Only vertices are stored in the data structure, but information about edges and faces of the surface is also present in the lists of neighbors. For instance, vertex 7 in Figure 3.2 has the neighbor list {`6, 0, 1, 2, 8, 11, 10, 9`}. Taking every second number in this list – 6, 1, 8, 10 – reveals that edges to nodes 6, 1, 8 and 10 emanate from node 7. Overlapping triples of neighbors – {6, 0, 1}, {1, 2, 8}, {8, 11, 10} and {10, 9, 6} (by wrapping around to the first neighbor) – give the four faces {7, 6, 0, 1}, {7, 1, 2, 8}, {7, 8, 11, 10} and {7, 10, 9, 6}, all written counterclockwise. In this way, every face of the surface is mentioned four times, once by each corner. Our algorithm, however, requires an additional table that lists every face exactly once. We generate it by visiting all vertices, putting a face in the table only when it is defined from the corner with the smallest id, i.e. when it is mentioned for the first time.

As the object lies completely within our data volume, we always find a closed surface. All vertices in the list are inner vertices of the surface, and therefore they all have an even number of neighbors. The data structure generalizes in a very natural way to the case of open surfaces, or surface patches, where vertices on the border of the surface have an odd number of neighbors. This possibility is not exploited here, but only later in the parametrization of surface patches (cf. section 3.5).

The correspondence between the surface net and a graph becomes clear when a vertex is interpreted as a node in the graph, an edge as an arc, and a face as a mesh (four-cycle)[40]. For a simply connected object we get a planar graph with the following topological properties: Four edges bound each face, and each edge bounds two faces and is bounded by two vertices. Depending on the local connectivity, each vertex bounds three to six edges. There are exactly two more vertices than faces; this follows from Euler's relation.

A surface with its two degrees of freedom is characterized by a polygonal description based on vertex coordinates with three spatial coordinates. Seeking for an appropriate parametrization, however, requires a description based on two parameters.

3.4 Parametrization of closed surfaces

A key step in the description of the form of a surface is the mapping of the surface to the parameter space, in our case the sphere. A one-to-one mapping must be constructed, i.e. any point on the surface has to map to exactly one point on the sphere, and vice versa. The location on the sphere corresponding to a surface point defines the *parameters* of the point. It can be represented in a computer as two polar or three Cartesian coordinates. Mapping a surface to the sphere assigns parameters to every surface point; therefore I also call it parametrization. The mapping must be continuous, i.e. neighboring points in one space must map to neighbors in the other space. It is possible and desirable to construct a mapping that preserves areas. The use of the cuberille notion gives special importance to case of square facets, which map to spherical quadrilaterals. Figure 3.5 symbolically illustrates this mapping of a selected facet from the object surface to a portion of space U. We recall from subsection 2.1.3, and in particular from example 6, that the parameter space $U = \Omega_3$ is a subset of of $\mathbb{R}^3$, but that it could be related to the rectangle $[0, \pi] \times [0, 2\pi) \subset \mathbb{R}^2$ through the bijection (7.4(appendix)). It is not possible in the general case to map every surface facet to a spherical square. Distortions cannot be avoided, but they should be minimal.

It emerges that the parametrization, i.e., the embedding of the object surface graph into the surface of the unit sphere, is a constrained optimization problem. The following paragraphs define the meaning of *variables*, *objective* (goal function), *constraints* and *starting values* in this context.

Variables The coordinates of all vertices can vary in the optimization. Using two (e.g. spherical) coordinates per vertex would be the most economic representation with respect to storage space, but this would make the equal treatment of all spatial directions difficult and pose the problem of discontinuity and singularities in the parameter space. Therefore we prefer Cartesian coordinates (u_0, u_1, u_2) for representing a location on the sphere, introducing one virtual degree of freedom per vertex. The number of variables is three times the number of vertices.

Constraints Two kinds of equalities and one kind of inequalities constrain the values that the variables can take.

1. The Euclidean norm of the coordinates of any vertex must be 1. This constraint compensates for the virtual degree of freedom and forces every vertex to lie on the unit sphere in parameter space.

2. We ask for *area preservation*, which in our context means that any object surface region must map to a region of proportional area on the sphere. To satisfy this requirement, we include one constraint

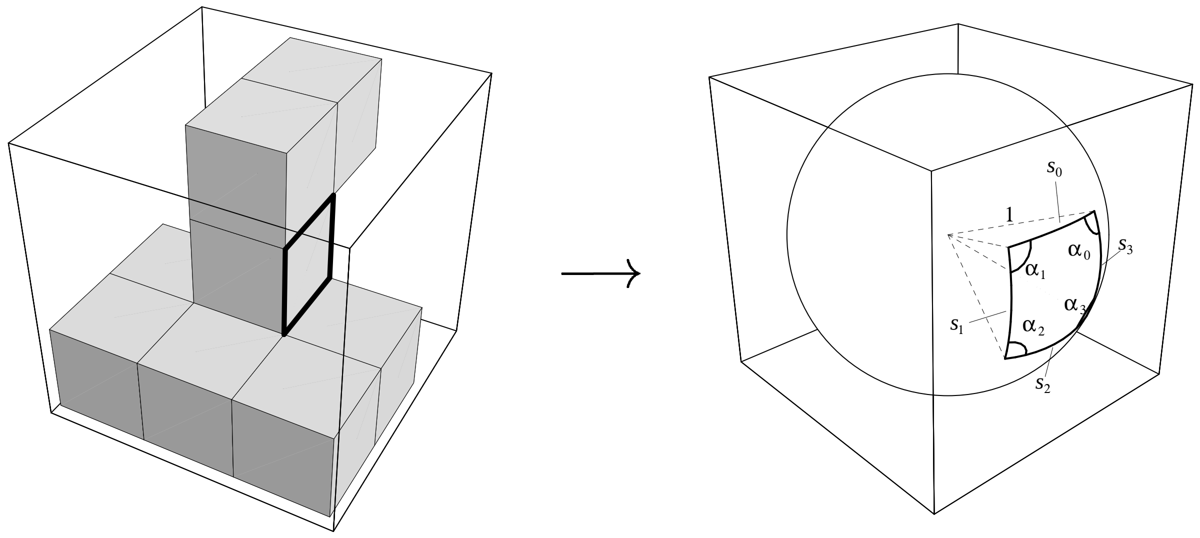

Figure 3.5: Every single face on the object's surface is mapped to a spherical quadrilateral. The sides of a spherical polygon are geodesic arcs on the sphere surface. As the sphere has unity radius, the length of a side s_i is equal to the corresponding center angle (in radian). The quadrilateral in this illustration is special in that its four sides $s_0 \cdots s_3$ are equal and its four angles $\alpha_0 \cdots \alpha_3$ are equal: it is the spherical analogue of a square.

   for each elementary facet (see Figure 3.5): the area of the corresponding spherical quadrilateral must be 4π divided by the total number faces.

3. No angle α_k of any quadrilateral on the sphere must become negative or exceed π.

Objective Function The *goal* is to minimize the distortion of the surface net in the mapping. It is conceptually similar to angle preservation, and it must tend to make the shape of all the mapped faces as similar to their original square form as possible. To fulfill this goal perfectly, a facet should map to a "spherical square" (see Figure 3.5). This can never be reached exactly for all faces except when the object has a very special form, e.g., consists of one single voxel. There are several ways to trade off between the distortions made at different vertices. We observe that the ideal shape of any face, a spherical square, minimizes the circumference $\sum_{i=0}^{3} s_i$ of any spherical quadrilateral with a given area. At the same time it maximizes $\sum_{i=0}^{3} \cos s_i$. These two measures are similar, but not equivalent if summed over the whole net, as they trade off among distortions differently. The second measure punishes too long sides more and honors too short sides less than the first one, which is a desirable effect. It is also simpler to calculate; the cosine of a side - and of the respective central angle - is the dotproduct of the vectors from the sphere center to the endpoints of the side.

Starting Values The variables in our optimization are the positions on the unit sphere to which the vertices are mapped. Therefore, *initial values* in this context means an first rough mapping of the object's surface to the sphere. It is important for the optimization algorithm that the sphere be completely covered with faces and none of them overlap, even in the beginning.

Chapter 4 describes the construction of an initial parametrization satisfying the last requirement, and it elaborates on the technical details of the optimization procedure. Figure 3.6 anticipates the parametrization result for the small object from Figure 3.5, called "duck" in the sequel.

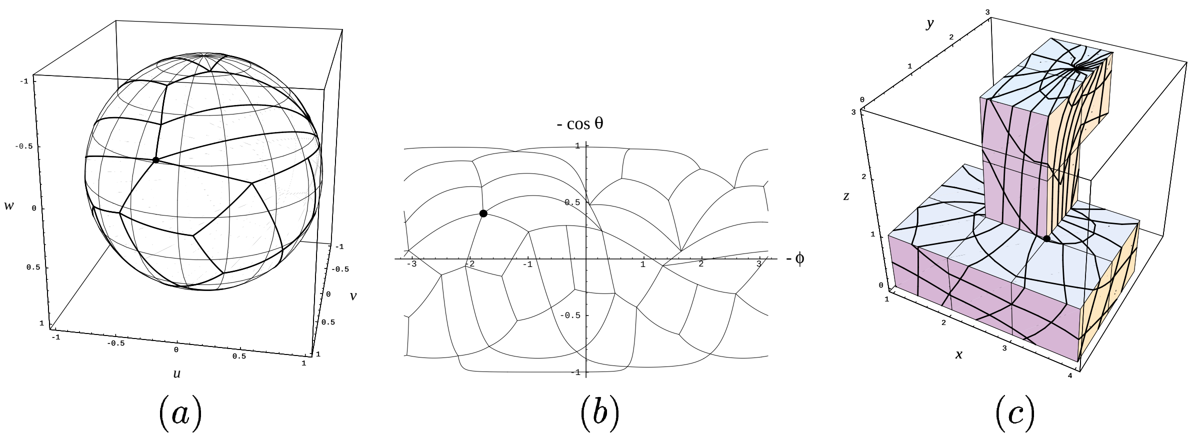

Figure 3.6: The parametrization achieved by the optimization is visualized three different ways. *(a)* The surface net is plotted on the spherical parameter space. Thick lines depict the edges of the original square faces. The equidistance for both θ and ϕ is $\frac{\pi}{8}$. *(b)* Cartesian interpretation of $(\phi, \cos\theta)$ gives an equal-area cylinder projection. The horizontal lines at ± 1 are the poles. *(c)* Conversely, the polar coordinate grid is drawn over the object.
For comparison, one vertex is marked with a black dot in all diagrams.