A Geometric Approach To Building Roof Wireframes

Kunal Chelani
Chalmers University of Technology
chelani@chalmers.se

Abstract

Wireframes are compact yet meaningful representations that easily convey the structure of the 3D scenes. Predicting frames from point clouds, and images independently has been studied in the past however, they haven't been shown to perform well on real world data (for point clouds) and in a sparse view setting (for images). This challenge presents this problem in the form of a real world application with multi modal pre-processed data. In this report, we provide details of our solution submitted to this challenge and discuss its strengths and weaknesses in order to promote interesting future work.

1. Introduction to the challenge

This is a report detailing our submission to the *S23DR* challenge [5] organized as part of the Urban Scene Modelling Workshop in conjunction with CVPR 2024. The objective of the challenge is to predict the structure of house roofs (in metric scale), given certain information about the geometry of the house. This information is in the form of images and a point cloud. The image data consists of two different types segmented images and a depth map from a sparse set of camera views with known poses. The two types of segmented images are 1) a ageneric segmentation based on the (ade20k) segmentation and the other is a domain specific segmentation labelling different parts of a house structure such as the pillars, walls, etc. The point cloud is obtained using Structure from Motion and in some cases have been observed to be merged with LiDAR data. The expected output is a wireframe or a graph with vertices having 3D positions and edges connecting these vertices that would form the 3D boundary of the roof.

2. Existing works on wireframes

2D Wireframe Prediction Line segments and junctions are important visual structures that are used for many computer vision application like image rectification, visual localization [8], Structure from Motion [1, 6, 7], image matching [10], etc. The problem of detecting line segments in images has been studied using both traditional and learnt approaches. [12] is a recent state of the art method that predicts wireframes using attraction fields where each pixel learns the distance from its closest line. Detecting edges or 2D wireframes and combining them from multiple views to obtain a consistent 3D structure could be one of the ways to approach the problem in the challenge, however it isn't part of our method.

3D Wireframe prediction A 3D wireframe is a parsimonious representation for a 3D scene. [16] predict 3D wireframes of buildings from a single RGB image by learning to predict the structure using the vanishing points corresponding to different sets of parallel lines. [6] revisited the line mapping problem, made several robust adjustments to 3D line triangulation and harnessed the accuracy of learnt line segment detectors [9, 12] to predict accurate line maps. [13] learn wireframe structures from such line clouds by learning to form appropriate junctions, however they require accurate line maps as inputs. [14] learn to predict wireframe structure from point clouds directly but their method was only shown to work for object level scales with reasonably accurate point clouds. [4] predict wireframes for building and room level real world scenes but require highly accurate and dense point clouds to do so.

Overall, none of these existing works have shown to predict wireframes from SfM points or from a sparse set of segmented images as input and hence this challenge presented the wireframe prediction in a different perspective.

3. Our approach

In our approach, we do not use the images from general segmentation (based on ade20k) and only use the domain specific segmented images. Therefore, by segmented images, we mean domain specific segmented images in the following.

Our approach is based on first predicting the set of 3D vertices of the wireframe and then connecting the appropriate ones. We use multi-view segmented images, the set of available SfM points and appropriately scaled depth maps to estimate the set of corner points that should belong to

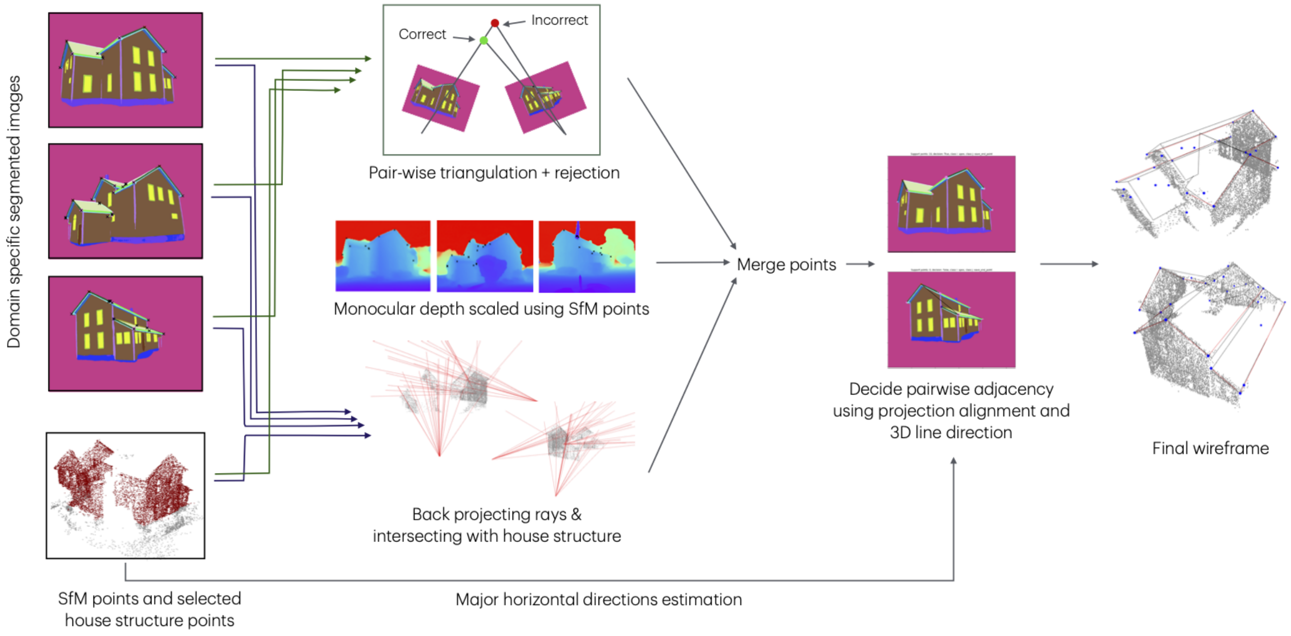
Domain specific segmented images
Correct
Incorrect
Pair-wise triangulation + rejection
Monocular depth scaled using SfM points
Back projecting rays &
intersecting with house structure
Merge points
Decide pairwise adjacency
using projection alignment and
3D line direction
Final wireframe
SfM points and selected
house structure points
Major horizontal directions estimation

Figure 1. Method pipeline

the roof wireframe. We use these different sources of information based on their expected accuracy and dependability. We then select the pairs of vertices that should form an edge based on the vertex classes (from segmentation), the direction of the resulting 3D line segment and its projection onto the segmented images. Figure 1 provides on overview of our approach.

In the following we detail the different steps of our pipeline, the design choices we make, the reason behind those choices and the possible drawbacks of these choices.

3.1. Pre-processing

The SfM points of the house given as input $\mathcal{S}$ already provide an approximate idea about the structure and boundary of the house. This can be used both in estimating the corner point positions and the edges between the points. The point set $\mathcal{S}$ contains the surrounding structures of the house, such as trees, the ground plane, nearby houses, etc which might make it difficult to extract the right information from this set of points. We therefore first clean $\mathcal{S}$ to only contain the structure of the house. $\mathcal{S}$ is upright in orientation, i.e., the Z direction is the opposite of gravity direction. We first remove the points close to the ground plane as this removes points that connect the walls of the house to nearby structures. We then perform a perform a simple DBSCAN [2] clustering with a reasonable threshold and select the points that belong to the largest cluster as the set of clean points. Figure 2 (top) shows examples of cleaned point clouds. We call the set of clean points representing the house structure as $\mathcal{H}$. This isn't a perfect approach for extracting the house structure but is reasonably robust in our experience.

Almost all edges of the roof are aligned with the directions of the walls and therefore it could be useful to obtain the two major directions of the walls of the house. The houses are assumed to be rectilinear, i.e., there are only two perpendicular directions along which the walls occur, which holds true in most cases. To extract these directions, we project the points onto the ground plane ($z = 0$) and then fit robust line using RANSAC [3] to estimate the set of points along one of the major horizontal direction. The other major direction is selected as the one perpendicular to the first one with $z = 0$. Figure 2 (bottom) shows successful extraction of major wall directions - shown by the axes. We call the 2x3 matrix stacking these two directions as rows $\mathbf{D}_h$, and the 1x3 vector pointing in the vertical direction $\mathbf{d}_v$.

3.2. Predicting 3D positions of corners

We focus on predicting three specific types of corner points - *apex*, *eave-end-points* and *flashing-end-points*. As shown in the figure 1, we have three different ways of predicting point positions. The first and the most important one is triangulation of matching 2D corners, the second one is using the monocular depth available as input and third one is using the intersection of back projected rays through 2D corners with the house structure. Since all three of these depend on accurately localizing the corner point in 2D, we discuss that first.

2D corner estimation Since the provided segmented images do not have pixel-wise classes but instead RGB values along with the color coding for each of the classes, we first

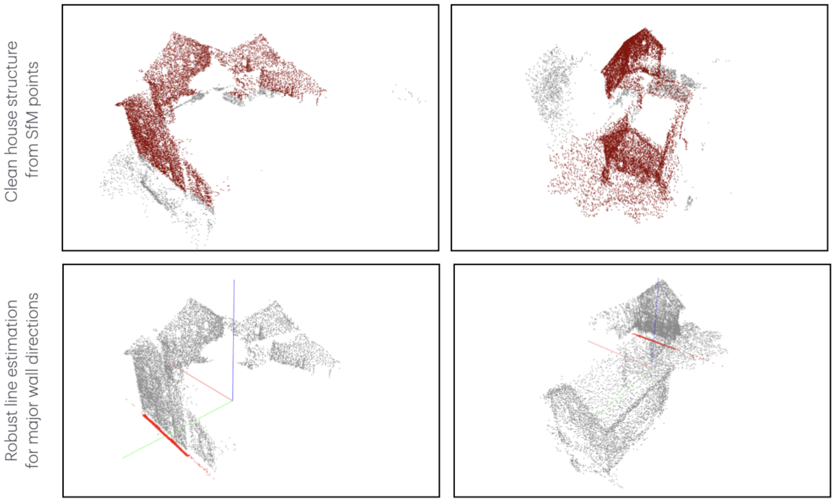
Clean house structure from SfM points
Robust line estimation for major wall directions

Figure 2. Pre-processing the point cloud. The point cloud is cleaned to remove points belonging to nearby structures of the house by clustering points and selecting those of the largest cluster, shown in red (top). The major wall directions are then estimated by projecting the points onto the horizontal plane and running robust line estimation on the projected points. The axis shows the directions of the three major directions.

need to predict the position of the 2D corner points from the ”blobs” of corners available in the segmented images. For each of the three vertex classes mentioned above, we select pixels in a segmented image that lie within a certain threshold of the true color coding corresponding to that vertex class. We then use connected component analysis (similar to that provided in the sample solution) to select a single point per vertex. There are several subtleties here though. In many cases, two different corners in 3D project to nearby positions in the images and hence the corresponding blobs overlap. Predicting a single vertex instead of 2 leads to a higher cost because of missing certain vertices. Eroding the blob instead of dilating it and then selecting the median of resulting connected components seemed like a viable strategy to overcome this problem, however, in practice it leads to too many detected vertices and the overhead of vertex deletion cost did not alleviate the issue in our experience. At the end of this step we have a set of 2D points V_i and corresponding vertex classes C_i associated with each of the images I_i having known camera intrinsics K_i and extrinsics $\mathbf{R}_i, \mathbf{t}_i$.

Triangulating 3D corners If a roof corner (junction) is visible in two or more images, the projections can be triangulated using the known camera poses to obtain the 3D point. The triangulated position is expected to be much more precise as compared to the monocular estimated depth. One way to triangulate would be to establish correspondences between the image points across different images and then triangulate, however since these are not natural images, it wasn't clear if any if the existing image matching techniques would work with these almost single color patches around the detected keypoints. On the other hand, as the camera poses are known, we can use epipolar constraints to filter out incorrect matches. For each image pair I_i and I_j, we triangulate each pair of points $v_i^k \in V_i, v_i^l \in V_j$ such that $c_i^k = c_j^l$ resulting and add the resulting 3D point to both v_i^k and v_j^l. We filter out points where the sum of re-projection errors in both images is more than 10 pixels, resulting in a set of 3D points $\mathcal{P}^{tri}_{filtered}$.

This filtering, however is not enough because on many occasions more than one 2D corners points of the same vertex class lie on the epipolar line corresponding to a corner point. We then filter out triangulated points which are far away from the house structure.

$$\mathcal{P}^{tri}_{final} = \{P^i : d(P^i, H) < d_{thresh} \forall P^i \in \mathcal{P}^{tri}_{filtered}\} \quad (1)$$

Here $d(P^i, H)$ denotes the minimum of a set of distance between the 3D point P^i and the set of 3D points H, and d_{thresh} is a class specific distance threshold. The assumption here is that H covers the whole house structure. This approach therefore fails to triangulate points when SfM points only cover a part of the house.

If all corner points were visible in at least two images, this step would allow us to estimate all 3D corners, however almost all samples have at least a section of the house visible only in a single image. We therefore need alternate ways to estimate 3D corner positions when they cannot be triangulated.

Using Monocular Depth Images We have monocular relative depth available as input. There isn't a common scale that aligns each depth map with the metric scale re-

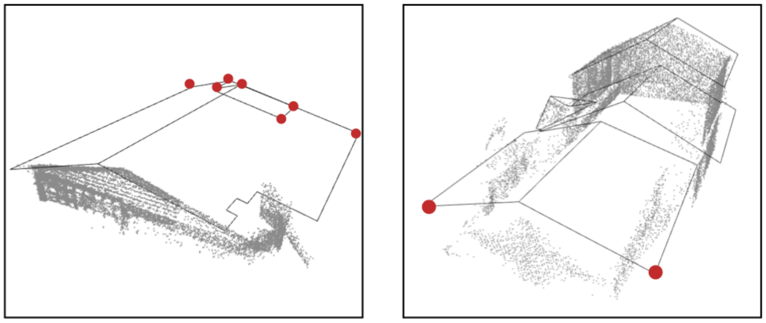

Figure 3. Intersecting back projecting rays gives good estimates for points in many cases, however there are several cases where it fails. Red points indicate failure corner points. **Left**: very limited structure of house contained in the points. **Right**: almost complete structure but missing partially incomplete.

constructions. To estimate this house-wise scale, we project the points in $\mathcal{H}$ into the camera corresponding to each depth map and scale the depth using the mean of ratios of depth at pixels where there is a projected 3D point. To avoid the impact of occluded points projecting even after z-buffering, we select only a fraction of points at a smaller depth. This part of the method probably has the most scope for improvement as there are several reasons leading to very noisy depth even after scaling the depth map using the sfm points. Here are the reasons:

1. Depth maps suffer from major occlusion in many cases. These occlusions are by nearby structures like trees, etc
2. The error in depth is somewhat proportional to the depth of the point - hence the far away corners are not at all accurately estimated. A noise model can perhaps be used to improve this estimation.
3. The corner points have a further ambiguity in depth as the estimated depth at edges is quite noisy. One possible way to overcome this error would be to select the minimum depth for the "blob" of segmented corner points.

After estimating the scale, we get a set of 3D points $\mathcal{P}^{mon}$ associated with all 2D corners V_i in all Images $I_i, i \in \{1 \dots N\}$

Intersecting back-projected rays with house structure In many cases the house structure $\mathcal{H}$ contains better estimates for the corner points than those estimated by the monocular depth. To use these estimates, we project the rays from camera center to the 2D corner points and intersect them with the house structure. In practice we select top-k closest points from $\mathcal{H}$ to the back-projected ray and select the one with least depth - for cases where a ray goes through a wall to intersect the walls behind. We call these set of estimated corner points $\mathcal{P}^{int}$. A re-occuring failure case for such an approach is house with partial SfM points, The left example in fig

Merging points We merge the points from the above three sources to give the final set of points. We iterate through the 2D vertices V_i and corresponding vertex classes C_i for all images $i \in \{1 \dots N\}$. If a triangulated point is available for V_i, we keep it as these are expected to be rather accurate. When triangulated points are not available, we move to the set $\mathcal{P}^{int}$. In failure cases such as those shown in figure 3, the intersection points all lie close together and are co-planar. We try to avoid using the intersection points in such cases by detecting if these conditions satisfy and instead use $\mathcal{P}^{mon}$ instead. Finally, we merge 3D points of the same classes lying within a small distance to each other to form asingle vertex.

5cm	10cm	25cm	50cm	100cm
0.11	0.35	0.63	0.70	0.73

Table 1. Accuracy of triangulated points: fraction of triangulated points within different distance threshold of a ground truth vertex. It can be seen that more than 60 percent of these triangulated points lie within 25 cm of a ground truth point

5cm	10cm	25cm	50cm	100cm
0.10	0.23	0.36	0.41	0.48

Table 2. Dependability of triangulated points: fraction of gt points per scene within different distance threshold of a predicted triangulated ground truth vertex.

3.3. Edge Selection

Once we have estimates of 3D corner points, we want to connect the ones that form the edges of the roof. We use the information of general line-segment directions between

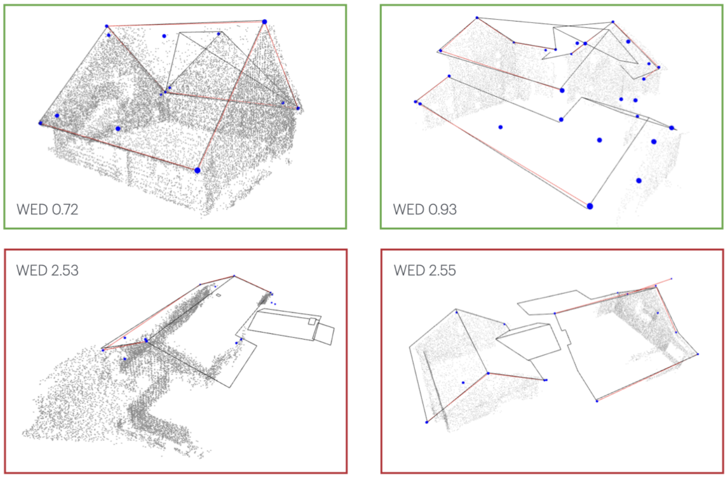

Figure 4. Qualitative results : **Successful** (top) and **failure** cases (bottom) of our approach. Our approach generally has a higher WED because it fails to predict presence of some of the corner vertices.

different classes of vertices to filter out possible edges. For example, an edge connecting a pair of *eave-end-points* generally lies along one of the wall directions $\mathbf{D}_h$ and it has no component along the vertical direction $\mathbf{d}_v$. By applying such rules for every pair of vertex categories, we select a set of potential edges. We then project these 3D line segments into the given cameras and check if the 2D line projection lies along the appropriate segmentation - for example a *ridge* connects two apex points. At the end of this step, we have the final roof wireframe.

4. Results and Discussion

The metric used here is the modified wireframe edit distance (WED). On the test set of the hoho dataset [5], our method reaches an average WED of 1.75 and 1.68 on the public and private splits.

4.1. Accuracy of triangulated points

Since triangulating the corners, in practice leads to accurate point positions, we analyze the fraction of triangulated points within different distances of the nearest ground truth points. Table 1 shows the accuracy of the triangulated 3D points. Even though such points are quite accurate, we miss estimating accurate positions for around half of the ground truth points and this can be attributed to not being able to detect such vertices in enough 2D images, hence the need for a data-driven approach.

4.2. Qualitative results

We present a few successful results of our approach (WED ≤ 1.0) and a few failure cases (WED ≥ 2.5) in figure 4.

4.3. Discussion and Future Work

Ours is a handcrafted approach, which has been attempted to made robust to different types of difficult data points. However, such an approach is of course not robust to all conditions and missing data ground truth vertices. As can be seen from the failure cases in figure 4, it is important to predict a vertex close to each of the ground truth vertices. This isn't always possible because of noisy and incomplete data with occluded parts. For edge prediction, we select a particular set of vertex classes and their expected directions - in many cases, these edges occur against our pre-set rules. A learning based approach would therefore be a much more interesting research avenue for solving this problem.

However, we believe that the geometric accuracy of the triangulated points should also be harnessed in obtaining the final wireframe. Taking inspiration from recent works in automated room layout prediction [11], one possible way to do this can be to use the triangulated points along with the house structure points $\mathcal{H}$, encode them using a transformer[15] and learn to auto-regressively produce the remaining corners given the partial information. If the 3D corner positions are accurately predicted, learning connections seems like an easier problem to solve.

Acknowledgements I thank Fredrik Kahl, Torsten Sattler, Assia Benbihi and Jonáš Kulhánek for the meaningful discussions on the topic of this challenge.

References

[1] Adrien Bartoli and Peter Sturm. Structure-from-motion using lines: Representation, triangulation, and bundle adjustment. *Computer Vision and Image Understanding*, 100(3): 416–441, 2005. 1

[2] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei

Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, page 226–231. AAAI Press, 1996. 2

[3] Martin A. Fischler and Robert C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24:381–395, 1981. 2

[4] Han Yue Shoujun Jia Jintao Li Junyi Wei, Hangbin Wu and Chun Liu. Automatic extraction and reconstruction of a 3d wireframe of an indoor scene from semantic point clouds. *International Journal of Digital Earth*, 16(1):3239–3267, 2023. 1

[5] Jack Langerman, Caner Korkmaz, Hanzhi Chen, Daoyi Gao, Ilke Demir, Dmytro Mishkin, and Tolga Birdal. S23dr competition at 1st workshop on urban scene modeling @ cvpr 2024. `https://huggingface.co/usm3d`, 2024. 1, 5

[6] Shaohui Liu, Yifan Yu, Rémi Pautrat, Marc Pollefeys, and Viktor Larsson. 3d line mapping revisited, 2023. 1

[7] Andre Mateus, Omar Tahri, A. Pedro Aguiar, Pedro U. Lima, and Pedro Miraldo. On incremental structure from motion using lines. *IEEE Transactions on Robotics*, 38(1):391–406, 2022. 1

[8] Branislav Micusik and Horst Wildenauer. Descriptor free visual indoor localization with line segments. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3165–3173, 2015. 1

[9] Rémi Pautrat, Daniel Barath, Viktor Larsson, Martin R. Oswald, and Marc Pollefeys. Deeplsd: Line segment detection and refinement with deep image gradients, 2023. 1

[10] Rémi Pautrat, Iago Suárez, Yifan Yu, Marc Pollefeys, and Viktor Larsson. Gluestick: Robust image matching by sticking points and lines together, 2023. 1

[11] Xinpeng Wang, Chandan Yeshwanth, and Matthias Nießner. Sceneformer: Indoor scene generation with transformers, 2021. 5

[12] ”Nan Xue, Tianfu Wu, Song Bai, Fu-Dong Wang, Gui-Song Xia, Liangpei Zhang, and Philip H.S. Torr. ”holistically-attracted wireframe parsing: From supervised to self-supervised learning”. *”IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)”*, 2023. 1

[13] Xuefei Zhe Di Kang Yajing Xu Peter Wonka Yicheng Luo, Jing Ren and Linchao Bao. Lc2wf:learning to construct 3d building wireframes from 3d line clouds. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2022. 1

[14] Konrad Schindler Jan Dirk Wegner Yujia Liu, Stefano D’Aronco. Pc2wf:3d wireframe reconstruction from raw poiunt clouds. In *Proceedings of the International Conference on Learning Representations*, 2021. 1

[15] Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip Torr, and Vladlen Koltun. Point transformer, 2021. 5

[16] 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 *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019. 1