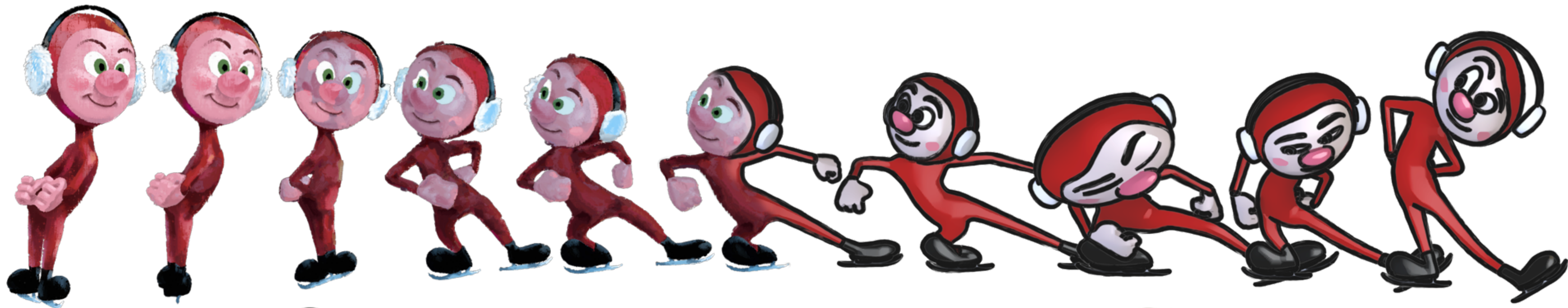


Temporally Coherent Stylization of 3D Animations

Pierre Bénard

Université de Bordeaux



Pixar Style



New Styles?



New Styles?



New Styles?



New Styles?



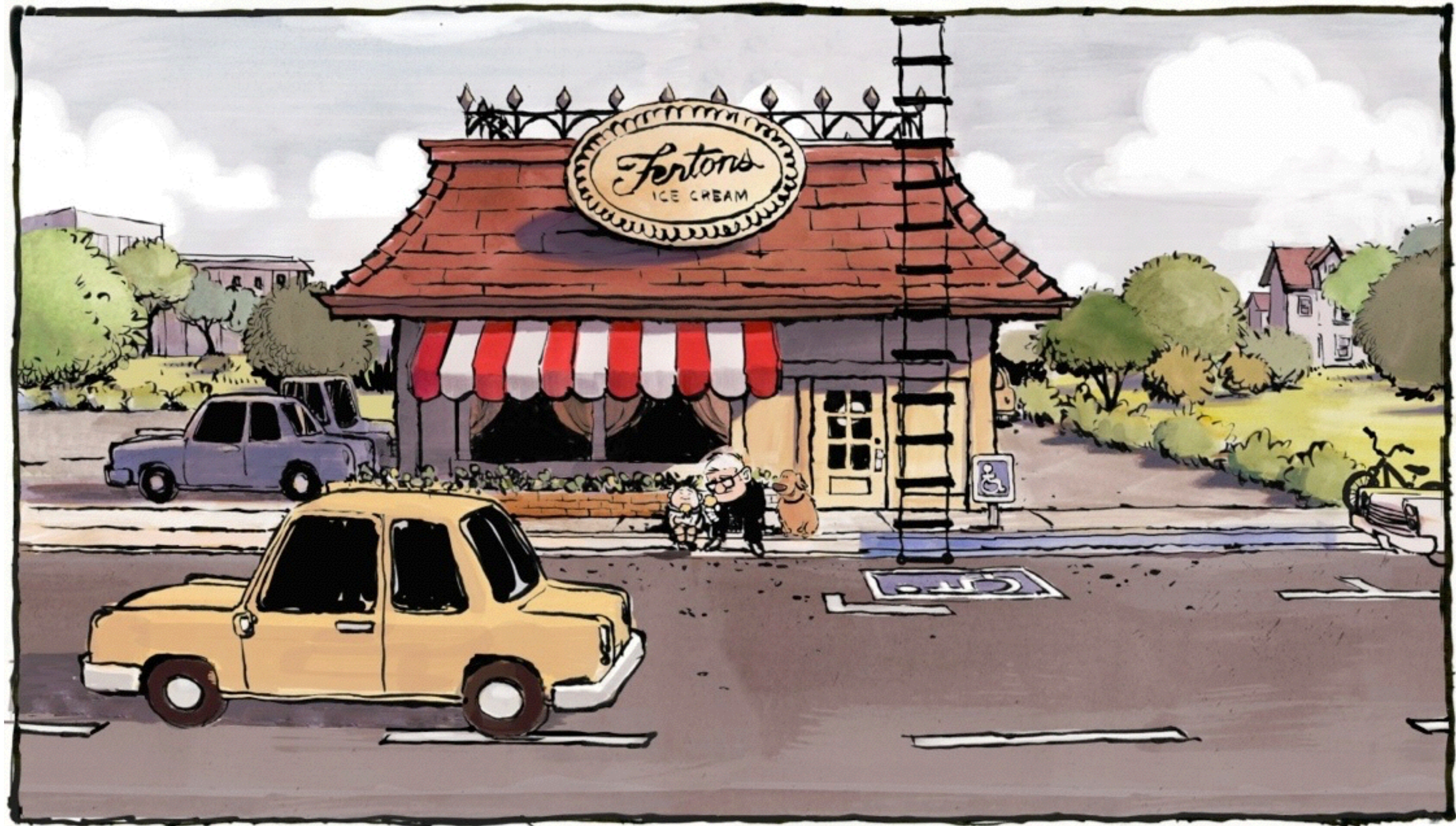
New Styles?



New Styles?

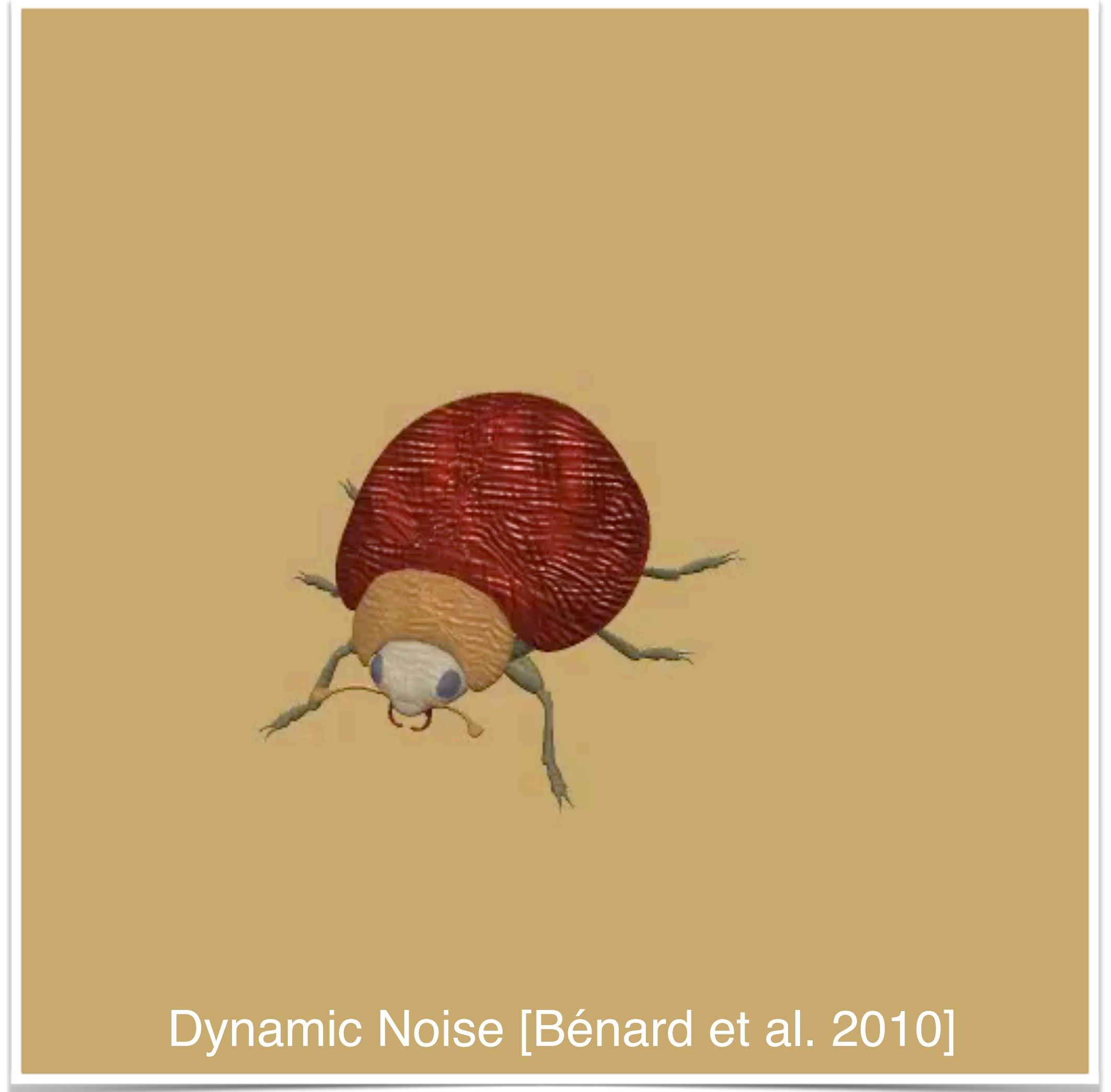


New Styles?



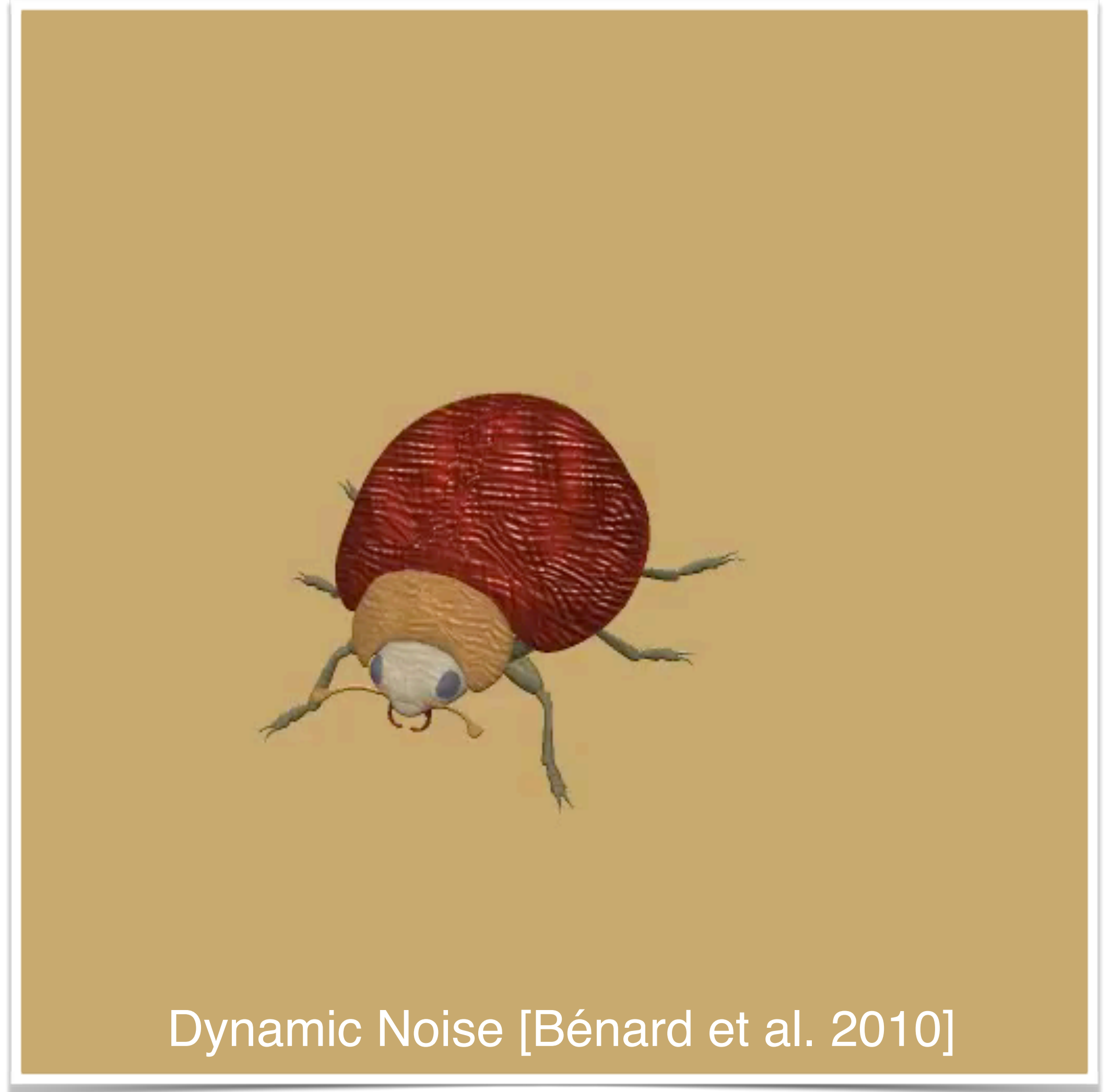
Options for stylization

Tailored rendering algorithms



Options for stylization

Tailored rendering algorithms



Options for stylization

Tailored rendering algorithms



Options for stylization

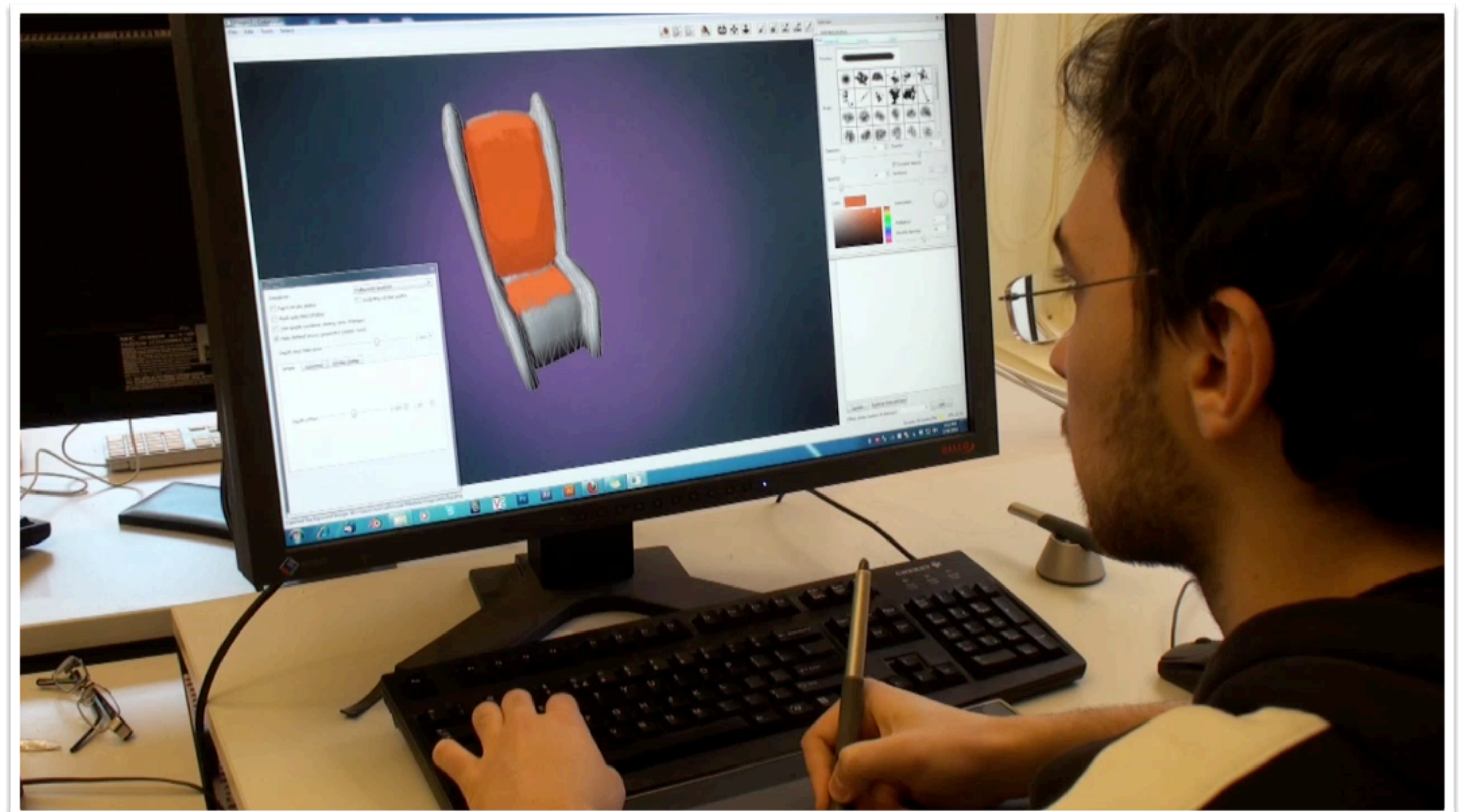
Tailored rendering algorithms



Options for stylization

Tailored rendering algorithms

Custom painting system

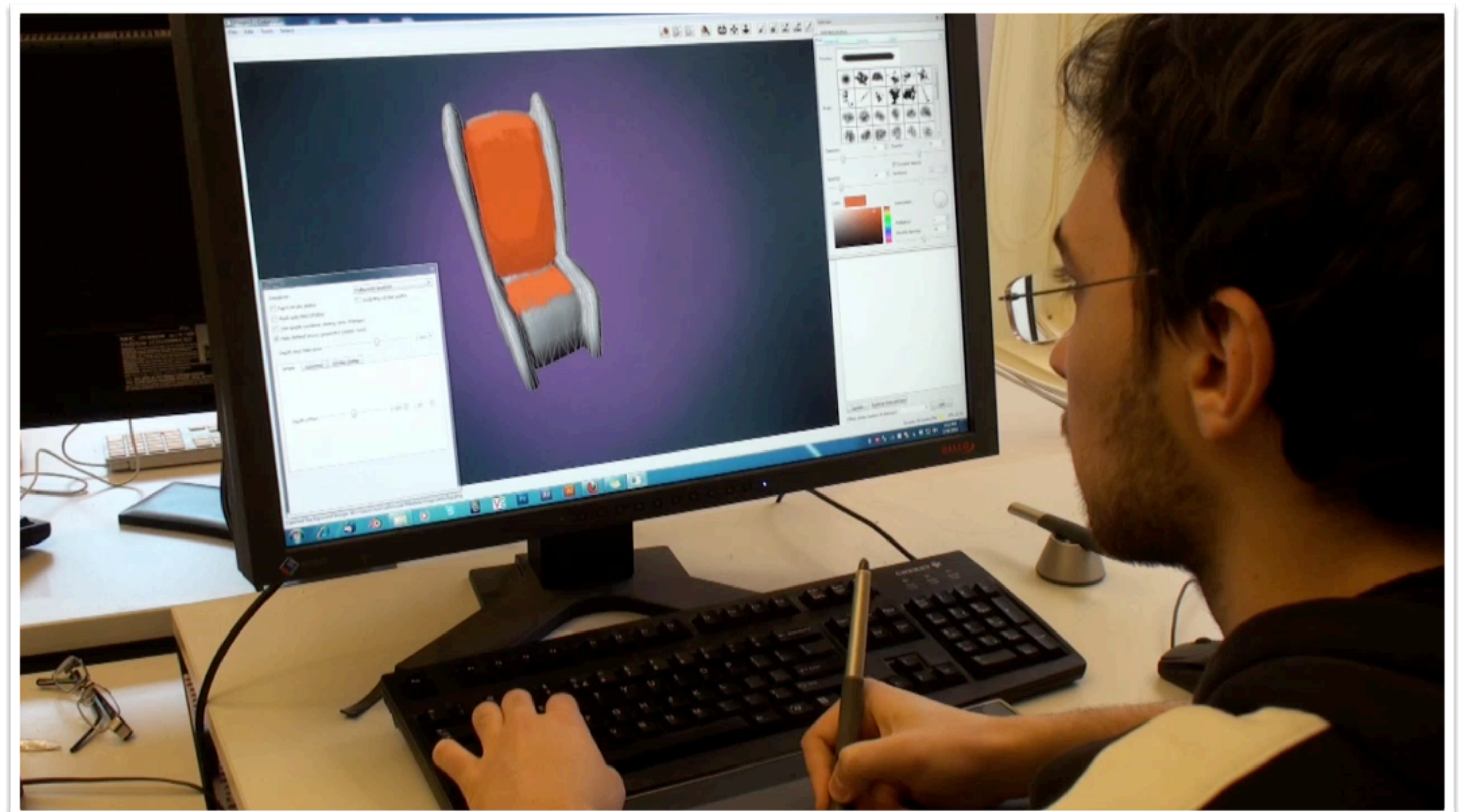


Overcoat [Schmid et al. 2011]

Options for stylization

Tailored rendering algorithms

Custom painting system



Overcoat [Schmid et al. 2011]

Options for stylization

Tailored rendering algorithms

Custom painting system

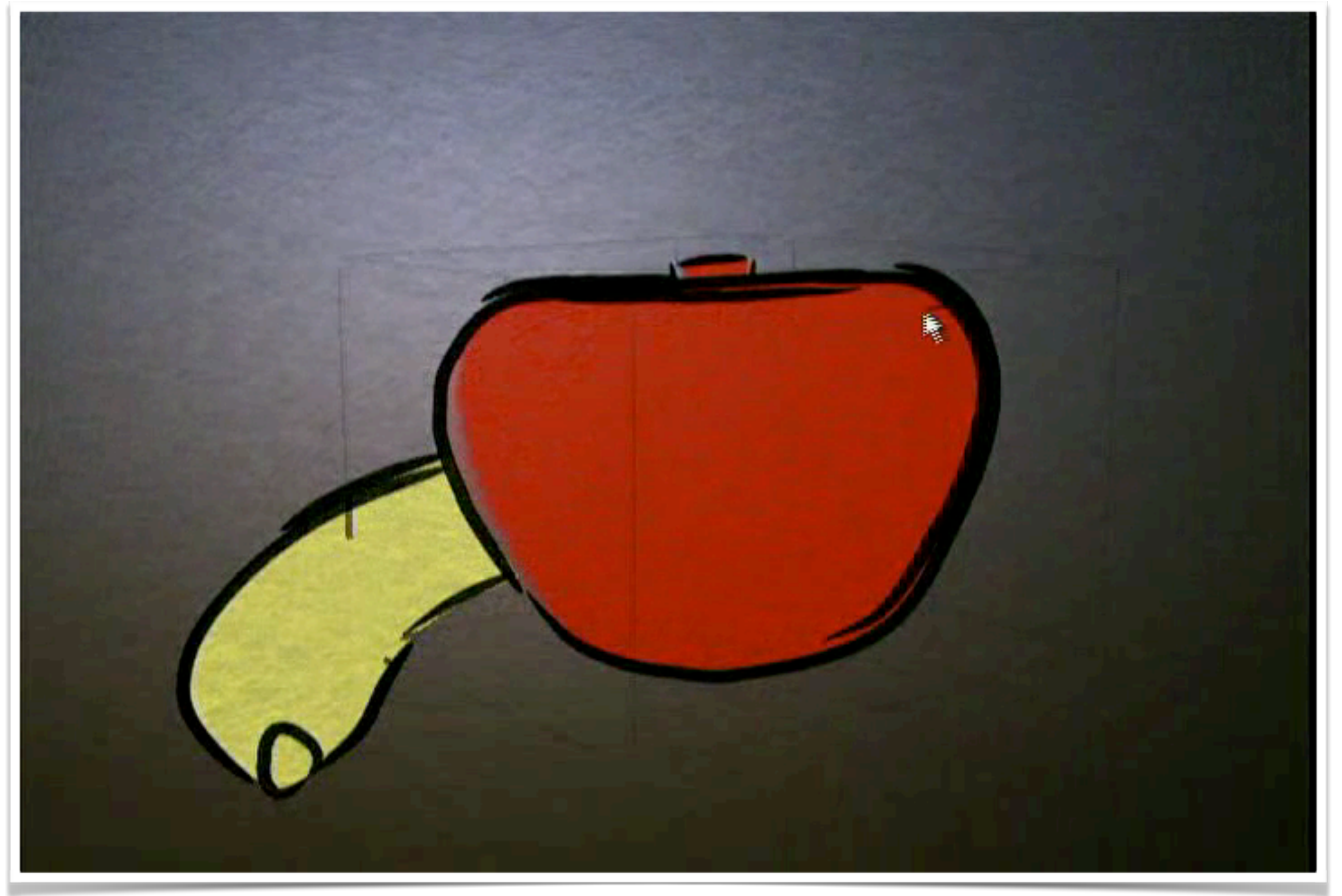


Painterly Characters Animation [Basset et al. 2014]

Options for stylization

Tailored rendering algorithms

Custom painting system



WYSIWYG NPR [Kalnins et al. 2002]

Options for stylization

Tailored rendering algorithms

Custom painting system

Synthesis by example



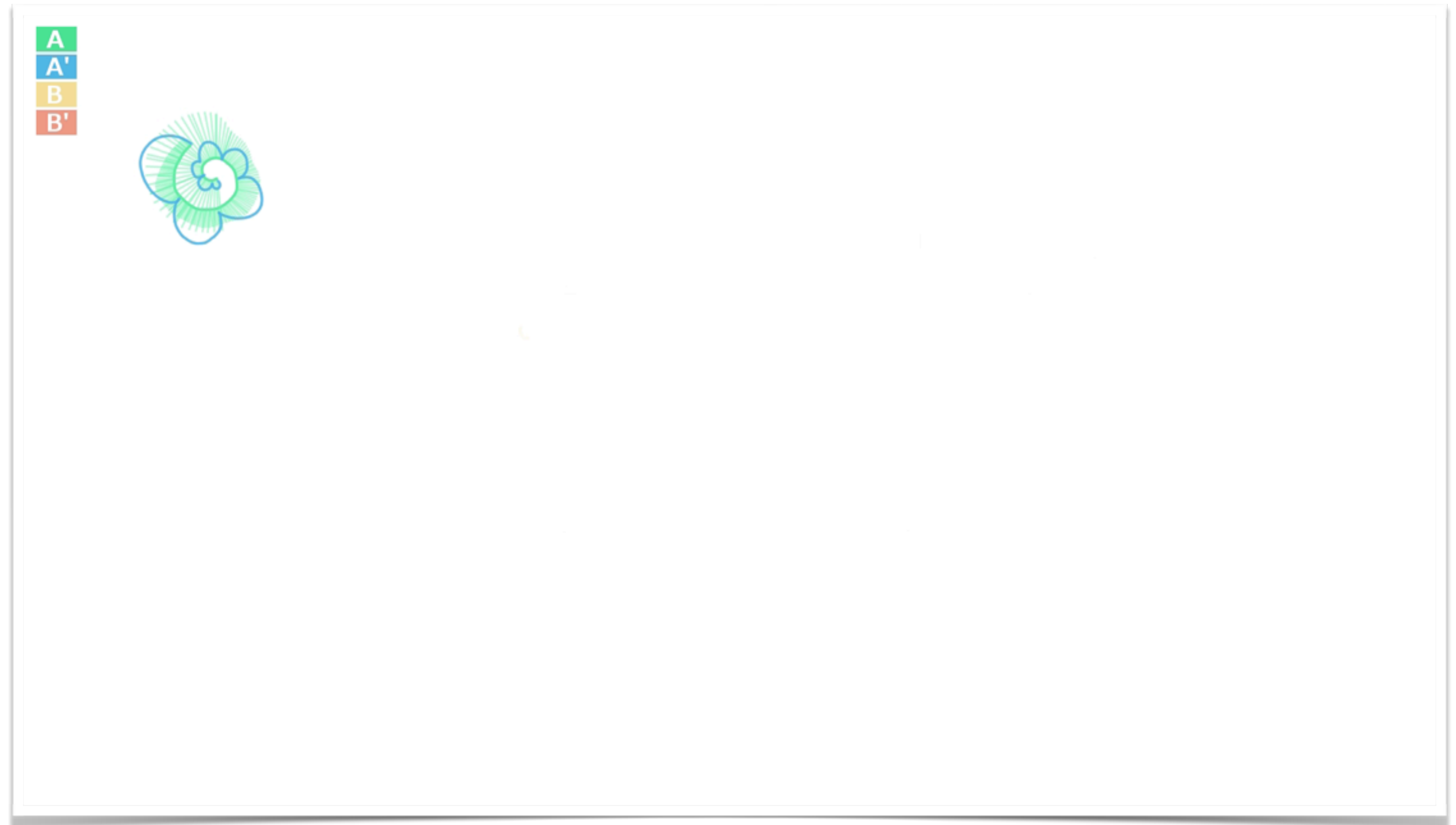
Image Analogies [Hertzmann et al. 2001]

Options for stylization

Tailored rendering algorithms

Custom painting system

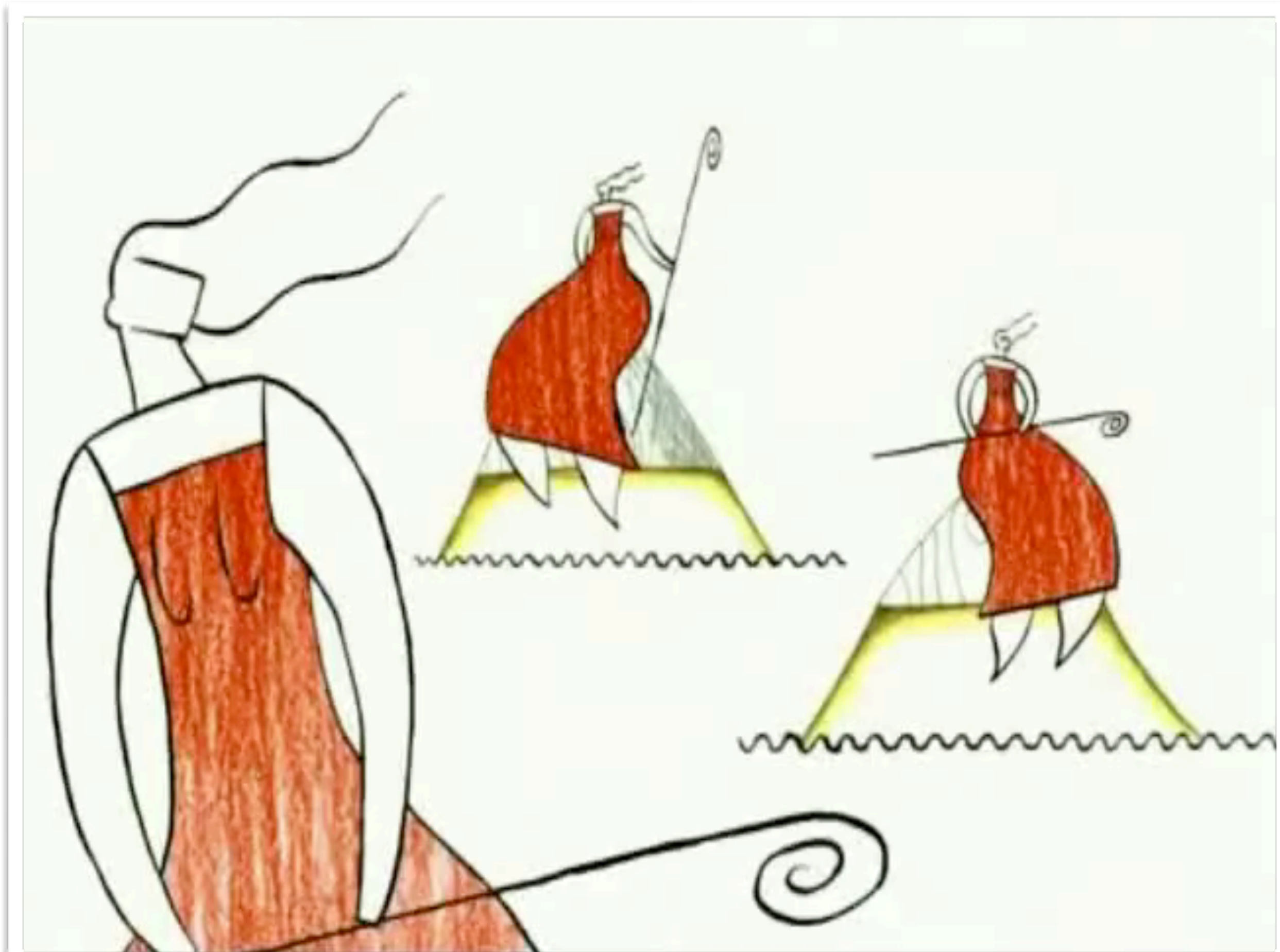
Synthesis by example



The Markov Pen [Lang et al. 2015]

Temporal Coherence Problem

Jeremy Depuydt, "Il pleut bergère"



2D frame-by-frame animation

Temporal Coherence Problem



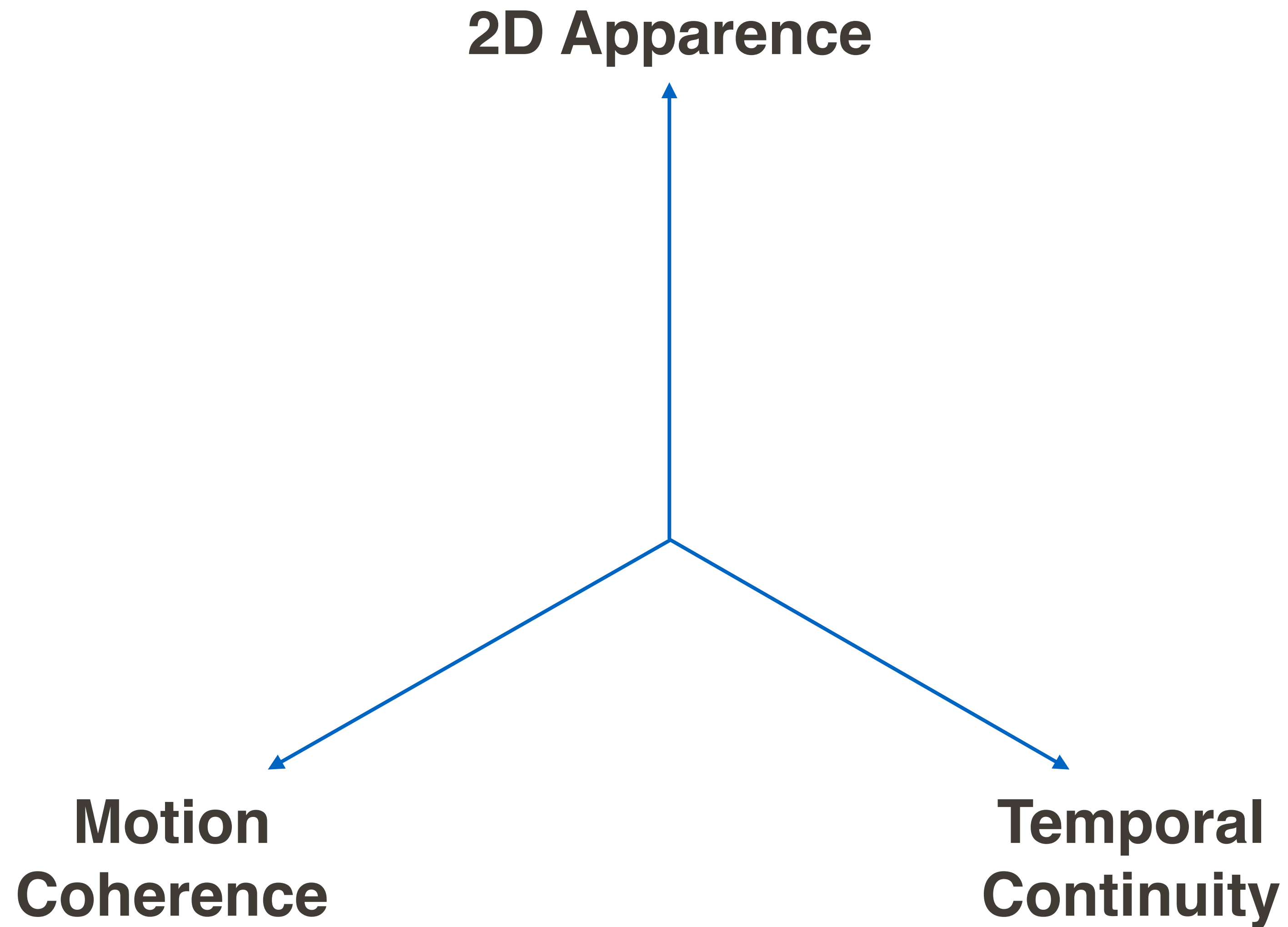
3D animations: Screen space mapping

Temporal Coherence Problem

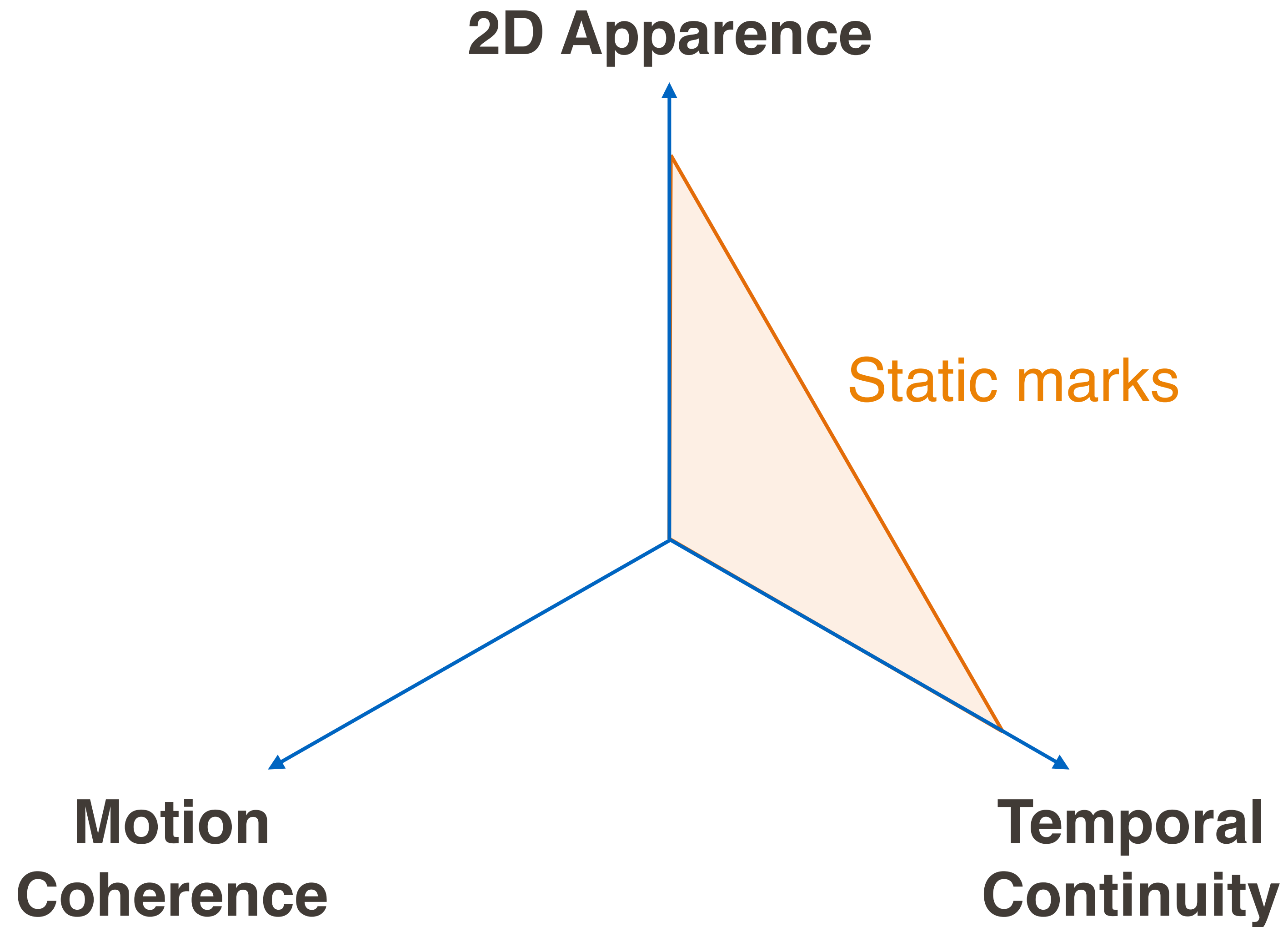


3D animations: Object space mapping

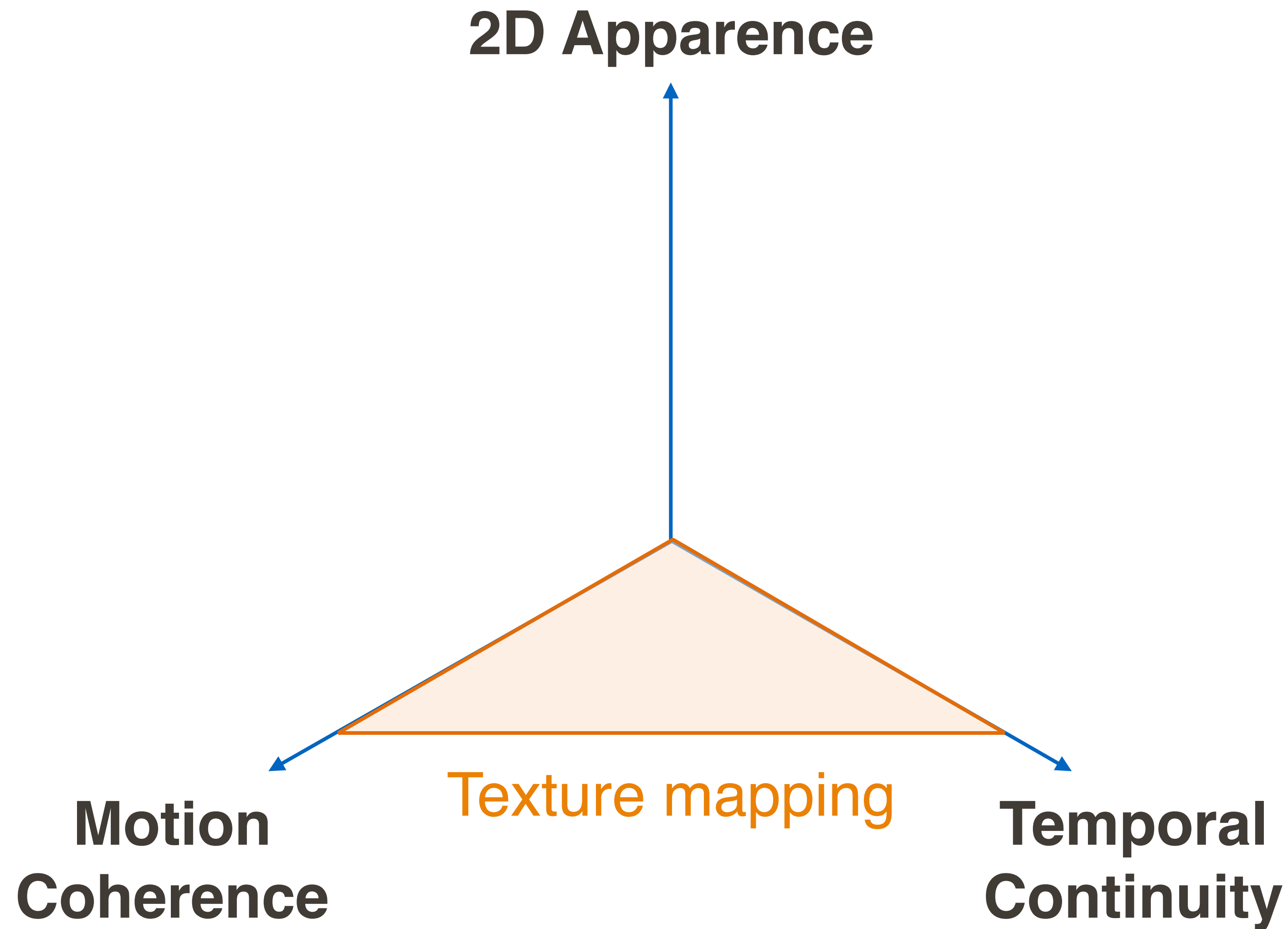
Temporal Coherence Problem



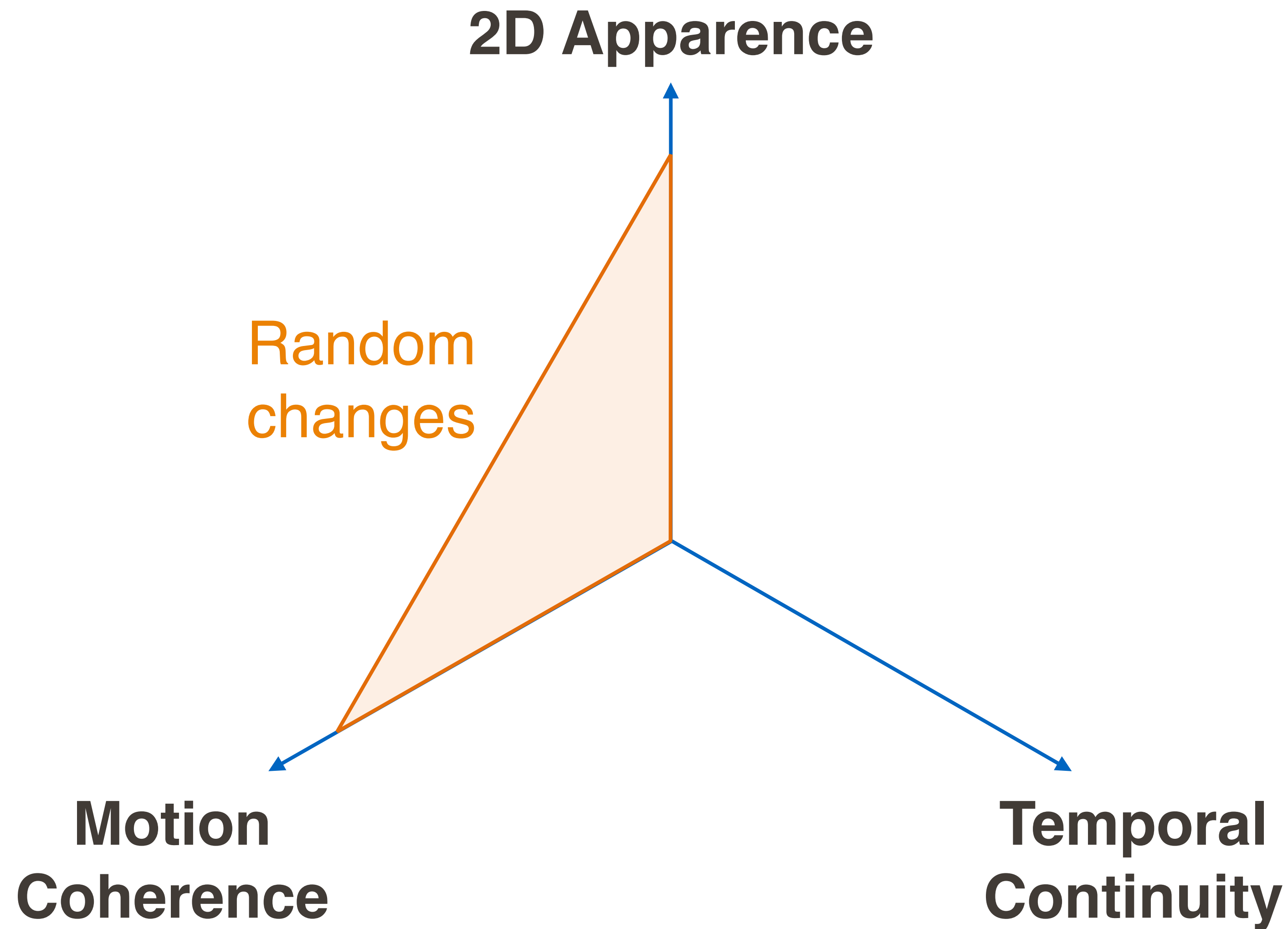
Temporal Coherence Problem



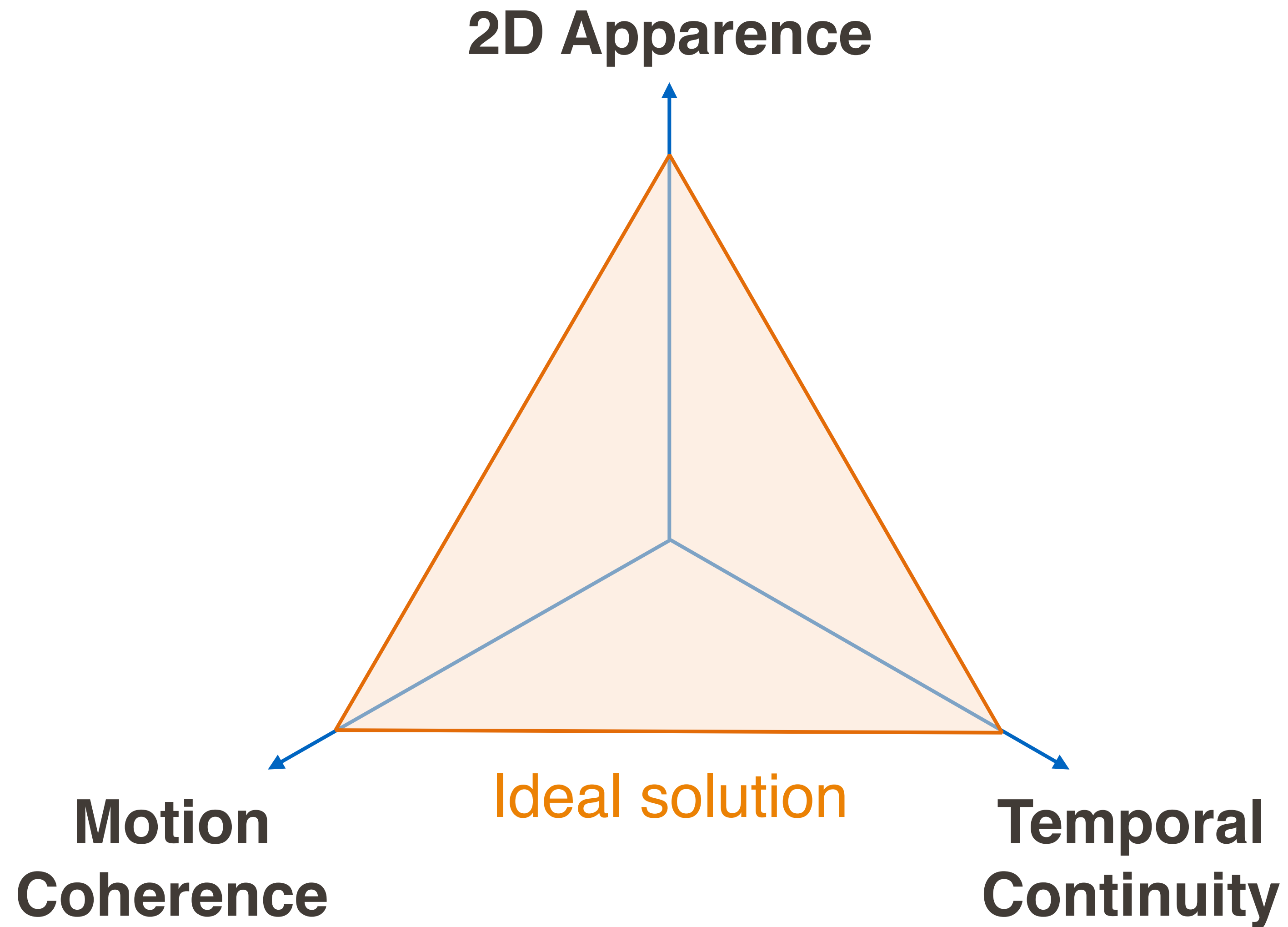
Temporal Coherence Problem



Temporal Coherence Problem



Temporal Coherence Problem



P. B  nard, F. Cole, M. Kass,
I. Mordatch, J. Hegarty, M.-S. Senn,
K. Fleischer, D. Pesare, K. Breeden

Pixar Animation Studios

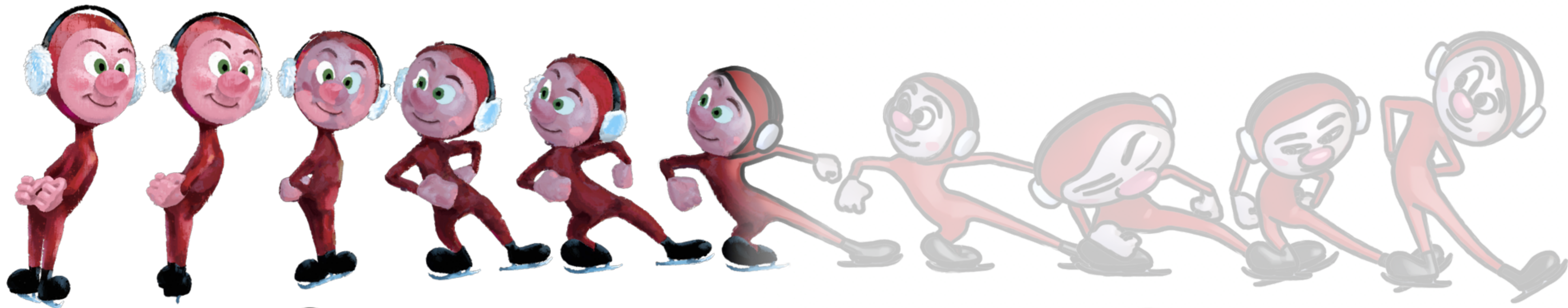
University of Toronto, University of Washington, Stanford University

Stylizing Animation by Example

P. B  nard, A. Hertzmann, M. Kass

*University of Toronto, Universit   de Bordeaux,
Adobe Research, Pixar Animation Studios*

Computing Smooth Surface Contours with Accurate Topology



CG rendered animation

Style Example



Simple CG rendering



Martin Sebastian Senn

Artist's painting

CG rendering



Artist's
keyframes

one every 10 - 15 frames



Stylized Result

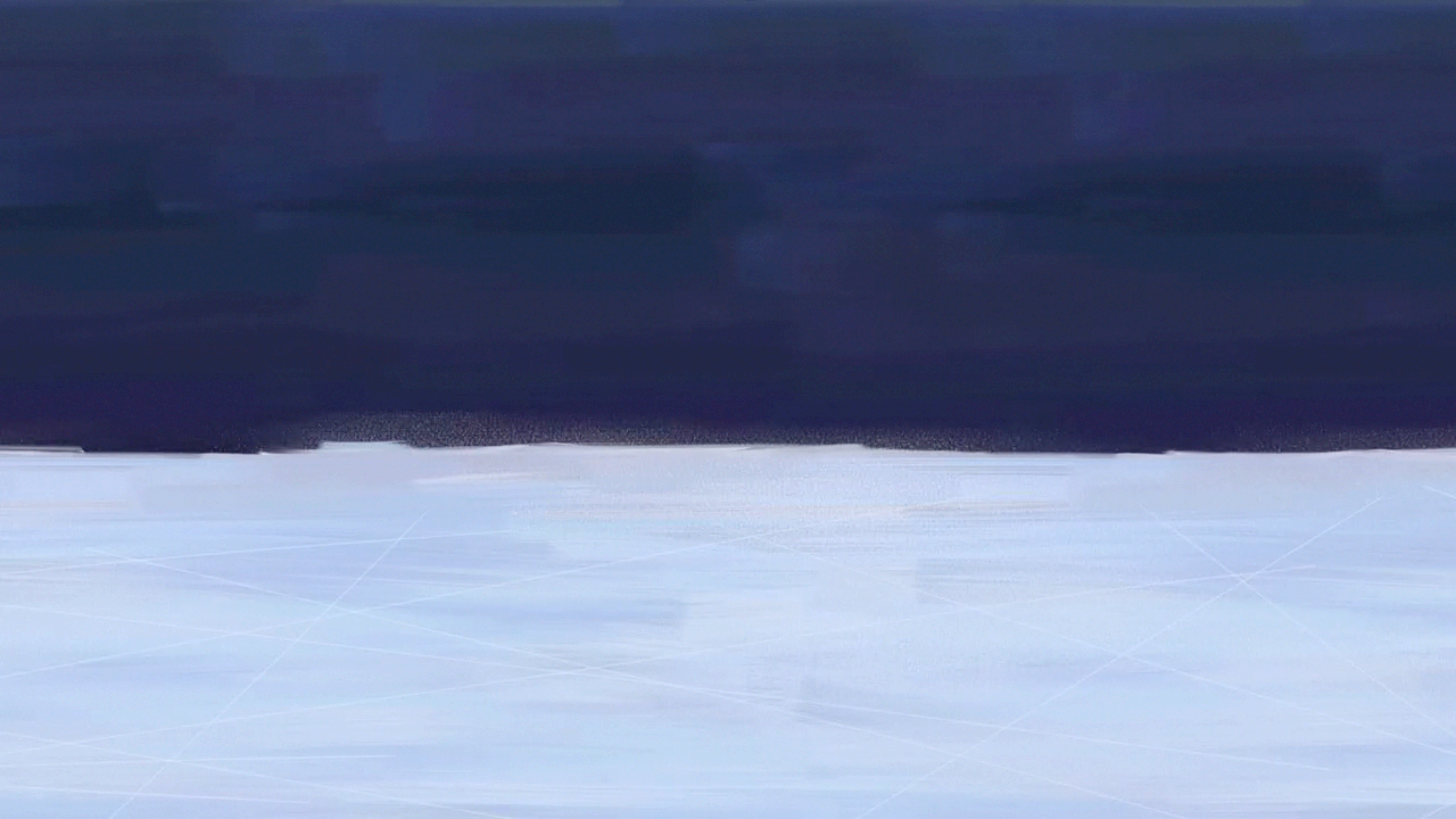


Image Analogies



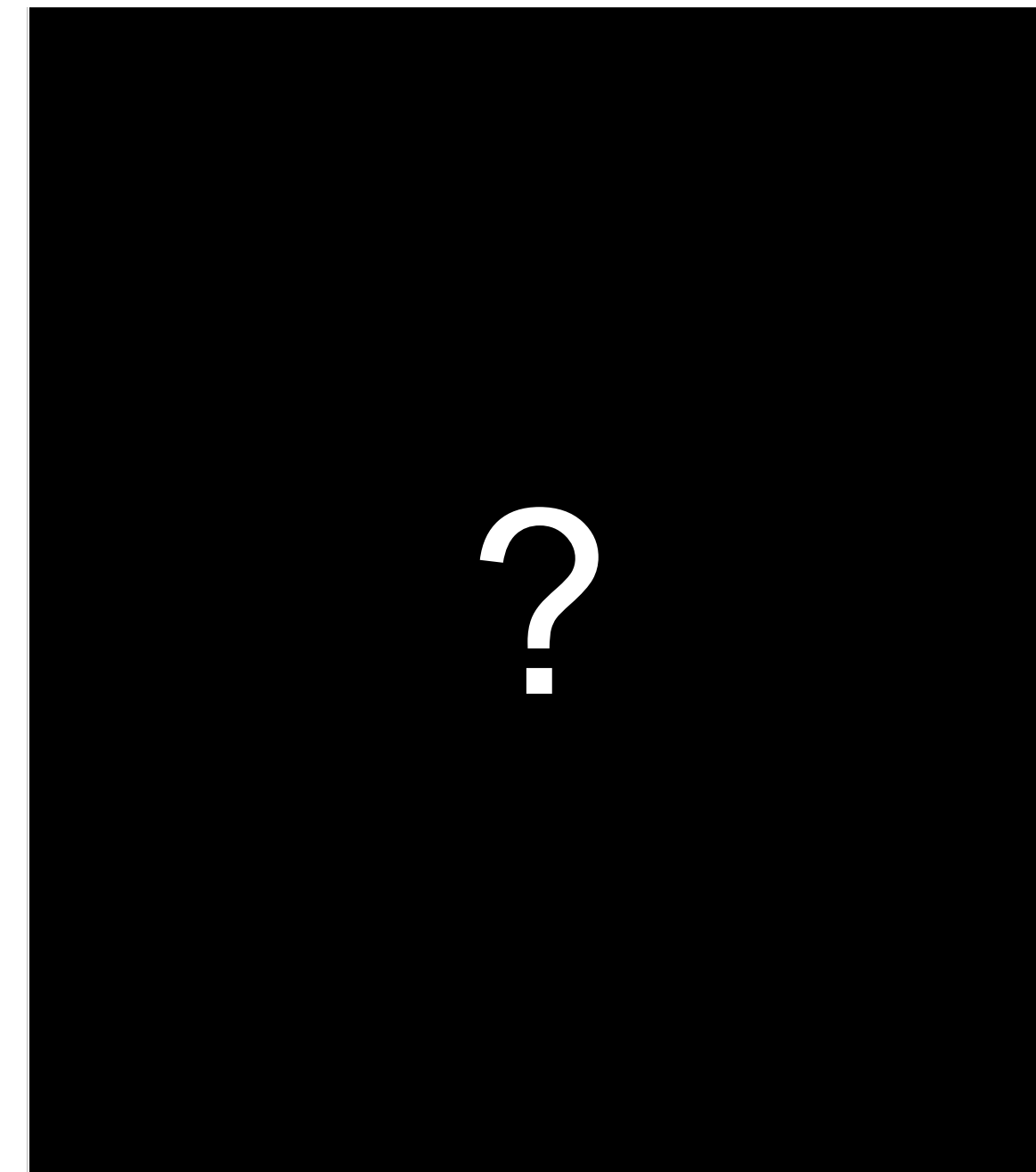
■
■



■ ■
■ ■



■
■



Style input S

Style output \hat{S}

Input image I

Output image \hat{I}

Image Analogies



■
■



■ ■
■ ■



■
■



Style input S Style output \hat{S}

Input image I Output image \hat{I}

Image Analogies



■
■



■ ■
■ ■



■
■



Style input S

Style output \hat{S}

Input image I

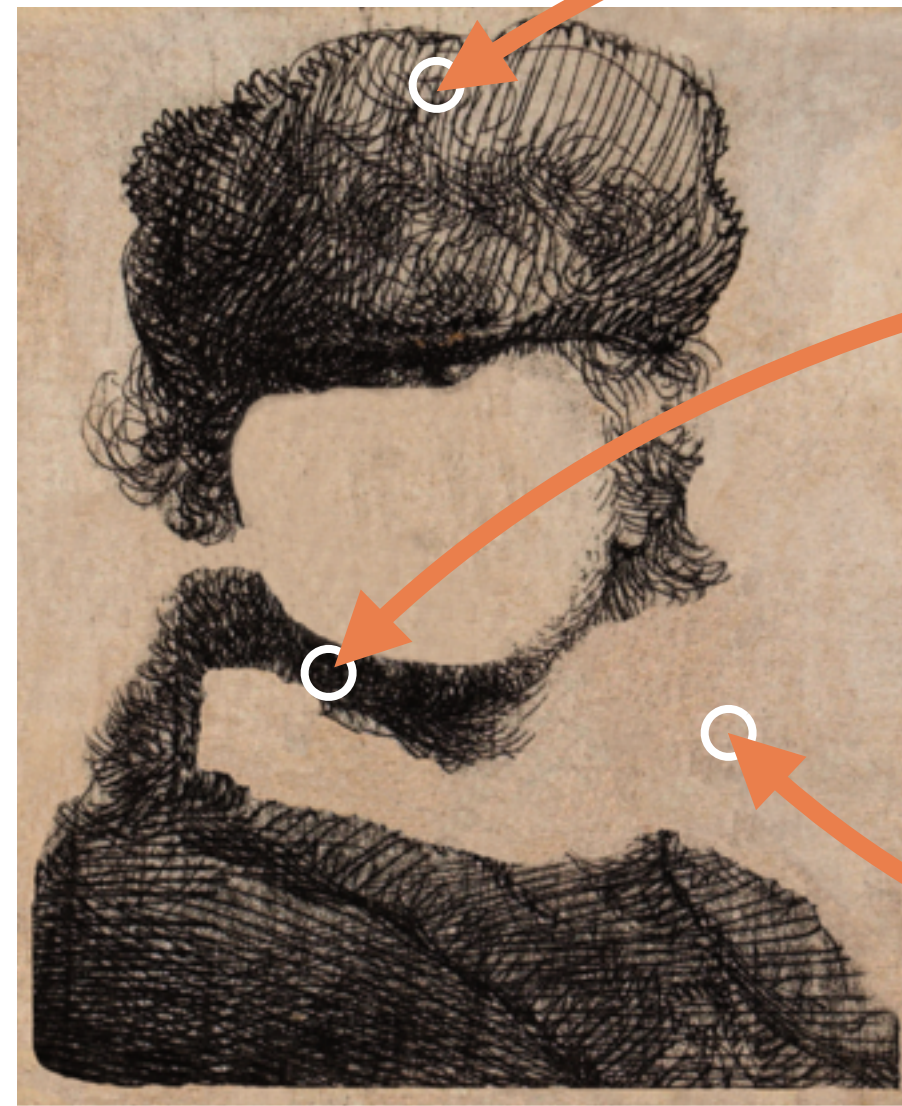
Output image \hat{I}

Image Analogies

$M(\mathbf{p})$



■
■



■ ■
■ ■



■
■



Style input S

Style output \hat{S}

Input image I

Output image \hat{I}

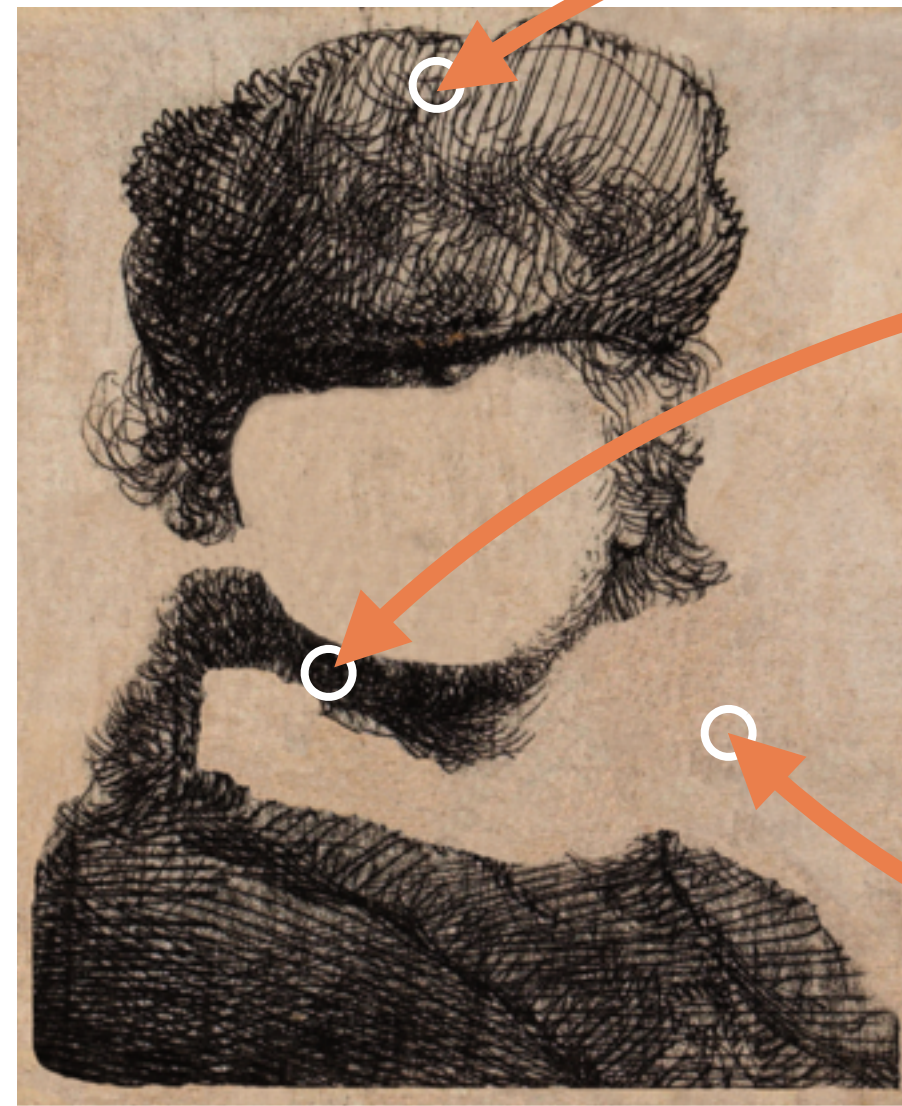
$$\hat{I}(\mathbf{p}) = \hat{S}(M(\mathbf{p}))$$

Image Analogies

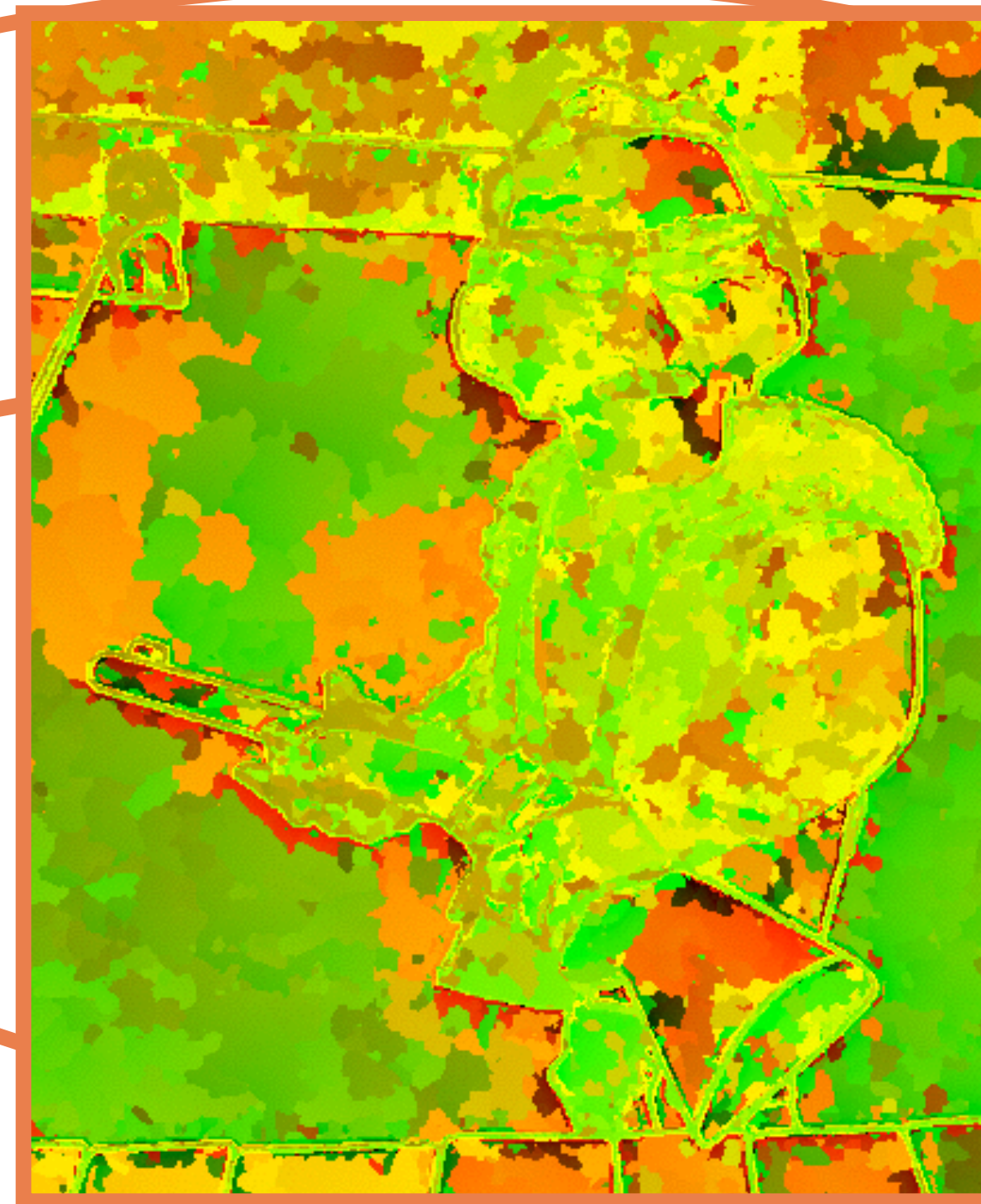
$M(\mathbf{p})$



■
■



■ ■
■ ■



■
■



Style input S

Style output \hat{S}

Input image I

Output image \hat{I}

$$\hat{I}(\mathbf{p}) = \hat{S}(M(\mathbf{p}))$$

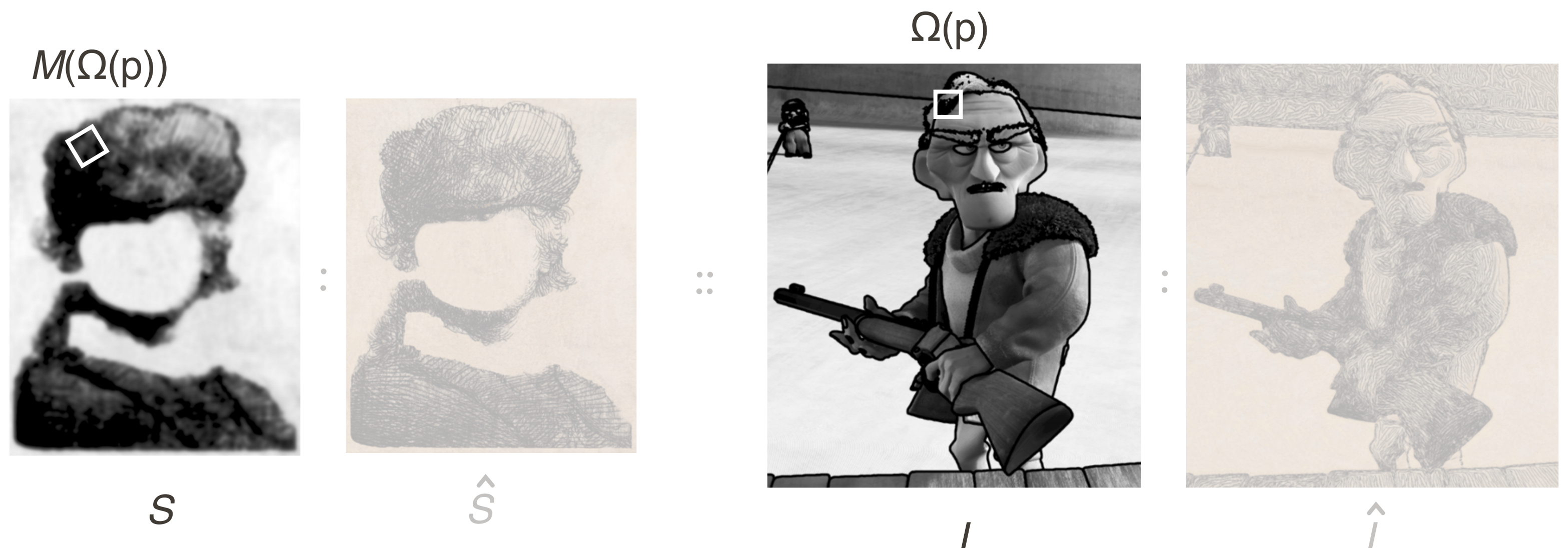
Goals for $M(p)$

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



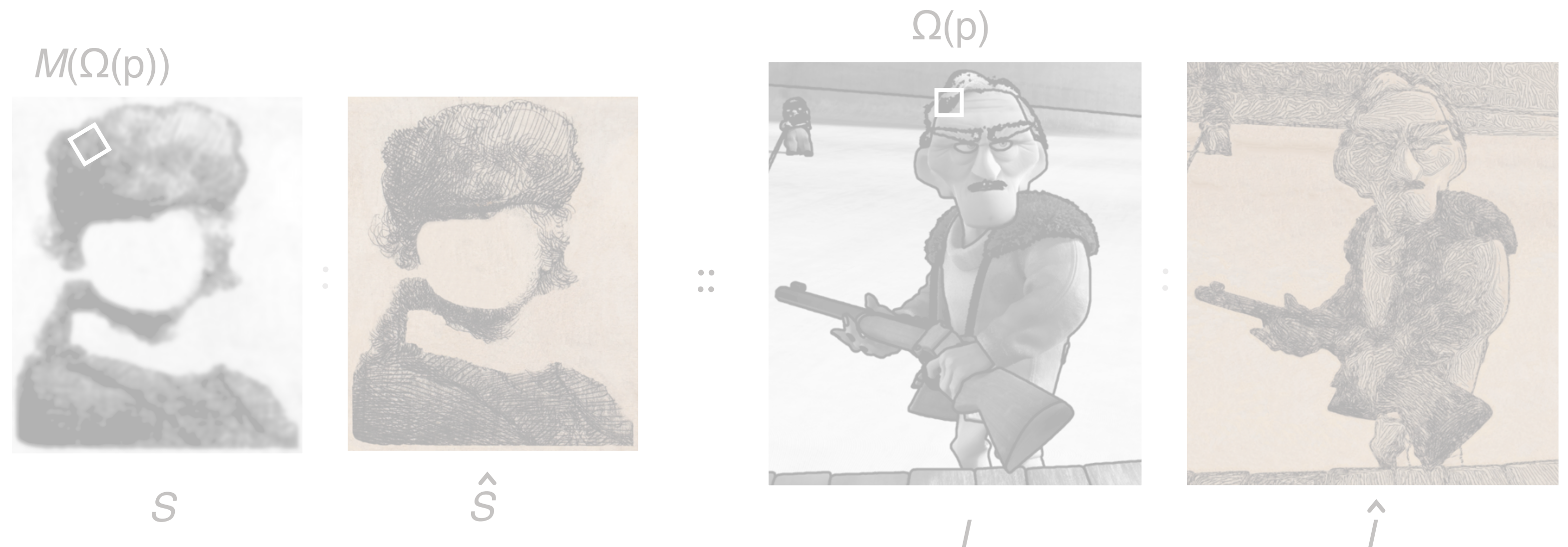
Goals for $M(p)$

- 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{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
- $M(p)$ is **continuous**

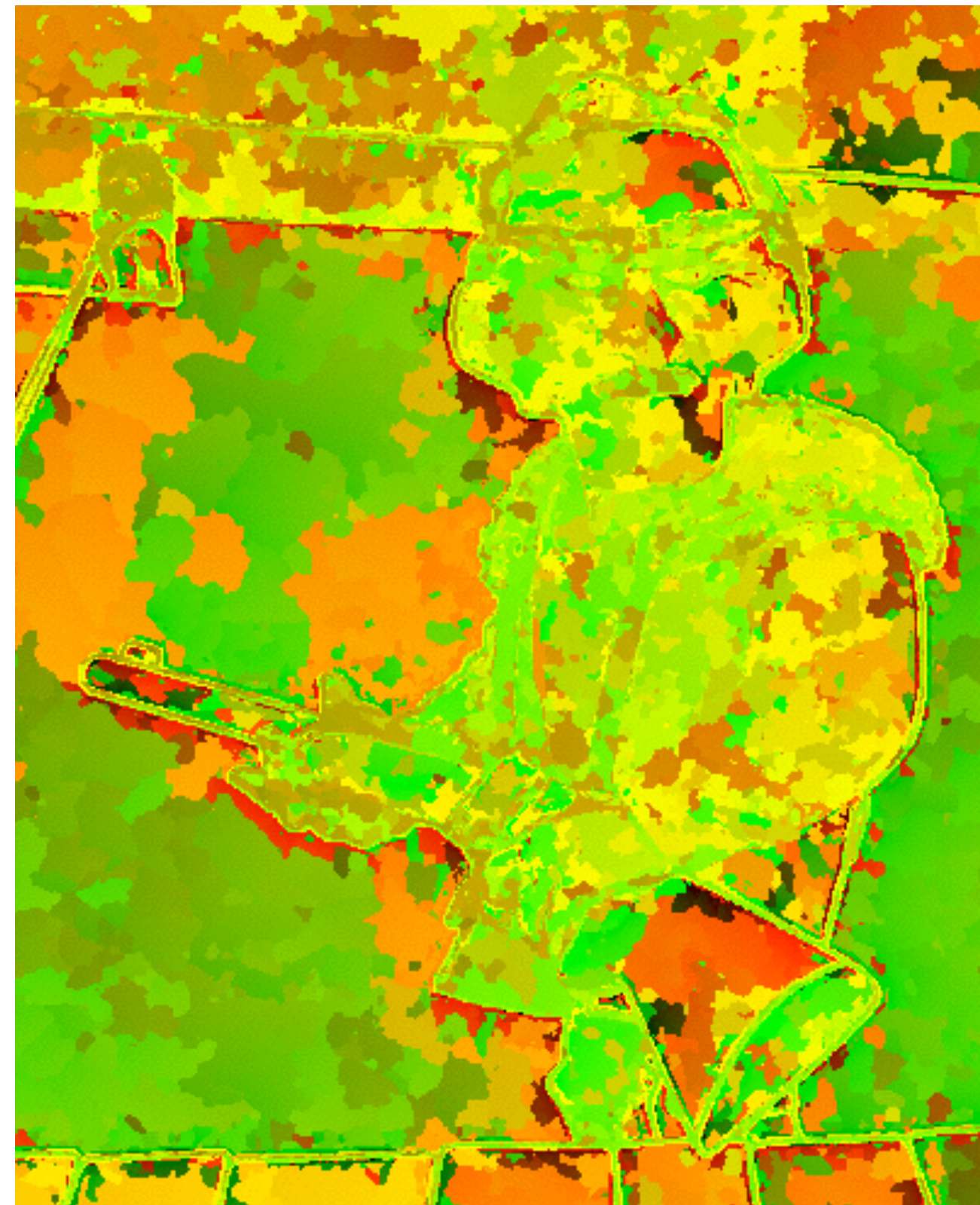


Optimization of $M(\mathbf{p})$

PatchMatch

[Barnes et al. 2009]

~5 minutes per frame
(1920×1080 pixels,
NVIDIA Quadro 500)

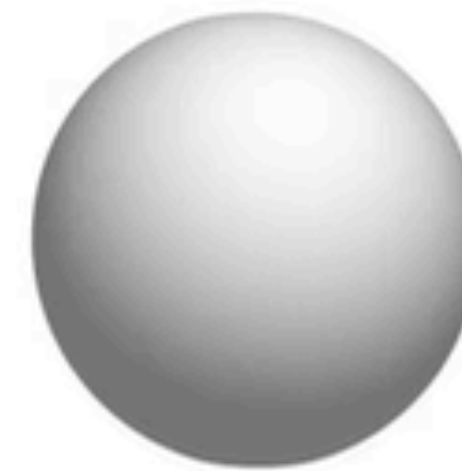


Offset field M



Output image I

Input Animation



Style pair



Input

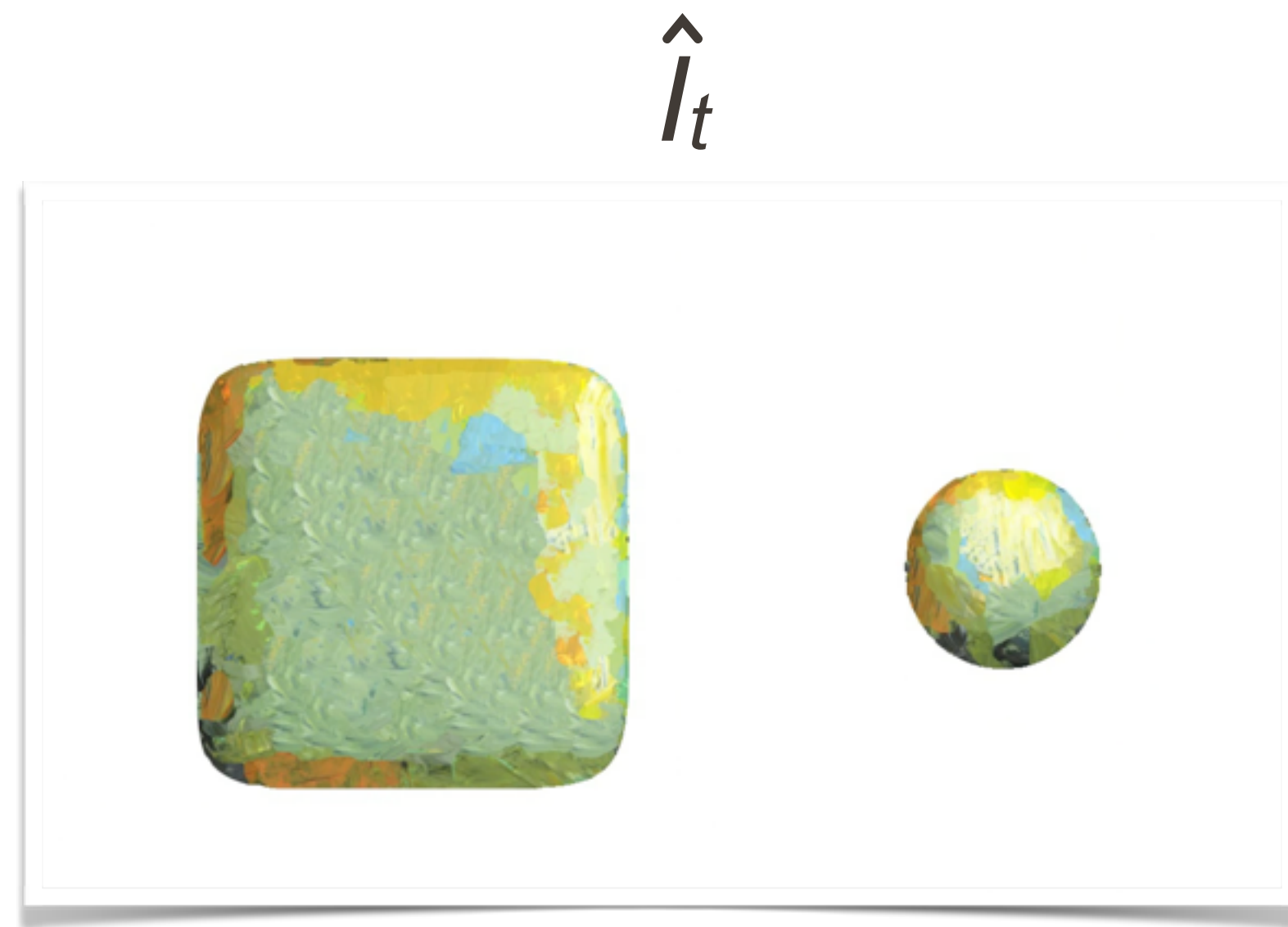


Output

Independent synthesis per frame

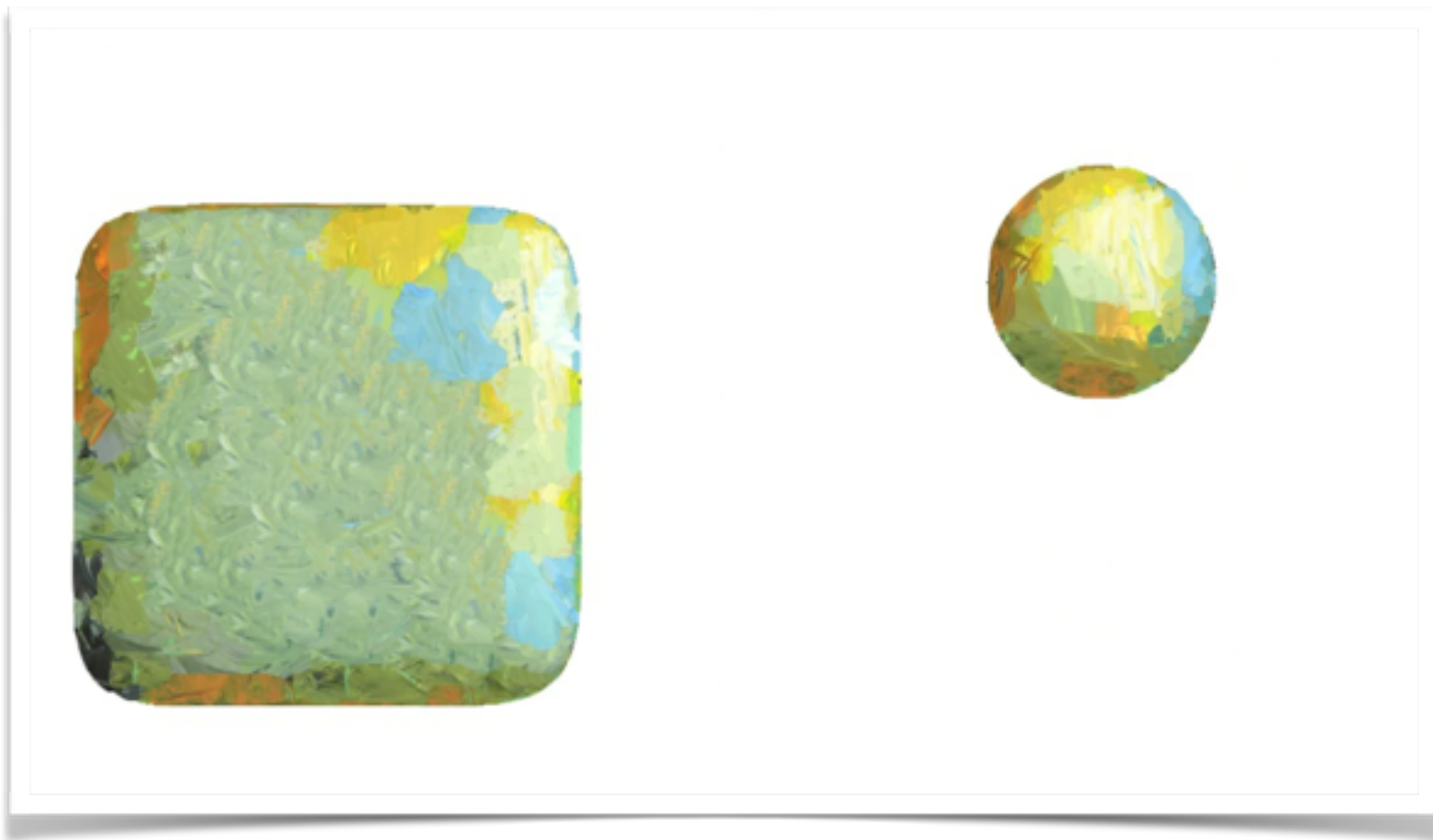


Temporal Coherence Goals



Temporal Coherence Goals

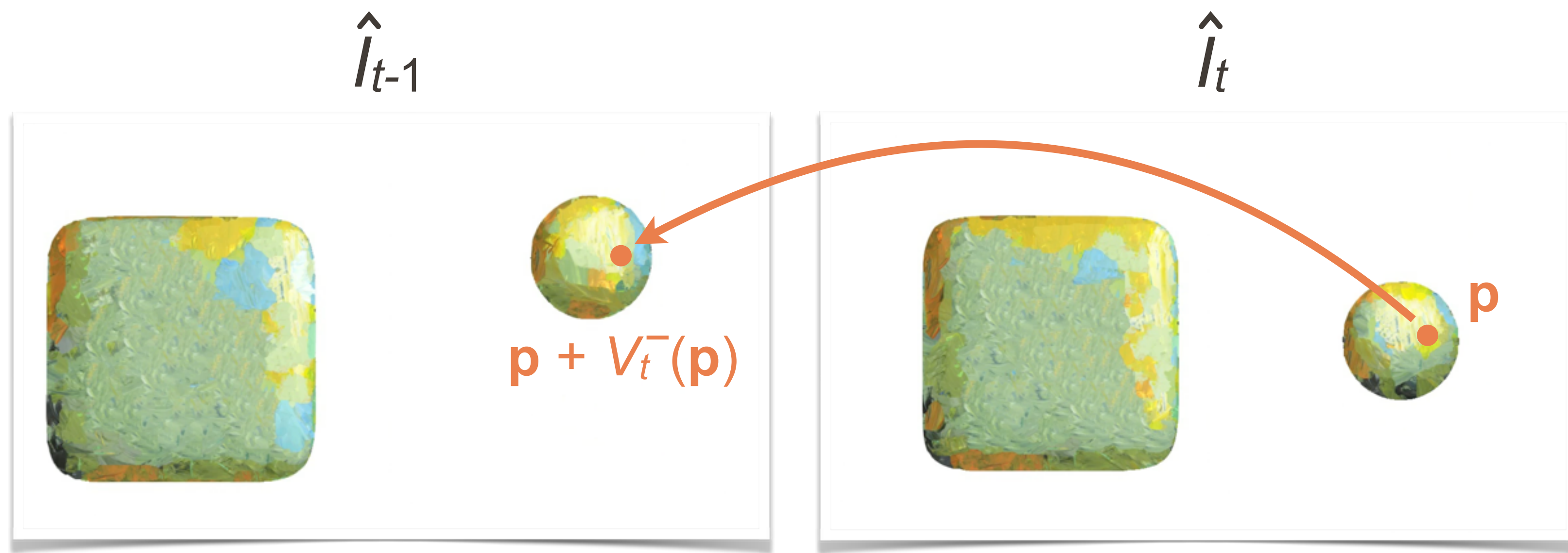
\hat{l}_{t-1}



\hat{l}_t

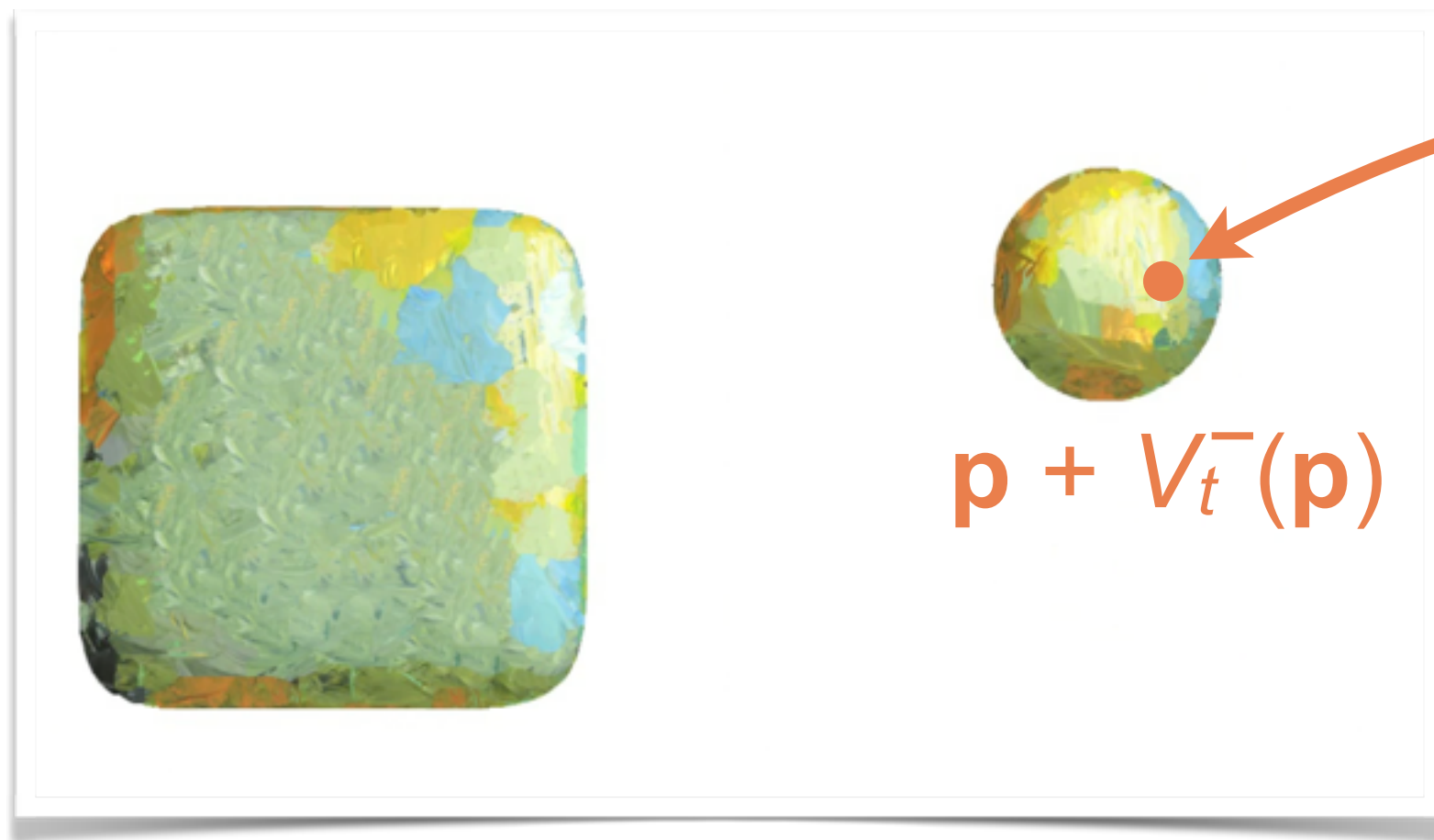


Temporal Coherence Goals

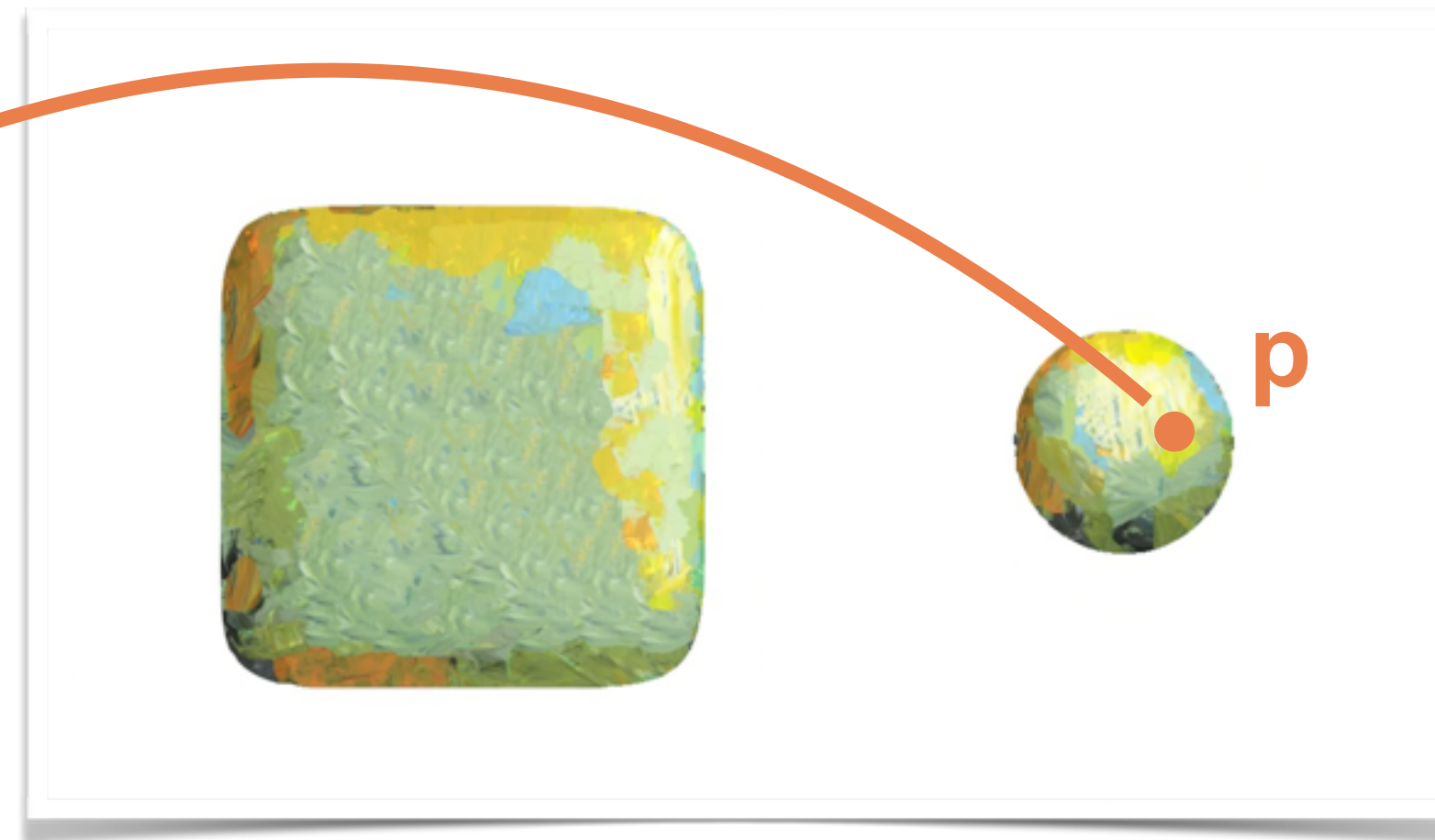


Temporal Coherence Goals

\hat{l}_{t-1}



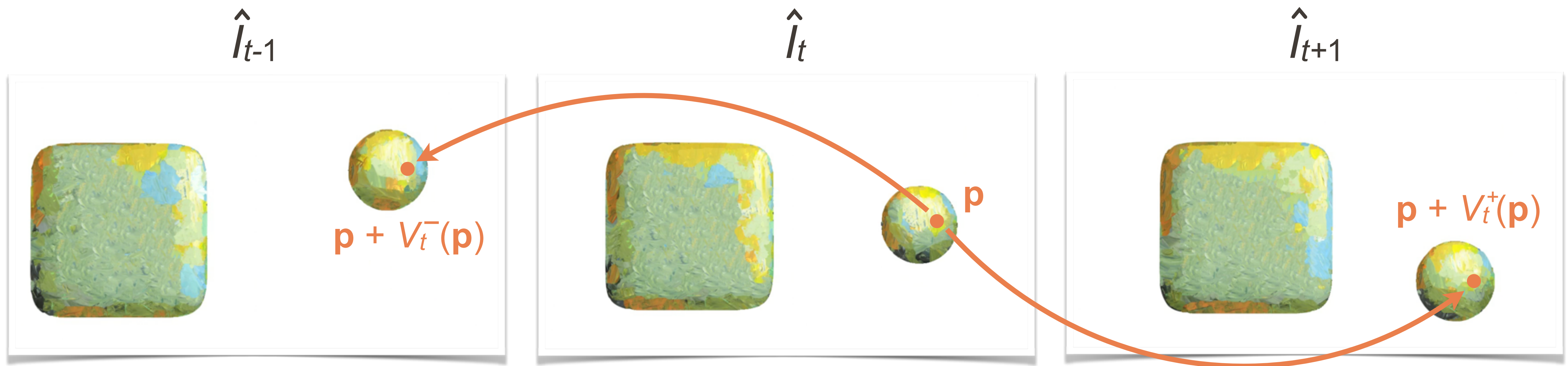
\hat{l}_t



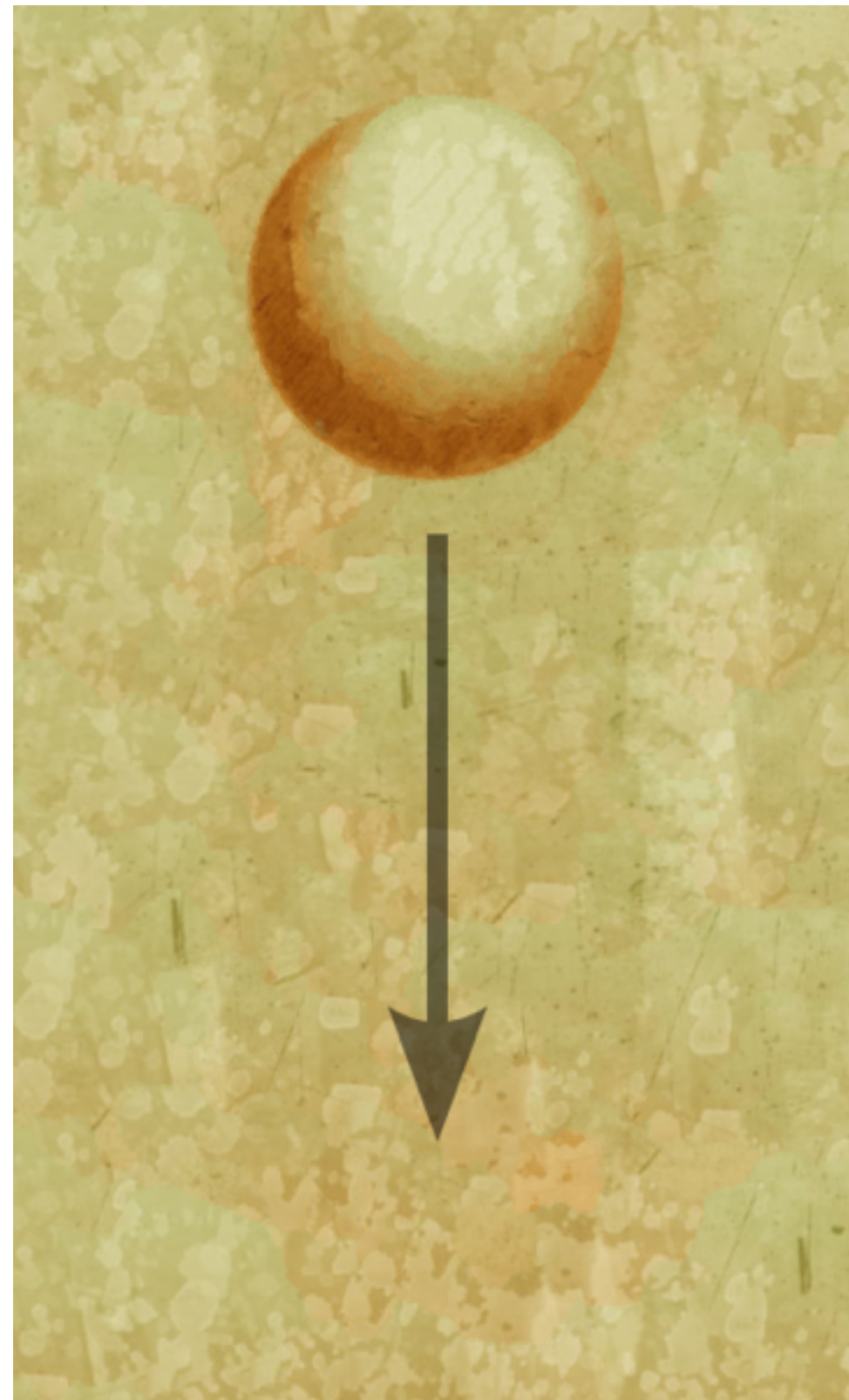
\hat{l}_{t+1}



Temporal Coherence Goals



Disocclusions leave trails

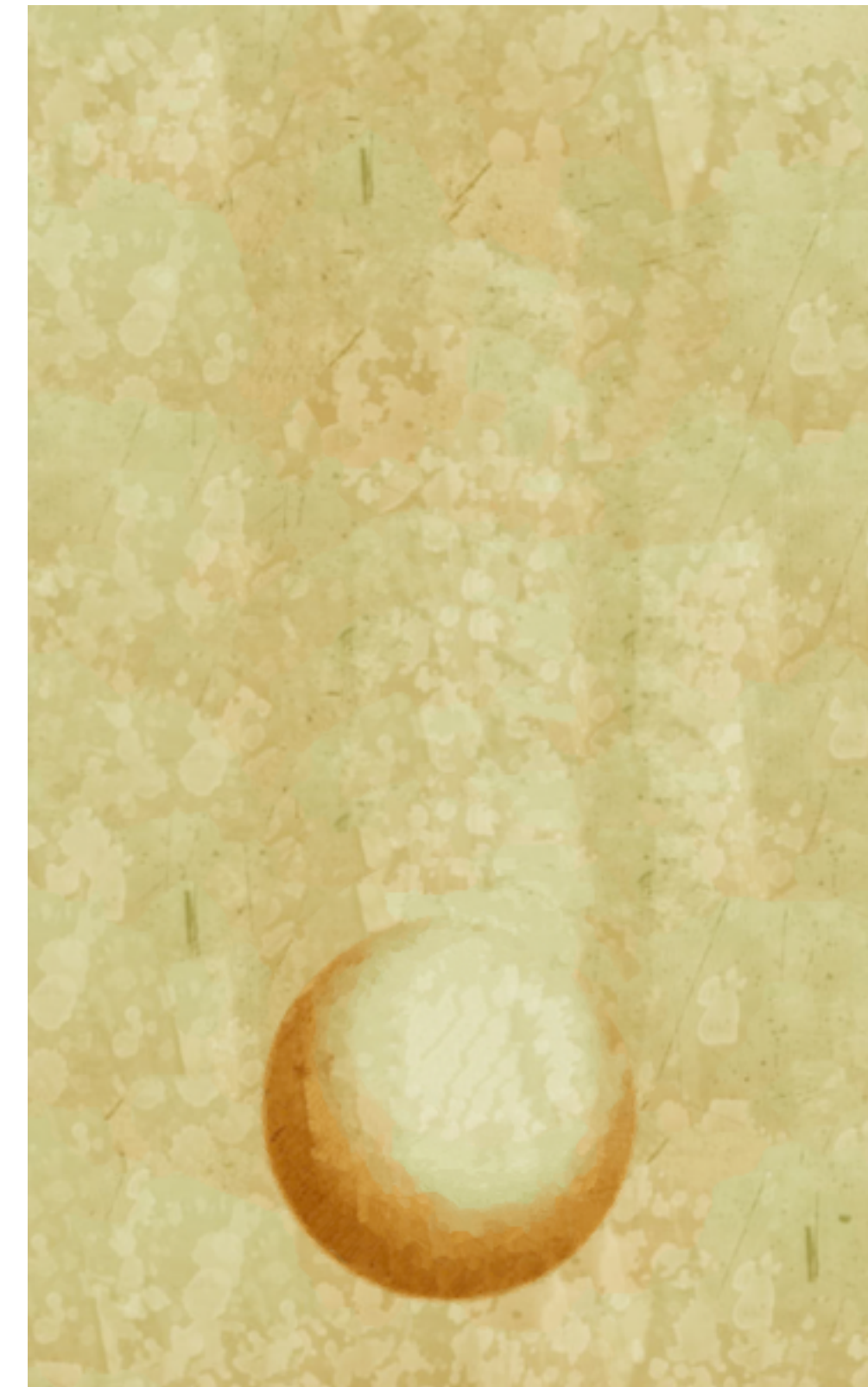


frame 0



frame 10

ignoring disocclusions

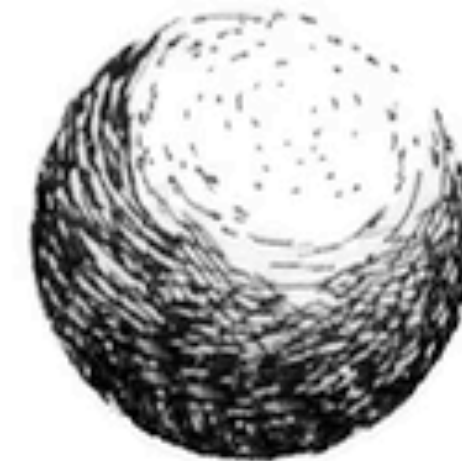
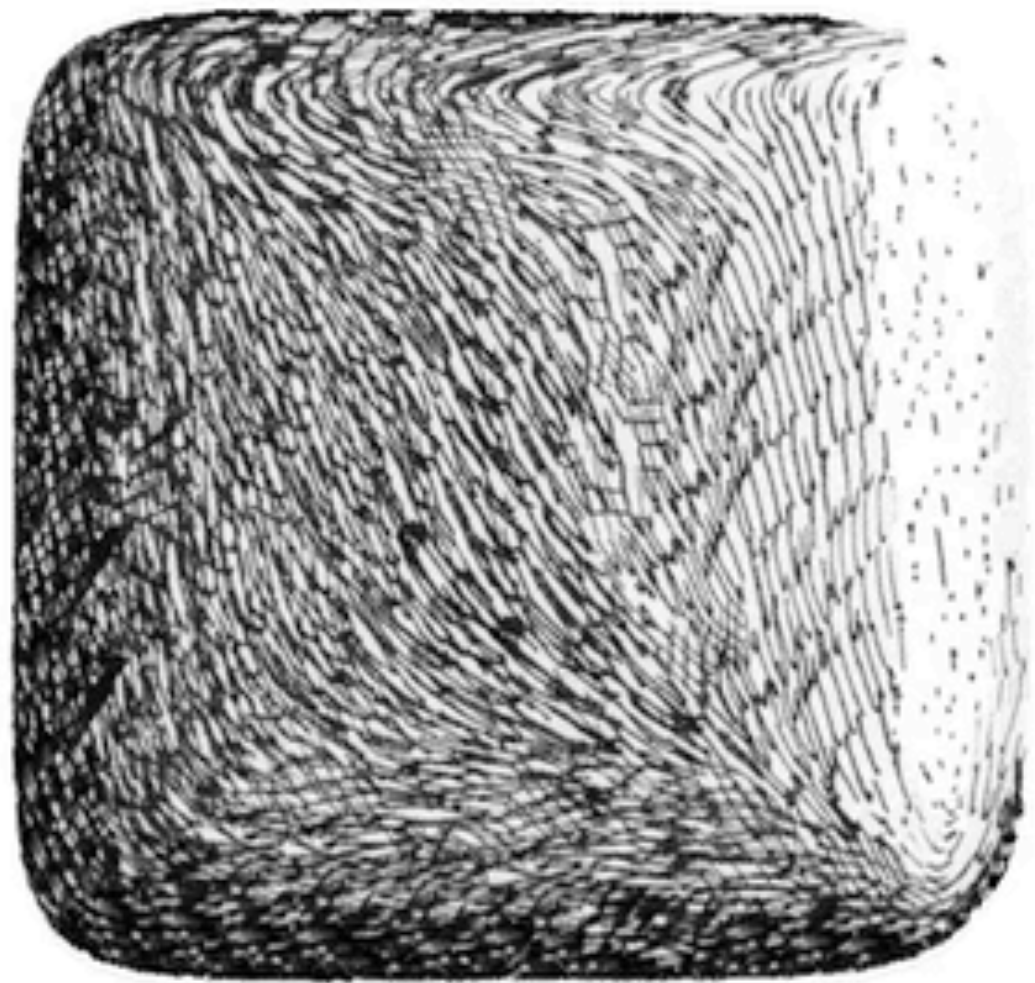


frame 10

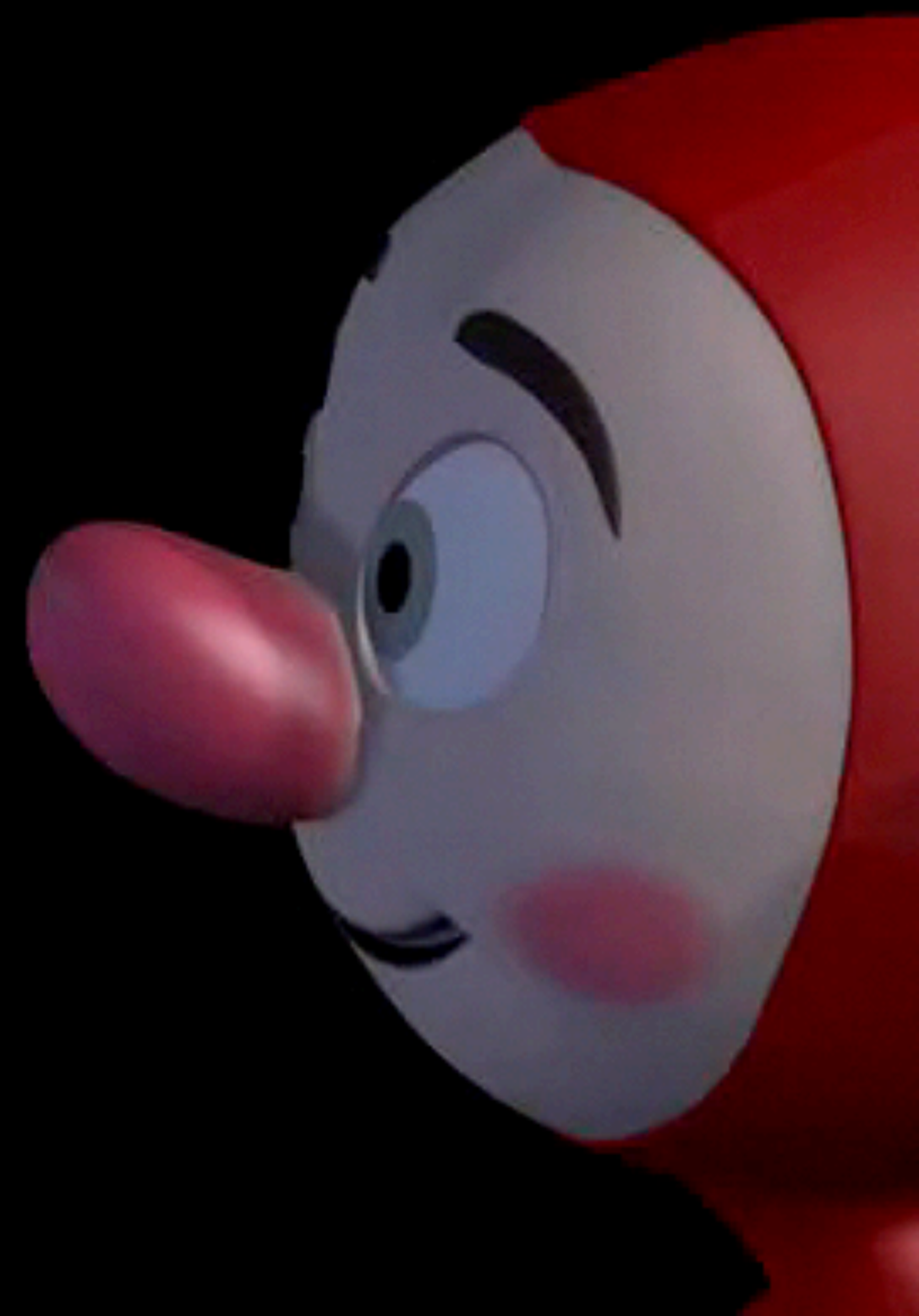
With temporal coherence



Hatching style



Input animation



Keyframe Setup

Input



Keyframe Setup

Input



Painted
keyframe 1

Keyframe Setup

Input



Painted
keyframe 1

Painted
keyframe 9

Input



Painted
keyframe 1

Style



Painted
keyframe 9

Handling Overdraw



Handling Overdraw

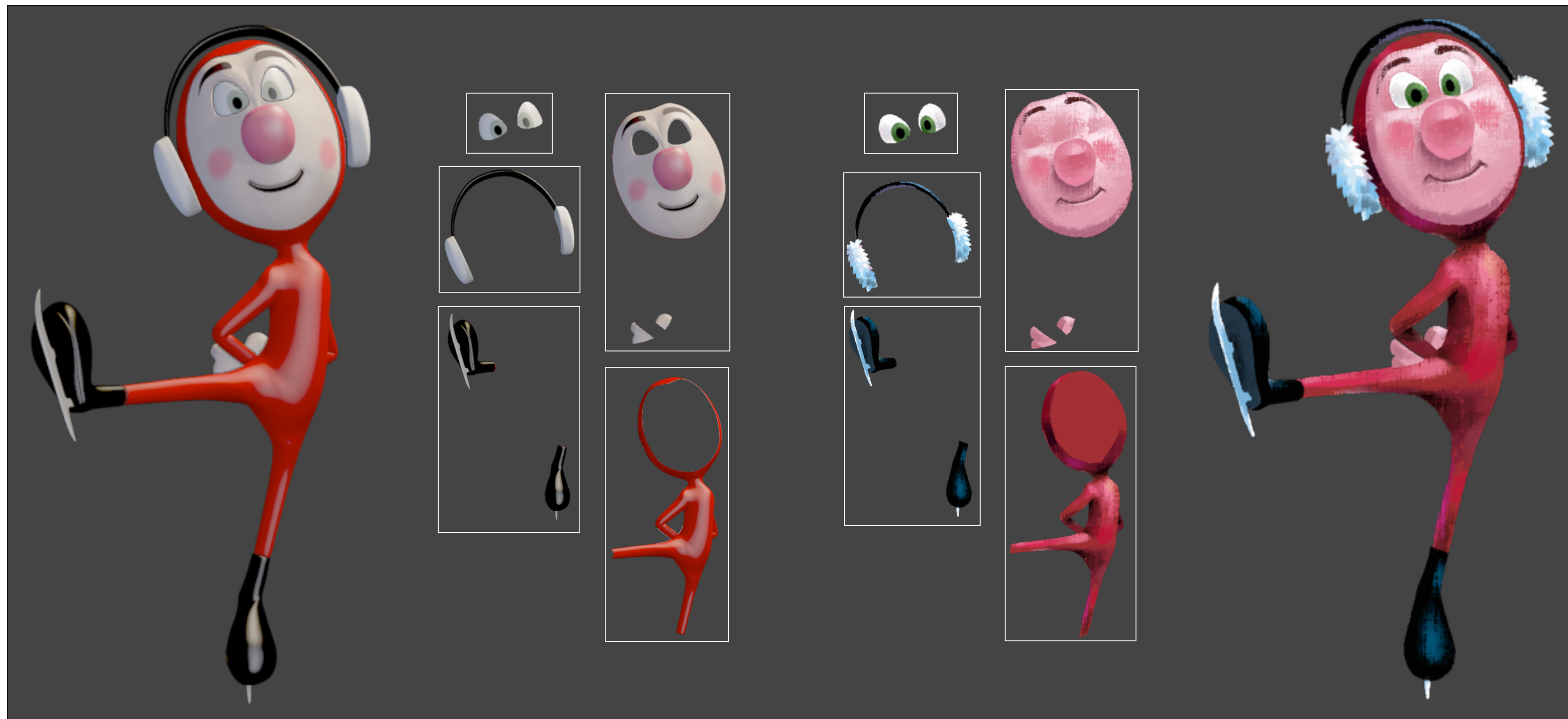


Input mask for earmuffs

Overdraw



Layer Decomposition



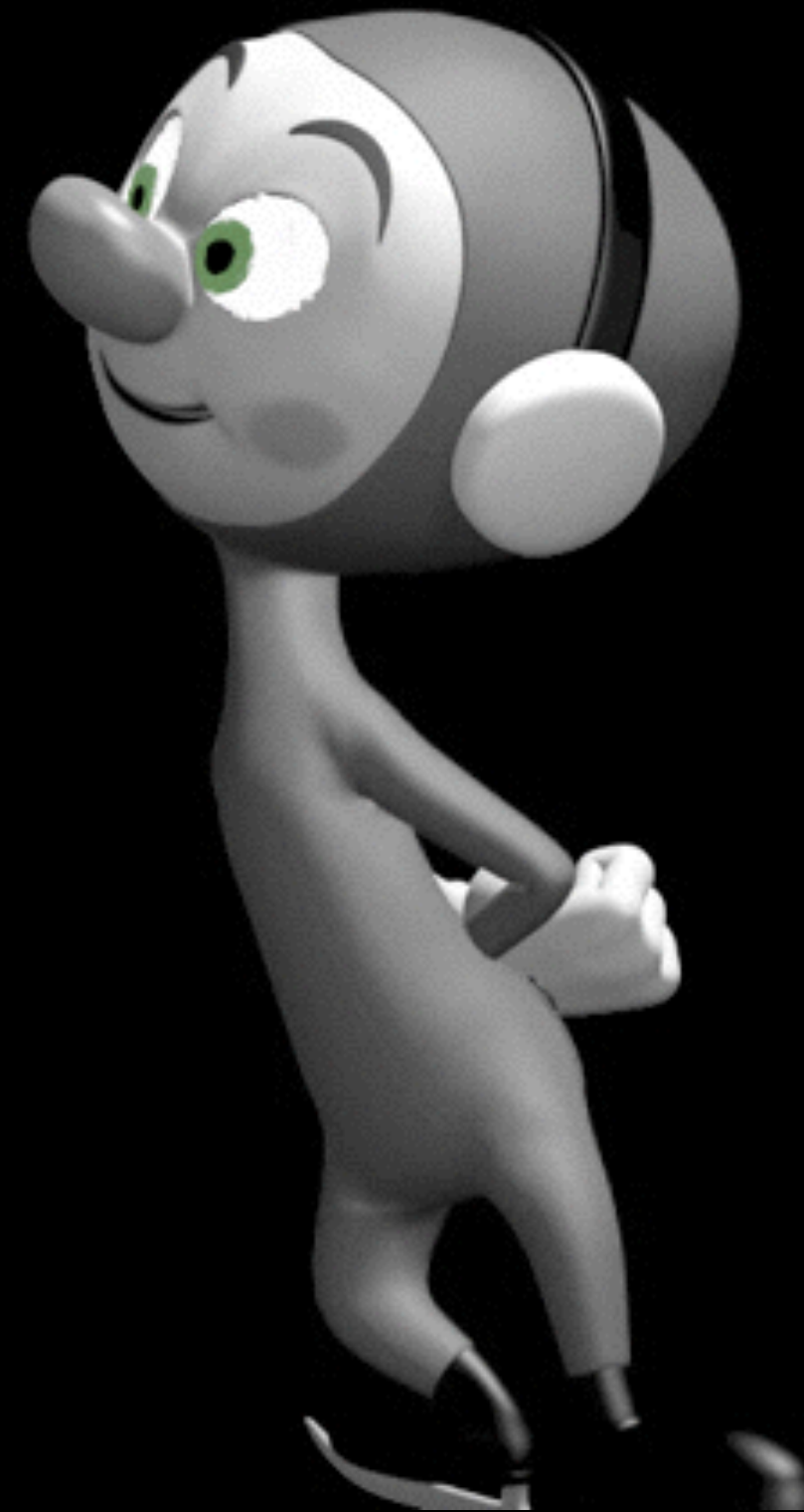
Input

Artist's painting

Keyframes



Keyframes



Keyframes



Keyframes



Keyframes



Keyframes



Keyframes



Keyframes



Keyframes



Keyframes



Keyframes



Keyframes



Keyframes



Keyframes



Keyframes



Synthesis result

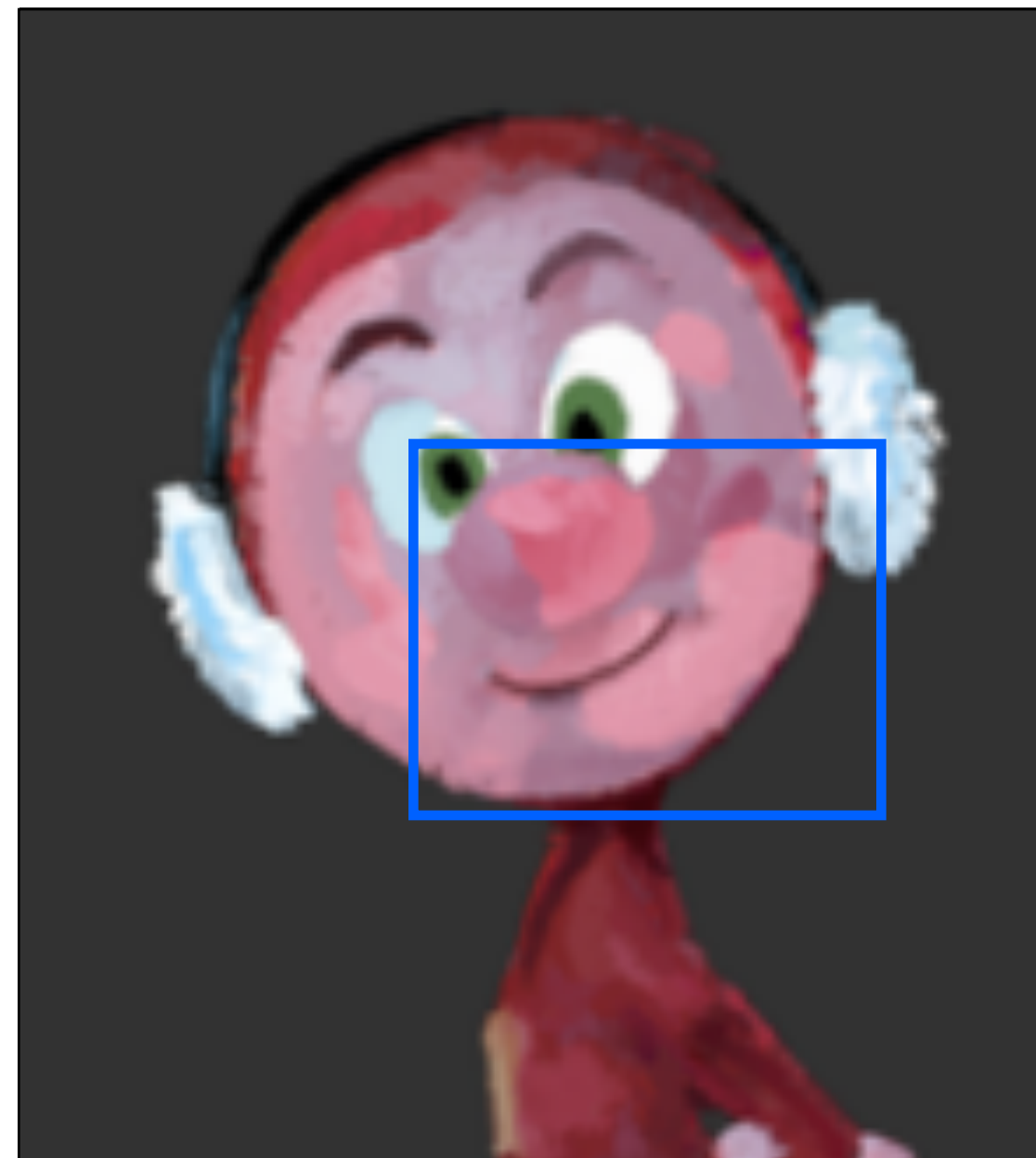
(half speed)





Limitations and Future Work

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



Output

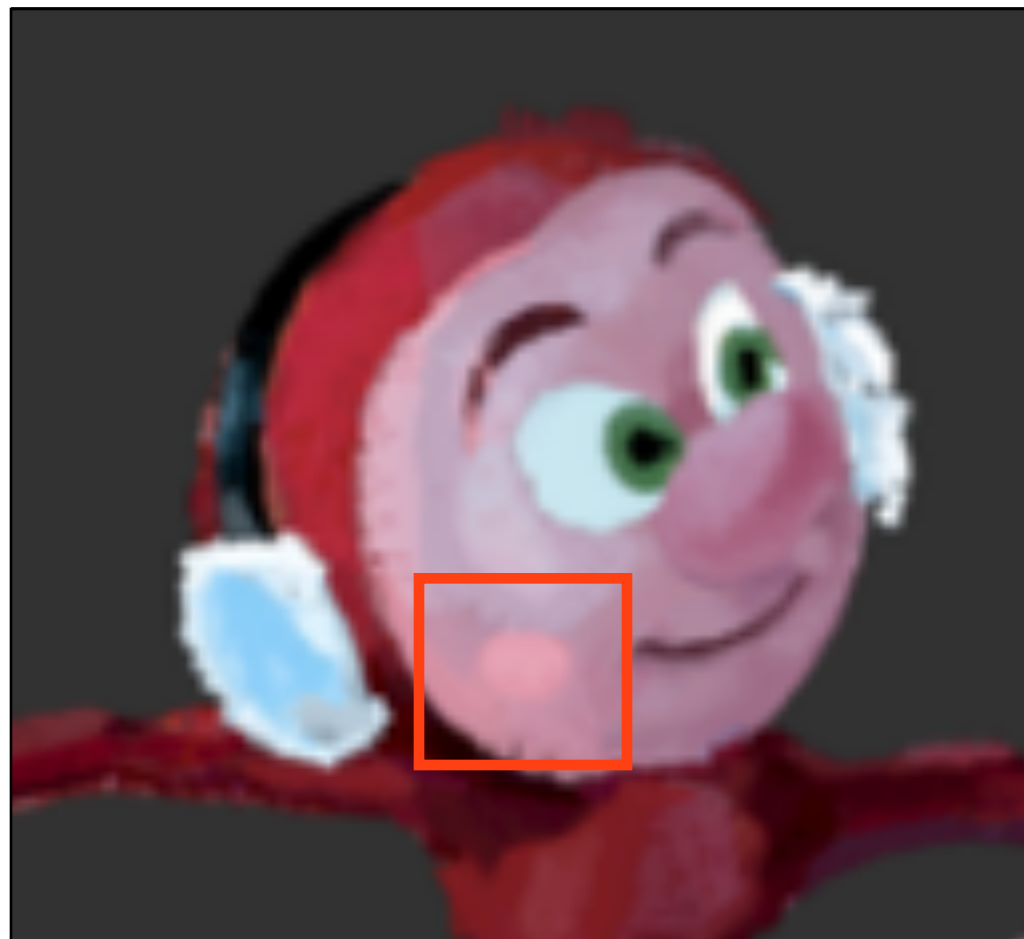


Keyframes

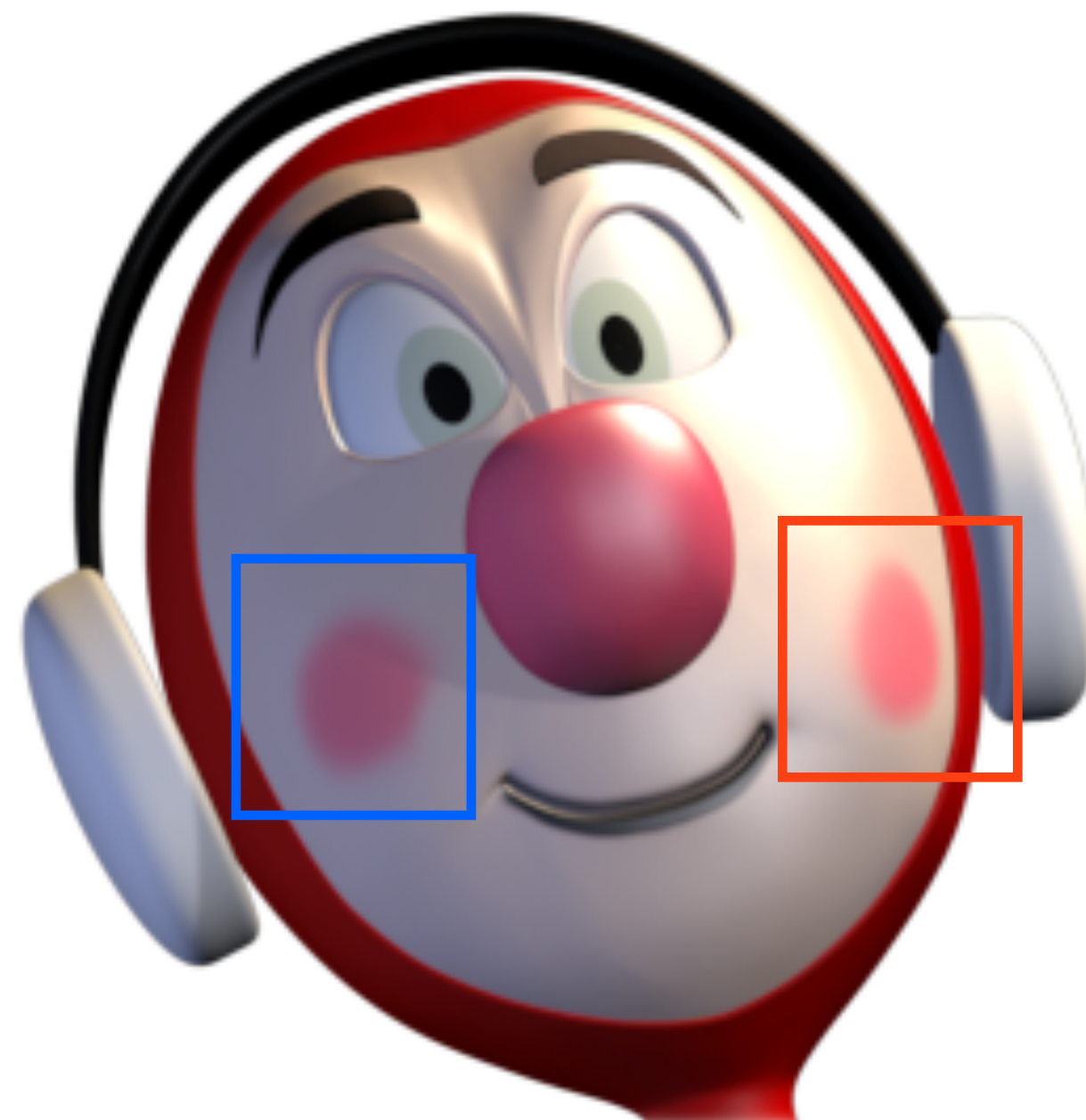
Limitations and Future Work

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

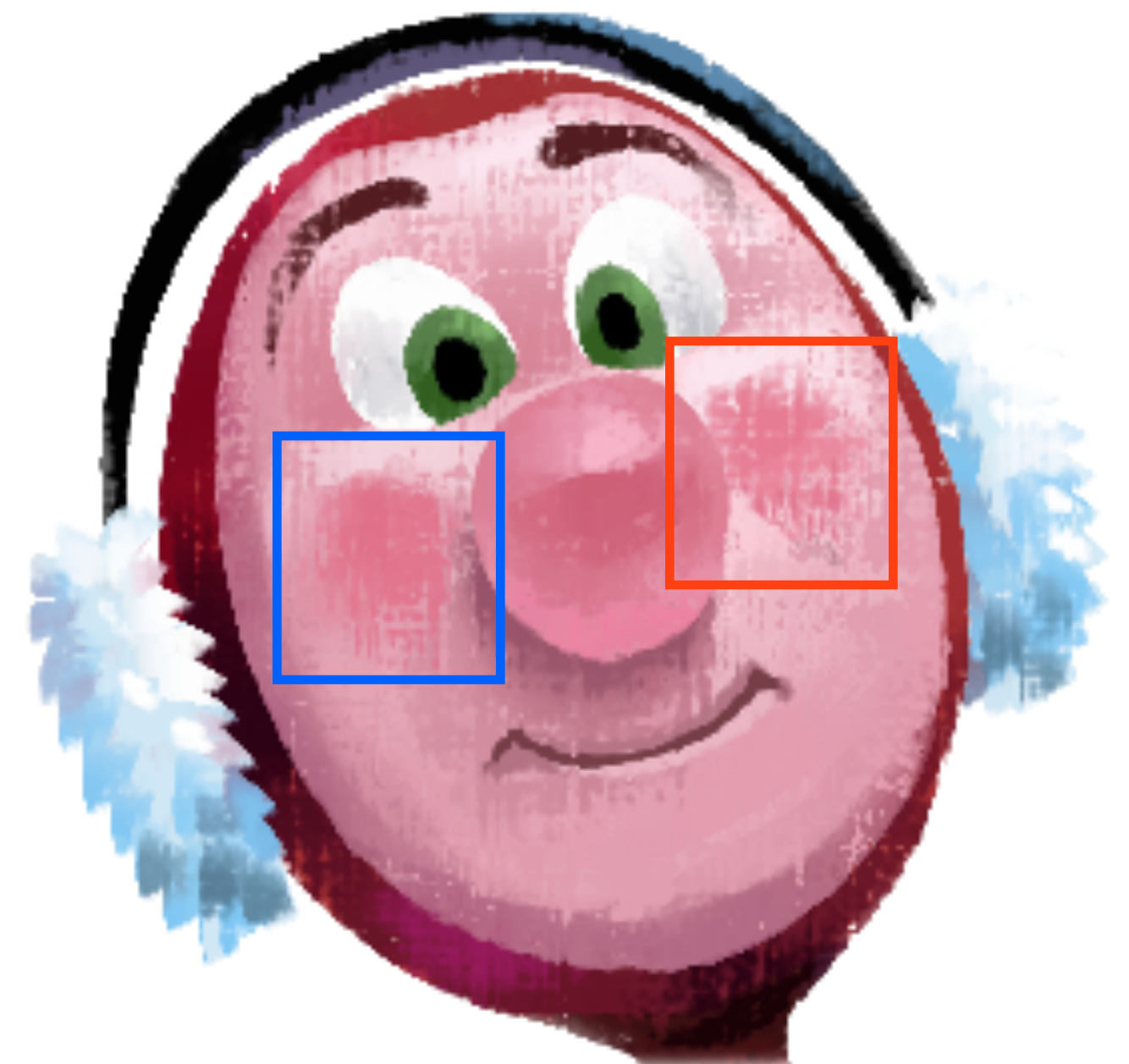
Better analogy inputs



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]

P. Bénard, F. Cole, M. Kass,
I. Mordatch, J. Hegarty, M.-S. Senn,
K. Fleischer, D. Pesare, K. Breeden

Pixar Animation Studios

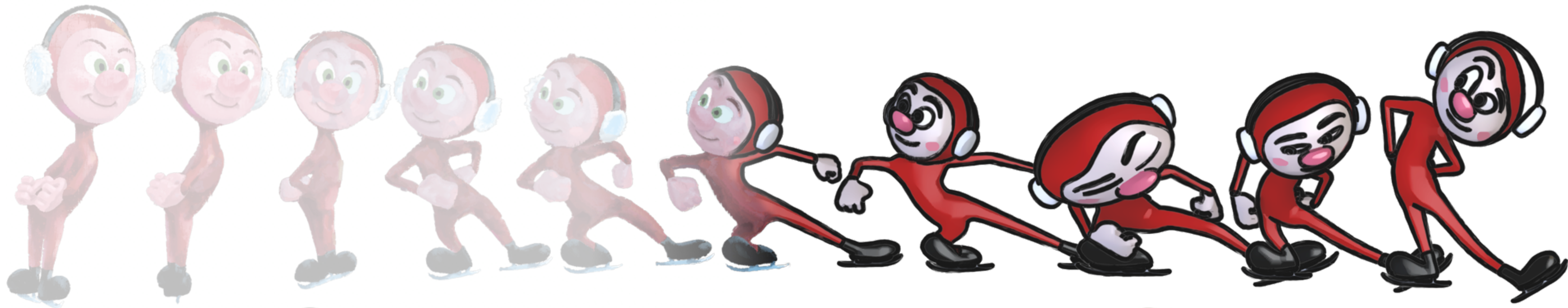
University of Toronto, University of Washington, Stanford University

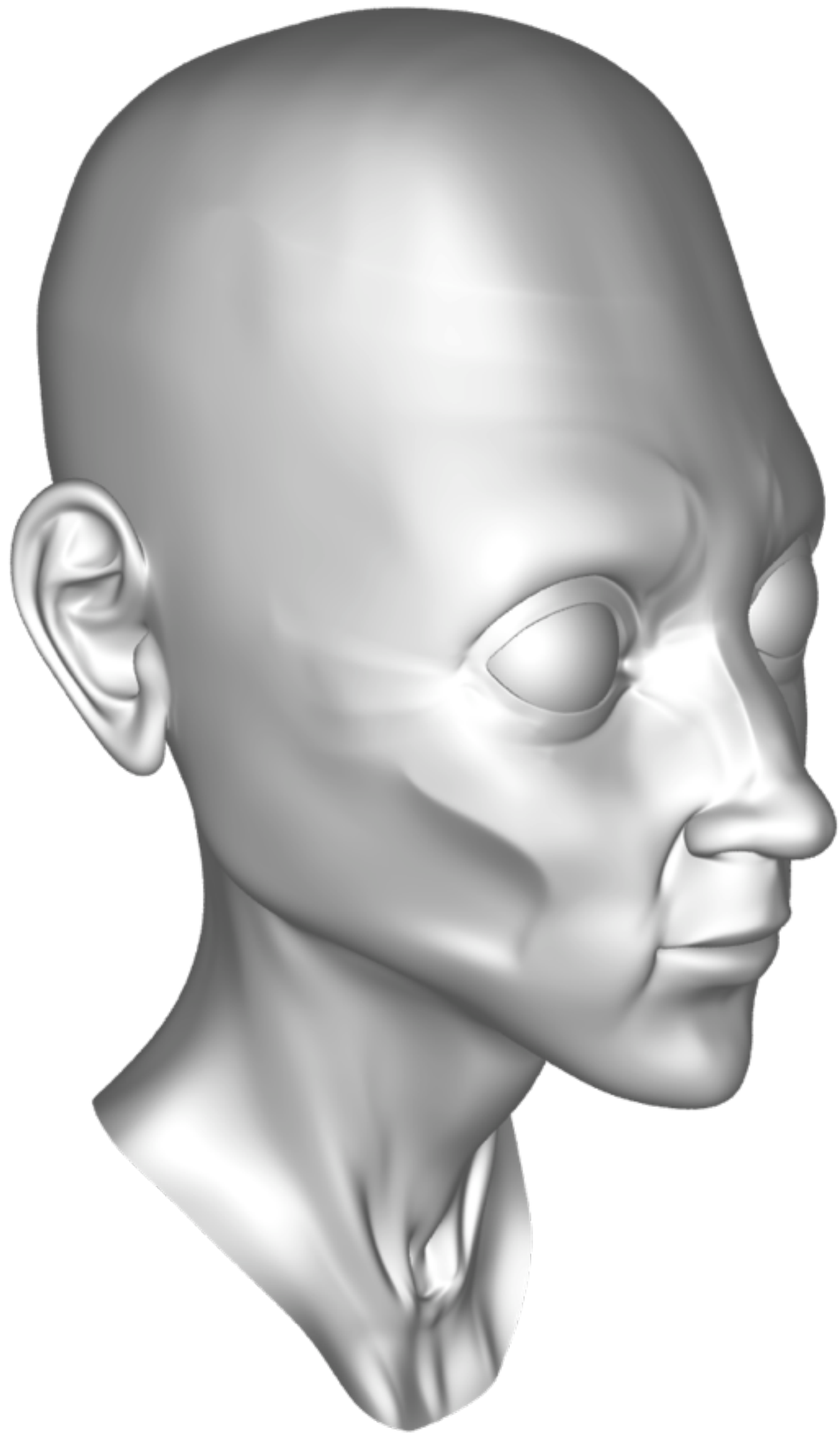
Stylizing Animation by Example

P. Bénard, A. Hertzmann, M. Kass

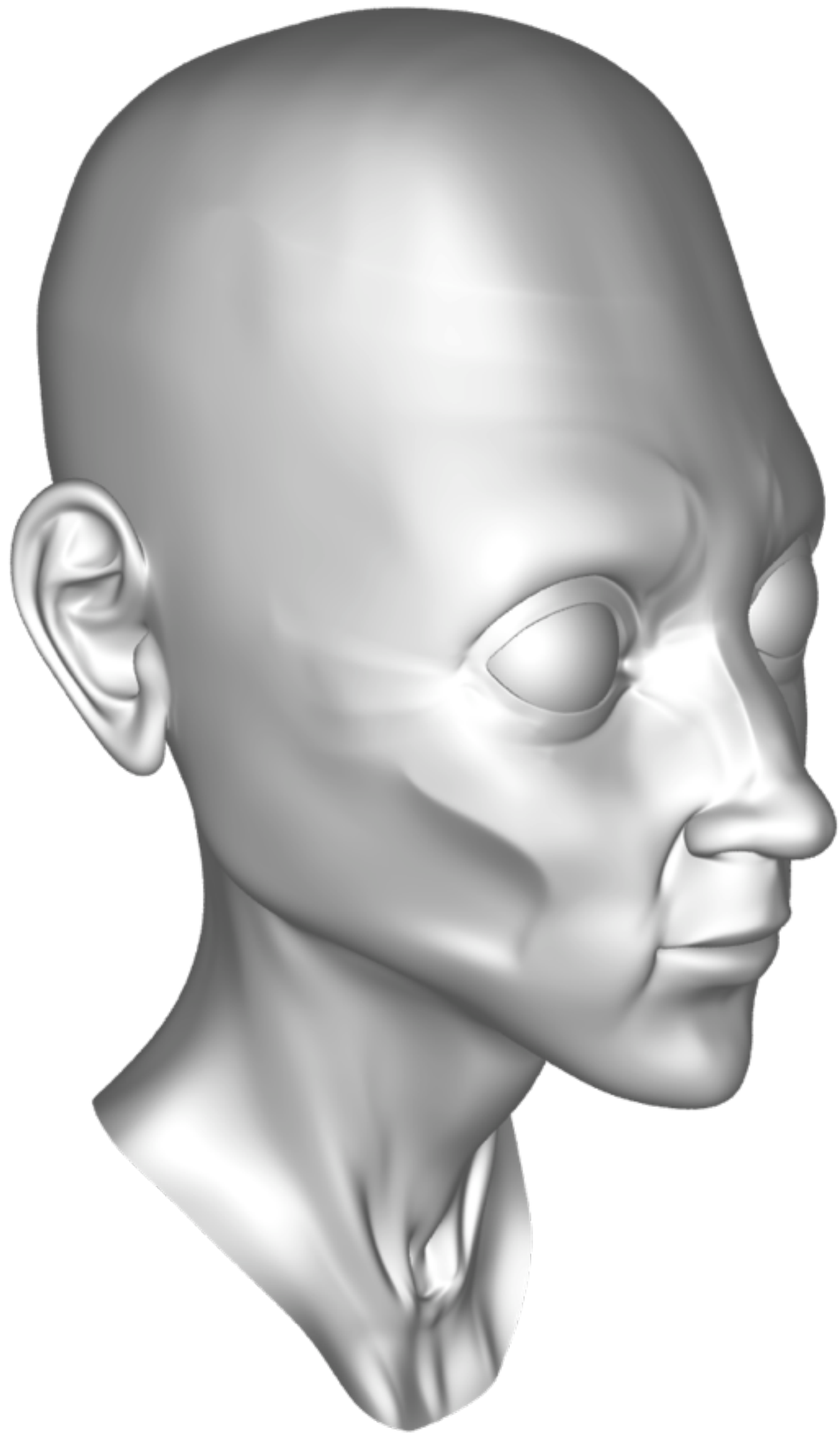
*University of Toronto, Université de Bordeaux,
Adobe Research, Pixar Animation Studios*

Computing Smooth Surface Contours with Accurate Topology

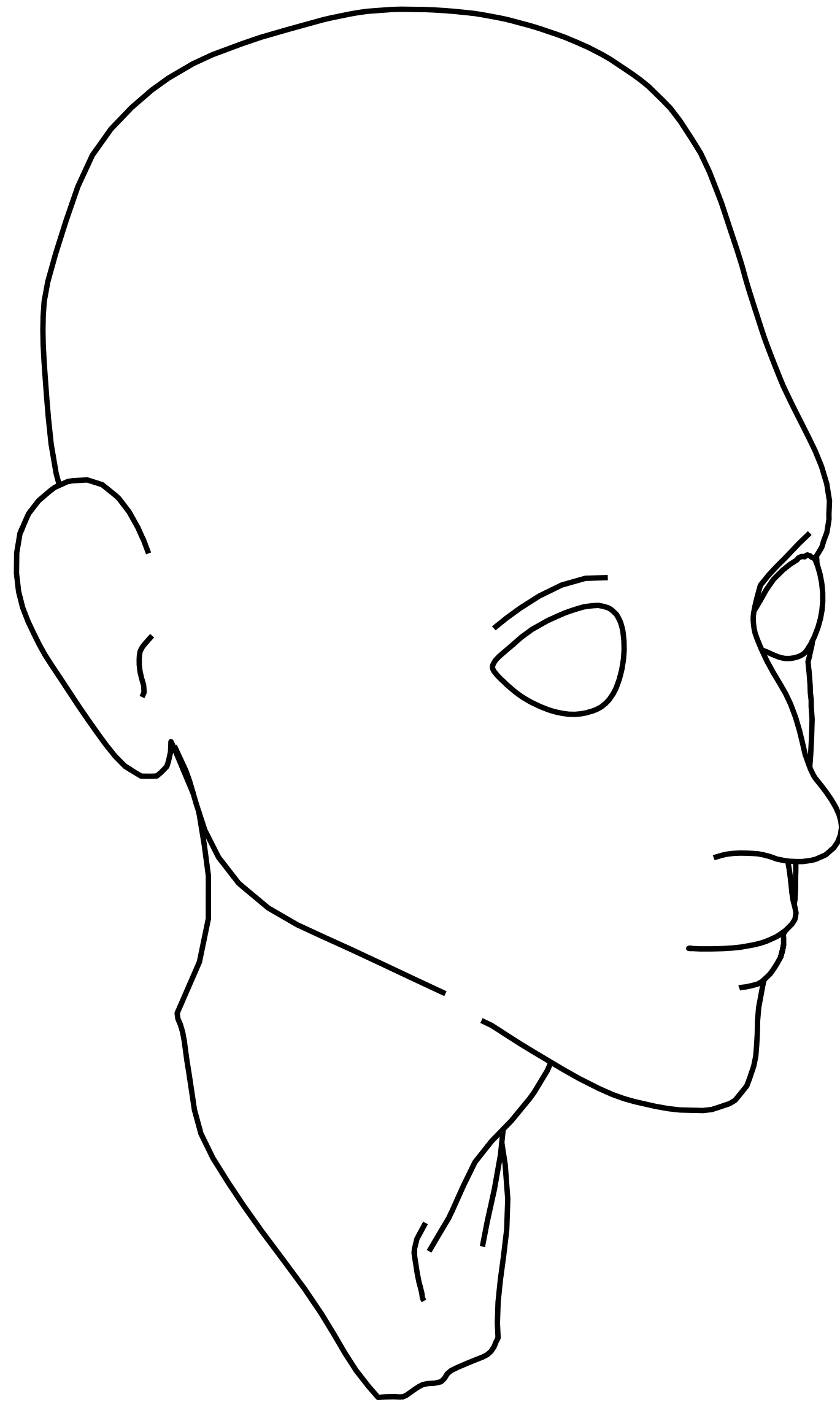




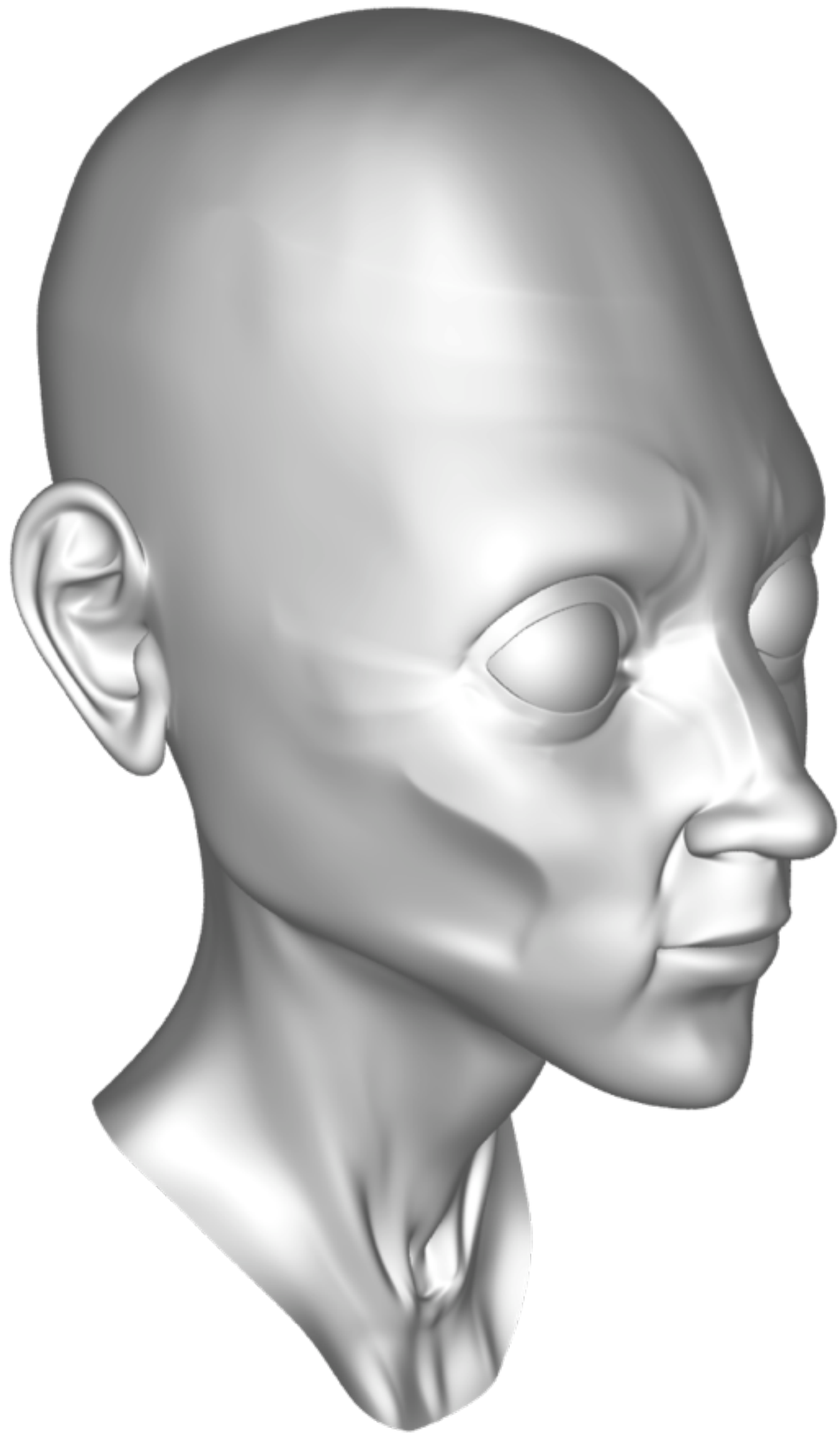
Smooth surface
[Catmull and Clark 1978]



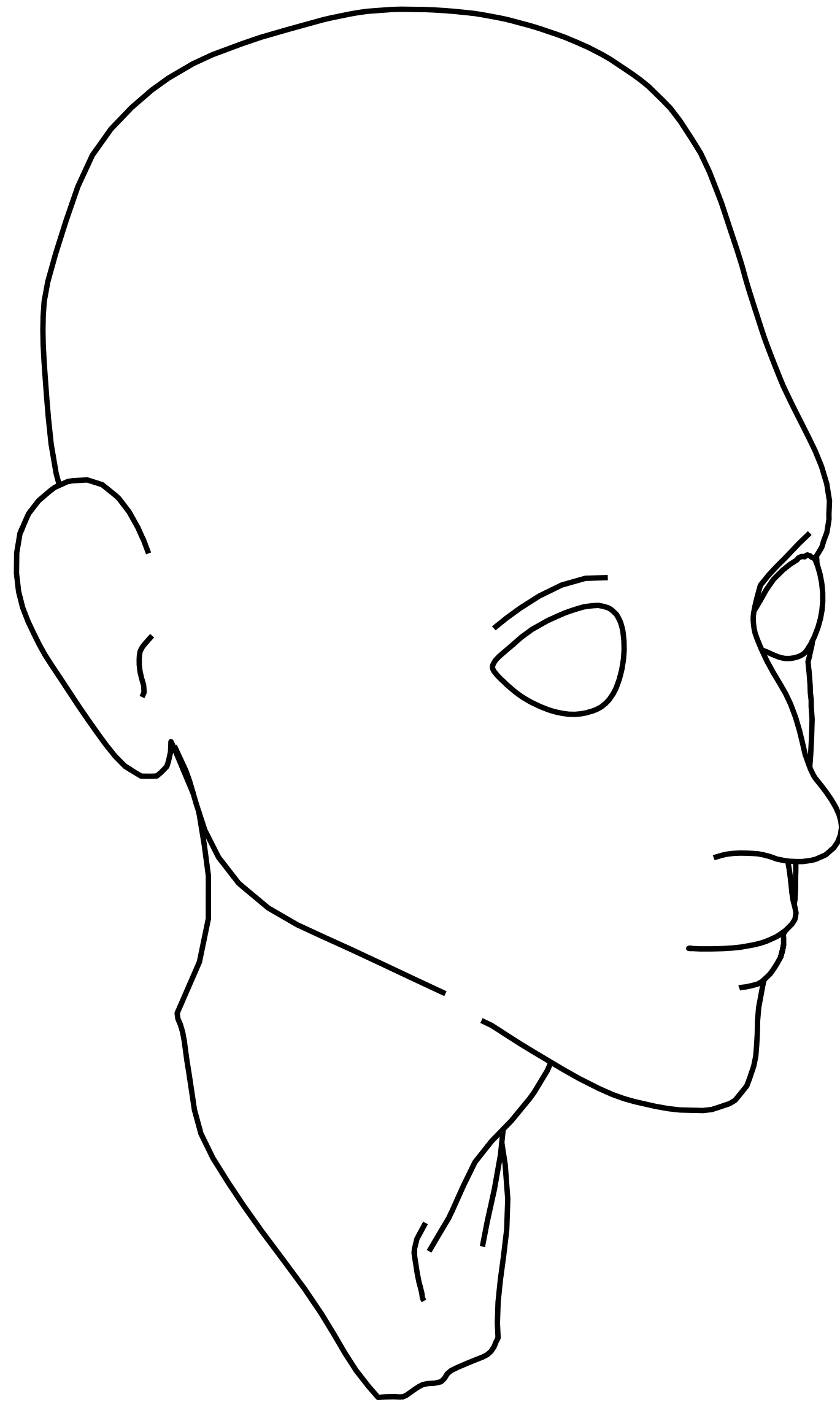
Smooth surface
[Catmull and Clark 1978]



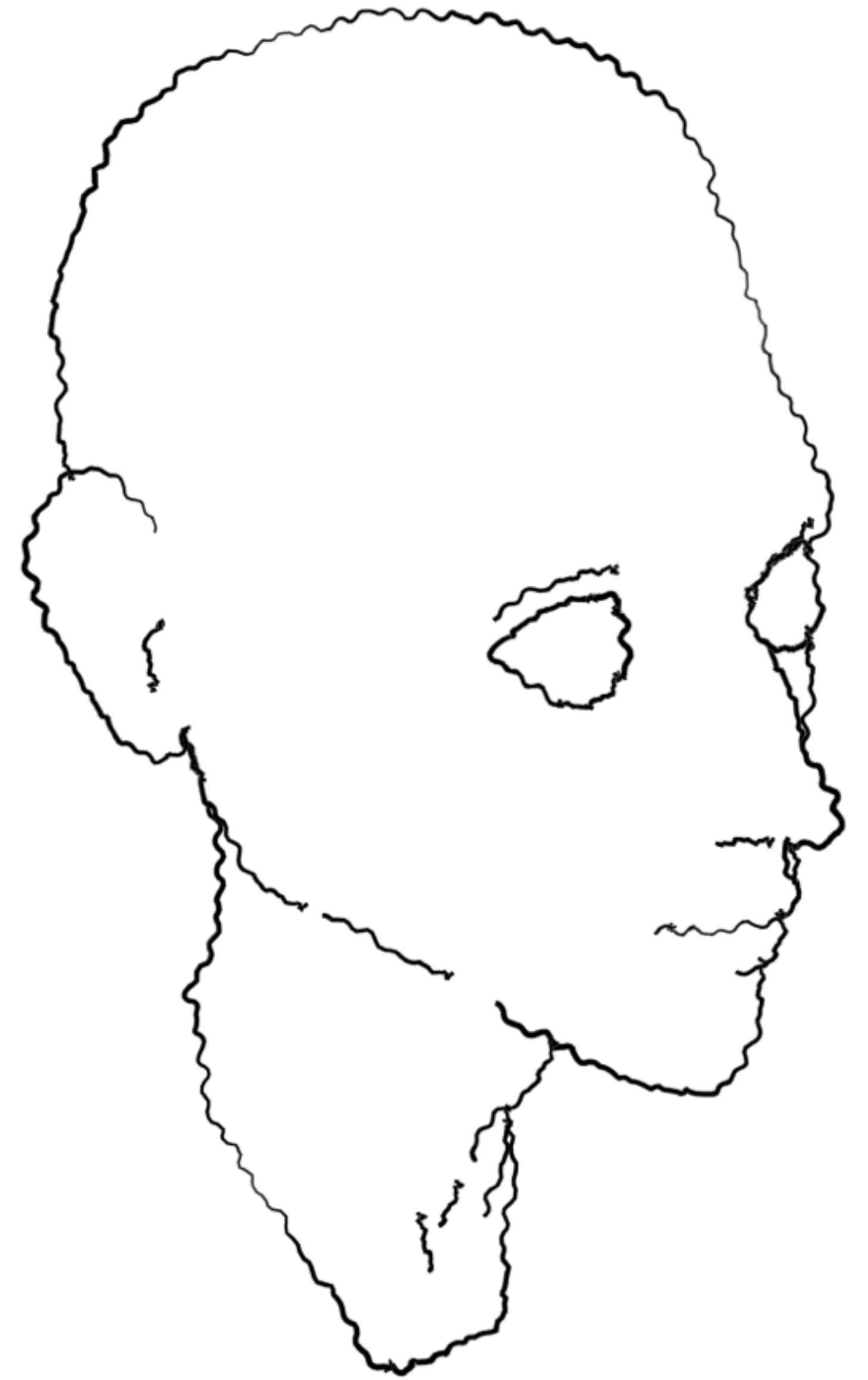
Occluding contours



Smooth surface
[Catmull and Clark 1978]



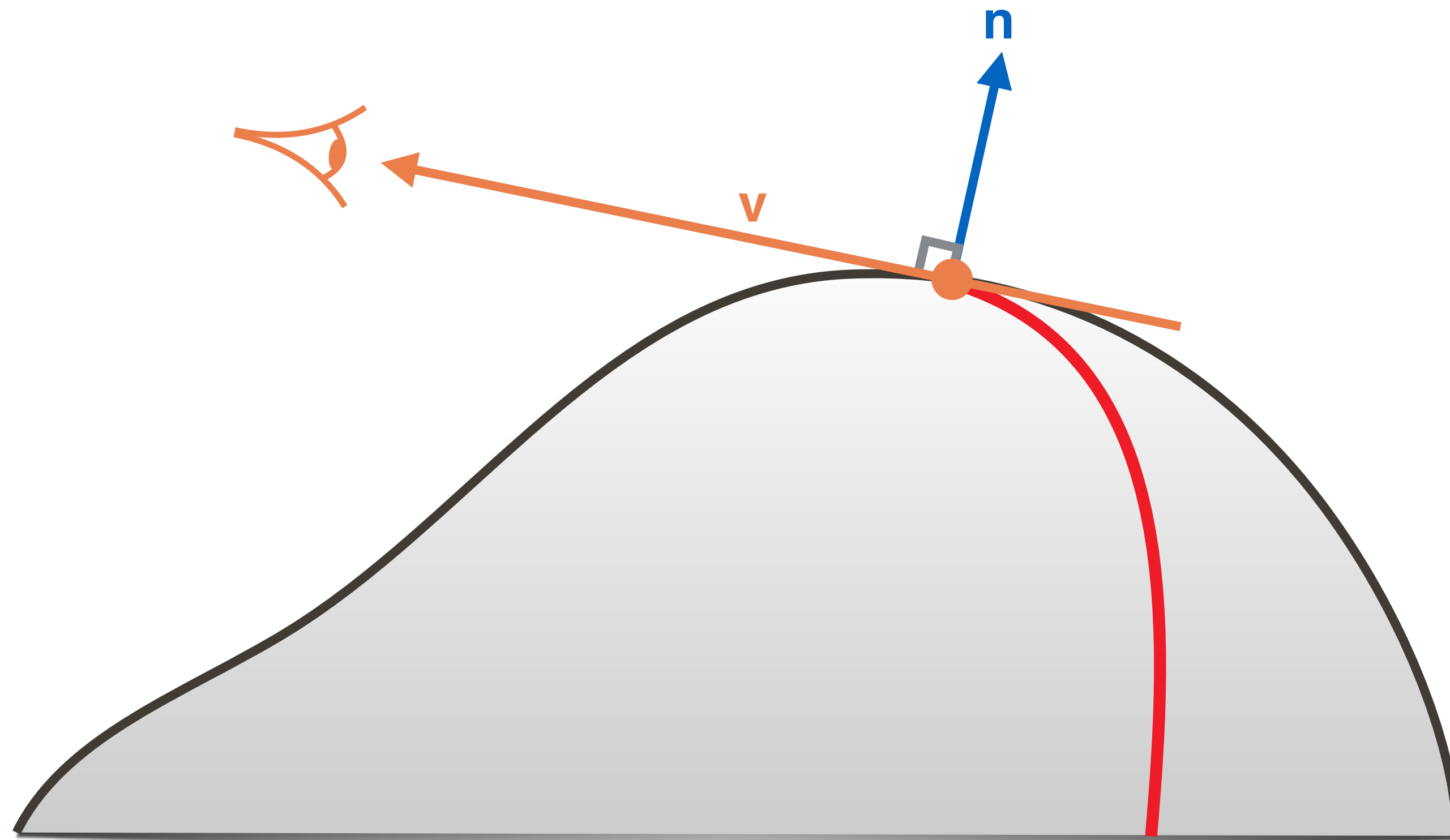
Occluding contours



Stylized rendering

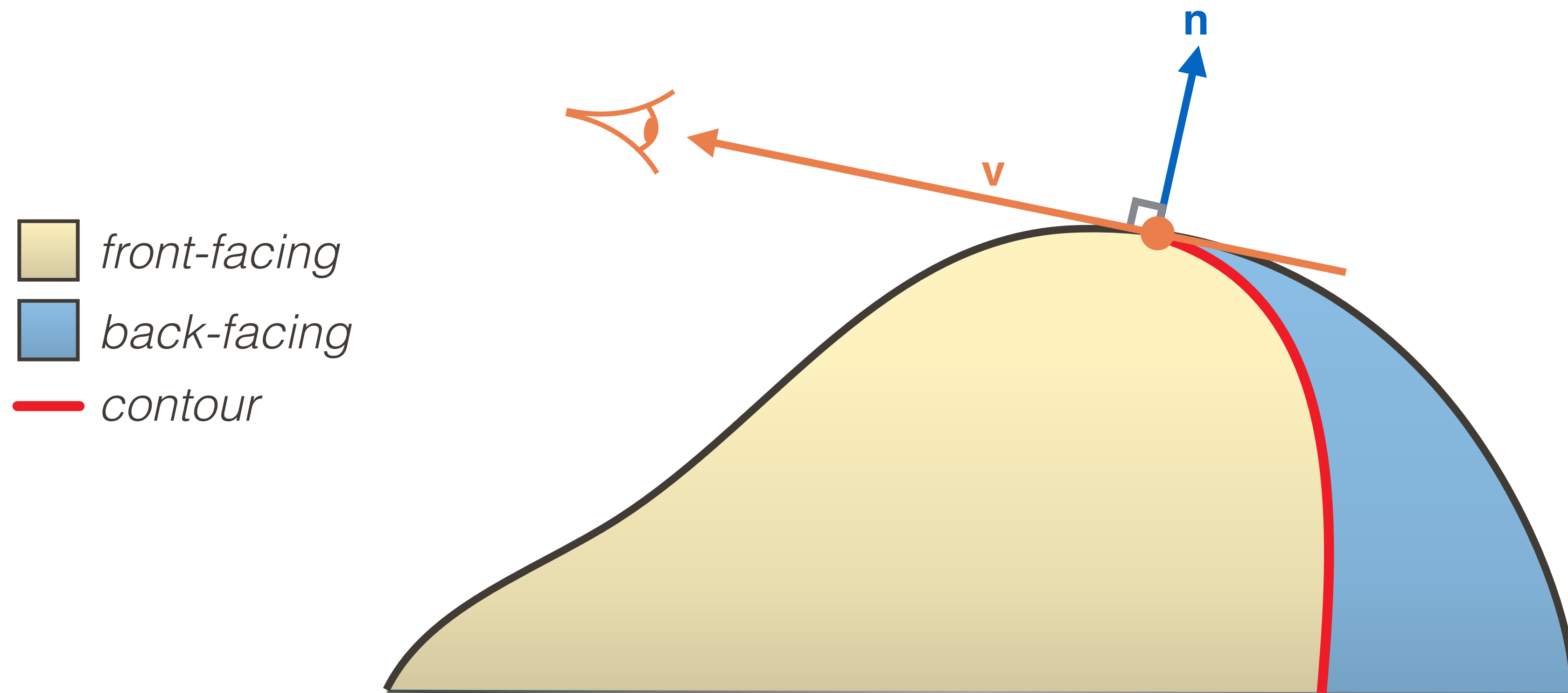
Occluding Contours

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



Occluding Contours

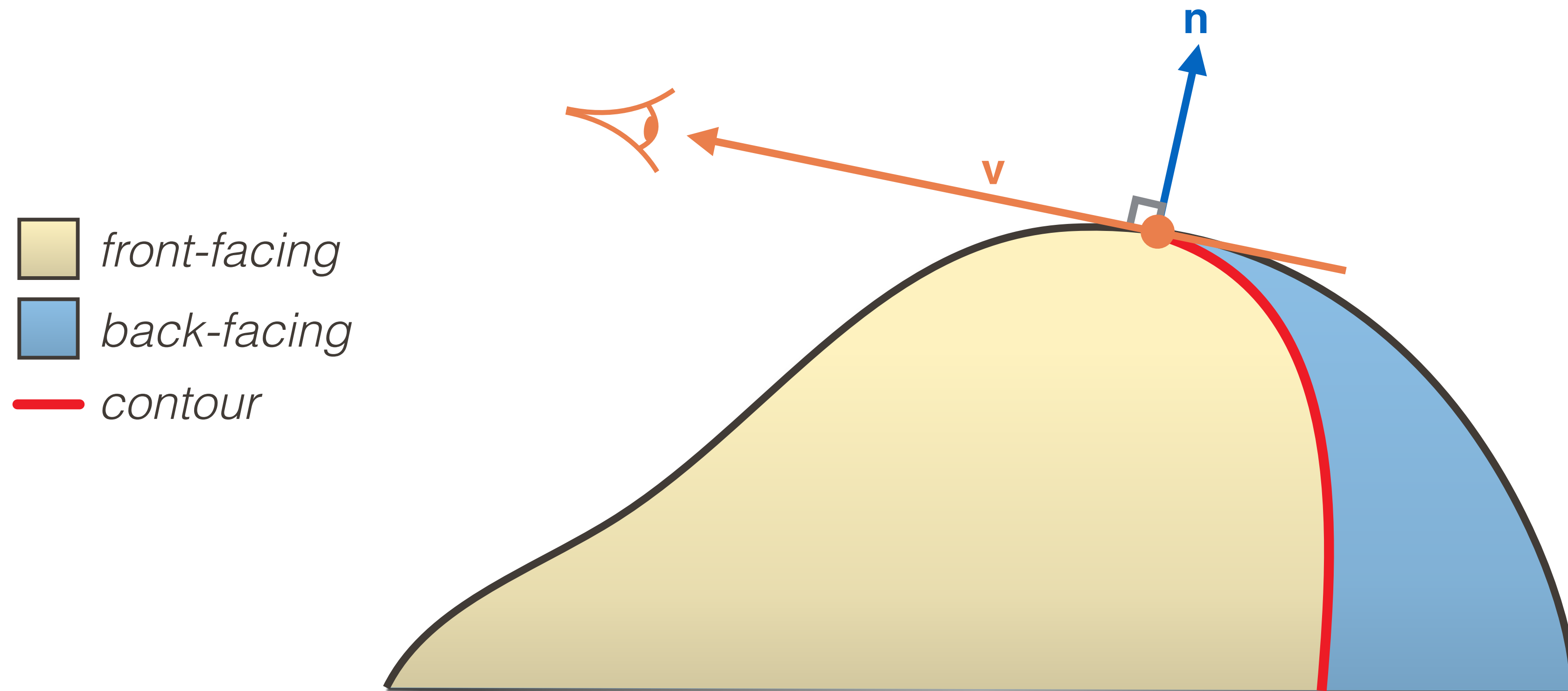
Definition: for a **smooth surface**, points at which $\mathbf{n} \cdot \mathbf{v} = 0$



Occluding Contours

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)

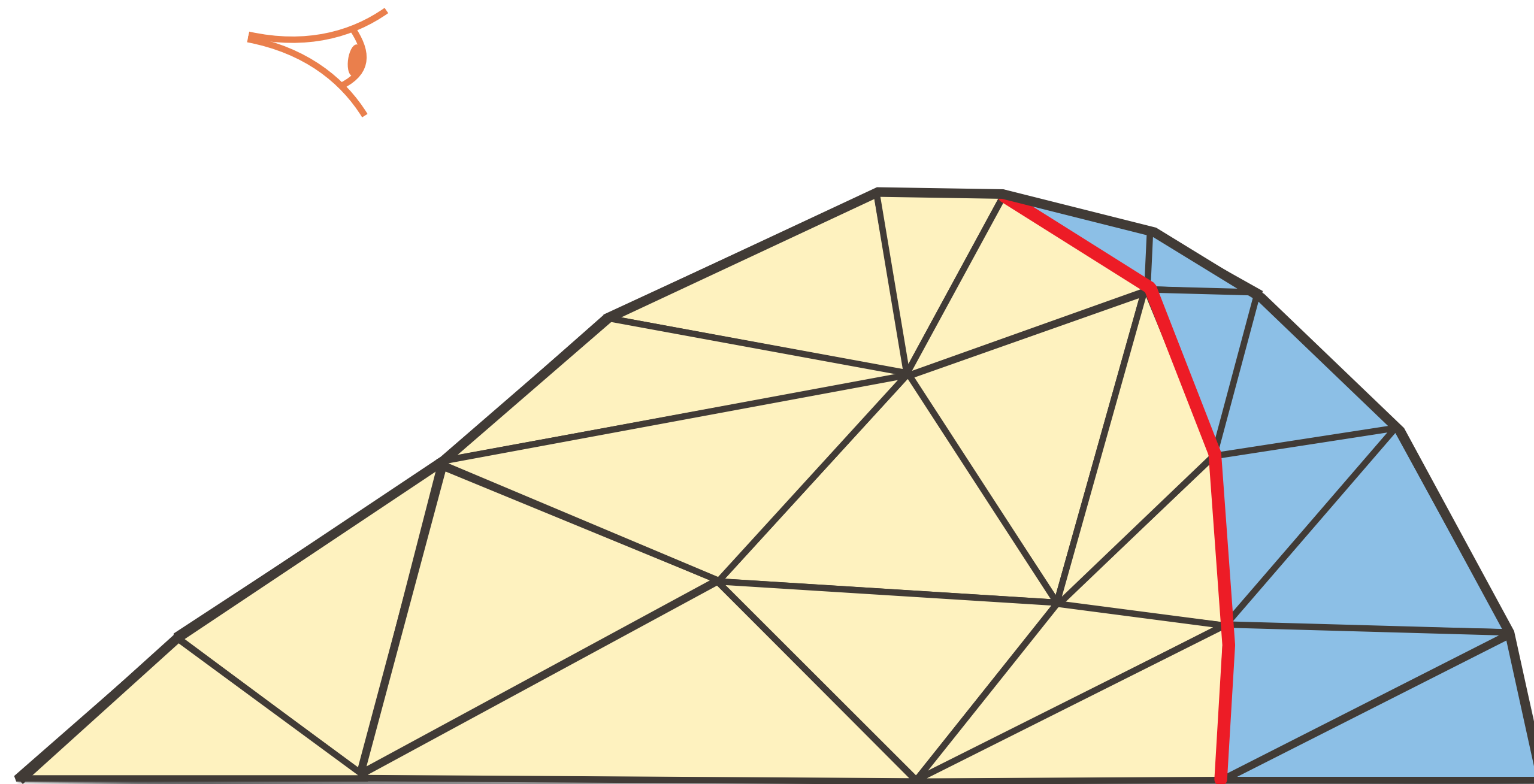


Mesh Contours

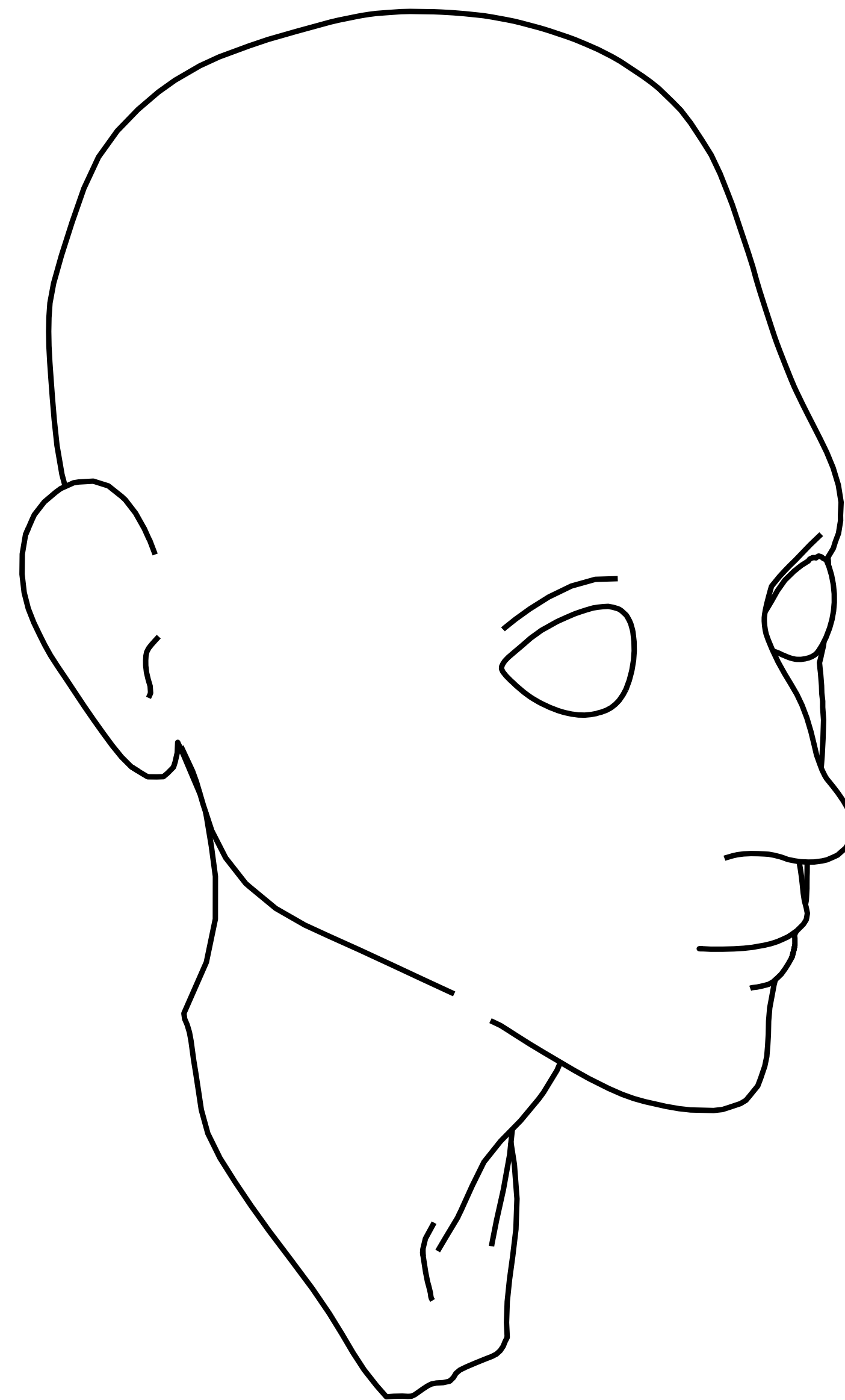
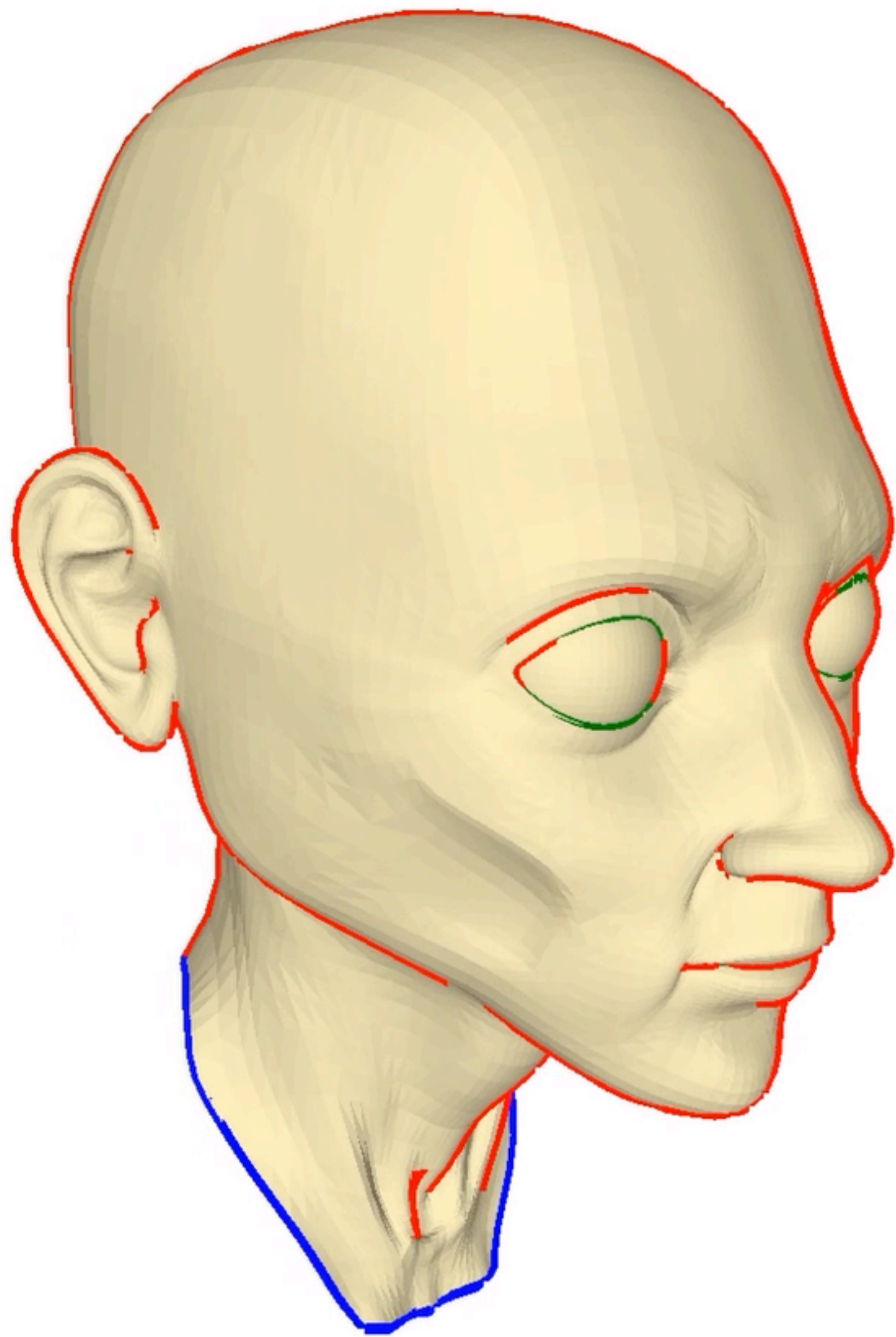
Definition: for a **discrete surface**, edges between front- and back-faces

⇒ Widely used in practice (simple and fast)

 *front-facing*
 *back-facing*
 *contour*

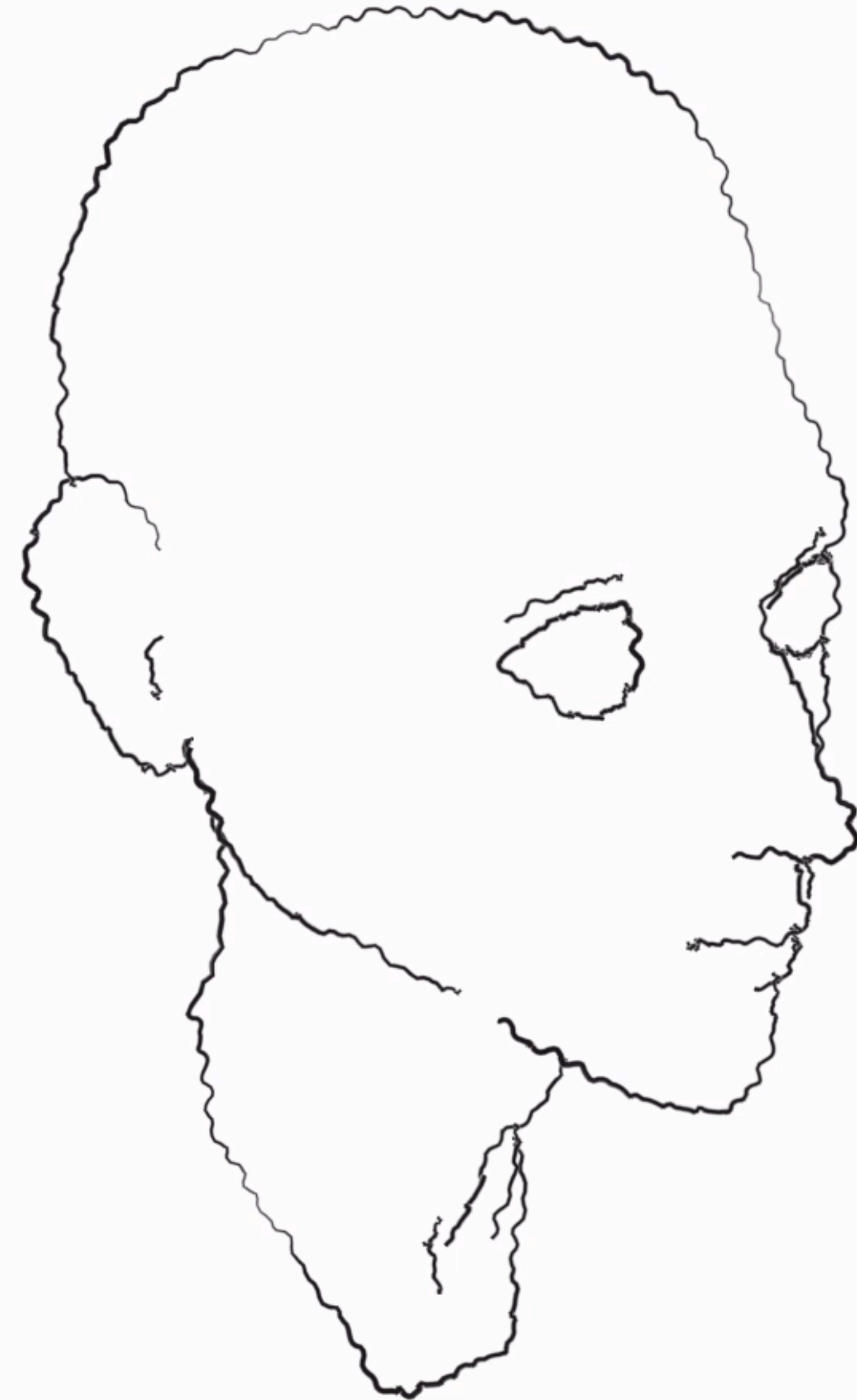


Mesh Contours



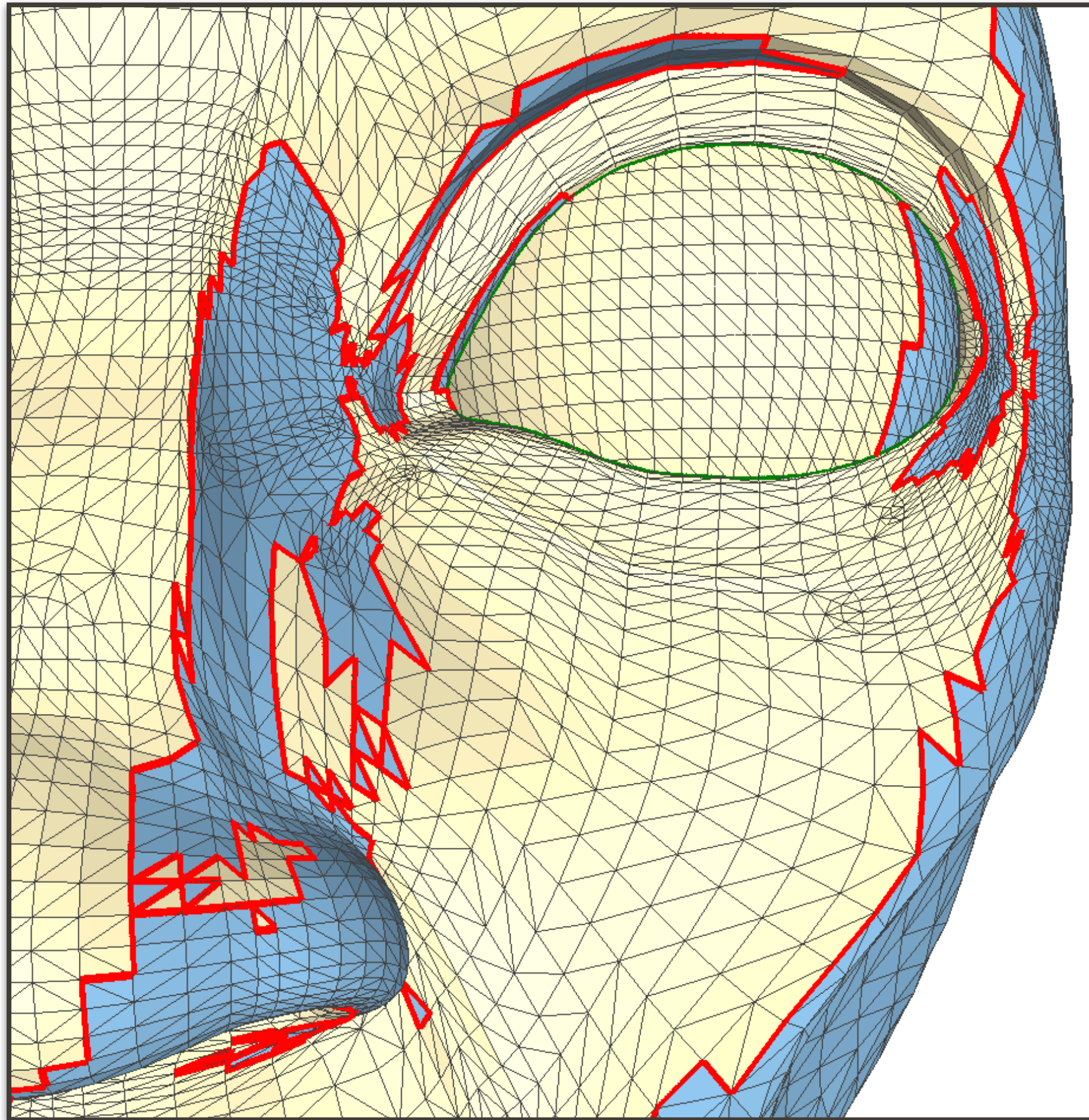
Stylized Mesh Contours

Spatial and **temporal** artifacts
(gaps, breaks)



Mesh Contours

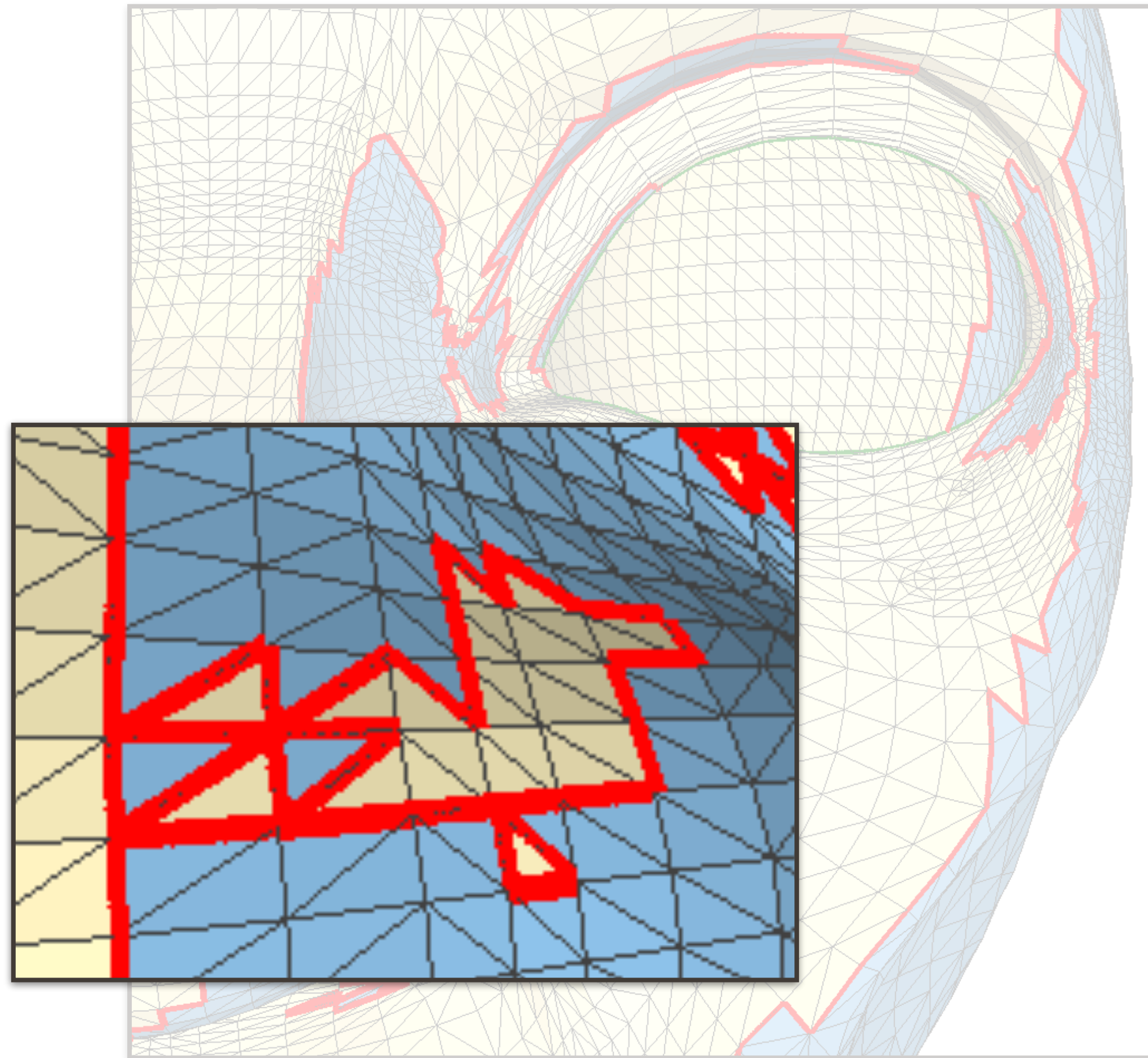
Issue: Overly **complex topology** (loops)



Side view

Mesh Contours

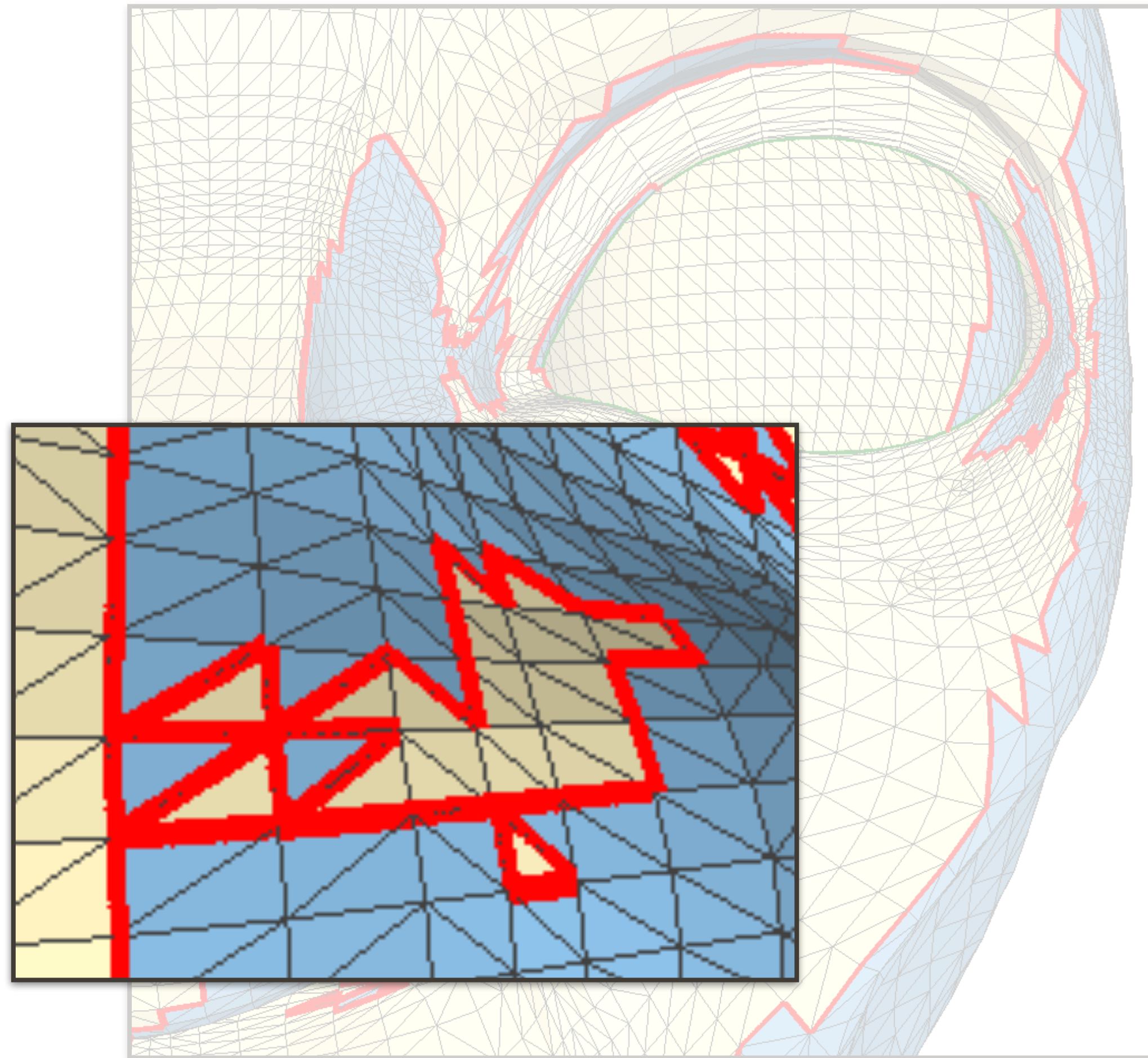
Issue: Overly **complex topology** (loops)



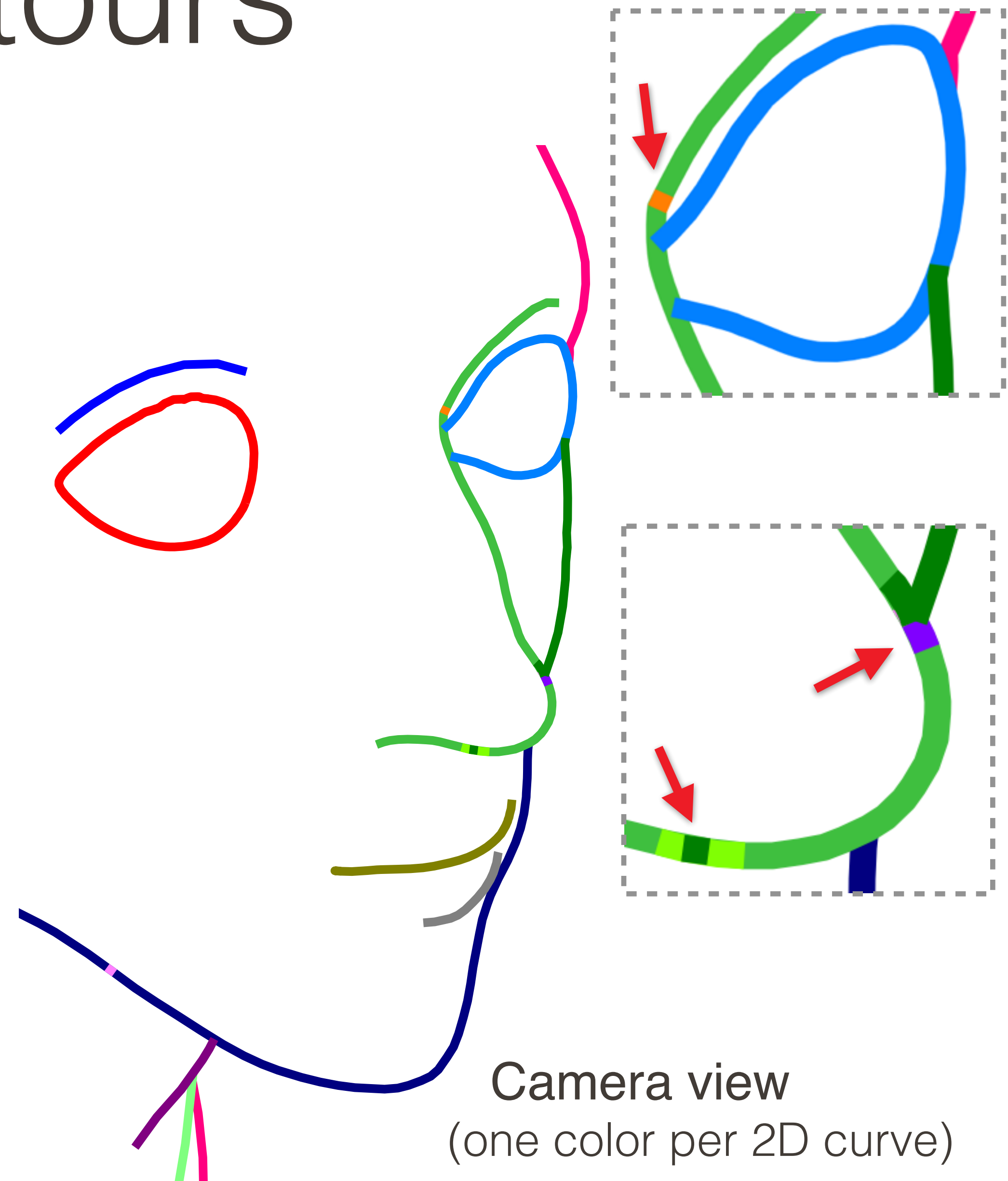
Side view

Mesh Contours

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



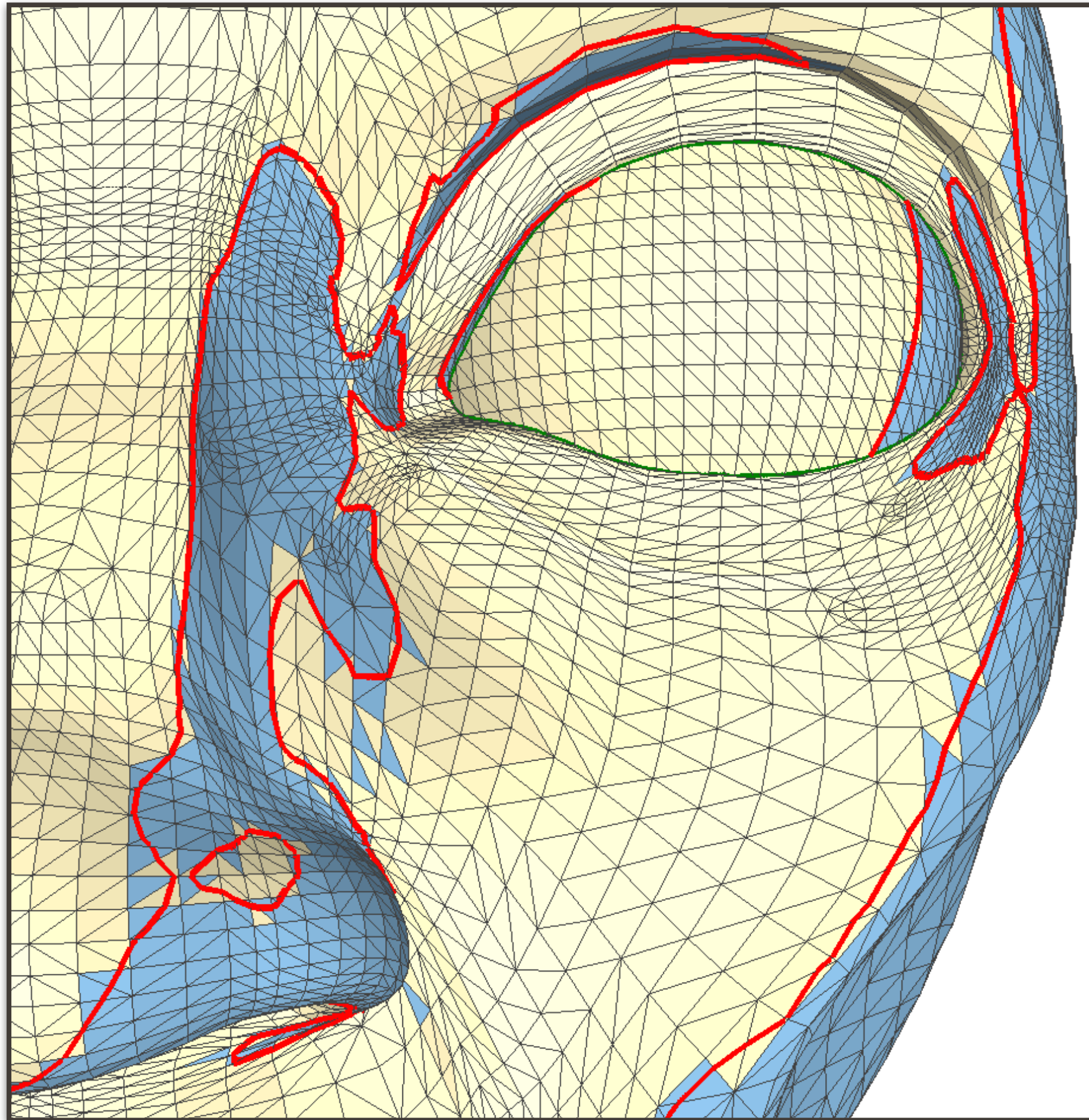
Side view



Interpolated Contours

[Hertzmann and Zorin 2000]

Issue: Surface-contour **visibility mismatched**

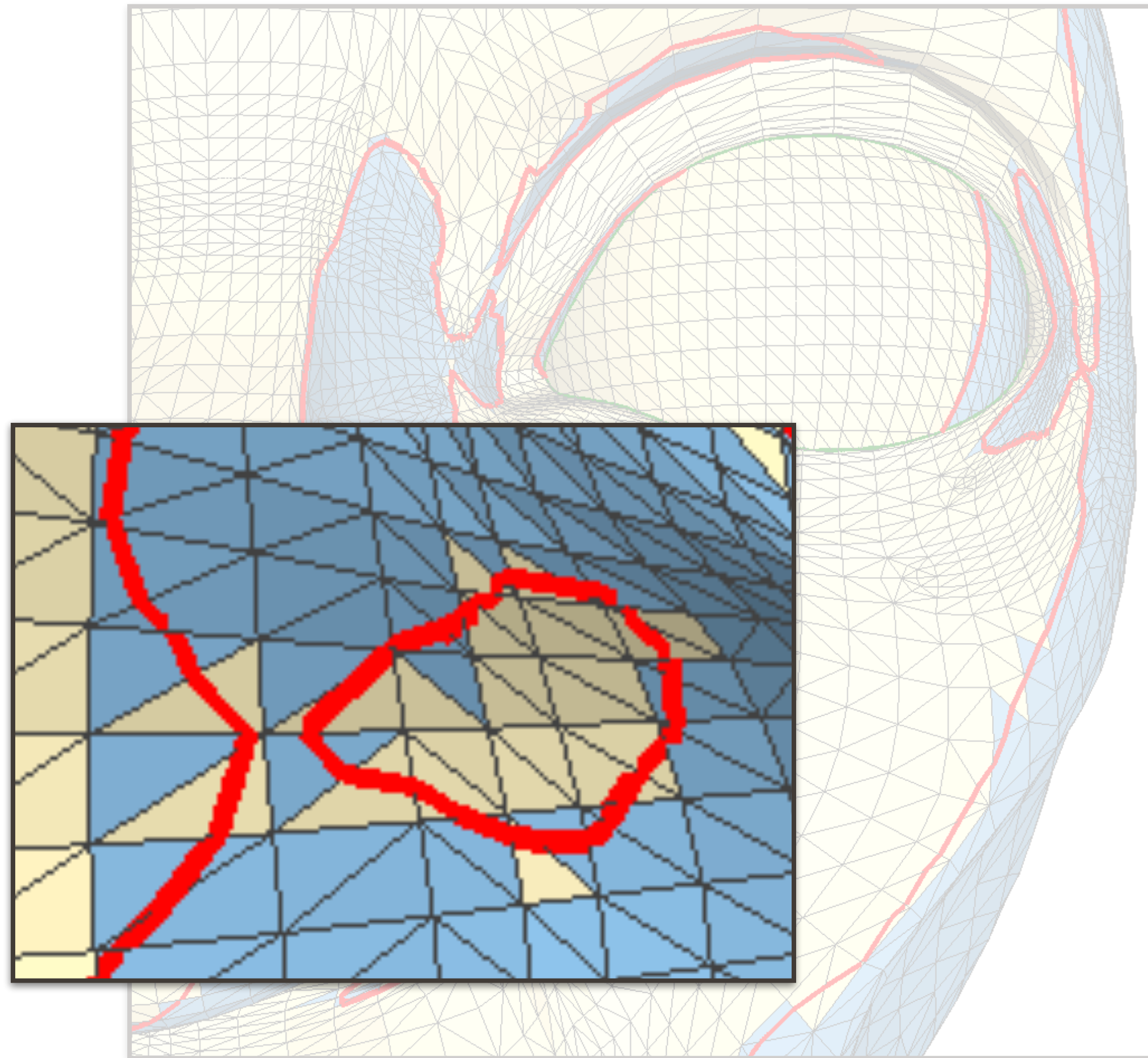


Side view

Interpolated Contours

[Hertzmann and Zorin 2000]

Issue: Surface-contour **visibility mismatched**

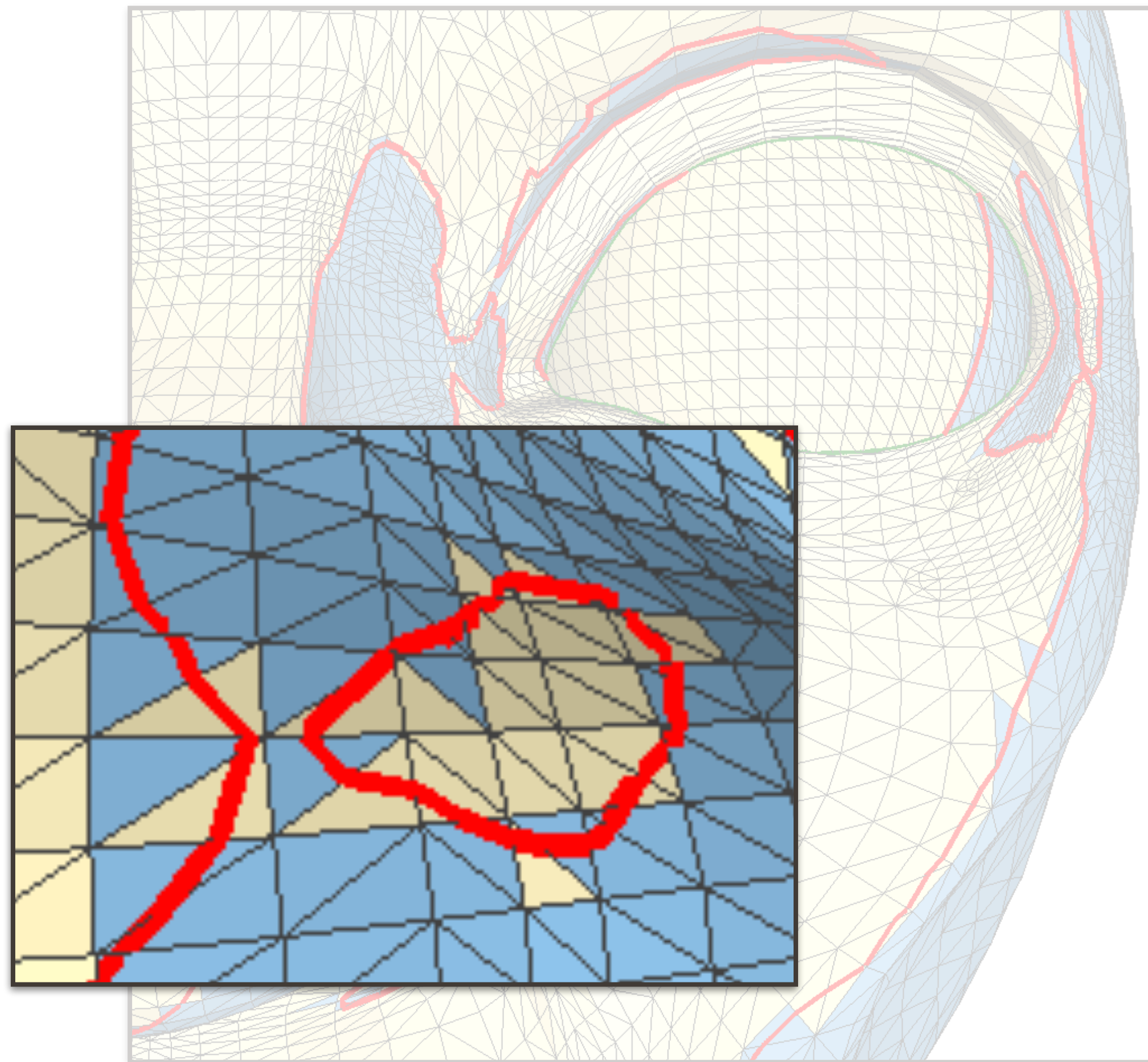


Side view

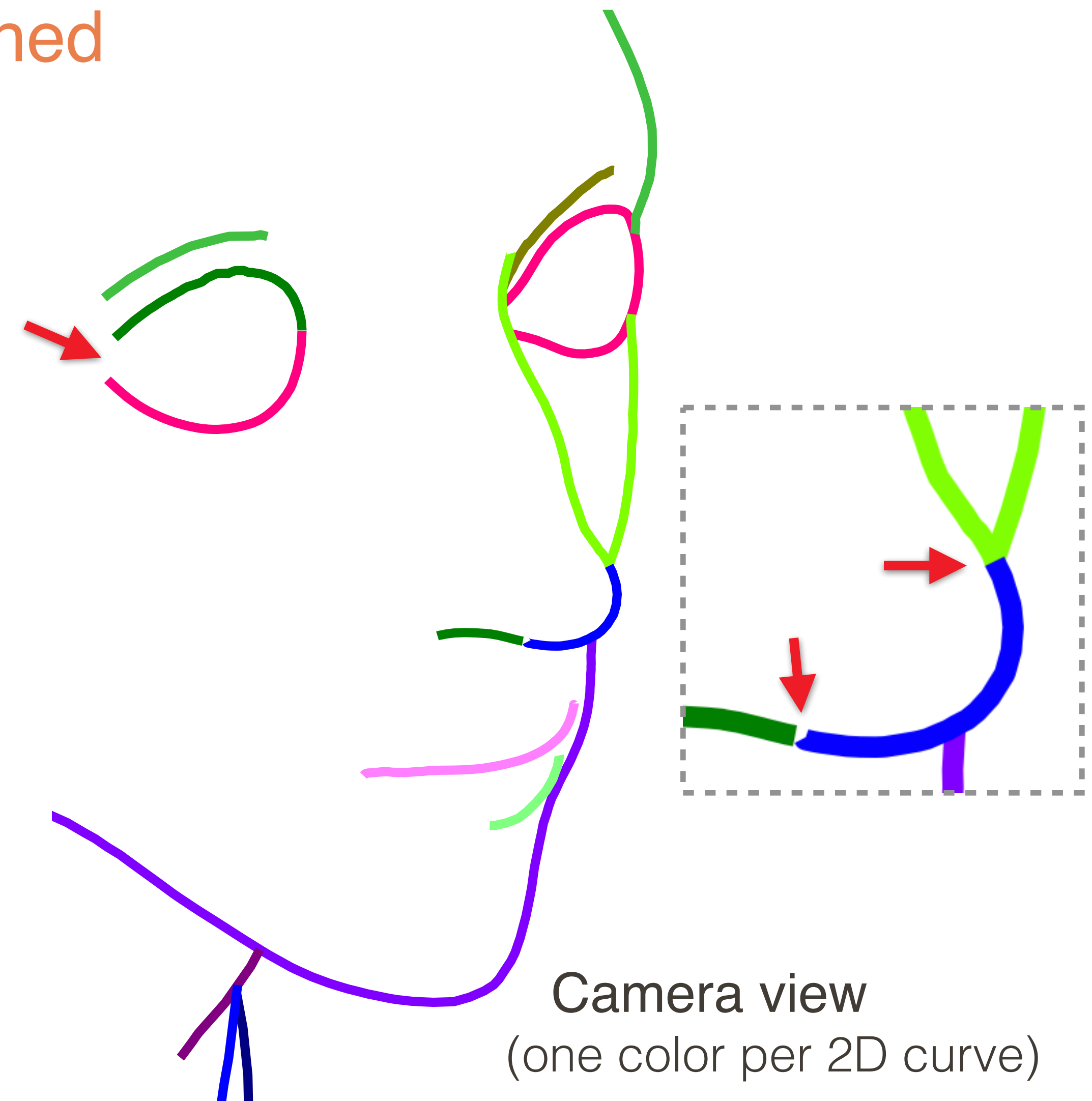
Interpolated Contours

[Hertzmann and Zorin 2000]

Issue: Surface-contour **visibility mismatched**



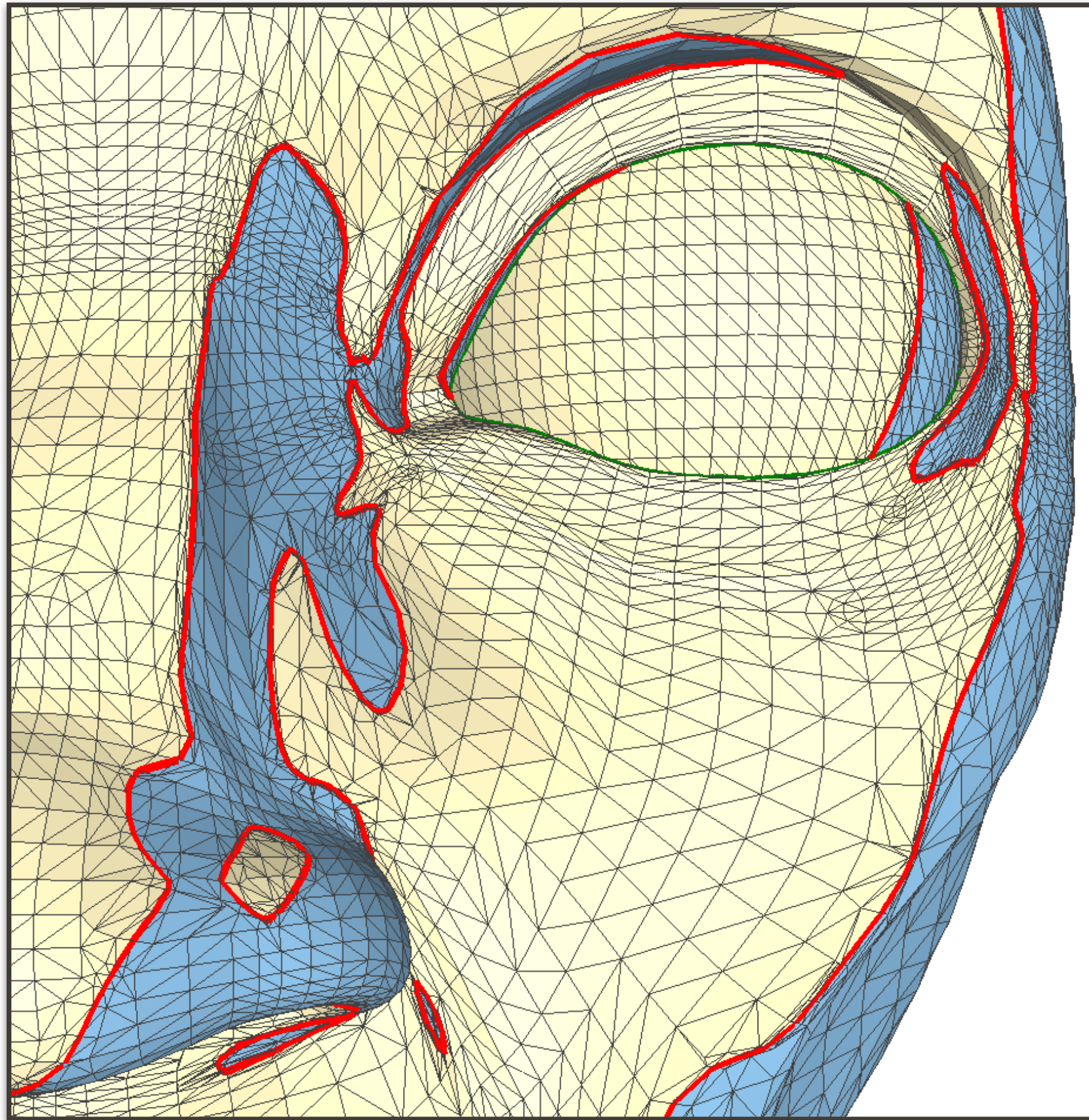
Side view



Camera view
(one color per 2D curve)

Our Method

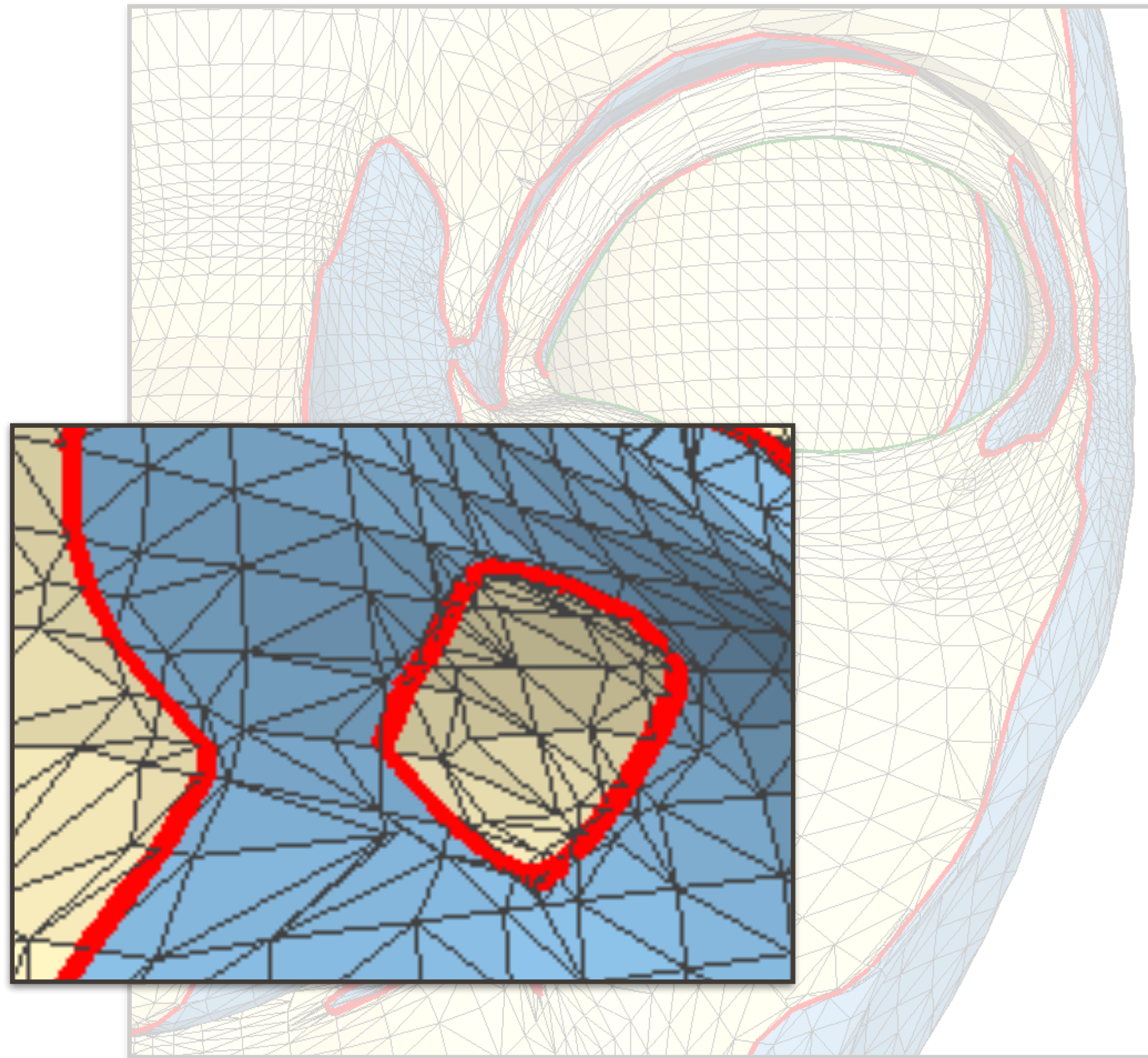
Goal: Mesh contours **partitioning** front- and back-faces



Side view

Our Method

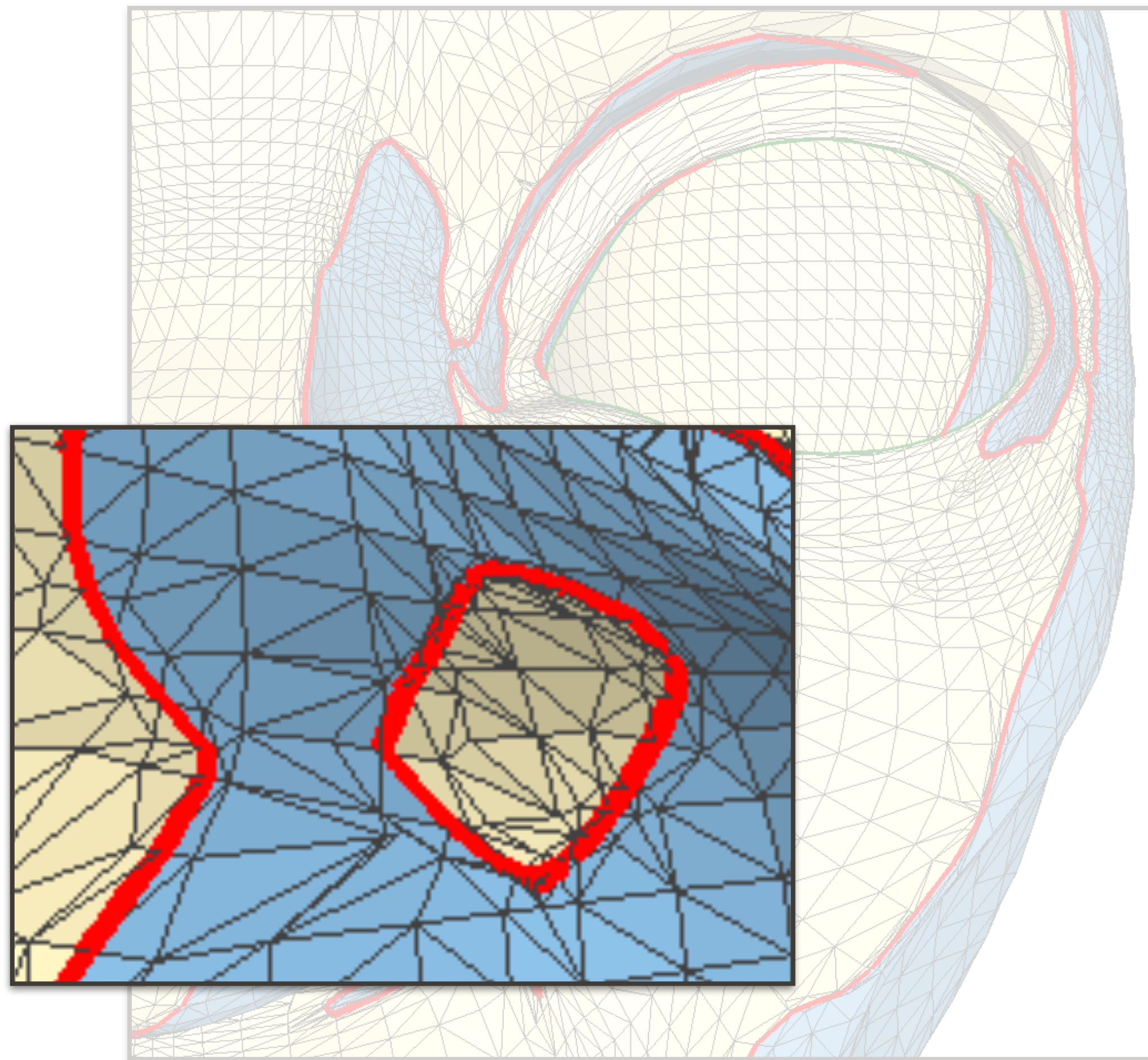
Goal: Mesh contours **partitioning** front- and back-faces



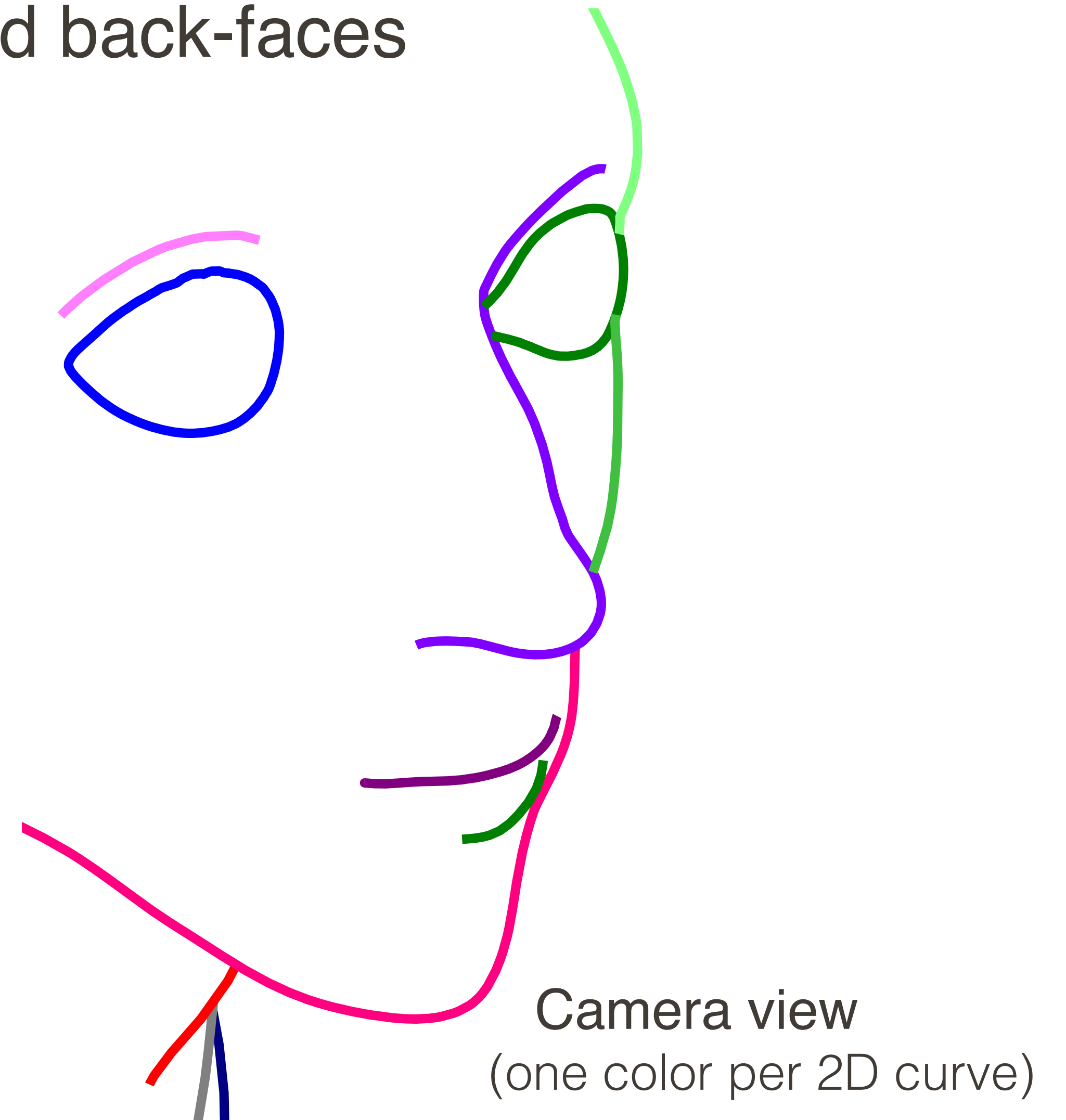
Side view

Our Method

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



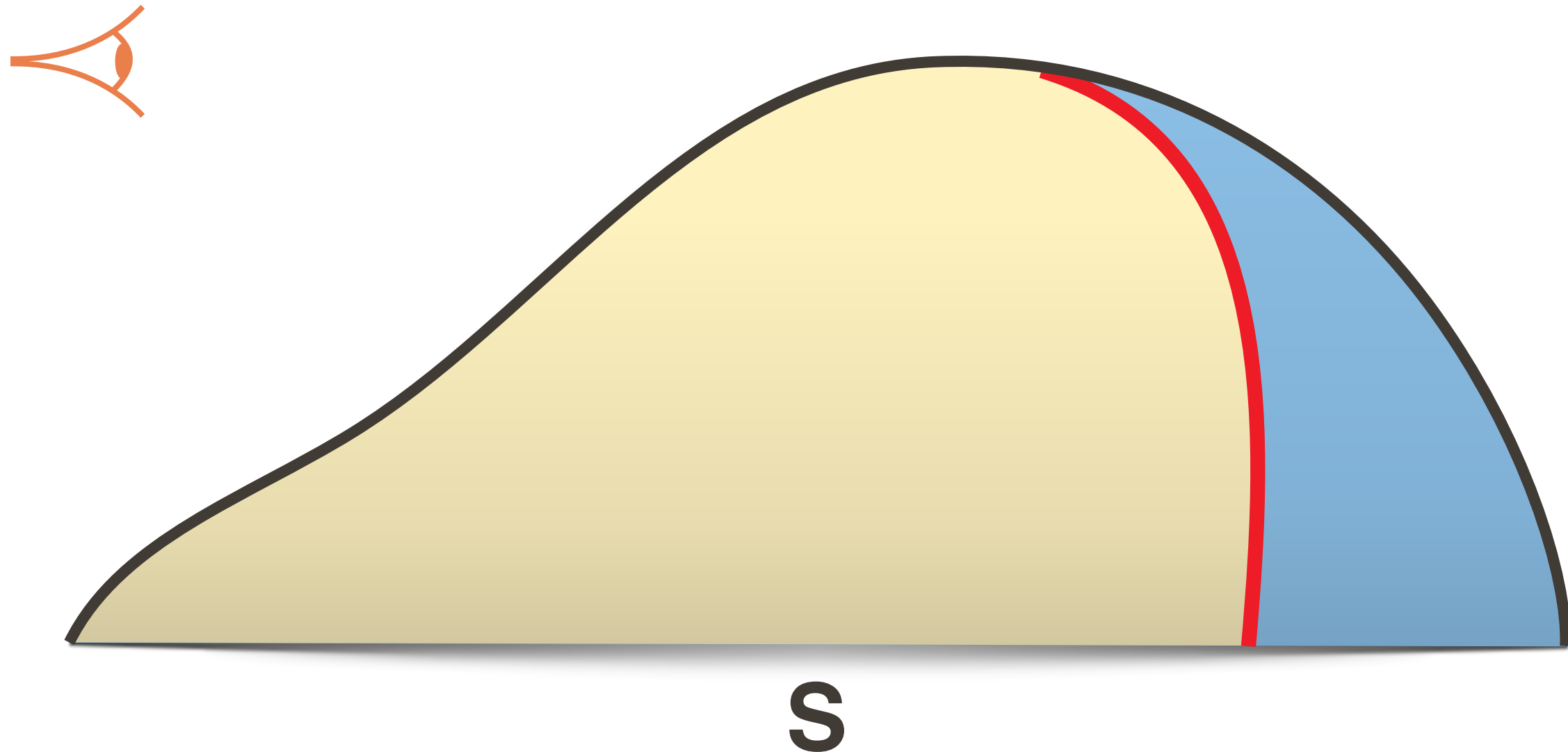
Side view



Camera view
(one color per 2D curve)

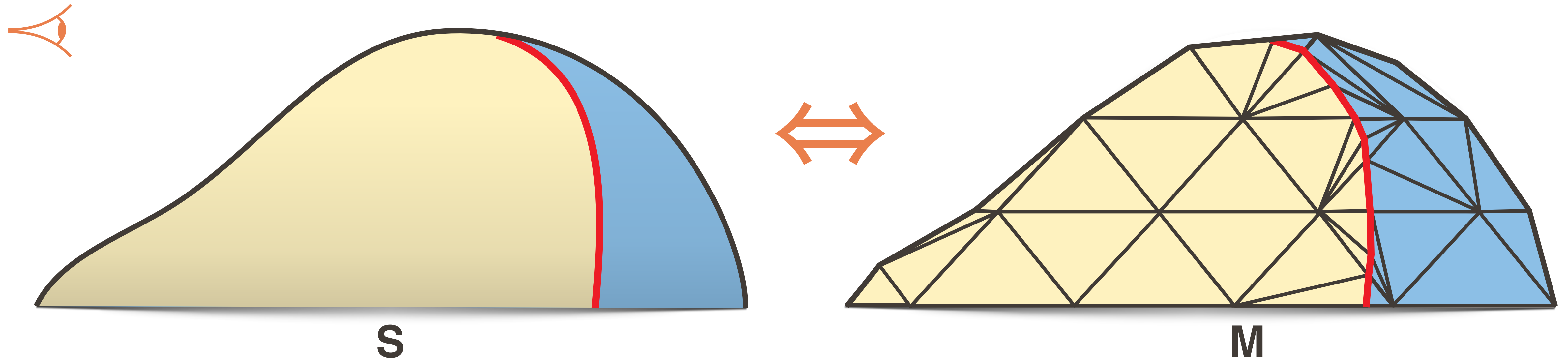
Our Method

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



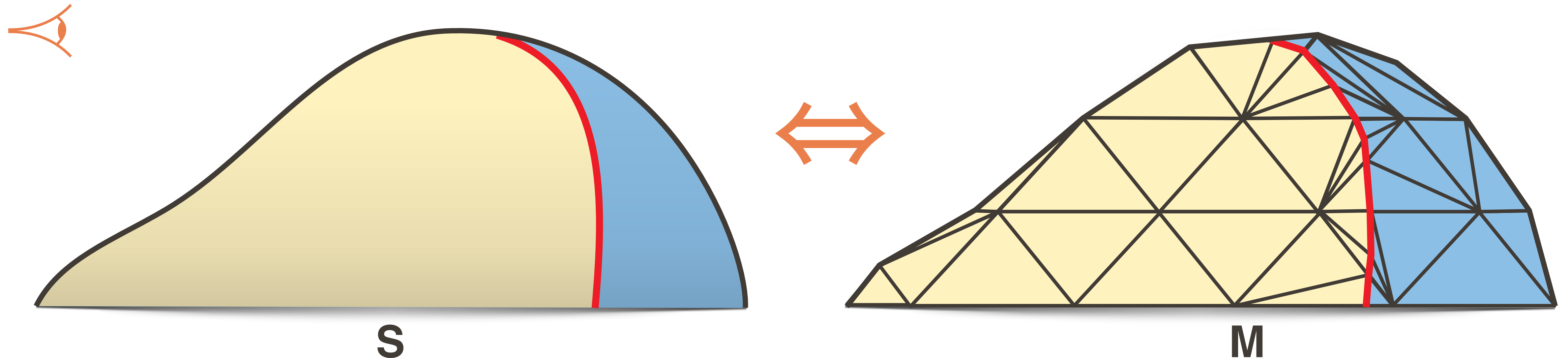
Our Method

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



Our Method

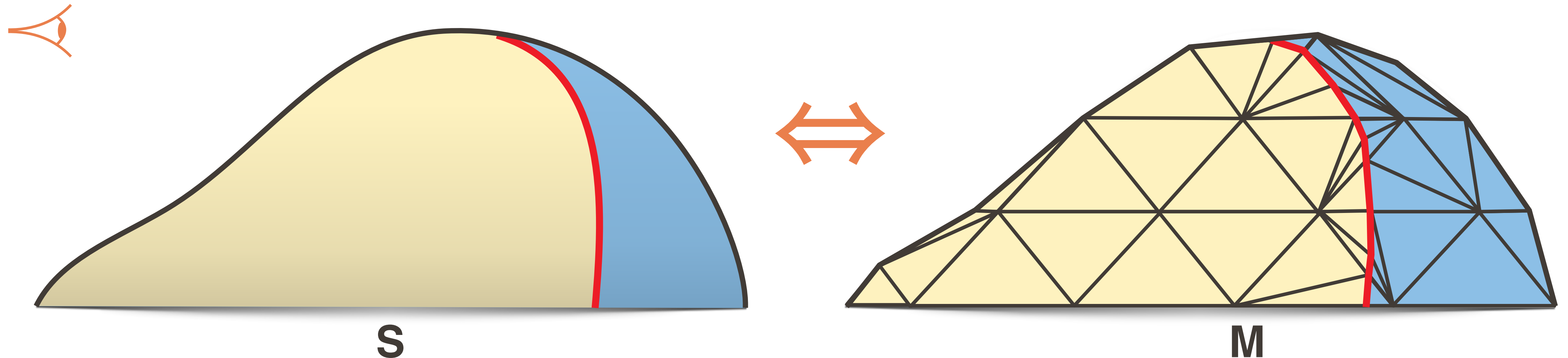
Goal: extract for **each viewpoint** a mesh **M** whose contours are **equivalent** to the contours of **S**.



Difficult, **global property**

Our Method

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

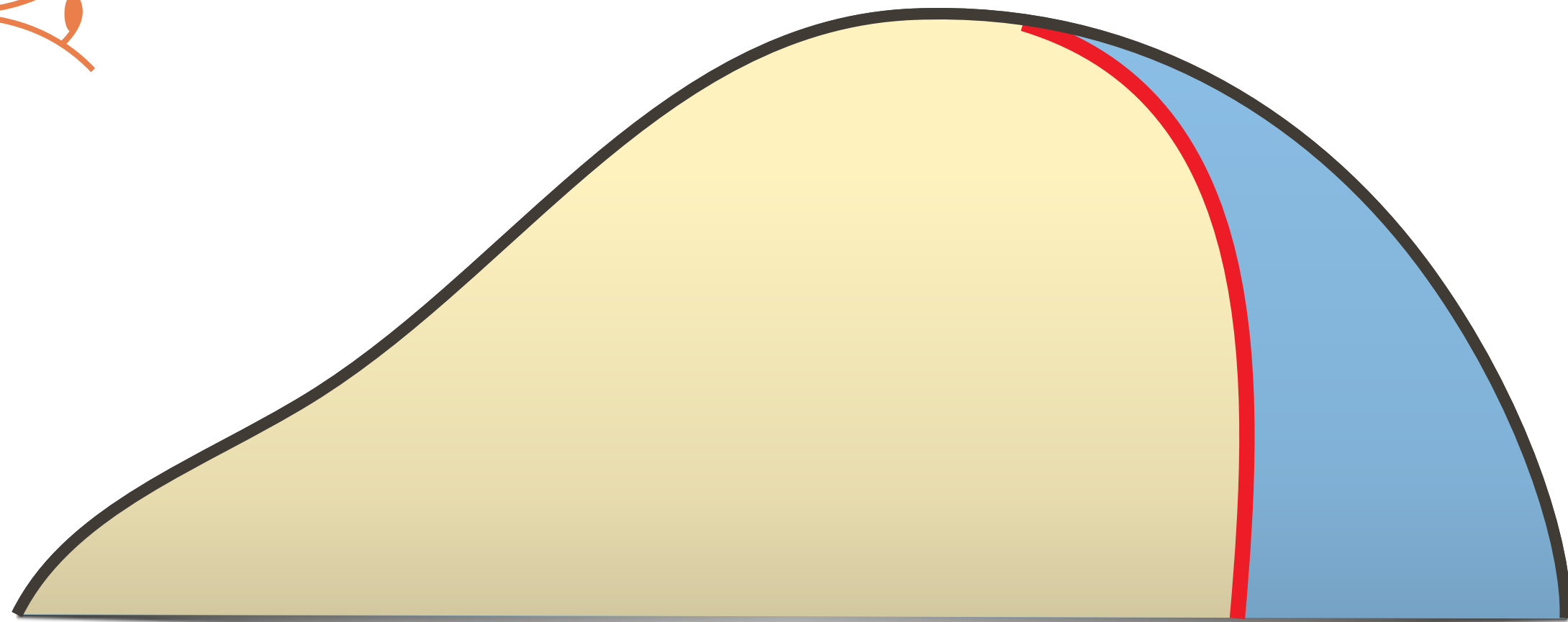
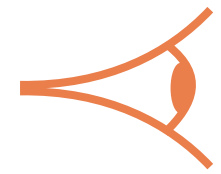


Difficult, **global property**

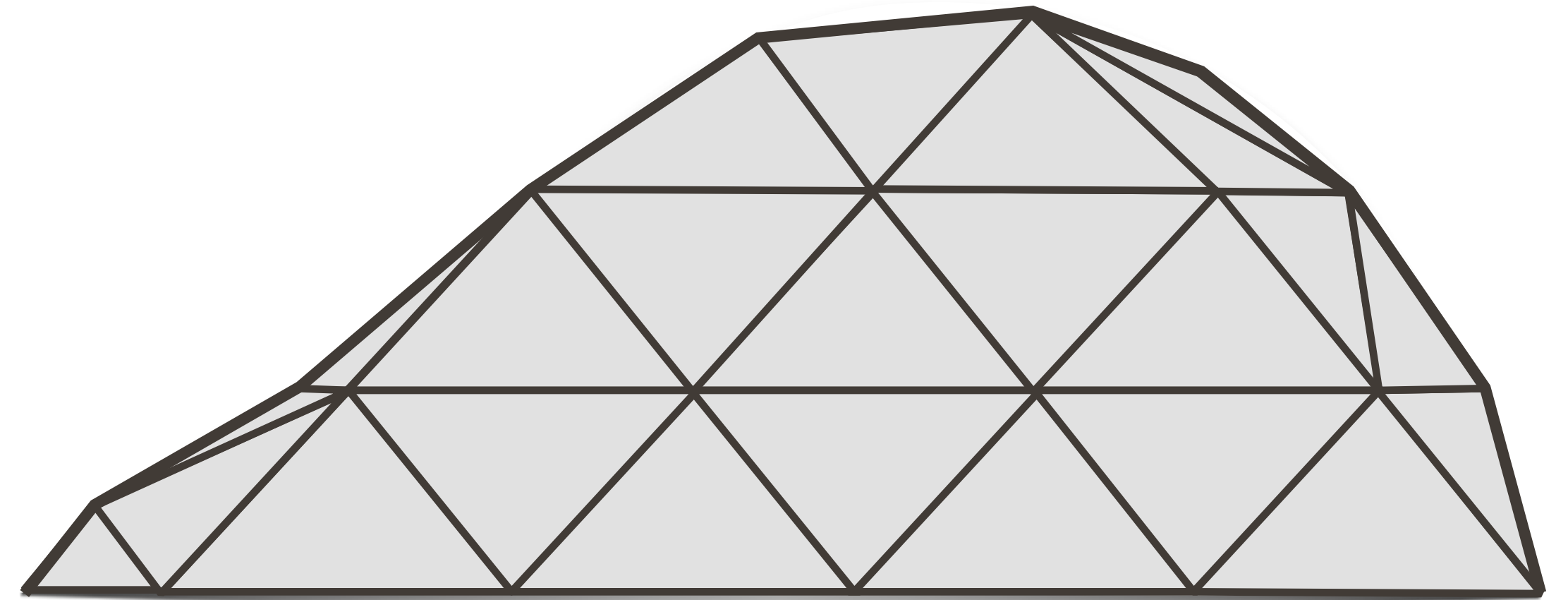
➔ local criterion: **Contour-Consistency**

Contour-Consistency

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



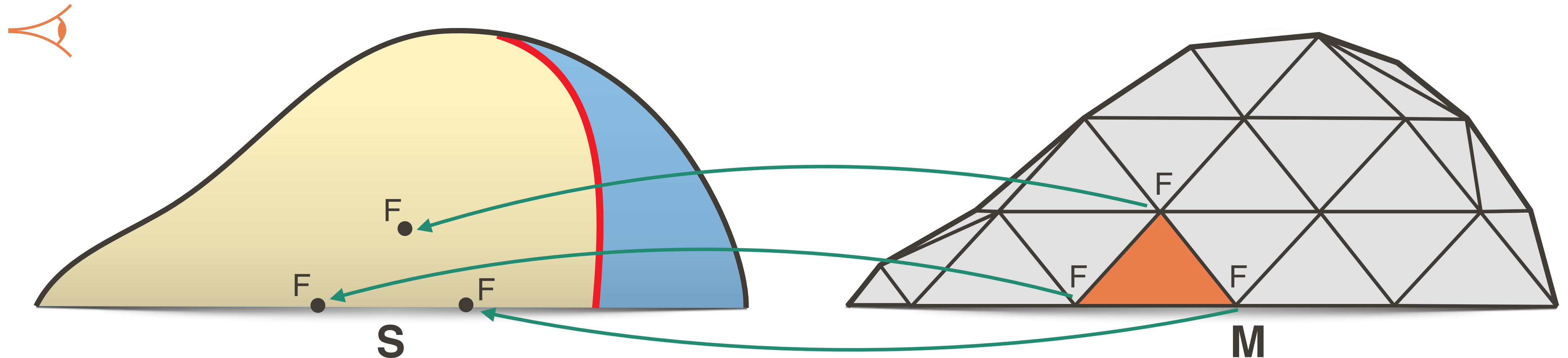
S



M

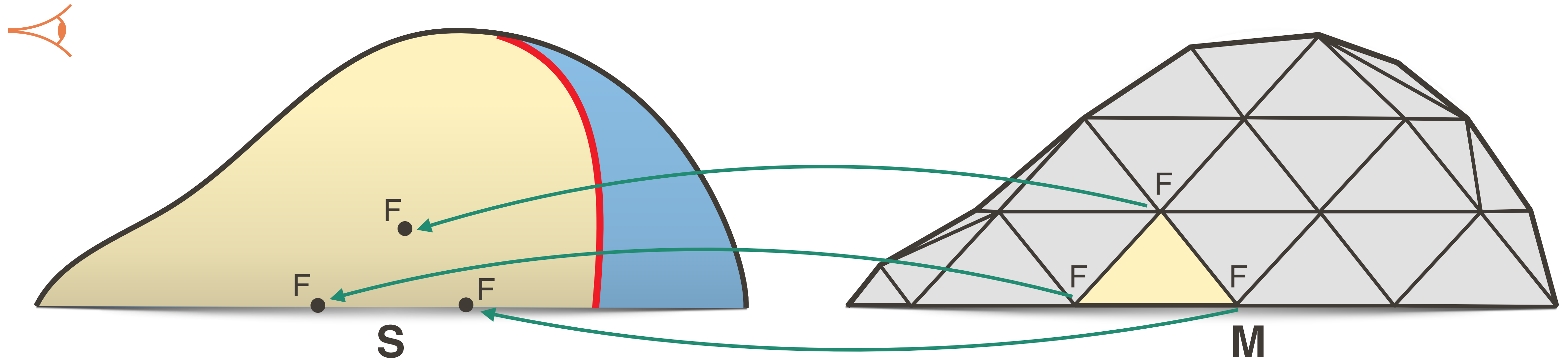
Contour-Consistency

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



Contour-Consistency

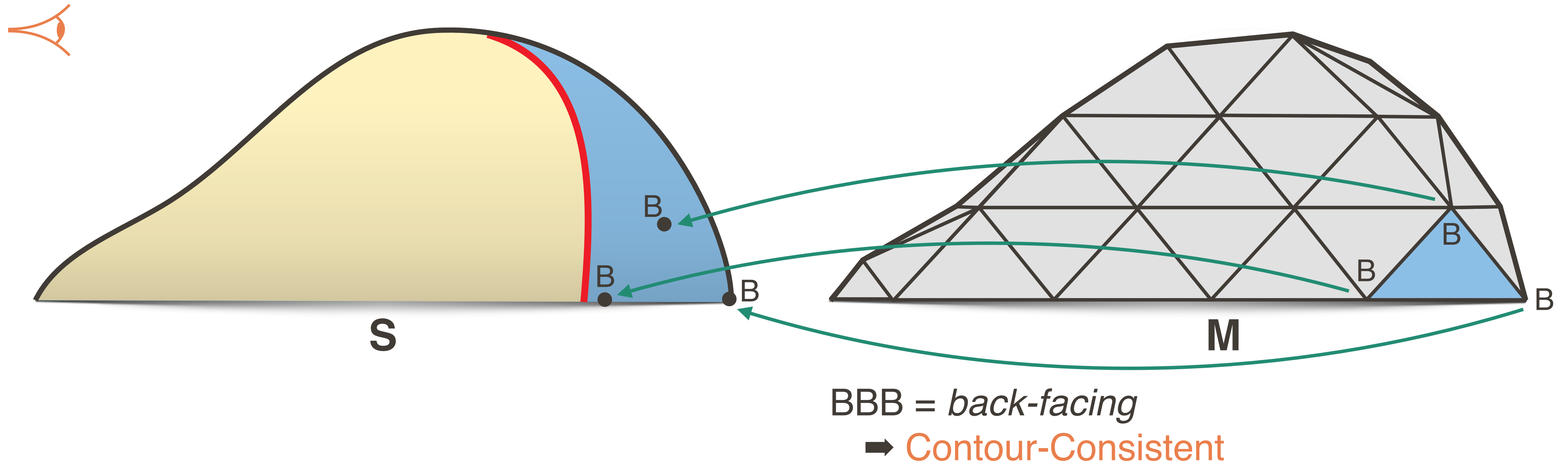
Determines if triangle's orientation **matches** smooth surface's orientation



FFF = *front-facing*
➔ **Contour-Consistent**

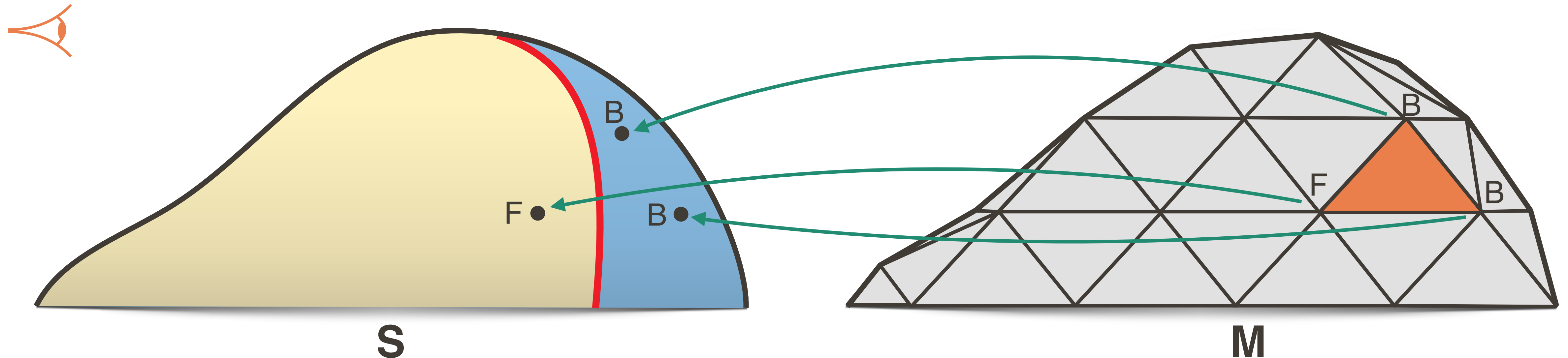
Contour-Consistency

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



Contour-Consistency

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

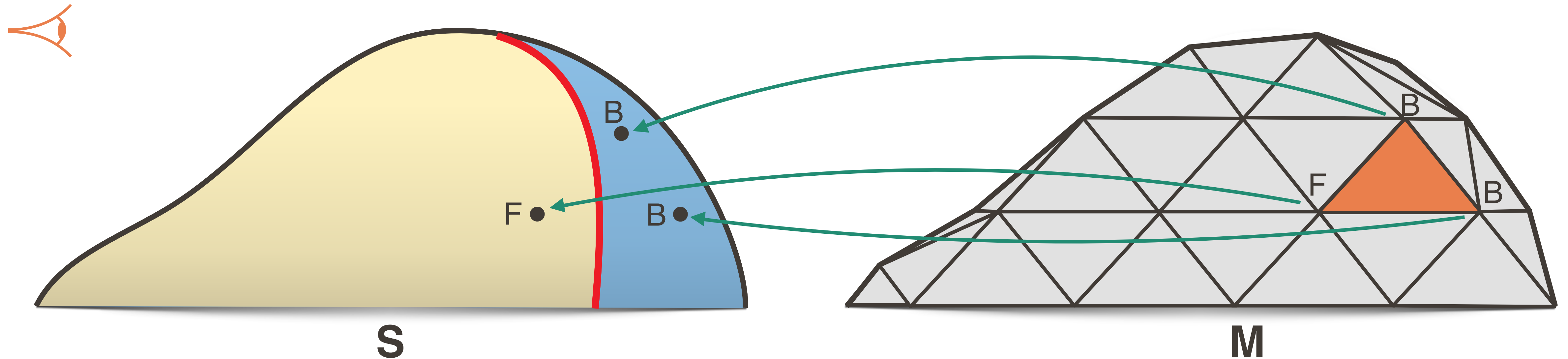


FFB

➡ **Not** Contour-Consistent

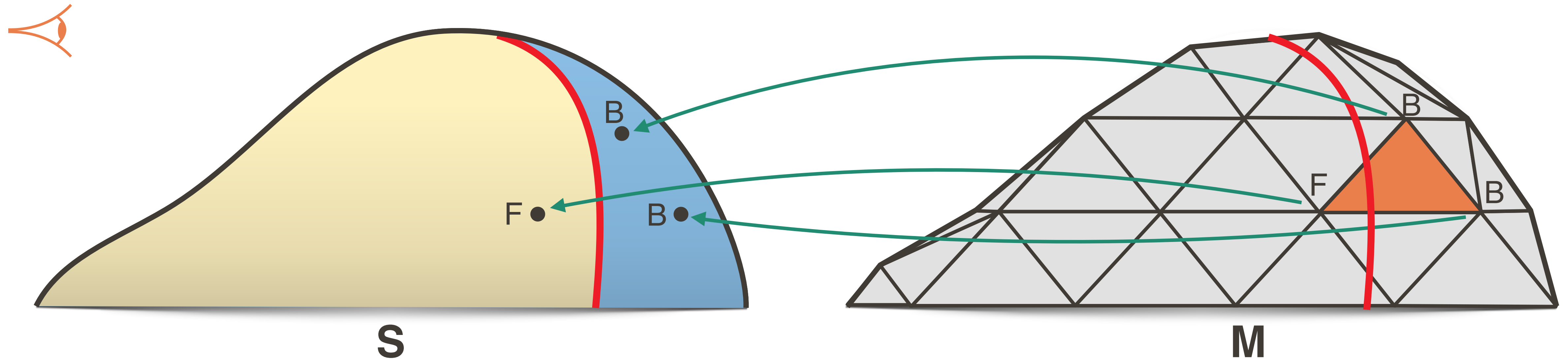
Contour insertion

Insert **contour edges** into the mesh.



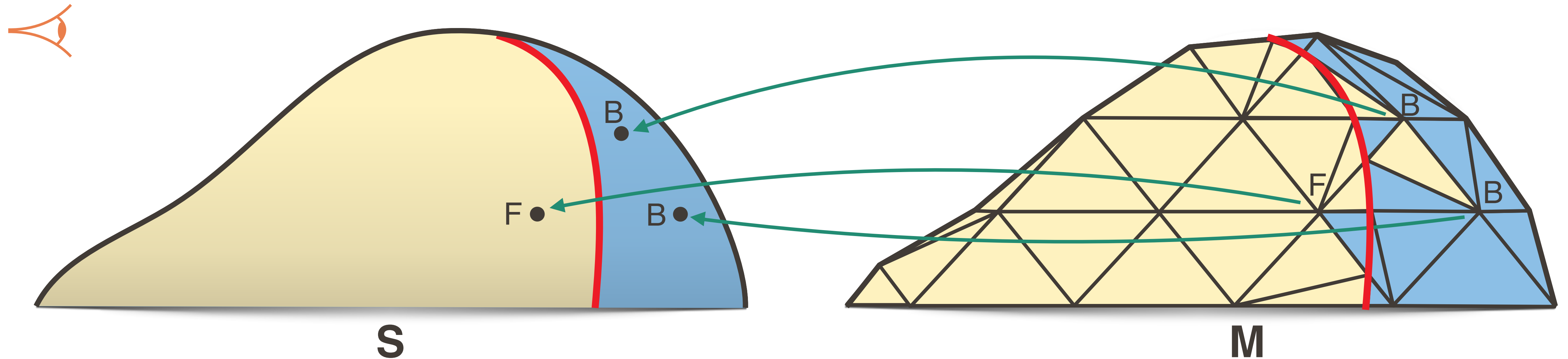
Contour insertion

Insert **contour edges** into the mesh.



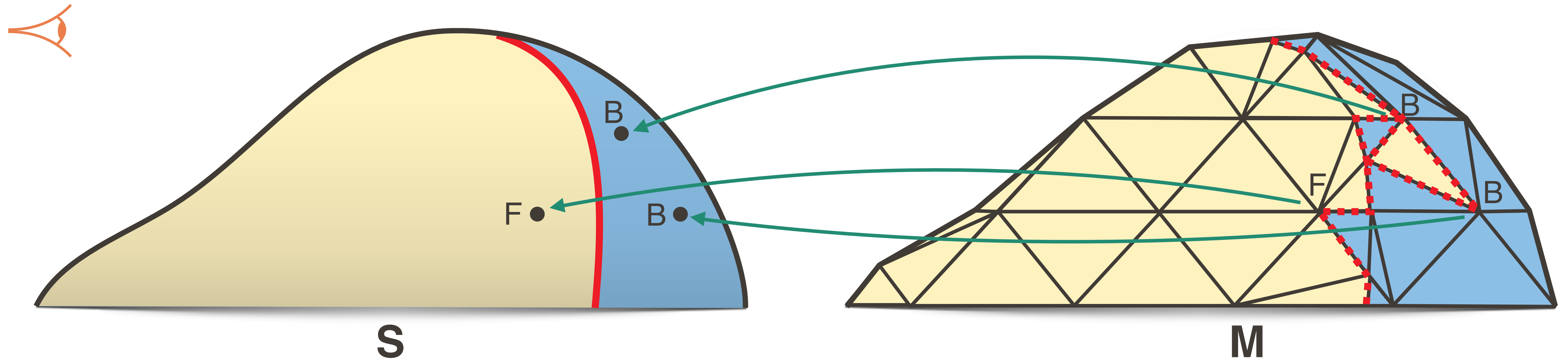
Contour insertion

Insert **contour edges** into the mesh.



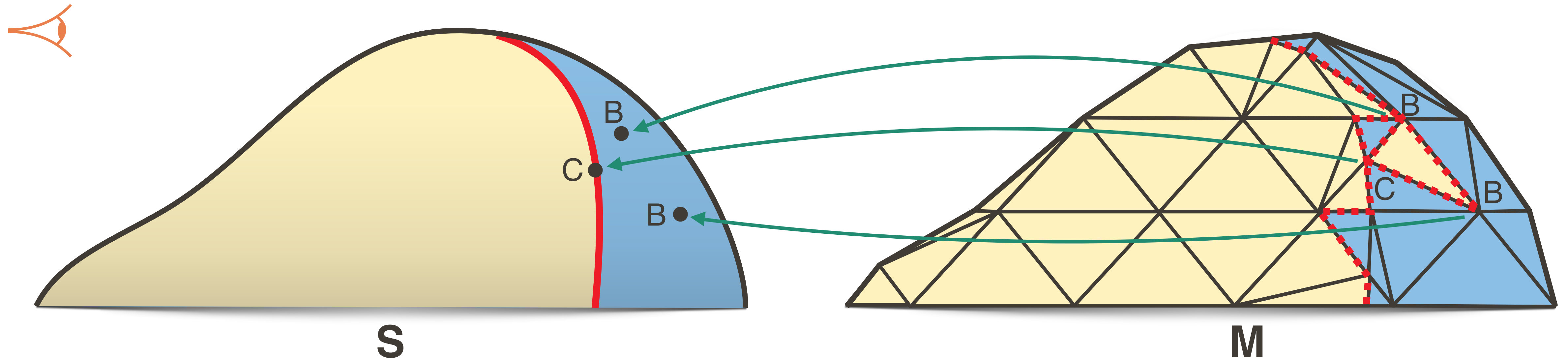
Contour insertion

Insert **contour edges** into the mesh.



Contour insertion

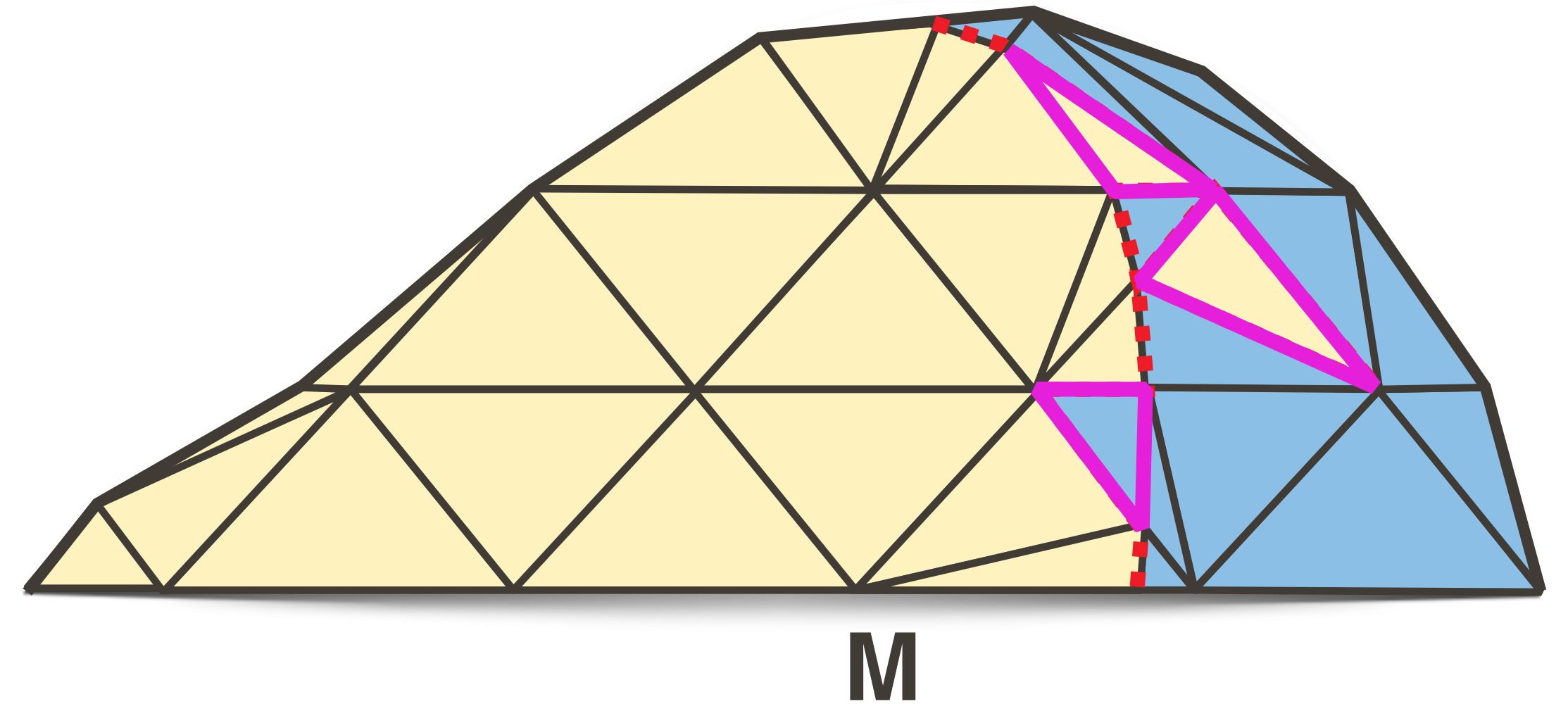
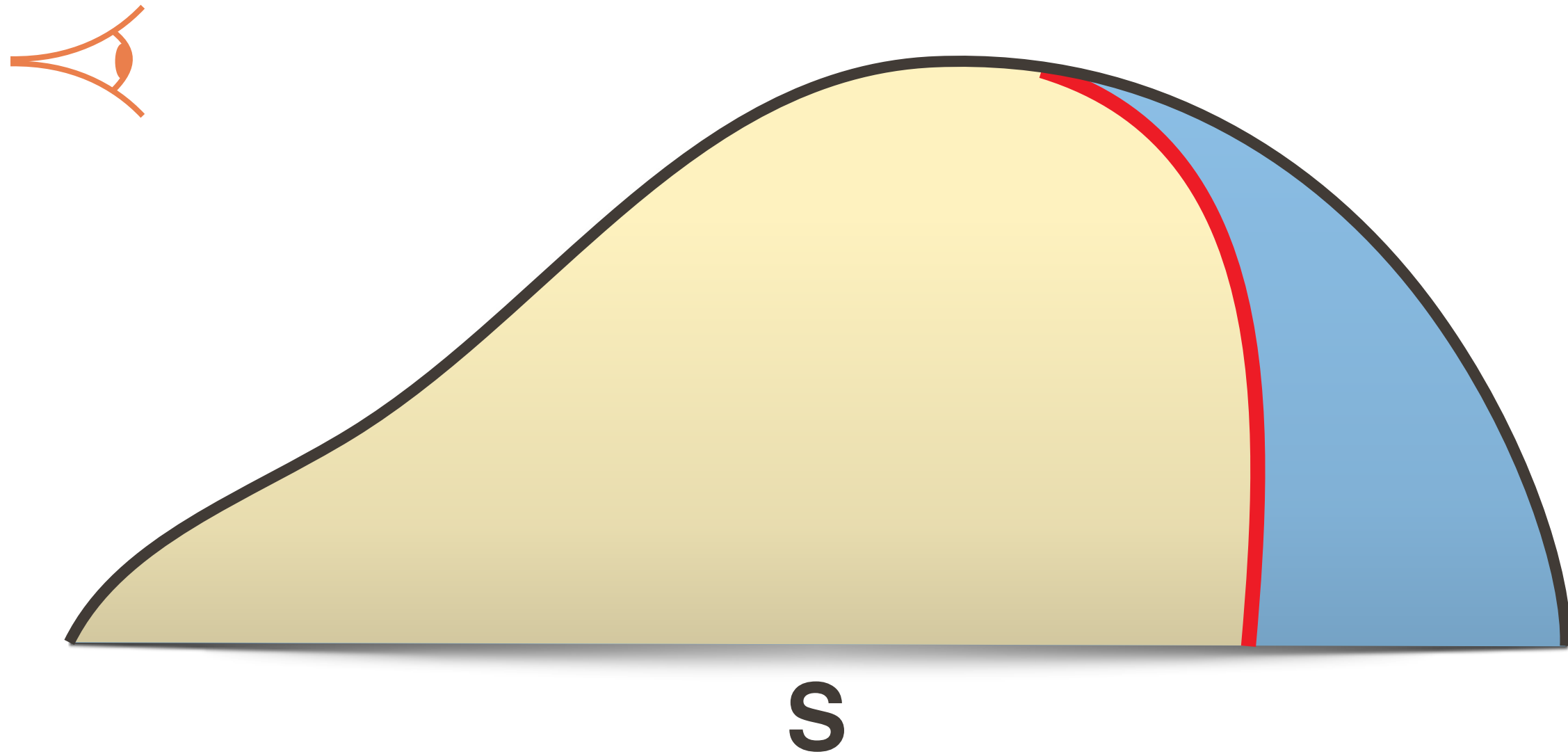
Insert **contour edges** into the mesh.



BBC \neq *front-facing*
→ **not** Contour-Consistent

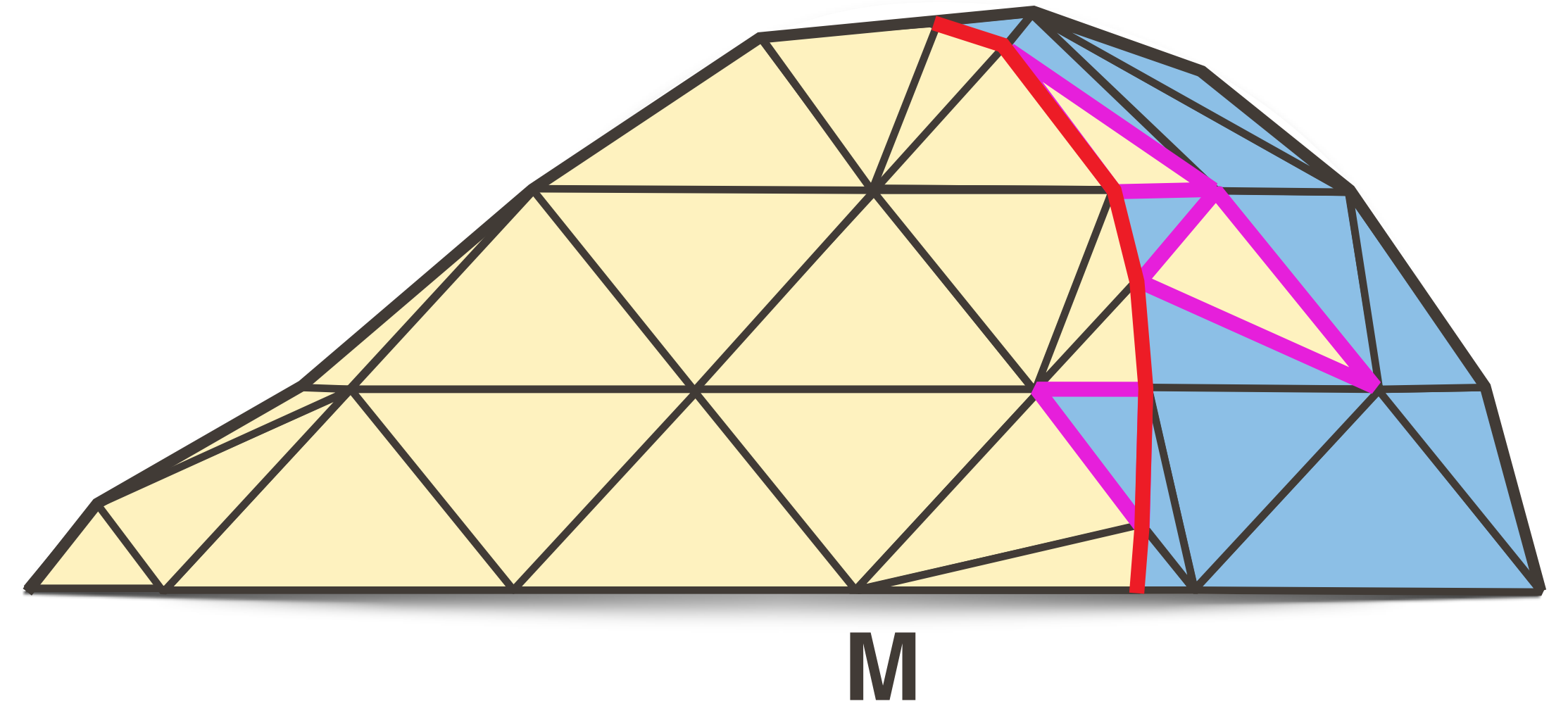
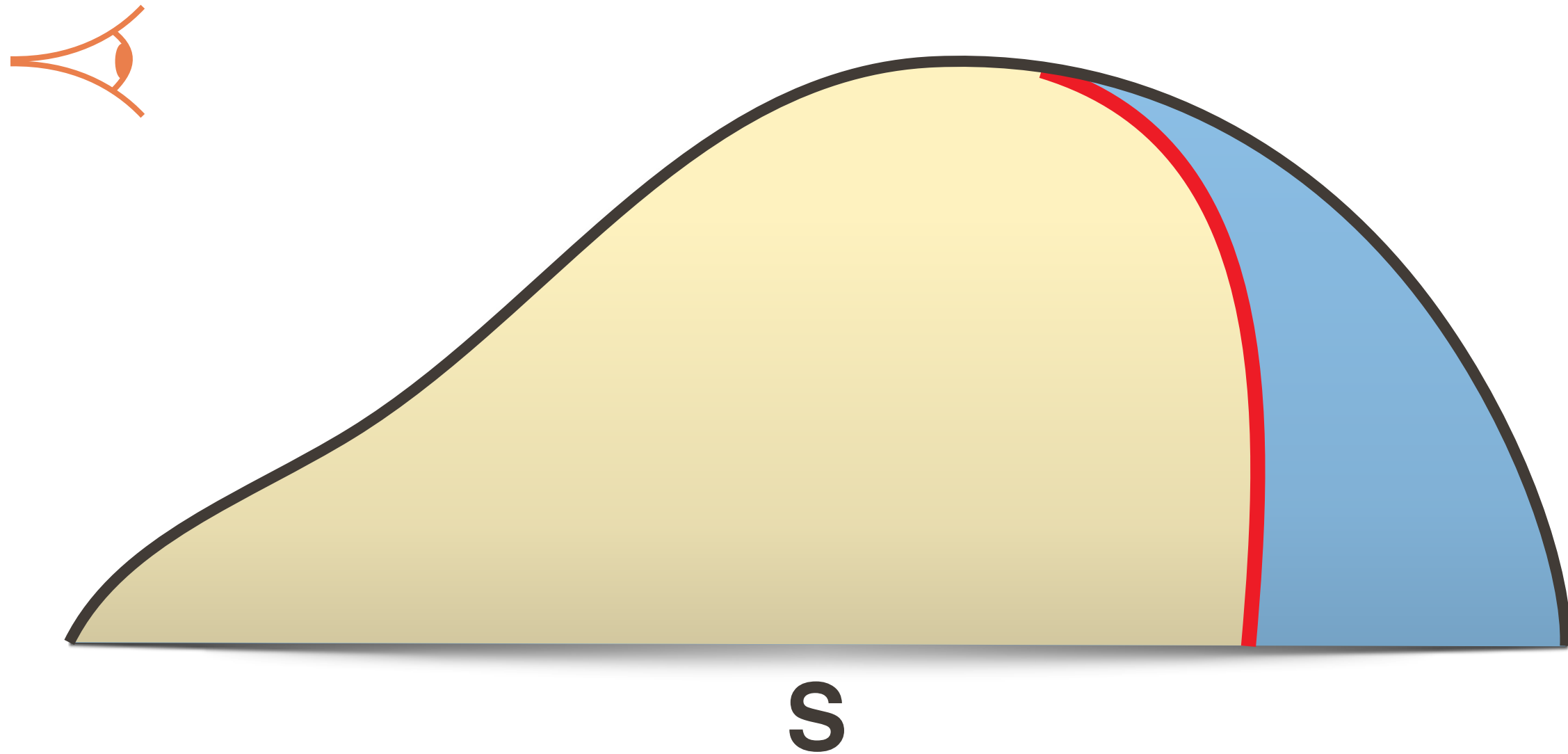
Contour insertion

Insert **contour edges** into the mesh.



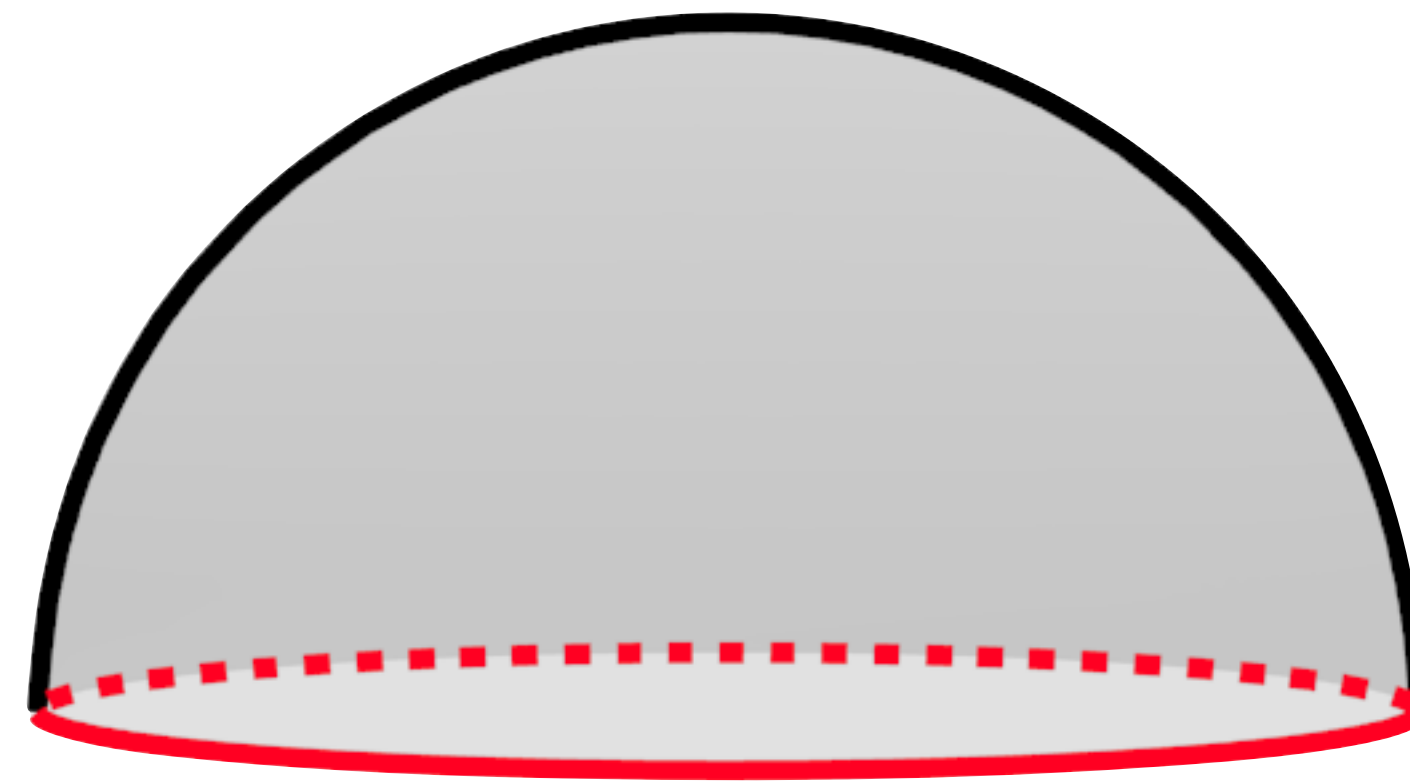
Contour insertion

Insert **contour edges** into the mesh.



Inconsistent Triangle

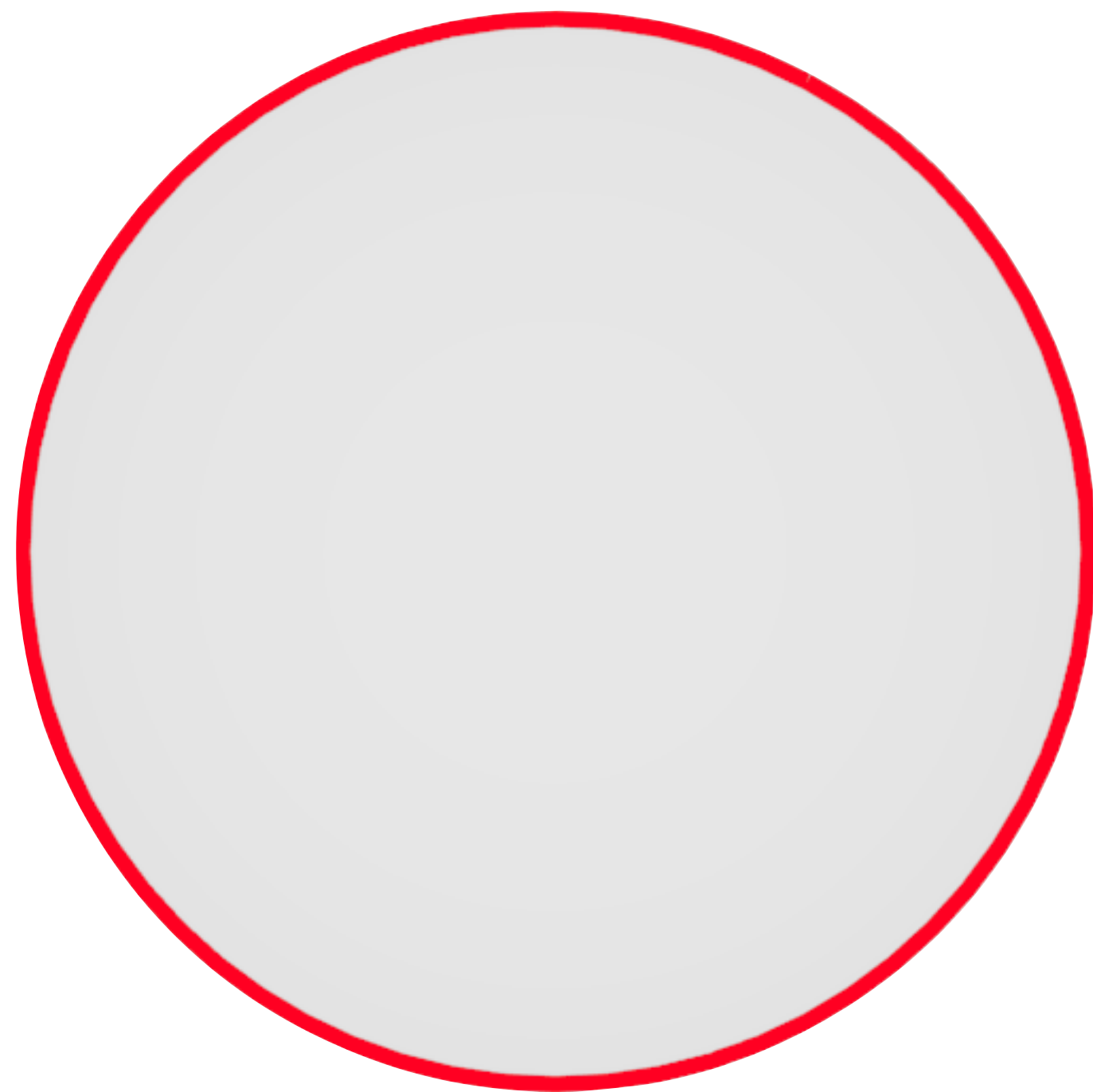
Intuition for a simple case
(hemisphere)



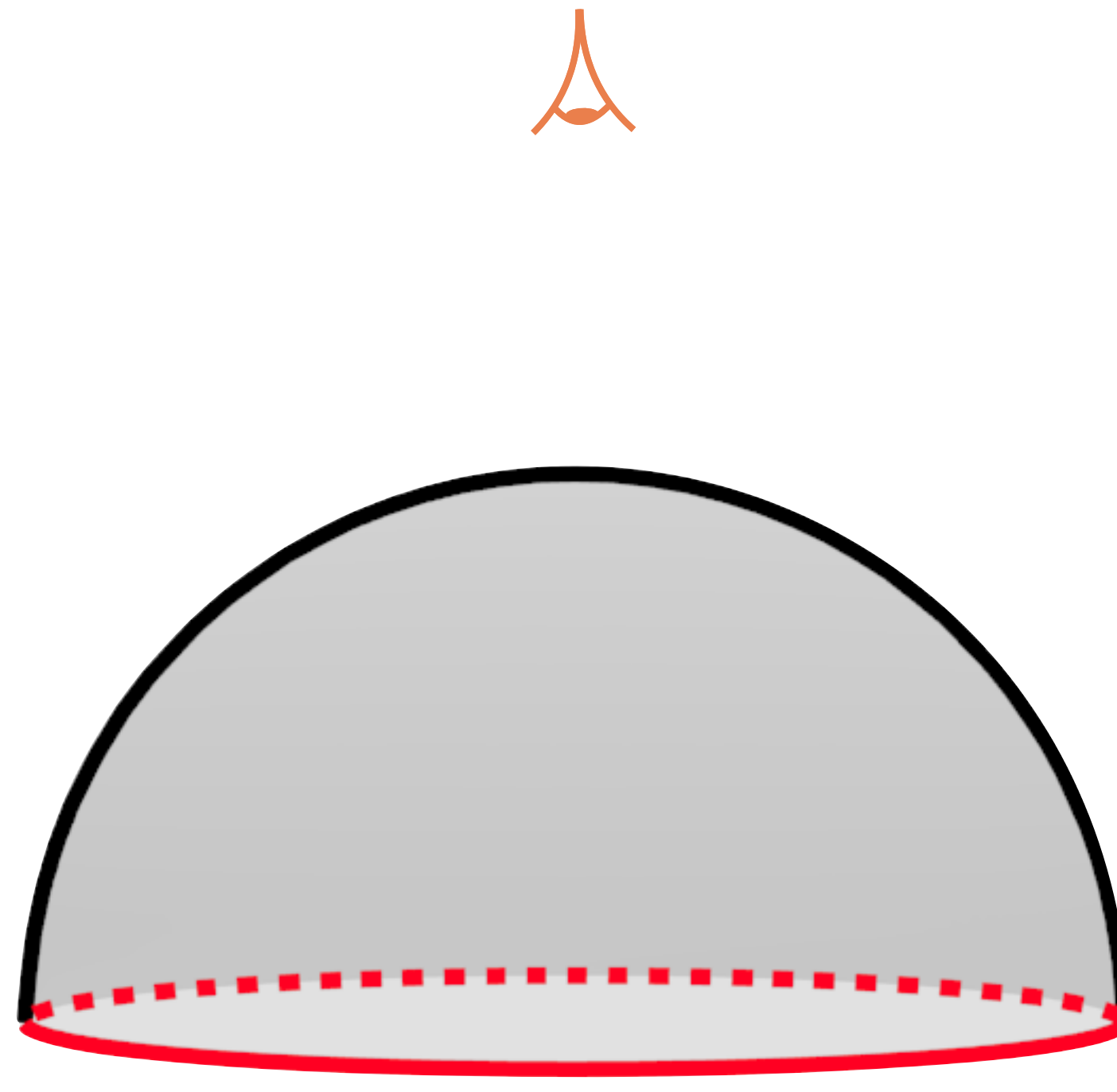
Side view

Inconsistent Triangle

Intuition for a simple case
(hemisphere)



Camera view

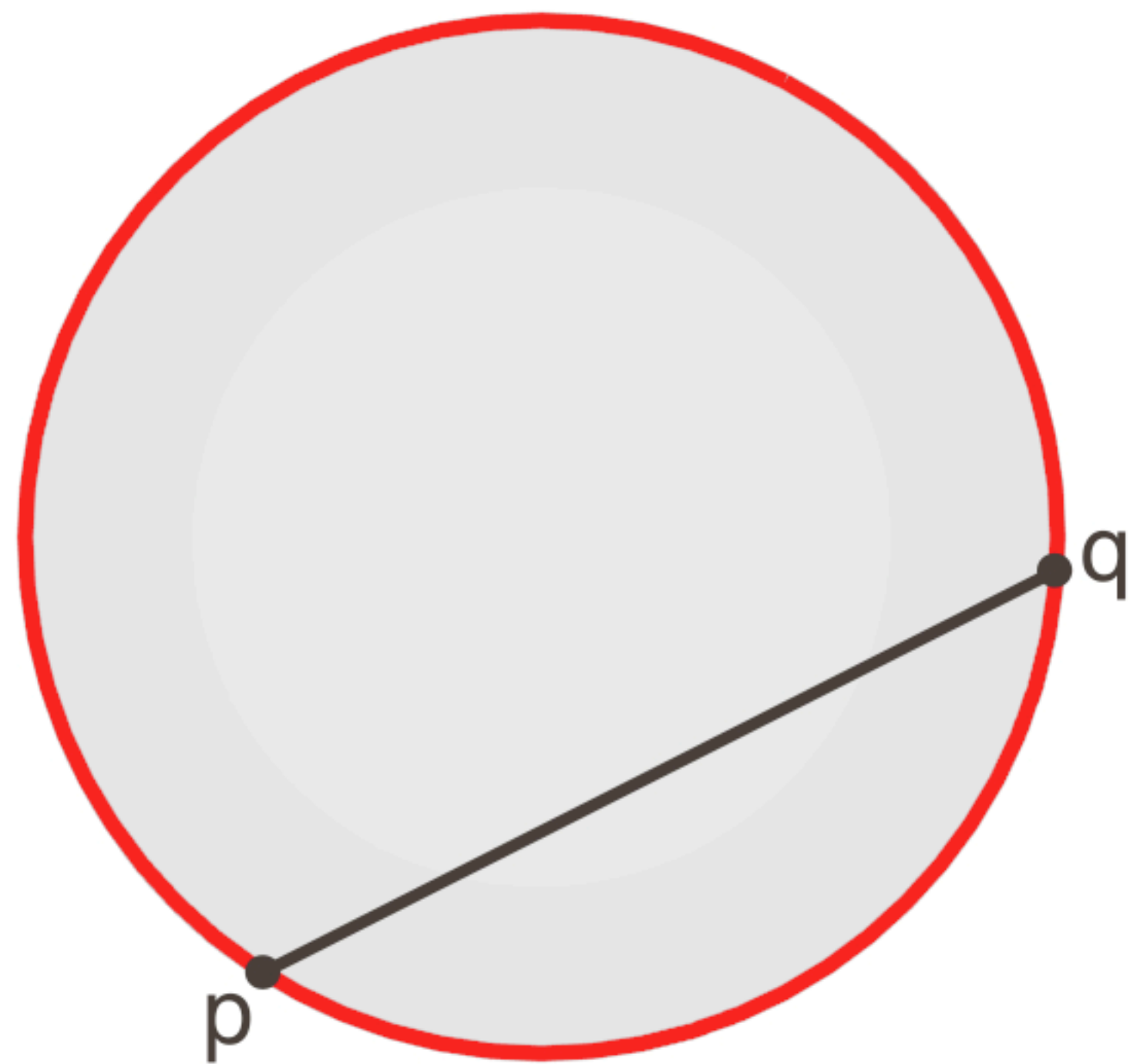


Side view

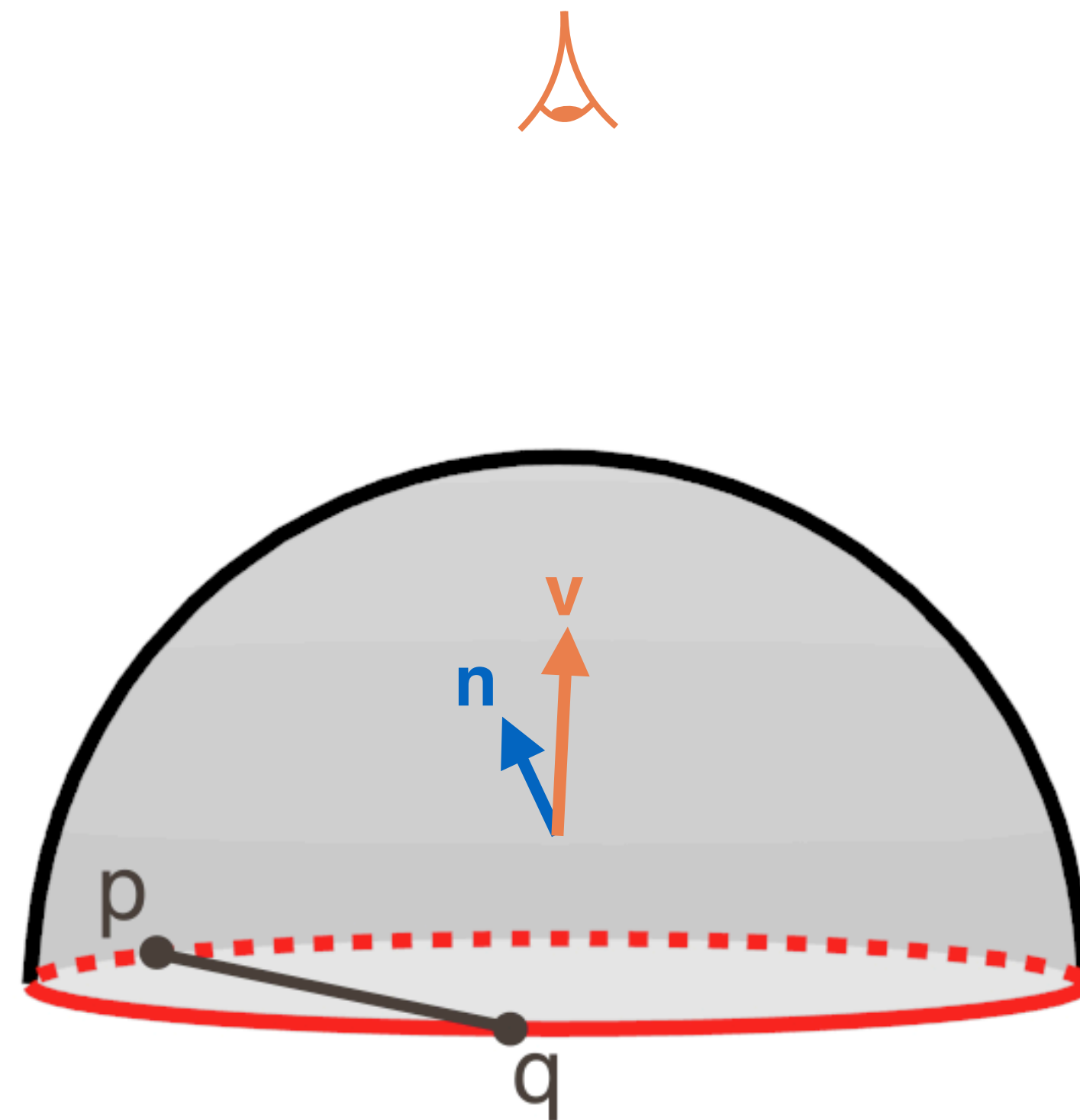
Inconsistent Triangle

Intuition for a simple case
(hemisphere)

Consistent Triangle



Camera view

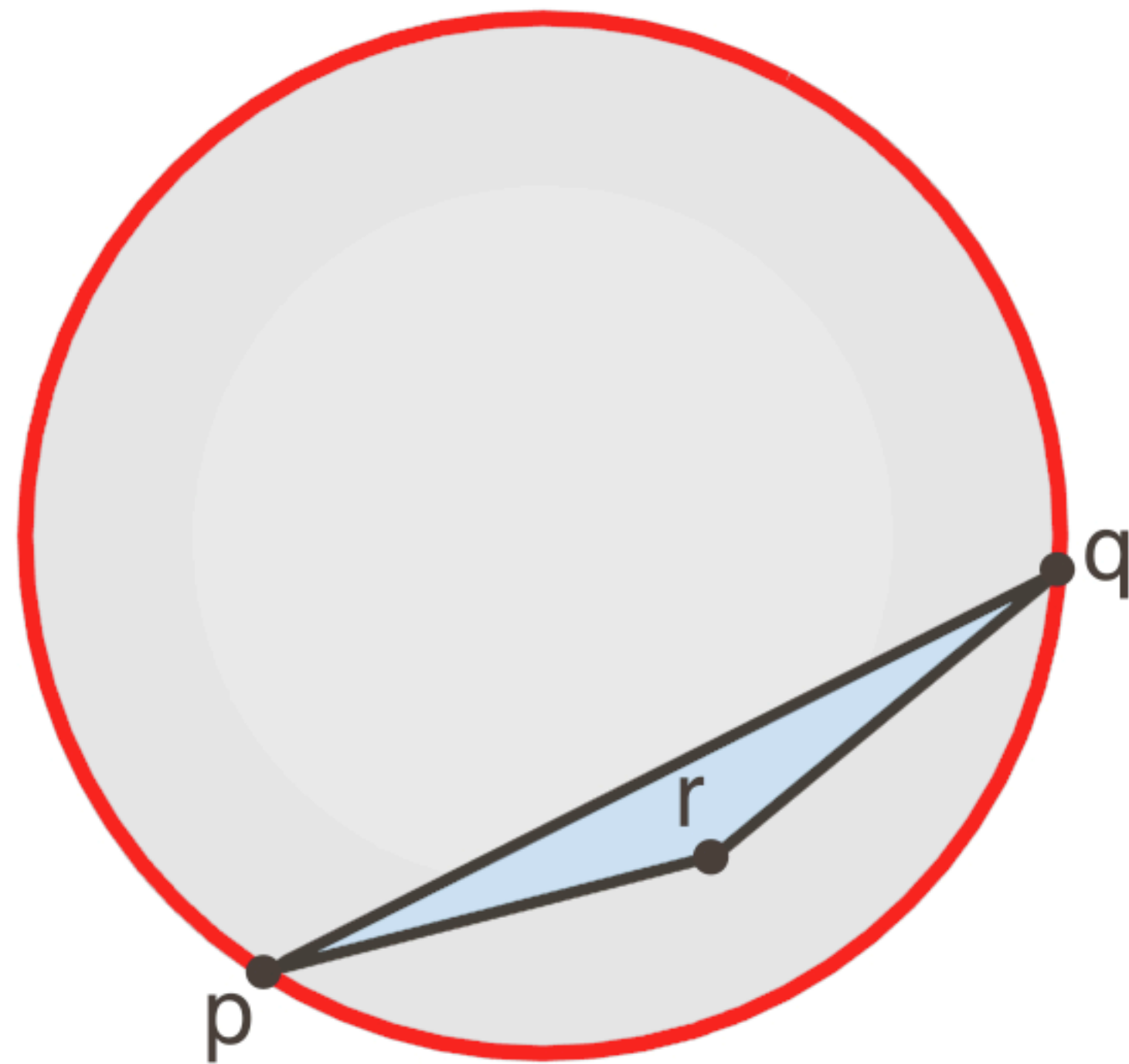


Side view

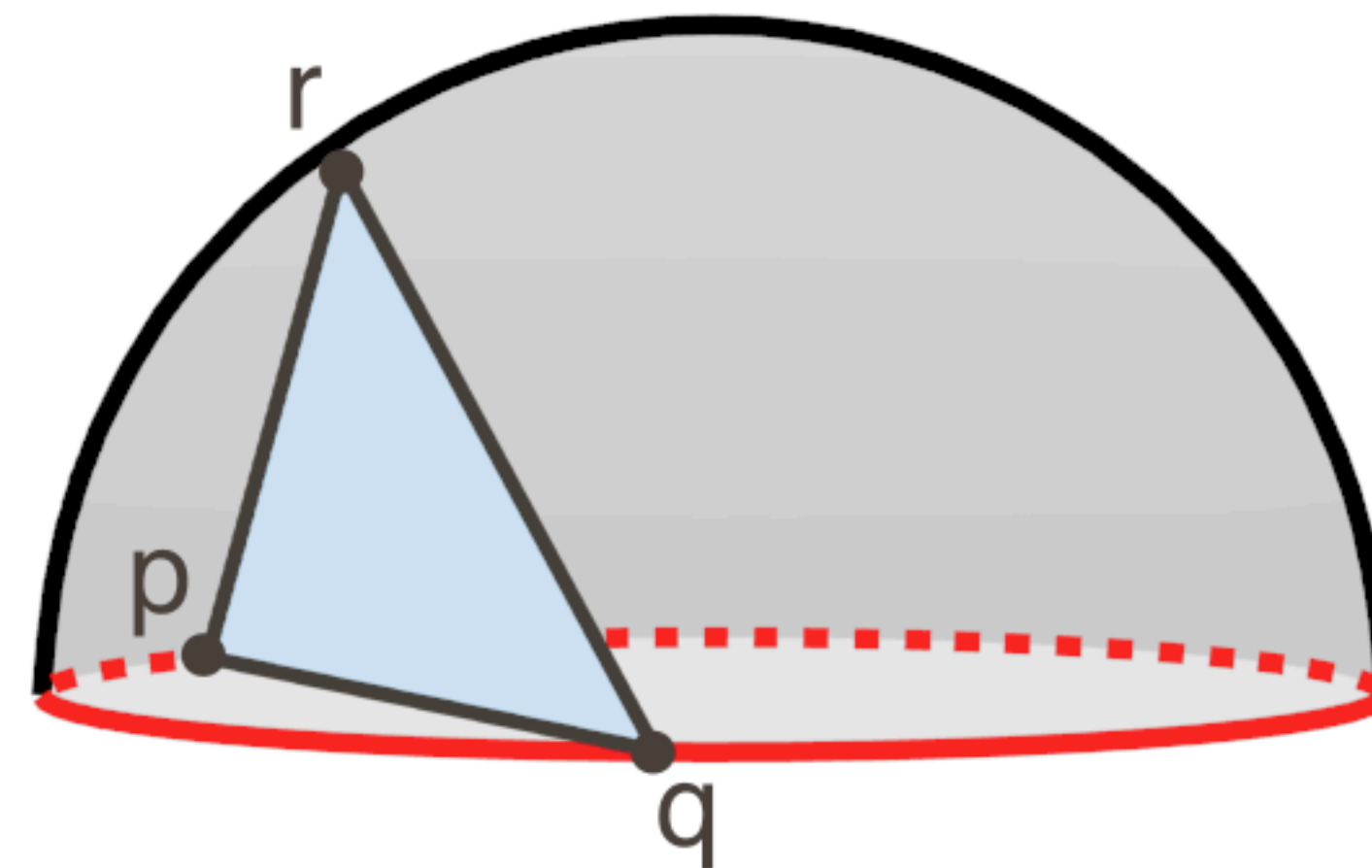
Inconsistent Triangle

Intuition for a simple case
(hemisphere)

Inconsistent Triangle



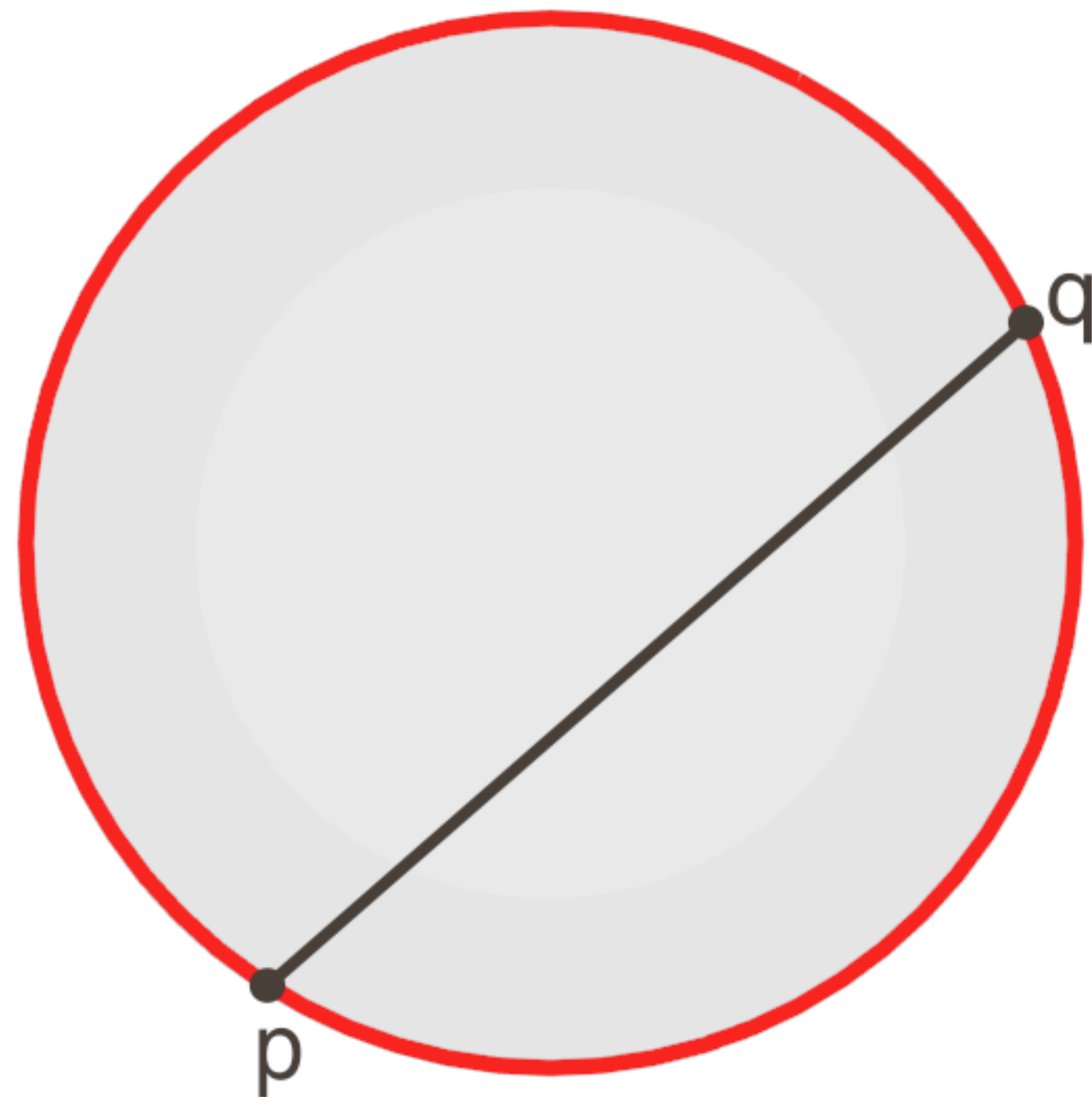
Camera view



Side view

Radial Triangle

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

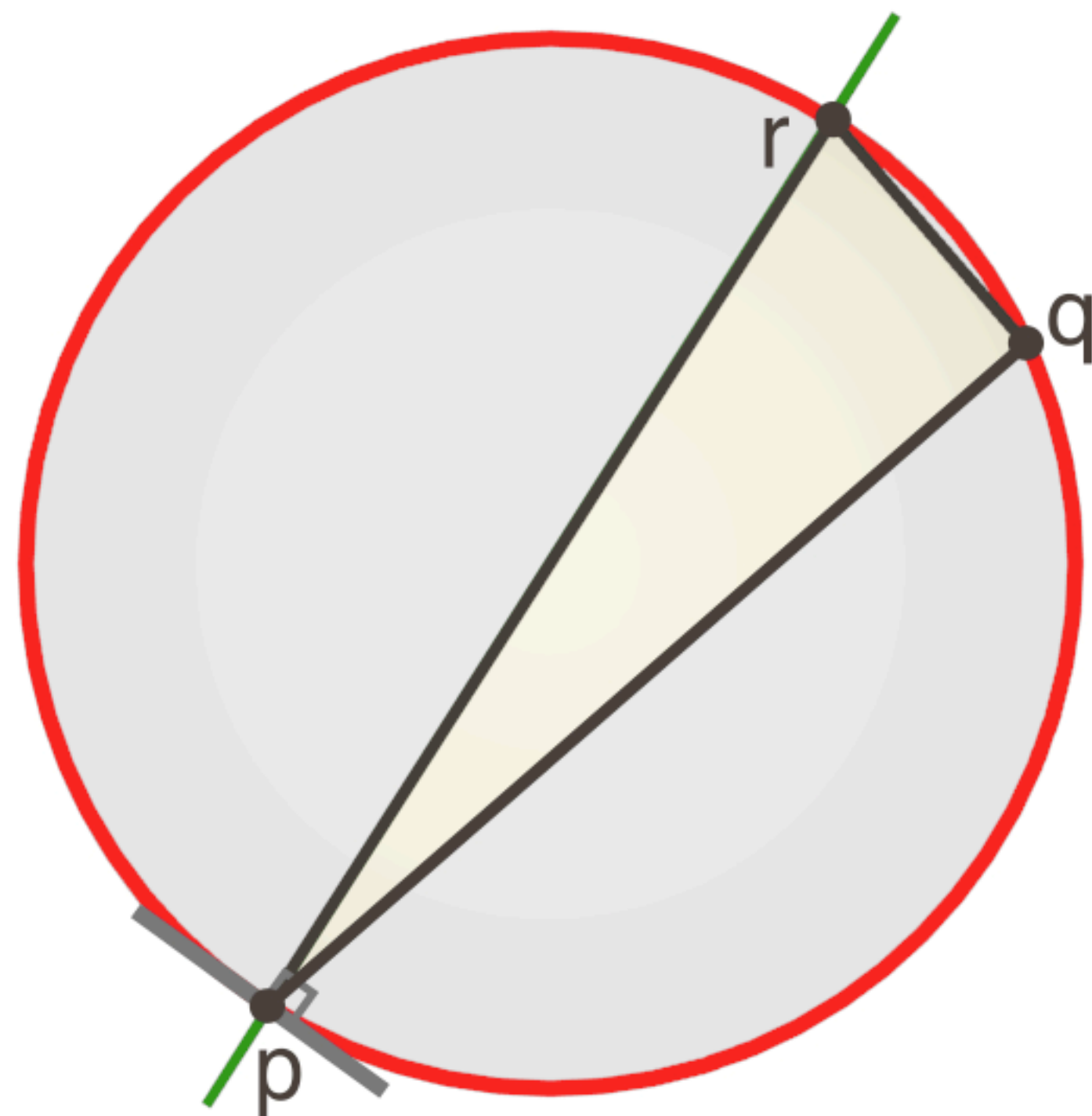


Camera view

Radial Triangle

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

⇒ Triangle **pqr** guaranteed to be **Contour-Consistent**

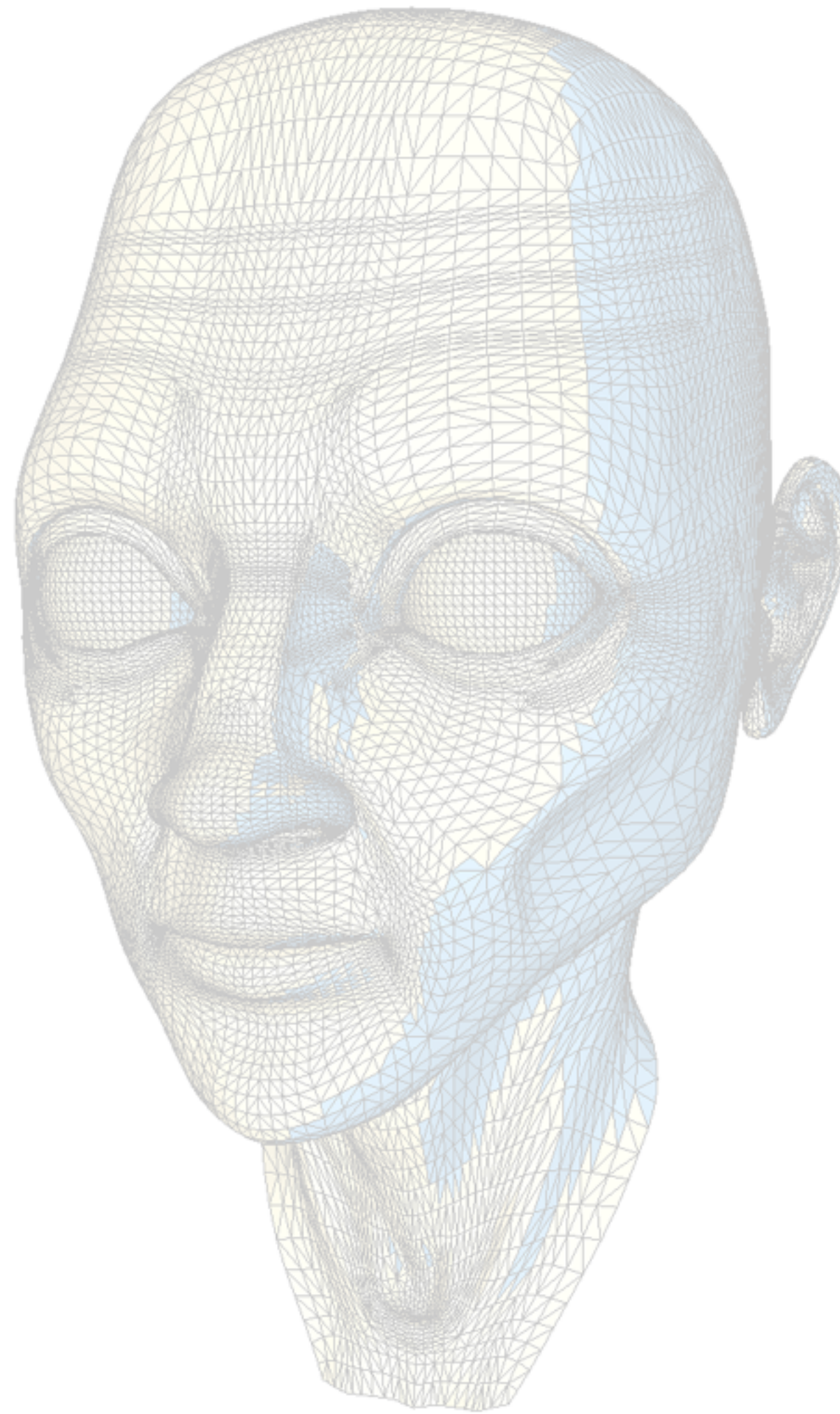


Camera view

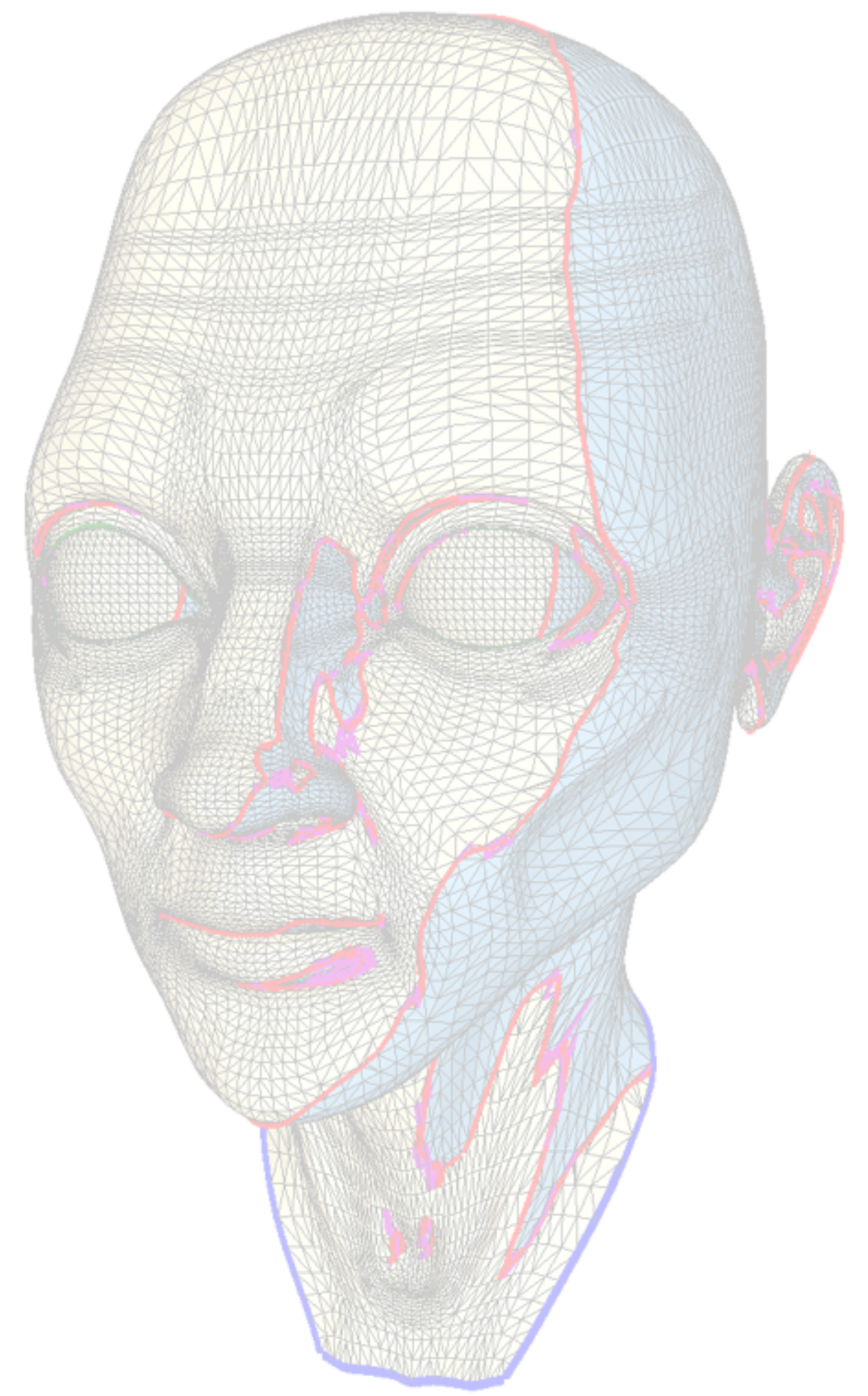
Adaptive Tessellation Algorithm



1. Smooth surface
[Catmull and Clark 1978]



2. Initial mesh

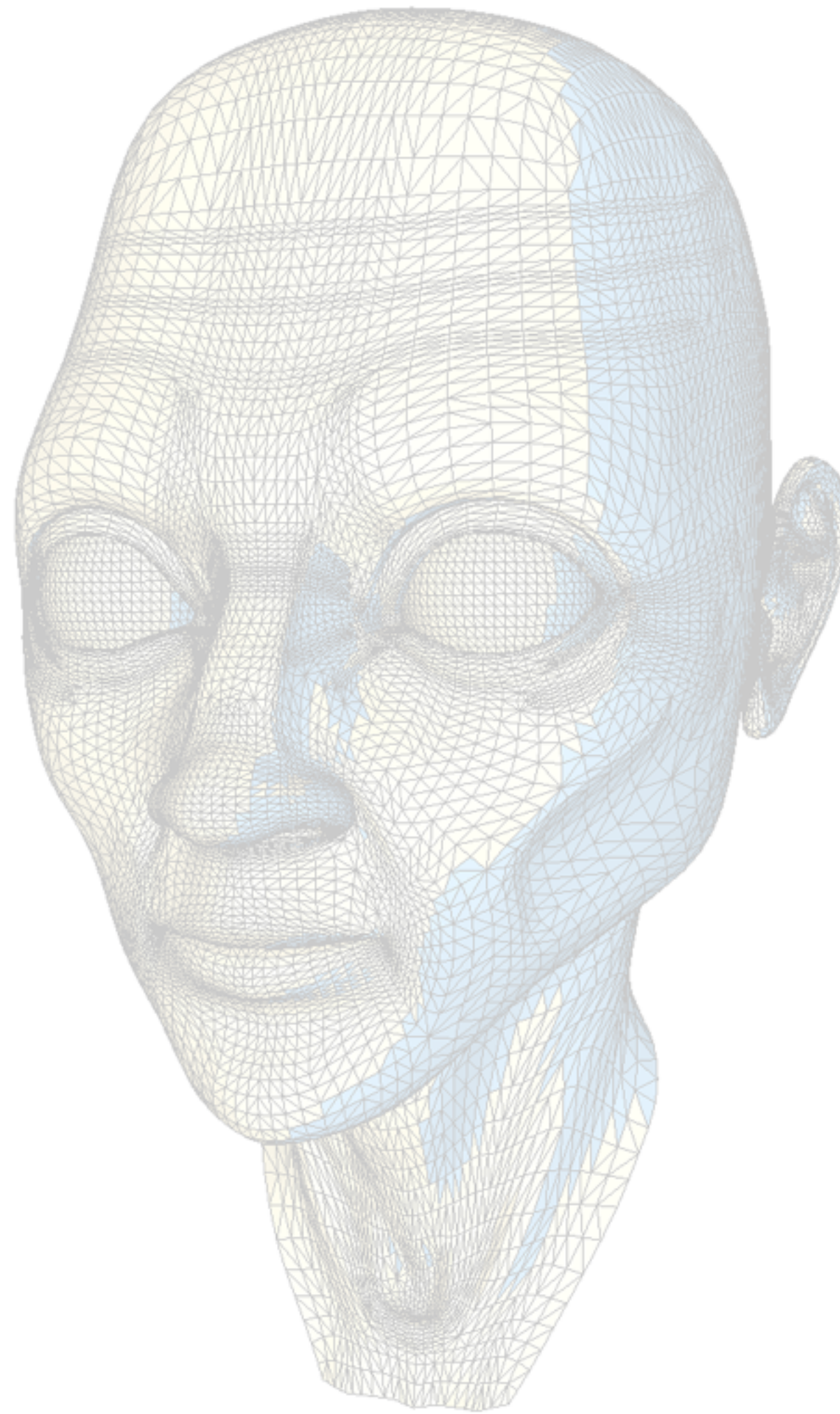


3. Contour insertion

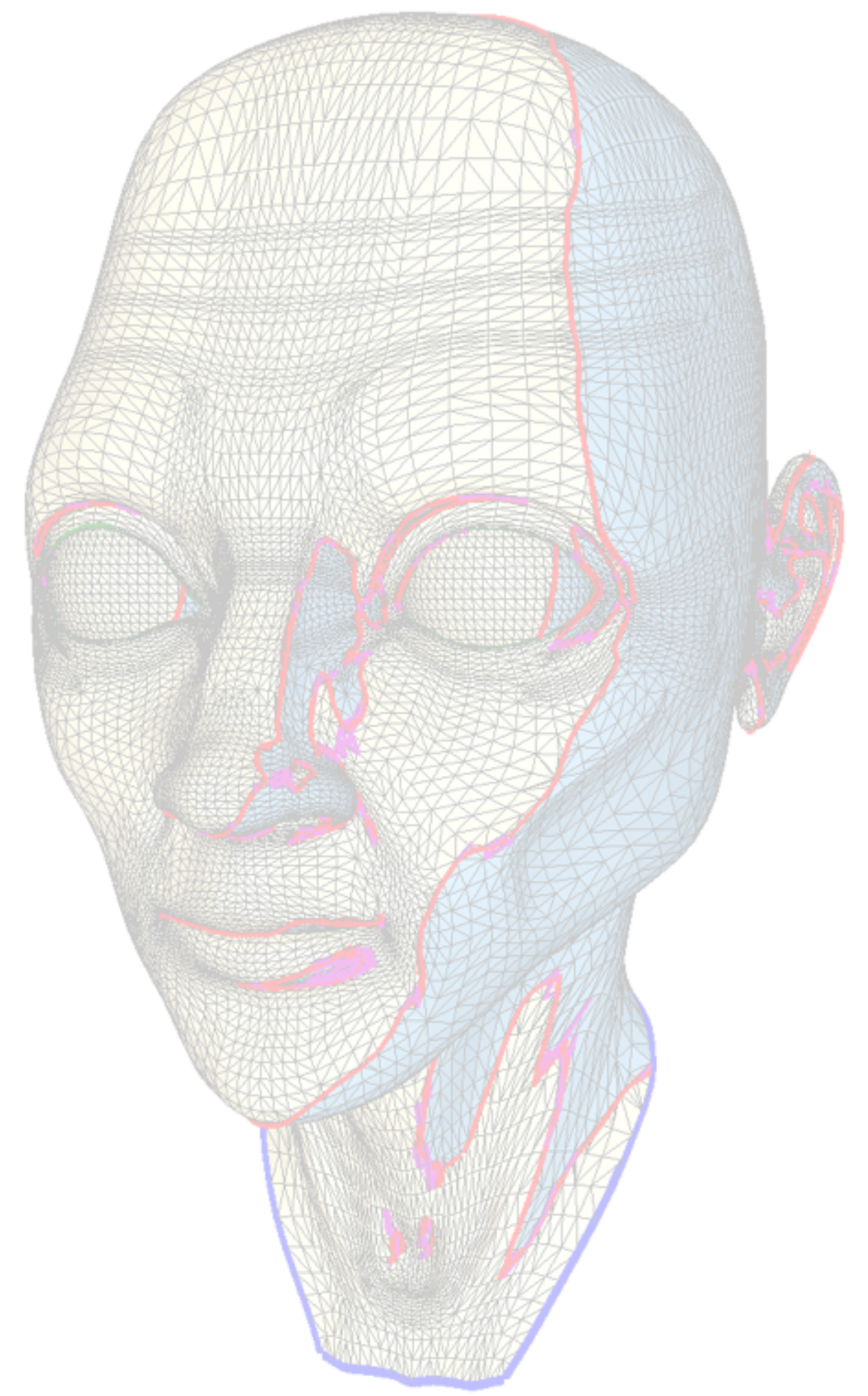
Adaptive Tessellation Algorithm



1. Smooth surface
[Catmull and Clark 1978]



2. Initial mesh

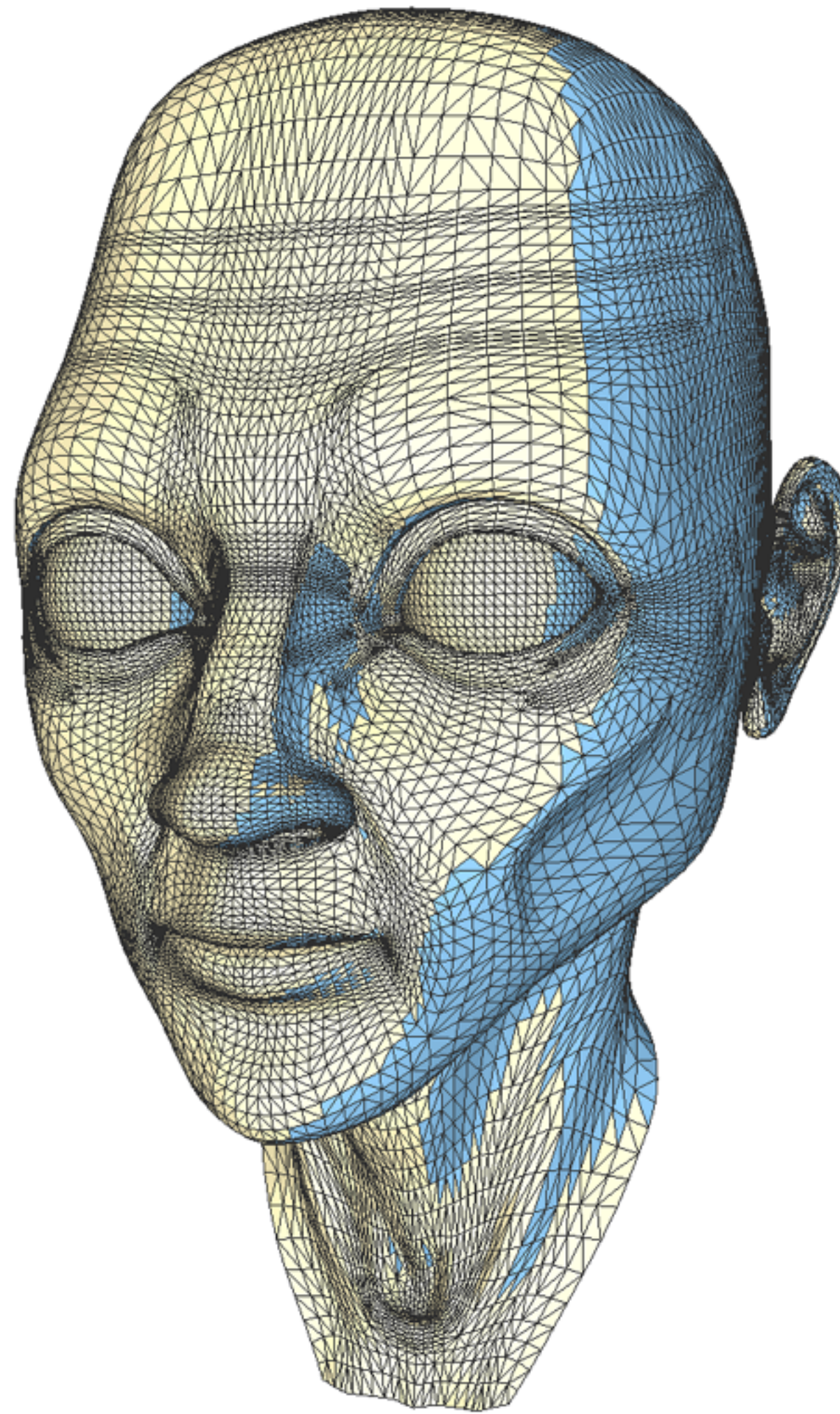


3. Contour insertion

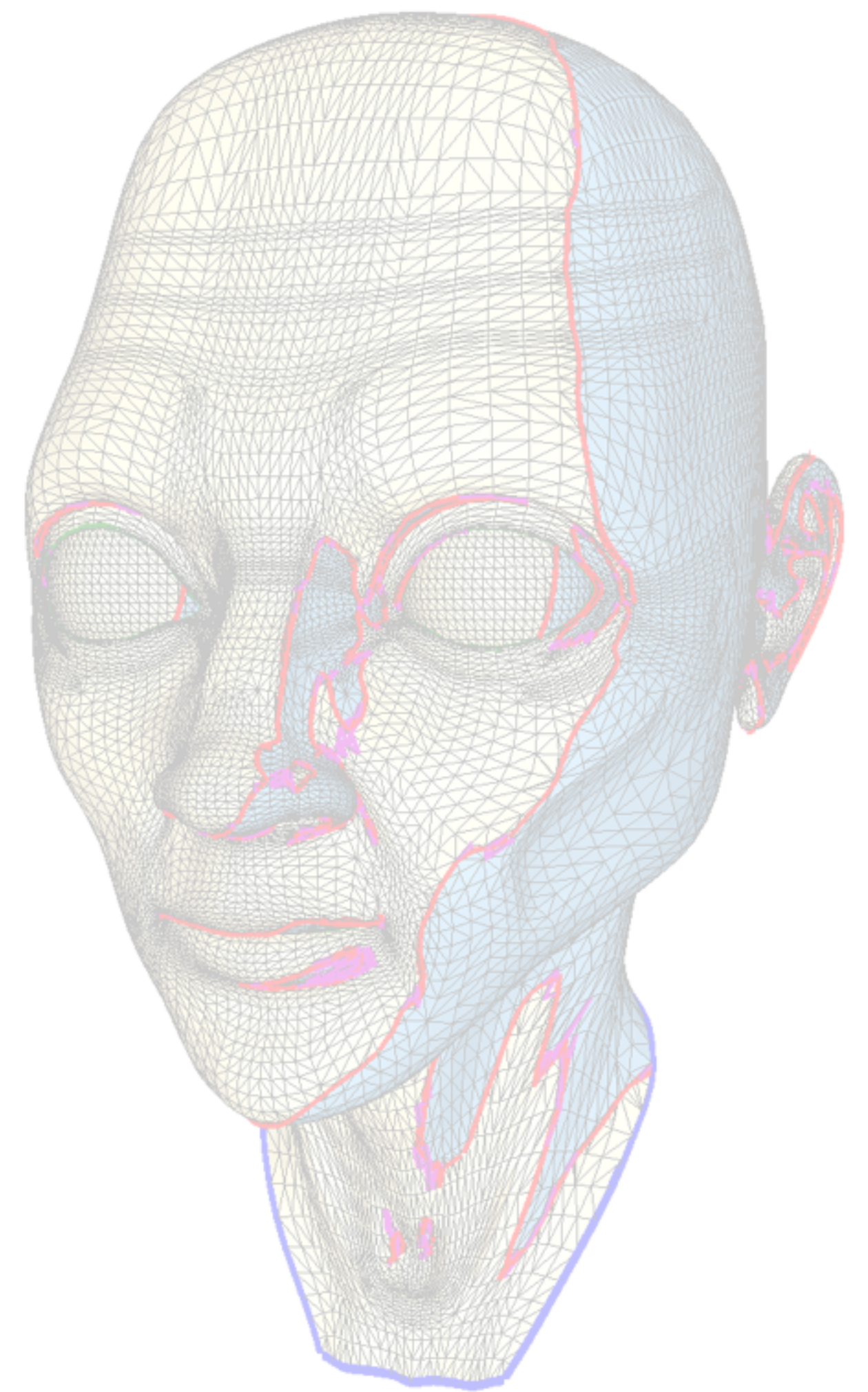
Adaptive Tessellation Algorithm



1. Smooth surface
[Catmull and Clark 1978]



2. Initial mesh

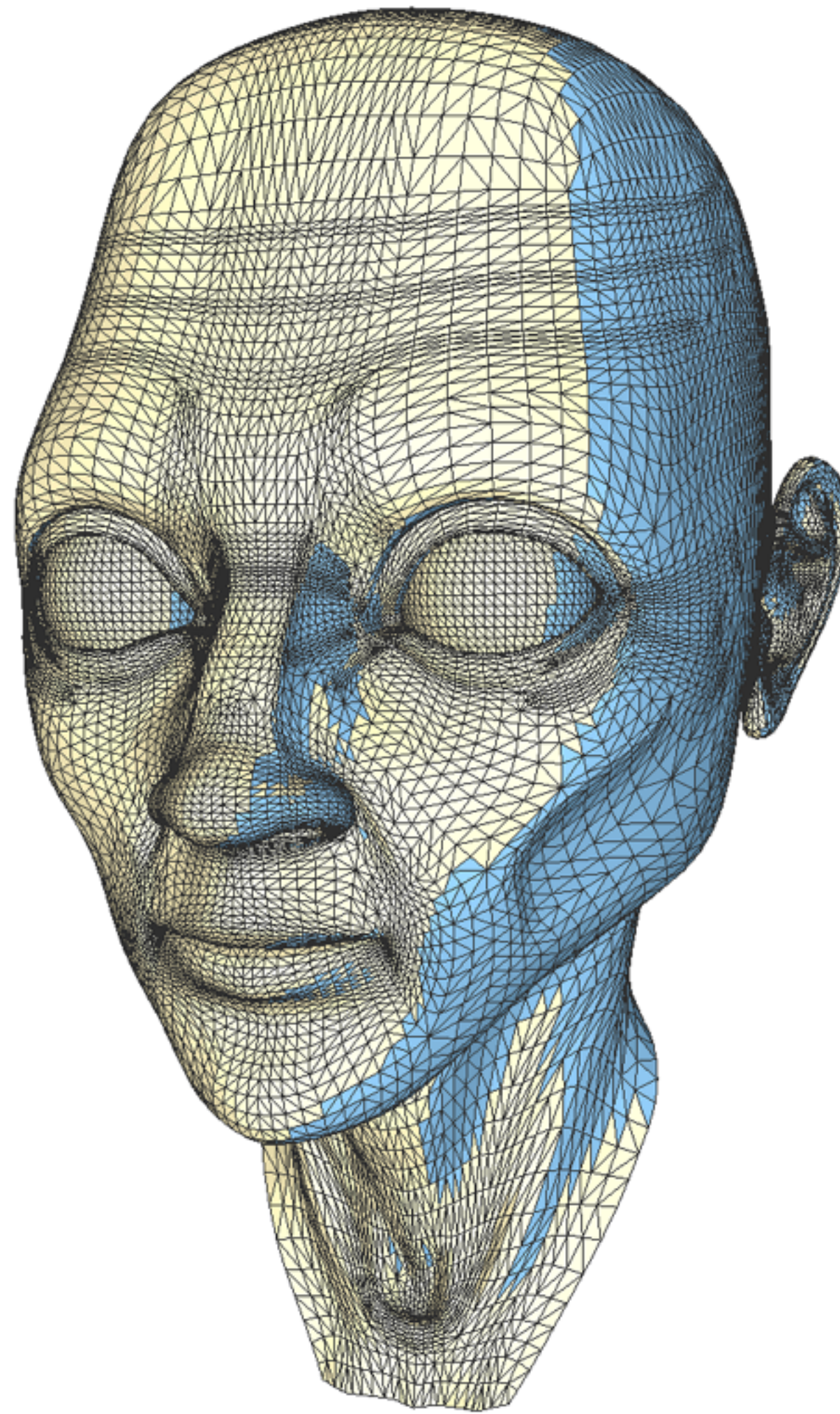


3. Contour insertion

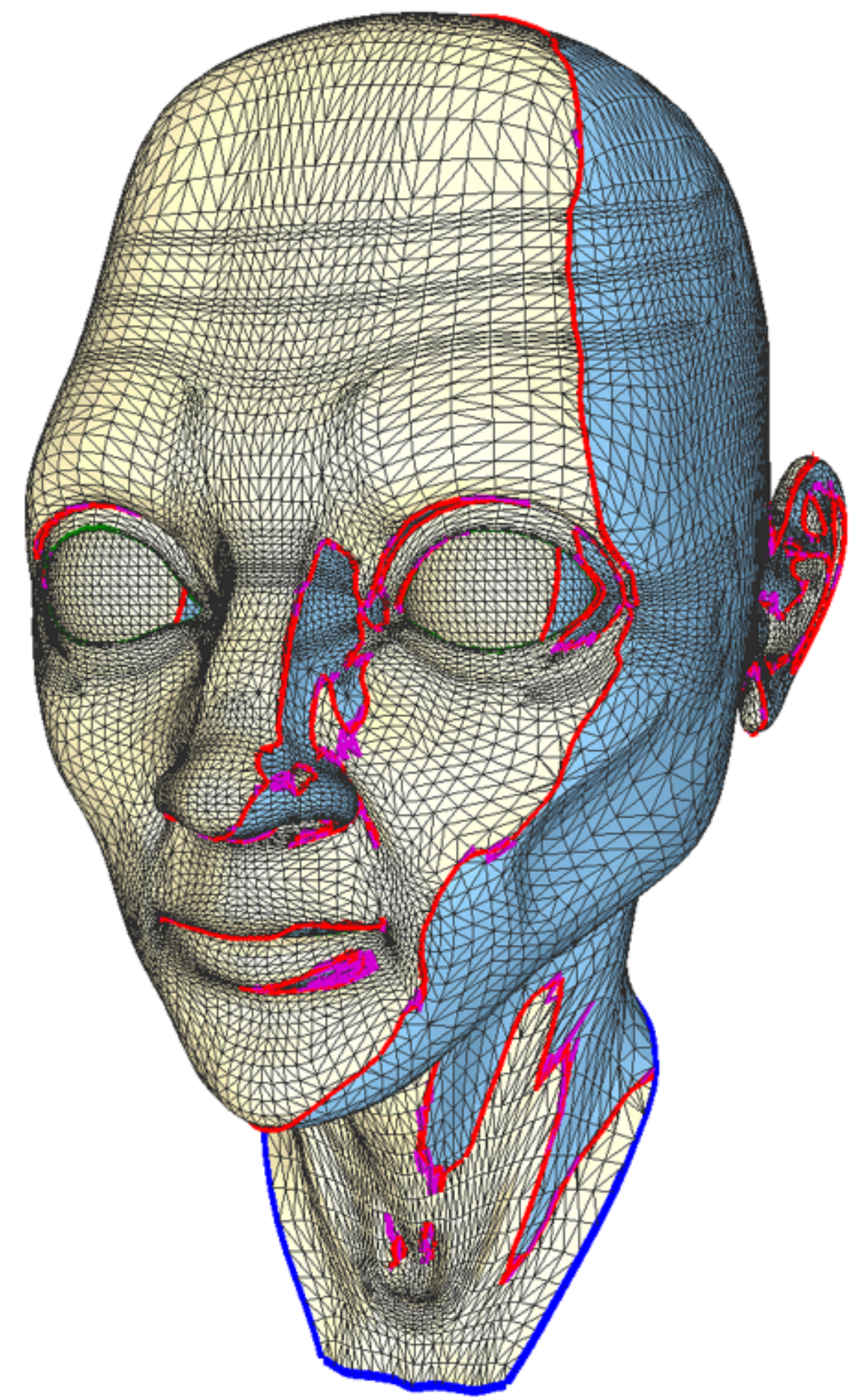
Adaptive Tessellation Algorithm



1. Smooth surface
[Catmull and Clark 1978]

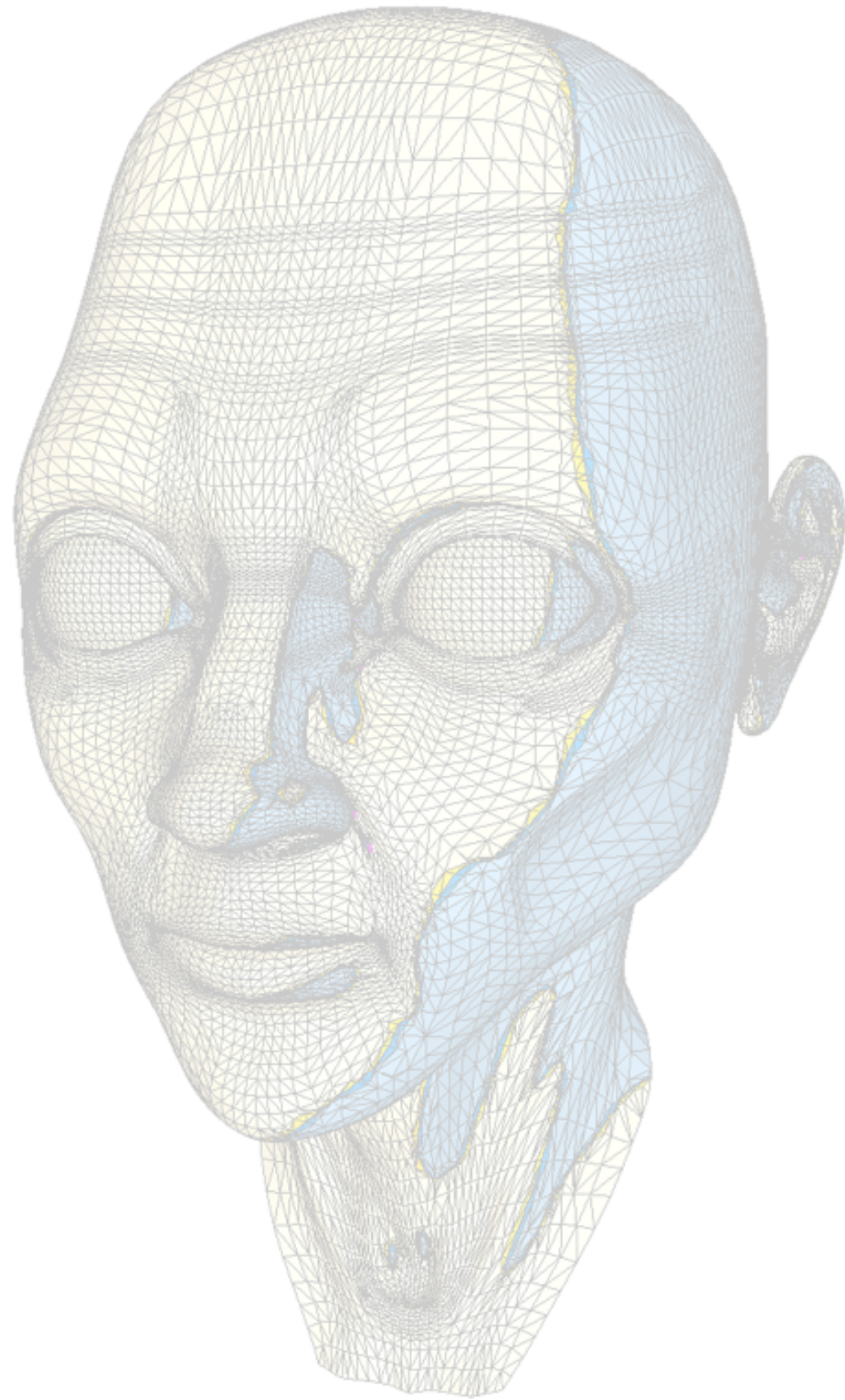


2. Initial mesh

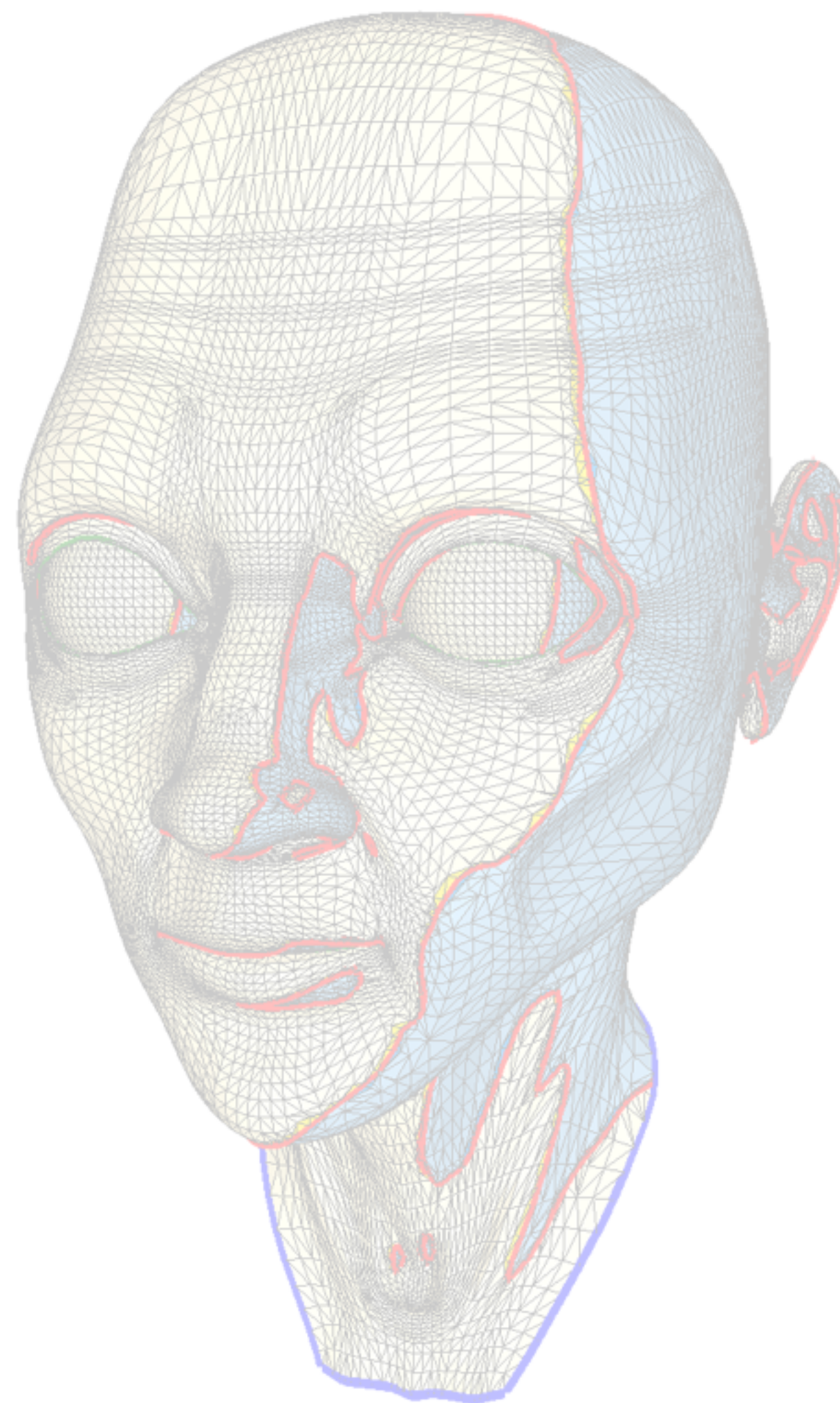


3. Contour insertion

Adaptive Tessellation Algorithm



4. “Radialization”

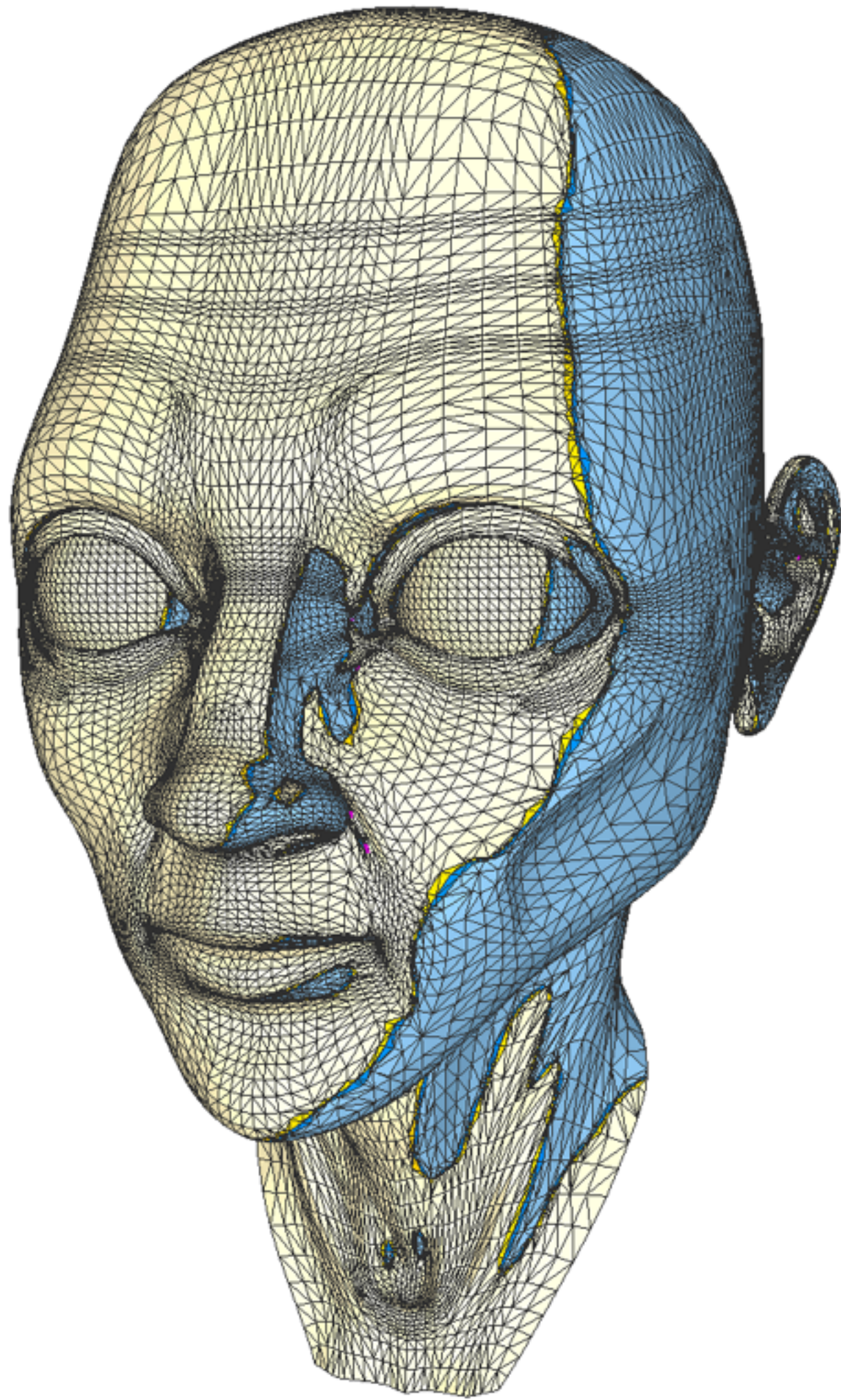


5. Optimization

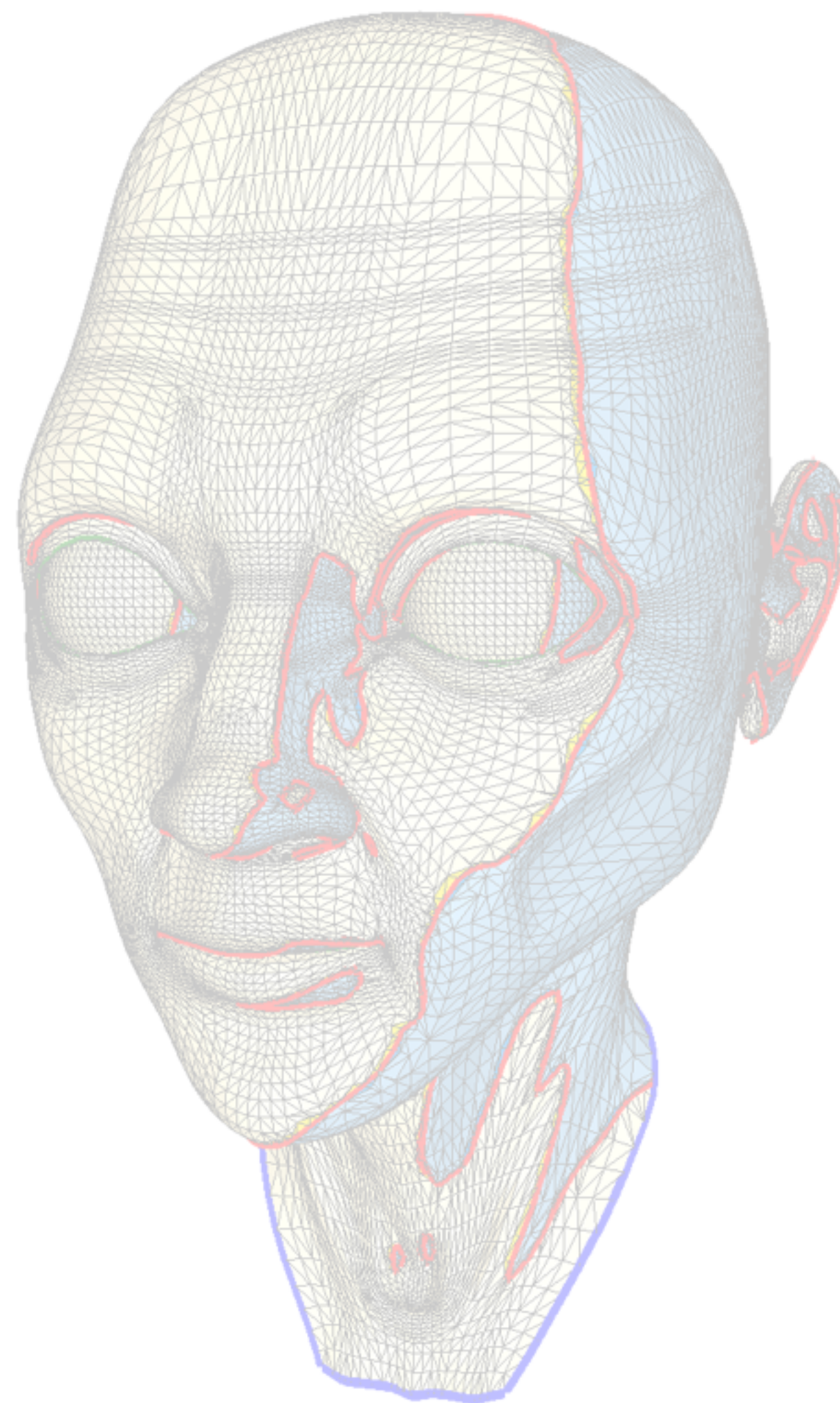


6. Contours

Adaptive Tessellation Algorithm



4. “Radialization”

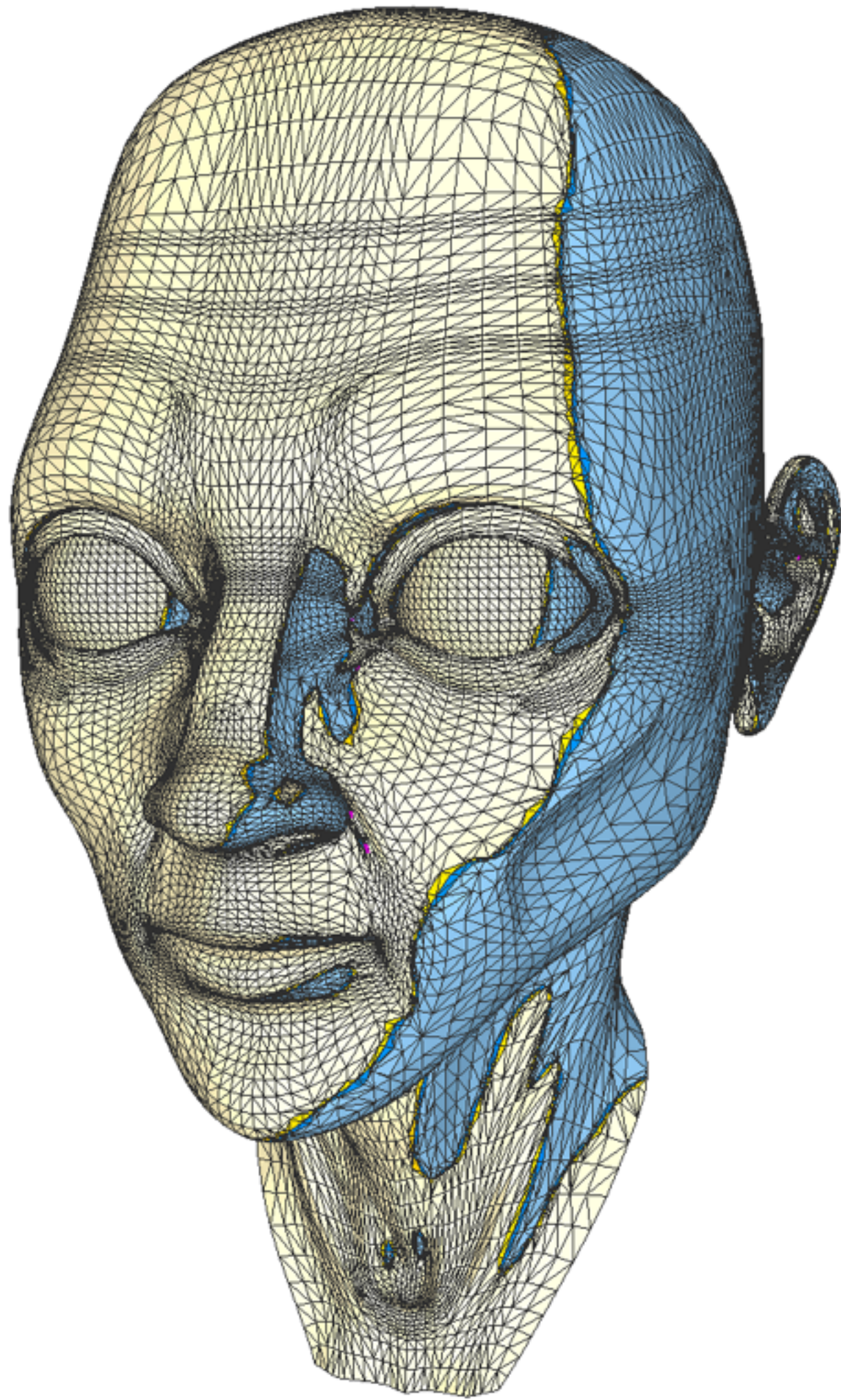


5. Optimization

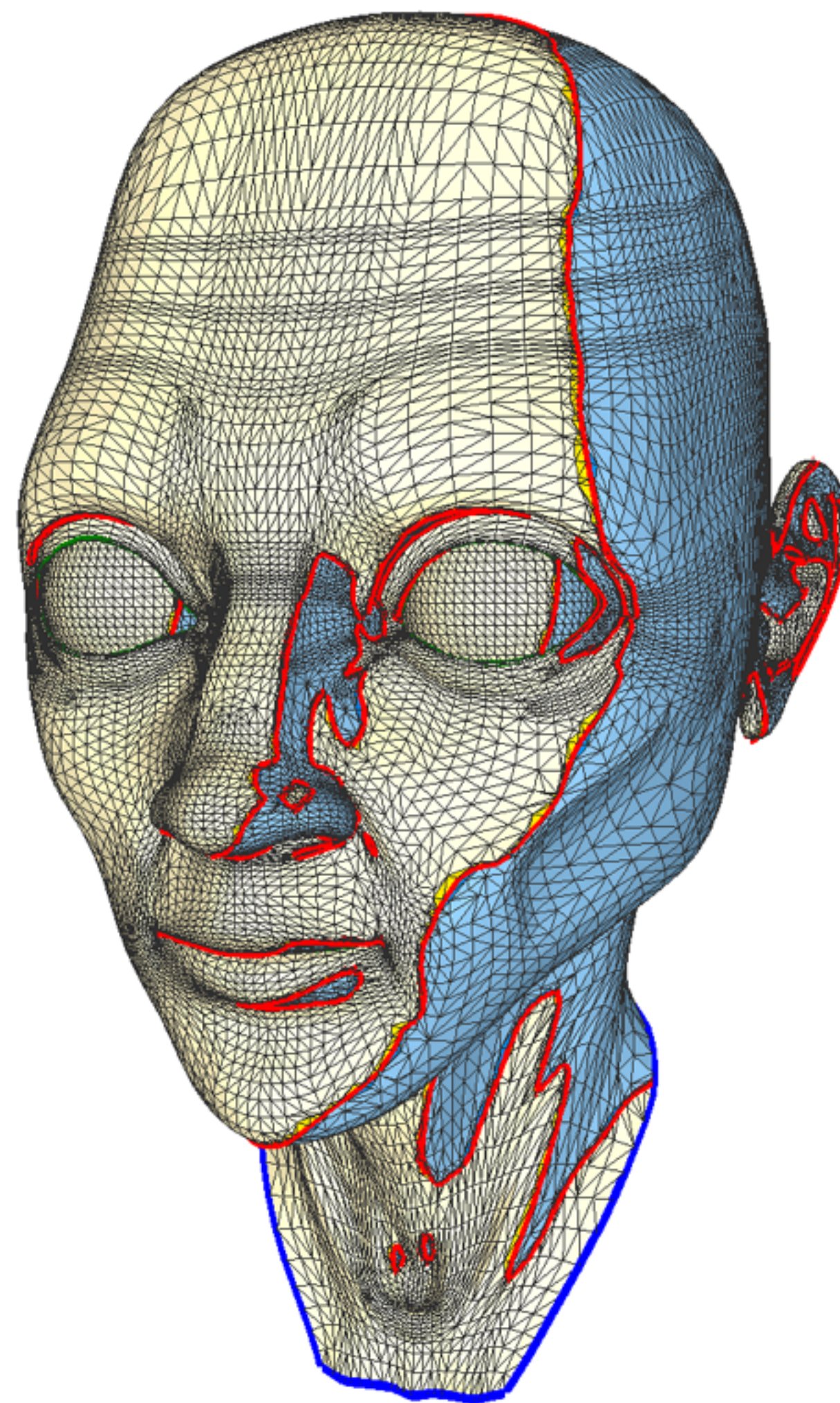


6. Contours

Adaptive Tessellation Algorithm



4. “Radialization”

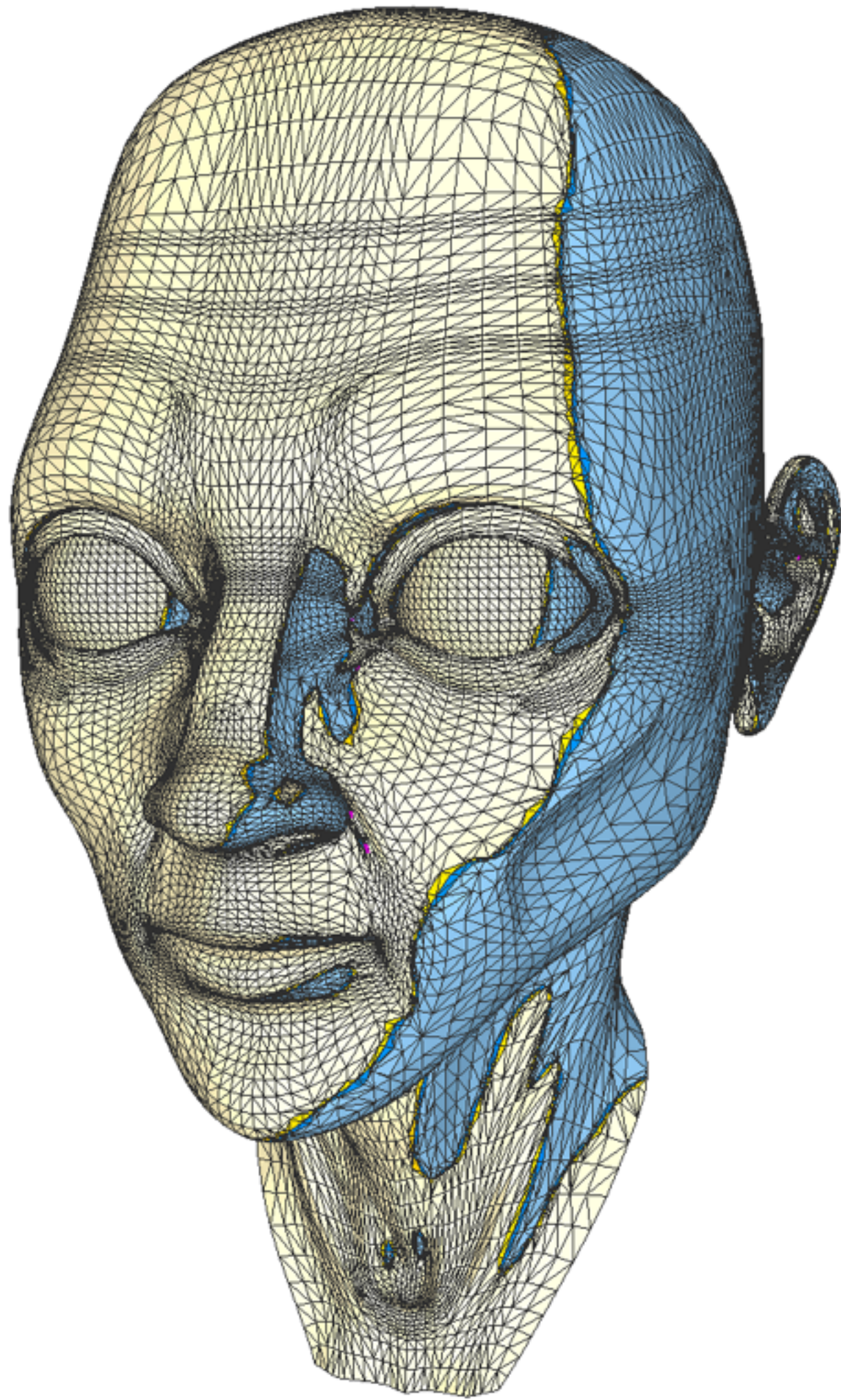


5. Optimization

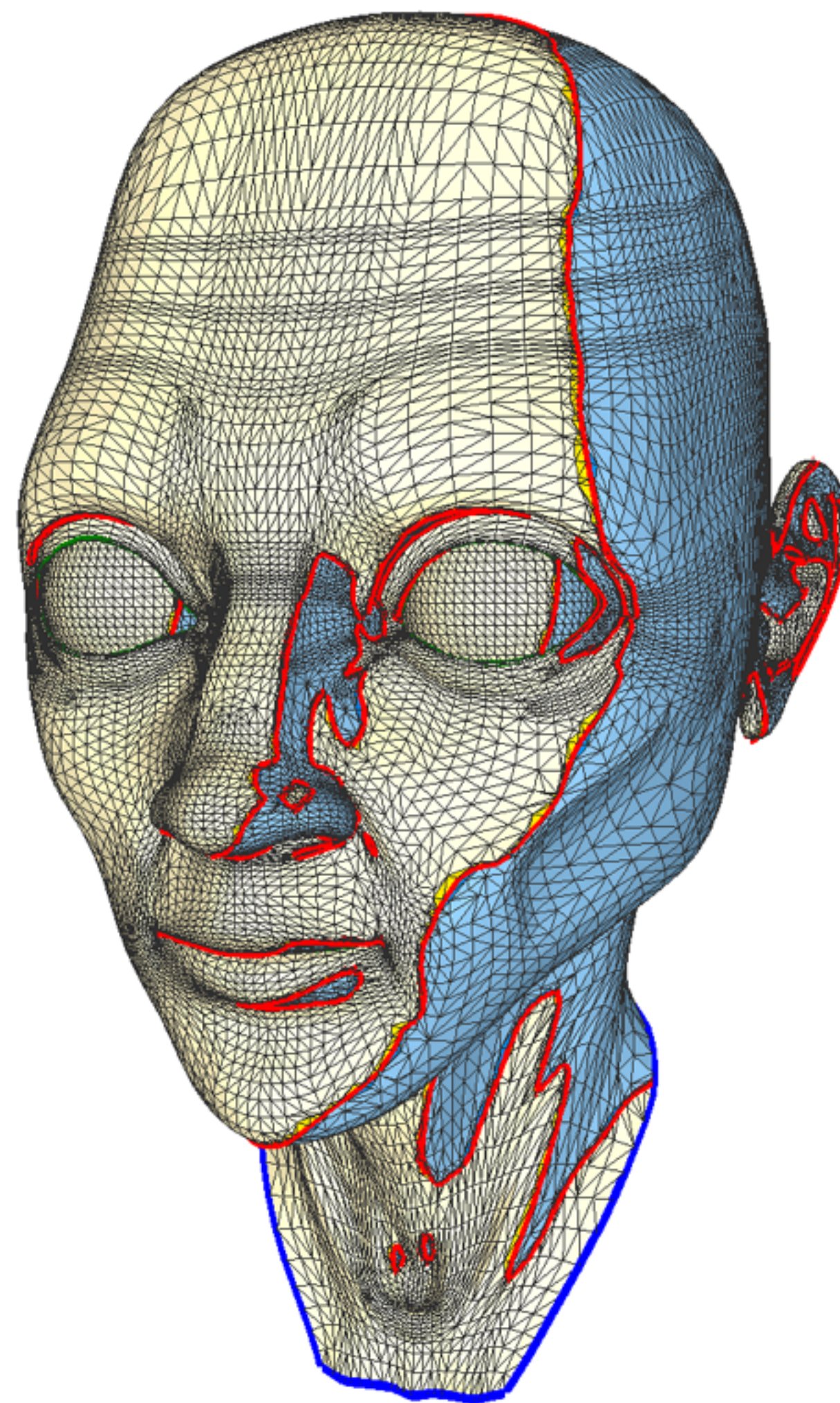


6. Contours

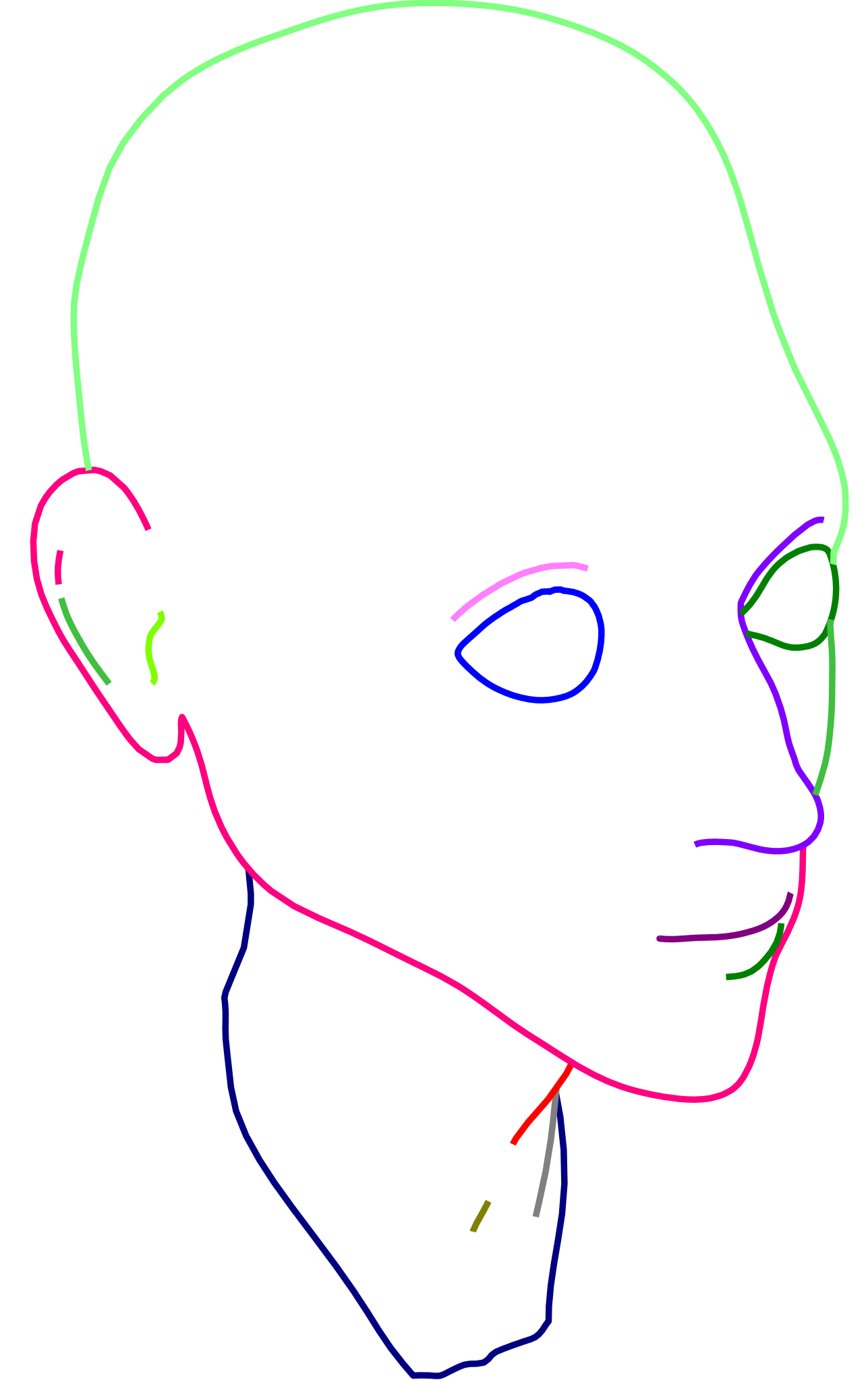
Adaptive Tessellation Algorithm



4. “Radialization”

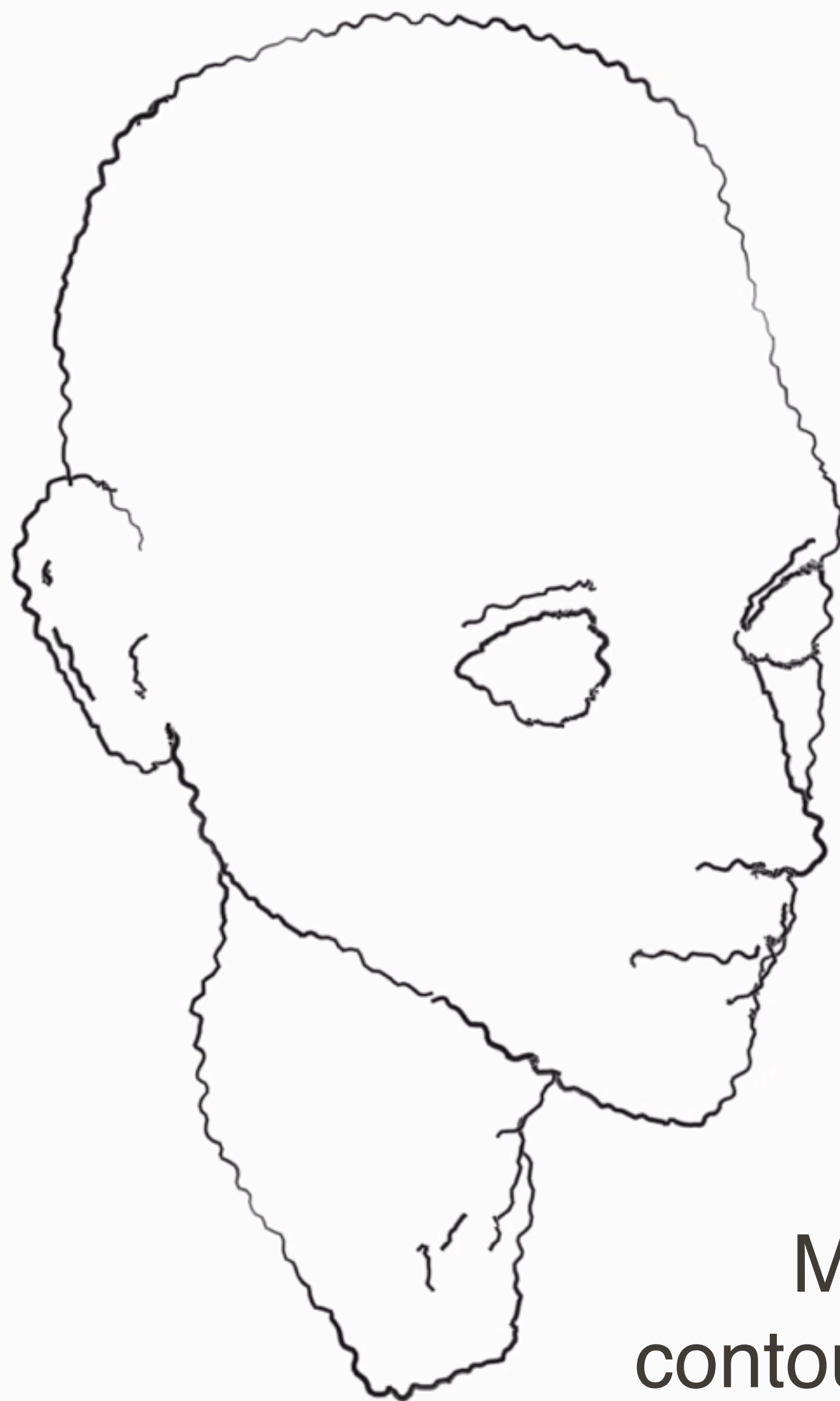


5. Optimization



6. Contours

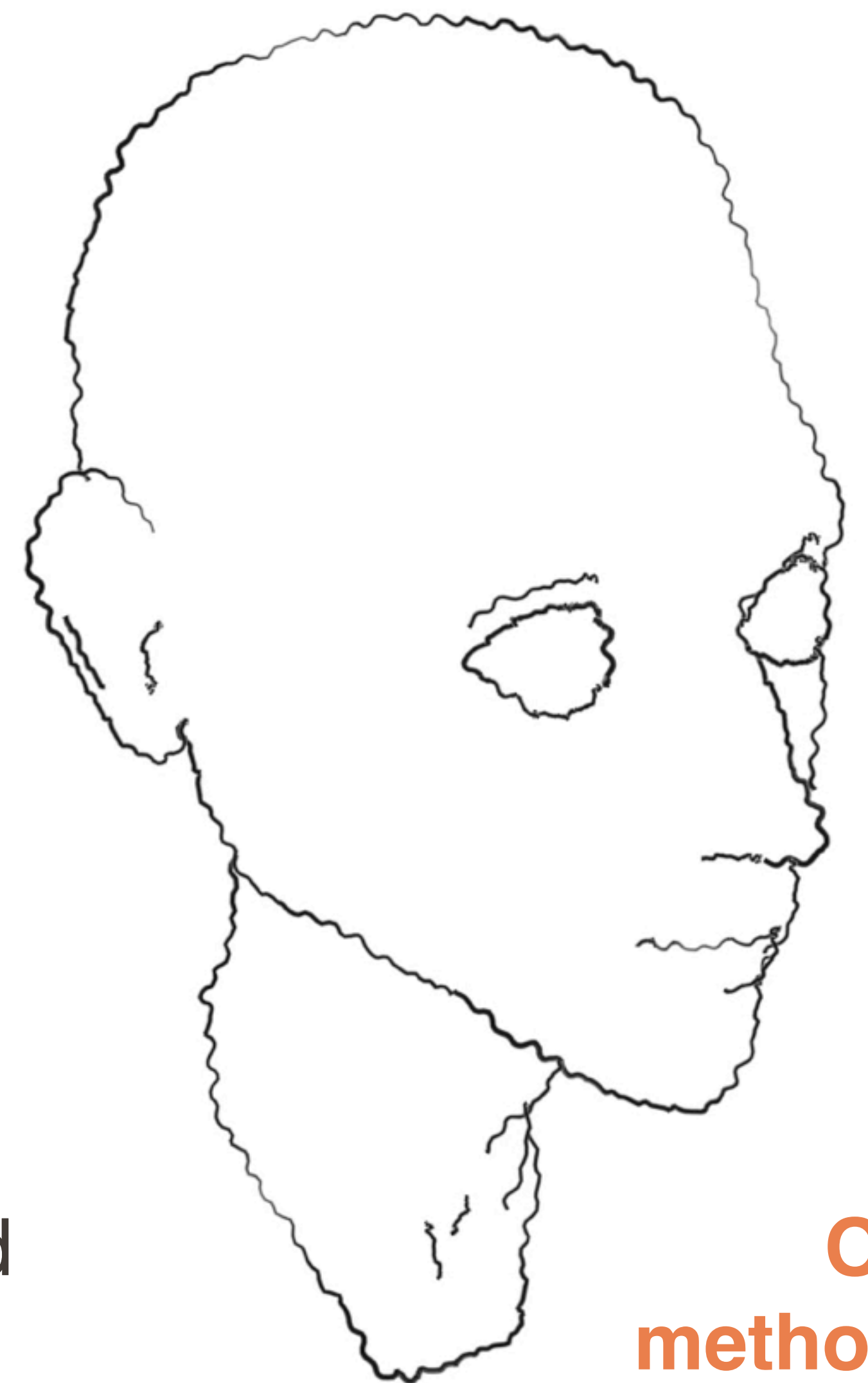
Comparison



Mesh
contours

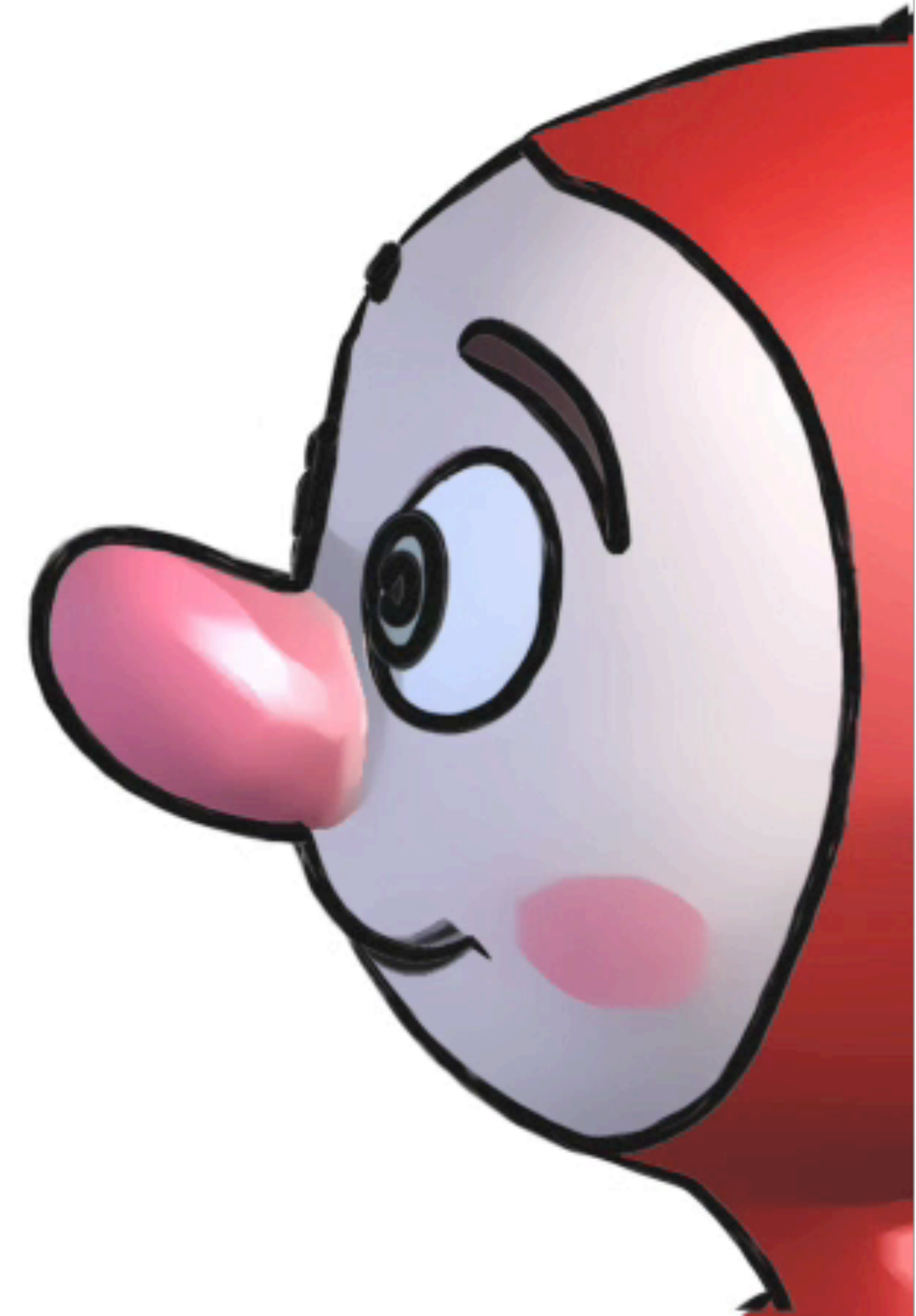


Interpolated
contours



**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](https://github.com/benardp/PaintTweenContours)

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](https://github.com/benardp/PaintTweenContours)

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