Extended Abstract

Motivation Computer-Aided Design (CAD) programs are the fundamental building blocks for a variety of scientific and engineering fields that requires precise modeling criteria. However, automating CAD designs have been challenging, because there is a limited amount of data available for training a fully supervised model. In this project, we explore the potential of using Reinforcement Learning to address this issue, by training a CAD agent to reconstruct the CAD command sequence from geometric rewards. By scaling up this approach, we can essentially leverage large-scale 3D or 2D data to train a CAD generator generating diverse assets with the full CAD command sequence.

Method We use Proximal Policy Optimization (PPO) to train a RL agent that predict the CAD commands given an input 3D shape. We develop a factorized policy distribution that is capable of outputting both discrete and continuous actions in order to accommodate the action space of CAD generation. Specifically, we model a categorical distribution over the discrete choices and a Gaussian distribution for the continuous parameters given a specific discrete action choice. We develop a custom policy network architecture to accommodate the above policy factorization. The factorized policy distribution has closed-form standard deviation and entropy terms, making it easy to integrate it into a traditional PPO pipeline. For the rewards, we use a combination of two rewards that both account for the geometric and formatting errors of the constructed CAD program w.r.t. the input shape. Specifically, the geometric rewards measures the intersection-over-union (IOU) between the current CAD reconstruction and the input shape. A higher IOU implies that the two shapes are geometrically close to each other and vice versa. A formatting reward is assigned to punish an episode whenever it fails to convert the current CAD command sequence into a valid CAD shape. This happens when either invalid parameters are predicted or the CAD executions fail. In these cases, a negative reward is added.

Implementation We implement a custom environment that builds the CAD program using the Open Cascade Paviot (2022). For the policy network, we sample points on both the current CAD shape and the ground truth shape and pass them through a PointNet Qi et al. (2016). Together with the current CAD command sequence, the encoded features are passed through the discrete head network to predict logits for the categorical distribution over the discrete actions. Finally, concatenate learned embeddings for each discrete action with the point features and the action sequence to output the continuous distributions for each discrete action choice. The PPO algorithm is implemented on top of TianShou Weng et al. (2022) framework.

Results We conducted overfitting experiments where a specialized agent is fitted to reconstruct one CAD shape using the abovementioned pipeline. We conducted two sets of experiments. The first set uses a toy setting to validate the effectiveness of using RL for CAD modeling, and the second experiment leverages the entirety of the above pipeline to train an RL agent for CAD modeling. The validation experiment was able to output a CAD sequence that reconstructs the rough geometry of the input shape. However, details are still missing, potentially due to the insensitivity of rewards to the details. The full experiment was only able to output correct geometry when the CAD sequence was short. This is potentially due to that we did not initialize our policy network from imitation learning, making the initial exploration phase difficult.

Discussion & Conclusion This project investigates the potential of using RL algorithms for CAD command sequence generation. Specifically, we developed a novel factorized action policy to accommodate the hybrid action space of CAD modeling, as well as designed specific rewards to encourage the RL to learn to reconstruct the input shape. Experimentally, the validation experiment shows the promise in this direction for further investigation. For future steps, we would like to pre-trained the action policy network with supervised training, so that the policy network starts from more informative knowledge about the CAD command sequence. We believe that pre-training would help the performance of RL for CAD generation. Further, we would like to incorporate more semantically informative rewards using other modalities such as images or segmentation masks to encourage the RL agent to fill in the details as well. Curriculum training could also be considered to encourage the RL agent to learn the rough geometry first before attending to the geometric details.

Learning CAD Program Generation using Reinforcement Learning

George Nakayama
Department of Computer Science
Stanford University
w4756677@stanford.edu

Abstract

This project investigates the potential of using RL algorithms for CAD command sequence generation. Specifically, we developed a novel factorized action policy to accommodate the hybrid action space of CAD modeling, as well as designed specific rewards to encourage the RL to learn to reconstruct the input shape. Our experiment demonstrates initial progress towards this direction, with the RL agent capable of reconstructing the 3D shapes with only geometric and format rewards – without using the full CAD command history as training data. By further expanding upon this project, we could enable large-scale training on 3D shape dataset using RL algorithms, alleviating the data bottleneck of CAD models with full modeling history.

1 Introduction

CAD programs refer to computer-aided design programs, which are the fundamental building blocks for a variety of scientific and engineering fields. Specifically, it represents 3D objects through sequences of geometric instructions, commonly referred to as CAD commands, which defines editable geometric components and operations. Despite the emergence of various 3D modeling software (e.g., AutoCAD, SketchUp, Rhino, and FreeCAD), the design workflow persists as a technically challenging and labor-intensive process: it is time-consuming and requires specialized expertise from designers and/or engineers. In the design phase, they use CAD drawings for their precision and ease of editability. During manufacturing, these drawings are converted into constraint-based parameter tables, and for simulation, they yield boundary-representation (B-Rep) data or textual geometry descriptions. While the full design history is not used for downstream applications, current CAD software requires experts to design and modify the model, while the CAD programs need to be frequently updated by communicating with the users. Therefore, it is desirable to develop a toolbox with which the expert, or even the non-expert, can easily design the CAD models by using simple instructions and illustrations to make the ideas in their mind easily come true.

With the advance of machine learning and AI generative models for 3D content, there are many works that generates 3D shapes from text, images, and other user-friendly inputs. However, in contrast to the fast development in 3D generative methods in other shape representations such as point clouds Zeng et al. (2022), voxels Ren et al. (2024), meshes Shen et al. (2024), and implicits Park et al. (2019), CAD program generation achieved limited success. So far, CAD program generation is mainly limited to supervised training using full history CAD programs. While this approach treats CAD programs essentially as text and thereby leverages the success of LLM pre-training, it achieved only limited generation performance primarily due to the limited amount of training data, making the training of large-scale neural networks capable of diverse output infeasible.

The goal of this project is to learn a policy network, which can act as a CAD program generative model, that is able to perform CAD program generation using reinforcement learning. The motivation behind such an approach is to be able to optimize the policy network on shapes without ground truth CAD programs using heuristic rewards, and therefore leverage existing large-scale 3D shape dataset such as ShapeNet Chang et al. (2015) and Objaverse Deitke et al. (2022). Hopefully, this can help the policy network to generate a greater variety of shapes without a groundtruth CAD program.

2 Related Work

CAD generation is a long-standing topic in computer graphics and machine learning. It can mainly be categorized into two sub-directions for research.

BREP-based Shape Generation BREP 3D models are depicted as graphs, incorporating both geometric primitives (e.g., parametric curves and surfaces) and topological primitives (e.g., vertices, edges, and faces) that trim and stitch surface patches to form solid models Xu et al. (2024a). Earliest works focused on BREP classification and segmentation, using a graphical neural network Willis et al. (2021); 10. (2020); Jayaraman et al. (2023), custom convolution kernels Lambourne et al. (2021), and hierarchical graph structures Jones et al. (2022, 2021); Bian et al. (2023) to leverage the graph properties of these shapes.

For generation tasks, previous approaches used predefined template curves and surfaces Sharma et al. (2020); Smirnov et al. (2021); Wang et al. (2022, 2020); Li et al. (2018). Specifically, PolyGen Nash et al. (2020), the pioneer work in this area, uses a pointer network Vinyals et al. (2017) with Transformers Vaswani et al. (2023) to generate n-gon meshes, which can be treated as a special case of BREP shapes with planar faces and straight edges. SolidGen Jayaraman et al. (2023) and BrepGen Xu et al. (2024a) can generate the entire BREP shape. SolidGen Jayaraman et al. (2023) first synthesizes vertices and then constructs them with the edge topology. BrepGen Xu et al. (2024a) progressively denoises the faces, edges, and vertices utilizing Diffusion models Ho et al. (2020). Although B-rep is a direct representation of the boundary of the CAD model, and these generative methods are able to obtain better performance because there is more data in this format; the generated results do not contain the modeling history of the generation, limiting their abilities to perform downstream editing or manipulation of the generated shapes.

CAD Program Generation The second area are methods that try to generate the full modeling history along with the final CAD program.

Existing CAD program generation methods are used for reverse engineering the full CAD program from input point clouds and/or images, text inputs. Point clouds are the most well-studied input modality in CAD reconstruction. The seminal work on point cloud-based CAD reconstruction, DeepCAD by Wu et al. (2021), proposed encoding CAD sketch-and-extrude sequences as special tokens. Beyond that, DeepCAD also proposed the first large-scale dataset of 180k hand-crafted CAD modeled scraped from the OnShape online repository. Subsequent works Chen et al. (2025); Khan et al. (2024a); Xu et al. (2022); Dupont et al. (2024); Ma et al. (2023) adopted the same CAD representation and trained on the same DeepCAD dataset. More recently, CAD-Recode Rukhovich et al. (2024) introduced a paradigm shift by representing CAD models as Python code, providing greater expressiveness and flexibility, and released a new training dataset of approximately 1 million procedurally generated CAD samples. More recently, works Chen et al. (2025); You et al. (2024); Yuan et al. (2024); Wang et al. (2025); Khan et al. (2024b) have explored CAD reconstruction from other input modalities, such as single- or multi-view images and natural language descriptions. These approaches extend the DeepCAD dataset by rendering synthetic views or generating textual captions for existing CAD models. Among them, CADCrafter Chen et al. (2025) proposes an unified framework that handles both single- and multi-view inputs using a latent diffusion model Rombach et al. (2021) to sample from the latent space of DeepCAD. For text-to-CAD generation, Text2CAD Khan et al. (2024b) uses a vision-language model (VLM) to generate captions for CAD programs in the DeepCAD dataset and trains an autoregressive model to predict the corresponding sketch-and-extrude sequences given these text inputs. Finally, with the advance of Multimodal LLMs OpenAI et al. (2024); Liu et al. (2024); Grattafiori et al. (2024), works such as CAD-GPT Kapsalis (2024) and CAD-MLLM Xu et al. (2024b) takes in multimodal input conditioning to reconstruct the desired CAD programs. They both fine-tune existing Multimodal LLMs on DeepCAD programs with multimodal annotations.

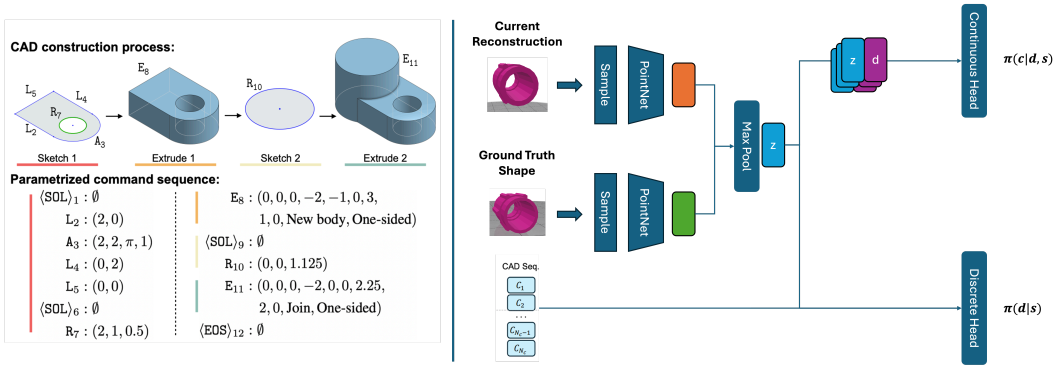


Figure 1: **Method Overview.** (Left) Example of a CAD construction process adopted from Deep-CAD Wu et al. (2021) using the sketch-extrude paradigm. (Right) The proposed factorized policy network that separates the discrete actions ($\pi(d|s)$) and the continuous actions ($\pi(c|d,s)$).

A pioneering work in this area is DeepCAD Wu et al. (2021). In this work, the authors proposed to focus on the type of CAD programs built solely with sequences of sketch-extrude operations, and proposed a dataset containing CAD programs with modeling history. Building on the task setup of DeepCAD, many works Xu et al. (2022); Ma et al. (2023); Khan et al. (2024a); Chen et al. (2025); Dupont et al. (2024); You et al. (2024); Ma et al. (2024) improves upon it by extending the method to more tasks such as CAD prediction from images You et al. (2024), voxels Lambourne et al. (2022), and texts Khan et al. (2024c); Wu et al. (2024). However, all of these works rely on full supervision using the DeepCAD dataset, fundamentally limiting the scale and variety of their generated CAD programs. In our work, we will also focus on generating CAD programs in this paradigm. However, in contrast to these methods, which require supervision on the modeling history, reinforcement learning allows us to update the policy network using 3D shapes without CAD modeling history.

Past literature also tried to use unsupervised learning approaches to directly generate CAD programs in the sketch-extrude paradigm Li et al. (2024); Jones et al. (2023). However, these approaches typically only allows for a limited sequence length, and thereby restricting the representation power of these methods. For us, however, the policy network can generate the operations in an auto-regressive manner; therefore, in theory, achieve infinite-length CAD operations.

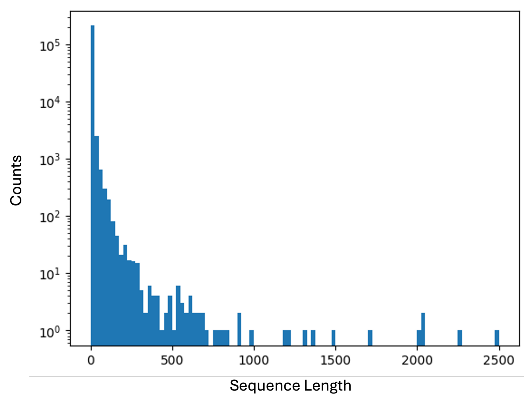
3 Method

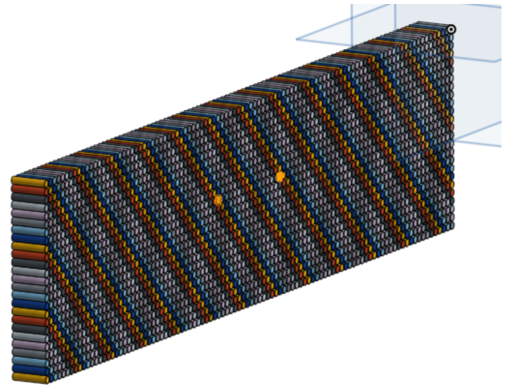
We detail our approach to using reinforcement learning for CAD program generation below. Specifically, Sec. 3.1 details the sketch-extruce CAD construction process we adopt for this paper. In Sec. 3.2 we present our data preparation process together with statistics of the datasets. Finally, in Sec. 3.3 and Sec. 3.4, we outline how we adopt Proximal Policy Optimization (PPO) Schulman et al. (2017) for CAD program generation as well as the rewards we use to training PPO.

3.1 CAD Representation for Neural Networks

CAD programs consist of two levels of representation. When users are designing a CAD model, they will perform a sequence of operations in a CAD software to create a solid shape. Typically, different CAD software contains a different set of operations. For example, users may draw a set of closed curves and lines on a 2D plane, and then perform extrusions on faces formed by the curves to convert them into 3D shapes. Other common operations include sweeping, revolving, and lofting, all of which define 3D shapes using 2D faces on the sketch. Additionally, different 3D primitives created this way are further processed by other operations such as a boolean union, difference, or intersection to create the final desired 3D model (see Fig. 1 (left)). We refer to such a specification as a CAD command sequence.

As the users are creating the CAD models using a sequence of commands, the CAD software builds a kernel representation of the CAD program, widely known as the boundary representation (or BREP) Xu et al. (2024a); Lambourne et al. (2021). BREP describes a solid purely by the topology and geometry of its outer shell: vertices store points, edges link the vertices following pre-defined





(a) Sequence Length Histogram of the DeepCAD Dataset.

(b) The CAD program with the longest sequence length in the DeepCAD dataset.

Figure 2: The DeepCAD Dataset

curves, and faces are formed by patching together edge loops on analytic or spline surfaces. While BREP is usually the output format for standard industry software, shapes represented this way cannot be as easily edited by casual users as the CAD command sequence.

In this work, we aim for a generative model of CAD command sequences. Specifically, following prior works Wu et al. (2021); Xu et al. (2022), we adopt the sketch-extrude paradigm for CAD modeling. While previous works use full supervised learning to learn how to generate CAD programs from human demonstrations, in this project we explore the usage of reinforcement learning to automatically discover CAD command sequence from 3D geometric rewards.

3.2 Data Preparation

This project requires both a CAD dataset with a full CAD modeling command sequence. To this end, we will use the DeepCAD dataset Wu et al. (2021), which contains around 120k CAD programs from the OnShape Repository, filtered to only contain sketch-extrude types of operation. The sequence length of shapes in the dataset varies, but with most of the shapes ranging between 1-4 operations in total. See Fig. 2 for the sequence length statistic (left) as well as the CAD program in the DeepCAD dataset with the longest sequence length (right).

DeepCAD dataset consists mostly of mechanical parts designed for engineering purposes. To increase the diversity of the data shapes, we also consider other datasets. While we did not have time to explore other datasets besides the DeepCAD dataset, to train the RL policy on shapes without groundtruth CAD operations, we also consider adopting the large-scale ShapeNet dataset Chang et al. (2015), which contains clean 3D models with manually verified category and alignment annotations. It covers 55 common object categories with about 51,300 unique 3D models. ShapeNet has been used for many 3D machine learning tasks due to its combination of variety, cleanness, and annotation richness. Thus, it will be a good starting point for this project to test the feasibility of fine-tuning the policy network for the generation of more complex shapes.

3.3 Factorized Hybrid Action Policy for PPO

After processing the DeepCAD dataset, we delineate how we design our policy network and adopt PPO for CAD generation with reinforcement learning.

As discussed in previous sections, modeling a CAD program requires both discrete actions, such as choosing types of curves to draw (lines, arcs, and circles), and the shape boolean operations to use (union, difference, and intersection), and continuous actions, including the parameters for each CAD modeling command. To this end, we require our action policy network to be able to output both continuous actions and discrete actions.

Below, we formalize the above discussion. Let D be the random variable corresponding to the discrete actions and C be the r.v. corresponding to the continuous actions. Given state s, we wish to model the action distribution $\pi_\theta\left(C, D|s\right)$, that is, the probability of choosing one continuous and discrete action given the current state s. Now, because the continuous actions in CAD programs are only determined after the discrete choices are made, we can factorize the action policy to

$$\pi_\theta\left(C, D|s\right) = \pi_\theta\left(C|D, s\right)\pi_\theta\left(D|s\right). \tag{1}$$

Compared to the left hand side, the right hand side's factorized distribution allows us to separate the prediction of the discrete distribution $\pi_\theta\left(D|s\right)$ and the continuous distributions $\pi_\theta\left(C|D, s\right)$ given a discrete action choice D. Comparing with other approaches to modeling a hybrid action space, such as discretizing the continuous space into fixed bins, this approach losslessly retains the full precision of the continuous parameters. Moreover, while other appaoches such as Hybrid PPO Fan et al. (2019) require a different algorithm to handle hybrid action sequences, the factorized policy can be plugged into any RL algorithms without modifications.

To model the factorized policy, we construct a custom policy network that respects the probability diagram. As shown on the right side of Fig. 1, we first use the information from the state s to predict the discrete action via the discrete head. Then, for each discrete action, we predict a separate continuous distribution (Gaussian in our experiments) that parametrizes $\pi_\theta\left(C|D, s\right)$. Notice that it is crucial to predict one distribution for each discrete action D, since the continuous part of the factorization given in Eq. 1 depends on the discrete action D.

3.4 Reward Design

How to design a reward function so that it can give the policy network meaningful optimization signal becomes one of the keys to the success of PPO training. We adopt two reward functions in our training. The first reward is a geometric one, where we encourage the reconstructed CAD sequence to be as close as possible to the input ground truth shape. To this end, we use intersection-over-union (IOU) as the geometric reward. Specifically, given two shapes S, S', the IOU reward is defined as

$$\mathrm{IOU}(S, S') = \frac{|S \cap S'|}{|S \cup S'|} \tag{2}$$

where $|S \cap S'|$ is the volume of the intersection of S and S' and $|S \cup S'|$ is the volume of their union. Notice that when $S = S'$, their IOU score will be 1, as their intersection and their union will equal exactly. In general, a higher value of IOU indicates a better similarity between the two shapes. Thus, PPO that maximizes this reward will encourage the reconstructed CAD sequence to be as geometrically close as possible to the target.

The second reward term we adopt is similar to the format reward used when training for LLM's reasoning capability DeepSeek-AI et al. (2025). Specifically, because the CAD reconstruction from a sequence of CAD commands sometime will fail to execute, we want to punish the agent from outputting such a command sequence. Therefore, we assign a negative reward $R_{\text{fail}}(S) = -10$ whenever the CAD execution of S fails.

In total, the reward at every environment step consists of the sum

$$R(S, S') = 100 \cdot \mathrm{IOU}(S, S') + R_{\text{fail}}(S). \tag{3}$$

4 Experimental Setup

Following what is described in Sec. 3, we implement a PPO training for CAD generation. Specifically, we implement a custom gym environment that builds the CAD program using the Open Cascade Paviot (2022). At every environment step call, the environment will build the CAD model from the current CAD command sequence and evaluate the aforementioned rewards.

As inputs to the policy network, the CAD shapes will first sample points from their surfaces, and be passed through a PointNet Qi et al. (2016) to be encoded as a latent feature. Together with the current CAD command sequence, the encoded feature are passed through the discrete head network parameterized as a 3 layer MLP to predict logits for the categorical distribution over the discrete actions. Finally, concatenate learned embeddings for each discrete action with the point features

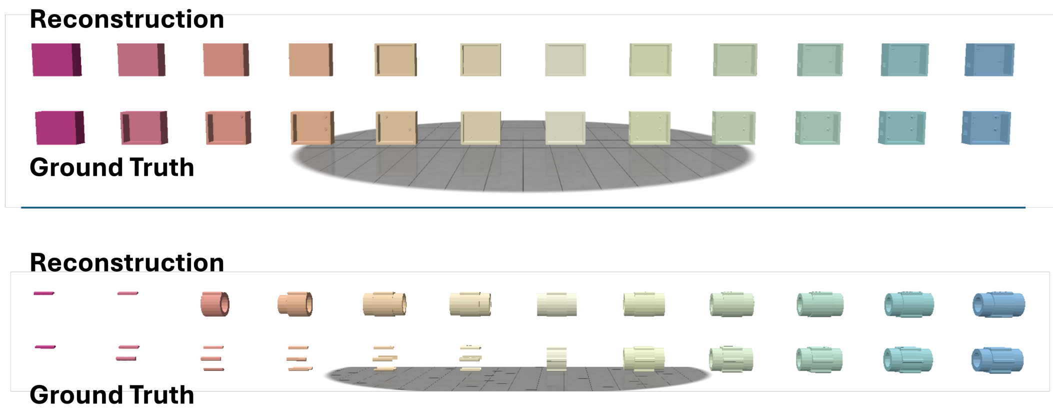


Figure 3: Visualization of the initial results.

and the action sequence to output the means of the continuous actions for each discrete choice. The variance of each continuous distribution is a learnable variable.

The PPO algorithm is implemented on top of TianShou Weng et al. (2022) framework, with a buffer size of 750 steps and a constant learning rate of $1e-3$. We use a clip ratio of 0.2 and an entropy regularization weight of 0.05. We also use advantage normalization and gradient norm clipping to stabilize the optimization.

5 Results

Due to time constraint, we only conducted overfitting experiments where a specialized agent is fitted to reconstruct one CAD shape using the abovementioned pipeline. We conducted two sets of experiments. The first set uses a toy setting to validate the effectiveness of using RL for CAD modeling, and the second experiment leverages the entirety of the above pipeline to train an RL agent for CAD modeling.

5.1 Validation Experiment

The initial experiment was conducted by training an RL algorithm to predict only the discrete actions. To make the RL task easier, we manually created a sequence of actions to take for the agent to reconstruct the entire CAD program. This way, the difficulty of predicting continuous actions is eliminated for now. The agent is trained with Chamfer Distance only w.r.t. to the target CAD program. In this way, supervised pre-training is not used.

Fig. 3 shows the results of a single CAD program overfitting under this setting. Notice that the reconstruction is able to get the overall geometry of the CAD program. However, it misses details such as holes on the side or protrusions at the bottom of the square on the top row, and the outer edges in the bottom row. This is due to the insensitivity of the reward function, a.k.a., Chamfer distance, w.r.t. to small geometric details. However, this do provide a promising starting point for using PPO for CAD modeling.

5.2 Full Experiment

We use the pipeline described in Sec. 3.3 and Sec. 3.4 to train a PPO algorithm to predict the CAD command sequence. Compared to the validation experiment above, the full experiment requires the agent to predict both continuous and discrete actions correctly for the final CAD model to be similar to the input shape. Thus, this setting is much harder compared to the previous one, which resulted in less performant results.

Fig. 4 shows a successful case in which the agent is able to reconstruct the input shape relatively well. In this case, the agent learned to place the primitives correctly at the right orientations and locations,

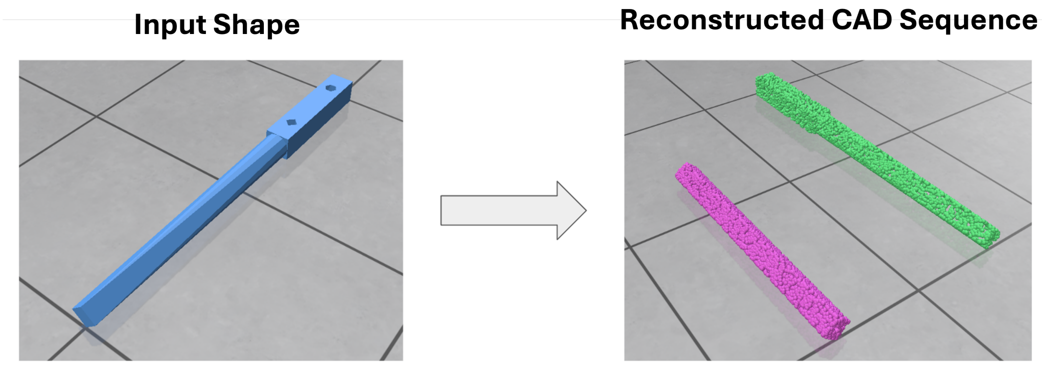


Figure 4: Visualization of results using the full pipeline in Sec. 3.

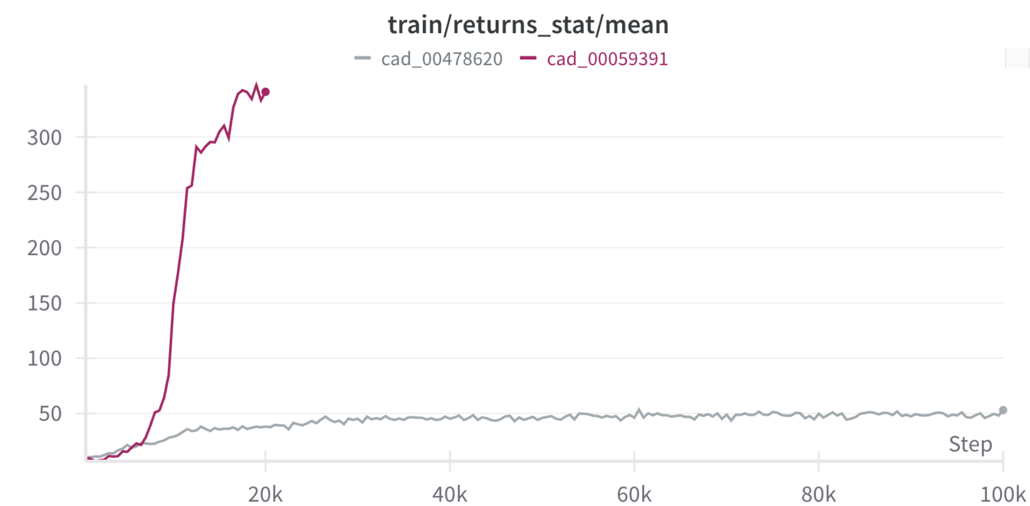


Figure 5: Reward comparison of two runs of different shapes.

as well as choose the correct Boolean operations. However, the results on more complicated shapes the involves either more steps for constructions or negative Boolean operations such as cutting or differencing, faced convergence issues. Specifically, as shown in Fig. 5, the PPO training is stuck at local minima for certain shapes (e.g., shape 00478620 in the figure), because the model did not learn to take the correct step and had a collapsing action distribution. This might be due to it requirement of using a shape intersection shape to obtain the final shape, which might be hard for the RL algorithm to learn, as the initial attempts of doing so would lead to negative rewards. Further investigation is needed in terms of the choice of hyperparamters as well as the RL algorithm for making the convergence better.

6 Discussion & Conclusion

This project investigates the potential of using RL algorithms for CAD command sequence generation. Specifically, we developed a novel factorized action policy to accommodate the hybrid action space of CAD modeling, as well as designed specific rewards to encourage the RL to learn to reconstruct the input shape. While the results are not as high quality as we had wished for, the validation experiment does show the promise in this direction for further investigation. For future steps, we would like to pre-trained the action policy network with supervised training, so that the policy network starts from more informative knowledge about the CAD command sequence. We believe that pre-training would help the performance of RL for CAD generation. Further, as noted in the validation experiment, the geometric reward currently does not reflect the accuracy of small geometric details. Thus, we would like to incorporate more semantically informative rewards using other modalities such as images or segmentation masks to encourage the RL agent to fill in the details as well. Curriculum training could also be considered to encourage the RL agent to learn the rough geometry first before attending to the geometric details.

7 Team Contributions

- **George Nakayama:** Sole author of the project.

Changes from Proposal Due to time and computational budget constraints, we did not have time to implement supervised fine-tuning as described in the proposal. We also did not get a chance to test on ShapeNet and other more large-scale datasets, as we have only conducted overfitting experiments.

References

2020. *Graph Representation of 3D CAD Models for Machining Feature Recognition With Deep Learning*. International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, Vol. Volume 11A: 46th Design Automation Conference (DAC). `https://doi.org/10.1115/DETC2020-22355` arXiv:https://asmedigitalcollection.asme.org/IDETC-CIE/proceedings-pdf/IDETC-CIE2020/84003/V11AT11A003/6587023/v11at11a003-detc2020-22355.pdf

Shijie Bian, Daniele Grandi, Tianyang Liu, Pradeep Kumar Jayaraman, Karl Willis, Elliot Sadler, Bodia Borijin, Thomas Lu, Richard Otis, Nhut Ho, and Bingbing Li. 2023. HG-CAD: Hierarchical Graph Learning for Material Prediction and Recommendation in Computer-Aided Design. *Journal of Computing and Information Science in Engineering* 24, 1 (10 2023), 011007. `https://doi.org/10.1115/1.4063226` arXiv:https://asmedigitalcollection.asme.org/computingengineering/article-pdf/24/1/011007/7047936/jcise_24_1_011007.pdf

Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. 2015. *ShapeNet: An Information-Rich 3D Model Repository*. Technical Report arXiv:1512.03012 [cs.GR]. Stanford University — Princeton University — Toyota Technological Institute at Chicago.

Cheng Chen, Jiacheng Wei, Tianrun Chen, Chi Zhang, Xiaofeng Yang, Shangzhan Zhang, Bingchen Yang, Chuan-Sheng Foo, Guosheng Lin, Qixing Huang, and Fayao Liu. 2025. CADCrafter: Generating Computer-Aided Design Models from Unconstrained Images. arXiv:2504.04753 [cs.CV] `https://arxiv.org/abs/2504.04753`

DeepSeek-AI, Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Daya Guo, Dejian Yang, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Haowei Zhang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Li, Hui Qu, J. L. Cai, Jian Liang, Jianzhong Guo, Jiaqi Ni, Jiashi Li, Jiawei Wang, Jin Chen, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, Junxiao Song, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Lei Xu, Leyi Xia, Liang Zhao, Litong Wang, Liyue Zhang, Meng Li, Miaojun Wang, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Mingming Li, Ning Tian, Panpan Huang, Peiyi Wang, Peng Zhang, Qiancheng Wang, Qihao Zhu, Qinyu Chen, Qiushi Du, R. J. Chen, R. L. Jin, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, Runxin Xu, Ruoyu Zhang, Ruyi Chen, S. S. Li, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shaoqing Wu, Shengfeng Ye, Shengfeng Ye, Shirong Ma, Shiyu Wang, Shuang Zhou, Shuiping Yu, Shunfeng Zhou, Shuting Pan, T. Wang, Tao Yun, Tian Pei, Tianyu Sun, W. L. Xiao, Wangding Zeng, Wanjia Zhao, Wei An, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, X. Q. Li, Xiangyue Jin, Xianzu Wang, Xiao Bi, Xiaodong Liu, Xiaohan Wang, Xiaojin Shen, Xiaokang Chen, Xiaokang Zhang, Xiaosha Chen, Xiaotao Nie, Xiaowen Sun, Xiaoxiang Wang, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xingkai Yu, Xinnan Song, Xinxia Shan, Xinyi Zhou, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, Y. K. Li, Y. Q. Wang, Y. X. Wei, Y. X. Zhu, Yang Zhang, Yanhong Xu, Yanhong Xu, Yanping

Huang, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Li, Yaohui Wang, Yi Yu, Yi Zheng, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Ying Tang, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yu Wu, Yuan Ou, Yuchen Zhu, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yukun Zha, Yunfan Xiong, Yunxian Ma, Yuting Yan, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Z. F. Wu, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhen Huang, Zhen Zhang, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhibin Gou, Zhicheng Ma, Zhigang Yan, Zhihong Shao, Zhipeng Xu, Zhiyu Wu, Zhongyu Zhang, Zhuoshu Li, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Ziyi Gao, and Zizheng Pan. 2025. DeepSeek-V3 Technical Report. arXiv:2412.19437 [cs.CL] `https://arxiv.org/abs/2412.19437`

Matt Deitke, Dustin Schwenk, Jordi Salvador, Luca Weihs, Oscar Michel, Eli VanderBilt, Ludwig Schmidt, Kiana Ehsani, Aniruddha Kembhavi, and Ali Farhadi. 2022. Objaverse: A Universe of Annotated 3D Objects. *arXiv preprint arXiv:2212.08051* (2022).

Elona Dupont, Kseniya Cherenkova, Dimitrios Mallis, Gleb Gusev, Anis Kacem, and Djamila Aouada. 2024. TransCAD: A Hierarchical Transformer for CAD Sequence Inference from Point Clouds. arXiv:2407.12702 [cs.CV] `https://arxiv.org/abs/2407.12702`

Zhou Fan, Rui Su, Weinan Zhang, and Yong Yu. 2019. Hybrid Actor-Critic Reinforcement Learning in Parameterized Action Space. arXiv:1903.01344 [cs.LG] `https://arxiv.org/abs/1903.01344`

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurelien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Roziere, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, Danny Wyatt, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Francisco Guzmán, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Govind Thattai, Graeme Nail, Gregoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel Kloumann, Ishan Misra, Ivan Evtimov, Jack Zhang, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Karthik Prasad, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, Khalid El-Arini, Krithika Iyer, Kshitiz Malik, Kuenley Chiu, Kunal Bhalla, Kushal Lakhotia, Lauren Rantala-Yeary, Laurens van der Maaten, Lawrence Chen, Liang Tan, Liz Jenkins, Louis Martin, Lovish Madaan, Lubo Malo, Lukas Blecher, Lukas Landzaat, Luke de Oliveira, Madeline Muzzi, Mahesh Pasupuleti, Mannat Singh, Manohar Paluri, Marcin Kardas, Maria Tsimpoukelli, Mathew Oldham, Mathieu Rita, Maya Pavlova, Melanie Kambadur, Mike Lewis, Min Si, Mitesh Kumar Singh, Mona Hassan, Naman Goyal, Narjes Torabi, Nikolay Bashlykov, Nikolay Bogoychev, Niladri Chatterji, Ning Zhang, Olivier Duchenne, Onur Çelebi, Patrick Alrassy, Pengchuan Zhang, Pengwei Li, Petar Vasic, Peter Weng, Prajjwal Bhargava, Pratik Dubal, Praveen Krishnan, Punit Singh Koura, Puxin Xu, Qing He, Qingxiao Dong, Ragavan Srinivasan, Raj Ganapathy, Ramon Calderer, Ricardo Silveira Cabral, Robert Stojnic, Roberta Raileanu, Rohan Maheswari, Rohit Girdhar, Rohit Patel, Romain Sauvestre, Ronnie Polidoro, Roshan Sumbaly, Ross Taylor, Ruan Silva, Rui Hou, Rui Wang, Saghar Hosseini, Sahana Chennabasappa, Sanjay Singh, Sean Bell, Seohyun Sonia Kim, Sergey Edunov, Shaoliang Nie, Sharan Narang, Sharath Raparthy, Sheng Shen, Shengye Wan, Shruti Bhosale, Shun Zhang, Simon Vandenhende, Soumya Batra, Spencer Whitman, Sten Sootla, Stephane Collot, Suchin Gururangan, Sydney Borodinsky, Tamar Herman, Tara Fowler, Tarek Sheasha, Thomas Georgiou, Thomas Scialom, Tobias Speckbacher, Todor Mihaylov, Tong Xiao, Ujjwal Karn, Vedanuj Goswami, Vibhor Gupta, Vignesh Ramanathan, Viktor Kerkez, Vincent Gonguet, Virginie Do, Vish Vogeti, Vítor Albiero, Vladan Petrovic, Weiwei Chu, Wenhan Xiong, Wenyin Fu, Whitney Meers, Xavier Martinet, Xiaodong Wang, Xiaofang Wang, Xiaoqing Ellen Tan, Xide Xia, Xinfeng Xie, Xuchao Jia, Xuewei Wang, Yaelle Goldschlag, Yashesh Gaur, Yasmine Babaei, Yi

Wen, Yiwen Song, Yuchen Zhang, Yue Li, Yuning Mao, Zacharie Delpierre Coudert, Zheng Yan, Zhengxing Chen, Zoe Papakipos, Aaditya Singh, Aayushi Srivastava, Abha Jain, Adam Kelsey, Adam Shajnfeld, Adithya Gangidi, Adolfo Victoria, Ahuva Goldstand, Ajay Menon, Ajay Sharma, Alex Boesenberg, Alexei Baevski, Allie Feinstein, Amanda Kallet, Amit Sangani, Amos Teo, Anam Yunus, Andrei Lupu, Andres Alvarado, Andrew Caples, Andrew Gu, Andrew Ho, Andrew Poulton, Andrew Ryan, Ankit Ramchandani, Annie Dong, Annie Franco, Anuj Goyal, Aparajita Saraf, Arkabandhu Chowdhury, Ashley Gabriel, Ashwin Bharambe, Assaf Eisenman, Azadeh Yazdan, Beau James, Ben Maurer, Benjamin Leonhardi, Bernie Huang, Beth Loyd, Beto De Paola, Bhargavi Paranjape, Bing Liu, Bo Wu, Boyu Ni, Braden Hancock, Bram Wasti, Brandon Spence, Brani Stojkovic, Brian Gamido, Britt Montalvo, Carl Parker, Carly Burton, Catalina Mejia, Ce Liu, Changhan Wang, Changkyu Kim, Chao Zhou, Chester Hu, Ching-Hsiang Chu, Chris Cai, Chris Tindal, Christoph Feichtenhofer, Cynthia Gao, Damon Civin, Dana Beaty, Daniel Kreymer, Daniel Li, David Adkins, David Xu, Davide Testuggine, Delia David, Devi Parikh, Diana Liskovich, Didem Foss, Dingkang Wang, Duc Le, Dustin Holland, Edward Dowling, Eissa Jamil, Elaine Montgomery, Eleonora Presani, Emily Hahn, Emily Wood, Eric-Tuan Le, Erik Brinkman, Esteban Arcaute, Evan Dunbar, Evan Smothers, Fei Sun, Felix Kreuk, Feng Tian, Filippos Kokkinos, Firat Ozgenel, Francesco Caggioni, Frank Kanayet, Frank Seide, Gabriela Medina Florez, Gabriella Schwarz, Gada Badeer, Georgia Swee, Gil Halpern, Grant Herman, Grigory Sizov, Guangyi, Zhang, Guna Lakshminarayanan, Hakan Inan, Hamid Shojanazeri, Han Zou, Hannah Wang, Hanwen Zha, Haroun Habeeb, Harrison Rudolph, Helen Suk, Henry Aspegren, Hunter Goldman, Hongyuan Zhan, Ibrahim Damlaj, Igor Molybog, Igor Tufanov, Ilias Leontiadis, Irina-Elena Veliche, Itai Gat, Jake Weissman, James Geboski, James Kohli, Janice Lam, Japhet Asher, Jean-Baptiste Gaya, Jeff Marcus, Jeff Tang, Jennifer Chan, Jenny Zhen, Jeremy Reizenstein, Jeremy Teboul, Jessica Zhong, Jian Jin, Jingyi Yang, Joe Cummings, Jon Carvill, Jon Shepard, Jonathan McPhie, Jonathan Torres, Josh Ginsburg, Junjie Wang, Kai Wu, Kam Hou U, Karan Saxena, Kartikay Khandelwal, Katayoun Zand, Kathy Matosich, Kaushik Veeraraghavan, Kelly Michelena, Keqian Li, Kiran Jagadeesh, Kun Huang, Kunal Chawla, Kyle Huang, Lailin Chen, Lakshya Garg, Lavender A, Leandro Silva, Lee Bell, Lei Zhang, Liangpeng Guo, Licheng Yu, Liron Moshkovich, Luca Wehrstedt, Madian Khabsa, Manav Avalani, Manish Bhatt, Martynas Mankus, Matan Hasson, Matthew Lennie, Matthias Reso, Maxim Groshev, Maxim Naumov, Maya Lathi, Meghan Keneally, Miao Liu, Michael L. Seltzer, Michal Valko, Michelle Restrepo, Mihir Patel, Mik Vyatskov, Mikayel Samvelyan, Mike Clark, Mike Macey, Mike Wang, Miquel Jubert Hermoso, Mo Metanat, Mohammad Rastegari, Munish Bansal, Nandhini Santhanam, Natascha Parks, Natasha White, Navyata Bawa, Nayan Singhal, Nick Egebo, Nicolas Usunier, Nikhil Mehta, Nikolay Pavlovich Laptev, Ning Dong, Norman Cheng, Oleg Chernoguz, Olivia Hart, Omkar Salpekar, Ozlem Kalinli, Parkin Kent, Parth Parekh, Paul Saab, Pavan Balaji, Pedro Rittner, Philip Bontrager, Pierre Roux, Piotr Dollar, Polina Zvyagina, Prashant Ratanchandani, Pritish Yuvraj, Qian Liang, Rachad Alao, Rachel Rodriguez, Rafi Ayub, Raghotham Murthy, Raghu Nayani, Rahul Mitra, Rangaprabhu Parthasarathy, Raymond Li, Rebekkah Hogan, Robin Battey, Rocky Wang, Russ Howes, Ruty Rinott, Sachin Mehta, Sachin Siby, Sai Jayesh Bondu, Samyak Datta, Sara Chugh, Sara Hunt, Sargun Dhillon, Sasha Sidorov, Satadru Pan, Saurabh Mahajan, Saurabh Verma, Seiji Yamamoto, Sharadh Ramaswamy, Shaun Lindsay, Shaun Lindsay, Sheng Feng, Shenghao Lin, Shengxin Cindy Zha, Shishir Patil, Shiva Shankar, Shuqiang Zhang, Shuqiang Zhang, Sinong Wang, Sneha Agarwal, Soji Sajuyigbe, Soumith Chintala, Stephanie Max, Stephen Chen, Steve Kehoe, Steve Satterfield, Sudarshan Govindaprasad, Sumit Gupta, Summer Deng, Sungmin Cho, Sunny Virk, Suraj Subramanian, Sy Choudhury, Sydney Goldman, Tal Remez, Tamar Glaser, Tamara Best, Thilo Koehler, Thomas Robinson, Tianhe Li, Tianjun Zhang, Tim Matthews, Timothy Chou, Tzook Shaked, Varun Vontimitta, Victoria Ajayi, Victoria Montanez, Vijai Mohan, Vinay Satish Kumar, Vishal Mangla, Vlad Ionescu, Vlad Poenaru, Vlad Tiberiu Mihailescu, Vladimir Ivanov, Wei Li, Wenchen Wang, Wenwen Jiang, Wes Bouaziz, Will Constable, Xiaocheng Tang, Xiaojian Wu, Xiaolan Wang, Xilun Wu, Xinbo Gao, Yaniv Kleinman, Yanjun Chen, Ye Hu, Ye Jia, Ye Qi, Yenda Li, Yilin Zhang, Ying Zhang, Yossi Adi, Youngjin Nam, Yu, Wang, Yu Zhao, Yuchen Hao, Yundi Qian, Yunlu Li, Yuzi He, Zach Rait, Zachary DeVito, Zef Rosnbrick, Zhaoduo Wen, Zhenyu Yang, Zhiwei Zhao, and Zhiyu Ma. 2024. The Llama 3 Herd of Models. arXiv:2407.21783 [cs.AI] `https://arxiv.org/abs/2407.21783`

Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising diffusion probabilistic models. In *Proceedings of the 34th International Conference on Neural Information Processing Systems* (Vancouver, BC, Canada) *(NIPS '20)*. Curran Associates Inc., Red Hook, NY, USA, Article 574, 12 pages.

Pradeep Kumar Jayaraman, Joseph G. Lambourne, Nishkrit Desai, Karl D. D. Willis, Aditya Sanghi, and Nigel J. W. Morris. 2023. SolidGen: An Autoregressive Model for Direct B-rep Synthesis. arXiv:2203.13944 [cs.LG] https://arxiv.org/abs/2203.13944

Benjamin Jones, Dalton Hildreth, Duowen Chen, Ilya Baran, Vladimir G. Kim, and Adriana Schulz. 2021. AutoMate: a dataset and learning approach for automatic mating of CAD assemblies. *ACM Trans. Graph.* 40, 6, Article 227 (Dec. 2021), 18 pages. https://doi.org/10.1145/3478513.3480562

Benjamin T. Jones, Michael Hu, Vladimir G. Kim, and Adriana Schulz. 2022. Self-Supervised Representation Learning for CAD. arXiv:2210.10807 [cs.CV] https://arxiv.org/abs/2210.10807

Benjamin T. Jones, Michael Hu, Milin Kodnongbua, Vladimir G. Kim, and Adriana Schulz. 2023. Self-Supervised Representation Learning for CAD. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 21327–21336.

Timo Kapsalis. 2024. CADgpt: Harnessing Natural Language Processing for 3D Modelling to Enhance Computer-Aided Design Workflows. arXiv:2401.05476 [cs.HC] https://arxiv.org/abs/2401.05476

Mohammad Sadil Khan, Elona Dupont, Sk Aziz Ali, Kseniya Cherenkova, Anis Kacem, and Djamila Aouada. 2024a. CAD-SIGNet: CAD Language Inference from Point Clouds using Layer-wise Sketch Instance Guided Attention. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 4713–4722.

Mohammad Sadil Khan, Sankalp Sinha, Talha Uddin Sheikh, Didier Stricker, Sk Aziz Ali, and Muhammad Zeshan Afzal. 2024b. Text2CAD: Generating Sequential CAD Models from Beginner-to-Expert Level Text Prompts. arXiv:2409.17106 [cs.CV] https://arxiv.org/abs/2409.17106

Mohammad Sadil Khan, Sankalp Sinha, Sheikh Talha Uddin, Didier Stricker, Sk Aziz Ali, and Muhammad Zeshan Afzal. 2024c. Text2CAD: Generating Sequential CAD Designs from Beginner-to-Expert Level Text Prompts. In *Advances in Neural Information Processing Systems*, Vol. 37. Curran Associates, Inc., 7552–7579. https://proceedings.neurips.cc/paper_files/paper/2024/file/0e5b96f97c1813bb75f6c28532c2ecc7-Paper-Conference.pdf

Joseph George Lambourne, Karl Willis, Pradeep Kumar Jayaraman, Longfei Zhang, Aditya Sanghi, and Kamal Rahimi Malekshan. 2022. Reconstructing editable prismatic CAD from rounded voxel models. In *SIGGRAPH Asia 2022 Conference Papers* (Daegu, Republic of Korea) *(SA '22)*. Association for Computing Machinery, New York, NY, USA, Article 53, 9 pages. https://doi.org/10.1145/3550469.3555424

Joseph G. Lambourne, Karl D.D. Willis, Pradeep Kumar Jayaraman, Aditya Sanghi, Peter Meltzer, and Hooman Shayani. 2021. BRepNet: A Topological Message Passing System for Solid Models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 12773–12782.

Lingxiao Li, Minhyuk Sung, Anastasia Dubrovina, L. Yi, and Leonidas J. Guibas. 2018. Supervised Fitting of Geometric Primitives to 3D Point Clouds. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2018), 2647–2655. https://api.semanticscholar.org/CorpusID:53715802

Pu Li, Jianwei Guo, Huibin Li, Bedrich Benes, and Dong-Ming Yan. 2024. SfmCAD: Unsupervised CAD Reconstruction by Learning Sketch-based Feature Modeling Operations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 4671–4680.

Haotian Liu, Chunyuan Li, Yuheng Li, Bo Li, Yuanhan Zhang, Sheng Shen, and Yong Jae Lee. 2024. LLaVA-NeXT: Improved reasoning, OCR, and world knowledge. https://llava-vl.github.io/blog/2024-01-30-llava-next/

Weijian Ma, Shuaiqi Chen, Yunzhong Lou, Xueyang Li, and Xiangdong Zhou. 2024. Draw Step by Step: Reconstructing CAD Construction Sequences from Point Clouds via Multimodal Diffusion.. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 27154–27163.

Weijian Ma, Minyang Xu, Xueyang Li, and Xiangdong Zhou. 2023. MultiCAD: Contrastive Representation Learning for Multi-modal 3D Computer-Aided Design Models. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management* (Birmingham, United Kingdom) *(CIKM '23)*. Association for Computing Machinery, New York, NY, USA, 1766–1776. https://doi.org/10.1145/3583780.3614982

Charlie Nash, Yaroslav Ganin, S. M. Ali Eslami, and Peter W. Battaglia. 2020. PolyGen: An Autoregressive Generative Model of 3D Meshes. *ICML* (2020).

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, Irwan Bello, Jake Berdine, Gabriel Bernadett-Shapiro, Christopher Berner, Lenny Bogdonoff, Oleg Boiko, Madelaine Boyd, Anna-Luisa Brakman, Greg Brockman, Tim Brooks, Miles Brundage, Kevin Button, Trevor Cai, Rosie Campbell, Andrew Cann, Brittany Carey, Chelsea Carlson, Rory Carmichael, Brooke Chan, Che Chang, Fotis Chantzis, Derek Chen, Sully Chen, Ruby Chen, Jason Chen, Mark Chen, Ben Chess, Chester Cho, Casey Chu, Hyung Won Chung, Dave Cummings, Jeremiah Currier, Yunxing Dai, Cory Decareaux, Thomas Degry, Noah Deutsch, Damien Deville, Arka Dhar, David Dohan, Steve Dowling, Sheila Dunning, Adrien Ecoffet, Atty Eleti, Tyna Eloundou, David Farhi, Liam Fedus, Niko Felix, Simón Posada Fishman, Juston Forte, Isabella Fulford, Leo Gao, Elie Georges, Christian Gibson, Vik Goel, Tarun Gogineni, Gabriel Goh, Rapha Gontijo-Lopes, Jonathan Gordon, Morgan Grafstein, Scott Gray, Ryan Greene, Joshua Gross, Shixiang Shane Gu, Yufei Guo, Chris Hallacy, Jesse Han, Jeff Harris, Yuchen He, Mike Heaton, Johannes Heidecke, Chris Hesse, Alan Hickey, Wade Hickey, Peter Hoeschele, Brandon Houghton, Kenny Hsu, Shengli Hu, Xin Hu, Joost Huizinga, Shantanu Jain, Shawn Jain, Joanne Jang, Angela Jiang, Roger Jiang, Haozhun Jin, Denny Jin, Shino Jomoto, Billie Jonn, Heewoo Jun, Tomer Kaftan, Łukasz Kaiser, Ali Kamali, Ingmar Kanitscheider, Nitish Shirish Keskar, Tabarak Khan, Logan Kilpatrick, Jong Wook Kim, Christina Kim, Yongjik Kim, Jan Hendrik Kirchner, Jamie Kiros, Matt Knight, Daniel Kokotajlo, Łukasz Kondraciuk, Andrew Kondrich, Aris Konstantinidis, Kyle Kosic, Gretchen Krueger, Vishal Kuo, Michael Lampe, Ikai Lan, Teddy Lee, Jan Leike, Jade Leung, Daniel Levy, Chak Ming Li, Rachel Lim, Molly Lin, Stephanie Lin, Mateusz Litwin, Theresa Lopez, Ryan Lowe, Patricia Lue, Anna Makanju, Kim Malfacini, Sam Manning, Todor Markov, Yaniv Markovski, Bianca Martin, Katie Mayer, Andrew Mayne, Bob McGrew, Scott Mayer McKinney, Christine McLeavey, Paul McMillan, Jake McNeil, David Medina, Aalok Mehta, Jacob Menick, Luke Metz, Andrey Mishchenko, Pamela Mishkin, Vinnie Monaco, Evan Morikawa, Daniel Mossing, Tong Mu, Mira Murati, Oleg Murk, David Mély, Ashvin Nair, Reiichiro Nakano, Rajeev Nayak, Arvind Neelakantan, Richard Ngo, Hyeonwoo Noh, Long Ouyang, Cullen O'Keefe, Jakub Pachocki, Alex Paino, Joe Palermo, Ashley Pantuliano, Giambattista Parascandolo, Joel Parish, Emy Parparita, Alex Passos, Mikhail Pavlov, Andrew Peng, Adam Perelman, Filipe de Avila Belbute Peres, Michael Petrov, Henrique Ponde de Oliveira Pinto, Michael, Pokorny, Michelle Pokrass, Vitchyr H. Pong, Tolly Powell, Alethea Power, Boris Power, Elizabeth Proehl, Raul Puri, Alec Radford, Jack Rae, Aditya Ramesh, Cameron Raymond, Francis Real, Kendra Rimbach, Carl Ross, Bob Rotsted, Henri Roussez, Nick Ryder, Mario Saltarelli, Ted Sanders, Shibani Santurkar, Girish Sastry, Heather Schmidt, David Schnurr, John Schulman, Daniel Selsam, Kyla Sheppard, Toki Sherbakov, Jessica Shieh, Sarah Shoker, Pranav Shyam, Szymon Sidor, Eric Sigler, Maddie Simens, Jordan Sitkin, Katarina Slama, Ian Sohl, Benjamin Sokolowsky, Yang Song, Natalie Staudacher, Felipe Petroski Such, Natalie Summers, Ilya Sutskever, Jie Tang, Nikolas Tezak, Madeleine B. Thompson, Phil Tillet, Amin Tootoonchian, Elizabeth Tseng, Preston Tuggle, Nick Turley, Jerry Tworek, Juan Felipe Cerón Uribe, Andrea Vallone, Arun Vijayvergiya, Chelsea Voss, Carroll Wainwright, Justin Jay Wang, Alvin Wang, Ben Wang, Jonathan Ward, Jason Wei, CJ Weinmann, Akila Welihinda, Peter Welinder, Jiayi Weng, Lilian Weng, Matt Wiethoff, Dave Willner, Clemens Winter, Samuel Wolrich, Hannah Wong, Lauren Workman, Sherwin Wu, Jeff Wu, Michael Wu, Kai Xiao, Tao Xu, Sarah Yoo, Kevin Yu, Qiming Yuan, Wojciech Zaremba, Rowan Zellers, Chong Zhang, Marvin Zhang, Shengjia Zhao, Tianhao Zheng, Juntang Zhuang, William Zhuk, and Barret Zoph. 2024. GPT-4 Technical Report. arXiv:2303.08774 [cs.CL] https://arxiv.org/abs/2303.08774

Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. 2019. DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Thomas Paviot. 2022. pythonocc. https://zenodo.org/record/7471333. https://doi.org/10.5281/zenodo.7471333 Accessed: 2025-06-08.

Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. 2016. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. *arXiv preprint arXiv:1612.00593* (2016).

Xuanchi Ren, Jiahui Huang, Xiaohui Zeng, Ken Museth, Sanja Fidler, and Francis Williams. 2024. XCube: Large-Scale 3D Generative Modeling using Sparse Voxel Hierarchies. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. 2021. High-Resolution Image Synthesis with Latent Diffusion Models. arXiv:2112.10752 [cs.CV]

Danila Rukhovich, Elona Dupont, Dimitrios Mallis, Kseniya Cherenkova, Anis Kacem, and Djamila Aouada. 2024. CAD-Recode: Reverse Engineering CAD Code from Point Clouds. *arXiv preprint arXiv:2412.14042* (2024).

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal Policy Optimization Algorithms. arXiv:1707.06347 [cs.LG] https://arxiv.org/abs/1707.06347

Gopal Sharma, Difan Liu, Evangelos Kalogerakis, Subhransu Maji, Siddhartha Chaudhuri, and Radomír Měch. 2020. ParSeNet: A Parametric Surface Fitting Network for 3D Point Clouds. arXiv:2003.12181 [cs.CV]

Tianchang Shen, Zhaoshuo Li, Marc Law, Matan Atzmon, Sanja Fidler, James Lucas, Jun Gao, and Nicholas Sharp. 2024. SpaceMesh: A Continuous Representation for Learning Manifold Surface Meshes. In *SIGGRAPH Asia 2024 Conference Papers (SA Conference Papers '24)* (Tokyo, Japan, December 3-6, 2024). ACM, New York, NY, USA, 11. https://doi.org/10.1145/3680528.3687634

Dmitriy Smirnov, Mikhail Bessmeltsev, and Justin Solomon. 2021. Learning Manifold Patch-Based Representations of Man-Made Shapes. In *International Conference on Learning Representations (ICLR)*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2023. Attention Is All You Need. arXiv:1706.03762 [cs.CL] https://arxiv.org/abs/1706.03762

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2017. Pointer Networks. arXiv:1506.03134 [stat.ML] https://arxiv.org/abs/1506.03134

Kehan Wang, Jia Zheng, and Zihan Zhou. 2022. Neural Face Identification in a 2D Wireframe Projection of a Manifold Object. arXiv:2203.04229 [cs.CV] https://arxiv.org/abs/2203.04229

Xiaogang Wang, Yuelang Xu, Kai Xu, Andrea Tagliasacchi, Bin Zhou, Ali Mahdavi-Amiri, and Hao Zhang. 2020. PIE-NET: parametric inference of point cloud edges. In *Proceedings of the 34th International Conference on Neural Information Processing Systems* (Vancouver, BC, Canada) *(NIPS '20)*. Curran Associates Inc., Red Hook, NY, USA, Article 1693, 12 pages.

Xilin Wang, Jia Zheng, Yuanchao Hu, Hao Zhu, Qian Yu, and Zihan Zhou. 2025. From 2D CAD Drawings to 3D Parametric Models: A Vision-Language Approach. In *AAAI*.

Jiayi Weng, Huayu Chen, Dong Yan, Kaichao You, Alexis Duburcq, Minghao Zhang, Yi Su, Hang Su, and Jun Zhu. 2022. Tianshou: a Highly Modularized Deep Reinforcement Learning Library. arXiv:2107.14171 [cs.LG] https://arxiv.org/abs/2107.14171

Karl D. D. Willis, Yewen Pu, Jieliang Luo, Hang Chu, Tao Du, Joseph G. Lambourne, Armando Solar-Lezama, and Wojciech Matusik. 2021. Fusion 360 gallery: a dataset and environment for programmatic CAD construction from human design sequences. *ACM Trans. Graph.* 40, 4, Article 54 (July 2021), 24 pages. https://doi.org/10.1145/3450626.3459818

Rundi Wu, Chang Xiao, and Changxi Zheng. 2021. DeepCAD: A Deep Generative Network for Computer-Aided Design Models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 6772–6782.

Sifan Wu, Amir Khasahmadi, Mor Katz, Pradeep Kumar Jayaraman, Yewen Pu, Karl Willis, and Bang Liu. 2024. CadVLM: Bridging Language and Vision in the Generation of Parametric CAD Sketches. arXiv:2409.17457 [cs.CV] https://arxiv.org/abs/2409.17457

Jingwei Xu, Zibo Zhao, Chenyu Wang, Wen Liu, Yi Ma, and Shenghua Gao. 2024b. CAD-MLLM: Unifying Multimodality-Conditioned CAD Generation With MLLM. arXiv:2411.04954 [cs.CV]

Xiang Xu, Joseph G Lambourne, Pradeep Kumar Jayaraman, Zhengqing Wang, Karl DD Willis, and Yasutaka Furukawa. 2024a. BrepGen: A B-rep Generative Diffusion Model with Structured Latent Geometry. *arXiv preprint arXiv:2401.15563* (2024).

Xiang Xu, Karl DD Willis, Joseph G Lambourne, Chin-Yi Cheng, Pradeep Kumar Jayaraman, and Yasutaka Furukawa. 2022. SkexGen: Autoregressive Generation of CAD Construction Sequences with Disentangled Codebooks. In *International Conference on Machine Learning*. PMLR, 24698–24724.

Yang You, Mikaela Angelina Uy, Jiaqi Han, Rahul Thomas, Haotong Zhang, Suya You, and Leonidas Guibas. 2024. Img2CAD: Reverse Engineering 3D CAD Models from Images through VLM-Assisted Conditional Factorization. arXiv:2408.01437 [cs.CV] https://arxiv.org/abs/2408.01437

Haocheng Yuan, Jing Xu, Hao Pan, Adrien Bousseau, Niloy J. Mitra, and Changjian Li. 2024. CADTalk: An Algorithm and Benchmark for Semantic Commenting of CAD Programs. arXiv:2311.16703 [cs.CV] https://arxiv.org/abs/2311.16703

Xiaohui Zeng, Arash Vahdat, Francis Williams, Zan Gojcic, Or Litany, Sanja Fidler, and Karsten Kreis. 2022. LION: Latent Point Diffusion Models for 3D Shape Generation. In *Advances in Neural Information Processing Systems (NeurIPS)*.