MAC420/5744: Introdução à Computação Gráfica

Marcel P. Jackowski
mjack@ime.usp.br

Aula #20: Princípios de animação

What is animation ?

- Modeling geometry = specifying shape
  - Use of meshes, hierarchies, curved surfaces, etc
- Animation = specifying shape as a function of time
  - Modeling done once per frame?
  - Needs to be smooth, concerted movement

Keyframes in hand-drawn animation

- End goal: a drawing per frame, with nice smooth motion
- "Straight ahead" is drawing frames in order (using a lightbox to see the previous frame or frames)
  - Hard to get a character to land
    - at a particular pose
    - at a particular time
- Instead, use key frames
  - draw important poses first, then fill in the in-betweens

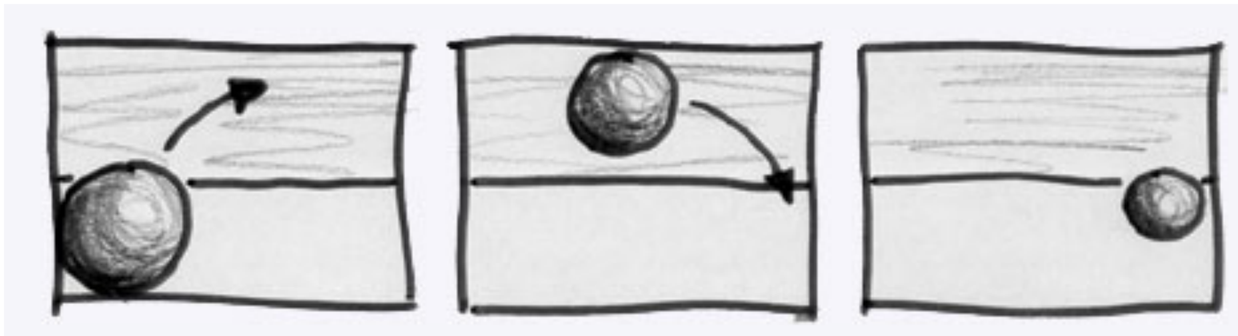
Keyframes in computer animation

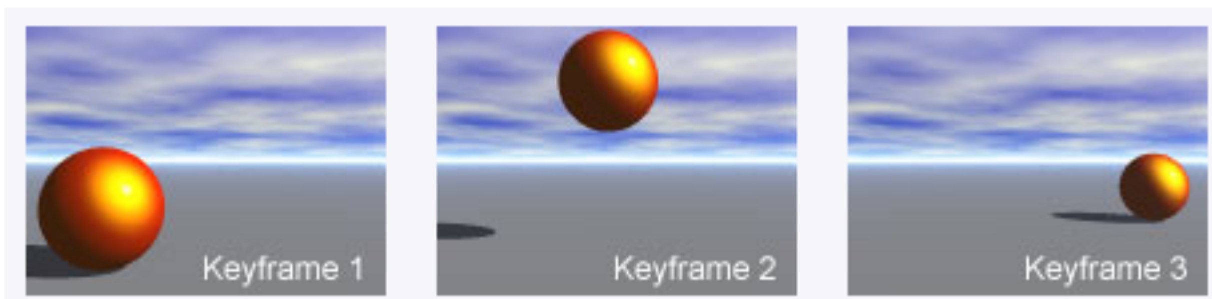
- Just as with hand-drawn animation, adjusting the model from scratch for every frame would be tedious and difficult
- Same solution: animator establishes the keyframes, software fills in the in-betweens
- Two key ideas of computer animation:
  - Create high-level controls for adjusting geometry
  - Interpolate these controls over time between keyframes

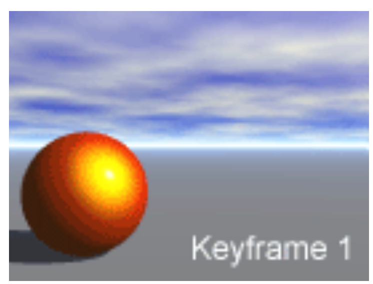
The most basic animation control

- Affine transformations position things in modeling
- Time-varying affine transformations move things around in animation
- A hierarchy of time-varying transformations is the main workhorse of animation
  - Basic framework within which all the more sophisticated techniques are built

Keyframe animation



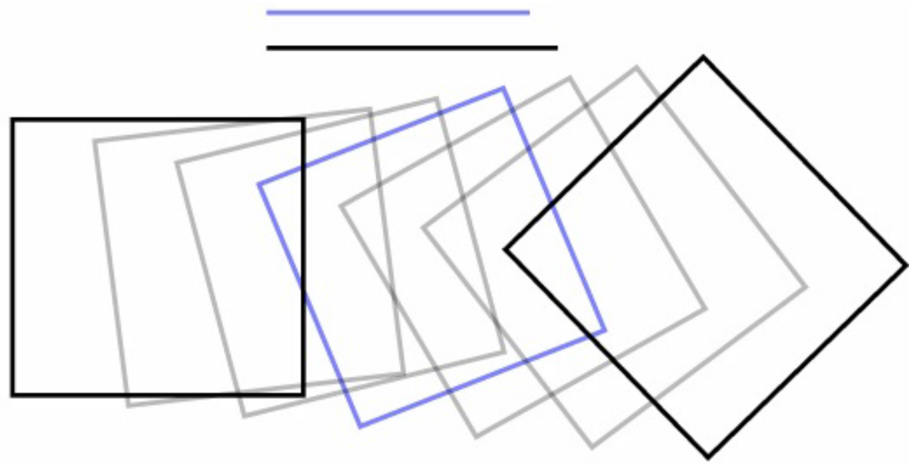




http://www.cadtutor.net/tutorials/bryce/keyframe-animation.php

Interpolating transformations

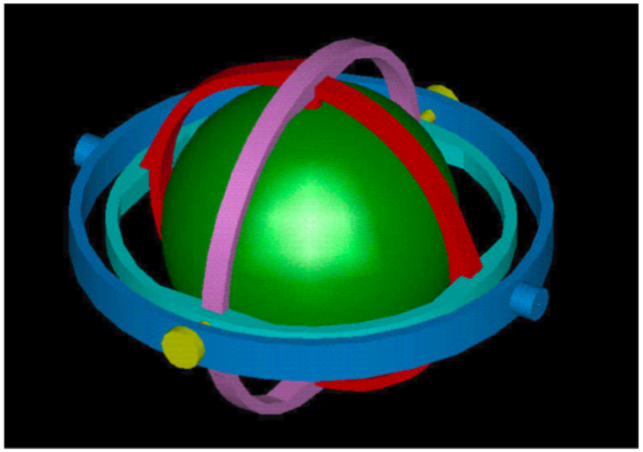
- Move a set of points by applying an affine transformation
- How to animate the transformation over time?
  - Interpolate the matrix entries from keyframe to keyframe?
  - This is fine for translations but bad for rotations



Parameterizing rotations

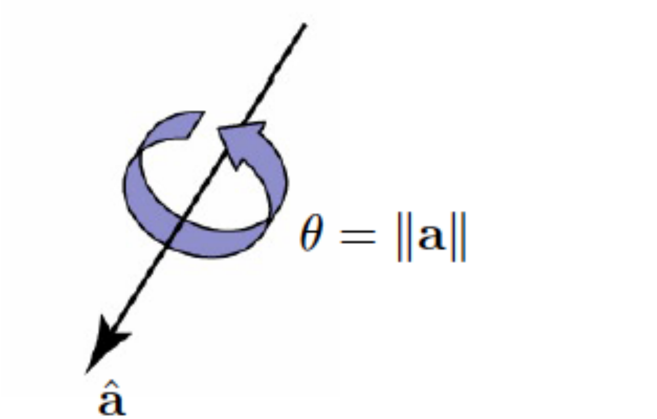
- Euler Angles
  - Rotate around x, then y, then z
  - Nice and simple

$$f(\alpha, \beta, \gamma) = R_{\mathbf{z}}(\gamma) R_{\mathbf{y}}(\beta) R_{\mathbf{x}}(\alpha)$$



- Axis/Angle
  - Specify axis to rotate around, then
  - Especify Angle by which to rotate

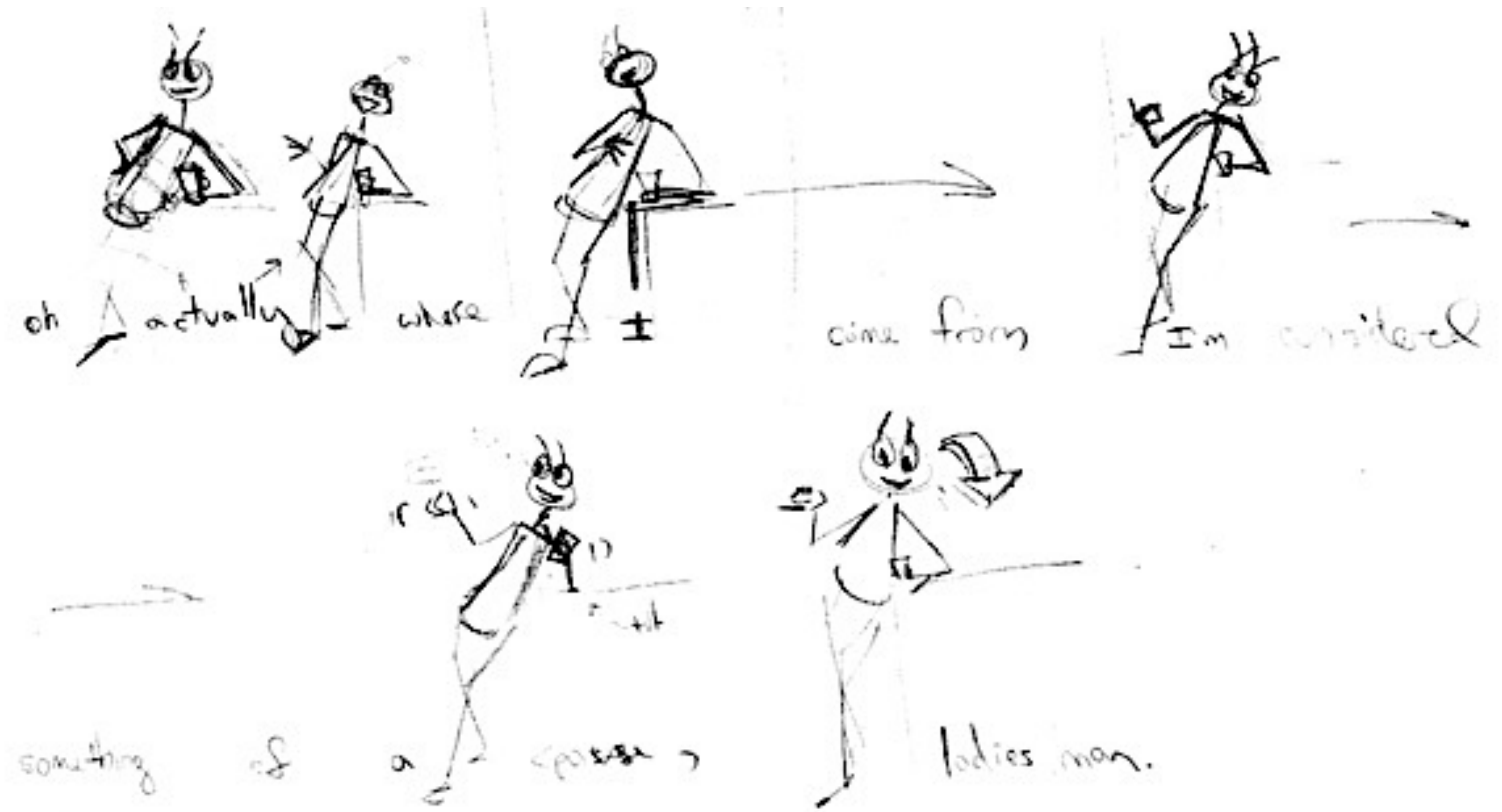
$$f(\mathbf{a}) = R_{\hat{\mathbf{a}}}(\|\mathbf{a}\|)$$



Approaches to animation

- Straight ahead
  - Draw/animate one frame at a time
  - Can lead to spontaneity, but is hard to get exactly what you want
- Pose-to-pose, top-down process
  - Plan shots using storyboards
  - Plan key poses first
  - Finally fill in the in-between frames

Pose-to-pose animation planning

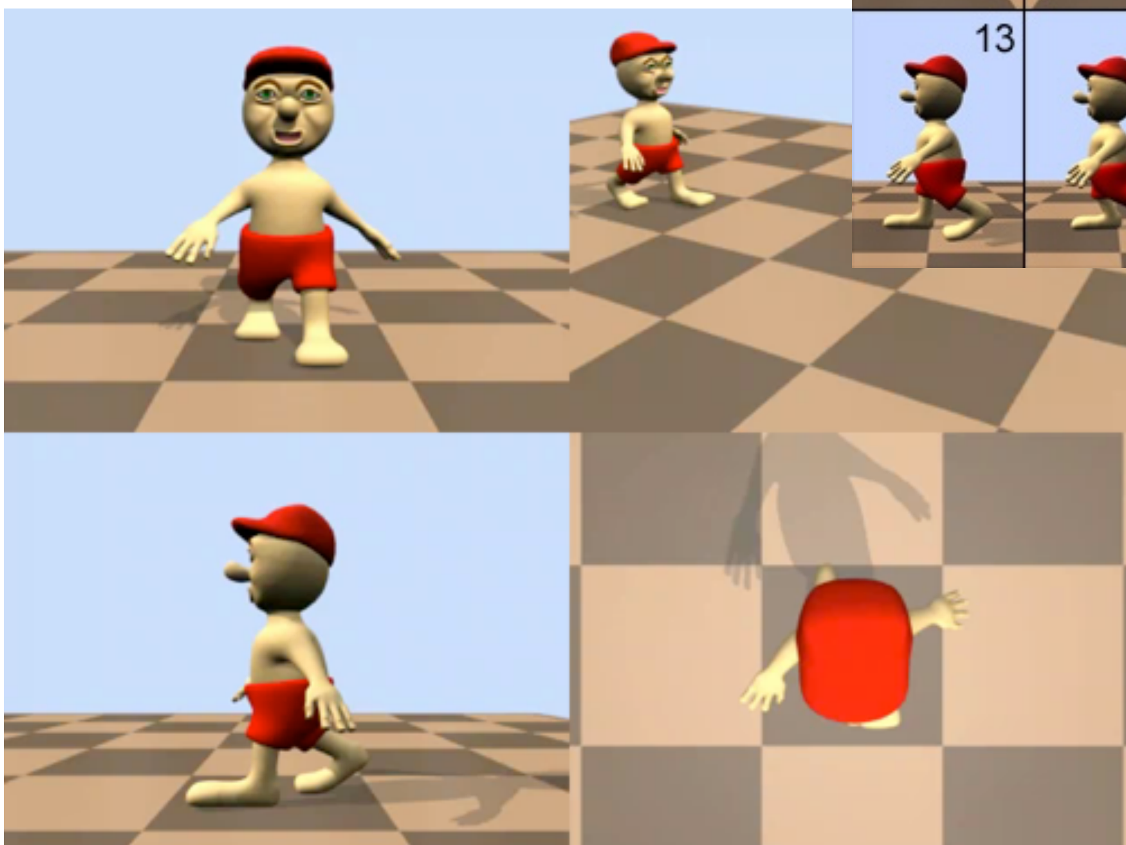


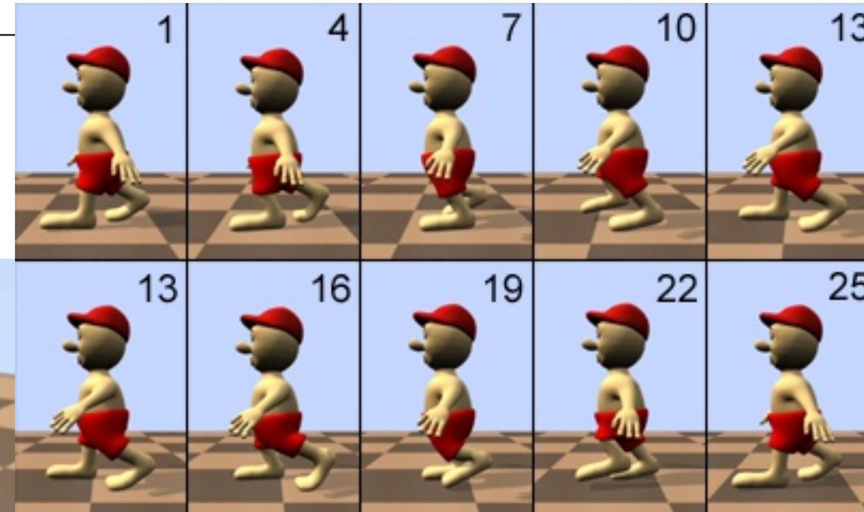
- First work out poses that are key to the story
- Next fill in animation in between

Keyframe animation

- Keyframing is the technique used for pose-to-pose animation
  - Head animator draws key poses—just enough to indicate what the motion is supposed to be
  - Assistants do “in-betweening” and draws the rest of the frames
  - Interpolation is the principal operation

Walk cycle

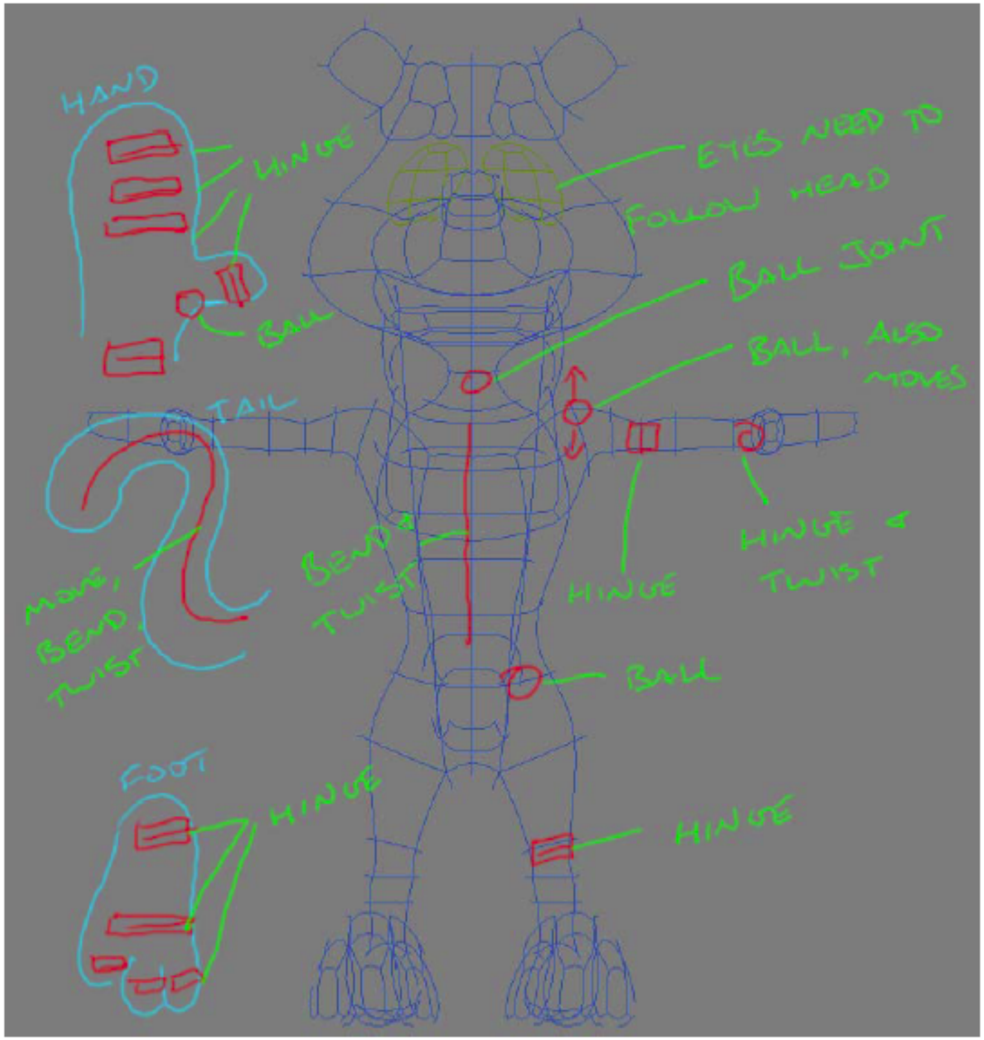




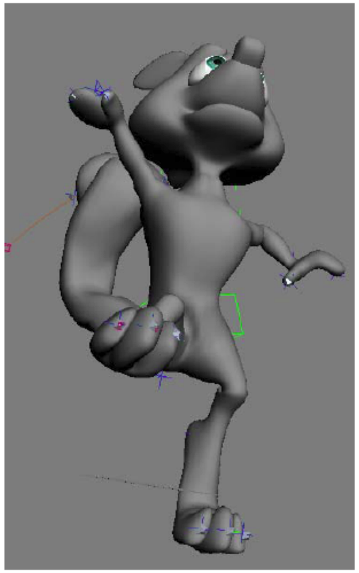
Controlling geometry conveniently

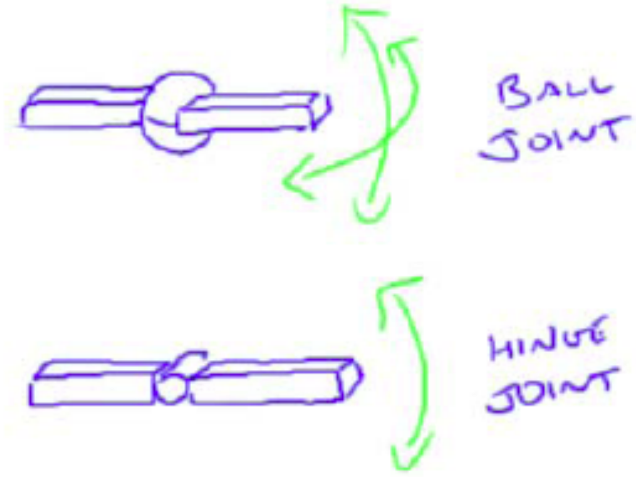
- Could animate by moving every control point at every keyframe
  - This would be labor intensive
  - It would also be hard to get smooth, consistent motion
- Better way: animate using smaller set of meaningful *degrees of freedom* (DOFs)
  - Modeling DOFs are inappropriate for animation
  - E.g. *"move one inch of left forearm"*
- Animation DOFs need to be higher level
  - E.g. *"bend the elbow"*

Characters with DOFs

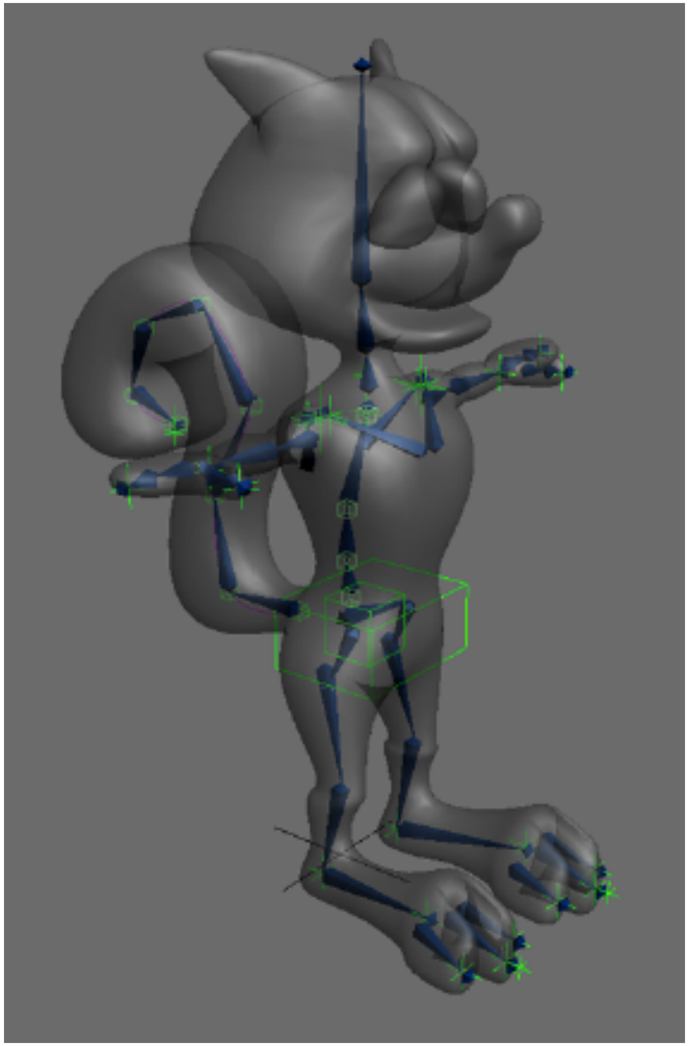


A visual description of the possible movements for the squirrel





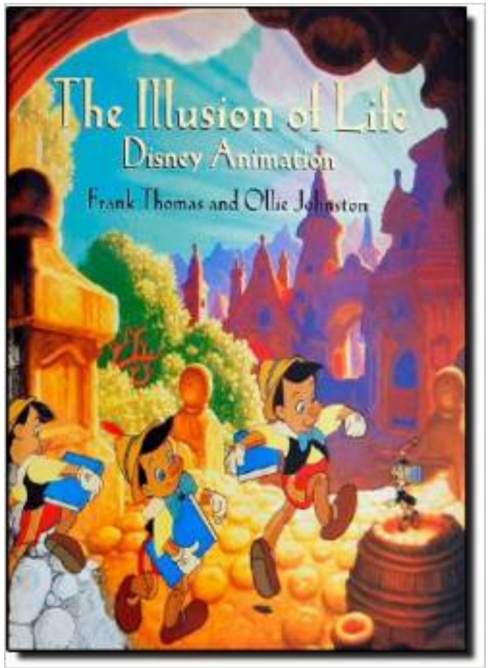
Rigged character



- Surface is deformed by a set of bones
- Bones are in turn controlled by a smaller set of controls
- The controls are useful, intuitive DOFs for an animator to use

The artistic process of animation

- What are animators trying to do?
  - Important to understand what tools they need
- Basic principles are universal across media
  - 2D hand-drawn animation
  - 2D and computer animation
  - 3D computer animation
- Widely cited set of principles laid out by Frank Thomas and Ollie Johnston in "The Illusion of Life" (1981)
  - 12 animation principles



1. Timing

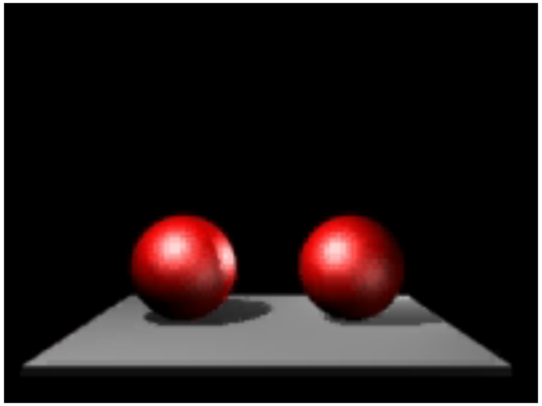
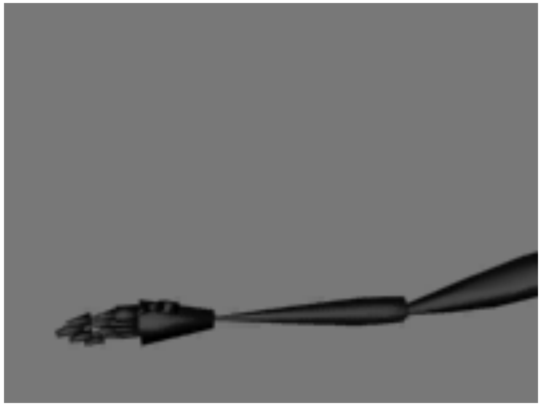
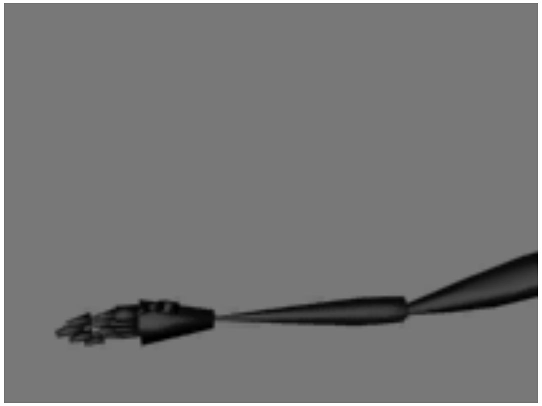
- Speed of an action is crucial to the impression it makes

60 fps 30 fps 5 fps

Michael Comet's examples: www.comet-cartoons.com

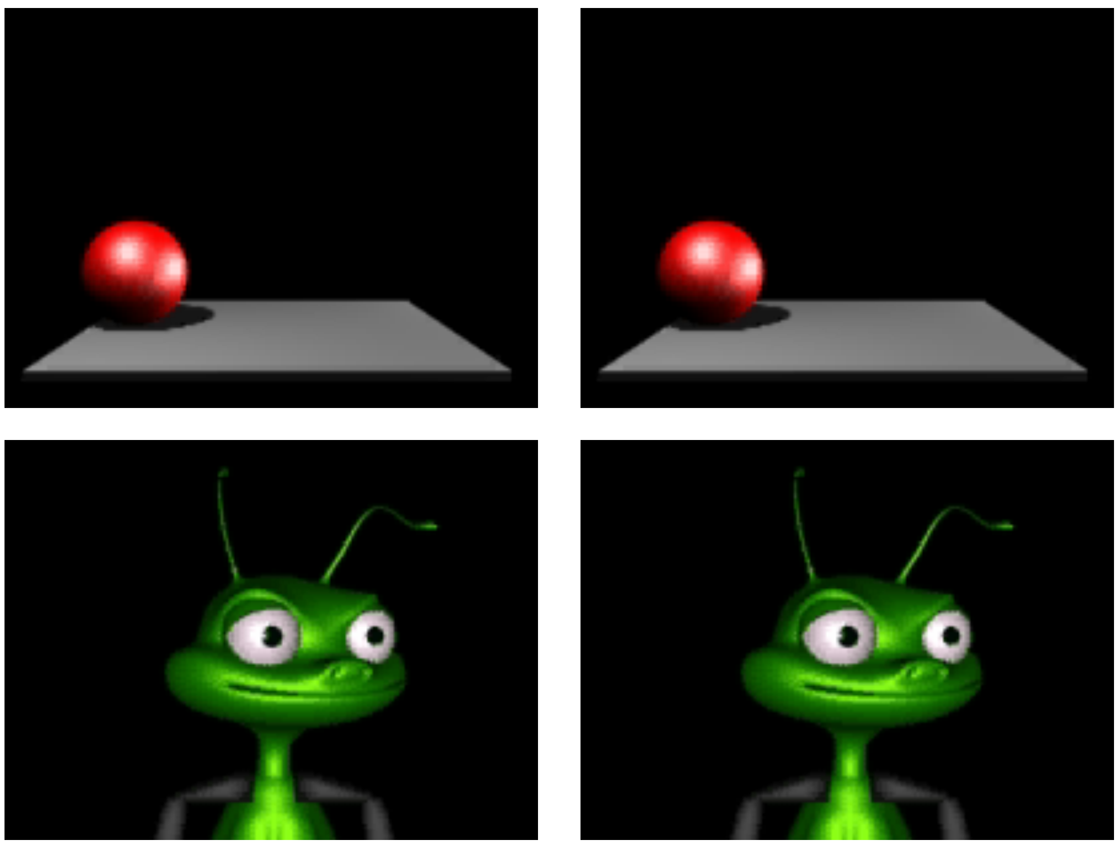
2. Ease in/out (or slow in/out)

- Real objects do not start and stop suddenly
  - Animation parameters shouldn't either
  - A little goes a long way (just a few frames acceleration or deceleration for "snappy" motions)

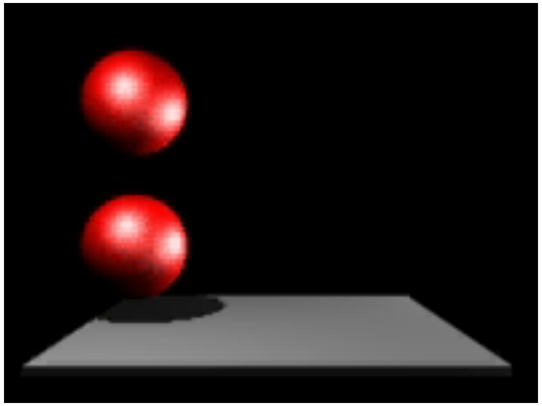
3. Moving in arcs

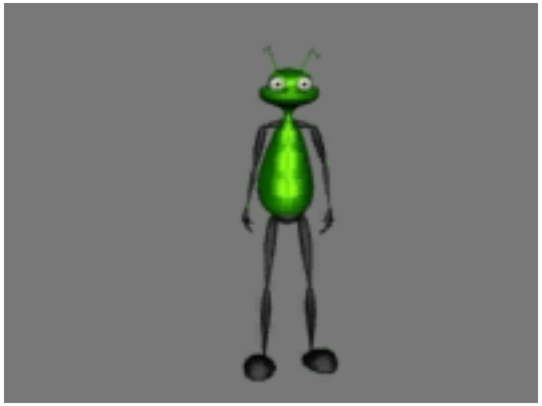
- Real objects also don't move in straight lines
  - Generally curves are more graceful and realistic

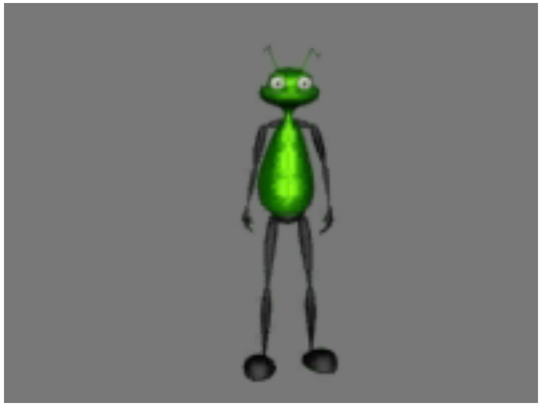


4. Anticipation

- Most actions are preceded by some kind of "wind-up"





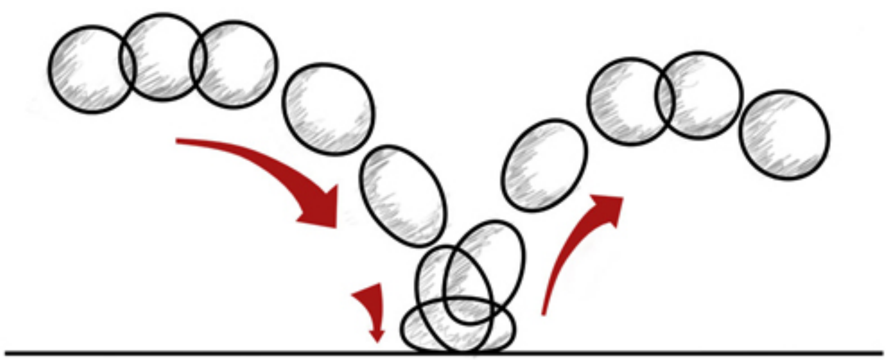


5. Exaggeration

- Animation is not always about modeling reality
- Exaggeration is very often used for emphasis

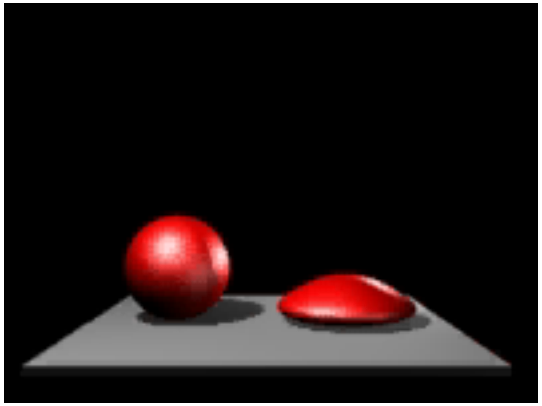
6. Squash and stretch

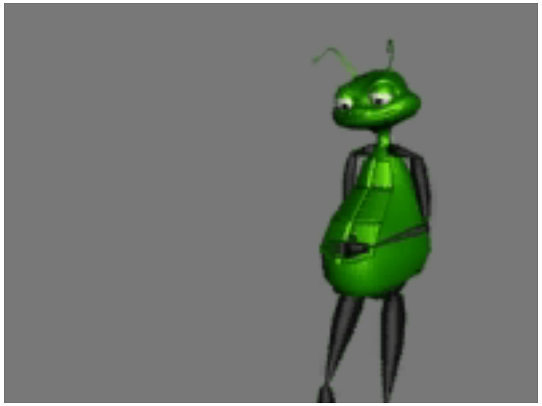
- Objects do not remain perfectly rigid as they move
- Adding stretch with motion and squash with impact:
  - Models deformation of soft objects
  - Indicates motion by simulating exaggerated "motion blur"



6. Squash and stretch (ii)

- Objects do not remain perfectly rigid as they move
- Adding stretch with motion and squash with impact:
  - Models deformation of soft objects
  - Indicates motion by simulating exaggerated "motion blur"



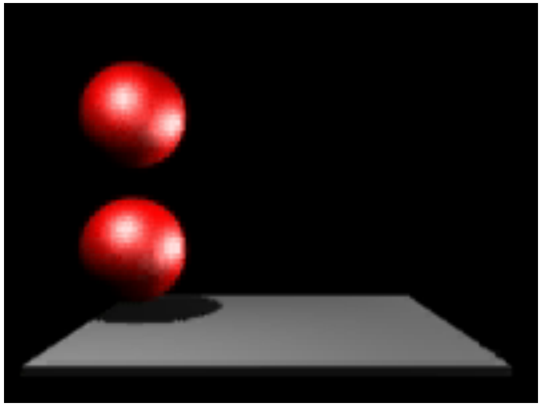


7. Secondary action

- Secondary action creates interest and realism in animation
- It should be staged such that it can be noticed but still *not overpower the main action*
- A good example of this is a character at a table acting and delivering their main acting
  - A side piece of acting business might be the character thumbing their fingers on the table
  - This isn't the main action say, perhaps it occurs as the other hand is more largely gesturing and your focus is on the face.

8. Follow through and overlapping action

- Follow through is the same as anticipation, only at the end of an action
- Overlapping action is an action that occurs because of another action

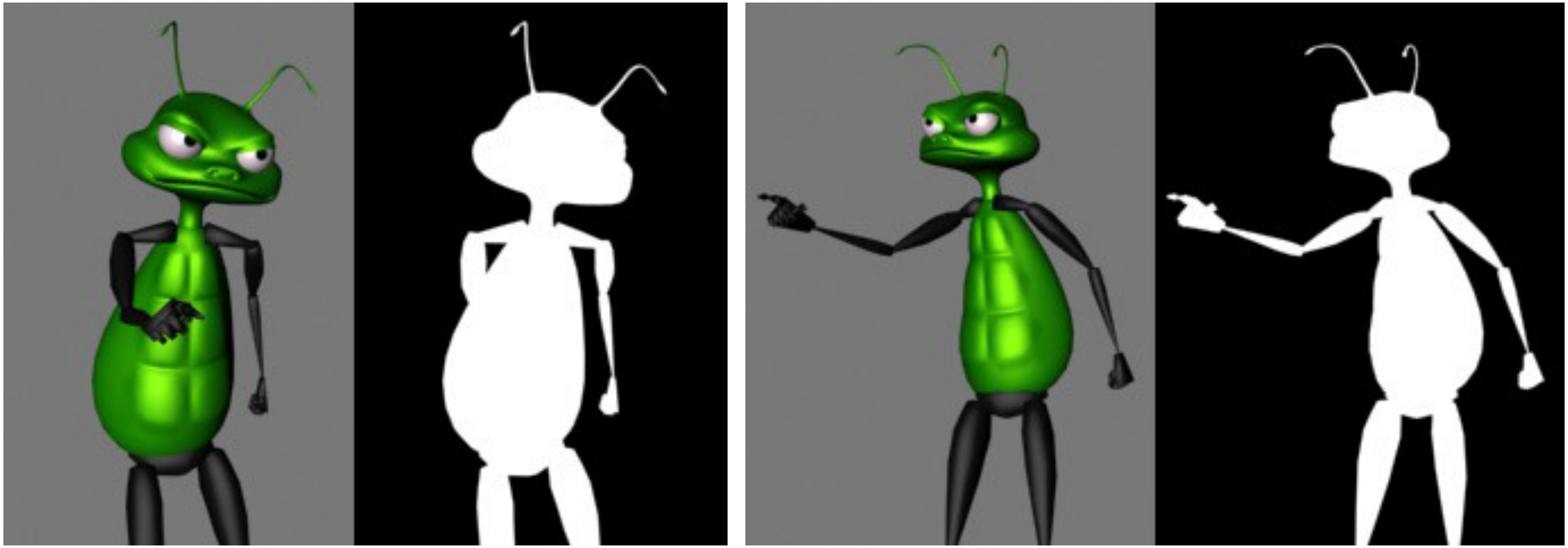


9. Animation strategy

- Straight ahead: draws or sets up objects one frame at a time in order
  - This approach tends to yield a more creative and fresh look but can be difficult to time correctly and tweak
- Pose to pose: excellent for tweaking timing and planning out the animation ahead of time
  - This is very useful when specific timing or action must occur at specific points
  - You always know exactly what will happen

10. Staging

- Want to produce clear, good-looking 2D images
  - Need good camera angles, set design, and character positions



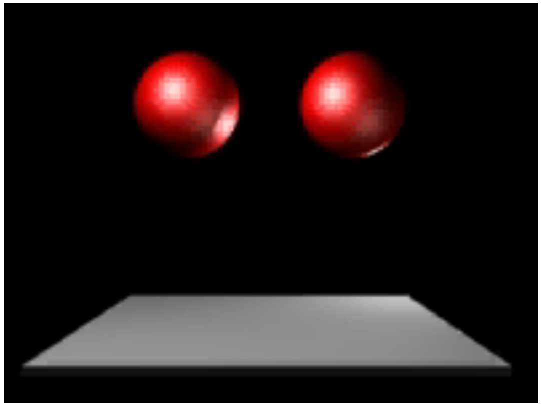
11. Appeal

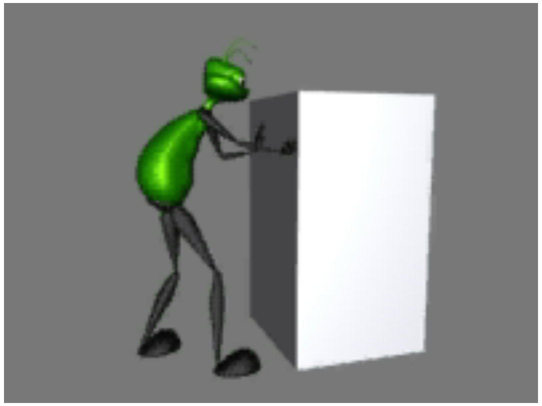
- Appeal means anything that a person likes to see
  - Quality of charm, design, simplicity, communication or magnetism
  - Appeal can be gained by correctly utilizing other principles such as exaggeration in design, avoiding symmetry, using overlapping action, and others.
- It's important to note that appeal doesn't necessarily mean good vs. evil
  - For example, in Disney's animated classic "Peter Pan", Captain Hook is an evil character, but most people would agree that his character and design has appeal

12. Personality

- Refers to the correct application of the other principles
  - Personality determines the success of an animation
- The animated creature really becomes alive and enters the true character of the role
- One character would not perform an action the same way in two different emotional states
- No two characters would act the same. It is also important to make the personality of a character distinct, but at the same time be familiar to the audience
- Personality has a lot to do with what is going on in the mind of the character, as well as the traits and mannerisms of the character

Principles in practice: weight





Controlling shape for animation

- Start with modeling DOFs (control points)
- Deformations control those DOFs at a higher level
  - E.g. Move first joint of second finger on left hand
- Animation controls control those DOFs at a higher level
  - E.g. open/close left hand
- Both cases can be handled by the same kinds of deformers

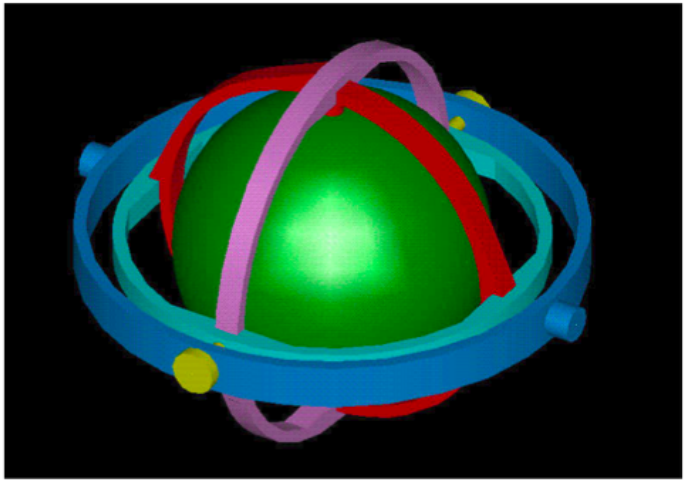
Interpolating transformations

- Linear interpolation of matrices is not effective
- One approach: always keep transformations in a canonical form (e.g. translate-rotate-scale)
  - Then the pieces can be interpolated separately
  - Rotations stay rotations, scales stay scales, all is good
- But you might be faced with just a matrix
  - What then?

Decomposing transformations

- A product M = TRS is not hard to take apart
  - Translation sits in the top right
  - RS still has three orthogonal columns
  - Scale factors are the lengths of the columns
- But that doesn't cover everything
  - Count DOFs: $3 + 3 + 3 < 12$
- If we allow S to be a scale along arbitrary axes then it will work, where
  - T is a translation
  - R is a rotation
  - S is a symmetric matrix (positive definite if no reflection)

Parameterizing rotations



- Euler angles
  - Rotate around x, then y, then z
  - Problem: gimbal lock
    - If two axes coincide, you lose one DOF
- Unit quaternions
  - A 4D representation (like 3D unit vectors for 2D sphere)
  - Good choice for interpolating rotations
- These are first examples of motion control
  - Matrix = deformation
  - Angles/quaternion = animation controls

Unit vectors

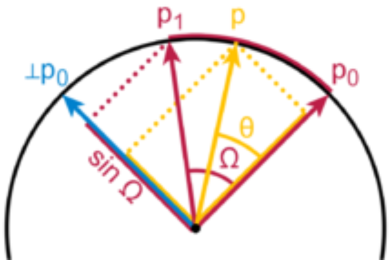
- When we want to represent directions we use unit vectors: points that are literally on the unit sphere in R^3
  - Now no points are special
  - Every point has a unique representation
  - Equal sized changes in coordinates are equal sized changes in direction
- Down side: one too many coordinates
  - Have to maintain normalization

Interpolating directions

- Interpolating in the space of 3D vectors is well behaved
  - Simple computation: interpolate linearly and normalize
- This is what we do all the time, e.g. with normals for fragment shading

$$\hat{\mathbf{v}}(t) = \mathrm{normalize}((1-t)\mathbf{v}_0 + t\mathbf{v}_1)$$

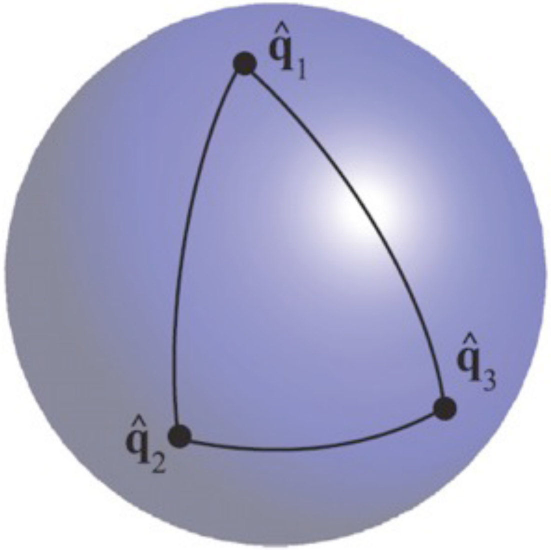
- For constant speed: spherical linear ir



$$P_S(t) = \frac{\sin[(1-t)\Omega]}{\sin(\Omega)} P + \frac{\sin[t\Omega]}{\sin(\Omega)} P'$$

Spherical linear interpolation

- Intuitive interpolation between different orientations
  - Nicely represented through quaternions
  - Useful for animation
  - Given two quaternions, interpolate between them
- Shortest path between two points on sphere



Quaternion interpolation

- Spherical linear interpolation naturally works in any dimension
- Traverses a great arc on the sphere of unit quaternions
  - Uniform angular rotation velocity about a fixed axis

$$\psi = \cos^{-1}(q_0 \cdot q_1)$$

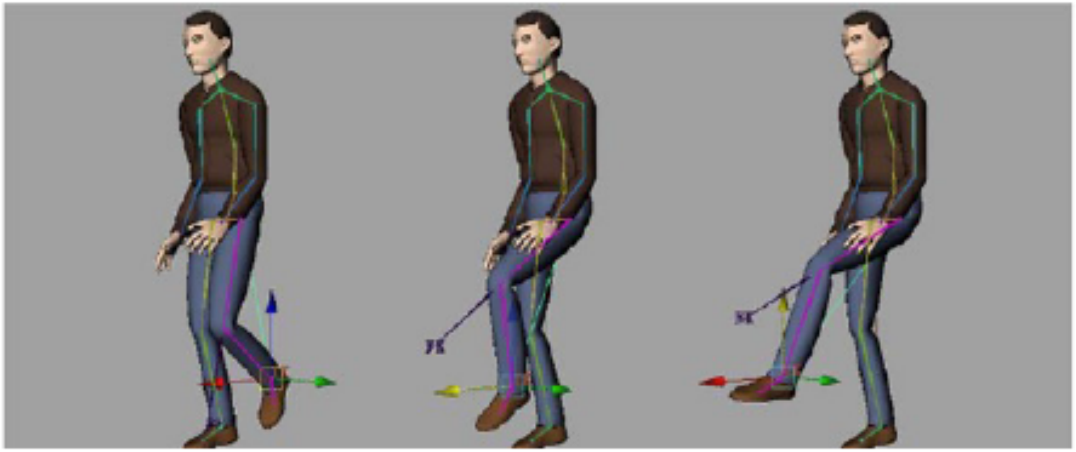
$$q(t) = \frac{q_0 \sin(1-t)\psi + q_1 \sin t\psi}{\sin \psi}$$

Hierarchies and articulated figures

- In mechanics, the relationship between DOFs and 3D pose is *kinematics*
- Robotics as source of mathematical methods
  - robots are transformation hierarchies
  - forward kinematics
  - inverse kinematics

Kinematics

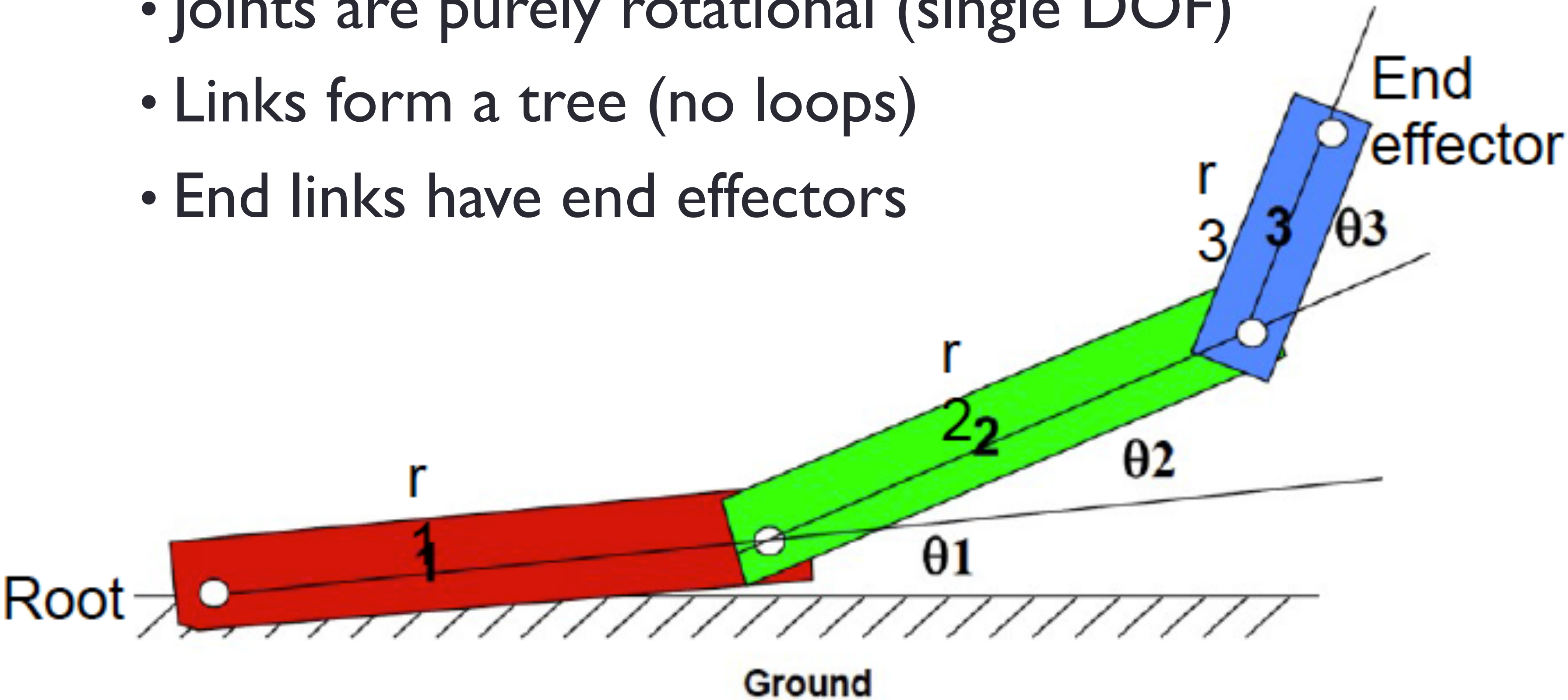
Forward Kinematics



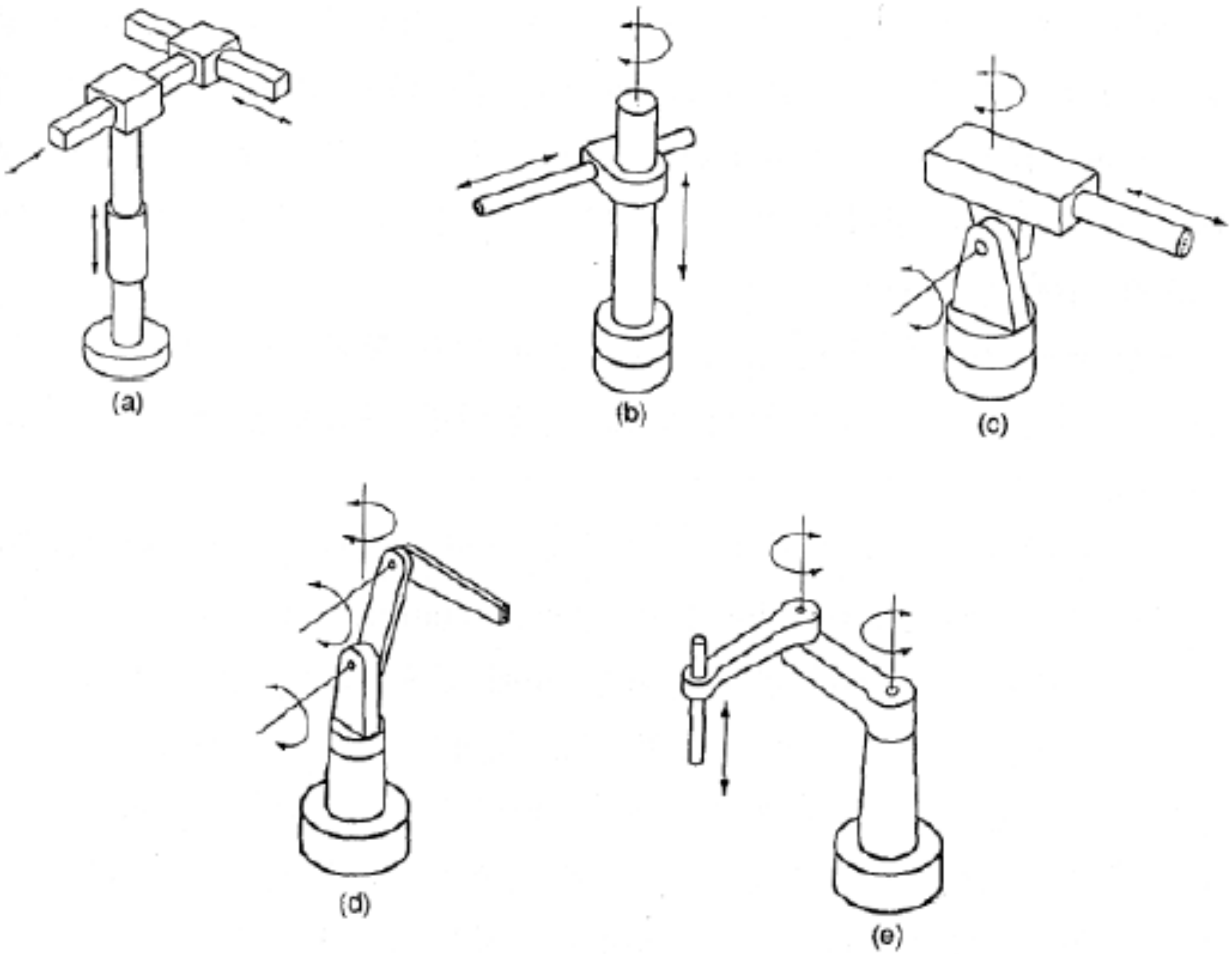
Inverse Kinematics

Rigid links and joint structure

- Links connected by joints
  - Joints are purely rotational (single DOF)
  - Links form a tree (no loops)
  - End links have end effectors



Articulation in robotics



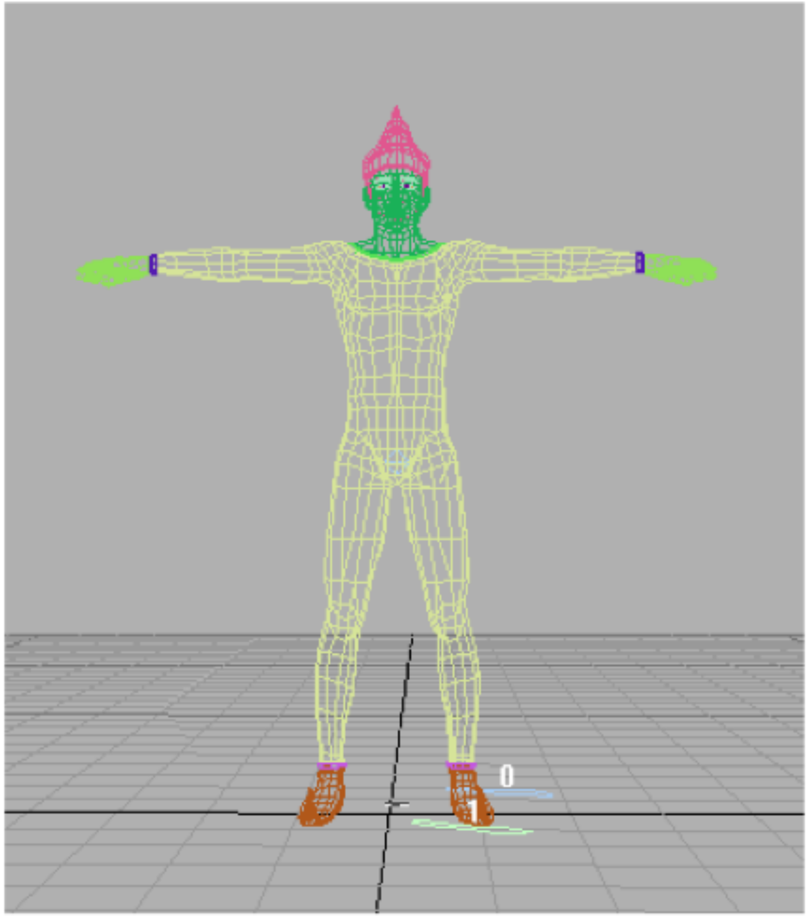
a. rectangular or cartesian
b. cylindrical or post-type
c. spherical or polar
d. joint-arm or articulated
e. SCARA (selective compliance assembly robot arm)

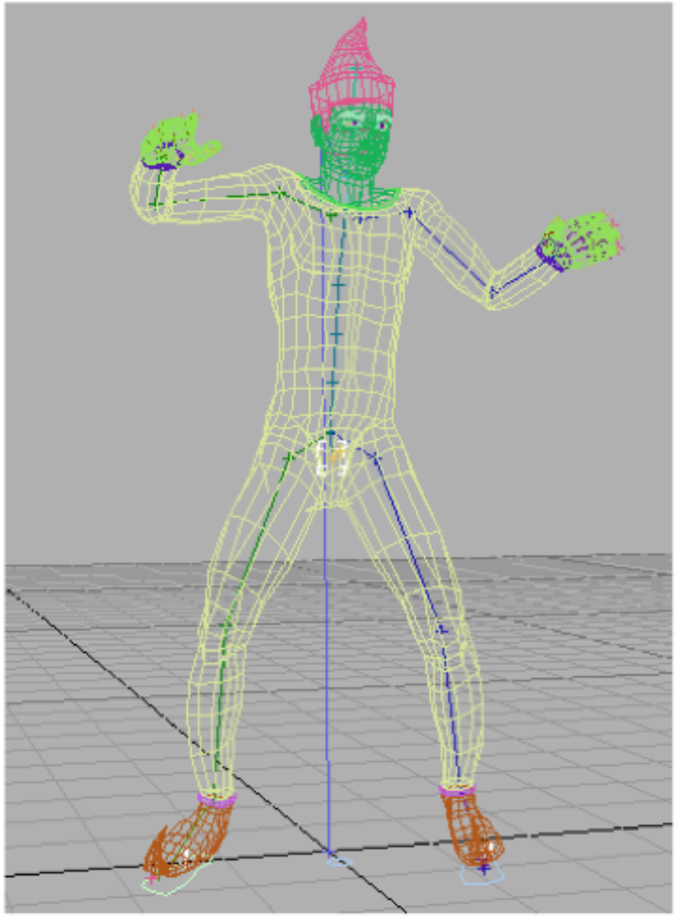
Basic surface deformation methods

- Mesh skinning: deform a mesh based on an underlying skeleton
- Blend shapes: make a mesh by combining several meshes
- Both use simple linear algebra
  - Easy to implement—first thing to try
  - Fast to run—used in games
- The simplest tools in the offline animation toolbox

Mesh skinning

- A simple way to deform a surface to follow a skeleton





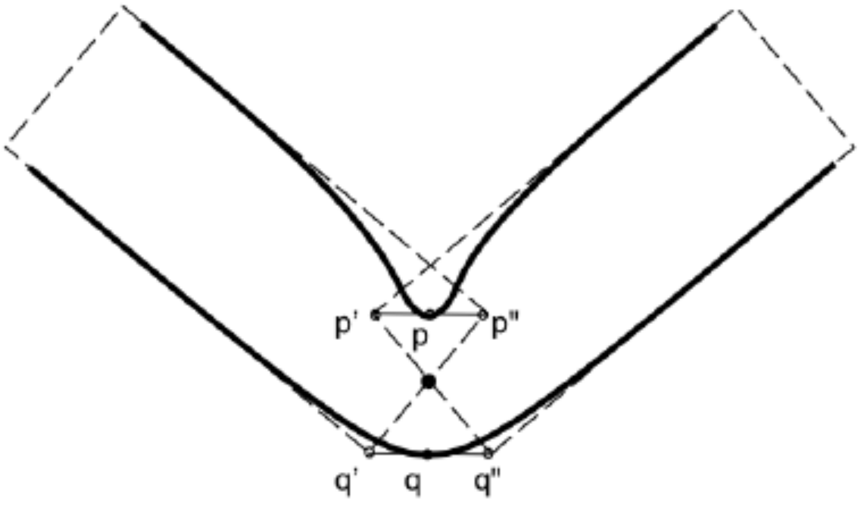
Mesh skinning math: setup

- Surface has control points p_i
  - Triangle vertices, spline control points, subdivide base vertices
- Each bone has a transformation matrix M_j
  - Normally a rigid motion
- Every point–bone pair has a weight w_{ij}
  - In practice only nonzero for small # of nearby bones
  - The weights are provided by the user

Mesh skinning math

- Deformed position of a point is a weighted sum
  - of the positions determined by each bone's transform alone
  - weighted by that vertex's weight for that bone

$$\mathbf{p}'_i = \sum_j w_{ij} M_j \mathbf{p}_i$$



[Lewis et al. SIGGRAPH 2000]

Mesh skinning

- Simple and fast to compute
  - Can even compute in the vertex stage of a graphics pipeline
- Used heavily in games
- One piece of the toolbox for offline animation
  - Many other deformers also available

Mesh skinning: classic problem

- Surface collapses on the inside of bends and in the presence of strong twists.

Figure 2: The 'collapsing elbow' in action, c.f. Figure 1.

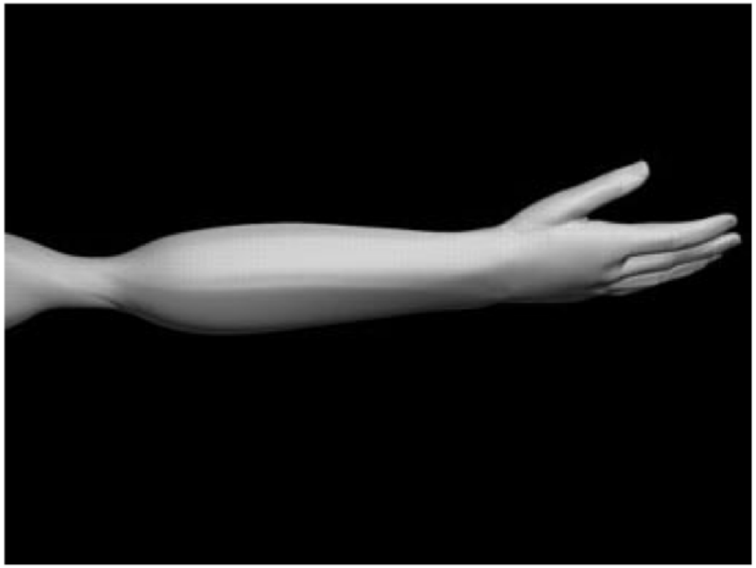
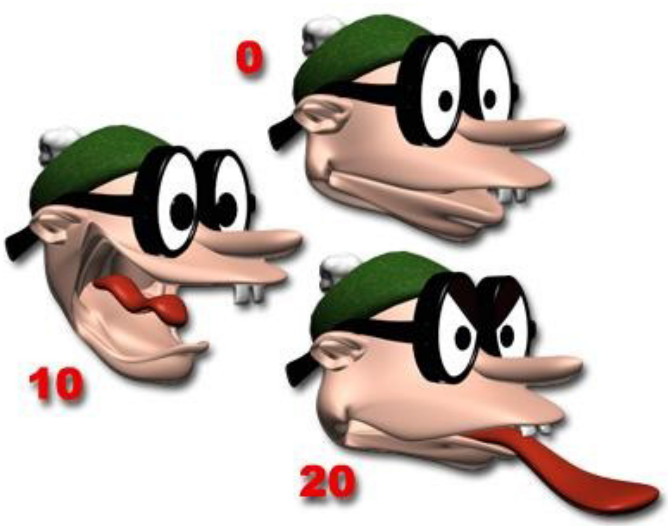
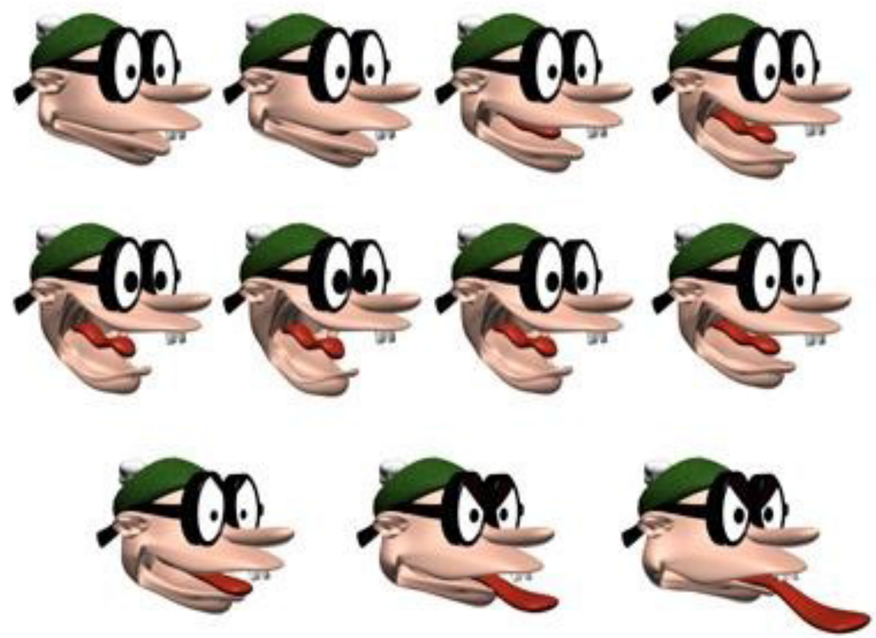


Figure 3: The forearm in the 'twist' pose, as in turning a door handle, computed by SSD. As the twist approaches 180° the arm collapses.

Blend shapes

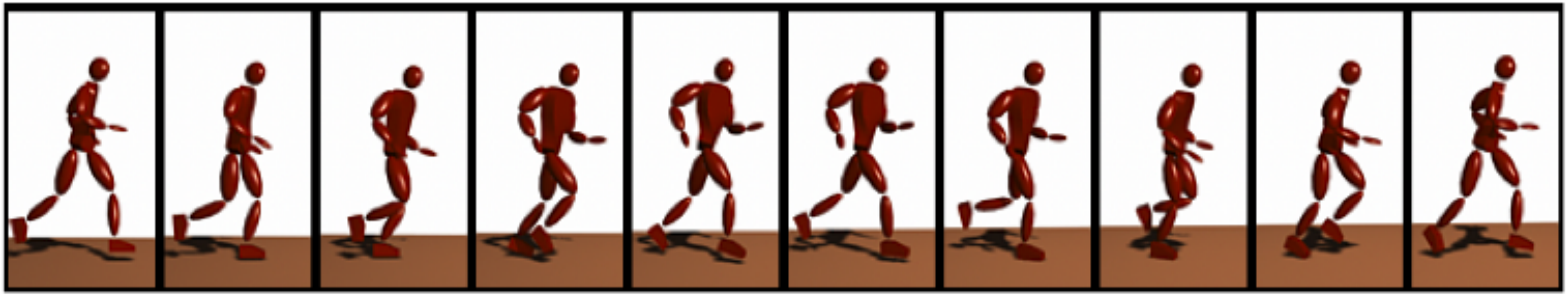
- Another very simple surface control scheme
- Based on interpolating among several key poses
  - Aka. blend shapes or morph targets





Motion capture

- A method for creating complex motion quickly: measure it from the real world

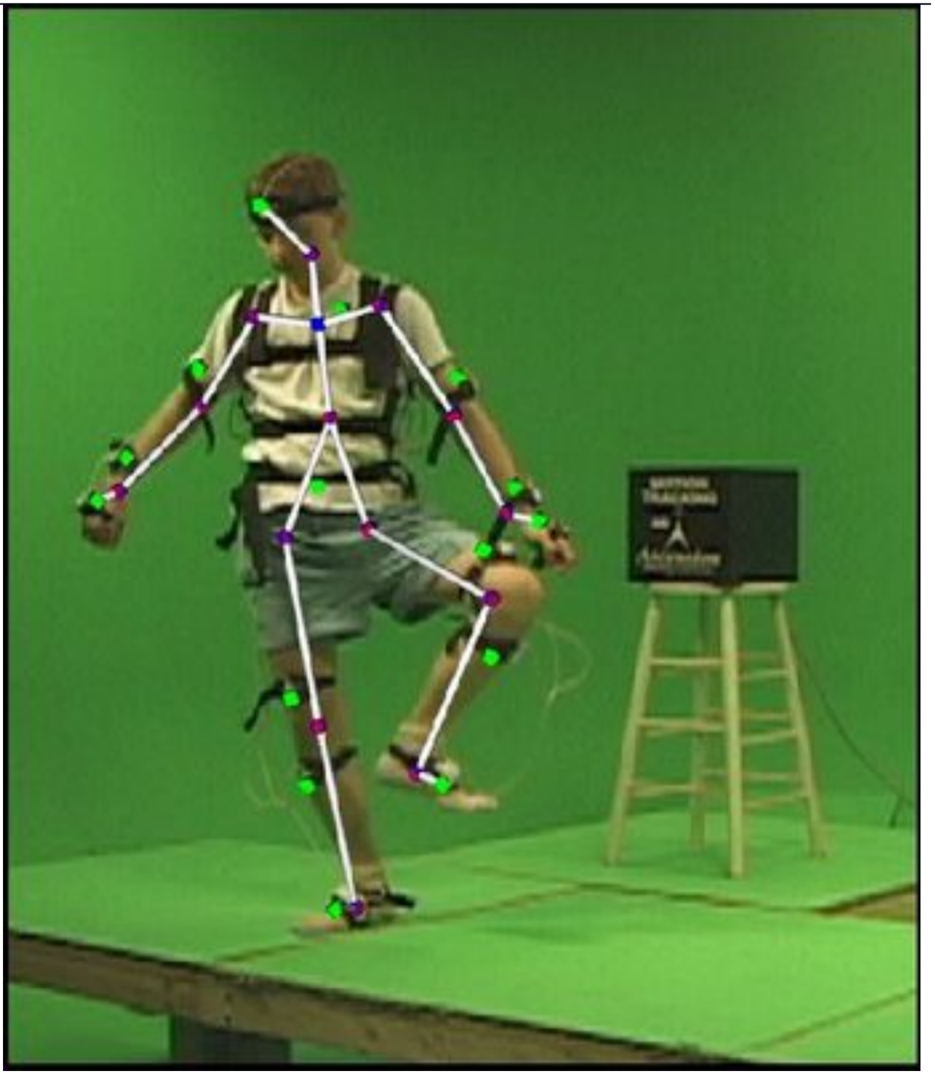


Motion capture in movies

Motion capture in games

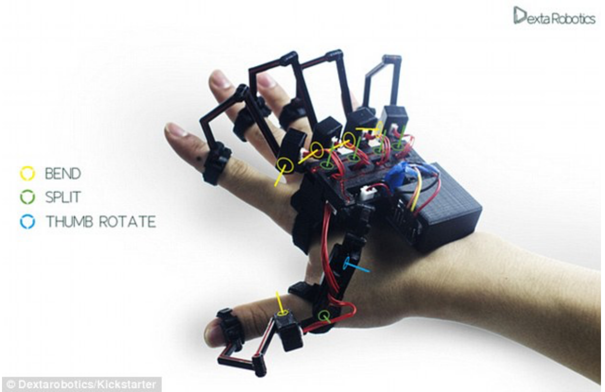
Magnetic motion capture

- Tethered
- Nearby metal objects cause distortions
- Low freq. (60Hz)



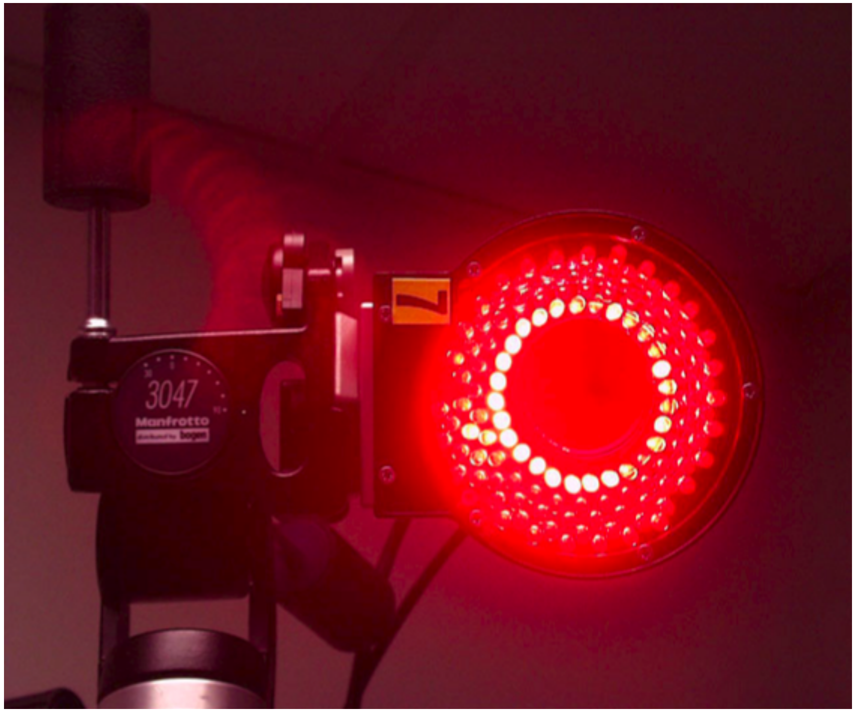
Mechanical motion capture

- Measures joint angles directly
- Works in any environment
- Restricts motion

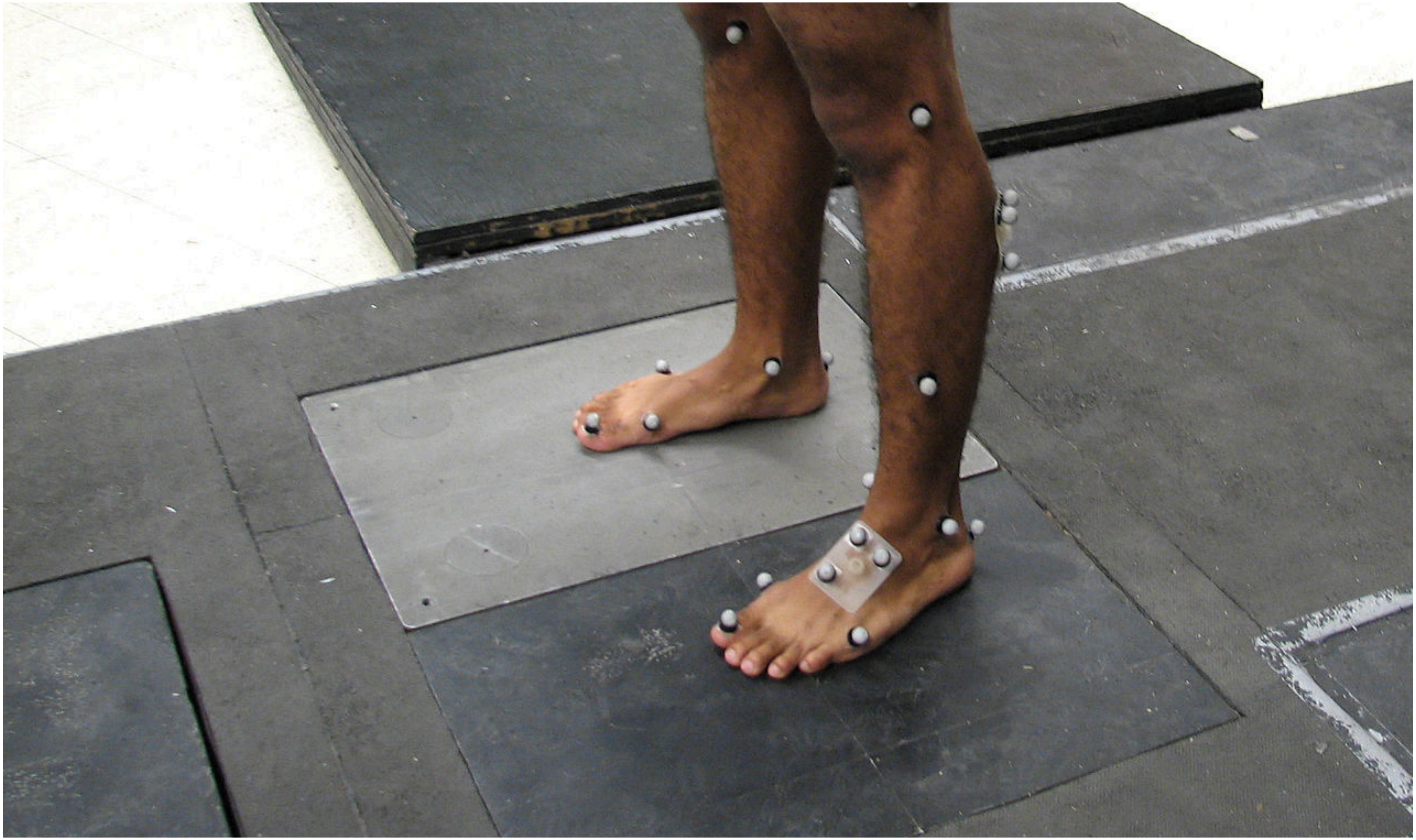


Optical motion capture

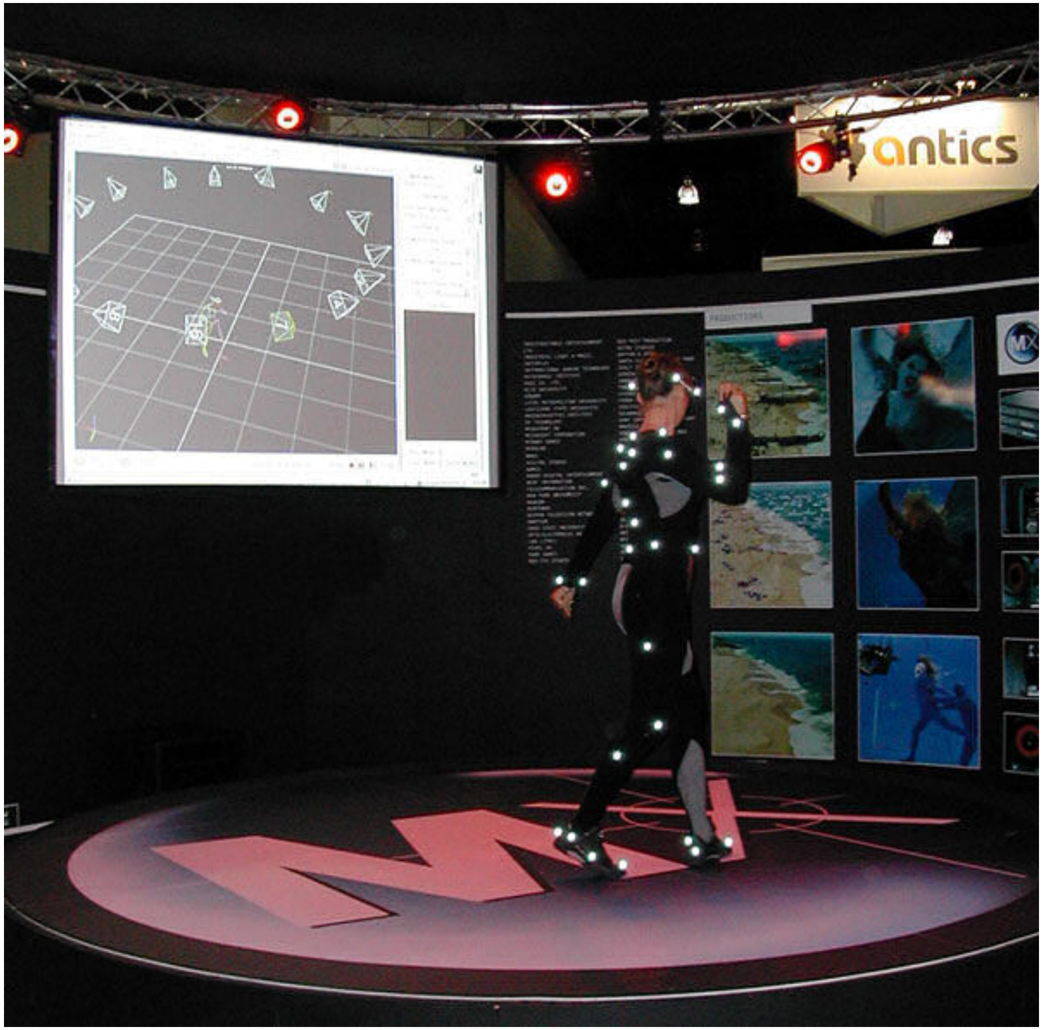
- Passive markers on subject, positions obtained by triangulation



Optical motion capture

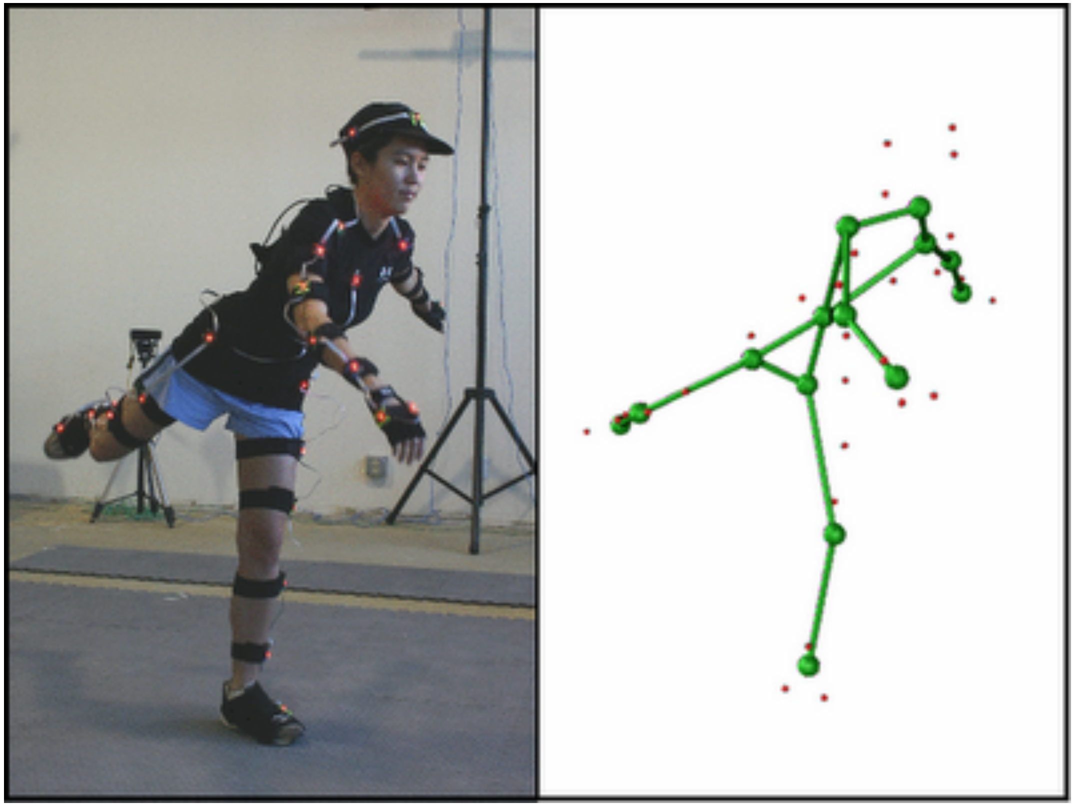


Optical motion capture



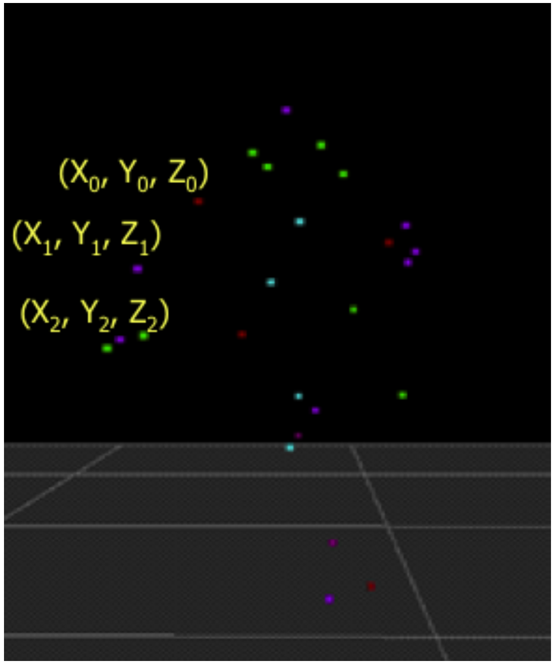
Optical motion capture

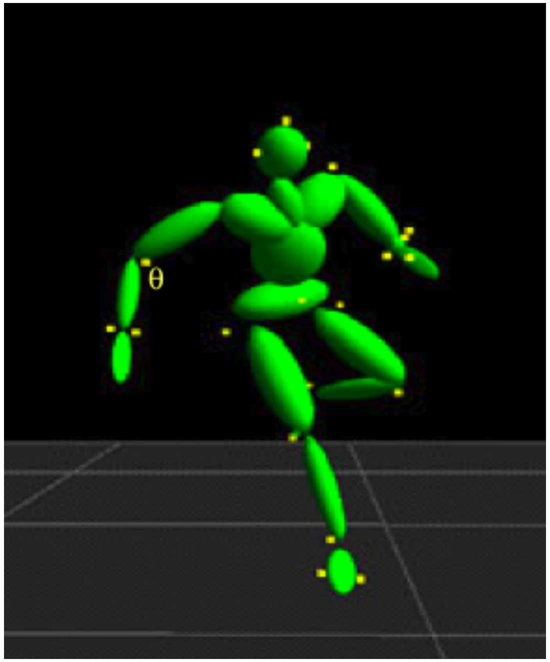
- 8 or more cameras
- Restricted volume
- High frequency (240Hz)
- Occlusions are troublesome



From marker data to usable motion

- Motion capture system gives inconvenient raw data
  - Optical is "least information" case: accurate position but:
    - Which marker is which?
    - Where are the markers are relative to the skeleton?





Motion capture data processing

- Marker identification: which marker is which
  - Start with standard rest pose
  - Track forward through time (but watch for markers dropping out due to occlusion!)
- Calibration: match skeleton, find offsets to markers
  - Use a short sequence that exercises all DOFs of the subject
  - A nonlinear minimization problem
- Computing joint angles: explain data using skeleton DOFs
  - A inverse kinematics problem per frame!

Motion capture in context

- Mocap data is very realistic
  - Timing matches performance exactly
  - Dimensions are exact
- But it is not enough for good character animation
  - Too few DOFs
  - Noise, errors from nonrigid marker mounting
  - Contains no exaggeration
  - Only applies to human-shaped characters
- Therefore *mocap* data is generally a starting point for skilled animators to create the final product

References

- Principles of animation
  - https://vimeo.com/93206523
- Real time Shape Deformation (SIGGRAPH 2014)
  - http://skinning.org