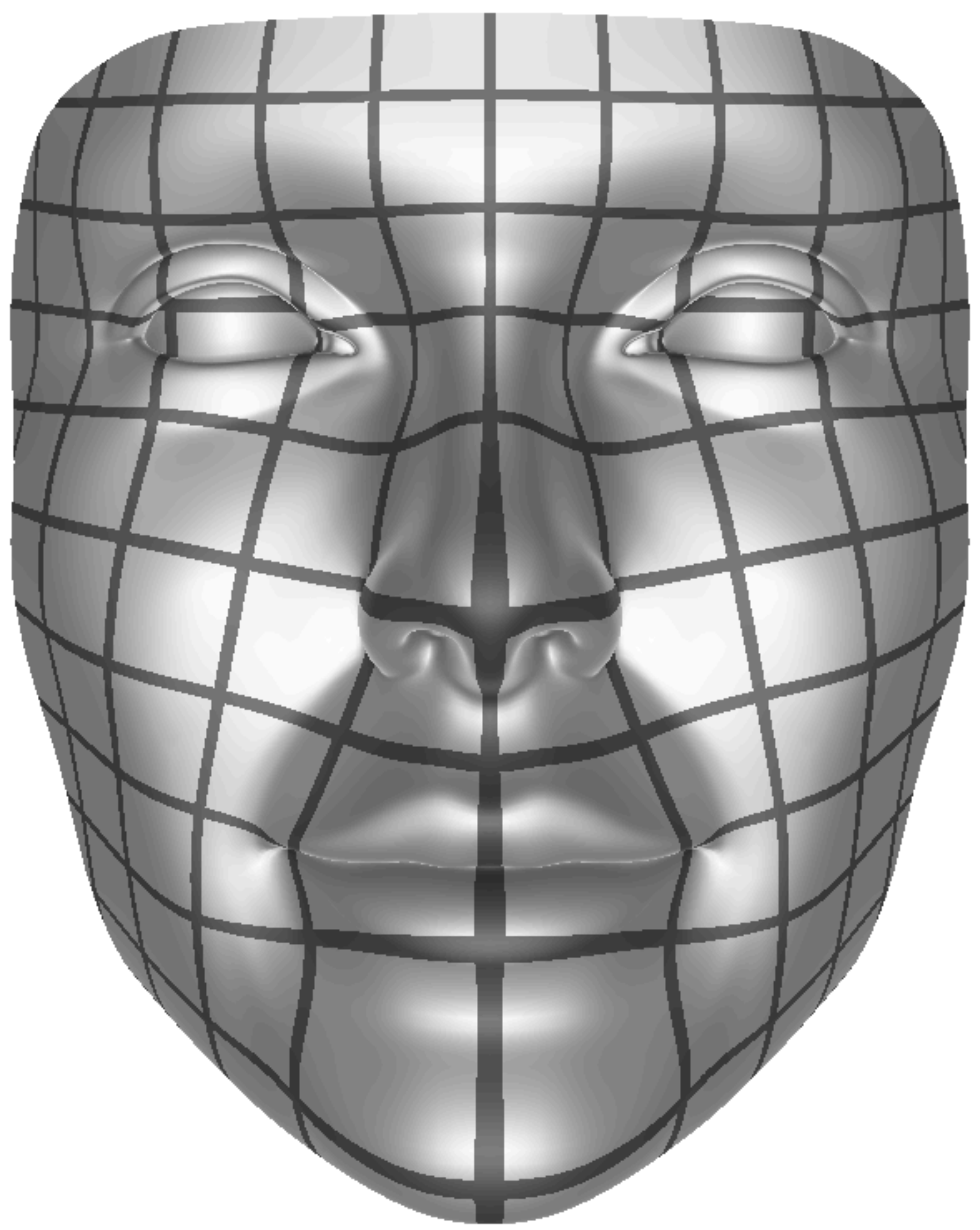
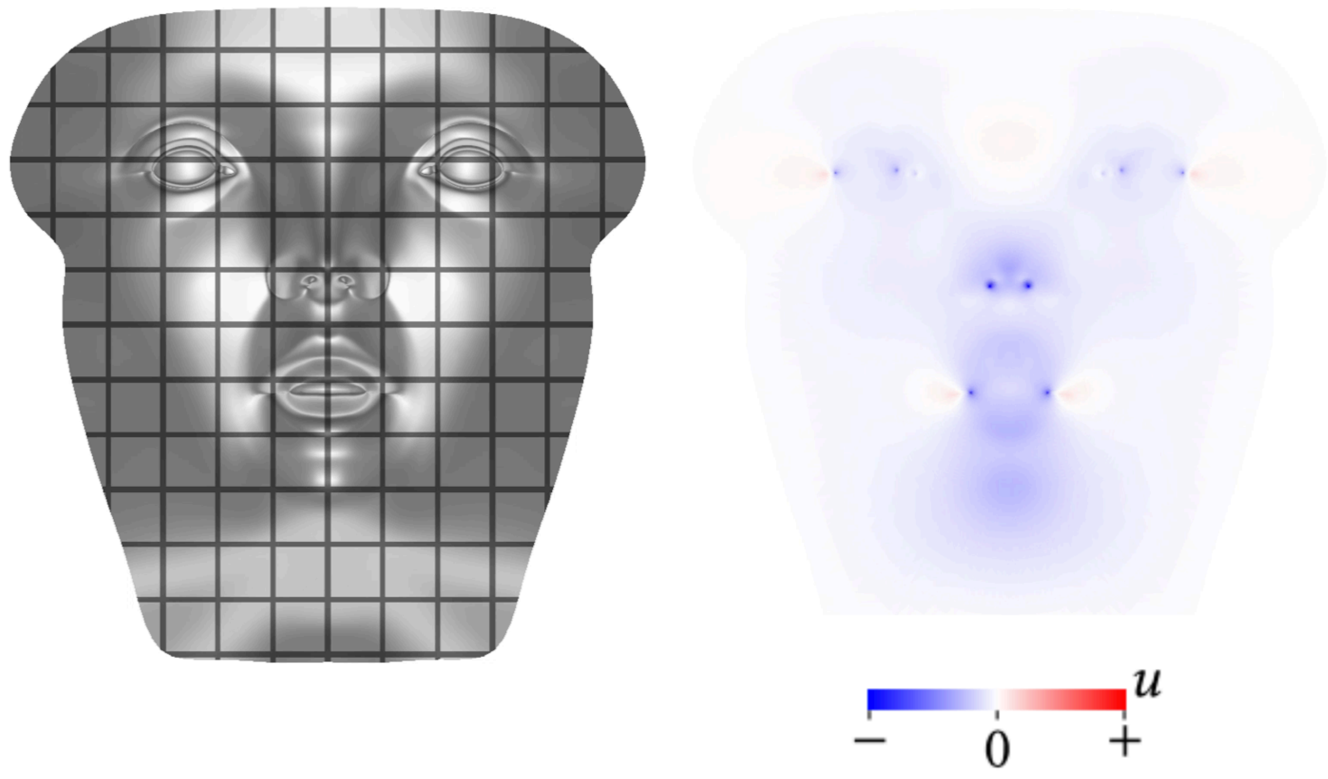
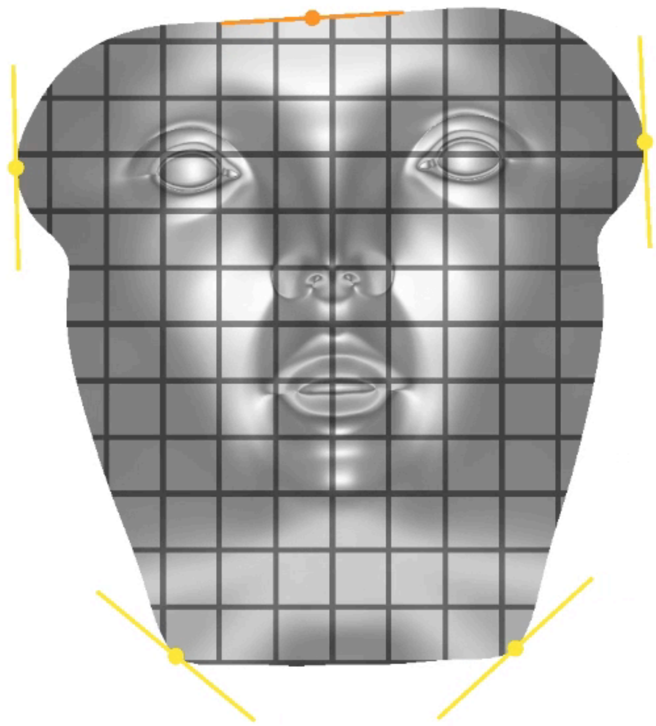
Boundary First Flattening (BFF)

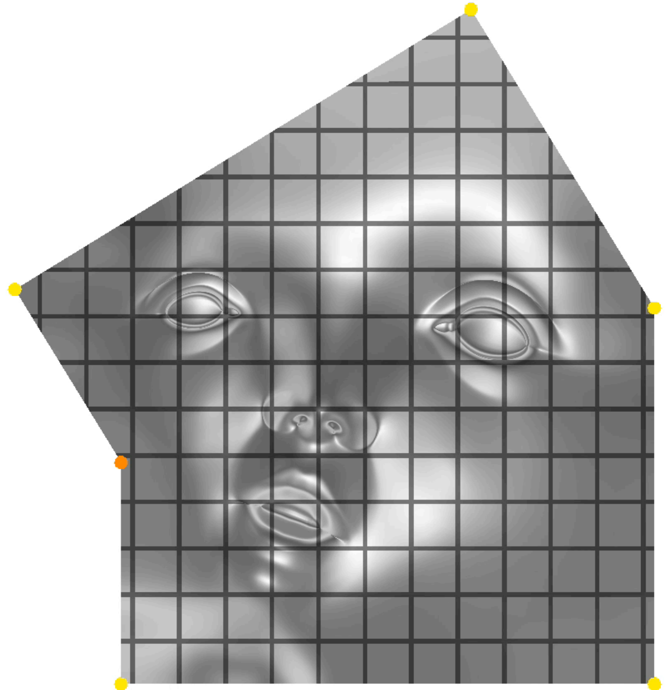




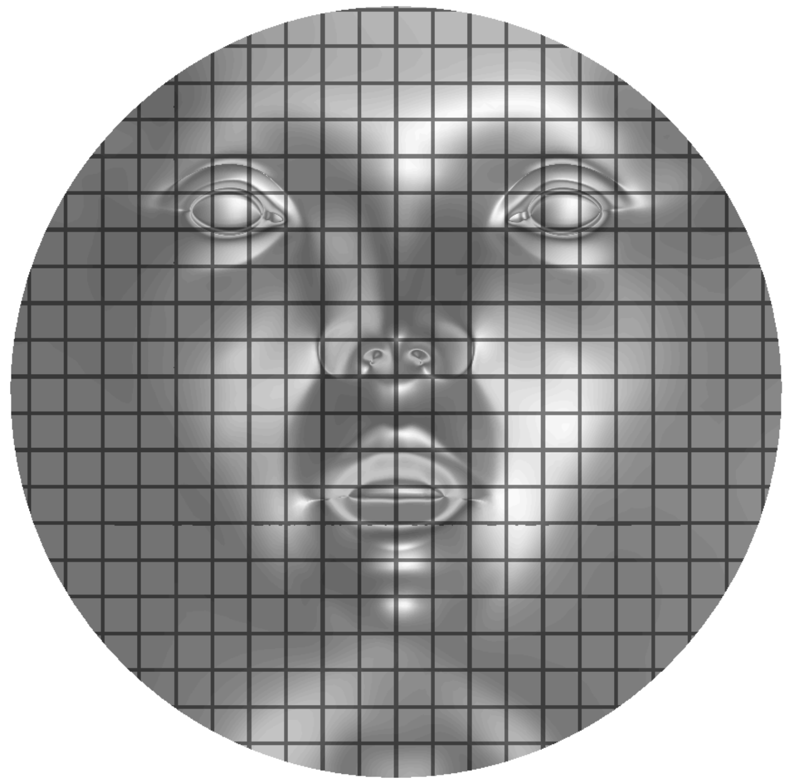
Minimal Area Distortion
(Fully Automatic)



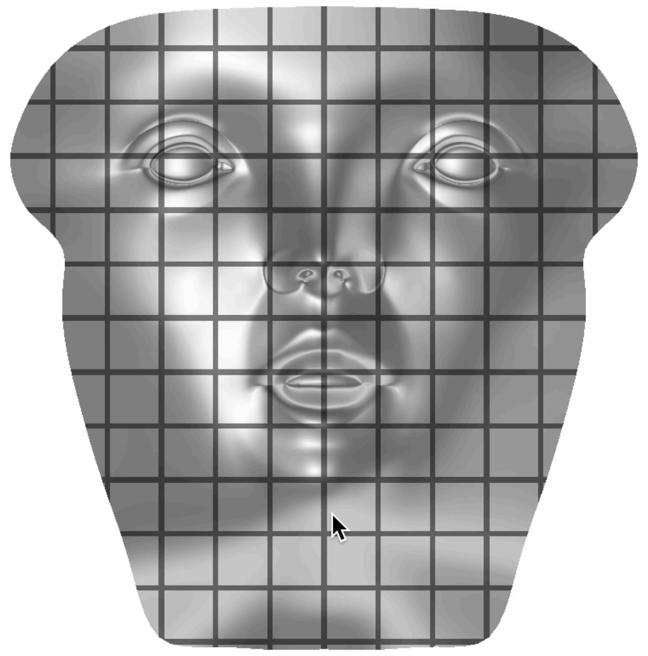
Direct Editing



Sharp Corners

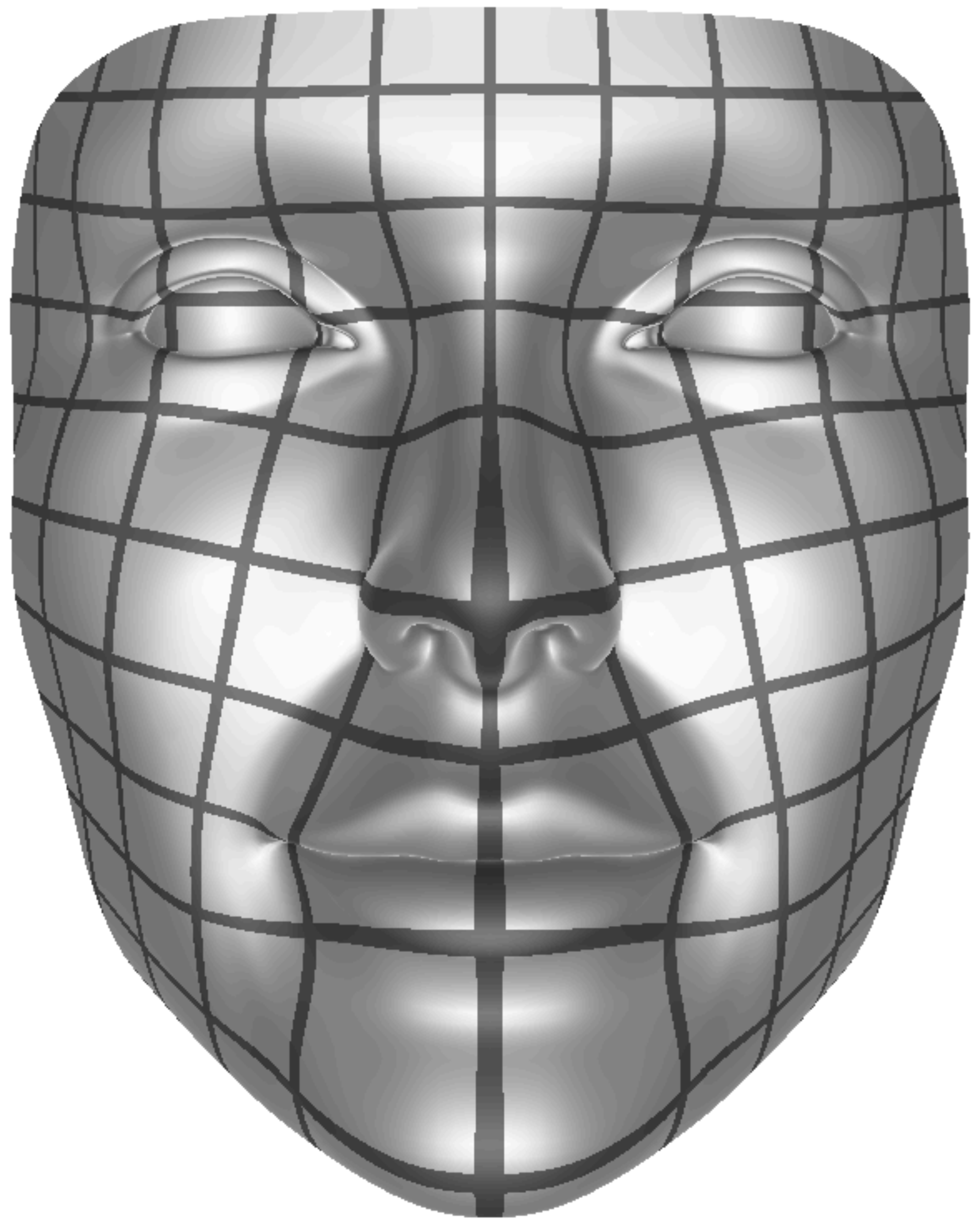


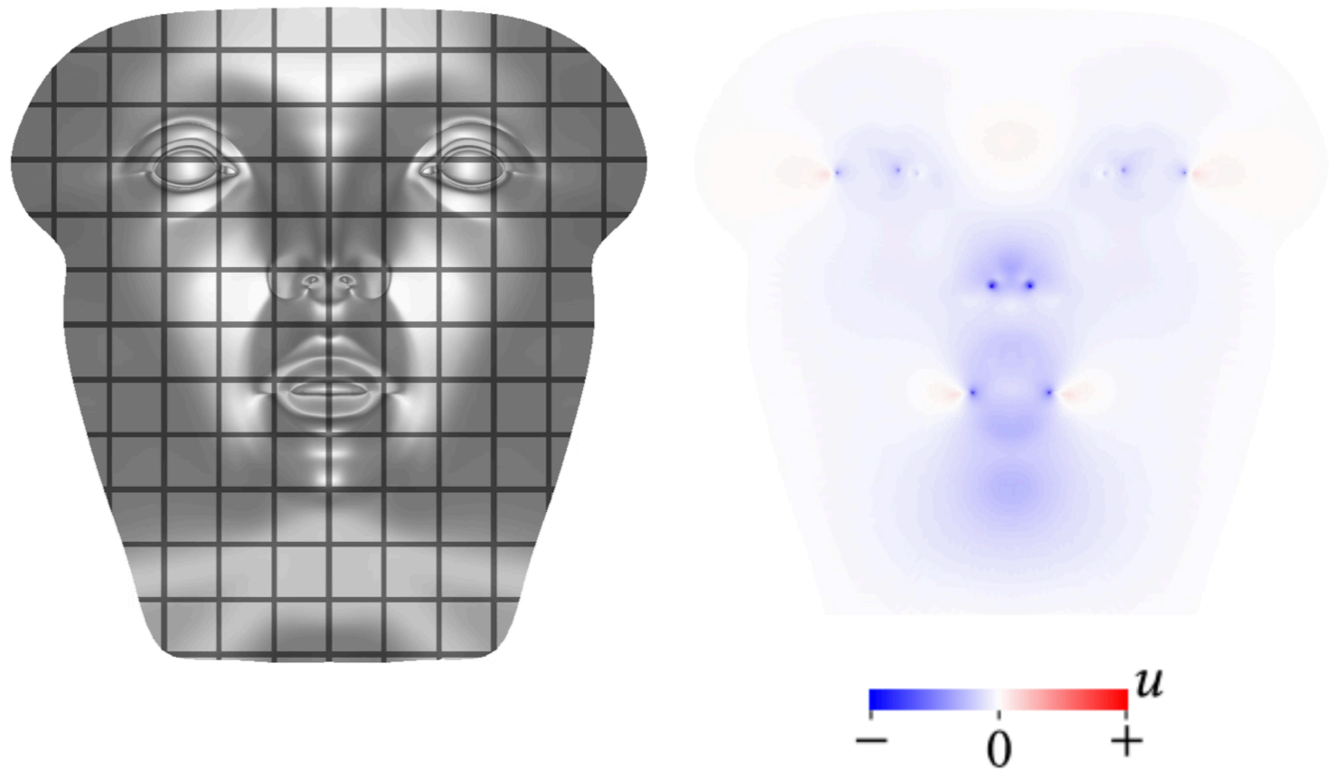
Arbitrary Target Shapes



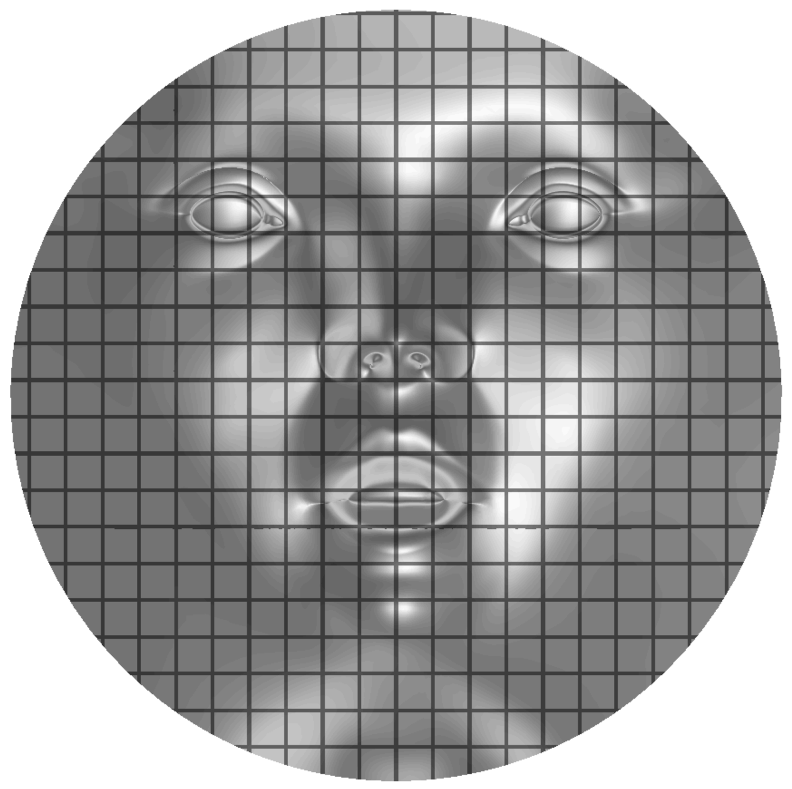
Cone Singularities

BFF: A New Paradigm for Conformal Parameterization

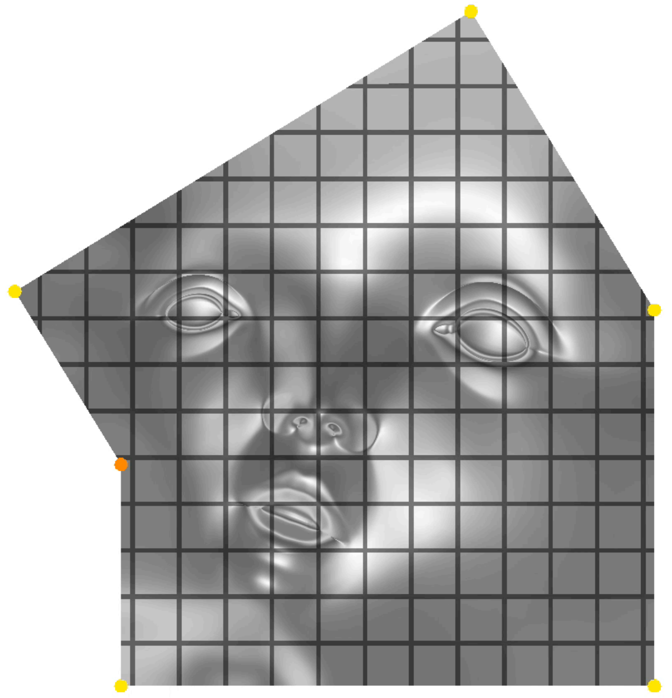




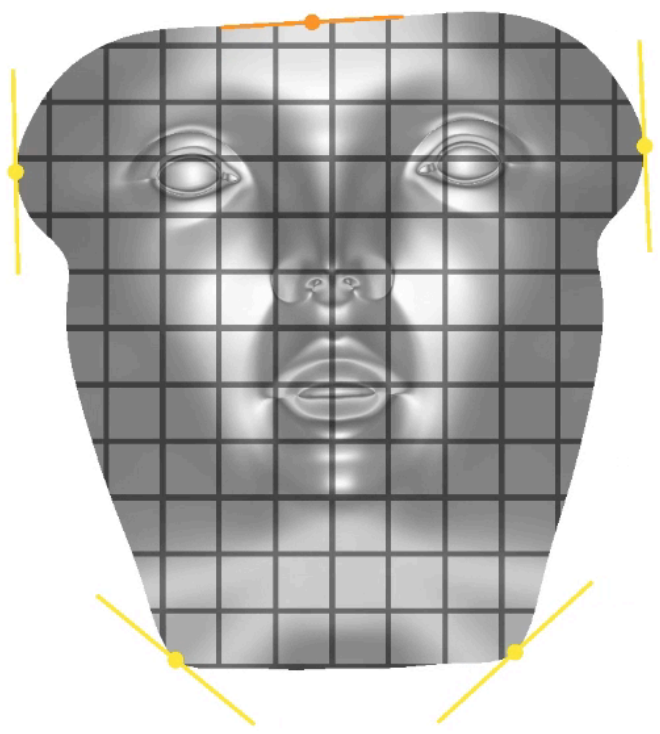
Minimal Area Distortion
(Fully Automatic)



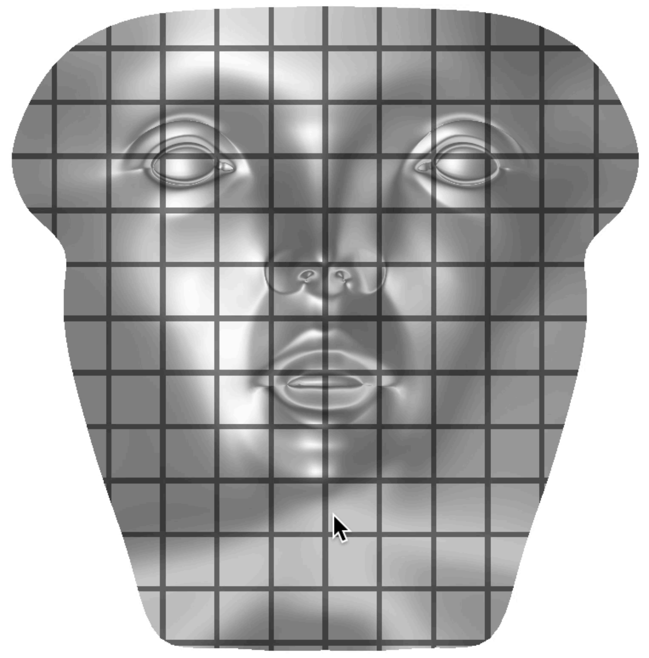
Arbitrary Target Shapes



Sharp Corners

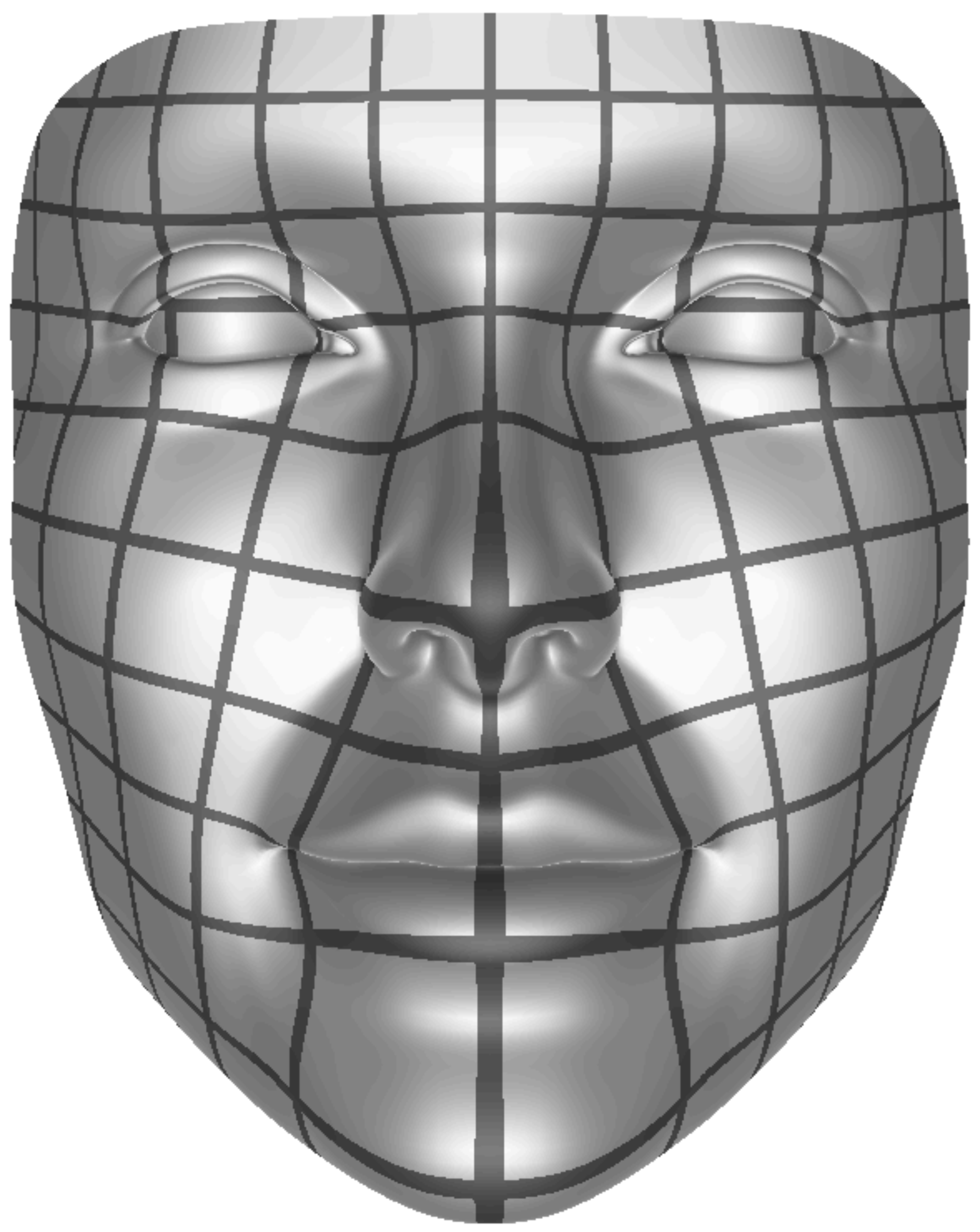


Direct Editing



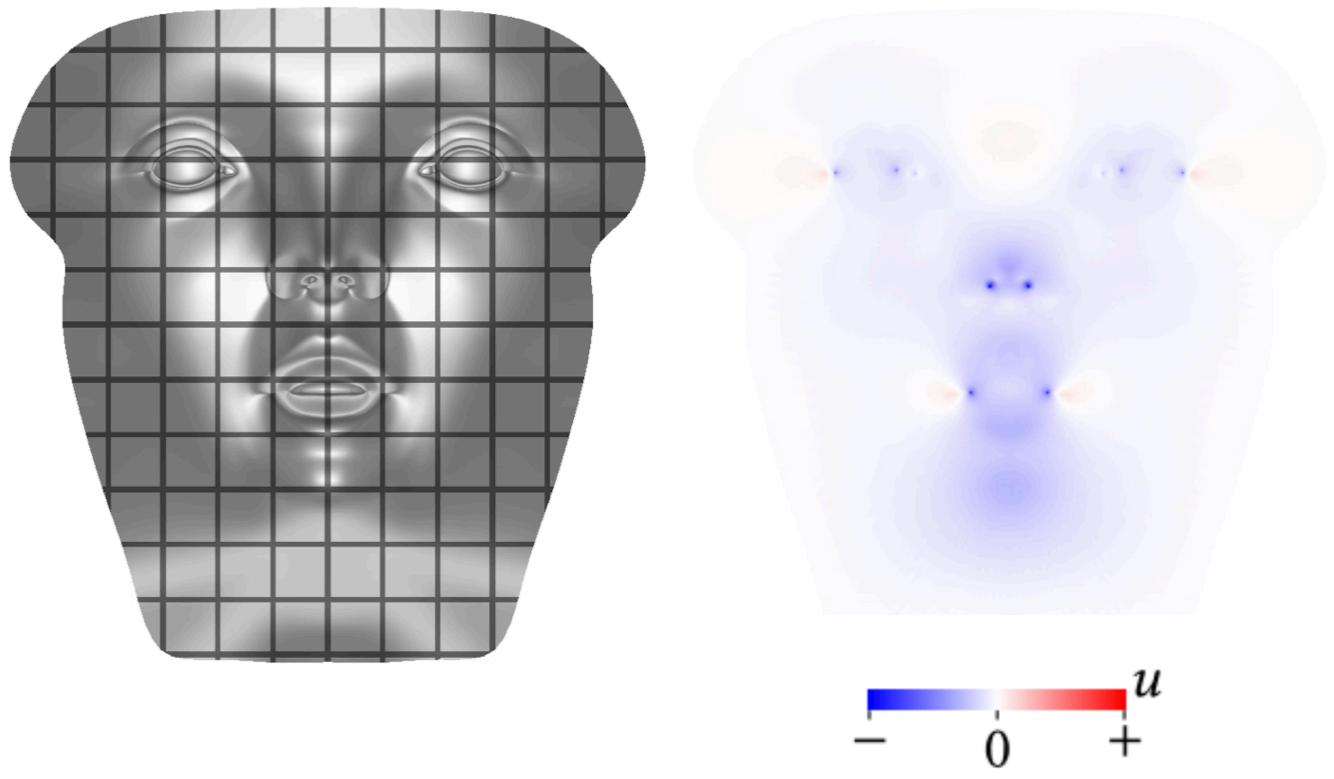
Cone Singularities

BFF: A New Paradigm for Conformal Parameterization

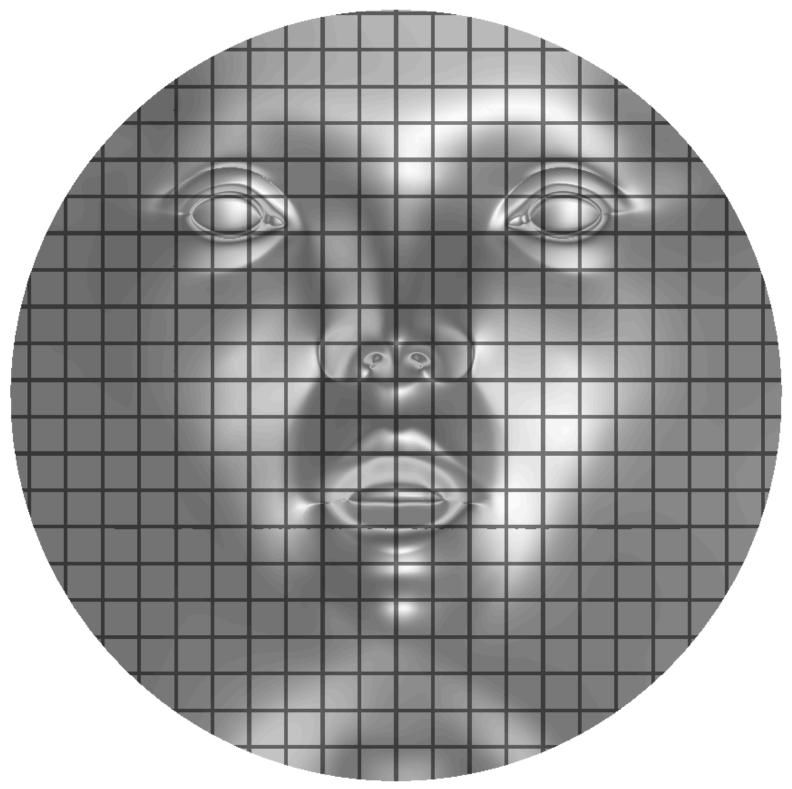


Quality of nonlinear methods

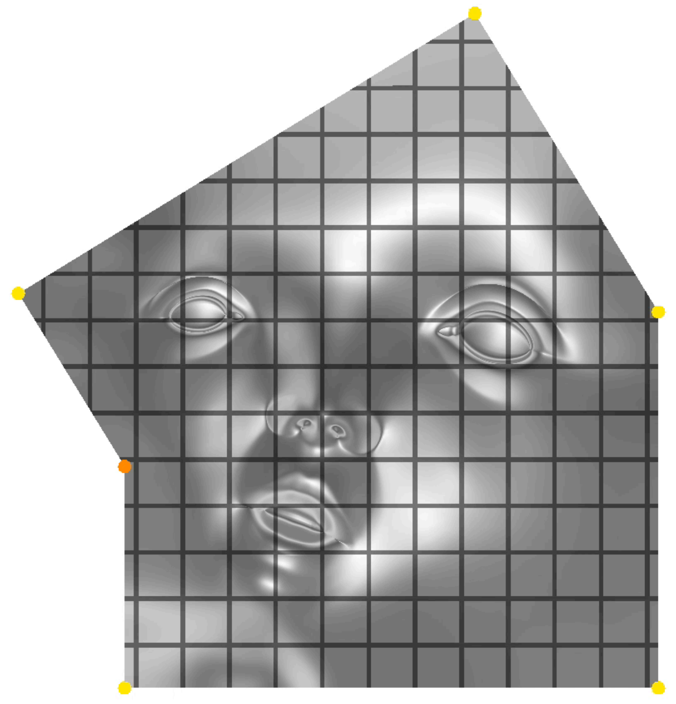
Faster than existing linear schemes



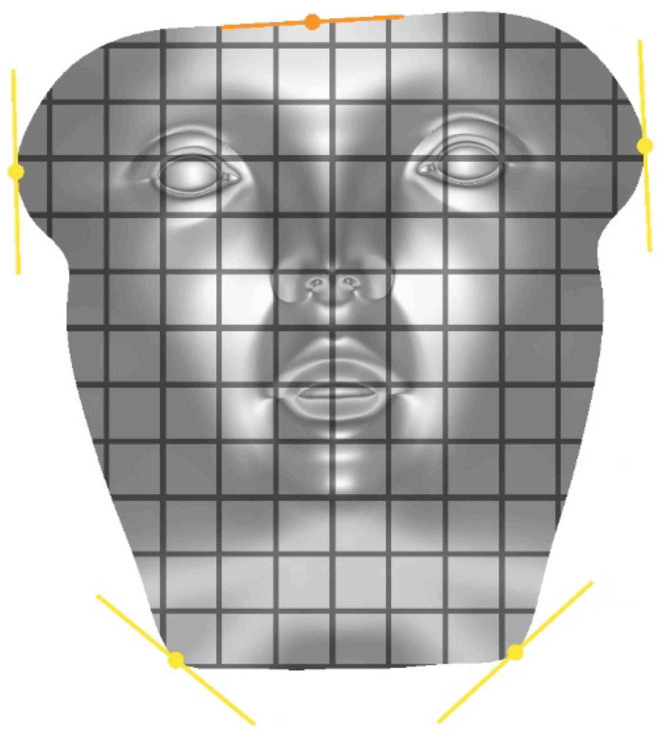
Minimal Area Distortion
(Fully Automatic)



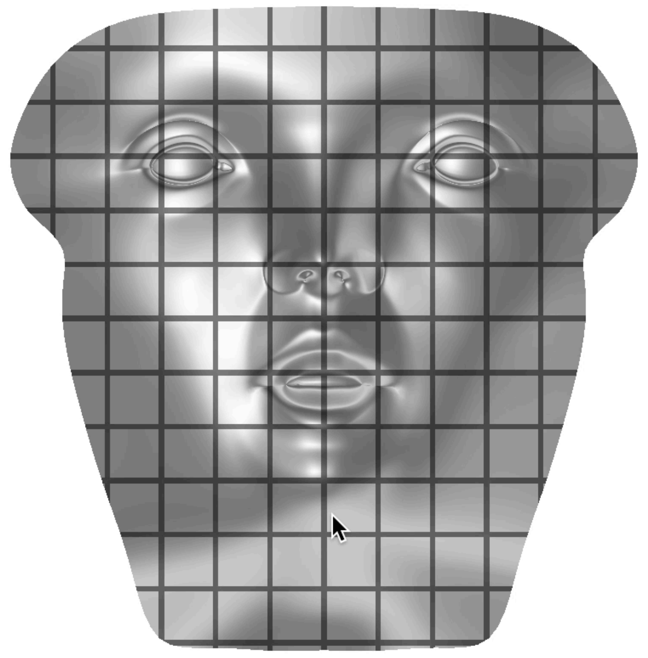
Arbitrary Target Shapes



Sharp Corners

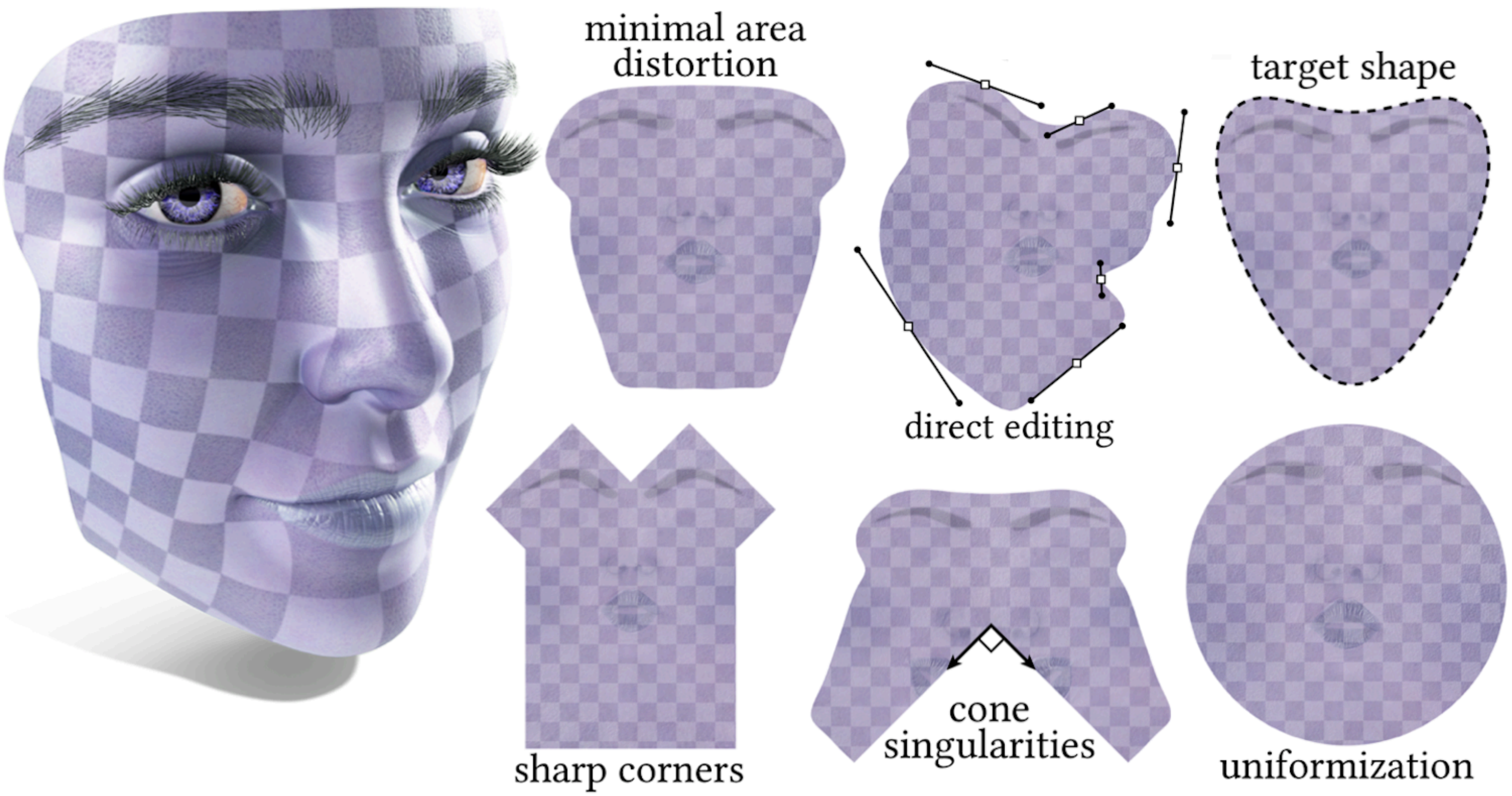


Direct Editing



Cone Singularities

Boundary First Flattening (BFF)



Full Control Over Target Shape

Real Time/ Scalable

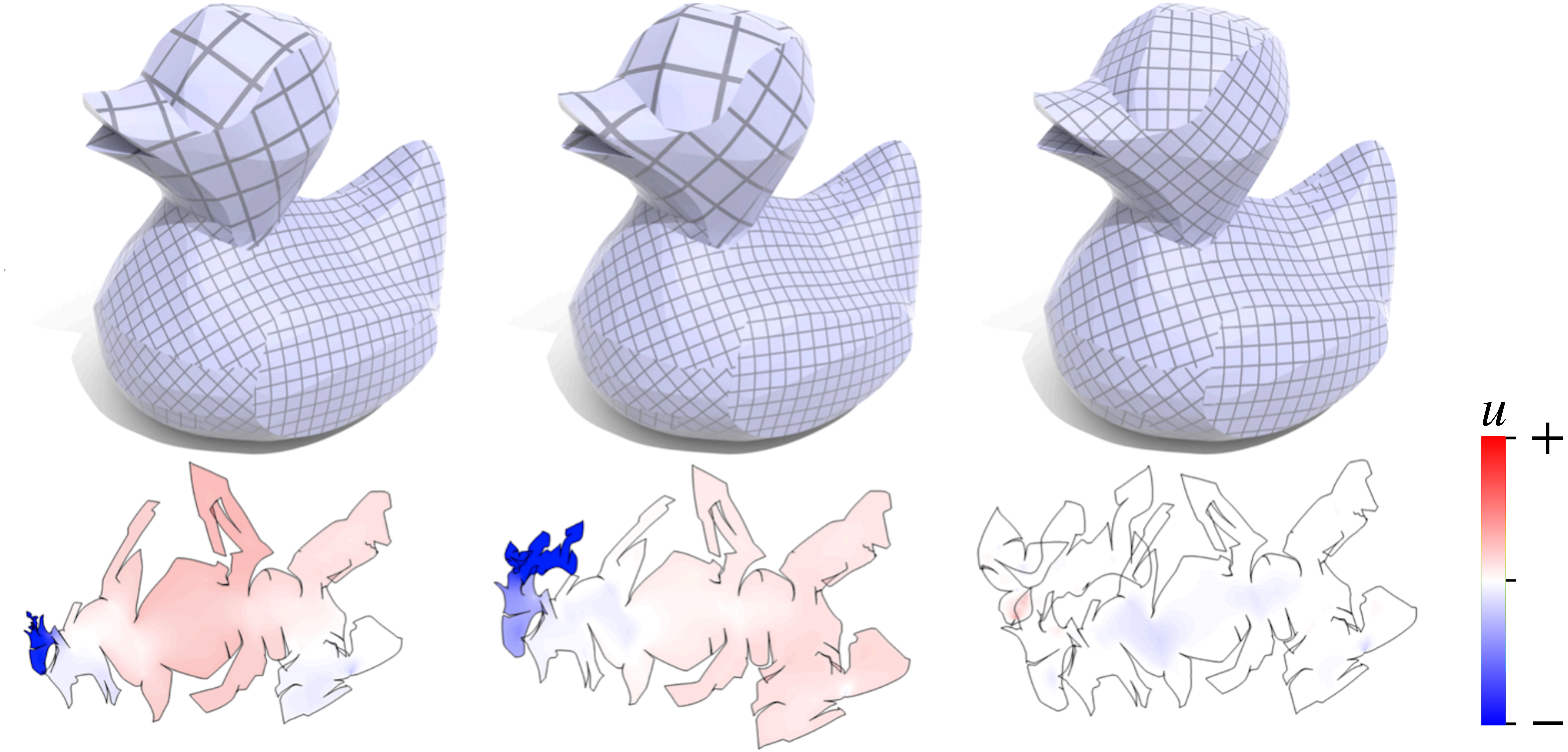
Robust

BFF is Robust

Spectral Conformal (SCP)
[Mullen et al 2008]

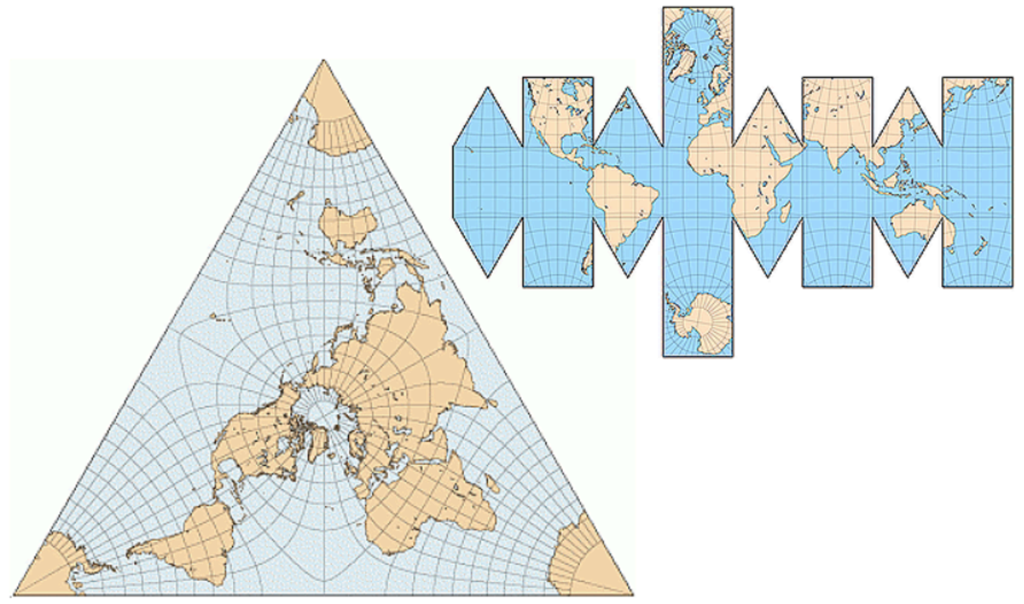
Angle Based (ABF)
[Zayer et al 2008]

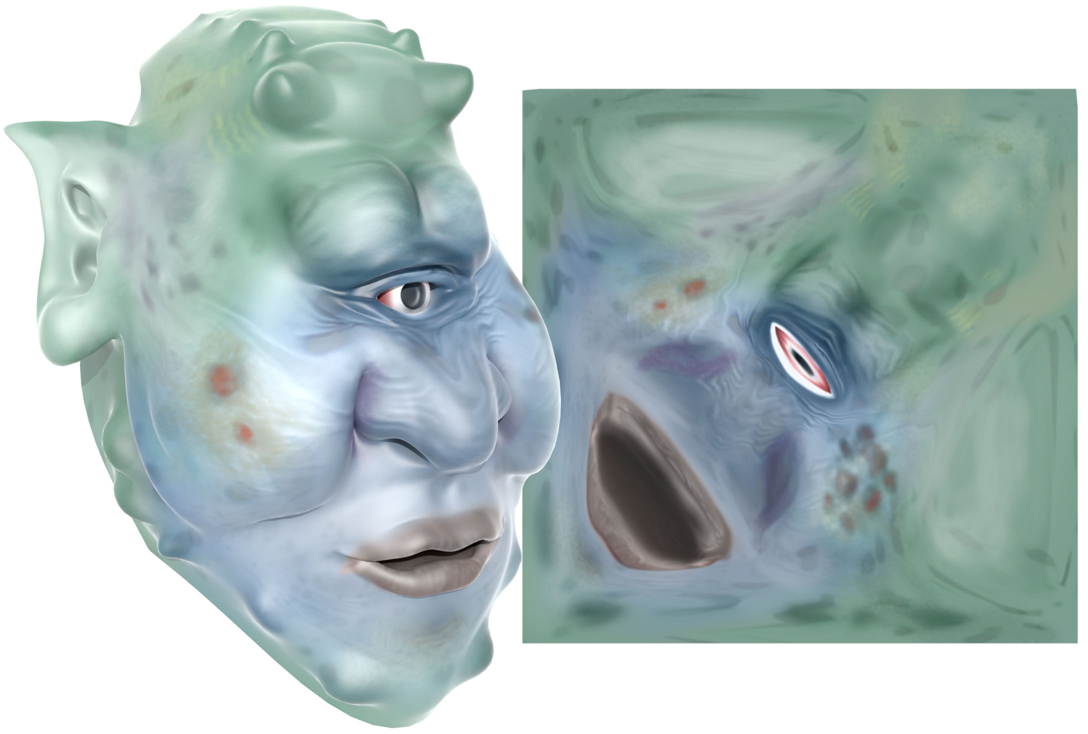
Boundary First (BFF)
[Sawhney et al 2017]

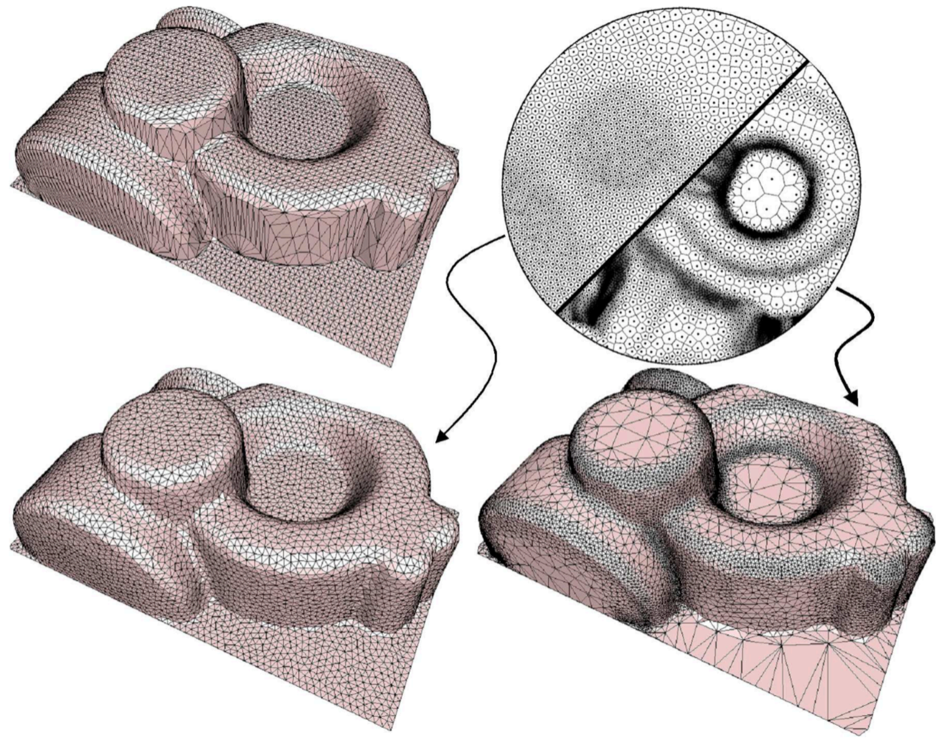


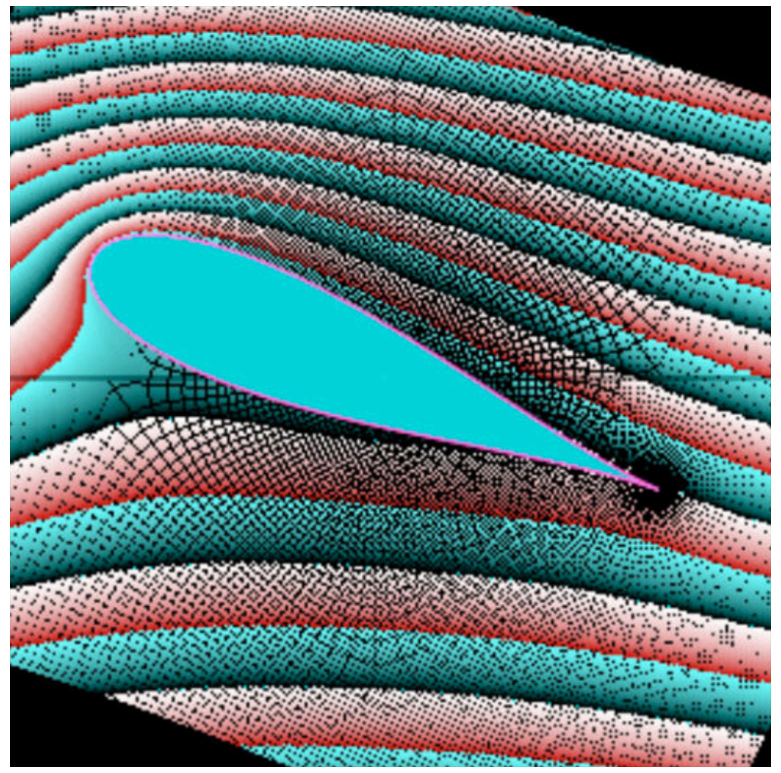
MOTIVATION

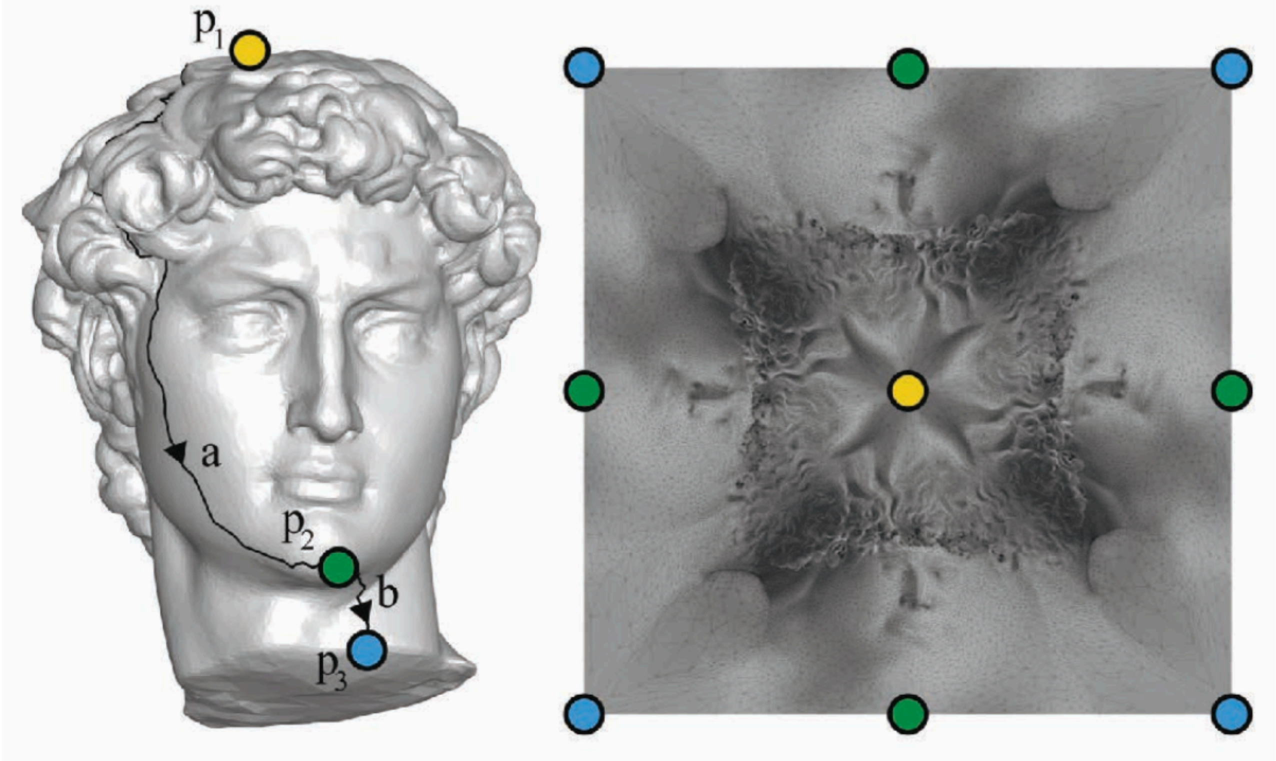
Conformal Flattening Applications


Cartography


Texture Mapping

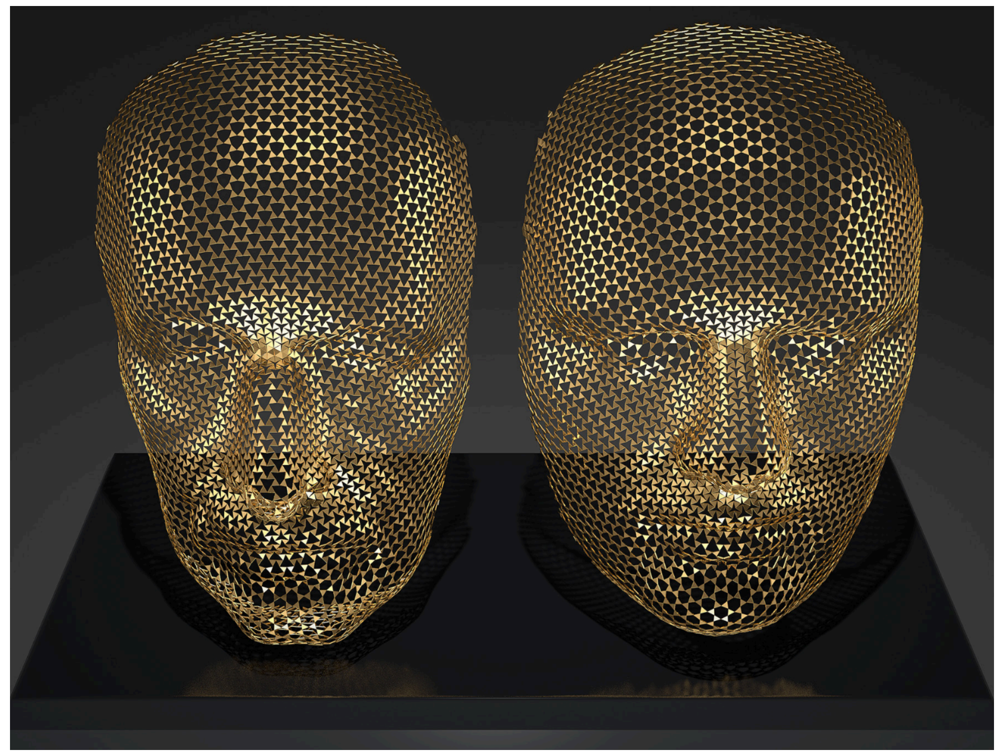

Remeshing


Physical Simulation



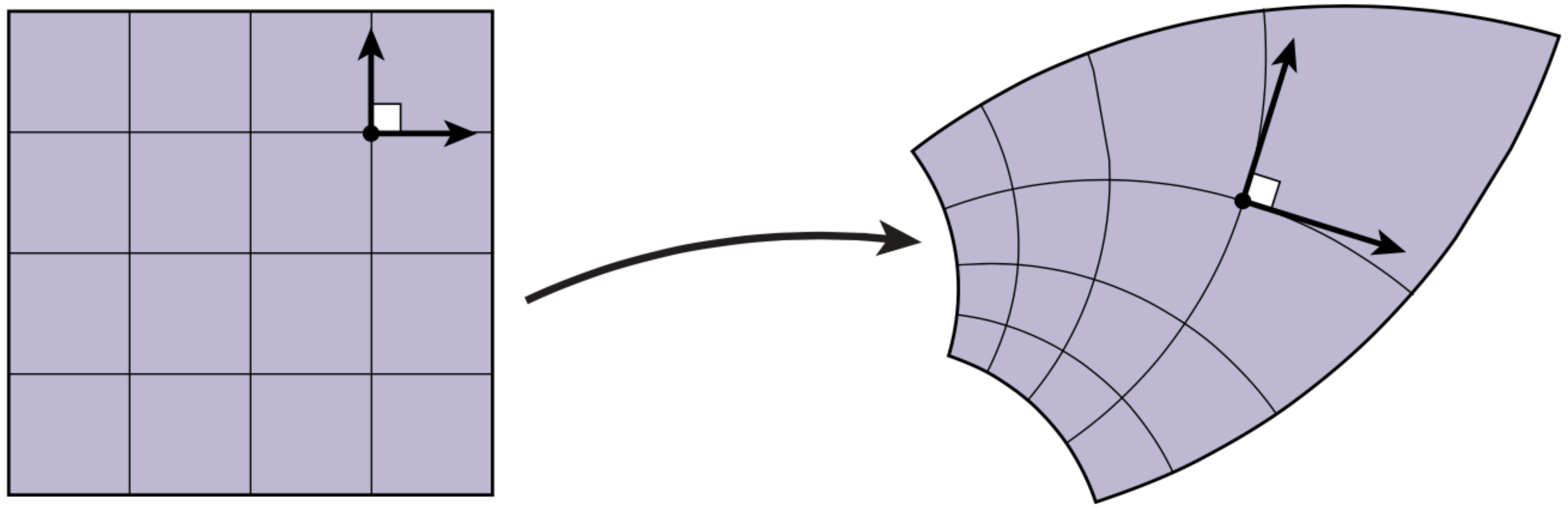
Machine Learning


Computational Design


3D Fabrication

Why Conformal?

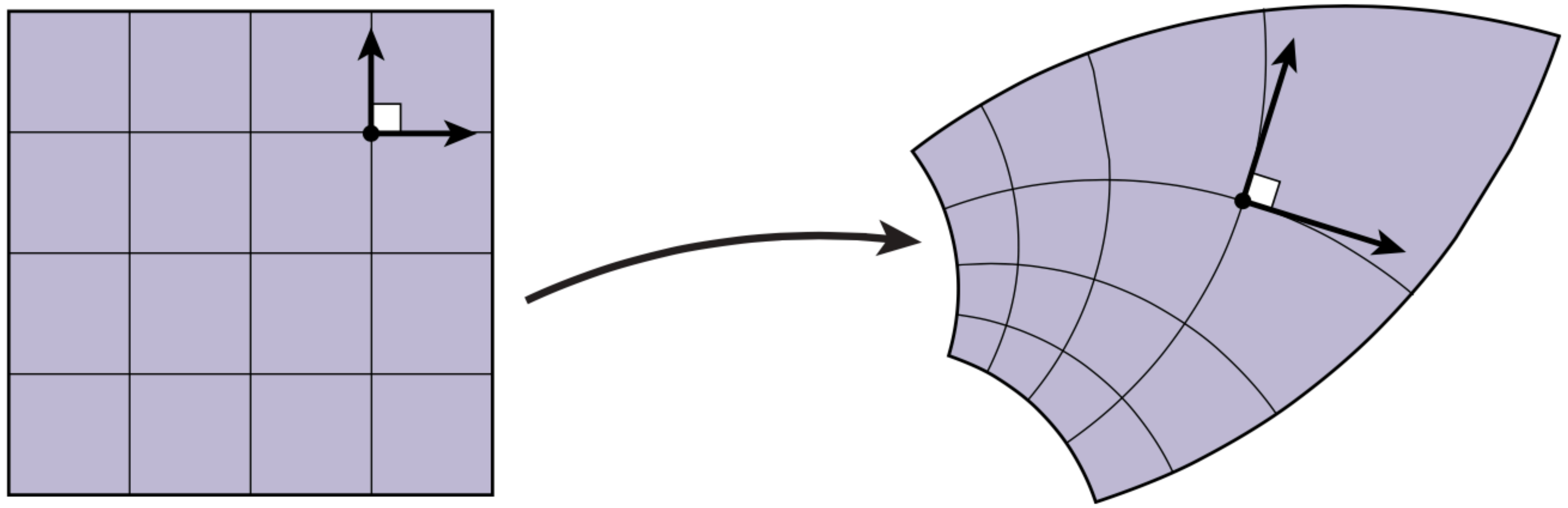
Why so much interest in maps that preserve angles?



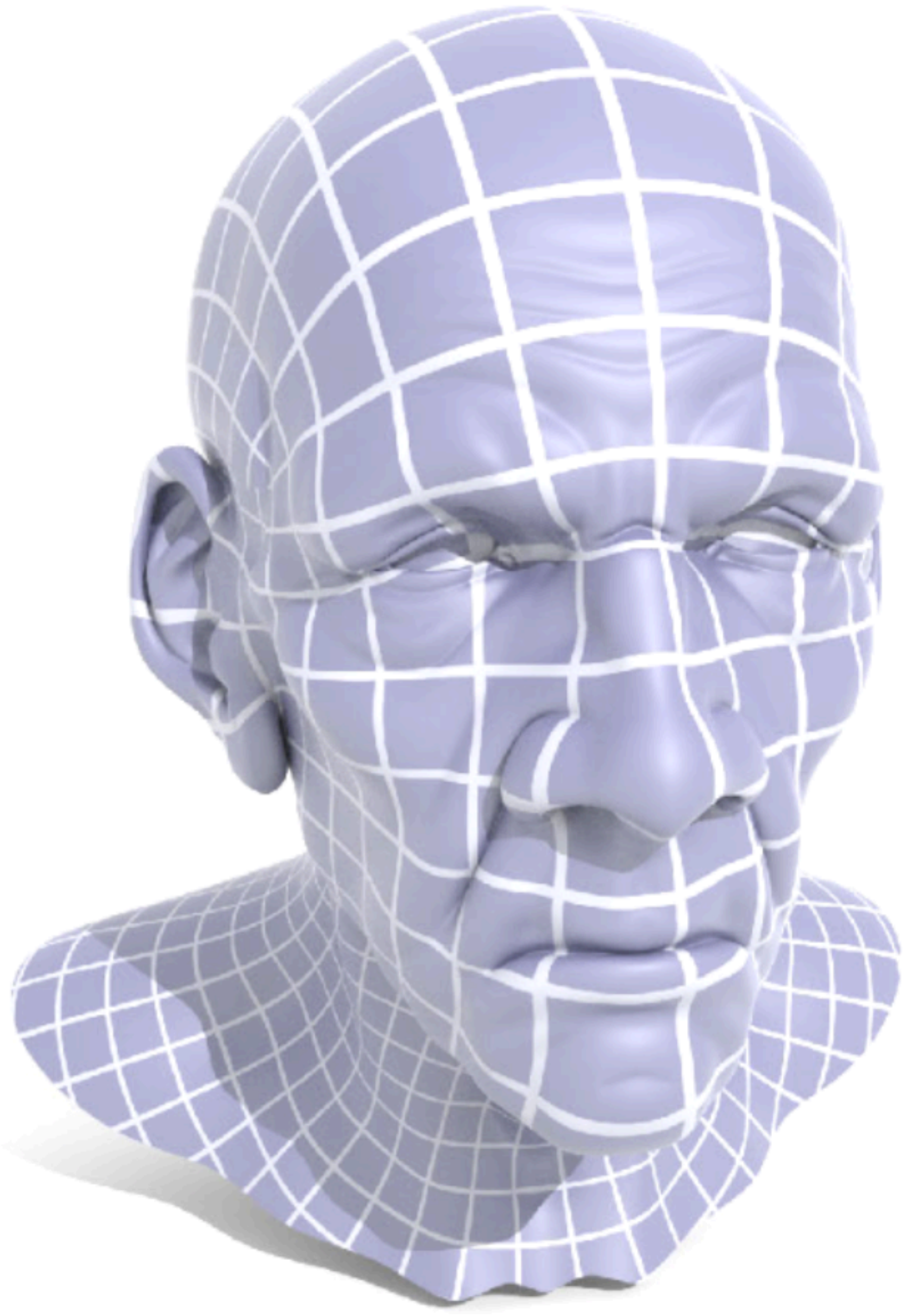
Why Conformal?

Why so much interest in maps that preserve angles?

QUALITY: Often comparable to nonlinear schemes



SLIM + PARDISO
[Rabinovich et al 2016]



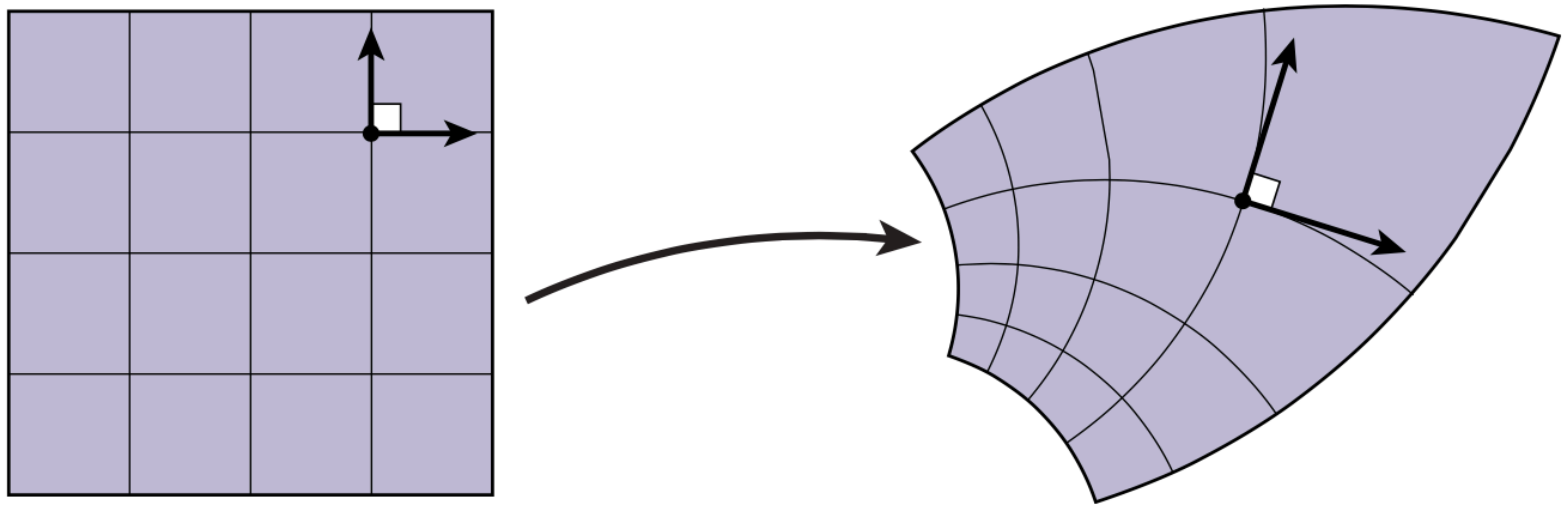
BFF
[Sawhney & Crane 2017]

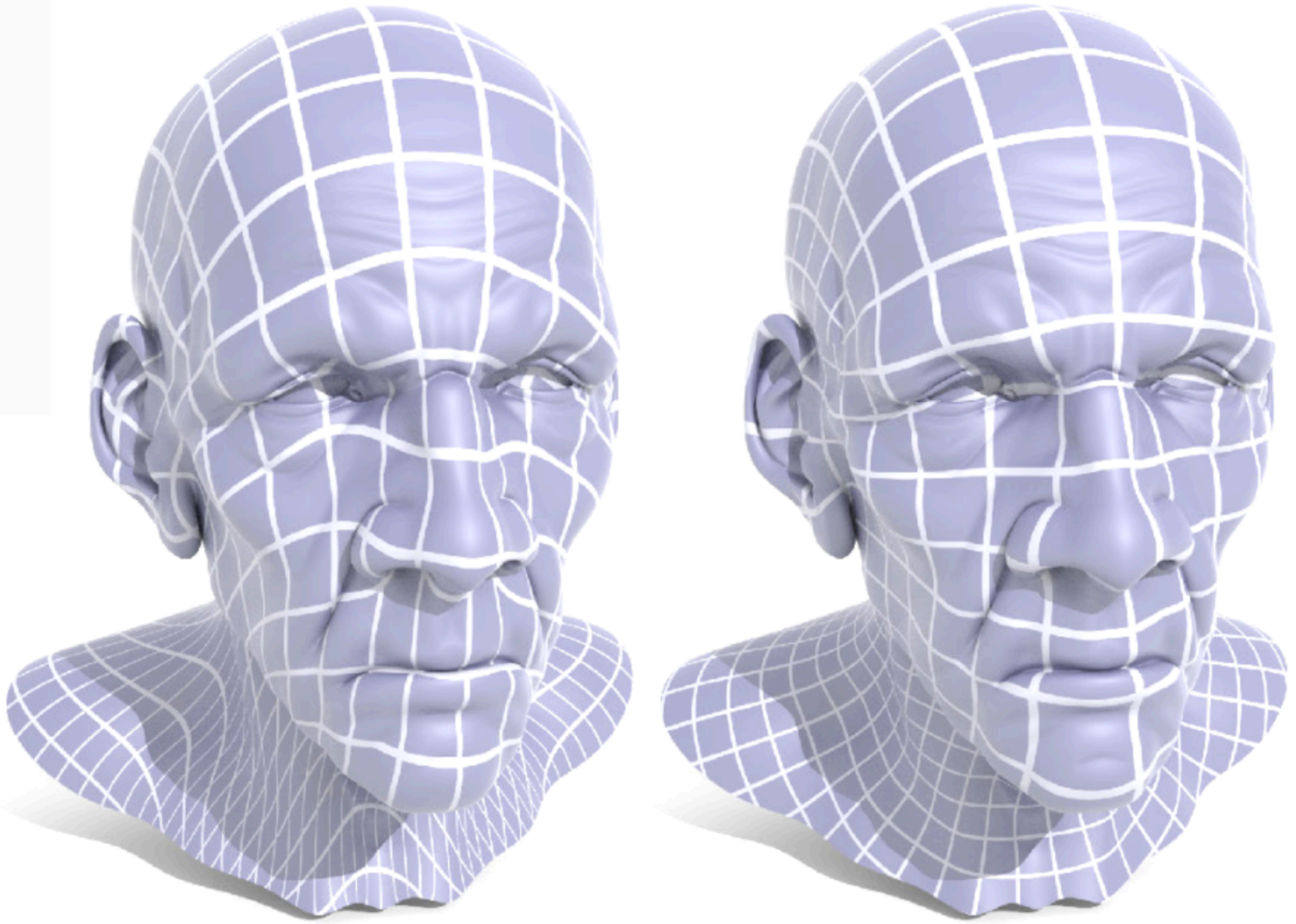
Why Conformal?

Why so much interest in maps that preserve angles?

QUALITY: Often comparable to nonlinear schemes

EFFICIENCY: Often only one sparse factorization!





SLIM + PARDISO
[Rabinovich et al 2016]
15.9s

BFF
[Sawhney & Crane 2017]
0.12s **(126x faster)**

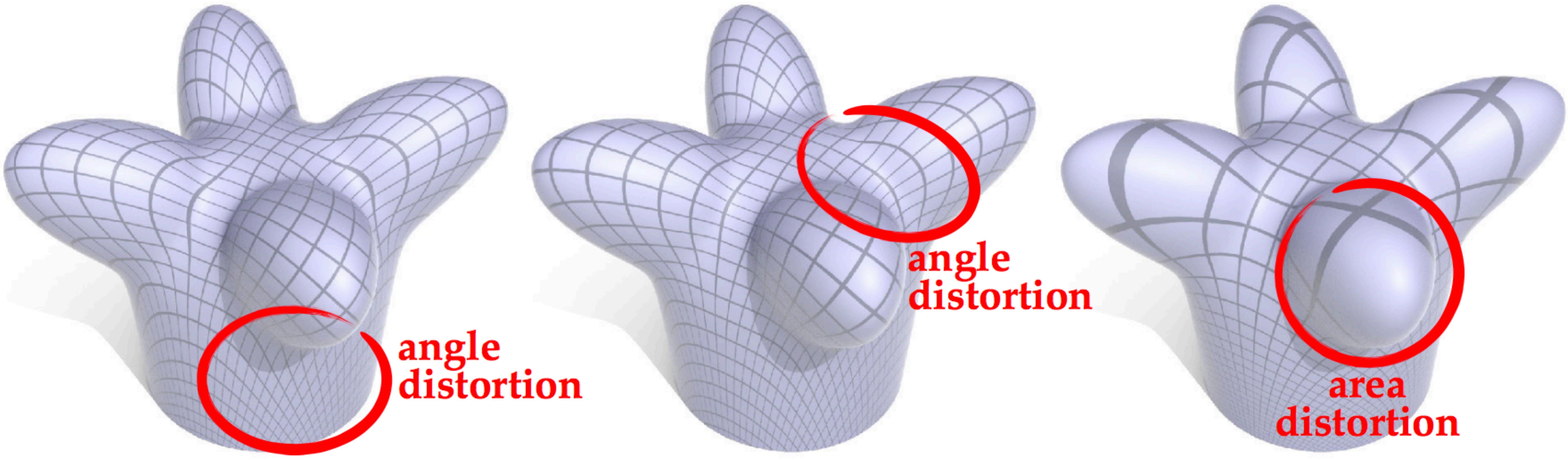
Why Conformal?

What about other energies like ARAP, Symmetric Dirichlet, …?

Why Conformal?

What about other energies like ARAP, Symmetric Dirichlet, …?

Distortion is inevitable



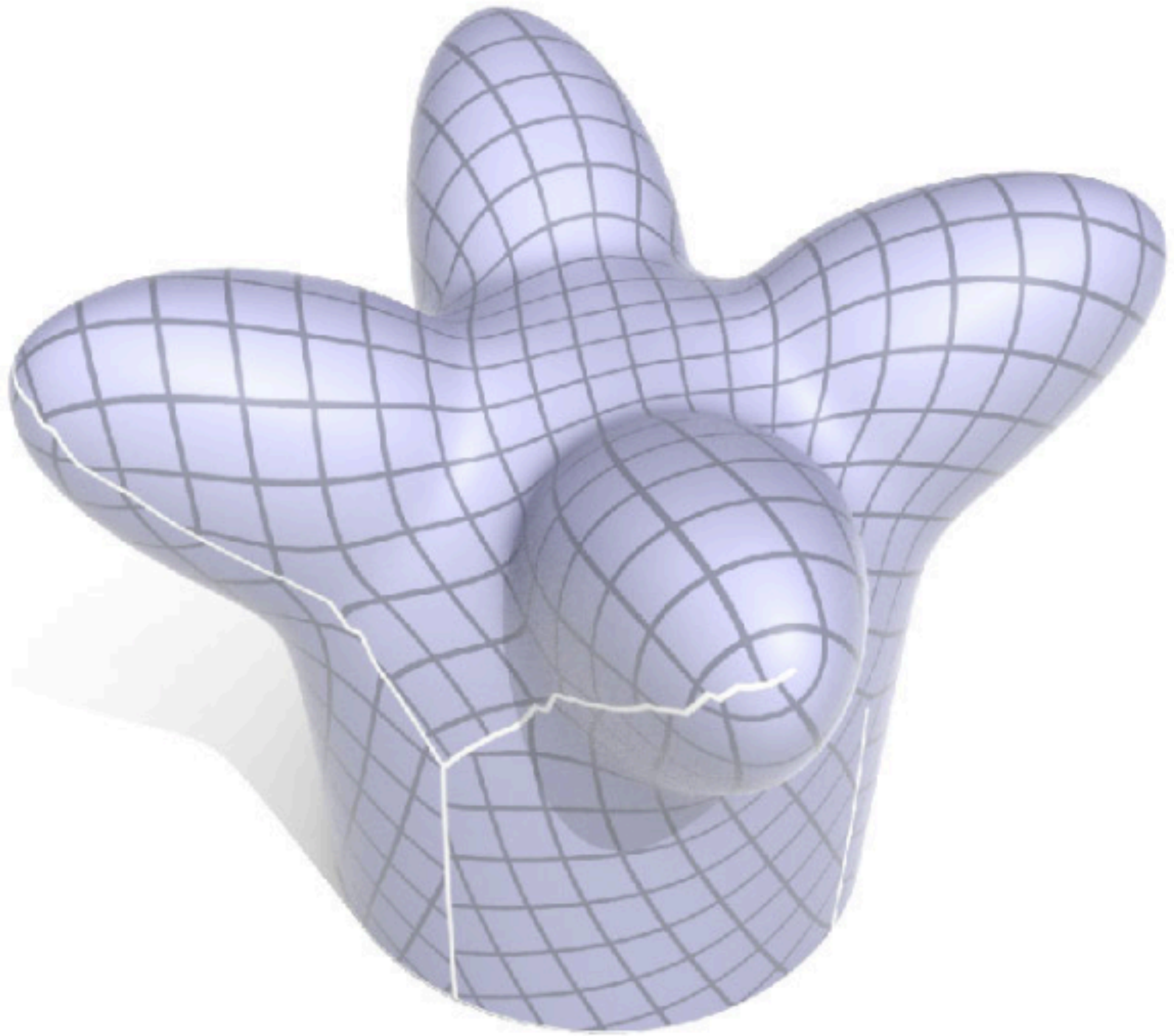
As Rigid As Possible Symmetric Dirichlet Conformal

Why Conformal?

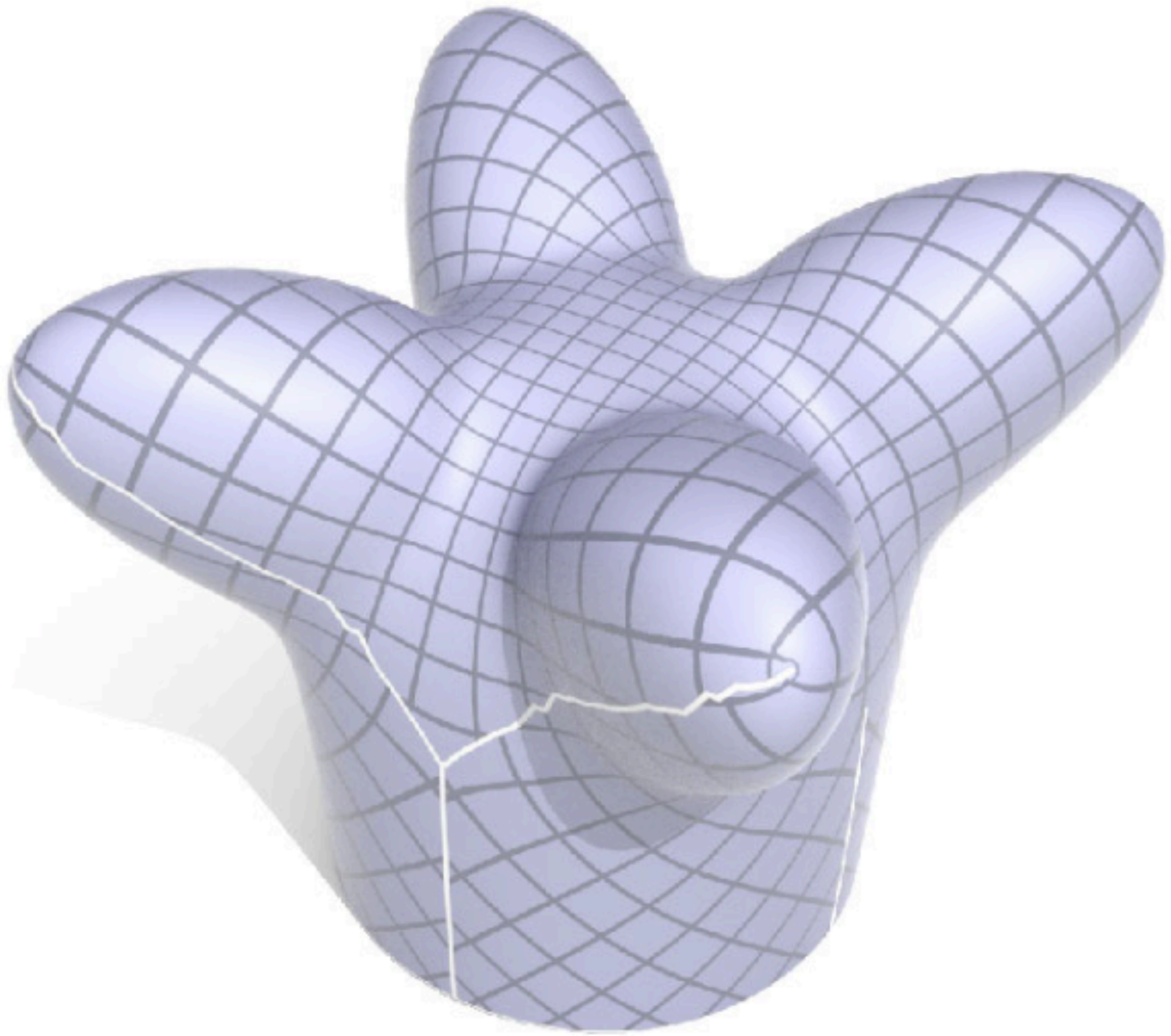
What about other energies like ARAP, Symmetric Dirichlet, …?

Distortion is inevitable

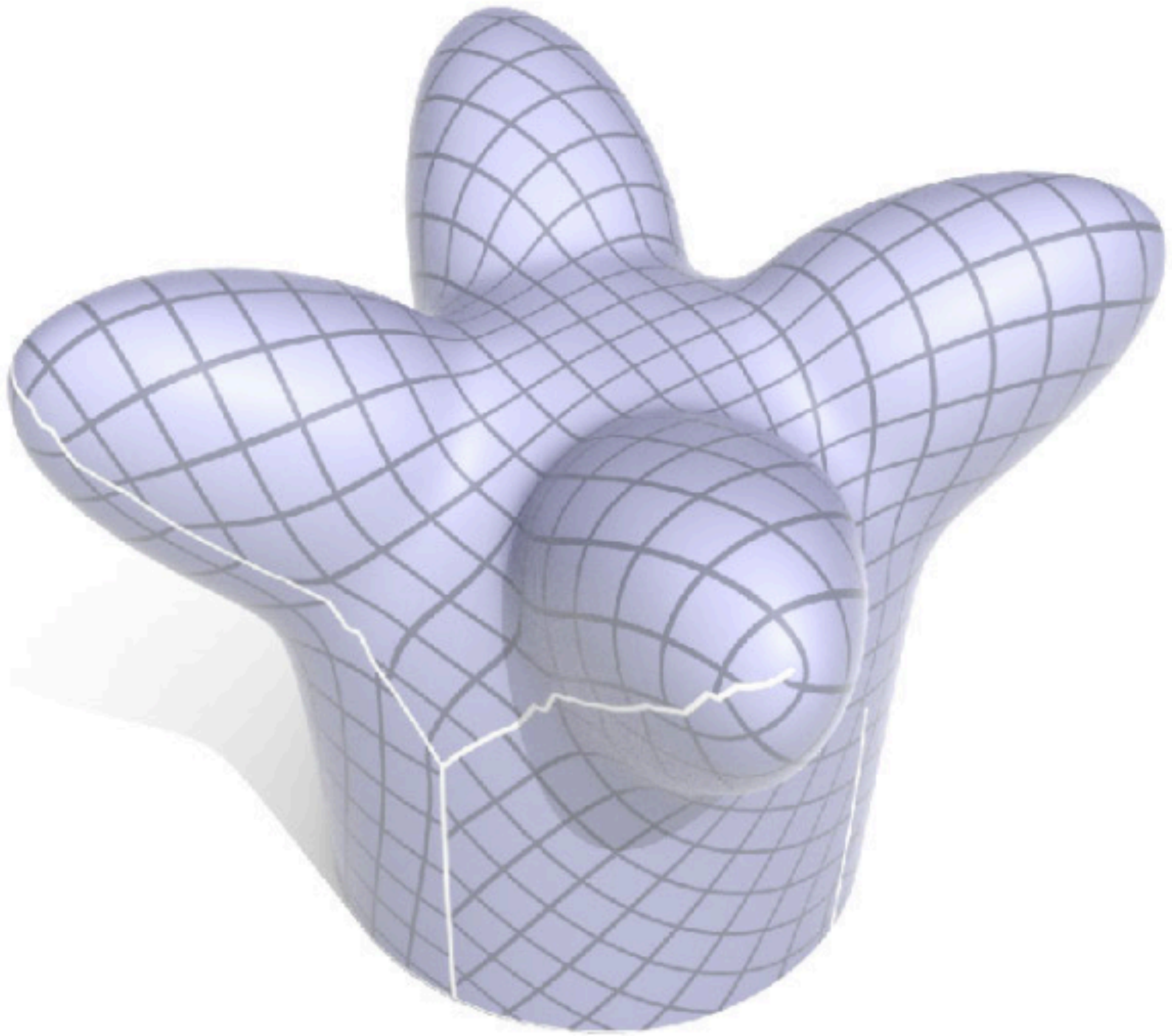
Choice of cuts is **far** more important, with conformal energies **much** cheaper to minimize



As Rigid As Possible



Symmetric Dirichlet



Conformal

Problems

Current linear conformal methods have two major shortcomings:

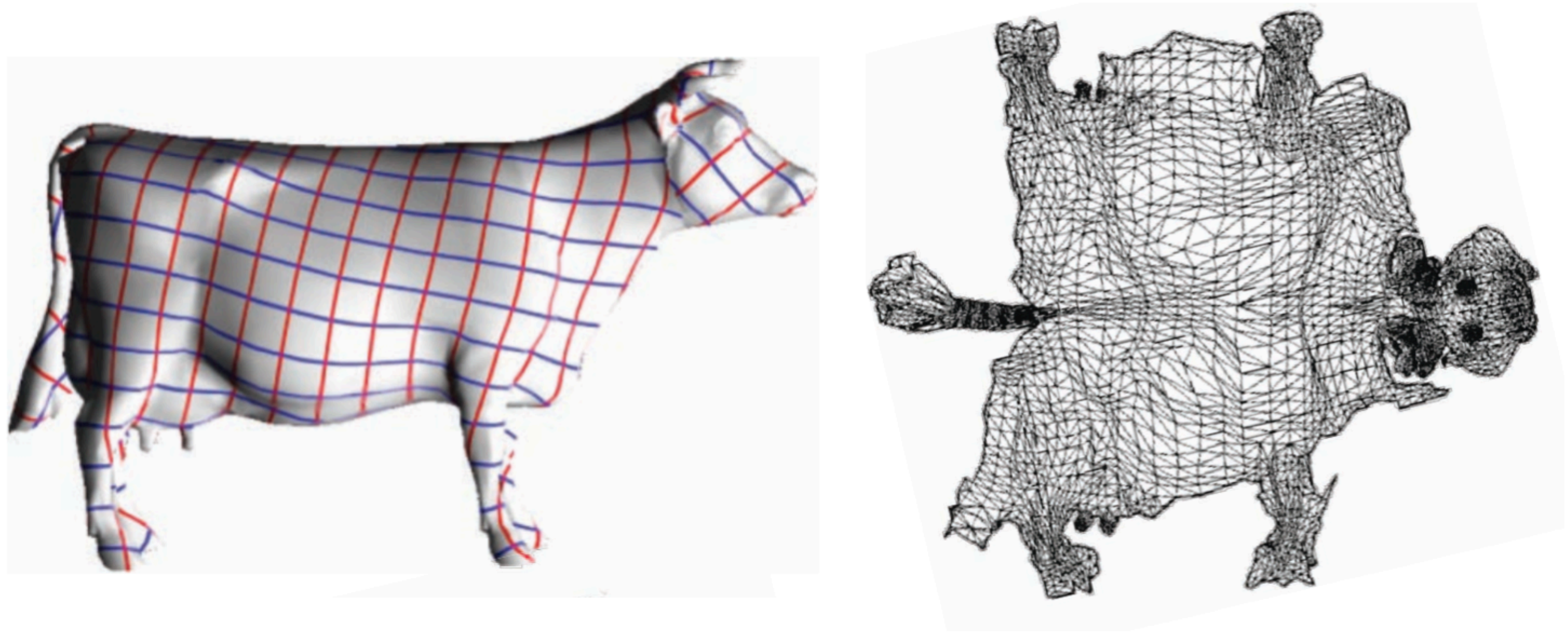
Problems

Current linear conformal methods have two major shortcomings:

NO BOUNDARY CONTROL

User obtains a single, automatic flattening

Must "take it or leave it,” irrespective of quality



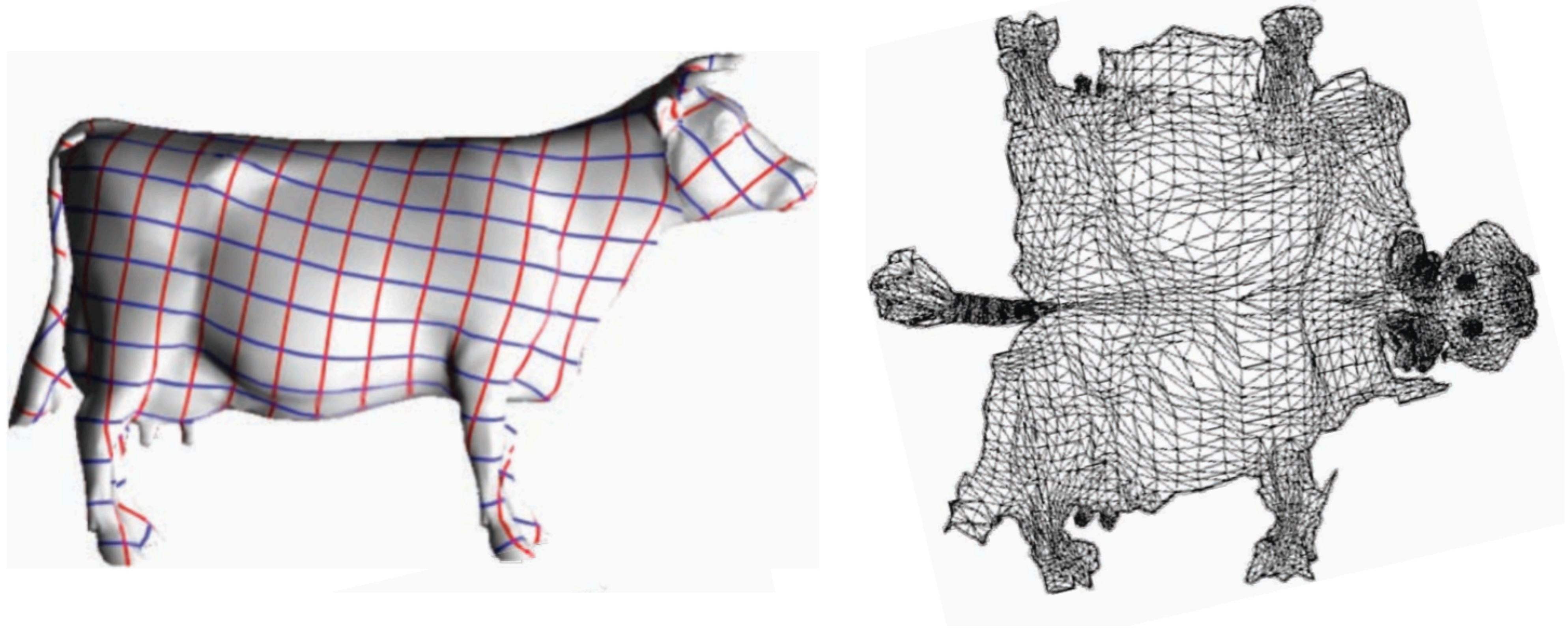
Problems

Current linear conformal methods have two major shortcomings:

NO BOUNDARY CONTROL

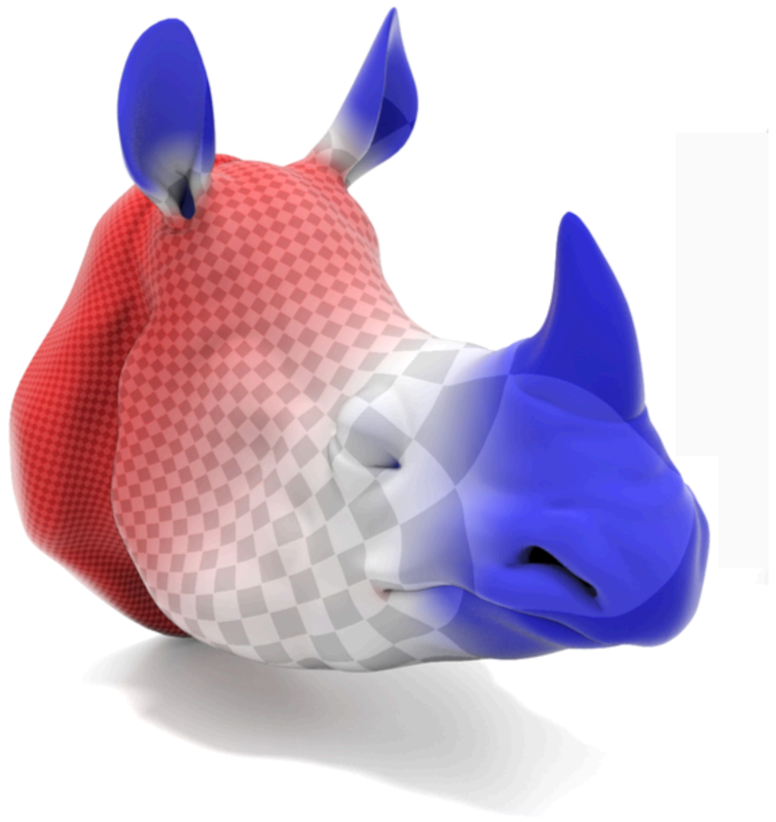
User obtains a single, automatic flattening

Must "take it or leave it," irrespective of quality



SCALE DISTORTION

Conformal maps can scale arbitrarily



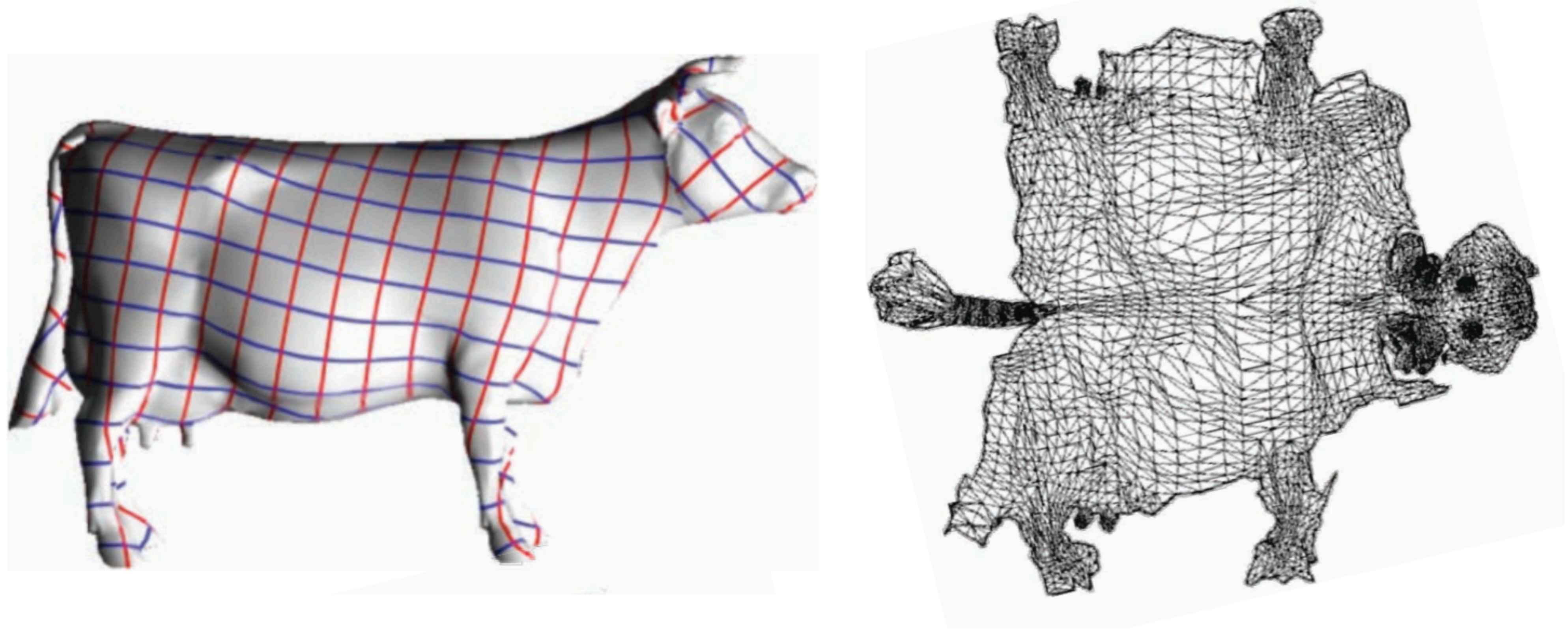
Problems

Current linear conformal methods have two major shortcomings:

NO BOUNDARY CONTROL

User obtains a single, automatic flattening

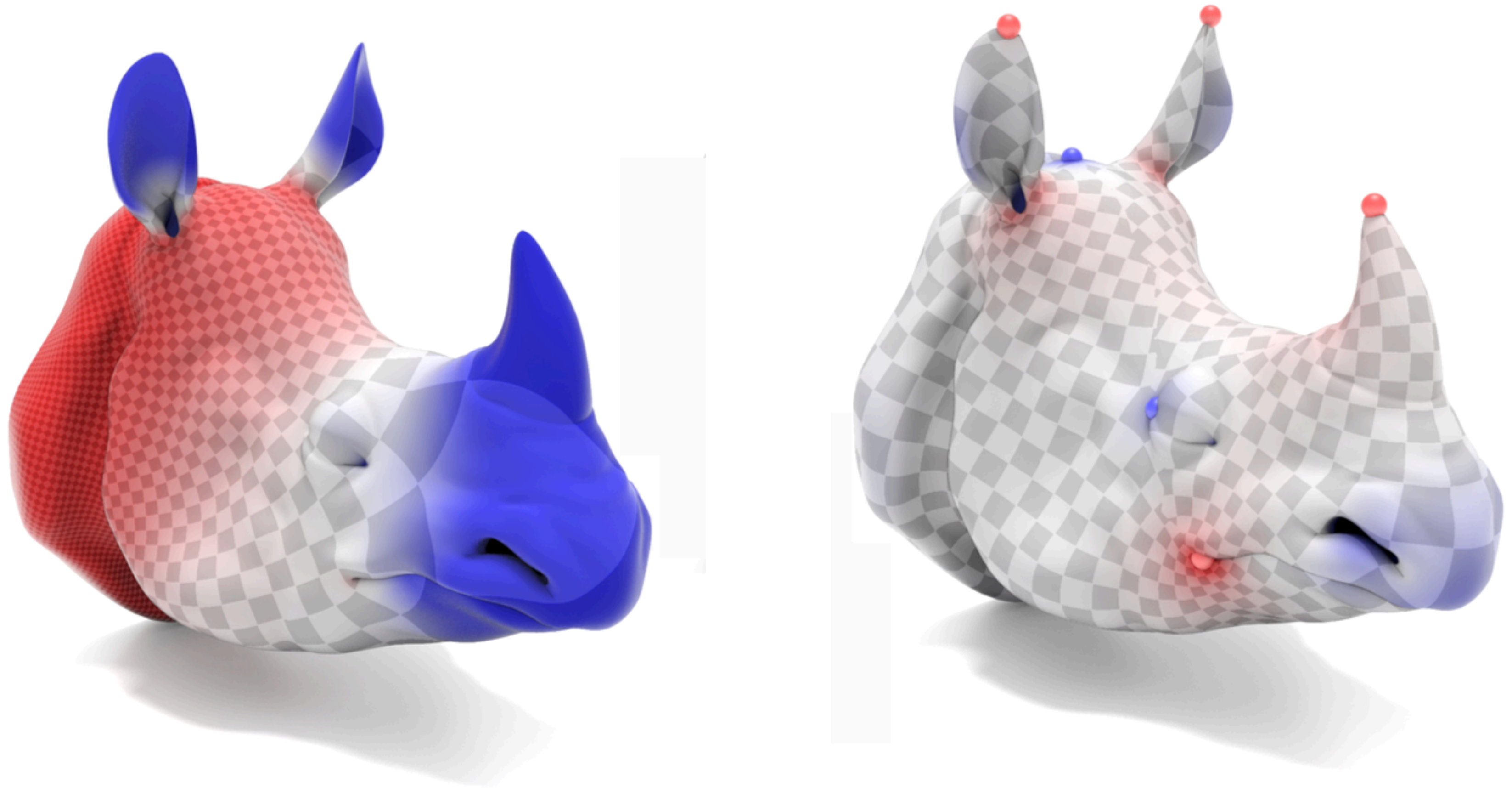
Must "take it or leave it," irrespective of quality



SCALE DISTORTION

Conformal maps can scale arbitrarily

Cone singularities mitigate scale distortion



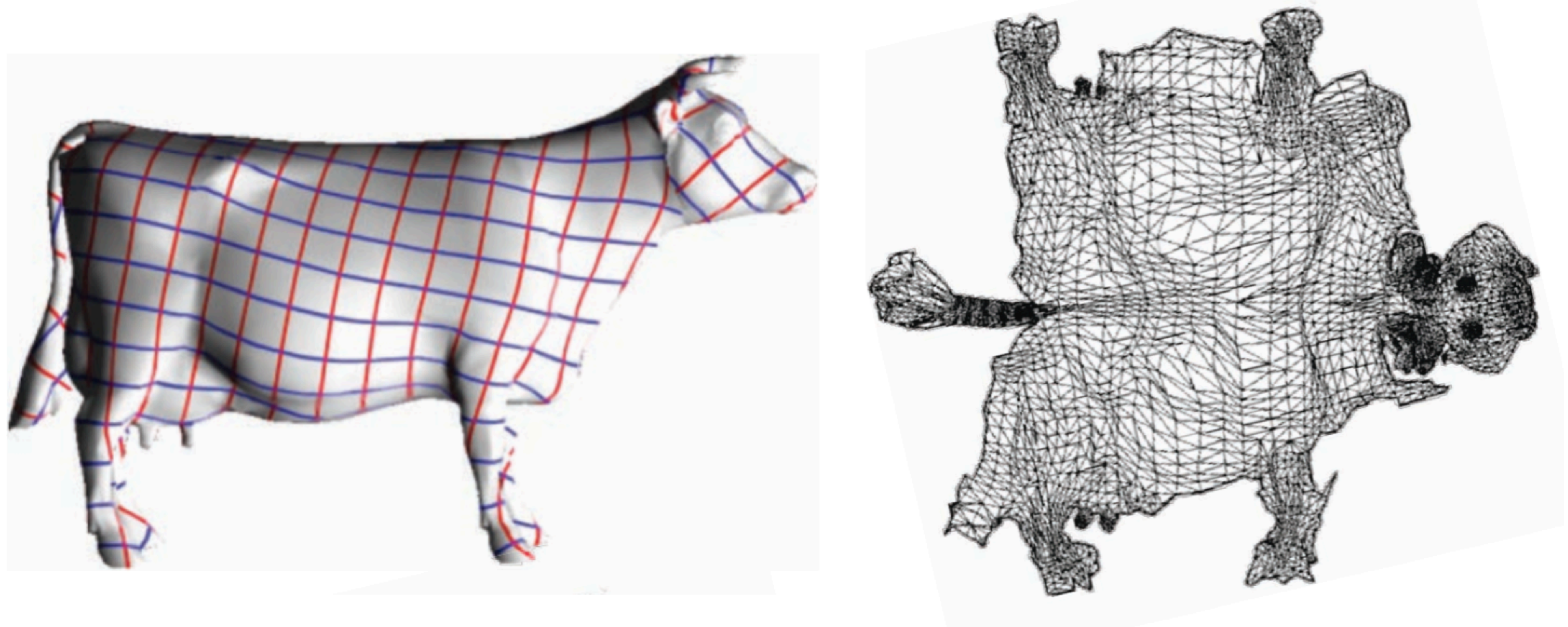
Problems

Current linear conformal methods have two major shortcomings:

NO BOUNDARY CONTROL

User obtains a single, automatic flattening

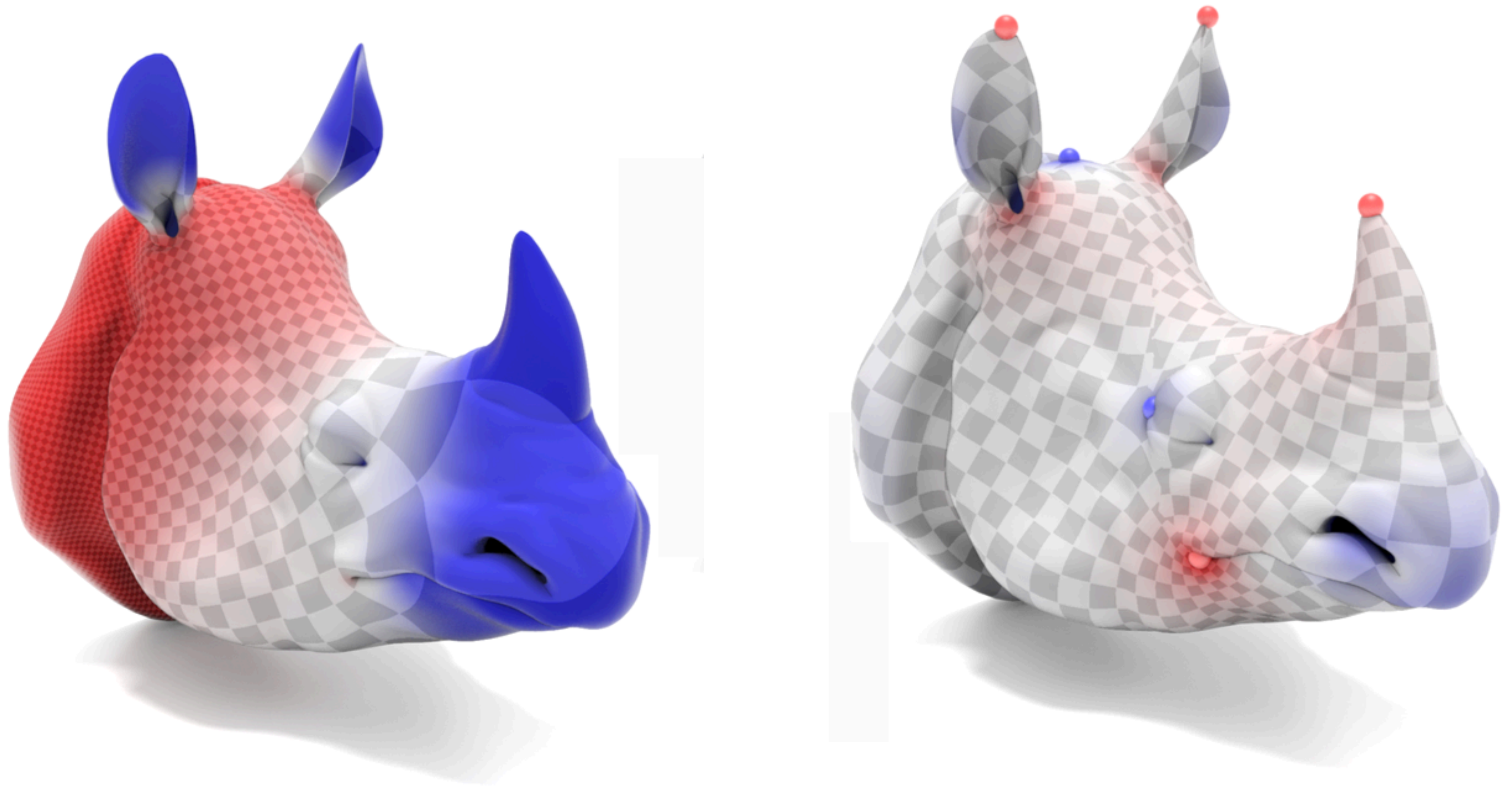
Must "take it or leave it,” irrespective of quality



SCALE DISTORTION

Conformal maps can scale arbitrarily

Cone singularities mitigate scale distortion



Optimal Cone Singularities for Conformal Flattening
[Soliman et al 2018]

RELATED WORK

“Free” Boundary Methods

ALGORITHM
Least Squares Conformal Maps (LSCM)
Spectral Conformal Parameterization (SCP)
Angle Based Flattening (ABF)
Linear Angle Based Flattening (LinABF)*

“Free” Boundary Methods

ALGORITHM	BOUNDARY CONTROL
Least Squares Conformal Maps (LSCM)	✗
Spectral Conformal Parameterization (SCP)	✗
Angle Based Flattening (ABF)	✗
Linear Angle Based Flattening (LinABF)*	✗

*can be modified to provide boundary control

“Free” Boundary Methods

ALGORITHM	BOUNDARY CONTROL
Least Squares Conformal Maps (LSCM)	✗
Spectral Conformal Parameterization (SCP)	✗
Angle Based Flattening (ABF)	✗
Linear Angle Based Flattening (LinABF)*	✗

Minimize discrete energy **without** explicit boundary constraints

*can be modified to provide boundary control

“Free” Boundary Conditions are Meaningless

Solution has no meaningful interpretation in the smooth setting!

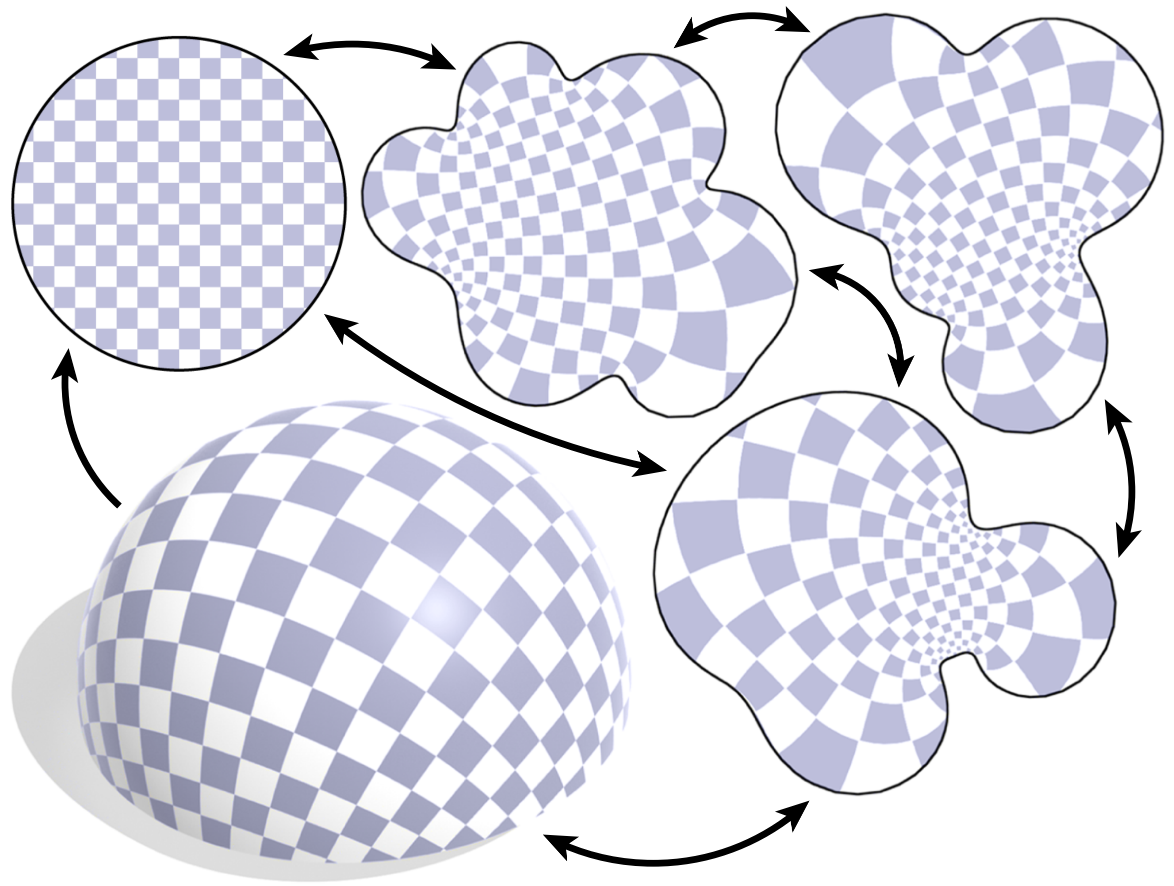
“Free” Boundary Conditions are Meaningless

Solution has no meaningful interpretation in the smooth setting!

SMOOTH SETTING

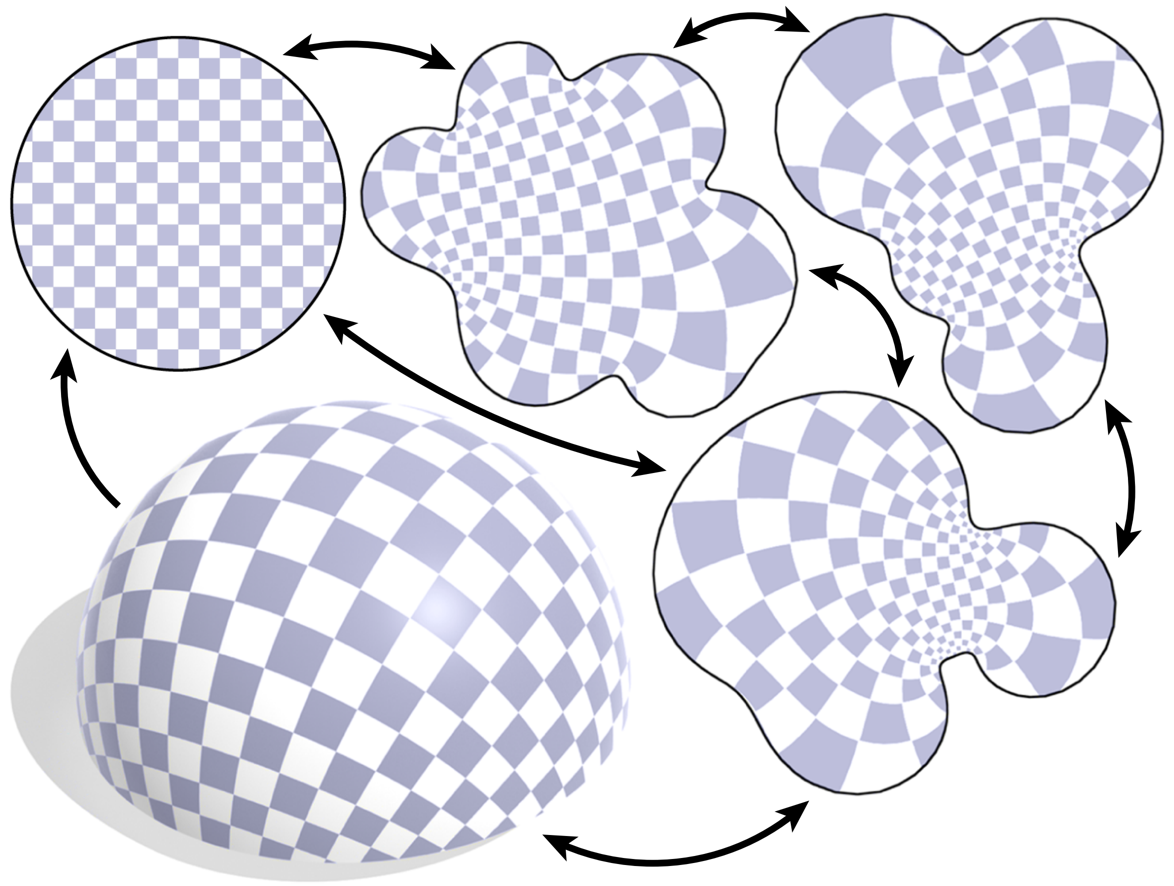
Enormous space of perfect conformal flattenings

Obtained by flattening and then applying in-plane maps



“Free” Boundary Conditions are Meaningless

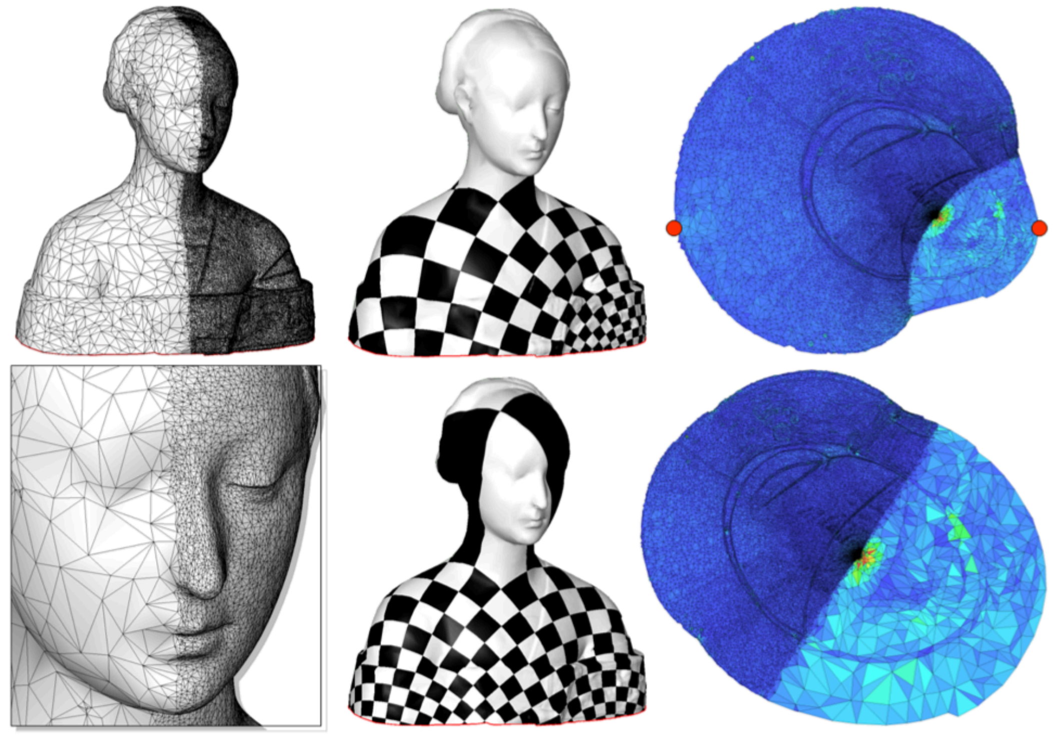
Solution has no meaningful interpretation in the smooth setting!



SMOOTH SETTING

Enormous space of perfect conformal flattenings

Obtained by flattening and then applying in-plane maps



DISCRETE SETTING

Unique solution **must** depend on discretization

Results change based on mesh or numerical treatment

Special Conformal Parameterization
[Mullen et al 2008]

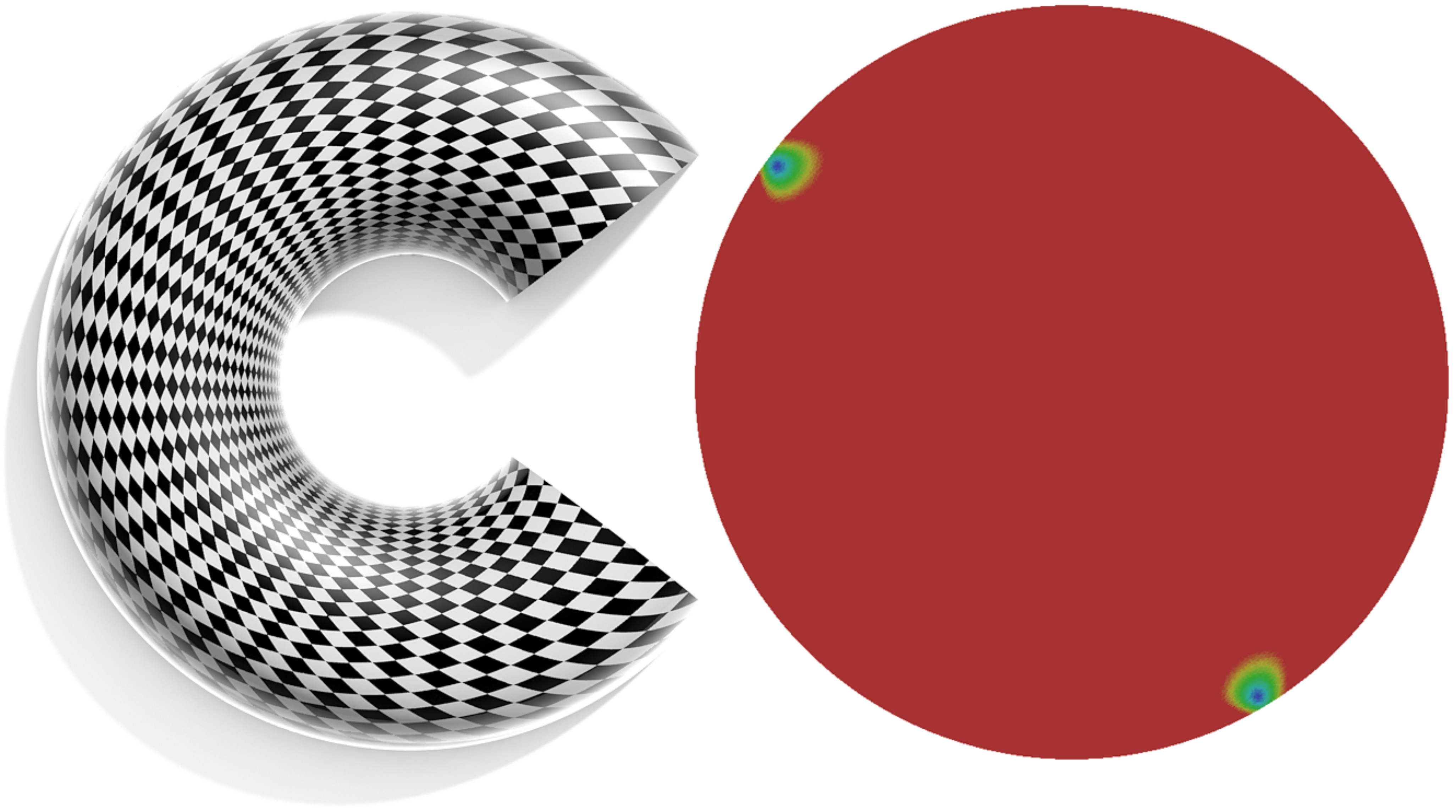
Forcing the Boundary

First attempt: Pin all boundary points to get desired shape

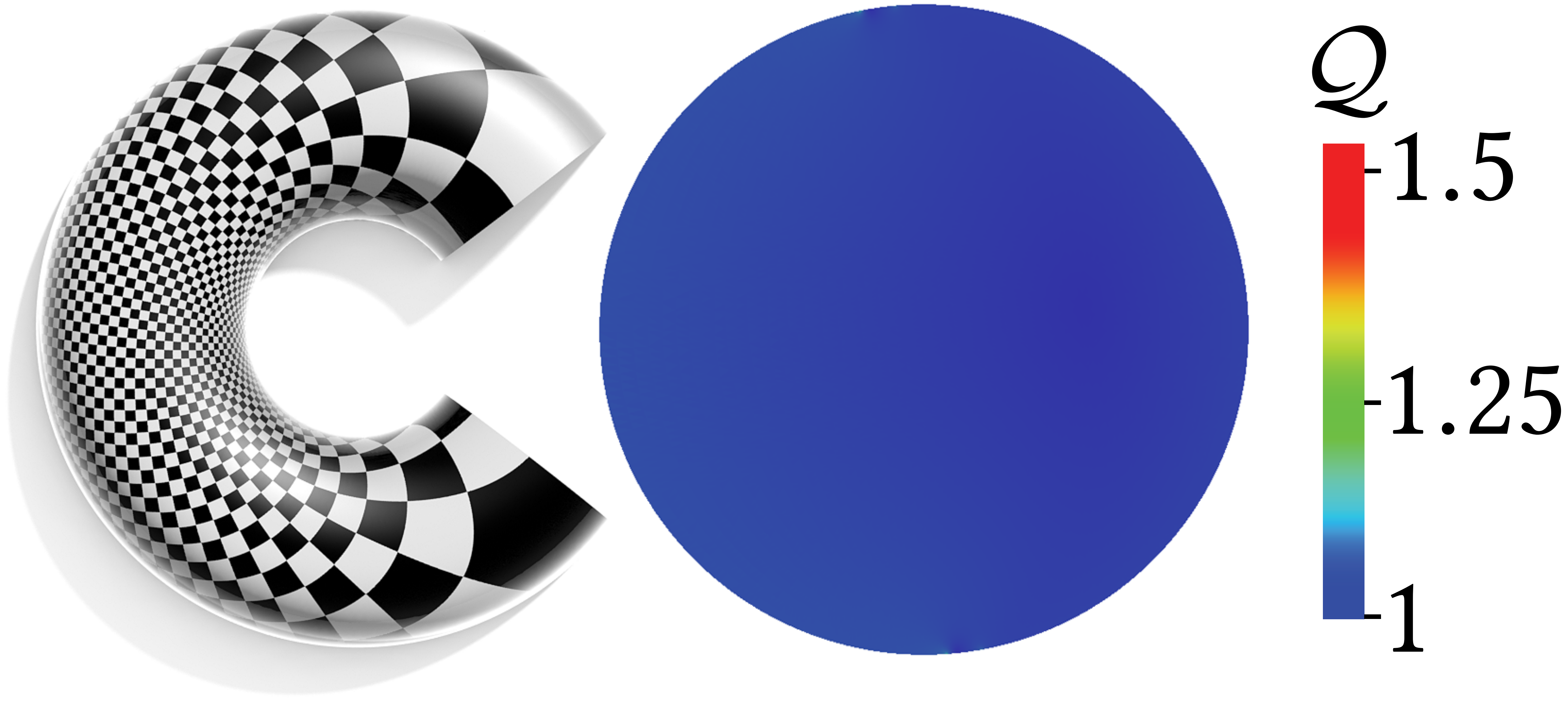
Forcing the Boundary

First attempt: Pin all boundary points to get desired shape

Least squares yields harmonic map with **severe angle distortion**:



Harmonic



Conformal

Prescribed Lengths and Angles

ALGORITHM	COMPLEXITY	BOUNDARY CONTROL
Circle Patterns (CP)	Nonlinear	✗ (only angles)
Conformal Equivalence (CETM)	Nonlinear	✓
Curvature Prescription (CPMS)*	Linear	✗
Boundary First Flattening (BFF)	Linear	✓

*can be modified to provide boundary control

Prescribed Lengths and Angles

ALGORITHM	COMPLEXITY	BOUNDARY CONTROL
Circle Patterns (CP)	Nonlinear	✗ (only angles)
Conformal Equivalence (CETM)	Nonlinear	✓
Curvature Prescription (CPMS)*	Linear	✗
Boundary First Flattening (BFF)	Linear	✓

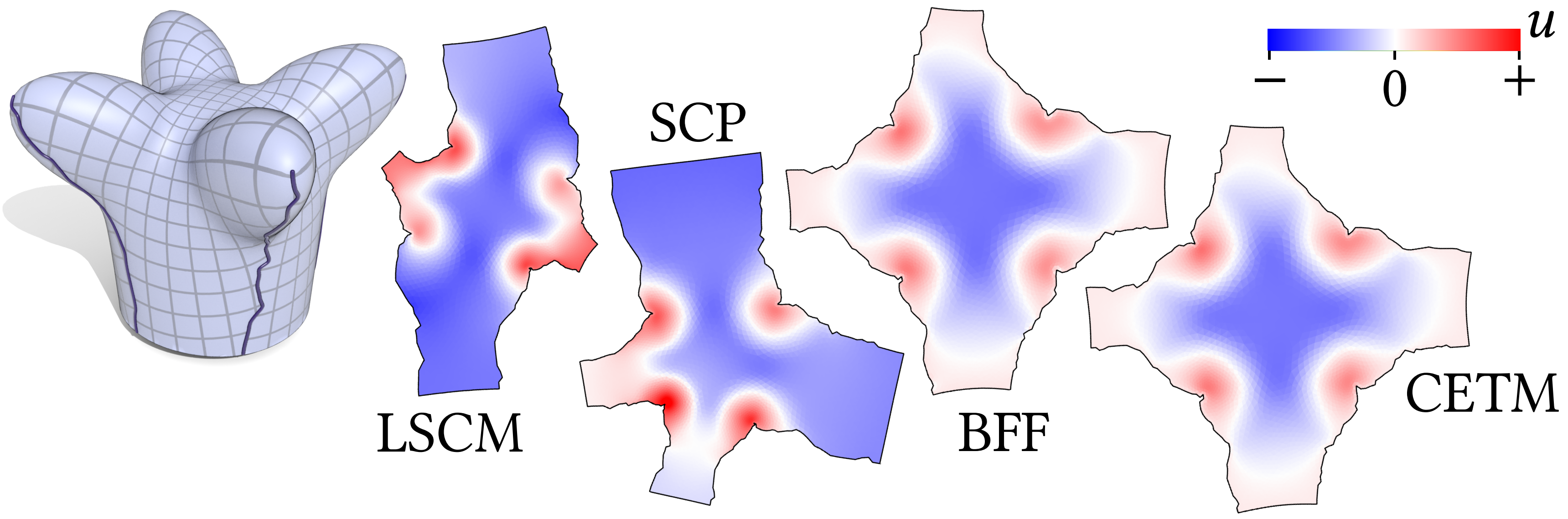
*can be modified to provide boundary control

Prescribed Lengths and Angles

ALGORITHM	COMPLEXITY	BOUNDARY CONTROL
Circle Patterns (CP)	Nonlinear	✗ (only angles)
Conformal Equivalence (CETM)	Nonlinear	✓
Curvature Prescription (CPMS)*	Linear	✗
Boundary First Flattening (BFF)	Linear	✓

BFF is faster than existing linear methods

Quality & control comparable to nonlinear schemes

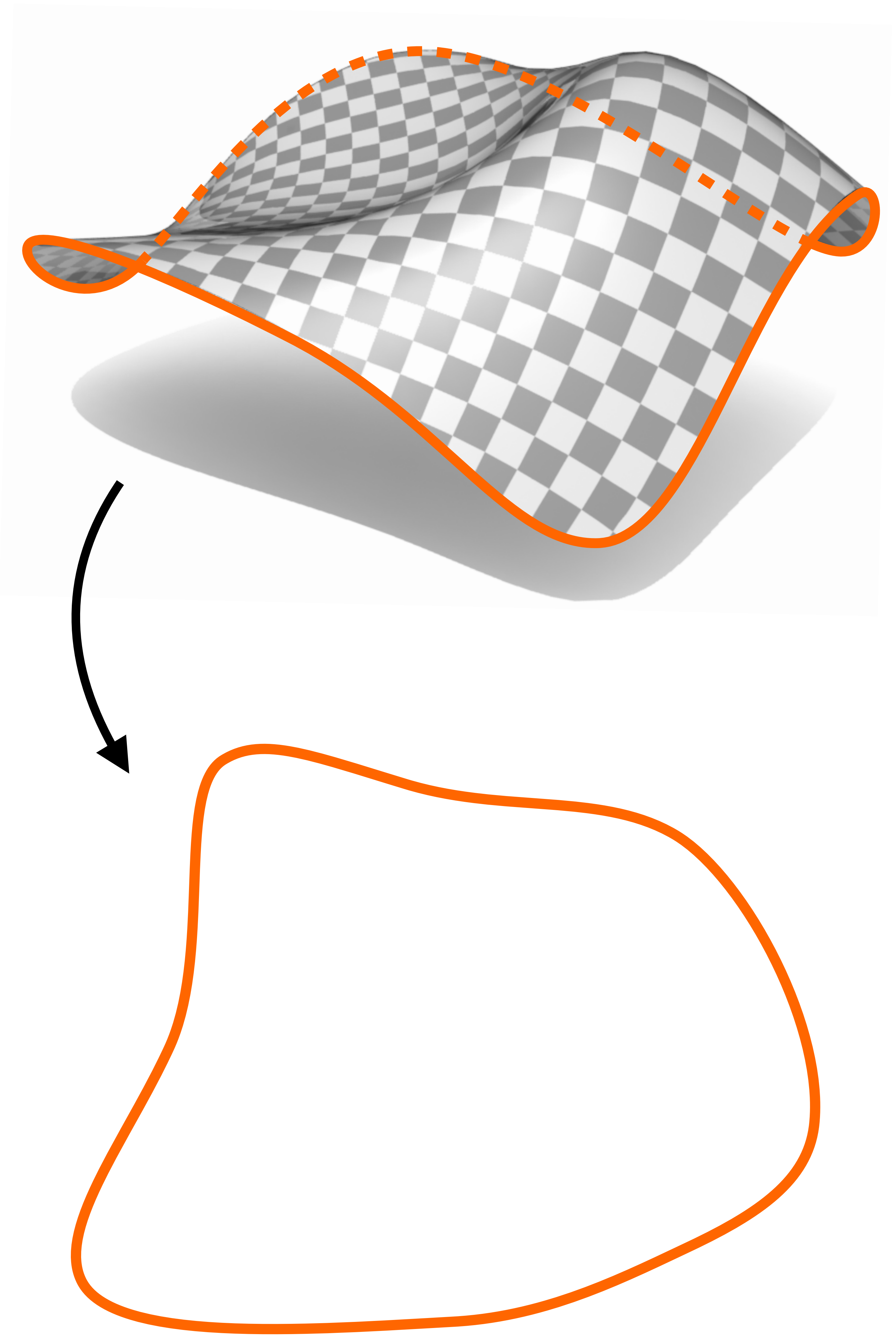


*can be modified to provide boundary control

SMOOTH THEORY

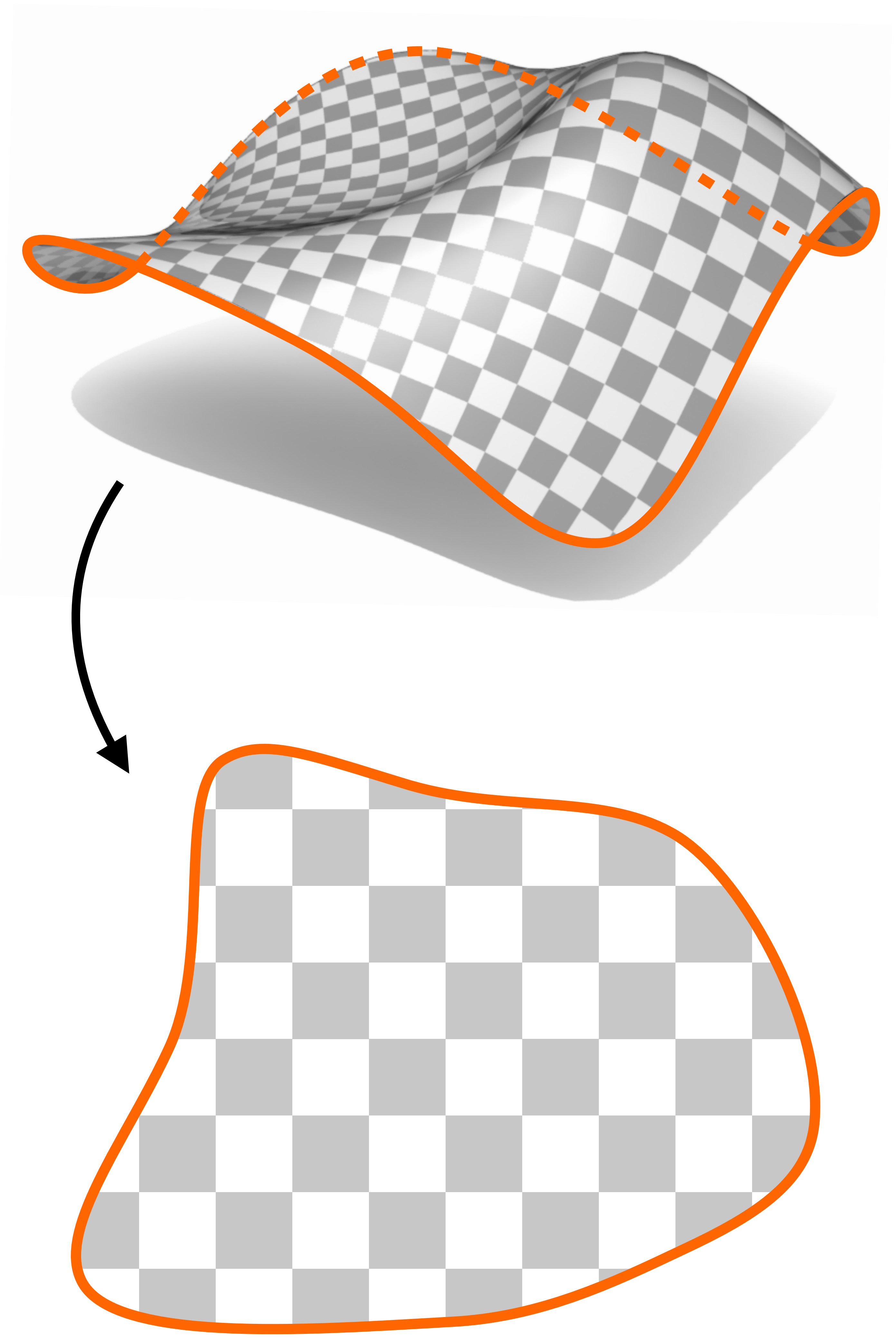
Key Idea

If you know the boundary of a conformal map,

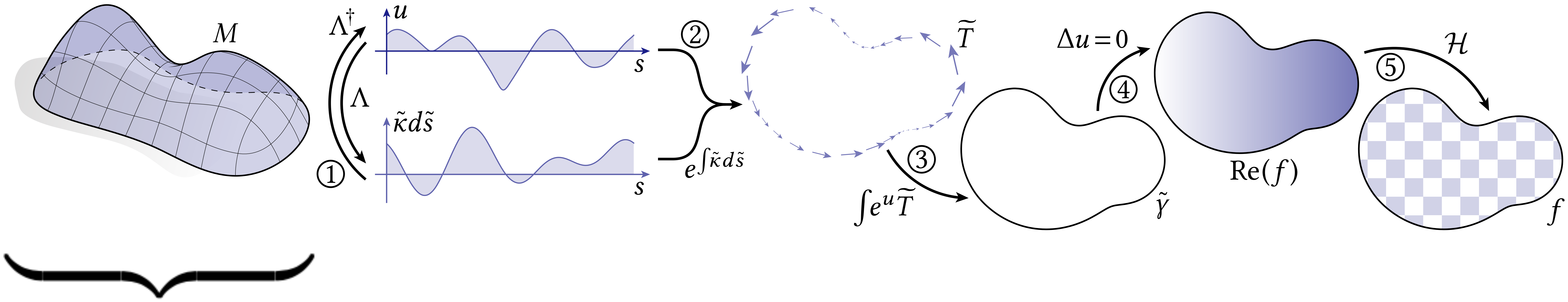


Key Idea

If you know the boundary of a conformal map, then extension to the interior is easy

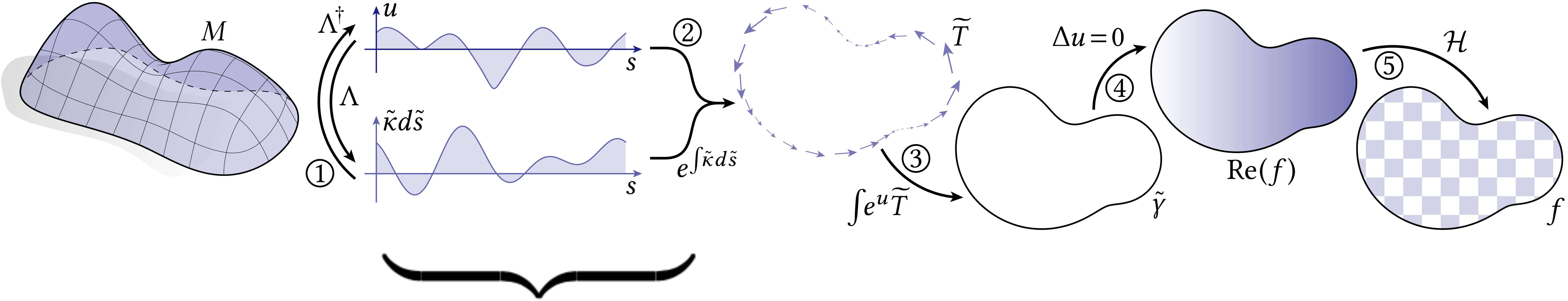


Algorithm Outline



Given a surface with either scale **or** curvature of target boundary curve

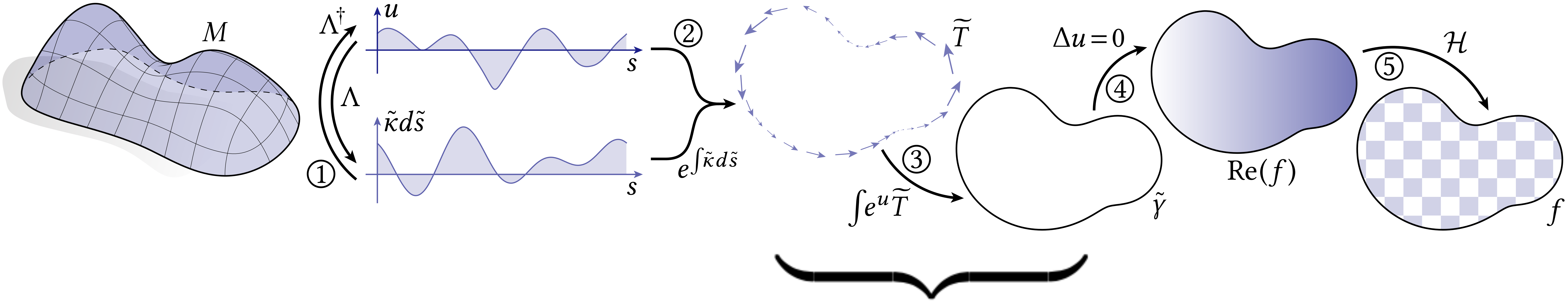
Algorithm Outline



Given a surface with either scale or curvature of target boundary curve

1. Solve *Yamabe Problem* to get complementary data (curvature or scale)

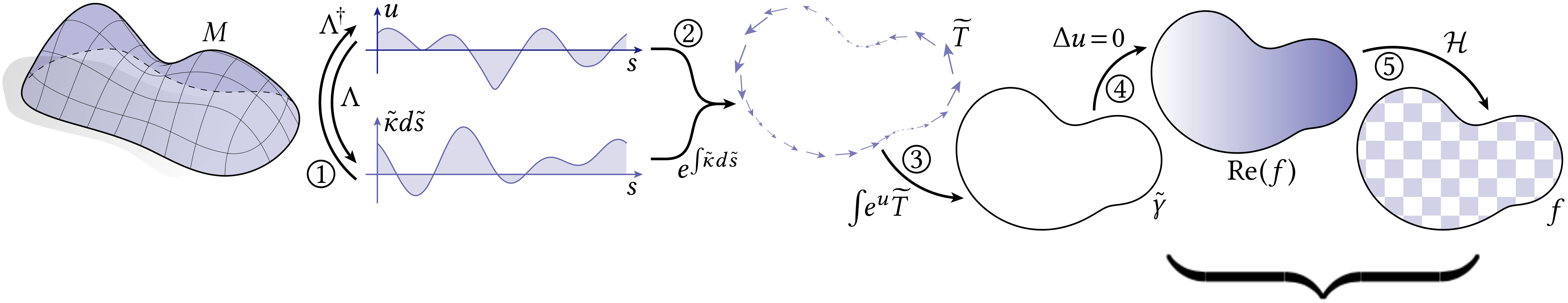
Algorithm Outline



Given a surface with either scale **or** curvature of target boundary curve

1. Solve *Yamabe Problem* to get complementary data (curvature or scale)
2. Integrate boundary data to get boundary curve

Algorithm Outline



Given a surface with either scale or curvature of target boundary curve

1. Solve *Yamabe Problem* to get complementary data (curvature or scale)
2. Integrate boundary data to get boundary curve
3. Extend boundary curve to a pair of *conjugate harmonic functions*

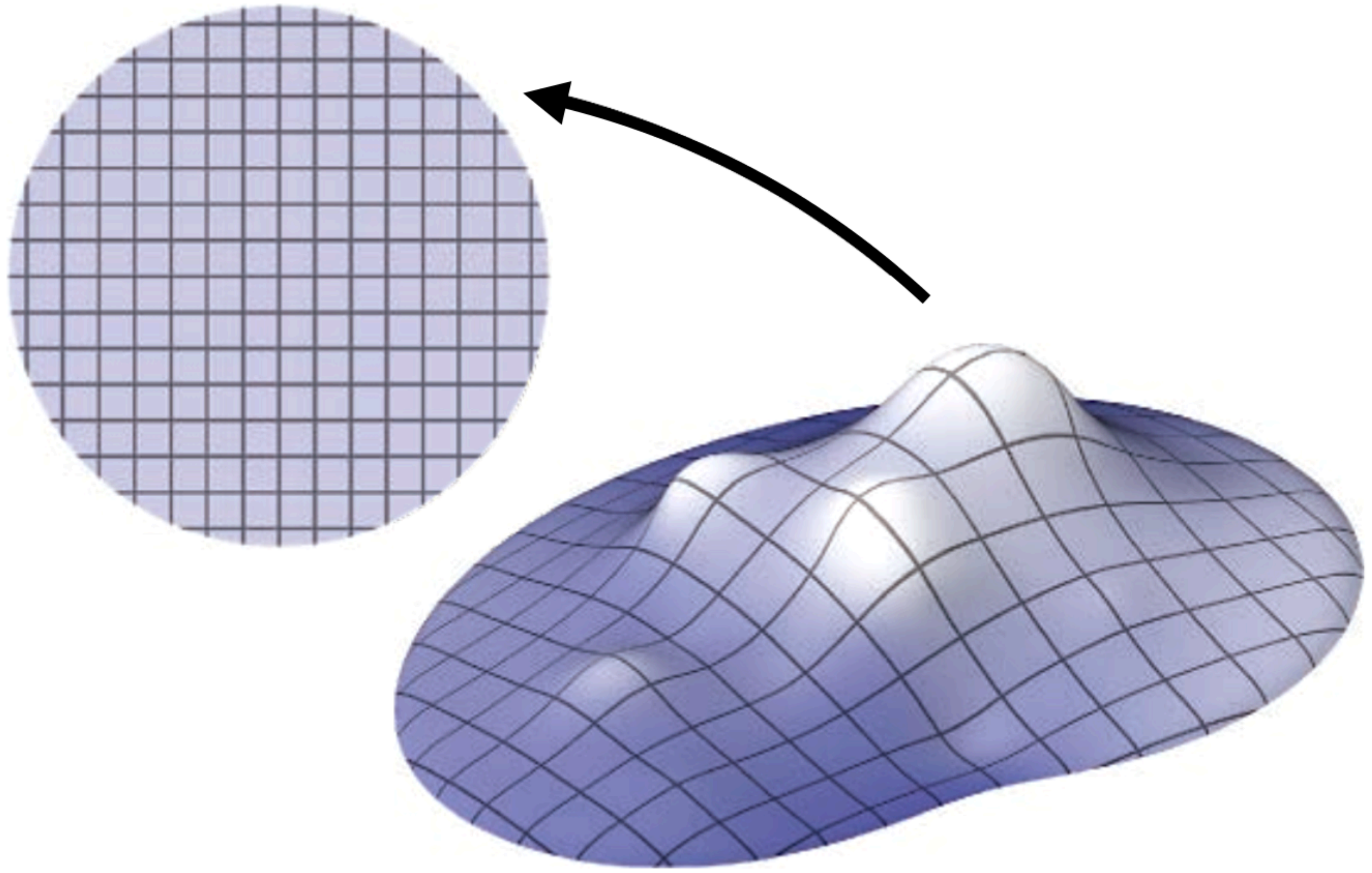
Compatibility of Boundary Data

Not every parameterized curve is
the boundary of a conformal map!

Yamabe Problem

Yamabe equation provides explicit relationship between conformal scaling and change in curvature:

$$\Delta u = K - e^{2u}\tilde{K} \qquad on\ M$$

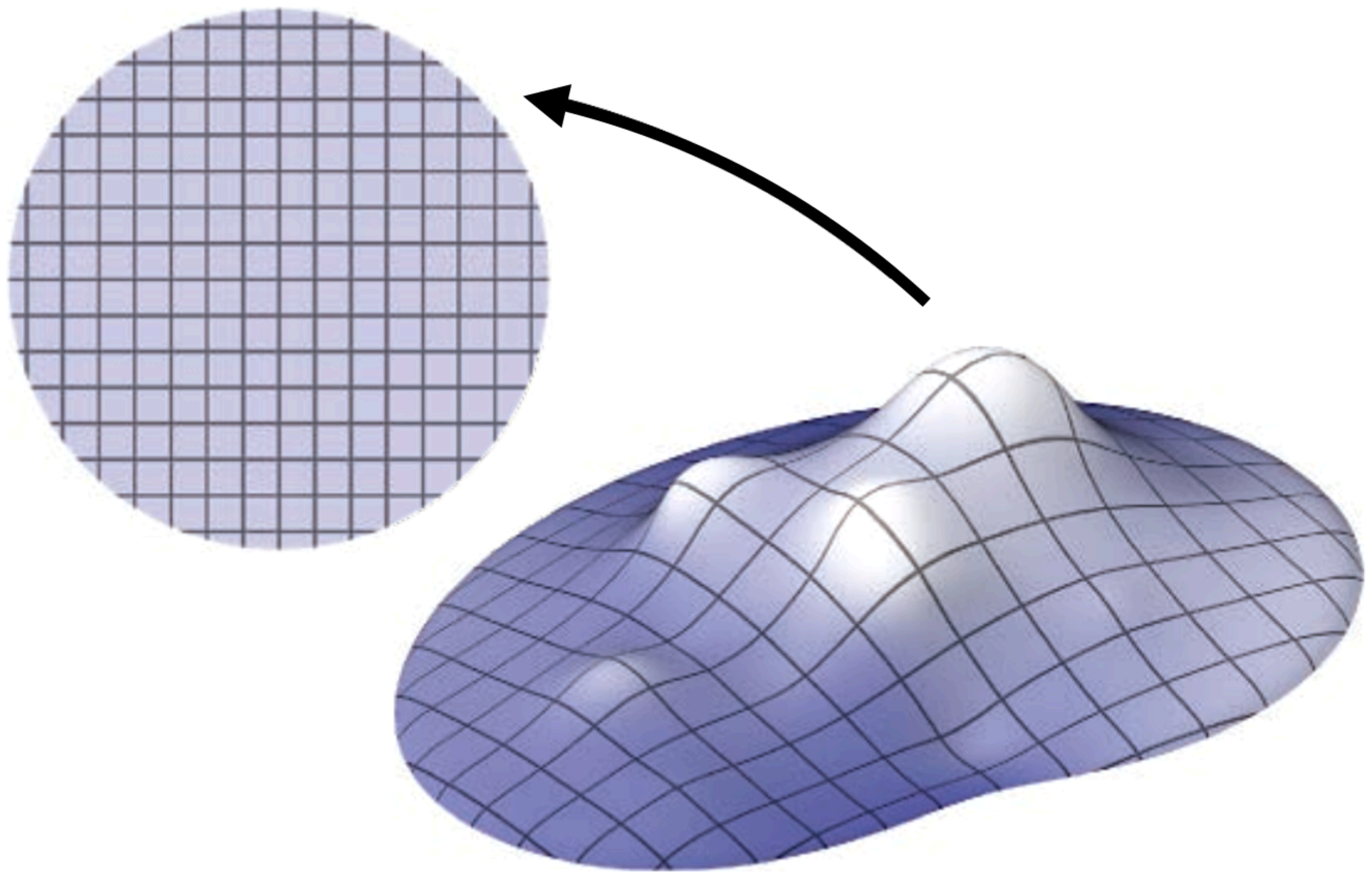


Yamabe Problem

Yamabe equation provides explicit relationship between conformal scaling and change in curvature:

log scale factor

$$\Delta u = K - e^{2u}\tilde{K} \qquad on\ M$$



Yamabe Problem

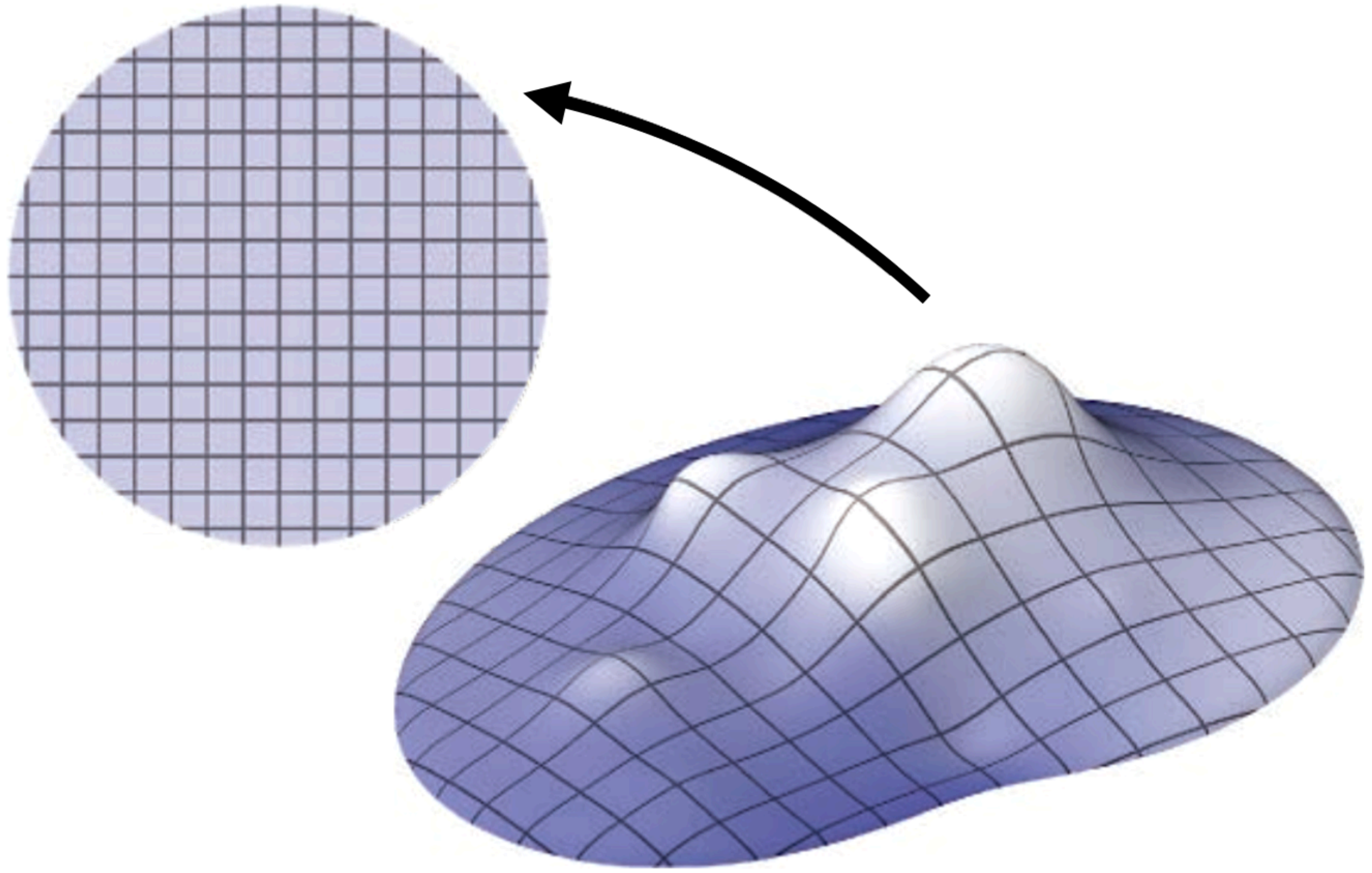
Yamabe equation provides explicit relationship between conformal scaling and change in curvature:

log scale factor

$$\Delta u = K - e^{2u}\tilde{K} \qquad on\ M$$

original
Gaussian curvature

new
Gaussian curvature



Yamabe Problem

Yamabe equation provides explicit relationship between conformal scaling and change in curvature:

log scale factor

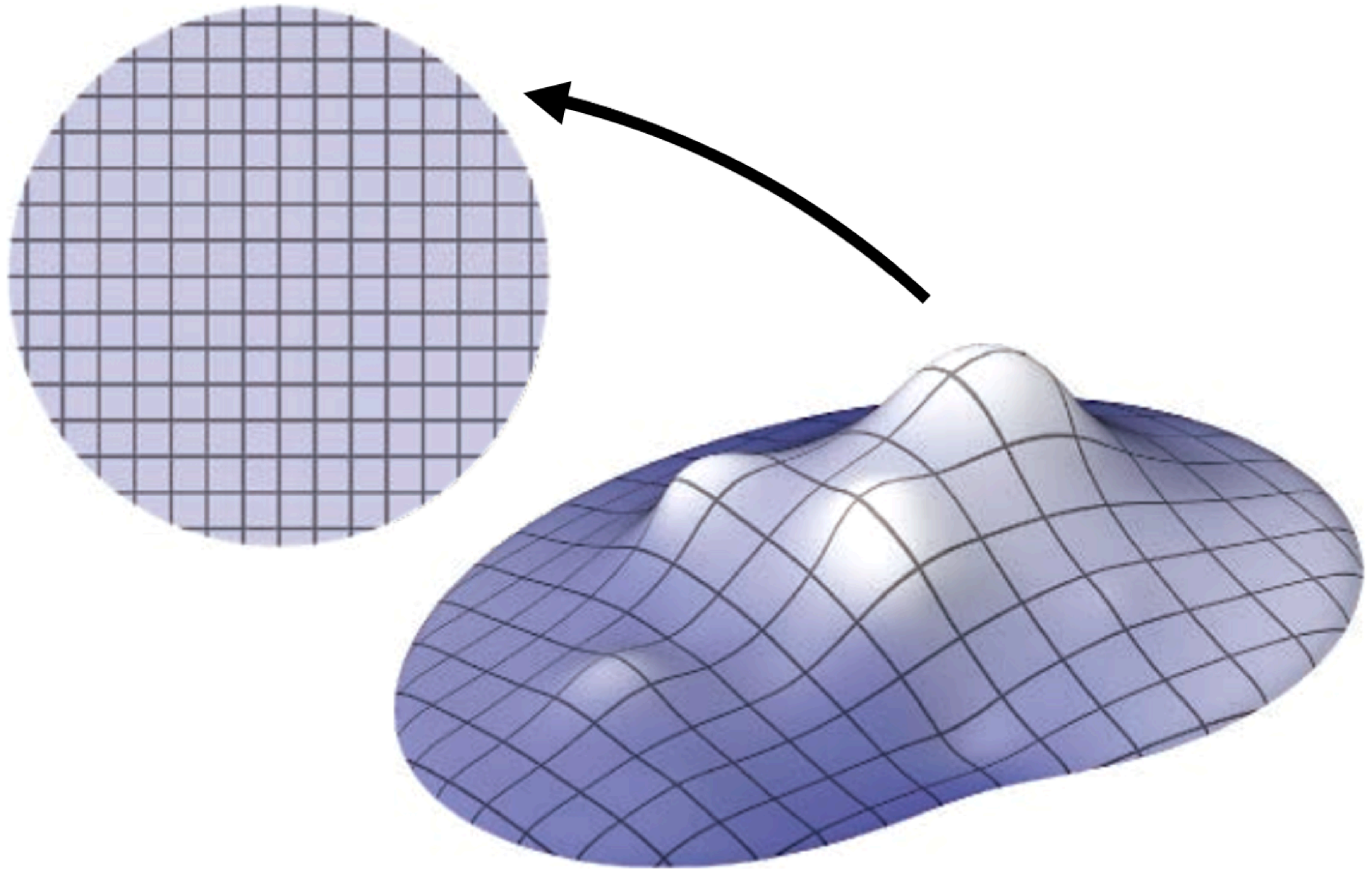
$$\Delta u = K - e^{2u}\tilde{K} \qquad on\ M$$

original
Gaussian curvature

new
Gaussian curvature

Cherrier boundary conditions:

$$\frac{\partial u}{\partial n} = \kappa - e^{u}\tilde{\kappa} \qquad on\ \partial M$$



Yamabe Problem

Yamabe equation provides explicit relationship between conformal scaling and change in curvature:

log scale factor

$$\Delta u = K - e^{2u}\tilde{K} \qquad \text{on } M$$

original
Gaussian curvature

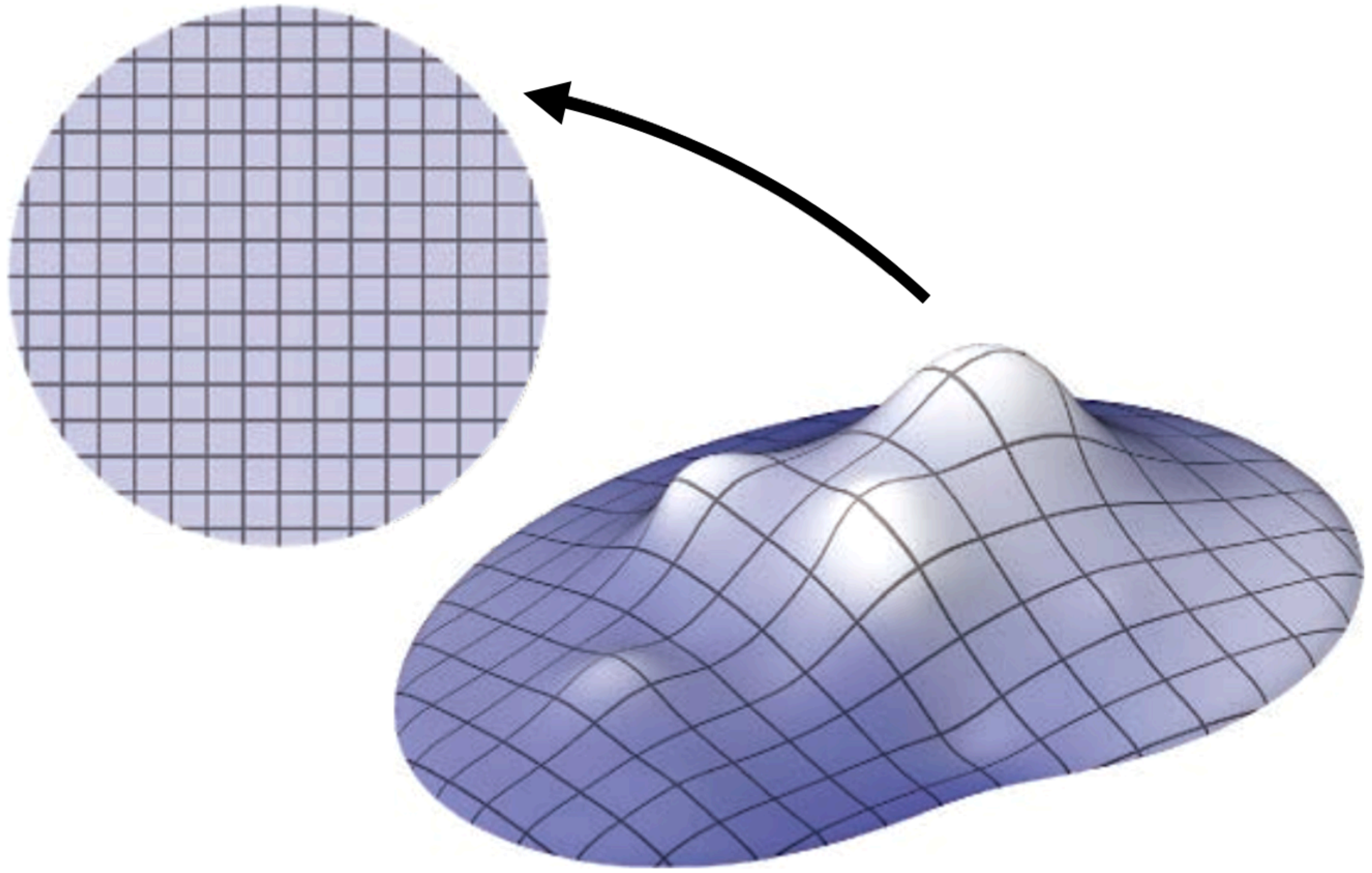
new
Gaussian curvature

Cherrier boundary conditions:

$$\frac{\partial u}{\partial n} = \kappa - e^{u}\tilde{\kappa} \qquad \text{on } \partial M$$

original
geodesic curvature

new
geodesic curvature



Yamabe Problem

Yamabe equation provides explicit relationship between conformal scaling and change in curvature:

log scale factor

$$\Delta u = K - e^{2u}\tilde{K} \qquad on\ M$$

original
Gaussian curvature

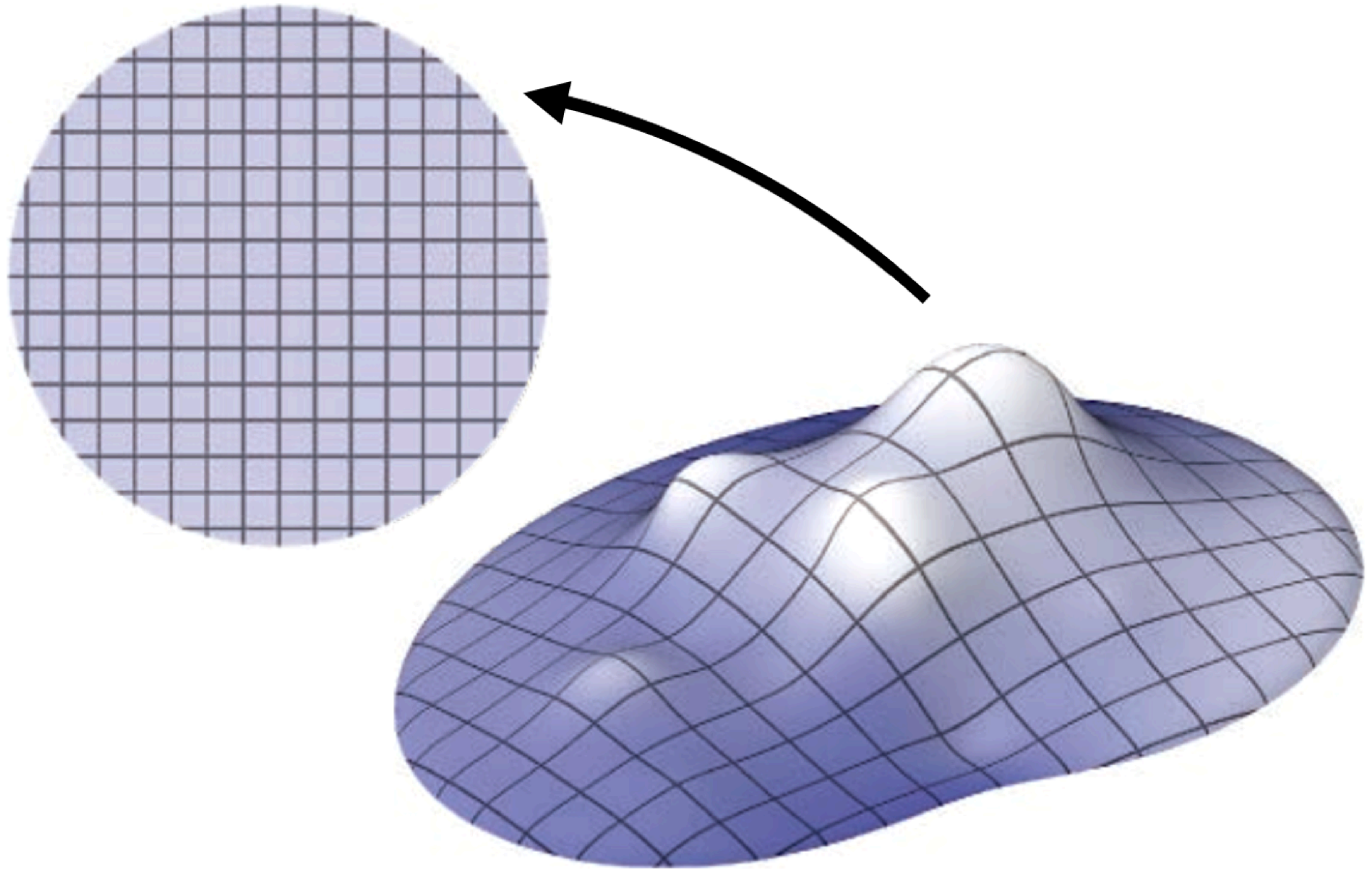
new
Gaussian curvature

Cherrier boundary conditions:

$$\frac{\partial u}{\partial n} = \kappa - e^{u}\tilde{\kappa} \qquad on\ \partial M$$

original
geodesic curvature

new
geodesic curvature



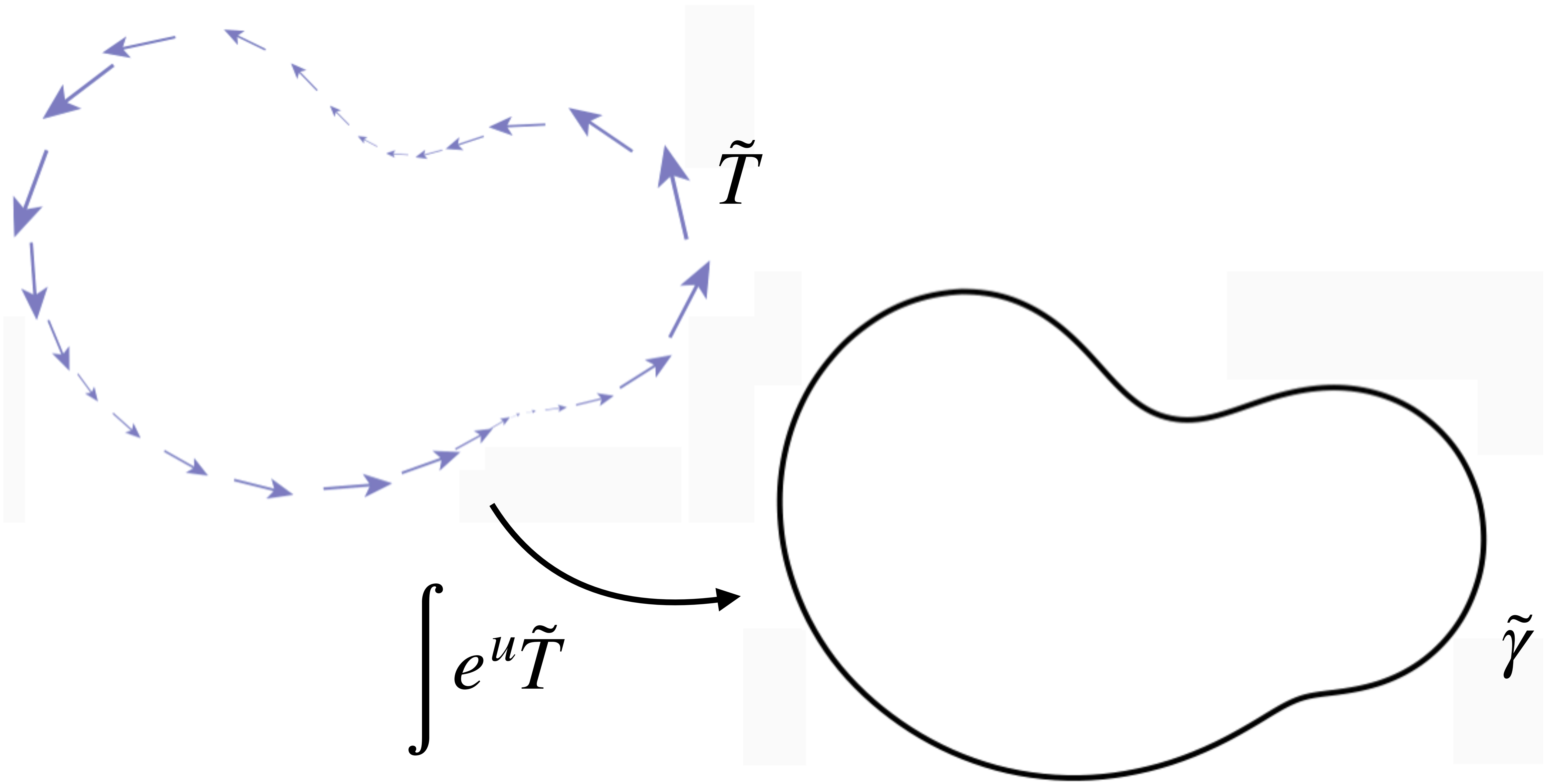
Can prescribe either curvature or scaling, **but not both!**

Curve Integration

Curvature and scaling determine a closed curve up to rigid transformation

Curve Integration

Curvature and scaling determine a closed curve up to rigid transformation

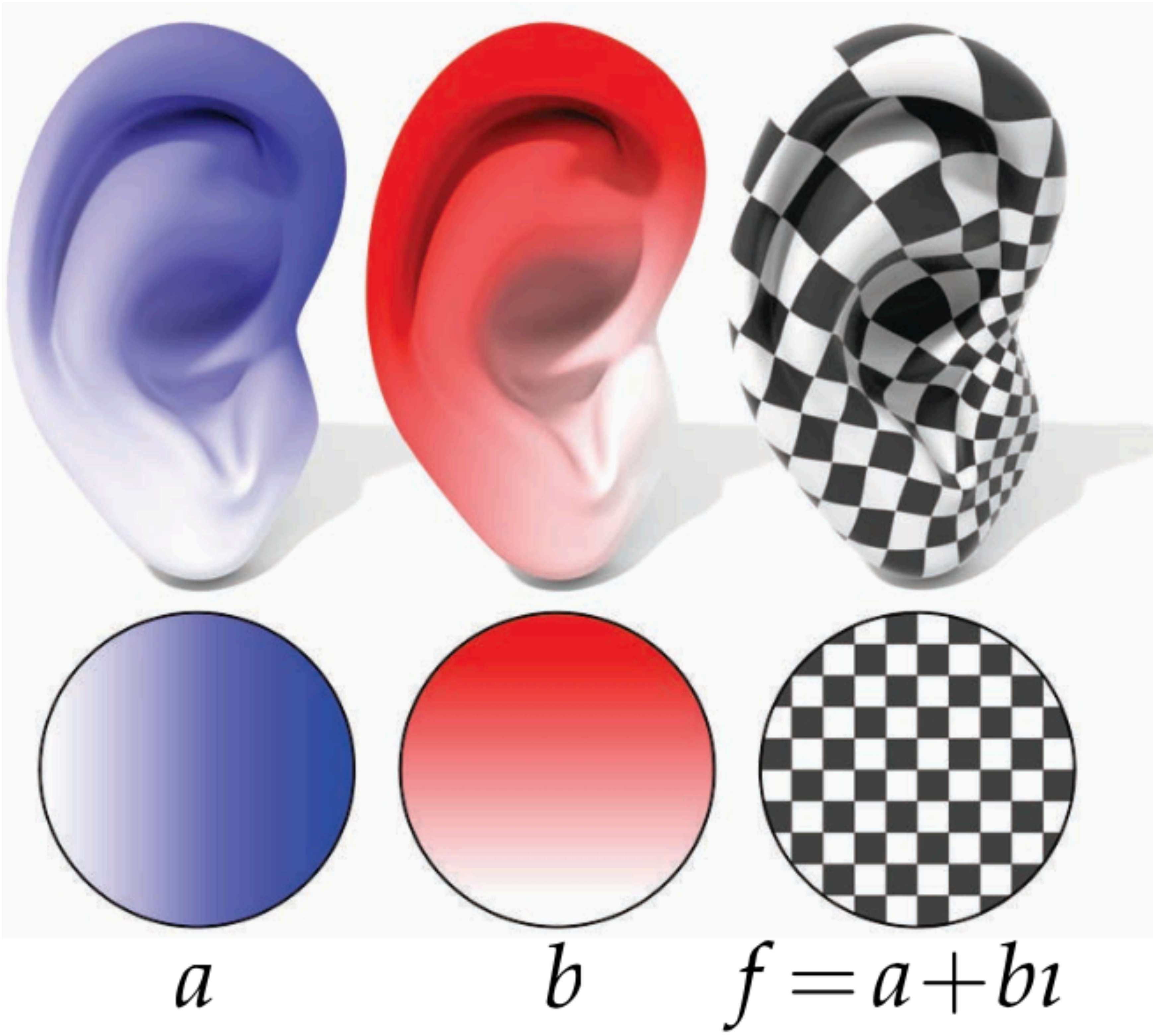


Conjugate Harmonic Functions

How do we find the solution on the interior?

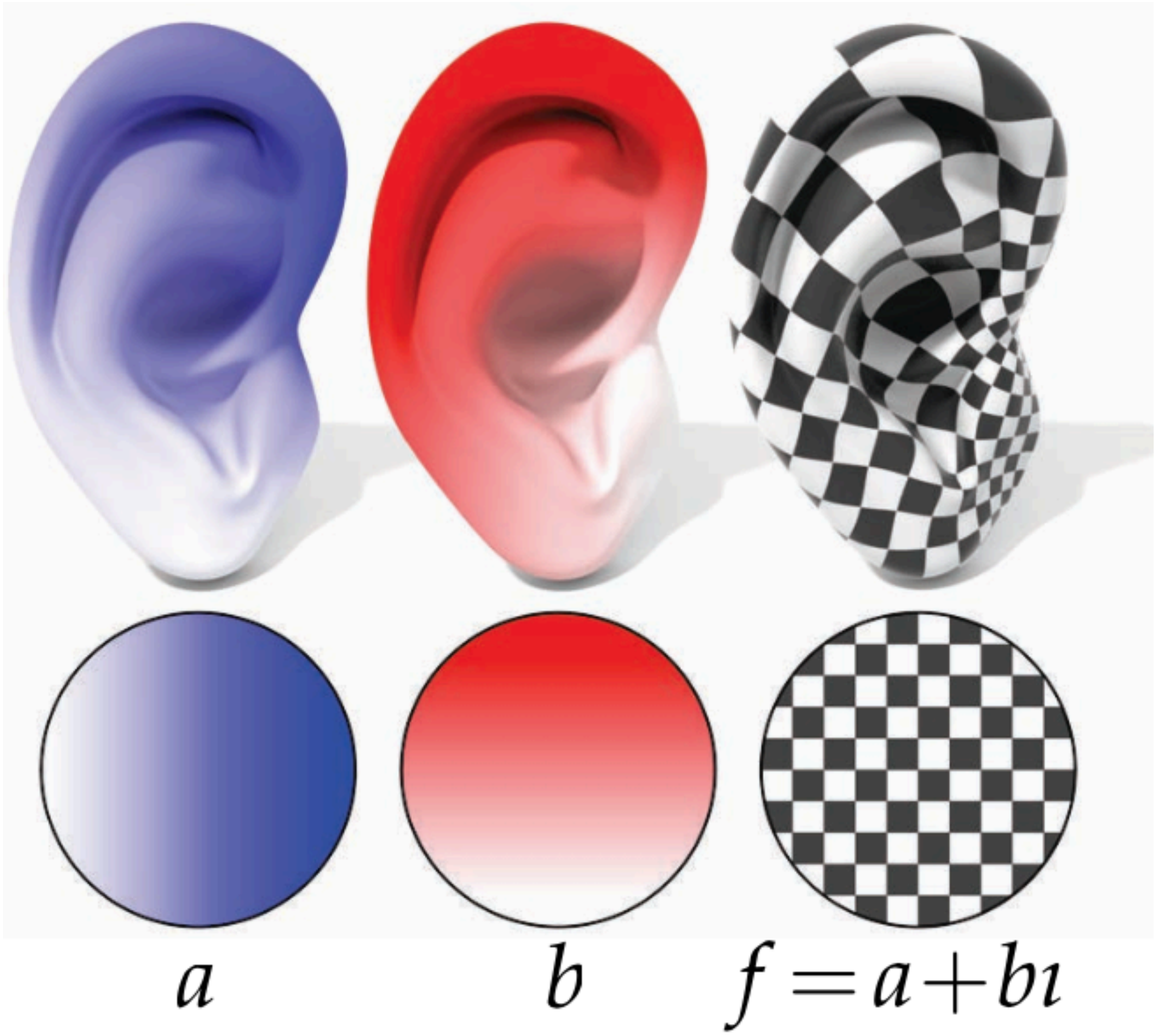
Conjugate Harmonic Functions

How do we find the solution on the interior?



Conjugate Harmonic Functions

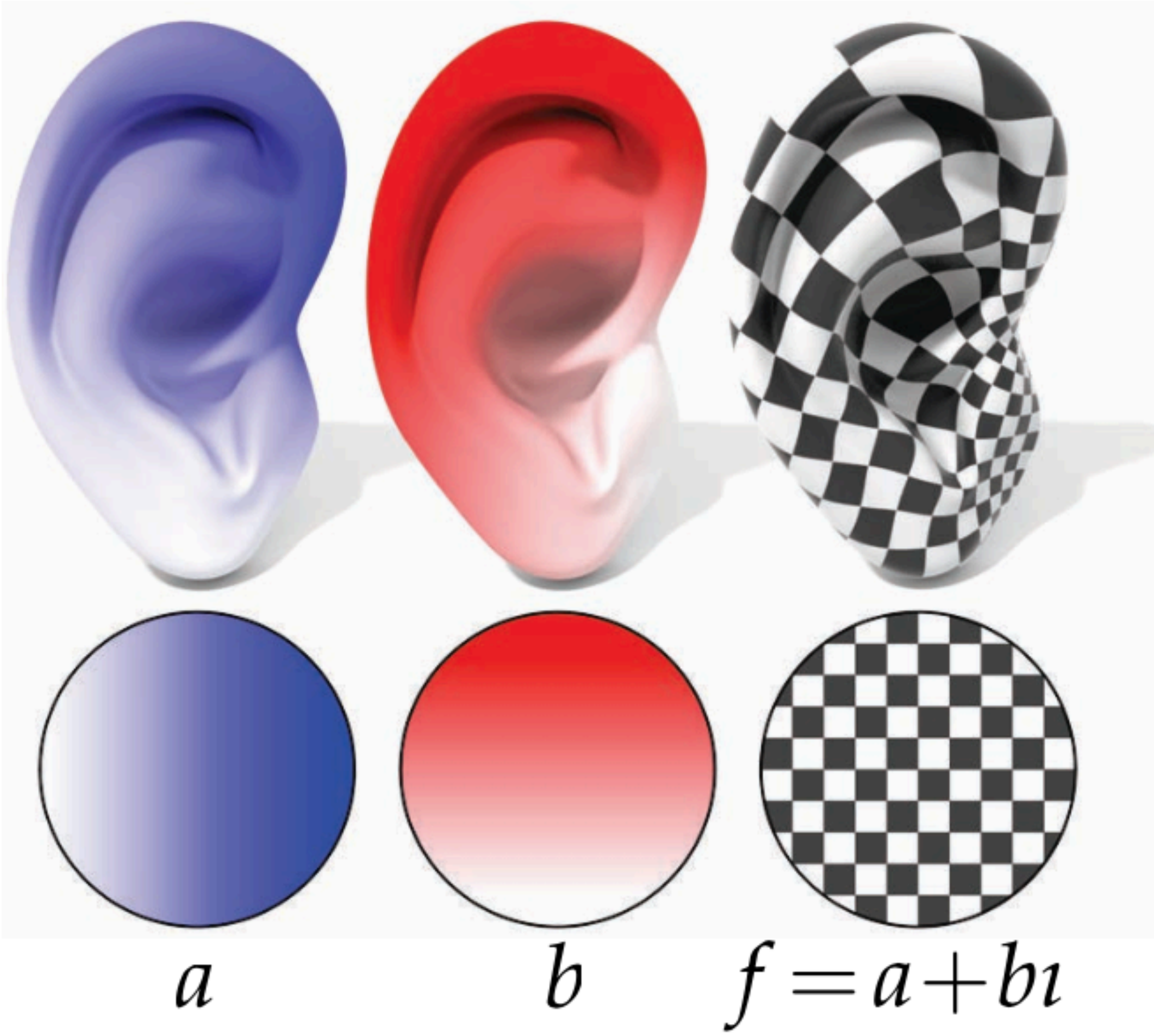
How do we find the solution on the interior?



$$J \nabla a = \nabla b$$

Conjugate Harmonic Functions

How do we find the solution on the interior?



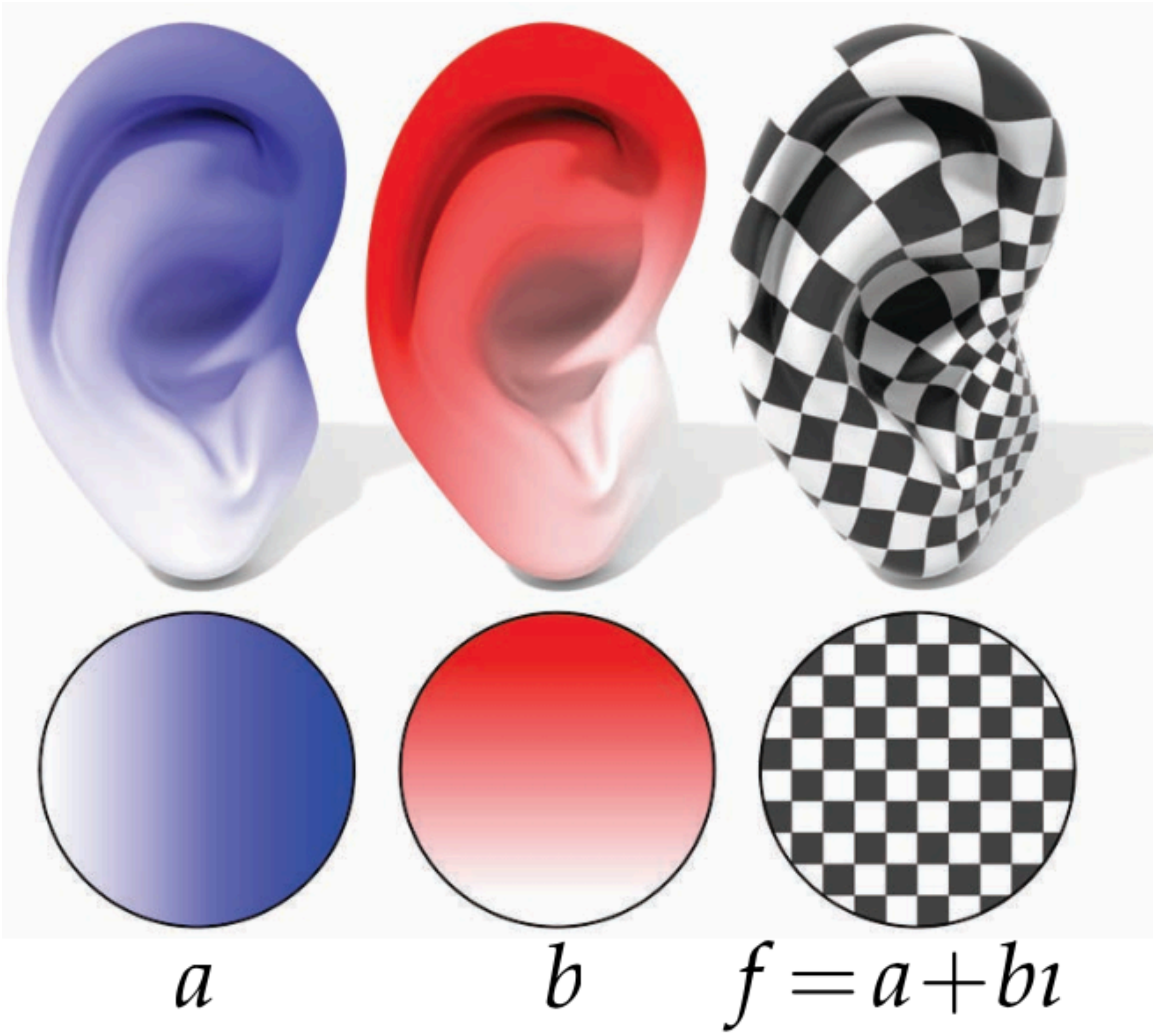
$$J \nabla a = \nabla b$$

$$\Delta a = 0$$

$$\Delta b = 0$$

Conjugate Harmonic Functions

How do we find the solution on the interior?

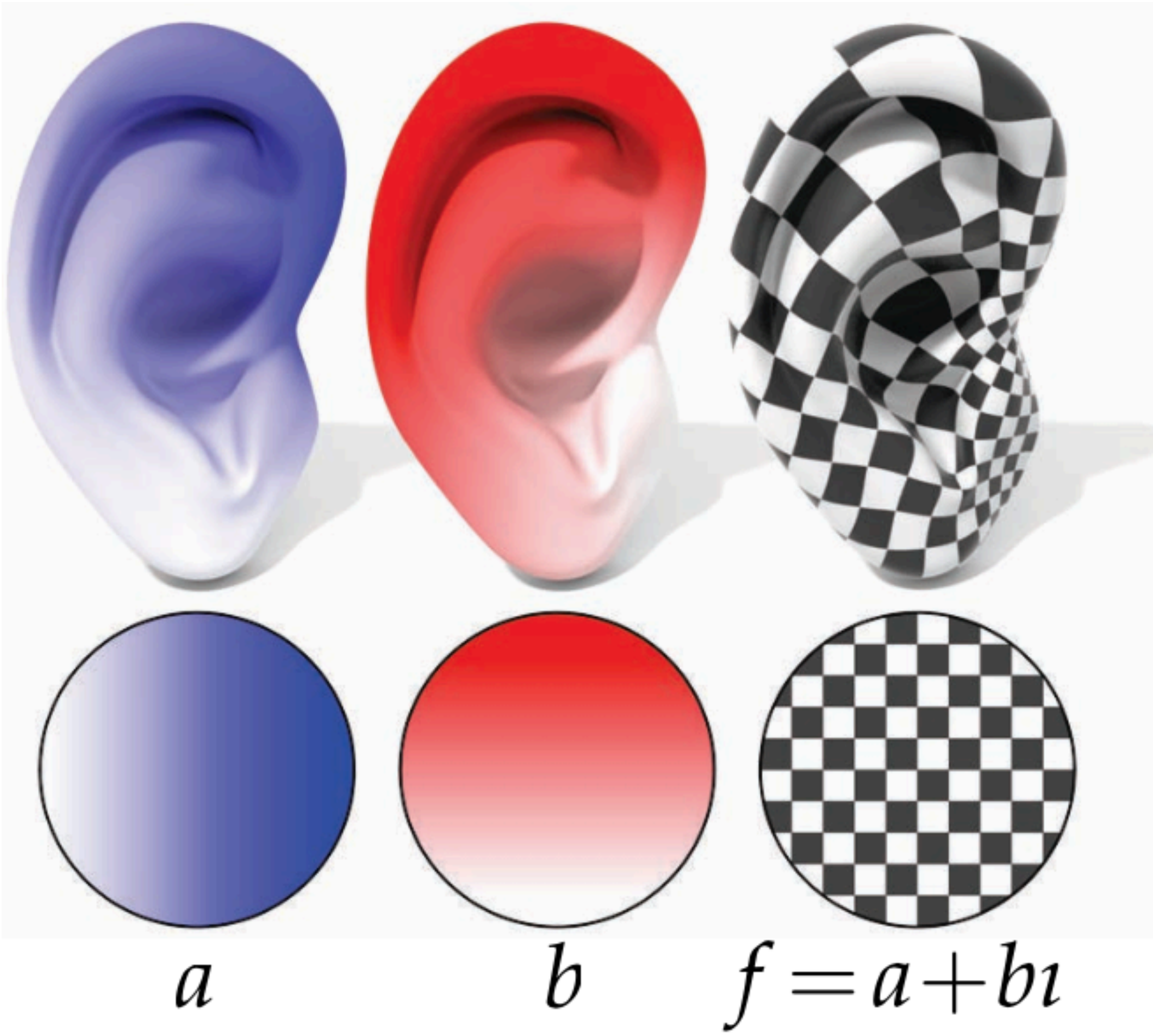


$$\boxed{\begin{gathered} J\nabla a = \nabla b \\ \Delta a = 0 \\ \Delta b = 0 \end{gathered}}$$

CONJUGATE HARMONIC PAIR

Conjugate Harmonic Functions

How do we find the solution on the interior?

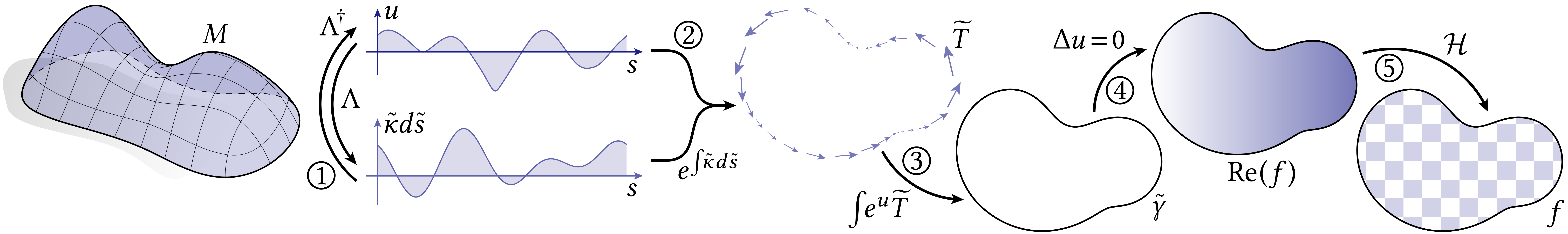


$$\boxed{\begin{gathered} J\nabla a = \nabla b \\ \Delta a = 0 \\ \Delta b = 0 \end{gathered}}$$

CONJUGATE HARMONIC PAIR

Fix ***a*** along the boundary and minimize conformal energy w.r.t. ***b*** (easy linear problem!)

Algorithm Outline



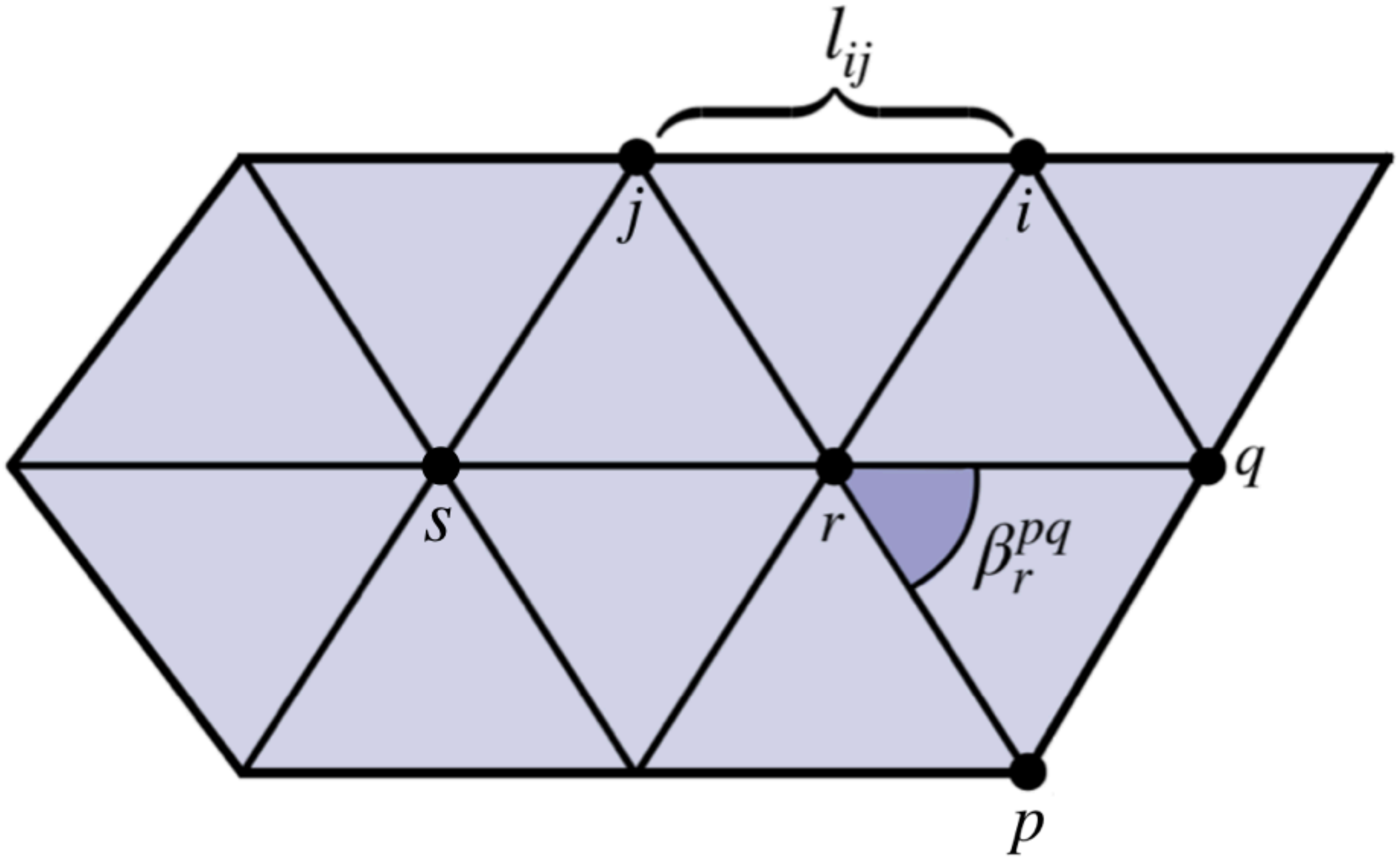
Given a surface with either scale **or** curvature of target boundary curve

1. Solve *Yamabe Problem* to get complementary data (curvature or scale)
2. Integrate boundary data to get boundary curve
3. Extend boundary curve to a pair of *conjugate harmonic functions*

DISCRETIZATION

Discretization

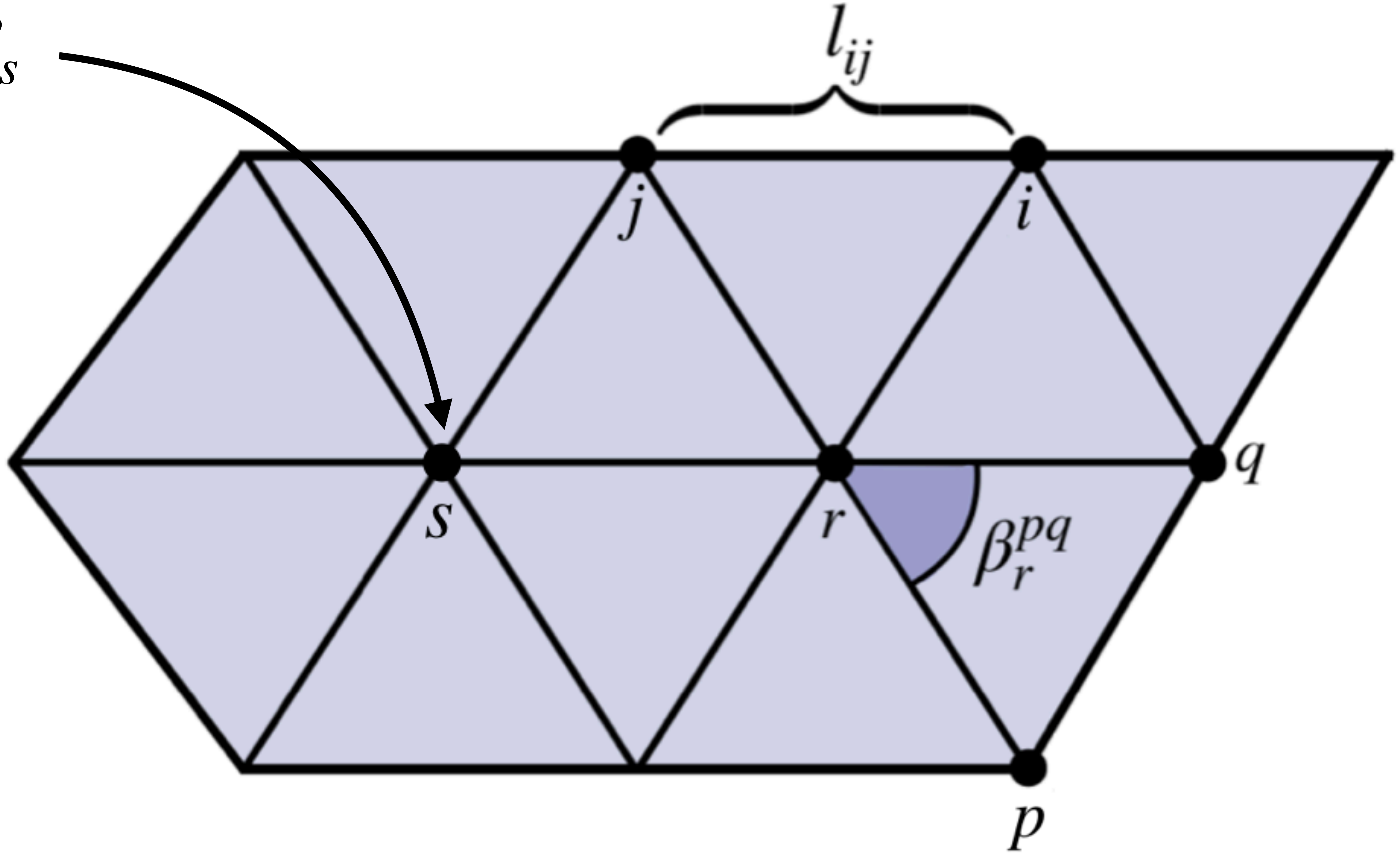
Discretize surface as a manifold triangle mesh with disk topology



Discretization

Discretize surface as a manifold triangle mesh with disk topology

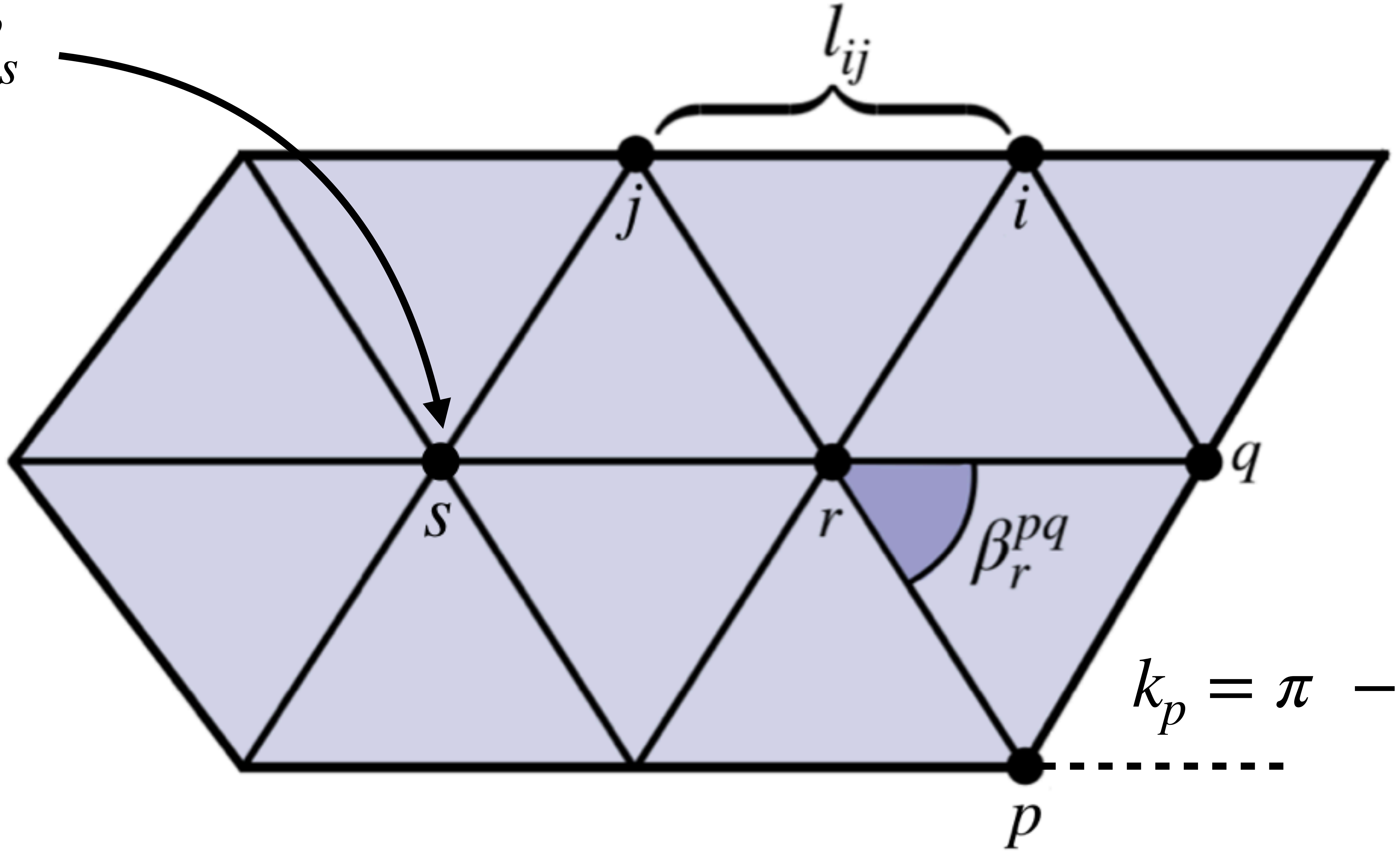
$$\Omega_s = 2\pi - \sum \beta_s$$



Discretization

Discretize surface as a manifold triangle mesh with disk topology

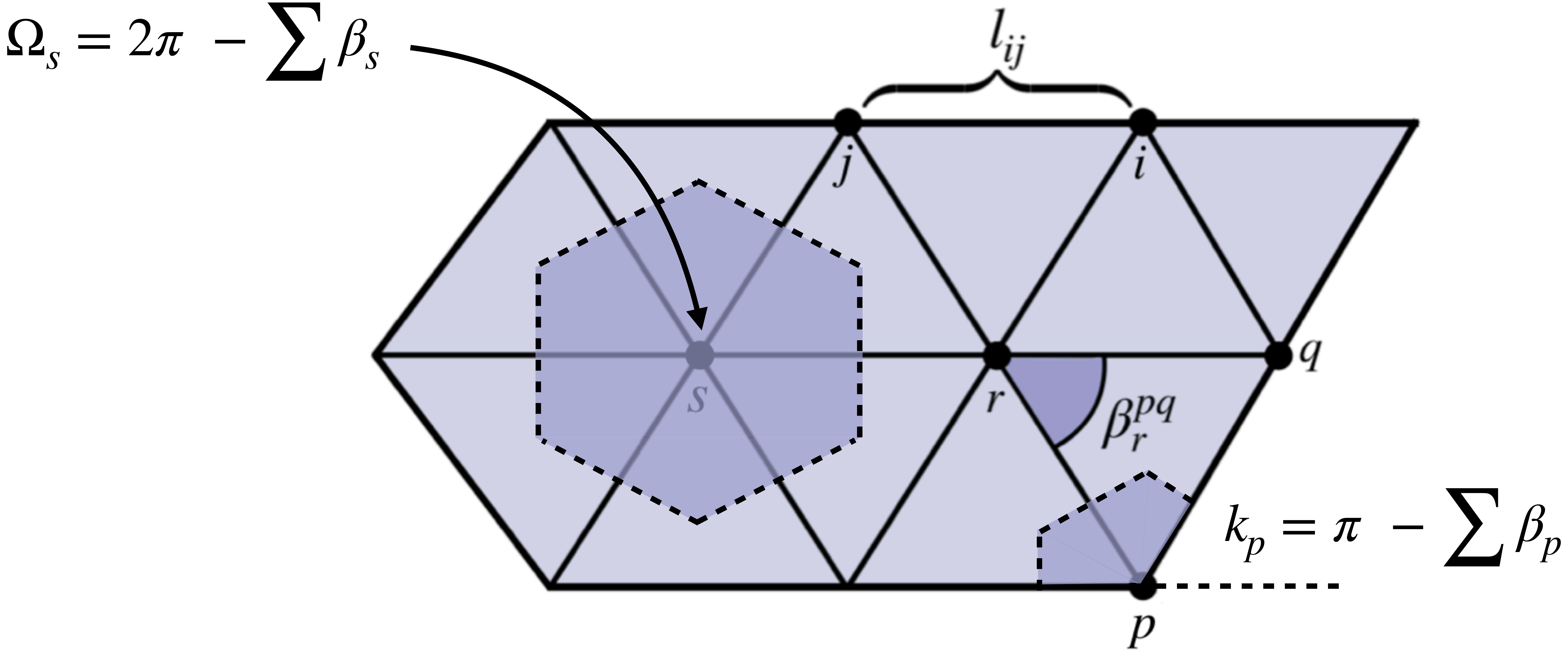
$\Omega_s = 2\pi \ - \sum \beta_s$



$k_p = \pi \ - \sum \beta_p$

Discretization

Discretize surface as a manifold triangle mesh with disk topology



$$\Omega_s = 2\pi - \sum \beta_s$$

$$k_p = \pi - \sum \beta_p$$

Discretizing the Yamabe Problem

Smooth *Yamabe Problem* is nonlinear

$$\Delta u = K - e^{2u}\tilde{K} \qquad on\ M$$

$$\frac{\partial u}{\partial n} = \kappa - e^{u}\tilde{\kappa} \qquad on\ \partial M$$

Discretizing the Yamabe Problem

Smooth *Yamabe Problem* is nonlinear

Integration over *dual volumes* yields **linear** relationships

$$\Delta u = K - e^{2u}\tilde{K} \qquad\qquad Au = \Omega - \tilde{\Omega} \qquad on\ M$$

$$\xRightarrow{integrating}$$

$$\frac{\partial u}{\partial n} = \kappa - e^{u}\tilde{\kappa} \qquad\qquad h = k - \tilde{k} \qquad on\ \partial M$$

Discretizing the Yamabe Problem

Smooth *Yamabe Problem* is nonlinear

Integration over *dual volumes* yields **linear** relationships

$$\begin{aligned} \Delta u &= K - e^{2u}\tilde{K} \\[2em] \frac{\partial u}{\partial n} &= \kappa - e^{u}\tilde{\kappa} \end{aligned} \qquad \xRightarrow{integrating} \qquad \begin{aligned} &\underset{\substack{\uparrow \\ \text{cotan} \\ \text{Laplace matrix}}}{A}u = \Omega - \tilde{\Omega} && on\ M \\[1em] &h = k - \tilde{k} && on\ \partial M \end{aligned}$$

Discretizing the Yamabe Problem

Smooth *Yamabe Problem* is nonlinear

Integration over *dual volumes* yields **linear** relationships

$$\Delta u = K - e^{2u}\tilde{K}$$

$$\frac{\partial u}{\partial n} = \kappa - e^{u}\tilde{\kappa}$$

$$\xrightarrow{\textit{integrating}}$$

$$\underset{\substack{\uparrow\\ \text{cotan}\\ \text{Laplace matrix}}}{A}u = \underset{\substack{\nwarrow\\ \text{old}\\ \text{angle defects}}}{\Omega} - \underset{\substack{\nwarrow\\ \text{new}\\ \text{angle defects}}}{\tilde{\Omega}} \qquad \textit{on } M$$

$$h = k - \tilde{k} \qquad \textit{on } \partial M$$

Discretizing the Yamabe Problem

Smooth *Yamabe Problem* is nonlinear

- Integration over *dual volumes* yields linear relationships

$$\Delta u = K - e^{2u}\tilde{K}$$

$$\frac{\partial u}{\partial n} = \kappa - e^{u}\tilde{\kappa}$$

$$\overset{\textit{integrating}}{\Longrightarrow}$$

$$\underset{\substack{\uparrow \\ \text{cotan} \\ \text{Laplace matrix}}}{A}u = \underset{\substack{\nwarrow \\ \text{old} \\ \text{angle defects}}}{\Omega} - \underset{\substack{\nwarrow \\ \text{new} \\ \text{angle defects}}}{\tilde{\Omega}} \qquad on\ M$$

$$\underset{\substack{\uparrow \\ \text{Neumann} \\ \text{boundary data}}}{h} = k - \tilde{k} \qquad on\ \partial M$$

Discretizing the Yamabe Problem

Smooth *Yamabe Problem* is nonlinear

Integration over *dual volumes* yields **linear** relationships

$$\Delta u = K - e^{2u}\tilde{K}$$

$$\frac{\partial u}{\partial n} = \kappa - e^{u}\tilde{\kappa}$$

$$\overset{integrating}{\Longrightarrow}$$

$$Au = \Omega - \tilde{\Omega} \qquad on\ M$$

cotan Laplace matrix

old angle defects

new angle defects

$$h = k - \tilde{k} \qquad on\ \partial M$$

Neumann boundary data

old exterior angles

new exterior angles

Discretizing the Yamabe Problem

Smooth *Yamabe Problem* is nonlinear

Integration over *dual volumes* yields **linear** relationships

$$\Delta u = K - e^{2u}\tilde{K}$$

$$\frac{\partial u}{\partial n} = \kappa - e^{u}\tilde{\kappa}$$

$$\xRightarrow{integrating}$$

$$\underset{\text{cotan Laplace matrix}}{A}u = \underset{\text{old angle defects}}{\Omega} - \underset{\text{new angle defects}}{\cancelto{0}{\tilde{\Omega}}} \qquad on\ M$$

$$\underset{\text{Neumann boundary data}}{h} = \underset{\text{old exterior angles}}{k} - \underset{\text{new exterior angles}}{\tilde{k}} \qquad on\ \partial M$$

Discretizing the Yamabe Problem

Smooth *Yamabe Problem* is nonlinear

Integration over *dual volumes* yields **linear** relationships

$$\Delta u = K - e^{2u}\tilde{K}$$

$$\xRightarrow{integrating}$$

$$\underset{\substack{\uparrow\\ \text{cotan}\\ \text{Laplace matrix}}}{A}u = \underset{\substack{\nwarrow\\ \text{old}\\ \text{angle defects}}}{\Omega} - \underset{\substack{\nwarrow\\ \text{new}\\ \text{angle defects}}}{\cancelto{0}{\tilde{\Omega}}} \qquad on\ M$$

$$\frac{\partial u}{\partial n} = \kappa - e^{u}\tilde{\kappa}$$

$$\underset{\substack{\uparrow\\ \text{Neumann}\\ \text{boundary data}}}{h} = \underset{\substack{\nwarrow\\ \text{old}\\ \text{exterior angles}}}{k} - \underset{\substack{\nwarrow\\ \text{new}\\ \text{exterior angles}}}{\tilde{k}} \qquad on\ \partial M$$

Can prescribe either exterior angles or scaling,
but not both!

Poincaré Steklov Operators

How do we switch between angles and scale factors?

$$Au = \Omega \qquad on\ M$$

$$h = k - \tilde{k} \qquad on\ \partial M$$

Poincaré Steklov Operators

How do we switch between angles and scale factors?

Rewrite integrated *Yamabe Problem* in block matrix form

$$\begin{bmatrix} A_{II} & A_{IB} \\ A_{IB}^T & A_{BB} \end{bmatrix} \begin{bmatrix} u_I \\ u_B \end{bmatrix} = \begin{bmatrix} \Omega \\ -(k - \tilde{k}) \end{bmatrix}$$

Poincaré Steklov Operators

How do we switch between angles and scale factors?

Rewrite integrated *Yamabe Problem* in block matrix form

$$\begin{bmatrix} A_{II} & A_{IB} \\ A_{IB}^T & A_{BB} \end{bmatrix} \begin{bmatrix} u_I \\ u_B \end{bmatrix} = \begin{bmatrix} \Omega \\ -(k - \tilde{k}) \end{bmatrix}$$

NEUMANN TO DIRICHLET MAP

DIRICHLET TO NEUMANN MAP

Poincaré Steklov Operators

How do we switch between angles and scale factors?

Rewrite integrated *Yamabe Problem* in block matrix form

$$\begin{bmatrix} A_{II} & A_{IB} \\ A_{IB}^T & A_{BB} \end{bmatrix} \begin{bmatrix} u_I \\ u_B \end{bmatrix} = \begin{bmatrix} \Omega \\ -(k - \boxed{\tilde{k}}) \end{bmatrix}$$

NEUMANN TO DIRICHLET MAP

Given $\tilde{k}$

DIRICHLET TO NEUMANN MAP

Poincaré Steklov Operators

How do we switch between angles and scale factors?

Rewrite integrated *Yamabe Problem* in block matrix form

$$\begin{bmatrix} A_{II} & A_{IB} \\ A_{IB}^T & A_{BB} \end{bmatrix} \begin{bmatrix} u_I \\ u_B \end{bmatrix} = \begin{bmatrix} \Omega \\ -(k - \tilde{k}) \end{bmatrix}$$

NEUMANN TO DIRICHLET MAP

Given $\tilde{k}$

Solve Neumann system above for u

DIRICHLET TO NEUMANN MAP

Poincaré Steklov Operators

How do we switch between angles and scale factors?

Rewrite integrated *Yamabe Problem* in block matrix form

$$\begin{bmatrix} A_{II} & A_{IB} \\ A_{IB}^T & A_{BB} \end{bmatrix} \begin{bmatrix} u_I \\ u_B \end{bmatrix} = \begin{bmatrix} \Omega \\ -(k - \tilde{k}) \end{bmatrix}$$

NEUMANN TO DIRICHLET MAP

Given $\tilde{k}$

Solve Neumann system above for u

Read off u_B

DIRICHLET TO NEUMANN MAP

Poincaré Steklov Operators

How do we switch between angles and scale factors?

Rewrite integrated *Yamabe Problem* in block matrix form

$$\begin{bmatrix} A_{II} & A_{IB} \\ A_{IB}^T & A_{BB} \end{bmatrix} \begin{bmatrix} u_I \\ \enclose{circle}{u_B} \end{bmatrix} = \begin{bmatrix} \Omega \\ -(k - \tilde{k}) \end{bmatrix}$$

NEUMANN TO DIRICHLET MAP

Given $\tilde{k}$

Solve Neumann system above for u

Read off u_B

DIRICHLET TO NEUMANN MAP

Given u_B

Poincaré Steklov Operators

How do we switch between angles and scale factors?

Rewrite integrated *Yamabe Problem* in block matrix form

$$\begin{bmatrix} A_{II} & A_{IB} \\ A_{IB}^T & A_{BB} \end{bmatrix} \begin{bmatrix} u_I \\ u_B \end{bmatrix} = \begin{bmatrix} \Omega \\ -(k - \tilde{k}) \end{bmatrix}$$

NEUMANN TO DIRICHLET MAP

Given $\tilde{k}$

Solve Neumann system above for u

Read off u_B

DIRICHLET TO NEUMANN MAP

Given u_B

Solve Dirichlet Problem $A_{II}u_I = \Omega \; - \; A_{IB}u_B$ for u_I

Poincaré Steklov Operators

How do we switch between angles and scale factors?

Rewrite integrated *Yamabe Problem* in block matrix form

$$\begin{bmatrix} A_{II} & A_{IB} \\ A_{IB}^T & A_{BB} \end{bmatrix} \begin{bmatrix} u_I \\ u_B \end{bmatrix} = \begin{bmatrix} \Omega \\ -(k - \tilde{k}) \end{bmatrix}$$

NEUMANN TO DIRICHLET MAP

Given $\tilde{k}$

Solve Neumann system above for u

Read off u_B

DIRICHLET TO NEUMANN MAP

Given u_B

Solve Dirichlet Problem $A_{II}u_I = \Omega - A_{IB}u_B$ for u_I

$\tilde{k} = k + A_{IB}^T u_I + A_{BB}u_B$

Poincaré Steklov Operators

How do we switch between angles and scale factors?

Rewrite integrated *Yamabe Problem* in block matrix form

$$\begin{bmatrix} A_{II} & A_{IB} \\ A_{IB}^T & A_{BB} \end{bmatrix} \begin{bmatrix} u_I \\ u_B \end{bmatrix} = \begin{bmatrix} \Omega \\ -(k - \tilde{k}) \end{bmatrix}$$

NEUMANN TO DIRICHLET MAP

Given $\tilde{k}$

Solve Neumann system above for u

Read off u_B

DIRICHLET TO NEUMANN MAP

Given u_B

Solve Dirichlet Problem $A_{II}u_I = \Omega \; - \; A_{IB}u_B$ for u_I

$\tilde{k} = k \; + \; A_{IB}^T u_I \; + \; A_{BB}u_B$

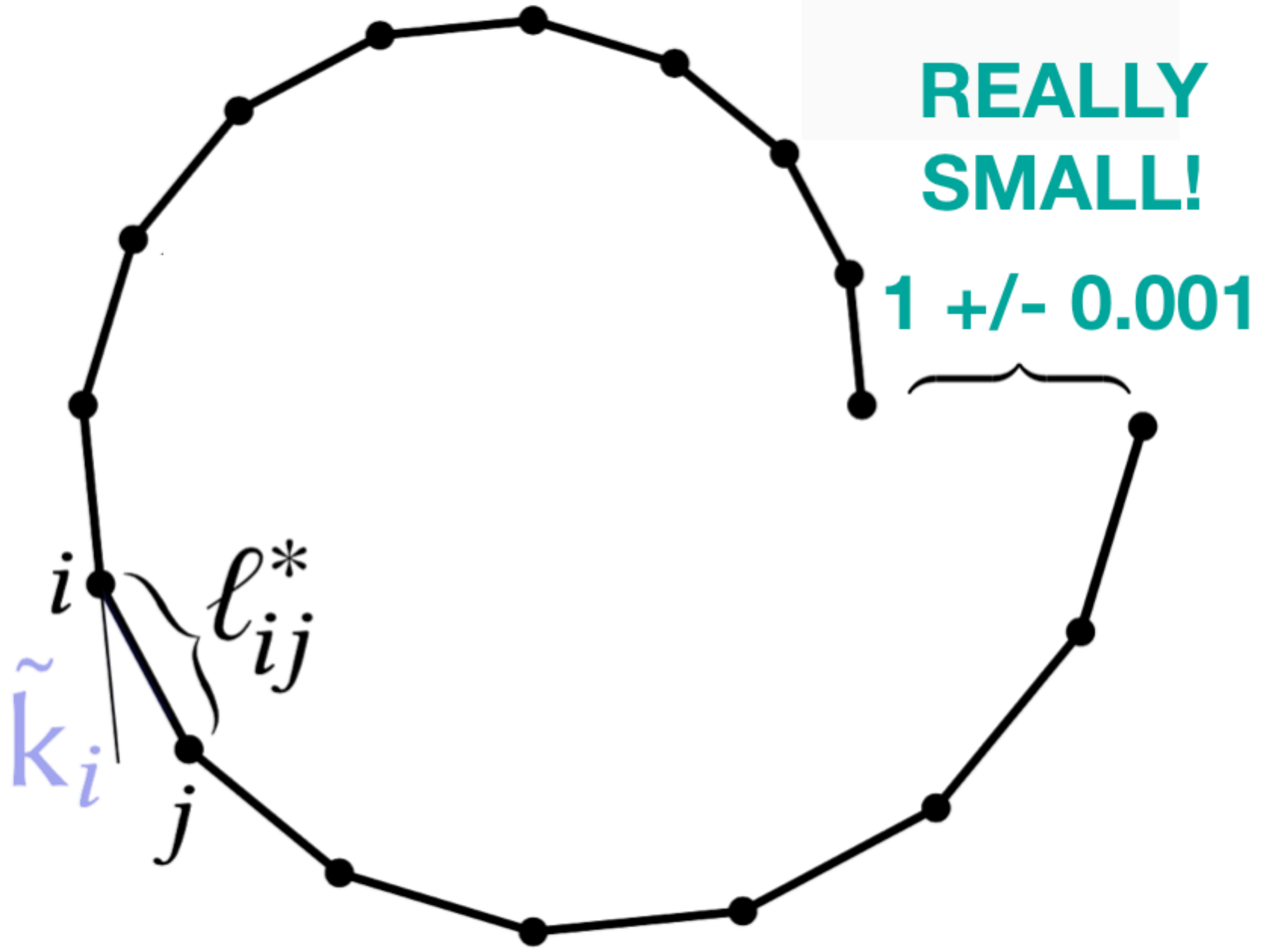
Angles exactly sum to **2π**!

Curve Integration

Rescale boundary edge lengths using scale factors u

Curve Integration

- Rescale boundary edge lengths using scale factors u
  - Extremely small discretization errors prevent curve from closing



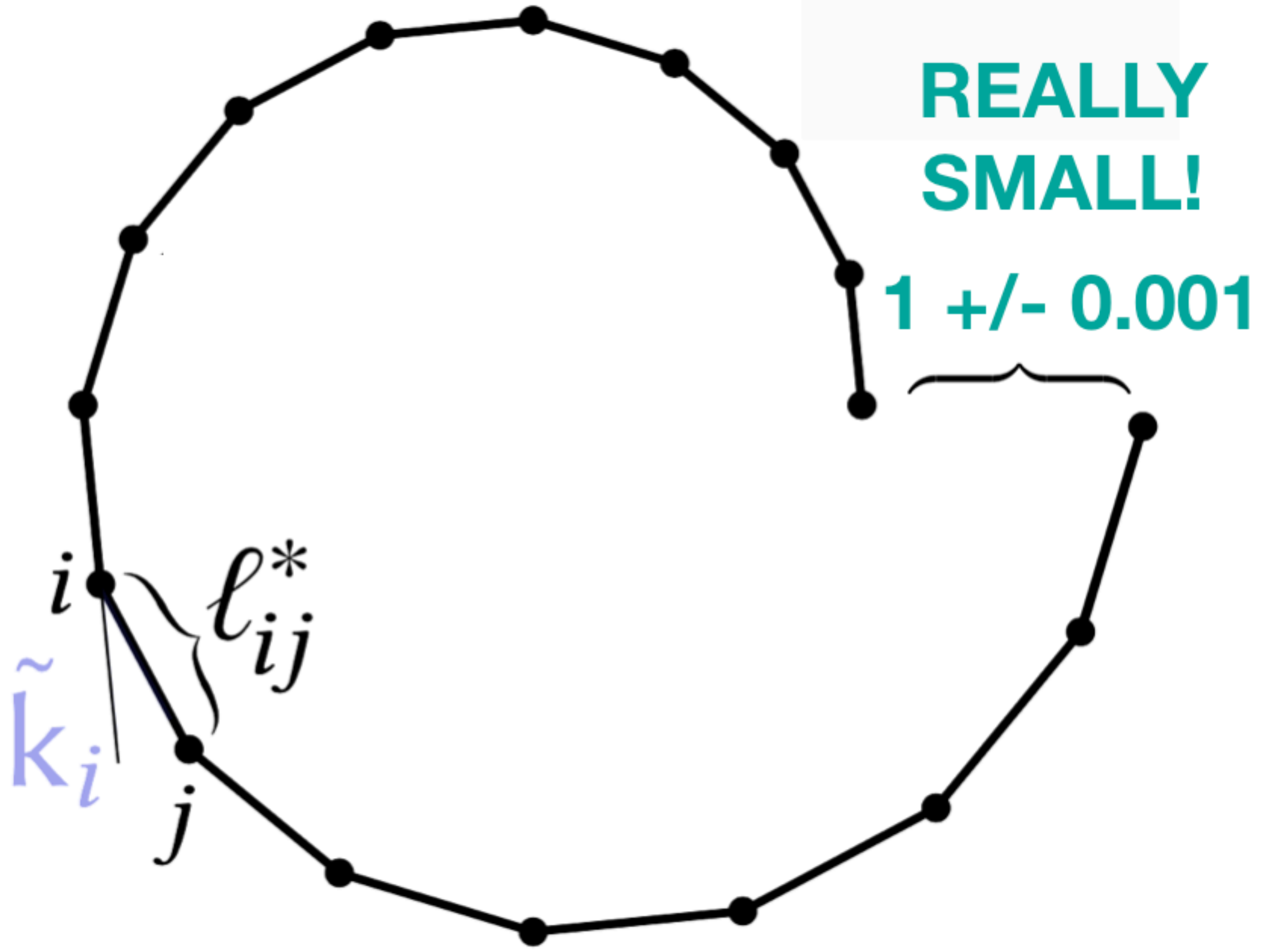
REALLY SMALL!

1 +/- 0.001

Curve Integration

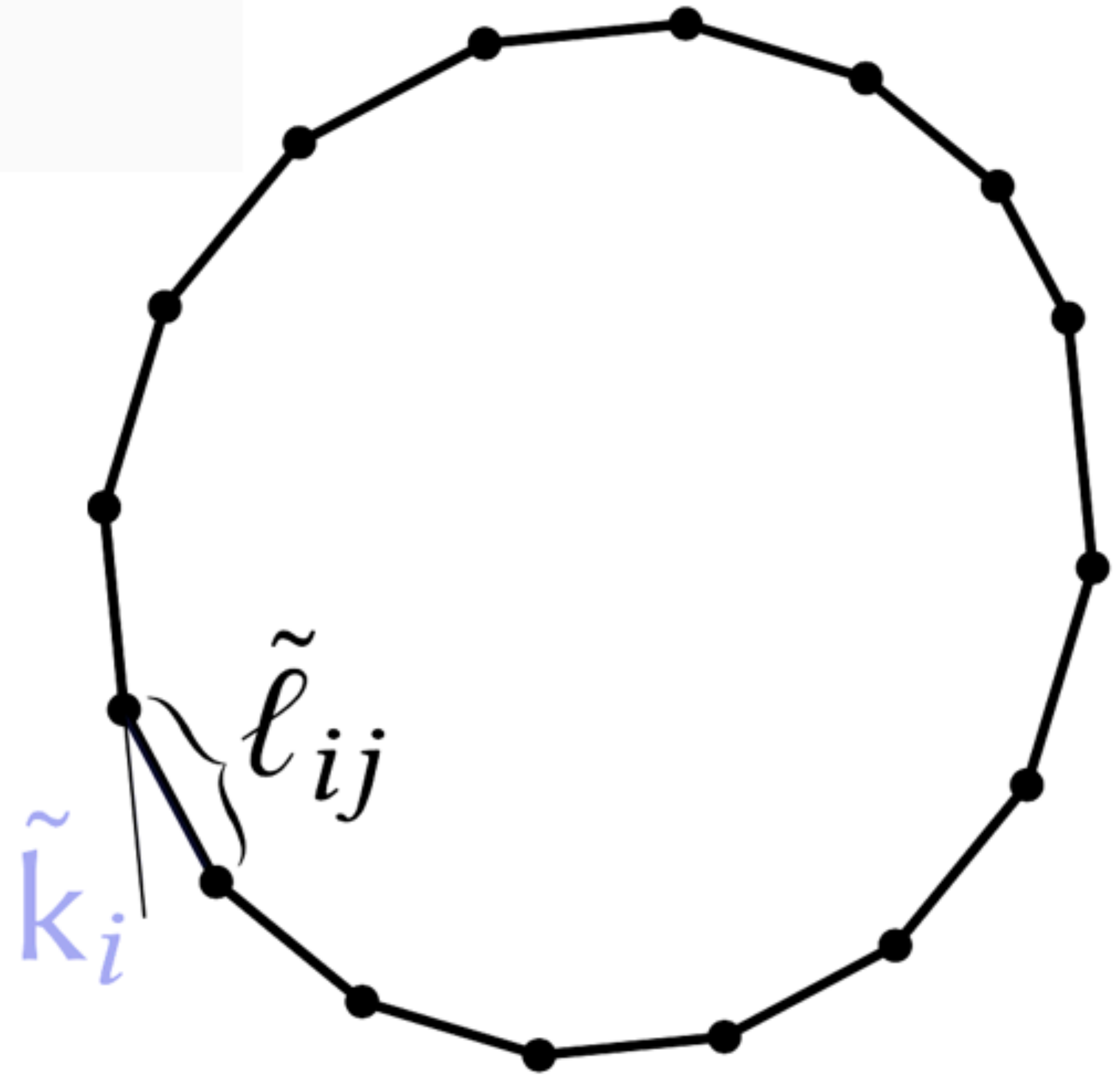
Rescale boundary edge lengths using scale factors u

- Extremely small discretization errors prevent curve from closing
- Formulate small least squares problem to adjust **only** lengths to close curve



REALLY SMALL!

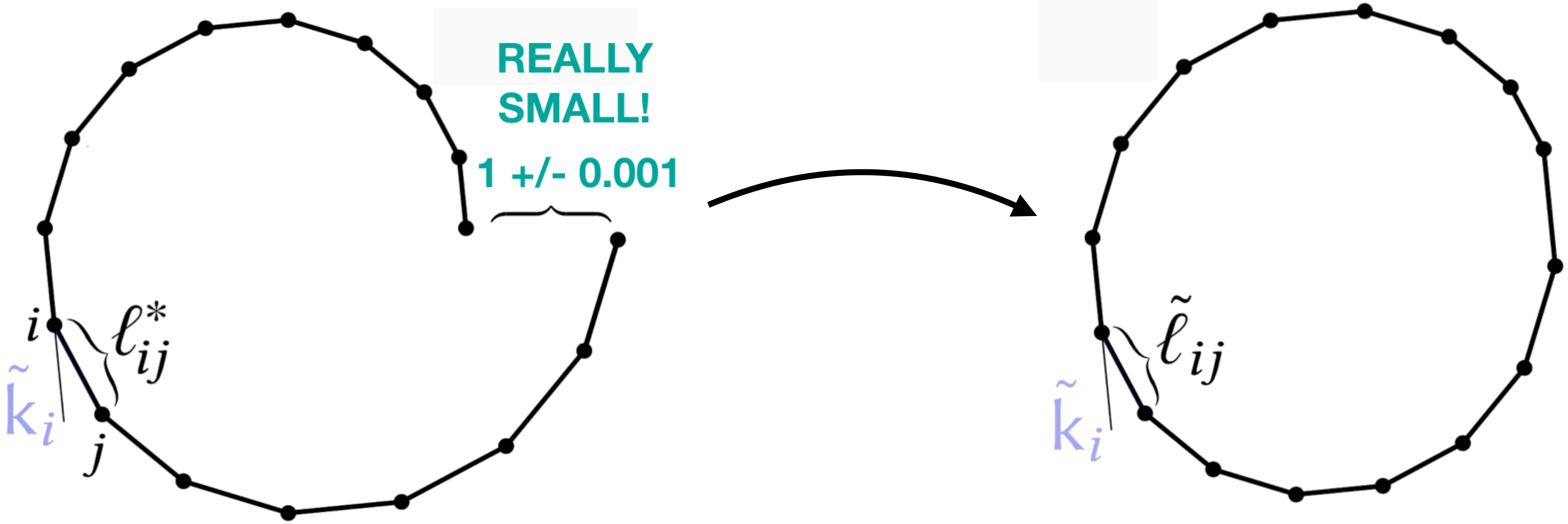
1 +/- 0.001



Curve Integration

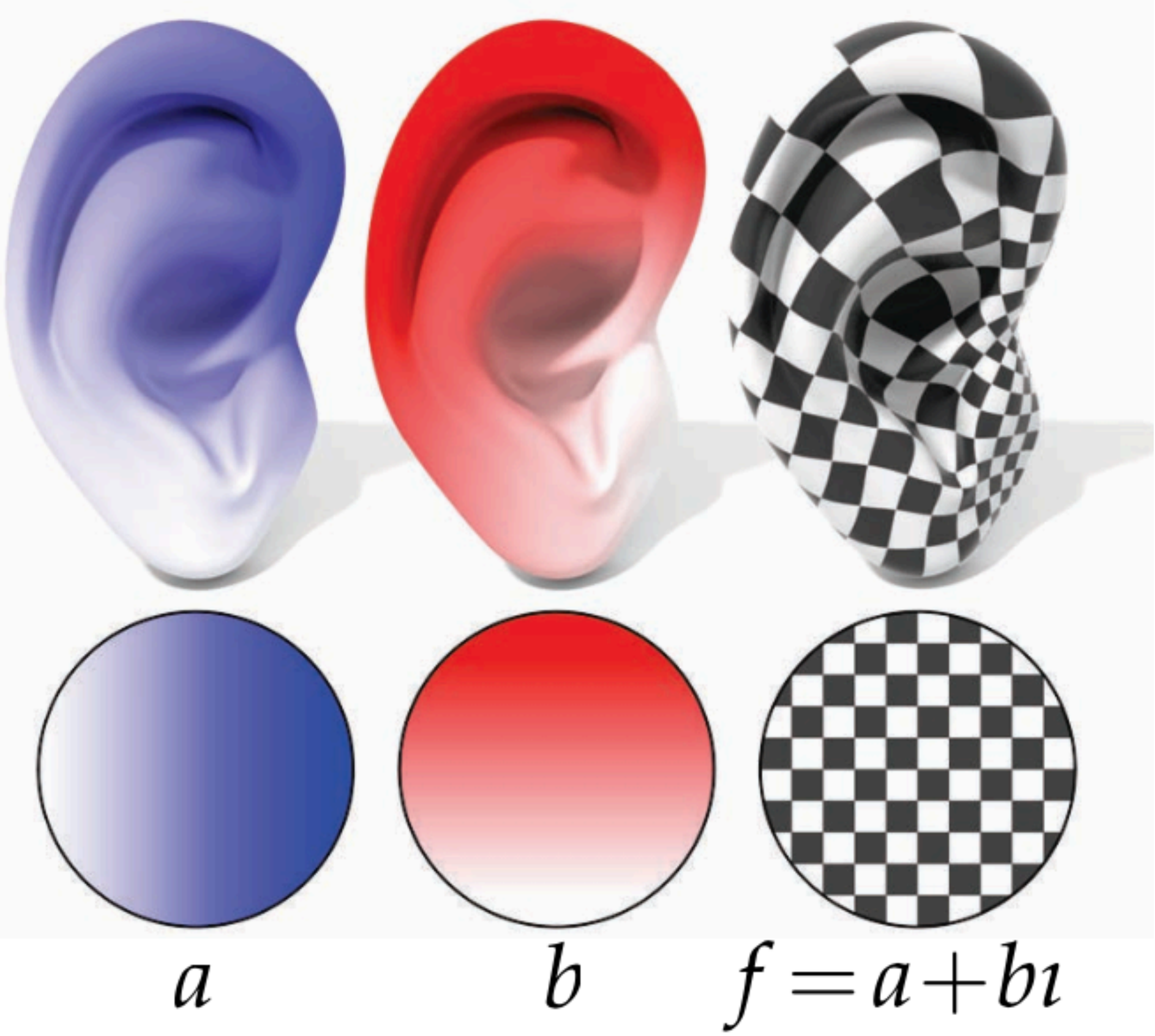
Rescale boundary edge lengths using scale factors u

- Extremely small discretization errors prevent curve from closing
- Formulate small least squares problem to adjust **only** lengths to close curve
- Exterior angles are **exactly** preserved



Harmonic Extension and Conjugation

Basic Idea: Fix one coordinate of $\tilde{\gamma}$, minimize discrete conformal energy w.r.t. other coordinate

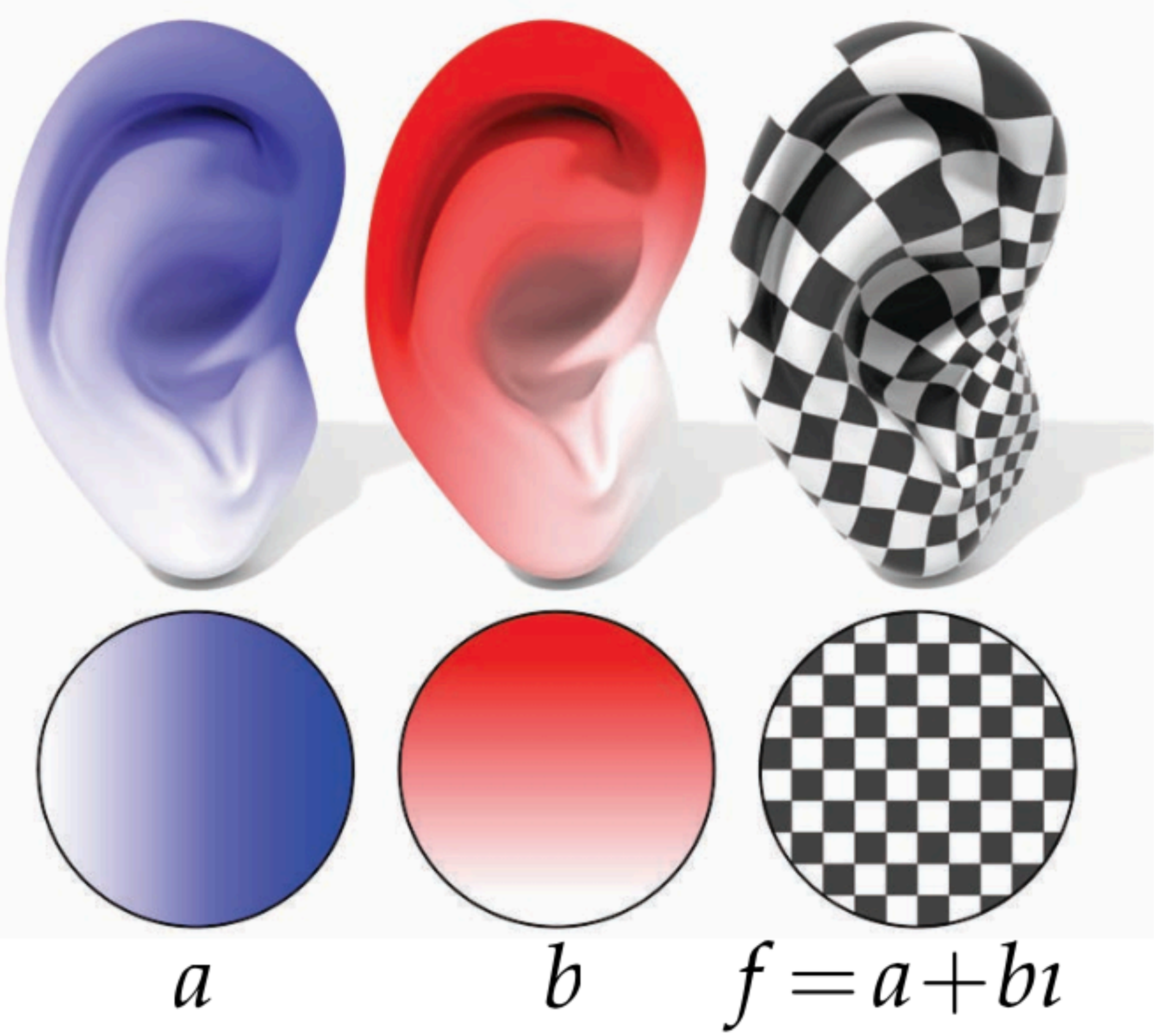


Harmonic Extension and Conjugation

Basic Idea: Fix one coordinate of $\tilde{\gamma}$, minimize discrete conformal energy w.r.t. other coordinate

$$\Delta a = 0 \ s.t. \ a|_{\partial M} = Re(\tilde{\gamma})$$

$$\Delta b = 0 \ s.t. \ \frac{\partial b}{\partial n} = Ha$$

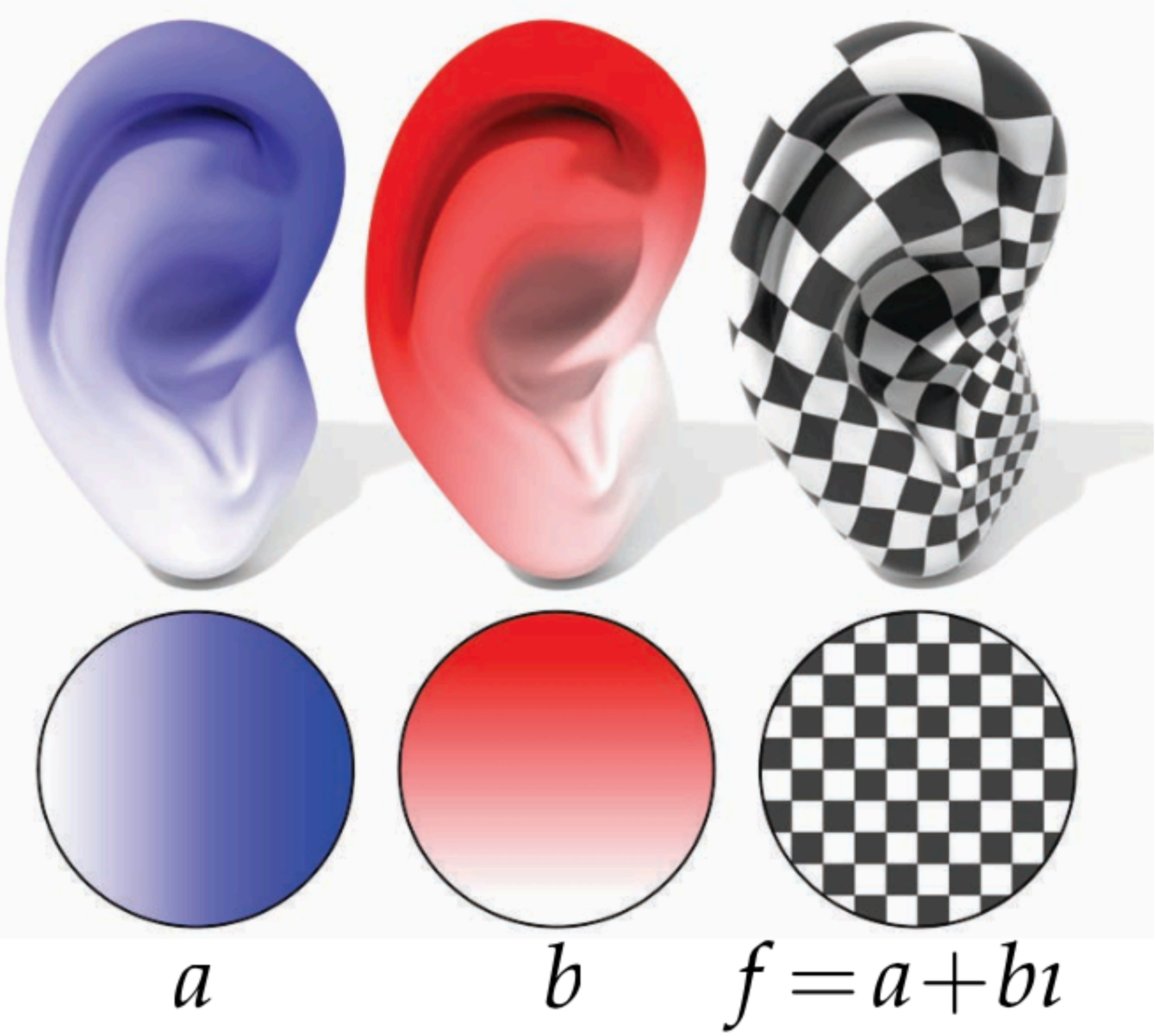


Harmonic Extension and Conjugation

Basic Idea: Fix one coordinate of $\tilde{\gamma}$, minimize discrete conformal energy w.r.t. other coordinate

$$\Delta a = 0 \; s.t. \; a|_{\partial M} = Re(\tilde{\gamma})$$

$$\Delta b = 0 \; s.t. \; \frac{\partial b}{\partial n} = \underset{\substack{\uparrow \\ \text{Hilbert Transform}}}{Ha}$$



Harmonic Extension and Conjugation

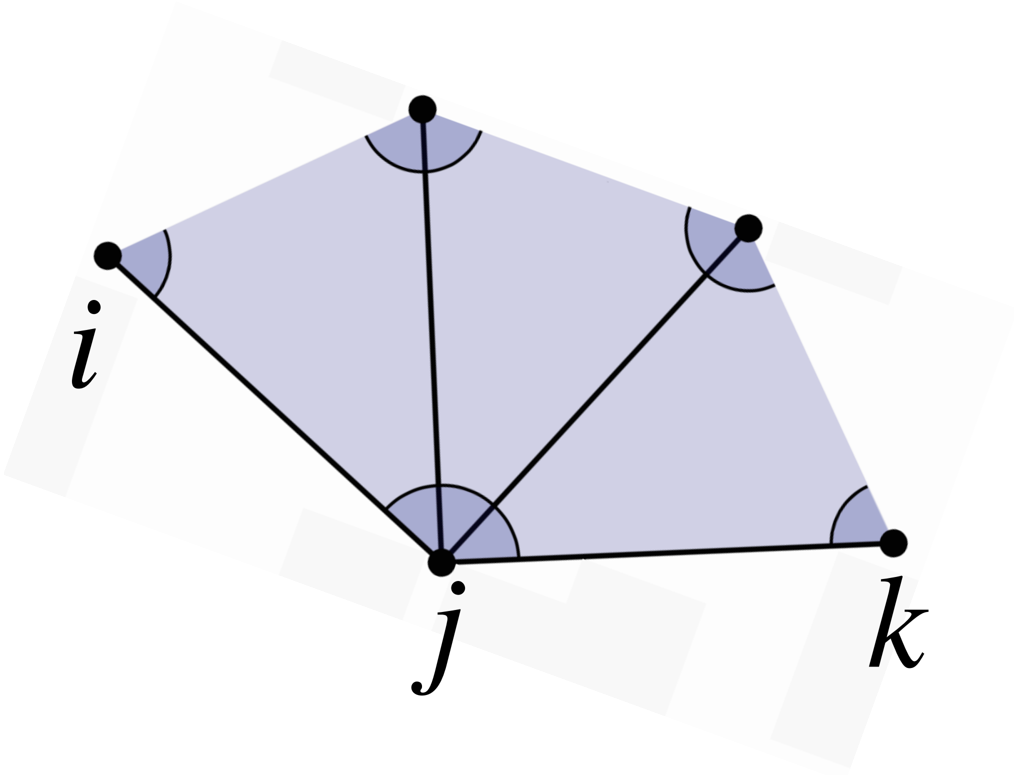
Basic Idea: Fix one coordinate of $\tilde{\gamma}$, minimize discrete conformal energy w.r.t. other coordinate

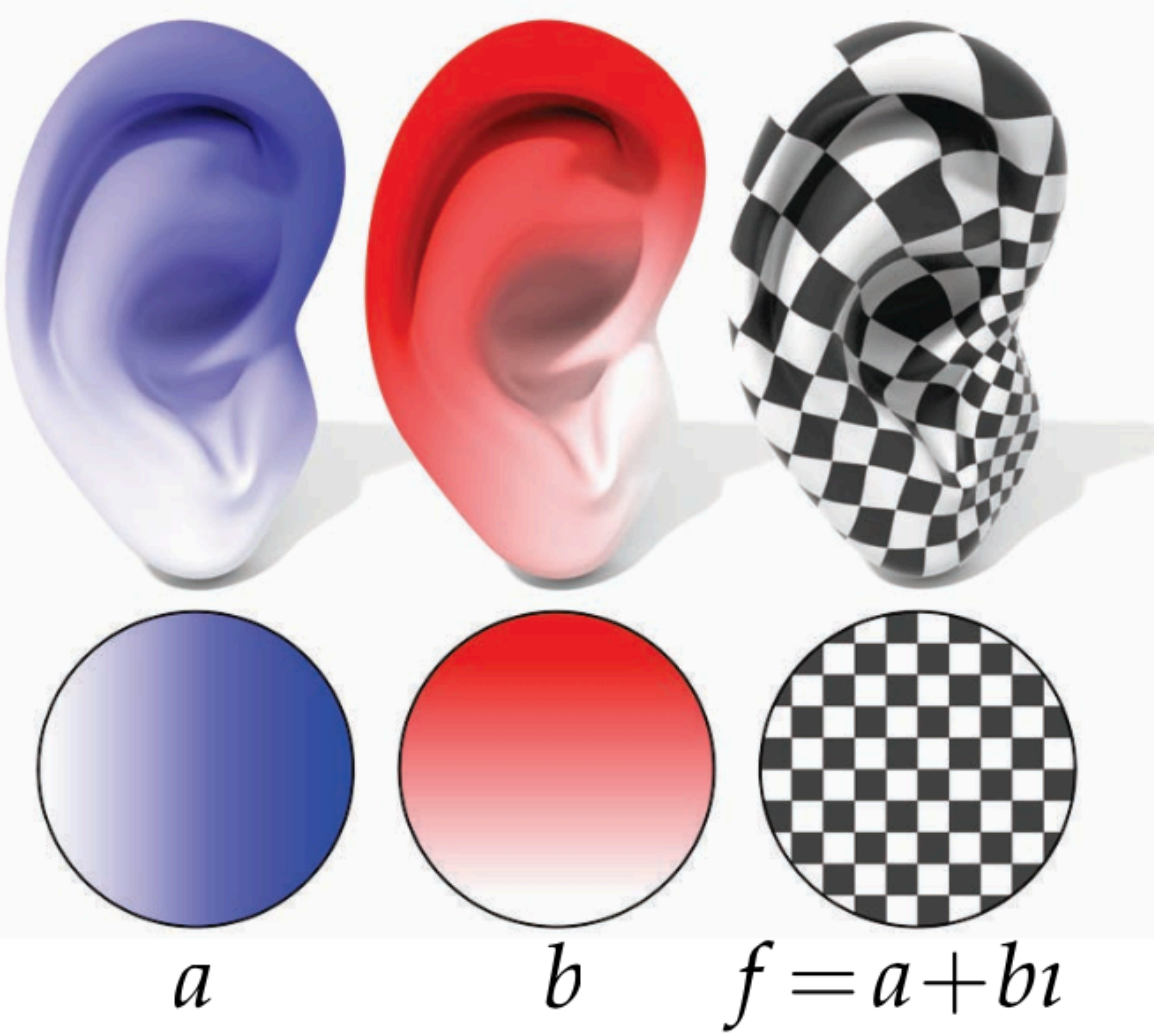
$$\Delta a = 0 \ s.t. \ a|_{\partial M} = Re(\tilde{\gamma})$$

$$\Delta b = 0 \ s.t. \ \frac{\partial b}{\partial n} = \underset{\uparrow}{H}a$$

Hilbert Transform

$$h_j = \frac{1}{2}(a_k - a_i)$$





Harmonic Extension and Conjugation

Basic Idea: Fix one coordinate of $\tilde{\gamma}$, minimize discrete conformal energy w.r.t. other coordinate

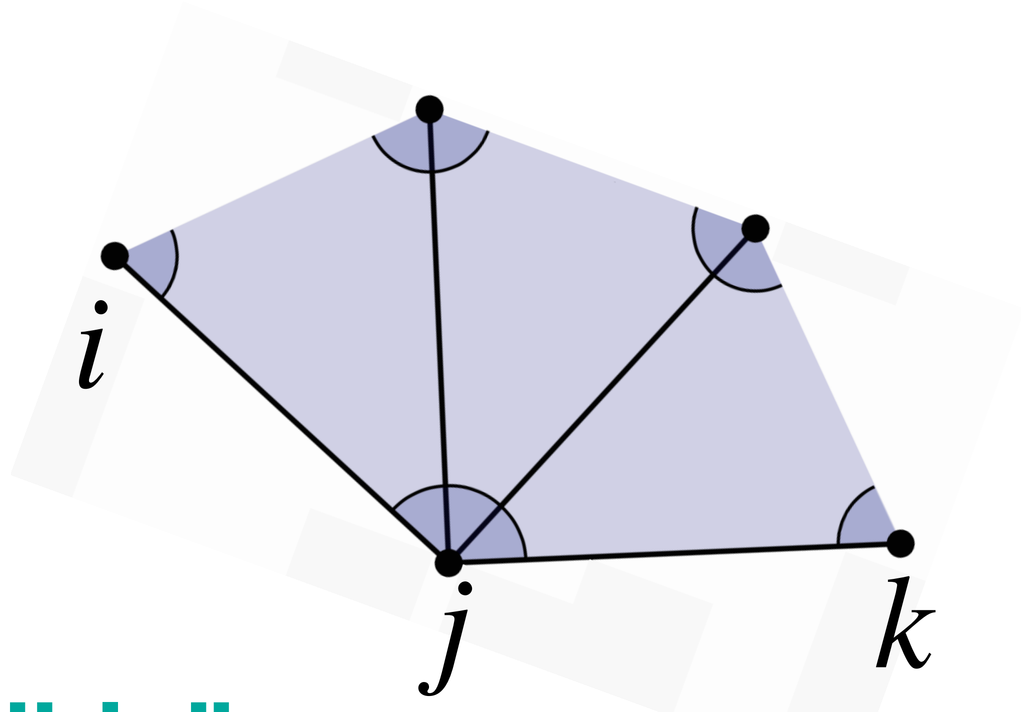
$$\Delta a = 0 \; s.t. \; a|_{\partial M} = Re(\tilde{\gamma})$$

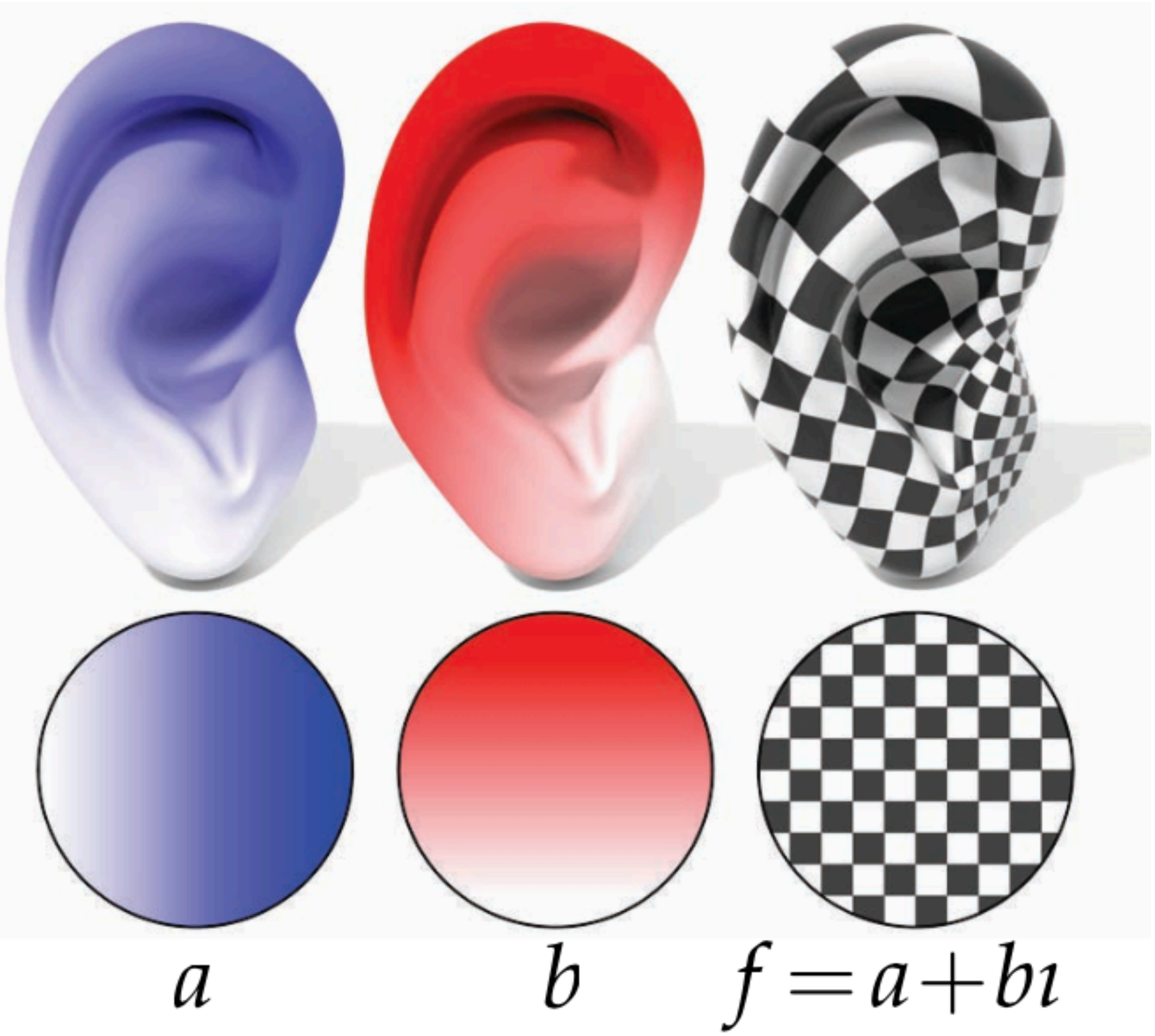
$$\Delta b = 0 \; s.t. \; \frac{\partial b}{\partial n} = \underset{\uparrow}{Ha}$$

Hilbert Transform

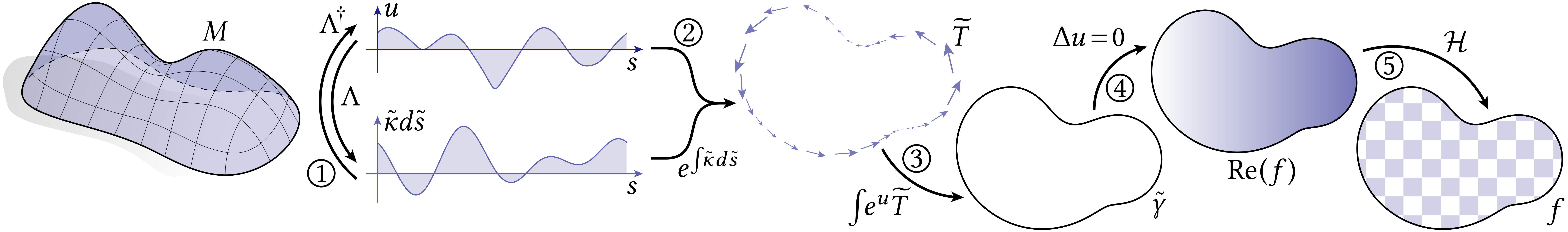
$$h_j = \frac{1}{2}(a_k - a_i)$$

"as conjugate as possible"





Algorithm Outline



Given a surface with either scale **or** curvature of target boundary curve

1. Solve *Yamabe Problem* to get complementary data (curvature or scale)
2. Integrate boundary data to get boundary curve
3. Extend boundary curve to a pair of *conjugate harmonic functions*

APPLICATIONS

Automatic Flattening

User does **not** have to specify boundary curve: automatically pick flattening with **minimal scale distortion**

Automatic Flattening

User does **not** have to specify boundary curve: automatically pick flattening with **minimal scale distortion**

Theorem. [Springborn, Schröder, Pinkall] *Let* (M, g) *be a surface with boundary. Then among all conformally equivalent flat metrics* $\tilde{g} = e^{2u}g$*, the ones with least area distortion are obtained if* $u|_{\partial M} = \text{const.}$

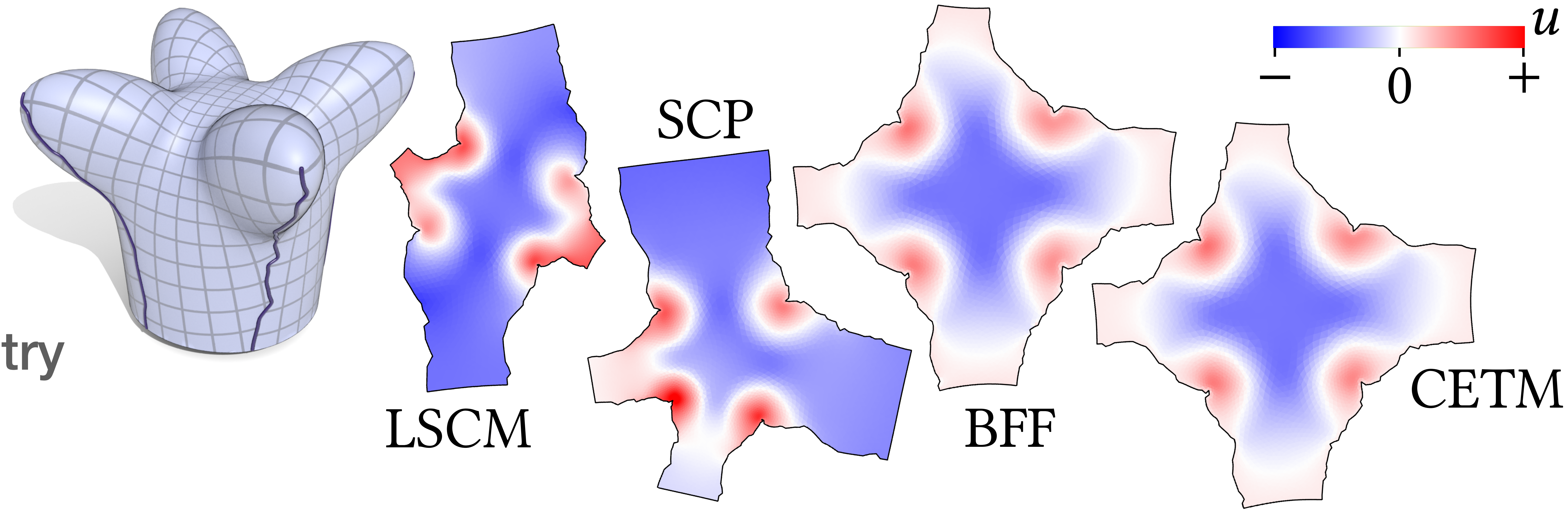
Automatic Flattening

User does **not** have to specify boundary curve: automatically pick flattening with **minimal scale distortion**

Theorem. [Springborn, Schröder, Pinkall] *Let* (M, g) *be a surface with boundary. Then among all conformally equivalent flat metrics* $\tilde{g} = e^{2u}g$, *the ones with least area distortion are obtained if* $u|_{\partial M} = \text{const.}$

Results indistinguishable from CETM

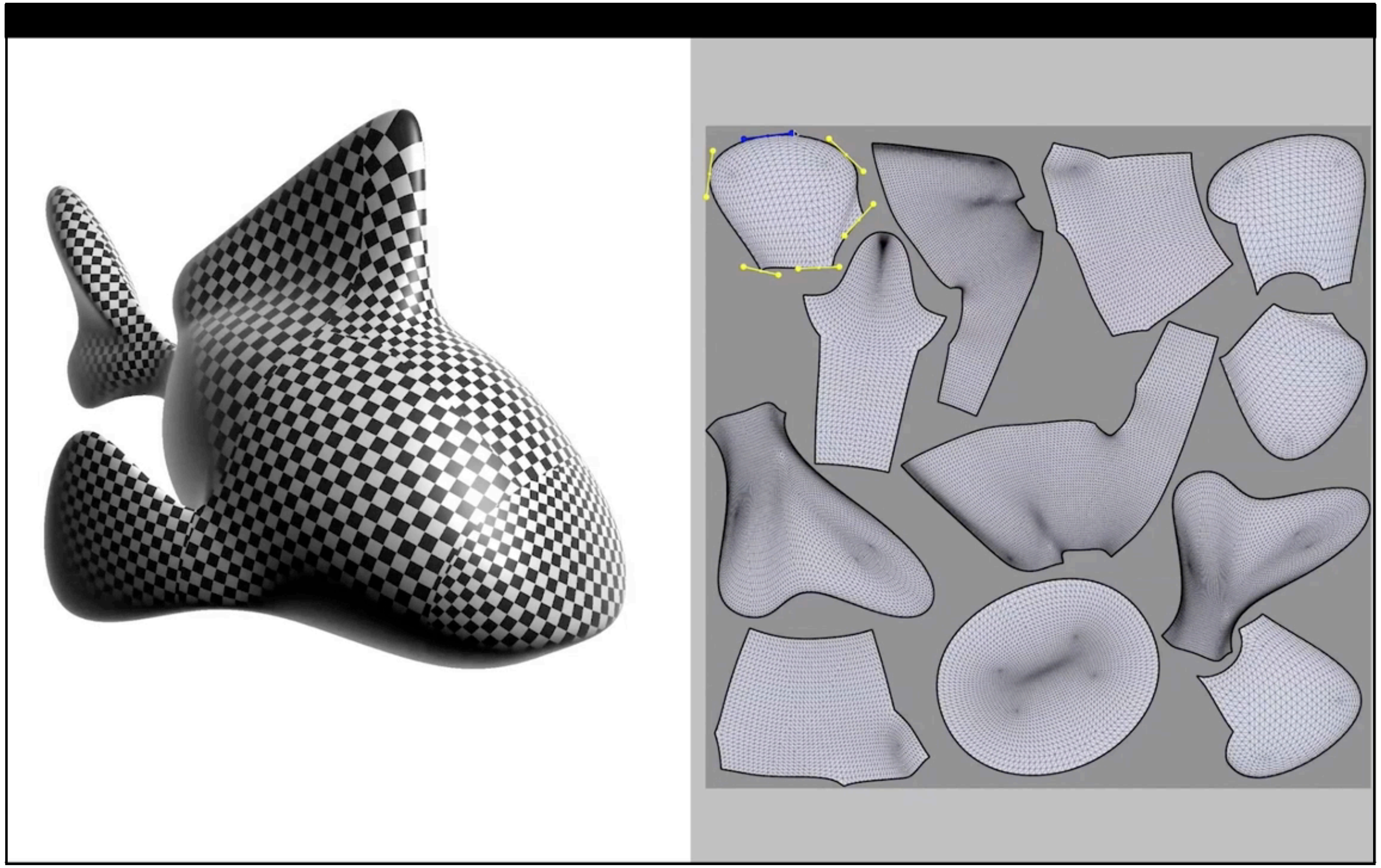
Better preservation of symmetry compared to LSCM and SCP



Direct Editing

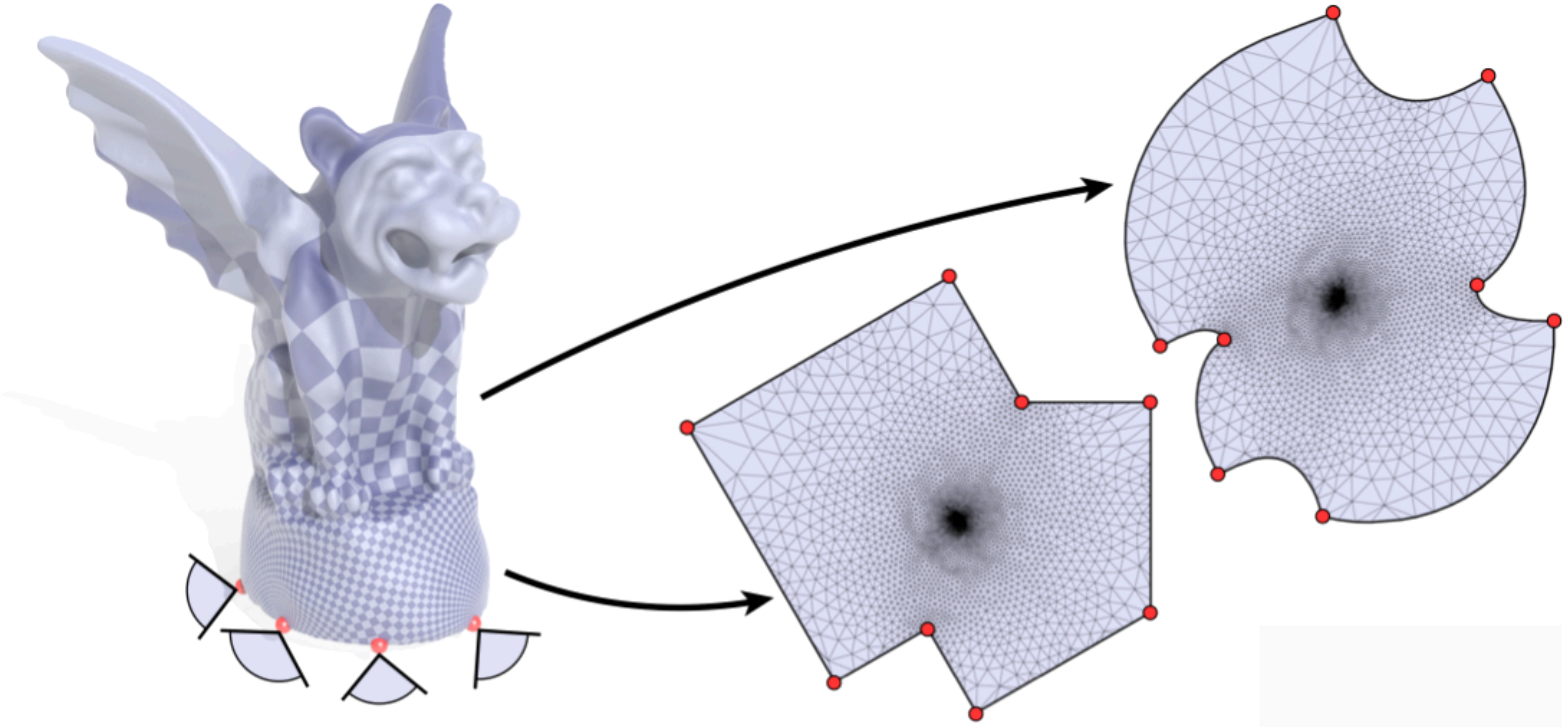
Spline based curve editor to manipulate target angles and lengths

Interactively and nonrigidly tweak a texture layout while remaining conformal



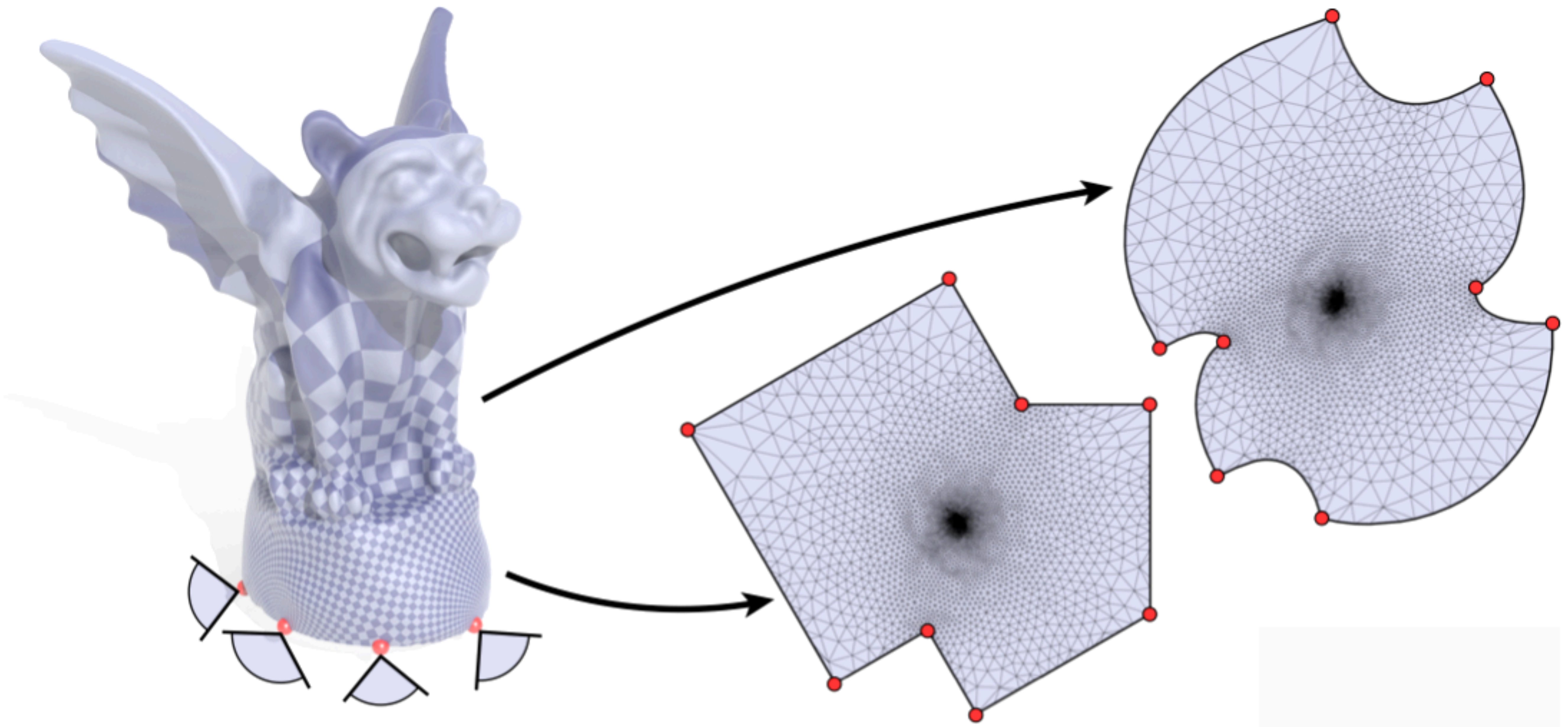
Exact Preservation of Sharp Corners

Harmonically extend both coordinates of $\tilde{\gamma}$ to **exactly** interpolate angles

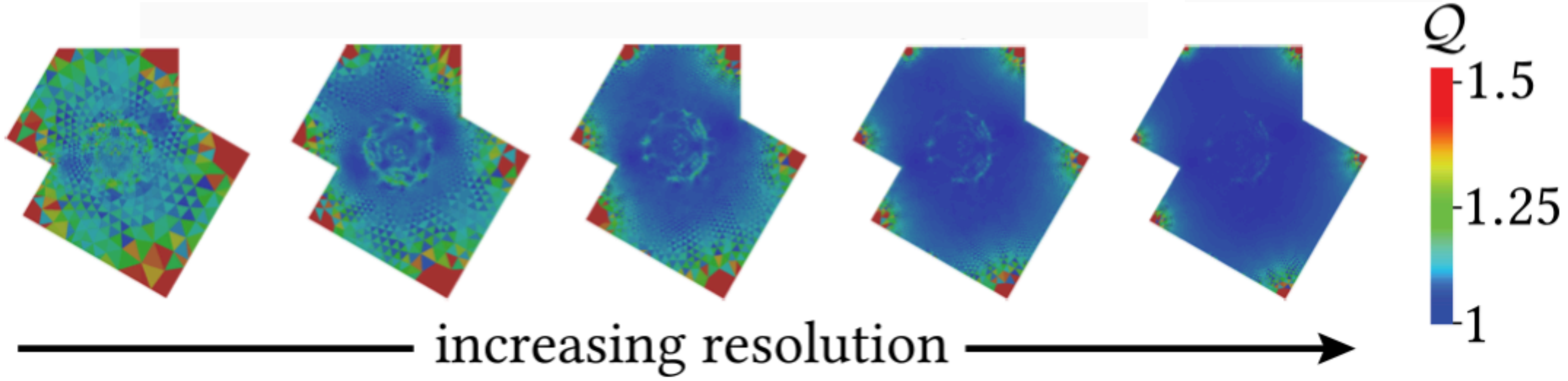


Exact Preservation of Sharp Corners

Harmonically extend both coordinates of $\tilde{\gamma}$ to **exactly** interpolate angles

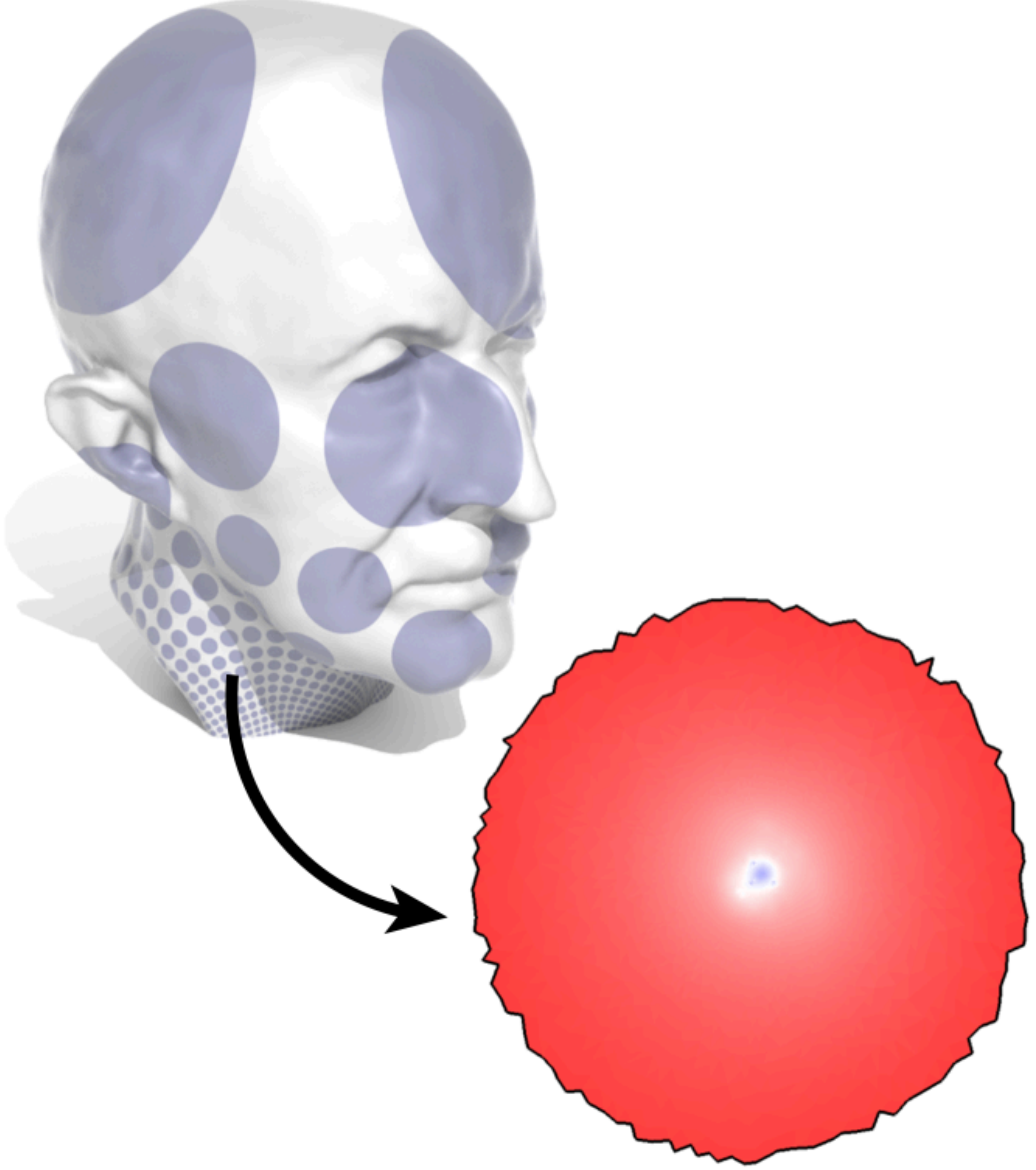


Converges to conformal map under refinement since $\tilde{\gamma}$ is approximately conformal



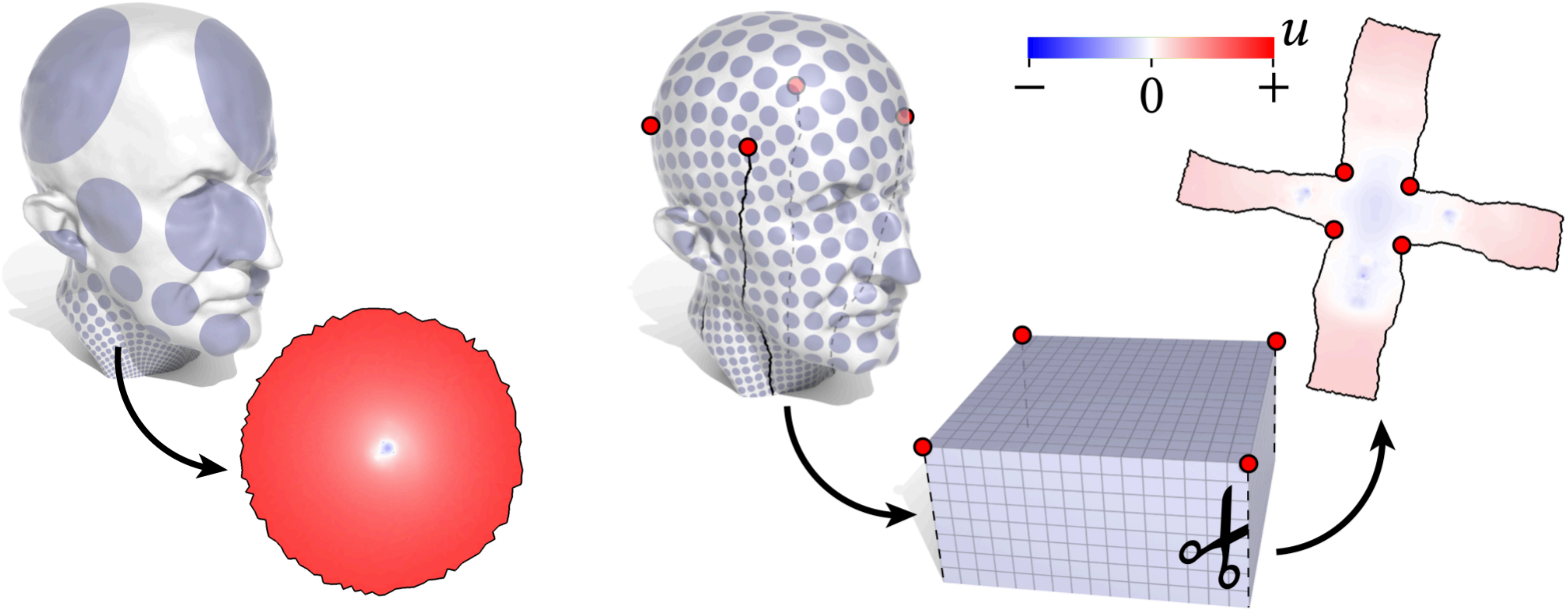
Seamless Cone Parameterization

Cone singularities offer a powerful technique for **mitigating scale distortion**



Seamless Cone Parameterization

Cone singularities offer a powerful technique for **mitigating scale distortion**



Seamless Cone Parameterization

Cone singularities in BFF

Seamless Cone Parameterization

Cone singularities in BFF

Solve *Yamabe Problem* with modified source term

$$\begin{bmatrix} A_{II} & A_{IB} \\ A_{IB}^T & A_{BB} \end{bmatrix} \begin{bmatrix} u_I \\ u_B \end{bmatrix} = \begin{bmatrix} \Omega - \Theta \\ -(k - \tilde{k}) \end{bmatrix}$$

cone angles

Seamless Cone Parameterization

Cone singularities in BFF

Solve *Yamabe Problem* with modified source term

$$\begin{bmatrix} A_{II} & A_{IB} \\ A_{IB}^T & A_{BB} \end{bmatrix} \begin{bmatrix} u_I \\ u_B \end{bmatrix} = \begin{bmatrix} \Omega - \boxed{\Theta} \\ -(k - \tilde{k}) \end{bmatrix}$$

cone angles

Cut through cones, prescribing u along cut

Seamless Cone Parameterization

Cone singularities in BFF

Solve *Yamabe Problem* with modified source term

$$\begin{bmatrix} A_{II} & A_{IB} \\ A_{IB}^T & A_{BB} \end{bmatrix} \begin{bmatrix} u_I \\ u_B \end{bmatrix} = \begin{bmatrix} \Omega - \Theta \\ -(k - \tilde{k}) \end{bmatrix} \longleftarrow \text{cone angles}$$

Cut through cones, prescribing u along cut

Maps are **seamless** by construction

Seamless Cone Parameterization

Cone singularities in BFF

Solve *Yamabe Problem* with modified source term

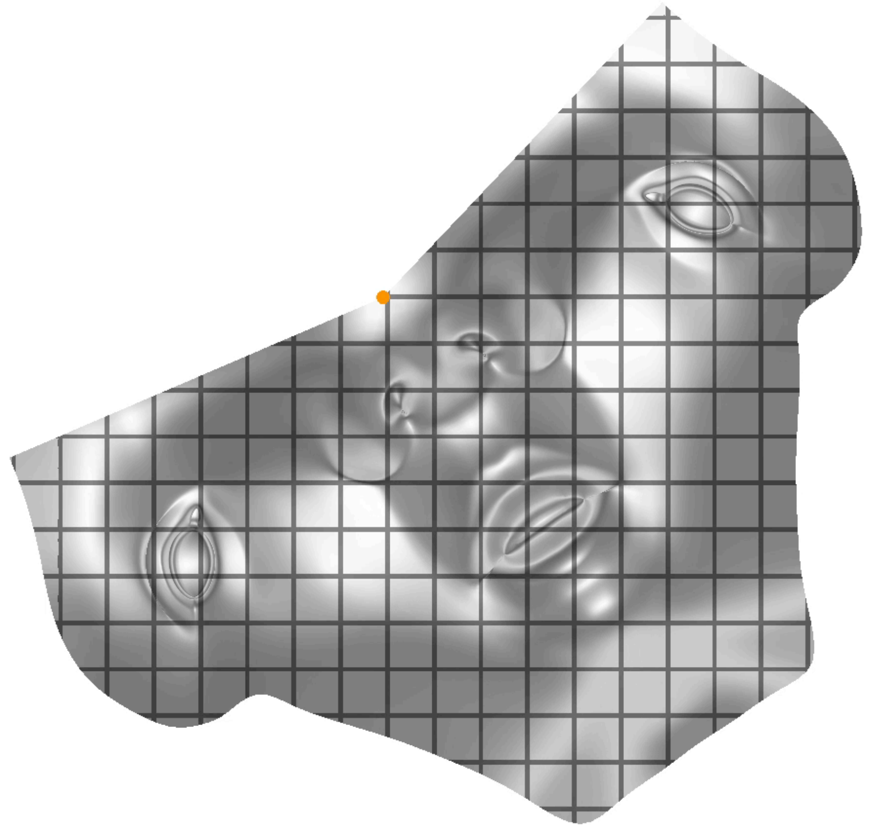
$$\begin{bmatrix} A_{II} & A_{IB} \\ A_{IB}^T & A_{BB} \end{bmatrix} \begin{bmatrix} u_I \\ u_B \end{bmatrix} = \begin{bmatrix} \Omega - \boxed{\Theta} \\ -(k - \tilde{k}) \end{bmatrix}$$

cone angles

Cut through cones, prescribing u along cut

Maps are **seamless** by construction

Allow interactive editing of cone angles



Seamless Cone Parameterization

Cone singularities in BFF

Solve *Yamabe Problem* with modified source term

$$\begin{bmatrix} A_{II} & A_{IB} \\ A_{IB}^T & A_{BB} \end{bmatrix} \begin{bmatrix} u_I \\ u_B \end{bmatrix} = \begin{bmatrix} \Omega - \Theta \\ -(k - \tilde{k}) \end{bmatrix}$$

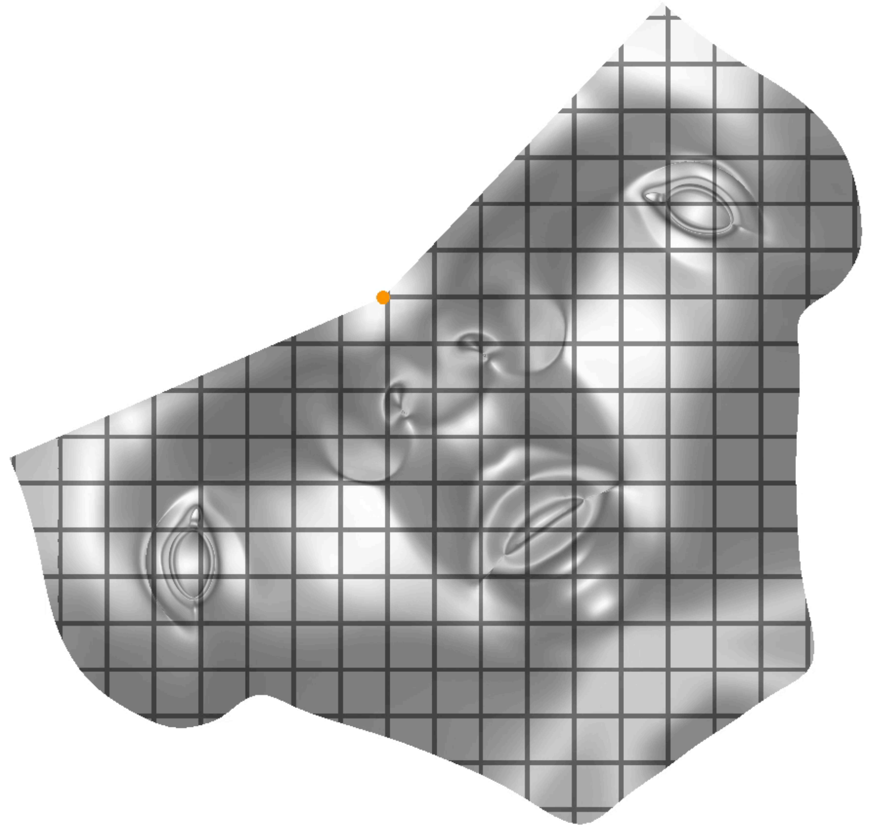
cone angles

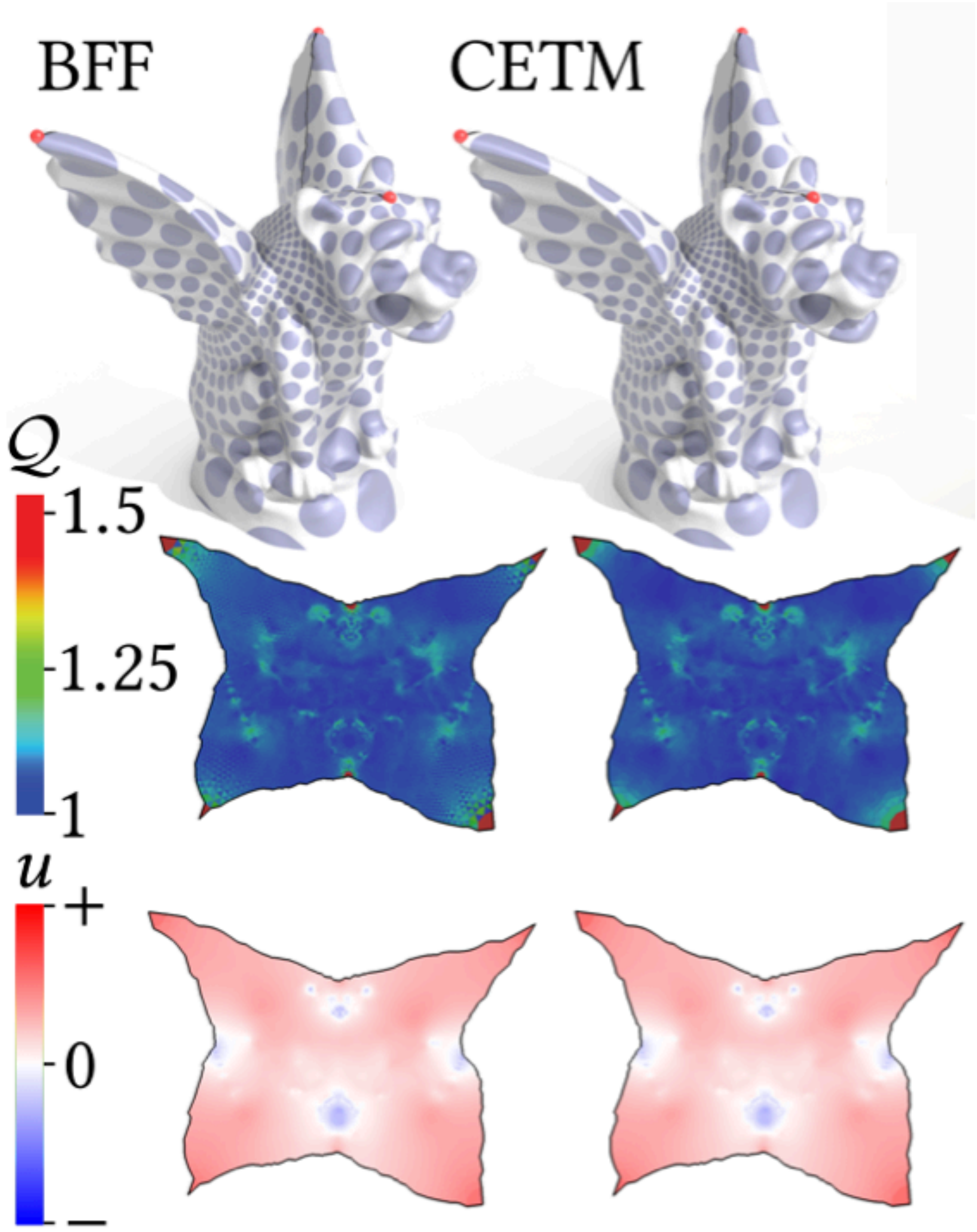
Cut through cones, prescribing u along cut

Maps are **seamless** by construction

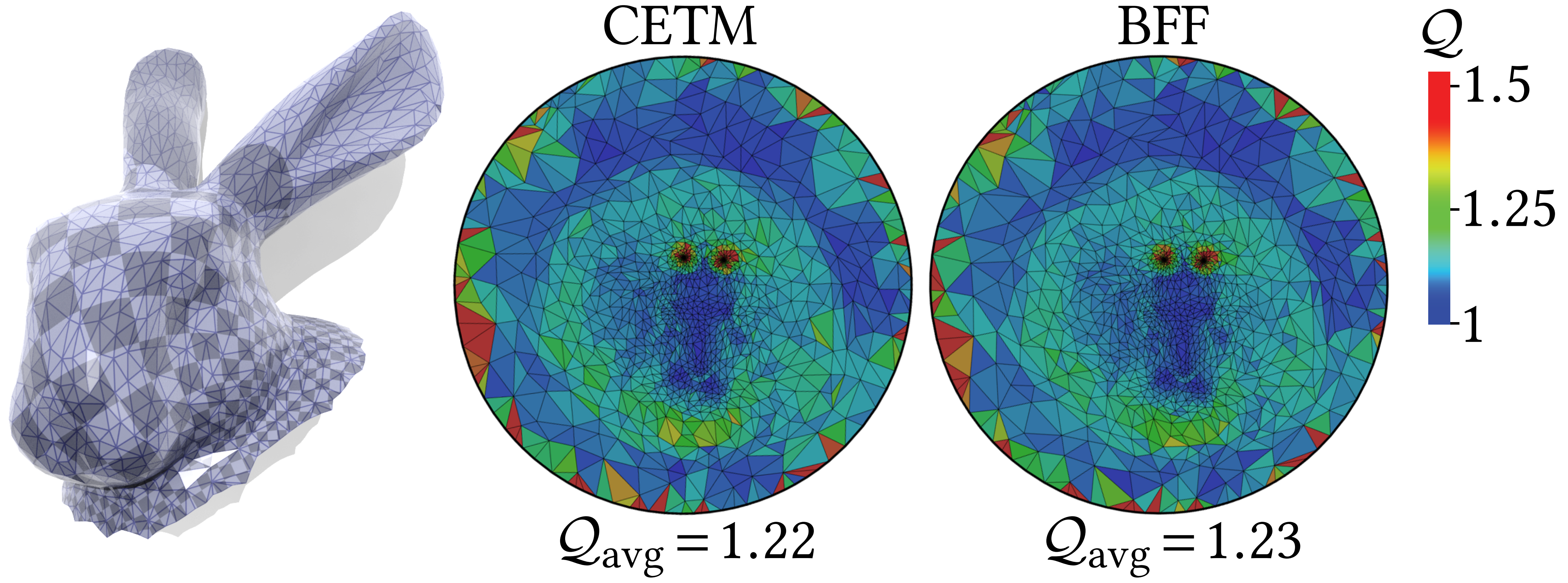
Allow interactive editing of cone angles

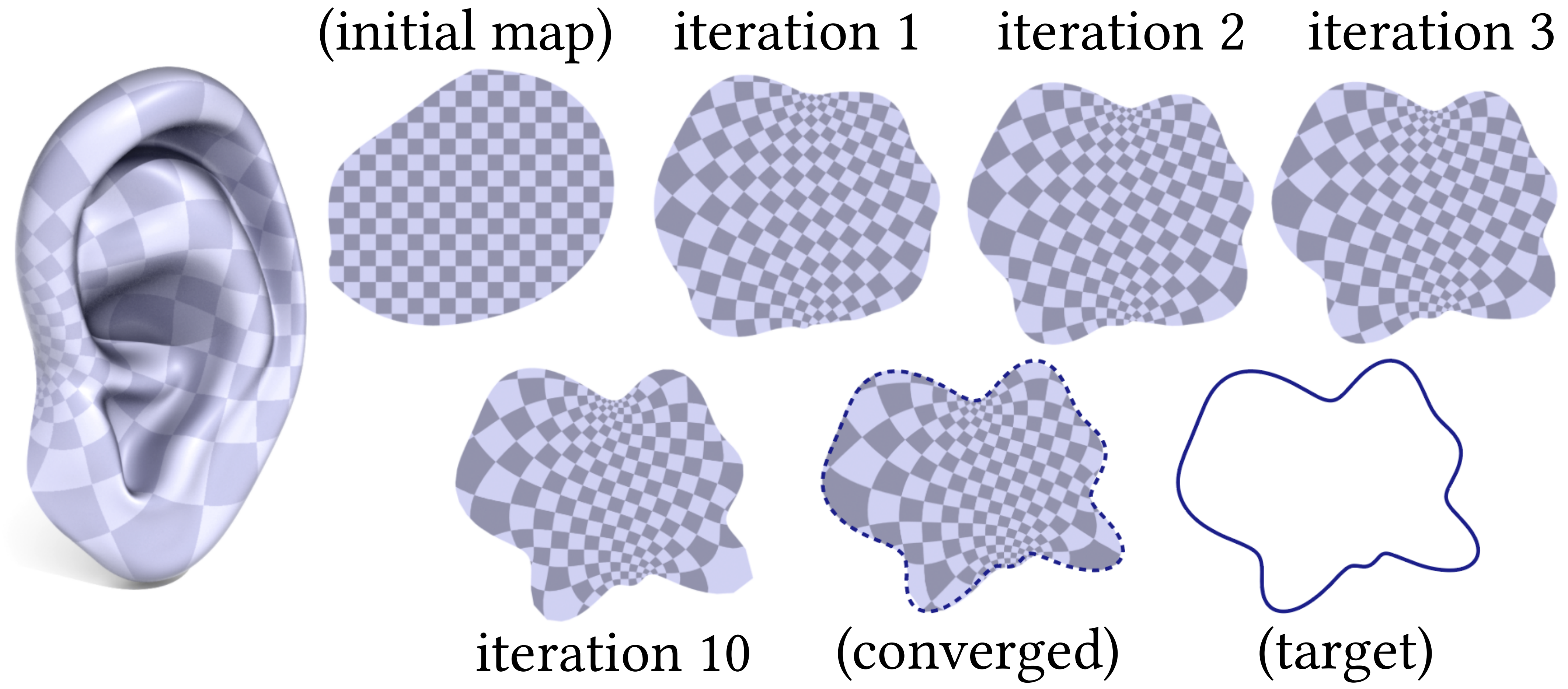
Results indistinguishable from CETM





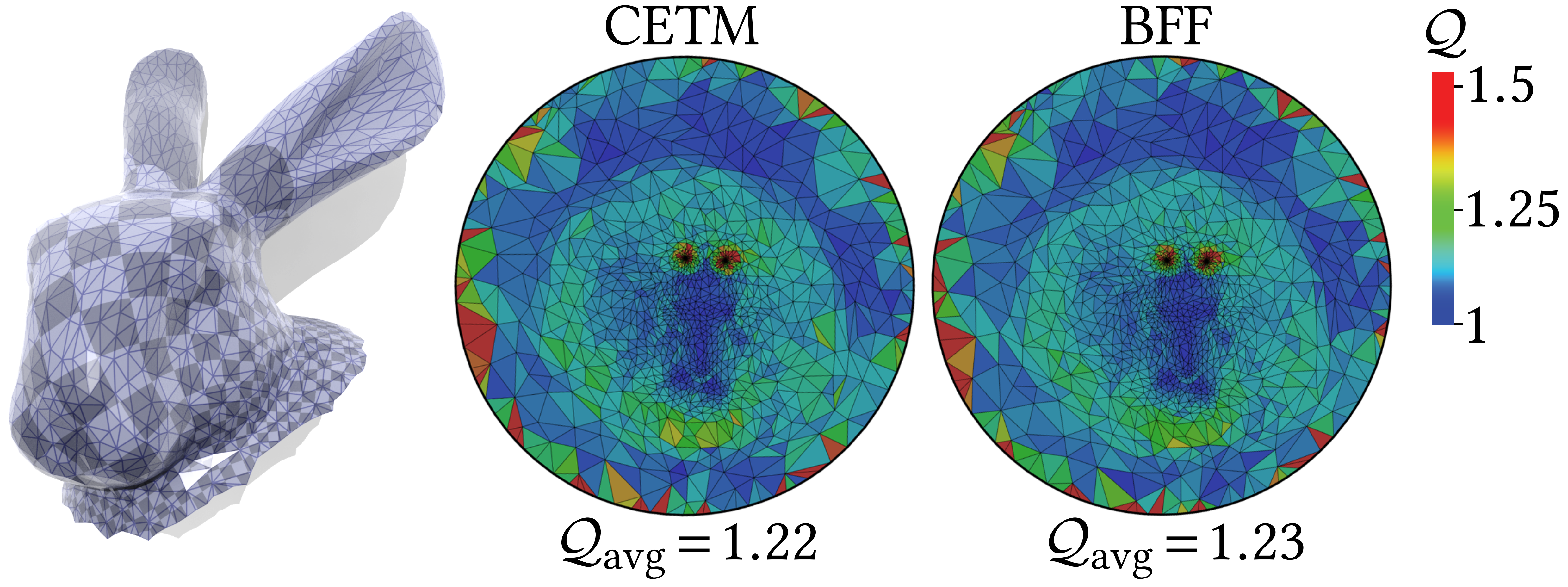
Uniformization and Arbitrary Target Shapes

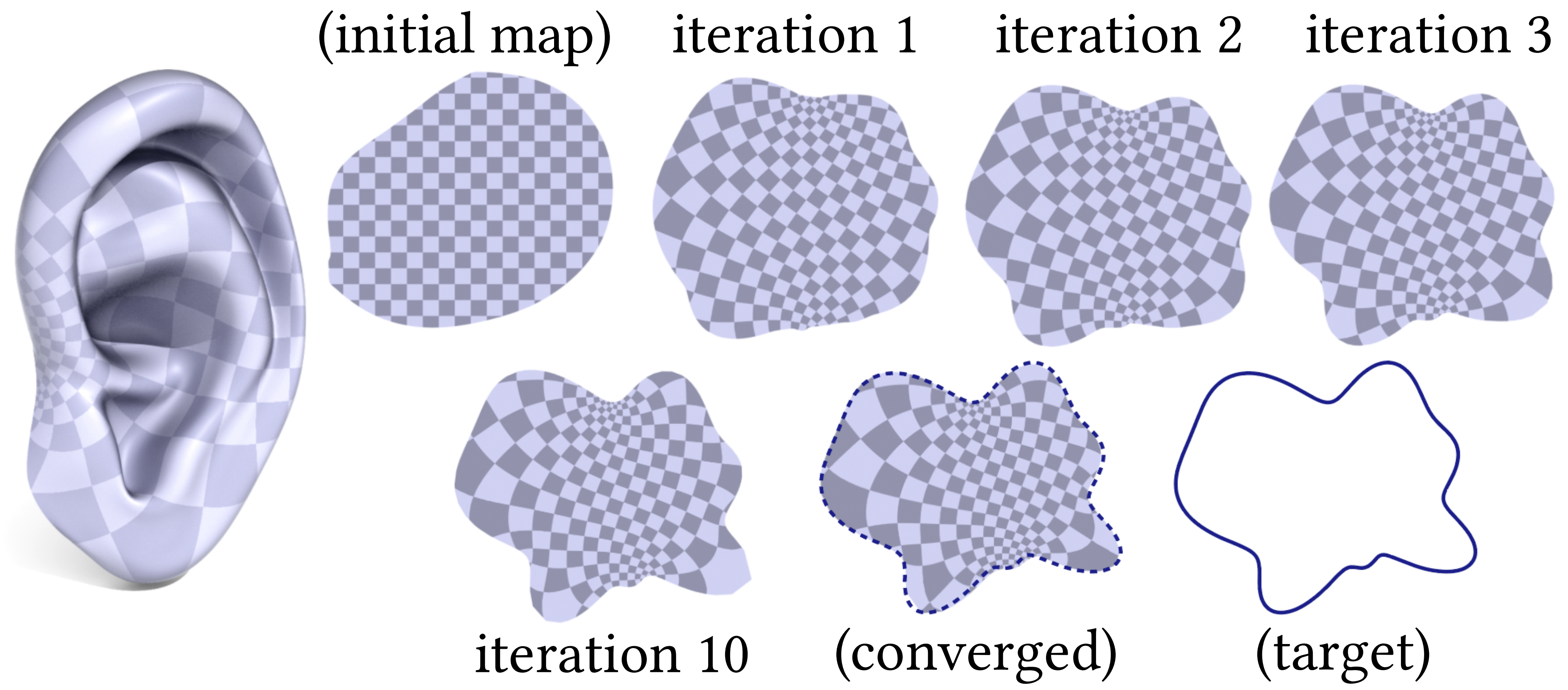




Iterative procedure converges in fewer than 10 iterations

Uniformization and Arbitrary Target Shapes

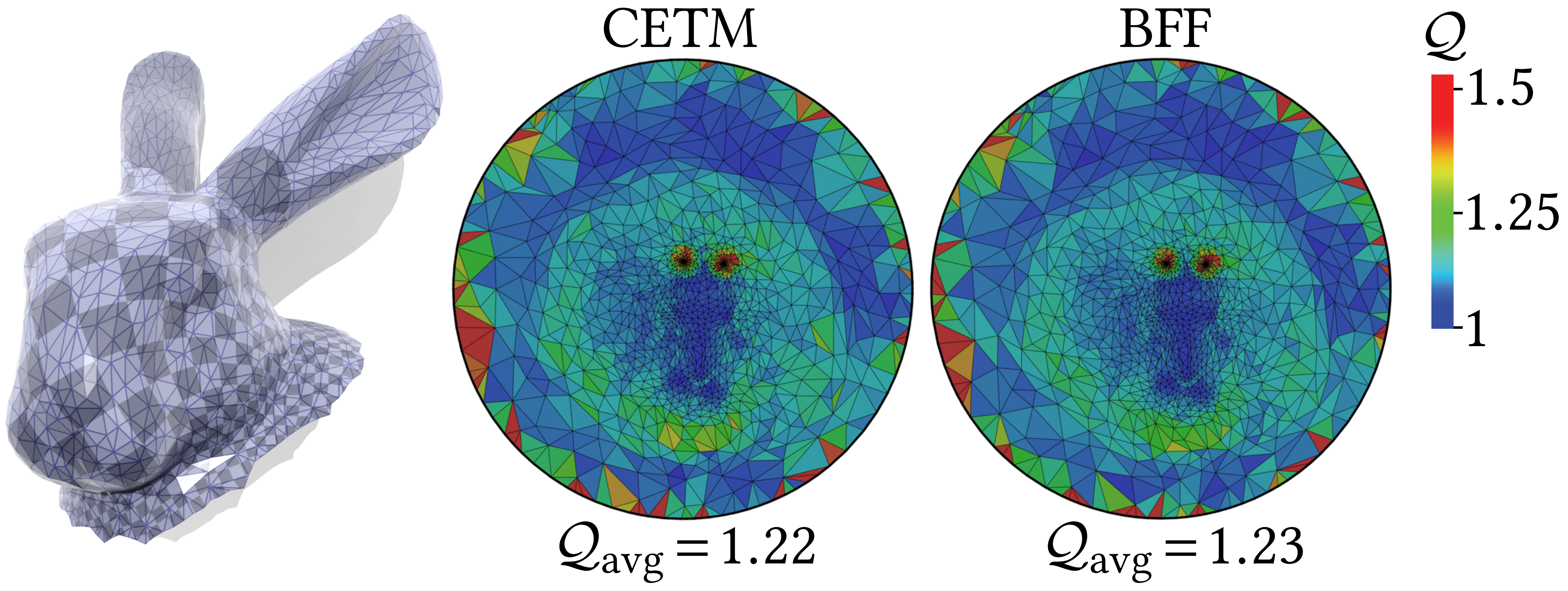


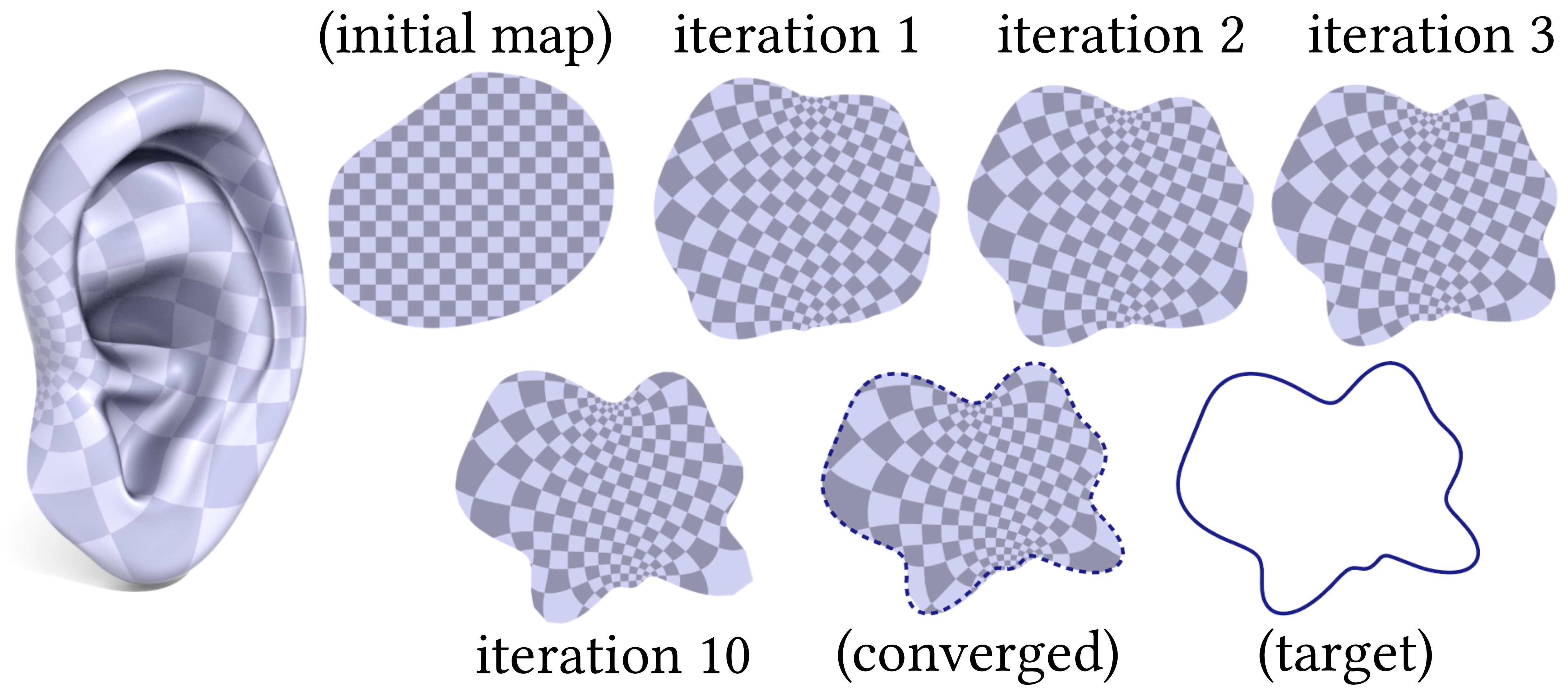


Iterative procedure converges in fewer than 10 iterations

Prescribing exterior angles does not work

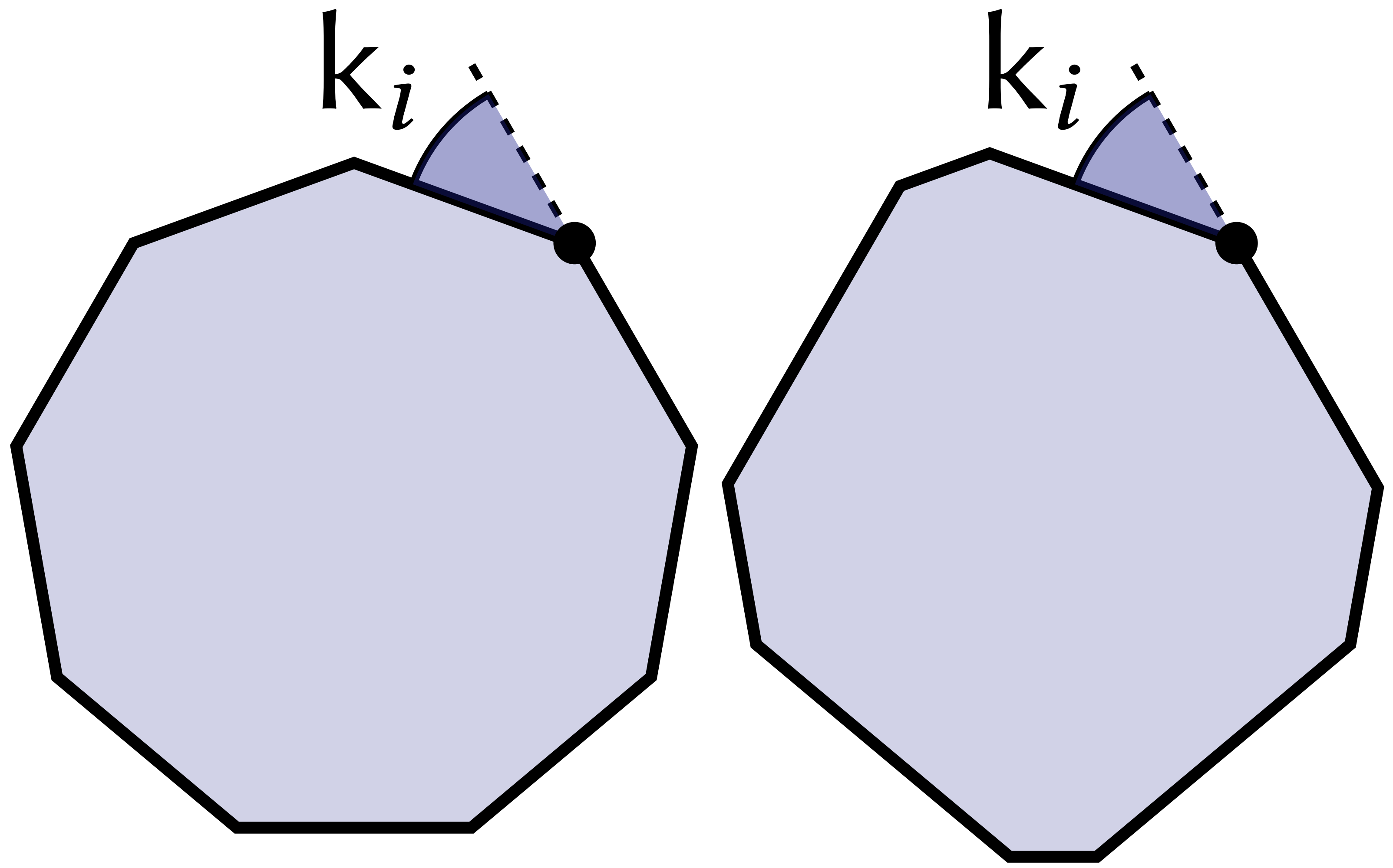
Uniformization and Arbitrary Target Shapes



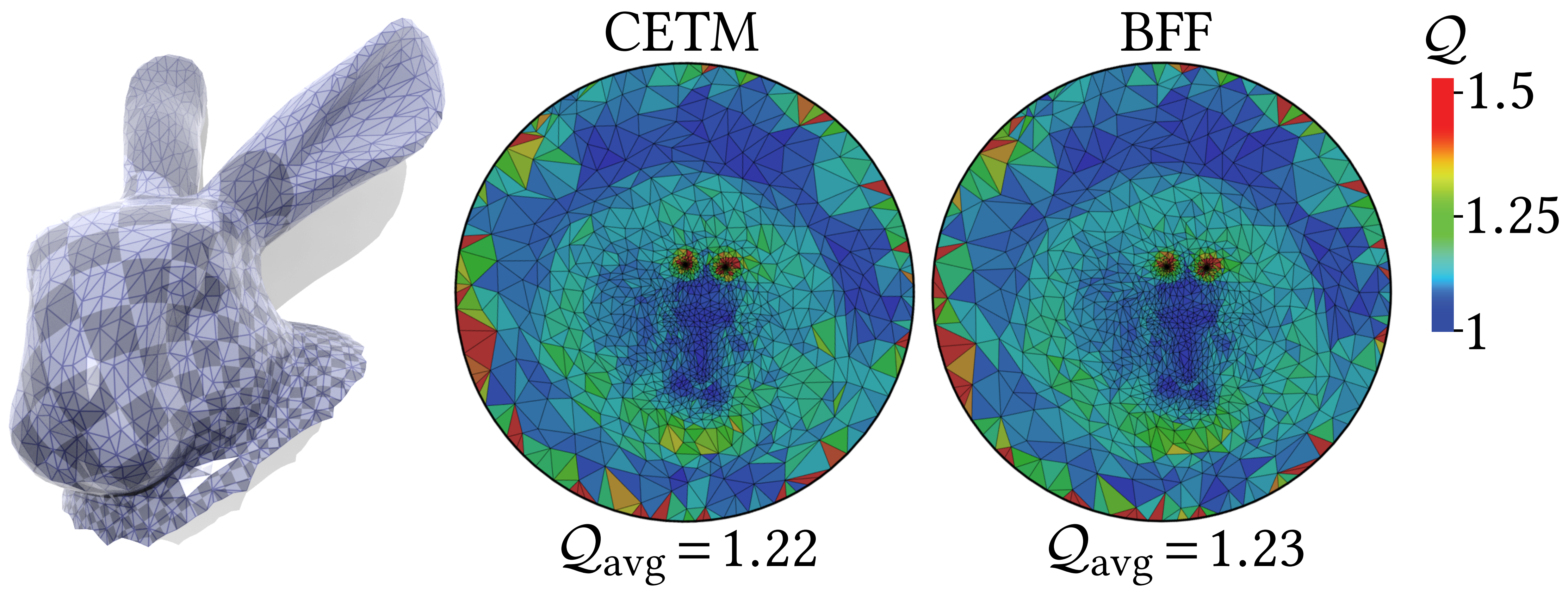


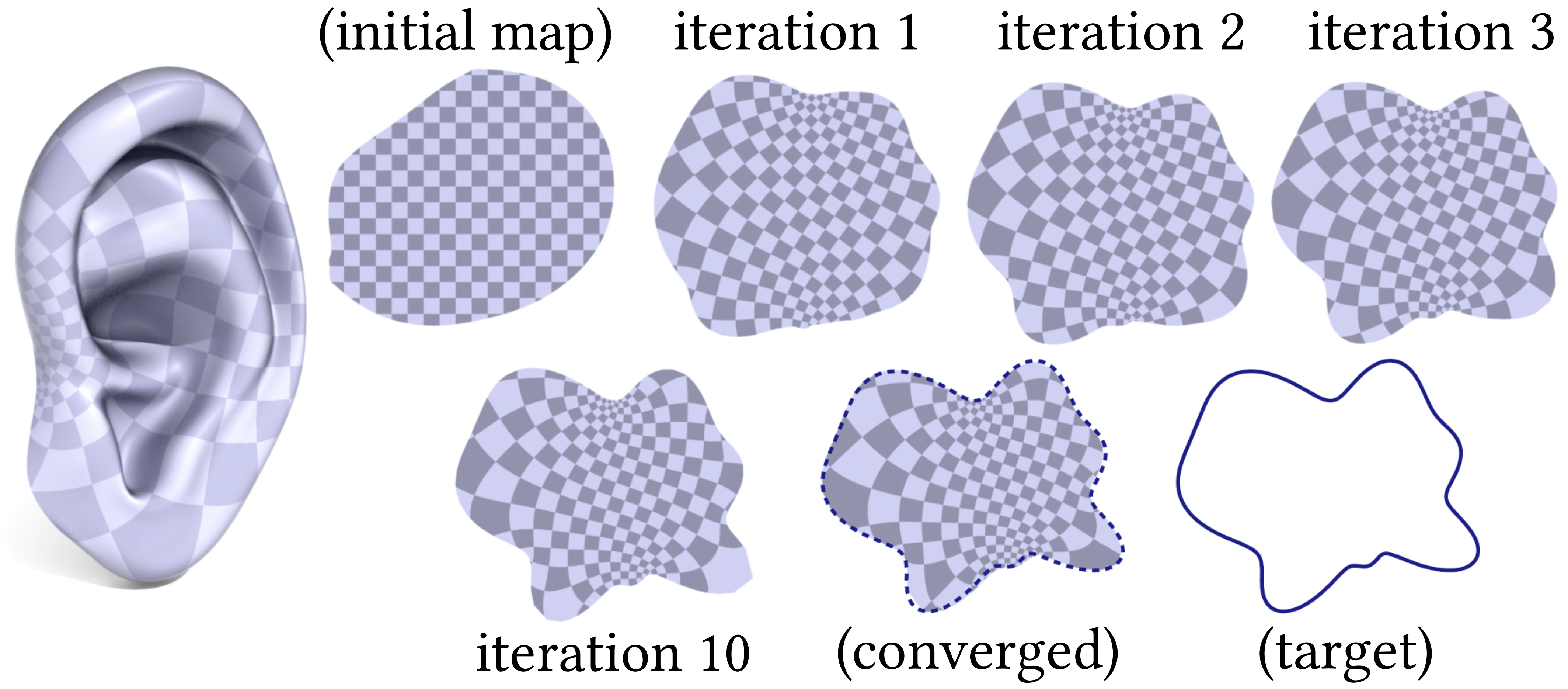
Iterative procedure converges in fewer than 10 iterations

Prescribing exterior angles does not work



Uniformization and Arbitrary Target Shapes

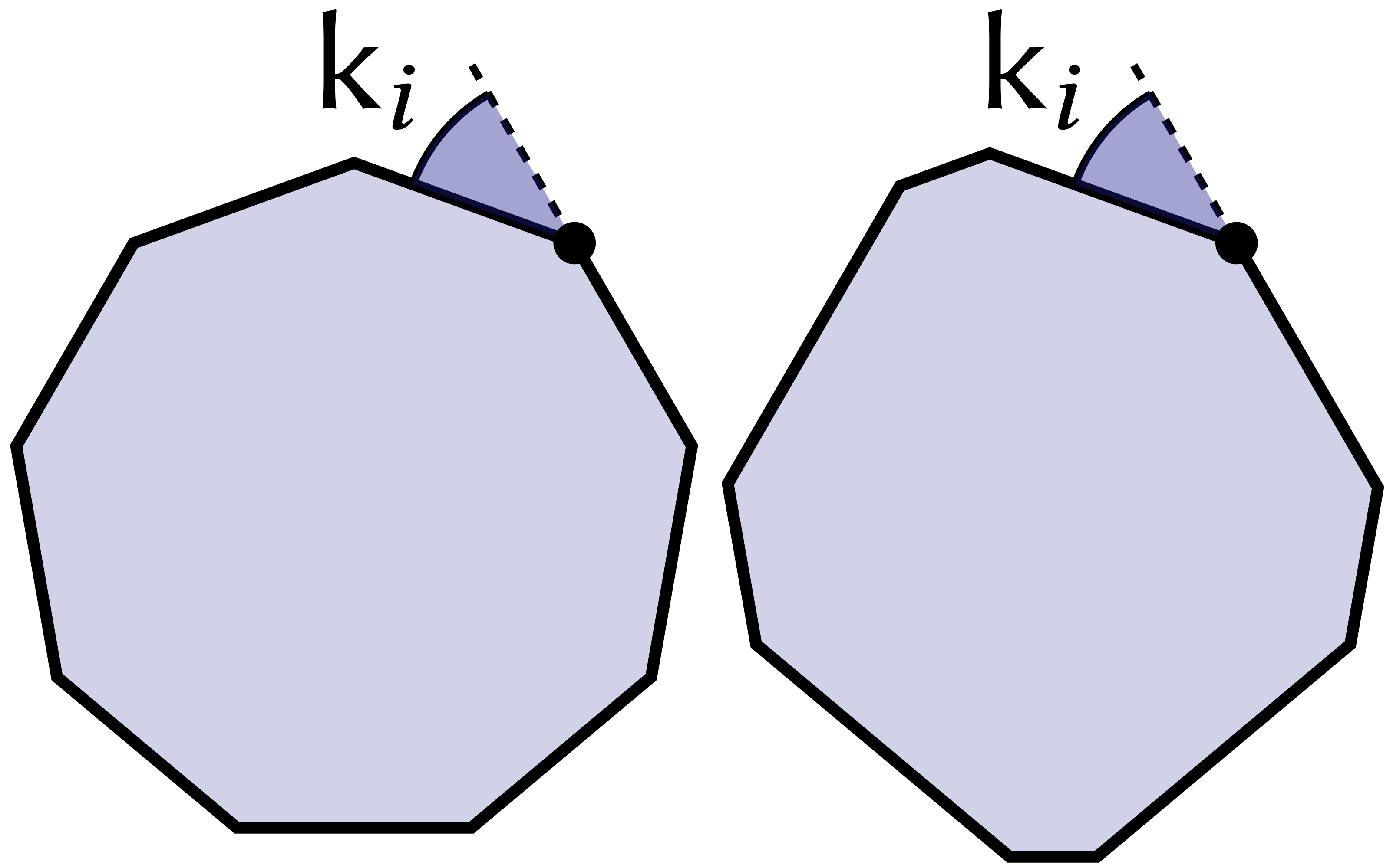




Iterative procedure converges in fewer than 10 iterations

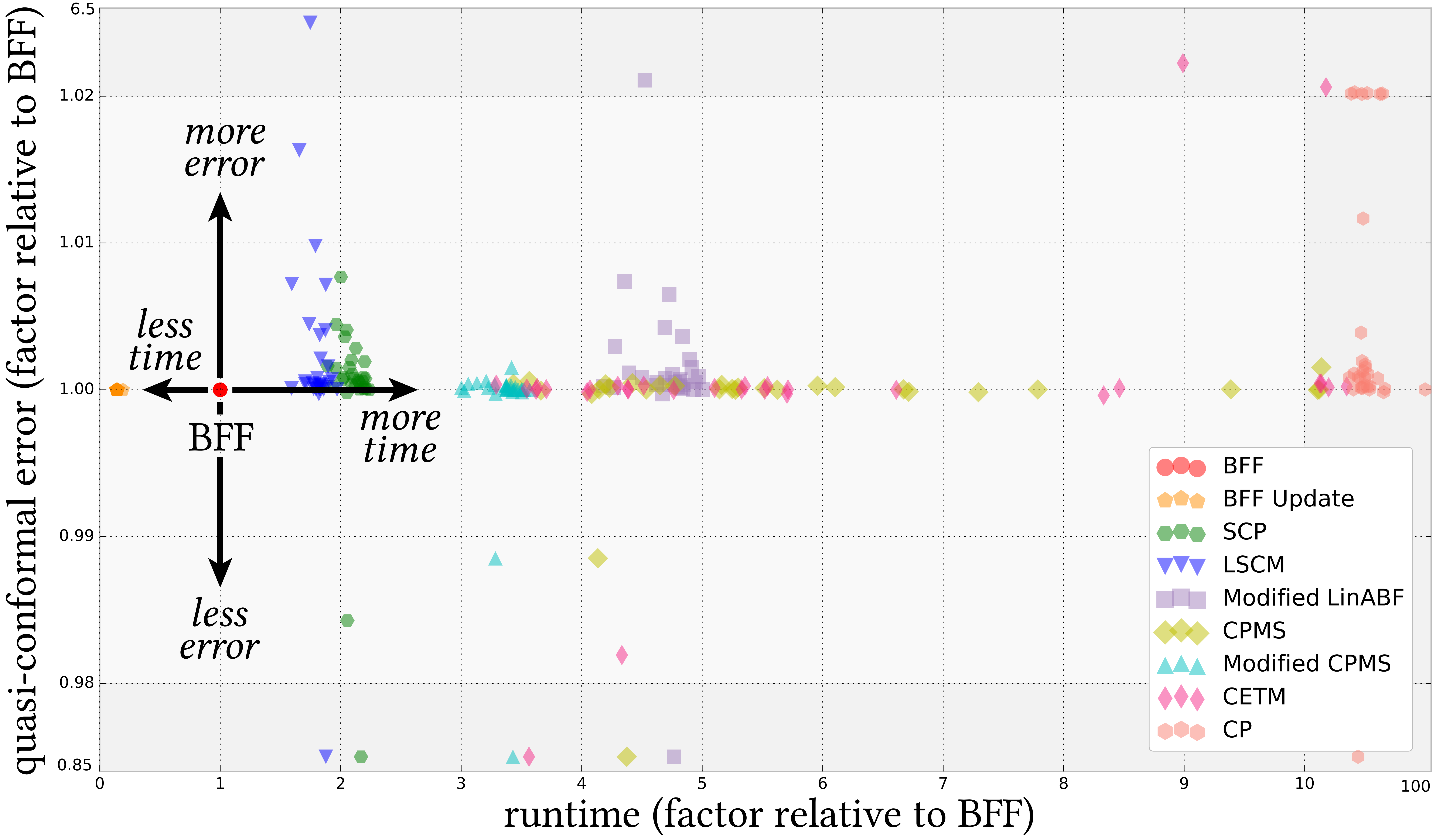
Prescribing exterior angles does not work

To uniquely prescribe target shape, need to control change in angle *per unit length*



EVALUATION

Performance



Fast Computation

Single Sparse Cholesky Factorization

$$\begin{bmatrix} A_{II} & A_{IB} \\ A_{IB}^T & A_{BB} \end{bmatrix} = \begin{bmatrix} L_{II} & 0 \\ L_{BI} & L_{BB} \end{bmatrix} \begin{bmatrix} L_{II}^T & L_{BI}^T \\ 0 & L_{BB}^T \end{bmatrix}$$

most expensive step in entire algorithm

Fast Computation

Single Sparse Cholesky Factorization

$$\begin{bmatrix} A_{II} & A_{IB} \\ A_{IB}^T & A_{BB} \end{bmatrix} = \begin{bmatrix} L_{II} & 0 \\ L_{BI} & L_{BB} \end{bmatrix} \begin{bmatrix} L_{II}^T & L_{BI}^T \\ 0 & L_{BB}^T \end{bmatrix}$$

most expensive step in entire algorithm

$$\implies A_{II} = L_{II} L_{II}^T$$

“for free”

Fast Computation

Single Sparse Cholesky Factorization

$$\begin{bmatrix} A_{II} & A_{IB} \\ A_{IB}^T & A_{BB} \end{bmatrix} = \begin{bmatrix} L_{II} & 0 \\ L_{BI} & L_{BB} \end{bmatrix} \begin{bmatrix} L_{II}^T & L_{BI}^T \\ 0 & L_{BB}^T \end{bmatrix}$$

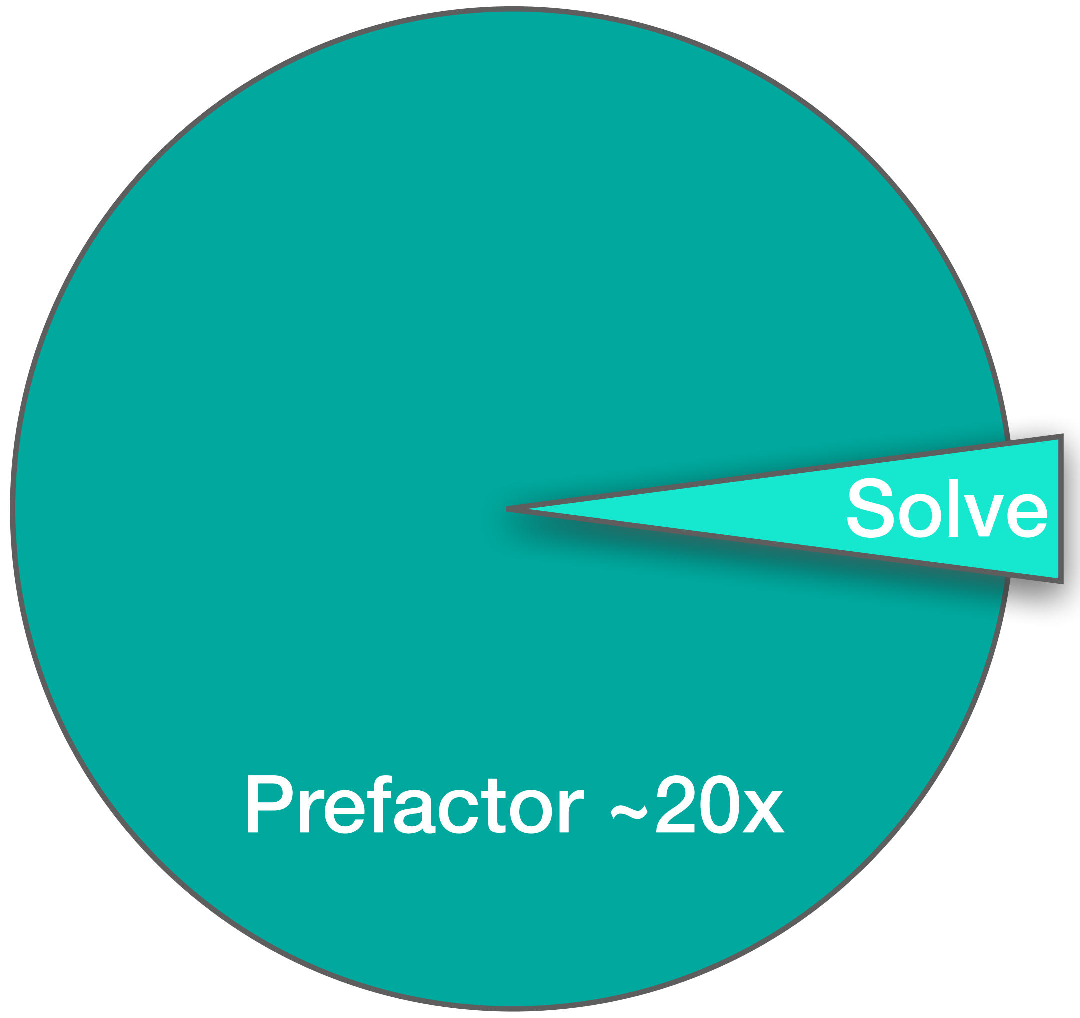
$$\Longrightarrow \quad A_{II} = L_{II} L_{II}^T$$

most expensive step in entire algorithm

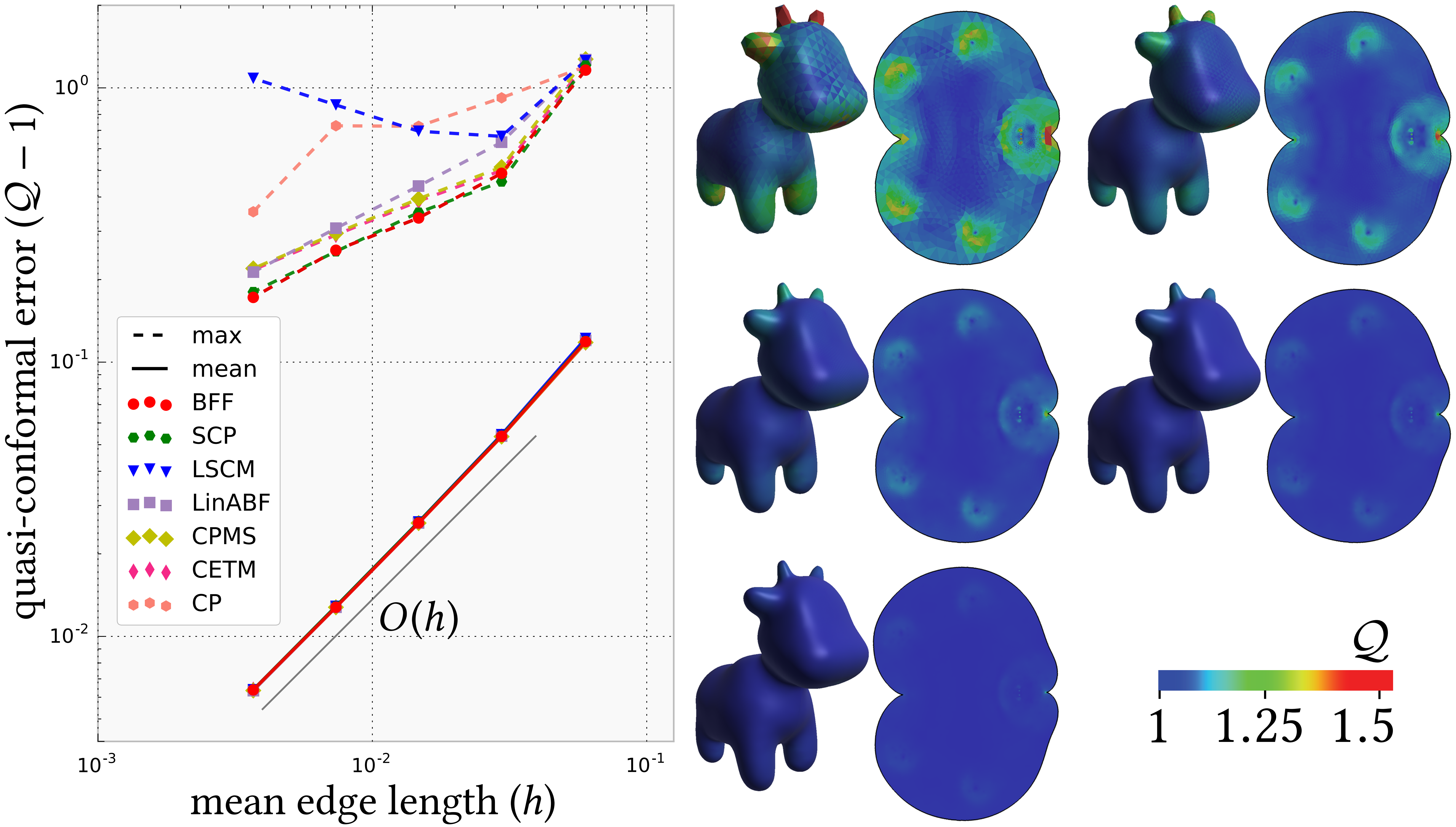
Backsubtitution

$$\begin{aligned} Ax_1 &= b_1 \\ Ax_2 &= b_2 \\ Ax_3 &= b_3 \\ &\vdots \end{aligned}$$

“for free”

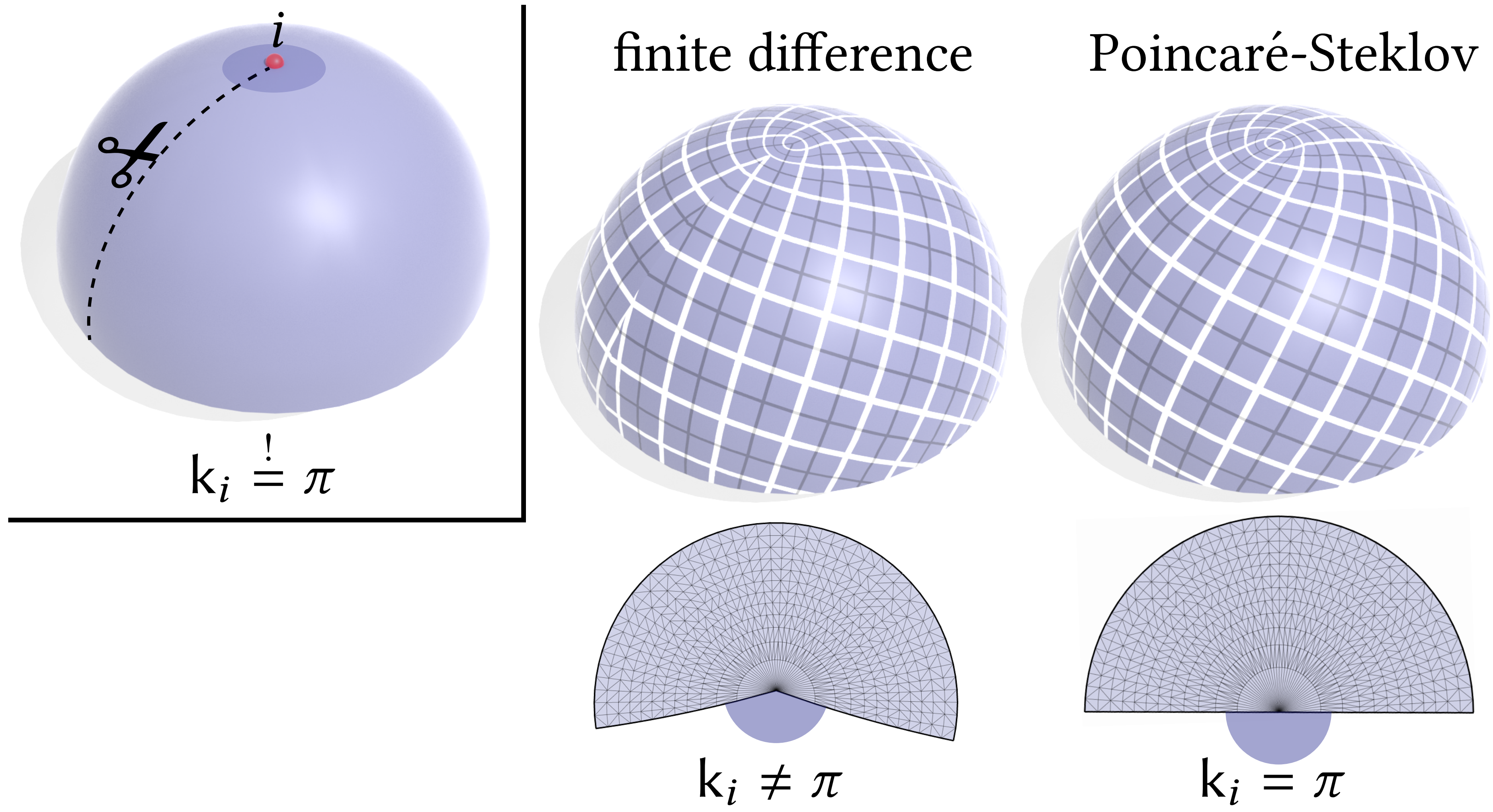


Convergence



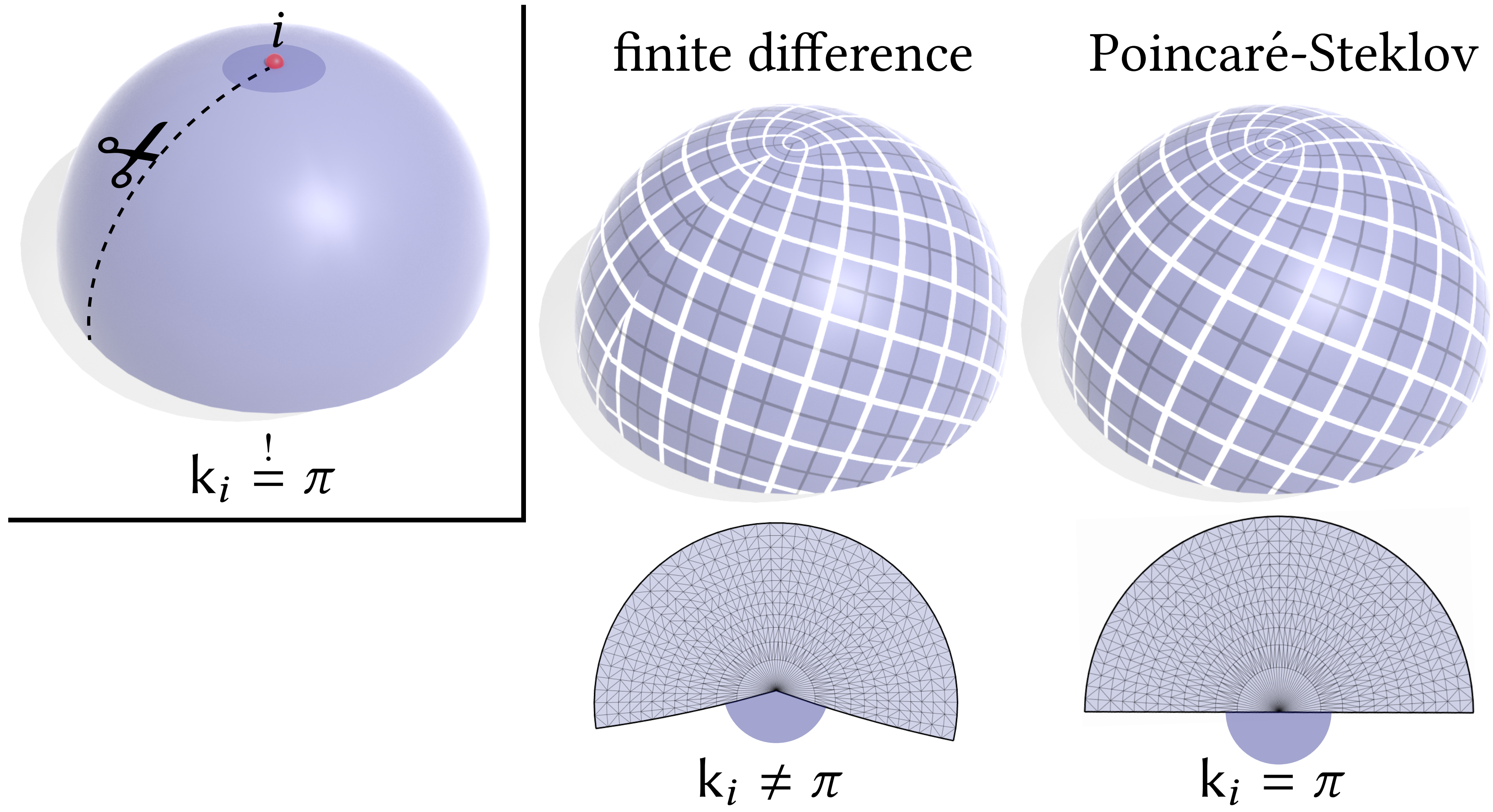
Numerical Robustness

Principled discretization of Poincaré Steklov operators guarantees **exact** integrability of exterior angles

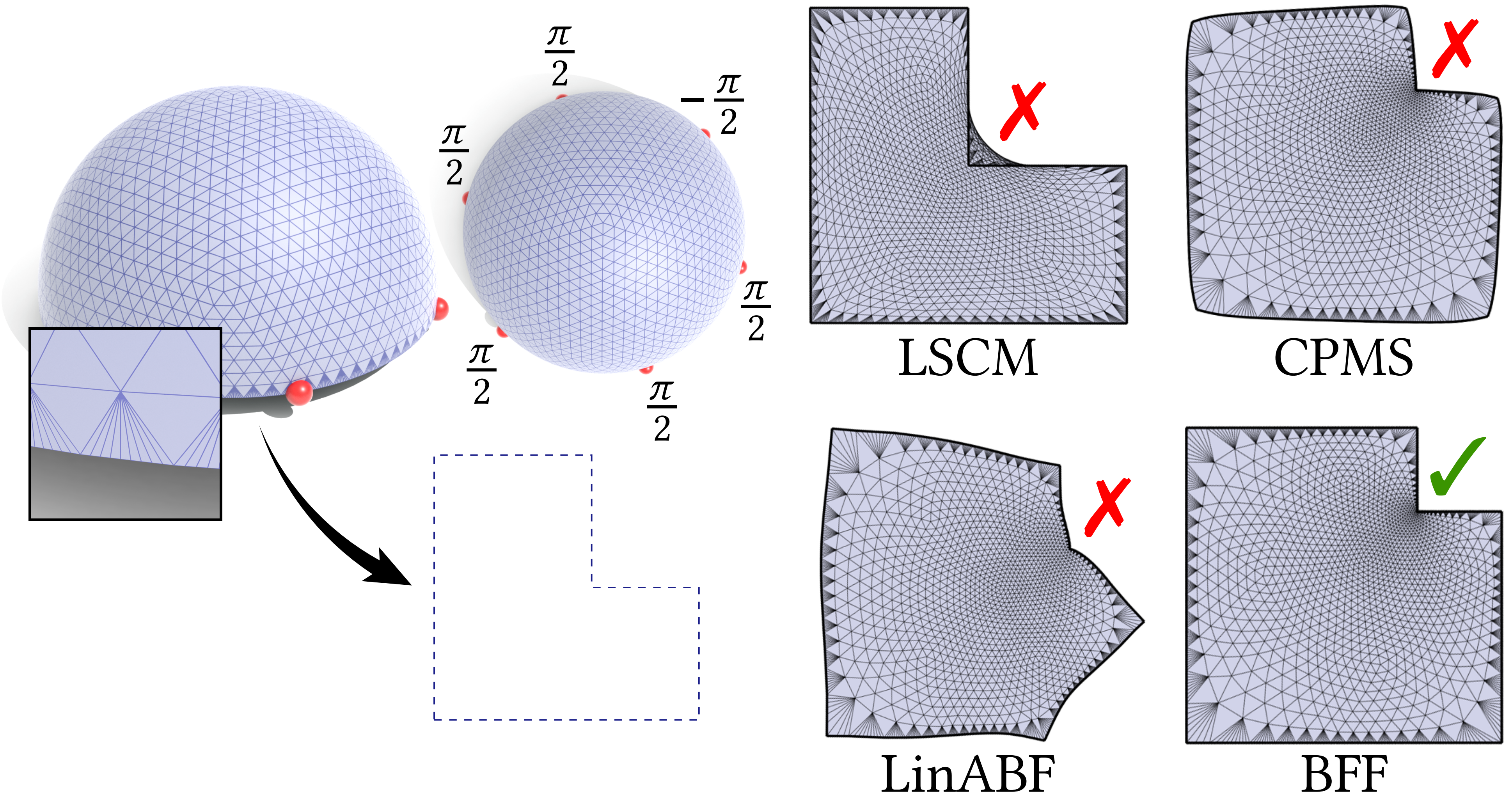


Numerical Robustness

Principled discretization of Poincaré Steklov operators guarantees **exact** integrability of exterior angles



Integrability of edge lengths enforced **only** along boundary



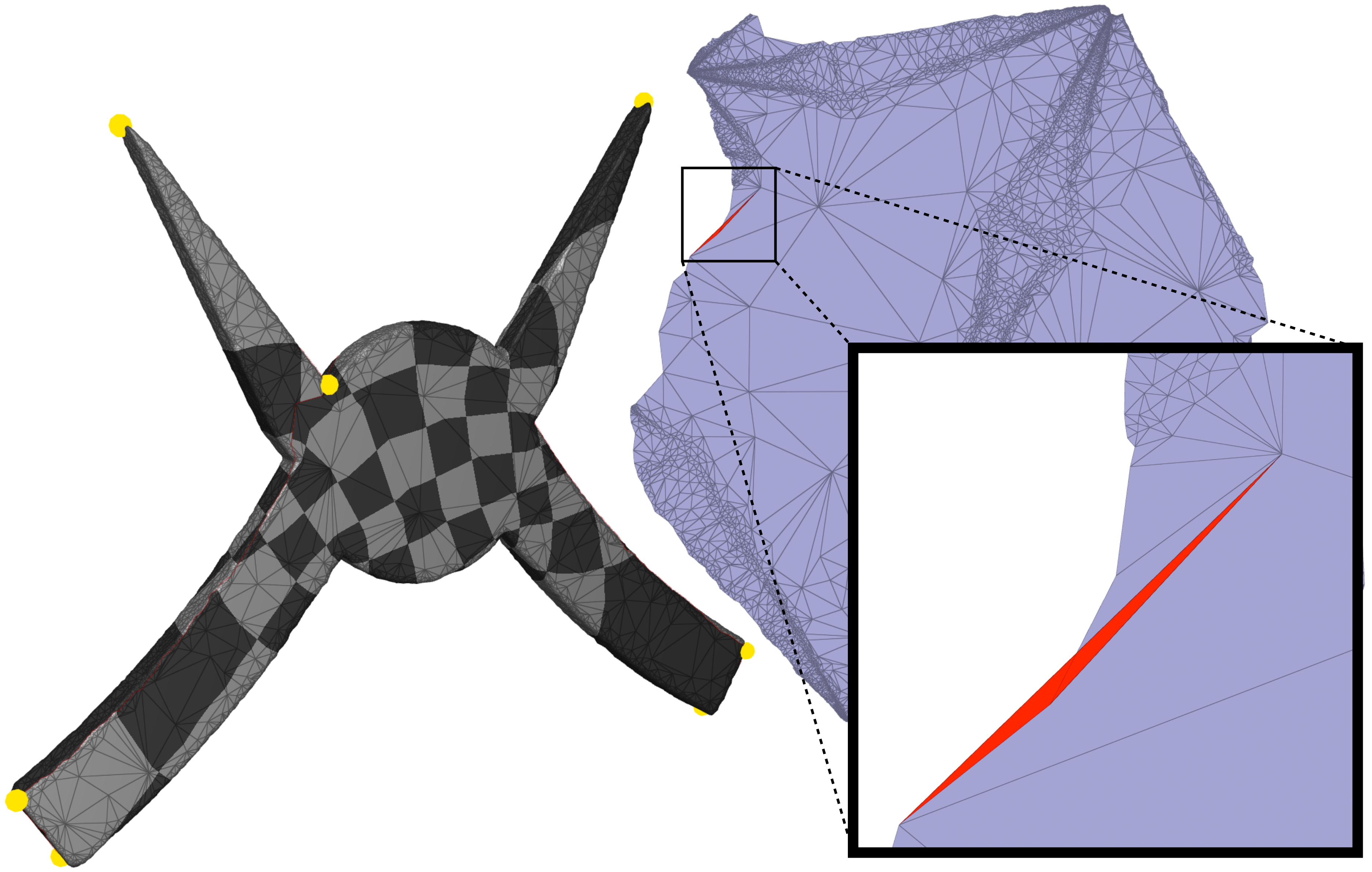
Injectivity (No Flipped Triangles)

BFF provides no guarantees, but maps are usually injective:

Injectivity (No Flipped Triangles)

BFF provides no guarantees, but maps are usually injective:

SHREC: 6/588 meshes; 1-2 flipped triangles

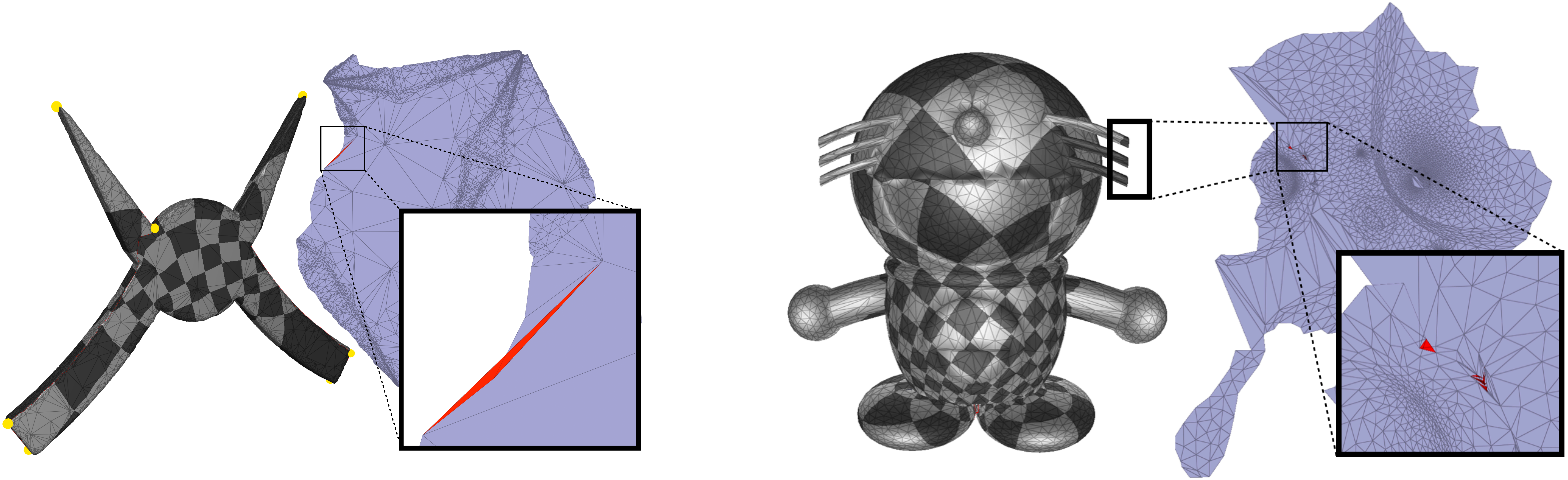


Injectivity (No Flipped Triangles)

BFF provides no guarantees, but maps are usually injective:

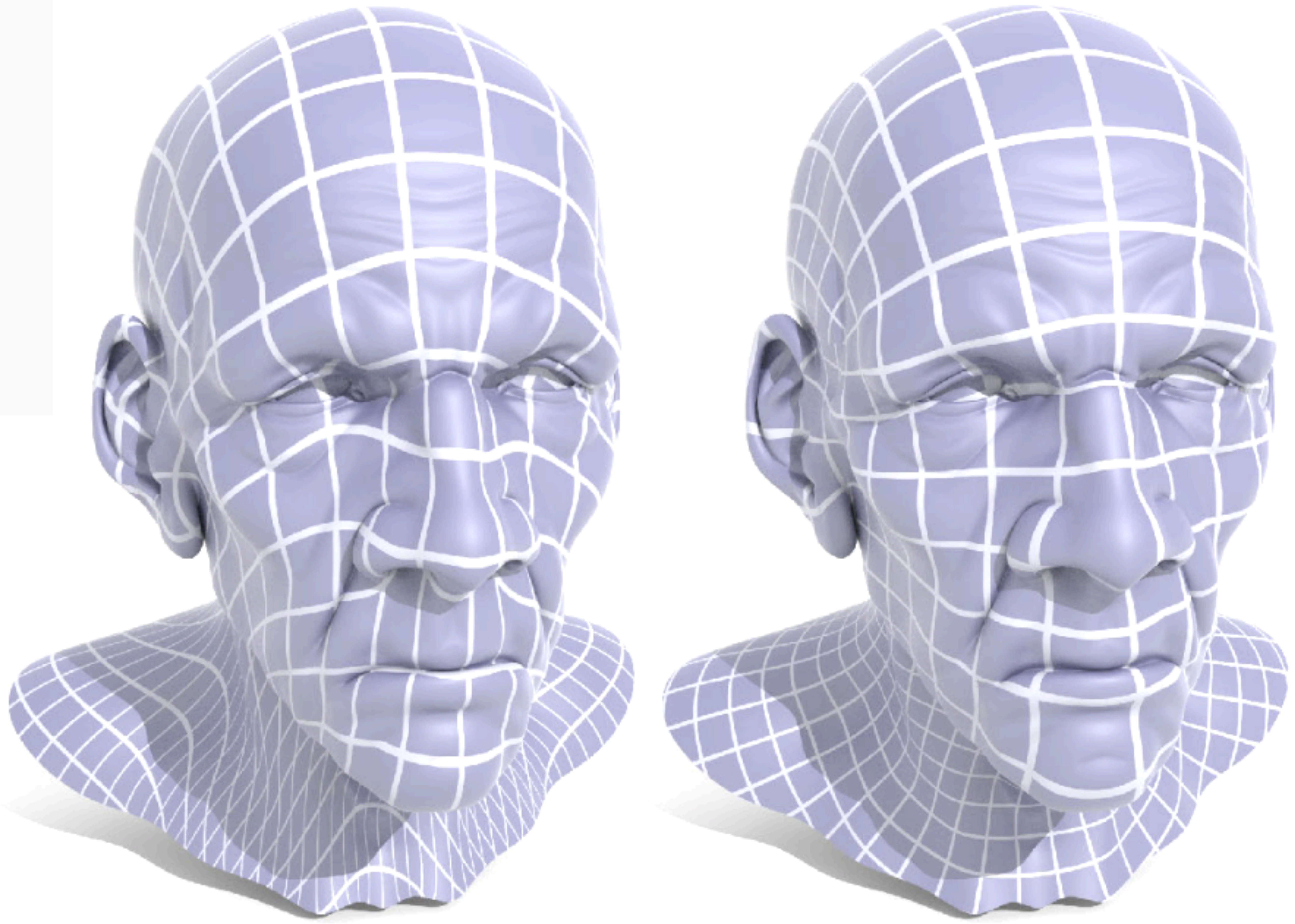
SHREC: 6/588 meshes; 1-2 flipped triangles

Myles & Zorin: 5/116 meshes; > 1% flipped triangles



Price of Guaranteed Injectivity

Editing can be **100’s** of times slower with injective methods



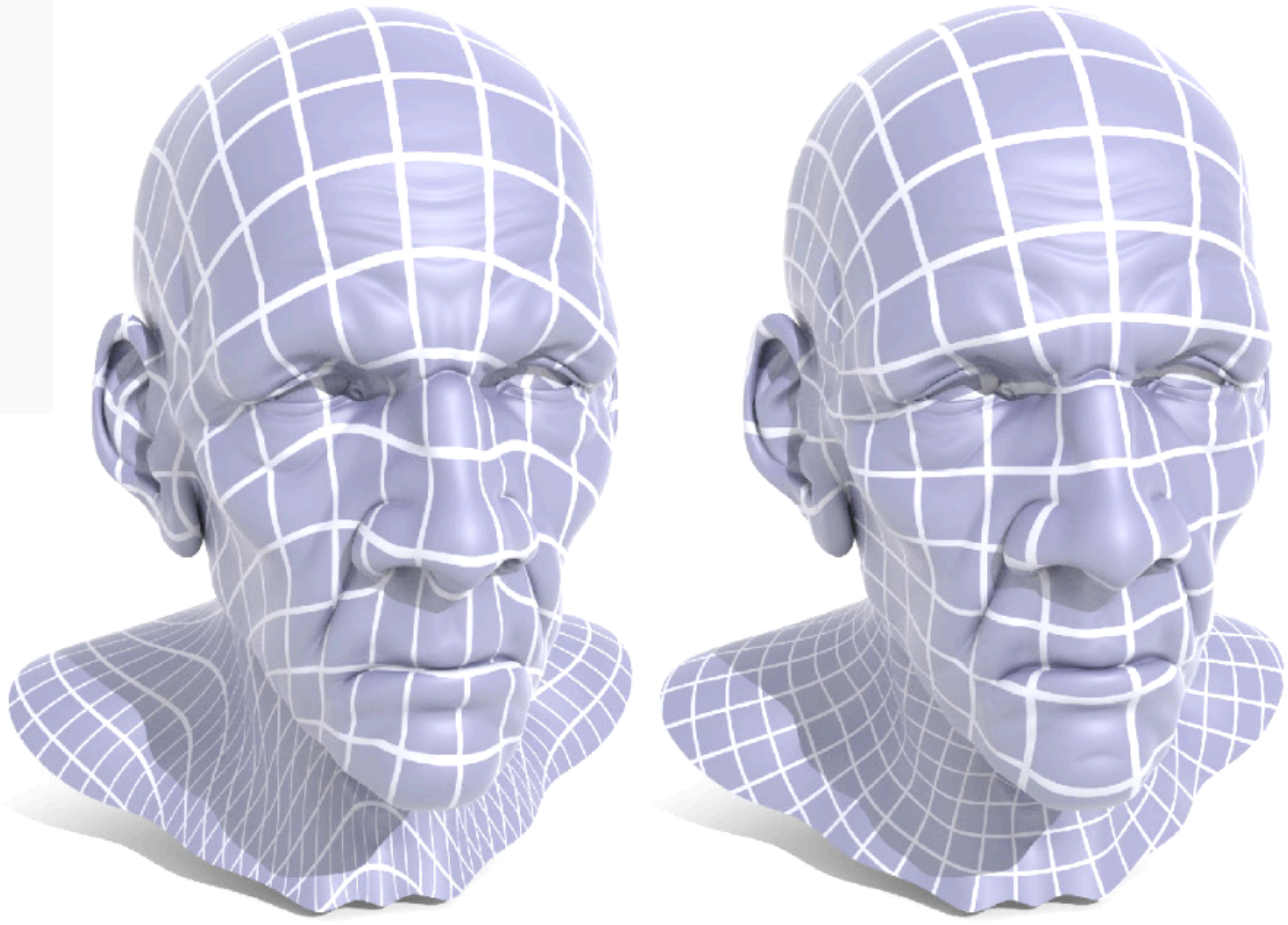
SLIM + PARDISO – 15.9s
[Rabinovich et al 2016]

BFF - 0.12s (*126x faster*)
[Sawhney & Crane 2017]

Price of Guaranteed Injectivity

Editing can be 100’s of times slower with injective methods

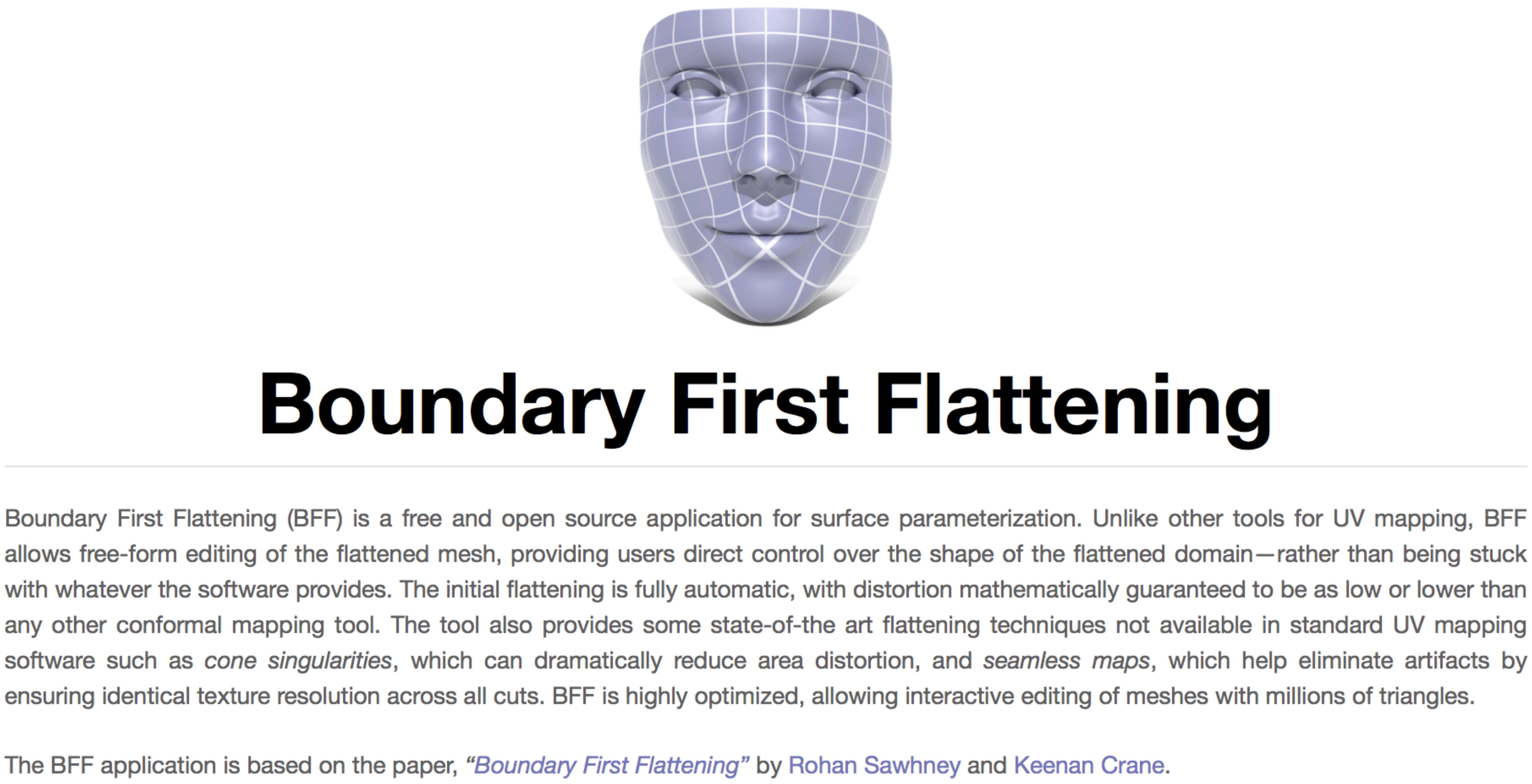
Best of both worlds:
use fast method like BFF
fallback if necessary



SLIM + PARDISO – 15.9s
[Rabinovich et al 2016]

BFF - 0.12s (*126x faster*)
[Sawhney & Crane 2017]

I Want My BFF To Be Your BFF



Boundary First Flattening

Boundary First Flattening (BFF) is a free and open source application for surface parameterization. Unlike other tools for UV mapping, BFF allows free-form editing of the flattened mesh, providing users direct control over the shape of the flattened domain—rather than being stuck with whatever the software provides. The initial flattening is fully automatic, with distortion mathematically guaranteed to be as low or lower than any other conformal mapping tool. The tool also provides some state-of-the art flattening techniques not available in standard UV mapping software such as *cone singularities*, which can dramatically reduce area distortion, and *seamless maps*, which help eliminate artifacts by ensuring identical texture resolution across all cuts. BFF is highly optimized, allowing interactive editing of meshes with millions of triangles.

The BFF application is based on the paper, *"Boundary First Flattening"* by Rohan Sawhney and Keenan Crane.

geometry.cs.cmu.edu/bff

Thanks!

BACKUP SLIDES

2D Shape Editing & Uniformization

Apply 2D conformal deformations to initial flattening?

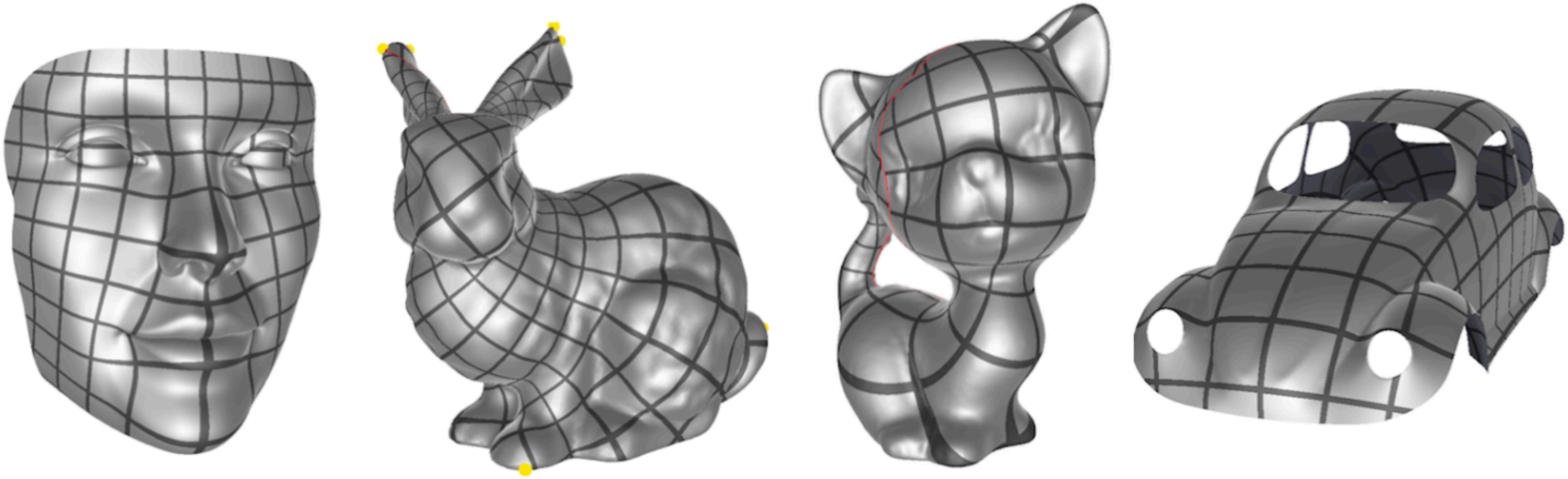
- Piecewise linear conformal maps do not compose
- Composition of methods offers no clear advantage in terms of speed or simplicity

Controllable Conformal Maps for Shape Deformation and Interpolation
[Weber et al 2010]

Topology

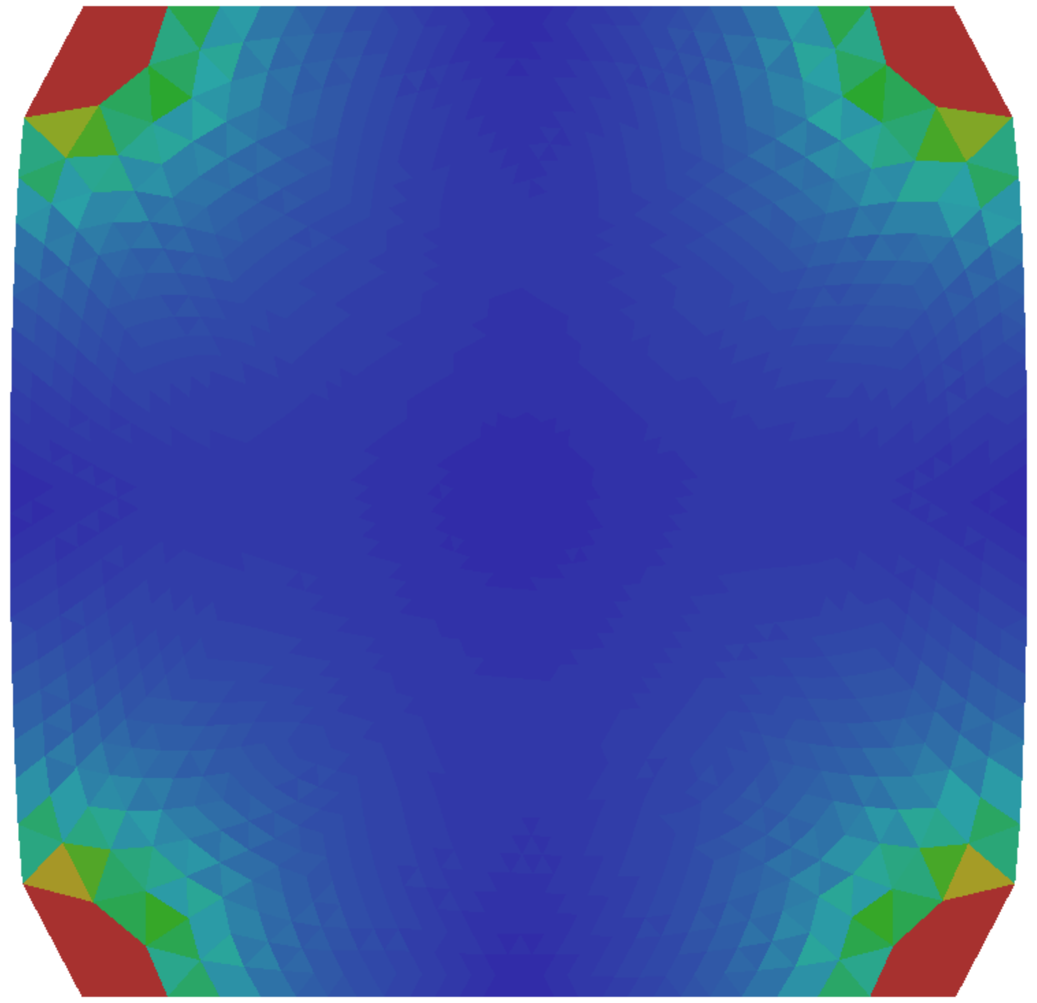
For multiply connected domains like annulus, *Hilbert transform* is not valid

- Fill holes with virtual faces to flatten
- Cut surface into one or more disks

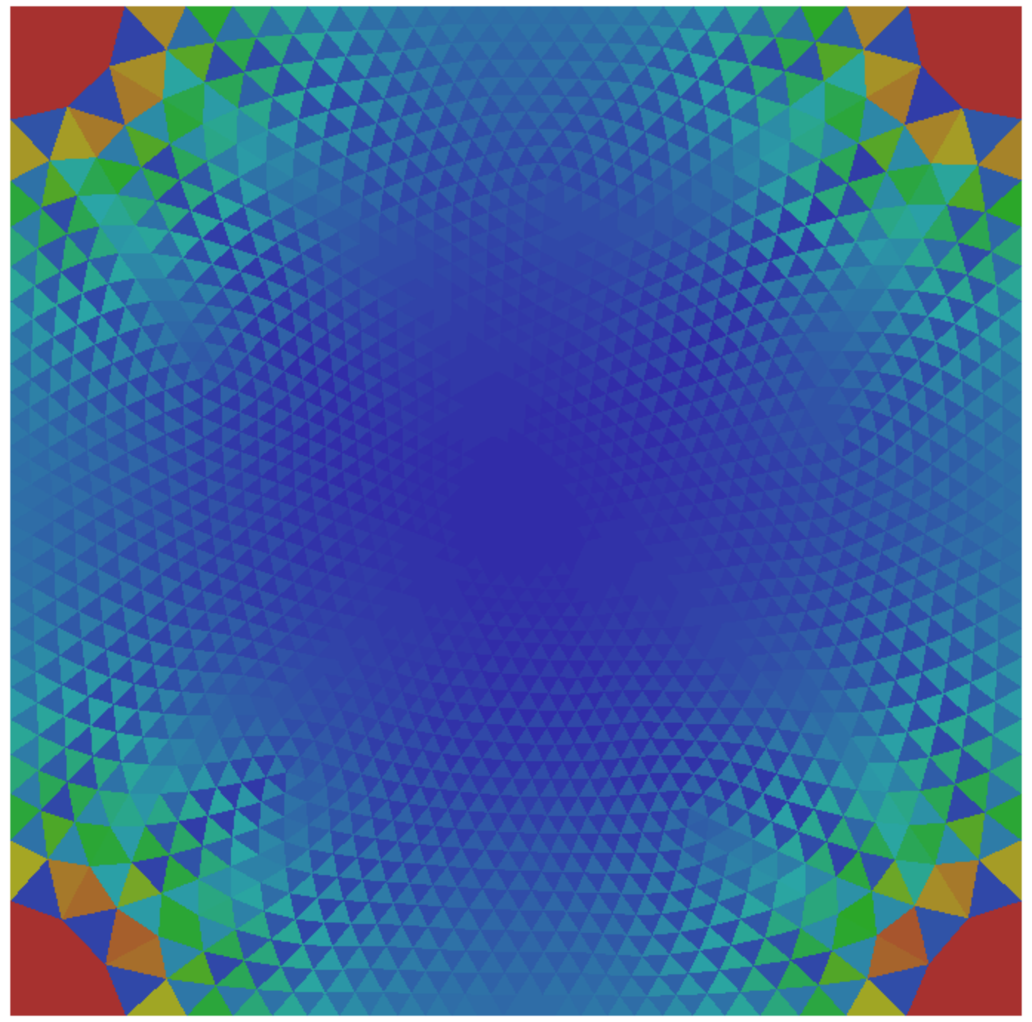


Harmonic vs Holomorphic Extension

Harmonic and *holomophic extension* of $\tilde{\gamma}$ converge to the **same** solution under refinement



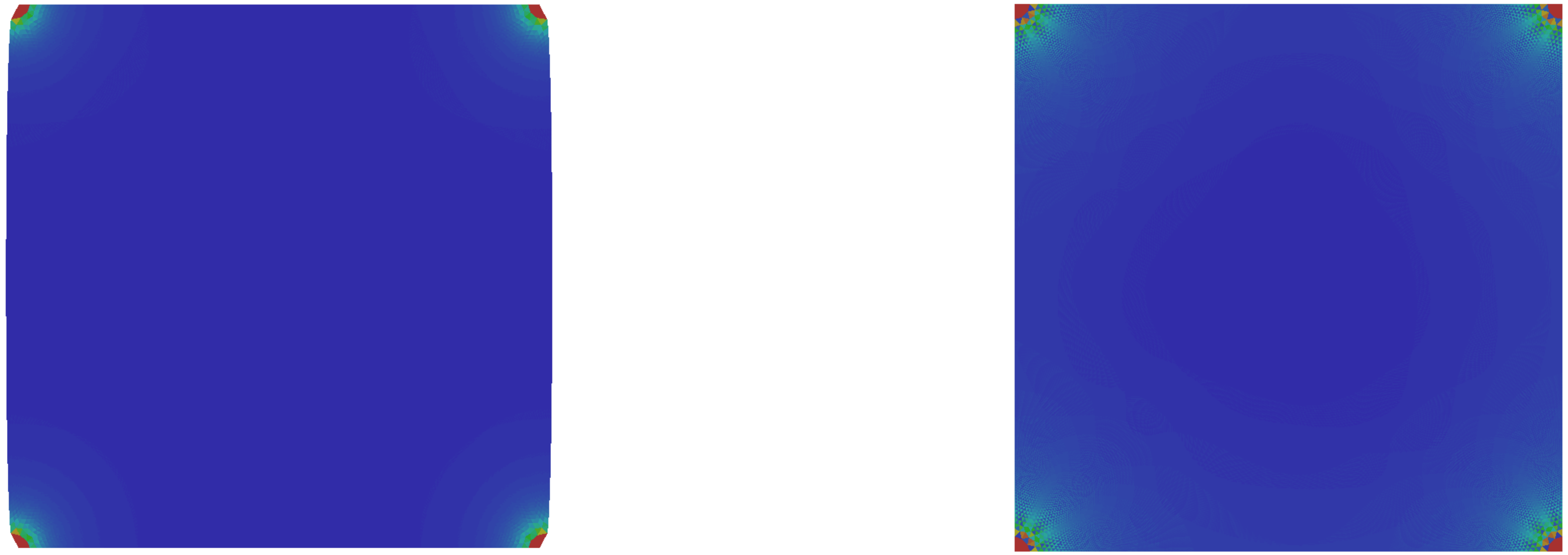
Holomorphic Extension



Harmonic Extension

Harmonic vs Holomorphic Extension

Harmonic and *holomophic extension* of $\tilde{\gamma}$ converge to the **same** solution under refinement



Holomorphic Extension

Harmonic Extension

More Details on Discretizing the Yamabe Problem (1)

Multiply *Yamabe equation* by *dA* and its boundary conditions by *ds*

$$\Delta u = K - e^{2u}\tilde{K} \qquad \Delta u \,.\, dA = K \,.\, dA - \tilde{K}\underbrace{e^{2u} \,.\, dA}_{d\tilde{A}} \qquad on \ M$$

$$\Longrightarrow$$

$$\frac{\partial u}{\partial n} = \kappa - e^{u}\tilde{\kappa} \qquad \frac{\partial u}{\partial n} \,.\, ds = \kappa \,.\, ds - \underbrace{\tilde{\kappa}e^{u} \,.\, ds}_{d\tilde{s}} \qquad on \ \partial M$$

More Details on Discretizing the Yamabe Problem (2)

Integrate over *dual volumes*

$$\Delta u \,.\, dA = K \,.\, dA - \tilde{K} \,.\, d\tilde{A} \qquad on\ M$$

$$\frac{\partial u}{\partial n} \,.\, ds = \kappa \,.\, ds - \tilde{\kappa} d\tilde{s} \qquad on\ \partial M$$

More Details on Discretizing the Yamabe Problem (2)

Integrate over *dual volumes*

$$\Delta u \,.\, dA = K \,.\, dA - \tilde{K} \,.\, d\tilde{A} \qquad\qquad Au = \Omega - \tilde{\Omega} \qquad on\ M$$

$$\Longrightarrow$$

$$\frac{\partial u}{\partial n} \,.\, ds = \kappa \,.\, ds - \tilde{\kappa} d\tilde{s} \qquad\qquad h = k - \tilde{k} \qquad on\ \partial M$$

Modification to CPMS

High Level Idea:

Employ *Yamabe Equation* to obtain scale information

Seek edge lengths that describe a flat surface via least squares layout

Modification:

Add boundary control with *Cherrier* boundary conditions

Comparison with BFF:

Least Squares layout does not respect boundary constraints

Amortized cost of editing a map with BFF is 30x faster compared to CPMS
(Layout matrix cannot be prefactored)

Modification to LinABF

High Level Idea:

Optimize corner angles β to find near flat metric

Find planar vertex positions approximating angles via least squares layout

Modification:

To prescribe exterior angles $\tilde{\kappa}$, add linear boundary constraints $\sum \beta_i^{jk} = \pi \; - \; \tilde{\kappa}_i$

To prescribe boundary lengths $\tilde{l}_{ij}$, add boundary condition $\prod_{ijk} \frac{sin \; \beta_i^{jk}}{sin \; \beta_j^{ji}} = \frac{\tilde{l}_{i-1,i}}{\tilde{l}_{i,i+1}}$

Comparison with BFF:

Artifacts due to linearization and least squares layout

Neither least squares matrix nor angle constraint matrix can be prefactored

I don’t know…

D’oh!