J. S., & Abdel-Malek, K. (2010). Dynamic motion planning of overarm throw for a biped human multibody system. *Multibody System Dynamics, 24,* 1-24.

Kim, Y. H., & Lewis, F. L. (1999). Neural network output feedback control of robot manipulators. *Robotics and Automation, IEEE Transactions on, 15,* 301.

Kwon, H.-J., Xiang, Y., Bhatt, R., Rahmatalla, S., Arora, J. S., & Abdel-Malek, K. (2014). Backward walking simulation of humans using optimization. *Structural and Multidisciplinary Optimization,* 1-11.

Li, Y., Li, C., & Song, R. (2007). A new hybrid algorithm of dynamic obstacle avoidance based on dynamic rolling planning and RBFNN. In *Robotics and Biomimetics, 2007. ROBIO 2007. IEEE International Conference on* (pp. 2064-2068): IEEE.

Marler, T., Arora, J., Beck, S., Lu, J., Mathai, A., Patrick, A., Swan, C. (2008), "Computational Approaches in DHM," in *Handbook of Digital Human Modeling for Human Factors and Ergonomics*, Vincent G. Duffy, Ed., Taylor and Francis Press, London, England.

Moeslund, T. B., Hilton, A., & Krüger, V. (2006). A survey of advances in vision-based human motion capture and analysis. *Computer vision and image understanding, 104*, 90-126.

Rahmatalla, S., Xiang, Y., Smith, R., Meusch, J., & Bhatt, R. (2011). A validation framework for predictive human models. *International journal of human factors modelling and simulation, 2*, 67-84.

Specht, D. F. (1991). A general regression neural network. *Neural Networks, IEEE Transactions on, 2*, 568.

Stakem, F., & AlRegib, G. (2008). Neural Networks for Human Arm Movement Prediction in CVEs. In *Proceedings of 3DPVT.*

Twomey, J. M., & Smith, A. E. (1998). Bias and variance of validation methods for function approximation neural networks under conditions of sparse data. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on, 28*, 417-430.

Wasserman, P. D. (1993). *Advanced methods in neural computing*: John Wiley & Sons Inc.

Xiang, Y., Arora, J. S., & Abdel-Malek, K. (2010). Physics-based modeling and simulation of human walking: a review of optimization-based and other approaches. *Structural and Multidisciplinary Optimization, 42*, 1-23.

Xiang, Y., Arora, J. S., & Abdel-Malek, K. (2012). Hybrid predictive dynamics: a new approach to simulate human motion. *Multibody System Dynamics, 28*, 199-224.

Xiang, Y., Arora, J. S., Rahmatalla, S., & Abdel-Malek, K. (2009). Optimization-based dynamic human walking prediction: One step formulation. *International Journal for Numerical Methods in Engineering, 79*, 667-695.

Xiang, Y., Arora, J. S., Rahmatalla, S., Marler, T., Bhatt, R., & Abdel-Malek, K. (2010). Human lifting simulation using a multi-objective optimization approach. *Multibody System Dynamics, 23*, 431-451.

Xiang, Y., Chung, H.-J., Kim, J. H., Bhatt, R., Rahmatalla, S., Yang, J., Marler, T., Arora, J. S., & Abdel-Malek, K. (2010). Predictive dynamics: an optimization-based novel approach for human motion simulation. *Structural and Multidisciplinary Optimization, 41*, 465.

Yoo, J.-H., Hwang, D., Moon, K.-Y., & Nixon, M. S. (2008). Automated human recognition by gait using neural network. In *Image Processing Theory, Tools and Applications, 2008. IPTA 2008. First Workshops on* (pp. 1): IEEE.

Zhang, B., Horváth, I., Molenbroek, J. F., & Snijders, C. (2010). Using artificial neural networks for human body posture prediction. *International Journal of Industrial Ergonomics, 40*, 414-424.

Zhao, H., Zheng, G., & Wen, W. (2010). Human-machine posture prediction and working efficiency evaluation of virtual human using radial basis function neural network. In *Intelligent Computing* *and Intelligent Systems (ICIS), 2010 IEEE International Conference on* (Vol. 1, pp. 406-410): IEEE.