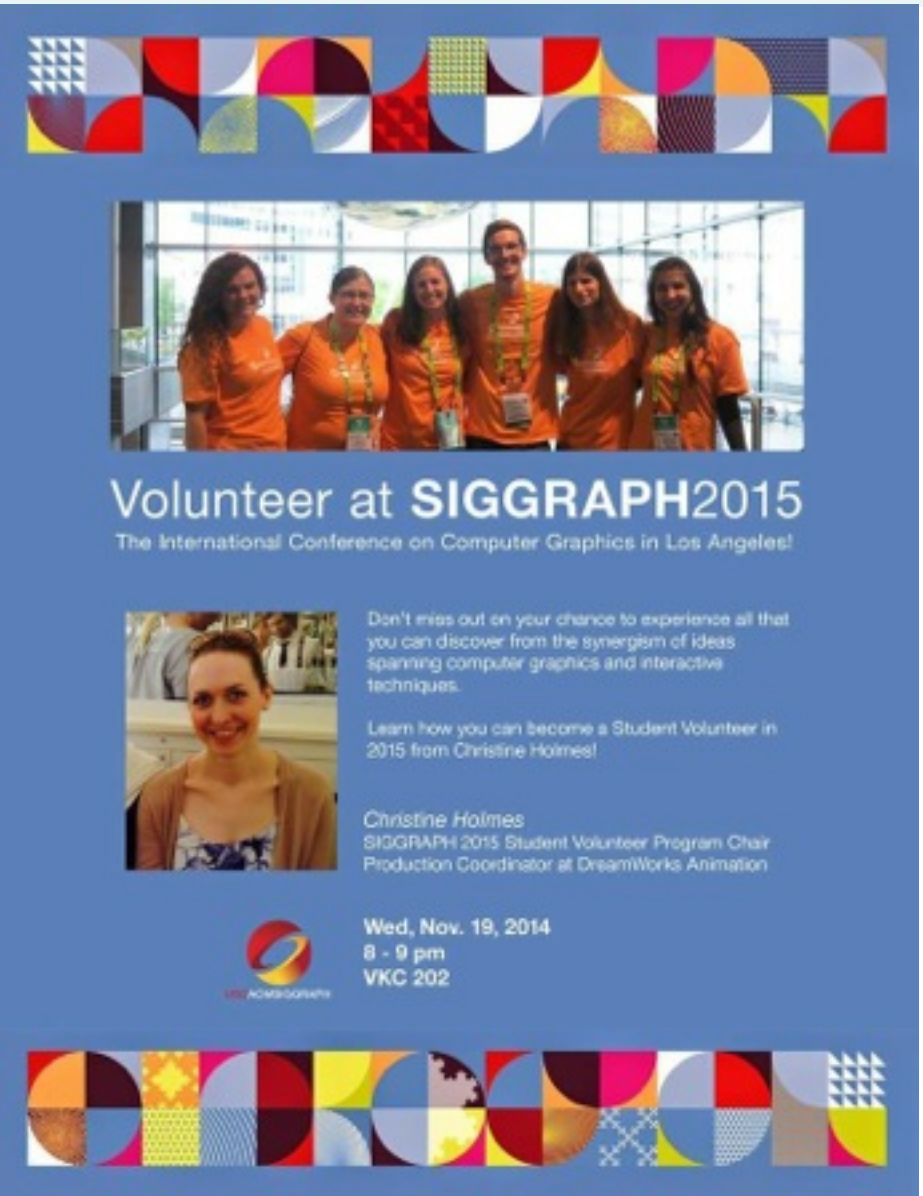
12.2 Quaternions and Rotations

Hao Li

http://cs420.hao-li.com

Volunteer at SIGGRAPH 2015



Rotations

- Very important in computer animation and robotics
- Joint angles, rigid body orientations, camera parameters
- 2D or 3D

Rotations in Three Dimensions

- Orthogonal matrices:

$RR^T = R^TR = I$
$\det(R) = 1$

$$R = \begin{bmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{bmatrix}$$

Representing Rotations in 3D

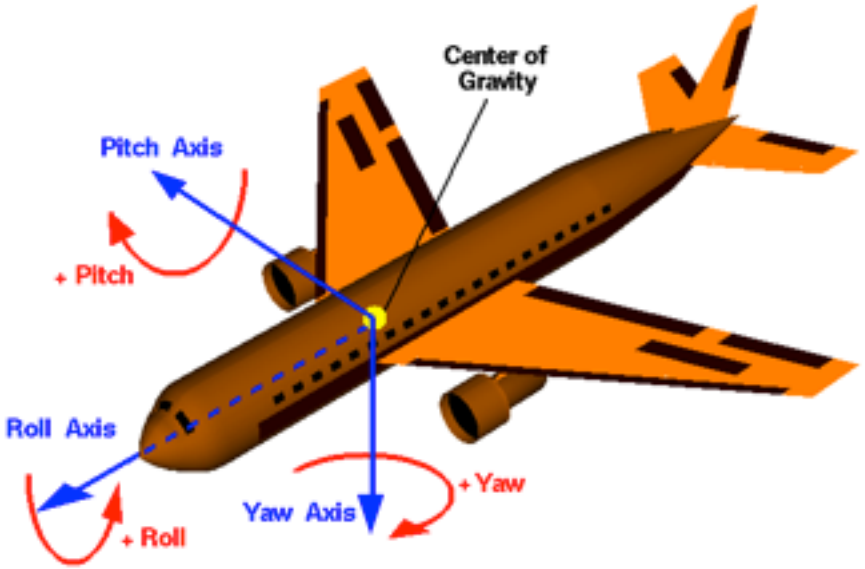
- Rotations in 3D have essentially three parameters

- Axis + angle (2 DOFs + 1DOFs)
  - How to represent the axis?
    Longitude / lattitude have singularities

- 3x3 matrix
  - 9 entries (redundant)

Representing Rotations in 3D

- Euler angles
  - roll, pitch, yaw
  - no redundancy (good)
  - gimbal lock singularities



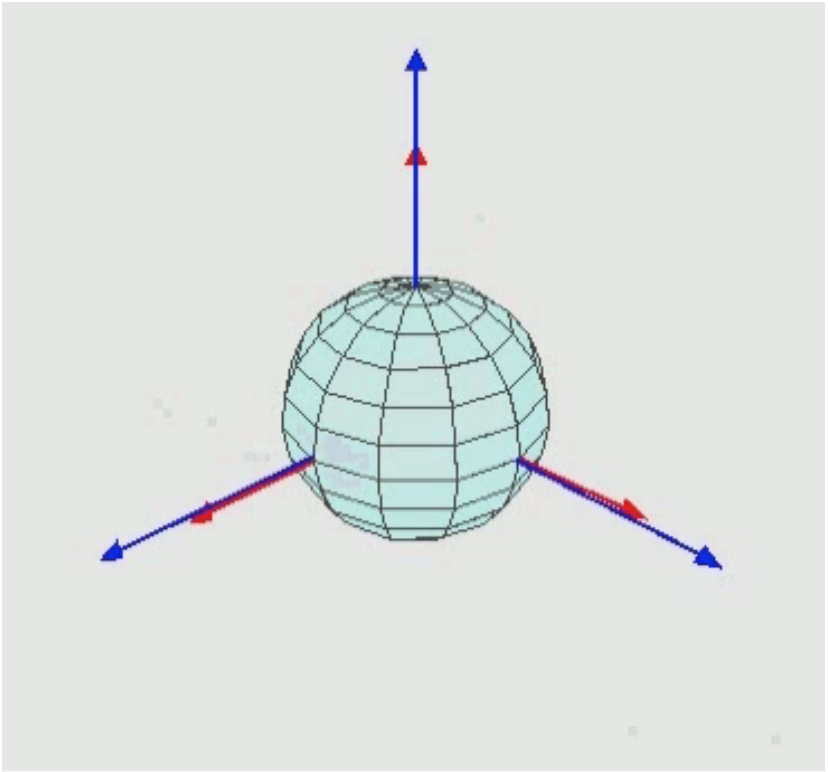
- Quaternions
  - generally considered the “best” representation
  - redundant (4 values), but only by one DOF (not severe)
  - stable interpolations of rotations possible

Euler Angles

1. Yaw
   rotate around y-axis

2. Pitch
   rotate around (rotated) x-axis

3. Roll
   rotate around (rotated) y-axis

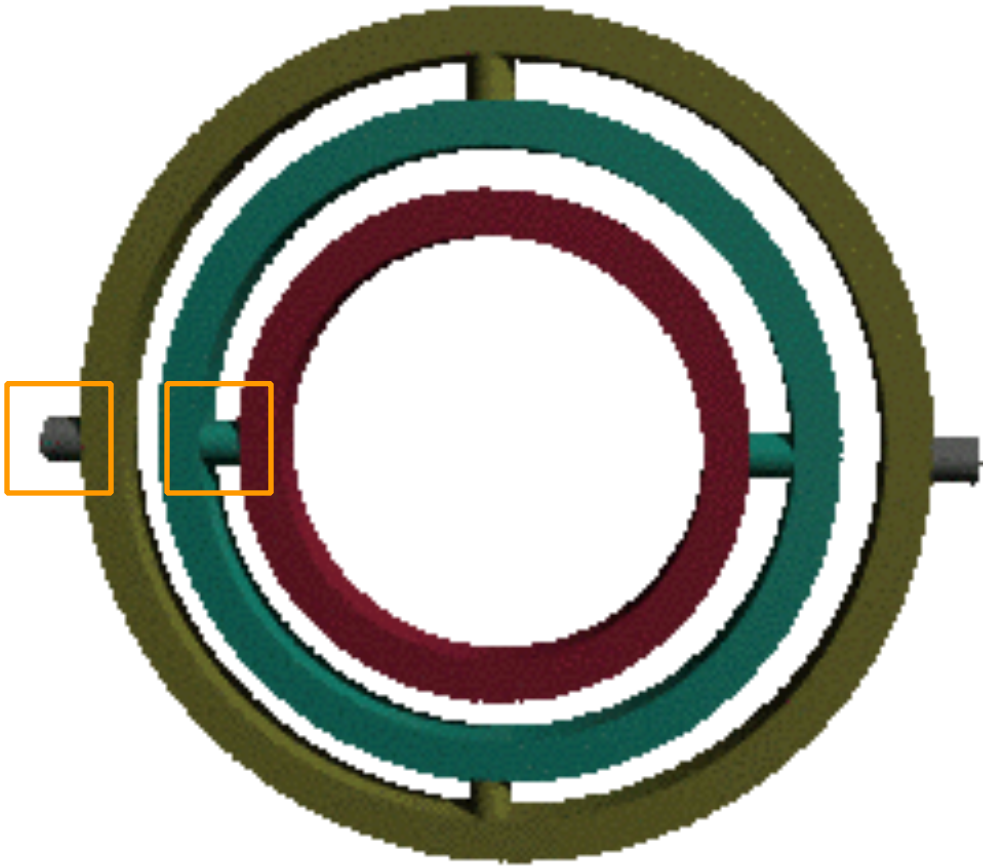


Gimbal Lock

When all three gimbals are lined up (in the same plane), the system can only move in two dimensions from this configuration, not three, and is in gimbal lock.

Gimbal Lock

When all three gimbals are lined up (in the same plane), the system can only move in two dimensions from this configuration, not three, and is in gimbal lock.



Outline

- Rotations
- Quaternions
- Motion Capture

Quaternions

- Generalization of complex numbers

- Three imaginary numbers: $\boldsymbol{i}$, $\boldsymbol{j}$, $\boldsymbol{k}$

  $\boldsymbol{i}^2 = -1$, $\boldsymbol{j}^2 = -1$, $\boldsymbol{k}^2 = -1$,
  $\boldsymbol{ij} = \boldsymbol{k}$, $\boldsymbol{jk} = \boldsymbol{i}$, $\boldsymbol{ki} = \boldsymbol{j}$, $\boldsymbol{ji} = -\boldsymbol{k}$, $\boldsymbol{kj} = -\boldsymbol{i}$, $\boldsymbol{ik} = -\boldsymbol{j}$

- $q = s + x\,\boldsymbol{i} + y\,\boldsymbol{j} + z\,\boldsymbol{k}$, s,x,y,z are scalars

Quaternions

- Invented by Hamilton in 1843 in Dublin, Ireland

- Here as he walked by on the 16th of October 1843 Sir William Rowan Hamilton in a flash of genius discovered the fundamental formula for quaternion multiplication

  $i^2 = j^2 = k^2 = i\,j\,k = -1$

  & cut it on a stone of this bridge.

Quaternions

- Quaternions are **not** commutative!

  $q_1 q_2 \neq q_2 q_1$

- However, the following hold:

  $(q_1 q_2) q_3 = q_1 (q_2 q_3)$
  $(q_1 + q_2) q_3 = q_1 q_3 + q_2 q_3$
  $q_1 (q_2 + q_3) = q_1 q_2 + q_1 q_3$
  $\alpha (q_1 + q_2) = \alpha q_1 + \alpha q_2$ (α is scalar)
  $(\alpha q_1) q_2 = \alpha (q_1 q_2) = q_1 (\alpha q_2)$ (α is scalar)

- I.e. all usual manipulations are valid, except cannot reverse multiplication order.

Quaternions

- Exercise: multiply two quaternions

  $(2 - \boldsymbol{i} + \boldsymbol{j} + 3\boldsymbol{k})\ (-1 + \boldsymbol{i} + 4\boldsymbol{j} - 2\boldsymbol{k}) = \ldots$

Quaternion Properties

- $q = s + x\,\boldsymbol{i} + y\,\boldsymbol{j} + z\,\boldsymbol{k}$
- Norm: $|q|^2 = s^2 + x^2 + y^2 + z^2$
- Conjugate quaternion: $q^* = s - x\,\boldsymbol{i} - y\,\boldsymbol{j} - z\,\boldsymbol{k}$
- Inverse quaternion: $q^{-1} = q^* / |q|^2$
- Unit quaternion: $|q| = 1$
- Inverse of unit quaternion: $q^{-1} = q^*$

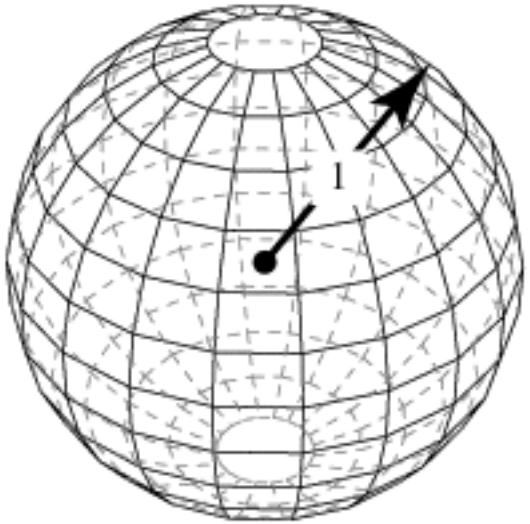
Quaternions and Rotations

- Rotations are represented by ***unit*** quaternions

- $q = s + x\,\boldsymbol{i} + y\,\boldsymbol{j} + z\,\boldsymbol{k}$

  $s^2 + x^2 + y^2 + z^2 = 1$

- Unit quaternion sphere (unit sphere in 4D)



Rotations to Unit Quaternions

- Let (unit) rotation axis be $[u_x, u_y, u_z]$, and angle θ
- Corresponding quaternion is

  $q = \cos(\theta/2) + \sin(\theta/2)\, u_x \boldsymbol{i} + \sin(\theta/2)\, u_y \boldsymbol{j} + \sin(\theta/2)\, u_z \boldsymbol{k}$

- Composition of rotations q_1 and q_2 equals $q = q_2\, q_1$
- 3D rotations do not commute!

Unit Quaternions to Rotations

- Let v be a (3-dim) vector and let q be a unit quaternion

- Then, the corresponding rotation transforms vector v to q ***v*** q^{-1}

  (***v*** is a quaternion with scalar part equaling 0, and vector part equaling v)

- For q = a+b**i**+c**j**+d**k**

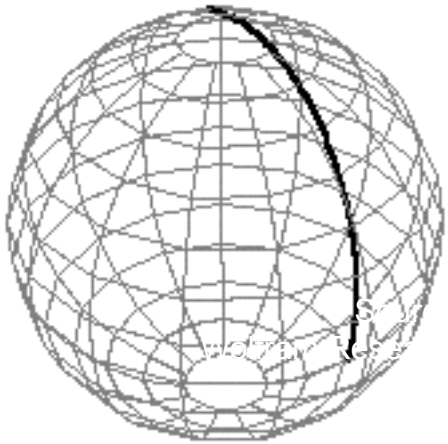
$$\begin{pmatrix} a^2+b^2-c^2-d^2 & 2bc-2ad & 2bd+2ac \\ 2bc+2ad & a^2-b^2+c^2-d^2 & 2cd-2ab \\ 2bd-2ac & 2cd+2ab & a^2-b^2-c^2+d^2 \end{pmatrix}$$

Quaternions

- Quaternions q and -q give the same rotation!

- Other than this, the relationship between rotations and quaternions is unique

Quaternion Interpolation



- Better results than Euler angles
- A quaternion is a point on the 4-D unit sphere
  - interpolating rotations requires a unit quaternion at each step -- another point on the 4-D sphere
  - move with constant angular velocity along the great circle between the two points
  - Spherical Linear intERPolation (SLERPing)
- Any rotation is given by 2 quaternions, so pick the shortest SLERP

SLERP

$$\mathrm{Slerp}(q_1, q_2, u) = \frac{\sin\big((1-u)\theta\big)}{\sin(\theta)} q_1 + \frac{\sin(u\theta)}{\sin(\theta)} q_2$$

$$\cos(\theta) = q_1 \cdot q_2 =$$

$$= s_1 s_2 + x_1 x_2 + y_1 y_2 + z_1 z_2$$

- u varies from 0 to 1
- $q_m = s_m + x_m \boldsymbol{i} + y_m \boldsymbol{j} + z_m \boldsymbol{k}$, for m = 1,2
- The above formula does not produce a unit quaternion and must be normalized; replace q by q / |q|

Quaternion Interpolation

- To interpolate more than two points:
  - solve a non-linear variational constrained optimization (numerically)

- Further information: Ken Shoemake in the SIGGRAPH '85 proceedings (Computer Graphics, V. 19, No. 3, P.245)

Outline

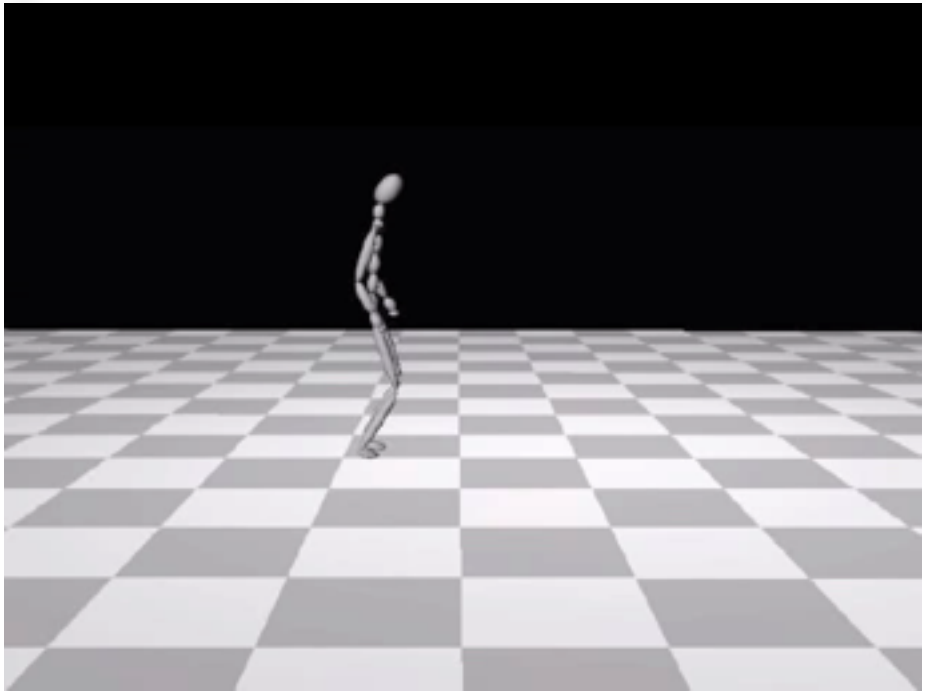
- Rotations
- Quaternions
- Motion Capture

What is Motion Capture?

- Motion capture is the process of tracking real-life motion in 3D and recording it for use in any number of applications.

Why Motion Capture?

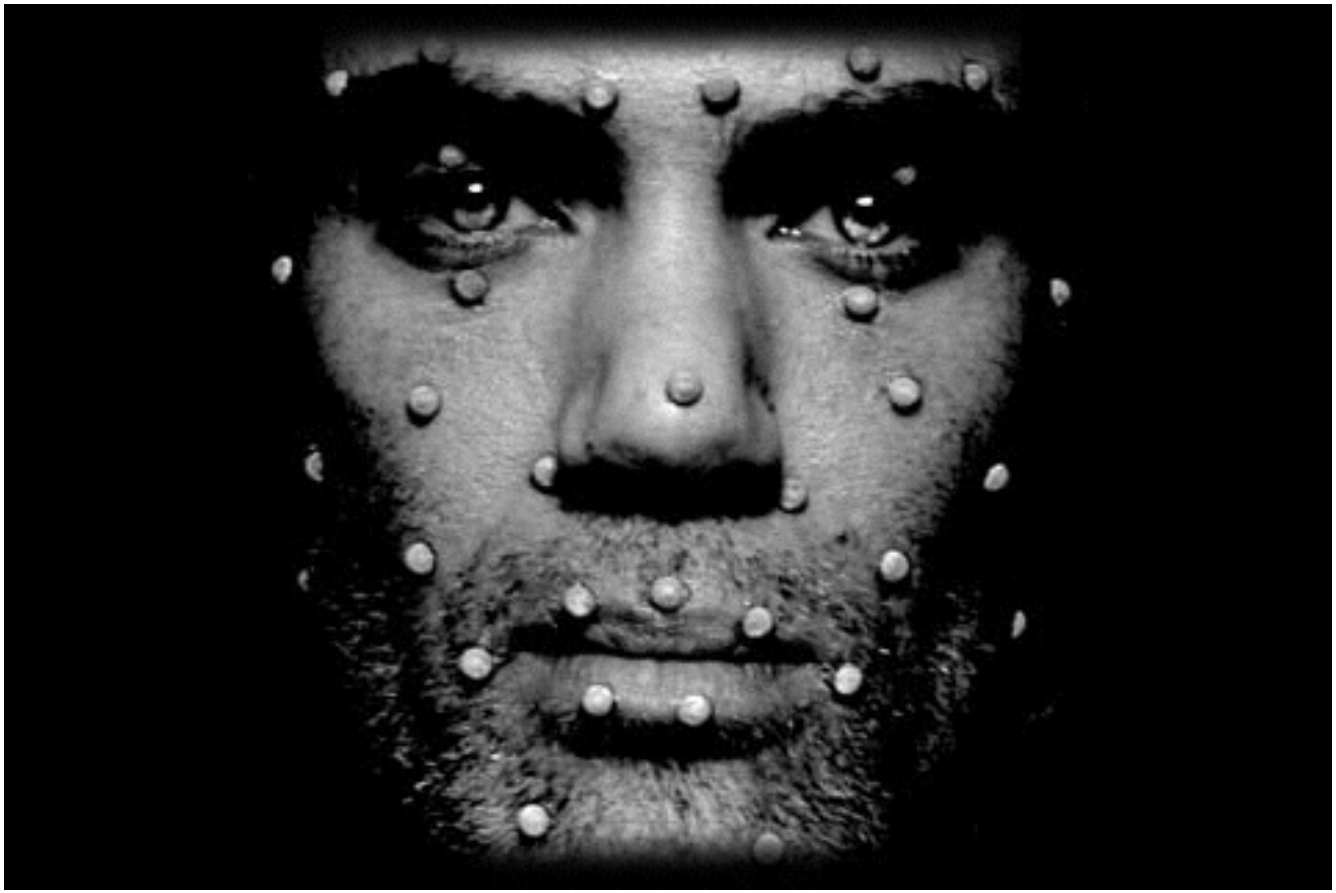
- Keyframes are generated by instruments measuring a human performer — they do not need to be set manually
- The details of human motion such as style, mood, and shifts of weight are reproduced with little effort



Mocap Technologies: Optical

- Multiple high-resolution, high-speed cameras
- Light bounced from camera off of reflective markers
- High quality data
- Markers placeable anywhere
- Lots of work to extract joint angles
- Occlusion
- Which marker is which? (correspondence problem)
- 120-240 Hz @ 1Megapixel

Facial Motion Capture



Mocap Technologies: Electromagnetic

- Sensors give both position and orientation
- No occlusion or correspondence problem
- Little post-processing
- Limited accuracy

Mocap Technologies: Exoskeleton

- Really Fast (~500Hz)
- No occlusion or correspondence problem
- Little error
- Movement restricted
- Fixed sensors

Motion Capture

- Why not?
  - Difficult for non-human characters
    - Can you move like a hamster / duck / eagle ?
    - Can you capture a hamster's motion?
  - Actors needed
    - Which is more economical:
      - Paying an animator to place keys
      - Hiring a Martial Arts Expert

When to use Motion Capture?

- Complicated character motion
  - Where “uncomplicated” ends and “complicated” begins is up to question
  - A walk cycle is often more easily done by hand
  - A Flying Monkey Kick might be worth the overhead of mocap
- Can an actor better express character personality than the animator?

Summary

- Rotations
- Quaternions
- Motion Capture