Contents lists available at ScienceDirect

Computer Aided Geometric Design

www.elsevier.com/locate/cagd

Quadrangulation of non-rigid objects using deformation metrics

Jiaran Zhou [a,c], Marcel Campen [b], Denis Zorin [c], Changhe Tu [a], Claudio T. Silva [c]

[a] *Shandong University, China*
[b] *Osnabrück University, Germany*
[c] *New York University, USA*

ARTICLE INFO

Article history:
Available online 22 March 2018

Keywords:
Quad mesh
Mesh generation
Cross field
Surface parametrization
Animation

ABSTRACT

We present a novel method to generate quad meshes for non-rigid objects. Our method takes into account the geometry of a collection of key poses in one-to-one correspondence or even an entire animation sequence. From this input, on a common computational domain, an extremal metric is computed that captures the local worst case behavior in terms of distortion as the object undergoes deformation. An anisotropic, non-uniformly sized quad mesh is then generated based on this metric. This mesh avoids undersampling when deformed into any of the poses specified in the input and thus reduces artifacts. Hence, in contrast to previous approaches which target static geometry, our method aims to optimize the mesh's adaptation to the shape for every pose expected during animation or deformation rather than for one specific reference state.

1. Introduction

In recent years a variety of versatile methods for the automatic quadrangular remeshing of surfaces has been developed. These methods typically optimize the properties of the quad mesh, such as element *size*, *anisotropy*, *orientation*, and mesh *connectivity*, with respect to the specific given surface geometry. Note that this is appropriate only when dealing with rigid objects with static geometric properties. At the same time, it is common for quad meshes to be used in animation or simulation. In this case, while the quad mesh might be well adapted to a particular specific state or pose (based on which it was generated and optimized), it may be a poor match for other poses or states of deformation (cf. Figs. 1, 8, and 9).

We present a method that aims to produce quadrangulations *adequate for every pose* during deformation or animation, rather than *optimal for a single pose* (cf. Figs. 1 and 2). If known a priori, the deformation can be given as input in form of a complete surface animation sequence. Otherwise, the expected space of deformation can be outlined by a set of extremal key poses.

Based on one-to-one correspondences between the key poses or animation frames we analyze the deformation structure and construct an extremal metric corresponding to the worst-case local metric behavior at every point of the surface. Furthermore, information about preferable mesh element orientation is determined jointly over the expected deformation. Together, this allows us to generate an adapted quad mesh with anisotropic element sizing that avoids undersampling artifacts due to being too coarse in any region for any pose when being deformed.

E-mail address: jiaran.zhou@gmail.com (J. Zhou).

https://doi.org/10.1016/j.cagd.2018.03.003
0167-8396/© 2018 Elsevier B.V. All rights reserved.

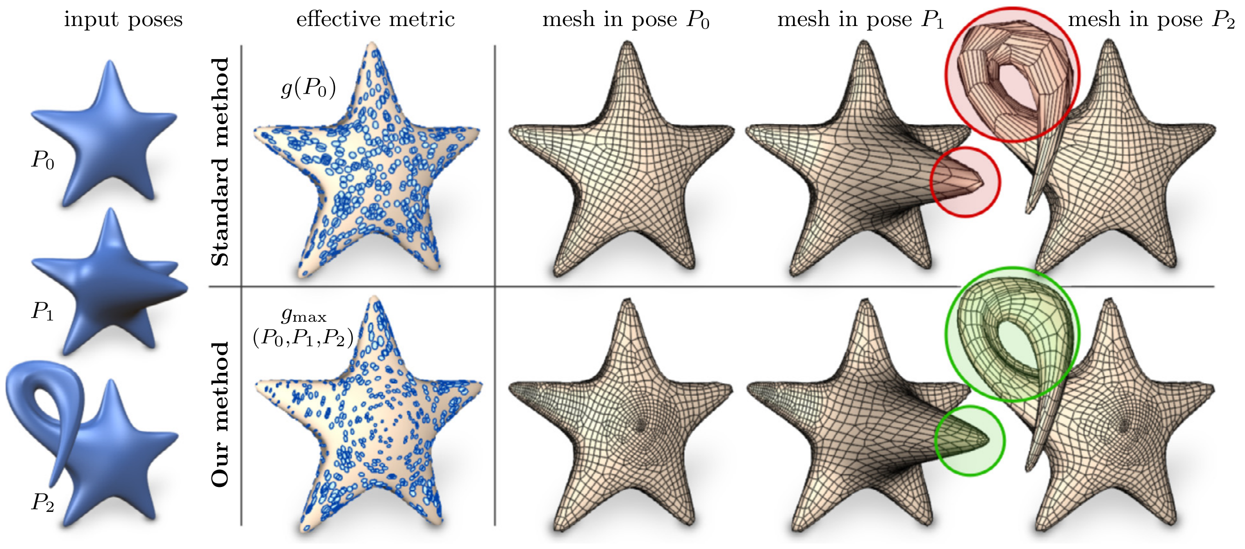

Fig. 1. Illustration of our deformation-aware mesh generation method in contrast to a standard, static method (Jakob et al., 2015). Three poses (left) are taken as input. The standard method optimizes the mesh with respect to the metric of one specific pose, P_0; ours uses an extremal metric it computes over all poses. In the top row, the resulting quad mesh deformed into the poses other than P_0 exhibits artifacts, especially in areas of significant stretching and bending (red). Our quad mesh (bottom row) shows better behavior (despite even having a slightly lower total number of elements) in these poses. Notice that this comes with an increased number of irregular vertices; these are induced by the stronger local variation in element size and shape due to deformation-awareness. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

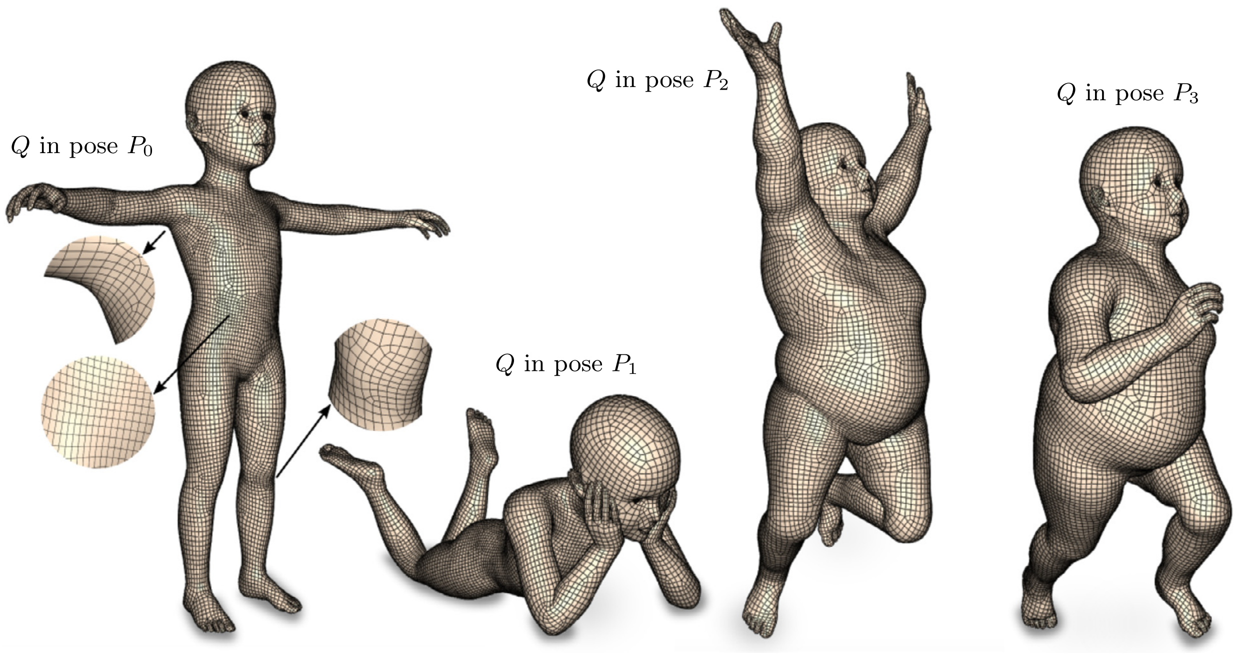

Fig. 2. The same quad mesh Q in four different poses for which it was jointly optimized using our technique with extremal metric. Notice, for instance, the higher density of the mesh in the left pose on the knees (because they get bent strongly in poses P_1 and P_2), under the arm (because it gets stretched in pose P_2), and on the belly (because poses P_2 and P_3 are more corpulent).

2. Related work

Quad meshing An account of modern quadrangular surface remeshing techniques is given in a recent survey (Bommes et al., 2013b). Parametrization based techniques have received the most attention in recent years for their flexibility and result quality (Kälberer et al., 2007; Bommes et al., 2009, 2013a; Pietroni et al., 2011; Liu et al., 2011; Myles and Zorin, 2013; Panozzo et al., 2014; Ebke et al., 2014; Campen et al., 2015; Campen and Zorin, 2017). These methods all optimize the mesh properties for one specific surface pose.

A step towards taking surface deformation or animation into account has been made by Marcias et al. (2013): the orientation of mesh elements is chosen based on the principal directions of local surface stretch during deformation. The work by Meng and He (2016) considers combined feature determination over multiple poses, again to influence mesh element orientation. Our contribution is orthogonal and complementary to these works: we focus on the determination of appropriate element *sizing*, *anisotropy*, and *shear*; it can be used in conjunction with these techniques that determine *orientation*, as well as other, e.g. principal curvature direction based techniques, as detailed in Section 5. Interactive, user-assisted quadrangulation techniques (Takayama et al., 2013; Marcias et al., 2015) are another option for mesh generation: the user can design the mesh taking knowledge about expected deformation behavior into account.

Cross fields A key component of quad remeshing techniques based on parametrization is the computation of 4-symmetric direction fields. These fields predetermine the parametrization's singularities and thus the quad mesh's irregular vertex structure, and provide directional guidance information for the parametrization's isolines (which correspond to the quad mesh's edges). Key contributions in this context concern the efficient handling of the 4-symmetry (Hertzmann and Zorin, 2000; Palacios and Zhang, 2007; Knöppel et al., 2013), control over the field topology (Li et al., 2006; Ray et al., 2008; Bommes et al., 2009; Crane et al., 2010), and generalization to non-orthonormal directions (Panozzo et al., 2014; Diamanti et al., 2014; Jiang et al., 2015). A recent survey (Vaxman et al., 2016) provides a detailed overview of the topic.

Custom metrics In some instances of previous work properties of cross fields or parametrizations are optimized with respect to a metric different from the standard Euclidean metric. For instance, the surface under consideration can be virtually smoothed in this way, geometrically (Ray et al., 2009) or even topologically (Ebke et al., 2014); the shape operator can be used as metric tensor to improve approximation properties (Kovacs et al., 2011; Heckbert and Garland, 1999). Also user-designed metrics can be taken into account, e.g. for the creation of cross or frame fields with spatially varying magnitude and skewness (Jiang et al., 2015; Pal et al., 2014).

Deformation-aware meshing A number of works have addressed the problem of generating *triangle* meshes for non-rigid models. They proceed by decimating an overly dense mesh, taking an animation sequence or individual key poses into account (Mohr and Gleicher, 2003; DeCoro and Rusinkiewicz, 2005; Landreneau and Schaefer, 2009; Lee et al., 2011). This is generally accomplished by averaging the employed decimation error functions over the animation frames or poses. In Section 4.1, we discuss why direct averaging can be inadequate in our context. Several authors proposed techniques to dynamically adjust the mesh connectivity during animation (Kircher and Garland, 2005; Huang et al., 2006; Payan et al., 2007) for application scenarios where this is appropriate. We focus here on the case of a quad mesh with static connectivity, because 1) this is more generally applicable in the context of animation and simulation and 2) the set of possible connectivity changes in quad meshes is less rich compared to the triangle case (Peng et al., 2011; Bommes et al., 2011; Tarini et al., 2010), even more so if not only proper connectivity but element quality is taken into consideration.

3. Overview

The input to our method is a set of N surfaces P_i, called poses, with pairwise diffeomorphisms $h_{ij} : P_i \to P_j$ mapping between them. When the poses are obtained through deformation from a reference pose, say P_0, such maps are of course immediately available.

The sequence of poses can be the frames of a complete animation sequence, or a set of key poses which ideally are extremal, in the sense that other expected deformation states are likely to lie within the convex hull, i.e. could be obtained by interpolation between some of the key poses. The input order of poses plays no role in our method.

With existing quad meshing algorithms one could generate a quad mesh optimized for an individual pose P_i (based on its Euclidean metric, or any other shape related metric, cf. Sections 2 and 4). By means of the maps h_{ij}, one can deform this quad mesh into a quad mesh for any other pose P_j. However, it may be unfit for those poses, with artifacts due to undersampling, stretch, or shear, as illustrated in Fig. 1.

In our approach we express the target metrics of the individual poses on one common computational domain and combine them into an extremal metric that expresses what quad mesh element sizing and anisotropy is appropriate for all poses, avoiding undersampling in all directions, for all poses (Section 4). Intuitively, elements are made smaller in regions that get stretched in some pose, so they do not become overly large when deformed accordingly. Similarly, information about preferred element orientation (based on shape-derived properties such as principal curvature directions, or on deformation-derived properties such as principal stretch directions, Marcias et al., 2013) is combined on the common domain (Section 5). A quad mesh can then be optimized with respect to this metric and orientation information using (variations of) state-of-the-art techniques (Section 6). We demonstrate that this strategy yields meshes which behave better when undergoing deformation (Section 7). Fig. 3 shows a visual overview.

4. Deformation-aware metric

In order to enable control over the sizing and anisotropy of mesh elements one can approach the quad mesh generation problem as follows: let a standard quad meshing method simply aim for unit square quads, while adjusting the notion of

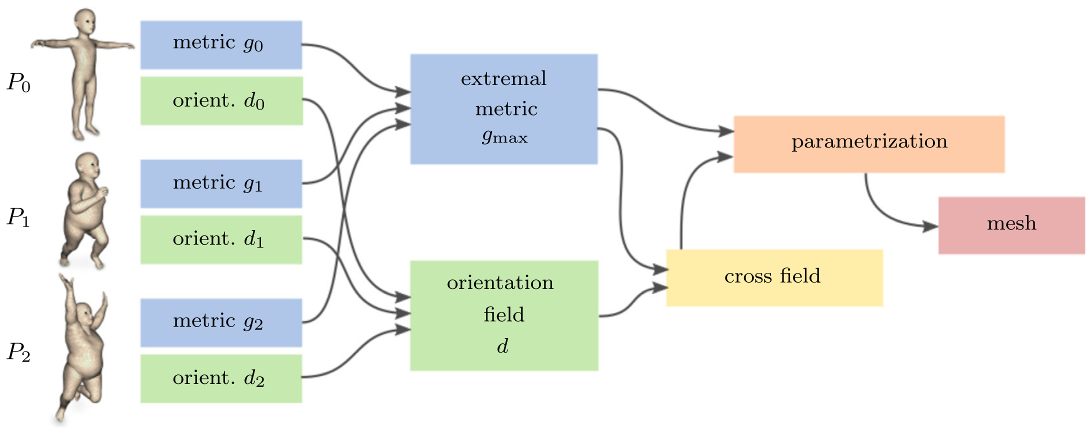

Fig. 3. Overview of our method: Multiple poses or frames (P_0, P_1, P_2) are taken as input. Metrics g_i and orientation fields d_i are computed per pose. The metrics are consolidated into an extremal metric $g_{\max}$ on a common domain. Orientation fields are combined through appropriate averaging operations. Based on this a cross (or frame) field and a field guided parametrization are computed. Finally, a quad mesh is extracted from the parametrization.

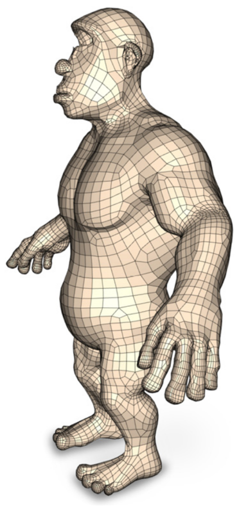

Fig. 4. The effect of the metric choice $g_i = sS_i^T S_i$ on an example model.

"unit square" by means of a custom metric on the surface. For a background on the concepts of differential geometry used in the following, we refer to the introduction by Frankel (2011).

Let g_i be a Riemannian metric tensor field on pose P_i. As usual, for a choice of local tangent space bases these quadratic forms can be represented by matrices, $g_i(x) \in \mathbb{R}^{2\times 2}$, for every point x on P_i.

The following types of metrics are of particular relevance:

- $g_i = sI$ uniform, isotropic
- $g_i = s_i(x)I$ non-uniform, isotropic
- $g_i = A_i(x)$ non-uniform, anisotropic

Here $I \in \mathbb{R}^{2\times 2}$ is the identity, $s \in \mathbb{R}$ is a global scaling factor, $s_i : P_i \to \mathbb{R}$ is a scalar sizing field on P_i, and $A_i : P_i \to \mathbb{R}^{2\times 2}$ is an arbitrary field of positive definite quadratic forms, i.e. symmetric positive definite matrices, on P_i.

Anisotropically sized elements are particularly of interest to minimize and equalize discretization errors. A choice of $g_i = sS_i$, where S_i is the shape operator on P_i (whose eigenvalues and eigenvectors are the principal curvatures $\kappa_{\min}$ and $\kappa_{\max}$ and their directions $d_{\min}$ and $d_{\max}$, respectively) is known to optimize the geometric approximation of the surface (Heckbert and Garland, 1999) by the resulting mesh in the limit of small quads. A choice of $g_i = sS_i^T S_i$ (cf. Fig. 4) is known to optimize approximation in terms of equalized surface normal error (Kovacs et al., 2011). In practice, to ensure positive definiteness in regions of vanishing surface curvature, these can be regularized by adding εI with $\varepsilon > 0$. More generally, for the approximation of arbitrary functions (surface signals), $g_i = sH_i$ is L_2-optimal (Nadler, 1986), where H is the function's Hessian.

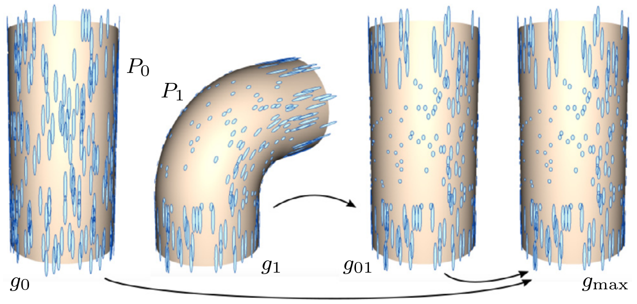

Fig. 5. Visualization of metric tensors at random points as ellipses which are unit circles under the respective metric. The metric shown here is based on the shape operator S. Metric g_1 on pose P_1 is mapped to pose P_0 as g_{01}. Then g_{00} ($=g_0$) and g_{01} are combined to $g_{\max}$ on P_0 (which in this particular example is essentially equal to g_{01}).

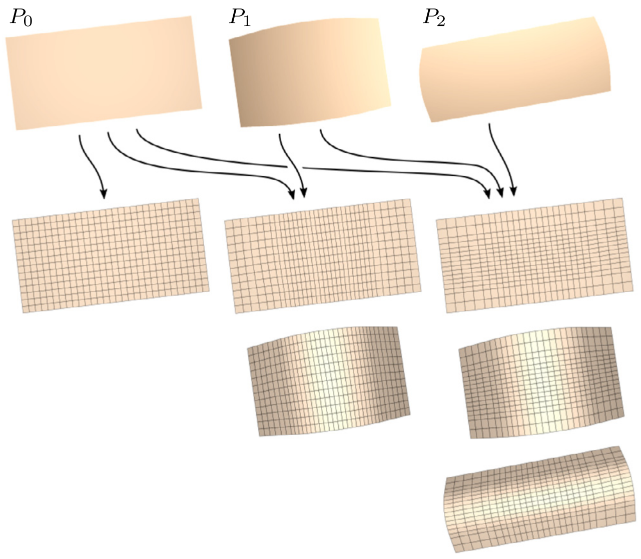

Fig. 6. The effect of the extremal metric $g_{\max}$ on a simple plane model, bent into different poses. Depending on whether only pose P_0 (left), poses P_0 and P_1 (middle), or poses P_0, P_1, and P_2 (right) are taken as input, the resulting quad mesh (shown deformed into the different poses below) will have its element sizing and anisotropy adapted accordingly. A metric based on shape operator S was used here, so element shapes are adapted to curvature: the most extreme curvature encountered locally in any of the poses determines the local element shape and size.

4.1. Metric consolidation

We can "transfer" the metric g_j from P_j to P_i as the pullback $h^*_{ij}g_j$ of g_j by h_{ij}. Let J_{ij} be the Jacobian of h_{ij}, then $h^*_{ij}g_j = J^{\mathsf{T}}_{ij} g_j \, J_{ij}$. In this way we can, in particular, obtain the set of N metric tensors $g_{0i} := h^*_{0i} g_i$ on P_0 (where $g_{00} := g_0$), cf. Fig. 5. Note that computing a quad mesh on P_0 based on metric g_{0i} and mapping the result to P_i via h_{0i} would yield the same result as performing the computation on P_i based on metric g_i (in a hypothetical exact setting, i.e. up to numerical or discretization effects). We are thus now able to perform computations targeted at specific poses P_i on a common computational domain; w.l.o.g. we choose P_0.

Our goal is to compute a quad mesh that is not optimized for a specific pose, but for all poses. We thus create a new metric on P_0 that takes all the metrics g_{0i} into account. A simple choice is the *average metric*

$$g_{\text{avg}} = \frac{1}{N} \sum_{i=0}^{N-1} g_{0i} \tag{1}$$

defined on P_0. This is akin to the average (or sum) error functions used in pose-aware triangle mesh decimation techniques (cf. Section 2). The problem with this averaging is that it implicitly weights local deformation states by how common they are in the pose set. For instance, consider a large set of poses of a character model where each pose is derived from a rest pose by exercising one joint to one of its extremes. In this case the average metric does hardly differ from the metric of the rest pose, because every surface region is in its rest state in all but one of the many poses.

Extremal metric We thus propose the use of an *extremal metric* rather than an average metric. For the above example this means that the metric locally reflects the surface stretch experienced in the poses that deform a region the most, rather than the rest pose (cf. Figs. 5 and 6).

One needs to decide whether the metric should be extremal in a maximum or a minimum sense; intuitively, whether it should be sensitive to the largest stretching or the largest squeezing of the surface. For the purpose of meshing, the former is relevant, because stretching leads to undersampling, whereas squeezing only leads to unnecessarily fine sampling (which is only an issue if mesh simplicity is of highest priority). We thus define

$$g'_{\max}(v, v) = \max_{i=0}^{N-1} g_{0i}(v, v) \tag{2}$$

for any tangent vector v. Notice that $g'_{\max}$ defined in this way is not a quadratic form, i.e. not a Riemannian metric, in general. We thus consider instead a tight bounding quadratic form $g_{\max}$ defined via

$$\det(g_{\max}) \to \min \ \text{ s.t. } \ g_{\max}(v, v) \geq g'_{\max}(v, v) \text{ for all } v. \tag{3}$$

4.2. Implementation details

The poses P_i are given as triangle meshes. We compute the necessary metric-related values per triangle or vertex of P_0 as follows.

Computing the Jacobians J_{0i} Let each triangle in each of the poses be equipped with a local 2D coordinate system. Let a, b, c be the coordinates of the three vertices of a triangle in P_0. The Jacobian of h_{0i} in this triangle is then computed as

$$J_{0i} = \begin{bmatrix} h_{0i}(b) - h_{0i}(a) & h_{0i}(c) - h_{0i}(a) \end{bmatrix} \begin{bmatrix} b - a & c - a \end{bmatrix}^{-1}.$$

Depending on the mesh generation technique to be used, a metric defined on a per-triangle or per-vertex basis might be required. In the per-vertex case, per-vertex Jacobians can be interpolated from those of the incident triangles (after rotations about the respective axes $n_t \times n_v$, aligning a triangle tangent plane, with normal n_t, to the vertex tangent plane, with normal n_v).

Computing the shape operator S We estimate principal curvatures $\kappa_{\min,i}$, $\kappa_{\max,i}$ and their directions $d_{\min,i}$, $d_{\max,i}$ on a pose P_i (Cazals and Pouget, 2003) (again with respect to local coordinate systems). Then $S_i = \kappa_{\min,i} d_{\min,i} d_{\min,i}^{\mathsf{T}} + \kappa_{\max,i} d_{\max,i} d_{\max,i}^{\mathsf{T}}$.

Computing the extremal metric g_{max} While it is long known that the optimization problem (3) has a unique solution (Behrend, 1938; John, 1948), no simple technique to find it if $N > 2$ is available. Even if we just require the inequality to hold for a sample of directions v, non-trivial iterative solution techniques are necessary (Kumar and Yildirim, 2005). As we need to perform this for every single triangle, we use an approximation instead:

Each metric g_{0i} can be written as $g_{0i} = \lambda_{ai} d_{ai} d_{ai}^{\mathsf{T}} + \lambda_{bi} d_{bi} d_{bi}^{\mathsf{T}}$, where λ_{ai}, λ_{bi} are the eigenvalues, and d_{ai}, d_{bi} the eigenvectors of g_{0i}. We construct $g_{\max} = \lambda_a d_a d_a^{\mathsf{T}} + \lambda_b d_b d_b^{\mathsf{T}}$, where the eigenvectors d_a and d_b are taken from g_{avg}, and the eigenvalues λ_a, λ_b are computed as:

$$\lambda_{a/b} = \max_{i=0}^{N-1} \ \lambda_{ai} (d_{a/b}^{\mathsf{T}} d_{ai})^2 + \lambda_{bi} (d_{a/b}^{\mathsf{T}} d_{bi})^2. \tag{4}$$

A simple calculation shows that we then have

$$g_{\max}(d_{a/b}, d_{a/b}) = g'_{\max}(d_{a/b}, d_{a/b}) = \max_{i=0}^{N-1} g_{0i}(d_{a/b}, d_{a/b}),$$

i.e. the approximation is exact in the principal directions d_a and d_b. One can furthermore show (cf. Appendix A) that for other directions v, the approximation error is bounded: $g'_{\max}(v, v)/g_{\max}(v, v) < 2$, i.e. the underestimation of lengths is bounded everywhere and in any direction by a factor $\sqrt{2}$. The approximation can thus easily be made conservative by scaling with a corresponding factor. This global scaling, of course, is without effect if the quadrangulation is performed with respect to a target mesh complexity rather than a specific target sizing, and might as well be omitted.

5. Element orientation

Often, orienting quad elements such that edges follow principal curvature directions, at least in significantly curved regions, is of interest (Bommes et al., 2009; Campen et al., 2016). Alternatively, the user may want to specify the orientation manually, usually in a sparse manner.

We can model such directional guidance objectives in a general way as follows. Let d_i be a tangent vector field on P_i whose direction specifies the desired edge orientation (e.g. the minimum principal curvature direction or another direction set by the user) and whose magnitude specifies the importance (e.g. it may be related to the local anisotropy of the surface, or be 0 in regions where there is no preference in terms of element orientation).

Let $d_{0i} := J_{i0} d_i$ be the pullback of d_i by h_{0i} on P_0. Note that these directions can obviously be contradictory: d_{0i} may differ from d_{0j} for $i \neq j$. This, for instance, is the case if d_i and d_j are principal curvature directions and the surface is bent differently in poses P_i and P_j. In such cases we strive to align to these directions as well as possible by striking a balance.

To this end, we compute a weighted average of the directions specified by d_{0i}, where the weights are dictated by the magnitudes $\|d_{0i}\|$. Here it is important to note that a cross field as well as a regular quad mesh, away from extraordinary vertices, has a 4-symmetric structure; whether we enforce element orientation according to a direction d, $d^{\perp}$, $-d$, or $-d^{\perp}$ is irrelevant. Here $d^{\perp}$ denotes a $\frac{\pi}{2}$-rotated (in the tangent plane) version of d. This invariance to rotations by multiples of $\frac{\pi}{2}$ must be considered to obtain a proper average. For instance, averaging d and $d^{\perp}$ should not result in (some multiple of) $d + d^{\perp}$, but in d (or any $k\frac{\pi}{2}$-rotation thereof, $k \in \mathbb{Z}$).

To achieve this invariance, we map the vector fields d_{0i} to 4th-order tensor fields (Palacios and Zhang, 2007) which are inherently invariant to rotations by multiples of $\frac{\pi}{2}$. In form of the so-called representation vector (Palacios and Zhang, 2007), they can be averaged by simple vector addition:

$$\bar{d}_{\text{avg}} := \frac{1}{N} \sum_{i}^{N-1} \bar{d}_{0i},$$

where $\bar{d}$ denotes the 4th-order tensor representation of d. The result is then mapped back to a vector field d_{avg}. Note that this, due to the involvement of the magnitude of d_{0i}, is a weighted average. A very similar $\frac{\pi}{2}$-symmetric averaging operation is performed in Marcias et al. (2013); some difference lies in that work using as final weight (i.e. magnitude of d_{avg}) the (weighted) average of the magnitudes of the d_{0i}, regardless of how compatible or contradictory the individual directions are. With the formulation here, the weights (encoded in the magnitude of the $\bar{d}_{0i}$) are attenuated depending on how strongly the individual directions differ.

5.1. Implementation details

Curvature guidance Some binary feature filters have been proposed in the past (Bommes et al., 2009; Nieser et al., 2012; Campen et al., 2016), that can be used to identify salient regions on meshes. We can apply them per pose P_i and set the directional guidance field d_i to a unit vector in minimum curvature direction in salient regions, and to zero everywhere else. Alternatively, a continuously weighted alignment term can be used by setting the local magnitude of d_i based on a curvedness measure on P_i, e.g., as in Kälberer et al. (2007), Ray et al. (2006), Knöppel et al. (2013).

Stretch guidance As proposed in Marcias et al. (2013), one can orient quad mesh elements according to the principal directions of stretch occurring when deforming P_0 to P_i. Here this corresponds to setting d_i to $(|\lambda_0|/|\lambda_1| - 1)s_0$, where $|\lambda_0| > |\lambda_1|$ are the singular values and s_0 is an eigenvector of J_{0i}. Such a choice of directional guidance puts a focus on element shear reduction.

Tensor conversion Let d be a tangent vector represented using coordinates in a local 2D tangent coordinate system. Using the standard identification of $\mathbb{R}^2$ and $\mathbb{C}$, we can write $d = r\mathrm{e}^{\mathrm{i}\theta}$, which is the complex representation of d. The representation vector of the 4th-order tensor then simply is $\bar{d} = r\mathrm{e}^{\mathrm{i}4\theta}$.

6. Mesh generation

Having constructed a metric g (with the case of particular interest here being $g = g_{\max}$) and a guidance field $d = d_{\text{avg}}$, both on $P = P_0$, we strive to generate a quad mesh that is as uniform as possible in g (i.e. with unit length edges and right face angles, measured in g), while edges tend to align to d. The meshing approaches based on cross field guided parametrization (cf. Section 2) are particularly suited for this task. There are two main categories: methods based on *global* parametrization (e.g. Bommes et al., 2009; Kälberer et al., 2007) or *local* parametrization (e.g. Jakob et al., 2015; Ray et al., 2006). The former approach generates a globally consistent integer grid map that implies a pure quad mesh, but requires rather involved measures to achieve robustness (Bommes et al., 2013a; Campen et al., 2015). The latter approach is simpler but generates a quad-dominant mesh only—which can be turned into a pure quad mesh by subdivision (introducing an additional irregular vertex for each non-quad) (Jakob et al., 2015).

Most of the meshing methods from these two categories were proposed in a formulation that assumes the Euclidean metric $g_{\text{Euclid}} = I$. Global parametrization methods can be adapted to a different metric g (in particular our $g_{\max}$) either through ‘metric uniformization by surface deformation’ as proposed by Panozzo et al. (2014), or through the use of g in the definition of parallel transport for the cross field generation and in the definition of norms in the parametrization energy.

The local parametrization based approach has the characteristic property of automatically introducing additional extraordinary vertices (besides those already imposed by the guiding cross field) where necessary to keep the mesh elements very close to their desired local size and shape. According to our experiments, this is particularly beneficial when the metric

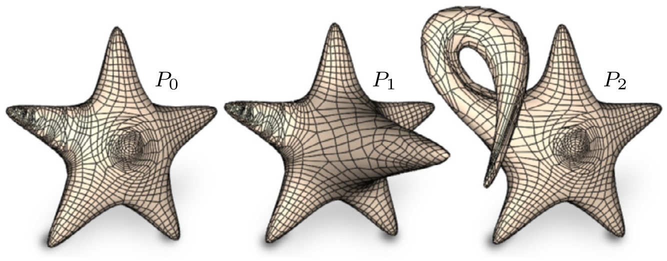

Fig. 7. The mesh of Fig. 1 bottom as generated when using Bommes et al. (2013a) instead of Jakob et al. (2015) with our technique. The number of non-regular vertices is lower, while quads are shaped worse.

varies significantly across the surface, e.g. due to strong deformations between poses. The global parametrization approach, by contrast, leads to a smaller number of extraordinary vertices in such cases, at the cost of higher element distortion (cf. Fig. 7). As the optimal trade-off is application dependent, we do not intend to advocate one approach over the other and note that either one can be used with our technique. As it more clearly conveys the effects of metric variations, we choose to use a local parametrization approach (Jakob et al., 2015) for demonstrations herein. We adapt it to custom metrics as detailed in the following.

6.1. Metric-aware local parametrization

Like most related methods, the recent local parametrization approach (Jakob et al., 2015) was described for g_{Euclid}. The authors state that it is straightforward to use custom metrics with this technique. While this is indeed true from a high level perspective, a number of non-trivial issues and details need to be addressed when taking a closer look on a lower level. We describe in the following how this method can be generalized to Riemannian metrics such as our $g_{\max}$. For brevity, we do not repeat the overall algorithm here but refer the reader to the original paper (Jakob et al., 2015).

Symmetry groups The method relies on the sets $\mathcal{R}$ and $\mathcal{T}$, which represent the cross field's rotational symmetry group and the integer grid's translational symmetry group, respectively. Under the Euclidean metric, the elements of $\mathcal{R}$ can be obtained from one representative element (unit vector) $\mathbf{o}$ via $\mathcal{R}(\mathbf{o}, \mathbf{n}, k) = \text{rot}(\mathbf{n}, k\frac{\pi}{2})\mathbf{o}$, where $\text{rot}(\mathbf{n}, \theta)$ is a rotation matrix around axis $\mathbf{n}$ by angle θ.

In our case, the four vectors of a metric-dependent cross are supposed to be in $\frac{\pi}{2}$ increments *with respect to* g. This is achieved using the definition $\mathcal{R}_g(\mathbf{o}, \mathbf{n}, k) = \sqrt{g}^{-1}\text{rot}(\mathbf{n}, k\frac{\pi}{2})\sqrt{g}\,\mathbf{o}$, where $\sqrt{g}$ is computed via singular value decomposition: $\sqrt{g} = U\sqrt{\Sigma}U^T$ where $U\Sigma U^T = g$. Note that if $\mathbf{o}$ is a unit vector *under* g, i.e. $\mathbf{o}^T g\,\mathbf{o} = 1$, so are the other resulting elements of $\mathcal{R}_g$. Hence $\mathcal{R}_g$ properly represents crosses that are orthonormal under g.

$\mathcal{T}_g$ in our case is supposed to represent a regular grid under g, i.e. a general oblique lattice in Euclidean space. It is simply spanned by the vectors of $\mathcal{R}_g$.

Metric transport In order to compare vectors from different tangent spaces, they must be transported to a common space. This is accomplished by a definition of parallel transport. A common choice in the discrete setting is *unfolding* via the rotation matrix $R_{ji} = \text{rot}(\mathbf{n}_j \times \mathbf{n}_i, \measuredangle(\mathbf{n}_j, \mathbf{n}_i))$, where $\mathbf{n}_i$ and $\mathbf{n}_j$ are the normals of two tangent spaces on P. The vector $\mathbf{o}_j$ from tangent space j transported to tangent space i then is computed as $\mathbf{o}_{ji} = R_{ji}\mathbf{o}_j$.

In our setting we also need to be able to transport metric tensors. We define g_{ji} (the tensor g_j from tangent space j transported to tangent space i) as $g_{ji} = R_{ij}^T g_j R_{ij}$, where $R_{ij} = R_{ji}^{-1}$. A simple calculation shows that $o_{ji}^T g_{ji}\, o'_{ji} = o_j^T g_j\, o'_j$ holds, i.e. this definition of metric transport is consistent with the above vector transport.

One situation where this metric transport is to be used is in the initial construction of a multiresolution hierarchy of P. When two vertices i and j are merged to a vertex v on a coarser level, a metric tensor for this vertex is interpolated; the tensors of i and j are transported to v and averaged: $g_v = \frac{1}{2}(g_{iv} + g_{jv})$. Further uses of metric transport occur in the following.

Period jumps When comparing and averaging crosses in neighboring tangent spaces i and j, the *period jump* k_{ij} needs to be determined. For orthogonal crosses this can be done as originally described: $k_{ij} = \text{argmin}_k \measuredangle(\mathbf{o}_i, \mathcal{R}(\mathbf{o}_{ji}, k))$, i.e. by just considering one pair of representative vectors. The result is independent of which vector $\mathbf{o}_i$ is used as representative for the cross at i.

With our metric g, the crosses are not orthogonal, as described above. The choice must thus be made differently in order to determine the period jump that best aligns the crosses, not just individual representative vectors. This is achieved by:

$$k_{ij} = \underset{k}{\text{argmin}} \sum_{0 \le l < 4} \measuredangle(\mathcal{R}_{g_i}(\mathbf{o}_i, l), \mathcal{R}_{g_{ji}}(\mathbf{o}_{ji}, l + k)).$$

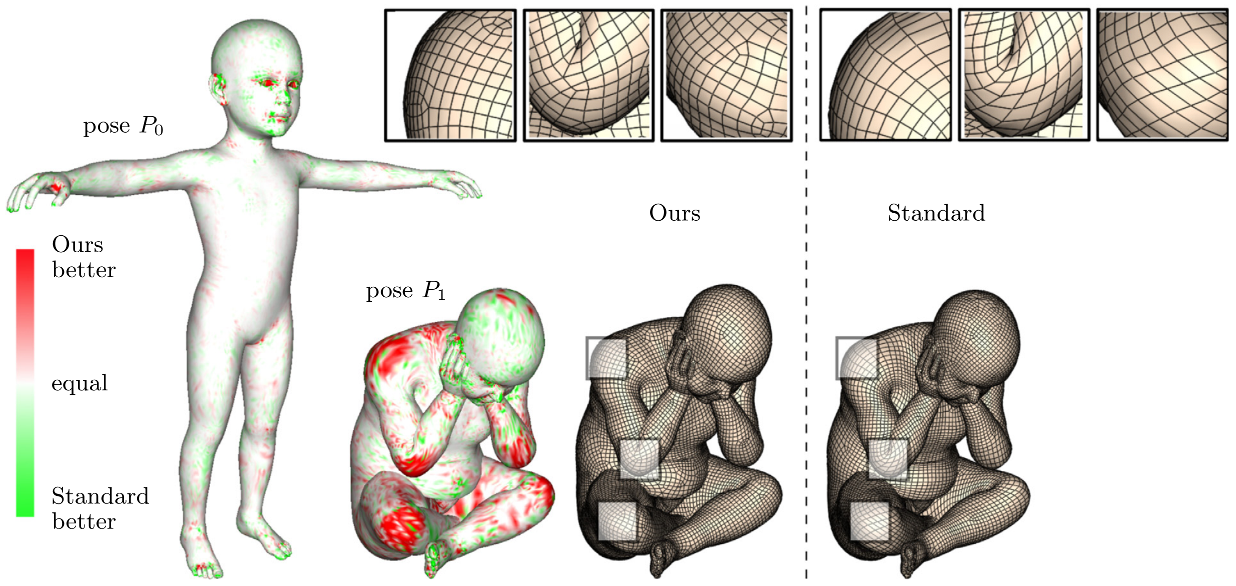

Fig. 8. Left: visualization of the *difference* in local shape approximation quality (pointwise input surface to quad mesh distance) when comparing our method (metric $g_{\max}$) and a standard, static method (effectively using metric g_0 only). In green regions, the mesh generated by the standard approach is better, in red regions, it is worse than our result. On P_0 the differences are minor, as expected. On P_1, the mesh optimized for P_0 using the standard, static approach (and then deformed to pose P_1) has significantly larger approximation error (red regions, especially on stretched and bent parts, such as knees, shoulders, elbows) than the mesh optimized over both poses using our method. Right: the two quad meshes in pose P_1; notice that long, stretched quad elements (in particular in direction of strong curvature) are avoided by our method, thereby reducing approximation error due to undersampling. It can also be observed how additional extraordinary vertices are introduced in order to facilitate the required mesh density transitions.

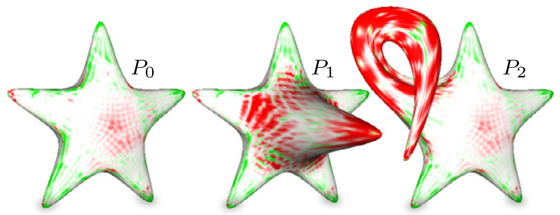

Fig. 9. Visualization of difference in local shape approximation quality (pointwise input surface to quad mesh distance) between our method, optimizing for all poses, and a method optimizing for a static reference pose (P_0) only. The corresponding quad meshes are depicted in Fig. 1. Color coding as in Fig. 8, i.e. in red regions our method is better.

In this way, the combined alignment of all four vector pairs is measured for each choice of period jump to determine the optimal one. By symmetry, the sum can actually be restricted to $0 \leq j < 2$ without changing the result.

Finally, it bears noting that the extrinsic smoothness measures which are demonstrated as being beneficial in Jakob et al. (2015) are not appropriate for our case of a deformable shape (or a shape in multiple poses): there is no static embedding based on which extrinsic measures could be evaluated. We make use of the intrinsic formulation instead, and use our field d as directional guiding input to this method. As suggested, one step of midpoint subdivision is performed on the resulting quad-dominant mesh in order to yield a quad-only mesh.

7. Results

When performing comparisons of different approaches based on different metrics g_a and g_b in the following, we rescale these metrics with a globally constant factor such that $\int_P \det g_a \, dA = \int_P \det g_b \, dA$. This leads to quad meshes of approximately equal complexity (number of elements), allowing for meaningful assessment of the improvements in terms of approximation quality.

Key pose sets In Figs. 8 and 9 we compare quad meshes optimized for a static geometry (pose P_0 of the input) with quad meshes optimized jointly for all given poses (two in Fig. 8, three in Fig. 9) with our method. The S-based metric was used in these examples. The local shape approximation quality in terms of pointwise input mesh to quad mesh distance is visualized over the surface.

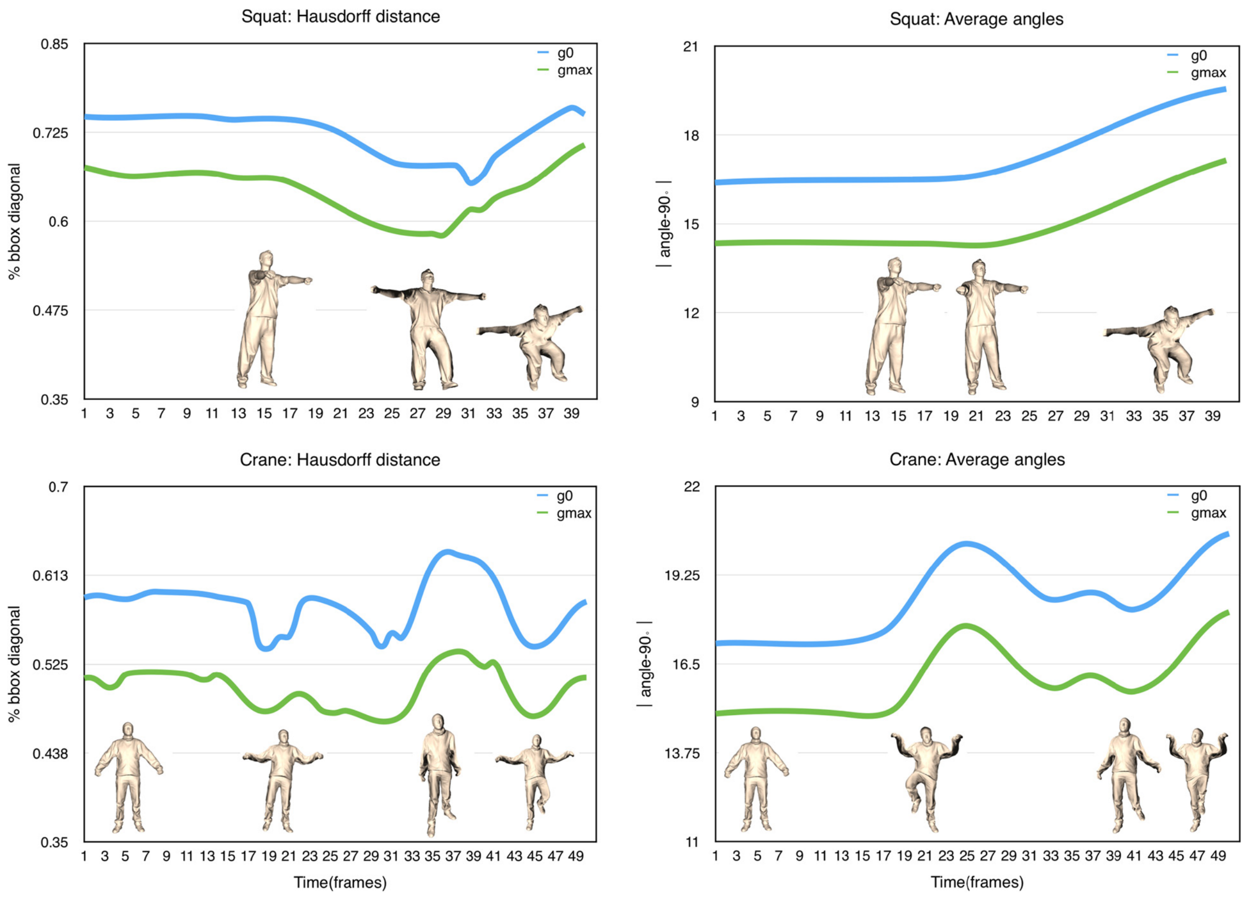

Fig. 10. The shape approximation quality (in terms of Hausdorff distance, left column) and element quality (in terms of quad corner angles, right column) of quad meshes during deformation according to given animation sequences (Squat, Crane), some frames of which are depicted below the graphs. We compare our method (green) with a standard approach (blue) that optimizes the mesh for frame 0 of the animation. See the text for more details on the algorithmic choices.

Animation sequences In Fig. 10 we consider the quality of quad meshes undergoing deformation according to given animation sequences. We compare meshes generated by our proposed method, optimized for the entire input animation sequences (using $g_{\max}$), with quad meshes generated for static pose P_0 (as with traditional quad mesh generation methods, using g_0). We used the S-based metric in this comparison, and made use of orientation fields d computed from the principal curvatures as described in Nieser et al. (2012). As can be observed, the Hausdorff distance from the input model to the quad mesh is lower when using our method, and the average element shape is better (element corner angles closer to 90°). Interestingly, in these examples this even holds for the frame P_0; we attribute this to the joint determination of the curvature-based orientation constraints over the entire sequence (cf. Section 5) in our method, which provides more stable, reliable guidance for the model. Possibly, the typically higher number of extraordinary vertices due to $g_{\max}$ also contributes to this effect.

Fig. 11 shows a similar comparison based on animation sequences. However, for this experiment we used our method in combination with the orientation guidance based on principal stretch directions proposed by Marcias et al. (2013), as detailed in Section 5.1. The further improvement due to the use of our $g_{\max}$ in addition to this animation-aware orientation guidance becomes clear in this experiment.

8. Limitations & future work

A high quality solution to the problem of determining suitable guidance fields d is desirable; those discussed in Section 5.1 are not always satisfactory. When working with a single static model, manual input (such as stroke guidance Ebke et al., 2016) can be reasonable, but for multiple poses and even sequences this is impractical; automatic solutions are important in this context.

Finding an exact solution to the tight bounding metric problem (3) would certainly be of value; though we expect the practical implications to be minor in the context at hand. On a related note, metrics that consider stretching *and* squeezing

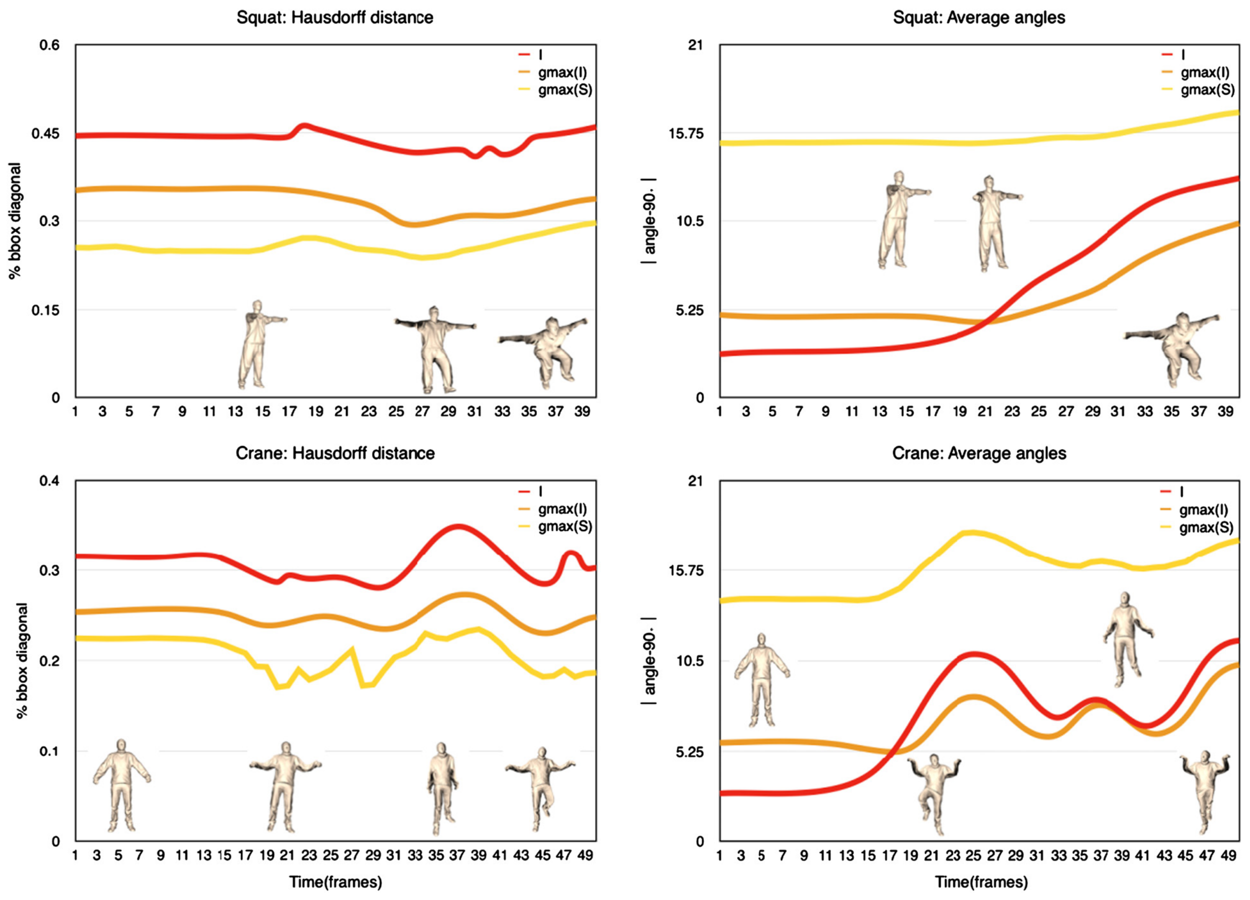

Fig. 11. The shape approximation quality (in terms of Hausdorff distance, left column) and element quality (in terms of quad corner angles, right column) of quad meshes during deformation according to given animation sequences (Squat, Crane), some frames of which are depicted. We compare meshes optimized for the standard Euclidean metric I of frame 1 (red graph) with meshes optimized for this metric extremally combined over all frames using our method (orange graph). It can be observed that in these examples the extremal metric reduces approximation error (left). As expected, the use of the S (shape-operator) metric in combination with our approach (yellow graph) leads to further error reduction. The element quality (in terms of corner angles) is best in the beginning of the sequences with the mesh optimized for frame 1 (red), while later the benefit of the extremal metric (orange) shows. The use of S (yellow) naturally leads to less square elements by design; it rather targets low approximation error. Note that the method leading to the red graph is conceptually equal to Marcias et al. (2013)—with the exception of the local technique (Jakob et al., 2015) rather than the global technique (Bommes et al., 2009) being used for mesh generation in the end.

could be of value in certain contexts; averages of $g_{\max}$ and an analogously defined $g_{\min}$ could provide benefits over a traditional g_{avg} (1) here.

As discussed in Section 6, the introduction of additional extraordinary vertices is of benefit for individual mesh element quality. Depending on the use case it is, however, likewise beneficial if the number of extraordinary vertices is small (Bommes et al., 2013b). For use cases that have a strong preference for meshes with a low number of extraordinary vertices, our method in combination with the chosen meshing strategy (Jakob et al., 2015) is likely not a viable choice, and so far no alternative automatic quad mesh generation method enables explicit and practical continuous control over the trade-off between element quality and extraordinary vertex count. Progress in this direction would thus certainly be of interest. Likewise no fully robust global parametrization based mesh generation technique (that does not introduce additional extraordinary vertices) is available, but would be of interest to reliably achieve low numbers of extraordinary vertices even with highly variable metrics.

It will be interesting to further explore the use of our method for different models of the same category (as opposed to different poses of the same model) to obtain high quality generic quad meshes for shape categories, essentially by a form of *co-quadrangulation*. Depending on the specific scenario, methods for non-rigid shape correspondence finding (van Kaick et al., 2011; Kim et al., 2011), including recent machine learning based variants (Boscaini et al., 2016), could play a central role in this for determining the diffeomorphisms between models.

Acknowledgements

This work was supported in part by: the Moore-Sloan Data Science Environment at NYU; NASA; DOE; NSF awards CNS-1229185, CCF-1533564, CNS-1544753, CNS-1730396; NSFC award 61332015. J. Zhou is supported by CSC scholarship No. 201506220093. C. T. Silva is also partially supported by the DARPA MEMEX and D3M programs. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of DARPA. The authors thank David Bommes for providing data for evaluation purposes.

Appendix A. Extremal metric approximation bound

For a given tangent vector v, let $i = \operatorname{argmax}_{i=0}^{N-1} g_{0i}(v, v)$. Let

$$g_{0i} = \mu_a e_a e_a^\intercal + \mu_b e_b e_b^\intercal, \quad g_{\max} = \lambda_a d_a d_a^\intercal + \lambda_b d_b d_b^\intercal$$

be expressions of the metrics in terms of their eigenvalues and eigenvectors. Then

$$g_{\max}(v, v) = (d_a^\intercal v)^2 \lambda_a + (d_b^\intercal v)^2 \lambda_b.$$

Noticing that $\lambda_{a/b} = g_{0i}(d_{a/b}, d_{a/b})$ (cf. Equation (4)), we obtain

$$\begin{aligned} g_{\max}(v, v) &= (d_a^\intercal v)^2 g_{0i}(d_a, d_a) + (d_b^\intercal v)^2 g_{0i}(d_b, d_b) \\ &= (d_a^\intercal v)^2 \left((e_a^\intercal d_a)^2 \mu_a + (e_b^\intercal d_a)^2 \mu_b\right) + (d_b^\intercal v)^2 \left((e_a^\intercal d_b)^2 \mu_a + (e_b^\intercal d_b)^2 \mu_b\right) \\ &= (e_a^\intercal d_a d_a^\intercal v)^2 \mu_a + (e_b^\intercal d_a d_a^\intercal v)^2 \mu_b + (e_a^\intercal d_b d_b^\intercal v)^2 \mu_a + (e_b^\intercal d_b d_b^\intercal v)^2 \mu_b \\ &= \left((e_a^\intercal d_a d_a^\intercal v)^2 + (e_a^\intercal d_b d_b^\intercal v)^2\right) \mu_a + \left((e_b^\intercal d_a d_a^\intercal v)^2 + (e_b^\intercal d_b d_b^\intercal v)^2\right) \mu_b. \end{aligned}$$

We compare this to $g'_{\max}(v, v) = g_{0i}(v, v) = (e_a^\intercal v)^2 \mu_a + (e_b^\intercal v)^2 \mu_b$, by considering the coefficients of μ_a and μ_b. In particular, we show in the following that these are related by

$$\left((e_a^\intercal d_a d_a^\intercal v)^2 + (e_a^\intercal d_b d_b^\intercal v)^2\right) \geq \frac{1}{2}(e_a^\intercal v)^2$$

(and analogously for the coefficient of μ_b). From this it immediately follows that $g'_{\max}(v, v)/g_{\max}(v, v) < 2$.

Due to orthonormality of d_a, d_b ($d_a^\intercal d_a = d_b^\intercal d_b = 1$ and $d_a^\intercal d_b = 0$), we have $e_a^\intercal d_a d_a^\intercal v + e_a^\intercal d_b d_b^\intercal v = e_a^\intercal (d_a d_a^\intercal + d_b d_b^\intercal) v = e_a^\intercal v$. Defining $p = e_a^\intercal d_a d_a^\intercal v$ and $q = e_a^\intercal d_b d_b^\intercal v$, we thus only need to show that $p^2 + q^2 \geq \frac{1}{2}(p+q)^2 = \frac{1}{2}(p^2 + q^2) + pq$ to prove the above inequality, or equivalently that $p^2 + q^2 \geq 2pq$. One easily verifies that this holds for any $p, q \in \mathbb{R}$.

References

Behrend, F., 1938. Über die kleinste umbeschriebene und die größte einbeschriebene Ellipse eines konvexen Bereichs. Math. Ann. 115 (1), 379–411.
Bommes, D., Zimmer, H., Kobbelt, L., 2009. Mixed-integer quadrangulation. ACM Trans. Graph. 28 (3), 77.
Bommes, D., Lempfer, T., Kobbelt, L., 2011. Global structure optimization of quadrilateral meshes. Comput. Graph. Forum 30 (2), 375–384.
Bommes, D., Campen, M., Ebke, H.-C., Alliez, P., Kobbelt, L., 2013a. Integer-grid maps for reliable quad meshing. ACM Trans. Graph. 32 (4), 98.
Bommes, D., Lévy, B., Pietroni, N., Puppo, E., Silva, C., Tarini, M., Zorin, D., 2013b. Quad-mesh generation and processing: a survey. Comput. Graph. Forum 32 (6), 51–76.
Boscaini, D., Masci, J., Rodolà, E., Bronstein, M., 2016. Learning shape correspondence with anisotropic convolutional neural networks. In: Advances in Neural Information Processing Systems 29, pp. 3189–3197.
Campen, M., Zorin, D., 2017. Similarity maps and field-guided T-splines: a perfect couple. ACM Trans. Graph. 36 (4).
Campen, M., Bommes, D., Kobbelt, L., 2015. Quantized global parametrization. ACM Trans. Graph. 34 (6).
Campen, M., Ibing, M., Ebke, H.-C., Zorin, D., Kobbelt, L., 2016. Scale-invariant directional alignment of surface parametrizations. Comput. Graph. Forum 35 (5).
Cazals, F., Pouget, M., 2003. Estimating differential quantities using polynomial fitting of osculating jets. In: Proc. SGP '03, pp. 177–187.
Crane, K., Desbrun, M., Schröder, P., 2010. Trivial connections on discrete surfaces. Comput. Graph. Forum 29 (5).
DeCoro, C., Rusinkiewicz, S., 2005. Pose-independent simplification of articulated meshes. In: SI3D. ACM, pp. 17–24.
Diamanti, O., Vaxman, A., Panozzo, D., Sorkine-Hornung, O., 2014. Designing N-PolyVector fields with complex polynomials. Comput. Graph. Forum 33 (5), 1–11.
Ebke, H.-C., Campen, M., Bommes, D., Kobbelt, L., 2014. Level-of-detail quad meshing. ACM Trans. Graph. 33 (6).
Ebke, H.-C., Schmidt, P., Campen, M., Kobbelt, L., 2016. Interactively controlled quad remeshing of high resolution 3d models. ACM Trans. Graph. 35 (6).
Frankel, T., 2011. The Geometry of Physics: An Introduction. Cambridge University Press.
Heckbert, P.S., Garland, M., 1999. Optimal triangulation and quadric-based surface simplification. Comput. Geom. 14 (1–3), 49–65.
Hertzmann, A., Zorin, D., 2000. Illustrating smooth surfaces. In: Proc. SIGGRAPH '00, pp. 517–526.
Huang, F.-C., Chen, B.-Y., Chuang, Y.-Y., 2006. Progressive deforming meshes based on deformation oriented decimation and dynamic connectivity updating. In: Proc. SCA '06, pp. 53–62.
Jakob, W., Tarini, M., Panozzo, D., Sorkine-Hornung, O., 2015. Instant field-aligned meshes. ACM Trans. Graph. 34 (6), 189.
Jiang, T., Fang, X., Huang, J., Bao, H., Tong, Y., Desbrun, M., 2015. Frame field generation through metric customization. ACM Trans. Graph. 34 (4), 40.
John, F., 1948. Extremum problems with inequalities as subsidiary conditions. In: Studies and Essays Presented to R. Courant on His 60th Birthday. Interscience Publishers, New York, pp. 187–204.

Kälberer, F., Nieser, M., Polthier, K., 2007. Quadcover—surface parameterization using branched coverings. Comput. Graph. Forum 26 (3), 375–384.
Kim, V.G., Lipman, Y., Funkhouser, T., 2011. Blended intrinsic maps. ACM Trans. Graph. 30 (4).
Kircher, S., Garland, M., 2005. Progressive multiresolution meshes for deforming surfaces. In: Proc. SCA '05, pp. 191–200.
Knöppel, F., Crane, K., Pinkall, U., Schröder, P., 2013. Globally optimal direction fields. ACM Trans. Graph. 32 (4).
Kovacs, D., Myles, A., Zorin, D., 2011. Anisotropic quadrangulation. Comput. Aided Geom. Des. 28 (8).
Kumar, P., Yildirim, E.A., 2005. Minimum-volume enclosing ellipsoids and core sets. J. Optim. Theory Appl. 126 (1), 1–21.
Landreneau, E., Schaefer, S., 2009. Simplification of articulated meshes. Comput. Graph. Forum 28 (2), 347–353.
Lee, H., Ahn, M., Lee, S., 2011. Displaced subdivision surfaces of animated meshes. Computers & Graphics 35 (3), 532–541.
Li, W.-C., Vallet, B., Ray, N., Lévy, B., 2006. Representing higher-order singularities in vector fields on piecewise linear surfaces. IEEE Trans. Vis. Comput. Graph. 12 (5), 1315–1322.
Liu, Y., Xu, W., Wang, J., Zhu, L., Guo, B., Chen, F., Wang, G., 2011. General planar quadrilateral mesh design using conjugate direction field. ACM Trans. Graph. 30 (6).
Marcias, G., Pietroni, N., Panozzo, D., Puppo, E., Sorkine-Hornung, O., 2013. Animation-aware quadrangulation. Comput. Graph. Forum 32 (5), 167–175.
Marcias, G., Takayama, K., Pietroni, N., Panozzo, D., Sorkine-Hornung, O., Puppo, E., Cignoni, P., 2015. Data-driven interactive quadrangulation. ACM Trans. Graph. 34 (4), 65.
Meng, M., He, Y., 2016. Consistent quadrangulation for shape collections via feature line co-extraction. Comput. Aided Des. 70, 78–88.
Mohr, A., Gleicher, M., 2003. Deformation Sensitive Decimation. Technical report. University of Wisconsin.
Myles, A., Zorin, D., 2013. Controlled-distortion constrained global parametrization. ACM Trans. Graph. 32 (4).
Nadler, E., 1986. Piecewise linear best L_2 approximation on triangulations. Approx. Theory 499–502.
Nieser, M., Palacios, J., Polthier, K., Zhang, E., 2012. Hexagonal global parameterization of arbitrary surfaces. IEEE Trans. Vis. Comput. Graph. 18 (6).
Pal, K., Schüller, C., Panozzo, D., Sorkine-Hornung, O., Weyrich, T., 2014. Content-aware surface parameterization for interactive restoration of historical documents. Comput. Graph. Forum 33 (2), 401–409.
Palacios, J., Zhang, E., 2007. Rotational symmetry field design on surfaces. ACM Trans. Graph. 26 (3).
Panozzo, D., Puppo, E., Tarini, M., Sorkine-Hornung, O., 2014. Frame fields: anisotropic and non-orthogonal cross fields. ACM Trans. Graph. 33 (4), 134.
Payan, F., Hahmann, S., Bonneau, G.P., 2007. Deforming surface simplification based on dynamic geometry sampling. In: Proc. Shape Modeling and Applications. SMI '07, pp. 71–80.
Peng, C.-H., Zhang, E., Kobayashi, Y., Wonka, P., 2011. Connectivity editing for quadrilateral meshes. ACM Trans. Graph. 30 (6).
Pietroni, N., Tarini, M., Sorkine, O., Zorin, D., 2011. Global parametrization of range image sets. ACM Trans. Graph. 30 (6).
Ray, N., Li, W.C., Lévy, B., Sheffer, A., Alliez, P., 2006. Periodic global parameterization. ACM Trans. Graph. 25 (4), 1460–1485.
Ray, N., Vallet, B., Li, W.C., Lévy, B., 2008. N-symmetry direction field design. ACM Trans. Graph. 27 (2), 10.
Ray, N., Vallet, B., Alonso, L., Levy, B., 2009. Geometry-aware direction field processing. ACM Trans. Graph. 29 (1).
Takayama, K., Panozzo, D., Sorkine-Hornung, A., Sorkine-Hornung, O., 2013. Sketch-based generation and editing of quad meshes. ACM Trans. Graph. 32 (4), 97.
Tarini, M., Pietroni, N., Cignoni, P., Panozzo, D., Puppo, E., 2010. Practical quad mesh simplification. Comput. Graph. Forum 29 (2), 407–418.
van Kaick, O., Zhang, H., Hamarneh, G., Cohen-Or, D., 2011. A survey on shape correspondence. Comput. Graph. Forum 30 (6), 1681–1707.
Vaxman, A., Campen, M., Diamanti, O., Panozzo, D., Bommes, D., Hildebrandt, K., Ben-Chen, M., 2016. Directional field synthesis, design, and processing. Comput. Graph. Forum 35 (2).