Roof Surface Reconstruction

Adesh Gupta
IIT Roorkee

June 2024

Abstract

This report describes a solution towards the S23DR [1] challenge. Targeting precise extraction of 3D wire frames from 2D images captured in urban scenes. This method is basically an improvement over the provided baseline submission and implements the naive approach a bit more smartly.

1 Introduction

The goal of this project was to develop an improved solution for the wireframe construction problem as part of the S23DR competition. The proposed method builds upon a previous handcrafted solution by incorporating enhanced techniques for vertex detection, line detection, missing vertex handling, line agglomeration, and depth estimation. Depth estimation being the major part, which led to significant improvement in WED score. The solution proposes the depth estimation at point using mean depth from N nearest values from the sparse depth map obtained by projecting the point cloud on a 2D camera plane.

2 Understanding the Dataset

We are operating on the HoHo dataset released by the problem statement. Each sample of the dataset is an urban house with information captured from at least 3 views. Each view has the following preprocessed data: gestalt segmentation, ade20K segmentation, monocular depth map, and camera parameters. A reconstructed 3D point cloud is also provided for each scene.

- **Camera matrices:** K, R, t Given according to the convention of all the scenes in a dataset iteration.
- **Gestalt Segmentation Map:** Detailed but not precise segmentation map of the house images. As given, gestalt is a domain specific model, which "sees-through-occlusions" and provides detailed information about house parts.
- **Ade20k Segmentation map:** These are the results of standard Ade20k segmentation pipeline which returns a map which creates the mask of house and separates nearby objects and obstructions such as trees.

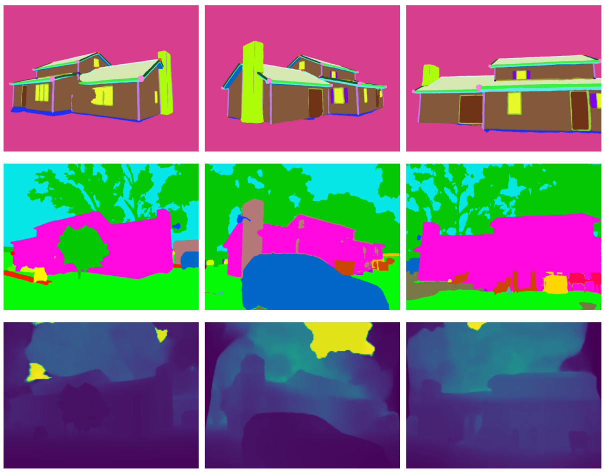


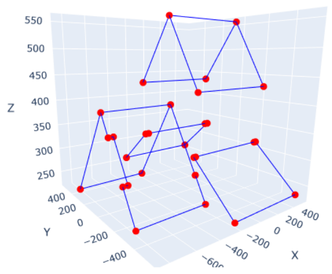
Figure 1: Top to bottom, gestalt segmentation, ade20k segmentation and estimated monocular depth map of three different images of a house.

- **Depth Cm:** Monocular depth estimation which is off from the ground truth and the page specially says that it is ground truth by no means. Better estimation is obtained by projecting point clouds. And ground truth can be obtained by projecting the actual mesh given in the side information.
- **Target variables:** On closer analysis the target variables are the vertices and edges of roofs of the houses. So the problem expects to predict the vertices of roofs of the houses and their connections.
- **Side info during training:** Side info includes the actual mesh data of the house as the ground truth and edge and vertex semantics which are mapped as apex, eave end points etc.

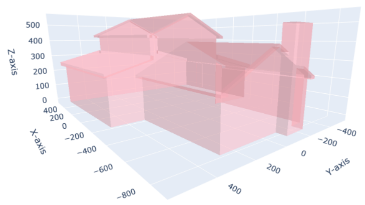
3 Deciding the approach to problem

So the main concern is now to predict the vertices and edges of the rooftops from given inputs. There can be several methods considered based on the different types of data we have been provided with.

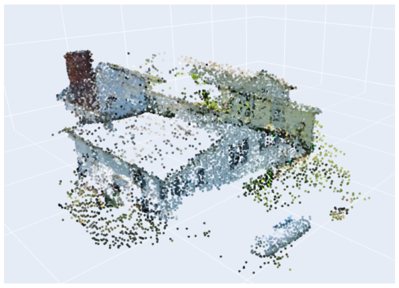
Gestalt segmentation map is seen to be very rich in the data and is used by the baseline submission for prediction. The baseline submission uses the gestalt segmentation to extract the vertices by making the mask



(a) Target vertices and edges



(b) Mesh in side info



(c) Point cloud from colmap (sparsity can be observed in roofs)

Figure 2: Data Visualizations

of apex and eave end point colors and finding their centroids. Edges are found in similar manner by using the segmented data and are converted to 3D world coordinates using the monocular depth provided. The baseline submission uses scale coefficient of 2.5 but it can be found that estimated depth is not that consistent with scale and is major source of error in the prediction of projected location of points in space.

Given with the point cloud data there is a possibility of using different deep learning pipelines for mesh reconstruction from point cloud like PC2WF [2]. But as the point cloud is not from sensors like LiDaR but only a colmap reconstruction, it is very sparse for this type of consideration. Also the sparsity increases for roof points which we want to predict hence this idea is dropped.

Hence we decide to use the segmentation map only but with improved depth information from point cloud after filtering and preprocessing the point cloud. We use a nearest neighbour interpolator to make predictions of depth map from sparse point cloud.

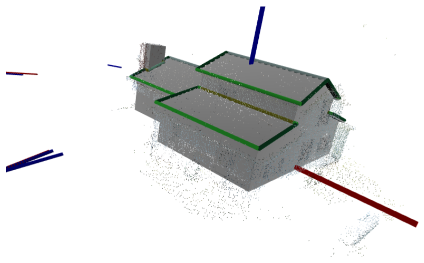


Figure 3: The above image represents that point cloud is also not very accurate for a mesh reconstruction

4 The method

4.1 Vertex Detection

The solution extracts vertex masks from the "gestalt" segmentation using a series of dilations, erosions, and connected components analysis to identify vertex centroids and radii.

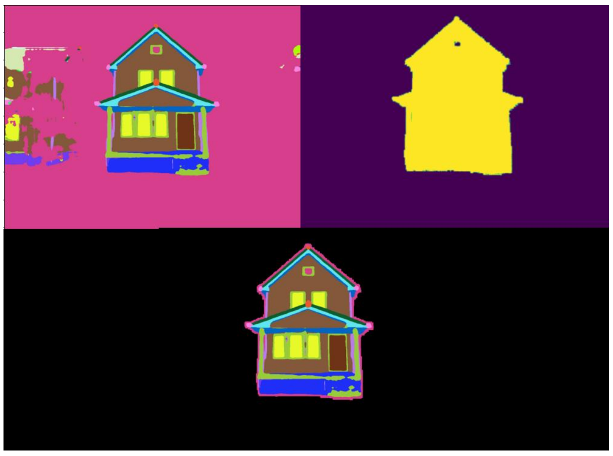


Figure 4: The unclear image is masked to remove remains of surroundings

4.2 Line Detection

The algorithm preprocesses the line mask almost in the same way as in vertex detection using connected component analysis. Ridge class requires additional dialation in the preprocessing step.

Next Hough line detection is used, which returns line ends. A special parameter expands the edge on both ends which allows lines to intersect the vertices in certain threshold. This step helps increasing the efficiency of the next step.

4.3 Missing Vertex Detection

The "gestalt" images have missing vertices, which are extremely easy to detect. Missing vertices, commonly occurring at the intersection of ridges and rakes, are detected by checking the proximity of their ends. If close enough and no other points are nearby, an inferred vertex is added.

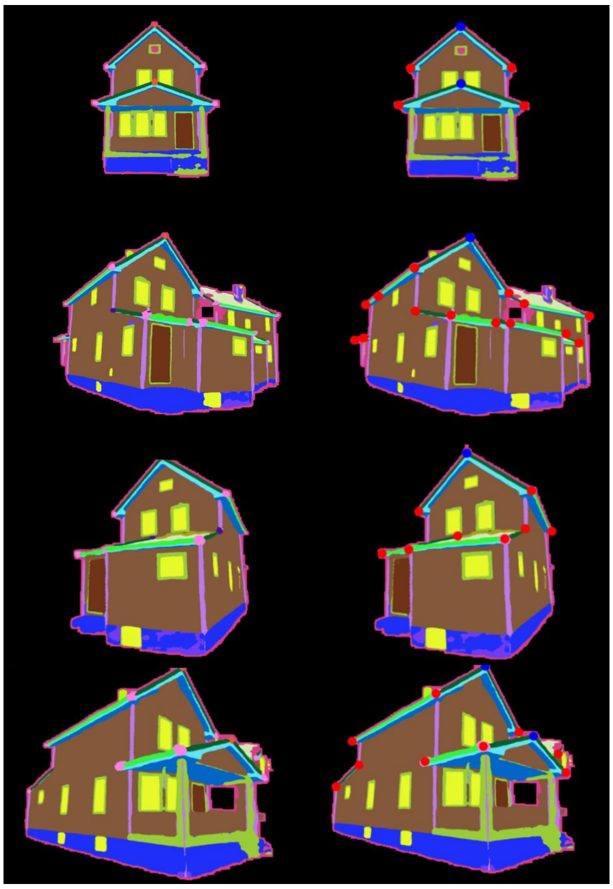


Figure 5: Using connected component analysis to find vertex centroids

4.4 Line Fusion

Line Fusion begins with the detection of all possible lines and the assignment of a corresponding vertex within a defined radius for each line endpoint. This process effectively maps arbitrary lines onto a predefined wireframe, filtering out edges that do not meet the vertex requirements. The next step involves verifying if the detected lines align with the fixed lines within a specified angular threshold, ensuring the lines fit appropriately within the wireframe structure.

The solutions involving radius and degree derive from the nature of 2D projections. The vertex size correlates with the distance to the camera, making it a valid method for restricting line connections. A KDTree data structure is particularly useful in this context, offering an efficient interface for querying points within a radius and reducing computational complexity compared to calculating distances between each element of two sets.

By ensuring lines pass through the vertices rather than just grazing them, the overall accuracy of the detected lines is significantly improved. This method minimizes errors caused by the ambiguous nature of vertex radius and the noise inherent in the detection process. The benefits of this approach become particularly evident when handling complex projections and ensuring that lines are correctly aligned within the 2D framework.

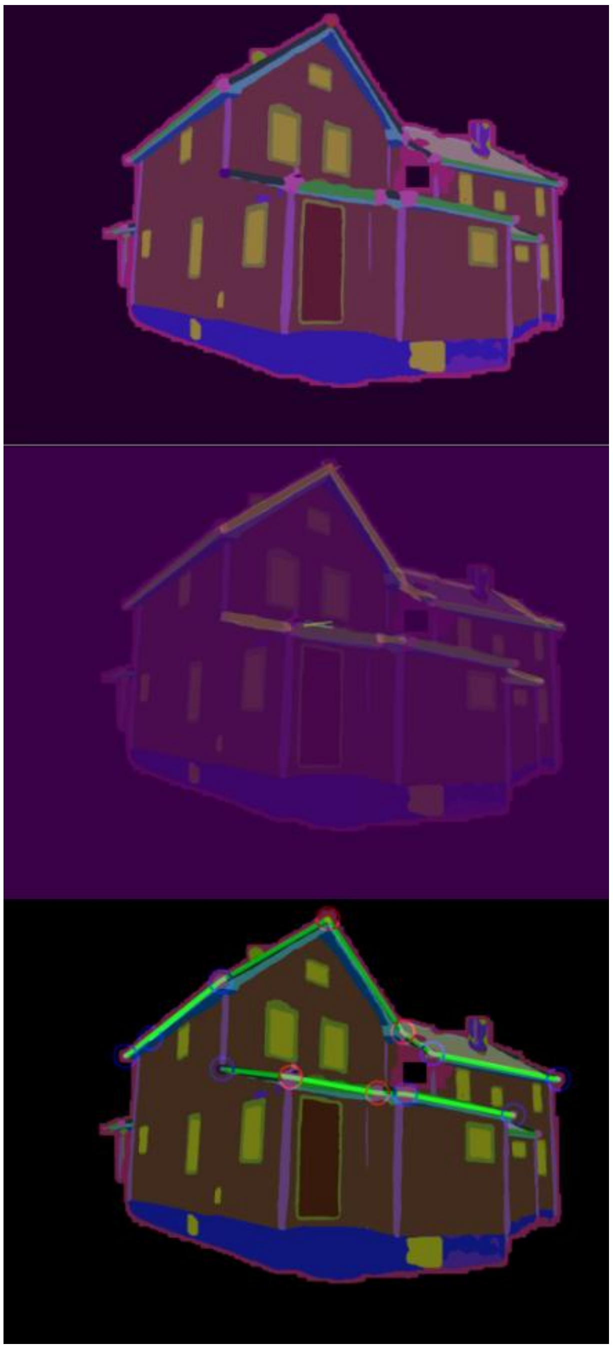


Figure 6: Using cleaned mask first connected components of lines are detected then Hough Line detection algorithm

4.5 The Depth Estimation

The solution uses projection of the point cloud for depth estimation. Initially it performs denoising and unwanted object removal on the point cloud using the DBSCAN algorithm.

With the preprocessing done point projection takes place which takes note of the projected point depth. In order to combat its sparsity, a variation of nearest neighbor interpolation, that rejects neighbors beyond a certain range and outputs a value that is a mean of N neighbors, estimates the found vertex depth. With that information the algorithm proceeds to transform the vertex coordinates into world coordinates. Lastly, it merges the vertices, removes vertices and edges with nans and returns the output.

Since not every image has key point information, the monocular depth map acts as a backup and is used as in the handcrafted solution.

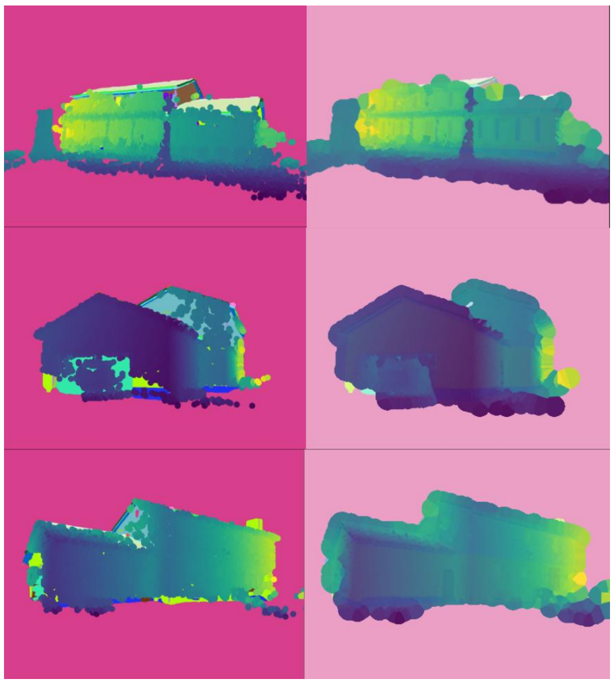


Figure 7: Left: Sparse depth map from projection, Right: Interpolated Depth map obtained from N nearest neighbours

5 Discussion

5.1 The results

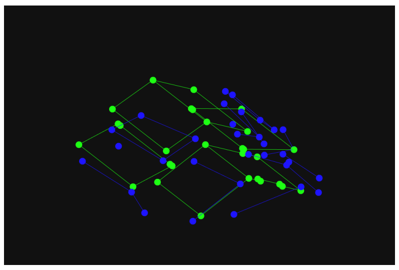
The submission was not done on the hugging-face space as the deadline was already over and I was not able to submit there. But I ran the metric myself over the first 1000 points of the data and it came out to be pretty satisfying. The mean WED was 1.95 which was a lot improved over the baseline submission.

- WED_mu (mean of WEDs): 1.958
- WED_p5 (5th percentile): 1.333
- WED_p25 (25th percentile): 1.695
- WED_p50 (50th percentile): 1.938
- WED_p75 (75th percentile): 2.193
- WED_p95 (95th percentile): 2.620

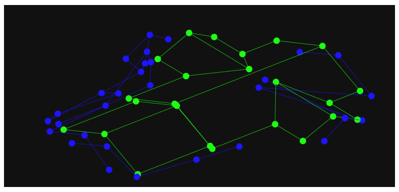
5.2 Area of improvement

Since the great weakness of the algorithm is the correct depth estimation, a more robust algorithm should be used instead, coming up with which would require additional research and expertise.

A reliable method can be encoding the data of sparse depth map obtained from projection and using it along with scaled monocular depth map to train a FCNN which predicts a complete depth map for the image. The training data can be easily generated by generating ground truth depth maps from the mesh given in side info along with camera matrix values. This method was explored but could not be implemented in the short amount of timeframe.



(a)



(b)

Figure 8: Prediction Visualizations

References

[1] 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. usm3d.github.io, 2024.

[2] Yujia Liu, Stefano D'Aronco, Konrad Schindler, and Jan Dirk Wegner. Pc2wf: 3d wireframe reconstruction from raw point clouds, 2021.