Intro to Keyframe Animation

Steve Marschner
CS 4620
Cornell University

Animation principles

The artistic process of animation

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

Timing

sad

happy

examples from *Luxo, Jr.* (Pixar, 1986)

Timing

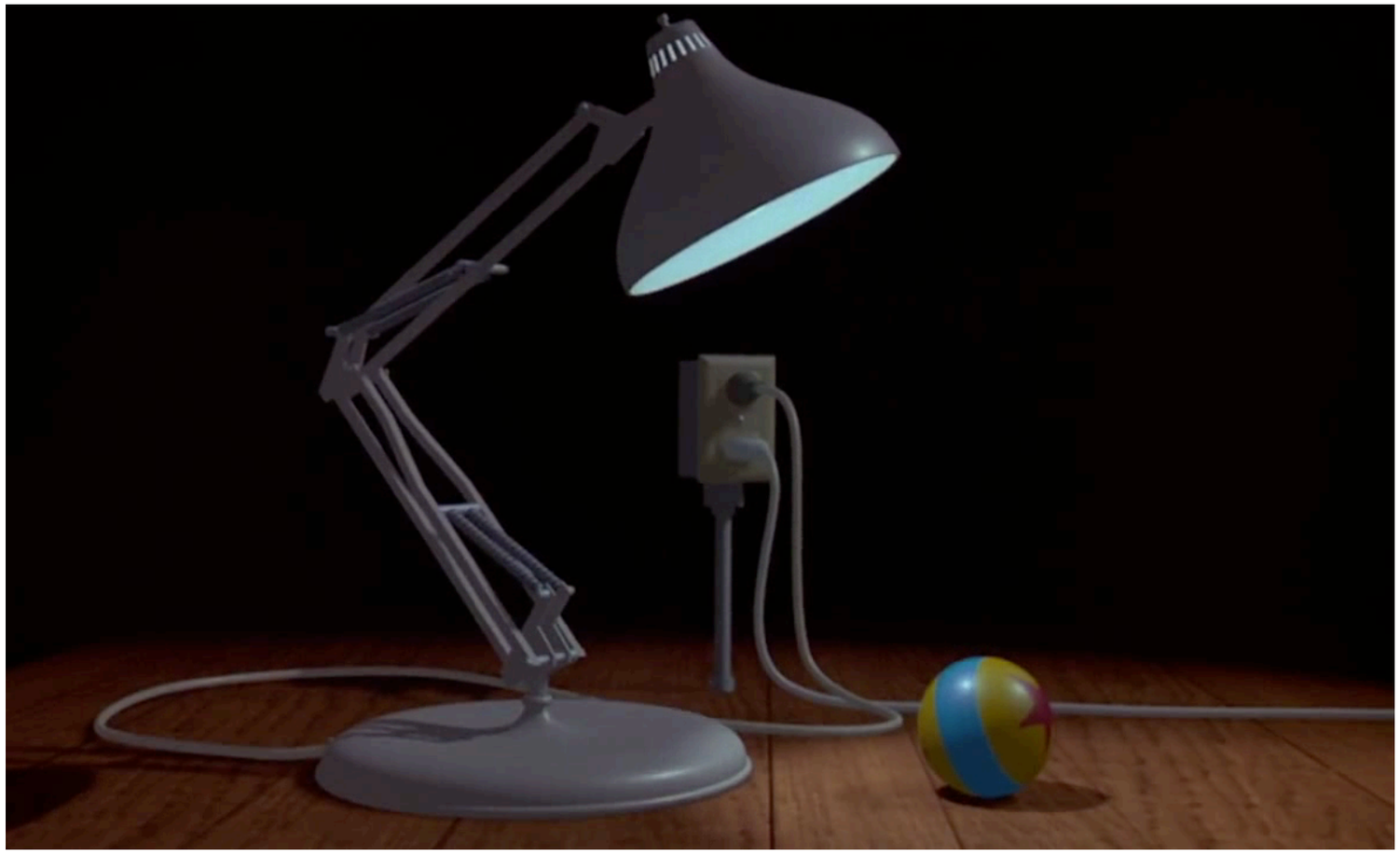
sad

happy

examples from *Luxo, Jr.* (Pixar, 1986)

Anticipation & Follow-through

cautious

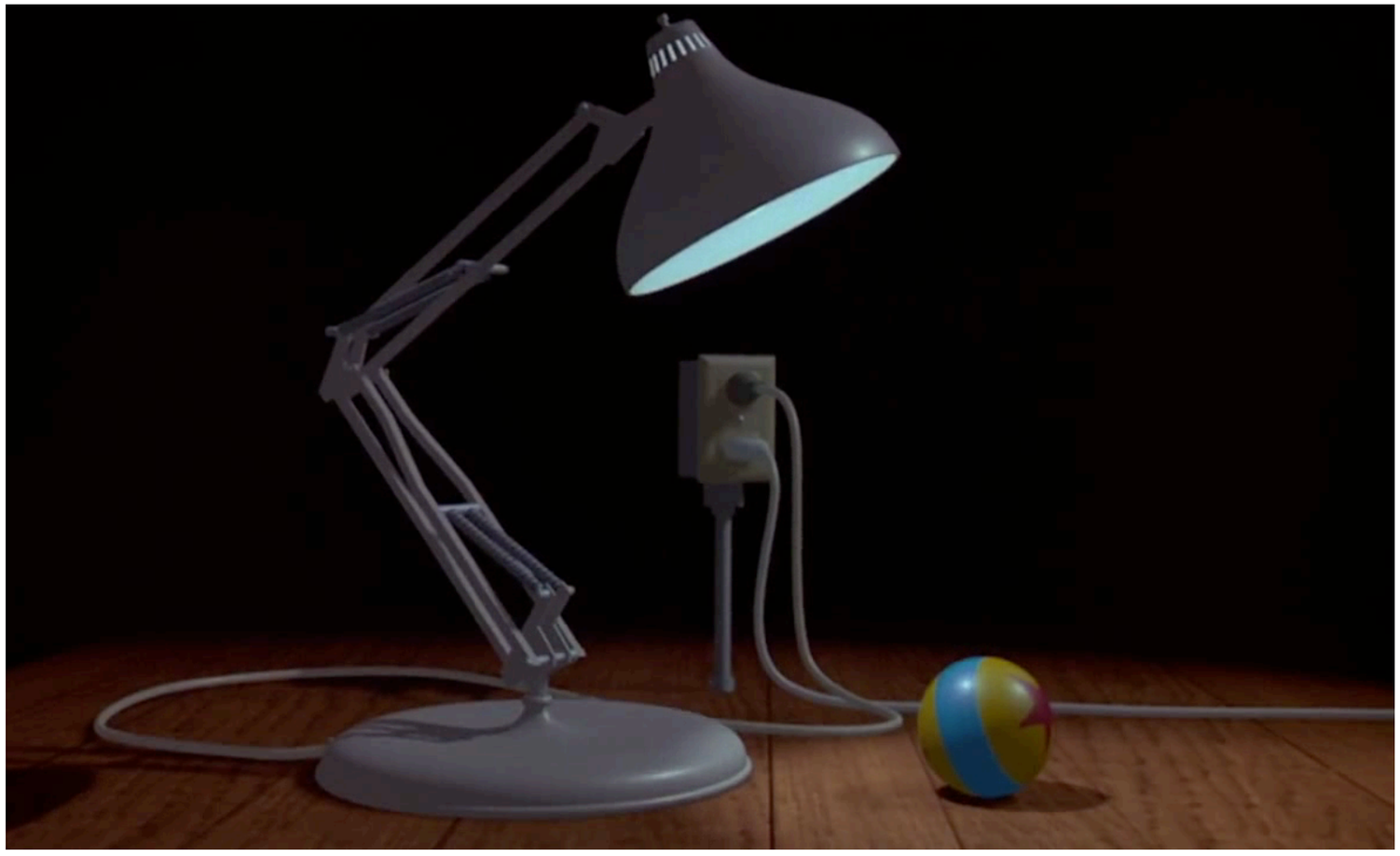


enthusiastic

(with previous instructors James/Bala)

Anticipation & Follow-through

cautious

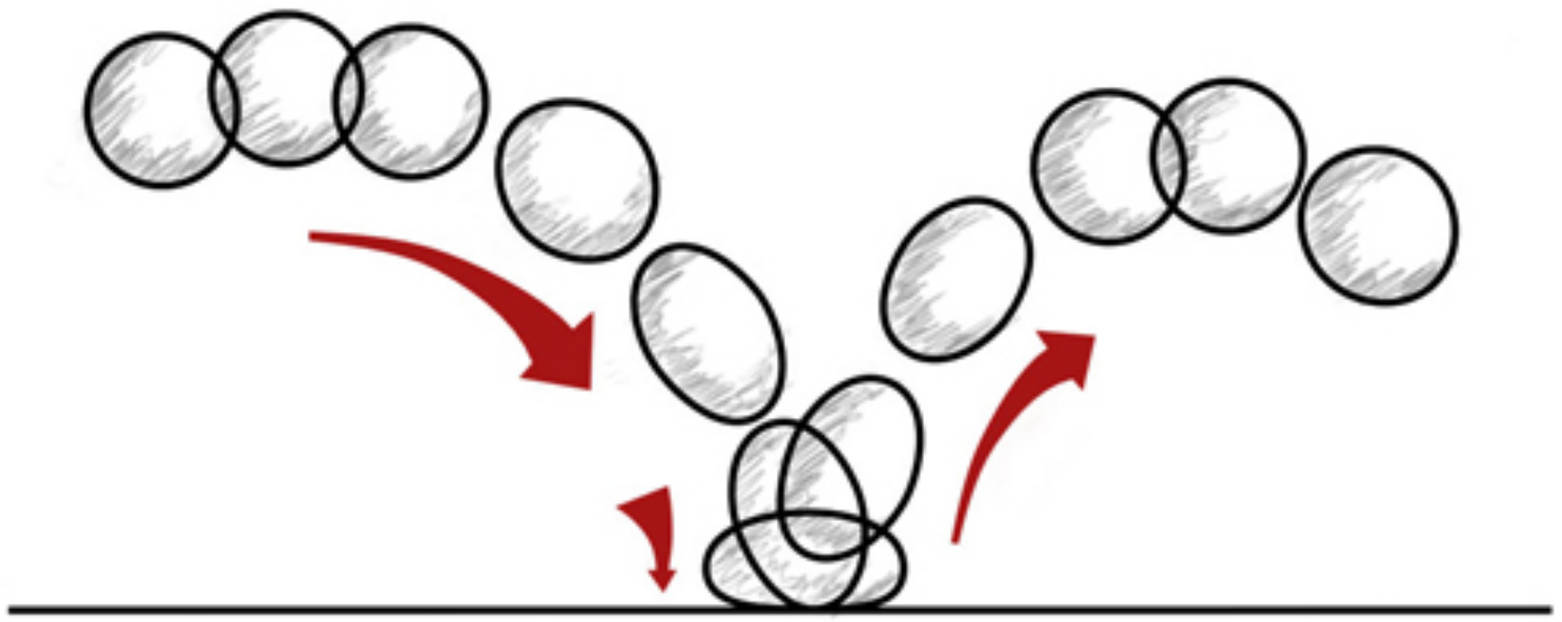


enthusiastic

(with previous instructors James/Bala)

Animation principles: squash & 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"



[www.animdesk.com]

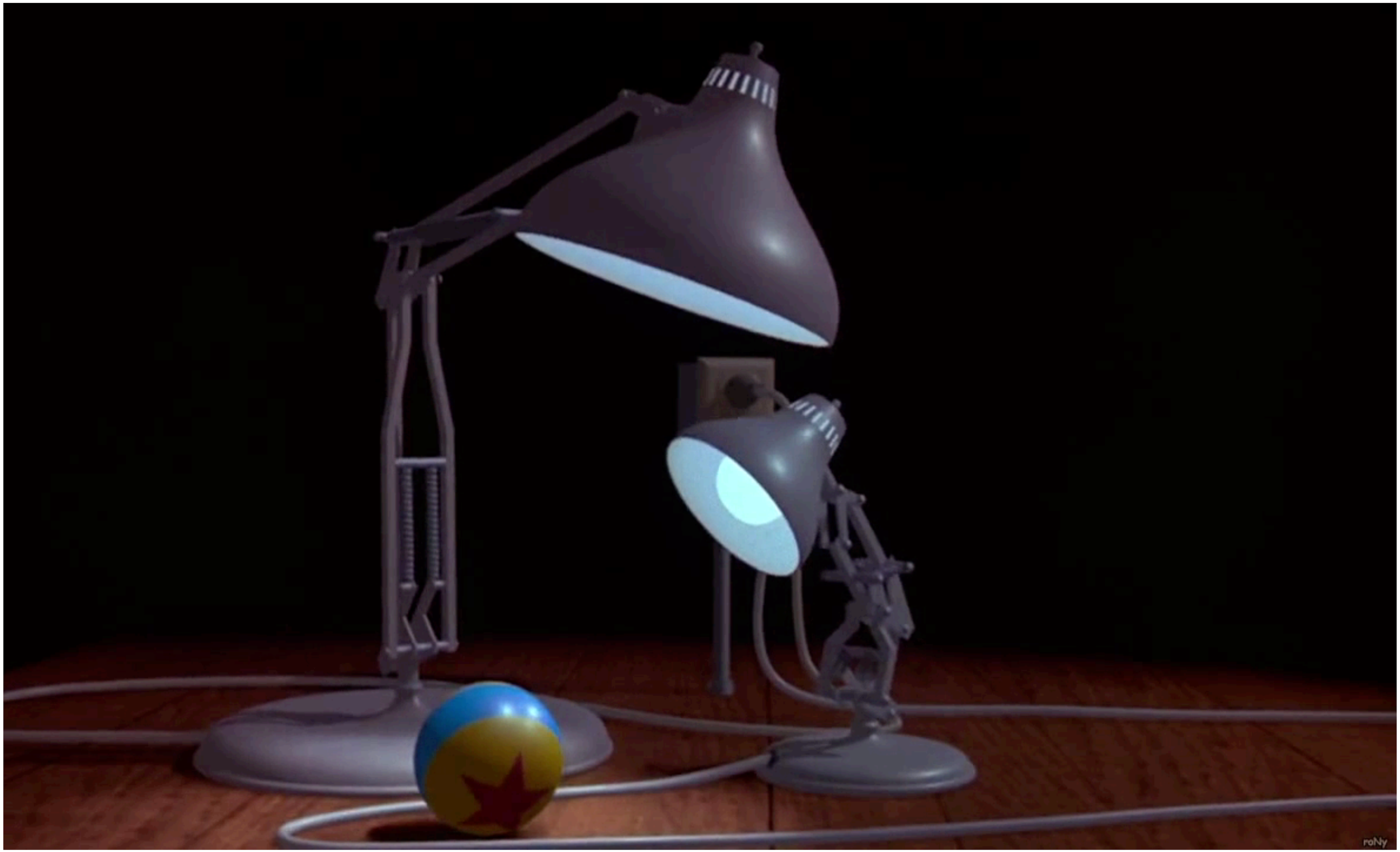
Follow-through

Follow-through

Overlapping action

Overlapping action

Staging and composition



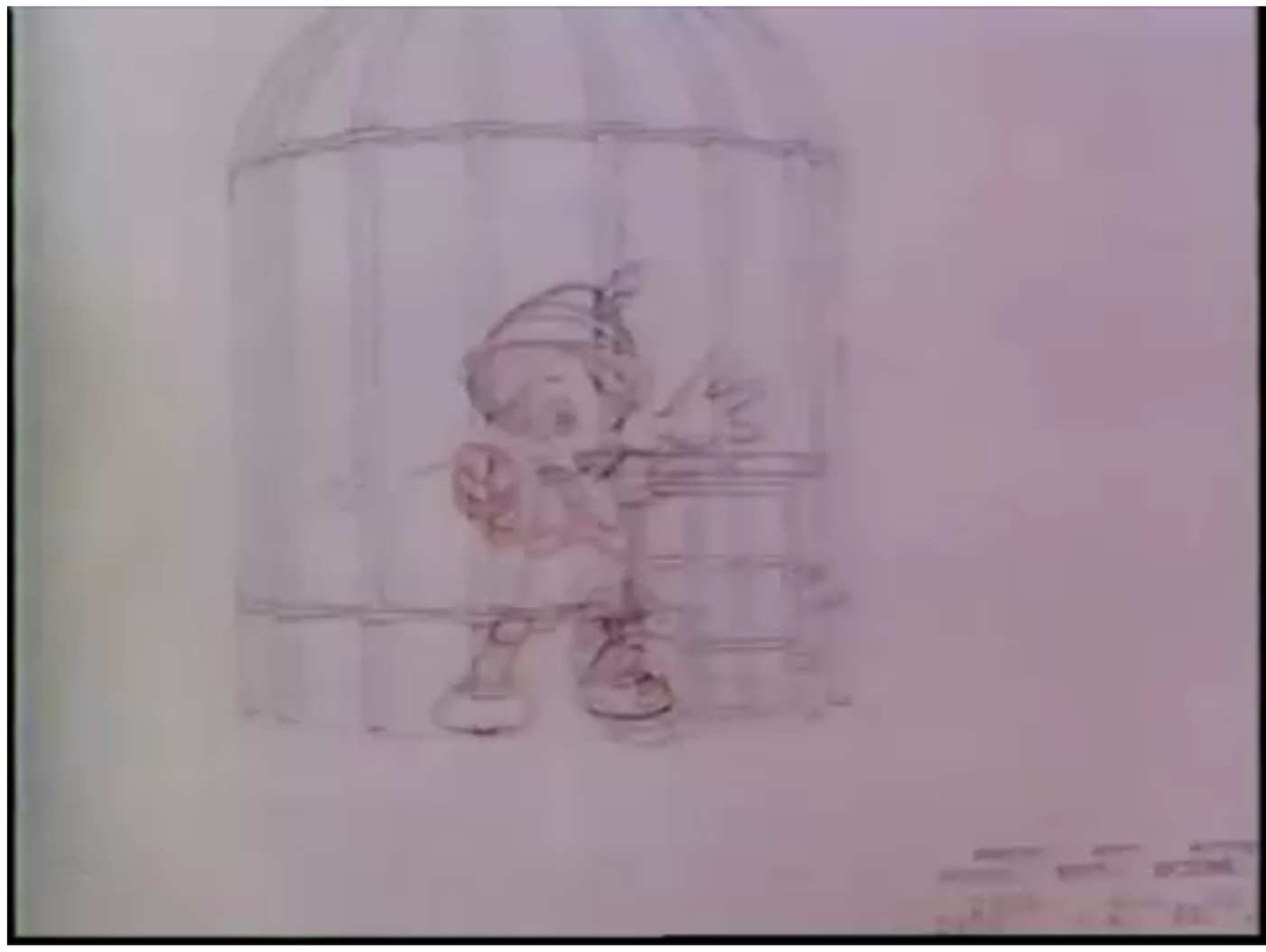
Keyframe animation

What is animation?

- Modeling = specifying shape
  - using all the tools we've seen: hierarchies, meshes, curved surfaces…
- Animation = specifying shape as a function of time
  - just modeling done once per frame?
  - yes, but need smooth, concerted movement

Keyframes in hand-drawn animation

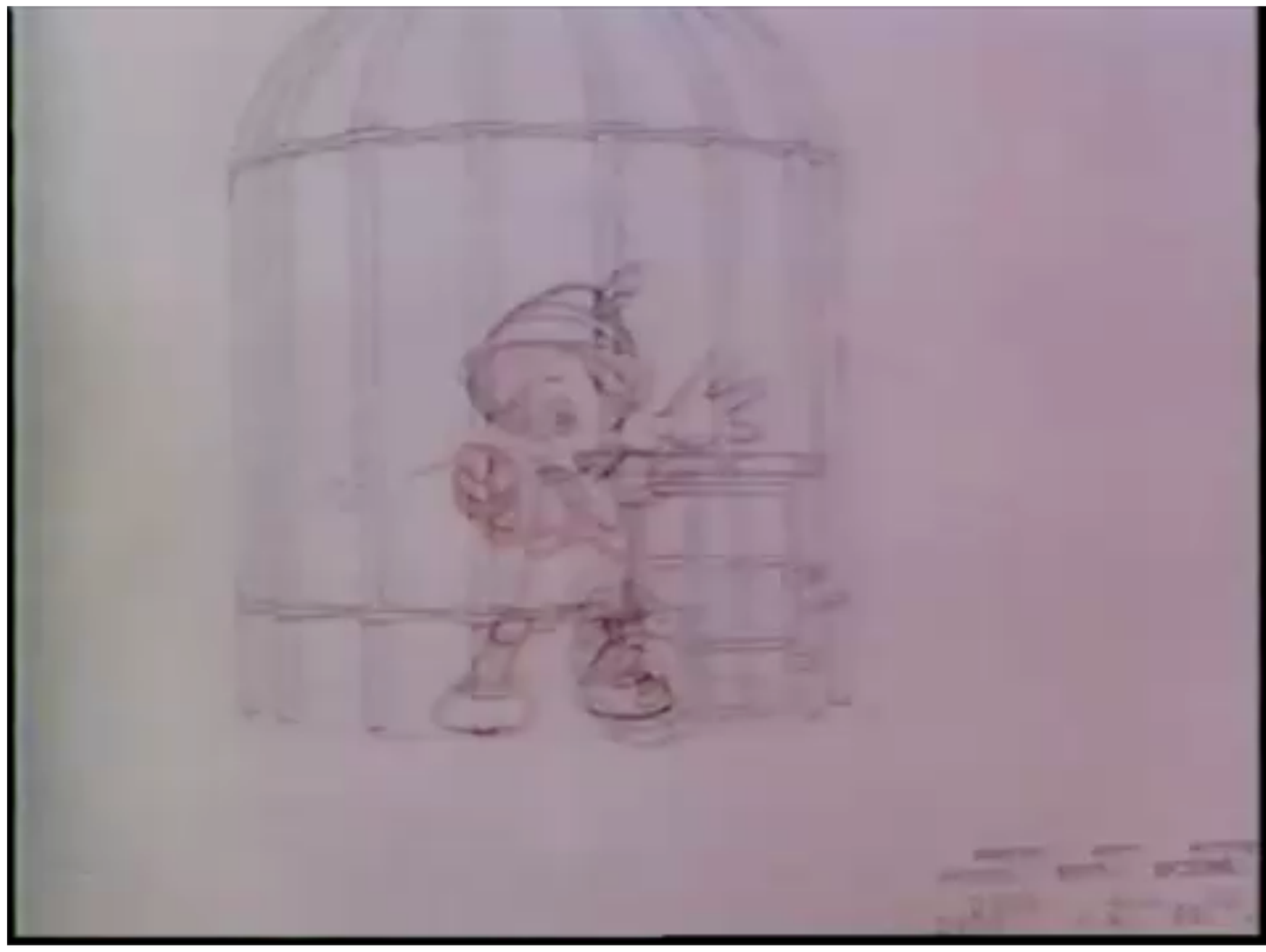
- End goal: a drawing per frame, with nice smooth motion
- "Straight ahead" is drawing frames in order (using a lightbox to see the previous frame or frames)
  - but it is hard to get a character to land at a particular pose at a particular time
- Instead use *key frames* to plan out the action
  - draw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



animation by Ollie Johnston, © Disney

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)
  - but it is hard to get a character to land at a particular pose at a particular time
- Instead use *key frames* to plan out the action
  - draw important poses first, then fill in the *in-betweens*

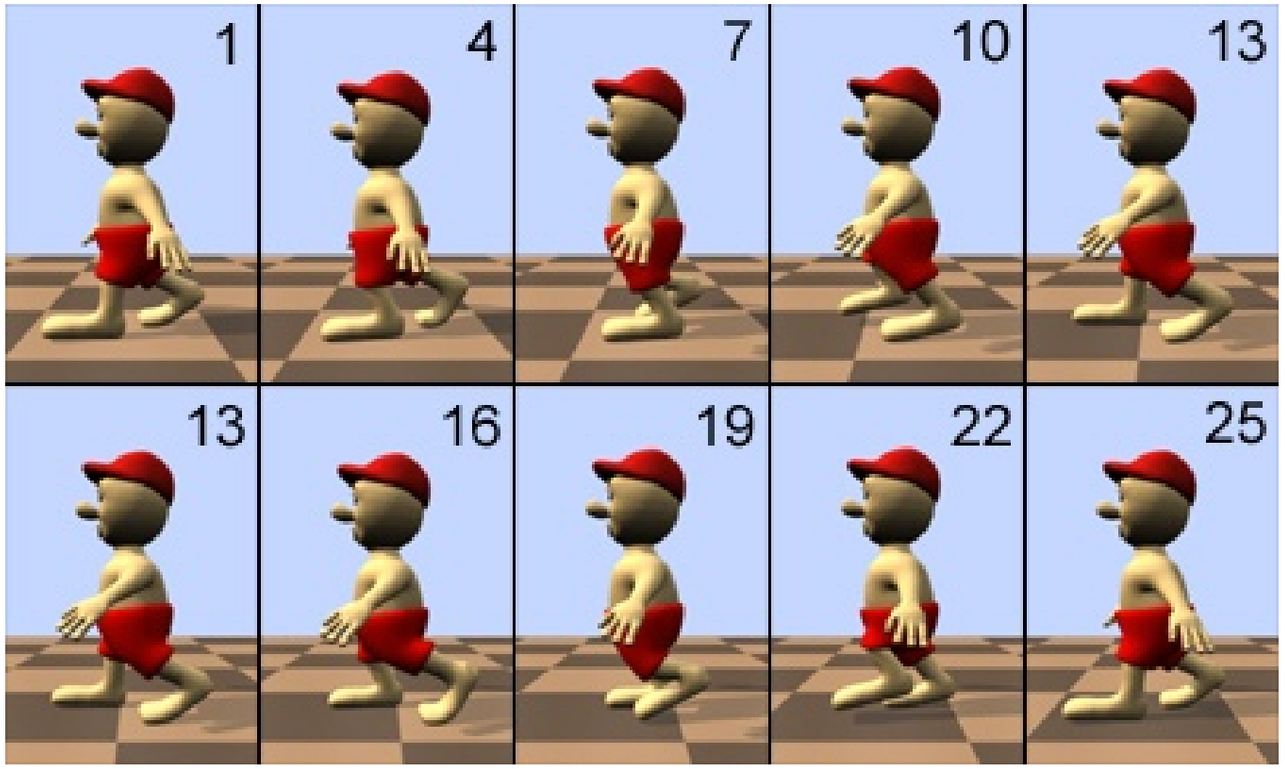


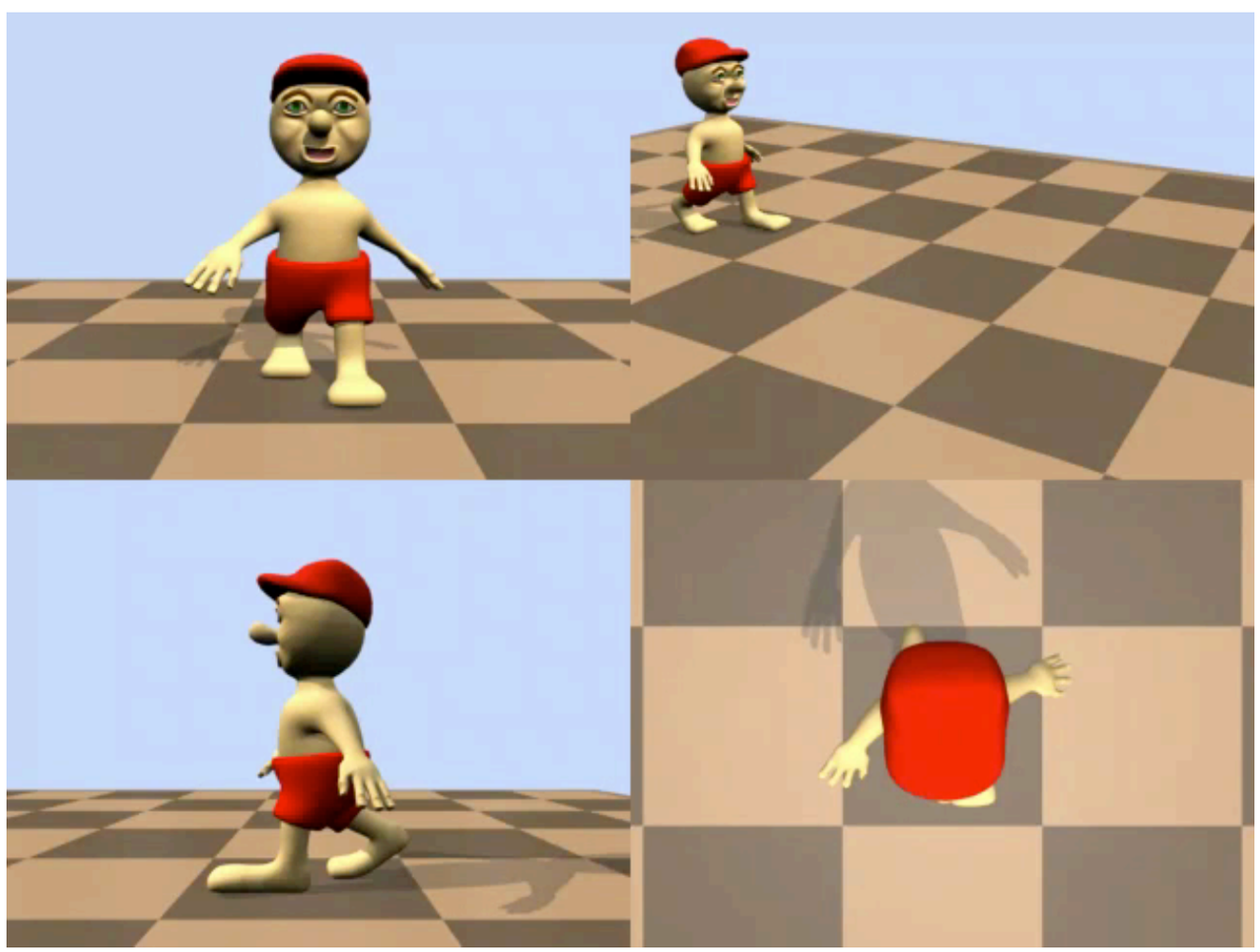
animation by Ollie Johnston, © Disney

Keyframe animation

- Hand-drawn 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
- Computer animation
  - in computer animation substitute "user" and "animation software"
  - *interpolation* is the principal operation
- Key ideas of computer animation
  - create *high-level* controls for adjusting geometry
  - *interpolate* these controls over time between keyframes

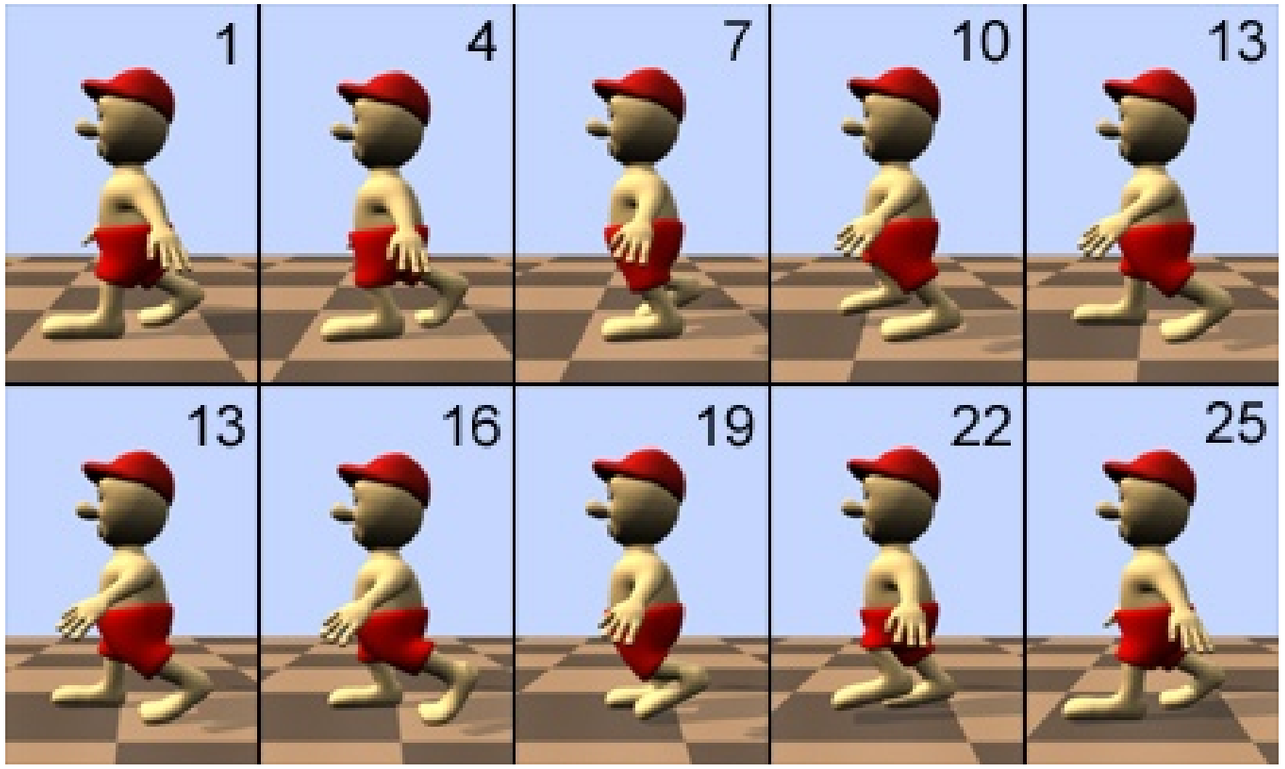
Walk cycle

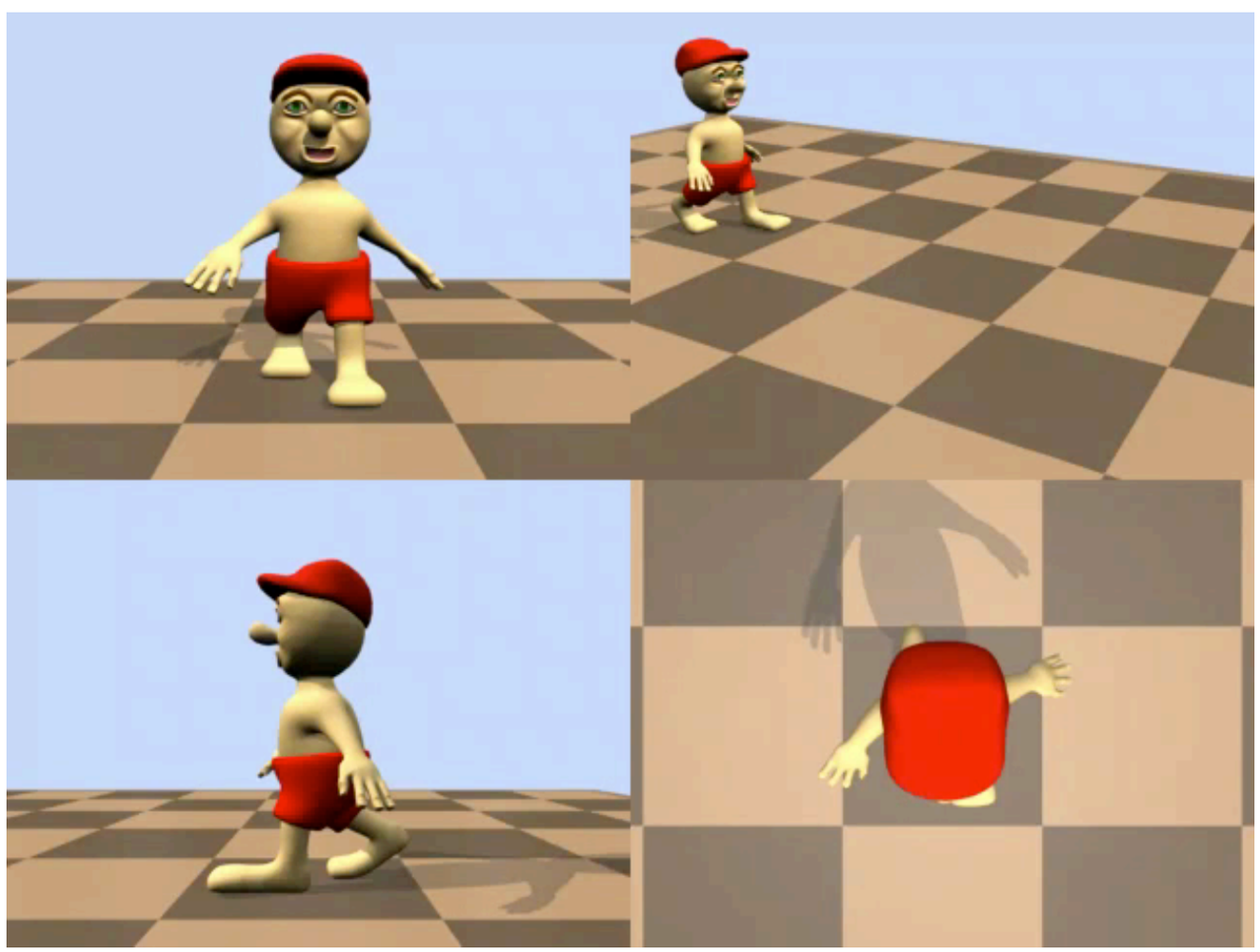




[Christopher Lutz http://www.animationsnippets.com]

Walk cycle





[Christopher Lutz http://www.animationsnippets.com]

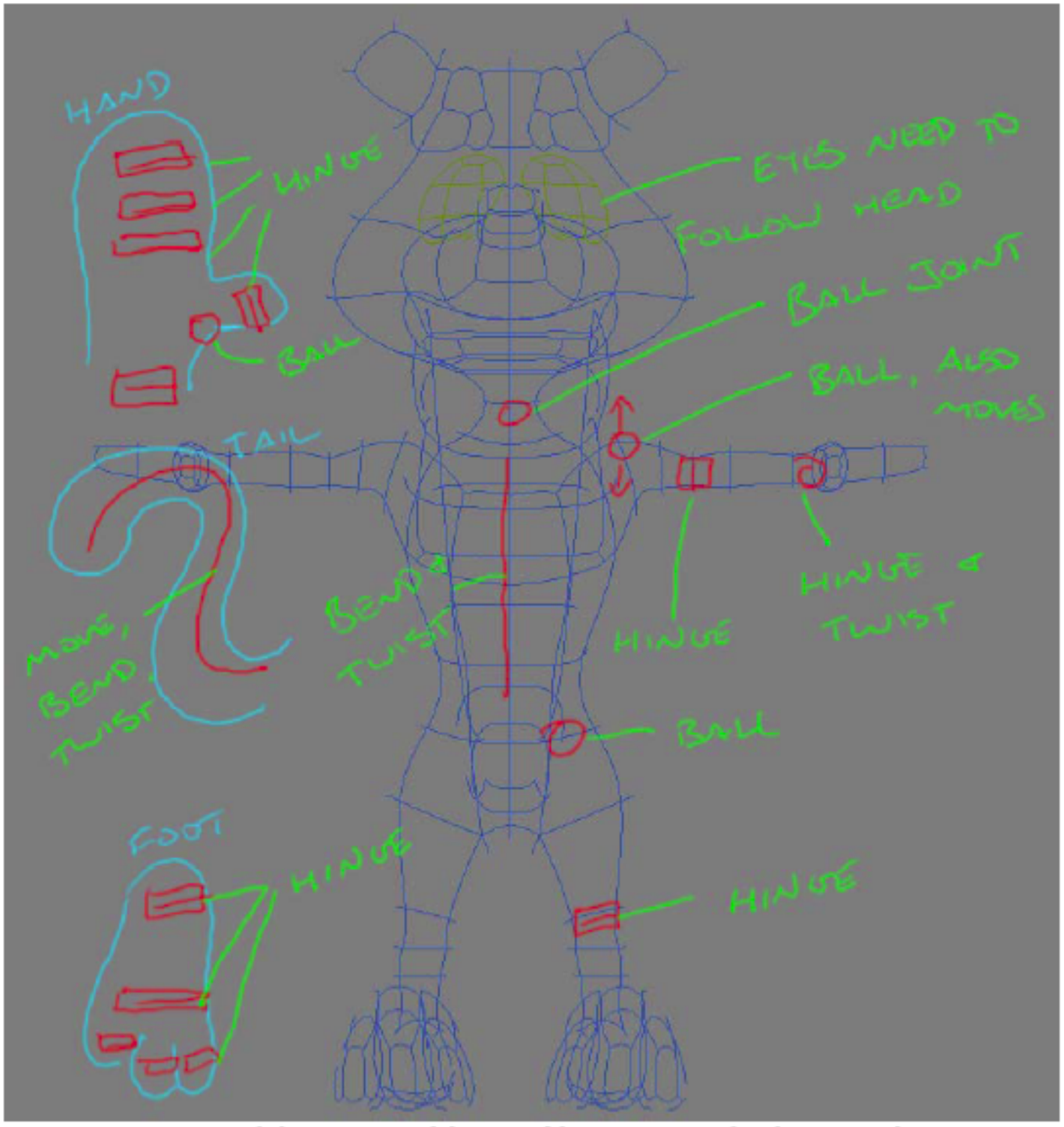
Controlling geometry conveniently

- Manually place every control point at every keyframe?
  - labor intensive
  - hard to get smooth, consistent motion
- Animate using smaller set of meaningful *degrees of freedom*
  - modeling DOFs are inappropriate for animation
    e.g. *"move one square inch of left forearm"*
  - animation DOFs need to be higher level
    e.g. *"bend the elbow"*

Controlling shape for animation

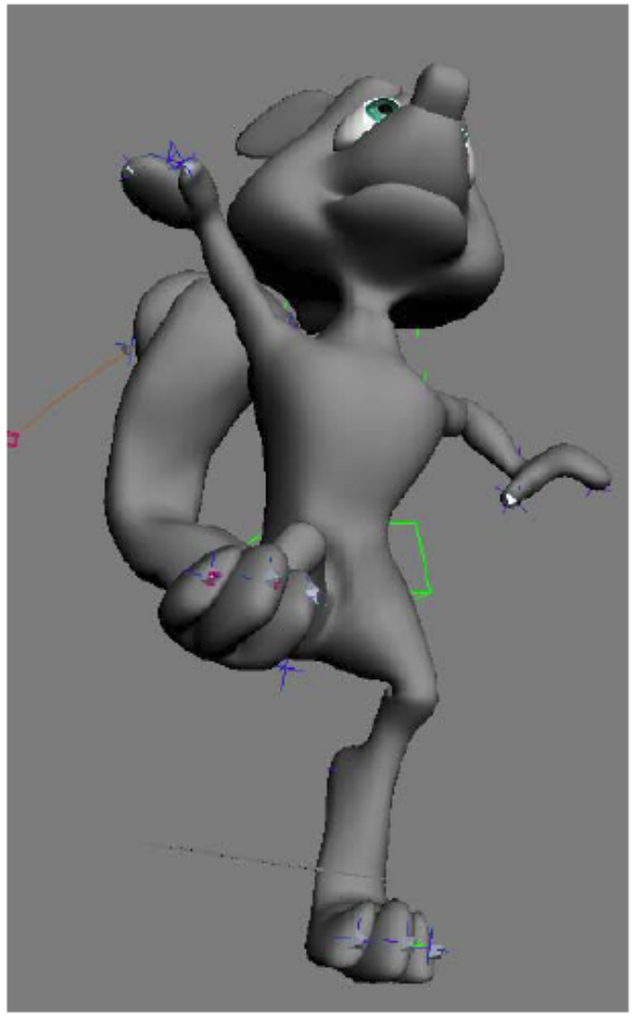
- Start with *modeling DOFs* (control points)
- *Deformations* control those DOFs at a higher level
  - Example: move first joint of second finger on left hand
- *Animation controls* control *those* DOFs at a higher level
  - Example: open/close left hand
- Both cases can be handled by the same kinds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

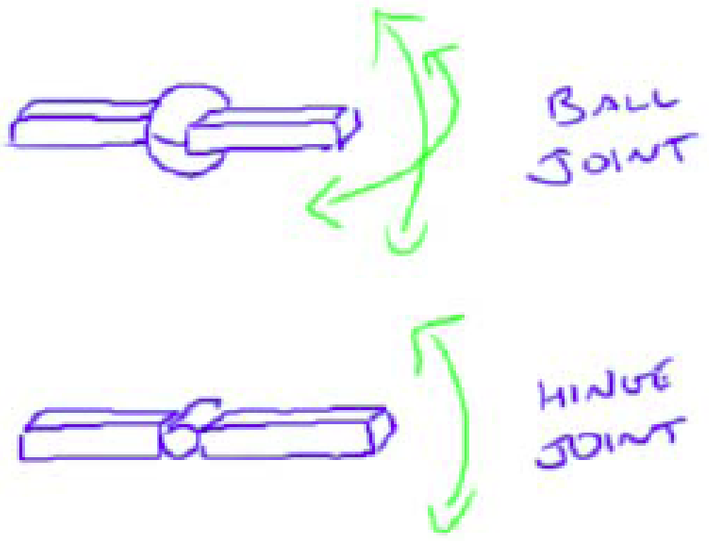
Character with DOFs



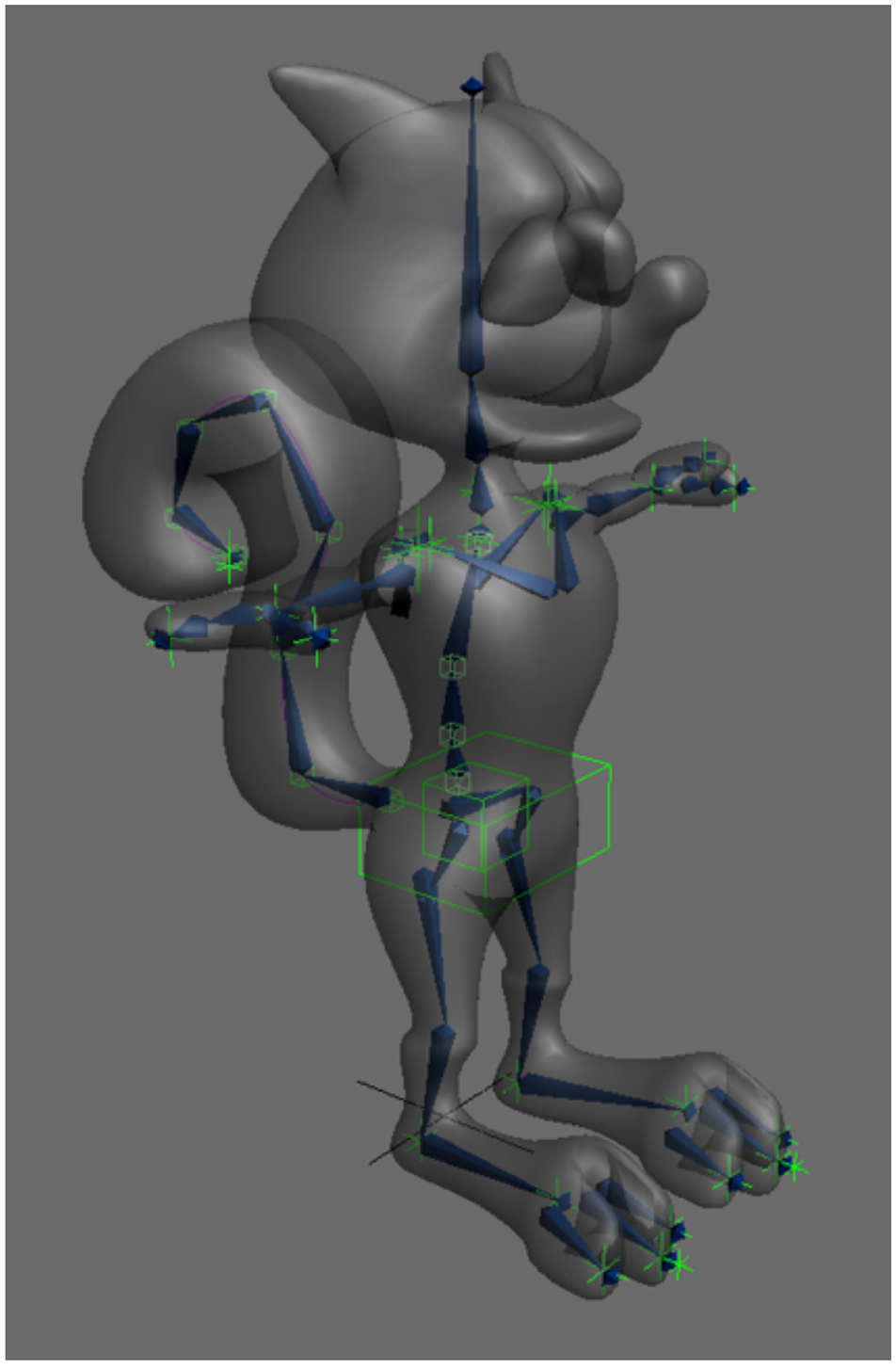
A visual description of the possible movements for the squirrel

[Greenberg/Pellacini | CIS 565]





Rigged character



[CIS 565 staff]

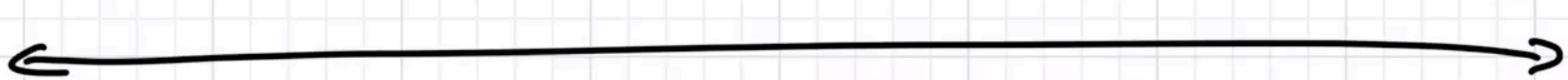
- Surface is deformed by a set of *bones*
- Bones are in turn controlled by a smaller set of *controls*
- The controls are useful, intuitive DOFs for an animator to use

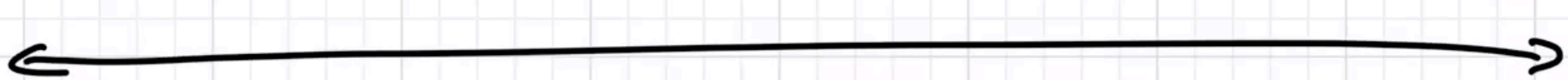
The most basic animation control

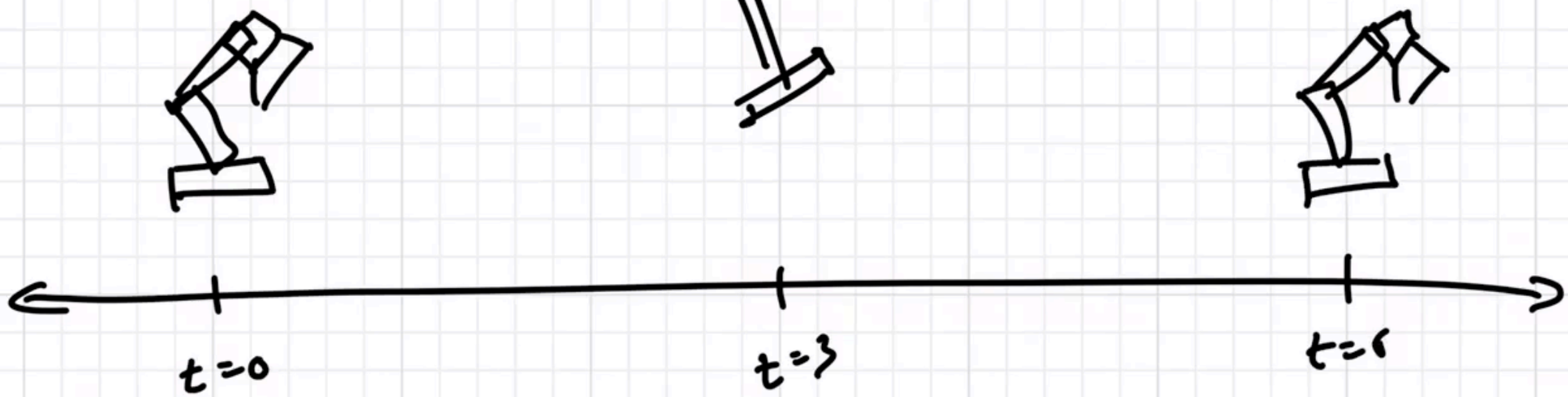
- Affine transformations position things in modeling
- Time-varying affine transformations move things around in animation
- A hierarchy of time-varying transformations is the main workhorse of animation
  - and the basic framework within which all the more sophisticated techniques are built

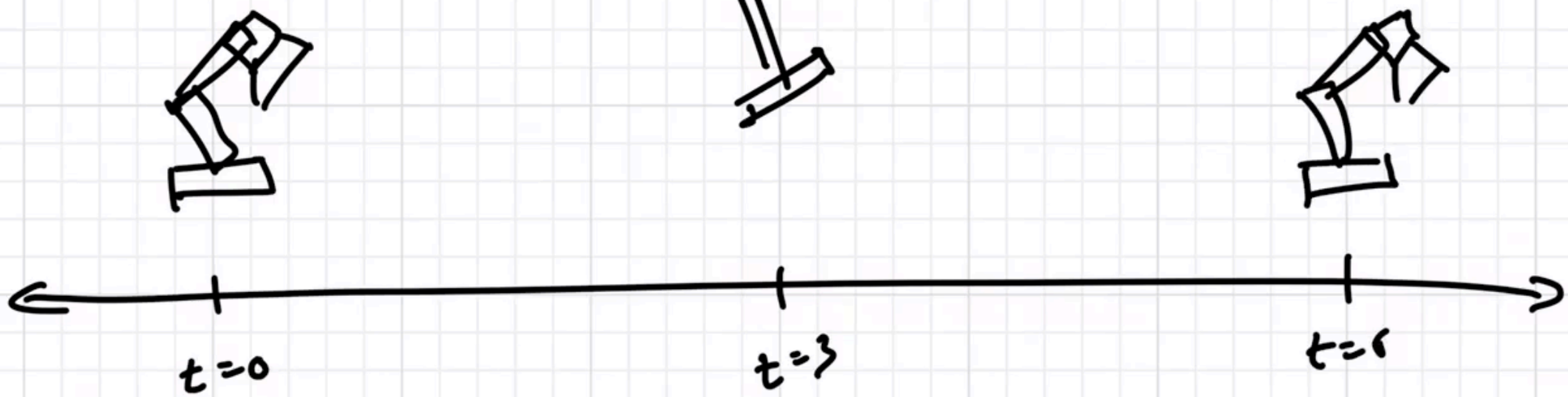

4.9

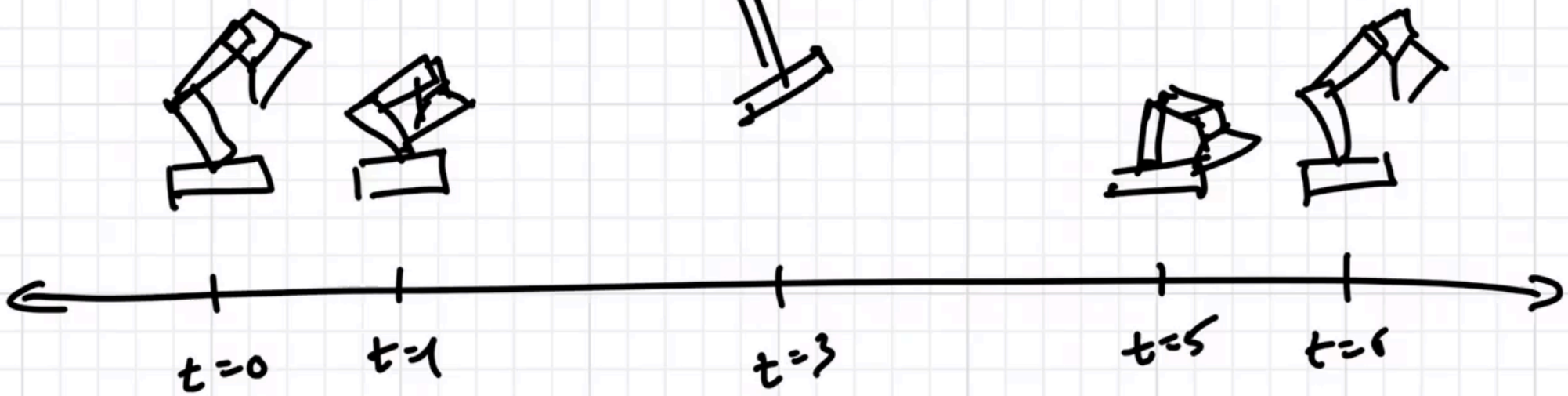

4.9

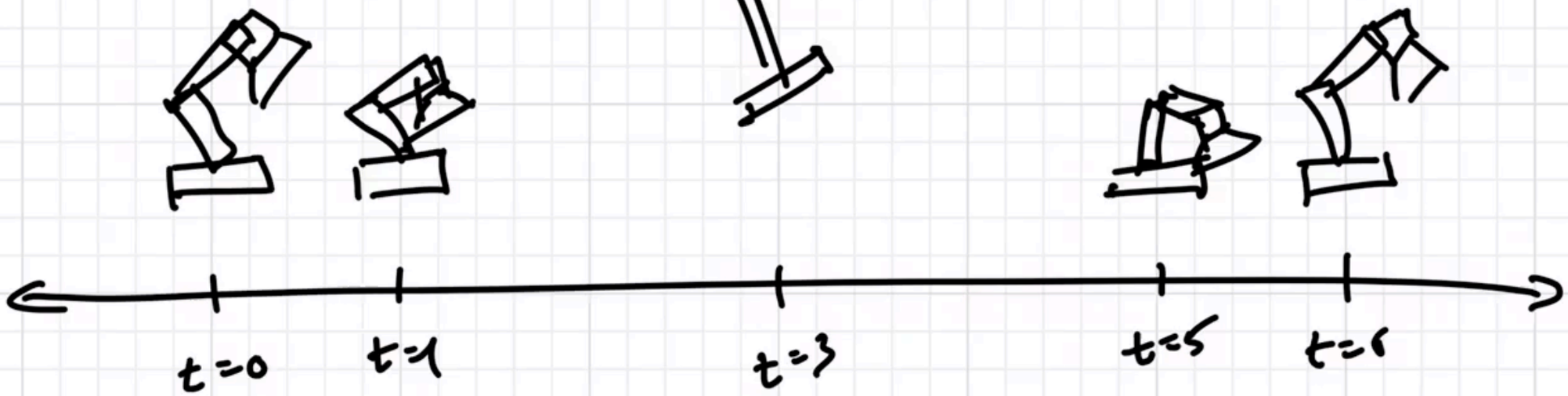


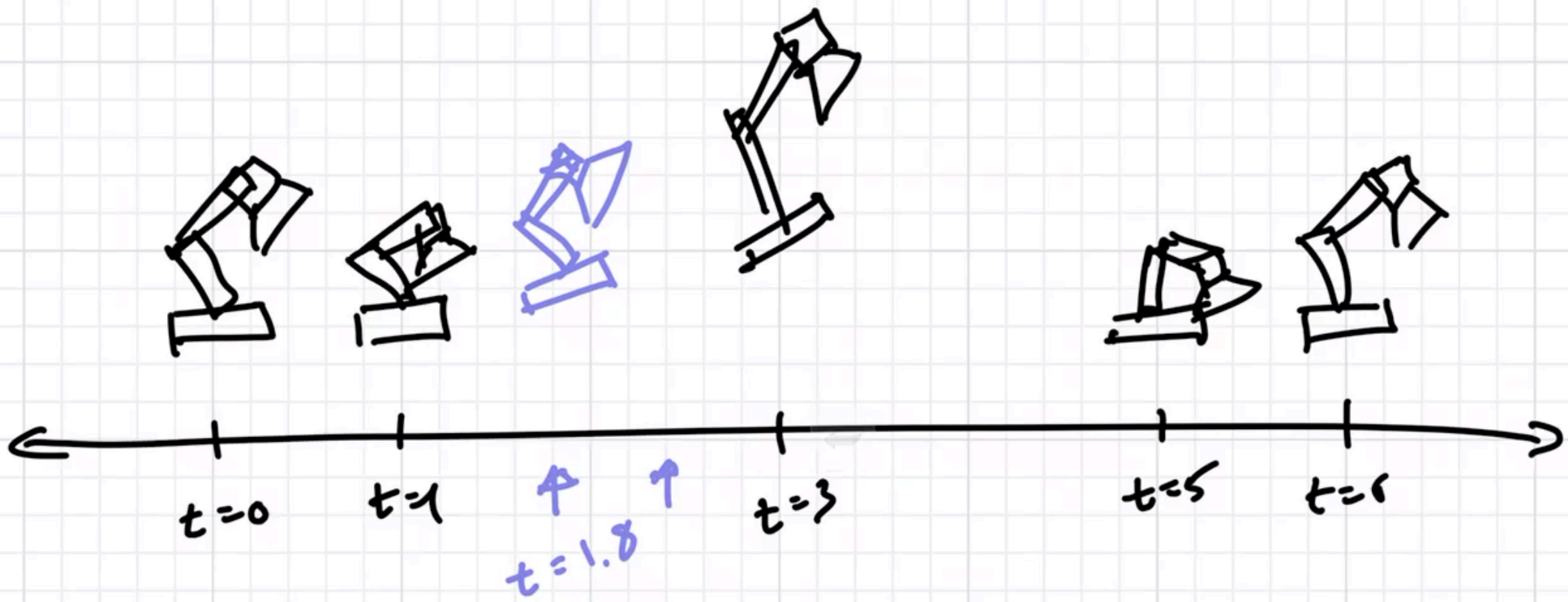


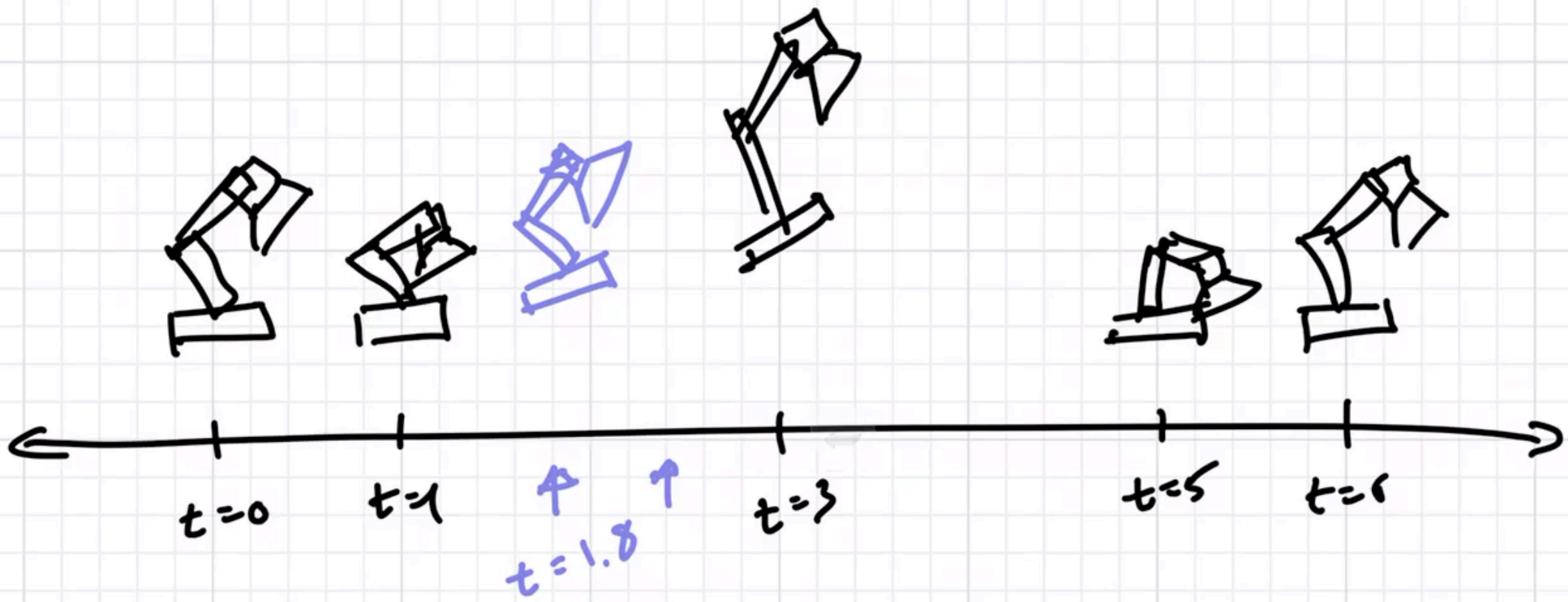

t=0
t=3
t=6

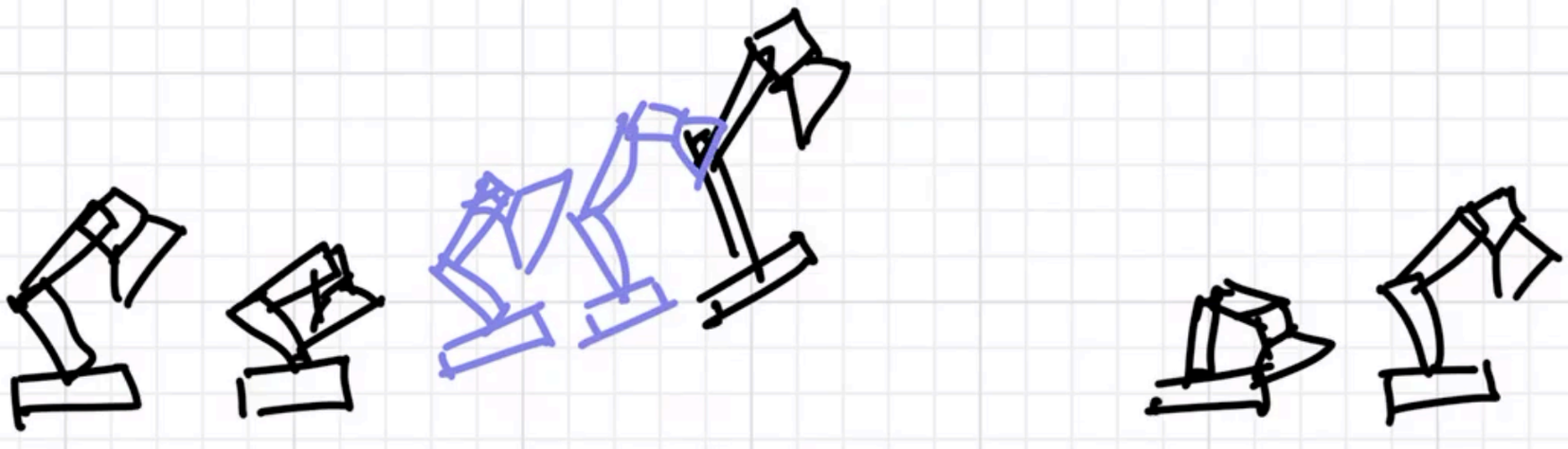

t=0
t=3
t=6

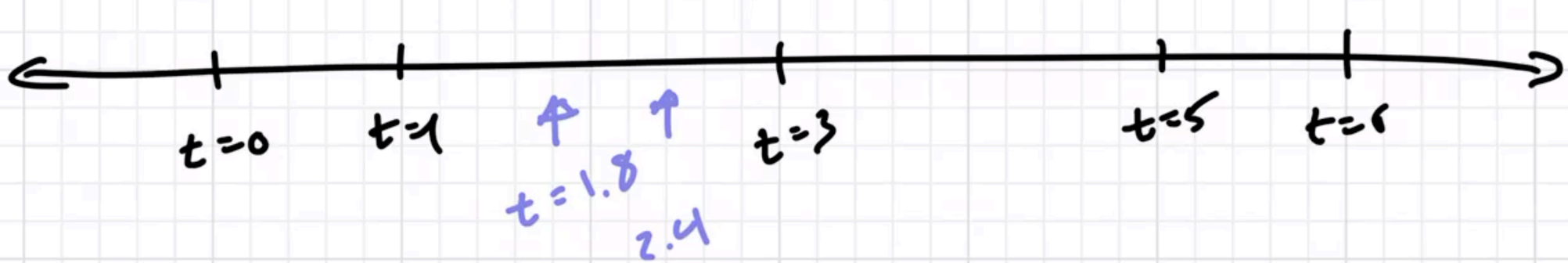

t=0
t=1
t=3
t=5
t=6

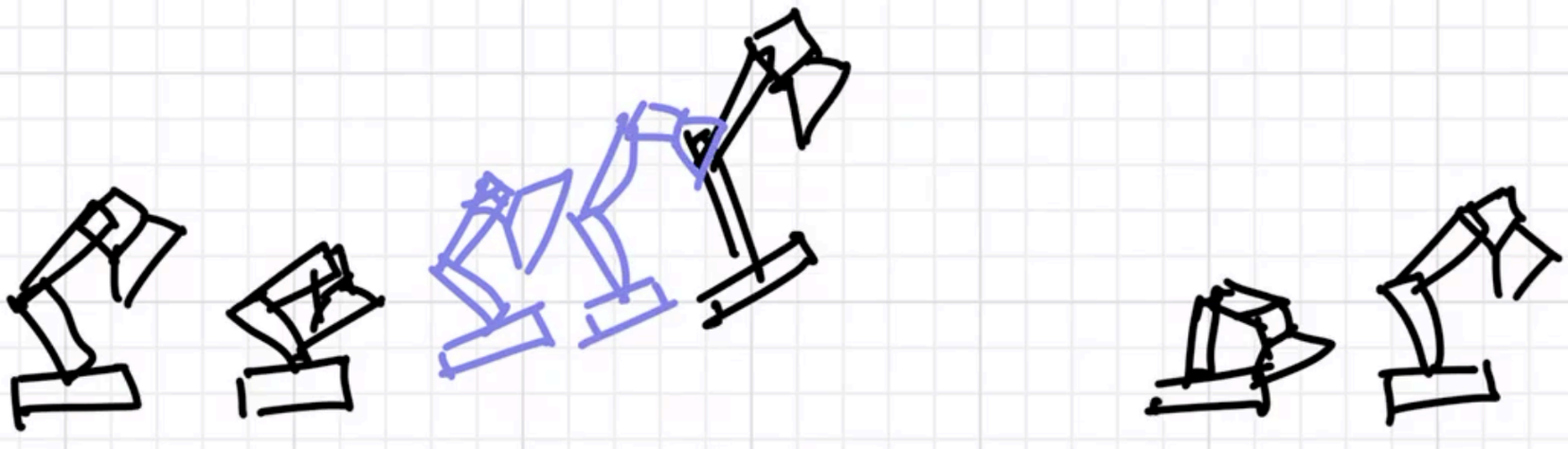

t=0
t=1
t=3
t=5
t=6

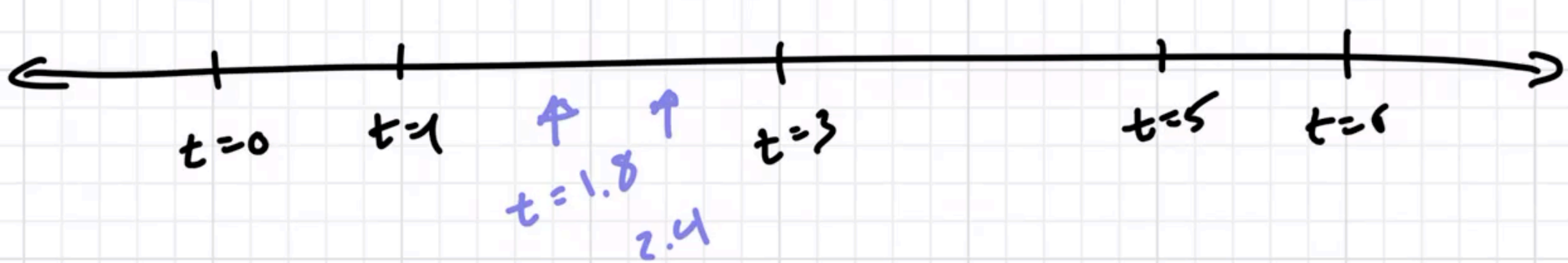

t=0
t=1
t=1.8
t=3
t=5
t=6


t=0
t=1
t=1.8
t=3
t=5
t=6




t=0
t=1
t=1.8
2.4
t=3
t=5
t=6

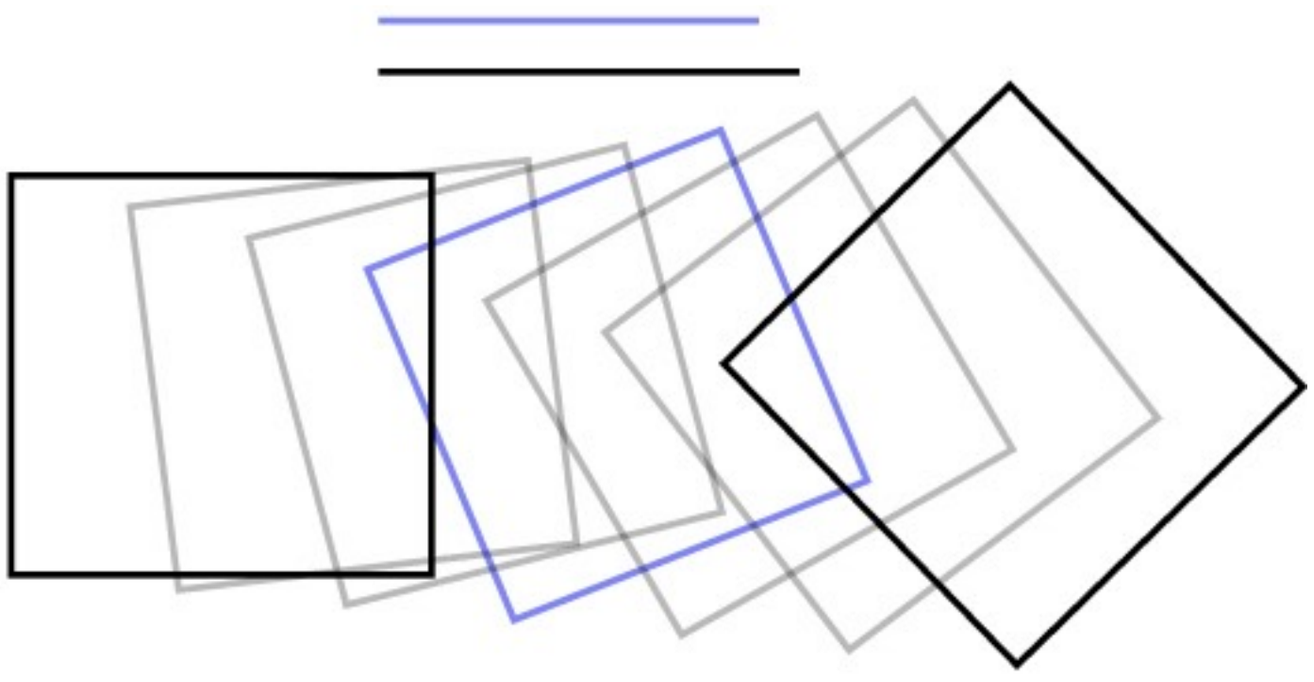



t=0
t=1
t=1.8
2.4
t=3
t=5
t=6

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?

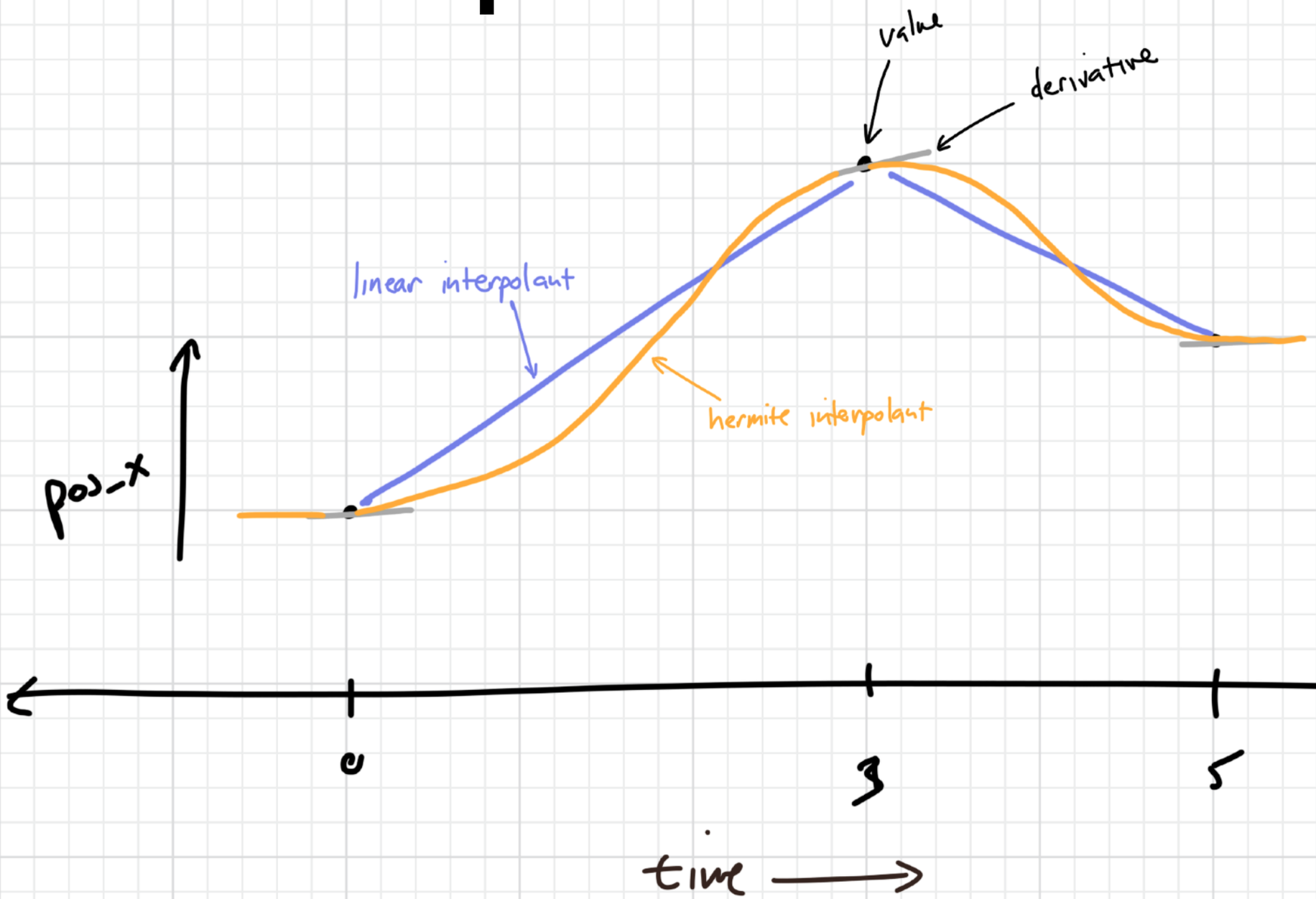
    this is fine for translations but bad for rotations



Interpolating transformations

- Linear interpolation of matrices is not effective
  - leads to shrinkage when interpolating rotations
- 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
- More abstractly: nodes expose scalar attributes to be interpolated
  - 2D: (x, y) translation; (x, y) scale; rotation angle θ
  - 3D: (x, y, z) translation; (x, y, z) scale; rotation quaternion (a, b, c, d) (more on 3D rotations soon…)
  - any other desired attributes to animate (color, etc.)

Smooth interpolation



Various techniques used; linear and Hermite splines are popular

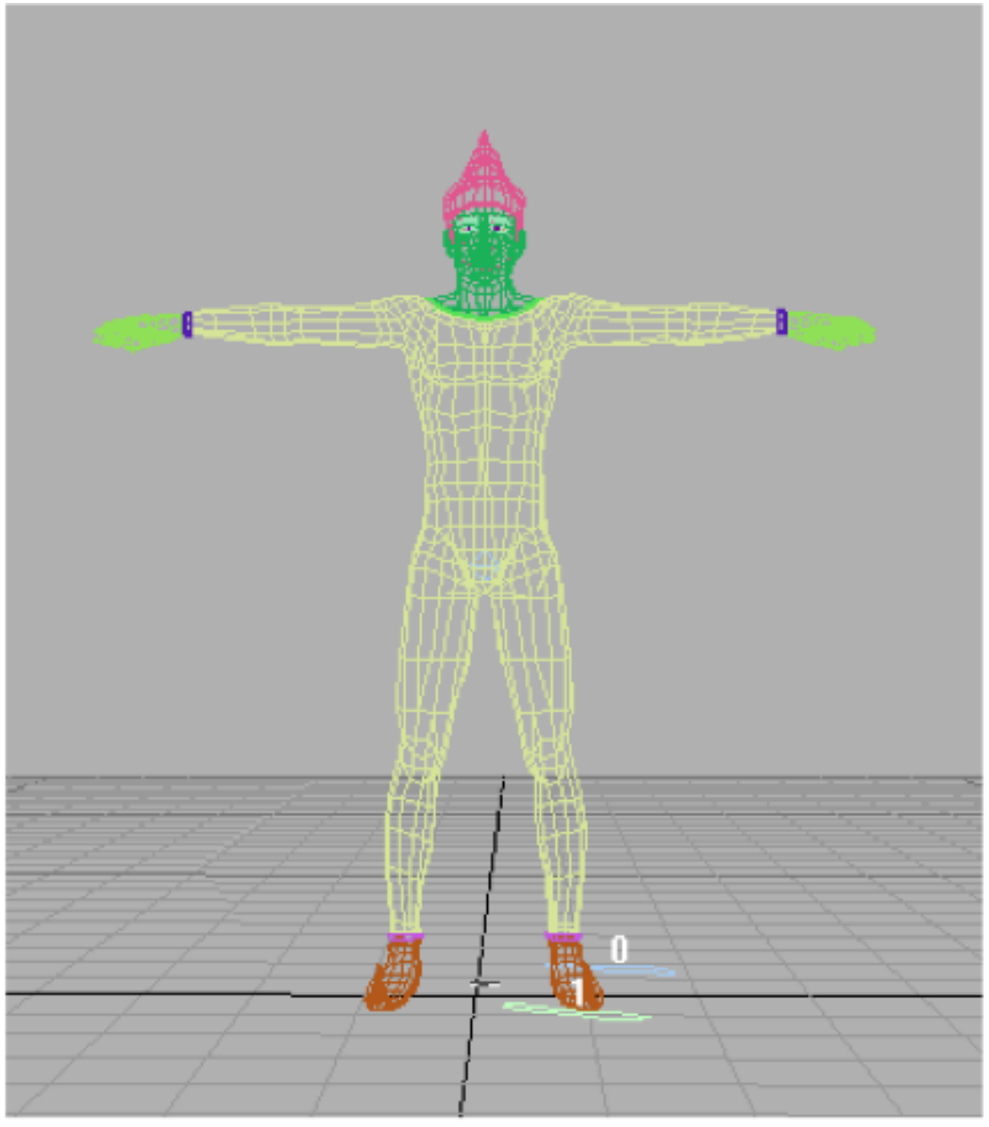
Deformation

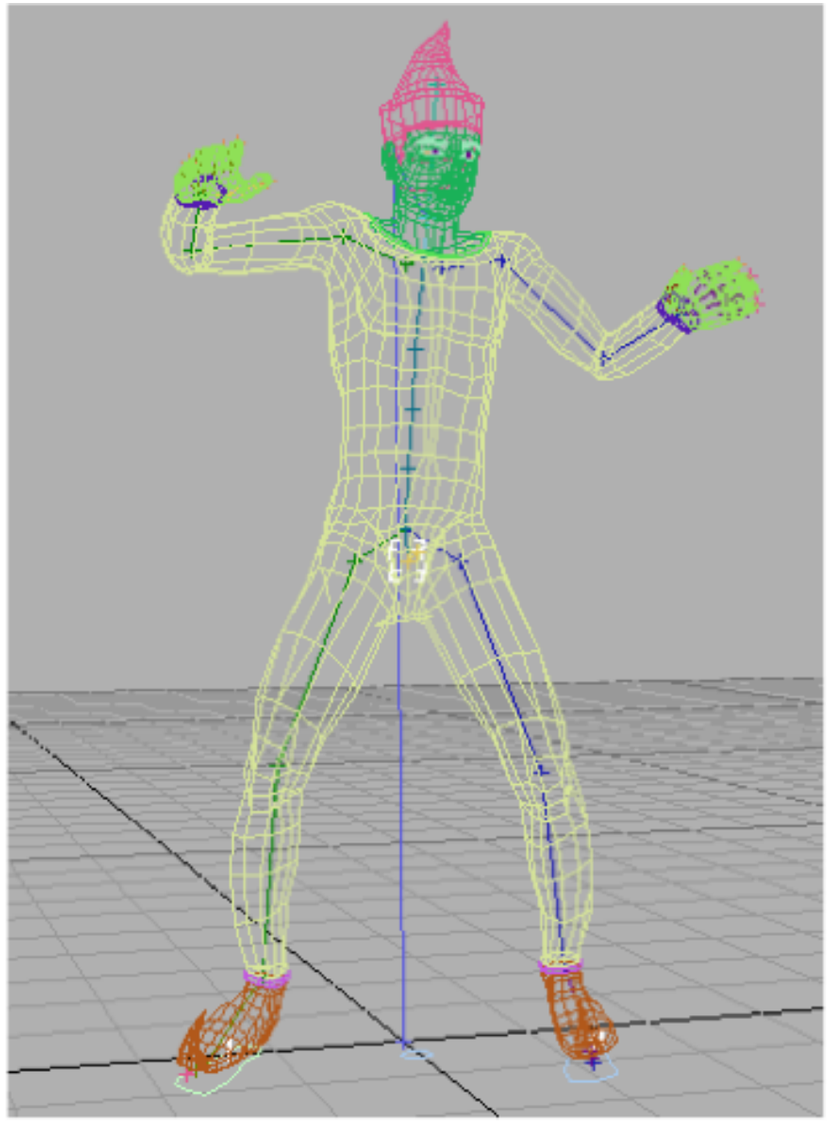
Basic surface deformation methods

- 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





[Sébastien Dominé | NVIDIA]

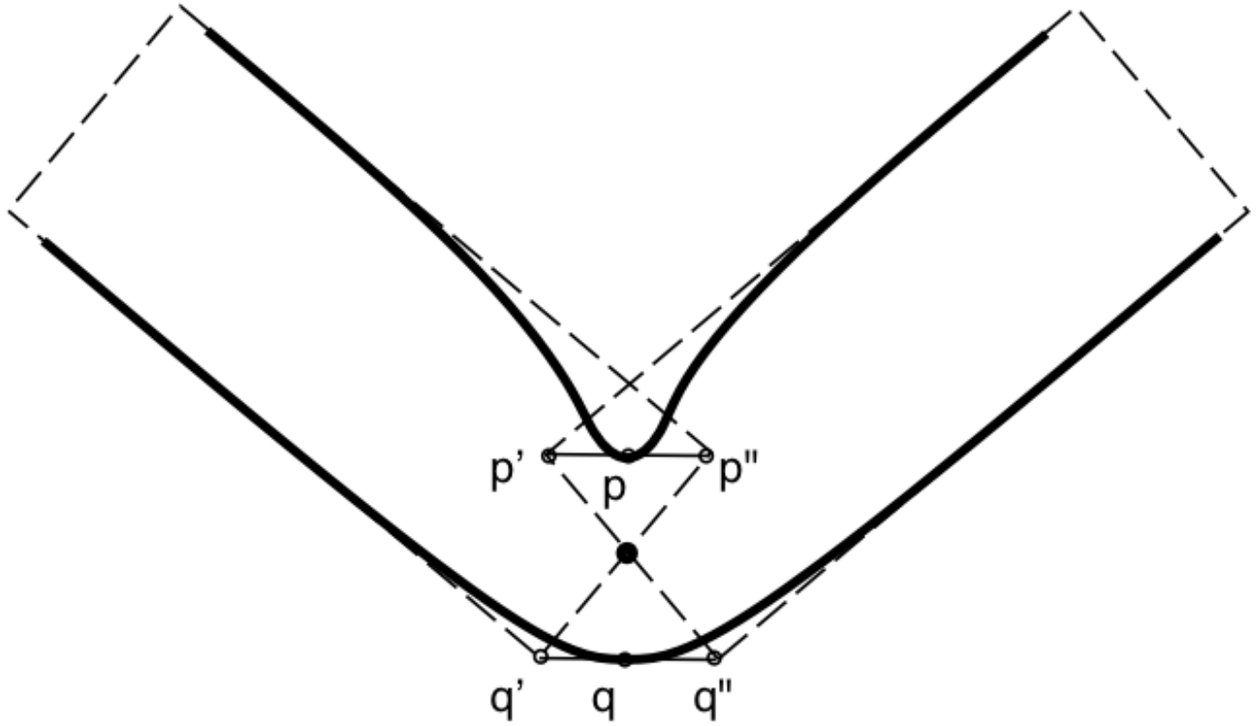
Mesh skinning math: setup

- Surface has control points $\mathbf{p}_i$
  - Triangle vertices, spline control points, subdiv 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

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



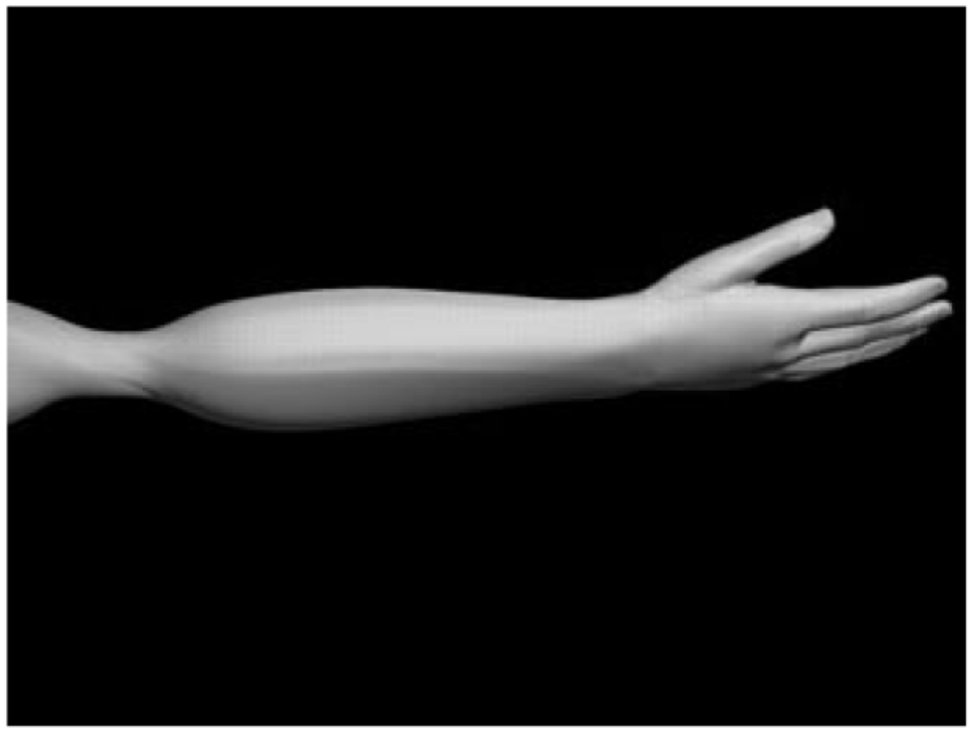
[Lewis et al. SIGGRAPH 2000]

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 problems

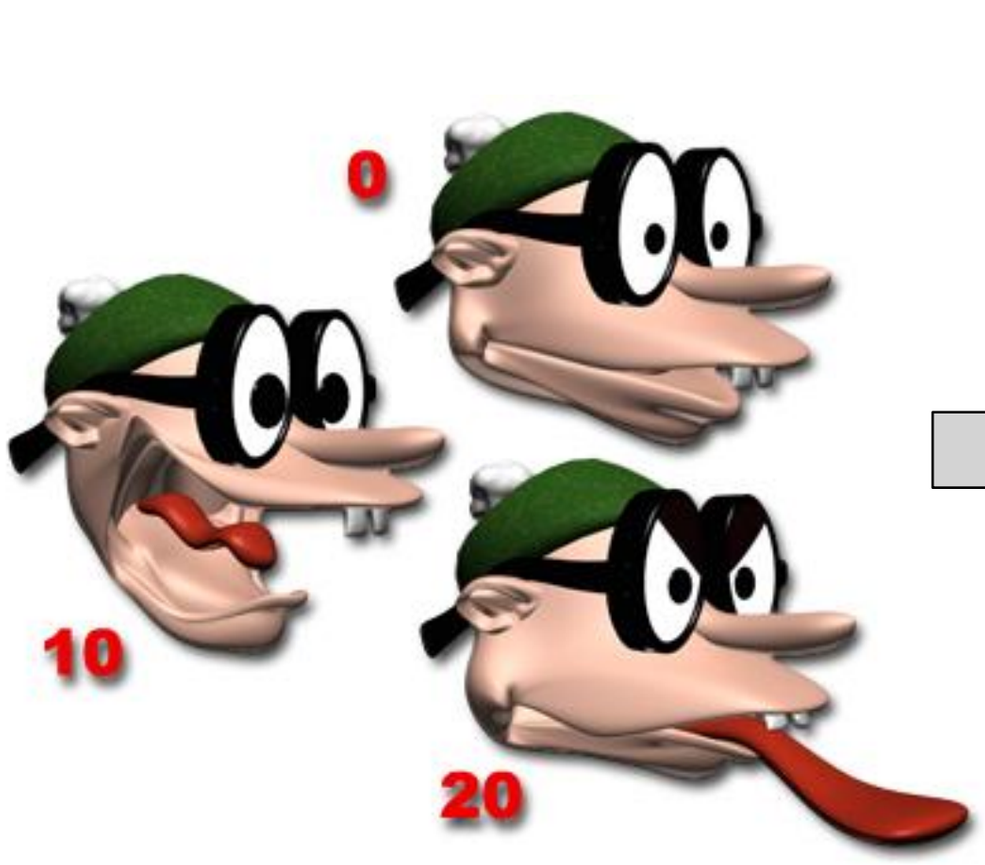
- Surface collapses on the inside of bends and in the presence of strong twists
  - Average of two rotations is not a rotation!
  - Add more bones to keep adjacent bones from being too different, or change the blending rules.

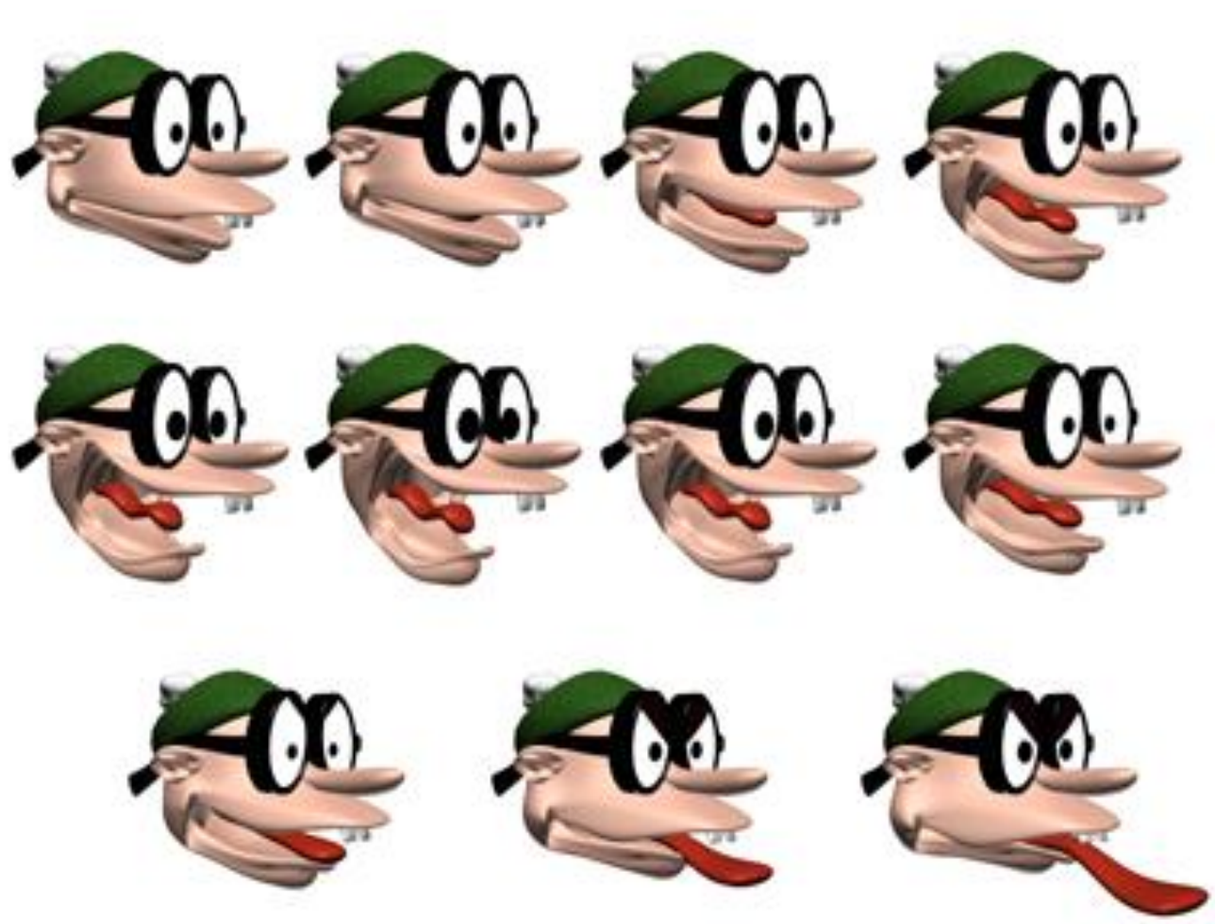


[Lewis et al. SIGGRAPH 2000]

Blend shapes

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





[3D Studio Max example]

Blend shapes math

- Simple setup
  - User provides key shapes—that is, a position for every control point in every shape: $\mathbf{p}_{ij}$ for point i, shape j
  - Per frame: user provides a weight w_j for each key shape

    Must sum to 1.0
- Computation of deformed shape

$$\mathbf{p}'_i = \sum_j w_j \mathbf{p}_{ij}$$

- Works well for relatively small motions
  - Often used for for facial animation
  - Runs in real time; popular for games