3Ts_Model - Sketch-Based Generative 3D Model for Architectural Design Process

David Chen
School of Architecture
Carnegie Mellon University
Pittsburgh, PA 15213
davidch2@andrew.cmu.edu

Chia Hui Yen
School of Architecture
Carnegie Mellon University
Pittsburgh, PA 15213
huiyenc@andrew.cmu.edu

Graham Felton
School of Architecture
Carnegie Mellon University
Pittsburgh, PA 15213
gtf@andrew.cmu.edu

Karthick Raja
School of Architecture
Carnegie Mellon University
Pittsburgh, PA 15213
kbangaru@andrew.cmu.edu

Abstract

Generative models offer exciting potential for accelerating architectural design iteration, however exisitng 3D synthesis models often struggle to produce outputs with the structural integrity and spatial coherence required for architectural applications. This paper introduces 3T3D, a novel approach for sketch-based 3D model generation tailored for the architectural design process. We propose a Vision Transformer (ViT) based architecture, leveraging a pre-trained DINOv2 encoder, to map three orthogonal 2D sketch inputs (edge maps) to a 3D representation encoded via triplanes. We explore two methods for encoding geometry within the triplanes: simple binary occupancy maps and a richer representation combining surface normal maps and Signed Distance Fields (SDFs). We also curate a dataset of architectural forms using a generative pipeline with Stable Diffusion XL for image generation, TripoSR for 3D mesh creation, and Informative Drawings for generating edge maps of the rendered 3D meshes. Evaluating our model using Chamfer Distance, our model trained on binary occupancy triplanes achieved a mean CD of 0.200, showing promise but indicating further optimization is needed when compared to state-of-the-art models. Qualitative results from the normal map with SDF model suggest further refinement of the architecture could be necessary to learn more detailed geometry information. Overall, 3T3D demonstrates the feasability of combining ViT with triplane representations for an intuitive sketch-to-3D design workflow and outlines key areas for future inquiry.

1 Introduction

1.1 Context and Motivation

Generative models allow for fast design iteration and a new interactive modality for the design domain. As we are all transitioning from the field of architecture into computational design, we would like to explore applications of these models into our domain as this has remained an area with little experimentation. Our project will be addressing 3D-aware, 2D to 3D model generation through the use of triplanar feature representations of 3D objects with a Vision Transformer Architecture. Our ultimate goal is to build a design tool that architects can use to generate 3D iterations of simple sketches in real time.

33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, Canada.

1.2 Problem Statement

How can we adapt 3D-aware generative models to produce high-quality design forms while maintaining structural integrity and spatial coherence? Existing synthesis models often struggle with generating structured, functionally relevant 3D outputs, as they are primarily designed for object-centric domains rather than architectural applications. We propose a Vision Transformer (ViT)-based architecture that leverages triplanar representations to better capture the spatial and hierarchical relationships inherent in architectural design, enabling controlled 3D generation from 2D sketches.

1.3 Contribution Overview

We build upon the 3D-aware conditional synthesis frameworks presented in pix2pix3D. However, instead of conditioning on label maps, our model conditions on sketches, allowing for the user to input a sketch to our model and receive a 3D model as output.

2 Literature Review

2.1 Background

The work of Deng et al. [2] discusses how 3D-aware synthesis can be achieved with conditional generative models and neural radiance fields, and can be done so efficiently through the use of the triplanar representation of 3d objects. We choose this project as a primary source due to its potential for application in the design domain, specifically its applications in the field of architectural design. While the primary purpose of the paper is for novel view synthesis, we see potential for its use as a "sketch-to-3d" design tool provided the underlying 3D representation can be extracted and reconstructed into a mesh.

Shue et al. [8] present an extension of the concept of triplanar representation by integrating it within a diffusion architecture, seeing potential in such a model to generate novel design explorations. However, their method is optimized for object-centric tasks (e.g., ShapeNet categories like cars and chairs) rather than structured environments like buildings. This limitation directly motivates the extensions of our project, as we aim to adapt triplanar diffusion techniques for architectural applications, where complex spatial arrangements and design constraints must be preserved.

3 Baselines

3.1 Baseline Selection

Both chosen baselines were selected due to their strong resonance with our project goals. Below is a brief description of the chosen baselines:

- **Pix2Pix3D** [2] — 3D-aware conditional generative model designed for controllable, multi-view-consistent image synthesis. Given a segmentation or edge map, the model generates an image of the model from different viewpoints while maintaining geometric consistency. It achieves this by learning a 3D scene representation that encodes color, density, and semantic labels at each 3D point, allowing it to synthesize both images and their corresponding pixel-aligned label maps. The underlying 3D representation is parameterized using triplane features and decoded using a lightweight MLP-based volume renderer. Figure 1 describes the model's architecture [2].
- **Neural Field Diffusion** [8] — Diffusion model to generate 3D neural fields by first converting 3D objects into triplane representations. A shared MLP decodes these triplane features into occupancy fields, representing the 3D structure. A 2D diffusion model (DDPM backbone) is then trained on the normalized triplane features to generate new samples. Once generated, the triplane features are decoded to reconstruct detailed 3D shapes. Figure 2 describes the architecture.

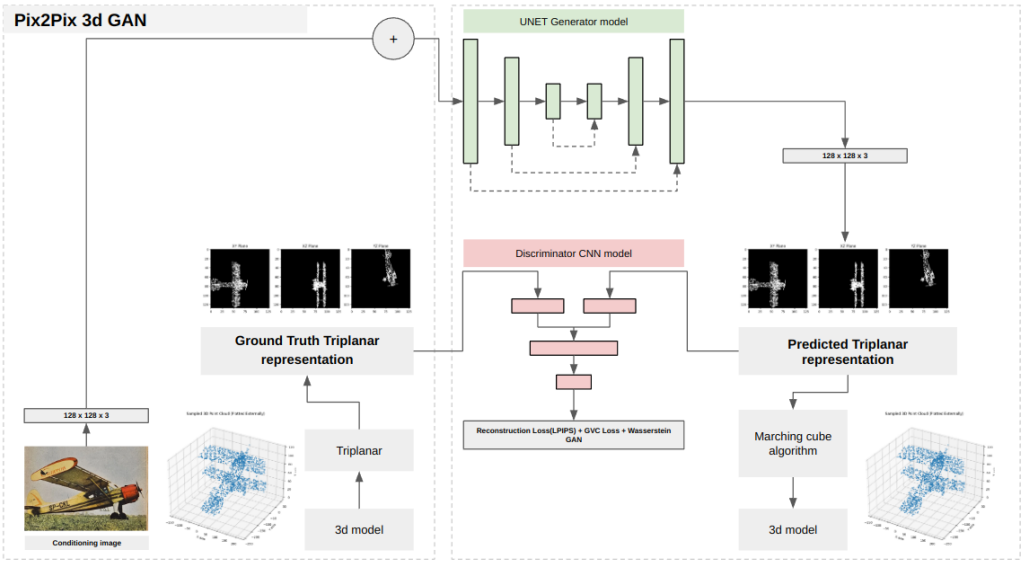


Figure 1: `pix2pix3D`[2] Model Architecture

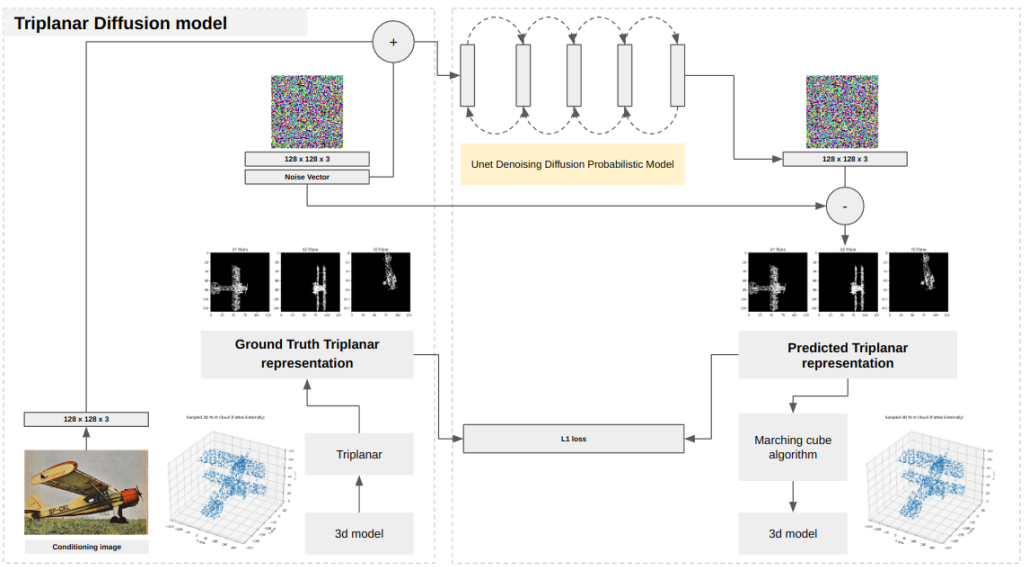


Figure 2: DDPM Baseline Architecture Diagram

3.2 Midterm Baseline Implementation

Following the architecture defined by Shue et al. [8], we created two toy prototypes to test training of the triplanar representation. Table 1 outlines the architecture of each model.

4 Datasets

Existing 3D architectural datasets often lack suitability for unique form exploration due to excessive detail granularity or insufficient quality [14][13]. To overcome this, we created a tailored architectural dataset. We first generated 1800 building images using Stable Diffusion XL [7], converted these to .obj meshes via TripoSR[10], and selected the best 500 candidates. For these meshes, we rendered

Table 1: Architecture of our two baseline models

Model	Architecture Overview
Triplane Encoder/Decoder	Encoder: `MLP` {`Linear(128)` → `ReLU` → `Linear(128)` → `ReLU` → `Linear(128)`} Decoder: `MLP` {`Linear(128)` → `ReLU` → `Linear(128)` → `ReLU` → `Linear(128)`}
Triplanar DDPM (Diffusion)	Diffusion Process: $Forward \rightarrow Reverse$ Unet Backbone: $Encoder \rightarrow Middle \rightarrow Decoder$ Encoder Block: `ResnetBlock` x2 → `LinearAttention` → `Downsample` Middle Block: `ResnetBlock` → `Attention` → `ResnetBlock` Decoder Block: `ResnetBlock` x2 → `LinearAttention` → `Upsample` ResnetBlock: `TimeEmbedding MLP` → `Block` x2 → `Residual`

aligned front, top (plan), and side (elevation) views. Finally, we generated corresponding edge maps from these views using Informative Drawings[1] to serve as sketch-like inputs for our model.

5 General Overview

We choose to follow the approach for representing 3D data as a triplane defined by Shue et al. [8]. Each element of our 3D dataset is converted into a triplanar representation by decomposing each scene into three orthogonal 2D feature maps $\mathbf{F}_{xy}, \mathbf{F}_{xz}, \mathbf{F}_{yz} \in \mathbb{R}^{N \times N \times C}$ with spatial resolution $N \times N$ and channel count C[8]. Thus the triplane can be defined as:

$$\text{NF}(\mathbf{x}) = \text{MLP}_\phi(\mathbf{F}_{xy}(\mathbf{x}) + \mathbf{F}_{xz}(\mathbf{x}) + \mathbf{F}_{yz}(\mathbf{x}))$$

To query a point $\mathbf{p}$, we sample points from each plane

$$\mathbf{v}_{xy} = \mathbf{F}_{xy}(x, y), \mathbf{v}_{xz} = \mathbf{F}_{xz}(x, z), \mathbf{v}_{yz} = \mathbf{F}_{yz}(y, z)$$

We can then combine them to get the final point:

$$\mathbf{n_p} = \mathbf{v}_{xy} + \mathbf{v}_{xz} + \mathbf{v}_{yz}$$

Which is then decoded by the MLP:

$$\text{NF}(\mathbf{n_p}) = \text{MLP}(\mathbf{n_p})$$

For sake of experimentation, we explored two possible methods for constructing triplanes encoded with different features:

1. **Binary Occupancy Maps:** Our first approach encoded simple occupancy. Each plane stored a binary value indicating whether the corresponding projected 2D location was inside or outside the object's projection onto that plane. Each plane $\mathbf{F}_n \in \{\mathbf{F}_{xy}, \mathbf{F}_{xz}, \mathbf{F}_{yz}\}$ has $C = 1$ channels where the occupancy value at coordinates (u, v) is $o_{uv} \in \{0, 1\}$ where 1 denotes a point being inside the object, 0 denotes it being outside.
2. **Normal Maps:** We additionally explored encoding the geometry using 3D normal vectors directly in the triplanes such that each plane $\mathbf{F}_n$(where $n \in \{xy, xz, yz\}$) is a feature grid such that $\mathbf{F}_n \in \mathbb{R}^{N \times N}$ stores 3D vectors. The feature at (u, v) is a vector $\mathbf{v}_{uv} = (n_x, n_y, n_z)$ where the value n defines the surface normal at the point. We combine this with an encoding of the Signed Distance Field of the model at the 3 viewpoints to test if having more information increases the quality of the generated model.
3. **Signed Distance Fields:** The Signed Distance Field (SDF) represents the shortest distance to the object's surface [5]. Each plane P_n stores the distance values ($C = 1$). The feature $\mathbf{F}_n(u, v) = d$ represents the signed distance where $d < 0$ indicates inside the surface, $d > 0$ indicates outside, and $d = 0$ indicates d is on the surface.

5.1 Architecture Summary

We choose to implement a Vision Transformer model based on the pretrained `DINOv2` as we find the model's ability to produce an implicit understanding of scene geometry paired with its ability to be

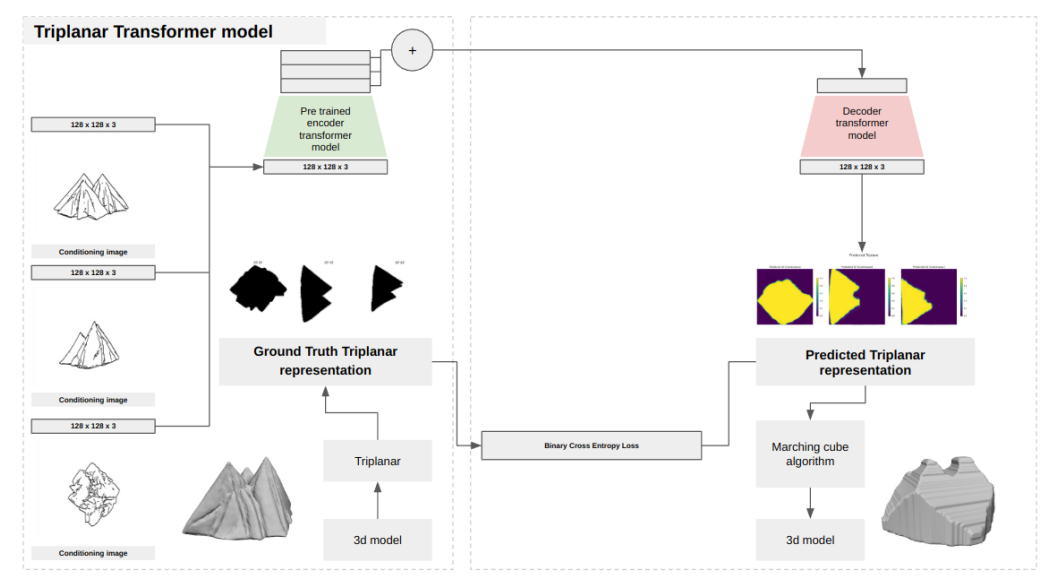


Figure 3: Vision Transformer Model Implementation

easily trained made it amenable for our project [4]. The implementation is detailed in Table 2 and Figure **??**. We used the Marching Cubes algorithm to extract the 3d surface mesh from an estimated volumetric occupancy field, which was constructed by sampling and combining data derived from the triplane representation.

Table 2: **Architecture Overview**. B: Batch size, N: Number of patches, D_{enc}: Encoder dim, D_{dec}: Decoder dim, H_{out}, W_{out}: Output resolution, C_{out}: Output channels. See code/previous table for full details.

Stage	Component / Operation	Key Details	Output Shape
		Encoder	
Input Encoder	3 Image Views + ViT	`DINOv2`	$3 \times [B, N, D_{enc}]$
		Decoder	
Input Fusion	Linear Projection + Add	Project each view embeddings to D_{dec}, Sum	$[B, N, D_{dec}]$
Core Decoder	Spatial PosEnc + Transformer	Reshape, Add 2D PosEnc, Transformer Layers	$[B, N, D_{dec}]$
Upsampling	Upsampling Neck	Reshape + ConvTranspose Stages	$[B, C_{neck}, H_{out}, W_{out}]$
Output Layer	Conv 1x1 + Tanh	Project to C_{out} channels, Apply Tanh	$[B, C_{out}, H_{out}, W_{out}]$

5.2 Evaluation Metrics - Loss

For training the ViT model using the Binary Occupancy Maps, the model is evaluated using Binary Cross-Entropy (BCE) Loss, defined as [6]:

$$\mathcal{L}_{\text{BCE}} = -\frac{1}{N}\sum_{i=1}^{N}[y_i \log(p_i) + (1 - y_i)\log(1 - p_i)]$$

where N is the number of samples in the batch, y_i represents the ground truth label $(0 \text{ or } 1)$ for the i^{th} sample, and p_i represents the predicted probability of the voxel being occupied. BCE is particularly suitable for this task since our model outputs a probability for each voxel in the 3D space, representing whether that voxel is occupied.

For the task of training the ViT Model on the combined Normal Map and SDF triplanes, we employ the L1 Loss since this task is the regression between two continuous variables. The L1 loss is defined as the mean of the absolute difference between between the predicted value $\hat{\mathbf{y}}_i$ and the ground truth

value $\mathbf{y}_i$ [6]:

$$\mathcal{L}_1 = \frac{1}{N}\sum_{i=1}^{N} \|\hat{\mathbf{y}}_i - \mathbf{y}_i\|$$

The Triplanar DDPM model utilizes Cross-Entropy loss, defined as [3]:

$$H(p,q) = -\sum_{x\in\mathcal{X}} p(x)\log q(x)$$

where $p(x)$ represents the true probability distribution of the data, $q(x)$ represents the predicted probability distribution, and $\mathcal{X}$ is the set of possible output categories. This loss function is chosen to allow for a multi-class output from the triplanes for conditioning of semantic classes.

5.3 Evaluation Metrics - Quantitative Comparisons

Comparing our model with our baselines proved challenging given the idiosyncratic nature of our model, thus we elect to compare our model using Bidirectional Chamfer Distance between the ground truth model and the predicted model, respectively $S_1, S_2 \subseteq \mathbb{R}^3$, which is given by the equation [11]:

$$d_{CD}(S_1,S_2) = \sum_{x\in S_1} \min_{y\in S_2}\|x-y\|_2^2 + \sum_{x\in S_1} \min_{y\in S_1}\|x-y\|_2^2$$

Our model trained on binary occupancy based triplanes seemed to best performance and we compare its results to that of our baseline selection, with its individual results in Table 3 and its results compared to others tabulated in Table 4. While our model's CD is lower than our chosen baselines as well as three state of the art models, we believe with a longer training time we can achieve a competitive score.

Table 3: Chamfer Distance (CD) values of our model.

Metric	**Value**
Total `.obj` models evaluated	300
Mean CD	0.200
Median CD	0.192
Min CD	0.075
Max CD	0.429

Table 4: Chamfer Distance (CD) results compared to baseline.

Model	**CD**
One-2-3-45	0.227
3T3D (Ours)	**0.200**
ZeroShape	0.160
TGS	0.122
OpenLRM	0.180
TripoSR	0.111

5.4 Experiments - Results

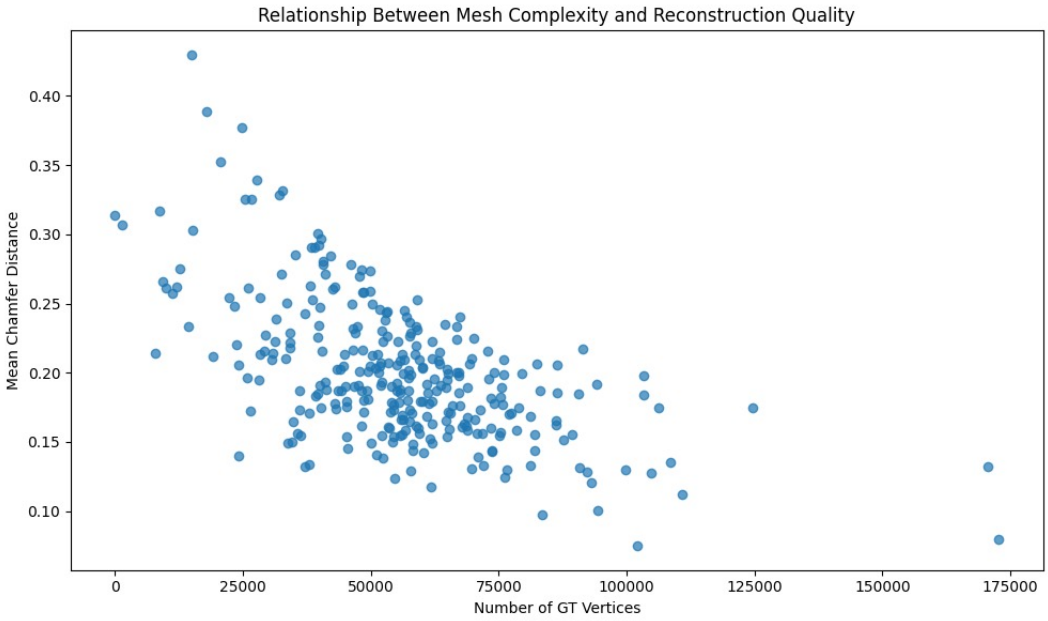
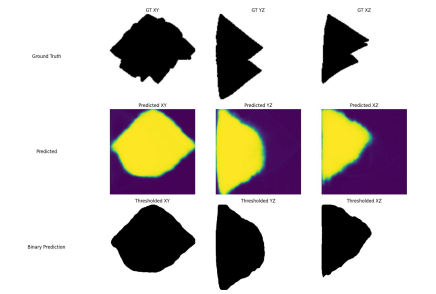
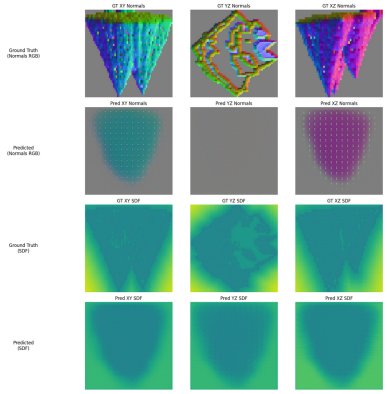


Figure 4: Scatterplot showing Chamfer Distance values obtained by from Binary Occupancy-trained model

The reconstruction quality of the Binary Occupancy-trained model relative to ground truth mesh complexity is illustrated in Figure 4. The scatter plot shows a general trend in which the Mean Chamfer Distance decreases as the number of ground truth vertices increases, suggesting the model achieves lower reconstruction error on more complex geometries within the tested range.



(a) Training results for binary occupancy triplanes



(b) Training results for triplanes with normal maps and SDF

Figure 5: Training results for both triplane methods

Triplanes generated at inference are qualitatively compared in Figure 5. Panel (a) displays the results from the binary occupancy trained model where smooth predictions are generated that approximate the ground truth shape silhouettes after thresholding. Panel (b) shows results for the model trained on normal maps and SDFs, demonstrating its ability to reconstruct both feature types, with predicted SDFs visually aligning well with the ground truth and predicted normal maps appearing somewhat smoothed. We believe the model encountered difficulty when learning the normal map with SDF

triplanes given the increase in information needed to be learned, suggesting more experimenting with changes to the architecture could be necessary in order to attain desired outcomes.

6 Proposed Extensions

Based on our initial experiments, several avenues for future work emerge. Firstly, the observed difficulty in effectively training the model using the combined Normal Map and Signed Distance Field (SDF) triplane representation suggests a need for further investigation. Exploring architectural modifications to the decoder and different feature fusion techniques could potentially improve the learning of the richer geometric features. Secondly, while the Binary Occupancy model demonstrated promising results, its quantitative performance (Mean Chamfer Distance of 0.200) indicates room for improvement compared to state-of-the-art methods. Further experimentation involving extended training durations, hyperparameter optimization, and potentially exploring more sophisticated data augmentation techniques for the sketch inputs could help bridge this performance gap. Additionally, investigating alternative input representations beyond edge maps, such as rough volumetric sketches or incorporating textual prompts, could broaden the model's applicability as an architectural design tool.

7 Conclusion

This project explored the development of a sketch-based generative 3D model, 3T3D, aimed at facilitating the architectural design process. By adapting a Vision Transformer (ViT) architecture using a pre-trained DINOv2 encoder, we investigated the use of triplanar representations for mapping 2D sketch inputs (derived from edge maps) to 3D geometry. We experimented with two triplane feature encodings: binary occupancy and a combination of normal maps and SDFs, creating a custom dataset leveraging Stable Diffusion XL, TripoSR, and Informative Drawings. Our quantitative evaluations, using Chamfer Distance, showed the binary occupancy model achieved a Mean CD of 0.200, indicating further optimization is needed to reach state-of-the-art performance. Qualitatively, both methods demonstrated potential, though the Normal Map + SDF approach highlighted challenges in learning complex feature representations, suggesting a need for architectural refinement. Future work should focus on optimizing training procedures, exploring architectural modifications to better handle rich geometric features like normals and SDFs, and potentially incorporating more diverse datasets or input modalities to enhance the model's robustness and utility as an intuitive tool for architectural design iteration.

8 Github Repository

Code for the project can be found in our github repository: https://github.com/1gfelton/11-685-Project

References

[1] Caroline Chan, Fredo Durand, and Phillip Isola. Learning to generate line drawings that convey geometry and semantics, 2022.

[2] Kangle Deng, Gengshan Yang, Deva Ramanan, and Jun-Yan Zhu. 3d-aware conditional image synthesis, 2023.

[3] Anqi Mao, Mehryar Mohri, and Yutao Zhong. Cross-entropy loss functions: Theoretical analysis and applications, 2023.

[4] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mahmoud Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Hervé Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. Dinov2: Learning robust visual features without supervision, 2024.

[5] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation, 2019.

[6] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.

[7] Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. Sdxl: Improving latent diffusion models for high-resolution image synthesis, 2023.

[8] J. Ryan Shue, Eric Ryan Chan, Ryan Po, Zachary Ankner, Jiajun Wu, and Gordon Wetzstein. 3d neural field generation using triplane diffusion, 2022.

[9] Florian Spiess, Raphael Waltenspül, and Heiko Schuldt. The sketchfab 3d creative commons collection (s3d3c), 2024.

[10] Dmitry Tochilkin, David Pankratz, Zexiang Liu, Zixuan Huang, Adam Letts, Yangguang Li, Ding Liang, Christian Laforte, Varun Jampani, and Yan-Pei Cao. Triposr: Fast 3d object reconstruction from a single image, 2024.

[11] Dmitry Tochilkin, David Pankratz, Zexiang Liu, Zixuan Huang, Adam Letts, Yangguang Li, Ding Liang, Christian Laforte, Varun Jampani, and Yan-Pei Cao. Triposr: Fast 3d object reconstruction from a single image, 2024.

[12] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes, 2015.

[13] Xinwei Zhuang, Zixun Huang, Wentao Zeng, and Luisa Caldas. Encoding urban ecologies: Automated building archetype generation through self-supervised learning for energy modeling, 2024.

[14] Xinwei Zhuang, Yi Ju, Allen Yang, and Luisa Caldas. Synthesis and generation for 3d architecture volume with generative modeling. *International Journal of Architectural Computing*, 21(2):297–314, 2023.