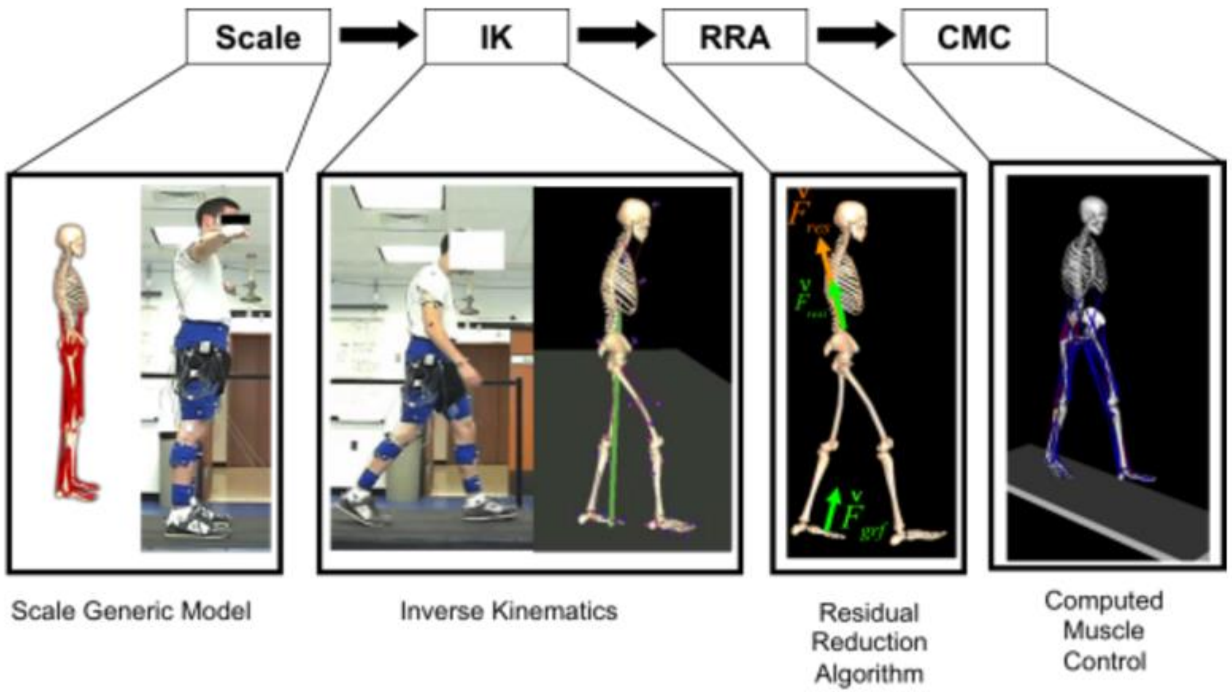
OpenSim

Musculoskeletal modeling

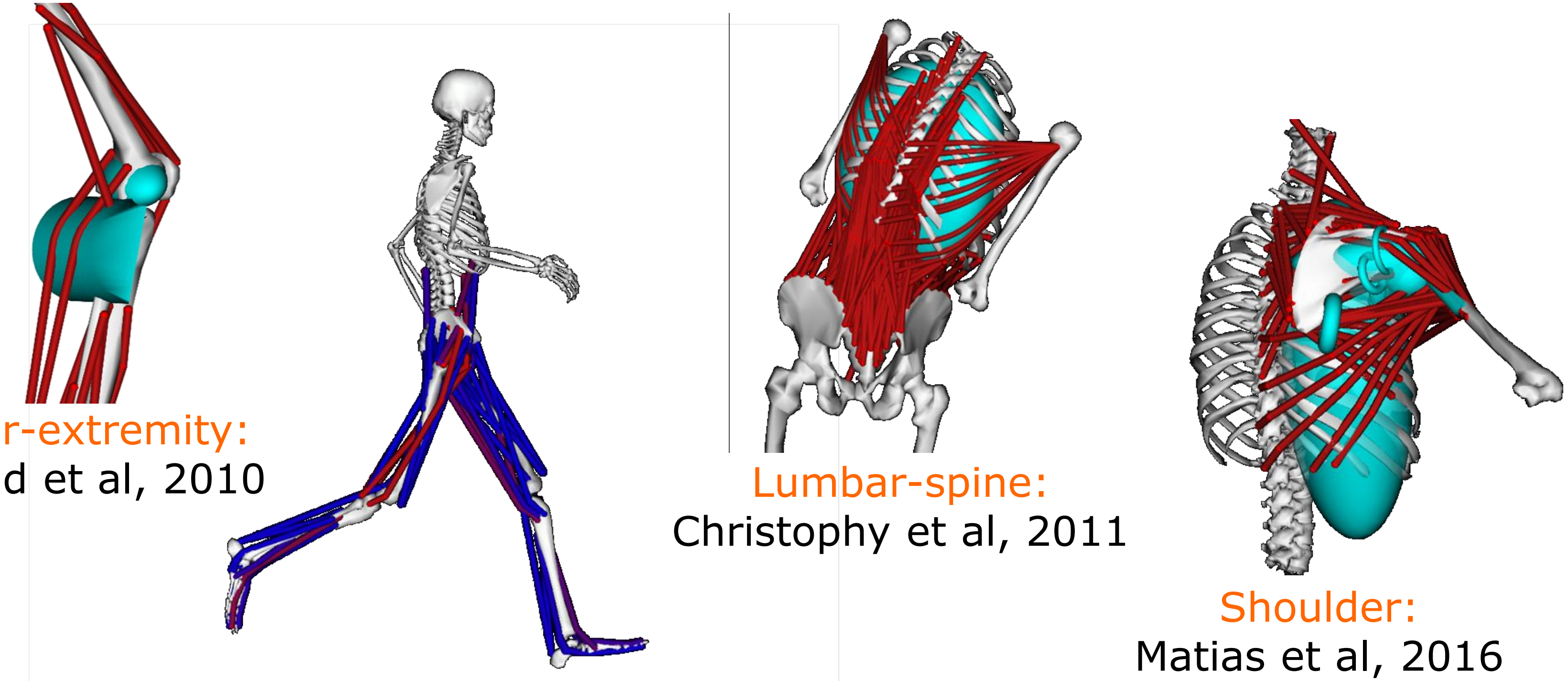
OpenSim workflow

OpenSim enables us to build, exchange, and analyze computer models of the musculoskeletal system and dynamic simulations of movement.



OpenSim model

An OpenSim model represents the dynamics of a system of rigid bodies and joints that are acted upon by forces to produce motion.



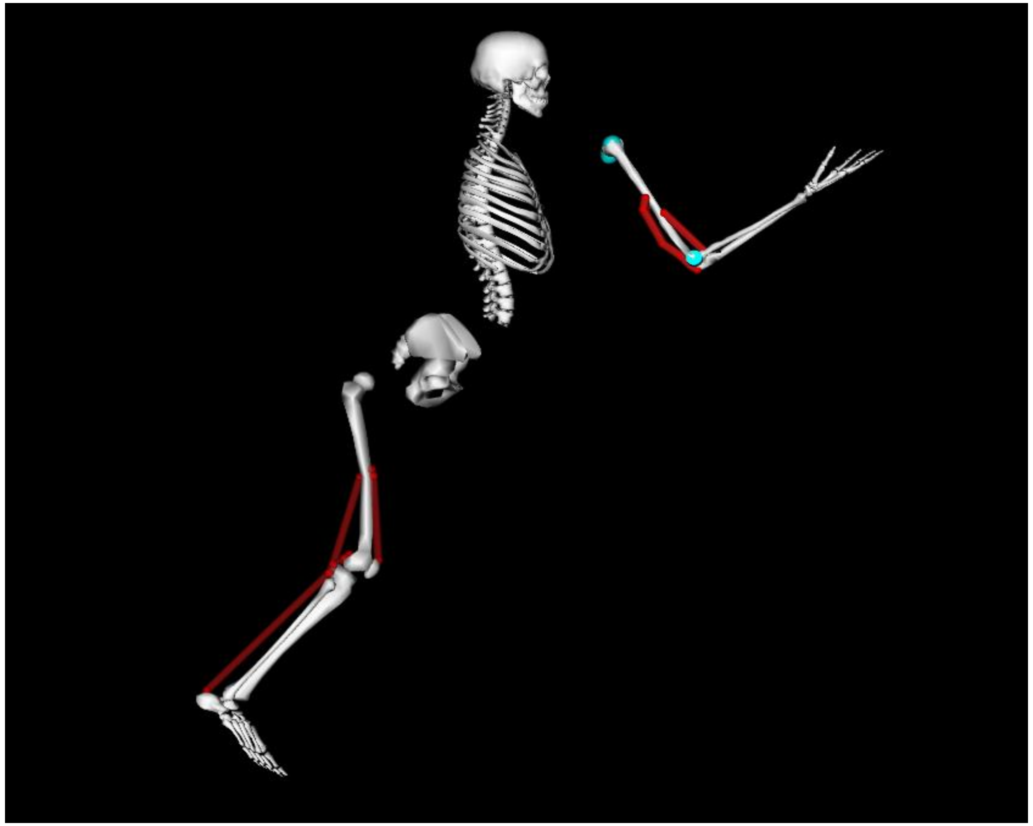
Lower-extremity: Arnold et al, 2010

Running: Hamner et al, 2010

Lumbar-spine: Christophy et al, 2011

Shoulder: Matias et al, 2016

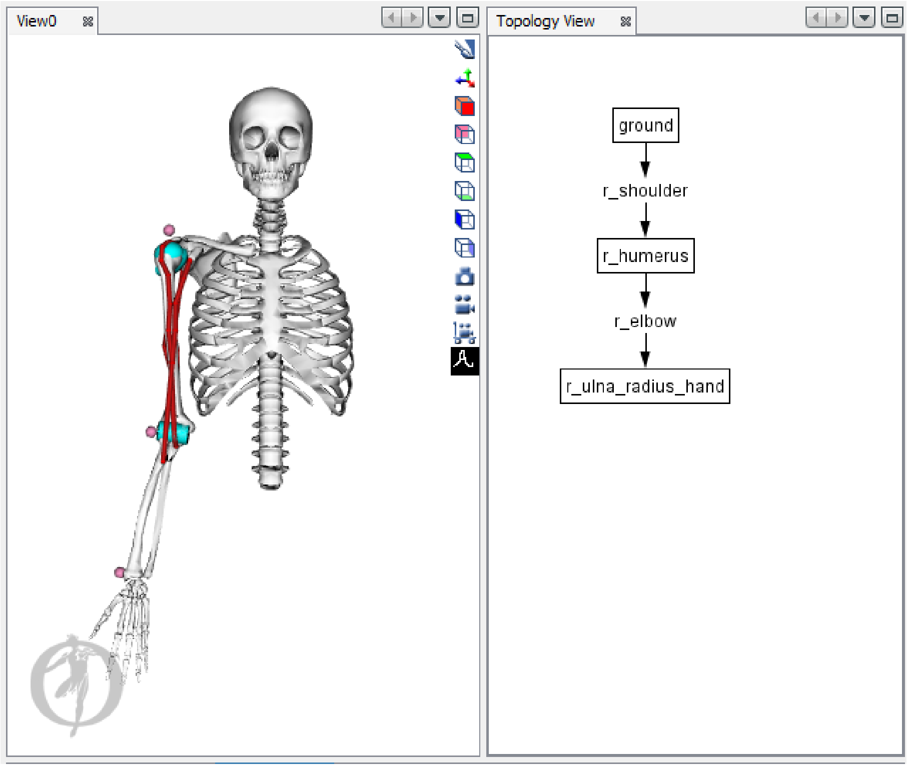
Components of an OpenSim Model

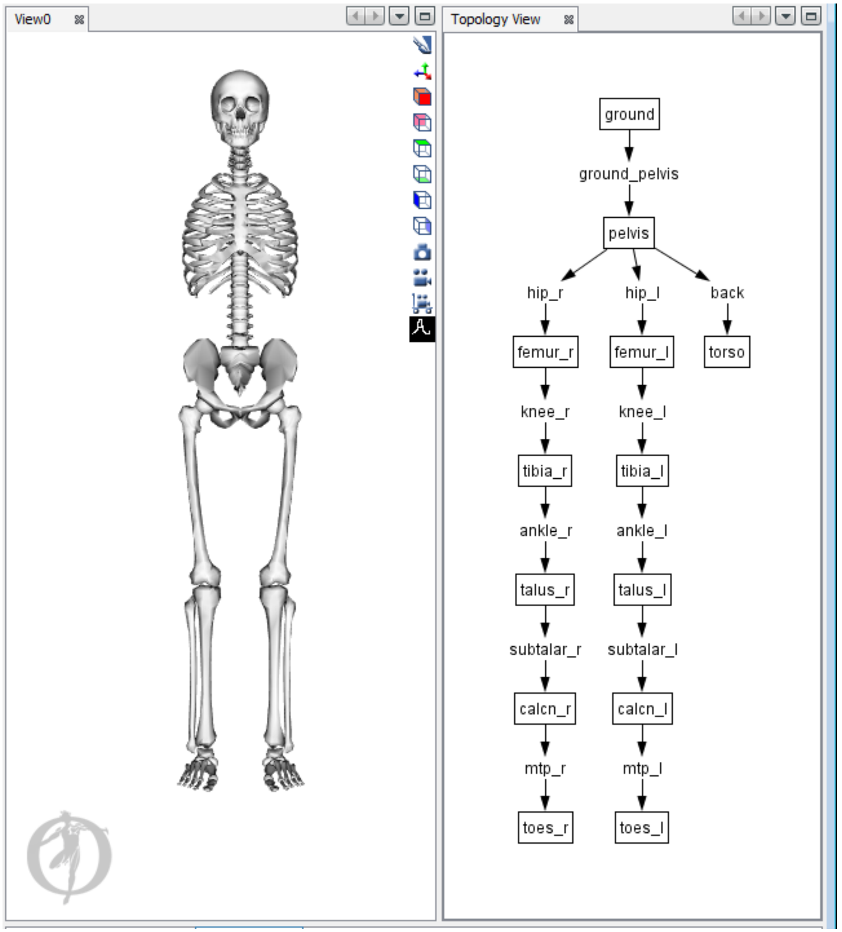


Bodies, joints, constraints, contact geometry,
forces, markers, and controllers

Tree Topology of Multibody Models

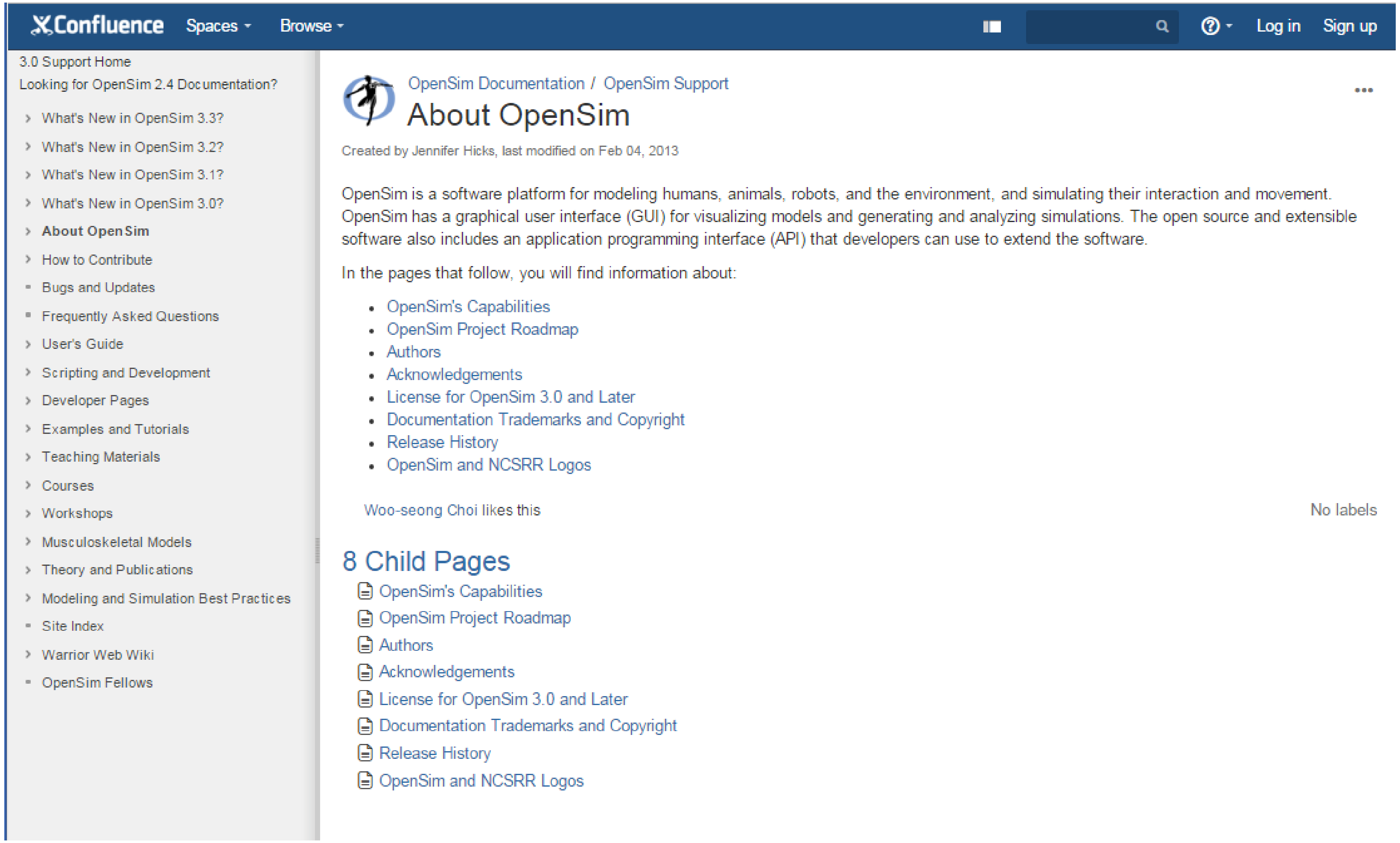
- You can view the topology of your model (Window>topology view).





How to find what you need

- Confluence website

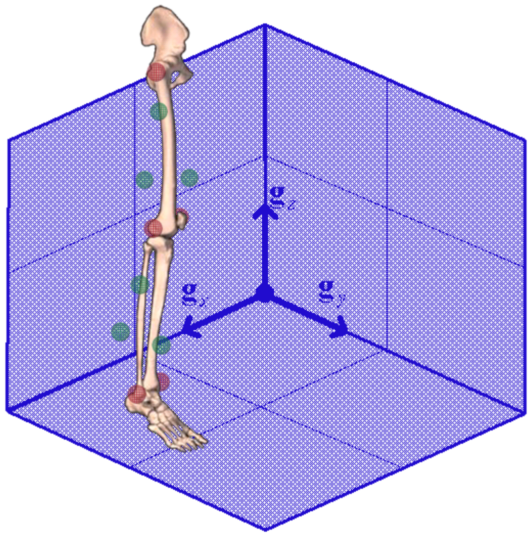


http://simtk-confluence.stanford.edu:8080/display/OpenSim/OpenSim+Support

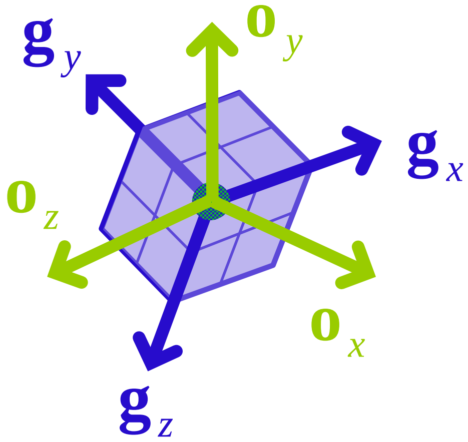
1 Import data in OpenSim

Coordinate Systems

Measure Markers in Lab Coordinate System



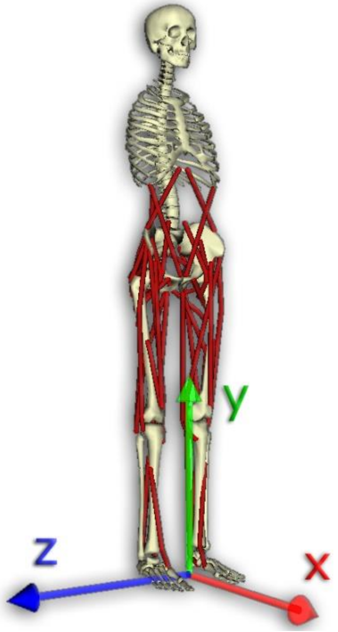
coordinate frame *g*



Rotation Matrix

$$^{A}\mathbf{R}^{B} = \begin{bmatrix} r_{xx} & r_{xy} & r_{xz} \\ r_{yx} & r_{yy} & r_{yz} \\ r_{zx} & r_{zy} & r_{zz} \end{bmatrix}$$

OpenSim Model Coordinate System



coordinate frame *o*

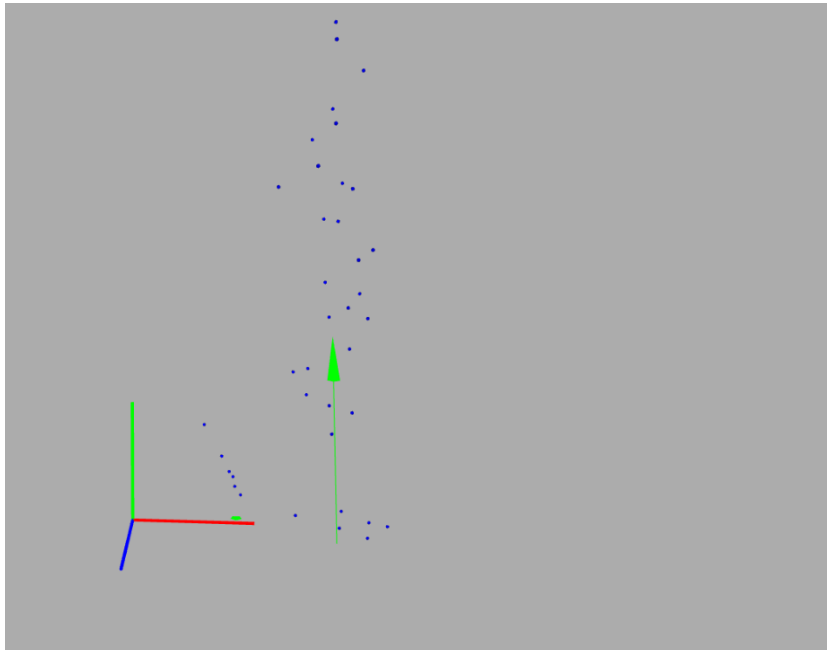
1) Import data in OpenSim – matlab example

...\Functions\ImportData_API.m

```
% c3d import-export
import org.opensim.modeling.*
c3dPath='C:\Users\u0088756\Documents\PostDoc\Data\DataNexus\ExpFriedl\TrapdoorExp\Friedl_v2\Session
c3d = osimC3D(c3dPath,1);       % on is with computation COP, 2 is applied as torques

% Get some stats...
nTrajectories = c3d.getNumTrajectories();
rMakers = c3d.getRate_marker();
nForces = c3d.getNumForces();
rForces = c3d.getRate_force();

% Get Start and end time
t0 = c3d.getStartTime();
tn = c3d.getEndTime();

% Rotate the data
c3d.rotateData('x',-90);
c3d.rotateData('y',180);

% Write the marker and force data to file
c3d.writeTRC('C:\Users\u0088756\Desktop\test_data_markers.trc');
c3d.writeMOT('C:\Users\u0088756\Desktop\test_data_forces.mot');
```

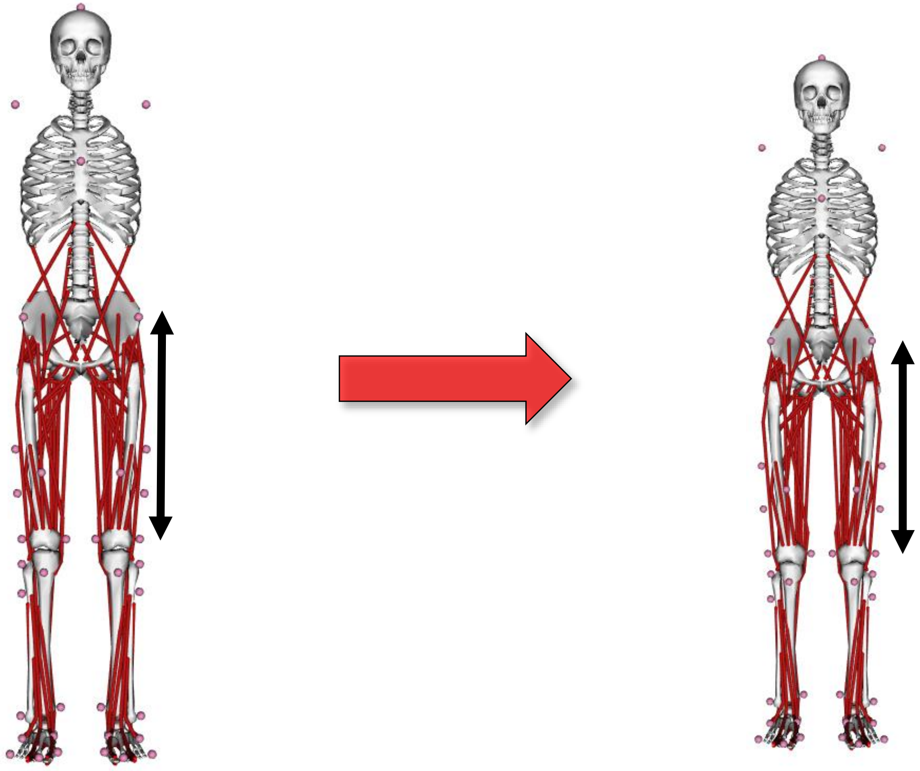
1) Import data in OpenSim – matlab example

Preview experimental data



2) Scaling Tool: Scale Model

Scale tool in OpenSim



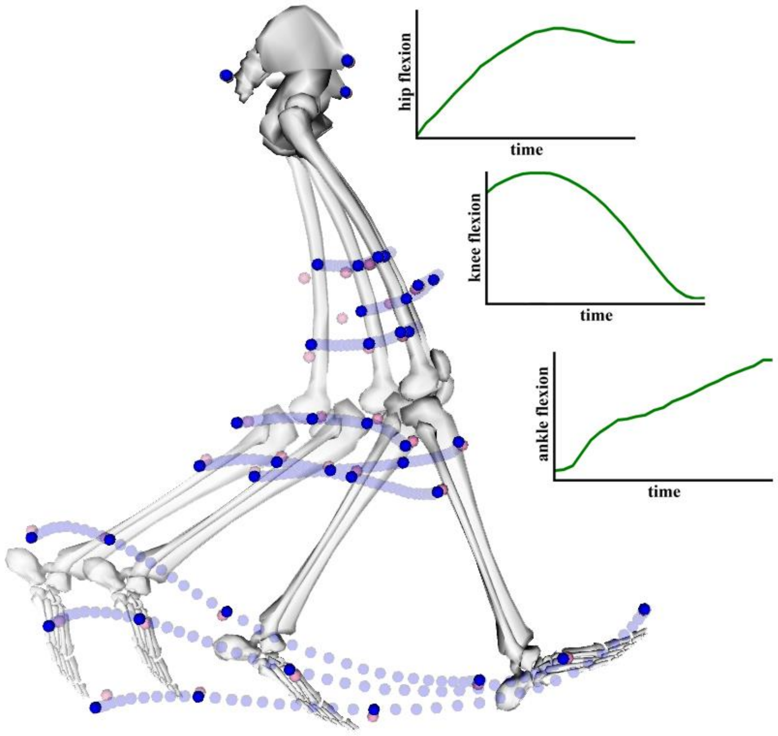
This step was already done

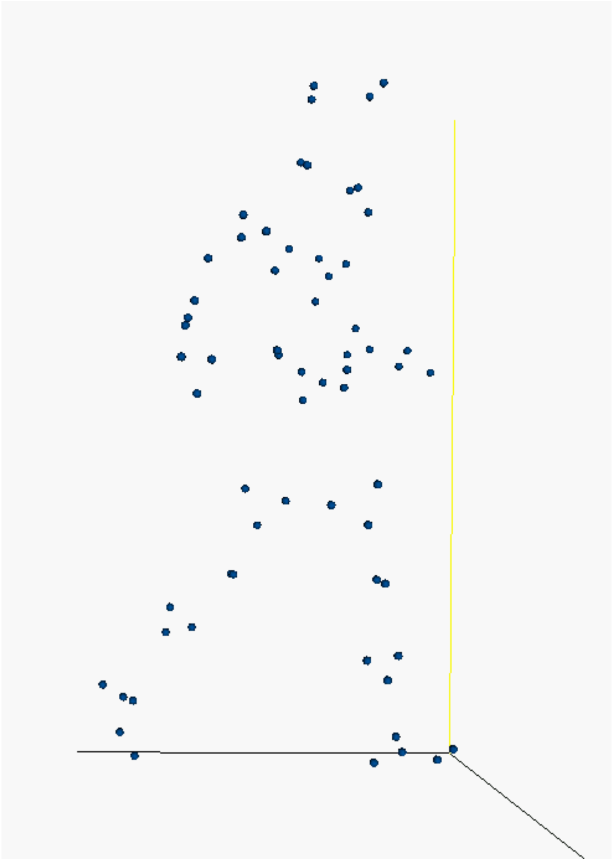
Scaled_model.osim

3) Inverse kinematics

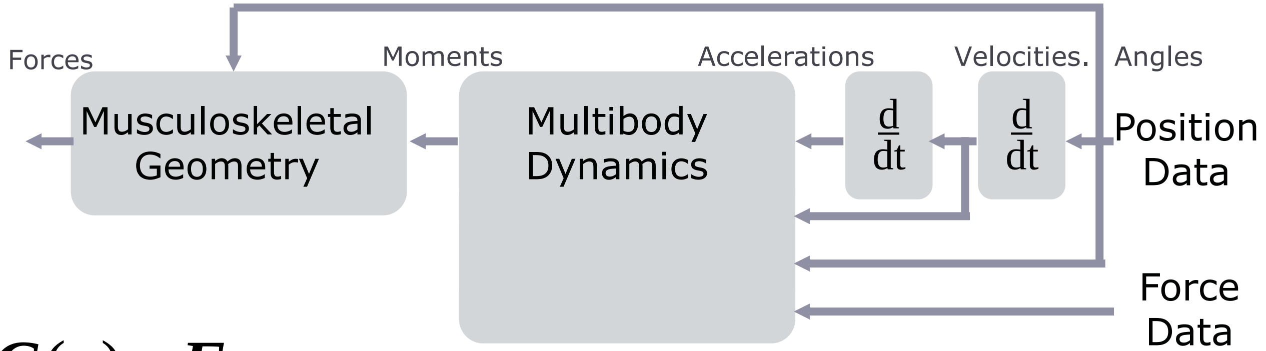
Weighted Least Squares Minimization

$$\min_{\boldsymbol{q}} \left[\sum_{m=1}^{\#\,\text{markers}} w_m \left\| \boldsymbol{x}_m^{\exp} - \boldsymbol{x}_m(\boldsymbol{q}) \right\|^2 \right.$$

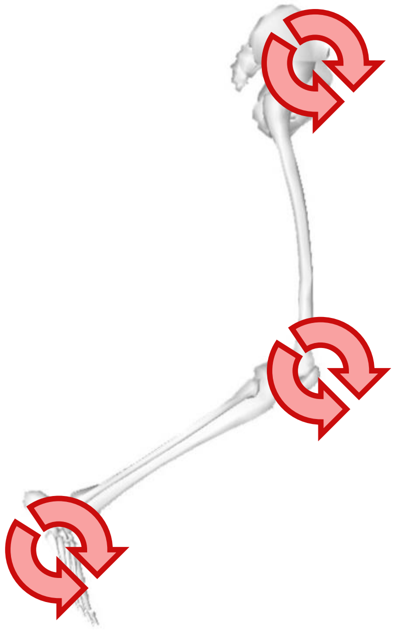


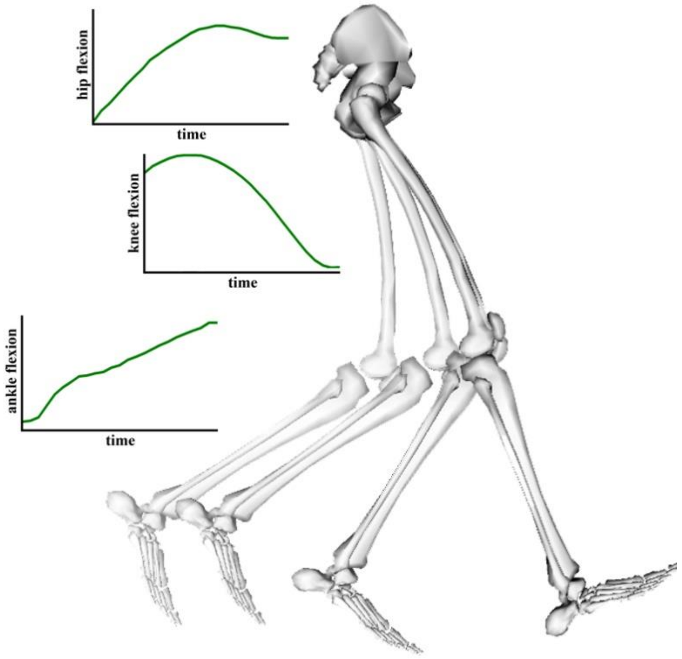


4) Run Inverse dynamics



$$\underbrace{\boldsymbol{\tau}}_{\text{unknowns}} = \underbrace{M(q)\ddot{q} - C(q,\dot{q}) - G(q) - F}_{\text{knowns}}$$





q = generalized coordinates
F = external forces
M(q) = mass matrix
C(q,qd) = coriolis terms
G(q) = gravitational terms

4) Run Inverse dynamics

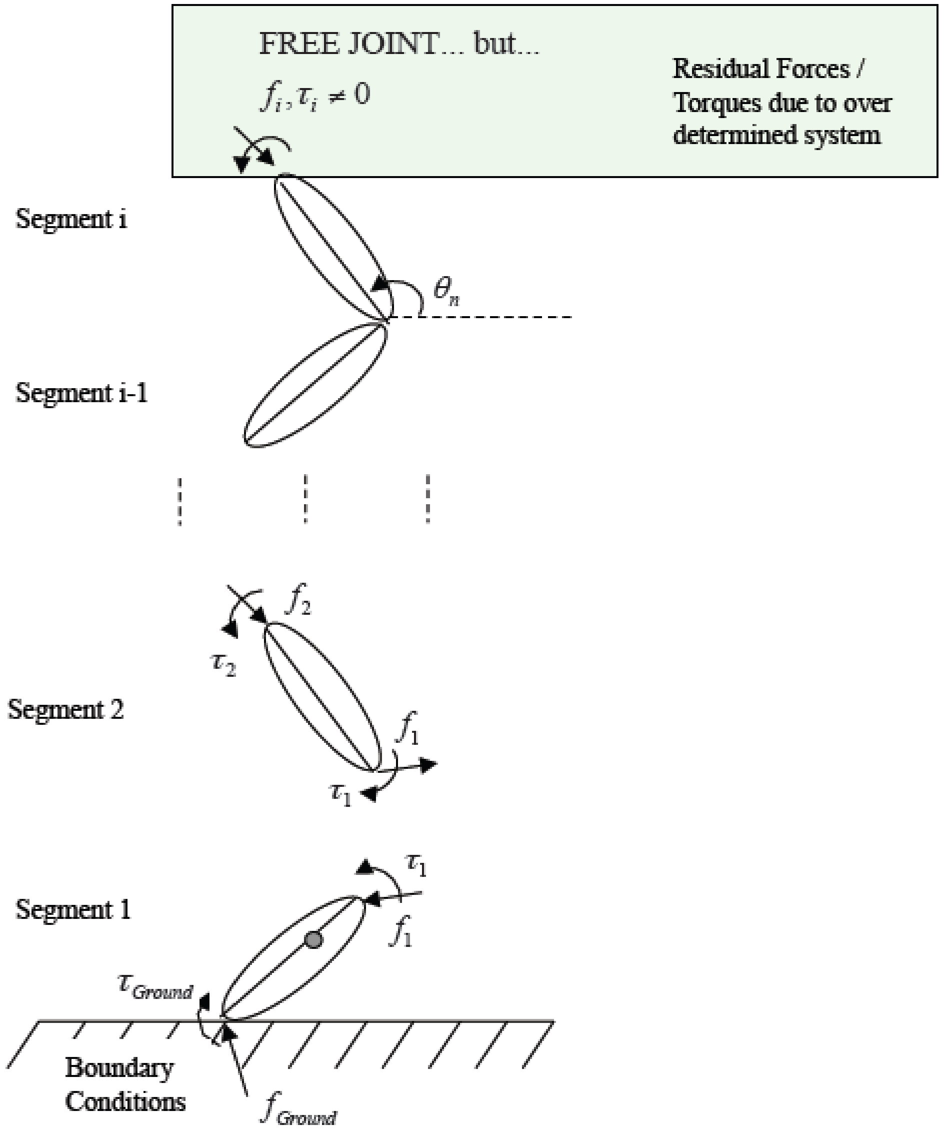
Solved algebraically from the ground up

$$x,\ \dot{x},\ \ddot{x} \qquad \Sigma F_x = m\ddot{x}$$

$$y,\ \dot{y},\ \ddot{y} \Rightarrow \Sigma F_y = m\ddot{y}$$

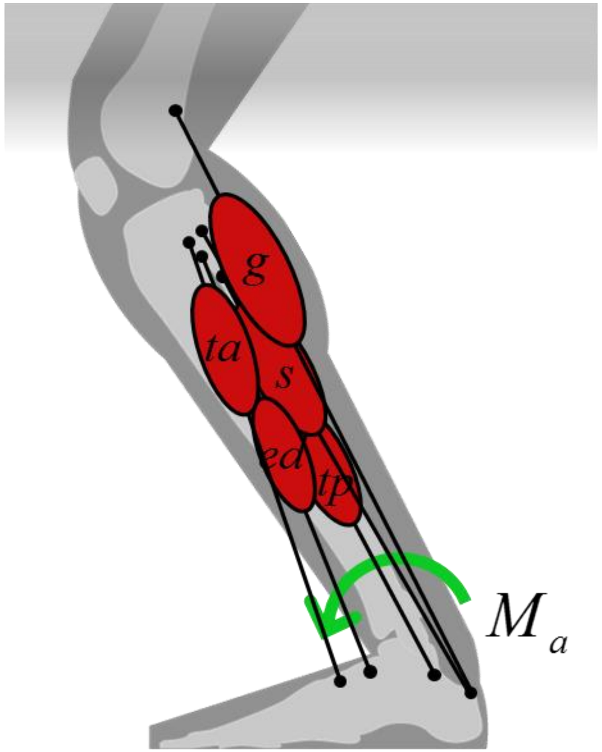
$$\theta,\ \dot{\theta},\ \ddot{\theta} \qquad \Sigma T = I\ddot{\theta}$$

Joint Moments that generate the motion



5) Solve for muscle forces

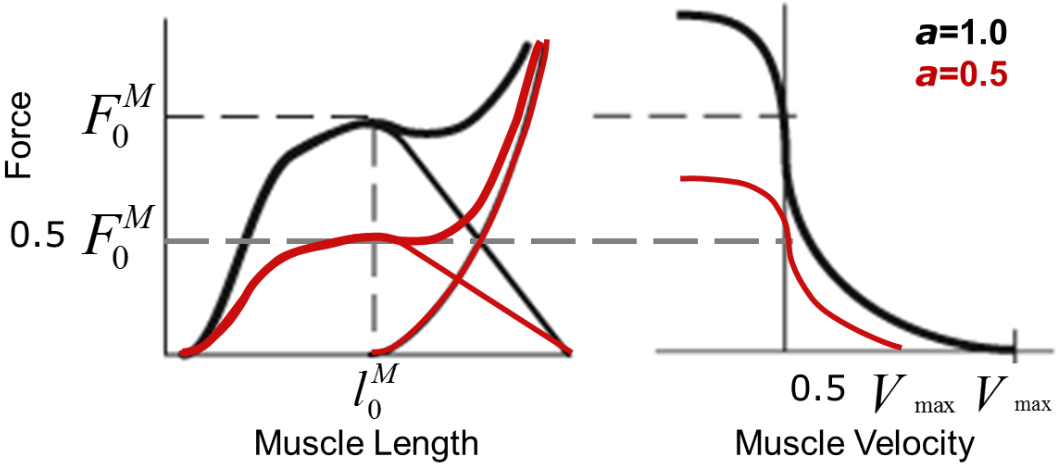
Static optimization



$$\tau = \sum_{i}^{nM} d_i F_i + \tau_{res}$$

With force as a function of:

Muscle fiber length, velocity and activation



6) Limitations of static optimization

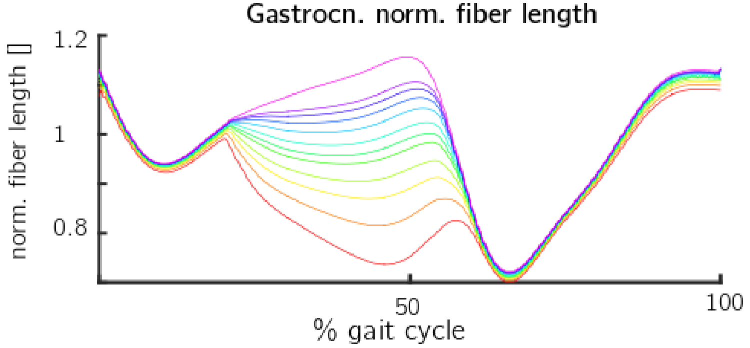
Accounting for contraction dynamics

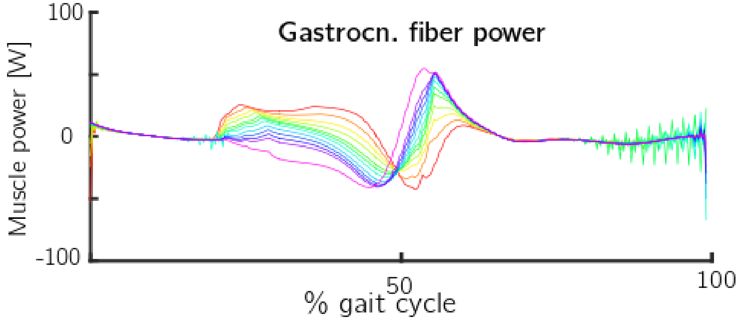
- Activation dynamics ?
- Compliant tendons ?

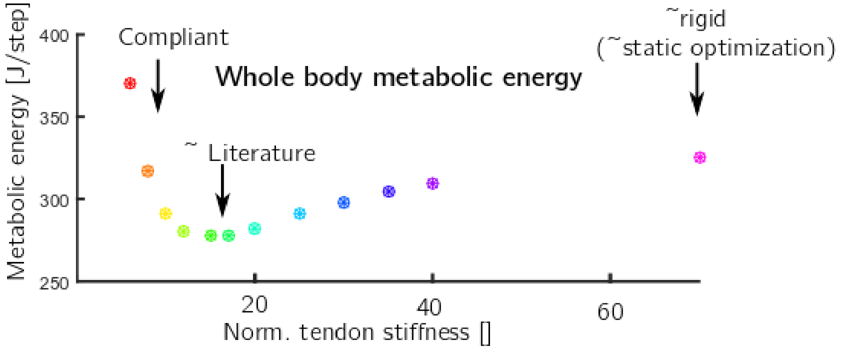
Minizing activations squared ?

…

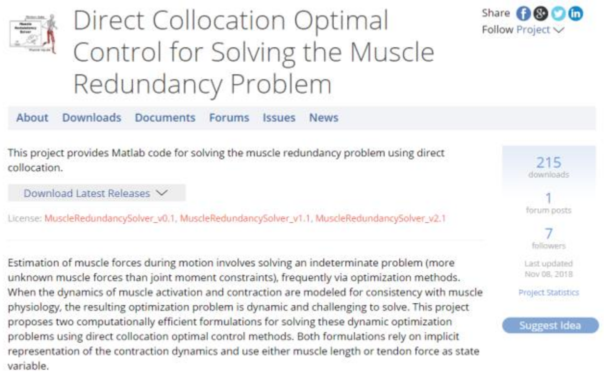
6) Simulation muscle-tendon dynamics







https://simtk.org/projects/optcntrlmuscle


Direct Collocation Optimal Control for Solving the Muscle Redundancy Problem

Share

Follow Project

About Downloads Documents Forums Issues News

This project provides Matlab code for solving the muscle redundancy problem using direct collocation.

Download Latest Releases

License: MuscleRedundancySolver_v0.1, MuscleRedundancySolver_v1.1, MuscleRedundancySolver_v2.1

Estimation of muscle forces during motion involves solving an indeterminate problem (more unknown muscle forces than joint moment constraints), frequently via optimization methods. When the dynamics of muscle activation and contraction are modeled for consistency with muscle physiology, the resulting optimization problem is dynamic and challenging to solve. This project proposes two computationally efficient formulations for solving these dynamic optimization problems using direct collocation optimal control methods. Both formulations rely on implicit representation of the contraction dynamics and use either muscle length or tendon force as state variable.

215 downloads

1 forum posts

7 followers

Last updated Nov 08, 2018

Project Statistics

Suggest Idea