3D Model Retargeting Using Offset Statistics

Xiaokun Wu†
Max-Planck-Institut für Informatik
Saarbrücken, Germany
xwu@mpi-inf.mpg.de

Chuan Li†
Utrecht University
Utrecht, Netherlands
cl.chuanli@gmail.com

Michael Wand
Utrecht University
Utrecht, Netherlands
M.Wand@uu.nl

Klaus Hildebrandt
Max-Planck-Institut für Informatik
Saarbrücken, Germany
klaus.hildebrandt@mpi-inf.mpg.de

Silke Jansen
Max-Planck-Institut für Informatik
Saarbrücken, Germany
sjansen@mpi-inf.mpg.de

Hans-Peter Seidel
Max-Planck-Institut für Informatik
Saarbrücken, Germany
hpseidel@mpi-inf.mpg.de

***Abstract*—Texture synthesis is a versatile tool for creating and editing 2D images. However, applying it to 3D content creation is difficult due to the higher demand of model accuracy and the large search space that also contains many implausible shapes. Our paper explores offset statistics for 3D shape retargeting. We observe that the offset histograms between similar 3D features are sparse, in particular for man-made objects such as buildings and furniture. We employ sparse offset statistics to improve 3D shape retargeting (i.e., rescaling in different directions). We employ a graph-cut texture synthesis method that iteratively stitches model fragments shifted by the detected sparse offsets. The offsets reveal important structural redundancy which leads to more plausible results and more efficient optimization. Our method is fully automatic, while intuitive user control can be incorporated for interactive modeling in real-time. We empirically evaluate the sparsity of offset statistics across a wide range of subjects, and show our statistics based retargeting significantly improves quality and efficiency over conventional MRF models.**

***Index Terms*—3D content creation; sparse offset statistics; graph-cut**

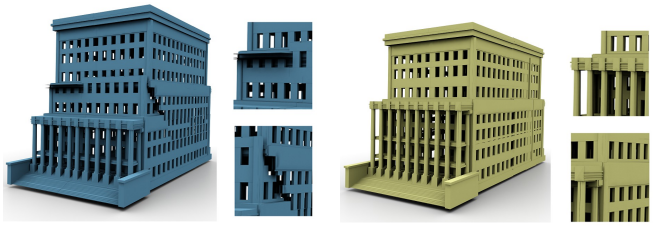
I. Introduction

Content creation is a central problem of compute graphics: Reducing the effort required for creating 3D models has been a long standing challenge in the field. In recent years, example-based model synthesis has received a lot of attention, in particular motivated by fast growing model repositories that could provide rich training data.

Synthesizing shapes from examples involves two key challenges: First, we have to characterize the shape space of plausible variants of the example data. Second, we also need an efficient algorithm to explore the shape space.

A necessary requirement for plausibility is local consistency: Shapes should form closed surfaces that resemble the input data at a local scale. This notion is captured by Markovian texture synthesis methods [26]. However, local consistency is not sufficient to characterize meaningful shapes (Figure 1-a); synthesizing complex models (beyond "texture" with stationary statistics) require additional structure to be maintained.

† These authors contributed equally to this work.



(a) 4992 nodes, 64 labels, 8.035 sec. (b) 378 nodes, 8 labels, 0.195 sec.

Fig. 1: 3D retargeting without (a) and with (b) offset statistics. Close inspections are shown on the right. The complexity of the the graph-cut problem and the corresponding optimization time are reported for each result.

In this paper we propose a simple but effective method to retarget 3D models using the statistics of low-level translational symmetries. Our method is inspired by the recent success of offset statistics in image inpainting [9] and editing [19]. Intuitively, it uses an MRF model to stitch translational copies of the input model, where the optimal stitching can be found using graph-cut based optimization. However, the method does not consider arbitrary translations but only those that are aligned with salient, frequently observed offsets between matching features. This additional regularization helps enforcing a more plausible global structure.

The key insight of our paper is the sparsity of offset statistics reveals important structural redundancy and can be used for searching good retargeted shapes. Frequently appearing offsets represent dominate repetitive structures in the model. They also indicate promising directions and spacings for retargeting the model. To utilize this information, we set up a transformation space that is the linear span of the dominant offset vectors. We use transformations inside this space to translate the input model. In this way, the structural regularities within the input model influence the distribution of the translational copies, so are more likely to be preserved in the retargeted model (Figure 1-b). In the meantime, since the number of dominant offsets is relatively small (also due to the sparsity), the retargeting is significantly more efficient (c.f. Figure 1).

A major advantage of our method is the low-level approach, which is simple and robust. In contrast, a lot of recent work has used much stronger regularity models, such as explicit algebraic models [6] or symmetry hierarchies [25], [11]. However, these methods require either clean data with exact regularity, or human assistance for pre-segmentation [13], [28], limiting the applicability in practice. While our method is limited by a simpler structure representation, texture synthesis can easily accommodate for imperfect regularity and other data imperfections, thereby complementing previous methods.

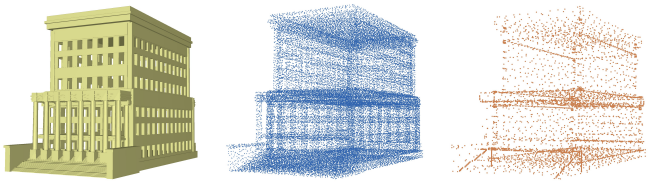
The main contributions of our paper include (i) an empirical study that confirms the sparsity of offset statistics in man-made 3D shapes, (ii) an algorithm for detecting dominant offsets from single input models, and (iii) a 3D model synthesis algorithm that utilizes offset statistics for automatic, high quality model retargeting at interactive speed.

II. Related Work

In this section we briefly review related work in 3D model synthesis, considering assembly and retargeting.

Model assembly: This category of methods decompose models into parts and recombine parts from different models into new shapes. Early work [8] allowed parts to be assembled interactively to form new shapes. Shapes can also be created by interpolating parts from two exemplar shapes, based on parts matching [11] or hand coded substructure [28]. More recently, large collection of shapes have been considered. Xu et al. [27] evolve an initial population of 3D models to produce generations of novel shapes, Averkiou et al. [1] analyze unorganized collections of 3D models to facilitate exploratory shape synthesis using high-level feedback. Kalogerakis et al. [12] use a generative probabilistic model of shape structure trained on a set of compatibly segmented shapes to learn the structural variability.

Rertarget based methods: Early work applied MRF-based texture synthesis to 3D modeling. For example, Merell et al. [15] stitch geometry using adjacency constraint to preserve the local consistency. However, MRF-based 3D model synthesis suffers from the lack of knowledge of higher level object structure, and the computational cost of generating detailed models is high [16]. In the past decade, great effort has been made to exploring shape structure for high quality 3D synthesis. For example, symmetry [17], [18] and partial symmetry [5], [4] are extracted from input models and used to regularize the synthesis model. This forms the basis for inverse procedural modeling, which represents families of shapes by automatically inferred context-free grammars [5]. Bokeloh et al. [6] propose an algebraic model that describes shapes in terms of regular patterns, interlinked with fixed topology. The shape space is parameterized by a small set of numerical parameters. While the structure model is richer than ours, the method requires exact regularity (grids of at least 3×3 rigidly identical copies each) to be applicable. More complex (architectural) models can be handled using manual user annotations [13].



(a) Input model (b) Dense samples (c) Features

Fig. 3: Sampling from input model.

Texture synthesis: Our paper revisits 3D texture synthesis, drawing inspirations from fruitful progress recently made in the 2D case [26]. Here, various forms of guidance have been employed to overcome the limitations of the MRF models. This can be achieved using an additional image layer with user annotations [10] or on-the-fly user interaction [2]. For image retargeting, results can be improved by better metrics for comparing MRFs. For example, [21] uses bidirectional matching and a gradual resizing procedure for retaining the image structure in retargeting. Our work is particularly inspired by the recent success of image statistics in image editing [19] and inpainting [9]. By applying this idea to 3D synthesis, we obtain plausible shape variations that previous 3D texture synthesis algorithms where not able to produce without user-guided analysis. This, together with its conceptual simplicity, which leads to robustness and easy implementation, makes our method appealing as a novel building block for 3D shape synthesis.

III. Method

Our method has two main steps: Offset statistics detection and model retargeting (see Figure 2 and 4).

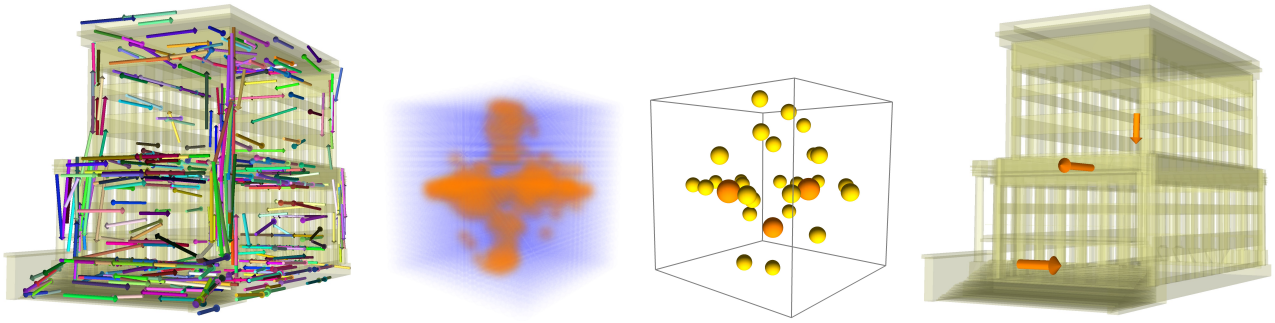
A. Offset Statistics Detection

In this step we compute offset statistics from the input triangle mesh. To compute features and their descriptors, we first perform Poisson disc sampling at two different scales (Figure 3). Low density samples are stored as features and the high density samples are used for computing feature descriptors. We include all mesh vertices as features to improve the matching. For each feature $\mathbf{x}$, we compute its SHOT [23] descriptor, denoted by $S(\mathbf{x})$.

Next, we match similar features in the input model based on their descriptors. For each feature $\mathbf{x}$, we find the offset vector $\mathbf{v}$ to its best match:

$$\mathbf{v} = \arg\min_{\mathbf{v}} |S(\mathbf{x}+\mathbf{v}) - S(\mathbf{x})| \quad s.t \;\; |\mathbf{v}| > \tau \qquad (1)$$

Here $\mathbf{v} = (u, v, w)$ is the translational mapping from feature $\mathbf{x}$ to its closet match. Figure 2-a illustrates some example mappings (only 10% of the mappings are shown for clarity). The similarity between two SHOT feature descriptors is measured by the L2 norm $|\cdot|$. The threshold τ (set to 0.15 in our experiments) avoids matchings between features that are too close. As noted by [9], such a minimum distance constraint is useful for avoiding trivial statistics.



(a) Feature Matching (b) 3D histogram (c) Histrogram peaks (d) Principle offsets

Fig. 2: Our analysis step a) automatically matches features, b-c) find dominates offsets, and d) extracts up to three principle offset vectors.

Next, we compute offset statistics from the matched features. To do so we bin all offsets into a 3D histogram, as shown in Figure 2-b. Bins with strong magnitude are rendered in warmer color. We then detect peaks in the histogram using standard non-maximum suppression (Figure 2-c) with a radius of 0.15. The detected peaks (yellow) are the dominate offsets.

In practice we usually have about 20 dominate offsets detected from a model. Although it is possible to keep all of them for later retargeting, this is not necessary as many of them are obtained by combining very few generator vectors, which we call *principle offsets*. (orange spheres / arrows in Figure 2-c, 2-d). Principle directions span a regular grid of offsets (a translational k-parameter group [18], [6]), providing the candidate offsets considered for model stitching. As our method aims at retargeting (i.e., resizing in principal directions), we keep up to three principle offsets for a single model (which is also the maximum number of independent generators of a single lattice [18]). In consequence, this approach more or less guarantees correct results if a 3D model actually contains a dominant grid pattern (Figure 4). However, graph-cut-based stitching, as a global optimization method, will often still find plausible solutions for models of much more complex structure (such as the top three examples in Figure 9).

In the next section we continue with the core retargeting algorithm, assuming the principle offsets are known. In Section V we provide details about how to extract the principle offsets.

B. Model Retargeting

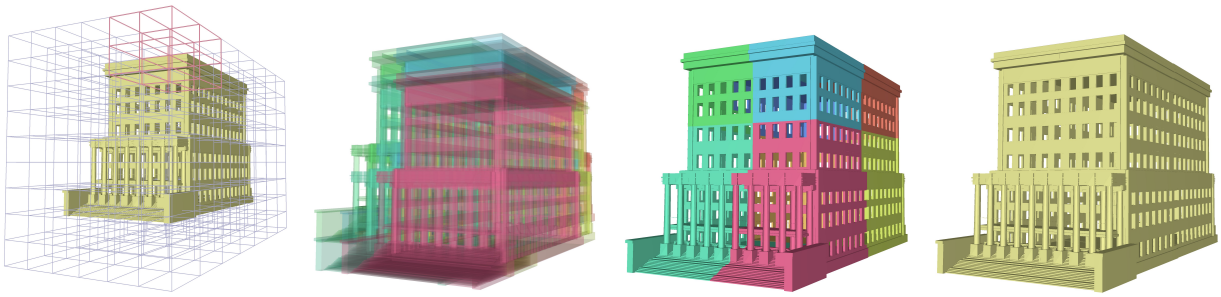
The basic idea is to retarget models by stitching together translational copies of the input model. Intuitively, one can discretize the retargeting space into units, such as regular voxels. Each voxel can be assigned with a label that indicates which translational copy's geometry is used for that voxel. The assignment of the labels can be solved using the standard multi-label graph-cut-based optimization [7]. However, stitching shapes together is non-trivial in practice, because graph-cut-based geometry sticthing can produce implausible shapes if the transformation between the two model copies to be compined is not chosen carefully. In our context, it is impossible to stitch shapes into a good model if the boundary of translational copies are not aligned with grid border. In theory, one can use larger number of copies to increase the chance of getting a good synthesized model (and such as strategy would also involve a yet to be defined criterion for rejecting bad offsets). However, this is not practical as the optimization cost quickly increases with the resolution of the discretization (number of nodes and the number of labels in the graph). In the meantime, even with a very fine discretization, graph-cut geometry synthesis can still produce invalid shapes as the structure of the input model is not being considered.

Our key insight is to use offset statistics to set up a proper transformation space preserving the structure of the input model. To do so, we define the *offset space* Φ as the span of the pre-detected principle offset vectors:

$$\Phi = \left\{ \sum_{i=1}^{K} \lambda_i \mathbf{v}_i | \lambda_i \in \mathbb{Z} \right\} \tag{2}$$

As previously discussed, the principle offsets $\{\mathbf{v}_i\}_{i=1}^{K}$ are generators for resizing a model in different canonical directions. The $\mathbb{Z}$ in Eq. 2 is the set of integer numbers, so λ_i is the number of integer steps to resize a model using $\mathbf{v}_i$. We restrict λ to integer values for discrete optimization. In this paper we use three principle offsets ($K = 3$) so the offset space contains $M = U \times V \times W$ translational copies of the input model, where $\{U, V, W\}$ are the steps for different principle offsets. Each copy has a unique offset label $\{l_i\}_{i=1}^{M} = (u_i, v_i, w_i)$, indicating the translational mapping between the copy and the original model in the units of principle offsets. For example, $l = (1, 0, 0)$ indicates the copy is shifted by one step in the first principle direction. It is clear that M is also the number of labels for the later multi-label graph-cut optimization.

We now define the *retargeting space* Ω as the volume of the synthesized model. We discretize Ω into a regular volume, once again, using the principle offsets as the generators so it is consistent with the offset space Φ. Specifically, the retargeting space has $N = (U_{in} + U) \times (V_{in} + V) \times (W_{in} + W)$ voxels, where U_{in}, V_{in}, W_{in} are the respective dimensions of the input model (in the units of principle offsets). Figure 4-a shows an input model, imposed by a $2 \times 2 \times 2$ offset space (red) and the resulting retargeting space (gray). Figure 4-b shows a montage



(a) Space discretization (b) Offset montage (c) Graph cut (d) Retargeting result

Fig. 4: Offset based retargeting.

of all translational copies rendered with different colors. These copies together fill up the retargeting space.

To stitch all copies together, we assign each voxel in Ω a offset label: $L(\mathbf{x}) = l_i$. Intuitively, it means that voxel $\mathbf{x}$ in the synthesized model has its geometry copied from voxel $\mathbf{x} - (v_i, u_i, w_i)$ in the input model. We find the optimal labeling assignment by minimizing the following MRF energy function:

$$E(L) = \sum_{\mathbf{x}\in\Omega} E_u(L(\mathbf{x})) + \sum_{(\mathbf{x},\mathbf{x}')|\mathbf{x},\mathbf{x}'\in\Omega} E_p(L(\mathbf{x}), L(\mathbf{x}')) \quad (3)$$

The unary term E_u is 0 if the label is valid (that is, $\mathbf{x}-(v_i, u_i, w_i)$ is a known voxel in the input model). Otherwise the term will receive infinite cost. The pairwise term E_p encourage smoothness between adjacent voxels in the 26-connected neighborhood. For each pair of neighboring voxels $(\mathbf{x}, \mathbf{x}')$ with label assignment $L(\mathbf{x}) = l_i$ and $L(\mathbf{x}') = l_j$, we define E_p as:

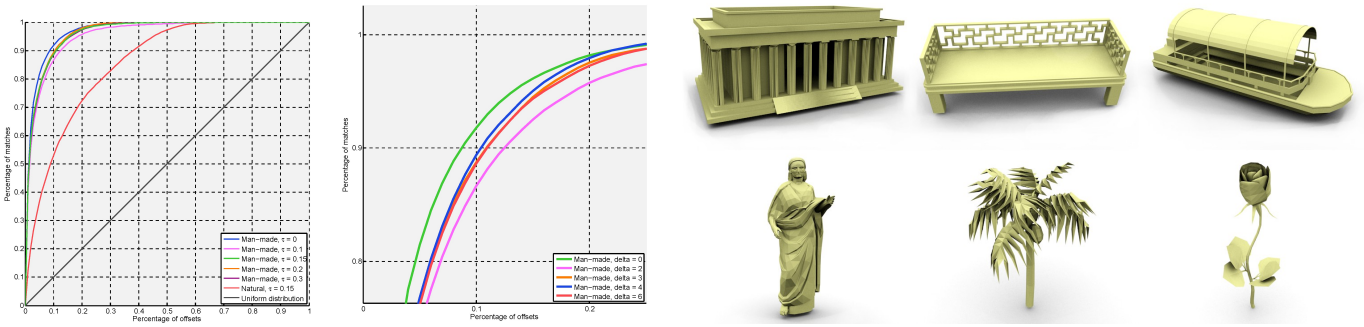
$$E_s(l_i, l_j) = d_H(G(\mathbf{x}+l_i), G(\mathbf{x}+l_j)) + d_H(G(\mathbf{x}'+l_i), G(\mathbf{x}'+l_j)) \quad (4)$$

This energy function penalizes stitching together voxels that have different geometries. Here $G(\mathbf{x})$ is the shape descriptor for voxel $\mathbf{x}$. We simply use the set of SHOT descriptors contained in the voxel. The similarity between two voxels are then computed as the Hausdorff distance d_H between the two sets, while similarity between individual SHOT features is measured using the L_2 norm. Notice the pairwise feature distances have already been computed in the previous step (Section III-A) and can be reused here. In practice it often contains empty voxels. In this case we assign zero distance if both voxels are empty, or infinite distance if only one of them is empty. It is easy to see E_s is submodular, so we can optimize Equation 3 using the alpha-expansion algorithm [7], which produces optimal label assignments for the retargeting space N. Figure 4-c, d show the synthesized model output by the graph-cut optimization, where colors in Figure 4-c reveal the offset labels assigned to different parts of the model. In Section VI we will show more results and compare our method with an alternative method.

IV. Sparsity of 3D Offset Statistics

The sparsity of offset statistics is the theoretical foundation for our method to work. In this section, we give some empirical evidence for the validity of this assumption using a study similar to [9]. We collect two datasets of 3D shapes, one for man-made objects and the other for natural objects. The man-made dataset (Figure 5-c, top row) contains 200 models, including architectures, furniture, vehicle etc. The natural dataset (Figure 5-c, bottom row) contains 120 models, such as human, animals and foliage. All models are pre-normalized and centered so the longest dimension of each model is scaled to $[-1, 1]$. As previously described (Figure 3), a dense point cloud is sampled for each model using Poisson sampling with 0.02 as the sampling space. A feature point cloud is then re-sampled from the dense point cloud using 0.06 as the sampling space. A 352 dimensional SHOT descriptor is computed for each feature. We use a fairly large radius (set to 0.2 in our experiment) to encode sufficient geometric information in the SHOT descriptors. For each feature, we find its closest match as defined in Equation 1. We then bin the offsets between matched features and sort the bins in the descending order of their magnitude (number of offsets in a bin). We use bin of size 0.05 in each dimension, and test different minimum distance thresholds τ (Figure 5-a, b) for precluding trivial infinitesimal similarity case.

In order to examine the sparsity of the offset statistics, we accumulate the bins and plot the cumulative distribution (averaged over all models) in Figure 5-a. The diagonal line (gray) is the cumulative distribution for a uniform statistics, indicating the most non-sparse distribution. In comparison, man-made objects have a clearly sparse distribution as the curve is highly non-linear. For example, when $\tau = 0.15$ (green curve), 79.9% of the offsets are distributed in only 7% of the bins, 97.1% of the offsets are distributed in 20% of the bins. We also observe that τ has little influence on the distribution – as the curves for different τ are very similar. This can be better seen in the enlarged plot (Figure 5-b). Notice although the curve of $\tau = 0$ (blue) is relatively more distanced to the others, the difference is not as significant as being observed in 2D images previously [9]. This is because the 3D models have higher accuracy so are less likely to have false positive matchings. Nonetheless in practice we still use $\tau = 0.15$ to eliminate the possibility of getting trivial peaks around $(0, 0, 0)$ in the 3D histogram. Figure 2 showed an exemplar model where three principle offsets are detected. Figure 6 shows



(a) Cumulative distribution (b) Zoom-in (c) Datasets: man-made (top) and natural (bottom) subjects

Fig. 5: Offset sparsity.

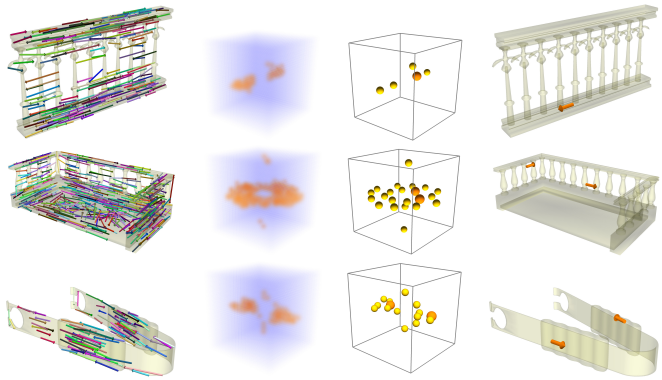


Fig. 6: Offset statistics for different structures. Top: 1D grid, Middle: 2D grid, Bottom: sheared 2D grid.

cases where only one or two principle offsets can be found in a model.

It is worth mentioning that the sparsity is not as pronounced for natural objects, as the red curve shows. For example, only 40.4% (compare to 79.9% of the man-made objects) of the offset are distributed in 7% of the bins. Although there is still some sparsity, as the cumulative curve is above the diagonal plot of an uniform distribution, we find it gives little advantage for later model retargeting.

V. Implementation

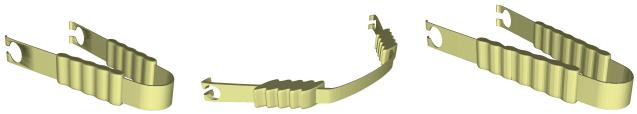
In this section we provide implementation details of our algorithm. In this paper we use the popular SHOT descriptor for matching features. But other descriptors, such as FPFH [20], can also be used. Since we only consider translational offsets, we do not need rotationally invariant feature matching. We therefore fix the local reference frame of each feature to be aligned with the world coordinate frame. Once features are matched, we bin them into a 3D histogram, where each bin is a cube of 0.05 on each dimension. We detect peaks using non-maximum suppression with a local window size of 0.15 (consistent with the value of τ in Equation 1). Each peak represents a dominant offset in the model. Since the dominant offsets are not very accurate at this stage, as they are defined at the resolution of the bins, we increase their precision using a global iterative closest point algorithm that is applied on the entire point cloud.

Next, we find up to three principle offsets (c.f. Section III-A) using a greedy selection strategy on a list of dominant offsets sorted in decreasing order of their histogram magnitudes. We select the first dominant offset in the list as the first principle offset $\mathbf{v}_1$, since it has the most matched features. The second principle offset $\mathbf{v}_2$ is the first remaining dominant offset that satisfies $-\cos\theta \leq \frac{\mathbf{v}_1 \cdot \mathbf{v}_2}{|\mathbf{v}_1||\mathbf{v}_2|} \leq \cos\theta$, which specifies a minimum angle between two principle offsets. In practice we set $\theta = \pi/3$. The third principle offset, if there is any, is set to be the first remaining dominant offset that satisfies the minimum angle check with both the first and the second principle offsets. Notice the number of detected principle offsets determines the degree of freedom for model retargeting. For example, if only one principle offset is detected, the model can only be resized in one direction.

For the graph-cut optimization, we discretize the embedding space of the retargeted model into a regular volume Ω as used in Eq. 3. The discretization uses principle offsets as the generators, so each voxel is of size $\{|\mathbf{v}_1|, |\mathbf{v}_2|, |\mathbf{v}_3|\}$, where $|\cdot|$ denotes the L2 norm. In cases where there are less than three principle offsets, we create up to two orthogonal dummy directions with length 0.2. These supportive offsets are only used to discretize the model, no retargeting will be performed along these directions.

To generate new models, we stitch together translational copies of the input model, as described in section III-B. The translations are restricted to multipliers of the voxel size for two important reasons: First, to keep the translations consistent with the repetitive structure of the input model. Second, to keep the number of labels for graph-cut optimization low for computational efficiency. In fact, the graph-cut is usually run on a graph of less than 1000 nodes with 30 labels, so the retargeting can be performed at an interactive speed. Despite using such relatively low complexity graph, our algorithm is still able to keep the retargeting quality high and preserve the exemplar structure in the synthesized models.

In practice we map the input model into a canonical space (i.e. 3D Cartesian coordinate system) if the detected principle offsets are not orthogonal to each other. The mapping is



(a) Input model (b) Canonical space (c) Final output

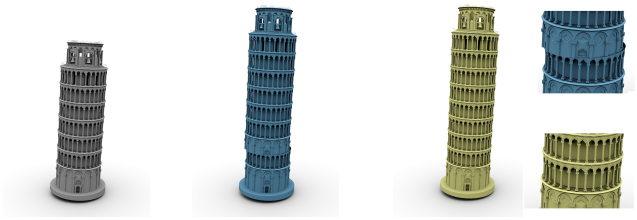
Fig. 7: A model with sheared 2D grid (a) is mapped into a canonical space (b) where the two principle offsets are orthogonalized. The retargeting is performed in the canonical space and mapped back to the normal space (c).

constructed by collecting all 3 principle offset vectors into a 3D transformation matrix, which is then applied to all model vertices. Figure 6 (bottom) shows an example where the object has two non-orthogonal principle offsets. Doing so allows the discretization of the model to be axis-aligned for fast geometry computation. Figure 7-a and 7-b show the same model in the ordinary space and in the canonical space. The retargeting is performed in the canonical space, and the result is mapped back to the ordinary space for displaying to users (Figure 7-c).

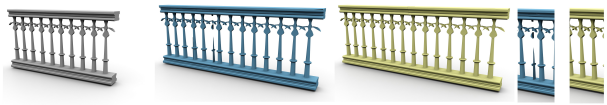
VI. Results

In this section we show some of our retargeting results. Figure 8 shows side-by-side comparisons between our results (yellow) and the conventional MRF based results without offset statistics (blue). The gray models are the input examples. The exact number of graph nodes, number of labels, and time for graph-cut optimization are also provided for each result. To generate the blue models, regular offset (set to 0.1) is used. Despite using significantly more complex graphs, conventional MRF produced shapes that do not preserve the structure of the input models, as highlighted in the close inspections. In contrast, our method uses simpler graphs and does not produce such artifacts. Our optimization (yellow models) is fast enough for real time modeling (as demonstrated in the supplementary video).

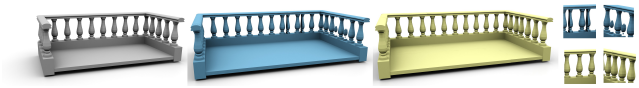
Figure 9 shows a gallery of our results. Notice the book shelf model (the first example) would be difficult to accomplish by methods such as [6]. Figure 10 shows the detected degrees of freedom (DoF) for resizing this model using Bokeloh et al's method [6]. As readers can see, the detection only covers a small part of the model due to irregularities in the shape. This lack of DoF can be an obstacle of synthesizing plausible new shapes. In particular, it often locks the entire shape from being changed. In contrast, a MRF based method such as ours is less rigid and can still produce convincing results by synthesizing the model as a "texture" and preserve fuzzy repetitions using offset statistics. Compare to other retargeting methods that explicitly model shape structure [22][14][15], our method is more flexible in different ways: we do not require prescribed procedural rules [22] nor manual decomposition [14], and perform less sensitively to the discretization of the shape space [15]. Compare to low level texture synthesis methods such as [24][3][29], our method handles man-made structures better due to the capture of offset statistics.



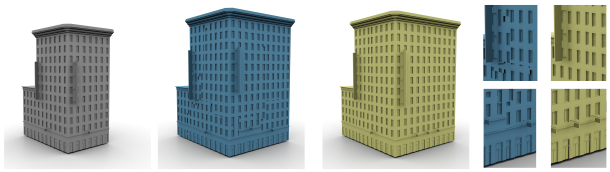
(a) Blue: 1600 nodes, 5 labels, 0.139 sec. Yellow: 360 nodes, 3 labels, 0.012 secs.



(b) Blue: 924 nodes, 8 labels, 0.082 sec. Yellow: 32 nodes, 6 labels, 0.008 secs.



(c) Blue: 2625 nodes, 25 labels, 0.956 sec. Yellow: 104 nodes, 9 labels, 0.013 secs.



(d) Blue: 7429 nodes, 54 labels, 5.151 sec. Yellow: 810 nodes, 8 labels, 0.072 secs.

Fig. 8: Qualitative comparison between retargeting without (blue) and with (yellow) offset statistics.

However, our current implementation has limitations for retargeting more complex structures. Figure 11 shows an example. In this case, the offset statistics is less sparse (top row, middle) and there are more than three dominant offsets (top row, right) that influence the structure of the model. We believe such a shape needs to be modeled by multiple grids, otherwise the retargeting produces artifacts as shown in Figure 11-b[1]. It is an interesting future work to extend our method to work with such more complexly structured models.

VII. Conclusion

Our paper explores offset statistics for 3D shape retargeting. The main contribution is to detect structural redundancy as sparse offset statistics, and use it to reduce the search space for good shapes. Our method works fully automatic, and produces significantly better results than conventional MRF

[1]This result uses the top three dominant offsets for discretizing the retargeting space into a single regular volume, and the top seven dominant offsets for generating translational copies of the input model.

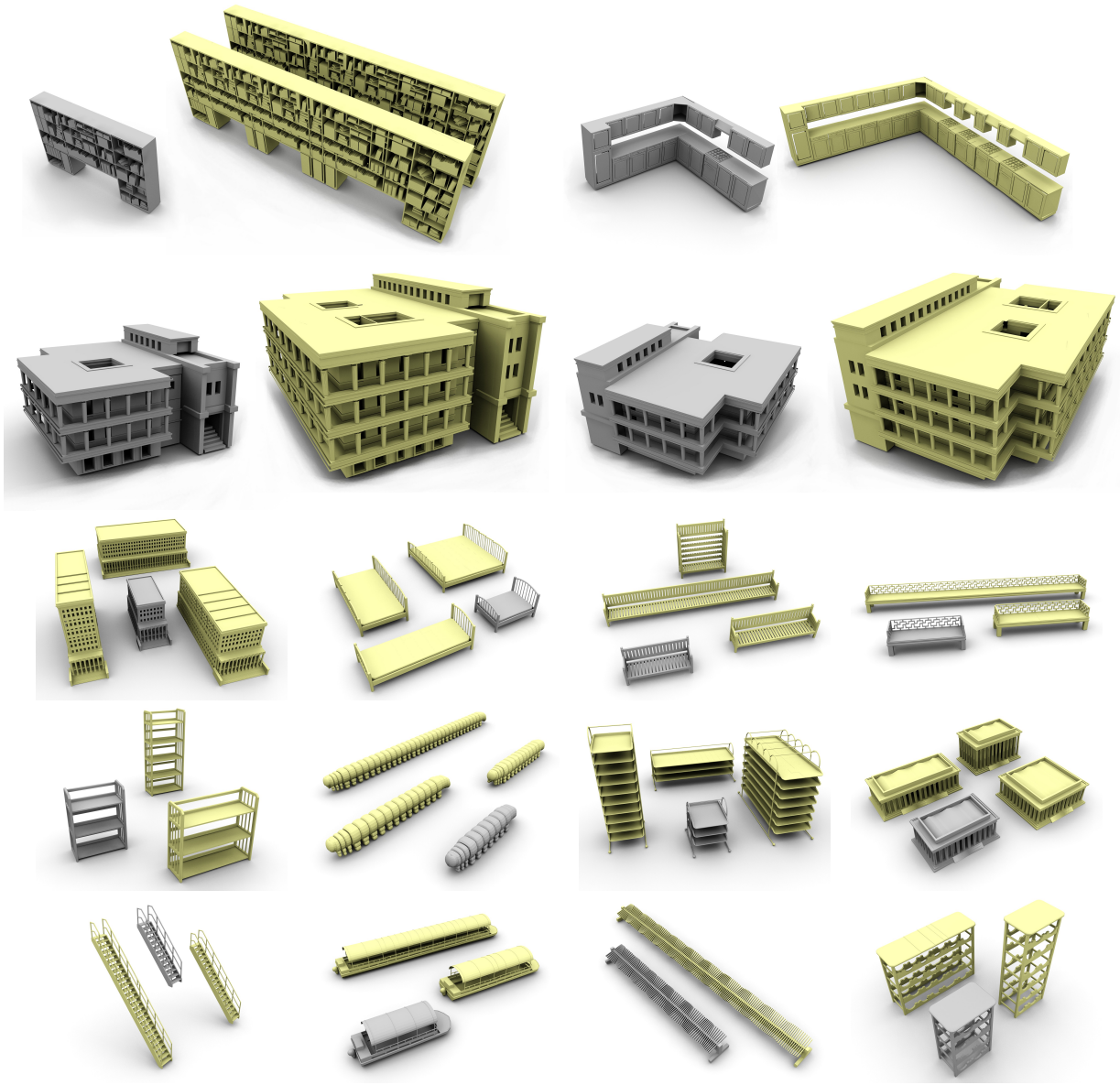


Fig. 9: Retargeting results.

models that do not use offset statistics. Nonetheless, in comparison to recent strongly structure-based methods (such as strictly symmetry- or regularity-based approaches), it retains the simplicity and generality of a texture-synthesis algorithm, in particular the ability to handle imperfect regularity. The computational cost is kept sufficiently low for real time modeling.

Our method is a small step to advance conventional MRF models for 3D modeling. It also opens many interesting future avenues. For example, how to model more complex shapes that can not be represented by a single grid. One possible solution might be to first segment the model into multiple grids, and use a second layer for capturing the relationship between these grids. Another interesting future path is to extend the model to non-translational mappings, for example rotation and scaling. Finally, it is interesting to apply offset statistics to applications such as structure from motion and shape completion.

VIII. Acknoledgement

This work has been partially supported by the Intel Visual Computing Institute, the Max-Planck-Center for Visual Computing and Communication, the international Max Planck Research School for Computer Science and Microsoft Research Cambridge. We thank Pushmeet Kohli and Martin Bokeloh for inspiring discussions. The input models are from the *Digimation Model Bank Library*.

References

[1] M. Averkiou, V. G. Kim, Y. Zheng, and N. J. Mitra. Shapesynth: Parameterizing model collections for coupled shape exploration and synthesis. *Comput. Graph. Forum*, 33(2):125–134, 2014.

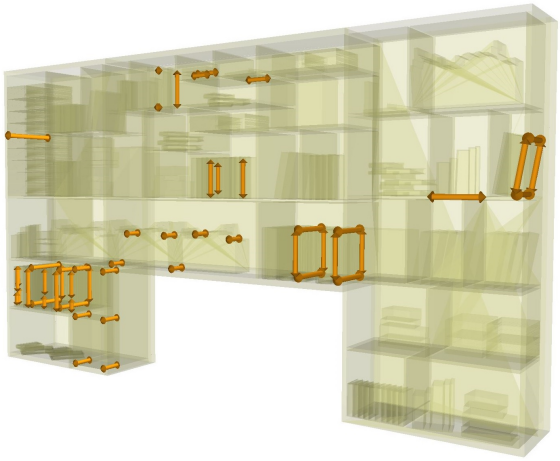
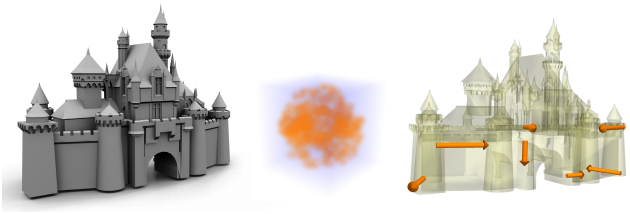
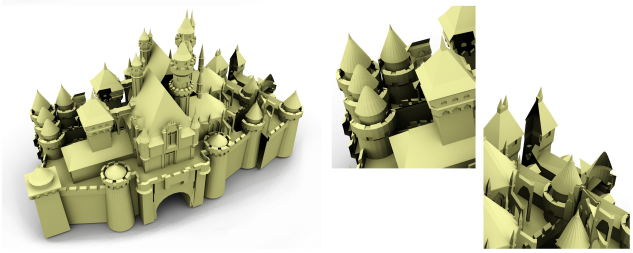


Fig. 10: Linear constraints detected by [6].



(a) Input model, 3D histogram and dominant offsets.



(b) Synthesized model with artifacts in close view.

Fig. 11: Limitation of our method.

[2] C. Barnes, E. Shechtman, A. Finkelstein, and D. B. Goldman. Patchmatch: A randomized correspondence algorithm for structural image editing. *ACM Trans. Graph.*, 28(3):24:1–24:11, July 2009.

[3] P. Bhat, S. Ingram, and G. Turk. Geometric texture synthesis by example. In *Proceedings of the Symposium on Geometry Processing*, pages 41–44, 2004.

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

[5] M. Bokeloh, M. Wand, and H.-P. Seidel. A connection between partial symmetry and inverse procedural modeling. *ACM Trans. Graph.*, 29(4), 2010.

[6] 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.

[7] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE TPAMI*, 23(11):1222–1239, 2001.

[8] T. Funkhouser, M. Kazhdan, P. Shilane, P. Min, W. Kiefer, A. Tal, S. Rusinkiewicz, and D. Dobkin. Modeling by example. In *ACM Trans. Graph. (Proc. Siggraph)*, pages 652–663, 2004.

[9] K. He and J. Sun. Statistics of patch offsets for image completion. In *ECCV*, pages 16–29, 2012.

[10] A. Hertzmann, C. E. Jacobs, N. Oliver, B. Curless, and D. H. Salesin. Image analogies. In *SIGGRAPH*, pages 327–340, 2001.

[11] A. Jain, T. Thormählen, T. Ritschel, and H.-P. Seidel. Exploring Shape Variations by 3D-Model Decomposition and Part-based Recombination. *Computer Graphics Forum*, 31(2):631–640, 2012.

[12] E. Kalogerakis, S. Chaudhuri, D. Koller, and V. Koltun. A Probabilistic Model of Component-Based Shape Synthesis. *ACM Transactions on Graphics*, 31(4), 2012.

[13] J. Lin, D. Cohen-Or, H. Zhang, C. Liang, A. Sharf, O. Deussen, and B. Chen. Structure-preserving retargeting of irregular 3d architecture. *ACM Trans. Graph.*, 30(6):183:1–183:10, 2011.

[14] J. Lin, D. Cohen-Or, H. Zhang, C. Liang, A. Sharf, O. Deussen, and B. Chen. Structure-preserving retargeting of irregular 3d architecture. *ACM Trans. Graph.*, 30(6):183:1–183:10, Dec. 2011.

[15] P. Merrell and D. Manocha. Model synthesis: A general procedural modeling algorithm. *IEEE TVCG*, 17(6):715–728, 2011.

[16] P. C. Merrell. Model synthesis. *PhD Thesis, Stanford University*, 2009.

[17] N. J. Mitra, L. Guibas, and M. Pauly. Partial and approximate symmetry detection for 3d geometry. *ACM Transactions on Graphics (SIGGRAPH)*, 25(3):560–568, 2006.

[18] M. Pauly, N. J. Mitra, J. Wallner, H. Pottmann, and L. J. Guibas. Discovering structural regularity in 3d geometry. *ACM Trans. Graph.*, 27(3), 2008.

[19] Y. Pritch, E. Kav-Venaki, and S. Peleg. Shift-map image editing. In *ICCV*, pages 151–158, 2009.

[20] R. B. Rusu, N. Blodow, and M. Beetz. Fast point feature histograms (fpfh) for 3d registration. In *ICRA*, pages 3212–3217, 2009.

[21] D. Simakov, Y. Caspi, E. Shechtman, and M. Irani. Summarizing visual data using bidirectional similarity. In *CVPR*, 2008.

[22] J. O. Talton, Y. Lou, S. Lesser, J. Duke, R. Měch, and V. Koltun. Metropolis procedural modeling. *ACM Trans. Graph.*, 30(2):11:1–11:14, Apr. 2011.

[23] F. Tombari, S. Salti, and L. di Stefano. Unique signatures of histograms for local surface description. In *ECCV*, pages 356–369, 2010.

[24] G. Turk. Texture synthesis on surfaces. In *Proceedings of SIGGRAPH*, pages 347–354, 2001.

[25] Y. Wang, K. Xu, J. Li, H. Zhang, A. Shamir, L. Liu, Z. Cheng, and Y. Xiong. Symmetry hierarchy of man-made objects. *Computer Graphics Forum (Proc. EUROGRAPHICS)*, 30(2), 2011.

[26] L.-Y. Wei, S. Lefebvre, V. Kwatra, and G. Turk. State of the art in example-based texture synthesis. Eurographics STARs, March 2009.

[27] K. Xu, H. Zhang, D. Cohen-Or, and B. Chen. Fit and diverse: Set evolution for inspiring 3d shape galleries. *ACM Trans. Graph.*, 31(4):57:1–57:10, 2012.

[28] Y. Zheng, D. Cohen-Or, and N. J. Mitra. *Smart Variations*: Functional substructures for part compatibility. *Comput. Graph. Forum*, 32(2):195–204, 2013.

[29] K. Zhou, X. Huang, X. Wang, Y. Tong, M. Desbrun, B. Guo, and H.-Y. Shum. Mesh quilting for geometric texture synthesis. *ACM Trans. Graph.*, 25(3):690–697, July 2006.