Behaviors (2 lectures):

low level
  keyframing
  motion capture
  simulation

high level (AI)
  finite state machines
  path planning
  group behaviors
Generating Motion

What Matters?

quality of motion appropriate for rendering style and frame rate
controllable from the UI
controllable from the AI
skills
personality
Keyframing

fine level of control

quality of motion depends on skill of animator
Motion Capture

natural-looking motion hard to generalize motions registration is difficult

images courtesy of the Microsoft Motion Capture Group
Simulation (broadly defined)

physics is hard
pseudo-physics is somewhat hard
control is very hard
genralization/interactivity

user/AI

desired behavior

control

forces and torques

model

graphics

numerical integrator

state
When to Use What Method?

keyframing

motion capture

simulation
When to Use What Method?

**keyframing**
- sprites and other simple animations
- non–human characters

**motion capture**
- human figures
- subtle motions, long moves

**simulation**
- passive simulations
- when interactivity is really important
Hand Drawn Animation: 2D

sketches
pencil tests
inking
coloring
digitize to sprites
Computer Animation: 2D or 3D

- sketches
- models and materials
- key configurations
- playback of motion
- or render to sprites

Improv, Perlin, NYU
Keyframing

iterate: adjust trajectory
play back motion

parameters:
locations
joint angles
shape -- flexible objects
material properties (color, texture)
camera motion (for animation)
lighting
Keyframing -- Interpolation

Inbetweening

Linear

Ease in/ Ease out
Spline-driven Animation

\[ x, y = Q(u) \text{ for } u: [0, 1] \]

equal arc lengths

equal spacing in \( u \)
Arc-length reparametrization

$s = A(u)$ where $s$ is arc length
reparam: $Q(u)$ to $Q(A^{-1}(s))$
need to find $u = A^{-1}(s)$

bisection search for a value of $u$ where $A(u) = s$ with a numerical evaluation of $A(u)$ (details in Watt and Watt)
Keyframing -- Constraints

Inverse Kinematics
Joint Limits
Position Limits
Kinematics -- the study of motion without regard to the forces that cause it.

Forward: \[ A = f(\alpha, \beta) \]

Inverse: \[ \alpha, \beta = f^{-1}(A) \]

draw graphics

specify fewer degrees of freedom

more intuitive control of dof

pull on hand

glue feet to the ground
Forward Kinematics

\[ x = L_1 \cos \theta_1 + L_2 \cos (\theta_1 + \theta_2) \]

\[ y = L_1 \sin \theta_1 + L_2 \sin (\theta_1 + \theta_2) \]

\[
\begin{bmatrix}
x \\
y \\
z \\
1
\end{bmatrix}
= \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}
\]

\[
\begin{bmatrix}
x \\
y \\
z \\
1
\end{bmatrix}
= \begin{bmatrix} \text{rot } \theta_1 \\ \text{trans } L_1 \\ \text{rot } \theta_2 \\ \text{trans } L_2 \end{bmatrix}
\]
Coordinate Systems

Figure 1. As animation progresses, the root moves away from the origin.

Figure 3. Zed in an animated jump. Z-translation is controlled by the animation. The Base is still on the floor.
Inverse Kinematics

\[ \theta_2 = \cos \left( \frac{x^2 + y^2 - L_1^2 - L_2^2}{2 L_1 L_2} \right) \]

\[ \theta_1 = \frac{-(L_2 \sin \theta_2)x + (L_1 + L_2 \cos \theta_2)y}{(L_2 \sin \theta_2)y + (L_1 + L_2 \cos \theta_2)x} \]

\[ \theta = f^{-1}(x) \]
What makes IK hard?

many dof—non-linear, transcendental equations
redundancies

choose solution that is "closest" to current configuration
move outermost links the most
energy minimization
minimum time

singularities

ill-conditioned near singularities
high state space velocities for
low cartesian velocities

goal of "natural looking" motion
minimum jerk
Motion Capture

What do we need to know?
- x, y, z
- pitch, roll, yaw

Errors cause
- joints to come apart
- links to grow/shrink
- bad contact points

Sampling Rate and Accuracy
Motion Capture

Goals
realistic motion
lots of different motions (300–1000)
contact

Appropriate game genres
sports
fighting
Applications
movies
tv shows
video games
performance animation
Production Pipeline

Calibration

Capture -> Skeleton Estimation

IK Processing

Ok? yes

Similar Analysis

Technology Calibration
Skeleton Estimation

Remaining Issues: modifying and controlling
Production Pipeline

Various phases of the motion capture process (Bodenheimer et al., Fig. 11)
Plan out Shoots Carefully

- know needed actions (80–100 takes/day)
- bridges between actions
- speed of actions
- starting/ending positions

- hire the right actor
- watch for idiosyncracies in motion
- good match in proportions

- marker/sensor placement
- capture enough information
- watch for marker movement

- check data part way through shoot
- videotape everything
Superguy's flowchart:

- Punch F from S
- Punch R from S
- Punch L from S

- Kick F from S
- Kick R from S
- Kick L from S

- Shoot F from S
- Shoot R from S
- Shoot L from S

- Get hit F from S
- Get hit R from S
- Get hit L from S

- Punch F from C
- Punch R from C
- Punch L from C

- Kick F from C
- Kick R from C
- Kick L from C

- Shoot F from C
- Shoot R from C
- Shoot L from C

- Get hit F from C
- Get hit R from C
- Get hit L from C

- Special Move #1
- Special Move #2
- Defend from S

- Stand
- Walk
- Run
- Get up to S
- Fall from S
- Lie Down
- Dead
- Crouch
Technology

**passive reflection**—Peak
hand or semi–automatically digitized
time consuming

no glossy or reflective materials
tight clothing
occlusion of markers by props
higher frames/second
Technology

passive reflection—Acclaim, Motion Analysis, ...
automatically digitized
240 Hz
not real-time
3+ markers/body part for 6 dof
2+ cameras for 3D position data
Technology

active light sources -- Optotrak

automatically digitized correspondence
256 markers
3,500 markers/second
Technology
electromechanical transducers
  Accension flock of birds
  Polhemus Fastrak

limited range/resolution
pigtail (new wireless system)
metal in the environment
  (treadmill, rebar!)
no identification problem
6 dof information
realtime
lower frequency: 30 to 144 Hz
few markers: \(\sim 13−18\)
Technology
exoskeleton + angle sensors
Analogous

pigtail
no identification problem
realtime
high frequency: 500Hz
not range limited
fit
rigid body approximation
Technology

mechanical motion capture

data glove
  low accuracy
  focused resolution

monkey
  high accuracy
  high data rate
  not realistic motion
  no paid actor
Technology Issues:
- resolution/range of motion
- calibration
- accuracy
- occlusion/correspondence

Animation Issues:
- style
- scaling
- generalization
Resolution

positioning of camera
Marker Placement

location should move rigidly with joint
stay away from bulging muscles, loose skin
shoulders: skeletal motion not closely tied to motion on skin

Calibration
zero position
fine calibration by hand
Finding Joint Locations

move markers to joint centers

assume rigid links, rotary joints

shoulder?
Extract Best Limb Lengths

use estimator to compute limb length

minimize or reject outliers
IK for Joint Angles

non-linear optimization
joint angles should be smooth
to allow resampling
minimize deviation between recorded data and model

\[ F(\theta) = \text{sum } (w_p(P-P')^2 + w_o(O-O')^2 + w_c c^2) \]
Accuracy

marker movement
sensor noise
skew in measurement time
data recording rate

filtering (requires high data rate)
Camera Calibration

**internal camera parameters**
- optical distortion of lens

**external parameters**
- position and orientation

**correlation between multiple cameras**
Model-based Techniques

restricted search space for markers

dynamics (velocity integration)
model of behavior
model of bodies for occlusion
Animation Issues: scaling

contact

movement style

inverse kinematics
Animation Issues: generalization

Interpolation Synthesis for Articulated Figure Motion

Wiley and Hahn
Vrais ’96

Initial Data  Resampled Data
Animation Issues: generalization

Motion Warping
Witkin and Popovic, Siggraph ’95

keyframes as constraints in a smooth deformation

keyframe placing the ball on the racket at impact
Animation Issues: generalization
Motion Editing with Spacetime Constraints
Gleicher
1997 Symposium on Interactive 3D Graphics
Animation Issues: blending

Efficient Generation of Motion Transitions using Spacetime Constraints
Rose, Guenter, Bodenheimer, Cohen
Siggraph ‘96
Simulation

modeling the real world with (simple) physics

realism

a set of rules

better interactivity

Objects or Characters?
Passive—no muscles or motors

Active—internal source of energy

Particle systems
leaves
water spray
Clothing

Running human
trotting dog
Swimming fish

User

Initial conditions

Model

State

Numerical integrator

Graphics

Desired behavior

Control

Forces and torques

Model

State

Numerical integrator

Graphics
Equations of Motion:

- water
- explosions
- rigid body models

Control Systems:

- wide variety of behaviors
- transitions between behaviors
- controllable by AI or UI
- robust
Equations of Motion:

\[ A = g \]

\[ V' = V + A \Delta t \]

\[ P' = P + \frac{V + V'}{2} \Delta t \]
Integrating in a 2d world

object.x += 2
object.xd = 2 pixels/timestep = 60p/s
timestep = 1/30fps

Pool Game

vertical wall:
xd = −xd

horizontal wall:
yd = −yd
Collision Detection

essential for many games
shooting
kicking
car crashes

expensive —– $n^2$ tests
Efficiency Hacks/Cheats

fewer tests—exploit spatial coherence
use bounding boxes/spheres
hierarchies of bounding boxes
Bounding Boxes

axis-aligned vs object-aligned

axis-aligned change as object moves approximate by rotating bbox

swept volume
Collision Detection

- convex objects
- look for separating plane
- test all faces
- test all edge from obj 1/vertex from obj2 pairs
- save separating plane for next iteration
Collision Detection

concave objects
break apart
convex hull
automatic or artist-created
Efficiency Hacks/Cheats

cheaper tests -- exploit temporal coherence
Efficiency Hacks/Cheats

\[ d = \sqrt{((x_1 - x_2)^2 - (y_1 - y_2)^2)} \]

cheaper distance calculation:

compare against \( d^2 \)

approximate calculation:

\[ d' = \text{abs}(x_1 - x_2) + \text{abs}(y_1 - y_2) \]

\[ - \min(\text{abs}(x_1 - x_2), \text{abs}(y_1 - y_2))/2 \]

Manhattan distance – shortest side

\[ x=3, \ y=4 \Rightarrow d = 5 \]

\[ d' = 3 + 4 - 1.5 = 5.5 \]
Collision detection: sprites

AND for each pixel in sprites
Integration of Technologies

- layering
- add hand/finger motion later
- facial animation
- use keyframing to modify data
- fix holes in data
- use motion capture data to drive simulation
The Jacobian

\[ f(\theta) = x \]

\( x \) is of dimension \( n \) (generally 6)
\( \theta \) is of dimension \( m \) (# of dof)

Jacobian is the \( n \times m \) matrix relating differential changes of \( \theta \) (\( d\theta \)) to differential changes of \( x \) (\( dx \))

\[ J(\theta) \, d\theta = dx \]

where the \( ij \)th element of \( J \) is

\[ J_{ij} = \frac{\delta f_i}{\delta x_j} \]

Jacobian maps velocities in joint angle space to velocities in cartesian space
IK and the Jacobian

\[ \theta = f^{-1}(x) \]
\[ dx = J \, d\theta \]
\[ d\theta = J^{-1} \, dx \]

Inverting the Jacobian

J is n x m— not square in general
compute pseudo–inverse

Singularities cause the rank of the Jacobian to change

Damped Least Squares:
find solution that minimizes

\[ ||J - dx||^2 + \lambda^2 ||d\theta||^2 \]