Announcements

- Next class: games (James Kuffner)
- Last class: image based rendering + demos of other graphics classes
Character Animation
Advantages & Disadvantages

Key Framing
• Low level control
• A lot of manual labor
• Experienced user

Motion Capture
• Realistic human performance
• Hard to modify and generalize

Constrained Optimization
• High level control
• Little manual labor
• Easy for a naive user
Create animation from simple input

- User provides only a small amount of information
- We synthesize a physically valid motion to match the sketch
Applications

Allow game players create rich set of motions

Currently:
Limited set of predefined carefully hand-picked motions
Applications

http://www.magickeys.com/books/

Allow children to tell stories
Applications

Firefighters training environment, ETC, CMU

Allow novice users to create effective training scenarios
Talk Overview

- Video Textures and Character Animation from Video
- Move Graphs and Motion Graphs
- Interpolation
- Optimization
Animation From Video
Problem Definition

given video clip
generate an infinite amount of similarly looking video
Pictures, Videos, Video Texture

Picture

Video

static
dynamic but finite
Pictures, Videos, Video Texture

Video Texture

infinite and dynamic
Application

• Instead of static photo on a computer screen

• Advertising - beach with palm trees blowing
• Games - dynamic backdrops
• Video Based Animation
Algorithm Overview

Input Video

Find Good Transitions

Transition Graph

Random Play

User Control

Frame Numbers

Rendering
Finding Good Transitions

Similar frames make good transitions

Can use $L_2$ distance to compare two images
Finding Good Transitions (contd.)

Compute $L_2$ distance $D_{i,j}$ between all images

Matrix

Transition Matrix

Transition Graph
During Video Texture Synthesis
Transition from $i$ to $j$ if successor of $i$ is similar to $j$

$D_{i+1,j}$ is small
Threshold

high $\sigma$        low $\sigma$
Preserving dynamics
Finding Good Transitions

Similar frames make good transitions

Can use $L_2$ distance to compare two images
Preserving dynamics

Cost for transition $i \rightarrow j$

$$C_{i \rightarrow j} = \sum_{k = -N}^{N-1} w_k D_{i+k+1, j+k}$$
Dead ends

No good transition at the end of sequence
No Dead Ends
Synthesis - Random Play

- Begin at some frame $i$
- Select next frame probabilistically based on $P_{ij}$, for $j = 0 \ldots n$
Rendering

- Problem: Visible “Jumps”
Rendering (contd..)

• Solution 1: Crossfade from one sequence to the other.

\[
\begin{align*}
\cdots & A_{i-2} & \frac{3}{4} A_{i-1} & \frac{2}{4} A_i & \frac{1}{4} A_{i+1} \\
+ & \frac{1}{4} B_{j-2} & + & \frac{2}{4} B_{j-1} & + & \frac{3}{4} B_j & B_{j+1} & \cdots \\
A_{i-2} & A_{i-1}/B_{j-2} & A_{i-1}/B_{j-2} & A_{i-1}/B_{j-2} & B_{j+1} & \cdots
\end{align*}
\]
Results (contd..)

- Waterfall (cross fading)
Results (contd..)

- Waterfall (frequent jumps & cross fading)
Results (contd..)

• Video Portrait (random play, with fading)

Useful for web pages
Results (contd..)

• Campfire (single loop, with fading)
Results (contd..)

• Blowing grass (with fading)
Region-based analysis

• Divide video up into regions (by hand)

• Generate a video texture for each region
Region-based analysis (contd..)

- Divide video up into regions (by hand)

- Generate a video texture for each region
Region-based analysis (contd..)

• Divide video up into regions (automatically)

• Generate a video texture for each region
Video-based animation

- Like sprites computer games
- Extract sprites from real video
- Interactively control desired motion

©1985 Nintendo of America Inc.
Video sprite extraction

background subtraction and velocity estimation
Video sprite result
Video sprite control

• Augmented transition cost:

\[ C'_{i \rightarrow j} = D_{i+1, j} \]

\[ C'_{i \rightarrow j} = \alpha C_{i \rightarrow j} + \beta \text{ angle} \]

Similarity term  Control term

vector to mouse pointer

velocity vector
Interactive fish
Video sprite example
• Animation Of Real Animals
  – animals are difficult to train & animate
• Animator imposes constraints to control the motion of sprite
• Find frame sequence that minimize the cost function describing desired animation
Character Animation From Video

Data capture

Extract sprites using chroma keying

Find transitions by comparing all pairs of frames

Constraints, e.g. motion trajectory

Find sequence of frames $s_1...s_n$ that shows desired animation

Render and composite
Results
Results
Motion Graphs
Video Textures for Motion Capture

• Each frame is a character Pose
• Find Good Transitions

• Pose:
  – root position & orientation
  – joint angles
Finding Transitions Between Poses

• Find good metric for
  – \( D_{ij} = \text{difference}(P_i, P_j) \)

• Include root position into metric

• Do not include into metric:
  – character X and Z root position
  – character orientation around Y axis
Finding Good Transitions (Contd..)

Possible Transition

Translated Motion

Rotated Motion

(a)

(b)
Generating Motion

Motion Capture Data

Find Good Transitions

Transition Graph

Random Play

Frame Numbers

New Motion Generation

Prune Transitions

Fade Transitions
Interactive Control of Avatars with Human Motion Data
J. Lee, J. Chai, P. S.A. Reitsma, J.K. Hodgins, N. S. Pollard

- Motion Textures
- Controlling avatar
  - Games & Virtual Environments

A lot of Mocap Data → Find Transitions → Motion Data Structure → Search Data Structure → New Motion

User Control
Motion Graphs

Raw Captured Motion Data

\[ \text{WALK}_1 \quad A_0 \rightarrow A_1 \rightarrow A_2 \rightarrow A_3 \rightarrow A_4 \rightarrow A_5 \rightarrow A_6 \rightarrow A_7 \]

\[ \text{WALK}_2 \quad B_0 \rightarrow B_1 \rightarrow B_2 \rightarrow B_3 \rightarrow B_4 \rightarrow B_5 \rightarrow B_6 \rightarrow B_7 \]

Store: pose and change in root position from previous frame
Motion Graphs

Raw Captured Motion Data

WALK_1

WALK_2

A_0 A_1 A_2 A_3 A_4 A_5 A_6 A_7

B_0 B_1 B_2 B_3 B_4 B_5 B_6 B_7
Motion Graphs
Jehee Lee et al. SIGGRAPH 2002

Raw Captured Motion Data

WALK₁
A₀ A₁ A₂ A₃ A₄ A₅ A₆ A₇

WALK₂
B₀ B₁ B₂ B₃ B₄ B₅ B₆ B₇
Motions abstracted as **high-level behaviors** and organized into a finite state machine (FSM).

*(in contrast to connections of individual poses)*
Behavior Based Graphs

Behavior Planning for Character Animation
M. Lau & J. Kuffner

Manually-Constructed Behavior FSM

+ Search Efficiency
+ Memory Usage
+ Intuitive Structure

- Requires segmented motion data
- Requires FSM with appropriate transitions
Examples
Motion Graphs: Summary

• Build on the idea that natural human motion contains many similar poses – transitions can be found easily

• Can synthesize long multi-behavior motions

• Restricted to motions in the database – can not synthesize variations
Motion Graphs: Summary

- Build on the idea that natural human motion contains many similar poses – transitions can be found easily
- Can synthesize long multi-behavior motions
- Restricted to motions in the database – can not synthesize variations
Motion Interpolation (introduced in 1995)

motion capture

interpolated motion

Resulting motion is natural
Close to physically correct in many cases
How do we compute interpolated motion?

key events

linear interpolation

$M_1$

$M_2$

$M_{\text{interpolated}}$

time of motion
How do we compute interpolated motion?

Contact Phase of Motion:

\[ M(t) = \begin{cases} 
P(t): \text{root position over time} \\
Q(t) = q_1(t) \ldots q_{60}(t): \text{all joint angles over time}
\end{cases} \]

Linear interpolation:

\[ M(t, w) = \begin{cases} 
P(t) = wP_1(t_1) + (1 - w)P_2(t_2) & \text{root position} \\
q_i(t) = wq_{1i}(t_1) + (1 - w)q_{2i}(t_2) & \text{joint angles}
\end{cases} \]
Verbs and Adverbs: Multidimensional Motion Interpolation
Charles Rose, Michael Cohen and Bobby Bodenheimer
Verbs and Adverbs: Multidimensional Motion Interpolation
Charles Rose, Michael Cohen and Bobby Bodenheimer
Verbs and Adverbs: Multidimensional Motion Interpolation
Charles Rose, Michael Cohen and Bobby Bodenheimer
Interpolation: Summary

• Interpolation produces surprisingly **natural motion**

• Motion is often close to **physically correct**

• Sequences must be **properly aligned** and of a **single behavior**
Each pose is represented by interpolation of existing poses:

\[ P_{\text{new}}(t) = P_1(t)w + P_2(t)(1-w) \]

\( P_i(t) \) path through the motion graph
Path Through Motion Graph

\[ P_{\text{new}}(t) = P_1(t)w + P_2(t)(1-w) \]

\( P_i(t) \) path through the motion graph

\( WALK_1 \)

\( WALK_2 \)

\( A_0, A_1, A_2, A_3, A_4, A_5, A_6, A_7 \)

\( B_0, B_1, B_2, B_3, B_4, B_5, B_6, B_7 \)
Path Through Motion Graph

\[ P_{\text{new}}(t) = P_1(t)w + P_2(t)(1-w) \]

\[ P_i(t) \] path through the motion graph
Interpolated Motion Graph

$P_1(t)$

poses from walk$_1$

A$_0$
A$_1$
A$_2$
A$_3$
A$_4$
A$_5$
A$_6$
A$_7$

poses from walk$_2$

B$_0$
B$_1$
B$_2$
B$_3$
B$_4$
B$_5$
B$_6$

$t_0$ $t_1$ $t_2$ $t_3$ $t_4$ $t_5$ $t_6$ $t_7$ $t_8$ $t_9$ $t_{10}$

time of synthesized motion
Interpolated Motion Graph

\[ P_{new}(t) = P_1(t)w + P_2(t)(1-w) \]

poses from walk$_1$

poses from walk$_2$

time of synthesized motion

Interpolating paths through motion graph
Why good representation?

\[ P_{\text{new}}(t) = P_1(t)w + P_2(t)(1-w) \]

\[ P_i(t) \] path through the motion graph

Unlike motion graph:
Can generate variations of motions in mocap database

Unlike interpolation:
Can optimize for complex longer motions
Do not need to pre-process motions into similar segments by hand

Retains naturalness and physical realism
Retains natural transitions of motion graphs
The benefit of interpolation

- 31% success with no interpolation
- 99% success with interpolation

270 sample points
Examples of motions – jumping across stones
Examples of motions – obstacle course
Examples of motions – obstacle course
Constrained Optimization

_Luxo Lamp_ (Spacetime Constraints, Witkin and Kass, SIGGRAPH 88)
Optimization Problem Setup

**Given:**
- Particle is at point $X_a$ at time $t_1$
- Particle is at point $X_b$ at time $t_n$

**Find:**
- Trajectory of the particle that minimizes fuel consumption

\[ \sum \text{forces} = m \cdot \ddot{x} \]

\[ f - mg = m \cdot \ddot{x} \]
Optimization Problem Setup

**Optimization Problem:**
- **unknowns**
  \[ x(t_1), x(t_2) \ldots x(t_n) \]
- **constraints**
  \[ x(t_1) = x_a \]
  \[ x(t_n) = x_b \]
  \[ F_{\text{min}} \leq f(t) \leq F_{\text{max}} \]
- **minimize**
  \[ R = \sum_{i=1}^{n} |f(t_i)|^2 \]
Optimization Problem Setup

**Optimization Problem:**

- **unknowns**
  \[ x(t_1), x(t_2), \ldots, x(t_n) \]

- **constraints**
  \[ x(t_i) = x_a \]
  \[ x(t_n) = x_b \]
  \[ F_{\text{min}} \leq f(t) \leq F_{\text{max}} \]

- **minimize**
  \[ R = \sum_{i=1}^{n} |f(t_i)|^2 \]
Optimization Problem Setup

**Number of unknowns?**

\[ n \]

**Optimization Problem:**

- **unknowns**
  \[ x(t_1), x(t_2) \ldots x(t_n) \]

- **constraints**
  \[ x(t_i) = x_a \]
  \[ x(t_n) = x_b \]
  \[ F_{\text{min}} \leq f(t) \leq F_{\text{max}} \]

- **minimize**
  \[ R = \sum_{i=1}^{n} |f(t_i)|^2 \]
unknowns:
P_x at time t_1 \ldots t_n
P_y at time t_1 \ldots t_n
Q_0 at time t_1 \ldots t_n
Q_1 at time t_1 \ldots t_n
Q_2 at time t_1 \ldots t_n
Q_3 at time t_1 \ldots t_n

\( (P_x, P_y) \) root position

6n unknowns
Human in 3D space

50 parameters (degrees of freedom)

50 unknowns

nonlinear optimization function

nonlinear constraints

root position

root orientation

joint angles

50 degrees of freedom
Use PCA to compute low-dimensional space

User input poses

Optimization
Announcements

- Course Evaluation is now open
- Until Monday, May 7th
- Please complete the evaluation
- We read it and listen to what you say