Motion Capture and Data Driven Animation

CSE 888 Au ’08

Neelima Karanam

Research Papers

- Achieving Good Connectivity in Motion Graphs

- Puppet Master: Designing Reactive Character Behavior by Demonstration

Achieving Good Connectivity in Motion Graphs

Liming Zhao and Alla Safonova
University of Pennsylvania

SCA 2008

Problem

- Smooth transitions
- Good connectivity

- Smooth - no visual discontinuity to the motion
- Connectivity - measure of how quickly one can transition from a pose in one behavior to a pose in another behavior

Contribution

- An algorithm that automatically constructs an unstructured motion graph with very smooth transitions and much better connectivity than offered by standard motion graphs

- They call this graph wcMG (well-connected Motion Graph)

- Algorithm
  - Guarantee the physical correctness of the transitions in the motion graph
  - Grow the overall graph size as little as possible while maintaining the best connectivity among the original poses

Contribution

- Resulting wcMG uses exactly the same representation as a standard motion graph

- Motions generated from graph wcMG require no post-processing

- Demonstrates better responsiveness to user control and better visual quality of generated motions than standard motion graphs

Standard Motion Graphs (MG)

- Input - set of motion capture clips
- Each frame is treated as a graph node and two nodes are connected by a directed edge if there exists a smooth transition from one to the other
- Metric for computing smooth transitions between the nodes in the graph - compare similarity of two nodes *i* and *j*, and if the similarity is lower than a user specified value, node *i* can be connected to node *j+1* and node *j* can be connected to node *i+1*
- Adopted Kovar's point cloud metric with weights on different joints from Wang and Bodenheimer's work

Standard Motion Graphs (MG)

- After thresholding the similarity value by a user-specified value for naturalness, a dense transition matrix is obtained, where rows and columns are the frames from the motion clips

- The transition value at row *i* and column *j* is

  $t = \exp(-s/\sigma)$

  where s is the similarity value between frame *i* and frame *j-1*

- Leave only transitions that represent local maxima in the transition matrix and compute the largest strongly connected component (SCC) of the graph to remove dead ends

Standard Motion Graphs (MG)

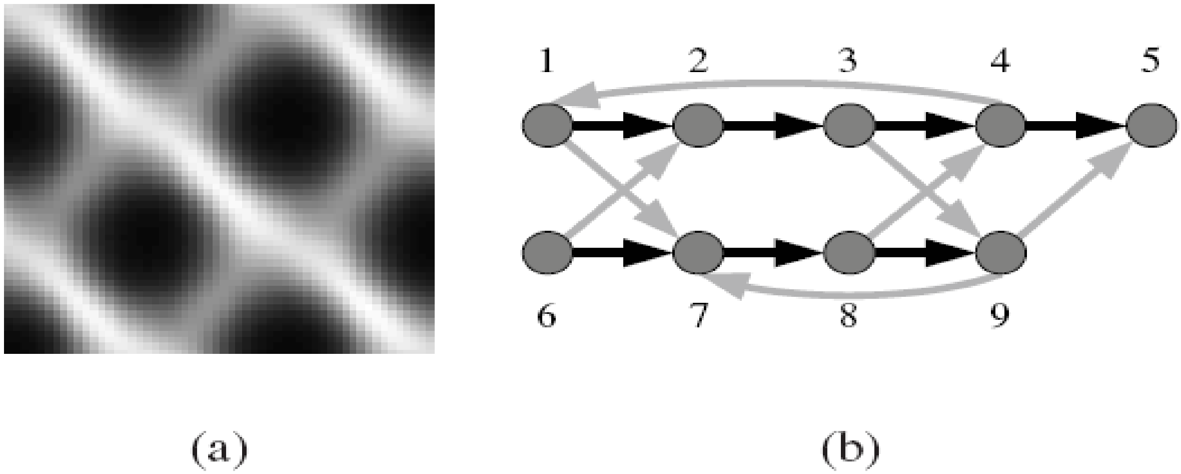


Figure 4: *Motion Graph construction. (a): Transition matrix. The entry at (i,j) contains the transition value from node i to node j. Bright values indicate good transitions and dark values indicate poor transitions. (b): A simple motion graph. Black edges are from the motion clips and gray edges are created through the similarity measurement. The largest strongly connected component is {1,2,3,4,7,8,9}.*

Well-connected Motion Graphs (wcMG)

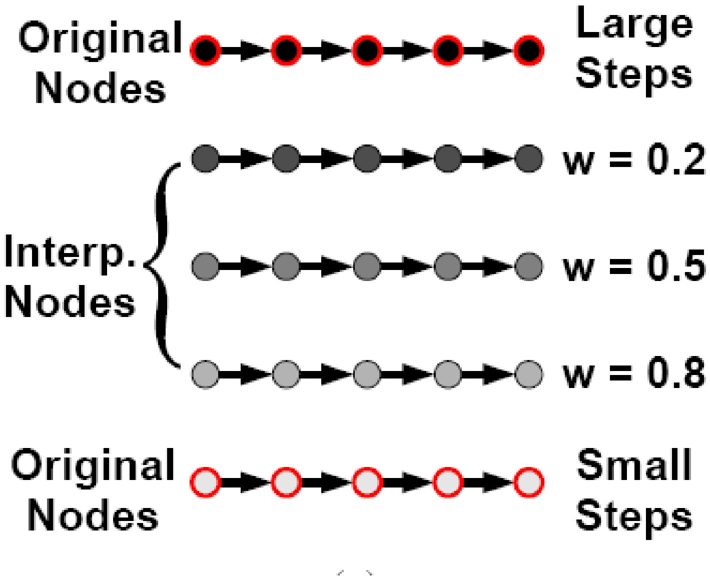
Construction Process

- ➢ Interpolation step
- ➢ Transition creation step
- ➢ Node reduction step

Well-connected Motion Graphs (wcMG)

Interpolation Step

- Given a motion data set, they first separate the motions into segments based on contact with the environment using the technique from Lee to identify the contacts
- The interpolated technique by Safonova and Hodgins is then used to interpolate all pairs of segments with the same contact information
- The original nodes, the interpolated nodes, and the natural transitions are all added to the wcMG

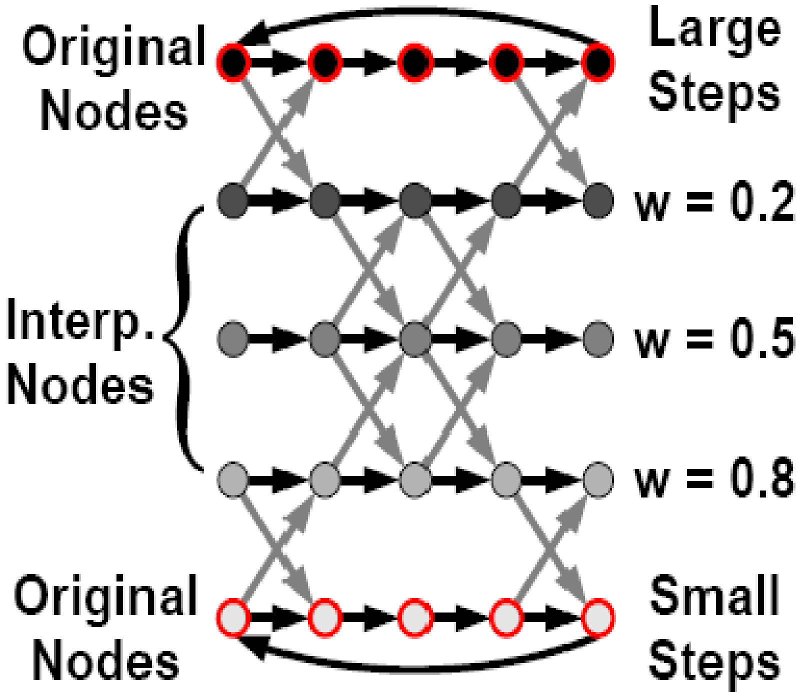


Interpolation Step

Well-connected Motion Graphs (wcMG)

Transition Creation Step

- At this stage, graph wcMG contains only natural (original or interpolated) edges between the original and the interpolated nodes
- Additional edges are added to the graph using the same process used in constructing the standard motion graphs

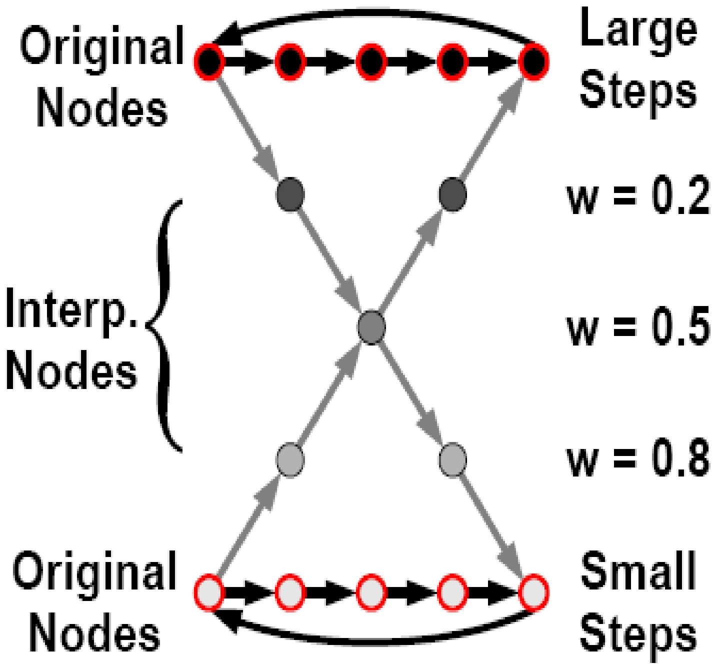


Transition Creation Step

Well-connected Motion Graphs (wcMG)

Node Reduction Step

- Remove all nodes and edges from the graph that are not necessary for connecting the original nodes
- Compute subset of nodes and edges, that best connect the original nodes - find the best paths between the original nodes
- Measure of best can be application based - they used a weighted average of the length and the smoothness of the transitions



Node Reduction Step

Well-connected Motion Graphs (wcMG)

Node Reduction Step

- First, *S* is initialized to include all the original nodes
- Next, for each of the original nodes, find the best transition path to all other original nodes in graph wcMG
- This is done by setting the transition cost according to the metric and running a Dijkstra's shortest path algorithm for each of the original nodes iteratively
- All nodes that belong to the best transition paths between the original nodes are added to *S* and *E*
- The final motion graph size increases linearly with the size of the motion data set - Graph obtained called *Optimal Graph*

Well-connected Motion Graphs (wcMG)

Trade-off between Size and Graph Quality

- Extend algorithm to allow a user to further decrease the size of the graph wcMG at the expense of the best transition path between the original nodes

- Introduce a sub-optimality scalar μ to the previous best path search process

- Using μ, sub-optimal graphs with smaller number of nodes and edges can be computed but at the expense of the optimal paths between the original nodes

Well-connected Motion Graphs (wcMG)

Trade-off between Size and Graph Quality

- First, *S* is initialized to include only the original nodes

- However, during each iteration of the Dijkstra’s shortest path search, the cost of each edge *e* is computed as

$$c = \begin{cases} t, & \text{if } i, j \in S, \\ \mu \cdot t, & \text{otherwise.} \end{cases}$$

where *i* and *j* are nodes that edge *e* connects and *t* is the cost of the edge

Well-connected Motion Graphs (wcMG)

Trade-off between Size and Graph Quality

- After each iteration, the nodes along the shortest path are added to the set *S* and this process repeats for every original node

- μ compromises between graph quality and graph size

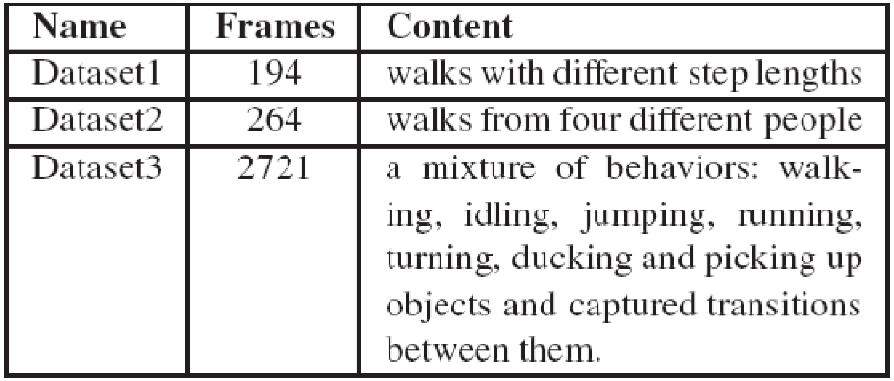
- As μ increases, it forces the search to reuse the nodes which are already in S, even though paths that pass through these nodes have worse quality than those paths outside *S*

- As a result, the size of *S* decreases drastically as μ increases since paths outside of *S* are penalized by μ times even if they are of lower transition cost or shorter length

Experimental Analysis

- Three data sets
- Computation time for each construction step of graph wcMG is as follows:
  - 7 seconds for Interpolation Step
  - 4.5 hours for Transition Creation Step
  - 0.5 to 1hr for Node Reduction Step

Table 1: *Data Set Description*



Name	Frames	Content
Dataset1	194	walks with different step lengths
Dataset2	264	walks from four different people
Dataset3	2721	a mixture of behaviors: walking, idling, jumping, running, turning, ducking and picking up objects and captured transitions between them.

Experimental Analysis

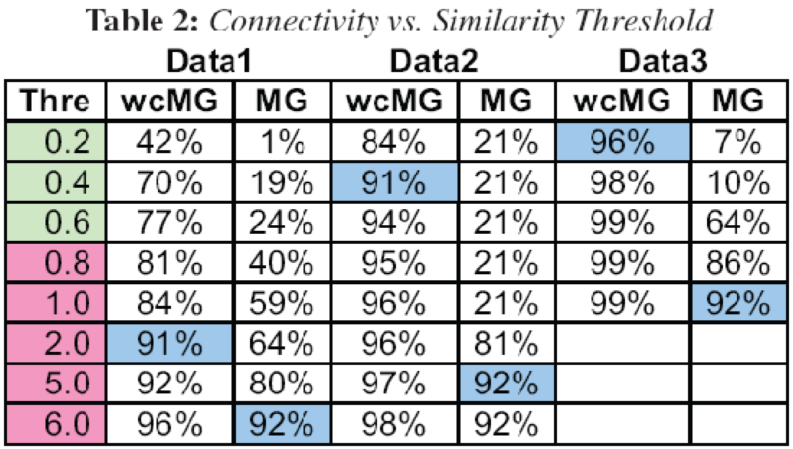


Table 2: *Connectivity vs. Similarity Threshold*

	Data1		Data2		Data3	
Thre	wcMG	MG	wcMG	MG	wcMG	MG
0.2	42%	1%	84%	21%	96%	7%
0.4	70%	19%	91%	21%	98%	10%
0.6	77%	24%	94%	21%	99%	64%
0.8	81%	40%	95%	21%	99%	86%
1.0	84%	59%	96%	21%	99%	92%
2.0	91%	64%	96%	81%		
5.0	92%	80%	97%	92%		
6.0	96%	92%	98%	92%		

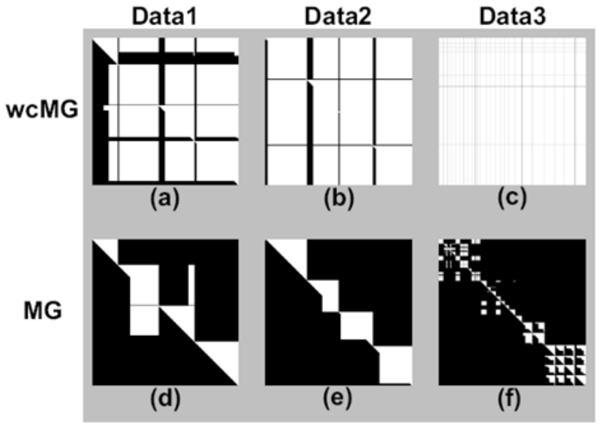


Figure 6: *Transitive Closure Matrix. Ideally, we would like to have a plain white image which indicates that every frame can transition to every other frame. The coverage of white area gives an estimate of the* connectivity.

Size of the Largest Strongly Connected Component

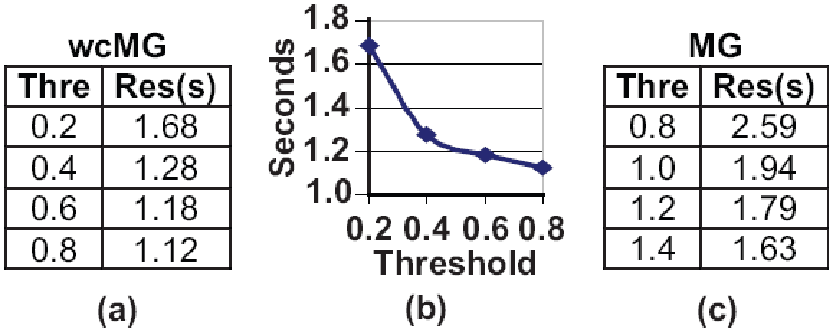
- SCC as a percentage of the original data set size - shows how well the nodes in the data set are connected to each other and how well the motion graphs are utilizing the motion data set
- Similarity threshold below 0.6 - smooth motions with no visual discontinuity
- Similarity threshold above 0.6 - visually noticeable discontinuity
- Graph wcMG offers better data set utilization with low similarity thresholds

Experimental Analysis

Time Between Frames

- Compute the average transition time between all pairs of the original nodes - this measure provides insight into how quickly it is possible to transition between different poses in the data set

- Transition time decreases monotonically as the similarity threshold increases, because lower quality transitions become possible



wcMG

Thre	Res(s)
0.2	1.68
0.4	1.28
0.6	1.18
0.8	1.12

(a)

(b)

MG

Thre	Res(s)
0.8	2.59
1.0	1.94
1.2	1.79
1.4	1.63

(c)

Figure 7: *Average Transition Time vs. Threshold. (a)Average Transition Time of graph wcMG at different similarity thresholds. (b) Plot of Average Transition Time vs. threshold for graph wcMG shows an exponential decrease. (c)Average Transition Time of graph MG at different similarity thresholds.*

Experimental Analysis

Local Maneuverability Measurement (LM)

- An alternative way to measure the graph connectivity is the local maneuverability measurement (LM) proposed by Reitsma and Pollard. They compute the average minimum time needed to transition to a particular behavior:

$$LM_k = \frac{1}{\|C\|} \sum_{c \in C} (0.5 * D_c + MDMPC_{c,k})$$

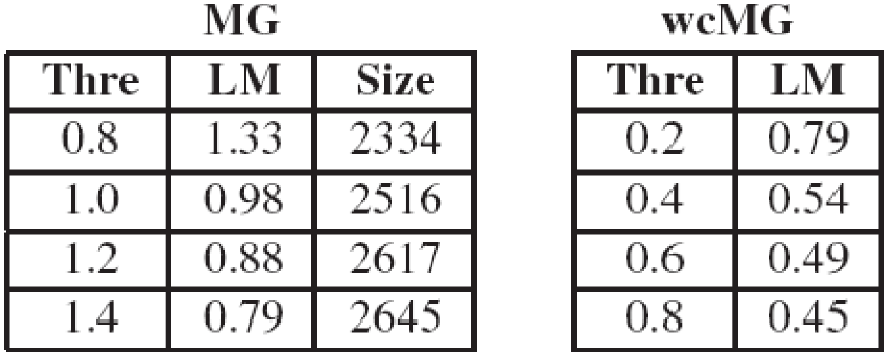
where C is the set of all the motion segments in the motion graph, $MDMPC_{c,k}$ is the minimum time to transition from the end of motion segment c to any instance of behavior k, and D_c is the motion segment length

- Compute average LM measurement over all behaviors:

$$LM = \frac{1}{\|K\|} \sum_{k \in K} (LM_k)$$

where K is all the behaviors in the motion graph

Table 3: *Local Maneuverability*



MG

Thre	LM	Size
0.8	1.33	2334
1.0	0.98	2516
1.2	0.88	2617
1.4	0.79	2645

wcMG

Thre	LM
0.2	0.79
0.4	0.54
0.6	0.49
0.8	0.45

Experimental Analysis

Graph Size

- The number of nodes and edges grow
  - Quadratically before node reduction
  - Linearly after node reduction

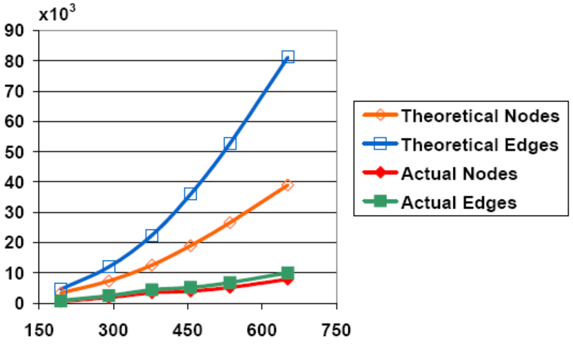
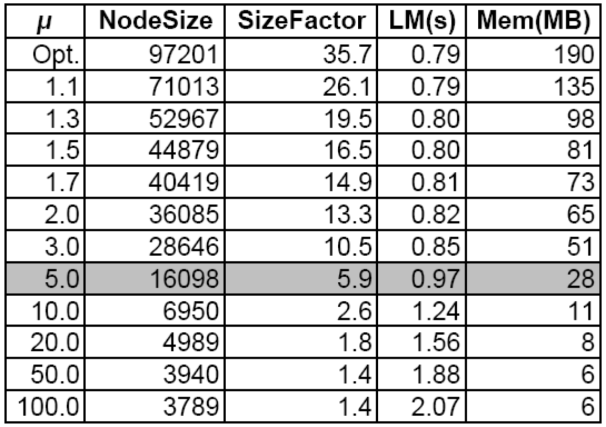


Figure 8: *Graph Size vs. Motion Data Size. The horizontal axis is the total number of frames in the database and the vertical axis is the number of nodes and edges.*

Parameter μ

- The user can choose an appropriate value for μ to balance between memory requirements and transition quality. They used μ = 5.0

Table 5: *The effects of the suboptimality parameter μ*



μ	NodeSize	SizeFactor	LM(s)	Mem(MB)
Opt.	97201	35.7	0.79	190
1.1	71013	26.1	0.79	135
1.3	52967	19.5	0.80	98
1.5	44879	16.5	0.80	81
1.7	40419	14.9	0.81	73
2.0	36085	13.3	0.82	65
3.0	28646	10.5	0.85	51
5.0	16098	5.9	0.97	28
10.0	6950	2.6	1.24	11
20.0	4989	1.8	1.56	8
50.0	3940	1.4	1.88	6
100.0	3789	1.4	2.07	6

Application to Interactive Control

- Train a control policy for both graphs wcMG and MG using value iteration technique
    - wcMG threshold 0.2
    - MG higher threshold to allow low quality transitions

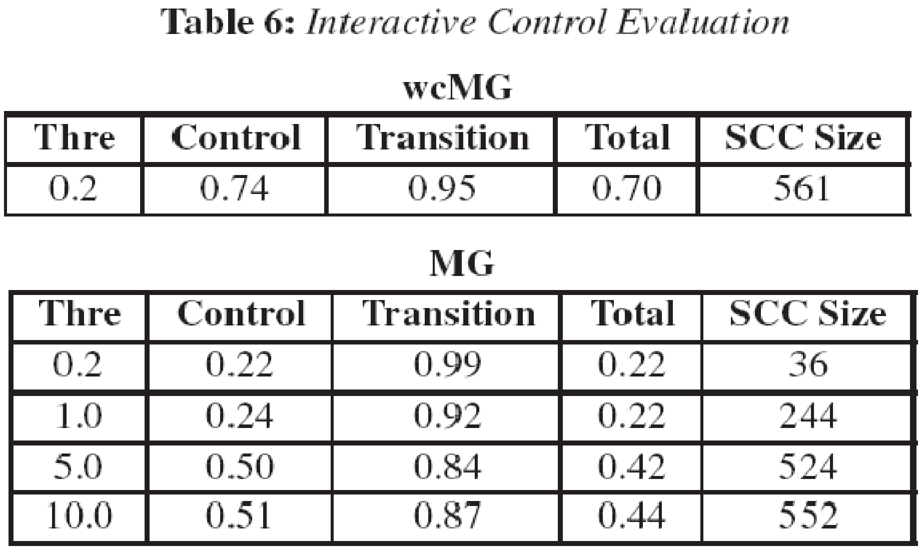


Table 6: *Interactive Control Evaluation*

wcMG

Thre	Control	Transition	Total	SCC Size
0.2	0.74	0.95	0.70	561

MG

Thre	Control	Transition	Total	SCC Size
0.2	0.22	0.99	0.22	36
1.0	0.24	0.92	0.22	244
5.0	0.50	0.84	0.42	524
10.0	0.51	0.87	0.44	552

Future Work

- Develop a more optimal algorithm

- Better segmentation and appropriate interpolation techniques to produce well-connected graphs for other data

Puppet Master: Designing Reactive Character Behavior by Demonstration

James E. Young,Takeo Igarashi and Ehud Sharlin
University of Calgary, The University of Tokyo
SCA 2008

Problem

- Interactive characters react convincingly to real-time user input while maintaining a coherent personality

- Programming such a character is a difficult problem, particularly when trying to achieve a certain personality or interaction style

Contribution

- Introduce a programming-by-demonstration approach to make the design of interactive character behavior accessible to artists
- In training phase, demonstrate paired motion of two characters, with one character reacting to the actions of the other. At run-time, the end-user controls the main character and the system generates, in real time, the reactive behavior of the other character with the characteristics observed in the training data
- Behavior synthesis algorithm - extension of image analogies algorithm
- The system learns reactive behavior from an example pair of motions and applies it to a new input motion
- Introduce meaningful behavior-related features, a method for balancing between the similarity and coherence metrics, and separately synthesize general motion trajectory and motion texture, integrating them during the final stages of motion synthesis

System Overview

- The ultimate goal is to allow the intuitive design of all aspects of behavior to create believable whole-body characters.

- As initial exploration, this paper focuses on character locomotion.

- Implemented two interfaces
    - Mouse-based GUI - sequential training, the direction a character is looking cannot be specified
    - Tabletop Tangible User Interface (TUI)

Algorithm

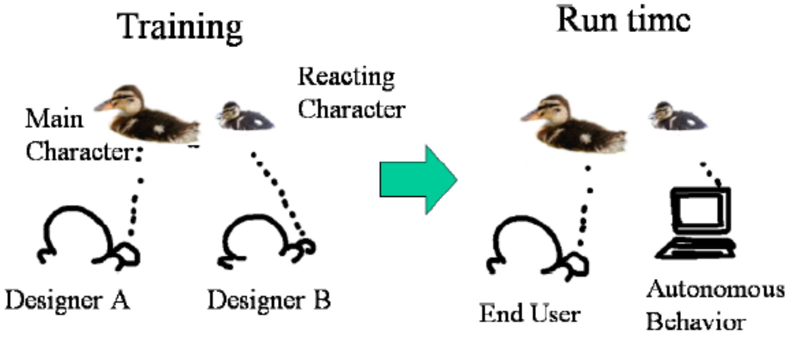


Figure 1: *First the behavior of a reactor is demonstrated in response to a main character's behavior. At run time, the reacting character's behavior is synthesized, reproducing the personality and emotion demonstrated in the training.*

```
Loop
    e = BestMatch($I^t$, $R^t$, $I$, $R$)
    newMovement = Generate(e)
    R.append(newMovement)
    I.append(getNewInput())
```

Data and Features

- Data - Time-dependent array containing the location *x*,*y* and the direction *d* of each entity at each time point

- Focus on relationship between the two entities and changes in local state and these features are evaluated over a time window to encapsulate a trend over time

- Features - velocity, relative position, normalized look direction, relative look direction, absolute movement direction, Δdirection

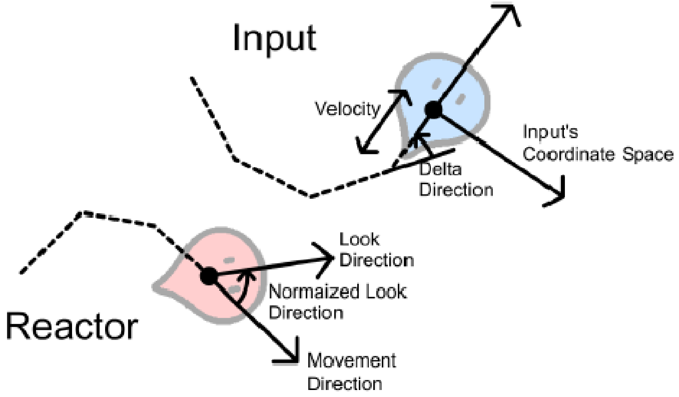


Figure 4: *Data features: All features except relative position are on both entities, but only shown on one for image clarity.*

BestMatch (Similarity Metric)

- Similarity metric heavily based on Image Analogies but applied to dynamically generate motion sequences

- This metric has two key components
    - Overall situational similarity
    - Generated path coherency

- These are checked in parallel and combined in each step over a given movement-history neighborhood *n*

Situation Similarity

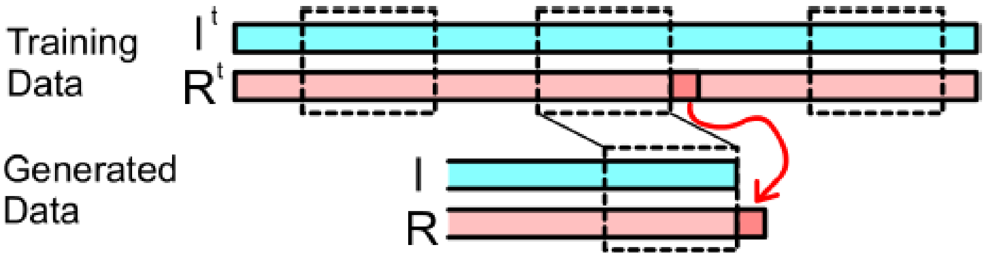
- Based on relationship between the two entities, using relative position, relative look direction and velocity features

- Compares the *n* most recent pieces of user input and generated output *I*,*R* to a moving window of size *n* over the training data I^t, R^t

- Features compared using Euclidean distance squared

- These distances are then summed over the window, providing a measure of similarity at that window location

- A smaller value represents a better match



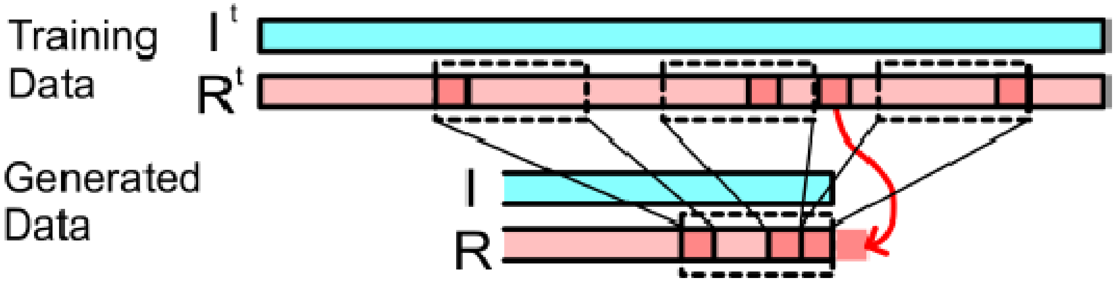
(a) Situation Similarity compares recent real-time data to the entire dataset.

Generated Path Coherency

- This emphasizes the shape, style and features of the generated path *R* in relationship to the trained R^t

- This metric uses normalized look direction, delta direction, relative position, and velocity

- This helps to ensure a generation that matches the characteristics of R^t when the situation similarity is weak



(b) Generated Path Coherency examines the regions of the training data recently used in generation.

Similarity Balancing

- The Image Analogies algorithm combines the two similarity metrics by statically weighting them with a coefficient *k* to add bias; the metric with the best weighted score is selected for that step.

- Coherence loop problem with the above approach

- Changed *k* to a dynamic value to target a situation-similarity-to-coherency-match ratio *t*.

- They used a 1:1 target ratio

- The data from R^t immediately following the source region is passed to the generation system

- Problem : noise; resulting in distracting rapid character movement

Output Generation

- Cannot copy data from BestMatch directly to the output because many features depend on history and are relative to the other entity

- Generation approach is to decompose emotion (the motion) into its
    - low-frequency part - intentional move to certain relational position
    - high-frequency part - texture of the motion

  and treat them separately.

General Trajectory Generation

- Motion is generation using relative position, normalized look direction and velocity
- Noise resulting from the BestMatch instability is dealt with by applying a linear smooth over a history of three samples

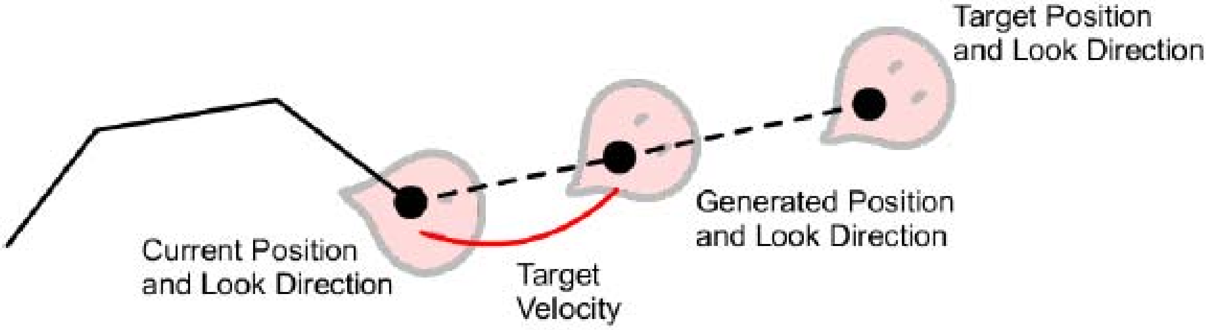


Figure 6: *To avoid drastic jumps, the reactor moves towards the target position with the velocity taken from training data.*

Detail Incorporation

- To restore detail removed by smoothing they do frequency analysis using Haar wavelets, extracting high-frequency detail from the target and directly incorporating it into the output.

- Apply Haar decomposition on the motion direction feature as this captures path texture irrespective of velocity

- The high-frequency data from the target is used to perturb the generated trajectory

Evaluating Puppet Master

- Evaluation consisted of two parts
    - Artist study - design new behaviors using the system
    - End-user study - interact with behaviors created in the first study

- Pucks are tracked at 100fps by a six-camera Vicon motion tracking system

Limitations and Future Work

- Current implementation does not handle dynamics over a large time scale and will fail to accurately represent an angry character gradually calming down

- Ultimate goal is to design all aspects of character behavior

Thank You!