Contents lists available at ScienceDirect

Computers & Graphics

journal homepage: www.elsevier.com/locate/cag

Inpainting Holes In Folded Fabric Meshes

Guillaume Gisbert, Raphaëlle Chaine, David Coeurjolly

Université de Lyon, INSA Lyon, CNRS, UCBL, LIRIS, UMR5205, F-69621 Villeurbanne, France

ARTICLE INFO

Article history:
Received June 5, 2023

Keywords: Hole Filling, Surface Inpainting, Fabric Reconstruction

ABSTRACT

When scanning real shapes, occlusion issues may lead to holes in the reconstructed surface which must be solved using an inpainting technique. When dealing with fabrics with folds, reconstruction gets even more challenging because these occlusion problems become almost inevitable and strong assumptions are implied on the physical model of the inpainted surface. We propose a framework to fill holes in triangle mesh surfaces representing fabrics. The method leverages the developable nature of fabrics to recover the intrinsic geometry of the missing patch in 2D. Our inpainting strategy is then based on a variational method to smoothly incorporate the patch into the surface by minimizing an isometric energy. The proposed approach allows us to produce folds and creases which are difficult to obtain with general purpose hole filling techniques. Moreover, our approach remains relevant in the case where the model is not provided by the digitization of a real fabric as for the acquisition from ancient statues with draperies.

1. Introduction

When capturing surfaces, the presence of holes in the digital model can often be attributed to flaws in 3D scanning techniques with poor quality equipment that are highly sensitive to the reflectance of scanned objects. Occlusion problems, instead, arise from the complexity of objects and the difficulty of exploring all their folds with sensors. Sometimes, the object of interest is already damaged or incomplete when scanned. In this case, an expert can create contours around the parts he wants to replace in order to complete the surface differently. This is particularly true in the field of archaeology and cultural heritage preservation where restoration is a crucial issue. In this work, we propose to focus on infilling holes on surfaces representing fabrics or clothes. Applications to this type of approach are useful for scanning and completing dressed characters. It is a challenging task due to the presence of occlusions when capturing a fabric, and it implies a strong prior on the physical model of the inpainted surface, modeled hereafter as *as-developable-as-possible*. In case the region to be completed does not fully correspond to real fabric, as in the case of a hole in the toga of a statue carved in stone, our approach remains valid as long as the developability defect is not too severe.

Loosely speaking, developable surfaces are the ones that can be flattened to the plane without any metric distortion. Developable surfaces meet a number of properties such as a null Gaussian curvature at each point except on crease edges, meaning that the curvature is null in at least one direction which is commonly called the ruling direction. They belong to the family of the ruled surfaces as they are surfaces containing a piece of straight line at each point.

All those properties are rather restrictive about the space of shapes they cover and fabrics do not fully check all of them. Their material usually allows some distortion and stretching which makes them not fully developable. However, we can consider that fibers of a non-elastic fabric correspond to constant-length curves embedded in the surface, hence the use of a con-

⋆Only capitalize first word and proper nouns in the title.

e-mail: guillaume.gisbert@liris.cnrs.fr (Guillaume Gisbert), raphaelle.chaine@liris.cnrs.fr (Raphaelle Chaine), david.coeurjolly@liris.cnrs.fr (David Coeurjolly)

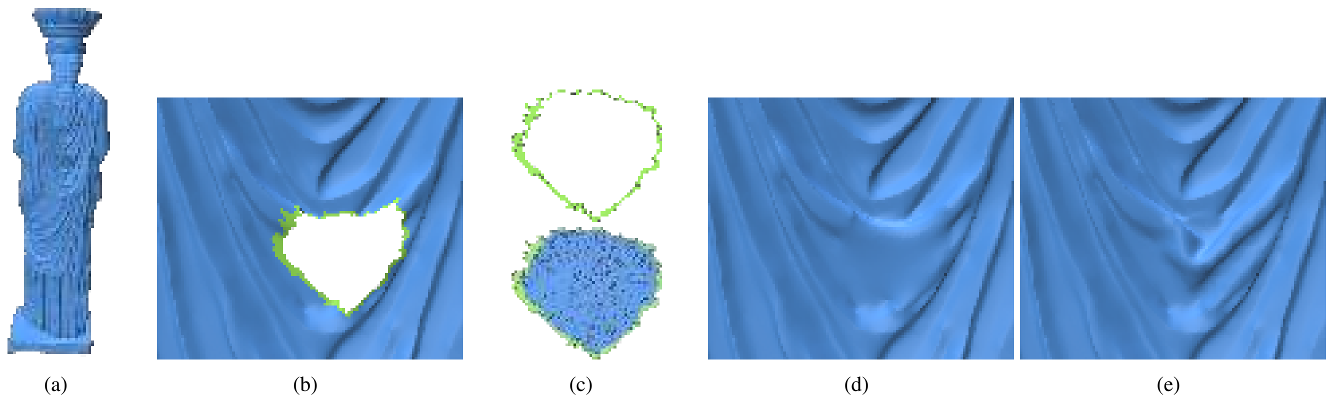


Fig. 1. Illustration of the overall pipeline: Given an input mesh of a developable fabric with a hole (a), the first step consists in estimating the missing 2D patch shape. For this, a parametrization algorithm is applied to flatten the triangle strip surrounding the hole, which is then triangulated in the 2D plane (b); The 3d geometry is subsequently restored in two steps, first by applying As-Rigid-As-Possible deformation to preserve some rigidity in every direction around the points (c), and second by optimizing a developability function in order to further curve the fabric model and its different folds. (d).

trolled developability property in our approach. In the following, we will refer to developable fabrics.

Our main contributions are:

1. We propose a method for filling holes in meshes approximating a piece of fabric when these holes are homeomorphic to disks.
2. Our approach favors the local developability of the inpainting surface but does not enforce it, allowing for some degree of distortion and stretching that are typical of clothes.

We demonstrate the effectiveness of our method by applying it to holes on a variety of surfaces, including both real-world and synthetic examples. Our results show that our approach can produce high-quality reconstructions of holes in both cloth-like and developable surfaces.

2. Related work

Various techniques and methods have been developed to address the problem of hole filling or inpainting in meshes. Those can be divided into multiple categories.

Generic mesh hole-filling. Liepa [1] proposed a popular and generic hole-filling method consisting in finding the minimal area triangulation of the 3D contour. The triangulation is refined to match the triangle size of the input mesh and smoothed using a fairing technique. Feng et al. [2] proposed using different methods depending on the hole size. For small areas, a simple centroid triangulation is enough, for medium areas they use a pipeline very similar to Liepa's [1] involving minimum area triangulation, refinement and fairing, and for big holes they use an advancing front method which consists in adding mesh element iteratively from the boundary.

Learning-based approaches. Deep learning models rapidly improved at generating data and can be used to solve the hole-filling task. Some methods discretize a point cloud or a mesh on a regular voxel grid before using 3D convolutional neural networks to process the data [3]. While taking advantage of powerful efficient architecture such as 3D-CNN, these strategies are inherently subject to the loss of geometric information because of the voxel grid resolution. Thus, many methods prefer to work directly with point clouds to exploit the full information available. Ren et al. [4] propose to use a multi-scale upsampling GAN to generate the missing points in a corrupted point cloud. Downsampling the input point cloud at different scales helps the reconstruction to be more robust and able to restore fine-grained details. These methods are however still struggling when the missing area is too large. While most learning-based approaches have the obvious disadvantage of requiring a lot of data, self-prior methods use only the input mesh.

Context-based methods. Some methods use the deep learning environment in a clever way allowing them to work solely on the input data. Point2Mesh [5] is a technique to reconstruct a watertight mesh from an input point cloud by optimizing the weights of a CNN to iteratively deform an initial mesh to shrink-wrap the given point cloud. The convolutional kernels are optimized globally to encourage self-similarities across the entire surface. Similarly, Hattori et al. [6] introduced Deep Image Prior, a method that tries to match a noisy mesh to the observed corrupted mesh using a graph convolutional network (CGN). The GCN's weight-sharing nature allows the unknown vertices to be mapped to a convincing surface even though no error metric is measured on them. While these self-prior methods use a deep learning framework, more traditional context-based methods also exist, such as the work from Harary et al. [7]. For any given hole, several patches are proposed based on a multiscale signature. They introduced a dissimilarity between two patches to establish a coherence error that has to be minimized. Context-based methods are generally rather dependent on the mesh and require to have many features such as the one in the missing area for proper reconstruction.

Implicit surfaces. Modeling the surface as the zero set of an implicit function is a classical tool for many geometry processing tasks such as surface reconstruction or fairing [8, 9]. Specifically for hole filling, Davis et al. [10] applied a volumetric diffusion process to extend the ambient signed distance function defined by the input surface, into the hole. While this approach is well adapted to geometrically and topologically complex holes, the reconstructed surface is however not guaranteed to blend well with the observed surface. However, the authors propose to adapt the diffusion process to a given problem enforcing some properties (such as curvature). Branch et al. [11] and Wue et al. [12] use a radial basis function interpolator combined with an iso-surface algorithm to fill the hole. Because the interpolating function cannot correctly fit surfaces with big holes, this technique is restricted to small areas. We can also mention the work from Tekumalla and Cohen [13] who use moving least square (MLS) instead. Overall, those methods tend to smooth the mesh in the damaged area and cannot render complex material behavior such as folds.

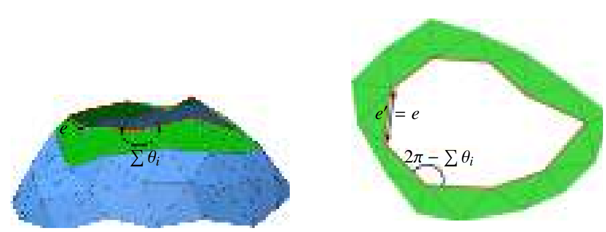
Variational approaches. Baumgärtner et al. [14] present an approach to solve mesh denoising and inpainting problems by regularizing the normal vector field using a discrete variant of the total variation. The missing area is first inpainted by solving a minimal surface problem before using their algorithm. This method performs well at preserving sharp features in the mesh. Pernot et al. [15] propose to fill holes while minimizing the curvature variation between the observed and inserted mesh. Features are not necessarily preserved but manual constraints can be added to help with the minimization. Bonneel et al. [16] also begin by recovering the missing mesh [1] before inferring differential properties on it. They solve a Mumford-Shah problem to obtain piecewise-smooth normals in the inpainted region and conform the mesh to them. These methods generally require a geometry to infer information. They are usually able to capture sharp features and propagate them in the inpainted mesh.

Developable surfaces. Developable surfaces have been extensively studied in geometry and computer graphics. [17] provides an excellent reminder of the geometric properties of developable surfaces. They are widely used in architectural design, art and product manufacturing because of their aesthetic appeal and their ease of fabrication. In fact, such surfaces can be made by cutting sheets of any flat material and bending them to recreate the desired object. We can also mention that the Gauss map of a developable surface is unidimensional [18]. Typical developable surfaces are cones and cylinders but some more complex shapes like the oloid also fit in this category. For fabrication purposes, many methods [19, 20, 18] were designed to approximate a given surface by piecewise developable surfaces with a minimum number of pieces. Because of their flattenability, developable surfaces are the perfect choice for modeling some materials such as paper and metal sheet.

Hole-filling with developable constraints. Some hole-filling techniques are specifically designed to work with developable surfaces such as the work from Frey [21] who proposes to approximate the missing mesh by triangles whose vertices are the boundary points exclusively. They project the boundary on some 2d plane and simulate the bending process to obtain the sheet. Their method is specifically designed for the fabrication of thin, flat sheet materials such as metal and is not suitable for handling fabric materials. Rose et al. [22] allows the user to draw a 3D polyline and propose a triangulation approximating a smooth developable surface within the polyline. The method is based on the fact that most edges in smooth developable surfaces are locally convex.

Fabric and cloth modeling. Simple triangulations of flattanable surfaces cannot render complex material behavior even though they can maintain some properties such as developability. Clothes are often modeled as developable surfaces in the sense that they are made from 2D patterns. However, the fabric usually allows some controlled stretching, making them not exactly developable. To our knowledge, there is no specific inpainting approach for fabrics. However, many cloth and fabric models have been developed for simulation purposes.

Some models are yarn-level, meaning that every single yarn strand is simulated to produce an animation [23, 24]. These techniques are computationally intensive and require several hours of calculations even for animations of a few seconds, yet, produce high-fidelity cloth representation and allow some behavior observable only with those models. These models excel at rendering knit patterns. More traditional models mesh-based are still able to achieve high fidelity in real-time. Most modern cloth simulation techniques rely on two terms [25, 26, 27]: a planar elasticity term that enforces isometric constraints, common choices being mass-spring systems and triangular finite element models, and bending elasticity constraints. The quadratic bending model from Bergou et al. [28] is a popular option. Our method relies on these fabric models and associated energies. It also happens that fabrics are modeled by implicit surfaces, without worrying about developability to solve untangling problems.



(a) (b)

Fig. 2. a) Initial surface mesh with a hole; b) The triangle strip around the hole boundary is first flattened using a parameterization algorithm preserving edge lengths and angles.

3. Our method

3.1. Framework outline

Given a discrete model of a fabric surface with a hole to be filled in, we need an inpainting model that supports the generation of folds and wrinkles, as opposed to conventional inpainting techniques that generally fill holes as smoothly as possible. For this purpose, our approach proceeds in two steps. First, we propose to compute an intrinsic representation of the missing patch (length, area) as a planar patch constrained by the geometrical information from the hole contour. This piece of fabric is subsequently deformed to fill the hole by taking inspiration from approaches used in simulation.

In this paper, we deal with surfaces represented by meshes. Without loss of generality, we suppose this surface has a single

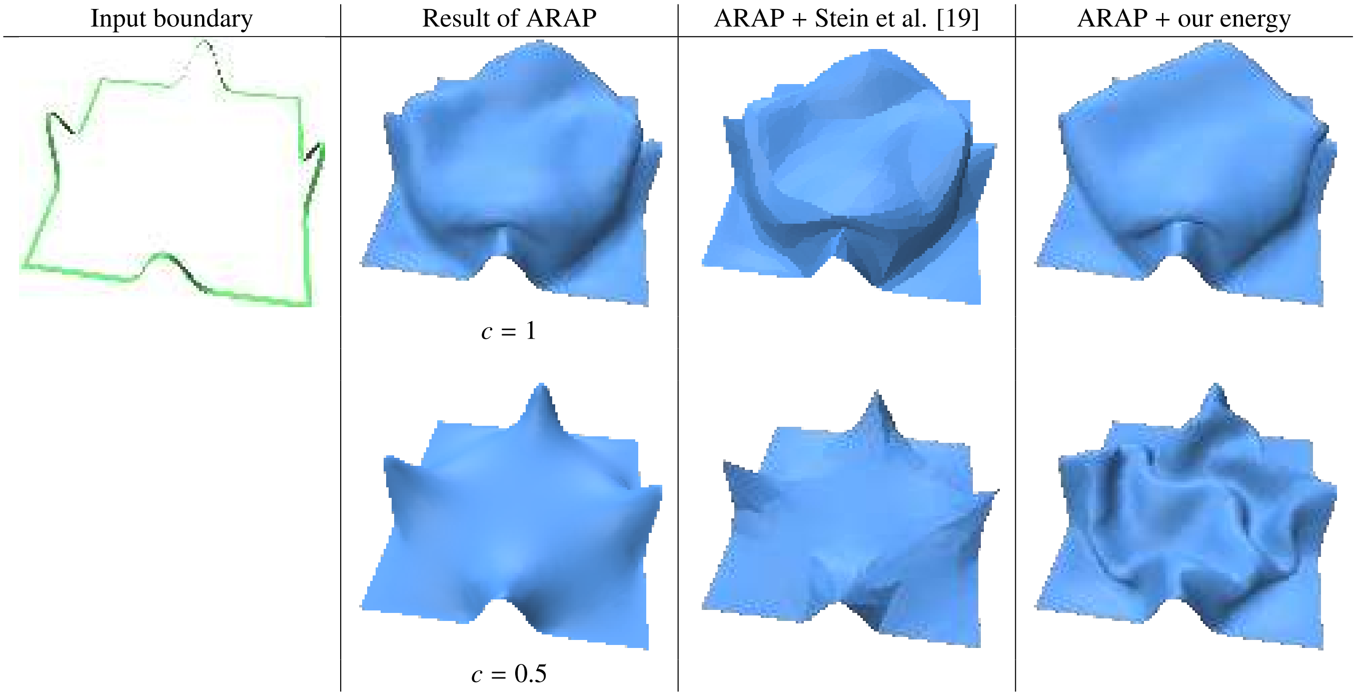


Fig. 3. ARAP tends to reduce the overall bending in the mesh and produces unrealistic meshes (the quadratic error in edge lengths is 1.91×10^{-3}, while the error in bending is 332.49). We introduce a scale factor $c \in [0, 1]$ which uniformly reduces the edge lengths targeted by ARAP. Lowering the length values only for ARAP promotes fold emergence during the following isometric optimization, resulting in a more realistic mesh (error in edge lengths: $6.85{\times}10^{-4}$, error in bending: 357.33). The developability energy from Stein et al. [19] does not produce satisfying results and is unable to make folds.

hole homeomorphic to a disk and that the hole was made in a developable fabric with no singular point.

To delimit the patch of fabric to be used for hole filling, we start by flattening the contour of the hole while preserving a prescribed angle at each contour point. This angle locally indicates the missing amount of fabric at each boundary point. That way, each point of the hole boundary belongs to an almost developable surface, in the sense of fabrics, with an angular defect that is nearly zero at each point when the fabric is flattened and practically zero when it is folded in space. This results into a target angle for the missing patch at each boundary point (See figure 2). The flattening problem is therefore equivalent to solving a problem of 2D polygon generation, with constraints on the edge lengths and the angles at vertices. However, to handle the problem even when the input surface is not fully developable, we resort to a more robust solution using a parameterization technique preserving both lengths and angles as described in Section 3.2. Once we have the contour of the planar patch of fabric that will be used to fill the hole, we mesh it directly in 2D, by using a 2D version of the triangulation algorithm used in [1].

This approach, which consists in estimating the intrinsic geometry of the patch i.e. the appropriate amount of missing material before positioning it into 3D is different from other general surface inpainting algorithms. Throughout the remaining steps, the distances and angles will be measured on this planar mesh, and our method aims at preserving these properties as much as possible when positioning the patch in 3D by finding a bijective mapping between the 2D patch and the 3D mesh which is consistent in terms of lengths and angles.

The second step of our approach is to move the boundary of the patch back to its 3D position and to move the interior of the patch in an *as-developable-as-possible* way. The deformations of developable surfaces are isometric and conformal in the sense that they preserve lengths and angles. When bending a developable surface, there is a preferential direction at each point in which the curvature remains zero. This direction of the tangential plane is the ruling direction. The surface can bend in the orthogonal direction only. The ruling direction is specific to each surface point and depends on the deformation. The ruling directions of the points are unknown in advance and we propose a strategy to let them emerge gradually through a sequence of two steps of variational optimizations.

First, we deform the surface by using as-rigid-as-possible deformations that are famous for their editing possibilities (ARAP [29]). From our flat patch of surface, they can be used to move the boundary vertices toward their target position in 3D while trying to preserve lengths and zero curvature in all directions around all the vertices. They thus satisfy the requirements for the deformation of developable surfaces but locally overconstrain them by seeking rigidity in all directions.

In practice, it is not always possible for the deformed surface to remain flat in all directions. This results in the emergence of a field of ruling directions over the surface, here corresponding to the minimal curvature direction at each point (see figure 8). From this initial state, we then propose to further deform the surface in a second step, in order to re-enforce the ruling directions by minimizing a developability energy inspired by cloth simulation. This helps to cancel the residual curvature in the ruling direction and to curve the surface in the maximal curvature direction. During this second step, new folds may also emerge if necessary (see figure 8). In practice, we do not need to extract the ruling directions explicitly. Our approach is detailed in Section 3.3. Figure 1 illustrates the entire pipeline.

3.2. *Estimation of the 2D patch*

As discussed in the previous subsection, the objective is to compute a 2D shape from a given set of edge lengths and angles at the boundary 2. Note that in the case of fully developable surfaces, the set of edge lengths and angles corresponds to a valid closed polygon. In real case scenario when the input mesh is not fully developable. We compute a polygon that best satisfies lengths and angles constraints in the sense of isometric mesh parameterization algorithms. For that purpose, we directly apply such a parameterization algorithm on the triangle strip surrounding the hole.

Parameterization algorithms aim at finding a mapping from a surface S in $\mathbb{R}^3$ to $\mathbb{R}^2$ by giving each vertex a 2D coordinate. This kind of algorithm is suitable for our problem and is used to flatten the geometry. The difficulty is to find a bijective parameterization of the hole with no intersection of the unfolded boundary in the plane, while best preserving lengths and angles.

Various algorithms have been developed for mesh parameterization, preserving various quantities. In order to reduce angular distortion, we chose the Least-Squares Conformal Maps [30] method (LSCM). LSCM minimizes the following energy:

$$E_{LSCM}(\mathbf{u}, \mathbf{v}) = \int_X \frac{1}{2} |\nabla \mathbf{u}^{\perp} - \nabla \mathbf{v}|^2 dA \,,$$

where u and v are vectors of 2D coordinates, $\perp$ denotes a counterclockwise 90° rotation and X is the domain to parameterize. The method preserves the angles, but the lengths are only proportional to a scaling factor. By fixing two points, lengths can be accurately maintained. On the strips of triangles that we encountered in practice, the algorithm behaves well and produces a bijection, even in cases where the input deviates from the assumption of developability.

The resulting 2d polygon is filled using an algorithm inspired by the first steps of [1] and provided by CGAL [31]. The algorithm first performs an optimal triangulation before refining it by adding vertices and flipping edges. The triangle density in the patch is determined based on the boundary density.

3.3. *3D embedding of the patch*

Once the shape of the planar missing patch is estimated, its triangulation needs to be brought back in 3D while creating folds that locally preserve the developability of the surface. For that, we wish to optimize an energy that measures the developability of our surface. We would like to emphasize that the previous determination of the planar patch is essential for this task because it already provides an estimation of the triangles with their target size, shape, and connectivity, but not their 3D embedding. Indeed, a conventional hole-filling technique followed by a cloth simulation-inspired energy would not suffice as the first step would fill the surface without consideration for the developability of the surface.

Our energy model is inspired by cloth simulation models requiring an isometric term and a bending term. To ensure isometry, we simply enforce edge lengths to be preserved using an l_2 penalty:

$$E_I = \frac{1}{2} \sum_{e_i \in \mathcal{E}} (\|\hat{e}_i\| - \|e_i'\|)^2 \,,$$

where $\hat{e}_i$ denotes the estimated edge length and e_i' denotes the target length. The gradient of this energy with respect to the position of vertex v_1 is the following:

$$\nabla_{v_1} E_I = -(\|\hat{e_{12}}\| - \|e_{12}'\|) . \frac{\hat{e_{12}}}{\|\hat{e_{12}}\|} \,,$$

If we use this isometric energy alone, the surface produced is not regular enough in terms of curvature (see Section 4.3), hence it usually comes with a term penalizing dihedral angles which smooth the surface out. We use the quadratic bending model from [28], which is formulated as follows:

$$E_B = \frac{1}{2} x^T (L^T M^{-1} L) x \,,$$

$$\nabla E_B = x(L^T M^{-1} L) \,,$$

where x denotes the vertex position vector, L corresponds to the discrete Laplace-Beltrami matrix and M is the lumped mass matrix. Our final energy model is simply expressed as:

$$E = E_I + \lambda E_B \,,$$

where λ is a user-defined parameter used to tailor the optimization process to the specific requirements of the application. This parameter serves as a pseudo-physical parameter that allows the user to adjust the material rigidity.

In our setting, the variables of the energy minimization are the vertex positions in the inpainted area. Of course, there is not only one solution to the positioning of our patch in 3D. It is quite possible to consider the flat patch, to bring up its boundary in 3D and to locally optimize the energy directly from this state (see Section 4.3). However, ARAP can be used to initialize our energy with a state that helps the extension of the boundary folds within the patch.

We minimize the energy using a gradient descent with a fixed step size. More advanced optimization techniques such as projective quasi-Newton solvers could have been used. In terms of performance, our current approach can still fill hole meshes with 3755 vertices in less than 3.73 seconds (see 4). Furthermore, the simplicity of our method allows us to perform a simple collision test during iterations as discussed in the next section.

As stated in the overall presentation of our approach, the optimization proceeds in two steps. The gradient descent of the developability energy is done only after applying the ARAP-optimization algorithm proposed in [29], initialized from the planar patch and constrained by the hole vertex positions as illustrated in Figure 3. ARAP penalizes non-isometric deformations and aims to keep the mesh as flat as possible. While this behavior can be desired for some applications, it is not fully prone to the emergence of folds and tends to smooth the mesh out.

However, the ARAP minimization may be a local minimum for our proposed energy function, which may limit the emergence of further folds in the mesh. To inject more freedom during our optimization, we slightly modify the ARAP step. The idea is to provide ARAP with an initial patch of reduced size to limit the number of folds it will generate, allowing it to focus on essential folds induced by the boundary. If ARAP has no room to create folds in a given direction it will temporarily stretch the edge lengths in that direction. Not all edges are equally stretched depending on their orientation. During the second optimization step following ARAP, stretched edges can recover their initial lengths and further folds can emerge between the first ones. For this temporary reduction purpose, a parameter $c \in [0, 1]$ is introduced to scale down the target edge lengths only for ARAP. This produces a satisfactory initial state for the algorithm, while promoting further emergence of folds when the edge lengths recover their initial target values during the second optimization step. Figure 3 illustrates the ARAP results for two different values of c demonstrating how the classic approach can become trapped in local minima.

3.4. Self-intersection and collision

When performing the gradient descent on the vertex positions, self-intersections can occur. To overcome this, we added a repulsion energy as a soft constraint between close vertices v_i and v_j:

$$E_C(i, j) = \frac{\kappa}{2} \max(r - \|v_i - v_j\|, 0)^2 ,$$

where κ is the repulsive strength and r is the collision radius threshold that should be adapted to the resolution of the mesh. This classical approach is computationally efficient compared to other techniques, though it requires small steps and cannot resolve existing self-intersections. Because our energy is also minimized via a gradient descent, the displacement steps at each iteration are small enough to be used in combination with this additional optimization. This proved to be sufficient in our experiments. Also, to reduce time complexity, we use a spatial hashing data structure, ideal for fixed-radius near-neighbor queries in low dimensions.

Finally, we also provide the method with a simple object/cloth collision tool that projects colliding vertices to the nearest valid position. Specifically, at each time step, we check for any potential intersection between the patch vertices and the objects of the scene by using the signed distance function (SDF) and its gradient. We implemented the method with simple shapes such as spheres and cylinders but the method is immediately adaptable to any shape given with its SDF and SDF gradient.

4. Results and discussions

In this section, we present the results of our experiments and some comparisons with existing work. We implemented the method in C++ using OpenMP for parallel computation. We used Geometry Central [32] as our primary geometry processing and meshes handling library and polyscope [33] for visualization. Additionally, LSCM was implemented using LibIGL [34], and the triangulation was performed using CGAL [31]. We share our code at: `https://github.com/g-gisbert/Inpainting-Holes-In-Folded-Fabric-Meshes`.

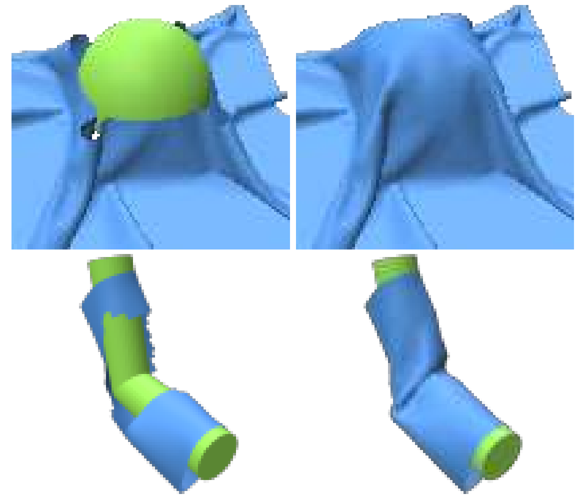


Fig. 4. When inpainting a hole, we can provide external constraints to conform the surface to a geometrical proxy (green sphere or tubular structure).

4.1. Selection of illustrative models

To illustrate our approach, we have applied it to different kinds of input mesh data. The meshes used for evaluation come from real-world scanned fabrics, scanned statues, or produced through a cloth simulation. First, we scanned two pieces of fabric (*Scanned Fabric 1* and *Scanned Fabric 2*). After meshing it, we created a hole in the area with lots of folds. The example *Curtain* was produced using the cloth simulation tool in Blender. We also took the example of fully developable surfaces (*Paper1* and *Paper2*). Finally, we took the meshes of two draped statues (*Lucy* and *Caryatid*), for which we deviate from the situation of almost developability.

4.2. Results analysis

First of all, results given in Figure 5 demonstrate the effectiveness of our method on various examples. Specifically, the first row showcases our ability to address a common challenge in hole-filling techniques - the presence of large holes. In this case, we were able to successfully extend the folds from one end to the other in *Curtain*. The results produced from *Scanned Fabric 1* and *Scanned Fabric 2* create folds in the missing area that are as convincing as those produced in the ground truth.

It took 3.70s in 3237 iterations to fill the hole with a patch of 3755 vertices for *Curtain*, 6.08s in 8181 iterations and 2399 vertices for *Scanned Fabric 1*, and 1.12s in 5780 iterations and 703 vertices for *Scanned Fabric 2*.

In Figure 6 we have considered two examples [35] of purely developable surfaces. Even if the hole geometry is complex, our method achieves high-quality results.

The Figure 7 shows experiments on scanned statues with comparisons to alternative approaches. Although the drapery looks realistic, they are not truly developable surfaces (the first

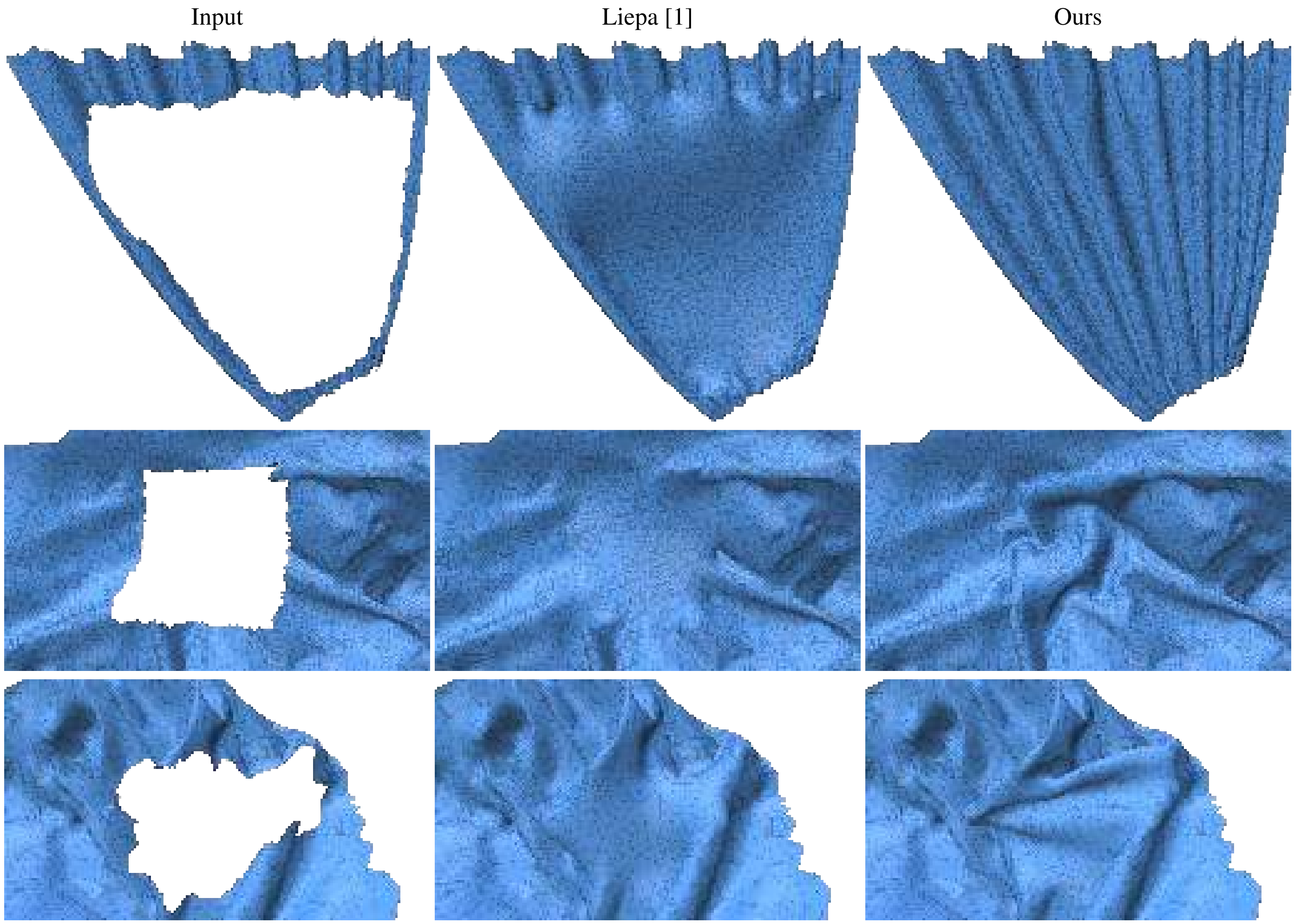


Fig. 5. Results of our method on some examples (resp. *Curtain*, *Scanned Fabric 1*, *Scanned Fabric 2*). Our method provides the emergence of convincing folds in the inpainted area while remaining consistent with the surrounding geometry.

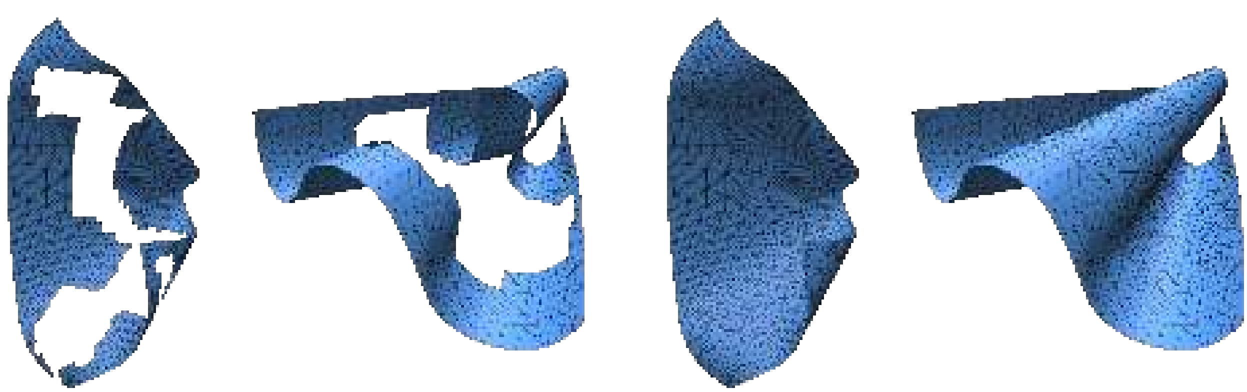


Fig. 6. Examples of purely developable surfaces reconstructed with our method (Paper1 and Paper2). The first two columns show the input meshes and the other two
show the inpainted meshes.

row shows a detail of *Lucy* while the second shows Caryatid). In practice, the LSCM parametrization is able to produce fine parametrization but the patch area is sometimes over-estimated or under-estimated leading to more or fewer folds than those on the original statue. [1] fills each hole in about 10ms and adds 2399 vertices in each case. Because [16] is initialized from [1], it also has 2399 vertices in the inpainted area and takes about 3min to compute. [5] computes both in about 5h and works with 14000 vertices. Finally, our method fills each hole in about 5s with 2399 vertices. Our method is relatively efficient with a calculation time that is within an order of magnitude of seconds, even though techniques like [1] can reach a triangulation 100 times faster. In contrast, our method is the only one to be able to render realistic folds. An interesting point is that the machine learning approach based on self-prior [5] failed to learn the content of the patch from the remaining parts of the statues, probably because the hole was too wide. The self-prior approach seems to still struggle with capturing fine details.

4.3. Developability evaluation and ablation study

The problem of filling holes in fabric surfaces does not lead to a unique developable solution. Hence, a geometric surface to surface correspondence metric, such as the Hausdorff distance, does not make much sense. Therefore, we illustrate the effects of our approach with some quantitative data regarding the output surface quality. In the first step of *as-rigid-as-possible* optimization, flatness is favored in all directions around the vertices. This results in a solution where both the mean curvature and the Gaussian curvature are minimized jointly, with some constraints at the boundary. In the second step, only the local developability property is reinforced, and a significant reduction of the residual Gaussian curvature can be obtained. This is what we observe on the graph of Figure 8, with a decrease in the Gaussian curvature mean and its variance.

In Figure 9, we also present reconstruction results using an alternative initialization to ARAP resulting in outputs with limited folds. Additionally, starting on from the ARAP surface, we demonstrate in this figure the relevance of the bending term in our energy model, when only keeping the isometric term.

4.4. Limitations

Although our method achieves high-fidelity reconstruction, it comes with a few limitations. First of all, our optimization steps are comparable to those used in cloth simulation. Thus, the method is computationally intensive and would particularly benefit from a GPU implementation that seems quite possible to obtain. Secondly, a fundamental assumption in the method is the developability of the triangle strip surrounding the hole. That is, we use the exact vertex positions and angles to estimate the 2D patch making the process sensitive to noise (see Figure 10). In some cases, it could be helpful to compute a final displacement of the patch that better takes into account the developability defect, especially in terms of the area of fabrics.

5. Conclusion

In this paper, we proposed a novel framework for filling holes in fabric meshes. The method involves estimating the shape of the hole in a plane and then minimizing a cloth simulation-based energy in two steps to obtain the final 3D shape. The framework naturally captures the developable nature of fabrics and yields high-fidelity reconstructions even for large holes. Our method demonstrates high fidelity in reconstructing both fabrics and developable surfaces, even when the holes are substantial in size. Moving forward, there is potential for expanding our research by incorporating more complex cloth models with seams and adapting the framework to work with piecewise developable surfaces, such as those found in clothes. For this, machine learning could be used to position the seams at the most plausible locations. Finally, the method could be extended to work with holes containing islands or of different nature. Regarding the input data, we also want to address the challenge of working directly on point clouds when the borders of the holes are not clearly identified.

6. Acknowledgements

This work is part of the e-Roma project from the French Agence Nationale de la Recherche (ANR-16-CE38-0009). It was funded, in whole or in part, by l'Agence Nationale de la Recherche (ANR), project StableProxies ANR-22-CE46-0006. For the purpose of open access, the author has applied a CC-BY public copyright licence to any Author Accepted Manuscript (AAM) version.

References

[1] Liepa, P. Filling Holes in Meshes. In: Kobbelt, L, Schroeder, P, Hoppe, H, editors. Eurographics Symposium on Geometry Processing. The Eurographics Association. ISBN 3-905673-06-1; 2003,doi:10.2312/SGP/SGP03/200-206.

[2] Feng, C, Liang, J, Ren, M, Qiao, G, Lu, W, Liu, S. A fast hole-filling method for triangular mesh in additive repair. Applied Sciences 2020;10(3). doi:10.3390/app10030969.

[3] Han, X, Li, Z, Haibin, H, Kalogerakis, E, Yu, Y. High-resolution shape completion using deep neural networks for global structure and local geometry inference. 2017,doi:10.1109/ICCV.2017.19.

[4] Ren, Y, Chu, T, Jiao, Y, Zhou, M, Geng, G, Li, K, et al. Multi-scale upsampling gan based hole-filling framework for high-quality 3d cultural heritage artifacts. Applied Sciences 2022;12(9). doi:10.3390/app12094581.

[5] Hanocka, R, Metzer, G, Giryes, R, Cohen-Or, D. Point2mesh: A self-prior for deformable meshes. ACM Trans Graph 2020;39(4). doi:10.1145/3386569.3392415.

[6] Hattori, S, Yatagawa, T, Ohtake, Y, Suzuki, H. Deep mesh prior: Unsupervised mesh restoration using graph convolutional networks. 2021. doi:10.48550/ARXIV.2107.02909.

[7] Harary, G, Tal, A, Grinspun, E. Context-based coherent surface completion. ACM Trans Graph 2014;33(1). doi:10.1145/2532548.

[8] Kazhdan, M, Bolitho, M, Hoppe, H. Poisson surface reconstruction. In: Proceedings of the fourth Eurographics symposium on Geometry processing; vol. 7. 2006, p. 0.

[9] Barill, G, Dickson, NG, Schmidt, R, Levin, DI, Jacobson, A. Fast winding numbers for soups and clouds. ACM Transactions on Graphics (TOG) 2018;37(4):1–12.

[10] Davis, J, Marschner, S, Garr, M, Levoy, M. Filling holes in complex surfaces using volumetric diffusion. In: Proceedings. First International Symposium on 3D Data Processing Visualization and Transmission. 2002, p. 428–441. doi:10.1109/TDPVT.2002.1024098.

[11] Branch, J, Prieto, F, Boulanger, P. A hole-filling algorithm for triangular meshes using local radial basis function. In: Pébay, PP, editor. Proceedings of the 15th International Meshing Roundtable. Berlin, Heidelberg: Springer Berlin Heidelberg. ISBN 978-3-540-34958-7; 2006, p. 411–431.

[12] Wu, XJ, Wang, MY, Han, B. An automatic hole-filling algorithm for polygon meshes. Computer-Aided Design and Applications 2008;5(6):889–899. doi:10.3722/cadaps.2008.889-899.

[13] Tekumalla, LS, Cohen, E. Reverse engineering point clouds to fit tensor product b-spline surfaces by blending local fits. ArXiv 2014;abs/1411.5993.

[14] Baumgärtner, L, Bergmann, R, Herrmann, M, Herzog, R, Schmidt, S, Vidal, J. Mesh denoising and inpainting using the total variation of the normal. 2020.

[15] Pernot, JP, Moraru, G, Véron, P. Filling holes in meshes using a mechanical model to simulate the curvature variation minimization. Computers & Graphics 2006;30(6):892–902.

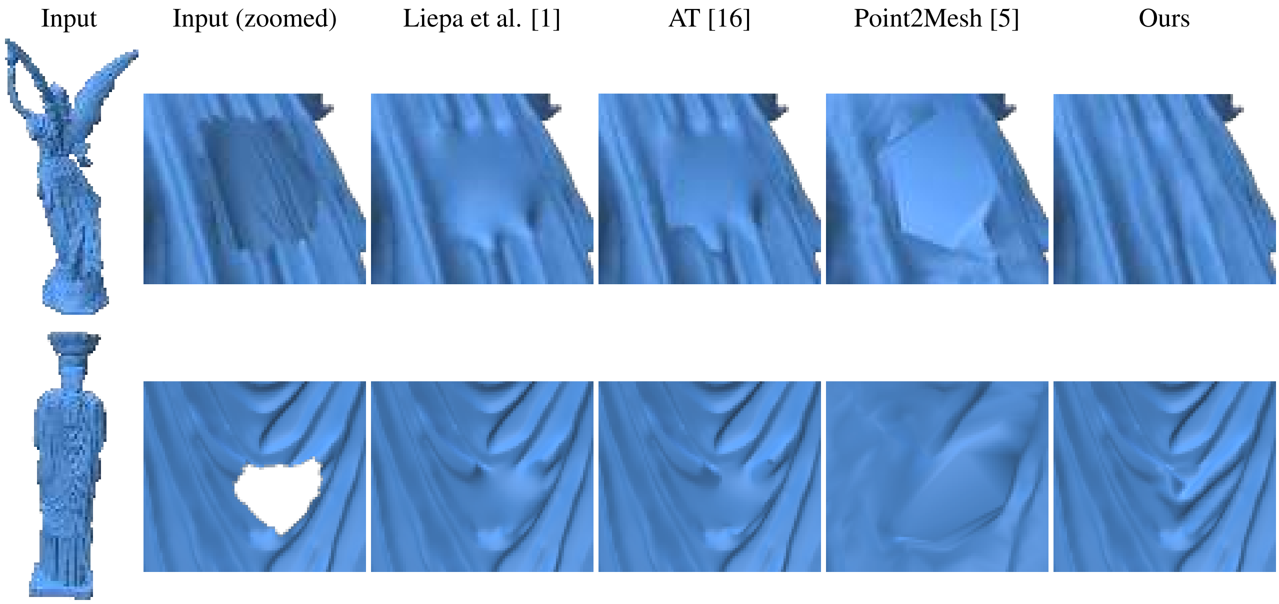


Fig. 7. Comparison between different hole filling methods in *Lucy* (first row) and *Caryatid* (second row). The results are given for different techniques: Liepa [1] (second column), Ambrosio Tortorelli [16] (third column), Self-prior [5] (fourth column), and our method (fifth column).

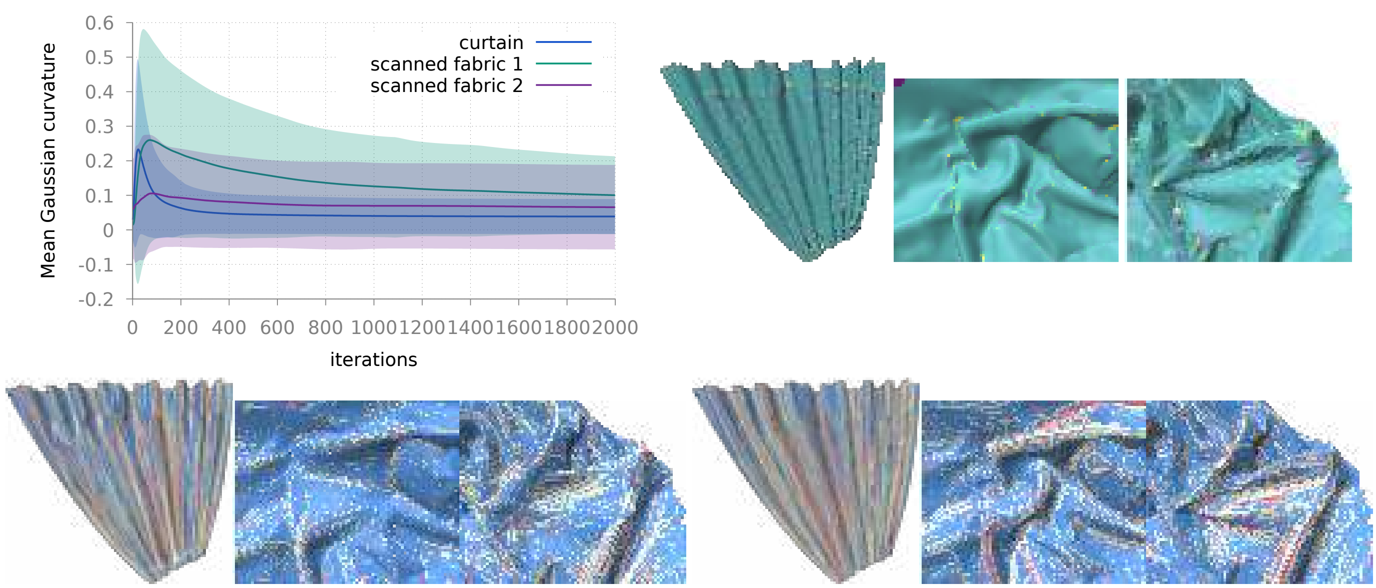


Fig. 8. Graph showing the evolution of the average Gaussian curvature and its variance on *Curtain*, *Scanned Fabric 1* and *Scanned Fabric 2* as a function of the number of iterations in the minimization process of the developability energy. The initial values are computed after applying ARAP. The top right figures show the distribution of the Gaussian curvature on the meshes. The bottom figures show the directions of minimum curvature (ruling directions) for respectively the initialization after ARAP and the minimization of our energy.

[16] Bonneel, N, Coeurjolly, D, Gueth, P, Lachaud, JO. Mumford-shah mesh processing using the ambrosio-tortorelli functional. Computer Graphics Forum (Proceedings of Pacific Graphics) 2018;37(7). doi:10.1111/cgf.13549.

[17] Lawrence, S. Developable surfaces: Their history and application. Nexus Network Journal 2011;13. doi:10.1007/s00004-011-0087-z.

[18] Binninger, A, Verhoeven, F, Herholz, P, Sorkine-Hornung, O. Developable approximation via gauss image thinning. Computer Graphics Forum 2021;40(5):289–300. doi:https://doi.org/10.1111/cgf.14374.

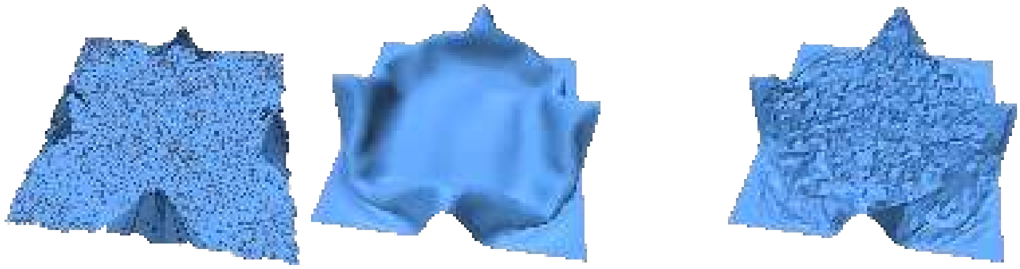
[19] Stein, O, Grinspun, E, Crane, K. Developability of triangle meshes. ACM Trans Graph 2018;37(4).

[20] Liu, HTD, Jacobson, A. Normal-driven spherical shape analogies. Computer Graphics Forum 2021;40(5):45–55. doi:https://doi.org/10.1111/cgf.14356.

[21] FREY, WH. Boundary triangulations approximating developable surfaces that interpolate a closed space curve. International Journal of Foundations of Computer Science 2002;13(02):285–302. doi:10.1142/S0129054102001096.

[22] Rose, K, Sheffer, A, Wither, J, Cani, MP, Thibert, B. Developable Surfaces from Arbitrary Sketched Boundaries. In: Belyaev, AG, Garland, M, editors. SGP '07 - 5th Eurographics Symposium on Geometry Processing. SGP '07 Proceedings of the fifth Eurographics symposium on Geometry processing; Barcelona, Spain: Eurographics Association; 2007, p. 163–172.

[23] Yuksel, C, Kaldor, JM, James, DL, Marschner, S. Stitch meshes for modeling knitted clothing with yarn-level detail. ACM Transactions on Graphics (Proceedings of SIGGRAPH 2012) 2012;31(3):37:1–37:12.



(a) (b) (c)

Fig. 9. Ablation study: Starting from the input contour of Figure 3, if we minimize our energy on a planar reconstruction (a) instead of a ARAP surface, the surface will often converge toward specific solutions which do not feel natural(b). When removing the bending energy term from our energy (starting from an initial ARAP step with $c = 0.5$), although the lengths are well preserved (quadratic error in edge lengths: 2.61×10^{-4}), we obtain a highly distorted and unsatisfactory result (c).

doi:10.1145/2185520.2185533.

[24] Sperl, G, Narain, R, Wojtan, C. Homogenized yarn-level cloth. ACM Transactions on Graphics (TOG) 2020;39(4).

[25] Wu, L, Wu, B, Yang, Y, Wang, H. A safe and fast repulsion method for gpu-based cloth self collisions. ACM Trans Graph 2020;40(1). doi:10.1145/3430025.

[26] Wang, H. Gpu-based simulation of cloth wrinkles at submillimeter levels. ACM Trans Graph 2021;40(4). doi:10.1145/3450626.3459787.

[27] Wu, B, Wang, Z, Wang, H. A gpu-based multilevel additive schwarz preconditioner for cloth and deformable body simulation. ACM Trans Graph 2022;41(4). doi:10.1145/3528223.3530085.

[28] Bergou, M, Wardetzky, M, Harmon, D, Zorin, D, Grinspun, E. A quadratic bending model for inextensible surfaces. In: Proceedings of the Fourth Eurographics Symposium on Geometry Processing. SGP '06; Goslar, DEU: Eurographics Association. ISBN 3905673363; 2006, p. 227–230.

[29] Sorkine, O, Alexa, M. As-rigid-as-possible surface modeling. 2007, p. 109–116. doi:10.1145/1281991.1282006.

[30] Lévy, B, Petitjean, S, Ray, N, Maillo t, J. Least squares conformal maps for automatic texture atlas generation. In: ACM, , editor. ACM SIGGRAPH conference proceedings. 2002,.

[31] The CGAL Project, . CGAL User and Reference Manual. 5.5.1 ed.; CGAL Editorial Board; 2022. URL: https://doc.cgal.org/5.5.1/Manual/packages.html.

[32] Sharp, N, Crane, K, et al. geometry-central. 2019. Www.geometry-central.net.

[33] Sharp, N, et al. Polyscope. 2019. Www.polyscope.run.

[34] Jacobson, A, Panozzo, D, et al. libigl: A simple C++ geometry processing library. 2018. Https://libigl.github.io/.

[35] Verhoeven, F, Vaxman, A, Hoffmann, T, Sorkine-Hornung, O. Dev2pq: Planar quadrilateral strip remeshing of developable surfaces. ACM Transactions on Graphics (TOG) 2022;41(3):29:1–18.

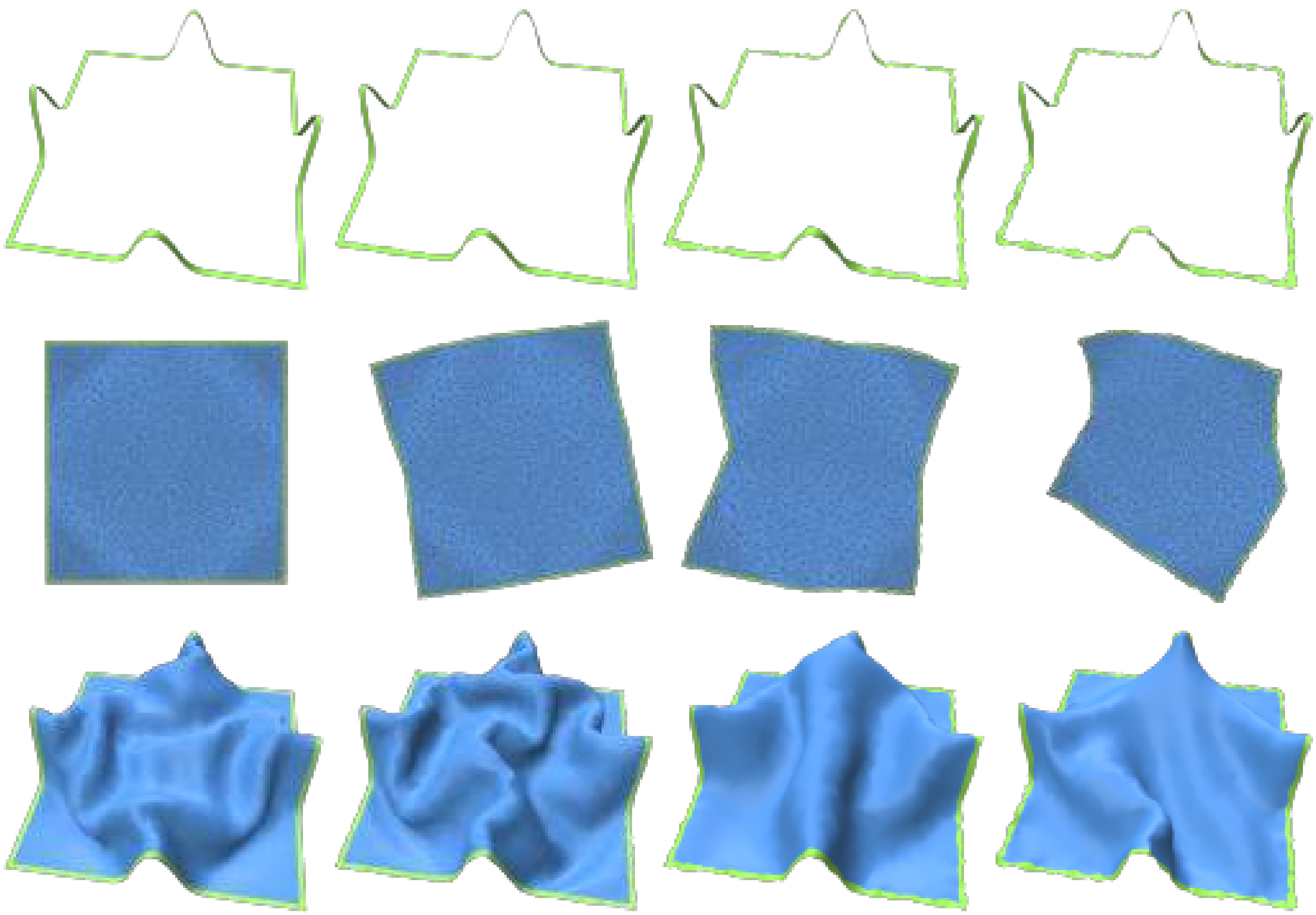


Fig. 10. Influence of noise in the flattening of the hole boundary (second row) and the resulting hole filling (third row). Each vertex of the input mesh has been randomly shifted by a 3D vector uniformly sampled in the unit square times σ (first row). The first column is free of noise, the second represents the result for σ, the third for 2σ and the last for 3σ.