Eurographics Symposium on Geometry Processing 2012
Eitan Grinspun and Niloy Mitra
(Guest Editors)

Volume 31 (2012), Number 5

Modeling Polyhedral Meshes with Affine Maps

Amir Vaxman

TU Vienna
avaxman@geometrie.tuwien.ac.at

Abstract

We offer a framework for editing and modeling of planar meshes, focusing on planar quad, and hexagonal-dominant meshes, which are held in high demand in the field of architectural design. Our framework manipulates these meshes by affine maps that are assigned per-face, and which naturally ensure the planarity of these faces throughout the process, resulting in a linear subspace of compatible planar deformations for any given mesh. Our modeling metaphors include classical handle-based editing, mesh interpolation, and shape-space exploration, all of which allow for an intuitive way to produce new polyhedral and near-polyhedral meshes by editing.

Categories and Subject Descriptors (according to ACM CCS): I.3.5 [Computer Graphics]: Picture/Computational Geometry and Object Modeling—Shape modeling, Shape editing Shape space, PQ/PHex meshes, design exploration

1. Introduction

Meshes with planar faces (classically and hereby denoted as *polyhedral meshes*) have recently come into prominence, for their benefits in the field of freeform architectural design, where they can be manufactured with relative ease. Designers and modelers thus require tools in which they can manipulate these planar meshes freely and intuitively, while maintaining the planarity constraints. There are several mesh modeling metaphors that designers would usually utilize. A common approach is handle-driven editing, in which vertex displacements, and a given Region-of-interest (ROI) are supplied, and the result mesh adheres to these constraints, while minimizing a set of fairness energies, such as the rigidity, or similarity, of edited faces with relation to the original mesh, fairness of curves on the surface, *et cetera*. Positional constraints, and sometimes scaling or rotational constraints can usually be incorporated into such systems with ease. A deformation tool is most often equipped with an interpolation tool, that allows the designer to navigate the range of middle shapes between two or more boundary shapes. Recently, designers have expressed interest in the ability to explore the shapes within the continuous range of a given initial mesh, and within certain compatibility conditions and fairness measures, without explicitly determining positional constraints.

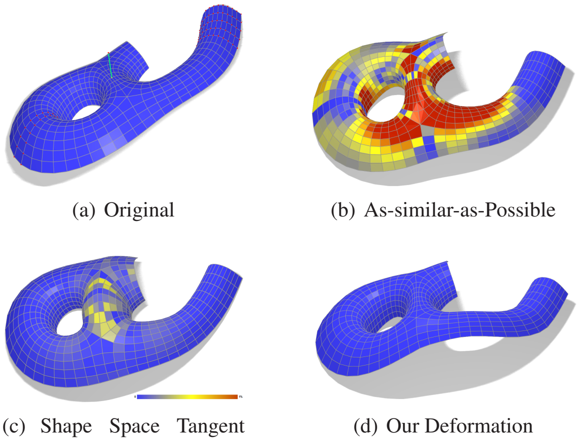
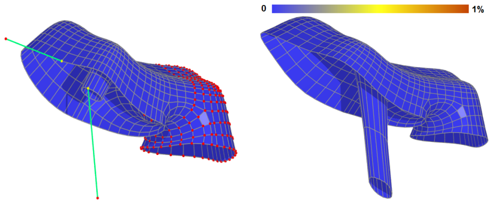
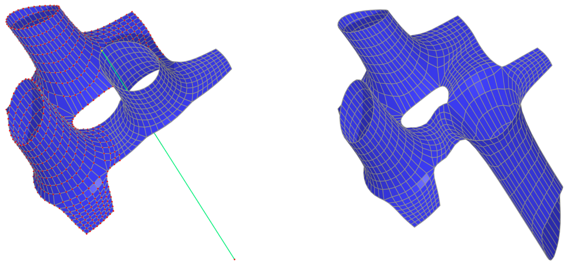

(a) Original
(b) As-similar-as-Possible
(c) Shape Space Tangent Exploration
(d) Our Deformation

Figure 1: *Comparison between several methods. On the upper right is a generalization of as-similar-as-possible deformation (in the spirit of [SA07]), which produces a smooth result, but does not preserve planarity at all. On the lower left, Planar shape space exploration by [YYPM11], which uses tangent space vectors for handle-driven exploration, and, thus, preserves planarity only up to first order. Our method, on the lower right, clearly preserves planarity, while still producing a smooth and intuitive result.*



(a) Original (b) Deformation

Figure 2: *As-rigid-as-possible deformation of the Half Tunnel model.*



(a) Original (b) deformation

Figure 3: *ARAP Deformation of the Gitter model. Notice that the pillar inverts nicely, while maintaining planarity and mesh quality.*

The majority of deformation methods in the literature work with triangular meshes for which the planarity constraint is trivial, and never regarded. Unfortunately, when dealing with general polyhedral surfaces, the planarity constraint is usually a third-order polynomial constraint in the coordinate values of the vertices of a mesh; therefore, is not trivially accommodated using any of these common techniques. However, the full subspace of planarity-preserving deformations is quite large, and contains many deformed meshes, in which the shape of the resulting faces is not so favorable. In this work we limit ourselves to deformations in which there is always a *single* affine transformation between corresponding polyhedral faces, which is a meaningful enough subspace to work with, and which allows for intuitive and large-scale deformations, with the mere price of linear compatibility constraints. Affine maps have the advantage of always preserving the planarity of faces *exactly*, and contain most adequate desired transformations within their range. It is important to mention that, in practice, most meshes are only near-polyhedral, with a given tolerance for the planarity of faces. Fortunately, except for extreme cases, this near-planarity would be preserved as well, especially when fairness measures are taken. Planarity of a quad face is measured by the percentage of the distance between the diagonals to the average diagonal length. Planarity of higher degree faces is measured as the average of the planarity values for each four consecutive vertices. The usual reasonable tolerance is 1%.

1.1. Our Contribution

We propose a novel editing framework, with face-based affine maps, which equips the designer with the following sets of tools:

- Handle-based deformation of planar meshes.
- An interpolation operator that allows for blending between different poses of the same topological mesh.
- Planar shape space exploration of the surroundings of a given mesh, with respect to several fairness measures.

We show that it is easy to obtain good results with these maps, in the price of linear compatibility constraints. Therefore, we can incorporate the richness of existing deformation metaphors for our planarity constraints. This paper is built in the following manner: In Section 2 we explain the basics of modeling with face-based affine maps. In Section 3 we present the handle-driven tool, in Section 4 the interpolation algorithm, and in Section 5 we construct the planar shape space exploration method.

1.2. Related work

Polyhedral Meshes. Polyhedral meshes rose into practical prominence only recently. The greatest emphasis was put on *planar quad* (PQ) meshes. Liu *et al.* [LPW*06] investigated their uses for architectural design, and presented an optimization process to obtain them from general quad meshes. Pottmann *et al.* [PLW*07] investigated the creation of multilayered construction with PQ meshes via the practice of *mesh offsets*. PQ meshes are commonly regarded as a discretization of *conjugate curve networks* [BS08] on surfaces. This is the principle behind the works of Liu *et al.* [LXW*11], and of Zadravec *et al.* [ZSW10], the latter creating quad-dominant networks. Special cases of PQ meshes, such as *circular* and *conical* meshes are considered to be especially beneficial ([BS08], [LPW*06]), since they are discretizations of curvature line networks, which are also a continuous special case of conjugate curve networks [dC76].
Hexagonal and hexagonal-dominant meshes have been scarcely investigated so far in a practical framework. Although they possess natural offsets, having minimal vertex degrees, and thus are paramount to the discretization of certain types of surfaces (e.g., minimal surfaces [PLW*07] and constant curvature surfaces [Mue11]), they are difficult to compute [WLY*08]. Unfortunately, the benefit of low vertex degree is paid with the obligation to have nonconvex faces in negative curvature regions. Pure hexagonal meshes have also been studied in the context of parametrization [NPP+12].

Mesh Deformation: A major research venue of geometry processing, mesh deformation is almost ubiquitously studied in the context of triangular meshes. A survey of most recent works can be found in [BKP*10]. We mention a few works to which our work relates. Sorkine and Alexa [SA07] try to achieve rigidity of deformation by minimizing the deviation of the deformation of the one ring of every vertex from a rotation, by iterating between local steps, in which the best fitting orthonormal transformation is found for a given transformation, and a global step, in which the mesh vertices try to adapt to these ideal rotations as best as possible, in the least squares sense. This approach was also used by [LZX*08] for mesh parametrization, incorporating conformal deformations. Lipman *et al.* [LSLCO05] attached discrete frames per vertex, deformed the meshes by updating these frames, and reconstructed through LS as well. Chao *et al.* [CPSS10] minimized the deformation differential dF of every triangle from the orthogonal group $SO(n)$ by Gauss-Newton minimization, and Fröhlich and Botsch [FB11] minimized discrete shell energies by optimizing the change in lengths and dihedral angles.

Deformation of polyhedral meshes: planarity-preserving deformation, or rather the creation of new polyhedral meshes from existing ones, are rare in the literature. Direct and simple approaches were introduced, such as mesh parallelism [PLW*07], and local modifications with a projective geometric framework [Hof11], and both indicated that they possess a limited capacity for general mesh design. The generalization of triangular methods in a straightforward manner will not preserve planarity in general, as it is not enforced as a condition (see Figure 1 for comparison). Even if a single set of frames, or rather a deformation Jacobian, is manipulated per face, the common least-squares integration step usually harms the planarity, which is never a problem for triangular meshes. The situation is even more complex in the presence of near-polyhedral faces, with no certain way to work with frames and induced frame coefficients. Our method enforces a single affine map per face, and thus preserves the planarity as a condition, even if rigidity or conformity should be less optimal as a result. The recent *shape space* metaphor [KMH07], which treats meshes of a given topology as points in a higher dimensional manifold, and, consequently, local deformations as tangent plane vectors, has been applied in a recent work [YYPM11], with a focus on planar mesh space exploration. At each point M on the shape space, a tangent vector is a local deformation which is orthogonal to the constraints gradients, i.e., does not change the constraint energy $E(M)=0$. Furthermore, a second-degree osculant to the shape space at M is computed, in order to achieve better contact with the shape space while deforming. Fairness energies can be defined on this space, and exploration of the *intrinsic Hessian* of these energies allows a designer to choose meshes in the surrounding of a given origin mesh, while preserving planarity and fairness. A handle-driven approach is also possible, with constraint energy minimization. However, planarity is only preserved to a second order, and the computation of the osculant and the tangent plane are cumbersome in both memory and space, and must be recomputed for each new origin mesh. Our method contains only linear constraints, and thus, we neither require the need for an osculant, nor recompute the tangent space per origin, as it is constant throughout. See a comparison of [YYPM11] with our method in Figure 1.

2. Modeling with Compatible Affine Maps

Given a single original polyhedral mesh P, i.e. a set of faces F, edges E, and vertices V with initial vertex positions $\{p_v\}$, we would like to obtain a new polyhedral mesh Q, with the same connectivity, and vertex position $\{q_v\}$. We assign an affine map to every face f of the mesh, that encodes the transformation of the vertices of a planar face in P to Q. These Affine maps are usually represented using 3×3 matrices A_f and row translation vectors t_f s.t. every point on the face f is transformed thus:

$$q_v = p_v A_f + t_f,\ v \in f \tag{1}$$

Instead of parametrizing the deformation space with the values $\{A_f, t_f\}$, it is more convenient to work directly with the matrices A_f and the resulting vertex positions q_v, since translations have no effect on the mesh qualities sought after, and since this representation is easy to work with. Naturally, not all parameter values are possible (or desired). However, we get the following set of linear compatibility conditions, that ensure that a parameter value is a valid deformation of the initial mesh: for every edge $e_{ij} \in E$, comprising vertices p_i, p_j, and which is adjacent to faces f, g, we must hold:

$$(p_i - p_j)A_f = (p_i - p_j)A_g = q_i - q_j. \tag{2}$$

These conditions simply ensure that affine maps transform both sides of an edge to the same vector, and that the edge vectors are an exact form (i.e., are integrable to actual vertex positions in space). The entire set of viable meshes from the original mesh is, therefore, spanned by the null space of a compatibility matrix C:

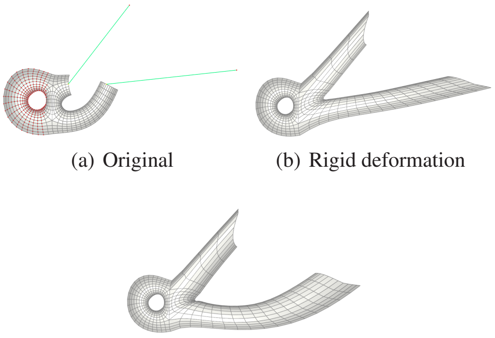
$$C\begin{pmatrix} A \\ q \end{pmatrix} = 0, \tag{3}$$

representing Equation 2. This constitutes as the linear *shape space* of affine-equivalent planar meshes, which we can explore with greater ease than the more general nonlinear planarity shape space. Notice that, since all triangles are affine-equivalent, all triangle meshes are trivially part of the same linear shape space, which can easily be represented by every vertex displacement configuration in $\mathbb{R}^3$. We will return to the shape space exploration metaphor in Section 5

3. Handle-driven shape design

Using the handle-driven metaphor, the designer first picks a ROI on the mesh, which will be the only deformable region, and then manipulates vertex positions inside the ROI, seeking to create a new mesh, adhering to some fairness measures as best as possible. We explore two such measures:

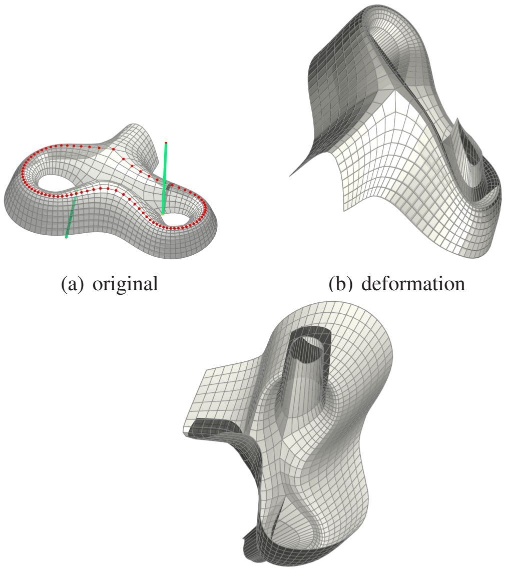
1. As-rigid-as-possible (ARAP) deformation, where maps seek to be as pure rotational as possible.



(a) Original (b) Rigid deformation

(c) Conformal deformation

Figure 4: *ARAP and ASAP Deformations of the Six model. The difference is evident in the stretching of the arm.*



(a) original (b) deformation

(c) another view

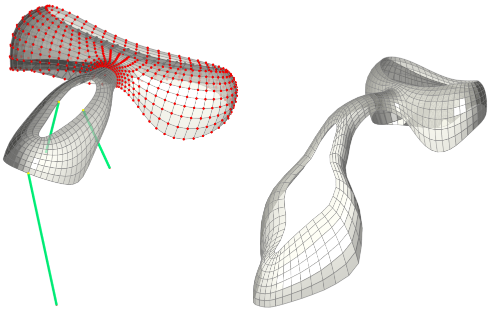
Figure 5: *ASAP deformation of the Aquadom model. The ridge is kept intact, one pillar is stretched, and the other is inverted.*

2. As-similar-as possible (ASAP), where maps seek to be conformal maps (i.e., rotational with uniform scale).

In our handle-driven approach, we follow the local-global map optimization method used by [SA07] and [LZX*08]. When the ROI and vertex displacement are chosen, we alternate between the following two steps, until convergence is reached:

- For a given affine map A_f, find the closest map T_f that perfectly preserves the energy.
- For a given desired map T_f, find the closest compatible set $\{A_f, q_v\}$

We next describe how both of these steps are carried out.



(a) Original (b) ASAP deformation

Figure 6: *ASAP deformation of the Yas model.*

3.1. Best approximating map

For a given affine map A_f, we find the closest map T_f, in the least squares sense, which perfectly preserves the fairness measure we seek. i.e., we seek T_f s.t. $E(T_f) = 0$, for a fairness measure E, and s.t. $T_f = argmin \left\|A_f - T_f\right\|^2$. Suppose that the positive singular value decomposition of A_f is $A_f = U\Sigma V^T$. It is then known that:

1. For the rigidity fairness measure, i.e. $E(T_f) = (T_f^T T_f - I)^2$, the best solution is obtained at $T_f = UV^T$. This is also known as the *Orthogonal Procrustes Problem*.
2. For the similarity measure, i.e. $E(T_f) = (T_f^T T_f - D)^2$, where D is scalar diagonal, the best solution is obtained at $T_f = USV^T$, where S is a scalar diagonal matrix as well. The scalar value S is the average of the diagonal (and only nonzero) values in Σ_2, the 2×2 matrix of the singular values of the transformation restricted to the plane.

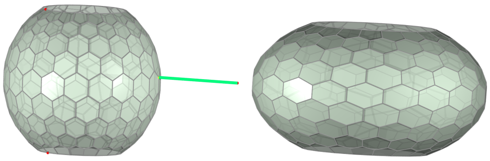
These are the same T_f matrices obtained by the analysis in [LZX*08].

3.2. Best approximating mesh

Using the prescribed affine maps T_f, computed in the former step, we would like to find the A_f which minimize the least squares distance to the perfect affine map, under the compatibility constraints C:

$$A_f = argmin \sum_{f\in F} w_f \left\|A_f - T_f\right\|^2,\ C\begin{pmatrix} A \\ q \end{pmatrix} = 0 \qquad (4)$$

The weights w_f are inverse face areas (measured approximately when the face is near-planar). This linear least-squares problem with linear constraints can be represented in a single linear system (see Appendix *A* for the derivation). Positional constraints can be moved to the right side of the equation, and their respective columns can be removed from this linear system. Upon the choice of the ROI, and the identity of the deforming vertices, the matrix representing this linear system remains constant and symmetric throughout



(a) Original (b) ASAP deformation

Figure 7: *ASAP deformation of a convex hexagonal mesh. Notice that the mesh strives for convexity, because negative curvatures are not achievable using convex faces, and affine maps preserve convexity.*

the iterations (albeit not positive-definite in general). Thus, we can pre-factor it with LDL^T factorization [Wat91], and a single iteration would then only cost the price of three matrix multiplications (two triangular, and one diagonal).

We iterate both of these steps until $max(\|P_{i+1} - P_i\|)$ has reached a tolerance value (set to 10^{-4} in all of our experiments). Usually, no more than 8–9 iterations are required for full convergence, even for large deformations. See Figures 2, 3, 4, 5, and 6 for examples. Notice that this approach is also reminiscent of the energy $|df - R|^2$ minimized by [CPSS10]. As they deal with triangular meshes, df is naturally piecewise constant, whereas we uphold this constness with the variables $A_f = df$.

There is, nevertheless, an interesting caveat to this approach: when manipulating meshes with many edges to some faces, such as hexagonal meshes (see Figures 7 and 8), and in certain classes of quad meshes, the constraint matrix may become overdetermined, and would not produce a consistent shape space for any given set of positional constraints. This is mostly evident in closed meshes of a higher genus. An exact analysis of the null space of the matrix C is given in Appendix B. In these cases, there cannot be a true single exact affine map transforming each face. Mathematically, the entire system is then solved by least squares to produce an approximate affine transformation of each face, and the constraints' error will grow with severely conflicting right-hand sides. In practice, we witnessed negligible constraint errors, if any, mostly occurring when we specifically tried to "tear the mesh apart", e.g. constrain several close vertices badly. In all the examples given, the constraint error $max(\|Cx\|)$ (where x is the configuration vector) was always less than 10^{-5}, and on average below 10^{-8}.

3.3. Bending Energies

Botsch *et al.* [FB11] seek to minimize both the discrete stretch energy (the change in edge lengths), and the discrete bending energy (the change in dihedral angles). However, our face-based energy only pertains to the former, as each face is studied separately, up to integrability. In order to minimize the bending energy in our linear setting, we introduce a regularization in the form of the minimization of the Dirichlet energy of the affine maps, when treated as a matrix field over triangles:

$$E_d(A) = \sum w_{fg}(A_f - A_g)^2, \qquad (5)$$

for each two adjacent faces f, g. The minimizers of this energy are affine maps which are *harmonic* with respect to the 2-form Laplacian [DKT05], when the weights w_{fg} are the values of the inverse Hodge star: $w_{fg} = \frac{\|e_{fg}\|}{\|e^*_{fg}\|}$, e^*_{fg} being the dual edge to e_{fg}. This regulization is a common practice in linear deformation methods. For instance, if we regard the affine maps as changes to frame systems, this energy can perhaps be viewed as a way to minimize the difference between transformed frames, with relation to their original values [LSLCO05]. Thus, when given a perfect map T_f, we now minimize the following energy:

$$A_f = argmin \sum_{f \in F} w_f \|A_f - T_f\|^2 + \lambda \sum w_{fg}(A_f - A_g)^2,$$
$$C \begin{pmatrix} A \\ q \end{pmatrix} = 0 \qquad (6)$$

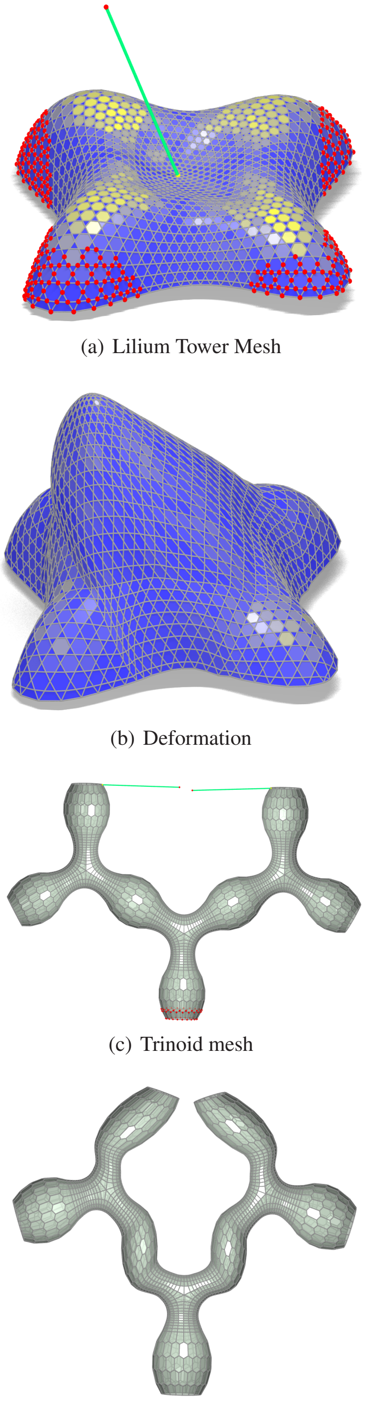
Lower values of λ will favor face shape, and higher values will favor equal transformations for all faces. We typically use λ =1–3 in our experiments.

3.4. Planarity of Curves

The affine framework can easily accommodate a constraint in which certain curves should remain planar throughout (such as floor, or ceiling, curves). If the plane in question is given, it is a simple extra linear condition. In the general case, an affine map can be attached to each such curve, as if it is a single planar polygon, like a face of the mesh. Of course, this curve would then be subjected to this single affine map, which might be prohibitive. See Figure 9 for planar-curve constrained deformations.

4. Interpolation

A modeling framework usually requires the ability to study the middle shapes between an original mesh and its deformation. Since our compatibility space is linear, any linear combination of the sort $A_f(t) = tI_f + (1-t)A_f, t \in [0,1]$ would produce a compatible polyhedral mesh. However, these meshes are not geometrically reasonable. We therefore follow the interpolation method of [ACL00], in the following manner: We take the polar decomposition of every $A_f = Q_f R_f$, and set our desired middle matrix to be $T_f = ((1-t)I + tQ_f)R_f(t)$. If R_f is represented by the axis-angle pair $(\hat{v}, \alpha)$, then $R_f(t)$ is the partial rotation matrix represented by the pair $(\hat{v}, t\alpha)$. We then proceed to solve one linear system, as in Section 3.3, with the desired matrices T_f. Notice that in general, large rotations cannot be interpolated satisfactorily using this method, (as noted by [FB11], for example), since it always takes the shortest rotation. This



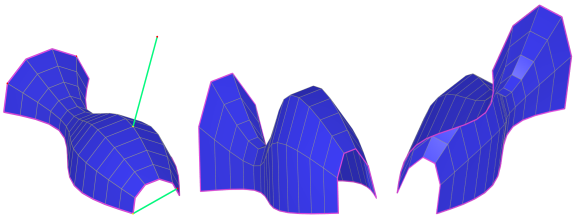
(a) Lilium Tower Mesh

(b) Deformation

(c) Trinoid mesh

(d) Deformation

Figure 8: *ASAP Deformations of hybrid-hexagonal meshes. To the Lilium tower mesh, deformation seems to planarize some of the hexes, which is actually an effect of stretching the hex along the approximate hex plane, increasing diagonal length without increasing diagonal distance.*



(a) Original (b) Deformation (c) Another view

Figure 9: *Deformation of the Vase model, maintaining the planarity of the base and the opening curves.*

can be alleviated somewhat by using higher λ values. However, this interpolation is quite efficient, since we need to factor the energy and constraint matrices only once. See Figures 10 and 11 for interpolation results, and the accompanying video. We can also successfully achieve *extrapolation* of meshes, beyond the boundaries of 0 and 1.

5. Shape Space Exploration

We next adhere to the metaphor of the *shape space*. Our ambient space is the entire set of matrices and vertex positions $\{A_f, q_v\}$. Our *Compatible* shape space is that of these sets which reside in the null space of the constraint matrix C. Fortunately, the null space is linear, and, therefore, our space has no curvature, unlike the planar shape space described in [YYPM11]. We thus acquire the full capacity of their exploration process, without the need for expensive computation of a second-order osculant.

Picking the origin, where all A_f are identities, we parametrize the null space of C, which also serves as the tangent space of the origin, with null base vectors $[e_1, e_2, ...e_n]$. It is worth noticing that, once we obtain these base vectors, we can directly alter their linear combinations for vertex displacements, regardless of the affine maps, and vice versa. As detailed in [YYPM11], we seek to minimize fairness energies defined in the entire ambient space, by the restriction of their gradients and Hessians to our shape *hyperplanar* space. If a point on the shape space is parametrized as $x = x_0 + \sum_{i=0}^{n} e_i u_i$ (we slightly abuse notation here, and use e_i to denote only its vertex position members), then the second order approximation of a function on this plane is:

$$\begin{aligned} F(x) &= F(x_0) + \sum_{i=0}^{n} (\nabla F^T \cdot e_i) u_i + \frac{1}{2} \mathbb{U}^T H_F^I \mathbb{U} \\ H_F^I &= [e_1, e_2, ...e_n]^T H_f [e_1, e_2, ...e_n] \end{aligned} \qquad (7)$$

Where $\mathbb{U}$ are the coefficients $\{u_i\}$, and H_F is the Hessian of F. For brevity, we explore the similarity and the Dirichlet energies, presented in section 3, which we have referred to before, although the full fairness energy range of [YYPM11] is available to us.

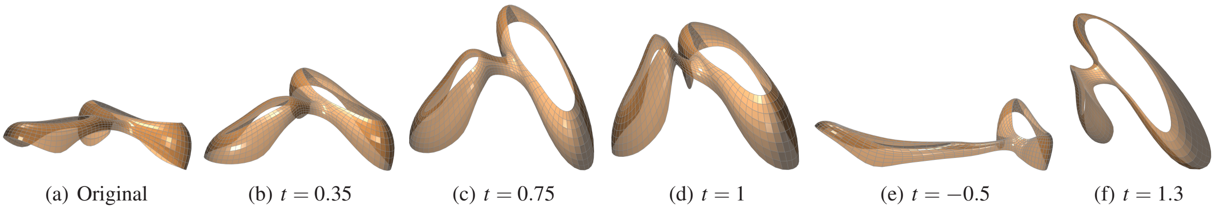
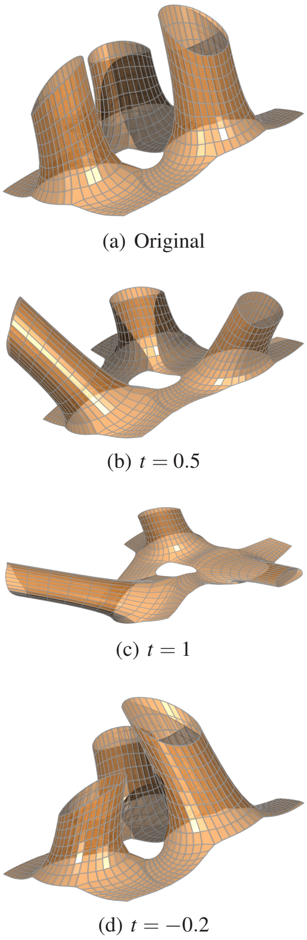

(a) Original (b) $t = 0.35$ (c) $t = 0.75$ (d) $t = 1$ (e) $t = -0.5$ (f) $t = 1.3$

Figure 10: *Deformation ($t = 1$), interpolation, and extrapolation. The planarity is naturally kept, and the results produce a smooth transition.*


(a) Original

(b) $t = 0.5$

(c) $t = 1$

(d) $t = -0.2$

Figure 11: *Interpolation and extrapolation of the Gitter model. The extrapolation provides an natural extension to the movement in the pillars.*

5.1. Subspace Exploration

After choosing a mixture of fairness energies: $F(x) = \sum_i \lambda_i F_i(x)$, we proceed by exploring directions on the shape hyperplanar space with extremal second derivatives. By looking at several low eigenfunctions of the intrinsic Hessian H_F^I at the origin, we are able to explore the surroundings of the the origin, by navigating upon these eigenfunctions. In fact, because of the linearity of space, and its promise of compatibility, the user is not very limited to navigation in a confined boundary, as made essential in [YYPM11]. However, the second order approximation of fairness energies deteriorates when we travel far from the origin. Fortunately, since the base we computed is constant throughout the space, it is easy to recompute the intrinsic Hessian, and its lowest eigenfunctions, from any given new point. We allow for navigation in the two-dimensional plane spanned by two low eigenvectors of the Hessian. See Figure 12 and 13 for examples. One should note that while navigation is set in a linear space, and subject to fairness energies, the near-planarity can be compromised by certain large-scale affine maps. This can also be regularized with proper energies. The disadvantage to the shape-space approach is that we require the *exact* null space of the matrix C, which might be trivial for several classes of meshes (see Appendix B for an exact analysis), but which is adequetly rich for typical architectural quad and hexagonal-dominant meshes. Fortunately, Approximate affine maps, which can be interpreted as working with the smallest singular values of C, are quite well-behaved in handle-driven exploration. We therefore conjecture that using the corresponding basis vectors would enrich the shape space with minimal harm. In this work, we only utilize exact null-space basis vectors.

Normal fibration space. It is worth noting, that an affine map has nine variables, whereas our degree of freedom is six. The extra degrees of freedom comprise the action of the affine map on the normal to an original face, which does not alter the result. That is a three dimensional *fiber space* for every mesh point on the shape space. Handle-driven exploration and interpolation disregard this problem, since the energies regulate the normal behavior automatically. In the shape-space approach, one should note that nontrivial eigenfunctions might result in walking on normal fibers. However, this is more a matter of redundancy, as these eigenfunctions simply inflate the null space of compatibility, and one can always find proper eigenfunctions to explore.

6. Discussion and Future Work

Implementation Details: Our computation code (the sparse LDL^T factorization and linear solving) was written in MATLAB, and exported to C++. The handle-driven approach is essentially real-time, but the shape space approach require pre-computation of the null space of a matrix, and eigenvalues extraction of the dense Hessian. This typically required several minutes for 1–2k meshes on an i7 CPU machine with 4Gb memory. However, once extracted per mesh, the exploration is real-time as well.

Discussion: We have shown several sets of tools for the design and editing of polyhedral, and near-polyhedral, meshes. The handle-driven tool is more light-weight, and suitable for large deformations and interpolation needs. However, as currently presented, it is limited to energies which can be put in terms of desirable affine maps per face, or form linear constraints. Of course, other intricate energies can be used, in a full scale optimization, with these linear constraints, but that is out of the scope of our paper. Our second approach removes the constraints by parametrizing the shape hyperplanar space, and thus allows for more freedom of expression, with all types of fairness energies involved, and also including the possibility of handle-driven approach by its own (as an unconstrained optimization process). However, this parametrization comes at a price of $O(n^2)$ memory and space complexity, which serves as an improvement from the method of [YYPM11], but might nevertheless be cumbersome for large meshes. The lightweight handle-driven approach also carries the price of sparse matrix factorization, and several repeated iterations, but its complexities are still far lesser than the ones the shape space approach introduces. We believe that the combination of these approaches provides a good set to work with; The user might manipulate a coarse mesh with the full extent of the shape space exploration, further subdivide it, and might then continue with the more lightweight handle-driven approach.

Using affine maps in a linear space also has some limitations. As we do not have the full scale of polyhedral deformations, we cannot totally guarantee properties other than planarity; for instance, concyclity, which, for some applications, needs to be exact. However, no other existing method can guarantee above second-order osculation with other properties, without resorting to full-blown constrained optimization, and our results are adequate in comparison. Another interesting limitation is the adherence to a specific mesh topology, which is an obiquitous problem for explicit deformation and shape-space algorithms. When mesh elements become too strained, the need to remesh restarts the computation. A shape space which is either insensitive of the meshing, or easy to re-adjust, for instance by connectivity editing [PZK+11], is still an open problem. As we have discussed before, it would be interesting to see if properties like concyclity, or conicality, can be totally accommodated using other types of linear spaces as well. Also, we would like to extend the method to include other types of surfaces, which have planarity constraints, such as Functional Webs [DPW11], and for conformal/rigid parametrization of polyhedral surfaces, such as the one presented in [LZX*08] for triangular meshes.

7. Acknowledgements

This work has been supported by Austrian Science Fund (FWF) under grant P23735-N13. We would like to thank Helmut Pottmann and Johannes Wallner for their major support, and many useful comments and suggestions, Roi Poranne, Amit Bermano and Ofir Weber for their aid and valueable comments, and the anonymous reviewers for their feedback.

References

[ACL00] Alexa M., Cohen-Or D., Levin, D.: .As-Rigid-as-Possible Shape Interpolation. *Proc. SIGGRAPH*, 2000, 157–164 5

[BKP*10] Botsch M., Kobbelt L., Pauly M., Alliez P., Levy, B.: Polygon Mesh Processing. AK Peters, 2010. 2

[BS08] Bobenko A. I., Suris Y. B.: Discrete Differential Geometry: Integrable Structure. *Graduate Studies in Mathematics, American Math. Soc.* Vol. 98. 2

[CPSS10] Chao I., Pinkall U., Sanan P., Schröder P.: A Simple Geometric Model for Elastic Deformations. *ACM Trans. Graph.* (2010), 38:1–38:6. 3, 5

[dC76] do Carmo M. P.: *Differential Geometry of Curves and Surfaces*. Prentice-Hall, 1976. 2

[DKT05] , Desbrun M., Kanso E., Tong, Y., Discrete Differential Forms for Computational Modeling. *ACM SIGGRAPH Courses*, 7, 2005. 5

[DPW11] Deng B., Pottmann H., Wallner J. Functional Webs for Freeform Architecture. *Computer Graphics Forum 30* 5 (2011), 1369–1378. 8

[FB11] Fröhlich S., Botsch M.: Example-Driven Deformations Based on Discrete Shells. *Comput. Graph. Forum 30*, 8 (2011), 2246–2257. 3, 5

[Hof11] Hoffmann T.: On Local Deformations of Planar Quad-Meshes. *Mathematical Software–ICMS, ed. K. Fukuda et al.*, 2010, 167–169 3

[KMH07] Kilian M., Mitra N. J., H.e P.: Geometric Modeling in Shape Space. *ACM Trans. Graphics 26*, 3 (2007), 64:1–8. 3

[LPW*06] Liu Y., Pottmann H., Wallner J., Yang Y.-L., Wang W.: Geometric Modeling with Conical Meshes and Developable Surfaces. *ACM Trans. Graphics 25*, 3 (2006), 681–689. 2

[LSLCO05] Lipman Y., Sorkine O., Levin D., Cohen-Or D.: Linear Rotation-Invariant Coordinates for Meshes. *ACM Trans. Graph. 24*, 3 (2005), 479–487. 3, 5

[LXW*11] Liu Y., Xu W., Wang J., Zhu L., Guo B., Chen F., Wang G.: General Planar Quadrilateral Mesh Design Using

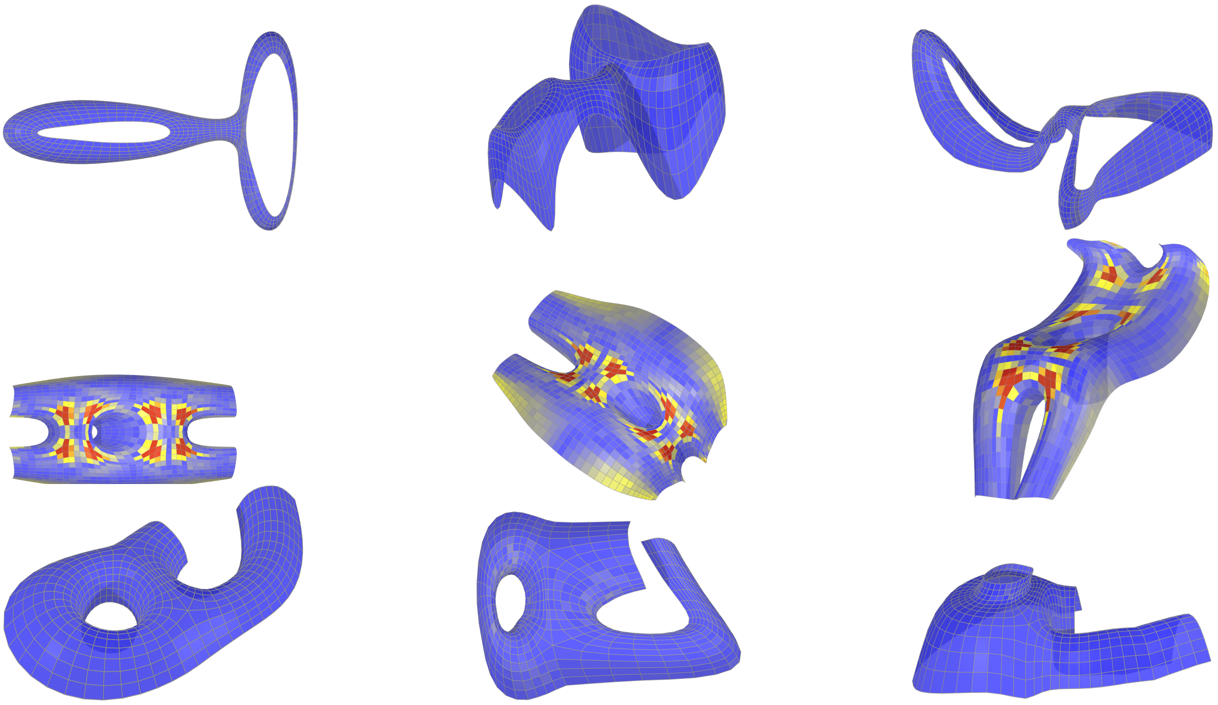


Figure 12: *Shape space exploration of several quad meshes. Originals are on the left column. Notice the Train Station model (middle), which has borderline near-planar faces, and that this property stays rather invariant.*

Conjugate Direction Field. *ACM Trans. Graph. 30*, 6 (2011), 140:1–10. 2

[LZX*08] Liu L., Zhang L., Xu Y., Gotsman C., Gortler S. J.: A Local/Global Approach to Mesh Parameterization. In *Proc. of the Symposium on Geometry Processing* (2008),1495–1504. 3, 4, 8

[Mue11] Müller, C.: Conformal Hexagonal Meshes. Geometriae Dedicata 154,1 (2011) 27–46 2

[NPP+12] Nieser, M., Palacios, J., Polthier, K., Zhang, E., Hexagonal Global Parameterization of Arbitrary Surfaces. *IEEE Trans. on Vis. and Comp. Graph 18*, 6 (2012), 865–878 2

[PLW*07] Pottmann H., Liu Y., Wallner J., Bobenko A., Wang W.: Geometry of Multi-Layer Freeform Structures for Architecture. *ACM Trans. Graphics 26*, 3 (2007), 65:1–11 2, 3

[PZK+11] Peng, C.-H., Zhang, E., Kobayashi, Y., Wonka, P. Connectivity Editing for Quadrilateral Meshes. *ACM Trans. Graphics 30*, 6 (2011), 141:1–12. 8

[SA07] Sorkine O., Alexa M.: As-Rigid-as-Possible Surface Modeling. In *Proc.* 5*th* *Symposium on Geometry processing* (2007), pp. 109–116. 1, 2, 4

[Wat91] Watkins, D. Fundamentals of Matrix Computations. New York: Wiley. 5

[WLY*08] Wang W., Liu Y., Yan D., Chan B., Ling R., Sun F.: Hexagonal Meshes with Planar Faces. *HKU CS Tech Report TR-2008-13* (2008). 2

[YYPM11] Yang Y.-L., Yang Y.-J., Pottmann H., Mitra N. J.: Shape Space Exploration of Constrained Meshes. *ACM Trans. Graph. 30*, 6 (Dec. 2011), 124:1–12. 1, 3, 6, 7, 8

[ZSW10] Zadravec M., Schiftner A., Wallner J.: Designing Quad-Dominant Meshes with Planar Faces. *Computer Graphics Forum 29(5)*, 2010, 1671–1679. 2

Appendix A: Linear Least Squares with Linear Constraints

given a linear least squares energy $E(x) = \|Ax - b\|^2$, with linear equality constraints $Cx = d$, a solution can be found with Lagrange multipliers λ by solving the following linear system:

$$\begin{pmatrix} A^TA & C^T \\ C & 0 \end{pmatrix} \begin{pmatrix} x \\ \lambda \end{pmatrix} = \begin{pmatrix} A^Tb \\ d \end{pmatrix} \tag{8}$$

If the constraints are overdetermined, or not consistent, The solution will fit the constraints in a least square manner.

Appendix B: The Dimension of the Null Space

We consider each Euclidean dimension separately, since both the constraints and the affine maps are separable. Thus, we have three variables of A per face, and one variable per vertex for q, which amount for $3|F| + |V|$ variables. Suppose that a face comprises n edges, we then have $n-1$ compatibility conditions, since the condition on the last edge is a trivial sum of the rest. In order to avoid counting the normal fibration space (see Section 5.1), we introduce one "phantom" condition per face, that transforms the normal to this face arbitrarily. We then arrive at n conditions per face, and $|H|$ conditions in total, where H is the group of halfedges. It is evident that in triangle meshes, where $|H| = 3|F|$, we have $|V|$ degrees of freedom, which reproduces the ability to move vertices arbitrarily within the trivial affine-equivalent space. Suppose that B is the group of boundary edges, we therefore obtain:

$$\#var = 3|F| + |V|, \#const = |H| = 2|E| - |B| \qquad (9)$$

In a pure quad mesh, where $|H| = 4|F|$, the null space dimension is $\chi + \frac{|B|}{2}$, where χ is the Euler characteristic. Therefore, closed quad meshes with no boundary rarely have *exact* nontrivial null spaces, although this might pose a significant problem only for the exact affine shape-space exploration process. However, architectural meshes usually have several meaningful boundaries (such as the ones presented in this paper), and, therefore, enjoy an adequate exact shape-space, as shown in the examples. Pure hexagonal meshes have an even more strict null space, but hexagonal-dominant meshes enjoy a greater possibility, because they combine low-degree faces. Notice that the said degrees of freedom are a lower bound, in a sense, as the constraints can become more dependent. For instance, the trivial space of uniformly-affine motions is always in the null space, and some symmetries may result in more dependancy as well.

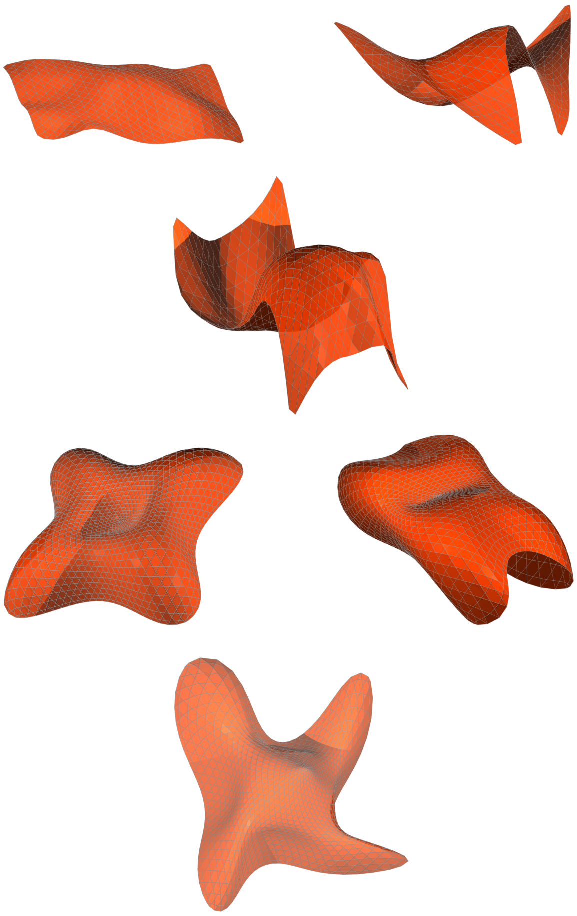


Figure 13: *Shape space exploration of general polyhedral meshes - the upper is quads and triangles, and the lower is hexagonal-triangular (Lilium model)*