Musculoskeletal Mechanics and Modeling

This article has not yet been published; it may contain inaccuracies, unapproved changes, or be unfinished.

- **Dr. Gerald E. Loeb**, University of Southern California, Los Angeles, CA, USA
- **Prof. Rahman Davoodi**, Research Assistant Professor, Los Angeles, California, USA

Dr. Gerald E. Loeb accepted the invitation on 15 October 2011

Contents

Introduction

A large part of our central nervous system is dedicated to control of movement. To gain a fuller understanding of the sensorimotor control of movement, we must study both the central nervous system and the musculoskeletal system that it controls. A control engineer would never design a control system before fully understanding the characteristics of the controlled plant. Similarly, the biological motor control system cannot be understood fully by studying the control circuits while ignoring the inherent properties of the musculoskeletal system that it must control. Models that capture the mechanical dynamics of the musculoskeletal system, especially when they are combined with the models of the control circuits in the central nervous system, allow us to study the control of movement in its entirety.

Models of Musculoskeletal Mechanics

To control the movement of the human body, the central nervous system generates neural commands to activate the contractile apparatus of the muscles. The forces generated by the muscles combine with inertia and external forces that may be acting on the body, resulting in observable movements. The forces and movements of the musculoskeletal system are measured by various proprioceptive sensors and relayed back to the central nervous system, providing it with the information it needs to make appropriate control decisions and adjustments (Figure 1).

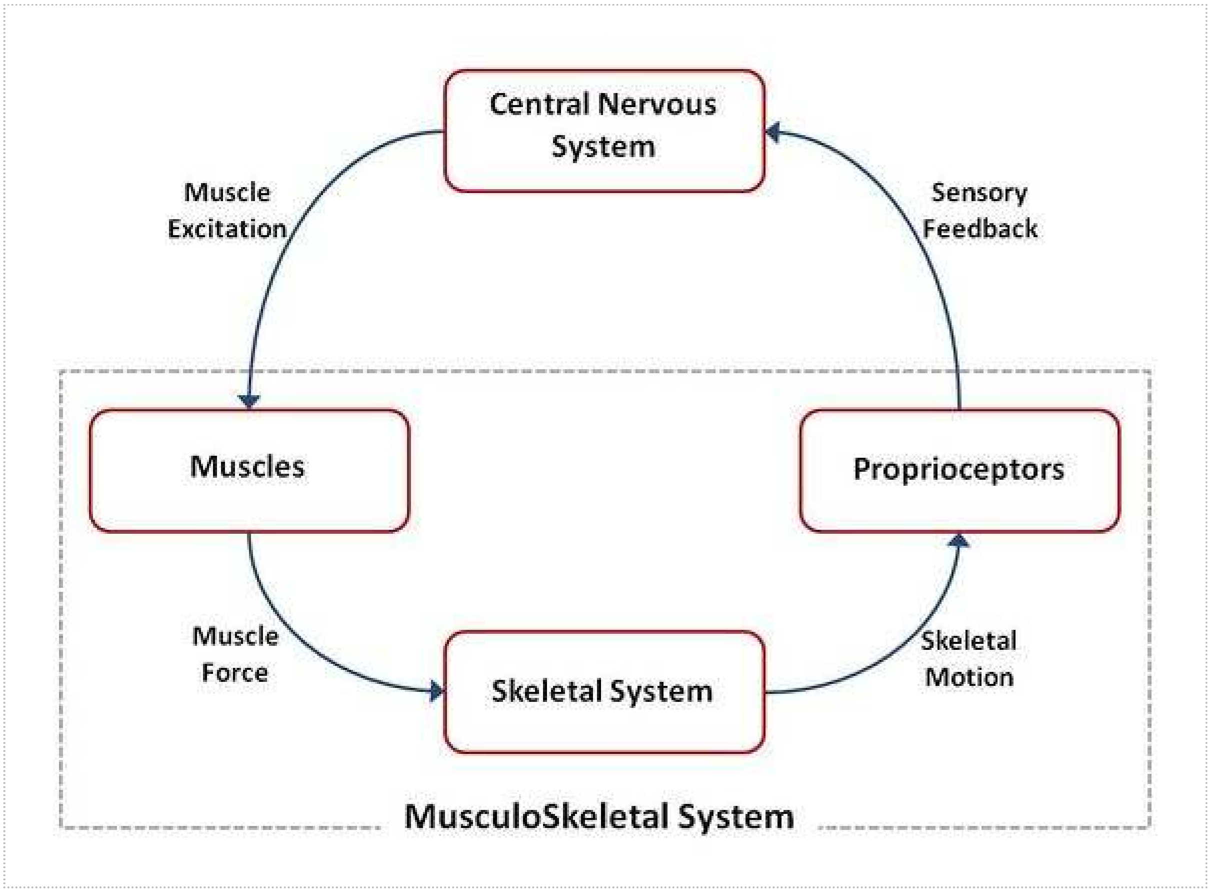


Figure 1: Human movement is the result of interactions between the central nervous system and the musculoskeletal system

Complete models of the musculoskeletal mechanics must therefore represent the dynamics of muscle force production, the dynamics of transduction in proprioceptors, and the dynamics of movement in the skeletal system. Models of muscles and proprioceptors are described in accompanying Scholarpedia articles "Muscle Physiology and Modeling" and "Proprioceptors and Models of Transduction", respectively. Here we will focus on models of the skeletal system.

Movement in Skeletal Systems

The skeletal system is composed of bones (aka rigid bodies, segments, or body segments) that are connected to each other via articulations or joints. The articulated bodies form a tree-like multibody system such as those in the arm and hand whose motion is influenced by external forces (applied by the muscles, gravity, and the environment) and the constraints imposed by the joints (Figure 2). The configuration of the body segments and the type of articulations determine all possible movements a skeletal system is capable of performing; a specific movement is determined by a specific combination of external forces. We can therefore describe the modeling of the skeletal mechanics in two parts: models of skeletal mechanics and models of external forces.

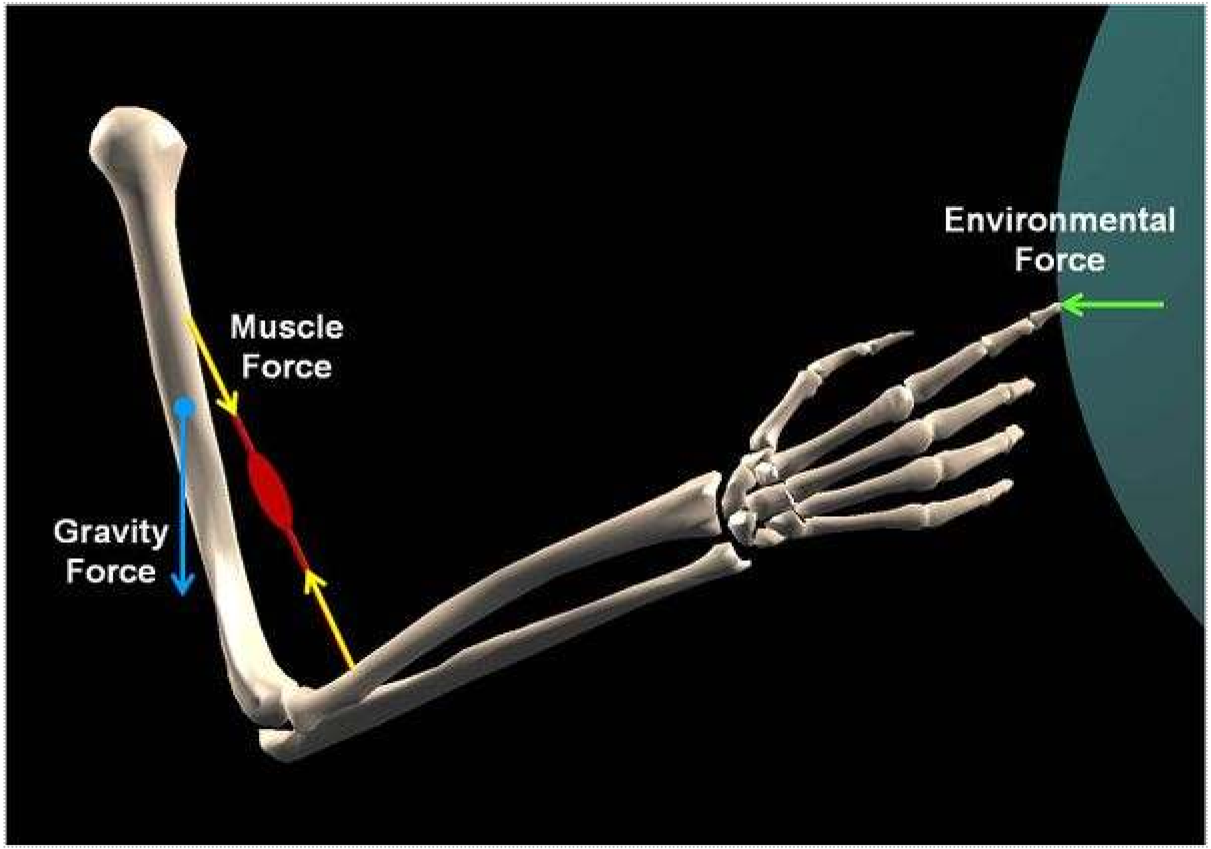


Figure 2: Skeletal system is a tree-like multibody system whose movement is influenced by the constraints imposed by its articulations and the external forces applied by the muscles, gravity, and the environment.

Skeletal Mechanics

Because the skeletal system is a multibody system, its movement can be modeled by mechanical dynamic formulation methods that were developed for other multibody systems such as those in robots, automobiles, and spacecraft (Erberhard & Schielen, 2006; Unda, de Jalon, Losantos, & Emparantza, 1987). To simulate the movement of the skeletal system, one must derive the equations that govern its motion and then solve the resulting equations of motion.

Newton-Euler method (Nikravesh, 1988) is probably the most straightforward multibody formulation method. In the first step of applying this method, the segments of the multibody system are treated independently assuming that they can have all possible translational and rotational movements (Figure 3). The application of Newton-Euler formulation results in differential equations that govern the translational and rotational movements of the body segments:

$$m\ddot{r} = f$$

$$J\ddot{\gamma} = \tau - \tilde{\dot{\gamma}}J\dot{\gamma} \tag{1}$$

Where, m and J are the mass and mass moments of inertia of the body segments, r and γ are the translations and rotations of the body segment in Cartesian coordinate system, f and τ are the external forces and moments applied to the body segments, and $\tilde{\dot{\gamma}}$ is the skew-symmetric matrix of $\dot{\gamma}$.

To ensure that the motion occurs only on the degrees of freedom allowed by the joints, the second step of formulation imposes joint constraints on the unconstrained equations of motion above. This is accomplished by additional algebraic equations that depend on the type of the joints in the system. For example, the algebraic constraints equations for a spherical joint connecting two bodies will simply require that there be a point P_1 on or attached to the first body and a point P_2 on or attached to the second body that must always be coincident at the center of the joint (Figure 3). This condition can be expressed by the following distance formula:

$$d = r_{p1} - r_{p2} = 0 \tag{2}$$

Where, r_{p1} and r_{p2} are the positions of the points in Cartesian coordinate system and d is the distance between the points.

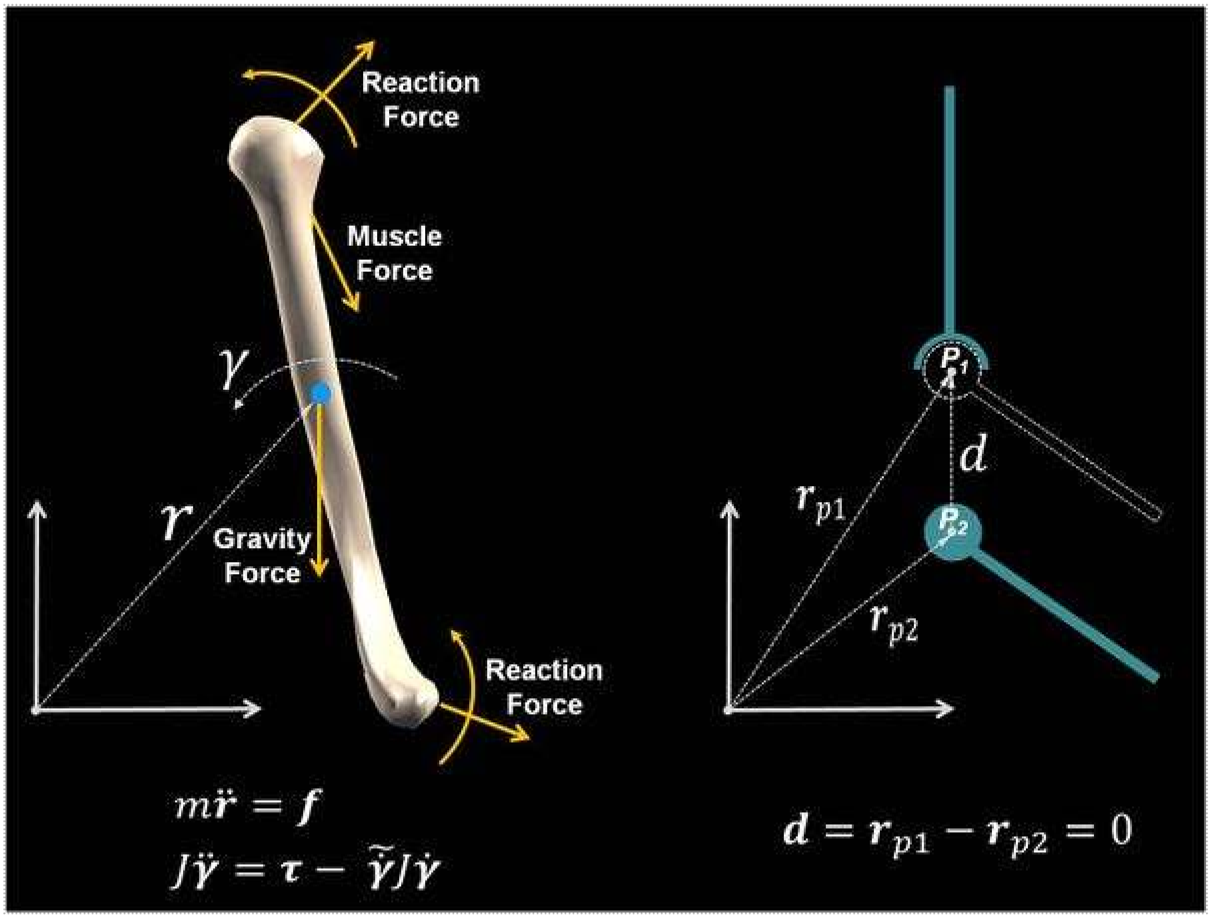


Figure 3: Application of Newton-Euler formulation involves two steps. First, equations of motion for each body segment are derived assuming that it can move freely but is subjected to external forces (left). Additional constraint equations are then imposed to ensure that the motion does not violate the joint constraints. For example, constraint equations to ensure that two body segments remain attached via a spherical joint are shown (right).

The application of the Newton-Euler and constraint equations results in a large set of mixed differential-algebraic equations including six equations of motion for each segment plus the constraint equations. In addition, the movement in these equations is expressed in Cartesian coordinate system, which is neither intuitive nor common for describing the movement in skeletal systems. Joint coordinate system, which expresses the movement as the relative motion of the neighboring body segments, is more intuitive and more common in the movement science community. By the use of the coordinate transformation and coordinate reduction methods, the mixed differential-algebraic equations in Cartesian coordinate system can be transformed to a smaller set of second order ordinary differential equations in joint coordinate system (Jerkovsky, 1978; Kim & Vanderploeg, 1986; Nikravesh, 1990) with the following general structure:

$$M(\theta)\ddot{\theta} + C(\theta, \dot{\theta}) + G(\theta) = F \qquad (3)$$

Where θ is the movement in joint coordinate system, $M(\theta)$ is the mass matrix, C is the Coriolis and centripetal terms, G is the gravity term, and F is the external forces and moments.

Once the equations of motion are derived, they must be solved to yield the motion of the skeletal system. Because the resulting set of equations is complex, usually no closed-form (analytic) solution can be found. These equations are almost always solved by the use of numerical integration that starts with known initial conditions at time zero (t_0) and steps through time to approximate the movement at discrete times $t_1, t_2, \dots t_n$ (Hairer, Norsett, & Wanner, 1993; Hairer & Wanner, 1996). At the start of integration where the initial conditions including positions and velocities are known, every term in the left side of Eq. (3) other than $\ddot{\theta}$ can be computed. Replacing these computed values and the known external forces (see the next section) in Eq. (3) yields the accelerations $\ddot{\theta}$ that can be integrated twice to obtain velocities $\dot{\theta}$ and positions θ at the end of the first time step (also the start of the next time step). The availability of the positions and velocities at the start of the next time step enables the numerical integrator to take subsequent integration steps to predict the movement of the skeletal system over time.

The traditional formulations presented above require computations that increase as a cubic function of the number of degrees of freedom in the multibody system. The development of recursive Order-n formulations, for which the computations increase linearly with the degrees of freedom, has greatly improved the computational efficiency of multibody simulations (Anderson, 1992; Featherstone, 1987). Ongoing research attempts to find formulation and solution methods that are even easier to implement and can simulate increasingly complex multibody systems at higher speed.

External Forces

As shown above, external forces are the inputs to and are required for the solution of the skeletal equations of motion. These forces are quantified by their magnitude, direction, and their point of application on the skeletal system. The gravitational force of each segment applies to its center of gravity and is directed toward the center of the earth (Figure 2); because the human or animal movements occur in relatively small spaces, the magnitude of the gravitational force can be assumed to be constant. The environmental forces arise when the skeletal system contacts the environment such as foot-ground and hand-tool contacts (Figure 2). These forces are modeled by equations that represent the dynamics of the contact between the body segment and the environment. The calculation and the application of a given muscle force is more complicated. The correct calculation and application of muscle force to the skeletal system requires knowledge of the correct path of the muscle from its origin to its insertion (Figure 4).

As the muscle stretches from its origin to insertion, it has to negotiate its path around bony surfaces and other muscles and soft tissue, resulting in a curved path (as opposed to a straight-line path from origin to insertion). The correct identification of the musculotendon path for all possible skeletal postures is essential for the correct modeling of the muscle force and its application to the skeletal system. Furthermore, the actual muscle force depends strongly on the musculotendon length and velocity, which can be obtained from skeletal posture only if the musculotendon path is known (see the accompanying article “Models of Muscle Physiology”). The muscle applies forces to the skeletal system wherever they come in contact with each other. These points of contact can be identified from the correct modeling of the musculotendon path as it comes in contact with bony surfaces and other tissue along the path. The direction of the muscle forces at the contact points depends on the path of the muscle on either side of those points. Again, accurate modeling of the direction of the muscle force also depends on accurate modeling of the musculotendon path. Modeling the musculotendon path as a straight line is frequently inaccurate, resulting in incorrect muscle forces and incorrect computation of how those forces are actually applied to the skeleton (Figure 4).

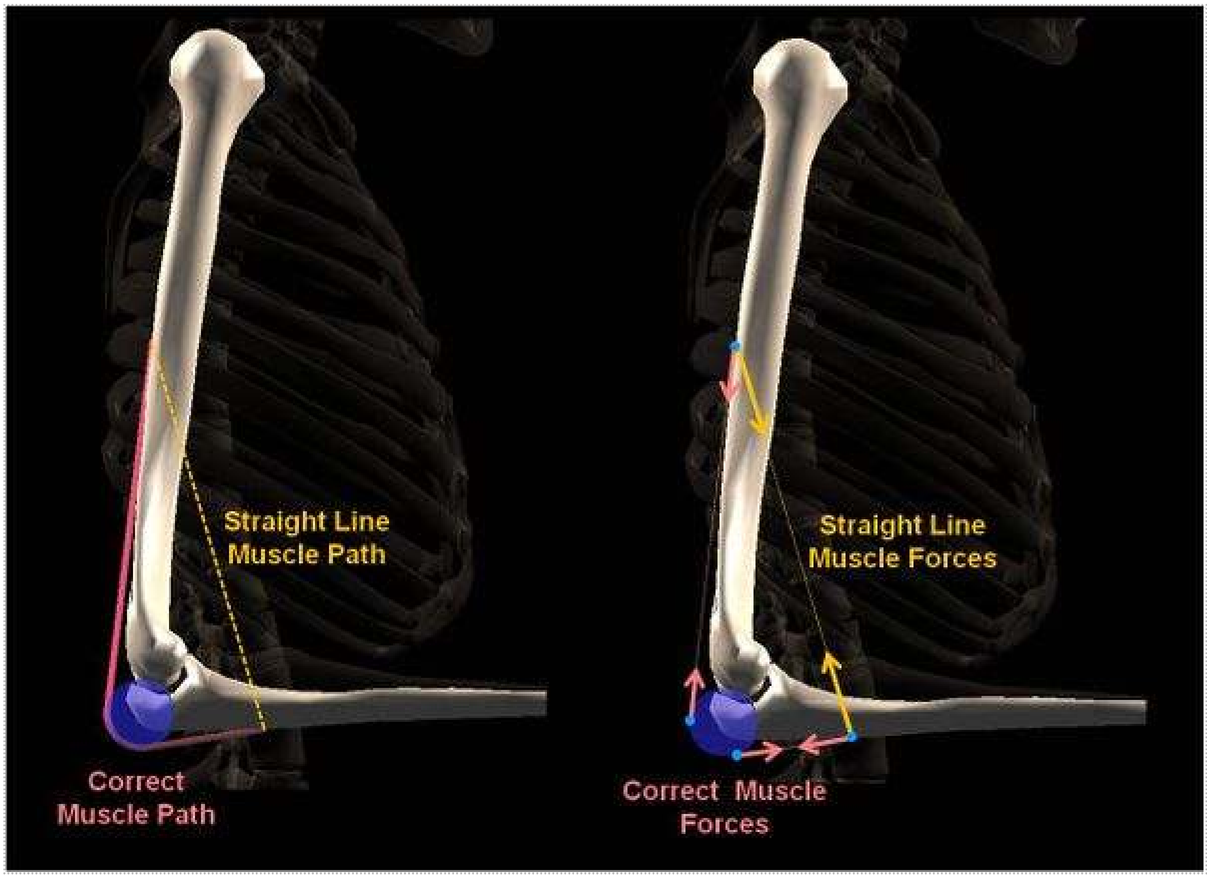


Figure 4: Calculation and application of the muscle forces on the skeletal system requires the correct modeling of the muscle path. The path of muscle yields the correct muscle length and its velocity of contraction/lengthening that are required in muscle force calculation. The path of the muscle also identifies the points where the muscle exerts force on the skeletal system and the direction of the muscle force. The figure shows two methods for modeling of the path of the triceps muscle: straight-line path and a path that uses a cylindrical wrapping object to model the bony surfaces over which the muscle wrap. The straight-line muscle path results in incorrect muscle length (left) and therefore incorrect muscle force, incorrect muscle force direction (right), and incorrect identification of the points where the muscle force must be applied to the body segments (right).

Musculotendon path has been modeled at varying levels of detail ranging from straight-line path (Raikova, 1992; Seireg & Arvikar, 1973) to centroid line method (Jensen & Davy, 1975) to finite element modeling of the path of individual muscle fibers (Blemker & Delp, 2006). Straight-line path model is easy to implement and can be adequate in simple models where the muscle has limited or no contact with the bones or tendon sheath (e.g. the biceps muscle flexing the elbow in its middle range). It is, however, inadequate for most anatomically realistic models and movements. The centroid line method represents the path of the muscle by lines that connect the centroids of the transverse cross section of the muscle along the path. This method offers a more accurate representation of the muscle path but it is impractical because the centroid lines must be obtained for all possible model postures, which is impractical. Finite element modeling is a powerful method that can help us gain insight into the aggregation and transmission of forces in individual muscle fibers to the whole muscle and to the skeletal system but it requires heavy computation that is currently impractical for modeling multiple muscle systems. The practical solution is therefore a compromise that treats the whole musculotendon as a single force-producing element (or multiple elements if the attachment sites are distributed such as in deltoid muscle (van der Helm & Veenbaas, 1991)). It models the musculotendon path as a frictionless elastic string that follows the shortest path from origin to insertion over the constraint objects (Garner & Pandy, 2000; Charlton & Johnson, 2001; Feng, Damsgaard, Rasmussen, & Christensen, 2002).

Typically, fixed points on the skeletal system are used to model the origin and insertion points. The bony prominences around which the muscles wrap are modeled by spherical or cylindrical wrapping objects. The fibrous bands of the retinacula that bind and keep the tendons close to the bones and the joint capsules are modeled by ring wrapping objects that constrain the path of the muscle to the inner side of the ring or via points through which the tendon must pass. The change in the musculotendon path due to the bulging or movement of the other muscles is modeled by moving via points that change their location as a function of the skeletal posture. Once the muscle attachment points and the path constraints are identified, a muscle path algorithm is used to find the shortest path from origin to insertion at any given skeletal configuration. Typically, the calculation involves the use of geometric principles to determine whether the straight-line path between the origin and insertion intersects the constraint objects in between, and if it does, finds the contact points on the objects. If there is no intersection or there are no constraint objects along the path, the shortest path is a straight line from origin to insertion. If there are constraints that intersect the path, the algorithm will find the shortest path that goes through and around the constraint objects that represent the real anatomical obstacles. The main advantage of this method is that it is generally applicable to all joint configurations. Once the obstacle set including the muscle attachment points and the constraint objects are specified for a single limb posture, the muscle path can be predicted for all possible limb postures. Further, it enables modeling of complex muscle paths with reasonable amounts of computation. Ongoing research investigates muscle path algorithms that can robustly predict the path of the muscle over multiple constraint objects in a computationally efficient manner (Marsden, Swailes, & Johnson, 2008; Marsden & Swailes, 2008; Audenaert & Audenaert, 2008). Figure 5 shows how a variety of simple constraint objects can be used to model complex musculotendon paths.

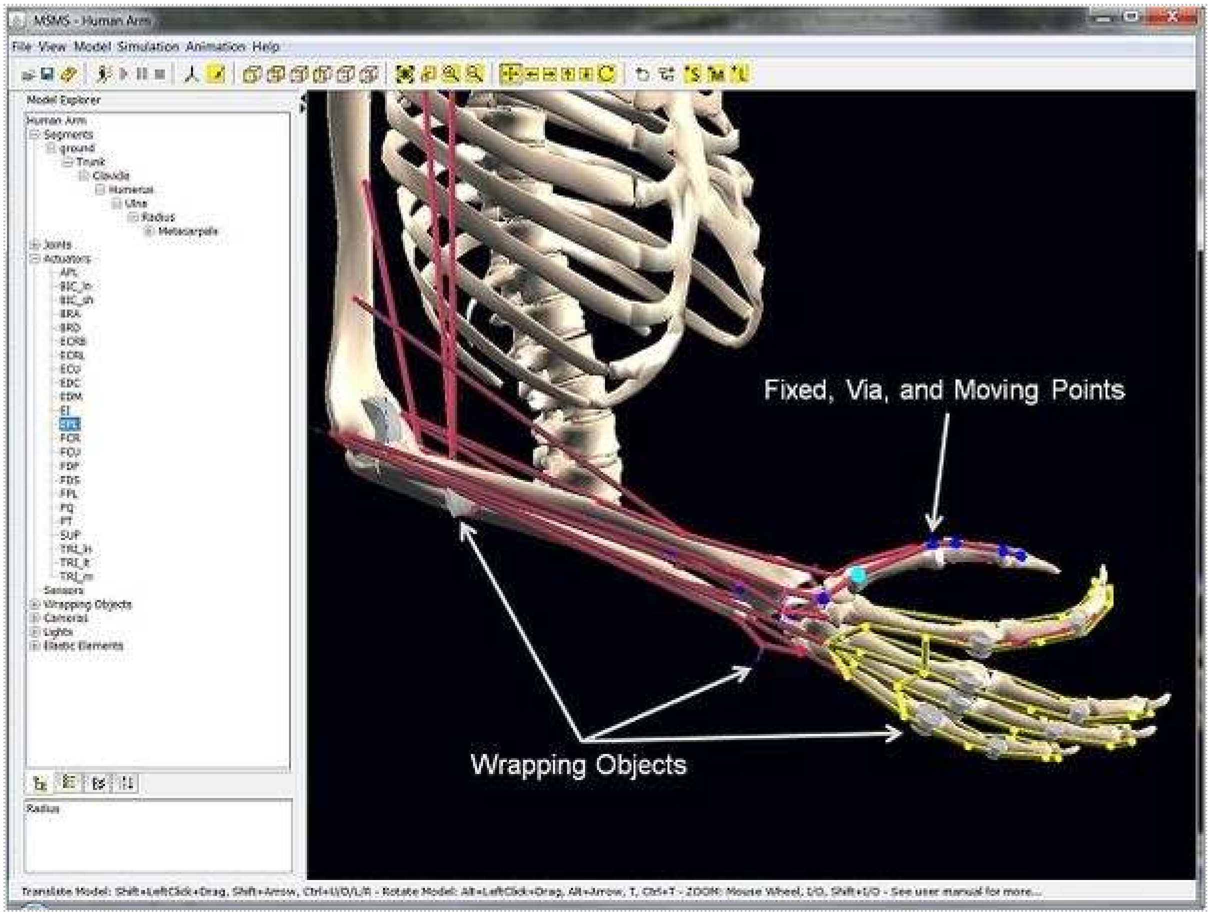
Software Tools for MusculoSkeletal Modeling

The modeling and simulation of movement in musculoskeletal systems involve processes that can be performed by any knowledgeable person, but the process is complex, prone to errors, and time-consuming. Fortunately, the most difficult processes such as the derivation of the equations of motion for the skeletal system and modeling of muscle force and muscle path can be either automated or greatly facilitated by the use of software tools.

In the past, musculoskeletal modelers have turned to multibody simulation software such as SD-Fast (http://www.sdfast.com) to automate the derivation of the equations of motion for the skeletal system. The modeler must describe the properties of the skeletal system in a text file that can then be used by the software to automatically generate the equations of motion for the skeletal system. Other multibody simulation software with interactive model building tools such as Working Model (http://www.design-simulation.com) and Adams (http://www.mscsoftware.com) have the advantage that the user can build the model of the skeletal system graphically. The main disadvantage of these primarily mechanical simulation packages is the lack of tools for modeling specialized biological components such as bones, muscles, and proprioceptors that require specially designed software tools.

The first specialized musculoskeletal modeling software, SIMM (http://www.musculographics.com) was developed in the 1990s followed by the development of newer software including those that are freely available to the public. AnyBody (http://www.anybodytech.com) and LifeModeler (http://www.lifemodeler.com) are commercial software that focus mainly on inverse simulation (calculation of the external forces causing a known movement) as opposed to forward simulation (prediction of movement caused by the known external forces). SIMM (http://www.musculographics.com) is a commercial musculoskeletal modeling software that originated from the bioengineering department of Stanford University. It can perform both forward and inverse simulations but its interactive graphical tools for editing models or building new models are rather limited. For example, new models must be described in text files before they can be opened in SIMM (http://www.musculographics.com) , making it difficult for non-experts to create new models. It also doesn't offer models of proprioceptors or support for closed-loop simulations of the sensorimotor control system, which limits its use in studies of the neural control of movement. OpenSim (http://opensim.stanford.edu) is another musculoskeletal modeling software from the same laboratory at Stanford University that is freely available to the public. OpenSim (http://opensim.stanford.edu) shares many of the features and algorithms in SIMM (http://www.musculographics.com) but it has no tools for building new musculoskeletal models and its tools for editing existing models are very limited. The users of OpenSim (http://opensim.stanford.edu) have therefore been advised to purchase and use the tools in SIMM (http://www.musculographics.com) to build and edit their musculoskeletal models.

MSMS (http://mddf.usc.edu) (MusculoSkeletal Modeling Software) is the latest freely available software package that was developed specifically to model musculoskeletal systems and to perform predictive forward dynamic simulations of the closed-loop sensorimotor control systems (Figure 5).



Translate Model: Shift+LeftClick+Drag, Shift+Arrow, Ctrl+U/D/L/R - Rotate Model: Alt+LeftClick+Drag, Alt+Arrow, T, Ctrl+T - ZOOM: Mouse Wheel, I/O, Shift+I/O - See user manual for more...

Figure 5: Software tools such as MSMS (shown) can facilitate the modeling of complex musculoskeletal systems.

A number of unique features in MSMS (http://mddf.usc.edu) are essential for conducting such simulations. In MSMS (http://mddf.usc.edu) , the modeler can interactively build models of complex musculoskeletal systems using graphical tools and wizards and simulate their movement in response to neural control inputs and external forces (Davoodi, Urata, Hauschild, Khachani, & Loeb, 2007). MSMS (http://mddf.usc.edu) offers a library of biological components involved in sensorimotor control of movement, including a detailed fiber-level model of muscle force in normal and pathological conditions (Song, Raphael, Lan, & Loeb, 2008), an accurate and validated model of muscle energetics (Tsianos, Cedric, & Loeb, 2012), and models of proprioceptors including muscle spindles and golgi tendon organs (Mileusnic, Brown, Lan, & Loeb, 2006; Mileusnic & Loeb, 2009). MSMS (http://mddf.usc.edu) also offers the capability to simulate its models off-line or in real-time virtual reality environment with the subject in the simulation loop (Davoodi & Loeb, 2012). MSMS (http://mddf.usc.edu) can be downloaded free of charge from the web page of Medical Device Development Facility (http://mddf.usc.edu) at the University of Southern California.

References

Anderson, K. S. (1992). An order-N formulation for motion simulation of general constrained multi-rigid body systems. Computers and Structures, 43(3), 565-572.

Audenaert, A., & Audenaert, E. (2008). Global optimization method for combined spherical–cylindrical wrapping in musculoskeletal upper limb modeling. Computer Methods and Programs in Biomedicine, 92, 8-19.

Blemker, S. S., & Delp, S. L. (2006). Rectus femoris and vastus intermedius fiber excursions predicted by three-dimensional muscle models. Journal of Biomechanics, 39, 1383-1391.

Charlton, I. W., & Johnson, G. R. (2001). Application of spherical and cylindrical wrapping algorithms in a musculoskeletal model of the upper limb. J.Biomech., 34(9), 1209-1216.

Davoodi, R., & Loeb, G. E. (2012). Real-time animation software for customized training to use motor prosthetic systems. IEEE Trans.Neural Syst.Rehabil.Eng, 20(2), 134-142.

Davoodi, R., Urata, C., Hauschild, M., Khachani, M., & Loeb, G. E. (2007). Model-based development of neural prostheses for movement. IEEE Transactions on Biomedical Engineering, 54(11), 1909-1918.

Erberhard, P., & Schielen, W. (2006). Computational dynamics of multibody systems: history, formalisms, and applications. ASME Journal of Computational and Nonlinear Dynamics, 1(1), 3-12.

Featherstone, R. (1987). Robotic Dynamics Algorithms Kluwer Academic Publishing.

Feng, G., Damsgaard, M., Rasmussen, J., & Christensen, S. T. (2002). Computational method for muscle-path representation in musculoskeletal models. Biological Cybernetics, 87, 199-210.

Garner, B. A., & Pandy, M. G. (2000). The Obstacle-Set Method for Representing Muscle Paths in Musculoskeletal Models. Comput.Methods Biomech.Biomed.Engin., 3(1), 1-30.

Hairer, E., Norsett, S. P., & Wanner, G. (1993). Solving ordinary differential equations I: Nonstiff problems (Second Edition ed.). Berlin: Springer Verlag.

Hairer, E., & Wanner, G. (1996). Solving ordinary differential equations II: Stiff and differential-algebraic problems (Second Edition ed.). Berlin: Springer Verlag.

Jensen, R. H., & Davy, D. T. (1975). An Investigation of Muscle Lines of Action About the Hip: A Centroid Line Approach VS the Straight Line Approach. Journal of Biomechanics, 8(2), 103-110.

Jerkovsky, W. (1978). The Structure of Multibody Dynamics Equations. Journal of Guidance and Control, 1(3), 173-182.

Kim, S. S., & Vanderploeg, M. J. (1986). A General and Efficient Method for Dynamic Analysis of Mechanical Systems Using Velocity Transformation. ASME Journal of Mechanisms, Transmissions, and Automation in Design, 108, 176-182.

Marsden, S. P., & Swailes, D. C. (2008). A novel approach to the prediction of musculotendon paths. Proc.Inst.Mech.Eng H., 222(1), 51-61.

Marsden, S. P., Swailes, D. C., & Johnson, G. R. (2008). Algorithms for exact multi-object muscle wrapping and application to the deltoid muscle wrapping around the humerus. Proc.Inst.Mech.Eng H., 222(7), 1081-1095.

Mileusnic, M. P., Brown, I. E., Lan, N., & Loeb, G. E. (2006). Mathematical models of proprioceptors. I. Control and transduction in the muscle spindle. Journal of Neurophysiology, 96(4), 1772-1788.

Mileusnic, M. P., & Loeb, G. E. (2009). Force estimation from ensembles of Golgi tendon organs. Journal of Neural Engineering, 6(3), 036001.

Nikravesh, P. E. (1988). Computer-Aided Analysis of Mechanical Systems Prentice-Hall.

Nikravesh, P. E. (1990). Systematic reduction of multibody equations of motion to a minimal set. International Journal of Non-Linear Mechanics, 25(2-3), 125-317.

Raikova, R. (1992). A general approach for modeling and mathematical investigation of the human upper limb. J Biomech., 25(8), 857-867.

Seireg, A., & Arvikar, R. J. (1973). A mathematical model for evaluation of forces in lower extremities of the musculo-skeletal system. J.Biomech., 6(3), 313-326.

Song, D., Raphael, G., Lan, N., & Loeb, G. E. (2008). Computationally efficient models of neuromuscular recruitment and mechanics. Journal of Neural Engineering, 5(2), 175-184.

Tsianos, G., Cedric, R., & Loeb, G. E. (2012). Mammalian Muscle Model for Predicting Force and Energetics During Physiological Behaviors. IEEE Trans.Neural Syst.Rehabil.Eng, 20(2), 117-133.

Unda, J., de Jalon, J. G., Losantos, F., & Emparantza, R. A. (1987). Comparative study of different formulations of the dynamic equations of constrained mechanical systems. ASME Journal of Mechanisms, Transmissions, and Automation in Design, 109, 466-474.

van der Helm, F. C., & Veenbaas, R. (1991). Modeling the mechanical effect of muscles with large attachment sites: application to the shoulder mechanism. J Biomech., 24(12), 1151-1163.

Internal references

Muscle Physiology and Modeling

Proprioceptors and Models of Transduction

External links

MSMS (http://mddf.usc.edu) , freely available software with interactive modeling tools for forward dynamic simulation of musculoskeletal and closed-loop sensorimotor control systems

OpenSim (http://opensim.stanford.edu) , freely available software with limited modeling tools for forward/inverse dynamic analysis of musculoskeltal systems

SIMM (http://www.musculographics.com) , commercial software for forward/inverse dynamic analysis of musculoskeletal systems

AnyBody (http://www.anybodytech.com) , commercial software for inverse dynamic analysis of musculoskeletal systems

LifeModeler (http://www.lifemodeler.com) , commercial musculoskeletal modeling software for inverse dynamic analysis of musculoskeletal systems

Sponsored by: Prof. Daniel Wolpert, University of Cambridge, UK

Category: Motor Control

This page was last modified on 8 April 2013, at 21:06.

This page has been accessed 8,960 times.