Pacific Graphics 2014
J. Keyser, Y. J. Kim, and P. Wonka
(Guest Editors)

Volume 33 (2014), Number 7

Real-Time Symmetry-Preserving Deformation

Xiaokun Wu [1] Michael Wand [2] Klaus Hildebrandt [1] Pushmeet Kohli [3] Hans-Peter Seidel [1]

[1]Max-Planck-Institut für Informatik [2]Utrecht University [3]Microsoft Research Cambridge

Abstract

In this paper, we address the problem of structure-aware shape deformation: We specifically consider deformations that preserve symmetries of the shape being edited. While this is an elegant approach for obtaining plausible shape variations from minimal assumptions, a straightforward optimization is numerically expensive and poorly conditioned. Our paper introduces an explicit construction of bases of linear spaces of shape deformations that exactly preserve symmetries for any user-defined level of detail. This permits the construction of low-dimensional spaces of low-frequency deformations that preserve the symmetries. We obtain substantial speed-ups over alternative approaches for symmetry-preserving shape editing due to (i) the sub-space approach, which permits low-res editing, (ii) the removal of redundant, symmetric information, and (iii) the simplification of the numerical formulation due to hard-coded symmetry preservation. We demonstrate the utility in practice by applying our framework to symmetry-preserving co-rotated iterative Laplace surface editing of models with complex symmetry structure, including partial and nested symmetry.

Categories and Subject Descriptors (according to ACM CCS): I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Curve, surface, solid, and object representations

1. Introduction

Content creation is one of the remaining open challenges in the field of computer graphics. While current 3D modeling and rendering techniques create high-quality depictions of an extremely broad variety of phenomena already, the creation of the corresponding digital 3D models is still tedious, technical, and expensive, and a lot of recent research is focusing on improvements in this respect. One such line of work is *structure-aware* shape modeling [MWZ*13]. Structure-aware methods try to reduce modeling costs and lift interaction with digital 3D models to a higher semantic level by an analysis-and-synthesis approach: A shape (or a collection of shapes) is first analyzed in order to detect important structural invariants. Afterwards, these constrains are used to narrow down the space of possible shapes considered in editing and shape synthesis, thus permitting the user to obtain plausible shape variations more quickly.

Our paper contributes to the area of structure-aware shape *deformation*. The task here is to find and apply structural constraints during shape editing, thereby making the space of possible deformations more easily navigable.

Several such structure-aware deformation methods have been proposed in the last few years (for example, see [KSSCO08, XZT*09, ZFCO*11]). A significant idea in this domain has been introduced by Gal et al. [GSMCO09]: Their "iWires" system detects relations of Euclidean geometry (parallelity, orthogonality, symmetry) within shapes uses them as invariants during shape deformation. In particular for man-made shapes, this provides a very useful tool to fully automatically determine a large class of plausible shape variations. In more recent work [BWKS11,ZFCO*11,BWSK12, KWW*14], the model has been simplified towards preserving the *symmetry structure* of the model, which reduces the approach to a simple, abstract principle. While such a formulation is formally appealing, it is limited by a complex implementation, approximation artifacts (in least-squares formulations), and computational costs.

In this paper, we study classes of shapes of fixed symmetry structure. We consider shapes with partial symmetries, i.e., there are one or more groups of affine transformations that are symmetry groups of subset of a shape, i.e., leave these subsets invariant. In particular, this includes the important class of symmetry under rigid mappings. Extending previous work on symmetry-invariant functions on shapes [KFR04,LCDF10,OMPG13,WSSZ14], we show that

the set of all deformations that preserve the symmetries of a shape forms an affine subspace of the set of all continuous spatial deformations.

We use this insight to devise an efficient and simple algorithm for constructing a basis for these deformation spaces: We perform Poisson-disc sampling of the domain while replicating points by the group action of the present symmetry groups. Further, we associate each point with the Jacobian of the transformation to handle general Euclidean symmetry, and equip it with a Gaussian radial basis function to span spaces that are band-limiting to the user's preference.

We apply this construction to create a symmetry-preserving subspace deformation method based on Laplace surface editing (in a non-linear variant with co-rotation of the local frame, to account for large deformations). Our technique preserves symmetries *exactly* while nonetheless being able to represent low-frequency deformations with a very small set of basis functions. The basis is redundancy-free in the sense of having symmetry backed-in already; no explicit constraints need to be solved for to maintain symmetry. This, along with the subspace approach, leads to a deformation method that is significantly faster than previous approaches and at the same time is numerically robust and very easy to implement. We also believe that the idea of a direct construction of a redundancy-free basis in symmetric domains might be useful beyond the application area of shape deformation that we explore as motivating application in this paper.

In summary, we make two main contributions: First, we characterize shape spaces of fixed symmetry as affine spaces. Second, we use this observation to directly construct a basis for the space of symmetry-preserving deformations, which leads to very simple and efficient shape editing algorithms.

2. Related Work

Shape deformation has become an indispensable tool in shape editing. Early methods used low-dimensional spaces of low-frequency deformations spanned by spline-bases that were explicitly controlled by the user [SP86, Coq90]. With increasing level-of-detail, these spaces become too large to be navigated explicitly by the human modeler, which led to a wide-spread adaptation of variational deformation models that are usually based on mimicking physical processes such as elastic or plastic deformations [TPBF87, WW92], or rely on smoothness assumptions for more general shape deformation [ABCO*03, BR07]. Geometric approximation such as Laplace surface modeling or as-rigid-as-possible deformations have recently become particularly popular in this context [SCOL*04, ATLF06, SA07, BS08]. Computational costs can be reduced by subspace methods [HSL*06, HSvTP11, JBK*12, vTSSH13] that restrict the set of feasible deformations to a low-dimensional subspace. Our approach follows this line of work, but focuses on preserving symmetries during shape editing.

Structure-aware deformation has been introduced in image processing through seam-carving [AS07] for image retargeting. A similar idea for shape-resizing has been introduced by Kraevoy et al. [KSSCO08], using differential properties of shapes to detect suitable deformation directions and a vulnerability score to protect complex structures. However, unlike our approach, this method only permits axis-aligned stretching of models and only protects salient area from deformation, rather than maintaining non-local symmetry relations. Retargeting has also been extended to noncontinuous operations [LCOZ*11, BWKS11], which is beyond the scope of this paper. Other deformation constraints include slippability for joint-detection [XZT*09].

The "iWires" system by Gal et al. [GSMCO09] is based on the observation that geometric relations such as parallel lines and planes, right angles, and symmetric and regular parts are characteristic for man-made-shapes. This method uses a list of Euclidean invariants and a greedy propagation algorithm to maintain these shape properties. Most of these invariants arise from the more concise assumption of preserving the *symmetry structure* of the input model, i.e., the algebraic relation between transformations that form regular correspondences within a shape [KWW*14]. Symmetry has been utilized as a tool for structuring and guiding shape editing in a number of recent shape editing approaches: Zheng et al. [ZFCO*11] enforce symmetry of object-aligned proxies for intuitive shape manipulation. Wang et al. [WXL*11] use hierarchical propagation of attributes in objects with complex symmetry patterns for structure-preserving shape editing.

Bokeloh et al. [BWKS11, BWSK12] use translational regularity as invariant for continuous shape deformation (we do not consider their additional option to model topological changes here). The main drawback of the first method is that it is a least-squares approach. Computational costs are still considerable despite the involved numerical treatment in their paper. Symmetries are maintained only approximately and traded-off against user constraints. In practice, this leads to the problem that the user needs to choose constraints that are roughly satisfiable; unsatisfiable constraint will lead to bending artifacts. Increasing the penalties for structure constraints reduces these but a large spread in penalties soon leads to ill-conditioned optimization problems. The second method [BWSK12] avoids these by fixing the null-space of the deformation energy through a singular-value-decomposition (SVD). However, the SVD creates a dense basis, which limits the size of models that can be handled due to the $\Omega(n^2)$ costs involved for n vertices. Our new method overcomes this problem by directly constructing a sparse basis. Further, our technique handles general Euclidean motions, not being limited to translational symmetry. The work of Kurz et al. [KWW*14] is closely related to ours, however, targeting at shape matching rather than editing. A least-squares formulation that is inaccurate and multiple orders of magnitude slower than our approach is used.

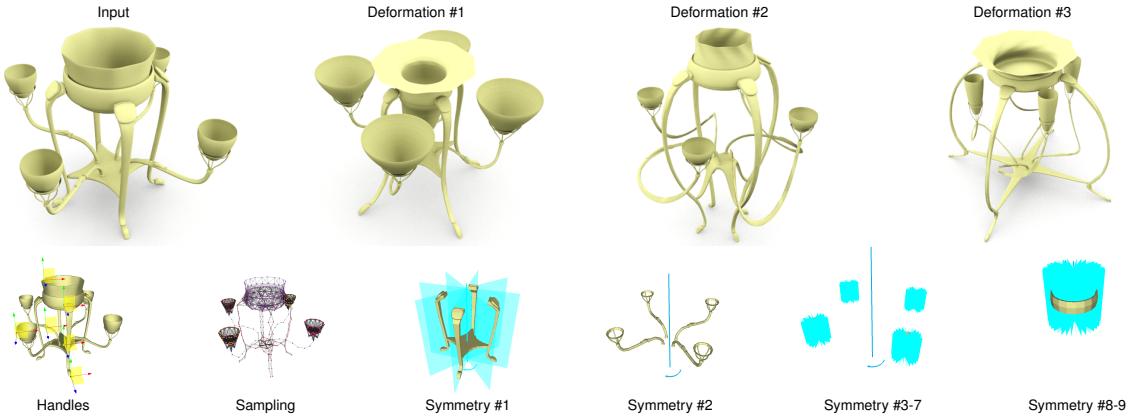


Figure 1: *Shapes generated with our symmetry-preserving modeling system are shown. These example show a combination of complex symmetries and large deformations. The symmetries are illustrated in the bottom row.*

Symmetry-invariant scalar functions have been studied by Lipman et al. [LCDF10]. Symmetry-invariant functions obviously form a linear space (linear combinations of symmetric functions maintain the symmetry property), and the paper proposes a spectral method for determining these spaces. Ovjanikov et al. [OMPG13] use the subspace property to factor out distracting variability in the context of shape matching. The concept has various further applications, such as symmetry-based shape descriptors [KFR04], and robust intrinsic symmetry detection [WSSZ14].

Our work extend these previous ideas towards *shape deformation*. We show that symmetry-preserving deformation functions form affine spaces (more specifically, the displacement fields added to the input form a linear space). Unlike the scalar case, local frames induced by the group action need to be taken into account to preserve symmetries. Assuming knowledge of the symmetry groups, our approach can directly construct a suitable basis without need for expensive eigenvalue decompositions. This construction provides a user-defined level-of-detail and, unlike spectral methods, can preserve the sparseness of traditional spatial bases for deformation fields.

3. Symmetries

In this section, we recap the definition of (partial) symmetry [MPWC12]. Let us consider a surface in $\mathbb{R}^3$ given by a two-manifold M and an embedding $x : M \mapsto \mathbb{R}^3$. An automorphism of M is a map $\varphi : M \mapsto M$ that is a homeomorphism, i.e., is continuous, bijective and has a continuous inverse. Under the composition of maps, the set of automorphisms of M forms a group that we denote by $\Psi(M)$. Symmetries of M relate to subgroups of $\Psi(M)$. Here, we are interested in symmetries induced by Euclidean motion of $\mathbb{R}^3$. A Euclidean motion is an affine map whose linear part is an orthogonal transformation. Examples are translations, reflections, and rotations as well as any composition of the these. The set of Euclidean motions forms a group $E(3)$ under the composition of maps. We consider symmetries with respect to Euclidean motion, because this leads to useful invariants for man-made shapes .part-based assembly is inherently prone to rigid redundancy).

For our experiments, we are using triangle meshes in $\mathbb{R}^3$. In this case, M is the surface mesh itself (or any isomorphic simplicial manifold) and x is the continuous and piecewise linear map that maps every vertex to its positions in $\mathbb{R}^3$.

Symmetry Any Euclidean motion g can be composed with the embedding x, which results in a new map $g \circ x : M \mapsto \mathbb{R}^3$. In general, this map will map M somewhere into space. However, there are surfaces and Euclidean motions such that the motion maps the surface onto itself. We are interested in these surfaces and motions. Loosely speaking, a (Euclidean) symmetry of the embedded surface (M,x) is subgroup G of $E(3)$ such that every $g \in G$ maps $x(M)$ to itself. More formally, we say that a subgroup $G \leq E(3)$ is a symmetry of (M,x) if there is a subgroup $\Phi \leq \Psi(M)$ such that for every $g \in G$ there is a $\varphi \in \Phi$ such that

$$g \circ x = x \circ \varphi \tag{1}$$

and the map $G \mapsto \Phi$ induced by this relation is a group isomorphism. The equality in (1) means that $g \circ x$ and $x \circ \varphi$ are the same map from M to $\mathbb{R}^3$.

Remark 1. *Note that this concept of symmetry also works if we relax the assumption that the surface is embedded. It suffices that x is locally an embedding, e.g., an immersion.*

Partial symmetry In addition to symmetries of the whole object, we consider symmetries of parts of the object and call them partial symmetries. This makes the concept more powerful as objects often exhibit only partial symmetries. To define partial symmetries, we consider a submanifold N of M, which need not be connected. Then, the restriction $x_{|N}$

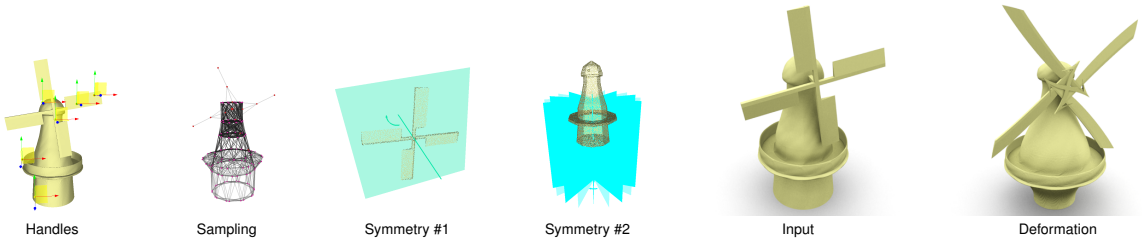


Figure 2: *A deformation of a wind mill model generated with out symmetry-preserving modeling system is shown.*

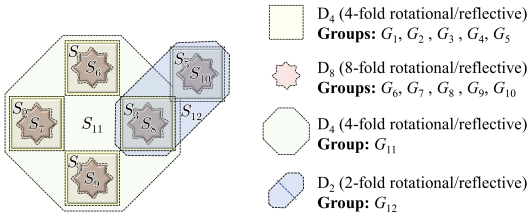


Figure 3: *Nested and overlapping symmetries. Regions with different symmetries overlap. For example, region S_1 is four-fold symmetric, and the subset S_6 is 8-fold symmetric. The 2-fold symmetry of S_{12} overlaps partially with S_{11}.*

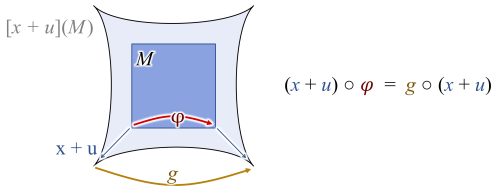


Figure 4: *If the symmetry transformation commutes with the deformation $x+u$ (the automorphism φ in the input domain turns into the extrinsic map g here), the deformed shape $[x+u](M)$ will have the same symmetry as $x(M)$.*

of x to N is an embedding of N in $\mathbb{R}^3$. A symmetry G of $(N, x_{|N})$ is a partial symmetry of (M,x).

Symmetry detection Various symmetry detection methods have been proposed in literature (see for example [MPWC12] for a recent survey). Detection is not a focus of our paper; we use both manual annotation as well as automatic detection based on Tevs et al.'s algorithm [THW*14] for our experiments. The resulting symmetry information is encoded as an annotation of the surface M with regions $S_i \subseteq M, i = 1 \dots m$, in which the geometry $x(M)$ is symmetric with respect to a symmetry group $G_i \leq E(3)$. Each group G_i and region S_i is maximal with respect to set inclusion, in that order (first maximizing the group size, then the area covered). The prioritized maximization implies that the regions S_i can be overlapping or even hierarchically nested; see Figure 3. Our current implementation does not handle continuous symmetries, which we leave for future work.

4. Symmetry-Preserving Deformation

We describe deformations of the surface by variations of x. For this we use a displacement map $u : M \mapsto \mathbb{R}^3$. Then, the sum $x+u$ describes the deformed surface. Self-intersections are permitted, as it is common for surface deformation methods. The resulting set of displacements forms a vector space. For triangle meshes, deformations are described by the displacements of the vertices. Then, the space of displacements equals $\mathbb{R}^{3n}$, where n is the number of vertices.

If we have a group g that describes symmetries of the surface, then a displacement u preserves the symmetry if

$$g \circ (x+u) = (x+u) \circ \varphi, \tag{2}$$

where φ is the automorphism induced by g. Figure 4 illustrates how this condition induces a symmetry-preserving deformation field; see Kurz et al. [KWW*14] for more details.

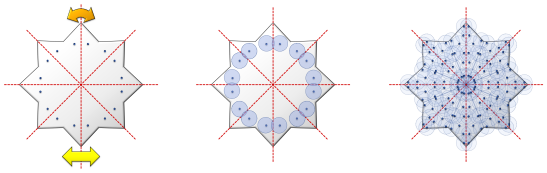
The basis of our surface modeling scheme is the observation that the set of all symmetry-preserving displacements forms a subspace of the vector space of all displacements (and thus the deformations themselves form an affine space).

Lemma 1. *Given a symmetry group G of a surface. The set of symmetry-preserving displacements forms a subspace of the vector space of all displacements.*

Proof. We have to show that for two arbitrary symmetry-preserving displacements u, v and any scalar $\lambda \in \mathbb{R}$, the displacement $\lambda u + v$ preserves the symmetry as well. We proceed in two steps: first we show that $u+v$ is symmetry preserving. Consider an arbitrary $g \in G$, and let φ denote the corresponding automorphism. Then, we will show that $g \circ (x+u+v) = (x+u+v) \circ \varphi$ holds. Since g is a Euclidean motion, there is an orthogonal matrix O and a translation t such that $g(p) = O(p) + t$ for all $p \in \mathbb{R}^3$.

$$\begin{aligned}
& g \circ (x+u+v) = (x+u+v) \circ \varphi \\
\Leftrightarrow \quad & g \circ (x+u+v) + g \circ x = (x+u+v) \circ \varphi + x \circ \varphi \\
\Leftrightarrow \quad & O(x+u+v) + t + O(x) + t = (x+u) \circ \varphi + v \circ \varphi + x \circ \varphi \\
\Leftrightarrow \quad & O(x+u) + t + O(x+v) + t = (x+u) \circ \varphi + (x+v) \circ \varphi \\
\Leftrightarrow \quad & g \circ (x+u) + g \circ (x+v) = (x+u) \circ \varphi + (x+v) \circ \varphi
\end{aligned}$$

In the first step, we used the definition of the symmetry (1).



(a) random sample (b) excluded discs (c) final sampling

Figure 5: *Symmetric sampling. (a) We pick a random point and apply all group transformations to it. (b) An exclusion disc is centered at each point for futher sampling. (c) New points are sampled from non-excluded area until the object is fully covered.*

The last equation holds by our assumption that u and v are symmetry-preserving. A similar argument shows that λu is symmetry-preserving. □

5. Direct Subspace Construction

In this section, we introduce an explicit construction of subspaces of the space of symmetry-preserving displacements. For this, we first generate a sampling of the surface that respects the symmetries (including partial symmetries) of the surface. This means every symmetry of the surface is a symmetry of the sampling as well. In addition, the sampling is an r-sampling of the surface, which means that for every point of the surface there is a point in the sampling at distance less than r. As a second step, we construct the space of symmetry-preserving displacements of the point sampling. Finally the displacements of the sampling are propagated to symmetry-preserving displacements of the surface. This construction offers two benefits. Most importantly, we obtain a low-dimensional subspace of the space symmetry-preserving deformations. Even if the underlying mesh is highly resolved, we can control the size of the subspace using the parameter r. This is important for obtaining an editing system that runs in real-time. A second benefit is that even if the input is a set of approximate symmetries, we can create a symmetric sampling. In this case, the sample points are not on the surface, but only close to it. To compute the deformations, we only need to preserve the exact symmetries of the sampling.

Symmetric sampling We propose a symmetry-aware Poisson disc sampling scheme that scatters points sparsely in the domain such that they conform to the underlying symmetry group structures. As input, we are given the surface M, annotated with potentially multiple, and potentially overlapping regions $S_i \subseteq M$ with symmetry groups $G_i \leq E(3)$, as discussed above. We now randomly start with a seed point, chosen with uniform probability from M. We then search for all annotations S_i this point is contained in. Then, for each symmetry group G_i, whose domain S_i contains this point, we add all the corresponding points transformed by its group action into sample set. To handle overlapping and nested symmetries correctly, we form the transitive closure: If the transformed sample point lies within a previously unseen area S_j, we recursively apply all transformations from G_j to this point. This process is continued until no new, previously non-sampled points are discovered anymore (a simple threshold of $10^{-4}\times$ the scene size is used to recognize doublet samples). All of those sample points will later be coupled, moving coherently and not providing more than three degrees of freedom altogether. For each of those newly added samples, we mark all points with a small radius ϕ_{sam} as invalid, which means they can not be picked later. We redo this process on the remaining point set, and iterate until all the points are either picked into the sample set, or marked as invalid.

Symmetry-preserving displacements of the sampling Let us first assume that the shape has only one constant symmetry group: During the sampling, we collect the transformations that map every seed point to its whole orbit. These can be used to construct the space of symmetry-preserving displacements of the sampling. For this, we use the following fact: whenever a point p is transformed by a Euclidean motion $g(p) = O(p) + t$, a displacement u of the point is transformed only by the orthogonal matrix O. Hence, we obtain a symmetry-preserving displacement of the sampling by displacing one vertex and propagating the displacement to the orbit of the point using only the orthogonal parts O of the Euclidean motions g (Figure 7(a)). The orbit of any sample point p has exactly three degrees of freedom that we obtain by applying this procedure to the unit displacements of p into each of the three coordinate directions. To generate a basis of the space of symmetry-preserving displacements, we construct the three basis vectors for every seed point we placed during sampling.

Partial, overlapping, and nested symmetry The same construction also works for nested and overlapping symmetry groups, where the transitive closure of the orbits is considered (Figure 8). The sampling algorithm generates these points by following and concatenating the local transformations during sampling. Hence, the local frames O are given by concatenations of the orthogonal mappings involved.

Degenerate samples A special case occurs if a sampling point is visited more than once but with different local frames O_i. This can happen on transformation-invariant sets, such as the diagonals in Figure 7(b): Here, we have orbits with four points from eight transformations, and each point has two different frames, differing by a reflection. The correct solution is obtained by reducing the dimension of the basis to those vectors v for which $O_i v = O_j v$ for all i, j, which yields is a linear system of equations. Due to the random sampling, this is rarely encountered in practice. In relevant cases, we can perform an SVD reduction of the null space to remove spurious degrees of freedom. If points do not perfectly overlap but only come close (which is still common

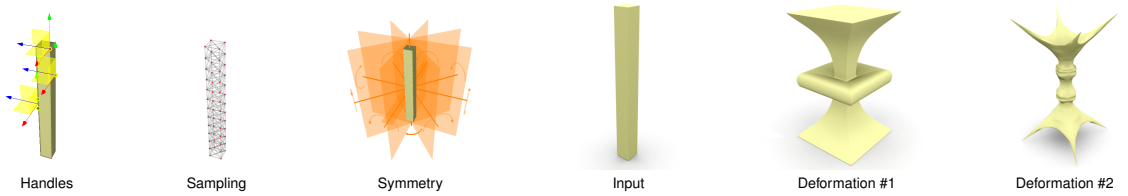
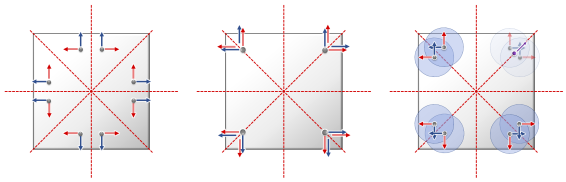


Figure 6: *Symmetry-preserving deformations of a bar are shown. The deformation are generated using three handles.*



(a) local frames (b) double frames (c) near-doublets

Figure 7: *Local frames. (a) Each sample point is associated with a local frame O. (b) If a point lies within a transformation-invariant set, it can have more than one frame $O_1, O_2, \ldots$. (c) The problem can be ignored for points in general position as the contributions of the radially-symmetric basis functions cancel out and the low-pass kernel maintains the band-limitation.*

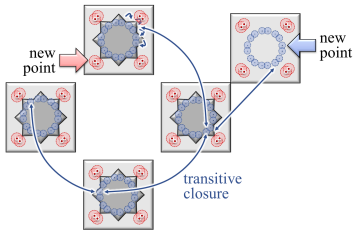


Figure 8: *Nested and overlapping symmetries are treated by propagating samples along transformations.*

close to transformation-invariant sets, see Figure 5(c)), we do not need to take special measures — the contributions of the basis functions cancel out exactly; we only obtain some overhead due to too dense sampling. The overhead is small as it only occurs at transformation-invariant sets of measure zero (reflection planes, rotation centers, Figure 7(c)).

Lifting the displacements To propagate a displacement of the sampling to a displacement of the surface, we compute for each basis vector $\bar{b}_i$ of the space of symmetry preserving displacements of the sampling a corresponding displacement vector b_i of the mesh. Then, the displacement $\bar{u} = \sum_i q_i \bar{b}_i$ of the sampling is lifted to the displacement $u = \sum_i q_i b_i$ of the mesh. The basis vectors $\bar{b}_i$ and b_i are a vector fields specifying a three dimensional vector for each sample point and mesh vertex. We denote these three dimensional vectors by $\bar{b}_i(\bar{v}_l)$ and $b_i(v_k)$. A displacement of a sampling point should only affects the displacement of the mesh vertices in a local neighborhood. We use Gaussian functions with standard deviation equal to the sampling density around every sample point to a assign influence weights to the mesh vertices. The Gaussian functions are cut-off (set to zero) for function values below 0.001. For each pair of a point $\bar{v}_l$ of the sampling and a vertex v_k of the mesh, we obtain a weight w_{kl}. Due to the compact support property this weight matrix is sparse. The basis vectors b_i are given by a partition-of-unity:

$$b_i(v_k) = \frac{1}{\sum_l w_{kl}} \sum_l w_{kl} \bar{b}_i(\bar{v}_l).$$

The basis b_i can be precomputed such that the Gaussians need not be evaluated in the interactive editing phase.

6. Symmetry-Preserving Editing

Once the subspace of symmetry-preserving displacements has been constructed, any deformation-based editing scheme could be used to produce symmetry-preserving deformations. Only the set of feasible displacements needs to be restricted to the subspace. However, as the meshes can be highly resolved, the computation of a deformation can be expensive. To be able to compute deformations of the surface in real-time, we restrict to low-frequency deformations that are liftings of displacements of the sampling. To compute the displacements of the sampling, we use an iterative corotated Laplace editing inspired by the approach proposed in [ATLF06]. The reasons for choosing this are approach are that on the one hand, we obtain a non-linear editing scheme that allows for large deformations, and, on the other hand, we only need minimal additional structure to compute the deformations. Namely, we need a Laplace matrix for the sampling and a list of neighbors for each vertex. There are different ways to get a Laplace matrix for the sampling. One is to compute the Laplace matrix of the original mesh and to restrict this matrix to the subspace generated by the sampling, see [HSL*06]. A second way is to compute a point-cloud Laplacian, see [LPG12], of the sampling. In our implementation, we specify a graph structure on the sampling: any pair of distinct points closer than $2r$ is connected by an edge. Then, we use the discrete Laplace (or Laplace–Kirchhoff) matrix of the graph, which is given as the difference of the degree matrix and the adjacency matrix. For Laplace editing, the Laplace matrix is applied not just to one function, but to the three component functions of the displacement vector

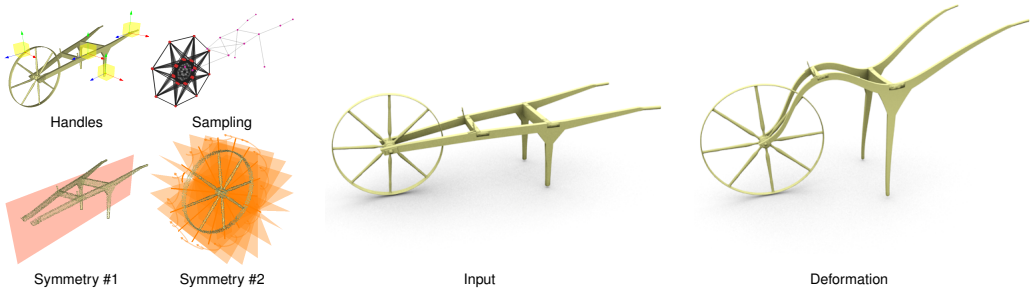


Figure 9: *A larger deformation of a yard tool generated with our symmetry-preserving modeling system is shown.*

field of sample points. Therefore, the $n \times n$ Laplace matrix is extended to a $3n \times 3n$ matrix by replacing every entry with the 3×3 matrix that is the entry times the 3×3 identity matrix. We denote this extended Laplace matrix by L.

Laplace editing The basis of the non-linear iterative co-rotated Laplace editing is the linear Laplace editing. For a thorough discussion of linear surface editing methods, we refer to [BS08]. Here we briefly review the variant of Laplace editing our scheme is based on. The deformation is computed by solving a quadratic minimization problem. The objective functional combines two quadratic functionals: one measures the deviation of the so-called Laplace coordinate and the other measures the deviation (in a least-squares sense) from user-specified constraints. We denote by $\bar{x}$ the vector listing the coordinates of the sample points and by $\bar{u}$ the displacement of the sampling points. We use the bar to distinguish the coordinates and displacements of the sampling from those of the surface mesh. The vector of Laplace coordinates is $\delta = L\bar{x}$ and the first quadratic functional is

$$E_L(\bar{u}) = \|L(\bar{x}+\bar{u}) - \delta\|^2 .$$

In our implementation the user can select handle regions in the sampling and assign desired positions to the selected sample points by rotating and translating the handles in space (see accompanying video). The deformed sampling will approximate these constraints. The corresponding least-squares functional is

$$E_C(\bar{u}) = \|A(\bar{u}) - a\|^2 ,$$

where a lists the desired displacements of all vertices in the handle regions. The matrix A is rectangular and has only one non-zero entry per row, which takes the value 1. The resulting deformation is given by the displacement that minimizes

$$E(\bar{u}) = E_L(\bar{u}) + \alpha E_C(\bar{u}) \qquad (3)$$

among all symmetry-preserving displacements. The parameter $\alpha \in \mathbb{R}_+$ controls how strongly the surface is pulled towards the user-specified handle positions.

To solve the quadratic program, we use the *null-space method*; see [NW06, Chapter 16.2]. Let U be the rectangular matrix whose columns are the basis vectors of the space of symmetry-preserving displacements of the sampling, and let q be the vector of coordinates with respect to the basis. To compute the minimizer of (3) in the space spanned by U, we have to solve the linear system

$$U^T(L^TL + \alpha A^TA)Uq = U^T(\alpha A^Ta + L^T(\delta - L\bar{x})) \qquad (4)$$

for q. Then, Uq is the solution in the space of symmetry-preserving displacements of the sampling. Since the symmetric, positive definite matrix $U^T(L^TL + \alpha A^TA)U$ only changes when new handle regions are selected or the weight α is modified, it is efficient to compute a Cholesky factorization of this matrix and to re-use it for solving the minimization problems. In addition, using a factorization speeds up the iterative co-rotated Laplace editing.

Iterative co-rotated Laplace editing A limitation of Laplace editing is that deformations that include larger rotational components lead to visually observable artifacts in the deformed surface, see [BS08]. Therefore, we use a non-linear variant obtained by iteratively applying Laplace editing. The Laplace coordinate can be interpreted as a vector field on the sampling specifying a 3-dimensional vector δ_i for every vertex. The Laplace coordinate is related to the discrete mean curvature normal and every δ_i should point into the direction of the surface normal at the corresponding sample point. The co-rotated Laplace editing process iterates a two-step procedure. First, the linear system (4) is solved. Then, the Laplace coordinates are rotated to point in normal direction of the deformed sampling. For rotating the Laplace coordinate of each vertex, we consider the undeformed and deformed local neighbors of the sample point and compute the rotation such that the rotated undeformed neighborhood best matches the deformed neighborhood. The neighborhoods are defined by the graph structure on the sample points. The rotation matrix can be computed using SVD, as described in (the additional material of) [SA07]. We stop the process when the maximum number of iterations (usually 5-10) has been reached. Once a displacement of the sampling is computed, we use the lifting process described in Section 4 to propagate the displacement of the sampling to a deformation of the surface mesh.

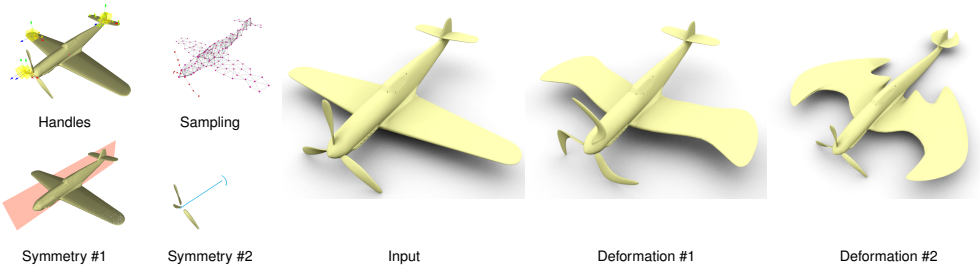


Figure 10: *Deformation generated with our symmetry-preserving modeling system are shown.*

Model		Mesh Info.		Sampling Info.			Timing (ms)				Frame rate
Name	Figure	Vertex (k)	Triangles (k)	Groups	Samples	DoF	Prefactor	Solving (5 Iter.)	Lifting	Rendering	Frames per s
Bar	6	4	8	1	96	18	13	2	3	12	58.8
Center piece	1	83	165	9	690	150	1300	21	32	12	15.4
Plane	10	181	359	2	160	234	75	4	51	18	13.7
Yard tool	9	50	97	2	78	27	55	3	22	5	33.3
Car	11	47	95	61	675	156	4160	35	3	10	20.8
Wind mill	2	20	40	2	156	33	97	5	7	6	55.6
Fan	12	2	4	3	72	27	43	3	1	13	58.8
Chair	12	15	58	2	32	39	4	2	1	16	52.6

Table 1: *Statistics and timings of the shown examples. From left to right: number of vertices, number of triangles, number of symmetry groups, number of sampling points, dimension of the space of symmetry-preserving deformations, timings for prefactoring the Laplace matrix, five iterations of co-rotated Laplace editing, lifting the solution from the samples to the surface mesh, and rendering the mesh, and the total number of frames per second.*

7. Experiments and Discussion

We tested our implementation of the proposed method on a set of shapes with varying complexity. Some shape have only a few symmetries, other have more complex nested symmetries. The mesh sizes vary from 4k to almost 200k vertices. Details are listed in Table 1. We tested the method with automatically detected and with manually specified symmetries. For example, the 61 symmetries of the car model (Figure 11) have been detected using the method proposed in [THW*14] and the 9 symmetries of the center piece (Figure 1) have be specified manually. For every example, the symmetries are illustrated in the corresponding figure. We tested with various sampling densities resulting in 72 to 690 sample points. We used random sampling and feature-sensitive sampling, where sharp features are selected first. We found the random sampling to produce the same quality of results as the feature-sensitive approach. Still, we show one example (the bar, Figure 6) produced with feature-sensitve sampling (here, we use the SVD-reduction described in Section 5 to remove doublet frames; this is not required/performed elsewhere). The dimensions of the spaces of symmetry-preserving displacements vary from 18 to 234. For many examples, we show large deformations, which the iterative co-rotated Laplace editing nicely supports.

Timings Our implementation of the symmetry-preserving modeling runs at 13-60 fps in our experiments. Table 1 lists details for the shown examples, including timings for prefactoring the Laplace matrix, 5 iterations of the co-rotated Laplace editing, lifting the deformation from the sampling to the surface mesh, and rendering. The timings were generated using a single-threaded C++ implementation running on an Intel® Xeon® E5-1620 processor at 3.60GHz with 16GB of RAM. The timings could be improved using parallelization, in particular, the time required for lifting the displacement from the sampling to the surface mesh. The shown run times demonstrate that, except for lifting the solution from the sampling to the surface and rendering the surface, the time needed to compute a deformation is independent of the resolution of the surface. It mainly depends on the dimension of the subspace of deformations, which in turn depends on the symmetry structure of the model and the resolution level of the desired editing operations. For example, though the car and the yard tool examples have a comparable number of vertices, there is a large difference in the time needed to compute the deformations, which is due to different sampling densities.

Comparison We compare our method to the recent scheme of Kurz et al. [KWW*14]. They propose to use symmetry-preserving deformation for template fitting to incomplete scan data. A comparison of results is shown in 12. A difference between the schemes is that their method enforces the symmetry in a least-squares sense, whereas our approach exactly preserves the symmetry. The resulting difference in visual quality can be seen in the figure. For example, our method better preserves the symmetry of the blades of the fan and better fits the desired positions of the legs of the armchair. To obtain the deformation that matches the given incomplete data, they combine handle-guided deformation and ICP. Let us compare the timings for the fan example. The timings listed in [KWW*14] are 2 seconds and 2 minutes per

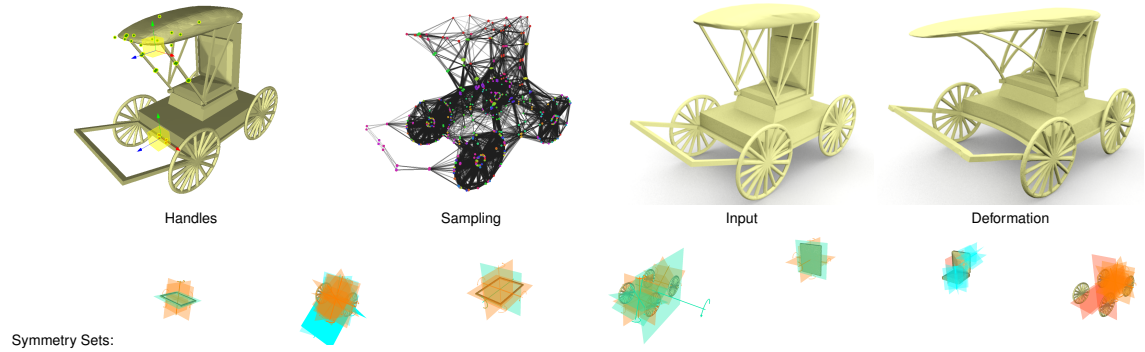


Figure 11: *A deformation that preserves 61 symmetries that were automatically detected is shown. Despite of the complex symmetry structures, the modeling system runs at real-time. Some of the detected symmetries are shown in the bottom row.*

iteration of the optimization, depending on whether linear or smooth basis functions (with wider support) are used. Their main bottleneck is the setup of the transfinite least-squares-constraints that involve numerical integration and fine discretization (disabling them reduces the computation times in their approach from 2 seconds to 0.4 seconds, still including ICP). Our results were produced using only handles. After 43ms for pre-factoring the Laplace matrix, the time per iteration of our optimization is less than 1ms. While the numerical implementation of Kurz et al. arguably leaves room for improvement, a performance gain of at least three orders of magnitude indicates clear advantages of our approach.

In the following, we discuss relations and differences of the proposed approach to the other previous work. Similar to our approach, the " iWires" framework [GSMCO09] aims at preserving geometric relations of a shape to be edited. Our symmetry-preservation model (based on [KWW*14]) captures most of their constraints (equal area, equal length, etc.). In addition to conceptual simplicity, this also provides a navigable shape space that can be utilized as prior in various applications. Some details are different: For example, our method does not explicitly parallel lines. However, as linear constraints, they could be added as special cases to our approach, too. Further, unlike our approach, we use dense relations on symmetric area rather than feature lines ("wires"), thereby no relying on the existence of such sharp features. As solver, iWires uses a greedy propagation algorithm that, unlike our method, does not guarantee to satisfy all constraints. Nonetheless, our method is still faster (Gal et al. report response times of 2-4 seconds).

Figure 12: *We compare results of our method with results produced by the scheme proposed by Kurz et al. [KWW*14].*

Wang et al. [WXL*11] use hierarchical propagation of attributes in objects with complex symmetry patterns for structure-preserving shape editing. The work differs from ours in that it builds a hierarchical representation, breaking cycles in symmetry-constraints by perceptual reasoning. In contrast, we utilize all symmetry information, including arbitrary cycles. By understanding the feasible set as a linear shape space, efficient navigation can still be guaranteed (whereas the method of Wang et al. utilizes direct edit propagation in the hierarchy).

8. Conclusion

We present a method for real-time symmetry-preserving shape modeling. The basis of the scheme is a construction of spaces consisting of low-frequency deformations that preserve the symmetry. Within these low-dimensional spaces, we apply a non-linear deformation-based editing scheme. We demonstrate real-time deformations that preserve the symmetries exactly and support large deformations. The method is much easier to implement than previous optimization-based methods and significantly faster.

Limitations and challenges Currently our scheme supports only finite symmetry groups. A direction of future work would be to integrate continuous symmetries, e.g., arbitrary

rotations around a fixed axis. Another limitation is that editing operations which require frequencies below the sampling density are not supported. Hence, very localized edits require a dense sampling. It would be interesting to integrate non-uniform samplings. This would allow for localized edits in selected regions while preserving the real-time performance. A challenging problem is to extend the framework to other types of symmetries like intrinsic or Möbius symmetries.

Acknowledgements

This work was supported by the international Max Planck Research School for Computer Science (IMPRS-CS), Microsoft Research Cambridge, and the Max Planck Center for Visual Computing and Communication (MPC-VCC).

References

[ABCO*03] ALEXA M., BEHT J., COHEN-OR D., FLEISHMAN S., LEVIN D., SILVA C. T.: Computing and rendering point set surfaces. *IEEE TVCG 9*, 1 (2003), 3–15. 2

[AS07] AVIDAN S., SHAMIR A.: Seam carving for content-aware image resizing. *ACM Trans. Graph. 26*, 3 (2007). 2

[ATLF06] AU O. K.-C., TAI C.-L., LIU L., FU H.: Dual Laplacian editing for meshes. *IEEE Transactions on Visualization and Computer Graphics 12*, 3 (2006), 386–395. 2, 6

[BR07] BROWN B., RUSINKIEWICZ S.: Global non-rigid alignment of 3-d scans. *ACM Trans. Graph. 26*, 3 (2007). 2

[BS08] BOTSCH M., SORKINE O.: On linear variational surface deformation methods. *IEEE Transactions on Visualization and Computer Graphics 14*, 1 (2008), 213–230. 2, 7

[BWKS11] BOKELOH M., WAND M., KOLTUN V., SEIDEL H.-P.: Pattern-aware shape deformation using sliding dockers. *ACM Transactions on Graphics 30*, 6 (2011). 1, 2

[BWSK12] BOKELOH M., WAND M., SEIDEL H.-P., KOLTUN V.: An algebraic model for parameterized shape editing. *ACM Transactions on Graphics 31*, 4 (2012). 1, 2

[Coq90] COQUILLART S.: Extended free-form deformation: a sculpturing tool for 3d geometric modeling. In *Proc. Siggraph* (1990), pp. 187–196. 2

[GSMCO09] GAL R., SORKINE O., MITRA N., COHEN-OR D.: iWires: An analyze-and-edit approach to shape manipulation. *ACM Trans. Graph. 28*, 3 (2009). 1, 2, 9

[HSL*06] HUANG J., SHI X., LIU X., ZHOU K., WEI L.-Y., TENG S.-H., BAO H., GUO B., SHUM H.-Y.: Subspace gradient domain mesh deformation. *ACM Trans. Graph. 25*, 3 (2006), 1126–1134. 2, 6

[HSvTP11] HILDEBRANDT K., SCHULZ C., VON TYCOWICZ C., POLTHIER K.: Interactive surface modeling using modal analysis. *ACM Trans. Graph. 30*, 5 (2011), 119:1–11. 2

[JBK*12] JACOBSON A., BARAN I., KAVAN L., POPOVIĆ J., SORKINE O.: Fast automatic skinning transformations. *ACM Trans. Graph. 31*, 4 (2012), 77:1–10. 2

[KFR04] KAZHDAN M., FUNKHOUSER T., RUSINKIEWICZ S.: Symmetry descriptors and 3d shape matching. In *Proc. Symposium on Geometry Processing (SGP)* (2004). 1, 3

[KSSCO08] KRAEVOY V., SHEFFER A., SHAMIR A., COHEN-OR D.: Non-homogeneous resizing of complex models. *ACM Trans. Graph. 27*, 5 (Dec. 2008), 111:1–111:9. 1, 2

[KWW*14] KURZ C., WU X., WAND M., THORMÄHLEN T., KOHLI P., SEIDEL H.-P.: Symmetry-aware template deformation and fitting. *Computer Graphics Forum* (2014). to appear (early view available). 1, 2, 4, 8, 9

[LCDF10] LIPMAN Y., CHEN X., DAUBECHIES I., FUNKHOUSER T.: Symmetry factored embedding and distance. *ACM Trans. Graph. 29* (2010), 103:1–12. 1, 3

[LCOZ*11] LIN J., COHEN-OR D., ZHANG H., LIANG C., SHARF A., DEUSSEN O., CHEN B.: Structure-preserving retargeting of irregular 3D architecture. *ACM Trans. Graph. 30*, 6 (2011). 2

[LPG12] LIU Y., PRABHAKARAN B., GUO X.: Point-based manifold harmonics. *IEEE Transactions on Visualization and Computer Graphics 18*, 10 (Oct. 2012), 1693–1703. 6

[MPWC12] MITRA N. J., PAULY M., WAND M., CEYLAN D.: Symmetry in 3d geometry: Extraction and applications. EUROGRAPHICS State-of-the-art Report, 2012. 3, 4

[MWZ*13] MITRA N., WAND M., ZHANG H., COHEN-OR D., BOKELOH M.: Structure-aware shape processing. Eurographics 2013 State-of-the-Art Report (STAR), April 2013. 1

[NW06] NOCEDAL J., WRIGHT S. J.: *Numerical Optimization (2nd edition)*. Springer, 2006. 7

[OMPG13] OVSJANIKOV M., MÉRIGOT Q., PĂTRĂUCEAN V., GUIBAS L.: Shape matching via quotient spaces. *Computer Graphics Forum (Proceedings of SGP)* (2013). 1, 3

[SA07] SORKINE O., ALEXA M.: As-rigid-as-possible surface modeling. In *Proceedings of Eurographics/ACM SIGGRAPH Symposium on Geometry Processing* (2007), pp. 109–116. 2, 7

[SCOL*04] SORKINE O., COHEN-OR D., LIPMAN Y., ALEXA M., RÖSSL C., SEIDEL H.-P.: Laplacian surface editing. In *Symposium on Geometry processing* (2004), pp. 175–184. 2

[SP86] SEDERBERG T. W., PARRY S. R.: Free-form deformation of solid geometric models. In *SIGGRAPH* (1986). 2

[THW*14] TEVS A., HUANG Q., WAND M., SEIDEL H.-P., GUIBAS L.: Relating shapes via geometric symmetries and regularities. *ACM Trans. Graph. 33*, 4 (2014), 119:1–12. 4, 8

[TPBF87] TERZOPOULOS D., PLATT J., BARR A., FLEISCHER K.: Elastically deformable models. In *Proceedings of SIGGRAPH* (New York, NY, USA, 1987), ACM, pp. 205–214. 2

[vTSSH13] VON TYCOWICZ C., SCHULZ C., SEIDEL H.-P., HILDEBRANDT K.: An efficient construction of reduced deformable objects. *ACM Trans. Graph. 32*, 6 (2013), 213:1–10. 2

[WSSZ14] WANG H., SIMARI P., SU Z., ZHANG H.: Spectral global intrinsic symmetry invariant functions. In *Graphics Interface* (2014). 1, 3

[WW92] WELCH W., WITKIN A.: Variational surface modeling. In *Computer Graphics (Proceedings Siggraph)* (1992), vol. 26. 2

[WXL*11] WANG Y., XU K., LI J., ZHANG H., SHAMIR A., LIU L., CHENG Z., XIONG Y.: Symmetry hierarchy of man-made objects. *Computer Graphics Forum 30*, 2 (2011). 2, 9

[XZT*09] XU K., ZHANG H., TAGLIASACCHI A., LIU L., LI G., MENG M., XIONG Y.: Partial intrinsic reflectional symmetry of 3d shapes. *ACM Transactions on Graphics, (Proceedings SIGGRAPH Asia 2009) 28*, 5 (2009), 138:1–138:10. 1, 2

[ZFCO*11] ZHENG Y., FU H., COHEN-OR D., AU O. K.-C., TAI C.-L.: Component-wise controllers for structure-preserving shape manipulation. *Computer Graphics Forum 30*, 2 (2011), 563–572. 1, 2