DYNAMIC SIMULATION OF MOVEMENT BASED ON OPENSIM AND MATLAB®/SIMULINK®

Misagh B. Mansouri and Jeffrey A. Reinbolt

University of Tennessee, Knoxville, TN, USA
email: misagh@utk.edu, web: http://rrg.utk.edu

INTRODUCTION

Numerical simulations are playing an important role in solving complex problems and have the potential to revolutionize medical decision making and treatment strategy. Musculoskeletal conditions cost the U.S. economy alone over $849 billion per year (7.7% of the U.S. Gross Domestic Product) and place great demands on healthcare systems worldwide [1]. This area could greatly benefit from computational tools offering greater understanding of human movement and predictive capabilities for optimal treatment planning.

Currently there is no freely available, open-source computational tool providing an interface between software packages for robust design and control as well as modeling and simulating neuromusculoskeletal systems. Others [2] have tools to convert musculoskeletal and kinetics models from commercial software [3] to Simulink® blocks. Simulink® extends MATLAB®, the leading mathematical computing software for engineers and scientist, with a graphical environment for rapid design, control, and simulation of dynamic systems. However, this package has limited resources for modeling and simulating neuromusculoskeletal systems. On the contrary, OpenSim [4] is a popular open-source platform for modeling, simulating, and analyzing these systems, but it is lacks the robust design and control tools of MATLAB®/Simulink®.

Our goal was to develop an interface between MATLAB®/Simulink® and OpenSim that combines their relevant strengths (e.g., rapid model-based design, control systems, numerical simulation, and human movement dynamics). The MATLAB®/Simulink® and OpenSim integrated platform contributes to the overall understanding of human movement and has the potential to improve surgical and rehabilitation treatment planning.

METHODS

We developed and demonstrated the MATLAB®/ Simulink® and OpenSim interface (Fig. 1) using a three step process.

Figure 1: New S-function (system function) interface linking rapid design and control tools of MATLAB®/Simulink® with neuromusculoskeletal dynamic simulation tools of OpenSim.

First, we developed a MATLAB® S-function (system function) based on an OpenSim model as a Simulink® block. This block interacts with the Simulink® engine, similar to built-in Simulink® blocks. The S-function works with any OpenSim model regardless of the number of input controls (e.g., muscle excitations or joint torques) or output states (e.g., kinematics).

Second, we created a generic open-loop Simulink® model that loads and executes the OpenSim-based S-function developed in the first step (Fig.2). To demonstrate the open-loop characteristics of this model, we used a simple human arm model with 2 degrees of freedom and 6 muscle-tendon actuators. Kinematics resulting from a Simulink® simulation of elbow flexion were directly compared with those from OpenSim forward dynamics.

Figure 2: Generic open-loop Simulink® model using the OpenSim-based S-function interface to generate output states (e.g., kinematics) from input controls (e.g., muscle excitations or joint torques).

Third, we created an example closed-loop Simulink® model by adding a proportional, integral, and derivative (PID) controller with feedback to extend the open-loop model created in the second step (Fig. 3). The open-loop model is limited to using fixed controls that cannot be changed by the resulting motion. On the other hand, many human movement applications require closed-loop control systems. To demonstrate the closed-loop characteristics of this Simulink® model, we used a human arm model balancing a pole free to rotate about the hand. This biomechanical system had 2 degrees of freedom, a constraint on elbow angle as a function of shoulder angle to reduce the number of controls, and 1 shoulder torque actuator.

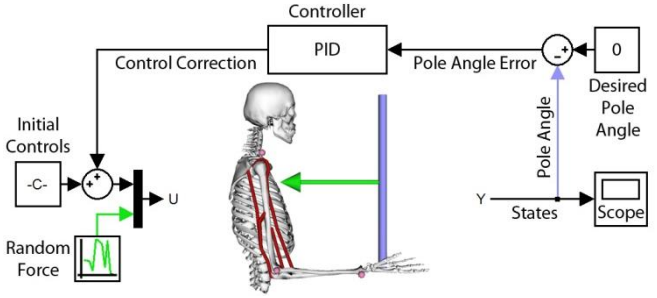

Figure 3: Example closed-loop Simulink® model extending the open-loop case with proportional, integral, and derivative (PID) control of a human arm balancing a pole despite force disturbances.

A PID controller was used to balance a pole above the hand by adjusting the torque at the shoulder (Fig. 3 & 4a). The controller gains were tuned using the classic Ziegler–Nichols method. The desired pole angle (measured from vertical) was 0 and it was used to compute a pole angle error and then a control correction. The initial controls, computed from inverse dynamics, were used to maintain arm position before control correction was necessary. The combination of control correction and initial controls is analogous (but not identical) to the human brain's output for the arm to balance a pole. Random force disturbances were added to the pole and the pole angle error was observed.

RESULTS AND DISCUSSION

The new interface between MATLAB®/Simulink® and OpenSim allowed rapid model-based design and numerical simulation of human movement using both open-loop (Fig. 2) and closed-loop (Fig. 3) control systems. For the open-loop case, Simulink® generated elbow flexion angle matched OpenSim within 0.39º RMS. This result indicated an OpenSim model and its controls behave similarly in Simulink® and OpenSim. For the closed-loop case, the PID controller successfully rejected random force disturbances (Fig. 4b) and balanced the pole above the hand with a maximum pole angle from vertical of 0.0177º (Fig. 4c).

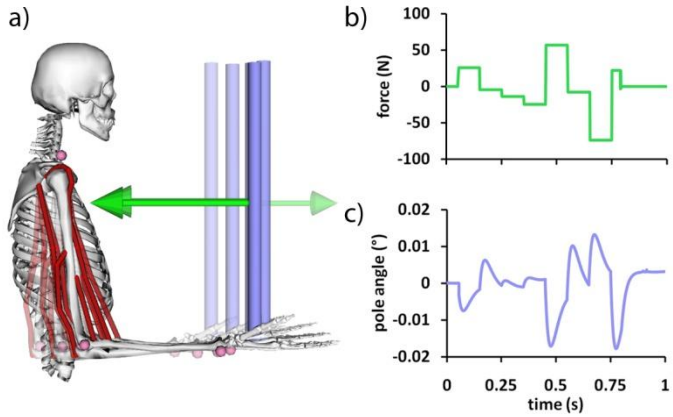

Figure 4: Dynamic simulation generated in Simulink® of (a) PID controlled human arm model balancing a pole (five time frame series from 0.54s to 0.74s shown). (b) Random force disturbances were added to the pole and (c) the pole angle measured from vertical remained small.

The potential to use integrated computational tools to better understand human movement and optimally design treatments is exciting. We have posted all source code, Simulink® model examples, and user documentation related to this work on a SimTK.org project dedicated to the interface (https://simtk.org/home/opensim_matlab). Once this project is complete and made public, it will not only integrate software tools, but also allow integration of neuroscientists, physiologists, biomechanists, and physical therapists to adopt, adapt, and generate new solutions for musculoskeletal conditions.

REFERENCES

1. Jacobs JJ, et al., *Burden of Musculoskeletal Diseases in the US*, 1st ed. AAOS, 2008.
2. Davoodi R, et al., *Med. Eng. Phys.*, **25**, 3–9, 2003.
3. Delp SL and Loan JP, *Comp. Science Eng.*, **2**, 46-55, 2000.
4. Delp SL, et al., *IEEE Trans. Biomed. Eng.*, **54**, 1940-1950, 2007.

ACKNOWLEDGEMENTS

Supported by a subaward from NIH Roadmap for Medical Research U54 GM072970.