UV Unwrapped: A Comprehensive Study of UV Mapping for 3D Models

Venika Sruthi Annam

May, 2023

Contents

1 Introduction

It is possible to find a correspondence between two surfaces with similar topology by establishing a one-to-one and onto mapping. If one of the surfaces is represented by a triangular mesh, the task of finding such a mapping is called mesh parameterization. The surface that the mesh is mapped to is known as the parameter domain. Parameterizations between surface meshes and various domains have many practical uses in fields such as computer graphics and geometry processing. There are many methods available for parameterizing meshes, each targeting different parameter domains and emphasizing different parameterization properties.

1.1 Applications

Surface parameterization was introduced to computer graphics as a method for mapping textures onto surfaces. Over the last decade, it has gradually become a ubiquitous tool, useful for many mesh processing applications, discussed below

1.1.1 Mesh Completion

Meshes from range scans often contain holes and multiple components.In many cases, prior knowledge about the overall shape of the scanned models exists. For instance, for human scans, templates of a generic human shape are readily available. We can use this prior knowledge to facilitate completion of scans by computing a mapping between the scan and a template human model. Researchers have developed a more generic and robust template-based approach for completion of any type of scans. The techniques typically use an inter-surface parameterization between the template and the scan.

1.1.2 Creation of Object Databases

Once a large number of models are parameterized on a common domain one can perform an analysis determining the common factors between objects and their distinguishing traits. For example on a database of human shapes the distinguishing traits may be gender, height, and weight. Objects can be compared against the database and scored against each of these dimensions, and the database can be used to create new plausible object instances by interpolation or extrapolation of existing ones.

1.1.3 Remeshing

Remeshing is a process in computer graphics and geometry processing that involves generating a new mesh with a different topology or structure from an existing one. The goal of remeshing is typically to improve the quality of the mesh, making it more suitable for various applications such as simulation, rendering, and animation.

One common approach to remeshing is to first compute a new mesh that approximates the original mesh's geometry and then refine it to achieve a better topology. This can be done by using various algorithms, such as Delaunay triangulation, Voronoi diagrams, or optimization-based techniques. These methods attempt to minimize the number of triangles or control their size and shape to achieve a more uniform distribution.

Another goal of remeshing is to reduce the complexity of the mesh. A high-resolution mesh can be very computationally expensive, making it challenging to manipulate and render. By reducing the number of vertices and triangles while preserving the surface's key features, remeshing can simplify the mesh without significantly affecting its appearance.

Remeshing has a wide range of applications, including 3D printing, computer-aided design, and medical imaging. It is often used as a pre-processing step before simulation or rendering to improve the accuracy and speed of these processes.

2 Fundamentals of UV Parametarization

2.1 Basic Definitions

Suppose that $\Omega \subset R^2$ is some simply connected region (i.e., without any holes), for example, the unit square: $\Omega = (u, v) \in R^2 : u, v \in [0, 1]$,or the unit disk: $\Omega = (u, v) \in R^2 : u^2 + v^2 \leq 1$, and that the

function $f : \Omega \to R^3$ is continuous and an injection (i.e., no two distinct points in Ω are mapped to the same point in R^3). We then call the image S of Ω under f a surface,$S = f(\Omega) = f(u, v) : (u, v) \in \Omega$, and say that f is a parameterization of S over the parameter domain Ω. It follows from the definition of S that f is actually a bijection between Ω and S and thus admits to define its inverse $f^{-1} : S \to \Omega$.

cylinder:

$$\begin{aligned} \textit{parameter domain:} \quad & \Omega = \{(u, v) \in \mathbb{R}^2 : u \in [0, 2\pi), v \in [0, 1]\} \\ \textit{surface:} \quad & S = \{(x, y, z) \in \mathbb{R}^3 : x^2 + y^2 = 1, z \in [0, 1]\} \\ \textit{parameterization:} \quad & f(u, v) = (\cos u, \sin u, v) \\ \textit{inverse:} \quad & f^{-1}(x, y, z) = (\arccos x, z) \end{aligned}$$

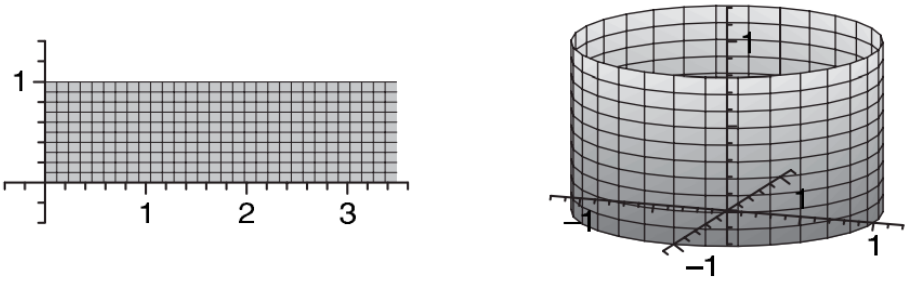


Figure 1: Cylinder UV Mapping.

paraboloid:

$$\begin{aligned} \textit{parameter domain:} \quad & \Omega = \{(u, v) \in \mathbb{R}^2 : u, v \in [-1, 1]\} \\ \textit{surface:} \quad & S = \{(x, y, z) \in \mathbb{R}^3 : x, y \in [-2, 2], z = \tfrac{1}{4}(x^2 + y^2)\} \\ \textit{parameterization:} \quad & f(u, v) = (2u, 2v, u^2 + v^2) \\ \textit{inverse:} \quad & f^{-1}(x, y, z) = (\tfrac{x}{2}, \tfrac{y}{2}) \end{aligned}$$

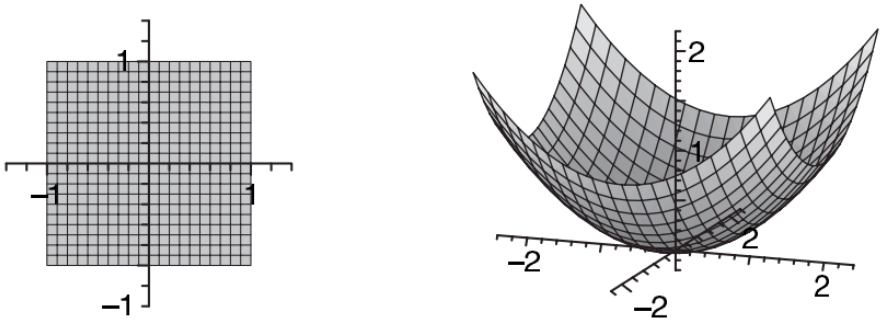


Figure 2: Paraboloid UV Mapping.

hemisphere (orthographic):

$$\begin{aligned} \textit{parameter domain:} &\quad \Omega = \{(u,v) \in \mathbb{R}^2 : u^2 + v^2 \leq 1\} \\ \textit{surface:} &\quad S = \{(x,y,z) \in \mathbb{R}^3 : x^2 + y^2 + z^2 = 1, z \geq 0\} \\ \textit{parameterization:} &\quad f(u,v) = (u, v, \sqrt{1-u^2-v^2}) \\ \textit{inverse:} &\quad f^{-1}(x,y,z) = (x,y) \end{aligned}$$

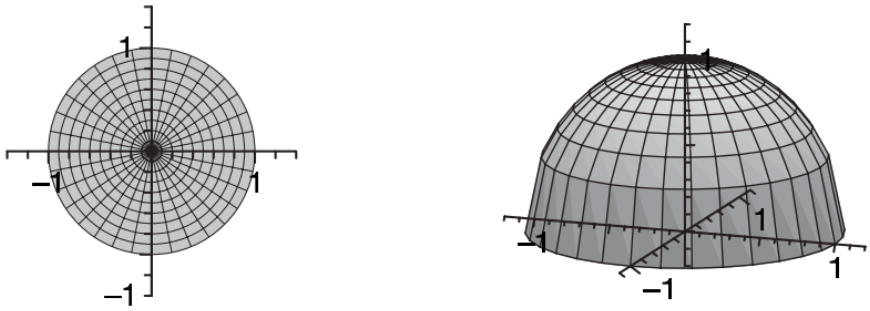


Figure 3: Hemisphere UV Mapping.

hemisphere (stereographic):

$$\begin{aligned} \textit{parameter domain:} &\quad \Omega = \{(u,v) \in \mathbb{R}^2 : u^2 + v^2 \leq 1\} \\ \textit{surface:} &\quad S = \{(x,y,z) \in \mathbb{R}^3 : x^2 + y^2 + z^2 = 1, z \geq 0\} \\ \textit{parameterization:} &\quad f(u,v) = (\tfrac{2u}{1+u^2+v^2}, \tfrac{2v}{1+u^2+v^2}, \tfrac{1-u^2-v^2}{1+u^2+v^2}) \\ \textit{inverse:} &\quad f^{-1}(x,y,z) = (\tfrac{x}{1+z}, \tfrac{y}{1+z}) \end{aligned}$$

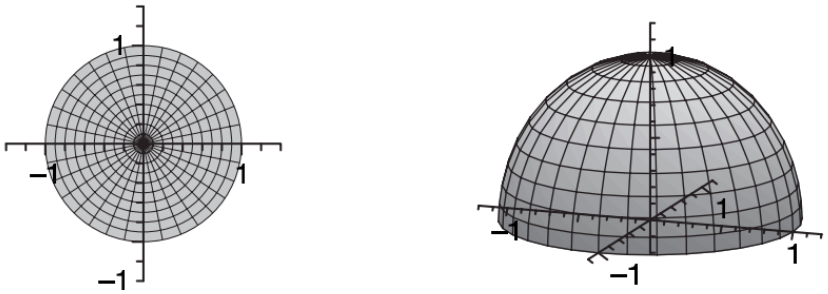


Figure 4: Hemisphere UV Mapping.

2.2 Intrinsic Properties

Although the parameterization of a surface is not unique—and we will later discuss how to get the "best" parameterization with respect to certain criteria—it nevertheless is a very handy thing to have as it allows us to compute a variety of properties of the surface. For example, if f is differentiable, then its partial derivatives

$$\frac{\partial f}{\partial u} \text{ and } \frac{\partial f}{\partial v}$$

span the local tangent plane, and by simply taking their cross product and normalizing the result we get the surface normal

$$\mathbf{n}_f = \frac{\frac{\partial f}{\partial u} \times \frac{\partial f}{\partial v}}{\left| \frac{\partial f}{\partial u} \times \frac{\partial f}{\partial v} \right|}.$$

To simplify the notation, we will often speak of f_u and f_v as the derivatives and of $\mathbf{n}_f$ as the surface normal, but we should keep in mind that formally all three are functions from R^2 to R^3. In other words, for any point $(u, v) \in \Omega$ in the parameter domain, the tangent plane at the surface point $f(u, v) \in S$ is spanned by the two vectors $f_u(u, v)$ and $f_v(u, v)$, and $\mathbf{n}_f(u, v)$ is the normal vector at this point[1]. Again, let us clarify this by considering two examples:

For the simple linear function $f(u, v) = (u, 1 - u, v)$ we get $f_u(u, v) = (1, -1, 0)$ and $f_v(u, v) = (0, 0, 1)$ and further $\mathbf{n}_f(u, v) = \left(\frac{-1}{\sqrt{2}}, \frac{1}{\sqrt{2}}, 0\right)$, showing that the normal vector is constant for all points on S. For the parameterization of the cylinder, $f(u, v) = (\cos u, \sin u, v)$, we get $f_u(u, v) = (-\sin u, \cos u, 0)$ and $f_v(u, v) = (0, 0, 1)$ and further $\mathbf{n}_f(u, v) = (\cos u, \sin u, 0)$, showing that the normal vector at any point $(x, y, z) \in S$ is just $(x, y, 0)$.

Note that in both examples the surface normal is independent of the parameterization. In fact, this holds for all surfaces and is therefore called an intrinsic property of the surface. Formally, we can also say that the surface normal is a function $n : S \to S^2$, where $S^2 = (x, y, z) \in R^3 : x^2 + y^2 + z^2 = 1$ is the unit sphere in R^3, so that $n(p) = n_f(f^{-1}(p))$ for any $p \in S$ and any parameterization f. As an exercise, you may want to verify this for the two alternative parameterizations of the hemisphere given above. Other intrinsic surface properties are the Gaussian curvature $K(p)$ and the mean curvature $H(p)$ as well as the total area of the surface $A(S)$. To compute the latter, we need the first fundamental form

$$I_f = \begin{pmatrix} \langle f_u, f_u \rangle & \langle f_u, f_v \rangle \\ \langle f_u, f_v \rangle & \langle f_v, f_v \rangle \end{pmatrix} = \begin{pmatrix} E & F \\ F & G \end{pmatrix},$$

where the product between the partial derivatives is the usual dot product in R^3. It follows immediately from the Cauchy-Schwarz inequality that the determinant of this symmetric 2×2 matrix is always non-negative, so that its square root is always real. The area of the surface is then defined as

$$A(S) = \iint_\Omega \sqrt{\det I_f}, \mathrm{d}u\mathrm{d}v.$$

Take, for example, the orthographic parameterization $f(u, v) = (u, v, \sqrt{1 - u^2 - v^2})$ of the hemisphere over the unit disk. After some simplifications we find that $\det I_f = \frac{1}{1-u^2-v^2}$ and can compute the area of the hemisphere as follows:

$$\begin{aligned} A(S) &= \iint_\Omega \sqrt{\det I_f}, \mathrm{d}u\mathrm{d}v \\ &= \int_{-1}^{1} \int_{-\sqrt{1-v^2}}^{\sqrt{1-v^2}} \sqrt{\frac{1}{1-u^2-v^2}}, \mathrm{d}u\mathrm{d}v \\ &= \int_{-1}^{1} \int_{-\sqrt{1-v^2}}^{\sqrt{1-v^2}} \frac{1}{\sqrt{1-u^2-v^2}}, \mathrm{d}u\mathrm{d}v \\ &= \int_{-1}^{1} \arcsin(u) \Big|_{-\sqrt{1-v^2}}^{\sqrt{1-v^2}}, \mathrm{d}v \\ &= \int -1^1 \pi, \mathrm{d}v \\ &= 2\pi, \end{aligned}$$

as expected.

In order to compute the curvatures we must first assume the parameterization to be twice differentiable, so that its *second order* partial derivatives

$$f_{uu} = \frac{\partial^2 f}{\partial u^2}, \qquad f_{uv} = \frac{\partial^2 f}{\partial u \partial v}, \qquad \text{and} \qquad f_{vv} = \frac{\partial^2 f}{\partial v^2}$$

are well defined. Taking the dot products of these derivatives with the surface normal then gives the symmetric 2×2 matrix that is known as the *second fundamental form*

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

Gaussian and mean curvature are finally defined as the determinant and half the trace of the matrix $\mathbf{I}_f^{-1}\mathbf{II}_f$, respectively:

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

and

$$H = \tfrac{1}{2}\operatorname{trace}(\mathbf{I}_f^{-1}\mathbf{II}_f) = \frac{LG - 2MF + NE}{2(EG - F^2)}.$$

For example, carrying out these computations reveals that the curvatures are constant for most of the surfaces from above:

$$\begin{aligned} \text{simple linear function:} \quad & K = 0, \quad H = 0, \\ \text{cylinder:} \quad & K = 0, \quad H = \tfrac{1}{2}, \\ \text{hemisphere:} \quad & K = 1, \quad H = -1. \end{aligned}$$

As an exercise, show that the curvatures at any point $\boldsymbol{p} = (x, y, z)$ of the paraboloid from above are $K(\boldsymbol{p}) = \frac{1}{4(1+z)^2}$ and $H(\boldsymbol{p}) = \frac{2+z}{4(1+z)^{3/2}}$.

2.3 Metric Distortion

Apart from these intrinsic surface properties, there are others that depend on the parameterization, most importantly the *metric distortion*. Consider, for example, the two parameterizations of the hemisphere above. In both cases, the image of the surface on the right is overlaid by a regular grid, which actually is the image of the corresponding grid in the parameter domain shown on the left. You will notice that the surface grid looks more regular for the stereographic than for the orthographic projection and that the latter considerably stretches the grid in the radial direction near the boundary.

To better understand this kind of stretching, let us see what happens to the surface point $f(u, v)$ as we move a tiny little bit away from (u, v) in the parameter domain. If we denote this infinitesimal parameter displacement by $(\Delta u, \Delta v)$, then the new surface point $f(u + \Delta u, v + \Delta v)$ is approximately given by the first order Taylor expansion $\tilde{f}$ of f around (u, v),

$$\tilde{f}(u + \Delta u, v + \Delta v) = f(u, v) + f_u(u, v)\Delta u + f_v(u, v)\Delta v.$$

This linear function maps all points in the vicinity of $\boldsymbol{u} = (u, v)$ into the tangent plane $T_{\boldsymbol{p}}$ at $\boldsymbol{p} = f(u, v) \in S$ and transforms circles around $\boldsymbol{u}$ into ellipses around $\boldsymbol{p}$ (see Figure 2.1). The latter property becomes obvious if we write the Taylor expansion more compactly as

$$\tilde{f}(u + \Delta u, v + \Delta v) = \boldsymbol{p} + J_f(\boldsymbol{u})\tbinom{\Delta u}{\Delta v},$$

where $J_f = (f_u\ f_v)$ is the *Jacobian* of f, i.e. the 3×2 matrix with the partial derivatives of f as column vectors. Then using the *singular value decomposition* of the Jacobian,

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

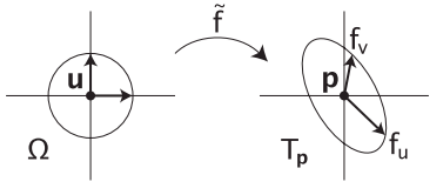
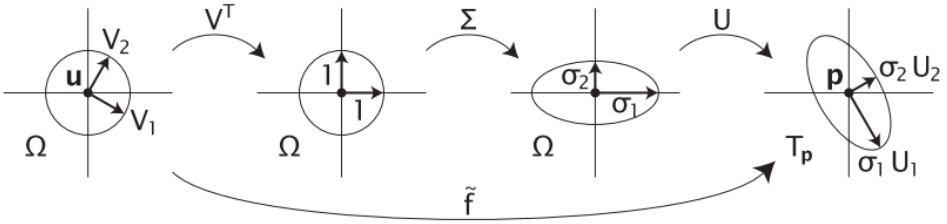


Figure 2.1: First order Taylor expansion $\tilde{f}$ of the parameterization f.



with *singular values* $\sigma_1 \geq \sigma_2 > 0$ and *orthonormal* matrices $U \in \mathbb{R}^{3\times 3}$ and $V \in \mathbb{R}^{2\times 2}$ with column vectors U_1, U_2, U_3, and V_1, V_2, respectively, we can split up the linear transformation $\tilde{f}$ as shown in Figure 2.2:

1. The transformation V^T first rotates all points around $\boldsymbol{u}$ such that the vectors V_1 and V_2 are in alignment with the u- and the v-axes afterwards.

2. The transformation Σ then stretches everything by the factor σ_1 in the u- and by σ_2 in the v-direction.

3. The transformation U finally maps the unit vectors $(1, 0)$ and $(0, 1)$ to the vectors U_1 and U_2 in the tangent plane $T_{\boldsymbol{p}}$ at $\boldsymbol{p}$.

As a consequence, any circle of radius r around $\boldsymbol{u}$ will be mapped to an ellipse with semi-axes of length $r\sigma_1$ and $r\sigma_2$ around $\boldsymbol{p}$ and the orthonormal frame $[V_1, V_2]$ is mapped to the orthogonal frame $[\sigma_1 U_1, \sigma_2 U_2]$.

3 Barycentric Mappings

3.1 Triangle Meshes

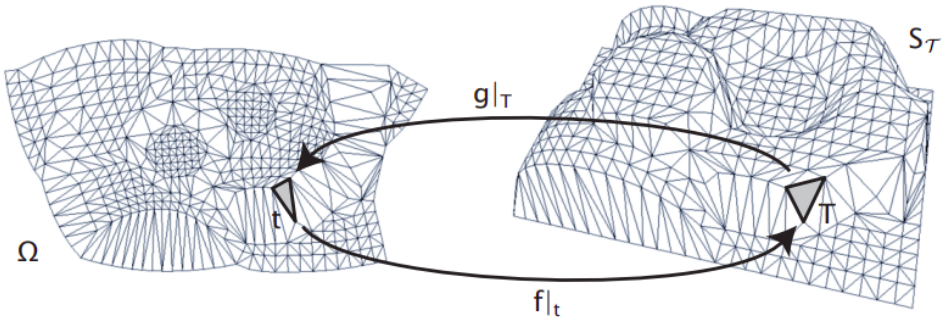
As in the previous chapter, let us denote points in $\mathbb{R}^3$ by $\boldsymbol{p} = (x, y, z)$ and points in $\mathbb{R}^2$ by $\boldsymbol{u} = (u, v)$. An *edge* is then defined as the convex hull of (or, equivalently, the line segment between) two distinct points and a *triangle* as the convex hull of three non-collinear points. We will denote edges and triangles in $\mathbb{R}^3$ with capital letters and those in $\mathbb{R}^2$ with small letters, for example, $e = [\boldsymbol{u}_1, \boldsymbol{u}_2]$ and $T = [\boldsymbol{p}_1, \boldsymbol{p}_2, \boldsymbol{p}_3]$.

A *triangle mesh* $S_{\mathcal{T}}$ is the union of a set of *surface triangles* $\mathcal{T} = \{T_1, \ldots, T_m\}$ which intersect only at common edges $\mathcal{E} = \{E_1, \ldots, E_l\}$ and vertices $\mathcal{V} = \{\boldsymbol{p}_1, \ldots, \boldsymbol{p}_{n+b}\}$. More specifically, the set of vertices consists of n *interior* vertices $\mathcal{V}_I = \{\boldsymbol{p}_1, \ldots, \boldsymbol{p}_n\}$ and b *boundary* vertices $\mathcal{V}_B = \{\boldsymbol{p}_{n+1}, \ldots, \boldsymbol{p}_{n+b}\}$. Two distinct vertices $\boldsymbol{p}_i, \boldsymbol{p}_j \in \mathcal{V}$ are called *neighbours*, if they are the end points of some edge $E = [\boldsymbol{p}_i, \boldsymbol{p}_j] \in \mathcal{E}$, and for any $\boldsymbol{p}_i \in \mathcal{V}$ we let $N_i = \{j : [\boldsymbol{p}_i, \boldsymbol{p}_j] \in \mathcal{E}\}$ be the set of indices of all neighbours of $\boldsymbol{p}_i$.

A parameterization f of $S_{\mathcal{T}}$ is usually specified the other way around, that is, by defining the inverse parameterization $g = f^{-1}$. This mapping g is uniquely determined by specifying the *parameter points* $\boldsymbol{u}_i = g(\boldsymbol{p}_i)$ for each vertex $\boldsymbol{p}_i \in \mathcal{V}$ and demanding that g is continuous and linear for each triangle. In this setting, $g|_T$ is the linear map from a surface triangle $T = [\boldsymbol{p}_i, \boldsymbol{p}_j, \boldsymbol{p}_k]$ to the corresponding *parameter triangle* $t = [\boldsymbol{u}_i, \boldsymbol{u}_j, \boldsymbol{u}_k]$ and $f|_t = (g|_T)^{-1}$ is the inverse linear map from t to T. The parameter domain Ω finally is the union of all parameter triangles (see Figure 3.1).

3.2 Parameterization by Affine Combinations

A rather simple idea for constructing a parameterization of a triangle mesh is based on the following physical model. Imagine that the edges of the triangle mesh are springs that are connected at the vertices. If we now fix the boundary of this spring network somewhere in the plane, then the interior of this network will relax in the energetically



most efficient configuration, and we can simply assign the positions where the joints of the network have come to rest as parameter points.

If we assume each spring to be ideal in the sense that the rest length is zero and the potential energy is just $\frac{1}{2}Ds^2$, where D is the spring constant and s the length of the spring, then we can formalize this approach as follows. We first specify the parameter points $\boldsymbol{u}_i = (u_i, v_i)$, $i = n+1, \ldots, n+b$ for the boundary vertices $\boldsymbol{p}_i \in \mathcal{V}_B$ of the mesh in some way (see Section 3.4). Then we minimize the overall spring energy

$$E = \tfrac{1}{2} \sum_{i=1}^{n} \sum_{j \in N_i} \tfrac{1}{2} D_{ij} \|\boldsymbol{u}_i - \boldsymbol{u}_j\|^2,$$

where $D_{ij} = D_{ji}$ is the spring constant of the spring between $\boldsymbol{p}_i$ and $\boldsymbol{p}_j$, with respect to the unknown parameter positions $\boldsymbol{u}_i = (u_i, v_i)$ for the interior points[1]. As the partial derivative of E with respect to $\boldsymbol{u}_i$ is

$$\frac{\partial E}{\partial \boldsymbol{u}_i} = \sum_{j \in N_i} D_{ij}(\boldsymbol{u}_i - \boldsymbol{u}_j),$$

the minimum of E is obtained if

$$\sum_{j \in N_i} D_{ij} \boldsymbol{u}_i = \sum_{j \in N_i} D_{ij} \boldsymbol{u}_j$$

holds for all $i = 1, \ldots, n$. This is equivalent to saying that each interior parameter point $\boldsymbol{u}_i$ is an *affine combination* of its neighbours,

$$\boldsymbol{u}_i = \sum_{j \in N_i} \lambda_{ij} \boldsymbol{u}_j, \tag{3.1}$$

with normalized coefficients

$$\lambda_{ij} = D_{ij} \Big/ \sum_{k \in N_i} D_{ik}$$

that obviously sum to 1.

By separating the parameter points for the interior and the boundary vertices in the sum on the right hand side of (3.1) we get

$$\boldsymbol{u}_i - \sum_{j \in N_i, j \le n} \lambda_{ij} \boldsymbol{u}_j = \sum_{j \in N_i, j > n} \lambda_{ij} \boldsymbol{u}_j,$$

and see that computing the coordinates u_i and v_i of the interior parameter points $\boldsymbol{u}_i$ requires to solve the linear systems

$$AU = \bar{U} \qquad \text{and} \qquad AV = \bar{V}, \tag{3.2}$$

where $U = (u_1, \ldots, u_n)$ and $V = (v_1, \ldots, v_n)$ are the column vectors of unknown coordinates, $\bar{U} = (\bar{u}_1, \ldots, \bar{u}_n)$ and $\bar{V} = (\bar{v}_1, \ldots, \bar{v}_n)$ are the column vectors with coefficients

$$\bar{u}_i = \sum_{j \in N_i, j > n} \lambda_{ij} u_j \qquad \text{and} \qquad \bar{v}_i = \sum_{j \in N_i, j > n} \lambda_{ij} v_j$$

and $A = (a_{ij})_{i,j=1,\ldots,n}$ is the $n \times n$ matrix with elements

$$a_{ij} = \begin{cases} 1 & \text{if } i = j, \\ -\lambda_{ij} & \text{if } j \in N_i, \\ 0 & \text{otherwise.} \end{cases}$$

Methods for efficiently solving these systems are described in Chapter 10 of these course notes.

3.3 Boundary Mapping

The first step in constructing a barycentric mapping is to choose the parameter points for the boundary vertices and the simplest way of doing it is to just project the boundary vertices into the plane that fits the boundary vertices best in a least squares sense. However, for meshes with a complex boundary, this simple procedure may lead to undesirable fold-overs in the boundary polygon and cannot be used. In general, there are two issues to take into account here: (1) choosing the shape of the boundary of the parameter domain and (2) choosing the distribution of the parameter points around the boundary.

Choosing the shape In many applications, it is sufficient (or even desirable) to take a rectangle or a circle as parameter domain, with the advantage that such a convex shape guarantees the bijectivity of the parameterization if positive barycentric coordinates like the mean value coordinates are used to compute the parameter points for the interior vertices. The convexity restriction may, however, generate big distortions near the boundary when the boundary of the triangle mesh ST does not resemble a convex shape. One practical solution to avoid such distortions is to build a "virtual" boundary, i.e., to augment the given mesh with extra triangles around the boundary so as to construct an extended mesh with a "nice" boundary.

4 Segmentations and Constraints

4.1 Segmentation

Planar parameterization is only applicable to surfaces with disk topology. Hence, closed surfaces and surfaces with genus greater than zero have to be cut prior to planar parameterization. As previously noted, greater surface complexity usually increases parameterization distortion, independent of the parameterization technique used. To allow parameterizations with low distortion, the surfaces must be cut to reduce the complexity. Since cuts introduce discontinuities into the parameterization, a delicate balance between the conflicting goals of small distortion and short cuts has to be achieved. It is possible to use constrained parameterization techniques to reduce cross-cut discontinuities. Cutting and chart generation are most commonly used when computing parameterizations for mapping of textures and other signals onto the surface. They are also used for applications such as compression and remeshing. The techniques for cutting surfaces can be roughly divided into two categories: segmentation techniques which partition the surface into multiple charts , and seam generation techniques which introduce cuts into the surface but keep it as a single chart. Multiple charts created by segmentation typically have longer boundaries than those created by seam cutting. However, they can often be more efficiently packed into a compact planar domain.

4.2 Multi-Chart Segmentation

Depending on the application, mesh segmentation techniques use different criteria for creating charts. For parameterization, surfaces are broken into several charts such that the parametric distortion when parameterizing each chart is sufficiently low, while the number of charts remains small and their boundaries are kept as short as possible. Since planes are developable by definition, one possible approach is to segment the surface into nearly planar charts. Planes are a special type of developable surfaces. Thus for parameterization purposes planar segmentation is over-restrictive and usually generates more charts than necessary. Several recent approaches focused on developable segmentation instead [Lévy et al., 2002; Zhou et al., 2004; Julius et al., 2005]. Lévy et al. [2002] proposed to detect high mean-curvature regions on the mesh and then generate charts starting from seeds which are farthest from those regions. This approach tends to capture many developable regions, but can also introduce charts which are far from developable.

4.3 Seams

It is possible to reduce the parameterization distortion without cutting the surface into separate patches by introducing multiple partial cuts or seams inside a single patch. This typically leads to shorter cuts than those created by segmentation.

The Seamster algorithm [Sheffer and Hart, 2002] considers the differential geometry properties of the surface, independent of a particular parameterization technique. It first finds regions of high Gaussian curvature on the surface and then uses a minimal spanning tree of the mesh edges to connect those. Finally it cuts the mesh along the tree edges. Sheffer and Hart [2002] scale the edge length by a visibility metric when computing the minimal spanning tree. This way they are able to trace the cuts through the less visible parts of the surface hiding the potential cross-cut discontinuities in texture or other maps on the surface. Genus reduction Seam cutting methods require an explicit preprocessing stage to convert surfaces with high genus into topological disks. The generation of minimal length cuts that convert a high genus surface into a topological disk is NP-hard.

5 OptCuts: Joint Optimization of Surface Cuts and Parameterization

Low-distortion mapping of three-dimensional surfaces to the plane is a critical problem in geometry processing. The intrinsic distortion introduced by these UV mappings is highly dependent on the choice of surface cuts that form seamlines which break mapping continuity. Parameterization applications typically require UV maps with an application-specific upper bound on distortion to avoid mapping artifacts; at the same time they seek to reduce cut lengths to minimize discontinuity artifacts. We propose OptCuts, an algorithm that jointly optimizes the parameterization and cutting of a three-dimensional mesh. OptCuts starts from an arbitrary initial embedding and a user-requested distortion bound. It requires no parameter setting and automatically seeks to minimize seam lengths subject to satisfying the distortion bound of the mapping computed using these seams. OptCuts alternates between topology and geometry update steps that consistently decrease distortion and seam length, producing a UV map with compact boundaries that strictly satisfies the distortion bound. OptCuts automatically produces high-quality, globally bijective UV maps without user intervention. While OptCuts can thus be a highly effective tool to create new mappings from scratch, we also show how it can be employed to improve pre-existing embeddings. Additionally, when semantic or other priors on seam placement are desired, OptCuts can be extended to respect these user preferences as constraints during optimization of the parameterization. We demonstrate the scalable performance of OptCuts on a wide range of challenging benchmark parameterization examples, as well as in comparisons with state-of-the-art UV methods and commercial tools.

5.1 Problem Statement

Given an input triangle mesh $M = (V, F)$ of a three-dimensional surface with vertices V, and faces F, we seek its UV embedding with connectivity $T^* = (V^T, F^T)$ and a corresponding two-dimensional embedding of vertex coordinates, $U^{\in} R^{2|V^|}$, that locally optimizes the constrained parametrization problem

$$\min_{T,U} E_s(T) \quad \text{s.t.} \quad E_d(T,U) \leq b_d \quad \text{and} \quad (T,U) \in I. \tag{1}$$

Here V^*T is a superset of V with possibly duplicated vertices, and F^*T is the set of original faces indexed into this new set of vertices. We use I to define the set of either locally injective or globally bijective UV maps. Energies E_s and E_d respectively measure seam quality (length) and map distortion while b_d is a user-specified upper bound on the acceptable distortion of the generated map. This optimization is always feasible as in the limit having all triangles separated would allow for zero distortion. In general, distortion measures are smooth albeit nonconvex, while seam length measures are nonsmooth as they increase by discrete amounts when interior mesh edges are cut along or seam edges are merged. Hence, generic optimization techniques cannot be easily applied.

5.2 Dual Objective

As a first step toward solving the problem at hand, we construct the Lagrangian for (1) as:

L(T,U, λ) = $Es(T) + \lambda(Ed(T,U) - bd)$

to form the equivalent saddle-point problem [Bertsekas 2016] defined over primal variables T, U and dual variable λ:

$\min_{T,U} \max_{\lambda \geq 0} L(T, U, \lambda)$

Here $\lambda \in R^+$ is the Lagrange multiplier for our distortion bound. On examination, the Lagrangian L can be seen as a multi-objective balancing between distortion and seam quality as dictated by λ. Here, however, λ effectively applies a local scaling between the seam and distortion terms that is implied by the user-specified distortion bound. As we iterate to solve (1), λ will grow when we threaten to violate our distortion bound prioritizing distortion minimization; similarly, λ will decrease toward 0 when our bound is strictly satisfied to prioritize seam quality.

5.3 Embedding Energy

Concretely we formulate our seam-quality energy as the normalized total seam length

$$E_s = \frac{1}{\sqrt{(\sum_{t \in \mathcal{F}} |A_t|)/\pi}} \sum_{i \in \mathcal{S}} |e_i| \tag{4}$$

where $\mathcal{S}$ is the set of all seam edges on the input surface and $|e_i|$ is the length of edge i in the input mesh. We measure distortion over the mapped domain using the symmetric Dirichlet energy [Smith and Schaefer 2015] normalized by surface area [1],

$$E_d = \frac{1}{\sum_{t \in \mathcal{F}} |A_t|} \sum_{t \in \mathcal{F}} |A_t|(\sigma_{t,1}^2 + \sigma_{t,2}^2 + \sigma_{t,1}^{-2} + \sigma_{t,2}^{-2}), \tag{5}$$

where $\mathcal{F}$ is the set of all triangles, $|A_t|$ is the area of triangle t on the input surface, and $\sigma_{t,i}$ is the i-th singular value of the deformation gradient of triangle t.

5.4 coupled discrete continous descent

To perform our primal update, we seek to minimize the Lagrangian over both continuous changes in vertex positions and discrete changes in topology. To optimize over topology we could potentially perform exhaustive search over the graph of all possible mesh changes. This approach, however, is intractable for any practical mesh size. Instead, we construct a local search algorithm for topological updates that is inspired by the standard descent process of optimizing a smooth energy over vertex positions. For smooth descent methods it is standard to formulate a local approximation of the energy function, use it to estimate the direction for the gradient descent step, and then search along the proposed direction to find the step magnitude that ensures significant decrease in energy. We extend this process to include search over discrete variations in topology. In analogy to seeking a continuous search direction, at the start of each new primal solve we will build many localized energy approximations to search for a likely candidate mesh operation to repeatedly propagate topology change, i.e., cutting or merging, over our UV mesh. Each inner iterate of the primal solve will successively apply this operation in combination with standard smooth descent to explore discrete-continuous descent until no further progress is made

6 Conclusion

The paper presents an algorithm for generating UV maps for 3D triangle meshes. UV maps are 2D representations of a 3D surface that can be used to apply textures and other visual effects to the surface. The algorithm seeks to find a UV map that minimizes distortion while also maintaining high seam quality. Seams are edges where the texture coordinates change abruptly, and they can have a significant impact on the visual quality of the surface.

The algorithm works by iteratively optimizing two objectives: seam quality and map distortion. The seam quality objective is measured by the length of the seams in the final UV map. The map distortion objective is measured by the amount of distortion in the UV map relative to the original 3D surface. The algorithm uses a Lagrangian formulation to balance these two objectives and to ensure that the distortion of the UV map does not exceed a user-specified bound.

To generate the UV map, the algorithm first creates a superset of the input mesh vertices by duplicating some vertices. It then iteratively optimizes the seam quality and map distortion objectives by adjusting the positions of the vertices in the superset. The algorithm uses a novel parameterization scheme that ensures that the UV map is locally injective, which means that adjacent triangles in the 3D surface are not mapped to overlapping regions in the UV map.

The paper presents experimental results that demonstrate the effectiveness of the algorithm compared to previous methods. The algorithm is able to generate high-quality UV maps with low distortion and high seam quality for a wide range of 3D surface geometries. The algorithm is also able to generate globally bijective UV maps, which means that the entire surface is mapped onto a non-overlapping region of the UV map. Overall, the algorithm is a significant improvement over previous methods for generating UV maps for 3D triangle meshes.