Animation by Example

Michael Gleicher
and the UW Graphics Group
University of Wisconsin- Madison
www.cs.wisc.edu/~gleicher
www.cs.wisc.edu/graphics

The Challenge

- High Quality, Expressive Motion
  - Need motion capture (examples)
- Flexible, long-running, controllable
  - Need synthesis
- Synthesis from Examples!

Survey of Techniques

Flexibility:
- Link motions to make sequences
- Blend motions to gain control

Use Databases of Examples:
- Find related motions in databases
- Combine data for interactive systems

Survey of Projects

- Motion Graphs
  - Link motions to make long sequences
- Snap Together Motion
  - Synthesis for interactive systems
- Match Webs
  - Find related motions in a database
- Registration Curves + Parametric Families
  - Combine motions to make spaces
- Plus some others…

Work with Lucas Kovar, Hyun Joon Shin, …

Idea:
Put Clips Together

- New motions from pieces of old ones!
- Good news:
  - Keeps the qualities of the original (with care)
  - Can create long and novel "streams" (keep putting clips together)
- Challenges:
  - How to connect clips?
  - How to decide what clips to connect?

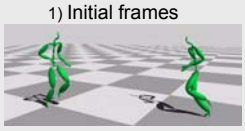
Connecting Clips
Transition Generation

- Transitions between motions can be hard
- Simple method work *sometimes*
  - Blends between aligned motions
  - Cleanup footskate artifacts
  - Just need to know when is "sometime"

What is Similar?

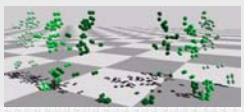
- Factor out invariances and measure

1) Initial frames

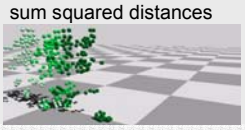


2) Extract windows

3) Convert to point clouds

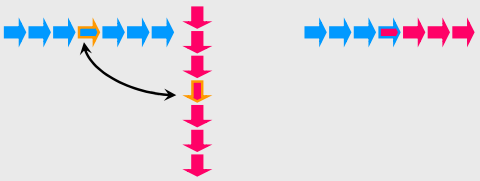


4) Align point clouds and sum squared distances



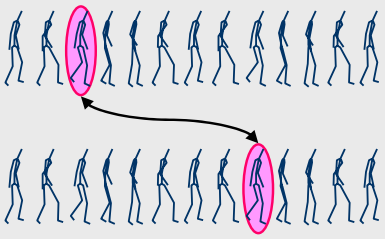
An easy point to miss: Motions are Made Similar

- "Undo" the differences from invariances when assembling
- Rigidly transform motions to connect



Building a Motion Graph

- Find Matching States in Motions

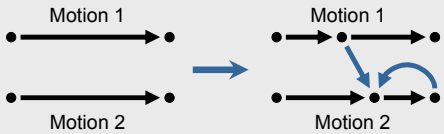


Motion Graphs

Kovar, Gleicher, Pighin '02

Start with a database of motions, each with type and constraint information.

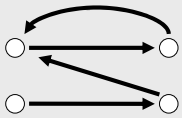
Goal: add transitions at opportune points.



Other Motion Graph-like projects elsewhere
Differ in details, and attention to detail

Motion Graphs

Idea: automatically add transitions within a motion database



Edge = clip

Node = choice point

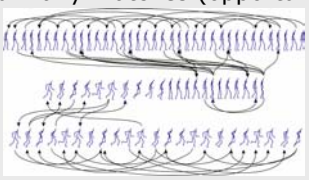
Walk = motion

Quality: restrict transitions

Control: build walks that meet constraints

Automatic Graph Construction

- Find many matches (opportunistic)



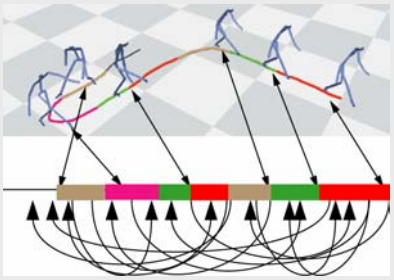
- Good: Automatic
- Good: Lots of choices

Using a motion graph

- Any walk on the graph is a valid motion
- Generate walks to meet goals
  - Random walks (screen savers)
  - Search to meet constraints
- Other Motion Graph like projects elsewhere
  - Differ in details, and attention to detail

An example:

Building a Motion Graph



An example:

Using a Motion Graph

- Given a path
- Find a motion that minimizes distance
- Combinatorial optimization

Video: mographs.avi

Why is this good?

- Search the graphs for motions
- Look ahead avoids getting stuck
- Cleanup motions as generated
- Plan "around" missing transitions
- Optimization gets close as possible

Not OK for Interactive Apps!

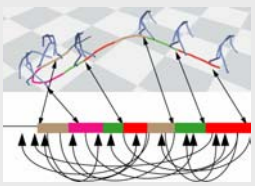
Need different tradeoffs

What about interactive?

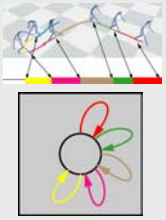
- Different set of tradeoffs!
- Runtime must be:
  - Responsive
  - Low overhead
- Willing to sacrifice quality to get

Contrived Graph Structure?

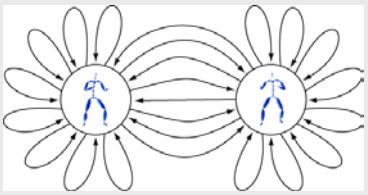
Search: Look ahead to get where you need to go



React: Always lots of choices. Something close to need.

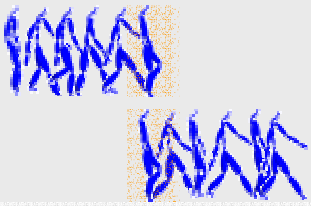


Gamers use these



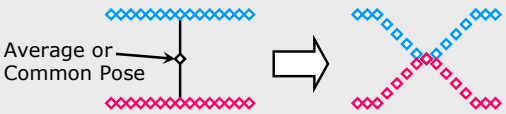
Snapable Motions

- What if motions matched exactly?
  - Match both state and derivatives
  - Match reasonably at a larger scale



Make motions match exactly

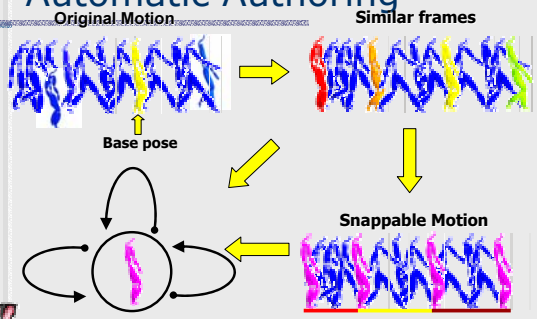
- Add in displacement maps
- Bumps we add to motions
- Modify motions to common pose
- Compute changes at author time



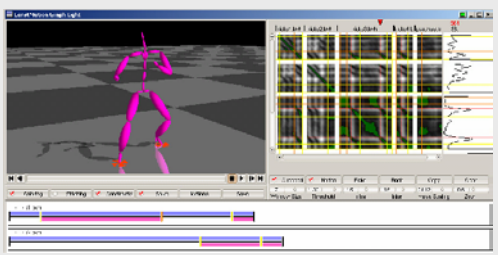
Semi-Automatic Graph Construction

- Pick set of *match frames*
  - User selects
  - System picks "best" one
- Modify motions to build hub node
- Check graph and transitions

Automatic Authoring

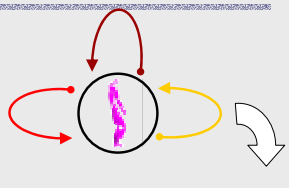


Building the Motion Graphs

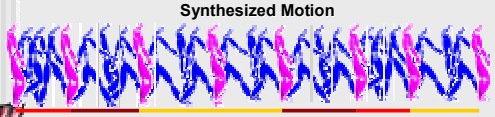


Video: stm-intro

Runtime



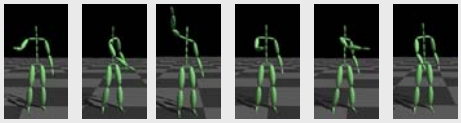
Synthesized Motion



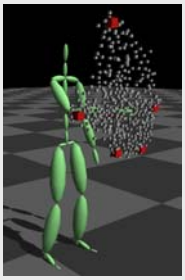
…

Limitations of Motion Graphs

- Graphs provide discrete choices
- Use pieces of the database
- Can't capture ALL examples
- Synthesize new motions between example by blending

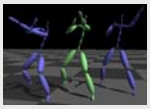


Motions Between examples

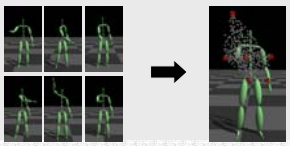


Parameterized Motions

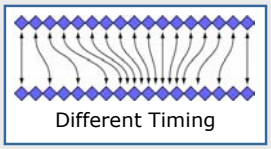
Blend captured motions to make new ones.



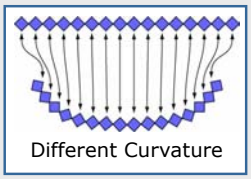
Create a natural parameterization for intuitive access to these new motions.



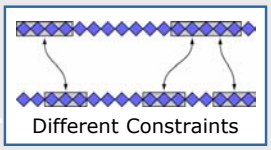
Blending requires similar motions



Different Timing



Different Curvature



Different Constraints

video: regCurves

Registration Curves

- Encode the relationships between *similar* motions

(video of pair blending apps)

If we have a big database…

- How do we find similar motions?
- How do we use several examples?

Adapting to Large Data Sets

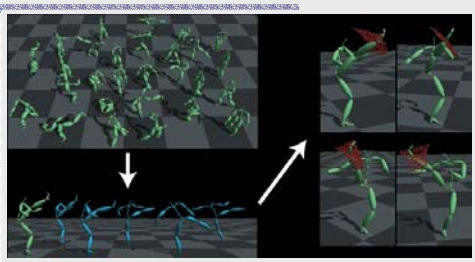
Previous: small, "contrived" data sets (e.g., Rose et al. '98, '02).

(Kovar and Gleicher '04): Adapt parameterized motions to large data sets

- Automatically find and extract examples
- Automated blending (K&G '03)
- Accurate and stable parameterization

Input: database + one example + parameterization function

Motion Families



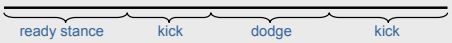
Database-Controllable Clip

Motion Families

- Match Webs
  - Search for similar motions
- Registration
  - Align motions for blending
- Parameterization
  - Define useful controls
- Sampling
  - Improve nearest neighbor interpolation

Finding Motions

Example motions are buried in longer motions.



Strategy: search for motion segments similar to a query.

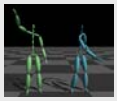
Why It Is Hard to Find Motions

- Motions can be different lengths.

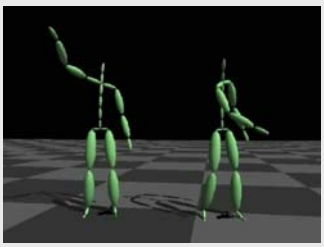
- Complicated distance metrics

$$D(F,F') = \min_{\theta,x_0,z_0} \sum_i \| p_i - T(\theta,x_0,z_0)p_i' \|^2$$

- Logically similar ≠ numerically similar.

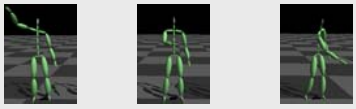


Similar?



Search Strategy

Find "close" matches and use as new queries.



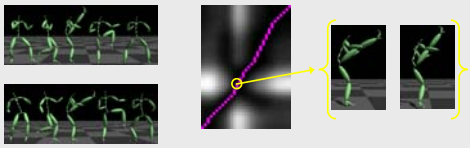
One search may involve many queries.

Precompute potential matches for interactivity.

Computing Distance Between Motions

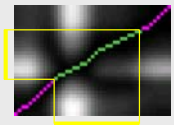
Distance between corresponding frames (in the best time warp)

- Factors out timing differences
- Allows arbitrary distance metrics for frames

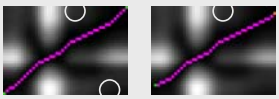


Precomputing Matches: Insights

Any subset of an optimal path is optimal.

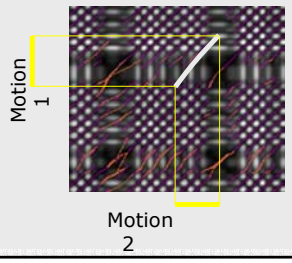


Optimal paths are redundant under endpoint perturbation.



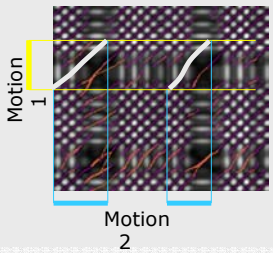
Precomputing Matches: Match Webs

Compute a grid of distances between pairs of frames and find long, locally optimal paths.



Precomputing Matches: Methods

At run time, intersect queries with the match web to find matches.



Search Results

- 37,000 frame data set with ten different kinds of motions.
- 50 minutes to compute match web
- 21MB on disk
- All searches (up to 97 matches) in ≤ 0.5s
- Manual verification of accuracy

Natural Parameterizations

Blend weights offer poor controls

We need more natural parameters.

$g(M) = p$

motion — parameters

reaching	hand position at apex
turning	change in hip orientation
jumping	max height of center of mass

From Parameters to Blend Weights

It is easy to map blend weights to parameters.

$$f(\mathbf{w}) = g(w_1\mathbf{M}_1 \oplus \ldots \oplus w_n\mathbf{M}_n) = \mathbf{p}$$

blend weights — blend — parameters

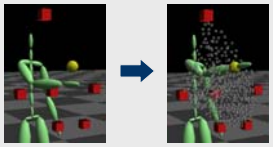
But we want $w=f^{-1}(p)$!

This has no closed form solution!

Building Parameterizations

Given samples (p,w), we can approximate f^{-1} with k nearest neighbor interpolation.

Accuracy: create new blends to get additional



Require "reasonable": $\sum_i w_i = 1$

$-\varepsilon \le w_i \le 1+\varepsilon$

Video: families

A Driving Application

Thanks!

- To the UW graphics gang.
- Animation research at UW is sponsored by the National Science Foundation, Microsoft, and the Wisconsin University and Industrial Relations program.
- House of Moves, IBM, Alias/Wavefront, Discreet, Pixar and Intel have given us stuff.
- House of Moves, Ohio State ACCAD, and Demian Gordon for data.
- And to all our friends in the business who have given us data and inspiration.