Quaternion-Based Analytical Inverse Dynamics for the Human Body

Andrew Du
under the mentorship of David Darrow

MIT PRIMES Conference

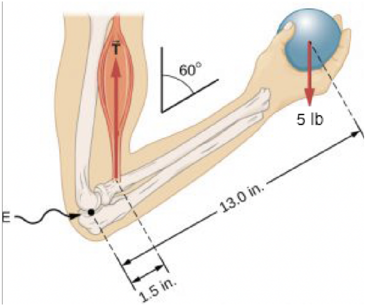
October 16, 2021

Inverse Dynamics for the Human Body

- **Inverse dynamics** is the calculation of joint forces and torques in a model, given certain known parameters.

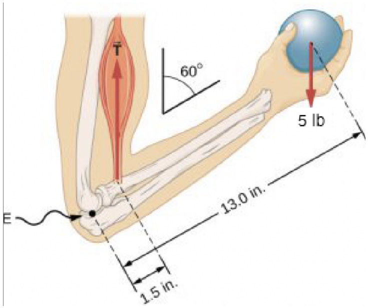
Inverse Dynamics for the Human Body

- **Inverse dynamics** is the calculation of joint forces and torques in a model, given certain known parameters.



Inverse Dynamics for the Human Body

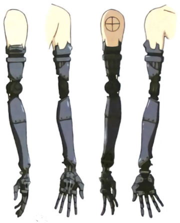
- **Inverse dynamics** is the calculation of joint forces and torques in a model, given certain known parameters.



- It's hard to measure these internal dynamics directly, so we need numerical models.

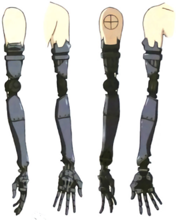
openoregon.pressbooks.pub

Why We Should Care



Prosthetic Design

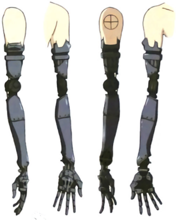
Why We Should Care



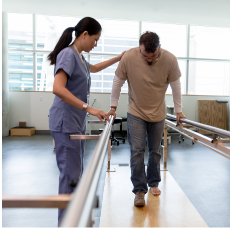
Prosthetic Design

Physical Therapy

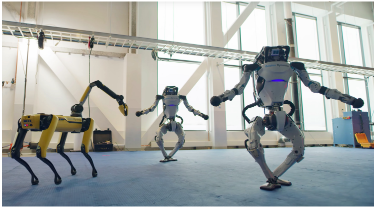
Why We Should Care



Prosthetic Design



Physical Therapy



Human-Inspired Robots

reddit.com/r/VioletEvergarden, static01.nyt.com, engadget.com

Overview

1. Existing methods and models

Overview

1. Existing methods and models
2. Theoretical background

Overview

1. Existing methods and models
2. Theoretical background
3. Our novel method

Some Terminology

Segment: A rigid part of the body that moves as one object.

Some Terminology

Segment: A rigid part of the body that moves as one object.

Distal, Proximal: Describing a segment or joint farther or closer to the torso, respectively.

Some Terminology

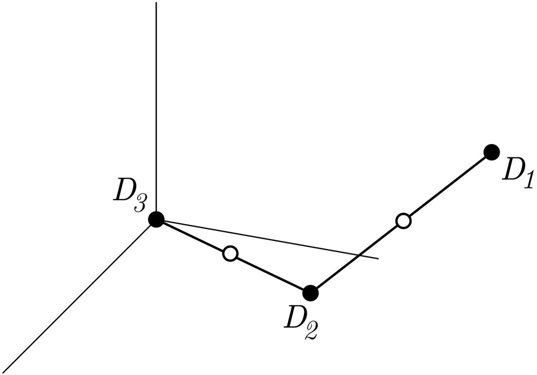
Segment: A rigid part of the body that moves as one object.

Distal, Proximal: Describing a segment or joint farther or closer to the torso, respectively.

ICS, SCS: The *inertial coordinate system* (ICS) and *segment coordinate system* (SCS) are the global and segment-specific coordinate axes, respectively.

The Traditional Way of Doing Things

Normally, we start off with a diagram like this:



The Traditional Way of Doing Things

For each segment, starting from the one farthest from the torso and moving inwards:

1. Calculate force in the ICS at the proximal end of the segment

The Traditional Way of Doing Things

For each segment, starting from the one farthest from the torso and moving inwards:

1. Calculate force in the ICS at the proximal end of the segment
2. Find moment at the proximal end in the SCS

The Traditional Way of Doing Things

For each segment, starting from the one farthest from the torso and moving inwards:

1. Calculate force in the ICS at the proximal end of the segment
2. Find moment at the proximal end in the SCS
3. Transfer force and moment to next segment

The Traditional Way of Doing Things

For each segment, starting from the one farthest from the torso and moving inwards:

1. Calculate force in the ICS at the proximal end of the segment
2. Find moment at the proximal end in the SCS
3. Transfer force and moment to next segment

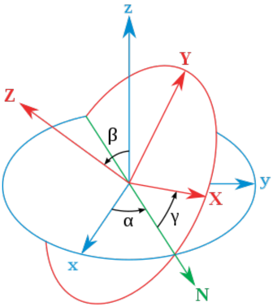
However, there are some shortcomings to this method.

The Orientation Problem

- The usual way of tracking the orientation of each segment uses **Euler angles**

The Orientation Problem

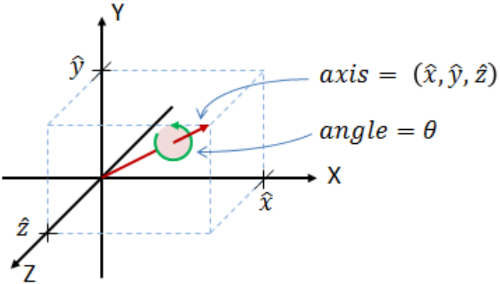
- The usual way of tracking the orientation of each segment uses **Euler angles**
- This structure suffers from singularities, or **gimbal lock**



en.wikipedia.org/wiki/Euler_angles

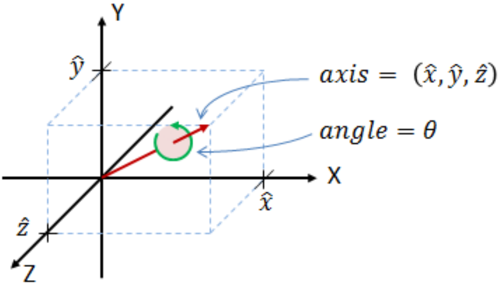
The Quaternion Solution

- Quaternions define a unique transformation and orientation in terms of an axis and an angle



The Quaternion Solution

- Quaternions define a unique transformation and orientation in terms of an axis and an angle



- The corresponding quaternion to such a transformation is

$$q = \cos\left(\frac{\theta}{2}\right) + \sin\left(\frac{\theta}{2}\right)\vec{\boldsymbol{u}}$$

danceswithcode.net/engineeringnotes/quaternions/

The Quaternion Solution

- While Euler angles use three coordinates, quaternions use four—but they avoid gimbal lock

The Quaternion Solution

- While Euler angles use three coordinates, quaternions use four—but they avoid gimbal lock
  - No three coordinate system can do this, for geometric reasons

The Quaternion Solution

- While Euler angles use three coordinates, quaternions use four—but they avoid gimbal lock
  - No three coordinate system can do this, for geometric reasons
- Quaternions are also very convenient for rotating vectors, as we simply **conjugate** them

The Quaternion Solution

- While Euler angles use three coordinates, quaternions use four—but they avoid gimbal lock
  - No three coordinate system can do this, for geometric reasons
- Quaternions are also very convenient for rotating vectors, as we simply **conjugate** them
  - Conjugating a vector simply entails multiplying it by the quaternion and its conjugate, in order: $qvq*$

The One Step Inverse Dynamic Method

- We can use screws to reduce the number of steps per segment

The One Step Inverse Dynamic Method

- We can use screws to reduce the number of steps per segment
- Screws are really just a concatenation of two specific kinds of vectors: a linear one and a related angular one

The One Step Inverse Dynamic Method

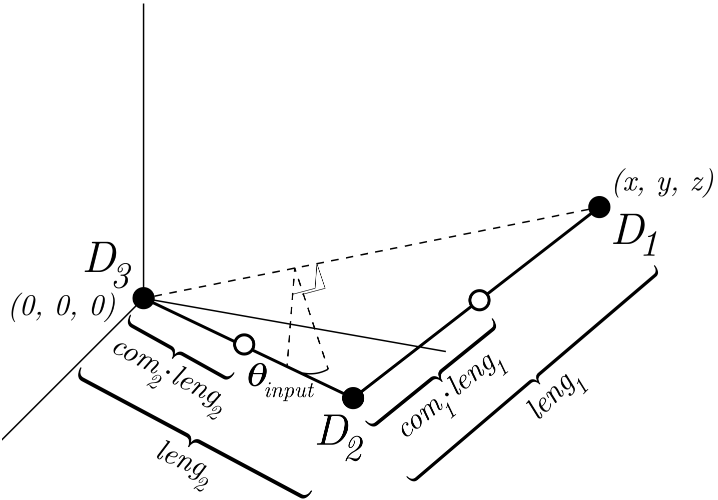
- We can use screws to reduce the number of steps per segment
- Screws are really just a concatenation of two specific kinds of vectors: a linear one and a related angular one
- In our case, we use the **wrench**, which is a force and moment vector combined.

The One Step Inverse Dynamic Method

- We can use screws to reduce the number of steps per segment
- Screws are really just a concatenation of two specific kinds of vectors: a linear one and a related angular one
- In our case, we use the **wrench**, which is a force and moment vector combined.
- Screw algebra gives a single step method of calculating the wrench at each subsequent joint (Dumas, 2004):

$$\begin{bmatrix} \vec{F}_i \\ \vec{M}_i \end{bmatrix} = \begin{bmatrix} m_i \mathbf{E}_{3\times3} & \mathbf{0}_{3\times3} \\ m_i \tilde{\mathbf{c}}_i & \mathbf{I}_i \end{bmatrix} \begin{bmatrix} \vec{a}_i - \vec{g} \\ \vec{\alpha}_i \end{bmatrix} + \begin{bmatrix} \mathbf{0}_{3\times1} \\ \vec{\omega}_i \times \mathbf{I}_i \vec{\omega}_i \end{bmatrix} + \begin{bmatrix} \mathbf{E}_{3\times3} & \mathbf{0}_{3\times3} \\ \tilde{\mathbf{d}}_i & \mathbf{E}_{3\times3} \end{bmatrix} \begin{bmatrix} \vec{F}_{i-1} \\ \vec{M}_{i-1} \end{bmatrix}$$

The Basic Model: A Diagram and a Brief Explanation



The Basic Model: Equations

We assume for now that the arm is not in motion. Then in this framework, all moments and forces can be obtained from a sequence of matrix products:

$$\text{Wrist:} \quad \begin{bmatrix} \vec{\boldsymbol{F}}_1 \\ \vec{\boldsymbol{M}}_1 \end{bmatrix} = \begin{bmatrix} \overrightarrow{\boldsymbol{m_0 g}} \\ \vec{0} \end{bmatrix}$$

$$\text{Elbow:} \quad \begin{bmatrix} \vec{\boldsymbol{F}}_2 \\ \vec{\boldsymbol{M}}_2 \end{bmatrix} = \begin{bmatrix} m_1 \mathbf{E}_{3\times 3} & \mathbf{0}_{3\times 3} \\ m_1 \tilde{\mathbf{c}}_1 & \mathbf{I}_1 \end{bmatrix} \begin{bmatrix} -\mathbf{g} \\ 0 \end{bmatrix} + \begin{bmatrix} \mathbf{E}_{3\times 3} & \mathbf{0}_{3\times 3} \\ \tilde{\mathbf{d}}_1 & \mathbf{E}_{3\times 3} \end{bmatrix} \begin{bmatrix} \vec{\boldsymbol{F}}_1 \\ \vec{\boldsymbol{M}}_1 \end{bmatrix}$$

$$\text{Shoulder:} \quad \begin{bmatrix} \vec{\boldsymbol{F}}_3 \\ \vec{\boldsymbol{M}}_3 \end{bmatrix} = \begin{bmatrix} m_2 \mathbf{E}_{3\times 3} & \mathbf{0}_{3\times 3} \\ m_2 \tilde{\mathbf{c}}_2 & \mathbf{I}_2 \end{bmatrix} \begin{bmatrix} -\mathbf{g} \\ 0 \end{bmatrix} + \begin{bmatrix} \mathbf{E}_{3\times 3} & \mathbf{0}_{3\times 3} \\ \tilde{\mathbf{d}}_2 & \mathbf{E}_{3\times 3} \end{bmatrix} \begin{bmatrix} \vec{\boldsymbol{F}}_2 \\ \vec{\boldsymbol{M}}_2 \end{bmatrix}$$

The Basic Model: Equations

We assume for now that the arm is not in motion. Then in this framework, all moments and forces can be obtained from a sequence of matrix products:

Wrist: $$\begin{bmatrix} \vec{F}_1 \\ \vec{M}_1 \end{bmatrix} = \begin{bmatrix} \overrightarrow{m_0 g} \\ \vec{0} \end{bmatrix}$$

Elbow: $$\begin{bmatrix} \vec{F}_2 \\ \vec{M}_2 \end{bmatrix} = \begin{bmatrix} m_1 \mathbf{E}_{3\times3} & \mathbf{0}_{3\times3} \\ m_1 \tilde{\mathbf{c}}_1 & \mathbf{I}_1 \end{bmatrix} \begin{bmatrix} -\mathbf{g} \\ 0 \end{bmatrix} + \begin{bmatrix} \mathbf{E}_{3\times3} & \mathbf{0}_{3\times3} \\ \tilde{\mathbf{d}}_1 & \mathbf{E}_{3\times3} \end{bmatrix} \begin{bmatrix} \vec{F}_1 \\ \vec{M}_1 \end{bmatrix}$$

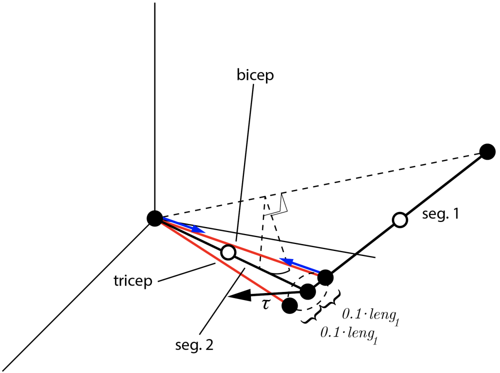
Shoulder: $$\begin{bmatrix} \vec{F}_3 \\ \vec{M}_3 \end{bmatrix} = \begin{bmatrix} m_2 \mathbf{E}_{3\times3} & \mathbf{0}_{3\times3} \\ m_2 \tilde{\mathbf{c}}_2 & \mathbf{I}_2 \end{bmatrix} \begin{bmatrix} -\mathbf{g} \\ 0 \end{bmatrix} + \begin{bmatrix} \mathbf{E}_{3\times3} & \mathbf{0}_{3\times3} \\ \tilde{\mathbf{d}}_2 & \mathbf{E}_{3\times3} \end{bmatrix} \begin{bmatrix} \vec{F}_2 \\ \vec{M}_2 \end{bmatrix}$$

The middle wrench is gone, as are both acceleration vectors:

$$\begin{bmatrix} \vec{F}_i \\ \vec{M}_i \end{bmatrix} = \begin{bmatrix} m_i \mathbf{E}_{3\times3} & \mathbf{0}_{3\times3} \\ m_i \tilde{\mathbf{c}}_i & \mathbf{I}_i \end{bmatrix} \begin{bmatrix} \cancel{\vec{a}}_i - \vec{g} \\ \cancel{\vec{\alpha}}_i \end{bmatrix} + \cancel{\begin{bmatrix} \mathbf{0}_{3\times1} \\ \vec{\omega}_i \times \mathbf{I}_i \vec{\omega}_i \end{bmatrix}} + \begin{bmatrix} \mathbf{E}_{3\times3} & \mathbf{0}_{3\times3} \\ \tilde{\mathbf{d}}_i & \mathbf{E}_{3\times3} \end{bmatrix} \begin{bmatrix} \vec{F}_{i-1} \\ \vec{M}_{i-1} \end{bmatrix}$$

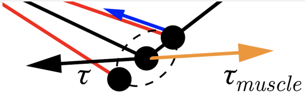
Adding Musculature

Before adding additional segments, we introduce a framework for incorporating muscles into the model.



Each muscle is treated as a tension between two fixed endpoints.

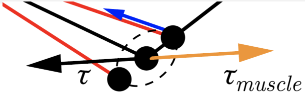
Changes to the Algorithm



We now calculate the wrenches twice.

1. Muscle-free wrench calculation

Changes to the Algorithm

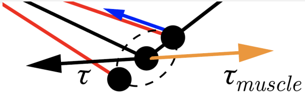


We now calculate the wrenches twice.

1. Muscle-free wrench calculation
2. Find muscle tension

$$\begin{bmatrix} \vec{\boldsymbol{F}}_i \\ \vec{\boldsymbol{M}}_i \end{bmatrix}_m = \sum_{\substack{1 \le j \le M, \\ n_d(j)+1=i}} \begin{bmatrix} \vec{\boldsymbol{u}}_j \cdot \mu_j \\ \vec{\boldsymbol{d}}_{n_d(j)} \cdot (1 - r_d(j)) \times \vec{\boldsymbol{u}}_j \cdot \mu_j \end{bmatrix}$$

Changes to the Algorithm



We now calculate the wrenches twice.

1. Muscle-free wrench calculation
2. Find muscle tension

$$\begin{bmatrix} \vec{F}_i \\ \vec{M}_i \end{bmatrix}_m = \sum_{\substack{1 \le j \le M, \\ n_d(j)+1=i}} \begin{bmatrix} \vec{u}_j \cdot \mu_j \\ \vec{d}_{n_d(j)} \cdot (1 - r_d(j)) \times \vec{u}_j \cdot \mu_j \end{bmatrix}$$

3. Recalculate the wrenches with a modified equation:

$$\begin{bmatrix} \vec{F}_{i+1} \\ \vec{M}_{i+1} \end{bmatrix} = \begin{bmatrix} m_i \mathbf{E}_{3\times 3} & \mathbf{0}_{3\times 3} \\ m_i \tilde{\mathbf{c}}_i & \mathbf{I}_i \end{bmatrix} \begin{bmatrix} -\mathbf{g} \\ 0 \end{bmatrix} + \begin{bmatrix} \mathbf{E}_{3\times 3} & \mathbf{0}_{3\times 3} \\ \tilde{\mathbf{d}}_i & \mathbf{E}_{3\times 3} \end{bmatrix} \begin{bmatrix} \vec{F}_i \\ \vec{M}_i \end{bmatrix} + \begin{bmatrix} \vec{F}_{i+1} \\ \vec{M}_{i+1} \end{bmatrix}_m$$

Adding the Hand

With our current setup, adding the hand and fingers encounters a few difficulties:

Adding the Hand

With our current setup, adding the hand and fingers encounters a few difficulties:

- Our current algorithm does not have a simple way to let segments converge

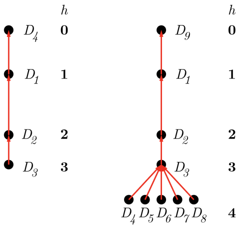
Adding the Hand

With our current setup, adding the hand and fingers encounters a few difficulties:

- Our current algorithm does not have a simple way to let segments converge
- There are exponentially more possible ways to hold an object as segments increase in number

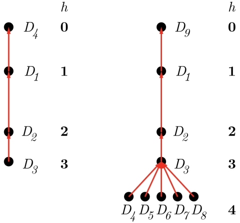
The New Indices

To allow for the implementation of a nonlinear structure, we use a tree:



The New Indices

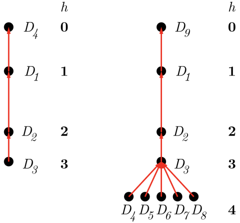
To allow for the implementation of a nonlinear structure, we use a tree:



- Our indices are now simply a labelling serving as a way to distinguish points

The New Indices

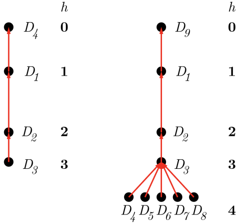
To allow for the implementation of a nonlinear structure, we use a tree:



- Our indices are now simply a labelling serving as a way to distinguish points
- We create a hierarchy based on the number of edges between the shoulder and each point.

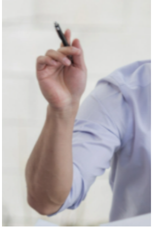
The New Indices

To allow for the implementation of a nonlinear structure, we use a tree:



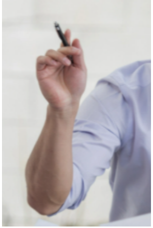
- Our indices are now simply a labelling serving as a way to distinguish points
- We create a hierarchy based on the number of edges between the shoulder and each point.
- Calculations now iterate along each hierarchy number

Orienting the Segments



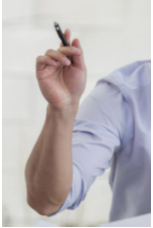
- Each additional segment we add to the model creates more degrees of freedom

Orienting the Segments



- Each additional segment we add to the model creates more degrees of freedom
- The human body usually uses the same orientation to hold an object

Orienting the Segments

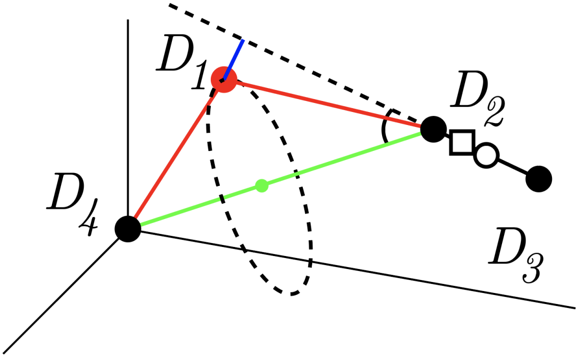


- Each additional segment we add to the model creates more degrees of freedom
- The human body usually uses the same orientation to hold an object
- We create a rudimentary way of predicting how the arm will naturally position itself

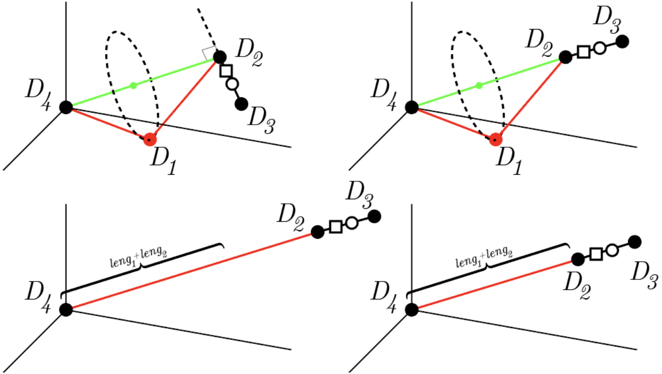
Simu Liu Stock Photo

Orienting the Segments

The gist of it is that we find whatever orientation minimizes bending at the wrist:



Orienting the Segments



Recap

- Quaternions in place of Euler angles

Recap

- Quaternions in place of Euler angles
- Screw algebra for efficiency

Recap

- Quaternions in place of Euler angles
- Screw algebra for efficiency
- Muscle integration

Recap

- Quaternions in place of Euler angles
- Screw algebra for efficiency
- Muscle integration
- Implementation of the hand

Acknowledgements

I'd like to thank David Darrow as my mentor for the PRIMES project, Dr. Daniel Nolte for suggesting the topic for this project, as well as Dr. Tanya Khovanova, Prof. Pavel Etingof, and Dr. Slava Gerovitch for running the PRIMES research program.