Pacific Graphics 2016
E. Grinspun, B. Bickel, and Y. Dobashi
(Guest Editors)

Volume 35 (2016), Number 7

Retargeting 3D Objects and Scenes with a General Framework

Chun-Kai Huang[1] Yi-Ling Chen[2] I-Chao Shen[3] Bing-Yu Chen[1]

[1]National Taiwan University [2]University of California, Davis [3]University of British Columbia

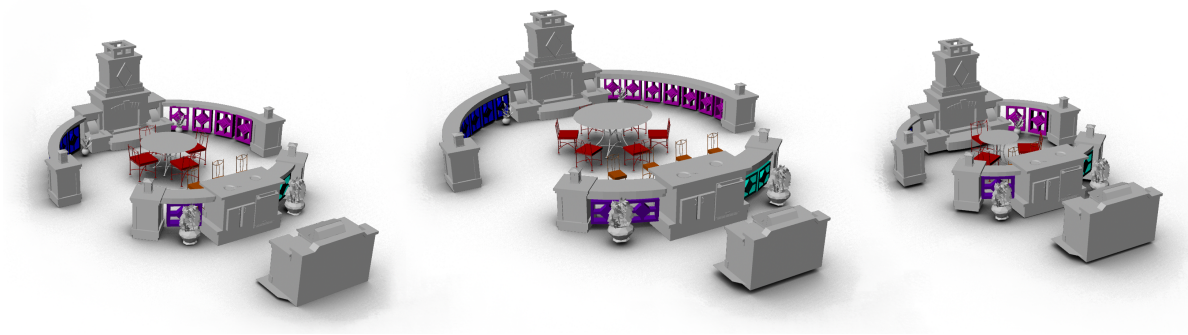


Figure 1: Given an input scene (left), the proposed method generates adaptive retargeting result (middle, right) which respect the original spatial arrangement by exploiting structural regularity. The color-coded parts are the detected regular patterns representing the semantic groupings of the objects in the scene. Our method can accomplish both object- and scene-level retargeting simultaneously and well preserve the underlying structures. The figures in this paper are best viewed in color/screen and significantly zoomed in.

Abstract

In this paper, we introduce an interactive method suitable for retargeting both 3D objects and scenes. Initially, the input object or scene is decomposed into a collection of constituent components enclosed by corresponding control bounding volumes which capture the intra-structures of the object or semantic grouping of objects in the 3D scene. The overall retargeting is accomplished through a constrained optimization by manipulating the control bounding volumes. Without inferring the intricate dependencies between the components, we define a minimal set of constraints that maintain the spatial arrangement and connectivity between the components to regularize the valid retargeting results. The default retargeting behavior can then be easily altered by additional semantic constraints imposed by users. This strategy makes the proposed method highly flexible to process a wide variety of 3D objects and scenes under an unified framework. In addition, the proposed method achieved more general structure-preserving pattern synthesis in both object and scene levels. We demonstrate the effectiveness of our method by applying it to several complicated 3D objects and scenes.

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

1. Introduction

In recent years, many structure-aware shape processing and editing techniques [GSMCO09, BWKS11, ZFCO*11, BWSK12] have attracted a lot of attentions in computer graphics. These techniques exploited structural properties, including *regularity* [PMW*08] and *symmetry* [MGP06, BBW*09], to easily alter existing 3D shapes while preserving their structures. Although being successful, most of these techniques are designed to work in "object-level", which mainly aims to faithfully preserve and reproduce the salient features of individual 3D objects.

A 3D scene can be regarded as a collection of 3D objects, which are often arranged by a set of implicitly defined rules for a multitude of purposes such as functional or aesthetic constraints. To retarget a 3D scene, the "object-level" methods are not directly applicable due to the necessity of inferring the unknown dependencies between the objects in the scene. Currently, there is still no method which can achieve object and scene retargeting simultaneously.

To faithfully edit a 3D scene, our goal is to provide a general retargeting framework that is structure-preserving in "object-level" and respects the original *arrangement* or *layout* in "scene-level". Initially, an input object or scene is decomposed into a collection of constituent components enclosed by corresponding *control bounding volumes*. The overall retargeting result is then obtained through a constrained optimization by manipulating the control bounding volumes. The idea of the volumetric controller (*e.g.*, axis-aligned boxes) is not new and has been exploited in object-level shape retargeting for man-made objects [ZFCO*11] or architecture models [LCOZ*11]. Local deformations are performed within individual controllers and the influences are propagated over the whole model to achieve global deformation. Differing from the previous methods, our volumetric representation is an abstraction of space decomposition since it is used to capture not only the intra-structures (*e.g.*, regular patterns) in the "object-level", but also the semantic grouping of the objects (*e.g.*, a row of trees) in the "scene-level".

Without inferring the intricate dependencies between the components, like [FSH11, XMZ*14], we define a minimal set of constraints that maintains the spatial arrangement and connectivity between the components to regularize the valid retargeting results. The default retargeting behavior can be easily modified by additional semantic constraints imposed by users. This strategy makes the proposed method highly flexible to process a wide variety of 3D objects and scenes under a general framework. In addition, it leads to an interactive tool that enables users to freely explore the space of valid solutions and gain fine control over the retargeting results by adding additional semantic configurations. In spite of its simplicity, the proposed method achieved more general structure-preserving pattern synthesis (*e.g.*, *rotational* patterns lacked in [BWSK12]) in both object- and scene-levels.

To summarize, the major contributions of this work are two-fold:

1. To the best of our knowledge, our method is the most general regularity-based 3D editing method in terms of the types of supported structural regularity. With the least assumptions of the underlying structures, the extracted regular patterns are inserted into control bounding volumes which facilitate user manipulation and achieve structure-preserving shape editing, resulting in a simple and general method.
2. The proposed method provides a general and unified 3D editing framework. We extend the notion of regularity analysis to the level of 3D scenes and demonstrate that the editing of 3D objects and scenes can be identically formulated as a constrained optimization problem. The resulting framework is highly flexible and configurable. In addition, it allows users to obtain desired editing results within a valid solution space regularized by carefully designed default and user constraints.

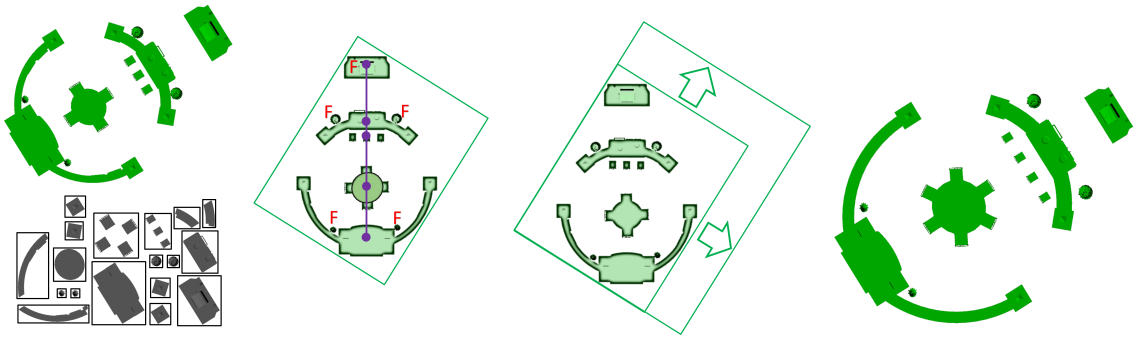
The rest of this paper is organized as follows. In Section 2, we surveyed several previous methods for structure analysis, shape editing and content creation of models and scenes. Section 4 and Section 5 describe the two main stages of our method: *scene analysis* and *scene editing*, respectively. In Section 6, we demonstrate several results of our method, compare with those of previous work and discuss the limitation of our approach. Finally, Section 7 concludes this paper.

2. Related Work

Structural analysis includes automatic detection of symmetries, regularity and repetitive structures in 3D objects and complex scenes. Regularity detection can be treated as a process that finds similar intra-shapes and a series of transformations which could be used to generate structural patterns. Several techniques that detect symmetries in 3D objects have been proposed, as surveyed in [MPWC13]. Pauly *et al.* [PMW*08] proposed to detect and extract regular patterns, and the structure discovery is performed by collecting patch similarities and analyzing pairwise transformation. Bokeloh *et al.* [BBW*09] proposed an efficient algorithm based on feature line matching to find rigid symmetries in general configurations by avoiding the transformation space clustering problem in [MGP06]. Although this method is both computation and memory efficient, it fails to extract rotational information from the feature lines.

Structure-preserving shape editing [MWZ*13] needs a high-level shape analysis process which often involves the extraction of the potential intra-structures or regular patterns embedded in 3D shapes. A considerable amount of research efforts have been made to develop various algorithms to understand such structures and manipulate 3D shapes based on them. Non-uniform shape deformation approaches often divide an input model into independent parts and process them differently according to vulnerability analysis [KSSCO08], mesh segmentation [XLZ*10], and descriptive set of proxies [GSMCO09, ZFCO*11]. Alhashim *et al.* proposed a shape stretching algorithm which is capable for reconstructing surface details, though high-level structural knowledge is not taken into account [AZL12]. Wang *et al.* introduced a hierarchical representation of man-made objects which encodes symmetry relationship between a model's constituent parts and demonstrated its application of structural shape editing [WXL*11]. Bokeloh *et al.* proposed various structure-aware shape editing techniques that adaptively insert or remove discrete regular patterns by algebraic models [BWSK12] or *sliding dockers* [BWKS11]. Milliez *et al.* extended the sculpting paradigm to structured shape modeling and employed a hybrid optimization scheme to improve stretching results when editing [MWCS13]. However, the real architectures and man-made objects are often more complicated due to the circular and scale patterns. Our method is able to detect these kinds of patterns and manipulate the structures.

Shape and scene synthesis methods are used for content creation and layout design. These methods take advantage of exemplar shapes to generate large amount of shape variations by using procedural modeling [PM01, WWSR03, MWH*06, YYT*11, TLL*11, VGDA*12, BSW13], inverse procedural modeling [BWS10, TYK*12, KWS16], and probabilistic model and inference [CKGK11, KCKK12, FRS*12, MSK10, YYW*12, PYW14, LVW*15, GJWW15]. In spite of the diversity of the mathematical models behind these approaches, they all aim to solve an essential problem: adjacency relationships between the exemplar shapes. Briefly speaking, most approaches seek for compatible ways to connect and rearrange the exemplar shapes to synthesize novel models or scenes, and have been intensively applied for various applications, such as architecture [WWSR03, MWH*06, MSK10] and urban modeling [PM01, YYW*12, TLL*11, VGDA*12], and fur-



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

Figure 2: Algorithmic flow of the proposed method. (a) Given an input scene $\mathcal{M}$, the structural analysis is performed to decompose $\mathcal{M}$ into a collection of control bounding volumes $\mathcal{V}$. (b) Users can associate $\mathcal{V}$ with additional semantic constraints ("F" means "fixed" and the purple line is an auxiliary plane for alignment). Note that only the global bounding volume $\hat{\mathcal{V}}$ is shown here. (c) By manipulating the global bounding volume $\hat{\mathcal{V}}$, the contents of $\mathcal{V}$ are updated to form the overall retargeting result in (d). Users may also edit the individual local bounding volumes to obtain various retargeting results.

niture [YYT*11] and façade design [BSW13, LW15]. Traditional procedural modeling methods [WWSR03, BWS10, TLL*11] are capable of generating a wide variety of models that are locally similar to the input shape. However, they mostly focus on single models that are composed of physically connected exemplar shapes and do not consider the possible deformations of the exemplar shapes.

Another line of researches exploit various auxiliary structures, such as rectangles [MSK10], quad-meshes [PYW14] or bounding boxes [BSW13,LCOZ*11], to facilitate layout generation or façade design. The exemplar shapes usually need to undergo appropriate deformations to fit to the auxiliary structures. Our method also utilizes a set of carefully constructed auxiliary structures, *i.e.*, control bounding volumes to facilitate the retargeting tasks, such as maintaining the structural regularity of the input models or scenes.

3. Overview

The input of our system is typically a polygonal mesh $\mathcal{M} \in \mathbb{R}^3$ representing the geometry of a source *object* or *scene*. For the case of a 3D scene, $\mathcal{M}$ can be further divided into n objects $\mathcal{M}_i \in \mathcal{M}$, $i = 1 \ldots n$, with unknown mutual relationship. Without loss of generality, we will hereinafter focus the discussion on scene retargeting. As will become clear soon, object retargeting is a special case in our method among which the problem is reduced to a scene containing only one object (*i.e.*, $n = 1$ and $\mathcal{M} \equiv \mathcal{M}_1$). Our goal is thus to retarget $\mathcal{M}$ into $\mathcal{M}'$ in a way such that $\mathcal{M}'$ preserves the original structures of all constituent objects $\mathcal{M}_i$ and maintains the spatial layout between $\mathcal{M}_i$. Following the *analyze-and-edit* paradigm [GSMCO09], the proposed method is composed of two main stages: *scene analysis* (Section 4) and *scene editing* (Section 5).

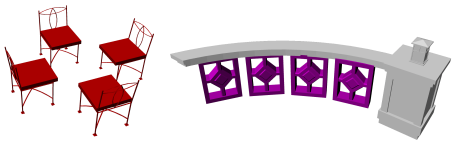
The goal of scene analysis is to encapsulate $\mathcal{M}$ into a set of control bounding volumes $\mathcal{V}_j \in \mathcal{V}$ capturing the sub-structures within $\mathcal{M}$ or semantic grouping of multiple $\mathcal{M}_i \in \mathcal{M}$. In this work, we mainly consider the *structural regularity* [MGP06, PMW*08] presented in 3D models (Section 4.1). Once $\mathcal{V}_j$ are properly constructed (Section 4.2), we can then model valid retargeting results by solving a constrained optimization problem. To keep generality, we exploit a minimal set of constraints required to keep the retargeting results visually natural and physically valid (Section 5.1). Specifically, a set of *positional constraints* are derived from the spatial arrangements of $\mathcal{M}_i$ and used to keep the relative order between them. Another set of *anchor constraints* accounts for the connectivity between $\mathcal{M}_i$. In the scene editing stage, we provide an easy-to-use system that allows users to interactively explore valid scene variations confined to the default constraints by manipulating $\mathcal{V}_j$. The default retargeting behavior can be easily modified by imposing additional constraints interactively (Section 5.2). Finally, the modified $\mathcal{V}_j$ are used to induce the overall scene synthesis results (Section 5.3).

4. Scene Decomposition and Analysis

In this section, we explain the construction of the main auxiliary structures, *i.e.*, control bounding volumes $\mathcal{V}_j \in \mathcal{V}$, which are used to facilitate adaptive scene retargeting. In this work, $\mathcal{V}_j$ is just a simple minimal bounding box with its main axes aligned to the x-, y- and z-axis of Euclidean space, which encloses the underlying 3D shape. Unlike previous methods [LCOZ*11, ZFCO*11], our $\mathcal{V}_j$ are not necessarily to capture physically connected geometric entities. Besides, $\mathcal{V}_j$ are also not required to be best fitted to the enclosed structures, as shown in the examples of Figure 2(a) and Figure 2(b). Each $\mathcal{V}_j$ is associated with a set of parameters $(\mathbf{o}_j, w_j, h_j, d_j)$, where $\mathbf{o}_j$ is the origin of $\mathcal{V}_j$, and w_j, h_j, d_j indicate the lengths of the three main axes of $\mathcal{V}_j$, respectively. As an user manipulates $\mathcal{V}_j$, the corresponding parameters change according to the type of structure contained in $\mathcal{V}_j$ and influence the retargeting results.

4.1. Structural Regularity Detection

Instead of detecting the structural regularity of a single model, we extract the regular patterns from $\mathcal{M}$ in both object- and scene-



(a) (b)

Figure 3: Examples of sub-shapes enclosed by the control bounding volumes. (a) semantic group of objects, (b) repeating regular sub-structures.

levels. Conceptually, the regularity detection is applied to all $\mathcal{M}_i \in \mathcal{M}$ and $\mathcal{M}$, separately. By this way, not only the sub-structures among $\mathcal{M}_i$ but also the groups of $\mathcal{M}_i$ resembling the regular patterns will be identified.

Following [PMW*08], we firstly perform uniform down-sampling on the 3D point cloud of $\mathcal{M}$. These samples are then clustered into different groups according to their mean and Gaussian curvatures. For each group, pairwise transformation (*e.g.*, translation or rotation) of every pair of points are estimated. If the point cloud constitutes a certain type of regular patterns, many identical pairwise transformations will be observed. All the estimated transformations are then mapped to another transformation space, where the identical transformations will be accumulated and form an uniformly-spaced grid. Finally, grid fitting is performed to extract the underlying regularity and estimate its parameters accordingly. More details regarding the regularity detection algorithm are referred to [MGP06, PMW*08].

For each detected regular pattern, we extract the corresponding 3D points from the input mesh, and randomly select one element as the base shape to generate the retargeting results. After all patterns are processed, the remaining 3D shapes are classified as rigid parts whose size can only be uniformly scaled during the retargeting process. Some examples of regularity detection are shown in Figure 1.

Based on the regularity detection, we represent each distinct regular pattern $\mathcal{P}_k$ as the following parametric form: $(\mathbf{c}_k, n_k, \mathcal{T}_k)$, which indicate the center of the starting pattern $\mathcal{P}_k^0$, *i.e.*, the base shape, the number of pattern element repetitions, and the generator transformation, respectively. We compute $\mathbf{c}_k$ as the mean position of the vertices of $\mathcal{P}_k^0$. Three types of transformations are considered, *i.e.*, scaling, rotation, and translation. According to the transformation types, the parameters of $\mathcal{T}_k$ are expressed by $(s, \mathbf{t}, \theta)$, which correspond to the scaling factor, translation vector and rotation angle. For rotational patterns, an additional parameter $\hat{\mathbf{c}}$, *i.e.*, rotation center, needs to be derived from the pattern elements.

Based on the pattern definition of [PMW*08], the helix- or spiral-like patterns can be generated by combining two transformations, *i.e.*, $Rot + Trans$, where $+$ denotes the combination of two transformations. The grid-structured patterns can be generated by a *commutative two-parameter group*, *i.e.*, $Trans \times Trans$, which is formed by two independent translations.

Regularity type	Size fixation	Ratio fixation
Rigid	w,h,d *or user define*	w,h,d *or user define*
Trans	two of w,h,d	–
Trans∗	one of w,h,d	the rest of w,h,d
Scale	–	two of w,h,d
Rot	one of w,h,d	the rest of w,h,d
$Rot + Trans$	–	two of w,h,d
$Rot \times Trans$	–	two of w,h,d
$Trans \times Trans$	one of w,h,d	–
$Trans \times Trans*$	–	two of w,h,d

Table 1: Regularization rules of $\mathcal{V}$ for various regularity types [PMW*08]. w,h,d indicate the lengths of $\mathcal{V}$ and are interchangeable among the same row of the table. Take *Trans∗* as an example, if h is fixed, then w and d are free to be adjusted while their ratio must remain constant.

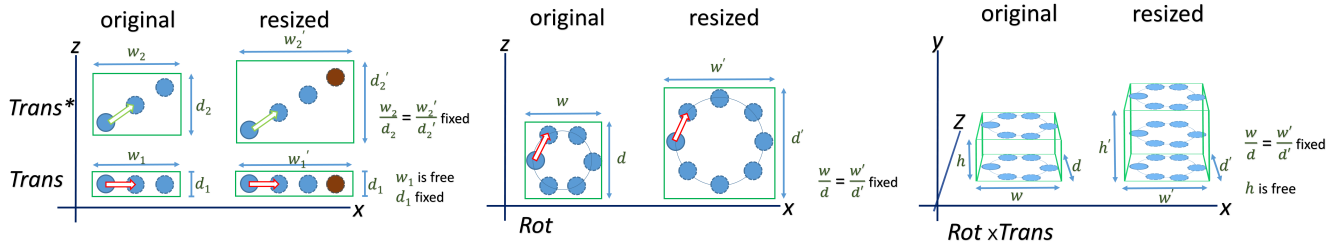
4.2. Control Bounding Volume Construction

Given a set of regular patterns $\mathcal{P}_k$ detected in the analysis phase, we thereby decompose $\mathcal{M}$ into a collection of constituent components enclosed by corresponding control bounding volumes $\mathcal{V}_j$ through the following steps:

1. For any $\mathcal{P}_k$, where $\mathcal{P}_k^0 \equiv \mathcal{M}_i$, create a bounding volume $\mathcal{V}_j$ to enclose all elements of $\mathcal{P}_k$, *i.e.*, , $\mathcal{P}_k^0 \ldots \mathcal{P}_k^{nk}$. In this case, $\mathcal{P}_k$ is composed of a subset of $\mathcal{M}_i$, $i = 1 \ldots n$, which are scattered around the scene, such as the chairs surrounding the table as shown in Figure 1 and Figure 3(a). Note that when $\mathcal{M}$ is a single object, there will not be this type of bounding volume.
2. For those $\mathcal{P}_k$, where $\mathcal{P}_k^0 \neq \mathcal{M}_i$ and $\mathcal{P}_k^0 \subset \mathcal{M}_i$, extract all elements of $\mathcal{P}_k$ from $\mathcal{M}_i$ and insert them into a new bounding volume $\mathcal{V}_j$, such as the shape in purple shown in Figure 3(b). Repeat the above procedure until all $\mathcal{P}_k$ are associated with a corresponding bounding volume.
3. After all $\mathcal{P}_k$ are processed, some objects $\mathcal{M}_i$ may still have remaining parts $\Delta\mathcal{M}_i$, such as the shape in grey shown in Figure 3(b), which are not contained in any bounding volume. We then create a bounding volume for every separate component in $\Delta\mathcal{M}_i$. In addition, the components in $\Delta\mathcal{M}_i$ are conditionally split into smaller disjoint pieces in order to minimize the overlapping of the corresponding $\mathcal{V}_j$. In the example shown in Figure 3(b), the bounding volume of the shape in gray contains that of the shape in purple. We thus search for a cutting plane to split the shape in gray such that the overlapping between the newly created bounding volumes are minimized. This splitting process is beneficial because it allows a wider range of movements between physically interconnected components.

Through the above procedure, we can then obtain our control bounding volumes $\mathcal{V} = \{\mathcal{V}_j \mid j = 1 \ldots m\}$. Note that a global bounding volume $\hat{\mathcal{V}}$ enclosing all $\mathcal{V}_j$ is also included into $\mathcal{V}$ to enable editing such as resizing the whole scene.

Bounding volume regularization: As explained earlier, we use minimum enclosing boxes to capture various structural regularities. In spite of its simplicity, such generalization enables us to deal with a wider range of regularity types comparing with those of previous



(a) (b) (c)

Figure 4: Illustrations of bounding volume regularization. (a) *Trans* and *Trans*$*$, (b) *Rot*, and (c) *Rot* $\times$ *Trans*. The $*$ sign indicates that the translation vector $\mathbf{t}$ is not aligned with any axis of the bounding volume.

methods, such as axis-aligned [LCOZ*11] and translational pattern synthesis [BWSK12]. Nevertheless, when manipulating $\mathcal{V}_j$, we still need a set of regularization rules to properly define the space of valid volume sizes in order to prevent from undesired results, such as squeezing $\mathcal{V}_j$ into thin plates. It is an important feature to an interactive editing tool like the proposed method.

The main consideration of the bounding volume regularization is to keep $\mathcal{V}_j$ as a minimal enclosing box after resizing, which is crucial to the semantic alignment process described in Section 5.2. Table 1 summarizes the regularization rules that we defined for different types of regular patterns and Figure 4 illustrates several cases among them. For example, as shown in Figure 4(a), when manipulating an axis-aligned translational pattern, the corresponding $\mathcal{V}_j$ is only allowed to be stretched along the direction of $\mathbf{t}$ while the other two dimensions remain constant. As for non-axis-aligned translational patterns, $\mathcal{V}_j$ can be stretched along the directions of the two components of $\mathbf{t}$ while the aspect ratio remains constant. Some other types of regularity are also shown in Figure 4(b) and Figure 4(c) and supplemental. Refer to the accompanying supplemental materials for more details.

5. Interactive Scene Retargeting

After the input scene is encapsulated into the control bounding volumes $\mathcal{V}$, users can then directly operate on $\mathcal{V}_j$ to obtain the desired retargeting results. To achieve this, the unknown parameters $(\mathbf{o}'_j, w'_j, h'_j, d'_j)$ associated with a modified $\mathcal{V}'_j$ need to be determined, which is formulated as a least square optimization problem with a default set of linear constraints that assist to maintain physical connectivity and spatial layout. The space of valid scene variations can be easily altered by interactively imposing additional semantic constraints.

5.1. Default Constraint Setup

Denote a point $\mathbf{p}$ lying within $\mathcal{V}_j$ as $(u, v, \omega) \in [0, 1]$, which is the local coordinate system of $\mathcal{V}_j$ spanned by its three axes and can be mapped to its corresponding position $\mathbf{x}$ in global coordinate system by the following transformation function C:

$$\mathbf{x} = C_j(\mathbf{p}) = \mathbf{o}_j + (u \cdot w_j,\ v \cdot h_j,\ \omega \cdot d_j). \tag{1}$$

For all the retargeting operations, we want to let $\mathbf{p}$ remain constant while its corresponding 3D position is updated by solving for new parameters of $\mathcal{V}'_j$.

Anchor constraints: To obtain visually plausible retargeting results, anchor constraints are exploited to enforce physical connectivity between subdivided sub-shapes among $\mathcal{M}_i$. During the construction of $\mathcal{V}_j$, we search and record the vertices shared by different volumes when extracting or splitting sub-shapes of $\mathcal{M}_i$. Denote the set of common vertices among two bounding volumes $\mathcal{V}_1$ and $\mathcal{V}_2$ as V. For every point in V, let $\mathbf{p}_1$ and $\mathbf{p}_2$ be the points represented by its local coordinates in $\mathcal{V}_1$ and $\mathcal{V}_2$, respectively. The anchor constraint can then be expressed by the following linear equation,

$$E_c = ||C_1(\mathbf{p}_1) - C_2(\mathbf{p}_2)||^2. \tag{2}$$

Note that not all vertices in V are required to be included in the least square optimization. This is because the updated $\mathcal{V}'_j$ will recompute all the remaining vertices with the new parameters which act like applying a similarity transformation to the sub-shape enclosed by $\mathcal{V}'_j$. As a result, we sparsely select a small portion of V to form the final linear system.

It is also worth noting that some real-world 3D scenes are composed of many components that are visually adjacent but physically disconnected because they are typically created by importing 3D objects from existing shape libraries. Several examples of such situations are illustrated in Figure 5. The television and wall look connected but share no common vertex in reality. On the other hand, the two rows of chairs apparently are aligned to the opposite sides of the dining table even though they are not connected. We treat such mutual relationships as implicit connectivity or alignment constraints. Specifically, for each pair of adjacent bounding volumes $\mathcal{V}_1$ and $\mathcal{V}_2$, if the enclosed 3D shapes possess physical contact with each other while no common vertex can be found, some additional anchor points are created to enforce physical connection (as shown in the red box of Figure 5). Otherwise, an additional *alignment* constraint, which will be explained later, is imposed on the closet faces of $\mathcal{V}_1$ and $\mathcal{V}_2$ (as shown in the blue box of Figure 5).

Positional constraints: The goal of the positional constraints is to roughly maintain the spatial arrangement between $\mathcal{M}_i$ when modifying the global volume $\hat{\mathcal{V}}$. In this work, we do not explicitly infer the mutual relationship between $\mathcal{M}_i$ and simply use the volume centers as reference points to place the modified $\mathcal{V}'_j$. The center of

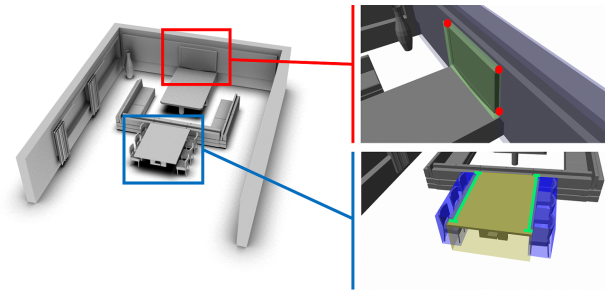


Figure 5: Illustration of analyzing the implicit mutual relationship between neighboring bounding volumes. In the region highlighted by the red box, the television and wall are physically contacted but share no common vertex. Several virtual anchor points (red dots) are created to maintain the connectivity. In the region highlighted by the blue box, our system automatically aligns the bounding volumes of the dining table and chairs since their opposite faces are almost coincided with each other.

$\mathcal{V}_j$ can be expressed by $C_j(\bar{\mathbf{c}}_j)$, where $\bar{\mathbf{c}}_j = (0.5, 0.5, 0.5)$. Since $\mathcal{V}_j$ are also enclosed in $\hat{\mathcal{V}}$, $\bar{\mathbf{c}}_j$ can also be expressed by the parametric form of the local coordinate system of $\hat{\mathcal{V}}$. Assume that the relative position of $C_j(\bar{\mathbf{c}}_j)$ in the global bounding volume $\hat{\mathcal{V}}$ is $\bar{\mathbf{c}}'_j$. For every $\mathcal{V}_j$, we thus impose a linear constraint as follows,

$$E_p = \sum_j^{\mathcal{V}} ||\hat{C}(\bar{\mathbf{c}}'_j) - C_j(\bar{\mathbf{c}}_j)||^2. \tag{3}$$

Intuitively, this type of constraints encourages $\mathcal{V}_j$ to adhere to the same relative position with respect to $\hat{\mathcal{V}}$ after retargeting. Since w_p is set to a relatively small value, the original layout can still be broken and users can obtain finer control over the positioning of $\mathcal{V}_j$ by adding more interactive or semantic constraints, as will be explained below.

It is worth noting that we do not explicitly consider symmetry relationship in this work. However, the positional constraints implicitly keep symmetric relationship when users only modify the global bounding volume $\hat{\mathcal{V}}$.

5.2. User Interaction

To understand the semantic meanings or contextual information of a 3D scene is very difficult. For example, when retargeting the scene shown in Figure 1, it might be more desirable to keep the chairs with a constant size, but it is also hard to be determined automatically. It is thus our strategy to rely on moderate user inputs to provide such information and focus on designing a flexible tool to meet the common needs of 3D scene retargeting.

Semantic constraints: In this work, we support two types of semantic constraints, *i.e.*, *scaling* and *alignment*. By default, we allow each $\mathcal{V}_j$ to be proportionally scaled with respect to $\hat{\mathcal{V}}$:

$$E_r = \sum_j^{\mathcal{V}} \sum_{a,b \in \{w,h,d\}} R_j(a,b) || \frac{L_j(a)'}{L_j(b)'} - \frac{L_j(a)}{L_j(b)} ||^2. \tag{4}$$

where L_j returns the width, height or depth of $\mathcal{V}_j$. This retargeting behavior can be modified by interactively specifying a bounding volume to fix its size by the following constraint:

$$E_f = \sum_j^{\mathcal{V}} \sum_{a \in \{w,h,d\}} S_j(a) ||L_j(a)' - L_j(a)||^2, \tag{5}$$

S_j and R_j are indicator functions which returns either 0 or 1 according to the regularization rules defined in Table 1 or the semantic constraints imposed by users. In Figure 2, a bounding volume annotated as "F" (fixed) indicates that its size is configured to be fixed.

An alignment constraint is particularly useful when one wants to enforce two volumes $\mathcal{V}_1$ and $\mathcal{V}_2$ to be aligned by a plane intersecting their centers or one of the six faces. Assuming that an auxiliary plane l is defined by a normal vector $\mathbf{n}_l$, and $\mathbf{d}_l$ is the distance between the origin and l. In the case of *align-by-center*, it can be achieved by imposing the following linear constraints:

$$E_a = ||\mathbf{n}_l \cdot C_1(\bar{\mathbf{c}_1}) - \mathbf{n}_l \cdot C_2(\bar{\mathbf{c}_2})||^2. \tag{6}$$

subject to

$$\begin{cases} \mathbf{n}_l \cdot C_1(\bar{\mathbf{c}_1}) - \mathbf{d}_l = 0 \\ \mathbf{n}_l \cdot C_2(\bar{\mathbf{c}_2}) - \mathbf{d}_l = 0. \end{cases} \tag{7}$$

The above constraints encourages the centers of $\mathcal{V}_1$ and $\mathcal{V}_2$ to be adhered to the same 3D plane, thus resulting in the effect of alignment. One example demonstrating the effect of alignment constraints is the outdoor scene shown in the bottom of Figure 7.

Interactive constraints: In this category of constraints, two types of user operations are supported, *i.e.*, *displacement* and *stretching*. The displacement constraints can be regarded as a variant of positional constraints with the reference point replaced with a specific location $\mathbf{v}$ where a user would like to move $\mathcal{V}_k$. It is particularly useful to apply displacement constraints to adjust the position of an individual object. Specifically, for every $\mathcal{V}_k$ that users edited,

$$E_i = \sum_k ||C_k(\bar{\mathbf{c}}_k) - \mathbf{v}||^2. \tag{8}$$

Stretching constraints are imposed when users attempt to *explicitly* enlarge or squeeze $\mathcal{V}_l$ into a new size (w,h,d). Specifically, for every $\mathcal{V}_l$ users manipulated, it can be expressed by the following equation:

$$E_s = \sum_l (||w'_l - w_l||^2 + ||h'_l - h_l||^2 + ||d'_l - d_l||^2). \tag{9}$$

Note that when stretching a bounding volume encapsulating a regular pattern, users can only specify a size compliant with the regularization rules listed in Table 1. During the runtime, the valid sizes are computed automatically when users manipulate a bounding volume. The overall retargeting result can be obtained by minimizing the total energy formed by summing up all energy terms:

$$E = w_c \cdot (E_c + E_f + E_r) + w_p \cdot E_p + w_a \cdot E_a + w_i \cdot (E_i + E_s), \tag{10}$$

where w_c, w_p, w_a, and w_i are weighting coefficients.

Model	# of Triangles	Time (second)
Arch Building	44,382	165
CTHS	19,260	341
Harbour House	244,360	731
Living Room	7,343	69
Modern House	120,461	206
Octagonal Pavilion	51,748	128
Pavilion	8,772	106
Stairs	5,859	65
Twin Stairs	159,908	256
Xylophone	12,480	71

Table 2: Computation time of regularity detection on various test models.

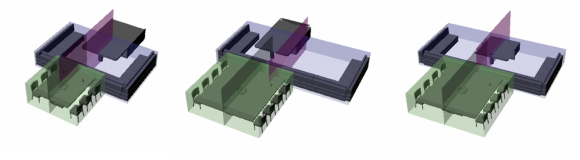


Figure 6: A demonstration of performing retargeting with and without alignment interactive and semantic constraint. Notice that we visualize the bounding volumes and center planes of two models (sofa and dinning table) are also visualized. (a) shows the initial configuration of the scene. (b) shows that the sofa is stretched by user manipulation and becomes mis-aligned with the dining table. (c) By imposing an alignment constraint, the mutual relationship between the two models is restored.

5.3. Optimization and Scene Synthesis

By solving the linear system formed by Eq. 10, the parameters of the modified bounding volumes can be determined, which can be accomplished by the traditional least square minimization method. To update the regular pattern $\mathcal{P}_k$, the center of the starting element $\mathcal{P}_k^0$ is firstly updated by the new local transformation C_j' of the enclosing $\mathcal{V}_j'$. A new number of repetition n_k' is derived to best fit to $\mathcal{V}_j'$ and $\mathcal{T}_k$ is updated accordingly. $\mathcal{P}_k'^0$ and $\mathcal{T}_k'$ together can then be exploited to reconstruct the retargeted regular pattern. The overall retargeting result is obtained by deriving the new 3D geometry within $\mathcal{V}_j'$ according to its new parameters.

6. Results and Discussions

Performance. We have implemented and evaluated the proposed system on a desktop PC with an Intel i7 2.7 GHz CPU and 8 GB RAM. To solve the least square optimization, we adopted the linear solver provided by the TACUS library [TRC03]. Empirically, we have set the weighting coefficients for various linear constraints as $w_c = 10.0$, $w_p = 0.01$ and $w_a = w_i = 3.0$, respectively. Note that we have assigned greater influences to interactive constraints over positional constraints to enable users to alter the spatial arrangement of 3D objects in the original scene. All the test 3D objects and scenes were downloaded from 3D Warehouse and TurboSquid.

Referring to Table 2, the most time-consuming part of our method is to perform regular pattern detection [PMW*08]. It typically requires several minutes according to the complexity of the 3D objects and scenes. After this preprocessing, all the editing operations can be accomplished in less than one second, which is a reasonable response time.

User editing. In our system, users can directly manipulate the control bounding volumes to carry out all model- and scene-editing operations. A resizing operation can be done by a simple drag-and-drop. In the simplest scenario, users can resize the global control bounding volume to rapidly generate several variations of the original models or scenes. The local control bounding volumes are updated accordingly by the default constraints, which mainly help maintain the spatial layout and physical connectivity between them.

Finer control can be obtained by manipulating the local control bounding volumes. Recall that we assign the semantic and interactive constraints with higher influences over the default constraints. Users can thus alter the original spatial layout by modifying the alignment among the objects of a 3D scene or changing the arrangement of the objects.

Figure 6 illustrates an example of altering the default retargeting behavior by user imposed constraints. Assuming the bounding volumes of the dining table and sofas are $\mathcal{V}_1$ and $\mathcal{V}_2$, respectively. Initially, the dining table and sofas are aligned-by-center (Figure 6(a)). By stretching $\mathcal{V}_2$, the sofas are elongated in one dimension but also mis-aligned with the dining table (Figure 6(b)). To restore the alignment between the sofas and dinning table, users can specify the center of $\mathcal{V}_2$ to lie on the auxiliary center plane of $\mathcal{V}_1$, *i.e.*, ., imposing Eq.(6) (Figure 6(c)). Note that in this example the chairs are aligned with the dining table by implicitly imposing additional alignment constraints as explained in Section 5.1. One can see that by using a combination of the interactive and semantic constraints, it is very flexible for users to generate a wide variety of retargeting results.

Figure 7 demonstrates the results of retargeting various 3D models and scenes. Note that the proposed method is particularly suitable for 3D models like architectures or man-made objects, which contain rich structural regularity. In the case of 3D scenes, some objects scattered around form a semantic group resembling structural regularity (*e.g.*, the chairs or trees). Once a user modifies the control bounding volume of such semantic groups, the objects in the group are updated simultaneously to reconstruct these patterns by inserting new elements or removing existing ones. One can see that by using the proposed system, it is very convenient to rapidly create a wide variety of scene variations that are not only visually similar but also preserve the underlying structures.

Figure 8 demonstrates examples of performing object- and scene-level retargeting simultaneously. The test 3D scenes contain 3D objects, *i.e.*, the main buildings, which possess rich regularity. One can see that both the object-level and scene-level structures are preserved well in the retargeted result.

Comparison. We compare our method with the state-of-the-art regularity-based shape editing method [BWSK12]. The algebraic model proposed in [BWSK12] characterizes 3D shapes in terms

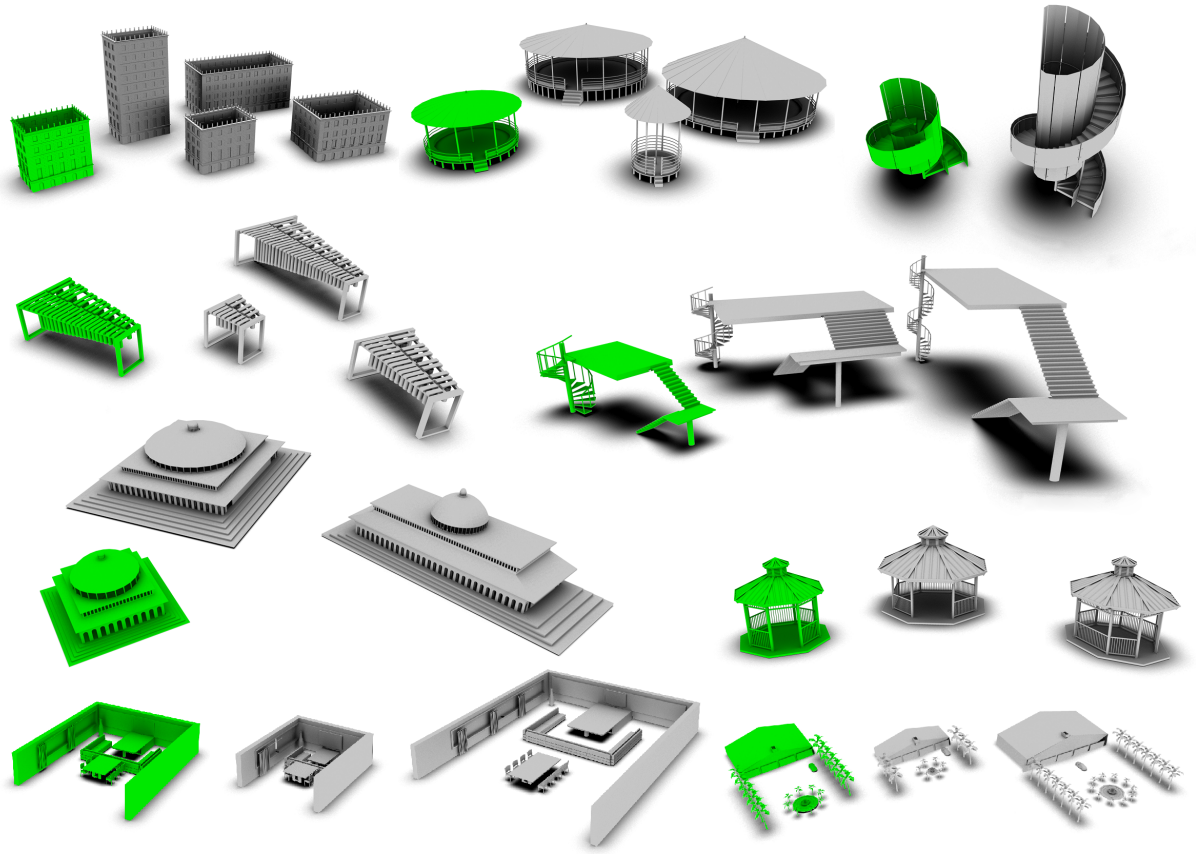


Figure 7: A variety of results for our test models and scenes. The original and retargeted models are shown in green and gray, respectively. Most of the results are automatically generated by stretching or squeezing the entire scene. Note that the car in the outdoor scene (rightmost of bottom row) is shifted to the middle by applying an additional alignment constraint.

of linked translational patterns and works well on many complex models. However, this model does not take rotational regularity into account, which is common in many 3D models. Besides, it is not trivial to extend this method to deal with 3D scenes. Figure 9 shows a comparison of the results of our method and [BWSK12]. Since [BWSK12] is not capable of handling the rotational patterns, the dome of the *Spanish Mission* model is identified as a rigid body and its size remains constant regardless of how the entire model is resized (notice that the relatively small size of the dome highlighted in red box). In contrast, our method can successfully deal with the rotational regularity. In Figure 9(b), we tried to derive similar editing results corresponding to Figure 9(a) by our approach in accordance with the repetitions of the railing. One can see that how the rotational pattern adaptively changes according to the resized models, even when the entire model is greatly enlarged. Benefiting from our default constraints, the resized dome remains aligned-by-center, resulting in more visually natural and pleasing results.

In Figure 10, we compare our method with [ZFCO*11], which also take advantage of volumetric controllers to accomplish structure-preserving shape editing. In this example, both methods tried to enlarge the model both horizontally and vertically. The results of [ZFCO*11] unnaturally stretches the base of the pavilion since the handrails are treated a rigid part of the model. Our method is more flexible and capable of generating more visually faithful variations of the input model since structural regularity is also taken into account. The default constraints also help to maintain the spatial arrangements between components. It is also worth noting that it will be difficult to apply [ZFCO*11] to deal with 3D scenes since it is a propagation-based method and requires the volumetric controllers to be connected.

Limitations. Our method still has several limitations. As a regularity-based retargeting method, our method is affected by the quality of regularity detection [PMW*08]. When the mean and Gaussian curvatures of the 3D models could not be well clustered into similar sets, it will cause unstructured pairwise transformations, resulting in failure of regularity extraction. For complex 3D objects without any regular pattern, our method can only produce a straightforward non-uniform resizing result of the retargeted 3D object. The salient structures may not be well preserved as in [KSSCO08]. We have not yet considered global symmetries or other semantic cues when operating on local bounding volumes. It might be improved by learning-based semantic regularity or con-

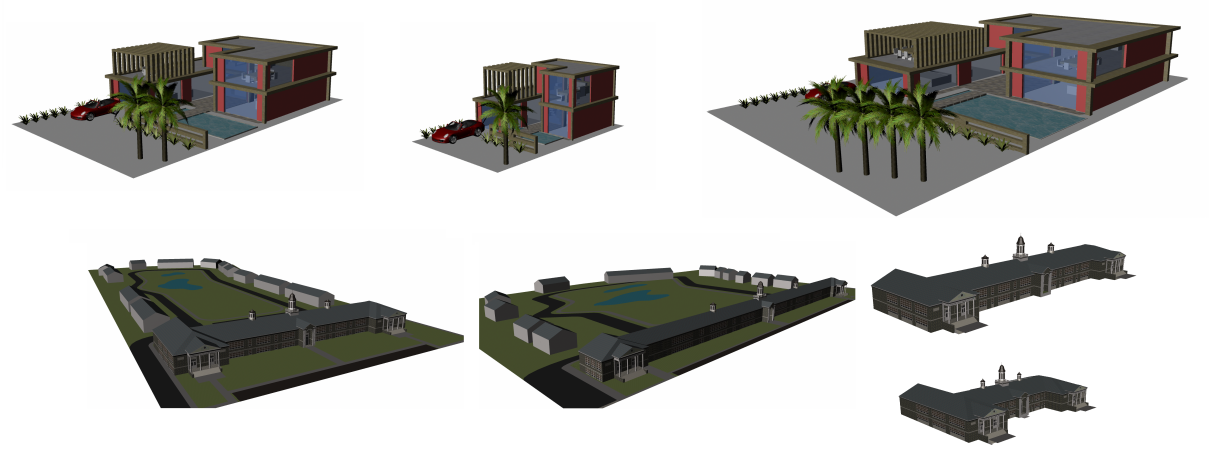


Figure 8: More results of 3D scene retargeting with textures. Note that our method does not take texture information into account when performing regularity detection and retargeting. The left are the input scenes and the others are the retargeted result. All of the results are automatically generated by stretching or squeezing the entire scene. Note that the CTHS (Collegeville-Trappe High School) scene (lower row) are substantially stretched and squeezed, while the structures and details are still well-preserved.

textual scene analysis [LCK*14]. Our approach relies on L2 minimization on a set of *soft* linear constraints. Although high weighting coefficients applied to semantic constraints can encourage the system to generate the desired results specified by users, it will be beneficial to consider *hard* constraints when certain constraints (*e.g.*, anchor constraints) should be enforced to produce the best results.

7. Conclusion

In this paper, we presented a general framework for interactively retargeting 3D models and scenes. The notion of structural analysis is extended to the level of 3D scenes, which enables us to formulate the editing of 3D objects and scenes identically under an unified framework. The proposed method works on a set of control bounding volumes, which are capable of modeling general structural regularity and facilitating user manipulation. It provides an interactive and easy-to-use tool that allows users to explore a properly defined space of scene variations by solving a constrained optimization problem and is highly flexible to alter the scene configurations by imposing new constraints. In spite of its simplicity, the proposed method achieves more general structure-preserving pattern synthesis in both object and scene levels.

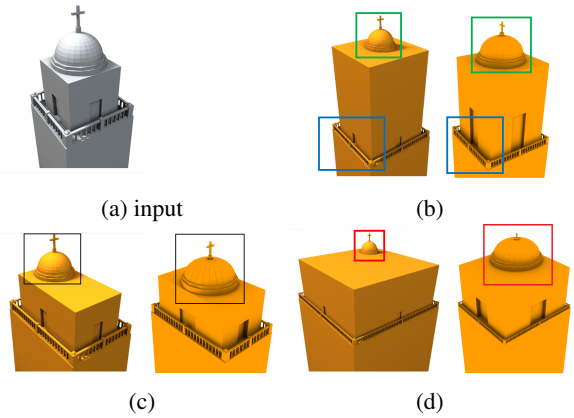


Figure 9: (a) shows a challenging 3D model and (b)-(d) compare the editing results generated by [BWSK12] (left) and our method (right). The railings (highlighted with blue box in (b)) show that our approach can well preserve translational patterns as [BWSK12]. Since [BWSK12] cannot handle rotational patterns, the dome is identified as a rigid body and its size remains constant regardless of how the entire model is resized. In our approach, the dome consisting of a rotational pattern can be adaptively synthesized in different configurations. Note that the positional constraints in our approach also help to keep the dome centered on the roof after resizing.

Acknowledgements

We thank anonymous reviewers for encouragements and thoughtful suggestions. This work was supported in part by the Ministry of Science and Technology under MOST-103-2218-E-002-030.

References

[AZL12] ALHASHIM I., ZHANG H., LIU L.: Detail-replicating shape stretching. *The Visual Comput. 28*, 12 (2012), 1153–1166. 2

[BBW*09] BOKELOH M., BERNER A., WAND M., SEIDEL H.-P., SCHILLING A.: Symmetry detection using feature lines. *Comput. Graph. Forum (Proc. of Eurographics '09) 28*, 2 (2009), 697–706. 1, 2

[BSW13] BAO F., SCHWARZ M., WONKA P.: Procedural facade variations from a single layout. *ACM Trans. Graph. 32*, 1 (2013), 8:1–8:13. 2, 3

[BWKS11] BOKELOH M., WAND M., KOLTUN V., SEIDEL H.-P.: Pattern-aware shape deformation using sliding dockers. *ACM Trans.*

(a) Input (b) Zheng *et al.* [ZFCO*11] (c) Our method

Figure 10: (a) Input model and the corresponding editing results with similar manipulation by (b) [ZFCO*11], and (c) our method. One can see that our method can better preserve the structural properties of the input model.

Graph. (Proc. of SIGGRAPH Asia '11) 30, 6 (2011), 123:1–123:10. 1, 2

[BWS10] BOKELOH M., WAND M., SEIDEL H.-P.: A connection between partial symmetry and inverse procedural modeling. *ACM Trans. Graph. (Proc. of SIGGRAPH '10) 29*, 4 (2010), 104:1–104:10. 2, 3

[BWSK12] BOKELOH M., WAND M., SEIDEL H.-P., KOLTUN V.: An algebraic model for parameterized shape editing. *ACM Trans. Graph. (Proc. of SIGGRAPH '12) 31*, 4 (2012), 78:1–78:10. 1, 2, 5, 7, 8, 9

[CKGK11] CHAUDHURI S., KALOGERAKIS E., GUIBAS L., KOLTUN V.: Probabilistic reasoning for assembly-based 3D modeling. *ACM Trans. Graph. (Proc. of SIGGRAPH '11) 30*, 4 (2011), 35:1–35:10. 2

[FRS*12] FISHER M., RITCHIE D., SAVVA M., FUNKHOUSER T., HANRAHAN P.: Example-based synthesis of 3D object arrangements. *ACM Trans. Graph. (Proc. of SIGGRAPH Asia '12) 31*, 6 (2012), 135. 2

[FSH11] FISHER M., SAVVA M., HANRAHAN P.: Characterizing structural relationships in scenes using graph kernels. *ACM Trans. Graph. (Proc. of SIGGRAPH '11) 30*, 4 (2011), 34:1–34:12. 2

[GJWW15] GUERRERO P., JESCHKE S., WIMMER M., WONKA P.: Learning shape placements by example. *ACM Trans. Graph. 34*, 4 (July 2015). 2

[GSMCO09] GAL R., SORKINE O., MITRA N. J., COHEN-OR D.: iWIRES: an analyze-and-edit approach to shape manipulation. *ACM Trans. Graph. (Proc. of SIGGRAPH '09) 28*, 3 (2009), 33:1–33:10. 1, 2, 3

[KCKK12] KALOGERAKIS E., CHAUDHURI S., KOLLER D., KOLTUN V.: A probabilistic model for component-based shape synthesis. *ACM Trans. Graph. (Proc. of SIGGRAPH '12) 31*, 4 (2012), 55:1–55:11. 2

[KSSCO08] KRAEVOY V., SHEFFER A., SHAMIR A., COHEN-OR D.: Non-homogeneous resizing of complex models. *ACM Trans. Graph. (Proc. of SIGGRAPH '08) 27*, 5 (2008), 111:1–111:9. 2, 8

[KWS16] KALOJANOV J., WAND M., SLUSALLEK P.: Building construction sets by tiling grammar simplification. *Comput. Graph. Forum* (2016). 2

[LCK*14] LIU T., CHAUDHURI S., KIM V. G., HUANG Q., MITRA N. J., FUNKHOUSER T.: Creating consistent scene graphs using a probabilistic grammar. *ACM Trans. Graph. 33*, 6 (Nov. 2014). 9

[LCOZ*11] LIN J., COHEN-OR D., ZHANG H. R., LIANG C., SHARF A., DEUSSEN O., CHEN B.: Structure-preserving retargeting of irregular 3D architecture. *ACM Trans. Graph. (Proc. of SIGGRAPH Asia '11) 30*, 6 (2011), 183:1–183:10. 2, 3, 5

[LVW*15] LIU H., VIMONT U., WAND M., CANI M.-P., HAHMANN S., ROHMER D., MITRA N. J.: Replaceable substructures for efficient part-based modeling. *Comput. Graph. Forum* (2015). 2

[LW15] LI C., WAND M.: Approximate translational building blocks for image decomposition and synthesis. *ACM Trans. Graph. 34*, 5 (Nov. 2015), 158:1–158:16. 3

[MGP06] MITRA N. J., GUIBAS L. J., PAULY M.: Partial and approximate symmetry detection for 3D geometry. *ACM Trans. Graph. (Proc. of SIGGRAPH '06) 25*, 3 (2006), 560–568. 1, 2, 3, 4

[MPWC13] MITRA N. J., PAULY M., WAND M., CEYLAN D.: Symmetry in 3D geometry: Extraction and applications. *Comput. Graph. Forum 32*, 6 (2013), 1–23. 2

[MSK10] MERRELL P., SCHKUFZA E., KOLTUN V.: Computer-generated residential building layouts. *ACM Trans. Graph. (Proc. of SIGGRAPH Asia '12) 29*, 6 (2010), 181:1–181:12. 2, 3

[MWCS13] MILLIEZ A., WAND M., CANI M.-P., SEIDEL H.-P.: Mutable elastic models for sculpting structured shapes. *Comput. Graph. Forum 32*, 2pt1 (2013), 21–30. Special Issue: Eurographics, May 2013, Girona, Spain. 2

[MWH*06] MÜLLER P., WONKA P., HAEGLER S., ULMER A., VAN GOOL L.: Procedural modeling of buildings. *ACM Trans. Graph. (Proc. of SIGGRAPH '06) 25*, 3 (2006), 614–623. 2

[MWZ*13] MITRA N., WAND M., ZHANG H., COHEN-OR D., BOKELOH M.: Structure-aware shape processing. In *Eurographics '13 STARs* (2013), pp. 175–197. 2

[PM01] PARISH Y. I. H., MÜLLER P.: Procedural modeling of cities. In *Proc. of ACM SIGGRAPH '01* (2001), pp. 301–308. 2

[PMW*08] PAULY M., MITRA N. J., WALLNER J., POTTMANN H., GUIBAS L.: Discovering structural regularity in 3D geometry. *ACM Trans. Graph. (Proc. of SIGGRAPH '08) 27*, 3 (2008), 43:1–43:11. 1, 2, 3, 4, 7, 8

[PYW14] PENG C.-H., YANG Y.-L., WONKA P.: Computing layouts with deformable templates. *ACM Trans. Graph. (Proc. of SIGGRAPH '14) 33*, 4 (2014), 99:1–99:11. 2, 3

[TLL*11] TALTON J. O., LOU Y., LESSER S., DUKE J., MĚCH R., KOLTUN V.: Metropolis procedural modeling. *ACM Trans. Graph. 30*, 2 (2011), 11:1–11:14. 2, 3

[TRC03] TOLEDO S., ROTKIN V., CHEN D.: TAUCS: A library of sparse linear solvers, 2003. Version 2.2. 7

[TYK*12] TALTON J., YANG L., KUMAR R., LIM M., GOODMAN N., MĚCH R.: Learning design patterns with bayesian grammar induction. In *Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology* (2012), UIST '12. 2

[VGDA*12] VANEGAS C. A., GARCIA-DORADO I., ALIAGA D. G., BENES B., WADDELL P.: Inverse design of urban procedural models. *ACM Trans. Graph. 31*, 6 (2012), 168:1–168:11. 2

[WWSR03] WONKA P., WIMMER M., SILLION F., RIBARSKY W.: Instant architecture. *ACM Trans. Graph. (Proc. of SIGGRAPH '03) 22*, 3 (2003), 669–677. 2, 3

[WXL*11] WANG Y., XU K., LI J., ZHANG H., SHAMIR A., LIU L., CHENG Z.-Q., XIONG Y.: Symmetry hierarchy of man-made objects. *Comput. Graph. Forum (Proc. of Eurographics '11) 30*, 2 (2011), 287–296. 2

[XLZ*10] XU K., LI H., ZHANG H., COHEN-OR D., XIONG Y., CHENG Z.-Q.: Style-content separation by anisotropic part scales. *ACM Trans. Graph. (Proc. of SIGGRAPH Asia '10) 29*, 6 (2010), 184:1–184:10. 2

[XMZ*14] XU K., MA R., ZHANG H., ZHU C., SHAMIR A., COHEN-OR D., HUANG H.: Organizing heterogeneous scene collection through contextual focal points. *ACM Trans. Graph. (Proc. of SIGGRAPH '14) 33*, 4 (2014), 35:1–35:12. 2

[YYT*11] YU L.-F., YEUNG S.-K., TANG C.-K., TERZOPOULOS D., CHAN T. F., OSHER S. J.: Make it home: Automatic optimization of furniture arrangement. *ACM Trans. Graph. (Proc. of SIGGRAPH '11) 30*, 4 (2011), 86:1–86:12. 2, 3

[YYW*12] YEH Y.-T., YANG L., WATSON M., GOODMAN N. D., HANRAHAN P.: Synthesizing open worlds with constraints using locally annealed reversible jump MCMC. *ACM Trans. Graph. (Proc. of SIGGRAPH '12) 31*, 4 (2012), 56:1–56:11. 2

[ZFCO*11] ZHENG Y., FU H., COHEN-OR D., AU O. K.-C., TAI C.-L.: Component-wise controllers for structure-preserving shape manipulation. *Comput. Graph. Forum (Proc. of Eurographics '11) 30*, 2 (2011), 563–572. 1, 2, 3, 8, 10