Taylor & Francis
Taylor & Francis Group

Computer Methods in Biomechanics and Biomedical Engineering

ISSN: 1025-5842 (Print) 1476-8259 (Online) Journal homepage: http://www.tandfonline.com/loi/gcmb20

Muscle wrapping on arbitrary meshes with the heat method

Omar Zarifi & Ian Stavness

To cite this article: Omar Zarifi & Ian Stavness (2016): Muscle wrapping on arbitrary meshes with the heat method, Computer Methods in Biomechanics and Biomedical Engineering, DOI: 10.1080/10255842.2016.1205043

To link to this article: http://dx.doi.org/10.1080/10255842.2016.1205043

COMPUTER METHODS IN BIOMECHANICS AND BIOMEDICAL ENGINEERING, 2016
http://dx.doi.org/10.1080/10255842.2016.1205043

Taylor & Francis
Taylor & Francis Group

Muscle wrapping on arbitrary meshes with the heat method

Omar Zarifi and Ian Stavness

Department of Computer Science, University of Saskatchewan, Saskatoon, Canada

ABSTRACT

Muscle paths play an important role in musculoskeletal simulations by determining a muscle's length and how its force is distributed to joints. Most previous approaches estimate the way in which muscles 'wrap' around bones and other structures with smooth analytical wrapping surfaces. In this paper, we employ Newton's method with discrete differential geometry to permit muscle wrapping over arbitrary polygonal mesh surfaces that represent underlying bones and structures. Precomputing distance fields allows us to speed up computations for the common situation where many paths cross the same wrapping surfaces. We found positive results for the accuracy, robustness, and efficiency of the method. However the method did not exhibit continuous changes in path length for dynamic simulations. Nonetheless this approach provides a valuable step toward fast muscle wrapping on arbitrary meshes.

ARTICLE HISTORY
Received 22 January 2016
Accepted 20 June 2016

KEYWORDS
Musculoskeletal simulation; muscle modeling; muscle wrapping; discrete differential geometry; geodesics

1. Introduction

Musculoskeletal simulation, with multibody dynamics and cable-like muscles, is a widely used tool in biomechanics research (Delp et al. 2007). The way in which muscles wrap around bones and other tissue forms the critical two-way connection between the 'musculo' and 'skeletal' parts of such simulations. Muscle wrapping defines how skeletal movements affect muscle forces, through changes in muscle length, and how muscle forces are distributed to joints, through muscle moment arms (Sherman et al. 2013). For this reason, muscle wrapping has a significant impact on musculoskeletal simulation efficiency and accuracy. The prevailing model for muscle wrapping represents the muscle as a massless, frictionless cable that follows the shortest path between an origin point and an insertion point while wrapping around geometric surfaces that represent bones and other structures. This shortest-path wrapping formulation presents an interesting computational problem that has been approached from many different angles.

Early attempts at shortest-path muscle wrapping in musculoskeletal models used spherical and cylindrical wrapping surfaces for which shortest paths can be computed analytically. Exact solutions have been reported for one (Charlton & Johnson 2001) and two adjacent wrapping surfaces (Marsden et al. 2008), while an iterative solution has been employed for multiple surfaces (Garner & Pandy 2000). Computing the shortest path along general smooth surfaces requires numerically integrating geodesic equations (Do Carmo 1976; Marsden & Swailes 2008). Other wrapping approaches approximate the shortest path by discretizing the muscle path and applying heuristic refinement (Delp & Loan 1995) or energy minimization (Audenaert & Audenaert 2008).

Previous wrapping methods rely on simplified shapes to serve as wrapping surfaces. Creating and tuning these shapes to achieve desired muscle paths (and moment arms) across a range of joint angles is an abstract and tedious process. To avoid this intermediate modeling step, a few algorithms that wrap muscles directly on bone shapes have been proposed using a penalty method based on a distance field to the bone surface (Marai et al. 2004) or a search method restricted to a convex volume enclosing a portion of the bone surface (Desailly et al. 2010). These methods showed promise for measuring instantaneous muscle length and moment arms, but were only evaluated in limited contexts and without attention to continuity in path length during skeletal movements. Gao et al. (2002) proposed an efficient method to wrap muscles over cross-sectional slices of a bone shape and reported that discretization led to non-smooth muscle activations in inverse-dynamics simulations. Such discontinuities in path length (and consequently muscle force) are problematic for forward-dynamics simulations which use numerical integration schemes that assume smoothly varying dynamics.

Recently, a new formulation of the shortest-path wrapping problem has been proposed that provides path

CONTACT Ian Stavness ian.stavness@usask.ca

length continuity and generalizes to arbitrary numbers of adjacent smooth wrapping surfaces (Stavness et al. 2012; Scholz et al. 2016). This *general wrapping method* represents a muscle cable as piecewise path of alternating straight and geodesic segments, whereby straight segments connect surfaces and geodesic segments wrap over surfaces. It uses Newton's method to quickly converge to an exact local shortest path. The approach still requires an additional modeling step of defining intermediary wrapping surfaces described by smooth analytical equations, rather than wrapping directly on a model's existing bone meshes. The Newton's solve also requires a good initial guess for fast convergence, and a robust initialization procedure has yet to be proposed.

Past wrapping approaches do not take advantage of precomputation that can be shared between muscle cables in the problem. Broad muscles or ligaments, such as the pectoral muscles in the chest, are often decomposed into multiple parallel cables that take similar paths around the same intermediate wrapping surface. Despite the fact that many muscle cables wrap around the same wrapping surface, past wrapping methods treat each muscle path independently. The lone exception is the work of Marai et al. (2004) who use a precomputed distance field for all muscles wrapping a single bone mesh. The lack of shared precomputation in past wrapping methods is a missed opportunity, particularly since there is a trend toward models of higher fidelity and therefore greater numbers of parallel muscle cables. For example, the recent lumbar spine by Christophy et al. (2012) includes 238 muscle elements.

In this paper, we extend the *general wrapping method* to work with arbitrary meshes and permit direct muscle wrapping on bone geometry. We use discrete differential geometry algorithms that allow us to precompute information for each wrapping surface that can be reused for all simulation steps and for all muscles wrapping over the same surface. We also propose a robust initialization procedure that can be used to provide a good initial guess to our method and other Newton's method based wrapping algorithms. We evaluate the robustness, accuracy, efficiency, and continuity properties of this new approach to muscle wrapping over bone surfaces.

2. Methods

Our wrapping method for arbitrary meshes involves three main parts: (1) a Newton's method formulation with two contact points per wrapping surface; (2) distance field computations on arbitrary meshes used to estimate tangents to the geodesic curve connecting the contact points; and (3) a heuristic initialization procedure to find a good initial guess for the Newton's solve. We implemented the method in Matlab and evaluated its convergence with a number of different arbitrary mesh shapes. We also evaluated wrapping accuracy compared to analytical cases, the robustness of the initialization procedure and the continuity of the path length in dynamic simulations.

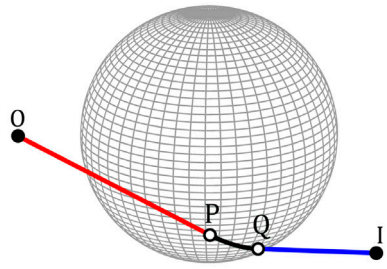


Figure 1. A valid wrapping path on a sphere.

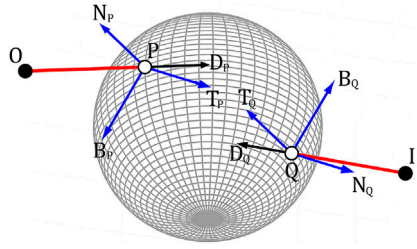


Figure 2. A sample configuration of *entry* and *exit* points, along with the vectors that define their respective trihedron.

2.1. Newton's method formulation

We adopt the general wrapping formulation proposed previously (Stavness et al. 2012; Scholz et al. 2016). Let **O** (origin) and **I** (insertion) be the endpoints of the path. The wrapped path consists of three segments: straight line between **O** and **P** (*entry*, located on the surface), surface geodesic between **P** and **Q** (*exit*, also on the surface), and a straight line between **Q** and **I**. The wrapping path is said to be valid if it is continuously differentiable; that is, if left and right unit tangents at both **P** and **Q** coincide (see Figure 1 for an example of a valid wrapping path around a sphere). Thus, our objective is to find a valid wrapping, given the endpoints of the path and the mesh to wrap around.

We opted for a symmetric problem formulation, where both *entry* and *exit* points are directly controlled (as opposed to the asymmetric, one-sided geodesic shooting method described in Scholz et al. (2016)). Given the points **P** and **Q** on the surface we define a trihedron at **P** consisting of three unit vectors: $\mathbf{T_P}$ (unit tangent of the shortest path from **P** to **Q**), $\mathbf{N_P}$ (unit normal to the surface at **P**), and $\mathbf{B_P} = \mathbf{T_P} \times \mathbf{N_P}$ (unit bitangent). The trihedron at **Q** is defined symmetrically (Figure 2). We

further define two other vectors:

$$\begin{aligned}\mathbf{D_P} &= \frac{\mathbf{P}-\mathbf{O}}{\|\mathbf{P}-\mathbf{O}\|},\\ \mathbf{D_Q} &= \frac{\mathbf{Q}-\mathbf{I}}{\|\mathbf{Q}-\mathbf{I}\|}.\end{aligned} \tag{1}$$

With the vectors defined above, our residual function is formed as follows:

$$\mathbf{f}(\mathbf{P},\mathbf{Q}) = \begin{pmatrix} f_1\\ f_2\\ f_3\\ f_4\end{pmatrix} = \begin{pmatrix} \mathbf{D_P}\cdot\mathbf{N_P}\\ \mathbf{D_P}\cdot\mathbf{B_P}\\ \mathbf{D_Q}\cdot\mathbf{N_Q}\\ \mathbf{D_Q}\cdot\mathbf{B_Q}\end{pmatrix}. \tag{2}$$

We note that the condition $\mathbf{f}(\mathbf{P},\mathbf{Q}) = 0$ is necessary, but not sufficient for a valid path; in particular, the condition holds when $\mathbf{D_P} = -\mathbf{T_P}$ (or likewise for $\mathbf{Q}$), despite such a path not being continuously differentiable. However, we assume that Newton's method will pursue a valid path when supplied with reasonably close initial conditions.

At each step of Newton's method, we set up a local parametrization (u, v) of the surface at $\mathbf{P}$ and $\mathbf{Q}$ such that

$$\begin{aligned}\mathbf{R}(0,0) &= \mathbf{P},\\ \frac{\partial\mathbf{R}}{\partial u}(0,0) &= \mathbf{T_P},\\ \frac{\partial\mathbf{R}}{\partial v}(0,0) &= \mathbf{B_P}\end{aligned} \tag{3}$$

and likewise for $\mathbf{Q}$. The residual function can then be expressed in terms of these coordinates: $\mathbf{f}(u_\mathbf{P}, v_\mathbf{P}, u_\mathbf{Q}, v_\mathbf{Q})$. Each Newton step will yield the predicted perturbations in these principal directions that will move $\mathbf{P}$ and $\mathbf{Q}$ closer to a valid configuration.

The vectors required in our residual function (surface normals and geodesic tangents) are computed at each vertex of the mesh (see Section 2.2). To find values of these vectors at arbitrary points on the surface we use linear interpolation with barycentric coordinates. That is, if $\mathbf{X} = \alpha\mathbf{v}_1 + \beta\mathbf{v}_2 + (1-\alpha-\beta)\mathbf{v}_3$ is a point in a triangle with vertices $\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3$ (where $0 \le \alpha, \beta \le 1$ are barycentric coordinates of $\mathbf{X}$), then value of a vector field at $\mathbf{X}$ is approximated by

$$\mathbf{V_X} = \alpha\mathbf{V}_1 + \beta\mathbf{V}_2 + (1-\alpha-\beta)\mathbf{V}_3, \tag{4}$$

where $\mathbf{V}_1, \mathbf{V}_2, \mathbf{V}_3$ are values of the vector field at $\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3$, respectively. This scheme provides a simple method of interpolation. It further allows for easy differentiation of interpolated vectors, which is necessary for Jacobian computations required by Newton's method. In particular, if a vector $\mathbf{Z}$ lies in the plane of $\mathbf{X}$'s triangle, then we can write

$$\mathbf{Z} = \mu(\mathbf{v}_1 - \mathbf{v}_3) + \nu(\mathbf{v}_2 - \mathbf{v}_3) \tag{5}$$

for some μ, ν. Then for a small perturbation t, we have

$$\begin{aligned}\mathbf{V}_{\mathbf{X}+t\mathbf{Z}} = {}&(\alpha+t\mu)\mathbf{V}_1 + (\beta+t\nu)\mathbf{V}_2\\ &+ (1-\alpha-\beta-t(\mu+\nu))\mathbf{V}_3,\end{aligned} \tag{6}$$

so that

$$\frac{\mathrm{d}\mathbf{V}_{\mathbf{X}+t\mathbf{Z}}}{\mathrm{d}t} = \mu(\mathbf{V}_1 - \mathbf{V}_3) + \nu(\mathbf{V}_2 - \mathbf{V}_3). \tag{7}$$

This fact, along with the normalization identity:

$$\frac{\mathrm{d}}{\mathrm{d}x}\frac{\mathbf{v}(x)}{\|\mathbf{v}(x)\|} = \frac{(\mathbf{v}(x)\cdot\mathbf{v}(x))\mathbf{v}'(x) - (\mathbf{v}(x)\cdot\mathbf{v}'(x))\mathbf{v}(x)}{\|\mathbf{v}(x)\|^3}, \tag{8}$$

allows us to calculate most of the entries of the Jacobian matrix of $\mathbf{f}$. Here we provide details of the derivatives of $\mathbf{f}$ with respect to changes in $u_\mathbf{P}, v_\mathbf{P}$ and note that, by symmetry, derivatives with respect to $u_\mathbf{Q}, v_\mathbf{Q}$ can be obtained analogously.

The first Jacobian term, $\partial f_1/\partial u_\mathbf{P}$, involves the rate of change of the *entry* and normal vector at $\mathbf{P}$ with respect to $u_\mathbf{P}$. This requires expressions for $\partial\mathbf{D_P}/\partial u_\mathbf{P}$ and $\partial\mathbf{N_P}/\partial u_\mathbf{P}$, since

$$\frac{\partial f_1}{\partial u_\mathbf{P}} = \frac{\partial(\mathbf{D_P}\cdot\mathbf{N_P})}{\partial u_\mathbf{P}} = \frac{\partial\mathbf{D_P}}{\partial u_\mathbf{P}}\cdot\mathbf{N_P} + \mathbf{D_P}\cdot\frac{\partial\mathbf{N_P}}{\partial u_\mathbf{P}}. \tag{9}$$

For $\partial\mathbf{D_P}/\partial u_\mathbf{P}$, from the definition of $\mathbf{D_P}$ and (8) it follows that

$$\frac{\partial\mathbf{D_P}}{\partial u_\mathbf{P}} = \frac{\|\mathbf{P}-\mathbf{O}\|^2\mathbf{T_P} - ((\mathbf{P}-\mathbf{O})\cdot\mathbf{T_P})(\mathbf{P}-\mathbf{O})}{\|\mathbf{P}-\mathbf{O}\|^3}. \tag{10}$$

For $\partial\mathbf{N_P}/\partial u_\mathbf{P}$, suppose that $\mathbf{P}$ is located in a triangle with vertices $\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3$ and its barycentric coordinates are $0 \le \alpha, \beta \le 1$. If we define $\tilde{\mathbf{N}}_\mathbf{P} = \alpha\mathbf{N}_1 + \beta\mathbf{N}_2 + (1-\alpha-\beta)\mathbf{N}_3$ (where $\mathbf{N}_1, \mathbf{N}_2, \mathbf{N}_3$ are surface normals at $\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3$, respectively), then $\mathbf{N_P} = \tilde{\mathbf{N}}_\mathbf{P}/\|\tilde{\mathbf{N}}_\mathbf{P}\|$. Finally, we let $\tilde{\mathbf{T}}_\mathbf{P} = \mu(\mathbf{v}_1 - \mathbf{v}_3) + \nu(\mathbf{v}_2 - \mathbf{v}_3)$ be the projection of $\mathbf{T_P}$ on the $\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3$ triangle. If we define $\mathbf{W} = \mu(\mathbf{N}_1 - \mathbf{N}_3) + \nu(\mathbf{N}_2 - \mathbf{N}_3)$, then

$$\frac{\partial\mathbf{N_P}}{\partial u_\mathbf{P}} = \frac{\|\tilde{\mathbf{N}}_\mathbf{P}\|^2\mathbf{W} - (\tilde{\mathbf{N}}_\mathbf{P}\cdot\mathbf{W})\tilde{\mathbf{N}}_\mathbf{P}}{\|\tilde{\mathbf{N}}_\mathbf{P}\|^3}. \tag{11}$$

Expressions (10) and (11) can be substituted into (9) to obtain $\partial f_1/\partial u_\mathbf{P}$. The same rationale applies to $\partial f_1/\partial v_\mathbf{P}$.

The second Jacobian term, $\partial f_2/\partial u_\mathbf{P}$, involves the rate of change of the bitangent vector to the geodesic at $\mathbf{P}$ with

respect to $u_{\mathbf{P}}$:

$$\frac{\partial \mathbf{B_P}}{\partial u_{\mathbf{P}}} = \frac{\partial(\mathbf{T_P} \times \mathbf{N_P})}{\partial u_{\mathbf{P}}} = \frac{\partial \mathbf{T_P}}{\partial u_{\mathbf{P}}} \times \mathbf{N_P} + \mathbf{T_P} \times \frac{\partial \mathbf{N_P}}{\partial u_{\mathbf{P}}}. \quad (12)$$

where $\partial \mathbf{T_P}/\partial u_{\mathbf{P}}$ can be computed similarly to $\partial \mathbf{N_P}/\partial u_{\mathbf{P}}$. The same procedure can be used to calculate $\partial f_2/\partial v_{\mathbf{P}}$.

The third Jacobian term, $\partial f_3/\partial u_{\mathbf{P}}$, involves $\partial \mathbf{D_Q}/\partial u_{\mathbf{P}}$ and $\partial \mathbf{N_Q}/\partial u_{\mathbf{P}}$. In our symmetric formulation of the residual function, **P** and **Q** are controlled independently, so that movement of **P** does not result in location change for **Q**. For this reason, $\mathbf{N_Q}$ and $\mathbf{D_Q}$ are constant with respect to perturbations of **P** and therefore

$$\frac{\partial f_3}{\partial u_{\mathbf{P}}} = 0, \frac{\partial f_3}{\partial v_{\mathbf{P}}} = 0. \quad (13)$$

The fourth Jacobian term, $\partial f_4/\partial u_{\mathbf{P}}$, involves $\partial \mathbf{B_Q}/\partial u_{\mathbf{P}}$. If **P** moves along the geodesic connecting **P** and **Q** (i.e. a small distance in the direction $u_{\mathbf{P}}$), the geodesic curve will shorten at **P**'s end, but the trihedron **Q** should remain unchanged. Thus, a small perturbation in the geodesic tangent direction of **P** should leave the vectors at **Q** unaffected, therefore:

$$\frac{\partial f_4}{\partial u_{\mathbf{P}}} = 0. \quad (14)$$

The same reasoning, however, does not hold for $\partial f_4/\partial v_{\mathbf{P}}$. If **P** moves perpendicular to the geodesic connecting **P** and **Q** (i.e. a small distance in the direction $v_{\mathbf{P}}$) the trihedron **Q** should rotate about $\mathbf{N_Q}$ and therefore modify the residual term f_4. For continuous surfaces, these derivatives can be computed via parallel transport of the geodesic curve (see Scholz et al. 2016, Section 3.3). For arbitrary meshes the parallel transport of geodesic curves is an open problem, therefore we currently use finite differences to estimate $\partial f_4/\partial v_{\mathbf{P}}$ (and by symmetry $\partial f_2/\partial v_{\mathbf{Q}}$).

2.2. Distance field computation

The Newton's method formulation described above requires computing the tangent vector at each end of a geodesic curve connecting points **P** and **Q** on a surface, i.e. $\mathbf{T_P}$ and $\mathbf{T_Q}$ in Equations (10) and (12). For continuous surfaces, the problem of finding a geodesic curve between two points is commonly formulated as a two-point boundary value problem (Gliklikh & Zykov 2007). For arbitrary meshes, we can recast this problem as that of finding the negative gradient of a distance field radiating from **P** measured at **Q** and vice versa. For this to be feasible, we require a fast method to calculate distance fields across arbitrary meshes.

Recently, Crane et al. (2013) proposed an efficient discrete differential geometry approach for approximating distance fields on arbitrary meshes that they term the *heat method*. Motivated by Varadhan's formula, this method utilizes the heat kernel to determine the distance field on the mesh from a given source set of vertices. The algorithm consists of three main steps:

(i) Integrate the PDE $\dot{w} = \Delta w$ to some time t.
(ii) Compute the negative normalized gradient: $\mathbf{Y} = -\nabla w/\|\nabla w\|$.
(iii) Compute the divergence of $\mathbf{Y}$ and solve $\Delta\phi = \nabla \cdot \mathbf{Y}$.

The Laplacian is discretized on the triangular mesh using the linear cotan scheme (MacNeal 1949) and the heat equation is integrated in time via a single Backward Euler step. Gradient and divergence operators are likewise discretized using standard linear schemes. After discretization, steps (i) and (iii) of the above algorithm reduce to solving sparse linear systems of equations, with constant symmetric positive-definite coefficient matrices that can be pre-factored. Thus, after the initial build phase, heat method allows us to quickly re-compute distance fields from changing source sets.

In order to visualize the wrapping path, we compute a geodesic path by integrating the negated gradient of the distance field starting at one of the contact points. In practice we blend the geodesics shot from the *entry* and *exit* points to obtain a visually smoother path. We note that finding the actual geodesic is only required for visualization, as the path length is given by the distance field, and the rate of path length change by the velocity of the contact points (see Scholz et al. 2016, Section 3.5).

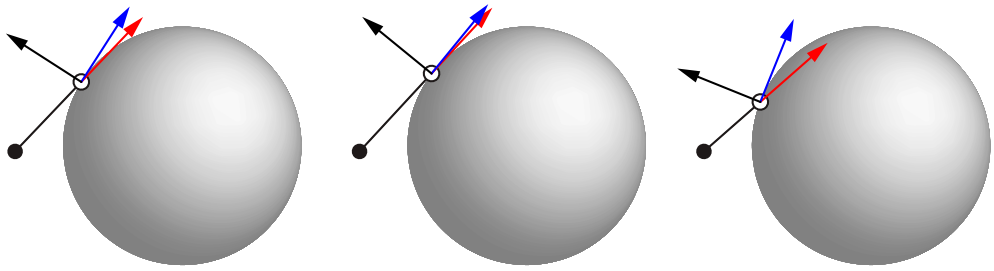
2.3. Initialization procedure

Newton's method works well when starting from an initial set of parameters that are close to the final solution. To acquire a good initial guess, we developed a simple automatic heuristic Marching algorithm. It is slower and less accurate than Newton's method, but more robust when the guess is far from the solution. The Marching procedure occurs for points **P** and **Q** simultaneously.

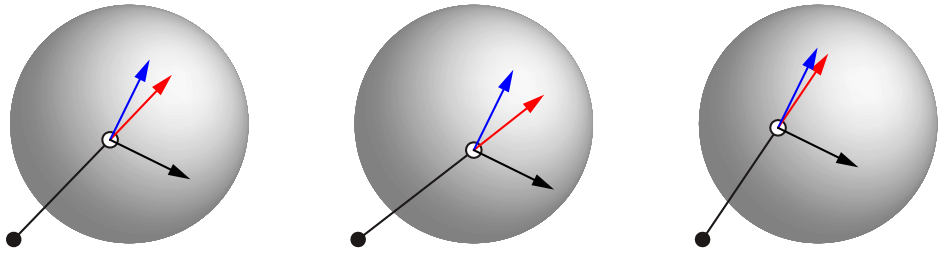
We define a two-dimensional residual vector:

$$\mathbf{e} = \begin{pmatrix} e_1 \\ e_2 \end{pmatrix} = \begin{pmatrix} (\mathbf{D_P} \times \mathbf{T_P}) \cdot \mathbf{B_P} \\ -(\mathbf{D_P} \times \mathbf{T_P}) \cdot \mathbf{N_P} \end{pmatrix}. \quad (15)$$

The point **P** is then moved in the direction of $\mathbf{T_P}$ by ke_1 and in the direction of $\mathbf{B_P}$ by ke_2, where k is a positive constant (Figure 3). This process is repeated until we arrive at a satisfactory initial guess for Newton's method, *i.e.* a user-defined convergence tolerance on $\|\mathbf{e}\|$. Note



(a) Marching in the tangent direction from a configuration with $e_1 > 0$ (left). Moving the surface point forward along the geodesic rotates $\mathbf{D}$ and $\mathbf{T}$ vectors towards each other (middle), while moving it backward increases the error (right).



(b) Marching in the bitangent direction from a configuration with $e_2 < 0$ (left). Moving the surface point forward along the bitangent increases the error (middle), whereas moving it backward rotates $\mathbf{D}$ and $\mathbf{T}$ towards each other (right).

Figure 3. Initialization procedure to align $\mathbf{D}$ (red) and $\mathbf{T}$ (blue) vectors by Marching in the tangent (a) and bitangent (b) directions.

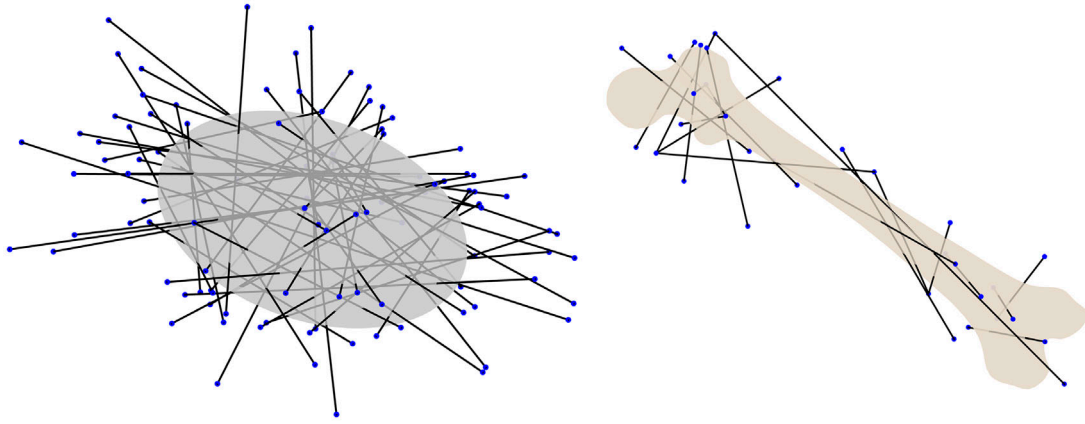


Figure 4. Test cases for robustness evaluation (only first 50 trials for the ellipsoid are shown).

that the initialization routine needs to be called only once, before the first wrapping is performed. Subsequent wrappings can use the configuration from the previous time step as a starting point.

Choice of k governs both the accuracy of the method as well as its convergence speed. If k is small, $\mathbf{P}$ is more careful with its movement, and the method can give a more accurate result. For larger k, we arrive at a reasonable guess in fewer iterations, at the cost of asymptotic accuracy. Since we use this initialization process only for acquiring of an initial guess, a larger k is preferred.

2.4. Evaluation

We evaluated the performance of our wrapping algorithm in a number of ways: robustness of the initialization scheme and accuracy, continuity, and efficiency of the wrapping scheme. Our test cases used three meshes: an ellipsoid mesh attained by stretching an icosahedral sphere (2562 vertices, 5120 faces), a femur mesh (5138 vertices, 10272 faces) and the well-known Stanford bunny mesh (2503 vertices, 4968 faces, with filled holes).

Robustness of the initialization method was evaluated both with the ellipsoid mesh and the femur mesh. For the ellipsoid, 500 trials were performed, with the endpoints of the path randomly generated such that the line segment connecting them intersected the ellipsoid. Due to difficulty in randomly generating trials for the femur, we manually crafted a test suite consisting of 20 representative experiments. Figure 4 shows the unwrapped test cases (only the first 50 trials for the ellipsoid are in the diagram). For both meshes, the initialization algorithm

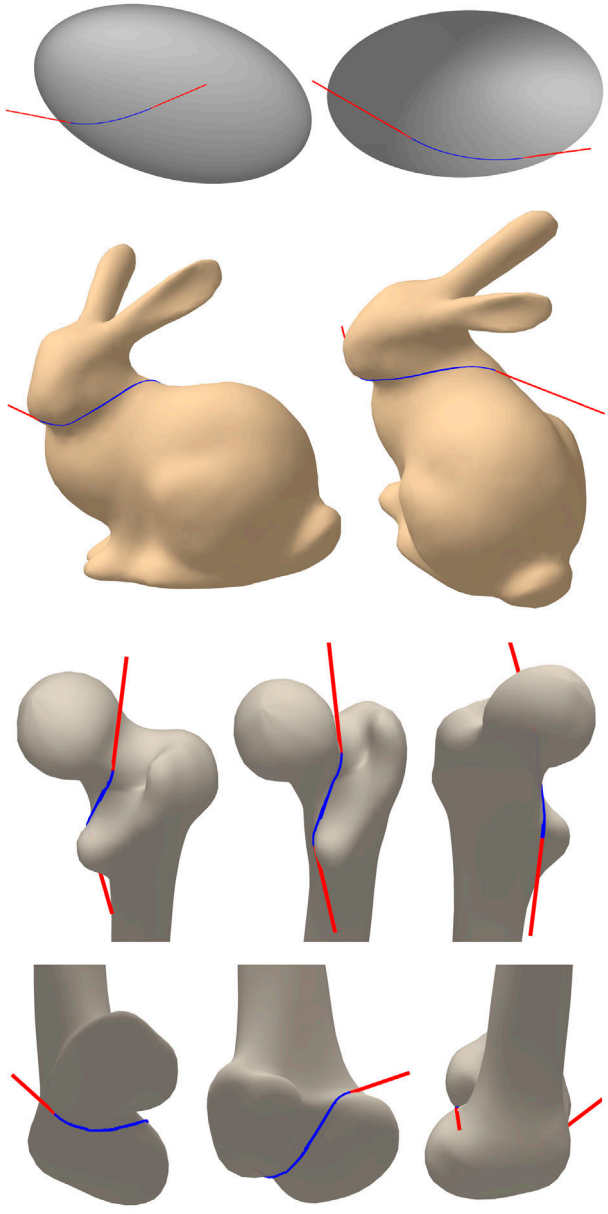


Figure 5. Wrapping results for ellipsoid, bunny, proximal femur and distal femur meshes.

was set to start searching at one of the triangles closest to the endpoint (for both origin and insertion points). The tests were performed with several combinations of convergence tolerances and step sizes (k constant for the Marching method). An iteration limit was set at 200; if the routine failed to converge within tolerance in 200 iterations, a failure was reported.

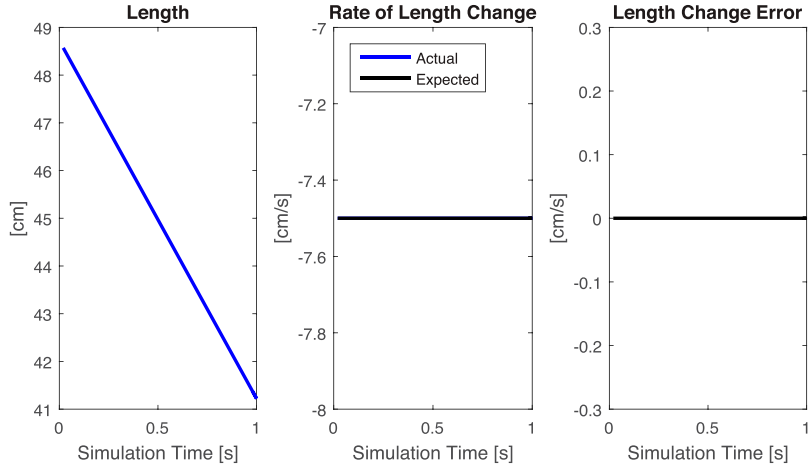
Accuracy of the wrapping method was assessed by comparing an analytical sphere solution to wrapping on a sphere mesh with three different levels of discretization (162, 642, and 2562 vertices, respectively). Five hundred test cases were randomly generated in a similar manner to the robustness test above. Our initialization scheme was used to find a satisfactory initial guess, and the iteration limit was set to 200. Of the 500 test cases, we counted the number of tests that failed to converge. For the tests that converged, we compared results obtained with our method to analytical solutions for a sphere. *Length error* was calculated at the total path length relative to the exact length of the analytical path. *Tangent error* was defined to be the average angle between the computed tangent vectors and their analytical counterparts at *entry* and *exit* points. Lastly, *surface error* was computed as the average distance between approximate and exact *entry* and *exit* points. Note that we did not compensate for the natural deviation caused by discretization of the sphere.

To assess the continuity properties of our wrapping method, we performed simulations in which one endpoint of the path was moved incrementally at each timestep with a constant velocity while the opposing endpoint and wrapping surface were stationary. We analyzed path continuity for endpoint motion in three directions: the tangent direction (to elicit an overall shortening of the path, but no change in the surface path segment), the negative normal direction (expected to bend the path around the surface), and the bitangent direction (expected to slide the path along the surface). For each trial, we recorded the evolution of path length over time, as well as its rate of change. This rate of change was then compared to the analytical result, which was computed using the method derived in Scholz et al. (2016). Our continuity tests were performed on the ellipsoid mesh with a fixed simulation timestep size of 0.02 s for 50 steps.

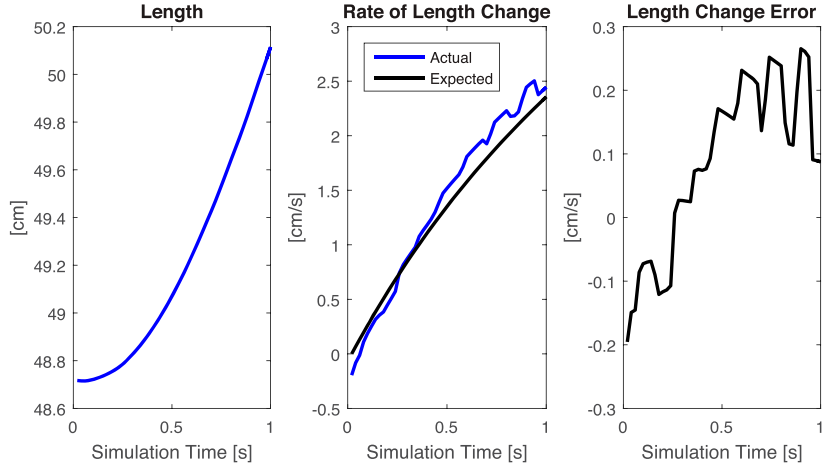
Convergence of our wrapping method was measured for 500 tests on the ellipsoid with randomly generated endpoints. Convergence tolerance was set to 10^{-8} and the initial guess was formed by first using our initialization procedure. We recorded the number of failures to converge (within 100 iteration limit). In addition, we counted the number of Newton iterations for each trial, as well as the running time. The tests were executed on a Windows 8.1 machine with an Intel Xeon E5-2637 CPU (running at 3.5 GHz) and 32 GB of RAM. Runtime is difficult to assess in practical terms because our prototype was implemented in Matlab, an interpreted environment, which does not produce timing results representative of a realistic fast implementation. For example, the distance field calculations in Matlab were several orders of magnitude slower when compared to the optimized C++ implementation of the heat method reported in Crane et al. (2013)[1]. Nevertheless, the computed speed-up factors allowed us to estimate the likely running times of a faster implementation.

3. Results

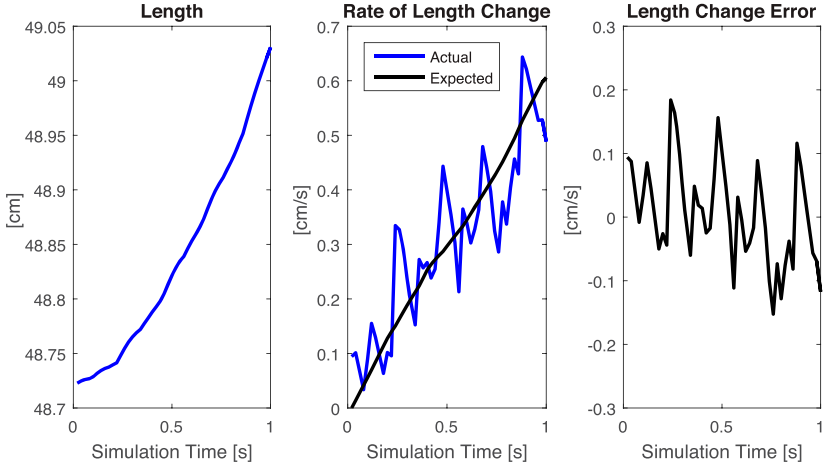
The initialization procedure worked well with the ellipsoid and succeeded to find a reasonable initial guess in



(a) Stretching of the overall path (endpoint motion in tangent direction). The surface segment length is constant while the endpoint segment lengthens.



(b) Bending of the overall path (endpoint motion in negative normal direction).



(c) Sliding of the overall path (endpoint motion in bitangent direction).

Figure 6. Continuity results for three tested path motions. The surface segment length and rate of length change are shown.

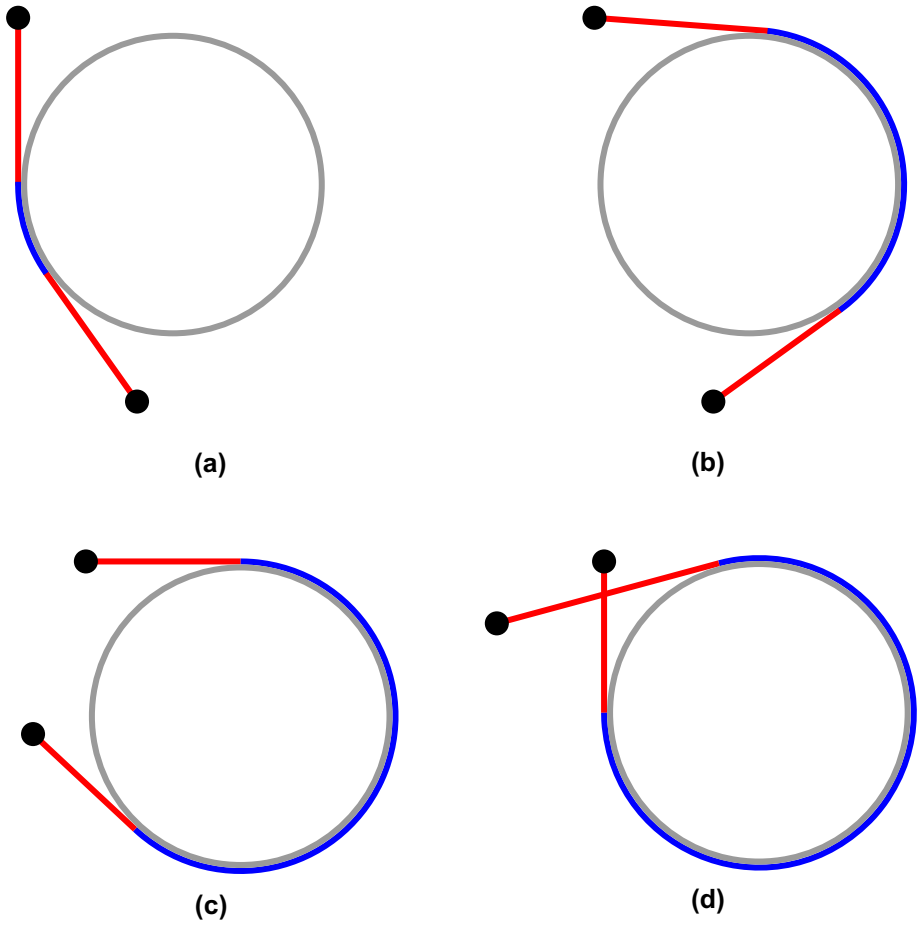


Figure 7. Example wrapping paths. Our approach requires that the surface wrapping segment (blue) is the locally shortest distance path between contact points. This approach works for configurations in which the path bends (a) or even takes the long way around an obstacle (b), but will fail for cases in which the path wraps completely around to enclose an obstacle (c, d).

almost all cases (Table 1). The trade-off between speed and accuracy was also evident: increasing the step size k from 0.125 up to 0.25 resulted in appreciably fewer iterations required on average, at the cost of a higher chance to fail (7 failed trials as opposed to none). A step size of $k = 0.2$ resulted the fewest failures and the second-fastest performance across all test cases. Robustness tests with the femur mesh resulted in higher chance of failure. We note that even in the failed cases, the algorithm still managed to get close to a satisfactory guess, but its inaccuracy prevented the convergence tolerance from being met. This can be seen by observing that the number of failures fell as convergence tolerance was loosened. Furthermore, for the most problematic trial, the algorithm managed to reduce the residual to 0.0519, which although above the tolerance of 0.05, still provided a sufficiently good initial guess that Newton's method converged within a couple of iterations.

The wrapping procedure was successful for all test meshes (Figure 5) and the sphere mesh results were reasonable close to the analytical solution for all trials (Table 2). Furthermore, the error and number of convergence failures decreased as the mesh resolution increased.

We found that path length was not smooth during movement simulations (Figure 6). Simulating path endpoint motion at the same speed in different directions produced different rates of path length change. Path shortening by displacement of one endpoint in the tangent direction produced the expected result of no change in surface segment length. However, both bending and sliding motions (displacement of the endpoint perpendicular to the tangent direction) produced errors in path length change. Path bending motions fared better than sliding motions, as they evolved more smoothly, but these discontinuities may hamper forward dynamics simulations with high accuracy tolerances.

We found consistently good convergence of the wrapping algorithm following initialization. Of the 500 trials comprising our efficiency test suite, the algorithm failed to converge within 100 iterations just once. For the rem-

Table 1. Robustness results of the initialization procedure for 500 random trials with an Ellipsoid mesh and 20 trials with the Femur mesh.

	Convergence tolerance	Iterations[a] (Failures[b])			
		$k = 0.125$	$k = 0.15$	$k = 0.2$	$k = 0.25$
Ellipsoid	0.05	32.29 (0)	25.89 (0)	19.17 (0)	15.30 (7)
Femur	0.05	61.44 (2)	51.11 (2)	45.05 (1)	36.94 (2)
Femur	0.06	58.16 (1)	48.21 (1)	41.95 (0)	34.32 (1)
Femur	0.07	53.79 (1)	51.95 (0)	38.70 (0)	31.63 (1)

[a]Average number of iterations until convergence.
[b]Total number of trials that failed to converge.

Table 2. Accuracy results of the wrapping algorithm for 500 random trials with a Sphere mesh.

	Mesh size vertices (faces)	Length error[a] (%)	Tangent error (°)	Surface error	Convergence failures
Sphere	162 (320)	0.99	1.25	0.0263	7
Sphere	642 (1280)	0.47	0.71	0.0150	3
Sphere	2562 (5120)	0.31	0.78	0.0153	2

[a]Length errors are relative to exact length.

aining 499 successes, an average of 4.94 iterations were required (with a standard deviation of 1.96 iterations), and the average running time was 2.40 s; this rate of convergence translates to a projected average running time of under 0.03 s for a faster implementation.

4. Discussion

Few previous muscle wrapping methods permit wrapping muscle cables directly on polygonal mesh representations of bones and other model structures. We accomplish this by using distance fields across a mesh to find the shortest path geodesic to connect *entry* and *exit* points of a wrapping surface. The distance field approach has the benefit that distance information can be precomputed and reused when wrapping multiple cables over the same wrapping surface. For example, three wrapping surfaces can accommodate over 40 wrapping muscles and ligaments in the Lenhart et al. (2015) knee model.

The distance field approach is limited to finding wrapping paths for which the surface segment is a locally shortest path along the surface. Our method will fail for wrapping paths that fully wrap around an enclosed surface. While such wrapping is observed in cable-driven mechanical systems, e.g. for pulleys, we know of no such cases for muscle and ligament wrapping in the human body. Muscle and ligament paths will generally bend around a bone and continue in the same direction (Figure 7(a) and (b)), rather than wrapping fully around an object and bending back in the opposite direction (Figure 7(c) and (d)). It is important to note that our method does permit the overall path to 'take the long way around' a structure (Figure 7(b)), as happens for some muscles in the shoulder.

The automatic initialization procedure performed well and was robust to a wide variety of initial conditions. The procedure was designed and tested for the case of a path wrapping a single obstacle. It is not clear whether the proposed heuristic will extend to the more general case of multi-obstacle wrapping. Nonetheless, the scheme may be useful in practice as many wrapping problems involve only single obstacles. For example, muscles in the Arnold et al. (2010) lower extremity model exclusively involve single-obstacle wrapping.

Accuracy, robustness and efficiency tests were generally positive, but our method did not provide continuous changes in path length during dynamic simulations. The continuity results were better for stretching and bending motions of the path (tangent and normal motions of the endpoint) than for sliding motions of the path (bitangent motion of the endpoint). The magnitude of length change error was comparable to errors found for wrapping methods that use a discretized elastic path (see Scholz et al. 2016, Figure 7(d)).

Previous wrapping studies have only reported tests for a small number of 'representative' wrapping configurations. In this study, we devised a test suite with a variety of meshes and a large random sample of test configurations. Results for a large number of test cases provide us with some confidence that our method will work for the large variety of wrapping conditions present in musculoskeletal models.

5. Future work

We did not achieve path length continuity with our current implementation. However, we expect that the continuity could be improved by replacing linear interpolation

(4) with a smoother scheme, such as using local quadratic patches (Nagata 2005) near each contact point. Smoother interpolation may also allow coarser meshes to be used without sacrificing accuracy.

We do not currently consider the problem of a wrapping path lifting off, or coming into contact with, a wrapping surface. At present, the contact problem would have to be handled outside of the wrapping algorithm, e.g. checking for lift off or contact before engaging wrapping. Likewise, while our approach works with non-convex shapes, such as the femur mesh used for testing, the wrapping segment remains on the surface, rather than lifting off over concave regions. The lift-off/contact problem has been examined by selected previous works (Gao et al. 2002; Scholz 2016), but an analysis with respect to path continuity for arbitrary meshes remains an open problem.

Our current implementation pre-factors the matrix used to compute distance fields and reuses that information in each step and for each strand over the same surface. We could instead precompute the entire set of distance fields for each mesh, often called an *affinity* matrix, which would significantly speed up the wrapping calls at the cost of greater memory usage. Indeed, our preliminary tests show that the run-time can be reduced by two orders of magnitude by taking advantage of such massive precomputing: the aforementioned efficiency tests on the ellipsoid required an average running time of just over 0.02 s (compared to 2.4 s). Surprisingly, the increase in memory usage associated with this modification is not too great in practice. For example, the symmetric *affinity* matrix for the ellipsoid mesh with 2562 vertices only requires 25 MB of memory (and scales quadratically with the number of vertices). Further, we would no longer need to store the matrix prefactorization, and therefore the net memory footprint increase would only be approximately 40% (from 15 to 25 MB).

6. Summary

We have developed and tested a new algorithm for muscle wrapping around arbitrary bone meshes in musculoskeletal models. The algorithm uses initialization and Newton's method to rapidly converge on a valid wrapping path based on distance fields computed across the mesh surface with the geodesic heat method. The heat method is well suited to the problem since, after the initial build phase, it allows for fast distance field computations from changing source points. Moreover, a single build operation is performed for each wrapping surface, regardless of the number of paths that wrap around it. While path length discontinuity remains a challenge for our discrete differential geometry scheme, the approach provides an important step toward allowing musculoskeletal modelers the freedom to use mesh representations of wrapping surfaces in their future models.

Note

1. An optimized library is provided at http://www.cs.columbia.edu/keenan/index.html#code.

Acknowledgements

We thank John Lloyd and Michael Sherman for their informative discussions.

Disclosure statement

No potential conflict of interest was reported by the authors.

Funding

This work was funded by the Natural Sciences and Engineering Research Council of Canada.

References

Arnold EM, Ward SR, Lieber RL, Delp SL. 2010. A model of the lower limb for analysis of human movement. Ann Biomed Eng. 38:269–279.

Audenaert A, Audenaert E. 2008. Global optimization method for combined spherical-cylindrical wrapping in musculoskeletal upper limb modelling. Comput Methods Programs Biomed. 92:8–19.

Charlton IW, Johnson GR. 2001. Application of spherical and cylindrical wrapping algorithms in a musculoskeletal model of the upper limb. J Biomech. 34:1209–1216.

Christophy M, Senan NAF, Lotz JC, OReilly OM. 2012. A musculoskeletal model for the lumbar spine. Biomech Model Mechanobiol. 11:19–34.

Crane K, Weischedel C, Wardetzky M. 2013. Geodesics in heat: a new approach to computing distance based on heat flow. ACM Trans Graphics (TOG). 32:152-1–152-11.

Delp SL, Anderson FC, Arnold AS, Loan P, Habib A, John CT, Guendelman E, Thelen DG. 2007. Opensim: open-source software to create and analyze dynamic simulations of movement. IEEE Trans Biomed Eng. 54:1940–1950.

Delp SL, Loan JP. 1995. A graphics-based software system to develop and analyze models of musculoskeletal structures. Comput Biol Med. 25:21–34.

Desailly E, Sardain P, Khouri N, Yepremian D, Lacouture P. 2010. The convex wrapping algorithm: a method for identifying muscle paths using the underlying bone mesh. J Biomech. 43:2601–2607.

Do Carmo MP. 1976. Differential geometry of curves and surfaces. Vol. 2. Upper Saddle River (NJ): Prentice-hall Englewood Cliffs.

Gao F, Damsgaard M, Rasmussen J, Christensen ST. 2002. Computational method for muscle-path representation in musculoskeletal models. Biol Cybern. 87:199–210.

Garner BA, Pandy MG. 2000. The obstacle-set method for representing muscle paths in musculoskeletal models. Comput Methods Biomech Biomed Eng. 3:1–30.

Gliklikh YE, Zykov P. 2007. On the two-point boundary-value problem for equations of geodesics. J Math Sci. 144:4409–4412.

Lenhart RL, Kaiser J, Smith CR, Thelen DG. 2015. Prediction and validation of load-dependent behavior of the tibiofemoral and patellofemoral joints during movement. Ann Biomed Eng. 43:2675–2685.

MacNeal RH. 1949. The solution of partial differential equations by means of electrical networks [dissertation]. Pasadena (CA): California Institute of Technology.

Marai GE, Laidlaw DH, Demiralp Ç, Andrews S, Grimm CM, Crisco JJ. 2004. Estimating joint contact areas and ligament lengths from bone kinematics and surfaces. IEEE Trans Biomed Eng. 51:790–799.

Marsden S, Swailes D. 2008. A novel approach to the prediction of musculotendon paths. Proc Inst Mech Eng H J Eng Med. 222:51–61.

Marsden S, Swailes D, Johnson G. 2008. Algorithms for exact multi-object muscle wrapping and application to the deltoid muscle wrapping around the humerus. Proc Inst Mech Eng H J Eng Med. 222:1081–1095.

Nagata T. 2005. Simple local interpolation of surfaces using normal vectors. Comput Aided Geom Des. 22:327–347.

Scholz A. 2016. Fast differential-geometric methods for continuous muscle wrapping over multiple general surfaces [dissertation]. Duisburg: Universität Duisburg-Essen. 2015.

Scholz A, Sherman M, Stavness I, Delp S, Kecskeméthy A. 2016. A fast multi-obstacle muscle wrapping method using natural geodesic variations. Multibody Sys Dyn. 36:195–219.

Sherman MA, Seth A, Delp SL. 2013. What is a moment arm? Calculating muscle effectiveness in biomechanical models using generalized coordinates. In: ASME 2013 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference. Portland (OR): American Society of Mechanical Engineers; p. V07BT10A052–V07BT10A052.

Stavness I, Sherman M, Delp S. 2012. A general approach to muscle wrapping over multiple surfaces. In: Proceedings of the American Society of Biomechanics. Gainesville (FL); p. 1–2. Abstract 97.