Multibody Dynamics

MAE/BIM 223

Fall 2019

Jason K. Moore

Slides with playable videos:

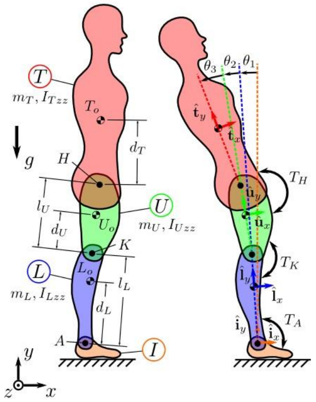
https://docs.google.com/presentation/d/e/2PACX-1vRxb31u7u0A4GDS2ZPe2JDFUMhSOFbdhegev6Xiupr9YPutoSDoK5-GQfhSbl-9i-v4ZtpBhmE1MLSv/pub?start=false&loop=false&delayms=3000

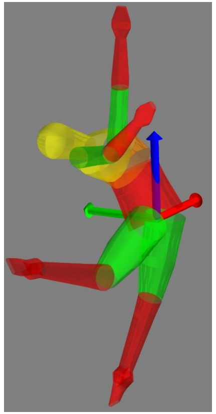
Primary Instructor: Jason K. Moore

BS @ ODU 2004, Fulbright TU Delft 2009, MS/PhD @ UCD 2007/2012

Assistant Professor of Teaching since 2015 @ UC Davis

Teaches: dynamics, vibrations, biomechanics, mech. design, computer graphics, and capstone





Interests

- Single Track Vehicle Dynamics and Control
- Sports Biomechanics
- Alternative Transportation
- Appropriate Technologies
- Engineering Education
- Scientific Computing
- Active in the SciPy community
- Developer of open source software: SymPy, Opty, etc
- Favorite conferences/communities: BMD, Dynamic Walking, ICSC, SciPy, PyCon, PyData

Where have I used multibody dynamics?

- Modeling a variety of bicycles and single track vehicles
- Modeling the biomechanics of human riding a bicycle
- Modeling lower limb walking and running
- Modeling rowing biomechanics
- Standing and balancing biomechanics
- Self balancing motorcycle dynamics
- Inertial measurement dynamics
- Bicycle transportation energetics
- Skiing biomechanics

Projects

Laboratorium of Marvelous Mechanical Motum

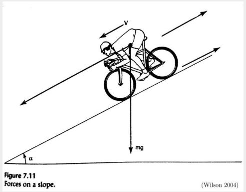
https://mechmotum.github.io/

What can you do with Multibody Dynamics?

Modeling and Simulation of Multibody Systems

Real World

Abstraction

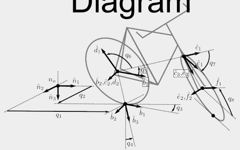


$$\frac{d}{dt}\begin{bmatrix} p_2 \\ q_5 \end{bmatrix} = \begin{bmatrix} -R_3/I_2 & -1/C_5 \\ 1/I_2 & -1/R_6C_5 \end{bmatrix}\begin{bmatrix} p_2 \\ q_5 \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix} e_1(t).$$

Free Body Diagram

Mathematical Model

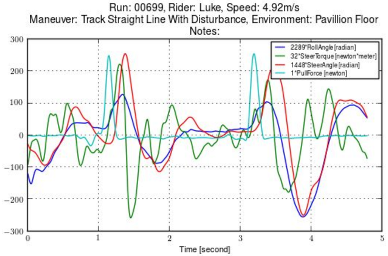
```
% forces acting on e-bicyclist
Fd = 1/2*rho*Cd*A*v^2;  % drag force [N]
Fr = Cr*m*g*cos(alpha);  % rolling resistant [N]
Fg = m*g*sin(alpha);  % gravity force [N]

% evaluate the four 1st order explicit ordinary differential equations
thetadot = omega;
omegadot = (Kt*i - bm*omega - Fd*R - Fr*R - Fg*R) / (2*J + m*R^2);
idot = (-Rw*i - Kt*omega + V)/L;
xdot = v*cos(alpha);

% pack the state derivatives in a vector
sdot = [thetadot; omegadot; idot; xdot];
```

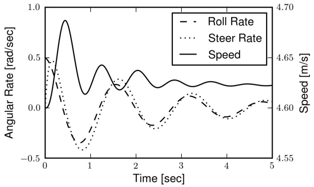


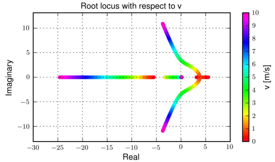
Computational Model

Measurements

Analysis & Simulation





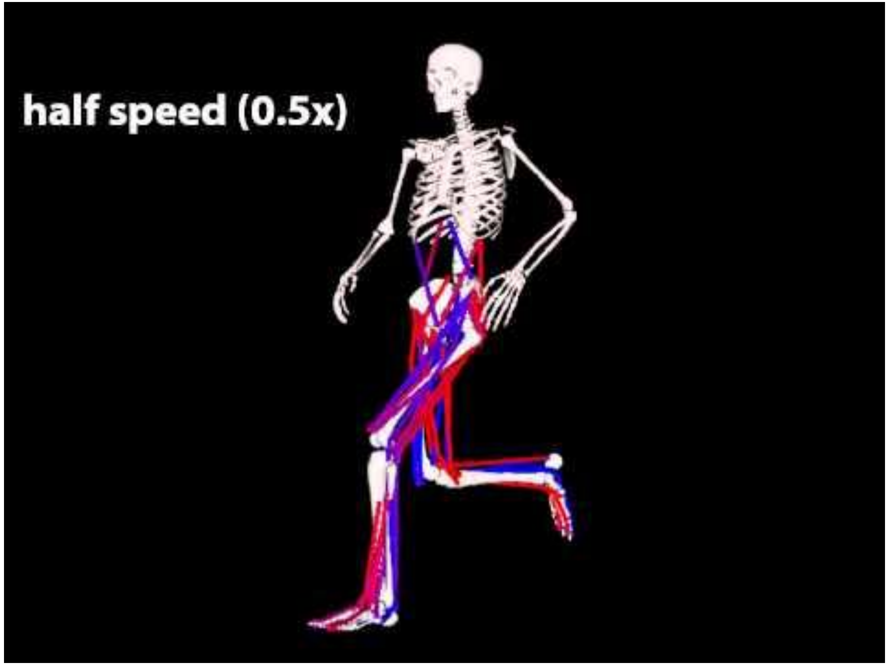


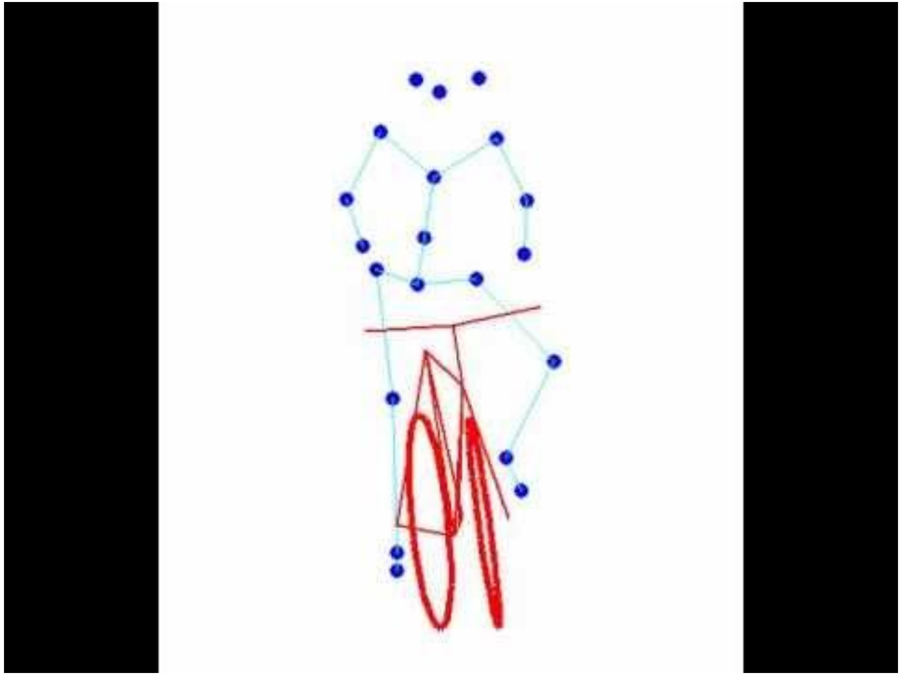
Validation

Measurements of real world

Prediction of real world

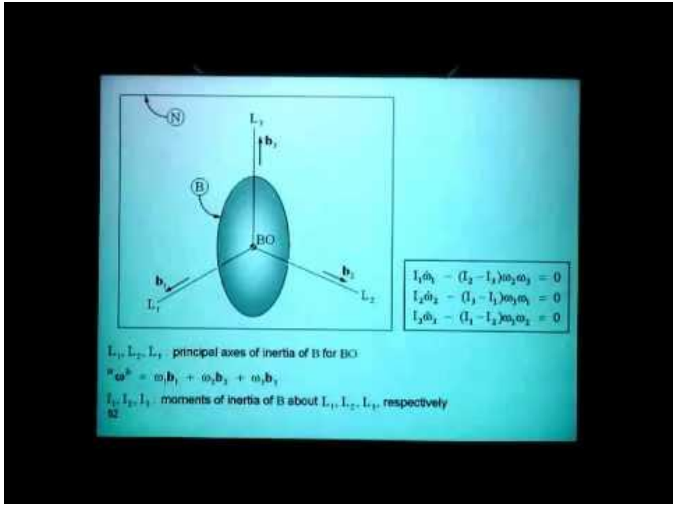
Biomechanics



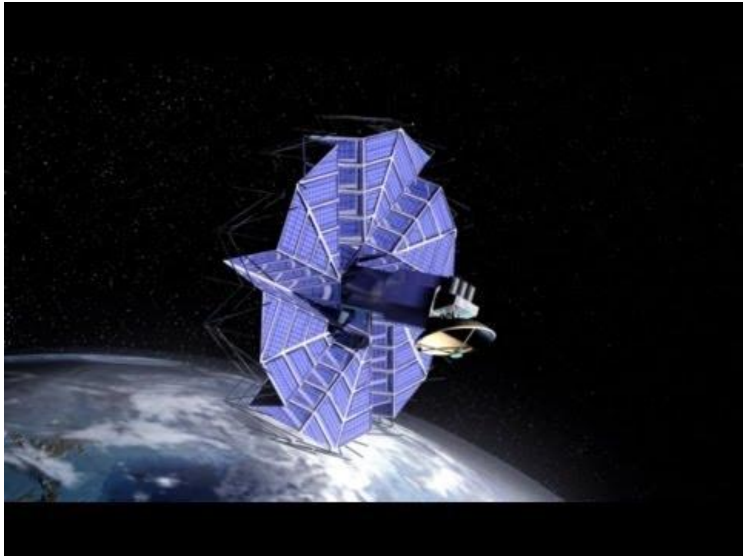


Spacecraft

Explorer 1 Anomaly (start @ 46:00)



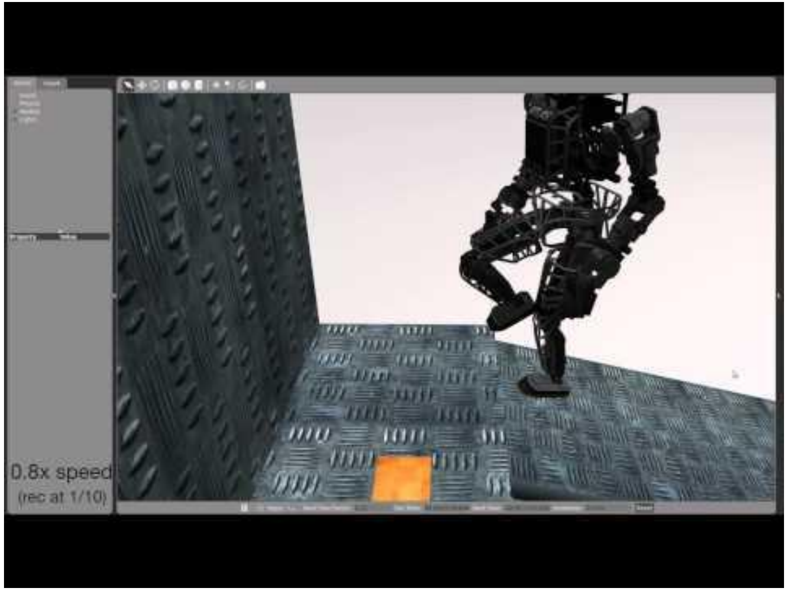
Origami Solar Array Design (BYU)



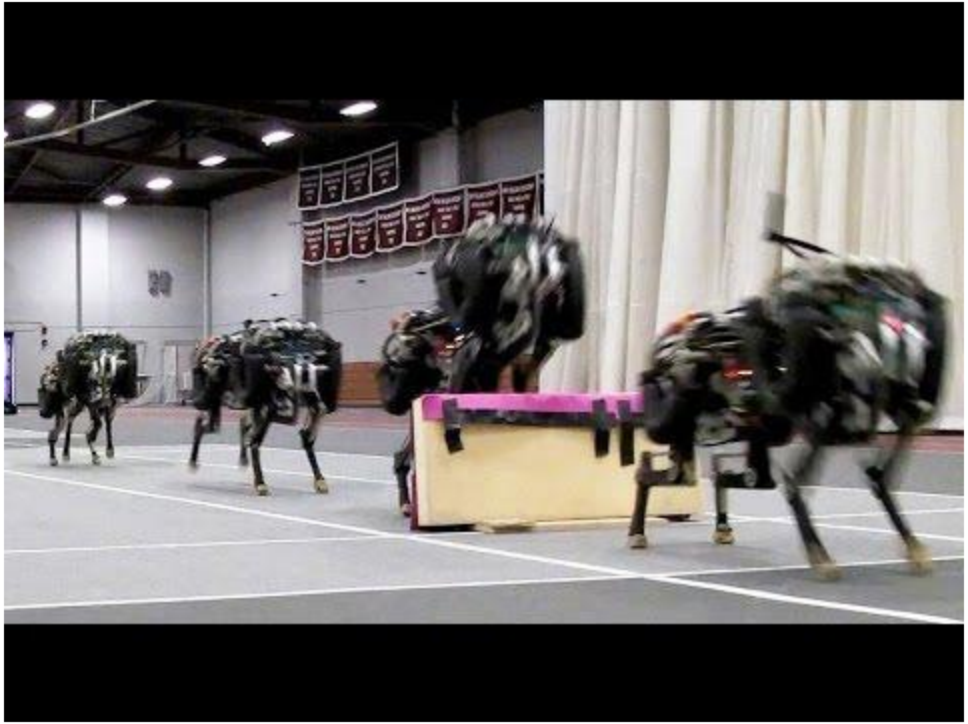
Robotics

Atlas (Boston Dynamics)

Atlas Gazebo Simulation

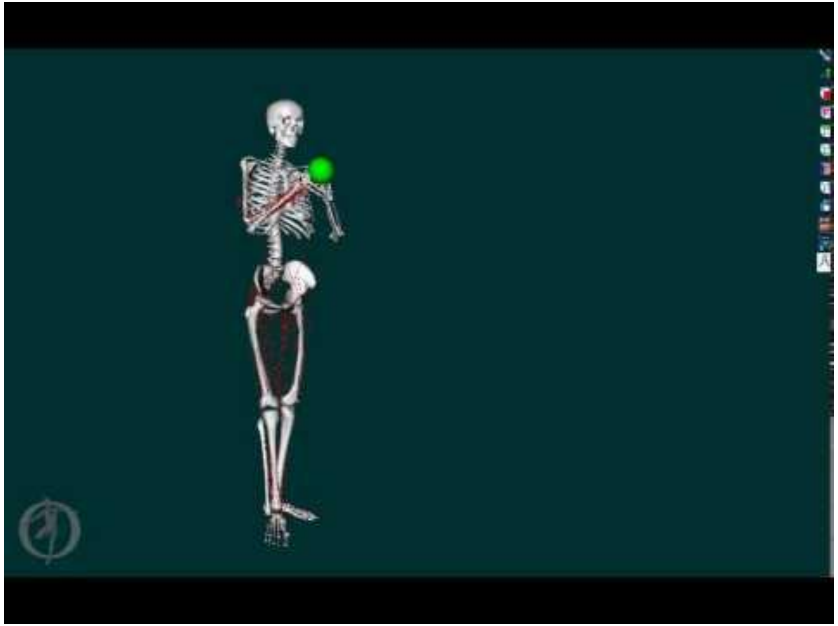


Biomimetic Robotics



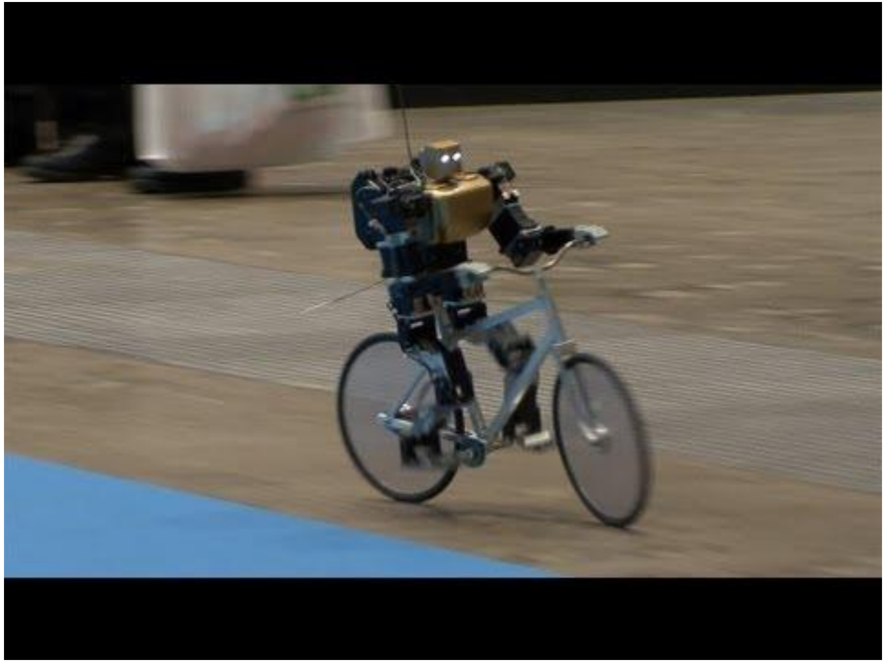
Sports Biomechanics

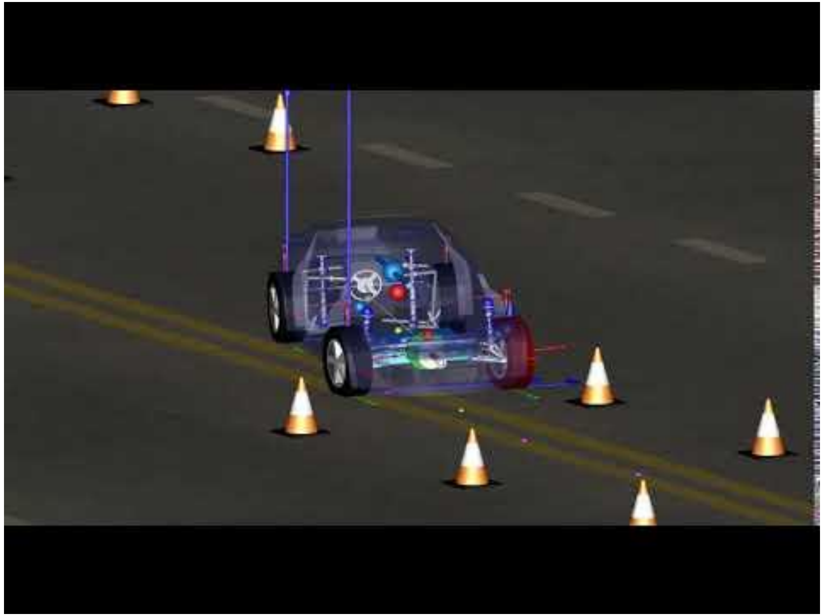
Baseball pitch simulation (Buffi)



Long jump Mocap

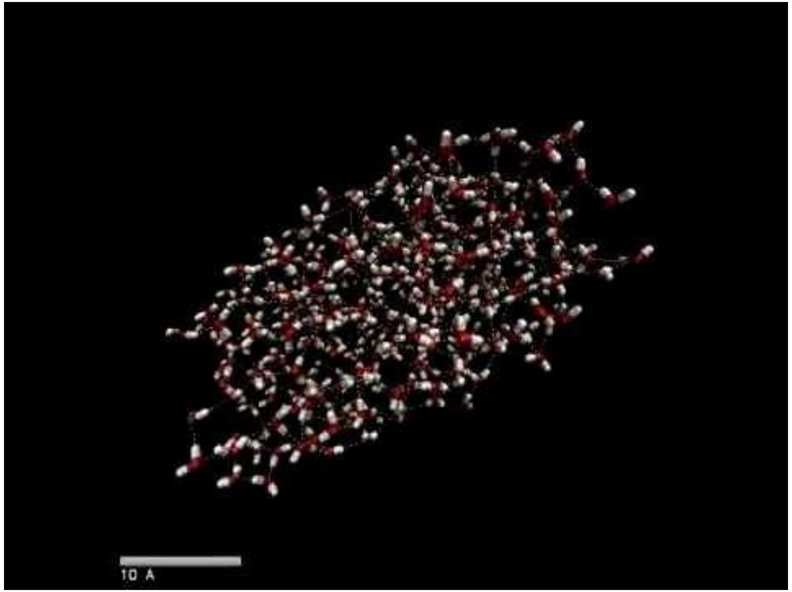
Vehicle Dynamics



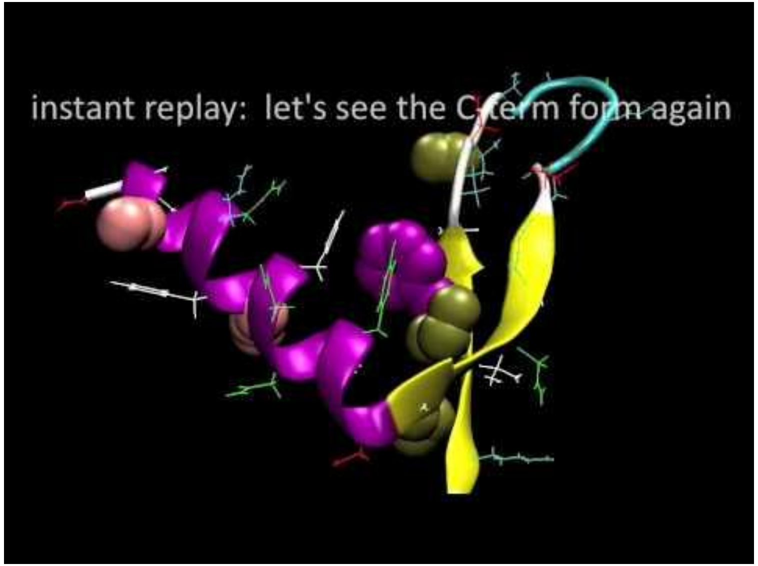


Molecular Dynamics

Ice Crystal Melting



Protein Folding



https://moorepants.github.io/mae223

jupyter.libretexts.org