A-Scan2BIM: Assistive Scan to Building Information Modeling

Weilian Song[1]
weilians@sfu.ca
Jieliang Luo[2]
rodger.luo@autodesk.com
Dale Zhao[2]
dale.zhao@autodesk.com
Yan Fu[2]
fuyan82@gmail.com
Chin-Yi Cheng[3,†]
cchinyi@google.com
Yasutaka Furukawa[1]
furukawa@sfu.ca

[1] Simon Fraser University
[2] Autodesk Research
[3] Google Research

Abstract

This paper proposes an assistive system for architects that converts a large-scale point cloud into a standardized digital representation of a building for Building Information Modeling (BIM) applications. The process is known as Scan-to-BIM, which requires many hours of manual work even for a single building floor by a professional architect. Given its challenging nature, the paper focuses on helping architects on the Scan-to-BIM process, instead of replacing them. Concretely, we propose an assistive Scan-to-BIM system that takes the raw sensor data and edit history (including the current BIM model), then auto-regressively predicts a sequence of model editing operations as APIs of a professional BIM software (i.e., Autodesk Revit). The paper also presents the first building-scale Scan2BIM dataset that contains a sequence of model editing operations as the APIs of Autodesk Revit. The dataset contains 89 hours of Scan2BIM modeling processes by professional architects over 16 scenes, spanning over $35,000\,m^2$. We report our system's reconstruction quality with standard metrics, and we introduce a novel metric that measures how "natural" the order of reconstructed operations is. A simple modification to the reconstruction module helps improve performance, and our method is far superior to two other baselines in the order metric. We will release data, code, and models at a-scan2bim.github.io.

1 Introduction

Building Information Modeling (BIM) serves as a modern foundation for building design, construction, and management. This comprehensive approach involves generating a complete digital representation of a building, integrating various engineering disciplines such as

[†] Work done while at Autodesk AI Lab

architecture, electrical, HVAC, and more. BIM transcends traditional CAD modeling by incorporating not only geometric data but also essential constraints and metadata. The process of creating a BIM model from a 3D scan of an existing building is referred to as Scan-to-BIM. Leveraging a BIM model significantly simplifies cost-saving assessments, such as heating optimization and structural analysis. Furthermore, it streamlines renovations by providing a consistent, underlying model for designers across all industries. Swiftly generating an architectural model through Scan-to-BIM serves as the initial step for diverse industries like Civil, Construction, MEP (Mechanical, Electrical, and Plumbing), among others.

Scan-to-BIM is a labor-intensive process that often demands numerous hours of manual work from a professional architect, even for a single building floor. There exists commercial and open-source software for automatic Scan-to-BIM, but in general architects do not use these software due to poor performance and integration with their workflows. Recognizing this challenge, the paper aims to assist architects in the Scan-to-BIM process rather than replace them entirely. Nonetheless, existing building reconstruction algorithms struggle with this assistive task, as their reconstruction approach differs significantly from that of an architect. Specifically, architects create a BIM model by executing a series of modeling editing operations using CAD software, while current algorithms [7, 8, 15, 24] typically reconstruct a model in a single step or sequentially but without enabling human-interactions.

This paper proposes an assistive Scan-to-BIM system that takes raw sensor data and edit history (including the current BIM model), then auto-regressively predicts a sequence of model editing operations as APIs for professional BIM software, specifically Autodesk Revit. The sequence is presented to the user in the Revit interface, who can either accept or reject the suggestions for other options. We focus on wall reconstruction, one of the main steps of Scan-to-BIM workflow taking up 80% of the modeling steps based on our data collection process with architects. Concretely, the system operates in two stages. First, we use a modified version of a state-of-the-art floorplan reconstruction system [8] to enumerate candidate walls. The modification is to estimate wall thickness and scale to building-scale scans. Secondly, an auto-regressive transformer network processes a set of candidate walls and the addition action history (with order information) to predict a future sequence of actions. The transformer learns a feature embedding for a candidate wall with a contrastive loss, such that the next action is closer to the latest one in the feature space. The wall addition action activates a corresponding wall addition API via a Revit Plugin, where nearby wall segments are automatically joined and elevated to 3D. These segments can also be interactively modified within Revit. We have collected the first building-scale Scan2BIM dataset, comprising 89 hours of modeling processes by architects across 16 scenes over 35,000 square meters.

We have included a supplementary video showcasing our system assisting a user in a real modeling scenario. We also conduct qualitative and quantitative evaluations of the proposed system against several baselines using 1-fold cross-validation. In addition to standard metrics for structured reconstruction [8], the paper assesses the "naturalness" of the reconstruction order by introducing a new metric inspired by the FID score, a standard metric for generative models [17]. Our experiments demonstrate that the proposed system more closely resembles approaches by architects than the baselines. In summary, the paper's contributions are threefold: 1) A transformer network with contrastive loss training that predicts a natural sequence of actions; 2) A Scan2BIM assistive system that directly drives professional CAD software throughout the BIM reconstruction process; and 3) A building-scale Scan2BIM dataset containing 89 hours of BIM modeling sequences by professional architects. To further promote the development Scan-to-BIM techniques, we will release all the data, models, and code.

2 Related works

Our paper introduces a new dataset of large-scale buildings, and proposes an interactive algorithm for structured reconstruction. We review datasets and methods related to architectural structured reconstruction, along with the literature of Scan-to-BIM.

Architectural structured reconstruction datasets have been introduced for modeling at the scene-level, floor-level, and city-level. Scene-level primarily focuses on semantic segmentation [1, 6, 9, 20], plane reconstruction [6, 9], and wire-frame parsing [11], while floor-level [12, 14, 15, 16] aims to obtain architectural structures such as walls and columns. HoliCity is a city-level dataset containing 3D structures of buildings along with street-level panoramic images and segmentation masks. In terms of data type, the floorplan dataset used by HEAT [8] is closely related to our dataset, as both the input is a point cloud and the output is a planar graph of edges. However, our floors are significantly larger and have more complex structures with walls of varying thickness. In terms of floor scale, S3D proposed by Armeni *et al.* [1] more closely resembles our dataset, as both consist of large office spaces. Our labels contain significantly more detailed geometry, as our dataset represents architectural models. Additionally, our dataset includes 16 floors compared to 5, covering over 35 000 square meters in comparison to 6, 000 square meters in their dataset.

Architectural structured reconstruction methods commonly use input data such as images, RGB-D scans, or 3D point clouds, and output the man-made structure in a planar graph representation. A two-stage pipeline has been the dominant approach to recover building structures, where geometry primitives such as corners are first detected, followed by their assembly process. Ikehata *et al.* [12], Liu *et al.* [15], and Chen *et al.* [7] employed optimization systems for the second stage, using predefined grammar, integer programming, or energy minimization techniques. These methods typically make strong assumptions about the structure, such as a Manhattan layout. Conv-MPN [24] improved upon previous works by proposing a fully-neural architecture, allowing for learnable topology inference. MonteFloor [21] incorporated Monte Carlo Tree Search while remaining fully differentiable and removing assumptions in earlier optimization-based methods. Finally, HEAT [8] achieved state-of-the-art outdoor and indoor reconstruction results through a transformer architecture [22], which learns to classify edge proposals by considering edge image features and global topology. 3D wireframe techniques [11, 23, 25, 26] often do not rely on primitives and directly predict the target geometries from learnable embeddings.

Scan-to-BIM There exists automatic Scan-to-BIM systems in the Civil Engineering community [13, 18, 19]. They often employ multi-step optimization systems such as semantic segmentation followed by plane fitting and wall reconstruction. To our knowledge, they are not evaluated on standard datasets and no code exists for comparison. Exceptions are with works from Bassier *et al.* [4, 5], where automatic systems are proposed and evaluated on the S3DIS [1]. Furthermore, these papers focus on system integration instead of technical contribution, which would be more critical for the computer vision community.

3 Assistive Scan-to-BIM dataset

We borrow building-scale scans from the “Computer Vision in the Built Environment” workshop series at CVPR [2]. We have used 16 scans from 11 buildings, with space types including office spaces, parking lots, medical offices, and laboratories. Point clouds are captured

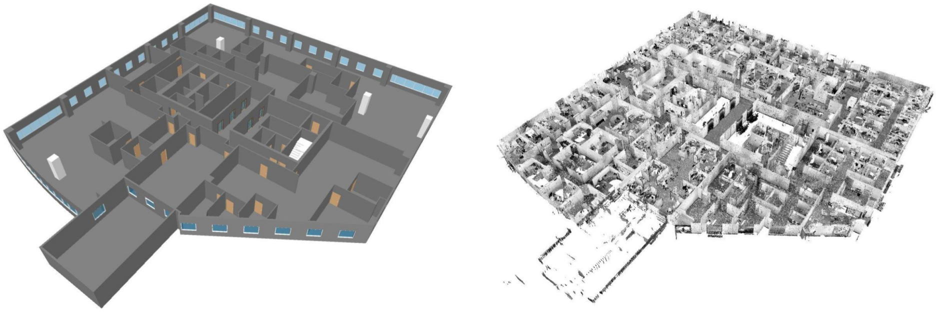


Figure 1: A BIM model and a scan from one floor with the ceiling removed. Walls, doors, windows, and columns account for 84.2%, 1.1%, 14%, and 0.6% of the modeling steps.

Table 1: Statistics of our proposed dataset. Average refers to average per floor.

General stats			Element counts			Element type counts		
	Total	Average		Total	Average		Total	Average
Floor size (m^2)	35528	2221	Walls	3142	196	Walls	189	12
Annotation time (hrs)	89	6	Doors	808	51	Doors	297	19
Modeling steps	45306	2832	Windows	3344	209	Windows	77	5
JSON data size (GB)	29.97	1.87	Columns	323	20	Columns	113	7

using professional surveying equipment, which is accurate to 1 cm. To obtain ground-truth BIM models, we hired 5 professional architects and asked them to model at Level of Development (LOD) 200, which includes floors, walls, doors, windows, columns, and stairs. Autodesk Revit software was used for modeling, which will be our target platform.

The main goal is to record modeling sequences, while Revit does not provide this functionality. We created a custom Revit plugin that saves the states of the BIM model before and after an operation, and programmatically translates the state difference into an equivalent operation as a Revit API call. Please see the supplementary for more details.

Table 1 shows the statistics of our dataset. Figure 1 visualizes a BIM model and a scan. Walls, doors, windows, and columns account for 84.2%, 1.1%, 14%, and 0.6% of the operations, respectively. A 3-channel top-down point density image is an input to our system: We slice the point clouds horizontally at 6.56, 8.2, and 12 feet above the ground, and calculate the point density within the three slices in the top-down view at one inch resolution.

4 Assistive Scan-to-BIM

Scan-to-BIM turns sensor data into a BIM model, where we focus on wall structures. As a BIM assistant, the proposed system is integrated with a BIM software, in our case Autodesk Revit. The section explains 1) The pre-processing module, in which we enumerate wall segment candidates by modifying the current state-of-the-art floorplan reconstruction system [8] to handle building-scale scans; 2) The next wall prediction module, which is the core of our system; and 3) The assistive Scan-to-BIM system that integrates the two modules with Revit. We refer to Figure 2 for an overview of our system.

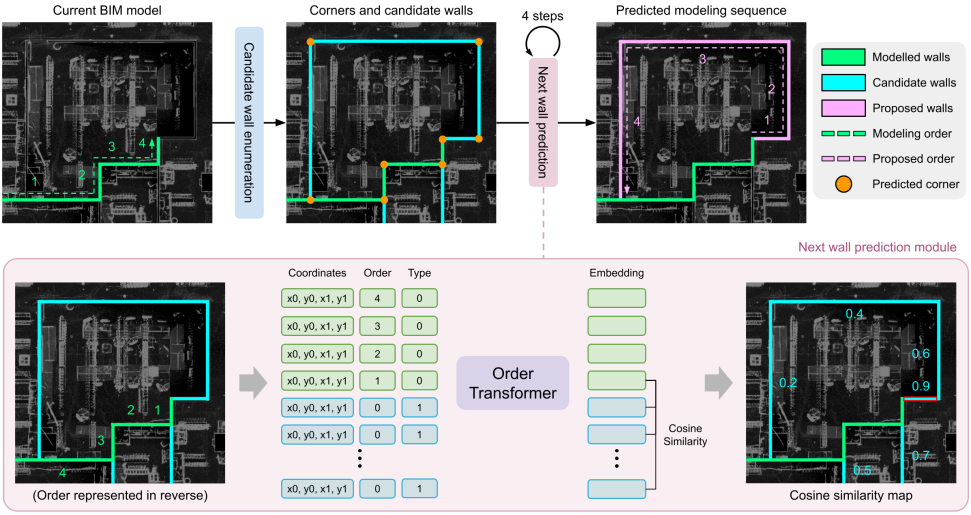


Figure 2: System overview. Given an existing BIM model as the current state, we first obtain wall candidates by enumerating corners and edges with thickness. We then auto-regressively predict an ordering to the candidate walls using a Transformer network.

4.1 Candidate wall enumeration

Given a density image and existing walls, we use the state-of-the-art floorplan reconstruction system HEAT [8] with a few key modifications to enumerate wall candidates in four steps: corner enumeration, wall enumeration, wall thickness prediction, and duplicate removal.

Corner enumeration is done by a HEAT corner detection module that uses a Transformer network to estimate the corner likelihood at every pixel and apply non-maximum suppression (NMS). Since our density image is large, we divide the image into 256 × 256 local windows with 64 pixels overlap, compute the pixel-wise likelihood by the HEAT module, and merge results while keeping the maximum value at the overlaps. The same NMS filter applies.

Wall enumeration is done by a HEAT edge classification module, where we modify the deformable attention layer to pool image features along very long edges in our task. Concretely, instead of sampling image features from one reference point (typically the edge center), we sample from N linear-interpolated reference points along an edge, and apply max-pooling. Lastly, Revit represents a T-junction as one long edge and one short edge. During training, we split the long edge into two at the junction point to obtain a proper planar graph structure.

Wall thickness prediction is obtained by an extra two-layer MLP which is added to the end of the image branch of the edge classifier above. The thickness can range from 1 inch to 84 inches with an increment of an inch. During training, we apply cross-entropy loss on all walls with known thickness while ignoring the rest.

Duplicate removal looks through the enumerated walls and retains only ones that do not exist yet. Given the coordinates of a pair of enumerated and existing walls, we compute the distances of their end-points. An enumerated wall is flagged as a duplicate and removed, if both corners are less than 10 pixels for at least one existing wall. We also perform the same filter between the enumerated walls themselves and remove duplicates.

4.2 Next wall prediction

Given an existing set of walls optionally with their edit history (i.e., the sequence in which they were added to the model) along with a set of wall candidates, we calculate a score for each candidate. The score estimates the likelihood of each candidate wall being the next addition to the model. The process runs auto-regressively to obtain future modeling sequences of arbitrary length. Please see the lower half of Figure 2 for a high-level overview.

A Transformer network is our architecture where an existing or a candidate wall is a node. The network contains six blocks of self-attention layers, and produces a 256-dimensional embedding vector for each node. The cosine similarity in the embedding space between the last existing wall to candidate walls determines the scores of the next one to be added (i.e., higher the more likely). We refer to the supplementary for the full architecture specifications.

Each wall node is a concatenation of three embeddings: 1) 256-dimensional sinusoidal embedding of the wall coordinates; 2) one of three learnable 128-dimensional wall type embeddings; and 3) 128-dimensional sinusoidal embedding of the timestep t when the wall was added. The timestamp (t) is assigned in reverse chronological order. Specifically, candidate walls are marked with $t = 0$. The last added wall is assigned $t = 1$. The second last added wall is given $t = 2$, and so on. For walls that have not been modified in the last 10 steps, we designate their timestamp as $t = 10$. For type embeddings, the three types of walls are ones with timestep $t = 0$, $1 \leq t < 10$, and $t \geq 10$, respectively. The concatenated embedding is then projected down to the dimension of 256 by a linear layer.

We train the Transformer network with a contrastive loss by constructing triplets with the last wall ($t = 1$), the ground-truth (GT) next wall, and all other candidate walls, such that the cosine embedding distance between the latest and GT next wall are closer than the rest of the candidates by a margin of 1. We train on GT edges and sequences, and found that the network also works well on our reconstruction results during test time.

4.3 Assistive Scan-to-BIM

The final assistive system combines the above modules and works with or without user interaction, from scratch or an existing BIM model. Regardless of the scenario, we first run the corner enumeration module and cache the result to be used in all subsequent steps.

Automatic mode reconstructs the BIM model without user-interaction by auto-regressively running the next wall prediction network, while taking the candidate wall with the highest score every time. Note that when starting from an existing model, we do not have the edit history information and set $t = 10$ for all existing walls.

Assistive mode aims to speed up manual modeling by offering wall auto-completions after each user interaction. A user interaction can be either a corner addition or wall addition/modification action, the former of which is a new action implemented by our plugin. Each interaction introduces new or modified corners, and we combine them with our cached corners and re-run the wall enumeration module to update the wall candidates. The system then auto-regressively predicts a future modeling sequence of length-N (N changeable by a user), and displays the sequence as special lines in Revit. The user can choose to accept the proposal, or instead change the modeling direction by requesting the top-3 next wall predictions and choosing one as the next step. Auto-completion then resumes. Please see the supplementary video for the demonstration of the mode.

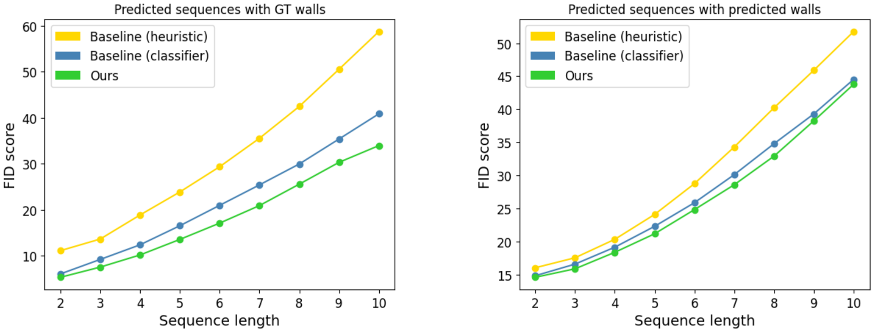


Figure 3: Main results, evaluating modeling sequence generated by different methods at different sequence lengths. Lower is better.

5 Experiments

Our system has been developed in Python 3.8 and PyTorch 1.12.1. The training process utilizes a single NVIDIA A100 GPU with 40GB of memory. For the hyper-parameters of the corner and edge networks, we refer to the HEAT model [8]. The learning rate for all networks is 2×10^{-4}. The batch sizes are 8, 1, and 128 for the corner, edge, and next wall prediction networks, respectively. For the edge network, we perform gradient accumulation every 16 steps, effectively yielding a batch size of 16. The training takes $93,000$, $210,000$, and $200,000$ steps for the three networks, with the learning rate decaying by a factor of 0.1 every $53,000$, $70,000$, and $100,000$ steps, respectively. For data augmentation, we use random rotation for corner training. For wall enumeration, we normalize the image and edges so that the longest edge does not exceed 1000 pixels, after which we apply random rotation and scaling. For the next wall prediction, we first center and normalize the edges by a maximum length of 1000, then perform random translation, rotation, and scaling. After fine-tuning the corner detector of HEAT, we obtain corner precision/recall of 83.70/71.10% under matching threshold of 30 inches.

5.1 Baseline methods

Heuristic: To determine wall addition sequence, one can greedily pick the candidate wall nearest to the last edit. Specifically, for each pair of walls, we identify the closest points and calculate their distance. We apply this process from the last wall in the sequence to all candidate walls, choosing the candidate wall with the smallest distance.

Classifier: In a more straightforward approach, one could train a classifier to distinguish between valid and invalid sequences. This classifier could enumerate all potential sequence candidates, selecting the one with the highest probability. To train such a classifier, we construct positive examples by considering all sub-sequences of the ground truth modeling steps, starting from the first step. For every positive sequence, we generate a negative example by substituting the last wall with one from a future step. To overcome class imbalance issue, we replicate positive examples to that of the negative ones. In terms of binary classification, we utilize the same architectural framework as the next wall prediction module. However, we introduce an extra classification token into the transformer network. This token undertakes

Table 2: Effect of number of reference points for deformable attention

	5 inches			15 inches			30 inches				Width
# points	Prec.	Recall	F-1	Prec.	Recall	F-1	Prec.	Recall	F-1	IoU	Acc.
1	59.29	41.01	48.09	67.1	45.95	54.09	70.23	48.03	56.57	0.3	79.56
8	59.42	43.51	49.6	66.82	48.41	55.4	69.53	50.45	57.7	**0.34**	78.85
16	**59.77**	**43.63**	**49.88**	**67.24**	**48.55**	**55.73**	**70.42**	**50.9**	**58.41**	**0.34**	**80.16**

Table 3: Entropy and accuracy vs history length

Hist. len	1	2	3	4	5	6	7	8	9
Accuracy	25.17	27.41	28.48	29.45	30.21	30.93	31.72	32.35	32.72
Entropy	2.32	2.26	2.19	2.15	2.13	2.09	2.05	2.02	2.01

self-attention with the walls, and a final linear layer is used to output the binary probability.

5.2 Novel order metric

Evaluating next wall prediction solely based on accuracy is overly stringent, given the ambiguous nature of the task. We propose a new metric to assess the "naturalness" of the predicted wall modeling sequence. This metric draws inspiration from the Fréchet Inception Distance (FID) score [10], a measure commonly used to evaluate the quality of samples produced by image generation models. Given two sets of real and predicted wall sequences, we initially generate a latent encoding for each sequence using a Temporal Convolutional Network (TCN) [3]. The TCN is trained to auto-encode random subsets of ground truth edges, with the output from the 6$^{\text{th}}$ hidden layer being used. For each set of encodings, we calculate two Gaussian distributions that capture the mean and variance of each latent dimension. The final score is determined by computing the Fréchet distance between these two distributions. A lower score indicates a better model. For more detailed information regarding the TCN architecture and its training process, please refer to the supplementary materials.

5.3 Quantitative evaluations

Figure 3 presents our main results, wherein we assess the sequences predicted by three competitive methods using our proposed order metric. To generate a sequence, we select one edge from the candidates as the starting point, and subsequently predict the next N steps autoregressively, up to a maximum of 10 steps. The candidate walls can be either ground truth (GT) or predicted ones. For the latter, we use raw predictions without any post-processing to maintain high recall. As evident, our method outperforms the others on both sets of walls, despite being trained solely on GT walls and sequences.

We evaluate the impact of our modifications to the deformable attention module (Sect. 4.1). As shown in Table 2, we vary the number of reference points and calculate precision/recall at different distance thresholds along with Intersection over Union (IoU) scores. To compute wall width accuracy, we collect matched walls under threshold of 30 inches, and consider width prediction to be correct if it's within 3 inches. The method of sampling only one point is equivalent to the original HEAT architecture. The results indicate that utilizing more points generally leads to superior performance.

Lastly, we investigate the effect of history length on the next wall prediction. More accurate and confident predictions should result from providing a longer historical context.

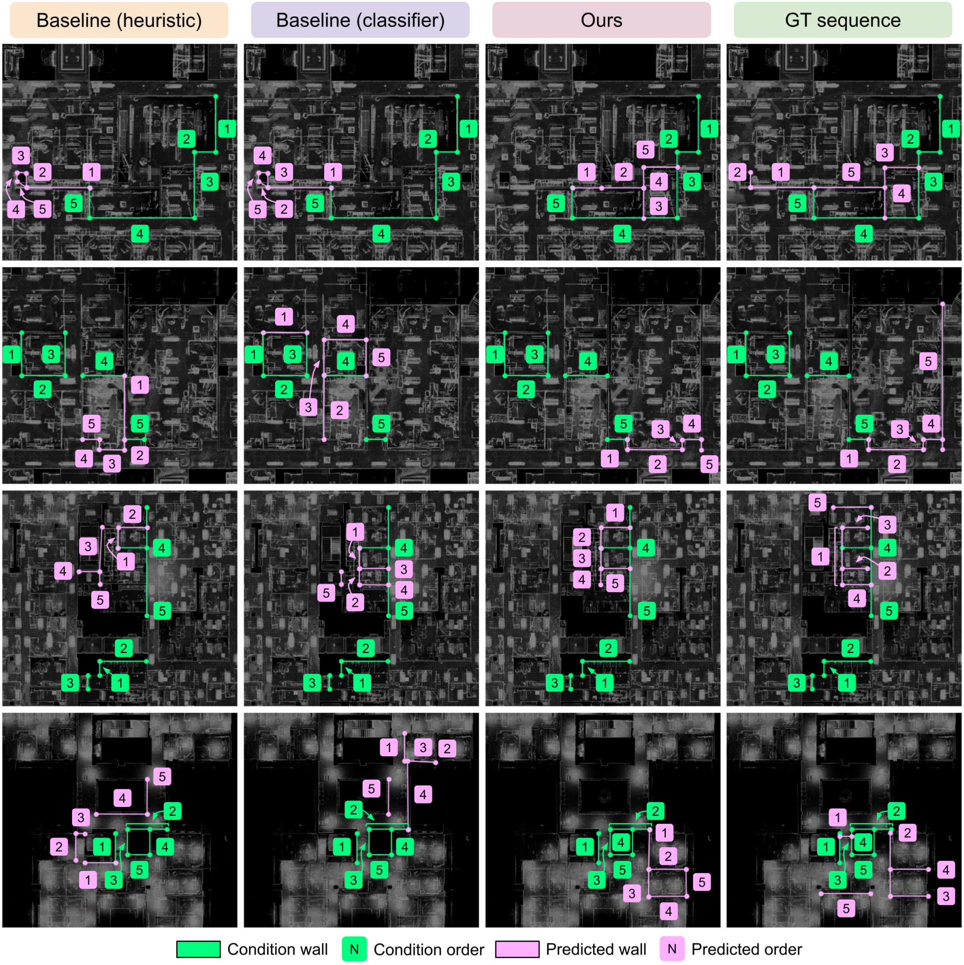


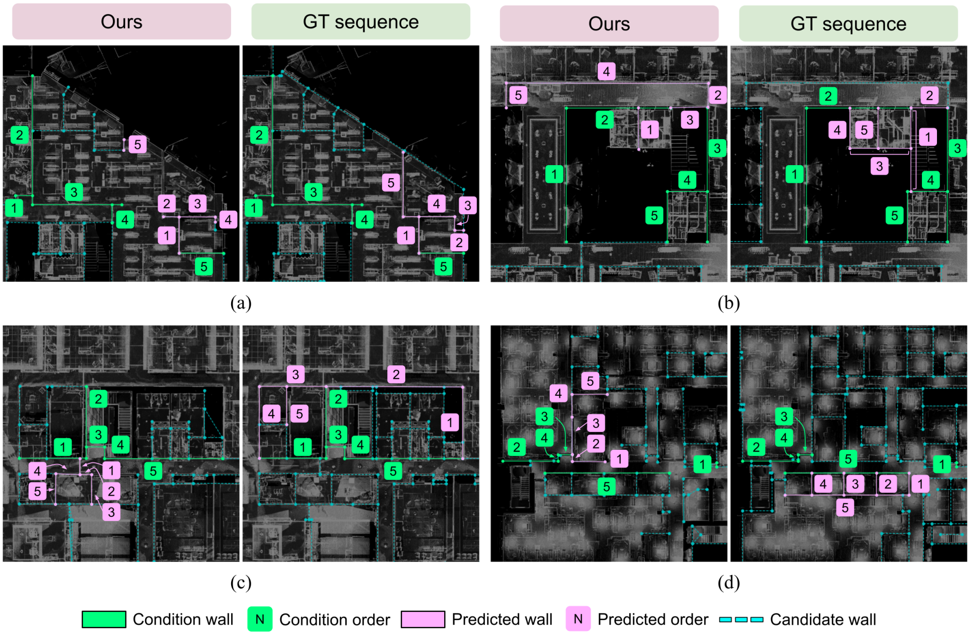
Figure 4: Qualitative comparison of predicted sequences. We note that for the right-most column, the pink walls and ordering are GT as well.

To do so, we provide GT history of different lengths, and predict from the remaining GT edges. In Table 3, we observe that as the history length increases, both the accuracy of the predicted next wall and the entropy decrease, thereby confirming our hypothesis.

5.4 Qualitative evaluations

Our system is designed to be interactive. We refer to the supplementary video for a demonstration of our system. The video depicts a real-world modeling scenario in which our system is integrated within Revit and supports the user by predicting future modeling sequences. Both automatic and assistive modes are demonstrated to highlight the flexibility of our tool.

Figure 4 illustrates some of the wall sequence predictions, where a ground truth (GT) sequence is a condition. Across all the columns, the green edges represent GT walls. In the first three columns, the pink edges correspond to the reconstructed results; for the rightmost



(a)

(b)

(c)

(d)

Figure 5: Failure cases of our method. (a) and (b) suffers from poor reconstruction results; (c) and (d) demonstrates incorrect predicted order.

column, they signify GT walls. For ease of visualization, the history and predicted lengths are capped at five. Our method yields a sequence ordering that more closely mirrors the GT sequence. In the first row, our method intuitively grasps the intention of an architect, choosing to model the smaller enclosed area first. In the second row, our sequence ordering closely matches the GT, with the only deviations occurring in the reconstruction results.

6 Limitations and conclusion

In this paper, we have introduced a neural network architecture for the prediction of natural sequences, a Scan2BIM assistive system seamlessly integrated with professional CAD software, and an extensive Scan2BIM dataset. Our method showcases strong performance in reconstruction tasks and surpasses two baseline models in next-wall prediction. However, our method still exhibits failure cases with reconstructions and order predictions, which need to be seriously considered before real-world usage (see Figure 5 for examples). With an understanding that there is room for further development, we pledge to continue this advancement by making all relevant data, models, and code accessible to the research community.

Acknowledgment This research is partially supported by NSERC Discovery Grants with Accelerator Supplements and the DND/NSERC Discovery Grant Supplement, NSERC Alliance Grants, and the John R. Evans Leaders Fund (JELF). This research was enabled in part by support provided by BC DRI Group and the Digital Research Alliance of Canada (alliancecan.ca). And finally, we want to thank all the architects who helped debug our Revit plugins and participated in the data collection effort.

References

[1] Iro Armeni, Ozan Sener, Amir R. Zamir, Helen Jiang, Ioannis Brilakis, Martin Fischer, and Silvio Savarese. 3D Semantic Parsing of Large-Scale Indoor Spaces. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1534–1543. IEEE. ISBN 978-1-4673-8851-1. doi: 10.1109/CVPR.2016.170. URL http://ieeexplore.ieee.org/document/7780539/.

[2] Iro Armeni, Erzhuo Che, Martin Fischer, Yasutaka Furukawa, Daniel Hall, Jaehoon Jung, Fuxin Li, Michael Olsen, Marc Polleyfeys, and Yelda Turkan. Computer vision in the built environment. https://cv4aec.github.io/, 2023.

[3] Shaojie Bai, J. Zico Kolter, and Vladlen Koltun. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv:1803.01271*, 2018.

[4] Maarten Bassier and Maarten Vergauwen. Topology Reconstruction of BIM Wall Objects from Point Cloud Data. 12(11):1800, . ISSN 2072-4292. doi: 10.3390/rs12111800. URL https://www.mdpi.com/2072-4292/12/11/1800.

[5] Maarten Bassier and Maarten Vergauwen. Unsupervised reconstruction of Building Information Modeling wall objects from point cloud data. 120:103338, . ISSN 0926-5805. doi: 10.1016/j.autcon.2020.103338. URL https://www.sciencedirect.com/science/article/pii/S0926580520309183.

[6] Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niebner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. Matterport3D: Learning from RGB-D Data in Indoor Environments. In *2017 International Conference on 3D Vision (3DV)*, pages 667–676. doi: 10.1109/3DV.2017.00081.

[7] Jiacheng Chen, Chen Liu, Jiaye Wu, and Yasutaka Furukawa. Floor-SP: Inverse CAD for Floorplans by Sequential Room-Wise Shortest Path. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 2661–2670. IEEE, . ISBN 978-1-72814-803-8. doi: 10.1109/ICCV.2019.00275. URL https://ieeexplore.ieee.org/document/9008810/.

[8] Jiacheng Chen, Yiming Qian, and Yasutaka Furukawa. HEAT: Holistic Edge Attention Transformer for Structured Reconstruction. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3856–3865. IEEE, . ISBN 978-1-66546-946-3. doi: 10.1109/CVPR52688.2022.00384. URL https://ieeexplore.ieee.org/document/9878511/.

[9] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Niessner. ScanNet: Richly-Annotated 3D Reconstructions of Indoor Scenes. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2432–2443. IEEE. ISBN 978-1-5386-0457-1. doi: 10.1109/CVPR.2017.261. URL https://ieeexplore.ieee.org/document/8099744/.

[10] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc. URL

https://papers.nips.cc/paper_files/paper/2017/hash/8a1d694707eb0fefe65871369074926d-Abstract.html.

[11] Kun Huang, Yifan Wang, Zihan Zhou, Tianjiao Ding, Shenghua Gao, and Yi Ma. Learning to Parse Wireframes in Images of Man-Made Environments. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 626–635. IEEE. ISBN 978-1-5386-6420-9. doi: 10.1109/CVPR.2018.00072. URL https://ieeexplore.ieee.org/document/8578170/.

[12] Satoshi Ikehata, Hang Yang, and Yasutaka Furukawa. Structured Indoor Modeling. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 1323–1331. IEEE. ISBN 978-1-4673-8391-2. doi: 10.1109/ICCV.2015.156. URL http://ieeexplore.ieee.org/document/7410513/.

[13] Jaehoon Jung, Cyrill Stachniss, Sungha Ju, and Joon Heo. Automated 3D volumetric reconstruction of multiple-room building interiors for as-built BIM. 38:811–825. ISSN 1474-0346. doi: 10.1016/j.aei.2018.10.007. URL https://www.sciencedirect.com/science/article/pii/S1474034618300600.

[14] Ahti Kalervo, Juha Ylioinas, Markus Häikiö, Antti Karhu, and Juho Kannala. CubiCasa5K: A Dataset and an Improved Multi-task Model for Floorplan Image Analysis. In Michael Felsberg, Per-Erik Forssén, Ida-Maria Sintorn, and Jonas Unger, editors, *Image Analysis*, Lecture Notes in Computer Science, pages 28–40. Springer International Publishing. ISBN 978-3-030-20205-7. doi: 10.1007/978-3-030-20205-7_3.

[15] Chen Liu, Jiajun Wu, Pushmeet Kohli, and Yasutaka Furukawa. Raster-to-Vector: Revisiting Floorplan Transformation. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2214–2222. doi: 10.1109/ICCV.2017.241.

[16] Nelson Nauata and Yasutaka Furukawa. Vectorizing World Buildings: Planar Graph Reconstruction by Primitive Detection and Relationship Inference. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision – ECCV 2020*, volume 12353, pages 711–726. Springer International Publishing. ISBN 978-3-030-58597-6 978-3-030-58598-3. doi: 10.1007/978-3-030-58598-3_42. URL https://link.springer.com/10.1007/978-3-030-58598-3_42.

[17] Nelson Nauata, Kai-Hung Chang, Chin-Yi Cheng, Greg Mori, and Yasutaka Furukawa. House-gan: Relational generative adversarial networks for graph-constrained house layout generation. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part I 16*, pages 162–177. Springer, 2020.

[18] Shayan Nikoohemat, Abdoulaye A. Diakité, Sisi Zlatanova, and George Vosselman. Indoor 3D reconstruction from point clouds for optimal routing in complex buildings to support disaster management. 113:103109. ISSN 0926-5805. doi: 10.1016/j.autcon.2020.103109. URL https://www.sciencedirect.com/science/article/pii/S0926580519306193.

[19] Sebastian Ochmann, Richard Vock, and Reinhard Klein. Automatic reconstruction of fully volumetric 3D building models from oriented point clouds. 151:251–262. ISSN 0924-2716. doi: 10.1016/j.isprsjprs.2019.03.017. URL https://www.sciencedirect.com/science/article/pii/S0924271619300863.

[20] Shuran Song, Fisher Yu, Andy Zeng, Angel X. Chang, Manolis Savva, and Thomas Funkhouser. Semantic Scene Completion from a Single Depth Image. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 190–198. IEEE. ISBN 978-1-5386-0457-1. doi: 10.1109/CVPR.2017.28. URL http://ieeexplore.ieee.org/document/8099511/.

[21] Sinisa Stekovic, Mahdi Rad, Friedrich Fraundorfer, and Vincent Lepetit. Monte-Floor: Extending MCTS for Reconstructing Accurate Large-Scale Floor Plans. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 16014–16023. IEEE. ISBN 978-1-66542-812-5. doi: 10.1109/ICCV48922.2021.01573. URL https://ieeexplore.ieee.org/document/9711164/.

[22] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is All you Need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc. URL https://proceedings.neurips.cc/paper_files/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html.

[23] Yifan Xu, Weijian Xu, David Cheung, and Zhuowen Tu. Line Segment Detection Using Transformers Without Edges. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

[24] Fuyang Zhang, Nelson Nauata, and Yasutaka Furukawa. Conv-MPN: Convolutional Message Passing Neural Network for Structured Outdoor Architecture Reconstruction. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2795–2804. IEEE. ISBN 978-1-72817-168-5. doi: 10.1109/CVPR42600.2020.00287. URL https://ieeexplore.ieee.org/document/9156819/.

[25] Yichao Zhou, Haozhi Qi, and Yi Ma. End-to-End Wireframe Parsing. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 962–971. IEEE, . ISBN 978-1-72814-803-8. doi: 10.1109/ICCV.2019.00105. URL https://ieeexplore.ieee.org/document/9008267/.

[26] Yichao Zhou, Haozhi Qi, Yuexiang Zhai, Qi Sun, Zhili Chen, Li-Yi Wei, and Yi Ma. Learning to Reconstruct 3D Manhattan Wireframes From a Single Image. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 7697–7706. IEEE, . ISBN 978-1-72814-803-8. doi: 10.1109/ICCV.2019.00779. URL https://ieeexplore.ieee.org/document/9010693/.