[Supplementary Material] Structured Outdoor Architecture Reconstruction by Exploration and Classification

Fuyang Zhang Xiang Xu Nelson Nauata Yasutaka Furukawa
Simon Fraser University, BC, Canada
{fuyangz, xuxiangx, nnauata, furukawa}@sfu.ca

The supplementary document provides 1) architecture specifications of our junction/edge classifiers and a baseline system (Per-edge model) and 2) additional experimental results.

1. Architecture specifications

1.1. Junction/edge classifiers

Table 1 shows the full architecture specifications of our junction/edge classifiers.

In our classifiers, the deep U-Net is composed of a stack of six downstream/upstream convolution layers (kernel=3, stride=2, padding=1), whose input is a building RGB image and output is a $256 \times 256 \times 64$ feature volume.

The shallow U-Net contains five downstream/upstream convolution layers (kernel=3, stride=2, padding=1). Shallow network takes the building feature volume, pretrained corner/edge confidence mask (2 dims), plus, corner/edge masks (2 dims) rendered from the input graph. The corned/edge confidence mask and corner/edge masks are first passed to a Conv-block, resulting in a $16 \times 256 \times 256$ feature, which concatenates with feature volume from deep U-Net as next stage input. The final output is a correct/incorrect prediction mask $256 \times 256 \times 1$.

Our pretrained model is also a U-Net. We train this network from scratch for segmenting corners and edges as two-channels. Supervision are coarser GT corners and edges masks rendered with 5 pixels of diameters and 3 pixels of thickness. Weights are kept fixed after training once.

1.2. Per-edge model

Per-edge model is a baseline model to produce the initial graphs (Sec. 6.1 in the main paper), which classifies correctness of every edge independently. We use the corner detection from Conv-MPN [3] to detect corners, then use a CNN-based edge classifier for every pair of corners independently.

The edge classifier is equivalent to Conv-MPN architecture without the message passing [3]. Specifically, a fully CNN network takes RGB image and one edge rendered mask as input (4 channels totally), then estimates the correctness of the corresponding edge. The supervision is the same as the edge label mentioned in "classification label generation" in Sec. 4.2. We refer to the Conv-MPN paper [3] for the full architectural specifications.

2. Additional experimental results

Fig. 1,2 show qualitative comparisons against the competing three systems from the ablation studies in the main paper, (1) *no data augmentation* (denoted as w/o exp), (2) *data augmentation by random sampling* (rand exp), and (3) *no geometry classification* (w/o classifiers). Two RL-based methods are also shown in Fig. 1,2, (4) REPL [1] and (5) Lin *et al*. [2]. Conv-MPN [3] reconstructions are used as initial models.

Fig. 3-34 show additional experimental results as Fig. 6 of the main paper.

Table 1. Architecture specification. $[\cdot]$ represents a layer block. $\times i$ denotes repeating the layers i times. The shallow U-Net is half resolution and 1 block less than the deep U-Net, except first layer takes output from deep U-Net as input. d = $\{32, 64, 128, 256, 512, 1024, 2048\}$. The output of the shallow U-Net after Sigmoid layer is applied a linear transform, mapping $[0, 1]$ to $[-1, 1]$ for calculating classification scores.

Module	Stage	Specification	Output Size
Deep U-Net	$Input$	$3 \times 3 \times 3, 32, stride = 1$ $32 \times 3 \times 3, 32, stride = 1$	$32 \times 256 \times 256$
	$Down_{\{1...6\}}$	$\begin{bmatrix} d_i \times 3 \times 3, d_i, stride = 2 \\ d_i \times 3 \times 3, d_{i+1}, stride = 1 \\ d_{i+1} \times 3 \times 3, d_{i+1}, stride = 1 \end{bmatrix} \times 6$	$2048 \times 4 \times 4$
	$Up_{\{1...6\}}$	$\begin{bmatrix} d_{i+1} \times 4 \times 4, d_i, stride = 2(deconv) \\ d_{i+1} \times 3 \times 3, d_i, stride = 1 \\ d_i \times 3 \times 3, d_i, stride = 1 \end{bmatrix} \times 6$	$32 \times 256 \times 256$
	Out	$64 \times 3 \times 3, 1, stride = 1$ $Sigmoid$	$64 \times 256 \times 256$
Shallow U-Net	$Input$	$4 \times 3 \times 3, 16, stride = 1$ $16 \times 3 \times 3, 16, stride = 1$	$16 \times 256 \times 256$
	$Down_{\{1...5\}}$	$\begin{bmatrix} (16+64) \times 3 \times 3, (16+64), stride = 2 \\ (16+64) \times 3 \times 3, 64, stride = 1 \\ 64 \times 3 \times 3, 64, stride = 1 \end{bmatrix}$ $\begin{bmatrix} d_i \times 3 \times 3, d_i, stride = 2 \\ d_i \times 3 \times 3, d_{i+1}, stride = 1 \\ d_{i+1} \times 3 \times 3, d_{i+1}, stride = 1 \end{bmatrix} \times 4$	$1024 \times 4 \times 4$
	$Up_{\{1...5\}}$	$\begin{bmatrix} d_{i+1} \times 4 \times 4, d_i, stride = 2(deconv) \\ d_{i+1} \times 3 \times 3, d_i, stride = 1 \\ d_i \times 3 \times 3, d_i, stride = 1 \end{bmatrix} \times 5$	$32 \times 256 \times 256$
	Out	$32 \times 3 \times 3, 1, stride = 1$ $Sigmoid$	$1 \times 256 \times 256$

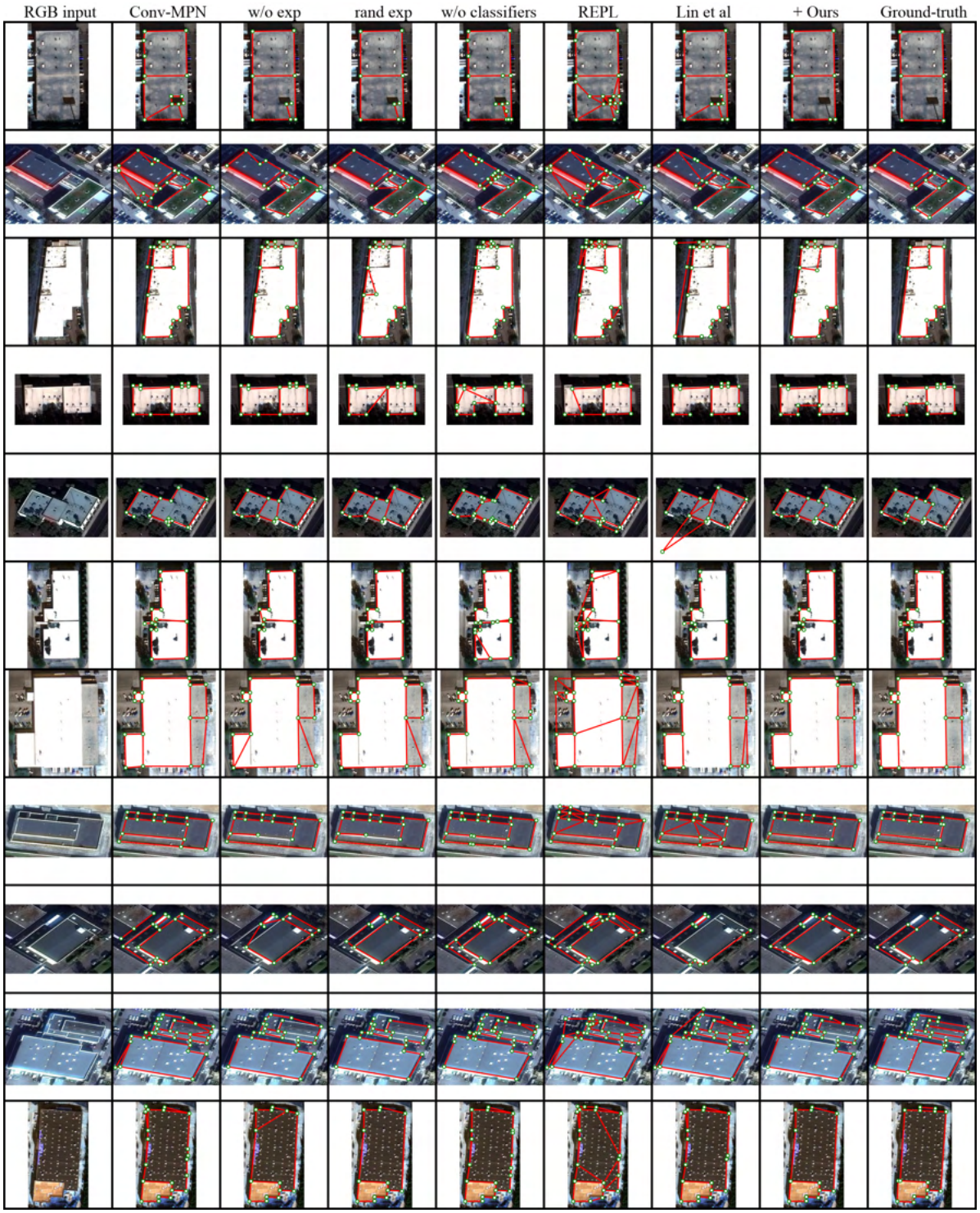

Figure 1. Additional qualitative comparisons against baseline systems and RL-based methods.

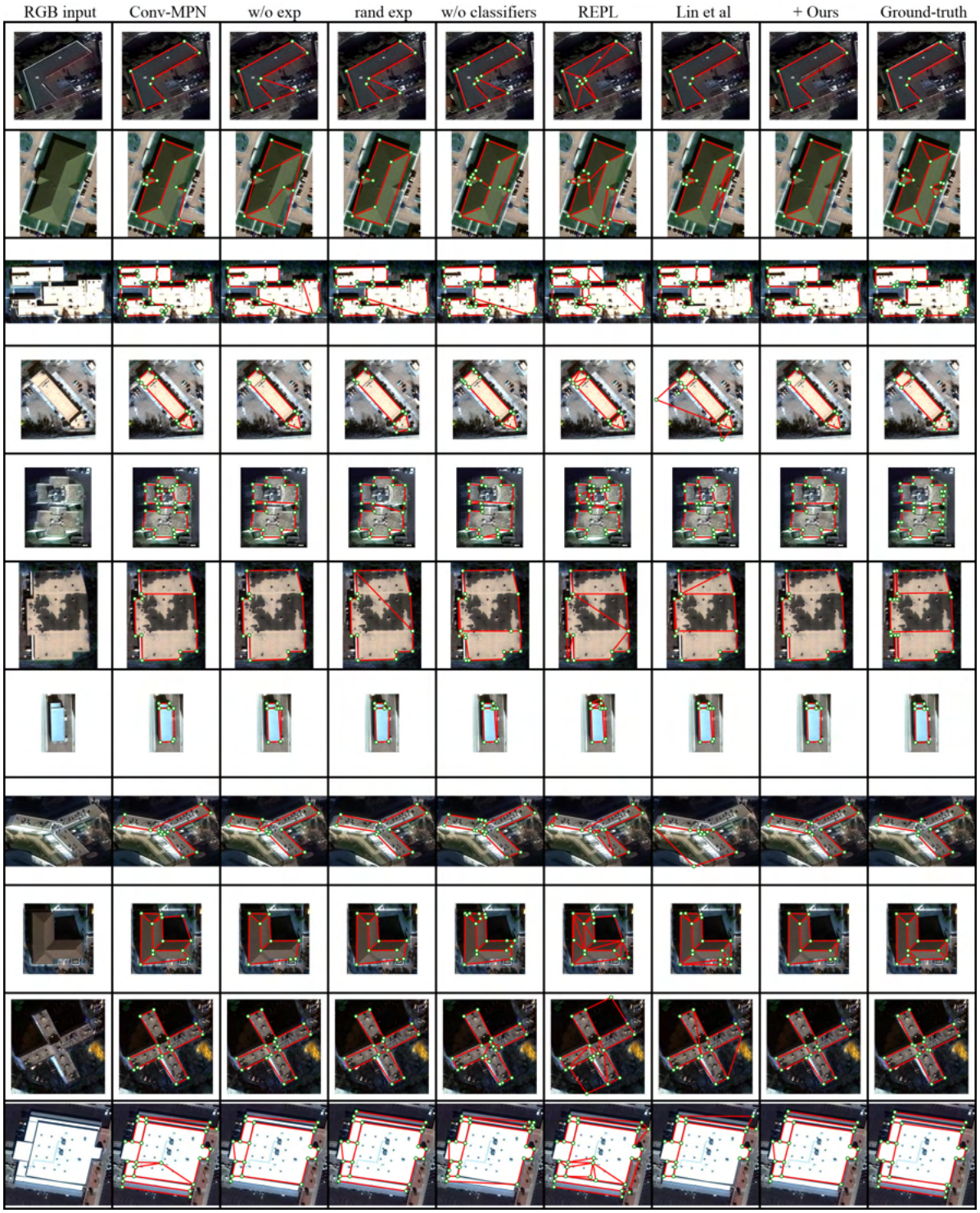

Figure 2. Additional qualitative comparisons against baseline systems and RL-based methods.

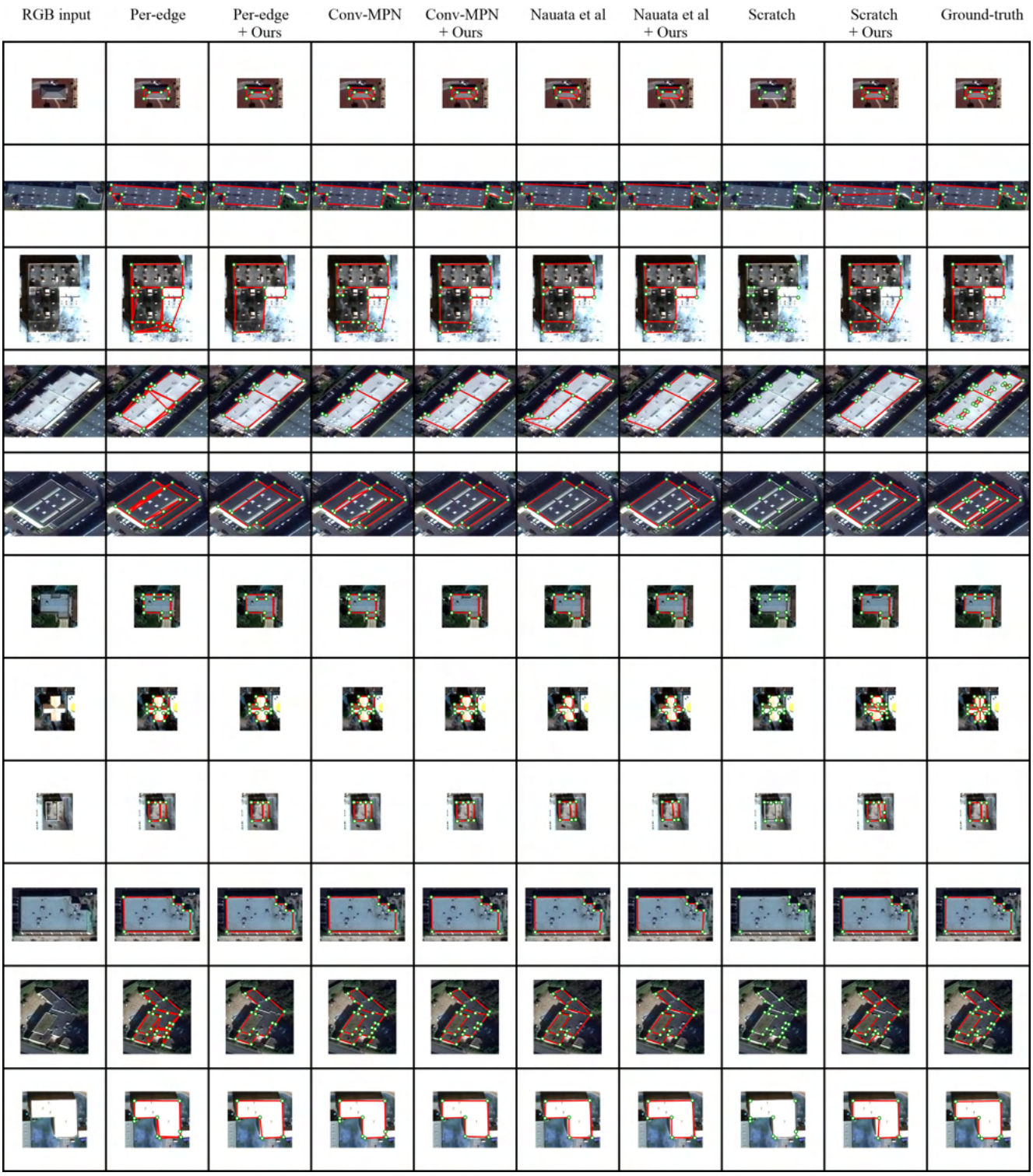

Figure 3. Additional qualitative results.

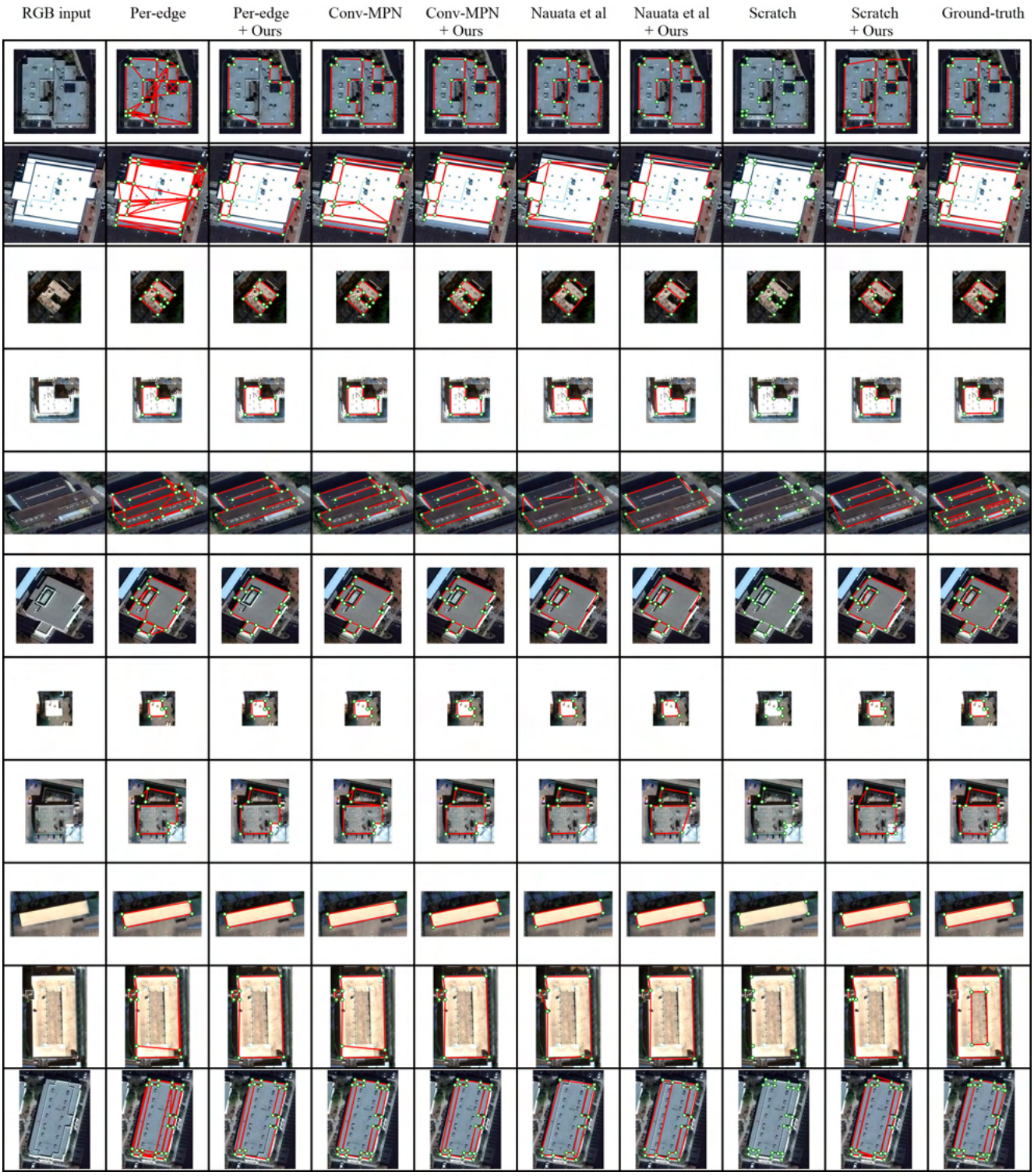

Figure 4. Additional qualitative results.

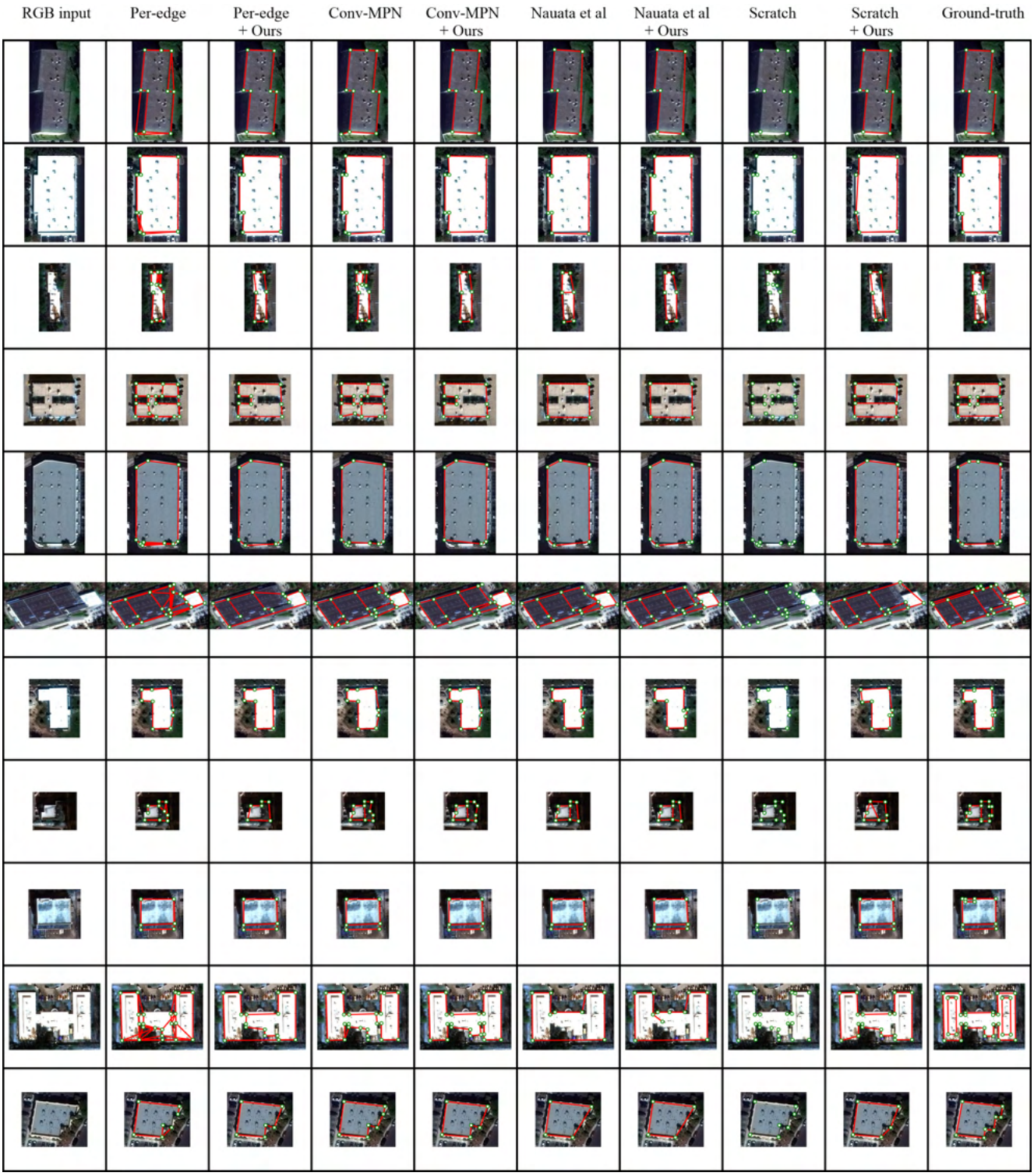

Figure 5. Additional qualitative results.

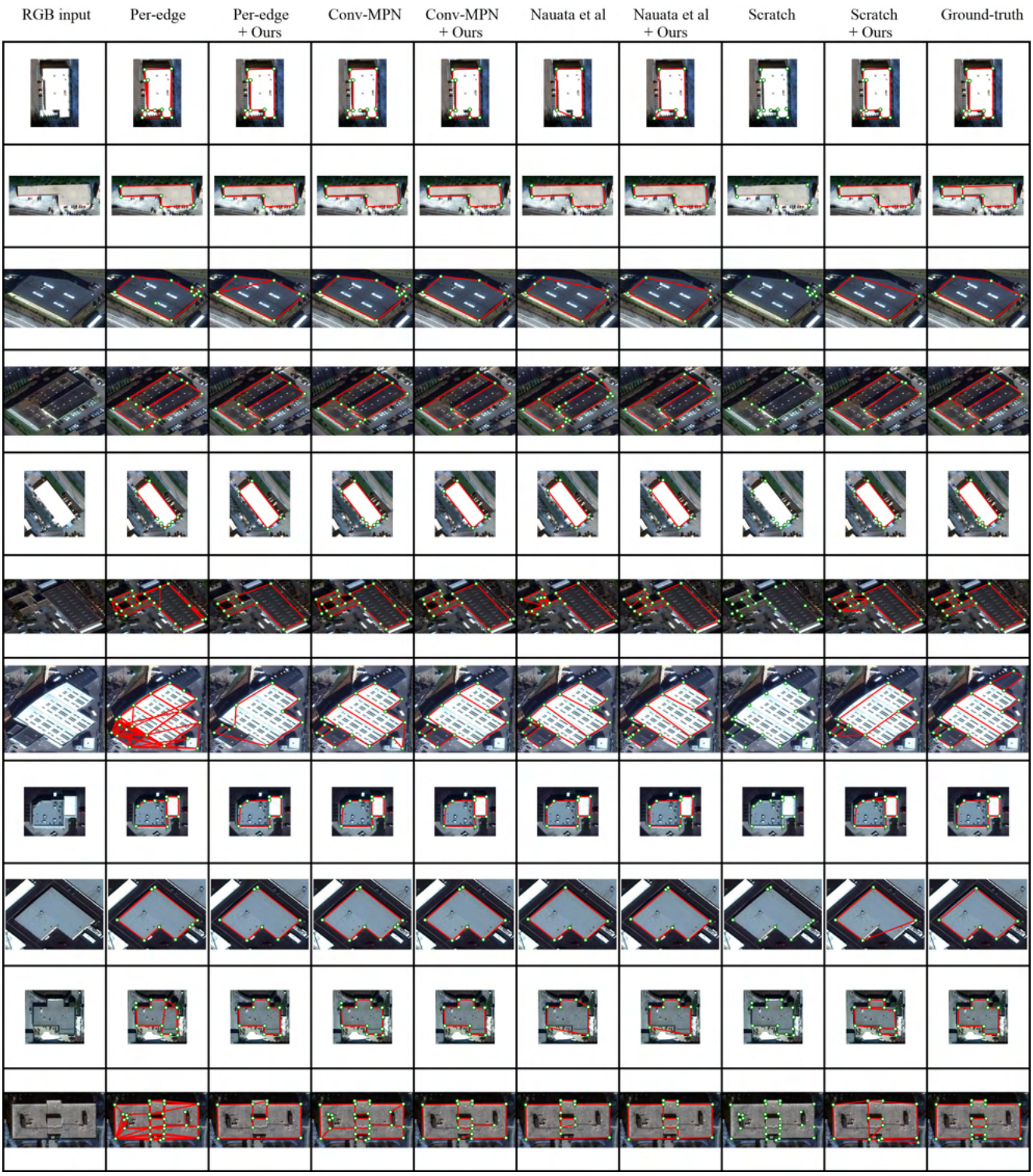

Figure 6. Additional qualitative results.

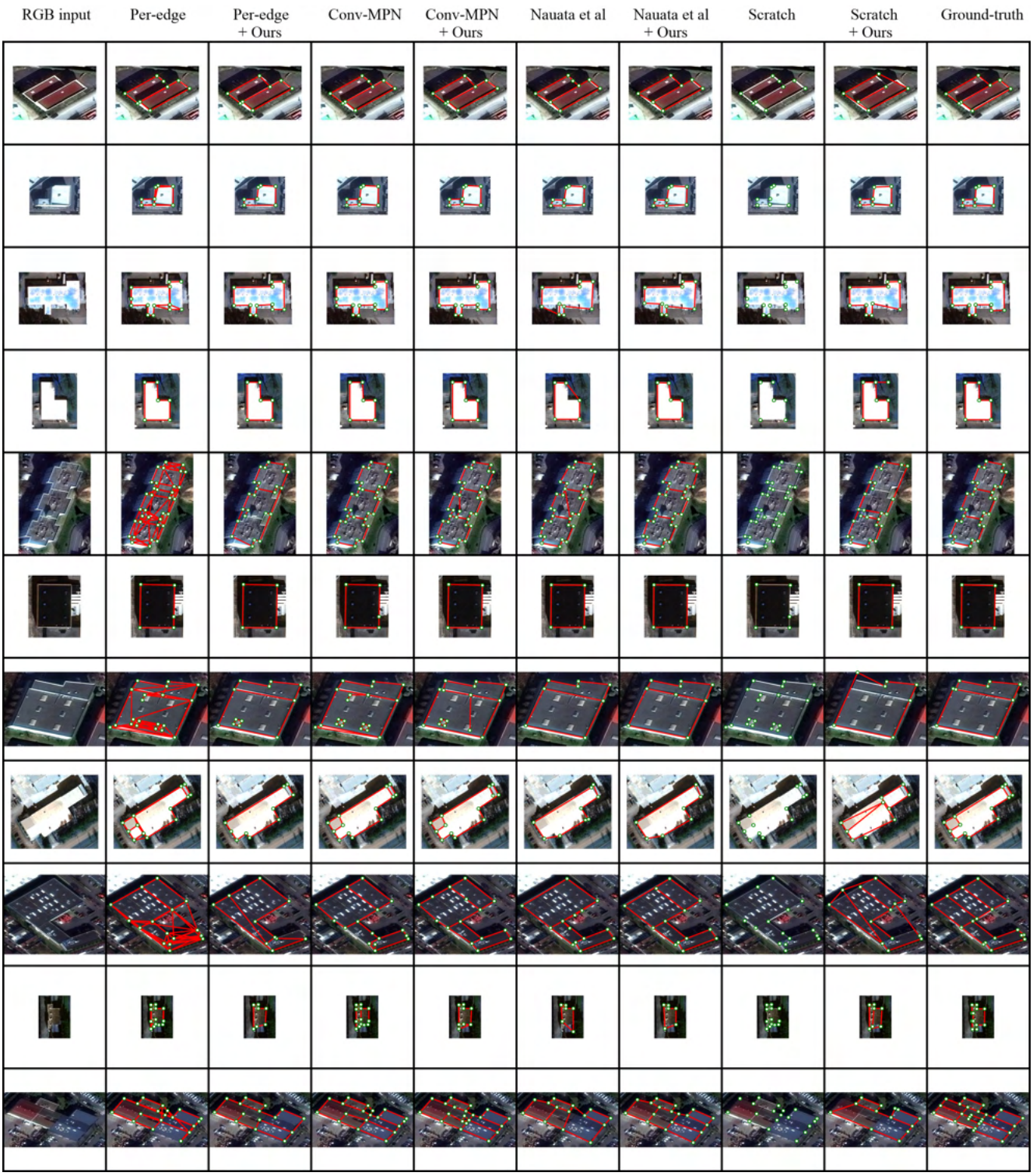

Figure 7. Additional qualitative results.

Figure 8. Additional qualitative results.

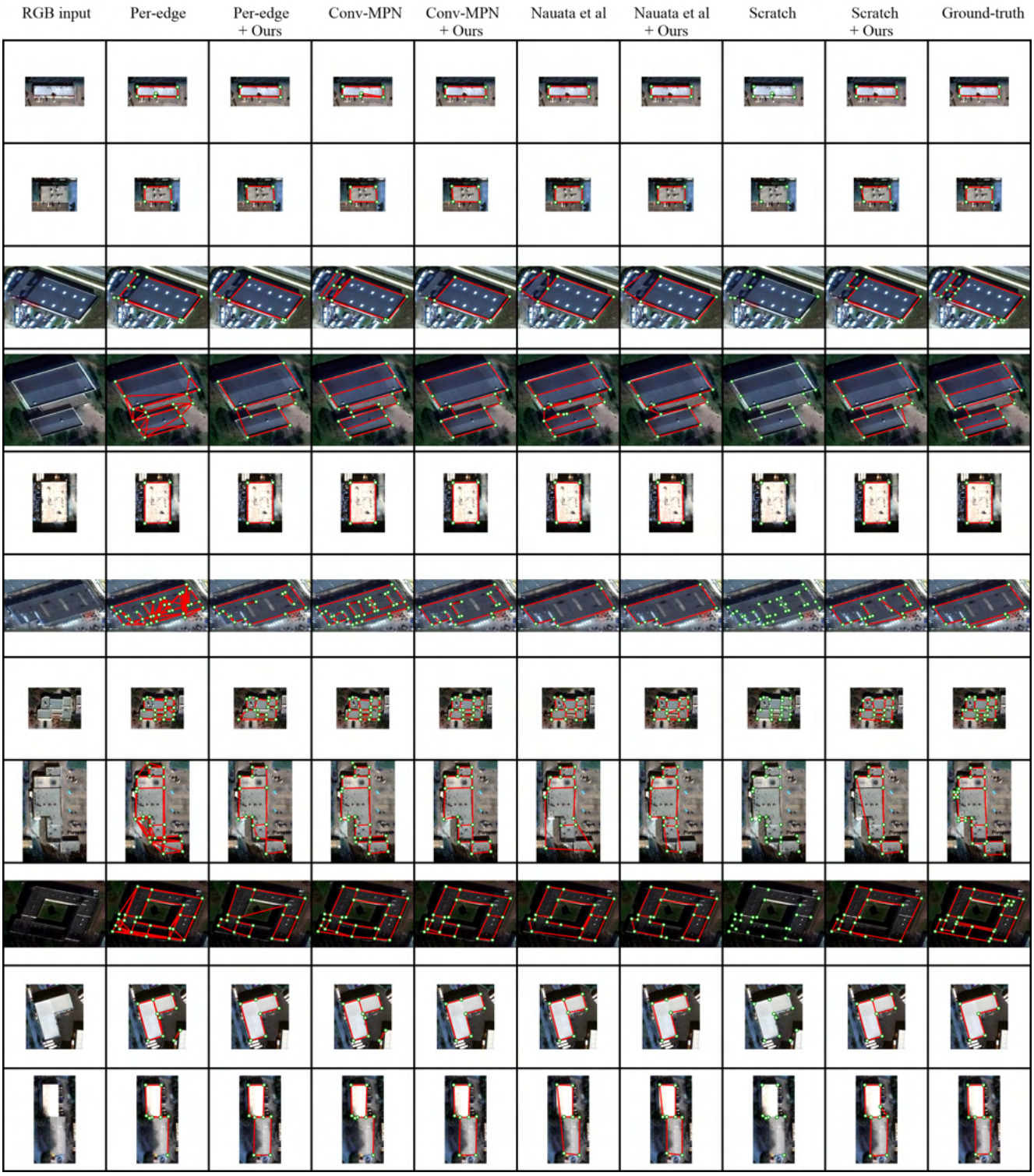

Figure 9. Additional qualitative results.

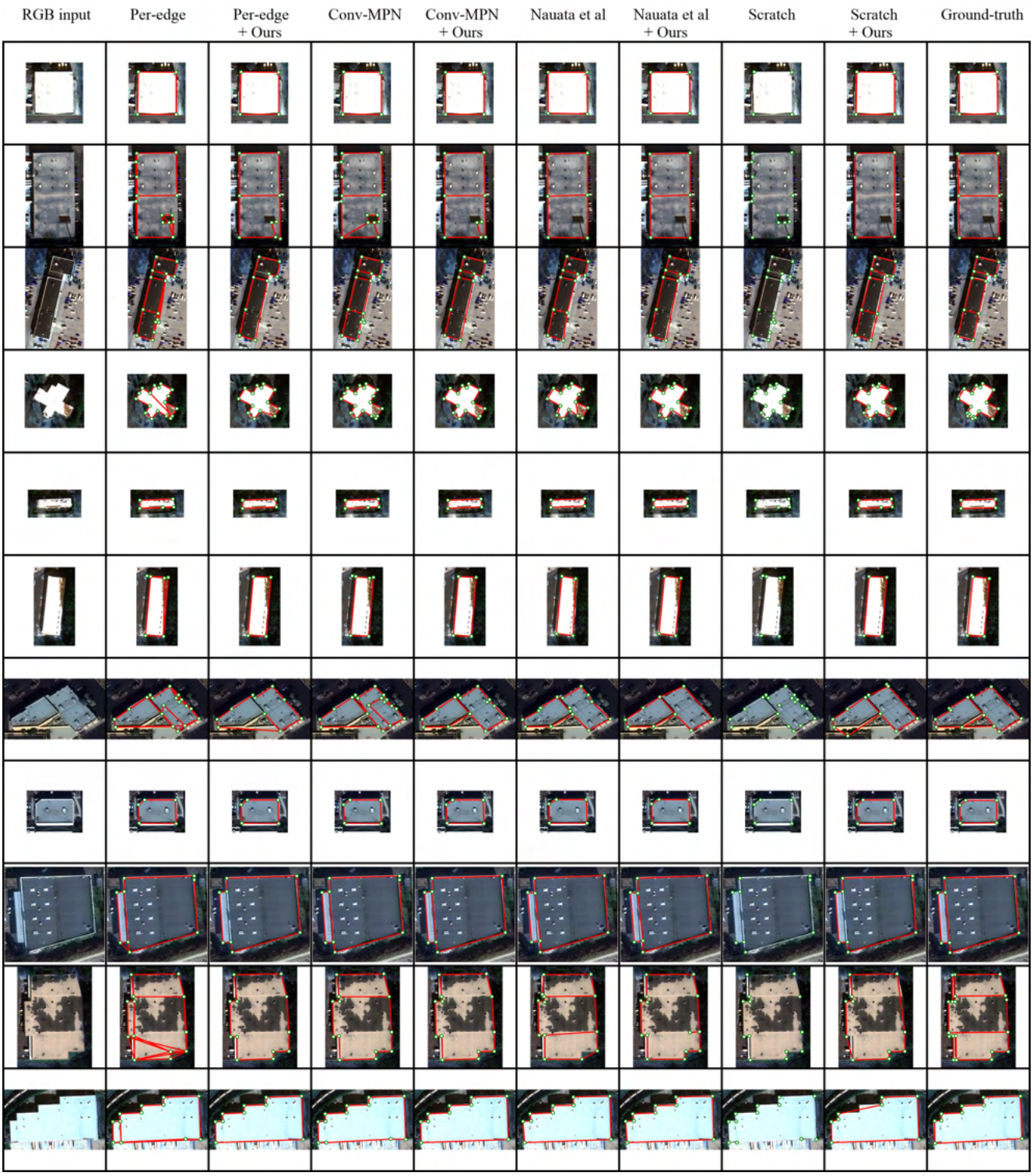

Figure 10. Additional qualitative results.

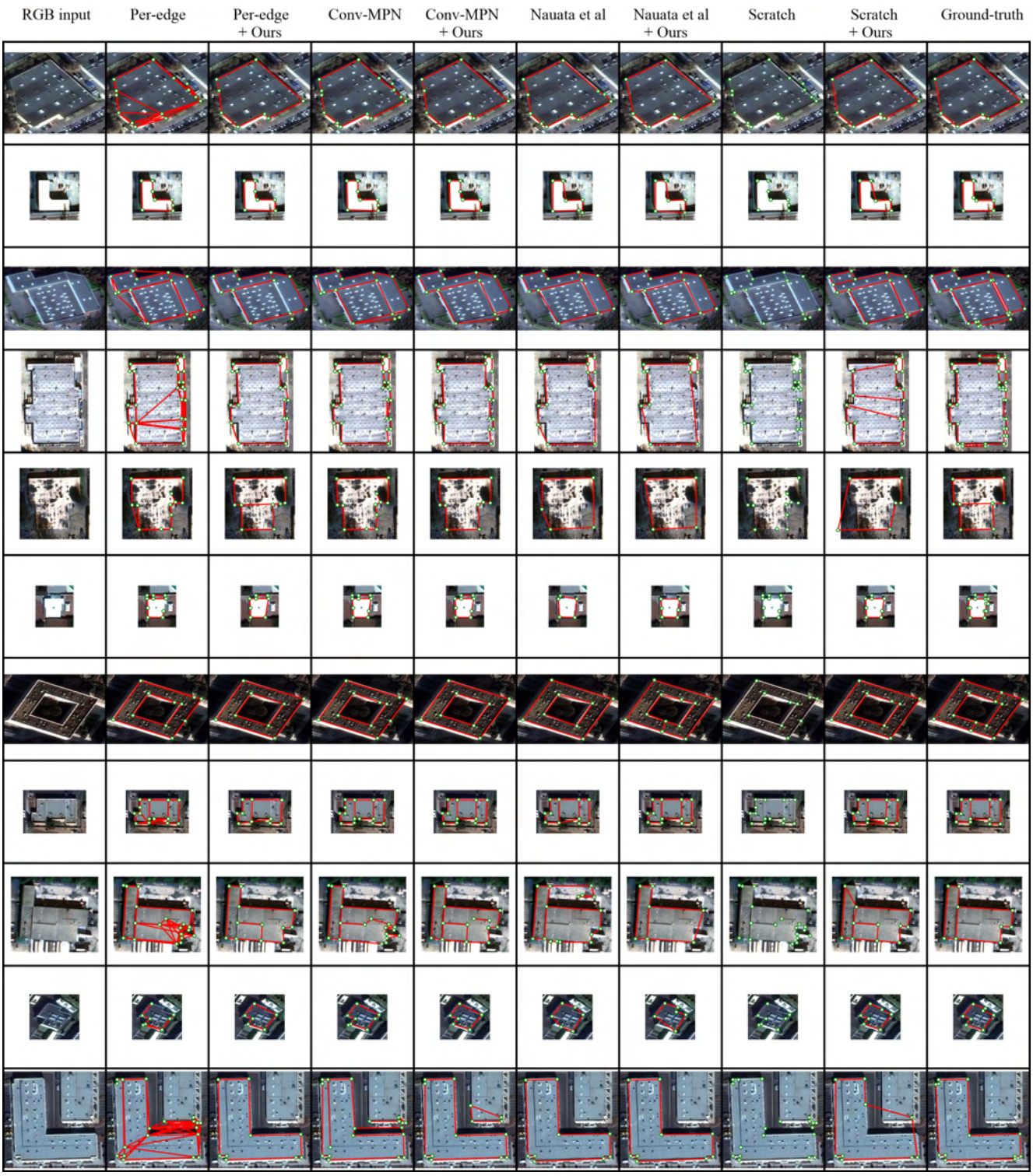

Figure 11. Additional qualitative results.

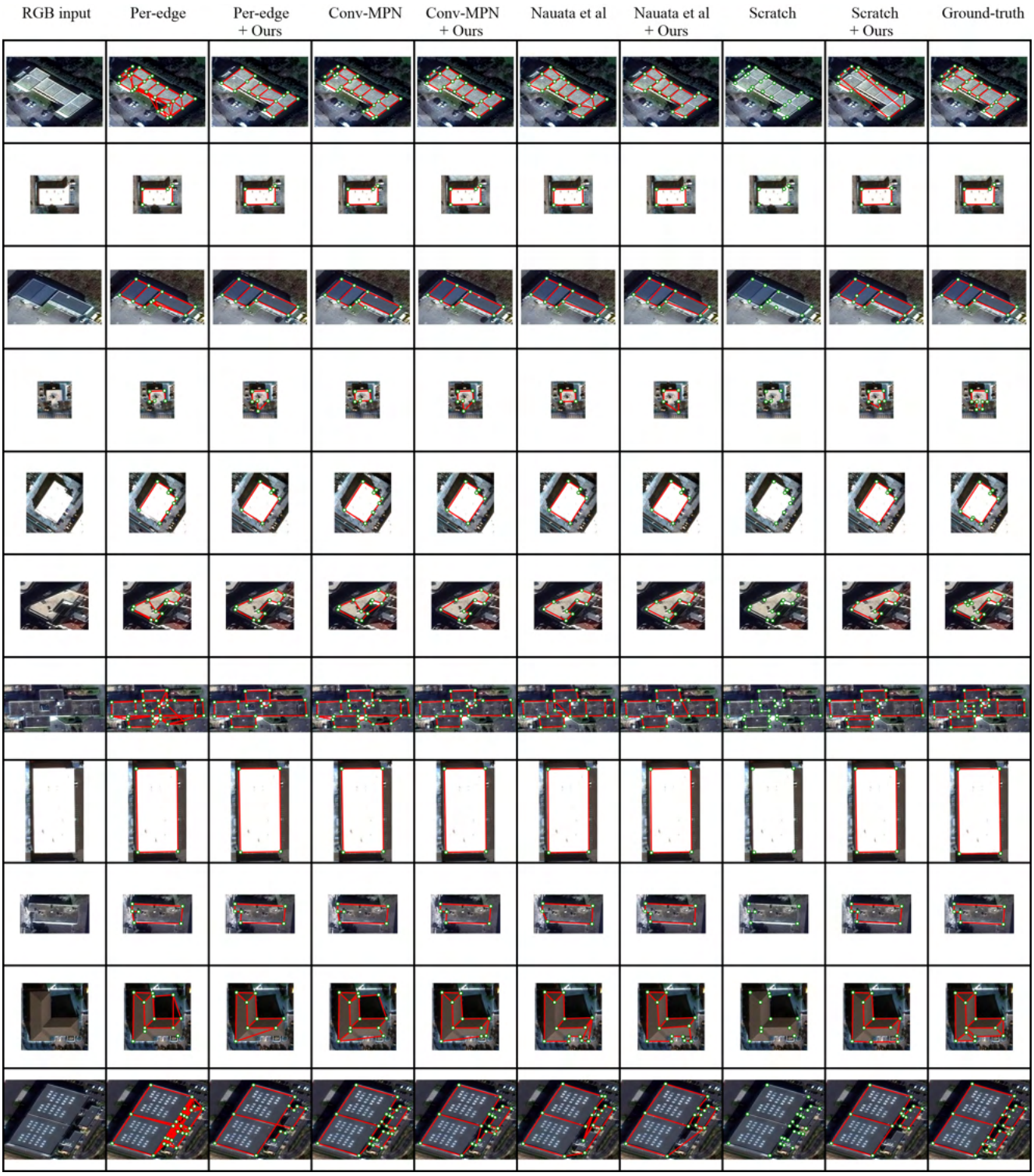

Figure 12. Additional qualitative results.

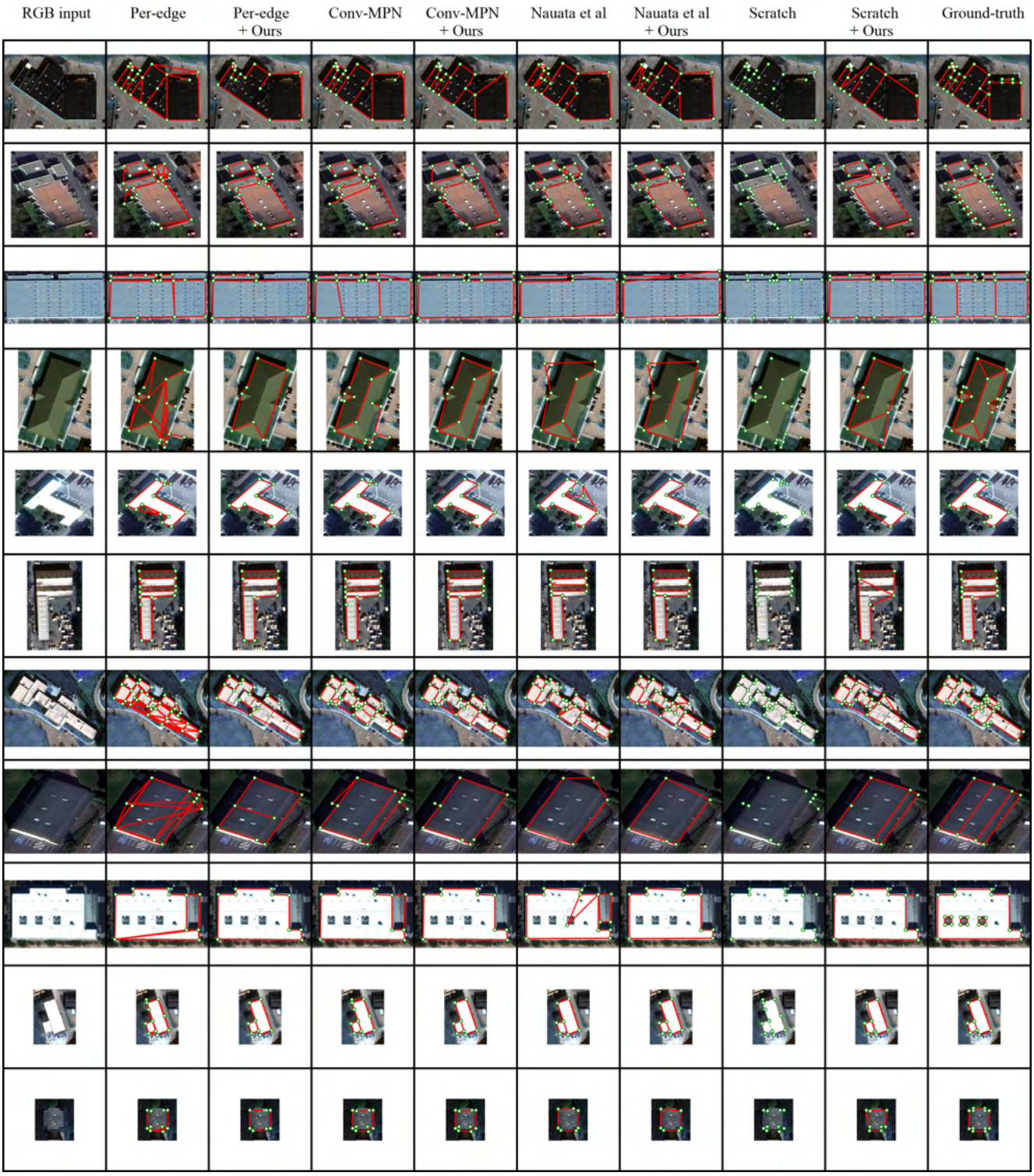

Figure 13. Additional qualitative results.

Figure 14. Additional qualitative results.

Figure 15. Additional qualitative results.

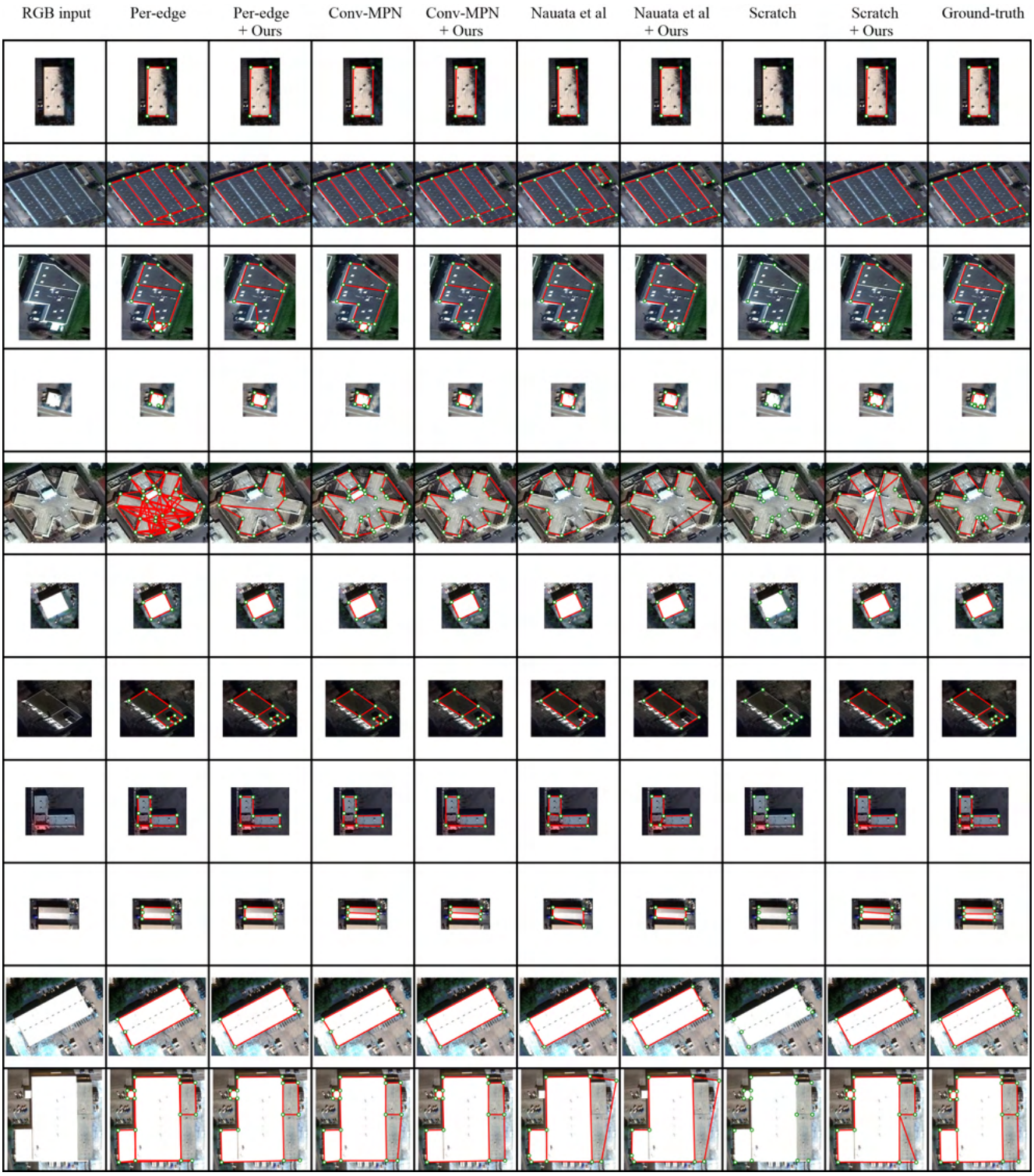

Figure 16. Additional qualitative results.

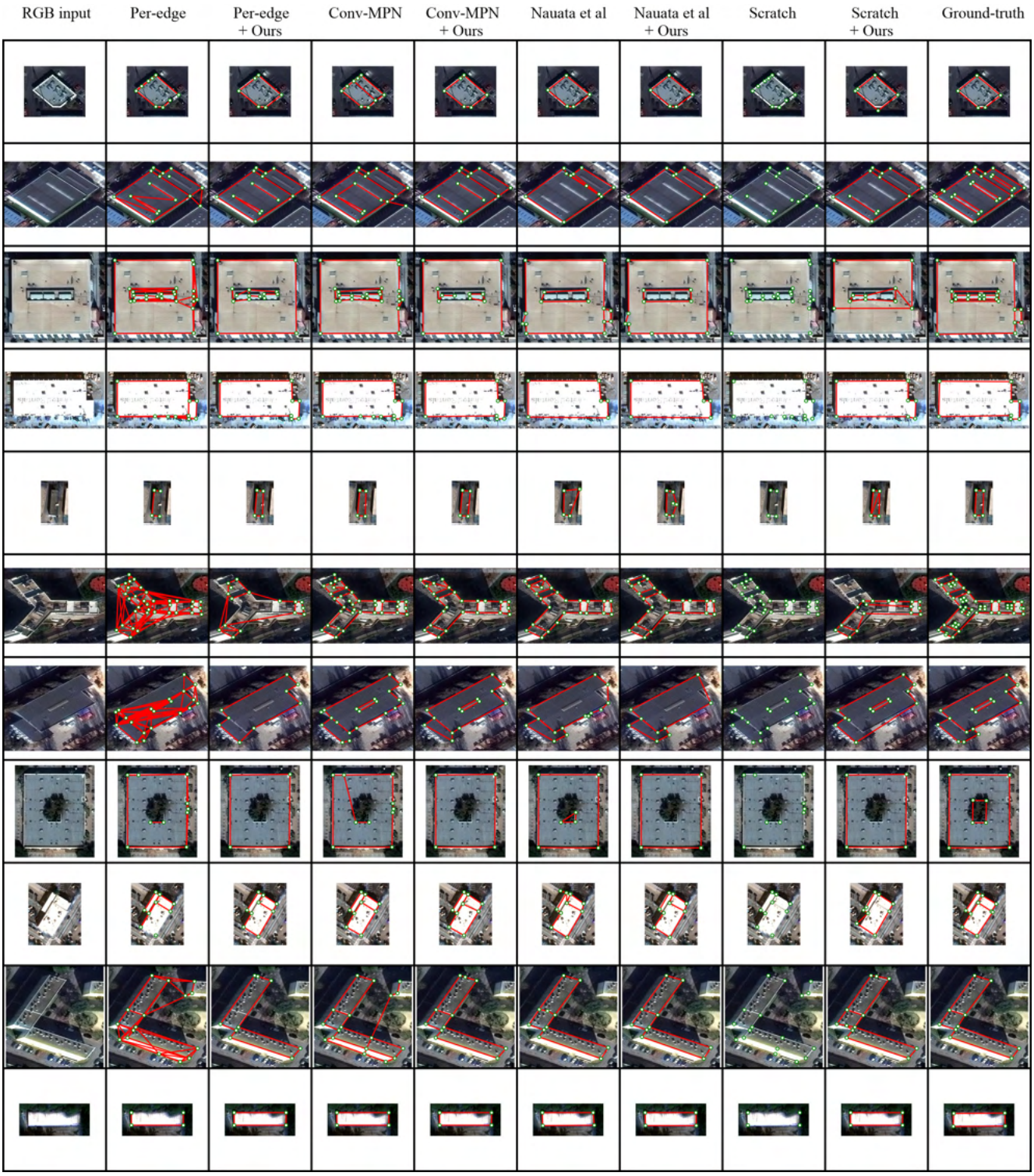

Figure 17. Additional qualitative results.

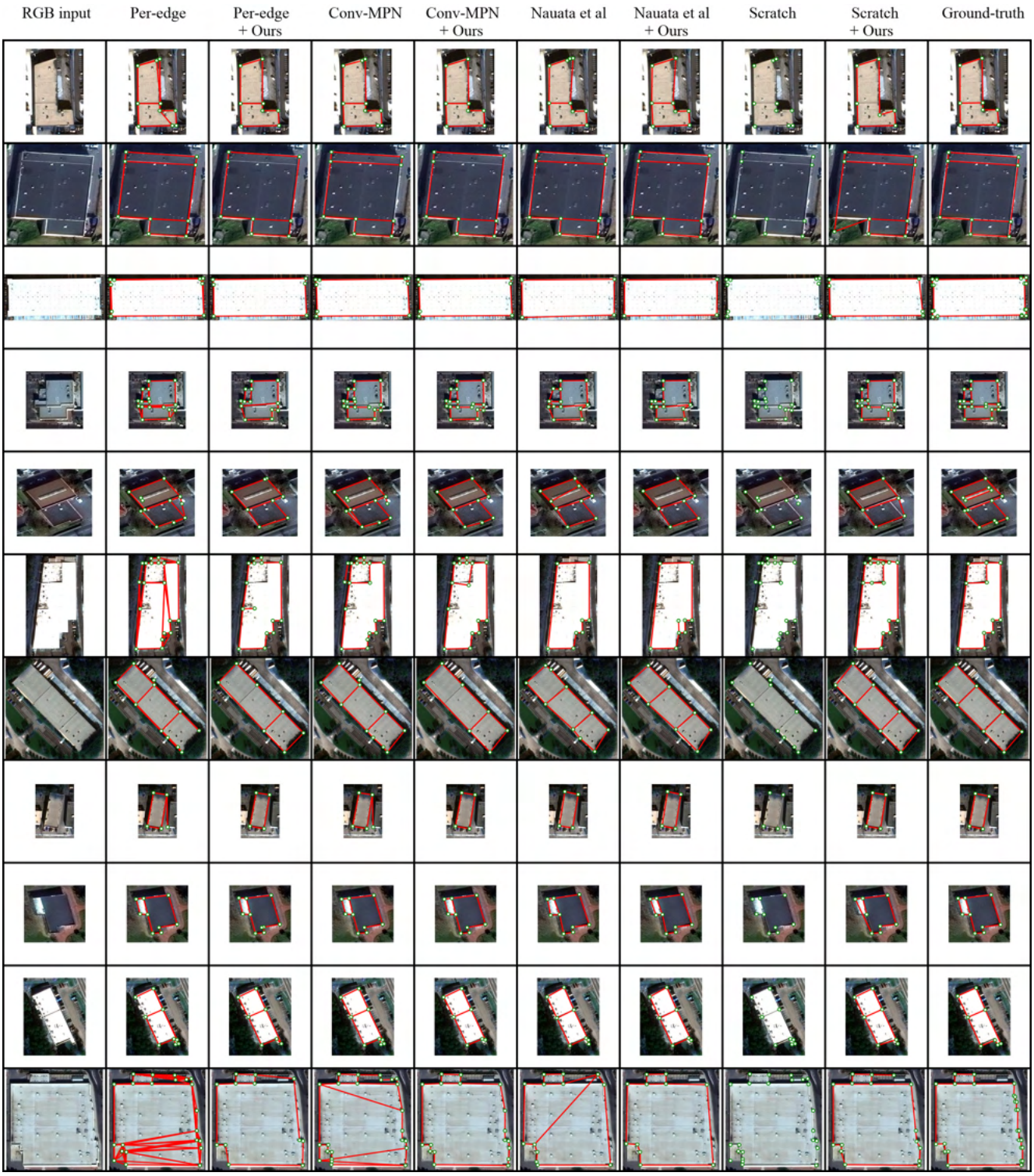

Figure 18. Additional qualitative results.

Figure 19. Additional qualitative results.

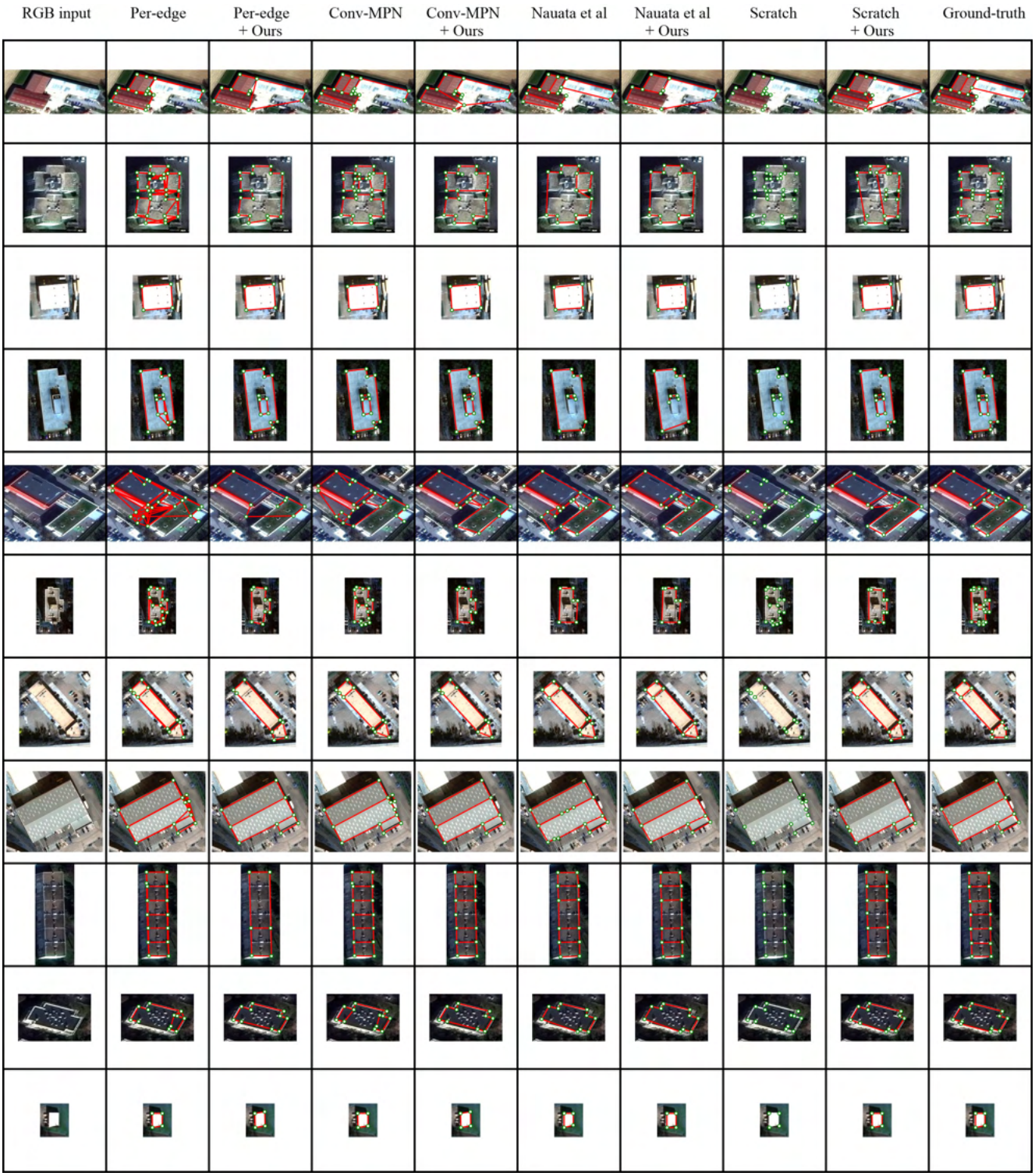

Figure 20. Additional qualitative results.

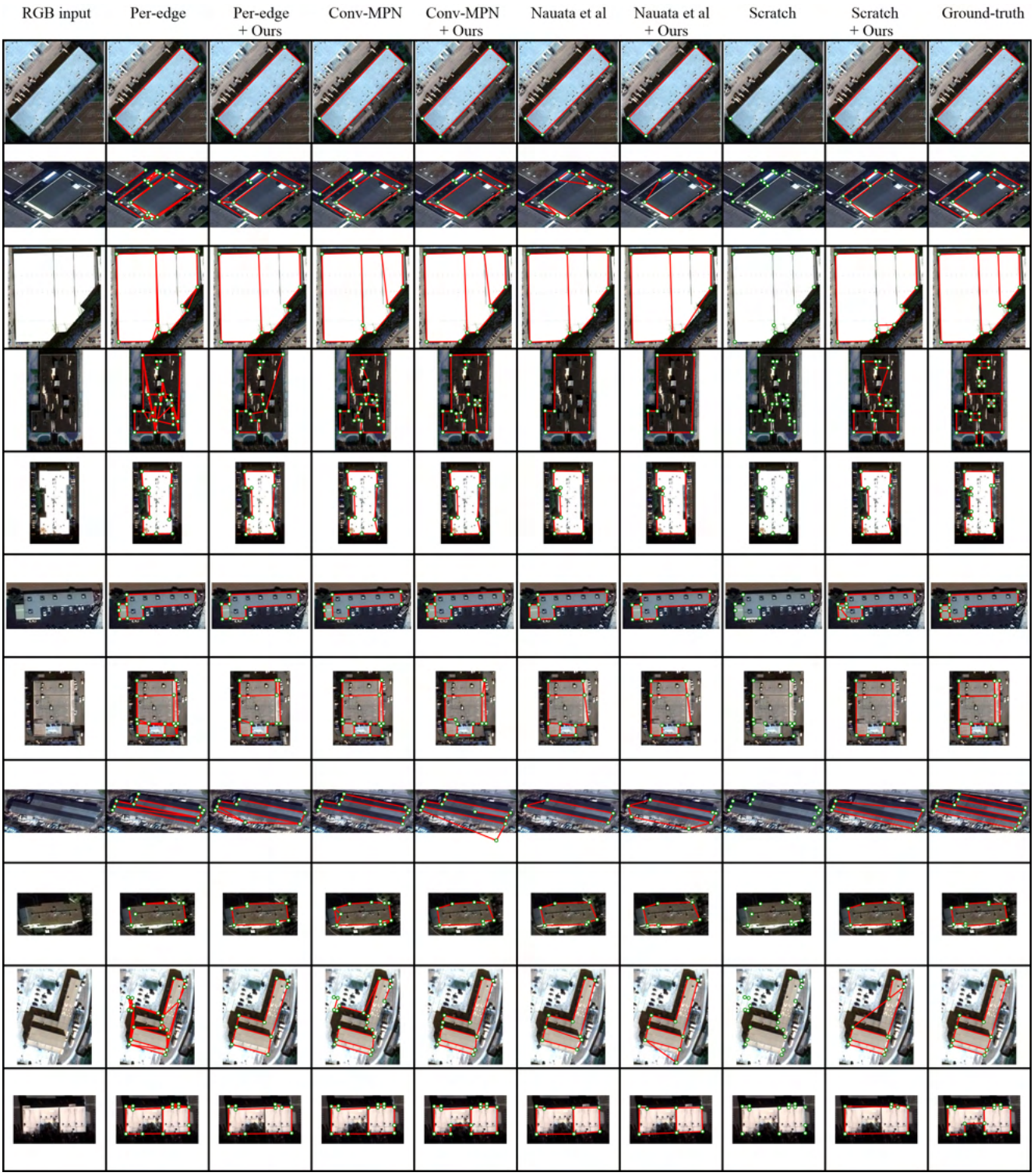

Figure 21. Additional qualitative results.

Figure 22. Additional qualitative results.

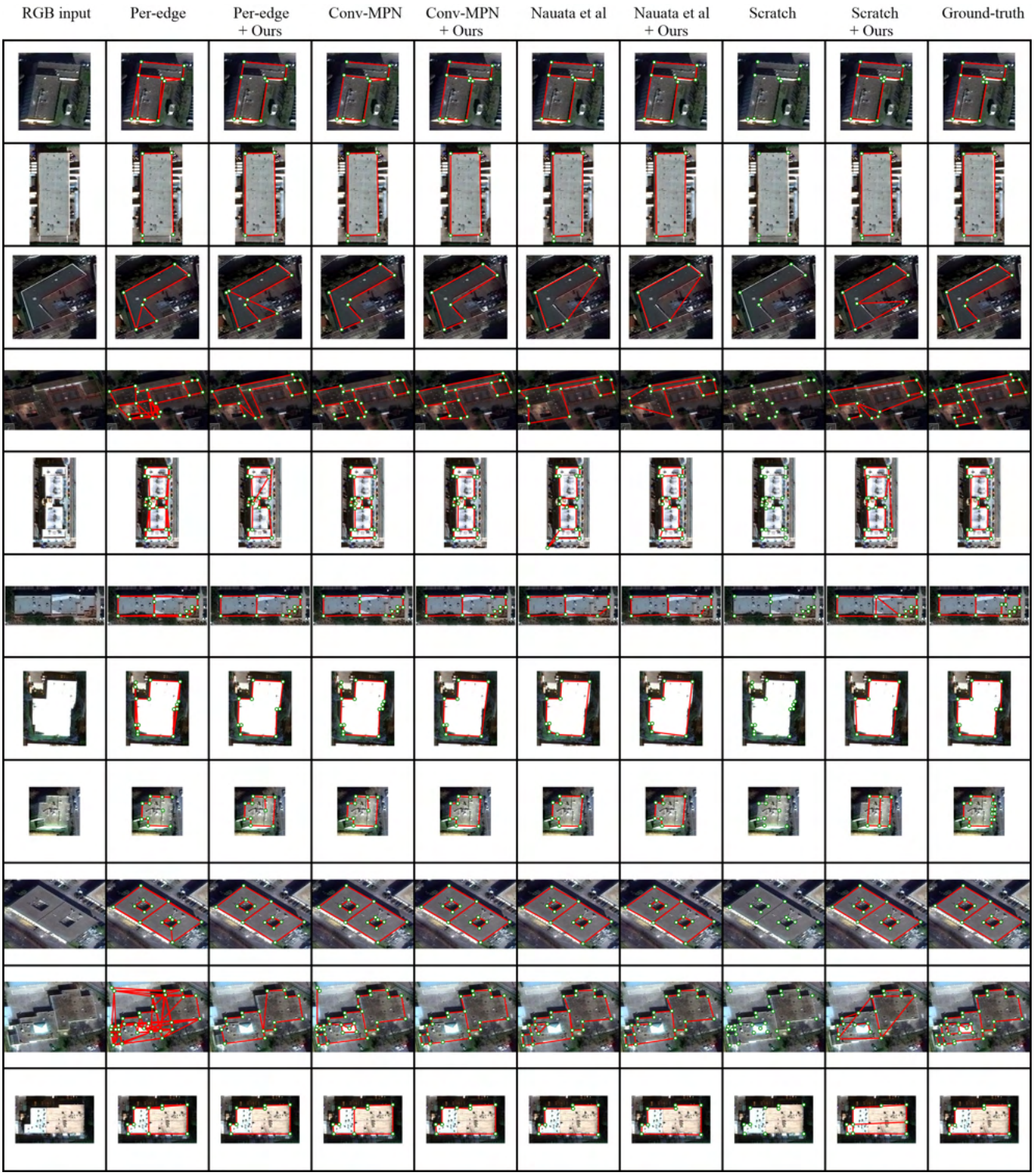

Figure 23. Additional qualitative results.

Figure 24. Additional qualitative results.

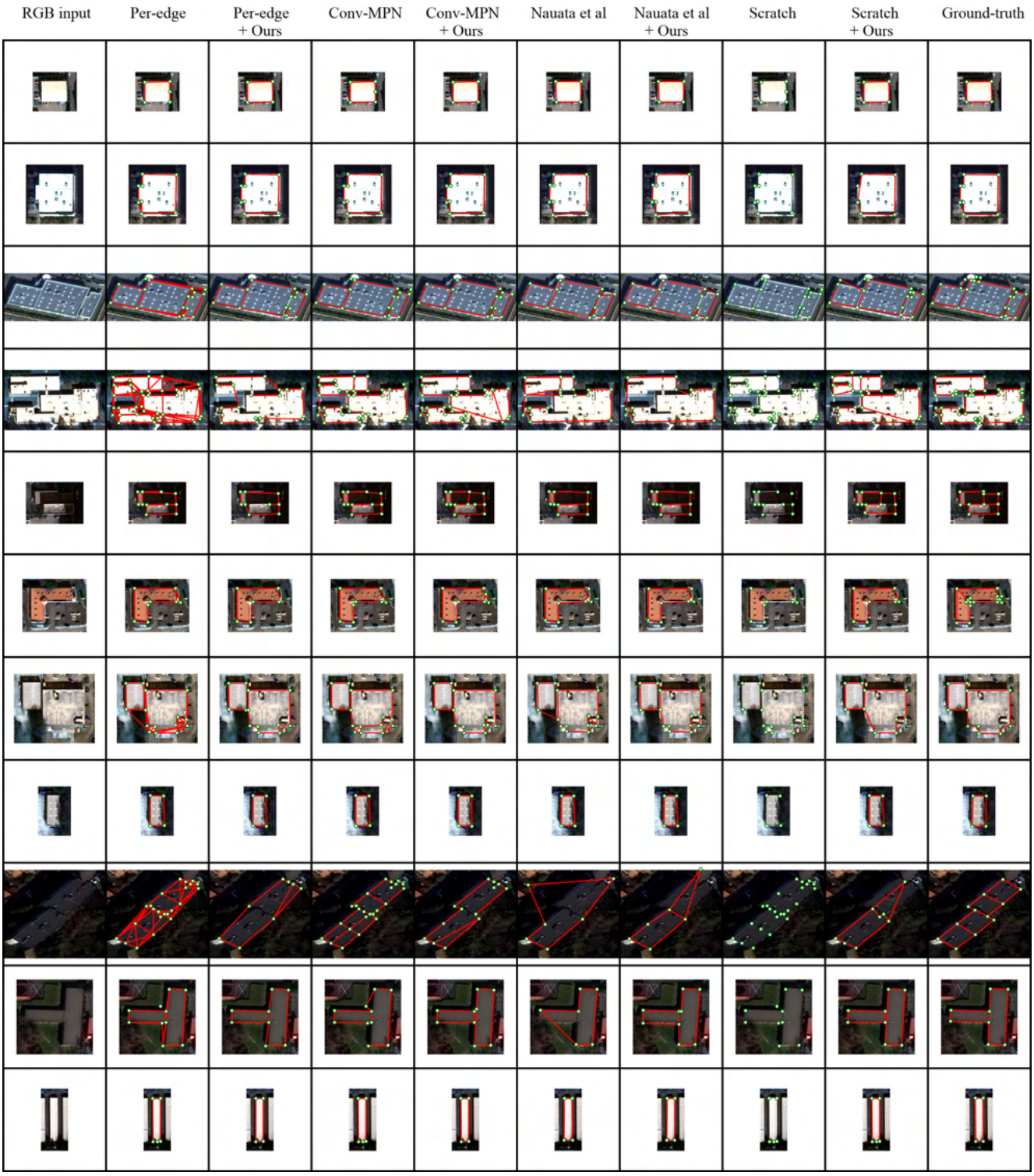

Figure 25. Additional qualitative results.

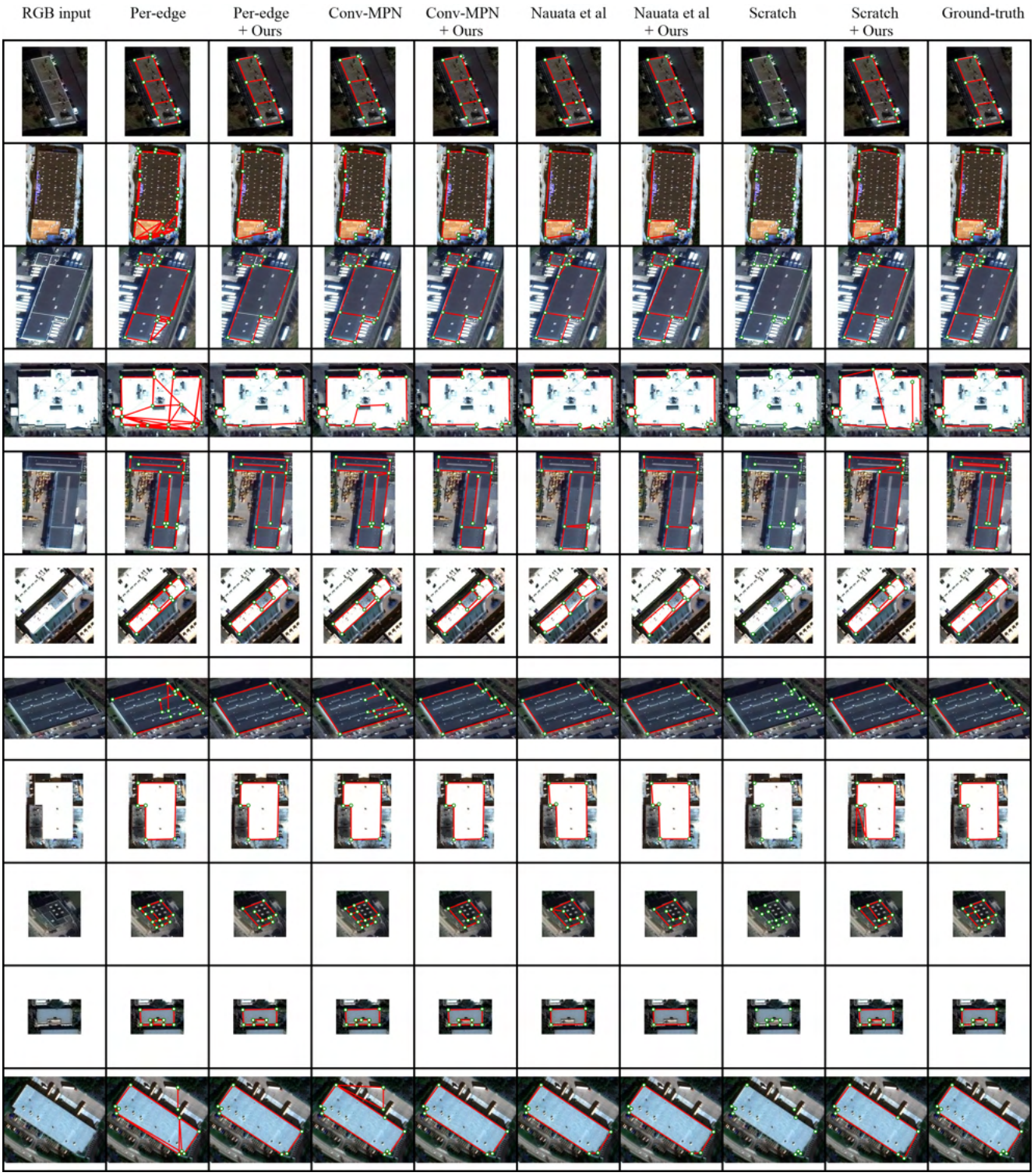

Figure 26. Additional qualitative results.

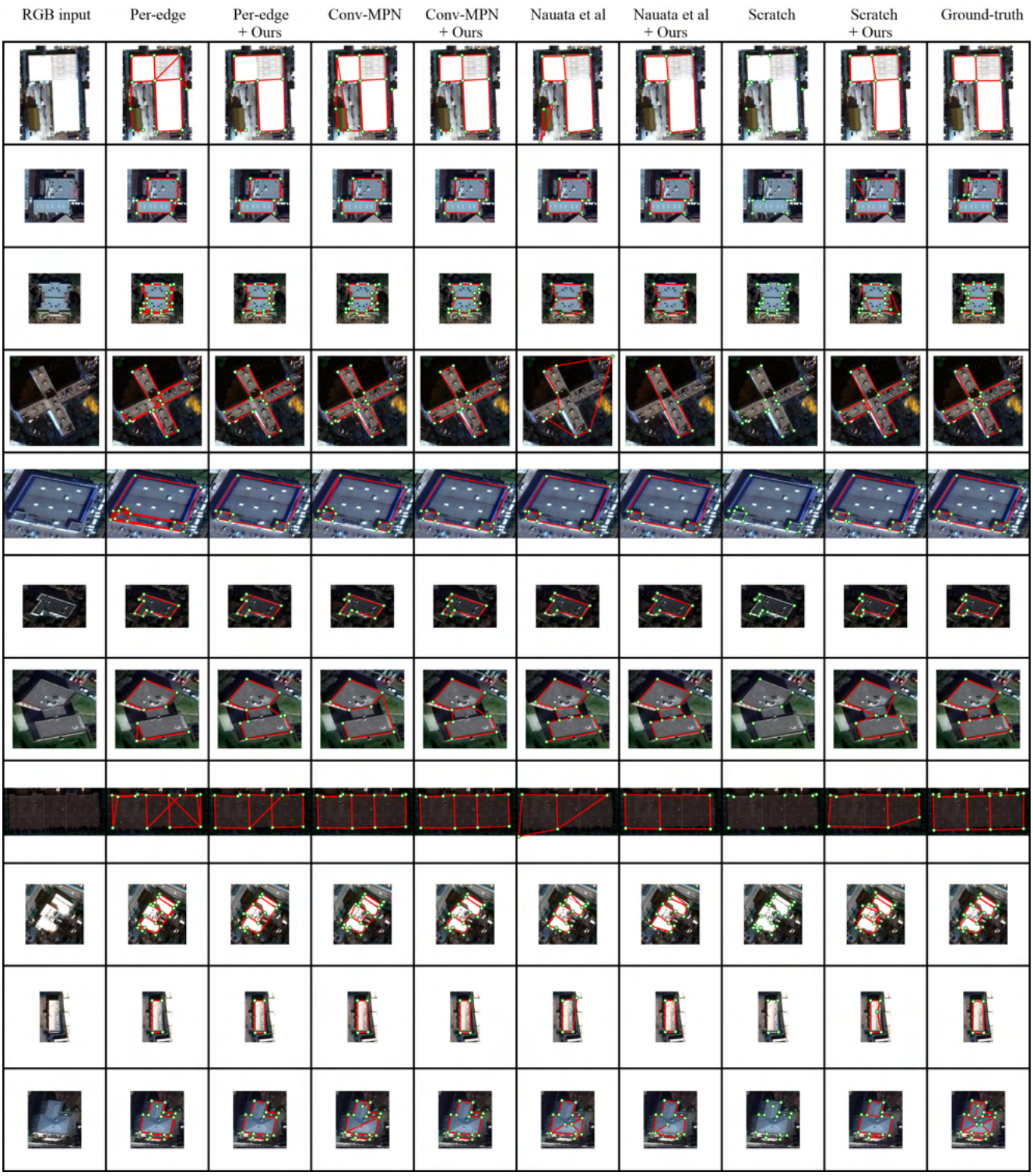

Figure 27. Additional qualitative results.

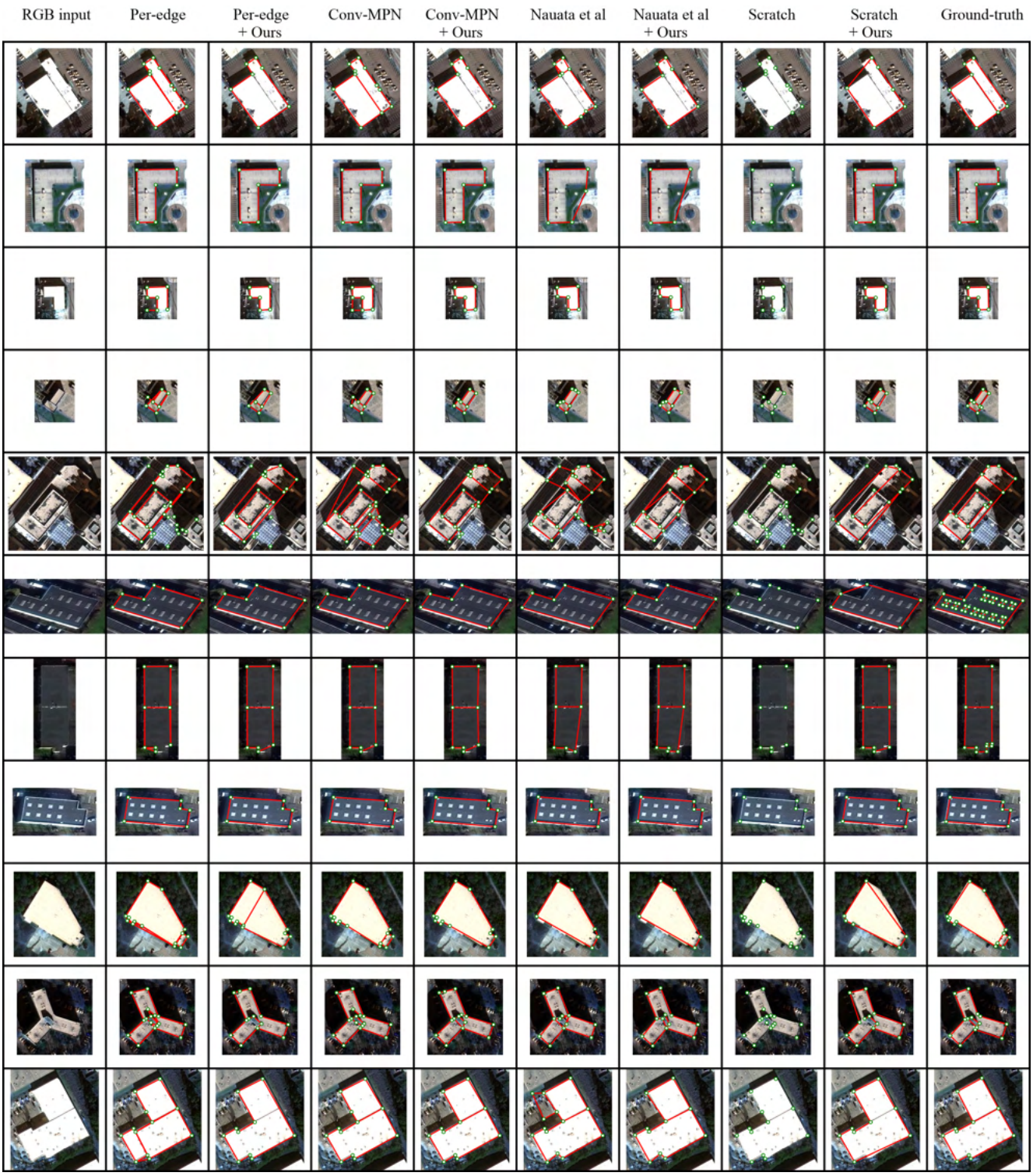

Figure 28. Additional qualitative results.

Figure 29. Additional qualitative results.

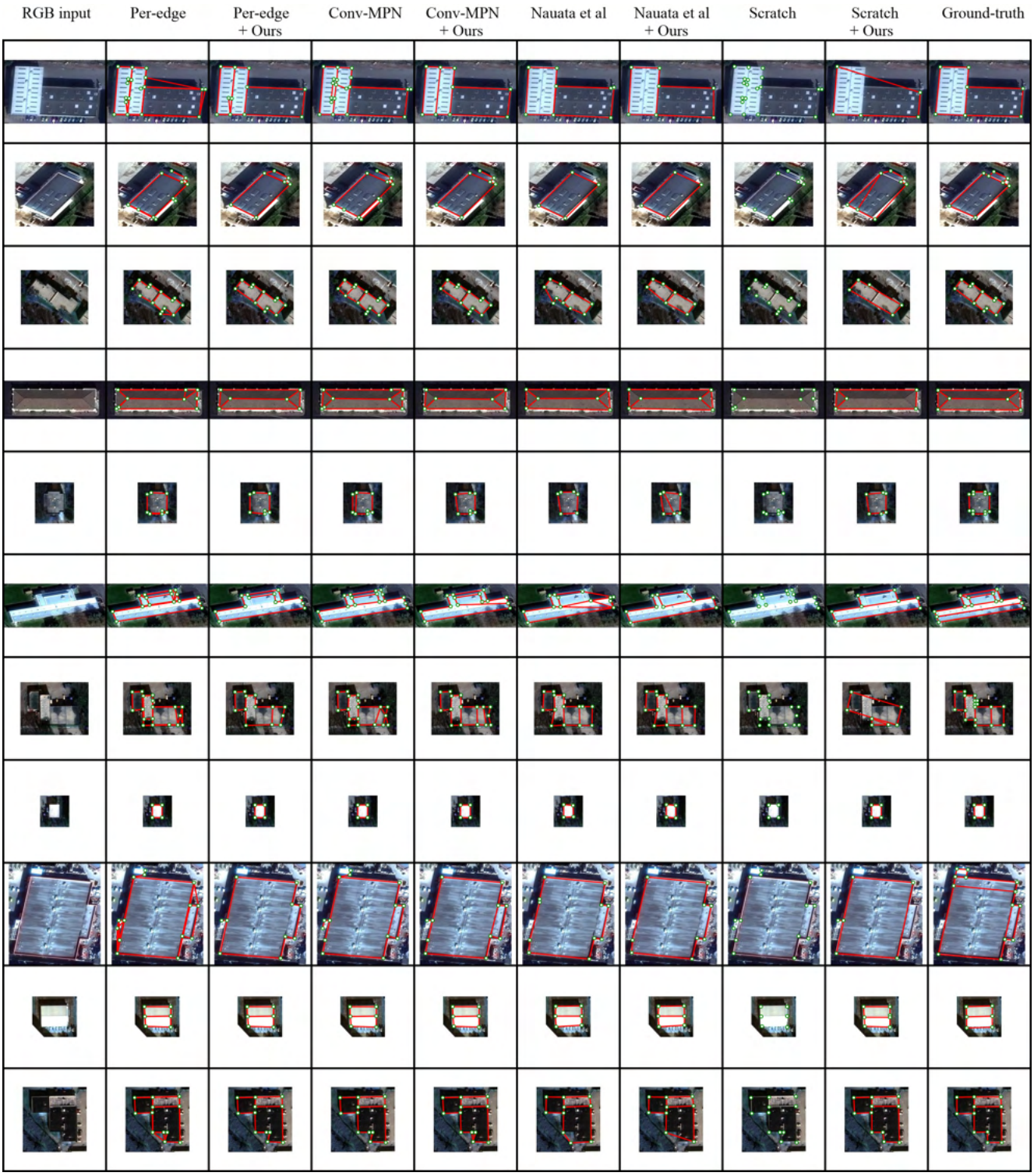

Figure 30. Additional qualitative results.

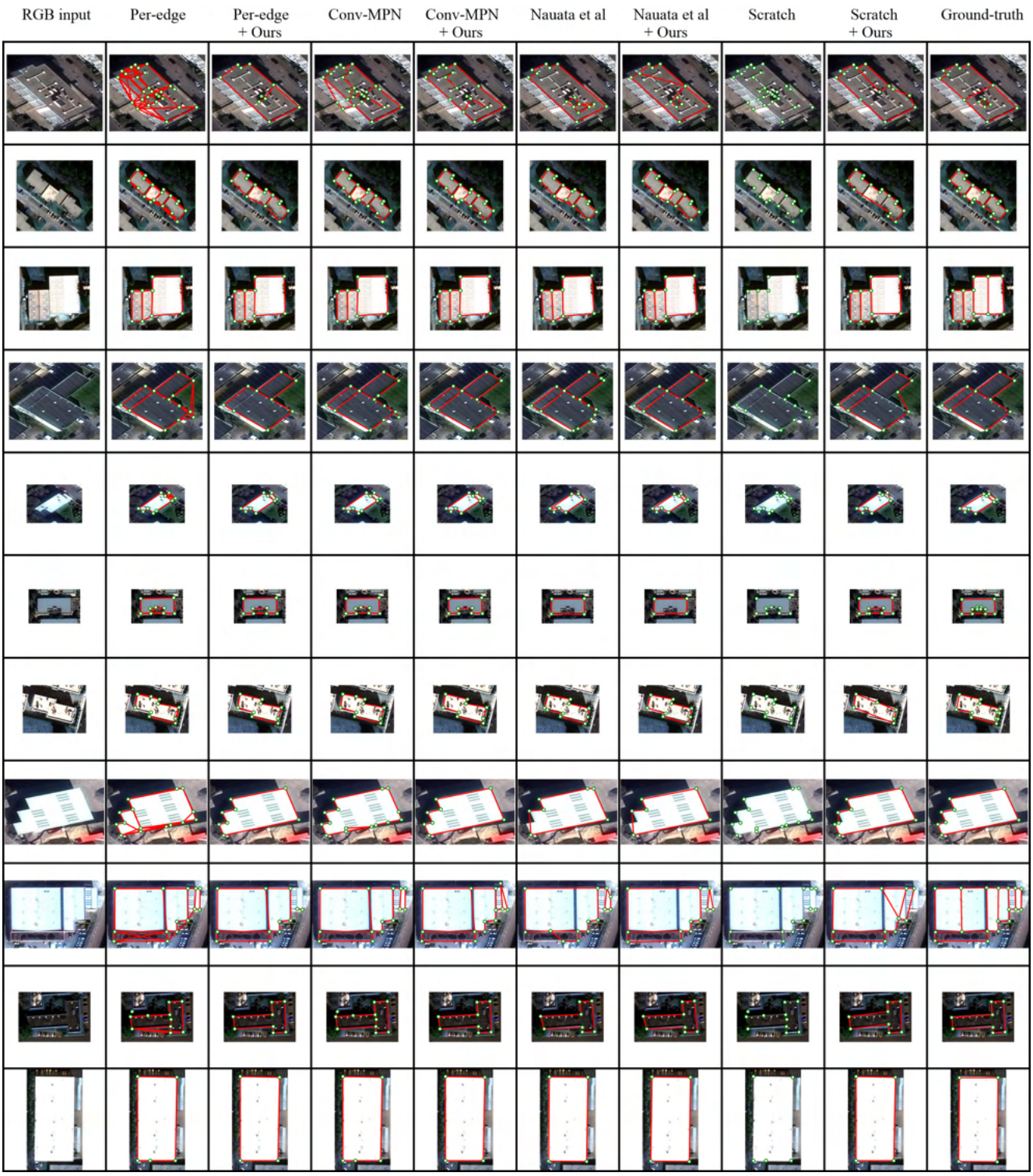

Figure 31. Additional qualitative results.

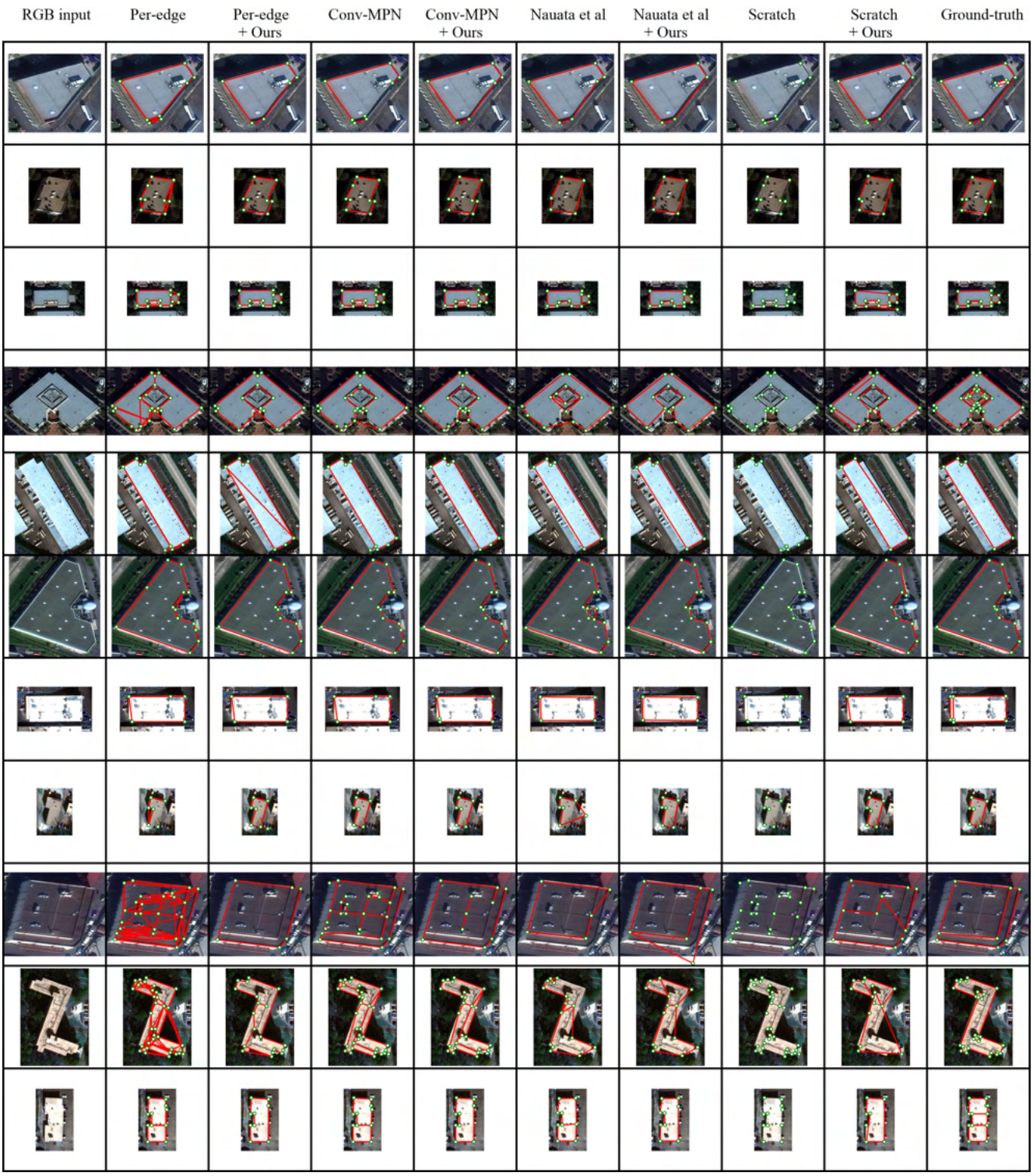

Figure 32. Additional qualitative results.

Figure 33. Additional qualitative results.

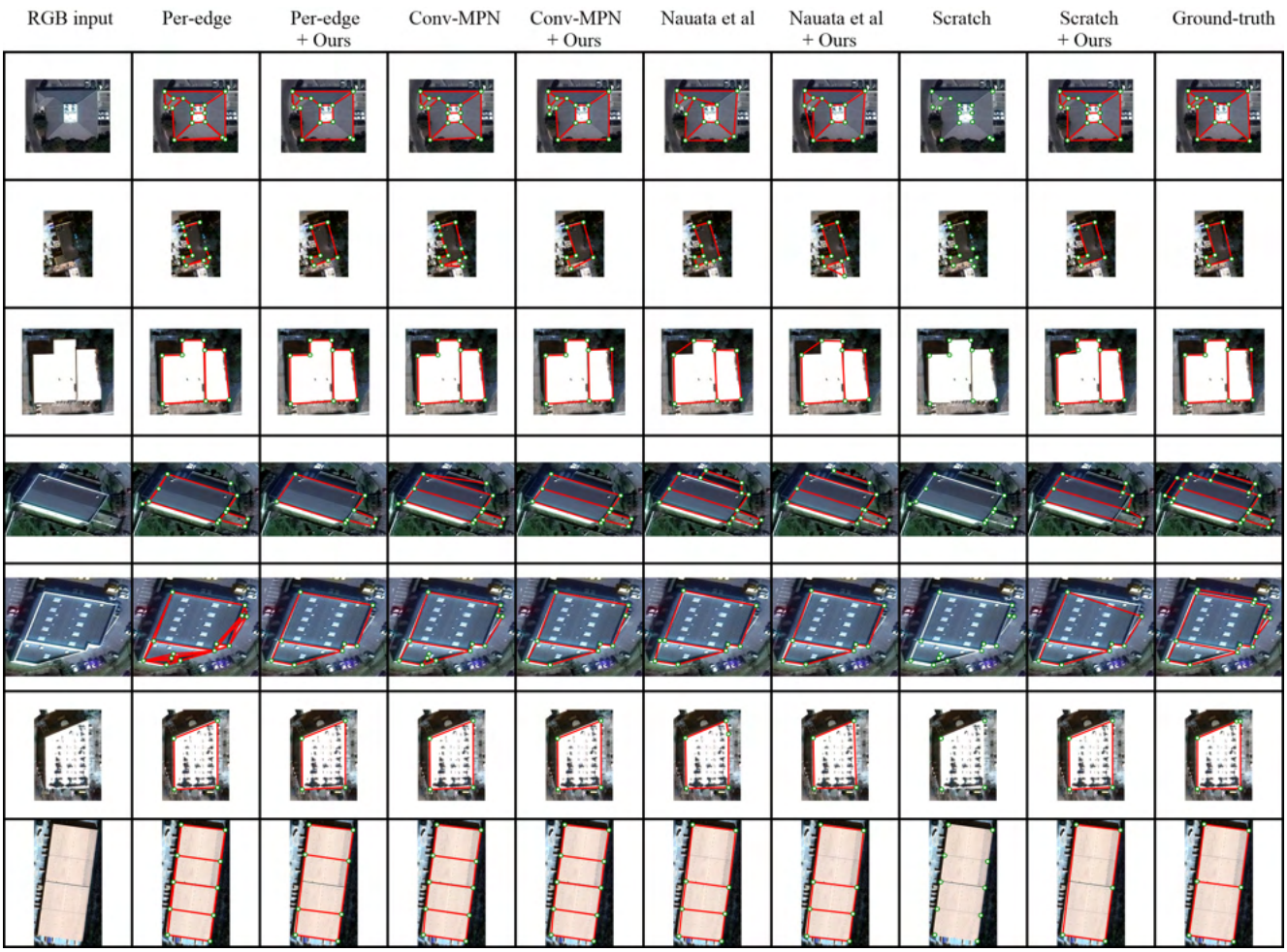

Figure 34. Additional qualitative results.

References

[1] Kevin Ellis, Maxwell Nye, Yewen Pu, Felix Sosa, Josh Tenenbaum, and Armando Solar-Lezama. Write, execute, assess: Program synthesis with a repl. *arXiv preprint arXiv:1906.04604*, 2019. 1

[2] Cheng Lin, Tingxiang Fan, Wenping Wang, and Matthias Nießner. Modeling 3d shapes by reinforcement learning. In *European Conference on Computer Vision*, pages 545–561. Springer, 2020. 1

[3] Fuyang Zhang, Nelson Nauata, and Yasutaka Furukawa. Conv-mpn: Convolutional message passing neural network for structured outdoor architecture reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2798–2807, 2020. 1