Learning to Reconstruct Compact Building Models from Point Clouds

Author: Zhaiyu Chen

Supervisors: Dr. Liangliang Nan, Dr. Seyran Khademi

June 29, 2021
• Introduction
• Related work
• Methodology
• Datasets
• Results and discussion
• Conclusions
• Introduction
• Related work
• Methodology
• Datasets
• Results and discussion
• Conclusions
Introduction: 3D building models

Applications of 3D building models [Biljecki et al., 2015]
Introduction: Point clouds

Acquisition of a point cloud

- Photogrammetry
- LiDAR (Light Detection and Ranging)

1 https://www.tudelft.nl/bk/onderzoek/projecten/geoinformation-technology-governance
Compact building models needed
Introduction: Piecewise planarity

Piecewise-planar building models

- **Ubiquitous in the built environment**
- Capturing both geometry and topology with non-uniformity
- Compact, efficient with sparse sets of parameters

---

Introduction: Piecewise planarity

Piecewise-planar building models

- Ubiquitous in the built environment
- Capturing both geometry and topology with non-uniformity
- Compact, efficient with sparse sets of parameters

Dense triangles (smooth)
326,234 facets

Sparse triangles
198 facets

Sparse polygons
61 facets
Introduction: The *reconstruction* problem

The goal

- Compact
- Watertight
- Accurate
- Efficient
Introduction: Challenges

- Compactness, watertightness, efficiency
- Limited input data quality
Introduction: Inspiration and research question

Point cloud  Surface mesh  Implicit function/field  Deep neural network

Shape representations [Mescheder et al., 2019]

How can deep implicit fields be used for compact building model reconstruction?
• Introduction
• Related work
• Methodology
• Datasets
• Results and discussion
• Conclusions
Related work: Shape reconstruction (smooth)

- Poisson reconstruction [Kazhdan et al., 2006]
- Points2Surf [Erler et al., 2020]

Massive triangles
Related work: Shape reconstruction (piecewise-planar)

- PolyFit [Nan and Wonka, 2017]

Scalability issue with its integer programming solver
Related work: Geometry simplification

- **Manhattan-world reconstruction** [Li et al., 2016b]
- **2.5D Dual Contouring** [Zhou and Neumann, 2010]

Not generic with only boxes
Related work: Geometry simplification

- Manhattan-world reconstruction [Li et al., 2016b]
- 2.5D Dual Contouring [Zhou and Neumann, 2010]

Not generic with only 2.5D
Related work: Surface approximation

- Quadric error metrics (QEM) [Garland and Heckbert, 1997]
- Variational shape approximation (VSA) [Cohen-Steiner et al., 2004]
- Structure-aware mesh decimation (SAMD) [Salinas et al., 2015]

Dependent on input mesh [Bouzas et al., 2020]
### Related work: Summary

<table>
<thead>
<tr>
<th>Related work</th>
<th>Characteristics</th>
<th>Category</th>
<th>Compact</th>
<th>Watertight</th>
<th>Generic</th>
<th>Efficient</th>
</tr>
</thead>
<tbody>
<tr>
<td>Poisson [Kazhdan et al., 2006]</td>
<td></td>
<td>RC</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Points2Surf [Erler et al., 2020]</td>
<td></td>
<td>RC</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>PolyFit [Nan and Wonka, 2017]</td>
<td></td>
<td>RC</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>SAMD [Salinas et al., 2015]</td>
<td></td>
<td>AP</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>VSA [Cohen-Steiner et al., 2004]</td>
<td></td>
<td>AP</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Manhattan-world [Li et al., 2016b]</td>
<td></td>
<td>SP</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>2.5D DC [Zhou and Neumann, 2010]</td>
<td></td>
<td>SP</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Ours</td>
<td></td>
<td>RC</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

Characteristics overview of related work

---

3 Only methods in comparison through experiments (with official open-source code); See in the thesis a complete literature review.

---

TU Delft
• Introduction
• Related work
• Methodology
• Datasets
• Results and discussion
• Conclusions
Methodology: Overview

Explicit

- Point cloud\textsuperscript{(a)}
  - RANSAC
  - Planar primitives\textsuperscript{(b)}
    - Adaptive binary space partitioning
    - Candidate cells\textsuperscript{(c)}

Implicit

- Query points\textsuperscript{(c)}
  - Neural implicit field\textsuperscript{(d)}
  - Markov random field
    - Graph-cut solver
      - Surface\textsuperscript{(f)}

Surface extraction

Overview of our framework
Methodology: Overview

Overview of our framework
Methodology: Adaptive binary space partitioning

RANSAC

Planar primitive detection
Methodology: Adaptive binary space partitioning

Planar primitive refinement
Methodology: Overview

Explicit

Point cloud\(^{(a)}\)

RANSAC

Planar primitives\(^{(b)}\)

Adaptive binary space partitioning

Candidate cells\(^{(c)}\)

Implicit

Query points\(^{(e)}\)

Neural implicit field\(^{(d)}\)

Markov random field

Graph-cut solver

Surface\(^{(f)}\)

Surface extraction

Overview of our framework
Methodology: Adaptive binary space partitioning

BSP-tree

A

A
Methodology: Adaptive binary space partitioning
Methodology: Adaptive binary space partitioning
Methodology: Adaptive binary space partitioning
Methodology: Adaptive binary space partitioning
Methodology: Adaptive binary space partitioning
Methodology: Overview

Explicit

- Point cloud \(^{(a)}\)
  - RANSAC
  - Planar primitives \(^{(b)}\)
    - Adaptive binary space partitioning
    - Candidate cells \(^{(c)}\)

Implicit

- Query points \(^{(e)}\)
  - Neural implicit field \(^{(d)}\)
    - Markov random field
    - Graph-cut solver
    - Surface \(^{(f)}\)

Surface extraction

Overview of our framework
Methodology: Occupancy learning in function space

Signed distance function

\[
SDF(x) = s : x \in \mathbb{R}^3, s \in \mathbb{R}.
\]

Surface at \( SDF(\cdot) = 0 \)
Methodology: Occupancy learning in function space

Signed distance function

\[ SDF(x) = s : x \in \mathbb{R}^3, s \in \mathbb{R}. \]

Surface at \( SDF(\cdot) = 0 \)

\[ SDF(x) \approx \tilde{f}_P(x) = s_\theta(x \mid z), \text{ with } z = e_\phi(P) \]

- \( P \): point cloud
- \( \theta \): NN parameters
- \( e \): encoder

Query points

Deep implicit field

Predicted distances
Methodology: Occupancy learning in function space

Points2Surf neural network architecture [Erler et al., 2020]

\[ SDF(x) \approx f_P(x) = s_\theta(x \mid z), \text{ with } z = e_\phi(P) \]

- \( f_P^d(x) = s_\theta^d(x \mid z_x^d), \text{ with } z_x^d = e_\phi^d(p_x^d) \) **Absolute distance**

- \( f_P^s(x) = \text{sgn}(\tilde{g}_P^s(x)) = \text{sgn}(s_\theta^s(x \mid z_x^s)), \text{ with } z_x^s = e_\psi^s(p_x^s) \) **Sign**
Methodology: Occupancy learning in function space

Training with loss function

\[ \sum_{(P, S) \in \mathcal{S}} \sum_{x \in \mathcal{X}_S} \mathcal{L}^d(x, P, S) + \mathcal{L}^s(x, P, S) \]

- \( \mathcal{L}^d(x, P, S) = \| \tanh \left( |\tilde{f}_P(x)| \right) - \tanh(|d(x, S)|) \|_2^2 \) \quad Error of distance prediction
- \( \mathcal{L}^s(x, P, S) = H(\sigma(\tilde{g}_P(x)), [f_S(x) > 0]) \) \quad Error of sign prediction

Sanity check: overfitting one shape
Methodology: Occupancy learning in function space

Signed distance voting

\[ \bar{SD}^P = \frac{1}{|P|} \sum_{i \in P} SD_i^P \]

Point cloud
Methodology: Occupancy learning in function space

Signed distance voting

\[ \overline{SD}^P = \frac{1}{|P|} \sum_{i \in P} SD_i^P \]

Candidate cells

SDF

Signed distance for each candidate
Methodology: Overview

Overview of our framework
Methodology: Surface extraction

Energy formulation (Markov random field)

\[ E(x) = D(x) + \lambda V(x) \]

\[ x_i = \{ \text{in}, \text{out} \} \]
Methodology: Surface extraction

Fidelity term (unary potential)

\[ D(X) = \frac{1}{|C|} \sum_{i \in C} d_i(C_i, x_i) \]

- \( d_i(C_i, x_i) = |\text{probability}(C_i) - x_i| \)
- \( \text{probability}(C_i) = \text{sigmoid}(SD_i \cdot \text{volume}_i) \)

Signed distance for each candidate
Methodology: Surface extraction

Complexity term (pairwise potential)

\[ V(X) = \frac{1}{A} \sum_{\{i,j\} \in C} a_{ij} \cdot 1_{x_i \neq x_j} \]

- \( \{i, j\} \in C \) represents pairs of adjacent polyhedra
- \( a_{ij} \) denotes the shared area

Less zigzagging
Methodology: Surface extraction

Graph-cut solver for the Markov random field

\[ E(x) = D(x) + \lambda V(x) \]

\[ x_i = \{\text{in, out}\} \]

Graph cuts [Boykov and Funka-Lea, 2006]
Methodology: Surface extraction

Graph-cut solver for the Markov random field

\[ E(x) = D(x) + \lambda V(x) \]

\[ x_i = \{\text{in, out}\} \]
• Introduction
• Related work
• Methodology
• Datasets
• Results and discussion
• Conclusions
Datasets: Helsinki

Simulated LiDAR scanning from CityGML models

- Point clouds
- Surface -> Sampled query points with signed distance values
Datasets: Helsinki full-view

Vertical Orbit

LiDAR Sensor

Building Mesh

Horizontal Orbit
Datasets: Helsinki full-view

Gaussian Noise
Datasets: Helsinki no-bottom

Vertical Orbit

LiDAR Sensor

Horizontal Orbit

Building Mesh
Datasets: Helsinki

Helsinki full-view

Helsinki no-bottom
Datasets: Shenzhen

Data courtesy of Linfu Xie [Xie et al., 2021]
## Datasets

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Perspective</th>
<th>Quantity</th>
<th>Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Helsinki full-view</td>
<td>Simulated LiDAR</td>
<td>✔️ ✔️ ✔️</td>
<td>768</td>
<td>Training + evaluation</td>
</tr>
<tr>
<td>Helsinki no-bottom</td>
<td>Simulated LiDAR</td>
<td>✔️ ❌ ✔️</td>
<td>768</td>
<td>Training + evaluation</td>
</tr>
<tr>
<td>Shenzhen</td>
<td>Real-world MVS</td>
<td>✔️ ❌ ✔️</td>
<td>6</td>
<td>Evaluation</td>
</tr>
</tbody>
</table>
• Introduction
• Related work
• Methodology
• Datasets
• Results and discussion
• Conclusions
Results & discussion: *Helsinki full-view*

- Point cloud
- Candidate polyhedra
- SDF
- Reconstructed

Signed Distance

TU Delft
Results & discussion: Helsinki full-view

Point cloud  Candidate polyhedra  SDF  Reconstructed
Results & discussion: Helsinki full-view

<table>
<thead>
<tr>
<th>Reference</th>
<th>Reconstructed</th>
<th>Error</th>
<th>Dist_{SMH}(%)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>0.04541</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0.04440</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0.11673</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0.22028</td>
</tr>
</tbody>
</table>
Results & discussion: Helsinki full-view
Results & discussion: Helsinki no-bottom
Results & discussion: Shenzhen

Point cloud
Candidate polyhedra
SDF
Reconstructed
Results & discussion: *Shenzhen*

<table>
<thead>
<tr>
<th>Reference</th>
<th>Reconstructed</th>
<th>Error</th>
<th>$Dist_{SR}$(%)</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1" alt="Reference Image 1" /></td>
<td><img src="image2" alt="Reconstructed Image 1" /></td>
<td><img src="image3" alt="Error Image 1" /></td>
<td>3.8815</td>
</tr>
<tr>
<td><img src="image4" alt="Reference Image 2" /></td>
<td><img src="image5" alt="Reconstructed Image 2" /></td>
<td><img src="image6" alt="Error Image 2" /></td>
<td>2.0045</td>
</tr>
<tr>
<td><img src="image7" alt="Reference Image 3" /></td>
<td><img src="image8" alt="Reconstructed Image 3" /></td>
<td><img src="image9" alt="Error Image 3" /></td>
<td>1.1059</td>
</tr>
<tr>
<td><img src="image10" alt="Reference Image 4" /></td>
<td><img src="image11" alt="Reconstructed Image 4" /></td>
<td><img src="image12" alt="Error Image 4" /></td>
<td>1.5759</td>
</tr>
<tr>
<td><img src="image13" alt="Reference Image 5" /></td>
<td><img src="image14" alt="Reconstructed Image 5" /></td>
<td><img src="image15" alt="Error Image 5" /></td>
<td>2.5830</td>
</tr>
<tr>
<td><img src="image16" alt="Reference Image 6" /></td>
<td><img src="image17" alt="Reconstructed Image 6" /></td>
<td><img src="image18" alt="Error Image 6" /></td>
<td>3.3763</td>
</tr>
</tbody>
</table>

Legend: Low - Red, High - Blue
Results & discussion: Shenzhen

Reconstruction from insufficient scans
Results & discussion: Comparison with smooth reconstruction

Point cloud

Poisson [Kazhdan et al., 2006]

Points2Surf [Erl er et al., 2020]

Ours

143,716 facets

34,050 facets

106 facets

211,572 facets

18,326 facets

123 facets
Results & discussion: Comparison with piecewise-planar reconstruction

Helsinki full-view

Shenzhen

Point cloud

PolyFit

Ours

[3D models of different locations and techniques, comparison with cubic geometries]

Manhattan-world

[Li et al., 2016b]

2.5D DC

[Zhou and Neumann, 2010]
Results & discussion: Comparison

643 facets

117 facets

PolyFit

[\text{Nan and Wonka, 2017}]

Ours
Results & discussion: Comparison

Point cloud

PolyFit

[Nan and Wonka, 2017]

Ours
Results & discussion: Comparison with surface approximation methods

Point cloud

QEM [Garland and Heckbert, 1997]

SAMD [Salinas et al., 2015]

VSA [Cohen-Steiner et al., 2004]

Ours
Results & discussion: Efficiency

![Bar chart showing running time for different building indices.

- Building index #1 to #8
- Y-axis: Running time (seconds)
- X-axis: Building index
- Legend: Candidates generation, Occupancy inference, Surface extraction

The chart illustrates the running time for each building index, with #6 having the highest running time.
Results & discussion: Efficiency

[Graphs showing comparison between Adaptive and Exhaustive methods for time partitioning and number of polyhedra vs. number of primitives.]
Results & discussion: Scalability
Results & discussion: Robustness to noise

0R

0.001R

0.005R

0.010R

0.050R

1 meter against 100 meter-sided building
Results & discussion: Impact of parameter $\lambda$

$$E(x) = D(x) + \lambda V(x)$$

$Dist_{SMH} = 0.04268$
#Facets = 184

$Dist_{SMH} = 0.04324$
#Facets = 170

$Dist_{SMH} = 0.07495$
#Facets = 236

$Dist_{SMH} = 0.07522$
#Facets = 220

$\lambda = 0.002$

$\lambda = 0.01$
Results & discussion: limitations

Accurate primitives

High-quality model

Inaccurate primitives

Low-quality model
Results & discussion: limitations

‘Caved’ artefact
Results & discussion: Applications

Building component analysis

BSP-tree

A

\[ \cap H_{1L} \]

\[ \cap H_{1R} \]

B

C

\[ \cap H_{2L} \]

\[ \cap H_{2R} \]

D

E
Results & discussion: Applications

- Compression
- Physical Simulation

Generic shape reconstruction
• Introduction
• Related work
• Methodology
• Results and discussion
• Conclusions
Conclusion: Research question revisited

How can deep implicit fields be used for compact building model reconstruction?

- Compactness and watertightness
- Generalisation
- Robustness
- Advantages & disadvantages
## Conclusion

### Contributions

- A learning-based framework to incorporate deep implicit fields into piecewise-planar urban building reconstruction
- An adaptive space partitioning strategy for cell complex construction
- An MRF formulation for efficient surface extraction
- Open synthetic building point cloud dataset
**Conclusion**

**Future work**

- End-to-end neural network architecture
- Extension to more general primitives

...
Conclusion

Source code

- https://github.com/chenzhaiyu/absp
- https://github.com/chenzhaiyu/points2poly

Dissemination

- Thesis & Slides available at TU Delft Repositories
- ISPRS Journal manuscript in progress
Thanks! Questions?