



Cross-Domain Latent Conditioning for Coherent Generative Synthesis

Doctoral Thesis

by

Adon Phillips

Thesis submitted to the University of Ottawa in partial fulfillment of the requirements
for the Doctor of Philosophy degree in Computer Science

School of Electrical Engineering and Computer Science
Faculty of Engineering
University of Ottawa

© Adon Phillips, Ottawa, Canada, 2025

“

*"If you wish to make an apple pie from scratch,
you must first invent the universe."*

— **Carl Sagan**, *Cosmos*
(Astronomer, Scientist)

ABSTRACT

Generative models have revolutionized synthesis research, enabling machines to produce images and text that rival human-created content. However, challenges in controllability, diversity, and coherence limit their application. This thesis investigates generative models with applications to multiple domains: texture synthesis, texture concept blending using vision-language model latent spaces, and language generation for procedural dialogue in computer role-playing games.

In texture synthesis, the thesis introduces a novel method for generating diverse, high-quality non-homogeneous textures from a single exemplar. The approach combines a Generative Adversarial Network (GAN) and a Variational Autoencoder within a unified architecture, that uses custom layers to share information during training. This method also presents a similarity loss term that promotes output diversity while enhancing quality. The architecture enables rapid training, producing superior results in less time when compared to state-of-the-art techniques.

The challenges of texture synthesis extend into the text domain, where describing complex textures is often difficult. This inspired the hypothesis that vision-language models, such as Contrastive Language-Image Pretraining (CLIP), might encode spatial information that could enable latent exploration, akin to GAN latent space traversal. To explore this, the thesis introduces a novel pipeline for constructing latent manifolds from a texture dataset that uses the CLIP model. This pipeline enables spatially optimized interpolation between user-selected inputs, for the generation of coherent and novel blends of visual concepts that are challenging to describe with text prompts.

Finally, using text synthesis for procedural dialogue generation, the thesis presents a structured framework that overcomes the limitations of traditional parser-based systems by integrating Large Language Models (LLMs) with a graph-based state tracking approach for use in computer role playing games. This approach frames interactions as puzzles with goal-based milestones to enforce narrative progression. The method empowers designers to craft abstract rules that enable players to invent solutions rather than adhere to predefined paths. A proof-of-concept game demonstrates how this system realistically enhances player freedom, validated through both qualitative and quantitative analysis. This method represents a move towards the long-standing goal of improving player agency in games.

ACKNOWLEDGEMENTS

Thank you to all whose presence, input, or support, whether direct or indirect, contributed to the completion of this thesis. This includes those at large who provided tools, hardware, software, source code, and data.

I acknowledge the support of the Natural Sciences and Engineering Research Council of Canada (NSERC).

TABLE OF CONTENTS

List of Figures **ix**

List of Tables **xi**

1 Introduction **1**

1.1 Context 4

 1.1.1 Texture Generation 4

 1.1.2 Dialogue Generation 5

1.2 Contributions 6

2 Background **10**

2.1 Generative Adversarial Networks 11

2.2 Variational Auto-Encoders 12

2.3 Latent Diffusion Models 13

2.4 LLMs and Text Transformers 15

 2.4.1 CLIP 17

3 Literature Review **19**

3.1 Texture Synthesis 20

 3.1.1 Traditional Example-based Methods 20

 3.1.2 Deep Generative Methods 23

3.2 Controllability of Texture Synthesis 29

 3.2.1 Conditional Image Synthesis 30

 3.2.2 Style Transfer 33

 3.2.3 Latent Space Navigation 34

3.3	CLIP	36
3.3.1	Application to Generative Models	37
3.3.2	Texture Synthesis Using CLIP-connected models	38
3.3.3	Multi-Modal Models Beyond CLIP	38
3.3.4	Metrics	39
3.4	Dialogue Control	39
3.4.1	Branching Tree and Rule-based Dialogue Systems	39
3.4.2	Learning-based Dialogue Control	40
3.4.3	Prompting Techniques in Dialogue Generation	41
3.4.4	Dialogue Generation	43
4	Diverse Non-homogeneous Texture Synthesis From A Single Exemplar	44
4.1	Introduction	46
4.2	Related Work	47
4.2.1	Single Exemplar Texture Synthesis GANs	47
4.2.2	Unconditional Single-Image GANs	49
4.3	Method	50
4.3.1	Network	51
4.3.2	Sampled Similarity Loss	55
4.3.3	Combined Diversity/Quality Metric	56
4.4	Experimental Results	57
4.4.1	Experimental Settings	58
4.4.2	Texture Synthesis Methods	58
4.4.3	Single-Image GAN Baselines	60
4.4.4	Training Time	62
4.4.5	Quantitative Evaluation	63
4.4.6	Limitations	68
4.5	Conclusion	68
4.5.1	Future Work	69
4.6	Summary	69

5	Blending Difficult to Describe Concepts in CLIP-Space	70
5.1	Introduction	71
5.1.1	Motivation and Background	73
5.1.2	Contributions	74
5.2	Related Work	75
5.3	Methodology	78
5.3.1	Overview	78
5.3.2	Construction of the Graph-Based Embedding Representation	82
5.3.3	Interpolation via Spatial Optimization	89
5.4	Experimental Setup	96
5.4.1	Datasets	96
5.4.2	Baselines	96
5.4.3	Model Selection and Embedding Generation	97
5.4.4	Evaluation Metrics	97
5.4.5	Implementation Details	97
5.5	Results and Discussion	98
5.5.1	Prompt Optimization for Graph Construction	98
5.5.2	Spatial Interpolation	100
5.5.3	Limitations	107
5.6	Conclusion	108
5.6.1	Future Work	109
5.6.2	Summary	110
6	Goal-oriented Conversational Games Using LLMs	111
6.1	Introduction	112
6.2	Background and Related Work	116
6.2.1	Agency	116
6.2.2	Graph Representations	118
6.2.3	LLMs	119
6.2.4	Tracking	120
6.3	Method	121

6.3.1	Motivation	121
6.3.2	Framework	122
6.3.3	Prompt Structure	127
6.3.4	Evaluation Approach	128
6.4	Proof-of-concept Game	129
6.4.1	Implementation Details	130
6.5	Results and Discussion	130
6.5.1	State Management Experiment	132
6.5.2	Gameplay Analysis	133
6.5.3	Discussion	134
6.6	Conclusion	138
6.6.1	Summary	140
7	Conclusions	141
7.1	Findings	141
7.1.1	Diverse Non-homogeneous Texture Synthesis From A Single Ex- emplar	141
7.1.2	Blending Difficult to Describe Concepts in CLIP-Space	142
7.1.3	Goal-oriented Interactions in Games Using LLMs	144
7.1.4	Future Work	145
	References	147
	Appendix A Components of LLM-based Analysis	180
A.1	Panel of Results shown to LLM Agent	181
A.2	Agent Conditioning Prompt	181
A.3	Raw Agent Analysis	182
	Appendix B IEEE Copyright Notice	223

LIST OF FIGURES

3.1	Image Quilting Example	23
3.2	Image-to-Image Example	30
4.1	Network Architecture	52
4.3	Qualitative Diversity vs. Texture Synthesis Methods	60
4.4	Qualitative Diversity vs. Single Image Methods (A)	61
4.4	Qualitative Diversity vs. Single Image Methods (B)	62
4.5	Additional diversity examples	64
4.6	Ablation of our Diversity Method	66
4.7	Additional Texture Outputs	67
4.8	Stochastic Failure Case	68
4.9	Periodic Failure Case	68
5.1	Pipeline Overview	74
5.2	Results by Number of Iterations	85
5.3	Visualization of the Graph	86
5.4	Convert Images to Embeddings	87
5.5	Interpolation Process	90
5.6	Prompt Optimization Comparison	100
5.7	Concept Blending via Spatial Interpolation	101
5.8	Full Interpolation Between Endpoints	102
5.9	Illustration of interpolation dimensions	103
5.10	Ablation of interpolation dimensions	104
5.11	A Sample Result Plate Used in the Agent-Based Analysis	106
5.12	Optimized vs. Non-optimized Ablation	107

6.1	Sample Screenshots from Our Game	114
6.2	Opening Dialogue Sequence	115
6.3	Menu-based Interface Example	117
6.4	System Overview Diagram	123
6.5	Overview of the QALM System	124
6.6	Structure of the Prompt	128
6.7	Information Flow Diagram	131
6.8	State Frequency Results	132
6.9	Character Swapping Dialogue Technique	133
6.10	Sample Dialogue Exchanges	135
A.1	Panel of Results for Analysis	181

LIST OF TABLES

4.1	Diversity Network Training/Processing Time	63
4.2	Quantitative Analysis using SIFID	64
4.3	Qualitative Analysis using Image Diversity	65
4.4	Ablation of Similarity Loss	66
4.5	Qualitative Analysis Summary	67
5.1	Projection CLIP Score Comparison	99
5.2	Average LPIPS and CLIP Score Between Linear and Spatial Methods . .	104
5.3	Agent-Based Qualitative Analysis Summary	106
5.4	Ablation of Path Optimization	107

Introduction

The rapid advancement of generative AI has reshaped the landscape of creativity, introducing capabilities that were once the domain of human ingenuity alone. Generative models, powered by deep learning, now enable the synthesis of images, textures, and dialogue with unprecedented realism and diversity. This shift is not only technological; it challenges long-held assumptions about the boundaries of machine intelligence and creative agency. Tasks that have traditionally been the sole domain of human designers, such as designing textures, or crafting dialogue, now have the potential to be part of hybrid workflows where humans use generative tools to realize their design goals. Innovations in the capacity of models to ingest large datasets and recognize complex patterns have rapidly expanded their capacity for generating outputs and their potential for addressing real world challenges.

These innovations have brought new possibilities across domains, from digital media to scientific research, fundamentally altering the way people create, iterate, and interact. This thesis investigates generative models as tools for both artistic expression and problem-solving, exploring their capabilities in synthesizing complex textures, guiding narrative-driven dialogue, and enabling blending through multi-modal embedding spaces. By addressing challenges in controllability, diversity, and coherence, this work aims to extend the state-of-the-art, demonstrating how generative AI not only augments human creativity but also lays the groundwork for a future where the boundaries between human and machine innovation become increasingly blurred.

Generative AI has emerged as a transformative technology, offering tools to synthesize images,

textures, and text with an ease and fidelity that was unthinkable a decade ago. This progress builds on the rapid evolution of machine learning models, which have shifted from rule-based systems to deep learning architectures capable of generating complex and often startlingly realistic outputs. Generative models have found applications across a wide array of domains, from art and entertainment to scientific research and human-computer interaction, enabling new ways to explore creativity and automate tasks. However, the field is far from reaching its full potential. Although generative AI has demonstrated impressive capabilities, its output often lacks the precision, controllability, and interpretability necessary for more demanding real-world applications. Furthermore, the underlying mechanisms of these models are of particular interest. Their ability to capture long-range dependencies, maintain coherence across varying conditions, and balance objectives such as structural fidelity and diversity, are important areas of active research.

In this thesis, we divide our efforts between image and text domains as we present work that merges elements from both the text and image investigations. Generative models represent a class of machine learning methods that enable the creation of new data samples by learning an underlying data distribution. Unlike discriminative models, which are focused on classifying or predicting based on input features, generative models aim to learn the data distribution itself, thereby enabling the generation of synthetic data that preserves key statistical properties of the original dataset. These models have improved significantly in recent years, yet challenges remain. As we investigate text and image synthesis, we focus on two of these challenges.

The first challenge is the synthesis of non-stationary or non-homogeneous textures, where the statistical properties change across the image plane. This is challenging, as assumptions about image statistics cannot be made as with more uniform textures and spatial variations are difficult to model while maintaining structural coherence. Furthermore, we describe a method that learns local and global relationships to synthesize such textures with high diversity.

Then, as an integration of work in both the text and image space, we examine how multi-modal approaches can be used to generate embedding spaces for image manipulation using pre-trained CLIP's [185] shared embedding space. CLIP in conjunction with latent diffusion models has made text guidance describing user intent in natural language the state-of-the-art approach to image generation. Unlike most natural images, textures have a lower level of semantic content. This

makes forming natural language descriptors as a text prompt to be used by systems like ChatGPT challenging. We propose an approach to visually manipulate difficult-to-describe textures by interpolating between user selected inputs with desirable features. By creating an extension of the CLIP embedding space, we can model the features of a dataset of images in three dimensions. Then, through interpolation, we can combine concepts in the source images to create complex and novel textures that would be difficult to describe in natural language. Furthermore, this ability to combine two distinct patterns that are both semantically and visually unrelated is a task that is not amenable to traditional learning methods such as supervised training. Achieving a coherent blend seems to be intractable. We will demonstrate that with our approach, this task becomes possible.

Finally, focusing on the realm of text synthesis, we investigate the challenge of player agency in role-playing games as an important use case for the study of goal-based interactions. Here, the challenge can be summarized in one question: How can we allow a player to invent their own solutions to in-game challenges while adapting the associated dialogue to player actions? Existing design approaches rely heavily on prescribing both puzzle structure and their solutions while also scripting the associated dialogue. Large language models can generate dialogue, but in constrained environments like games, they can quickly lose coherence. We present a method to structure game interactions that allows the assessment free-form player choices while also providing guidance to LLM-based characters, allowing them to remain coherent and aware of the game state.

As generative models are increasingly deployed in multi-modal settings involving images, text, and video, their utility depends critically on their ability to produce coherent outputs. In this context, coherence refers to the preservation of structure and meaning across spatial, temporal, and semantic dimensions in a manner consistent with human perception and expectations. For visual domains such as textures and natural images, this means accurately modelling local and global structures that capture relative orientations, proximity, and variations that align with real-world phenomena. In semantic domains like dialogue generation, coherence requires that content remain contextually appropriate, with the sequence of tokens reflecting consistent style, emotion, and intent. This thesis addresses coherence as a core requirement for generative systems, and proposes methods to improve and maintain it across modalities, thereby advancing the reliability and

applicability of generative models in open-ended, user-driven tasks.

1.1 Context

1.1.1 Texture Generation

We are interested in visual textures; their importance for and their use in digital media and thus how we can create new textures and control their appearance in a practical way. Texture is found in the variations of a surface and is implicit in the materials, natural and synthetic that make up our world. Variations in colour, orientation, shape, and shading and many more visual characteristics contribute to how we humans perceive texture. We are particularly interested in textures whose visual characteristics are non-uniform in spatial extent and where that variation itself is non-uniform such that boundaries between texture and patterns may be gradual or abrupt. An example of such a texture might be a weathered metal surface with rusty patches and peeling paint or a rocky surface with a large fissure dividing geologic layers of differing composition along with some patches of vegetation. Such textures are said to be non-homogeneous. Some non-homogeneous textures have the anisotropic property where internal patterns are expressed in a specific orientation [278] [283].

Such textures are not well suited to classical texture synthesis methods and only recently have learning-based approaches demonstrated progress in this area. Many recent works describe deep learning methods as a continuation along the path of previous research where, in the general case, texture synthesis implies an example-based model where the spatial resolution of an input exemplar is expanded to some increased size. This texture expansion task may be useful in the situation where one has a precise source image for the desired output qualities, but without sufficient resolution. Though in practical workflows where the required spatial resolution is much larger and the visual requirements of the output texture usually cannot be described by an existing exemplar or the required aesthetic details must be refined in response to other workflow inputs or to a specific art direction, such work is of limited utility. Furthermore, controlling the synthesis of highly variable non-homogeneous textures can be very challenging and thus artists must often use manual methods to achieve the desired output and meet creative requirements. Chapter 4. will

focus on methods that address the synthesis and control problem for real-world non-homogeneous textures. We will propose a novel approach for high-quality, high-diversity synthesis of such textures. Then in Chapter 5, we will demonstrate a framework for combining concepts within these textures by constructing a latent manifold for deep interpolation.

1.1.2 Dialogue Generation

In this work, we are interested in enhancing interactive dialogue systems, focusing on structured interactions that enable player agency while retaining coherence and goal-oriented progression in a gaming context. We make progress toward a vision for games where players can make meaningful choices that influence the game’s narrative. Dialogue is a critical aspect in digital media, particularly in games and interactive storytelling, where it serves as the primary means of narrative development and player engagement. Traditional dialogue systems are often constrained by rigid, parser-based designs that limit player freedom, reducing the potential for creative or varied responses. In a gaming context, these systems are frameworks that enable players to interact with characters and storylines, usually through a dialogue selection interface, that have the potential to enhance immersion and narrative depth. Traditional dialogue systems rely on predefined scripts and dialogue trees, limiting these choices and often resulting in repetitive, predictable exchanges. Recent advancements, particularly with Large Language Models (LLMs), have introduced the potential for more realistic, contextually adaptive responses that enhance player agency. However, without structured guidance, these models may lack consistency and alignment with game objectives.

The need for structured guidance relates directly to the role of latent space in LLMs. In the latent space of an LLM, inputs are mapped to high-dimensional representations where relationships between words, phrases, and concepts are encoded. However, this space lacks a built-in awareness of the specific narrative goals or character consistency required in a game. Without structured guidance, the model’s latent space will respond to inputs based on general patterns learned from broad datasets, rather than any tailored structure to progress through game-specific objectives. Our method presented in Chapter 4 introduces a novel approach to enabling goal-based interactions that address the limitations of existing methods.

1.2 Contributions

This thesis presents several key contributions across three core areas: texture synthesis, structured dialogue generation, and concept blending in embedding spaces. Each chapter builds upon prior work to advance methods that enable diverse and high-quality content generation and control. In the following, we outline the main content and contributions of each chapter.

Chapter 4: Diverse non-homogeneous texture synthesis

In chapter 4, we present a model capable of generating diverse variations of complex, non-homogeneous textures, addressing the challenge of capturing long-range, non-local features. This model combines a Generative Adversarial Network (GAN) and a Variational Autoencoder (VAE) within an architecture that enables information sharing through a novel parameter sharing approach. This approach achieves greater diversity by sampling from the VAE’s latent space and incorporating features from the decoder into the GAN’s internal layers. Supporting this method, we also devise a similarity loss term that further encourages diverse output while also improving the overall quality. Using our approach, it is possible to produce diverse results over the entire sample size taken from a single model that can be trained in approximately 15 minutes. We show that our approach obtains superior performance when compared to SOTA texture synthesis methods and single image GAN methods using standard diversity and quality metrics. Based on the work described in Chapter 4., we summarize our contributions as follows:

- Developed a texture synthesis network combining GAN and VAE properties to generate diverse, non-uniform textures from a single exemplar by sampling from the VAE’s latent space and injecting parameters into GAN layers during training.
- Introduced a training regime that allows independent but jointly-trained VAE and GAN networks, which, by sharing parameters between network blocks, can complete training faster than other methods while producing varied outputs from a single input.
- Proposed a sampling similarity loss as a new regularization technique for GANs, enhancing output diversity and overall quality.

The work in chapter 4 led to the following journal publication:

A. Phillips, J. Lang, D. Mould, *Diverse non-homogeneous texture synthesis from a single exemplar*. *Computers & Graphics* 124: 104099 (2024)

Chapter 5: Using CLIP to build latent representations for manifold interpolations

chapter 5, we introduce a method that explores how the embedding space defined by CLIP can be extended to function as an explorable latent representation of any dataset. This enables a new mode of visual interaction with CLIP-space by using source images to compute optimized embeddings. By constructing a spatial graph using the principle components of the embeddings in 3 dimensional space, we create an embedding manifold suitable for interpolation. We assemble embeddings generated from the Describable Textures Dataset (DTD) [37] into the graph, computed from pairwise cosine similarities, which represents the structural relationships among textures in the dataset. Using this structure, we can select nodes (embeddings) within the space to act as end-points for a spatially informed interpolation. Using this method to define meaningful interpolation paths, we can generate novel combinations of the features and concepts present in the input images. This provides a novel control scheme for blending texture concepts to produce interesting combinations that would be difficult to describe using standard prompting methods.

- We show that a constructed latent representation of using CLIP forms an embedding manifold suitable for interpolation tasks.
- Using this method, we texture concepts using embeddings derived from the DTD [38], providing a foundation for spatial interpolation within CLIP-space.
- A spatial interpolation method leveraging CLIP embeddings to create coherent texture blends, showing that spatial relationships between embeddings improve the quality and interpretability of blended outputs compared to traditional linear interpolation.
- A projection technique that maps images to their corresponding CLIP embeddings, allowing for a virtual CLIP-space where concepts can be navigated visually. This approach reduces the dependency on precise textual prompts.

The work in chapter 5 is currently under submission

Chapter 6: Control and constraint of LLM dialogue for structured interactions

In chapter 6, we investigate guidance and control methods for dialogue generation in a game environment using LLMs. It has been shown that unrealistic parser-based dialogue systems limit player agency. LLM characters can enhance agency but lack structure and measurable objectives. We propose a framework for structured interactions that tracks player progress through specific objectives, while also improving character LLM responses. This approach frames interactions as puzzles with states representing goal-based milestones. We use an LLM to analyze dialogue history and enforce state transitions for state awareness and to enable specific actions like tailored LLM prompts and multi-modal content changes. This results in a robust dialogue state tracking system for goal-based interactions. Using our method, a designer can craft transition rules as abstract goals that allow players to invent their own solutions rather than discovering the designer’s intent. We demonstrate this with a hostage scenario game, where the player negotiates with a hostage-taker adversary. The game’s effectiveness is assessed through qualitative gameplay analysis and a quantitative evaluation of our state tracking method. From this work, we present the following contributions:

- A method to enable goal-based interactions with three cooperating elements:
 - A graph-based method that structures interactions using node-based objectives.
 - A question answering LLM system, which measures progress toward objectives and manages game state transitions.
 - A dialogue LLM structure, conditioned on designer-specified character elements, with state-level reinforcement and tuning to maintain focus on the narrative.
- A proof-of-concept game that enables players to freely improvise dialogue and invent novel solutions to objectives while adhering to the narrative structure of the graph.

The work in chapter 6 led to the following journal publication:

A. Phillips, J. Lang and D. Mould, *Goal-Oriented Interactions in Games Using LLMs*.
IEEE Transactions on Games, doi: 10.1109/TG.2024.3515807.

Background

This thesis builds on foundational advances in generative models that have evolved in recent years. Broadly, these generative models aim to estimate a probability distribution with the goal of generating new samples. Given a dataset X , by modelling the underlying data distribution $P(x)$ of samples $x \in X$, we can learn an approximate distribution $P^*(x)$ that allows us to generate new samples by drawing from $P^*(x)$. By modifying the objective function and varying the approach to learning the distribution, multiple techniques have emerged, each with distinct advantages. In this thesis, we use four different generative approaches to synthesize images and text. While we do not contribute to the underlying foundational models' architecture, we use this chapter to describe the technology critical to our work.

Each approach involves a neural network that learns a mapping function to transform the input data into the desired output. This mapping function is constructed as a composition of simpler functions, referred to as the layers of the neural network. Using many layers in combination allows the model to approximate functions that improve upon the capabilities of traditional methods. The mapping function is optimized during the network's training phase. Training can be supervised, where the network's output is compared to a ground-truth dataset, and a loss value is computed to quantify the error. Alternatively, in unsupervised learning, the network is trained using unlabelled data to identify patterns or structure without explicit targets. In both cases, a differentiable loss function is defined to estimate the error in the network's predictions. The gradient of this loss function with respect to the network parameters is then calculated, enabling iterative optimization of these parameters through backpropagation and gradient-based methods.

In this chapter, we provide a brief mathematical background of the underlying models that form the basis of our investigations.

2.1 Generative Adversarial Networks

A Generative Adversarial Network (GAN) [69] is a framework designed to generate new data samples that resemble a given dataset. A GAN consists of two neural networks, a generator G and a discriminator D , which are trained simultaneously through an adversarial process. The generator takes a random noise vector z from a prior distribution $p(z)$ and maps it to the data space to produce a synthetic sample $\hat{x} = G(z)$. The discriminator takes either a real data sample x or a synthetic sample \hat{x} and predicts whether the input is real or generated.

The goal of the generator is to produce samples that are indistinguishable from the real data, while the goal of the discriminator is to correctly identify whether a sample is real or generated. This adversarial process is often framed as a minimax game, where the generator tries to minimize the probability of the discriminator making a correct prediction, while the discriminator tries to maximize this probability. The objective function for this game can be expressed as:

$$\min_G \max_D V(D, G) = \mathbb{E}_{x \sim p_{\text{data}}(x)} [\log D(x)] + \mathbb{E}_{z \sim p(z)} [\log(1 - D(G(z)))] \quad (2.1)$$

Here, $p_{\text{data}}(x)$ represents the distribution of the real data, while $p(z)$ is the prior distribution over the latent space, typically a standard normal distribution. The first term, $\mathbb{E}_{x \sim p_{\text{data}}(x)} [\log D(x)]$, measures the discriminator's ability to correctly identify real samples, and the second term, $\mathbb{E}_{z \sim p(z)} [\log(1 - D(G(z)))]$, measures its ability to correctly identify generated samples. During training, the discriminator D is updated to maximize this objective, while the generator G is updated to minimize it.

Often in practice, this minimax game can be challenging to train due to issues such as vanishing gradients and mode collapse. To address these issues, alternative loss functions and training techniques have been proposed. One common approach is to replace the minimax loss with a non-saturating loss for the generator [69], which can be written as:

$$\min_G -\mathbb{E}_{z \sim p(z)}[\log D(G(z))] \quad (2.2)$$

This formulation helps maintain gradient flow to the generator during training, leading to more stable convergence. Since the introduction of this method, many training methods have been proposed [151, 4, 73, 144, 104, 160, 17] that improve various aspects of GAN training, yet the challenge remains an open area of investigation.

Despite these challenges, the GAN framework has had significant success in the field of machine learning and computer vision. In Chapter 4, we use GANs' ability to synthesize fine details to generate high-quality texture outputs.

2.2 Variational Auto-Encoders

A VAE [120] can be used to map the input data to a probability distribution in latent space. We represent this distribution as a Gaussian distribution with a mean and a diagonal covariance matrix using a VAE encoder. Given some input x , the goal is to learn the underlying probability distribution $p(x)$ from which the input was sampled. This is achieved through the introduction of a latent variable z and modeling the conditional distribution $p(x|z)$. The latent variables capture the hidden structure and features of the data, allowing the model to generate new samples by sampling from the latent space. Since the true posterior distribution $p(z|x)$ is generally intractable, the VAE is designed to approximate the posterior as $q(z|x)$. Specifically, the encoder maps the input data x to the latent space, by parameterizing the approximate posterior $q(z|x)$ by the mean $\mu(x)$ and variance $\sigma^2(x)$.

To learn the generative model, the VAE component maximizes the Evidence Lower Bound (ELBO) [119], which serves as a lower bound on the log-likelihood of the data. This objective indirectly minimizes the difference between the approximate posterior $q(z|x)$ and the true posterior $p(z|x)$. We can express the ELBO as:

$$\text{ELBO} = \mathbb{E}_{q(z|x)}[\log p(x|z)] - \text{KL}(q(z|x)||p(z)) \quad (2.3)$$

where the first term, $\mathbb{E}_{q(z|x)}[\log p(x|z)]$, is the expected reconstruction loss under the approximate posterior $q(z|x)$, which measures how well the model can reconstruct the input data from the latent variable z . The second term, $\text{KL}(q(z|x)||p(z))$, is the Kullback-Leibler (KL) divergence [125] between the approximate posterior $q(z|x)$ and the prior $p(z)$, which acts as a regularization term, encouraging the approximate posterior to stay close to the prior.

Since the encoder/decoder is fully convolutional, we conceptualize the input x as a set of local patches containing spatial information that the encoder maps to a latent space with a Gaussian distribution represented by the mean $\mu(x)$ and variance $\sigma^2(x)$. By aggregating information from multiple patches, the encoder captures both local and global structures in the image. The decoder network generates the output image by reconstructing local image patches from the latent space. These patches are combined, preserving the spatial layout, to generate a complete image. Despite the probabilistic approach’s ability to capture diversity, our experiments show that the VAE’s output quality is generally lower than that of GANs trained on the same task. In our method described in Chapter 4, we combine VAE and GAN models to improve diversity in single image texture synthesis.

2.3 Latent Diffusion Models

Latent Diffusion Models (LDMs) [196] have emerged as a leading image synthesis method, building upon the principles of diffusion processes and variational inference. Unlike traditional generative models, LDMs implement a diffusion process in the latent space to generate high-quality images. The core idea is to learn a diffusion process that gradually transforms simple noise into image data.

Given an input image x , the goal of an LDM is to learn the underlying data distribution $p(x)$. This is achieved by introducing a latent variable z and modelling the conditional distribution $p(x|z)$. The model comprises a forward diffusion process that maps the input data to a latent space and a reverse diffusion process that reconstructs the data from the latent space.

The forward diffusion process adds Gaussian noise to the latent variable z over a series of T timesteps, producing a sequence of latent variables $z_{t=0}^T$. Starting from the initial latent variable

z_0 , the process can be described as:

$$z_t = \sqrt{\alpha_t}z_{t-1} + \sqrt{1 - \alpha_t}\epsilon_t, \quad \epsilon_t \sim \mathcal{N}(0, I) \quad (2.4)$$

Here, α_t is a noise schedule parameter, and ϵ_t is Gaussian noise. As t increases, the latent variable z_t becomes increasingly noisy, eventually approximating a standard normal distribution $p(z_T) \approx \mathcal{N}(0, I)$.

The reverse diffusion process aims to recover the original data from the noisy latent variable z_T . This is modelled by a parameterized neural network θ , which estimates the reverse process by learning the conditional distribution $p_\theta(z_{t-1}|z_t)$. The reverse diffusion process can be expressed as:

$$z_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left(z_t - \frac{1 - \alpha_t}{\sqrt{1 - \alpha_{t-1}}} \epsilon_\theta(z_t, t) \right) + \sqrt{1 - \alpha_t} \epsilon_t \quad (2.5)$$

where $\epsilon_\theta(z_t, t)$ is the predicted noise at timestep t by the neural network. The objective of the model is to minimize the difference between the true noise ϵ_t and the predicted noise $\epsilon_\theta(z_t, t)$, typically using a mean squared error (MSE) loss:

$$L_{\text{diffusion}} = \mathbb{E}_{z_t, \epsilon_t} [|\epsilon_t - \epsilon_\theta(z_t, t)|^2] \quad (2.6)$$

This training objective encourages the model to accurately predict the reverse diffusion process, enabling the generation of realistic images from the latent space.

One of the advantages of LDMs is their ability to handle high-dimensional data efficiently. By operating in a lower-dimensional latent space, the model reduces computational complexity while still capturing important features of the data. Additionally, LDMs can incorporate conditional information, such as class labels or textual descriptions, to guide the image synthesis process, making them suitable for controlled generation. In our work in Chapter 6, we rely on latent diffusion, specifically Stable Diffusion [196], to generate images based on our optimized prompt

method.

2.4 LLMs and Text Transformers

Large Language Models, such as GPT-3 [21], use the self-attention mechanism within transformer architectures to produce contextually relevant and syntactically coherent text. When trained on extensive text corpora, this architecture enables the models to generalize across a wide range of domains without requiring extensive task-specific fine-tuning.

The transformer architecture is composed of an encoder and a decoder, each built from multiple layers of self-attention and feed-forward neural networks. For LLMs like GPT-3, only the decoder stack is used. The self-attention mechanism is defined as:

$$\text{Attention}(Q, K, V) = \text{softmax} \left(\frac{QK^T}{\sqrt{d_k}} \right) V \quad (2.7)$$

where Q (queries), K (keys), and V (values) are projections of the input embeddings, and d_k is the dimensionality of the keys. This mechanism computes relevance weights between each pair of tokens, allowing the model to capture both local and long-range dependencies within the text, thus preserving the contextual relationships necessary for coherent generation.

The training objective of LLMs is to maximize the likelihood of observed data sequences. Given a sequence of tokens $x = (x_1, x_2, \dots, x_T)$, the model learns the probability distribution $p(x)$ by factorizing it as a product of conditional probabilities:

$$p(x) = \prod_{t=1}^T p(x_t | x_1) \quad (2.8)$$

This autoregressive modelling ensures that each token is predicted based on all preceding tokens, resulting in coherent and contextually consistent output. Training involves minimizing the negative log-likelihood of the data:

$$L = -\mathbb{E}_{x \sim p_{\text{data}}(x)} \left[\sum_{t=1}^T \log p(x_t | x_1) \right] \quad (2.9)$$

Transformers also integrate positional encodings [242] to account for the sequential nature of language, as they do not inherently capture token order. These encodings are added to the input embeddings and are defined as:

$$PE_{(pos, 2i)} = \sin\left(\frac{pos}{10000^{2i/d_{\text{model}}}}\right), \quad PE_{(pos, 2i+1)} = \cos\left(\frac{pos}{10000^{2i/d_{\text{model}}}}\right) \quad (2.10)$$

where pos is the token's position, i is the dimension, and d_{model} is the model's dimensionality. These encodings provide the model with the necessary positional information to preserve the order of tokens in a sequence. These modes form the basis of our approach described in Chapter 5.

In modern LLMs such as GPTs, positional encodings are implemented differently. Rather than sinusoidal function, GPTs use learned positional embeddings, where each sequence is associated with a trainable vector that is optimized during training [184, 21]. Formally, the positional encoding for a sequence of length T with embedding dimension d_{model} can be represented as:

$$PE = \begin{bmatrix} \mathbf{p}_0 \\ \mathbf{p}_1 \\ \vdots \\ \mathbf{p}_{T-1} \end{bmatrix}, \quad \text{where } \mathbf{p}_i \in \mathbb{R}^{d_{\text{model}}} \quad (2.11)$$

Here, PE is a matrix of shape $T \times d_{\text{model}}$, and each row \mathbf{p}_i represents a trainable vector corresponding to position i . These embeddings are learned during training and added element-wise to the input token embeddings¹.

¹This description is derived from a huggingface implementation described in detail at: <https://huggingface.co/blog/designing-positional-encoding>

2.4.1 CLIP

Contrastive Language-Image Pre-training (CLIP) [185] is a neural network model that learns to connect images and text by training on a diverse dataset of text-image pairs. CLIP uses the natural correspondence between text and images to create a joint embedding space where semantically related text and images are close to each other in this space. This approach enables zero-shot capabilities, allowing the model to perform various tasks without specific task-related training.

CLIP consists of two encoders: an image encoder and a text encoder. The image encoder is typically a vision transformer (ViT) [51] or a convolutional neural network (CNN) [130], while the text encoder is based on a transformer architecture [242]. The image encoder maps an image I to an embedding vector v_I , and the text encoder maps a text description T to an embedding vector v_T . The goal of training is to ensure that matching image-text pairs have similar embeddings, while non-matching pairs do not.

The training process uses a contrastive learning objective, which maximizes the similarity of the embeddings of matching pairs and minimizes the similarity of non-matching pairs. Given a batch of N image-text pairs, the similarity between an image embedding v_{I_i} and a text embedding v_{T_j} is measured using the cosine similarity:

$$\text{sim}(v_{I_i}, v_{T_j}) = \frac{v_{I_i} \cdot v_{T_j}}{|v_{I_i}| |v_{T_j}|} \quad (2.12)$$

where $|v_{I_i}|$ and $|v_{T_j}|$ represent the Euclidean norms of the embedding vectors.

The contrastive loss function, sometimes called InfoNCE loss, for a single batch can be written as:

$$L = -\frac{1}{N} \sum_{i=1}^N \left[\log \frac{\exp(\text{sim}(v_{I_i}, v_{T_i})/\tau)}{\sum_{j=1}^N \exp(\text{sim}(v_{I_i}, v_{T_j})/\tau)} + \log \frac{\exp(\text{sim}(v_{I_i}, v_{T_i})/\tau)}{\sum_{j=1}^N \exp(\text{sim}(v_{I_j}, v_{T_i})/\tau)} \right] \quad (2.13)$$

where τ is a temperature parameter that controls the sharpness of the distribution, adjusting the

concentration of the embedding distributions and controlling how strictly the model enforces separation between matching and non-matching pairs. The first term in the loss function encourages the image embedding to be close to the correct text embedding, while the second term does the same for the text embedding.

By training on a large, diverse dataset of text-image pairs, CLIP learns a robust and flexible embedding space that captures the semantics of both modalities. This joint embedding space enables zero-shot learning, where the model can perform tasks such as image classification, object detection, and image captioning without any task-specific fine-tuning. For example, given a set of category names as text prompts, CLIP can classify an image by finding the text prompt with the highest similarity to the image embedding. In Chapter 6, we use CLIP to construct an external embedding space suitable for latent-space-like interpolation to generate interesting combinations of texture concepts.

Literature Review

In this chapter, we review the foundational literature relevant to the investigations in this thesis. We emphasize broad coverage of methods and frameworks that motivate the challenges addressed in subsequent chapters. The review considers key developments across texture synthesis, generative model control, multi-modal embedding spaces, and dialogue systems, tracing their progression from early approaches to contemporary methodologies. The chapter tracks overarching trends, such as the evolution from non-parametric texture synthesis to deep generative methods, the emergence of user-guided generative control, and the growing role of multi-modal embeddings in aligning visual and textual data.

By presenting this broad review, the chapter sets the context for the focused discussions in later chapters. For example, the review of texture synthesis provides a foundation for understanding the challenges of non-homogeneous texture synthesis, addressed in Chapter 4. Similarly, the exploration of multi-modal embeddings here outlines their limitations in capturing non-semantic details, which motivates the embedding-driven texture generation explored in Chapter 5. The discussion on generative control methods ties directly to the latent space exploration detailed in Chapter 6, while the discussion of dialogue systems introduces the methods that inform the control and tracking methods presented in Chapter 6. This approach integrates a wide-ranging view of the literature with a clear trajectory toward the specific contributions of this thesis discussed in later chapters.

3.1 Texture Synthesis

In computer graphics, a typical conception of texture synthesis is known as example-based texture synthesis. In this case, one would supply an algorithm with an exemplar containing the desired qualities; then, from this small sample, the algorithm would generate a larger output texture. The output should be a natural variation of the original that appears to have been produced by the same process as the exemplar, maintaining a statistical and perceptual match without undesirable defects such as repetitions, seams, or noise. In this section, we will discuss the relevant classes of synthesis and associated works that have led us to modern learning-based methods. For a more comprehensive coverage of texture synthesis, please refer to the general survey from Akl et al. [3] and the survey of patch-based approaches by Barnes and Zhang [11].

3.1.1 Traditional Example-based Methods

Non-Parametric, Example-based

Pixel-based Non-parametric texture synthesis algorithms have evolved from the simple idea that each pixel depends on its neighbours [95]. The values of the pixels in the neighbourhood of the target pixel, when compared with all possible neighbourhoods in the source, determine the target pixel's value according to the best match (the best match is not always a good match). This idea provides surprisingly good results without the more complex statistical models or density estimations of its predecessors. The body of subsequent contributions follows multiple paths, though many are concerned with speed improvements due to the high computational cost associated with matching neighbourhoods and reducing artifacts and other distortions (sometimes referred to as “garbage”).

Wei and Levoy proposed using tree-structured vector quantization to improve neighbourhood search times. The authors used the tree-structure at multiple scales of a Gaussian pyramid to condition subsequent layers, beginning with the exemplar and a random noise field corresponding to the size of the desired output and equalized to the histogram of the input ¹. This obtains an order

¹This also acts as a control channel to guide the structure of the output which can act as a type of style transfer.

of magnitude improvement in synthesis time and improves output quality [251]. Unfortunately, the smoothing introduced by the Gaussian pyramid tends to blur the appearance of small objects and fine details in the source texture.

Later, Ashikhmin showed that for some natural textures, larger neighbourhoods previously shown to improve quality in multi-resolution synthesis may not be necessary and further, that an exhaustive search of all neighbourhoods is not required. The author proposes a method of pixel growing that encourages copying to form large regions based on a small number of candidates using an L-shaped “causal neighbourhood”. The algorithm tracks the original locations from which synthesized pixels were obtained and uses them to collect new candidates for the next pixel to be synthesized. This results in an irregular patch-like approach with minimal seam artifacts that greatly reduces search time and obviates the need for tree search while maintaining output quality. However, in some cases the significant reduction of candidates can reduce quality as the likelihood of poor matching increases dramatically. The authors attempt to mitigate this effect using random candidates in the case where all candidates for a given location are poor [7]. This method was not dependent on scan-line order (though the paper runs the algorithm in scan-line order) and could be run iteratively across multiple passes to increase similarity to the exemplar, two features that would make further improvements possible.

A common property of the above works is that synthesized pixel values are fixed regardless of quality and a poor match will effect future neighbourhood comparisons, potentially causing artifacts or worse, garbage regions. Each synthesized pixel has some degree of uncertainty related to its compatibility with its neighbours; a high degree of uncertainty is likely to produce a perpetually out-of-place pixel. To satisfy a minimal acceptable level of uncertainty given incomplete information at decision time, when new information becomes available, it is possible to backtrack to the uncertain decision point and reevaluate given the new information. Harrison applied this concept to texture synthesis, allowing the algorithm to continuously resolve a given pixel value at some location until an acceptable threshold of uncertainty was reached. In the same work, Harrison [80] also demonstrated that a random sequence of pixels to resolve was superior to scan-line-based approaches. Early in the synthesis process, the generated pixel neighbourhoods are sparse, leading to long-range information influencing the next pixel’s value. Then, as the process

continues and the image field becomes more densely populated, more local information is included and is therefore implicitly multi-scale due to the random pixel filling. When combined with the backtracking method, the in-progress generation can be refined from coarse to fine as values are re-chosen based on new information. This method produces a higher quality output, but runtime is slower than that of Ashikhmin [7]. The method is also not suitable for non-homogeneous textures.

Patch-based

The pixel-by-pixel methods described thus far attempt to find a three-way balance between computational complexity, number of searches, and output quality. One might imagine that a pixel is a “patch” within the image of size one and extend pixel-based methods by increasing the patch size from a single pixel to a larger region. In this case, an algorithm should also search the exemplar for candidate patches that match the current neighbourhood, but the search now must consider patch border compatibility to minimize seams and disjoint features. The benefit of searching for patches and not pixels is the resulting reduction in computational complexity. When synthesizing pixels, in many cases the values are predetermined especially in texel regions, but the values must be determined through the naive and expensive search process. Efros and Freeman pointed to this fact as their primary motivation for their foundational work in patch-based texture synthesis the eponymous “Image Quilting” named after their proposed method [52].

Assembling an image in a patch-wise manner requires that algorithms match features between adjacent patch placements and smooth the seams between them. Efros and Freeman’s image quilting approach seeds the output with a random patch and proceeds greedily in scan-line order, overlapping patch borders to define an error surface that can be used to minimize the final border path between patches in order to smooth edges. This minimum-cut approach usually produces cleaner results than the feathering used by Liang et al. [142], but it is difficult to obtain visually pleasing blending on a consistent basis due to the coarseness of the minimization across varying degrees of error while targeting the overall minimal cost. Candidate patches are selected within some threshold that constrains the total error over the overlap. This method successfully reduced the computational cost of Efros and Leung’s pixel-based approach while also improving robustness to generating garbage regions. However, both pixel and patch-based methods are challenged by

non-stationary exemplars.

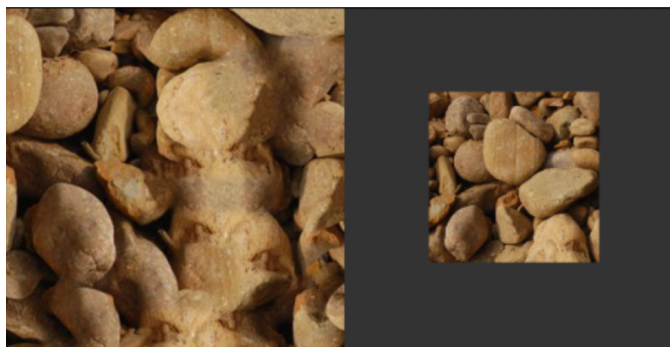


Figure 3.1: **Image Quilting Example.** Output via Image Quilting using patch size of 20 and border overlap of size 10. Source image is on the right.

The work of Efros and Freeman inspired a large follow-on body of work that explores improvement in patch boundary blending. One approach by Kwatra et al. [126] abandons the grid-based placement of patches in favour of an optimal irregular patch (derived from rectangular candidates). The shape of the patch is determined via graph cut, which finds the optimal boundary within the overlap region by formulating the problem as a graph where nodes represent pixels and edges represent similarity or cost. Since the method has an $O(n^2)$ complexity, the authors developed a Fast Fourier Transform acceleration to significantly reduce the compute time.

3.1.2 Deep Generative Methods

Parametric texture synthesis approaches seek to find a model that describes the underlying statistical structure of visual textures, which can then be used to generate samples that, in the limit, are perceptually equivalent to the estimated statistics of an example texture. The resulting model is advantageous compared to parametric approaches as it facilitates the generation of new texture samples with equivalent statistical properties as opposed to remixing an exemplar². This family of synthesis methods is inspired, at least initially, by the idea that texture images with equivalent first and second-order statistics would be visually, perceptually indistinguishable for

²This is a reductive description of non-parametric approaches that discounts the high quality results obtained from a simple formulation

humans. This version of the conjecture was later disproven [181, 105] and the search for more complex descriptors continued along this line of inquiry. Traditional methods tackle this challenge by designing descriptors to represent image features that parameterize a statistical model. The features target multiple aspects of image structure. For example, in the case of Portella and Simoncelli, one of the most successful parametric models, they compute four high-level statistical classes, each responsible for multiple statistics such as multi-scale cross-correlation of subband magnitudes to capture edges and corners and numerous others amounting to 710 handcrafted parameters. New images can be sampled from a parameter space established from example textures. Their work produces good results for many examples, and while the authors emphasize the universality of their method, there are many cases where their approach fails³. Such traditional approaches have the immense challenge of manually designing descriptors capable of representing the diversity of natural and other classes of textures; a challenge which has become, like computer vision focused handcrafted feature approaches, become suboptimal following the rise of deep neural networks.

Deep neural networks also learn parametric models. However, in the case of convolutional neural networks, feature detectors are learned hierarchically directly from image pixels during training. These detectors are able to learn progressively more complex, spatially invariant features as the input passes through deeper layers of the network, each layer building on the work of the previous layers. This is said to mimic the function of the visual cortex in mammals [92, 261] and has been shown to outperform handcrafted features designed for tasks across computer vision and graphics. In 2015 Gatys et al. took advantage of this observation in their work “Texture Synthesis Using Convolutional Neural Networks”. As one of the first applications of CNNs to texture synthesis, the authors largely follow the structure of Portella and Simoncelli, with the crucial substitution of learned feature detectors in place of handcrafted statistics.

Working on the definition that textures are stationary and independent of spatial information, Gatys et al. insightfully employ the Gram matrix to compute correlations between feature responses as:

³This includes non-homogeneous textures which some researchers do not consider to be true textures

$$G_{ij}^l = \sum_k F_{ij}^l F_{jk}^l \quad (3.1)$$

Where G computes the pairwise inner product of feature maps for each layer l and when computed across all layers, provides a stationary description of a texture exemplar as the spatial information is discarded. The result of this computation serves as a second-order summary statistic that encodes correlations between features across the spatial extent of an image. Here F_{ij}^l and F_{jk}^l are activations of feature maps i and j in layer l .

The authors use the off-the-shelf VGG architecture pre-trained on the ImageNet dataset as a feature extractor for texture image examples to be synthesized. At the time of generation, the network begins with an image initialized to white noise and is passed through the CNN model where the SSE loss is evaluated between the gram matrices of extracted features from the target and the image being synthesized, which is then back-propagated through the network to iteratively minimize the distance between the in-process image and the target via gradient descent. Their results, as shown in the paper, demonstrate a significant improvement over Portella and Simoncelli and showed the advantage of CNN-based features for this task. Additional testing shows good performance on non-stationary texture examples with moderately sized features, though due to the relatively small receptive field imparted by the VGG network, the scale of features within the target is limited. The method struggles with highly periodic texture examples. The approach is also highly data-driven, dependent on the also expensive pre-training of the VGG network on ImageNet. The authors note that their method is substantially more computationally intensive than traditional methods due to the optimization process. This can lead to inference times required to generate a new output measured in minutes. Subsequent work attempts to alleviate some of these problems. Snelgrove [220] adds a multi-scale Gaussian pyramid to achieve high-resolution textures that is able to capture very fine details but still suffers from high computational demand. Risser et al. [194] showed that many of these issues with the Gatys et al.'s method arise due to instabilities, which, as the authors showed, are due to the fact that textures can have different variances but the same Gram matrix. For example, textures where brightness and contrast vary across the image are particularly susceptible to this issue. To address this, Risser et al. [194] developed a robust method

using their novel histogram losses combined with Gram matrix statistics. Their histogram losses uses the standard histogram matching technique, but instead of operating on image pixels the matching is computed on activations for each convolutional feature map at each layer and matched to the histogram for the activations of the input texture to obtain a remapped set of activations. The Frobenius norm distance is used between the source and target activations and that loss is back propagated using gradient descent.

Other works [240, 103], seek to improve the efficiency of Gatys et al. by offloading the optimization to a training stage, leaving a feed-forward convolutional network to perform inference in real time instead of solving an optimization problem iteratively for each texture. In general, the output quality and diversity falls short of the optimization-based methods since the feed-forward network is only an approximation of the optimization process. Several methods have narrowed the quality gap [141, 238, 239, 141, 213] with the optimization-based approach of Gatys et al., obtaining nearly identical visual quality. Shi et al. devised a method to “simulate” the optimization process via FF network using layer-wise gradients on (Gatys et al. ’s) texture loss to predict the next modification to the input. The authors demonstrate improved performance and diversity compared to previous methods. Notably, both the optimization and feed-forward approaches struggle with periodic textures when using the Gram matrix descriptor as the spatial regularity that drives the periodicity is discarded.

GAN based Texture Synthesis

With the introduction of Generative Adversarial Networks (GANs) [69, 4] a new line of inquiry opened with the potential to improve texture synthesis further. One criticism of deep, feed-forward approaches that use convolutional features is that they often rely on a network trained on large-scale datasets such as ImageNet which may not contain higher-level feature maps that are compatible with various texture classes that one may be interested in. For example, many papers make use of satellite imagery for texture examples, but generic datasets such as ImageNet do not contain this class of imagery and thus structural (e.g. street grids) and other important features are absent and are likely not to be represented in the output of such a texture. Generally, GAN-based models learn image features directly from the textures they are trained on and thus

are much better at representing the target image statistics in the synthesized output.

One of the first applications of GAN to texture synthesis was that of Markovian GAN (MGAN) by Li and Wand [136], who used adversarial training to learn a mapping between Markovian patches and similar patches on a complex manifold. They showed that GANs can generalize to produce different versions of the same content in a Markovian setting. Another early model whose approach relies on the learned statistics of the target was the SpatialGAN (or SGAN) by Jetchev and Bergmann [98]. In this work the authors trained a GAN to transform a “spatial tensor” noise array into the output RGB image which is a departure from the standard single vector approach and allows sampling of a larger noise tensor and thus makes an arbitrary output size possible. The authors effectively take advantage of a Markovian assumption of local statistics where they treat the generation of spatially distinct image locations as independent which results in plausible output at a larger scales as the local output resembles the original image. In addition to the arbitrary output size, their images are able to represent structural information of the exemplar with much more accuracy versus their comparators. As with feed-forward methods (Note: GAN inference is also feed-forward) the generation of new images can be done in real-time. The authors trained the SGAN on multiple input textures (on the order of five images) and found that the output contains properties of each image in a reasonably well interpolated, and blended way, but without structure or plausibility depending on the subject. That is to say that the output is a mixup of all of the input images. This is the first indication that multiple exemplars might be of benefit for this task. The authors introduce a follow-up work known as the Periodic Spatial GAN (PSGAN) [16] with two major contributions. The first deals with the mixup behaviour of SGAN through the introduction of “global and period dimensions”. Where SGAN randomly samples from a latent vector Z to assemble the final image from “local dimensions”, PSGAN extends Z to represent three layers: locally independent, spatially global, and periodic. These layers, or dimensions can be sampled according to the authors’ design to impart an effect on the whole image and allows for the selection of the type of structure in the output since each subvector through Z spans the three dimensions and corresponds to a specific and localized area of the target texture. Therefore, independent spatial regions can include textures sampled from the manifold learned by the network and can be smoothly interpolated in the output when variance within the global Z spaces is gradual. The periodic part of Z embeds channel-wise spatial functions that parameterize

positional and periodic features for the output and are learned via MLP from the example texture. This has the effect of transforming the spatial coordinate system of the output indexed by μ and λ , aligning and blending the regional patches in the correct orientation. When tested on non-homogeneous textures, PSGAN is unable to produce a coherent image when expanding the exemplar as might be expected, though when blending samples from the manifold, textures with non-stationarity may emerge, but in unpredictable and often unpleasing arrangements.

Of the deep texture synthesis methods presented thus far, none are capable of capturing and reproducing the non-stationary features common in many natural images. Long-range, global features and spatially varying non-homogeneous features have been notoriously difficult to handle without some form of external guidance mechanism. The question of how to deal with non-stationary exemplars was answered by Zhou et al. [285] who provided a training approach whereby the source texture block ($k \times k$) is taken as a crop of a larger source texture, then during adversarial training the output of the generator is compared with a target block (of size $2k$) using a learned adversarial loss combined with L_1 and a style loss based on a pre-trained VGG-19 network. Using this method, the network synthesizes new content as the spatial size of the output is doubled. Since the network is trained using sub-blocks or patches from the input, output diversity is constrained, though the authors propose simply selecting alternative source blocks from the larger input as a different starting point for the synthesis as a mitigation. Given the non-stationarity of the input, this approach does facilitate additional diversity. In the case of more stationary or stochastic exemplars randomly reshuffling tiles on the larger source provides some additional diversity. Overlooking the typical expensive training cost, failure cases and artifacts, many consider this work to be the state-of-the-art for deep, exemplar-based synthesis with respect to the quality of the output for non-stationary exemplars. We note the significant caveat, that the user requires a source texture of size larger than the expected output to train the network.

The challenge of diversity is shared across the texture synthesis landscape. In almost all cases of example-based texture synthesis, regardless of category the single exemplar (or some small sample $n < 10$) is the sole source of information to be used to build a model or from which to sample directly. For our work diversity is critical in order to enable control of the output and perhaps more importantly to enable creativity while still producing a plausible output. In general

and for most applications GANs require a large scale dataset to produce high-quality results, but for texture synthesis operating on a single image might be considered part of the definition itself. One work that shares some of our goals and makes use of additional data to achieve the required diversity is TileGAN by Frühstück et al. [59]. TileGAN like the work of Zhou et al. above focuses on non-stationary textures, but with the addition of a formal “artistic” control scheme and arbitrary spatial output size. The TileGAN method uses “latent tiles”, which define the spatial layout of the output based on a randomly generated or user specified guidance map of lower resolution. The latent tiles are sampled from a dataset distribution to approximate the guidance map based on a $L2$ norm compatibility search between tiles. This is followed by an iterative refinement stage to ensure the minimal dissimilarity between the 4-connected neighbourhood tiles. Once the latent field is set the final image is generated. The authors train their model using datasets ranging in size between 18 and 65,000 tiles. With this scale of data and this method, diversity can still be an issue as optimal patches may be repeated in the latent field leading to repeating features.

3.2 Controllability of Texture Synthesis

Using GANs for texture synthesis has led to impressive results that surpass what was possible using previous methods. Improved image quality and real-time image generation (after the network is trained) are important advances, but a third attribute of control is a critically important attribute that is required for more widespread and practical use of GANs, especially for the task of texture synthesis. In this section, we will identify the various modes of control that have emerged in the literature and group major works according to their control mechanism.

There is a broad spectrum of control methods described in the literature, from single exemplar to elaborate interactive colour painting done using physical media. The mode and technique of control is often tied to a particular use-case or it may be needed to overcome challenges related to the type of texture being targeted. While some methods are baked-in to the approach at a fundamental level (e.g. the noise tensor of PSGAN), here we are interested in control schemes that are user-facing and do not require advanced technical training to understand and manipulate the control input(s).

3.2.1 Conditional Image Synthesis

A broad class of output control involves conditioning the output generation on an input prior that defines the visual and structural qualities of the desired output. The specific type of conditioning used as input varies greatly. Class conditional methods restrict generation to individual categories, text-to-image methods [70, 188, 189, 269] use textual inputs to control the content and appearance of synthesized images and recently GAN-based image-to-image translation [90, 95, 289, 290] uses images of various forms to affect the characteristics of the output. With “pix2pix” [95] by Isola et al. often being used as a baseline for comparison given its versatility.

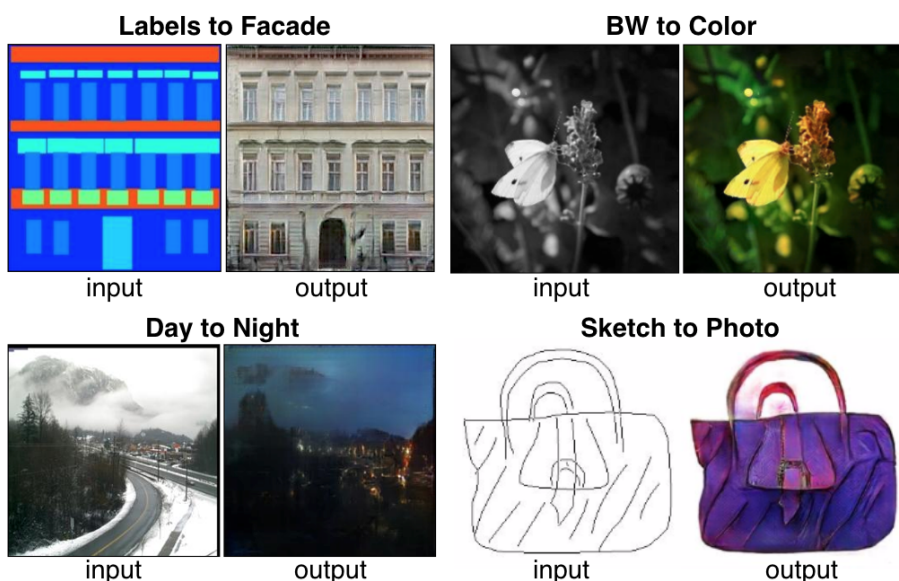


Figure 3.2: **Image-to-Image Example.** Various modes of image-to-image translation by Isola et al. [95] (Image used with permission from author)

Segmentation Map

A segmentation map is a pseudo-colour representation of an image where each colour region represents pixels with some shared characteristics. Usually, these regions correspond to a class of pixels with some semantic meaning. For example, a photograph taken of an outdoor vista might contain classes for “sky”, “mountain”, “grass”, etc. and the segmentation map of that image would map all the “grass pixels” to a single colour in one or more regions. When used as an input

to an image synthesis algorithm, the segmentation map instructs the algorithm to generate pixels that have the characteristics that correspond to the class of each target region.

In *Image Analogies* by Hertzmann et al. [82] the authors introduce the idea of texture-by-numbers which allows complex scenes containing multiple textures to be generated based on a simple user created representation. The segmentation map may be edited interactively by the user in a “painting-like” manner. A more recent work by Park et al. [172] use a GAN-based method to translate the input segmentation map into photographs while also significantly improving the generation time. Their segmentation painting tool operates in real-time with a simple brush and palette interface. The results overcome the significant challenge of synthesizing the region borders in a plausible way. For example, more naive approaches could not include regional variations that are a consequence of predictable physical phenomena. Ripples of water at the boulder and water boundary or foamy surf at the interface between sand and sea. However, due to the large-scale datasets (e.g. 41,000 landscape images) used by Park et al., their model draws from a large distribution of such interface examples and is able to incorporate this added realism at generation time.

Texture synthesis is used loosely in describing these works and this approach, as the outputs here are natural images and the “textures” are, with the addition of quite a lot of intelligence, effectively sampled from the model as a kind of database. Could a segmentation map be used as a guidance channel for the synthesis of true textures? In the case of non-stationary examples such a map could provide a powerful means of artistic control as the “painting” interface described is a natural mechanism for our target user. Unlike natural images, the low-level textures they contain and their respective features are not easily classified into semantically meaningful regions. To overcome the lack of semantic meaning for weathered textures, Chen et al. [34] substitute the segmentation map with an “age map” [14] for their image-to-image translation model. The age map defines the “weatheredness” or age of each pixel and can be used to control spatially long-range and temporary varying features. This leads to the question of whether such attributes appear as latent factors in the manifold of generative models.

Sketch-to-Image

From an artistic perspective, sketching is a less detailed, preliminary step towards a more detailed drawing or painting that may also stand alone as a completed work. This definition serves a similar purpose in the context of the sketch-to-image task, where the sketch is a sparse representation of object boundaries and coarse features. The sketch is also preliminary, but here a model is trained to fill in the details. In the literature a sketch can take various forms, from a hand drawn, freeform “doodle” to a generated edge-map processed from a photograph.

Under this regime, the output image is conditioned on the sketch input which is consistent with an image-to-image translation configuration with a pix2pix baseline. One example is SketchyGAN by Chen and Hays [32] who generate realistic images from crude hand-drawn sketches. Their model can generate images from a diverse set of fifty object classes based on their assembled dataset of 75,000 human sketches paired with photographs which is augmented by over two million Flickr images paired with generated edge maps. Given the significant differences between sketches and edge-maps, the early training is weighted heavily towards edge-map training and gradually moves toward sketch-based input pairs. The authors acknowledge that the results were often not photorealistic. Output images captured the structure of the sketch, but the texture contents, pixel colours, as well as contours and boundary transitions were often not realistic. This is partially attributed to the class diversity of the dataset.

In some cases, authors have used additional control parameters to constrain the output. For example, Sangkloy et al. [200], using their conditional GAN Skribbler, permit the sketch to be augmented with arbitrary colour strokes in user selected regions of the sketch. Compared to the groundtruth output without colour hints, colour augmented sketches colourize the output in a plausible way and maintain consistency between semantic regions. Interestingly underlying textures are consistent regardless of the indicated colour or the presence of a colour stroke on the input. This leads to a lack of realism manifesting as a translucent colour-by-numbers effect.

Instead of using a colour parameter to guide the model’s synthesis for sketched regions, Xian et al. [257] developed TextureGAN that combines an input sketch with user-provided texture patches. Their goal was to bridge the gap between the semantic knowledge of the network and the

user’s preference for regional texture characteristics. To enable this function, the user may “drag” one or two texture swatches onto target regions within the provided sketch. The authors apply a local texture loss that facilitates the synthesis of textures that were not present in the training data. Segmentation masks and foreground-on-white-background images are used during training to ensure that the texture synthesis respects the semantic boundaries of the sketch.

Another parameter augmentation method combines hand-drawn sketches with a style transfer exemplar. In their Sketch-to-Art work, Liu et al. [145] use a style and content loss in their feature map transfer layer that extracts style information from feature maps. By providing their network with a sketch (edge-map) and style exemplar as inputs, the system creates a rendering that matches the structural information provided by the sketch and the colour palette of the example style image is also clearly adopted. The model appears to lack semantic knowledge regarding the content of the sparse representations. The distinction between “sky” and “ground” is consistent, though textures in the example style that represent semantic objects as understood by humans are not mapped as expected to the sketched representations of similar objects.

3.2.2 Style Transfer

The style of an image is a combination of aesthetic characteristics such as colour palette, brush strokes, or more abstract concepts like cartoon versus caricature and many other factors that are distinct from the content of the image. Sometimes the style of the image is synonymous with the artistic style one might associate with a specific artist or the artistic medium such as water colour paintings. More broadly, the style could relate to the lighting present in an image by time of day or in the case of Jang et al. the temporal style of human motion [97]. In each case it is easy to imagine the separation between these qualities and the content - the objects present in the image. Style transfer is the task of applying the style from an example image to the content of another. This work has seen the most success in the artistic domain [147], but various photorealistic transfers are also possible, but this application is usually accomplished using image-to-image translation [131].

The general style transfer workflow as commonly presented in the literature is to obtain a pre-existing style exemplar as a single image to be applied to the user’s target content image via the algorithm. In many cases, impressive results are possible, but the opportunity for granular

user control is limited. Since the algorithmic definition of style is a rigid statistical construction, the aesthetic and perceptual control is contained in the gestalt of the exemplar. Some authors have mitigated the lack of control in various ways: Manually specifying regions where content should be preserved [53], controlling stroke size [102], and controlling glyph deformation via shape-matching [262].

3.2.3 Latent Space Navigation

A GAN generator maps an input z sampled a random distribution to a particular output image, where z is an n -dimensional vector (usually $100 \geq n \leq 512$). As values in z change, the generated image is also changed. Each dimension or factor of z corresponds to some change in the features of the output. When a dimension affects multiple features, the changes it produces along a given axis are difficult interpret and the latent space is said to be entangled. It is possible that dimensions of z each manipulate a single feature along their axes such that a change in that dimension has a singular, interpretable response (e.g. changing hair colour from dark to light) in the output. In this case the network has learned a disentangled and more compact representation of the data. In general GANs do not learn disentangled representations automatically and while recent improvements in dataset structure and training regimes have demonstrated structural improvements to the latent space, the extent of the disentanglement for a given dataset/model is difficult to quantify [281]. Furthermore, even with a disentangled model in hand interpreting the meaningful control directions is not straightforward [78]. Work is on-going to understand the structure of latent space and develop techniques to discover, interpret and control it [15, 77, 78, 237, 244, 273]. Here we discuss the utility of latent codes for controlling image output both as a control scheme and specifically for texture synthesis.

The canonical image class used for latent space disentanglement and control is that of human faces. [115, 111, 211]. Facial images are particularly amenable to manipulation via latent code due to the spatial and scalar regularity of a relatively small set of semantic attributes and the availability of large, high-quality, labelled datasets. For faces at least, this seems to correspond to local isometry on the embedding where the magnitude of change in the embedding space is correlated to the magnitude of change in the input space [49]. Authors in this domain exploit the

mechanism of disentangled latent space as represented by a set of control tasks which we will briefly outline.

Since each latent dimension maps to a single semantic feature, in the ideal case, the features of a single image remain fixed while the target feature is changed. For example, remaining in the face synthesis domain, attributes such as hair colour, age, pose, and even presence of glasses may be changed while the face identity remains intact [218]. This demonstrates the increased generalization capacity of disentangled models that allows attributes to be customized beyond the combinations that are present in the source dataset.

The nature of the disentangled manifold affords a variety of control schemes. Dang et al. introduce user control via slider controls, each connected to a dimension of the latent code discovered to have a specific effect [42]. In their GANSlider work they introduce a “filmstrip” slider design that presents multiple preview images along the slider controlled dimension. Their work is an extensive study of the UI design space for interactive models that provides insights regarding UI parameters such as how the number of slider dimensions increases task difficulty and the importance of performant visual feedback. Their results validate the slider control as a viable control method, but their conclusions are most relevant to actions of non-expert (non-artist) users.

Separately from the latent code embeddings of GANs, latent spaces that are implicitly disentangled can be created using dimensionality reduction techniques. Zsolnai-Fehér et al. [292] construct a non-linear 2D latent space that describes a material model used for 3D rendering with the goal of navigating the output space of different materials for user selection. The authors augment their 2D latent space with a heat-map overlay based on learned user preferences. The heat-map visualizes regions in the 2D latent space with the expected scores for materials. This allows users to first visually explore different materials, but also quickly locate regions that are likely to be favourable, before fine-tuning their preference based on similar materials in the highlighted region. For each point in the latent-space the candidate material rendering is synthesized, in real-time allowing for a quick assessment rather than waiting for a physically-based renderer to converge.

Beyond generating novel images through the exploration of image parameters in the latent space, it is now possible to apply latent code axis manipulations to real images. Given a pre-trained GAN

model and some target real image, GAN inversion [93, 106, 173, 287] seeks to recover the latent code that corresponds to the target image from the existing latent space. However, recovering the latent code is not an easy problem and requires either a learning-based method to map the image into a latent space or a optimization-based approach to solve a minimization objective between the real image and the generator’s output given some latent code z . More recently GAN based approaches have been used to locate the optimal latent code [192],[250] in an attempt to maintain the quality of optimization methods while reducing the computational burden.

3.3 CLIP

Multi-model embedding spaces for text and images define a semantic mapping, in a single mathematical space where their meanings align. This enables geometric reasoning between embeddings where the “distance” is a reflection of their relationship (e.g. shorter distances represents greater semantic similarity). This cross-modal understanding can be useful in classification and search tasks, for example. Models such as Contrastive Language-Image Pretraining (CLIP) [185] and ALIGN (A Large-scale Image and Noisy-text embedding) [100] use contrastive learning to train dual encoders to maximize similarity between paired inputs while also minimizing similarity for unrelated pairs. Training these models generally requires very large datasets of image-text pairs with continuously increasing size[169, 227]. LAION-5B [203] contains over 5 billion pairs for improved generalization across semantically diverse subject matter.

The strength of these aligned embedding spaces has led to state-of-the-art models from zero-shot classification [185, 280, 270], image retrieval [33, 8], and text-to-image generation [187, 186]. This has led to significant change, particularly the improvement in how easily high-quality images can be generated using a text interface to accurately generate output faithful to the user’s intent. However, by optimizing embeddings for high-level semantic alignment can lead to limitations in some domains. This kind of broad-semantic training may not adequately capture the features and relationships present in textures, especially complex patterns or non-homogeneous texture. We find these types of images to be difficult to describe, motivating the work in Chapter 6.

3.3.1 Application to Generative Models

Text-to-image synthesis is perhaps the most prominent application of multi-modal embedding spaces, having demonstrated the greatest utility when used in conjunction with diffusion models. While denoising diffusion probabilistic models were initially trained on small datasets for unconditional or simple class-conditional generation [47, 166]. The need for improved classifier guidance led to improved conditioning methods and eventually more robust conditioning using CLIP encodings. Using CLIPs text encoder to better align the generation with text input led to significantly improved results [198, 9]. Later latent diffusion models [196] made it feasible to generate higher resolution outputs by operating on a compressed latent representation of the training data rather than directly in pixel space. This, combined with CLIP integration, represented a significant improvement in output quality and relevance, which has made text-based guidance as prompts the standard mode of interaction. This is evidenced most notably by the rise of services such as DALL-E [19], Midjourney⁴, and Stable Diffusion [196].

Still within the realm of image synthesis multi-modal embeddings have been used for related control tasks like style transfer and image editing. Patashnik et al. [174] used CLIP to add text control to StyleGAN [111]. Their method uses a latent mapping to align the CLIP embedding space with the StyleGAN latent space, enabling text-driven manipulation of generated images. Frans et al. [56] proposed a method to use text prompts to produce stroke-based drawings using Bézier curves. Their method uses CLIP as a metric to maximize the similarity between prompt and generated drawings without requiring any training. Multi-modal approaches have also been adapted for domain-specific tasks, including scientific visualization and medical imaging. In these contexts, embeddings are used to condition outputs on specialized textual descriptions, facilitating the generation of illustrative or diagnostic images [277, 67]. For example, models conditioned on domain-specific embeddings can generate synthetic medical scans that align with specific clinical features, addressing data scarcity issues in machine learning for healthcare [277].

Despite these advances, applying multi-modal embeddings to domains like texture synthesis remains challenging. Most existing techniques prioritize high-level semantic alignment, which limits their capacity to capture difficult-to-describe structural details.

⁴<https://www.midjourney.com/>

3.3.2 Texture Synthesis Using CLIP-connected models

Recent work has explored the application of CLIP-connected models to texture synthesis, using multi-modal embeddings to guide generation and control tasks. Song et al. [223] introduce a framework for generating artistic texture and style transfers using CLIP to minimize semantic differences between images and text prompts. We note that their method generates patterns based on semantic objects in contrast to traditional textures and materials. Guerrero-Viu [72] et al. apply the more common approach of manipulating CLIP embeddings to condition diffusion-based generation. Their method enables text-based adjustments to semantic attributes of textures to affect attributes like weathering and roughness. Their method attempts to preserve the properties of the input image during the generation process. Chen et al. [29] use text prompts to apply textures to 3D geometry by computing similarity between the text and image embeddings using CLIP and iteratively during the synthesis process to measure semantic accuracy to maintain coherence.

3.3.3 Multi-Modal Models Beyond CLIP

While CLIP has emerged as the standard multi-modal embedding approach, other models such as ALIGN [100], BLIP [139], and Florence [267, 258] use similar but complementary methods. ALIGN improved on early CLIP with larger, higher quality training data [100]. ALIGN uses EfficientNet [231] for the image encoder versus ViT [51] in CLIP. While CLIP and ALIGN are focused on the alignment task in a shared embedding space, BLIP [139] adds the a generation component that is designed specifically for vision-language tasks (e.g. image captioning). Finally, SigLIP [268] introduces a novel approach that uses a pairwise sigmoid loss instead of the softmax normalization used in standard contrastive learning, allowing for independent computation of image-text pair similarities without requiring a global comparison across the batch. While the pairwise sigmoid loss facilitates the use of larger batch sizes, the benefits saturate at a batch size of approximately 32,000.

3.3.4 Metrics

Evaluating the quality of outputs from multi-modal generative models presents significant challenges, particularly for text-conditioned tasks. Traditional metrics such as Fréchet Inception Distance (FID) [83] and Learned Perceptual Image Patch Similarity (LPIPS) [271] focus on perceptual similarity and diversity but do not adequately measure alignment between text prompts and generated images. To address this, CLIP-based similarity metrics have been introduced, using cosine similarity between text prompt embeddings and images to quantify semantic alignment [185, 110]. The work of Rombach et al. [196] integrates perceptual and CLIP-based metrics to evaluate latent diffusion models, demonstrating how these combined approaches can assess both semantic alignment and structural fidelity.

3.4 Dialogue Control

3.4.1 Branching Tree and Rule-based Dialogue Systems

Historically, branching/decision tree and rule-based dialogue systems have been the basis of interactive narratives within video games, directing conversational flow through predefined paths [199]. These systems enable designers to create structured dialogue trees with multiple, but limited story paths, with the goal of granting players a sense of agency by allowing them to select responses that influence the storyline. However, this is time-consuming and expensive [26, 252]. Such systems are often used in role-playing games, where dialogue choices impact relationships and outcomes within the game world, though players' ability to impact outcomes is minimal. [107].

In a branching dialogue system, each player choice leads to a specific node within a dialogue tree, establishing a clear but fixed path [199]. Although effective in conveying narrative details and predefined choices, such structures lack the flexibility to adapt meaningfully to player choices or dynamically changing contexts, which can limit player agency as the system can only respond to predefined inputs, reducing immersion [226].

Traditional rule-based systems enhance branching trees by applying rules or conditions to dialogue nodes, allowing for more contextually relevant interactions. However, this approach still requires

manual scripting of each potential interaction, which can also become resource intensive in larger games [6]. Such systems also struggle to maintain coherence in more dynamic or open-ended gameplay scenarios, where player actions are less predictable and may not fit neatly within a structured dialogue tree [235].

Hybrid approaches, combining branching dialogue trees with dialogue generation-based responses, represent a promising direction in game dialogue systems. For example, designers may use branching trees to structure major narrative beats, while employing LLMs to generate context-sensitive filler dialogue, enhancing immersion without sacrificing story control [266]. Weir et al. presented a method to generate branching dialogue based on player choices, trained on a dataset of dialogues from an existing game [252].

3.4.2 Learning-based Dialogue Control

With the rise machine learning techniques for text synthesis, dialogue systems in games are shifting from predefined branching trees to more adaptable architectures. These models generate responses based on learned patterns from large corpora, enabling a greater variety of responses that can adapt to unforeseen player inputs [21]. Unlike scripted branching trees, these model generate responses dynamically, providing the foundation for a more varied and user-controlled game path. This has led to a new class of interactive systems that balance narrative coherence with the flexibility to respond to complex player behaviour [60, 230] and game management [286]. This approach can afford more emergent interactions that may enhance player participation [175].

Much of the recent work in dialogue systems began in the context of “chatbots” to serve either as task-oriented agents solving specific needs, like booking restaurants, or as open-domain systems engaging in unrestricted conversations [195, 165]. Task-oriented systems traditionally use a pipeline with modules such as natural language understanding and reinforcement learning components, though end-to-end models have begun to improve performance [165]. Open-domain chatbot-style dialogue systems include generative, retrieval-based, and ensemble approaches, each with strengths in response flexibility, coherence, and context relevance depending on the task [138, 137]. Generative models used in this context are often based on sequence-to-sequence architectures to dynamically produce responses [243, 206]. Retrieval-based models, by contrast,

select suitable responses from a predefined set, which helps ensure coherence by relying on existing conversational data [208, 137]. Ensemble methods combine these approaches to balance creativity and coherence, enhancing dialogue systems' adaptability to varied user inputs and improving the quality of interactions [208, 275].

3.4.3 Prompting Techniques in Dialogue Generation

Prompt Engineering for LLMs

When working with LLM, prompt engineering has emerged as a primary tool for controlling model responses and guiding output in desired directions. Prompt engineering involves carefully designing input prompts to incorporate relevant context, objectives, and user intent, which helps to ensure that generated responses are contextually relevant and aligned with user expectations [22]. This technique has become particularly valuable in scenarios where fine-grained control over model behaviour is required, such as in interactive AI, dialogue systems, and automated content generation [190].

Recent work in prompt engineering has introduced methods to balance coherence, specificity, and creativity in model outputs. Techniques such as prompt chaining, context anchoring, and instruction-based prompts have proven effective for guiding LLMs in tasks that demand high semantic consistency and adaptability [146]. These methods address common challenges in model-mediated conversational interactions, such as semantic drift, where responses lose relevance to the initial input, and output variability [135], which can impact the coherence of multi-turn interactions. Additionally, prompt engineering has been applied to enhance model performance on specific tasks, including knowledge retrieval, sentiment control, and persona-based interactions [248].

Multi-turn Prompting and Few-shot Learning

Few-shot and multi-turn prompting strategies have emerged as effective methods for enhancing the consistency and coherence of LLMs across extended interactions. Few-shot learning provides models with examples within the prompt itself, guiding the language model towards specific response patterns or conversational styles without extensive retraining [22]. By incorporating examples that represent desired behaviours, few-shot prompts enable models to adapt to various

contexts and tasks, allowing for increased versatility and adaptability, particularly in low-data or rapid-deployment scenarios [146].

Multi-turn prompting builds on this by structuring prompts to retain and reference conversational context across multiple turns, which is especially valuable in goal-oriented applications. This approach may allow LLMs to maintain long-range coherence, tracking user intent [265]. Multi-turn prompts are particularly effective in systems that require responses that adapt to user inputs over time, such as customer service or task-based assistants, where retaining context is essential to achieving effective dialogue flow and maintaining user engagement [279, 265]. This research has particular relevance to player interactions within a game context, as conversational applications increasingly demand context-aware responses [190].

Indirect Control

In generative models, latent space navigation provides a mechanism for controlling model outputs by manipulating internal representations. While direct manipulation of latent spaces has been successfully applied in fields like image synthesis—enabling fine-grained control over visual attributes [79], applying similar approaches in dialogue generation is less straightforward. In dialogue systems, latent representations are often high-dimensional and difficult to interpret, making direct control challenging. Instead, indirect methods such as reinforcement learning from human feedback (RLHF) and prompt conditioning offer viable ways to guide latent space response [170].

RLHF has been used to align model outputs with user expectations by training models to maximize a reward signal based on human feedback, thereby shaping responses toward desired conversational traits [291]. This approach has been critical to the enhancement of GPT-like models' coherence, adaptability, especially in applications that required context-sensitive responses [225]. Prompt conditioning also indirectly influences latent representations by structuring the input prompt to adjust responses without retraining or direct fine-tuning [249]. As an alternative to human feedback, Zhou et al. [282] use LLMs to provide feedback to their RL agent specifically for NPC improvements in story-telling tasks.

3.4.4 Dialogue Generation

The field of procedural dialogue and narrative generation has rapidly evolved in recent years, bridging authored content and generative approaches to improve narrative flexibility. A notable early contribution came from the game *Fallen London* [54, 124], which introduced modular, self-contained narrative units called “Storylets”. The use of storylets has since become a foundational concept in modern interactive systems. Tools such as StoryAssembler [63] used libraries of storylets selected at runtime based on game context to improve variation and reusability. Work by Kreminski and Wardrip-Fruin [124] formalized the design space of storylet-based narrative systems and highlighted their potential for enabling context-sensitive storytelling.

Deliberative narrative systems extend the modularity of storylets by using planner-based generation methods to construct narrative paths. These systems construct story paths by sequencing authored narrative fragments that direct the game flow. For example, the Lume system [152] uses a combination of authored content nodes and runtime selection to produce interactive, responsive stories.

More recent research has focused more on storytelling using large language models. Peng et al. [175] used LLMs to generate unscripted narratives based on player input. Their method used a knowledge graph to demonstrate how LLMs can lead to emergent dialogue based on user input. Similarly, Drama Llama by Sun et al. [228] is a framework that defines a system based on the storylets concept called “triggers”. The framework allows authors to define the narrative beats or key moments or events in the story, in natural language, but delegates the details of the narrative text to an LLM. This approach lowers the authorial burden while maintaining human control of the high-level story structure while guiding the LLM generation and preserving the story structure.

Diverse Non-homogeneous Texture Synthesis

From A Single Exemplar

Texture refers to the visual and tactile qualities of a surface, defined by variations in elements such as colour, shape, pattern, and structure. In digital media, texture plays a key role in conveying material properties, enhancing realism, and creating aesthetic appeal. For example, the texture of weathered wood, smooth marble, or rough asphalt communicates essential information about the material and its environment.

In computer graphics and generative models, texture is often treated as a statistical property of an image or a pattern that can be synthesized or manipulated. This involves capturing both local features, such as fine details and patterns, and global relationships, such as overall structure and variation across a surface. Textures can be categorized as homogeneous (uniform across their extent) or non-homogeneous (exhibiting significant variation and complexity). Non-homogeneous textures, such as rusted metal, cracked paint, or natural geological formations, are particularly challenging to model and synthesize due to their irregularity and lack of repetitive patterns.

In this chapter, we investigate the specific role of texture synthesis within the larger context of generative models for creativity, with a focus on diversity and controllability. These attributes are important not only for generating realistic images, but also for enabling a wide range of possible outputs that could be useful in creative and scientific applications. Traditional methods in texture synthesis provided limited diversity and struggled with complex, non-homogeneous textures. Our

exploration seeks to address these gaps by introducing novel architectures and training techniques that generate a wider variety of texture samples while preserving essential visual qualities of the input.

We focus on textures in a creative context, but the method is also applicable to other image modes such as medical image generation, for example.

Diversity in generated textures is a long-standing challenge in computer vision and graphics. High variability within an output is desirable for applications in creative fields, where diversity and the ability to choose from a wide range of options are critical to creative workflows. Techniques using GANs and VAEs have advanced the state of the art, but they often struggle to balance quality with diversity, particularly for textures with irregular, non-stationary structures. Models, such as StyleGAN [111, 113, 112] and BigGAN [20], have previously addressed some aspects of diversity but are typically designed for large, labelled datasets rather than textures or single image exemplars.

In this chapter, we build upon these advancements by developing a model that can generate a diverse range of high-quality textures from a single exemplar. By blending the strengths of GANs and VAEs in a hybrid, parameter-sharing architecture, we achieve a model that can capture the fine details and complex structures characteristic of non-homogeneous textures. The method operates on a latent representation of the input as the VAE samples from this space as a source of diversity. Our work also introduces a new loss function approach that incorporates a similarity loss term that encourages output diversity without sacrificing quality. This model allows for rapid training and produces outputs that are superior to existing state-of-the-art single-exemplar GAN methods, as demonstrated in our evaluation.

The method presented in this chapter is part of the thesis’s broader investigation into mechanisms of control for generative models looking to creative outputs with application to fields requiring content generation. The connection of the work to latent sampling lays the foundation for the questions that motivate the work in the subsequent chapter.

4.1 Introduction

Stationary textures, whose characteristics do not vary spatially, have been studied extensively, but non-homogeneous textures have remained challenging [285, 59]. Non-homogeneous textures are those whose visual characteristics vary across the image plane. Examples might include a weathered metal surface with rusty patches and peeling paint, or a rocky surface with large and varying cracks. In this paper, we describe an approach for synthesizing multiple non-homogeneous texture instances from a single exemplar.

Recently, a new class of single-image Generative Adversarial Networks (GANs) for natural images has emerged. As with parametric texture synthesis models, the single-image GAN seeks to model the internal image statistics [207]. However, the single-image GAN framework is capable of generating non-homogeneous textures, an area where existing texture approaches struggle. A key objective is to generate diverse variations given only a single exemplar, a challenge which we tackle in this paper. Our proposed method is designed to increase diversity of output, while maintaining high quality.

In a typical GAN framework, the training data includes many images and captures a wide range of features so as to be able to generalize across multiple modes, representing the variation in the data distribution. Capturing these modes is critical for generating diverse samples. Single-image GANs do not have access to modes in the traditional sense and instead are forced to rely on the content of the single exemplar, including both low-level features and large-scale image elements.

Our objective is to use a single exemplar to generate structurally diverse outputs. We take inspiration from SinGAN [207] and other follow-up works such as ConSinGAN [85] as we introduce a novel architecture and training regime designed to better learn the distribution of the input image, exploiting the capabilities of both Generative Adversarial Networks (GAN) [69] and Variational Autoencoders (VAE) [120, 84, 74]. We base our work on the premise that the VAE will provide the initial structure and stabilize the early training and the GAN will refine the fine details in later stages.

This paper introduces a new training method and architecture for single-exemplar texture synthesis.

The architecture is a combination of a GAN and VAE, which share information via structurally identical, independent blocks (SIIBs) present in both networks. This approach achieves greater diversity by sampling from the VAE’s latent space and incorporating features from the decoder into the GAN’s internal layers. We demonstrate that by combining the networks in this way, we can generate multiple high-quality variations of the input exemplar from a single model. To assess our method, we perform multiple experiments and comparisons both with existing texture synthesis methods and with single-image GANs.

We present the following contributions:

- A dual-objective texture synthesis network that combines the properties of GAN and VAE based networks, capable of generating a diverse set of non-homogeneous textures from a single training exemplar. We achieve this by sampling from the VAE’s latent space and processing the parameters so that they can be injected into the GAN’s internal layers during training.
- The associated training regime that allows the VAE and GAN to remain independent networks but be trained jointly, sharing parameters between the SIIBs present in both networks in an alternating fashion. Using this approach, our method can produce a trained model in approximately 15 minutes; the resulting model can generate a diverse collection of interesting outputs despite having been trained on only a single input exemplar.
- A new regularization technique in the form of a sampling similarity loss for GAN training. The similarity loss preserves output diversity and improves the overall quality of the output.

4.2 Related Work

4.2.1 Single Exemplar Texture Synthesis GANs

Algorithmic methods such as Image Quilting by Efros and Freeman [52] reassemble a new image using patches copied from the exemplar. The early parametric methods by Portilla and Simoncelli [181] involved designing descriptors to represent image features and parameterize a statistical model. Deep generative methods for texture synthesis aim to create new texture

samples with equivalent statistical properties to a given example texture. Such approaches using convolutional neural networks (CNNs) have been shown to outperform handcrafted features. Gatys et al. [65] demonstrated the advantages of CNN-based features by employing the Gram matrix to compute correlations between feature responses. While their method showed significant improvement, it was computationally intensive and its best performance was limited to stochastic textures. This is due to the use of the Gram matrix which computes the pairwise inner product of feature maps for each network layer, resulting in a stationary description of a texture exemplar with the spatial information discarded. Subsequent work aimed to improve efficiency by offloading optimization to a training stage, allowing for real-time inference. Generative Adversarial Networks (GANs) [69] enable further improvement, as they learn image features directly from the textures they are trained on using an end-to-end optimization.

Notable GAN-based texture synthesis methods include the Markovian GAN (MGAN) [136], SpatialGAN (SGAN) [98], Deep Correlations (DeepCor) [205] and Periodic Spatial GAN (PS-GAN) [16]. However, despite the improvements from these and other methods, synthesis of non-homogeneous texture remains a challenge. One approach that has shown promise in this domain is to employ an additional guidance mechanism, such as label maps [149] or procedural priors [71, 118]. Zhou et al. [285] developed a texture expansion approach that operates on non-stationary textures, learning the underlying distribution and synthesizing pixels to match that distribution, thereby enabling the expansion. However, their method suffers from a lack of diversity. Our method learns an unconditional model from random noise based on the information contained in the exemplar without guidance maps or other prior. Zhou et al. [284] later showed good results, focusing on a conditional GAN to control the structure of texture outputs. Using their guided correspondence method, they demonstrate fine-grained control of the spatial output. Xie et al. [260] propose a texture expansion model that combines generative adversarial networks and graph convolutional networks (GCNs) [122]. The GAN learns local texture details while the GCN captures global pixel dependencies using a graph similarity loss. The model also incorporates an attention mechanism to focus on important texture regions.

4.2.2 Unconditional Single-Image GANs

Single-image GANs are unconditional models designed to generate diverse samples from input noise based on a natural image exemplar. The approach generates high-quality texture, but the overall diversity of the output is limited. Despite this, the unconditional nature of these networks has been shown to be effective for a variety of image manipulation tasks [207]. This approach has also been extended to video synthesis [5, 75] where a video can be generated from a single image or new diverse videos trained from a single video [76]. A key factor in the success of this approach is the computation of overlapping (virtual) image patches at multiple scales. Learning image statistics of individual patches along with the distribution of patches maintains coherence in the output, leading to plausible synthetic outputs, even for natural images where relationships between spatial components is important (consider the locations of sky and ground pixels in images of outdoor scenes). Despite excellent results, these methods have a limited ability to generate diverse outputs. The diversity arises as a natural consequence of noise injection methods that have become a standard component in the training pipeline for GAN models as a method of stabilizing training and adding diversity. In the case of SinGAN [207], the authors combine noise maps with upsampled outputs of the previous training scale as input for the current scale as a way of promoting diversity.

It is also possible to extract some level of diversity by finding a balance during the optimization process as a form of exploration vs. exploitation experiment. Hinz et al. [85] demonstrate that the level of diversity can be modulated by adjusting the learning rate during training, though this comes at the cost of image fidelity and may have other negative effects on the optimization process. While focused on video synthesis, Gur et al. [75] use a VAE-based approach that demonstrates the VAE’s potential for producing diverse outputs. Their method produces highly diverse output by combining VAE and GAN but with reduced quality and long training times. Our method builds on each of these single-image methods to reduce the trade-off between quality and diversity while also reducing the overall training time. Combining autoencoders and GANs to form a hybrid network is not new. Larsen et al. [128] proposed a hybrid VAE/GAN that leverages the GAN discriminator to replace the typical VAE reconstruction error, resulting in significantly improved reconstruction quality of the generated output. VEEGAN [224] includes an autoencoder into

the GAN loss function that operates on representation vectors to better recover the true data distribution and thus avoid the mode collapse problem. Bao et al. [10] combine a VAE and GAN to map an input image to a latent representation that is used as input into the GAN along with the desired category. In our approach, we maintain two separate networks and avoid the challenges [10] encountered in naive combinations while retaining the standard configuration of both networks. Shocher et al. [214] learn the internal patch distribution of the input using an automorphic approach, single image GAN. They use a second encoder-encoder network to reduce the effects of mode collapse on the patch reproduction. Their method preserves the overall structure of the output (distribution) but with a different arrangement, resulting in changes to smaller image structures while maintaining good coherence. The approach requires up to four hours of training on a single V-100 GPU per image.

4.3 Method

We present a novel training approach and corresponding architecture intended for single-exemplar texture synthesis. The method is motivated by the observation that VAE sampling could impart additional feature variation during GAN upsampling and refinement. VAEs are known for their ability to learn a latent space representation of the input data, allowing for the generation of new data instances by sampling from this latent space. The process of sampling can introduce feature variations that might not be explicitly present in the source data.

The proposed training method is fundamentally a composite of two core components: a GAN and a VAE network that share information during the training process. We define *structurally identical, independent blocks* (SIIBs) to exchange information between the VAE and GAN networks. An SIIB consists of a designated block of convolutional layers, present in both the VAE and the GAN network as a pair. One pair of SIIBs appears at each scale of the training. We train each scale for n iterations, where the VAE is trained for a portion of n after which the parameters of the VAE's SIIB are copied into the GAN's paired SIIB. The GAN continues training the remainder of the n iterations until the training for the given scale is complete. Then the parameter weights are copied back to the VAE and the process continues as the next scale is added to the network.

Both networks are trained at multiple scales following a discrete, logarithmic progression. During training, the gradient only flows through either the VAE blocks or the GAN blocks at one time. Generally, if we specify S scales where S is a hyperparameter, we only include the VAE network in a subset S_{VAE} of S , where S_{VAE} is also a hyperparameter. The value is selected empirically based on image size.

4.3.1 Network

We use a standard convolutional, variational autoencoder. Our encoder, using convolutional layers, maps the input image into a compact latent space from which we sample a latent variable. We deviate slightly from the usual decoder as we upsample until the tensor shape matches the spatial size and depth required by the SIIB. The output of the SIIB requires an additional convolutional layer to achieve the desired output shape which is used to compute the loss. This convolutional architecture enables our VAE to effectively capture local spatial structures in images and generate realistic outputs.

Using the VAE network, we randomly sample from the latent space z at each iteration and process the decoded output through the SIIB to learn features from this probabilistic sampling. We use this approach to introduce additional diversity to the GAN by injecting the VAE parameters into the internal layers of the GAN. As the GAN is trained, we can add additional diversity by sharing the GAN parameters and repeating the sampling process. In this way we can learn an unconditional GAN from random noise while also maintaining the integrity of the individual networks' contribution instead of creating a hybrid model (as do Bao et al. [10], for example).

The VAE features have a large influence in the early stages of training. We only need to take advantage of the probabilistic properties of the VAE at lower scales; thus, most of the training time in our model is dedicated to the GAN. We use a standard GAN formulation with a generator and discriminator. The generator learns a mapping between an input latent vector $z_{w \times h}$ and some generated image output $x_{w \times h}$, representing increasing spatial resolutions in our multi-scale training regime. We use a standard patch-GAN discriminator [95] that is initialized at each stage with the parameters of the previous stage (see Figure 4.1). In the generator, the SIIB weights are used to initialize the SIIB for the next scale. The weights of previous SIIBs are not frozen and

thus the gradient flows through all SIIB scales.

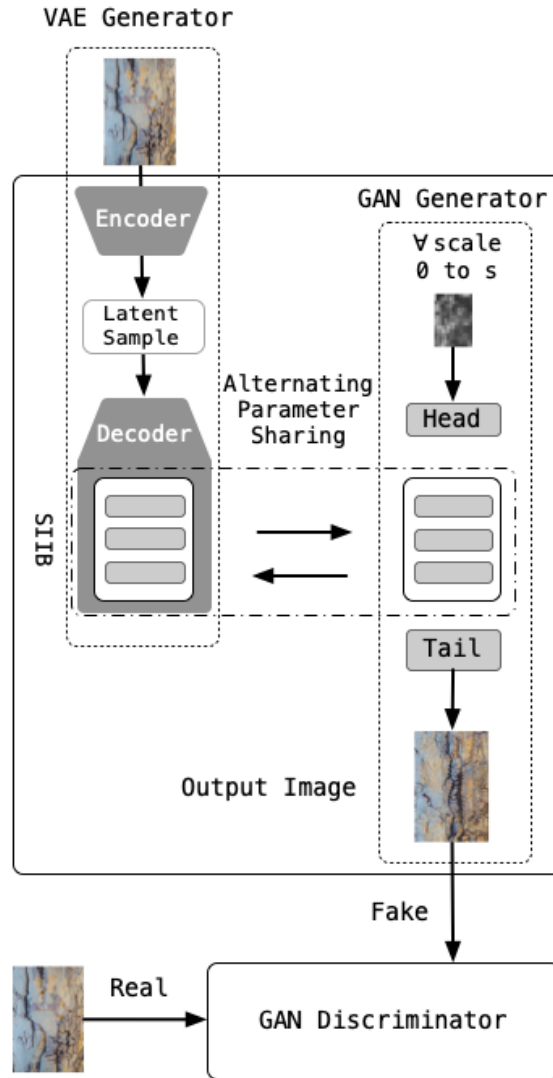


Figure 4.1: **Architecture.** The VAE and GAN components of our method. We sample from the latent space and process the results in a CNN block. This allows progressive but alternating training with the GAN generator by sharing parameters. In so doing, we can take advantage of the probabilistic sampling of the VAE while refining details using the GAN in later stages. A new SIIB is added to the network at each scale.

Discrete Multi-scale We train the GAN using multiple scales from coarse to fine. The coarser

scales learn the global arrangement of image features, dominant shapes, and coarse patterns. At the finer scales, the model focuses on texture details, i.e., localized information including more subtle changes to color and intensity. We use residual connections [81] between the input features and the new SIIB at each scale to progressively refine the details at each scale and stabilize the training process. The network learns relationships between different levels of abstraction, which leads to improved consistency of the output [95].

There are numerous approaches to multi-scale training [115, 111, 113, 109, 207, 85]. Here, we adopt a discrete growing approach [85], where after the initialization of the network, the depth of the network is increased by adding the SIIBs between the fixed head and tail blocks of the networks. A new SIIB is added after the specified number of iterations per scale is complete and the training of the next scale begins. These SIIBs are identical between the GAN and VAE: one block in each network for each scale. The addition of each new block corresponds to the next scale towards the target output resolution.

The output from each processing stage is upsampled and integrated with the input of the subsequent stage. The integrated output, after being processed through the network, contributes to the final output through a skip connections, where features from various scales are dynamically combined and preserved up to the final layer. With each scale, the depth of the network is increased.

The configuration of the resolution at each scale is important. We observed that when scaling output resolutions geometrically, the results were inconsistent and required more stages to achieve reasonable output. This is consistent with the findings of Hinz et al. [85], and following their recommendations, we train more scales at lower (coarse) resolution, using this progression:

$$x_s = r^{(S-1 \log(S)) * \log(S-s)+1}, \quad (4.1)$$

where x_s is the output at scale s , and S is the total number of scales. The base scaling factor r is computed as follows:

$$r = \left(\frac{d}{\min(x_w, x_h)} \right)^{1/S} \quad (4.2)$$

where d is the smallest pixel dimension in the lowest scale.

Using x_s to scale each image dimension, we build an image pyramid with more images at low resolution. This results in a more stable training process than the traditional geometric approach, which either does not converge or converges with lower-quality output [85].

Training Since the structure of the growing blocks is shared between VAE and GAN generators, the training of the two networks proceeds together. Unlike Gur et al. who train a single network, passing the output of the VAE as the input to the GAN, our training is performed jointly, but with different objectives and in an alternating pattern. The training proceeds with S scales up to the final scale s_{final} , from coarse to fine, for n iterations per scale. The GAN is trained up to s_{final} , but generally the VAE only needs to be trained for a subset of S . At each scale, the VAE is trained for $n \cdot p$ iterations, where p is a hyperparameter for the proportion (default (0.8)) of each scale that the VAE generator will be trained. By adjusting p , the contribution of the VAE and thus the diversity can be modulated. The GAN is trained for the remaining iterations. The value of p can be tuned for a given input. A poorly selected value of p results in a reduction of fine details in the output, worse if p is far out of range.

As the training alternates, the parameters of the VAE SIIB(s) are copied into the corresponding block within the GAN generator. When the training for each scale is complete, the GAN copies its block parameters back to the VAE block before beginning the next scale. This parameter sharing effectively allows us to optimize for different objectives jointly. This has the effect of imparting more diversity into the network than is possible with the GAN alone, while also maintaining a high degree of quality with respect to the input, given the large structural deviations from the original.

We use the adversarial loss from Gulrajani et al. [73] to compute the distances between patches from the generated and true distribution of overlapping, virtual patches. Combined with the patch-GAN discriminator [95], the adversarial loss includes a gradient penalty term to enforce the Lipschitz constraint. The constraint limits the norm of the gradient to 1 along uniform samples taken along lines sampled from the data and generator distributions.

4.3.2 Sampled Similarity Loss

Overfitting is a common problem for GANs, especially when training with limited data [73, 116], and was often a problem in our experiments with single-image GANs. For example, we frequently encountered the scenario where we must train to some minimum number of iterations per scale to obtain acceptable quality, but as the training time increases, the network begins to overfit on the input and diversity present in intermediate output is lost as training converges towards the input image. Tasks such as recognition or classification usually obtain significant benefit from data augmentation [217]. However, when training GAN models with augmented data, the network may learn the distribution that contains the augmentations, which may then appear in the synthetic output. In the single-image case, the effects of augmentation are much stronger and approaches like that proposed by Karras et al. [116] are not able to prevent the augmentations from leaking into the output. In the target domain of non-stationary textures, some augmentations such as rotations and reflections may be acceptable, though in our experiments we often observed a negative impact on the plausibility of the output and hence do not recommend relying on data augmentation in this application.

We observed that a key marker of impending overfitting is the reduction of diversity within some sample set. This manifests as spatial features converging to match the same features in the input, but this can only be evaluated over a multi-image sample. To compensate for this problem, we propose an additional regularization to enforce diversity in the form of a sampled similarity loss (SL). In the early stages of the training, the diversity remains high because of the larger receptive field. In later stages, when the GAN is trained without the VAE, our similarity loss mitigates the propensity of the network to overfit.

Sampling Given a set of images $I = I_1, I_2, \dots, I_n$ with $n \geq 2$, we propose a function to measure the diversity among them using a feature extractor F , which is a pre-trained deep neural network model. This function extracts high-level features from a set of predefined layers in the feature extractor F . To compute the SL, we use the pre-trained feature extraction network VGG19 [219] trained on ImageNet [44]. We operate within the sample space of a given training procedure of our model at an interval of 100 iterations. At each interval, we sample n (25 is the default) fake images from the current model state and compute the average pairwise $L1$ norm of the differences

between the feature representations extracted via selected layers (layer indices 2, 7, 12, 21, 28 corresponding to ReLU1-1, ReLU2-1, ReLU3-1, ReLU4-1, and ReLU5-1 in VGG19). We also normalize the feature vectors to emphasize patterns of features over magnitudes. A scaling hyperparameter ($\lambda_{scaling}$) allows control the overall strength of the regularization.

To summarize, let us denote by I the set of all images in the sample, L the set of target layers indices in the feature extractor model ([2, 7, 12, 21, 28]) and w_l the weight for each layer $l \in L$ (by default all weights are set to 1). The features of image i at layer l are denoted $f_{i,l}$, and the normalized version is given by $f'_{i,l}$.

SL computation ($loss_{div}$): For each layer $l \in L$, compute the SL as the average L1 norm of the feature differences across all pairs of images:

$$p_l = w_l \cdot \frac{1}{\binom{n}{2}} \sum_{i \neq j} |f'_{i,l} - f'_{j,l}|_1 \quad (4.3)$$

where $\binom{n}{2}$ is the number of unique pairs of images.

The final SL is the average SL across all layers:

$$\phi_{SL} = \frac{1}{\frac{1}{|L|} \sum_{l \in L} p_l \cdot \lambda_{scaling} + \epsilon} \quad (4.4)$$

4.3.3 Combined Diversity/Quality Metric

To better reflect the desired balance of quality and diversity in the analysis of our results, we summarize the existing standard metrics into a single value that captures how well a given method reduces the diversity vs. quality trade-off. The goal of this metric is to balance quality and diversity, while focusing on the importance of quality. Squaring the diversity in the denominator creates a non-linear relationship meant to capture the concept that excessive diversity without quality is undesirable.

To compute the Combination of Quality and Diversity (CQD) summary statistic, we first calculate

the standard deviation at each pixel location across all images and color channels (i,j,k) from a sample of outputs. Following the approach of Shaham et al. [207], we compute the mean of the standard deviation of the pixels in the sample. In our method, we perform the computation separately for each color channel to capture the full range of color information. The result is then normalized by dividing the mean standard deviation by the standard deviation of the corresponding pixels in the training image, as follows:

$$\sigma_{i,j,k}(P) = \sqrt{\frac{1}{N} \sum_{n=1}^N (P_{n,i,j,k} - \bar{P}_{i,j,k})^2} \quad (4.5)$$

Where $P_{n,i,j,k}$ is the intensity value of the pixel at position (i, j, k) in image n . $\bar{P}_{i,j,k}$ is the mean intensity value of the pixel at position (i, j, k) over all images.

$$\delta_{div} = \frac{\frac{1}{M} \sum_{i,j,k} (\sigma_{i,j,k})(I_{fake})}{\sigma_{i,j,k}(I_{real})} \quad (4.6)$$

Then, we combine SIFID with the diversity metric as follows:

$$CQD = \frac{SIFID}{\delta_{div}^2} \quad (4.7)$$

Where lower values of SIFID represent higher quality.

4.4 Experimental Results

We compare our results with those from both standard texture synthesis approaches and single-image GAN models. We demonstrate increased ability to generate diverse textures compared to previous methods.

4.4.1 Experimental Settings

All results in this section use the same configuration parameters. Here we describe the details of the configuration we used.

Dataset Since we present a single-image method, the network is only trained on the exemplar itself. The individual images used in our analysis were selected to represent a variety of natural non-homogeneous texture types. Each result is the output of an individual training run.

Training configuration For these experiments, we train each target for 6 scales of increasing size beginning at $d = 25\text{px}$, resulting in 400px on the longest dimension. We train for 2500 iterations per scale using the learning rate schedule described by Hinz et al. [85]. The joint training is performed for 3 out of 6 scales, where the VAE is trained for 1250 of 2500 iterations per scale and the GAN trains the remainder. After the first 3 scales the GAN continues training alone. Training the VAE for additional scales results in lower output quality.

Each SIIB uses 3 2D convolutional layers with 64 in and out channels and a kernel size of 3. After each layer, we use instance normalization followed by leaky ReLU activation.

Training and inference results were obtained using a single RTX4090 GPU. CPU-only methods were computed using a 2.6GHz-4.5GHz 6-core Intel Core i7 processor.

Sampled Similarity Loss We sample for and compute the similarity loss every 100 iterations at all scales. We apply a scaling factor of 1.2 to the computation of the loss. In cases of long training (> 5000 iterations), eventually the loss becomes ineffective, as the effect of overfitting becomes overwhelming.

Additional Settings In both VAE and GAN we employ gradient norm clipping to stabilize the training, with no appreciable underfitting.

4.4.2 Texture Synthesis Methods

Figure 4.3 presents a comparison between our method and a selection of texture synthesis methods known to have reasonable performance for non-homogeneous textures. To accommodate 1:1

aspect ratio requirements, we center-crop on the long axes from our original source images. Our method captures the information available in the input more effectively to produce plausible variations while maintaining a reasonable level of quality. Zhou et al. [285] learns the distribution but without diversity. They specialize in upsampling by synthesizing new local features. Sendik et al. [205] shows the most promise in learning the distribution, but the plausibility and quality are not fully realized. Finally, Efros and Freeman’s patch-based approach [52] captures details, but the Markovian assumption of locality does not hold for non-homogeneous exemplars. In ??, we show outputs at different aspect ratios and resolutions.

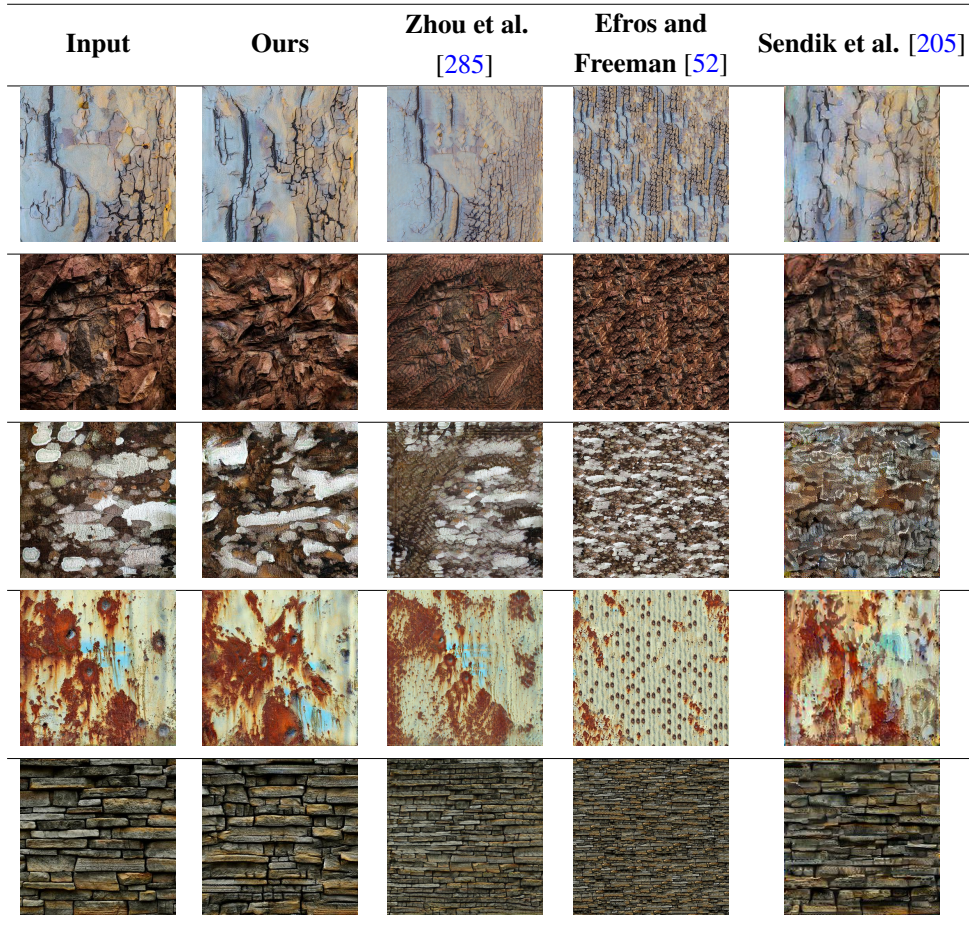


Figure 4.3: **Qualitative Diversity vs. Texture Synthesis Methods.** Compared to other texture synthesis methods, we are better able to produce natural feature variations at the same scale as the original while approaching the quality of the original input image. Due to variations in image handling across methods it was not possible to train all compared methods to an equivalent size; some resizing was performed (Efros and Freeman: 11px downsample (ds), Zhou et al.: 400px ds)

4.4.3 Single-Image GAN Baselines

In Figures 4.4a. and 4.4b., we compare our results to popular single-image GANs. Our method is capable of more diverse results, while maintaining plausibility. As diversity increases away from the global structure of the input, one might expect to see a degradation in the overall quality. Instead, we note that the image quality of our method is largely maintained and approaches or

matches the quality of Hinz et al. [85] which has less diversity. The results from Shaham et al. [207] and Hinz et al. [85] demonstrate limited ability to produce diverse results and converge towards the global structure of the input with minor variations of large-scale structures. We suspect the poor results from Shaham et al. are related to their scaling strategy used to decide the training stages. The network of Gur et al. [75] produces a similar level of diversity as our method, but the results are of lower quality, with a pixelated, grainy appearance (which disproportionately affects quality metrics).

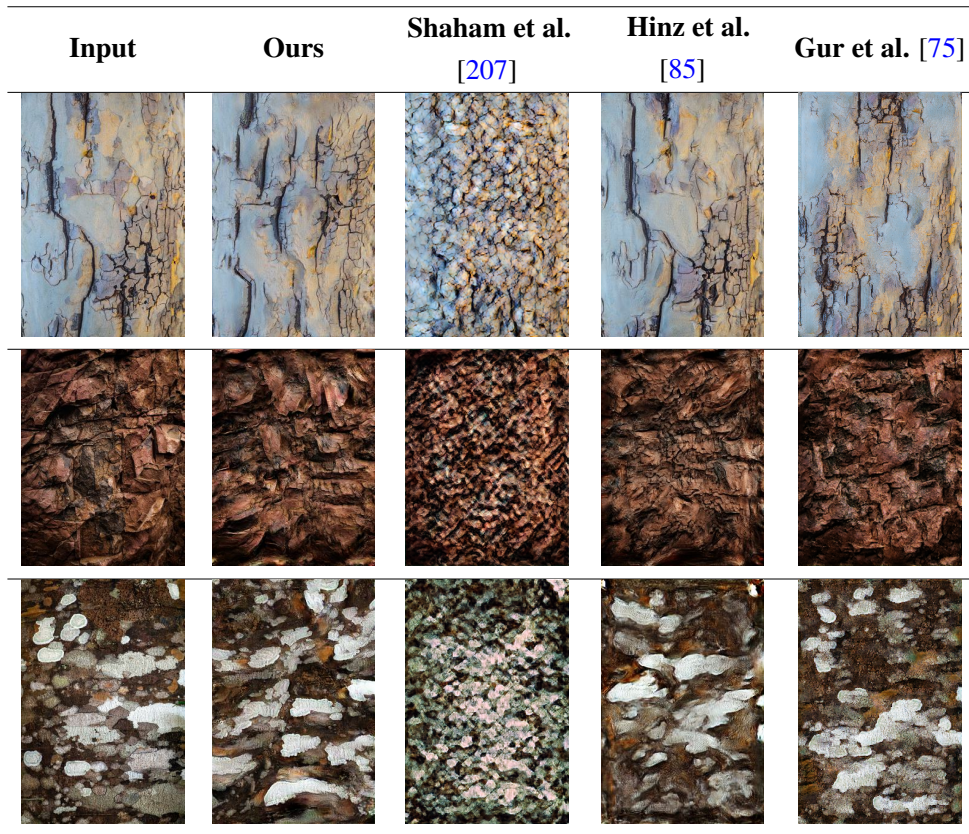


Figure 4.4: (A) **Qualitative Diversity vs. Single Image Methods.** Here we present an assessment of the quality and diversity of our method compared to single image methods. The same hyper-parameters were used for each image for the respective methods. We show that our method is better able to produce variations that accurately reproduce features of the exemplar at the appropriate scale compared to the other methods.

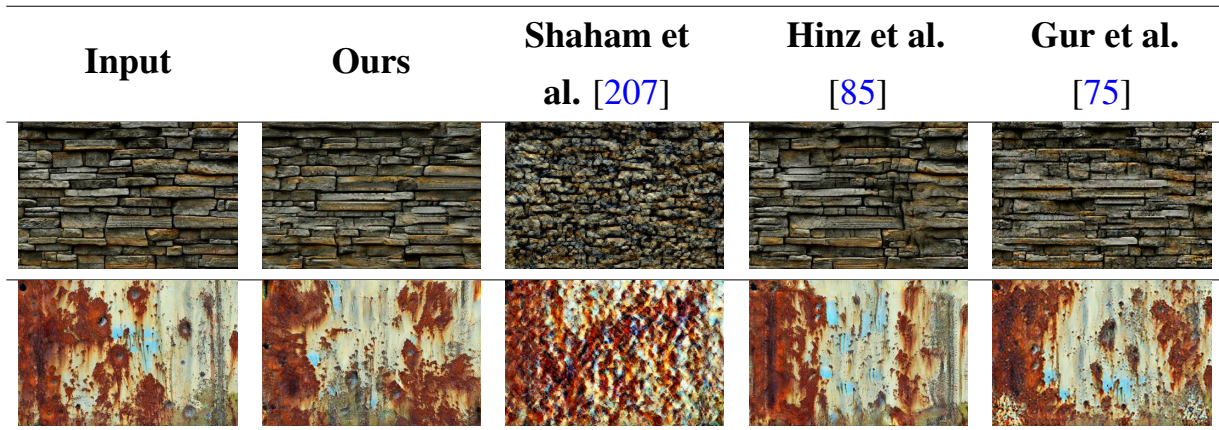


Figure 4.4: (B) **Qualitative Diversity vs. Single Image Methods.** Here we present an assessment of the quality and diversity of our method compared to single image methods. The same hyper-parameters were used for each image for the respective methods. We show that our method is better able to produce variations that accurately reproduce features of the exemplar at the appropriate scale compared to the other methods.

4.4.4 Training Time

Table 4.1 summarizes the training time across the considered models. For each method, we used the authors’ recommended training approach with respect to hyperparameter settings and made adjustments to ensure convergence and prevent overfitting. All networks can produce images in under 100ms at inference time (we produce up to 800x534 images in approximately 10ms, for example). The exception is Efros and Freeman, where the algorithm must be run once for each desired image. Since this approach is non-parametric and does not learn a model, compute time scales linearly with the number of desired outputs.

Model	Average Training time (min)
Shaham et al.	30
Hinz et al.	18
Gur et al.	90
Zhou et al.	80
Efros and Freeman [†]	3
Sendik et al. *	13
Ours	13

Table 4.1: **Training/Processing Time.** Average training times for each method when producing a model for each of the five test images. * This method was trained using CPU. †This method does not involve training; processing time is reported instead.

4.4.5 Quantitative Evaluation

Quality We report the Single Image FID (SIFID) metric proposed by Shahan et al. [207], a specialization of Fréchet Inception Distance [83] designed for single images. The metric compares activation statistics between generated samples and the real image. The SIFID scores serve as a confirmation that the output quality approaches that of the original rather than a reliable measurement of absolute quality. We found that despite the additional diversity imparted by our method, our results’ SIFID scores are very close to those of Hinz et al., whose method achieved the highest scores shown in Table 4.2. Hinz et al.’s textures have a spatial output arrangement very similar to that of the exemplar image, much more so than ours. This suggests that our method is effective at preserving detail while providing the desired level of variation.

	Shaham et al.	Hinz et al.	Gur et al.	Ours
Cracks	13.6714	0.0345	13.7316	0.0458
Rock Face	15.0323	0.0350	15.0698	0.0313
Lichen	17.0920	0.0804	17.1601	0.0599
Weathered	18.2520	0.0384	18.1574	0.0427
Stone Wall	17.4986	0.0241	17.4591	0.0393

Table 4.2: **Quantitative Analysis using SIFID.** SIFID scores for a sample of 25 images. Lower is better. Hinz et al. obtain good results. However, even with our increased diversity, we remain competitive. Due to the limitations of the SIFID metric, the true quality of Gur et al. is not fairly represented in our opinion; visual inspection of Gur et al. suggest their output is also competitive.

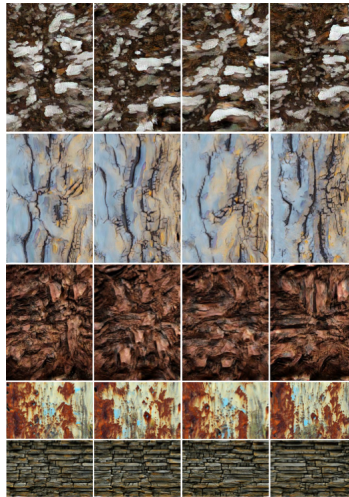


Figure 4.5: **Additional diversity examples** Here we show additional output samples of the five test images.

Diversity Ablation We measure diversity using the standard deviation described in (see Section 4.3.3). In Table 4.3., we show that our method is able to generate diverse variations of the input.

We compute the result of our proposed method against the baseline where diversity components are not computed. In Figure 4.6, we show that our claimed contributions are responsible for

affecting the change in the output consistent with our results with respect to diversity and quality. The figure compares one set of intra-model samples generated using our proposed method with another set from the identical network and hyperparameters but with the diversity promoting mechanisms disabled. The results demonstrate a clear difference in the level of diversity between the two models. The baseline samples do show some intra-sample diversity, but the variation is limited to localized, smaller scale structures, while the global arrangement of features is largely unchanged. We conclude that our proposed method is responsible for the improved diversity. We note that in the case of Hinz et al. [85], per our experiments, reducing the number of scales has the greatest effect on diversity though still to a limited degree, but this comes at reduced quality. This scale dependent diversity is only apparent at the model level. That is, within a single trained model the intra-sample diversity is low. Here, we show that our method overcomes these limitations and provides high intra-sample diversity within a single model.

Table 4.4 shows the effect of our sampled similarity loss on output diversity. When training models for 4500 iterations per scale (vs. 2000 in the above), the SL works to maintain diversity. The table shows the diversity score with and without the SL activated. We also include the associated SIFID which further illustrates how the SL also contributes to improved output quality.

	Shaham et al.	Hinz et al.	Gur et al.	Ours
Cracks	1.0984	0.5600	0.8520	<u>0.8764</u>
Rock Face	0.9903	0.6053	0.7254	<u>0.7973</u>
Lichen	1.0266	0.8805	0.8856	<u>0.9557</u>
Weathered	0.5060	0.5915	0.8161	<u>0.7062</u>
Stone Wall	0.9966	0.5515	0.7232	<u>0.8512</u>

Table 4.3: **Qualitative Analysis using Image Diversity.** Aggregated from a sample of 25 images. Higher numbers are better. Shaham et al. obtain the highest scores for all but one image. We note that the diversity score is not indicative of the plausibility of the output. For additional clarity, we underline the second-highest value.

	Diversity \uparrow		SIFID \downarrow	
	SL Off	SL On	SL Off	SL On
Cracks	0.6900	0.7334	0.0518	0.0406
Rock Face	0.7803	0.7827	0.0196	0.0291
Lichen	0.8762	0.9118	0.0574	0.0470
Weathered	0.6068	0.6179	0.0666	0.0404
Stone Wall	0.7991	0.8726	0.0529	0.0400

Table 4.4: **Ablation of Similarity Loss.** Here we show the effect of the similarity loss (SL) on diversity and quality. We train the model with identical settings except enabling the SL. We also show the corresponding SIFID score. The results demonstrate that use of our SL results in images with improved diversity while also improving image quality according to standard metrics across a diverse set of images.

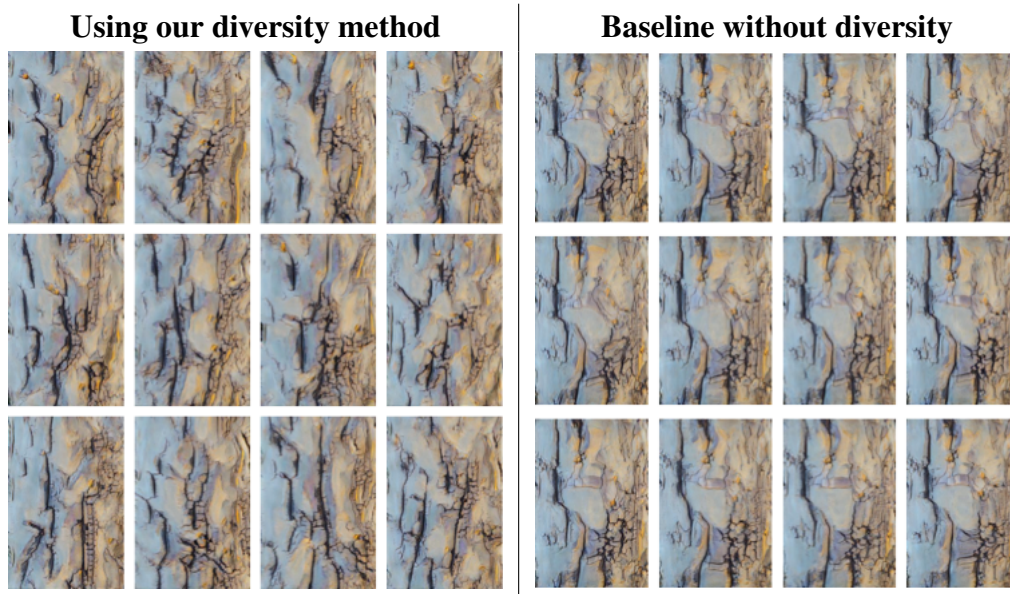


Figure 4.6: **Ablation of our Diversity Method.** Left: random sample of images generated with diversity-promoting mechanisms enabled. Right: baseline results without diversity promotion. The left figure contains significantly more spatial and feature diversity across the sample, while the right image shows limited intra-sample diversity and close resemblance to the input image.

Finally, Table 4.5 shows the results from the integrated diversity/quality metric (see Section 4.3.3.). These results capture a summary of the SIFID quality metric results (see Table 4.2. and Table 4.3.). Our method is effective at narrowing the distance between the quality / diversity trade-off. In Figures 4.5 and 4.7, we show additional examples.

	Shaham et al.	Hinz et al.	Gur et al.	Ours
Cracks	11.3324	0.1099	18.9152	0.0641
Rock Face	15.3272	0.0954	28.6358	0.0492
Lichen	16.2192	0.1037	21.8801	0.0656
Weathered	71.2894	0.1098	27.2642	0.0857
Stone Wall	17.6175	0.0792	33.3801	0.0542

Table 4.5: **Qualitative Analysis Summary.** Results of the summary statistic CQD used to combine quality and diversity metrics. Lower is better.

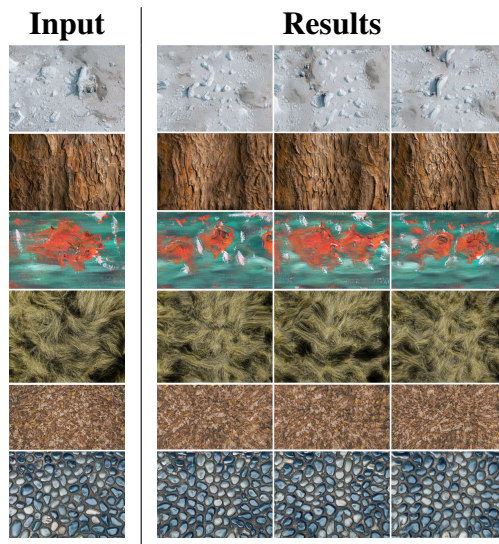


Figure 4.7: **Additional Exemplars.** Here we show additional results on various texture inputs.

4.4.6 Limitations

Our model struggles with highly stochastic textures containing high-frequency fine-details (see Figure 4.8). This behaviour is common to other GAN-based methods that we have studied. We suspect that this is related to the size of the receptive field at the fine scales. Also, since our model is designed specifically to impart variations in the output, it is not well suited to operate on regular/periodic textures (see Figure 4.9). Inputs with regular shapes and repeating, regular patterns can be reproduced but not without distortion.



Figure 4.8: **Stochastic Failure Case.** Stochastic textures are challenging for our method. We note that comparative methods experience a similar failure.

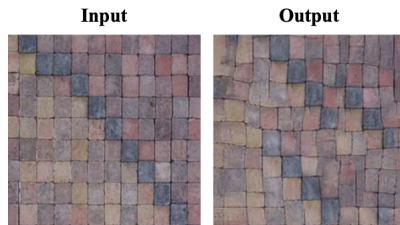


Figure 4.9: **Periodic Failure Case** This image demonstrates line distortion that may occur when generating regular patterns, despite good diversity in the overall pattern.

4.5 Conclusion

In this work, we have introduced a novel method for creating new non-homogeneous textures from a single image. We proposed a shared, multi-scale feature extractor block design that enables the progressive introduction of additional diversity by learning a latent representation that can be sampled, decoded and injected into the GAN. The method optimizes two different objectives by alternating training of the VAE and the GAN and copying parameters of the shared blocks

between networks. Combined with the proposed diversity penalty, the intra-sample diversity and the output quality is further improved in later scales when the GAN is trained alone. We demonstrated that this approach is better able to exploit the information available in the source image while also reducing training time. Compared to previous texture synthesis approaches and the single-image GAN which we build upon, we have shown that our method is able to impart significant diversity, with quality approaching that of the input exemplar.

4.5.1 Future Work

There are a number of avenues for future investigation. Our method is well suited to fast generation of textures, leading to an interest in a method for dynamic combinations of individual outputs in a seamless way. This has potential applications in game development, aiming at the creation of dynamic environments. Another important direction is further development of reliable metrics for measuring diversity. We envision a more robust measurement that considers the spatial distribution of features in the output using a global and local measure of entropy when compared with the input exemplar. This would avoid the pathological outcome where a random signal naïvely results in a high diversity.

4.6 Summary

In this chapter, we addressed the challenge of synthesizing diverse, non-homogeneous textures from a single exemplar, introducing a hybrid architecture that combines GAN and VAE components to provide both diversity and structural coherence. This approach offers a method to guide texture synthesis to balance the complementary strengths of two models. In the next chapter, we expand on the theme of control in generative systems by focusing on the application of CLIP embedding spaces for blending and interpolating concepts. While chapter 5 shifts from example-based texture synthesis to multi-modal embedding-driven generation, chapters 4 and 5 explore different mechanisms for steering generative outputs: one through architectural design and regularization, the other through semantic alignment and traversal of embedding spaces. This progression is reflective of the evolving approaches to control in generative tasks, moving from single-domain methods to broader, cross-domain synthesis techniques.

Blending Difficult to Describe Concepts in CLIP-Space

We continue our investigation into texture synthesis and control, while incorporating ideas from text-based synthesis and multi-modal generative models. Texture synthesis poses unique challenges that distinguish it from other forms of generative image tasks. Unlike scenes or objects, textures often lack clear semantic descriptors, making them difficult to describe accurately using text-based prompts. This becomes particularly problematic when attempting to achieve specific creative outcomes or replicate complex, non-uniform visual patterns. In earlier experiments using latent diffusion models, we explored text-guided texture generation but encountered significant limitations due to the inherent challenges in describing textures. These challenges motivated the need to better understand how texture characteristics are encoded within models such as CLIP and to investigate alternative methods for guiding synthesis. Specifically, we consider how image-based inputs, rather than textual descriptions, might offer a more precise and intuitive mechanism for conveying intent, enabling more effective control of texture generation. We build on these insights to propose an experimental method of guided synthesis for difficult-to-describe textures.

This chapter introduces a technique for blending abstract concepts by extending the CLIP model's embedding space into a traversable representation. Here, we aim to generate image-based outputs that integrate visual elements from inputs that have their respective embeddings aligned within a shared latent space. Synthesizing coherent combinations of difficult-to-describe textures that

can reflect properties of both inputs represents a new form of control when generating creative outputs based on image datasets without training a custom or fine-tuned model.

Through a novel projection and optimization approach to spatially position the contents of a target dataset into an interactive CLIP-like embedding space, we demonstrate the generation of intermediate representations that effectively blend features from distinct inputs. By using this external embedding structure, the method supports direct interactions within the embedding space, making interesting and coherent transformation of the input possible. The approach enables the generation of abstract outputs guided by input images without relying on users to produce descriptive prompts. This addresses a common challenge in generative content creation, particularly for textures that often lack clear descriptions.

The chapter's findings contribute to our understanding of how CLIP can be used for novel use cases beyond its original intent, demonstrating the potential for spatially informed interpolation within embedding spaces that can be constructed for existing datasets. This direction could improve applications in content generation in the early stages of creative workflows, where generating unique image concepts is important.

5.1 Introduction

In recent years, text-to-image generative models have transformed image synthesis, enabling high-quality image creation from natural language prompts. Models like CLIP (Contrastive Language-Image Pretraining) [185] have made an important contribution, aligning visual and textual data from large datasets to form contextual associations. However, certain classes of visual information, particularly textures, present unique challenges, as texture concepts often lack clear semantic descriptions, making them difficult to represent accurately using standard prompt-based methods.

Even more challenging is the creation of coherent blends between distinct textures, which remains a largely unsolved problem. In particular, blending textures that differ in semantic content, spatial arrangement, or internal structure is extremely challenging. Traditional algorithmic methods and supervised training methods are ill-suited to this task as paired training data for texture transitions

is practically nonexistent and there is no practical way to define groundtruth. This makes supervised approaches difficult if not impossible.

This task is not only technically challenging, it is also poorly supported by current models and workflows. There is no general, scalable method for blending arbitrary textures in a way that simultaneously preserves visual coherence, enables diversity in outputs, and supports user control. Despite this lack of method, the task represents functionality that has significant utility in creative workflows. Tools for exploration of image libraries, specifically the image space between reference images, could be a powerful application.

This chapter introduces a novel approach to address this challenge that uses the CLIP embedding space as a latent domain for direct visual concept blending. Unlike prompt-based exploration, which relies on text-driven cues, our method enables interpolation within a constructed latent representation of CLIP’s embedding space to generate blended representations of texture concepts. By constructing a manifold and navigating through the embedding space, the approach bypasses the need to explicitly formulate text prompts when dealing with hard-to-describe textures or concepts.

Previous works have explored text-driven synthesis for texture generation, but our approach offers an alternative by using the geometric properties of a graph-based representation of CLIP’s embedding space. The graph organizes CLIP-aligned embeddings into a latent representation that enables interpolation paths that blend visual features from input textures acting as endpoints of the interpolation path. This leads to the generation of novel and coherent texture outputs.

As such, given two textures as input, we would like to create a third image that blends the textures while preserving visual characteristics and patterns from each. We draw inspiration from CLIP’s capacity to associate complex concepts across text and image representations as the means to explore a method for combining textures directly in the embedding space. Herein, we describe a process that uses the CLIP model to enable the blending of images and the concepts they contain through interpolation.

5.1.1 Motivation and Background

Since the utility of CLIP embeddings derives from the strength of the diverse semantic training [67], of particular interest is whether this embedding space captures enough structured visual and semantic information to support the synthesis of coherent, mixed-concept outputs, especially when the information is difficult to describe. While some textures are amenable to a concise description, e.g., “sandy texture” or “rough bark”, it is challenging to capture subtle variations and other associations that may be present or desirable. The environment, scale, and other ambiguities complicate descriptions and their interpretations. To overcome these challenges, it would be useful to avoid the use of text prompts and navigate to a desirable output using a visual approach using images as input. By blending image embeddings, we test the limits of the ability of CLIP to represent combinations of concepts. If blended embeddings produce meaningful and coherent output, it may indicate that CLIP has the capacity to generalize and encode not just individual concepts but also hybrid or mixed concepts.

By constructing a manifold within CLIP-space, we enable interpolation that would not otherwise be possible, avoiding the need for prompt-based experimentation and instead directly exploring visual combinations. This construction may enable continuous and meaningful traversal between representations. This approach also supports use cases in creative image generation and multi-modal dataset augmentation, particularly where specific or rare textures are difficult to describe textually.

Our approach begins by constructing a CLIP-space representation of the Describable Textures Dataset (DTD) [38], providing a framework for spatial interpolation. To achieve this, we derive CLIP embeddings for each image in the dataset, which we then organize in a low-dimensional manifold suitable for smooth interpolation. By incorporating a graph structure informed by local embedding geometry, we enable coherent transitions across the space, allowing for blending that preserves key visual attributes of each endpoint (see Figure 5.1).

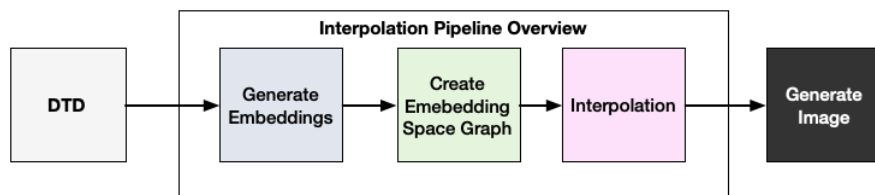


Figure 5.1: **Pipeline Overview** A high-level view of the overall pipeline. Each component is explained in detail below. We use the coloured boxes to map to the detailed diagrams throughout this chapter.

To assess the effectiveness of our method, we evaluate our spatial interpolation approach qualitatively and quantitatively. Our results demonstrate that CLIP can be used to create a virtual latent space that can blend visual concepts, retaining features from both endpoints while introducing novel, coherent textures that go beyond prompt-based generative techniques.

5.1.2 Contributions

In this work, we will describe the following contributions:

- **Graph-Based Representation of CLIP-Space:** We construct a graph-based latent representation of texture concepts using embeddings we generate using the describable textures dataset. This establishes a framework for spatial interpolation in CLIP-space (Section 5.3.2).
- **Novel Spatial Interpolation Technique:** We propose a spatial interpolation method that uses CLIP embeddings to produce coherent blends of textures, demonstrating that incorporating spatial relationships between embeddings enhances the quality and interpretability of combined outputs compared to traditional linear interpolation (Section 5.3.3).
- **An Iterative Projection Technique:** We develop a projection technique that maps images to their corresponding CLIP embeddings more accurately. When combined with an existing optimization method [253], it allows us to create embeddings that more closely represent the features of the input images in CLIP-space (Section 5.3.2).

5.2 Related Work

Our work draws on advances in diffusion models, vision-language representations trained by contrastive learning, latent space exploration, and optimization techniques, particularly those involving CLIP embedding space. We review each relevant area below.

Diffusion Latent Diffusion Image Synthesis [222, 86, 188, 196, 179] initially explored the capabilities of early diffusion models, specifically Diffusion Probabilistic Models (DPMs), which excelled at estimating the density of pixel distributions. However, these models, which operate directly in pixel space, faced challenges such as costly training and suboptimal image quality despite their strong generative potential. A significant advancement was achieved by decoupling the compression and generation phases within the diffusion process. Using a Variational Autoencoder (VAE) [121], high-dimensional inputs were efficiently mapped into a more manageable latent representation. Further enhancements were introduced by incorporating a UNET-based encoder / decoder framework [197] to handle denoising tasks within this latent space. This approach not only boosted efficiency, but also maintained high image quality. Additionally, it allowed for the effective modelling of conditional distributions using the denoising autoencoder framework.

Contrastive Learning Vision language models (VLM) [100] use contrastive learning [100, 31] to align visual and textual representations by maximizing the agreement between corresponding pairs of images and text while minimizing the agreement between non-corresponding pairs. The CLIP [185] VLM does this by mapping visual and textual data into a shared embedding space. Using a contrastive objective, the model attempts to maximize the cosine similarity between matched text and image embeddings encoded by CLIP’s respective encoders while minimizing the cosine similarity between unmatched pairs. In the embedding space, the distance between similar vector representations should be small. Initially, the CLIP model in Radford et al.’s work [185] was trained on 400M image-text pairs, but this was later expanded significantly when trained using the LAION-5B dataset [203] of 5.85B image-text pairs. By training on such a large dataset, the model learns a wide variety of concepts and associated feature representations, allowing for generalization beyond the explicit training [185]. This leads to powerful zero-shot capabilities across tasks such as classification [39],[159] and object detection/segmentation [210],[259],

[101],[259]. When combined with a latent diffusion model (LDM), CLIP improves the zero-shot generation capacity for novel prompt inputs [186]. In this case, an LDM (e.g. Stable Diffusion) can generate images of subject matter that was not included in the training data. In our work, we seek to understand how to modify the CLIP embedding space to function as a latent domain for novel image generation and concept merging, specifically for textures.

Latent Space Exploration Latent spaces of generative models have long been an important mechanism used to exert control over image output. In generative adversarial networks (GAN) [69] understanding the structure of the latent manifold, where factors of a latent input vector z are mapped via the generator to the features of an output image, has led to powerful techniques for image control such as direction finding [212, 68, 96] and latent interpolation [1, 167, 114]. Since the latent space distribution used by GANs is predefined, interpolation between vectors usually results in smooth transitions. As with GANs, VAEs [121] sample latent vectors, although the distribution is modelled as a Gaussian where the encoder is trained under a KL divergence constraint, leading to semantically meaningful interpolations.

This has led to much work in understanding the organization of this space for improved disentanglement for image editing and exploration [209, 23]. Unlike VAE and GAN latent space, in LDMs, the points in the latent space do not correspond to the output samples. Instead, the latent space exists in the high-dimensional, intermediate noisy states learned during training. Here, individual points and steps map to a partially denoised component of the output as it progresses through the denoising process. This space between pure noise and fully denoised output is continuous and can produce high-quality samples along the denoising trajectory. Some LDMs like Stable Diffusion [196] use CLIP embeddings to guide the denoising process by conditioning the process to align the generated image with the semantic meaning contained in the prompt. By concatenating Gaussian noises at each diffusion step, Wu et al. [256] create a higher-dimensional latent code than the standard approach. By manipulating the latent codes, various aspects of image generation can be explored.

Optimization As part of our pipeline, we must obtain a text embedding from an image input. This is similar to the long-standing image captioning task [88], which typically requires training a dedicated model. More recently, optimization methods have emerged that can improve prompt

quality. Qin et al. [183] optimize embedding combinations using gradient descent using their They show an improvement in knowledge extraction from LMs, but their method requires access to the internal probability distribution from the pre-trained language model. Pryzant et al. [182] showed that an optimization based on gradient descent using text-based iterations results in improved prompt effectiveness [182]. Their method relies on a reasonably high-quality initial prompt along with access to an LLM API, which is not always practical. Recently, some authors have begun working directly with CLIP embeddings [140].

CLIP Embedding Space Most similar to the prompt extraction component of our method is the work of Wen et al. [253]. The authors use a gradient-based method to optimize text embeddings based on cosine similarity using kNN search in the embedding space. They showed that optimizing directly in the CLIP embedding space can produce improved human-readable prompts. Later Ding et al. [48] demonstrated that CLIP embeddings can be used to derive prompts from images through a similar projection mechanism. Ding et al. extract image embeddings using a projection technique that fine-tunes the cross-attention layers within the Stable Diffusion model (SD) [179] U-Net. Guerrero-Viu et al. introduce a diffusion-based approach for continuous texture editing within a localized “CLIP-space”, allowing precise control over specific texture attributes (e.g., weathering or scale) while preserving identity. Their use of CLIP embeddings is constrained to narrowly defined regions in CLIP-space, derived from contrastive prompt clusters like “new wood” vs. “aged wood,” enabling targeted attribute adjustments rather than broad concept blending. In contrast, our work explores broader interpolation across the CLIP embedding manifold, blending distinct visual textures to create novel combinations, and using a more expansive subset of CLIP-space beyond two contrasting prompts. Similarly, Deckers et al. [43] introduces methods for direct prompt embedding interpolation, in which they demonstrate an alternative to traditional prompt engineering. They develop three techniques: metric-based optimization, iterative human feedback, and seed-invariant prompt embedding manipulation. These methods offer users control over LDM output. Wen et al. [140] train a text decoder to convert CLIP’s decoder into a sentence generator. The authors use a projection mechanism to map visual embeddings into the CLIP text embedding space to improve captioning performance. Their work demonstrates the potential of projection techniques when using CLIP embeddings for visual-language tasks.

5.3 Methodology

Our method takes two textures containing different features, patterns, and concepts and creates an output that intends to capture elements from both in a cohesive and coherent output texture. This is accomplished by a processing pipeline containing three stages: (a), generating of embeddings from input images that both accurately represent the input and also align with the CLIP embedding space; (b), constructing a graph that captures visual and semantic relationships between the generated embeddings; finally (c), an interpolation method that optimizes a path through the graph between two endpoint textures, guiding the interpolation to an output texture that preserves meaningful visual features and patterns from both input textures.

5.3.1 Overview

Our method is divided into three core components: embedding generation, graph creation, and interpolation (see Figure 5.1). The embedding generation stage takes the input images from the source dataset (Describable Textures Dataset (DTD) [38]) and generates embeddings. The generated embeddings undergo an optimization process to better align the features of the input within the CLIP embedding space that implicitly share properties with the CLIP model. These properties place embeddings with similar features closer to each other in a high-dimensional space. The graph creation stage organizes the generated embeddings by connecting them according to the similarity of their features in a reduced dimensional space. The dimensionality reduction improves computational efficiency and interpretability. The final interpolation stage uses the graph to perform a spatial interpolation between two user-selected images taken from the embedding space. The interpolation process optimizes a path between the selected images that balances smooth transitions and semantic consistency to blend the features, patterns, and concepts they contain.

Generate embeddings

The first stage of our process produces CLIP-aligned embeddings from images that we sequentially add to the latent representation. The process consists of the following steps:

- **Convert input image into a CLIP image embedding.** We use CLIP’s image encoder to

generate an embedding representation of the input image. This embedding contains a set of feature vectors that describe the visual and spatial features that we want to represent in CLIP-space.

- **Optimize the embeddings.** We need to convert the image features into an embedding that has better alignment with text features in CLIP’s latent space. Starting with a prompt embedding P that is initialized with random token feature vectors, we iteratively refine P by computing the loss with respect to the target feature embeddings from the input image. The optimization has two components:
 - **Projection.** During the optimization, we project P onto the feature vectors representing tokens from CLIP’s vocabulary to ensure that the P contains coherent concepts understood by the model.
 - **Gradient Descent.** At each iteration, we update P to improve its alignment in CLIP-space. This involves two steps. First, we perform the projection step to refine P by aligning its feature vectors with the nearest matches in CLIP’s token embeddings space. Next, using Wen et al.’s [253] optimization method, we compute the loss as the negative cosine similarity between P (after the projection) and the target feature embeddings extracted from the input image that describe its spatial and semantic features. This loss quantifies the alignment between the optimized embedding and the target features. We then iteratively update P using gradient descent, minimizing the loss and improving the alignment of P with the target features in CLIP’s shared embedding space. P is updated in place.

We perform these steps for each image in the dataset, so that together they result in a high-dimensional representation of the source dataset. The optimization process ensures that P captures the semantic and spatial characteristics of the input image while remaining interpretable in the CLIP embedding space. Without this optimization, the embedding would not be a good representation of the features in the source image.

Graph Construction

In the graph construction process, we use the embeddings extracted during the first stage to create a graph-based representation. The graph organizes the embeddings into a connected framework in

three dimensional space that enables interpolation by directly relating the embeddings. The graph defines paths that represent relationships between embeddings. Organizing the embeddings as a graph helps to define smooth paths that avoid inconsistent transitions between embeddings. In the graph, nodes represent embeddings and edges represent the weighted similarity relationships between embeddings. The construction of the graph involves the following steps:

- **Project high-dimensional embeddings into 3D space.** We project the high-dimensional embeddings into 3D space to improve computational efficiency, while maintaining relative proximity relationships from the original space. This also improves the interpretability of the spatial relationships we want to exploit in the interpolation.
- **Connect the Embeddings.** We create edges between embeddings by a set of nearest neighbours for each node using cosine distance.
- **Expand the neighbourhood.** In order to ensure that the interpolation path does not only have access to direct neighbours of each node, we expand the neighbourhood to capture longer range relationships. This means that we are not constrained by very local transitions that might exclude more meaningful paths. It also allows additional flexibility for the interpolation to find smoother paths via more natural trajectories.

The result of graph construction is an embedding space projected into 3D space, that effectively represents the structure of CLIP-space and supports direct interpolation between embeddings. By capturing local relationships through the graph structure, the approach aims to enable smooth and coherent interpolation paths that reflect the geometry of the embedding space.

Interpolation

The final stage of our method is the spatial interpolation. We use this process to interpolate between two embeddings. The goal of this interpolation is to generate intermediate embeddings between two input textures using the graph-based embedding space described above. The spatial interpolation is based on the following steps:

- **Endpoint selection.** A user selects two textures from the dataset. We refer to these as endpoints since they mark the ends of the interpolation path. The interpolation aims to smoothly blend

these concepts while preserving their unique characteristics.

- **Find initial path.** The selected endpoints are located in the space and a path is estimated between them using spherical linear interpolation (slerp). Slerp preserves the geometric properties of the hyperspherical embedding space in the interpolated points more effectively than linear interpolation
- **Extract subgraph.** We map intermediate points on the initial interpolation path from the previous step to the nearest respective nodes. Using these nodes, we extract a subgraph that contains all the direct neighbours from each node. To expand the coverage and not exclude long-range relationships from the subgraph, we add additional neighbours through a hierarchical process. Using the subgraph improves computational efficiency. Future steps operate on the subgraph directly.
- **Optimization.** Using the subgraph, we refine the initial interpolation path to balance cosine similarity with smooth transitions while avoiding low-density or semantically inconsistent regions of the embedding space. The optimization minimizes a cost function designed to capture these properties. The result of the optimized spatial interpolation is a progressive blending along a sequence of embeddings, with the most balanced contribution from each end point being near the midpoint. We use these points to compute the final interpolation in high-dimensional space.
- **Final Interpolation.** Finally, we perform a dense interpolation between the embeddings along the optimized path. Using slerp, we generate points between each pair of optimized points resulting in the final interpolated embeddings between user selected endpoints.

Once the final interpolation path is defined, the final stage involves the generation of the output image using a latent diffusion model (we use the open source Stable Diffusion [196] model). To do this, we must decode the embedding using the projection approach we used when creating the embedding representation described above. As before, this associates the embedding with valid tokens which we then pass to the latent diffusion generator in the form of a prompt. The decoded prompts are machine-optimized token sequences that are not intended to be user-facing in our approach. They are used internally to communicate between the interpolation process and the

LDM.

Our approach combines embedding generation, graph construction, and spatial interpolation to create a framework for blending two textures while preserving visual characteristics and patterns from each. In the remainder of this section, we will provide more detail into the workings of each component.

5.3.2 Construction of the Graph-Based Embedding Representation

In the context of this work, a latent representation is a spatial arrangement of CLIP embeddings, where the distance between embeddings is defined by cosine similarity. This is required since the CLIP shared embedding space used by generative models like Stable Diffusion [196] is only defined implicitly within the CLIP model’s parameters. As such, the “CLIP-space” of the model cannot be directly interacted with. However, it is common to refer to single embeddings as localized representations of the larger CLIP-space, which are sometimes referred to as “CLIP-space”. To make a representation of CLIP-space outside of the model’s parameter space and beyond single embeddings that is suitable for exploration, we compute a spatial dataset of embeddings as a graph using the DTD. This dataset of embeddings serves as an explicit, external representation of the “CLIP-space” that can be used to understand relationships between embeddings and perform actions like interpolation. The graph is a projection of the original embeddings into a reduced 3D space to facilitate spatial actions but retains a mapping to the original embeddings.

The construction of the embedding representation is done in two stages. In the first stage we must convert each image in the source dataset into high-dimensional embeddings that approximate a CLIP-space manifold. In the second stage we use the generated embeddings to create a graph-based representation that captures the semantic and geometric properties of the space in a structure that enables interpolation.

Convert Images to Embeddings

To create the latent representation, we must convert source texture images to high-dimensional embeddings. In the first step of this process we use the CLIP image encoder to produce an

embedding T . T is a real-valued matrix containing 16 feature vectors that describe spatial and semantic information about the source image. We will use this embedding to learn a new embedding P that aligns with CLIP’s embedding space.

Since the initial conversion of inputs to CLIP embeddings is a lossy process and we want the resulting embedding to match the input as closely as possible, we must optimize the feature embeddings to achieve this. Furthermore, since the semantic relationships learned by CLIP are critical to this task, we optimize based on cosine similarity.

Optimize Embeddings

The purpose of the optimization is to position the features of T within the CLIP embedding space by learning a new CLIP embedding P .

To perform the optimization, we iteratively refine the embedding P using gradient descent following the approach proposed by Wen et al. [253]. We initialize P using random token feature vectors from the CLIP vocabulary, matching the shape of T . The vectors contain the learned semantic features of the tokens in the shared image-text embedding space. During each iteration, we project the P embedding onto the CLIP token embedding space using our projection approach within Wen et al.’s optimization loop, then calculate the loss between P and the target feature embeddings.

In the projection process, we use the text embedding layer extracted from the pre-trained CLIP model that contains learned embeddings for the tokens in the CLIP vocabulary. These embeddings serve as centroids that conceptually represent the semantic meanings of tokens. Using the centroids as reference points, we align the evolving embeddings within the semantic structure of CLIP-space.

Formally, using embedding P , we compute the Euclidean distance between each feature vector P_i in P and centroids C_j as $d_{ij} = \|P_i - C_j\|_2$, where P_i is the i -th feature vector and C_j is the associated j -th centroid.

The centroids are initialized from the weights of CLIP’s token embedding layer, which is the learned embedding matrix of size $V \times D$, where V represents the size of the CLIP vocabulary and D represents the embedding dimension. Each row of this matrix contains the embedding of

a specific token in the vocabulary. We use these rows to initialize each centroid. The goal is to compute how strongly each embedding’s feature vectors are associated with each centroid. To do this we compute assignment scores a_{ij} using the softmax function:

$$a_{ij} = \frac{\exp(-d_{ij}/\tau)}{\sum_k \exp(-d_{ik}/\tau)}, \quad (5.1)$$

where τ is the regularization parameter. The score represents the probability that the i -th embedding is assigned to the j -th centroid. By adjusting τ , we can control the strength of the association between a centroid and an embedding as a weighted assignment. A large τ will have a smoothing effect, which means that the embedding assignment scores will be spread more evenly across multiple centroids. The concept of weighted assignment serves as an analogy for how we think about semantic texture concepts, where an embedding may belong to multiple centroids such as rough and grainy. Smoothing will allow the assignment to span these centroids with similar scores, versus a hard assignment to one or the other (using a very low τ).

Next, we update the centroids C_j using a weighted average based on the embeddings and their assignment scores:

$$C'_j = \frac{\sum_i a_{ij} \mathbf{P}_i}{\sum_i a_{ij}} \quad (5.2)$$

Where a_{ij} is the assignment score between feature vector \mathbf{P}_i as weights and centroid C_j . This adjustment moves C_j to a new position based on the embedding assignments as a normalized weighted average. To adjust the centroids gradually and prevent abrupt changes, we apply a learning rate α , resulting in the updated centroids: $C_j^{\text{new}} = (1 - \alpha)C_j + \alpha C'_j$.

The adjustment process involves iteratively computing the assignment scores a_{ij} and updating the centroids C_j for n iterations. After the adjustment process is complete, we determine the closest centroid for each embedding by calculating $d_{ij} = \|\mathbf{P}_i - C_j\|_2$ and finding the index $j^* = \arg \min_j d_{ij}$. The projected embeddings are then assigned to the closest centroid:

$$\mathbf{P}_i^{\text{proj}} = \mathbf{C}_{j^*}.$$

After the projection step is completed, the cosine similarity between the projected embeddings and the target features is computed for use during gradient descent. The objective during gradient updates is to maximize this similarity. To do this, we minimize a loss function defined as the negative mean cosine similarity using Wen et al.’s method:

$$\mathcal{L} = -\text{cosim}(\mathbf{P}, \mathbf{T}) \tag{5.3}$$

The gradient of the loss with respect to \mathbf{P} is computed as:

$$\nabla_{\mathbf{P}} \mathcal{L} = \frac{\partial \mathcal{L}}{\partial \mathbf{P}} \tag{5.4}$$

We update the embeddings \mathbf{P} using the AdamW optimizer with learning rate η and weight decay. The accuracy of the embedding improves with the number of iterations (n_{iter}), but with diminishing returns. Since we use this process to generate embeddings for a large dataset (5651 images in the DTD), we limit n_{iter} to 800 to balance computational efficiency and performance. When applying the method to a single image, a higher value of n_{iter} , closer to 2000-4000, is more appropriate to achieve better representation quality (see Figure 5.2).

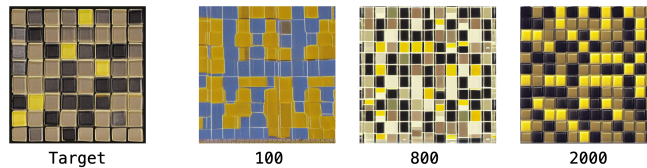


Figure 5.2: Results by Number of Iterations This figure shows an example of the difference between different iteration settings for the optimization. We show the target image that we want to match with the optimization, followed by the results of the optimization according to the value for n_{iter} . We see a progressive improvement from 100 to 2000 iterations. The cosine similarity scores for these values were 0.391, 0.457, and 0.479 respectively.

Using this approach we have a process that, given an input texture, generates an optimized

embedding that closely represents the visual features and patterns of the target image, while aligning within the CLIP embedding space. With this optimized embedding we can construct a graph-based latent representation that enables interpolation between embeddings.

Graph-Based Representation

The graph-based representation serves as a structural framework to organize the optimized embeddings generated in the earlier stages of our method (see Section 5.3.2). The graph is constructed based on the spatial similarity between our computed embeddings, providing a spatial representation of high-dimensional CLIP-space.

Under the manifold hypothesis, it is reasonable to expect that the most important dimensions lie on or near a lower-dimensional manifold [15, 180, 117]. In the graph, each node is an embedding and the edges represent proximity according to cosine similarity. The goal of our graph construction approach is to accurately connect similar embeddings while avoiding connection to dissimilar embeddings. We analyze the geometry of the embedding space to inform graph structure and interpolation. The embeddings are normalized to unit length, with a mean norm of 1.0 and negligible variation, constraining them to a high-dimensional hypersphere that supports the spherical data manifold.

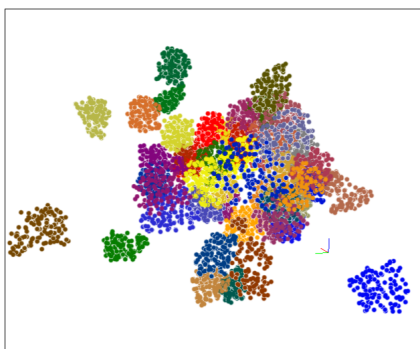


Figure 5.3: **3D Visualization.** Here we show a 3D representation of the dataset of embeddings based on UMAP. This is a spatial representation of the nodes that will form the graph.

The graph is constructed by first projecting each of the high-dimensional embeddings into a lower-dimensional (3D) space using UMAP (Uniform Manifold Approximation and Projection)

[155] while retaining a mapping to the original embeddings. We compute an initial set of k nearest neighbours where $k = \max(5, \lfloor \log(n) \rfloor)$ and n is the number of embeddings, according to cosine distance. Edges are created between nodes and neighbours and assigned weighting (weight = $\frac{1}{1+\text{distance}}$) such that the value is always positive (see Figure 5.4).

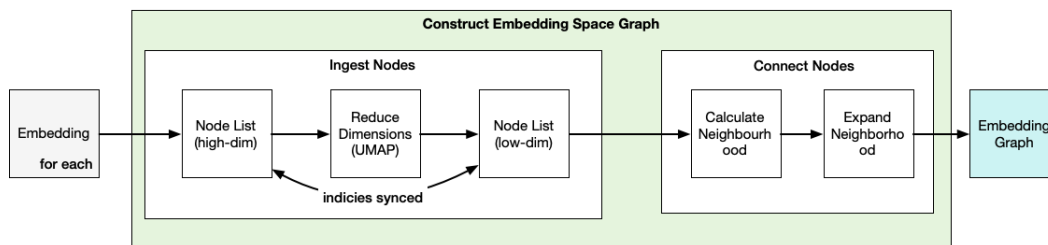


Figure 5.4: **Convert Images to Embeddings** This diagram summarizes the construction of the embedding space graph using the high-dimensional embeddings optimized from the DTD.

We found the initial k neighbours (where k is the base number of nearest neighbours) to be insufficient to represent potentially influential embeddings due to the limited capture of longer-range relationships and the potential for disconnected clusters, which can result in the graph under-representing broader structural patterns in the data. To counter this, we adopt a hierarchical approach that incrementally adds new connections by expanding the neighbourhood size at successive levels. Specifically, at each hierarchical level ℓ (default $L = 3$), where ℓ is the number of levels, the neighbourhood is expanded to $k_\ell = 2^\ell$. For each node, the k_ℓ nearest neighbours are computed based on cosine distance. Edges are then added to nodes in the farther half of these neighbours to expand connectivity without excessive density and to preempt overlap, with weights assigned as above. The default values for neighbourhood sizes are assigned heuristically based on the need to find a balance between capturing a larger neighbourhood and managing computational complexity.

Algorithm 1 Embedding Space Graph Construction: Part 1

Require: High-dimensional embeddings \mathcal{E} , Base number of neighbours $k_{\text{base}} = \max(5, \lfloor \log(|V|) \rfloor)$, Hierarchical levels L

Ensure: Graph $G = (V, E)$ where V represents embeddings and E represents weighted edges

1: **Project Embeddings into 3D Space:**

2: Reduce the dimensionality of \mathcal{E} to 3 dimensions.

3: **Initialize Graph:**

4: Set $V \leftarrow \{v_i \mid e_i \in \mathcal{E}\}$, where each node v_i corresponds to an embedding e_i .

5: Attach node attributes (e.g., paths, images, and prompts) to each v_i .

6: Set $E \leftarrow \emptyset$ (empty edge set).

7: **Compute Pairwise Neighbourhoods:**

8: **for** each embedding $e_i \in \mathcal{E}$ **do**

9: Compute pairwise distances $d(e_i, e_j)$ to all other embeddings using cosine distance.

10: Select k_{base} nearest neighbours $\mathcal{N}_{\text{base}}(e_i) = \{e_{i_1}, e_{i_2}, \dots, e_{i_{k_{\text{base}}}}\}$ based on ascending $d(e_i, e_j)$.

11: **end for**

12: **Add Neighbourhood Edges:**

13: **for** each node $v_i \in V$ **do**

14: **for** each neighbour $e_j \in \mathcal{N}_{\text{base}}(e_i)$ **do**

15: Compute edge weight $w(v_i, v_j) = \frac{1}{1+d(e_i, e_j)}$.

16: Add edge $(v_i, v_j, w(v_i, v_j))$ to E .

17: **end for**

18: **end for**

Algorithm 2 Embedding Space Graph Construction: Part 2

19: **Expand Connectivity with Hierarchical Neighbourhoods:**

20: **for** each level ℓ from 1 to L **do**

21: Set neighbourhood size $k_\ell = 2^\ell$.

22: **for** each node $v_i \in V$ **do**

23: Compute k_ℓ nearest neighbours $\mathcal{N}_\ell(e_i)$ using $d(e_i, e_j)$.

24: **for** each neighbour $e_j \in \mathcal{N}_\ell(e_i) \setminus \mathcal{N}_{\text{base}}(e_i)$ **do**

25: Add edges to the farther half of neighbours:

$$w(v_i, v_j) = \frac{1}{1+d(e_i, e_j)}.$$

26: Add edge $(v_i, v_j, w(v_i, v_j))$ to E .

27: **end for**

28: **end for**

29: **end for**

30: **Ensure Graph Connectivity:**

31: If G contains isolated components, add edges between nodes in disconnected components using shortest pairwise distances.

32: **return** G

5.3.3 Interpolation via Spatial Optimization

The proposed interpolation method addresses the problem of generating images that lack a clear semantic description and the challenge of blending texture concepts. This is achieved by optimizing interpolation paths between texture embeddings, over a large embedding space (see Section 5.3.2) to ensure that transitions maintain semantic coherence and adhere to the manifold geometry. This differs from existing work that seeks to interpolate between hyperlocal embeddings to make subtle changes to prompt text, as in Deckers et al. [43] and Guerrero-Viu et al. [72]. Our method identifies an updated path through the embeddings space between user-selected endpoints, where endpoints are embeddings taken from the graph and visualized as images for their selection. The user selects these endpoints so as to direct the interpolation toward the desired output. When operating on the graph, we use the low-dimensional representation for more efficient

computations through the interpolation process.

A preliminary analysis of naïve paths through the low-dimensional embedding space computed using slerp found that the approximate path contains sub-optimal nodes which we refer to as noise. This noise manifests as seemingly unrelated concepts present in the output images at the end of the generation pipeline. Our initial assumption was that similarity between embeddings would correlate with a path that maintains gradual transitions and consistent visual features; what we define as a smooth trajectory in the embedding space, but we find that similarity alone does not guarantee a smooth path. We found that the trajectory within CLIP-space has a substantial effect on the interpolations and thus our optimization-based interpolation approach is designed to find intermediate nodes along the interpolation path that balance semantic proximity with favourable trajectories.

The interpolation begins as a first guess at the interpolation path using spherical linear interpolation (slerp) [215] to generate n (we use $n = 5$) anchor points along the geodesic of the full embedding space to connect the endpoints. The slerp-generated points are mapped to the nearest embedding nodes in the graph using cosine distance. Using this initial path, we extract a *subgraph* by adding all the direct neighbours from the main graph and connect them in the subgraph, where the direct neighbours reflect the hierarchical node additions in the full graph (see Algorithm 3). By operating on the subgraph during the interpolation optimization, we reduce the required computation (see Algorithm 3 and Figure 5.5).

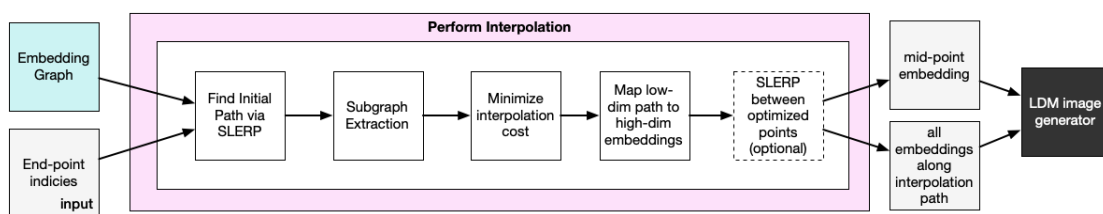


Figure 5.5: Interpolation Process The interpolation process generates an optimized path between two user selected images from the dataset called endpoints. Using the embedding graph and the endpoints an initial path is estimated using slerp. A subgraph is extracted around the path to be used in the optimization of the cost function. The optimized low-dimensional points on the path are mapped back to the nearest non-duplicated embedding in the space. Optionally, the path can be made more dense by interpolating additional intermediate points to smooth transitions. Stable diffusion is used to generate the final output images based on the embeddings.

Optimization

The optimization process aims to construct an interpolation path that balances several objectives: smooth transitions between embeddings, traversal through dense regions and semantic consistency along the path. These objectives are encoded into the cost function which consists of three terms: smoothness, density, and consistency.

The output of the optimization step is a sequence of points in the high-dimensional embeddings space. This sequence include the user selected endpoints as well as the optimized intermediate points. The intermediate points are initially optimized

A balance between proximity via cosine similarity and trajectory smoothness is achieved through the optimization of the path along the hypersphere that favours soft curvature through dense areas, while avoiding regions of semantic inconsistency. We design a cost function to address the challenges of traversing this graph and perform the optimization using the L-BFGS-B algorithm [162] to compute the gradients of the cost with respect to each point on the path as $\hat{P} = \arg \min_P \mathcal{L}(P)$. In Equation (5.5), we show the components of this loss function.

$$\mathcal{L} = \text{Tr}(P^\top LP) + \lambda \sum_{i=1}^{n-1} \frac{\|P_{i+1} - P_i\|}{\rho(i)} + \alpha \sum_{i=1}^{n-1} \mathcal{M}(P_i) \quad (5.5)$$

The optimization proceeds over the nodes and edges of the subgraph as indicated by the points P . The cost is computed over three terms that encode the qualities we want the traversal path to have. We discuss each term (smoothness, density and consistency) individually as follows:

Smoothness

To ensure smooth transitions between textures during interpolation, the method relies on the graph Laplacian [13, 288], which encodes the local geometry of the embedding space. The Laplacian matrix, L , is derived from the subgraph’s adjacency matrix, where each entry reflects the connectivity between neighbouring embeddings. The Laplacian acts as a regularizer during the interpolation process, penalizing paths that deviate from smooth, manifold-aligned transitions defined by the graph structure. Mathematically, the Laplacian is used to enforce smoothness by

minimizing the following term in the cost function:

$$\text{Smoothness Term} = \text{Tr}(P^\top LP) \quad (5.6)$$

where P is the matrix of embeddings where each row represents a point along the interpolation path and L is the constant Laplacian of the subgraph with entries on the order of approximately 500. This localizes the scope of the Laplacian for a more manageable computation. The smoothness term ensures that the interpolation path follows the underlying structure of the manifold, discouraging abrupt transitions between nodes and encouraging smooth transitions across the texture space. We compute the trace as a measure of smoothness along the interpolation path. By minimizing the trace term, the method ensures that the intermediate steps respect the local relationships between embeddings, by penalizing deviations.

Density

The density term in the cost function is designed to guide the interpolation path through regions of the embedding space that are densely populated with relevant texture embeddings, avoiding areas where data points are sparse and less reliable. By incorporating local density information, the method ensures that the interpolation path favours regions that contain well-defined related embeddings, which are more likely to correspond to meaningful texture concepts.

To compute the density, the local neighbourhood around each embedding is analyzed using a nearest neighbour search. For each node i , the local density $\rho(i)$ is estimated as the inverse of the mean cosine distance to its k nearest neighbours. The local density at node i is defined as:

$$\rho(i) = \frac{1}{\frac{1}{k} \sum_{j=1}^k d(i, j) + \epsilon} \quad (5.7)$$

where $d(i, j)$ represents the distance between node i and its j -th nearest neighbour, and ϵ is a small constant to ensure numerical stability. This density estimate helps in distinguishing between dense, semantically rich regions of the embedding space and sparse, less meaningful regions.

The density term in the cost function is then defined as:

$$\text{Density Term} = \sum_{i=1}^{n-1} \frac{\|P_{i+1} - P_i\|}{\rho(i)} \quad (5.8)$$

where P_i and P_{i+1} represent consecutive points along the interpolation path. This term penalizes paths that pass through low-density areas by scaling the distance between successive points by the local density $\rho(i)$. Minimizing this term encourages the path to stay within regions of higher density, thus ensuring that the interpolation passes through areas where the texture embeddings are more likely to be semantically meaningful. As a result, the interpolation path remains faithful to the underlying structure of the embedding space, leading to a more coherent blending of texture concepts.

Consistency

The third term in the cost function is a consistency term that penalizes the interpolation path for traversing regions of the embedding space where the texture representations are less well-defined or semantically weak. This term ensures that the path stays within semantically coherent regions, avoiding areas where embeddings may not strongly correspond to meaningful texture concepts.

This term evaluates the deviation of intermediate points from core regions of the embedding space, penalizing paths that pass through semantically irrelevant areas. It uses an attention-weighted penalty function, where the penalty increases if the intermediate points diverge from well-defined texture embeddings. The term is defined as:

$$\text{Consistency Term} = \sum_{i=1}^{n-1} \mathcal{M}(P_i) \quad (5.9)$$

where $\mathcal{M}(P_i)$ measures the semantic distance between the path point P_i and the closest embeddings in the space, weighted by their relevance:

$$\mathcal{M}(P_i) = \sum_{j=1}^m (1 - \max(\text{softmax}(\text{similarity}(\mathbf{P}_i, \mathbf{E}_j)))) d(\mathbf{P}_i, \mathbf{E}_j) \quad (5.10)$$

where $\text{similarity}(\mathbf{P}_i, \mathbf{E}_j)$ is a vector of similarity scores between path point \mathbf{P}_i and all embeddings \mathbf{E}_j in the subgraph to given embeddings with higher similarity more weighting. $d(P_i, E_j)$ is the Euclidean distance between the path point and the embeddings, and the softmax over the cosine similarity ensures stronger penalties for transitions through less relevant areas. By minimizing this term, the method keeps the interpolation path aligned with semantically meaningful regions.

Final Interpolation and Image Output

To generate interpolated transitions between high-dimensional data points, we use `slerp` to compute the complete set of points along the interpolation path. Given the set of high-dimensional anchor points received from the optimization step $\mathcal{V} = \mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n$, where each $\mathbf{v}_i \in \mathbb{R}^{16 \times 1024}$, we interpolate between consecutive embeddings \mathbf{v}_i and \mathbf{v}_{i+1} to densify the path. The number of interpolation steps n_i determines the total number of interpolated points along the path. Specifically, $(n_i - 1)$ interpolated embeddings are generated between each pair of embeddings, leading to a total of $(n_i - 1) \times (n - 1)$ interpolated embeddings. Then for each step t , where $t = 1, 2, \dots, n_i - 1$, we compute an interpolation fraction $f_t = t/n_i$. This fraction determines the relative position along the geodesic between two consecutive embeddings.

For example, if we have 5 anchor points along the path and $n_i = 5$, then between v_1 and v_2 we compute 4 equidistant intermediate points ($t = \frac{1}{5}, \frac{2}{5}, \frac{3}{5}, \frac{4}{5}$) and add them to the final path. This process is repeated for each pair of consecutive anchor points, resulting in a densely interpolated path.

The output of the above process is a sequence of embeddings along the interpolation path. Given these high-dimensional embeddings, in order to visualize the interpolated embeddings, we use our projection method (see Section 5.3.2) to decode the embedding into a text prompt. We then use the LDM model Stable Diffusion to generate images.

Algorithm 3 Spatial Interpolation for High-Dimensional Embeddings

Require: Graph $G = (V, E)$, High-dimensional embeddings \mathcal{E} , beginning embedding e_1 , ending embedding e_2 , Number of interpolation steps N

Ensure: Output embeddings $\{e_1, \dots, e_N\}$

1: **Step 1: Map Embeddings to Graph Nodes**

2: Identify source node $v_s \in V$ and target node $v_t \in V$ corresponding to e_1 and e_2 by minimizing pairwise cosine distance.

3: **Step 2: Subgraph Extraction**

4: Extract subgraph $G_{\text{sub}} = (V_{\text{sub}}, E_{\text{sub}})$ containing the shortest path $\mathcal{P}_{v_s \rightarrow v_t}$ and neighbouring nodes.

5: Compute the graph Laplacian \mathcal{L} for G_{sub} .

6: **Step 3: Dimensionality Reduction and Path Initialization**

7: Reduce embeddings $\mathcal{E}_{\text{sub}} \rightarrow \mathcal{E}_{\text{low}}$.

8: Perform slerp between $\mathcal{E}_{\text{low}}(v_s)$ and $\mathcal{E}_{\text{low}}(v_t)$ to initialize path $\mathcal{P}_{\text{init}}$.

9: **Step 4: Path Optimization**

10: Optimize the interpolation path in low-dimensional space starting from $\mathcal{P}_{\text{init}}$.

11: Obtain the optimized low-dimensional path \mathcal{P}_{opt} .

12: **Step 5: Map Back to High-Dimensional Space**

13: **for** each point $p_i \in \mathcal{P}_{\text{opt}}$ **do**

14: Find the closest high-dimensional embedding $e_i \in \mathcal{E}_{\text{sub}}$ using Euclidean distance.

15: Ensure diversity by avoiding duplicates in selected embeddings.

16: **end for**

17: **Step 6: High-Dimensional Slerp Refinement**

18: Perform slerp between consecutive high-dimensional embeddings $\{e_1, \dots, e_N\}$ to create a dense interpolation path.

19: **return** $\{e_1, \dots, e_N\}$

5.4 Experimental Setup

We design a series of experiments to evaluate our proposed spatial interpolation method and embedding projection technique with the following objectives:

- **Projection technique.** Compare our approach with a state-of-the-art baseline [253] to assess its ability to improve the output embedding’s representation of the input image.
- **Spatial interpolation.** Evaluate the quality of the texture blending, comparing spatial interpolation using our graph to linear interpolation to measure blending capacity.
- **Quantitative and qualitative analysis.** Use LPIPS and CLIP scores for quantitative assessment and an LLM-based analysis of visual quality and novelty of the outputs.
- **Ablation study.** Investigate the impact of the path optimization on interpolation quality.

This section will describe the dataset, baselines and metrics used in our experiments and a presentation of the results.

5.4.1 Datasets

To evaluate our proposed spatial interpolation method, we use the Describable Textures Dataset (DTD) [38], which consists of 5,640 texture images grouped into 47 categories. Each category (e.g., dotted, scaly, and bubbly) has approximately 120 images, covering a wide range of textures. We do not use the category labels directly in our approach, but they could serve to group the source images in a potential interface where endpoints are selected by the user. For each image in the DTD, we compute high-dimensional CLIP embeddings. We also generate images associated with each embedding, forming the dataset that will make up our spatial representation.

5.4.2 Baselines

To measure the performance of our spatial interpolation approach, we include a linear interpolation baseline, which interpolates directly between two endpoint embeddings without incorporating spatial relationships. This baseline offers a comparison to measure the influence of the spatial

information within the CLIP-space. We assess the performance of our embedding projection approach using the method of Wen et al. [253] as a baseline which is directly comparable.

5.4.3 Model Selection and Embedding Generation

For this study, we use the CLIP model ViT-H-14 [185] for embedding generation. The embeddings are optimized for 800 steps to achieve a high-quality latent representation, which enables more coherent interpolation between endpoints.

5.4.4 Evaluation Metrics

We evaluate the quality of interpolations using both quantitative and qualitative assessments:

1. LPIPS (Learned Perceptual Image Patch Similarity) [272]: Measures perceptual similarity between interpolated images and the original endpoints. LPIPS captures high-level feature coherence, aligning closely with human perception.
2. CLIP Score [185]: Measures semantic similarity between generated images and endpoint embeddings, providing insight into how well the interpolations retain the overall texture concept of the endpoints.
3. Qualitative Evaluation: We further assess the interpolation quality on dimensions of feature preservation, perceptual blending, and visual novelty. In additional experimental evaluation, we use a LLM agent to rate the interpolations on a 1 to 10 scale, focusing on the presence of endpoint features, smoothness in blending, and novelty in the generated textures.

For each method, and for each of twenty sample endpoints, we generate sixty images using random seeds. We then present the average for each metric across all endpoint pair experiments.

5.4.5 Implementation Details

All projection/optimization experiments were conducted using the AdamW optimizer [150] with a learning rate of 0.1. To ensure stability and consistency in embedding projections, embeddings are

normalized and initialized in UMAP space, and paths are optimized with L-BFGS-B to fine-tune the path along the embedding manifold.

For the LLM-based analysis, we used the ChatGPT 4o model [168].

5.5 Results and Discussion

In this section, we present the results of our assessment, where we focus on two main components: the prompt optimization method we use to align and generate embeddings for use in the graph-based latent representation and the the evaluation of our optimized spatial interpolation approach. For each component, we conduct both quantitative and qualitative analyses using LPIPS and CLIP score along with the set of qualitative dimensions defined above. Additionally, we propose and demonstrate an experimental hybrid analysis that uses an LLM-based agent to analyze the results of the spatial interpolation and linear interpolation and provide an detailed analysis. The LLM-based analysis can provide a deeper analysis of the image features and their presence of lack-there-of in the output. Caution is required when drawing any conclusions from the result, as the details of the deep feature based analysis are not traceable as the LLM’s process is a black box and internal influences that could bias the analysis cannot be quantified.

5.5.1 Prompt Optimization for Graph Construction

To evaluate the effectiveness of our projection optimization method for generating latent representations in CLIP space, we compared our approach with the projection method used by Wen et al.’s state-of-the-art method [253] on images from the DTD. For each pair of images that form endpoints for the interpolation in our experiment-set containing 40 endpoints sampled from across the dataset, we optimized the associated CLIP embedding to produce an optimized prompt. We repeated this process for 30 images per endpoint per run using random seeds for LDM generation, to establish a statistical basis for comparison for a total of 1200 images in the assessment. We tested Wen et al.’s method using CLIP models ViT-H-14 and ViT/B/32; ViT-H-14 provided the best output. We use Wen et al.’s optimization method during the comparison, only substituting the projection methods during the experiments.

Each optimized prompt was used to generate the evaluation images using LDM with random seeds with the goal of reproducing the input endpoint image as closely as possible. Each output image was evaluated by computing its CLIP score against the endpoint it was derived from. The CLIP score quantifies the alignment between an embedding and its target representation, providing a metric for assessing semantic coherence. Our method achieved higher CLIP scores on average for endpoint inputs, suggesting that it captures relevant visual features and concepts more effectively than Wen et al.’s approach.

Quantitative Results Table 5.1 summarizes the average CLIP scores for our method versus Wen et al.’s method for each endpoint. The results indicate that our optimization approach consistently outperforms the baseline, showing improvements in aligning the generated prompt embeddings with target endpoints. These quantitative results were supported by visual analysis, as our optimized prompts produced outputs closer to the target with fewer deviations.

Comparison	Average CLIP Score
Ours on Endpoint A	0.8202
Ours on Endpoint B	0.8280
Wen et al. on Endpoint A	0.7734
Wen et al. on Endpoint B	0.7528

Table 5.1: **Projection CLIP Score Comparison.** Here we compare the results of our projection approach against the projection method used in the state-of-the-art method presented by Wen et al.. Using the CLIP score metric. We find that our method results in higher average CLIP scores when tested against the evaluation dataset. We compute the average on the generated blended images for each endpoint pair.

Qualitative Analysis The qualitative results align with the quantitative findings, showing that prompts generated by our method yield outputs that are visually closer to the original textures (see figure 5.6). Also, the images generated with our optimized prompts maintained the core features of the target textures, as an improvement over the baseline. This indicates that our prompt optimization approach produces embeddings that are more effective at capturing difficult-to-describe visual concepts in CLIP space.

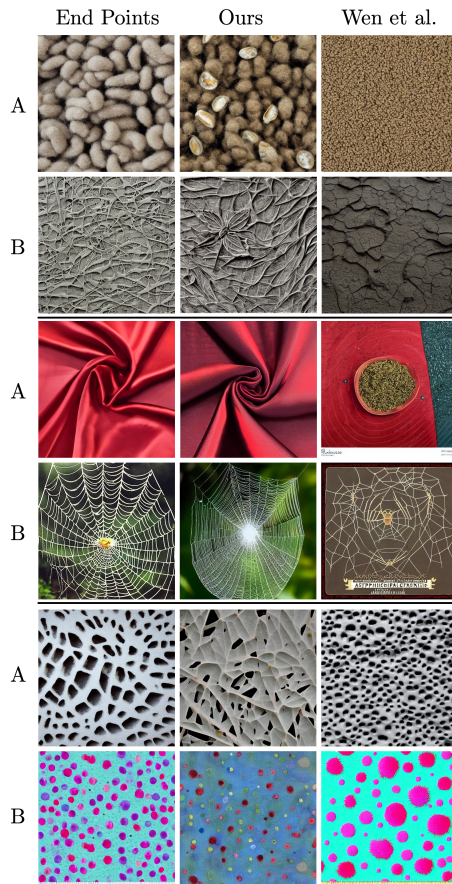


Figure 5.6: **Prompt Optimization Comparison.** We show a sample of optimization output comparing our result to that of Wen et al.. The purpose of this process is to optimize a single embedding so that the image generated from the result matches the endpoint image as closely as possible. We find the qualitative output supports the quantitative score as the output from our process seem to more closely reproduce the endpoint images.

5.5.2 Spatial Interpolation

The results demonstrate that interpolation over a manifold constructed from CLIP embeddings enables the blending of texture concepts into unique and novel combinations. This method enables exploration of embeddings in the CLIP space, uncovering intermediate representations that would be challenging to achieve with traditional prompt-based techniques. By making use of the pre-trained CLIP model, this approach combines two images in a contextually meaningful way,

aligning them with similar points in the embedding space and eliminating the need for complex prompt engineering.

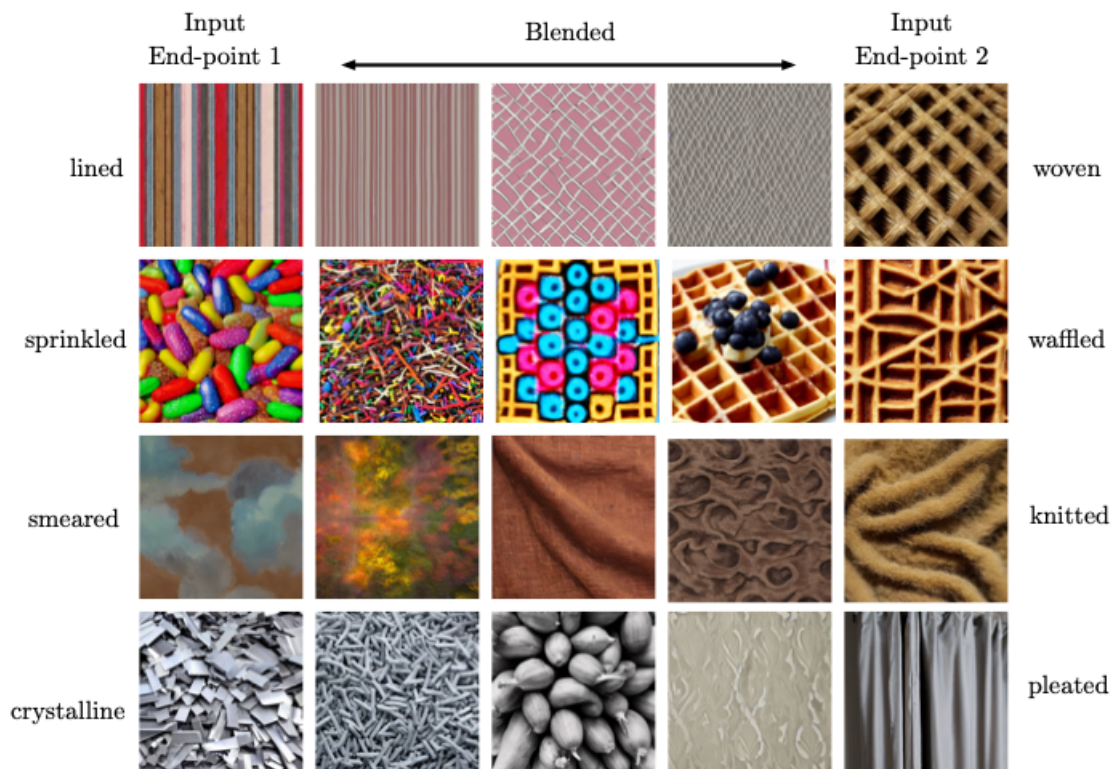


Figure 5.7: **Concept Blending.** We have four example interpolations where endpoint images are selected from different categories (e.g. linen and woven). We visualize the results of the interpolation using three samples from the interpolation path with the mid-point demonstrating the blended result. The blended results contain elements of each endpoint from at least one feature dimension (e.g. colour).

The blending task provides some insights about the structure of CLIP embeddings. Unlike a direct interpolation between two embeddings, the spatial interpolation smoothly and progressively traverses the manifold resulting in more coherent blending along the way (see Figure 5.7). Instead of two distinct concepts residing in isolation, there exists a gradient of related ideas, showing that CLIP’s embedding space does not store concepts as isolated points but seem to be interconnected. This gradient means that even subtle shifts in an embedding can yield meaningful changes. We can control the number of intermediate points that are interpolated between consecutive points

along the path. In Figure 5.8, we show all of the intermediate images along the gradient when using 16 optional interpolated points.

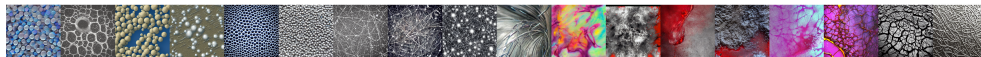


Figure 5.8: **Full Interpolation Between Endpoints.** Here we show the interpolation between “bubbly” and “crosshatched” endpoints with 16 interpolated points. The figure is meant to illustrate the high-level progression along the path. Please zoom in to see details.

Linear vs. Spatial Interpolations

We compare our spatial interpolation method with a baseline of linear interpolation. For linear interpolation, we compute intermediate embeddings by directly interpolating between two CLIP embeddings using a weighted average. We sample pairs of textures from the DTD dataset, setting each sample as an endpoint and apply both linear and spatial interpolation methods to examine the quality of the resulting blends. For each method, we generated 60 samples using different seeds to assess the consistency of the interpolation.

Qualitative Assessment

The spatial interpolation approach incorporates spatially relevant embeddings over the CLIP manifold to improve texture blending. We assess the output using the following qualitative metrics:

- **Feature preservation:** Assesses the retention of key visual characteristics from each endpoint to ensure distinctive traits such as texture and structure are consistently integrated in the blended output.
- **Perceptual blending:** Measures the smoothness of transitions between endpoints to evaluate if the interpolation achieves a coherent, visually continuous merge of both textures.
- **Visual novelty:** Evaluates the introduction of unique, contextually relevant features in the blend, to indicate the interpolation’s capacity for creative and original combinations.

Visually, we observe that the resulting textures incorporate features from both endpoints in coherent, novel arrangements. Features from the endpoints are preserved and blended in the

interpolation such that the output often contains the DTD category concepts as annotated in the input endpoints. Successful concept blending often emerges along various visual dimensions, including structure, spatial separation, and colour (see 5.9 A), which serve as pathways for effective combination. Additionally, more abstract dimensions, such as frequency of detail and geometry (see 5.9 B), may manifest based on the properties of the embeddings along the interpolation path. The results are often novel, and sometimes contain “creative” or unexpected arrangements that would be difficult to anticipate or describe, yet exhibit properties of the endpoints.

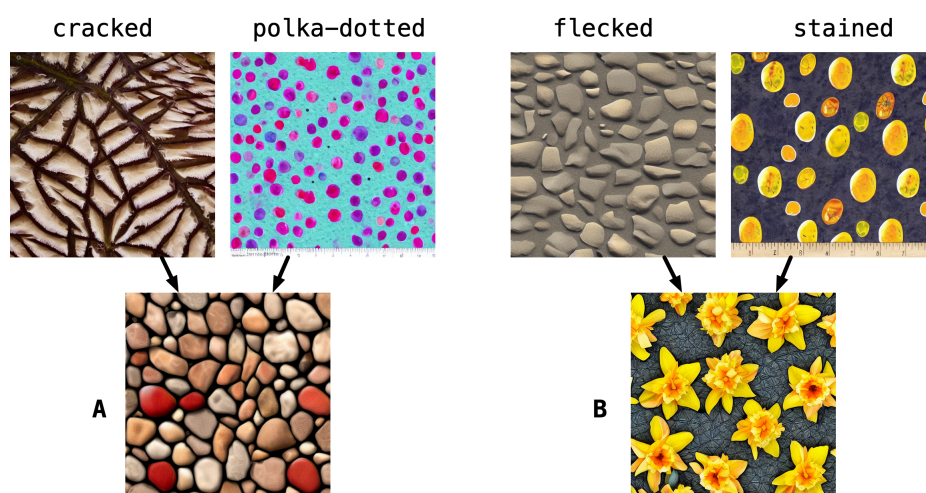


Figure 5.9: **Illustration of interpolation dimensions.** In Sample A, the structure from the cracked input is maintained, with a balance between the angular features of the cracked texture and the round features of the polka-dotted texture. Some colour blending is also observed, as seen in the red-coloured stones. In Sample B, the high-frequency background and colours from the stained sample are retained, combined with the angular geometry from the flecked sample. This results in an abstract blend, where the flowers incorporate the stained sample’s colours and the flecked sample’s angular details.

In comparison, the baseline method, while able to produce interesting images, often lacks the ability to blend features from the input endpoints consistently and in a manner that is easy to identify. The baseline results frequently contain noisy, strongly semantic, and unrelated content that degrades the output quality. Although the baseline method is much easier to implement and can produce good results for less complex endpoints, the overall blending capacity is limited (see Figure 5.10).

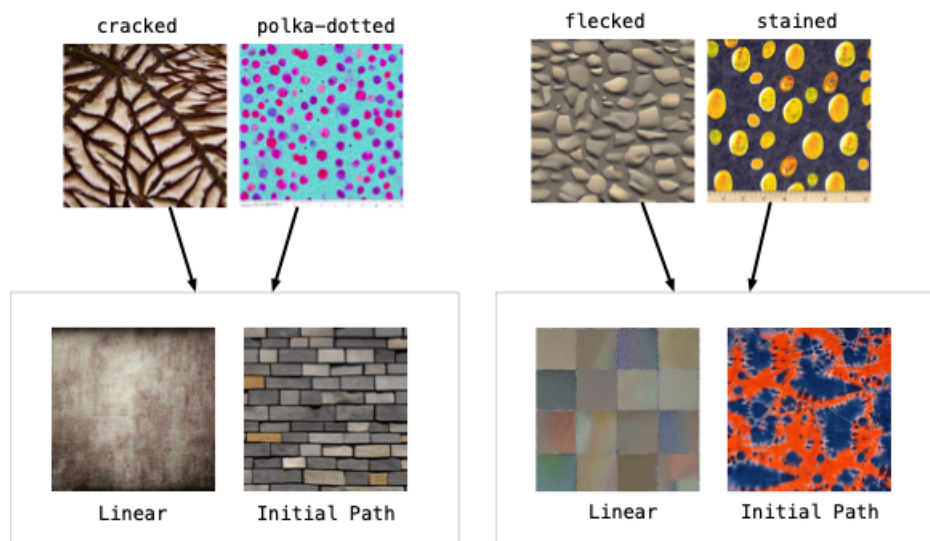


Figure 5.10: **Ablation of interpolation dimensions.** To contrast with Figure 5.9, we show the results of the linear interpolation and the interpolation from the initial path midpoint before optimization.

Quantitative Assessment

We use LPIPS [272] to quantify the perceptual similarity between interpolated outputs and both original endpoint images. LPIPS evaluates image similarity based on high-level feature embeddings, providing a metric that aligns more closely with human perception than pixel-wise comparisons. We also compute the CLIP score to measure semantic similarity between the interpolated outputs and the original endpoint images. Since we are working with textures, we use the CLIP score to assess how well the overall concept or texture style of the endpoints is retained in the interpolations (see Table 5.2).

	Average LPIPS ↓	Average CLIP Score ↑
Linear - Begin	0.7806	0.6644
Linear - End	0.7626	0.6791
Spatial - Begin	0.7453	0.6993
Spatial - End	0.7687	0.7028

Table 5.2: **Average LPIPS and CLIP Score.** This summarizes the data contained in the results across twenty endpoint pairs and sixty samples generated per pair. Lower values indicate a higher perceptual similarity. The LPIPS predicted score ranges between 0 and 1.

The results demonstrate that our spatial method outperformed the linear method in texture interpolation, both in perceptual and CLIP similarity to the original textures. Lower LPIPS scores for the spatial method indicate higher perceptual coherence, suggesting that the interpolated images more closely resemble the visual qualities of the endpoint textures. Additionally, the spatial approach achieved higher CLIP scores compared to the end points, indicating better preservation of the overall texture style or concept. While the magnitude of the difference is difficult to interpret, the quantitative results align with our qualitative findings.

Hybrid Evaluation

In an experiment designed to evaluate the blending of complex visual concepts, we used a large language model (LLM)-based agent as an objective evaluator of a set of generated images. This approach takes advantage of the LLM’s capacity detailed visual analysis, offering an alternative to traditional human evaluation. We asked the LLM agent to view a set of experimental results (see Figure 5.11) containing a sequence of output images and their category labels from both the spatial and linear interpolation methods. The LLM agent is prompted to perform an analysis, that compares the results.

The experimental setup conditions the LLM through a structured prompt (see Section A.2) that is consistent with our qualitative assessment framework. The prompt instructs the LLM to assign a rating across each dimension, ensuring consistency in evaluating the visual characteristics of the output. This prompt-driven conditioning method not only aligns with our human evaluation criteria but also allows for a detailed comparison of subjective and automated analysis techniques.

The findings from this experiment are consistent with our qualitative and quantitative analysis. The LLM agent found that the spatial method outperformed the baseline across all metrics, finding that visual novelty was higher, texture features were better preserved and blending the spatial results. The full synopsis and complete results can be found in appendix A.

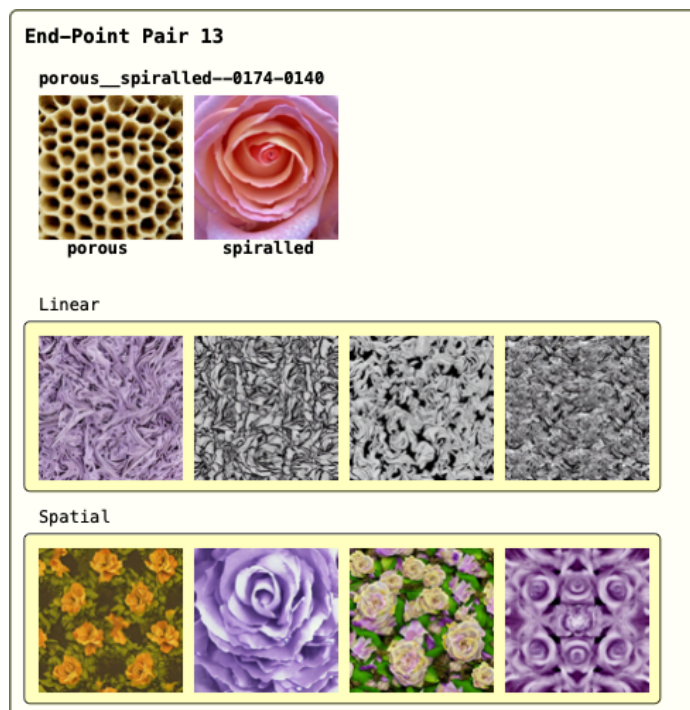


Figure 5.11: **Result Plate Sample.** A sample of one of the twenty results plates shown individually to the LLM agent for evaluation. The complete set can be viewed in Section A.1.

Metric	Linear Mean	Spatial Mean
Feature Preservation	2.85	7.1
Perceptual Blending	2.55	6.8
Visual Novelty	2.65	7.55

Table 5.3: **Agent-Based Qualitative Analysis Summary.** The results of the automated analysis along our qualitative dimensions. The table shows that the spatial approach outperforms the linear approach according to the agent.

Ablation of Optimization

To measure the effect of our path optimization on the interpolation path, we extract the embeddings along the initial path over the manifold and generate 60 images for each resultant point. We do this for each endpoint pair in the experiment set and compute the LPIPS between an image sampled near the mid-point. When compared to output from the optimized path, there is a noticeable

qualitative difference between methods. This is supported by the LPIPS score (see Table 5.4), though LPIPS does not accurately capture the difference in visual quality. In Figure 5.12, we highlight an example that is representative of the optimization’s ability to better integrate features from both endpoints.

Comparison	Average LPIPS
Initial Path - Begin	0.7611
Initial Path - End	0.7487
Spatial Path - Begin	0.7670
Spatial Path - End	0.7652

Table 5.4: **Ablation of Path Optimization.** The results on the LPIPS metric when prompts are extracted from the initial path without optimization.

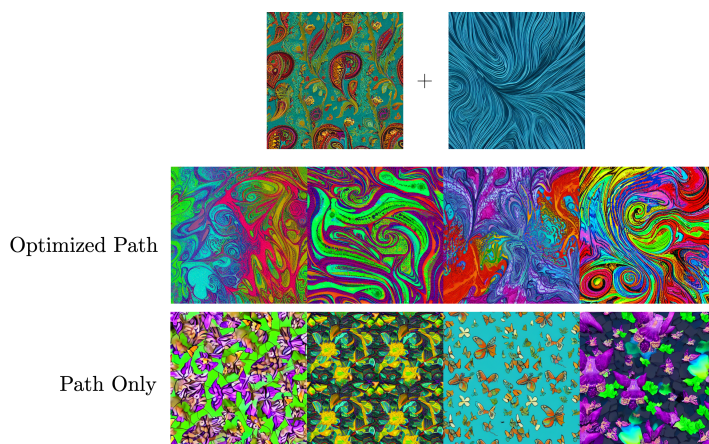


Figure 5.12: **Optimized vs. Non-optimized.** Here we show example outputs from the initial images generated using the path embeddings compared with the images generated from interpolated embeddings on the optimized path.

5.5.3 Limitations

Constructing the latent representation requires approximately 30 to 60 seconds to generate each embedding from the respective input image. For a small dataset this requirement is minimal; however, for a large dataset of thousands of images, ingestion time can become onerous. Many

avenues are available to optimize the construction of the embedding space, including reducing the dimensionality prior to the embedding optimization at ingest time, though these measures are out of scope of this work. We take advantage of the pre-trained CLIP model which offers significant diversity and robustness, but the model may not be well suited for textures. The work may be improved by reducing this reliance and exploring subject-matter-specific training.

While the proposed method demonstrates improved performance over the baseline and naïve approaches, it remains sensitive to noisy regions of the embedding space. This can lead to lower-quality or disjoint outputs along the interpolation path. For example, when the path strays into noisy regions, images may contain non-textural content, and often contain grotesque humanoid formations. More work is needed to understand noisy regions coalesce and how the discrete embedding dimensions are related between nearby embeddings. This could help improve the quality of the embeddings and the resulting images.

Our evaluation relies on LPIPS and CLIP scores for quantitative assessment, but these metrics may not capture important aspects of the method’s performance. Similarly, the agent-based analysis might miss highly subjective qualities that a human observer would appreciate. The integration of features sometimes happens in unpredictable, yet interesting and aesthetically pleasing ways, which are more suitable for human analysis. We envision the LLM-based evaluation as a first step towards agentic evaluations that remain neutral, without favouring one method over another. However, we acknowledge the significant challenge of ensuring that no external factors compromise the neutrality of the LLM-based agent.

5.6 Conclusion

This work demonstrates the efficacy of using CLIP-space as a latent domain for blending visual textures, enabling the generation of coherent and novel combinations of complex, difficult-to-describe concepts. By constructing a spatially-informed interpolation manifold over CLIP embeddings, our approach uses the semantic structure of high-dimensional space to improve blending quality beyond traditional linear interpolation methods. The results show potential for this technique to enhance multi-modal generative models and applications in creative and

exploratory image synthesis, reducing dependence on prompt engineering of textures and offering a more visually-grounded approach to image generation.

5.6.1 Future Work

Future research could investigate blending multiple endpoints within the CLIP-space, going beyond binary combinations to more complex interpolations that involve multiple source embeddings. This could enable richer, layered concept synthesis, particularly valuable for generating patterns or scenes in creative applications. Also, we think that extending the pipeline to more diverse datasets (e.g., photographic or illustrative datasets) would test the robustness of CLIP-space as a general-purpose embedding space. Investigating whether CLIP’s embedding structure retains coherence across varied visual domains would be useful towards validating the approach’s versatility. An open question remains regarding how temporal or sequential data could be represented in a similar space to capture multiple possible trajectories in video, for example.

While this work functions as a proof-of-concept for externalizing CLIP-space, given the results, there is an opportunity for a more interactive system that enables a more natural exploration by dynamically presenting paths through the space by presenting pre-computed options. This is practical given the relative speed with which interpolations can be generated. Better understanding the meaning of embedding dimensions could add additional granularity for controlling the output without relying on text prompts.

There is a larger question about how to construct explorable embedding spaces as we have described. Future work might involve designing learned, navigable spaces that focus on efficiency and coherence, while optimizing for smoothness and interpretability over the manifold. This would require incorporating task-specific constraints into the training process, such as regularization to enforce smooth transitions and alignment with meaningful dimensions. Such an approach could facilitate task-specific spaces more easily and mitigate some of the limitations of the proposed method.

5.6.2 Summary

This chapter explored the use of CLIP embedding spaces for interpolating between texture concepts. The work demonstrated how CLIP-space can be externalized and navigated for interpolation in a way similar to how GANs use latent codes. Chapter 5 moved the focus of the thesis towards multi-modal models where text prompts can be used as a form of control for generative models. This establishes a foundation for further exploration of text-based control and guidance. In the next chapter we move away from image synthesis to work that uses dialogue synthesis. We will describe a method of conditioning and controlling LLMs to facilitate goal-based interactions in games.

Goal-oriented Conversational Games Using LLMs

A large component of generative model research and development is motivated by achieving greater control over dynamic content to enable content that responds more accurately to user intent. This is particularly evident in interactive dialogue systems, where users increasingly expect conversations that adapt fluidly to their choices and behaviours. Traditional systems, often built on rule-based or tree-based designs, are constrained by rigid structures that struggle to deliver the fluidity necessary for truly interactive experiences. These limitations are particularly pronounced in games, where dialogue must not only engage the player but also reflect their progression through the game and align with overarching narrative objectives.

Procedural dialogue, which generates conversations dynamically instead of relying on static, predefined scripts, represents a promising step toward addressing these challenges. By allowing non-player characters (NPCs) to respond in real time to a player's actions, procedural dialogue systems offer the potential to enhance player agency and create more immersive interactions. For example, a player negotiating with an NPC might employ persuasion, intimidation, or barter, expecting the dialogue to adapt seamlessly to their chosen strategy. In theory, such systems could integrate player-driven interaction with the structured narrative and gameplay requirements of the game.

However, achieving this level of integration remains a significant challenge. Procedural systems

struggle to maintain coherence, believability, and alignment with both narrative and gameplay objectives, especially when faced with the unpredictability of player inputs. Compounding this issue is the complexity of tracking changes in game state and ensuring logical progression, all within a dynamic and evolving environment. Traditional rule-based approaches are often ill-suited to this task, as they lack the flexibility required to accommodate the diversity and creativity of player actions.

Generative models, particularly large language models (LLMs), present an opportunity to overcome these limitations by enabling contextually adaptive and semantically rich dialogue. Yet, their successful application in games depends on the development of mechanisms that can track player progression, manage game states, and align generated dialogue with both narrative goals and player actions. This integration can be conceptualized as a mediation task, where a third-party observer evaluates dialogue and game states against designer-defined abstractions of scenarios. The observer's role is to strike a balance between the generative flexibility of LLMs and the structural consistency needed to ensure coherence, narrative focus, and alignment with gameplay objectives.

This chapter introduces a framework for structuring interactions for goal-oriented dialogue generation in adventure style, role-playing games, and integrating the adaptability of LLMs with a controlled graph-based structure. By framing dialogues as puzzles with specific goals, the approach ensures that interactions not only respond to player inputs but also contribute to a coherent narrative flow. This hybrid framework addresses the need for dynamic yet structured dialogues that are consistent with both the immediate context and the game's overarching narrative. Combined with a novel method to assess the LLM's internal state we can control the game's direction while allowing the player freedom to invent solutions to the games challenges without prescribing rules, outcomes or associated dialogue.

6.1 Introduction

Player agency in games, manifesting in part as the freedom to conceive and pursue a wide range of options in solving problems, has historically been constrained by pre-planned solutions

implemented by the game’s developers. LLMs show some promise in weakening these constraints: procedural text, for example in dialogue, can appropriately respond to arbitrary actions from the players. Contrast the flexible conversations LLMs are capable of with the static, menu-driven conversations common in modern computer role playing games (CRPGs). However, LLMs suffer from many problems, including poor coherence over lengthy interactions. Further, by themselves they are not games, since they lack a quantifiable outcome [199]. In this paper, we propose an approach that uses LLMs to provide flexibility and adapt to player decisions, while mitigating their weaknesses with a designer-crafted supervisory structure. We will primarily describe dialogue-based interaction, illustrating the approach with a small *conversational puzzle* [108, 132, 134] style RPG which consists of a conversation between a player and a computer-controlled character (non-player character, or NPC). However, the approach is general and can handle any game scenario whose state and objectives can be described in text.

CRPGs provide players with a wide range of possible activities and narrative paths. However, player actions are often restricted by pre-set dialogues and choices. Tabletop roleplaying games (TTRPGs – e.g., Dungeons and Dragons), the analog antecedent of CRPGs, use a human gamemaster to referee the game and to create in-game dialogue, among other tasks. The TTRPG’s gamemaster can improvise content and adapt to player choices, providing far greater flexibility than is found in CRPGs. The game AI Dungeon [129] uses an LLM to approximate the improvisational nature of TTRPGs, but the experience lacks structure. In this paper, we use LLMs for their flexibility, like AI dungeon, but also enforce structured progress. We use two LLMs, one to adjudicate progress through the game and one to generate dialogue and scene description, analogous to the dual roles of the human gamemaster.

We represent a game scenario as a *puzzle graph*, where each node identifies the current situation, and edges represent legal transitions. A particular graph is created and populated by a human designer; the puzzles, characters, and environment are handcrafted. A game consists of a sequence of turns, where the player provides a dialogue utterance or description of an action, and the system responds. A *dialogue LLM* (DLM) provides immediate responses to the player’s text, while a *question-answering LLM* (QALM) assesses the current situation and determines whether the designer-provided criteria for state transitions have been met. The player seeks to navigate the

such as graphics, sound, or animation, at appropriate moments, including specifically scripted dialogue if needed. While our game did not use scripted dialogue, one might imagine a designer not wishing a pivotal moment in the game’s story to depend on LLM-generated text.

```
PC: "To whom am I speaking?"
NPC: "It doesn't matter who you're talking to! What matters is you listen to what I say or people start getting hurt!"
NARRATOR: "Back and forth with a phone pressed to ear, the tension in the bank robber's stance is apparent to all. Eyes darting to the police lights visible through the slivers of uncovered windows, the gravity of the situation weighs heavily. The negotiator's voice is calm, an attempt to pierce the robber's agitated state with a sense of stability and understanding. Silence shrouds the space as a terse response is awaited."
PC: "I understand. Go ahead."
NPC: "Fine. First, I want a bus here in 30 minutes to take us to the airport. No funny stuff, or I swear, this will end badly for everyone."
PC: "Okay. And second?"
NPC: "I want $500,000 in unmarked bills! No dye packs, no trackers, or people start getting hurt!"
```

Figure 6.2: **Opening sequence.** PC dialogue is written by the player, while NPC and Narrator dialogue is procedurally generated by the DLM.

For our evaluation, in addition to qualitative discussion, we also report a quantitative assessment of our state tracking approach, showing that it is superior to a baseline approach of letting the LLM estimate the state directly.

This paper makes the following contributions:

- An overall framework for LLM-enabled games, with three cooperating elements:
 - A graph-based method to add structure to interactions through node-based objectives.
 - The QALM, used to measure progress toward objectives and manage game state transitions.
 - The DLM, conditioned on designer specified character elements augmented with state-level reinforcement and tuning to remain focused on the designed narrative.
- A proof-of-concept game using the framework, where the player can freely improvise dialogue and invent novel solutions to objectives, while remaining within the narrative constraints of the graph.

6.2 Background and Related Work

6.2.1 Agency

Agency in games, particularly role-playing games (RPGs), refers to the player’s ability to make meaningful choices that influence the game’s narrative, mechanics, or environment [164, 25]. Agency is shaped by the structure of the game logic that mediates interactions, providing limitations and constraints. The degree to which agency as freedom is constrained is a function of both design and technological limitations. However, there must be alignment between what the player is likely to expect and the logic of the game [164], reasonable completeness of options according to the context of the game world [246], and consistency with the player’s mental model of the world or situation [232]. Failing to account for any of these diminishes agency. Balancing player freedom with coherent storytelling in conjunction with these factors is a significant challenge, since diverse and often unpredictable player actions must be accommodated without reducing narrative quality. Accurately and coherently responding to freeform player input remains a largely unresolved problem.

Early text-based adventures such as “Colossal Cavern” [41] and “Zork” [94] provided the foundation for narrative-driven games, but they constrained player interactions to simple commands and limited choices. In early games, the interaction was primarily focused on the environment with little or no NPC involvement. Such games relied on basic methods of natural language analysis based on grammars [255], text parsers [2], and rule-based systems [177].

Since then, contributors to the literature have sought to create systems that dynamically adapt to player behaviour. Procedural content generation techniques [234, 263, 241] enable game environments and quests to respond to individual choices, giving an increased sense of control and ownership. Adaptive dialogue systems allow NPCs to react contextually to players’ decisions [28] and player modeling techniques can predict and respond to player choices [233, 264].

One widely adopted strategy to enhance perceived agency is through illusory choices: decisions that feel impactful but ultimately lead to minimal narrative divergence [232]. Tanenbaum and Tanenbaum argue that true agency arises when players can make meaningful choices with sig-



Figure 6.3: **Menu-based Interface.** Menu-driven dialogue from Starfield [18]. Representative of modern RPG dialogue interfaces.

nificant narrative consequences, achievable through “communicative competence,” where both player and game system understand and respond to each other’s actions. Achieving this remains a substantial technical challenge. In branching-path narratives, illusory agency often relies on converging outcomes despite initially divergent choices, an effective tactic when immediate feedback convinces players of their choices’ significance [123, 246]. This pragmatic approach helps manage the complexity of narrative design but avoids the costs associated with offering truly divergent paths.

Choice-based dialogue systems, where players select from predefined menu options, have long been dominant (e.g., Figure 6.3). This formalism gives designers control over story outcomes but limits player agency by restricting choices to predefined items, often with unclear effects on the narrative. While modern systems expand on this framework by dynamically adjusting options based on factors like prior decisions, character skills, or inventory items, agency remains constrained. Innovations such as real-time decisions (“Indigo Prophecy” [24], “Until Dawn” [62]) and consequence visualization (“Life is Strange” [50]) have added immediacy and impact, while games like “Oxenfree” [191] refined these mechanics with strong writing and aesthetics, reinforcing the dominance of choice-based narratives.

The game “Façade” [153] is a noteworthy milestone in player agency using freeform conversational interfaces. The game used intent detection (e.g., player’s intent is a greeting) with associated behavioral actions (e.g., a dialogue response: “Hello, please come in”) to respond to player input. Façade contained thousands of hand-crafted rules and used extensive dialogue authoring [154].

Despite the vast rule-set, the connections between the dialogue and the player’s actions were often unclear. Imperfect NLU led to understanding failures, and the designers wished to ensure plausible outcomes regardless, with strategies such as deflection or context shifting. We share many of the goals of this project.

After “Façade”, some games sought to use free-form text for player-NPC conversations, albeit with limited success [221, 216]. Lessard [133] explored dialogue-centric mechanics through experimental games that introduced that explore the concept of conversational puzzles. In “A Tough Sell”, players use free-form text to persuade Snow White to eat a poisoned apple. The game’s parser detects predetermined utterances that progress the game, while attempting to assemble contextual responses. Although the game has the mechanics of a conversation, it relies on scripted or rules-based dialogue that mimicked but did not replicate true conversations. Challenges such as maintaining agency and avoiding repetitive, unnatural responses persist.

The dependency on choice-based dialogue was reinforced by the difficulty of in-context NLU. Despite advancements in word embeddings [157, 176], deep learning NLP techniques like LSTMs [87, 229], early transformers like BERT [46], and attention [242], free-form text understanding remained challenging. However, by the late 2010s, vast datasets and enhanced computational power enabled larger transformer models, leading to GPT-2 [184] and its successors. These systems were capable of generating contextually relevant text.

6.2.2 Graph Representations

Graph-based methods have been the foundation of NPC behavior modeling, narrative structures and dialogue management. Early approaches, such as finite state machines (FSMs) defined deterministic state transitions for controlling various game functions continue to be used due to their simplicity and predictability especially in resource constrained environments [158, 64]. Behavior trees allowed more dynamic decision making by NPCs for tasks such as combat and navigation. Some games use behavior trees for such tasks to save memory [204]. To manage narrative progression and experience, story graphs have been used to model story events and choices. However, despite the wide adoption, the extensive hand-authoring limits the possible narrative paths [193]. To address this challenge, NPC actions can be dynamically filtered to

ensure goal-directed and believable behavior while preserving player actions. This approach anticipates long-term consequences of pruning decisions, maintaining narrative coherence and player agency through techniques such as intentionality pruning along with various other graph pruning criteria [247].

More recently, these methods have been combined with machine learning methods. Knowledge-infused graphs, for example, have been applied to generate contextually relevant responses by combining structured representations with statistical models [99, 6]. Similarly, graph neural networks have been used to infer and predict narrative states in dynamic environments, demonstrating potential for real-time narrative adaptation [202, 201]. Hybrid approaches that embed graph structures into reinforcement learning pipelines have also been explored, allowing systems to optimize decision-making policies in interactive settings [66].

6.2.3 LLMs

Recent methods use LLMs to improve upon these traditional methods. Some methods focus on improving the LLM agent’s ability to make in-game decisions, with the goal of providing more appropriate responses. By positioning interactions between a player and LLM as a two-player game, concepts from game theory can be used to train the LLM through reinforcement learning, where the model learns to maximize rewards based on decision making [148]. Other methods generate context-aware responses from NPCs, enabling more meaningful and dynamic interactions with players based on the player’s previous actions, reducing repetitive dialogue and enhancing both realism and immersion [40, 61]. Adaptive dialogue systems can adapt to player input by generating responses that not only adapt to immediate player inputs but also account for the broader context of the player’s path through the game, with the goal of making dialogue feel more personalized [35]. Additionally, integrating LLM-generated dialogue closely with game mechanics allows player choices in conversations to have tangible effects on gameplay, such as influencing quest availability or altering NPC behavior, creating a more interactive and responsive game world [58]. These methods also extend to quest generation. Ashby et al. [6] proposed a framework for RPGs that uses fine-tuned GPT-2 models to dynamically generate personalized quests and NPC dialogue based on player input. Their approach enhances immersion by setting

generated content in the game world, producing contextually relevant dialogues that approach the quality of hand-crafted content.

Consistent long-term dialogue is a significant technical challenge for LLMs. One limitation is the models' finite context window, which restricts their ability to remember and reference past interactions. This can lead to inconsistencies, contradictions, and lack of coherence in responses, particularly when conversations contain subtle details or where the focus of the conversation changes, characteristics common to dialogue based-games. One solution has been to increase the token limit of the context, but even as context window sizes expand, models are unable to accurately recall and integrate details from earlier points in the dialogue history. Typical attention techniques struggle to cover long conversation histories [58, 245] due to the computational complexity of attention and the importance of short baseline latency and time to first token. Potentially more problematic is the recency bias associated with a single initialization prompt, which can further cause LLMs to become repetitive or hyper-focused on a small subset of the desired behaviour or topic [276]. This may break the social immersion, potentially limiting the practicality of conversational LLMs in games, despite the clear benefits.

6.2.4 Tracking

Instead of focusing on LLM-based enhancements to dialogue understanding and quality for improved freeform interactions, we look to intervene at a higher level to structure interactions around designer intent, while providing the ability to inject guidance to the LLM characters at opportune moments. To do this, we must track progress such that the game engine (i.e., the system which executes game code) is aware of the state of the game, its milestones and endpoints. This is a challenge not encountered in the works we have discussed previously. Existing dialogue state tracking (DST) techniques, while adept at managing task-oriented dialogues, are tailored for structured interactions where task and intent are usually clear and based on hand-labeled intent + slot datasets for mundane tasks (e.g., setting an alarm or checking weather) [30]. DST usually relies on domain-specific parsers to update slot values based on parsing utterances directly [254]. Traditional DST focuses on tracking a user's goal and constraints [163], whereas in our work, states are used to track narrative development in a puzzle context. Our conversational setting is

distinguished by its procedural characters and the adversarial dynamics; intents are not static but evolve, and dialogue is a strategic element. Although recent advancements have begun to adapt DST for more complex, unstructured conversations [143], a gap remains in addressing dialogues where information exchange is intertwined with strategic gameplay.

6.3 Method

6.3.1 Motivation

Our objective is to provide players significant flexibility in their approach to solving the game’s puzzles, while requiring neither scripted dialogue nor predefined solutions. The designer will direct game activity at a high level, both through creating prompts that condition the LLM on the current game situation and by defining intermediate states en route to the endgame (with the player either winning or losing).

LLMs have great promise to satisfy our goals. However, the typical strategy is to use a starting prompt plus incrementally updating with dialogue history, which can lead to meandering and at times incoherent exchanges. Limitations on context window length pose further challenges.

In our conception, the game’s structure is defined by three core components:

Puzzle Graph: The puzzle graph provides a dynamic framework for structuring gameplay that uses states narrative guides. Nodes represent designer-provided objectives that control the flow of the game. Edges represent legal transitions between states based on designer-specific criteria.

State Transition Conditions: These are designer-defined criteria that determine when the game transitions from one state to another. These conditions are expressed as natural language questions evaluated by a question-answering LLM, providing oversight to ensure transitions occur logically and in alignment with the game’s objectives. Transition details are described in natural language and can be made quite general, easing the burden on the designer while allowing the player broad scope for creativity and potentially providing high replay value.

Conditioning Prompts: Prompts define the context of the game and establish rules for NPC

behavior. They also provide state-specific rules that reinforce details of the designer’s narrative objectives.

6.3.2 Framework

The proposed system is a framework for a goal-oriented interaction, which can be a standalone puzzle or integrated into a larger game to drive a sequence of interactions. Our method employs two LLM modules that interact with player input to enable the gameplay (see Figure 6.4). The first module is responsible for the structure and flow of the game and includes the DLM component which produces the character dialogue utterances. The second module is the state tracker, which uses the QALM to analyze the game history via the memory to determine the next state. If the transition rules are satisfied the game will progress and new state-specific details will be passed to the game engine.

The game design is specified using three major artifacts: the DLM conditioning, the puzzle graph structure, and the graph transition conditions used by the QALM. The DLM instance is conditioned using a prompt-based approach to define the backstory, character personality, setting details, and any other content of interest. The structure of the game is determined by the puzzle graph, which represents possible game states including the start and end points and the stages of the puzzle objectives that make up the game. The designer can provide state-level guidance to the DLM to direct the interaction by including instructions or fine narrative details. The graph transition conditions dictate how the player advances through the puzzle graph. The game engine assesses the transition conditions to determine the next state, with the QALM evaluating the history of dialogue between player and NPC to determine whether a given query is true or false. At each turn, if the conditions for a given transition are satisfied, the next state will be entered, otherwise the game will remain in the current state as the objectives have not been completed. The transition rules and state prompts can be modified dynamically based on additional context as required.

In practical terms, these artifacts are specified entirely using text with a dictionary syntax (except for the natural language initialization), with no special technical knowledge required by the designer. It would be trivial to employ a form-based approach to remove the dictionary formatting

burden for additional simplification of the specification task.

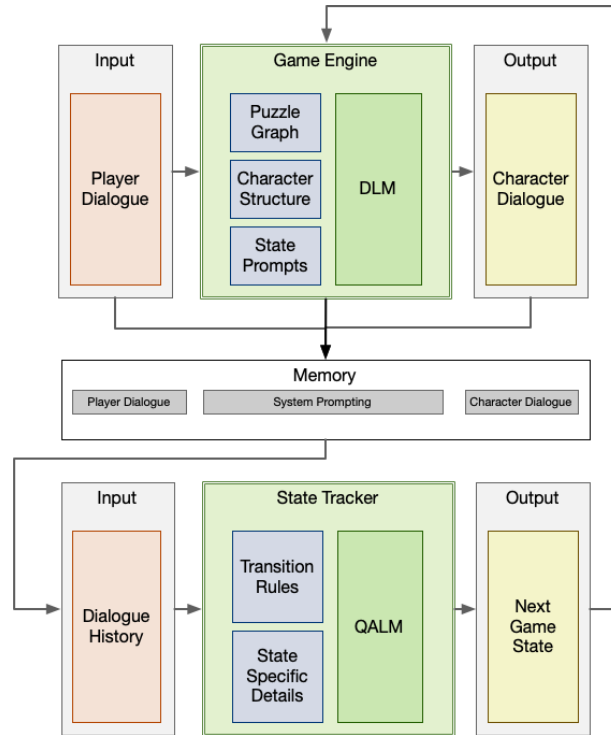


Figure 6.4: **System Overview.** Here we show the two LLM containers that interact to facilitate gameplay. The player interacts with the DLM character. The dialogue is analyzed to determine the state of the game using predefined rules. Blue elements show artifacts produced by designers to govern the game and game progress (© 2024 IEEE).

Game State

Our graph representation contains two types of state node: *objective states* and *event states*. Objective states are the goals that the player works towards, while event states contain gameplay interactions that are not part of the puzzle’s critical path. For example, in our game, we use event states to organize an emotion management challenge as an additional branch of gameplay that must be resolved before the player can continue towards the primary objectives (the NPC becomes enraged and the situation must be deescalated before negotiation can continue); many other uses can be imagined.

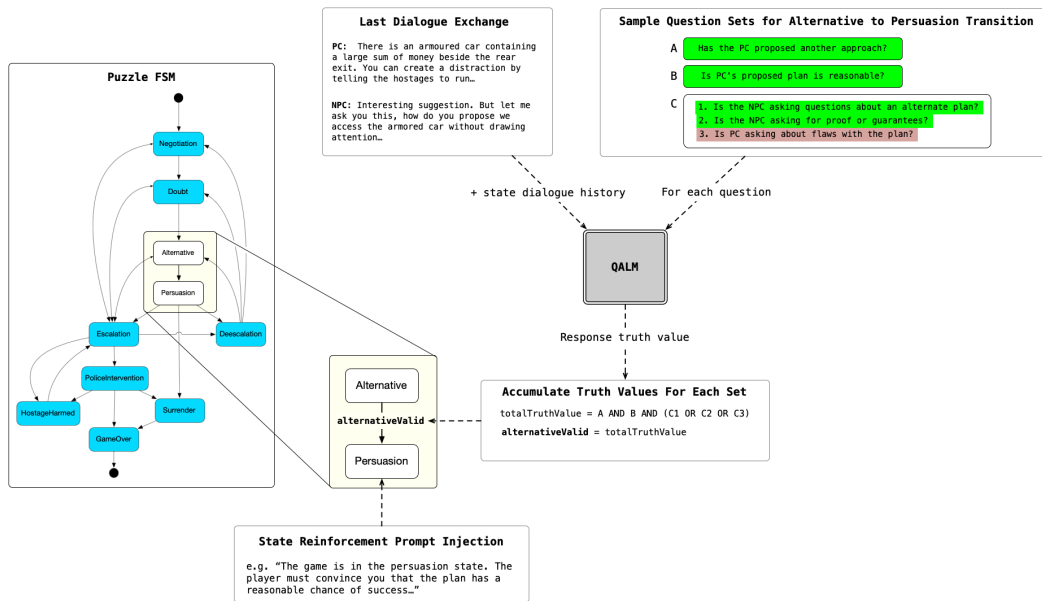


Figure 6.5: **QALM Overview**. Left: the graph governing the entire game. Right: the QALM deciding whether to trigger a transition. The dialogue history and the question set are taken as input. In the top right, we show an example set of questions, with some mandatory (A) and some optional (C). The QALM evaluates the set and returns a truth value for the given state transition. If a transition is triggered, a reinforcement prompt for the new state is included in the prompt block and the user continues to the next stage of the puzzle (© 2024 IEEE).

Using a graph allows the designer to plan the challenges and obstacles the player will face. State-specific prompts provide guidance to the DLM, guiding it to generate appropriate dialogue responses and advising on when to disclose narrative details and clues that will steer players through the puzzle. It gives the designer control over the pacing for the gameplay arc as well as how and where puzzle elements interact with the narrative. Directives can be written using natural language. The LLM is able to interpret statements like: "You can see officers assembling outside the building." or "If the player offers food, then you must only accept pizza". Statements prefixed with "You Must" may have more impact depending on the model used. The graph conceptualizes the structure of the puzzle; navigation of the graph is governed by the game engine and QALM, as described in Section 6.3.1.

The game state can also hold additional variables that contain information relevant to the DLM

character that the DLM can directly control. For example, a character may have an inventory of items related to the narrative that must be tracked. In the case of the negotiation game, the DLM tracks the number of hostages and can modify the state variable if the number changes. Also, at each turn, the DLM reports an emotion-level value (roughly following Plutchik’s wheel of emotions [178]). The UI has an emotion meter based on this value that provides immediate visual feedback on the player’s actions. It helps players gauge the impact of their dialogue choices and adjust their strategy accordingly.

State Tracking

With a graph-based representation of the game, the system must decide when to transition to a new state based on the unscripted dialogue between player and LLM. For each edge of the state graph, we formulate a set of closed-ended questions that characterize the completion of goal-based state objectives. Questions are written in natural language and are addressed to a neutral arbiter. For example, suppose the objective state requires the player to trade an item with the NPC, but we don’t want to specify anything further. Questions might include the following: “Did the player trade an item with NPC?”, or “Did NPC accept PC’s trade?”. A transition question set may contain one or more questions, whose boolean truth values are combined into a single boolean expression with a designer-specified logic function.

A second LLM functions as a question answering system (QALM), used to evaluate individual transition questions. The QALM provides true or false responses to the input questions. It receives the complete dialogue history up to the point of the current evaluation as input (see Figure 6.5).

Using the full history (or state-specific subset) provides extra context for the QALM to reason about the answer using underlying details even when the correct answer may not be captured in a single dialogue exchange pair. By using an independent LLM instance solely focused on boolean QA, rather than attempting to parse utterances directly, we can robustly evaluate state transition criteria. The QALM provides a reliable method to track the open-ended interaction through the graph model.

Algorithm 4 State Transition Evaluation in Puzzle Graph

```

1: Input: Puzzle graph  $G$ , Current state  $S$ , Transition question set  $Q$ , Conversation history  $H$ 
2: Output: Boolean decision to transition to a new state
3: function EVALUATETRANSITION( $G, S, Q, H$ )
4:   for each possible edge  $p$  accessible from  $S$  do
5:      $totalTruthValue \leftarrow 0$ 
6:     for each question  $q$  in  $Q$  do
7:       if  $q$  is a group then
8:          $groupTruthValue \leftarrow \text{False}$ 
9:         for each  $subQ$  in  $q$  do
10:           $ans \leftarrow \text{QALM}(H, subQ)$ 
11:          if  $ans$  is True then
12:             $groupTruthValue \leftarrow \text{True}$ 
13:            break
14:          end if
15:        end for
16:         $totalTruthValue \leftarrow totalTruthValue + groupTruthValue$ 
17:      else
18:         $ans \leftarrow \text{QALM}(H, q)$ 
19:         $totalTruthValue \leftarrow totalTruthValue + ans$ 
20:      end if
21:    end for
22:    if  $totalTruthValue$  is True then
23:      return True ▷ Transition is triggered
24:    end if
25:  end for
26:  return False ▷ No transition triggered
27: end function
28: function QALM( $H, q$ )
29:   Initialize QALM instance with conversation history  $H$ 
30:   Evaluate against question  $q$ 
31:   Receive boolean answer  $ans$ 
32:   return  $ans$ 
33: end function

```

6.3.3 Prompt Structure

The narrative and behavioural details for the scenario are specified using natural language in a prompt. The prompt is a block of text containing context, instructions, and dialogue history (as memory) that is passed to the DLM with each interaction (see Figure 6.6).

The prompt refers both to the dynamic context and to the mechanisms that influence and shape the dialogue-based interaction. It encompasses the details, constraints, and structures that govern how interactions unfold, including the roles of participants, the format of dialogue, and the affordances that guide interactions. The character’s backstory, motivations, and objectives as well as the setting, technological mediators (e.g. telephone) and temporal considerations are also included.

Details are described using natural language in the prompt initialization block. We use element-specific directives to capture different components of the game. This includes descriptions of the NPC’s role, objectives, and motivations. It also includes the greater narrative and backstory for the character and the game.

After each exchange, the DLM returns the NPC’s dialogue response and emotion value, as well as any directives for the game engine such as playing a sound or displaying an image. Each dialogue pair is added to the prompt as part of the dialogue history. Some sample visuals are shown in Figure 6.1.

The prompt includes a specific block for error correction directives. These directives aim to prevent undesirable content or actions of the LLM. Correction directives are usually prefixed with the text "DO NOT" (e.g. “DO NOT apologize”).

Using our state tracking method (Section 6.3.1), we can inject state-specific prompt directives. It is helpful to include directives for actions specific to the current state. We also reinforce important elements included in the initialization prompt at the state level. This is especially important as the length of the interaction increases, since more recent dialogue has a higher weighting in the transformer-based architecture of the LLM [91]. The full dialogue history is accumulated in the prompt as the game progresses. This serves as a memory mechanism for the system to retain the overall context of the conversation, but may cause the DLM to forget important elements of

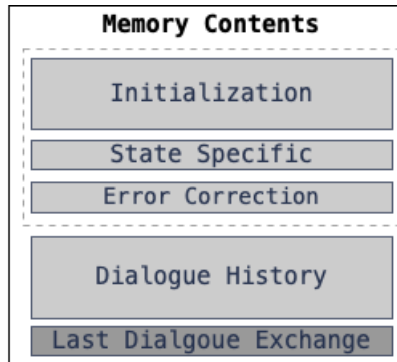


Figure 6.6: **The structure of the prompt.** The initialization block remains constant, the other blocks are updated with new content after each exchange. Error correction directives are part of the initialization and are used to correct undesired output observed during testing. The state specific prompt is defined for each game state. The appropriate prompt is included for the current state. A prompt with this structure is submitted to the DLM for each dialogue submission by the player (© 2024 IEEE).

the initialization or early dialogue. This further motivates the need for state tracking for timely content injection and DLM reinforcement.

6.3.4 Evaluation Approach

Quantitative Evaluation

We compared the performance of our state tracking method with the DLM’s ability to predict the next state directly. Both methods were run simultaneously, using the same input information at each conversational exchange to predict the subsequent game state. During gameplay, we collected data on the previous game state and the predictions generated by each method. The DLM was provided with the complete state and transition information, including detailed descriptions of both. The ground truth dataset was annotated following QALM’s transition parameters to evaluate each exchange and its history against the internal transition conditions. Performance was assessed by comparing the predicted states from both methods to the annotated ground truth using metrics such as accuracy and consistency with the transition rules.

Qualitative Evaluation

We qualitatively evaluated the impact of the game structure imposed by the puzzle graph in a series of approximately fifty complete playthrough sessions. Through these sessions, we aimed to assess the game’s flexibility and coherence. We judged flexibility based on the game’s ability to support diverse strategies in achieving objectives; coherence was judged according to the degree to which the computer-generated dialogue maintained internal consistency and responded to the player’s dialogue, and on how well the evolving conversation adhered to the predefined narrative.

Across multiple playthroughs, we maintained three distinct objectives, not all of which were active in every playthrough. First, we sought to reach an end state; primarily, we attempted to achieve the game’s goals, but we also assessed whether the game correctly adjudicated a loss state. Second, we approached the game with diverse strategies and personas, seeking to explore the flexibility and adaptability of the method. Third, we acted in a malicious capacity to test the game’s ability to maintain coherence despite deliberate disruptions; for example, we introduced extraneous events and non sequiturs to determine how well the game could recover.

6.4 Proof-of-concept Game

We demonstrate our method in a proof-of-concept game: a hostage negotiation scenario where the player takes the role of a police negotiator seeking to secure the safe surrender of a bank robber holding hostages inside a bank (see Figure 6.1). The DLM controls the NPC bank robber / hostage taker, conditioned with a detailed backstory, with escape as his primary goal to support the benevolent motivation of helping his ailing sister. See Figure 6.4 for a visualization of the information flow.

The game proceeds in a series of rounds. In each round, the player describes an action or utterance, and the system responds and updates. If the game reaches a designated endgame state, the game finishes; otherwise, the cycle continues. In some games or interactions, the player may also have the option to terminate (perhaps merely ending the conversation, or, if available, taking drastic action such as calling in the SWAT team).

Hostage scenarios are high-stakes conflicts requiring emotional intelligence and verbal tactics to avoid violence [156]. The player seeks a non-lethal resolution while the NPC robber aims for escape; these asymmetric goals create opportunities for interesting dialogue. The emotional state of the NPC is critical to the scenario; the player must keep the NPC calm in order to succeed. Limiting the scope of interaction to the bank creates a controlled environment to experiment with game mechanics and player responses [199].

6.4.1 Implementation Details

We use the OpenAI GPT 3.5t [21] language model, capable of handling a wide variety of scenarios without fine-tuning. The game engine uses JSON objects to extract the different components of the DLM responses. In our experiments, open source models [236, 36] were not capable of consistent formatting.

The structured output enables the separation of dialogue and system messages. This distinction is crucial as the player interacts with the DLM in the role of a character, while the game engine communicates with the LLM at a system level, issuing procedural, state, and other prompt-based commands in parallel. Character and system interactions must remain distinct to maintain the integrity of the gameplay experience.

6.5 Results and Discussion

A key question for our design is whether the formal Graph + QALM arrangement is necessary. We performed a quantitative experiment in which we compared state transitions using the QALM vs. the state transitions suggested by the DLM directly. We found that the QALM produced considerably more plausible state transitions than did the DLM, as judged by a human referee. We perform a qualitative evaluation of the gameplay to assess impact of the QALM as well as the DLM responses and the overall experience. Details follow.

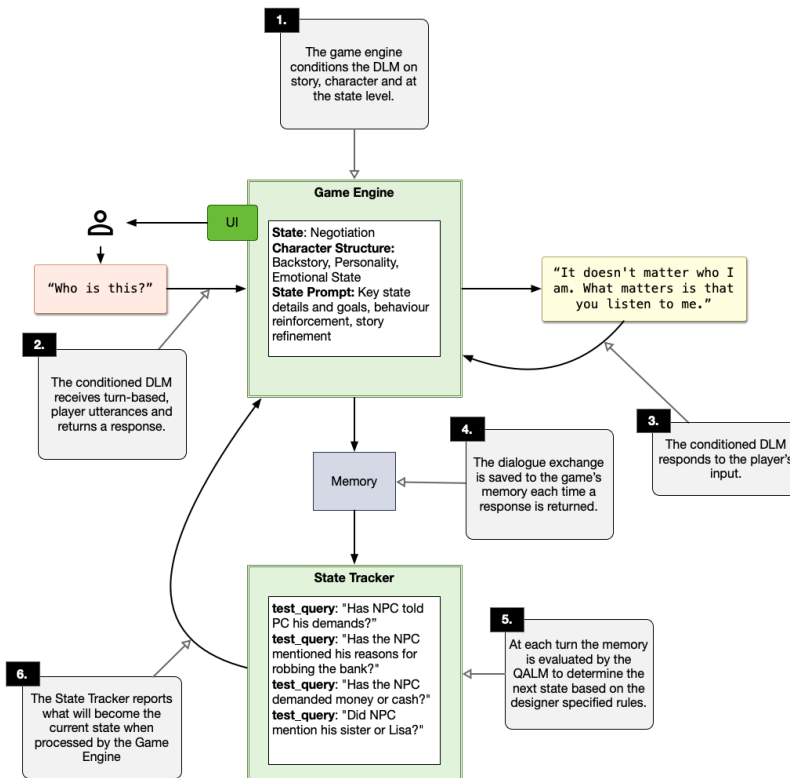


Figure 6.7: **Information Flow.** We show how the information flows through the components of the system (©2024 IEEE).

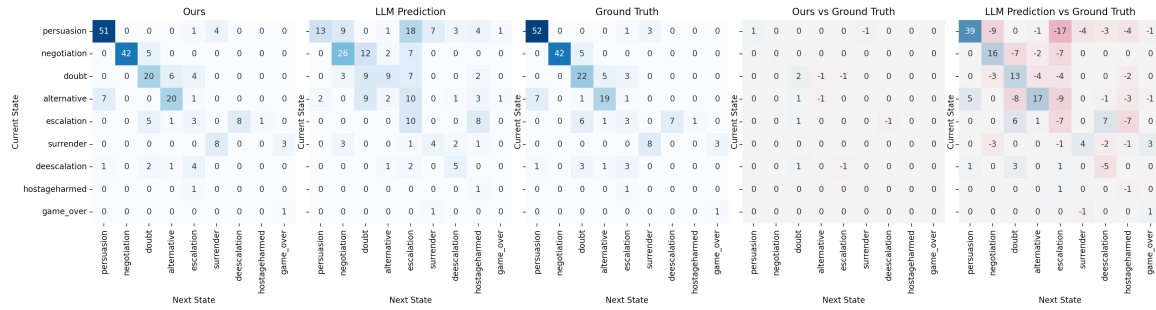


Figure 6.8: **State Frequency**. We compare our state prediction capacity vs. the LLM’s capacity. Most transitions are self-transitions, as shown by the transition frequency on the diagonal. Our method matches the ground-truth more closely than LLM predictions. In the two right-most plots, we visualize the deviations from the ground-truth. The ideal deviation matrix would be all zeros (© 2024 IEEE).

6.5.1 State Management Experiment

The results of our quantitative experiment as a transition frequency heatmap between states and the magnitude of the differences, shown in Figure 6.8. The plot shows the heatmap for both methods with the ground truth for comparison. The LLM without the QALM struggles to accurately predict the correct game state, often jumping erratically between states. This could be jarring for the player as state-specific cues like graphics may not reflect the dialogue and change too frequently.

As we have discussed, a conversation in a game setting requires some structure. Our puzzle graph provides designers the ability to craft this structure and requires the DLM and the NPC to conform to that structure. We have shown that the DLM itself is unable to accurately keep the state of the game consistent with the intention of the design. When defining game objectives using our graph-based method combined with the QALM for state management, we can impose a structure with defined end-points while taking advantage of the DLM’s ability to allow a flexible solution space. This highlights an important distinction between this method and open-ended narrative experiences.

6.5.2 Gameplay Analysis

Flexibility

Our method provides players with the ability to invent their own solutions to the challenges presented by the game while tracking their effect. In the context of the proposed scenario, upon learning the antagonist’s motivation to help his sister, the player must consider how to use this information to formulate a solution. Solutions such as arranging treatment facilitating surrender, co-plotting an escape plan resulting in capture, and aggressively discrediting the antagonist’s plan, were successful strategies, where others such as firmly advocating a peaceful surrender or simply providing cash were not. The most implausible ideas were often rejected as being part of a trap, for example. None of these reactions were scripted in any way. Importantly, the game can determine when the antagonist has capitulated and the next state as planned by the designer can begin. This is presented as a shift in direction and tone by the character, unlike the meandering that can occur in an unstructured interaction.

```
PC: "My brother is a doctor who treats desperate patients using experimental methods."  
NPC: "You're saying your brother can help? How can I trust that? If there's even a slightest chance... I need guarantees, not just empty promises!"  
PC: "I can get him on the phone. You can ask him questions."  
NPC: "Do that. Put him on the line. I want to hear it straight from him. And I swear, if this is a setup, I'll know, and it won't end well for anyone here."  
PC: "I conferenced him in. He's on the line now. His name is Dr. Jones."  
NPC: "Dr. Jones? Tell me about these treatments. How can you help Lisa? And how is this different from what we've already tried?"  
PC as Dr. J: "I am sorry to hear about Lisa. We treat people like her using experiment gene editing techniques. The treatment is free."  
NPC: "Free? No one does anything for free. What's the catch? She's all I got, and if you're telling the truth... Look, just swear to me this is real. Swear it!"
```

Figure 6.9: **Character Swap**. The player invents Dr. Jones and takes on the persona. NPC adapts.

The freeform text interactions enabled players to experiment with creative interactions, such as role-switching between characters during a conversation, shown in Figure 6.9. The NPC was able to recognize and appropriately respond to these new roles, both semantically and tonally, revealing character traits that were not directly specified. The QALM was able to track state across these context switches. Also, in some instances, the DLM introduced spontaneous actions and emotional descriptions within its dialogue, using conventions from early text-based forums and multi-user dungeons. These actions, often denoted by asterisks (e.g., *Slams fist on table*), added

another layer of interaction during gameplay. The identification of these emergent interaction modes suggests an additional dimension of player exploration, aligning with the exploratory engagement described by Montfort [161].

Response quality

During gameplay, DLM responses are natural and retain appropriate context as the game progresses. However, the responses are difficult to evaluate in terms of quality and complexity. For example, the NPC may seem one-dimensional: it is narrowly focused on achieving its own goals. However, this serves to keep the game on track, compensating for adversarial or otherwise unfocused inputs.

While individual responses were often realistic and contributed to the overall depth of the game (e.g., Figure 6.10), the broader quality of the conversation demonstrated potential for improvement. Specifically, the DLM’s reasoning capabilities seemed constrained, with the NPC primarily relying on emotional responses and threats rather than employing more nuanced strategic moves. For example, a strategic move in the proposed game might be a controlled display of aggression (without harming hostages) or making incremental concessions.

6.5.3 Discussion

The qualitative analysis highlights several key aspects of the method’s goals. For instance, the system allows for mid-conversation persona switching, demonstrating the emergent gameplay that Montfort [161] describes as a source of pleasure in interactive fiction. Additionally, the game dynamically adjudicates player intent based on situational constraints, effectively balancing player agency with narrative coherence [27]. By imposing structure on otherwise open-ended interactions, the game quantifies player progress toward specified goals, enhancing the overall coherence of the narrative. In the context of recent advancements in LLM-driven gaming [148, 40, 61], our work facilitates measurable, goal-oriented outcomes and improvisational capabilities that adapts to player actions.

The quantitative evaluation demonstrates the strengths of our approach, showing that the QALM-

A

PC: "Here is my brother's research website. It shows the people he has helped against all odds. Check it and tell me what you think"
NPC: "Fine, I'll take a look at this so-called research website of your brother's. But don't think for a second that this will change my mind. I'm only doing this to satisfy my curiosity. If I find any hint of deception or manipulation, negotiator, you'll pay the price. Don't test my patience."
PC: "What do you think?"
NPC: "I've taken a look at the website, negotiator. It seems like your brother has indeed helped people in similar situations. I can't deny that there is a chance, a glimmer of hope. But understand this, negotiator, my sister's life is in the balance. If I surrender, will you make sure that she gets the treatment she needs? Will you guarantee her safety and well-being? Because if you can't, then all this talk means nothing to me."

B

NPC: "Proceed with caution and ensure the safe transition of the hostages. I am entrusting their well-being to you. Any harm that befalls them will have dire consequences. Uphold your commitment to a peaceful and safe resolution. Let's move forward."

C

NPC: "You're damn right I'm angry! Don't push me any further or else another one of these hostages will meet the same fate."

D

PC: "I understand your skepticism. We can discuss your demands. It's crucial that we keep everyone safe while doing so. If money is what you need there's a way to address that without harm. Can we agree to focus on a peaceful resolution that meets your needs and ensures everyone's safety?"
NPC: "I've heard enough empty promises from people like you. Peaceful resolution? Safety for everyone? Spare me the clichés. I don't believe you have any real intention of meeting my needs. Until I see concrete action, I won't consider anything you say."

Figure 6.10: **Game Quotes.** A selection of interactions between the player (PC) and LLM character (NPC) as follows: **A.** An exchange during persuasion where the player invokes an external website. **B.** An example of the "role-reversal" issue. **C.** A heated exchange from the hostage taker during an escalation event. **D.** In this example the hostage taker is taking the player to task for being vague.

based state management improves the coherence and flow of gameplay. By decoupling state management from dialogue generation, our method ensures that player actions are evaluated against clear criteria, leading to more consistent gameplay experiences with minimal out of character interruptions. The puzzle graph framework, in particular, allows for structured progression through the game, maintaining narrative integrity while still accommodating player improvisation. This scaffolding effectively uses the capabilities of the LLM, turning what could be an unstructured interaction into a well-defined and engaging game experience.

The proposed game scenario uses five core stages. The interaction within each state is partially compartmentalized, with additional external guidance provided as required. This has the potential to improve scalability for larger designs. During our evaluation, we employed a context window of 16k tokens. This was sufficient to contain the complete history of focused sessions, while longer sessions exceeded the limit. Experiments with a smaller 4k token context showed that when simply using sliding window truncation, the flow of the game was not affected. This is partially the result of targeted state-level reinforcement and compartmentalized design of each state. However, retaining longer term utterances with important content may require a more advanced memory management protocol, a possible direction for future work. Given the rapid increase in context window sizes, the context window may not represent a long-term challenge. However, the increased computational cost associated with larger context size, along with memory and latency considerations could be a barrier for implementing long-form interactions using open source models without state-of-the-art memory management (e.g. [127, 55]).

Rigid and explicit structure limits player choices. In conventional systems, game states are predefined, and dialogue trees prescribe available options. Consequences of choices are predetermined. In contrast, our proposed method encourages diverse player choices, using states as waypoints that guide transitions between emergent states. These emergent states and corresponding dialogue arise dynamically in response to user input during the journey between explicit waypoints. By offering the user the ability to make choices unconstrained by menus, the method supports a high degree of completeness of options. Similarly, through the conditioning of the NPC and the associated continuous tracking and updating of state information, the player's expectations are satisfied. Thus, player agency is improved [164, 246, 232]. Furthermore, previous constraints encountered with pre-scripted choices are ameliorated, as the LLM performs much of the heavy lifting to enable more freeform interaction.

To formalize this concept, we define *hard agency* as a form of player interaction in which solutions are not predefined, and the player must independently conceptualize and initiate actions. The system must dynamically interpret and respond to these actions without relying on scripted outcomes, as in our method. This form of agency does not preclude structure or constraint. Rather than limiting interaction to a fixed set of inputs, designers support hard agency by shaping

the problem space through contextual cues or situational constraints. These guide the player's reasoning without prescribing specific paths or solutions, enabling freedom of action within a meaningful structure.

In contrast, *soft agency* offers the player apparent choice through a bounded set of predefined actions, giving the impression of control while limiting the scope for improvisation. These actions may be explicitly presented through interfaces such as dialogue menus, or embedded in environmental affordances. In this mode, constraint is applied at the level of available actions, where the designer must define in advance what the player can do and how the system will respond. While this can create the appearance of agency, it often restricts meaningful player improvisation and limits the scope of player authorship.

While the definitions focus on the structure of action and solution space, each mode of agency is often associated with distinct systemic patterns. Hard agency tends to appear in systems that support emergent outcomes, where player actions may reshape future interactions in ways not explicitly anticipated by the designer. In soft agency, longer-term consequences can also occur, but these are typically prescribed—defined in advance as part of a branching narrative or decision tree. As a result, their scope and impact are constrained, even when they extend across multiple encounters. These associations are not absolute as open-ended problem solving can occur within tightly constrained action spaces (as in chess), and designer-imposed outcomes can dominate even systems that appear to offer wide expressive freedom (as in illusory choices). What ultimately distinguishes hard agency is not surface-level expressiveness, but the system's capacity to interpret, adapt to, and sustain unscripted player actions. This places a technical demand on the system itself. Supporting hard agency requires new methods that can interpret improvised player actions, maintain persistent world state, and produce coherent responses that are not predefined. Traditional content pipelines, which rely on prescribed events, branching structures, or fixed affordance maps, are not well suited to systems in which player intent and system behaviour develop in response to one another over time.

Limitations

The proof-of-concept relies on a large-scale commercial LLM API. This is impractical for game studios aiming to publish using the proposed method. However, rapidly advancing open-source models, which can be fine-tuned and run on end-user hardware, offer a viable alternative. Many perform as well or better than our baseline model across many tasks [274]. Techniques like quantization significantly reduce memory usage and inference times, making these models feasible even for lower-end GPUs [45, 57]. However, even with these optimizations long context windows remain problematic due to the non-linear scaling of computational requirements. While challenges remain for mobile devices, we do not expect computational requirements to hinder the adoption of LLMs for many game types and devices in the near to medium term.

The DLM does not always adhere to the prompt directives, sometimes causing the NPC to respond out of character. For example, the NPC may attempt to protect the hostages (see Figure 6.10 B), or object in some way to the game situation as if applying a moral judgment, incompatible with the character. Such deviations can be easily corrected at the global or state level, but are only feasibly uncovered over many gameplay testing sessions. Whenever we discovered a deviation, we added a corrective prompt directive to the behavior correction block of the prompt (see Section 6.3.3). With sufficient testing of a game version, most consistency issues can be corrected. However, a persistently adversarial player can eventually cause failures via creative attempts to confuse the DLM through out-of-context utterances or other attacks. Many such attacks can be handled via pre-processing player input, though investigating this fully was out of scope for the proof-of-concept.

6.6 Conclusion

We presented a novel approach to goal-oriented conversational games, integrating structured state management with the flexibility of large language models. The introduction of the puzzle graph in tandem with the QALM and DLM LLM instances allowed for the measurement of player progress toward objectives, leading to a more coherent and structured interaction.

Our graph-based method structures the game objectives as stages of a puzzle, using closed-

ended questions to create a fact-based representation of objective completion. We showed that by evaluating truth values of this representation against dialogue history between a player and adversarial LLM character, we not only accurately track the game state but also make progress on the challenges of player agency and narrative coherence that are often encountered in open-ended interactions.

To evaluate our method, we created a new type of game that extends the concept of the conversational puzzle. We demonstrated that an appropriately conditioned LLM-based character can serve as a realistic adversary, providing plausible dialogue while also adapting and reacting to diverse and creative player improvisation. This is a significant improvement when compared to standalone LLMs that often produce shapeless, meandering interactions.

We supported these claims with a detailed qualitative analysis of gameplay interactions and a quantitative evaluation of the state tracking performance of our method compared to baseline LLM generated state predictions.

Building on the findings of this work, several potential directions are possible. One critical area involves expanding the application of the puzzle graph and QALM framework to more diverse and complex game scenarios. Using the method to manage interactions across multiple characters, while incorporating dynamic changes to objectives based on events in a wider game context is a natural next step. Similarly, embedding LLM-enabled interactions within more conventional games (especially, but not limited to, CRPGs) would create more lively and believable settings and gameplay.

Additionally, improving the strategic reasoning capabilities of the dialogue language models (DLMs) is an interesting avenue that could pose more elaborate challenges to players. This could involve the development of more sophisticated training methodologies, to enable more nuanced and contextually appropriate decision-making in adversarial environments. Finally, addressing the ethical implications and safety concerns associated with AI-driven gameplay, particularly in maintaining alignment and avoiding unintended outcomes, represents an important consideration when developing such characters.

6.6.1 Summary

In this chapter, we developed a structured approach to integrating large language models into interactive dialogue systems designed for goal-oriented gameplay. We proposed an LLM-based question-answering system that observes the gameplay to detect designer-specified waypoints. Waypoints are based on state graphs that encode puzzles and narrative structure and the associated transition rules. The method is able to track player actions and direct state transitions to ensure that interactions remain coherent and true to the designer’s specification. By closely tracking the game state, we enable periodic memory updates and additional conditioning directives that improve the NPC’s coherence and response quality. The work makes a step towards solving long-standing challenges that have restricted player agency and demonstrates that free-form, non-deterministic gameplay could become a component of open-world, role-playing games in the future.

In the next and final chapter, we will present our conclusions and summarize our findings. We highlight future work and identify an theme of inbestigation that could offer improvement to each area of work presented in this thesis.

Conclusions

In this thesis, we have introduced a series of methods aimed at enhancing coherence and guidance across content generation tasks in both image and text domains. Our work addresses persistent challenges in texture synthesis, dialogue systems for interactive gaming, and visual interaction and control of LDMs using custom embedding spaces. The contributions presented in this thesis advance the state-of-the-art by improving the diversity achievable in single-image texture synthesis, enabling greater player agency in solving in-game objectives, and proposing a novel framework for leveraging the CLIP model as a generator of structured latent spaces.

7.1 Findings

Throughout this thesis, we have described a series of coherence-enhancing methods that improve the utility of generative models for creative and interactive applications.

7.1.1 Diverse Non-homogeneous Texture Synthesis From A Single Exemplar

1. **Joint optimization by parameter sharing** We showed that by combining a GAN and VAE into a jointly optimized network we could encourage the complementary strengths of each network. The combination is made possible through the development of a parameter sharing technique that iteratively trains layers of each network with alternating objectives. The combination of

models and the efficiency of the joint layers also results in very fast training times. We found that this approach can generate highly diverse non-homogeneous textures while maintaining visual coherence, training a model on a single example within fifteen minutes.

The finding is important, as the trade-off between diversity and quality, which is a significant challenge faced by texture synthesis methods. The ability to reduce this trade-off to produce virtually unlimited variations of the input which approaching the original quality of the exemplar could be used as a tool in texture production workflows where designers require options derived from existing assets or previous stages of the creative process.

2. **Sampled similarity loss**

We described a regularizing loss function that uses feature representations from a pre-trained network to measure diversity during the training process. The method samples a set of generated images at regular intervals during training, then computes the average pairwise differences between feature representations for a selection of layers in the network. The value is used to penalize low diversity values in the overall loss calculation. We found that this loss reduced the model's tendency to overfit on the the input exemplar resulting in preserved or improved diversity of the output images.

Overfitting is a well-documented challenge in training GAN-based models. Particularly challenging is the case where there is only a single exemplar due to limited source data to draw from. The similarity loss method combined with the associated training regime represents a simple but effective mechanism to reduce overfitting with data constrained training environments.

7.1.2 Blending Difficult to Describe Concepts in CLIP-Space

1. **Creation of traversable CLIP-spaces**

The work demonstrates that the latent space of a pre-trained multi-modal model, such as CLIP, can be extended into a traversable representation for specific datasets. By aligning textual and visual embeddings within this shared latent space, the embeddings can be used to construct an external graph-based representation that enables interpolation and blending between concepts

in the space.

This finding demonstrates the flexibility of CLIP and its embedding space to adapt to novel tasks. The ability to create a structured latent representation for a custom dataset expands the applicability of CLIP beyond its original scope, adding utility as a foundation for specialized synthesis tasks. This contributes to ongoing research on the reusability of pre-trained models, where the embedding structure can be leveraged for creative exploration and data-driven synthesis.

2. Coherent concept blending

The chapter presents a technique for combining visual concepts within the latent space, generating outputs that blend features from the input while retaining features and combining them in interesting ways. By interpolating embeddings along meaningful trajectories, the method creates novel visual-textual combinations that reflect underlying feature similarities.

This approach demonstrates how embedding-based interpolation can align and unify concepts across modalities, making it particularly useful for creative applications where textual descriptions and visual references must be synthesized into cohesive outputs.

3. Spatially informed interpolation

The research uses spatial relationships within the constructed latent representation to enable interpolation between concepts. This approach produces blended outputs that are informed by the feature-level properties of the embedding space as governed by spatial/semantic proximity, offering a novel method for exploring latent relationships within multi-modal data.

This work provides insight into the geometric properties of shared embedding spaces, demonstrating how structured traversal within these spaces supports the generation of semantically blended outputs. The findings highlight the role of spatial relationships in embedding spaces, such as alignment and interpolation, in guiding generative tasks. Using these properties, the study provides some evidence for the potential of traversable embedding spaces as a tool to add visual control when interacting with CLIP-like embeddings. This approach lays the groundwork for further exploration into additional interaction modes using CLIP.

7.1.3 Goal-oriented Interactions in Games Using LLMs

1. **Unscripted puzzle solutions** Our work showed that an LLM-based dialogue engine can accurately adapt to fundamental narrative changes contributed on-the-fly by the player. Specifically, given a human-designed goal or puzzle, without specifying valid solutions, our method facilitates player agency by allowing the player to invent their own solution. Using our question/answering LLM approach, the system evaluates the plausibility of these actions and dynamically incorporates the implications into the narrative while honouring the overall game design, scenario, and associated constraints. We found that the structured approach we presented performed better than an unstructured, LLM controlled alternative.

Giving the player the agency to craft their own solutions in response to game stimuli has been a long-imagined goal of game designers. Our approach offer one potential avenue, that could be incorporated into future game system architectures. However, many questions remain about how

2. **Procedural dialogue**

In the chapter we presented a novel approach to managing goal-based interactions with LLMs. Using our method, we showed that after capturing the human-designed game in a state-graph it is possible to accurately predict the current game state. Unlike typical, open-ended interactions, tracking the game state facilitates the measurement of game objectives and allows the definition of transitions and endpoints as necessary components of this kind of game. Furthermore, knowledge of the game state enables periodic updates to the LLM’s memory and timely injection of reminders and state-specific details. As a result the improvisation of game dialogue can be carefully controlled by the designer, resulting in improved narrative coherence throughout a gameplay session.

Our findings demonstrate how tracking interactions with LLMs can enable goal-based interactions, something that has only been approached in restricted “chatbot” style scenarios. We think this could represent an important step towards incorporating LLMs into open-world RPGs for more dynamic interactions.

7.1.4 Future Work

The work in this thesis covered a lot of ground, from texture synthesis to procedural dialogue control. We suggest several functional extensions of the work including the following directions:

In Chapter 4, future work may focus on extending the model to better handle structured textures, such as those with internal symmetries or periodic features. This could involve architectural modifications designed to preserve global coherence, through alignment mechanisms or constraints that favour regularity when detected. Another important direction is the introduction of user-controllable parameters or interactive guidance that influence the synthesis process. The method's ability to rapidly generate new variations suggest the potential for dynamic composition of higher spatial resolution images. Here multiple outputs could be seamlessly combined to form larger, spatially coherent textures that remain representative of the input. This capacity would be particularly useful in contexts such as cinematic or game environment applications, where continuous variation across larger surfaces is required. In terms of the evaluation, existing metrics do not adequately capture the properties we wish to capture for generated textures. There is a need for the development of improved diversity measures, possibly based on global and localized entropy to better compare the spatial distribution of features compared to the input. Such a metric could better quantify meaningful variation from randomness to measure texture diversity in a more principled way.

For Chapter 5, future work could explore richer interpolation in CLIP-space by blending multiple source embeddings and give the user control of the preferred elements of each. Extending the method to operate on more diverse datasets of natural images would test the robustness and versatility of CLIP-space across visual domains. This could further extend to temporal data, as in video or animation to capture sequential trajectories. The proof-of-concept described in Chapter 5, suggests potential for a higher level of interaction both by dynamically adding new images to the constructed space and a direct visual exploration method to allow users to navigate paths based on visual feedback. More broadly, constructing explorable, task-specific CLIP-based spaces may benefit from learned manifold optimized for smooth transitions and coherence, to better accommodate control inputs instead of relying on the fixed graph.

Further development of the goal-oriented framework in Chapter 6 could involve multi-agent frameworks in which agents reason about and incorporate elements of previous choices and interactions according to designer specified relationships between characters and story elements. This build directly on the architecture presented in Chapter 6, which provides an interface for the specification of designer intent and progress tracking. This direction also furthers the core goal of enhancing player agency by providing a mechanism for the effects of player choices to reverberate in various ways across characters and events. While our approach provides the underlying architecture for these ideas, additional focus is needed to build more specialized models that improve character specification combined with user intent and event importance inference to better characterize improvised interactions to add more depth to subsequent interactions. Finally, to ensure that multi-character, multi-goal experience are scalable, more work is required to better track and respond to player choices. We want to support long range dependencies across a range of characters and story events. A key question involves understanding how to separate concerns between models: Should we track long term dependencies with a single tracking LLM or distribute state knowledge across all characters? These directions point toward systems that are not only reactive to input but actively plan story elements around improvised interactions. They expand the generative capacity of the methods presented in this thesis while maintaining a focus on controllability, adaptability, and human-guided design.

By taking a step back, we observe that the methods presented are entirely reactive. That is, they respond directly to the input provided without explicitly modelling or understanding user intent. The language models we used have an implicit reasoning capacity where training conditions the model to provide responses that are statistically likely to align with user intent. In Chapter 6, we described a method for attaching intent detection, by defining explicit rules that provide sufficient context for the LLM (the QALM) to better contextualize both player and designer intent, which also enables goal tracking.

It may be the case that some LLMs have an emergent reasoning capacity [89], where reasoning is defined as the ability to generate step-wise processes for problem solving. While this may help in some types of interactions where logical sub-tasks are easily identified, it is not likely to support reasoning about user intent. We think a stronger form of explicit training is required to build the

capacity to model and reason about intent. For example, future work could explore fine-tuning large language models on datasets specifically curated to capture diverse or tasks-specific user intents. This could include multi-turn dialogues in which user goals are only partially revealed or evolve over time, forcing the model to infer intent dynamically. Furthermore, incorporating reinforcement learning techniques, such as Reinforcement Learning with Human Feedback (RLHF) [171], could further refine the model's ability to prioritize and align with user intent in real-time interactions.

Another promising area for investigation in this context is the integration of external modules explicitly designed for intent recognition and reasoning. For example, a dedicated intent parser or a symbolic reasoning module could preprocess inputs to identify potential goals, constraints, or ambiguities before passing the refined context to the LLM. This hybrid approach may bridge the gap between implicit reasoning within the LLM and the need for explicit, actionable intent modeling and is a natural extension of our work.

This concept could be extended to include multi-agent interactions. For example, in Chapter 5, we explored procedural dialogue control. A logical next step could involve building systems in which multiple agents dynamically adapt to each other's inferred intents, leading to richer and more coherent emergent behaviours. This could be particularly impactful in games or more collaborative tasks.

Ultimately, systems that better understand and react to user intent could improve the results for the creative tasks explored in this thesis and extend to future agents facilitate Human-AI co-creation for more complex tasks such as scientific discovery.

References

- [1] Rameen Abdal, Yipeng Qin, and Peter Wonka. “Image2StyleGAN: How to Embed Images into the StyleGAN Latent Space?” In: *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. Seoul, Korea (South): IEEE, Oct. 2019, pp. 4431–4440.
- [2] Alfred V. Aho and Jeffrey D. Ullman. *The Theory of Parsing, Translation, and Compiling. 1: Parsing*. Prentice-Hall, 1972.
- [3] Adib Akl et al. “A Survey of Exemplar-Based Texture Synthesis Methods”. In: *Computer Vision and Image Understanding* 172 (July 2018), pp. 12–24.
- [4] Martin Arjovsky, Soumith Chintala, and Léon Bottou. “Wasserstein Generative Adversarial Networks”. In: *Proceedings of the 34th International Conference on Machine Learning (ICML)*. Vol. 70. Proceedings of Machine Learning Research. 2017, pp. 214–223.
- [5] Rajat Arora and Yong Jae Lee. “SinGAN-GIF: Learning a Generative Video Model from a Single GIF”. In: *2021 IEEE Winter Conference on Applications of Computer Vision (WACV)*. Waikoloa, HI, USA: IEEE, Jan. 2021, pp. 1309–1318.
- [6] Trevor Ashby et al. “Personalized Quest and Dialogue Generation in Role-Playing Games: A Knowledge Graph- and Language Model-based Approach”. In: *Proc. of the 2023 CHI Conference on Human Factors in Computing Systems, CHI 2023*. ACM, 2023, 290:1–290:20.
- [7] Michael Ashikhmin. “Synthesizing Natural Textures”. In: *Proceedings of the 2001 Symposium on Interactive 3D Graphics - SI3D '01*. The 2001 Symposium. Not Known: ACM Press, 2001, pp. 217–226.

- [8] Alberto Baldrati et al. “Composed Image Retrieval using Contrastive Learning and Task-oriented CLIP-based Features”. In: *ACM Trans. Multim. Comput. Commun. Appl.* 20.3 (2024), 62:1–62:24.
- [9] Jason Baldridge et al. “Imagen 3”. In: *CoRR* abs/2408.07009 (2024). arXiv: [2408.07009](https://arxiv.org/abs/2408.07009).
- [10] Jianmin Bao et al. “CVAE-GAN: Fine-Grained Image Generation through Asymmetric Training”. In: *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*. IEEE Computer Society, 2017, pp. 2764–2773.
- [11] Connelly Barnes and Fang-Lue Zhang. “A Survey of the State-of-the-Art in Patch-Based Synthesis”. In: *Computational Visual Media* 3.1 (Mar. 2017), pp. 3–20.
- [12] Leandro Motta Barros and Soraia Raupp Musse. “Planning Algorithms for Interactive Storytelling”. In: *Comput, Entertain*, (Jan. 2007), 4–es.
- [13] Mikhail Belkin and Partha Niyogi. “Laplacian Eigenmaps for Dimensionality Reduction and Data Representation”. In: *Neural Comput.* 15.6 (2003), pp. 1373–1396.
- [14] Rachele Bellini, Yanir Kleiman, and Daniel Cohen-Or. “Time-Varying Weathering in Texture Space”. In: *ACM Transactions on Graphics* 35.4 (July 2016), pp. 1–11.
- [15] Yoshua Bengio, Aaron C. Courville, and Pascal Vincent. “Representation Learning: A Review and New Perspectives”. In: *IEEE Trans. Pattern Anal. Mach. Intell.* 35.8 (2013), pp. 1798–1828.
- [16] Urs Bergmann, Nikolay Jetchev, and Roland Vollgraf. “Learning Texture Manifolds with the Periodic Spatial GAN”. In: *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*. Ed. by Doina Precup and Yee Whye Teh. Vol. 70. Proceedings of Machine Learning Research. PMLR, 2017, pp. 469–477.
- [17] David Berthelot, Tom Schumm, and Luke Metz. “BEGAN: Boundary Equilibrium Generative Adversarial Networks”. In: *CoRR* abs/1703.10717 (2017). arXiv: [1703.10717](https://arxiv.org/abs/1703.10717).

- [18] Bethesda Game Studios. *Starfield*. Video Game. 2023.
- [19] James Betker et al. “Improving Image Generation with Better Captions”. In.
- [20] Andrew Brock, Jeff Donahue, and Karen Simonyan. “Large Scale GAN Training for High Fidelity Natural Image Synthesis”. In: *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.
- [21] Tom B. Brown et al. “Language Models Are Few-Shot Learners”. In: *Advances in Neural Information Processing Systems 33, NeurIPS 2020, December 6-12, 2020, Virtual*. Ed. by Hugo Larochelle et al. 2020.
- [22] Tom B. Brown et al. “Language Models Are Few-Shot Learners”. In: *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, Virtual*. Ed. by Hugo Larochelle et al. 2020.
- [23] Christopher P. Burgess et al. “Understanding Disentangling in β -VAE”. In: *CoRR* abs/1804.3599 (2018). arXiv: [1804.03599](https://arxiv.org/abs/1804.03599).
- [24] David Cage. *Fahrenheit: Indigo Prophecy*. Video game. 2005.
- [25] Gordon Calleja. *In-Game: From Immersion to Incorporation*. 1st. The MIT Press, 2011.
- [26] Maria Fernanda Caropreso et al. “Template Authoring Environment for the Automatic Generation of Narrative Content”. In: *Journal of Interactive Learning Research* 23.3 (July 2012), pp. 227–249.
- [27] Elin Carstensdottir et al. ““Naked and on Fire” : Examining Player Agency Experiences in Narrative-Focused Gameplay”. In: *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. CHI '21. New York, NY, USA: Association for Computing Machinery, May 2021, pp. 1–13.
- [28] Marc Cavazza, Fred Charles, and Steven J. Mead. “Character-Based Interactive Storytelling”. In: *IEEE Intell. Syst.* 17.4 (2002), pp. 17–24.

- [29] Dave Zhenyu Chen et al. “Text2Tex: Text-driven Texture Synthesis via Diffusion Models”. In: *IEEE/CVF International Conference on Computer Vision, ICCV 2023, Paris, France, October 1-6, 2023*. IEEE, 2023, pp. 18512–18522.
- [30] Qian Chen, Zhu Zhuo, and Wen Wang. “BERT for Joint Intent Classification and Slot Filling”. In: *CoRR abs/1902.10909* (2019). arXiv: [1902.10909](https://arxiv.org/abs/1902.10909).
- [31] Ting Chen et al. “A Simple Framework for Contrastive Learning of Visual Representations”. In: *ICML*. Vol. 119. Proceedings of Machine Learning Research. PMLR, 2020, pp. 1597–1607.
- [32] Wengling Chen and James Hays. “SketchyGAN: Towards Diverse and Realistic Sketch to Image Synthesis”. In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Salt Lake City, UT: IEEE, June 2018, pp. 9416–9425.
- [33] Yanbei Chen and Loris Bazzani. “Learning Joint Visual Semantic Matching Embeddings for Language-Guided Retrieval”. In: *ECCV (22)*. Vol. 12367. Lecture Notes in Computer Science. Springer, 2020, pp. 136–152.
- [34] Li-Yu Chen, I-Chao Shen, and Bing-Yu Chen. “Guided Image Weathering Using Image-to-Image Translation”. In: *SIGGRAPH Asia 2021 Technical Communications*. SA '21: SIGGRAPH Asia 2021. Tokyo Japan: ACM, Dec. 2021, pp. 1–4.
- [35] Chuanqi Cheng et al. ““In Dialogues We Learn”: Towards Personalized Dialogue Without Pre-defined Profiles through In-Dialogue Learning”. In: *CoRR abs/2403.03102* (2024). arXiv: [2403.03102](https://arxiv.org/abs/2403.03102).
- [36] Wei-Lin Chiang et al. *Vicuna: An Open-Source Chatbot Impressing GPT-4 with 90%* ChatGPT Quality*. Mar. 2023.
- [37] Mircea Cimpoi et al. “Describing Textures in the Wild”. In: *2014 IEEE Conference on Computer Vision and Pattern Recognition*. 2014 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Columbus, OH, USA: IEEE, June 2014, pp. 3606–3613.

- [38] Mircea Cimpoi et al. “Describing Textures in the Wild”. In: *2014 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2014, Columbus, OH, USA, June 23-28, 2014*. IEEE Computer Society, 2014, pp. 3606–3613.
- [39] Kevin Clark and Priyank Jaini. “Text-to-Image Diffusion Models are Zero Shot Classifiers”. In: *NeurIPS*. 2023.
- [40] Samuel Rhys Cox and Wei Tsang Ooi. “Conversational Interactions with NPCs in LLM-Driven Gaming: Guidelines from a Content Analysis of Player Feedback”. In: *Chatbot Research and Design, CONVERSATIONS 2023*. Vol. 14524. Lecture Notes in Computer Science. Springer, 2023, pp. 167–184.
- [41] Will Crowther and Don Woods. *Colossal Cave Adventure*. Video Game. 1976.
- [42] Hai Dang, Lukas Mecke, and Daniel Buschek. “GANSlider: How Users Control Generative Models for Images Using Multiple Sliders with and without Feedforward Information”. In: *CHI Conference on Human Factors in Computing Systems*. Apr. 2022, pp. 1–15. arXiv: [2202.00965](https://arxiv.org/abs/2202.00965) [cs].
- [43] Niklas Deckers, Julia Peters, and Martin Potthast. *Manipulating Embeddings of Stable Diffusion Prompts (Version 1.1)*. <https://doi.org/10.5281/zenodo.11203881>. Accessed on YYYY-MM-DD. May 2024.
- [44] Jia Deng et al. “ImageNet: A large-scale hierarchical image database”. In: *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009), 20-25 June 2009, Miami, Florida, USA*. IEEE Computer Society, 2009, pp. 248–255.
- [45] Tim Dettmers et al. “LLM.int8(): 8-bit Matrix Multiplication for Transformers at Scale”. In: *CoRR abs/2208.07339 (2022)*. arXiv: [2208.07339](https://arxiv.org/abs/2208.07339).
- [46] Jacob Devlin et al. “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”. In: *Proc. 2019 Conf. NA Chapt.* Ed. by Jill Burstein, Christy Doran, and Thamar Solorio. Association for Computational Linguistics, 2019, pp. 4171–4186.

- [47] Prafulla Dhariwal and Alexander Quinn Nichol. “Diffusion Models Beat GANs on Image Synthesis”. In: *NeurIPS*. 2021, pp. 8780–8794.
- [48] Yuxuan Ding et al. “The CLIP Model is Secretly an Image-to-Prompt Converter”. In: *NeurIPS*. 2023.
- [49] David L. Donoho and Carrie Grimes. “Image Manifolds Which Are Isometric to Euclidean Space”. In: *Journal of Mathematical Imaging and Vision* 23.1 (July 2005), pp. 5–24.
- [50] Dontnod Entertainment. *Life Is Strange*. Video Game. 2015.
- [51] Alexey Dosovitskiy et al. “An Image Is Worth 16x16 Words: Transformers for Image Recognition at Scale”. In: *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.
- [52] Alexei A. Efros and William T. Freeman. “Image Quilting for Texture Synthesis and Transfer”. In: *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques - SIGGRAPH '01*. The 28th Annual Conference. Not Known: ACM Press, 2001, pp. 341–346.
- [53] Michael Elad and Peyman Milanfar. “Style Transfer Via Texture Synthesis”. In: *IEEE Transactions on Image Processing* 26.5 (May 2017), pp. 2338–2351.
- [54] Failbetter Games. *Fallen London*. Browser-based interactive narrative game. 2009. URL: <https://www.fallenlondon.com/>.
- [55] William Fedus, Barret Zoph, and Noam Shazeer. “Switch Transformers: Scaling to Trillion Parameter Models with Simple and Efficient Sparsity”. In: *J. Mach. Learn. Res.* 23 (2022), 120:1–120:39.
- [56] Kevin Frans, Lisa B. Soros, and Olaf Witkowski. “CLIPDraw: Exploring Text-to-Drawing Synthesis through Language-Image Encoders”. In: *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*. Ed. by Sanmi Koyejo et al. 2022.

- [57] Elias Frantar et al. “GPTQ: Accurate Post-Training Quantization for Generative Pre-trained Transformers”. In: *CoRR* abs/2210.17323 (2022). arXiv: [2210.17323](https://arxiv.org/abs/2210.17323).
- [58] Jonas Freiknecht and Wolfgang Effelsberg. “Procedural Generation of Interactive Stories using Language Models”. In: *FDG '20: International Conference on the Foundations of Digital Games, Bugibba, Malta, September 15-18, 2020*. Ed. by Georgios N. Yannakakis et al. ACM, 2020, 97:1–97:8.
- [59] Anna Frühstück, Ibraheem Alhashim, and Peter Wonka. “TileGAN: Synthesis of Large-Scale Non-Homogeneous Textures”. In: *ACM Transactions on Graphics* 38.4 (Aug. 2019), pp. 1–11. arXiv: [1904.12795](https://arxiv.org/abs/1904.12795) [cs].
- [60] Roberto Gallotta et al. “Large Language Models and Games: A Survey and Roadmap”. In: *CoRR* abs/2402.18659 (2024). arXiv: [2402.18659](https://arxiv.org/abs/2402.18659).
- [61] Roberto Gallotta et al. “Large Language Models and Games: A Survey and Roadmap”. In: *CoRR* abs/2402.18659 (2024). arXiv: [2402.18659](https://arxiv.org/abs/2402.18659).
- [62] Supermassive Games. *Until Dawn*. Video Game. 2015.
- [63] Jacob Garbe et al. “StoryAssembler: an engine for generating dynamic choice-driven narratives”. In: *FDG*. ACM, 2019, 24:1–24:10.
- [64] Pablo García-Sánchez. “Georgios N. Yannakakis and Julian Togelius: Artificial Intelligence and Games - Springer, 2018, Print ISBN: 978-3-319-63518-7, Online ISBN: 978-3-319-63519-4, <https://doi.org/10.1007/978-3-319-63519-4>”. In: *Genet. Program. Evolvable Mach.* 20.1 (2019), pp. 143–145.
- [65] Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. “Texture Synthesis Using Convolutional Neural Networks”. In: *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*. Ed. by Corinna Cortes et al. 2015, pp. 262–270.

- [66] Dibya Ghosh, Abhishek Gupta, and Sergey Levine. “Learning Actionable Representations with Goal Conditioned Policies”. In: *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.
- [67] Loris Giulivi and Giacomo Boracchi. “Concept Visualization: Explaining the CLIP Multi-modal Embedding Using WordNet”. In: *International Joint Conference on Neural Networks, IJCNN 2024, Yokohama, Japan, June 30 - July 5, 2024*. IEEE, 2024, pp. 1–9.
- [68] Lore Goetschalckx et al. “GANalyze: Toward Visual Definitions of Cognitive Image Properties”. In: *ICCV*. IEEE, 2019, pp. 5743–5752.
- [69] Ian J. Goodfellow et al. “Generative Adversarial Networks”. In: *Commun. ACM* 63.11 (2020), pp. 139–144.
- [70] Shuyang Gu et al. “Vector Quantized Diffusion Model for Text-to-Image Synthesis”. In: *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). New Orleans, LA, USA: IEEE, June 2022, pp. 10686–10696.
- [71] P. Guehl et al. “Semi-Procedural Textures Using Point Process Texture Basis Functions”. In: *Computer Graphics Forum* 39.4 (July 2020), pp. 159–171.
- [72] Julia Guerrero-Viu et al. “TexSliders: Diffusion-Based Texture Editing in CLIP Space”. In: *ACM SIGGRAPH 2024 Conference Papers, SIGGRAPH 2024, Denver, CO, USA, 27 July 2024- 1 August 2024*. Ed. by Andres Burbano, Denis Zorin, and Wojciech Jarosz. ACM, 2024, p. 7.
- [73] Ishaan Gulrajani et al. “Improved Training of Wasserstein GANs”. In: *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*. Ed. by Isabelle Guyon et al. 2017, pp. 5767–5777.

- [74] Kamal Gupta, Saurabh Singh, and Abhinav Shrivastava. “PatchVAE: Learning Local Latent Codes for Recognition”. In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*. Computer Vision Foundation / IEEE, 2020, pp. 4745–4754.
- [75] Shir Gur, Sagie Benaim, and Lior Wolf. “Hierarchical Patch VAE-GAN: Generating Diverse Videos from a Single Sample”. In: *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*. Ed. by Hugo Larochelle et al. 2020.
- [76] Niv Haim et al. “Diverse Generation from a Single Video Made Possible”. In: *Computer Vision - ECCV 2022 - 17th European Conference, Tel Aviv, Israel, October 23-27, 2022, Proceedings, Part XVII*. Ed. by Shai Avidan et al. Vol. 13677. Lecture Notes in Computer Science. Springer, 2022, pp. 491–509.
- [77] Jun Han et al. *Disentangled Recurrent Wasserstein Autoencoder*. Jan. 2021. arXiv: [2101.07496 \[cs\]](https://arxiv.org/abs/2101.07496).
- [78] Erik Härkönen et al. *GANSpace: Discovering Interpretable GAN Controls*. Dec. 14, 2020. arXiv: [2004.02546 \[cs\]](https://arxiv.org/abs/2004.02546).
- [79] Erik Härkönen et al. “GANSpace: Discovering Interpretable GAN Controls”. In: *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, Virtual*. Ed. by Hugo Larochelle et al. 2020.
- [80] Paul Harrison. “Image Texture Tools”. PhD thesis. Monash University, 2005.
- [81] Kaiming He et al. “Deep Residual Learning for Image Recognition”. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*. IEEE Computer Society, 2016, pp. 770–778.
- [82] Aaron Hertzmann et al. “Image Analogies”. In: *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques - SIGGRAPH '01*. The 28th Annual Conference. Not Known: ACM Press, 2001, pp. 327–340.

- [83] Martin Heusel et al. “GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium”. In: *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*. Ed. by Isabelle Guyon et al. 2017, pp. 6626–6637.
- [84] Irina Higgins et al. “beta-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework”. In: *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017.
- [85] Tobias Hinz et al. “Improved Techniques for Training Single-Image GANs”. In: *IEEE Winter Conference on Applications of Computer Vision, WACV 2021, Waikoloa, HI, USA, January 3-8, 2021*. IEEE, 2021, pp. 1299–1308.
- [86] Jonathan Ho, Ajay Jain, and Pieter Abbeel. “Denoising Diffusion Probabilistic Models”. In: *NeurIPS*. 2020.
- [87] Sepp Hochreiter and Jürgen Schmidhuber. “Long Short-Term Memory”. In: *Neural Comput.* 9.8 (1997), pp. 1735–1780.
- [88] Md Zakir Hossain et al. *A Comprehensive Survey of Deep Learning for Image Captioning*. Oct. 2018. arXiv: [1810.04020](https://arxiv.org/abs/1810.04020) [cs, stat].
- [89] Jie Huang and Kevin Chen-Chuan Chang. “Towards Reasoning in Large Language Models: A Survey”. In: *Findings of the Association for Computational Linguistics: ACL 2023, Toronto, Canada, July 9-14, 2023*. Ed. by Anna Rogers, Jordan L. Boyd-Graber, and Naoaki Okazaki. Association for Computational Linguistics, 2023, pp. 1049–1065.
- [90] Xun Huang et al. *Multimodal Unsupervised Image-to-Image Translation*. Aug. 2018. arXiv: [1804.04732](https://arxiv.org/abs/1804.04732) [cs, stat].
- [91] Yunpeng Huang et al. “Advancing Transformer Architecture in Long-Context Large Language Models: A Comprehensive Survey”. In: *CoRR abs/2311.12351 (2023)*. arXiv: [2311.12351](https://arxiv.org/abs/2311.12351).

- [92] D. H. Hubel and T. N. Wiesel. “Receptive Fields of Single Neurones in the Cat’s Striate Cortex”. In: *The Journal of Physiology* 148.3 (1959), pp. 574–591.
- [93] Minyoung Huh et al. *Transforming and Projecting Images into Class-conditional Generative Networks*. Aug. 2020. arXiv: [2005.01703](https://arxiv.org/abs/2005.01703) [cs].
- [94] Infocom. *Zork*. Video Game. 1980.
- [95] Phillip Isola et al. *Image-to-Image Translation with Conditional Adversarial Networks*. Nov. 2018. arXiv: [1611.07004](https://arxiv.org/abs/1611.07004) [cs].
- [96] Ali Jahanian, Lucy Chai, and Phillip Isola. “On the "Steerability" of Generative Adversarial Networks”. In: *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.
- [97] Deok-Kyeong Jang, Soomin Park, and Sung-Hee Lee. “Motion Puzzle: Arbitrary Motion Style Transfer by Body Part”. In: *ACM Transactions on Graphics* 41.3 (June 2022), pp. 1–16.
- [98] Nikolay Jetchev, Urs Bergmann, and Roland Vollgraf. *Texture Synthesis with Spatial Generative Adversarial Networks*. Sept. 2017. arXiv: [1611.08207](https://arxiv.org/abs/1611.08207) [cs, stat].
- [99] Shaoxiong Ji et al. “A Survey on Knowledge Graphs: Representation, Acquisition, and Applications”. In: *IEEE Trans. Neural Networks Learn. Syst.* 33.2 (2022), pp. 494–514.
- [100] Chao Jia et al. “Scaling up Visual and Vision-Language Representation Learning with Noisy Text Supervision”. In: *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*. Ed. by Marina Meila and Tong Zhang. Vol. 139. Proceedings of Machine Learning Research. PMLR, 2021, pp. 4904–4916.
- [101] Siyu Jiao et al. “Learning Mask-aware CLIP Representations for Zero-Shot Segmentation”. In: *NeurIPS*. 2023.

- [102] Yongcheng Jing et al. “Stroke Controllable Fast Style Transfer with Adaptive Receptive Fields”. In: *Computer Vision – ECCV 2018*. Ed. by Vittorio Ferrari et al. Vol. 11217. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2018, pp. 244–260.
- [103] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. *Perceptual Losses for Real-Time Style Transfer and Super-Resolution*. Mar. 2016. arXiv: [1603.08155](https://arxiv.org/abs/1603.08155) [cs].
- [104] Alexia Jolicoeur-Martineau. “The Relativistic Discriminator: A Key Element Missing from Standard GAN”. In: *International Conference on Learning Representations (ICLR)*. 2019.
- [105] Bela Julesz. “Experiments in the Visual Perception of Texture”. In: *Scientific American* 232.4 (Apr. 1975), pp. 34–43.
- [106] Kyoungkook Kang, Seongtae Kim, and Sunghyun Cho. “GAN Inversion for Out-of-Range Images with Geometric Transformations”. In: *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*. 2021 IEEE/CVF International Conference on Computer Vision (ICCV). Montreal, QC, Canada: IEEE, Oct. 2021, pp. 13921–13929.
- [107] Mubbasir Kapadia et al. “Computer-Assisted Authoring of Interactive Narratives”. In: *Proceedings of the 19th Symposium on Interactive 3D Graphics and Games, San Francisco, CA, USA, February 27 - March 01, 2015*. Ed. by John Keyser et al. ACM, 2015, pp. 85–92.
- [108] Veli-Matti Karhulahti. “Fiction Puzzle: Storable Challenge in Pragmatist Videogame Aesthetics”. In: *Philos. Technol.* 27.2 (June 2014), pp. 201–220.
- [109] Animesh Karnewar and Oliver Wang. “MSG-GAN: Multi-Scale Gradients for Generative Adversarial Networks”. In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*. Computer Vision Foundation / IEEE, 2020, pp. 7796–7805.

- [110] Vladimir Karpukhin et al. “Dense Passage Retrieval for Open-Domain Question Answering”. In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*. Ed. by Bonnie Webber et al. Association for Computational Linguistics, 2020, pp. 6769–6781.
- [111] Tero Karras, Samuli Laine, and Timo Aila. “A Style-Based Generator Architecture for Generative Adversarial Networks”. In: *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*. Computer Vision Foundation / IEEE, 2019, pp. 4401–4410.
- [112] Tero Karras et al. “Alias-Free Generative Adversarial Networks”. In: (), p. 31.
- [113] Tero Karras et al. “Analyzing and Improving the Image Quality of StyleGAN”. In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*. Computer Vision Foundation / IEEE, 2020, pp. 8107–8116.
- [114] Tero Karras et al. “Analyzing and Improving the Image Quality of StyleGAN”. In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Seattle, WA, USA: IEEE, June 2020, pp. 8107–8116.
- [115] Tero Karras et al. “Progressive Growing of GANs for Improved Quality, Stability, and Variation”. In: *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018.
- [116] Tero Karras et al. “Training Generative Adversarial Networks with Limited Data”. In: *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*. Ed. by Hugo Larochelle et al. 2020.
- [117] Bobak T. Kiani, Jason Wang, and Melanie Weber. “Hardness of Learning Neural Networks under the Manifold Hypothesis”. In: *CoRR abs/2406.01461 (2024)*. arXiv: [2406.01461](https://arxiv.org/abs/2406.01461).

- [118] Hansoo Kim et al. “Edge-Based Procedural Textures”. In: *The Visual Computer* 37.9-11 (Sept. 2021), pp. 2595–2606.
- [119] Diederik P. Kingma and Max Welling. “An Introduction to Variational Autoencoders”. In: *Found. Trends Mach. Learn.* 12.4 (2019), pp. 307–392.
- [120] Diederik P. Kingma and Max Welling. “Auto-Encoding Variational Bayes”. In: *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*. Ed. by Yoshua Bengio and Yann LeCun. 2014.
- [121] Diederik P. Kingma and Max Welling. “Auto-Encoding Variational Bayes”. In: *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*. Ed. by Yoshua Bengio and Yann LeCun. 2014.
- [122] Thomas N. Kipf and Max Welling. “Semi-Supervised Classification with Graph Convolutional Networks”. In: *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017.
- [123] Christoph Klimmt, Tilo Hartmann, and Andreas Frey. “Effectance and Control as Determinants of Video Game Enjoyment”. In: *Cyberpsychology Behav. Soc. Netw.* 10.6 (2007), pp. 845–848.
- [124] Max Kreminski and Noah Wardrip-Fruin. “Sketching a Map of the Storylets Design Space”. In: *ICIDS*. Vol. 11318. Lecture Notes in Computer Science. Springer, 2018, pp. 160–164.
- [125] Solomon Kullback and Richard A. Leibler. “On Information and Sufficiency”. In: *The Annals of Mathematical Statistics* 22.1 (1951), pp. 79–86.
- [126] Vivek Kwatra et al. “Graphcut textures: image and video synthesis using graph cuts”. In: *ACM Trans. Graph.* 22.3 (2003), pp. 277–286.

- [127] Woosuk Kwon et al. “Efficient Memory Management for Large Language Model Serving with PagedAttention”. In: *Proceedings of the 29th Symposium on Operating Systems Principles, SOSP 2023, Koblenz, Germany, October 23-26, 2023*. Ed. by Jason Flinn et al. ACM, 2023, pp. 611–626.
- [128] Anders Boesen Lindbo Larsen et al. “Autoencoding beyond pixels using a learned similarity metric”. In: *Proceedings of the 33rd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*. Ed. by Maria-Florina Balcan and Kilian Q. Weinberger. Vol. 48. JMLR Workshop and Conference Proceedings. JMLR.org, 2016, pp. 1558–1566.
- [129] Latitude. *AI Dungeon*. <https://play.aidungeon.io>. Interactive storytelling game powered by AI. 2019.
- [130] Yann LeCun et al. “Gradient-Based Learning Applied to Document Recognition”. In: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324.
- [131] Hsin-Ying Lee et al. *DRIT++: Diverse Image-to-Image Translation via Disentangled Representations*. Dec. 2019. arXiv: [1905.01270](https://arxiv.org/abs/1905.01270) [cs].
- [132] J. Lessard. *A Tough Sell*. LabLabLab, 2014.
- [133] Jonathan Lessard. “Design Rationale for Natural Language Game Conversations”. In: *Proc. of the 10th International Conference on the Foundations of Digital Games, FDG 2015, Pacific Grove, CA, USA, June 22-25, 2015*. Ed. by José P. Zagal, Esther MacCallum-Stewart, and Julian Togelius. 2015.
- [134] Jonathan Lessard. “Designing Natural-Language Game Conversations”. In: *Proc, DiGRA-FDG* (2016).
- [135] Chenliang Li et al. “Addressing Semantic Drift in Generative Question Answering with Auxiliary Extraction”. In: *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 2: Short Papers), Virtual Event, August 1-6, 2021*. Ed. by Chengqing Zong et al. Association for Computational Linguistics, 2021, pp. 942–947.

- [136] Chuan Li and Michael Wand. *Precomputed Real-Time Texture Synthesis with Markovian Generative Adversarial Networks*. Apr. 2016. arXiv: [1604.04382](https://arxiv.org/abs/1604.04382) [cs].
- [137] Jiwei Li et al. “A Persona-Based Neural Conversation Model”. In: *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*. The Association for Computer Linguistics, 2016.
- [138] Jiwei Li et al. “Adversarial Learning for Neural Dialogue Generation”. In: *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*. Ed. by Martha Palmer, Rebecca Hwa, and Sebastian Riedel. Association for Computational Linguistics, 2017, pp. 2157–2169.
- [139] Junnan Li et al. “BLIP: Bootstrapping Language-Image Pre-training for Unified Vision-Language Understanding and Generation”. In: *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*. Ed. by Kamalika Chaudhuri et al. Vol. 162. Proceedings of Machine Learning Research. PMLR, 2022, pp. 12888–12900.
- [140] Wei Li et al. “DeCap: Decoding CLIP Latents for Zero-Shot Captioning via Text-Only Training”. In: *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023.
- [141] Yijun Li et al. “Diversified Texture Synthesis with Feed-Forward Networks”. In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Honolulu, HI: IEEE, July 2017, pp. 266–274.
- [142] Lin Liang et al. “Real-Time Texture Synthesis by Patch-Based Sampling”. In: *ACM Transactions on Graphics* 20.3 (July 2001), pp. 127–150.
- [143] Lizi Liao et al. “Dialogue State Tracking with Incremental Reasoning”. In: *Trans. Assoc. Comput. Linguistics* 9 (2021), pp. 557–569.
- [144] Jae Hyun Lim and Jong Chul Ye. “Geometric GAN”. In: *Advances in Neural Information Processing Systems (NeurIPS)*. Vol. 30. 2017, pp. 5247–5257.

- [145] Bingchen Liu, Kunpeng Song, and Ahmed Elgammal. *Sketch-to-Art: Synthesizing Stylized Art Images From Sketches*. Oct. 2020. arXiv: [2002.12888 \[cs, eess\]](#).
- [146] Pengfei Liu et al. “Pre-Train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing”. In: *Acm Computing Surveys* 55.9 (2023), 195:1–195:35.
- [147] Songhua Liu et al. *AdaAttN: Revisit Attention Mechanism in Arbitrary Neural Style Transfer*. Aug. 2021. arXiv: [2108.03647 \[cs\]](#).
- [148] Yang Liu, Peng Sun, and Hang Li. “Large Language Models as Agents in Two-Player Games”. In: *CoRR* abs/2402.08078 (2024). arXiv: [2402.08078](#).
- [149] Yitzchak David Lockerman et al. “Multi-Scale Label-Map Extraction for Texture Synthesis”. In: *ACM Transactions on Graphics* 35.4 (July 2016), pp. 1–12.
- [150] Ilya Loshchilov and Frank Hutter. “Decoupled Weight Decay Regularization”. In: *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.
- [151] Xudong Mao et al. “Least Squares Generative Adversarial Networks”. In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. 2017, pp. 2794–2802.
- [152] Stacey Mason, Ceri Stagg, and Noah Wardrip-Fruin. “Lume: a system for procedural story generation”. In: *FDG*. ACM, 2019, 15:1–15:9.
- [153] Michael Mateas and Andrew Stern. *Facade*. Video Game. 2005.
- [154] Michael Mateas and Andrew Stern. “Façade: An Experiment in Building a Fully-Realized Interactive Drama”. In: *Game Developers Conference (GDC’03)*. 2003.
- [155] Leland McInnes and John Healy. “UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction”. In: *CoRR* abs/1802.03426 (2018). arXiv: [1802.03426](#).
- [156] Michael J. McMains and Frederick J. Lanceley. “The Use of Crisis Intervention Principles by Police Negotiators”. In: *Journal of Police Crisis Negotiations* 3.1 (2003), pp. 3–30.

- [157] Tomas Mikolov et al. “Distributed Representations of Words and Phrases and Their Compositionality”. In: *Advances in Neural Information Processing Systems, Lake Tahoe, Nevada, United States*. Ed. by Christopher J. C. Burges et al. 2013, pp. 3111–3119.
- [158] Ian Millington and John Funge. *Artificial Intelligence for Games, Second Edition*. Morgan Kaufmann, 2009.
- [159] Muhammad Jehanzeb Mirza et al. “LaFTer: Label-Free Tuning of Zero-shot Classifier using Language and Unlabeled Image Collections”. In: *NeurIPS*. 2023.
- [160] Takeru Miyato et al. “Spectral Normalization for Generative Adversarial Networks”. In: *International Conference on Learning Representations (ICLR)*. 2018.
- [161] Nick Montfort. *Twisty Little Passages: An Approach to Interactive Fiction*. MIT Press, 2005.
- [162] Jose Luis Morales and Jorge Nocedal. “Remark on "algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound constrained optimization"”. In: *ACM Trans. Math. Softw.* 38.1 (2011), 7:1–7:4.
- [163] Nikola Mrksic et al. “Multi-Domain Dialog State Tracking Using Recurrent Neural Networks”. In: *Proc. of the 53rd Annual Meeting of the Association for Computational Linguistics ACL 2015, Beijing, China*. The Association for Computer Linguistics, 2015, pp. 794–799.
- [164] J. Murray. *Hamlet on the Holodeck: The Future of Narrative in Cyberspace. the Future of Narrative in Cyberspace*. The MIT Press, 2017, p. 324.
- [165] Jinjie Ni et al. “Recent Advances in Deep Learning Based Dialogue Systems: A Systematic Survey”. In: *Artificial Intelligence Review* 56.4 (2023), pp. 3055–3155.
- [166] Alexander Quinn Nichol and Prafulla Dhariwal. “Improved Denoising Diffusion Probabilistic Models”. In: *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*. Ed. by Marina Meila and Tong Zhang. Vol. 139. Proceedings of Machine Learning Research. PMLR, 2021, pp. 8162–8171.

- [167] Yotam Nitzan et al. “Face Identity Disentanglement via Latent Space Mapping”. In: *ACM Trans, Graph*, 39.6 (2020), 225:1–225:14.
- [168] OpenAI. *ChatGPT: Chat Generative Pre-trained Transformer (Version 4.0)*. <https://openai.com/chatgpt>. Accessed: YYYY-MM-DD. 2024.
- [169] OpenAI. “GPT-4 Technical Report”. In: *Front. Psychol.* (Mar. 2023). arXiv: [2303.08774](https://arxiv.org/abs/2303.08774) [cs.CL].
- [170] Long Ouyang et al. “Training Language Models to Follow Instructions with Human Feedback”. In: *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*. Ed. by Sanmi Koyejo et al. 2022.
- [171] Long Ouyang et al. “Training Language Models to Follow Instructions with Human Feedback”. In: *NeurIPS*. 2022.
- [172] Taesung Park et al. *Semantic Image Synthesis with Spatially-Adaptive Normalization*. Nov. 2019. arXiv: [1903.07291](https://arxiv.org/abs/1903.07291) [cs].
- [173] Gaurav Parmar et al. “Spatially-Adaptive Multilayer Selection for GAN Inversion and Editing”. In: *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). New Orleans, LA, USA: IEEE, June 2022, pp. 11389–11399.
- [174] Or Patashnik et al. “StyleCLIP: Text-Driven Manipulation of StyleGAN Imagery”. In: *2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021*. IEEE, 2021, pp. 2065–2074.
- [175] Xiangyu Peng et al. “Player-Driven Emergence in LLM-Driven Game Narrative”. In: *IEEE Conference on Games, CoG 2024, Milan, Italy, August 5-8, 2024*. IEEE, 2024, pp. 1–8.

- [176] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. “Glove: Global Vectors for Word Representation”. In: *Proc. of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014*. ACL, 2014, pp. 1532–1543.
- [177] Andrew Plotkin. *Rule-Based Programming in Interactive Fiction*. Web Page. 2023.
- [178] Robert Plutchik. “A Psychoevolutionary Theory of Emotion.” In: *Soc. Sc. Inform.* 21.4-5 (1982), pp. 529–553.
- [179] Dustin Podell et al. “SDXL: Improving Latent Diffusion Models for High-Resolution Image Synthesis”. In: *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024.
- [180] Phillip Pope et al. “The Intrinsic Dimension of Images and Its Impact on Learning”. In: *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.
- [181] Javier Portilla and Eero P. Simoncelli. “A Parametric Texture Model Based on Joint Statistics of Complex Wavelet Coefficients”. In: *International Journal of Computer Vision* 40.1 (2000), pp. 49–70.
- [182] Reid Pryzant et al. “Automatic Prompt Optimization with “Gradient Descent” and Beam Search”. In: *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. Singapore: Association for Computational Linguistics, 2023, pp. 7957–7968.
- [183] Guanghai Qin and Jason Eisner. “Learning How to Ask: Querying LMs with Mixtures of Soft Prompts”. In: *NAACL-HLT*. Association for Computational Linguistics, 2021, pp. 5203–5212.
- [184] Alec Radford et al. “Language Models Are Unsupervised Multitask Learners”. In: *OpenAI blog* (2019).
- [185] Alec Radford et al. “Learning Transferable Visual Models From Natural Language Supervision”. In: *ICML*. Vol. 139. Proceedings of Machine Learning Research. PMLR, 2021, pp. 8748–8763.

- [186] Aditya Ramesh et al. *Hierarchical Text-Conditional Image Generation with CLIP Latents*. Apr. 2022. arXiv: [2204.06125 \[cs\]](#).
- [187] Aditya Ramesh et al. *Zero-Shot Text-to-Image Generation*. Feb. 2021. arXiv: [2102.12092 \[cs\]](#).
- [188] Aditya Ramesh et al. “Zero-Shot Text-to-Image Generation”. In: *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*. Ed. by Marina Meila and Tong Zhang. Vol. 139. Proceedings of Machine Learning Research. PMLR, 2021, pp. 8821–8831.
- [189] Scott Reed et al. *Generative Adversarial Text to Image Synthesis*. June 2016. arXiv: [1605.05396 \[cs\]](#).
- [190] Laria Reynolds and Kyle McDonell. “Prompt Programming for Large Language Models: Beyond the Few-Shot Paradigm”. In: *CHI '21: CHI Conference on Human Factors in Computing Systems, Virtual Event / Yokohama Japan, May 8-13, 2021, Extended Abstracts*. Ed. by Yoshifumi Kitamura et al. ACM, 2021, 314:1–314:7.
- [191] K. Riach and S. Stuard. *Oxenfree*. Video Game. 2016.
- [192] Elad Richardson et al. “Encoding in Style: A StyleGAN Encoder for Image-to-Image Translation”. In: *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). Nashville, TN, USA: IEEE, June 2021, pp. 2287–2296.
- [193] Mark O. Riedl and Robert Michael Young. “Narrative Planning: Balancing Plot and Character”. In: *J. Artif. Intell. Res.* 39 (2010), pp. 217–268.
- [194] Eric Risser, Pierre Wilmot, and Connelly Barnes. *Stable and Controllable Neural Texture Synthesis and Style Transfer Using Histogram Losses*. Feb. 2017. arXiv: [1701.08893 \[cs\]](#).

- [195] Alan Ritter, Colin Cherry, and William B. Dolan. “Data-Driven Response Generation in Social Media”. In: *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, EMNLP 2011, 27-31 July 2011, John McIntyre Conference Centre, Edinburgh, UK, A Meeting of SIGDAT, a Special Interest Group of the ACL*. ACL, 2011, pp. 583–593.
- [196] Robin Rombach et al. “High-Resolution Image Synthesis with Latent Diffusion Models”. In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*. IEEE, 2022, pp. 10674–10685.
- [197] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. “U-Net: Convolutional Networks for Biomedical Image Segmentation”. In: *Medical Image Computing and Computer-Assisted Intervention - MICCAI 2015 - 18th International Conference Munich, Germany, October 5 - 9, 2015, Proceedings, Part III*. Ed. by Nassir Navab et al. Vol. 9351. Lecture Notes in Computer Science. Springer, 2015, pp. 234–241.
- [198] Chitwan Saharia et al. “Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding”. In: *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*. Ed. by Sanmi Koyejo et al. 2022.
- [199] Katie Salen Tekinbaş and Eric Zimmerman. *Rules of play. game design fundamentals*. Katie Salen and Eric Zimmerman. Cambridge, Mass., 2004.
- [200] Patsorn Sangkloy et al. *Scribbler: Controlling Deep Image Synthesis with Sketch and Color*. Dec. 2016. arXiv: [1612.00835 \[cs\]](https://arxiv.org/abs/1612.00835).
- [201] Anurag Sarkar and Seth Cooper. “Procedural Content Generation using Behavior Trees (PCGBT)”. In: *Joint Proc. of the AIIDE 2021 Workshops, Oct. 11-12, 2021*. Ed. by Brent Harrison. Vol. 3217. CEUR Workshop Proc. CEUR-WS.org, 2021.
- [202] Franco Scarselli et al. “The Graph Neural Network Model”. In: *IEEE Trans. Neural Networks* 20.1 (2009), pp. 61–80.

- [203] Christoph Schuhmann et al. “LAION-5B: An open large-scale dataset for training next generation image-text models”. In: *NeurIPS*. 2022.
- [204] Yoones A. Sekhavat. “Behavior Trees for Computer Games”. In: *Int. J. Artif. Intell. Tools* 26.2 (2017), 1730001:1–1730001:28.
- [205] Omry Sendik and Daniel Cohen-Or. “Deep Correlations for Texture Synthesis”. In: *ACM Trans. Graph.* 36.5 (2017), 161:1–161:15.
- [206] Iulian Vlad Serban et al. “A Hierarchical Latent Variable Encoder-Decoder Model for Generating Dialogues”. In: *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*. Ed. by Satinder Singh and Shaul Markovitch. AAAI Press, 2017, pp. 3295–3301.
- [207] Tamar Rott Shaham, Tali Dekel, and Tomer Michaeli. “SinGAN: Learning a Generative Model From a Single Natural Image”. In: *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*. IEEE, 2019, pp. 4569–4579.
- [208] Mingyue Shang et al. “Learning to Converse with Noisy Data: Generation with Calibration”. In: *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*. Ed. by Jérôme Lang. ijcai.org, 2018, pp. 4338–4344.
- [209] Hang Shao, Abhishek Kumar, and P. Thomas Fletcher. “The Riemannian Geometry of Deep Generative Models”. In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. Salt Lake City, UT, USA: IEEE, June 2018, pp. 428–4288.
- [210] Sheng Shen et al. “How Much Can Clip Benefit Vision-and- Language Tasks?” In: (2022).
- [211] Yujun Shen et al. *Interpreting the Latent Space of GANs for Semantic Face Editing*. Mar. 2020. arXiv: [1907.10786](https://arxiv.org/abs/1907.10786) [cs].

- [212] Yujun Shen et al. “Interpreting the Latent Space of GANs for Semantic Face Editing”. In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Seattle, WA, USA: IEEE, June 2020, pp. 9240–9249.
- [213] Wu Shi and Yu Qiao. “Fast Texture Synthesis via Pseudo Optimizer”. In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). Seattle, WA, USA: IEEE, June 2020, pp. 5497–5506.
- [214] Assaf Shocher et al. “InGAN: Capturing and Retargeting the "DNA" of a Natural Image”. In: *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*. IEEE, 2019, pp. 4491–4500.
- [215] Ken Shoemake. “Animating rotation with quaternion curves”. In: *Proceedings of the 12th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 1985, San Francisco, California, USA, July 22-26, 1985*. Ed. by Pat Cole, Robert Heilman, and Brian A. Barsky. ACM, 1985, pp. 245–254.
- [216] Emily Short. *Blood & Laurels*. Video game. 2014.
- [217] Connor Shorten and Taghi M. Khoshgoftaar. “A Survey on Image Data Augmentation for Deep Learning”. In: *Journal of Big Data* 6.1 (Dec. 2019), p. 60.
- [218] Alon Shoshan et al. *GAN-Control: Explicitly Controllable GANs*. Oct. 2021. arXiv: [2101.02477 \[cs\]](https://arxiv.org/abs/2101.02477).
- [219] Karen Simonyan and Andrew Zisserman. “Very Deep Convolutional Networks for Large-Scale Image Recognition”. In: *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*. Ed. by Yoshua Bengio and Yann LeCun. 2015.
- [220] Xavier Snelgrove. “High-Resolution Multi-Scale Neural Texture Synthesis”. In: *SIGGRAPH Asia 2017 Technical Briefs*. SA '17: SIGGRAPH Asia 2017. Bangkok Thailand: ACM, Nov. 2017, pp. 1–4.
- [221] Ocelot Society. *Event[0]*. Video Game. 2016.

- [222] Jiaming Song, Chenlin Meng, and Stefano Ermon. “Denoising Diffusion Implicit Models”. In: *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.
- [223] Yiren Song. “CLIPTexture: Text-Driven Texture Synthesis”. In: *ACM Multimedia*. ACM, 2022, pp. 5468–5476.
- [224] Akash Srivastava et al. “VEEGAN: Reducing Mode Collapse in GANs using Implicit Variational Learning”. In: *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*. Ed. by Isabelle Guyon et al. 2017, pp. 3308–3318.
- [225] Nisan Stiennon et al. “Learning to Summarize from Human Feedback”. In: *CoRR* abs/2009.01325 (2020). arXiv: [2009.01325](https://arxiv.org/abs/2009.01325).
- [226] Christina R. Strong and Michael Mateas. “Talking with NPCs: Towards Dynamic Generation of Discourse Structures”. In: *Proceedings of the Fourth Artificial Intelligence and Interactive Digital Entertainment Conference, October 22-24, 2008, Stanford, California, USA*. Ed. by Christian Darken and Michael Mateas. The AAAI Press, 2008.
- [227] Chen Sun et al. “Revisiting Unreasonable Effectiveness of Data in Deep Learning Era”. In: *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*. IEEE Computer Society, 2017, pp. 843–852.
- [228] Yuqian Sun et al. “Drama Llama: An LLM-Powered Storylets Framework for Authorable Responsiveness in Interactive Narrative”. In: *CoRR* abs/2501.09099 (2025).
- [229] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. “Sequence to Sequence Learning with Neural Networks”. In: *Advances in Neural Information Processing Systems 27, December 8-13 2014, Montreal, Quebec, Canada*. Ed. by Zoubin Ghahramani et al. 2014, pp. 3104–3112.

- [230] Penny Sweetser. “Large Language Models and Video Games: A Preliminary Scoping Review”. In: *ACM Conversational User Interfaces 2024, CUI 2024, Luxembourg, July 8-10, 2024*. Ed. by Mateusz Dubiel et al. ACM, 2024, p. 45.
- [231] Mingxing Tan and Quoc V. Le. “EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks”. In: *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*. Ed. by Kamalika Chaudhuri and Ruslan Salakhutdinov. Vol. 97. Proceedings of Machine Learning Research. PMLR, 2019, pp. 6105–6114.
- [232] Karen Tanenbaum and Theresa Jean Tanenbaum. “Agency as commitment to meaning: communicative competence in games”. In: *Digit. Creativity* 21.1 (2010), pp. 11–17.
- [233] David Thue et al. “Interactive Storytelling: A Player Modelling Approach”. In: *Proceedings of the Third Artificial Intelligence and Interactive Digital Entertainment Conference, June 6-8, 2007, Stanford, California, USA*. Ed. by Jonathan Schaeffer and Michael Mateas. The AAAI Press, 2007, pp. 43–48.
- [234] Julian Togelius et al. “Search-Based Procedural Content Generation: A Taxonomy and Survey”. In: *IEEE Trans. Comput. Intell. AI Games* 3.3 (2011), pp. 172–186.
- [235] Emmett Tomai, David Salinas, and Rosendo Salazar. “A Rule-Based Framework for Modular Development of In-Game Interactive Dialogue Simulation”. In: *Intelligent Narrative Technologies IV, Papers from the 2011 AIIDE Workshop, Stanford, California, USA, October 10-11, 2011*. Vol. WS-11-18. AAAI Technical Report. AAAI, 2011.
- [236] Hugo Touvron et al. “LLaMA: Open and Efficient Foundation Language Models”. In: *CoRR* abs/2302.13971 (2023). arXiv: [2302.13971](https://arxiv.org/abs/2302.13971).
- [237] Luan Tran, Xi Yin, and Xiaoming Liu. “Disentangled Representation Learning GAN for Pose-Invariant Face Recognition”. In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Honolulu, HI: IEEE, July 2017, pp. 1283–1292.

- [238] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. “Improved Texture Networks: Maximizing Quality and Diversity in Feed-Forward Stylization and Texture Synthesis”. In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Honolulu, HI: IEEE, July 2017, pp. 4105–4113.
- [239] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. *Instance Normalization: The Missing Ingredient for Fast Stylization*. Nov. 2017. arXiv: [1607.08022](https://arxiv.org/abs/1607.08022) [cs].
- [240] Dmitry Ulyanov et al. *Texture Networks: Feed-forward Synthesis of Textures and Stylized Images*. Mar. 2016. arXiv: [1603.03417](https://arxiv.org/abs/1603.03417) [cs].
- [241] Susanna Värtinen, Perttu Hämäläinen, and Christian Guckelsberger. “Generating Role-Playing Game Quests With GPT Language Models”. In: *IEEE Trans. Games* 16.1 (2024), pp. 127–139.
- [242] Ashish Vaswani et al. “Attention Is All You Need”. In: *Advances in Neural Information Processing Systems 30, December 4-9, 2017, Long Beach, CA, USA*. Ed. by Isabelle Guyon et al. 2017, pp. 5998–6008.
- [243] Oriol Vinyals and Quoc V. Le. “A Neural Conversational Model”. In: *CoRR* abs/1506.05869 (2015). arXiv: [1506.05869](https://arxiv.org/abs/1506.05869).
- [244] Binxu Wang and Carlos R. Ponce. *The Geometry of Deep Generative Image Models and Its Applications*. Mar. 2021. arXiv: [2101.06006](https://arxiv.org/abs/2101.06006) [cs, math].
- [245] Qingyue Wang et al. “Recursively Summarizing Enables Long-Term Dialogue Memory in Large Language Models”. In: *CoRR* abs/2308.15022 (2023). arXiv: [2308.15022](https://arxiv.org/abs/2308.15022).
- [246] Noah Wardrip-Fruin et al. “Agency Reconsidered”. In: *Proceedings of the 2009 DiGRA International Conference*. Ed. by Tanya Krzywinska, Helen W. Kennedy, and Barry Atkins. Digital Games Research Association, 2009.
- [247] Stephen G. Ware et al. “Multiagent Narrative Experience Management as Story Graph Pruning”. In: *IEEE Trans. Games* 15.3 (2023), pp. 378–387.

- [248] Jason Wei et al. “Chain-of-Thought Prompting Elicits Reasoning in Large Language Models”. In: *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*. Ed. by Sanmi Koyejo et al. 2022.
- [249] Jason Wei et al. “Finetuned Language Models Are Zero-Shot Learners”. In: *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022.
- [250] Tianyi Wei et al. “E2Style: Improve the Efficiency and Effectiveness of StyleGAN Inversion”. In: *IEEE Transactions on Image Processing* 31 (2022), pp. 3267–3280. arXiv: [2104.07661](https://arxiv.org/abs/2104.07661) [cs].
- [251] Li-Yi Wei and Marc Levoy. “Fast Texture Synthesis Using Tree-Structured Vector Quantization”. In: *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques - SIGGRAPH '00*. The 27th Annual Conference. Not Known: ACM Press, 2000, pp. 479–488.
- [252] Nathaniel Weir et al. “Ontologically Faithful Generation of Non-Player Character Dialogues”. In: *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, EMNLP 2024, Miami, FL, USA, November 16-16, 2024*. Ed. by Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen. Association for Computational Linguistics, 2024, pp. 9212–9242.
- [253] Yuxin Wen et al. “Hard Prompts Made Easy: Gradient-Based Discrete Optimization for Prompt Tuning and Discovery”. In: *NeurIPS*. 2023.
- [254] Jason D. Williams, Antoine Raux, and Matthew Henderson. “The Dialog State Tracking Challenge Series: A Review”. In: *Dialogue Discourse* 7.3 (2016), pp. 4–33.
- [255] W. A. Woods. “Transition Network Grammars for Natural Language Analysis”. In: *Commun. ACM* 13 (1970), pp. 591–606.

- [256] Chen Henry Wu and Fernando De La Torre. “A Latent Space of Stochastic Diffusion Models for Zero-Shot Image Editing and Guidance”. In: *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*. Paris, France: IEEE, Oct. 2023, pp. 7344–7353.
- [257] Wenqi Xian et al. *TextureGAN: Controlling Deep Image Synthesis with Texture Patches*. Apr. 14, 2018. arXiv: [1706.02823](https://arxiv.org/abs/1706.02823) [cs].
- [258] Bin Xiao et al. “Florence-2: Advancing a Unified Representation for a Variety of Vision Tasks”. In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2024, Seattle, WA, USA, June 16-22, 2024*. IEEE, 2024, pp. 4818–4829.
- [259] Johnathan Xie and Shuai Zheng. “Zero-Shot Object Detection through Vision-Language Embedding Alignment”. In: *IEEE International Conference on Data Mining Workshops, ICDM 2022 - Workshops, Orlando, FL, USA, November 28 - Dec. 1, 2022*. Ed. by K. Selçuk Candan et al. IEEE, 2022, pp. 1–15.
- [260] Shasha Xie et al. “GAGCN: Generative adversarial graph convolutional network for non-homogeneous texture extension synthesis”. In: *IET Image Process.* 17.5 (2023), pp. 1603–1614.
- [261] Daniel L K Yamins and James J DiCarlo. “Using Goal-Driven Deep Learning Models to Understand Sensory Cortex”. In: *Nature Neuroscience* 19.3 (Feb. 2016), pp. 356–365.
- [262] Shuai Yang et al. “Controllable Artistic Text Style Transfer via Shape-Matching GAN”. In: *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. 2019 IEEE/CVF International Conference on Computer Vision (ICCV). Seoul, Korea (South): IEEE, Oct. 2019, pp. 4441–4450.
- [263] Georgios N. Yannakakis and Julian Togelius. “Experience-Driven Procedural Content Generation”. In: *IEEE Trans. Affect. Comput.* 2.3 (2011), pp. 147–161.
- [264] Georgios N. Yannakakis et al. “Player Modeling”. In: *Artificial and Computational Intelligence in Games*. Ed. by Simon M. Lucas et al. Vol. 6. Dagstuhl Follow-Ups. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2013, pp. 45–59.

- [265] Zihao Yi et al. “A Survey on Recent Advances in LLM-Based Multi-turn Dialogue Systems”. In: *CoRR* abs/2402.18013 (2024). arXiv: [2402.18013](https://arxiv.org/abs/2402.18013).
- [266] Qing Ru Yong and Alex Mitchell. “From Playing the Story to Gaming the System: Repeat Experiences of a Large Language Model-Based Interactive Story”. In: *Interactive Storytelling - 16th International Conference on Interactive Digital Storytelling, ICIDS 2023, Kobe, Japan, November 11-15, 2023, Proceedings, Part I*. Ed. by Lissa Holloway-Attaway and John T. Murray. Vol. 14383. Lecture Notes in Computer Science. Springer, 2023, pp. 395–409.
- [267] Lu Yuan et al. “Florence: A New Foundation Model for Computer Vision”. In: *CoRR* abs/2111.11432 (2021). arXiv: [2111.11432](https://arxiv.org/abs/2111.11432).
- [268] Xiaohua Zhai et al. “Sigmoid Loss for Language Image Pre-Training”. In: *IEEE/CVF International Conference on Computer Vision, ICCV 2023, Paris, France, October 1-6, 2023*. IEEE, 2023, pp. 11941–11952.
- [269] Han Zhang et al. *StackGAN++: Realistic Image Synthesis with Stacked Generative Adversarial Networks*. June 2018. arXiv: [1710.10916](https://arxiv.org/abs/1710.10916) [cs, stat].
- [270] Renrui Zhang et al. “Tip-Adapter: Training-Free Adaption of CLIP for Few-Shot Classification”. In: *ECCV (35)*. Vol. 13695. Lecture Notes in Computer Science. Springer, 2022, pp. 493–510.
- [271] Richard Zhang et al. *The Unreasonable Effectiveness of Deep Features as a Perceptual Metric*. Apr. 2018. arXiv: [1801.03924](https://arxiv.org/abs/1801.03924) [cs].
- [272] Richard Zhang et al. “The Unreasonable Effectiveness of Deep Features as a Perceptual Metric”. In: *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*. Computer Vision Foundation / IEEE Computer Society, 2018, pp. 586–595.
- [273] Zhenxing Zhang and Lambert Schomaker. *Optimized Latent-Code Selection for Explainable Conditional Text-to-Image GANs*. Apr. 2022. arXiv: [2204.12678](https://arxiv.org/abs/2204.12678) [cs].
- [274] Justin Zhao et al. “LoRA Land: 310 Fine-tuned LLMs that Rival GPT-4, A Technical Report”. In: *CoRR* abs/2405.00732 (2024). arXiv: [2405.00732](https://arxiv.org/abs/2405.00732).

- [275] Tiancheng Zhao, Ran Zhao, and Maxine Eskénazi. “Learning Discourse-level Diversity for Neural Dialog Models Using Conditional Variational Autoencoders”. In: *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*. Ed. by Regina Barzilay and Min-Yen Kan. Association for Computational Linguistics, 2017, pp. 654–664.
- [276] Zihao Zhao et al. “Calibrate Before Use: Improving Few-shot Performance of Language Models”. In: *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*. Ed. by Marina Meila and Tong Zhang. Vol. 139. Proceedings of Machine Learning Research. PMLR, 2021, pp. 12697–12706.
- [277] Zihao Zhao et al. “CLIP in Medical Imaging: A Comprehensive Survey”. In: *CoRR* abs/2312.07353 (2023). arXiv: [2312.07353](https://arxiv.org/abs/2312.07353).
- [278] Dongxiao Zhou. “Texture Analysis and Synthesis Using a Generic Markov-Gibbs Image Model”. In: *Online ()*, p. 172.
- [279] Jie Zhou et al. “Graph Neural Networks: A Review of Methods and Applications”. In: *AI Open* 1 (2020), pp. 57–81.
- [280] Kaiyang Zhou et al. “Learning to Prompt for Vision-Language Models”. In: *Int. J. Comput. Vis.* 130.9 (2022), pp. 2337–2348.
- [281] Sharon Zhou et al. *Evaluating the Disentanglement of Deep Generative Models through Manifold Topology*. Mar. 2021. arXiv: [2006.03680](https://arxiv.org/abs/2006.03680) [cs, stat].
- [282] Wei Zhou, Xiangyu Peng, and Mark O. Riedl. “Dialogue Shaping: Empowering Agents through NPC Interaction”. In: *Proceedings of the Experimental Artificial Intelligence in Games Workshop Co-Located with the 19th AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE 2023), Salt Lake City, Utah, USA, October 8, 2023*. Ed. by Abdelrahman Madkour et al. Vol. 3626. CEUR Workshop Proceedings. CEUR-WS.org, 2023.
- [283] Yang Zhou et al. “Analysis and Controlled Synthesis of Inhomogeneous Textures”. In: *Computer Graphics Forum* 36.2 (May 2017), pp. 199–212.

- [284] Yang Zhou et al. “Neural Texture Synthesis with Guided Correspondence”. In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2023, Vancouver, BC, Canada, June 17-24, 2023*. IEEE, 2023, pp. 18095–18104.
- [285] Yang Zhou et al. *Non-Stationary Texture Synthesis by Adversarial Expansion*. May 2018. arXiv: [1805.04487](https://arxiv.org/abs/1805.04487) [cs].
- [286] Andrew Zhu et al. “CALYPSO: LLMs as Dungeon Masters’ Assistants”. In: *CoRR* abs/2308.07540 (2023). arXiv: [2308.07540](https://arxiv.org/abs/2308.07540).
- [287] Jiapeng Zhu et al. “In-Domain GAN Inversion for Real Image Editing”. In: *Computer Vision – ECCV 2020*. Ed. by Andrea Vedaldi et al. Vol. 12362. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2020, pp. 592–608.
- [288] Jun-Yan Zhu et al. “Generative Visual Manipulation on the Natural Image Manifold”. In: *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part V*. Ed. by Bastian Leibe et al. Vol. 9909. Lecture Notes in Computer Science. Springer, 2016, pp. 597–613.
- [289] Jun-Yan Zhu et al. *Toward Multimodal Image-to-Image Translation*. Oct. 2018. arXiv: [1711.11586](https://arxiv.org/abs/1711.11586) [cs, stat].
- [290] Jun-Yan Zhu et al. *Unpaired Image-to-Image Translation Using Cycle-Consistent Adversarial Networks*. Aug. 2020. arXiv: [1703.10593](https://arxiv.org/abs/1703.10593) [cs].
- [291] Daniel M. Ziegler et al. “Fine-Tuning Language Models from Human Preferences”. In: *CoRR* abs/1909.08593 (2019). arXiv: [1909.08593](https://arxiv.org/abs/1909.08593).
- [292] Károly Zsolnai-Fehér, Peter Wonka, and Michael Wimmer. “Gaussian Material Synthesis”. In: *ACM Transactions on Graphics* 37.4 (Aug. 2018), pp. 1–14.

Components of LLM-based Analysis

A.1 Panel of Results shown to LLM Agent

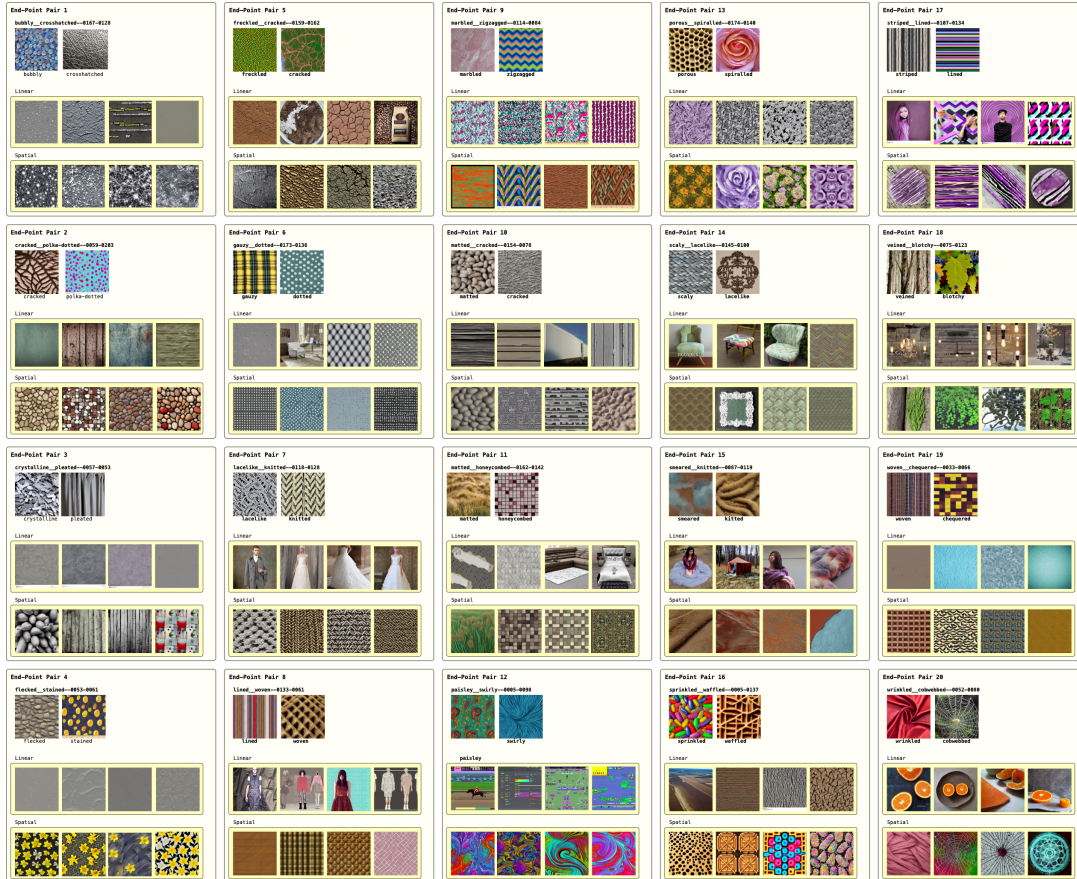


Figure A.1: **Panel of Results.** A set of twenty experiment with randomly sampled end-points and results for each approach. Each experiment is shown to the evaluation agent sequentially.

A.2 Agent Conditioning Prompt

I have an interesting job for you

I will ask you to act in the role of an impartial observer for a user study, where to you will provide an analysis and your opinion of a series of images in two categories. We are conducting an experiment to evaluate two distinct interpolation methods. The goal is to understand their

performance across a number of metrics and form an unbiased overall conclusion about the method. You will see a series of experiments. Each experiment contains two input texture images. Each texture represents particular category or concept. For each experiment there are results for each method - Linear and Spatial. The results are a set of four images sampled from a larger set of images generated by each method. These are the images you will assess. For the assessment you will analyze the images according to the following metrics:

Feature Preservation: * Metric: How well distinctive features of the end-point images are preserved throughout the interpolation. * How to measure: Compare key texture elements (lines, patterns, shapes) between the original images and intermediate frames, assigning a score based on how recognizable these features remain as the interpolation progresses. Perceptual Blending: * Metric: Extent of visual blending between the two textures. * How to measure: Rate the degree to which intermediate images exhibit a harmonious combination of elements from both end-points, ensuring no one texture dominates or disappears too early in the interpolation. Visual Novelty: * Metric: Degree of new, unique textures that emerge during interpolation. * How to measure: Evaluate whether intermediate images introduce novel patterns or combinations not fully present in either of the two end-point images. Higher scores for more creative or unexpected combinations.

For each metric you will perform the analysis to the best of your ability. You will provide a quantitative value for each metric according to your assessment. The values must be provided in a scientific way such that the values are a true representation of each metric for each method. You should perform the assessment in two passes. In the first pass you will get a sense of all the images in the experiment, then you will perform the analysis. The analysis should be “normalized” across all the end-point examples such that the best performing example should receive the highest overall score. For each sub-experiment I will first provide the source images, then the linear samples and finally the spatial samples present the results in a table.

A.3 Raw Agent Analysis

Here we include the raw output from the LLM agent’s analysis of the evaluation set.

End-Point Pair 1: bubbly__crosshatched-0167-0128

I see the first image set for the experiment with the pair "bubbly" and "crosshatched" as the two source textures, followed by the Linear and Spatial interpolation results. I will first conduct a quick pass through all images and then provide my analysis.

First Pass - General Observations

- **Source Images:**

- **Bubbly:** Rounded bubble-like structures, smooth, with a clear repeating pattern.
- **Crosshatched:** Sharp, interwoven lines resembling a network of thin threads or fibers.

- **Linear Method:**

- The images seem to progressively degrade from clear structures to more abstract or texture-less surfaces.
- Initial frames capture some bubbly texture, but towards the end, elements of either texture seem to lose clarity.

- **Spatial Method:**

- The textures seem more consistent, with a greater emphasis on blending the two textures.
- Clearer preservation of both bubbly and crosshatched features with novel patterns emerging, particularly with lines and intersecting patterns.

Second Pass - Metric Assessment

Now, I will assess the three metrics: Feature Preservation, Perceptual Blending, and Visual Novelty, for each method.

Method	Feature Preservation	Perceptual Blending	Visual Novelty
Linear	5/10	4/10	3/10
Spatial	8/10	7/10	9/10

Detailed Analysis

1. Feature Preservation:

- **Linear:** Initially, the bubbly texture holds fairly well, but by the third and fourth frames, the key elements from both source textures degrade significantly. The last frame shows minimal resemblance to either starting point.
- **Spatial:** The spatial method demonstrates superior preservation of features. Both the crosshatched lines and the bubbly shapes remain visible throughout the frames, though transformed in various ways, they stay recognizable.

2. Perceptual Blending:

- **Linear:** The blending seems more abrupt, with one texture dominating and then fading, rather than a smooth transition between the two. The intermediate images feel more like disjointed stages rather than blended transitions.
- **Spatial:** There is a much smoother and coherent blend in the spatial method. The elements from both textures intermingle gradually, creating a harmonious combination of crosshatched lines and bubbly patterns in the middle frames.

3. Visual Novelty:

- **Linear:** The interpolation does not appear to generate any particularly novel or creative new textures. Most frames tend to lose distinctiveness, and the results seem somewhat predictable.
- **Spatial:** The spatial method introduces interesting novel patterns, particularly in the middle frames, where both source textures interact to form new, intricate line and bubble patterns that neither source texture has individually.

End-Point Pair 2: cracked__polka-dotted–0059-0203

I see the second pair of images with the source textures "cracked" and "polka-dotted" followed by the Linear and Spatial interpolation results. I will follow the same process as before, starting with a quick pass through the images, followed by a detailed assessment across the three metrics.

First Pass - General Observations

- **Source Images:**

- **Cracked:** A surface with deep cracks resembling dried mud or wood bark.
- **Polka-dotted:** Bright background with pink polka dots against a blue-green field.

- **Linear Method:**

- The intermediate images transition to more muted, textured surfaces that do not clearly carry forward elements of the polka dots or cracks. The progression is somewhat uniform and lacks distinct features from either endpoint.

- **Spatial Method:**

- The intermediate images contain stone-like patterns, with a mix of shapes and sizes reminiscent of pebbles or tiles. There is a clearer combination of both concepts—cracks evolving into stone-like structures, while polka-dots transform into the pattern of coloured stones.

Second Pass - Metric Assessment

Method	Feature Preservation	Perceptual Blending	Visual Novelty
Linear	4/10	3/10	2/10
Spatial	7/10	8/10	9/10

Detailed Analysis

1. Feature Preservation:

- **Linear:** Both the cracked texture and polka-dots lose their defining features early in the interpolation. The intermediate images resemble surfaces with texture, but they don't maintain the distinct crack or dot patterns. By the final frame, neither source texture is very apparent.
- **Spatial:** The cracks from the "cracked" texture evolve into stone-like structures, and the polka dots appear to transform into coloured spots among the stones. Though modified, the defining features are more recognizable than in the Linear method.

2. Perceptual Blending:

- **Linear:** The transitions between the textures are not particularly smooth, with the results feeling like unrelated, independent surfaces rather than a blend. There is little sense of both textures contributing to the intermediate images.
- **Spatial:** The blending is more successful here, with both concepts merging into a stone pattern that contains elements of cracks and the polka dots. The evolution from cracked surfaces to the stone-like structures feels more natural and consistent.

3. Visual Novelty:

- **Linear:** The images do not introduce novel elements, as the textures transition into generic, weathered surfaces with limited visual interest.
- **Spatial:** The Spatial method creates a new texture pattern that resembles an intricate pebble mosaic, where both source textures contribute to the result. The introduction of unique stone shapes and coloured patterns makes these images more visually novel.

End-Point Pair 3: crystalline__pleated-0057-0053

In the third pair of textures, “crystalline” and “pleated,” followed by the Linear and Spatial interpolation results.

First Pass - General Observations

- **Source Images:**

- **Crystalline:** An image of sharp-edged, angular metallic shards, suggestive of crystallized forms.
- **Pleated:** A smooth, fabric-like material with vertical pleats, soft and repetitive in structure.

- **Linear Method:**

- The images appear to focus on circular, repetitive textures that don’t immediately reflect elements of either crystalline or pleated patterns. The frames seem uniform and abstract.

- **Spatial Method:**

- The Spatial method offers more dynamic and varied results, with a progression from organic shapes to wooden textures, and even some colourful abstract patterns at the end. The textures from both the crystalline and pleated source images are more clearly visible.

Second Pass - Metric Assessment

Method	Feature Preservation	Perceptual Blending	Visual Novelty
Linear	3/10	2/10	3/10
Spatial	8/10	7/10	8/10

Detailed Analysis

1. Feature Preservation:

- **Linear:** The distinctive sharpness of the crystalline shards and the pleated fabric pattern are not well-preserved. The images focus on circular abstract patterns that lack a clear connection to either source texture. The pleated texture is minimally reflected by the last image, but it's too subtle to score highly.
- **Spatial:** The Spatial method does a much better job of preserving the crystalline and pleated elements. The earlier frames exhibit organic, crystalline-like formations, while the later frames show patterns resembling pleats. Both features are maintained across the progression of images.

2. **Perceptual Blending:**

- **Linear:** There is little evidence of blending in the Linear method. The transitions seem uniform and static, and the two source textures don't interact or blend in a meaningful way.
- **Spatial:** In the Spatial method, the textures blend more smoothly, with organic transitions between the crystalline and pleated elements. The combination feels more cohesive, especially in the middle frames, where both textures contribute to the results.

3. **Visual Novelty:**

- **Linear:** The images generated by the Linear method lack novelty, as they focus on a single abstract pattern throughout the interpolation. There is little creativity or surprise in the results.
- **Spatial:** The Spatial method introduces novel combinations, particularly in the later frames where the textures evolve into unexpected patterns. The final frame, in particular, stands out as a creative result, with an abstract, colourful design that combines both elements in a visually engaging way.

End-Point Pair 4: flecked__stained–0053-0061

I see the fourth pair of textures, "flecked" and "stained," followed by the Linear and Spatial interpolation results. I will now perform the analysis using the same process.

First Pass - General Observations

- **Source Images:**

- **Flecked:** A smooth surface with pebble-like formations scattered across it.
- **Stained:** A dark background with bright yellow and orange circular stains resembling spots or blotches.

- **Linear Method:**

- The Linear method produces images that are mostly uniform grey textures, lacking any visible transition between the flecked and stained patterns. The frames seem overly simplified and featureless.

- **Spatial Method:**

- The Spatial method generates highly detailed images that evolve from floral-like patterns to more abstract feather-like formations, with a significant preservation of colour and structure from the stained image. The results are more visually engaging.

Second Pass - Metric Assessment

Method	Feature Preservation	Perceptual Blending	Visual Novelty
Linear	2/10	2/10	2/10
Spatial	9/10	9/10	8/10

Detailed Analysis

1. Feature Preservation:

- **Linear:** The flecked texture is only barely visible in the first frame, and the stained features are almost completely absent. By the end of the interpolation, the images are nearly indistinguishable from one another, with little preservation of the original textures.

- **Spatial:** The Spatial method clearly preserves the features of both source textures, especially the bright yellow colours from the stained texture. The final frames show a strong presence of both floral and feather-like patterns that seem derived from both sources.

2. **Perceptual Blending:**

- **Linear:** There is little perceptual blending, as the images show no significant combination of the source textures. The transition is static and doesn't bring elements from the flecked or stained textures together.
- **Spatial:** The blending is very effective, with both textures being represented in a harmonious manner. The floral-like patterns from the stained texture are merged with elements from the flecked texture, resulting in a smooth and visually balanced transition between the two.

3. **Visual Novelty:**

- **Linear:** The Linear method does not introduce any novel elements. The results are visually plain and offer little creativity in the blending of the textures.
- **Spatial:** The Spatial method introduces interesting floral and abstract patterns that weren't present in either of the source textures. This novelty adds an element of creativity and uniqueness to the interpolation.

End-Point Pair 5: freckled__cracked-0159-0162

For this fifth pair of textures, "freckled" and "cracked," I will follow the same procedure to provide a detailed analysis of the Linear and Spatial interpolation methods.

First Pass - General Observations

- **Source Images:**
 - **Freckled:** A pattern of tightly packed, small, circular spots in a green and yellow colour scheme.

– **Cracked:** A dried, cracked surface with deep fissures in a dark reddish-brown background.

• **Linear Method:**

– The Linear method transitions from what appears to be a flat, soil-like texture to a cracked surface, with no evidence of the freckled pattern. The images are somewhat predictable and visually resemble everyday objects (e.g., coffee grounds or cracked soil).

• **Spatial Method:**

– The Spatial method produces more intricate patterns, with the cracked texture morphing into more complex, detailed surfaces. The freckled elements are largely absent, but the cracked pattern is maintained throughout with some interesting colour variations.

Second Pass - Metric Assessment

Method	Feature Preservation	Perceptual Blending	Visual Novelty
Linear	4/10	3/10	4/10
Spatial	8/10	6/10	7/10

Detailed Analysis

1. Feature Preservation:

- **Linear:** The Linear method largely preserves the cracked texture, especially toward the later frames. However, the freckled pattern is completely absent, and the final frames focus more on the cracked, dried-soil appearance.
- **Spatial:** The cracked texture is well-preserved and becomes more visually detailed as the interpolation progresses. Although the freckled texture is not fully retained, some of its finer elements may be integrated subtly in the earlier frames.

2. Perceptual Blending:

- **Linear:** The blending of the two textures is weak, as the images mostly transition from flat

to cracked textures without incorporating features from both patterns. The interpolation feels more like a shift from one texture to another, rather than a blend.

- **Spatial:** The Spatial method exhibits stronger blending, particularly in the middle frames where the cracked texture gains additional complexity. While not perfect, the transition from freckled to cracked is smoother and more integrated than in the Linear method.

3. **Visual Novelty:**

- **Linear:** The images in the Linear method are visually similar to standard textures, such as soil or cracked earth, and lack unique or unexpected combinations. There's minimal novelty in the transition.
- **Spatial:** The Spatial method introduces more intricate, detailed patterns that deviate from the basic cracked texture. The subtle combination of features, especially the variations in the cracked surface, adds some visual interest and novelty.

End-Point Pair 5: freckled__cracked-0159-0162

For this fifth pair of textures, "freckled" and "cracked," I will follow the same procedure to provide a detailed analysis of the Linear and Spatial interpolation methods.

First Pass - General Observations

- **Source Images:**
 - **Freckled:** A pattern of tightly packed, small, circular spots in a green and yellow colour scheme.
 - **Cracked:** A dried, cracked surface with deep fissures in a dark reddish-brown background.
- **Linear Method:**
 - The Linear method transitions from what appears to be a flat, soil-like texture to a cracked surface, with no evidence of the freckled pattern. The images are somewhat predictable and

visually resemble everyday objects (e.g., coffee grounds or cracked soil).

- **Spatial Method:**

- The Spatial method produces more intricate patterns, with the cracked texture morphing into more complex, detailed surfaces. The freckled elements are largely absent, but the cracked pattern is maintained throughout with some interesting colour variations.

Second Pass - Metric Assessment

Method	Feature Preservation	Perceptual Blending	Visual Novelty
Linear	4/10	3/10	4/10
Spatial	8/10	6/10	7/10

Detailed Analysis

1. Feature Preservation:

- **Linear:** The Linear method largely preserves the cracked texture, especially toward the later frames. However, the freckled pattern is completely absent, and the final frames focus more on the cracked, dried-soil appearance.
- **Spatial:** The cracked texture is well-preserved and becomes more visually detailed as the interpolation progresses. Although the freckled texture is not fully retained, some of its finer elements may be integrated subtly in the earlier frames.

2. Perceptual Blending:

- **Linear:** The blending of the two textures is weak, as the images mostly transition from flat to cracked textures without incorporating features from both patterns. The interpolation feels more like a shift from one texture to another, rather than a blend.
- **Spatial:** The Spatial method exhibits stronger blending, particularly in the middle frames where the cracked texture gains additional complexity. While not perfect, the transition from freckled to cracked is smoother and more integrated than in the Linear method.

3. **Visual Novelty:**

- **Linear:** The images in the Linear method are visually similar to standard textures, such as soil or cracked earth, and lack unique or unexpected combinations. There's minimal novelty in the transition.
- **Spatial:** The Spatial method introduces more intricate, detailed patterns that deviate from the basic cracked texture. The subtle combination of features, especially the variations in the cracked surface, adds some visual interest and novelty.

End-Point Pair 6: gauzy__dotted-0173-0136

Let's analyze the sixth pair of textures, "gauzy" and "dotted," and perform the assessment for the Linear and Spatial methods.

First Pass - General Observations

- **Source Images:**
 - **Gauzy:** A plaid fabric pattern with a loose weave, featuring yellow and black stripes.
 - **Dotted:** A simple, blue-green fabric with white polka dots arranged in a regular grid.
- **Linear Method:**
 - The Linear method transitions from a flat, grid-like texture into a fabric-like image and finally back into a more patterned appearance. The plaid or gauzy texture does not seem well-preserved, and the transition feels abstract.
- **Spatial Method:**
 - The Spatial method demonstrates a stronger combination of the polka dots and some underlying structure, though the plaid (gauzy) pattern is not immediately visible. The results are more coherent and textured than the Linear method.

Second Pass - Metric Assessment

Method	Feature Preservation	Perceptual Blending	Visual Novelty
Linear	3/10	4/10	3/10
Spatial	7/10	6/10	7/10

Detailed Analysis

1. Feature Preservation:

- **Linear:** The Linear method does not preserve the key features of either source texture very well. The gauzy plaid pattern is almost entirely absent, and the polka dots, though visible, are distorted into more abstract forms.
- **Spatial:** The dots are well-preserved in the Spatial method, and though the gauzy plaid texture is not immediately visible, there are hints of underlying structure, especially in the final frames where the dots and structure interact more clearly.

2. Perceptual Blending:

- **Linear:** The blending of the textures is weak, with the interpolation leaning more toward grid patterns and abstract representations that do not capture the essence of either source. There is little evidence of a smooth blend between gauzy and dotted.
- **Spatial:** The Spatial method achieves better blending, with a more coherent transition from polka dots to textured patterns. Although the gauzy plaid pattern is still subtle, the overall impression is of a smoother combination between the two textures.

3. Visual Novelty:

- **Linear:** The Linear method does not introduce much novelty, as the results focus on repetitive, abstract patterns that lack visual interest or creativity. The final frames do not offer any unexpected combinations or features.
- **Spatial:** The Spatial method introduces more novelty, particularly in the later frames where

the textures become more intricate, with the dots interacting with an underlying structure. This creates a visually interesting effect that deviates from the original textures.

End-Point Pair 7: lacelike__knitted-0118-0128

For the seventh pair, "lacelike" and "knitted," I will again follow the same process to assess the Linear and Spatial methods.

First Pass - General Observations

- **Source Images:**

- **Lacelike:** A delicate, openwork pattern with intricate designs resembling lace fabric.
- **Knitted:** A woven pattern with repeating stitches, typical of a knitted fabric.

- **Linear Method:**

- The Linear method produces images that seem to lean toward fashion photography, showing models in clothing, particularly dresses. There is little connection to the lacelike or knitted textures, making the transition somewhat disconnected from the source textures.

- **Spatial Method:**

- The Spatial method focuses more on the knitted patterns, preserving the stitch details and evolving from loose to tightly woven designs. The lacelike structure is less prominent, but the results are consistent in maintaining fabric-like textures.

Second Pass - Metric Assessment

Method	Feature Preservation	Perceptual Blending	Visual Novelty
Linear	3/10	3/10	2/10
Spatial	8/10	7/10	6/10

Detailed Analysis

1. Feature Preservation:

- **Linear:** The Linear method does not effectively preserve the lacelike or knitted features. The images transition to a series of fashion photos that don't reflect the original textures. There is minimal evidence of the source textures in the final results.
- **Spatial:** The Spatial method does a much better job at preserving the knitted texture, with clear, well-defined stitch patterns throughout the frames. Although the lacelike features are not as prominent, the overall structure remains consistent with a fabric-like appearance.

2. Perceptual Blending:

- **Linear:** There is little perceptual blending between the two textures in the Linear method, as the transition results in fashion-like images that don't carry forward the lacelike or knitted features. The blending feels disconnected.
- **Spatial:** The Spatial method achieves better blending, especially with the knitted texture evolving smoothly through the frames. While the lacelike features are somewhat lost, the overall blend of fabric-like patterns is cohesive.

3. Visual Novelty:

- **Linear:** The Linear method introduces images that feel out of place, as the fashion-based results do not offer much in terms of novelty related to the textures. The images are more predictable and less creative in how they interpret the source textures.
- **Spatial:** The Spatial method introduces more visually interesting knitted patterns, with intricate and varied stitch designs. While the novelty is not as high as in some previous pairs, the results are still more creative than the Linear method's fashion-oriented images.

End-Point Pair 8: lined__woven-0133-0061

For this eighth pair, "lined" and "woven," I will conduct the analysis using the same process as before.

First Pass - General Observations

- **Source Images:**

- **Lined:** A pattern with alternating vertical coloured stripes in a repeating sequence.
- **Woven:** A dense weave with a criss-crossing, textured structure, resembling a wicker basket.

- **Linear Method:**

- The Linear method again features fashion-oriented images, showing models in various outfits. There is little evidence of either the lined or woven patterns in the resulting images.

- **Spatial Method:**

- The Spatial method focuses more on the structure and appearance of the textures, maintaining woven and plaid-like patterns. The lined and woven elements are more clearly integrated, with variations in colour and structure.

Second Pass - Metric Assessment

Method	Feature Preservation	Perceptual Blending	Visual Novelty
Linear	3/10	3/10	3/10
Spatial	8/10	7/10	7/10

Detailed Analysis

1. Feature Preservation:

- **Linear:** The Linear method again focuses on fashion designs, with minimal preservation of the lined or woven textures. The final images do not strongly resemble the original patterns.
- **Spatial:** The Spatial method does a better job of preserving both the lined and woven textures, especially in the middle frames where a plaid-like pattern emerges. The criss-cross weave is also maintained and evolves through the frames.

2. **Perceptual Blending:**

- **Linear:** The blending between the two textures is weak, as the images shift focus to fashion and away from the structural elements of the original textures. There is little evidence of smooth transitions between lined and woven features.
- **Spatial:** The Spatial method offers smoother blending, with lined and woven elements coming together in a balanced way. The images are consistent with a mix of plaid and woven patterns, making the blending more harmonious.

3. **Visual Novelty:**

- **Linear:** The images produced by the Linear method offer minimal novelty, as the fashion-centric results do not introduce creative new elements based on the source textures.
- **Spatial:** The Spatial method introduces more novel patterns, particularly in the plaid and woven combinations. The textures become more intricate and visually interesting as the interpolation progresses.

End-Point Pair 9: marbled__zigzagged-0114-0084

For this ninth analysis with the pair "marbled" and "zigzagged," I will conduct the evaluation of both the Linear and Spatial methods.

First Pass - General Observations

- **Source Images:**

- **Marbled:** A soft pink and white stone pattern with natural veining.
- **Zigzagged:** A vibrant, multi-coloured zigzag pattern with sharp, repeating lines.

- **Linear Method:**

- The Linear method creates images with complex, colourful patterns that seem far removed from both the marbled and zigzagged textures. The results resemble abstract prints, but neither source texture is immediately identifiable.

- **Spatial Method:**

- The Spatial method maintains a closer relationship to the source textures. The marbled texture transitions into a more colourful, patterned surface, and the zigzagged elements are retained with clear visual transformations.

Second Pass - Metric Assessment

Method	Feature Preservation	Perceptual Blending	Visual Novelty
Linear	4/10	4/10	5/10
Spatial	8/10	7/10	8/10

Detailed Analysis

1. Feature Preservation:

- **Linear:** The Linear method produces abstract results that do not preserve the marbled or zigzagged features very well. While colourful, the results lose the defining characteristics of both source textures, leaning towards more generalized abstract patterns.
- **Spatial:** The Spatial method preserves both the marbled and zigzagged features more effectively. The zigzag patterns are retained in multiple frames, and the marbled texture transforms into something new while still reflecting its original structure.

2. Perceptual Blending:

- **Linear:** The blending between the marbled and zigzagged textures is minimal. The images seem more focused on creating abstract patterns rather than achieving a smooth blend between the two source textures.
- **Spatial:** The Spatial method achieves a more coherent blend, with zigzag and marbled elements being combined in interesting ways. The textures complement each other as the frames progress, creating a more balanced transition.

3. **Visual Novelty:**

- **Linear:** The Linear method introduces some visual novelty through the creation of abstract, colourful patterns. However, the departure from the original textures makes the novelty feel disconnected from the source material.
- **Spatial:** The Spatial method introduces more visually creative results, maintaining ties to the original textures while creating new patterns that feel organic and engaging. The final frames, in particular, show a creative blend of both marbled and zigzagged features.

End-Point Pair 10: matted__cracked-0154-0078

For this tenth pair, "matted" and "cracked," I will proceed with the analysis of both the Linear and Spatial methods.

First Pass - General Observations

- **Source Images:**
 - **Matted:** A soft, wool-like texture, where the fibres are clumped together, creating a padded appearance.
 - **Cracked:** A rough, cracked surface with sharp lines, resembling dried mud or a cracked wall.
- **Linear Method:**

- The Linear method produces images that resemble wood planks or siding, progressing towards abstract, building-like structures. The matted or cracked features are not strongly preserved, and the results feel more like architectural textures.

- **Spatial Method:**

- The Spatial method maintains stronger ties to both the matted and cracked textures, showing clear transitions between the two. The matted texture evolves into a more structured pattern, and the cracked lines are evident in multiple frames.

Second Pass - Metric Assessment

Method	Feature Preservation	Perceptual Blending	Visual Novelty
Linear	4/10	3/10	3/10
Spatial	8/10	7/10	7/10

Detailed Analysis

1. Feature Preservation:

- **Linear:** The Linear method does not preserve the matted or cracked textures very well. The images shift towards architectural surfaces, and the defining features of the original textures are lost in the process.
- **Spatial:** The Spatial method preserves the matted texture in the earlier frames, where the clumps of fibres remain visible. The cracked lines are also well-preserved and transformed as the interpolation progresses.

2. Perceptual Blending:

- **Linear:** The blending between the two textures is minimal. The images shift abruptly from one form to another without smoothly incorporating elements from both source textures.
- **Spatial:** The Spatial method achieves a much better blend, with a gradual transition between

the matted and cracked features. The textures interact and evolve in a more cohesive manner, resulting in a smoother perceptual blend.

3. **Visual Novelty:**

- **Linear:** The Linear method introduces minimal visual novelty, as the images focus more on wood-like structures and buildings rather than creative interpretations of the source textures.
- **Spatial:** The Spatial method introduces more novelty, particularly in the final frames where the matted texture transforms into a more complex and structured form, and the cracked lines create interesting patterns.

End-Point Pair 11: matted__honeycombed–0162-0142

For the eleventh pair, "matted" and "honeycombed," I will proceed with the analysis of the Linear and Spatial methods using the same approach as before.

First Pass - General Observations

- **Source Images:**
 - **Matted:** An image resembling dense grass or fur, with a soft and textured appearance.
 - **Honeycombed:** A pattern of small hexagonal tiles with red accents, creating a structured, repeating geometric grid.
- **Linear Method:**
 - The Linear method produces images that resemble tiled or patterned surfaces with furniture or fabric-related objects. While the hexagonal (honeycombed) structure is somewhat visible in the earlier frames, the matted texture is not clearly preserved, and the final images lean towards manufactured objects like couches or beds.
- **Spatial Method:**

- The Spatial method provides more variety, preserving the matted texture in the first frame and transitioning into detailed tile or patterned designs that maintain elements from the honeycombed source. The transition feels more natural, and the combination of the two textures is more evident.

Second Pass - Metric Assessment

Method	Feature Preservation	Perceptual Blending	Visual Novelty
Linear	4/10	4/10	3/10
Spatial	8/10	7/10	8/10

Detailed Analysis

1. Feature Preservation:

- **Linear:** The Linear method shows some preservation of the honeycombed texture, but the matted texture is largely lost in the final images. The focus shifts towards furniture and patterned fabric, which departs from the original textures.
- **Spatial:** The Spatial method maintains both the matted and honeycombed features more effectively. The matted texture is clearly present in the first frame, and the honeycombed grid evolves into a more complex pattern throughout the interpolation.

2. Perceptual Blending:

- **Linear:** The blending between the matted and honeycombed textures is weak. The images shift focus towards unrelated objects and lose the core characteristics of both textures.
- **Spatial:** The Spatial method achieves a smoother blend, with elements from both textures interacting more harmoniously. The honeycombed and matted textures combine to create interesting, tile-like patterns that evolve throughout the frames.

3. Visual Novelty:

- **Linear:** The Linear method introduces minimal visual novelty. The images become more focused on furniture and textiles, which detracts from the creativity of combining the two source textures.
- **Spatial:** The Spatial method introduces more novel combinations, particularly in the later frames where the tile patterns evolve into unique designs that retain elements of both the matted and honeycombed textures.

End-Point Pair 12: paisley__swirly-0005-0098

For the twelfth pair, "paisley" and "swirly," I will proceed with the analysis for both the Linear and Spatial methods using the same approach as before.

First Pass - General Observations

- **Source Images:**
 - **Paisley:** A vibrant pattern with ornamental motifs, often featuring tear-drop shapes and intricate designs.
 - **Swirly:** A smooth, flowing pattern with curves and whorls, giving a sense of fluid motion in a blue colour scheme.
- **Linear Method:**
 - The Linear method produces images that appear more like abstract glitch art or digital interfaces, and they do not reflect either of the original textures. The results look unrelated to the paisley or swirly patterns.
- **Spatial Method:**
 - The Spatial method preserves both the paisley and swirly features, producing vibrant, fluid patterns that merge both textures beautifully. The results are colourful and consistent, with swirling motion blended with ornamental elements.

Second Pass - Metric Assessment

Method	Feature Preservation	Perceptual Blending	Visual Novelty
Linear	2/10	2/10	3/10
Spatial	9/10	8/10	9/10

Detailed Analysis

1. Feature Preservation:

- **Linear:** The Linear method does not preserve the features of either the paisley or swirly patterns. The images focus on abstract digital scenes and interface-like visuals that deviate completely from the original textures.
- **Spatial:** The Spatial method does an excellent job at preserving the paisley and swirly patterns. The vibrant swirls and ornamental motifs are maintained throughout the frames, blending the two source textures seamlessly.

2. Perceptual Blending:

- **Linear:** The blending between the two textures is minimal. The images do not display any recognizable transitions between paisley and swirly elements, and the results feel disconnected from the source textures.
- **Spatial:** The Spatial method achieves a much smoother blend between the paisley and swirly patterns. The combination is visually striking, with swirling elements gradually incorporating the ornamental paisley designs in a harmonious way.

3. Visual Novelty:

- **Linear:** The Linear method introduces abstract, glitch-art-like patterns, which may be visually novel but are far removed from the original textures. The departure from the source materials lowers the relevance of this novelty.
- **Spatial:** The Spatial method produces highly novel, vibrant combinations of the paisley and

swirly textures. The visual creativity is high, with new swirling and ornamental patterns that evolve from the original textures in unexpected ways.

End-Point Pair 13: porous__spiralled-0174-0140

For the thirteenth pair, "porous" and "spiralled," I will now proceed with the analysis of the Linear and Spatial methods.

First Pass - General Observations

- **Source Images:**

- **Porous:** A honeycomb structure with repeating hexagonal cavities, giving a porous, geometric appearance.
- **Spiralled:** A close-up of a rose, with spiralling petals creating a soft and organic texture.

- **Linear Method:**

- The Linear method produces images that seem more abstract and textural, with a focus on swirling lines and repetitive patterns. The spiralled features of the rose are somewhat visible, but the porous structure is not clearly retained.

- **Spatial Method:**

- The Spatial method retains much more of the spiralled features, with roses and floral elements being very apparent throughout the frames. The porous texture is less prominent, but the final images maintain a strong connection to the organic, spiralled source.

Second Pass - Metric Assessment

Method	Feature Preservation	Perceptual Blending	Visual Novelty
Linear	4/10	5/10	4/10
Spatial	8/10	7/10	8/10

Detailed Analysis

1. Feature Preservation:

- **Linear:** The Linear method somewhat preserves the spiralled, rose-like features, but the images become more abstract and disconnected from the original textures. The porous honeycomb structure is barely visible, and the swirling patterns are more dominant.
- **Spatial:** The Spatial method preserves the spiralled rose texture very well. The floral patterns are clear and vibrant, and although the porous structure is less prominent, the overall connection to the original textures is stronger.

2. Perceptual Blending:

- **Linear:** The blending between the two textures is moderate. The images seem to merge swirling patterns with some hints of floral structures, but the porous texture is largely absent, making the blend incomplete.
- **Spatial:** The Spatial method achieves a better balance between the two textures, with roses and floral elements blending naturally across the frames. The transitions between the spiralled and porous features are smoother and more visually consistent.

3. Visual Novelty:

- **Linear:** The Linear method introduces some novelty through its abstract patterns, but the connection to the original textures weakens the overall novelty. The results feel more like random textures than creative blends of the source materials.
- **Spatial:** The Spatial method introduces much more novelty, particularly in the later frames where the roses and floral patterns evolve into intricate and vibrant designs. The creative use of the spiralled texture leads to unique and visually engaging results.

End-Point Pair 14: scaly__lacelike-0145-0100

For the fourteenth pair, "scaly" and "lacelike," I will proceed with the analysis of both the Linear and Spatial methods using the same criteria as before.

First Pass - General Observations

- **Source Images:**

- **Scaly:** A texture resembling fish scales or roof shingles, with overlapping, repeating elements.
- **Lacelike:** An intricate, decorative pattern resembling lacework, with openwork designs and ornamental shapes.

- **Linear Method:**

- The Linear method focuses on furniture-like designs, specifically chairs and fabric patterns. The scaly and lacelike textures are not strongly preserved, and the final images feel disconnected from the original textures, instead resembling upholstery fabrics.

- **Spatial Method:**

- The Spatial method retains more of the original textures, particularly the scaly features, which are clearly visible throughout the frames. The lacelike elements are subtly integrated, creating intricate patterns that resemble fabric or ornamental designs with a stronger link to the original textures.

Second Pass - Metric Assessment

Method	Feature Preservation	Perceptual Blending	Visual Novelty
Linear	3/10	3/10	4/10
Spatial	8/10	7/10	7/10

Detailed Analysis

1. Feature Preservation:

- **Linear:** The Linear method does not preserve the scaly or lacelike textures well. The focus shifts to furniture design, and the original textures are barely visible. The images mostly resemble fabric patterns with little connection to the original textures.
- **Spatial:** The Spatial method does a much better job preserving the scaly texture, which is evident in most of the frames. The lacelike elements are also present, particularly in the middle frames, where they blend with the scaly structure to create a more intricate design.

2. Perceptual Blending:

- **Linear:** The blending between the scaly and lacelike textures is weak. The focus on furniture and fabric makes the blend feel disjointed, with neither texture dominating or interacting in a meaningful way.
- **Spatial:** The Spatial method achieves a better blend, with the scaly texture being the dominant feature, and the lacelike patterns adding ornamental details. The textures blend more naturally, resulting in a cohesive design that integrates both elements.

3. Visual Novelty:

- **Linear:** The Linear method introduces some novelty in terms of furniture design and fabric patterns, but it feels disconnected from the original textures. The results are not very creative in terms of combining the scaly and lacelike elements.
- **Spatial:** The Spatial method introduces more visually novel results, with intricate designs that combine the scaly and lacelike textures. The final frames, in particular, show creative interpretations of the textures, producing visually interesting patterns that remain connected to the source materials.

End-Point Pair 15: smeared__knitted-0087-0119

For the fifteenth and final pair, "smeared" and "knitted," I will now analyze both the Linear and Spatial methods using the same process as before.

First Pass - General Observations

- **Source Images:**

- **Smeared:** A texture with blended, smeared colours, creating a soft, cloud-like effect with no sharp edges or lines.
- **Knitted:** A close-up of a knitted fabric with a fuzzy texture and well-defined stitches.

- **Linear Method:**

- The Linear method produces images that appear to focus more on scenes involving people or objects, particularly fabric-related scenes. The original textures are not strongly preserved, and the images shift towards a more photographic style with less connection to the smeared or knitted patterns.

- **Spatial Method:**

- The Spatial method retains more of the original texture elements, particularly the smeared colours and knitted texture. The images evolve in a way that preserves the soft, blended qualities of the smeared texture, while also maintaining some structural elements from the knitted source.

Second Pass - Metric Assessment

Method	Feature Preservation	Perceptual Blending	Visual Novelty
Linear	3/10	4/10	4/10
Spatial	8/10	7/10	7/10

Detailed Analysis

1. Feature Preservation:

- **Linear:** The Linear method does not preserve the smeared or knitted textures very well. The images become more photographic and focus on scenes involving objects or people, which detracts from the original textures. There is little evidence of either texture being maintained.
- **Spatial:** The Spatial method does a much better job at preserving the smeared texture, which remains visible throughout the frames. The knitted texture is also present, though more subtly, and the overall preservation of features is stronger compared to the Linear method.

2. Perceptual Blending:

- **Linear:** The blending between the smeared and knitted textures is weak. The images transition away from the original textures and focus more on scenes that are unrelated to the source materials. The blend feels disconnected and incomplete.
- **Spatial:** The Spatial method achieves a much better blend, with the soft, smeared colours and the knitted texture interacting more naturally. The images retain a sense of both textures, and the transitions between them are smoother and more cohesive.

3. Visual Novelty:

- **Linear:** The Linear method introduces some novelty through the photographic scenes, but this novelty feels disconnected from the source textures. The results do not creatively explore the combination of smeared and knitted patterns.
- **Spatial:** The Spatial method introduces more visual novelty, especially in the later frames where the smeared colours and knitted elements create new, textured patterns. The results feel more creative and connected to the original textures, leading to more interesting outcomes.

End-Point Pair 16: sprinkled__waffled-0005-0137

For the sixteenth and final pair, "sprinkled" and "waffled," I will now analyze both the Linear and Spatial methods following the same process.

First Pass - General Observations

- **Source Images:**

- **Sprinkled:** A colourful assortment of sugar sprinkles in various shapes and sizes.
- **Waffled:** A structured waffle pattern with criss-crossing ridges and an overall grid-like texture.

- **Linear Method:**

- The Linear method produces images that seem to focus on landscapes and abstract textures, moving away from the original sprinkled and waffled textures. The results resemble fields or land surfaces, with little connection to the playful, colourful nature of the source textures.

- **Spatial Method:**

- The Spatial method retains the structured waffle texture and colourful sprinkled elements, creating vibrant and intricate patterns that combine both textures effectively. The final images are playful and visually engaging, with a clear connection to the source textures.

Second Pass - Metric Assessment

Method	Feature Preservation	Perceptual Blending	Visual Novelty
Linear	4/10	3/10	3/10
Spatial	9/10	8/10	9/10

Detailed Analysis

1. Feature Preservation:

- **Linear:** The Linear method does not preserve the sprinkled or waffled textures well. The images focus more on abstract, landscape-like textures that diverge from the original patterns. The sprinkled and waffled elements are barely recognizable.
- **Spatial:** The Spatial method does an excellent job preserving both the waffle grid structure and the colourful sprinkles. The textures are clearly visible throughout the frames, with the elements from both sources remaining prominent.

2. Perceptual Blending:

- **Linear:** The blending between the sprinkled and waffled textures is weak. The images do not show a clear transition between the two, and the focus on abstract patterns creates a disjointed blend.
- **Spatial:** The Spatial method achieves a much better blend between the two textures, with the sprinkled and waffled elements interacting harmoniously. The transitions between the patterns feel smoother and more cohesive, making the blending more successful.

3. Visual Novelty:

- **Linear:** The Linear method introduces some novelty through the abstract, landscape-like textures, but the departure from the original source materials weakens the connection to the sprinkled and waffled elements.
- **Spatial:** The Spatial method introduces significant visual novelty, with colourful and playful combinations of the sprinkled and waffled textures. The final frames are particularly creative, blending the textures in unexpected ways while maintaining the original features.

End-Point Pair 17: striped__lined–0107-0134

For the seventeenth pair, "striped" and "lined," I will now proceed with the analysis for both the Linear and Spatial methods using the same criteria as before.

First Pass - General Observations

- **Source Images:**

- **Striped:** A pattern of vertical black and white stripes with some variation in thickness and direction.
- **Lined:** A vibrant, multi-coloured pattern with horizontal lines in various colours, including purple, pink, blue, and green.

- **Linear Method:**

- The Linear method produces images that include a mixture of abstract patterns and scenes with people. The striped and lined textures are visible in some frames but not strongly maintained, and the images appear to be more focused on fashion or photography rather than the textures.

- **Spatial Method:**

- The Spatial method retains more of the striped and lined features, with abstract patterns that maintain the linear qualities of both source textures. The images feel more connected to the original striped and lined patterns, blending them in visually interesting ways.

Second Pass - Metric Assessment

Method	Feature Preservation	Perceptual Blending	Visual Novelty
Linear	4/10	3/10	4/10
Spatial	8/10	7/10	7/10

Detailed Analysis

1. Feature Preservation:

- **Linear:** The Linear method partially preserves the striped and lined patterns, but the focus shifts towards fashion or photographic elements, which detracts from the original textures. The striped and lined patterns are visible but not well-preserved.
- **Spatial:** The Spatial method does a better job at preserving both the striped and lined textures. The abstract patterns maintain the linear qualities of the original textures, with the stripes and lines clearly visible throughout the frames.

2. Perceptual Blending:

- **Linear:** The blending between the striped and lined textures is weak. The images focus more on scenes with people and abstract designs, which creates a disconnect from the source textures. The blend between the two textures feels incomplete.
- **Spatial:** The Spatial method achieves a smoother blend, with the striped and lined textures interacting more cohesively. The abstract patterns retain the essence of both source textures, making the blending feel more natural and integrated.

3. Visual Novelty:

- **Linear:** The Linear method introduces some visual novelty through its use of abstract patterns and photographic elements, but the departure from the original textures weakens the connection to the striped and lined elements.
- **Spatial:** The Spatial method introduces more novelty, with creative abstract designs that combine the striped and lined textures in new ways. The patterns feel more cohesive and visually engaging, with a strong connection to the original textures.

End-Point Pair 18: veined__blotchy-0075-0123

For the eighteenth pair, "veined" and "blotchy," I will now analyze both the Linear and Spatial methods using the same process.

First Pass - General Observations

- **Source Images:**

- **Veined:** A natural pattern resembling tree bark with distinct lines and textures, creating a rough, vertical appearance.
- **Blotchy:** A colourful pattern resembling leaves with random patches of colour, typically found in nature.

- **Linear Method:**

- The Linear method produces images of lamps and interior decor, completely departing from the veined and blotchy textures. The focus shifts toward architectural or decorative elements, with minimal connection to the original textures.

- **Spatial Method:**

- The Spatial method retains more of the natural qualities of both the veined and blotchy textures. The final images feature nature-inspired patterns that combine the linear veining of trees and the blotchy leaf-like elements, maintaining a strong connection to the source textures.

Second Pass - Metric Assessment

Method	Feature Preservation	Perceptual Blending	Visual Novelty
Linear	3/10	3/10	3/10
Spatial	8/10	7/10	8/10

Detailed Analysis

1. Feature Preservation:

- **Linear:** The Linear method does not preserve the veined or blotchy textures well. The images focus on lamps and decor, and the original patterns are lost in the process. There is little evidence of either texture in the final results.
- **Spatial:** The Spatial method does a much better job preserving both the veined and blotchy textures. The final images maintain the natural, textured qualities of both source images, particularly in the middle frames, where the patterns of leaves and bark blend well.

2. Perceptual Blending:

- **Linear:** The blending between the veined and blotchy textures is weak. The images transition into unrelated objects (lamps and decor), making the blend between the two textures feel incomplete.
- **Spatial:** The Spatial method achieves a much better blend, with elements from both textures interacting naturally. The veined and blotchy patterns complement each other, creating a more cohesive blend that feels connected to nature.

3. Visual Novelty:

- **Linear:** The Linear method introduces some novelty in terms of interior decor and lighting design, but this feels disconnected from the original textures. The departure from the source materials weakens the overall novelty.
- **Spatial:** The Spatial method introduces more visual novelty by combining the veined and blotchy textures in new and interesting ways. The final images are creative and visually engaging, with organic patterns that feel fresh and unique.

End-Point Pair 19: woven__chequered-0033-0066

For the nineteenth pair, "woven" and "chequered," I will now proceed with the analysis of both the Linear and Spatial methods, as before.

First Pass - General Observations

- **Source Images:**

- **Woven:** A complex pattern with vertical and horizontal lines that interlace, resembling a woven fabric.
- **Chequered:** A bold, geometric pattern with alternating squares in yellow, red, and black, creating a strong grid-based design.

- **Linear Method:**

- The Linear method produces images with smooth, abstract textures that do not strongly reflect the woven or chequered patterns. The images appear as muted, soft surfaces with no distinct structural features from the original textures.

- **Spatial Method:**

- The Spatial method retains much more of the original woven and chequered features. The images display intricate, grid-like patterns that merge the woven and chequered elements, creating dynamic designs with both geometric structure and interwoven qualities.

Second Pass - Metric Assessment

Method	Feature Preservation	Perceptual Blending	Visual Novelty
Linear	4/10	3/10	4/10
Spatial	8/10	7/10	8/10

Detailed Analysis

1. Feature Preservation:

- **Linear:** The Linear method does not preserve the woven or chequered textures very well. The images are more abstract and lack the grid or woven structures that define the source textures, making them difficult to identify in the final results.
- **Spatial:** The Spatial method does a much better job at preserving both the woven and chequered textures. The grid-based patterns from the chequered source are clearly visible, and the woven interlacing adds complexity to the final designs.

2. Perceptual Blending:

- **Linear:** The blending between the woven and chequered textures is weak. The images are soft, smooth surfaces that do not reflect the structural qualities of either source texture. The blend feels disconnected from the original materials.
- **Spatial:** The Spatial method achieves a much more cohesive blend, with both the woven and chequered textures interacting to create intricate, structured designs. The blend feels organic and well-integrated, maintaining the essence of both patterns.

3. Visual Novelty:

- **Linear:** The Linear method introduces some novelty through the abstract, soft-textured surfaces, but these results feel disconnected from the original textures and do not offer much creative exploration of the woven and chequered elements.
- **Spatial:** The Spatial method introduces significant visual novelty, with creative combinations of the woven and chequered patterns. The designs are complex and visually engaging, showing new interpretations of the source textures in dynamic ways.

End-Point Pair 20: wrinkled__cobwebbed-0052-0080

For the twentieth and final pair, "wrinkled" and "cobwebbed," I will now analyze both the Linear and Spatial methods using the same process.

First Pass - General Observations

- **Source Images:**

- **Wrinkled:** A rich, silky fabric with a wrinkled texture, creating soft folds and a glossy surface.
- **Cobwebbed:** A detailed spiderweb, with thin, intricate strands forming a circular web structure, set against a natural background.

- **Linear Method:**

- The Linear method produces images that focus on orange slices and objects, with no clear connection to the wrinkled or cobwebbed textures. The focus shifts to still life photography, with minimal preservation of the original patterns.

- **Spatial Method:**

- The Spatial method retains more of the original textures, particularly the cobweb structure, which is visible throughout the frames. The wrinkled texture is also subtly integrated, creating a blend between the smooth, fabric-like qualities and the intricate web structure.

Second Pass - Metric Assessment

Method	Feature Preservation	Perceptual Blending	Visual Novelty
Linear	3/10	2/10	3/10
Spatial	9/10	8/10	8/10

Detailed Analysis

1. Feature Preservation:

- **Linear:** The Linear method does not preserve the wrinkled or cobwebbed textures well. The focus is entirely on still life photography with oranges, and the original patterns are not visible. The images feel disconnected from the source materials.
- **Spatial:** The Spatial method does an excellent job preserving the cobweb structure, which is clearly visible in multiple frames. The wrinkled texture is also present, though more subtle, and the overall preservation of both features is much stronger than in the Linear method.

2. Perceptual Blending:

- **Linear:** The blending between the wrinkled and cobwebbed textures is minimal, as the images transition to still life photography with no connection to the original textures. The blend feels incomplete and disjointed.
- **Spatial:** The Spatial method achieves a much better blend, with the cobweb and wrinkled textures interacting smoothly. The transitions between the two textures are cohesive, and the final images feel balanced, integrating both elements in a visually engaging way.

3. Visual Novelty:

- **Linear:** The Linear method introduces some novelty through the still life scenes, but these results are disconnected from the original textures and do not explore the combination of wrinkled and cobwebbed patterns in a creative way.
- **Spatial:** The Spatial method introduces significant visual novelty, with intricate combinations of the cobwebbed and wrinkled textures. The final images are creative, and the interplay between the smooth fabric-like textures and the web structures creates a visually intriguing result.

IEEE Copyright Notice

In reference to IEEE copyrighted material which is used with permission in this thesis, the IEEE does not endorse any of University of Ottawa's products or services. Internal or personal use of this material is permitted. If interested in reprinting/republishing IEEE copyrighted material for advertising or promotional purposes or for creating new collective works for resale or redistribution, please go to:

http://www.ieee.org/publications_standards/publications/rights/rights_link.html

to learn how to obtain a License from RightsLink. If applicable, University Microfilms and/or ProQuest Library, or the Archives of Canada may supply single copies of the dissertation.