Shape Optimization for Human-Centric Products with Standardized Components

Tsz-Ho Kwok, Charlie C.L. Wang*

Department of Mechanical and Automation Engineering
The Chinese University of Hong Kong

Corresponding author. Tel: (+852) 3943 8052. Fax: (+852) 2603 6002. Email: cwang@mae.cuhk.edu.hk

Abstract

In this paper, we present an optimization framework for automating the shape customization of human-centric products, which can be mounted on or embedded in human body (such as exoskeletal devices and implants). This kind of products needs to be customized to fit the body shapes of users. At present, the design customization for freeform objects is often taken in an interactive manner that is inefficient. We investigate a method to automate the procedure of customization. Major difficulty in solving this problem is caused by the not freely changed shape of components. They should be selected from a series of standardized shapes. Different from the existing approaches that allow fabricating all components by customized production, we develop a new method to generate customized products by using as-many-as-possible standardized components. Our work is based on a mixed-integer shape optimization framework.

Keywords: human-centric, product design, shape optimization, mixed-integer, standardized components

1. Introduction

Human-centric products start to be widely used in many medical and sport applications, and the shape of these products must be adaptive to the shape of human bodies. Three-dimensional (3D) shape of human bodies can be obtained by 3D scanners. After establishing the correspondences among different human bodies [1, 2], the product originally designed for a specific body can be warped to a shape to fit another body [3]. In this way, the product can be reused and customized for another person. It is good enough for transferring the design of some products such as clothes, shoes, and furniture as all components in these products can be fabricated in a low cost. However, it doesn't apply to electronic or mechanical products. Some of their components, e.g., nuts, chipsets, or joints, cannot be freely varied. Instead, these components are usually fabricated in a standard way – i.e., each of them has a series of variations that are manufactured by mass production. These components are called *standardized components*, and the series of variations is called *element library* in the rest of this paper. Note that, as fabricated by mass production, shape variations of these components are in a stepwise manner. For example, the wheels of bicycles as standardized components can only be either 24 or 26 inch but can never be 24.3 inch. The classification of components into standardized series is mainly based on the procedure of manufacturing, which is beyond the scope of this paper. At present, the customization of products with standardized components is often conducted by designers in an interactive manner. The shape optimization step becomes the bottleneck of designing human-centric products. The major challenge comes from the lack of an effective shape optimization method that allows a component to be varied in a stepwise manner. The problem becomes more challenging when the shape variation is driven by human bodies and the product's shape is required to well fit the human body's shape. In this paper, we investigate techniques to automate the step of customization.

A *Mixed-Integer As-Rigid-As-Possible* (MI-ARAP) shape optimization framework is developed in this paper. The shape optimization is formulated as an as-rigid-as-possible deformation problem that is computed on volumetric meshes. During the optimization, some components are restricted to be one of the pre-defined shapes in the element library, which is actually a variational formulation with integer variables. It means that the component is not allowed to deform continuously from one shape to another shape. To keep the spatial relationship between a product and human bodies, we may use continuous deformation methods [4, 3] to warp the shape of a product designed for one human body into a shape fit another human body. This shape obtained by continuous deformation is called the *initial shape*, which is the input of our optimization. After obtaining the *initial shape* by warping a product from one human body to another one, the standardized components are warped into a non-standard shape (e.g., see the middle of Fig.1). These distorted components will be replaced by the most suitable shapes in the element libraries. Unfortunately, if all the standardized components are simply replaced at the same time, the nearby components could become incompatible. An iterative rounding algorithm is investigated in this paper to solve this problem. An example to illustrate the function of our approach is shown in Fig.1. As-many-as-possible standardized components are used during the shape optimization, so that an economical solution is developed to realize the customization. This is because the cost of fabricating standardized components by mass produc-

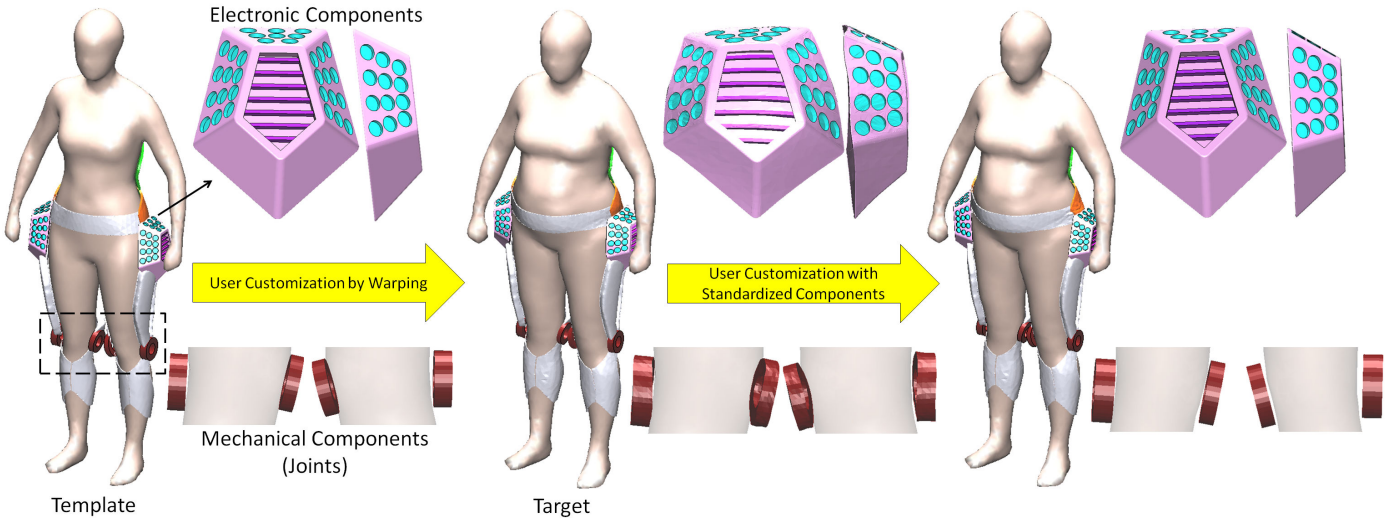

Figure 1: An illustration of the proposed human-centric design automation with standardized components on an example of exoskeleton customization, where the red joints (highlighted by dashed lines) and the pink blocks are standardized components. (Left) A given design is defined on the template human body model H. (Middle) The design is warped from the template H to fit a target human body model H' – it is easy to find that the standardized components are also warped to fit the body shape. (Right) The proposed MI-ARAP shape optimization framework enforces the shape of standardized components – see the red joints and the pink blocks.

tion is much lower than the cost of customized production for non-standard shapes.

1.1. Related Works

The related techniques of deformation, structure preserved shape processing and mixed-integer optimization are reviewed as follows.

Deformation techniques

Tremendous efforts have been made in the CAD and graphics communities to develop user-friendly deformation tools for the computation of 3D natural deformation. Generally, the existing methods can be classified into surface-based or volume-based approaches (details can be found in the book of Botsch et al. [5]). Surface-based methods [6, 7, 8, 9, 10, 11, 12, 13] usually compute the displacement functions attached on the original surface S to transfer the original surface into a new shape S'. A high degree of control can be provided by these methods. On the other hand, the robustness and efficiency of these methods are strongly affected by the mesh complexity and the quality of triangles on S. In a large literature of surface-based deformation methods, the recent methods based on the as-rigid-as-possible (ARAP) strategy (ref. [13, 14]) try to reduce the total elasticity of all triangular elements, which can generate very natural deformation results when bending and twisting elastic objects. The main problem of these methods is that the ARAP energy is only evaluated on the surface, which does not intrinsically preserve the volume of the original model. Volume-based methods [15, 16, 17] can solve these problems quite well. In our shape optimization framework, the volume-based ARAP energy is employed to evaluate the elasticity of components. However, none of prior volume-based deformation approaches consider the problem of deformation with stepwise shapes.

All above approaches focus on the shape manipulation of surface S itself. To reuse the designs on new models, we need a deformation method that is able to transform the objects located around a human body H to the space around another human body H'. By using the freeform deformation technique [4, 18], triangles on the surface of H can be used as handles to deform models around H to a new shape around H' if the mapping from H to H' is available [19, 20, 2, 1, 21]. Similar shape transformations can also be realized by building cages around the surfaces of H and H' [22, 23] or by formulating a spatial warping function [3, 24]. The recent development focuses on how to preserve the shape preference of the original design around H when transforming it to the product around H' (ref. [25, 26]). The shape transformation approach developed by Brouet et al. [26] is based on the idea of ARAP surface modeling [13, 14] and the deformation transfer approaches [27, 28, 29, 30]. However, the ARAP formulation is nonlinear, Sorkine et al. [31] and Igarashi et al. [13] designed a two-step editing process for this optimization. Similarly, Schaefer et al. [32] implemented a moving least squares framework for 2D space warping. Later, this so-called local/global approaches [14, 33] have been extended to work on volumetric meshes [34, 35]. Specifically, in a local phase, the transformation matrix of each element is first evaluated and manipulated to indicate the deformation preferences. This is followed by a global phase to ensure that the transformations applied to the elements around each vertex are compatible, i.e., corresponding vertices in different elements are in consistent positions. A recent study [36] shows that such a local/global computation converges very fast. Our shape optimization framework will adopt this strategy of local/global computation.

Structure-aware shape processing

The recent development of shape processing techniques in the

community of geometric modeling extends the concept of parametric modeling in CAD/CAM systems to objects with complex geometry. As the input models of these approaches are piecewise linear surfaces without any structural information, two phases are conducted: 1) shape analysis to find out structural constraints and 2) shape manipulation that preserves the constraints.

Following the idea of feature-preserving in image warping, Kraevoy et al. [37] proposed a method to resize geometric models, meanwhile preserve the structures by computing a non-homogeneous volumetric mapping on grids. A more flexible method is presented in the work of Gal et al. [38] to preserve structures on man-made models by using a few special 1D wires. This work is extended by Zheng et al. [39] to manipulate the shape of models by their constituent components. As a result, users have easier control over the shape of models by using component-wise controllers. Another recent work presented by Bokeloh et al. [40] can detect continuous and discrete regular patterns. These patterns are preserved in a variational deformation framework by integrating a discrete algorithm that adaptively inserts or removes repeated elements in regular patterns. Our method for computing customized products will integrate discrete algorithms into variational optimization framework to progressively preserve the shape of standardized components. Structure preservation is also formulated as a constrained modification in image-based modeling of architectures [41]. The recent work in image-based modeling [42] tries to solve such constrained editing problems by first determining a minimal set of vertices to be updated and then computing actual position updates to satisfy all constraints. However, the above structure-aware shape processing approaches consider neither the shape adaptation on human bodies nor the constraints of standardized components.

Mixed-integer optimization
Solving a mixed-integer optimization problem that has both continuous and discrete unknowns is NP-hard [43]. Different from optimization problems only have continuous unknowns, the computational domain of mixed-integer optimization is formed by many isolated sub-regions, and the discrete unknowns can only be changed in a jump from one sub-region to another one. As a result, solving such problems may need to test all discrete possibilities. However, this is not practical when the number of discrete unknowns is large. A more practical scheme was introduced by Bommes et al. [44]. Their approach adopts a greedy strategy to first solve the relaxed problems by treating the discrete unknowns as continuous variables, and then projects the solution by rounding the discrete unknown with smallest rounding error into integer incrementally. The nature of using standardized components in the customization of products is similar to the mixed-integer optimization - the variation of standardized components can only jump from one predefined shape to another shape. In other words, the optimization domain is also discretized. An iterative greedy rounding scheme is investigated in our shape optimization framework.

1.2. Main Results

The contributions of research work presented in this paper are twofold.

1. *Mixed-integer shape optimization*:
   The shape optimization of products is formulated as an as-rigid-as-possible deformation problem that is computed on volumetric meshes. Specifically, some sub-domains of a deformed object are restricted to a series of pre-defined shapes, which is a variational formulation with integer variables. An iterative rounding algorithm is developed to solve this mixed-integer optimization problem.
2. *Design transfer with standardized components*:
   After obtaining the initial shape of a product with assembled components based on the mapping between spaces around human bodies, the components are replaced by standardized components progressively. As-many-as-possible standardized components are used meanwhile the elastic energy is minimized for other components. Shapes of the standardized components are selected from their corresponding element libraries, and the best-fit shapes are found automatically.

To the best of our knowledge, this is the first approach in literature to allow using standardized components in the customization of freeform objects for fitting human bodies.

The rest of the paper is organized as follows. The basic MI-ARAP framework is first introduced in section 2, and it is further extended for standardized components in section 3. After that, the formulation of adding hard constraints into the optimization is presented in section 4. In section 5, a flexible encoding is described to formulate the relationship between assembled components. Finally, different examples are shown in section 6, and our paper ends with the conclusion section.

2. Mixed-Integer As-Rigid-As-Possible Shape Optimization Framework

Shape optimization is formulated under an ARAP framework with mixed-integer variables. Specifically, the standardized components can only jump from a pre-defined shape to another one rather than a continuous deformation in $\mathbb{E}^3$. Without loss of generality, we assume that the components to be deformed have been tessellated into tetrahedral meshes. Four vertices of a tetrahedron t have their rest positions $\{\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3, \mathbf{p}_4\}$ before deformation and the current positions $\{\tilde{\mathbf{p}}_1, \tilde{\mathbf{p}}_2, \tilde{\mathbf{p}}_3, \tilde{\mathbf{p}}_4\}$ in the deformation. The current positions at the beginning of optimization can be obtained by spatial warping functions [3], and the current positions in the later steps of iteration will be updated during the optimization. Given the current positions and the rest positions of t, a linear transformation between the rest and the current shapes can be defined by a transformation matrix $\mathbf{T}$ and a displacement vector $\mathbf{d}$ as

$$\mathbf{T}\mathbf{p}_i + \mathbf{d} = \tilde{\mathbf{p}}_i, \quad i = 1, 2, \cdots, 4. \tag{1}$$

$\mathbf{d}$ can be eliminated by subtracting the last equation from the others, such that the formulation is invariant to translation. Then, we can have $\mathbf{TP} = \tilde{\mathbf{P}}$ with

$$\begin{aligned}\mathbf{P} &= [\mathbf{p}_1 - \mathbf{p}_4 \;\; \mathbf{p}_2 - \mathbf{p}_4 \;\; \mathbf{p}_3 - \mathbf{p}_4] \\ \tilde{\mathbf{P}} &= [\tilde{\mathbf{p}}_1 - \tilde{\mathbf{p}}_4 \;\; \tilde{\mathbf{p}}_2 - \tilde{\mathbf{p}}_4 \;\; \tilde{\mathbf{p}}_3 - \tilde{\mathbf{p}}_4]\end{aligned}.$$

As a result, the transformation matrix $\mathbf{T}$ can be obtained by $\mathbf{T} = \tilde{\mathbf{P}}\mathbf{P}^{-1}$, which includes both the scaling and the rotation. In the ARAP formulation, it will be enforced to a rigid-body transformation only has rotation. According to the discussion in the paper of Liu et al. [33], the scaling matrix and the rotational matrix can be decoupled by the "signed version" of *singular value decomposition* (SVD) into $\mathbf{T} = \mathbf{U}\Sigma\mathbf{V}^T$. Due to the reason that $\mathbf{T}$ is a square matrix with positive determinant, $\mathbf{U}$ and $\mathbf{V}$ are the rotational matrices, and Σ is a scaling matrix written as $\Sigma = diag\{\sigma_1, \sigma_2, \sigma_3\}$. σ_1, σ_2 and σ_3 are the scaling factors of $\mathbf{T}$ in three orthogonal directions. Then, a pure rotational matrix can be derived from $\mathbf{T}$ as

$$\mathbf{L} = \mathbf{U}\mathbf{V}^T. \tag{2}$$

The optimization targets on minimizing the difference between $\mathbf{T}$ and $\mathbf{L}$. Therefore, the optimization energy that is going to be minimized can be written as

$$\min_{\tilde{\mathbf{p}}_1 \dots \tilde{\mathbf{p}}_n} \sum_t \Delta_t \|\mathbf{T}_t - \mathbf{L}_t\|_F^2, \tag{3}$$

where $\|\cdot\|_F$ is the Frobenius norm, and Δ_ts are the volumes of tetrahedra serving as weights of the systems. As $\mathbf{T}$ is in a linear form of t's vertices, Eq.(3) can be reformulated into the form of

$$\min_{\mathbf{x}} \sum \|\mathbf{A}\mathbf{x} - \mathbf{b}\|^2 \tag{4}$$

with $\mathbf{x}$ being the vertex positions. Details of the reformulation can be found in Appendix A. The positions under ARAP deformation can be determined by computing the least-square solution of $\mathbf{Ax} = \mathbf{b}$. After that, this procedure is iterated to compute new transformation matrices $\mathbf{L}_t$ by using the new vertex positions, and then update the positions of vertices by these newly determined matrices. The iteration is stopped when the ARAP energy in Eq.(3) converges.

2.1. Iterative Rounding Algorithm

Different from the above continuous ARAP deformation, for the mixed-integer shape optimization problem, some tetrahedra (e.g., r) must be deformed to one of k pre-defined shapes: $\mathbf{S}_r^j = \{\mathbf{q}_1^j, \mathbf{q}_2^j, \mathbf{q}_3^j, \mathbf{q}_4^j\}$, $(j = 1, \dots, k)$ - i.e., the pre-defined shapes are served as the discrete variables here. These tetrahedra are called *integer tetrahedra*, and r must be projected to one of these k shapes in optimization. An iterative rounding algorithm is developed as follows (see the flow chart in Fig.2 as well).

1. All the integer tetrahedra that have not been rounded are stored in a list, *IntList*. For each tetrahedron $r \in IntList$, the shape similarities between it and its corresponding pre-defined shapes are computed.
2. In the set of pre-defined shapes for r, the shape, $\mathbf{S}_r^j$, that is the most similar to r is selected to generate a score of similarity as SIM_r;

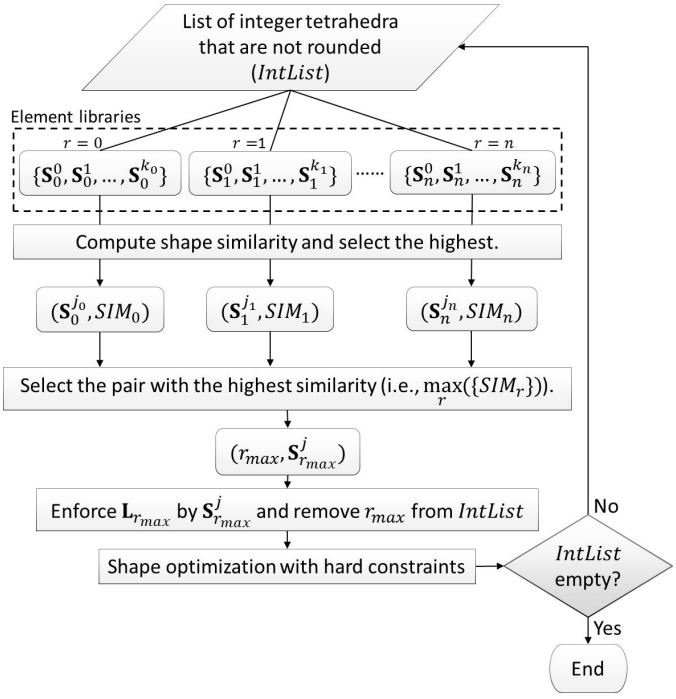

Figure 2: Flow chart of the iterative rounding algorithm.

3. Among all the pairs for different integer tetrahedra, the one having the highest similarity (i.e., $\max_r(\{SIM_r\})$) is selected: $(r_{max}, \mathbf{S}_{r_{max}}^j)$.
4. The selected integer tetrahedron, r_{max}, is rounded by setting its rest position to be $\mathbf{S}_{r_{max}}^j$,
   i.e., $\mathbf{P} \leftarrow \mathbf{Q}^j = [\mathbf{q}_1^j - \mathbf{q}_4^j \;\; \mathbf{q}_2^j - \mathbf{q}_4^j \;\; \mathbf{q}_3^j - \mathbf{q}_4^j]$.
5. Computing $\mathbf{L}'_{r_{max}} = \mathbf{UV}$ from $\mathbf{T}'_{r_{max}} = \tilde{\mathbf{P}}(\mathbf{Q}^j)^{-1} = \mathbf{U}\Sigma\mathbf{V}$.
6. Adding r_{max} to the set of rounded tetrahedra, R, and converting r_{max} to hard constraints with $\mathbf{L}_{r_{max}} = \mathbf{L}'_{r_{max}}$ in the rest steps of shape optimization as

$$\min_{\tilde{\mathbf{p}}_1 \dots \tilde{\mathbf{p}}_n} \sum_t \Delta_t \|\mathbf{T}_t - \mathbf{L}_t\|_F^2 \quad s.t., \forall r \in R, \mathbf{T}_r = \mathbf{L}'_r. \tag{5}$$

7. Optimization with hard constraints (in Eq.(5)) is employed to update the positions of vertices. Details will be discussed in section 4.
8. Go back to step 2 until all integer tetrahedra have been rounded (i.e., *IntList* is empty).

This is a bare form of the MI-ARAP shape optimization framework. A more sophisticated approach will be discussed in section 3 to round all tetrahedra belonging to the same component together. Before that, we discuss the method to compute shape similarities on tetrahedra.

2.2. Shape Similarity

Assume having an element library of pre-defined shapes $\{\mathbf{S}_r^1, \mathbf{S}_r^2, \dots, \mathbf{S}_r^k\}$ for an integer tetrahedron r, we need to find out the best-fit to define the target shape of r. To evaluate the shape similarity, ARAP energy defined in Eq.(3) can be used. Specifically, ARAP energy is computed for each of these shapes in element library by setting the rest position of r to be $\mathbf{S}_r^j$,

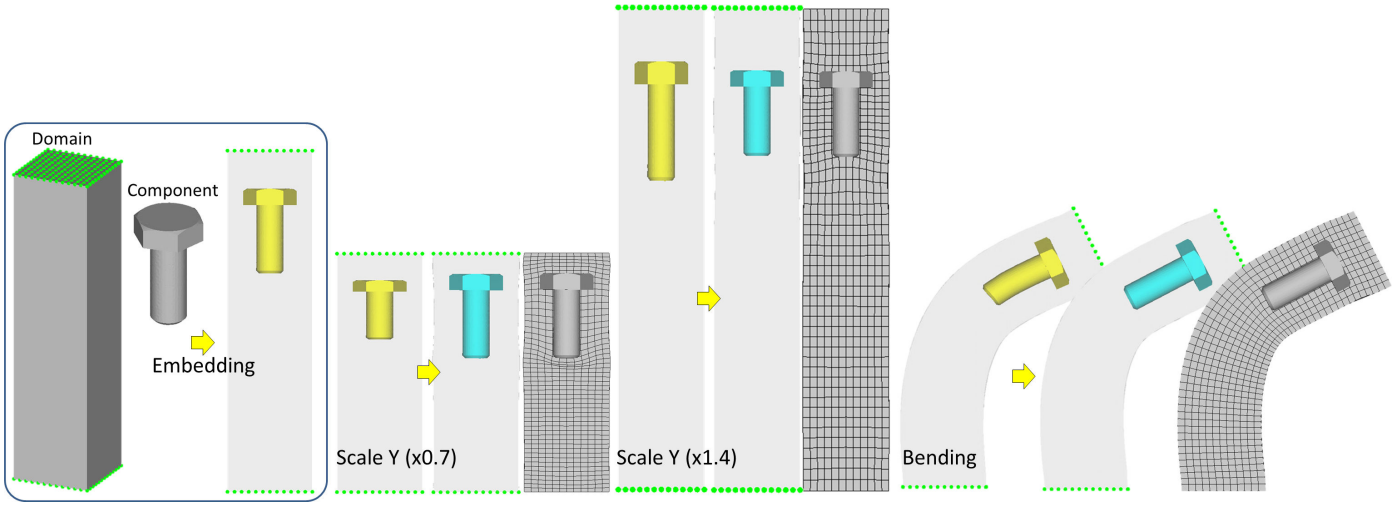

Figure 3: An illustration of the MI-ARAP optimization by a bolt embedded in a rectangular domain: This figure demonstrates how the mapping performs when the bolt is rigidly transferred to different target domains. The green points are fixed to serve as pins during the deformation.

i.e., $\mathbf{P} \leftarrow \mathbf{Q}^j = [\mathbf{q}_1^j - \mathbf{q}_4^j \quad \mathbf{q}_2^j - \mathbf{q}_4^j \quad \mathbf{q}_3^j - \mathbf{q}_4^j]$. For a pre-defined shape, a transformation matrix can be obtained from $\mathbf{T}^j = \tilde{\mathbf{P}}(\mathbf{Q}^j)^{-1}$. $\mathbf{T}^j$ is then decomposed to $\mathbf{T}^j = \mathbf{U}^j\Sigma^j\mathbf{V}^j$ where $\Sigma^j = diag\{\sigma_1^j, \sigma_2^j, \sigma_3^j\}$. As a result, the ARAP energy for the shape $\mathbf{S}_r^j$ is defined as

$$E(\mathbf{S}_r^j) = \sum_{i=1}^{3}(\sigma_i^j - 1)^2. \tag{6}$$

The smaller ARAP energy is, the higher shape similarity is given (i.e., $SIM \propto 1/E$). The optimal case gives $E(\mathbf{S}_r^j) = 0$ under which condition the pre-defined shape $\mathbf{S}_r^j$ has the same shape of r.

In our iterative rounding algorithm, Eq.(6) is employed to measure the shape similarity. It can be easily extended to measure the shape similarity for a group of tetrahedra.

3. Design Transfer with Standardized Components

Different from the bare form of MI-ARAP presented in the above section, we need to process components in design transfer. A component usually consists of a group of tetrahedra. As a result, we have to process the tetrahedra in a group all together.

3.1. Group-based Iterative Rounding

Given a product Υ that has a set of components, the components can be classified into non-standardized and standardized. Each standardized component ($\omega_l \in \Upsilon : l = 1, \ldots, L$) is associated with an element library having k pre-defined shapes – denoted by $\{\omega_l^j\} : j = 1, \ldots, k$. A standardized component is composed of a set of integer tetrahedra, and its shape is defined as

$$\omega_l^j = \bigcup_r \mathbf{S}_{r,l}^j,$$

where $\mathbf{S}_{r,l}^j \in \omega_l^j$ is the shape of an integer tetrahedron r. Rounding a standardized component ω_l into a shape ω_l^j can be realized by rounding each tetrahedron r of ω_l to a pre-defined shape $\mathbf{S}_{r,l}^j$. Furthermore, measuring the shape similarity between a component and the pre-defined shapes stored in the element library is based on measuring the shape similarity of all tetrahedra constituting the component, i.e.,

$$E(\omega_l^j) = \frac{1}{|\omega_l|}\sum_{r\in\omega_l} E(\mathbf{S}_{r,l}^j), \tag{7}$$

where $|\omega_l|$ is the number of tetrahedra in ω_l.

Assuming the shape of a product Υ is originally designed for a human body H, it is transferred to another body H' by a warping function such as t-FFD [4, 18] to give an initial shape Υ'. As t-FFD tends to preserve the spatial relationship between points, the shape of Υ' will fit the body shape H' (as shown in the middle of Fig.1). However, the warping result deforms all components in Υ including those standardized components, and this should be corrected through the MI-ARAP shape optimization. To prepare for the shape optimization with standardized components, volumetric cross-parameterization [45] is computed for the standardized components, $\{\omega_l^j\}$, in the same series (e.g., among the bolts from M4 to M7 in Fig.5).

We start the design automation by warping all tetrahedra of Υ to a shape Υ' around H' by t-FFD. Therefore, the current positions of tetrahedra used in the MI-ARAP framework can be obtained. Without loss of generality, we assume Υ' is the optimal shape for H', so that the shapes of tetrahedra in Υ' are used as rest shapes while applying the MI-ARAP shape optimization. The tetrahedra do not belong to any standardized components are called *flexible* - the target transformation matrix for them will be obtained from their shape in Υ' as the rest positions. Moreover, some vertices can be specified by users to serve as pins in the process. Their positions are fixed during the optimization.

3.2. Behavior of MI-ARAP Optimization

To verify the performance of our MI-ARAP optimization framework, the following experiments are taken. Three tests

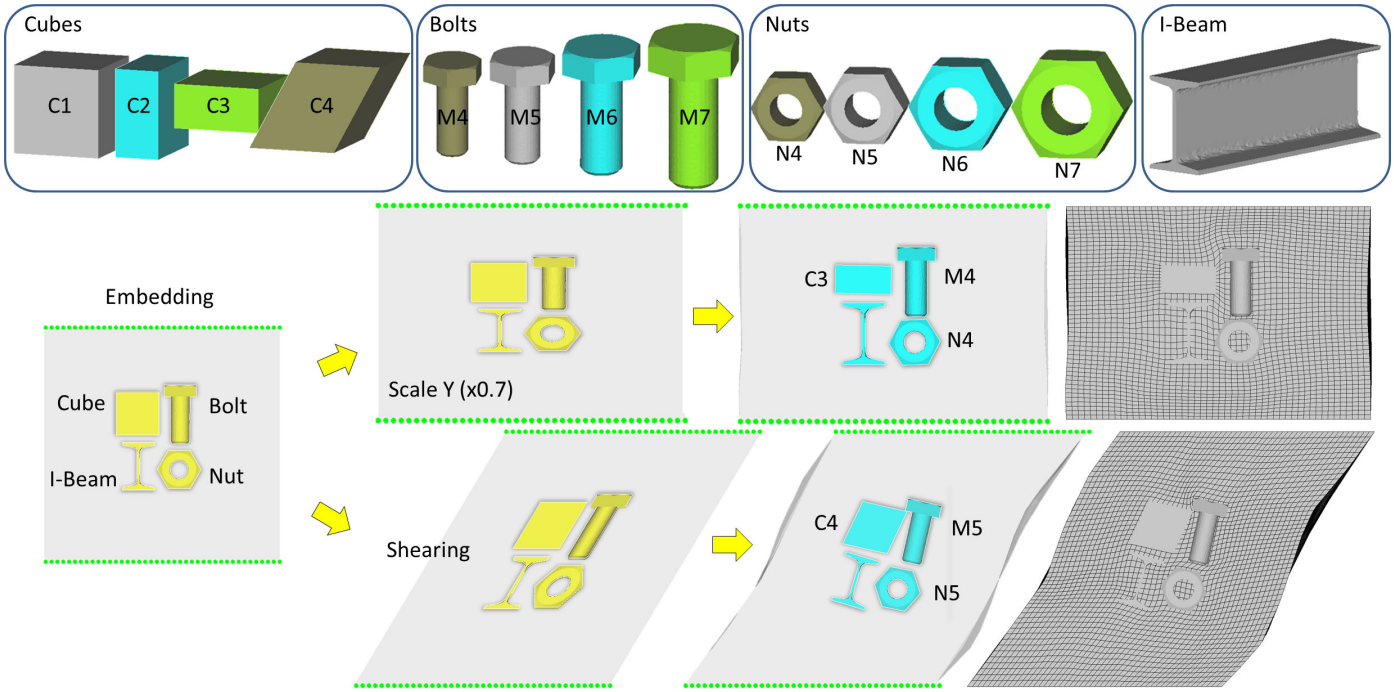

Figure 5: An illustration of shape matching in MI-ARAP: There are four elements in the element libraries of cubes, bolts, nuts and I-beam. Different types of deformations will make the components converge to different standardized components under the MI-ARAP shape optimization framework. When applying shearing and scaling deformations on the domain with standardized components (in yellow) shown on the left, the embedded cube is rounded to C4 and C3, the bolt is rounded to M5 and M4, and the nut is rounded to N5 and N4 respectively. Note that the region in gray is filled by flexible tetrahedra.

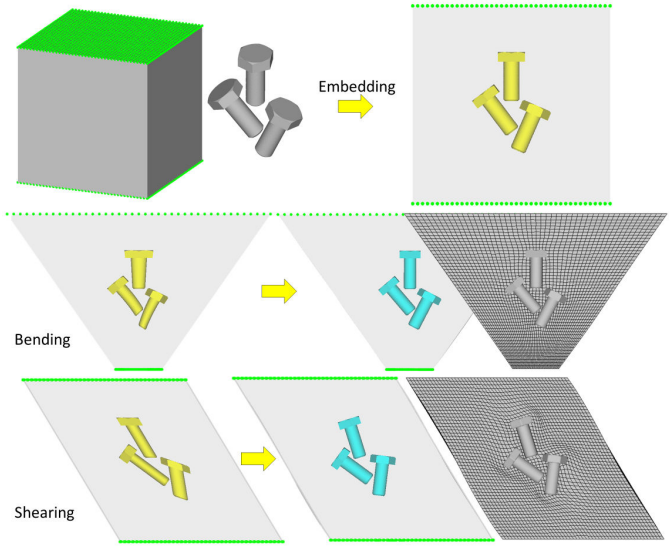

Figure 4: An illustration of the behavior when multiple components are embedded in the domain of MI-ARAP shape optimization.

are conducted to study the optimization behavior by scaling, blending or shearing a template domain. First, a standardized component is embedded in the domain (see Fig.3), and the component is enforced to deform back to its original shape by the MI-ARAP shape optimization when letting the element library have only one element. This test is used to check if the optimization can converge. Second, we take a similar test but having multiple standardized components (as shown in Fig.4), and the components are put close to each other. We aim at showing the behavior of MI-ARAP optimization when dealing with interaction between components. Third, we embed different standardized components in a domain, and each of the components has its own element library (see Fig.5). This is to test whether the shape matching algorithm can successfully select a shape from the element library to well-fit the target domain.

The first experiment is shown in Fig.3. The standardized component is a bolt embedded in a rectangular domain. Integer tetrahedra, which form the standardized component, are shown in yellow color. When the domain is deformed, the component is warped together with the domain. The shape optimization algorithm is expected to deform the component back to its original shape (but may result in different position and orientation). The deformed shape is input as the current position, and the original shape serves as the rest position. Our MI-ARAP framework can successfully deform the component to its original shape. The cross-section of computational domain in volumetric mesh is also shown to illustrate the distortion.

The second experiment is shown in Fig.4. Multiple bolts are embedded in a domain, and they need to be deformed back to their original shapes. Although they are very close to each other, the computation of our optimization approach can still converge to the original shape and generate no intersection. In short, the framework has no difficulty to handle multiple components.

In the third experiment shown in Fig.5, four components are embedded into the template domain, including a cube (ω_1), a bolt (ω_2), a nut (ω_3), and an I-beam (ω_4). The cube, the bolt and

the nut have four different standardized shapes in their element libraries, and the I-beam has only one shape, i.e.,

- Cube (ω_1): ω_1^1=C1, ω_1^2=C2, ω_1^3=C3, ω_1^4=C4;
- Bolt (ω_2): ω_2^1=M4, ω_2^2=M5, ω_2^3=M6, ω_2^4=M7;
- Nut (ω_3): ω_3^1=N4, ω_3^2=N5, ω_3^3=N6, ω_3^4=N7;
- I-beam (ω_4) is rigid.

In this test, shearing and scaling are applied. Our approach can automatically select the best shapes from the element libraries of components, and optimize their positions and orientations. {C4, M5, N5} are selected for the shearing example, and {C3, M4, N4} are selected for the scaling example. Referring to the cubic component, the sheared cube (C4) is selected for the shearing example, and the scaled cube (C3) is selected for the scaling example. This experiment demonstrates the ability of selecting components' shapes.

4. Enforcement of Hard Constraints

For the rounding procedure, we need to optimize the objective function meanwhile enforcing hard constraints in step 7. Simply giving a higher weight for the integer tetrahedra is not able to deform a component to a pre-defined shape exactly. In our implementation, once the ARAP energy is minimized, a further step is applied to project and lock the positions of corresponding vertices to enforce the pre-defined shapes. Locking the positions of vertices can be realized by moving all the known values (locked positions) to the right hand side of the linear system, which avoids adding equations into the system for hard constraints. The equation system processed in this way is stable during the optimization, and the pre-defined shape can be exactly preserved.

4.1. Reformulation

The ARAP shape optimization in Eq.(3) is reformulated to

$$\min_{\tilde{\mathbf{p}}_1 \dots \tilde{\mathbf{p}}_n} \sum_t w_t \tilde{\Delta}_t \|\mathbf{T}_t - \mathbf{L}_t\|^2 \tag{8}$$

by using different weights as

$$\tilde{\Delta}_t = \min(\frac{\Delta_{max}}{\Delta_t}, \delta)$$

with Δ_{max} being the maximal value of $\{\Delta_t\}$. In our implementation, $\delta = 5$ works well for all examples. The weight w_t is assigned as 500 for an integer tetrahedron, and $w_t = 1$ is employed for the rest tetrahedra.

The weights are set in this way because of the following reasons. First, the integer tetrahedra should have a higher priority to ensure that the shape is as-close-as-possible to the pre-defined shape. Second, distortion is introduced when the integer tetrahedra are changed. This distortion is disturbed to the whole mesh, and a larger tetrahedron should have larger room to absorb the distortion. This formulation is different from the general ARAP deformation, in which a larger tetrahedron is set to be more rigid in order to sustain the overall rigidity. Inversely, a tetrahedron with larger volume is expected absorb more distortion here. The maximal volume, $\Delta_{\max}$, is used to prevent extremely large weights that may lead to instable numerical systems.

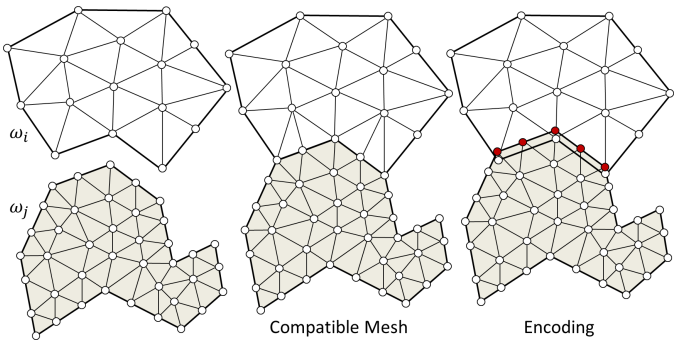

Figure 6: An illustration of incorporating the assembly constraints among multiple components: (Left) Given two components, ω_i and ω_j, which will be in contact. (Middle) The configuration of these two fixed components can be realized by generating compatible mesh models. (Right) A more general assembly constraints can be formulated by encoding the vertices (red) on the contact surfaces to another component's tetrahedra. The encoding is realized by barycentric coordinate.

4.2. Project to Position Constraints

The optimization will be iterated until the ARAP energy converges, which can be detected by the change of ARAP energy in successive iterations. After that, a further projection step is applied to finalize the shapes of the integer tetrahedra and fix their vertices' positions as the hard constraints. This iterative rounding process has two phases. The first phase is to round the components to their pre-defined shapes and assign them with higher weights in Eq.(8). The second phase projects the components' vertices. Again, the greedy rounding approach is based on the measurement of shape similarity.

Assume a component have m vertices, the rest positions of a pre-defined shape are $\mathbf{q}_i$ ($i \in 1 \dots m$). Their corresponding current positions are $\tilde{\mathbf{q}}_i$ ($i \in 1 \dots m$). As $m > 4$, we can determine $\mathbf{T}$ and $\mathbf{d}$ in $\mathbf{T}\mathbf{q}_i + \mathbf{d} = \tilde{\mathbf{q}}_i$ by a least square solution. After that, the exact positions for all the vertices in a component can be computed by $\mathbf{L}\mathbf{q}_i + \mathbf{d}$, where $\mathbf{L}$ is the rigid transformation derived from $\mathbf{T}$ (e.g., by Eq.(2)). These positions are the projected positions, which are fixed as hard constraints in the later iterations of optimization.

5. Products with Assembled Components

The basic assumption of our method discussed in above sections is that the components are meshed together with the domain consists of flexible tetrahedra. In other words, we assume that a compatible mesh can be generated for all the components as well as the domain of computation (see the middle of Fig.6). However, it relies too much on the quality of meshing results. In many cases, it is hard to generate a compatible mesh to embed all the components. A more flexible method is to encode the relationship between components in the numerical system.

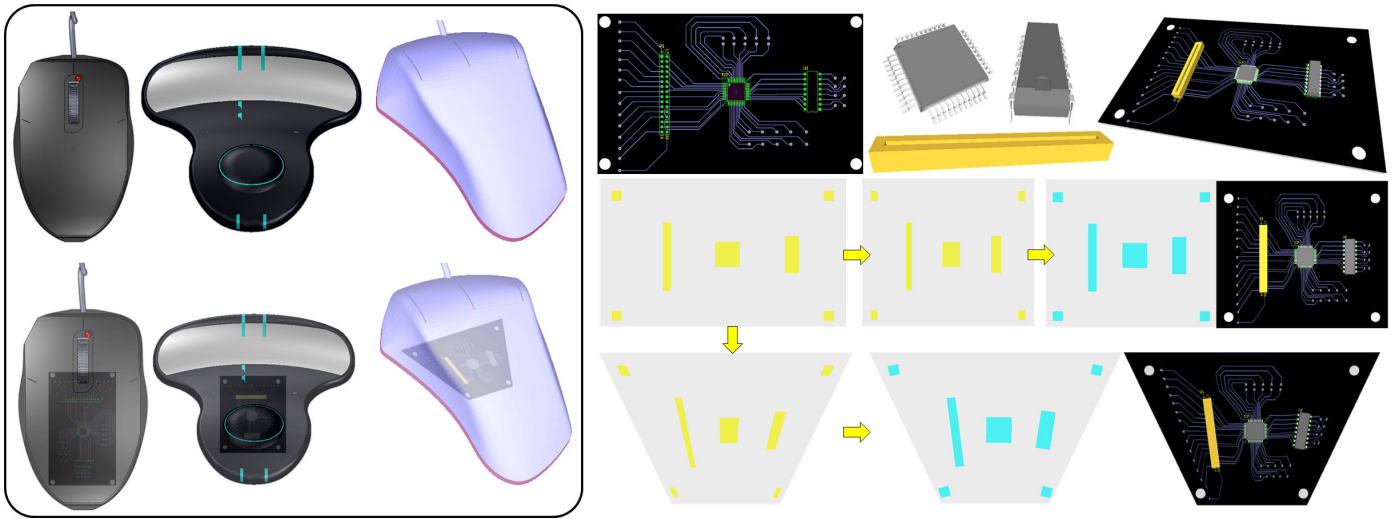

Figure 7: In many electronic products (e.g., mouse), circuit board is an essential part. With the help of our framework, the design of printed circuit board can be reused for human-centric electronic products. Note that, the mapping is free of self-intersection, so that the conductive traces will not intersect among each others.

Given two components ω_i and ω_j as what is shown in Fig.6 (left), the contact surfaces must be deformed together when ω_i and ω_j are contacted in the original design. This relationship can be formulated as hard constraints. Instead of generating a compatible mesh for the components, we encode the ω_j's vertices near the contact surfaces onto ω_i (see the right of Fig.6). Specifically, the vertices in red which belong to ω_j are represented by the barycentric coordinate of the tetrahedra that they are encoded to. In other words, they are not variables anymore. For instance, if a vertex $\mathbf{p}_i$ is encoded to a tetrahedron having vertices $\{\mathbf{p}_a, \mathbf{p}_b, \mathbf{p}_c, \mathbf{p}_d\}$ with the barycentric coordinates $\{u_0, u_1, u_2, u_3\}$ $(u_0 + u_1 + u_2 + u_3 = 1)$, $\mathbf{p}_i$ can be expressed by $\mathbf{p}_i = u_0\mathbf{p}_a + u_1\mathbf{p}_b + u_2\mathbf{p}_c + u_3\mathbf{p}_d$ in the formulation. As a result, each component of a product can be tessellated independently. This encoding method can also be applied to constrain the relationship between fully overlapped meshes, e.g., domain $\rightarrow$ component.

6. Experimental Examples

In this paper, we present a MI-ARAP shape optimization framework. This framework supports design transfer of products with standardized components. This feature is important because many mechanical and electronic components have a high cost in customization. Making use of standardized components, which are manufactured by mass production, can reduce the cost of building customized products. Examples are shown in this section.

Our implementation is written in C++ code and the results are generated on an Intel® Core^{TM} i7-3770S CPU at 3.10GHz with 8GB RAM. The shape optimization can be computed efficiently on our program. For example, the bicycle model that has eight components result in 182,509 tetrahedra with 45,379 vertices in total. This leads to a least-square system having the matrix $\mathbf{A}$ (see Eq.(4)) with dimension: $547,627 \times 45,379$ at the beginning of optimization. The size of linear system is reduced after converting rounded components into hard constraints in term of positions. In our experimental tests, the result of a bicycle example can be obtained in 78 seconds on the above platform.

Exoskeleton

An exoskeleton is designed to help paralyzed people to walk again. As the exoskeleton is mounted on a human body (see Fig.1), it should be customized so that the person wearing it will not feel uncomfortable. Most of the components in the exoskeleton are non-standardized. However, the controllers (see the pink blocks in Fig.1) need to be rigid as they are electronic components. The joints (in red) that are mechanical components need to be rigid too. After specifying these constraints in our MI-ARAP framework, the optimization will preserve the shape of the mechanical and electronic components while determining how the shapes of other non-standardized components should be.

Circuit Board

For human-centric electronic products, i.e., hair dryer or mouse, there are many electrical parts inside. Circuit board is an essential part. A circuit board is usually integrated with a lot of electronic components (e.g., DSP chips and sockets). The shapes of these components are standardized, and our method can help to realize customization by using such components. See the example shown in Fig.7, there are a DSP, a DIP16, and a ram socket on a printed circuit board (PCB). The PCB can be customized based on different applications, and the examples of scaling and blending the template domain are shown in Fig.7. Freeform deformations can also be taken in other applications. As shown in Fig.7, the MI-ARAP framework can produce customized results while keeping the components in standard shapes. More importantly, our method can ensure an intersection-free mapping. It means that the conductive traces will not have interference with each other. As a result, the customized PCB can be directly manufactured, and the components can be installed on

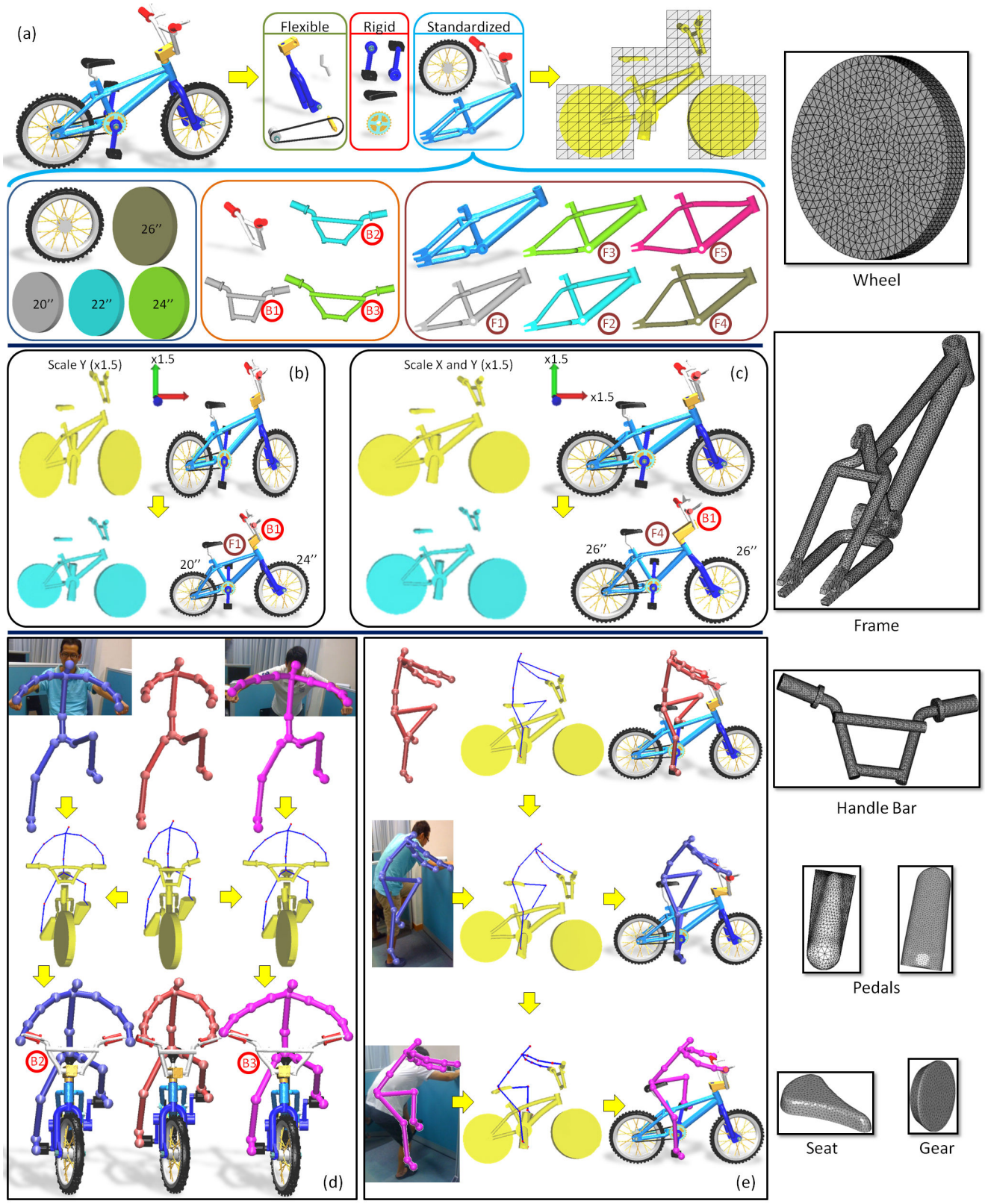

Figure 8: An illustration of using our MI-ARAP shape optimization framework for customizing bicycles: (a) A bicycle is constructed by flexible, rigid and standardized components and the volumetric meshes of different components are shown in the right, (b) a result of shape optimization for customization driven by simply scaling y-coordinates, (c) different result can be obtained by scaling both x- and y-coordinates, (d) results of customization driving by skeletons of users captured by Kinect sensors, and (e) another view for the results of skeleton-driven bicycle customization.

the circuit board without any difficulty.

Bicycle

A bicycle is composed of many components (see Fig.8). Some of them are not allowed to change (e.g., belt and adjusters). Some of them are unchangeable (e.g., pedals, seat saddle and gears), which are required to be rigid. Some of them are standardized components (e.g., wheels, handlebar and frame), where each of them are accompanied with an element library. The components are embedded in a design domain in terms of volumetric mesh that covers all the components of a bicycle. For simplicity, the complicated components (e.g., pedals, gears and wheels) are replaced by their convex hull in the optimization.

Given this configuration, the lists of components $\{\omega_l^k\}$ in element libraries are summarized are follows:

- Wheels (ω_1): $\omega_1^1 = 20''$, $\omega_1^2 = 22''$, $\omega_1^3 = 24''$, $\omega_1^4 = 26''$;
- Handlebar (ω_2): ω_2^1=B1, ω_2^2=B2, ω_2^3=B3;
- Frame (ω_3): ω_3^1=F1, ω_3^2=F2, ω_3^3=F3, ω_3^4=F4, ω_3^5=F5;
- Pedal (ω_4), seat saddle (ω_5), and gear (ω_6) are rigid.

First, the selection of combinations is driven by applying global scaling. The domain is scaled in Y and XY direction, and the scaling results are used as the target domain. After that, our shape optimization algorithm is applied to find out the best-fit components and the overall mapping. Our method can suggest the best combination and successfully find out the relationship between them. For example, for scaling 1.5 times in Y direction, the combination {20″, 24″, B1, F1} is selected; and {26″, 26″, B1, F4} is selected for scaling in both X and Y directions by a factor of 1.5. The results can be found at the middle row of Fig.8.

Second, the customization is driven by human skeletons, which can be generated by Kinect [46]. First of all, the joints of a template skeleton is captured for a new customer. The new positions of points are used as handles to deform the template domain and the components to get the initial shapes. In our implementation, ARAP deformation driven by position-handles is employed. Once the initial shapes are ready, they are passed to our framework to work out the optimized solution that is the best combination for the new design. Note that, for example in the bottom row on the left of Fig.8, the handlebar cannot exactly fit to the hand of the skeletons. It is due to the reason that handlebar is a standardized component, so that the system can only output an as-close-as-possible result. This is the tradeoff in using standardized components when the shapes defined in element library are not sufficient. Otherwise, the handlebar needs to be fabricated in a customized manner.

With the help of Kinect, we can quickly capture the skeleton of a new customer. By our approach, the best combination and configuration for different customers can be formed automatically. Customized bicycles with standardized components can be produced thereafter.

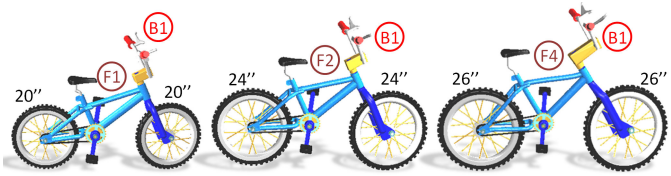

Figure 9: A series of conventional combination of bicycles with standard wheel sizes as 20″, 24″ and 26″.

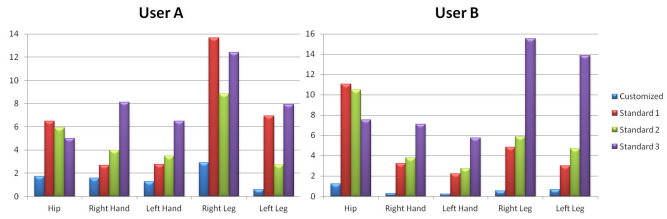

Figure 10: Distances between the positions of joints and the target corresponding points on the bicycles, where the customized bicycles generated by our approach is compared with three standard bicycles shown in Figure 9. The results are compared on two users – the ones shown in Figures 8(d) and 8(e).

6.1. Verification and comparison

We verify the functionality of our approach on the example of customized bicycles by experimental tests. On the template designed for a well-fitting bicycle, the closest points to five joints – hip, right hand, left hand, right ankle, left ankle – are measured and used as benchmarks for verifying the quality of a new design. For a better design, the distances between the joints and these benchmarks should be shorter. Therefore, these distances are used as fitness-metrics for quality measurement in our experimental tests.

We first try to fit three conventional combinations of bicycles with standard wheel sizes as 20″, 24″ and 26″ (see Fig.9) onto two users that have been shown in Fig.8(d). Rigid body registration by using the joints and their corresponding points on the bicycles is employed to determine the best position/orientation of the standard bicycles. Specifically, a rigid transformation centered at the average position of all joints is determined to transfer the standard bicycles onto an 'optimal' position/orientation according to the users. Details about this anchor points-based rigid body transformation can be found in [3]. Note that, when computing the rigid transformation, all joints (not only the five joints mentioned above) of user's skeleton are used to determine an optimal solution. These results are compared with the results obtained by our optimization framework. The distances between the five joints and the benchmarks on the bicycles are measured and shown in Fig.10. It is easy to find that our results present much less errors on the fitness-metrics. These tests verify that our shape optimization approach can improve the fitness of customized design while still using standardized components. A more practical verification can only be taken by collaborating with the industrial manufacturing companies.

7. Conclusion and Discussion

In this paper, a shape optimization framework is presented for the design automation of human-centric products. This framework solves a mixed-integer shape optimization problem, and provides a function to generate customized product by using standardized components. *As-Rigid-As-Possible* (ARAP) energy is reformulated for this purpose, and hard constraints on transformation matrices are included by a projection-based approach. A shape matching algorithm is proposed to find out the best-fit combination for the standardized components.

7.1. Limitations

The experimental tests are encouraging. However, there are also some limitations. First, the weights in ARAP energy are based on the volume of tetrahedron. If the mesh quality is poor, the weight must be carefully handled; otherwise, it may make the linear equation system unstable. Second, this approach includes the meshes of all components in the same system of numerical optimization. When the size of mesh is very large, it may cause memory consumption problems.

Although self-intersection rarely occurs when testing the examples shown in this paper, it is hard to guarantee intersection-free when extreme cases are processed. Self-intersection happens when any tetrahedron becomes a void. However, it can be easily checked during the deformation of tetrahedra. Actually, in our current implementation, we have already checked this singularity. If there is any tetrahedron becoming singular, the positions of its vertices will be locked to prevent self-intersection.

On the other aspect, the convergence of our shape optimization depends on the number of standardized components stored in the element library. When there are too few components, the shape optimization may be stuck and introduces large error on the results of computation. Moreover, we expect that the volumetric meshes of all components stored in the same element library have the same connectivity. Although they can be obtained by the surface cross-parameterization [1, 2] plus spatial warping [3], a self-intersection free volumetric mesh may still hard to obtain when the components in an element library have tremendous shape-variation.

7.2. Future work

In the near future, we plan to work on a few extensions of this approach. First, we will consider how to handle the models with large size of tetrahedra. Second, the framework is planned to use in real products design to further verify its robustness in practical usage. Lastly, the series of standardized components are currently determined according to the manufacturing process. A possible future work is to analyze the demanded best-fit components and develop an algorithm to generate the classification automatically from the statistical data.

References

[1] T. H. Kwok, Y. Zhang, C. C. L. Wang, Constructing common base domain by cues from Voronoi diagram, Graphical Models 74 (4), GMP2012.

[2] T. H. Kwok, Y. Zhang, C. C. L. Wang, Efficient optimization of common base domains for cross parameterization, IEEE Transactions on Visualization and Computer Graphics 18 (10) (2012) 1678 –1692.

[3] C. C. L. Wang, K. C. Hui, K. M. Tong, Volume parameterization for design automation of customized free-form products, IEEE Transactions on Automation Science and Engineering 4 (1) (2007) 11 –21.

[4] C. C. L. Wang, Y. Wang, M. M. F. Yuen, Design automation for customized apparel products, Computer-Aided Design 37 (7) (2005) 675–691.

[5] M. Botsch, L. Kobbelt, M. Pauly, P. Alliez, B. Levy, Polygon Mesh Processing, AK Peters, 2010.

[6] G. H. Bendels, R. Klein, Mesh forging: editing of 3D-meshes using implicitly defined occluders, in: Proceedings of the 2003 Eurographics/ACM SIGGRAPH symposium on Geometry processing, SGP '03, 2003, pp. 207–217.

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

[8] R. Zayer, C. Rössl, Z. Karni, H.-P. Seidel, Harmonic guidance for surface deformation, Computer Graphics Forum 24 (3) (2005) 601–609.

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

[10] Y. Yu, K. Zhou, D. Xu, X. Shi, H. Bao, B. Guo, H.-Y. Shum, Mesh editing with poisson-based gradient field manipulation, ACM Transactions on Graphics 23 (3) (2004) 644–651.

[11] A. Sheffer, V. Kraevoy, Pyramid coordinates for morphing and deformation, in: Proceedings of the 3D Data Processing, Visualization, and Transmission, 2nd International Symposium, 3DPVT '04, IEEE Computer Society, 2004, pp. 68–75.

[12] S. Fröhlich, M. Botsch, Example-driven deformations based on discrete shells, Computer Graphics Forum 30 (8) (2011) 2246–2257.

[13] T. Igarashi, T. Moscovich, J. F. Hughes, As-rigid-as-possible shape manipulation, ACM Transactions on Graphics 24 (3) (2005) 1134–1141.

[14] O. Sorkine, M. Alexa, As-rigid-as-possible surface modeling, in: Proceedings of the fifth Eurographics symposium on Geometry processing, SGP '07, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 2007, pp. 109–116.

[15] K. Zhou, J. Huang, J. Snyder, X. Liu, H. Bao, B. Guo, H.-Y. Shum, Large mesh deformation using the volumetric graph laplacian, ACM Transactions on Graphics 24 (3) (2005) 496–503.

[16] M. Botsch, M. Pauly, M. Gross, L. Kobbelt, PriMo: coupled prisms for intuitive surface modeling, in: Proceedings of the fourth Eurographics symposium on Geometry processing, SGP '06, Eurographics Association, 2006, pp. 11–20.

[17] M. Botsch, M. Pauly, M. Wicke, M. H. Gross, Adaptive space deformations based on rigid cells, Computer Graphics Forum 26 (3) (2007) 339–347.

[18] K. G. Kobayashi, K. Ootsubo, t-FFD: free-form deformation by using triangular mesh, in: Proceedings of the eighth ACM symposium on Solid modeling and applications, SM '03, ACM, New York, NY, USA, 2003, pp. 226–234.

[19] V. Kraevoy, A. Sheffer, Cross-parameterization and compatible remeshing of 3D models, ACM Transactions on Graphics 23 (3) (2004) 861–869.

[20] A. Sheffer, E. Praun, K. Rose, Mesh parameterization methods and their applications, Foundation and Trends in Computer Graphics and Vision 2 (2) (2006) 105–171.

[21] N. Umetani, D. M. Kaufman, T. Igarashi, E. Grinspun, Sensitive couture for interactive garment modeling and editing, ACM Transactions on Graphics 30 (4) (2011) 90:1–90:12.

[22] S. D. Porumbescu, B. Budge, L. Feng, K. I. Joy, Shell maps, ACM Transactions on Graphics 24 (3) (2005) 626–633.

[23] R. W. Sumner, J. Schmid, M. Pauly, Embedded deformation for shape manipulation, ACM Transactions on Graphics 26 (3).

[24] T. Milliron, R. J. Jensen, R. Barzel, A. Finkelstein, A framework for geometric warps and deformations, ACM Transactions on Graphics 21 (1) (2002) 20–51.

[25] Y. Meng, C. C. L. Wang, X. Jin, Flexible shape control for automatic resizing of apparel products, Computer-Aided Design 44 (1) (2012) 68 – 76, Digital Human Modeling in Product Design.

[26] R. Brouet, A. Sheffer, L. Boissieux, M. P. Cani, Design preserving garment transfer, ACM Transactions on Graphics 31 (4) (2012) 36:1–36:11.

[27] R. W. Sumner, J. Popović, Deformation transfer for triangle meshes, ACM Transactions on Graphics 23 (3) (2004) 399–405.
[28] K. Zhou, W. Xu, Y. Tong, M. Desbrun, Deformation transfer to multi-component objects, Computer Graphics Forum 29 (2) (2010) 319–325.
[29] M. Ben-Chen, O. Weber, C. Gotsman, Spatial deformation transfer, in: Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, SCA '09, ACM, New York, NY, USA, 2009, pp. 67–74.
[30] I. Baran, D. Vlasic, E. Grinspun, J. Popović, Semantic deformation transfer, ACM Transactions on Graphics 28 (3) (2009) 36:1–36:6.
[31] O. Sorkine, D. Cohen-Or, Y. Lipman, M. Alexa, C. Rössl, H. P. Seidel, Laplacian surface editing, in: Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing, SGP '04, ACM, New York, NY, USA, 2004, pp. 175–184.
[32] S. Schaefer, T. McPhail, J. Warren, Image deformation using moving least squares, ACM Transactions on Graphics 25 (3) (2006) 533–540.
[33] L. Liu, L. Zhang, Y. Xu, C. Gotsman, S. J. Gortler, A local/global approach to mesh parameterization, in: Proceedings of the Symposium on Geometry Processing, SGP '08, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 2008, pp. 1495–1504.
[34] S. Zhang, A. Nealen, D. Metaxas, Skeleton based as-rigid-as-possible volume modeling, in: The 31st annual conference of the European Association for Computer Graphics, EG'10, 2010, pp. 21–24.
[35] A. Jacobson, I. Baran, L. Kavan, J. Popović, O. Sorkine, Fast automatic skinning transformations, ACM Transactions on Graphics 31 (4) (2012) 77:1–77:10.
[36] S. Bouaziz, M. Deuss, Y. Schwartzburg, T. Weise, M. Pauly, Shape-Up: Shaping discrete geometry with projections, Computer Graphics Forum 31 (5) (2012) 1657–1667.
[37] V. Kraevoy, A. Sheffer, A. Shamir, D. Cohen-Or, Non-homogeneous resizing of complex models, ACM Transactions on Graphics 27 (5) (2008) 111:1–111:9.
[38] R. Gal, O. Sorkine, N. J. Mitra, D. Cohen-Or, iWIRES: an analyze-and-edit approach to shape manipulation, ACM Transactions on Graphics 28 (3) (2009) 33:1–33:10.
[39] Y. Zheng, H. Fu, D. Cohen-Or, O. K. C. Au, C. L. Tai, Component-wise controllers for structure-preserving shape manipulation, Computer Graphics Forum 30 (2) (2011) 563–572.
[40] M. Bokeloh, M. Wand, V. Koltun, H.-P. Seidel, Pattern-aware shape deformation using sliding dockers, ACM Transactions on Graphics 30 (6) (2011) 123:1–123:10.
[41] M. Cabral, S. Lefebvre, C. Dachsbacher, G. Drettakis, Structure-preserving reshape for textured architectural scenes, Computer Graphics Forum (2009) 469–480.
[42] M. Habbecke, L. Kobbelt, Linear analysis of nonlinear constraints for interactive geometric modeling, Computer Graphics Forum (2012) 641–650.
[43] C. Floudas, Nonlinear and Mixed-Integer Optimization: Fundamentals and Applications, Topics in Chemical Engineering, Oxford University Press, USA, 1995.
[44] D. Bommes, H. Zimmer, L. Kobbelt, Practical mixed-integer optimization for geometry processing, in: Proceedings of the 7th international conference on Curves and Surfaces, Springer-Verlag, Berlin, Heidelberg, 2012, pp. 193–206.
[45] X. Li, X. Guo, H. Wang, Y. He, X. Gu, H. Qin, Harmonic volumetric mapping for solid modeling applications, in: Proc. ACM symp. on Solid and physical modeling, 2007, pp. 109–120.
[46] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, A. Blake, Real-time human pose recognition in parts from single depth images, in: Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition, CVPR '11, IEEE Computer Society, Washington, DC, USA, 2011, pp. 1297–1304.

Appendix A: Formulation of ARAP as Least-Square Minimization

We reformulate the global optimization problem of ARAP in Eq.(3) into a form of least-square minimization. First of all, let $\mathbf{T}$ and $\mathbf{L}$ in Eq.(3) be

$$\mathbf{T} = \begin{bmatrix} t_{00} & t_{01} & t_{02} \\ t_{10} & t_{11} & t_{12} \\ t_{20} & t_{21} & t_{22} \end{bmatrix}, \mathbf{L} = \begin{bmatrix} l_{00} & l_{01} & l_{02} \\ l_{10} & l_{11} & l_{12} \\ l_{20} & l_{21} & l_{22} \end{bmatrix},$$

and the minimization problem of Eq.(3) can be rewritten as

$$\min \sum_t \Delta_t \sum_{i,j} \|t_{ij} - l_{ij}\|^2.$$

Recall that the transformation matrix $\mathbf{T} = \tilde{\mathbf{P}}\mathbf{P}^{-1}$. As $\mathbf{P}^{-1}$ is computed by the original position of a tetrahedron that is known, we can assume that

$$\mathbf{P}^{-1} = \begin{bmatrix} a_{00} & a_{01} & a_{02} \\ a_{10} & a_{11} & a_{12} \\ a_{20} & a_{21} & a_{22} \end{bmatrix}.$$

Therefore,

$$\mathbf{T} = \begin{bmatrix} \tilde{p}_1^1 - \tilde{p}_4^1 & \tilde{p}_2^1 - \tilde{p}_4^1 & \tilde{p}_3^1 - \tilde{p}_4^1 \\ \tilde{p}_1^2 - \tilde{p}_4^2 & \tilde{p}_2^2 - \tilde{p}_4^2 & \tilde{p}_3^2 - \tilde{p}_4^2 \\ \tilde{p}_1^3 - \tilde{p}_4^3 & \tilde{p}_2^3 - \tilde{p}_4^3 & \tilde{p}_3^3 - \tilde{p}_4^3 \end{bmatrix} \begin{bmatrix} a_{00} & a_{01} & a_{02} \\ a_{10} & a_{11} & a_{12} \\ a_{20} & a_{21} & a_{22} \end{bmatrix}$$

with $\tilde{\mathbf{p}}_\mathbf{i} = [\tilde{p}_i^1\ \tilde{p}_i^2\ \tilde{p}_i^3]^T$. As a result, the elements in $\mathbf{T}$ can be expanded as

$$t_{ij} = a_{0j}\tilde{p}_1^i + a_{1j}\tilde{p}_2^i + a_{2j}\tilde{p}_3^i - (a_{0j} + a_{1j} + a_{2j})\tilde{p}_4^i.$$

The minimization problem can then be rewritten in matrix form that yields

$$\min_{\tilde{\mathbf{p}}_1 \cdots \tilde{\mathbf{p}}_n} \|\mathbf{A}\mathbf{x} - \mathbf{b}\|_2^2,$$

where $\mathbf{x}$ is a vector containing $\tilde{\mathbf{p}}_1 \ldots \tilde{\mathbf{p}}_n$, $\mathbf{b}$ is a vector containing entries from $\mathbf{L}$, and $\mathbf{A}$ is a sparse matrix with coefficients obtained from $\mathbf{P}^{-1}$.