Higher-Order Continuity for Smooth As-Rigid-As-Possible Shape Modeling

Annika Oehri
ETH Zurich

Philipp Herholz*

Olga Sorkine-Hornung
ETH Zurich

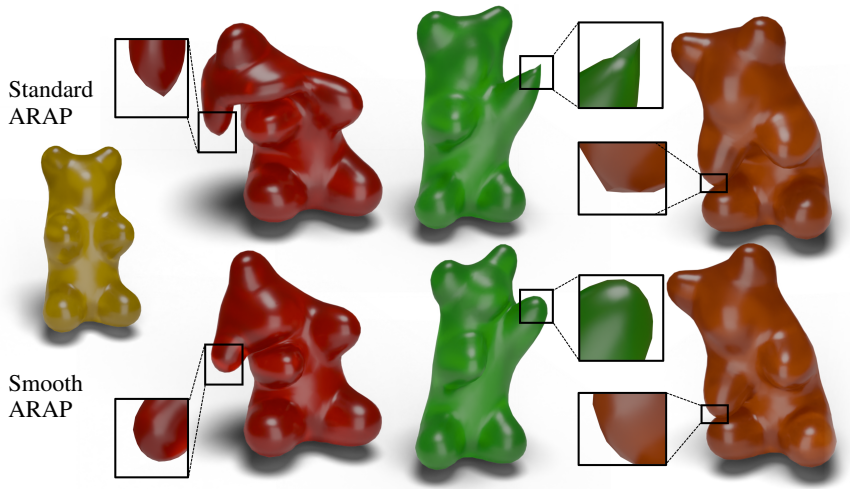

Figure 1. Deformations of a gummy bear, the original mesh depicted in yellow. The standard ARAP method [Sorkine and Alexa 2007] can yield spikes, as visible in the highlighted regions. In contrast, our method produces smooth deformations.

Abstract

We propose a modification of the As-Rigid-As-Possible (ARAP) mesh deformation energy with higher-order smoothness, which overcomes a prominent limitation of the original ARAP formulation: spikes and lack of continuity at the manipulation handles. Our method avoids spikes even when using single-point positional constraints. Since no explicit rotations have to be specified, the user interaction can be realized through a simple click-and-drag interface, where points on the mesh can be selected and moved around while the rest of the mesh surface automatically deforms accordingly. Our method preserves the benefits of ARAP deformations: it is easy to implement and thus useful for practical applications, while its efficiency makes it usable in real-time, interactive scenarios on detailed models.

*Work done at ETH Zurich

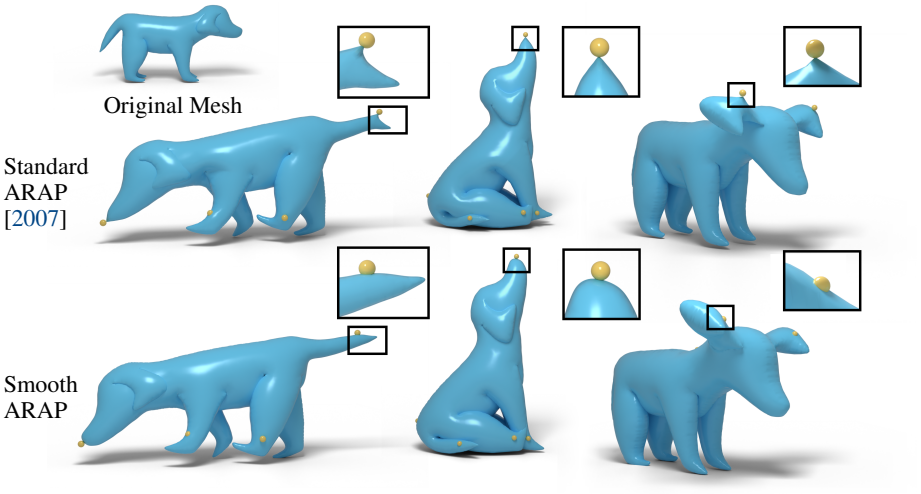

Figure 2. Interactive deformation of a dog model from the Monster Mash system [Dvorožňák et al. 2020] via single-point handles, indicated by yellow spheres. Some marked regions are shown in close-ups to better visualize the spikes produced by ARAP [Sorkine and Alexa 2007] and the absence of spikes in our smooth formulation.

1. Introduction

Shape deformation using user-defined, click-and-drag position constraints is a versatile method with numerous applications in industrial and artistic design [Sorkine and Botsch 2009]. A prominent application is 3D shape modeling and animation for lay users, exemplified by the Monster Mash system [Dvorožňák et al. 2020]. Monster Mash allows users to draw a 2D sketch of a shape, automatically inflates it into a 3D shape, and then enables users to interactively deform this mesh by simply clicking on surface points and moving them around, using As-Rigid-As-Possible (ARAP) shape deformation as the backbone [Sorkine and Alexa 2007].

ARAP [Sorkine and Alexa 2007] is a mesh deformation method that preserves the shape's overall appearance and geometric details by minimizing non-rigid local transformations, building on the fact that our understanding of shape is usually independent of its orientation and translation. The method is very efficient due to its formulation of the objective function, which, albeit highly nonlinear, can be quickly minimized via local-global optimization using sparse linear solvers. Additionally, a slight perturbation of the positional constraints usually causes only a small change to the resulting surface, making it especially useful for interactive applications such as Monster Mash. However, when deforming a mesh using the classical ARAP approach, spikes can appear, as depicted in Figures 1, 2, and 3. This lack of smoothness at the constraints is particularly visible with fine mesh resolution and when using single-point handles. A smoother deformation is often desired and expected, where no newly introduced sharp features distort the shape.

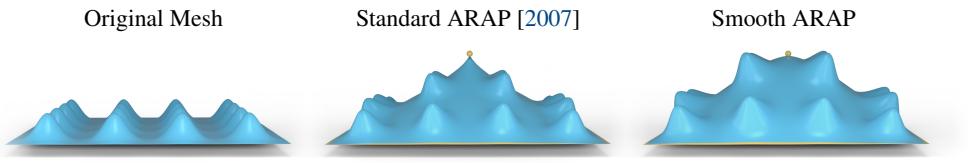

Figure 3. Interactive deformation of a bumpy plane by dragging a point handle in the center of the mesh while fixing the boundary vertices.

We propose a modification of the original ARAP method that yields smoother results by raising the order of the objective function while staying within the efficient and easy-to-implement local-global optimization framework. Our method enables effective use of single-point handles, which greatly simplify the interaction metaphor of the deformation without introducing spikes. An added benefit of employing point constraints is the ability to perform fast updates of the underlying system matrix factorization, affording fast addition and removal of point handles and making the deformation overall more fluid and responsive to user interactions.

1.1. Related Work

As-Rigid-As-Possible shape deformation [Sorkine and Alexa 2007] is achieved by minimizing an objective function termed *ARAP energy* that encourages local transformations of the surface to be as close to rigid as possible, while satisfying the positional constraints imposed by the manipulation handles. To minimize this nonlinear energy, a local-global optimization approach is adopted. In the local step, the optimal rigid transformation of each local neighborhood on the mesh is computed by solving the Procrustes problem. In the global step, the previously found rotations are kept fixed, and the vertex positions are updated by solving a sparse linear system. This process is iterated until convergence. The method is very robust, it is efficient thanks to precomputation and reuse of the sparse factorization of the system matrix and achieves compelling results through its automatic preservation of the relative orientation of local shape details. However, the deformed surface tends to lack tangent continuity at the positional constraints, and especially when single-point handles are used, spikes appear at the point constraints (see Figures 3 and 2). This phenomenon stems from the Poisson equation solved in the global step of ARAP, which cannot afford simultaneous positional and derivative constraints.

Smoothness is commonly achieved by applying regularization. While many approaches tackle the problem in an energy-independent manner, it can be beneficial to consider the specific objective energy at hand. The method of smoothed quadratic energies on meshes [Martinez Esturo et al. 2015] offers a way of enforcing lower variation of the energy function that is by construction problem-specific. The proposal is to add a term that penalizes the total squared variation of the local energy, favoring minima where the energy is distributed more evenly across the whole mesh.

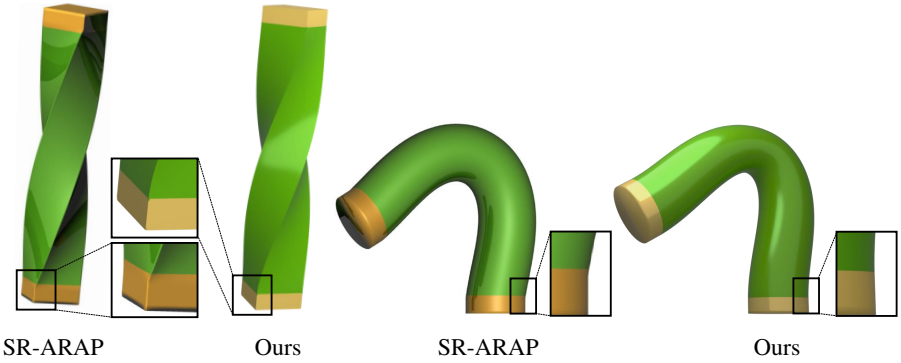

Figure 4. Comparison between SR-ARAP and our smooth ARAP on the benchmark survey examples [Botsch and Sorkine 2008]. The SR-ARAP results are taken from Figures 1 and 9 of Levi and Gotsman [2015], and our results are shown in a similar view. It can be seen that our global approach produces smoother transitions to the handles, while the locally smoothed rotations still have visible kinks, as highlighted.

This approach can be applied to different types of energies and application settings. Applied to ARAP, it yields a similar higher-order energy to ours, but with a different smoothing term, which uses the original ARAP neighborhood $\mathcal{N}$ instead of Laplacian vectors as we do; see Equation (1). Martinez Esturo et al. [2015] focused mainly on 2D examples, while we demonstrate that our smooth ARAP performs well in complex and interactive 3D shape deformations.

Levi and Gotsman [2015] proposed another form of smoothness regularization that specifically targets the surface deformation of ARAP, termed *SR-ARAP energy*. The smoothness of rotations is achieved by adding a term to the surface energy that penalizes the difference between the rotations of neighboring edge sets, to encourage neighborhoods to transform as a unit. The local step of ARAP is adapted such that the best rotation is found not only based on the deformed edge positions but also on the previously computed neighborhood rotations. This approach smooths rotations only locally, and the overall system remains Poisson-based, meaning positions from the global step may not reflect a smooth deformation, even with smooth rotations, as shown in Figure 4. Notably, the authors do not show results with single-point handles. In our method, we address the smoothness issue in the *global* step, by raising the order of the partial differential equation and employing the bi-Laplace operator, resulting in smooth deformation results even around point handles.

2. Energy Definition and Method

2.1. Problem Statement and Notation

Let $\mathcal{M} = (\mathcal{V}, \mathcal{T})$ be a given manifold, orientable triangle mesh with n vertices, where $\mathcal{V}$ is the vertex set with corresponding vertex positions $\mathbf{V} \in \mathbb{R}^{n\times 3}$ and $\mathcal{T}$ is the

face set, where we define the set of all triangles containing vertex v to be $\mathcal{T}_v = \{t \in \mathcal{T} | v \in t\}$. Our goal is to determine the new deformed configuration $\mathcal{M}' = (\mathcal{V}', \mathcal{T})$ satisfying some positional constraints, namely that the *handle vertices* are at the specified positions. At the same time, we want to minimize an energy to preserve the shape locally while avoiding spikes.

Let A_v denote the Voronoi area of the vertex $v \in \mathcal{V}$, and let all such areas be accumulated on the diagonal of the lumped mass matrix $\mathbf{M} \in \mathbb{R}^{n \times n}$. We use a half-edge data structure for the mesh and denote by $\mathcal{E} \subset \mathcal{V}^2$ the set of *half-edges* (or *directed edges*). Each half-edge belongs to a triangle, and we adopt the standard convention that the half-edges in a triangle are oriented counterclockwise. We use $e = (u, v)$ to refer to the half-edge e going from vertex u to v, and $\mathbf{e} = \mathbf{v} - \mathbf{u}$ to denote the corresponding vector in $\mathbb{R}^3$, which makes use of the vector representation $\mathbf{v}$ of vertex v. Additionally, we distinguish between the original and deformed configurations by marking the deformed configuration with a prime. For example, $\mathbf{e}, \mathbf{e}' \in \mathcal{E}$ both refer to the same half-edge e in terms of mesh connectivity, but $\mathbf{e}'$ uses the updated vertex positions $\mathbf{V}'$.

The standard ARAP energy can be defined over different neighborhoods [Jacobson et al. 2012], such as spokes-only [Sorkine and Alexa 2007], per-triangle [Liu et al. 2008], or spokes-and-rims [Chao et al. 2010]. We use spokes-and-rims and define the corresponding half-edge set for vertex v as

$$\mathcal{N}_v = \{e \in \mathcal{E} \mid e \in t \text{ and } t \in \mathcal{T}_v\}.$$

The weight w_e of each half-edge e is the cotangent of the angle α_e opposite of e in e's triangle (see Figure 5):

$$w_e = \cot \alpha_e.$$

The area-corrected cotan Laplacian vector at vertex v is defined in the standard way as

$$\boldsymbol{\ell}_v = \sum_{\{e \in \mathcal{E} | v \in e\}} \frac{w_e}{2A_v} d_e^v \mathbf{e}, \tag{1}$$

with d_e^v being the sign factor coming from the derivative of the edge vector $\mathbf{e}$ with regards to vertex $\mathbf{v}$:

$$d_e^v = \begin{cases} 1 & \text{if } \mathbf{e} = (\mathbf{v} - \mathbf{u}), u \in \mathcal{V}; \\ -1 & \text{if } \mathbf{e} = (\mathbf{u} - \mathbf{v}), u \in \mathcal{V}; \\ 0 & \text{else.} \end{cases}$$

These weights are accumulated in the cotan Laplacian matrix $\mathbf{L} \in \mathbb{R}^{n \times n}$, such that $(\mathbf{M}^{-1}\mathbf{L}\mathbf{V})_v = \boldsymbol{\ell}_v{}^{\mathsf{T}}$, where $(\mathbf{X})_v$ refers to row v of matrix $\mathbf{X}$.

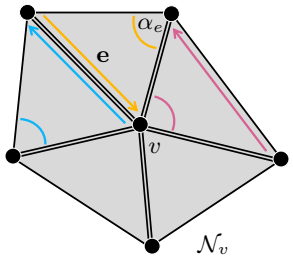

Figure 5. The spokes and rims neighborhood $\mathcal{N}_v$ around some vertex v. The spokes are the edges that directly involve v and thus appear twice in the set, represented through two different directions (or half-edges, see the blue and orange arrows). The rims, on the other hand, only appear in the set once as half-edges (one example is marked in pink). The angle opposite a half-edge used in the cotangent weight definition is marked in the same color as the half-edge. For example, the cotangent weight of the yellow half-edge $\mathbf{e}$ is α_e, thus $w_e = \cot \alpha_e$.

2.2. Energy and Method

With this notation, we now define the energy we propose to minimize in order to obtain natural-looking as well as smooth results. Our smooth ARAP energy E consists of two parts that are weighted against each other by the parameter $\lambda \in [0, 1)$, which controls how much the smoothness of the original mesh is preserved:

$$E = (1 - \lambda)E_{\text{ARAP}} + \lambda E_{\text{smooth}}. \tag{2}$$

The first term corresponds to the standard ARAP energy:

$$E_{\text{ARAP}} = \sum_{v \in \mathcal{V}} A_v \sum_{e \in \mathcal{N}(v)} \frac{w_e}{3A_v} \left\| \mathbf{e}' - \mathbf{R}_v \mathbf{e} \right\|^2, \tag{3}$$

$$\text{where} \quad \mathbf{R}_v = \underset{\mathbf{R}_v \in SO(3)}{\operatorname{argmin}} \sum_{e \in \mathcal{N}(v)} w_e \|\mathbf{e}' - \mathbf{R}_v \mathbf{e}\|^2. \tag{4}$$

Note that we added a coefficient of a third due to the overlap of neighborhoods, simply to get a nicer final equation to solve that does not have a scaled Laplacian matrix. The novel higher-order term to encourage smoothness is defined such that non-rigid transformations of the Laplacian vectors (rather than edge vectors, as for standard ARAP) increase the energy:

$$E_{\text{smooth}} = \sum_{v \in \mathcal{V}} A_v \|\boldsymbol{\ell}'_v - \mathbf{R}_v \boldsymbol{\ell}_v\|^2. \tag{5}$$

Note that the rotation $\mathbf{R}_v$ is the same as in the ARAP term in Equation (4).

To compute the deformed mesh, we can keep the local-global strategy from ARAP:

1. **Initialization:** Initialize $\mathcal{M}'$ in some way. We follow the method of Sorkine and Alexa [2007] and use naive Laplacian editing for the non-interactive ex-

amples unless stated otherwise. In the interactive setting, the mesh from the previous frame is used as initialization.

2. **Local step:** Compute the optimal rotation $\mathbf{R}_v$ for each vertex v using Equation (4), keeping the deformed vertex positions $\mathbf{V}'$ fixed (i.e., using the values from the previous iteration).

3. **Global step:** Solve a linear system for the new vertex positions $\mathbf{V}'$ while fixing the rotations $\mathbf{R}_v$ computed in the local step.

4. Go to step 2 until convergence.

2.3. Rotation Fitting

The optimal rotation $\mathbf{R}_v$ is determined as in the original ARAP method. Specifically, we define an auxiliary matrix $\mathbf{S}$ and find the desired $\mathbf{R}_v$ through its singular value decomposition (SVD) $\mathbf{S} = \mathbf{U}\mathbf{\Sigma}\mathbf{W}^\mathsf{T}$:

$$\mathbf{S} = \sum_{e \in \mathcal{N}(v)} w_e \, \mathbf{e}\, \mathbf{e}'^\mathsf{T} = \mathbf{U}\mathbf{\Sigma}\mathbf{W}^\mathsf{T},$$

$$\mathbf{R}_v = \mathbf{W} \begin{pmatrix} 1 & & \\ & 1 & \\ & & \det \mathbf{W}\mathbf{U}^\mathsf{T} \end{pmatrix} \mathbf{U}^\mathsf{T}.$$

Please refer to the original paper by Sorkine and Alexa [2007] for more details.

Note that in the definition of $\mathbf{R}_v$ in Equation (4), the rotation is only fitted to the original, non-smooth term. Thus, the local step is not guaranteed to be optimal for the energy E as a whole, as it might increase the value of the smooth term E_{smooth}. We call the rotation fitting from ARAP *edge-only*, as it considers only the mesh edge orientations for the rotation. To make the local step optimal for the whole energy, one would also have to consider the Laplacian vectors from the smooth energy term. We call this *full* rotation fitting in the following. While the standard SVD approach can be extended to also include Laplacian vectors to get the result of the full rotation fitting, there are some downsides to it. In certain settings, visible artifacts appear. An issue of this approach is that it is always possible to perfectly align the original Laplacian to the new one, so the global optimization can only work on the scale of the Laplacians rather than their orientation when the smoothness λ is very high, as well as it not having any direct awareness of bending in the local step. In practice, this can result in nonintuitive rotations, as depicted in Figure 6. It can be seen that the full rotation fitting does not enforce rigidity well, as the spikes do not preserve their local orientation to the mesh. Figure 7 shows that the rigidity is actually lower than for the initialization (even though the total energy E did decrease). While a trade-off

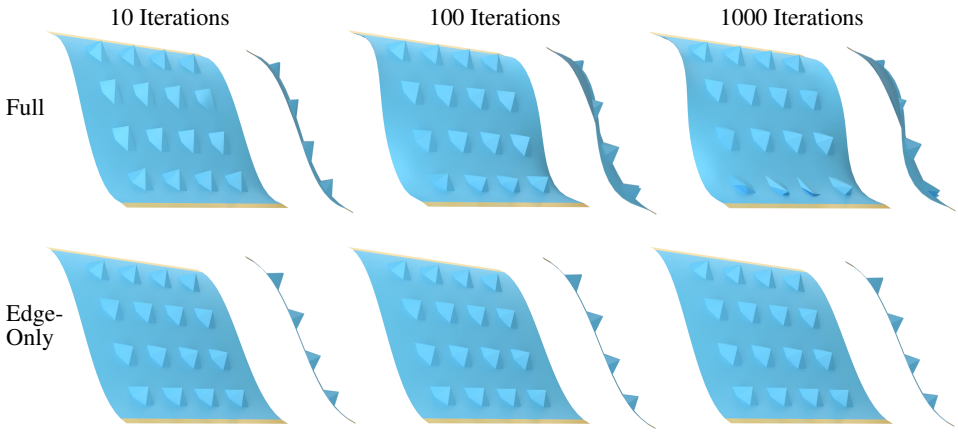

Figure 6. Deformations of a plane with orthogonal spike-like details, comparing full rotation fitting (including the vertex Laplacian vector) to our proposed edge-only formulation. The results after an increasing amount of iterations are displayed from two different views, to better show the orientation of the spikes.

between smoothness and rigidity is expected, the observed behavior in this case is undesirable.

Using rotations fitted only to edges does a much better job at preserving the orientation of local details. Another observation is that the mesh curves more around the border when using full rotation fitting (see Figure 6). Note that this curving at the border also happens with Laplacian editing [Sorkine et al. 2004], see, e.g., the bumpy plane in Figure 10. While in some other cases full rotation fitting does not present any observable artifacts, we never found it to be beneficial to the result, and it generally poses a disadvantage regarding the convergence and stability of the method. This can also be seen in the graph in Figure 7, where the energy when using full rotation

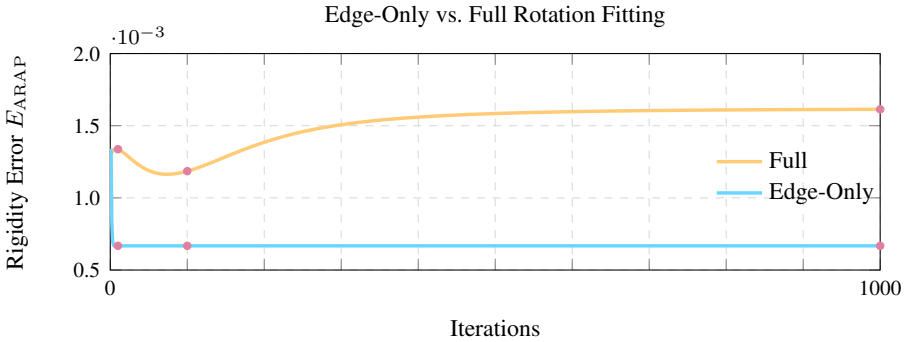

Figure 7. Convergence behavior of the spiky plane experiment (see Figure 6) in terms of rigidity (original ARAP energy E_{ARAP}) for both versions of rotation fitting. Pink dots mark the results shown in Figure 6.

fitting needs significantly longer to converge. Even though the edge-only rotation fitting does not use the optimal rotations for the smooth term in the local step, we have not observed energy fluctuations when evaluating the total energy with this rotation. Therefore, edge-only rotation fitting is our chosen approach.

2.4. Vertex Position Optimization

To deform the mesh, we need to find the new vertex positions $\mathbf{V}'$ that minimize the energy E. This can be achieved by differentiating the energy with regards to $\mathbf{V}'$. For the original ARAP term, the derivative has the following form:

$$\frac{\partial}{\partial \mathbf{V}'} E_{\text{ARAP}} = 2\mathbf{L}\mathbf{V}' - 2\mathbf{b}, \tag{6}$$

where $\mathbf{b}$ is a collection of the original edges rotated by the rotations determined in the local step. More specifically, row p of $2\mathbf{b}$ contains the derivative of E_{ARAP} with regards to vertex p, so

$$\mathbf{b}_p = \sum_{v \in \mathcal{V}} \sum_{e \in \mathcal{N}(v)} \frac{d_e^p w_e}{3} \mathbf{R}_v \mathbf{e}.$$

We rewrite the smooth energy term using a few matrices and an area-weighted Frobenius norm $\|\mathbf{B}\|_{FM}^2 := \sum_{i,j} A_i \|b_{i,j}\|^2$:

$$E_{\text{smooth}} = \sum_{v \in \mathcal{V}} A_v \|\boldsymbol{\ell}'_v - \mathbf{R}_v \boldsymbol{\ell}_v\|^2 \tag{7}$$

$$= \|\mathbf{L}\mathbf{V}' - \mathbf{R}\|_{FM}^2, \tag{8}$$

where $\mathbf{R}$ is constant in the wanted $\mathbf{V}'$, and each row i corresponds to the Laplacian $\boldsymbol{\ell}_i$ being rotated by $\mathbf{R}_i$, so $(\mathbf{R})_i = (\mathbf{R}_i \boldsymbol{\ell}_i)^\mathsf{T}$ holds. We calculate the derivative using these defined matrices:

$$\frac{\partial}{\partial \mathbf{V}'} E_{\text{smooth}} = 2\mathbf{L}^\mathsf{T}\mathbf{M}^{-1}\mathbf{L}\mathbf{V}' - 2\mathbf{L}^\mathsf{T}\mathbf{M}^{-1}\mathbf{R}. \tag{9}$$

Putting the two terms together and setting them to 0 to find the minimum, the global step amounts to solving the following sparse linear system:

$$\left(\lambda \mathbf{L}^\mathsf{T}\mathbf{M}^{-1}\mathbf{L} + (1-\lambda)\mathbf{L}\right)\mathbf{V}' = \lambda \mathbf{L}^\mathsf{T}\mathbf{M}^{-1}\mathbf{R} + (1-\lambda)\mathbf{b}. \tag{10}$$

To incorporate the handle positions, we solve this system in a constrained manner using substitution. This amounts to erasing the rows and columns corresponding to constrained positions and updating the right-hand side of the equation accordingly. This is a common approach in deformation that is also taken by the original ARAP method [Sorkine and Alexa 2007].

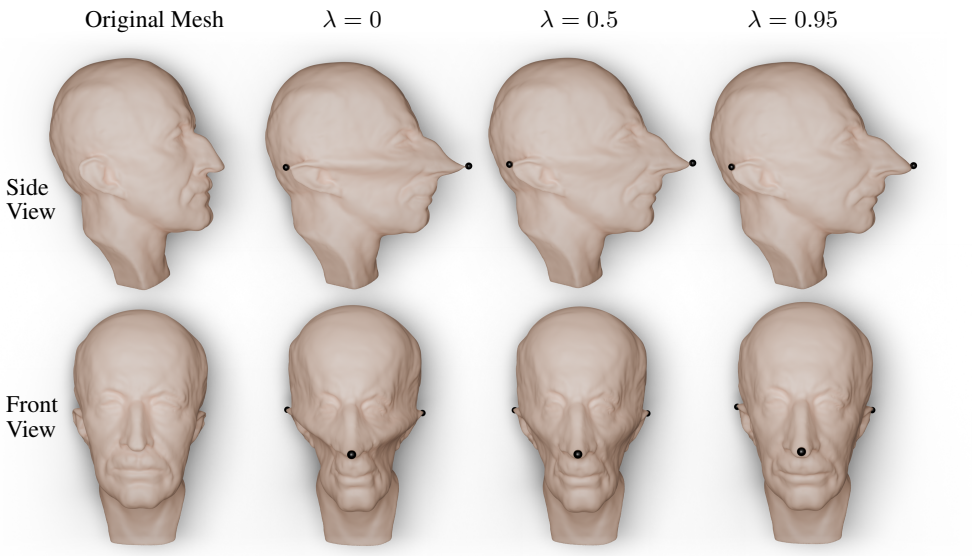

Figure 8. Influence of smoothness parameter λ when translating a single vertex on the nose of the Max Planck mesh away from the face, while a point on each ear is fixed, as marked by the black spheres.

2.5. Smoothness Control

The user-chosen deformation parameter λ controls the smoothness. In general, the higher the chosen λ, the smoother the result. This effect can be observed in Figure 8, where it can be seen that increasing values of λ both reduces the spikes around the handles, as well as limits how much the face can cave in due to the deformation. Alternatively, λ may also be used to regularize the deformation result. For most meshes, there are no problems when choosing an arbitrarily large λ. However, in certain challenging situations, setting λ too high can result in unnecessary rotations. This can happen for meshes that have long parts attached via very small areas, so the rotation of the long appendages can happen without having much impact on the overall energy. One such example is the dog model from Monster Mash [Dvorožňák et al. 2020]; see Figure 9. We only observed this phenomenon in the interactive setting, where it is easier for the system to get stuck in a bad local minimum while deforming the mesh. In these special cases, reducing λ helps mitigate the problem: in the mentioned example in Figure 9, this lets the dog's legs converge to a more natural position. Having escaped the bad local minimum, the user is often free to increase λ again if a smoother result is desired, as also shown in the figure. Thus, choosing a non-zero value for the original term is beneficial, and for challenging meshes in dynamic settings it is advised to use a λ significantly lower than 1. In general, no tuning is required for the smoothness parameter however, as for most meshes, choosing a high value is fine. In most examples in this paper, we opted for $\lambda = 0.95$, but for increased

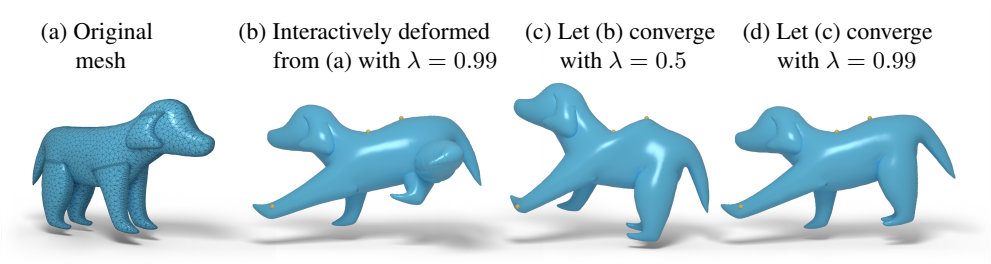

Figure 9. Influence of smoothness parameter λ to regularize deformations. When dynamically deforming difficult meshes such as the dog, whose highly irregular triangulation can be seen in (a), attached parts can rotate more than necessary, as can be seen in (b). Lower smoothness, such as $\lambda = 0.5$, can recover a better solution again, see (c). Initializing with that, the smoothness can also be reset and still produces nice results, as can be seen in (d) with $\lambda = 0.99$.

smoothness in very high-resolution meshes, one may also increase it further. In the challenging interactive examples described before, λ should be lowered (for example to 0.7) if artifacts appear, but while $\lambda = 0.99$ seems to create problems for some extreme deformations, we do not observe the same for $\lambda = 0.95$, so one can usually still use high smoothness.

2.6. Linear Solves with Efficient Updates

Performing interactive deformations amounts to repeated solves of the linear system in Equation (10), which we abbreviate as $\mathbf{A}\mathbf{V}' = \mathbf{r}$. We regularize the system to remove its rank deficiency by encouraging solutions to stay close to the previous iteration:

$$\tilde{\mathbf{A}}\mathbf{V}' = \tilde{\mathbf{r}} \quad \text{with} \quad \tilde{\mathbf{A}} = \mathbf{A} + \epsilon\mathbf{I}, \quad \tilde{\mathbf{r}} = \mathbf{r} + \epsilon\mathbf{V}'_{\text{prev}}. \tag{11}$$

We opt for $\epsilon = 10^{-8}$, and $\mathbf{V}'_{\text{prev}}$ denotes the vertex positions from the last iteration. We would like to avoid re-factoring the matrix each time constraints are added or removed. Furthermore, we aim to achieve this in a way that is solver agnostic and does not rely on special functionality for up- and downdates [Davis and Hager 2009; Herholz and Sorkine-Hornung 2020]. To this end, we consider the n_d linear position constraints $\mathbf{H}\mathbf{V}' = \mathbf{C}$, where $\mathbf{C} \in \mathbb{R}^{n_d \times 3}$ contains handle positions and $\mathbf{H} \in \mathbb{R}^{n_d \times n}$ is a sparse matrix with a single 1 for each row selecting a constrained degree of freedom. We strive to find an approximate solution to Equation (11) subject to linear constraints. Introducing a matrix $\mathcal{L} \in \mathbb{R}^{n_d \times 3}$ of Lagrange multipliers, we arrive at the following Karush–Kuhn–Tucker (KKT) system:

$$\begin{pmatrix} \tilde{\mathbf{A}} & \mathbf{H}^\mathsf{T} \\ \mathbf{H} & 0 \end{pmatrix} \begin{pmatrix} \mathbf{V}' \\ \mathcal{L} \end{pmatrix} = \begin{pmatrix} \tilde{\mathbf{r}} \\ \mathbf{C} \end{pmatrix}. \tag{12}$$

We define the matrix

$$\mathbf{Q} = \tilde{\mathbf{A}}^{-1}\mathbf{H}^\mathsf{T}, \quad \mathbf{Q} \in \mathbb{R}^{n \times n_d}, \tag{13}$$

and get the following result from Gaussian elimination on the KKT system in Equation (12):

$$\begin{pmatrix} \tilde{\mathbf{A}} & \mathbf{H}^\mathsf{T} \\ 0 & -\mathbf{Q}^\mathsf{T}\mathbf{H}^\mathsf{T} \end{pmatrix} \begin{pmatrix} \tilde{\mathbf{V}}' \\ \mathcal{L} \end{pmatrix} = \begin{pmatrix} \tilde{\mathbf{r}} \\ \mathbf{C} - \mathbf{Q}^\mathsf{T}\tilde{\mathbf{r}} \end{pmatrix}. \tag{14}$$

The system is now solved in two steps. First, the small dense $n_d \times n_d$ system

$$-\mathbf{Q}^\mathsf{T}\mathbf{H}^\mathsf{T}\mathcal{L} = \mathbf{C} - \mathbf{Q}^\mathsf{T}\tilde{\mathbf{r}} \tag{15}$$

is solved to obtain the Lagrange multipliers $\mathcal{L}$, which we then use to solve for $\mathbf{V}'$.

This approach allows reuse of the initial factorization of $\tilde{\mathbf{A}}$. When constraining a new vertex, we can add a new column to $\mathbf{Q}$ by solving a linear system using the previously computed factorization. Removing a constraint means dropping the corresponding column in $\mathbf{Q}$. Note that this method enables fast updates as long as only a few dynamic constraints are present, as we have to solve the dense system in Equation (15), which scales with the number of constraints. For the Armadillo mesh shown in Table 1, the solve for the Lagrangian coefficients when having two constrained vertices takes 0.868 ms. For fifty constrained vertices, it already takes 8.26 ms. For the substitution approach on the other hand, the solve time goes down with the amount of constrained vertices. The presented mechanism with handle updates is thus particularly well suited for our smooth deformation energy, as it enables the use of small handles, namely point handles, without spiking artifacts.

For few constrained vertices, factorization and solving times are very similar for the substitution and updating approach. The main advantage of the updating method thus lies in the speedup of adding new handles, as can be seen in Table 1. While this requires re-factorization of the system matrix for the standard approach, the updating setting avoids this and efficiently updates the constraint matrices instead.

Mesh	# Vertices	Solver	Method	Factorization	Handle	Solve
Armadillo	172,974	Eigen	Standard	3,358	3,358.2	103.0
			Updating	3,341	27.5	104.7
		CHOLMOD	Standard	2,043	2,043.4	78.0
			Updating	2,146	44.9	78.9
Spot	2,930	Eigen	Standard	14	14.0	1.5
			Updating	8	0.8	1.6
		CHOLMOD	Standard	7	7.2	1.4
			Updating	7	1.8	1.4

Table 1. Example runtimes for factorization, adding handles, and solving for the usual substitution method and the one with efficient updates. Mesh complexity is given through the number of vertices, and runtime is in milliseconds. Both meshes have two handle vertices.

Other solvers substitute known degrees of freedom and solve a smaller system by discarding the linear equations associated with the constrained degrees of freedom. In contrast, our approach considers the energy at constrained vertices as well. Despite this discrepancy, we observed no visual differences in practice. Both methods are well established for minimizing constrained energies in deformation [Botsch and Sorkine 2008].

3. Results and Discussion

Using the presented method, we obtain deformation results that are both smooth and intuitive. In particular, local features rotate with the rest of the mesh, as is the case in the original ARAP method, and at the same time no spikes appear even with large deformations on single-point handles.

A comparison of different methods on a benchmark of standard examples from the deformation survey of Botsch and Sorkine [2008] can be found in Figure 10. The corresponding statistics, such as runtime and mesh size, are given in Table 2. It can be seen that our method produces very natural-looking results that don't buckle due to the increased smoothness when compared to standard ARAP. At the same time, the overall benefits of ARAP are preserved, such as the rotation of local features with the rest of the mesh. In terms of efficiency, it can be seen that for high-resolution meshes, the smooth ARAP method needs a little longer for factorization, as the bi-Laplacian system matrix is less sparse. However, our method is usually faster overall since the smoothness regularization we use shows quicker convergence in many cases. This is the case in all examples shown in Table 2, except for the bar example, as the result of ARAP stays very close to the bi-Laplacian initialization. When using the original mesh to initialize instead, the difference becomes clearer again: the standard method takes 0.72 seconds, while the smooth version only needs 0.45 seconds.

Mesh	Complexity		Method	Factorization	Solving	Iterations
	# Faces	# Vertices				
Knubbel	80,000	40,401	Original	0.248	2.44	61
			Smooth	0.253	1.90	44
Cylinder	9,600	4,802	Original	0.013	1.05	294
			Smooth	0.013	0.10	25
Cactus	10,518	5,261	Original	0.013	1.66	415
			Smooth	0.013	0.78	173
Bar	12,106	6,084	Original	0.020	0.17	41
			Smooth	0.023	0.42	89

Table 2. Mesh complexity through the number of triangles and vertices, runtime (in seconds), and convergence information of the survey benchmark examples shown in Figure 10.

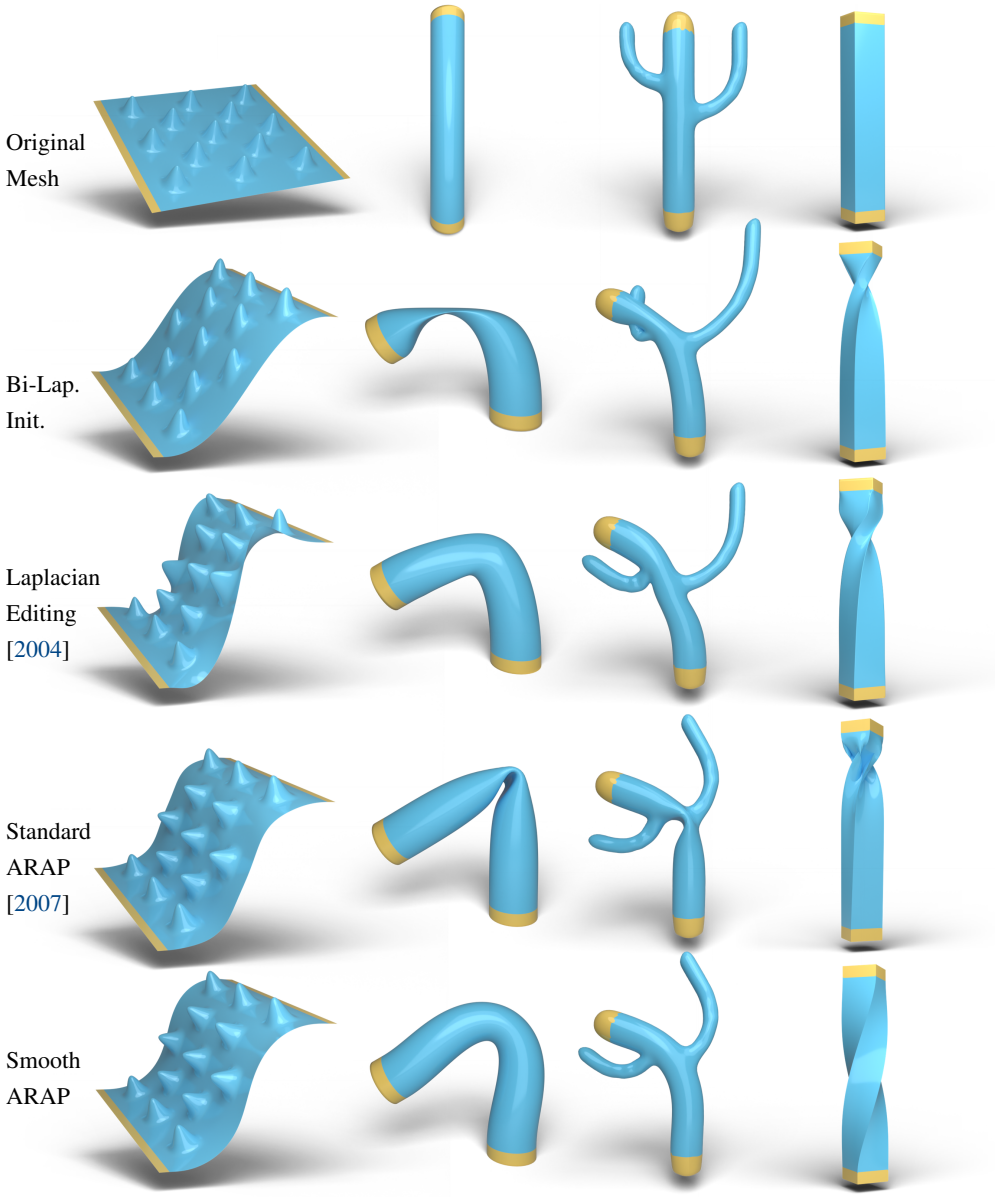

Figure 10. Comparison of different deformation methods using the standard survey examples [Botsch and Sorkine 2008]. The handles are marked in yellow and fixed to the prescribed positions from the survey benchmark; the depicted naive bi-Laplacian initialization is used to initialize both versions of ARAP. The results use smoothness coefficient $\lambda = 0.95$, and the method is run until convergence, which we set to be a relative change of the mesh $< 10^{-4}$.

Furthermore, we demonstrate the robustness of our method by showing the result of different possible initializations in Figure 11. Our smooth ARAP energy is able to produce good results for very complex deformations with a significant amount of local details even from very bad initializations. This can also be seen in Figure 12,

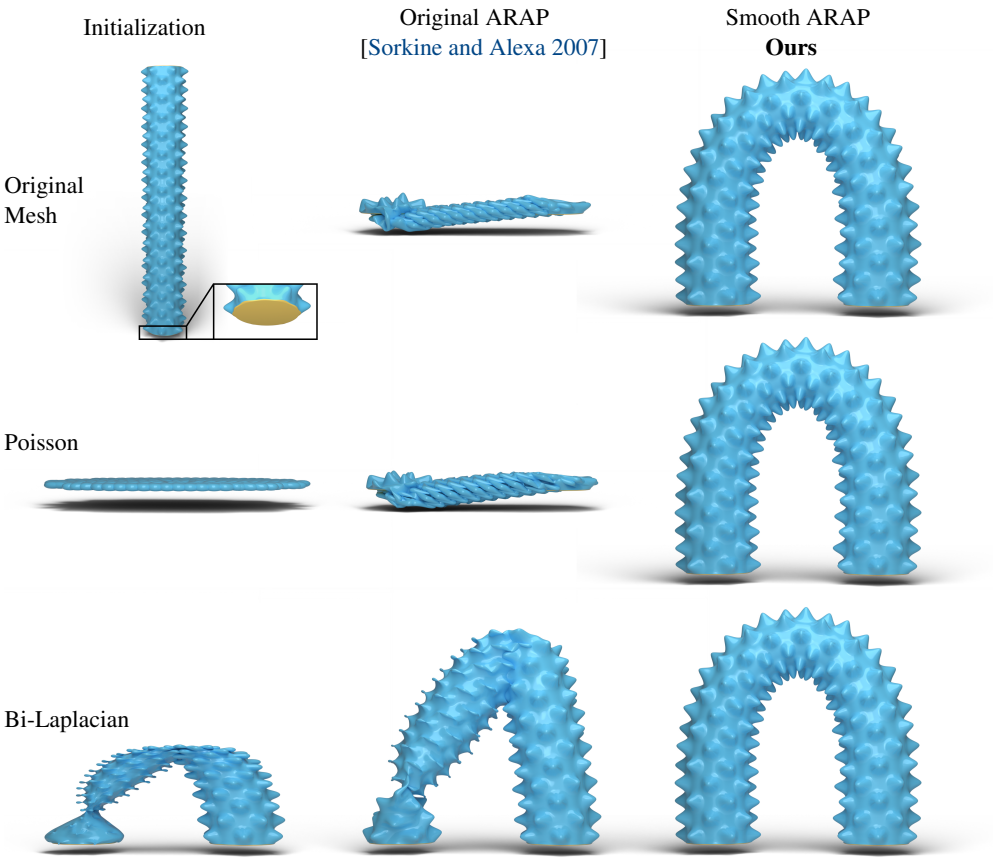

Figure 11. Robustness to initialization: A cylinder with bumps is deformed through standard and smooth ARAP using different initialization methods: namely, initializing via the original mesh and solving a Poisson or a bi-Laplacian equation with constrained handle positions. Solving the Poisson equation amounts to applying Poisson mesh editing [Yu et al. 2004] with the positional constraints and without rotation propagation, and solving the bi-Laplacian equation is applying Laplacian mesh editing without rotation handling [Sorkine et al. 2004]. The handles that were used are at the bottom and top of the cylinder and are again marked in yellow and are highlighted from a different perspective in the original mesh for better visibility.

where an unfortunate initial configuration of the dog is shown, from which our method manages to recover by rotating the entire mesh and achieving a plausible final result that is very different than its initialization.

Another disadvantage of our method is that it is no longer parameter-free; however, settling on λ is very easy. For most meshes, using any high λ such as 0.95 works very well, and it may be further increased to, for example, 0.999 should an even smoother result on a high-resolution mesh be desired. Only in very few challenging examples, λ should not be set too high, but it can simply be reduced, should the artifacts mentioned in Section 2.5 be observed.

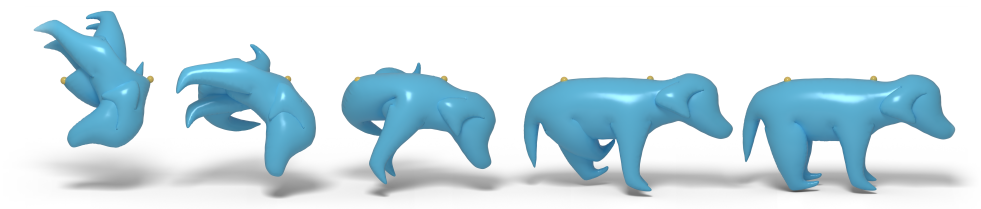

Figure 12. Robustness to bad configurations: A dog mesh is in a difficult configuration, but converges to a good result nevertheless, with some intermediate steps shown.

4. Conclusion, Limitations, and Future Work

We presented an adaptation of the classical As-Rigid-As-Possible deformation approach that produces smoother results at the constrained handles. The method benefits from the same ease of implementation as the original ARAP, with the key steps only being applying SVD in the local step and solving a sparse linear system in the global step. Our new method is robust to different initializations and is able to recover good solutions from bad starting configurations. The added smoothness term provides more natural results in many cases and ensures that single-point control handles can be used without introducing any artifacts. This is especially beneficial in interactive settings, as point handles are the easiest interaction mechanism for user-controlled deformations.

An inherited limitation we encounter from the standard ARAP is its mesh dependence. Specifically, the neighborhoods over which rotations are fitted depend on the triangulation, so that the shapes and sizes of the 1-ring neighborhoods affect the results to varying degrees. This issue has been recently tackled via intrinsic neighborhood construction using Voronoi cells [Finnendahl et al. 2023], and it would be an interesting avenue for future work to find a method for resolving mesh dependency without impacting the simplicity and efficiency of the deformation method.

Acknowledgements

This work was supported in part by the ERC Consolidator Grant No. 101003104 (MYCLOTH). We are grateful to Marcel Padilla for his insightful comments.

Index of Supplemental Materials

The code can be found under `https://github.com/oehria/smooth-arap`, with a publication snapshot at `https://jcgt.org/published/0014/01/10/smooth-arap.zip`. It contains a non-interactive implementation to recreate the survey examples with our method (see Figure 10), the interactive framework, and the code for the solver with efficient updates. Additionally, a video is provided that showcases the interactive application in use.

References

BOTSCH, M. AND SORKINE, O. On linear variational surface deformation methods. *IEEE Transactions on Visualization and Computer Graphics*, 14(1):213–230, 2008. URL: https://doi.org/10.1109/TVCG.2007.1054. 201, 210, 211

CHAO, I., PINKALL, U., SANAN, P., AND SCHRÖDER, P. A simple geometric model for elastic deformations. *ACM Transactions on Graphics*, 29(4):38:1–38:6, July 2010. URL: https://doi.org/10.1145/1778765.1778775. 202

DAVIS, T. A. AND HAGER, W. W. Dynamic supernodes in sparse Cholesky update/downdate and triangular solves. *ACM Transactions on Mathematical Software*, 35(4):27:1–27:23, February 2009. URL: https://doi.org/10.1145/1462173.1462176. 208

DVOROŽŇÁK, M., SÝKORA, D., CURTIS, C., CURLESS, B., SORKINE-HORNUNG, O., AND SALESIN, D. Monster Mash: A single-view approach to casual 3D modeling and animation. *ACM Transactions on Graphics (Proceedings of SIGGRAPH ASIA)*, 39(6):214:1–214:12, 2020. URL: https://doi.org/10.1145/3414685.3417805. 199, 207

FINNENDAHL, U., SCHWARTZ, M., AND ALEXA, M. ARAP revisited: Discretizing the elastic energy using intrinsic Voronoi cells. *Computer Graphics Forum*, 42(6):e14790, 2023. URL: https://doi.org/10.1111/cgf.14790. 213

HERHOLZ, P. AND SORKINE-HORNUNG, O. Sparse Cholesky updates for interactive mesh parameterization. *ACM Transactions on Graphics (Proceedings of SIGGRAPH Asia 2020)*, 39(6):202:1–202:14, 2020. URL: https://doi.org/10.1145/3414685.3417828. 208

JACOBSON, A., BARAN, I., KAVAN, L., POPOVIĆ, J., AND SORKINE, O. Fast automatic skinning transformations. *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH)*, 31(4):77:1–77:10, 2012. URL: https://doi.org/10.1145/2185520.2185573. 202

LEVI, Z. AND GOTSMAN, C. Smooth rotation enhanced as-rigid-as-possible mesh animation. *IEEE Transactions on Visualization and Computer Graphics*, 21(2):264–277, 2015. URL: https://doi.org/10.1109/TVCG.2014.2359463. 201

LIU, L., ZHANG, L., XU, Y., GOTSMAN, C., AND GORTLER, S. J. A local/global approach to mesh parameterization. *Computer Graphics Forum*, 27(5):1495–1504, 2008. URL: https://doi.org/10.1111/j.1467-8659.2008.01290.x. 202

MARTINEZ ESTURO, J., RÖSSL, C., AND THEISEL, H. Smoothed quadratic energies on meshes. *ACM Transactions on Graphics*, 34(1):2:1–2:12, December 2015. URL: https://doi.org/10.1145/2682627. 200, 201

SORKINE, O. AND ALEXA, M. As-rigid-as-possible surface modeling. In *Proceedings of EUROGRAPHICS/ACM SIGGRAPH Symposium on Geometry Processing*, SGP '07, pages 109–116. Eurographics Association, 2007. URL: https://dl.acm.org/doi/10.5555/1281991.1282006. 198, 199, 200, 202, 203, 204, 206, 211, 212

SORKINE, O. AND BOTSCH, M. Interactive shape modeling and deformation. In *Eurographics 2009—Tutorials*, pages 11–37. The Eurographics Association, 2009. URL: https://doi.org/10.2312/egt.20091068. 199

SORKINE, O., COHEN-OR, D., LIPMAN, Y., ALEXA, M., RÖSSL, C., AND SEIDEL, H.-P. Laplacian surface editing. In *Proceedings of the 2004 EUROGRAPHICS/ACM SIGGRAPH Symposium on Geometry Processing*, pages 179–188. ACM Press, 2004. URL: https://doi.org/10.1145/1057432.1057456. 205, 211, 212

YU, Y., ZHOU, K., XU, D., SHI, X., BAO, H., GUO, B., AND SHUM, H.-Y. Mesh editing with Poisson-based gradient field manipulation. *ACM Transactions on Graphics*, 23(3): 644–651, August 2004. URL: https://doi.org/10.1145/1015706.1015774. 212

Author Contact Information

Annika Oehri
Dept. of Computer Science
ETH Zurich
Switzerland
https://oehria.github.io/

Philipp Herholz
https://phherholz.github.io/

Olga Sorkine-Hornung
Dept. of Computer Science
ETH Zurich
Switzerland
https://igl.ethz.ch/people/sorkine/

Annika Oehri, Philipp Herholz, and Olga Sorkine-Hornung, Higher-Order Continuity for Smooth As-Rigid-As-Possible Shape Modeling, *Journal of Computer Graphics Techniques (JCGT)*, vol. 14, no. 1, 198–215, 2025
http://jcgt.org/published/0014/01/10/

Received:	2025-01-18		
Recommended:	2025-04-21	Corresponding Editor:	Eric Lengyel
Published:	2025-06-06	Editor-in-Chief:	Eric Haines