Conformal Mesh Parameterization

Richard Liu

May 17th, 2022

Introduction

Surface parameterization: some mapping $f : S \to \Omega$ where $S \subset \mathbb{R}^3$ is a 3D surface and $\Omega \subset \mathbb{R}^2$ or $\mathbb{R}^3$

Introduction

Surface parameterization: some mapping $f : S \to \Omega$ where $S \subset \mathbb{R}^3$ is a 3D surface and $\Omega \subset \mathbb{R}^2$ or $\mathbb{R}^3$

Historical motivation: cartography

Introduction

Surface parameterization: some mapping $f : S \to \Omega$ where $S \subset \mathbb{R}^3$ is a 3D surface and $\Omega \subset \mathbb{R}^2$ or $\mathbb{R}^3$

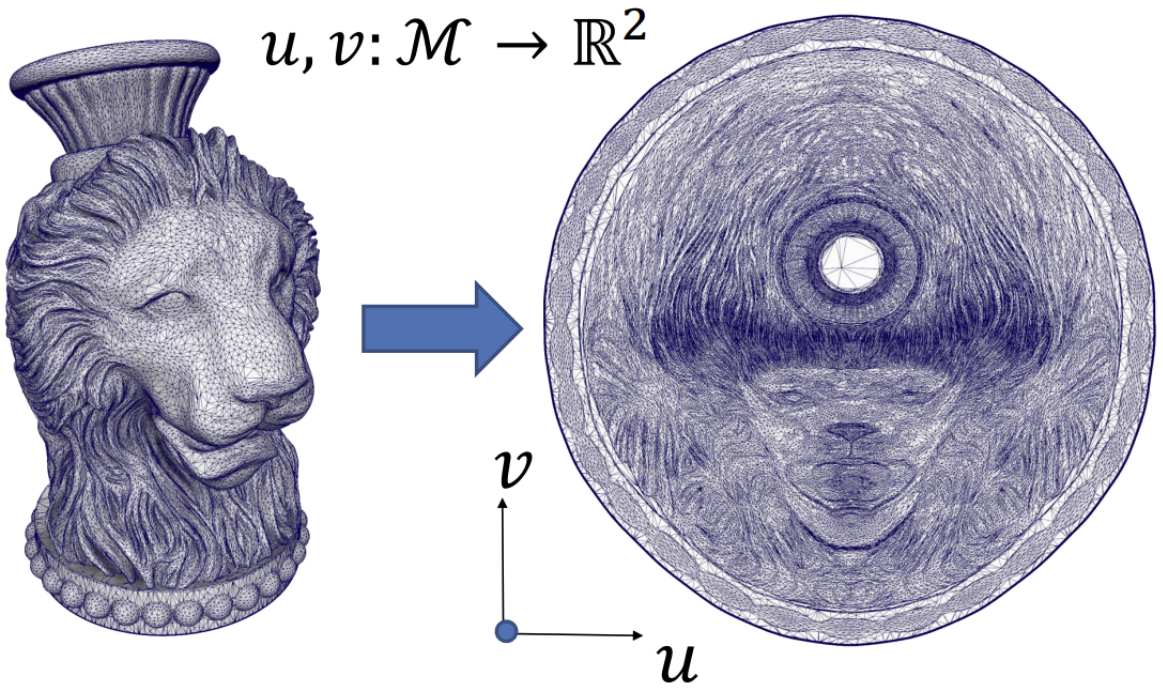
Historical motivation: cartography

Turns out it is **impossible** to make a 2D map of the Earth without some distortion. (Gauss's *Theorema Egregium*, 1827)

Introduction

Surface parameterization: some mapping $f : S \to \Omega$ where $S \subset \mathbb{R}^3$ is a 3D surface and $\Omega \subset \mathbb{R}^2$ or $\mathbb{R}^3$

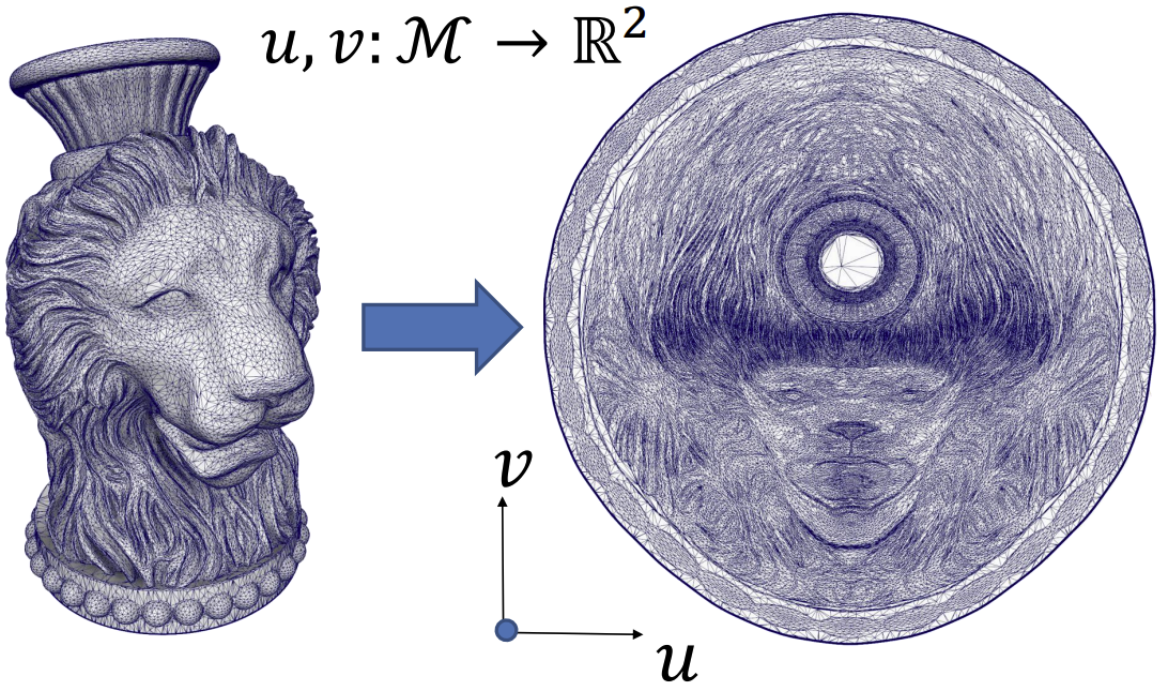
Mesh parameterization: when S is a (triangle) mesh and $\Omega \subset \mathbb{R}^2$, then we define a piecewise linear function $f : S \to \Omega$.



Introduction

Surface parameterization: some mapping $f : S \to \Omega$ where $S \subset \mathbb{R}^3$ is a 3D surface and $\Omega \subset \mathbb{R}^2$ or $\mathbb{R}^3$

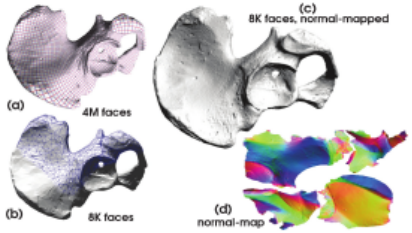
Mesh parameterization: when S is a (triangle) mesh and $\Omega \subset \mathbb{R}^2$, then we define a piecewise linear function $f : S \to \Omega$.



Mesh parameterization is also commonly known as **UV mapping**.

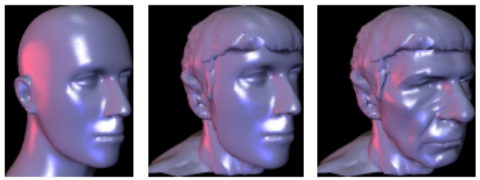
Mesh Parameterization Applications

Texture Mapping

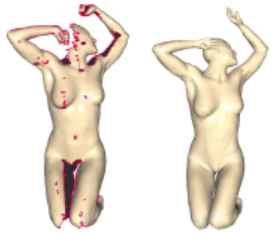


Normal Mapping

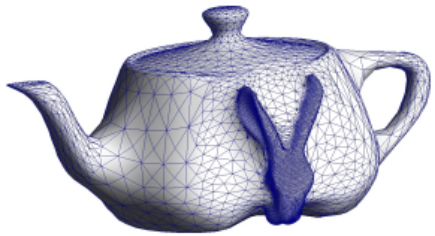
Detail Transfer



Morphing



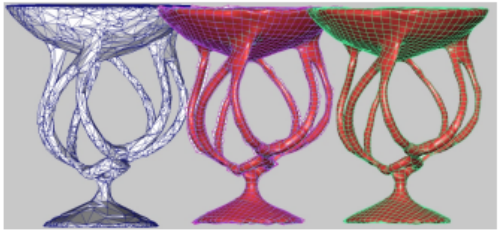
Mesh Completion



Editing

Databases

Remeshing

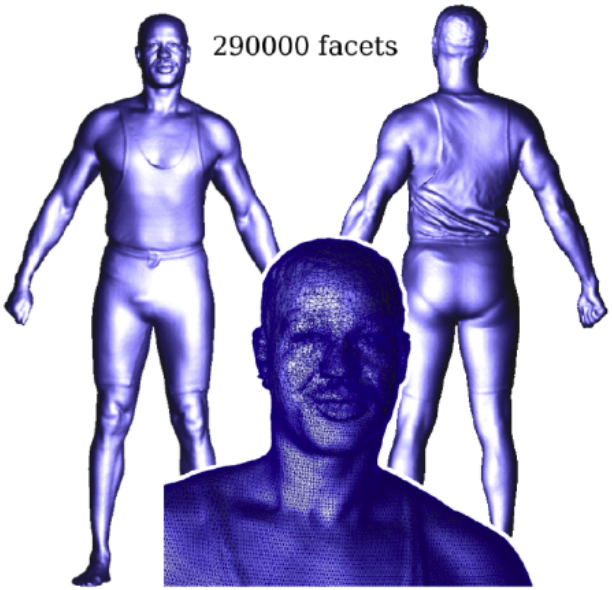


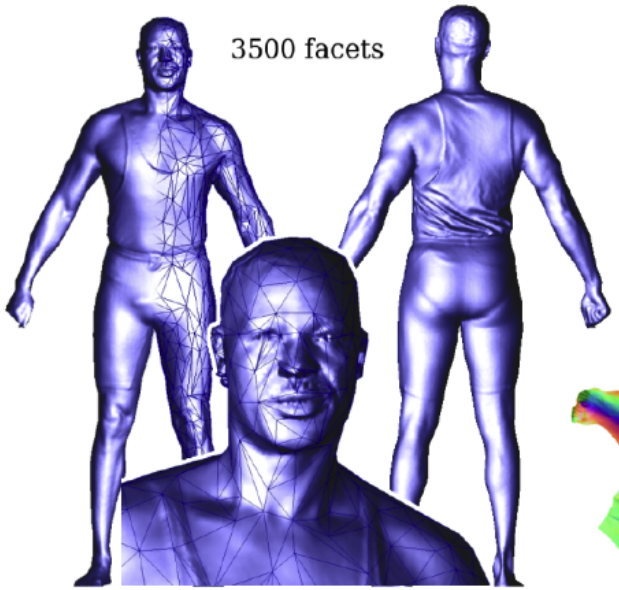
Surface Fitting

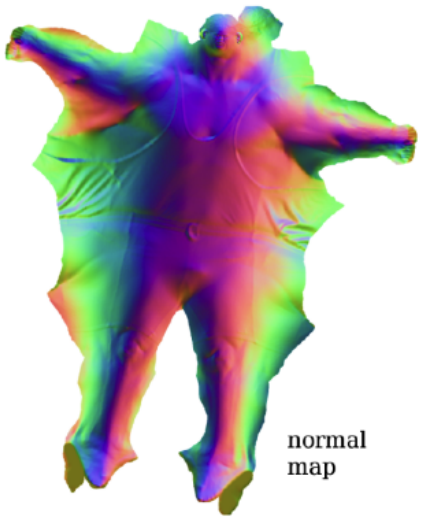
Figure: Parameterization Applications

Mesh Parameterization Applications

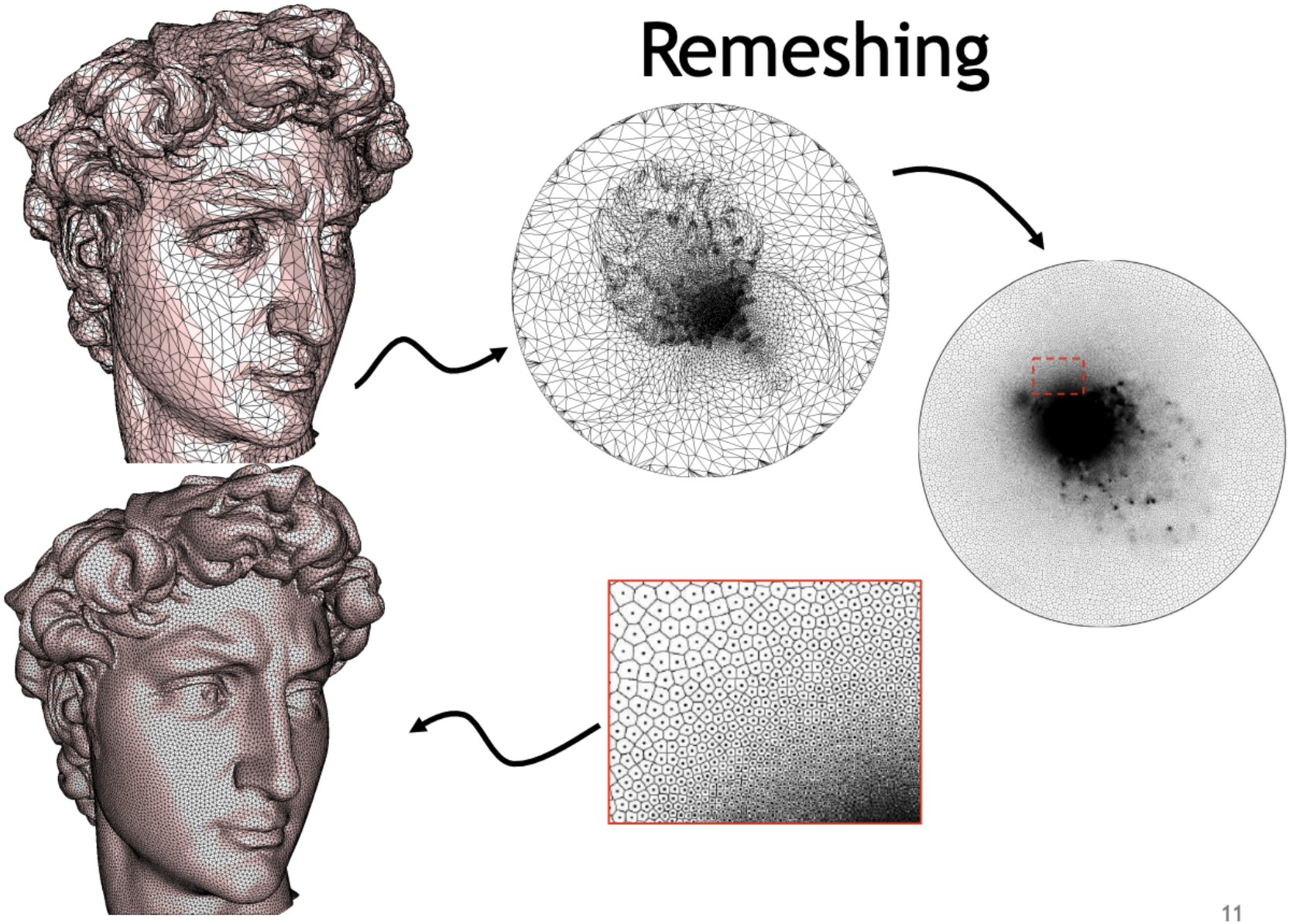
- Normal Mapping







Mesh Parameterization Applications



Mesh Parameterization Applications

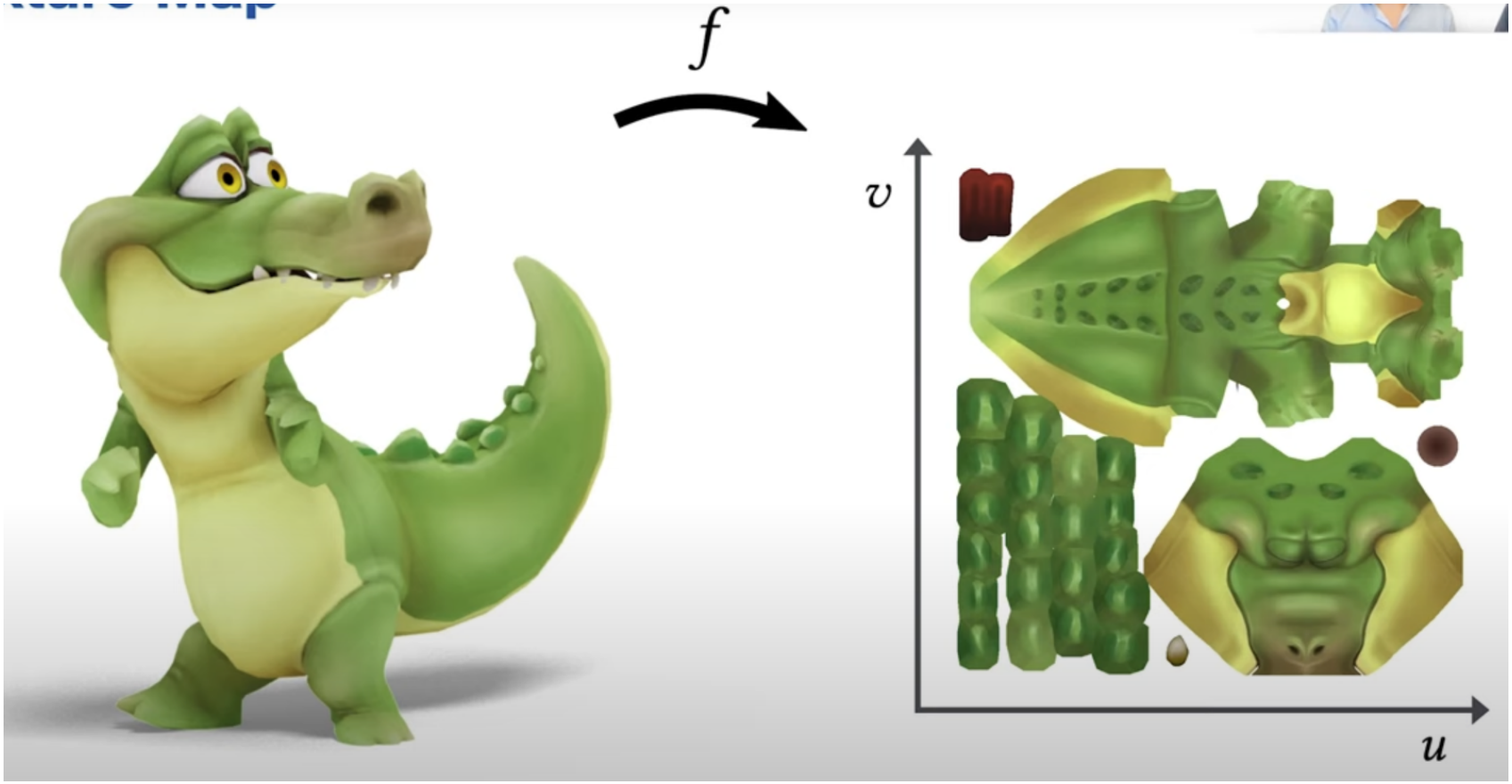


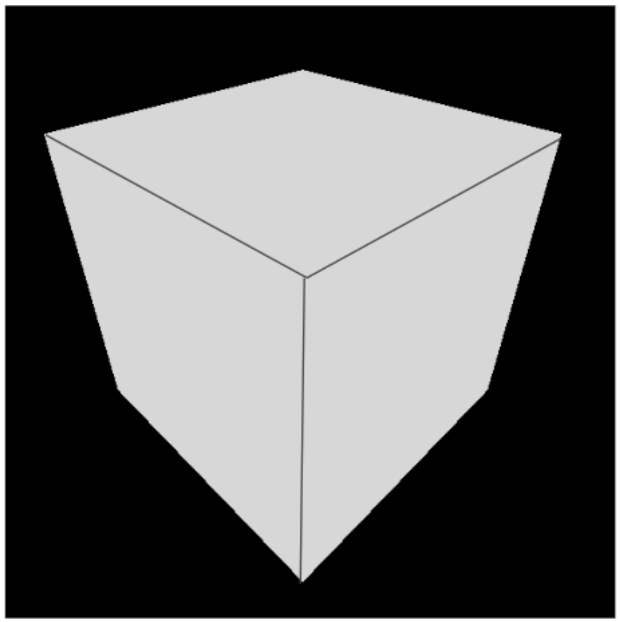
Figure: Texture Mapping

Texture Mapping

Image from Vallet and Levy, techreport INRIA

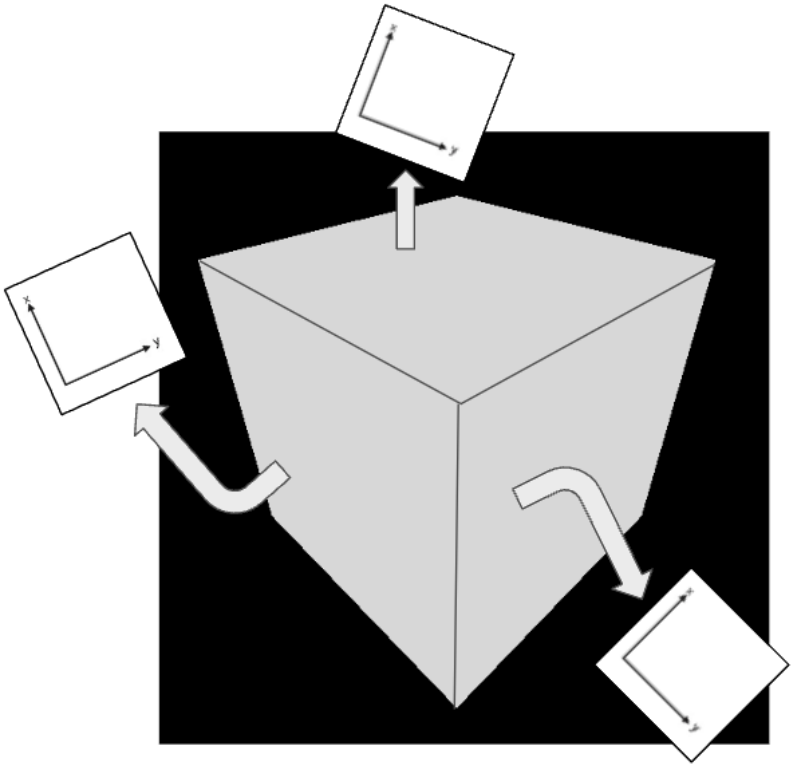
Introduction

Recall: in assignment 2, you were asked to generate a texture map for each face of the cube



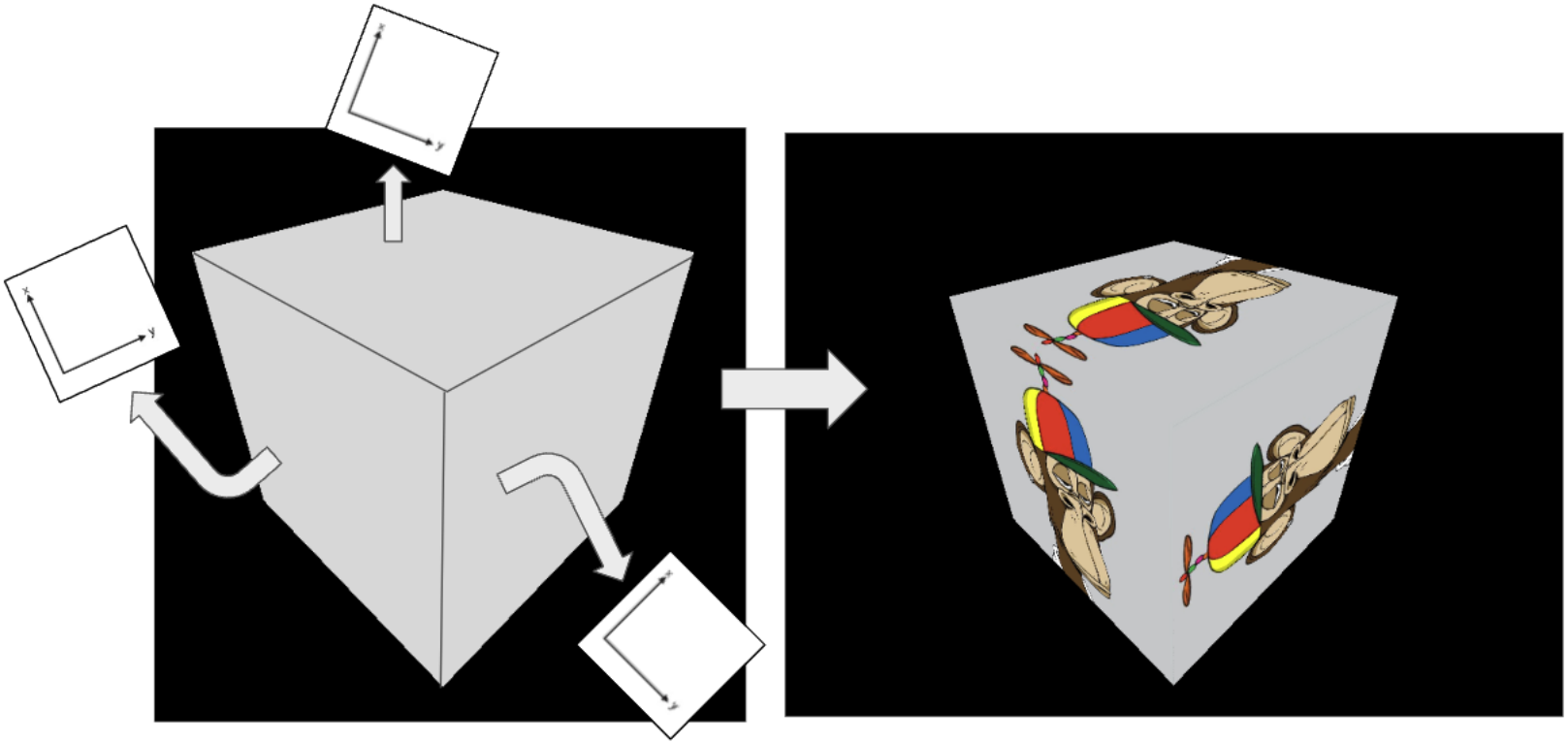
Introduction

Recall: in assignment 2, you were asked to generate a texture map for each face of the cube



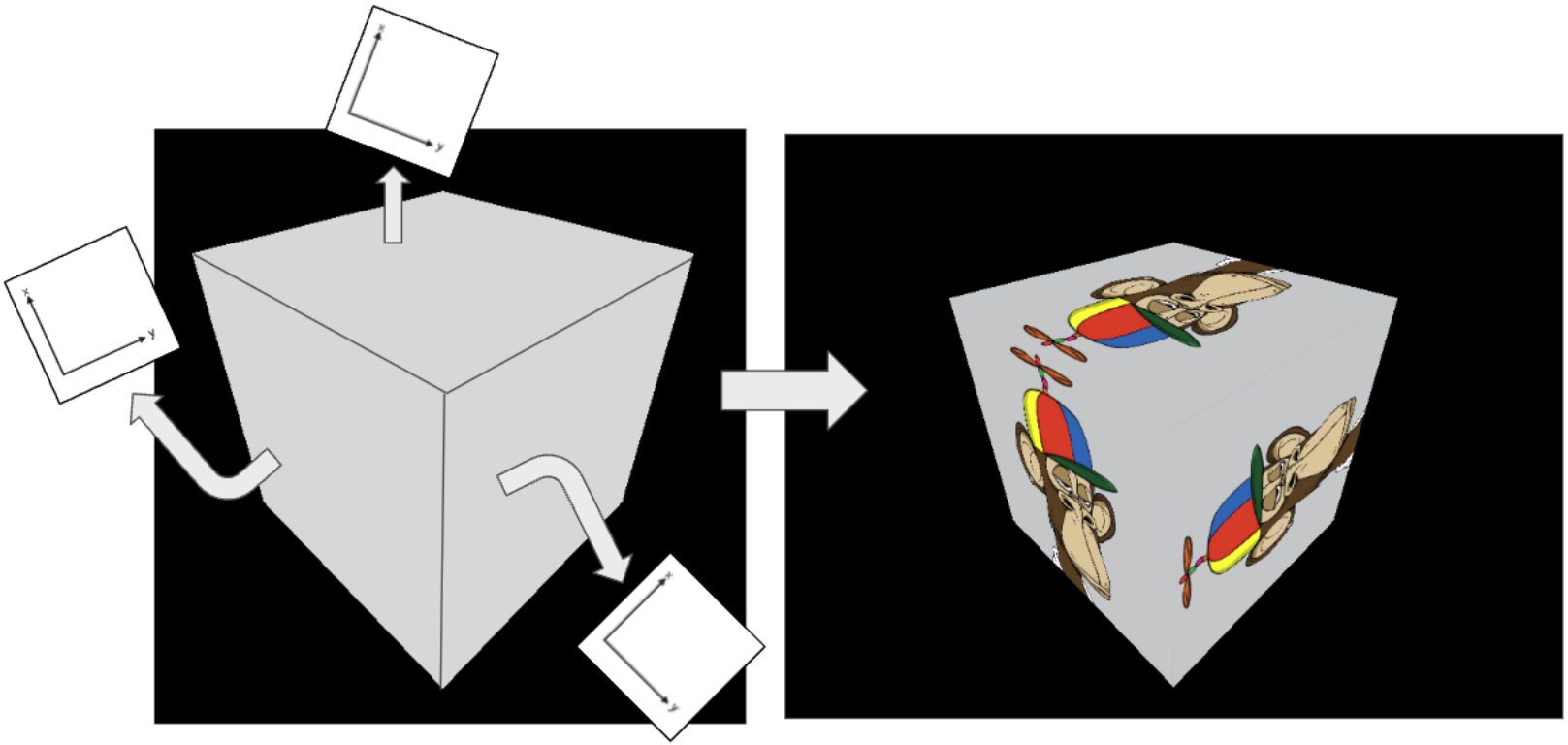
Introduction

Recall: in assignment 2, you were asked to generate a texture map for each face of the cube



Introduction

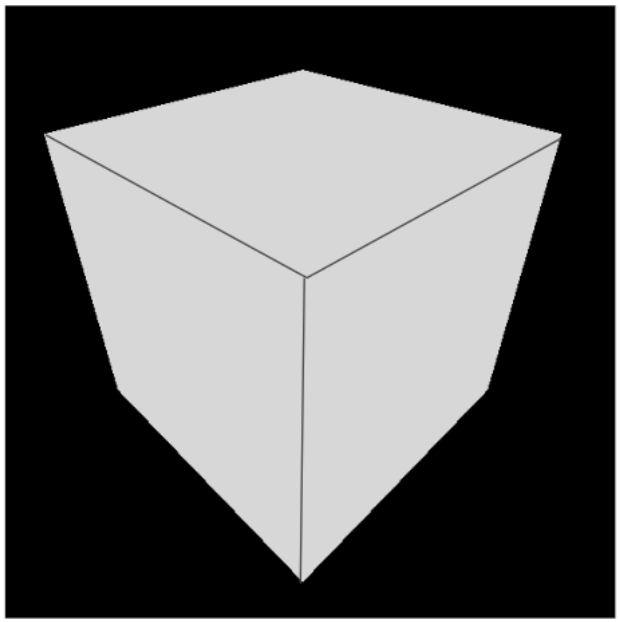
Recall: in assignment 2, you were asked to generate a texture map for each face of the cube



This is an example of a parameterization. Namely, a piecewise linear map $f : \mathbb{R}^3 \to \mathbb{R}^2$.

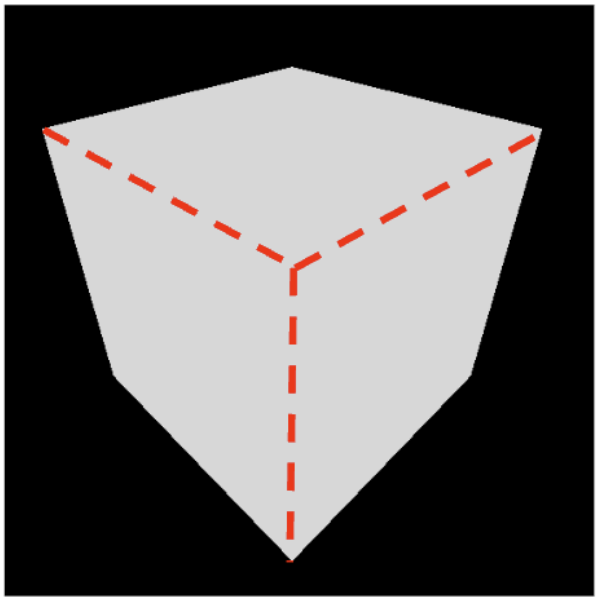
Introduction

Okay, but what if we want to paste the **whole image** onto the whole cube?



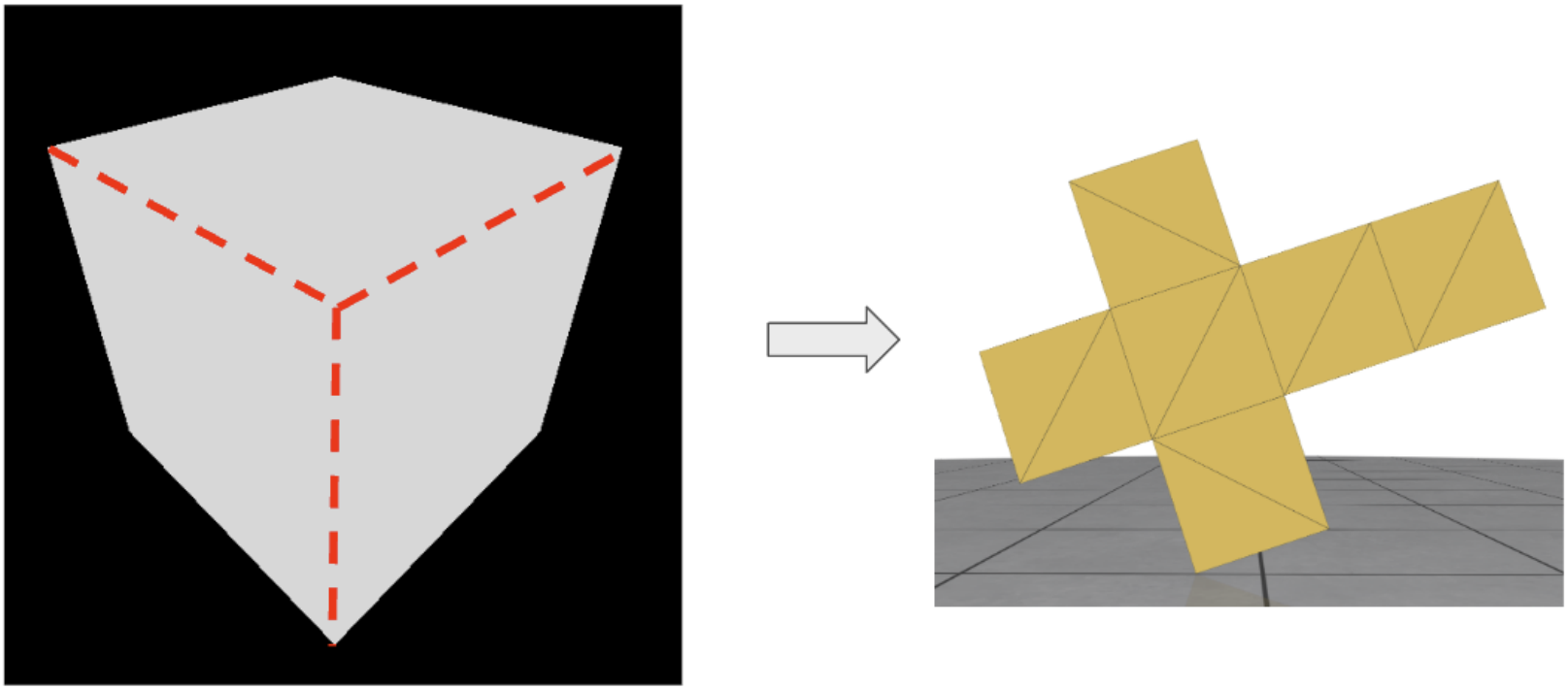
Introduction

Okay, but what if we want to paste the **whole image** onto the whole cube?



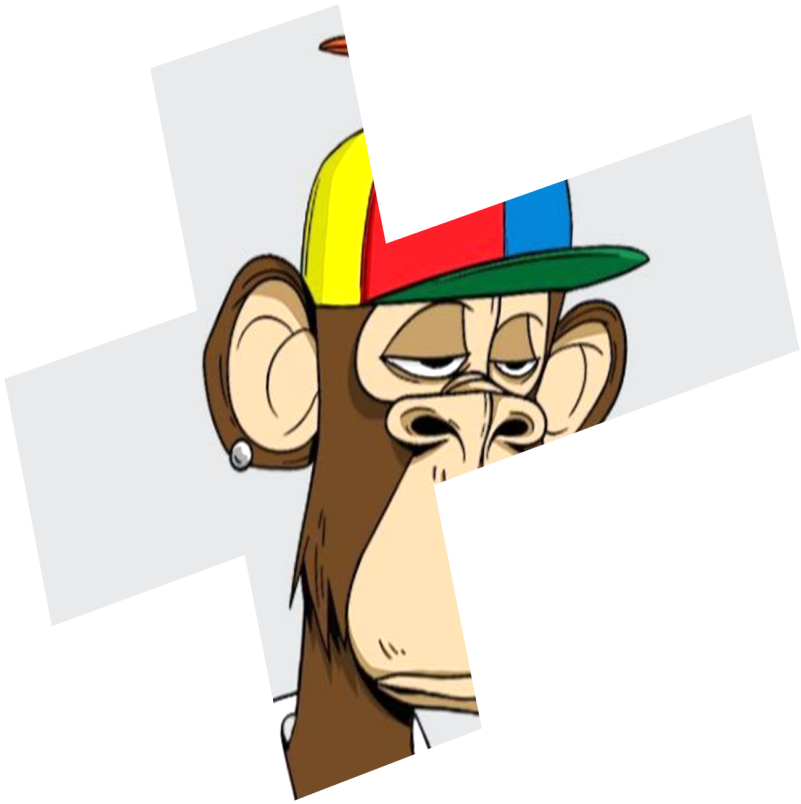
Introduction

Okay, but what if we want to paste the **whole image** onto the whole cube?



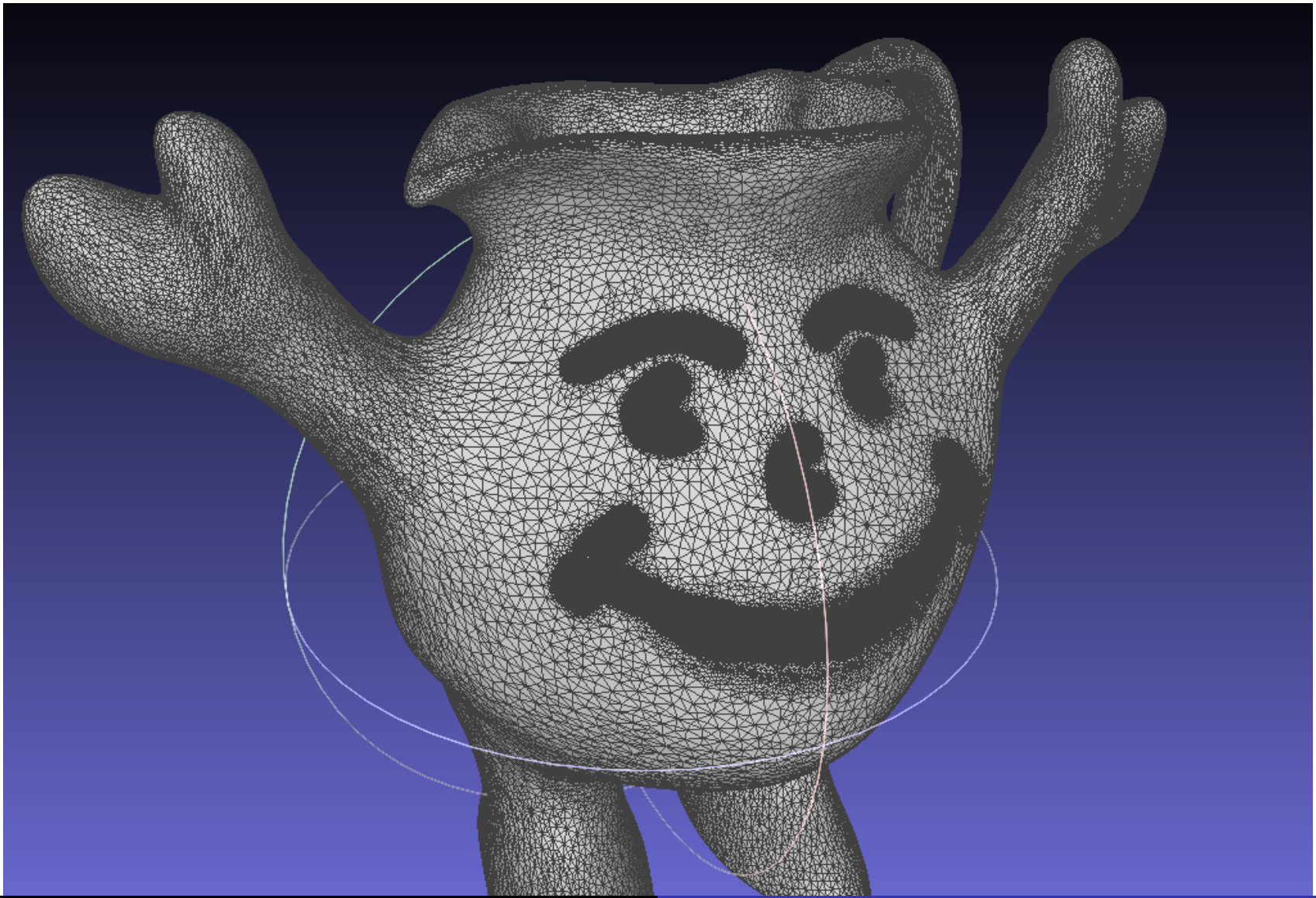
Introduction

Okay, but what if we want to paste the whole image onto the **whole cube**? (Result computed by Least Squares Conformal Map)



Introduction

Okay, but what about something a bit more challenging?



Introduction

This lecture: establish mathematical foundations for classical parameterization techniques in geometry processing.

Introduction

This lecture: establish mathematical foundations for classical parameterization techniques in geometry processing.

We will build up towards a derivation of the **LSCM (Least Squares Conformal Maps)** method and pseudocode implementation.

Least Squares Conformal Maps for Automatic Texture Atlas Generation

Bruno Lévy Sylvain Petitjean Nicolas Ray Jérome Maillot*

ISA (Inria Lorraine and CNRS), France

Introduction

This lecture: establish mathematical foundations for classical parameterization techniques in geometry processing.

We will build up towards a derivation of the **LSCM (Least Squares Conformal Maps)** method and pseudocode implementation.

Least Squares Conformal Maps
for Automatic Texture Atlas Generation

Bruno Lévy Sylvain Petitjean Nicolas Ray Jérome Maillot*

ISA (Inria Lorraine and CNRS), France

First, some analysis review.

Mathematical Framework

Given a function $f : X \to Y$,

Definition

f is **injective** or **one-to-one**, if $\forall x, x' \in X, f(x) = f(x') \Rightarrow x = x'$.

Mathematical Framework

Given a function $f : X \to Y$,

Definition

f is **injective** or **one-to-one**, if $\forall x, x' \in X, f(x) = f(x') \Rightarrow x = x'$.

Definition

f is **surjective** or **onto**, if $\forall y \in Y, \exists x \in X$ s.t. $y = f(x)$.

Mathematical Framework

Given a function $f : X \to Y$,

Definition

f is **injective** or **one-to-one**, if $\forall x, x' \in X, f(x) = f(x') \Rightarrow x = x'$.

Definition

f is **surjective** or **onto**, if $\forall y \in Y, \exists x \in Xs.t.y = f(x)$.

Definition

f is **bijective** if f is both injective and surjective. Equivalently, f is bijective iff it is **invertible**.

Mathematical Framework

Given a function $f : X \to Y$,

Definition

f is **injective** or **one-to-one**, if $\forall x, x' \in X, f(x) = f(x') \Rightarrow x = x'$.

Definition

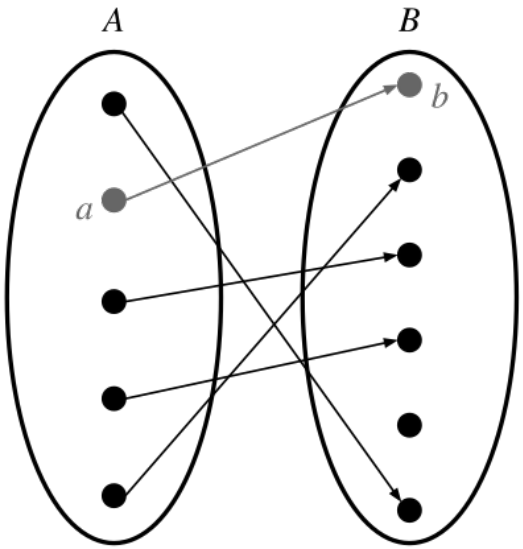
f is **surjective** or **onto**, if $\forall y \in Y, \exists x \in Xs.t.y = f(x)$.

Definition

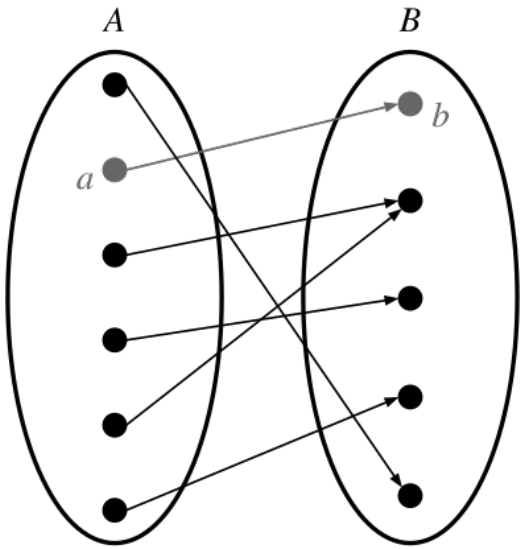
f is **bijective** if f is both injective and surjective. Equivalently, f is bijective iff it is **invertible**.

Mathematical Framework

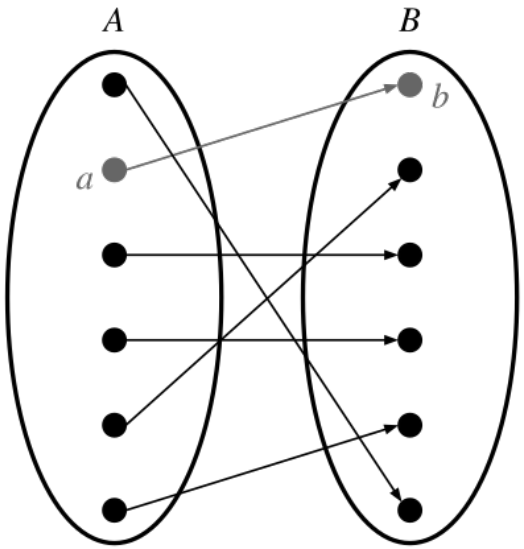
(a) ***Injective*** but **non-surjective**



(b) **Non-Injective** but ***surjective***



(c) **Injective** and **surjective** (***bijective***)



Mathematical Framework

Definition

Let $\Omega \subset \mathbb{R}^2$ be a **simply connected** (without holes) region. Let $f : \Omega \to \mathbb{R}^3$ be continuous and injective. The image of f is called a **surface**

$$S = f(\Omega) = \{f(u, v) : (u, v) \in \Omega\}$$

We say that f is a **parameterization** of S over the **parameter domain** Ω.

Mathematical Framework

Definition

Let $\Omega \subset \mathbb{R}^2$ be a **simply connected** (without holes) region. Let $f : \Omega \to \mathbb{R}^3$ be continuous and **injective**. The image of f is called a **surface**

$$S = f(\Omega) = \{f(u, v) : (u, v) \in \Omega\}$$

We say that f is a **parameterization** of S over the **parameter domain** Ω.

Note: By construction, $f : \Omega \to S$ is trivially surjective, so f is **bijective**.

Mathematical Framework

Definition

Let $\Omega \subset \mathbb{R}^2$ be a **simply connected** (without holes) region.
Let $f : \Omega \to \mathbb{R}^3$ be continuous and injective. The image of f is called a **surface**

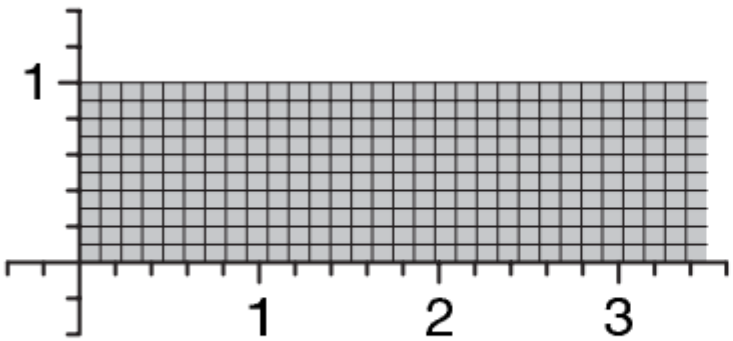
$$S = f(\Omega) = \{f(u, v) : (u, v) \in \Omega\}$$

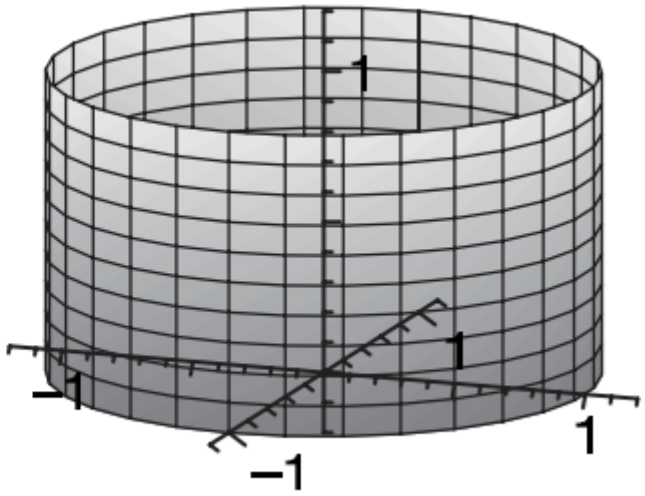
We say that f is a **parameterization** of S over the **parameter domain** Ω.

Note: By construction, $f : \Omega \to S$ is trivially surjective, so f is **bijective**.
Note 2: In practice we care about the inverse map $f^{-1} : S \to \Omega$, but we formulate in this way because reasons. (How do we know this inverse exists?)

Mathematical Framework

Example

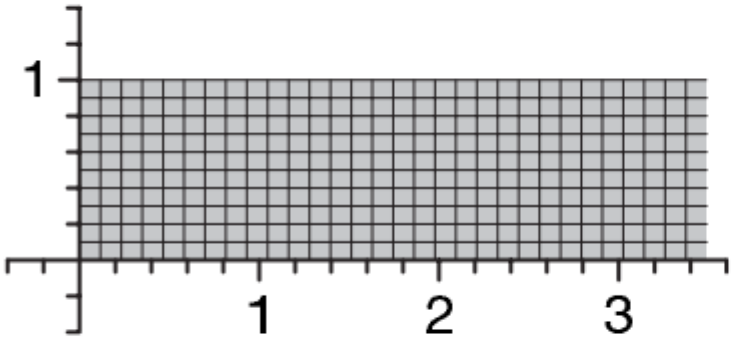


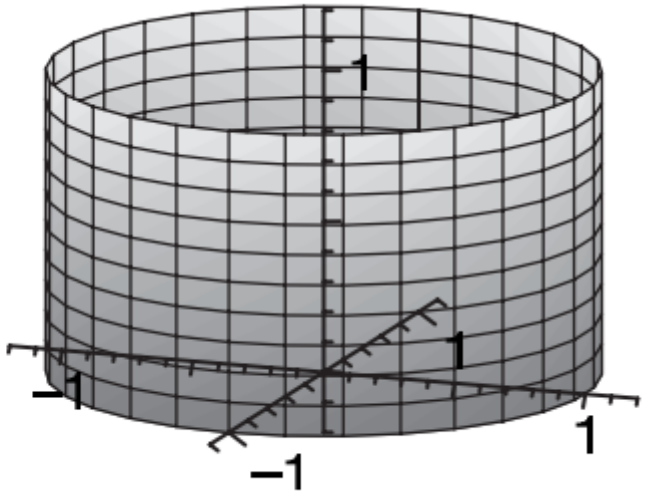


- Parameter domain: $\Omega = \{(u, v) \in \mathbb{R}^2 : u \in [0, 2\pi), v \in [0, 1]\}$

Mathematical Framework

Example

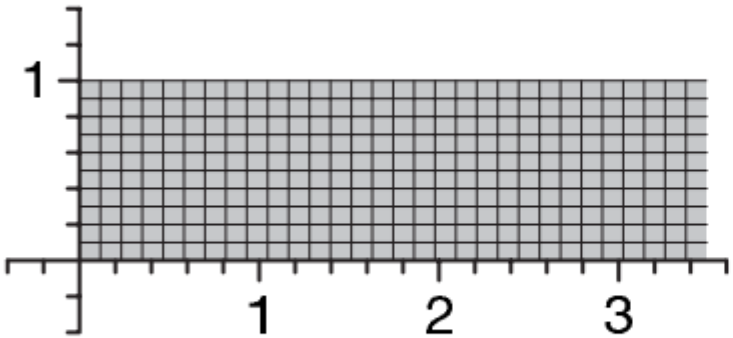


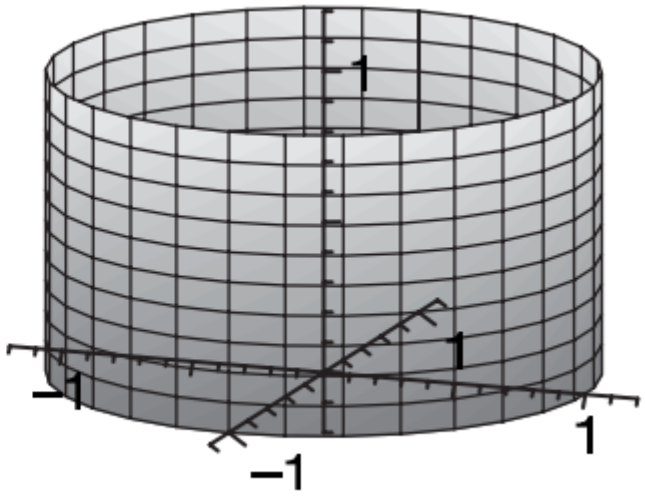


- Parameter domain: $\Omega = \{(u, v) \in \mathbb{R}^2 : u \in [0, 2\pi), v \in [0, 1]\}$
- Surface: $S = \{(x, y, z) \in \mathbb{R}^3 : x^2 + y^2 = 1, z \in [0, 1]\}$

Mathematical Framework

Example





- Parameter domain: $\Omega = \{(u, v) \in \mathbb{R}^2 : u \in [0, 2\pi), v \in [0, 1]\}$
- Surface: $S = \{(x, y, z) \in \mathbb{R} : x^2 + y^2 = 1, z \in [0, 1]\}$
- Parameterization: $f(u, v) = (\cos u, \sin u, v)$
- Inverse: $f^{-1}(x, y, z) = (\arctan y/x, z)$

Mathematical Framework

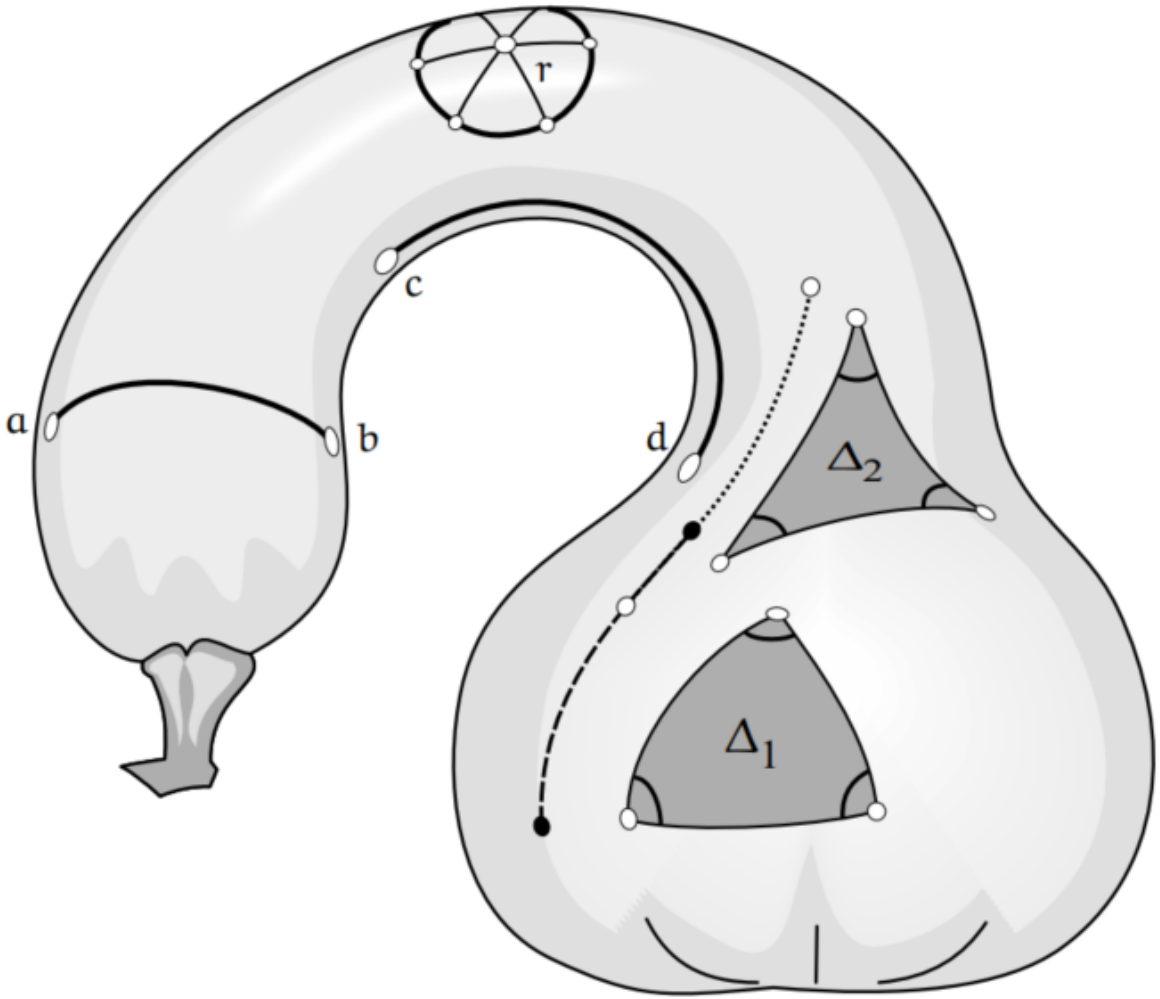
Remark

A parameterization $f : \Omega \to S$ is never unique. Given any bijection $\gamma : \Omega \to \Omega$, $g = f \circ \gamma$ is a parameterization of S over Ω.

Mathematical Framework

We can use f for deriving some key **intrinsic surface properties**, or properties that are **independent** of how the surface sits in space (extrinsic geometry).

Mathematical Framework



[1.9] *The* **intrinsic geometry** *of the surface of a crookneck squash:* **geodesics** *are the equivalents of straight lines, and triangles formed out of them may possess an angular excess of either sign, depending on how the surface bends:* $\mathcal{E}(\Delta_1) > 0$ *and* $\mathcal{E}(\Delta_2) < 0$.

Mathematical Framework

Definition

A parameterization $f : \Omega \subset \mathbb{R}^2 \to S \subset \mathbb{R}^3$ is **regular** if the tangent vectors $f_u = \frac{\partial f}{\partial u}$ and $f_v = \frac{\partial f}{\partial v}$ are always linearly independent.

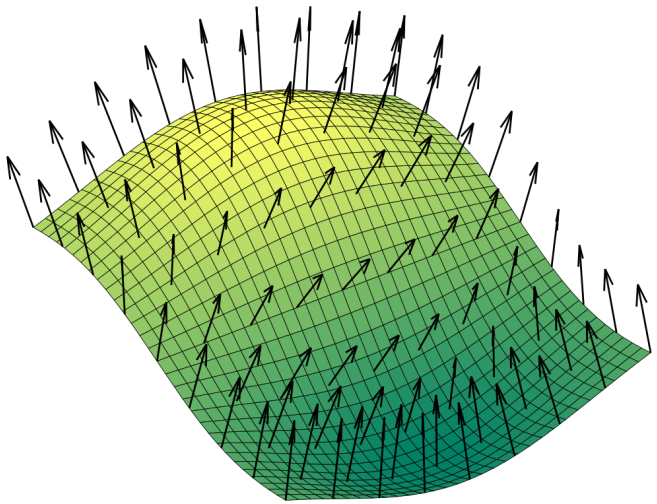
Note: f_u, f_v are functions from $\mathbb{R}^2$ to $\mathbb{R}^3$ and span the local tangent plane.

Mathematical Framework

Definition

Given a **regular** parameterization f, the **surface normal** n_f is defined as

$$n_f = \frac{f_u \times f_v}{||f_u \times f_v||}$$



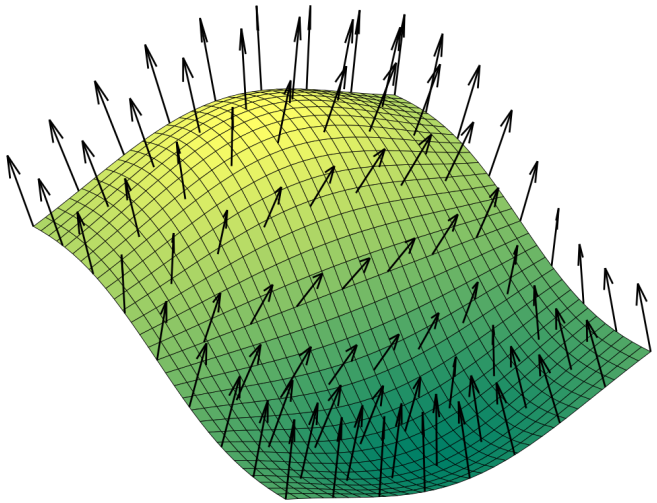
Mathematical Framework

Definition

Given a **regular** parameterization f, the **surface normal** n_f is defined as

$$n_f = \frac{f_u \times f_v}{||f_u \times f_v||}$$

Note: Cross product of linearly dependent vectors is 0, so regularity is required for n_f to be nonzero everywhere.

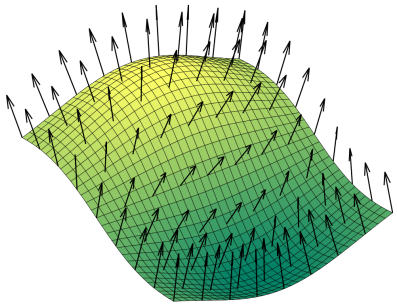


Mathematical Framework

Definition

Given a **regular** parameterization f, the **surface normal** n_f is defined as

$$n_f = \frac{f_u \times f_v}{||f_u \times f_v||}$$



The surface normal is an **intrinsic property**, so can be computed with **any valid parameterization function!**

Mathematical Framework

We can also apply f towards deriving the **first and second fundamental forms**.

Mathematical Framework

We can also apply f towards deriving the **first and second fundamental forms**.

They are fundamental precisely because they characterize the **key metric properties** of a surface, such as the gaussian curvature, mean curvature, and surface area.

Mathematical Framework

Definition

Given parameterization f, the **first fundamental form** is defined as

$$\mathrm{I}_f = \begin{pmatrix} f_u \cdot f_u & f_u \cdot f_v \\ f_v \cdot f_u & f_v \cdot f_v \end{pmatrix} = \begin{pmatrix} E & F \\ F & G \end{pmatrix}$$

Mathematical Framework

Definition

Given parameterization f, the **first fundamental form** is defined as

$$\mathrm{I}_f = \begin{pmatrix} f_u \cdot f_u & f_u \cdot f_v \\ f_v \cdot f_u & f_v \cdot f_v \end{pmatrix} = \begin{pmatrix} E & F \\ F & G \end{pmatrix}$$

Area of a Surface

Given parameterization $f : \Omega \to S$, the area $A(S)$ can be found

$$A(S) = \int_\Omega \sqrt{\det(\mathrm{I}_f)} du dv$$

Mathematical Framework

Definition

Given a twice-differentiable parameterization f, the **second fundamental form** is defined as

$$\mathrm{II}_f = \begin{pmatrix} f_{uu} \cdot n_f & f_{uv} \cdot n_f \\ f_{uv} \cdot n_f & f_{vv} \cdot n_f \end{pmatrix} = \begin{pmatrix} L & M \\ M & N \end{pmatrix}$$

Mathematical Framework

Definition

The Gaussian curvature K is

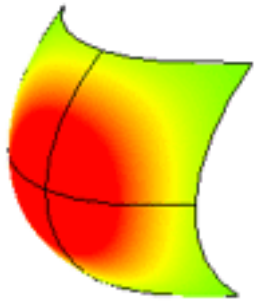
$$K = \det(\mathrm{I}_f^{-1}\mathrm{II}_f) = \frac{\det \mathrm{II}_f}{\det \mathrm{I}_f} = \frac{LN - M^2}{EG - F^2} = \kappa_1\kappa_2$$

where κ_1, κ_2 are the **principal curvatures** (the curvatures along the directions the surface bends the most).

Mathematical Framework

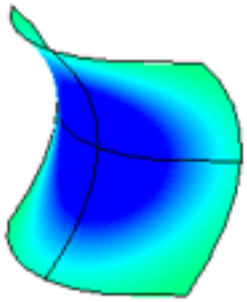
Positive curvature

A positive Gaussian curvature value means the surface is bowl-like.



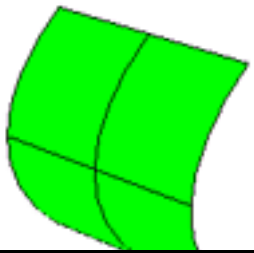
Negative curvature

A negative value means the surface is saddle-like.



Zero curvature

A zero value means the surface is flat in at least one direction. (Planes, cylinders, and cones have zero Gaussian curvature).



Mathematical Framework

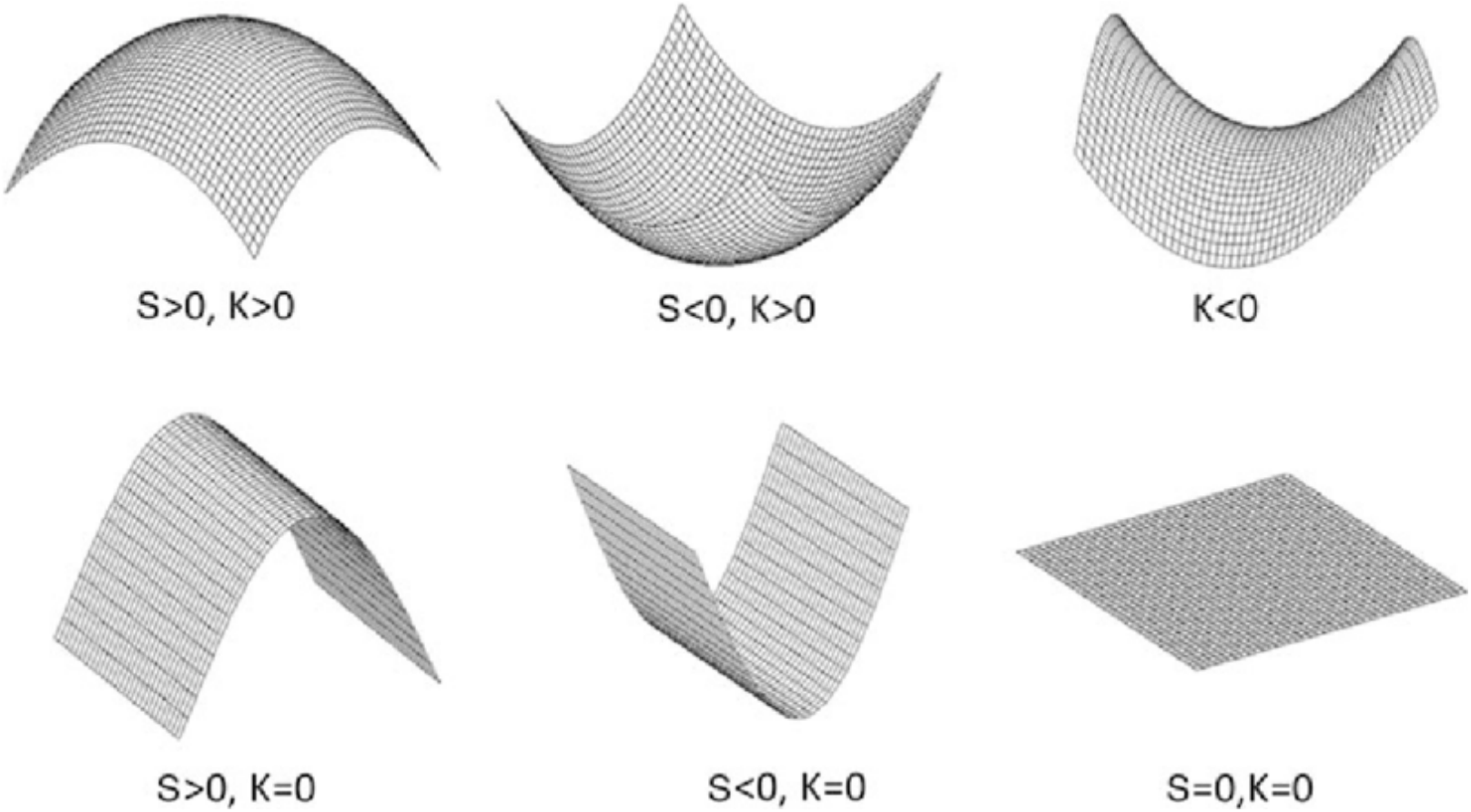
Definition

The mean curvature S is

$$S = \frac{1}{2}\text{trace}(\text{I}_f^{-1}\text{II}_f) = \frac{LG - 2MF + NE}{2(EG - F^2)} = \frac{\kappa_1 + \kappa_2}{2}$$

where κ_1, κ_2 are the **principal curvatures** (the curvatures along the directions the surface bends the most).

Mathematical Framework



K = Gaussian curvature (**intrinsic** = does not change with movement of surface)
S = Mean curvature (**extrinsic** = changes with movement of surface)

Mathematical Framework

Definition

A surface S is **developable** if $\forall p \in S$, $K(p) = 0$, i.e. the Gaussian curvature is 0 everywhere on S.

Mathematical Framework

Definition

A surface S is **developable** if $\forall p \in S$, $K(p) = 0$, i.e. the Gaussian curvature is 0 everywhere on S.

Back to Gauss's *Theorema Egregium*...

Theorem

(Gauss, 1827) Globally isometric parameterizations (from 3D to 2D) only exist for developable *surfaces (i.e. $K = 0$ everywhere)*

Mathematical Framework

Definition

A surface S is **developable** if $\forall p \in S$, $K(p) = 0$, i.e. the Gaussian curvature is 0 everywhere on S.

Back to Gauss's *Theorema Egregium*...

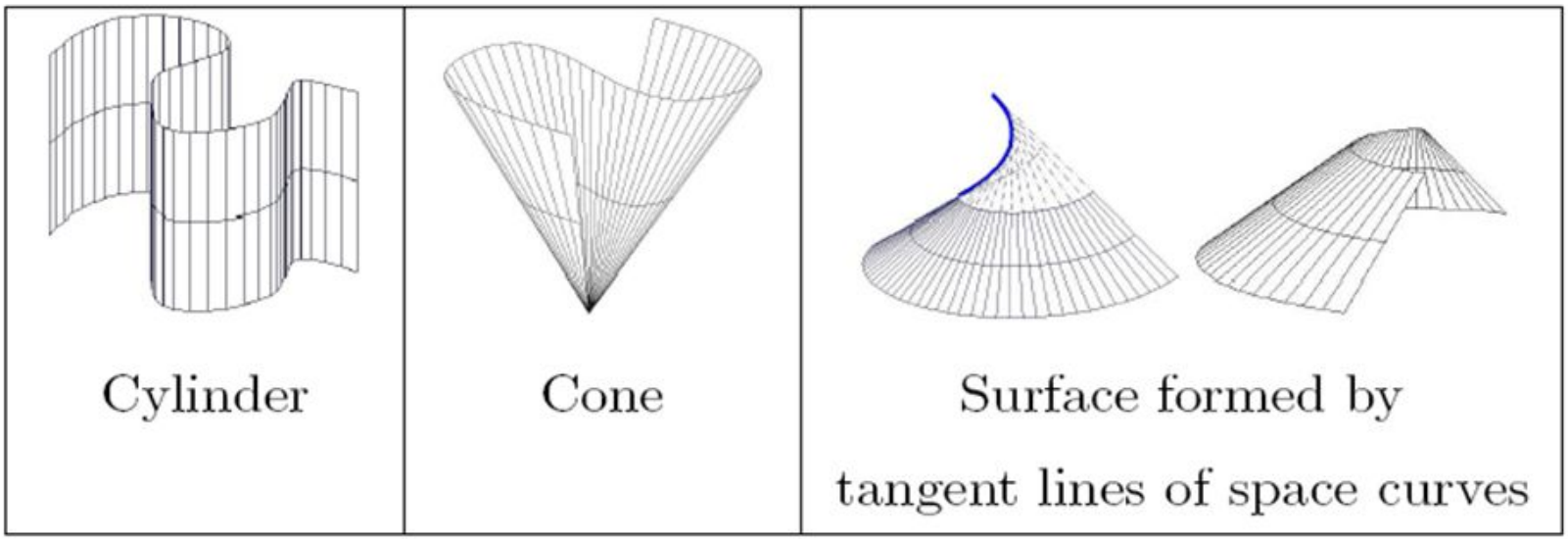
Theorem

(Gauss, 1827) Globally isometric parameterizations (from 3D to 2D) only exist for developable *surfaces (i.e. $K = 0$ everywhere)*

Isometric: zero-distortion. More on this soon...

Developable Surface

Three types of developable surfaces



Mathematical Framework

Definition

The **Jacobian** of parameterization f is the 3 x 2 matrix of partial derivatives of f.

$$J_f = (f_u, f_v)$$

Mathematical Framework

Definition

For any $m \times n$ matrix J, the **singular value decomposition** (SVD) is given by

$$J = U\Sigma V^T$$

where Σ is an $m \times n$ diagonal matrix, and U and V are $m \times m$ and $n \times n$ orthonormal matrices, respectively.

Mathematical Framework

Definition

For any $m \times n$ matrix J, the **singular value decomposition** (SVD) is given by

$$J = U\Sigma V^T$$

where Σ is an $m \times n$ diagonal matrix, and U and V are $m \times m$ and $n \times n$ orthonormal matrices, respectively.

By the above, the SVD of the Jacobian is

$$J_f = U \begin{pmatrix} \sigma_1 & 0 \\ 0 & \sigma_2 \\ 0 & 0 \end{pmatrix} V^T$$

where σ_1, σ_2 are the **singular values**.

Mathematical Framework

Protip: There is an easier way to get the singular values of the Jacobian.

Mathematical Framework

Protip: There is an easier way to get the singular values of the Jacobian.

Remark

We can write the first fundamental form as

$$\mathrm{I}_f = J_f^T J_f = \begin{pmatrix} f_u \cdot f_u & f_u \cdot f_v \\ f_v \cdot f_u & f_v \cdot f_v \end{pmatrix} = \begin{pmatrix} f_u^T \\ f_v^T \end{pmatrix} (f_u \; f_v)$$

It is clear I_f is **symmetric**.

Mathematical Framework

There is an easier way to get the singular values of the Jacobian.

Remark

We can write the first fundamental form as

$$\mathrm{I}_f = J_f^T J_f = \begin{pmatrix} f_u^T \\ f_v^T \end{pmatrix} (f_u \ f_v)$$

It is clear I_f is **symmetric**.

Eigenvalues of I_f (2x2 matrix) are given by

$$\lambda_{1,2} = \frac{1}{2}((E+G) \pm \sqrt{4F^2 + (E-G)^2}$$

.

Mathematical Framework

Remark

For a matrix A, the singular values are the square roots of the eigenvalues of $A^T A$.

Mathematical Framework

Remark

For a matrix A, the singular values are the square roots of the eigenvalues of $A^T A$.

The singular values of J can be found using the eigenvalues of I_f

$$\sigma_1 = \sqrt{\lambda_1}$$
$$\sigma_2 = \sqrt{\lambda_2}$$

Mathematical Framework

Remark

For a matrix A, the singular values are the square roots of the eigenvalues of $A^T A$.

The singular values of J can be found using the eigenvalues of I_f

$$\sigma_1 = \sqrt{\lambda_1}$$
$$\sigma_2 = \sqrt{\lambda_2}$$

σ_1 and σ_2 tell us **everything** about the **metric distortion** induced by the parameterization.

Properties of Parameterizations

Parameterizations induce distortion in **lengths**, which can be further divided into distortion in **angles** and distortion in **areas**.

Properties of Parameterizations

Definition

A parameterization is **conformal**, or **angle-preserving**, when the singular values of the Jacobian are equal, i.e. $\sigma_1 = \sigma_2$.

Properties of Parameterizations

Definition

A parameterization is **conformal**, or **angle-preserving**, when the singular values of the Jacobian are equal, i.e. $\sigma_1 = \sigma_2$.

Definition

A parameterization is **equiareal/authalic**, or **area-preserving**, when the singular values of the Jacobian multiply to 1, i.e. $\sigma_1 \sigma_2 = 1$.

Properties of Parameterizations

Definition

A parameterization is **conformal**, or **angle-preserving**, when $\sigma_1 = \sigma_2$.

Definition

A parameterization is **equiareal/authalic**, or **area-preserving**, when $\sigma_1 \sigma_2 = 1$.

Definition

A parameterization is **isometric**, or **length-preserving** iff it is conformal and equiareal, i.e. $\sigma_1 = \sigma_2 = 1$.

Properties of Parameterizations

Back to maps! Recall from Gauss's theorem that we cannot make a "perfect" (isometric) 2D map of the Earth...

Properties of Parameterizations

Back to maps! Recall from Gauss's theorem that we cannot make a "perfect" (isometric) 2D map of the Earth...

But we can make perfectly **conformal** or **equiareal** maps.

Properties of Parameterizations

Back to maps! Recall from Gauss's theorem that we cannot make a "perfect" (isometric) 2D map of the Earth...

But we can make perfectly **conformal** or **equiareal** maps.

The famous **Mercator projection** is a conformal map! "North" and "south" correspond to "up" and "down" on the map.

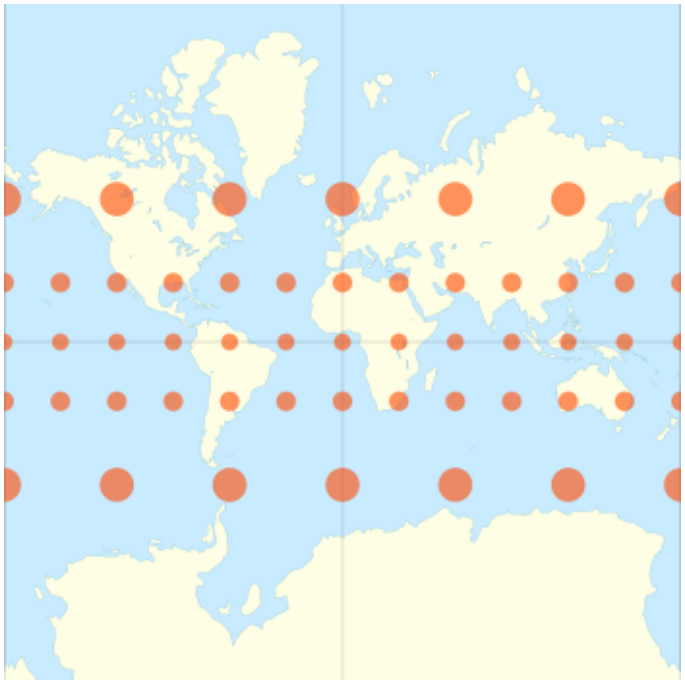
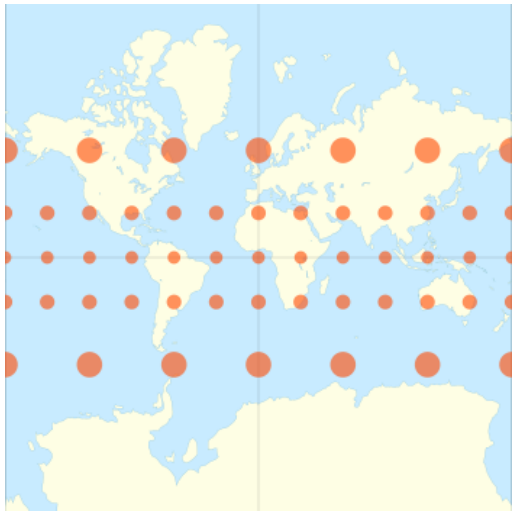


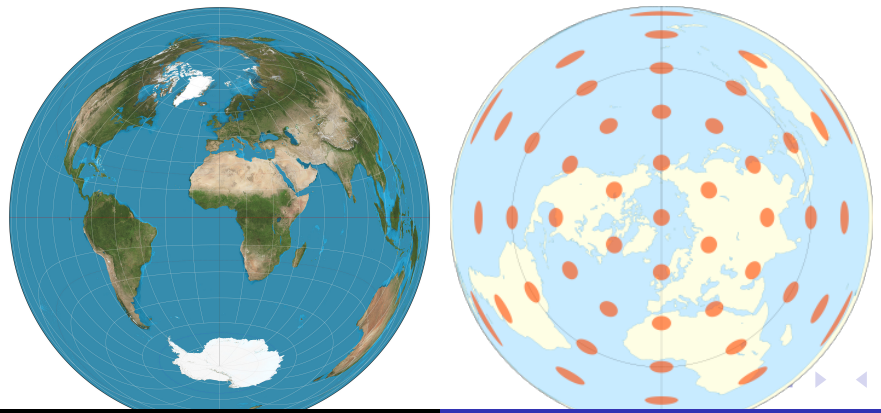
Figure: Mercator projection

Properties of Parameterizations

The famous **Mercator projection** is a conformal map! "North" on the globe is the same as "up" on the map.



Compare this to a **Lambert map**, which is equiareal.



Properties of Parameterizations

Note how we represent the distortion of the mappings with **circles**.

Properties of Parameterizations

Note how we represent the distortion of the mappings with **circles**.

Intuition: the singular values induce **scaling** of the tangent vectors of the mapped surface (represented as a local circle).

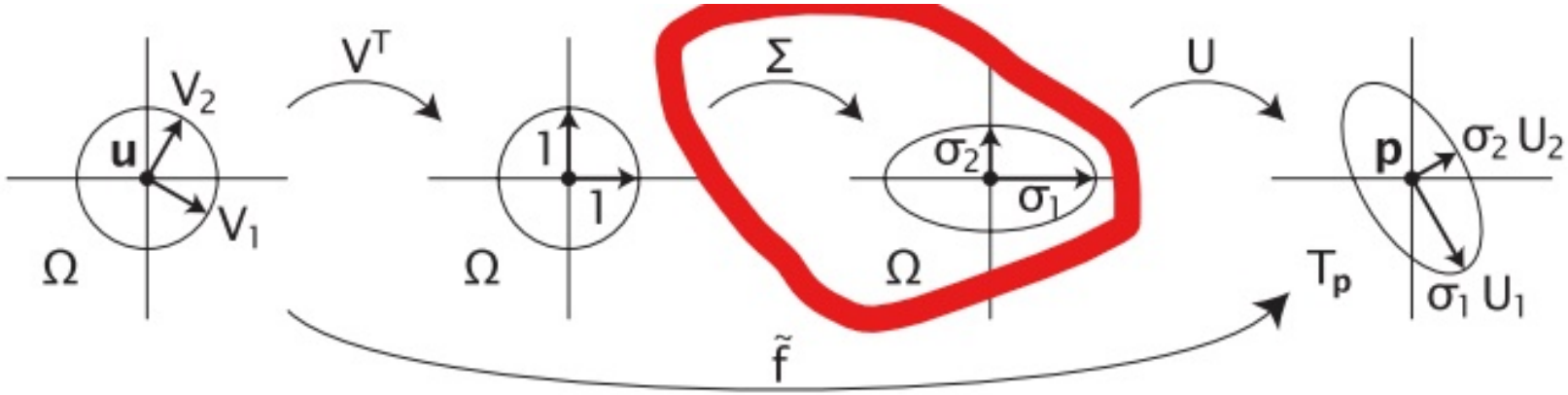


Figure: SVD Decomposition of mapping $\tilde{f}$

Discrete Setting

Note that everything up until this point has been formulated in the continuous setting. So what changes when we consider the discrete mesh setting?

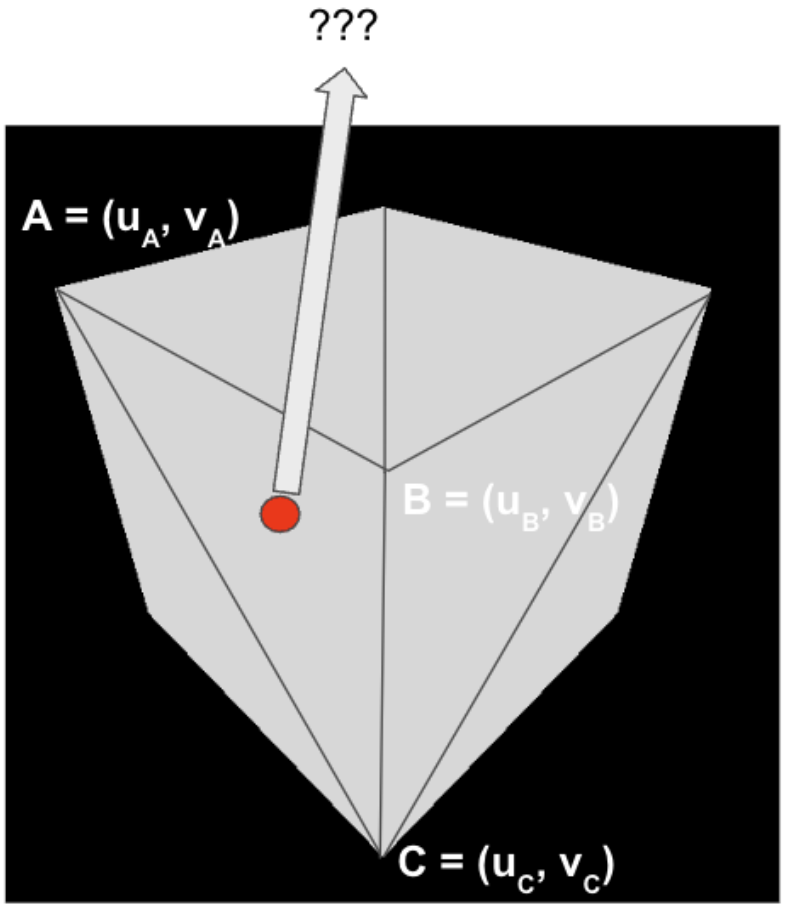
Discrete Setting

Note that everything up until this point has been formulated in the continuous setting. So what changes when we consider the discrete mesh setting?

f can now be considered a **piecewise linear map**. Specifically, we only have to care about how **vertices are mapped to the plane**.

Discrete Setting

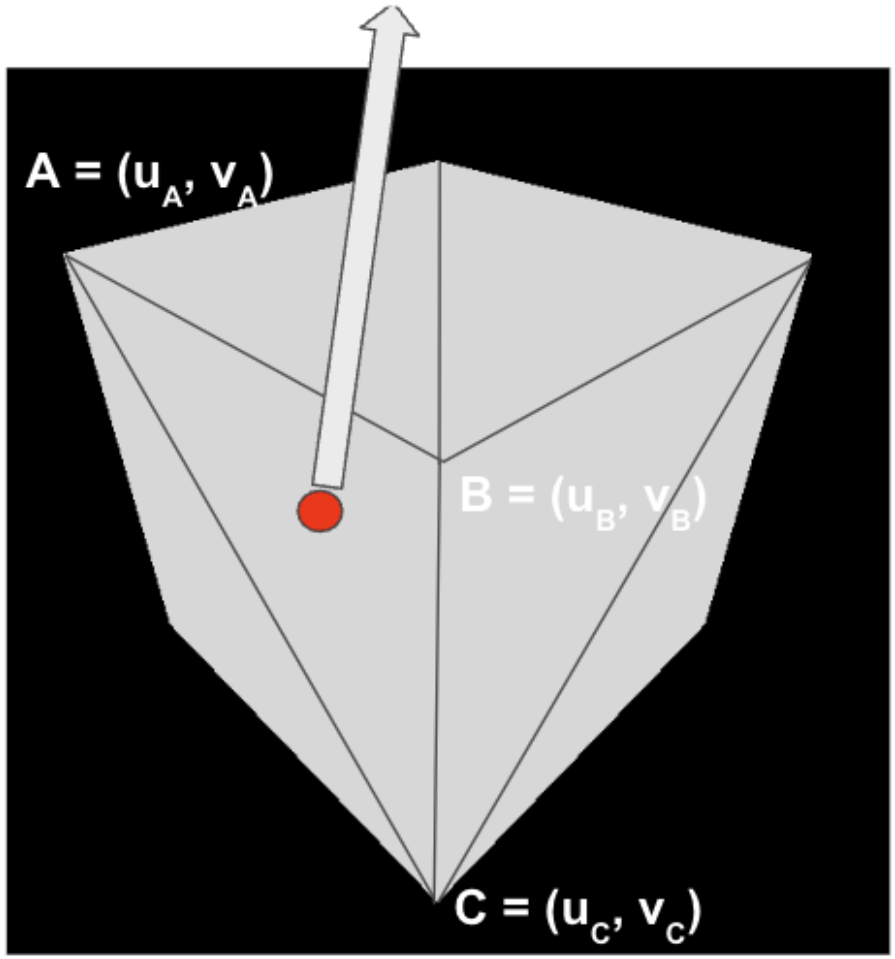
How to get coordinates of the red point with respect to A, B, C?



Discrete Setting

It's our old friend barycentric coordinates!

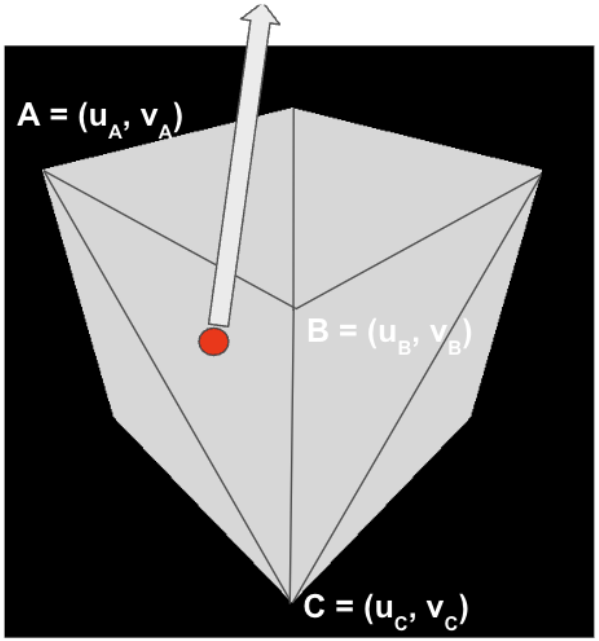
$$p_{UV} = \alpha A + \beta B + \gamma C$$



Discrete Setting

It's our old friend barycentric coordinates!

$$p_{UV} = \alpha A + \beta B + \gamma C$$



Our parameterization function is now $f : \Omega \to V$ where V is the **set of vertex positions** in the mesh.

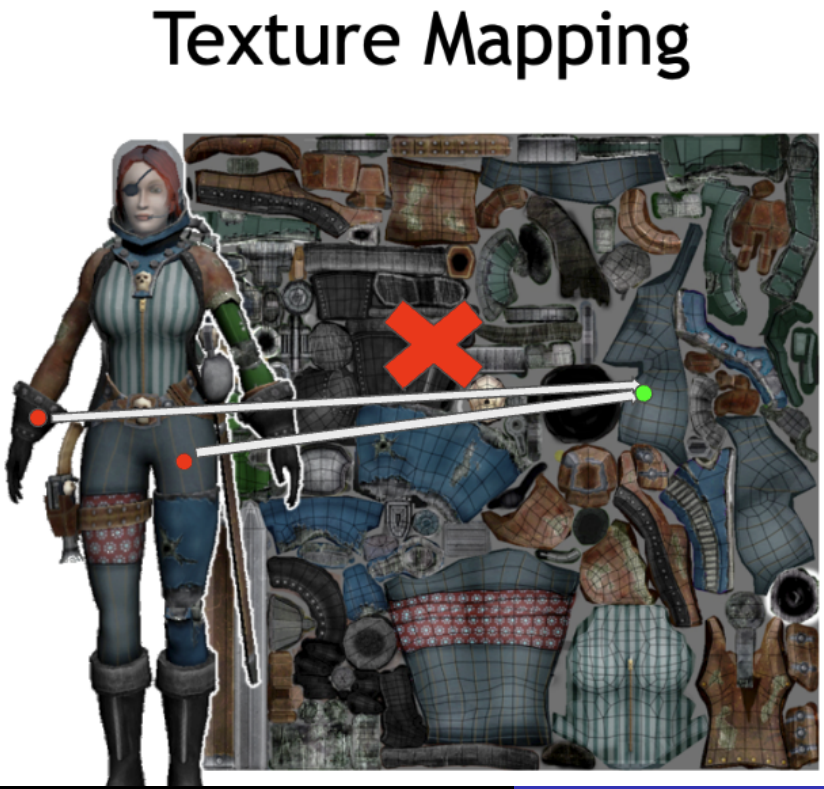
Mesh Parameterization Properties

We already mentioned **conformal**, **equiareal**, and **isometric** maps. Another important property for applications to meshes is **bijectivity**.

Mesh Parameterization Properties

We already mentioned **conformal** and **equiareal**, and **isometric** maps. Another important property for applications to meshes is **bijectivity**.

e.g. For texture mapping, want to be able to annotate parts of the texture with reference to unique region of surface



Mesh Parameterization Properties

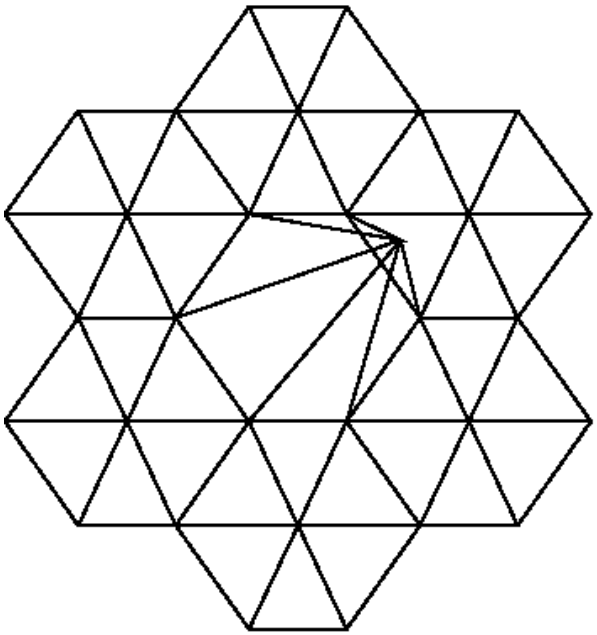
Definition

A mesh parameterization is **locally injective** if no triangles change orientation ("flip" or "fold over") during the parameterization.

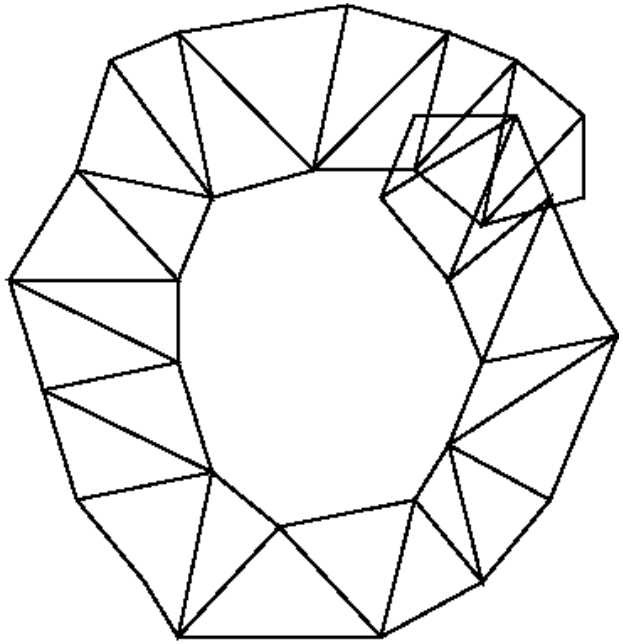
Definition

A mesh parameterization is **globally bijective** if it is locally injective and the boundary of the parameterization does not intersect itself.

Mesh Parameterization Properties



Triangle Flip

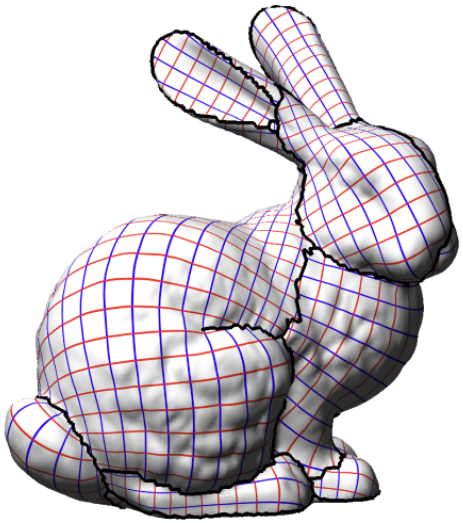


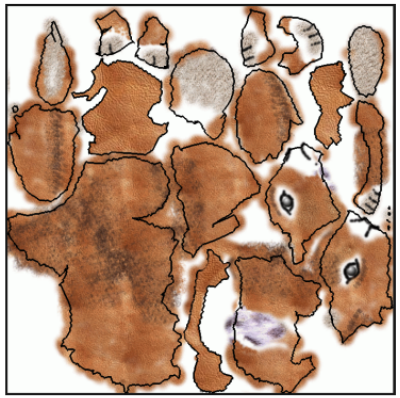
Boundary Intersection

Least Squares Conformal Maps for Automatic Texture Atlas Generation

Bruno Lévy Sylvain Petitjean Nicolas Ray Jérome Maillot*

ISA (Inria Lorraine and CNRS), France





LSCM

LSCM. (Levy et al. 2002) The least squares conformal maps method seeks to minimize the following **conformal energy**

$$E_{LSCM} = E_C = \frac{1}{2}\int_S ||f_v - \text{rot}_{90}(f_u X)||^2 dp = \frac{(\sigma_1 - \sigma_2)^2}{2}$$

LSCM

LSCM. (Levy et al. 2002) The least squares conformal maps method seeks to minimize the following **conformal energy**

$$E_{LSCM} = E_C = \frac{1}{2}\int_S ||f_v - \text{rot}_{90}(f_u X)||^2 dp = \frac{(\sigma_1 - \sigma_2)^2}{2}$$

Intuition: We want the singular values to be equal ($\sigma_1 = \sigma_2$), aka **conformality**!

LSCM

Recall how the singular values of the Jacobian scales tangent vectors.

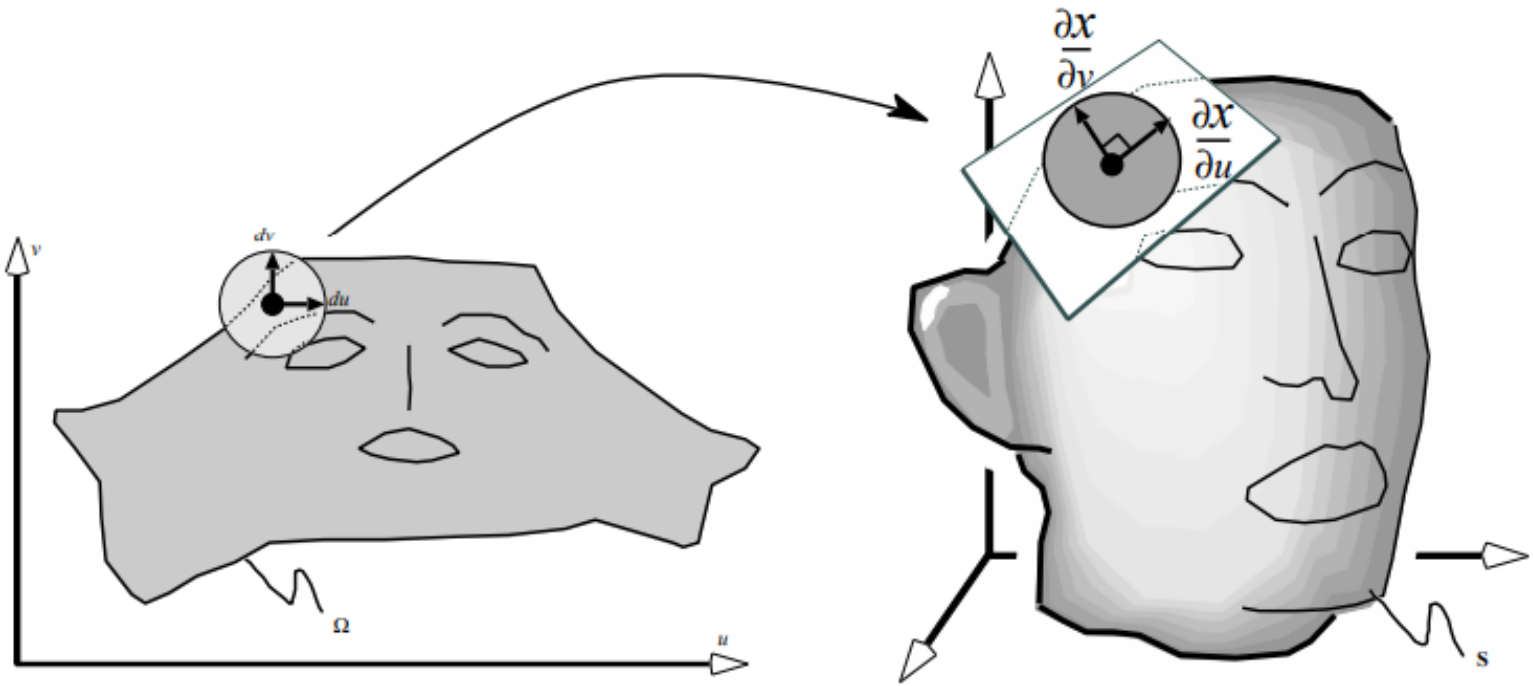


Figure 4.9: A conformal parameterization transforms an elementary circle into an elementary circle.

Paper's perspective: Cauchy-Riemann equations from complex analysis.

Theorem

If a function $f(z) = u(x, y) + iv(x, y)$ is complex differentiable (holomorphic), then it satisfies the partial differential equations

$$\frac{\partial u}{\partial x} = \frac{\partial v}{\partial y}$$
$$\frac{\partial u}{\partial y} = -\frac{\partial v}{\partial x}$$

LSCM

Paper's perspective: Cauchy-Riemann equations from complex analysis.

Theorem

If a function $f(z) = u(x, y) + iv(x, y)$ is complex differentiable (holomorphic), then it satisfies the partial differential equations

$$\frac{\partial u}{\partial x} = \frac{\partial v}{\partial y}$$
$$\frac{\partial u}{\partial y} = -\frac{\partial v}{\partial x}$$

Holomorphic implies conformality. This is equivalent to a condition on the Jacobian.

Forget the fancy math! **This is equivalent to a condition on the Jacobian.**

$$i\frac{\partial f}{\partial x} = \frac{\partial f}{\partial y} \Leftrightarrow \begin{pmatrix} a & -b \\ b & a \end{pmatrix}$$

This is equivalent to a condition on the Jacobian.

$$i\frac{\partial f}{\partial x} = \frac{\partial f}{\partial y} \Leftrightarrow \begin{pmatrix} a & -b \\ b & a \end{pmatrix}$$

LSCM - Least Squares Conformal Map

We want the Jacobian

$$\begin{pmatrix} \partial_x u & \partial_y u \\ \partial_x v & \partial_y v \end{pmatrix}$$

to be a similarity matrix

$$\begin{pmatrix} \alpha & -\beta \\ \beta & \alpha \end{pmatrix}$$

$$\mathcal{D}_{\text{LSCM}} = (\partial_x u - \partial_y v)^2 + (\partial_y u + \partial_x v)^2$$

LSCM

Hold on a second ... several sneaky things happening here.

LSCM

Hold on a second ... several sneaky things happening here.

The f in the Cauchy-Riemann system represents our **inverse parameterization function** $f^{-1}(x, y) = (u, v)$. So what's the Jacobian again?

LSCM

Hold on a second ... several sneaky things happening here.

The f in the Cauchy-Riemann system represents our **inverse parameterization function** $f^{-1}(x, y) = (u, v)$. So what's the Jacobian again?

Theorem

Inverse Function Theorem. *The matrix inverse of the Jacobian matrix of an invertible function is the Jacobian matrix of the inverse function.*

LSCM

Hold on a second ... several sneaky things happening here.

The f in the Cauchy-Riemann system represents our **inverse parameterization function** $f^{-1}(x, y) = (u, v)$. So what's the Jacobian again?

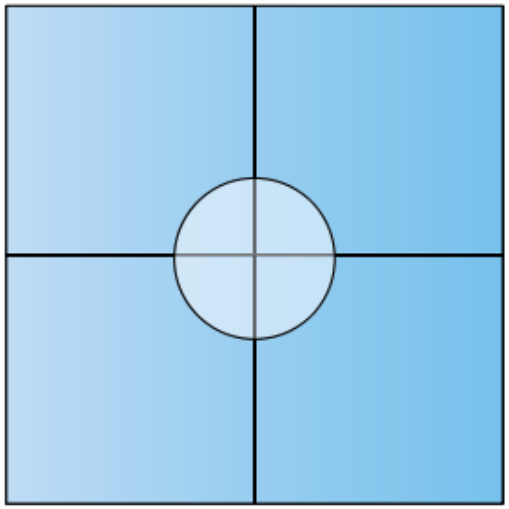
Theorem

Inverse Function Theorem. *The matrix inverse of the Jacobian matrix of an invertible function is the Jacobian matrix of the inverse function.*

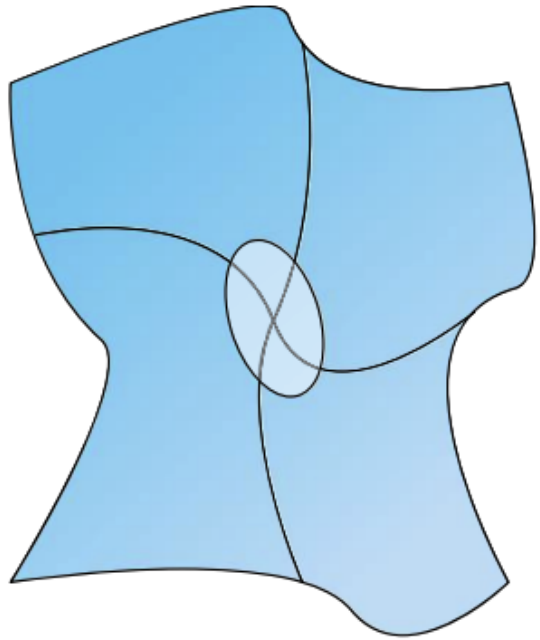
By definition of parameterization, f is injective. So $J_{f^{-1}} = J_f^{-1}$!

LSCM

$$\begin{pmatrix} u_x & u_y \\ v_x & v_y \end{pmatrix} = \begin{pmatrix} \nabla u \\ \nabla v \end{pmatrix}$$

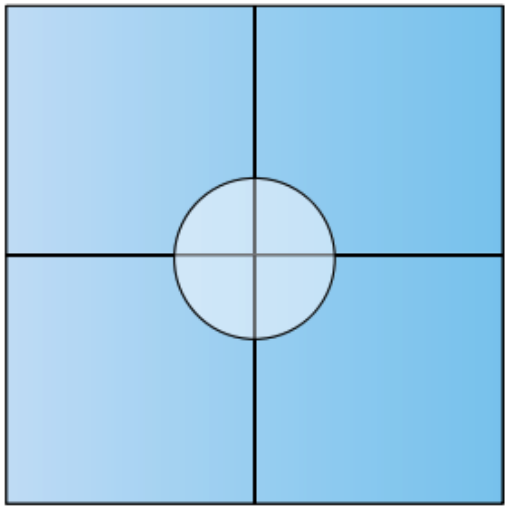


$$\mathbf{f}(x, y) = (u, v)$$



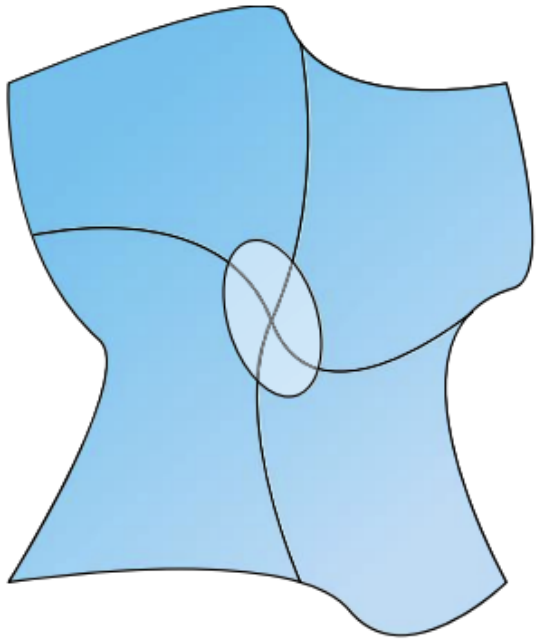
LSCM

$$\begin{pmatrix} u_x & u_y \\ v_x & v_y \end{pmatrix} = \begin{pmatrix} \nabla u \\ \nabla v \end{pmatrix}$$



Where did z go????

$$\mathbf{f}(x, y) = (u, v)$$

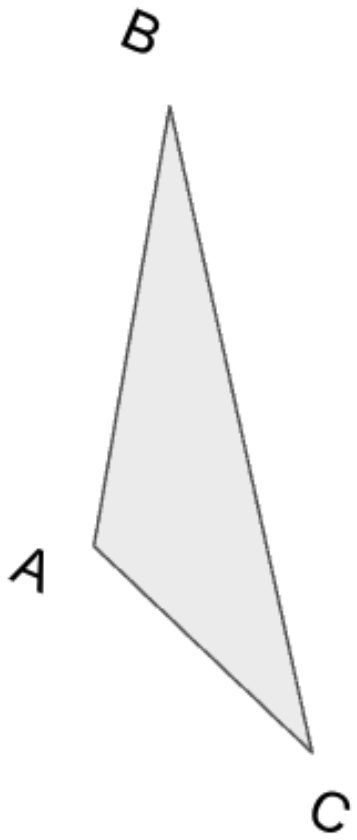


LSCM

Every triangle in the mesh can be defined by its own (local) coordinate system.

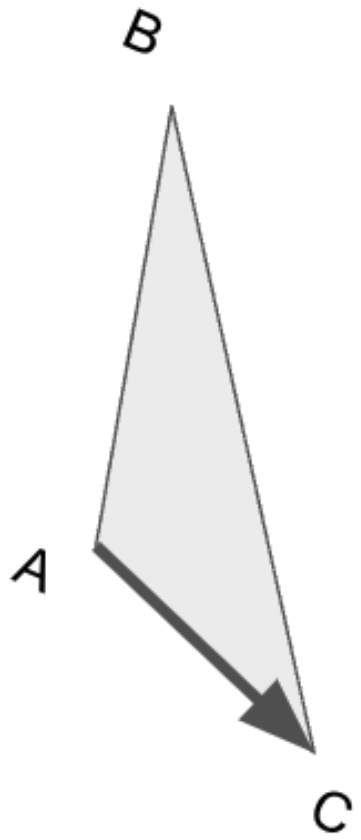
LSCM

Every triangle in the mesh can be defined by its own (local) coordinate system.



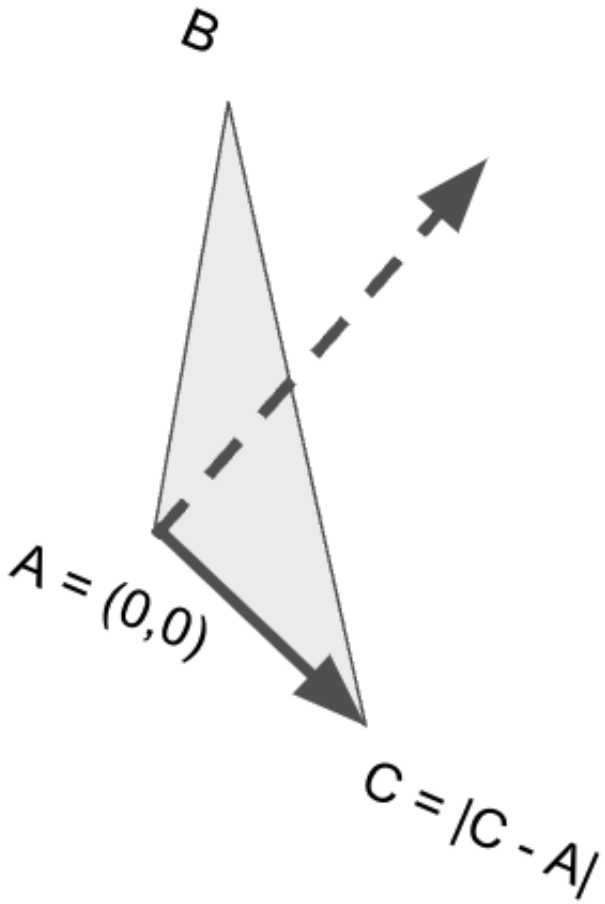
LSCM

Every triangle in the mesh can be defined by its own (local) coordinate system.



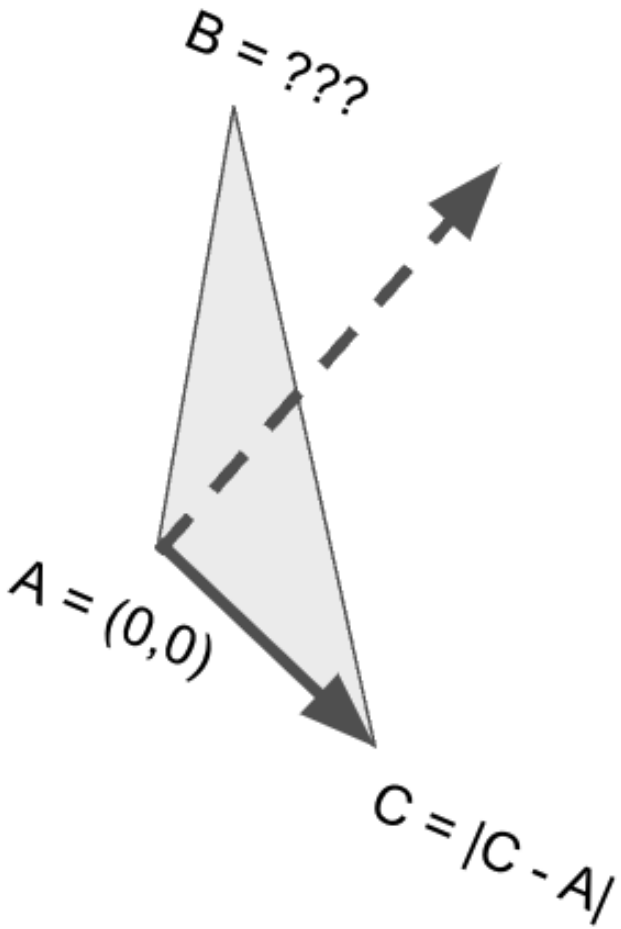
LSCM

Every triangle in the mesh can be defined by its own (local) coordinate system.



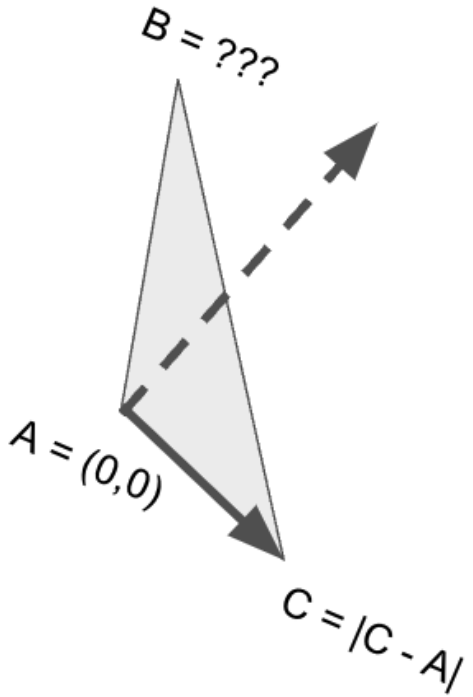
LSCM

Every triangle in the mesh can be defined by its own (local) coordinate system.



LSCM

Every triangle in the mesh can be defined by its own (local) coordinate system.



Use vector projection to get magnitude of (B-A) in the direction of the local coordinate vectors.

LSCM

One last problem: the current distortion formula has a degenerate solution.

LSCM - Least Squares Conformal Map

We want the Jacobian

$$\begin{pmatrix} \partial_x u & \partial_y u \\ \partial_x v & \partial_y v \end{pmatrix}$$

to be a similarity matrix

$$\begin{pmatrix} \alpha & -\beta \\ \beta & \alpha \end{pmatrix}$$

$$\mathcal{D}_{\text{LSCM}} = (\partial_x u - \partial_y v)^2 + (\partial_y u + \partial_x v)^2$$

LSCM

One last problem: the current distortion formula has a degenerate solution.

LSCM - Least Squares Conformal Map

We want the Jacobian

$$\begin{pmatrix} \partial_x u & \partial_y u \\ \partial_x v & \partial_y v \end{pmatrix}$$

to be a similarity matrix

$$\begin{pmatrix} \alpha & -\beta \\ \beta & \alpha \end{pmatrix}$$

$$\mathcal{D}_{\text{LSCM}} = (\partial_x u - \partial_y v)^2 + (\partial_y u + \partial_x v)^2$$

When can all the partial derivatives of a function be 0?

LSCM

One last problem: the current distortion formula has a degenerate solution.

LSCM - Least Squares Conformal Map

We want the Jacobian	to be a similarity matrix
$\begin{pmatrix} \partial_x u & \partial_y u \\ \partial_x v & \partial_y v \end{pmatrix}$	$\begin{pmatrix} \alpha & -\beta \\ \beta & \alpha \end{pmatrix}$

$$\mathcal{D}_{\text{LSCM}} = (\partial_x u - \partial_y v)^2 + (\partial_y u + \partial_x v)^2$$

f can just map everything to a constant, so $D_{LSCM} = 0$!

LSCM

One last problem: the current distortion formula has a degenerate solution.

LSCM - Least Squares Conformal Map

We want the Jacobian	to be a similarity matrix
$\begin{pmatrix} \partial_x u & \partial_y u \\ \partial_x v & \partial_y v \end{pmatrix}$	$\begin{pmatrix} \alpha & -\beta \\ \beta & \alpha \end{pmatrix}$

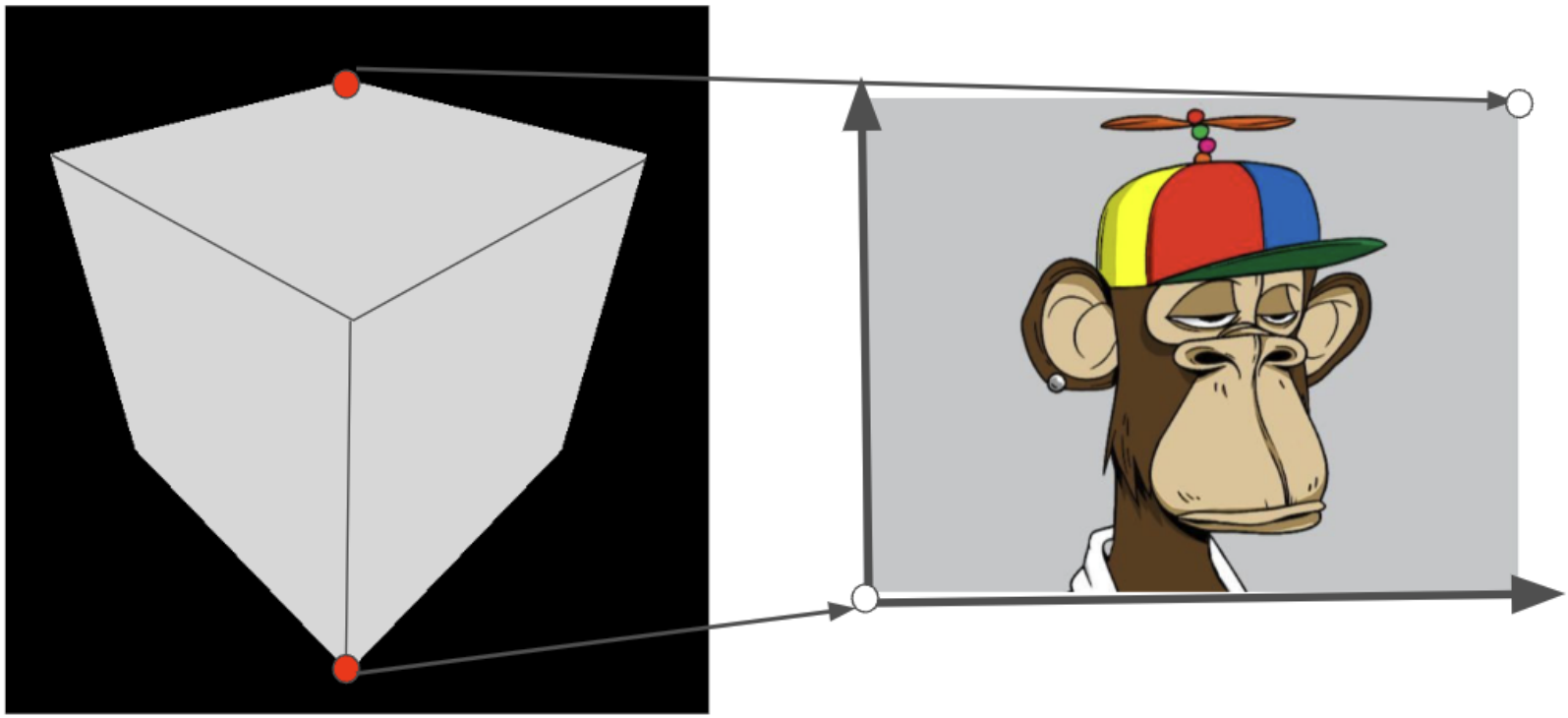
$$\mathcal{D}_{\text{LSCM}} = (\partial_x u - \partial_y v)^2 + (\partial_y u + \partial_x v)^2$$

f can just map everything to a constant, so $D_{LSCM} = 0$!

We can resolve this by setting **boundary constraints**, in this case **pinning two vertices** to the corners of our texel grid ((0,0) and (1,1)).

LSCM

We can resolve this by setting **boundary constraints**, in this case **pinning two vertices** to the corners of our texel grid ((0,0) and (1,1)).



LSCM

Finally, we can write the LSCM objective function as a linear least squares problem.

Rewriting the objective function with only real matrices and vectors yields

$$C(\mathbf{x}) = \|\mathcal{A}\mathbf{x} - \mathbf{b}\|^2, \tag{4}$$

with

$$\mathcal{A} = \begin{pmatrix} \mathcal{M}_f^1 & -\mathcal{M}_f^2 \\ \mathcal{M}_f^2 & \mathcal{M}_f^1 \end{pmatrix}, \quad \mathbf{b} = -\begin{pmatrix} \mathcal{M}_p^1 & -\mathcal{M}_p^2 \\ \mathcal{M}_p^2 & \mathcal{M}_p^1 \end{pmatrix} \begin{pmatrix} \mathbf{U}_p^1 \\ \mathbf{U}_p^2 \end{pmatrix},$$

LSCM

Finally, we can write the LSCM objective function as a linear least squares problem.

Rewriting the objective function with only real matrices and vectors yields

$$C(\mathbf{x}) = \|\mathcal{A}\mathbf{x} - \mathbf{b}\|^2, \tag{4}$$

with

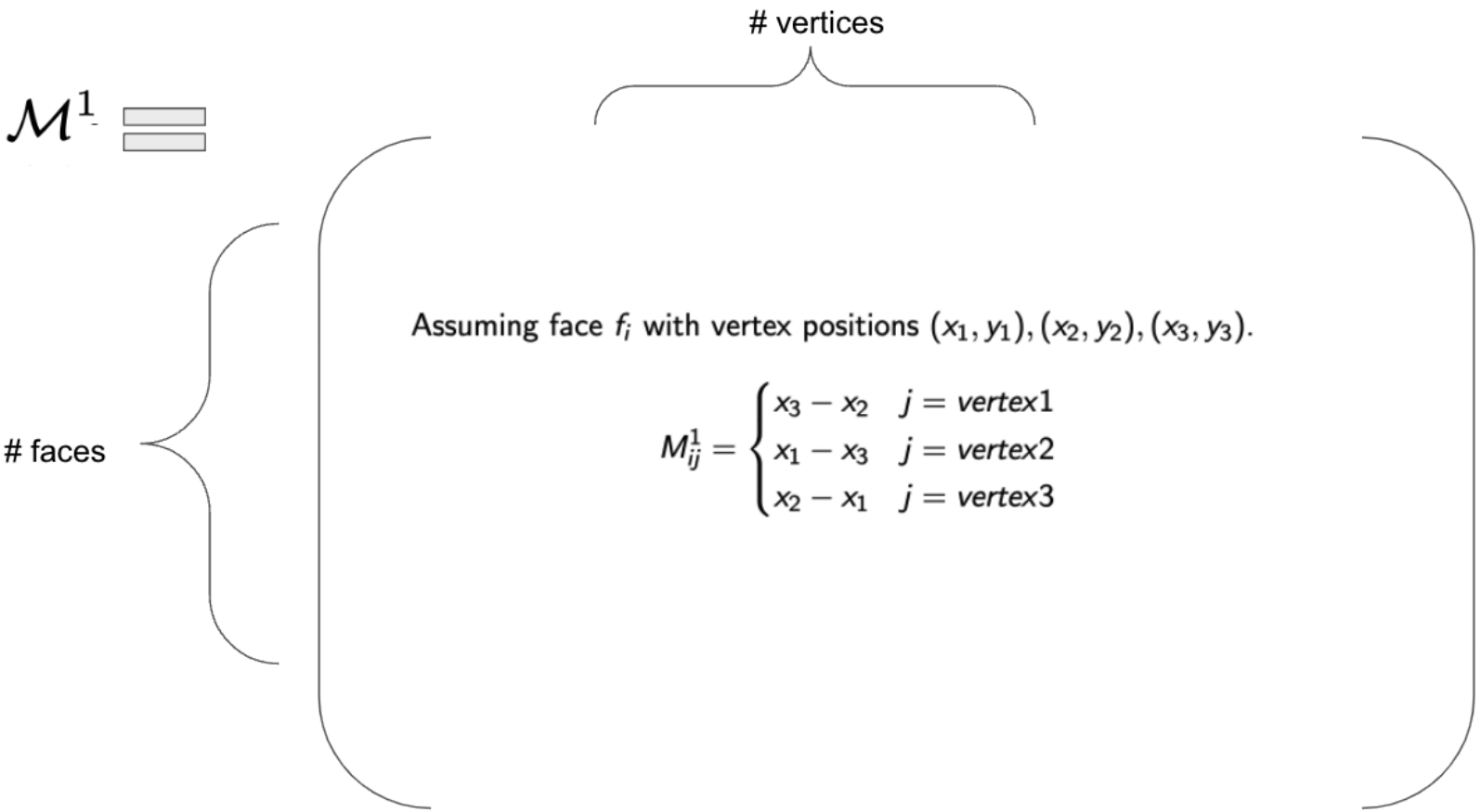
$$\mathcal{A} = \begin{pmatrix} \mathcal{M}_f^1 & -\mathcal{M}_f^2 \\ \mathcal{M}_f^2 & \mathcal{M}_f^1 \end{pmatrix}, \quad \mathbf{b} = -\begin{pmatrix} \mathcal{M}_p^1 & -\mathcal{M}_p^2 \\ \mathcal{M}_p^2 & \mathcal{M}_p^1 \end{pmatrix} \begin{pmatrix} \mathbf{U}_p^1 \\ \mathbf{U}_p^2 \end{pmatrix},$$

Look familiar?

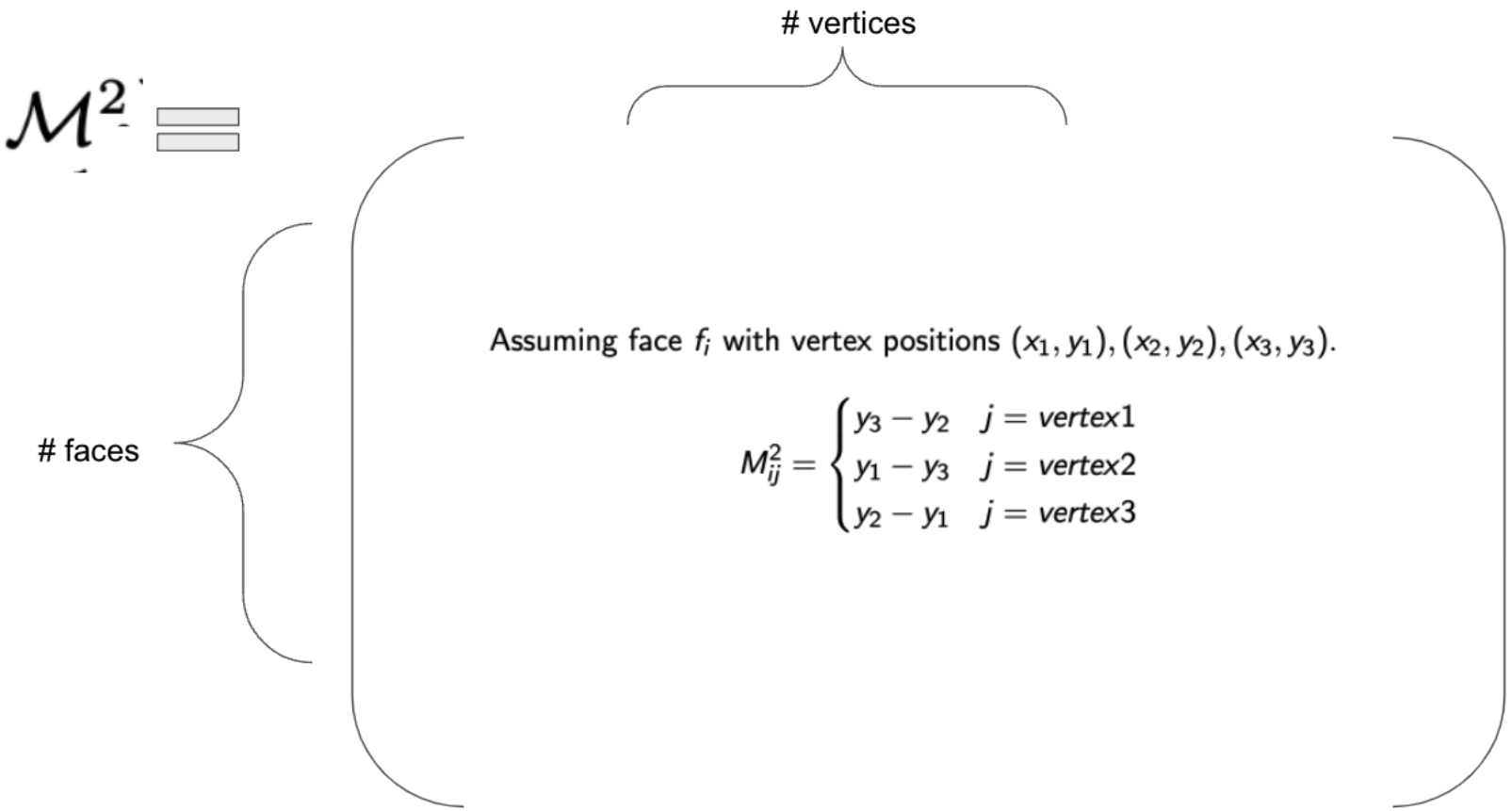
LSCM - Least Squares Conformal Map

We want the Jacobian $\begin{pmatrix} \partial_x u & \partial_y u \\ \partial_x v & \partial_y v \end{pmatrix}$ to be a similarity matrix $\begin{pmatrix} \alpha & -\beta \\ \beta & \alpha \end{pmatrix}$

LSCM


vertices
$\mathcal{M}^1$
faces
Assuming face f_i with vertex positions $(x_1, y_1), (x_2, y_2), (x_3, y_3)$.
$M_{ij}^1 = \begin{cases} x_3 - x_2 & j = vertex1 \\ x_1 - x_3 & j = vertex2 \\ x_2 - x_1 & j = vertex3 \end{cases}$

LSCM


vertices
faces
Assuming face f_i with vertex positions $(x_1, y_1), (x_2, y_2), (x_3, y_3)$.
$$M_{ij}^2 = \begin{cases} y_3 - y_2 & j = vertex1 \\ y_1 - y_3 & j = vertex2 \\ y_2 - y_1 & j = vertex3 \end{cases}$$

LSCM

$$\mathcal{M}_f^1 = \mathcal{M}^1 \begin{pmatrix} \\ \\ \end{pmatrix} \quad \text{Indices of non-pinned vertices}$$

$$\mathcal{M}_p^1 = \mathcal{M}^1 \begin{pmatrix} \\ \\ \end{pmatrix} \quad \text{Indices of pinned vertices}$$

$$\mathcal{M}_f^2 = \mathcal{M}^2 \begin{pmatrix} \\ \\ \end{pmatrix} \quad \text{Indices of non-pinned vertices}$$

$$\mathcal{M}_p^2 = \mathcal{M}^2 \begin{pmatrix} \\ \end{pmatrix} \quad \text{Indices of pinned vertices}$$

$$\begin{pmatrix} \mathbf{U}_p^1 \\ \mathbf{U}_p^2 \end{pmatrix} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \end{bmatrix}$$

LSCM

LSCM Pseudocode

def LSCM(V: vertices, F: faces):

LSCM

LSCM Pseudocode

def LSCM(V: vertices, F: faces):

1. Choose two vertices (b1, b2) to pin
    - Use vertices that are far apart for better results!

LSCM

LSCM Pseudocode

def LSCM(V: vertices, F: faces):

1. Choose two vertices (b1, b2) to pin
    - Use vertices that are far apart for better results!
2. For each triangle, convert the incident vertices to local coordinates

LSCM

LSCM Pseudocode

def LSCM(V: vertices, F: faces):

1. Choose two vertices (b1, b2) to pin
   - Use vertices that are far apart for better results!
2. For each triangle, convert the incident vertices to local coordinates
3. Construct the **A** and **b** matrices from (4).

LSCM

LSCM Pseudocode

def LSCM(V: vertices, F: faces):

1. Choose two vertices (b1, b2) to pin
   - Use vertices that are far apart for better results!
2. For each triangle, convert the incident vertices to local coordinates
3. Construct the **A** and **b** matrices from (4).
4. Solve the least squares equation for **x** in (4).
   - This will give you back your UV coordinates in a 1D vector in the form $(u_1, u_2, ..., u_{|V|}, v_1, v_2, ..., v_{|V|})$ **with the pinned vertices removed**
   - Make sure to **add the pinned vertices back** and reshape!

Thank you!

Interested in learning more (+ spicing things up with neural networks)? Join our lab @ https://3dl.cs.uchicago.edu/

Resources

"Least Squares Conformal Maps for Automatic Texture Atlas Generation", Levy et al 2002

Mesh Parameterization: Theory and Practice (2008)

Mesh Parameterization Methods and Their Applications