Computer Graphics

Texture mapping and parameterization

By Olga Sorkine
Some slides courtesy of Pierre Alliez and Craig Gotsman

The plan for today

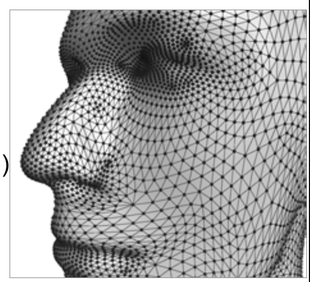
- Reminder: triangle meshes
- What is parameterization and what is it good for:
  - Texture mapping
  - Remeshing
- Parameterization
  - Convex mapping
  - Harmonic mapping

Triangle mesh

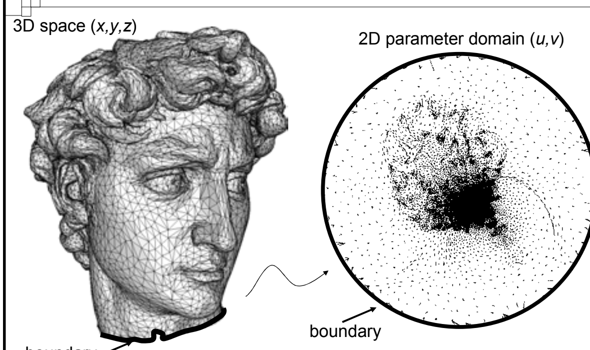
- Discrete surface representation
- Piecewise linear surface (made of triangles)

Triangle mesh

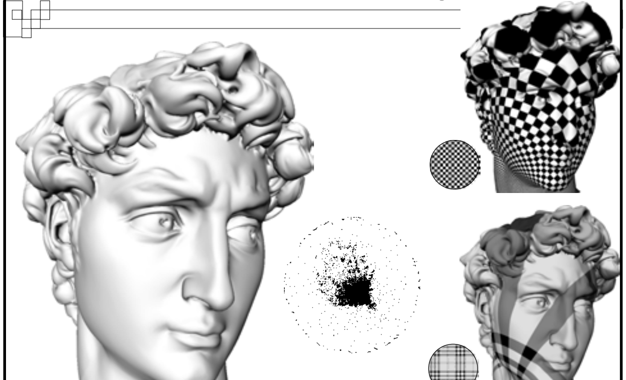
- Geometry:
  - Vertex coordinates
    - (x_1, y_1, z_1)
    - (x_2, y_2, z_2)
    - . . .
    - (x_n, y_n, z_n)
- Connectivity (the graph)
  - List of triangles
    - (i_1, j_1, k_1)
    - (i_2, j_2, k_2)
    - . . .
    - (i_m, j_m, k_m)



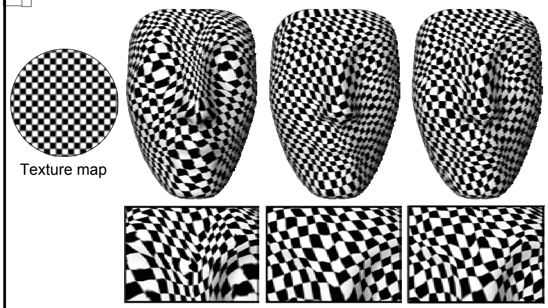
2D parameterization



Application - Texture mapping



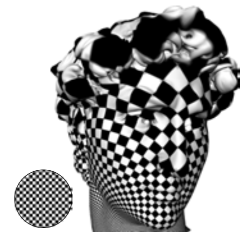
Distortion minimization



Kent et al ‘92 Floater 97 Sander et al ‘01

Requirements

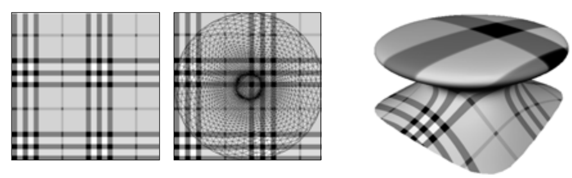
- Bijective (1-1 and onto): No triangles fold over.
- Minimal “distortion”
  - Preserve 3D angles
  - Preserve 3D distances
  - Preserve 3D areas
  - No “stretch”



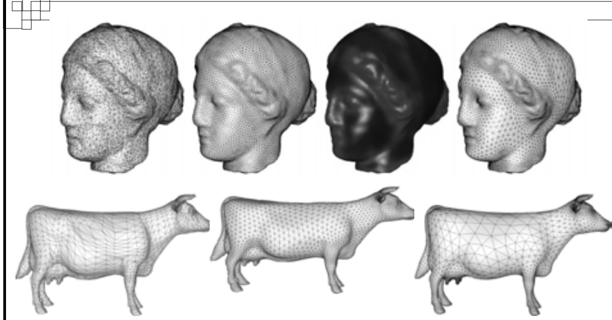
Applications

- Texture Mapping
- Remeshing
- Surface Reconstruction
- Morphing
- Compression

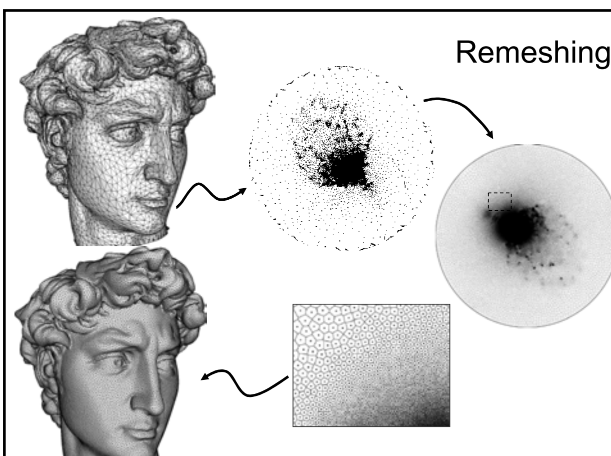
More texture mapping



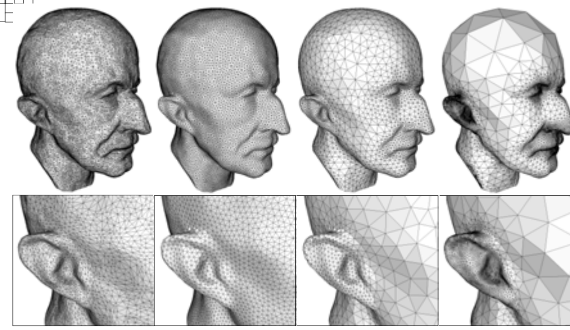
Remeshing examples



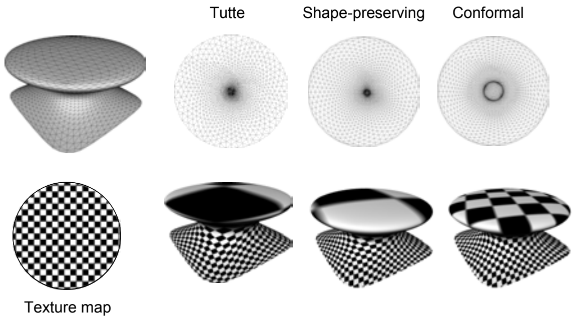
Remeshing



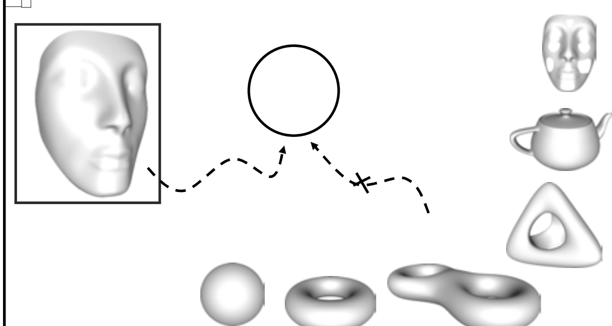
More remeshing examples



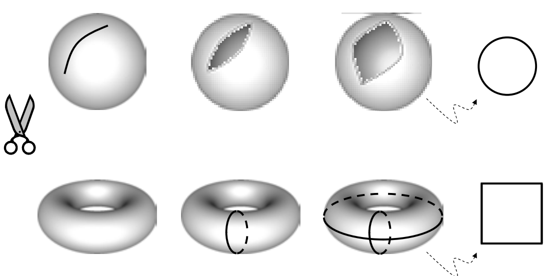
Conformal parametrization



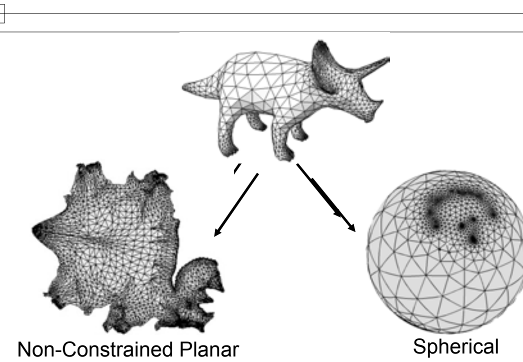
Non-simple domains



Cutting

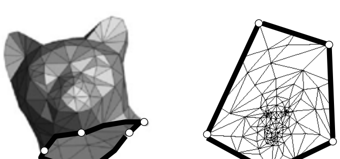


Parameterization of closed genus-0 triangle meshes



Convex mapping (Tutte, Floater)

- Works for meshes equivalent to a disk
- First, we map the boundary to a convex polygon
- Then we find the inner vertices positions



$\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_n$ – inner vertices; $\boldsymbol{v}_{n+1}, \ldots, \boldsymbol{v}_N$ – boundary vertices

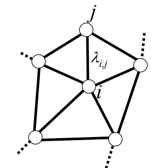
Inner vertices

- We constrain each inner vertex to be a weighted average of its neighbors:

$$\boxed{\boldsymbol{v}_i = \sum_{j \in N(i)} \lambda_{ij} \boldsymbol{v}_j, \quad i = 1,\ 2,\ \dots,\ n}$$

$$\lambda_{ij} = \begin{cases} 0 & i, j \ \text{ are not neighbors} \\ > 0 & (i, j) \in E \ \ \text{(neighbours)} \end{cases}$$

$$\sum_{j \in N(i)} \lambda_{ij} = 1$$



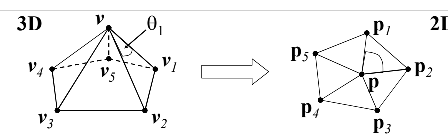
Linear system of equations

$$\boldsymbol{v}_i - \sum_{j \in N(i)} \lambda_{ij} \boldsymbol{v}_j = 0, \quad i = 1,\ 2,\ \dots,\ n$$

$$\boldsymbol{v}_i - \sum_{j \in N(i) \setminus B} \lambda_{ij} \boldsymbol{v}_j = \sum_{k \in N(i) \cap B} \lambda_{ik} \boldsymbol{v}_k, \quad i = 1,\ 2,\ \dots,\ n$$

$$\begin{pmatrix} 1 & & -\lambda_{1,j_1} & & -\lambda_{1,j_{d1}} \\ & 1 & & & \\ & & 1 & & \\ -\lambda_{4,j_1} & & & \ddots & \\ & & -\lambda_{n,j_5} & & 1 \end{pmatrix} \begin{pmatrix} \boldsymbol{v}_1 \\ \boldsymbol{v}_2 \\ \\ \vdots \\ \boldsymbol{v}_n \end{pmatrix} = \begin{pmatrix} \sigma_1 \\ \sigma_2 \\ \\ \\ \sigma_n \end{pmatrix}$$

Shape preserving weights



To compute $\lambda_1, \dots, \lambda_5$, a local embedding of the patch is found:

1) $\| \mathbf{p}_i - \mathbf{p} \| = \| \mathbf{x}_i - \mathbf{x} \|$

2) $angle(\mathbf{p}_i, \mathbf{p}, \mathbf{p}_{i+1}) = (2\pi / \Sigma\theta_i)\ angle(\boldsymbol{v}_i, \boldsymbol{v}, \boldsymbol{v}_{i+1})$

$$\exists\ \lambda_i, \ \begin{cases} \mathbf{p} = \Sigma\, \lambda_i\, \mathbf{p}_i \\ \lambda_i > 0 \\ \Sigma\, \lambda_i = 1 \end{cases} \quad \Rightarrow \text{ use these } \lambda \text{ as edge weights.}$$

Linear system of equations

- A unique solution always exists
- Important: the solution is legal (bijective)

- The system is sparse, thus fast numerical solution is possible
- Numerical problems (because the vertices in the middle might get very dense…)

Harmonic mapping

- Another way to find inner vertices
- Strives to preserve angles (conformal)
- We treat the mesh as a system of springs.
- Define spring energy:

$$\boxed{E_{harm} = \frac{1}{2} \sum_{(i,j) \in E} k_{i,j} \left\| \boldsymbol{v}_i - \boldsymbol{v}_j \right\|^2}$$

where $\boldsymbol{v}_i$ are the flat position (remember that the boundary vertices $\boldsymbol{v}_{n+1}, \dots, \boldsymbol{v}_N$ are constrained).

Energy minimization – least squares

- We want to find such flat positions that the energy is as small as possible.
- Solve the linear least squares problem!

$$\boldsymbol{v}_i = (x_i, y_i)$$

$$E_{harm}(x_1, \dots, x_n, y_1, \dots, y_n) = \frac{1}{2} \sum_{(i,j) \in E} k_{i,j} \left\| \boldsymbol{v}_i - \boldsymbol{v}_j \right\|^2 =$$

$$= \frac{1}{2} \sum_{(i,j) \in E} k_{i,j} \left((x_i - x_j)^2 + (y_i - y_j)^2 \right).$$

E_{harm} is function of 2n variables

Energy minimization – least squares

- To find minimum: $\nabla E_{harm} = 0$

$$\boxed{\begin{aligned} \frac{\partial}{\partial x_i} E_{harm} &= \frac{1}{2} \sum_{j \in N(i)} 2 k_{i,j} (x_i - x_j) = 0 \\ \frac{\partial}{\partial y_i} E_{harm} &= \frac{1}{2} \sum_{j \in N(i)} 2 k_{i,j} (y_i - y_j) = 0 \end{aligned}}$$

- Again, $x_{n+1}, \ldots, x_N$ and $y_{n+1}, \ldots, y_N$ are constrained.

Energy minimization – least squares

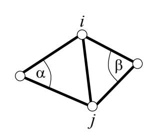
- To find minimum: $\nabla E_{harm} = 0$

$$\boxed{\begin{aligned} \sum_{j \in N(i)} k_{i,j} (x_i - x_j) &= 0, \quad i = 1, 2, \ldots, n \\ \sum_{j \in N(i)} k_{i,j} (y_i - y_j) &= 0, \quad i = 1, 2, \ldots, n \end{aligned}}$$

- Again, $x_{n+1}, \ldots, x_N$ and $y_{n+1}, \ldots, y_N$ are constrained.

The spring constants $k_{i,j}$

- The weights $k_{i,j}$ are chosen to minimize angles distortion:
  - Look at the edge (i, j) in the 3D mesh
  - Set the weight $\boxed{k_{i,j} = \cot \alpha + \cot \beta}$



3D

Discussion

- The results of harmonic mapping are better than those of convex mapping (local area and angles preservation).
- But: the mapping is not always legal (the weights can be negative for badly-shaped triangles…)
- Both mappings have the problem of fixed boundary – it constrains the minimization and causes distortion.
- There are more advanced methods that do not require boundary conditions.

See you next time