APPENDIX OF RESEARCH ARTICLE

MorphCut: Efficient Convex Decomposition of 3D Building Models for Urban Morphological Analytics

Yijie Wu[a, d, e], Fan Xue[a,b*], Liangliang Nan[c], Longyong Wu[a], Jantien Stoter[c], and Anthony G. O. Yeh[a,b]

[a]Faculty of Architecture, The University of Hong Kong, Pokfulam, Hong Kong, China;
[b]National Center of Technology Innovation for Digital Construction Hong Kong Branch, The University of Hong Kong, Pokfulam, Hong Kong, China;
[c]Urban Data Science, Delft University of Technology, 2628 BL, Delft, The Netherlands;
[d]JC STEM Lab of Earth Observations, Department of Land Surveying and Geo-Informatics, The Hong Kong Polytechnic University, Hung Hom, Hong Kong, China;
[e]Research Centre for Artificial Intelligence in Geomatics, The Hong Kong Polytechnic University, Hung Hom, Hong Kong, China.

ARTICLE HISTORY
Compiled August 30, 2025

*Corresponding author. Tel: +852 3917 4174, Fax: +852 2559 9457, Email: xuef@hku.hk

Appendix A. Algorithms of BBnB and Fast BBnB

Algorithm 1 presents the procedure of BBnB for building decomposition. Nodes that need further branching are stored in an ordered set $\mathcal{Q}$, which is sorted in ascending order according to $f'(\mathbf{n})$. $f'(\mathbf{n})$ denotes a redefined objective function for BBnB search, introduced in Section 3.3.2. Algorithm 2 presents Fast BBnB integrating the two acceleration mechanisms introduced in Section 3.3.3. In Algorithm 2, the objective function and node branching of Algorithm 1 are updated according to ϵ and l_i.

Algorithm 1: BBnB search for the convex decomposition of buildings

```
Input: A node $\mathbf{n}_0$ containing the singleton building cell $R$
Output: Optimal convex parts $\Theta$
$\mathcal{Q} \leftarrow \{\mathbf{n}_0\}$ ;                              // An ordered set $\mathcal{Q}$
$B \leftarrow +\infty$ ;                                // Upper bound initialized
while $\mathcal{Q} \neq \varnothing$ do
    $\mathbf{n} \leftarrow \mathcal{Q}$.pop();          // Pop the 'best' node with the lowest $f'(\mathbf{n})$
    $\mathcal{C} \leftarrow \mathbf{n}$.branch();                     // Branch of the node as Fig. 5
    for $\mathbf{n}_i \in \mathcal{C}$;                             // Loop for $i$-th node of $\mathbf{n}$
     do
        if $\mathbf{n}_i$ is a leaf and $f'(\mathbf{n}_i) < B$ then
            $B \leftarrow f(\mathbf{n}_i)$;                          // Update the bound $B$
            $\mathcal{Q}$.prune($B$);                 // Remove $\mathbf{n}_k$ in $\mathcal{Q}$ if $f'(\mathbf{n}_k) >= B$
            $\Theta \leftarrow$ Parts($\mathbf{n}_i$);
        else if $f'(\mathbf{n}_i) < B$ then
            $\mathcal{Q}$.add($\mathbf{n}_i$);
```

Algorithm 2: Fast BBnB search for the convex decomposition of buildings

```
Input: A building singleton cell $R$ of the building, coefficient $\epsilon$ of $h(\mathbf{n})$, and
       levels of detail $L = \{l_0, l_1, \ldots, l_L\}$
Output: Sub-optimal convex parts $\Theta$
$\mathcal{C} \leftarrow \{R\}$
for $l_i \in L$;                                 // Loop the $i$-th LoD $L$
 do
    $\mathcal{L} \leftarrow \varnothing$;                              // Parts of $l_i$ initialized
    for $c_j \in \mathcal{C}$;                          // Decompose $j$-th part in $C$
     do
        $\mathcal{L} = \mathcal{L} \cup$ BBnB($c_j$, $\epsilon$, $l_i$) ;   // $\epsilon$ and $l_i$ are two new parameters for
         Alg. 1
    $\mathcal{C} \leftarrow \mathcal{L}$;                  // Update parts for next level of detail
$\Theta \leftarrow \mathcal{C}$;
```

Appendix B. Supplementary details of methodology

B.1. Cutting plane proposal of concave edges

As shown in Fig. B.1, we categorize concave edges into six types based on their directions, which are detailed below.

(1) **Horizontal concave edges** are formed by two inclined planes (Fig. B.1(a)) or by a vertical and a horizontal plane (Fig. B.1(b)). Cutting planes always include the two forming planes. Additional horizontal and vertical planes are added for more regular decomposition.

(2) **Inclined concave edges** are formed by either two inclined planes (Fig. B.1(d)) or a vertical and an inclined plane (Fig. B.1(e)). The two forming planes are used as potential cutting planes.

(3) **Vertical concave edges** are formed by two vertical planes (Fig. B.1(f)). the cutting planes include both forming planes and one of their symmetric joint planes.

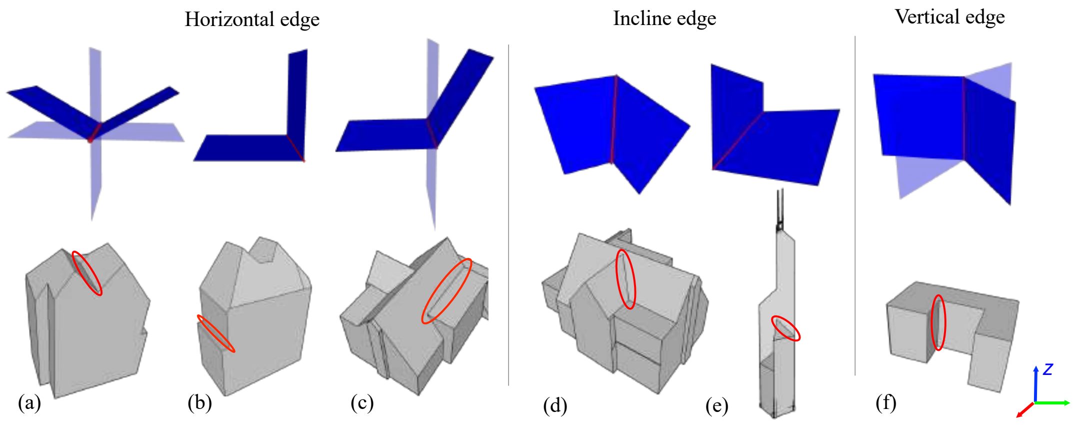


Figure B.1. Concave edges and potential cutting planes. We categorize the concave edges into six types depending on the line directions and the orientations of their intersecting planes. Planes of the adjacent faces on the building surfaces are shown in dark blue, while the light blue ones are additional planes for potentially better decomposition.

B.2. More examples of centralized and decentralized decomposition

Fig. B.2 presents more examples of centralized and decentralized decomposition.

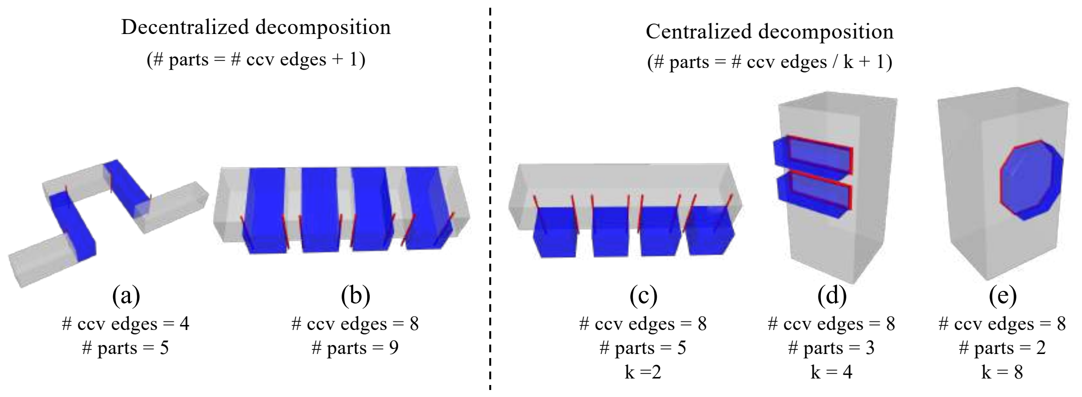


Figure B.2. Numbers of concave edges and decomposed parts in decentralized and centralized decomposition. # ccv edges and # parts refer to the number of concave edges and parts, respectively. k indicates the number of concave edges surrounding a part attached to a central part. Concave edges are highlighted as red segments.

Appendix C. Supplementary details of experiments

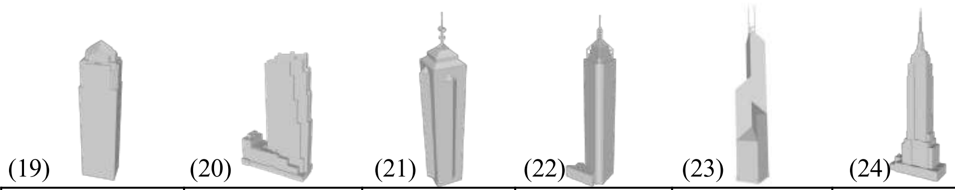
C.1. Details of selected samples from three datasets

The test samples and urban scenes in the experiments are selected from three open datasets (in alphabetical order):

- 3D-BIT00 (LandsD, 2024): 3D building models from Hong Kong, China. This dataset includes many high-rise buildings with complex footprints, created based on official records of architectural drawings from the Hong Kong government.
- City3D (Huang, Stoter, Peters, & Nan, 2022): 3D building models from the Netherlands, the City of Surrey in British Columbia, Canada, and Vaihingen, Germany. The dataset includes over 20,000 3D buildings automatically reconstructed from point clouds and footprints from AHD3 (AHN3, 2018), DALES (Varney, Asari, & Graehling, 2020), and Vaihingen (Rottensteiner et al., 2012) by the City3D algorithm (Huang et al., 2022).
- NYC 3D Model (DCP, 2018): 3D building models from New York City, United States. The dataset was publicly released by the Department of City Planning of New York City based on the aerial survey of 2014. It includes all buildings in New York as of 2014, with iconic structures reconstructed in greater detail.

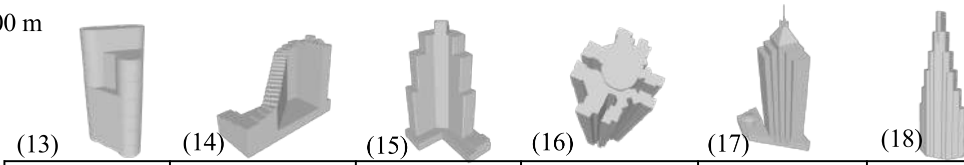
C.2. Method implementation

Our solution was implemented in C++ and compiled with the C++17 standard (ISO, 2014). In preprocessing, we obtained the manifold reference surfaces using ManifoldPlus (Huang, Zhou, & Guibas, 2020). Region growing, 3D triangulation, and mesh operation were implemented based on the Shape Detection package (Oesau et al., 2024), 3D Triangulation (Jamin, Pion, & Teillaud, 2024), and Combinatorial Map (Damiand, 2024) of CGAL (version 6.0), respectively. The ray intersection for interior and exterior point classification was implemented based on the open-source code by A. Yu and Shang (2019).



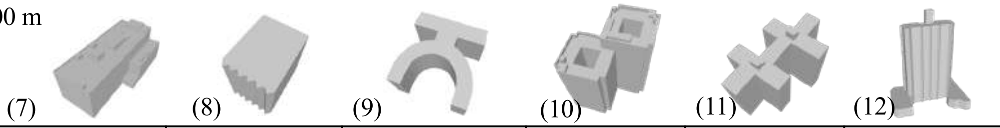
57*46*203	170*130*261	59*59*342	78*83*349	67*67*366	260*127*444
76, 117, 96	127, 160, 123	73, 61, 61	40, 26, 42	87, 86, 97	251, 402, 214
3DBIT00	NYC3D	3DBIT00	3DBIT00	3DBIT00	NYC3D

H=200 m



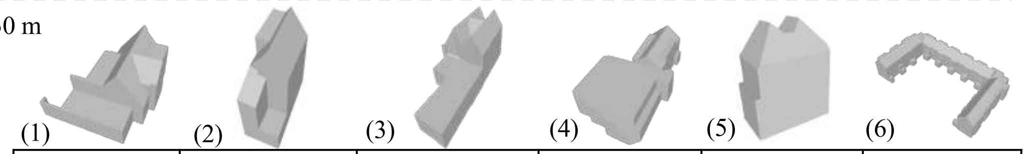
54*55*105	157*122*106	100*72*129	43*48*156	98*66*187	30*56*171
76, 54, 38	76, 59, 79	78, 83, 89	96, 64, 107	141, 122, 95	51, 22, 27
3DBIT00	NYC3D	NYC3D	3DBIT00	3DBIT00	3DBIT00

H=100 m



87*79*31	54*45*43	111*109*43	61*59*69	84*62*77	94*99*90
46, 33, 35	20, 9, 20	35, 12, 27	28, 23, 28	35, 21, 43	53, 49, 88
3DBIT00	3DBIT00	3DBIT00	3DBIT00	NYC3D	NYC3D

H=30 m



20*12*8	12*12*12	23*27*12	36*47*13	13*12*14	81*74*15
31, 21, 40	16, 8, 9	24, 18, 27	40, 23, 26	16, 3, 9	103, 127, 144
City3D	City3D	City3D	City3D	City3D	City3D

Figure C.1. Profile of the 24 selected samples. The first rows of the tables indicate the length*width*height in meters; the second rows display the numbers of planar regions, concave edges, and potential cutting planes for $l_i = 10^{-4}$; the third rows present the source datasets, City3D (Huang et al., 2022), 3DBIT00 (LandsD, 2024), and NYC3D (DCP, 2018).

C.3. Computational facilities

Experiments on the 24 selected samples were conducted on a MacBook Pro with the M1 chip (8 cores, 16GB RAM). The urban-scale validation was established on a workstation with an Intel Core i7 CPU (2.9 GHz, 8 cores) and 128 GB of RAM. Besides, as SAMPart3D requires GPU acceleration, we tested it on high-performance computing (HPC) facilities equipped with NVIDIA Tesla L40S 48 GB PCIe GPU.

C.4. Details of baseline method configurations

C.4.1. Detailed configurations for VHACD

VHACD has three major tunable parameters: (i) voxel resolution, (ii) convexity threshold, and (iii) number of parts. Based on empirical testing, we set the voxel resolution to VHACD's maximum allowable value of 10,000,000 for sufficient decomposition quality. For the convexity threshold and number of parts at each LoD, we referenced the corresponding values from MorphCut's results. Specifically, convexity thresholds were set to 0.2, 0.1, 0.05, and 0.01 for $L = \{10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}\}$, respectively. In our experiments, VHACD tended to produce significantly more parts than MorphCut. To ensure a fair comparison, we constrained VHACD to output the same number of parts as MorphCut at each LoD. As a result, the compactness values of VHACD matched those of MorphCut (see Fig. 7(a)). Moreover, VHACD is not robust in handling open-bottom building models. To offset this topological issues, we took the reconstructed singleton building cells as VHACD's inputs.

C.4.2. Detailed configurations for COACD

Similar to VHACD, we tuned the convexity thresholds and numbers of parts for COACD, while keeping other parameters—such as MCTS search depth and number of iterations—at their default values based on our empirical testing. The convexity thresholds were aligned with those used for MorphCut and VHACD. However, setting the threshold to 0.01 for the finest LoD significantly increased COACD's processing time on medium to high complexity buildings, without yielding the desired convexity. Therefore, we adjusted the threshold to 0.02 for the finest LoD. Despite this tuning, COACD occasionally produced far more parts than MorphCut at fine LoDs. To ensure a fair comparison, we also set the numbers of parts of COACD to be identical to MorphCut's. Conversely, at coarse LoDs, COACD tended to produce fewer parts than MorphCut, even when given the same convexity thresholds and target part counts, as shown in Fig. 7. Similar to VHACD, COACD also encountered topological issues when handling open-bottom building models. Thus, we took the singleton building cells as their inputs.

C.4.3. Detailed configurations for COMPOD-PSDR

COMPOD takes point clouds with detected planar regions as input. To prepare compatible input, we sampled the input meshes into point clouds with normals consistent with the meshes and detected planar regions using PSDR (Sulzer, Yu, & Lafarge, 2023; M. Yu & Lafarge, 2022), which is designed for high-quality plane detection and recommended by the authors of COMPOD. The sampling interval was set to 0.25 m to balance partitioning fidelity and efficiency, and the distance thresholds for plane detection matched those used in MorphCut's preprocessing. By default, COMPOD's merging process produces slightly more parts than MorphCut. To ensure fair comparisons, we predefined the number of parts for COMPOD at each LoD in $L = \{10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}\}$ to match those of MorphCut. However, due to COMPOD's local greedy merging strategy, the process often terminated early, resulting in more parts than predefined. Consequently, COMPOD-PSDR yielded higher normalized part counts than MorphCut, as shown in Fig. 7.

C.4.4. Detailed configurations for SAMPart3D

For SAMPart3D, most parameters were kept at their default settings. Sixteen rendered views were used for SAM-based segmentation. For each sample, training was required to generate 3D features, with the number of training epochs set to 5000 as recommended. We also tested a reduced setting of 2000 epochs for efficiency, which led to noticeable performance degradation. SAMPart3D provides four default LoD settings, which we adopted directly. However, these settings produced no significant differences in results. Therefore, we report only the results from one LoD and exclude SAMPart3D from comparisons involving LoD variation.

C.5. Details of evaluation metrics

Denoted the set of decomposed parts as Θ, the average convexity is calculated as:

$$\text{Convexity}(\Theta) = \frac{\sum_{p\in\Theta} \text{Convexity}(p)}{|\Theta|}, \tag{C1}$$

where Convexity(p) is the convexity of a part. Convexity(p) is commonly measured by the volume ratio of a part p (Mamou, Lengyel, & Peters, 2016) to its convex hull:

$$\text{Convexity}(p) = \frac{\text{Volume}(p)}{\text{Volume}(\text{CH}(p))}, \tag{C2}$$

.

Besides, we normalize the number of decomposed parts same as $f(\mathbf{n})$ defined in Eqn. 3, which is

$$\frac{|\Theta|}{|\text{ConcaveEdges}(R)| + 1}, \tag{C3}$$

where R refers to the singleton building cell reconstructed during preprocessing.

We measured the overall geometric deviations between the input building models and the reconstructed singleton building cells by computing their Hausdorff distance, which is formulated as:

$$d_{\text{H}}(I, R) = \max\{\sup_{i\in I} d(i, R), \sup_{r\in R} d(r, I)\}, \tag{C4}$$

where $d(i, R)$ refers to the closest distance from i to R; $\sup_{i\in I} d(i, R)$ indicates the maximum $d(i, R)$ for all $i \in I$. However, self-intersections and open bottoms in the input models can lead to inaccurate Hausdorff distance estimates, as shown in Fig. C.2. Therefore, we use the median of $\{d(i, R)|i \in I\}$ and 75th percentile of $\{d(r, I)|r \in R\}$ to estimate the $\sup_{i\in I} d(i, R)$ and $\sup_{r\in R} d(r, I)$, respectively.

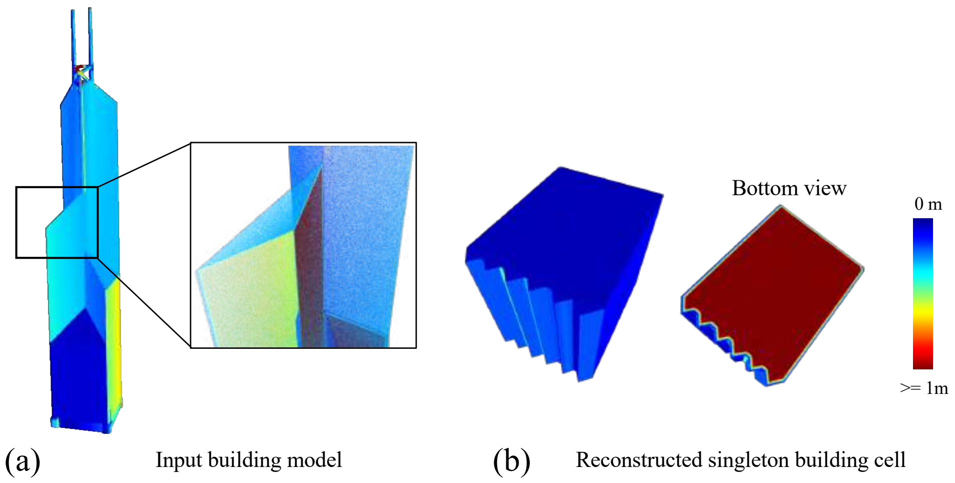


Figure C.2. Incorrect geometric deviation between the input building models and singleton building cells due to topological defects.

C.6. Details of comparisons with baseline methods

C.7. Details of parameter sensitivity tests

Table C1. Parameter analysis of ϵ in $f'(\mathbf{n})$ (Eqn. 6) on 19 samples[†]. (Best values are in bold.)

	$\epsilon = 2.0$	$\epsilon = 4.0$
Convexity ↑	0.95 ± 0.05	$\mathbf{0.96 \pm 0.04}$
Normalized number of parts ↓	$\mathbf{0.35 \pm 0.26}$	0.39 ± 0.30
Decomposition time (sec.) ↓	$\mathbf{12.28 \pm 15.67}$	12.51 ± 16.16

[†]5 samples excluded due to the processing time > 5 min.
↑: higher is better; ↓: lower is better.

We tested three different numbers of LoD L as:

(1) $\{10^{-2}, 10^{-4}\}$,
(2) $\{10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}\}$, and
(3) $\{5 \times 10^{-1}, 10^{-1}, 5 \times 10^{-2}, 10^{-2}, 5 \times 10^{-3}, 10^{-3}, 5 \times 10^{-4}, 10^{-4}\}$.

All settings terminate at 10^{-4}.

Table C2. Parameter analysis of number of LoD in Fast BBnB search on 19 samples[†]. (Best values are in bold.)

	$\|L\| = 2$	$\|L\| = 4$	$\|L\| = 8$
Convexity ↑	0.991 ± 0.02	0.991 ± 0.014	$\mathbf{0.992 \pm 0.014}$
Normalized number of parts ↓	$\mathbf{0.566 \pm 0.237}$	0.573 ± 0.256	0.575 ± 0.232
Decomposition time (sec.) ↓	15.73 ± 23.89	$\mathbf{14.33 \pm 21.03}$	14.42 ± 21.06

[†]5 samples excluded due to the processing time of $|L| = 2$ longer than 5 min.
↑: higher is better; ↓: lower is better.

C.8. Parameter settings for urban-scale validation

In the urban-scale tests, we used identical decomposition parameters ($k = 4.0$, $\epsilon = 2.0$, $L = 10^{-1}, 10^{-2}, 10^{-3}$) with two resolution schemes. For 3DBIT00, preprocessing

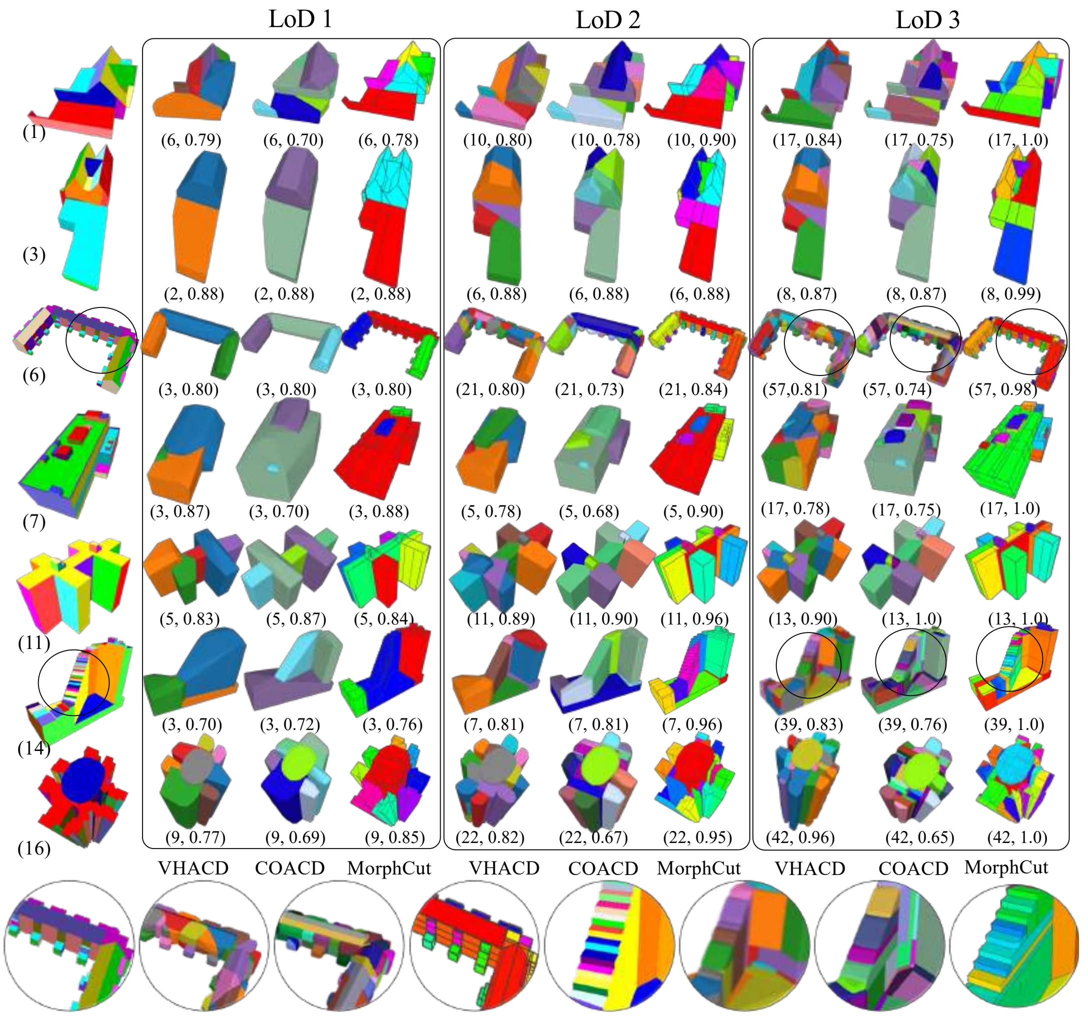


Figure C.3. Comparisons between VHACD, COACD, and MorphCut (Ours) across different LoDs. (•, •) refers to the number of decomposed parts and the average convexity.

Table C3. Parameter analysis of k in $h(\mathbf{n})$ on 24 samples. (Best values are in bold.)

	$k = 2$	$k = 4$	$k = 8$
Convexity ↑	0.990 ± 0.013	$\mathbf{0.991 \pm 0.012}$	0.989 ± 0.017
Normalized number of parts ↓	0.540 ± 0.25	$\mathbf{0.536 \pm 0.288}$	0.553 ± 0.248
Decomposition time (sec.) ↓	21.37 ± 60.21	$\mathbf{21.09 \pm 57.62}$	37.71 ± 136.73

↑: higher is better; ↓: lower is better.

employed a 4 m sampling interval and 1 m region-growing threshold, while City3D used the automatic tuning method from Section 4.3.4 with $f_{\mathrm{d}} = 0.01$. Further details on parameters and sensitivity analysis are provided in Sections 4.1.2 and 4.3.

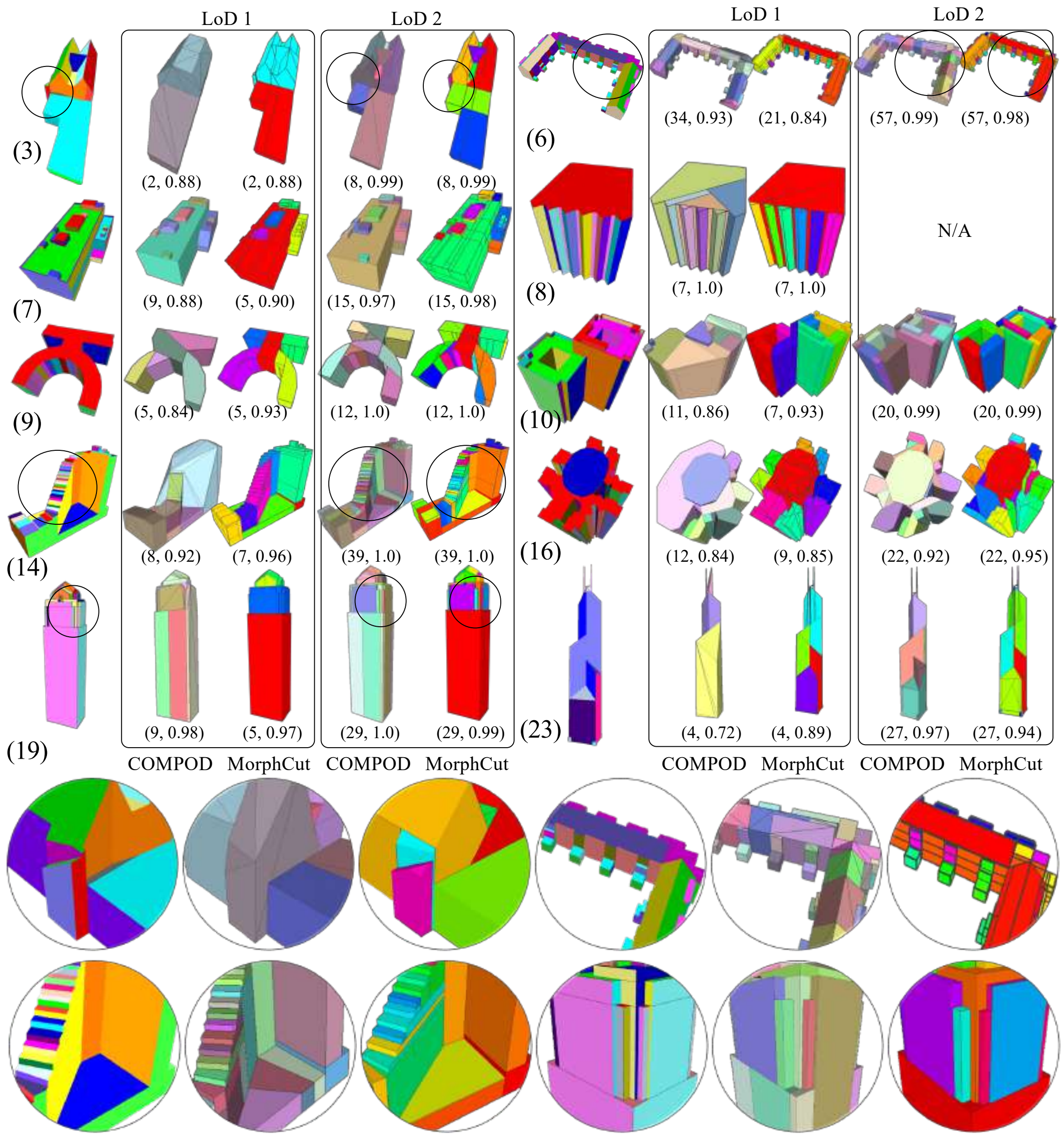


Figure C.4. Comparisons between COMPOD-PSDR and MorphCut (Ours). LoD 1 corresponds to the lowest LoD result by COMPOD, with results of similar LoD by MorphCut for comparisons. (●, ●) refers to the number of decomposed parts and the average convexity.

Table C4. Parameter analysis of resolution in preprocessing. (Best values are in bold.)

	$f_{\mathrm{d}} = 0.00625$†	$f_{\mathrm{d}} = 0.0125$	$f_{\mathrm{d}} = 0.025$	$f_{\mathrm{d}} = 0.05$	$f_{\mathrm{d}} = 0.1$‡
HD (m) ↓	**0.24 ± 0.15**	0.33 ± 0.39	0.45 ± 0.48	0.66 ± 0.63	2.57 ± 4.37
# Parts ↓	36.5 ± 23.2	30.1 ± 17.8	20.7 ± 11.2	11.7 ± 6.0	**5.3 ± 3.0**
Prep. time (sec.) ↓	34.7 ± 38.4	16.1 ± 13.8	10.0 ± 9.2	7.2 ± 7.2	**5.6 ± 6.0**
Decomp. time (sec.) ↓	29.1 ± 50.8	12.2 ± 13.3	5.6 ± 5.6	1.8 ± 1.3	**0.5 ± 0.4**

† Two samples excluded due to fine details causing decomposition time > 5 min.
‡ One sample excluded due to no interior cells labeled.
↑: higher is better; ↓: lower is better.

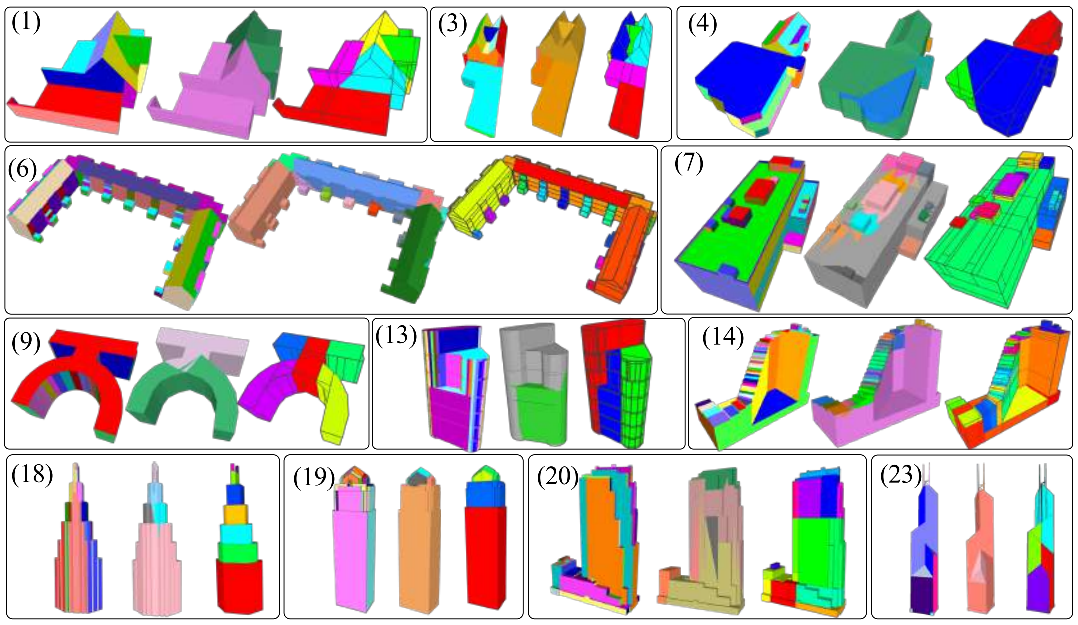


Figure C.5. Comparisons between SAMPart3D and MorphCut (Ours).

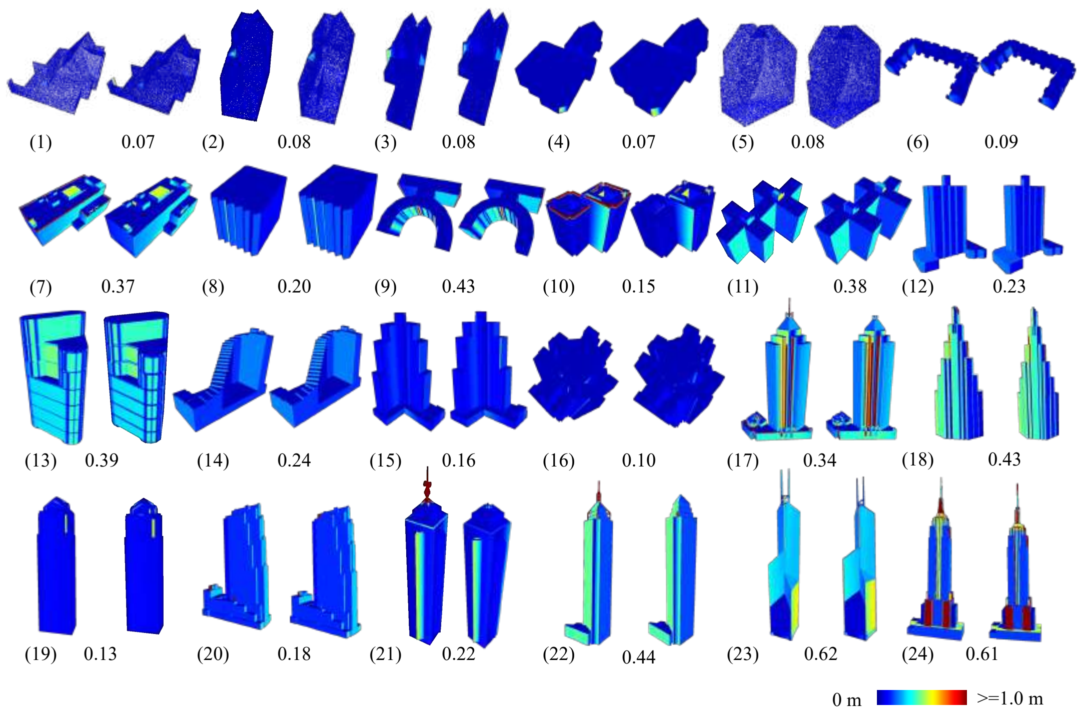


Figure C.6. Reconstruction error of singleton building cells from preprocessing. For each sample, the left and right columns colorize the input model and reconstructed cell in terms of the nearest point distance to each other, with the value beneath referring to the Hausdorff distance in meters. Significant geometric deviations (dark red regions) only occur at trivial details omitted during reconstruction, such as the spires in Samples (17), (21), (22), (23), and (24). Besides, the medium deviations, colorized in cyan, came from the plane fitting of region growing, e.g., Samples (7) and (13). The deviations could be greater for close parallel planes forming small setbacks or protrusions, such as the red planar regions in Samples (17) and (24).

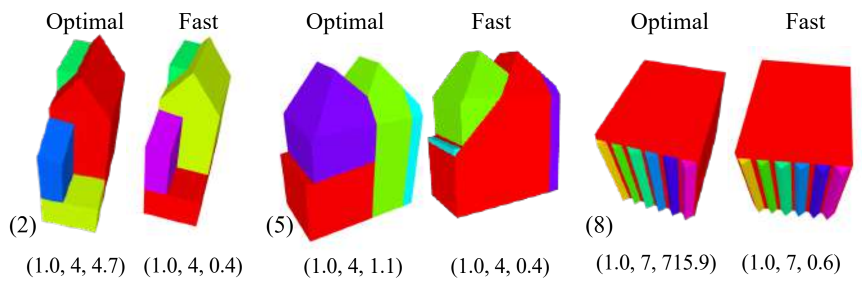


Figure C.7. Comparison between optimal and fast search on Samples (2), (5), (8). (•, •, •) indicate the convexity, number of parts, and decomposition time (sec.). Both approaches achieved the same level of convexity and number of parts. However, the decomposition time increased drastically with shape complexity. For example, the decomposition time for Sample (8) using the optimal search was 1,193 times longer than that of the fast search, highlighting the necessity and effectiveness of our fast BBnB design.

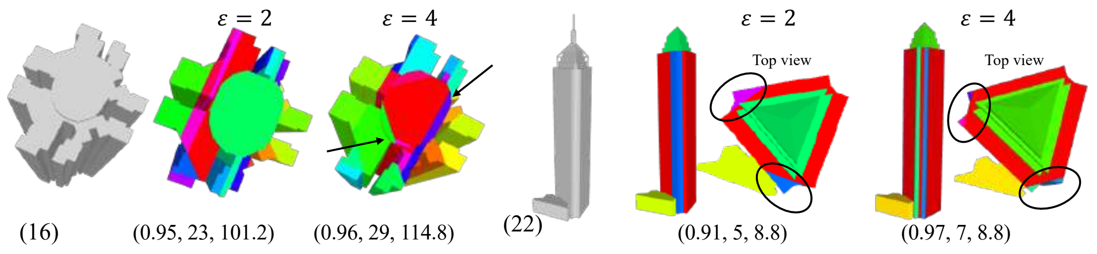


Figure C.8. Decomposition results for different ϵ values on Samples (16) and (22). (•, •, •) represents convexity, number of parts, and decomposition time (sec.), respectively.

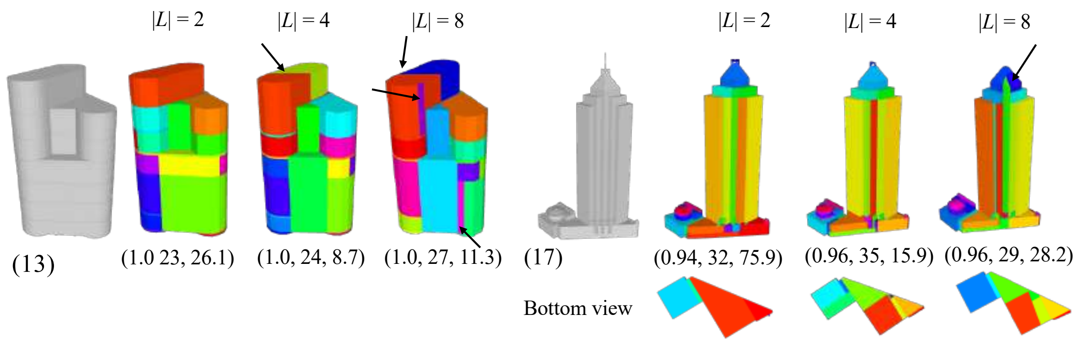


Figure C.9. Decomposition with different numbers of LoD on Samples (13) and (17). (•, •, •) represents the convexity, number of parts, and decomposition time (sec.), respectively.

Appendix D. Objectives for building decomposition of various patterns

To formulate the convex decomposition of 3D building models towards different patterns, we extend the objective function $f(\mathbf{n})$ defined in Eqn. 3 by introducing new terms for specific patterns. The objective function is then renamed as $f_o(\mathbf{n})$ where o indicates the given pattern. Similar to the BBnB search for $f(\mathbf{n})$, we combine $f_o(\mathbf{n})$ with the same heuristics $h(\mathbf{n})$ in Eqn. 5 into $f'_o(\mathbf{n})$ for adopting BBnB search. Meanwhile, to ensure the optimality in BBnB search, $f_o(\mathbf{n})$ should be monotonically non-decreasing. In the following, we show the formulation of $f_o(\mathbf{n})$ for main-volume, symmetrized, centralized, and decentralized convex decomposition.

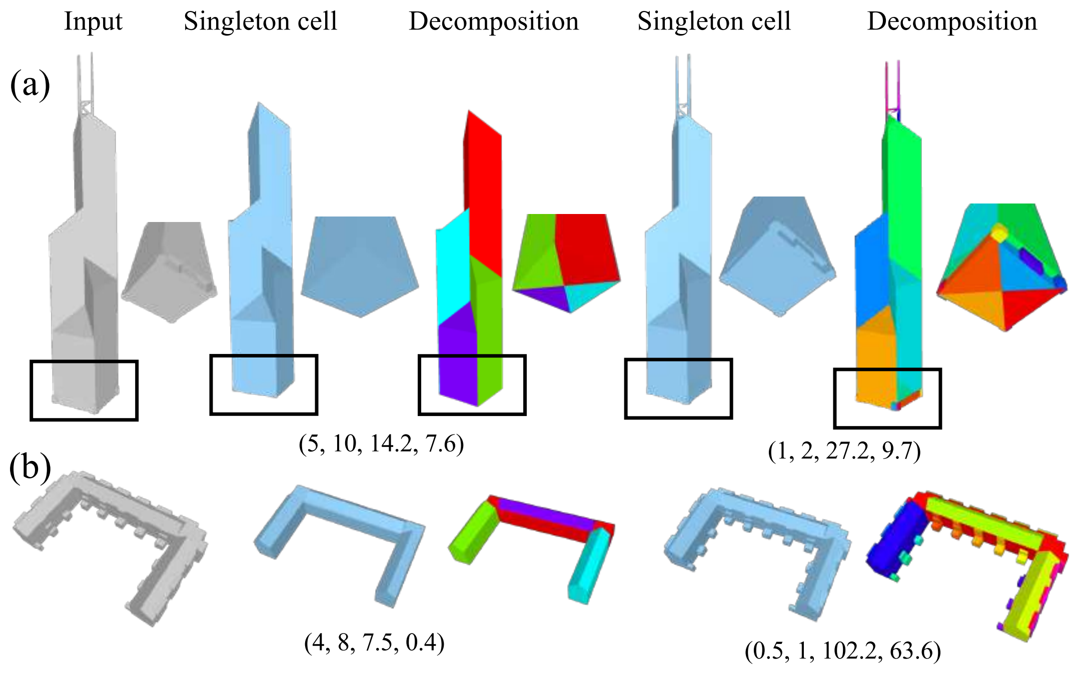


Figure C.10. Resolution for reconstructing singleton building cells in preprocessing. (●, ●, ●, ●) represents the distance threshold (m) of region growing, sampling interval (m), preprocessing time (sec.), and decomposition time (sec.).

D.1. Objective: main-volume

Main-volume decomposition simultaneously minimizes the number of parts and the difference ratio between the total volume and the maximal volume of the parts:

$$f_{\text{mv}}(\mathbf{n}) = f(\mathbf{n}) + \lambda_{\text{mv}}(1 - \text{MaxVolumeRatio}(\mathbf{n})) \tag{D1}$$

where MaxVolumeRatio($\mathbf{n}$) refers to the ratio of the maximal volume of the parts in the node $\mathbf{n}$ over their total volume; λ_{mv} weights between $f(\mathbf{n})$ and the maximal-volume term. Along the branching path of a node, since parts are split rather than merged, the maximal volume of any part in the node can only decrease. Therefore, the term $(1 - \text{MaxVolumeRatio}(\mathbf{n}))$ is monotonically non-decreasing and so is f_{mv}.

D.2. Objective: Symmetrized

The symmetrized decomposition maximizes the global symmetry in terms of the vertical symmetric plane of a building (Fig. D.1). As illustrated in Fig. D.1(b) and (c), a symmetrized decomposed part p should have a corresponding best-matching part under a symmetric transformation. Their intersection volume divided by the volume of p, termed intersection-over-self (IoS), should be close to 1. Therefore, we define the objective of symmetrized decomposition as:

$$f_{\text{sym}}(\mathbf{n}) = f(\mathbf{n}) + \lambda_{\text{sym}}(1 - \sum_{|\text{PotentialCuts}(p)|=0, \text{where} p \in \text{Parts}(\mathbf{n})} \text{VolumeRatio}(p, \mathbf{n}) \cdot \text{IoS}(p)) \tag{D2}$$

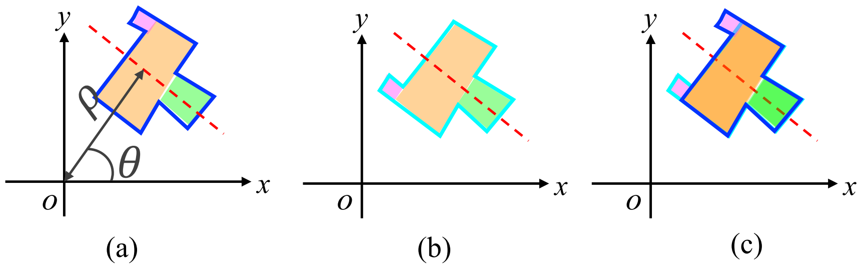


Figure D.1. Global symmetric plane (vertical) of a building. (a) presents the top view of a buildings' vertical global symmetric plane. (b) shows the building after a reflective symmetric transformation. (c) overlaps the building before and after the transformation.

where $|\text{PotentialCuts}(p)|$ refers to the number of the potential cutting planes of the part p; VolumeRatio(p, n) calculates the ratio of p's volume over the total volume of all parts of $\mathbf{n}$; and λ_{sym} weights the fewer-part and symmetric terms. To ensuring the monotonicity of f_{sym}, we only accumulate the symmetric metrics of the indecomposable parts p for which there is no potential cutting planes available. As the best symmetrically matched part may be decomposed into smaller ones and cannot grow bigger, the $\text{IoS}(p)$ is monotonically non-increasing and $f_{sym}(\mathbf{n})$ is monotonically non-decreasing.

D.3. Objective: Centralized

For centralized decomposition, we detect the decentralized parts of a node and accumulate their volume ratio. The decentralized parts are detected as parts with two adjacent parts. To guarantee the monotonicity, we only accumulate the volume ratio of parts that do not have potential cutting planes, as well as their two adjacent parts, i.e., indecomposable decentralized parts. As an adjacent part with potential cutting planes can be decomposed into more parts, the number of decentralized parts could decrease, which may decrease the f_{cen} if we count such decomposable parts in the decentralized parts. The objective towards centralized decomposition is thus formulated as:

$$f_{\text{cen}}(\mathbf{n}) = f(\mathbf{n}) + \lambda_{\text{cen}} \sum_{p \in \text{DecentralizedParts}(\mathbf{n})} \text{VolumeRatio}(p, \mathbf{n}) \tag{D3}$$

where $\text{DecentralizedParts}(\mathbf{n})$ counts the indecomposable decentralized parts only; λ_{cen} weights the fewer-part and centralized term.

D.4. Objective: Decentralized

Opposite to the centralized decomposition, the search counts the centralized parts with more than two adjacent parts. To ensure the monotonicity, we only count the indecomposable centralized parts but allow their adjacent parts to be decomposable.

We formulate the decentralized objective as:

$$f_{\text{dec}}(\mathbf{n}) = f(\mathbf{n}) + \lambda_{\text{dec}} \sum_{p \in \text{CentralizedParts}(n)} (|\text{AdjacentParts}(p)| - 2) \cdot \text{VolumeRatio}(p, n) \tag{D4}$$

where $|\text{AdjacentParts}(p)|$ refers to the number of adjacent parts of p; λ_{dec} weights the fewer-part and decentralized terms.

D.5. Visualization of decomposition with different objectives

Fig. D.2 displays the decomposed results towards different patterns. "FP" refers to fewer-part decomposition, the fundamental type defined in Section 3.3.1.2. "MV" decomposition emphasizes the main volume of a building, making the results of Samples (1) and (5) preserve their most significant convex parts from cutting. "SYM" mode decomposes buildings to enforce global symmetry (Wu, Xue, Li, & Chen, 2024; Xue, Lu, Webster, & Chen, 2019), resulting in distinct decompositions for Samples (5) and (10). Meanwhile, centralized ("CEN") and decentralized ("DEC") decompositions represent two contrasting patterns. Centralized decomposition often resembles "FP", "MV", or "SYM". In contrast, decentralized decomposition produces noticeably different results for all 6 samples in Fig. D.2. However, the relationship between building morphology and decomposition patterns, along with their implications for different aspects of urban analytics, remains an open question for future research.

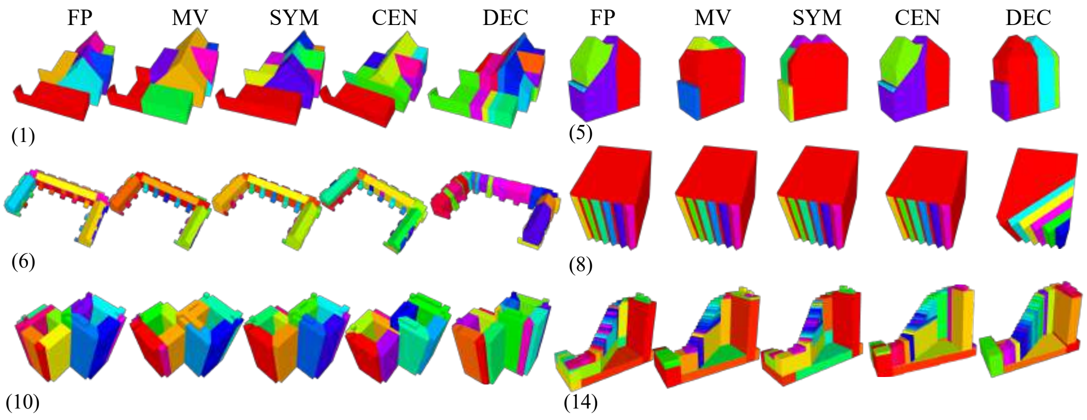


Figure D.2. Building decomposition with different patterns on 6 samples. FP, MV, SYM, CEN, and DEC refer to fewer-part, main-volume, symmetrized, centralized, and decentralized decomposition, respectively.

Appendix E. Supplementary visualization of method limitations

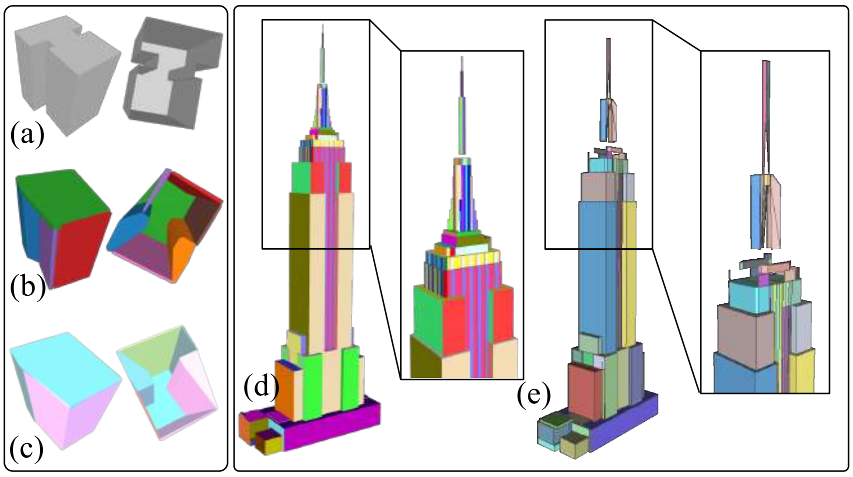


Figure E.1. Fidelity issues of VHACD, COACD, and COMPOD-PSDR. Left: Erroneous decomposition of open-bottom building models by VHACD and COACD. Right: Missed interior parts by COMPOD-PSDR. (a) and (d): Input building models. (b), (c), and (e): Results of VHACD, COACD, and COMPOD-PSDR, respectively.

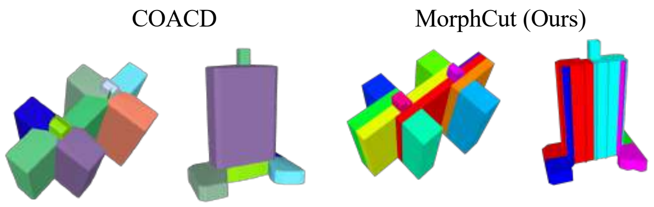


Figure E.2. Sub-optimal decomposition results of Samples (11) and (12) by MorphCut due to acceleration.

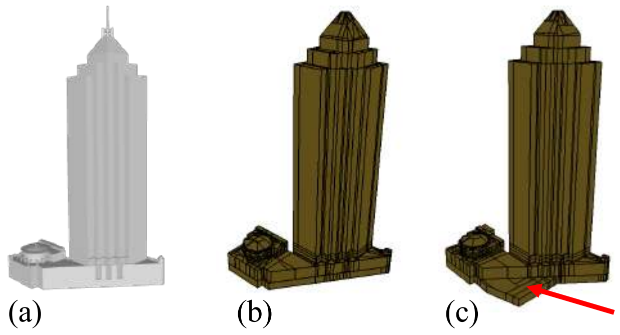


Figure E.3. Occasional errors in labeling the interior and exterior cells. (a) Input mesh. (b) A correct space labeling result. (c) Problematic labeling of cells, indicated by the red arrow.

References of Appendix

AHN3. (2018). *Actueel Hoogtebestand Nederland (AHN)*. `https://www.ahn.nl/`.

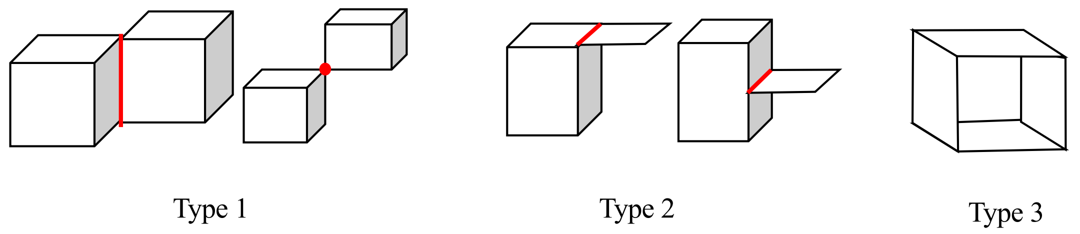


Figure E.4. Three typical types of non-manifold geometry. Type 1 can be readily addressed by extending MorphCut to allow multiple singleton building cells within the bounding box, though such cases are rare in real-world buildings. In contrast, Types 2 and 3 lack straightforward extensions for MorphCut and pose greater challenges. These structures are more common in practice, often serving as transitional spaces between building interior and exterior, such as overhanging roofs and open-sided shelters like bus stations.

Damiand, G. (2024). Combinatorial maps. In *CGAL User and Reference Manual.* CGAL Editorial Board. Retrieved from `https://doc.cgal.org/6.0.1/Manual/packages.html#PkgCombinatorialMaps`

DCP, N. (2018). *NYC 3D Model by Community District.* `https://www.nyc.gov/site/planning/data-maps/open-data/dwn-nyc-3d-model-download.page`.

Huang, J., Stoter, J., Peters, R., & Nan, L. (2022). City3D: Large-scale building reconstruction from airborne LiDAR point clouds. *Remote Sensing*, *14* (9), 2254. (doi:10.3390/rs14092254)

Huang, J., Zhou, Y., & Guibas, L. (2020). ManifoldPlus: A Robust and Scalable Watertight Manifold Surface Generation Method for Triangle Soups. *arXiv preprint arXiv:2005.11621.*

ISO. (2014). *C++17.* `https://en.cppreference.com/w/cpp/17`.

Jamin, C., Pion, S., & Teillaud, M. (2024). 3D triangulations. In *CGAL User and Reference Manual.* CGAL Editorial Board. Retrieved from `https://doc.cgal.org/6.0.1/Manual/packages.html#PkgTriangulation3`

LandsD, H. (2024). *3D-BIT00.* `https://www.landsd.gov.hk/en/survey-mapping/mapping/3d-mapping.html`.

Mamou, K., Lengyel, E., & Peters, A. (2016). Volumetric hierarchical approximate convex decomposition. In *Game Engine Gems.* CRC Press.

Oesau, S., Verdie, Y., Jamin, C., Alliez, P., Lafarge, F., Giraudot, S., ... Anisimov, D. (2024). Shape detection. In *CGAL User and Reference Manual.* CGAL Editorial Board. Retrieved from `https://doc.cgal.org/5.6.1/Manual/packages.html#PkgShapeDetection`

Rottensteiner, F., Sohn, G., Jung, J., Gerke, M., Baillard, C., Benitez, S., & Breitkopf, U. (2012). The ISPRS benchmark on urban object classification and 3D building reconstruction. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, *1* (1), 293–298. (doi:/10.5194/isprsannals-I-3-293-2012)

Sulzer, R., Yu, M., & Lafarge, F. (2023). *pyPSDR: Planar Shape Detection from Point Clouds.* GitHub Repository. Retrieved from `https://github.com/raphaelsulzer/psdr`

Varney, N., Asari, V. K., & Graehling, Q. (2020). DALES: A large-scale aerial LiDAR data set for semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)* (pp. 186–187). (doi:10.1109/CVPRW50498.2020.00101)

Wu, Y., Xue, F., Li, M., & Chen, S.-H. (2024). A novel building section skeleton for compact 3D reconstruction from point clouds: A study of high-density urban scenes. *ISPRS Journal of Photogrammetry and Remote Sensing*, *209*, 85–100. (doi:10.1016/j.isprsjprs.2024.01.020)

Xue, F., Lu, W., Webster, C. J., & Chen, K. (2019). A derivative-free optimization-based approach for detecting architectural symmetries from 3D point clouds. *ISPRS Journal of Photogrammetry and Remote Sensing*, *148*, 32–40. (doi: 10.1016/j.isprsjprs.2018.12.005)

Yu, A., & Shang, B. (2019). *Triangle mesh to signed-distance function (SDF).* `https://github.com/sxyu/sdf`. GitHub.

Yu, M., & Lafarge, F. (2022). Finding good configurations of planar primitives in unorganized point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 6367–6376). (doi:10.1109/CVPR52688.2022.00626)