DOI: 10.1111/cgf.12344

COMPUTER GRAPHICS *forum*
Volume 33 (2014), number 6 pp. 205–219

Symmetry-Aware Template Deformation and Fitting

C. Kurz[1], X. Wu[1], M. Wand[1,2], T. Thormählen[3], P. Kohli[4] and H.-P. Seidel[1]

[1]MPI Informatik Saarbrücken, Germany
{ckurz,xwu,mwand,hpseidel}@mpi-inf.mpg.de, M.Wand@uu.nl
[2]Universiteit Utrecht, Netherlands
[3]Philipps-Universität Marburg, Germany
thormae@informatik.uni-marburg.de
[4]Microsoft Research Cambridge, UK
pkohli@microsoft.com

Abstract

In this paper, we propose a new method for reconstructing 3D models from a noisy and incomplete 3D scan and a coarse template model. The main idea is to maintain characteristic high-level features of the template that remain unchanged for different variants of the same type of object. As invariants, we chose the partial symmetry structure of the template model under Euclidian transformations, i.e. we maintain the algebraic structure of all reflections, rotations and translations that map the object partially to itself. We propose an optimization scheme that maintains continuous and discrete symmetry properties of this kind while registering a template against scan data using a deformable iterative closest points (ICP) framework with thin-plate-spline regularization. We apply our new deformation approach to a large number of example data sets and demonstrate that symmetry-guided template matching often yields much more plausible reconstructions than previous variants of ICP.

Keywords: mesh deformation, template fitting, deformable ICP, symmetry

ACM CCS: I.3.5 [Computer Graphics]: Computational Geometry and Object Modelling; I.2.10 [Artificial Intelligence]: Vision and Scene Understanding

1. Introduction

Content creation has become one of the main challenges of contemporary computer graphics: While numerous techniques are known for representing and rendering complex scenes, the creation of the 3D models itself is still a tedious task that requires artistic skills and a high level of technical expertise. Broadly, we can distinguish two approaches:

One option is to create virtual objects from scratch using 3D modelling software. In this domain, a lot of recent research has focused on making editing more efficient [MWZ*13]. The main idea is to define a *structure* model that characterizes properties common to a larger class of similar shapes and detect and maintain such structural properties in shapes during interactive editing.

A second alternative is to perform 3D scanning to create virtual replicas of real-world objects. However, this requires that a physical object that closely matches our requirements is available. Even then, the scanning approach itself is troubled by data quality issues: Any optical acquisition method suffers from occlusion problems so that the scanned object is usually only captured partially. Furthermore, noise, structured outliers and inaccurate registration trouble the process, in particular for inexpensive consumer equipment, such as the Microsoft Kinect™ [IKH*11].

Our paper addresses some of the problems that arise when utilizing 3D scanning for content creation. Our approach is based on the observation that large collections of 3D shapes are available in libraries such as Trimble/Google 3D Warehouse™. If the library directly provides what we need, we are of course all set. However, this is not likely because shape spaces of non-trivial classes of shapes (such as furniture, household items, etc.) are high-dimensional and usually have more degrees of freedom than a shape library can reasonably sample. Thus, it is quite likely that the shape that we actually want to acquire is still quite different in *geometry* from any readily available model. It is much more likely that we can find a template model that does have a different geometry but that is still very similar in *structure*: For many classes of objects (in particular, man-made shapes) there are high-level structural invariants that are shared among large sets of geometry of related functionality [MWZ*13].

The goal of our paper is to utilize structure priors to better adapt an existing template model to scanned data with only a small amount of user intervention. When successful, this addresses the problems with 3D scanning discussed above: The template will fill in acquisition holes, suppress noise and outliers, and we can potentially transfer a handcrafted, well-designed 3D mesh to the unstructured point cloud data of the scanner [KS05].

Our structure model is based on symmetry: We leverage previous work to automatically detect all partial extrinsic Euclidian symmetries of the template shape, including continuous and discrete symmetries [MGP06, PMW*08, BWKS11]. We then deform the template using a smooth free-form deformation but maintain the detected *symmetry structure* as an invariant: Whenever two parts of the geometry had originally been related by a rigid transformation, this must still be the case in any deformed variant of the shape.

We propose a new optimization algorithm that formulates symmetry-aware shape deformation as a quadratic energy (co-rotated according to latent transformation variables) that combines a standard thin-plate spline (TPS) regularizer with symmetry preservation.

Conceptually, our method is based on previous work in structure-aware shape deformation [KSSCO08, HMC09, GSMCO09, ZFCO*11, WXL*11, BWKS11, BWSK12]. We make two important conceptually novel contributions: First, our technique is based only on the very basic assumption of preserving the algebraic symmetry structure of the partial Euclidian symmetries of a 3D shape. Formally, this is captured in a novel formulation as commutativity of deformation and pairwise symmetry transformations. Secondly, to the best of our knowledge, we present the first technique that uses symmetry-aware deformation for template fitting to noisy scanner data, which facilitates the creation of high-quality 3D meshes from even low-quality 3D scans.

We evaluate our method by studying a number of practical examples of shape acquisition tasks. We acquire 3D shapes at varying quality levels using KinectFusion [IKH*11] and afterwards use template models of different levels of complexity to improve the raw scanner data. We compare against previous baseline methods such as deformable iterative closest points (ICP) and previous structure-aware deformation models. We obtain more plausible reconstructions, in particular in partial scans with a lot of missing data and scans with high noise level. We believe that our method provides a valuable tool for incorporating structural knowledge from templates into 3D scans.

2. Related Work

A number of structure models have been proposed in literature, as well as algorithms for maintaining them under shape alterations. In the following, we review such previous approaches, with an emphasis on deformation models, template fitting and database-driven methods.

Deformation models. Shape deformation has a long tradition in computer graphics. Suitable deformations can be computed by explicitly constructed basis functions [JSW05, JMD*07, LLCO08, BCWG09] with suitable smoothness properties or alternatively by variational methods, such as elasticity models [TPBF87, BS08]. We use a variational TPS framework [ACP03, BR07] as basis for our method that aims at general smooth deformations, which we subsequently augment with a new model for preserving algebraic symmetry structure.

Structure-aware deformation. A seminal approach for structure-aware deformation was seam carving [AS07], retargeting images for different aspect ratios. Kraevoy *et al.* [KSSCO08] propose a similar idea for geometric objects: Their method performs slippage and curvature analysis to determine how vulnerable a piece of surface is with respect to stretch in the direction of the three coordinate axes. The method, however, is strictly limited to axis-aligned resizing.

The *iWires* system [GSMCO09] proposes a more general set of structural invariants. The model preserves non-local properties, including symmetry, parallelity and similarity to basic geometric shapes such as circles. Huang *et al.* [HMC09] propose similar ideas for 2D images. Our approach is strongly inspired by this previous work. However, unlike iWires, our method is solely based on symmetry assumptions.

This yields a very simple, variational framework that, unlike the original iWires, can be adapted for our application of template registration. Furthermore, while our method covers symmetry more completely, other geometric relations are not captured, such as parallelity or right angles in non-symmetric shapes (in symmetric shapes, parallel lines and specific angles, such as 60°, 90°, arise automatically from symmetry constraints).

More recently, Bokeloh *et al.* [BWKS11, BWSK12] use translational symmetry invariants to generalize the resizing method of Kraevoy *et al.* [KSSCO08]. The method supports topological changes (inserting and removing repeating elements) but is limited to translational resizing. Our approach is complementary to theirs: We support general Euclidian symmetry (including rotations and reflections) but we remain in the domain of homeomorphic deformations, i.e. we perform a continuous deformation that keeps the original object topology. In this setting, we can formulate our objective as a simple co-rotated least-squares problem and we do not require complex discrete optimization.

Recent work on smart deformation tools also includes the method of [CM09], which resizes objects using retargeting of geometry images, and [XWY*09] that try to guess joint properties to build a semi-articulated deformation model. [WXL*11] and [ZFCO*11] have proposed two symmetry-based mesh editing techniques that examine symmetry relations in order to derive modification rules.

In general, we have to emphasize that all of the deformation models cited above have been employed for user-guided shape deformation; the aspect of regularizing deformable shape matching is a novel contribution of this paper.

Editing by part-based assembly. Some approaches for structure-aware editing are not using deformation but explore combinatorial rules for reassembling 3D shapes [Mer07, BWS10, JTRS12, KCKK12]. Several of these part-based editing systems rely on databases of 3D shapes to retrieve a suitable part. [CKGK11] even take the semantic relationships of parts into account. User interaction [FKS*04], [PMG*05] is a viable approach in this area. Our work focuses on continuous deformation; part-based reconstruction is currently out of scope.

Structure-aware template fitting. A given template mesh can be deformed to be fitted to input data (such as laser scans, photographs or videos) to repair topology and geometry of ambiguous data [KS05]. This is achieved by combining the ICP approach with a suitable deformation model [HTB03, ACP03, PMG*05, KS05, BR07, WJH*07, ARV07]. General deformable ICP approaches often suffer from overparametrization and the local smoothness constraints typically fall short in preserving the characteristic global structures of the object. This issue is addressed by template fitting approaches that only consider a certain class of objects, such as the class of faces [BSVS04] or human bodies [ASK*05, HSS*09], and, consequently, can rely on a model-specific low-dimensional parametric shape space. In a more general approach, [XZZ*11] utilize the component-wise deformation approach by Zheng *et al.* [ZFCO*11] to fit a template to photographs. Recently, a part-based approach was introduced which reconstructs a 3D model from very noisy and incomplete 3D point clouds via part-based assembly [SFCH12]. Here, individual parts of the assembly are kept rigid, whereas our approach allows non-rigid deformation. A direct reconstruction method that conjectures symmetries in order to complete shapes has been proposed by Thrun and Wegbreit [TW05]. It does not use template geometry but examines how visibility and geometry are consistent with different symmetry hypotheses. This is fully automatic, however, the lack of a template provides less control and the lack of a deformation model means that the method cannot provide structure-aware editing of the results.

Scan processing. Kim *et al.* [KMYG12] have demonstrated an approach to acquire indoor environments from single-view scans using primitive-based 3D models from a separate learning stage. In [KMHG13], they introduce a shape descriptor for user guidance in interactive scanning. These approaches are orthogonal to our algorithm and could serve in a pre-processing stage, in order to extract suitable input from large-scale data and to select template models automatically.

Surface reconstruction. GlobFit by Li *et al.* [LWC*11] augments a local primitive detection approach for surface reconstruction based on random sample consensus by enforcing global relations between the primitives. Our method is not limited to models consisting of basic primitives.

3. Overview

Let $\mathcal{S} \subset \mathbb{R}^3$ denote the *template model* that is the input to our algorithm. For simplicity, we assume that $\mathcal{S}$ is given as a triangle mesh (of arbitrary topology), but it is rather straightforward to generalize our method to other input representations. Furthermore, let $\mathcal{D} = \{\mathbf{d}_1, ..., \mathbf{d}_n\} \subset \mathbb{R}^3$ denote the input data obtained from the 3D scanner, modelled directly as an unstructured cloud of 3D points.

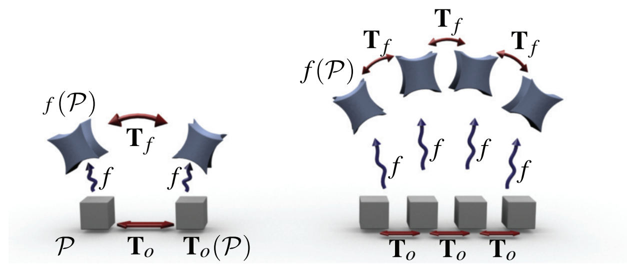

Figure 1: *Two types of symmetry constraints. Left: the basic constraint assures that two pieces of geometry are identical up to a transformation $\mathbf{T}_f \in \mathcal{G}$. Right: by sharing latent transformations $\mathbf{T}_f$ among multiple instances, regular patterns can be modelled.*

Given a number of external deformation constraints, manually defined or derived from a set of target data points $\mathcal{D}$, our approach allows to estimate an optimal deformation of the 3D surface $\mathcal{S}$ while keeping its high-level characteristic surface properties intact. The high-level structure is modelled as the discrete and continuous symmetries of the template.

Symmetry is defined with respect to a *group of admissible transformations* $\mathcal{G}$, consisting of homeomorphisms (bijective, in both ways continuous mappings) $\mathbf{T} : \mathbb{R}^3 \to \mathbb{R}^3$. In this paper, we consider the group of Euclidian transformations $\mathcal{G} = E(3)$, i.e. translations, rotations and reflections.

Given the template $\mathcal{S}$, we are trying to estimate an output surface $f(\mathcal{S})$ that has the same *algebraic symmetry structure*. This means that $f(\mathcal{S})$ might have a very different geometry but the symmetries $\mathbf{T}_o$ of $\mathcal{S}$ and their mutual relations should be preserved: If two submeshes $\mathcal{P}$, $\mathcal{Q} \subseteq \mathcal{S}$ are symmetric, then the same should hold for the respective output surfaces $f(\mathcal{P})$, $f(\mathcal{Q})$. The output surfaces might have very different geometry and a different transformation $\mathbf{T}_f$ relating the two submeshes, though. Likewise, the relations between symmetries should be preserved: For example, if a number of symmetric parts are aligned on a regular grid, this structure should still be present in the deformed model. These two types of symmetry constraints are illustrated in Figure 1. We use the term ‘algebraic symmetry structure’ to capture the notion that we only preserve the fact that geometry is related by a Euclidian transformation (including equality of the transformations involved) but do not fix the concrete mapping itself.

In the following, we present our framework: First, we describe our deformation model (Section 4). Then, we introduce our notion of symmetry and describe the symmetry detection pipeline (Section 5), followed by design choices and implementation strategies (Section 6), results (Section 7) and the chosen parameters (Section 8) We discuss the limitations of our approach (Section 9) before concluding the paper.

4. Deformation Model

The basis of our method is a free-form deformation model, which we extend subsequently to incorporate symmetry preservation. The deformation model itself (Sections 4.1–4.3) is not novel—we lay out the details here for completeness. In principle, our method could be combined with the majority of variational deformation models in literature. The novelty is the addition of symmetry constraints in Section 4.4.

4.1. Representation

In order to compute a deformation, we embed the surface $\mathcal{S}$ into a volume $\mathcal{V} \subset \mathbb{R}^3, \mathcal{S} \subset \mathcal{V}$ and deform this volume using a deformation field $f : \mathcal{V} \to \mathbb{R}^3$. This approach has the benefit of making the deformation independent of the representation of $\mathcal{S}$ so that arbitrary types of input geometry and general surface topology can be handled easily. Following [HSL*06, SSP07], we use a subspace method to discretize the deformation field f. We create a number of nodes $\mathbf{u}_1, ..., \mathbf{u}_k \subset \mathbb{R}^3$ and centre radial basis functions b around these to define the deformation field:

$$f(\mathbf{x}) = \sum_{i=1}^{K} \tilde{\mathbf{u}}_i \, b(||\mathbf{x} - \mathbf{u}_i||). \tag{1}$$

Here, $\tilde{\mathbf{u}}_i \in \mathbb{R}^3$ are the displaced positions of the nodes $\mathbf{u}_i$. As radial basis functions, we used uniform cubic tensor-product B-splines that provide second-order smoothness with minimal support. We have also experimented with radial basis functions created from Wendland functions. They yield visually identical results; adjusting the spacing for minimal overlap was more difficult, causing higher computational costs.

We place the nodes by discretizing $\mathcal{S}$ to a regular grid of user-specified spacing ϵ_{grid}. Then, we add additional grid points such that every surface point is overlapped by four B-spline functions in x-, y- and z-directions to obtain a valid B-spline basis. This guarantees that the basis functions and their derivatives are well defined on $\mathcal{S}$.

The deformation field f is estimated using a standard variational approach: We setup an energy function $E(f)$ that is minimized by an optimal f.

$$E = \lambda_c E_c + E_d + \lambda_r E_r + \lambda_s E_s. \tag{2}$$

The energy consists of several terms which model separate aspects: E_c (handle constraints) and E_d (ICP-like constraints) describe external deformation constraints, E_r is the TPS regularizer that encourages smoothness, E_s preserves similarity of symmetric parts as well as similarity of transformations in regular structures. Each term is weighted by a parameter (λ_c, λ_r, and λ_s) to control its influence relative to the ICP-like constraints. The unknowns of the minimization are the displaced node positions $\tilde{\mathbf{x}}_i$ that constitute f. In addition, we will also introduce additional latent variables (i.e. variables that are derived implicitly from the context) that model the transformations later.

4.2. External deformation constraints

Handle constraints. The first energy term E_c accounts for manual user constraints. We use the standard *handle* model [BKS03, BK04] which imposes a series of *position constraints* $\mathcal{C}_i = (\mathrm{p_i}, \mathrm{q_i})$ by specifying a one-to-one mapping between an initial point $\mathbf{p}$ on $\mathcal{S}$ and a target point $\mathcal{D}$ Point:

$$E_c(f, \mathcal{C}) = \sum_{\mathcal{C}_i \in \mathcal{C}} \| f(\mathbf{p}_i) - \mathcal{D}Point_i \|^2. \tag{3}$$

ICP-like constraints. The data term E_d of Equation (2) ensures that f is formed in a way that makes $\mathcal{S}$ match the target surface $\mathcal{D}$. This is achieved by formulating a series of ICP-like constraints [HTB03, WJH*07] between $\mathcal{S}$ and $\mathcal{D}$:

$$E_d(f, \mathcal{D}) = \sum_{\mathbf{d}_i \in \mathcal{D}} w_i \left\langle f(\mathbf{p}_j) - \mathbf{d}_i, \, \mathbf{n}_i \right\rangle^2, \tag{4}$$

with w being a *weighting factor* that penalizes outliers, $\mathbf{p}$ a sample point on $\mathcal{S}$ and $\mathbf{n}$ the normals corresponding to $\mathbf{d}$. The *closest point index* j is selected in a way that makes $\mathbf{p}_j$ the point in $\mathcal{S}$ closest to $\mathbf{d}_i$.

4.3. TPS deformation model

The regularizer term E_r of Equation (2) governs the structure of the deformation field where it is underconstrained. We use a standard formulation based on a TPS deformation model [ACP03, BR07] which discourages bending in $\mathcal{S}$:

$$E_r(f) = \int_{\mathcal{V}} \|\mathcal{H}_f(\mathbf{x})\|_F^2 \, d\mathbf{x}, \tag{5}$$

with $\mathcal{H}_f(\mathbf{x})$ the *Hessian matrix* of f at position $\mathbf{x}$, and $\| \cdot \|_F$ the Frobenius norm. This *TPS* energy encourages smoothness by penalizing second derivatives.

4.4. Preserving symmetries

We now augment the TPS deformation model such that it preserves the shape of symmetric parts. We formulate a constraint that two pieces $\mathcal{P}, \mathcal{Q} \subseteq \mathcal{S}$ should be symmetric according to a transformation from $\mathcal{G}$:

$$E_s\left(f, \mathbf{T}_f \mid \mathcal{P} \sim \mathbf{T}_o(\mathcal{P}), \mathbf{T}_o\right) = \int_{\mathcal{P}} \|\mathbf{T}_f f(\mathbf{x}) - f(\mathbf{T}_o \mathbf{x})\|^2 \, d\mathbf{x}. \tag{6}$$

This energy has four arguments (two constants and two unknowns): f is the unknown deformation. $\mathbf{T}_o$ the original (known) transformation that maps a fixed piece $\mathcal{P} \subseteq \mathcal{S}$ to $\mathbf{T}_o(\mathcal{P}) \subseteq \mathcal{S}$, in the original model. Accordingly, $\mathbf{T}_f$ is an unknown transformation that matches the deformed versions, i.e. mapping $f(\mathcal{P})$ to $f(\mathbf{T}_o(\mathcal{P}))$. This *latent* variable is only computed implicitly in order to optimize for best symmetry preservation and minimum deformation.

The setup is illustrated in Figure 1 (left) in a commuting diagram: In order to preserve the original symmetries $\mathbf{T}_o$, the deformation function f must commute with any symmetry transformation $\mathbf{T}$ in the regions of $\mathcal{S}$ that are symmetric. In other words, f must provide the prescribed symmetry already. If we required a strict preservation of the original symmetry transformations $\mathbf{T}_o$, we would strongly restrict the deformation to maintain all original symmetry properties (absolute rotation axes and reflection planes in space, relative displacements and rotations, etc.). Thus, we only aim at preserving the algebraic structure of the symmetries. This means, it becomes permissible to replace $\mathbf{T}_o$ with a new (yet to be determined) corresponding transformation $\mathbf{T}_f$ when moving it out of the argument of $f(\cdot)$ to the outside. In Figure 1 (left), this means that the paths $\mathbf{T}_f \circ f$ and $f \circ \mathbf{T}_o$ must lead to identical results. Equation (6) penalizes violations of this commutative behaviour in a least squares sense using an integral, transfinite constraint that covers the domain $\mathcal{P} \subseteq \mathcal{S}$ that is symmetric.

We will use this constraint in two flavours: First, for simple pairwise symmetries, we employ Equation (6) as is to enforce a similar shape. Secondly, for complex patterns where a group of transformations is generated by a small set of generator transformations, as shown in Figure 1 (right), we use shared transformation variables $\mathbf{T}_n$. For example, if n shapes $\mathcal{P}_i$, $i = 1..n$ have originally been aligned on a regular grid, we would constrain all $f(\mathcal{P}_i)$, $i = 1..n-1$ to be symmetric to $f(\mathcal{P}_{i+1})$ under the same transformation $\mathbf{T}_f$. In Section 5.1, we describe in more detail how symmetry groups are identified.

Solving the system. If all transformations are known, Equation (6) is a quadratic energy. In this case, we can just determine the gradient with respect to the $\tilde{\mathbf{u}}_i$ and add the resulting linear expression to the linear system obtained previously. In other words, we co-rotate the linear system with the latent transformations. The linear system is solved using simple, plain conjugate gradients.

In order to solve for the unknown transformation $\mathbf{T}_f$, we again use an iterative approach. We start with $\mathbf{T}_f = \mathbf{T}_o$ and solve the linear system. Afterwards, we perform shape matching between $f(\mathcal{P})$ and $f(\mathcal{Q})$ to estimate a new transformation. As we know the correspondences through f, this is very easy: We first fit a least-squares optimal affine map and then perform a polar decomposition of the linear part to project it back to $E(3)$. In case that multiple parts $\mathcal{P}_i$ correspond under the same, unknown transformation $\mathbf{T}_f$, we can apply the same principle. The only difference is to compute the least-square fits to the difference vectors between all pairs $\{i, i+1\}$, not only a single pair.

5. Symmetry Detection

We now have a numerical tool to express symmetry of geometry (by using symmetry constraint energy terms E_s) as well as the similarity of transformations (by sharing latent transformation variables among E_s terms). This requires analysis of the symmetry structure of the input and determination of a set of symmetry constraints.

5.1. Structuring the symmetries

Symmetry. We denote the set of symmetries extracted from $\mathcal{S}$ with respect to a *symmetry transformation* $\mathbf{T} \in \mathcal{G}$ as

$$\xi(\mathbf{T}) := \{\mathbf{x} \in \mathcal{S} \mid \mathbf{T}(\mathbf{x}) \in \mathcal{S}\}. \quad (7)$$

In other words, we intersect the object with a transformed version of itself in order to find symmetric parts: $\xi(\mathbf{T}) = \mathcal{S} \cap \mathbf{T}(\mathcal{S})$. In order to avoid spurious matches, we exclude results where the area of $\xi(\mathbf{T})$ is too small.

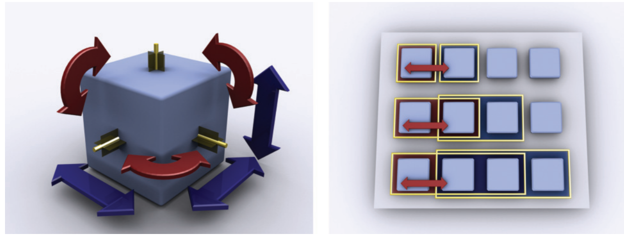

Figure 2: *Left: A cube is symmetric under rotations by 90° (red arrows) and mirroring (blue arrows) across the main axis. The 48 configurations form the full octahedral symmetry group [Mil72]. Right: Multiple transformations map symmetric subsets to each other in a grid of symmetric pieces.*

Symmetry groups. We formulate our analysis in terms of symmetry groups [MPWC13]. Let $\mathcal{P} \subseteq \mathcal{S}$ be a fixed piece of geometry and $\mathcal{T} \subseteq \mathcal{G}$ be a set of transformations that map $\mathcal{P}$ back to $\mathcal{S}$. We denote the geometry created by applying these transformations as

$$\mathcal{P}_{\mathcal{T}} := \bigcup_{\mathbf{T} \in \mathcal{T}} \mathbf{T}(\mathcal{P}). \quad (8)$$

If $\mathcal{T}$ is closed under multiplication, i.e. any product of elements is again element of $\mathcal{T}$, the 3D object $\mathcal{P}_{\mathcal{T}}$ forms a *symmetry group*: It is globally symmetric under the group action of any $\mathbf{T} \in \mathcal{T}$. An example for such a symmetry group is a cube, as shown in Figure 2. It is symmetric under 90° rotations and mirroring along all axes.

We often do not observe full symmetry groups but only excerpts, such as the ones shown in Figure 2. In particular, no finite translational symmetry group exists. Therefore, we interpret $\mathcal{T}$ as a subset of a larger, non-observed proper symmetry group $\mathcal{T}'$ whenever at least three repetitions of a transformation are found [BWKS11].

Euclidian symmetry groups. Euclidian symmetry is very well understood; a full classification of all subgroups of $E(3)$ is well known [Hah02]. There are discrete and continuous groups: The discrete groups consist of a countable set of transformations, being generated by between 1 and 3 rotations and/or translations, as well as additional involutions (i.e. reflections/rotations by 180°). In the continuous case, the generators can include instantaneous motions [GG04].

The symmetry groups are implicitly captured by computing all pairwise transformations within $\mathcal{S}$; each element of the group will show up once in a pairwise match. Nonetheless, it is useful to explicitly compute the groups and use them in the optimization. Whenever a structure is generated by a small set of generating transformations, we only enforce symmetry under the action of the generators using Equation (6). In Figure 1 (right), this means that we only constraint the left three elements being mapped to the right three under a single pair of transformations $\mathbf{T}_o, \mathbf{T}_f$ in the source/target domain, respectively. This has two effects: First, this still ensures that the full symmetry group is preserved, because the generators are sufficient to establish the whole group (because the area overlaps, all further transformations are constrained implicitly). In particular, all the transformations that generate the same group are implicitly represented by the same transformation variable (as motivated in Section 3, and Figure 1, right). Secondly, this construction also makes sure that complex group structures (such as a large grid) have the same weight as simple, pairwise symmetries (such as a simple reflection). Without factoring complex groups into a minimal set of generators, the weight of the resulting least-squares constraints would increase because of the many resulting pairwise constraints. As a side effect, the computation costs for building the system matrix are reduced considerably because redundant pairwise constraints do not need to be processed (Figure 2, right).

5.2. Symmetry detection

We use a symmetry detection algorithm closely following the method previously described in [BWKS11]. However, we extend the algorithm to also output rotational symmetry groups. Conceptually, this does not make the detection algorithm more challenging. As described in their paper, we detect discrete translational, reflective and (now also) rotational symmetries by matching edges in the triangle mesh. These matches yield potential generators that are first fused into 1-parameter groups $\{\mathbf{T}^{\mathbb{Z}}\}$ and later into more complex structures. For continuous symmetries, the approach uses slippage analysis [GG04]. Because detecting rotational slippage in meshes is brittle, with difficult thresholding problems, we rely on discrete rotational symmetries of the mesh to detect these. The remaining translational slippage properties (colinearity of mesh edges and coplanarity of mesh faces) are detected directly by comparing normals in the triangle mesh. The two main limitations of this approach are that we (i) need a triangulation where non-flat edges are consistent with the rotational symmetry (thus excluding scanner data as templates) and (ii) we will only obtain cylindrical symmetries for spheres (if the meshing is consistent with that). It is important to note that symmetry detection is not a contribution of this paper; many alternative strategies for this task (such as [MGP06, GCO06, PMW*08, BWSK12]) could be applied as well.

Post-processing. Two filters are used to avoid spurious matches: First, in case the algorithm reports mostly overlapping area with groups where one is a subset of the other, we delete the smaller one. Secondly, very small regions of symmetry, below 0.025 area units for a scene scaled to a unit-bounding box, are discarded to avoid spurious matches. Furthermore, area is only reported as discrete symmetry when it is enclosed by sharp boundaries, as computed by region growing from the discrete feature that triggered the detection. This avoids ghosting artefacts where discrete symmetries bleed into continuous ones: For example, a pair of chairs on a ground plane report only the chairs, not also part of the plane as being symmetric. The plane is nonetheless reported as a continuous symmetry. Sample detection results for several data sets can be found in Figure 3.

6. Implementation

We now combine the variational model of Section 4 with symmetry information (Section 5) to build a symmetry-aware deformable ICP algorithm.

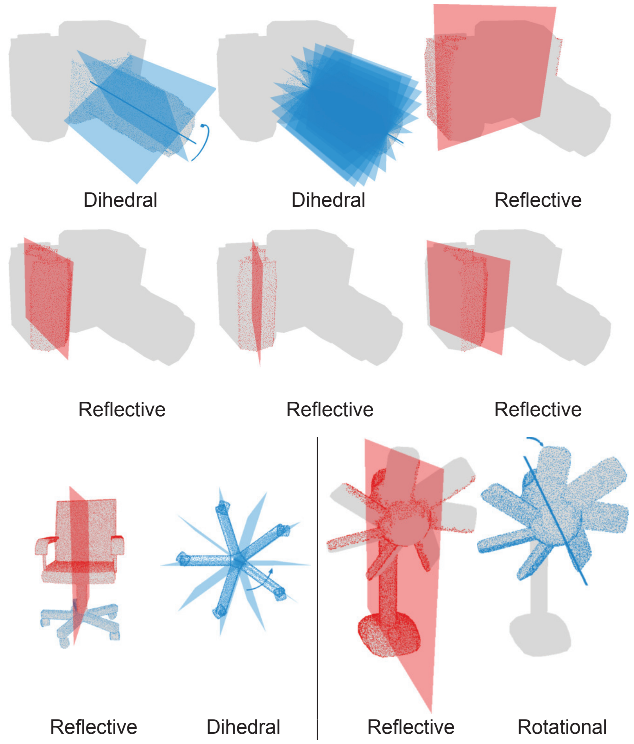

Figure 3: *Symmetry detection results for the DSLR camera (first and second row), office chair (third row, left), and small fan (third row, right) template meshes. Only the points comprised by the respective symmetry are rendered over the grey model backdrop. The reflection planes of the reflective symmetries are depicted in red. The rotational symmetries are depicted by the rotation axis in blue and a small arrow indicating the rotation. For dihedral symmetries, the reflection planes are also shown in blue.*

ICP-like constraints. In order to perform deformable ICP, we maintain a current estimate of the deformed template (initialized by a manual rigid alignment with scaling). For each data point, we compute the closest surface point in the current deformed template shape and create a least-squares point-to-plane attraction constraint employing Equation (4). The surface normal of the scanner data is estimated by a plane fit of the 20 nearest neighbours based on principal component analysis. In addition to these *forward* constraints, we can optionally also include *backward* constraints, where we exchange the role of template and data. Backward constraints are useful if the data are almost complete. For partial data, only forward constraints can be used.

Pruning for robustness. We also prune implausible correspondences to make the algorithm more stable, in a two-stage test: We first remove constraints whose distance $\|f(\mathrm{p}) - \mathbf{d}\|$ is above the t_{d}th percentile of all distances. The parameter t_d has to be chosen by the user according to the expected amount of outliers. Secondly, to avoid oscillation, we use a threshold t_{nf} to determine whether a constraint is located in the near field. We do not remove constraints with a distance below t_{nf} even if their distance lies above the t_{d}th percentile.

As ICP itself is not the focus of this work, we keep our implementation rather basic; more sophisticated correspondence filtering methods can be found in [RL01].

Basis functions. We use linear basis functions during the initial iterations, followed by a final update using smooth B-spline basis functions. As B-splines have a support of four intervals, requiring $4^3 = 64$ matrix entries per constraint, while linear functions only require $2^3 = 8$, this reduces the computational effort significantly. We have also experimented with Wendland functions which yielded visually similar results at higher costs. Therefore, we prefer the B-spline basis.

Constraint sampling. Since the resolution of the discretiziation grid is typically much lower than the average sample spacing of the embedded surfaces, performing all calculated constraints leads to heavy oversampling. We can save running time by sampling all constraints at a sampling factor $\epsilon_{\text{sampling}}$ that is chosen below the Nyquist limit of the discretization grid. In our implementation, a choice of $\epsilon_{\text{sampling}} = 0.25\epsilon_{\text{grid}}$ worked well.

Coplanarity constraints (continuous symmetry). For each set of adjacent planar triangle faces, we take the corresponding point set of $P_i \in S$, and constrain those points to stay on a plane. In particular, we project the points onto the surface along normal direction, and compute the distance for each point pair. Then, we can get a sum-of-squares energy similar to Equation (4), which is solved using the same technique.

Colinearity line constraints (continuous symmetry). We subdivide each feature line (i.e. continuous list of colinear edges in the template mesh) into directed segments using consecutive constraint points extracted at grid cell intersections, then compute the optimal solution to keep their direction vectors the same.

7. Results

We performed scans of everyday objects with a Microsoft Kinect and KinectFusion to obtain target surfaces $\mathcal{D}$ for our examples. These are denoted Target in the result figures. As the ICP implementation is not the focus of this paper, we simplified the scans by removing parts like the floor and the background, which are not related to the original object.

For each object scanned, we then searched a similar 3D model in shape libraries, specifically in Digimation 'The Archive' (DSLR camera, TV set, bowl) and in Google 3D Warehouse (all other models except where noted otherwise), to which we applied the symmetry detection algorithm of Section 5 after we had scaled it to unit length. Next, we manually perform a coarse rigid alignment with scaling. From this initial alignment, we apply our new algorithm as well as several other variants of the ICP algorithm, for comparison.

These variants are Rigid ICP, Affine ICP, Deformable ICP (with TPS regularizer, Equation 5), and our new deformable ICP with additional Symmetry constraints. In order for the results to be comparable, we use the same implementation: Deformable is obtained by switching off the symmetry constraints, and Affine in addition uses a very large weight λ_r such that only affine mappings are obtained. In all cases, we perform 100 ICP iterations (to guarantee convergence). We use linear basis functions, followed by a single iteration with the smooth basis functions in case of Deformable and Symmetry. For Rigid ICP, we use a state-of-the-art ICP implementation [MGPG04].

Symmetry-aware fitting. Figures 4–16 show the results obtained from our approach and various baseline methods. Our symmetry constraints are in particular helpful in inferring the shape of the object in regions of missing data: Figures 4 and 5 show examples where missing and corrupted data are inter and extrapolated using symmetry. In Figure 4, rows 1 and 2, for example, the complete backside of the cooking pot is missing in the scanned data. Our results do not show distortions in the regions of missing data, unlike traditional deformable ICP. The same effect is visible in Figure 4, row 3, which shows a low-quality scan of a pan made of hard-to-acquire reflective metal. Again, maintaining the symmetry of the template avoids implausible deformations.

Figure 5 illustrates a limitation: The handle of the cup is not rigidly symmetric (except from cross-sectional reflection); thus, false ICP correspondences can distort the shape.

Figure 6 shows the results for the camera data set. The rotational symmetry of the lens is preserved by rotational symmetry constraints while the continuous symmetry constraints keep the case of the camera in a rectangular shape. This is a considerable improvement over deformable ICP alone which exhibits numerous distortions and local deformations. Parameters have to be chosen carefully, though; we had to increase the weights for continuous symmetry by a factor of 10 for this example. A comparison to standard parameters is included in the figure. Generally, the need to choose parameters is a limitation of our least-squares formulation.

Traditional deformable ICP. Without the additional symmetry constraints, the deformable ICP is more susceptible to distortions and local deformations. Small irregularities can be compensated (Figure 4, rows 8 and 10) but in regions of missing or highly corrupted data (e.g. Figure 4, rows 1 and 2), simple smoothness is not sufficient. Preserving symmetry leads to more plausible results while still matching the input data: Figure 7 shows an overlay of the different ICP variants with the scanned data.

Rigid ICP. As a sanity check, we also register the scaled template with the data using standard rigid ICP. If the shape of the template mesh is very close to that of the object scanned, this method can produce reasonable results (Figure 4, rows 1, 2 and 10). Often, the available template is too different to give a good alignment (for example, Figure 8). Figure 6 shows a complex example: We reconstruct a plausible camera model from a template that is quite different from the input data; the original, rigidly aligned model does not resemble the target data well (in particular, lens diameter/length).

Further baseline tests. As an additional comparison, we compare to ICP with affine mappings (i.e. permitting rotation, translation, shear and scaling). For some template/scan pairs, this works well (for example, Figure 4, rows 1 and 2) but affine mappings lack the flexibility of general deformations and often cannot match the scan well. Nonetheless, the risk of artefacts increases, too: Particularly, the shearing can lead to heavy distortions. Examples for this can be found in Figure 4, rows 3 and 8, or Figure 9 (bottom).

Figure 4: *Comparison of the results of different ICP variants. From top to bottom: Cooking pot (single-view, front); the same cooking pot (single-view, back); frying pan (single-view); rounded cup (single-view); chair (full scan); the same chair (single-view); armchair (full scan); office chair (full scan); the same office chair (single-view); oval table (full scan); bar table (single-view).*

Figure 5: *Single-view scan of a cone-shaped cup. The symmetry-guided ICP is much closer to the target shape. Only the handle of the cup shows some distortion because the symmetry structure could not be detected properly.*

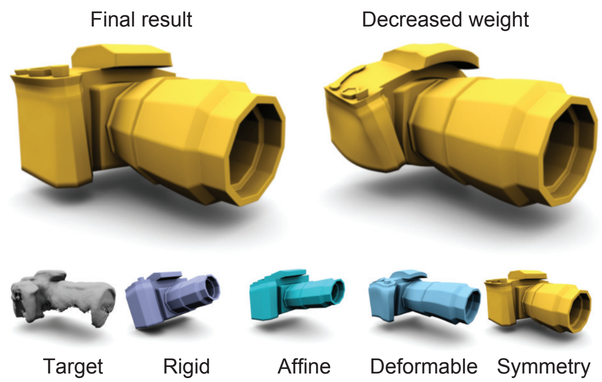

Figure 6: *Scan of a DSLR camera. There are significant differences in proportion between the actual object and the template model. The rotational symmetry of the lens is preserved by the deformable ICP with symmetry constraints. To the best of our knowledge, achieving results of this quality is impossible with methods described in the literature so far. The final result was obtained by increasing the weight of the continuous symmetry constraints. The top right shows the intermediate step with decreased weights.*

Previous structure-aware deformation. We also compare our method to previous work by Bokeloh *et al.* [BWKS11]. Their method does not perform scan registration, but we use their deformation model within our framework. The main difference is that their model only considers translational symmetries. These are (continuous) colinearity, and coplanarity, as well as (discrete) regular grids with at least three instances in each direction. In all of our examples, no discrete grids show up; we therefore reproduce their model by only using continuous translational constraints. As shown in Figure 10, the full set of symmetry constraints yields significantly better results. As seen well in the camera example, the reduced model only maintains straight lines and planes. In contrast, our full model also preserves rotational symmetry and, more importantly, the relations between straight lines and planes by establishing non-local reflective and rotational relations between them. This is clearly visible within the body of the camera or the blades of the fan. The follow-up paper [BWSK12] uses the same translational structure model but hard constraints to reduce residual bending. This crucially relies on the linearity of the group actions, which excludes the rotations supported in our framework.

The continuous symmetry constraints are useful in many applications, but there are cases where these additional constraints are not desired. Figure 9 presents corresponding results for an hourglass

Figure 7: *Overlays of the deformed models and the corresponding target shapes. From top to bottom: Cone-shaped cup, rounded cup and office chair (single-view). In complex cases where no tight template is available, only the deformable ICP variants can match the data closely. Then, the symmetry-guided model proposed in this paper preserves the structure better (as shown in the earlier figures).*

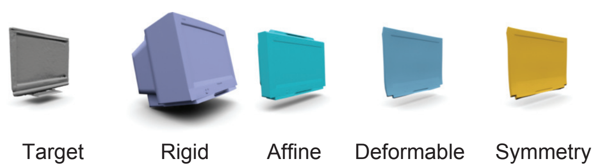

Figure 8: *Scan of an LCD monitor; the template is a CRT TV. We had to disable the coplanarity constraints, as the ICP was otherwise prevented from gradually establishing correspondences in the area that was not initially touched by the target surface, preventing most of the deformation.*

and a bowl, both objects that exhibit rotational symmetries only, where the continuous symmetries have been disabled. The continuous symmetry constraints try to prevent the deformation field from adapting straight edges and planar surfaces to small local deformations in the scanned data. For the hourglass, the cylinder that was used as a template model is just an approximation of a real cylinder. The side consists of many planar surfaces that create coplanarity constraints and the edges that connect them create colinearity constraints. If enforced, these constraints prevent the cylinder from adapting the typical hourglass shape. The situation with the bowl is similar, although there are less planar surfaces. The cylinder used as template mesh for the hourglass was modelled in a professional 3D modelling suite. For the bowl, we used ICP constraints from the template shape to the scanned data in addition to the normal ICP

Figure 9: *Scan of an hourglass (top) and of a bowl (bottom). Due to the nature of the scanned objects (curved surfaces are predominant in the scans), continuous symmetry constraints were not enforced. For the hourglass, for example, the continuous symmetries detected in the mantle of the cylinder would prevent the deformation to the hourglass shape. The symmetry-guided ICP provides excellent results. The traditional deformable ICP exhibits distortions.*

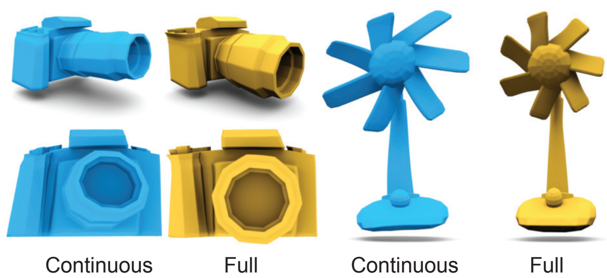

Figure 10: *This figure provides a comparison between [BWKS11, BWSK12]* (Continuous)*, which uses just continuous symmetry constraints, and the deformable ICP with all symmetry constraints* (Full)*. The full set of symmetry constraints governs the deformation on a more global scale and provides better results. The continuous symmetry constraints enforce planar surfaces and straight edges only in localized parts without higher level consistency.*

constraints. While this is often detrimental, particularly if the scan has lots of missing data, in this case it prevented that the whole scanned data were being fitted by only a part of the template.

Quantitative evaluation. A quantitative evaluation of the chair data sets (Figure 4, rows 5 and 6) with ground truth data is shown in Figure 11. The results and the high-quality ground truth reference scan can be seen in Figure 12. In addition, Table 1 provides *root-mean-square error (RMSE)* values. The RMSE values are calculated as

$$e_{\mathrm{RMSE}}(\mathcal{S}, \mathcal{D}) = \sqrt{\frac{1}{\sum_i w_i} \sum_i w_i \left\|\mathbf{p}_j - \mathbf{d}_i\right\|^2}, \qquad (9)$$

with $\mathbf{d}_i \in \mathcal{D}$ and $\mathbf{p}_j \in \mathcal{S}$. The *closest point index* j is selected in a way that makes $\mathbf{p}_j$ the point in $\mathcal{S}$ closest to $\mathbf{d}_i$, as in Equation (4). The weighting factors w_i are chosen as the combined area of all triangles comprising the respective vertex $\mathbf{d}_i$. The values given in the table were obtained by averaging the RMSE from the deformed

Figure 11: *Error visualization of the chair data sets (Figure 4, rows 5 and 6). The upper and lower row use different scans, as shown in Figure 12 (*Full *and* Single*). For visualization, the error values are normalized per data set (row) and range from the lowest error observed (blue) to the highest error observed (red).* Symmetry *exhibits higher error values due to the additional constraints. For the single-view scan (bottom row), the advantage of the additional constraints can be seen:* Deformable *shows a higher error in the rear right leg due to missing data. The scan that served as reference can be found in Figure 12 (right). The corresponding RMSE values are summarized in Table 1.*

Figure 12: *Depiction of the scans used to generate the results for the chair data sets (left and middle) and the scan used to generate the error visualizations shown in Figure 11 (right).*

template mesh to the reference scan and from the reference scan to the deformed template mesh. The deformable ICP variants fit the data more closely. Symmetry exhibits slightly higher error values than Deformable, especially at the borders of the mesh. This is the expected results, as all the additional symmetry constraints influence the result of the optimization procedure, preventing Symmetry from fitting the data as closely as Deformable. The RMSE is also slightly increased in this case. For the single-view scan, Symmetry achieves a better value for the RMSE than Deformable. Again, this is consistent

Table 1: *RMSE values corresponding to the error visualizations shown in Figure 11. The deformable ICP variants exhibits lower errors;* Symmetry *is more constraint and therefore produces a slightly higher error in general. For the single-view scan of the chair, however, the error in* Deformable *is higher due to missing data that leaves one of the chair's legs unconstrained. This is consistent with our expectations. If data are missing, the symmetry constraints allow a better prediction of the shape in those areas. If all the data are available, the data cannot be accommodated to the extend that* Deformable *does.*

	RMSE			
Object	Rigid	Affine	Deformable	Symmetry
Chair, full	0.0296	0.0339	0.0239	0.0246
Chair, single	0.0307	0.0292	0.0246	0.0234

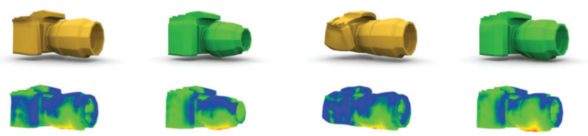

Figure 13: *Comparison between the results for the camera data set (Figure 6) and the results without updates to the symmetry transformations T_f. When the transformations are not updated, the scanned data cannot be accommodated well. To further illustrate the effect, the second row visualizes the error with respect to the scanned data. The scan is missing the whole bottom side of the camera and parts of the bottom of the lens, which can be seen in the visualization as high error values in the bottom part of the lens.*

with the expectations. A lot of data of the rear right of the chair are missing, and Deformable is unconstrained in these areas. Symmetry makes a better prediction of the object shape in this case, and hence the lower RMSE value.

Update of the symmetry transformations T_f. Figure 13 examines the effect of the update of the symmetry transformations T_f on the final result for the camera data set. To this end, the results were recreated without updating the initial transformations. As expected, the lacking update has a serious impact on the algorithm's capabilities to accommodate the scanned model.

Timings. All results were generated using a single-threaded C++ implementation running on an Intel Core i7-840QM processor at 1.87 GHz with 8 GB of RAM. The runtimes for Deformable and Symmetry are summarized in Table 2. Building the system matrix dominates the runtime. Here, symmetry constraints are more costly than other constraint because we have to integrate over large, overlapping areas. The type of basis function also affects the runtime; linear functions have small support (two overlapping functions) while B-splines (overlap factor 4) and Wendland functions (we have used an overlap of 7) are much more costly. We belief that the timings

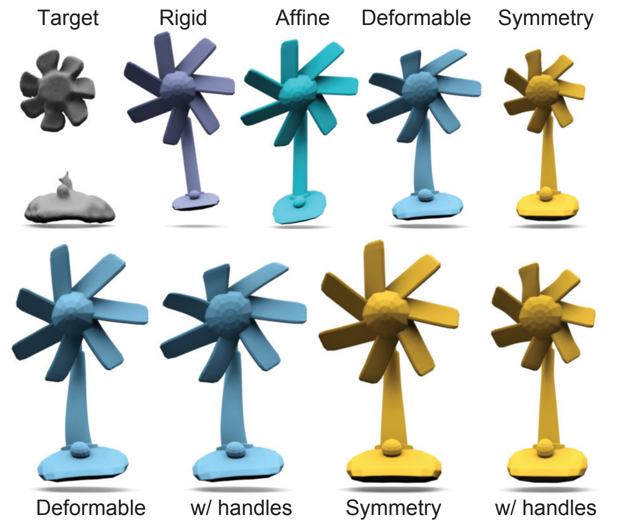

Figure 14: *Scan of a small fan. For the traditional deformable ICP and the symmetry-guided ICP, three handles have been used to shorten the blades of the fan and make the hub less pronounced in the final result. For the symmetry-guided ICP this is sufficient to get a good result due to the propagation of the deformation by the symmetry constraints. For traditional deformable ICP, only the topmost blade is shortened. Comparisons with and without handles are on the bottom. To avoid distortions introduced by the handle constraints, the weights of the symmetry constraints have been increased 10-fold.*

Figure 15: *Full scan of a stepladder (top) and the armchair from Figure 4, row 7 (different template). The symmetry-guided ICP produces a result that is very close to the template model and therefore not a good fit for the data. Spurious symmetries detected for the template in combination with the continuous symmetry constraints prevent most deformation.*

can still be improved substantially; optimizing the numerics this is not in the focus of our paper.

User guidance. Deformable ICP often requires user guidance in addition to a rigid initialization. Again, symmetry constraints reduce the efforts: Figure 14 shows an example. We register a scan of fan against a template (the template shape was created manually for this task). In the initial results (Figure 14, bottom), the size of the rotor

Table 2: *Average time per iteration for traditional deformable ICP and symmetry-guided ICP. The upper part of the examples uses Wendland kernels as basis functions, the lower part uses a cubic B-spline basis. Due to the reduced overlap, the computation times are much shorter (see text). Visually, there is no noticeable difference.*

	Linear basis functions		Smooth basis functions	
Object	Deformable	Symmetry	Deformable	Symmetry
Bar table	0.48 s	2.51 s	0.65 min	19.87 min
Cooking pot	1.18 s	22.86 s	1.97 min	195.23 min
Cone-shaped cup	0.85 s	7.52 s	4.96 min	31.30 min
Rounded cup	0.99 s	7.70 s	5.24 min	84.14 min
Frying pan	1.50 s	6.11 s	1.40 min	53.01 min
Chair, full	0.46 s	2.19 s	1.96 min	24.92 min
Chair, single	0.35 s	2.49 s	0.86 min	24.66 min
Oval table	1.17 s	9.06 s	2.26 min	69.47 min
Armchair	2.74 s	14.82 s	4.68 min	99.19 min
Armchair, simple	1.19 s	3.44 s	2.82 min	26.07 min
Square table	1.48 s	12.53 s	2.76 min	126.70 min
Stepladder	2.51 s	4.93 s	1.03 min	46.03 min
Office chair, full	0.68 s	2.15 s	1.62 min	15.05 min
Office chair, single	0.65 s	1.94 s	1.28 min	14.16 min
Hourglass	1.59 s	7.94 s	0.63 min	6.84 min
Bowl	1.15 s	3.02 s	0.73 min	2.28 min
Fan	0.40 s	1.96 s	0.17 min	1.54 min
Camera	0.93 s	4.56 s	0.53 min	3.51 min
TV	1.55 s	3.20 s	0.59 min	2.11 min

is not correctly matched. The problems occur because the small sides of the blades provide insufficient point-to-plane constraints to successfully match the data. To fix the problem, three handle constraints are sufficient—two at the topmost blade (downward displacement) and one at the hub (fixing the position). We also increase the weight of the symmetry constraints 10-fold to avoid distortions due to the pointwise handles. To achieve the same result with deformable ICP alone, many more, consistently placed handles would be needed. Figure 14 (bottom) shows a comparison for the results of the fan data set with and without handle constraints. Deactivating symmetry constraints leads to worse results. Symmetry-aware deformation with handles can be used to edit and fine-tune the results with greatly reduced effort. A sample editing session for this data set (which demonstrates this improvement) can be found in the video S1 in the Supporting Information.

8. Parameters

If not noted otherwise, the parameters described in this section apply to all results shown in this paper.

Deformation model. We choose a value of 0.1 for the grid spacing ϵ_{grid}, and set the weight of the symmetry constraints λ_s to 3 and the weight of the regularizer λ_r to 1. For the weight of the handle constraints λ_c, we choose a value of 100. In the case of the affine ICP, λ_r is set to 10 000.

Our empiric evaluation showed that setting the weight of the symmetry constraints λ_s at least three times as high as the weight of the ICP-like constraints (that have an implicit weight of 1) is sufficient. For all our examples, the constraints were then respected during the optimization procedure. The one exception was Figure 6, where the weights of the continuous symmetry constraints were not high enough to prevent the distortion of the camera body favoured by the ICP-like constraints. (We increased the weights of the continuous symmetry constraints to a value of 30 for the generation of the final result.) In general, choosing lower values for λ_s removes the bias towards a symmetric solution. The results are then similar to those of traditional deformable ICP. Figure 14 shows results with manually placed handle constraints. The weight of the handle constraints, λ_c, is significantly higher than the weights of the other constraints, leading to local deformations of the fan blade where the constraint is placed. We therefore increased the value of λ_s to 30. With this setting, the local deformations are negligible. The downside is that the handle constraint are no longer perfectly fulfilled.

Symmetry detection. We use a uniform angle threshold of 1° for checking colinearity and coplanarity, and use $4\min(\epsilon_{\text{grid}}, \mu)$ as the distance threshold for checking feature and geometry compatibility, where μ is the length of shortest feature line. After symmetry detection, we examine the symmetries found by the algorithm and set a model-specific cut-off threshold based on the number of supporting points. This cut-off threshold is typically below 50%. All symmetries with less than this number of supporting points are discarded. During the detection of the coplanarity constraints, we require that the planes we select cover at least 1% of the total surface area.

ICP-like constraints. We set the percentile threshold t_d to 80 and the near field threshold t_{nf} to 0.05. The uniform setting for the latter is motivated by the fact that all input models are scaled to unit length during pre-processing.

9. Limitations

Fixed structure. An important limitation is that the symmetries are taken as detected in the template: If the structure differs from the data this creates bias unless we use manual intervention. Manual intervention is used in two examples: For the bowl and hourglass shown in Figure 9, the continuous symmetries where deactivated to be able to use the overly simple templates. We also observed certain issues with the use of particular configurations of symmetry constraints. If multiple symmetries comprise the same parts of the template mesh (e.g. a reflective symmetry that extends into a part that also has a rotational symmetry), such configurations can locally lock the template mesh in place preventing it from deforming (Figure 15; to a lesser extent Figure 4, rows 8 and 10). Again, selectively breaking constraints of the template would be required here.

Local convergence. Sometimes symmetry constraints can prevent the local ICP from converging to a stable set of correspondences. In Figure 16, the horizontal component of the ICP constraints is countered by the symmetry constraints, while the vertical component makes the table extend to the bottom. In subsequent steps, different correspondences are chosen, increasing the effect with each iteration. The same problem also affects affine registration. A similar situation occurs in Figure 4, row 11, for the affine registration. Placing a single handle constraint largely eliminates this behaviour. The traditional deformable ICP does not show these issues, as the regularizer by itself does not influence the ICP constraints as strongly. In one case, Figure 8, the coplanarity constraints almost completely prevented the initial correspondences from updating which lets the algorithm converge with too little deformation. In summary, our algorithm is a locally convergent shape matching technique that requires good initialization and possibly user guidance; it should be considered as a refined (deformable) ICP algorithm, not a fully automatic shape matching system.

Figure 16: *Partial scan of a square table. The ICP algorithm fails to find a stable set of correspondences for the constraint variants affine ICP and symmetry-guided ICP. The affine variant completely diverges. The bottom row shows results with a single handle constraint attached to a table leg.*

10. Conclusion and Future Work

We have presented a constrained optimization framework for symmetry-aware template mesh deformation that fits a user-provided 3D model to low-quality scanning data. The symmetry structure of the template geometry is preserved in a least-squares sense. We have validated the method by reconstructing a number of low-quality KinectFusion scans using suitably chosen template models. In comparison to previous methods, we obtain more plausible reconstructions that match the scanned geometry closely even if the template is quite different.

In future work, we would like to automatically select suitable subsets of symmetry constraints obtained from analysing both the template and the input data (which is more challenging due to potentially high noise levels and distortions). It would also be interesting to investigate fully automatic alignment of the template mesh and scanned data. To this end, symmetry properties such as symmetry axes and planes might be useful invariants that could be matched even if the geometry differs substantially. Currently, the runtime costs of construction and evaluation of the symmetry constraints make the framework not suitable for real-time user interaction. We would like to investigate the reformulation of the numerics for higher speed and possibly a suitable general-purpose computation on graphics processing units implementation to make the approach more interactive.

Another avenue for future research is the automatic selection of a suitable template mesh from a database (e.g. following the approach in [KMHG13], possibly by conjointly performing a symmetry analysis), which would then no longer require the user to select a suitable model manually.

We think that in the future there will be an increasing demand for more general structure models, beyond rigid symmetry. Incorporating weakly supervised machine learning to establish correspondences and learn more general groups of admissible mappings might be a potential avenue towards addressing this difficult challenge.

We would also like to investigate methods to overcome the limitations of the fixed template (which include fixed topology), such as the combination of a coarse initial template with implicit function fitting. This might enable us to perform topology-varying deformation while preserving the symmetry structure of the template.

Acknowledgements

This work has been partially funded by the Max Planck Center for Visual Computing and Communication (BMBF-FKZ01IMC01), the Software Campus (MPG [BMBF-FKZ01IS2055]), the Microsoft Research PhD Scholarship program and the Intel Visual Computing Institute. In addition, we thank Martin Bokeloh for constructive discussions.

References

[ACP03] Allen B., Curless B., Popović, Z.: The space of human body shapes: Reconstruction and parameterization from range scans. In *ACM SIGGRAPH 2003 Papers* (New York, NY, USA, 2003), SIGGRAPH '03, ACM, pp. 587–594.

[ARV07] Amberg B., Romdhani S., Vetter T.: Optimal step nonrigid ICP algorithms for surface registration. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (Minneapolis, MN, USA, 2007), IEEE.

[AS07] Avidan S., Shamir, A.: Seam carving for content-aware image resizing. *ACM Transactions on Graphics 26*, 3 (July 2007), 10:1–10:9.

[ASK*05] Anguelov D., Srinivasan P., Koller D., Thrun S., Rodgers J., Davis J.: Scape: Shape completion and animation of people. *ACM Transactions on Graphics*, *24*, 3 (July 2005), 408–416.

[BCWG09] Ben-Chen M., Weber O., Gotsman C.: Variational harmonic maps for space deformation. *ACM Transactions on Graphics 28*, 3 (July 2009), 34:1–34:11.

[BK04] Botsch M., Kobbelt L.: An intuitive framework for real-time freeform modeling. *ACM Transactions on Graphics 23*, 3 (August 2004), 630–634.

[BKS03] Bendels G. H., Klein R., Schilling A.: Image and 3D-object editing with precisely specified editing regions. In *Vision, Modeling and Visualisation 2003*. T. Ertl, B. Girod, G. Greiner, H. Niemann, H.-P. Seidel, E. Steinbach, R. Westermann (Eds.). Akademische Verlagsgesellschaft Aka GmbH, Berlin (November 2003), pp. 451–460.

[BR07] Brown B. J., Rusinkiewicz S.: Global non-rigid alignment of 3-D scans. *ACM Transactions on Graphics 26*, 3 (July 2007), 21:1–21:9.

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

[BSVS04] Blanz V., Scherbaum K., Vetter T., Seidel H.-P.: Exchanging faces in images. In *Proceedings of EUROGRAPHICS* '04 (Grenoble, France, 2009), vol. *23*, Blackwell, pp. 669–676.

[BWKS11] Bokeloh M., Wand M., Koltun V., Seidel H.-P.: Pattern-aware shape deformation using sliding dockers. *ACM Transactions on Graphics 30*, 6 (December 2011), 123:1–123:10.

[BWS10] Bokeloh M., Wand M., Seidel H.-P.: A connection between partial symmetry and inverse procedural modeling. *ACM Transactions on Graphics 29*, 4 (July 2010), 104:1–104:10.

[BWSK12] Bokeloh M., Wand M., Seidel H.-P., Koltun V.: An algebraic model for parameterized shape editing. *ACM Transactions on Graphics 31*, 4 (July 2012), 78:1–78:10.

[CKGK11] Chaudhuri S., Kalogerakis E., Guibas L., Koltun V.: Probabilistic reasoning for assembly-based 3D modeling. *ACM Transactions on Graphics 30*, 4 (July 2011), 35:1–35:10.

[CM09] Chen L., Meng X.: Anisotropic resizing of model with geometric textures. In *2009 SIAM/ACM Joint Conference on Geometric and Physical Modeling* (New York, NY, USA, 2009), SPM '09, ACM, pp. 289–294.

[FKS*04] Funkhouser T., Kazhdan M., Shilane P., Min P., Kiefer W., Tal A., Rusinkiewicz S., Dobkin D.: Modeling by example. *ACM Transactions on Graphics 23*, 3 (August 2004), 652–663.

[GCO06] Gal R., Cohen-Or D.: Salient geometric features for partial shape matching and similarity. *ACM Transactions on Graphics 25*, 1 (January 2006), 130–150.

[GG04] Gelfand N., Guibas L. J.: Shape segmentation using local slippage analysis. In *Proceedings of the 2004 Eurographics/ACM SIGGRAPH Symposium on Geometry Processing* (New York, NY, USA, 2004), SGP '04, ACM, pp. 214–223.

[GSMCO09] Gal R., Sorkine O., Mitra N. J., Cohen-Or D.: iWires: An analyze-and-edit approach to shape manipulation. *ACM Transactions on Graphics 28*, 3 (July 2009), 33:1–33:10.

[Hah02] Hahn T.: *International Tables for Crystallography, Volume A: Space Group Symmetry*. Springer, Berlin, 2002.

[HMC09] Huang Q., Mech R., Carr N.: Optimizing structure preserving embedded deformation for resizing images and vector art. In *Proceedings of Pacific Graphics* (Jeju, Korea, 2009), vol. *28*, pp. 1887–1896.

[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 Transactions on Graphics 25*, 3 (July 2006), 1126–1134.

[HSS*09] Hasler N., Stoll C., Sunkel M., Rosenhahn B., Seidel H.-P.: A statistical model of human pose and body shape. *Computer Graphics Forum 28*, 2 (2009), 337–346.

[HTB03] Hahnel D., Thrun S., Burgard W.: An extension of the ICP algorithm for modeling nonrigid objects with mobile robots. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence* (San Francisco, CA, USA, 2003), IJCAI'03, Morgan Kaufmann Publishers Inc., pp. 915–920.

[IKH*11] Izadi S., Kim D., Hilliges O., Molyneaux D., Newcombe R., Kohli P., Shotton J., Hodges S., Freeman D., Davison A., Fitzgibbon A.: Kinectfusion: Real-time 3D reconstruction and interaction using a moving depth camera. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology* (New York, NY, USA, 2011), UIST '11, ACM, pp. 559–568.

[JMD*07] Joshi P., Meyer M., DeRose T., Green B., Sanocki T.: Harmonic coordinates for character articulation. *ACM Transactions on Graphics 26*, 3 (July 2007), 71:1–71:9.

[JSW05] Ju T., Schaefer S., Warren J.: Mean value coordinates for closed triangular meshes. *ACM Transactions on Graphics 24*, 3 (July 2005), 561–566.

[JTRS12] Jain A., Thormählen T., Ritschel T., Seidel H.-P.: Exploring shape variations by 3D-model decomposition and part-based recombination. *Computer Graphics Forum 31*, 2pt3 (May 2012), 631–640.

[KCKK12] Kalogerakis E., Chaudhuri S., Koller D., Koltun V.: A probabilistic model for component-based shape synthesis. *ACM Transactions on Graphics 31*, 4 (July 2012), 55:1–55:11.

[KMHG13] Kim Y. M., Mitra N. J., Huang Q., Guibas L.: Guided real-time scanning of indoor objects. *Computer Graphics Forum 32*, 7 (2013), 177–186.

[KMYG12] Kim Y. M., Mitra N. J., Yan D.-M., Guibas L.: Acquiring 3D indoor environments with variability and repetition. *ACM Transactions on Graphics 31*, 6 (November 2012), 138:1–138:11.

[KS05] Kraevoy V., Sheffer A.: Template-based mesh completion. In *Proceedings of the Third Eurographics Symposium on Geometry Processing* (Aire-la-Ville, Switzerland, 2005), SGP '05, Eurographics Association.

[KSSCO08] Kraevoy V., Sheffer A., Shamir A., Cohen-Or D.: Non-homogeneous resizing of complex models. *ACM Transactions on Graphics 27*, 5 (December 2008), 111:1–111:9.

[LLCO08] Lipman Y., Levin D., Cohen-Or D.: Green coordinates. *ACM Transactions on Graphics 27*, 3 (August 2008), 78:1–78:10.

[LWC*11] Li Y., Wu X., Chrysathou Y., Sharf A., Cohen-Or D., Mitra N. J.: Globfit: Consistently fitting primitives by discovering global relations. In *ACM SIGGRAPH 2011 Papers* (New York, NY, USA, 2011), SIGGRAPH '11, ACM, pp. 52:1–52:12.

[Mer07] Merrell P.: Example-based model synthesis. In *I3D 07: Proceedings of the 2007 Symposium on Interactive 3D Graphics and Games* (Seattle, WA, USA, 2007), ACM Press, pp. 105–112.

[MGP06] Mitra N. J., Guibas L. J., Pauly M.: Partial and approximate symmetry detection for 3D geometry. *ACM Transactions on Graphics 25*, 3 (July 2006), 560–568.

[MGPG04] Mitra N. J., Gelfand N., Pottmann H., Guibas L.: Registration of point cloud data from a geometric optimization perspective. In *Proceedings of the 2004 Eurographics/ACM SIGGRAPH Symposium on Geometry Processing* (New York, NY, USA, 2004), SGP '04, ACM, pp. 22–31.

[Mil72] Miller W.: *Symmetry Groups and Their Applications*. Academic Press, Waltham, MA, USA, 1972.

[MPWC13] Mitra N. J., Pauly M., Wand M., Ceylan D.: Symmetry in 3D geometry: Extraction and applications. *Computer Graphics Forum 32*, 6 (2013), 1–23.

[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), 2013.

[PMG*05] Pauly M., Mitra N. J., Giesen J., Gross M., Guibas L. J.: Example-based 3D scan completion. In *Proceedings of the Third Eurographics Symposium on Geometry Processing* (Aire-la-Ville, Switzerland, 2005), SGP '05, Eurographics Association.

[PMW*08] Pauly M., Mitra N. J., Wallner J., Pottmann H., Guibas L. J.: Discovering structural regularity in 3D geometry. *ACM Transactions on Graphics 27*, 3 (August 2008), 43:1–43:11.

[RL01] Rusinkiewicz S., Levoy M.: Efficient variants of the ICP algorithm. In *Proceedings of 3DIM '01* (Quebec City, Canada, 2001), pp. 145–152.

[SFCH12] Shen C.-H., Fu H., Chen K., Hu S.-M.: Structure recovery by part assembly. *ACM Transactions on Graphics 31*, 6 (November 2012), 180:1–180:11.

[SSP07] Sumner R. W., Schmid J., Pauly M.: Embedded deformation for shape manipulation. In *ACM SIGGRAPH 2007 Papers* (New York, NY, USA, 2007), SIGGRAPH '07, ACM.

[TPBF87] Terzopoulos D., Platt J., Barr A., Fleischer K.: Elastically deformable models. In *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 1987), SIGGRAPH '87, ACM, pp. 205–214.

[TW05] Thrun S., Wegbreit B.: Shape from symmetry. In *Proceedings of the International Conference on Computer Vision (ICCV)* (Beijing, China, 2005), IEEE.

[WJH*07] Wand M., Jenke P., Huang Q., Bokeloh M., Guibas L., Schilling A.: Reconstruction of deforming geometry from time-varying point clouds. In *Proceedings of the Fifth Eurographics Symposium on Geometry Processing* (Aire-la-Ville, Switzerland, 2007), SGP '07, Eurographics Association, pp. 49–58.

[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 (Special Issue of Eurographics) 30*, 2 (2011), 287–296.

[XWY*09] XU W., WANG J., YIN K., ZHOU K., VAN DE PANNE M., CHEN F., GUO B.: Joint-aware manipulation of deformable models. *ACM Transactions on Graphics 28*, 3 (July 2009), 35:1–35:9.

[XZZ*11] XU K., ZHENG H., ZHANG H., COHEN-OR D., LIU L., XIONG Y.: Photo-inspired model-driven 3D object modeling. *ACM Transactions on Graphics 30*, 4 (July 2011), 80:1–80:10.

[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.

Supporting Information

Additional Supporting Information may be found in the online version of this article at the publisher's web site:

Video S1