A Rotation-Strain Method to Model Surfaces using Plasticity

JIAHAO WEN, University of Southern California, USA
BOHAN WANG, University of Southern California, Massachusetts Institute of Technology, USA
JERNEJ BARBIČ, University of Southern California, USA

Modeling arbitrarily large deformations of surfaces smoothly embedded in three-dimensional space is challenging. The difficulties come from two aspects: the existing geometry processing or forward simulation methods penalize the difference between the current status and the rest configuration to maintain the initial shape, which will lead to sharp spikes or wiggles for large deformations; the co-dimensional nature of the problem makes it more complicated because the deformed surface has to locally satisfy compatibility conditions on fundamental forms to guarantee a feasible solution exists. To address these two challenges, we propose a rotation-strain method to modify the fundamental forms in a compatible way, and model the large deformation of surface meshes smoothly using plasticity. The user prescribes the positions of a few vertices, and our method finds a smooth strain and rotation field under which the surface meets the target positions. We demonstrate several examples whereby triangle meshes are smoothly deformed to large strains while meeting user constraints.

CCS Concepts: • Computing methodologies → Shape modeling, Physical simulation.

Additional Key Words and Phrases: surfaces, shape modeling, rotation, strain, deformation

1 INTRODUCTION

Modeling surfaces and their deformations is a central topic in computer graphics. In the context of surface modeling, a surface representing the shape needs to be stretched, sheared, or rotated to meet arbitrary user constraints. Users intuitively expect that the deformation should always preserve the intrinsic original shape of the object, while being globally smooth. To achieve this goal, standard geometric shape modeling methods and physically-based simulation methods penalize the changes between the current status and the rest configuration in a smooth manner to maintain the origin shape. Given point constraints, however, these standard methods either produce spikes or excessive curvatures under large deformation. To address this, plastic deformation can be used because it changes the rest shape directly. Wang et al. [Wang et al. 2021] used plastic strains to deform volumetric meshes of template human organs to match medical images of a real subjects, producing smooth results satisfying sparse position constraints. At a first glance, it seems that applying such methodology—i.e., the plastic deformation method—directly onto surface is straightforward. However, this is not the case and we notice large difficulties therein. When we talk about the surface mesh deformation, we are referring to embedding a 2D manifold into 3D space, which is a co-dimensional problem compared to a 3D volumetric mesh. Therefore, the plasticity Degrees of Freedom (DoFs) contain intrinsic (in-plane) and extrinsic parts (bending). Moreover, given some arbitrarily defined local plasticity, we must ensure that such a surface even exists (at least locally), so that we can reconstruct the mesh vertex positions by solving for a static equilibrium under the given plastic strains. In other words, the definition of intrinsic and extrinsic plasticity must satisfy compatibility conditions on fundamental forms. To achieve these requirements, we propose a novel differential representation of surface shape deformation, which models the local surface deformation using a $3 \times 3$ rotation and a $2 \times 2$ symmetric matrix. The rotation globally orients every local surface “patch” in the world, and the symmetric matrix models an arbitrarily oriented (large) strain of the surface. This new representation is very versatile and can easily model large spatially varying surface strains.

The representation can be seen as an extension of rotation-strain coordinates [Huang et al. 2011] from solids in $\mathbb{R}^3$ to surfaces embedded into $\mathbb{R}^3$. It automatically makes the parameterization gradient locally integrable, i.e., the surface locally satisfies the Gauss-Codazzi equations [Chen et al. 2018a] and therefore locally exists. This greatly improves the convergence of shape optimization. To use our representation to perform geometric shape modeling, we define an optimization problem that optimizes for smooth spatially varying $3 \times 3$ rotations $R$ and $2 \times 2$ symmetric matrices $\delta$ so that the resulting surface matches the user constraints as closely as possible. Our optimization successfully handles difficult yet sparse point constraints in a few different examples and produces smooth output shapes under large and anisotropic shape deformation.
we removed the landmarks at the base of the ears and kept only a single (a), observe how the stretched ear grows to meet the landmark, while tweaking the parameters of ShapeOp did not improve the result. In case (b), observe how the stretched ear grows to meet the landmark, while preserving its inherent geometric shape (the ears are still curled). In case (b), we removed the landmarks at the base of the ears and kept only a single ear landmarked. As expected, the single landmark now causes the entire bunny to uniformly grow in size.

Figure 2. Our method scales the object in a manner that preserves the shape: Observe that ShapeOp [Bouaziz et al. 2014] produce suboptimal results, under identical inputs as our method (a): the bunny has fixed landmarks (red dots) at the bunny’s bottom and the base of the ears, and a landmark on each ear which is positioned to command the ear to stretch. Tweaking the parameters of ShapeOp did not improve the result. In case (a), observe how the stretched ear grows to meet the landmark, while preserving its inherent geometric shape (the ears are still curled). In case (b), we removed the landmarks at the base of the ears and kept only a single ear landmarked. As expected, the single landmark now causes the entire bunny to uniformly grow in size.

demonstrate that these effects are difficult to achieve with previous methods. Our contributions are mostly two-fold:

- To the best of our knowledge, we are first to use plasticity and FEM to successfully model surface meshes undergoing large strains and rotations, specified by point constraints on the boundary of the object. Through plasticity, we can generate results that are impossible to obtain using elastic deformation, such as bending and inflating an object. Compared to previous works which mainly penalize the deformation magnitude, our method additionally defines “smoothness” energy on plasticity DoFs so that we can generate smooth results. By experiments, our method succeeds in the extreme benchmarks and produces expected results that previous methods cannot achieve.

- We propose a new Rotation-Strain method to modify the first and second fundamental forms of a (discretized) surfaces embedded into the 3-dimensional space. Compared to [Chen et al. 2018a] and [Kircher and Garland 2008], our method can cleanly decouple the intrinsic and extrinsic DoFs, thus guaranteeing compatibility for each local patch (triangle in discrete configuration), which is proved to be crucial for decreasing the objective energy.

2 RELATED WORK

Modeling shape deformation is an important topic in computer graphics and encountered in many sub-disciplines, including geometric modeling [Alexa et al. 2006; Botsch and Sorkine 2008], physically based FEM simulation [Sifakis and Barbič 2015] and mesh non-rigid registration [Allen et al. 2003]. In general, a shape deformation problem can be treated as an optimization problem whose objective function is defined as the “smoothness” of the shape, combined with some user constraints.

Different definitions of the “smoothness” energies give rise to various output properties. The smoothness of the shape has been formulated through variational methods [Botsch and Kobbelt 2004; Wang et al. 2015], Laplacian surface editing [Sorkine et al. 2004], as-rigid-as-possible (ARAP) deformation [Igarashi et al. 2005; Sorkine and Alexa 2007], coupled prisms (PRIMO) [Botsch et al. 2006], and partition-of-unity interpolation weights such as bounded bi-harmonic weights (BBW) [Jacobson et al. 2011] and quasi-harmonic weights [Wang and Solomon 2021]. Nonetheless, existing methods produce suboptimal deformations when the shape undergoes large rotations and large strains simultaneously (Figure 2), or they are not designed for meeting user vertex constraints. Variational methods suffer from large rotations, while ARAP and PRIMO produce spiky shapes given difficult constraints. These problems have been discussed and illustrated by [Botsch and Sorkine 2008; Wang et al. 2021]. BBW generates smooth interpolation weights that can be used for interpolating transformations across the entire shape, but it takes a substantial amount of time to compute the weights. In addition, to generate interpolation weights, the handles must be predefined. To overcome the problem caused by large rotations in variational methods, the smoothness energy has been defined to penalize the Laplacian after the rotations [Bouaziz et al. 2012, 2014]. While the resulting shapes are C2-smooth, the method generates wiggly artifacts similar to variational methods [Wang and Solomon 2021]. To address this problem and handle large strains properly, the rest shape can be reset during each deformation iteration [Gilles and Magnenat-Thalmann 2010; Gilles et al. 2006; Schmid et al. 2009]. Doing so, however, loses important features of the original input shape, such as not being able to preserve the volume and the sharp features. In contrast, our method can solve these problems and handle large rotations and large strains correctly.

When performing non-rigid registration on a template mesh to match the target shape, a smoothness energy is often needed in the objective function. Existing methods penalize the affine transformations between neighboring vertices or triangles [Allen et al. 2003; Amberg et al. 2007; Li et al. 2008]. Such energy is combined with dense correspondences to achieve deforming the template mesh to match the target. When the smoothness energy is combined with sparse inputs, however, artifacts similar to those in variational methods may appear, as demonstrated by [Wang et al. 2021]. Deformation transfer is able to handle sparse markers without visible artefacts [Sumner and Popović 2004], but it cannot handle large rotations. Kircher et al. solved the problem by decomposing the affine transformations into the rotational and shear/stretch parts, and treated them separately [Kircher and Garland 2008].

Physically-based FEM simulation can also be used to deform the shape. An object can either be treated as volumetric object or surface object. For volumetric objects, the elastic energy penalizes volume changes, large strains, and gives C0 continuity around constraints [Sifakis and Barbič 2015], similar to ARAP. For surface objects, they are modeled using cloth simulation. The cloth elastic energy is decoupled into two terms, namely, in-plane elastic energy and bending energy. The in-plane elastic energy is defined in a similar fashion to the elastic energy for 3D volumetric objects [Volino et al. 2009], and, as a result, it shares the same artifacts as volumetric simulation. On the other hand, the bending energy is modeled as the change of curvature. In fact, penalizing the bi-Laplacian mentioned above in variational methods can also be considered as a
type of a bending energy, as it models the change of mean curvature. For a comprehensive comparison of bending models, we refer readers to [Chen et al. 2018a]. In general, elastic models are always large-strain-unfriendly, because they penalize the changes to the rest shape.

Plastic deformation can be used to solve such problems, because it models shapes that undergo permanent large deformation and therefore it is in principle a natural choice to do shape modeling. However, plastic deformation is only widely used in forward simulation [Baraff 2001; Chen et al. 2018b; Irving et al. 2004; Müller and Gross 2004; O’Brien et al. 2002; Stomakhin et al. 2013]. Wang et al. [Wang et al. 2021] introduce plastic deformation to shape modeling, but they only addressed volumetric objects (tetrahedral meshes in 3D), which limits the applications. In our work, we address surfaces in 3D. Our definition of DoFs makes it possible to handle both rotations and anisotropic stretching, while their method only permits symmetric $3 \times 3$ plastic transformations, i.e., (anisotropic) stretching only. Because we formulate the plastic deformations on the surface object, our method uses fewer DoFs, which leads to faster performance. Last but most important, presence of the co-dimension makes the problem substantially more difficult, due to the necessity to model compatible first and second fundamental forms. [Lipman et al. 2005] proposed a rotation-invariant representation of surface meshes using discrete frames and fundamental form coefficients. It decouples frame rotation from the deformation of the tangent space and solves them in two separate steps. However, it doesn’t guarantee the compatibility of fundamental forms. [Wang et al. 2012] used edge lengths and dihedral angles as primary variables to compute discrete fundamental forms, and derived local and global compatibility conditions. However, their method penalizes the distance between the current shape to the compatible initial shape in the reconstruction of the surface similarly, and will still suffer from sharp spikes under large deformation (Figure 7). [Chern et al. 2018] discusses the isometric immersion problem, taking as input an orientable surface triangle mesh annotated with edge lengths only and outputting vertex positions. [Martinez Esturo et al. 2014] penalizes the magnitude of energy gradients over the whole mesh and achieves smooth shapes compared to ARAP, however, as discussed in [Wang et al. 2021], such methods suffer from wiggles under large deformations. Although some previous methods discussed different ways to discretize fundamental forms and derive the compatibility of surface meshes, it is very hard to define plasticity and the corresponding smoothness based on their definitions. That is why we choose to follow [Chen et al. 2018a].

Parameterizing plastic deformation on a surface is not straightforward: an incorrect choice of the model will produce degenerate outputs and non-convergence of optimization. To achieve our goal, we first choose a state-of-the-art thin-shell simulation model [Chen et al. 2018a]. It not only defines a physically-accurate simulation, but also gives a proper space to define plastic deformations. We also experimented with other elastic cloth models such as [Grinspun et al. 2003], but the plastic deformation is limited by the elastic model and causes artefacts (Figure 4). Methods that define the plastic deformation on thin-shells are rare. We first attempted to use the plastic model defined in [Chen et al. 2018a], but the optimization failed to converge even on a simple input. Directly decomposing a plastic deformation gradient into a rotation and a symmetric matrix is also not doable in practice [Kircher and Garland 2008] (Section 3.1).

### 3 PLASTICITY OF SURFACES

Before describing our method for shape modeling of surfaces, we give a brief review of the surface theory in differential geometry, following [Chen et al. 2018a].

#### 3.1 Continuous Formulation

**Thin-Shell Representation.** Our modeling of surfaces stems from the continuous mechanics where a surface is modeled as a thin shell $S \in \mathbb{R}^3$ of thickness $h > 0$, parameterized over a domain $\Omega \subset \mathbb{R}^2$ and an embedding $\phi : \Omega \times [-h/2, h/2] \to \mathbb{R}^3$, with $S$ the image of $\phi$. By the Kirchhoff-Love assumption, the entire shell volume can be represented only in terms of the shell’s mid-surface $r : \Omega \to \mathbb{R}^3$,

$$\phi(u, v, t) = r(u, v) + tn(u, v),$$  

where $n = (r_u \times r_v) / ||r_u \times r_v||$ is the midsurface normal, $r_u = \partial r / \partial u$, $r_v = \partial r / \partial v$, and $t \in [-h/2, h/2]$. The first and second fundamental forms of the surface are

$$a = F^TF, \quad b = -N^TF, \quad \text{where}$$

$$F = [r_u \ r_v], \quad \text{and} \quad N = [n_u \ n_v].$$

In this paper, we use $\tilde{a}$ and $\tilde{b}$ to denote the first and second fundamental forms in the initial rest state. We assume that the parameterization is non-degenerate, i.e., matrix $F$ is rank 2.

**Plasticity.** We consider that a surface changes its shape through “plasticity” when it is assigned different first and second fundamental forms $\tilde{a}$ and $\tilde{b}$ (at each location on the surface) from the initial rest state (Figure 3). Chen et al. [Chen et al. 2018a] employed plasticity for thin-shell forward simulation purposes, by directly modifying $\tilde{a}$ and $\tilde{b}$ into $\tilde{a}$ and $\tilde{b}$, using some procedural formulas. However, such an approach runs into a substantial limitation that was readily apparent in our shape deformation system: arbitrary $\tilde{a}$ and $\tilde{b}$ may not satisfy the local compatibility conditions (Gauss-Codazzi equations [Weischedel 2012]). In other words, given arbitrary first and second fundamental forms $\tilde{a}$ and $\tilde{b}$, we can’t guarantee that such a surface exists, not even locally. According to our experiments, setting $\tilde{a}$ and $\tilde{b}$ without regarding to compatibility leads to poor search directions and inability to decrease the energy for solving shape deformation (Section 4). Therefore, what is needed is a new approach that modifies the first and second fundamental forms in a compatible manner. For each parameter $\sigma \in \Omega$ defining a surface point $r(\sigma)$, the vectors $t_1(\sigma) = r_u \quad t_2(\sigma) = r_v$ span the tangent plane. The unit normal at $r(\sigma)$ can be computed as $n = (t_1 \times t_2) / ||t_1 \times t_2||$.

Then, we can define a non-degenerate local frame at every point $\sigma$
We represent the symmetric matrix $\overline{S}$, which means that $\overline{S} = S$. Therefore, the first and second fundamental forms $\overline{R}$, $\overline{D}$ are defined by
definitions\[ D = \begin{bmatrix} t_1 & t_2 & n \end{bmatrix} \]Our idea is to modify the local frame at every point using a $3 \times 3$ rotation matrix $\tilde{R}(\sigma)$ and a $2 \times 2$ symmetric matrix $S(\sigma)$, leading to
definitions\[ D' = \tilde{R} \begin{bmatrix} t_1 & t_2 & n \end{bmatrix} \begin{bmatrix} S & 0 \\ 0 & 1 \end{bmatrix} \]Intuitively, we (anisotropically) scale the UV space at point $\sigma$ using $S$ and then change the local curvature by a spatially varying rotation matrix $\tilde{R}$. Because $(\tilde{R}t_1) \times (\tilde{R}t_2) = \tilde{R}(t_1 \times t_2)$, the normal in the new frame $D'$ will always automatically be perpendicular to the tangent plane, which is spanned by the columns of matrix $\tilde{R}(t_1, t_2)S$. The new surface can be locally parameterized via $\zeta \mapsto r'(\zeta) = \tilde{R}(\sigma)r(\sigma + s \zeta)$; By derivation, one can verify that
definitions\[ F' = [r_{u}' r_{v}'] = RFS, \quad n' = Rn, \]
which means that $D'$ is the local frame of the new embedding $r'$. Therefore, the first and second fundamental forms $\tilde{a}$ and $\tilde{b}$ derived from $r'$ are automatically compatible, since they correspond to an actual locally defined surface. Note that $S$ is an intrinsic variable which is unrelated to the embedding of the surface in $\mathbb{R}^2$, while $R$ is extrinsic, defining the embedding in $\mathbb{R}^2$. Our method cleanly decouples the intrinsic and extrinsic DoFs, so that $S$ and $R$ are independent of each other. Observe that $R$ has 3 DoFs because it is a rotation matrix in $\mathbb{R}^2$, and $S$ also has 3 DoFs because it is a symmetric matrix; and therefore the space of local surface modifications is 6-dimensional. In our implementation, we use the exponential map and the Rodrigues’ rotation formula to parameterize $R$ into a 3-dimensional vector $\theta$. We also represent the symmetric matrix $S$ as a 3D vector $s$.

**Difference to Polar Decomposition.** We note that there is an alternative approach to change the local frame. Namely, one can perform polar decomposition of $D$ directly by $D = U A$, where $U$ is a $3 \times 3$ rotation matrix and $A$ is a $3 \times 3$ symmetric matrix [Kircher and Garland 2008]. The new frame then becomes $D' = U'A'$. Such an approach is less elegant for surface modeling because the change in the UV space follows the transformation $T = A'A^{-1}$, which is not a symmetric matrix, i.e., not a pure scaling matrix. Therefore, the transformation $T$ includes a rotational component in the UV space, which should enter into $U'$. Thus, this decomposition can’t decouple the DoFs cleanly as we expected. Also, according to our experiments, one cannot obtain an effective search direction with this method to decrease the objective.

**Elastic Energy.** The second component of our modeling system is elastic. Similarly to [Chen et al. 2018a], for simplicity, we assume that the shell’s material is homogeneous and isotropic, and adopt the St. Venant-Kirchhoff thin-shell constitutive law. We note that most of the in-plane deformation is already handled by the plasticity in our method, and therefore the elasticity only undergoes small in-plane strain, justifying the choice of St. Venant-Kirchhoff. The elastic energy density can then be approximated ([Weischedel 2012], [Chen et al. 2018a]) up to $O(h^3)$ by
definitions\[ W(u, v) = \frac{h}{4}||\tilde{a}^{-1}a - I||_{SV}^2 + \frac{h^3}{12}||\tilde{a}^{-1}(b - \bar{b})||_{SV}^2 \sqrt{\det \tilde{a}} \]where $|| \cdot ||_{SV}$ is the “St. Venant-Kirchhoff norm”
definitions\[ ||M|| = \frac{c_1}{2}tr^2 M + c_2tr(M^2), \]
and $c_1, c_2$ are material parameters related to Young’s modulus $E$ and Poisson’s ratio $\nu$.
definitions\[ c_1 = \frac{E\nu}{1 - \nu^2}, \quad c_2 = \frac{E}{2(1 + \nu)} \]
The elastic energy of the entire surface is
definitions\[ V[r] = \int_{\Omega} W(u, v)dudv. \]
Note that we first attempted to use the hinge-based energy to define the bending energy [Grinspun et al. 2003], but the results contained substantial artifacts and the resulting deformation was not invariant of discretization, as shown in Figure 4. Using the energy defined based on fundamental form energy produced no such bias [Chen et al. 2018a]. Therefore, we abandoned the hinge-based energy.

### 3.2 Discretization

We approximate the mid-surface $r$ with a triangle mesh with $n$ vertices and $m$ triangles. In the following paragraphs, we use bold symbol to represent the quantities for the entire mesh and non-bold text to represent the quantities for a single vertex or element. The positions of the vertices $\mathbf{x} = \{x_1, \ldots, x_n\} \in \mathbb{R}^3$ take the place of the embedding function $r$. We assume that the first and second fundamental forms are constant over each triangle. We can now give a discrete form for Equations 2.7.

**Discrete Shell Model.** Let $f_{ijk}$ be a triangle with vertices $x_i, x_j, x_k$, and let $T$ be the canonical unit triangle with vertices $(0, 0), (1, 0), (0, 1)$. Then, locally the face $f_{ijk}$ is embedded by the affine function
definitions\[ r_{ijk} : T \rightarrow \mathbb{R}^3, \quad r_{ijk}(u, v) = x_i + u(x_j - x_i) + v(x_k - x_i). \]
The Euclidean metric on \( f_{ijk} \) pulls back to the first fundamental form
\[
a_{ijk} = \begin{bmatrix} ||x_j - x_i||^2 & (x_j - x_i) \cdot (x_k - x_i) \\ (x_j - x_i) \cdot (x_k - x_i) & ||x_k - x_i||^2 \end{bmatrix}. \tag{12}
\]
The 2 \( \times \) 2 matrix \( a_{ijk} \) is always positive semi-definite. The second fundamental form can be discretized as
\[
b_{ijk} = \begin{bmatrix} (n_i - n_j) \cdot (x_i - x_j) & (n_i - n_j) \cdot (x_i - x_k) \\ (n_i - n_k) \cdot (x_i - x_j) & (n_i - n_k) \cdot (x_i - x_k) \end{bmatrix}, \tag{13}
\]
where \( n_i \) is the mid-edge normal on the edge \( e_i \) opposite to vertex \( i \) [Chen et al. 2018a]. If \( e_i \) is a boundary edge, \( n_i \) equals to the face normal, otherwise, \( n_i \) is the average of the normals of the two faces incident at \( e_i \). Matrix \( b_{ijk} \) is always symmetric but not in general positive-definite. We can now give a discrete formulation of the elastic energy
\[
W_{ijk} = \left( \frac{h}{8} \right) ||\tilde{a}_{ijk} a_{ijk} - I ||_{SY}^2 + \frac{h^3}{24} ||\tilde{a}_{ijk} (\hat{b}_{ijk} - b_{ijk}) ||_{SY}^2 \sqrt{\det \tilde{a}_{ijk}}. \tag{14}
\]
where an additional division by two is due to the canonical triangle \( T \) having area \( \frac{1}{2} \).

**Discrete Plasticity.** Let \( \tilde{x} \) represent the mesh positions in the rest state, and suppose the mesh at some location locally undergoes a change \( (R, S) \) as described earlier. Then, by Equation 5, the plastic first fundamental form of the deformed surface corresponding to \((R, S)\), in the discrete setting, is
\[
\tilde{a}_{ijk} = S^T \begin{bmatrix} ||\tilde{x}_j - \tilde{x}_i||^2 & (\tilde{x}_j - \tilde{x}_i) \cdot (\tilde{x}_k - \tilde{x}_i) \\ (\tilde{x}_j - \tilde{x}_i) \cdot (\tilde{x}_k - \tilde{x}_i) & ||\tilde{x}_k - \tilde{x}_i||^2 \end{bmatrix} S. \tag{15}
\]
The second fundamental form is
\[
\hat{b}_{ijk} = 2S^T \begin{bmatrix} (\bar{n}_i - \bar{n}_j)^T R (\tilde{x}_j - \tilde{x}_i) & (\bar{n}_i - \bar{n}_j)^T R (\tilde{x}_k - \tilde{x}_i) \\ (\bar{n}_i - \bar{n}_k)^T R (\tilde{x}_i - \tilde{x}_j) & (\bar{n}_i - \bar{n}_k)^T R (\tilde{x}_i - \tilde{x}_k) \end{bmatrix} S, \tag{16}
\]
where \( \bar{n}_i \) represents the modified mid-edge normal of the plastic configuration after spatial varying rotations have been applied to triangle \( f_{ijk} \) and its neighboring triangles in the initial state.

## 4 PLASTIC-ELASTIC SURFACE SHAPE DEFORMATION

Given an input triangle mesh, our goal is to deform the input mesh to match the user-defined target points (landmarks), while ensuring smooth deformation (Figure 5). The position-based landmark constraints are freely defined by the user, whereby a selected point on an initial surface is manually corresponded to a target location. As explained in Section 3, to model plasticity, we define a \( 3 \times 3 \) rotation matrix and a \( 2 \times 2 \) symmetric scaling matrix for each triangle. After the parameterization, each triangle contains 6 DoFs given by \( \theta, s \), respectively. Finally, we group the DoFs of all triangles into a global vector \( p \in \mathbb{R}^{6m} \).

### 4.1 Computing Vertex Positions from Plastic Strains

Combining \( p \) with initial status \( \tilde{a}, \tilde{b} \) produces the current modified plasticity \( \tilde{a}, \tilde{b} \). Then, minimizing the elastic energy corresponding to \( \tilde{a}, \tilde{b} \) gives the vertex positions \( x \). To uniquely determine the local minimum, we define a proper boundary condition to eliminate the singularity of the elastic energy caused by its invariance on global rotation and translation. To achieve this, we freeze a few selected vertices, which is commonly done in the existing methods. In our implementation, if the landmarks already include anchored (i.e., permanently fixed) vertices, we simply select those vertices and treat them as fixed constraints. If the landmarks don’t include any anchored vertices, we fix 3 arbitrary vertices which are not intended to move during the optimization. Therefore, the construction of \( x \) can be formulated as
\[
\arg\min_x E_e(x, p) + E_c(x) \tag{17}
\]
where \( E_e = \sum_{ijk} W_{ijk} \) is the total discrete elastic energy, \( E_c = ||Cx - d||^2 \) is the quadratic position-based constraint energy. \( C \) is a constant selection matrix and \( d \) is the fixed position vector. Minimizing Equation 17 is equivalent to solving for the stationary point of:
\[
f_e(x, p) + f_c(x) = 0, \tag{18}
\]
where \( f_c = \frac{\partial E_c}{\partial x} \) is elastic force, and \( f_e = \frac{\partial E_e}{\partial x} \) is constraint force.

### 4.2 Shape Deformation Using Plastic Strains

We would like to discover vertex positions \( x \) and a smooth change of plastic strains \( p \) such that the surface meets user-defined constraints:
\[
\arg\min_{p, x} ||Lp||^2 + \alpha E_l(x) + \beta ||p - p_0||^2 \tag{19}
\]
s.t. \( f_e(p, x) + f_c(x) = 0. \tag{20}\)
where \( L \) is the Laplacian operator on the surface mesh, and \( p_0 \) is the initial value of \( p \), i.e., identity matrices for \( s \) and zeros for \( \theta \). The Laplacian term enforces the smoothness of \( p \), i.e., the change...
We now explain how we compute the Laplacian $L_i$. Where

$$\sum_{j \in N(i)} \text{mat}(s_i) - Q_{ji} \text{mat}(s_j) Q_{ji}^T,$$

(21)

where $N(i)$ represents the triangles adjacent to triangle $i$ and $\text{mat}(s)$ converts the vector $s$ back to its matrix form.

The rotation matrix $R$ is always given in the global coordinate system, and hence no special treatment is required; we directly apply the triangle-wise Laplacian:

$$L_{\theta,i} = \sum_{j \in N(i)} \theta_i - \theta_j,$$

(22)

where $\theta = \log(R) \in \mathbb{R}^3$ is the exponential map representation of the rotation $R$. It should be noted that this measurement is not invariant of global rotation. However, our rotation field is local with respect to the rest shape. Due to performance considerations, we use this simple method and found that it works well in all the benchmarks and test cases. If global rotation is needed, we can insert an additional first step, namely first performing rigid Iterative Closest Point (ICP) to align the orientation of the object.

4.4 Optimization

To solve our optimization problem, we employ a Gauss-Newton solver in the same way in [Wang et al. 2021]. To improve the convergence of Gauss-Newton solver, we update the rest shape by the output vertex positions of the previous iteration at each Gauss-Newton iteration. To compensate for this update, we also re-initialize $p$ to $p_0$ at the beginning of each iteration. In our implementation, we use inexact Hessian [Chen et al. 2018a] and project local Hessian to the cone of symmetric positive semi-definite matrices prior to assembly to improve the performance.

5 RESULTS

We give several examples that demonstrate shape deformation involving large rotations and strains while meeting non-trivial user constraints (Figures 7,8,9). These examples also provide comparisons to ARAP [Sorkine and Alexa 2007], Primo [Botsch et al. 2006], [Wang et al. 2012], and ShapeOp [Bouaziz et al. 2014]. We also test our model using standard shape deformation benchmark [Botsch and Sorkine 2008], as shown in Figure 10. Figure 11 gives another interesting application. Table 1 gives the performance of each demo.

Table 1. The setup and performance of each example. The first half of the table shows the number of vertices ($\#vtx$), the number of triangles ($\#tri$), and the number of landmarks ($\#land$). In addition, the table also lists the performance of each example. Column $\#iter$ denotes the number of Gauss-Newton iterations performed in each example. Column $\ell_{opt}$ gives the errors of landmark constraints, given that the diagonal length of the bounding box of the mesh is $100$ for all examples. Finally, column $\ell_{err}$ shows the optimization time cost per iteration. We also measured the time cost per iteration of the volumetric method from [Wang et al. 2021] for the muscle example. For muscle, it takes $5$% per iterations and $4$ iterations in total, $6x$ slower per iteration and $3x$ slower in total than our method.

| Model   | $\#vtx$ | $\#tri$ | $\#land$ | $\#iter$ | $\ell_{opt}$ | $\ell_{err}$ |
|---------|---------|---------|----------|----------|-------------|-------------|
| leaf    | 233     | 411     | 16       | 13       | 0.3s        | 0.06s       |
| plane   | 441     | 800     | 17       | 6        | 0.5s        | 0.15s       |
| squarespike | 441 | 800     | 84      | 13       | 0.7s        | 0.35s       |
| sphere  | 642     | 1,280   | 6        | 5        | 0.6s        | 0.57s       |
| arch    | 1,756   | 3,508   | 200      | 13       | 3.6s        | 0.22s       |
| torus   | 2,304   | 4,608   | 50       | 5        | 2.4s        | 0.11s       |
| bunny   | 2,503   | 4,968   | 9        | 11       | 3.4s        | 0.19s       |
| muscle  | 3,288   | 6,572   | 650      | 9        | 9.2s        | 0.21s       |
| cylinder| 4,802   | 9,600   | 682      | 20       | 11.6s       | 0.01s       |
| cactus  | 5,261   | 10,518  | 864      | 14       | 15.5s       | 0.02s       |
| twist bar | 6,084 | 12,106  | 962      | 15       | 20.7s       | 0.15s       |
| horse   | 8,451   | 16,843  | 21       | 9        | 8.5s        | 0.16s       |
| fish    | 10,786  | 21,568  | 26       | 18       | 11.1s       | 0.14s       |

6 CONCLUSION

We gave a new differential representation for surface deformation that enables modeling shape deformations consisting of large spatially varying rotations and strains. Our formulation could be extended to other manifolds embedded in higher-dimensional spaces, for example, curves in three dimensions; or, more generally, $m$-dimensional manifolds embedded in $\mathbb{R}^n$, for $m < n$. If local rotations across the surface accumulate beyond $2\pi$, the spatially varying rotation field may have a discontinuity where the rotation “jumps” by a multiple of $2\pi$. While we did not run into this issue in our examples, this is a well-known problem in modeling rotational fields on surfaces, and there are standard solutions for it. While our method produces quality shapes that contain large strains and rotations and that precisely meet user constraints, the price to pay for this is that the method runs offline and is not interactive. This is because our method needs to solve an optimization problem for the plastic strains, which necessitates solving large sparse systems of
Fig. 7. **Large strain editing of a bending tube**: Similar to [Wang et al. 2021], we use landmark (red dots) to grow a bending tube in the middle part. Our method can produce a symmetric and smooth "arch". The result of shapeOp is not as smooth as ours. The Primo and ARAP suffer from sharp spikes. We tuned the parameters of both ShapeOp and Primo and tried various settings and the results weren’t improved.

Fig. 8. **Comparison to ARAP [Sorkine and Alexa 2007] and ShapeOp [Bouaziz et al. 2014]**: We apply a single landmark (red dot) to add a "hump" onto the horse and another landmark to lift the tail slightly, while preserving the remaining shape. Our method produces a smooth large deformation. ARAP produces a spike, and ShapeOp distorts a hoof, generates a strange stretched belly and produces a non-smooth hump. We tweaked the ShapeOp parameters and tried multiple settings; a better result could not be produced.

Fig. 9. **Comparison to [Wang et al. 2012]** Wang et al. [Wang et al. 2012] does not address large strain editing; it produces spikes at the constraints, as shown in their result. Our method produces smooth large deformations.

Further speedups could be obtained using multigrid or specially designed preconditioners for our problem.

**ACKNOWLEDGMENTS**

This research was sponsored in part by NSF (IIS-1911224), USC Annenberg Fellowship to Jiahao Wen and Bohan Wang, Bosch Research and Adobe Research.

**REFERENCES**

M. Alexa, A. Angelidis, M.-P. Cani, S. Frisken, K. Singh, S. Schkolne, and D. Zorin. 2006. Interactive Shape Modeling. In ACM SIGGRAPH 2006 Courses. 93.

Brett Allen, Brian Curless, and Zoran Popović. 2003. The Space of Human Body Shapes: Reconstruction and Parameterization from Range Scans. *ACM Trans. Graph.* 22, 3 (2003), 587–594.
