Final Report

"As rigid as possible" - Group 05

1 Abstract

In this assignment we successfully implemented smooth surface deformation of general shapes. Shape deformation has many applications in geometry processing; we focused on interactive handle-based deformation. We propose an implementation of the surface deformation algorithm "as-rigid-as-possible" (ARAP) based on [7]. We conclude with a discussion of different implementation issues and a general performance of the method.

2 Introduction

Interactive mesh deformation has wide use cases in the fields of animation and computer graphics. For example, it can be used for unwrapping a mesh [3] or morphing one shape into another, which is highly used in animation and computer games [1],[5].
The most important criterion for mesh deformation is to preserve the topology and scale thus imposing an "as-rigid-as-possible" constraint [7]. To achieve this, we can set up an energy term which tries to transform cells $\mathcal{C}$ in the original mesh $\mathcal{S}$ to the new cell $\mathcal{C}'$ in mesh $\mathcal{S}'$, resulting in the following equation:

$$\begin{aligned} E(\mathcal{S}') &= \sum_{i=1}^{n} E(\mathcal{C}_i, \mathcal{C}_i') \\ &= \sum_{i=1}^{n} w_i \sum_{j \in \mathcal{N}(i)} w_{ij} \|(p_i' - p_j') - R_i(p_i - p_j)\|^2 \end{aligned} \tag{1}$$

Where the weights w_{ij} are the cotangent weights

$$w_{ij} = \frac{1}{2}(\cot \alpha_{ij} + \cot \beta_{ij}) \tag{2}$$

and α_{ij}, β_{ij} are the opposite angles of the edge e_{ij}.
To minimize the energy $E(\mathcal{S}')$, we need to find a set of rotations $\{R_i\}$ and points p'. This can be done by alternating the following two steps until convergence:

- Fix the points p' and find the set of rotations $\{R_i\}$ to minimize $E(\mathcal{S}')$
- Fix the rotations $\{R_i\}$ and find points p' that minimize $E(\mathcal{S}')$

Finding the right rotations $\{R_i\}$ is also called the local optimization step and likewise finding the points p' the global optimization step.

3 Related Work

Our project is based on Sorkine's work [6], which proposes the Laplacian for interactive surface editing while preserving the details of the geometry.
Igarashi et. al [4] use a similar approach to [6], but define an error term that is based on whole triangles not on the separate vertices. This method is refined in [3] to allow placement of handles/constraints at arbitrary positions, not only at the vertex locations. They also describe how this method could be used for mesh unfolding.
The as-rigid-as-possible paradigm can also be used for surface morphing, i.e. interpolating between two or more different meshes to get a seamless animation between different stages of animation [1],[5].

4 Method

We have implemented the as-rigid-as-possible deformation technique using C++. We used the sparse Cholesky solver provided with the `Eigen` library[1].
To get familiar with the ARAP paradigm, we first implemented the algorithm in a separate repository which is based on an exercise for a mesh processing course of the university of Toronto[2]. At this point, we relied on predefined methods provided from the library `libigl`[3]. As soon as we had a working implementation of the main algorithm, we integrated this approach in our main repository, where we replaced the usage of libigl's predefined methods with our own implementation.
In the meantime, basic user interactions with our application were implemented. Since `libigl` also provides basic UI handling and loading of meshes, it serves as a general mesh processing pipeline and the user interface.

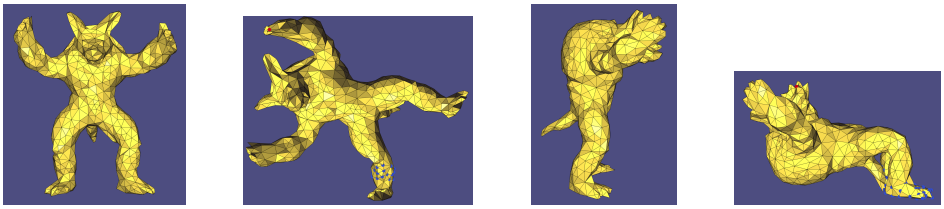
In the final application, the user can load a selected mesh, place multiple control points and deform the mesh via click and drag. The previous frame serves as an initial guess as suggested by [7]. The iterative "flip-flop" optimization technique described in section 2 runs continuously, such that the user can observe how the mesh is successively optimized after moving one control point.
The user can furthermore specify whether to use cotangent or uniform weights for eq. 2. For the latter, he/she also needs to be specify which value to use (default is set to 1). We decided to add this feature, so the user can compare how the resulting deformation is influenced by the weights.

5 Experiments & Analysis

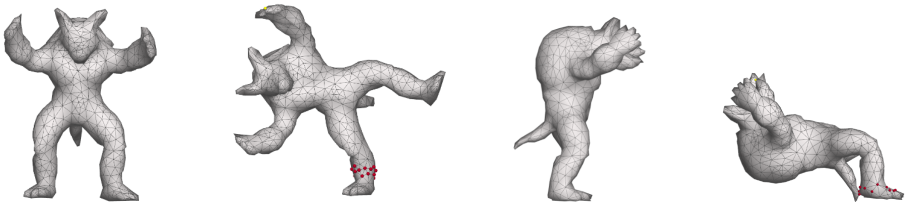
In the following we present different deformation results obtained with our implementation.

[1] `http://eigen.tuxfamily.org`
[2] `https://github.com/alecjacobson/geometry-processing-deformation`
[3] `https://libigl.github.io/`

Figure 1 show example deformations of a mesh consisting of roughly 1k vertices and compares them to similar deformations from [7]: As in the reference paper, we obtain natural deformations for arbitrary triangular meshes by only translating corresponding control points. Note that no skeleton had to be specified in order for this to work.

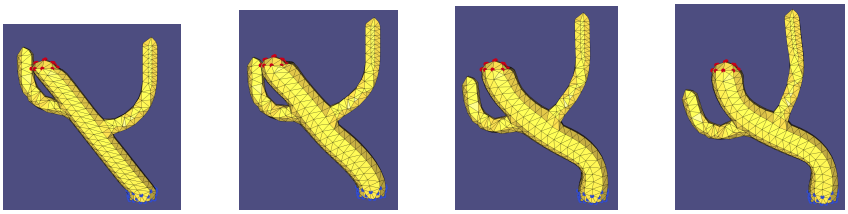


(a) (b) (c) (d)



(e) (f) (g) (h)

Figure 1: Example deformation from our application ((a) - (d)) compared to results from [7] ((e) - (h)). (a) and (c) show views of the original model; the rest of the images display editing results. In our application the static constraints are denotes in blue and the handle anchors in red.



(a) 1st iteration (b) 8th iteration (c) 30th iteration (d) 60th iteration

Figure 2: Successive iterations of the as-rigid-as-possible optimization. The original mesh is shown in figure 1.

Our application runs the iterative optimization scheme continuously, as described in the previous section. Figure 2 shows an example of how an example mesh develops over the time.

The chosen weighting scheme in eq. 1 influences the resulting deformation, as demonstrated in figure 3. Using uniform weights in the deformation energy only considers the topology of the mesh, but not the geometry [2]. Cotangent weights approximate the geometric properties of the mesh better, such that the optimization is as mesh-independent as possible. This effect is clearly visible in figure 3.

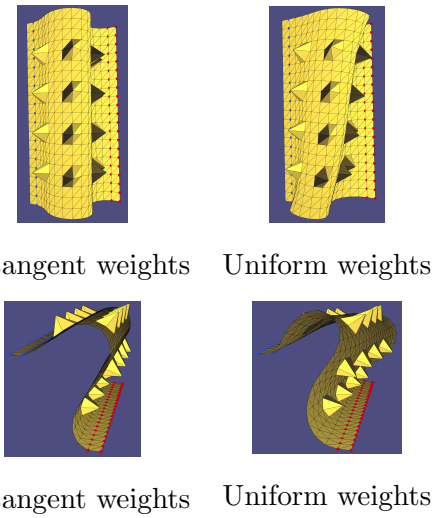


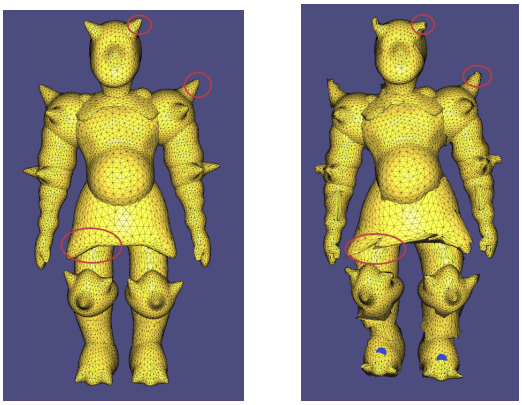
Figure 3: Demonstration of how the weighting scheme in the energy formulation influences the resulting deformation, using cotangent and uniform weights ($w_{ij} = 1$).

6 Conclusion

We conclude that our application allows an interactive manipulation of small meshes ($\approx$ 1000 vertices) and even can work on some larger meshes. The problem with the latter is the very slow precomputation step and some misbehaviour in low volume areas, like spikes and thin clothing (fig. 4). A solution to minimize the computation time would be to leverage GPUs as proposed by Zollhöfer et. al [8]. Using an influence area, e.g. vertices that are far away from the handles should not be affected by the optimization, could also increase the optimization speed and even reduce the misbehaviour in some regions. For example in fig. 4 the head should not be affected by the manipulation of the feet, hence keeping its original shape.

Currently the mesh is expected to be closed but it is possible to remove this constraint by introducing a prepossessing step to close the mesh or better handle boundary cases with different weighting schemes.

An interesting addition to the possible deformation actions would be a rotational handle interaction to allow 6 DoF mesh manipulation. This should be quite simple to integrate, because it most likely is only a change in user interaction not in the optimization itself.



(a) Original mesh (b) Deformed mesh

Figure 4: Mesh with more polygons. We can see that for example the spikes don't get deformed correctly.

References

[1] Marc Alexa, Daniel Cohen-Or, and David Levin. As-rigid-as-possible shape interpolation. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 157–164, 2000.

[2] Uwe Hahne. Weighting in laplacian mesh editing. Master's thesis, Bauhaus-Universit at Weimar, 2006.

[3] Takeo Igarashi and Yuki Igarashi. Implementing as-rigid-as-possible shape manipulation and surface flattening. *journal of graphics, gpu, and game tools*, 14(1):17–30, 2009.

[4] Takeo Igarashi, Tomer Moscovich, and John F Hughes. As-rigid-as-possible shape manipulation. *ACM transactions on Graphics (TOG)*, 24(3):1134–1141, 2005.

[5] Ya-Shu Liu, Han-Bing Yan, and Ralph R Martin. As-rigid-as-possible surface morphing. *Journal of computer science and technology*, 26(3):548–557, 2011.

[6] Olga Sorkine. Laplacian mesh processing. *Eurographics (STARs)*, 29, 2005.

[7] Olga Sorkine and Marc Alexa. As-rigid-as-possible surface modeling. In *Symposium on Geometry processing*, volume 4, pages 109–116, 2007.

[8] Michael Zollhöfer, Ezgi Sert, Günther Greiner, and Jochen Süßmuth. Gpu based arap deformation using volumetric lattices. In *Eurographics (Short Papers)*, pages 85–88, 2012.