Almost isometric mesh parameterization through abstract domains

Nico Pietroni, Marco Tarini, and Paolo Cignoni

Abstract—In this paper we propose a robust, automatic technique to build a global hi-quality parameterization of a two-manifold triangular mesh. An adaptively chosen 2D domain of the parameterization is built as part of the process. The produced parameterization exhibits very low isometric distortion, because it is globally optimized to preserve both areas and angles. The domain is a collection of equilateral triangular 2D regions enriched with explicit adjacency relationships (it is abstract in the sense that no 3D embedding is necessary). It is tailored to minimize isometric distortion, resulting in excellent parameterization qualities, even when meshes with complex shape and topology are mapped into domains composed of a small number of large continuous regions. Moreover, this domain is in turn remapped into a collection of 2D square regions, unlocking many advantages found in quad-based domains (e.g. ease of packing). The technique is tested on a variety of cases, including challenging ones, and compares very favorably with known approaches. An open source implementation is made available.

Index Terms—Modeling,Surface Parameterization.

1 INTRODUCTION: OBJECTIVES

A Parametrization is commonly defined as a bijective mapping between a two-dimensional domain D and a two-manifold surface M embedded in $\mathbb{R}^3$. "Good" parameterizations are a prerequisite in modelling and rendering techniques, like remeshing, morphing, texture mapping and others. However, finding a good parametrization for a given mesh is in general a challenging task. This paper introduces a new technique to address this problem in an automatic way.

We argue that a good parameterization is characterized by:

Low distortion: the ideal parametrization is isometric, i.e. it fully preserves areas and angles. Since this is not possible in the general case, low-distortion parametrization are sought, distortion being a measure, variously defined, of how far a mapping is from being isometric. Parameterization showing low distortions are clearly more usable (e.g. they produce more regular resampling in remeshing, even texel distribution in texture mapping, etc). To achieve isometry, area-preservation is as important as conformality: our technique seeks to fulfill both objectives.

- *N. Pietroni, M. Tarini and P. Cignoni are with the Visual Computing Lab at the ISTI - Instituto of Science e Tecnologie dell'Informazione, CNR - National Research Council, Pisa, Italy E-mail: pietroni, tarini, cignoni@isti.cnr.it*
- *M. Tarini is also with DICOM, Department of Informatics and Communication, Universit dell'Insubria, Varese, Italy.*

Space left blank for data about submission timings etc

Low domain complexity: another factor that determines the usefulness of the parametrization is, informally speaking, how complex its domain is. A domain can be considered the more "simple" the more fully the following set of interconnected properties applies:

- canonical tangent directions in the domain, remapped over the mesh, always varies smoothly;
- pairs of close positions on the mesh are mapped into pairs of close positions on the domain;
- points on the domain have a well defined, easy to find, large neighborhoods around them;
- the domain is closed to interpolation, i.e., it is possible to interpolate between positions on it;
- the domain can be embedded in a 2D region (e.g. a square) without leaving unused space.

Most applications of parameterization benefit from these properties. For example, their fulfillment helps to perform geometry processing on domain space, or to define texture images that can be shared by different meshing of the same object, etc. Domain "simplicity" could also be defined as the ability to give the illusion that the domain is a seamless, continuous, regularly shaped 2D region, so that it can be used as such.

Examples of very "simple" domains are the one used in single-disk parameterizations, octahedral maps [1], poly-cube maps [2], and others. Conversely, an atlas domain is less "simple", and even less so the more charts it has and the more irregular their boundaries are. As a hypothetical, extreme example, consider a parameterization which maps each triangle of M into a separate 2D triangle in D of the same size: it would be fully isometric but with a "complex" domain to the point of being useless.

In general distortion and domain complexity are often two contrasting objectives, and good balance is needed.

Our method is designed to achieve excellent results both in producing "simple" domains and in achieving low distortions of the mapping.

2 Related work

A large number of papers on surface parameterization have been recently published (refer to [3] for a survey). We only cover the ones most relevant to our approach.

Angle-preserving (conformal) mappings are useful in some context and studied on their own (see [26] and [27] for a recent solid theory in the discrete setting). As mentioned, for our purposes, area-preservation (as in [28] or [5]) is just as important.

A category of approaches addresses parameterization of single disks (see [4] for a survey). The parameterization here is a continuous seamless function, mapping the boundary of the domain over the boundary of the mesh. Single disk parameterizations are ideal in terms of domain simplicity, but they are directly applicable only if the 3D mesh is topologically a disk; even then, they lead to high distortions unless the mesh has almost zero Gaussian curvature everywhere. Still, those approaches are always relevant because they are used in many others, including ours, in the form of a subphase of a global process (Sec. 5.2). In our framework, any method that optimizes over a flat domain can be adopted. Linear methods would clearly be faster and more robust, but better parameterizations which target *both* area and angle preservation are achieved by nonlinear ones. We adopt the system proposed in [5], which extends conformal energy of [6] to explicitly take into account area-preservation, but others, like the ones described in [7] or [8], could be used just as well.

A class of approaches, countering shortcomings of direct single-disk parameterization, is based on the introduction of seams ("cuts") on the mesh. Cuts are used for two purposes: lower distortion by moving curvature on the boundary, and reduce the genus of the mesh.

When seams subdivide the mesh into detached regions, each to be parameterized separately, the technique falls into the "atlas" category, also called "multi-chart" (e.g. [9], [10], [11]). Multi-charts, often manually crafted, are more widely used in practice than studied in theory, especially for texture mapping.

In other approaches, e.g. [12], [13], [14], seams do not split the mesh into separate parts. A recent trend [13], [14], [15] uses cone singularities [13] to ameliorate the effects of these cuts: points on the mesh (the cone singularities) are selected to "soak" all the Gaussian curvature, leaving all the rest of the mesh at zero-curvature. A global system is run to minimize distortion before cuts take place, by solving for some metric quantities e.g. of each edge. The actual cutting is postponed to a separate phase, thus ensuring that the two sides of the cut are coherent and therefore easily matched (e.g. zippered in a remeshing approach) afterward.

Despite these differences and progresses, all cut-based approaches share drawbacks which are inherent to the presence of seams, going against our goal of domain simplicity. This is an intrinsic limitation of the produced domain, regardless of the way it is computed.

Another class of solutions takes a different view on the problem (among others, [16], [2], [17], [1]): the domain of the parameterization is a surface embedded in $\mathbb{R}^3$, sharing the same genus, and possibly the same general shape, of the original mesh. This surface is then, in turn, mapped into the final flat domain. The composition of the two mapping can be considered just another multi-chart approach; however, the second mapping is simple, being controlled by a few parameters, or can even be kept implicit. In other words, the surface in 3D serves as a **spatial metaphor** to describe how 2D charts are defined and related with respect to each other. The metaphor is exploited to improve on domain "simplicity", and therefore the usability of the parameterization.

In [1], the domain surface is a regular octahedron (passing through a sphere, as in [18]), and the second mapping is fully implicit; this can only be used for genus-0 objects.

In [2], which is designed for texture mapping, the domain surface is defined by an ad-hoc polycube, and the second mapping is kept simple so that it can be performed only at the very last moment, during the texture look-up operation in the fragment shader of a texture-mapped rendering. In this way the system provides an illusion of a texture domain as simple and seamless as the one used in single-disks parameterization; however, no technique is proposed to automatically build such a parameterization for a given mesh.

In [17], which extends [16] by adding a global optimization phase, a low resolution mesh is used as parameterization domain (here called base mesh). In this sense, it is similar to the one presented here. We improve over it in a number of conceptual and practical ways, ultimately achieving much lower distortions while improving on domain simplicity.

Base meshes are also common in approaches which seek to parameterize several meshes in a mutually compatible way [19], [20], [21], e.g. for morphing application. Here all models must be mapped over a shared base mesh, in a semantically meaningful way. This can only be achieved by user intervention: user-identified points over the original meshes are used to define the base mesh. Our approach can in principle be extended similarly for the same purpose, letting an user identify points over the mesh, but we strive to achieve a fully automatic method (for a single mesh). In [19], as an application of their work, authors propose the key idea that the base mesh can be kept abstract, without any embedding in a 3D space. We take that idea and expand it into a fully automatic novel approach to parameterize a single mesh.

While the 3D parametrization domain is often based on triangles [19], [20], [17], [16], [1], there are clear advantages in adopting a quad-based domain, like in [2], [22], [23], [24]. Texture mapping applications lend more naturally to quads, texture being rectangular in nature.

Packing of 2D quad patches into a single 2D domain is also easier and bypasses the problem of non axis-aligned patch boundaries. A quad based parametrization also supports more easily GPU-based geometry processing in domain space. However quad-based domains are harder to construct and to deal with. As a side contribution, in Sec. 6.0.1 we offer a solution that, while still using triangle-based 3D domains, uses 2D quad patches inheriting many advantages of quad based domains.

Contributions: Our approach belongs to the class of solutions which uses a 3D surface as intermediate parametrization domain. However, the used "metaphor" is not of a surface embedded in $\mathbb{R}^3$ but rather an *abstract* surface, i.e. one where vertex coordinates are absent; the abstract surface is defined only by its connectivity and the assumption that each of its face represents a unit-sized equilateral triangle. The abstract surface is a metric surface with cone singularities [25]. It is easy to see that resorting to equally sized, equally sided triangles as domain improves domain simplicity. Introduction of special "interpolation domains" (Sec. 4.1) further improves the usability of that structure.

Domain simplicity is a structural characteristic of the produced output, but it is also exploited during its construction, in particular in a robust global optimization phase of the mapping (Sec. 5.5). We found that the use of flexible local operations (Sec. 5.3 and Sec. 5.4) leads to the adaptive production of a domain which fits the input mesh better than in previous approaches.

It is now recognized that to achieve lowest distortion mappings between the two surfaces it is crucial that both share similar values of discrete scale-dependent Gaussian curvature. This objective is not explicitly sought in our technique, but it is nevertheless reached to a good extent (Sec. 7.2). We believe that this is one of the reasons explaining why our results perform better in terms of isometry compared to older similar approaches like [17].

Since we strive to optimize the overall isometry, area *and* angle preservations are both given the same importance and explicitly sought in our approach. This basic choice motivates many of our design decisions.

3 Overview

The technique proposed here takes as input a 2-manifold mesh M, and returns a parametrization $\theta : D \rightarrow M$ mapping an appropriate 2D parametrization domain D over M.

M is defined as a collections of vertices v_i^M in $\mathbb{R}^3$ plus the connectivity (triangles t_i^M and edges e_i^M). For now, we assume M to be triangular, and closed, but each of these assumptions could be easily worked around to deal with more general cases. We do not make any assumption on the genus of M, nor on the quality of its meshing: the proposed technique works with arbitrary topology, and is robust with respect to uneven triangulations (e.g. meshes with differently sized or thin triangles, Fig. 10).

The output is a parametrization, i.e. a 1-to-1 mapping θ from a fitting 2D domain space D to M. As common, the system defines θ by tabling its inverse $\phi : M \rightarrow D$, i.e. by explicitly assigning to each vertex v_i^M of the original mesh M a position $\phi(v_i^M) \in D$.

In our approach we focus on identifying a proper *parameterization domain* D which is fitting for the given input mesh. Similarly to [17], [16], this domain is obtained starting from the original connectivity of M and performing a sequence of local operations over it. The mapping ϕ is updated during each operation.

Before describing in details the steps of our approach in Sec. 5.1, we characterize the structural properties of the domain we use.

4 Structure of the parameterization domain

Our parameterization domain D, a flat metric surface with cone singularities, consists in a collection of unit-sided equilateral 2D triangles $D_0..D_{N-1}$, termed *sub-domains*. A position in D is expressed as a triple (i, α, β), where the $i \in [0..N-1]$ is an index of one sub-domain, and (α,β) are the first two barycentric coordinates identifying a position inside the triangular sub-domain D_i (the third one is implicit, by difference).

The integer number N, i.e. the number of sub-domains composing D, typically ranges between a minimum of 4 and a maximum of a few hundreds, according to the topological and shape complexity of M.

Domain D is enriched with an explicit adjacency information between its sub-domains D_i, i.e. a consistent, reciprocal neighborhood relationship between them. This connectivity define D as 2-manifold, closed, and well oriented, meaning that each side of each sub-domain is shared by, i.e. considered coincident to, one side of another sub-domain. In this paper we refer to edges e_i^D of D (sides shared by two of its sub-domains) or vertices v_i^D of D (corners shared by k of its sub-domain). However, this does not imply that the domain D can be seen as the a triangle mesh embedded in a three dimensional Euclidean space, because 3D positions are absent. Its sub-domains D_i rather compose an *abstract mesh*, i.e. one defined by its connectivity, and the assumption that its faces are equilateral unit-sided triangles. It can well be the case that no 3D-embedded mesh exists with such characteristics.

One key ingredient of our approach is the use of special *interpolation domains* which let us to interpolate positions over the domain D . Consequently, by assigning explicitly a position $\phi(v^M)$ over D to each vertex v^M of M, we are also defining mapping between any point in M into a position over D, as well as a mapping from any triangle t^M of M to a region $\phi(t^M)$ in D (region which can span over multiple adjacent sub-domains).

4.1 Interpolations domains

In order to deal with entities on the mesh (e.g. triangles, segments, or entire mesh regions) which are mapped by

ϕ over multiple sub-domains of D, we use a temporary ad-hoc abstraction we call interpolations-domain.

An *interpolation domain* is defined as a continuous and convex 2D region E with an associated function g_E which maps it into a subset of D (the *image* of that domain); the image spans over a small number of adjacent sub-domains $\{D_{j0}..D_{jk}\}$. Function g_E consists of a piece-wise, usually rigid transformation, defined for each h (the transforamtion is not rigid in the case of irregular vertices, see below). Importantly, all functions g_E are invertible, and fast to compute both ways.

Notation: we denote the two coordinates inside an interpolation domain E by (s,t), to distinguish them from (x,y,z), used for the object coordinates of mesh M, and from (i,α,β) used for a point in parameter space D.

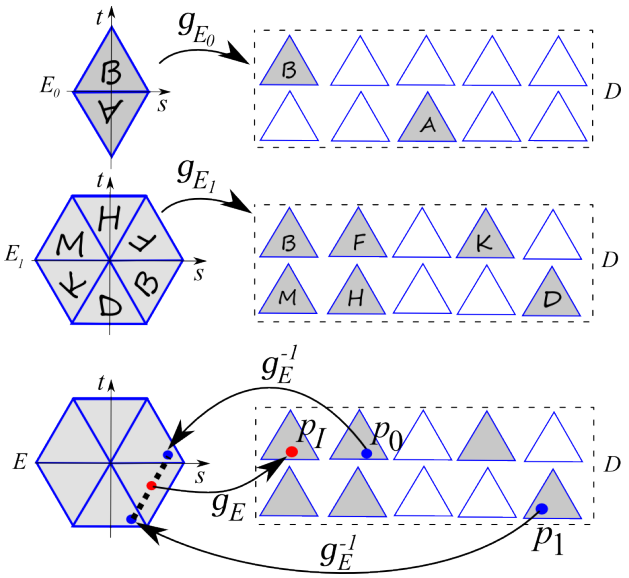


Fig. 1. Interpolation domains: star- and diamond-domains (degree 6), and their associated functions g_E. Bottom: an interpolation domain is used to interpolate across two points in D (see equation 1).

One basic use of interpolation domains is to let us linearly interpolate across two points in D, even if they belong to different sub-domains (hence their name). Given two positions $p_0 = (i_0, \alpha_0, \beta_0)$ and $p_1 = (i_1, \alpha_1, \beta_1)$ in D, we find the interpolated position p_I though a properly selected *interpolation domain* E:

$$p_I = g_E(\ I(\ g_E^{-1}(p_0)\ ,\ g_E^{-1}(p_1)\ )\ )\ , \qquad (1)$$

where I is the standard linear interpolation function in E (Fig. 1, bottom).

Indices i_0 and i_1 determine which interpolation domain must be used (we use different kinds, see later).

If D_{i_0} and D_{i_1} share one edge e^D then E is shaped as the diamond composed by two unit-sized equilateral triangles matched side by side. We call this type of interpolation domain *diamond domain* (Fig. 1, top). Function g_E rigidly maps the two triangles into D_{i_0} and D_{i_1} respectively, rotating them to preserve their adjacency.

If D_{i_0} and D_{i_1} share a vertex v^D but no edges, we use a type of interpolation domain which we term *star-domain* (Fig. 1, middle): in this case E is a zero-centered, regular k-agon, k being the degree of v^D as defined by the connectivity of D (the number of subdomains sharing v^D); E is rescaled so that its total area matches the area of k equilateral unit-sided triangles. Function g_E linearly maps each slice of E into one of the sub-domains of D sharing vertex v^D, rotating them so to preserve adjacencies.

The associated function g_E is easily computed by testing the argument against k half-lines passing trough the center of E. The inverse is also trivial.

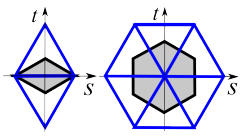
Note that when $k \neq 6$, g_E it is not a piecewise rigid transformation, since it includes a non-uniform rescaling (because the sides of the k-agon cannot be unitary). However, $det(\nabla g_{Es}^{-1} | \nabla g_{Et}^{-1})$, that is, the point-wise area distortion introduced by g_E, is constantly 1 over all E. We have chosen an area-preserving g_E, because we favor area-preservation over angle preservation, but conformal g_E could be adopted too (e.g. exponential maps). Another alternative is the use of functions in between of these two extremes (purely conformal g^E, and purely area-preserving g^E), defining them (and their inverse) by linear interpolation among the two.

A third, simplest kind of parametric domain is the *face-domain*. A face-domain E is shaped as an equilateral triangle and is mapped by g_E over a single sub-domain D_i.

An interpolation domain can also be used to interpolate inside triangles defined over D. Given three points (i_0, α_0, β_0), (i_1, α_1, β_1) and (i_2, α_2, β_2), the corresponding triangle can be spanned using a proper interpolation domain E: specifically, E is the face-domain for D_{i_0} if $i_0 = i_1 = i_2$; else, a diamond-domain if $\{i_0, i_1, i_2\} \subset \{i_a, i_b\}$ and D_{i_a} and D_{i_b} are adjacent. Else, the star-domain relative to vertex v^D if D_{i_0}, D_{i_1} and D_{i_2} share v^D.

We are interested only in triangles of points in D where one of the above conditions apply,

4.2 Partitioning the domain space D



We define three alternative sets of interpolation domains so that for each set the images of the associated g form a disjoint partition covering the entire domain space D. The images of triangle-domains already form a first partition of D. Other two are composed of interpolation domains of a new kind: *half-diamond domains* and the *half-star domains*, obtained restricting respectively the diamond and star domains as defined above.

A half-diamond domain is a sub-region of a diamond domain E defined as the diamond that has the longest diagonal corresponding to the shortest diagonal of E, and the shortest diagonal as the line connecting the two

barycenters of the two equilateral triangles forming E (grayed area in the inset image above). Analogously, an half-star domain is a subpart of a star domain E, and it is defined as the k-agon which connects all the barycenters of the slices of E. Half-diamond and half-star domains are are associated to the same g_E function of the corresponding diamond and star domain.

Three sets of interpolation domains now partition the entire parametric domain D with their images:

1) a set of face domains, one for each sub-domain D_i of D;
2) a set of half-diamond domains, one for each edge of D;
3) a set of half-star domains, one for each vertex of D.

Any point $p \in D$ belongs to the image of one face domains, one half-diamond domain, and one half-star domain.

If one chooses any of these three partitions, then it is immediate, given a position $p \in D$, to find the (unique) interpolation domain E in the chosen set, and also a point $(s,t) \in E$, such that $g_E(s,t) = p$. Specifically, a point $p = (i, \alpha, \beta)$ is in the image of: the face-domain associated to sub-domain D_i, the half-diamond domain associated to the edge opposite to the vertex of D_i corresponding to its smaller barycentric coordinate (α, β or $\gamma = 1 - \alpha - \beta$), and the half-star domain associated to the vertex of D_i corresponding to the largest barycentric coordinate. Point (s,t) is found as $g_E^{-1}(p)$.

A fundamental property of these sets of interpolation domains is that every point in p in D is mapped by g_E in a position inside an interpolation domain which is far from the border of that interpolation domain, in at least one of the three listed sets (a property which does not hold if any set is removed from the list). In other words, every point p falls away from the border of either the face domain, the half-diamond domain, or the half-star domain (see Fig. 2). We refer to this property by saying that the set of partitions is *free from permanent boundaries*.

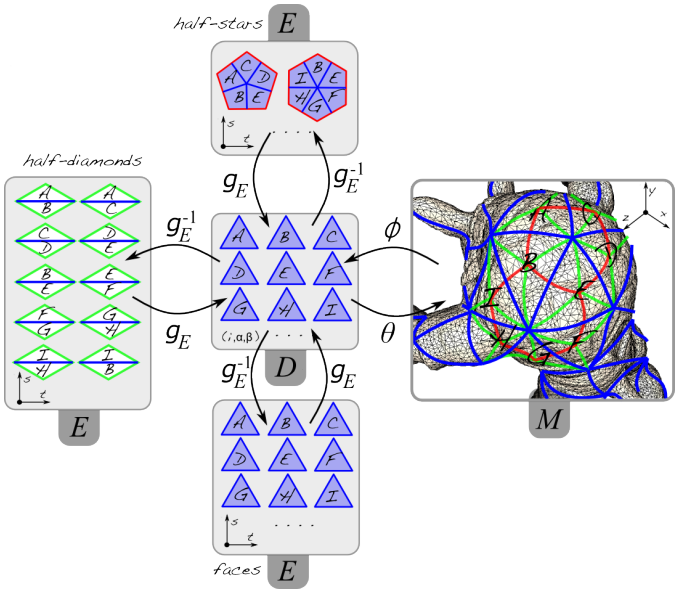


Fig. 2. Relationship between two-manifold mesh M, parametrization domain D (a collection of equilateral triangles with explicitly defined adjacencies), and the three auxiliary sets of interpolation domains that partition D (and, consequently, M).

5 Building Domain and Parameterization

Given a mesh M, we want to find a parametrization domain D and a mapping $\phi : M \rightarrow D$. Following to similar approaches (e.g. [17]), we build both D and ϕ incrementally: we start with a trivial mapping of M to itself and apply a sequence of local modifications and optimizations affecting both, interleaving the process with a global optimization phase.

Even if this sequence of local operations cannot be guaranteed to converge in a global optimum, the results are consistently good (a similar issue is present in most mesh simplification methods).

The general algorithm is presented in Sec. 5.1. Further subsections detail the various sub-phases (edge collapses, 5.3, edge flips, 5.4, local optimization of ϕ, 5.2, its global optimization, 5.5, including a discussion on its convergence in 5.6, then measurements of paths and regions, 5.7, and finally determination of the ideal number of domains, 5.8).

Goal: Let us state a desiderata on D and ϕ that is necessary to achieve isometry, and will be explicitly sought.

ϕ and D implicitly partition the area of mesh M is into *regions*, $\theta(D_0)..\theta(D_N)$ (recall $\theta = \phi^{-1}$): each region is the portion of mesh mapped by ϕ into a sub-domain. Two adjacent regions $\theta(D_i), \theta(D_j)$ on M are separated by a *path* embedded in 3D, denoted by $\theta(e_k^D)$, with e_k^D being the edge in D shared by D_i and D_j (Fig. 3). In other words, regions and paths are 2D and 1D entities on the surface M which are mapped by ϕ into one sub-domain of D, and one edge of D respectively. Observe that a path does not in general pass through vertices or edges of M, as a region can include only subparts of triangles of M; also, a path is not, in general, a geodesic on the M, nor it would be beneficial that it was.

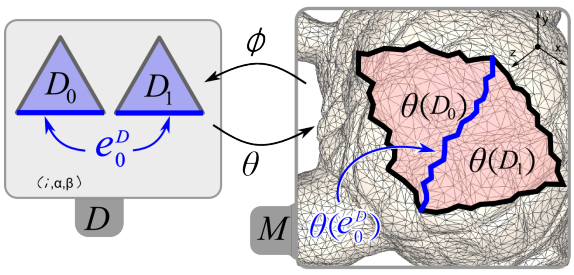


Fig. 3. Two regions $\theta(D_0)$ and $\theta(D_1)$ and a path $\theta(e_0^D)$ on mesh M defined by the domain D and the mapping ϕ. The path is lies on the surface of the mesh but does not in general pass through its edges.

Our target is to have (as much as possible) each region $\theta(D_j)$ of the same area, and each path $\theta(e_k^D)$ of the same length. It is easy to see that when these conditions are fulfilled the domain D allows for maximally isometric mapping ϕ (since subdomains in D are defined as equilateral unit sized triangles).

5.1 General algorithm

1) Start up: Given M, we trivially build an initial domain D, making a sub-domain D_j for each triangle in the mesh M. An initial function ϕ is likewise defined trivially as one that maps each vertex v^M of M in an corner of a sub-domain corresponding to any of the faces of M sharing v^M. Note that this initial couple (D, ϕ) does not constitute an isometric parameterization because each triangle of M, regardless of its shape and size, is mapped over an equally sized equilateral triangle.

2) Decimation: By means of edge collapses (Sec. 5.3), the number of sub-domains will be progressively reduced, until the desired N is reached (N is found by the system as the best one in an interval specified by the user, see Sec. 5.8).

At every step, each edge e_i^D of D is a potential candidate for an edge-collapse. Just like in mesh simplification approaches [29], a cost is associated to each of them; at each step the least-costly is determined and executed, followed by an update of the costs of the operations associated to any affected edge. A heap structure is used to speed up identifications of the minimal cost operation.

The cost we associate to the collapse of edge e_k, shared by sub-domain D_i and D_j, is inversely proportional to

$$Area(\theta(D_i)) + Area(\theta(D_j)) + Length(\theta(e_k^D))^2 \quad (2)$$

I.e. we systematically collapse the edges of D corresponding to the shortest paths, and remove the sub-domains of D corresponding to the smallest regions, so survivors tend to be equally sized regions separated by equally-long paths.

2A) Healing of overlong edges: As a result of the collapse of an edge e_i, the path associated to a few edges $e_{j0}..e_{jK}$ incident to the collapsed ones usually gain some length. Sometimes their length will exceed too much the current average one, against our goal (having all paths of the same length). If this happens away from the end of the process, when the average path length is still very small with respect to the final one, then the overlong paths will just survive the process until any other paths will match their length. However, toward the end of the process, the overlong paths must be dealt with directly, by means of an edge flip (Sec. 5.4).

We define a gain for a flip of edge e_i as

$$Length(\theta(e_i^D)) - Length(\theta(\tilde{e}_i^D)) \quad (3)$$

where $\tilde{e}_i^D$ is the edge resulting from the potential flip of e_i^D. The gain is positive if the length of the path in M corresponding to E_i is shortened by the flip.

We adopt the following heuristic: when the current domain D is composed by exactly $\lfloor 1.5^h N \rfloor$ with $h \in \{0, 1, 2\}$ (that is, twice toward the end of the process, than once more at the end) we pause the decimation and rate the potential flip of each edge (this choice is not crucial, as long a this operation is performed a few times toward the end of the process). The flips with a positive gain are performed, starting from the one with the biggest gain. In practical cases, almost all potential edge-flips are found to have a negative gain and are not executed. The few ones that are applied, however, lower considerably the conformal distortion of the resulting mapping. Observe that the edge-flipping operation by itself affects angle-distortion only and does not have any effect on area-distortion, as the sum of the areas of the two affected regions of M does not change.

2B) Periodical global optimization: Both the measures for costs (or gains) of edge-collapses and edge-flips depend on the areas and lengths of regions (on M), and therefore on the current mapping ϕ. Therefore, in order to better drive the choice of the collapse/flip to perform, it helps globally optimizing ϕ (Sec. 5.5) at regular intervals. To save time, this global optimization can be stopped before full convergence is reached. We perform a rough global optimization step of this kind every time the number of domains of D is reduced by a constant factor k, e.g. 10 (performances are not significantly affected by the choice of k).

3) Final global optimization: After the domain has been reduced to the appropriate number of sub-domains by means of edge-collapses and edge-flips, a final global optimization phase (Sec. 5.5) is executed until convergence.

5.2 Local optimization

A local optimization of ϕ is necessary after one local change to ϕ and/or its domain D, like an edge collapse (Sec. 5.3) or edge flip (Sec. 5.4). Also, global optimization is achieved by a proper succession of local optimizations (Sec. 5.5) .

Local optimization is performed over any chosen interpolation domain E. Given E, function ϕ will be optimized for all vertices v_j^M of M such that $\phi(v_j^M)$ lie in the image of E. In order to do that, we extract the sub-mesh M' composed by all such vertices and the faces connecting them. Sub-mesh M' is open (there are border edges).

We define a parametrization of M' over E, by assigning to each vertex v_j^M of M' the parametric position $(s_j, t_j) = g_E^{-1}(\phi(v_j))$, $(s_j, t_j) \in E$. At this point, we are free to minimize the isometric distortion over E, using any single-patch parametrization optimization technique.

We use the technique described in [5], which minimizes a combined measure of area and angle distortions, thus reaching a good isometry of the parametrization:

$$d_a^\gamma \cdot d_c \quad (4)$$

where d_c measures conformal distortion using MIPS [6], d_a measures area distortion (both recentered in 1), and γ is a parameter used to specify the relative importance of the two factors. In all our experiments we adopted $\gamma = 3$, which gives a good balance.

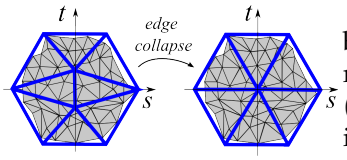
The parametric positions (s_j, t_j) of vertices v_j^M which are on the boundaries of M' are fixed during the optimization, while interior vertices are moved into a new position (s'_j, t'_j). Since the boundary is fixed and the optimization is guaranteed not to fold faces (i.e. that every triangle of M' is assigned to a positive area triangle in E), new positions (s'_j, t'_j) are necessarily inside E too.

At the end of the local optimization, mapping ϕ is updated for all vertices v_j^M of M which are interior vertices of M' using function g_E associated to E: $\phi(v_j^M)$ is redefined to $g_E(s'_j, t'_j)$

Note that, if the interpolation domain E includes several sub-domains in its image, it is possible that a certain number of vertices of M will be mapped, by the updated ϕ, on a different sub-domain of D. In this way mesh vertices are to naturally allowed to migrate from a sub-domain to another in the process of minimizing global distortion.

However, vertices of M initially mapped by ϕ near the boundary of the image of E (specifically the ones which are connected in M to a vertex outside the image of E) will not change their assigned position on parametric space D. This ensures that no folds are created outside the part of M affected by the local optimization of ϕ. Inside the affected part, no fold can be introduced either because neither the single-patch optimization of M over E, nor g_E, ever introduce internal folds.

5.3 Edge collapses



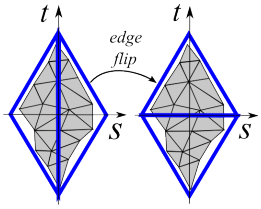
Edge collapses [30] can be performed on the domain D: the edge $e^D = (v_a^D, v_b^D)$ of D is collapsed in a new vertex v_n^D, resulting in the removal of two adjacent sub-domains D_i and D_j sharing that edge. Adjacency structures of D are updated to reflect the change (a preliminary integrity test must be performed on the connectivity structure to ensure that executing the collapse does not lead to inconsistencies [30]).

The mapping ϕ must be updated to reflect the change on D in all vertices v_j^M of M for which $\phi(v_j^M)$ was assigned to one of the sub-domains of D affected by the collapse. To do this, we use the star-domain E relative to the vertex v_n^D of D (Sec. 4.1). Two positions inside E are chosen for the positions corresponding to v_a^D and v_b^D. By doing so we are implicitly assigning a 2D position $(s_j, t_j) \in E$ to all mesh vertex v_j^M such that $\phi(v_j^M) \in D_i$, with D_i being a sub-domain of D sharing v_a^D or v_b^D.

The two positions in E are found by minimizing the isometric distortion of the mapping of all sub-domains of D affected by the collapse into E. The position of the central-vertex v_c^D in the star domain E is also shifted from the origin to minimize the introduced isometric distortion with a nonlinear least squares minimizer [31]: v_c^D is positioned so to minimize the differences between the minimal and maximal $Area(\theta(D_i))$, and minimal and maximal $Length(\theta(e_i^D))$, for all edges e_i^D and domains D_i sharing v_c^D (the values are computed over the mesh, Sec. 5.7).

At this point, the values of all $\phi(v_j)$ are reassigned to $g_E(s_j, t_j)$. After the reassignment, mapping ϕ is locally optimized over the same star-domain (Sec. 5.2) in order to ameliorate the distortion introduced locally.

5.4 Edge flips



An edge flip is another operation that can be trivially performed over the connectivity structure of D. A flip targets an edge e_i^D of the D, considers the quadrilateral composed by the two faces D_i and D_j sharing e_i^D, and divides it along the other diagonal.

Function ϕ is updated to reflect the edge flip using the diamond domain E associated to the flipped edge.

After performing an edge flip, ϕ is locally optimized (Sec. 5.2) over the four star-domains associated to each of the four vertices of D_i and D_j.

5.5 Global optimization

In this step the mapping $\phi : M \to D$ is optimized globally over all M. This is done in a series of epochs: in each epoch, we adopt each of the three partition described is Sec. 4.2 in succession (see Fig. 4). In each epoch, ϕ is optimized in each interpolation domain composing the current partition (see Sec. 4.2). After each epoch, vertices are redistributed over the other two partitions. The process is iterated until convergence (Sec. 5.6).

Inside each partition, any triangle of M which is not entirely inside a single interpolation domain will be fixed (none of its vertices will be moved during the optimization). However, thanks to the fact that the partition-set is free from permanent-boundaries (Sec. 4.2), each vertex will not be fixed for at least one of the partitions, meaning that the entire mapping will be optimized without any artificial constraint on the position of image $\phi(v_i^M)$ of a mesh vertex v_i^M.

To perform all the optimizations of a given partition P, we assign each vertex v^M of M, v^M being inside the image of a given interpolation domain $E_i \in P$, to the position $(s, t) = g_{Ei}(\phi(v^M))$. We also fix in (s, t) space all vertices belonging to triangles t^M of M which lie across two or more interpolation domains (those triangles will be inconsistent in E, having negative or too large areas, but that is irrelevant because their effect is applied only to vertices which are fixed). We can then relax the mapping globally as an unique system.

The process is bound to converge (Sec. 5.6), and, empirically, we found that this happens after a small number of epochs (ten or less). This is also due to the

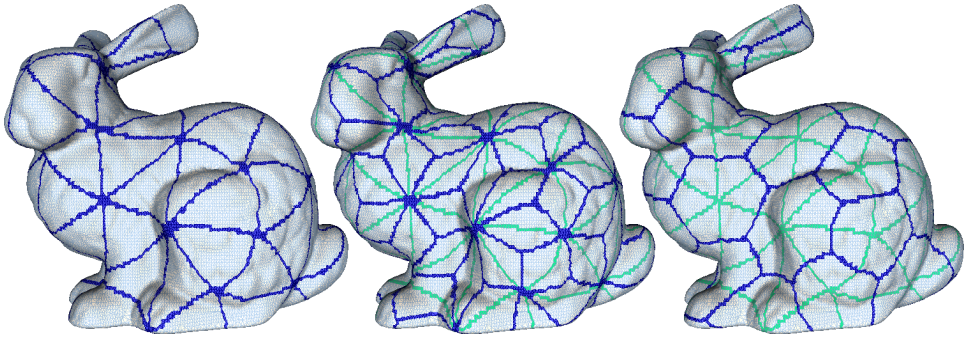


Fig. 4. The original mesh partitioned into the images of interpolation-domains; from left to right: triangle-domains, half-diamond domains, and half-star domains (triangles shared by different sub-domains are in green, and by different interpoaltion domains are in blue).

fact that in our approach ϕ is already close to optimum whenever the process is started (Sec. 5.1).

The effect of global optimization of ϕ can be interpreted in two ways: the images $\phi(v_i^M)$ of vertices $v_i^M \in M$ can be seen as moving over the abstract surface D to minimize overall distortion, i.e. improving the match the shape and relative sizes of triangles $\phi(t_j^M)$ on D with those of the original mesh triangles t_j^M; importantly, in this process vertices are ultimately free to migrate from any sub-domain to another, when this helps improving isometry. Equivalently, the process can be seen as letting the images $\theta(D_i)$ of sub-domain D_i "crawl" over the surface of the mesh M, so that they assume increasingly equilateral shapes and more equally-distributed areas.

5.6 Global optimization convergence

An inherent problem of the global optimization approach is that mapping g is isometric only in the case of any triangle, (half-)diamond, and (half-)star domains of valency 6, but not for star domains of valency different from 6 (*irregular domains*, corresponding to cone singularities). When E_i is an irregular domain then optimizing isometry of $g_{Ei}^{-1} \circ \phi$, as we propose, could backfire and worsen of isometry of ϕ. Note that such domains are needed, otherwise the partition sets would not be free from permanent-boundaries. Also, note that no isometric mapping g exists for them.

The problem can be solved, in principle, by adapting the local (single-patch) optimization process, for non-valency-6 star-domains alone, so that the optimization minimizes the isometry of ϕ rather then that of $\phi \circ g_{Ei}^{-1}$. Since the local process works on values $g_{Ei}^{-1}(\phi(v^M))$, this would require that whatever distortion measure was originally minimized by the single-patch optimization is combined with g_{Ei}. This is not always practical.

We prefer to adopt a work-around which is more modular with any single-patch optimization, allowing direct adoption of a given method. We notice that the problem only arises for a small minority of domains, and, even in these cases, the optimization of $g_{Ei}^{-1} \circ \phi$ will often improve isometry of ϕ as well. Therefore, we just check that the distortion of ϕ is not increased locally by a local optimization over irregular domains, by integrating the used measure for all triangles. If that is not the case we refuse to perform the update (5.2), effectively reverting that specific local optimization.

Enforcing the global distortion to be strictly decreasing guarantees that a convergence will be reached in any case. From a practical point of view, a local optimization (over an irregular domain) is almost never refused. The few exceptions happen when the system already close to optimum, i.e. in the last epochs of the final global optimization step. At that stage, very few mesh vertices are still changing sub-domain, so optimization steps over star-domains are not much needed anyway.

Another problem arising from the non iso-metricity of g for irregular domains is that images of mesh triangles can occasionally appear as folded, which is a problem when the adopted single-patch optimization is not robust in that respect. Such cases can be easily detected and solved by various methods. For example, we use a quick linear fold-reversing optimization, affecting only to the folded triangle and its adjacent vertices. Since triangle folding is a very rare occurrence this operation does not disrupt the overall energy minimization.

5.7 Efficient measurements of paths and regions

In order for the described approach to be practical, we need a quick way to estimate the quantities $Area(\theta(D_i))$ (the area on M of a region on the mesh corresponding to a sub-domain D_i), and $Length(\theta(e_i^D))$ (the length on M of a path corresponding to an edge e_i^D of D).

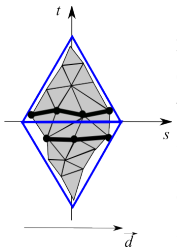
After [17], $Area(\theta(D_i))$ is approximated by associating to each vertex v^M of M the area of its Voronoi region on M, and by summing up all areas associated to vertices v_h of M mapped by current ϕ into a position inside D_i.

The length of the path $Length(\theta(e_i^D))$ on M cannot be reliably approximated by either the Euclidean distance of its end-points, or by the length of the geodesic arch connecting these endpoints. Instead, we use the diamond domain E associated to edge e_i^D (Sec. 4.1).

Let D_i and D_j be the two sub-domains sharing e_i^D. For any triangle t_i^M of M with one vertex being mapped by ϕ in D_i and other two in D_j, or viceversa, we consider

the edge connecting the latter two vertices (thick lines in the inset image). Let A be the set of the indices of these edges. Remember that $g_E^{-1}(\phi(e_i^M))$ is the counter-image, on E, of mesh edge e_i^M. Let the unit vector $\overrightarrow{d} \in E$ be the small diagonal of diamond E, i.e. the direction of the line of E which is mapped by g_E into domain edge e_i^D. Then the length of the path $\theta(e_i^D)$ is reliably approximated by

$$\frac{1}{2} \cdot \sum_{i \in A} ||e_i^M|| \cdot \left| \left(\overrightarrow{d} \bullet \frac{g_E^{-1}(\ \phi(e_i^M)\ )}{||\ g_E^{-1}(\ \phi(e_i^M)\ )\ ||} \right) \right| \qquad (5)$$



In words, paths lengths are approximated by walking on mesh edges, but each edge length,computed in $\mathbb{R}^3$, is weighted by a factor, computed in (s,t) space, determining how much that edge goes in the intended path direction. Since mesh edges fully inside either D_i or D_j are summed up, the final result must be halved. As an implementation note, lengths of all paths can be computed by iterating once over every edge of M and summing up its contribution to the only appropriate domain edge (if any). Similarly, areas can be estimated for all regions with a single iteration over all faces of M.

5.7.1 Warming-up steps

The technique described above to estimate areas of regions and lengths of paths delimiting them is only applicable after sub-domains have several mesh-vertices mapped over them. At the beginning of the process, when there is a sub-domain for each triangle of M, we need a fall-back strategy: we associate to all vertices v_i^D of D a 3D position (so the mesh which D represents is no more abstract), initially copied from M, and we estimate areas of regions and lengths of paths using standard area computations and Euclidean distances (as a consequence, only for the edge-collapses executed early in the process, we must also assign a 3D positions to the vertex resulting from the collapse; such position is determined maximizing preservation of total area and angles after the collapse, using [31]).

As soon as the number h of vertices of M assigned to a domain is larger than H_{min} (H_{min} being a constant, we use $H_{min} = 15$), the measures of region areas and path lengths are performed with the technique described above (Sec. 5.7). The transition is made smooth by interpolating the two estimates linearly with weight h/H_{min}.

When the warming-up phase of the process is over, and sub-domains of D count each more than H_{min} vertices mapped over them, domain D becomes fully abstracted and 3D positions of its vertices are no longer computed or used. In other words, the domain D starts with a $\mathbb{R}^3$ embedding taken from M, but is progressively made abstract as its geometric complexity is reduced.

5.8 Determining the ideal number of sub-domains

During the domain decimation process described in 5.1 we can keep track of the global distortion $Dist(D,\phi)$ of current couple (D,ϕ), defined as the area-weighted sum over all triangles of M of the isometric distortion (4).

To hasten the measurement, we adopt the partition of M in triangle-domains, and add up distortions only of triangles which are inside a single domain. The error introduced by this approximation is acceptable.

The total distortion, after a warming-up phase, shows an overall increase as the number of domain decreases, which is expected. However it also fluctuates unpredictably up and down with high-frequency, breaking the monotonicity: for example often a specific edge-collapse worsens dramatically the mapping, while subsequent collapses ameliorate the effect of the first one.

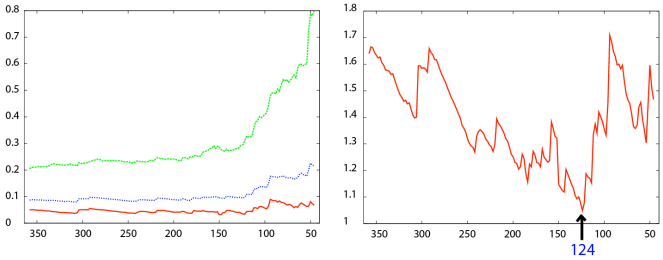


Fig. 5. Left: plots of area distortion (red) [5], angle distortion (green) [6], and combined (eq. 4), all recentered in 0, in units $10^{-}2$. Right: plot of function (6) over the diminishing number of $|D|$ for the Armadillo dataset. The sweet-spot for this dataset is found at $|D| = 124$.

When the number of desired sub-domains is not determined by the application context, we can locate the "sweet-spot" where to stop the decimation process. To determine this number we keep track of the value

$$Dist(D,\phi) \cdot \sqrt{|D|} \qquad (6)$$

and choose its minimum in the interval where $|D|$ (the number of sub-domains composing D) is inside an user specified interval. The decimation is then back-tracked to the ideal number of sub-domains.

The distortion is weighted with $|D|^{1/2}$ in order to penalize mappings which requires more sub-domains, to pursue our goal on domain simplicity. We are encouraged in the choice of the exponent $1/2$ because we noticed that, using that value, for reasons that should be investigated, equation (6) tends to exhibit oscillations around a constant value (Fig. 5).

6 Uses of the Parameterization

Many typical uses of surface parameterization (e.g. texture mapping and remeshing) can benefit from the presence of an adjacency information among sub-domains of D. Each point in D is guaranteed to have a large, easily identified neighborhood around it (possible spanning over a few sub-domains). As noted, thanks to the presence of interpolation-domains, it is possible to process entities on the mesh which lie across several sub-domains. Thus we consider the resulting structure to be conveniently "simple", in the sense described in Sec. 1.

6.0.1 Remeshings and geometry processing

A remeshing of M (or a texture for it) can be obtained by sampling regularly every interpolation domain of any of the three possible partitions of D (Sec. 4.2). However, adoption of half-diamond domains offers some extra advantage. Half-diamond domains can be sampled on a grid with lines parallel to its sides, storing 3D positions of the associated point on M in a $N \times N$ square patch (possibly with normals or other attributes).

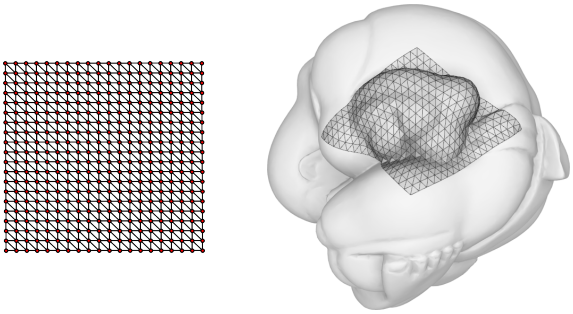


Fig. 6. The implicit connectivity among the vertices inside a squared resampled geometry patch which stores a part the remeshings of M (on the right).

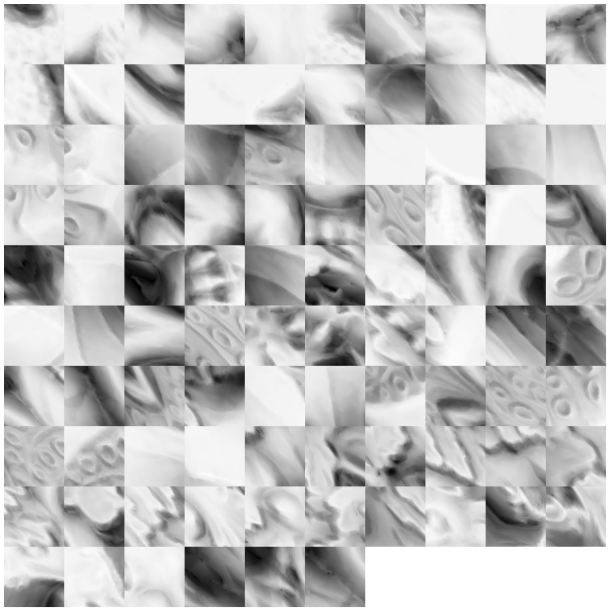


Fig. 7. A packing of the squared patches obtained sampling every half-diamond domain of D for the Gargoyle dataset.

Square patches can then be easily packed (Fig. 7). Similarly to geometry images [12], each patch implicitly defines a triangle connectivity of the sampled points (Fig. 6): each 2×2 quad of samples is split always along the same diagonal, resulting into two quasi-equilateral triangles in 3D space. Each interior point of a patch has 6 implicit neighbors, which (if the mapping is sufficiently isometric) can be considered to be at the same distance in $\mathbb{R}^3$: four over horizontal and vertical directions in the patch, and two over the fixed diagonal direction (the other diagonal should never be used). In other words, even though the mapping between the half-diamond domain and the square is not rigid, the introduced distortion is of a very predictable kind and end-applications using the remeshing can easily account for that.

An explicit adjacency information among square patches can be stored and accessed to easily perform geometry processing tasks on the mesh. For example, consider a scenario of a GPU based simulation of a phenomenon which takes place on the surface of the mesh (heat propagation, for example); the simulation computes the evolution over time a state stored in a texture with one texel for each sample, requiring access to neighbor samples. By accessing a simple connectivity structure storing patch-to-patch adjacency, a GPU based application can easily access any neighbor of a given point, including the ones on the border of the patch.

7 Results and Discussion

Fig. 13 and Table 3 show results obtained with a number of real-world datasets (coming from range scanning).

Processing times are acceptable for a typically preprocess steps as mesh parametrization, not exceeding few minutes for customer level computers (Table 3 reports times obtained on a Core2 Quad CPU, 2.4GHz, 2Gb RAM). The most time consuming steps are the ones involving the non-linear optimizations of each domain. Switching to a faster (e.g. linear), or GPU based, method can be expected to considerably shorten times.

7.1 Assessment

We can assess the quality of the parameterizations resulting from our method by different means:

7.1.1 Direct parametrization assessment

We can employ common quality measures for parametrizations, targeting area- and angle- distortion, such as one-way L2 stretch efficiency [32] (see corresponding column of Table 3 for a few of the obtained values, and Table 2 for a comparison with the state of the art). To integrate the measure over the entire mesh, we directly sum up (area weighted) contributions of each triangles of t^M computed over its image on an appropriately chosen interpolation domain E embedding all its three vertices. We select an irregular star-domains only if no other domain is available, which happens for exactly a single triangle t^M for each occurrence of a non-valency-6 domain vertex (that is, for a negligible minority of triangles of M). Since mapping g_E is an isometry in every other case (leaving any measure unaffected) the result is a very reliable approximation.

Lastly, when an embedding in 3D of D exists (with unit sided equilateral triangles), it can be displayed to provide an additional visual assessment (Fig. 11).

7.1.2 Remeshings-based assessment

In order to assess our results against other works from the literature, we also compare the remeshings implicitly defined by our parametrization (Sec. 6.0.1) against these obtained either by other parameterizations approaches, or by remeshing methods. We compared remeshings which approximately feature the same number of faces (we have chosen the sampling step over domain space that achieve this matching the most).

As a side note, the two tasks (parametrization and remeshing) do overlap but are separated: a parametrization is useful in many other contexts outside remeshing (e.g. texture mapping), and a remeshing can be obtained without going through a parametrization. Still, they are related because the list of desiderata of a remeshing task are matched by the results obtained by naturally remeshing through "good" parameterizations.

Remeshings can be visually compared for a qualitative assessment (Figures 8 and 12). As a quantitative assessment, merits of a given remeshing, regardless of its source, can be numerically assessed (Table 1) by measuring:

1) how uniform the areas of its polygons are: specifically the standard deviation, minimum and maximal of triangle area, as percentage of the average;
2) how equilaterally shaped its faces are: specifically, the minimum wedge angle over all mesh, and the average of the minimal angles of each face;
3) how uniform its edges are (the standard deviation, minimum and maximal edge length, as percentage of the average edge length);
4) how regular the connectivity is, simply by measuring the number (or percentage) of irregular vertices (since we are considering triangle-based remeshings, irregular vertices are non-valency 6 vertices).

The first point closely relates to area preservation: if the remeshing is obtained from an *area-preserving* parametrization, the standard deviation would be zero. The second point closely relates to conformality: angle preserving parametrizations result in equilateral remeshings (minimal angles of each face being 60 degrees). The third point relates to isometricity, and thus can be considered as a combination of the previous two: length preserving parametrization would produce constant edge length. The last point relates to what we termed the "simplicity" of the domain: a parameterization defined over a "less-continuous" domain (one composed of more patches), or with more irregular vertices (more cone singularities), is clearly less simple and produces resampling with more irregular vertices.

7.2 Discussion

In Table 1 we show the measures described in Sec. 7.1.2, computed for remeshings made against the parameterizations resulting from our and a few state-of-the-art similar approaches, as well as against another few state-of-the-art remeshing techniques; Fig. 8 and 12 provide

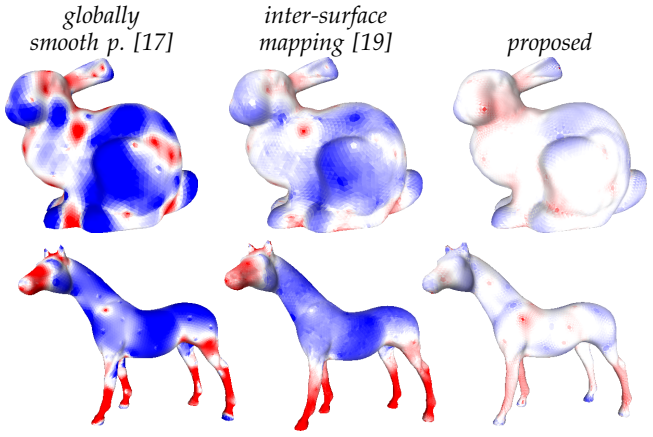


Fig. 9. Models color-coded to reflect area preservation of parameterizations obtained with different approaches. White areas are area-preserving parts of the mesh, while areas that ϕ shrinks or expands by a factor 2 or more are colored in pure blue and pure red, respectively.

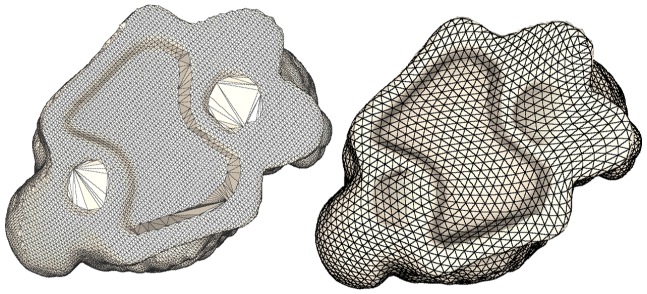


Fig. 10. The presented technique is robust with thin and uneven sized triangles in the input mesh (left). The quality of the obtained parametrization is testified by the remeshing shown on the bottom (Bunny dataset). The input mesh M was obtained with a direct re-triangulation of the large holes present on the Stanford Bunny model.

side-to-side visual comparisons. Table 2 compares L2 stretch efficiency [32] of the parameterizations.

Results indicate that the proposed technique represents a significant improvement over previous techniques. In spite of using a parametrization domain composed of fewer (often by a factor of 2) base domain tri-

dataset name	**new**	[17]	[19]	[19]*	[35]	[11]	[23]
Bunny	**1.04**	0.80	0.92	0.72	-	1.02	0.97
Horse	**1.06**	-	-	0.40	1.11	1.03	-
Venus	**1.03**	-	-	0.95	-	1.02	-
David Head	**1.06**	0.76	0.90	-	-	-	-
Armadillo	**1.12**	-	-	0.53	-	-	-

TABLE 2
L2 stretch efficiency [32] (hypothetical best is 1) of the results obtained with the proposed method, and a few ones reported by various parametrization approaches [17], [19], [35], [11], [23]. Column [19]* refers to octahedral parametrization domain.

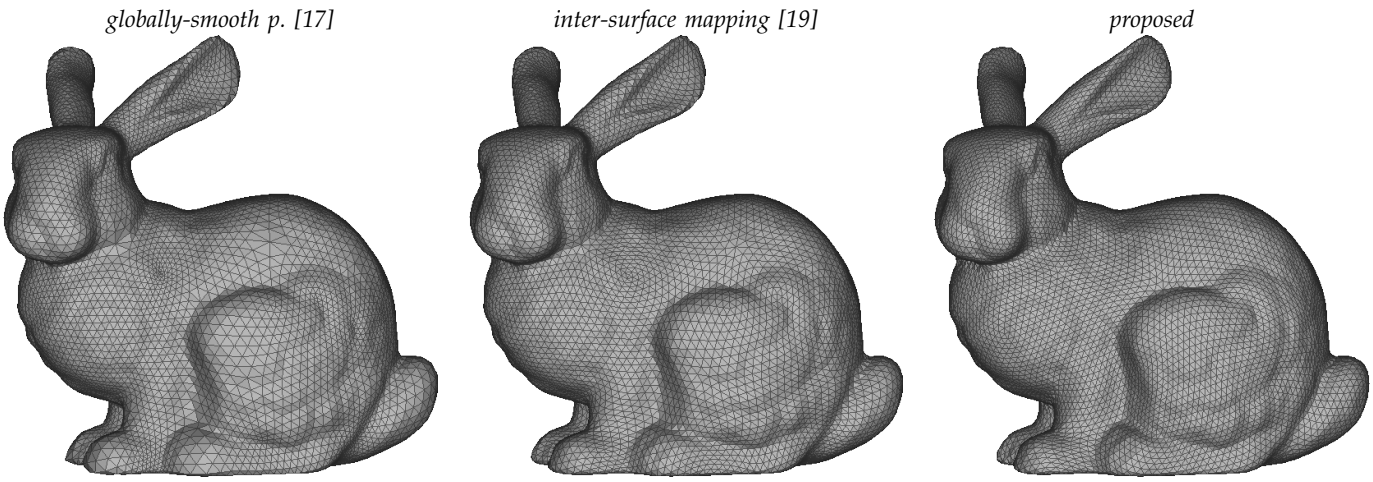


Fig. 8. Visual comparison (via remeshings, see Sec. 7.1.2) of our result on the bunny model against those from [17] and [19]. See Fig. 12 for other examples, and refer to Table 1 and 3 for data sizes and quantitative assessments.

remeshing			*(domain simplicity)*		*(area preservation)*			*(conformality)*		*(isometricity)*		
dataset name	technique	vertices	vertices of (base) domain	irregual vertices (%)	Areas (% of μ) min	max	σ	Angles min	μ of min($\triangle$)	Edges (% of μ) min	max	σ
hypothetical best $\rightarrow$			0	0	100	100	0	60	60	100	100	0
Bunny	[17]	9334	150	1.03	7.22	554.42	46.89	4.53	**54.28**	23.25	331.01	23.24
	[19]	9602	152	1.03	26.54	204.16	22.60	**16.46**	50.21	27.89	**178.35**	15.11
	new	10439	**74**	**0.49**	**31.81**	**196.65**	**11.76**	15.00	45.25	**42.40**	200.83	**14.81**
Horse	[17]	8834	140	**0.86**	1.79	642.71	60.17	**12.48**	**50.20**	13.32	439.56	33.90
	[19]	8834	140	**0.86**	13.77	268.41	44.87	8.22	47.16	29.43	216.32	25.45
	new	6339	**134**	1.27	**40.52**	**215.32**	**13.87**	10.98	45.18	**40.55**	**197.92**	**18.78**
David head (low res)	[17]	5529	90	0.71	7.93	596.89	51.77	7.66	**52.36**	20.58	332.61	26.63
	[19]	5698	91	0.71	5.19	265.88	25.34	4.29	49.69	21.21	229.10	16.31
	new	5441	**39**	**0.34**	**24.10**	**181.10**	**14.20**	**10.80**	45.77	**24.03**	**182.90**	**14.90**
Igea	[17]	9730	15	0.22	4.02	355.19	39.16	20.31	**57.47**	19.07	211.79	20.27
	[33]	9240	–	4.40	31.45	279.04	39.16	25.80	53.30	47.27	190.75	18.52
	new	8672	**12**	**0.08**	**47.66**	**142.83**	**7.57**	**25.22**	47.99	**58.14**	**145.30**	**13.01**
David head	[34]	10783	–	35.80	10.51	300.94	18.83	4.34	**49.88**	26.59	305.85	**15.08**
	new	11102	**39**	**0.17**	**20.80**	**194.71**	**13.11**	**12.00**	45.55	**34.92**	**210.46**	15.90

TABLE 1

Statistical measures over the remeshings we obtained through the parameterizations resulted from the proposed approach, compared with those resulting from other known parametrization [17], [19] or remeshing [33], [34] techniques. See Sec. 7.1.2 for a more detailed explanation of column values. Remeshing are visible in Fig. 12 and 8.

angles, meaning a more regular and simple domain, the parametrization introduces a much lower area distortion (see also Fig. 9). This is not achieved at the expense of angle preservation, which is almost unaffected (and often even improved). Measured L2 stretch efficiency (Table 2) confirms this result. Whenever a direct comparison was possible, our technique proved to produce the least stretched parametrizations in spite of its superior domain simplicity, i.e. small number of identical patches separated by regular straight boundaries (actually the only technique that we found to reach only slightly lower stretches is [11], an atlas based approach with several small patches featuring long, irregularly shaped borders). Visual comparisons confirm these findings (Fig. 8 and 12).

Part of the reason is that our method allows to adopt any single-chart optimization technique, including those which explicitly take in account area preservation as well as angle preservation [5]. Another key aspect lies in the way the domain space is likewise constructed seeking both objectives (area and angle preservation). Use of both edge flips and edge collapses is crucial to achieve this result (as is the ability to measure lengths and areas on the original mesh to better drive the choices of which operation to perform). We believe that, as a result, better, more fitting parameter domain spaces D are produced.

In facts, we noticed that the degree of the vertices of the domain D tends to be proportional to the discrete scale-dependent Gaussian curvature of M [36] computed at the appropriate scale (i.e. the average path length for

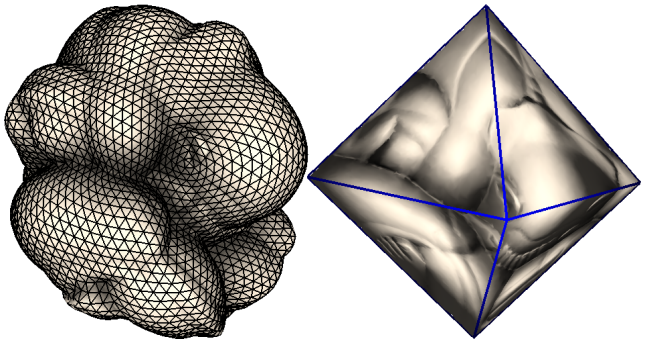


Fig. 11. A remeshing of the OmoTondo dataset, and its domain (embedded in $\mathbb{R}^3$ for illustration purposes). With the presented technique, spherical parameterizations as in [1] arise naturally, in the special case when the input shape is similar to a sphere.

dataset name	$\|M\|$ ($\sharp\triangle$)	$\|D\|$ ($\sharp\triangle$)	*(distortion)* Area	Angle	L^2	*time* (s)
Moai	16426	20	0.39	5.32	1.03	97
Omotondo	30000	8	0.33	7.01	1.03	368
Armadillo	40000	124	2.13	17.30	1.12	181
Gargoyle	49980	64	0.92	12.47	1.07	317
Bunny	69664	142	0.64	7.63	1.04	289
Fertility	80000	204	1.29	12.46	1.06	357
Rampant	99746	262	1.26	14.98	1.17	540

TABLE 3
Examples of obtained parametrizations. For each dataset, we report: number of original faces, number of sub-domains composing the domain, area distortion [5], angle distortion [6] (both recentered in 0, in 10^{-2} units), L2 stretch efficiency [32], and total processing time.

that model). Vertices of D with >6 degree are usually mapped in parts of the mesh with strong negative curvature, and the opposite. This correspondence contributes strongly to lower angle and area distortions.

We can offer an informal explanation for this correspondence. Consider a vertex v_i^D on D and the k vertices in its 1 star v_j^D, $j \in \{1..k\}$. If each path among two connected vertices is actually the same length ω, which happens if our goal is fulfilled (Sec. 5), then $\forall j$, $\theta(v_j^D)$ is on the perimeter (of an approximation of) a disks or radius ω around $\theta(v_k^D)$, and the length of its perimeter is approx. $k\omega$. From here the phenomenon can be derived.

Compare with approaches where cone-singularities locations are carefully selected to soak distortion, in a process driven by Gaussian curvature [14], [15]. Non valency 6 vertices of D are our cone-singularities: instead of directly identifying a good location for them, we let it emerge as a side effect of an easily pursed goal (that all paths over the mesh have the same length) eventually achieving a similar result. We believe this can be practical and advantageous because it naturally mediates the need of good cone-singularities positioning with other useful desiderata (like domain simplicity, i.e. the fact that the domain is composed by the fewest faces, and strictly equilateral ones). Hybrid method can be obtained: our method can naturally be extended to let an user (or an automatic Gaussian-curvature driven method) mark a few vertices over the input mesh to host cone-singularities: the local operations would then be constrained to maintain these vertices in D.

It is interesting to note that our technique, when the input mesh is almost spherical, gives as output exactly a spherical geometry-image parametrization (Fig. 11). Our method to identify the ideal number of sub-domains (Sec. 5.8) prescribes a domain with 8 faces, resulting in a connectivity of an octahedron. In this sense, this technique can be see as an extension of [1] for meshes with more complex shapes and genuses. Not surprisingly, another local minimum of function (6) happens at 20 faces, when the connectivity is that of an icosahedron.

In summary, we have shown how good parameterizations of arbitrary genus meshes can be automatically constructed, with very low isometric distortion and defined over a convenient adaptive domain with is easy to use in many application requiring the optimization.

Limits: One intrinsic limitation of our approach is that the topology of the original mesh is preserved by the parametrization domain D. This means that meshes with small topological features, for example meshes with topological noise or small holes and missing triangles, must be pre-processed in order to remove these inconsistencies before the technique described here can be applied [37], [38]. The worst cases consist in meshes with topological features (e.g. small handles) that must be preserved correctly and still are very small compared to the size of the rest of the mesh: in these cases either the domain D must be composed by a big number of sub-domains, or the distortion will locally be large.

While the method is robust and consistently produces good results (except for the above cases), it cannot be expected to be fully independent on initial mesh tessellation. Starting with similar but not identical meshes (for example, two different meshing of the same geometry) the method produces base domains D are usually very similar but sometimes not identical (unless base domain is very simple, as in the example visible in Fig. 11).

Implementation: An implementation of the described technique is available within the open-source 3D mesh processing system MeshLab [39].

Acknowledgments

We would like to thank Andrei Khodakovsky, Vitaly Surazhsky, John Schreiner and Peter Schröder for sharing their data for comparison purposes. We are also thankful to Miguel A. Otaduy for the useful discussions.

The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement n. 231809 (IP project "3D-COFORM").

REFERENCES

[1] E. Praun and H. Hoppe, "Spherical parametrization and remeshing," *ACM Trans. Graph.*, vol. 22, no. 3, pp. 340–349, 2003.

[2] M. Tarini, K. Hormann, P. Cignoni, and C. Montani, "Polycube-maps," *ACM Trans. Graph.*, vol. 23, no. 3, pp. 853–860, 2004.

[3] A. Sheffer, E. Praun, and K. Rose, "Mesh parameterization methods and their applications," *Found. Trends Comp. Graph. Vis.*, vol. 2, no. 2, pp. 105–171, 2006.

[4] M. S. Floater and K. Hormann, "Surface parameterization: a tutorial and survey," in *Adv. in Multires. for Geom. Model.*, ser. Math. and Vis. Springer, 2005, pp. 157–186.

[5] P. Degener, J. Meseth, and R. Klein, "An adaptable surface parameterization method," in *Proc. of the 12th Intern. Meshing Roundtable*, 2003, pp. 201–213.

[6] K. Hormann and G. Greiner, "MIPS: An efficient global parametrization method," in *Curve and Surface Design*, ser. Innov. in Appl. Math. Vanderbilt Uni. Press, 2000, pp. 153–162.

[7] P. Mullen, Y. Tong, P. Alliez, and M. Desbrun, "Spectral conformal parameterization," *Comp. Graph. Forum*, vol. 27, no. 5, pp. 1487–1494, Jul. 2008.

[8] L. Liu, L. Zhang, Y. Xu, C. Gotsman, and S. J. Gortler, "A local/global approach to mesh parameterization," *Comp. Graph. Forum*, vol. 27, no. 5, pp. 1495–1504, Jul. 2008.

[9] N. A. Carr, J. Hoberock, K. Crane, and J. C. Hart, "Rectangular multi-chart geometry images," in *Eurographics Symp. on Geom. Proc.*, 2006, pp. 181–190.

[10] B. Lévy, S. Petitjean, N. Ray, and J. Maillot, "Least squares conformal maps for automatic texture atlas generation," *ACM Trans. Graph.*, vol. 21, no. 3, pp. 362–371, 2002.

[11] E. Zhang, K. Mischaikow, and G. Turk, "Feature-based surface parameterization and texture mapping," *ACM Trans. Graph.*, vol. 24, no. 1, pp. 1–27, 2005.

[12] X. Gu, S. J. Gortler, and H. Hoppe, "Geometry images," *ACM Trans. Graph.*, pp. 355–361, 2002.

[13] L. Kharevych, B. Springborn, and P. Schröder, "Discrete conformal mappings via circle patterns," *ACM Trans. Graph.*, vol. 25, no. 2, pp. 412–438, 2006.

[14] M. Ben-Chen, C. Gotsman, and G. Bunin, "Conformal flattening by curvature prescription and metric scaling," *Comp. Graph. Forum*, vol. 27, no. 2, pp. 449–458, 2008.

[15] Y.-L. Yang, J. Kim, F. Luo, S.-M. Hu, and X. Gu, "Optimal surface parameterization using inverse curvature map," *IEEE Trans. on Visualiz. and Comp. Graph.*, vol. 14, no. 5, pp. 1054–1066, 2008.

[16] A. W. F. Lee, W. Sweldens, P. Schröder, L. Cowsar, and D. Dobkin, "Maps: Multiresolution adaptive parameterization of surfaces," *Comp. Graph. Proc.*, pp. 95–104, 1998.

[17] A. Khodakovsky, N. Litke, and P. Schröder, "Globally smooth parameterizations with low distortion," *ACM Trans. Graph.*, vol. 22, no. 3, pp. 350–357, 2003.

[18] C. Gotsman, X. Gu, and A. Sheffer, "Fundamentals of spherical parameterization for 3d meshes," *ACM Trans. Graph.*, vol. 22, no. 3, pp. 358–363, 2003.

[19] J. Schreiner, A. Asirvatham, E. Praun, and H. Hoppe, "Inter-surface mapping," *ACM Trans. Graph.*, vol. 23, no. 3, pp. 870–877, 2004.

[20] V. Kraevoy and A. Sheffer, "Cross-parameterization and compatible remeshing of 3d models," *ACM Trans. Graph.*, vol. 23, no. 3, pp. 861–869, 2004.

[21] E. Praun, W. Sweldens, and P. Schröder, "Consistent mesh parameterizations," in *Comp. Graph. Proc.* ACM Press / SIGGRAPH, 2001, pp. 179–184.

[22] F. Kälberer, M. Nieser, and K. Polthier, "Quadcover - surface parameterization using branched coverings," *Comp. Graph. Forum*, vol. 26, no. 3, pp. 375–384, Sep. 2007.

[23] S. Dong, P.-T. Bremer, M. Garland, V. Pascucci, and J. C. Hart, "Spectral surface quadrangulation," *ACM Trans. Graph.*, vol. 25, no. 3, pp. 1057–1066, 2006.

[24] Y. Tong, P. Alliez, D. Cohen-Steiner, and M. Desbrun, "Designing quadrangulations with discrete harmonic forms," in *Eurographics Symp. on Geom. Proc.*, 2006, pp. 201–210.

[25] M. Jin, J. Kim, F. Luo, and X. Gu, "Discrete surface ricci flow," *IEEE Trans. Vis. Comp. Graph.*, vol. 14, no. 5, pp. 1030–1043, 2008.

[26] B. Springborn, P. Schröder, and U. Pinkall, "Conformal equivalence of triangle meshes," *ACM Trans. Graph.*, vol. 27, no. 3, pp. 1–11, 2008.

[27] F. Luo, X. Gu, and J. Dai, *Variational Principles for Discrete Surfaces*. High Education Press and Int'l Press, 2007.

[28] A. Dominitz and A. Tannenbaum, "Texture mapping via optimal mass transport," *IEEE Trans. Vis. Comp. Graph.*, [in press].

[29] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle, "Mesh optimization," in *Comp. Graph. Proc.* ACM Press / SIGGRAPH, 1993, pp. 19–26.

[30] T. K. Dey, H. Edelsbrunner, S. Guha, and D. V. Nekhayev, "Topology preserving edge contraction," *Publ. Inst. Math.*, vol. 66, pp. 23–45, 1999.

[31] M. Lourakis, "Levmar: Levenberg-marquardt nonlinear least squares algorithms in C/C++," http://www.ics.forth.gr/lourakis/levmar/, Jul. 2004.

[32] P. V. Sander, J. Snyder, S. J. Gortler, and H. Hoppe, "Texture mapping progressive meshes," in *Comp. Graph. Proc.* New York, NY, USA: ACM Press / SIGGRAPH, 2001, pp. 409–416.

[33] V. Surazhsky and C. Gotsman, "Explicit surface remeshing," in *Eurographics Symp. on Geom. Proc.*, 2003, pp. 17–28.

[34] V. Surazhsky, P. Alliez, and C. Gotsman, "Isotropic remeshing of surfaces: a local parameterization approach," in *Proc. of 12th Inter. Meshing Roundtable*, Sep. 2003, pp. 215–224.

[35] R. Zayer, B. Lévy, and H.-P. Seidel, "Linear angle based parameterization," in *Eurographics Symp. on Geom. Proc.*, 2007, pp. 135–141.

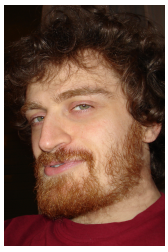
[36] H. Pottmann, J. Wallner, Y.-L. Yang, Y.-K. Lai, and S.-M. Hu, "Principal curvatures from the integral invariant viewpoint," *Comp. Aided Geom. Des.*, vol. 24, no. 8-9, pp. 428–442, 2007.

[37] I. Guskov and Z. J. Wood, "Topological noise removal," in *Graph. Interf. 2001*. Canadian Info. Proces. Soc., 2001, pp. 19–26.

[38] P. Liepa, "Filling holes in meshes," in *Eurographics Symp. on Geom. Proc.*, 2003, pp. 200–205.

[39] P. Cignoni, M. Corsini, and G. Ranzuglia, "MeshLab: an open-source 3D mesh processing system," *ERCIM News*, no. 73, pp. 45–46, April 2008. [Online]. Available: http://meshlab.sourceforge.net

Nico Pietroni is a researcher at the Istituto di the Scienza e Tecnologie dell'Informazione (ISTI) of the National Research Council (CNR) in Pisa, Italy. His research interests include mesh parametrization, texture synthesis and deformable object modeling. He received in 2003 an advanced degree in Computer Science (Laurea) from the University of Pisa and in 2009 a Ph.D. Degree in Computer Science at the University of Genova.



Marco Tarini (Ph.D. 2003, Univ. of Pisa) is an Assistant Professor at the Univ. of Insubria (Varese, Italy) and an Associate Researcher with CNR-ISTI. He teaches and researches in Computer Graphics, his main published contributions being in 3D surface acquisition, modelling, parametrization, real-time rendering, and scientific visualization. Marie Curie Mobility Fellow in 2001 (spent in MPI-Saarbrüken). He received "Best Young Researcher" award by the Eurographics association in 2006.

Paolo Cignoni is a Senior Research Scientist with CNR-ISTI. He received a Ph.D. Degree in C.S. at the University of Pisa in 1998. He has been awarded "Best Young Researcher" by the Eurographics association in 2004. His research interests cover Computer Graphics fields ranging from visualization and processing of huge 3D datasets, to 3D scanning in the cultural heritage field and to Scientific Visualization. He has published more than one hundred papers in international journals and conferences.

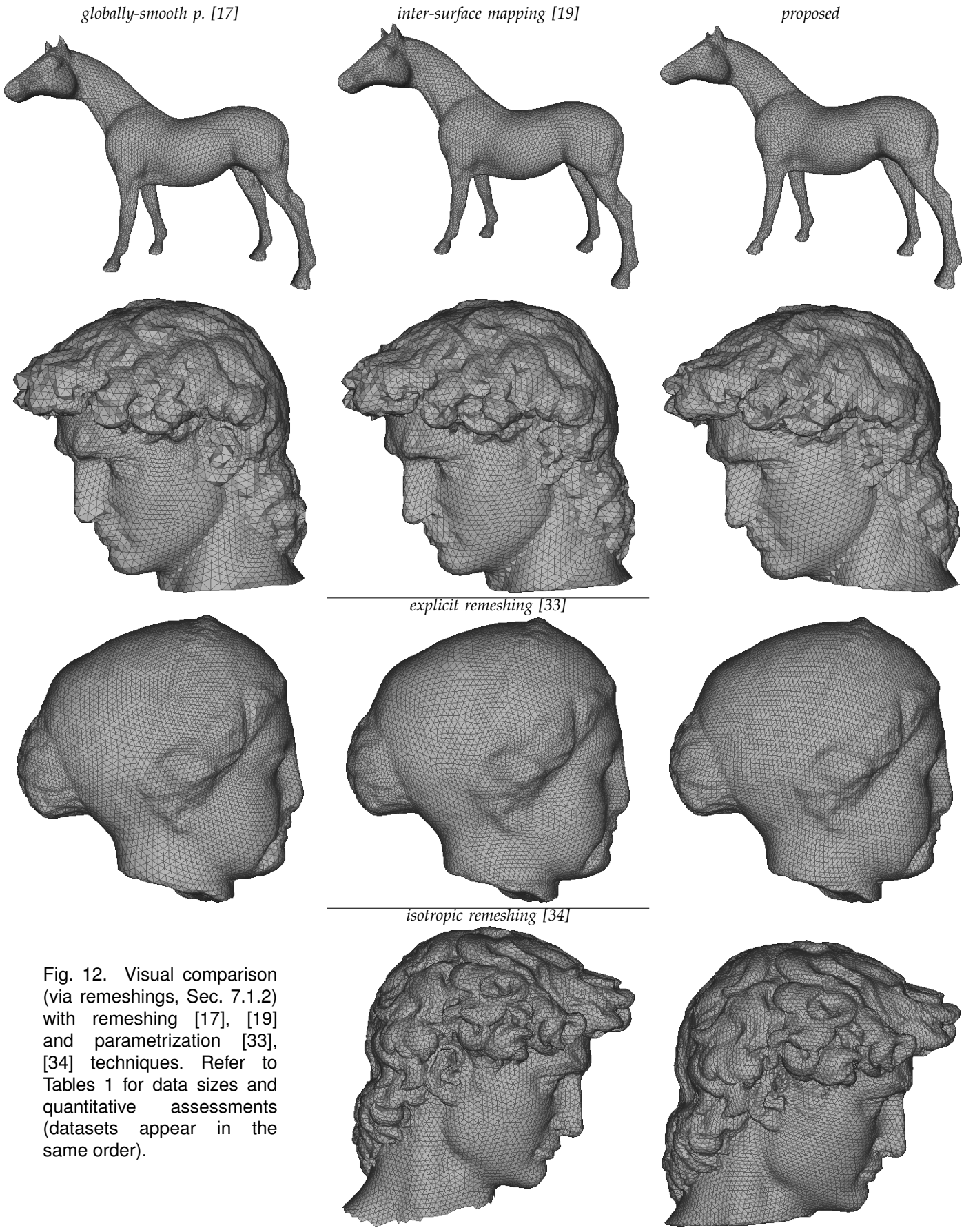


Fig. 12. Visual comparison (via remeshings, Sec. 7.1.2) with remeshing [17], [19] and parametrization [33], [34] techniques. Refer to Tables 1 for data sizes and quantitative assessments (datasets appear in the same order).

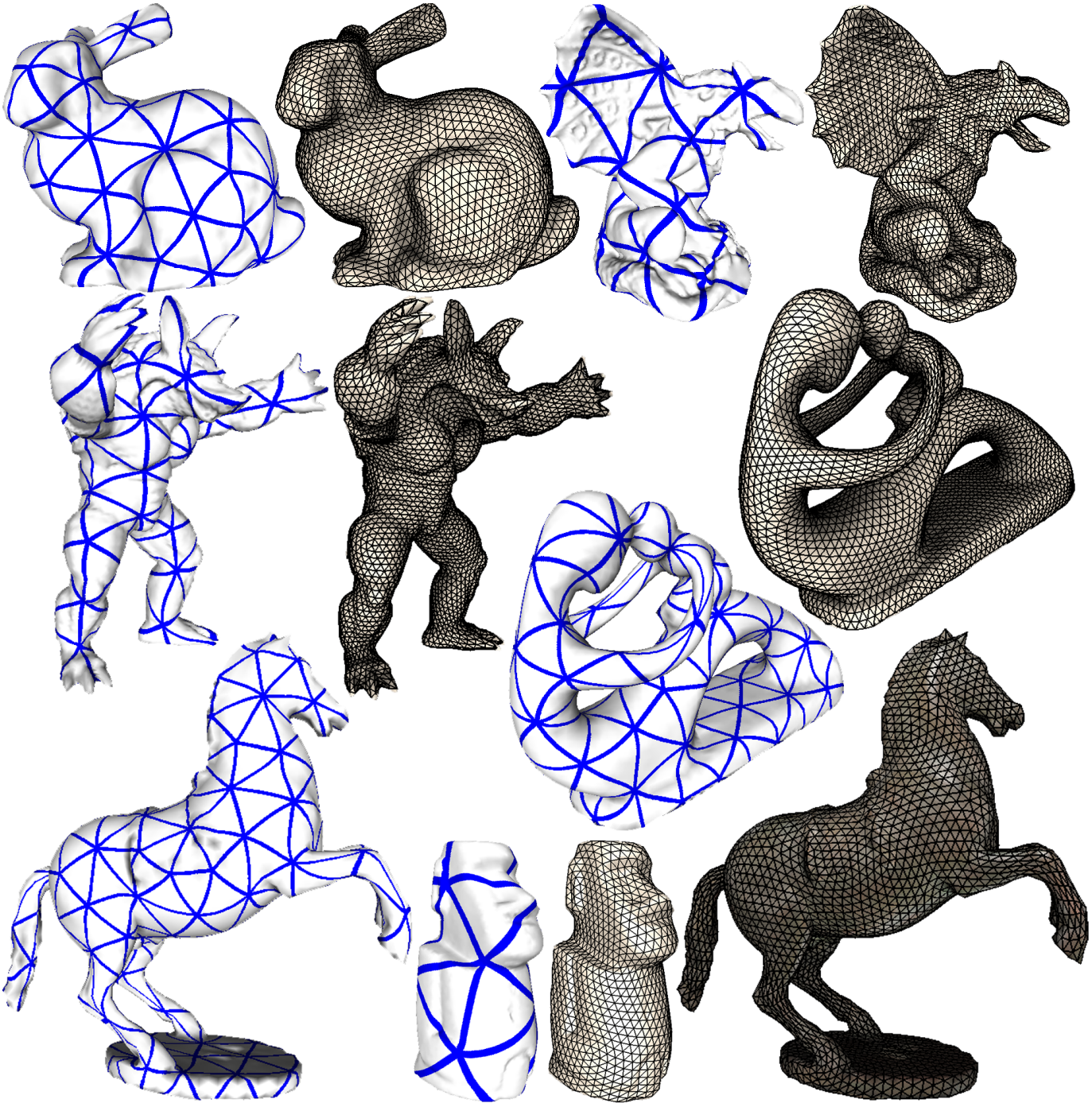


Fig. 13. Examples of the application of the presented technique to various datasets (see Table 3). For each mesh we show a remeshing done using the parameterization, and the subdivision of the mesh using the triangle-domain partition.