# 以代數表示的結構模型之模型編輯

黃群凱* 羅聖傑* 沈奕超† 陳炳宇‡
*‡國立臺灣大學 †中央研究院
*{chinkyell, forestking}@cmlab.csie.ntu.edu.tw †joshshen@citi.sinica.edu.tw
‡robin@ntu.edu.tw

## ABSTRACT
直至今日，產生3D模型仍然是一個耗費大量時間與功夫的工作。而這些人造的3D模型在建造的時候往往帶有結構特性，像是樓梯的台階、建築物的門柱與窗戶等。具有結構性的模型會重複的使用相同的模組，而且模組之間具有特定的位移、旋轉或縮放的特性。我們的方法採用最新的模型結構的對稱搜尋技術將模型中相同的結構，以單一的模組與代數轉換來表示。在這套代數的結構模型下，我們提供使用者進行移動單一模組的操作；試圖找出最適合這個操作的模型變化。並且使用我們所設計的最佳化演算法，能夠在整個模型在保有相同結構的前提下，計算出結構模型最好的參數且能夠符合使用者操作的結果。

## Categories and Subject Descriptors
I.3.5 [**Computer Graphics**]: Computational Geometry and Object Modeling

## General Terms
Algorithms

## 1. INTRODUCTION
3D content creation is the most fundamental and tedious task in computer graphics. However, with the advent developments in easy shape acquisitions and analyses, a whole new bunch of shape creation and editing method has emerged that supports creativities from various existing concepts, e.g. from photographs [19], videos [16], or existing model databases [6, 14].

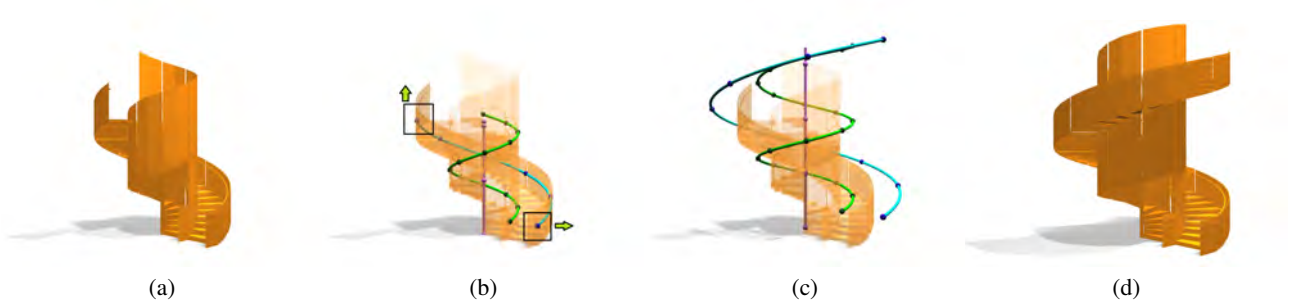
In recent year, lots of 3D model generating researches have been published. For example, procedural modeling which follows sets of rules and configure by parameters to creates diverse 3D contents. Model retrieval technique helps user to working with large collections of 3D models. User can combine retrieval contents to synthesize new 3D scenes. However, the goal of our work is different to the above techniques. We focus on user-specified man-made model and provide a shape editing environment, which can create a structure-adapting result by interactive manipulation.

We observed that most of the man-made shapes such as architectures contain structural regularity patterns that are fundamental in almost all styles of architectures. Based on this observation, we can represent the man-made shape with the regularity patterns as an algebraic model. In such shapes, every regular pattern repeatedly appears in a predictable relation such as translation, rotation, and scale, so the original shape can be reorganized by sets of regularity and relation of some regular elements. Therefore, it is important to adjust the amount of patterns in a logical way to make the appearing times being correctly and reproduce another shape which structurally adapted to the original one.

The first step of our method is to analyze and decompose an input mesh to a set of regularities, and then record the transformations of each pattern and parameters of the original shape. Then, We provide a user control mechanism to let users select an interesting part of the shape and grab it to the desired position. The result can perfectly satisfy both the user-specified interaction and remain structurally adapted. Our contribution is to provide a generalized algebraic model to represent a structural shape, and find the degree of freedom of shape editing while maintaining global characteristics.

## 2. RELATED WORK
**Regularity detection and extraction.** A key component of our approach is to automatically obtain repetitive elements from an input shape. Regularity detection can be treated as an algorithm that can find intra-shape symmetries. Most model retrieval works [7] [15] are designed to search in object database by computing the shape descriptors to find another similar shape comparing to the input shape. Partial shape matching is closer to pattern detection. Gal and Cohen-Or [8] tried to identify matching parts of meshes. Niloy *et al.* [12] regarded the problem as voting for dominant pairwise transformation. Our approach uses the method of Pauly *et al.* [13] to detect and extract the regular patterns, and structure discovery is performed by collecting patch similarities and analyzing pairwise transformation. Bokeloh *et al.* [2] find rigid symmetries in general configurations, and avoid the problem of clustering the transformation space comparing to previous voting algorithm. They get a good recognition performance without additional assumptions. Unfortunately, the line feature could not extract the rotational information.

**Structure-preserving editing.** In recent years, structure-aware shape editing [11] is a popular topic. Non-uniform shape deformation approaches treat single model as several parts according to vulnerable analysis [10] and mesh segmentation [18]. In order to prevent some parts of shape undergo undesirable deformation, [10] construct a vulnerability map and deform the shape using



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

**Figure 1: This paper presents a shape editing algorithm that adapts the structure of a given 3D shape (a). Our approach discovered the structure regularity that represents repetitive elements as nodes and transformation path as curves (b), respectively. Users can interactively select two nodes to move to the desired position. Our algorithm then optimizes the translation, rotation, and scaling parameters to generate a new structure (c). The result reshaped by each element based on optimizing the parameters (d) that satisfies user-specified constraints while maintaining its global characteristics with the given shape.**

space-deformation method. Besides, Gal *et al.* [9] provided an analyze-and-edit shape editing approach that preserve important object features by modifying the intelligent wires. These wires are grouped similar to structure discovering. Cabral *et al.* [5] reshape the geometries and textures in a couple during interactively modifying the length of edges. Alhashim *et al.* [1] replicate surface details during non-uniform stretching by adopting texture synthesis technique. Some researches aim for resembling a new 3D shape from intra-shape of an input model. Wang *et al.* [17] decompose and edit an object while respecting symmetry relations. However, this approach does not provide length and scale parameters that leads to limited editing results. Bokeloh *et al.* [3] introduced a structure-aware deformation technique that adaptively inserts or removes discrete regular patterns when deforming shape by user interaction. It is combined elastic deformation while maintaining structure and adapting the repetition count. Then, they [4] further improved the prior work by an elegant algebraic model and degree of freedom of shape editing by null space analysis. However, the real architectures and man-made objects are much more complicated due to the circular design and scale component. Our method is more general by automatically detecting these kinds of structure. Furthermore, our work extends the range of result variations.

## 3. GENERALIZED ALGEBRAIC MODEL

Our algebraic model is based on structure regularities, which are derived from an input 3D object by discovering repetitive patterns. We denote a user given shape as $\mathbf{S} \subset \mathbf{R^3}$. Similar to most structure detection algorithms, we try to find a set of generative intra-shapes $\mathbf{s}_i = (\mathbf{P}_0, \mathbf{T}_i, \mathbf{n}_i), \mathbf{0} \leq i \leq \mathbf{m}$ such that the union of $\mathbf{s}_0 \cup \cdots \cup \mathbf{s}_m$. can cover entire $\mathbf{S}$, where each intra-shape $\mathbf{s}_i$ is composited by pattern $\mathbf{p}_k, \mathbf{0} \leq k \leq \mathbf{n}_i$, which is derived by applying transformation $\mathbf{T}_k$ to $\mathbf{P}_0$, and $\mathbf{n}_i$ is the maximum number of this set. By this definition, the regular patterns are easy to manipulate by changing the start position of $\mathbf{P}_0$ and count of elements $\mathbf{n}_i$. However, manipulation of the algebraic model is not so quite simple, since each modification for any pattern will affect neighboring patterns recursively. Therefore, our approach is to detect structural patterns in $\mathbf{S}$ as much as possible and find the all possibility of modification parameters that maintain the structure of the original shape. Then, computing a best combination of these modifications is suitable for editing.

Our approach utilizes a reliable structure discovering algorithm [13] for decomposing the input shape. We will precisely define how this intra-shape to organize an algebraic model.

**Structural Regularity.**

We consider most regular patterns of man-made objects can be described by some single elements, a subspace of affine transformations, and the repetitive count of the elements. The subspace of affine transformations is orientation- and similarity-preserving, which can be composed of uniform scaling, rotation, and translation. We use homogeneous coordinates $\mathbf{H}$ to represent the transformation, which is define as:

$$\mathrm{H} = \begin{pmatrix} s \cdot \mathbf{R} & \mathbf{t} \\ \mathbf{0} & 1 \end{pmatrix}, \tag{1}$$

where $s$ is the uniform scaling factor, $\mathbf{R}$ is a rotation matrix, and $\mathbf{t}$ is a translation vector. For simply defining a transformation, we can restrict only one parameter (such as translation, rotation, and uniform scaling) at a time. If there is a helix, our approach can combine two transformations such as adding the rotational parameter and translational parameter. In the following, we will describe different combinations of the transformations.

1**-parameter groups.** A single regular pattern has common parameters: the origin of the pattern, the number of repetition counts, the type and parameters of the transformation. For each first element of the regular pattern, we compute its center as the origin of the pattern. The type of the transformation indicates the three types: scale, rotation, and translation. Typically, the number of repetition counts will be always non-negative. For the rotational pattern, additional parameters are needed, which are two positions to present the structure: one is the center of the rotation as the origin of the pattern and another is the center of the first element. In our approach, we modify the radius of the rotational structure instead of changing the number of repetition. For more details, the count of the elements of a circle is fixed. If the structure is a half circle, it will not be changed after manipulation. The parameters of the transformation is specified by its type such as $s$, $\mathbf{t}$, and $\Theta$ corresponding to the uniform scaling factor, translational vector, and rotational degree. The axis of the rotation is an extra information of the transformation for constructing the rotational matrix. These parameters are defined when the patterns are detected and decomposed from the input shape. Furthermore, the transformation and specified parameters will stay constant in our approach. To achieve moving and resizing the whole patterns, the origin of the pattern and the number of repetition counts (radius when rotational pattern) are manipulated.

2**-parameter groups.** When the structure is complex such as he-

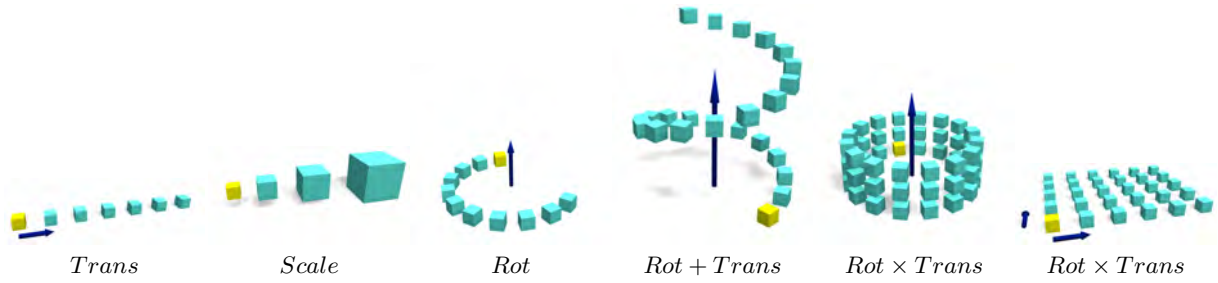



**Figure 2: Schematic illustration of parameter groups. Every yellow elements denote the first element of a pattern. The transformation derived from the parameter can transfer first element to next element, and generate all elements of the pattern by accumulative transformation. $Rot + Trans$ simply combine two parameters in transformation. $Rot \times Trans$ and $Rot \times Trans$ combine all possibility of accumulative transformation.**

lix or amphitheatre-like shape (Figure 2 $Rot + Trans$ and $Rot \times Trans$), we combine two 1-parameter transformations to represent. For example, if the vertical size of windows of a building on the wall is $m$ and the horizontal size is $n$, it obviously needs two independent vectors as a horizontal vector and a vertical vector to retrieve all positions of the windows. In this case, we combine two 1-$translation$-parameter groups. Since the whole structure of the windows has $m \times n$ elements (windows) and composited as a grid, the transformation of 1-parameter-groups is combined by multiplication. Similar to [13], we called it as translation-multiple-translation ($Trans \times Trans$). For homogeneous coordinates, a translation-add-translation ($Trans + Trans$) structure is not existed since adding two translations can be represented by a single translation. Rotation-multiple-translation (amphitheatre-like) structure and rotation-add-translation (helix) structure are often existed in man-made objects. Figure 2 shows all structures handled by our approach. We do not consider other combinations and more than 2-parameter groups since they are not common in real cases. To combine the counts of the elements of groups is dependence. In the above $Trans \times Trans$ case, the first parameter must be non-negative typically. The second parameter will be at least one of the number of repetition counts because it exists logically. Obviously, the result will be the same no matter which regular type is the first parameter, so we set the rotation pattern always be the first parameter in the 2-parameter group for convenience.

**Rigid groups.** Refer to the definition in [13], after the structure detection algorithm being applied, the pattern groups will cover most of the input shape. In most cases, the residual parts are not always fragment but surrounding one of the patterns. It causes that the algebraic model has no editing degree of freedom because the resizing of the elements conflicts the residual parts. We therefore reduce the sampling space by half and then detect again until there is no surrounding residual part. After all, we set these residual fragments as rigid parts and record its origin only.

**Connections.** The given shape decomposes the parameter groups and rigid groups. The parameter groups only maintain the structure of the regular patterns, so we need additional knowledge to adapt the entire structure while resizing and moving the regular patterns. In our approach, a regular pattern is decomposed, and will construct a connection linked from the origin of the boundary element to a next origin belongs to other patterns. These connections will be used to record how groups are organized each other. Our shape structure-aware editing is based on how to keep connections invariant and change the parameter of the parameter groups at the same time.

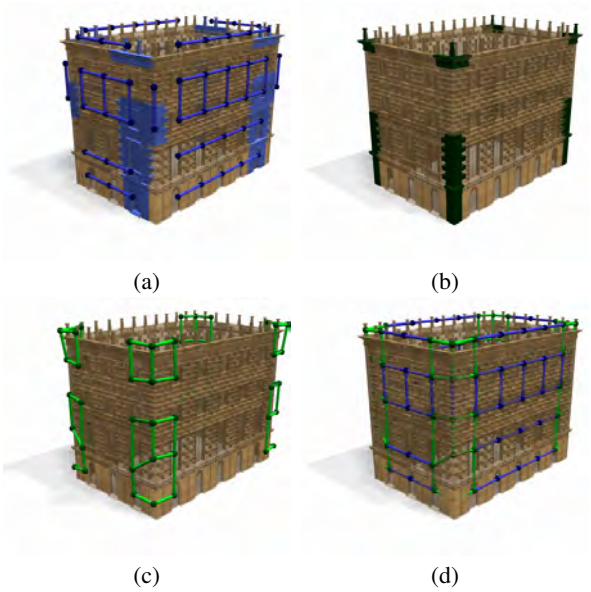



**Figure 3: For a user given shape. Our approach discover its structure and decomposed to several patterns. These patterns are analysed to find transformation of elements. We save the first repetitive element and need parameter of transformation, and it denoted as blue node and blue regions in (a). The residual shapes will be saved as rigid part (b). After all, we construct linked information to ensure global characteristic can be maintained (c). Combining all of above is our general algebraic model (d).**

Figure 3 shows how to generate a general algebraic model.

## 4. ANALYSIS AND EDITING

Based on the definition of our general algebraic model, the parameters of groups will keep constant excluding the origin of the first element of the patterns and the number of repetition counts (radius when rotational patterns). Our purpose is clear and simple that the parameters of the transformation keep constant to ensure maintaining the structure of the regular patterns, and keep the connection constant to ensure each pattern structure linked correctly respect to the original shape. Figure 4 describes that we cannot modify a parameter simply. If one parameter is changed, a serial parameters will be changed, respectively. In this section, we describe how to find dependent parameters and formulate as a quadric problem to

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

**Figure 4: This figure shows that why the algebraic model cannot change parameters simply. (a) There are three distinguished patterns and the deep blue lines are the connections. (b) The cube pattern resizes the count of elements but violates the connections. (c) To prevent the violation, the pyramid pattern changes the origin of the start element. However, it still violates another connection. (d) The connections are legal by resizing the count of elements of the sphere pattern. It totally modifies three parameters in this simple case. (e) When resizing the count of the sphere pattern by odd number, there is no way to satisfy the connections.**

satisfy the user interaction.

**Valid modification.**

We create a linear system to represent these parameters that will be constant and those parameters that can be changed as variables. The offsets of the connections are placed on the right-hand side. In this case, the kernel of matrix $A$ can be computed by Gaussian elimination. This infers an idea that the kernel of $A$ keeps constant right now. We just need to find how the right-hand side will be constant, too. The null space of the matrix $A$ is our answer, since the variables changed according to the basis of null space of the matrix $A$. The right-hand side ensures to be the same, so the valid and freedom manipulation will be the null space of this system. Obviously, manipulating by these null space basics will not change the content of the linear system.

The linear system:

$$\mathbf{A}x = \mathbf{b}, \tag{2}$$

where the variable $x$ is all of the positions of the origin of the first element and $k$ is the number of repetition counts (or radius when rotational pattern). The integer $k$ is specified the $k$-parameter groups. The matrix $A$ needs to explain a subtraction of both side origin of the connection, and $\mathbf{b}$ is the subtraction result (offset of both side origin of the connection). For each connection, we try to derive $k$ translation vectors (specified the $k$-parameter groups). This information is used to compute the origin of the elements of the connection groups specified. In our experience, the matrix $\mathbf{A}$ is undetermined in most cases. For extracting valid modification, we derive the null space of the matrix $\mathbf{A}$ by singular value decomposition (SVD) techniques. A basis of the null space is those columns that are associated with singular values equal to zero. We consider the singular values to be zero if their values are less than 1% of the largest singular value.

Let $N(\mathbf{A})$ be the null space of the matrix $\mathbf{A}$ and $\mathbf{N}_A$ be a basis of $N(\mathbf{A})$. The valid modification will be the combination of $\mathbf{N}_A$. We represent the modification space by the function of $\mathbf{N}_A$, and $\lambda$ is a parameter for the function of $\mathbf{N}_A$. Then, possible valid results of the algebraic model will be a function of the parameter $\lambda$:

$$\mathbf{x}(\lambda) = \mathbf{x_0} + \mathbf{N}_A * \lambda, \tag{3}$$

where $\mathbf{x_0}$ is the initial state of variable $\mathbf{x}$. This function still has invalid result because the number of repetition counts could be negative.

**Nonnegative.**

For the definition of the 2-parameter groups, we have to keep the first parameter being non-negative and the second parameter being at least one. We then set the boundary conditions to prevent $\mathbf{x}(\lambda)$ containing negative number of repetition counts. We construct a matrix $\mathbf{M}$ to satisfy all non-negative constraints by the express equation $\mathbf{M}(\mathbf{x}) \geq \mathbf{m}$. We hope the result of $\lambda$ can satisfy the above inequalities, so we map this system from the domain of variable $\mathbf{x}$ into the domain of parameter $\lambda$:

$$\mathbf{M}'(\lambda) \geq \mathbf{m}', \tag{4}$$

where $\mathbf{M}' = \mathbf{M}\mathbf{N}_A$ and $\mathbf{m}' = \mathbf{m} - \mathbf{M}\mathbf{x_0}$.

**User interactive constraints.**

By the above two expresses, we know that how to retrieve a valid result from the function of the parameter $\lambda$ and its boundary constraints. For now, we transfer the user editing to the numerical constraints. In our approach, we offer users to pick an element $e$ of the pattern to grab it to a wondering position $p$. Then, we derive the offset of the origin of the selected element $e$ and the position $p$. We formulate a quadratic objective term:

$$E_{interactive}(\mathbf{x}) = (e_{origin} - p)^2. \tag{5}$$

The user interactively grabs only an element once. We then offer another user constraint that the user can select several elements $m$ to fix the elements in the current position. We define the quadratic objective term as:

$$E_{fixed}(\mathbf{x}) = (m_{origin} - p)^2. \tag{6}$$

**Quadratic optimization.** We then sum the user interactive objective terms to quadratic form:

$$\begin{aligned} E(\mathbf{x}) &= E_{interactive}(\mathbf{x}) + E_{fixed}(\mathbf{x}) \\ &= \mathbf{x}^T\mathbf{Q}\mathbf{x} + \mathbf{x}^T\mathbf{q}. \end{aligned} \tag{7}$$

Then, we transfer the objective formulation from the domain of variable $x$ into the domain of parameter $\lambda$ like the boundary condition.

$$E'(\lambda) = \lambda^T((\mathbf{N}_A^T)\mathbf{Q}\mathbf{N}_A)\lambda + \lambda^T(\mathbf{N}_A^T)\mathbf{q}. \tag{8}$$

In the end, we minimize the quadratic objective function subject to the boundary conditions.

$$\arg\min_{\lambda} E'(\lambda) \; s.t. \; \mathbf{M}'(\lambda) \geq \mathbf{m}'. \tag{9}$$

(a) Colosseo (b) Curve stair (c) Harbour house

**Figure 5: User given shapes (grey) and correspondence editing results (orange).**

## 5. RESULT AND CONCLUSION

The implementation of our approach first utilized the structure detection algorithm [13]. The extracted structural patterns and element transformation help us to construct the parameters and rigid groups. After collecting linking information of different patterns, the general algebraic model is established. Then, we use the linear system and singular value decomposition (SVD) techniques to derive the valid modification of the algebraic model. The objective function is used to satisfy the user specified constraints through grabbing an element to a desired position. The quadratic function optimizes the best start positions of the patterns and the number of the repetitive counts. The result model is easy to produce with our representation, and the number of elements are rounded to integer instead of directly using the real number after the optimization. To justify the start position of the pattern, we fix the number of the repetitive counts and optimize the objective function again without manipulation. The result will not be break due to the gap of real number and integer. For rotational pattern, the increasing or decreasing radius makes the result breaking. Hence, we apply the uniform scaling on the orthogonal direction corresponding to the rotational axis to ensure that the elements can connect correctly, like the horizontal editing shown in Figure 1.

**Editing results and shapes analysis.**

We have tested a large number of structural shapes obtained from well-known 3D model repositories, such as Google 3D Warehouse. Figure 5 shows the original shapes and editing results by our system. Our structure detection algorithm constructs lots of regular patterns from the given shapes. Every pattern has at least one parameter to control the number of repetitive counts. It seems that the degree of editing freedom is huge. However, the editing behavior always conflicts each other due to maintaining the original structure. By the analysis of our algebraic model and valid modification, the degree of control is less than 10 in most of our testes. This knowledge helps us to understand that the real variation of the input shapes.

**Performance.**

Our system was implemented and evaluated on a consumer-grade desktop PC with an Intel i7 2GHz CPU and 8GB RAM. We use LAPACK library for singular value decomposition and MOSEK library for quadratic optimization. The statistics of our approach are shown in Table 1. The time of synthesis depends on the topology of the shapes and the size of the editing result, so it is omitted.

**Limitation and future work.**

| Model | Tri. Number | Structure detection | Analysis |
|---|---|---|---|
| Twin stair (teaser) | 159,908 | 256s | 0.49s |
| Colosseo | 65,069 | 156s | 0.48s |
| Harbour house | 244,360 | 731s | 1.21s |
| stair | 2,422 | 35s | 0.31s |

**Table 1: Statistics for our approach and test shapes. Structure detection time contains the construction of the general algebraic model. Analysis time is measured for one of the user interactive editing.**

Our approach heavily depends on the structure detection algorithms. The problem will occur when the detection algorithm misunderstanding the structure of the shapes. It may make our general algebraic model incorrect. Even the structure detection is reliable, the extracted patches may loss the details or misalign to the original shapes. Our approach cannot tolerate these problems since we used a simple shape duplication to synthesize the result. In future work, we will focus on how to extract a suitable intermediate data for our approach. In the synthesis stage, the duplication produces lots of unnecessary vertices even polygons. It is also an important improvement for producing useful results.

**Conclusion.**

We have presented a shape-editing approach that enables users to rapidly produce results which have the same structure with the input shapes. Our approach starts from the regular pattern detection and shape understanding. To maintain the global characteristics while resizing the shape by keeping the structural content and link of the patterns, we use the linear system to derive the valid modification of our general algebraic model. The valid modification also reveals the shape variations. Our method satisfied the user interaction by representing it as a quadratic constraint and optimizing a quadratic objective function.

ACKNOWLEDGMENTS are optional

## 6. 致謝

本論文感謝國家科學委員會經費補助，計畫編號：NSC101-2221-E-002-200-MY2。

## 7. REFERENCES

[1] I. Alhashim, H. Zhang, and L. Liu. Detail-replicating shape stretching. *The Visual Computer*, 28(12):1153–1166, 2012.

[2] M. Bokeloh, A. Berner, M. Wand, H.-P. Seidel, and A. Schilling. Symmetry detection using feature lines. *Comput. Graph. Forum*, 28(2):697–706, 2009.

[3] M. Bokeloh, M. Wand, V. Koltun, and H.-P. Seidel. Pattern-aware shape deformation using sliding dockers. *ACM Trans. Graph.*, 30(6):123:1–123:10, 2011.

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

[5] M. Cabral, S. Lefebvre, C. Dachsbacher, and G. Drettakis. Structure-preserving reshape for textured architectural scenes. *Comput. Graph. Forum*, 28(2):469–480, 2009.

[6] S. Chaudhuri, E. Kalogerakis, L. Guibas, and V. Koltun. Probabilistic reasoning for assembly-based 3D modeling. *ACM Trans. Graph.*, 30(4):35:1–35:10, 2011.

[7] D.-Y. Chen, X.-P. Tian, Y.-T. Shen, and M. Ouhyoung. On visual similarity based 3D model retrieval. *Comput. Graph. Forum*, 22(3):223–232, 2003.

[8] R. Gal and D. Cohen-Or. Salient geometric features for partial shape matching and similarity. *ACM Trans. Graph.*, 25(1):130–150, 2006.

[9] R. Gal, O. Sorkine, N. J. Mitra, and D. Cohen-Or. iWIRES: an analyze-and-edit approach to shape manipulation. *ACM Trans. Graph.*, 28(3):33:1–33:10, 2009.

[10] V. Kraevoy, A. Sheffer, A. Shamir, and D. Cohen-Or. Non-homogeneous resizing of complex models. *ACM Trans. Graph.*, 27(5):111:1–111:9, 2008.

[11] N. Mitra, M. Wand, H. Zhang, D. Cohen-Or, and M. Bokeloh. Structure-aware shape processing. In *Eurographics State-of-the-art Report (STAR)*, pages ???–???, 2013.

[12] N. J. Mitra, L. J. Guibas, and M. Pauly. Partial and approximate symmetry detection for 3D geometry. *ACM Trans. Graph.*, 25(3):560–568, 2006.

[13] M. Pauly, N. J. Mitra, J. Wallner, H. Pottmann, and L. Guibas. Discovering structural regularity in 3D geometry. *ACM Trans. Graph.*, 27(3):43:1–43:11, 2008.

[14] J. O. Talton, D. Gibson, L. Yang, P. Hanrahan, and V. Koltun. Exploratory modeling with collaborative design spaces. *ACM Trans. Graph.*, 28(5):167:1–167:10, 2009.

[15] J. W. H. Tangelder and R. C. Veltkamp. A survey of content based 3D shape retrieval methods. *Multimedia Tools Appl.*, 39(3):441–471, 2008.

[16] A. van den Hengel, A. R. Dick, T. Thormählen, B. Ward, and P. H. S. Torr. VideoTrace: rapid interactive scene modelling from video. *ACM Trans. Graph.*, 26(3):86:1–86:6, 2007.

[17] Y. Wang, K. Xu, J. Li, H. Zhang, A. Shamir, L. Liu, Z.-Q. Cheng, and Y. Xiong. Symmetry hierarchy of man-made objects. *Comput. Graph. Forum*, 30(2):287–296, 2011.

[18] K. Xu, H. Li, H. Zhang, D. Cohen-Or, Y. Xiong, and Z.-Q. Cheng. Style-content separation by anisotropic part scales. *ACM Trans. Graph.*, 29(6):184:1–184:10, 2010.

[19] K. Xu, H. Zheng, H. Zhang, D. Cohen-Or, L. Liu, and Y. Xiong. Photo-inspired model-driven 3D object modeling. *ACM Trans. Graph.*, 30(4):80:1–80:10, 2011.