Animation

Overview
Traditional Animation
Keyframe Animation
Interpolating Rotation
Other Animation Techniques

COMPUTER GRAPHICS

15-462

23 Mar. 2000

A Unified View of Computer Animation

- **Models have parameters**
  - **Polygon positions, normals, spline control points, joint angles, camera parameters, lights, color**
  - ***n* parameters define an n-dimensional *state space***
  - **Values of *n* parameters = point in *state space***
- **Animation defined by path through state space**
  - **To produce animation:**
    - **1. start at beginning of state space path**
    - **2. set the parameters of your model**
    - **3. render the image**
    - **4. move to next point along state space path, repeat 2**
  - **Path usually defined by a set of motion curves**
    - **one for each parameter**
- **Every animation technique reduces to specifying the state space trajectory**

Animation vs. Modeling

- **Modeling and animation are tightly coupled**
  - **Modeling: what are the control knobs and what do they do?**
  - **Animation: how to vary them to generate desired motions?**
- **Building models that are easy to control is an important part of doing animation**
  - **Hierarchical modeling can help**
- **Where does modeling end and animation begin? Sometimes a very fuzzy distinction...**

Overvie

- **Animation techniques**
  - **Traditional animation (frame-by-frame)**
  - **Keyframing**
  - **Procedural**
  - **Behavioral**
  - **Performance based (motion capture)**
  - **Physically based (dynamics)**
- **Modeling issues**
  - **Rotations**
  - **Inverse kinematics**

Traditional Cel Animation

- Film runs at 24 frames per second (fps
  - That’s 1440 pictures to draw per minute
  - 1800 fpm for video
- Productions issues:
  - Need to stay organized for efficiency and cost reasons
  - Need to render the frames systematically
- Artistic issues:
  - How to create the desired look and mood while conveying story?
  - Artistic vision has to be converted into a sequence of still frames
  - Not enough to get the stills right--must look right at full speed
  - Hard to “see” the motion given the stills
    - » watch a video frame by frame - things can look very strange

---

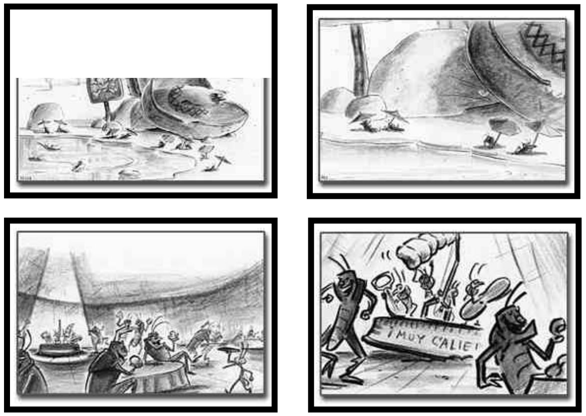
Traditional Animation: The Process

- Story board
  - Sequence of drawings with descriptions
  - Story-based description
- Key Frames
  - Draw a few important frames as line drawings
    - » e.g., beginning of stride, end of stride
  - Motion-based description
- Inbetweens
  - Draw the rest of the frames
  - People who draw these don't get paid much
- Painting
  - Redraw onto acetate *Cels*, color them in
  - These people get paid even less

Layered Motion

- It’s often useful to have multiple layers of animation
  - How to make an object move in front of a background?
  - Use one layer for background, one for object
  - Can have multiple animators working simultaneously on different layers
- Transparent acetate allows multiple *layers*
  - Draw each separately
  - Stack them together on a copy stand
  - Transfer onto film by taking a photograph of the stack

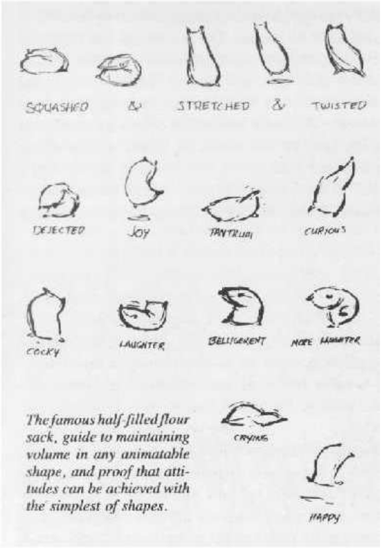
Story Boarding (from “A Bug’s Life”)



Principles of Traditional Animation
[Lasseter, SIGGRAPH 1987]

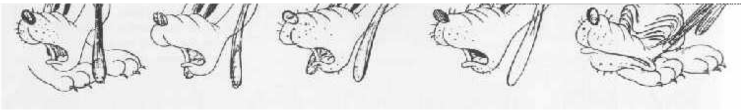
- **Stylistic conventions followed by Disney's animators and others *(but this is not the only interesting style, of course)***
- **From experience built up over many years**
  - **Squash and stretch -- use distortions to convey flexibility**
  - **Timing -- speed conveys mass, personality**
  - **Anticipation -- prepare the audience for an action**
  - **Followthrough and overlapping action -- continuity with next action**
  - **Slow in and out -- speed of transitions conveys subtleties**
  - **Arcs -- motion is usually curved**
  - **Exaggeration -- emphasize emotional content**
  - **Secondary Action -- motion occurring as a consequence**
  - **Appeal -- audience must enjoy watching it**

Principles of Traditional Animation

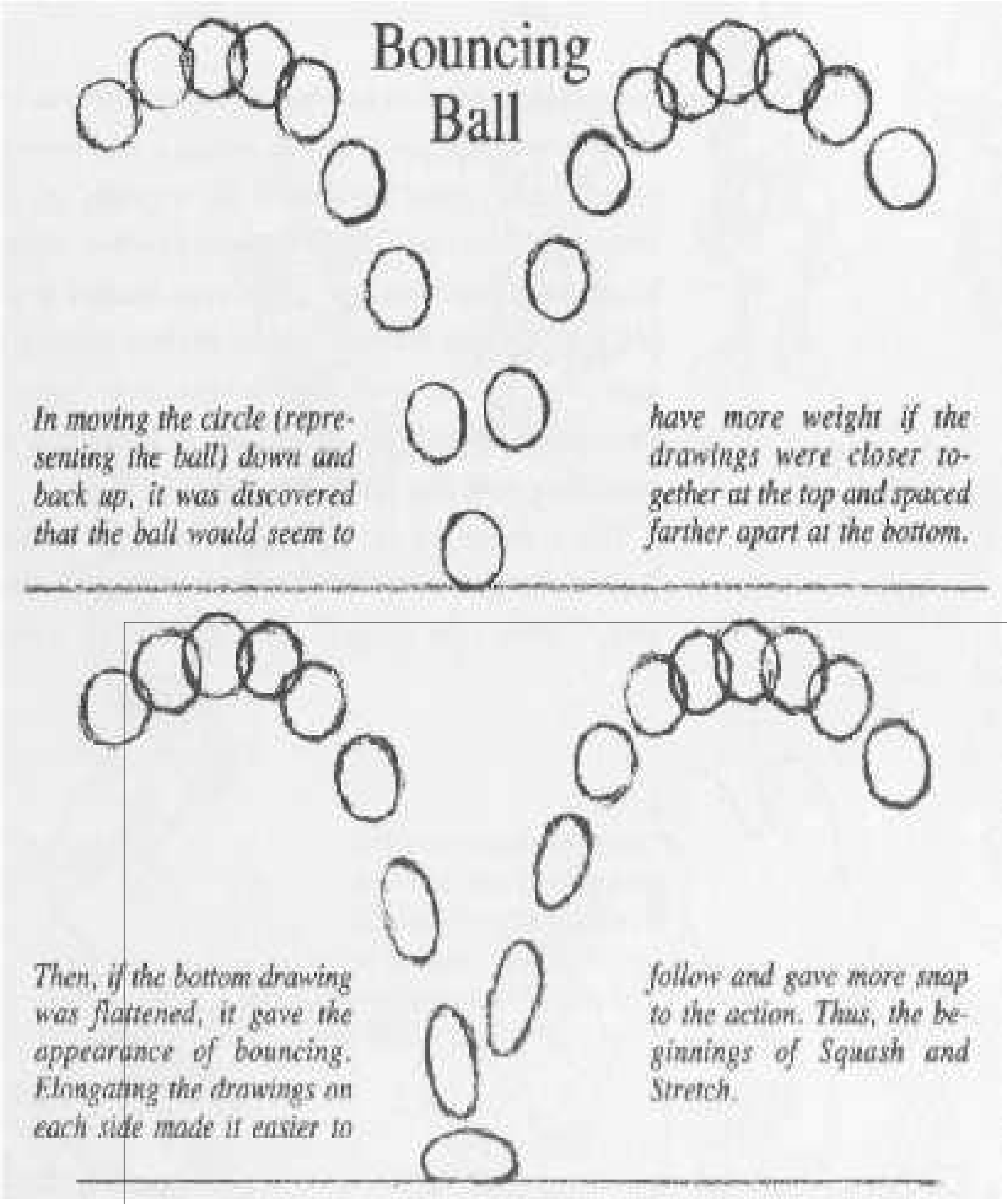


The famous half-filled flour sack, guide to maintaining volume in any animatable shape, and proof that attitudes can be achieved with the simplest of shapes.

Squash and Stretch



Squash and Stretch

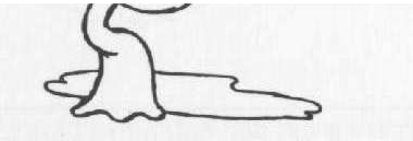


Bouncing Ball

In moving the circle (representing the ball) down and back up, it was discovered that the ball would seem to have more weight if the drawings were closer together at the top and spaced farther apart at the bottom.

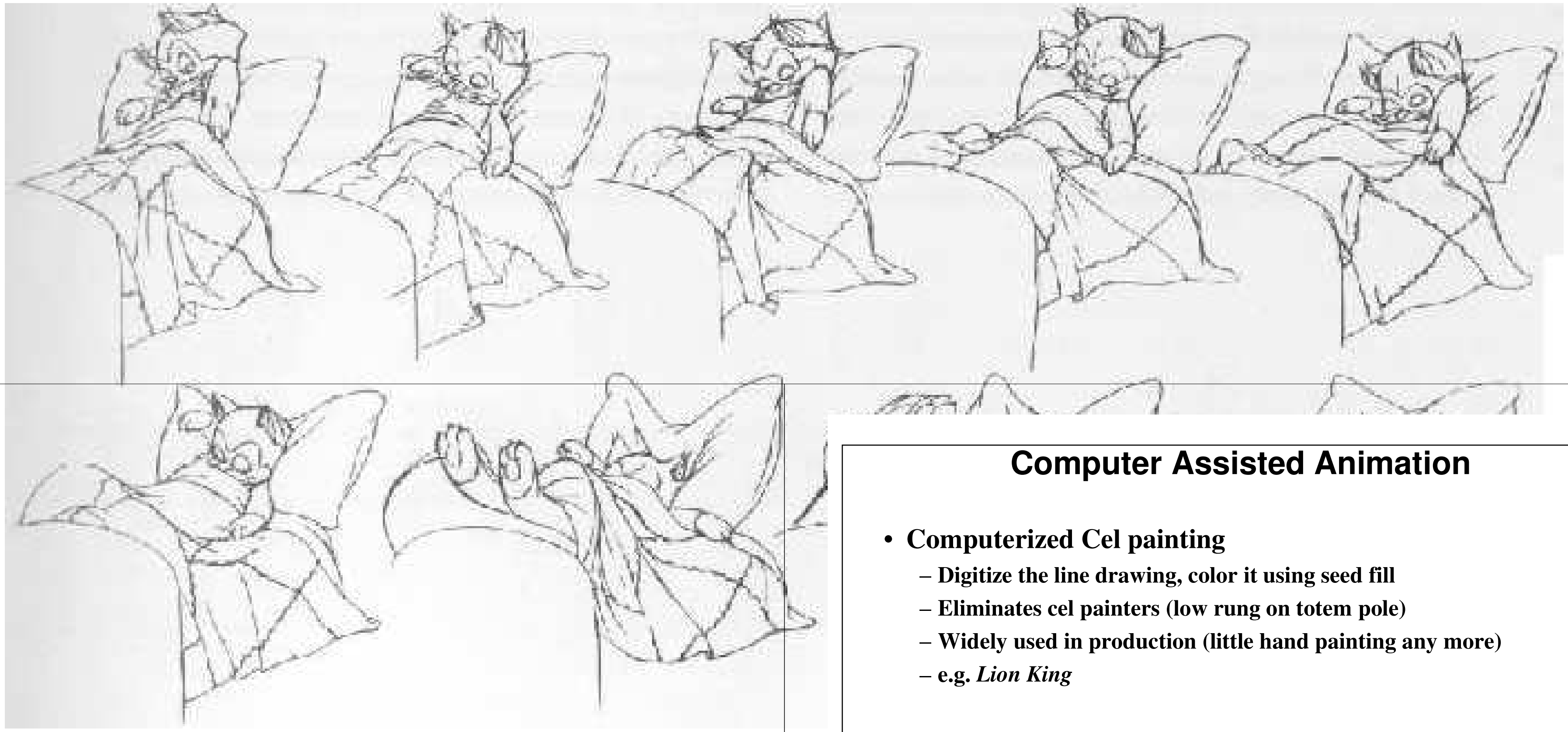
Then, if the bottom drawing was flattened, it gave the appearance of bouncing. Elongating the drawings on each side made it easier to follow and gave more snap to the action. Thus, the beginnings of Squash and Stretch.

Anticipation



Follow Through

Secondary Action



Computer Assisted Animation

- Computerized Cel painting
  - – Digitize the line drawing, color it using seed fill
  - – Eliminates cel painters (low rung on totem pole)
  - – Widely used in production (little hand painting any more)
  - – e.g. *Lion King*
- Cartoon Inbetweening
  - – Automatically interpolate between two drawings to produce inbetweens (*a la* morphing)
  - – Hard to get right
    - » inbetweens often don't look natural
    - » what are the parameters to interpolate? Not clear...
    - » not used very often

True Computer Animation

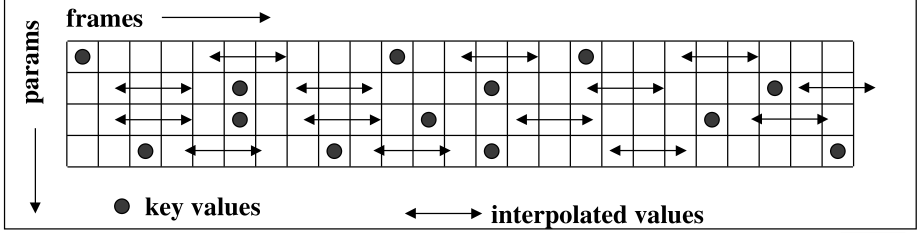
- Generate the images by rendering a 3-D model
- Vary the parameters to produce the animation
- Brute force
  - Manually set the parameters for each and every frame
  - For an *n* parameter model: *1440n* values per minute
- Traditional keyframing
  - Lead animators draw the important frames
  - Underpaid drones draw the inbetweens
- Computer keyframing
  - Lead animators create the important frames with 3-D computer models
  - Unpaid computers draw the inbetweens
  - The dominant production method

Interpolation

- Hard to interpolate hand-drawn keyframes
  - Computers don’t help much
- The situation is different in computer animation:
  - Each keyframe is a defined by a bunch of parameters (state)
  - Sequence of keyframes = points in high-dimensional state space
- Computer inbetweening interpolates these points
- How? You guessed it: splines

Keyframing Basics

- Despite the name, there aren’t really keyframes, *per se.*
- For each variable, specify its value at the “important” frames. Not all variables need agree about which frames are important.
- Hence, *key values* rather than key frames
- Create path for each parameter by interpolating key values



Keyframing: Issues

- What should the key values be?
- When should the key values occur?
- How can the key values be specified?
- How are the key values interpolated?
- What kinds of BAD THINGS can occur from interpolation?
  - Invalid configurations (pass through walls)
  - Unnatural motions (painful twists/bends)

Keyframe Animation: Production Issues

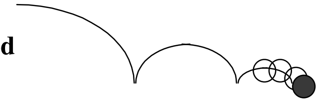
- How to learn the craft
  - apprentice to an animator
  - practice a lot
- Pixar starts with animators, teaches them computers

- Gives good control over motion
- Eliminates half the labor of traditional animation
  - But still very labor-intensive
- Impractical for complex scenes with everything moving

How Do You Interpolate Between Keys?

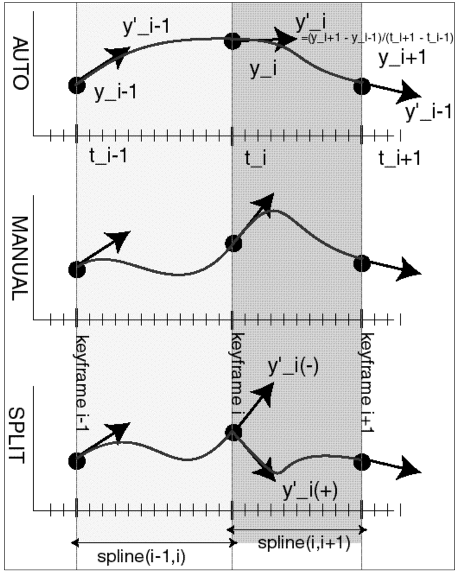
- Splines: non-uniform, C^1 is pretty good
- Velocity control is needed at the keyframes
- Classic example - a ball bouncing under gravity
  - zero vertical velocity at start
  - high downward velocity just before impact
  - lower upward velocity after
  - motion produced by fitting a smooth spline looks unnatural
- What kind of spline might we want to use?

Hermite is good



- What kind of continuity do we want?

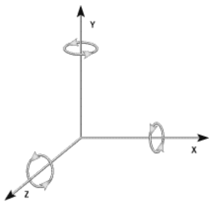
How Do You Interpolate Between Keys?



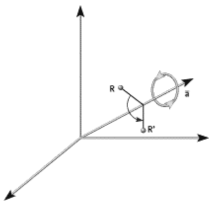
Problems with Interpolation

- Splines don’t always to the right thing
- Classic problems
  - Important constraints may break between keyframes
    - » feet sink through the floor
    - » hands pass through walls
  - 3D rotations
    - » Euler angles don’t always interpolate in a natural way
- Classic solutions:
  - More keyframes!
  - Quaternions help fix rotation problems

Interpolating Rotations



Euler angles



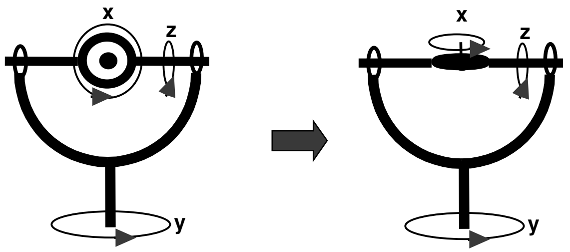
Axis-angle

Q: What kind of compound rotation do you get by successively turning about each of the 3 axes at a constant rate?

A: Not the one you want

- Good for single-axis rotations
- Awkward for other rotations

Gimbal Lock



Gimbal

Locked Gimbal

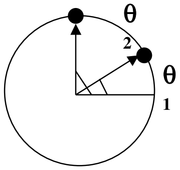
A *Gimbal* is a hardware implementation of Euler angles (used for mounting gyroscopes, expensive globes)

Gimbal lock is a basic problem with representing 3-D rotations using Euler angles

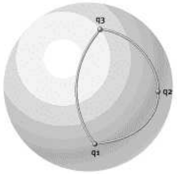
---

Quaternion Rotation

- We can think of rotations as lying on an n-D unit sphere



1-angle (θ) rotation
(unit circle)



2-angle (θ-ϕ) rotation
(unit sphere)

- Interpolating rotations means moving on n-D sphere
  - Can encode position on sphere by unit vector
  - SLERP: *Spherical Linear Interpolation*
    - » take shortest path between two points on unit sphere
  - How about 3-angle rotations?

Quaternion Rotation

- A quaternion is a 4-D unit vector q = [x y z w]
  - It lies on the unit hypersphere $x^2+y^2+z^2+w^2=1$
- For rotation about (unit) axis *v* by angle θ
  - vector part (sin θ/2) *v* = [x y z]
  - scalar part cos θ/2 = w
- The rotation matrix corresponding to a quaternion i

$$\begin{bmatrix} 1-2y^2-2z^2 & 2xy+2wz & 2xz-2wy \\ 2xy-2wz & 1-2x^2-2z^2 & 2yz+2wx \\ 2xz+2wy & 2yz-2wx & 1-2x^2-2y^2 \end{bmatrix}$$

- Quaternion Interpolation
  - represent rotation as quaternion
  - SLERP: linearly interpolate quaternions q1 and q2 and normalize
  - convert to rotation matrix to apply the rotation
- Only a unit quaternion encodes a rotation - normalize by dividing by $\sqrt{x^2+y^2+z^2+w^2}$

Where did this come from??

- These are the same old quaternions from math:

$$q = W + X\hat{i} + Y\hat{j} + Z\hat{k}$$

$$\hat{i}^2 = \hat{j}^2 = \hat{k}^2 = -1$$

$$\hat{i}\hat{j} = \hat{k} = -\hat{j}\hat{i}$$

- *W*, *X*, *Y*, and *Z* are real.
- Consider a normalized quaternion *q* and a non-normalized quaternion *v*.

$$v = a\hat{i} + b\hat{j} + c\hat{k}$$

$$q^{-1} = W - X\hat{i} - Y\hat{j} - Z\hat{k}$$

Where did this come from?? (2)

- Write out $v' = qvq^{-1}$ and expand it like a matrix equation for the *i*, *j*, and *k* terms.

$$\begin{pmatrix} a' \\ b' \\ c' \end{pmatrix} = \begin{pmatrix} 1-2(Y^2+Z^2) & 2(XY-ZW) & 2(XZ+YW) \\ 2(XY+ZW) & 1-2(X^2+Z^2) & 2(YZ-XW) \\ 2(XZ-YW) & 2(YZ+XW) & 1-2(X^2+Y^2) \end{pmatrix} \begin{pmatrix} a \\ b \\ c \end{pmatrix}$$

Where did this come from??? (3)

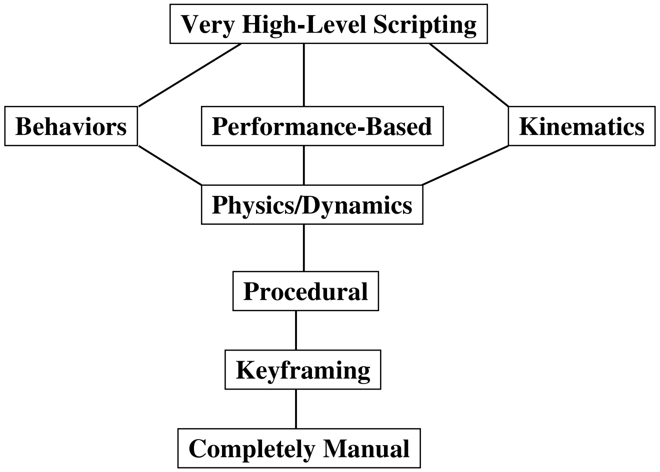
- This is the same rotation matrix we saw earlier
- It's orthogonal and has determinant 1, so it must be a pure rotation.
- It leaves the vector $(X,Y,Z)^T$ invariant, so it must be around that axis.
- To find the angle, consider a rotation purely about X and use the half-angle formulas from trig.
- Note that it's double-valued: two quaternions give the same rotation.
- The final result: a rotation about $(X,Y,Z)^T$ by an angle Θ can be represented as the quaternion:

$$q = \cos\frac{\theta}{2} + (X\hat{i} + Y\hat{j} + Z\hat{k})\sin\frac{\theta}{2}$$

Quaternion Interpolation

- Interpolating quaternions produces 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
- To interpolate more than two points:
  - solve a non-linear variationa constrained optimization (numerically)
- Further information: Ken Schoemake in the Siggraph '85 proceedings (*Computer Graphics,* V. 19, No. 3, P.245)

A Hierarchy of Animation Processes



Procedural Animation

- Define the motion using formulas
  - Hand-crafted
  - Physically based
- The animator must be a programmer
- Keyframing starts to become procedural as expressions are added
- At some level of complexity it becomes easier/more efficient than keyframing.

Dynamics

- Generate motion by specifying mass and force, apply physical laws (e.g., Newton's laws)
- Simulates physical phenomena
  - gravity
  - momentum (inertia)
  - collisions
  - friction
  - fluid flow (drag, turbulence, ...)
  - solidity, flexibility, elasticity
  - fracture

Dynamics – Simple Example

- Most differential equations cannot be solved symbolically (closed form), must be solved numerically.
- To numerically solve the differential equation F=ma, the simplest approach is Euler's method:

```
initialize x and dx (position and velocity)
choose some small time step dt
loop
    ddx = F()/m       // ddx=acceleration
    dx += ddx*dt      // dx=velocity
    x += dx*dt        // x=position
    t += dt
```

- F might be gravity, or any other forces as a function of time, position, or other state

Practical Physics/Dynamics

- This ranges from “Objects don’t go through walls” up to full fluid dynamics and finite element modeling

- Or, the animation can be generated with no realistic physics at all

- In general, you need to make things run fast
  - Quick convergence of iterative schemes
  - good approximate algorithms

- Viewers can be pretty tolerant of things like incorrect bounces

- This shift from accuracy to fast-and-looks-good is what distinguishes physically based CG from numerical science.

Behavioral Animation

- Define rules for the way an object behaves and interacts
  - models respond to their changing environment
  - programs implement the rules
- Classic example: “boids” (Craig Reynolds)
  - object’s motion is a simple function of nearby objects
  - emergent behavior: flocking
  - *Lion King* wildebeest stampede was choreographed this w
- Related
  - particle systems: usually really dumb behaviors
- Smarter entities
  - autonomous agents: hopefully really sophisticated behaviors

---

Performance-based Animation

- Record the animation from live action
  - simplest method - rotoscope (trace) over video of real motions
- Real time input devices
  - electronic puppeteering
- Motion capture
  - track motion of reference points
    - » body or face
    - » magnetic
    - » optical
    - » exoskeletons
  - convert to joint angles (not so straightforward)
  - use these angles to drive an articulated 3-D model
  - These motion paths can be *warped*
- New industry: CG cartoons

Kinematics & Inverse Kinematics

- Kinematics
  - gives motions in terms of joint angles, velocities, and positions
  - used by most keyframing and procedural animation systems
- *Inverse* kinematics
  - determine joint angles from positions
  - e.g. “calculate the shoulder, elbow, and wrist rotation parameters in order to put the hand here”
  - better for interaction
  - sometimes underdetermined (i.e. many combinations of joint angles to achieve a given end result)
  - used a lot in robotics
- Constrained Optimization
  - Get from A to B doing the minimum possible work

“High-level” Animation

- **Main idea: piece together complex animations from a library of component motions**

- **Scripting languages**
  - **describe the events**
  - **describe their sequence**

- **“Task” level animation**
  - ***go to the kitchen for a coke, drink it, walk the dog***
  - **fine in principle, but we don’t know how to do it yet**

- **Blends into AI and behavioral animation**