Eurographics Symposium on Geometry Processing (2006)
Konrad Polthier, Alla Sheffer (Editors)

PriMo: Coupled Prisms for Intuitive Surface Modeling

Mario Botsch — ETH Zurich
Mark Pauly — ETH Zurich
Markus Gross — ETH Zurich
Leif Kobbelt — RWTH Aachen

Abstract

We present a new method for 3D shape modeling that achieves intuitive and robust deformations by emulating physically plausible surface behavior inspired by thin shells and plates. The surface mesh is embedded in a layer of volumetric prisms, which are coupled through non-linear, elastic forces. To deform the mesh, prisms are rigidly transformed to satisfy user constraints while minimizing the elastic energy. The rigidity of the prisms prevents degenerations even under extreme deformations, making the method numerically stable. For the underlying geometric optimization we employ both local and global shape matching techniques. Our modeling framework allows for the specification of various geometrically intuitive parameters that provide control over the physical surface behavior. While computationally more involved than previous methods, our approach significantly improves robustness and simplifies user interaction for large, complex deformations.

Categories and Subject Descriptors (according to ACM CCS): I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling Geometric Transformations;

1. Introduction

In recent years, significant progress has been made in establishing triangle meshes as a representation for advanced geometric modeling. One of the most challenging geometry processing operations for meshes is high quality shape deformation. Ideally, mesh editing should be interactive and intuitive at the same time, but the large space of possible shape modifications often leaves the effects of user-controlled constraints hard to predict. With presently available methods achieving interactive or even real-time performance on large triangle meshes [BK04, SCOL*04, LSLCO05, BK05], the amount of "guidance" required from the designer remains a major bottleneck. Very often, the inherent limitations of the underlying deformation models force designers to split up complex deformations into a sequence of smaller ones.

Physically accurate surface deformations require the minimization of non-linear stretching and bending energies, resulting in the well known thin-plate and thin-shell functionals [TPBF87,CG91,WW92]. Linearization of curvatures with respect to a fixed reference mesh, an approximation often utilized by current mesh modeling approaches, leads to parametric distortions for large deformations and thus to a degradation of the surface. As a consequence, complex deformations have to be split up into multiple smaller ones, which complicates the overall modeling process.

We propose a new intuitive and robust shape modeling approach based on a *non-linear* surface deformation model. Our approach is inspired by the physical behavior of thin shells and computes intuitive deformations by emulating the natural material behavior of surfaces we experience in real-life. Rather than simulating accurate deformation physics we achieve *physically plausible* behavior while retaining interactive performance. Although our method is computationally more involved than previous approaches, it trades computational effort with the time the designer requires for guiding the modeling process.

Our underlying surface deformation model is based on a layer of *rigid* prisms which is enveloping the mesh faces. The prisms are coupled through non-linear elastic energies, which naturally resist stretching and bending and thus emulate the mechanical behavior of thin shells and plates. The rigidity of the prisms prevents them from degenerating even under extreme deformations, thus making our method numerically stable. The prisms' rigid motions are guided by a global shape matching procedure. We adapt and improve techniques developed for simultaneous registration of multiple objects [PLH02] for the efficient solution of the underlying geometric optimization. Its robustness and efficiency enable our surface model to be incorporated into an interactive and intuitive shape deformation application.

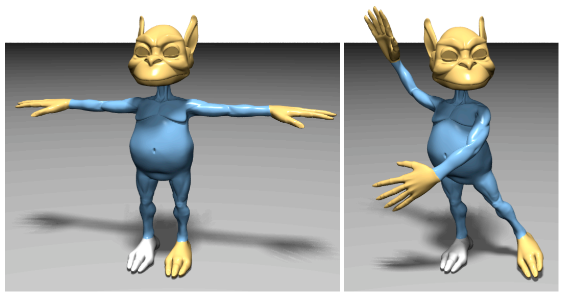


Figure 1: *The Goblin was posed by prescribing position and orientation for its head and right hand. For the left hand and foot positions were constrained only, thus enabling the automatic optimization of their orientations. The natural bending of joints was easily achieved by reducing the surface stiffness in these regions. The whole editing session took less than 5 minutes (see the accompanying video).*

Contributions. Our major contribution is a non-linear surface deformation model based on elastically connected rigid prisms, which features

- Robust and physically plausible large-scale deformations
- Intuitive preservation of surface details
- Hard constraints for positions and orientations
- Constraint-based and force-based deformations
- Intuitive geometric parameters for surface modeling

In addition, we present a robust and efficient numerical optimization method which combines local and global shape matching techniques. Based on these components, our surface deformation approach allows for intuitive and interactive shape deformations, as shown in Figure 1.

2. Related Work

Several shape editing approaches, like freeform deformation [SP86] and Wires [SF98], focus on shape *design* rather than on physically inspired shape *deformation*. While these methods typically provide more flexibility than physically-based ones, they in turn require more guidance if physically plausible deformations are actually desired.

The *design* of high quality surfaces is typically based on a constrained minimization of curvature energies [WW92, MS92], which results in thin-plate surfaces with planar rest states. Similarly, high quality physically-based *deformations* minimize stretching and bending (i.e., the change of curvature) under deformation constraints, which corresponds to thin-shell models of non-planar rest states [TPBF87, CG91].

Most recent physically inspired mesh deformation approaches can be categorized into two classes, one minimizing bending energies, the other one modifying differential coordinates. Figure 2 compares the behavior of a subset of the methods outlined below on simple synthetic examples.

Shape modeling based on a discretization of variational bending energy minimization (VARMIN in Figure 2) is mathematically well understood and yields smooth and tangent-continuous deformations [KCVS98, GSS99, BK04]. Similarly, [BK05] minimizes an analogous energy for space deformations (RBF in Figure 2). However, since for these approaches all computations and linearizations are performed w.r.t. a *fixed* reference mesh, large deformations might lead to shape distortions.

To correctly deform fine surface details, the above methods require a multi-scale decomposition, which splits a surface into a smooth base surface (low frequencies) and displacement vectors (high frequencies). Changing the smooth base surface and adding the details back onto it then yields the desired multi-scale deformation [KVS99]. The bottom row of Figure 2 shows the advantage of the *multi-scale* technique VARMIN over the *single-scale* RBF method, although even the multi-scale technique distorts the left-most bumps in a counter-intuitive manner.

Displacement volumes [BK03] encode the high frequencies by prism elements enclosed between the original and the base surface, which avoids detail distortion, but comes at the considerably higher cost of a non-linear detail reconstruction. Notice that displacement volumes are a multi-scale representation only, not a surface deformation technique, like the one presented in this paper. Although both representations (displacement vectors/volumes) can be combined with any underlying deformation technique, the required multi-scale decomposition can become quite involved for geometrically or topologically complex models.

To avoid the multi-scale decomposition, other methods modify differential surface properties instead of its spatial coordinates, and then solve a linear Poisson system for a deformed surface with the desired differential coordinates [LSCO*04, SCOL*04, YZX*04, ZRKS05, LSLCO05].

From this class, the methods of [YZX*04, ZRKS05] use gradients of affine deformations, i.e., their rotation and scale/shear components, for transforming surface gradients (GRAD), similar to [SP04, SZGP05]. As a consequence, these methods work well for rotations, but are insensitive to translations: Adding a translation to a given deformation does not change its gradient, and thus has no influence on the resulting surface gradients. But since even pure translations induce local rotations of tangent planes (Figure 2, bottom), these methods are counter-intuitive for modifications containing large translations. Although a special treatment of pure translations might be possible, deformations containing rotations and translations remain problematic.

In contrast, the shape editing approach of [SCOL*04] implicitly solves for local rotations of vertex neighborhoods, but due to linearizations their method has problems with large rotations, as was shown in their follow-up paper [LSLCO05]. In that paper, Lipman et al. minimize bending by preserving relative per-vertex orientations. They first

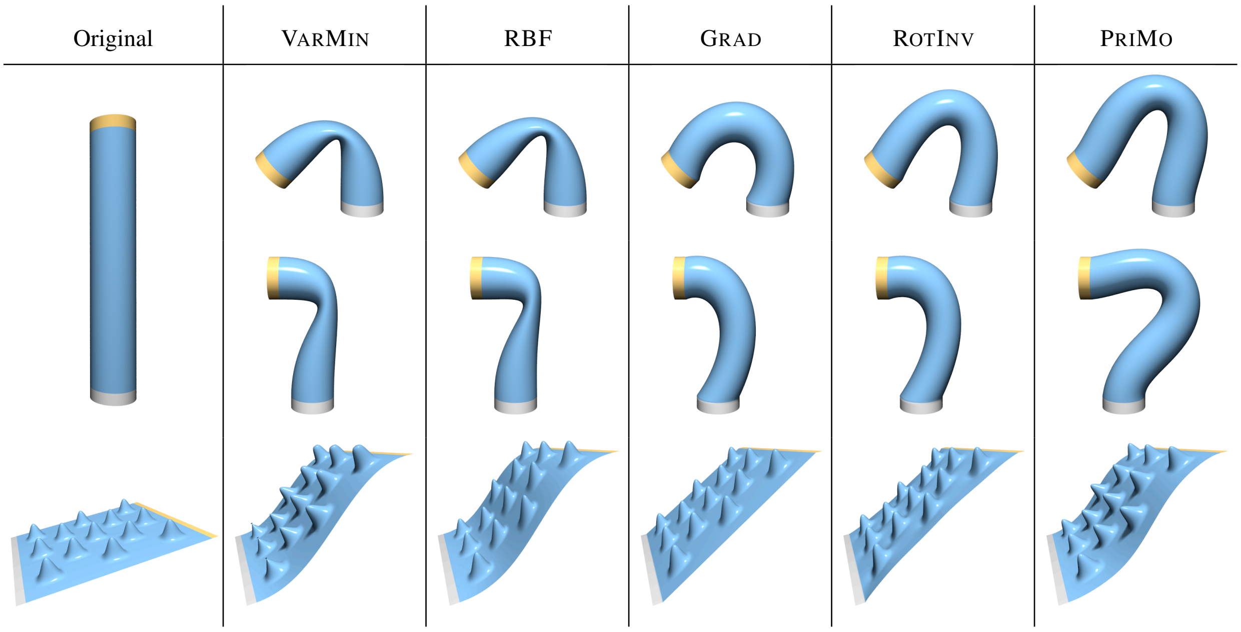


Figure 2: *Comparison of* VARMIN [BK04]*,* RBF [BK05]*,* GRAD [ZRKS05]*,* ROTINV [LSLCO05]*, and the proposed* PRIMO*, based on large, one-step deformations by rotations* (top)*, rotations and translations* (center)*, and pure translations* (bottom)*. While* VARMIN *and* RBF *work well for translations, they have problems with large rotations, whereas* GRAD *and* ROTINV *exhibit exactly the opposite behavior. Deformations containing large rotations as well as translations are therefore difficult for all of them. Moreover, the linear approaches lead to a noticeable loss of material for the two cylinder examples. In contrast, our non-linear deformable surface model* PRIMO *successfully handles all cases.*

solve a linear system for per-vertex orientations, and from those reconstruct vertex positions in a second step. Since the first system does not consider position constraints, their technique also neglects the connection between translations and rotations. While their method works very well even for large rotations, it exhibits the same translation-insensitivity as gradient-based methods (ROTINV in Figure 2).

The sketch-based deformation methods of [NSACO05, ZHS*05] provide more guidance to the system by deforming curves on the surface and propagating their local rotations over a region of interest. Since both methods are based on differential coordinates, they are in principle affected by translation-insensitivity, but the dense curve constraints avoid these problems in general. While a curve-based modeling metaphor would also be possible for our representation, we focus on sparse deformation constraints and a simple click & drag user interface.

Notice that all deformations of Figure 2 were done in a single large step, although the linear methods would perform better when splitting them into multiple smaller ones. But this requires to re-factorize the involved matrices in each step, which considerably increases computation costs. Moreover, since it changes the reference parametrization, a simple "undo" operation by moving the constrained vertices back to their original positions is disabled.

These inherent limitations of linear methods motivated us to investigate *non-linear* deformation techniques. In this context, Sheffer and Kraevoy [SK04] proposed non-linear, rigidly invariant pyramid coordinates. Their method corresponds to a non-linear extension of differential coordinates, and was shown to be capable of large deformations.

In contrast, we employ a constrained minimization of a non-linear bending energy. Since we aim at a qualitative emulation of the mechanical behavior of thin shells, we can provide more intuitive parameters for controlling the surface deformation. Notice that our method does not require a multi-scale decomposition, since our non-linear optimization correctly accounts for the coupling of translations and local frame rotations.

Although our approach is related to recent sophisticated shell simulations [GHDS03, WSG05], the latter usually follow different goals, since they are interested in the dynamic behavior of objects, including masses, inertia, and collisions. Their involved computations typically are not designed to robustly handle the arbitrarily extreme user constraints of interactive modeling applications. Instead of a *physically accurate* fully dynamic simulation we are explicitly targeting *physically plausible* shape deformations only, and thus can trade off physical accuracy for computational efficiency and numerical robustness.

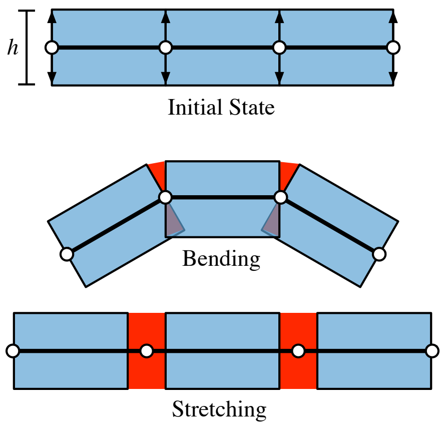


Figure 3: *After extruding prisms from the faces of the input mesh along vertex normals* (top), *the distortion of elastic joints between neighboring prisms is used to measure deformation energy, e.g., for bending or stretching.*

3. Prism Representation

Thin shells are volumetric objects of almost vanishing thickness and can therefore be modeled by a thin volumetric layer wrapped around a center surface [Bat95]. Following this intuition we consider the input mesh as the center surface, and the volume enclosed between two offset surfaces as the volumetric layer. This layer is built by an extrusion along vertex normals, which results in a (in general non-orthogonal) prism P_i for each mesh face F_i (cf. Figure 3, top).

A standard FEM formulation would be the straightforward way to generalize the shell's deformation energy to these prisms. However, most finite element techniques become numerically instable as soon as elements degenerate, since then neither volumes, areas, nor gradients can be computed robustly [ITF04]. Unfortunately, this kind of degeneracies is particularly likely to occur in interactive modeling applications, for instance due to extreme forces or constraints applied by the user.

In order to ensure numerical robustness even under extreme deformations, we prevent prisms from degenerating by keeping them rigid. We connect the rigid prisms along their common faces by *elastic joints*, which are stretched under deformations. The amount of stretching then yields the desired deformation energy (cf. Figure 3).

For the definition of the elastic joint energy we consider two neighboring prisms P_i and P_j. In the undeformed state the two prisms share a common face, but after a deformation these side faces might no longer coincide. The face of P_i neighboring P_j is a rectangular bi-linear patch $\mathbf{f}^{i\to j}(u,v)$, $(u,v)\in[0,1]^2$, which interpolates its four corner vertices $\{\mathbf{f}_{00}^{i\to j},\mathbf{f}_{10}^{i\to j},\mathbf{f}_{01}^{i\to j},\mathbf{f}_{11}^{i\to j}\}$. Analogously we denote the opposite face by $\mathbf{f}^{j\to i}(u,v)\subset P_j$ (cf. Figure 4).

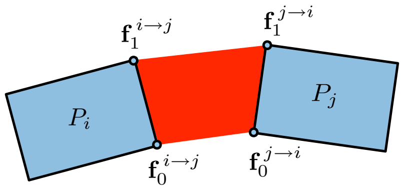


Figure 4: *Notation for prism elements.*

We define the energy between P_i and P_j as

$$E_{ij} := \int_{[0,1]^2} \left\| \mathbf{f}^{i\to j}(u,v) - \mathbf{f}^{j\to i}(u,v) \right\|^2 dudv \ , \qquad (1)$$

which corresponds to an integral over infinitesimal elastic forces, that can be thought of as fibers of the elastic joint. As shown in the appendix, the above equation evaluates to a simple quadratic expression in the four difference vectors $(\mathbf{f}_{kl}^{j\to i} - \mathbf{f}_{kl}^{j\to i})$, $k,l\in\{0,1\}$. The deformation energy of the whole mesh can now be defined as an accumulation of pairwise energies E_{ij}

$$E := \sum_{\{i,j\}} w_{ij}\cdot E_{ij} \ , \quad w_{ij} := \frac{\|\mathbf{e}_{ij}\|^2}{|F_i|+|F_j|} \ , \qquad (2)$$

where the energy contribution of each pair P_i, P_j is weighted by the areas of the corresponding mesh faces F_i, F_j, and the squared length of their shared edge $\mathbf{e}_{ij}$ [GHDS03]. Notice that due to the zero rest length of the elastic joints the initial (undeformed) configuration of prisms is the unique global minimum of the energy, and any bending, shearing, twisting, or stretching increases it.

Our prism-based modeling metaphor works as follows: The user prescribes positions and/or orientations of an arbitrary subset of prisms. The optimization technique described in the next section then finds individual rigid motions for all unconstrained prisms, such that the global deformation energy (2) is minimized. The deformed surface mesh is finally derived from the resulting prisms by updating the position of each unconstrained vertex using the average transformation of its incident prisms.

As an additional benefit besides robustness, the prism formulation provides geometrically intuitive parameters for controlling the surface behavior: The extrusion amount h, i.e., the layer's thickness, determines the local surface stiffness, since for taller prisms the same bending angle induces a higher joint stretching (cf. Figure 5).

Another important property of our formulation is that it is not restricted to pure triangle meshes, but can also be applied to arbitrary polygonal meshes. Extruding a prism from an n-gon then simply generates an n-sided prism instead of a triangular one. Especially when dealing with regular quad meshes generated by CAD systems, inserting an arbitrary diagonal edge to split quads into triangles leads to asymme-

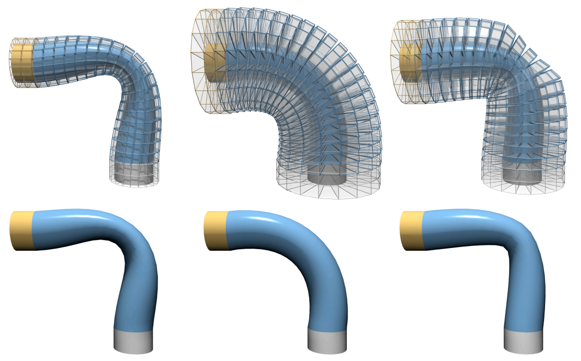


Figure 5: *The prisms' height intuitively controls the surface's resistance to bending. Local stiffness adjustment additionally allows to concentrate bending at desired joint locations, like the Goblin's shoulders and elbows in Figure 1.*

tries. In contrast, our method allows to directly process these meshes and thereby preserve their inherent symmetries.

Notice that the initial prism generation might lead to local self-intersections in the offset surfaces in regions of high curvature, which would lead to locally inverted prisms. However, as our energy only considers the elastic joints *between* prisms, it is not negatively affected by these configurations, which also holds for prisms interpenetrating during deformations, like those inside the cylinder in Figure 5.

4. Numerical Solution

In this section we propose a robust and efficient technique for the constrained minimization of the deformation energy defined in Equation (2). Our approach is based on generalized shape matching and adapts both local and global shape matching techniques in order to combine them to a hierarchical multigrid solver.

The user controls the surface deformation by constraining the position and/or orientation of certain prisms (respectively faces). The optimization then finds optimal rotations $\mathbf{R}_i$ and translations $\mathbf{t}_i$ for the unconstrained prisms P_i, such that the deformation energy (2) is minimized (we replace (u,v) by $\mathbf{u}$ for notational convenience):

$$\min_{\{\mathbf{R}_i,\mathbf{t}_i\}} \sum_{\{i,j\}} w_{ij} \int_{[0,1]^2} \left\| \mathbf{R}_i \mathbf{f}^{i\to j}(\mathbf{u}) + \mathbf{t}_i - \mathbf{R}_j \mathbf{f}^{j\to i}(\mathbf{u}) - \mathbf{t}_j \right\|^2 d\mathbf{u}. \tag{3}$$

This minimization actually corresponds to a generalized global shape matching problem: Discretizing the integrals by summations over sample points $\mathbf{f}^{i\to j}(u_k,v_k)$ would lead to a global alignment problem for multiple point sets, where rigid motions are to be found to minimize the sum of squared point distances.

As a consequence, the minimization (3) can be thought of as global alignment of prisms based on *continuous* face correspondences, instead of *discrete* point correspondences. This continuous formulation is mathematically more elegant compared to a sufficiently dense point-sampling of prism faces, and is also quite efficient, since the involved integrals evaluate to simple quadratic functions, as shown in the appendix.

For the global alignment of multiple point sets a large variety of techniques has been proposed, being based on either local pairwise alignment or simultaneous global registration. We will adapt both techniques to our problem in Section 4.1 and Section 4.2, and combine both them to an efficient multigrid solver in Section 4.3.

4.1. Local Shape Matching

A common approach to global registration is based on iterated pairwise alignment. In each iteration one prism P_i is randomly chosen and its position and orientation is optimized w.r.t. the remaining ones, which are kept fixed. Since each iteration minimizes the *local* shape matching error of P_i and does not change the other prisms, the *global* shape matching error decreases monotonically.

When picking a prism P_i for optimization, we have to match its faces $\mathbf{f}^{i\to j}$ to the corresponding faces $\mathbf{f}^{j\to i}$ of its neighbors, which we denote as $P_j,\ j\in\mathcal{N}_i$. Finding the best rigid motion $(\mathbf{R}_i,\mathbf{t}_i)$ yields a weighted pairwise shape matching problem

$$\min_{\mathbf{R}_i,\mathbf{t}_i} \sum_{j\in\mathcal{N}_i} w_{ij} \int_{[0,1]^2} \left\| \mathbf{R}_i \mathbf{f}^{i\to j}(\mathbf{u}) + \mathbf{t}_i \;-\; \mathbf{f}^{j\to i}(\mathbf{u}) \right\|^2 d\mathbf{u}\ , \tag{4}$$

for which a simple closed-form solution can be computed by generalizing the method of [Hor87] to continuous face correspondences (see appendix).

This iterated pairwise matching is efficient and simple to implement, since each matching only requires an eigenvector decomposition of a 4×4 matrix (67k matches/sec on a 3.2GHz P4). Müller et al. [MHTG05] successfully used a similar discrete formulation for their deformation approach, but the number of clusters to be matched in their case is rather small compared to our number of prisms.

The main limitation of the local matching is that it corresponds to an error diffusion, and hence exhibits the typical behavior of iterative smoothers: for large systems the high frequencies of the error are rapidly attenuated, but the low frequencies — which correspond to the desired global deformations — take impractically long to converge.

4.2. Global Shape Matching

Instead of iterated pairwise registrations, several techniques for the simultaneous registration of multiple point sets have been proposed, see [KLMV05] and the references therein. Most of these methods factorize dense matrices, whose dimensions are proportional to the number of objects to be matched. While this is not critical when matching < 100 objects, in our setting a large number of prisms would lead to prohibitively complex matrices.

In contrast, Pottmann et al. [PLH02] propose an iterative simultaneous registration which involves solving *sparse* linear systems only, and hence can be adapted to our problem. Their technique corresponds to a Newton-type minimization of the registration error: In each iteration a linear system is solved for a descent direction, which corresponds to an *affine* motion per prism. A projection of those onto the manifold of rigid motions results in a *rigid* update for each prism. This process is iterated until convergence.

The descent direction of the Newton-type iteration requires first-order approximations $\mathbf{A}_i$ of rigid motions $(\mathbf{R}_i, \mathbf{t}_i)$, which can be formulated in terms of linear and angular velocities $\mathbf{v}_i$ and ω_i:

$$\mathbf{R}_i(\cdot) + \mathbf{t}_i \;\approx\; (\cdot) + \omega_i \times (\cdot) + \mathbf{v}_i \;=:\; \mathbf{A}_i(\cdot) \;. \tag{5}$$

Reformulating the energy minimization (3) in terms of these first-order approximations yields

$$\min_{\{\mathbf{v}_i,\omega_i\}} \sum_{\{i,j\}} w_{ij} \int_{[0,1]^2} \left\| \mathbf{A}_i\left(\mathbf{f}^{i\to j}(\mathbf{u})\right) - \mathbf{A}_j\left(\mathbf{f}^{j\to i}(\mathbf{u})\right) \right\|^2 d\mathbf{u} \;. \tag{6}$$

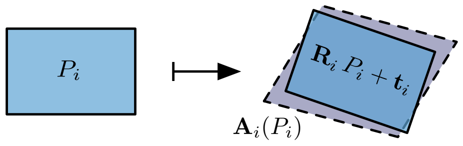
As all integrals can again by evaluated analytically, (6) represents a standard quadratic minimization in the linear and angular velocities, the optimal values for which can be found by solving a sparse linear system [PLH02].

The resulting optimal velocities $(\mathbf{v}_i, \omega_i)$ correspond to the Newton descent direction and represent first-order approximations $\mathbf{A}_i$. Since those are *affine* transformations, they have to be projected back onto the manifold of rigid motions before applying them to the prisms P_i. For this step, [PLH02] propose to choose $(\mathbf{R}_i, \mathbf{t}_i)$ as the helical motion associated with $(\mathbf{v}_i, \omega_i)$. However, this method turned out to be restricted to very small update steps, which is sufficient for registering pre-aligned point sets, but in our case leads to impractically slow convergence for large deformations.

We therefore propose to project $\mathbf{A}_i$ by finding the "closest" rigid motion $(\mathbf{R}_i, \mathbf{t}_i)$, where we measure distances of transformations by comparing their effects on the prism P_i. We find the closest rigid motion by minimizing

$$\min_{\mathbf{R}_i,\mathbf{t}_i} \int_{[0,1]^2} \left\| \mathbf{R}_i \mathbf{f}^{i\to j}(\mathbf{u}) + \mathbf{t}_i \;-\; \mathbf{A}_i\left(\mathbf{f}^{i\to j}(\mathbf{u})\right) \right\|^2 d\mathbf{u} \;, \tag{7}$$

which yields another local shape matching problem, as depicted in the following figure.



This geometrically intuitive projection operator allows for much larger update steps compared to the helical motions of [PLH02], which reduces the number of required Newton-type iterations by a factor of about 50 in all our examples. Although our projection is computationally more involved, its costs are still small compared to solving (6). Hence, the overall performance increases by roughly the same factor.

Finally, the Newton-like descent direction has to be scaled by a suitable step size λ. We thus derive the rigid motions $(\mathbf{R}_i, \mathbf{t}_i)$ by projecting *scaled* velocities $(\lambda\mathbf{v}_i, \lambda\omega_i)$ instead, where we simply start with $\lambda = 1$ and subsequently halve λ until the new rigid motions are found to decrease the energy (3). Although more elaborate methods exist [PHYH04], this simple technique turned out to be sufficient.

The computational complexity of the non-linear optimization is dominated by factorizing and solving the linear system corresponding to the minimization of (6) in each iteration. Since the matrix is sparse (about 16 non-zeros/row on average), symmetric, and positive definite, an efficient sparse Cholesky solver can be used [TCR03]. We can additionally exploit that the non-zero structure of the matrix stays fixed throughout all iterations. This allows us to precompute the symbolic part of the factorization [BBK05], which saves about 40% of the total time per iteration.

Combining the matching-based projection (7) with the symbolic pre-factorization reduces computation time by two orders of magnitude compared to [PLH02]. However, the optimization still achieves only 6500 prism updates per second on a 3.2GHz P4, which is not sufficient for interactive deformations of complex meshes. Given these limitations, neither the local nor the global shape matching yields a practically useful minimization technique by itself. But combining their respective strengths allows us to derive an efficient hierarchical method, as we will show in the next section.

4.3. Hierarchical Shape Matching

To maximize computational efficiency, we perform the shape matching on a multigrid hierarchy. For multigrid methods on irregular triangle meshes the successively coarser levels are built by mesh decimation [AKS05]. However, our framework does not require the hierarchy levels to represent consistent triangulations, since prisms can be generated from arbitrary polygons. This enables us to conveniently build the hierarchies levels (typically about 4) by successive clustering of neighboring faces and combining their corresponding prisms by considering them as one single rigid group.

A common practice for hierarchical multigrid solvers of linear systems [AKS05] is to use a *direct* solver on a coarse hierarchy level to obtain a low frequency approximation of the solution, which then is successively refined on higher levels using *iterative* techniques. Similarly, we start by applying the *global* shape matching on the coarsest hierarchy level in order to efficiently compute the low frequencies of the deformation. Since even for detailed surface meshes the shape deformations generally are smooth (low frequency) functions, this initial approximation typically is already very close to the exact solution. Since the *local* shape matching corresponds to an iterative error diffusion, we apply a few iterations (typically 2) on each finer hierarchy level, which rapidly smooths out the remaining high frequency errors.

Since we do not require consistent triangulations or sophisticated multigrid pre-conditioning, our hierarchical solver is considerably easier to implement compared to traditional multigrid techniques. The efficient combination of global and local shape matching yields a robust hierarchical non-linear optimization, which provides shape deformations of moderately complex models at interactive rates. Even our two most complex models, the 100k triangle Dragon of Figure 8 and the 180k triangle Goblin of Figure 1, can be edited interactively at one frame/sec (see the accompanying video).

4.4. Robustness

One of the main advantages of our method — and the main difference to existing shell-based techniques — is that during a deformation the shape quality of prisms will not degrade, since the individual prisms are kept rigid, which guarantees numerical robustness even for extreme deformations.

Even the initial shape quality of the prisms — which depends on the input mesh — only has a minor influence on the robustness of our method. The local and global shape matching techniques only fail for prisms that degenerate to a single line, which requires their corresponding triangles to degenerate to single points. However, the more likely cases of needle triangles (one extremely short edge) or caps triangles (one large angle) do not cause numerical problems, as long as stable normals can be computed for the prism extrusion. This allows us to process even meshes of low initial quality, which would be very likely to cause problems for classical FEM simulations.

A thorough convergence analysis of the Newton-like global shape matching can be found in [PHYH04]. In all our experiments the global matching converged robustly, with even extreme deformations requiring < 10 iterations. In theory, the minimization cannot be guaranteed to find the global minimum. Extreme user constraints that enforce the surface to form self-intersections might steer the iteration into a local minimum. However, as soon as the constraints are relaxed again, the optimization typically recovers, which is shown in Figure 6 and the accompanying video.

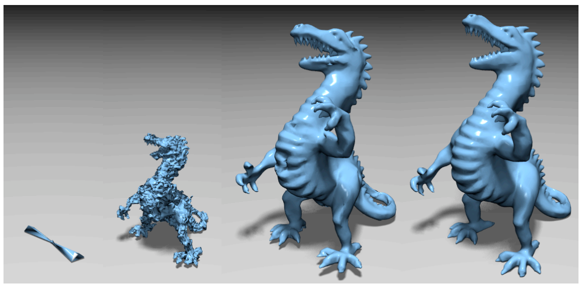


Figure 6: *The robust global optimization is able to fully recover the dragon model after fixing two prisms on the feet, collapsing all other prisms into one point, and randomly perturbing their orientations* (left). *The images show results after 1, 10, and 25 iterations of the global matching procedure.*

5. Results

In this section we show the flexibility of our prism-based modeling framework on a range of examples, including complex shape deformations and general surface processing.

In addition to robustness, our prism formulation also provides interesting, geometrically intuitive parameters for controlling the surface behavior. The rest state of the optimization can be adjusted by explicitly changing the prism shapes. Figure 5 already showed how surface stiffness can be specified in terms of prism heights. In addition to that, adjusting the prisms' widths allows to locally increase or decrease surface area. In the left image of Figure 7 the dragon model is T-Rex'ed by shrinking its arms and super-sizing its head.

Besides height and width, the prisms' deviation from orthogonality yields another interesting parameter. When prisms are generated by extrusion along vertex normals (as described in Section 3), the initial configuration is the rest state of the optimization. In contrast, extruding orthogonal prisms along face normals leads to a non-vanishing initial energy, which tries to achieve a locally planar state. Interpolating the extrusion directions between vertex normals and face normals therefore blends between a thin shell and thin plate behavior.

The latter tries to locally decrease curvature, which smooths the surface. However, since the size of prisms is kept fixed, the surface area is preserved, which avoids the typical shrinkage of Laplacian smoothing (cf. Figure 7, center). Moreover, extrapolating the face normals across vertex normals locally increases curvature and thus can be used for surface detail enhancement (cf. Figure 7, right).

Having these geometrically intuitive surface parameters at hand, the user can deform surfaces by simply selecting handle regions and moving them to their desired position. The respective transformations of the underlying prisms are automatically derived from these face constraints. Figure 8 shows a large-scale deformation of a complex dragon model

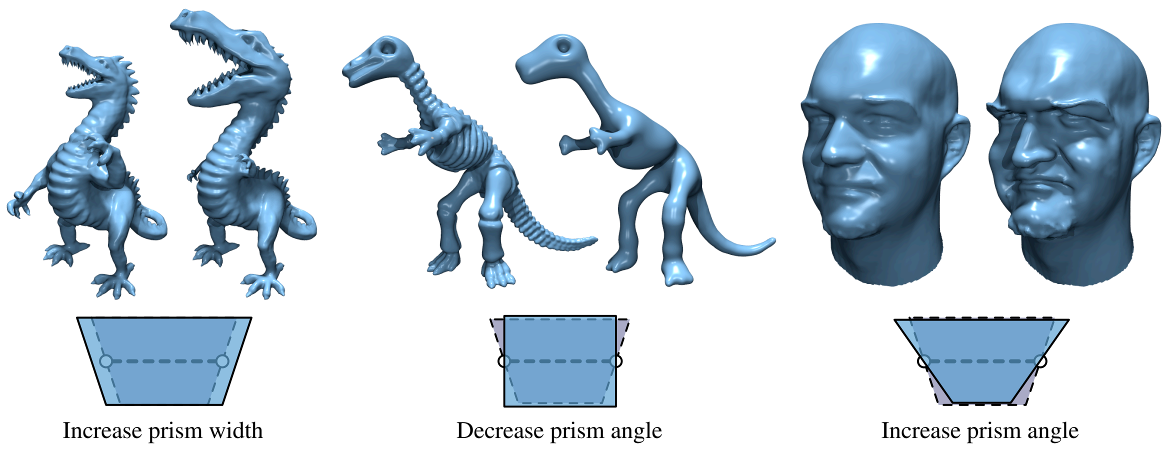


Figure 7: *Changing the prism shapes provides geometrically intuitive parameters for controlling the surface behavior. Adjusting the width of prisms can be used to locally shrink or enlarge surface area, which was done to convert the dragon to a T-Rex* (left). *Changing the prism's deviation from orthogonality blends between thin-shell and thin-plate behavior, which allows for non-shrinking smoothing* (center). *Increasing the prism angles instead amplifies surface curvature, and therefore enhances local surface details* (right).

and compares the result to the linear methods discussed in Section 2. While all linear methods fail to produce the desired result, our non-linear surface model deforms naturally, which can also be observed in the accompanying video.

Instead of fully constraining a prism's position and orientation, both the local and global shape matching formulations also allow to freeze either of them separately, such that the other term is free to be optimized. This enables a simple click & drag metaphor, where the user constrains the position of a dragged surface point, while its orientation is automatically optimized. This kind of interface would not be possible with methods based on differential coordinates, which require both rotation and translation constraints.

In addition to controlling surface deformations by enforcing hard constraints, our method also supports user-specified forces acting on the model. The squared point distances in Equation (3) can be interpreted as energies of zero-length springs, such that the shape matching solves for the steady-state of a (mass-less) spring system. User-defined spring forces can therefore be incorporated by adding surface points and corresponding target positions to the shape matching system. Depending on the application, this force-based modeling metaphor might provide physically more intuitive results, since enforcing hard constraints would correspond to extremely high forces.

The force-based metaphor, in combination with local stiffness control, was used to pose the Goblin model shown in Figure 1 in less than 5 minutes. Another example is shown in Figure 9, where a user-defined force pulls the Beetle's front upwards, performed for both a rather stiff and a more flexible surface material.

The limitation of our method is its computational performance, which restricts the global shape matching to about 10k prisms for interactive modeling. However, our hierarchical optimization provides interactive response rates even for complex meshes by performing the global optimization on a coarser level, for which 10k triangles are sufficient, since global shape deformations typically correspond to smooth functions.

6. Conclusion

We presented a non-linear surface deformation model based on elastically coupled rigid prisms, which allows for intuitive and physically plausible geometric modeling. In the past, non-linear techniques were rarely considered for interactive modeling applications because of their seemingly prohibitive computation costs, complicated implementation, and notorious numerical instabilities. In contrast, our new method combines ease of implementation and extreme robustness, while still achieving interactive rates for moderately complex models.

One promising direction for future work would be the application of our prism-based framework to physically inspired dynamic simulations, since in this context numerical robustness is also of major importance. The generalization of our global shape matching framework from thin shell surfaces to fully volumetric objects would also be an interesting extension.

Acknowledgments The authors are grateful to Jan Möbius for implementing [LSLCO05]. The Dinosaur and Head models (Figure 7) are courtesy of Cyberware, the Dragon model (Figure 8) is from the Stanford scanning repository.

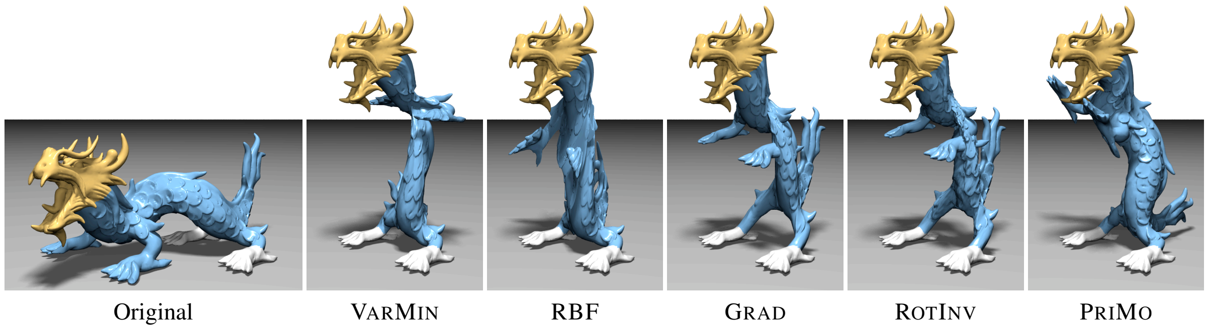


Figure 8: *The crouching dragon was lifted by fixing its hind feet and moving its head to the target position in a single step. Similar to Figure 2 the linear deformation methods yield counter-intuitive results, which even contain severe self-intersections. In contrast, our* PRIMO *technique leads to a very natural deformation.*

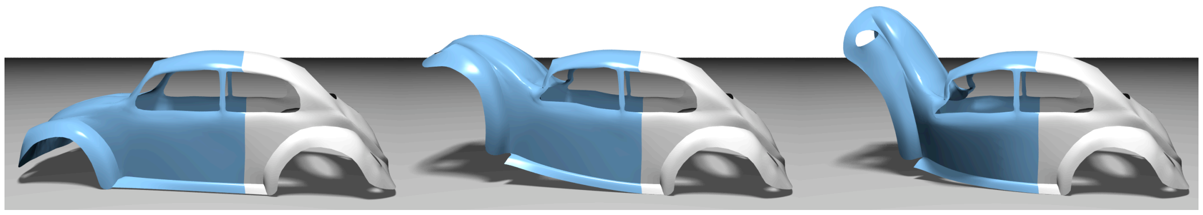


Figure 9: *In addition to hard constraints, our framework can also incorporate user-defined forces. In this example a force tries to lift the car's front, and center and right image show results for the same force on a rather stiff and a more flexible surface material, respectively.*

References

[AKS05] AKSOYLU B., KHODAKOVSKY A., SCHRÖDER P.: Multilevel Solvers for Unstructured Surface Meshes. *SIAM Journal on Scientific Computing 26*, 4 (2005). 6, 7

[Bat95] BATHE K.-J.: *Finite Element Procedures*. Prentice Hall, 1995. 4

[BBK05] BOTSCH M., BOMMES D., KOBBELT L.: Efficient linear system solvers for geometry processing. In *11th IMA conference on the Mathematics of Surfaces* (2005). 6

[BK03] BOTSCH M., KOBBELT L.: Multiresolution surface representation based on displacement volumes. In *Proc. of Eurographics 03* (2003). 2

[BK04] BOTSCH M., KOBBELT L.: An intuitive framework for real-time freeform modeling. In *Proc. of ACM SIGGRAPH 04* (2004). 1, 2, 3

[BK05] BOTSCH M., KOBBELT L.: Real-time shape editing using radial basis functions. In *Proc. of Eurographics 05* (2005). 1, 2, 3

[CG91] CELNIKER G., GOSSARD D.: Deformable curve and surface finite-elements for free-form shape design. In *Proc. of ACM SIGGRAPH 91* (1991). 1, 2

[GHDS03] GRINSPUN E., HIRANI A. N., DESBRUN M., SCHRÖDER P.: Discrete shells. In *Proc. of ACM SIGGRAPH/Eurographics symposium on Computer animation (SCA) '03* (2003). 3, 4

[GSS99] GUSKOV I., SWELDENS W., SCHRÖDER P.: Multiresolution signal processing for meshes. In *Proc. of ACM SIGGRAPH 99* (1999). 2

[Hor87] HORN B. K. P.: Closed-form solution of absolute orientation using unit quaternions. *Journal of the Optical Society of America 4*, 4 (1987). 5, 10

[ITF04] IRVING G., TERAN J., FEDKIW R.: Invertible finite elements for robust simulation of large deformation. In *Proc. of ACM SIGGRAPH/Eurographics symposium on Computer animation (SCA) '04* (2004). 4

[KCVS98] KOBBELT L., CAMPAGNA S., VORSATZ J., SEIDEL H.-P.: Interactive multi-resolution modeling on arbitrary meshes. In *Proc. of ACM SIGGRAPH 98* (1998). 2

[KLMV05] KRISHNAN S., LEE P. Y., MOORE J. B., VENKATASUBRAMANIAN S.: Global registration of multiple 3D point sets via optimization-on-a-manifold. In *Proc. of Eurographics symposium on Geometry Processing 05* (2005). 6

[KVS99] KOBBELT L., VORSATZ J., SEIDEL H.-P.: Multiresolution hierarchies on unstructured triangle meshes. *Comput. Geom. Theory Appl. 14*, 1-3 (1999). 2

[LSCO*04] LIPMAN Y., SORKINE O., COHEN-OR D., LEVIN D., RÖSSL C., SEIDEL H.-P.: Differential coordinates for interactive mesh editing. In *Proc. of Shape Modeling International 04* (2004). 2

[LSLCO05] LIPMAN Y., SORKINE O., LEVIN D., COHEN-OR D.: Linear rotation-invariant coordinates for meshes. In *Proc. of*

ACM SIGGRAPH 05 (2005). 1, 2, 3, 8

[MHTG05] MÜLLER M., HEIDELBERGER B., TESCHNER M., GROSS M.: Meshless deformations based on shape matching. In *Proc. of ACM SIGGRAPH 05* (2005). 5

[MS92] MORETON H. P., SÉQUIN C. H.: Functional optimization for fair surface design. In *Proc. of ACM SIGGRAPH 92* (1992). 2

[NSACO05] NEALEN A., SORKINE O., ALEXA M., COHEN-OR D.: A sketch-based interface for detail-preserving mesh editing. In *Proc. of ACM SIGGRAPH 05* (2005). 3

[PHYH04] POTTMANN H., HUANG Q.-X., YANG Y.-L., HU S.-M.: *Geometry and convergence analysis of algorithms for registration of 3D shapes*. Tech. Rep. 117, Vienne University of Technology, 2004. 6, 7

[PLH02] POTTMANN H., LEOPOLDSEDER S., HOFER M.: Simultaneous registration of multiple views of a 3D object. *Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences 34*, 3A (2002). 1, 6, 10

[SCOL*04] SORKINE O., COHEN-OR D., LIPMAN Y., ALEXA M., RÖSSL C., SEIDEL H.-P.: Laplacian surface editing. In *Proc. of Eurographics symposium on Geometry Processing 04* (2004). 1, 2

[SF98] SINGH K., FIUME E.: Wires: A geometric deformation technique. In *Proc. of ACM SIGGRAPH 98* (1998). 2

[SK04] SHEFFER A., KRAEVOY V.: Pyramid coordinates for morphing and deformation. In *Proc. of Symp. on 3D Data Processing, Visualization and Transmission (3DPVT) '04* (2004). 3

[SP86] SEDERBERG T. W., PARRY S. R.: Free-form deformation of solid geometric models. In *Proc. of ACM SIGGRAPH 86* (1986). 2

[SP04] SUMNER R. W., POPOVIĆ J.: Deformation transfer for triangle meshes. In *Proc. of ACM SIGGRAPH 04* (2004). 2

[SZGP05] SUMNER R. W., ZWICKER M., GOTSMAN C., POPOVIĆ J.: Mesh-based inverse kinematics. In *Proc. of ACM SIGGRAPH 05* (2005). 2

[TCR03] TOLEDO S., CHEN D., ROTKIN V.: Taucs: A library of sparse linear solvers. http://www.tau.ac.il/∼stoledo/taucs, 2003. 6

[TPBF87] TERZOPOULOS D., PLATT J., BARR A., FLEISCHER K.: Elastically deformable models. In *Proc. of ACM SIGGRAPH 87* (1987). 1, 2

[WSG05] WICKE M., STEINEMANN D., GROSS M.: Efficient animation of point-sampled thin shells. In *Proc. of Eurographics 05* (2005). 3

[WW92] WELCH W., WITKIN A.: Variational surface modeling. In *Proc. of ACM SIGGRAPH 92* (1992). 1, 2

[YZX*04] YU Y., ZHOU K., XU D., SHI X., BAO H., GUO B., SHUM H.-Y.: Mesh editing with Poisson-based gradient field manipulation. In *Proc. of ACM SIGGRAPH 04* (2004). 2

[ZHS*05] ZHOU K., HUANG J., SNYDER J., LIU X., BAO H., GUO B., SHUM H.-Y.: Large mesh deformation using the volumetric graph Laplacian. In *Proc. of ACM SIGGRAPH 05* (2005). 3

[ZRKS05] ZAYER R., RÖSSL C., KARNI Z., SEIDEL H.-P.: Harmonic guidance for surface deformation. In *Proc. of Eurographics 05* (2005). 2, 3

Appendix A: Continuous Face-Based Shape Matching

We show how to extend the local and global shape matching approaches of [Hor87] and [PLH02] from discrete point correspondences to continuous face correspondences.

Suppose we are given two functions $a(\mathbf{u})$ and $b(\mathbf{u})$ defined by bi-linear interpolation of four values $\{a_{00}, a_{10}, a_{01}, a_{11}\}$ and $\{b_{00}, b_{10}, b_{01}, b_{11}\}$, respectively. Then their L_2 inner product simplifies to a weighted sum of 16 combinations of corner values:

$$\begin{aligned}\int_{[0,1]^2} a(\mathbf{u}) \cdot b(\mathbf{u})\, d\mathbf{u} &= \frac{1}{9} \sum_{i,j,k,l=0}^{1} a_{ij} \cdot b_{kl} \cdot 2^{(-|i-k|-|j-l|)} \\ &=: \langle a, b \rangle_2 \ .\end{aligned}$$

Using this, the continuous pairwise energy of Equation (1) evaluates to

$$E_{ij} = \left\langle \mathbf{f}^{i \to j} - \mathbf{f}^{j \to i}, \mathbf{f}^{i \to j} - \mathbf{f}^{j \to i} \right\rangle_2 \ .$$

In order to generalize the local shape matching of [Hor87], we first compute the weighted centroids $\mathbf{c}^i$ and $\mathbf{c}^*$ of the two face sets to be aligned, which leads to

$$\left.\begin{matrix}\mathbf{c}^i \\ \mathbf{c}^*\end{matrix}\right\} = \frac{1}{\sum_{j \in \mathcal{N}_i} w_{ij}} \sum_{j \in \mathcal{N}_i} \frac{w_{ij}}{4} \sum_{k,l=0}^{1} \left\{\begin{matrix}\mathbf{f}_{k,l}^{i \to j} \\ \mathbf{f}_{k,l}^{j \to i}\end{matrix}\right. \ .$$

To derive the optimal rotation $\mathbf{R}_i$ according to [Hor87], we build the matrix $\mathbf{N} =$

$$\begin{pmatrix} S_{xx}+S_{yy}+S_{zz} & S_{yz}-S_{zy} & S_{zx}-S_{xz} & S_{xy}-S_{yx} \\ S_{yz}-S_{zy} & S_{xx}-S_{yy}-S_{zz} & S_{xy}+S_{yx} & S_{zx}+S_{xz} \\ S_{zx}-S_{xz} & S_{xy}+S_{yx} & -S_{xx}+S_{yy}-S_{zz} & S_{yz}+S_{zy} \\ S_{xy}-S_{yx} & S_{zx}+S_{xz} & S_{yz}+S_{zy} & -S_{xx}-S_{yy}+S_{zz} \end{pmatrix}$$

from the component-wise L_2 inner products

$$\begin{aligned} S_{xx} &= \sum_{j \in \mathcal{N}_i} w_{ij} \left\langle \left(\mathbf{f}^{i \to j} - \mathbf{c}^i\right)_x, \left(\mathbf{f}^{j \to i} - \mathbf{c}^*\right)_x \right\rangle_2 \ , \\ S_{xy} &= \sum_{j \in \mathcal{N}_i} w_{ij} \left\langle \left(\mathbf{f}^{i \to j} - \mathbf{c}^i\right)_x, \left(\mathbf{f}^{j \to i} - \mathbf{c}^*\right)_y \right\rangle_2 \ , \end{aligned}$$

and analogously for the other components. The eigenvector corresponding to the largest eigenvalue of $\mathbf{N}$ gives the optimal rotation $\mathbf{R}_i$ when interpreted as a unit quaternion. The optimal translation finally is $\mathbf{t}_i = \mathbf{c}^* - \mathbf{R}_i \mathbf{c}^i$.

For the generalization of the global shape matching approach of [PLH02] we have to adjust the linear system corresponding to the minimization of Equation (6). Assume two corresponding points $\mathbf{p}^{i \to j}$ and $\mathbf{p}^{j \to i}$ sampled from neighboring faces $\mathbf{f}^{i \to j}(\mathbf{u})$ and $\mathbf{f}^{j \to i}(\mathbf{u})$. Their contribution to the global energy is

$$w_{ij} \left\| \left(\mathbf{p}^{i \to j} + \omega_i \times \mathbf{p}^{i \to j} + \mathbf{v}_i\right) - \left(\mathbf{p}^{j \to i} + \omega_j \times \mathbf{p}^{j \to i} + \mathbf{v}_j\right) \right\|^2 \ ,$$

which gives four 6×6 matrix blocks. From those the global block structure of the matrix and the numeric values are easily derived. The continuous formulation then only requires to replace the involved products of the form $(\mathbf{p}^{i \to j})_x \cdot (\mathbf{p}^{j \to i})_y$ by the inner products $\left\langle \left(\mathbf{f}^{i \to j}\right)_x, \left(\mathbf{f}^{j \to i}\right)_y \right\rangle_2$ in the respective matrix entries.