Intuitive and Efficient Roof Modeling for Reconstruction and Synthesis

Jing Ren\textsuperscript{1,2}, Biao Zhang\textsuperscript{1}, Bojian Wu\textsuperscript{2}, Jianqiang Huang\textsuperscript{2}, Lubin Fan\textsuperscript{2}, Maks Ovsjanikov\textsuperscript{3}, Peter Wonka\textsuperscript{1}

\textsuperscript{1}KAUST, \textsuperscript{2}Alibaba Group, \textsuperscript{3}Ecole Polytechnique, IP Paris
Introduction

Problem Formulation

Input

Output
Existing Methods
Commercial Software: 3ds Max

Modeling time 3ds Max: 7min, Ours: 2min

Modeling Operations
• Add vertex
• Add face
• Cut faces
• Move vertex
Existing Methods
Commercial Software: SketchUp

Modeling time SketchUp: 25min, Ours: 2min

Modeling Operations
1. Build roof beams
2. Add planar roof tops

Extra complexity!
Existing Methods

Straight Skeleton Based Methods

**Input** 2D roof outline  \hspace{2cm} **Output** a roof embedded in 2D

**Method** solve for the roof topology & roof embedding at the same time

- Blue planes: stemming from the outlined edges
- Red structure: formed by intersection among blue planes

Reference:
1. Aichholzer and Aurenhammer, “Straight Skeletons for general polygonal figures in the plane”, 1996
2. Eppstein and Erickson, “Raising roofs, crashing cycles, and playing pool: Applications of a data structure for finding pairwise interactions”, 1999
Existing Methods

Straight Skeleton Based Methods

Problems wrong (and much more complicated) roof topology with spurious additional vertices
Expected Problems cannot handle the case where a face contains multiple outline edges.
## Existing Methods

| Feature                                      | Commercial software | Straight skeleton | Our goal |
|----------------------------------------------|---------------------|-------------------|----------|
| Topologically correct                       | 🙃                   | ☹                 | ☹        |
| Interactive editing                         | 🙃                   | ☹                 | ☹        |
| Robust to noisy input                       | 🙃                   | ☹                 | ☹        |
| Light user input                            | ☹️                  | ☹️                | ☹️       |
| Efficient for roof construction             | ☹️                  | ☹️                | ☹️       |
| Easy to use for beginners                   | ☹️                  | ☹️                | ☹️       |
Methodology

Overview

Q1 How to \textbf{model} a roof?
Q2 How to describe whether a 3D roof is \textbf{planar}?
Q3 How to design \textbf{different regularizers} for different use cases?
Roof Topology

Graph \((V, F)\)

\(|V| = n\)

Roof Embedding

2D embedding:

\((V, F, X \in \mathbb{R}^{n \times 2})\)

3D embedding:

\((V, F, X \in \mathbb{R}^{n \times 3})\)
**DEFINITION 1** We call a 3D embedding of a roof graph **valid** if each 3D roof face is **planar** and the roof has **non-zero height**.

**DEFINITION 2** We call a 2D embedding of a roof graph **valid** if there exists a **valid 3D embedding** such that the projection of the 3D embedding in the \(xy\) plane is exactly the same as the 2D embedding.
Methodology

Notations

- **Roof Edge**
- **Outline Edge**

- **Roof Vertex**
- **Outline Vertex**
Methodology

Notations

Roof Graph

Dual Graph

Complete Dual Graph

Equivalent topology representation primal graph $\leftrightarrow$ dual graph
**Observation** roof outline $\rightarrow$ roof topology is NOT injective

- Roofs with **different style** (topology) can have the **same outline**

**Solution** outline + adjacency to specify the roof topology
Observation multiple valid roof embeddings exist
• Roofs with the same topology can have different embeddings
Solution regularizers to rule out undesirable embeddings
Methodology

Co-planarity Metric

Covariance

\[ \text{Cov}(X) = (X - \text{mean}(X))^T (X - \text{mean}(X)) \]

- Symmetric & Positive semi-definite
- Smallest eigenvalue \( \geq 0 \)

Summary \( \sigma_1(\text{Cov}(X)) \) measures the co-planarity error of a point set \( X \)
Methodology

Validity Formulation

Input  roof outline + topology (primal/dual graph)
Output a valid 3D roof embedding
  • valid: each 3D roof face is planar
Solution optimization-based formulation
  • variables: the 3D positions for each vertex
  • objective: the vertices in each face are co-planar

\[
\text{E}_{\text{planarity}}(X) = \sum_{i=1}^{n_f} \sigma_1 \left( \text{Cov}(X_{f_i}) \right)
\]

❖ Cov(X): covariance matrix of X
❖ \(\sigma_1(A)\): the smallest eigenvalue of A
❖ \(X_{f_i}\): vertex positions of the \(f_i\)-th face
Methodology
Planarity Optimization

\[
\min_{X_R} E_{\text{planarity}}(X)
\]

\[x^*_z = h\]

**Variables** 3D positions for roof vertices \(X_R\)
- Fixed outline: same setting as straight skeleton based methods

**Hard constraint** avoid zero-height roof
- Hyper-parameter \(h\) allows us to control the overall height of the constructed roof

\(\sigma_1(A)\): the smallest eigenvalue of \(A\)
\(X_{f_i}\): vertex positions of the \(f_i\)-th face
\(E_{\text{planarity}}(X) = \sum_{i=1}^{n_f} \sigma_1(\text{Cov}(X_{f_i}))\)
Planarity & User Input

Methodology

\[
\min_{X_R} E_{\text{planarity}}(X) + \|\bar{X}_R - \bar{X}_{user}\|_F^2
\]

\[x^*_z = h\]

- \(\text{Cov}(X)\): covariance matrix of \(X\)
- \(\sigma_1(A)\): the smallest eigenvalue of \(A\)
- \(X_{f_i}\): vertex positions of the \(f_i\)-th face
- \(E_{\text{planarity}}(X) = \sum_{i=1}^{n_f} \sigma_1(\text{Cov}(X_{f_i}))\)
- \(\bar{X}\): the 2D projection of \(X\)

User input not valid but provides good initial locations

- enforce 2D projections of \(X\) to be close to user input
Methodology

Planarity & Aesthetic Constraints

- $\text{Cov}(X)$: covariance matrix of $X$
- $\sigma_1(A)$: the smallest eigenvalue of $A$
- $X_{f_i}$: vertex positions of the $f_i$-th face
- $\text{Eplanarity}(X) = \Sigma_{i=1}^{n_f} \sigma_1 \left( \text{Cov}(X_{f_i}) \right)$
- $d(e_p, e_q)$: distance between two parallel vectors

**Parallel** $e_p \parallel e_{p_1} \parallel e_{p_2}$

$$E = \left\| d(e_q, e_{q_1}) - d(e_q, e_{q_2}) \right\|_F^2$$
Methodology

Planarity & Aesthetic Constraints

- $\text{Cov}(X)$: covariance matrix of $X$
- $\sigma_1(A)$: the smallest eigenvalue of $A$
- $X_{f_i}$: vertex positions of the $f_i$-th face
- $E_{\text{planarity}}(X) = \sum_{i=1}^{n_f} \sigma_1 \left( \text{Cov}(X_{f_i}) \right)$
- $a(e_p, e_q)$: angle between two parallel vectors

**Intersecting** $e_q \land e_{q_1} \land e_{q_2}$

$$E = \| a(e_q, e_{q_1}) - a(e_q, e_{q_2}) \|^2_F$$
Methodology

Planarity & Aesthetic Constraints

\[
\min_{X_R} E_{\text{planarity}}(X) + E_{\text{aesthetic}}(X)
\]

\[x^*_z = h\]

\[
E_{\text{aest.}}(X) = \sum_{p \in \{I\}} \|a(e_p, e_{p_1}) - a(e_p, e_{p_2})\|_F^2 + \sum_{q \in \{I\}} \|d(e_q, e_{q_1}) - d(e_q, e_{q_2})\|_F^2
\]

- Cov\( (X) \): covariance matrix of \( X \)
- \( \sigma_1(A) \): the smallest eigenvalue of \( A \)
- \( X_{fi} \): vertex positions of the \( f_i \)-th face
- \( E_{\text{planarity}}(X) = \sum_{i=1}^{n_f} \sigma_1 \left( \text{Cov}(X_{fi}) \right) \)

**Aesthetic constraints** more plausible local minima
- Yellow edges: angle bisectors
- Green edges: equal distance to the parallel outline edges

\( a(e_p, e_q) \): angle between two vectors
\( d(e_p, e_q) \): distance between two parallel vectors
Methodology
Planarity & Free Outlines

\[
\min_{x_{xyz}, x_z} E_{\text{planarity}}(X) + \eta \text{Var}(x_z) + \eta \text{Var}(x_z) \\
\]

\[x_z^* = h\]

Set outline vertices as free variables

- Add extra constraints: subset of outline vertices in similar height

- \(\text{Cov}(X)\): covariance matrix of \(X\)
- \(\sigma_1(A)\): the smallest eigenvalue of \(A\)
- \(X_{f_i}\): vertex positions of the \(f_i\)-th face
- \(E_{\text{planarity}}(X) = \sum_{i=1}^{nf} \sigma_1(\text{Cov}(X_{f_i}))\)
- \(\text{Var}(x)\): variance of a vector \(x\)

Hexagonal Pavilion

- Set outline vertices as free variables
- Add extra constraints: subset of outline vertices in similar height

- \(\text{Cov}(X)\): covariance matrix of \(X\)
- \(\sigma_1(A)\): the smallest eigenvalue of \(A\)
- \(X_{f_i}\): vertex positions of the \(f_i\)-th face
- \(E_{\text{planarity}}(X) = \sum_{i=1}^{nf} \sigma_1(\text{Cov}(X_{f_i}))\)
- \(\text{Var}(x)\): variance of a vector \(x\)
Set outline vertices as free variables

- red: outline vertices with non-zero height
Results

Hakka Tulou, China

Nagoya Castle, Japan

Feasibility model roofs with different styles:
- Approximate curved roofs
- Roofs with vertical facades
- Roofs with inner courtyards
- Outline edges in different height

(Reference images from internet)
Results

Compare to Straight Skeleton Based Methods

Each Row

1. Input aerial images
2. Our reconstructed roofs (texture)
3. Our reconstructed roofs (geometry)
4. Straight Skeleton
5. Weighted Straight Skeleton (wss)

Inconsistencies
Error introduced by wss
Results

Compare to Straight Skeleton Based Methods

| No. | 1  | 2  | 3  | 4  | 5  | 6  | 7  | 8  | 9  | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
|-----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| \(n_v\) | 22 | 24 | 25 | 25 | 27 | 27 | 32 | 33 | 33 | 33 | 34 | 35 | 36 | 39 | 39 | 51 |
| \(n_f\) | 12 | 17 | 14 | 14 | 15 | 16 | 17 | 20 | 18 | 20 | 18 | 21 | 19 | 22 | 21 | 38 |

| #err | \(\text{Ours}\) | ss | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0  |
|------|----------------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| \(\text{wss}\) | 0  | 2  | 0  | 0  | 1  | 2  | 1  | 2  | 0  | 3  | 2  | 2  | 1  | 3  | 2  | 4  |

| \(\vec{n}_v\) | \(\text{Ours}\) | ss | 22 | 24 | 25 | 25 | 27 | 27 | 32 | 33 | 33 | 33 | 34 | 35 | 36 | 39 | 39 | 51 |
|---------------|----------------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| \(\text{wss}\) | 22 | 22 | 26 | 26 | 30 | 28 | 34 | 40 | 34 | 38 | 38 | 40 | 38 | 40 | 44 | 50 |    |

| \(\vec{n}_f\) | \(\text{Ours}\) | ss | 12 | 17 | 14 | 14 | 15 | 16 | 17 | 20 | 18 | 20 | 18 | 21 | 19 | 22 | 21 | 38 |
|---------------|----------------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| \(\text{wss}\) | 12 | 12 | 14 | 14 | 16 | 15 | 18 | 21 | 18 | 20 | 20 | 21 | 20 | 21 | 23 | 26 |    |

| \(t\) (s) | \(\text{Ours}\) | ss | 89.4 | 115 | 153 | 97.9 | 114 | 92.9 | 151 | 145 | 155 | 167 | 148 | 158 | 179 | 199 | 178 | 284 |
|------------|----------------|----|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| \(\text{wss}\) | -   | 300 | -    | -    | 60   | 180  | 180  | 300  | -    | 120  | 180  | 60   | 480  | 180  | 360  |    |    |

More expressive!

Baselines
- Straight Skeleton (ss)
- Weighted Straight Skeleton (wss)
## Results

### Compare to Commercial Software

| No. | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
|-----|---|---|---|---|---|---|---|---|---|----|----|----|----|----|----|----|
| **Ours** | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| **Valid** | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| **3ds Max** | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| **SketchUp (SU)** | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| **#err** | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| **3D** | 9 | 12 | 10 | 11 | 12 | 16 | 15 | 11 | 16 | 15 | 16 | 18 | 17 | 20 |
| **SU** | 20 | 16 | 21 | 16 | 27 | 33 | 21 | 25 | 38 | 28 | 30 | 33 | 33 | 29 | 42 |
| **poly (%)** | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| **3D** | 75 | 65 | 71 | 79 | 67 | 69 | 82 | 75 | 83 | 60 | 72 | 57 | 79 | 73 | 71 | 42 |
| **SU** | 0 | 24 | 26 | 33 | 20 | 23 | 20 | 22 | 7.0 | 10 | 11 | 9.8 | 3.8 | 15 | 14 | 0 |
| **Over** | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| **3D** | 22 | 24 | 25 | 27 | 27 | 32 | 33 | 33 | 33 | 34 | 35 | 36 | 39 | 39 | 51 |
| **SU** | 22 | 29 | 26 | 25 | 28 | 36 | 33 | 52 | 39 | 40 | 37 | 46 | 42 | 58 |
| **Over** | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| **3D** | 12 | 17 | 14 | 15 | 16 | 17 | 20 | 18 | 20 | 18 | 21 | 19 | 22 | 21 | 38 |
| **SU** | 32 | 33 | 35 | 33 | 31 | 43 | 50 | 41 | 43 | 58 | 46 | 51 | 52 | 55 | 82 |
| **Over** | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| **3D** | 1.5 | 1.9 | 2.6 | 1.6 | 1.9 | 1.6 | 2.5 | 2.4 | 2.6 | 2.8 | 2.5 | 2.6 | 3.0 | 3.3 | 3.0 | 4.7 |
| **SU** | 12 | 16 | 15 | 12 | 20 | 21 | 22 | 21 | 25 | 19 | 32 | 22 | 14 | 25 | 36 |
| **Is roof planar?** | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| **Topological errors** | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| **Ratio of the polygon faces in the roof** | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| **number of vertices on the constructed roofs** | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| **number of faces on the constructed roofs** | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |

### Baselines
- 3ds Max (3D)
- SketchUp (SU)

**3ds Max allows non-planar faces!**

**Manually triangulate faces to enforce planarity in SU!**

**More efficient**
Applications
Interactive Editing

Editing Operations optimization-based formulation allows interactive editing
✓ Move a vertex
✓ Move an edge
✓ Snap an edge
✓ Merge two faces
✓ Split a face
✓ Force two faces to be adjacent
Applications

Roof-Image Paired Dataset

Dataset: polygonal roof meshes paired with images

- >3K .obj
- Texture coordinates
- Face labels
- Roof synthesis
- Roof segmentation
- Roof detection

https://github.com/llorz/SGA21_roofOptimization/tree/main/RoofGraphDataset
Applications

Roof Synthesis from Scratch

**Face Adjacency Prediction**

- Transformer
- MLP
- GCNs
- Roof Optimization

Training set: 2105 samples, Testing set: 210 samples
Applications
Roof Synthesis from Scratch

✓ Synthesize roofs from scratch
✓ Synthesize roofs with different styles with the same outline
Summary

**Goal** roof modeling + roof embedding

**Baselines** commercial software & straight skeleton based methods

**Our Solution** roof graph representation + optimization-based construction

- Efficient & Flexible
- Interactive editing
- Image-Roof paired dataset
- Automatic roof synthesis: outline generation + adjacency prediction
Limitations & Future Work

Limitations

• Cannot directly handle curved roofs including stadiums and skyscrapers
• Did not model roof textures
• Did not touch on automatic reconstruction from images

Future Work

• End-to-end roof reconstruction from images
• Practical constraints for roof fabricability
• Roof texture synthesis
• ……
Intuitive and Efficient Roof Modeling for Reconstruction and Synthesis

Jing Ren¹,², Biao Zhang¹, Bojian Wu², Jianqiang Huang², Lubin Fan², Maks Ovsjanikov³, Peter Wonka¹
¹KAUST, ²Alibaba Group, ³Ecole Polytechnique, IP Paris

Acknowledgement The authors thank the anonymous reviewers for their valuable comments. Parts of this work were supported by the KAUST OSR Award No. CRG-2017-3426, the ERC Starting Grant No. 758800 (EXPROTEA), the ANR AI Chair AIGRETTE, and Alibaba Innovative Research (AIR) Program. We would like to thank Guangfan Pan and Jiacheng Ren for helping modeling roofs in 3ds Max and SketchUp, Jialin Zhu for helping designing the web-based roof annotation UIs, Jianhua Guo and Tom Kelly for helping with the comparison to the weighted straight skeleton. We thank Muxingzi Li for helping editing the supplementary video. We also thank Chuyi Qiu, Tianyu He, and Ran Yi for their valuable suggestions and comments.

Code & Data https://github.com/llorz/SGA21_roofOptimization
Supplementary materials
**Remark 1** The intersecting line of two adjacent 3D planar faces with fixed outline edges, is either parallel to both outline edges, or intersects the two outline edges at the same point. The same conclusion holds when we project the 3D planar faces to $xy$-plane.

Use Remark 1 to check if a 2D embedding is valid or not.
Methodology

Spectral Initialization

\[
\min_{\vec{X}_R} \left\| \left( \vec{X}_O \right)^T \mathcal{L}_V \left( \vec{X}_O \right) \right\|_F^2
\]

- \( \vec{X}_O \): 2D position of the outline vertices
- \( \vec{X}_R \): 2D position of the roof vertices
- \( A_V \): vertex adjacency matrix of the roof graph, \( A_V(i, j) = 1 \) iff \( v_i \sim v_j \)
- \( \mathcal{L}_V \): graph Laplacian of the roof graph, \( \mathcal{L}_V = \text{diag}(1^T A_V) - A_V \)

No user input as initial embedding

- Random/Zero initialization can lead to undesirable global minima
- Spectral initialization can avoid self-intersections
Results

Face with Multiple Outline Edges

More expressive

- **Primal graph**: can model this case directly
- **Dual graph**: the face adjacency matrix can be modified to model this case
Applications
Interactive Editing

Input image

Straight Skeleton

Weighted
Straight Skeleton

Ours
Applications
Interactive Editing

Input image

Straight Skeleton

Weighted Straight Skeleton

Ours
Applications

Roof Synthesis from Scratch

Input nothing

Output a 2D outline as a sequence of vertices \( \{v_1, v_2, \ldots, v_n\} \)

Tokenization (flattened) vertex position values belong to \( \{1, 2, \ldots, 2^b\} \)

Architectures

- 6 blocks: self-attention + MLP
- Embedding dimension = 384
- Self-attention: 12 heads
- MLPs: hidden dimension = 1536

| token | \( v_1 \) | \( v_2 \) | \( v_3 \) | \( v_4 \) | \( \ldots \) | \( v_{2\|O}-1 \) | \( v_{2\|O} \) | \( s \) |
|-------|-----------|-----------|-----------|-----------|-------------|-------------|-------------|------|
| position | 1 | 1 | 2 | 2 | \( \ldots \) | \( n_O \) | \( n_O \) | \( n_O + 1 \) |
| coord | 1 | 2 | 1 | 2 | \( \ldots \) | 1 | 2 | 1 |
Applications
Roof Synthesis from Scratch

Face Adjacency Prediction

**Input** outline \( \{v_1, v_2, \ldots, v_n\} \)

**Output** probability \( p_{ij} \) of the face \( f_i \) being adjacent to the face \( f_j \)

**Architectures** we stack following basic building blocks 4 times
- **Adjacency block**
- **Edge block**
- **Global block**

Each block updates representation vectors (adjacency, edge, global).

**Loss objective** binary cross entropy
Applications
Roof Synthesis (Resolve Ambiguities in Predicted Adjacencies)

**Case 01** interior vs. exterior region

1. **Resolve ambiguity 01**: check if each edge in the dual graph (adjacency) lies in the **interior** of the roof
2. **Resolve ambiguity 02**: detect edge **intersections** in the dual graph, then remove the confliction by
   - Greedy (most likely one)
   - Sampling (all possible ones)

**Algorithm** extract valid dual graph

- **Exterior Adjacency** $(f_1, f_3)$
- **Interior Adjacency** $(f_1, f_3)$

**Case 02** conflicting adjacencies
Results
Evaluate SketchUp

- The artist hid some edges to make the constructed roof visually consistent with the input image.
  - $n_1$ reports the number of faces that are visible
  - $n_2$ reports the actual number of faces that were created
  - $\text{ERR} = n_2 - n_1$
  - Red: polygonal faces in the created roof mesh

\[ \text{ERR} = 28 \]
\[ \text{ERR} = 29 \]
Existing Methods
Commercial Software: 3ds Max

Input

- Add vertex
- Add face

- Cut faces
- Move vertex

Non planar!
Existing Methods

Commercial Software: SketchUp

Input
Specify roof topology

1. Build roof beams
2. Add planar roof tops

Extra complexity!