Green Coordinates for Triquad Cages in 3D

JEAN-MARC THIERY, Adobe Research, France
TAMY BOUBEKEUR, Adobe Research, France

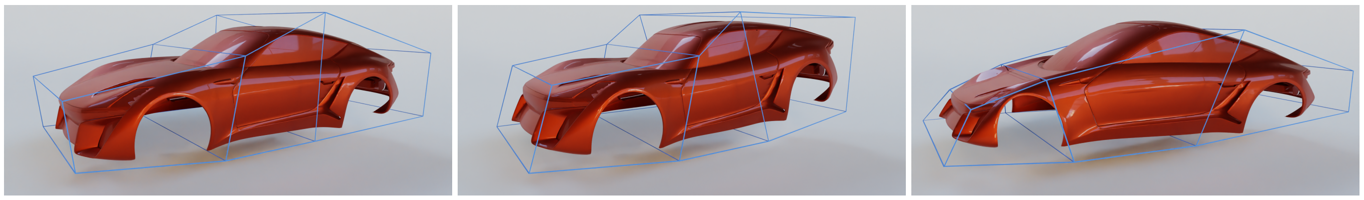


Fig. 1. Our coordinates encode shapes (left, red) into (tri)quad cages (left, blue), and produce quasi-conformal deformations when manipulating the cage (middle and right), avoiding cage triangulation — and its many artifacts — while enabling Green coordinates deformation for quad-based design workflows.

We introduce Green coordinates for triquad cages in 3D. Based on Green's third identity, Green coordinates allow defining the harmonic deformation of a 3D point inside a cage as a linear combination of its vertices and face normals. Using appropriate Neumann boundary conditions, the resulting deformations are quasi-conformal in 3D, and thus best-preserve the local deformed geometry, in that volumetric conformal 3D deformations do not exist unless rigid. Most coordinate systems use cages made of triangles, yet quads are in general favored by artists as those align naturally onto important geometric features of the 3D shapes, such as the limbs of a character, without introducing arbitrary asymmetric deformations and representation. While triangle cages admit per-face constant normals and result in a single Green normal-coordinate per triangle, the case of quad cages is at the same time more involved (as the normal varies along non-planar quads) and more flexible (as many different mathematical models allow defining the smooth geometry of a quad interpolating its four edges). We consider bilinear quads, and we introduce a new Neumann boundary condition resulting in a simple set of four additional normal-coordinates per quad. Our coordinates remain quasi-conformal in 3D, and we demonstrate their superior behavior under non-trivial deformations of realistic triquad cages.

ACM Reference Format:
Jean-Marc Thiery and Tamy Boubekeur. 2022. Green Coordinates for Triquad Cages in 3D. 1, 1 (September 2022), 7 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Cage coordinates express space as a linear combination of the elements of a cage. In 3D, this cage takes the form of a manifold surface mesh, and any modification to its embedding yields a space deformation. A popular use of such coordinates is therefore to encode the geometry of a high-resolution shape, possibly made of multiple components and not necessary manifold, in the cage coordinate system defined by a coarse bounding cage mesh, using the latter to perform high-resolution freeform deformation by simply moving its vertices. A number of cage coordinate systems have been proposed in the literature, many of which aiming at generalizing barycentric coordinates to non-simplicial domains. Among them, and in the specific application scenario of freeform deformation, Green coordinates [Lipman et al. 2008] stand as a salient example: coordinates being expressed w.r.t. to both cage vertex positions and cage face normals, the deformations induced by such a system are locally quasi-conformal and turn out, in practice, to produce extremely pleasing deformations. Unfortunately, as with most cage coordinate systems, Green coordinates are defined for triangle meshes only, while the vast majority of hand-crafted meshes come in either quad or triquad format, which typically capture the local geometric anisotropy more efficiently and are more natural to manipulate.

We introduce an evolution of Green coordinates designed for triquad cages. The main challenge in this context relates to quads, which cannot be assumed to be planar in practice. We show that expressing Green coordinates on a per-face-corner basis and computing them through a robust Riemann summation scheme provides a freeform deformation framework that enjoys all the good properties of Green coordinates, while being compatible with the de facto standard of quad-dominant meshes. Our experiments against Quad Mean Value Coordinates and Triangle Green Coordinates clearly show the superiority of our approach for a variety of modeling tasks, at a moderate additional computational cost.

1.1 Related Work

Prior art on cage coordinates and free-form deformation in general being extremely vast, we refer the interested reader to recent surveys [Floater 2015; Hormann and Sukumar 2017; Jacobson et al. 2014] and focus here on the most relevant cage-based 3D deformation methods.

Barycentric coordinates w.r.t. triangle cages. The vast majority of existing cage coordinates allow defining barycentric weights $\{w_i\}$ for a 3D point η w.r.t. the vertices $\{v_i\}$ of a cage coming in the form of a closed manifold triangle mesh i.e., $\eta = \sum_i w_i v_i$. Mean-Value Coordinates [Floater 2003] (MVC) rely on the mean-value theorem and allow defining weights for points everywhere in 3D [Ju et al.

Authors' addresses: Jean-Marc ThieryAdobe Research, France, jthiery@adobe.com; Tamy BoubekeurAdobe Research, France, boubek@adobe.com.

2005] — not just the interior of the cage — which makes them convenient for multiresolution modeling [Borosán et al. 2010; Huang et al. 2006]. Coming with a closed-form expression, these coordinates admit known formulas for their gradients and Hessians [Thiery et al. 2014], which makes them a good subspace candidate for variational optimizations. Harmonic coordinates [Joshi et al. 2007] (HC) rely on a volumetric grid to diffuse the cage vertex basis functions inside the cage. Contrary to MVC, those are defined inside the cage only and do not come with a closed-form expression, but are however positive by constraint, which is a good property for cage-based deformations e.g., translating parts of the cage in a given direction does not result in parts of the model being translated in the opposite direction. An important drawback of these coordinates is that each coordinate has to be solved for the entire domain at the same time. In that sense, they cannot be computed easily on a per-point basis, and inserting a new mesh vertex requires computing every coordinate from scratch. Positive Mean-value coordinates [Lipman et al. 2007] address this issue, and extend MVC by considering the cage visibility function to ensure positive coordinates. The proposed implementation uses the GPU to render the cage from the point of view of the evaluation point, with visibility being simply determined using the z-buffer.

Polygonal cages. As triangles are rather limiting for shape modeling applications, and quads are generally preferred by 3D artists, a few coordinates were extended to comply with polygonal cages. Spherical barycentric coordinates [Langer et al. 2006] extend MVC and allow using cages made of arbitrary planar convex n-gons. Mean-Value coordinates for Quad cages [Thiery et al. 2018] (QMVC) extend MVC as well, and cope with cages made of triangles and non-planar quads — to a certain extent, as some extreme configurations are not handled by this method. As shown in these two papers, the resulting mesh deformations are then free of the artifacts which typically arise from arbitrary triangulation of the cage quads.

Green coordinates. Vastly different from all previous coordinate systems in spirit, Green coordinates [Lipman et al. 2008] do not target interpolating deformations, and allow expressing a 3D point as a harmonic linear combination of the cage vertices as well as the cage triangle normals. This translates into non-trivial deformations — rotations are inferred by cage vertex translations, thanks to the use of the cage normals — and quasi-conformal behavior in 3D. Their good volume deformation properties make Green coordinates an ideal choice for subspace variational deformations, as demonstrated by Ben-Chen et al. [Ben-Chen et al. 2009], who use them to obtain "almost" cage-independent as-rigid-as-possible deformations by constraining their derivatives. We aim at providing Green coordinates for cages made of triangles and non-planar quads, offering coordinates that are to GC what QMVC are to MVC.

1.2 Contributions

We propose Green coordinates for triquad cages and make the following technical contributions:

(1) We introduce a formulation of the deformation using explicitly the on-quad-varying normal, resulting in superior deformation behavior under non-trivial cage deformations (large stretch and shear, twist). Our formulation results in per-quad-corner dedicated coordinates.
(2) We show how to bake the validity conditions (linear precision) directly in the computations, by introducing tessellation-independent geometric invariants.
(3) We introduce an efficient tessellation-based adaptive Riemann summation allowing for fast approximate, linearly-precise, smooth computations of our coordinates.

2 BACKGROUND: GREEN COORDINATES FOR TRIANGULAR CAGES IN 3D

Using Green's third identity, we can express a harmonic function in a bounded 3D domain Ω from its boundary conditions as

$$f(\eta) = \int_{\xi \in \partial\Omega} f(\xi) \frac{\partial G}{\partial n}(\xi, \eta) d\xi - \int_{\xi \in \partial\Omega} G(\xi, \eta) \frac{\partial f}{\partial n}(\xi) d\xi, \quad (1)$$

with $G(\xi, \eta) := \frac{-1}{4\pi \|\xi - \eta\|}$ solution to $\triangle_\xi G(\xi, \eta) = \delta_0(\|\xi - \eta\|)$.

We consider here the cage $\partial\Omega$ as a non-intersecting closed manifold triangle mesh; we note deformed quantities with an apostrophe and rest-pose quantities without it.

Lipman et al. proposed Green coordinates [Lipman et al. 2008] for triangle cages, by setting the following Dirichlet and Neumann conditions on the cage $\partial\Omega$:

$$f(\xi) = \sum_i \Gamma^i(\xi) v'_i \quad (2)$$

$$\frac{\partial f}{\partial n}(\xi) = \sigma_j n'_j \; \forall \xi \in t_j, \quad (3)$$

Γ^i being the "hat basis function" that takes value 1 on vertex i, 0 at the other vertices and is linear on each triangle (in particular, its support is the set of faces adjacent to vertex i, noted $F_1(i)$), σ_j (resp. n'_j) being the conformality factor (resp. normal) of linearly-deformed triangle t_j, both quantities being constant across t_j as they depend on the (constant) triangle linear map only.

While the Dirichlet condition is rather natural — the triangles' geometry is obtained from linear interpolation using $\{\Gamma^i\}_i$ — the Neumann condition is arbitrary and set to obtain empirically appropriate deformations. Noting (e_1, e_2) (resp. (e'_1, e'_2)) two rest pose (resp. deformed) edges of triangle t_j, σ_j can be computed as

$$\sigma_j = \sqrt{\frac{\|e'_1\|^2\|e_2\|^2 + \|e_1\|^2\|e'_2\|^2 - 2(e'_1 \cdot e'_2)(e_1 \cdot e_2)}{2\|e_1 \times e_2\|^2}} \quad (4)$$

Setting these boundary conditions results in the following compact expression for f:

$$f(\eta) = \sum_{i \in \mathcal{V}} \phi_i(\eta) v'_i + \sum_j \psi_j(\eta) \sigma_j n'_j \text{ , with} \quad (5)$$

$$\phi_i(\eta) := \int_{\xi \in F_1(i)} \Gamma^i(\xi) \frac{\partial G}{\partial n}(\xi, \eta) d\xi \quad (6)$$

$$\psi_j(\eta) := \int_{\xi \in t_j} -G(\xi, \eta) d\xi \quad (7)$$

Note that these two conditions are *conflicting each other* — harmonic functions are uniquely defined by either condition, and f

does not respect any of the two conditions in the end (if it did, Green coordinates would then be extremely similar to Harmonic coordinates [Joshi et al. 2007]). As noted in [Lipman et al. 2008], the use of this Neumann condition ensures scale invariance and results in practice in quasi-conformal 3D spatial deformations, as observed experimentally.

Note however that this formulation comes with a baked-in ill behavior for degenerate faces: while the unit normal is formally not defined for zero area triangles, σ_j may be non-zero for those, resulting in an ill-defined behavior in this (highly unconventional) case (consider for example $e'_1 = 0$, $e'_2 \neq 0$ in Eq. (4)).

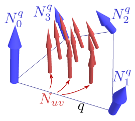
3 GREEN COORDINATES FOR QUAD CAGES

Our approach consists in approximating coordinates propagating in space an ideal quad deformation model. Our approximation is smooth and robust, inducing 4 extra coordinates per quad — one per quad corner — while coping naturally with the presence of triangles in the cage. At the core of our method, a robust Riemann summation scheme is proposed in the form of an adaptive triangulation of the quad uv domain which we designed in an output-sensitive way i.e., depending on the evaluation position. Last, we show how to recover the coarse deformation behavior induced by the original Neumann conditions of the triangle GC, resulting in smooth symmetric deformations that closely follow the limit case observed under limit refinement of the bilinear quads.

3.1 Quad deformation model

We consider the case of bilinear quads, which are good candidates for intuitive cage-based modeling, as supported in [Thiery et al. 2018]. Noting a quad q with corners $(q_0, q_1, q_2, q_3) \in \mathbb{R}^{3\times4}$, its bilinear sheet is given by $\sum_{k=0}^{3} b^k_{uv} q_k$ with $(b^0_{uv}, b^1_{uv}, b^2_{uv}, b^3_{uv}) = ((1-u)(1-v), u(1-v), uv, (1-u)v) \in \mathbb{R}^4$ *the bilinear coordinates* at parameter $(u,v) \in \mathbb{R}^2$, and its bilinear quad is its restriction to $(u,v) \in [0,1]^2$. Given this parameterization, the tangent vectors $\partial_x q_{uv}$, normal n_{uv} and surface element dq_{uv} can be obtained as

$$\begin{cases} \partial_u q_{uv} & = (1-v)(q_1 - q_0) + v(q_2 - q_3) \\ \partial_v q_{uv} & = (1-u)(q_3 - q_0) + u(q_2 - q_1) \\ N_{uv} & := \partial_u q_{uv} \times \partial_v q_{uv} = \sum_{k=0}^{3} b^k_{uv} N^q_k \\ n_{uv} & = N_{uv}/\|N_{uv}\| \\ dq_{uv} & = \|N_{uv}\| du dv. \end{cases} \tag{8}$$



with $N^q_k := (q_{k+1} - q_k) \times (q_{k+3} - q_k)$ (indices being taken as modulo 4) being the *unnormalized normal at corner* k of the quad q (see inset). It is worth insisting on this: while the unit normal n_{uv} is not a bilinear interpolant, **the non-normalized normal N_{uv} is a simple bilinear function interpolating** $\{N^q_k\}$ (see 3rd line of Eq. (8)).

Neumann conditions and associated coordinates. We use the *deformed area over rest-pose area* ratio as a baseline for our Neumann conditions. Our (uv-varying) **area-based** Neumann condition reads therefore

$$\begin{cases} \frac{\partial f}{\partial n}(q_{uv}) & = \sigma^q_{uv} n'_{uv} \; \forall q_{uv} \in q \\ \sigma^q_{uv} & := \|N'_{uv}\|/\|N_{uv}\| \end{cases} \tag{9}$$

leading to our area-based Green coordinates for triquad cages:

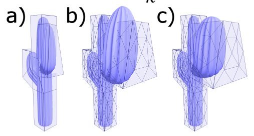
$$f(\eta) = \sum_i \phi_i(\eta) v'_i + \sum_{t\in\mathcal{T}} \psi_t(\eta)\sigma_t n'_t + \sum_{q\in Q}\sum_{k=0}^{3} \psi^q_k(\eta) N^{q\prime}_k \tag{10}$$

where a quad's contribution $\phi^q_k(\eta)$ to the ϕ coordinate of its corner vertices and the per-corner ψ coordinates $\psi^q_k(\eta)$ can be obtained as

$$\phi^q_k(\eta) = \int_{u,v=0}^{1} \frac{b^k_{uv}(q_{uv} - \eta)\cdot N_{uv}}{4\pi\|q_{uv} - \eta\|^3} du dv \tag{11}$$

$$\psi^q_k(\eta) = \int_{u,v=0}^{1} \frac{b^k_{uv}}{4\pi\|q_{uv} - \eta\|} du dv \tag{12}$$

In particular, we note that a point η will have in the end $|\mathcal{V}| + |\mathcal{T}| + 4|Q|$ coordinates for an input cage made of $|\mathcal{V}|$ vertices, $|\mathcal{T}|$ triangles and $|Q|$ quads, and that the quads' stretching conditions are directly met through the use of the unnormalized N^q_k.



While using this Neumann condition is key to make appear simple forms for our ψ^q coordinates, Lipman's choice of stretch factor leads to deformations that follow the cage more intuitively. This is illustrated in the inset figure (a: input, b: area-based formulation, c: Lipman's formulation). In this example, which is representative of all our experiments, using the area-based formulation leads to an exaggerated protuberance outside the deformed cage. We show in Sec. 3.5 how to retain the simplicity of our coordinates while inserting back the behavior shown by Lipman et al., by analyzing the ratio between both stretch factors in average over the quad.

3.2 Smooth approximate coordinates

Unfortunately, *to the best of our knowledge*, the integrals in Eqs.(11) and (12) do not admit closed-form expressions. We proceed quad-per-quad, and derive two integral geometric quantities that are invariant to the surface interpolating the 4 edges of the quad. This permits to substitute triangles to the curved quad, and obtain exact computation of these integral invariants as a result (while integrating functions on the curved quad remains difficult). We use these invariants to establish constraints on a per-quad basis, which ensure the validity of our coordinates i.e., linear precision.

More precisely, we consider a quad q, and a tessellation of q into triangles $t^j = (t^j_0, t^j_1, t^j_2) \in \mathbb{R}^{3\times3}$ (we will discuss later the choice of the tessellation). Considering their respective contributions in the discretization of the integrals in Eq. (1), we derive

$$\sum_{k=0}^{3} \phi^q_k(\eta) q_k + \psi^q_k(\eta) N^q_k = \sum_j \sum_{k=0}^{2} \phi^{t^j}_{t^j_k}(\eta) t^j_k + \sum_j \psi_{t^j}(\eta) n_{t^j} \tag{13}$$

Noting that $\int_{\xi\in q} \frac{\partial G}{\partial n}(\xi,\eta) d\xi =: \omega_q(\eta)/(4\pi)$, $\omega_q(\eta)$ denoting the signed solid angle of q at point η, we further conclude that

$$\sum_{k=0}^{3} \phi^q_k(\eta) = \sum_j \sum_{k=0}^{2} \phi^{t^j}_{t^j_k}(\eta) \tag{14}$$

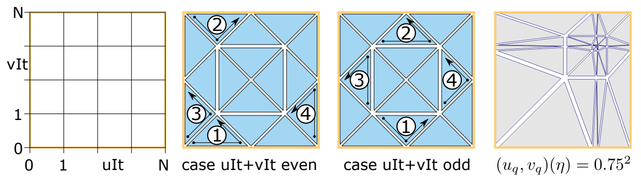


Fig. 2. Adaptive *uv*-tessellation for the Riemann summation. *First*: *uv*-grid. *Second and third*: Triangles covered in order as described in Algo. 1, using $n = 1$. *Fourth*: Adaptive patterns centered in $(0.75, 0.75)$, using $n = 2$.

Noting $\Phi := (\phi_0^q, \phi_1^q, \phi_2^q, \phi_3^q, \psi_0^q, \psi_1^q, \psi_2^q, \psi_3^q) \in \mathbb{R}^8$ our unknowns (omitting η for clarity), Eq. (13) and (14) can be put in matrix form:

$$A_q \cdot \Phi = m_q(\eta) \in \mathbb{R}^4, \tag{15}$$

$$A_q := \begin{pmatrix} q_0 & q_1 & q_2 & q_3 & N_0^q & N_1^q & N_2^q & N_3^q \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \end{pmatrix} \in \mathbb{R}^{4\times 8} \tag{16}$$

Lemma 3.1. *A_q is always of rank 4 for any non-degenerate quad.*

Proof. We first show that the first three lines are independent. If it weren't true, there would be a linear combination of those lines resulting in the null 8D line vector. There would exist a non-zero 3D vector d, whose dot product with q_k is zero $\forall k$, implying that all four corners belong to the plane passing through the origin with normal d. If it were true, all corner normals N_k^q would be colinear with d and therefore $N_k^q \cdot d \neq 0$ (as $N_k^q = 0\ \forall k$ iif the quad is degenerate). The first three lines span therefore a space of rank 3. We show now that the last line is independent from the first three. If it were not the case, there would be a non-zero 3D vector d such that all corners belong to the same plane with normal d at distance $\|d\|^{-1}$ from the origin ($\Leftrightarrow d \cdot q_k = 1\ \forall k$), implying once again that N_k^q is colinear with $d\ \forall k$ and that $N_k^q \cdot d \neq 0$, finishing the proof. □

We note $\bar{\Phi}$ the least-norm solution to Eq. (15), and $\kappa_i \in \mathbb{R}^8, i \in [0, 3]$ the four 8D vectors spanning the null space of A_q, all obtained using the singular value decomposition (SVD) of A_q.

The solution to our problem takes the form $\Phi = \bar{\Phi} + \sum_{i=0}^{3} \lambda_i \kappa_i$, λ_i being the missing components along the null space of A_q. Note that, κ_i being in the null space of the *linear precision constraints* (Eq. (13)), *any choice* of λ_i leads to linearly-precise coordinates.

Following [Thiery et al. 2018], we start by computing a smooth approximation $\tilde{\Phi}$ of Φ using a smooth adaptive Riemann summation approximating Eqs. (11) and (12). We finally set $\lambda_i = \frac{\tilde{\Phi}\cdot\kappa_i}{\|\kappa_i\|^2} \frac{\|\bar{\Phi}\|^2}{\bar{\Phi}\cdot\bar{\Phi}}$.

3.3 Robust Riemann summation

While in [Thiery et al. 2018], three fourth of the solution's spectrum was constrained by the validity equations and could serve as a strong basis to robustly correct the estimate of the component along the null space, we can only constrain four eighths (half) of the solution, and we observe that a standard Riemann summation (as in [Thiery et al. 2018]) is not robust enough to estimate the remaining four unknown components λ_i. We cope with that issue by designing a robust and efficient triangulation-based approximation. We triangulate the *uv* square domain $[0, 1]^2 =: \bigcup\{t\}$, and rewrite Eqs.(11) and (12) as

$$\begin{aligned} \phi_k^q(\eta) &= \sum_t \int_{(u,v)\in t} \frac{b_{uv}^k (q_{uv} - \eta)\cdot N_{uv}}{4\pi\|q_{uv} - \eta\|^3} dudv \\ &\simeq \sum_t b_{uv_t}^k \int_{(u,v)\in t} \frac{(q_{uv} - \eta)\cdot N_{uv}}{4\pi\|q_{uv} - \eta\|^3} dudv \\ &= \sum_t b_{uv_t}^k \int_{\xi\in t(q)} \frac{\partial G}{\partial n}(\xi,\eta) d\xi = \sum_t \frac{b_{uv_t}^k \omega_t(\eta)}{4\pi} \end{aligned} \tag{17}$$

$$\begin{aligned} \psi_k^q(\eta) &= \sum_t \int_{(u,v)\in t} \frac{b_{uv}^k}{4\pi\|q_{uv} - \eta\|} dudv \\ &\simeq \sum_t \frac{b_{uv_t}^k}{\|N_{uv_t}\|} \int_{(u,v)\in t} \frac{\|N_{uv}\|}{4\pi\|q_{uv} - \eta\|} dudv \\ &= \sum_t \frac{b_{uv_t}^k}{\|N_{uv_t}\|} \int_{\xi\in t(q)} -G(\xi,\eta) d\xi \simeq \sum_t \frac{b_{uv_t}^k \psi^t(\eta)}{\|N_{uv_t}\|}, \end{aligned} \tag{18}$$

$\omega_t(\eta)$ denoting the signed solid angle of t at η, uv_t denoting the *uv*-location of the center of t (obtained by simple averaging in *uv*-space), and $t(q)$ denoting the 3D embedding of t on the (in general, curved) quad q. Our final approximation makes therefore the triangular Green coordinates [Lipman et al. 2008] appear.

Note that these integrals (Eq. (17),(18)) are approximated at two different levels. First, we assume that b_{uv}^k and $b_{uv}^k/\|N_{uv}\|$ vary relatively slowly on t, and we consider them constant and fixed at its center. Second, we assume that, even if t is by definition flat in the *uv*-domain, its 3D embedding $t(q)$ on q is not, and the last integrals are not strictly equivalent to the expressions given in [Lipman et al. 2008]. Nevertheless, we found this approximation to be robust enough to estimate correctly the components along the null space of Eq. (15). To tile the *uv*-domain with triangles, we use a strategy inspired by the one used in [Thiery et al. 2018]:

(1) We start by computing an appropriate *uv*-location $(u_q, v_q)(\eta)$ using the proposed *uv*-projection operator $\mathcal{P}_q : \mathbb{R}^3 \to \mathbb{R}^2$ defined in [Thiery et al. 2018]. $\mathcal{P}_q$ is designed to converge to the orthogonal projection operator near q, while smoothly transiting to a simple averaging operator far away from q.
(2) Given $(u_q, v_q)(\eta)$, we design an adaptive *uv*-grid-pattern (Eq. (19)), and we tile the *uv*-domain atop this pattern, following Algo. 1 (see Fig. 2, 2nd and 3rd; 4th image showcases the pattern obtained for $n = 2$).

In order to *concentrate* the sampling around $(u_q, v_q)(\eta)$ to better account for the expected energy concentration resulting from the use of a $1/\|\xi - \eta\|^k$ averaging kernel, we use the following procedure to compute the *uv*-pattern of size $(2n + 1)^2$ (for $x = u$ and v):

$$x_i := \begin{cases} \left[1 - \left(\frac{n-i}{n}\right)^m\right] x_q(\eta) & \forall\ i < n \\ x_q(\eta) & i == n \\ (x_q(\eta) - 1)\left[1 - \left(\frac{i-n}{n}\right)^m\right] + 1 & \forall\ n+1 \le i \le 2n \end{cases} \tag{19}$$

In practice, we use $m = 3$ in our implementation, which we found empirically to give good results.

ALGORITHM 1: Adaptive Riemann summation

```
// Input: evaluation point $\eta$, quad $q$
// Output: estimate of Eqs. (11) and (12)
$\tilde{\Phi} \leftarrow 0; N = 2n + 2;$
$\mathbf{u}, \mathbf{v}$: arrays of size $N + 1 = 2n + 3$ centered on $(u_q, v_q)(\eta)$ //Eq. (19)
for $vIt = 0$ to $n$ do
    for $uIt = vIt$ to $2n - vIt$ do
        if $uIt + vIt \% 2 == 0$ then
            Proc$(uIt, uIt + 2, uIt + 1, vIt, vIt, vIt + 1)$;
            Proc$(uIt, uIt + 1, uIt + 2, N - vIt, N - vIt - 1, N - vIt)$;
            Proc$(vIt, vIt + 1, vIt, uIt, uIt + 1, uIt + 2)$;
            Proc$(N - vIt, N - vIt, N - vIt - 1, uIt, uIt + 2, uIt + 1)$;
        else
            Proc$(uIt, uIt + 1, uIt + 2, vIt + 1, vIt, vIt + 1)$;
            Proc$(uIt, uIt+2, uIt+1, N - vIt - 1, N - vIt - 1, N - vIt)$;
            Proc$(vIt + 1, vIt + 1, vIt, uIt, uIt + 2, uIt + 1)$;
            Proc$(N - vIt - 1, N - vIt, N - vIt - 1, uIt, uIt + 1, uIt + 2)$;
return $\tilde{\Phi}$

Proc$(u_1, u_2, u_3, v_1, v_2, v_3)$ :
$\mathbf{u_i} = \mathbf{u}[u_i], \mathbf{v_i} = \mathbf{v}[v_i], \ \forall i$
$\bar{u} = (\mathbf{u_1} + \mathbf{u_2} + \mathbf{u_3})/3$; $\bar{v} = (\mathbf{v_1} + \mathbf{v_2} + \mathbf{v_3})/3$;
$t \leftarrow \{q_{\mathbf{u_1 v_1}}; q_{\mathbf{u_2 v_2}}; q_{\mathbf{u_3 v_3}}\}$ // tessellated triangle
$\tilde{\Phi} = \tilde{\Phi} + \left(b_{\bar{u}\bar{v}}\omega_t(\eta)/(4\pi); \psi^t(\eta) b_{\bar{u}\bar{v}}/\|N^q_{\bar{u}\bar{v}}\|\right)$
```

3.4 Robust evaluation of the invariants

We discuss here what are the required properties of the tessellation $\{t^j\}$ needed to evaluate robustly the right-hand side of Eqs. (13),(14):

$$\begin{cases} \sum_j \sum_{k=0}^2 \phi^{t^j}_{t^j_k}(\eta) t^j_k + \sum_j \psi_{t^j}(\eta) n_{t^j} \\ \sum_j \sum_{k=0}^2 \phi^{t^j}_{t^j_k}(\eta) \end{cases}$$

To obtain computations that are equivalent to the smooth integrals on the quad, it is sufficient to ensure that η *remains on the correct side of the tessellated surface* (equivalently, η should not be inside the volume delimited by q and its tessellation). This condition ensures that the notion of *interior/exterior* remains unchanged from the point of view of η (in particular, the second invariant is the solid angle of q at η). To obtain valid tessellations for arbitrary $\eta \in \Omega$, we use the adaptive tessellation introduced in the previous paragraph (we use 4 triangles for this, i.e., $n = 0$), and uses the fact that $\mathcal{P}_q$ tends to the orthogonal projection operator around q.

3.5 Mimicking Lipman et al.' Neumann conditions

It is possible to "insert back" in some sense the Neumann conditions presented in [Lipman et al. 2008]. Indeed, while both stretch factors

$$\sigma^L_{uv} := \sqrt{\frac{\|\partial'_u\|^2\|\partial_v\|^2 + \|\partial_u\|^2\|\partial'_v\|^2 - 2(\partial'_u \cdot \partial'_v)(\partial_u \cdot \partial_v)}{2\|\partial_u \times \partial_v\|^2}} \quad (20)$$

$$\sigma^A_{uv} := \frac{\|\partial'_u \times \partial'_v\|}{\|\partial_u \times \partial_v\|} = \frac{\|N'_{uv}\|}{\|N_{uv}\|} \quad (21)$$

(where $\partial_u, \partial_v, \partial'_u, \partial'_v$ denote the tangent vectors) differ for a given quad deformation, we observe that their ratio does not vary too much across the quad. We can therefore factor it outside the integrant without noticeably deviating from the global deformation behavior induced by the Green coordinates under limit refinement of the quads into triangles (see Fig. 3 and next section), as this pointwise deviation is visibly averaged out once integrated:

$$\int_{u,v=0}^{1} \overbrace{G(q_{uv}, \eta)}^{:=G_{uv}} \sigma^L_{uv} n'_{uv} dq_{uv} =^1 \int_{u,v=0}^{1} G_{uv} \frac{\sigma^L_{uv}}{\sigma^A_{uv}} N'_{uv} du dv =^2$$

$$\sum_{k=0}^{3} \left(\int_{u,v=0}^{1} G_{uv} \frac{\sigma^L_{uv}}{\sigma^A_{uv}} b^k_{uv} du dv \right) N^{q'}_k \simeq \sum_{k=0}^{3} \sigma^k_q \left(\int_{u,v=0}^{1} G_{uv} b^k_{uv} du dv \right) N^{q'}_k$$

Based on this empirical observation, we compute σ^k_q, the per-quad corner's correction factor, as

$$\sigma^k_q := \int_{u,v=0}^{1} b^k_{uv} \sigma^L_{uv} dq_{uv} \Big/ \int_{u,v=0}^{1} b^k_{uv} \sigma^A_{uv} dq_{uv} \quad (22)$$

that estimates in average the uv-varying ratio $\sigma^L_{uv}/\sigma^A_{uv}$, using b^k_{uv} as importance sampler and accounting for varying local area density (using dq_{uv} and not "just" $dudv$ as differential element). We approximate these expressions using a fixed regular uv-pattern. These can be slightly simplified, as $dq_{uv}/\|\partial_u \times \partial_v\| = dq_{uv}/\|N_{uv}\| = dudv$. Using these correction factors leads to the following final expression for our Green coordinates for triquad cages:

$$f(\eta) = \sum_{i \in \mathcal{V}} \phi_i(\eta) v'_i + \sum_{t \in \mathcal{T}} \psi_t(\eta) \sigma_t n'_t + \sum_{q \in Q} \sum_{k=0}^{3} \psi^q_k(\eta) \sigma^k_q N^{q'}_k \quad (23)$$

Note that approximating these integrals using a basic fixed pattern is in this case harmless: **i)** Since these computations are performed independently of η, the final correction step results in smooth spatial deformations for varying η (we merely multiply smooth functions by constant scalars in Eq. (23)). **ii)** Since the various terms in Eq. (22) are regular w.r.t. $\{v'_i\}$, it also leads to smooth deformations as a function of $\{v'_i\}$ (ensuring smooth update of the mesh under smooth cage modification). We use in our implementation a simple 5×5 regular pattern to approximate Eq. (22). Note however that Eq. (22) is still ill-defined for degenerate quads q'. In this case, N'_{uv} is null on the whole uv-domain, the denominator in Eq. (22) becomes null, and multiplying these factors by the four (equal, null) per-corner $N^{q'}_k$ amounts to *normalizing* the null 3D vectors $N^{q'}_k$. In that sense, our corrected formula inherits in this (unconventional) situation the ill-defined behavior of [Lipman et al. 2008], as discussed at the end of Section 2. Fig. 3 shows the different flavors of deformations that can be obtained using either the *uncorrected* (area-based) expression or the *corrected* (conformal-factor-based) expression.

4 ANALYSIS AND DISCUSSION

We present our main results in Fig. 4, and we compare our coordinates with QMVC [Thiery et al. 2018] and GC [Lipman et al. 2008]. As one can see, our coordinates avoid strong asymmetric artifacts resulting from arbitrary triangulations of the quads (similarly to QMVC) and provide quasi-conformal deformations that avoid loss of geometric details in highly-stretched regions (similarly to GC).

^{1}using $\sigma^A_{uv} = \|N'_{uv}\|/\|N_{uv}\|$, $N'_{uv} = \|N'_{uv}\| n'_{uv}$ and $dq_{uv} = \|N_{uv}\| dudv$
^{2}using $N'_{uv} = \sum_{k=0}^{3} b^k_{uv} N^{q'}_k$

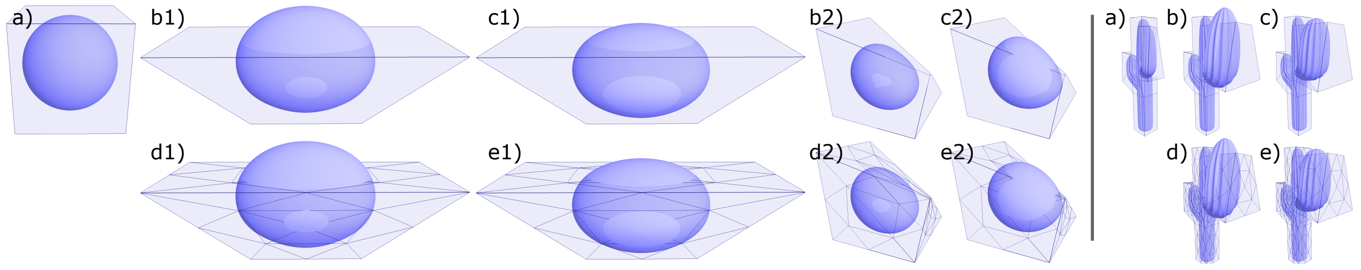


Fig. 3. a) Input. bX) Area-based QGC. cX) Conformal-factor-based QGC obtained using correction factors. dX) Area-based triangular GC. eX) Conformal-factor-based triangular GC. The triangular GC are obtained by cutting each quad into 16 triangles.

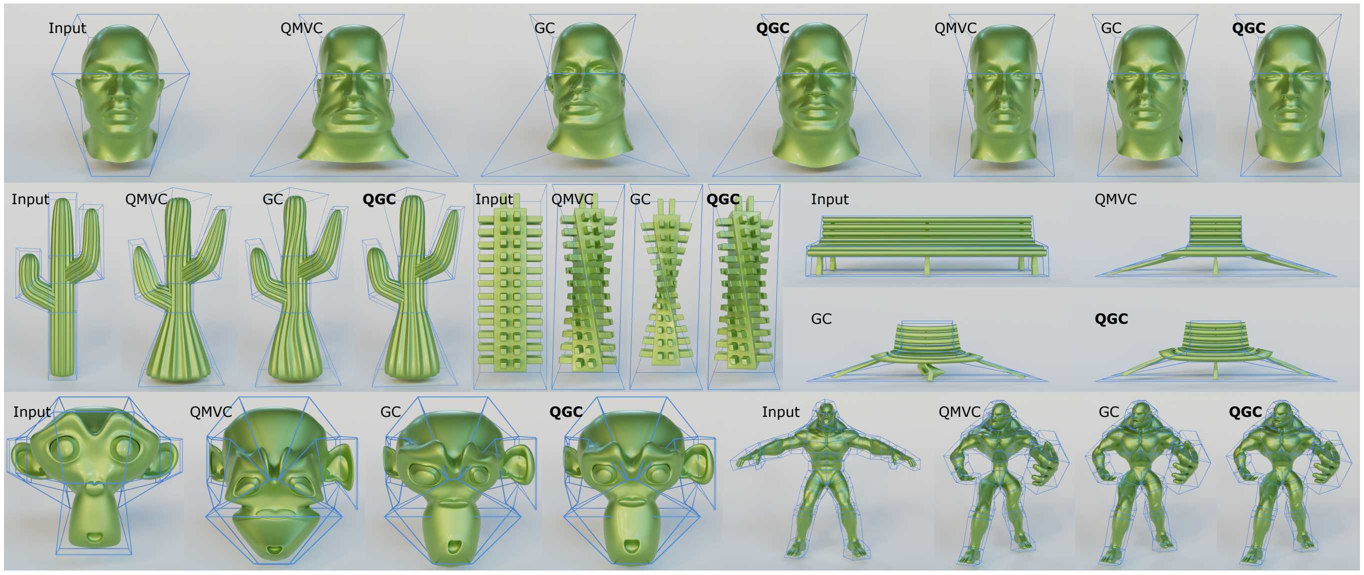


Fig. 4. We compare our Green Coordinates for Quad cages (QGC) with Mean-Value Coordinates for Quad cages (QMVC) and Green Coordinates (GC).

Implementation notes. The various A_q matrices depend on the rest-pose cage only (neither on the deformed cage, nor on the evaluation point η), and the SVD of A_q as well as the computation of its null space can be performed in a preprocess, which accelerates the computation of the coordinates. While performing such computations on the GPU is a bit difficult, we could imagine performing this step on the CPU at cage-loading time while performing the per-mesh-vertex computations of the coordinates on the GPU. Fine-tuning the computations to obtain the best possible performances is one of our future work. We also noted that numerical instabilities could happen for extremely large models (probably due to the many multiplications of very small numbers with very large ones in the Riemann summation). We recommend scaling down the input rest-pose cage and mesh in a preprocess, which does not change anything to the results since our formulation is scale-invariant.

Timings. We detail in Tab.1 the timings for the computations of our coordinates. As discussed just above, some computations involve the input rest pose cage only, some require the rest pose cage and the rest pose mesh, and others involve the rest pose cage and deformed cage only, as they are factorized for all the mesh vertices: The four per-quad correction factors (Eq. (22)) depend indeed on the cage only and not on the evaluation point η, and they are computed and stored at each frame before the update of the mesh transformation. This step, as well as the computation of the SVD of the various 4×8 A_q matrices (for which we use the Eigen library [Guennebaud et al. 2010]), take no more than a few microseconds per quad, and are omitted here. Our CPU implementation was evaluated on a 12-core *Intel i7-10850H CPU @ 2.70GHz* with *32 GB* of RAM.

Mesh	$\#\mathcal{V}_M$	$\#\mathcal{V}$	$\#\mathcal{T}$	$\#\mathcal{Q}$	GC	TGC-16	TGC-64	**QGC**
SpikyBar	60802	8	0	6	30	169	580	382
Head	24658	12	0	10	18	130	395	249
Cactus	98820	36	0	34	174	2053	9202	6311
Bench	65430	24	0	22	91	905	4200	2728
Monkey	126290	38	2	35	232	2703	11925	8919
Car	64962	16	0	14	64	581	2442	1681
Beast	28388	162	44	138	236	2686	10663	7354

Table 1. Timings (in ms). $\#\mathcal{V}_M$/: mesh vertices. $\#\mathcal{V}/\#\mathcal{T}/\#\mathcal{Q}$: cage vertices/triangles/quads. GC: Green coordinates (each quad has been subdivided into two triangles). TGC-X: Tesselation-based Green coordinates (each quad has been subdivided into X triangles).

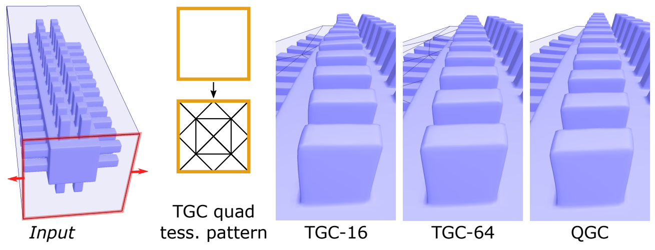


Fig. 5. One cage quad is stretched in the x-direction. The jaggy deformation artifact (third) stems from the approximation of the (symmetric) bilinear quad deformation with the (asymmetric) triangle deformation introduced in the quad tessellation (each quad was tesselated into X triangles for TGC-X).

Comparison with a dense cage tessellation approach. We compare our technique with an approach consisting in tessellating each quad of the cage (both in rest-pose and deformed state similarly).

While formally, our formulation should tend to the tessellation-based deformations (under infinite refinement) when using the area-based Neumann condition, our corrected formulation (Eq. (23)) should approximate only the tessellation-based deformation under infinite refinement when using the conformal-factor Neumann condition, since our uv-varying corrected Neumann condition does not match exactly the uv-varying Neumann condition of Lipman et al. [Lipman et al. 2008]. Nevertheless, as seen in Fig. 3, our coordinates provides deformations that are overall extremely similar, even at large scale, for both types of Neumann conditions.

An important point to consider is that, even if a large number of triangles are used per quad — entailing large memory and computational workload cost — the tessellation approach always results in deformations suffering from asymmetric artifacts stemming from the triangulation. Fig. 5 illustrates such asymmetric deformation artifacts obtained for a simple sheer of the cage quads (thus with constant normal across the tessellated quad), when each quad is tessellated into 16 triangles. Note that, while the various ϕ coordinates w.r.t. the inserted cage vertices can trivially be factored back into the ϕ coordinates of the original cage vertices, the same can not be done for the ψ coordinates: it seems difficult to express the various introduced $\sigma_t \psi_t n_t$ as a linear combination of per-corner quantities (such as N_q^k) and their contribution cannot be easily factored into a simple expression making 4 terms appear only. This implies that the naïve tessellation approach requires significantly more coordinates per input quad, while we introduce only 4 corner coordinates per input quad without suffering from asymmetric deformation artifacts.

Limitations. The most notable limitation of our work comes from our approximation strategy. While it guarantees smooth coordinates, it prevents us from deriving simple closed-form expressions for those. As a result, it also prevents us from computing derivatives (such as gradients and Hessians) which are useful for some applications, such as variational shape deformation [Ben-Chen et al. 2009]. Designing such an implicit variational framework using our coordinates is one future research direction we plan to pursue.

Secondly, since our formulation incorporates directly the stretch conditions into the product of the coordinates and the non-normalized corner normals, it might be difficult to play with this parameter in order to deviate from quasi-conformal deformations if desired by the artist. One way to address this could be to apply a different per quad corner's correction factor (Eq. (22)) to account for per vertex (or per face corner) scalar functions provided by the artist.

Last, our coordinates are only defined for the quads that pass the validity test required for the projection operator [Thiery et al. 2018] to be defined everywhere in space. Designing a projection operator allowing for a larger family of quads is an interesting direction of research, keeping in mind however that there is a limit to this quest, as we know already that some quads (planar non-convex quads for example, whose spatial bilinear geometry is not injective) can never be used for cage-based modeling by definition.

Conclusion. Our approach combines the best of both worlds: as with QMVC, arbitrary triquad meshes, natural to edit interactively, can be used as the control cage of a high-resolution shape; and as with GC, local quasi-conformality — which translates into far better structure and geometric details preservation — is provided. In the future, exploring how such properties can benefit beyond deformation would be an interesting venue for research, as cage coordinates in general allow diffusing any on-surface signal from the cage to the high resolution shape.

REFERENCES

Mirela Ben-Chen, Ofir Weber, and Craig Gotsman. 2009. Variational harmonic maps for space deformation. *ACM ToG* 28, 3 (2009), 1–11.
Péter Borosán, Reid Howard, Shaoting Zhang, and Andrew Nealen. 2010. Hybrid Mesh Editing.. In *Eurographics (short papers)*. 41–44.
Michael S Floater. 2003. Mean value coordinates. *Computer aided geometric design* 20, 1 (2003), 19–27.
Michael S Floater. 2015. Generalized barycentric coordinates and applications. *Acta Numerica* 24 (2015), 161–214.
Gaël Guennebaud, Benoit Jacob, et al. 2010. Eigen. *URl: http://eigen. tuxfamily. org* 3 (2010).
Kai Hormann and N Sukumar. 2017. *Generalized barycentric coordinates in computer graphics and computational mechanics.* CRC press.
Jin Huang, Xiaohan Shi, Xinguo Liu, Kun Zhou, Li-Yi Wei, Shang-Hua Teng, Hujun Bao, Baining Guo, and Heung-Yeung Shum. 2006. Subspace gradient domain mesh deformation. In *ACM SIGGRAPH 2006 Papers*. 1126–1134.
Alec Jacobson, Zhigang Deng, Ladislav Kavan, and John P Lewis. 2014. Skinning: Real-time shape deformation. In *ACM SIGGRAPH 2014 Courses*. 1–1.
Pushkar Joshi, Mark Meyer, Tony DeRose, Brian Green, and Tom Sanocki. 2007. Harmonic coordinates for character articulation. *ACM TOG* 26, 3 (2007), 71–es.
Tao Ju, Scott Schaefer, and Joe Warren. 2005. Mean value coordinates for closed triangular meshes. In *ACM Siggraph 2005 Papers*. 561–566.
Torsten Langer, Alexander Belyaev, and Hans-Peter Seidel. 2006. Spherical barycentric coordinates. In *Symposium on Geometry Processing*. 81–88.
Yaron Lipman, Johannes Kopf, Daniel Cohen-Or, and David Levin. 2007. GPU-assisted positive mean value coordinates for mesh deformations. In *SGP*. 117–123.
Yaron Lipman, David Levin, and Daniel Cohen-Or. 2008. Green coordinates. *ACM ToG* 27, 3 (2008), 1–10.
Jean-Marc Thiery, Pooran Memari, and Tamy Boubekeur. 2018. Mean value coordinates for quad cages in 3D. *ACM ToG* 37, 6 (2018), 1–14.
Jean-Marc Thiery, Julien Tierny, and Tamy Boubekeur. 2014. Jacobians and Hessians of mean value coordinates for closed triangular meshes. *The Visual Computer* 30, 9 (2014), 981–995.