Controllable Structured Object Generation

A thesis submitted in partial fulfillment
of the requirements for the degree of

Master of Science
in
Computer Science and Engineering by Research

by

Amruta Muthal
2019900002
amruta.muthal@research.iiit.ac.in

Advisor: Dr. Ravi Kiran Sarvadevabhatla

INTERNATIONAL INSTITUTE OF
INFORMATION TECHNOLOGY
HYDERABAD

International Institute of Information Technology Hyderabad
500 032, India

December 2025

International Institute of Information Technology Hyderabad
Hyderabad, India

CERTIFICATE

This is to certify that work presented in this thesis proposal titled ***Controllable Structured Object Generation*** by *Amruta Muthal* has been carried out under my supervision and is not submitted elsewhere for a degree.

Date	Advisor: Dr. Ravi Kiran Sarvadevabhatla

Acknowledgments

This thesis would not have been possible without the guidance, support, and encouragement of several individuals who have profoundly influenced both my academic journey and personal growth. First and foremost, I extend my deepest gratitude to my supervisor, Dr. Ravi Kiran Sarvadevabhatla, for his unwavering support throughout this research endeavor. Despite a demanding schedule, he was incredibly generous with his time, accommodating my needs with a flexibility that went far beyond his obligations, allowing me to pursue this work while also holding a full-time tech job. His guidance proved invaluable during moments when I felt stuck or uncertain about the direction of my work. More importantly, he helped me navigate the crucial transition from industrial practice to academic research, illuminating the fundamental differences between these two worlds while showing me how to leverage my industry experience as a strength in academic work. His mentorship taught me not only the rigor and depth required in academic research but also how to think critically and systematically about complex problems. The skills and perspectives I gained under his supervision will undoubtedly shape my approach to research and problem-solving for years to come.

I owe an immeasurable debt of gratitude to my parents Dr. Jyotsna Umredkar and Dr. Priyadarshan Muthal, whose unconditional love, unwavering support, and constant encouragement. Their belief in my abilities and the ambition they inspired, always urging me to aspire for what is more and embrace the highest ideals. To my brother, Amartya Muthal, I extend heartfelt thanks for being a constant source of camaraderie and joy of learning together. His own academic journey has been a constant source of inspiration for me. Most profoundly, I wish to honor the memory of my late grandmother, Shanta Umredkar, whose influence on my life extends far beyond what words can express. Though she is no longer here to see this milestone, her values, her wisdom, and her unwavering belief in education and personal growth were the motivating factors that made me restart my educational journey. This work stands as a testament to the foundation she helped build in me, and I carry her memory with gratitude and love. Most of all, I thank my husband, Priyank Jain, who stood by me through the highs and lows of this academic journey. He recreated the lab environment I needed at home, helping me frame arguments and structure my thesis. His support helped me see past my self-doubt to bring this research to completion. To all those mentioned here and to the many others who supported me along this journey, thank you.

Abstract

While text-to-image (T2I) models have demonstrated remarkable improvements in spatial consistency and positioning of large scene components, they struggle with fine-grained structural control, particularly for anatomically complex objects with multiple interconnected parts. This limitation manifests as anatomical inconsistencies, missing or hallucinated components and incorrect part connectivity, critical failures for applications requiring precise structural fidelity. Recent developments in controlled image generation have produced models capable of high-fidelity synthesis. However, these models either rely on sophisticated conditioning inputs such as detailed pose skeletons or segmentation masks that require human expertise.

We introduce PLATO, a novel two-stage framework that bridges this gap by enabling precise, part-controlled object generation from simple, intuitive inputs: an object category and a list of constituent parts. The first stage employs PLayGen, our novel part layout generator built upon a Graph Convolutional Network Variational Autoencoder (GCN-VAE) architecture. PLayGen consists of a GCN Refinement Decoder that iteratively refines part placements through graph-based message passing, capturing inter-part spatial relationships during refinement. To address the challenge of training with non-intersecting bounding boxes and small parts, we introduce the Dynamic Margin IOU (DyI) loss, which dynamically adjusts margins based on box dimensions to ensure meaningful gradients throughout training. Additionally, our Differential Refinement loss scheme enables the model to learn both coarse positioning and fine-grained adjustments simultaneously. The overall loss formulation also has the effect of stabilizing the training process while improving structural coherence.

In the second stage, PLayGen's synthesized layout guides a custom-tuned ControlNet-based diffusion model through a novel visual conditioning format using color-coded inscribed ellipses combined with part-aware text prompts. This design enforces both spatial constraints and semantic consistency, resulting in anatomically accurate, high-fidelity object generations that faithfully contain precisely the user-specified parts.

We observed that standard layout and image quality metrics do not sufficiently capture the nuances of structural composition. To address this gap for a rigorous structural consistency evaluation, we also propose novel part-level evaluation measures including Disconnection Percentage (measuring layout connectivity), Part-Layout IOU (quantifying spatial adherence) and Part-Layout F1 score (assessing part presence accuracy). Experiments on PASCAL-Parts and PartImageNet datasets demonstrate that PLATO significantly outperforms state-of-the-art approaches across both layout quality and structural controllability.

Contents

List of Figures

List of Tables

Symbols

G	Graph representing object part layout
X	Node feature matrix (part layout)
A	Adjacency matrix (connectivity between parts)
v	Node in the graph (representing a part)
p	Maximum number of parts across all object categories
x, y	Bounding box starting coordinates
w, h	Bounding box width and height
b	Binary flag indicating part presence
x_{min}, y_{min}	Minimum coordinates of bounding box
x_{max}, y_{max}	Maximum coordinates of bounding box
a_{box}	Bounding box area
$\widehat{X}_{\text{coarse}}$	Coarse layout prediction
$\widehat{X}_{\text{fine}}$	Fine layout prediction
$\widehat{X}_{\text{n}}$	Layout at refinement iteration n
$\widehat{P}_{\text{c}}$	Predicted part presence vector
O_c	One-hot encoded object class vector
L_{KL}	Kullback-Leibler divergence loss
L_{rec}	Reconstruction loss
L_{coarse}	Coarse reconstruction loss
L_{fine}	Fine reconstruction loss
L_{MSE}	Mean Squared Error loss
L_{BCE}	Binary Cross Entropy loss
$L_{pairMSE}$	Pairwise MSE loss
L_{DyI}	Dynamic Margin IOU loss
λ	Weight multiplier for KL divergence term
θ	Adaptive hyperparameter for balancing DyI and MSE
θ_1	Reconstruction error threshold
θ_2	Train-validation loss difference threshold
$alpha_w, alpha_h$	Margin multipliers for width and height

m_w, m_h	Computed margins for width and height
X_{pad}	Padded ground truth layout
$\widehat{X}_{\mathrm{fp}}$	Padded fine prediction layout
n	Current refinement iteration number
N	Total number of refinement iterations

List of Related Publications

[P1] Amruta Muthal, Varghese P Kuruvilla and Ravi Kiran Sarvadevabhatla, **"PLATO: Generating Objects from Part Lists via Synthesized Layouts"**, in proceedings of *ACM International Conference on Multimedia*, 2025.

Chapter 1

Introduction

1.1 Motivation

The challenge of object generation in computer vision is primarily driven by the dual goals of photorealism and control. Although foundational methods like Generative Adversarial Networks (GANs) and Variational Autoencoders (VAEs) established core principles, the landscape has been reshaped by diffusion models. Architectures from leading research labs, including DALL-E [1], Imagen [2], and the open-source Stable Diffusion [3], demonstrated unprecedented realism by leveraging Convolutional U-Net backbones, becoming the cornerstones of modern text-to-image synthesis. New entrants like FLUX [4] are now progressively replacing or augmenting these backbones with more scalable Transformer-based architectures. This architectural progress, combined with the ever-deepening semantic understanding provided by Large Language Models (LLMs), has pushed the quality of image generation to remarkable new heights. However, this very success has highlighted a crucial gap, While models can now render stunning textures and complex scenes, precise control over object structure remains a fundamental challenge. This gap in controllability is the primary motivation for this thesis. We trace the trajectory of generative model development to demonstrate how the research frontier has logically progressed from high-level semantic guidance towards the challenge of fine-grained, structurally aware, part-level object synthesis.

1.1.1 The Dual Pursuit of Photorealism and Control

A long-standing objective in computer vision has been the synthesis of images from abstract descriptions. In recent years, significant progress has been made through the development of deep generative models. The core pursuit within this field is twofold: achieving high-fidelity, photorealistic outputs and enabling precise, user-driven control over the generated content. Although the challenge of photorealism has seen remarkable advancements, the quest for granular, intuitive control remains an active and critical area of research. This thesis aims to address the specific need for part-level object composition through a two-stage framework.

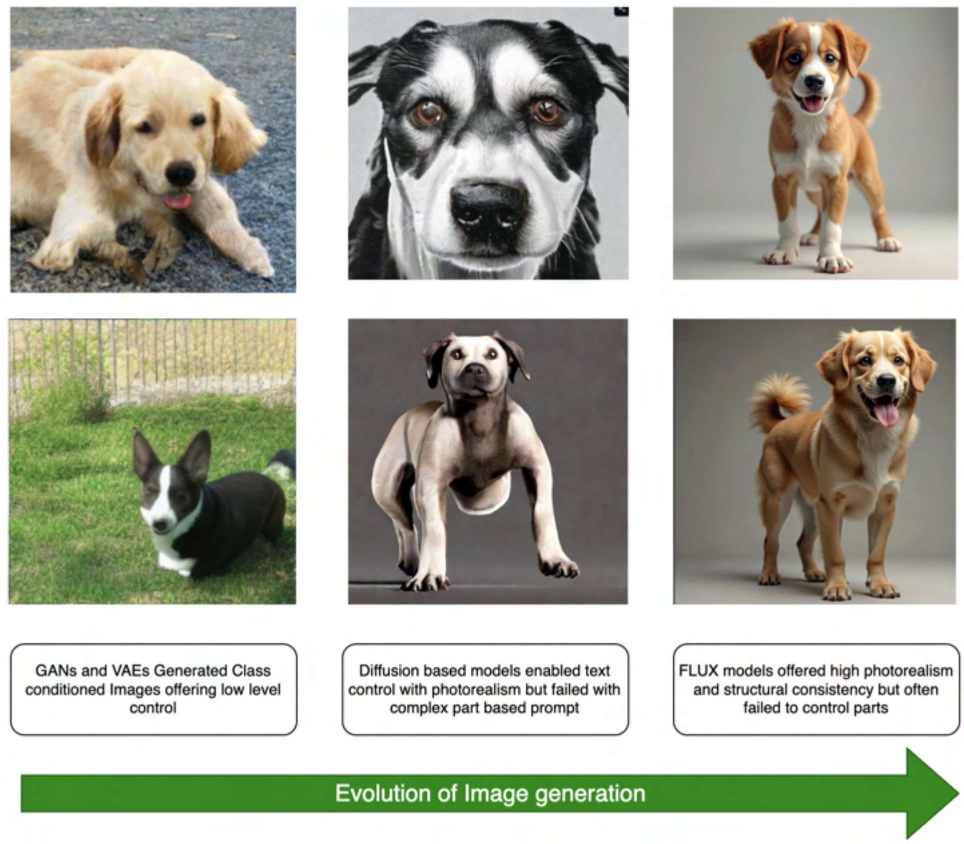


Figure 1.1 Evolving quality of Image Generation: From the early days of GAN [5] and VAE [6, 7] based generation models, photorealism has been the primary goal. With the development of Diffusion based model that enabled free text guidance of the generation process. A granular structural control, however, still remains a challenge

1.1.2 Evolution of Generative Control: From Latent Space to Semantic Guidance

The evolution of generative models is characterized by progressively more sophisticated methods of control. Foundational architectures, such as Variational Autoencoders (VAEs) and Generative Adversarial Networks (GANs), established the viability of image synthesis from learned data distributions. In these models, control was primarily exercised through the manipulation of latent space vectors. This method, while powerful, was often non-intuitive and could result in entangled attribute modifications, where altering one feature would undesirably affect others.

A paradigm shift occurred with the advent of Diffusion Models. These models operate by learning to reverse a forward diffusion process that systematically adds Gaussian noise to an image until it becomes pure noise. The reverse process, executed by a neural network, iteratively denoises a random tensor into a coherent image. The efficacy of modern diffusion models, such as those in the Stable Diffusion family [3], stems from the synergy between a U-Net-based denoising autoencoder and powerful text encoders, such as CLIP [8]. The text encoder provides rich semantic guidance from natural language prompts, which is injected into the U-Net via cross-attention mechanisms. This architecture enabled high-level semantic control, allowing users to generate complex scenes and objects from text. However, this also exposed a structural deficit in these models which result in generation of semantically plausible but structurally inconsistent outputs.

Figure 1.2 Evolution of Generative Control: While modern image generation models can produce images of high photorealism and render customized textures unlocking a new level of creative control, they do not yet directly tackle the problem of structural control instead relying on language semantics to transform user prompts to images.

1.1.3 The Bottleneck of External Spatial Conditioning

The structural limitations of purely text-guided models prompted the development of methods for more explicit spatial control. This led to a class of models that augment large, pretrained diffusion models with conditioning inputs. The most prominent example is ControlNet [9] which creates a trainable copy of the diffusion model's encoding blocks to processes a spatial condition (e.g., a canny edge map, human pose, or segmentation mask). The output is then fed into the original, frozen diffusion model via zero-convolution layers. These specialized layers are initialized with zero weights, ensuring that the new conditioning module does not negatively impact the vast knowledge stored in the pretrained model during early stages of training.

This design allows for robust learning of spatial control without requiring catastrophic retraining of the base model. Related approaches include InstanceDiffusion [10] and GLIGEN [11], which allows conditioning on bounding boxes and other fine-grained modalities, have further demonstrated the power of this paradigm. However, all these methods share a fundamental bottleneck: their effectiveness is wholly dependent on the quality and precision of the user-provided conditioning signal. They solve the problem of how to follow a guide, but not where the guide comes from, shifting the burden of structural definition entirely to the user.

1.1.4 Intrinsic Part-Based Generation

The reliance on external guides defines the current research frontier and motivates a critical question: instead of requiring a human to provide a detailed structural blueprint, can a model learn to generate a plausible blueprint itself? This question frames the challenge of intrinsic, part-controllable object generation.

This level of control, the most granular in the spectrum (illustrated in Figure 1), aims to compose an object from its constituent parts based on a learned, internal understanding of object structure. As the historical trajectory (summarized in Figure 2) indicates, the field consistently moves towards more explicit and detailed forms of control. We posit that the next logical step is to internalize the generation of the structural guide itself.

This thesis directly addresses this challenge. We propose a framework that first learns to generate plausible, part-based structural layouts and subsequently uses these layouts to steer a high-fidelity diffusion model. By doing so, we aim to bridge the gap between high-level semantic intent and fine-grained, structurally-aware image synthesis, thereby addressing the critical bottleneck in the current state-of-the-art.

1.2 Problem Statement

We explore the intricate problem of generating structured objects with high fidelity and part-level control with three focus areas.

- **Generation of High-fidelity layout of structured object with multiple part.** Our aim is to develop a Variational model that can generate a plausible pose given an object class and a list of parts

- **Adapting standard image generation and control models for part-controllable object generation.** Given a structured layout as a conditioning input, we aim to finetune standard generation models to the specific task of object generation.

- **Evaluation Framework for assessing part controllability and generation fidelity.** To evaluate our proposed architecture, we develop a framework of multiple metrics that assess part level fidelity and overall photo realism.

1.3 Research Challenges

- Data quality and size. Fine grained part data is scarce and of low quality

- Generating layouts as graphical structures with good generalization over different size of parts and realistic relative placements

- Adapting conditioned generation models for object generation

1.4 Thesis Contributions

As a contribution to this thesis, we propose a two stage architecture to solve the problem of high quality object generation and propose methods to evaluate this model. Below is a summary

- An improved GCN VAE with a Graph-based Part Layout Refinement block to generate the Object Part Layout
- Dynamic IOU loss for improving the fidelity of generated small part and Differential loss schemes to balance the objective of reconstruction and refinement.
- Custom conditioning methods for generating objects from part layouts using a Task Fine-Tuned ControlNet.
- Novel Evaluation metrics to measure part level fidelity, connectedness of generated layout and overall visual quality of the Generated object.

The code, model artifacts and datasets for this project can be found at https://plato-gen.github.io/

1.5 Structure of the proposal

In this chapter, we introduce the problem domain and the motivation behind it. We provide a brief overview of the research challenges in this domain, existing literature and its limitation in regard to the challenges of this domain. In *Chapter 2*, we describe the key methods used to approach these problems and provide a background on which these methods are built. **Chapter 3** covers the Layout Generation component, PLayGen, of our proposed generation framework and including the ablation studies used to choose various model parameters. *Chapter 4* details the development of the Image generation component including the effect of various design choices. In *Chapter 5* we describe the experimental setup used to train and compare our generation framework to the systems described in Chapter 2. We also present the evaluation framework developed of measuring structural consistency that we use to quantify the efficacy of each system. Finally, Chapter 6 concludes this thesis by summarizing the results obtained thus far. It provides an overview of the work conducted and presents the future exploration paths that emerge from this work.

Chapter 2

Related Works

The field of generative vision models has undergone significant changes from the early days of probabilistic models that generate images within a limited data distribution to the seemingly boundless text controllable large generative models that can bring the wildest imaginations to life. In this chapter, we delve deeper into the shifts in this generation paradigm and highlight the aspects of controllability that are central to our research problem, concluding with the specific research gap that we aim to solve with this proposal.

2.1 Evolution of High-Fidelity Generation and the Emergence of the Control Problem

The foundational era of deep generative models was led by two primary architectural paradigms: Variational Autoencoders (VAEs) [6] and Generative Adversarial Networks (GANs) [5] which together established the probabilistic and adversarial frameworks that shaped subsequent generative model research. VAEs employed an encoder-decoder setup with a reconstruction objective to learn a compressed, continuous latent representation of data which could be sampled to generate new images. Although this architecture enabled learning a structured representation of visual features, its objective function often prioritized pixel-level accuracy at the expense of high-frequency details and inter-pixel dependencies, leading to images that could lack photorealistic sharpness [12, 13].

Owing to a competitive generator-discriminator architecture, GANs achieved a significant advance in photorealism. Its adversarial setup created a dynamic loss function that steers generator to learn the true data distribution that can fool the discriminator. This competitive pressure allowed GANs to produce the sharp textures and fine details that VAEs often missed, setting a new standard for photorealism. However, the adversarial training objective was inherently unstable and prone to mode collapse due to the min-max optimization which adversely affected the convergence of gradient descent methods. Despite multiple GAN variations developed to address these challenges, training stability remains a problem.

Diffusion models [14, 15] developed as a new paradigm that offered both high fidelity photorealism and training stability. Early versions of these models utilized a U-Net [16] backbone trained on the objective of iteratively removing Gaussian noise from a corrupted image. This step-by-step denoising process proved to be far more stable than the single-shot generation of GANs. During generation, the model began with pure random noise and repeatedly applied its learned denoising function, allowing complex and coherent details to emerge gradually and reliably. The U-Net architecture had its own limitations owing to the convolutional layers. While highly effective for image-to-image tasks, the U-Net's convolutional layers possess a strong inductive bias towards local operations, processing information within limited receptive fields. Capturing long-range spatial dependencies—such as the complex relationship between an object and its shadow or ensuring anatomical consistency across a large figure, required deep stacks of layers to gradually expand the model's view of the image. This approach was computationally inefficient and still fell short of effectively modeling the global, non-local interactions necessary for perfect structural coherence.

Transformer-based architectures, originally developed for natural language processing, offered a direct solution to this challenge. Instead of relying on local convolutions, the core of a Transformer is the self-attention mechanism. This mechanism reconceptualized an image (or, more commonly, its latent representation) as a sequence of patches or tokens. The self-attention operation enabled every token in the sequence to directly attend to and weigh the importance of every other token, regardless of its position. This fundamental difference allowed the model to capture global context and model long-range dependencies from its earliest layers, providing a more holistic and immediate understanding of the overall image structure compared to the hierarchical approach of CNNs.

A key advantage of this architectural design was its superior scalability. Transformers have consistently demonstrated that their performance improves more predictably and effectively with increases in model parameters and training data. This property has been leveraged by the latest generation of text-to-image models, such as Stability AI's Stable Diffusion 3 [17] and Google's Imagen 2 [2], which have either partially or fully replaced the U-Net backbone with more scalable Transformer-based designs. This architectural shift represents a significant step towards models that possess a more robust understanding of complex spatial relationships, a critical prerequisite for achieving the fine-grained, structurally-aware control that is the focus of current research.

Multimodal Guidance was the final architectural intervention that unlocked the full potential of diffusion models. This was achieved by augmenting the U-Net with a powerful, pre-trained text encoder (like CLIP) and connecting them via cross-attention mechanisms. This works by converting a user's text prompt into a rich numerical embedding that is injected into the U-Net at each denoising step. The cross-attention layers allow the U-Net to selectively focus on different parts of the text prompt as it generates the image, conditioning the entire refinement process on the prompt's semantic meaning. This was a revolutionary improvement over the abstract latent-space manipulation of VAEs and GANs, providing an intuitive and expressive way for users to control image generation with natural language. This architectural synergy between a U-Net denoiser and a text encoder became the cornerstone of modern text-to-image synthesis.

The primary reason for structural failures lies in the models' training objective [18, 19]. They are trained on vast internet datasets to become powerful correlation engines, not symbolic reasoners. The objective function rewards the model for learning that certain textures, colors, and shapes are statistically likely to appear together. For instance, a model learns the texture of *fur*, the shape of an *ear*, and the color *black* are often associated with the word *cat*. It does not, however, learn a formal, symbolic rule such as *a cat must have exactly four legs connected to a torso.* Various works highlight this phenomenon through the lens of visual question answering studies [19] and spatial and compositional reasoning [18] and anatomical consistency in generations [20]. In case of generating objects, it leads to anatomically inconsistencies like extra limbs or misplaced features, as the model is simply generating a statistically probable assembly of parts rather than following a coherent blueprint.

While the cross-attention mechanism is revolutionary for linking text to image content, it is fundamentally a soft probabilistic tool. It excels at ensuring the *what* (semantic content) is present but is less precise about the *how many* (counting) or *where* (precise spatial relationships) [21, 22]. Counting is a well-known weakness; a prompt for *a person with five fingers* may still yield a hand with six or four fingers because the model doesn't possess a discrete counting module. It only approximates what a *hand* statistically looks like [23]. Similarly, complex spatial prompts requiring a precise understanding of part connectivity can fail because the attention mechanism may struggle to disentangle object attributes from their required geometric relationships [18, 24].

Finally, there is a significant gap between the high-level semantic information in a text prompt and the low-level geometric arrangement of pixels in an image. These models learn to bridge this gap directly without creating an intermediate explicit structural representation of the objects and scene. They can generate outputs that are semantically plausible but geometrically or physically impossible, such as a *phantom limb* that is disconnected from a body or parts that overlap in an unnatural way. The model satisfies the semantic prompt, but without an underlying structural scaffold, it fails to produce a coherent and consistent final image.

2.2 Introduction of User defined Structural Control

The structural limitations inherent in purely text-guided models prompted the development of methods for more explicit spatial control. This led to a new class of models designed to augment large, pre-trained diffusion models with an additional, non-textual conditioning input. The core idea was not to retrain the massive base models, but to add a small, trainable module that could interpret a spatial guide—such as a human pose skeleton, a depth map, or a segmentation mask—and steer the generation process accordingly. This approach effectively offloaded the responsibility of defining the image's structure from the model to the user, who now provided the explicit blueprint for the model to follow.

The most prominent example of this paradigm is ControlNet [9]. Its architectural innovation was to create a trainable copy of the encoding blocks of a large, pre-trained diffusion model while keeping the original model's weights frozen. This trainable copy is tasked with processing the user-provided spatial condition (e.g., a canny edge map). Its output is then fed back into the corresponding layers of the frozen, original model via specialized *zero-convolution* layers. These layers are initialized with zero weights, which is a crucial detail: it ensures that during the early stages of training, the new control module does not disrupt the vast knowledge stored in the pre-trained model. This architectural paradigm was not an isolated success; related approaches further demonstrated its power. T2I-Adapters introduced similarly lightweight modules for control, while GLIGEN [11] specialized in providing spatial guidance through bounding boxes, and InstanceDiffusion [10] focused on enabling fine-grained, instance-level control using keypoints, bounding boxes and segmentation masks. Together, these models established a robust framework for conditioning powerful, pre-trained diffusion models on explicit user-provided structures.

This wave of solutions was a major breakthrough, enabling a degree of fine-grained control that was previously impossible. However, all of these methods share a fundamental bottleneck: their effectiveness is entirely dependent on the quality and precision of the user-provided conditioning signal. They masterfully solve the problem of how to follow a guide, but do not address the question of where the guide comes from. This architectural choice shifts the burden of complex structural definition—often requiring tedious or expert-level work—entirely to the user. This limitation defined the next major research frontier: creating models that could generate this plausible structural guide themselves.

2.3 Generating High level structural control with Low level user inputs

The reliance on external user-provided guides defined the primary bottleneck of the first wave of controllable models. This prompted a critical research question: instead of requiring a human to provide a detailed structural blueprint, could a model learn to generate a plausible blueprint itself? This challenge, automating the generation of the structural guide, was through two types of methods top-down scene layout generation and bottom-up part-based object synthesis.

The first paradigm focused on generating layouts for entire scenes, often leveraging the power of Transformers and Large Language Models (LLMs). Transformer-based approaches such as BLT (Bidirectional Layout Transformer) [25] model layouts as ordered token sequences, leveraging attention mechanisms to capture dependencies between spatial elements. Similarly, LayoutFormer++ [26] extends this idea to document and scene layout generation, emphasizing semantic alignment and diversity across configurations. These transformer-based frameworks demonstrate how autoregressive and attention-driven models can encode global scene structure while maintaining flexibility in layout composition.

Models such as LayoutFormer++ [26] and LayoutDM [27] were developed to arrange multiple distinct objects in a 2D space, offering flexible control over the overall composition. However, these approaches, while powerful for scene-level tasks, proved ill-suited for composing a single, complex object from its constituent parts. Generating realistic objects involves far stricter requirements, such as maintaining precise part connectivity, preserving relative sizes, and enforcing specific spatial alignments.

Scene-level methods focused primarily on coarse spatial arrangements and lacked the fine-grained structural coherence crucial for generating interconnected parts such as the limbs of an animal.

A second, more direct paradigm tackled the specific challenge of generating the internal structure of a single object from a list of its parts. This task is fundamentally a graph problem, where parts are nodes and their physical connections are edges. Although graphs had been explored in other layout domains, MeronymNet [28] was a key predecessor that specifically attempted unified, part-aware object generation. It correctly identified that a graph-based architecture was the right approach to explicitly model part-to-part connectivity. However, Its implementation suffered from training instability due to multiple competing losses and its reliance on a simple Multi-Layer Perceptron (MLP) decoder made it difficult to enforce the strict structural constraints required for coherent object anatomy. This demonstrated that while the graph-based approach was promising, a more robust decoding architecture was needed to realize its full potential. This left a clear research gap for a model that could stably generate high-fidelity, structurally-coherent object part layouts.

2.4 Introducing PLATO

The evolution of generative vision models reveals a consistent path toward greater fidelity, controllability, and semantic alignment. From VAEs to GANs and diffusion models, each architectural development has been in the direction of improving realism, training stability, or multi-modal guidance. Despite these advances, a fundamental challenge of structural reasoning and consistency remains. These gaps are largely attributed to a lack of explicit mechanisms for understanding or enforcing the compositional and geometric rules that govern object structure. While some of these gaps have been bridged using cross-attention mechanisms between text and image, the alignment is still probabilistic in nature and struggles with discrete or structurally grounded reasoning. This has been a source of anatomically inconsistent and numerical errors in generated images such as misplaced or hallucinated parts, which is the central problem being addressed in our work.

Subsequent architectural innovations, most notably ControlNet and related controllable diffusion frameworks, successfully reintroduced structural guidance by allowing users to provide explicit spatial conditions. While these methods shift the responsibility of anatomical consistency on the users to provide high quality inputs, they can be a good scaffold to control structures through intermediate representations, owing to their ability of imposing harder constraints on the denoising process during inference. Thus, providing a skeletal structure as an intermediate control mechanism becomes a viable approach to ensure structural consistency. This breaks down the larger problem into two steps, the first being that of generating a control structure and the second of developing a control model that can be used to condition an image generator.

PLATO tackles the first step using a part layout generator — building on top of Graph VAEs — with a novel Graph Refinement Decoder instead of a standard MLP to generate a blueprint of where each part should be placed in an image. This stochastic model is trained with novel losses such as the Dynamic Margin IOU loss to improve resolution of small parts and inter-part connectivity while also stabilizing the training process and Refinement loss to enable course and fine learning and enable refinement in the generation mode. To transform this layout of parts into a photo-realistic image, we fine-tune a ControlNet model on a custom control format that can encode part level details. Due to its modular setup, PLATO can be easily modified to work with small and large models alike.

Most of the works that we considered measure the overall statistical similarity of the generated distribution to training and testing data such as Image-FID [29] and Layout-FID [30] and measure semantic consistency with CLIP score [31]. While these measures work well for general images, they do not correctly identify the misses in anatomical accuracy. To mitigate this issue, we also established a suite of metrics to quantify the structural consistency of objects in a generated image. Metrics using as the Disconnection Percentage, Part Layout F1 score measure the the consistency of generated part layouts, while Part Layout F1 measures structural adherence in the final generated image.

Chapter 3

Generating Structured Object Layout

Because object parts are highly interconnected, representing an object's layout as a graph is an effective way to model the dependencies between them. Our objective is to generate layouts conditioned on object categories and part lists, making a Graph Convolutional Network Variational Autoencoder (GCN-VAE) [7] as an ideal base framework. This network uses Graph Convolution Network layers [32] to encode and decode the layout graph, while a variational objective facilitates stochastic generation.

The GCN-VAE training objective has two standard components: a Kullback-Leibler (KL) divergence loss $\mathcal{L}_{KL}$ defined over the latent embedding and a reconstruction loss $\mathcal{L}_{rec}$. Typically, the reconstruction loss for bounding boxes combines a Mean Squared Error (MSE) term for box coordinates and a Binary Cross Entropy (BCE) term for part presence labels [28]. In addition to these, we introduce a novel Dynamic Margin IOU loss section 3.4 (DyI) which we describe in 3.4.

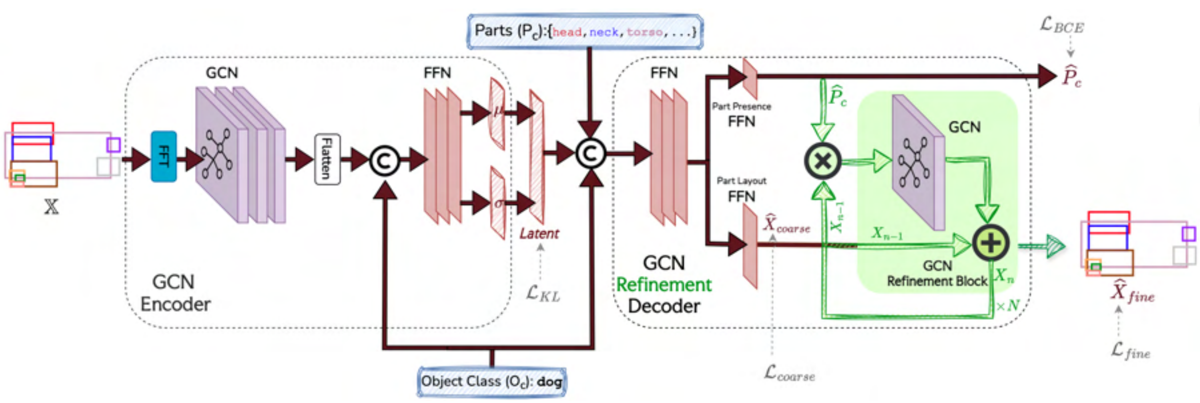


Figure 3.1 PLayGen Architecture: Our layout generator is a parts and object category conditioned GCN-VAE with a novel GCN Refinement Decoder which outputs the final generated part layout $\widehat{X}_{fine}$. The input to encoder is a part layout graph $\mathbb{G} = (\mathbb{X}, \mathbb{A})$, but $\mathbb{A}$ is omitted for simplicity. Losses wrt various intermediate and final outputs are shown in grey. Note that X_{coarse} is used within GCN Refinement Block only once, i.e. $X_{coarse} = X_0$ (section 3.3).

This chapter details our key contributions to improve upon this base model for the specific task of part layout generation. We begin by defining our data representation strategy and establishing the optimal format for encoding bounding box and connectivity information. We then introduce our primary architectural modification: a novel GCN Refinement Decoder designed to progressively improve layout fidelity. To train this new architecture effectively, we propose two advancements in loss formulation: the Dynamic Margin IOU (DyI) loss, which stabilizes training for non-intersecting parts, and a Differential Refinement loss scheme to balance coarse placement with fine-grained adjustments. Each of these contributions is validated through a series of ablation studies that quantify the increment in reconstruction power of the network. Together, these innovations lead to a more stable and accurate model that demonstrates better generalization over complex object part layouts.

3.1 Data representation

Our goal is to have a single model which can generate part layouts for multiple object categories. To accommodate this requirement, we assign a unique index to each part. Let p be the maximum number of parts possible across all object categories. To generate realistic layouts for objects, it is important to capture the relationships between different parts. To achieve this, we represent the object as a graph $\mathbb{G} = (\mathbb{X}, \mathbb{A})$ with p nodes. Each node $v \in \mathbb{X}$ represents an object part bounding box information and $\mathbb{A}$ is the adjacency matrix that captures the connective relationships among parts.

3.1.1 Finding Optimal Bounding Box Representation

The node features in $\mathbb{X}$ must encode both the bounding box coordinates and the presence of each part. To determine the most effective encoding scheme, we evaluated three configurations for the node feature vector:

- **XY-HW:** Each node is represented as $[x, y, w, h, b]$, where $[x, y, w, h]$ are the box's starting coordinates, width, and height, and b is a binary flag indicating the part's presence. Absent parts are represented by a zero vector $[0, 0, 0, 0, 0]$.
- **XY-MinMax:** Each node is represented as $[x_{min}, y_{min}, x_{max}, y_{max}, b]$, using the coordinates of the box's corners.
- **XY-MinMax + Area:** This configuration augments the XY-MinMax representation by adding the box area, $[x_{min}, y_{min}, x_{max}, y_{max}, a_{box}, b]$,

Table 3.1 Effect of Bounding box representation

Model	XY-HW	XY-MinMax	XY-MinMax + Box area
$mIOU_{rec}$	**.559**	.513	.528

As shown in table 3.1, the **XY-HW** configuration yielded the best reconstruction results, and we therefore select it for our model.

3.1.2 Effect of Part connectivity configurations

The adjacency matrix $\mathbb{A}$ encodes the natural, physical connections between parts (e.g., a lower arm connects to an upper arm, which connects to the torso). For any given object class, this connectivity is fixed, meaning $\mathbb{A}$ remains constant across all instances during both training and inference. To validate this choice, we experimented with three different connectivity configurations table 3.2:

- **Fixed (Baseline):** A predefined matrix based on the natural anatomical connectivity of parts for each object class.
- **Identity:** An identity matrix, assuming no connectivity between different parts.
- **Fully Connected:** A matrix where every part is connected to every other part

The experiment confirmed that using a **Fixed** connectivity matrix yields significantly better reconstruction performance. Consequently, we use this approach to represent all object layouts in our dataset.

Table 3.2 Effect of Connectivity Matrix

Model	Baseline	Identity	Fully Connected
$mIOU_{rec}$	**.559**	.201	.502

3.2 GCN Encoder

In the first step, The encoder first transforms the node features $\mathbb{X}$ to its Fourier representation. This transformation, inspired by recent works [10, 33], enables capturing the high-frequency details inherent in the distribution of part geometries and locations. Following this, the Fourier-transformed features are processed through a series of GCN layers to produce a single feature vector that represents the entire graph. To incorporate category-specific information, the one-hot encoded object class vector O_c, is concatenated with this graph representation. This combined vector is then passed through a Feed-Forward Network (FFN) to produce the mean and variance parameters required for the VAE's latent space (fig. 3.1).

3.3 GCN Refinement Decoder

The decoding process begins by combining the latent representation from the VAE encoder with the part presence vector P_c and the object class vectorO_c. This concatenated embedding is passed through an FFN, which then splits the output into two branches: 1) the **Part Layout FFN** generates an initial, coarse prediction of the bounding box layouts, denoted as $\widehat{X}_{coarse}$. 2) the Part Presence FFN predicts the part presence vector, $\widehat{P_c}$.

We found that generating the final layout in a single step does not fully resolve the interconnectedness of parts resulting in poor fidelity and disconnected parts. To address this, we introduce a novel **GCN Refinement Block** that progressively improves the layout quality through an iterative process. This block takes the layout from the previous step ($\widehat{X}_{n-1}$) and predicts a corrective residual. This residual is then added to the input to produce a more refined layout, $\widehat{X}_n$. The iterative update rule is defined as:

$$\widehat{X}_n = \widehat{X}_{n-1} + GCN(\widehat{X}_{n-1} \cdot \widehat{P}_c, \mathbb{A}) \tag{3.1}$$

The process is initialized with the coarse prediction, $\widehat{X}_0 = \widehat{X}_{coarse}$ and the final output after N refinement steps is the high-fidelity layout, $\widehat{X}_{fine}$.

3.3.1 Effect of GCN Refinement iterations

To validate the effectiveness of our GCN Refinement Block, we conducted an ablation study. We compared a baseline model, which generates layouts in a single step $\widehat{X} = \widehat{X}_{coarse}$), against versions of our model with varying numbers of refinement iterations (N=1, 2, and 5). For the baseline model, the coarse loss term (L_{coarse}) was removed from the training objective. As shown in Table 3.3, iterative refinement provides a clear benefit to the final reconstruction quality.

Table 3.3 Effect of number of GCN refinements

Model	Baseline	n=1	n=2	n=5
$mIOU_{rec}$	.518	.547	**.559**	.527

3.4 Dynamic Margin IOU Loss

A common challenge with bounding box regression is the *zero-gradient problem* that occurs when conventional Intersection over Union (IOU) loss is used in the training objective. When a predicted box and a ground-truth box do not overlap, the IOU is zero, which provides no gradient for the model to learn from. This makes it impossible for the network to distinguish between a near miss and a completely incorrect prediction. While alternative formulations like GIOU and CIOU [34–36] have been proposed, we found that they led to unstable convergence and poor placement of smaller parts in our multi-loss training environment. Details of this evaluation are presented in table 3.4

To overcome this, we propose a novel Dynamic Margin IOU (DyI) loss. The core idea of DyI is to dynamically expand the boundaries of both the predicted and ground-truth boxes with a calculated margin, ensuring they always intersect and thus provide a usable gradient.

This is achieved in three steps:

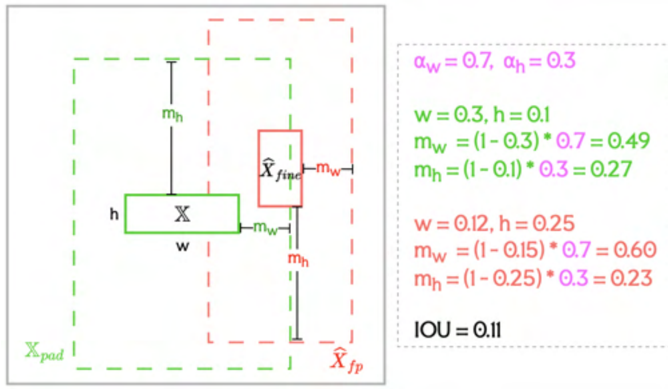


Figure 3.2 Dynamic Margin IOU: To compute IOUs for non-intersecting bounding boxes, we pad the boxes with margins generated by our novel Dynamic IOU loss (section 3.4). Here, green solid box represents a part bounding box from ground truth and red solid box represents its corresponding generated box. The IOU is computed between the dotted counterparts as described in section 3.4.

- **Learn Margin Multipliers:** The network first learns a set of margin multipliers, α_w (for width) and α_h (for height), for each part's bounding box using an FFN (eq. (3.2)). Hence α_w and α_h are p-dimensional vectors.

- **Calculate Margins:** These multipliers are scaled by the inverse of the ground-truth box dimensions to compute the final margins, m_w and m_h (eq. (3.3)). This inverse relationship ensures that smaller boxes have larger margins, which enables a non-zero overlap with their targets during training.

- **Compute Padded IOU:** The original boxes are padded with these margins (as illustrated in Figure 3.2), and the final IOU loss is calculated on these expanded boxes. The complete DyI loss, $\mathcal{L}_{DyI}$, combines this padded IOU loss with a regularization term for the margins (eq. (3.6)).

$$[\alpha_w, \alpha_h] = FFN(\mathbb{X}, \widehat{X}_{fine}) \tag{3.2}$$

$$[m_w, m_h] = (1 - [w, h]) \cdot [\alpha_w, \alpha_h] \tag{3.3}$$

$$\mathbb{X}_{pad} = Pad(\mathbb{X}, m_w, m_h) \tag{3.4}$$

$$\widehat{X}_{fp} = Pad(\widehat{X}_{fine}, m_w, m_h) \tag{3.5}$$

$$\mathcal{L}_{DyI} = -log(IOU(\mathbb{X}_{pad}, \widehat{X}_{fp})) + \alpha_w^T \mathbf{1}_p + \alpha_h^T \mathbf{1}_p \tag{3.6}$$

3.4.1 Enabling loss computation in absence of intersections

Consider two boxes which are farthest apart on the normalized square canvas side length 1. Let one of them be the groundtruth box A with coordinates $(0, 0); (w_1, h_1)$, and the other be predicted box B with coordinates$(w_2, h_2); (1, 1)$ as shown in fig. 3.3.

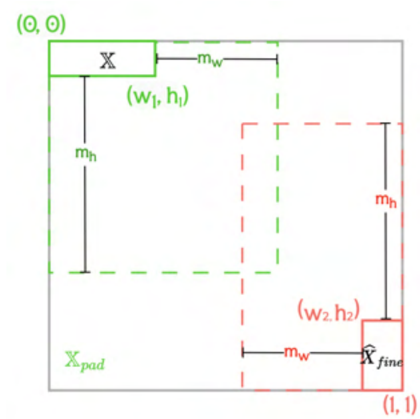


Figure 3.3 Proof of Intersection with dynamic margins: Here the box at the top left as coordinates (0, 0) is considered the ground truth box and the one at the bottom right at (w_2, h_w) is the predicted box. m_w and m_h are the computed margins. The dotted boxes represent the effective bounding boxes after margins are added.

Let m_w be margin added to boxes for DyIoU computation (Fig.3 in paper). The two boxes intersect along X axis when

$$w_2 - (w_1 + m_w) > 0 \tag{3.7}$$

Or,

$$m_w > w_2 - w_1 \tag{3.8}$$

Since $m_w = (1 - w_1)\alpha_w$ (eq:2 in paper), we get

$$2\alpha_w > (w_2 - w_1)/(1 - w_1) \tag{3.9}$$

Thus, for an intersection, $\alpha_w > (w_2 - w_1)/2(1 - w_1)$.

Now, $w_2 < 1$ implies that $(w_2 - w_1)/(1 - x_1) < 1$ always holds true. Thus, for a guaranteed intersection α_w should be $> 1/2$.

At the beginning of training α_w is initialized to 0.5. As the training progresses, this lower bound on α_w decreases with gap $(w_2 - w_1)$ between predicted and ground truth box, allowing for nonzero IoU (and gradient) during rest of training also. Similarly, the reasoning holds for α_h along the Y axis.

To demonstrate the effectiveness of our loss fomulation, we compare our it against GIOU [36], DIOU [35] and CIOU [34]. The baseline model uses standard IOU formulation. Of these, models

GIOU and CIOU losses failed to converge and generate out of range values. We select DyI in our final model formulation owing to it's higher reconstructions score.

Table 3.4 Comparison of IOU formulations

Model	Baseline	GIOU	DIOU	CIOU	DyI
$mIOU_{rec}$	.497	-	.501	-	**.535**

3.5 Differential Refinement loss

As described previously (section 3.3), the network outputs a coarse layout representation $\widehat{X}_{coarse}$ and a final layout $\widehat{X}_{fine}$. When the same reconstruction loss is applied to both these outputs, the optimization tends to produce a trivial solution where the GCN Refinement block (section 3.3) behaves like an identity function (see fig. 3.4). To avoid this, we first define dedicated loss functions ($\mathcal{L}_{coarse}, \mathcal{L}_{fine}$) for each type of output separately.

$\mathcal{L}_{coarse}$ consists of $\mathcal{L}_{MSE}$, a MSE term defined over intermediate bounding box generations $\widehat{X}_{coarse}$. Likewise, $\mathcal{L}_{fine}$ consists of a similar MSE term defined over final bounding box generations $\widehat{X}_{fine}$. A pairwise MSE term defined over bounding box pairs ($\mathcal{L}_{pairMSE}$) and BCE defined over part presence labels ($\mathcal{L}_{BCE}$) is included [28]. To balance the DyI loss (section 3.4) with MSE for $\widehat{X}_{fine}$, we define $\mathcal{L}_{fine} = \mathcal{L}_{DyI}(1+\theta) + \mathcal{L}_{MSE}(1-\theta) + \mathcal{L}_{pairMSE} + \mathcal{L}_{BCE}$ where θ is an adaptive hyperparameter. The final reconstruction loss is defined as $\mathcal{L}_{rec} = \mathcal{L}_{coarse} + \mathcal{L}_{fine}$.

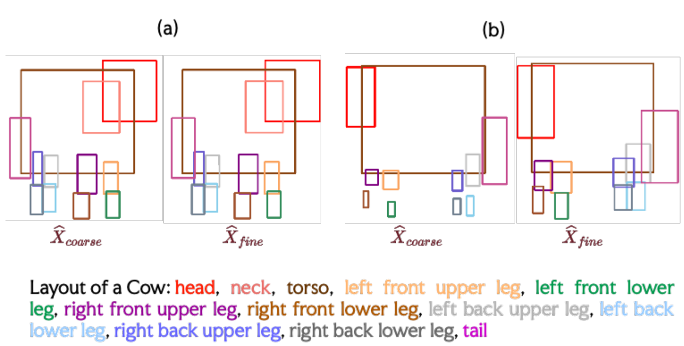


Figure 3.4 Effect of Differential Refinement loss: (section 3.5) In (a), $\mathcal{L}_{fine}$ is applied to both $\widehat{X}_{coarse}$ and $\widehat{X}_{fine}$. The layouts are almost identical. (b) For $\widehat{X}_{coarse}$, MSE loss is applied as $\mathcal{L}_{coarse}$ and $\mathcal{L}_{fine}$ is applied to $\widehat{X}_{fine}$. In this case, $\widehat{X}_{coarse}$ shows approximate placements and shapes of parts, and $\widehat{X}_{fine}$ shows the refined layouts with correct intersections between parts.

3.6 Incremental effects of Design Choices

To validate our methodology, we conducted a final ablation study to measure the cumulative effect of our proposed modifications on the model's reconstruction performance. We started with a baseline GCN-VAE and progressively integrated each of our primary contributions.

The results, presented in Table 3.5, show a clear and positive trend. The baseline model's performance is significantly improved by the introduction of the **Dynamic Margin IOU (DyI) loss**, which addresses the zero-gradient issue for non-intersecting boxes.

Although addition of **GCN Refinement Block** alone does on improve the reconstruction performance of the model, we observed that optimal reconstruction performance, is achieved only when this block is trained with the **Differential Refinement loss**. This final configuration, which leverages all of our proposed innovations, demonstrates the most accurate reconstruction of part layouts.

This analysis demonstrates that each design choice increases the model's overall effectiveness. The full generative performance of these choices and the final model will be evaluated in Chapter 5 using a comprehensive suite of metrics designed to assess structural integrity and layout quality.

Table 3.5 Effect of GCNVAE design choices

Metric	Baseline	+ DyI (a)	(a) + GCN Refinement Block (b)	**(b) + Differential Refinement Loss (c)**
$mIOU_{rec}$	.490	.535	.497	**.559**

Chapter 4

Generating Objects from Layouts

The first stage of our pipeline, PLayGen, successfully generates plausible and structurally coherent part-based layouts. The second and final stage, which is the focus of this chapter, is to translate these abstract layouts into high-fidelity, photorealistic object images. This process requires a generative model that can accurately interpret spatial instructions while synthesizing complex textural details. This chapter details the methodology for transforming a generated layout, $\widehat{X}_{fine}$, into a final image. We begin by justifying the selection of a conditional diffusion model architecture (fig. 4.1). We then describe the process of creating a custom-tuned ControlNet, including the design of the visual conditioning signal and the formulation of text prompts that guide the synthesis. Finally, we outline the fine-tuning process that enables our model to generate objects that faithfully adhere to the input part-based layout.

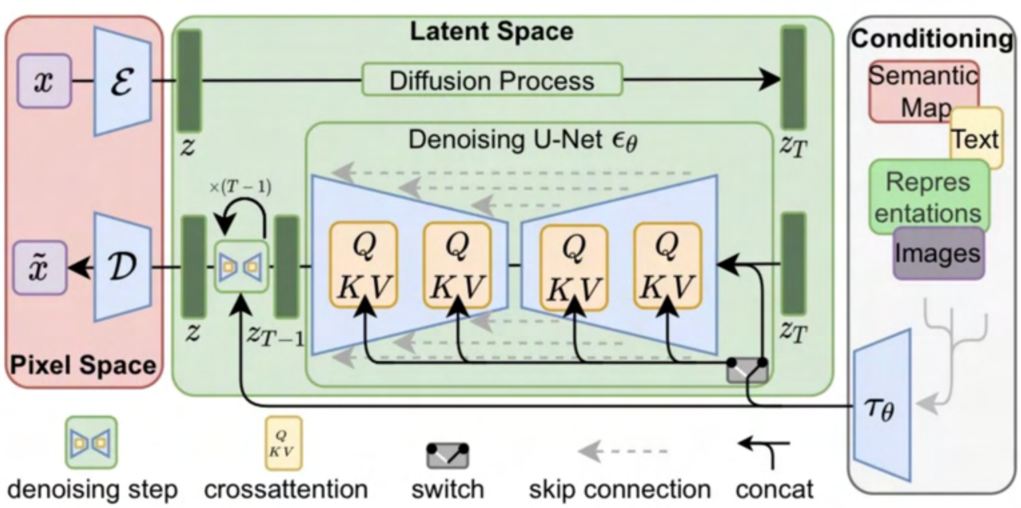


Figure 4.1 Diffusion Architecture with conditioning: These models enable generation using various conditioning methods like text, semantic maps, etc. and can also be adapted for custom representations.

4.1 Architecture: A Custom Inpainting ControlNet

To exercise precise control over the composition, pose, and layout of subjects in a generated image, we required an architecture that could be conditioned on an external spatial guide. Architectures like ControlNet [9] and GLIGEN [11] are designed for this purpose, steering a powerful pre-trained diffusion model to conform to a specified structure. Given its versatility and proven success with various control mechanisms (e.g., canny maps, keypoints, segmentation maps), we selected. The ControlNet architecture (fig. 4.2) is uniquely suited for our task. It works by creating a trainable copy of the encoding blocks of a large, pre-trained diffusion model (in our case, Stable Diffusion 2.1 [3]). This trainable encoder is tasked with processing our custom spatial control image. The original, frozen model's weights are preserved, retaining the vast knowledge of image synthesis it already possesses. The outputs from the trainable encoder are then fed into the frozen model's decoder via lightweight zero-convolution layers. This design allows the learned spatial guidance to be additively injected into the generation process without destabilizing the fine-tuned weights of the original model, ensuring stable training. To further

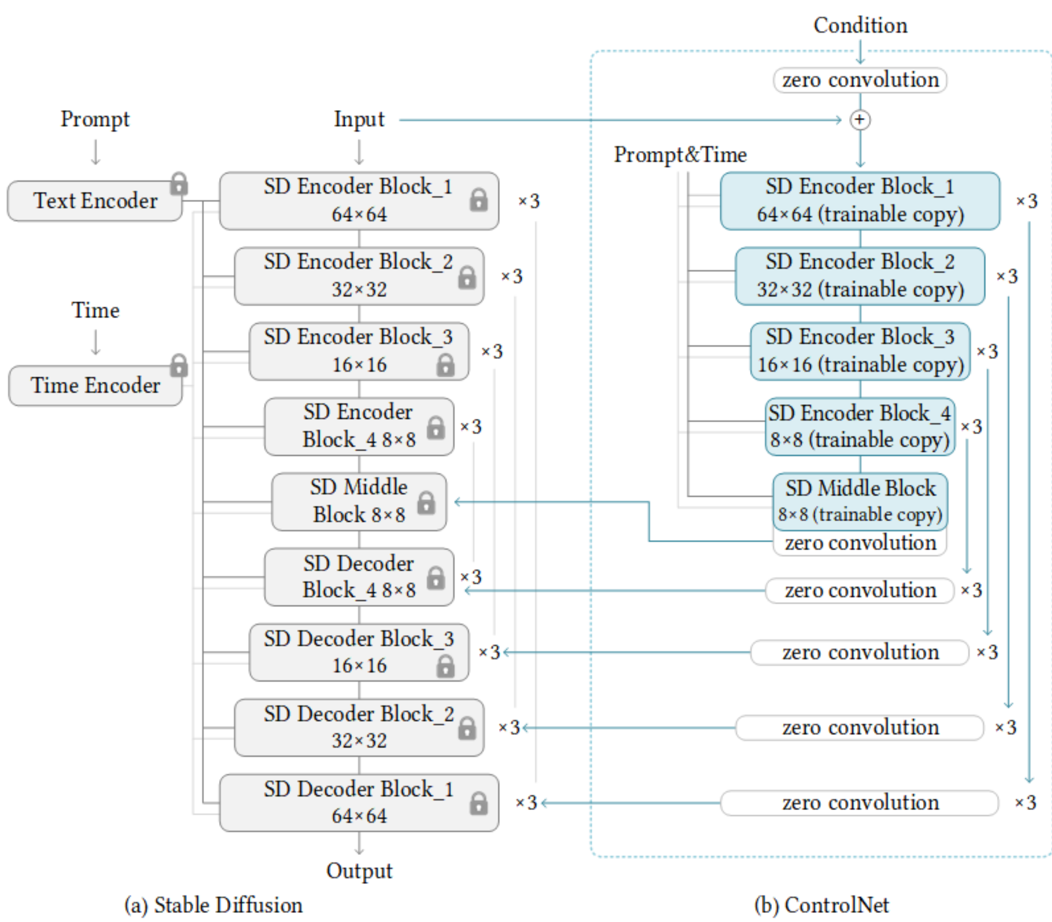


Figure 4.2 ControlNet Architecture: The zero convolution and encoder copy enables learning control for new representations while preserving general learning in the Stable Diffusion U-Net.

refine the generation process, we integrated our custom ControlNet into an inpainting pipeline. Along with the control image, we provide a binary mask corresponding to the overall bounding box of the object. This mask directs the model to limit the object generation in the desired region, preventing it from generating spurious artifacts in the background and ensuring the object is correctly positioned on the canvas.

Figure 4.3 Generating Objects: The part bounding box layout $\widehat{X}_{fine}$ generated by PLayGen is first transformed into a color-coded Control Image, with the boxes represented as filled ellipses. A masked version of the object's bounding boxes is created as Inpainting Mask. These control images and a text prompt with object and part list details is used to guide a custom fine-tuned Inpainting ControlNet diffusion model which generates the final object image.

4.2 Model Training and Fine-Tuning

Training a ControlNet encoder from scratch on our custom dataset proved insufficient due to the dataset's limited size, resulting in images that failed to achieve the desired part-level control or realism. To overcome this, we adopted a fine-tuning approach. We selected a pre-trained ControlNet checkpoint that was originally trained on segmentation maps from the ADE20K dataset, as this was the control type most analogous to our color-coded part layouts. Using the PASCAL-Parts dataset, we fine-tuned this model with our custom control images (inscribed ellipses) and descriptive text prompts. The training objective was masked to the object region using a latent transformation of the object mask, concentrating the loss calculation on the area of interest. This strategy allowed us to leverage the powerful prior knowledge of the pre-trained model and adapt it efficiently to our specific task of high-fidelity, part-controllable object generation.

The control image, its associated mask and a descriptive prompt generated from the part list and object category are then used as input to a custom configured Inpainting [37] ControlNet [9] pipeline to generate the final part and category conditioned object image. Here, the mask image controls the positioning of the object through inpainting on the blank canvas. The ControlNet component uses the control image and the descriptive prompt to generate fine-grained part-level details of the object. We compare the generation quality using a finetuned ControlNet and a U-Net initialized ControlNet trained only on our custom data set in (fig. 4.4).

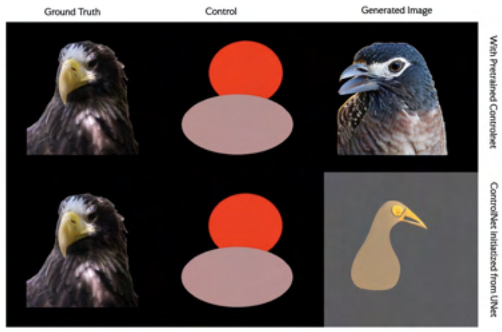


Figure 4.4 Comparing Finetuned Segmentation ControlNet vs. U-Net initialized ControlNet: Training a new ControlNet initialized from the diffusion U-Net weights for our custom conditioning format suffers from poor generation quality due to insufficient data size.

4.3 Conditioning Signal and Prompt Design

The effectiveness of ControlNet is entirely dependent on the quality of its conditioning signal. We therefore experimented with various formats for both the visual control image and the accompanying text prompt to identify the optimal combination for our task.

4.3.1 Visual Conditioning Format

The bounding box layout from PLayGen must first be converted into an image that the ControlNet encoder can interpret. We explored three different representations for this control image:

1. **Rectangular Boxes:** Each part bounding box is rendered as a filled, uniquely colored rectangle.
2. **Inscribed Ellipses:** An x-y oriented ellipse is inscribed within each part's bounding box and filled with a unique color.
3. **2D Point Distributions:** A set of normally distributed points is plotted, centered in each bounding box, with a spread proportional to the box's dimensions.

As shown in the comparative results in Figure 4.3, the **Inscribed Ellipses** provided the best performance. We hypothesize that the hard, artificial edges of rectangular boxes can improperly bias the model, whereas the softer, more organic shape of an ellipse provides a better approximation of a body parts without imposing unnatural geometric constraints that led to a cropping effect on the generated images.

4.3.2 Text Prompt Formulation

Alongside the visual guide, a text prompt provides crucial semantic context. We tested three levels of detail in our prompts:

1. **Basic prompt:** A segmented image of a <object> on a black background.

2. **Stylized prompt:** A segmented sprite of a real <object> with photo-realistic features and body structure on a black background.

3. **Prompt with parts:** A segmented sprite of a real <object> with photo-realistic details and body structure on a black background. Image shows: <object>'s <part1>, <part2>, <part3>, ...

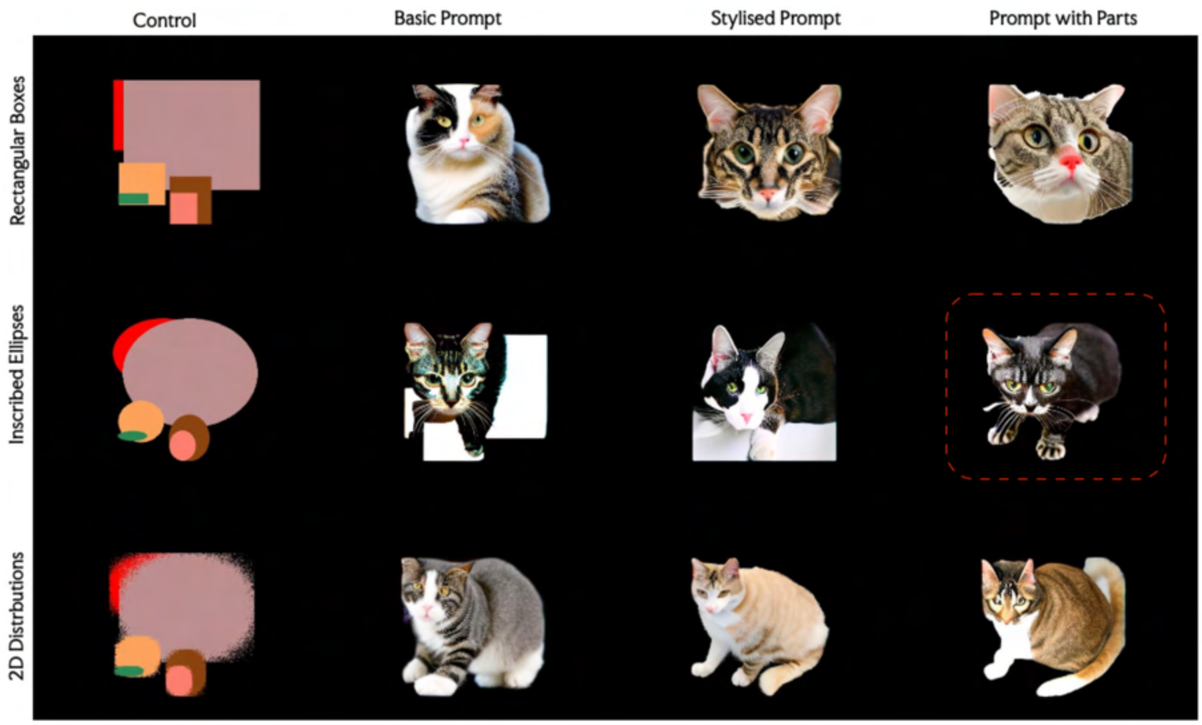


Figure 4.5 Effect of conditioning and prompt variations: Cat generations with the part list head, left front leg, right front leg, left front paw, right front paw for different conditioning and prompt types. The red dotted box represents the conditioning and promoting style we choose - Inscribed ellipses for control images and prompt with parts.

Our experiments demonstrated that the **Prompt with Parts** was the most effective. This detailed prompt reinforces the visual information contained in the control image, ensuring the model correctly identifies and renders each component, leading to superior realism and adherence to the input layout.

Chapter 5

Experimental Setup and Results

This chapter presents a comprehensive evaluation of our proposed framework, PLATO. We validate its effectiveness through a series of quantitative and qualitative experiments, comparing its performance against state-of-the-art models in layout and image generation. We discuss the datasets and the training setup used for this framework. To assess the quality of generated structures, we introduce a novel evaluation framework that includes new metrics designed to quantify their part-level fidelity and structural integrity. Key among these are the Disconnection Percentage (Disc%), which measures the structural coherence of generated layouts, alongside Part-Layout-IOU (PL-IoU) and Part-Layout-F1 (PL-F1), which assess the precise adherence of final images to the conditioning part layout. Using this robust framework, we evaluate PLATO against state-of-the-art models. The results demonstrate its superior ability to generate structurally coherent layouts and high-fidelity images with precise part-level control.

5.1 Experimental Setup

5.1.1 Datasets

For this work, the PASCAL-Parts [38] and PartImageNet [39] datasets were selected as they provide the granular, part-level annotations necessary for generating detailed object part layouts. PASCAL-Parts served as the primary dataset for training and quantitative evaluation, while PartImageNet was used for additional qualitative analysis. Our experiments focused on 10 object classes chosen for their non-trivial structural complexity: aeroplane, bicycle, bird, cat, cow, dog, horse, motorbike, person, and sheep. This dataset was preprocessed to remove extreme layouts that can worsen the quality of generation. This resulted in a working dataset of 3000 unique samples, containing a total of 30 unique part types across all categories that were condensed to a 16 part encoding. A detailed list of parts for each category can be found in table 5.1

To prepare the data for the model, a consistent pre-processing and augmentation pipeline was applied. Each object image was first cropped from its original scene, and the associated part masks were converted into bounding box representations. To expand the dataset and improve model robustness, data augmentation was performed using rotation and flipping. Finally, to ensure a standardized input scale for the network, all bounding box layouts were normalized relative to the largest object dimension. This dataset is used by the first stage of the architecture *PLayGen*.

Object Class	Part list
Aeroplane	body, right wing, engine 1, engine 2, wheel 1, wheel 2, wheel 3, wheel 4, tail
Bird	head, neck, torso, left wing, right wing, left leg, right leg left foot, right foot, tail
Bicycle	body, front wheel, back wheel, handlebar, saddle
Cat	head, neck, torso, left front leg, right front leg, left back leg, right back leg, left front paw, right front paw, left back paw, right back paw, tail
Cow	head, neck, left horn, right horn, torso, left front upper leg, left front lower leg, right front upper leg, right front lower leg, left back upper leg, left back lower leg, right back upper leg, right back lower leg, tail
Dog	head, torso, left front leg, left front paw, right front leg, right front paw, left back leg, left back paw, right back paw back leg leg, right back paw, muzzle, tail
Horse	head, neck, torso, left front upper leg, left front lower leg, left front hoof , right front upper leg, right front lower leg, right front hoof, left back upper leg, left back lower leg, right back upper leg, right back hoof right back lower leg, left back hoof, tail
Motorbike	body, front wheel, back wheel, handlebar, saddle
Person	head, neck, torso, left lower arm, left upper arm, left hand, right lower arm, right upper arm, right hand, left lower leg, left upper leg, left foot, right lower leg, right upper leg, right foot
Sheep	head, neck, left horn, right horn, torso, left front upper leg, left front lower leg, right front upper leg, right front lower leg, left back upper leg, left back lower leg, right back upper leg, right back lower leg, tail

Table 5.1 Object-wise list of parts

5.1.2 Training setup

There are two stages in the training process of the PLATO framework is divided, corresponding to the layout generation **PLayGen** and image synthesis **ControlNet** module. For **PLayGen**, a cyclic annealing process is used to balance the reconstruction and the KL divergence loss term using a λ multiplier to increase the weight of the KL term as the training progresses. Early in the training, the model operates as a denoising encoder, optimizing the reconstruction loss. At this stage, the KL loss is turned off and the value of λ is zero. To determine the training epoch where the variational objective can be turned on again, we use two criteria: the reconstruction error for the bounding boxes should be below a configurable threshold θ_1 and the difference between the training and validation reconstruction loss should be below another configurable threshold θ_2. When both these conditions are met the λ value progressively increases via the annealing process. For $\mathcal{L}_{fine}$ (section 3.5), we decay θ as training progresses to gradually reduce the contribution of MSE term.

For training the ControlNet Stable diffusion setup, we use the derived layouts to generate conditioning images as described in chapter 4 and the object images as targets. Each training example contains the layout control image, target object image, rectangular object mask that corresponds to the object region of the control image. The descriptive prompt containing the object class and parts present is used to further condition the generation. We use a Controlnet Inpainting training objective where loss is masked using an latent transformation of the object mask image. This setup allows the network to learn to extrapolate from the layout control image to the object independent of the background. When trained in this fashion, the framework is able to generate independent objects on a blank canvas, as well as inpainting them on a vibrant background image. We use a pretrained SD2.1 [3] as a base model and fine-tuned a Controlnet Segmentation checkpoint trained on ADE20K [40].

Component	Parameter	Value
PLayGen	Learning Rate	8e-5
	Batch Size	128
	Epochs	500
	Latent dimension size	64
	Number of GCN Refinement Iterations	2
ControlNet + InPainting	Learning Rate	5e-6
	Resolution	512
	Batch Size	6
	Gradient Accumulation Steps	3
	Epochs	5

Table 5.2 Parameter values used in training

5.1.3 Evaluation framework

To assess the efficacy of our proposed asset generation framework, we evaluate with a set of novel metrics that quantify layout fidelity, structural connectivity, and part-level control along with standard generation metrics that measure overall image quality.

5.1.3.1 Layout Fidelity and Structural Integrity

We evaluate the similarity of generated layouts to real-world layouts using Layout-FID (Fréchet Inception Distance) [41]. This metric quantifies the statistical distance between the feature distributions of generated and ground truth layouts in a learned embedding space. For feature extraction, we retrain the embedding model [42] on part layout data. While Layout-FID effectively measures the overall similarity of the generated layout structure to actual layouts, it may not fully capture the nuances of structural connectivity. To address this, we introduce a novel metric **Disconnection Percentage (Disc %)**. It is defined as the fraction of part box intersection pairs missing from a generated layout wrt ground truth object adjacency matrix. This metric penalizes generations containing part boxes disconnected from the main object.

5.1.3.2 Part-Level Control and Adherence

To evaluate the model's ability to adhere to input conditioning at the part level, we introduce two additional metrics that operate on parts detected in the final generated image. We use a pre-trained part segmentation pipeline [43] to detect part-level bounding boxes in the generated images. The first metric, **Part-Layout-IOU** (PL-IoU), calculates the average per-part Intersection over Union (IOU) between the bounding boxes of segmented parts in the generated output and their corresponding parts in the conditioning layout. A high PL-IOU score signifies that the generated layout is structurally consistent and can effectively guide the generation process. Further, to evaluate the model's ability to generate images with a precise list of parts in provided in the input, we introduce an additional **Part-Layout-F1** (PL-F1) score. Let the list of part labels in the generated image be P' (as detected by the part segmentation output). Let the input part list for generating the conditioning layout be P_c'. PL-F1 is defined similar to the standard F1-score as PL-F1 $= 2|P_c' \cap P_c|/(|P_c'| + |P_c|)$.

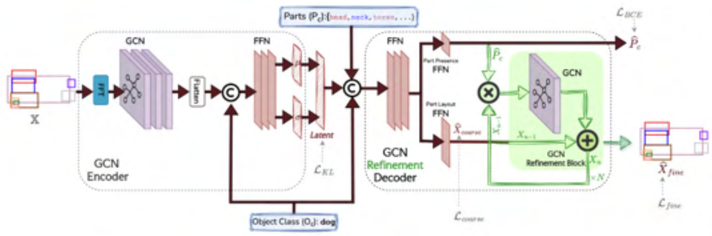
5.1.3.3 Overall Image Quality and Semantic alignment

We assess the overall visual quality of the generated images using the standard Fréchet Inception Distance (FID). To evaluate the semantic consistency between the generated image and the intended part composition, we use the multi-modal CLIP-Score [31]. This metric calculates the cosine similarity between the CLIP embeddings of the generated image and a corresponding text prompt that explicitly enumerates the list of expected parts. A higher CLIP-Score indicates a stronger semantic alignment between the visual content of the generated image and the textual description of its constituent parts.

5.1.4 Selection of Baselines

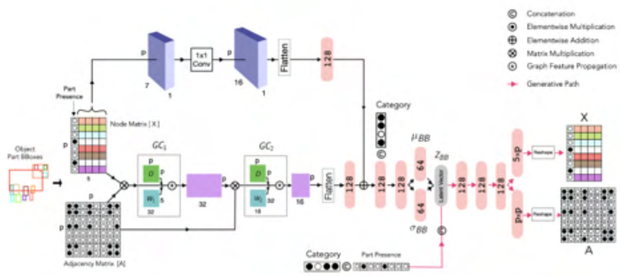
Given PLATO's two-stage generation process, we compare the performance of each stage with equivalent models and end-to-end generation with T2I models. Figure 5.2 compares the generation process of our selected baselines with PLATO.

For comparing the quality of **Layout Generation**, we select layout generation models for scenes and document layouts such as BLT [25], LayoutNuwa [44], LayoutFormer++ [26], LayoutDM [45] and the BoxGCNVAE of Meronymnet [28] to compare their results with PLayGen. To compare the **Layout-to-Image** generation quality of our custom-tuned ControlNet with others, we select conditioned generation models such as LayoutDiffusion [30], InstanceDiffusion [10] and GLIGEN [11]. In this evaluation, the conditioning input is the same for all and is generated using PLayGen. We compare **end-to-end** generation quality with T2I models like Stable Diffusion (SD) [3], Stable Diffusion XL (SDXL) [46] and FLUX [4].



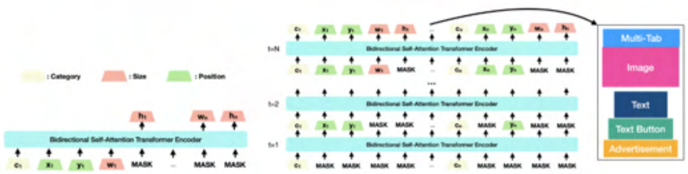
PLayGen

Model Type: GCN VAE
No. of Parameters: 180-200K
Base Model: None
Box Encoding: [p, x, y, w, h]
Training Type: Complete training
Encoder Type: GCN + FFN
Decoder Type: GCN refinement
Losses: BCE, MSE, Dyl, KL on Layouts



Meronymnet

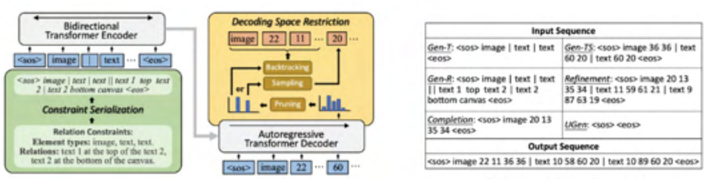
Model Type: GCN VAE
No. of Parameters: 180-200K
Base Model: None
Box Encoding: [p, x min, y min, x max, y max]
Training Type: Complete training
Encoder Type: GCN + FFN
Decoder Type: FFn
Losses: BCE, MSE, IOU, KL on Layouts



(a) BLT Training Phrase. (b) BLT Iterative Decoding Process.

BLT

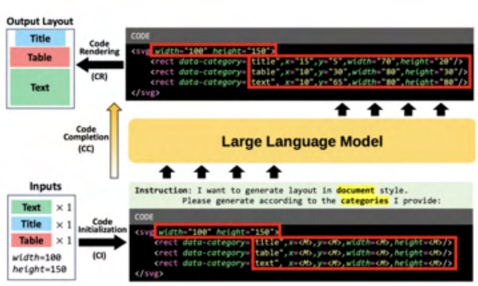
Model Type: Bilateral Transformer
No. of Parameters: 13M
Base Model: None
Box Encoding: [part, x, y, w, h] as tokens
Training Type: Complete training
Encoder Type: Transformer
Decoder Type: Transformer
Losses: Masked Token Prediction



(a) Model Architecture (b) Input and Output Sequences

LayoutFormer++

Model Type: Transformer
No. of Parameters: 59M
Base Model: None
Box Encoding: [part, pos, x, y, w, h] as tokens
Training Type: Complete training
Encoder Type: Bidirectional transformer
Decoder Type: Autoregressive Transformer
Losses: Next Token Prediction



LayoutNuwa

Model Type: LLM
No. of Parameters: 7B
Base Model: Llama 7B
Box Encoding: [part_type, x, y, w, h] as SVG tags
Training Type: Supervised Instruction Tuning
Encoder Type: -
Decoder Type: Transformer
Losses: Next Token Prediction

Figure 5.1 Layout Generator Architectures: Transformer based architectures like BLT [25] and LayoutFormer++ [26] do not capture interconnectedness of parts well. Whereas LLM based architectures like LayoutNuwa [44] treat the layout as a sequence of tokens. A GCN-VAE proves to be best suited to capture inter-part dependencies which is our chosen base architecture.

5.2 Discussion of Results

To highlight the improvements brought out by our proposed framework, we perform a comparative study of each stage — Layout Generation and Layout to Image Generation — along with the end to end performance. We also discuss the emergent properties of the framework that showcase its generalization of structural understanding as it pertains to poses of objects.

5.2.1 Layout Generation

The Layout-FID scores in table 5.3 suggest that PLayGen outperforms existing methods and its generations best emulate the distribution of real layouts. Similarly, the Disconnection Percentage (Disc%) scores show that PLayGen generates the largest proportion of correctly connected part layouts. PLayGen and MeronymNet [28] both use GCN-VAE and have a similar (though not identical) structure for the encoder. However, PLayGen's significantly higher scores can be attributed to the effectiveness of PLayGen's novel GCN Refinement block section 3.3 coupled with the novel DyI loss section 3.4, which collectively enhance part placements and part-relative refinements as demonstrated in the architecture discussion in chapter 3. PLayGen outperforms significantly larger models despite its compact size, highlighting the efficiency and effectiveness of our approach as shown in table 5.3.

Model	Layout-FID↓	Disconnection% ↓	Model Size ↓
LayoutFormer++ [26]	46.9	9.6	59 M
LayoutNuwa [44]	114.6	8.5	7 B
Meronymnet [28]	92.1	1.1	200 K
BLT [25]	182.7	44.5	13 M
LayoutDM [45]	30.3	0.5	12.4 M
PLayGen (Ours)	**26.3**	**0.3**	**200 K**

Table 5.3 Comparing part layout generation models

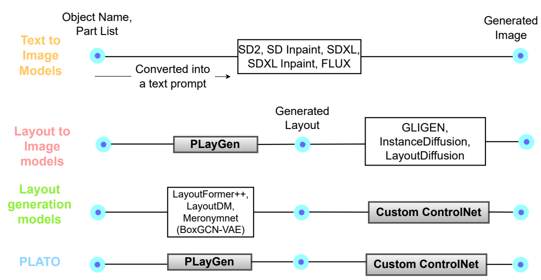
This result can be better understood when we compare the architectures of these generators. Looking at fig. 5.1, we see that both BLT and LayoutFormer++ use a transformer-based sequential approach for encoding and decoding bounding boxes which does not capture the inter-part relationships sufficiently with smaller data sizes. LayoutNuwa on the other hand finetunes an LLM to generate layouts with no emphasis on the structural interconnections.

5.2.2 Layout to Image Generation

For this comparison, we use the layout generated by PLayGen as a conditioning input for image generation models such as InstanceDiffusion [10], GLIGEN [11] and LayoutDiffusion [30]. Refer to pink rows in table of fig. 5.2. As evidenced by the higher PL-F1 and PL-IoU scores, we see that PLATO outperforms other models in generating objects consistent with input part layout. This is largely due to its ability to tackle the complexity in part-level conditioning when large number of parts are present, a task for which PLATO's generator is custom-tuned. The relatively worse scores for other models suggest limitations in their ability to faithfully translate structured layouts into coherent images.

5.2.3 Zero shot T2I Generation

For this comparison, we use a text prompt that contains both the object class and its associated part list to generate the object image. Since these models do not explicitly generate intermediate layouts, the Layout-IOU column is left blank in orange rows of the table in fig. 5.2. Once again, the results show PLATO's generations to be superior, particularly in its ability to render precisely the specified input parts. Among the T2I models, FLUX's slightly better Clip-Score can be attributed to its higher textural quality. However, FLUX's lower PL-F1 score shows that it struggles with part level controllability.



Model	FID ↓	PL-F1 ↑	Clip Score ↑	PL-IoU ↑
SD2 [34]	197.2	0.43	25.2	-
SD Inpaint [34]	197.0	0.14	25.7	-
SD XL [28]	191.8	0.57	23.9	-
SD XL Inpaint [28]	264.3	0.58	24.5	-
FLUX [18]	180.4	0.68	**28.8**	-
InstanceDiffusion [40]	192.5	0.60	22.6	0.05
GLIGEN [21]	162.8	0.66	21.2	0.06
LayoutDiffusion [42]	175.2	0.63	22.0	0.04
LayoutDM [12]	135.5	0.70	21.0	0.13
LayoutFormer++ [13]	156.5	0.52	24.8	0.04
Meronymnet (BoxGCN-VAE) [1]	164.0	0.64	27.0	0.18
PLATO (Ours)	**125.5**	**0.76**	28.0	**0.21**

Figure 5.2 Comparison of Baselines: (Right) compares image level metrics for several models. Zero shot text $\rightarrow$ image:orange, layout $\rightarrow$ image:pink, part $\rightarrow$ layout:lightgreen, PLATO:lightblue. (Left) shows the manner in which final image generations are obtained. Blue concentric circles denote inputs and outputs of components. Components reused from PLATO are shown in bold.

5.2.4 Qualitative Results and Analysis

While quantitative metrics validate the model's performance against predefined benchmarks, they do not fully capture the richness of its operational behavior. This section presents a qualitative analysis framed through the lens of emergent properties that lead to realistic resolution of the object.

5.2.4.1 Visual impact of Layout Generation Quality

In fig. 5.3, we study the efficacy of baseline layout generators as conditioning models by using their generated layouts in the object generation stage of PLATO. Layouts with poorly connected parts result in distorted generations or completely missed parts. Although the object generation model can accommodate minor connectivity violations, more egregious cases like that of `cow` and `bird` generated by LayoutFormer++ in fig. 5.3 result in unrecognizable objects. Out of order positional parts (left/right, upper/lower) and incorrect overlap between parts result in a complete loss of controllability. In these cases, due to conflicting conditioning input, the image generation pipeline produces images that only loosely follow the specified prompt text.

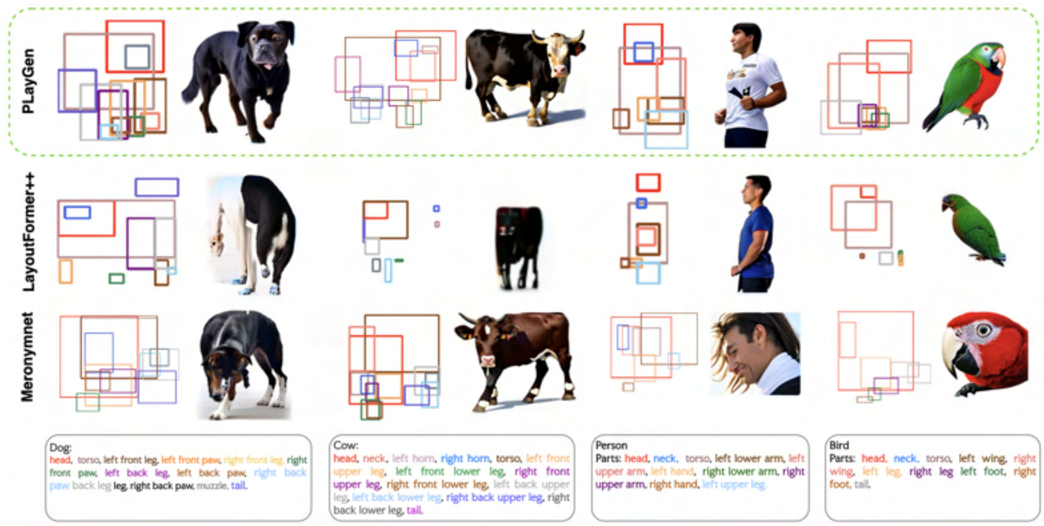


Figure 5.3 Qualitative comparisons: PLayGen layouts follows inter-part connectivity constraints and positional placement resulting in higher quality images. In case of Meronymnet, notice the left front paw for dog appearing above the head, disconnected from the left front leg resulting in distorted face and front legs. Also the relatively disproportionate of parts seen in case of the birds layouts lead to missing parts. In case of LayoutFormer++, we see that disconnected parts result in phantom limbs and missing head.

5.2.4.2 Effect of Variable Part list

As fig. 5.4 shows, PLayGen is able to generate more plausible part layouts compared to other layout generation models which enables more realistic generated objects. This results also highlights PLayGen's ability to correctly resolve part geometries and interconnections (e.g. the realistic arrangement of upper/lower arms of `person` and the directional resolution of left/right horns of `cow`), to generate high quality layouts for rigid and non-rigid objects.

5.2.4.3 Implicit Pose Understanding

An effect of part controllability is demonstrated in fig. 5.5 where a certain list of parts results in a specific orientation of the object. For example, if the right wing of a `bird` is visible but left is not, the only plausible pose is with the bird facing right. We show that this holds true across objects of varying complexity of parts, showcasing our proposed framework's ability for implicit pose control.

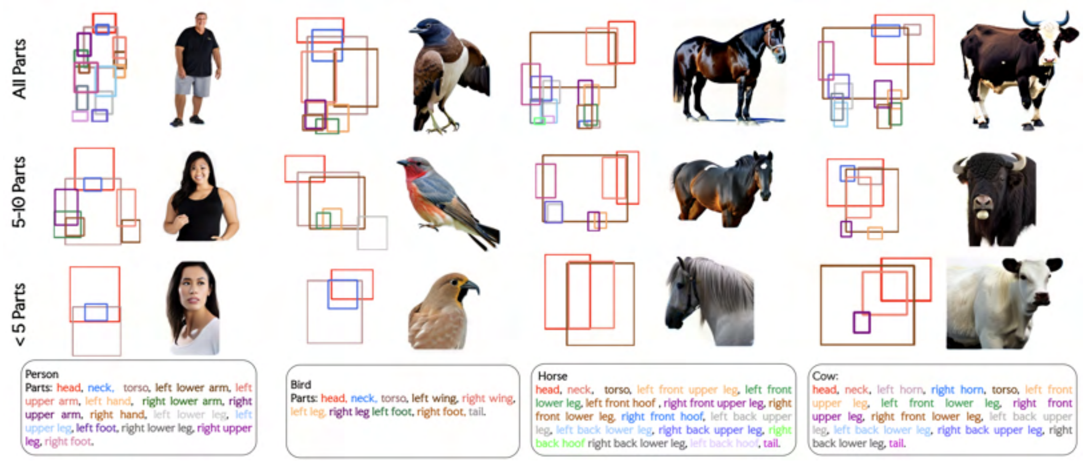


Figure 5.4 Handling variable part list: Our model is able to correctly resolve left/right upper/lower placement for any given part list. Notice in case of cow, that the orientation of horns is always maintained. The person layouts has all sub-parts of arms and legs connected in correct sequence.

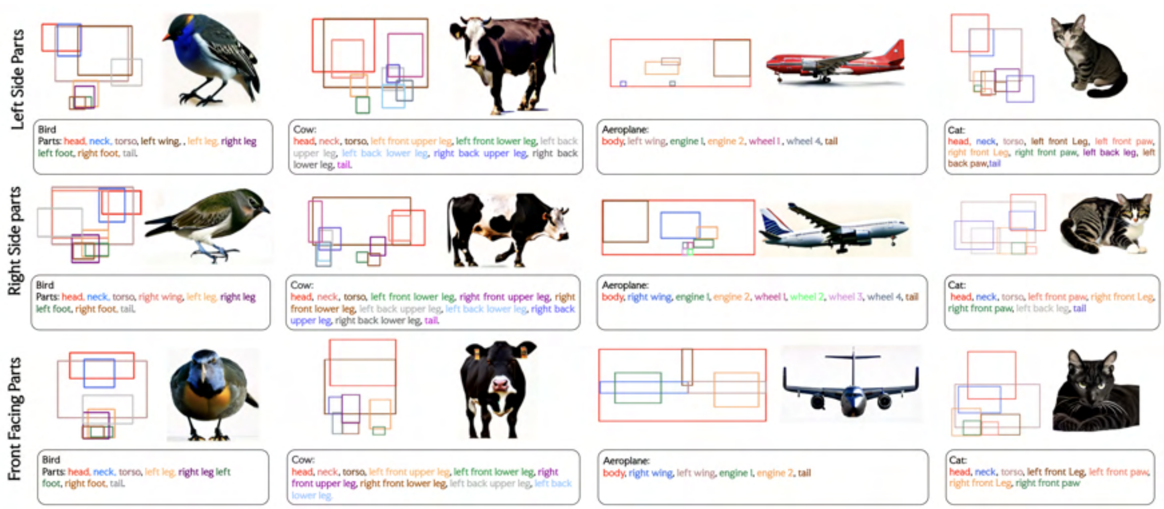


Figure 5.5 Implicit Pose understanding: We show that different configurations of part list result in pose variations. Layouts where left side parts are more visible than right lead to a left facing object as seen in top row. Similarly, layouts with mostly front parts leads to a front facing object.

5.2.4.4 Effect of Incoherent Layouts

There are two types of errors seen in the Layout generation stage. 1) **Disconnected layout** where one or more parts are disconnected from the main object layout. In cases where the disconnected part is small and far from the main layout, the ControlNet model tends to hallucinate a connecting part digressing from the user provided input or ignore this part, resulting in it being missing in the final generated image. 2) **Unresolved or highly overlapping parts**, where parts are not sufficiently differentiated in the generated layout. This causes the conditioning image to have multiple overlapping parts because of which a deformed object is generated.

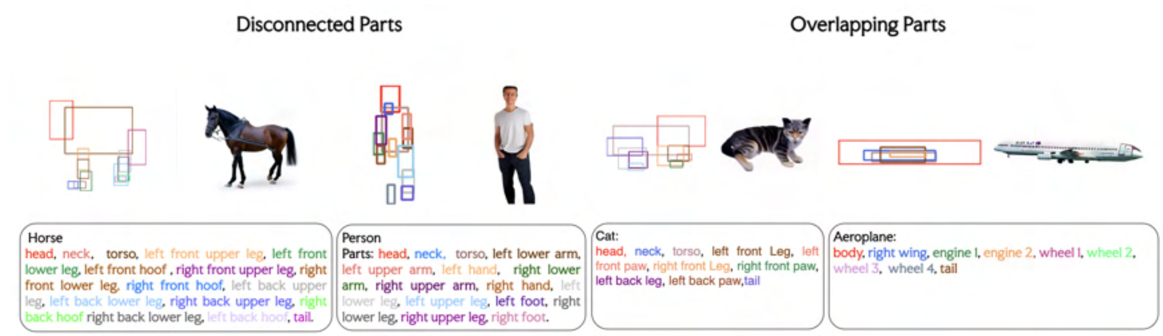
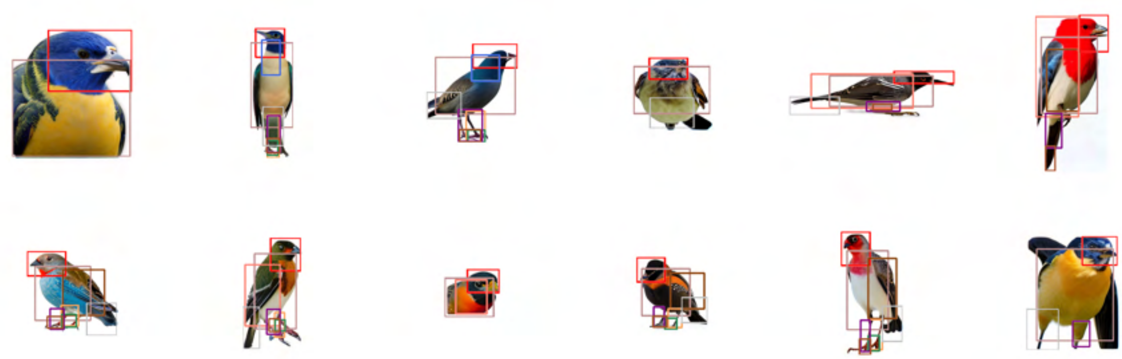
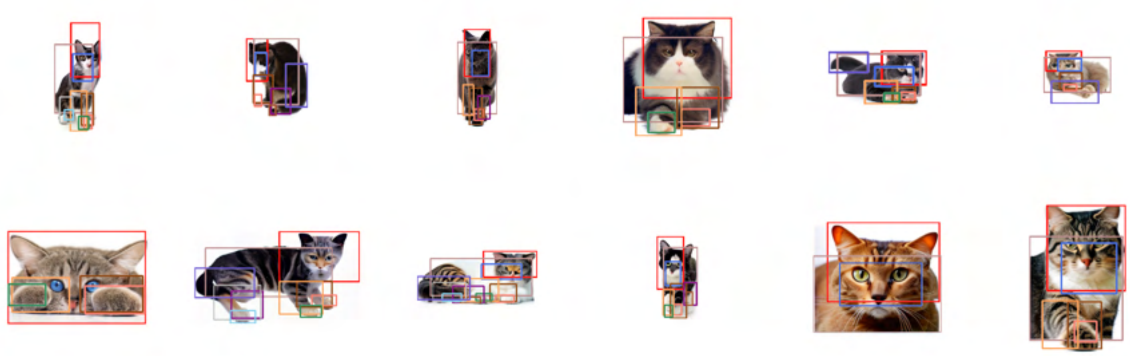


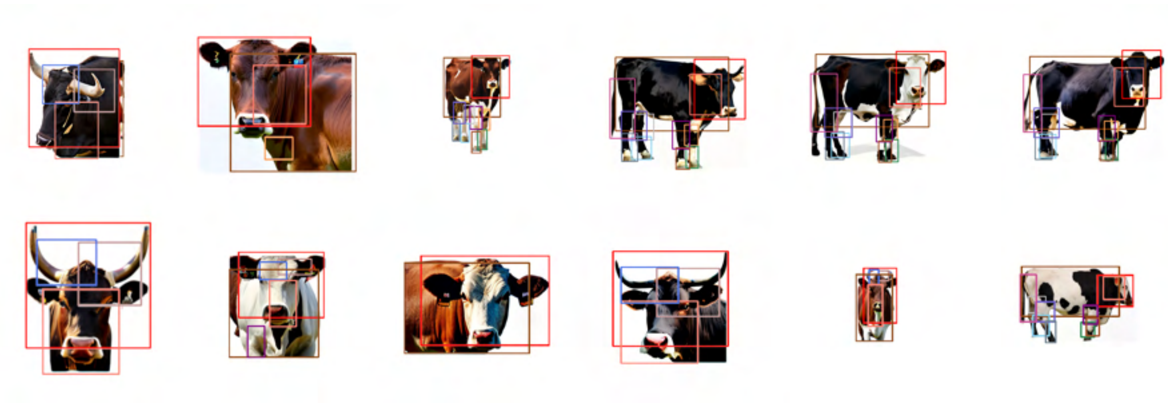
Figure 5.6 Resolution of incoherent part layouts: In the first two examples, parts not present in the output conditioning are hallucinated - right front upper leg in case of horse and and right upper leg in case of person. In the third example the cat's back legs are fused in due to overlapping bounding boxes. In case of the aeroplane, the overlapping bounding boxes are leading to poor resolution of direction resulting in image having two fronts.

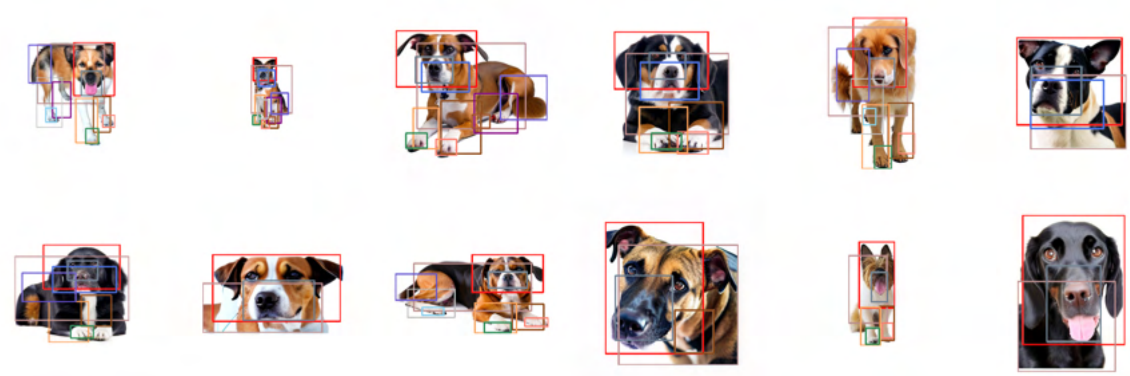
5.3 Examples of E2E generation

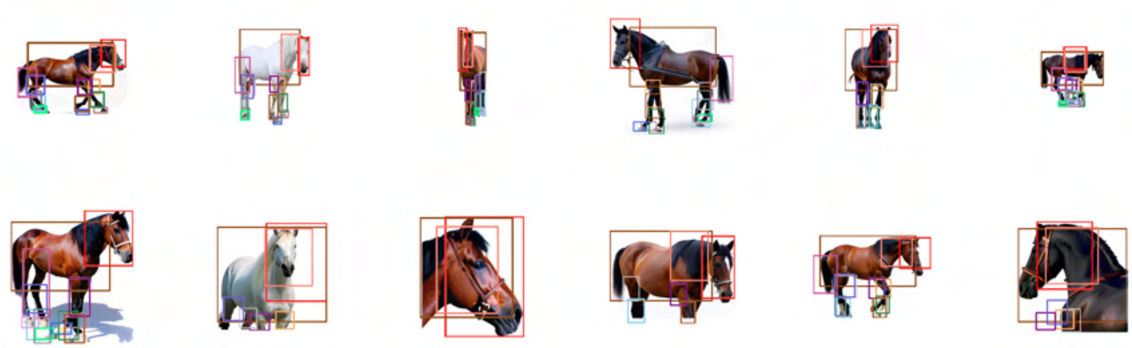
To further showcase the diversity of objects generated through our pipeline, we present a set of generations for each class. These cover different number of parts present in the generated layout, resulting in multiple object poses.

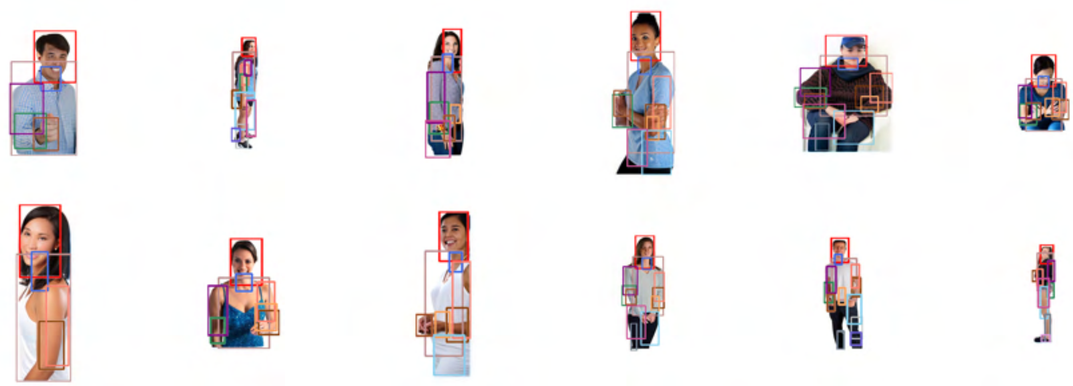


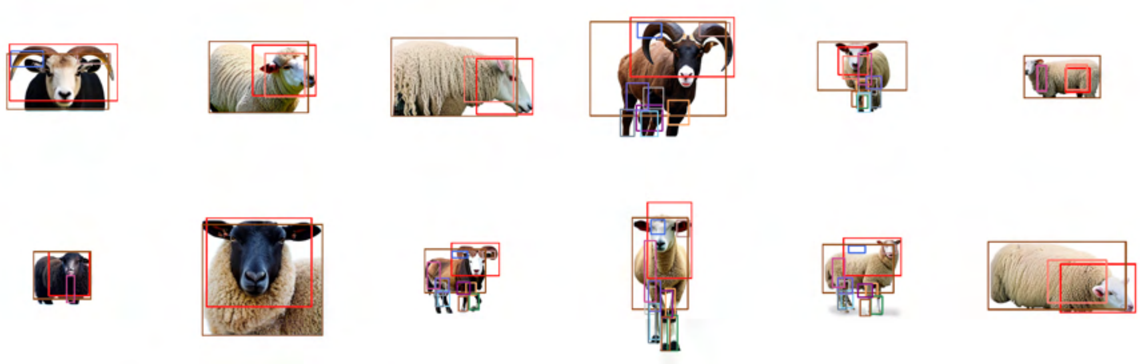












Chapter 6

Conclusion and Future works

In this chapter, we summarize the research contribution of this work and discuss further directions of research that can be undertaken towards developing a robust generalizable controllable generation framework.

6.1 Summary of Contributions

In this thesis, we focused on the task of Controllable Object Generation using simple user inputs such as an object class and associated part list. In applications like animation graphic design, such level of control on the generation process can enable the use of Large generation models without being limited to scene based generation. Towards this goal, we focused on two key areas of work: 1) A Stochastic layout generation model that improves structurally consistency across categories 2) Conditioning models to utilize this layout for creating anatomically consistent objects.

In PLayGen, a GCN VAE based layout generator, we introduce a novel GCN Refinement decoder for iterative refinement of object part layouts. We also introduce Dynamic Margin IOU loss for improving reconstruction quality of small parts in object layouts and Differential Refinement loss scheme to enable effective refinement and stabilize the overall training procedure. The combination of these improvements lead to high-fidelity, part-consistent object generation across diverse categories, from animate beings to inanimate objects. Although introduced for part layouts, PLayGen's novel formulation and compute-efficient design could also benefit layout-aware generative applications in other domains.

For the purpose of converting the object part layouts to images, we developed a Custom ControlNet extension that effectively uses the layout for generating anatomically consistent objects. We studied the effect of different finetuning, conditioning and prompting styles to develop a framework that works both for independent and scene conditioned object generation using a combination of ControlNet and Inpainting. The design of this framework allows it to be used for both large and small models thus allowing a versatile use for different levels of generation quality.

To demonstrate the efficacy of our two stage framework, we developed metrics to quantify structural consistency of layouts and generated such as the Disconnection Percentage which measures adherence to expected connectedness between parts of a layout and Part Layout F1 score which measure how accurate are the parts generated in the final image. Along with these, we use the Image-FID and Layout-FID to measure overall quality and CLIP score to quantify prompt adherence. Through this robust evaluation framework, we demonstrate PLATO's ability to generate realistic, anatomically accurate images, even with complex configurations.

Thus, PLATO is a novel framework for object generation that addresses limitations of existing text-to-image models by enabling precise part-level control. It opens new directions for expanding object categories and refining part-level detail towards controllable image synthesis in content creation and design.

6.2 Future Works

- **Improving Generalization and Granularity with synthetic data:** By using Larger generative model like FLUX a dataset of large variety of objects and poses can be created fo which part level segmentation can be obtained using prompt based segmentation pipelines like SAM and DINO which can segment parts for any customized part list.

- **Architectural improvements:** With larger datasets the architectural complexity, such as the use of Graph transformer, can be added to PLayGen. Further, the conditioning mechanism can be improved by using parts embedding networks to replace one hot encoding with text embeddings for parts. This can increase generalization across similar part names and create a more natural generation process.

- **Agentic Generation framework:** LLMs can be used to infer required parts from user prompts which can enable a more conversational and iterative generalization process that can be extended for creation of custom character structures.

- **Iterative Part level Editing:** Modifying the reconstruction objective of the Variational auto-encoder to a graph completion task can result in a framework that is capable of editing the graph from user instructions. This will allow for a more versatile creativity tool.

Bibliography

[1] A. Ramesh, M. Pavlov, G. Goh, S. Gray, C. Voss, A. Radford, M. Chen, and I. Sutskever, “Zero-shot text-to-image generation,” 2021. [Online]. Available: https://arxiv.org/abs/2102.12092

[2] C. Saharia, W. Chan, S. Saxena, L. Li, J. Whang, E. Denton, S. K. S. Ghasemipour, B. K. Ayan, S. S. Mahdavi, R. G. Lopes, T. Salimans, J. Ho, D. J. Fleet, and M. Norouzi, “Photorealistic text-to-image diffusion models with deep language understanding,” 2022. [Online]. Available: https://arxiv.org/abs/2205.11487

[3] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, “High-resolution image synthesis with latent diffusion models,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 10 684–10 695.

[4] B. F. Labs, “Flux,” https://github.com/black-forest-labs/flux, 2024.

[5] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial networks,” *Communications of the ACM*, vol. 63, no. 11, pp. 139–144, 2020.

[6] D. P. Kingma, “Auto-encoding variational bayes,” *arXiv preprint arXiv:1312.6114*, 2013.

[7] T. N. Kipf and M. Welling, “Variational graph auto-encoders,” 2016. [Online]. Available: https://arxiv.org/abs/1611.07308

[8] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever, “Learning transferable visual models from natural language supervision,” 2021. [Online]. Available: https://arxiv.org/abs/2103.00020

[9] L. Zhang, A. Rao, and M. Agrawala, “Adding conditional control to text-to-image diffusion models,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2023, pp. 3836–3847.

[10] X. Wang, T. Darrell, S. S. Rambhatla, R. Girdhar, and I. Misra, “Instancediffusion: Instance-level control for image generation,” 2024. [Online]. Available: https://arxiv.org/abs/2402.03290

[11] Y. Li, H. Liu, Q. Wu, F. Mu, J. Yang, J. Gao, C. Li, and Y. J. Lee, “Gligen: Open-set grounded text-to-image generation,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 22 511–22 521.

[12] A. Dosovitskiy and T. Brox, “Generating images with perceptual similarity metrics based on deep networks,” 2016. [Online]. Available: https://arxiv.org/abs/1602.02644

[13] A. B. L. Larsen, S. K. Sønderby, H. Larochelle, and O. Winther, “Autoencoding beyond pixels using a learned similarity metric,” 2016. [Online]. Available: https://arxiv.org/abs/1512.09300

[14] J. Ho, A. Jain, and P. Abbeel, “Denoising diffusion probabilistic models,” 2020. [Online]. Available: https://arxiv.org/abs/2006.11239

[15] J. Sohl-Dickstein, E. A. Weiss, N. Maheswaranathan, and S. Ganguli, “Deep unsupervised learning using nonequilibrium thermodynamics,” 2015. [Online]. Available: https://arxiv.org/abs/1503.03585

[16] O. Ronneberger, P. Fischer, and T. Brox, “U-net: Convolutional networks for biomedical image segmentation,” 2015. [Online]. Available: https://arxiv.org/abs/1505.04597

[17] P. Esser, S. Kulal, A. Blattmann, R. Entezari, J. Müller, H. Saini, Y. Levi, D. Lorenz, A. Sauer, F. Boesel, D. Podell, T. Dockhorn, Z. English, K. Lacey, A. Goodwin, Y. Marek, and R. Rombach, “Scaling rectified flow transformers for high-resolution image synthesis,” 2024. [Online]. Available: https://arxiv.org/abs/2403.03206

[18] F. Liu, G. Emerson, and N. Collier, “Visual spatial reasoning,” *Transactions of the Association for Computational Linguistics*, vol. 11, pp. 635–651, 2023. [Online]. Available: https://aclanthology.org/2023.tacl-1.37/

[19] K. Zhou, E. Lai, W. B. A. Yeong, K. Mouratidis, and J. Jiang, “Rome: Evaluating pre-trained vision-language models on reasoning beyond visual common sense,” 2023. [Online]. Available: https://arxiv.org/abs/2310.19301

[20] S. A. Haider, S. Prabha, C. A. Gomez-Cabello, S. Borna, S. M. Pressman, A. Genovese, M. Trabilsy, A. Galvao, K. T. Aziz, P. M. Murray, Y. Parte, Y. Yu, C. Tao, and A. J. Forte, “A validity analysis of text-to-image generative artificial intelligence models for craniofacial anatomy illustration.” *Journal of clinical medicine*, vol. 14, 2025.

[21] A. Izadi, M. A. Banayeeanzade, F. Askari, A. Rahimiakbar, M. M. Vahedi, H. Hasani, and M. S. Baghshah, “Visual structures helps visual reasoning: Addressing the binding problem in vlms,” 2025. [Online]. Available: https://arxiv.org/abs/2506.22146

[22] P. Rahmanzadehgervi, L. Bolton, M. R. Taesiri, and A. T. Nguyen, “Vision language models are blind: Failing to translate detailed visual features into words,” 2025. [Online]. Available: https://arxiv.org/abs/2407.06581

[23] X. Guo, Z. Huang, Z. Shi, Z. Song, and J. Zhang, "Your vision-language model can't even count to 20: Exposing the failures of vlms in compositional counting," 2025. [Online]. Available: https://arxiv.org/abs/2510.04401

[24] I. Stogiannidis, S. McDonagh, and S. A. Tsaftaris, "Mind the gap: Benchmarking spatial reasoning in vision-language models," 2025. [Online]. Available: https://arxiv.org/abs/2503.19707

[25] X. Kong, L. Jiang, H. Chang, H. Zhang, Y. Hao, H. Gong, and I. Essa, "Blt: Bidirectional layout transformer for controllable layout generation," 2022. [Online]. Available: https://arxiv.org/abs/2112.05112

[26] Z. Jiang, J. Guo, S. Sun, H. Deng, Z. Wu, V. Mijovic, Z. J. Yang, J.-G. Lou, and D. Zhang, "Layoutformer++: Conditional graphic layout generation via constraint serialization and decoding space restriction," 2023. [Online]. Available: https://arxiv.org/abs/2208.08037

[27] J. Li, J. Yang, A. Hertzmann, J. Zhang, and T. Xu, "Layoutgan: Generating graphic layouts with wireframe discriminators," 2019. [Online]. Available: https://arxiv.org/abs/1901.06767

[28] R. Baghel, A. Trivedi, and R. K. Ravichandran, Tejas a nd Sarvadevabhatla, "Meronymnet: A hierarchical model for unified and controllable multi-category object generation," in *Proceedings of the 29th ACM International Conference on Multimedia*, 2021, pp. 318–326.

[29] M. Fréchet, "Sur la distance de deux lois de probabilité."

[30] G. Zheng, X. Zhou, X. Li, Z. Qi, Y. Shan, and X. Li, "Layoutdiffusion: Controllable diffusion model for layout-to-image generation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2023, pp. 22 490–22 499.

[31] J. Hessel, A. Holtzman, M. Forbes, R. L. Bras, and Y. Choi, "Clipscore: A reference-free evaluation metric for image captioning," 2022. [Online]. Available: https://arxiv.org/abs/2104.08718

[32] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," 2017. [Online]. Available: https://arxiv.org/abs/1609.02907

[33] M. Tancik, P. Srinivasan, B. Mildenhall, S. Fridovich-Keil, N. Raghavan, U. Singhal, R. Ramamoorthi, J. Barron, and R. Ng, "Fourier features let networks learn high frequency functions in low dimensional domains," *Advances in neural information processing systems*, vol. 33, pp. 7537–7547, 2020.

[34] Z. Zheng, P. Wang, D. Ren, W. Liu, R. Ye, Q. Hu, and W. Zuo, "Enhancing geometric factors in model learning and inference for object detection and instance segmentation," 2021. [Online]. Available: https://arxiv.org/abs/2005.03572

[35] Z. Zheng, P. Wang, W. Liu, J. Li, R. Ye, and D. Ren, "Distance-iou loss: Faster and better learning for bounding box regression," 2019. [Online]. Available: https://arxiv.org/abs/1911.08287

[36] H. Rezatofighi, N. Tsoi, J. Gwak, A. Sadeghian, I. Reid, and S. Savarese, "Generalized intersection over union: A metric and a loss for bounding box regression," 2019. [Online]. Available: https://arxiv.org/abs/1902.09630

[37] A. Lugmayr, M. Danelljan, A. Romero, F. Yu, R. Timofte, and L. Van Gool, "Repaint: Inpainting using denoising diffusion probabilistic models," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 11 461–11 471.

[38] X. Chen, R. Mottaghi, X. Liu, S. Fidler, R. Urtasun, and A. Yuille, "Detect what you can: Detecting and representing objects using holistic models and body parts," 2014. [Online]. Available: https://arxiv.org/abs/1406.2031

[39] J. He, S. Yang, S. Yang, A. Kortylewski, X. Yuan, J.-N. Chen, S. Liu, C. Yang, Q. Yu, and A. Yuille, "Partimagenet: A large, high-quality dataset of parts," 2022. [Online]. Available: https://arxiv.org/abs/2112.00933

[40] B. Zhou, H. Zhao, X. Puig, T. Xiao, S. Fidler, A. Barriuso, and A. Torralba, "Semantic understanding of scenes through the ade20k dataset," 2018. [Online]. Available: https://arxiv.org/abs/1608.05442

[41] H.-Y. Lee, L. Jiang, I. Essa, P. B. Le, H. Gong, M.-H. Yang, and W. Yang, "Neural design network: Graphic layout generation with constraints," 2020. [Online]. Available: https://arxiv.org/abs/1912.09421

[42] K. Kikuchi, E. Simo-Serra, M. Otani, and K. Yamaguchi, "Constrained graphic layout generation via latent optimization," in *Proceedings of the 29th ACM International Conference on Multimedia*, ser. MM '21. ACM, Oct. 2021, p. 88–96. [Online]. Available: http://dx.doi.org/10.1145/3474085.3475497

[43] P. Gupta, R. Singh, P. Shenoy, and R. Sarvadevabhatla, "Olaf: A plug-and-play framework for enhanced multi-object multi-part scene parsing," 2024. [Online]. Available: https://arxiv.org/abs/2411.02858

[44] Z. Tang, C. Wu, J. Li, and N. Duan, "Layoutnuwa: Revealing the hidden layout expertise of large language models," 2023. [Online]. Available: https://arxiv.org/abs/2309.09506

[45] N. Inoue, K. Kikuchi, E. Simo-Serra, M. Otani, and K. Yamaguchi, "LayoutDM: Discrete Diffusion Model for Controllable Layout Generation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023, pp. 10 167–10 176.

[46] D. Podell, Z. English, K. Lacey, A. Blattmann, T. Dockhorn, J. Müller, J. Penna, and R. Rombach, "Sdxl: Improving latent diffusion models for high-resolution image synthesis," 2023. [Online]. Available: https://arxiv.org/abs/2307.01952