Temporally Coherent Stylization of 3D Animations

Pierre Bénard

Université de Bordeaux
Pixar Style
New Styles?
New Styles?
New Styles?
New Styles?
New Styles?
New Styles?
New Styles?
Options for stylization

Tailored rendering algorithms

Dynamic Noise [Bénard et al. 2010]
Options for stylization

Tailored rendering algorithms

Dynamic Noise [Bénard et al. 2010]
Options for stylization

Tailored rendering algorithms

Active Strokes [Bénard et al. 2012]
Options for stylization

Tailored rendering algorithms

Active Strokes [Bénard et al. 2012]
Options for stylization

Tailored rendering algorithms

Custom painting system

Overcoat [Schmid et al. 2011]
Options for stylization

Tailored rendering algorithms

Custom painting system

Overcoat [Schmid et al. 2011]
Options for stylization

Tailored rendering algorithms

Custom painting system

Painterly Characters Animation [Basset et al. 2014]
Options for stylization

Tailored rendering algorithms

Custom painting system

WYSIWYG NPR [Kalnis et al. 2002]
Options for stylization

Tailored rendering algorithms

Custom painting system

Synthesis by example

Image Analogies [Hertzmann et al. 2001]
Options for stylization

Tailored rendering algorithms

Custom painting system

Synthesis by example

The Markov Pen [Lang et al. 2015]
Temporal Coherence Problem

2D frame-by-frame animation

Jeremy Depuydt, "Il pleut bergère"
Temporal Coherence Problem

3D animations: Screen space mapping
Temporal Coherence Problem

3D animations: Object space mapping
Temporal Coherence Problem

2D Appearance

Motion Coherence

Temporal Continuity
Temporal Coherence Problem

2D Appearance

Static marks

Motion Coherence

Temporal Continuity

Temporal Coherence Problem

2D Appearance

Motion Coherence  Texture mapping  Temporal Continuity

Temporal Coherence Problem

2D Apparence

Random changes

Motion Coherence

Temporal Continuity
CG rendered animation
Style Example

Simple CG rendering

Artist’s painting

Martin Sebastian Senn
CG rendering

Artist’s keyframes
one every 10 - 15 frames
Stylized Result
Image Analogies

Style input $S$  :  Style output $\hat{S}$  :  Input image $I$  :  Output image $\hat{I}$

after [Hertzmann et al. 2001]
Image Analogies

Style input $S$ :: Style output $\hat{S}$ :: Input image $I$ :: Output image $\hat{I}$

after [Hertzmann et al. 2001]
Image Analogies

Style input $S$  Style output $\hat{S}$  Input image $I$  Output image $\hat{I}$

after [Hertzmann et al. 2001]
Image Analogies

\[ \hat{i}(p) = \hat{S}(M(p)) \]

after [Hertzmann et al. 2001]
Image Analogies

\[ \hat{i}(p) = \hat{S}(M(p)) \]

after [Hertzmann et al. 2001]
Goals for $M(p)$

- Every neighborhood $\Omega(p)$ in $\hat{I}$ matches its corresponding neighborhood $M(\Omega(p))$ in $\hat{S}$
Goals for $M(p)$

- Every neighborhood $\Omega(p)$ in $\hat{I}$ matches its corresponding neighborhood $M(\Omega(p))$ in $\hat{S}$

- Every neighborhood $\Omega(p)$ in $I$ matches its corresponding neighborhood $M(\Omega(p))$ in $S$
Goals for $M(p)$

• Every neighborhood $\Omega(p)$ in $\hat{S}$ matches its corresponding neighborhood $M(\Omega(p))$ in $\hat{S}$.

• Every neighborhood $\Omega(p)$ in $S$ matches its corresponding neighborhood $M(\Omega(p))$ in $S$.

• $M(p)$ is *continuous*.
Optimization of $M(p)$

PatchMatch
[Barnes et al. 2009]

~5 minutes per frame
(1920×1080 pixels, NVIDIA Quadro 500)
Input Animation
Style pair
Independent synthesis per frame
Temporal Coherence Goals

\( \hat{I}_t \)
Temporal Coherence Goals

$\hat{l}_{t-1}$

$\hat{l}_t$
Temporal Coherence Goals

\[ \hat{I}_{t-1} \rightarrow p + V_t^- (p) \rightarrow \hat{I}_t \]
Temporal Coherence Goals

\[ \hat{I}_{t-1} \xrightarrow{p + V_t^- (p)} \hat{I}_t \xrightarrow{p} \hat{I}_{t+1} \]
Temporal Coherence Goals

\[ \hat{l}_{t-1}, \hat{l}_t, \hat{l}_{t+1} \]

\[ p + V_{t-}^-(p) \]

\[ p + V_{t+}^+(p) \]
Disocclusions leave trails

frame 0

frame 10

ignoring disocclusions

frame 10
With temporal coherence
Hatching style
Input animation
Keyframe Setup

Input
Keyframe Setup

Input

Painted

keyframe 1
Keyframe Setup

Input

Painted keyframe 1

...
Handling Overdraw
Handling Overdraw

Input mask for earmuffs
Layer Decomposition

Input

Artist’s painting
Keyframes
Keyframes
Keyframes
Keyframes
Keyframes
Keyframes
Keyframes
Keyframes
Keyframes
Keyframes
Keyframes
Keyframes
Keyframes
Synthesis result

(half speed)
Limitations and Future Work

Smooother color transitions
e.g., [Darabi et al. 2012]
Limitations and Future Work

Smooter color transitions
e.g., [Darabi et al. 2012]

Better analogy inputs

Output

CG input

Artist’s keyframe
Follow-up work: Fluid Stylization

“Stylized Keyframe Animation of Fluid Simulations” [Browning et al. 2014]
Follow-up work: Temporal Noise Control

“Color Me Noisy” [Fišer et al. 2014]
Stylizing Animation by Example

P. Bénard, F. Cole, M. Kass, I. Mordatch, J. Hegarty, M.-S. Senn, K. Fleischer, D. Pesare, K. Breeden
Pixar Animation Studios
University of Toronto, University of Washington, Stanford University

Computing Smooth Surface Contours with Accurate Topology

P. Bénard, A. Hertzmann, M. Kass
University of Toronto, Université de Bordeaux, Adobe Research, Pixar Animation Studios
Smooth surface
[Catmull and Clark 1978]
Smooth surface
[Catmull and Clark 1978]

Occluding contours
Smooth surface
[Catmull and Clark 1978]

Occluding contours

Stylized rendering
Definition: for a smooth surface, points at which $n \cdot v = 0$
Occluding Contours

**Definition:** for a smooth surface, points at which \( \mathbf{n} \cdot \mathbf{v} = 0 \)
Definition: for a smooth surface, points at which $\mathbf{n} \cdot \mathbf{v} = 0$

- No closed-form expression [Elber and Cohen 1990]
- Unstable visibility (boundary between visible and invisible)
Mesh Contours

Definition: for a discrete surface, edges between front- and back-faces

→ Widely used in practice (simple and fast)
Stylized Mesh Contours

Spatial and temporal artifacts (gaps, breaks)

Stylization based on “Coherent Stylized Silhouettes” [Kalnins et al. 2003]
Mesh Contours

**Issue:** Overly complex topology (loops)

Side view
Mesh Contours

**Issue:** Overly complex topology (loops)
Mesh Contours

Issue: Overly complex topology (loops)
Issue: Surface-contour visibility mismatched

Interpolated Contours

[Hertzmann and Zorin 2000]
Interpolated Contours

Issue: Surface-contour visibility mismatched

[Hertzmann and Zorin 2000]
Interpolated Contours

Issue: Surface-contour visibility mismatched

[Hertzmann and Zorin 2000]
Our Method

**Goal:** Mesh contours *partitioning* front- and back-faces

Side view
Our Method

**Goal:** Mesh contours *partitioning* front- and back-faces

Side view
Goal: Mesh contours partitioning front- and back-faces

Our Method

Camera view
(one color per 2D curve)
Our Method

**Goal**: extract for each viewpoint a mesh $M$ whose contours are equivalent to the contours of $S$. 
Our Method

**Goal:** extract for each viewpoint a mesh $M$ whose contours are equivalent to the contours of $S$. 

![Diagram showing the comparison between $S$ and $M$.](image)
Goal: extract for each viewpoint a mesh $M$ whose contours are equivalent to the contours of $S$.

Difficult, global property
Our Method

Goal: extract for each viewpoint a mesh $M$ whose contours are equivalent to the contours of $S$.

Difficult, global property

$\Rightarrow$ local criterion: Contour-Consistency
Contour-Consistency

Determines if triangle’s orientation matches smooth surface’s orientation
Contour-Consistency

Determines if triangle’s orientation **matches** smooth surface’s orientation
Contour-Consistency

Determines if triangle’s orientation *matches* smooth surface’s orientation

FFF = *front-facing*  
⇒ *Contour-Consistent*
Contour-Consistency

Determines if triangle’s orientation matches smooth surface’s orientation

BBB = back-facing

Contour-Consistent
Contour-Consistency

Determines if triangle’s orientation matches smooth surface’s orientation

FFB ➡ Not Contour-Consistent
Contour insertion

Insert *contour edges* into the mesh.
Contour insertion

Insert **contour edges** into the mesh.
Contour insertion

Insert **contour edges** into the mesh.
Contour insertion

Insert **contour edges** into the mesh.
Contour insertion

Insert *contour edges* into the mesh.

![Diagram showing contour insertion](image)

**BBC ≠ front-facing**

⇒ *not* Contour-Consistent
Contour insertion

Insert **contour edges** into the mesh.
Contour insertion

Insert contour edges into the mesh.
Inconsistent Triangle

Intuition for a simple case (hemisphere)
Inconsistent Triangle

Intuition for a simple case (hemisphere)
Intuition for a simple case (hemisphere)
Intuition for a simple case (hemisphere)

**Inconsistent Triangle**

Camera view

Side view
Radial Triangle

Triangle with one vertex $p$ on the contour, and one vertex $r$ on the radial plane of $p$.
Radial Triangle

Triangle with one vertex $p$ on the contour, and one vertex $r$ on the radial plane of $p$

$\rightarrow$ Triangle $pqr$ guaranteed to be Contour-Consistent
Adaptive Tessellation Algorithm

1. Smooth surface
   [Catmull and Clark 1978]

2. Initial mesh

3. Contour insertion
Adaptive Tessellation Algorithm

1. Smooth surface
   [Catmull and Clark 1978]
2. Initial mesh
3. Contour insertion
Adaptive Tessellation Algorithm

1. Smooth surface  
   [Catmull and Clark 1978]

2. Initial mesh

3. Contour insertion
Adaptive Tessellation Algorithm

1. Smooth surface
   [Catmull and Clark 1978]

2. Initial mesh

3. Contour insertion
Adaptive Tessellation Algorithm

4. “Radialization”

5. Optimization

6. Contours
Adaptive Tessellation Algorithm

4. “Radialization”
5. Optimization
6. Contours
Adaptive Tessellation Algorithm

4. “Radialization”
5. Optimization
6. Contours
Adaptive Tessellation Algorithm

4. “Radialization”
5. Optimization
6. Contours
Comparison

Stylization based on [Kalnins et al. 2003]

Mesh contours

Interpolated contours

Our method
Our method
(composite with cartoon colors)
Limitations

Contour-Consistency guaranteed almost everywhere

⇒ Algorithm fully ensuring consistency
Limitations

Contour-Consistency guaranteed almost everywhere
⇒ Algorithm fully ensuring consistency

Computationally expensive (a few minutes per image)
⇒ Real-Time GPU tessellation
What’s next?

**Full geometrically accurate** line rendering pipeline, including:

- Advanced 2D simplification (e.g., based on line density [Grabli et al. 2010])
- Temporally coherent stylization [Buchholz et al. 2011; Bénard et al. 2012]
- Other line definitions (suggestive contours [DeCarlo et al. 2003])
What’s next?

Full geometrically accurate line rendering pipeline, including:

- Advanced 2D simplification (e.g., based on line density [Grabli et al. 2010])
- Temporally coherent stylization [Buchholz et al. 2011; Bénard et al. 2012]
- Other line definitions (suggestive contours [DeCarlo et al. 2003])

Stylized rendering of “regular” 3D animations not sufficient

→ Move toward stylized animation (e.g., motion, deformation)
Thanks!

Source code: https://github.com/benardp/PaintTween
contours

Research
Pierre Bénard
Igor Mordatch
Forrester Cole
James Hegarty
Davide Pesare
Michael Kass
Martin Senn
Katherine Breeden
Aaron Hertzmann
Kurt Fleischer

Art Production
Teddy Newton
Andrew H. Schmidt
Danny Nahmias
Paul Aichele
Don Shank
Brian Tindall
Mach Kobayashi
Samuel Lehmer
Sanjay Patel
Jamie Frye
Bernard Haux
Ryan Dale
Chris Landreth

Special thanks to
Tony DeRose, Sue Kalache, Mark Meyer, Fernando DeGoes,
Joëlle Thollot, Ralph Eggleston, Angus MacLane, Guido Quaroni,
Ed Catmull, John Lasseter and Jim Morris
Thanks!

Source code: https://github.com/benardp/PaintTween/contours

Research
Pierre Bénard
Igor Mordatch
Forrester Cole
James Hegarty
Davide Pesare
Michael Kass
Martin Senn
Katherine Breeden
Aaron Hertzmann
Kurt Fleischer

Art Production
Teddy Newton
Andrew H. Schmidt
Danny Nahmias
Paul Aichele
Don Shank
Brian Tindall
Mach Kobayashi
Samuel Lehmer
Sanjay Patel
Jamie Frye
Bernard Haux
Ryan Dale
Chris Landreth

Special thanks to
Tony DeRose, Sue Kalache, Mark Meyer, Fernando DeGoes,
Joëlle Thollot, Ralph Eggleston, Angus MacLane, Guido Quaroni,
Ed Catmull, John Lasseter and Jim Morris