Video Textures

Arno Schödl
Richard Szeliski
David H. Salesin
Irfan Essa

presented by Marco Meyer
Motivation

• Images
  – lack of dynamics

• Videos
  – finite duration
  – lack of „timeless“ quality of image
Motivation

Image $\leftrightarrow$ Video Texture $\leftrightarrow$ Video

endless $\leftrightarrow$ endless & in motion $\leftrightarrow$ in motion

Applications:
- dyn. scenes on the web
- dyn. background
- user–controlled animation
Outline

- Introduction
- Representation
- MAIN SYSTEM
- Extensions
- Results
- Future Work & Discussion
Introduction

• If two frames are similar, we can create a transition

• Find sequence with respect to global structure

• Smooth the selected transitions
Representation

- VT can be seen as Markov processes

- Matrix of probabilities
  - matrix mostly sparse

- Set of explicit links with associated probability
Outline

• Introduction
• Representation
• MAIN SYSTEM
• Extensions
• Results
• Future Work & Discussion

Analysis

Synthesis
  random play
  video loop

Rendering
Analysis

- Preparation steps
  - brightness equalization
  - video stabilisation

- We need a measure of similarity
  - $L_2$ distance
  $\rightarrow$ distance–matrix $D$

$$D_{ij} = \|I_i - I_j\|_2$$
Probability matrix

- Map distances to probabilities

\[ P_{ij} \propto \exp\left(-\frac{D_{i+1,j}}{\sigma}\right) \]

- Normalize so that

\[ \sum_j P_{ij} = 1 \]
Analysis

Video

\[ P_{ij} \propto \exp\left(-\frac{D_{i+1,j}}{\sigma}\right) \]

high \( \sigma \)  
low \( \sigma \)
Preserving Dynamics
Preserving Dynamics

• Optical flow computation
  – bad in untextured areas

• Compare temporally adjacent frames
  – match subsequences instead of single frames
  – filter matrix with a diagonal kernel with weights $w_k$

\[
D'_{ij} = \sum_{k=-m}^{m-1} w_k D_{i+k, j+k}
\]
Preserving Dynamics

$D_{ij}$

swing to the right

$P_{ij}$

swing to the left

$D'_{ij}$

swing to the right

$P'_{ij}$
Dead Ends

- Only window is considered
  - we might get into dead end
  - try to predict future cost
- Anticipated future cost
  - propagate future costs backward

\[
D''_{ij} = (D'_{ij})^p + \alpha \cdot \sum_k P''_{jk} D''_{jk}
\]

with

\[
P''_{ij} \propto \exp\left(- \frac{D''_{i+1,j}}{\sigma}\right)
\]

- Solve by iteration
Dead Ends

- Only window is considered
  → we might get into dead end
  → try to predict future cost
- Anticipated future cost
  - propagate future costs backward

\[ D_{ij}'' = (D_{ij}')^p + \alpha \cdot \min_k D_{jk}'' \]

- Solve with Q-Learning
• Most effective if solved from last to first row
Pruning

• Goal
  – to suppress non-optimal transitions
  – to save storage space

• Paradigms
  1. keep only local maxima
  2. keep those transitions with probabilities above threshold
Synthesis

- Random play
  - begin anywhere but in dead end
  - select next frame according to $P_{ij}$
  - VT is never repeated exactly the same way

- Video loop
  - fixed length loop
  - for conventional digital video players
Video loop

- To combine loops → ranges must overlap
- forward transitions not considered
- Loop length
  \[
  \sum_{i} \#exec_i \cdot \text{length\_of\_loop}_i
  \]
Optimal loop algorithm

- Multiset describes loop with focus to:
  - length
  - range

- Straightforward approach:
  - exponentially in #transitions

- Optimal loop algorithm:
  - select set of transitions (dyn. programming)
  - schedule primitive loops
Select set of transitions

Alg:
- foreach cell
  - if L>=length of transition of current column
  - find legal lowest cost multiset of length L, where the transition of current column appears at least once
## Select set of transitions

### Diagram
- **Alg:**
  - foreach cell
  - if $L \geq$ length of transition of current column
  - find legal lowest cost multiset of length $L$, where the transition of current column appears at least once

### Table
<table>
<thead>
<tr>
<th>L</th>
<th>A(2)</th>
<th>B(3)</th>
<th>C(4)</th>
<th>D(5)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td>B(3)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>BB(6)</td>
<td>D(5)</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Select set of transitions

Alg:
- foreach cell
  - if \( L \geq \) length of transition of current column
  - find legal lowest cost multiset of length \( L \), where the transition of current column appears at least once

<table>
<thead>
<tr>
<th>( L )</th>
<th>A(2)</th>
<th>B(3)</th>
<th>C(4)</th>
<th>D(5)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td><strong>B(3)</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td><strong>BB(6)</strong></td>
<td><strong>D(5)</strong></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Select set of transitions

Alg:
- foreach cell
  - if $L \geq$ length of transition of current column
  - find legal lowest cost multiset of length $L$, where the transition of current column appears at least once
Select set of transitions

Alg:
  • foreach cell
  • if $L \geq$ length of transition of current column
  • find legal lowest cost multiset of length $L$, where the transition of current column appears at least once

<table>
<thead>
<tr>
<th>L</th>
<th>A(2)</th>
<th>B(3)</th>
<th>C(4)</th>
<th>D(5)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td><strong>B(3)</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>BB(6)</td>
<td></td>
<td>D(5)</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>BBB(9)</td>
<td>C(4)</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>BBBB(12)</td>
<td>DD(10)</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td><strong>A(2)</strong></td>
<td><strong>BBBBB(15)</strong></td>
<td>CD(9)</td>
<td>CD(9)</td>
</tr>
<tr>
<td>6</td>
<td><strong>AB(5)</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Select set of transitions

Alg:
- foreach cell
- if L >= length of transition of current column
  - find legal lowest cost multiset of length L, where the transition of current column appears at least once
Select set of transitions

Alg:
- foreach cell
- if $L \geq$ length of transition of current column
  - find legal lowest cost multiset of length $L$, where the transition of current column appears at least once

<table>
<thead>
<tr>
<th>$L$</th>
<th>A(2)</th>
<th>B(3)</th>
<th>C(4)</th>
<th>D(5)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td>B(3)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>BB(6)</td>
<td>D(5)</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>BBB (9)</td>
<td>C(4)</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>BBBB(12)</td>
<td></td>
<td>DD(10)</td>
</tr>
<tr>
<td>5</td>
<td>A(2)</td>
<td>BBBB(15)</td>
<td>CD(9)</td>
<td>CD(9)</td>
</tr>
<tr>
<td>6</td>
<td>AB(5)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Video Textures
Select set of transitions

Alg:
• foreach cell
  • if L>=length of transition of current column
  • find legal lowest cost multiset of length L, where the transition of current column appears at least once

<table>
<thead>
<tr>
<th>L</th>
<th>A(2)</th>
<th>B(3)</th>
<th>C(4)</th>
<th>D(5)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>B(3)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>BB(6)</td>
<td>D(5)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>BBB (9)</td>
<td>C(4)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>BBBB(12)</td>
<td>DD(10)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>A(2)</td>
<td>BBBB(15)</td>
<td>CD(9)</td>
<td>CD(9)</td>
</tr>
<tr>
<td>6</td>
<td>AB(5)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Select set of transitions

Alg:
- foreach cell
  - if $L \geq$ length of transition of current column
  - find legal lowest cost multiset of length $L$, where the transition of current column appears at least once

<table>
<thead>
<tr>
<th>L</th>
<th>A(2)</th>
<th>B(3)</th>
<th>C(4)</th>
<th>D(5)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td>B(3)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>B(3)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td>C(4)</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>B(3)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>A(2)</td>
<td>B(3)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>AB(5)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
**Select set of transitions**

Alg:
- foreach cell
  - if \( L \geq \text{length of transition of current column} \)
  - find legal lowest cost multiset of length \( L \), where the transition of current column appears at least once

<table>
<thead>
<tr>
<th>( L )</th>
<th>A(2)</th>
<th>B(3)</th>
<th>C(4)</th>
<th>D(5)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>B(3)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>BB(6)</td>
<td></td>
<td>D(5)</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>BBB (9)</td>
<td>C(4)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>BBBB(12)</td>
<td>DD(10)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>A(2)</td>
<td>BBBBB(15)</td>
<td>CD(9)</td>
<td>CD(9)</td>
</tr>
<tr>
<td>6</td>
<td>AB(5)</td>
<td>AB(5)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Select set of transitions

Alg:
- foreach cell
  - if $L \geq$ length of transition of current column
  - find legal lowest cost multiset of length $L$, where the transition of current column appears at least once

- time $O(L^2N^2)$
- space $O(LN)$
- optionally store pointers + range instead of full description

<table>
<thead>
<tr>
<th>$L$</th>
<th>A(2)</th>
<th>B(3)</th>
<th>C(4)</th>
<th>D(5)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td>B(3)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>BB(6)</td>
<td></td>
<td>D(5)</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>BBB(9)</td>
<td>C(4)</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>BBBB(12)</td>
<td>DD(10)</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>A(2)</td>
<td>BBBB(15)</td>
<td>CD(9)</td>
<td>CD(9)</td>
</tr>
<tr>
<td>6</td>
<td>AB(5)</td>
<td>AB(5)</td>
<td>CC(8)</td>
<td>DDD(15)</td>
</tr>
</tbody>
</table>
Schedule primitive loops

Alg:

Given:
- length = 17
- transitions ABCDD occur

current schedule:
Schedule primitive loops

Alg:
- Schedule transition $i \rightarrow j$ that starts at the end
- 1 or more ranges

Given:
- length = 17
- transitions ABCDD occur

current schedule:
A

invariant: always in first range
Schedule primitive loops

Alg:
• Schedule transition $i \rightarrow j$ that starts at the end
  $\rightarrow$ 1 or more ranges
• Schedule any* transition in first range that starts after $j$

invariant: always in first range

* deterministic or stochastic

Given:
• length = 17
• transitions ABCDD occur

current schedule:
A D
Schedule primitive loops

Alg:
- Schedule transition $i \rightarrow j$ that starts at the end
  - 1 or more ranges
- Schedule any* transition in first range that starts after $j$

Invariant: always in first range

* deterministic or stochastic

Given:
- length = 17
- transitions ABCDD occur

current schedule:
A D C
Schedule primitive loops

Alg:
- Schedule transition $i \rightarrow j$ that starts at the end
  $\rightarrow$ 1 or more ranges
- Schedule any* transition in first range that starts after $j$

invariant: always in first range

* deterministic or stochastic

Given:
- length = 17
- transitions ABCDD occur

current schedule:

A D C D
Schedule primitive loops

Alg:

- Schedule transition $i \rightarrow j$ that starts at the end
  - $1$ or more ranges
- Schedule any* transition in first range that starts after $j$

Invariant: always in first range

* deterministic or stochastic

Given:

- length = 17
- transitions ABCDD occur

Current schedule:

A D C D B
Rendering

- Jump cut
  - noticeable transitions
- Cross-fade
  - blur if misaligned
- Morph
  - common features get aligned
Rendering

- Multi-way cross-fade
  - multiple frames displayed at a time
  - weighted average of all participating frames
Outline

• Introduction
• Representation
• MAIN SYSTEM
• Extensions
• Results
• Future Work & Discussion
3D Video Texture

- replace video with video texture
- create novel view
Motion factorization

- Divide original video into independently moving parts
  - manual
  - automatic
Video–based Animation

- Control where, how & how long to replay
- Add user–controlled term to error function based on:
  - velocity of video sprite
  - position of frame
Video-based Animation
Video Sprites

1. Object extraction (bg subtraction, blue screen)
2. Store velocity of centroid
3. Center object

→ Analysis + alpha, direction, speed
→ Synthesis + velocities
Video Sprites

Video Textures
Results

- Simple algorithm
- Several extensions & applications
- User control
- Combine video textures
Results

- Algorithm fails at complex and high-structured video material
Future Work & Discussion

- Distance metrics
- Blending
- Variety
- Control tools