3D Reconstruction of Building Morphology from Façade Drawings using Transformed-based Mono-depth Estimation

Carrara A., Nousias S., Dubey R., Borrman A.
Technical University of Munich, Germany
andrea.carrara@tum.de,

Abstract. In this paper we introduce a transformer-based system that reconstructs a 3D building model from line drawing of building facades. Specifically, we generate the dataset of drawings from 3D geometric models of houses and extract the rendered lateral views associated with the relative depth map. The dataset is used to fine-tune the Dense Prediction Transformer model to learn the depth maps of singular views. Each depth map predicted from the transformer model is combined with the relative lateral view and converted into a point cloud. Using plane detection and rigid transformations of the points, we implement a recombination mechanism that aims to correctly reconstruct the building geometry. We develop a hybrid recombination system that combines the predicted depth maps and the orthogonal projection in the 3D space of the point clouds to correct imprecisions of the point conversion. Finally, we regenerate the mesh corresponding to the recovered point cloud using a patch-based Delaunay triangulation.

1 Introduction

Despite the tremendous advances in building information modelling in the last decade, technical drawings (i.e., building plans) are still the primary communication medium between architects/engineers and contractors. The content of the drawing includes in detail the structure and features of the building elements, being illustrated in a standardized way. In complex projects, many technical drawings are available to represent all aspects of a building at various levels of detail. The software used to create these drawings imitates the centuries-old way of working using a drawing board.

However, line drawings cannot be comprehensively understood by computers. The information they contain can only be partially interpreted and processed by computational methods. Basing the information flow on drawings alone therefore fails to harness the great potential of information technology for supporting project management and building operation. A key problem is that the consistency of the diverse technical drawings can only be checked manually. This is a potentially massive source of errors, particularly if we take into account that the drawings are typically created by experts from different design disciplines and across multiple companies. Design changes are particularly challenging: if they are not continuously tracked and relayed to all related plans, inconsistencies can easily arise and often remain undiscovered until the actual construction – where they then incur significant extra costs for ad-hoc solutions on site. In conventional practice, design changes are marked only by means of revision clouds in the drawings, which can be hard to detect and ambiguous.

The limited information depth of technical drawings also has a significant drawback in that information on the building design cannot be directly used by downstream applications for any kind of analysis, calculation, and simulation, but must be re-entered manually which again requires unnecessary additional work and is a further source of errors (Borrmann et al., 2018). The same holds true for the information handover to the building owner after the construction is finished. He must invest considerable effort into extracting the required information for operating the building from the drawings and documents and enter it into a facility management system.

This is where Building Information Modeling comes into play. By applying the BIM method, a much more profound use of computer technology in the design, engineering, construction, and operation of built facilities is realized. Instead of recording information in drawings, BIM stores, maintains and exchanges information using comprehensive digital representations: the building information models. Developing methods to interpret 2D drawings of a building to create a 3D BIM representation would allow to combine conventional drawing-based workflows with modern BIM technology and open the possibility to create building models for existing facilities where only drawings are available.

Multiple families of methods were proposed in the literature already since 2005, entailing rule-based 3D reconstruction from orthographic views, convex optimization-based reconstruction from multi-view capturing, shape matching from database. A significant boost to the field took place with the rise of deep learning. Since the derivation of rulesets for shape estimation is extremely complex for most designs, networks translating photorealistic images to shape from shape where designed. Especially the adoption of attention networks in the multi-view setup yielded promising outcomes. However, most of the aforementioned methods require multiple views of the object or light reflectance information.

In this paper, we introduce a novel method to create a 3D building model from simplistic line schematics corresponding to distinct perspective views of the edifice using the content of facade drawings that represents only the exterior visible information of the construct. The views have a difference of 90 degrees between them. A depth prediction transformer trained with transfer learning and used to derive depth information allowing for the indicative estimation of point cloud data relevant to the 3d surface. The four generated point clouds corresponding to each of the side views are subsequently recombined to form the joint point cloud and surface. This paper presents the first step towards full BIM model reconstruction from drawings considering the exterior reconstruction as initial stage of the procedure.

2 Related Work

3D reconstruction from multiple images is the creation of three- dimensional models from a set of images. It is the reverse process of obtaining 2D images from 3D scenes. The problem is ill posed since the image does not give us enough information to reconstruct a 3D scene. This is due to the nature of the image forming process that consists of projecting a three-dimensional scene onto a two-dimensional image. During this process, the depth is lost. The points visible in the images are the projections of the real points on the image.

Several methodologies have appeared in the literature already from 2005. Lee et al. (2005) devised a rule-based ensemble of methods to reconstruct the 3D shape of CAD models given individual orthographic line drawings. Yet complex manually derived decision trees are required to process not only the visibly defined part but also information relevant to the inner parts of the model. Multiview volumetric silhouette extraction was proposed by (Liu et al. 2006) with the derived models failing in the case of non-convex shapes. Kolev et al. (2009) proposed convex optimization in a multi-view setup (16 to 47 images) achieving high object completeness (91%-99%) and accuracy (0.53mm-1 mm). However, their methods required a high number of images from multiple angles in high compute times (55m -10h). Xue et al. (2012) proposed shape matching from a 3D shape database to reconstruct the 3D geometry of a line drawing depicting visible and non-visible line segments. Mildenhall et al. (2020)

proposed the Neural Radiance Field based method proposing a fully connected neural network that can generate views of complex 3D scenes, based on a partial set of 2D images. It is trained to replicate input views of a scene using a rendering loss. Wang et al. (2021) presented a Transformer based method using as input 12 to 24 views of objects to reconstruct a voxel representation of the initial object achieving high quality results. Yet their method required multiple shaded views from multiple angles. Peng et al. (2022) achieved similar outcomes using an updated transformer-based architecture with just three colored views.

3 Methodology

3.1 Dataset Creation

The process follows the creation of the dataset from the geometry of 3D models of different buildings. The collection of buildings is the House3K dataset (PERALTA 2020) that is divided into twelve distinctive styles of building, each containing 50 house geometries created procedurally. For each batch, five different textures were applied forming the sets. In the scope of the geometry the textures are excluded and only models with different geometry were adopted in the research. The extraction of the sketches is done with the 3D modelling software Blender. The process of the dataset creation is automated in such a way that the software imports the model, rotates the geometry four times and for each rotation it extracts the view captured from the camera and the relative depth map. The rendering of the camera is aided by the plug-in Freestyle that renders the image in a non-photo-realistic manner to represent the view in a technical drawing style. The system is designed to automatically resize the 3D model to fit the camera view and maintain the same camera settings for a uniform reconstruction for all the samples in the database.

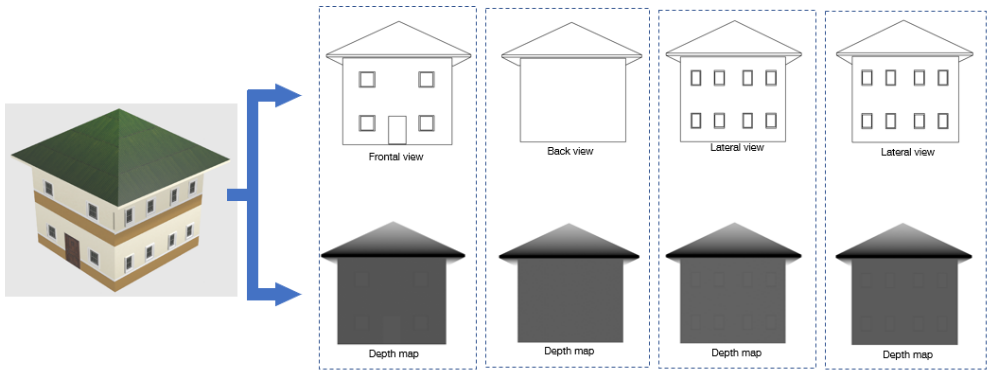


Figure 1: The dataset creations automatically extract each lateral view from the model with the relative depth map.

3.2 Depth prediction

The architecture for estimation of the depth is the Dense Prediction Transformer (DPT) model, the overall structure follows the encoder-decoder architecture using vision transformers as backbone (Ranftl, Bochkovskiy and Koltun, 2021). Vision transformer is used as encoder and it creates patches from the image, operating on bag-of-words manner. Each patch is embedded in a feature space and transformers operate on them using multi-headed self-attention (MHSA) that compares each element with others in the image on a global setting. The multi-headed self-attention computes in parallel multiple attention modules, passes the results to a final head

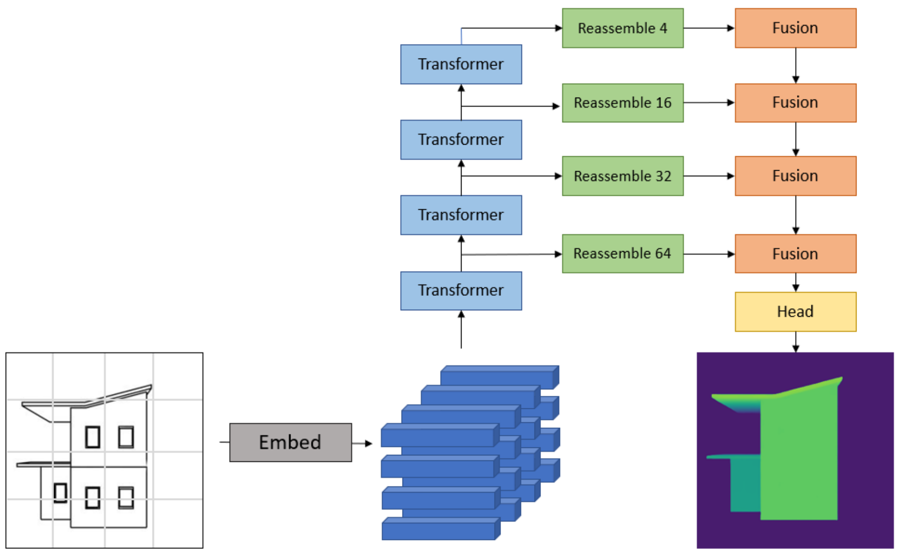


Figure 2: Encoder-decoder with a transformer backbone as encoder. The decoder reassembles features from the transformers and converts them for the final dense predictions.

where the final attention is computed to produce the attention score; this mechanism accounts the possibility of each patch to influence others in the image and aids the prediction based on this feature. The base transformer processing is repeated multiple times in parallel on the image to extract features on different resolutions of prediction. Each output is reassembled from the patch level to an image-like representation given the positional encodings from the original input. From the parallel execution of multiple attention modules, the reassembled elements are multiple, referring to the same input and representing different resolutions of the image. The last part of the neural network consists in the fusion of the multi-resolution elements with a spatial-concatenation operation of the original feature dimension. Technically, the up sampling of the matrix is obtained with a series of transpose convolutional layers adapted to the relative resolution of each feature. The model was originally trained on common depth prediction tasks i.e., ADE20K, NYUv2, KITTI, and reached state of the art in Pascal Context.

One of the main difficulties adopting this model is the problem of fine-tuning it on smaller datasets given the fact that the model was trained with an affine-invariant loss, that means that the predictions are arbitrarily scaled and shifted leading to mismatch on the magnitude of the prediction that dominate the loss function. To overcome the problem, the fine-tuning has been achieved with a scale and shift invariant loss function. The loss function computes scale and shift for the input and the target, applies this change to the input and computes a custom loss combining the mean squared error, a gradient based loss, and a regularization loss. The effect of this adjustment. The four perspectives produced by the neural network are fed into the 3D reconstruction, which combines them to create the model's surface. Prior to training the neural network to judge the proper reconstruction process, the data pipeline was created using the ground truth depths.

3.3 Multi-view reconstruction

Each view is converted in the point cloud using the pinhole camera settings used in the dataset creation algorithm. Given depth value d at (u, v) image coordinate and the depth scale, the x and y focal length of the camera (fx and fy) and the x and y principal points offsets (cx and cy), the corresponding 3d point is defined as follows:

$$z = d \,/\, depth\ scale$$
$$x = (u - cx) * z \,/\, fx$$
$$y = (v - cy) * z \,/\, fy$$

The image is a black and white 2D matrix and the depth map has the same structure representing the distance to the camera for each matrix element. Every pixel of the image is combined with the relative predicted depth from the depth transformer network and applying the reverse transformation taking into consideration the pinhole settings of the generator camera give the three dimensions of each point. The assumption of this element is the ability of the neural network when trained enough to convert unseen images in the settings on which the algorithm reconstructs a 3D representation.

The transformation in point clouds accounts for the deformations generated by the perspective views and transforms the points in an orthogonal reference system. The building's façade can always be precisely recreated using the point cloud generated from the ground truth data, however this is not always the case when using the depth created by the neural network. The edges represent a point of transition between two different depth regions in the training data, and the model predicts a fuzzy distinction between those two different depth areas, which makes it complicated for the network to predict the depth of edges points exactly. As a result, the point cloud is created with noise, which causes mistakes in the 3D reconstruction. While the model learns through a scale-and-shift invariant loss that reduces the magnitude difference between the pre-trained samples and fine-tuned dataset, another problem that occurs when applying the depth estimate from the neural network is the lack of precision in the absolute values of depth. The networks successfully learn how to rebuild the surface's relative depths but can only approximate the actual distance from the camera that would result in a flawless surface reconstruction.

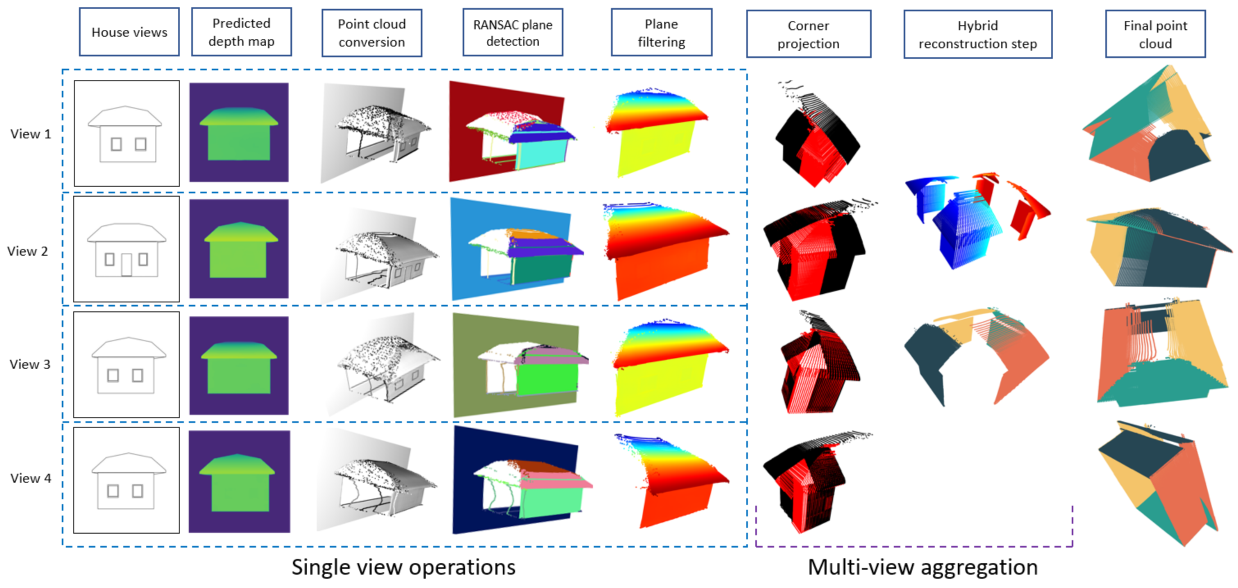


Figure 3: Intermediary steps of reconstruction from the initial drawings to the aggregation of the point clouds

We employ a reconstruction process to clean up the point cloud, match the various viewpoints, and reconstruct the 3D model to address these issues brought on by the neural network's prediction inaccuracies. Given the structure of the house that are structured of rectangular shapes we use RANSAC to extract the fundamental surfaces in the point cloud excluding the noise points and the background. The RANSAC algorithm is an iterative method that optimizes plane fitting in a set of points removing outliers (Derpanis, 2010). We adopt a multi-iterative RANSAC algorithm to fit multiple planes on the point cloud and then remove the group of points that account for less than 5% of the total number of points in the cloud. This allows to remove points that are still undesirable from the correct reconstruction of the façade but follow a structured path in the point cloud.

From the specifics of the dataset construction, in case of perfect estimation of the depth map of each view, the reconstruction would require the simple rotation of each view in the point of mass of the complete surface. It is not the case for the predicted depth map, since the network learns from a scale-and-shift loss and does not forecast the correct absolute distance from the camera for data that was not used for training. A collision or a gap between the facades results from an erroneous combination of the facades caused by the lack of accuracy in the absolute distance.

Because the point clouds in this scenario do not share enough common points across the various viewpoints, established approaches for point cloud registration like Iterative Closest Point and Coherent Point are ineffective. The edges points connecting one orthogonal façade to the other are the only elements they share. Nevertheless, the network loses accuracy at the same precise locations that are necessary for connecting the two contiguous point clouds, making it exceedingly difficult to accomplish an effective registration.

To reconstruct contiguous views, we define a rule-based method based on prior knowledge specific to dataset construction. To meet the connection criteria, the edges of each view must be connected. To accomplish this, the point cloud is divided based on the values of the y-axes, and the points closest to the delimitation line on the maximum or minimum of the reference axis are then extracted. This allows us to simply retain the points required for the best reconstruction, and it also produces a proxy point cloud for the registration sub-task that should resemble the original one.

Each view has two sets of vectors representing the most exterior points in the left and right part of the points cloud, which in the initial image would represent the black edges of the drawing. To have a first correct disposition of the point clouds, we iteratively minimize the distance of the vectors using rigid transformations of the points. This makes it possible to have an improved initial setting point distribution that more closely resembles the original model, each view is still not perfectly aligned because there are still gaps and incongruencies.

On top of the first rigid transformation algorithm, we develop another reconstruction step based on the projections of the edges in the orthogonal setting. The algorithm excludes the first and last percentage of each drawing and recreates the missing part with an orthogonal projection of the edges of the central part of the point cloud. The two views to merge are divided into vertical patches and every patch is projected with the other one, on the collision of the planes corresponding to each patch the 90-degree point cloud is generated.

3.4 3D Mesh creation

The 3D reconstruction is defined as a surface reconstruction given the reconstructed point cloud. We create a mesh from the point cloud using the two-dimensional Delaunay filter on the set of points (Fortune et al. 1995). Delaunay triangulation creates convex meshes, the model of the houses that we are trying to reconstruct have very often concave elements in it. To overcome the limitation of the triangulation algorithm we define a rule-based approach for the reconstruction. We implement a multi RANSAC plane detection on the figure: we split the overall point cloud in different primary geometric planes and then we run the triangulation on vertical patches of each resulting plane from the RANSAC algorithm. Since the Delaunay triangulation optimizes the triangulation on points generating a convex mesh, we divide our complex geometry in convex primary planes and then reassemble the sub-divided elements to obtain the model. The final output is a geometrical mesh that stores the points and the triangular faces generated by the triangulation. In this way we avoid the convexity of the created mesh and ensure similarity with the target point cloud.

4 Experimental evaluation

4.1 Simulation setup

For the creation of the dataset 2D views of 3D models are captured via pinhole camera with focal length $f_x = 711.11$, focal length, $f_x = 1066.66$ and principal point o $= [256{,}256]$. To generate each side view, we rotate the object around the z-axis by $90°´$around the center of mass and rotation origin (0,0,0). In total the data is comprised of 1800 images with dimensions 512x512 pixels equally distant from the camera in the rendering program.

The images in the custom dataset are synthetically made and do not resemble actual scenes captured with cameras but rather abstractive line representations of the captured object. Such an example is presented in Figure 5a. The object's background was assigned a large integer value equal to 65504. To avoid overfitting with the background we normalized the background value based on the relative distance to the camera. The dataset is divided into 80% for training and 20% for testing.

Pytorch is used to create the DPT-large model, which is then trained on GPU. The loss function is a combination of mean squared error on projected depth using the scale and shift transformation and a regularization loss with alpha 0.5 based on gradient loss. The results are extracted via bicubic interpolation by the head that outputs the final depth map from the recombination decoder. We trained the network with a batch size of 4 and Adam with a learning rate of 1e-5 for the backbone and 1e-4 for the decoder weights.

The intrinsic of the pinhole camera defines the point cloud formation. Each RANSAC plane detection is performed 500 times with a minimum ratio of 0.005 and a threshold of 0.00001. The settings have been empirically validated to identify the appropriate amount of detail on numerous models. To rebuild from the boundaries of the views, 30 points were sampled along the y-axis using a 1% threshold. Through projection, 20% of the view's exterior section is recreated. The statistical outlier removal method is used to repair errors in point clouds with a minimum of 20 points and a standard deviation ratio of 2. The 3D mesh reconstruction uses the 2D Delaunay filter with an alpha value of 3.

4.2 Results

Figure 4 presents the training and validation loss for the current dataset. In total the training loss decreased by 83% and the validation loss by 53%. Qualitative evaluation of the model outcomes is presented in Figure 5 where the depth prediction is evaluated for various training epochs. It is visible that depth prediction improves allowing the model to define the depth value of specific details also learning to represent details of buildings that vary significantly from the images on which it was pre-trained on. The model at first applies the deformation of perspective analysis and then learns the absence of floor and the main components of the building.

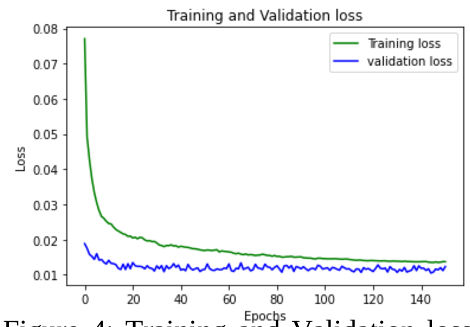


Figure 4: Training and Validation loss on the drawings' dataset

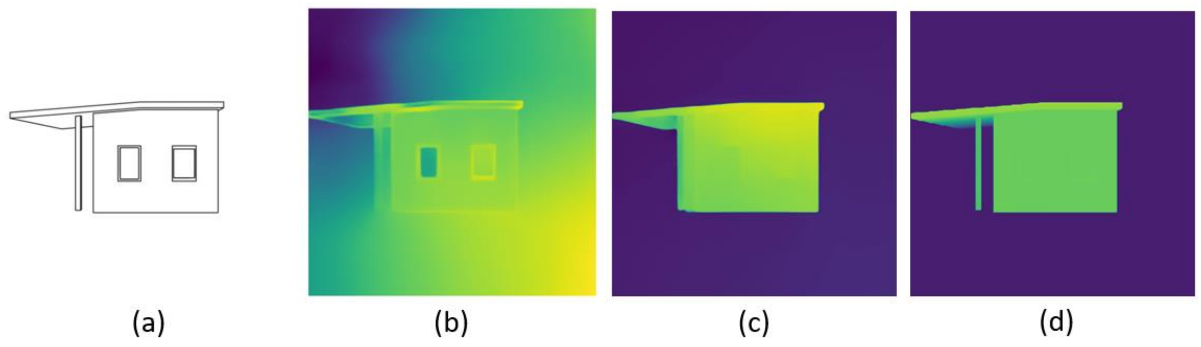
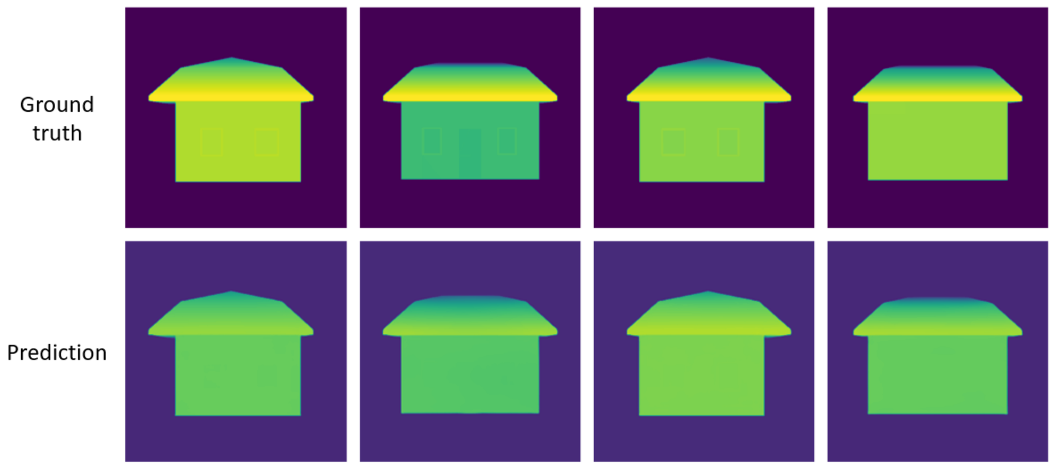


Figure 5: Results of training on a lateral view in the test set. (a) input image to the network, (b) prediction of the base model, (b) prediction at epoch 0 in the training, (b) is prediction at epoch 150.



MSE excluding background:	0.0194	0.0399	0.0661	0.0487	Mean MSE: 0.0435
MSE including background:	0.0611	0.1190	0.2099	0.1396	Mean MSE: 0.1323

Figure 6: Results of mean squared error between ground truth and prediction for a building in the test set including and excluding the background.

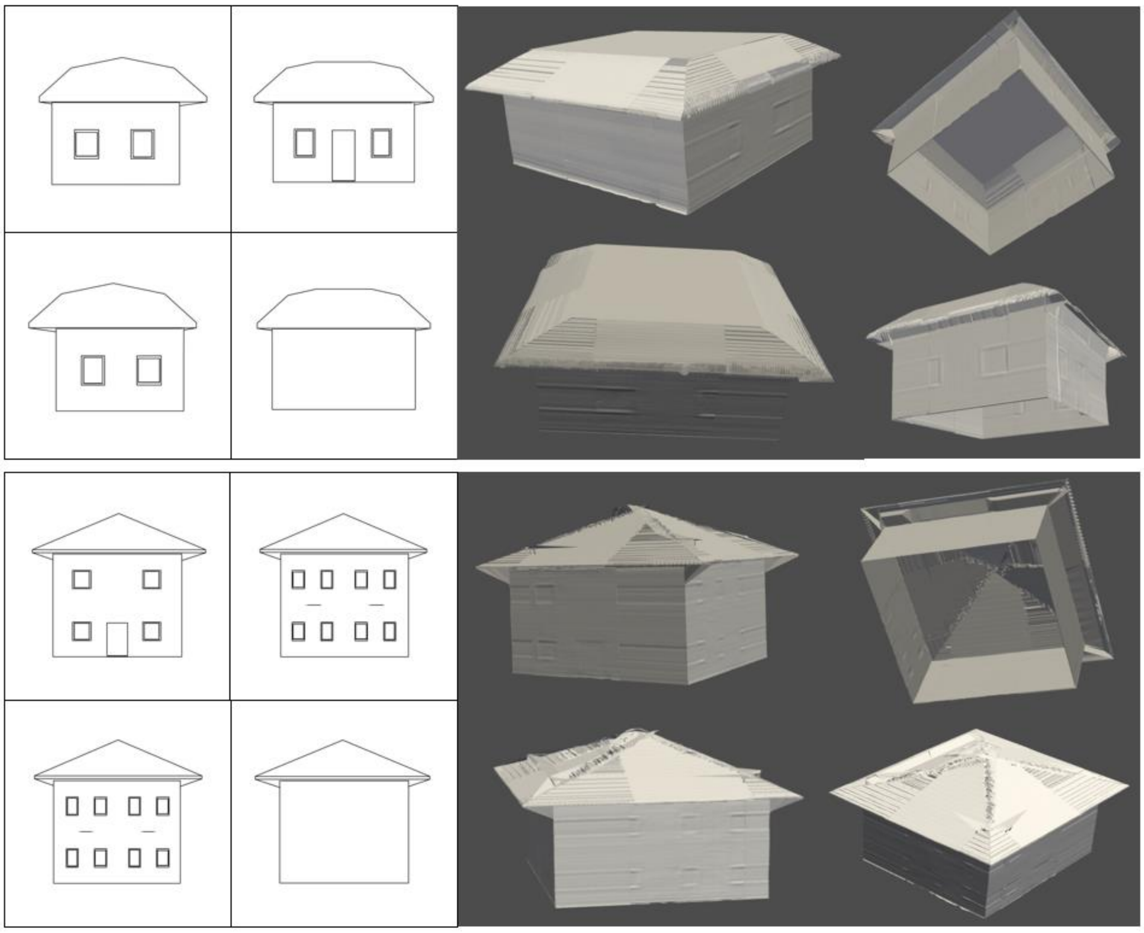


Figure 7: Results of training of 3D reconstruction from four lateral views of two different buildings

The prediction outcomes are evaluated using the mean squared error metric. The background is removed from the reconstruction when the depth map is transformed to point cloud format. Therefore, for each view, we evaluate the prediction error including and omitting the backdrop. The findings from the test set of 100 images reveal a mean MSE score of 0.07414 including the background and a value of 0.17949 omitting these pixels from the computation. The results reveal that the background dominates the loss function during training, decreasing the deep learning model's ability to improve predictions for facades. The overall pipeline of reconstruction is effective but strongly dependent on precise depth predictions of the model. Even if the error from the prediction is low, the conversion to point cloud might bring some outlier that falsely reconstruct the overall geometry. The post-processing and reconstruction mechanisms decrease the outliers correcting the point clouds and reconstructing the parts that would lead to deformations of the overall structure. Figure 7 presents a qualitative evaluation of the final geometry reconstruction. The prediction pipeline is designed to be generic in the reconstruction and focused on maintaining the building's details.

5 Conclusion and future steps

In this paper, we presented a novel method that generates 3D buildings from line drawings depicting lateral views of the building. From a dataset of 3D models of houses, we developed an automation system that renders each side in a drawing style, extracts the depths and uses the data to fine-tune a transformer based deep learning model to learn the depth of each view. The model learns to improve the predictions for the specific dataset, generating realistic relative depths but missing the correct absolute distances. To overcome this problem, each depth and image is converted into a point cloud, cleaned from outliers, and recombined through a hybrid system of predicted and projected depth. The recombined point cloud is converted into a mesh following a patch-based Delaunay triangulation on multiple planes of the point cloud to assess concavity of the final model. The results are a promising first into the processing of actual technical drawings that typically have significantly more "noise". To achieve a complete generation of a BIM model from drawings, floorplans and sections must also be considered to reproduce the interior of the buildings correctly.

References

Derpanis, K.G. (2010). Overview of the RANSAC Algorithm. [online] Available at: http://rmozone.com/snapshots/2015/07/cdg-room-refs/ransac.pdf.

Fortune, S. (1995). Voronoi diagrams and Delaunay triangulations Lecture Notes Series on Computing, pp.225–265. doi:https://doi.org/10.1142/9789812831699_0007.

Kim, S., Park, S. and Yu, K. (2018). Application of Style Transfer in the Vectorization Process of Floorplans (Short Paper). [online] Dagstuhl Research Online Publication Server. Doi: 10.4230/LIPIcs.GISCIENCE.2018.39.

Kolev, K. *et al.* (2009) 'Continuous global optimization in multiview 3D reconstruction', *International Journal of Computer Vision*, 84(1), pp. 80–96. Available at: https://doi.org/10.1007/s11263-009-0233-1.

Lee, H. and Han, S. (2005) 'Reconstruction of 3D interacting solids of revolution from 2D orthographic views', *CAD Computer Aided Design*, 37(13), pp. 1388–1398. Available at: .

Borrmann, A., König, M., Koch, C. and Beetz, J. (2018). Building Information Modeling: Why? What? How? Building Information Modeling, [online] pp.1–24. doi:https://doi.org/10.1007/978-3-319-92862-3_1.

Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R. and Ng, R. (2020). NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. arXiv:2003.08934 [cs]. [online] Available at: https://arxiv.org/abs/2003.08934.

Peng, K. *et al.* (2022) 'TMVNet:Using Transformers for Multi-view Voxel-based 3D Reconstruction', *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, 2022-June, pp. 221–229. Available at: https://doi.org/10.1109/CVPRW56347.2022.00036.Peralta, D. (2020) "Next-Best View Policy for 3D Reconstruction," Lecture Notes in Computer Science, pp. 558–573. Available at: https://doi.org/10.1007/978-3-030-66823-5_33.

Ranftl, R., Bochkovskiy, A. and Koltun, V. (2021). Vision Transformers for Dense Prediction. arXiv:2103.13413 [cs]. [online] Available at: https://arxiv.org/abs/2103.13413v1.

Saleh, S., Manoharan, S., Nine, J. and Hardt, W. (2022). Perception of 3D Scene Based on Depth Estimation and Point-Cloud Generation. Intelligent Decision Technologies, pp.495–508. doi:10.1007/978-981-19-3444-5.

Tutzauer, P. and Haala, N., 2015. Façade reconstruction using geometric and radiometric point cloud information. The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences, 40(3), p.247.

Wang, D. *et al.* (2021) 'Multi-view 3D Reconstruction with Transformers', *Proceedings of the IEEE International Conference on Computer Vision*, pp. 5702–5711. Available at: https://doi.org/10.1109/ICCV48922.2021.00567.

Xue, T., Liu, J. and Tang, X. (2012) 'Example-based 3D object reconstruction from line drawings', *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 302–309. Available at: https://doi.org/10.1109/CVPR.2012.6247689