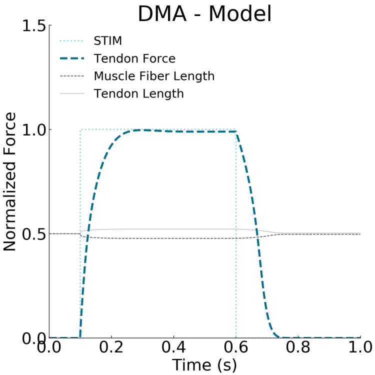
Neuromechanics of Human Motion

Limb Kinematics

Joshua Cashaback, PhD

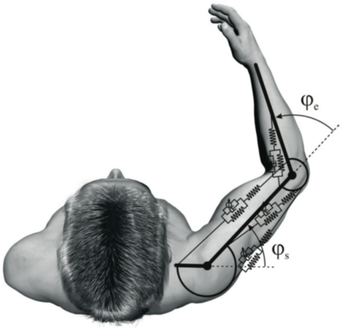
Recap — Muscle Modelling



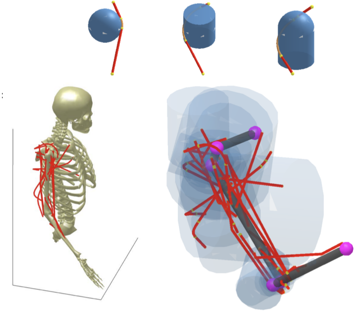
Recap — Cross-bridge vs. Hill Model

1. Hill Models
   a. combine equations to solve muscle force (F_{MF})
   b. fits data well
2. Cross-Bridge Models
   a. More macroscopic variables
   b. Emergent phenomena

Recap — Musculoskeletal Model



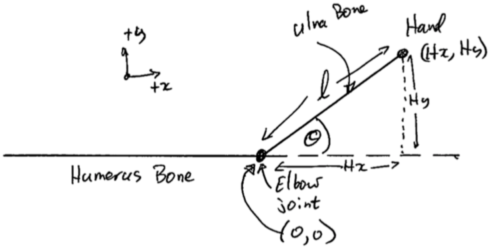
Schematic of "full-blown" musculoskeletal model described in Kistemaker et al. (2010).



Lecture Objectives — Kinematics

1. 1DOF
2. Forward Kinematics
3. Jacobian: $J(\theta)$, and its time derivative: $\dot{J(\theta)}$
   - relationship between joint space and hand space
   - velocities and accelerations
4. Inverse Kinematics
5. 2DOF
6. Redundancy
7. Minimum Jerk Trajectories
8. Endpoint Variance

Elbow and Hand



Schematic of a simple kinematic model of the elbow joint

θ: elbow angle

l: length of lower arm to hand midpoint (*assume a rigid wrist)

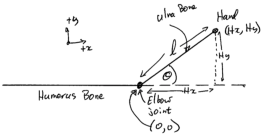
H_x, H_y: hand coordinates; E_x, E_y: elbow coordinates

Forward Kinematics — 1DOF

Forward Kinematics

Forward Kinematics

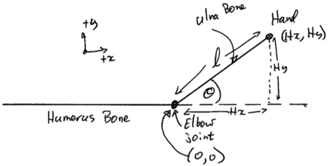
Go from intrinsic variable (joint space) to extrinsic variables (hand space)



Schematic of a simple kinematic model of the elbow joint

1. Position
2. Velocity
3. Acceleration
4. further derivatives (e.g., jerk, ...)

Hand Position — 1 DOF



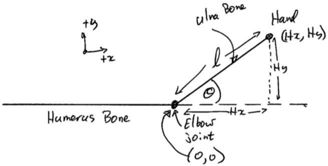
Schematic of a simple kinematic model of the elbow joint

$$H_x = l \cdot cos(\theta)$$

$$H_y = l \cdot sin(\theta)$$

SOH-CAH-TOA

Hand Position — 1 DOF



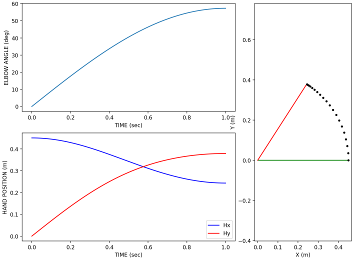
Schematic of a simple kinematic model of the elbow joint

$$l = 0.46m; \theta = 35^o$$

$$H_x = l \cdot cos(\theta) = 0.46 \cdot cos\left(\frac{35\pi}{180}\right) = 0.38m$$

$$H_y = l \cdot sin(\theta) = 0.46 \cdot sin\left(\frac{35\pi}{180}\right) = 0.26m$$

Hand Position — 1 DOF



forward: record elbow angle and calculate hand position

inverse: record hand position and calculate elbow angle

$$l = 0.45m; \theta(rads) = sin(2\pi t_i/4); t_i = linspace(0, 1, 200)$$

Hand Velocity — 1 DOF

Goal: relate joint velocity ($\frac{d\theta}{dt} = \dot{\theta}$) to hand velocity ($\frac{dH}{dt} = \dot{H}$)

Hand Velocity — 1 DOF

Goal: relate joint velocity ($\frac{d\theta}{dt} = \dot{\theta}$) to hand velocity ($\frac{dH}{dt} = \dot{H}$)

$$\frac{dH}{dt} = ? \cdot \frac{d\theta}{dt}$$

Hand Velocity — 1 DOF

Goal: relate joint velocity ($\frac{d\theta}{dt} = \dot{\theta}$) to hand velocity ($\frac{dH}{dt} = \dot{H}$)

$$\frac{dH}{dt} = ? \cdot \frac{d\theta}{dt}$$

We can use the Chain Rule! $e.g.$, $\frac{dz}{dx} = \frac{dz}{dy} \cdot \frac{dy}{dx}$

Hand Velocity — 1 DOF

Goal: relate joint velocity ($\frac{d\theta}{dt} = \dot{\theta}$) to hand velocity ($\frac{dH}{dt} = \dot{H}$)

$$\frac{dH}{dt} = ? \cdot \frac{d\theta}{dt}$$

We can use the Chain Rule! $e.g.$, $\frac{dz}{dx} = \frac{dz}{dy} \cdot \frac{dy}{dx}$

$$\frac{dH}{dt} = \frac{dH}{d\theta} \cdot \frac{d\theta}{dt}$$

Hand Velocity — 1 DOF

Goal: relate joint velocity ($\frac{d\theta}{dt} = \dot{\theta}$) to hand velocity ($\frac{dH}{dt} = \dot{H}$)

$$\frac{dH}{dt} = ? \cdot \frac{d\theta}{dt}$$

We can use the Chain Rule! $e.g.$, $\frac{dz}{dx} = \frac{dz}{dy} \cdot \frac{dy}{dx}$

$$\frac{dH}{dt} = \frac{dH}{d\theta} \cdot \frac{d\theta}{dt}$$

$\frac{dH}{d\theta}$ is known as a Jacobian (often written as $J(\theta)$), which is a matrix of first order, partial derivates (Click Me: Wikipedia).

The Jacobian

Remember that

$$H_x = l \cdot cos(\theta)$$

$$H_y = l \cdot sin(\theta)$$

From this information, we can calculate the Jacobian

$$J(\theta) = \frac{dH}{d\theta} = \begin{bmatrix} \frac{\partial Hx}{\partial \theta} \\ \frac{\partial Hy}{\partial \theta} \end{bmatrix}$$

$$J(\theta) = \frac{dH}{d\theta} = \begin{bmatrix} -l \cdot sin(\theta) \\ l \cdot cos(\theta) \end{bmatrix}$$

Thats it! Now we can calculate $\dot{H}$ since we have $J(\theta)$ and $\dot{\theta}$.

*reminder: $\frac{dcos(\theta)}{d\theta} = -sin(\theta)$ and $\frac{dsin(\theta)}{d\theta} = cos(\theta)$

$J(\theta)$ with Sympy — Python Code

```
#### 1DOF Jacobian ####
# import sympy
from sympy import *
# define these variables as symbolic (not numeric)
a,l = symbols('a l')
# forward kinematics for Hx and Hy
hx = l*cos(a)
hy = l*sin(a)
# use sympy diff() to get partial derivatives for Jacobian matrix
J11 = diff(hx,a)
J12 = diff(hy,a)
print(J11)
print(J12)
```

Pro Tip: run sympy in a separate script from other coding

Hand Velocity — 1 DOF

$\dot{H} = J(\theta) \cdot \dot{\theta}$

equivelently,

$\frac{dH}{dt} = \frac{dH}{d\theta} \cdot \frac{d\theta}{dt}$

and in expanded form

$$\begin{bmatrix} \dot{H}x \\ \dot{H}y \end{bmatrix} = \begin{bmatrix} -l \cdot sin(\theta) \\ l \cdot cos(\theta) \end{bmatrix} \begin{bmatrix} \dot{\theta} \end{bmatrix}$$

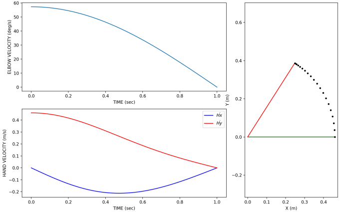
Example: $\begin{bmatrix} -1.31 \\ 1.88 \end{bmatrix} = \begin{bmatrix} -0.46 \cdot sin(\frac{35\pi}{180}) \\ 0.46 \cdot cos(\frac{35\pi}{180}) \end{bmatrix} \begin{bmatrix} 5 \end{bmatrix}$

University of Delaware

Hand Velocity — Python Code

```
# Hand Velocity and Hand Acceleration - matrix multiplication in python
import numpy as np
import math
l = 0.46
Theta = 35 * math.pi / 180.
Theta_dot = np.array([[5.0]]) # set up as a 1x1 array
Theta_ddot = np.array([[2.0]])
J11 = -l * math.sin(Theta)
J21 = l * math.cos(Theta)
J = np.array([[J11],[J21]]) # set up as a 2x1 array
# Hand Velocity
Hdot = np.dot(J,Theta_dot) # np.dot does matrix multiplication
```

Hand Velocity — 1 DOF



1. $l = 0.46m; \theta(rads) = sin(2\pi t_i/4); \dot{\theta}(rads/s) = cos(2\pi t_i/4); t_i = linspace(0, 1, 200)$; *known exact solution from angle to velocity

2. calculate angular velocity from recorded elbow angle (e.g., numerical differentiation), then calculate hand velocity

Hand Acceleration — 1 DOF

Goal: relate joint acceleration ($\ddot{\theta}$) to hand acceleration ($\ddot{H}$)

Hand Acceleration — 1 DOF

Goal: relate joint acceleration ($\ddot{\theta}$) to hand acceleration ($\ddot{H}$)

Hand velocity can be expressed as

$$\dot{H} = J(\theta) \cdot \dot{\theta}$$

Hand Acceleration — 1 DOF

Goal: relate joint acceleration ($\ddot{\theta}$) to hand acceleration ($\ddot{H}$)

Hand velocity can be expressed as

$$\dot{H} = J(\theta) \cdot \dot{\theta}$$

The time derivative of the above equation gives hand acceleration:

$$\ddot{H} = \frac{d\dot{H}}{dt} = \frac{d}{dt}(J(\theta) \cdot \dot{\theta})$$

Hand Acceleration — 1 DOF

Goal: relate joint acceleration ($\ddot{\theta}$) to hand acceleration ($\ddot{H}$)

Hand velocity can be expressed as

$$\dot{H} = J(\theta) \cdot \dot{\theta}$$

The time derivative of the above equation gives hand acceleration:

$$\ddot{H} = \frac{d\dot{H}}{dt} = \frac{d}{dt}(J(\theta) \cdot \dot{\theta})$$

We can solve $\frac{d}{dt}(J(\theta) \cdot \dot{\theta})$ using the Product Rule!

$e.g.$, $\frac{d}{dx}(u \cdot v) = \frac{du}{dx} \cdot v + u \cdot \frac{dv}{dx}$

Hand Acceleration — 1 DOF

Goal: relate joint acceleration ($\ddot{\theta}$) to hand acceleration ($\ddot{H}$)

Hand velocity can be expressed as

$$\dot{H} = J(\theta) \cdot \dot{\theta}$$

The time derivative of the above equation gives hand acceleration:

$$\ddot{H} = \frac{d\dot{H}}{dt} = \frac{d}{dt}(J(\theta) \cdot \dot{\theta})$$

We can solve $\frac{d}{dt}(J(\theta) \cdot \dot{\theta})$ using the Product Rule!
$e.g., \frac{d}{dx}(u \cdot v) = \frac{du}{dx} \cdot v + u \cdot \frac{dv}{dx}$

$$\ddot{H} = J(\dot{\theta}) \cdot \dot{\theta} + J(\theta) \cdot \ddot{\theta}$$

Hand Acceleration — 1 DOF

Goal: relate joint acceleration ($\ddot{\theta}$) to hand acceleration ($\ddot{H}$)

Hand velocity can be expressed as

$$\dot{H} = J(\theta) \cdot \dot{\theta}$$

The time derivative of the above equation gives hand acceleration:

$$\ddot{H} = \frac{d\dot{H}}{dt} = \frac{d}{dt}(J(\theta) \cdot \dot{\theta})$$

We can solve $\frac{d}{dt}(J(\theta) \cdot \dot{\theta})$ using the Product Rule!

$e.g.$, $\frac{d}{dx}(u \cdot v) = \frac{du}{dx} \cdot v + u \cdot \frac{dv}{dx}$

$$\ddot{H} = \dot{J(\theta)} \cdot \dot{\theta} + J(\theta) \cdot \ddot{\theta}$$

The Jacobian Time Derivative ($\dot{J(\theta)}$)

The Jacobian is a function of angle, and angle is a function of time (i.e., $J(\theta(t))$; note the t is usually dropped in the notation)

For equations of this form (e.g., f(g(x))), we can again apply the chain rule to calculate the time derivative.

$$\dot{J(\theta)} = \frac{dJ(\theta)}{dt} = \frac{dJ(\theta)}{d\theta} \cdot \frac{d\theta}{dt}$$

$$\dot{J(\theta)} = \begin{bmatrix} \frac{d(-l \cdot sin(\theta))}{d\theta} \cdot \frac{d\theta}{dt} \\ \frac{d(l \cdot cos(\theta))}{d\theta} \cdot \frac{d\theta}{dt} \end{bmatrix}$$

$$\dot{J(\theta)} = \begin{bmatrix} -l \cdot cos(\theta) \cdot \dot{\theta} \\ -l \cdot sin(\theta) \cdot \dot{\theta} \end{bmatrix}$$

$\dot{J}(\theta)$ with Sympy — Python Code

```
#### 1DOF Jacobian DOT ####
from sympy import *
# define these variables as symbolic (not numeric)
a, l, t = symbols('a l t')
# forward kinematics for Hx and Hy
J11 = -l*sin(a(t))
J21 = l*cos(a(t))
# use sympy diff() to get derivatives for Jacobian DOT matrix
J11dot = diff(J11,t)
J21dot = diff(J21,t)
print(J11dot)
print(J21dot)
```

Hand Acceleration — 1 DOF

$$\ddot{H} = \dot{J(\theta)} \cdot \dot{\theta} + J(\theta) \cdot \ddot{\theta}$$

In expanded form

$$\begin{bmatrix} \ddot{H}x \\ \ddot{H}y \end{bmatrix} = \begin{bmatrix} -l \cdot cos(\theta) \cdot \dot{\theta} \\ -l \cdot sin(\theta) \cdot \dot{\theta} \end{bmatrix} \begin{bmatrix} \dot{\theta} \end{bmatrix} + \begin{bmatrix} -l \cdot sin(\theta) \\ l \cdot cos(\theta) \end{bmatrix} \begin{bmatrix} \ddot{\theta} \end{bmatrix}$$

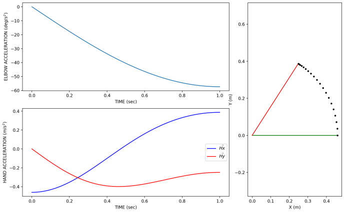
Example:

$$\begin{bmatrix} -9.95 \\ -5.84 \end{bmatrix} = \begin{bmatrix} -0.46 \cdot cos(\frac{35\pi}{180}) \cdot 5 \\ -0.46 \cdot sin(\frac{35\pi}{180}) \cdot 5 \end{bmatrix} \begin{bmatrix} 5 \end{bmatrix} + \begin{bmatrix} -0.46 \cdot sin(\frac{35\pi}{180}) \\ 0.46 \cdot cos(\frac{35\pi}{180}) \end{bmatrix} \begin{bmatrix} 2 \end{bmatrix}$$

Hand Acceleration — Python Code

```
# Hand Velocity and Hand Acceleration - matrix multiplication in python
import numpy as np
import math
l = 0.46
Theta = 35 * math.pi / 180.
Theta_dot = np.array([[5.0]]) # set up as a 1x1 array
Theta_ddot = np.array([[2.0]])
J11 = -l * math.sin(Theta)
J21 = l * math.cos(Theta)
J = np.array([[J11],[J21]]) # set up as a 2x1 array
# Hand Velocity
Hdot = np.dot(J,Theta_dot) # np.dot does matrix multiplication
Jdot11 = -l * math.cos(Theta) * Theta_dot[0,0] # [0,0] extract element from 1x1 array
Jdot21 = -l * math.sin(Theta) * Theta_dot[0,0]
Jdot = np.array([[Jdot11],[Jdot21]])
# Hand Acceleration
Hddot = np.dot(Jdot, Theta_dot) + np.dot(J,Theta_ddot)
```

Hand Acceleration — 1 DOF



1. $l = 0.46m; \theta(rads) = sin(2\pi t_i/4); \dot{\theta}(rads/s) = cos(2\pi t_i/4); \ddot{\theta}(rads/s^2) = -sin(2\pi t_i/4); t_i = linspace(0, 1, 200);$
   *known exact solution from angle to velocity to acceleration

Forward Kinematics Summary — 1 DOF

Position:

$$H_x = l \cdot cos(\theta)$$

$$H_y = l \cdot sin(\theta)$$

Velocity:

$$\dot{H} = J(\theta) \cdot \dot{\theta}$$

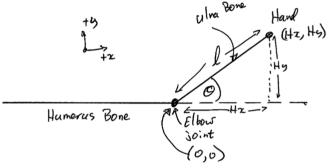
Acceleration:

$$\ddot{H} = \dot{J(\theta)} \cdot \dot{\theta} + J(\theta) \cdot \ddot{\theta}$$

Inverse Kinematics — 1DOF

Joint Angle — 1 DOF

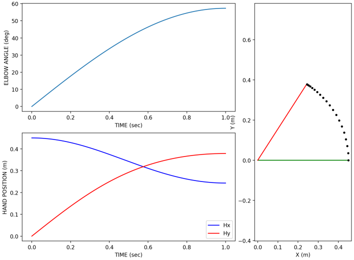
Go from extrinsic variables (hand space) to intrinsic variables (joint space)



Schematic of a simple kinematic model of the elbow joint

$$\theta = arctan\left(\frac{H_y}{H_x}\right)\frac{180}{\pi} = arctan\left(\frac{0.26}{0.38}\right)\cdot\frac{180}{\pi} = 35^{\circ}$$

Inverse Kinematics — 1 DOF



Inverse kinematics: record hand position and calculate elbow angle

Angular Velocity — 1 DOF

$$\dot{H} = J(\theta) \cdot \dot{\theta}$$

$$\dot{\theta} = J(\theta)^{-1} \cdot \dot{H}$$

*can find $J(\theta)^{-1}$ using a pseudo-inverse
(use the np.linalg.pinv() function in python)

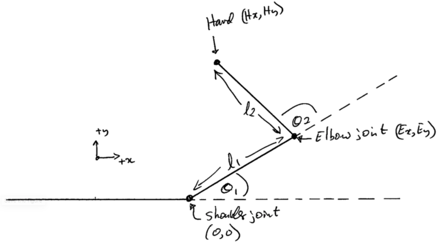
Angular Acceleration — 1 DOF

$$\ddot{H} = \dot{J(\theta)} \cdot \dot{\theta} + J(\theta) \cdot \ddot{\theta}$$

$$\ddot{\theta} = J(\theta)^{-1}(\ddot{H} - \dot{J(\theta)} \cdot \dot{\theta})$$

Forward Kinematics — 2DOF

Shoulder, Elbow and Hand



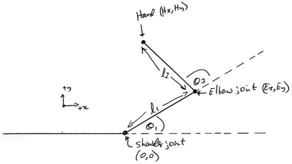
Schematic of a simple kinematic model of a two-joint arm

θ_1: shoulder angle; θ_2: elbow angle

$l_1(0.34)$: length of upper arm; $l_2(0.46)$: length of lower arm

$S_x = 0, S_y = 0$: shoulder coordinates; E_x, E_y: elbow coordinates; H_x, H_y: hand coordinates

Hand Position — 2 DOF



Schematic of a simple kinematic model of a two-joint arm

$$E_x = l_1 cos(\theta_1)$$

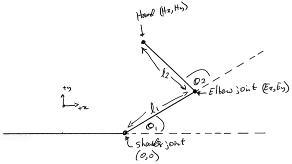
$$E_y = l_1 sin(\theta_1)$$

$$H_x = E_x + l_2 cos(\theta_1 + \theta_2)$$

$$H_y = E_y + l_2 sin(\theta_1 + \theta_2)$$

$l_1(0.34)$; $l_2(0.46)$; *elbow angle relative to upper arm

Hand Position — 2 DOF

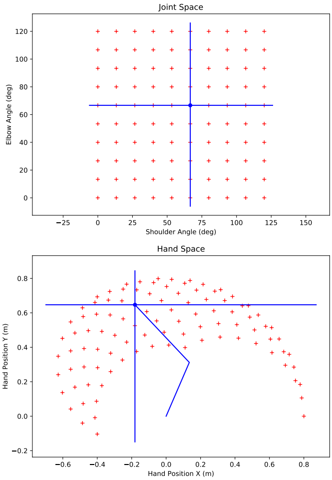


Schematic of a simple kinematic model of a two-joint arm

Alternatively,

$$H_x = l_1 cos(\theta_1) + l_2 cos(\theta_1 + \theta_2)$$
$$H_y = l_1 sin(\theta_1) + l_2 sin(\theta_1 + \theta_2)$$

Forward Kinematics — 2 DOF



Hand Velocity — 2 DOF

$$\dot{H} = J(\theta) \cdot \dot{\theta}$$

where

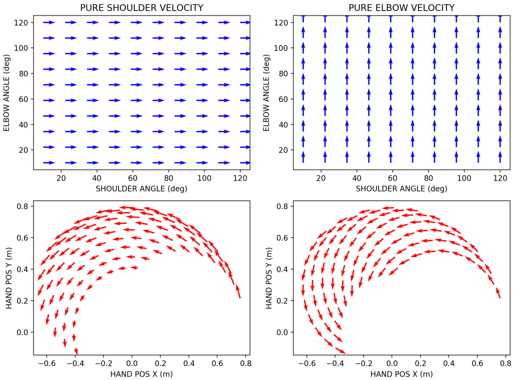
$$J(\theta) = \frac{dH}{d\theta} = \begin{bmatrix} \frac{\partial H_x}{\partial \theta_1} & \frac{\partial H_x}{\partial \theta_2} \\ \frac{\partial H_y}{\partial \theta_1} & \frac{\partial H_y}{\partial \theta_2} \end{bmatrix}$$

Thus,

$$\begin{bmatrix} \dot{H}x \\ \dot{H}y \end{bmatrix} = \begin{bmatrix} -l_1 sin(\theta_1) - l_2 sin(\theta_1 + \theta_2) & -l_2 sin(\theta_1 + \theta_2) \\ l_1 cos(\theta_1) + l_2 cos(\theta_1 + \theta_2) & l_2 cos(\theta_1 + \theta_2) \end{bmatrix} \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \end{bmatrix}$$

1. take the partial derivatives to find $J(\theta)$ by hand and sympy
2. reminder: $\frac{dcos(\theta_1+\theta_2)}{d\theta_1} = -sin(\theta_1 + \theta_2); \frac{dsin(\theta_1+\theta_2)}{d\theta_2} = cos(\theta_1 + \theta_2)$
3. $J(\theta)$ is a 2x2 matrix

Hand Velocity — 2 DOF



Each arrow represents a unit vector of velocity in joint space (top row) or hand space (bottom row)

Hand Acceleration — 2 DOF

$$\ddot{H} = \dot{J(\theta)} \cdot \dot{\theta} + J(\theta) \cdot \ddot{\theta}$$

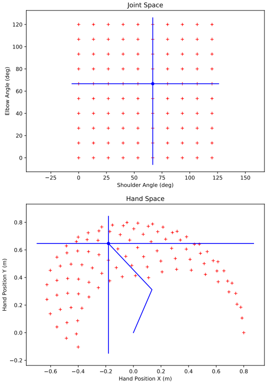
in expanded form:

$$\begin{bmatrix} \ddot{H}x \\ \ddot{H}y \end{bmatrix} = \begin{bmatrix} -l_1 cos(\theta_1)\dot{\theta}_1 - l_2 cos(\theta_1 + \theta_2)(\dot{\theta}_1 + \dot{\theta}_2) & -l_2 cos(\theta_1 + \theta_2)(\dot{\theta}_1 + \dot{\theta}_2) \\ -l_1 sin(\theta_1)\dot{\theta}_1 - l_2 sin(\theta_1 + \theta_2)(\dot{\theta}_1 + \dot{\theta}_2) & -l_2 sin(\theta_1 + \theta_2)(\dot{\theta}_1 + \dot{\theta}_2) \end{bmatrix} \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \end{bmatrix} +$$
$$\begin{bmatrix} -l_1 sin(\theta_1) - l_2 sin(\theta_1 + \theta_2) & -l_2 sin(\theta_1 + \theta_2) \\ l_1 cos(\theta_1) + l_2 cos(\theta_1 + \theta_2) & l_2 cos(\theta_1 + \theta_2) \end{bmatrix} \begin{bmatrix} \ddot{\theta}_1 \\ \ddot{\theta}_2 \end{bmatrix}$$

1. take the time derivatives of $J(\theta)$ to find $\dot{J(\theta)}$ by hand and sympy
2. note: $\frac{dsin(\theta_1(t)+\theta_2(t))}{dt} = cos(\theta_1 + \theta_2)\dot{\theta}_1 + cos(\theta_1 + \theta_2)\dot{\theta}_2 = cos(\theta_1 + \theta_2)(\dot{\theta}_1 + \dot{\theta}_2)$
   . e.g., ab + ac = a(b+c)
3. $\dot{J(\theta)}$is a 2x2 matrix

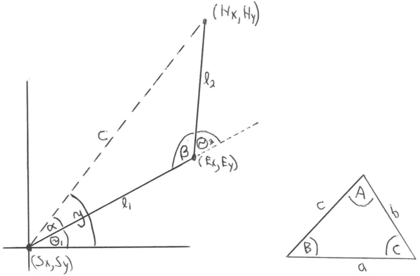
Inverse Kinematics — 2DOF

Inverse Kinematics — 2 DOF



Here we want to go from hand space to joint space

Joint Angles — 2 DOF



cosine law: $c^2 = a^2 + b^2 - 2ab \cdot cos(C)$

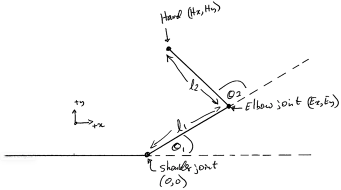
1. Use trig to calculate joint angles from hand coordinates
2. Knowns: H_x, H_y, l_1, l_2; Unknowns:$\theta_1, \theta_2, c, \alpha, \beta, \gamma$
3. Test calculations with hand in each quadrant
   . Tips: math.atan2(y,x) when calculating γ and $\theta_2 \geq 0$

Angular Vel. and Accel. — 2 DOF

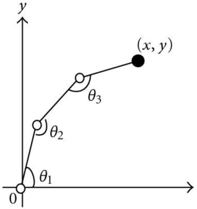
$$\dot{\theta} = J(\theta)^{-1} \cdot \dot{H}$$

$$\ddot{\theta} = J(\theta)^{-1}(\ddot{H} - \dot{J(\theta)} \cdot \dot{\theta})$$

Redundancy



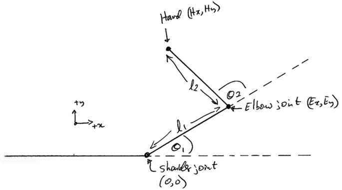
Schematic of a simple kinematic model of a two-joint arm



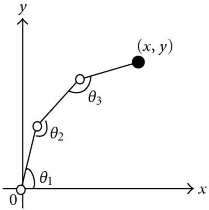
Redundancy

Degrees-of-Freedom (DOF) $>$ Task Space Dimensions

Redundancy



Schematic of a simple kinematic model of a two-joint arm

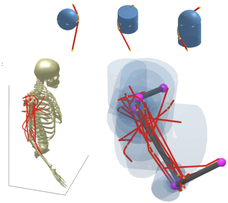


Redundancy

Degrees-of-Freedom (DOF) $>$ Task Space Dimensions

- 2 DoF moving in 2D space = not redundant (1 solution)
- 3 DoF moving in 2D space = redundant (∞ solutions)
- If redundant, need to estimate joint angles directly!

Redundancy



Humans are highly redundant (e.g., reaching in a 3D (x,y,z) space)

1. Shoulder (3DOF)
2. Elbow (1DOF)
3. Wrist (2DOF)

Many joints also translate (up to 6DOF per joint!)

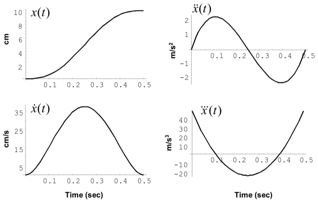
The Curse of Redundancy

The Curse:

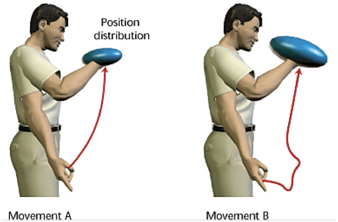
1. infinite ways to accomplish task goals
2. how does the brain decide which action to take???

Does the Brain Care about Kinematics?

minimum jerk trajectories



minimum end-point variance



More about these later in the course

Summary

1. Forward Kinematics
   - going from joint space to hand space
   - position, velocity, acceleration, etc,
   - Jacobian: $J(\theta)$, and its time derivative: $\dot{J}(\theta)$
2. Inverse Kinematics
3. going from hand space to joint space
4. Redundancy
5. Does the Brain Care about Kinematics???

Questions???

Next Class

Dynamics

- one-link arm (pendulum)
- two-link arm (double pendulum)
- Euler-Lagrange Equations

Assignment 4

See Handout

Acknowledgements

Paul Gribble

Dinant Kistemaker