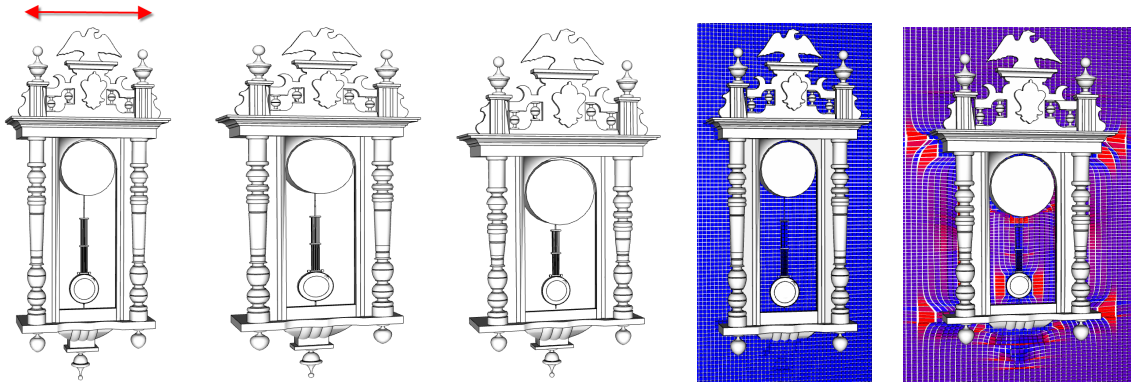
Non-homogeneous Resizing of Complex Models

Vladislav Kraevoy
University of British Columbia

Alla Sheffer
University of British Columbia

Ariel Shamir
The Interdisciplinary Center

Daniel Cohen-Or
Tel-Aviv University



(a) (b) (c) (d) (e)

Figure 1: *Resizing a clock model (267 connected components): Standard non-uniform scale distorts the shape of parts of the model, e.g. the dial (b). Our approach resizes the clock in a more natural manner protecting its shape (c). (d) and (e) show part of the protective grid before and after resizing.*

Abstract

Resizing of 3D models can be very useful when creating new models or placing models inside different scenes. However, uniform scaling is limited in its applicability while straightforward non-uniform scaling can destroy features and lead to serious visual artifacts. Our goal is to define a method that protects model features and structures during resizing. We observe that typically, during scaling some parts of the models are more vulnerable than others, undergoing undesirable deformation. We automatically detect vulnerable regions and carry this information to a protective grid defined around the object, defining a vulnerability map. The 3D model is then resized by a space-deformation technique which scales the grid non-homogeneously while respecting this map. Using space-deformation allows processing of common models of man-made objects that consist of multiple components and contain non-manifold structures. We show that our technique resizes models while suppressing undesirable distortion, creating models that preserve the structure and features of the original ones.

Keywords: 3D meshes, Scaling, Resizing, Transformations

1 Introduction

Digital 3D models are an inherent part of many applications in entertainment, design, and engineering. Discrete digital models, such as meshes, allow greater flexibility for modifications and adjustments than any preceding physical model. Still, the creation of such models from scratch is non-trivial, and there is an emerging trend toward reuse of existing models, parts, or designs. To reuse such man-made models some reshaping may be needed to better fit with other models or parts. One of the principal ways of reshaping objects is through resizing. By resizing we mean scaling or stretching the object along several orthogonal directions or dimensions to fit a new prescribed size. Resizing is often necessary to satisfy engineering requirements, to conform to related models or parts within an assembly or scene, or simply to generate variations of an existing model (Figure 2).

Resizing by simply applying a global scale usually does not suffice, as models often contain various parts and features that can be distorted by such scaling. Most notably, global *non-uniform* scaling changes the surface curvature in parts of the model distorting their shape as shown in Figure 1 (b). Clearly, the visual distortion increases as a function of the scale magnitude. However, this distortion is not distributed uniformly across the surface. The visual artifacts are localized in specific, vulnerable, regions of the surface. Other regions are indifferent to the scale regardless of its magnitude. This observation suggests that resizing must be distributed *non-homogeneously* throughout the model, protecting some parts, while possibly stretching others excessively. In contrast to local shape distortion, human perception seems far less sensitive to uniform scale of parts, even if some parts change their relative scale in the model. For instance, while the proportions between the dial and the pendulum of the clock Figure 1 (c) change, the model appears visually correct. This suggests that uniform scale of internal parts could be used to counteract undesirable non-uniform scaling.

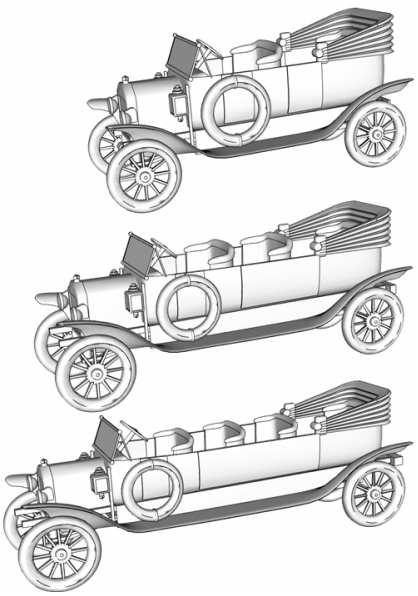


Figure 2: *The car (top) is stretched by our method to convert it into a three- and four- seater. The seats were manually added.*

Non-homogeneous resizing of an object could potentially rely on an internal parametric representation of the model. Such representations contain parameters and constraints that try to capture the designer's intent and define semantics that facilitate future modifications. This allows certain parts to be resized differently than others. However, such representations are available only in models created by specific applications such as CAD editors. Most discrete digital models such as meshes have no notion of semantic parts or constraints. The possibility of extracting this high-level information from digital models (e.g., using reverse engineering) is still limited. Instead, we present a method that uses low-level analysis of the models to automatically detect vulnerability given the resizing axes.

We also observe that most man-made models consist of multiple connected components, non-manifold structures, and intertwined features (e.g., see Figure 3). Surface-based techniques (see [Botsch and Sorkine 2008]) deal with manifolds and cannot handle individual parts while accounting for spatial relationships between them. To protect global structures and preserve spatial relations between object components our method uses a space-deformation technique. This allows handling such complex models naturally, supporting multiple components, avoiding self-intersections, and preserving global relations.

1.1 Method Overview

The method we introduce in this paper is derived from the above observations. First, for each given scaling direction, we measure local surface vulnerability by analyzing a combination of local differential surface properties. This measure distinguishes between regions that can scale non-uniformly along the given direction and those vulnerable to such scale. Next, we embed the model in a *protective volumetric grid* and gather the local surface measures to define the vulnerability of each cell to spatial deformation. Using this input, the resizing is applied non-homogeneously to the volumetric grid, protecting the model by respecting the cells' vulnerability. To counteract shape distortion and excessive stretch we allow all cells to scale uniformly with respect to one another.

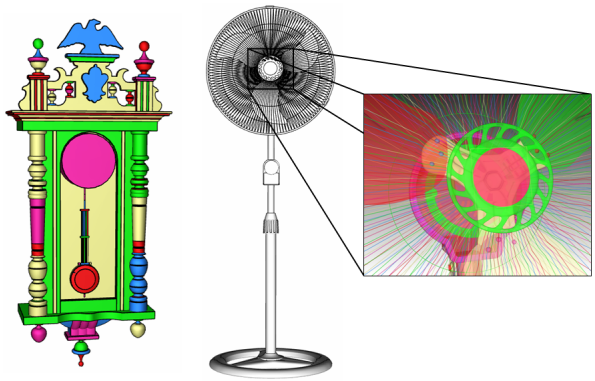


Figure 3: *Man made digital objects often contain numerous components, and complex, non-manifold structures. The coloring illustrates the different connected components of the models.*

The resizing operator is expressed as an aggregate of compatible local transformations that are obtained by optimizing a quadratic energy functional. With this optimization, regions that are less vulnerable to the stretch adhere to the global transformation in a coherent manner, while vulnerable regions locally adhere to the source shape to avoid distortion. Figures 1(d,e) and 5 show the protective grid before and after resizing. The illustration shows how the stretch is distributed non-homogeneously throughout the volume to protect the vulnerable parts of the model.

This approach bypasses the difficult reverse engineering task of detecting the global structure, while implicitly preserving the main characteristic features of the models under the stretch operation. We demonstrate the effectiveness of our resizing method on a diverse set of 3D models, generating scaled models that better preserve semantic structures of the originals.

2 Background

The problem of resizing 3D models has not received much attention so far. There is a substantial body of work studying the somewhat related problem of reverse engineering, which aims to approximate a given 3D model with parametric parts [Attene et al. 2007; Attene et al. 2006; Várady and Martin 2002; Benkö et al. 2001]. Resizing a parametric primitive is a trivial operation, but resizing a composition of such parts in a way that preserves the global design or engineering rules remains difficult. The goal of this paper is not to solve any reverse engineering problems, but to achieve similar effects visually by simpler means.

Our analysis of mesh vulnerability is related to other works on mesh analysis. Lee et al. [2005] introduce mesh saliency as a measure of the visual importance of surface details. They measure saliency as a combination of curvature at different scales. Gal and Cohen-Or [2006] have also based their feature saliency on surface curvature. They identify significant features to facilitate partial matching. Slippage analysis, presented in [Gelfand and Guibas 2004], provides another analysis tool. Slippable motions are motions which, when applied to a surface, slide the transformed version against the stationary version without forming any gaps. A slippable motion of each point P on a surface must be tangential to the surface at that point. If only rigid motions are considered, slippage analysis can detect rotationally and translationally symmetrical shapes such as planes, spheres, and cylinders, which are often found as components of mechanical parts.

Resizing can be seen as a special case of model deformation. Most

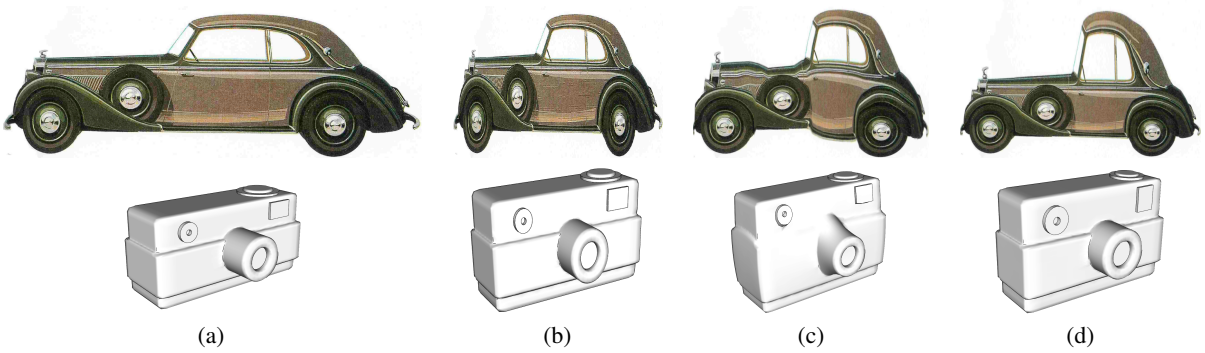


Figure 4: *2D and 3D comparisons of our method to shape-preserving/as-rigid-as-possible deformation. (a) Original models; (b) a simple non-uniform scale (shrinking the car and stretching the camera); (c) as-rigid-as-possible (top) and shape-preserving (bottom) deformation weighted to protect vulnerable parts (for the car only the wheels are classified as such); (d) our non-homogeneous resizing.*

mesh deformation techniques target natural models defined by a single manifold surface [Botsch and Sorkine 2008] and as such are unsuitable for our purposes. Masuda et al. [2007] extend these techniques to deforming CAD models by combining volumetric and surface-based deformation and using a user provided importance map to disribute the distortion across the model. Those and other typical deformation formulations penalize shear as much as bending, they tend to compensate for shear by introducing locally rigid rotations which nevertheless accumulate, leading to global distortion (Figure 4(c)(bottom)). Such formulations cannot be used in a resizing setup where shear along the resizing axes is not only unavoidable, but in large regions of the models perfectly desirable. Space-deformation methods [Sederberg and Parry 1986; Coquillart 1990; Ju et al. 2005] transform the embedding space rather than specific representation of the geometry. However, existing space-deformation methods treat the space homogeneously without being aware of the features of the objects being transformed.

Recently, content-aware resizing had been addressed in image-space [Gal et al. 2006; Avidan and Shamir 2007; Wolf et al. 2007]. However, both in images and in video, the nature of the problem, and hence the solution, is different. Image space is discrete, and resizing individual pixels is prone to aliasing artifacts. Avidan and Shamir [2007] show that by distributing the error among a finite set of seams, the majority of the pixels remain unchanged and the general look of the image is preserved. In contrast, the methods employed in [Gal et al. 2006; Wolf et al. 2007] use a global approach where the error is distributed across all pixels. Such a global approach has the advantage that important regions can rotate or uniformly scale, distributing the error to less important parts. When resizing geometric models, aliasing is not a concern, therefore we also take the global approach.

Finally, an interesting insight into resizing patterns in natural environments can be found in [Thompson 1942].

3 Our Approach

While a one-directional stretch, say a horizontal one, can distort vulnerable surface parts, it does preserve structural features such as straight lines and parallelism, as well as orthogonality between axis-aligned lines. Minimizing the distortion of the vulnerable regions with non-homogeneous similarity or shape-preserving transformations [Gal et al. 2006; Popa et al. 2006; Wolf et al. 2007] or as-rigid-as-possible deformations [Alexa et al. 2000; Sorkine and Alexa 2007; Igarashi et al. 2005] simply redistributes the distortion, pushing it to other parts of the model (Figure 4(c) (top)). A better way to preserve structure while resizing is to minimize bending, accepting a certain degree of shear and allowing uniform scale. The low-distortion result in Figure 4(d) preserves the global structure while introducing significant per-triangle shear and some changes in scale.

The vulnerability of the surface to resizing depends on the resizing direction and is fairly independent of the actual amount of stretch (see Section 4). Therefore, we first estimate the degree of *vulnerability* on the mesh given the resizing axes and then use this information inside a linear resizing formulation. This formulation effectively minimizes the visual shape distortion by distributing the resizing non-homogeneously across the input model. Hence, we target a non-homogeneous scaling transformation with the following requirements:

- *Modulation:* It should locally control the deviation from a uniform scale based on the level of vulnerability of local regions.
- *Compatibility:* It should preserve shape continuity by using local transformations that agree on shared vertices.
- *Axis alignment:* It should locally preserve orthogonality and avoid rotations, so as to prevent surface distortion.
- *Consistency:* It should obey the global non-uniform scale.

Our method addresses the tradeoff between these possibly conflicting requirements using a two-step procedure, where each step is framed as a quadratic constrained optimization problem, solved using a linear system of equations. The system yields a non-homogeneous transformation defined across the model bounding box that generates the desired global resizing, while preserving the global structure of the model and minimizing the visual deformation of its features (Figure 5).

4 Estimating Vulnerability

Surface vulnerability to nonuniform scale is strongly linked to the direction of the scale. For instance, let's consider a cylinder, it is not vulnerable to scaling along its axis, but is vulnerable, to different degrees, to scaling in other directions. Our directional vulnerability measure is based on estimating the effect a non-uniform scale may have on the model, and is based on two components, *slippage* and *normal curvature*.

Slippage: Slippage analysis [Gelfand and Guibas 2004] estimates surface persistence subject to a given transformation (or transfor-

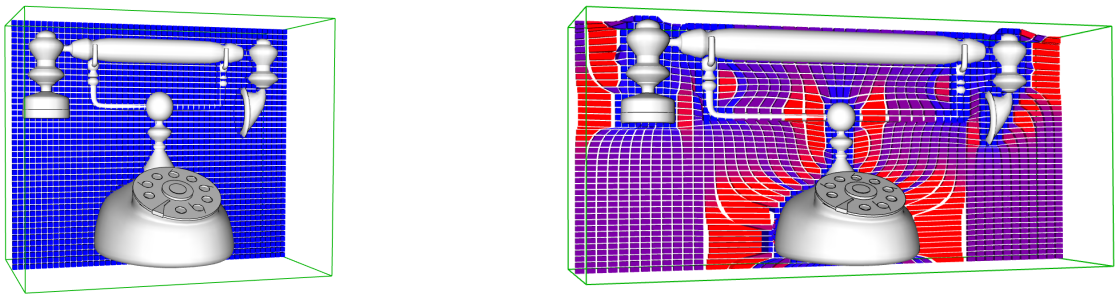


Figure 5: *One slice of the protective grid containing the phone model before (left) and after (right) resizing. The change illustrates how the space deforms non-homogeneously to protect vulnerable parts of the model. Note that the bounding box is not tight to increase flexibility.*

mation type). Specifically, it measures whether a local region on a surface remains on the surface after the transformation is applied. It is easy to see that a surface patch is slippable subject to non-uniform scale, if and only if the surface normal across the patch is perpendicular to the scale axis. On typical scanned or remeshed models slippage can be measured around mesh vertices, by projecting the normals in each vertex's umbrella onto the scaling axis and summing up the projection lengths, possibly with some weighting. However, man-made models, such as the fuel-tank (Figure 6), often contain very large triangles, spanning significant portion of the model, leading to unreliable vertex-based slippage estimation. Most notably, since the meshes are coarse, we rarely obtain absolute zero slippage (consider for instance the vertices bounding the cylindrical part of the canister in Figure 6). Thus, we choose to compute slippage on mesh faces, using the projection of the face normal onto the scale axis as per-triangle slippage measure. To eliminate local singularities we average the face slippage with slippage on neighboring faces using weighting based on the faces' normal similarity and length of shared edges.

Normal Curvature: While slippage analysis can discriminate between vulnerable (non-slippable) and non-vulnerable (slippable) regions, it is not sufficiently discriminative for our purposes. For example, consider two objects: a cone aligned with the scaling axis and a sphere. When stretching the cone, the cone's slope changes while its overall shape is preserved. In contrast, on the sphere every local region is deformed. The slippage metric would classify both surfaces as vulnerable, without clearly distinguishing between the different degrees of vulnerability. The differentiating factor in this and similar scenarios is the normal curvature of the surface in the direction of the scaling axis, which predicts the amount of surface bending subject to the scale. We measure normal curvature at mesh vertices, projecting the scale axis to the surface tangent plane. Prior to the computation, we segment the mesh along sharp creases to obtain more reliable results.

Given a scaling axis u, the per-face vulnerability metric θ^u combines slippage with normal curvature,

$$\theta^u = s^u(\epsilon_\kappa + \kappa^u),$$

where s^u measures per-face slippage for the axis u and κ^u measures the per-face normal curvature, computed as average of vertex curvatures. We add ϵ_κ to the curvature to prevent zero curvature from canceling non-zero slippage (we use $\epsilon_\kappa = 1e^{-4}$). To concentrate the scaling in zero vulnerability regions, we clamp both slippage and normal curvature at fairly small values of 0.1 and $1/(3d)$, respectively, where d is the bounding-box diagonal. We then scale both to a unit range for compatibility. Figure 6 shows the vulnerability of the fuel tank model along the principal axes.

Transfer to Grid: To perform the actual resizing we embed the

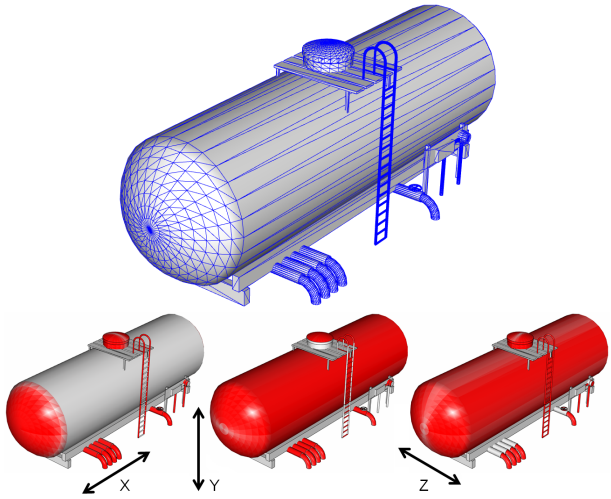


Figure 6: *A 3D fuel tank model and its face vulnerability values in the three major directions - red indicates high vulnerability.*

model in a *protective volumetric grid* (Figure 7). Using a grid-based space-deformation allows us to apply resizing to complex man-made models that often have multiple connected components. Surface-based transformation computation would have difficulties in these cases. In addition, a volumetric structure allows preserving spatial relation among parts which are not necessarily geodetically close. Using surface-based representation, only geodesic proximity can be accounted for. Space deformation alleviates both concerns by implicitly taking spatial relationships into account.

To convert the surface vulnerability values to grid cell values, the vulnerability per grid cell is defined as the maximum vulnerability value of the mesh faces that intersect it. The choice of maximum makes the vulnerability assignment conservative, reducing the likelihood of feature distortion.

5 The Resizing Operator

Given a global resizing transformation $< S^x, S^y, S^z >$, we distribute it among a grid of cells aligned with the resizing axes. Our vulnerability analysis associates each grid cell c with a triplet of scalars $< \theta_c^x, \theta_c^y, \theta_c^z >$ that indicates how vulnerable the cell is to scale along each direction. Using this input, we first compute scale-only transformation gradients for each cell such that the aggregate of all the cells' scales amounts to the global resizing transformation. Given the gradients, the method computes full per-cell trans-

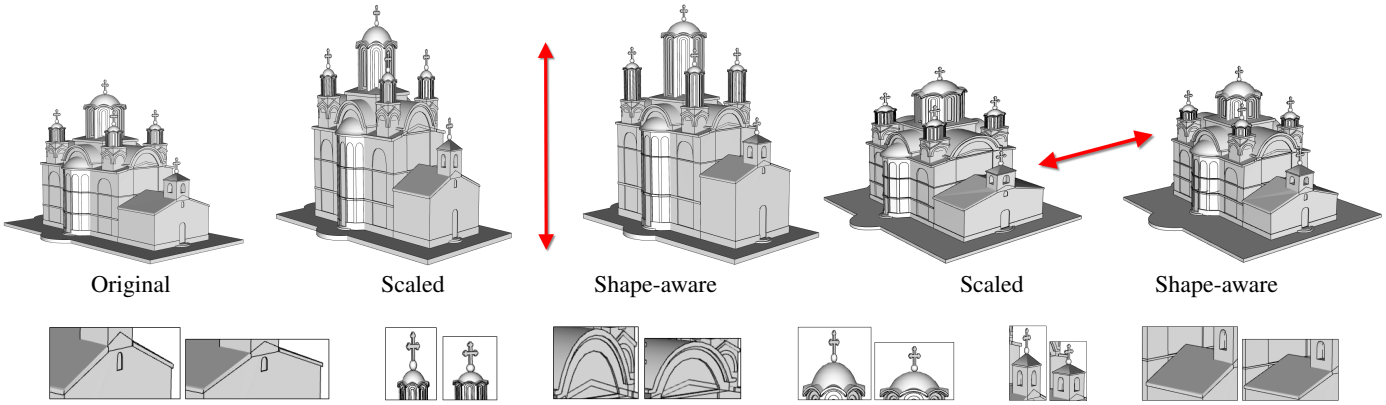


Figure 9: *Scaling a complex church model without destroying its unique architectural style. Note the preservation of details in the close-up pairs (in each pair the left is simple scaling and the right is our approach).*

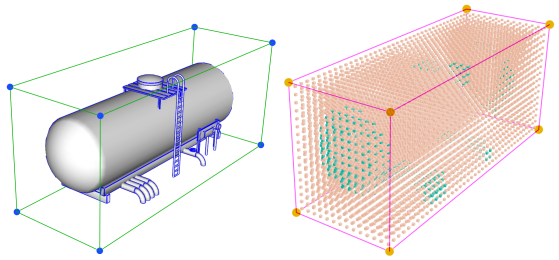


Figure 7: *The volumetric* protective grid *of the fuel tank.*

formations enforcing compatibility between adjacent cells. Finally, in each cell the transformation is carried back to the object using interpolation (Figure 8).

5.1 Cell Scale Calculation

The scaling transformations $< s_c^x, s_c^y, s_c^z >$ for each grid cell should satisfy the aforementioned requirements of modulation, compatibility, axis-alignment and consistency. Since we only consider scales, axis alignment is satisfied by default. Combining the other three requirements, we obtain the per-cell scales as the solution of a constrained minimization problem.

To account for modulation, we consider the scaling of a particular cell with respect to two directions u and v, where $u, v \in \{X, Y, Z\}, u \neq v$. Its resistance to change with respect to these two directions is given by θ_c^u and θ_c^v, respectively. Since globally the grid must scale non-uniformly, the role of θ_c^u and θ_c^v is to measure how much of the non-uniformity can be accommodated by the cell c. We assume that the worst per-cell scale in each direction is given by the corresponding global scale. Note that in reality some cells might scale more than that, since other cells scale less or not at all. However, the global scale gives a good approximation of the worst case. Hence, in the first step we define the acceptable per-cell scale $\tilde{s}_c^u$ and $\tilde{s}_c^v$ for cell c by a linear combination of uniform and global scales weighted by the vulnerability in the given direction:

$$\tilde{s}_c^u = \theta_c^u \cdot 1 + (1 - \theta_c^u) \cdot S^u$$

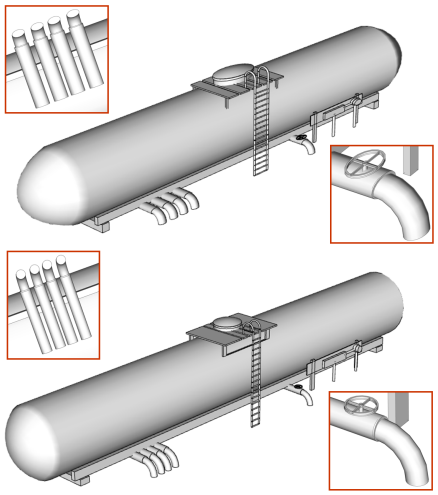


Figure 8: *The fuel tank stretched by a factor of two, (top) simple scale, (bottom) non-homogeneous resizing. Note the preservation of fine details in our result.*

$$\tilde{s}_c^v = \theta_c^v \cdot 1 + (1 - \theta_c^v) \cdot S^v.$$

Since we allow uniform scale, we are interested in *relative* scale or *aspect ratio* between s_c^u and s_c^v rather than their absolute values. Hence, we can minimize:

$$(\frac{s_c^u}{s_c^v} - \frac{\tilde{s}_c^u}{\tilde{s}_c^v})^2.$$

To avoid non-linear optimization, we replace this term by a similar but simpler quadratic expression,

$$L_c(u, v) = (\frac{s_c^u}{\tilde{s}_c^u} - \frac{s_c^v}{\tilde{s}_c^v})^2.$$

The degree to which we want this function to be minimized for each cell depends on how much we want the aspect ratio to be preserved and is a function of the smaller of the two vulnerabilities θ_c^u or θ_c^v. In other words, if the scale in one of the directions is free to change, then there is only minor need for the aspect ratio to be preserved.

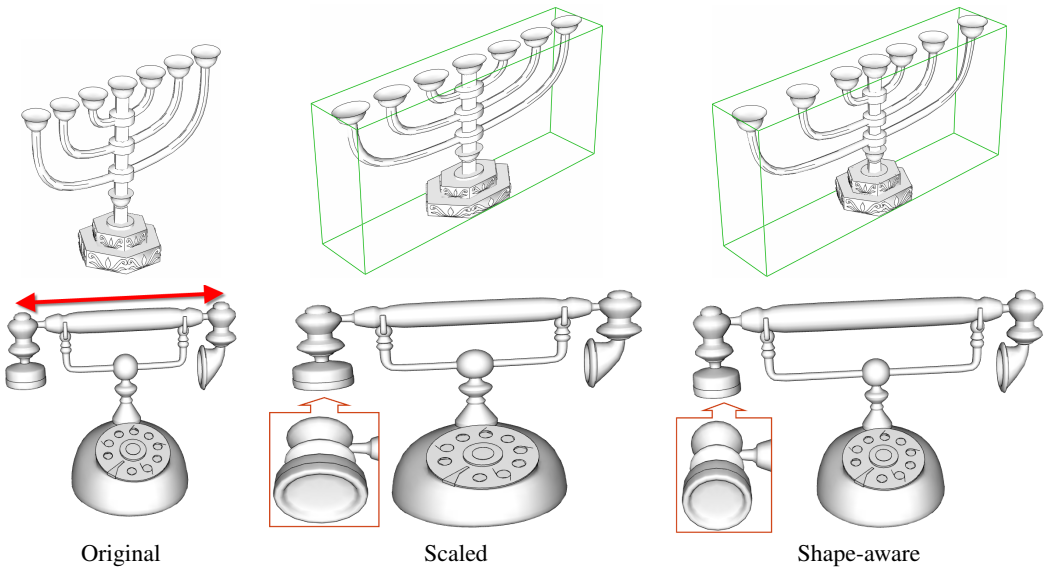


Figure 10: *Resizing decorative models: Stretching a candelabrum consisting of 87 parts and an antique phone consisting of 22 parts. The middle column shows the result of a simple scale. On the right are the non-homogeneous results.*

Assembling all the cells and the three pairwise scales together we obtain:

$$F_m = \sum_c \omega_c^{xy} L_c(x,y) + \omega_c^{yz} L_c(y,z) + \omega^{xz} L_c(x,z),$$

where $\omega_i^{uv} = \max(\min(\theta_c^u, \theta_c^v), 10^{-4})$ controls the distribution of the non-uniform scale through the domain. ω^{uv} is always positive, as even in cells that are free to scale in some direction, excessive scale is undesirable. Rather, we want to distribute the scale more evenly throughout the model when possible, taking advantage of uniform scaling as necessary.

To satisfy the compatibility requirement, adjacent cells must apply the same scale to their shared faces. Given two cells c and d, adjacent along axis w, compatibility requires their scales in the two directions u, v orthogonal to w to be the same:

$$s_c^u = s_d^u$$
$$s_c^v = s_d^v.$$

However, since compatibility cannot always be guaranteed, we phrase it as a least-squares penalty term added to the minimization functional:

$$F_c = \sum_{c,d} \phi_c \phi_d ((s_c^u - s_d^u)^2 + (s_c^v - s_d^v)^2), \qquad (1)$$

where the summation runs over all pairs of adjacent cells, and ϕ_i is an indicator function with value one if the cell i intersects the surface and 0.1 if it does not. This implies that compatibility is more important along the surface. We found that preserving some degree of compatibility outside the model captures better the model's global structure, reducing distortion and enforcing more global changes.

Lastly, to enforce the global scaling, we must require that for each row of cells in any direction, the combined scale is equal to the input scale in that direction:

$$\sum_i s_{i,j,k}^x = S^x, \quad \forall j,k,$$
$$\sum_j s_{i,j,k}^y = S^y, \quad \forall i,k,$$
$$\sum_k s_{i,j,k}^z = S^z, \quad \forall i,j.$$

The desired per-cell scales are obtained by optimizing $F_m + \alpha F_c$ subject to the global scaling constraints. The weight α is set to $4/3$. The resulting linear system is solved using a preconditioned MINRES solver [Toledo 2003]. For our matrix which is of the shape $\begin{pmatrix} A & B^T \\ B & 0 \end{pmatrix}$ we use $\begin{pmatrix} D & 0 \\ 0 & BD^{-1}B^T \end{pmatrix}$, where $D = diag(A)$, as the preconditioner.

5.2 Global Transformation Assembly

After the first step, we obtain $< s_c^x, s_c^y, s_c^z >$ for each grid cell. We now need to combine all cells together and derive the locations of the grid vertices. Had the compatibility constraints been fully satisfied, it would have been enough to scale each cell as specified and simply assemble them, using an iterative process, which sets the appropriate per-cell translations progressively. However, as cell scales might not be fully compatible, we use a least-squares formulation to obtain the coordinates for all the grid vertices at once, while keeping the per-cell scales as much as possible. When compositing the cells together, we want to preserve axis alignment, as shearing the grid can introduce undesirable bending in the resized model. We solve for each coordinate independently, to prevent undesirable rotations. Scale preservation for a direction u is expressed as

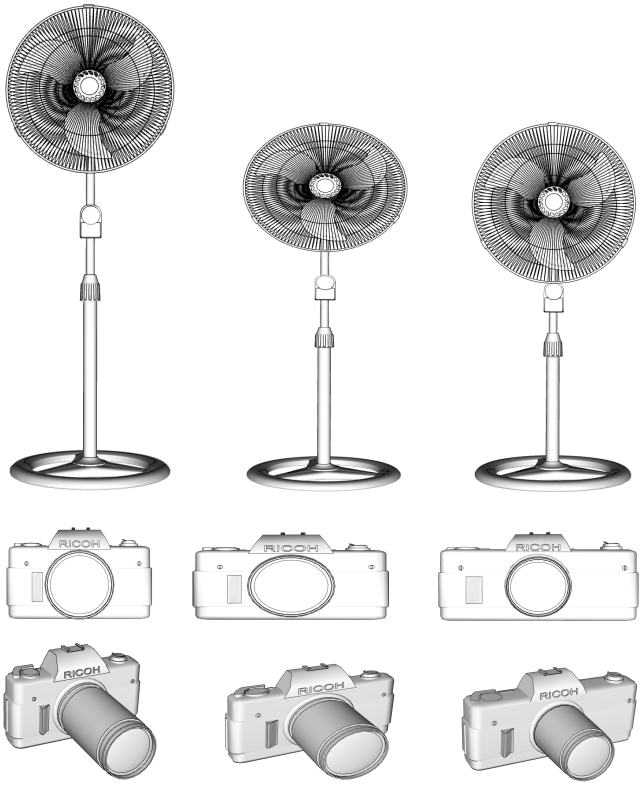
$$F_p = \sum_c \theta_c^u \sum_{(e_1,e_2)\in h} ((e_1^u - e_2^u) - s_c^u)^2,$$

where the index h runs over the four u-aligned edges (e_1, e_2) of

each cell, and θ_c^u is the directional vulnerability. Orthogonality is expressed as follows:

$$F_o = \sum_c \phi_c \sum_{(e_1,e_2)\in o} (e_1^u - e_2^u)^2,$$

where the index o runs over the eight edges orthogonal to u. We minimize $F_p + F_o$, assigning equal weight to scale and orthogonality preservation, subject to enforcing the u values for vertices on the bounding box of the object to match the new scale. The three linear systems obtained are symmetric positive definite and are solved using a conjugate gradient solver with incomplete Cholesky preconditioner [Toledo 2003].



Original Scaled Non-homogeneous

Figure 11: *Preserving structural features when resizing mechanical shapes: shrinking a fan and stretching a camera.*

6 Results

Throughout the paper we demonstrate the application of our method to a large variety of models, including elaborate decorative shapes such as the clock (Figure 1) the candelabrum, and the antique phone (Figure 10); mechanical shapes such as the fan, the camera (Figure 11), the wrench (Figure 12), and the car (Figure 2); and architectural structures such as the church (Figure 9).

As demonstrated by the candelabrum and the clock examples, our method faithfully preserves the intricate shapes of the model features, taking advantage of uniform scale to prevent bending and excessive stretch. It accurately maintains structural features, such as straight lines, parallelism and orthogonality in CAD models such as the camera, the fan (Figure 11) or the wrench. The spatial grid allows for effective distribution of scaling throughout the less vulnerable regions of the models, as demonstrated by the scaling of the church example, where it takes the viewer some time to realize which parts were stretched and which scaled uniformly.

The fountain (Figure 13) is another interesting example, as it contains both man-made and natural shapes (humans). The method automatically detects the vulnerability of the human figures to stretching. By implicitly taking spatial relationships into account in the grid optimization it adjusts the scaling of the rest of the scene to protect the human figures, while preserving the horizontal and vertical fountain structures.

Figure 2 shows an application of resizing in a modeling setup, easily converting the two-seat car to a three-seat or even a four-seat car by simply stretching it and adding more seats.

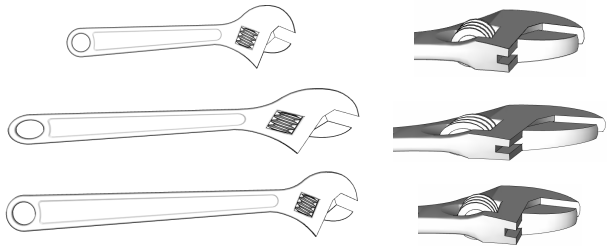


Figure 12: *Stretching a wrench (top) using simple stretch (middle) and non-homogeneous resizing (bottom).*

While the focus of our work is on the resizing of man-made models where the preservation of structure is most critical, our method works on "organic" models as well (Figure 15(top)). However since such models contain less evident structures, the gain is less significant. Another interesting example is shown in Figure 15(bottom). In this case the microscope model is resized in a direction that is misaligned with major model parts. Our resizing method provides intuitive results in such cases as well.

Finally, we note that typical man-made models usually have sufficient variation in vulnerability across the model. This variation can be exploited by our approach for effective non-homogenous resizing. This is true even in cases where no part of the model has absolute zero vulnerability (see Figure 16).

Table 1 summarizes the statistics for the demonstrated models. Most of the models are quite complex, with many having dozens of connected components and fairly large face counts. Interestingly, in our experiments a rather coarse grid is sufficient to capture the vulnerability metric across the models. This is because in our setup the grid resolution need not capture all the model details, but only the local vulnerability. Most of the models were stretched or squashed by factors of 1.5 to 2, as these appear to be typical for realistic resizing scenarios. We ran the experiments on Pentium 4 3.0GHz workstation and measured both the initialization and actual resizing times. The initialization cost includes the vulnerability computation and initialization of the Choleski preconditioners used for global transformation assembly. Initialization is performed only once after the user selects the scaling axes and grid resolution and is independent of the amount of scaling applied to the specified axes. The times for both steps range from around 0.5 seconds to 15 seconds.

Limitations: There are times when parts of the model are not vulnerable to scale in a certain direction, but scaling them non-uniformly adversely affects the object's design. In Figure 14, scaling the Japanese lantern stretches the rims of two lantern parts in an

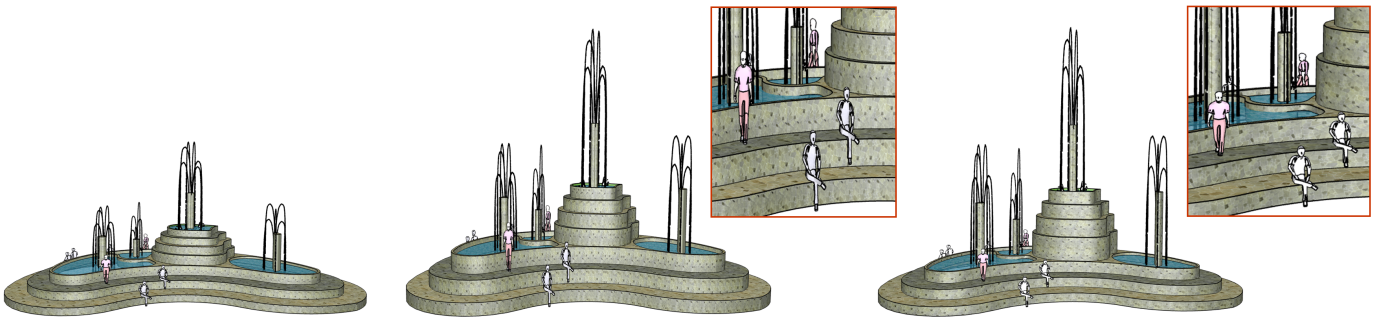


Figure 13: *Fountain model with crowd. In contrast to homogeneous scale which distorts the human figures (middle), our method automatically detects their vulnerability and protects them (right).*

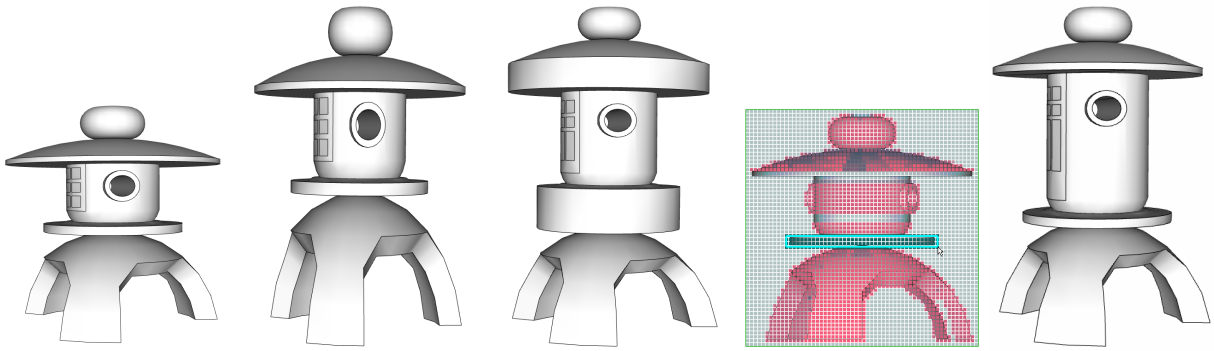


Figure 14: *A Japanese lantern is stretched in an unnatural manner by the automatic method. This can be easily corrected by explicitly protecting the rim regions, using the visualized user interface.*

Model	Num. faces	Num. comp.	Grid resolution	Stretch	Init. time	Run time
Car	29636	40	30,73,35	1.8/2.2	7.3	4.2
Candelabrum	11284	87	80,64,25	1.65	8.8	6.5
Clock	55725	267	50,115,22	1.4	8.7	5.8
Fan	501956	309	20,20,57	0.65	1.5	0.7
Phone	29416	22	50,29,42	1.5	4.5	2.6
Fuel tank	9236	61	40,14,17	2.0	0.8	0.4
Church	39306	129	80,52,50	1.4	15.2	11.8
Camera	6868	29	60,38,56	1.8	9.6	4.8
Wrench	3092	4	60,5,17	1.7	2.5	1.6
Lantern	2125	7	60,61,61	1.5	15.5	10.9
Fountain	42988	766	50,23,38	1.5	2.4	1.5

Table 1: *Model statistics. All times are measured in seconds.*

unnatural manner. To introduce design intent into the vulnerability map, we allow the users to change the vulnerability values on the grid as shown in the figure. Using the new vulnerability map, the method creates a more desirable resized model.

It is important to note that while our method creates models with no visible artifacts, we do not claim that it can precisely preserve the surface structures and curvature in an engineering sense. Intelligent models that exactly preserve complex spatial relations and constraints between parts require high-level reverse engineering.

7 Conclusions

Resizing provides an effective mechanism for creating rich geometric content. However, naive resizing or resizing using existing deformation tools can create visible artifacts. This work presents a novel resizing method that protects geometric features and model structures. We reap the benefits of using a space deformation method which allows dealing with complex models consisting of numerous parts and non-manifold structures. As demonstrated by the examples, the method resizes models in an intuitive manner. Our method augments the set of tools that simplify the creation of 3D geometry, enabling reuse and redesign of existing models.

Our protective grid uses a distortion vulnerability measure defined on 3D objects. We believe that this scheme could be extended to other vulnerability measures which protect other types of features as well. These could include saliency measures, perceptually important quantities, as well as high level information such as symmetry. In a similar manner, the vulnerability measure itself could be useful in other contexts beyond resizing.

Acknowledgments

We gratefully acknowledge the comments of the anonimous reviewers. This work was partially supported by grants from NSERC, MITACS NCE, the Israeli Ministry of Science, and the Israel Science Foundation.

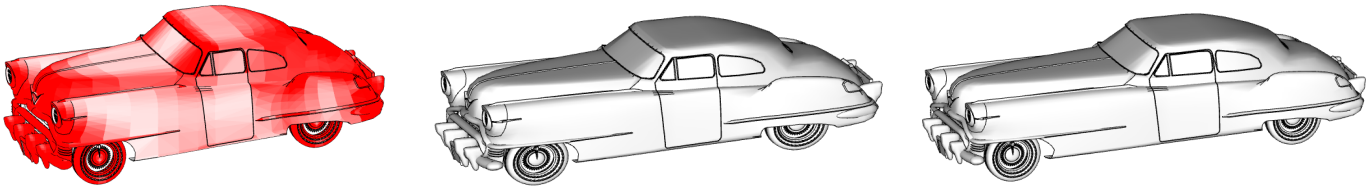


Figure 16: *An example where no part has zero-vulnerability in the scaling direction. From left to right: original model (the coloring expresses the vulnerability), simple homogeneous resizing, non-homogeneous resizing.*

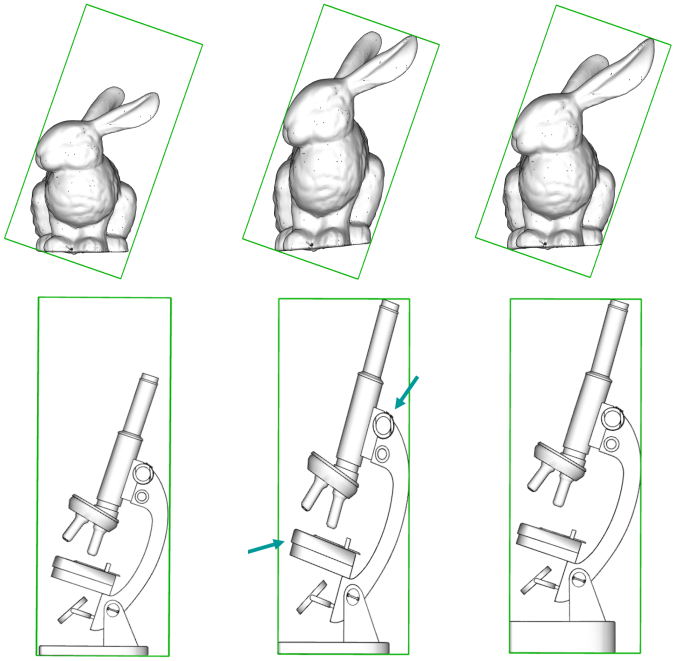


Figure 15: *Additional examples. From left to right: original model, simple homogeneous resizing, non-homogeneous resizing.*

References

ALEXA, M., COHEN-OR, D., AND LEVIN, D. 2000. As-rigid-as-possible shape interpolation. In *Siggraph 2000, Computer Graphics Proceedings*, ACM Press, ACM SIGGRAPH, Addison Wesley Longman, K. Akeley, Ed., 157–164.

ATTENE, M., FALCIDIENO, B., AND SPAGNUOLO, M. 2006. Hierarchical mesh segmentation based on fitting primitives. *The Visual Computer 22*, 3, 181–193.

ATTENE, M., ROBBIANO, F., SPAGNUOLO, M., AND FALCIDIENO, B. 2007. Semantic annotation of 3d surface meshes based on feature characterization. In *Proc. The Second International Conference on Sematic and Digital Media Technologies*.

AVIDAN, S., AND SHAMIR, A. 2007. Seam carving for content-aware image resizing. *ACM Trans. Graph. 26*, 3, 10.

BENKÖ, P., MARTIN, R. R., AND VÁRADY, T. 2001. Algorithms for reverse engineering boundary representation models. *Computer Aided Design 33*, 11, 839–851.

BOTSCH, M., AND SORKINE, O. 2008. On linear variational surface deformation methods. *Transactions on Visualization and Computer Graphics 14*, 1, 213–230.

COQUILLART, S., 1990. Extended free form deformation: a sculptural tool for 3D geometric modelling.

GAL, R., AND COHEN-OR, D. 2006. Salient geometric features for partial shape matching and similarity. *ACM Trans. Graph. 25*, 1, 130–150.

GAL, R., SORKINE, O., AND COHEN-OR, D. 2006. Feature-aware texturing. In *Proceedings of Eurographics Symposium on Rendering*, 297–303.

GELFAND, N., AND GUIBAS, L. J. 2004. Shape segmentation using local slippage analysis. In *SGP '04: Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing*, ACM Press, New York, NY, USA, 214–223.

IGARASHI, T., MOSCOVICH, T., AND HUGHES, J. F. 2005. As-rigid-as-possible shape manipulation. *ACM Transactions on Computer Graphics 24*, 3.

JU, T., SCHAEFER, S., AND WARREN, J. 2005. Mean value coordinates for closed triangular meshes. *Proceedings of ACM SIGGRAPH 2005 24*, 3, 561–566.

LEE, C.-H., VARSHNEY, A., AND JACOBS, D. W. 2005. Mesh saliency. *ACM Transactions on Graphics 24*, 3, 659–666. (Proc. ACM SIGGRAPH 2005).

MASUDA, H., YOSHIOKA, Y., AND FURUKAWA, Y. 2007. Preserving form features in interactive mesh deformation. *Comput. Aided Des. 39*, 5, 361–368.

POPA, T., JULIUS, D., AND SHEFFER, A. 2006. Material-aware mesh deformations. In *SMI '06: Proceedings of the IEEE International Conference on Shape Modeling and Applications 2006 (SMI'06)*.

SEDERBERG, T., AND PARRY, S. 1986. Free-form deformation of solid geometric models. *ACM SIGGRAPH Computer Graphics 20*, 4, 151–160.

SORKINE, O., AND ALEXA, M. 2007. As-rigid-as-possible surface modeling. *Proceedings of the fifth Eurographics symposium on Geometry processing*, 109–116.

THOMPSON, D. W. 1942. *On Growth and Form. The Complete Revised Edition*. Reprinted by Dover, New York, 1992.

TOLEDO, S., 2003. Taucs: A library of sparse linear solvers, version 2.2, http://www.tau.ac.il/ stoledo/taucs. http://www.tau.ac.il/ stoledo/taucs.

VÁRADY, T., AND MARTIN, R. R. 2002. Reverse engineering. In *Handbook of Computer Aided Geometric Design*, G. Farin, J. Hoschek, and M. S. Kim, Eds. Springer, 651–681.

WOLF, L., GUTTMANN, M., AND COHEN-OR, D. 2007. Non-homogeneous content-driven video-retargeting. In *Proceedings of the Eleventh IEEE International Conference on Computer Vision (ICCV-07)*.