Sketch-conditioned Image Generation centered on Existing Artist Workflows

by

Dagmar Lukka Loftsdóttir

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

Master of Science

Department of Computing Science

University of Alberta

Abstract

Many works of art are created through the process of an artist sketching and then incrementally increasing the fidelity of the artwork. This requires significant amounts of work and effort throughout, but not all steps require the same amount of artistic input. Certain parts only require following the logical consequences of an earlier decision. Other parts require multiple iterations to visualize the effect of a decision before it is executed. This thesis provides two contributions, each of which integrates with existing artist workflows.

Firstly, we propose SketchBetween, a VQ-VAE-based system which automates a part of the animation workflow which leaves full creative control in the hands of the artist. This system takes as input fully rendered keyframes and sketched in-between frames and returns a fully rendered animation. We evaluate the system by comparing it to a baseline and performing an ablation study. We also analyze a case study of its performance on human-generated input. The results of all evaluations demonstrate the systems effectiveness at automating the rendering of in-between frames based on sketches.

Secondly, we propose the task of generating images conditioned on sketches and palettes, and propose a preliminary system to perform this task using a VQ-VAE. We performed an error analysis in order to identify current limitations, to which we propose mitigations and future work.

Preface

The work in Chapter 3 of this thesis has been published at the 17th International Conference on the Foundations of Digital Games (FDG) 2022, and further presented at the Creative AI Across Modalities workshop at the 37th AAAI conference on artificial intelligence (AAAI) 2023. Other work in this thesis may be expanded upon and submitted for publication at relevant research avenues.

Til Felix

Takk fyrir sönginn og samveruna öll þessi ár.

Acknowledgements

I am immensely grateful for the support of my supervisor, Dr. Matthew Guzdial, whose guidance has been an invaluable resource to me in the work leading up to this thesis. I would like to thank my supervisory committee for their expertise and careful consideration of this thesis. I have the immense honor of having been awarded a DeepMind scholarship which has given me access to both a source of funding as well as the DeepMind scholar community and my DeepMind mentor, both of which have helped me to grow as a researcher. I would like to acknowledge the support of the Natural Sciences and Engineering Research Council of Canada (NSERC), Alberta Machine Intelligence Institute (Amii) and the Digital Research Alliance of Canada.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation

When creating a frame-by-frame 2D animation, an artist must make many artistic decisions that shape the final artifact. However, there is more to animation than making decisions, one must also execute on the logical consequences of making these decisions. For example, certain animation workflows involve a large amount of time spent following through on the logical consequences of the choices that have already been made. Frame-by-frame 2D animation is a medium of art where this is especially true. In large-scale productions, this manifests in certain parts of the process being exported to studios of workers who do not make as many decisions. For smaller scale productions, this is not feasible, and animators must themselves take on this labor.

With the spread and advancement of automation of labor-intensive processes via machine learning, the prospect of automating parts of animation has a certain appeal. However, many existing approaches are formulated without considering the existing workflows and processes developed by artists, and are therefore not easily adopted. To many artists, the author included, creative control plays a key role in art creation, which detracts from the appeal of existing systems. It is paramount to consider the current practices in a domain when envisioning ways to improve it. Therefore, we suggest a system which closely adheres to a common workflow of frame-by-frame 2D animation and retains creative control to a great extent by automating a part of the workflow that is known by practitioners to be largely a matter of execution.

-

1.2 Animation

Animation is an artform in which frames of art are shown to the viewer at a rate which gives the artwork the illusion of motion. There are many different kinds of animation, but the focus of this thesis is so-called frame-by-frame animation, wherein each frame is a two-dimensional illustration. Certain media necessitate the creation of 2D animations for cutscenes, backgrounds, effects, objects or characters. The creation of these types of art assets takes both a high level of skill and a great deal of time [46]. The process of 2D animation typically breaks down into first creating keyframes that outline some action, and then filling in the frames between these keyframes to give the appearance of continuous motion. The frames between keyframes are called in-between frames. This is a resource intensive process as it requires someone skilled in visual arts to render many frames for every second of animation. However, this task is not the most demanding of an animator's particular skillsket, as rendering these in-between frames involves rote, repetitive work.

We propose a system which takes as input fully rendered keyframes that define the design and art style of the subject, and sketches for in-between frames. The system then produces a fully rendered sprite animation. Existing approaches to video generation and synthesis thus far have not adhered to this standard workflow for animation. Many of the existing methods use previous frames as a prior to generate future frames [32], [35], [48], which requires the full rendering of several adjacent frames and surrenders creative control of the overall motion. Others rely on transferring a complete motion onto a source image [43]. Both of these methods cannot handle certain common motions, such as turning around. This is because they offer only one perspective of the subject at hand to the model at inference time. Therefore, the model must itself fill in details of things that are occluded early in the animation and revealed later. This may leave a character off-model and decreases the consistency of generated animation snippets over a larger context. We propose

a task for the automated rendering of in-between frames given sketches to define the motion, and keyframes to define style. Our input includes both a keyframe at the start and at the end to capture information about parts of the subject that may be obstructed in either single keyframe. We additionally propose SketchBetween, a system to solve this task. Our system consists of a VQ-VAE [32] that encodes rich information in its latent space about both the desired style and motion and uses this to generate a fully rendered animation. As seen in Figure 3.1, SketchBetween can capture stylized motion with accurate character designs.

There is no direct comparison we can make to other work, as our proposed task is not one that has been studied before. However, it builds upon work in adjacent fields and similar tasks. We evaluate SketchBetween against a strong baseline method adapted to our task using the structural similarity index metric (SSIM) [52] and the peak signal-to-noise ratio (PSNR) measured between the original animation and the model's recreation. Compared to the baseline, we see a clear indication of its appropriateness for the target task.

1.3 Thumbnail Generation

Many artistic processes involve making decisions about the placements of colour. There are good practices and guidelines that can help an artist achieve an aesthetically pleasing composition, but the artist must choose between infinitely many different ways to lay out colours. Therefore, artists will often make quick drafts of different variations of a piece, or make different versions of a character to visually explore the effects of different decisions. The process of making previews of different ideas for a piece or designing different versions of a character is often referred to as **thumbnailing**. Different compositions of colour can change the mood of a piece, and different distributions of colours can communicate different things about a character.

This process can be time consuming as an artist may need to produce many variations of the same idea before committing to a single version. We propose a method that automates colouring in an image given a sketch and a

palette. This method allows an artist to generate multiple different outputs for the same palette, or generate outputs for multiple different palettes for the same sketch. This can help facilitate the ideation process when designing a character or creating a composition. This could help reduce the amount of labour while allowing artists to consider more variations.

Automated colouration is a task which has existing approaches, however these approaches have issues with controllability, which is important to preserve an artist's creative control. Existing work proposes changing the colour scheme of a given image, but in order to use these methods the image must already have been fully rendered, and not just a sketch [1]. Other methods allow for the use of un-coloured lineart as input but do not produce outputs that are faithful to the colours chosen by the artist [6].

1.4 Research Questions

The research questions of this thesis are as follows:

- Can a VQ-VAE be used to automate the rendering process of in-between frames given rendered keyframes and sketches of the in-between frames?
- Can a VQ-VAE be used to generate thumbnails given a sketch and a palette as input?

1.5 Contributions

Our contributions are summarized as follows:

- We introduce a problem formulation for the conditional generation of in-between frames which adheres to existing animation workflows.
- We propose a system using a VQ-VAE as a solution to this problem formulation, which we call SketchBetween.
- We construct a baseline for the evaluation of SketchBetween based on a system developed for an adjacent problem. We then evaluate SketchBetween against this baseline.

- We evaluate the soundness of the problem formulation through an ablation study where we evaluate models trained on variations of the problem.
- We introduce a problem formulation for conditional image generation using sketches and palettes.
- We explore a system constructed for conditional generation via sketches and palettes which is based on a VQ-VAE.
- We evaluate this system and examine its outputs to identify future work.

1.6 Thesis Outline

This thesis consists of seven chapters. This first chapter is an introduction to the thesis, and the rest of the thesis is structured as follows: Chapter 2 provides relevant background information to understand the context of the thesis as well as its technical aspects. Chapter 3 introduces SketchBetween, a problem formulation and a proposed system to automate part of a common 2D animation workflow. Chapter 4 introduces a problem formulation and proposed system to automate sketch colourization to allow artists to iterate on ideas faster. In the final chapter, Chapter 5 we review the conclusions of this thesis, propose future work and speculate on potential impact.

Chapter 2

Background & Related Work

In this chapter we cover the background required to engage with this thesis. We explain the area of visual arts the thesis touches on along with concepts and prior work in computing science. In section 2.1 we delve into visual art, specifically digital frame-by-frame animation. In section 2.2 we discuss the prior technical works on which we base our research. In the following section, section 2.3 we perform a literature review for our work on sprite animation synthesis.

2.1 Art

In order to contextualize this thesis, we must introduce the reader to relevant areas within visual art. Firstly, we broadly discuss the medium of digital art in subsection 2.1.1. Following this, we discuss frame-by-frame 2D animation in section 2.1.3.

2.1.1 Digital art

Humans have a long history of creating art. The mediums we use to make art have evolved as humans invent new technologies, allowing for new forms of expression. A recent example of this was the development of digital mediums, colloquially referred to as digital art. These mediums are a product of software that allows a human to produce a fully digital artifact. Algorithmic tools have been created which allow artists to both work faster and create in new ways.This development introduced new art movements and styles [33].

2.1.2 Colour representations

Colour is perceived by humans using cones and rods in the eyes. The defacto colour representation on computers uses the same three colours that humans have cones to perceive. This representation is called RGB, which stands for red, green and blue. However, there exist different ways to represent colour [20]. Relevant to this thesis is the HSV colour space, which is a transformation of the RGB colour space that encodes colours in properties that align better with human perception of colours. HSV stands for hue, saturation and value. Additionally, the YUV colour space is relevant to a metric used in our evaluations. The YUV colour space, similarly to HSV, isolates the value of a colour to one channel, Y. The U and V channels encode the remaining colour information, but are not analogous to any colour concepts on their own.

2.1.3 Frame-by-Frame Animation

Animation is a genre of visual art where multiple images give the illusion of motion. This is a popular artform for storytelling, as well as appearing frequently in video games. Animation is an expensive medium, requiring both expertise and significant manual labor [46].

In two-dimensional frame-by-frame animation the images are all produced by hand. This involves drawing multiple images called frames and showing them to the viewer in quick succession such that there are multiple frames per second. In frame-by-frame animation, many different methods and workflows exist. However, there is a common process broken down in steps, going from a rough to polished animation. Certain steps and general ideas are common enough for a shared vocabulary amongst practitioners.

Generally, animations are first sketched, then inked, then rendered. The sketching process can be further broken down into first creating keyframes and then in-betweens. Keyframes are the frames which define the action of the animation. After they are completed, in-betweens are added "between" the keyframes to give the illusion of motion. At this point the animation is then inked and rendered. The rendering can also be broken down further into

steps, but this process is not strictly relevant to the thesis, so it is omitted. This process is illustrated in Figure 2.1.

Technological advancements have led to the birth of new styles of animation. Notably, puppet-based animation and 3D animation. Both of these modes of animation involve the use of a model. The model is a collection of 2D images for puppet-based animation and a 3D model for 3D animation. The animation itself then consists of moving parts of the model. These styles of animation have their own workflows and processes, parts of which have been inherited from the more traditional frame-by-frame method. However, two-dimensional frame-by-frame animation is still frequently used as it inherently allows for a significant freedom of expression, which in some cases one can lose due to the constrictions imposed by puppet-based animation and 3D animation.

2.2 Machine Learning

Machine learning is an area of artificial intelligence which involves learning approximations of functions. There are three branches of machine learning, supervised learning, unsupervised learning and reinforcement learning [28]. The first two, supervised learning and unsupervised learning are relevant to this thesis.

Supervised learning is the branch of machine learning that describes methods which approximate a function $F(X) = Y$, where X is an input and Y is the desired output. For these methods, examples of X and Y are used to learn a function $f(X) = Y'$ such that $Y' = Y$ for all X in some distribution. This group of methods is useful for learning mappings between inputs and outputs, such as learning to predict the sentiment of a given piece of text or predicting what part of an image is the background.

Unsupervised learning is the branch of machine learning where the goal is to learn some underlying structure in data without having any known outputs or labels. This area of machine learning is useful for uncovering patterns in large amounts of data, and is used for tasks such as grouping similar things

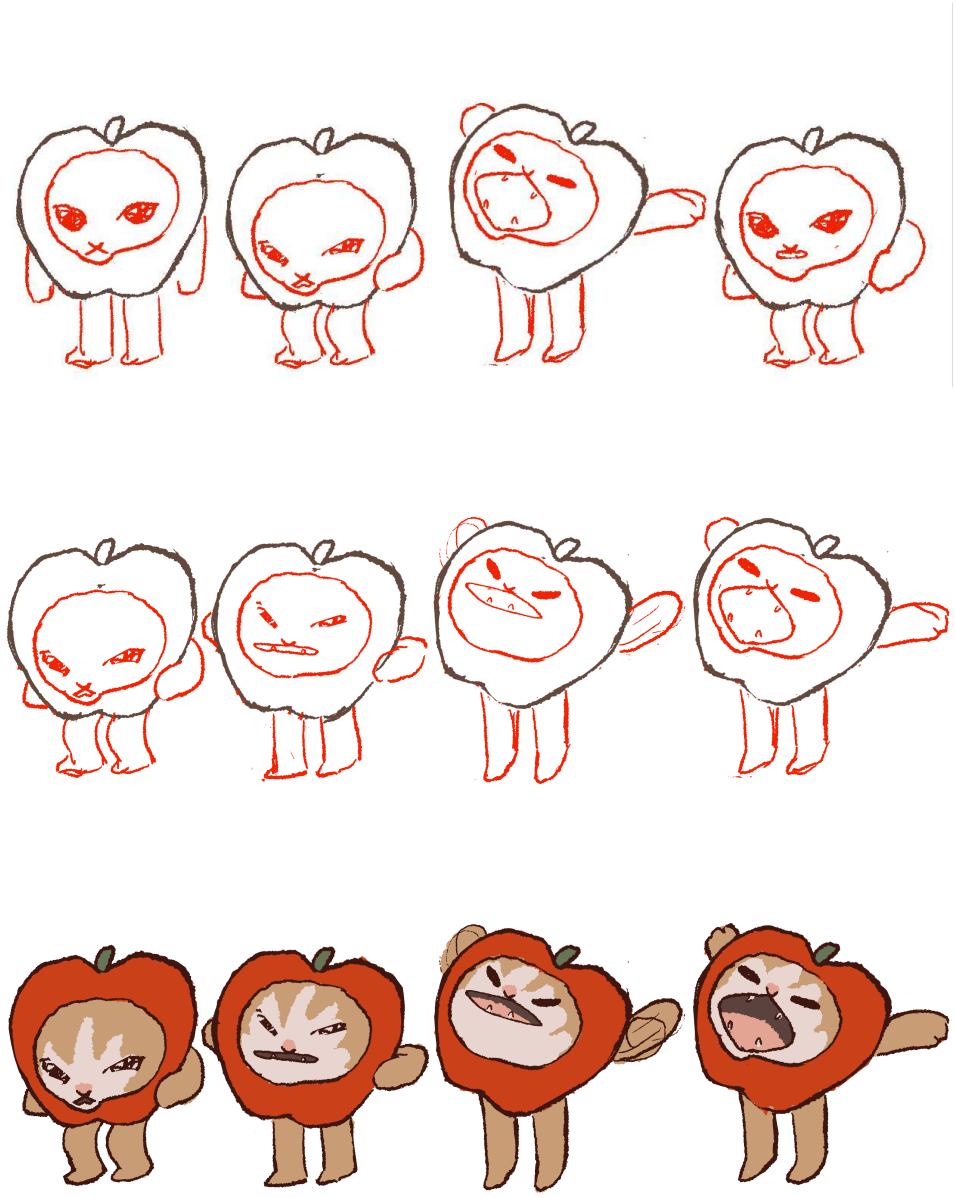

Figure 2.1: Three rows of images. The top row shows sketched keyframes which define the entire action of a cat-apple yawning. The second row shows the second and third keyframes with two in-between frames added. These two in-betweens help to add the illusion of motion. The third row is the finalized version of the four frames in the second row.

together and detecting outliers.

In this thesis, we make use of both supervised and unsupervised learning. We use supervised learning to go from an input of five frames, two of which are rendered, to an output of those same frames fully rendered. We also use supervised learning to take an input of a sketch and palette and output a rendered image based on the sketch with the given palette. We use unsupervised learning to extract palettes from training images.

2.2.1 Artificial Neural Networks

Artificial neural networks (ANNs) [16], [27] are approximation algorithms that operate on data. The data consists of input-output pairs for which the artificial neural network learns a mapping. It learns this mapping by processing a significant number of input-output pairs and optimizing the parameters belonging to the neurons in the network.

Neurons are the fundamental building block of an artificial neural network. They take one or more inputs and transform them according to learned parameters for each input, sum up all of the transformed inputs, then apply a non-linear function, producing a single output. The simplest version of an artificial neural network consists of multiple neurons, organized in layers such that the outputs of one layer become the inputs to each neuron in the next layer. There exist many variations of neural networks for different use-cases, but they largely make use of the same fundamental idea of a neuron.

The parameters of neurons in a neural network are optimized to minimize a loss function. Intuitively, this loss function describes the difference in the generated output and the desired output on a given batch of data. This optimization happens through a process called back-propagation. This process calculates each parameter's contribution to the loss in order to make a small nudge in whatever direction would decrease the loss. This nudging of parameters is called gradient descent, since the direction of the nudge is determined by calculating the gradient of the loss function with respect to the parameter. Calculating a gradient is required, therefore the network and loss function must be differentiable.

Artificial neural networks are powerful function approximators and we make use of them throughout this thesis.

Convolutional neural networks (CNNs)

Convolutional neural networks (CNNs) [31] are a type of ANN (See section 2.2.1) which are particularly useful in processing images. The typical image is a grid of pixels where each pixel stores the value of red, green and blue intensity to represent the pixel's colour. Therefore, an image is a three-dimensional tensor containing $H \times W \times 3$ numbers. If each of these numbers were used as input to a neural network it would quickly become intractable to process images of any reasonable size. To mitigate this problem, a CNN does not connect the output of every neuron as input to every neuron in the next layer. Instead, the neurons are arranged in three-dimensional tensors called kernels that are applied to the entire image through discrete convolution. Intuitively this means that the kernel slides across the image and calculates outputs at fixed intervals. This produces intermediate outputs at each layer that are also three-dimensional tensors. The benefits of this method are that the same parameters are shared across the entire image, meaning that it can recognize the same features located anywhere in the image. This sharing of parameters also significantly reduces the cost of processing images.

Given that the premise of this thesis has to do with visual art, we make use of convolutional neural networks because of their compatibility with image data.

2.2.2 Vector-Quantized Variational Autoencoders (VQ-VAEs)

Autoencoders are a type of artificial neural network[2] (See section 2.2.1). They embed their inputs into a latent space through an encoder, and then translate from the latent space to an output via a decoder. The networks are trained simultaneously, learning to both encode and decode the data. The purpose of embedding the data into a latent space is to condense information. The dimensionality of the embedding is smaller than that of the input, meaning it

contains less information. The encoder must then learn to encode only what is most valuable to the decoder in order to translate it back into a larger dimensionality.

This model benefits a variety of tasks. It has been used for error-detection and to map one image to another while retaining some semantic information. There are a number of extensions to autoencoders which have further use-cases.

One such model is the variational autoencoder [18]. One shortcoming of the standard autoencoder is that its latent space can be sparse, which makes sampling from it difficult. Sampling a point in the latent space of a typical autoencoder between two well-defined points is not guaranteed to be well defined.. In this case, there is limited value in the latent space since it cannot be easily used to generate new things. The variational autoencoder mitigates this problem by constraining the latent space to be biased towards a normal distribution. Another kind of autoencoder which has shown great performance in image generation is the vector-quantized variational autoencoder (VQ-VAE) [32]. It offers similar guarantees as a VAE through discretizing the latent space. It does this through constraining the latent space to a defined codebook of latent vectors. When encoding, a VQ-VAE encoder takes the input X and embeds it into a latent space as a set of vectors, where each vector describes a part of the original image. Each of these vectors is swapped out for its closest member in a learned codebook. This discretizes the latent space. We include the structure of this model in Figure 2.2.

VQ-VAEs have proven useful, particularly in image-related tasks. The quality of VQ-VAEs to represent parts of the image as vectors in its latent space helps VQ-VAEs represent pixel art [40]. Certain art styles, such as pixel art, have a more discrete quality than photorealistic images. Throughout this thesis we seek to learn a mapping from sketches and some additional information to rendered artistic images. VQ-VAEs were therefore chosen as a candidate model. VQ-VAEs and other discrete autoencoders have additionally shown success in a variety of image generation tasks [30], [32].

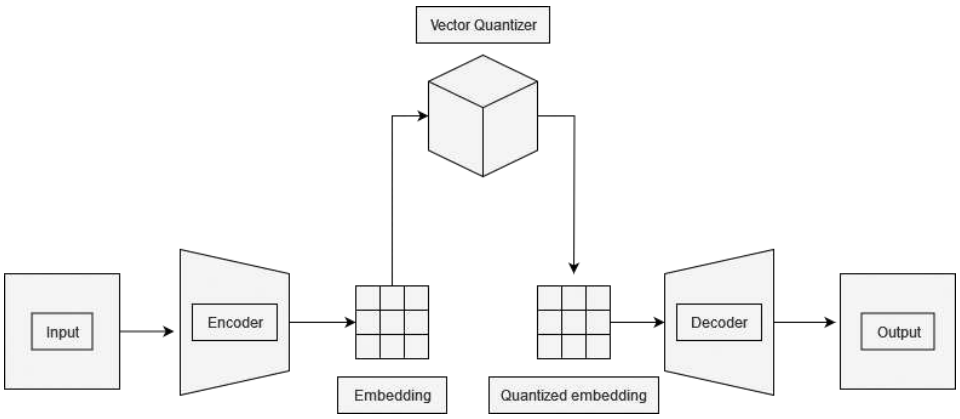

Figure 2.2: The model structure of a VQ-VAE. The arrows in the image show data flowing through the model in a forward pass. Data flows from left to right, through the encoder, then the vector quantizer, then the decoder. This diagram is based on the diagram in the original VQ-VAE paper [32].

2.2.3 Clustering

Clustering is a kind of unsupervised machine learning process. This means that the task is not to approximate a function with known outputs, but instead to try to approximate some pattern in the input data itself. Clustering in particular is the task of splitting data into groups. In this thesis, we make use of K-means clustering [24], which is a method of creating groups, or clusters, out of data through calculating points which represent the centroid or center of the different clusters. K-means clustering finds K different clusters, each represented with a centroid of the mean of the points in the cluster. In this thesis we use K-means clustering in order to extract palettes from images.

2.3 Animation via Machine Learning

Our proposed task is not one that has previously been researched specifically. However, there is existing research on adjacent tasks which informed the work in this thesis.

2.3.1 Video Frame Interpolation

Video Frame Interpolation is a task with the goal of predicting a frame between two adjacent frames in a video. Machine-learning based methods have shown

success on this task, benchmarked on real-life video [3], [23]. Most relevant to our work, Siyao Li et al. [22] introduce the problem of video interpolation on animated video. Their method uses segment-guided matching and a recurrent flow refinement network to generate the interpolated frame. This task, however, differs from ours since these methods only produce a single frame at a time via interpolation whereas we transform frames from sketches to fully rendered frames.

2.3.2 Video Generation

Video generation involves a model which predicts future frames given a set of prior frames. There are many approaches in prior work to achieve this goal. VQ-VAEs [32] have shown success at this task from their conception, as the original paper demonstrates their ability to generate future frames conditioned on prior frames. Expanding upon this work, a number of researchers have used transformers to predict future frames in the latent space of a VQ-VAE [35], [48]. This system would not fit into an animation workflow, due to the effort required to render several frames at the start of an animation, and because of the loss of control over the motion itself.

2.3.3 Video-to-Video Synthesis

The field of video-to-video synthesis is concerned with mapping one video domain to another [50]. An example of this would be mapping semantic segmentation masks to fully rendered video or mapping video of an action performed by one human onto video of another human. Our task can be considered a subtask of video-to-video synthesis as both the input and output are videos. However, our task relates specifically to animation and not to the more generalized video-to-video synthesis of prior work.

Generative adversarial networks (GANs) and methods built on them have previously been applied with good success to several tasks in this field [26], [49], [50]. But GAN-based models cannot be applied to new subjects without fine-tuning to the new subject. In our case, we want to generate a specific output that's strongly conditioned on an input sketch. As such, a GAN would be less

suited to this task than more freeform animation. However, the structure of a VQ-VAE should prove well-suited to this task without requiring a second discriminator model.

2.3.4 Image Animation

Image animation refers to the task of transferring the motion from a video onto an image. The result of this is the subject of the image moving in the same way as the subject of the video. Yoon et al. [54] present a method of animating humans from a single image guided by body poses. Their method puts an emphasis on preserving the identity of textures and garments in the synthesized images. Siarohin et al. [43] demonstrate a method which transfers motion from a driving video onto a source image, using keypoint detection and local affine transformations, which are transformations that preserve lines and points locally. The keypoints and their transformations provide additional context to the generator model that produces the frames. However, sprite animation, as described in section 2.5.1, often involves stylized motion that cannot be described with the same methods that perform well with real-life video. An artist may choose to exaggerate shapes and certain aspects of movement to achieve their desired effect to imbue the animation with feelings of weight, speed, or emotion [46]. We found in our experiments that the work of Siarohin et al. performed less well on these stylized animations as we demonstrate in chapter 3.

2.4 Conditional Image Generation

Chapter 4 is about the generation of artistic images given a sketch and a palette. The generation of images given some context is called conditional image generation and is a field with significant amounts of prior research which we used to inform our work. Common conditional image generation tasks include conditioning on segment masks, edges, depth maps and text descriptions. This section is also relevant to the work in Chapter 3 as one could think of the task of filling in sketches in an animation as an extension of conditional

image generation.

2.4.1 Creating conditional features

Extracting Sketches

For the purpose of training a model conditioned on both sketches and colour palettes, we require both for every image in the training data. However, we found no dataset that contained hand-curated triplets of images, sketches and palettes. Therefore, we decided to generate both the sketch and palette. Both the task of generating sketches and the task of generating palettes have existing approaches. Sketches are an intermediate step in the process of creating an artwork. What constitutes a sketch can vary greatly across artists and pieces, but the purpose of a sketch is to make some of the broader artistic decisions early on. This helps the artist have context for further decisions, as well as allowing them to correct things that look off before more time is invested. Sketches often take the form of rough linework that may include strokes only meant to aid an artist in making more technically accurate artistic decisions. This is not always the case, but for the purpose of this thesis, we went with this kind of sketch.

Prior work has generated lineart and sketches for images using edgemaps as a proxy in the past [13], [15]. This method creates thin lines where there is high contrast in an image. This results in a sort of lineart in some cases, but can produce unwanted results. One case where this type of method performs poorly is when lineart is already present in the image. In this case, it may outline the lineart itself, which is undesirable. It also fails to generate sketches appropriate for certain artstyles, such as pixel art. Additionally, this creates thin lines which run sparsely along the contours of an image. This is not a common method of sketching, as creating such detailed and high-fidelity lines contradicts the idea of using sketches to quickly make some broader artistic decisions. Artists frequently sketch with many overlapping strokes and emphasize certain strokes, making heavy use of abstraction in early stages [8]. To achieve something closer to this version of a sketch, we adapt the edgemap

method. We use Canny edge detection [4] to find the edgemap. To do this we must first blur the image. Instead of running Canny edge detection only once at one level of blurriness, we run it multiple times at different blurriness levels. This results in a softer looking sketch which is more abstract. However, it still suffers from some of the same issues as typical edge detection, it will sometimes outline lineart, and struggles for some art styles.

Extracting Palettes

Palettes are a list of colours that are representative of the colours in an image. In certain situations, an artist may already have a desired colour palette in mind for their work before committing to a larger piece. The practice of thumbnailing can bridge the gap between choosing a palette and applying it in a final piece, allowing an artist to preview the effect it has [8]. In Chapter 4 we present initial research steps towards the automation of the generation of thumbnails based on sketches and palettes. As we discussed above, we need palettes in order to condition on them. There is no database of image-sketch-palette triplets, so we must generate both sketches and palettes. In order to generate palettes we make use of K-means clustering (see section 2.2.3) on RGB format images. We chose to use K-means clustering as it is a known method of producing palettes given an image [58]. Other methods use greedy algorithms to create a palette given an image [12]. On large images, K-means is too slow to use to compute palettes [58]. However, for the purpose of creating thumbnails and character sketches, low fidelity images are acceptable as they are not the end product, but instead a tool for ideation. We are therefore able to use K-means clustering. We also note that in the literature, palette extraction is largely proposed for the task of image editing, specifically recolouring an image. This is fundamentally a different task than ours, and many methods of palette extraction are designed to facilitate this goal, such as making all blue objects in an image green.

2.4.2 Text-conditioned Image Generation

Text-conditioned image generation models have seen a boom in popularity recently. Both the discrete VAE based model DALL-E [37] and the diffusion-based models GLIDE and DALL-E 2 [30], [36] have shown a strong ability to generate different art styles through text-to-image generation, however they can't be sufficiently conditioned on temporal information such as adjacent frames for use in our task of rendering in-between frames. This method is also not sufficient for the task of generating images conditioned on sketches and palettes.

2.4.3 Image colourization and colour Editing

Changing the colours in an image without altering its semantic information is a task which has a history, and lends context to the task of colourizing a sketch [1]. Generating an image given a sketch can be considered a subtask of this task. However, colourizing using only sketches and colours is a task which provides less information to a model, as in more typical image colourization tasks a model receives either a greyscale image or a full-colour image to recolour.

2.4.4 Sketch-conditioned Image Generation

In a seminal paper, Isola et al. propose pix2pix, a generative adversarial network [11] which can synthesize images conditioned on various things, including edgemap sketches [15]. pix2pix produces a number of artifacts which makes it a difficult tool to use for the ideation process of creating artwork, as the artifacts can alter the colours present in the image.

Sangkloy et al. propose Scribbler [39], a method of generating images based on sketches which have had colour strokes painted onto the sketches to imply an object should be that colour. Scribbler is a GAN-based network and was trained on photorealistic images. When applied to artwork it produces images which are rendered in a photorealistic way, albeit with significant artifacts in the form of soft gradients in colour. The results are nonetheless impressive,

but the model does not adhere well to the colour added onto the sketch. It will use similar colours, but this deviation from the chosen colour as well as the soft gradient artifact makes this method less useful for thumbnails where a designer may want specific colours to communicate their ideas.

Similarly, Ci et al. [6] use a GAN to generate images using sketches and colour scribble hints in the same way as Scribbler. However, they specifically train on illustrations. Their method shows impressive results but also suffers from having subtle gradients and low fidelity to the specific colours a user scribbles.

More recently, diffusion models [53] have been prominent in conditional image generation [7], [13], [30], [56]. In their proposed system ControlNet, Zhang and Agrawala use edgemap-sketches and text prompts to generate images [56]. ControlNet operates on top of a large diffusion model to allow for the additional inputs, including sketches, to be used. Their method allows for a great deal of additional control when compared with the more standard prompt-based diffusion models.

Huang et al. propose Composer, a method for altering an image by decomposing it into eight different components, including a sketch and palette [13]. A user can then edit any or all of these components and output an altered version of the original image. The suggested framework shows good results as a tool for image editing, which could in theory be used to condition on sketches and palettes alone. However, they do not explicitly evaluate this set of conditions, and they have not made their model or code available. They also do not provide results on sprites or similar illustrations.

Both Composer and ControlNet are diffusion models [53] with 5 billion parameters. Recent research has cast doubts on the novelty of images generated via diffusion, demonstrating that large diffusion models have in some cases memorized images from their training data [44]. Data leakage is problematic when the output is expected to be relatively novel, such as when creating thumbnails. For example, plagiarism of character designs is an unwanted outcome to most artists.

2.5 Downstream Tasks

The research conducted for this thesis was done with certain downstream tasks in mind, which will be detailed in this section.

2.5.1 Sprite Animation

A sprite animation is a 2D animation used primarily in video games to give the illusion of motion in a virtual world. These can be important to the immersion and visual storytelling of a game. However, animation is a difficult skill to master and a time-consuming process. The process of frame-by-frame animation is described in Section 2.1.3. We believe that SketchBetween, which is introduced in Chapter 3 can be used to cut down on time spent rendering in-between frames, a part of the animation workflow which typically involves very little artistic decision making.

2.5.2 Thumbnailing for concept art

Artist sometimes produce thumbnails before committing to the creation of a larger artwork. This is true both for digital and traditional mediums. A thumbnail allows an artist to evaluate their ideas and composition before putting in the time and resources to complete a full piece. Often an artist will create many thumbnails, exploring different variations of a piece and challenging themselves to expand on ideas in multiple ways [8]. Thumbnails can vary in fidelity and abstraction, but the value of thumbnails is to outline a piece at a high level and make some of the more abstract artistic decisions before starting work on the final artwork. An artist may have ideas or feelings they want to express through an artwork. To this end, they may want to use shapes, colours and symbols that are understood by their audience as tools to evoke those ideas and feelings. Iterating on a thumbnail allows an artist to optimize the use of these tools in their work. Character designs can also be iterated on through the creation of thumbnails.

We propose that our tool, as described in Chapter 4, can allow artists to more quickly iterate on thumbnails. An artist can make changes to the

sketch and palette quickly, allowing them to explore different shape and colour combinations to effectively communicate their ideas.

Chapter 3

SketchBetween: Video-to-Video Synthesis for Sprite Animation

Games and other media often necessitate the creation of 2D animations for cutscenes, backgrounds, effects or sprites, which are animated art assets of objects or characters in a game. The creation of these types of art assets takes both a high level of skill and a great deal of time [46]. The process of 2D animation typically breaks down into first creating keyframes that define some action, and then filling in the frames between these keyframes to give the appearance of motion. This is a resource intensive process as it takes someone with a high skill level a long time, but parts of this process are not the most demanding of their skill.

We propose a system which takes as input fully rendered keyframes that define the design and art style of the subject, and sketches for each frame that define the motions in between these keyframes. The system then produces as an output a fully rendered sprite animation. Existing approaches to video generation and synthesis thus far have not adhered to this standard workflow for animation. Many of the existing methods use previous frames as a prior to generate future frames [32], [35], [48], which requires the full rendering of several adjacent frames. Others rely on transferring a complete motion onto a source image [43]. Both of these methods offer only one perspective of the subject at hand and any information required for the full motion relies only on what was present in the training set, which may leave a sprite off-model. We propose a task for the automated rendering of in-between frames given

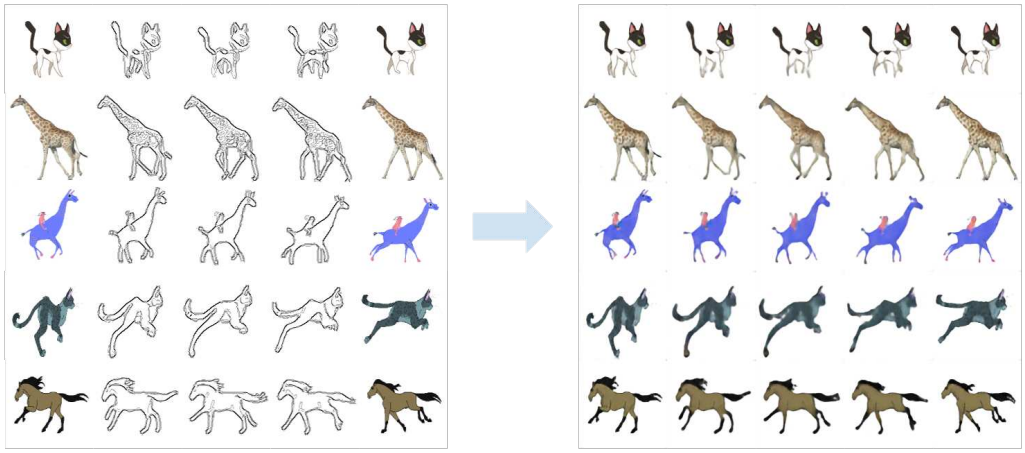

Figure 3.1: Selected input samples and the corresponding output from SketchBetween.

sketches to define the motion, and keyframes to define style. Both a keyframe at the start and at the end are included to capture information about parts of the subject that may be obstructed in either single keyframe. We additionally propose SketchBetween, a system to solve this task. Our system consists of a VQ-VAE [32] that encodes rich information in its latent space composed of the desired style and motion and leverages this to generate a fully rendered animation. As can be seen in Figure 3.1, SketchBetween is able to capture stylized motion while staying accurate to character designs.

There is no direct comparison we can make to other work, as our proposed task has not appeared in prior work. However, it builds upon work in adjacent fields and similar tasks. We evaluate SketchBetween against a strong baseline method adapted to our task in terms of the structural similarity index metric (SSIM) [52] and the peak signal-to-noise ratio (PSNR) measured between the original animation and the model's recreation. Compared to the baseline, we see a clear indication of its appropriateness for the target task.

3.1 System Overview

In this section we overview the process of training and applying our SketchBetween system. SketchBetween consists of a VQ-VAE which takes as input two keyframes and the sketches between them and outputs a fully rendered ani-

mation. We train this system using the MGIF dataset [42], which consists of animations of cartoon animals in motion. We chose five frames for our experiments, but experiments performed with three frames had comparable results. We implemented and trained the system using Keras [5]. Where parameter values are not specifically mentioned in this text we used the default values of the implementation.

3.1.1 Data

To train our model we require a dataset of existing animations. We chose the MGIF dataset [42]. The MGIF dataset consists of 1000 videos of cartoon animals walking, running and jumping. Each video has been resized to 128×128 pixels and has a white background. There is a good deal of variance in terms of species and art styles present in the data. We use the included train-test split, where 900 gif format videos are for training and another 100 are reserved for testing. The videos are of different lengths, and we had to exclude any that were shorter than 5 frames since we chose $N = 5$ for our experiments. This exclusion criteria applied to 33 videos in the training set and 2 in the test set.

Given our problem formulation in which we transform from keyframes and sketch in-between frames to a final animation we needed to reprocess the data. A dataset including hand-drawn sketches of animations as well as the final output would be ideal. Datasets of still images and corresponding artist sketches exist in the literature [21], [51], [55]. However, no such dataset exists for animations. To generate the input to our model we sample N frames of an animation and generate sketches of all but the first and last frames using Canny edge detection [4], inspired by the method used to train the pix2pix image-to-image translation method pix2pix. In order to be able to generate convincing sketches for the many different art styles present in the dataset, we opted to average the Canny edges detected for four different kernel sizes (3, 5, 7 and 9). The generated input can be seen on the left-hand side of Figure 3.1. The task is to generate the full N frames, and as such we use the original frames not processed into sketches as our expected output during training. To achieve

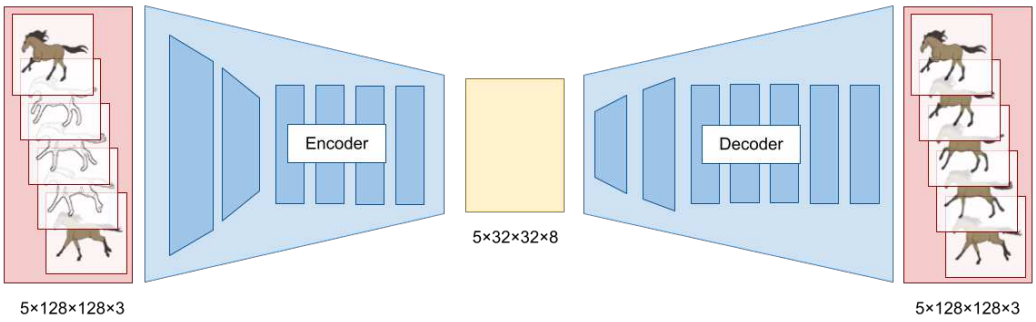

Figure 3.2: Diagram of the structure of the SketchBetween model.

better generalization during training, we augment the data by randomly shifting the hue and saturation as well as flipping horizontally before the sketch generation step. Each of these augmentations was applied with a 50% chance. The hue was shifted randomly by up to 180° and the saturation was randomly increased or decreased by up to 20%.

3.1.2 Model

Our problem formulation requires a model which can take in N frames where frame 0 and frame $N-1$ are fully rendered and frames 1 to $N-2$ are sketches. This model must then output a fully rendered animation. We chose to train a vector-quantized variational auto-encoder (VQ-VAE) DBLP:journals/corr/abs-1711-00937 due to their success as a generative model, particularly in the task of art generation and representation [37], [40]. The inputs to our VQ-VAE are the stacked input frames in RGB format consisting of the keyframes and the sketches between them. The size of our input, barring the batch-size dimension, is $5 \times 128 \times 128 \times 3$. We chose to use RGB because this is the default format of the MGIF dataset. However, this method should work just as well, if not better, if the images are transformed into the HSV format. We expect this due to the previous success of the HSV format applied to sprite-generating procedural content generation (PCG) tasks [10].

An overview of the model structure can be seen in Figure 3.2. The encoder consists of six layers of convolutions with ReLu [29] activation functions and batch normalization [14] between each layer of convolutions. We use mostly

$3\times3\times3$ kernels, analogous to the encoder structure of GLIDE [37], throughout our encoder. The exception to this structure are layers of $1\times1\times1$ convolutions in the final two layers to adjust the size of each encoding vector, much like in PixelVQ-VAE [40]. To increase the size of the patch each encoding vector represents, we use a stride of 2 along the x and y axes of each frame in the first two layers. We found that the performance of our model was sensitive to the dimensionality of our encodings (D) and the size of our codebook (C), and our final values for these hyperparameters were $D = 8$ and $C = 256$. The progression of the filter sizes is $3 \rightarrow 32 \rightarrow 64 \rightarrow 64 \rightarrow 128 \rightarrow 64 \rightarrow D$. The decoder consists of transposed convolutions with ReLu and batch normalization to project the encoding vectors to the output images. The structure is similar to the encoder, with $3 \times 3 \times 3$ convolutions in most layers. The second-to-last layer has $1 \times 1 \times 1$ convolutions to decrease the filter sizes towards the end goal of having only 3 filters for the red, green and blue channels of the output images. However, we found empirically that the final decoder layer performed better with a $3 \times 3 \times 3$ kernel size. The progression of the filter sizes for the decoder is $D \rightarrow 128 \rightarrow 64 \rightarrow 64 \rightarrow 64 \rightarrow 32 \rightarrow 16 \rightarrow 3$. We additionally found that we achieved better performance with a slightly larger decoder than encoder. We anticipate this is because the task of generating an embedding representation from the animation is in some way easier than generating the fully rendered animation from the embedding representation.

3.1.3 Training

The model was implemented and trained using Keras [5]. To train Sketch-Between we use a loss based on the structural similarity index metric (SSIM) [52], defined as $1 - SSIM$, which produced better recreations than a mean-squared-error (MSE) loss. We optimize with a lookahead [57] adam optimizer [19]. The model is trained for 100 epochs, with a learning rate of 0.001.

The empirical analysis used to determine certain parameters and parts of the model structure were performed on the training set SSIM loss.

3.2 Evaluation

We created SketchBetween as part of a push to make 2D animation more accessible. Therefore, human evaluation would be ideal, but as an approximation we provide two forms of quantitative evaluation of our proposed contributions. To evaluate SketchBetween we construct a baseline from a method developed for a related task. To evaluate our proposed problem formulation we perform an ablation study to validate the efficacy of our chosen formulation for the task.

To evaluate our model on the test set we take every five neighboring frames, including all overlapping sections, and generate inputs for our model. The numbers presented are the reconstruction metrics of only the three in-between frames, excluding the first and last frame as their reconstruction is less relevant to the task. The metrics improve marginally if the two keyframes' reconstructions are included, but we felt the relevant component of our model was the reconstruction of the sketched in-between frames. We use two metrics to evaluate the quality of reconstructions. The structural similarity index metric (SSIM) [52] as well as the peak signal-to-noise ratio (PSNR) between the original frame and the recreation. The SSIM measures the structural similarity between the original and the reconstruction in terms of luminance, contrast and structure. The PSNR measures the ratio between the maximum values of a signal and the noise present. As such, it gives a low value when the value of noise in a signal is stronger and a high value when noise is less present. This is useful to evaluate the amount of noise introduced via a process, such as reproduction through a VQ-VAE.

3.2.1 Baseline

We evaluate our method against a baseline we constructed by adapting a first-order motion model (FOMM) [43] to our task. FOMM transfers motion from a video onto an image. It is only trained on videos which drive the motion, and makes the assumption that an input image derives from the same distribution. To adapt it to our task we train the FOMM on a dataset consisting of the

original MGIF dataset with the addition of sketched versions of each gif using our sketch generation method. This ensures that keypoint detection is trained on both sketches and rendered images simultaneously and therefore should provide a model which can transfer sketch motion onto a rendered keyframe. We use this baseline to transfer the motion from the generated sketches of the test set onto the first fully rendered frame. We average the measured SSIM and PSNR over every frame in the generated animation. FOMM generates each animation in full from the sketched version and the first frame so there is a difference in the number of datapoints being evaluated as SketchBetween generates 3 frames from sketches for every 5 adjacent frames.

3.2.2 Ablation Study

We perform an ablation study to show the efficacy of the key components of our problem formulation. Ideally we would use human evaluation, but as an approximation our ablation study evaluates the reconstruction metrics of the ablated models with and without certain components. Firstly, we compare our system with and without the sketches between the keyframes. To do this we train a SketchBetween model on our dataset without generating sketches for the in-between frames, leaving them blank. We did this because we believe the sketches provide valuable information about the motion present in the animation. Secondly we compare our system with and without the second keyframe in place. To do this we train a SketchBetween model on our dataset but generate a sketch for the final frame in place of the fully rendered image. We believe that both the first and last keyframe are of value to the model to properly handle cases where parts of the subject are obstructed at the start or end and become revealed during the motion. In both cases the metrics are reported on a version of the test set which has the same ablation to its structure.

Table 3.1: Comparison between our method and baselines on the MGIF dataset. ↑ means that a higher value is better.

Model	SSIM (↑)	PSNR (↑)
FOMM	0.799	19.46
Ours	**0.943**	**27.48**

Table 3.2: Comparison between our complete method and ablated versions of it. ↑ means that a higher value is better.

Model	SSIM (↑)	PSNR (↑)
No sketch	0.883	20.19
No final image	0.938	25.4
Full model	**0.943**	**27.48**

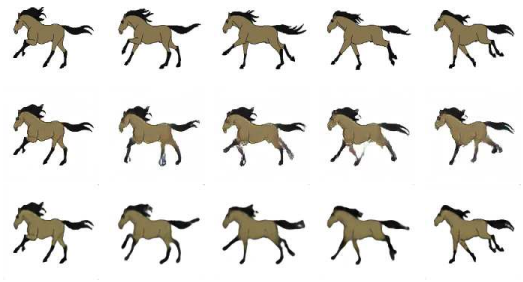

Figure 3.3: Visual comparison of the recreation of a horse running animation. The top row is the original animation. The middle animation is the FOMM recreation and the bottom row is the SketchBetween recreation.
Three rows of images. The top row shows an artist rendered animation of a horse running. The middle row shows a recreation of that animation with the FOMM baseline. The bottom row shows a recreation of that animation with SketchBetween.

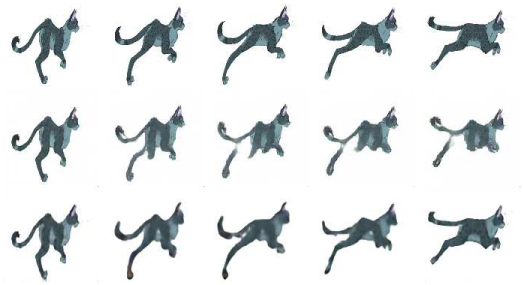

Figure 3.4: Visual comparison of the recreation of a cat leaping animation. The top row is the original animation. The middle animation is the FOMM recreation and the bottom row is the SketchBetween recreation.
Three rows of images. The top row shows an artist rendered animation of a cat running. The middle row shows a recreation of that animation with the FOMM baseline. The bottom row shows a recreation of that animation with SketchBetween.

Figure 3.5: Visual comparison of the recreation of a pixel-art donkey jumping animation. The top row is the original animation. The bottom row is the animation generated by SketchBetween.
Three rows of images. The top row shows an artist rendered animation of a pixel-art donkey jumping. The middle row shows a recreation of that animation with the FOMM baseline. The bottom row shows a recreation of that animation with SketchBetween.

3.3 Results

A comparison of the results of our model and the baseline can be seen in Table 3.1. Our method achieves higher scores on this task for both metrics. Additionally, a visual comparison of selected representative recreations can be seen in Figures 3.3 and 3.4. The baseline fails to capture changes in the shapes of objects that are not affine transformations of a part of the subject, but instead animated stylistically, such as the horse's mane and tail flowing behind it.

The results of our ablation study on the task formulation can be seen in Table 3.2. We identify that the inclusion of all elements does provide the best results according to our metrics. The inclusion of the sketches is especially important. This is in line with the intuition of sketches providing valuable information about how the subject should be rendered in the in-betweens. However, for our dataset, the inclusion of the final keyframe is not hugely significant. This is perhaps because the animations provided in the MGIF dataset have similar motions where obstructed portions of the subject are most frequently limbs, which are typically symmetrical.

3.4 Case Study

Since SketchBetween is intended to be used as a tool by animators we performed a case study of how it might work on real-life examples, such as walk

cycles for game characters. We obtained 4 animations from 3 different artists, where the artists rendered the first and last frames and sketched three frames of motion in between. The artists were asked to provide a short animation of an animal in motion. Artwork was provided by Twitter users @TeethyFish (fish, chinchilla) and @K3rryberry (bunny) and the author of this thesis (fox).

We ran these animations through SketchBetween to see how the model handles human-made sketches. The results of this can be seen in Figure 3.6. Notably some patterns on the animals that would have been added to the sketches via the canny edge detection are not present in the human-made sketch. The model still attempts to generate them, but they are less consistent and crisp than patterns seen in Figure 3.1. Prior to running SketchBetween on these animations we were convinced that it would work best on the chinchilla and fox, and worst on the fish. We were surprised to see that the model performed best on the fish and the fox and worst on the chinchilla. We speculate that the large areas with low texture on the chinchilla may have contributed to the amount of artifacts in the output. It should be noted that while the model performs well on the fox, this was drawn by the thesis author and thus they could have unconsciously drawn a well-suited example to SketchBetween. Notably this example includes smaller motion than in the other examples. Conversely, the bunny example is of a creature jumping like in Figure 3.5 which depicts a large motion at a low framerate which the model seems to handle poorly.

The artists have visibly different sketching styles but notably all the human-made sketches are fairly clean and do not contain much of the artifacts that artists may use to assist in structuring the form of their artwork. It is therefore unclear to what extent SketchBetween can handle the variance of sketching methods used by different artists. We leave a full investigation of this topic to future work.

3.5 Discussion and Future Work

We have shown through our experiments that our model outperforms a baseline for our task, and that the formulation of this task provides sufficient

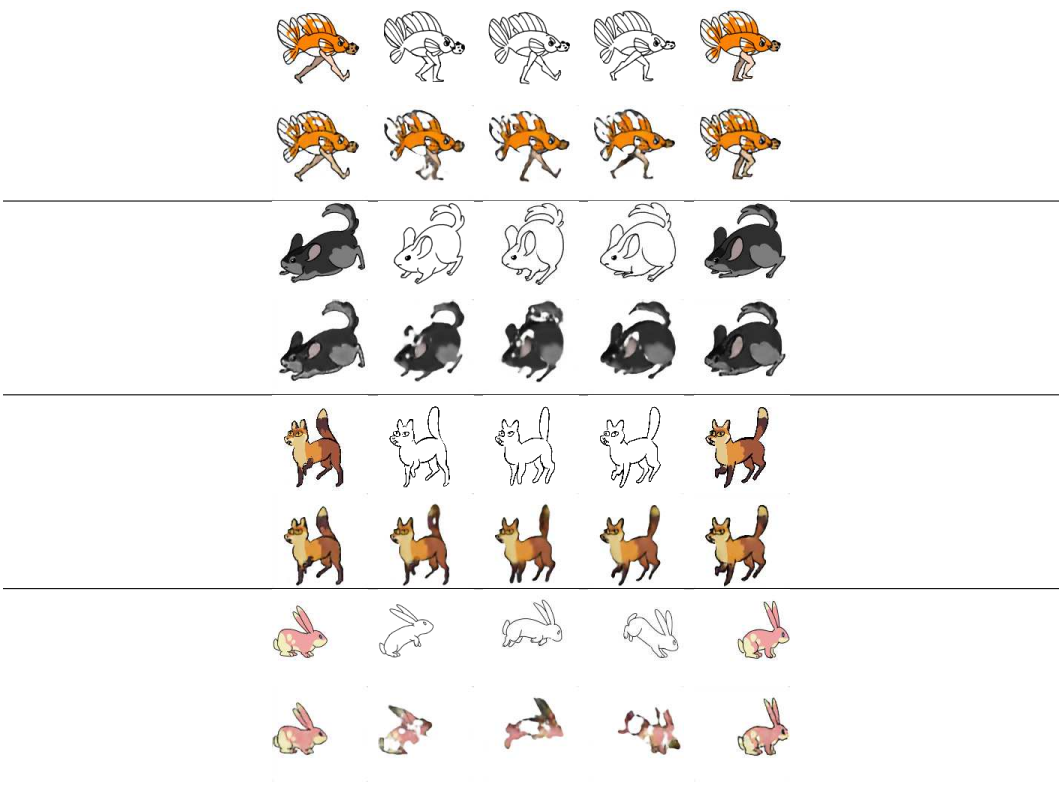

Figure 3.6: Animations by various artists. For each section the top row is an animation with artist-drawn sketches between the first and last frame. The bottom row is the SketchBetween output for this animation.

information to construct models with good outcomes. The system may therefore be able to save artists' time when producing animations. It is difficult to quantify how much time can be saved by using a tool such as this due to the high variance in how artists operate, resulting from differences in process and level of skill. However, the proposed tool could address one of the more tedious parts of the animation process.

Our model does have certain shortcomings. There are a number of videos in the test set for which our model performs quite poorly. These are videos with motions that are not common in the training set, one example can be seen in Figure 3.5. Jumping is an action that is not common in the dataset and the particular stylization of this donkey's jump makes it an outlier. This effect can also be seen in the bunny in Figure 3.6.

Additionally, our output videos are more blurry than desired, and details tend to be lost. A future avenue of research could be to explore diffusion-based video-to-video synthesis, since our method is analogous to a sort of inpainting of the in-between frames, and diffusion models have shown success at inpainting [30]. Another potential direction is to apply an adversarial loss [26], [49], [50], or to use a more refined loss than our SSIM-based one. For example, one based on analogs between frames, inspired by previous work in deep visual analogy-making [38].

The case study exposed further avenues for improvement. It highlights a weakness of the dataset used to train the model, which is that it consists of only three types of movements: walking, running and jumping. The dataset also largely contains animations that are produced at a high frames-per-second (fps) and therefore only have subtle changes between adjacent frames. The effect of this is that animations with more drastic changes, such as the bunny in Figure 3.6 are poorly handled. This could be mitigated with more data, but also by omitting some number of frames during training to artificially decrease the fps. The case study also suggests that adding more variance into the generated sketches to capture more art styles would be valuable. Adapting this method to be more iterative could also prove to be useful, as design tools with iterative design processes have been shown to be effective [17]

The ablation study shows that while the system performs better with the sketches available between the keyframes, there is value in exploring a system which omits this additional work for artists. In animations with a high fps this may shave additional time off, however it may negatively impact animations with large motions between keyframes such as the bunny in Figure 3.6 where there is more motion information in the sketches.

3.6 Conclusions

This chapter proposes a problem formulation and model solution for sprite animation generation. We focus on rendering sketches between rendered keyframes. We additionally propose a method to do this using a VQ-VAE called Sketch-Between. We showed that our method outperforms a strong baseline, and that the formulation of the task is valuable through an ablation study. Given our results we believe that this research direction has the potential to help democratize 2D animation in game development.

Chapter 4

Generation of Thumbnails Conditioned on Sketches and Palettes

In this chapter we present the second contribution of this thesis, a preliminary investigation of the generation of images conditioned on sketches and palettes. We provide an overview of the motivation and justify the value in solving such a task. We then detail the structure of a preliminary system designed for this task. The system is evaluated via a preliminary error analysis where its output is analyzed to identify shortcomings. We propose mitigations and outline future work.

4.1 Introduction

When creating visual artwork there is often an ideation phase. Artists may put time and effort into deciding what the subjects of a piece will look like, especially to optimize the effect of visual language. An artist may want to make an enemy in a game look as scary as possible, but the protagonist of a children's book look cute. Achieving the desired effect sometimes requires iterating on a character design. Similarly, the visual composition of a piece may be finalized by considering lower-fidelity versions to preview the effect of a certain colour palette or composition of subjects in the piece. This too is often an iterative process, resulting in multiple of these lower-fidelity pieces called thumbnails [8].

There exists a body of work focusing on similar tasks to the one we propose. However, they are not easily used by artists for the purpose of quickly iterating on design decisions. The task of recolouring artwork has been extensively researched, and has many existing approaches. However, prior work generally includes as an input an image where objects have already been coloured in, pixels contain information about colour intensity, and segments of colour blocks exist [1]. Other work exists that does go from lineart to rendered works, but they require that the input indicates where colours should go, and they do not adhere well to the specific chosen colours [6], [39].

We propose the task of automatically generating thumbnails given a sketch and a colour palette. This could allow artists to iterate faster, and the generated works may also inspire an artist with a variation they had not thought of themselves. We additionally provide a preliminary system for this task. The system consists of a VQ-VAE [32] trained on images and sketches and palettes generated from those images. The preliminary system performs somewhat poorly, and we provide an analysis of the output as well as proposed mitigation strategies and future work.

4.2 System Overview

In this section we overview the process of training a system for the proposed task. The system consists of a VQ-VAE trained on sketches and palettes to produce rendered images. We trained the system using the MGIF dataset [42], which contains animations of creatures in motion. This system is implemented and trained using Keras [5]. Where parameter values are not specifically mentioned in this text we used the default values of the implementation.

4.2.1 Data

Training a model on the proposed task requires relevant data. There is no dataset containing the equivalent stages of an artist's workflow, namely sketches, palettes and subsequent finished thumbnails. Furthermore, there is no dataset containing image, sketch and palette triplets. Fortunately we can

generate proxies for the missing sketch and palett information, such that we require only images. Most image datasets consist of photographs, while we are interested in datasets containing digital illustrations or high-quality scans of physical artwork. At this preliminary stage, we propose using the MGIF dataset [42] which we used in Chapter 3, as it contains images similar to certain thumbnails, namely characters in a neutral pose with minimal rendering.

MGIF

The MGIF dataset is a dataset of illustrated animations of animals in motion [42]. It is a useful dataset for this project due to the presence of eccentric and colourful characters on a white background, illustrated in a variety of different styles. The dataset animations are variable in length, but were split into individual frames for this project. In total there are 1000 animations in the dataset. The authors of the dataset provide a split of 900 animations for training and 100 for testing. We reserved 100 of the 900 training animations for validation. We extracted the frames from each animation, resulting in 20713 individual images in the training dataset, 939 in the validation dataset and 2426 in the witheld test dataset.

4.2.2 Preprocessing

In order to train a model to generate images conditioned on sketches and palettes, we must have triplets of sketch, palette, and image on which to train. However, no such dataset exists, so we opted to estimate both the sketches and palettes.

Sketch Generation

Sketch generation is the process of generating approximations of sketches given an image. This process is described in Chapter 3. We use the same method in this work.

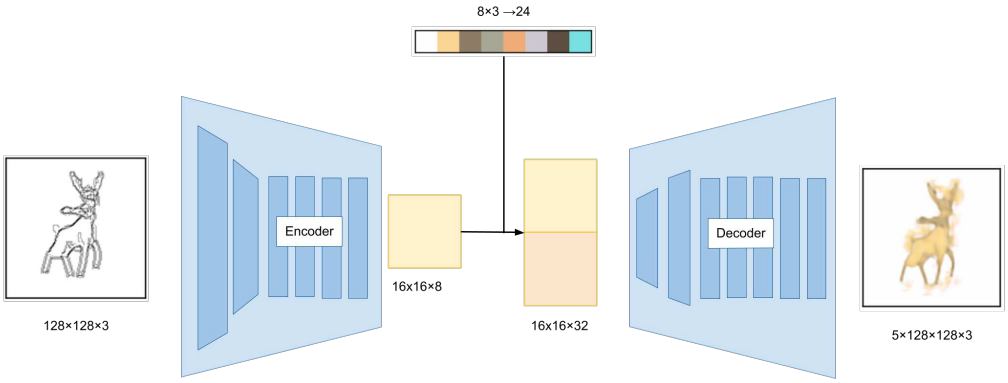

Figure 4.1: Diagram of the structure of the VQ-VAE-based model. Each vector in the quantized latents has the entire palette appended to it.

Palette Generation

Palette generation has been attempted in prior work (see Chapter 2.4.1). For our approach. we used K-means clustering [24] on the image pixel data in RGB format to identify 8 clusters of colours present in the image. The number of clusters we chose was somewhat arbitrary and based only on a visual analysis of the data. The illustrations are largely flat and cartoony, resulting in relatively few colours present in each character. The centers of the identified clusters are the colours used to create the palette. We arrange the colours in a palette in decreasing order of cluster size. We used the scikit-learn implementation of K-means [34]

We chose to cluster the colours in the RGB colour space instead of the HSV colour space because the RGB colour space has no non-unique colours, whereas in the HSV colour-space there are multiple representations of certain colours, namely all colours with zero saturation or value. The palette is converted to HSV format before being used in the system.

4.2.3 Model

We propose using a VQ-VAE due to its performance in our previous work relating to sketch-conditioned generation in Chapter 3. For this formulation,

we use an encoder and vector-quantizer $EVQ()$ which takes the input sketch $\hat{X}$ of size $H \times W \times 3$ and produces a quantized latent representation $Z = EVQ(\hat{X}) \in \mathbf{R}^{L \times R \times V}$, where L and R are the height and width of the latent representation and V is the size of each embedding. The dimensionality of the two inputs poses a problem. A $H \times W$ image, such as a sketch, has the shape $H \times W \times 3$, but a palette of 8 colours has the shape 8×3. These sizes are by default not compatible with each other when using a convolutional model, such as convolutional VQ-VAEs. However, there are ways to mitigate this. To condition the model on the palette as well, we flatten the 8×3 palette and append a vector of size 24 to every embedding vector in the quantized latents, producing a new latent of size $L \times R \times (V + 24)$. This is then passed to the decorder in order to produce a reconstruction of the original artwork. We include a diagram of the model structure in Figure 4.1

We tried a number of configurations of hyperparameters, and found that empirically the following structure worked best, although we believe that further work in tuning hyperparameters would be worthwhile. The encoder consists of a series of convolutional blocks, where each block is a convolutional layer followed by a ReLu activation [29], batch normalization [14] and a 10% dropout [41]. The kernel size is 5×5 throughout the network, with the exception of the final layer, which has a 1×1 convolution. The first three layers have a stride of 2×2 and the progression of features is $3 \to 32 \to 64 \to 64 \to 128 \to 8$ The encoder outputs an embedding which is fed to the vector quantizer. The vector quantizer has a learned codebook of size $C = 128$, and each vector in this codebook is of size $V = 8$. The size of the quantized embedding is $16 \times 16 \times 8$, and in order to include the palette in the context we append the 24 length palette vector to each encoding, resulting in a $16 \times 16 \times 32$ final embedding. This is passed to the decoder. The decoder has a very similar structure to the encoder, but instead of convolution layers it uses transposed convolution layers to project the encoding into image space. The first three layers use a stride of 2×2 to upscale the image back to the appropriate size, and the kernels are similarly 5×5 throughout the decoder with the exception of a 1×1 transposed convolution in the last layer. The

progression of features is $8 \rightarrow 128 \rightarrow 128 \rightarrow 64 \rightarrow 64 \rightarrow 32 \rightarrow 3$

4.2.4 Training

Our system is implemented and trained using the machine learning Python library Keras [5]. To train this system we use a custom loss function in place of the standard reconstruction loss for the VQ-VAE. This custom loss is a weighted sum of an SSIM-based loss, defined as $1 - SSIM$ and the mean squared error (MSE) between the original image and the output. The custom loss is thus defined as $\alpha(1 - SSIM) + \beta(MSE)$, and we empirically found an α of 0.7 and a β of 0.3 to work well. We previously used only a SSIM-based loss in Chapter 3. The addition of the MSE component is to improve the model's ability to correctly apply the colours in the palette to the image. SSIM was originally defined for grayscale images but is often calculated on the YUV colour-space, of which the first channel is the grayscale representation of an image. SSIM is averaged over all channels when calculated for a YUV image. This results in SSIM reflecting colour similarity, but the differences in the U and V channels independently may not reflect the difference perceived by humans as well as in the HSV colour space.

We optimize with a lookahead [57] adam optimizer [19]. The model is trained with an early stopping mechanism that stops the training if no improvement is observed for 20 epochs, starting after the 40th epoch. The model trained for 71 epochs before stopping due to this mechanism.

Data Augmentation

To achieve better generalization during training, we augment the data by randomly modifying it. Each batch of images is flipped with a 50% chance, and the hue and saturation are randomly shifted up or down by between 0 and 50% of their maximum values, with a 50% chance to do so per batch of images. We randomly select up to four additional transformations, zooming, shearing, shifting and rotating the image slightly, each with a 30% chance of application. Every augmentation done to the image is also applied to both the sketch and the palette as appropriate. We note that only hue and saturation

Table 4.1: Quantitative results from running our model on the MGIF dataset. ↑ means that a higher value is better.

Model	SSIM (↑)	PSNR (↑)
Ours	0.442	13.55

transformations are applied to the palette as the other transformations are unlikely to majorly affect the colour distribution.

4.3 Evaluation

This system was motivated by a desire to produce something useful to artists in the ideation and development of thumbnails. Therefore, the most reasonable way to evaluate it would be to perform a human subject study. However, this is not feasible at this time and perhaps ill-advised at such a preliminary stage. Therefore we use quantitative metrics measured on the held-out test portions of our datasets to evaluate the system's ability to reconstruct an image given the image's approximated sketch and palette.

In order to evaluate the models we use the structural similarity image metric (SSIM) [52] as well as the peak signal-to-noise ratio (PSNR) between the reconstruction and the original image as we did in Chapter 3.

This work is preliminary and this initial progress does not warrant evaluating against a baseline, or a human expert evaluation other than our own, as the resulting artifacts are still lacking in many ways. For this reason, we simply report the metrics for various hyperparameters as well as visualizations of outputs given these hyperparameters.

4.4 Results

In this section we present the results of our evaluation of this preliminary work. In Table 4.1 we present the SSIM and PSNR values of the model on the held-out training set. When contrasted with the metrics for a similar model on a similar task in Table 3.1 in Chapter 3, it becomes clear that this model has a significantly lower capacity to reconstruct the original image.

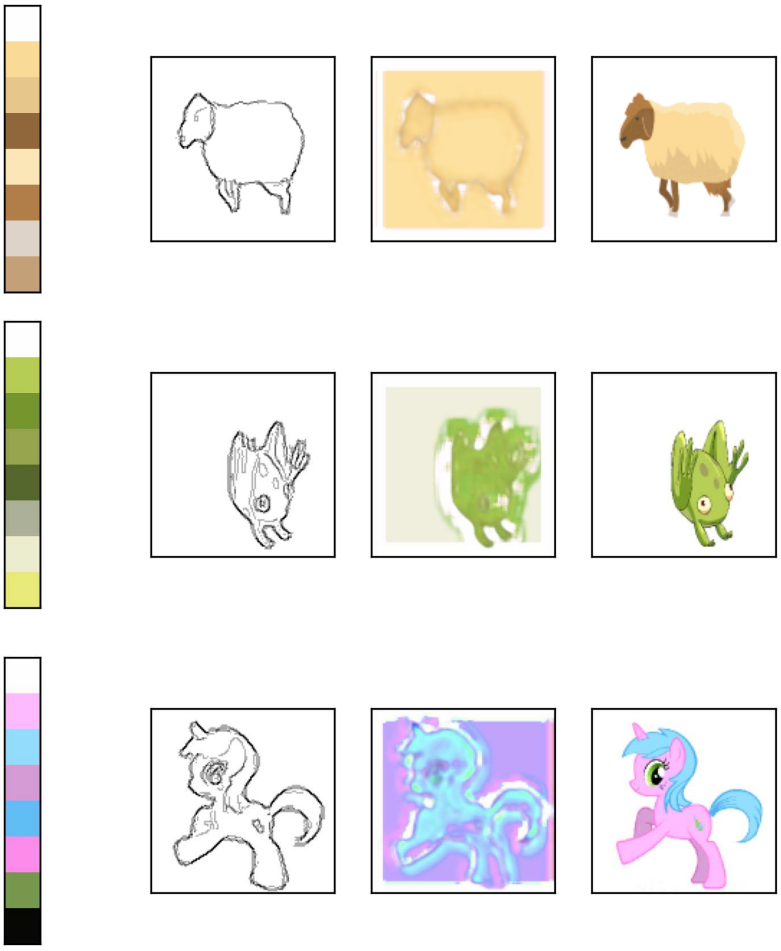

Figure 4.2: A cherry-picked sample of images from the test set which demonstrate the model inappropriately colouring the background. There are three rows of four images each, each row is one sample. In each row, from left to right are the input palette, the input sketch, the model output and the actual image from the test set.

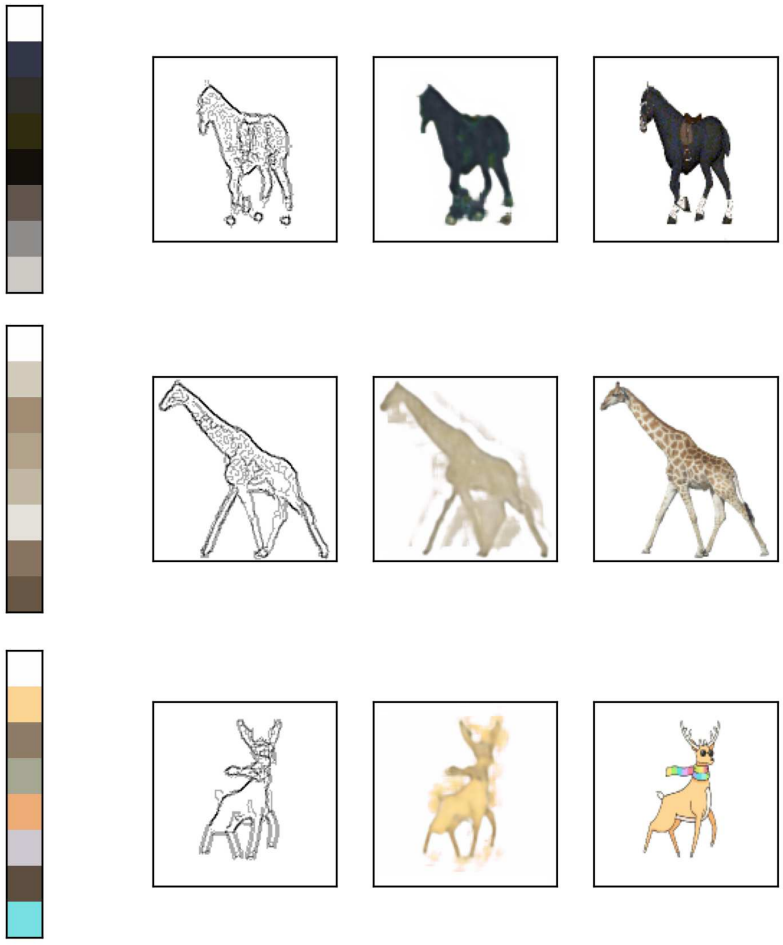

Figure 4.3: A cherry-picked sample of images from the test set which demonstrate the model only using one of the palette colours on the character. There are three rows of four images each, each row is one sample. In each row, from left to right are the input palette, the input sketch, the model output and the actual image from the test set.

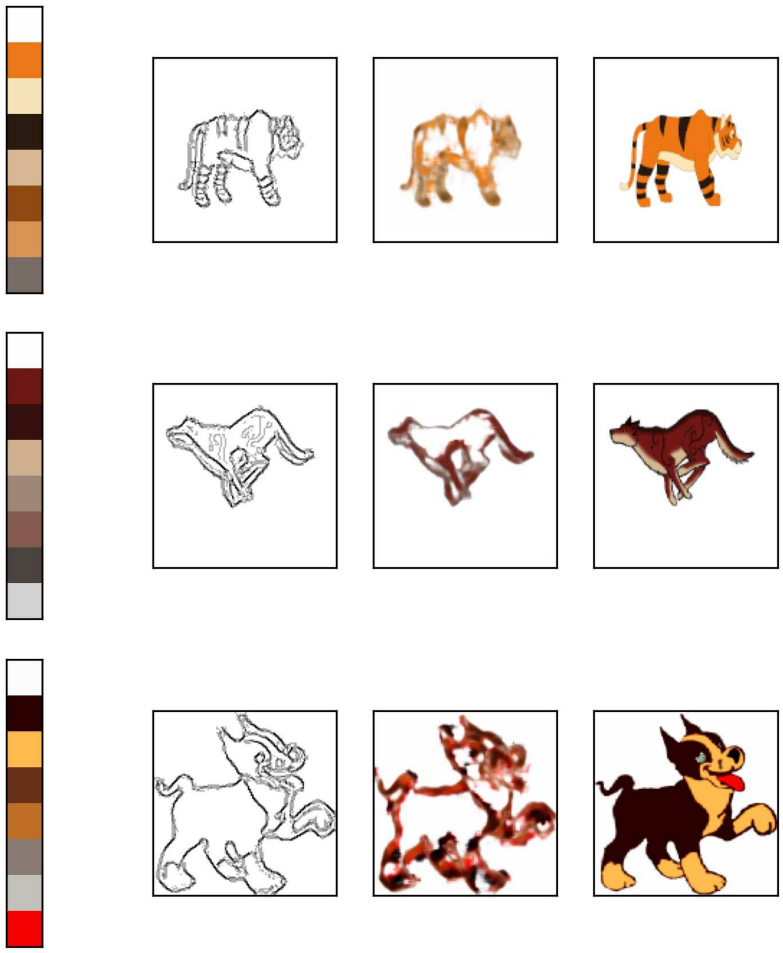

Figure 4.4: A cherry-picked sample of images from the test set which demonstrate the model leaving white space in the middle of a character. There are three rows of four images each, each row is one sample. In each row, from left to right are the input palette, the input sketch, the model output and the actual image from the test set.

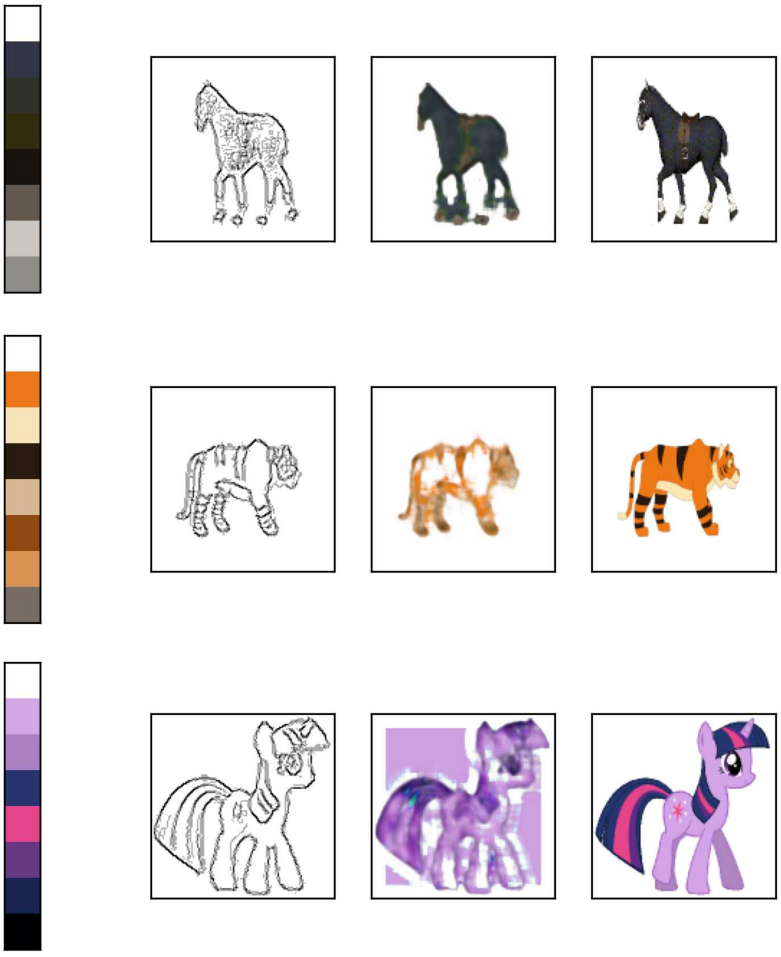

Figure 4.5: A cherry-picked sample of images from the test set which demonstrate the model overfitting to characters or patterns that are overrepresented in the training set. There are three rows of four images each, each row is one sample. In each row, from left to right are the input palette, the input sketch, the model output and the actual image from the test set.

In Figures 4.5, 4.4, 4.2, 4.3 we show cherry-picked examples of model outputs from the test dataset which exhibit some characteristic we find to be of interest. Namely, we have chosen these images as they exemplify problematic output from the model. The model produces, with little predictability, several types of unwanted artifacts. The groups of unwanted artifacts were determined by the author of this thesis through examining the output of the model on the test set and noting common errors.

The model sometimes fails to fill in the middles of creatures, as can be seen in Figure 4.4, and it also sometimes fills the background with the inappropriate colour as can be seen in Figure 4.2. In addition to this, the model tends to produce creatures which only contain the most common colour in the palette aside from the omnipresent white background, as is shown in Figure 4.3. However, the model does show some capacity for using more colours as can be seen in Figure 4.5. This may not be cause for celebration, however, as we noted that this largely happened in cases where the character had an expected colouration, such as an orange tiger with white stripes, or the character was a popular one. In both cases, we expect these colourations to exist in other animations present in the training set.

4.5 Discussion and Future Work

Our greatest takeaway from this project is that the task of generating images given a sketch and a palette is difficult. SketchBetween [25] (See Chapter 3) was more easily able to learn to colour in the sketches in the in-between frames with a similar, if not more powerful, model on this task. The difficulty of this task has also been noted before in a survey by Anwar et al. [1]. With less information available, and the additional task of needing to come up with the colour layout, the model needs to have a stronger understanding of how colours generally appear in images. To this end, we believe that a larger and more diverse dataset may prove useful as we found that the validation loss was significantly higher than the training loss despite our regularization and data augmentation methods. The dataset we used includes a good deal of variance

in colour and shape, but it has the property that most of the canvas is almost always white, so it may be easy for the model to get caught in a local minima of producing largely white images. This would produce a low mean squared error as well as a fairly low SSIM score for the white parts of the image. We also find that the model often leaves large amounts of white space within the body of a creature, potentially because the area is spatially far from the nearest lineart, and the model has lost context. This can be seen when comparing the creatures in Figure 4.4 to the creatures in Figure 4.3 where the sketch is much more dense. For the above reasons, we theorize that a transformer [47] could be used to increase coherency within the image. We have implemented a variant of a model suggested by Esser et al. [9] which employs a transformer to go from the latent space of one VQ-VAE to another. However, in initial experiments we find that this model performs even worse, which we believe is due to the increased complexity and capacity of the model, which could lead to overfitting. We believe that the most beneficial next step is to make use of a larger dataset of artistic images. However, we are hesitant to make use of further arbitrary image datasets as they are often scraped based on online search results with little care for data ownership or usage rights. This is the case for the MGIF dataset [42], which contains many licensed characters and artwork by people who have not consented to having their work used in this way. We believe that the choice of dataset must be purposeful and done with consideration for the origin of the data.

4.6 Conclusions

In this chapter we have introduced a preliminary system for the generation of images based on sketches and palettes. Our finding is that this task is more difficult than the task presented in Chapter 3, and a similar model is insufficient to produce good results with the same dataset. We analyze the output of the system, identify specific shortcomings and propose mitigations that may be applied in future work.

Chapter 5

Conclusions and Discussion

5.1 Potential Impact

We hope that our work thus far, and continued research into this topic, can prove useful to artists as well as researchers in this field. In this section we evaluate the potential impact of what we put forth in this thesis, both in terms of value and risk to stakeholders.

5.1.1 Value in use by artists

Our goal is to provide artists with systems that can be easily adopted to facilitate their existing workflows. We consider the work presented in this thesis to be a significant step towards that goal for the artform of animation. The system we propose in Chapter 3 is formulated with short-form sprite animation in mind and has not been tested outside of this scope, but due to the overlap in workflows we believe that our work provides a jump-off point to adapt similar methods to long-form animation, such as for film and television.

The system proposed in this thesis provides animators with a tool to speed up their workflow, which we believe is most impactful for animators who work outside of the animation industry, such as indie game developers and hobbyist creators. Culture benefits from the work of these artists, and we are happy to put our time towards lowering the threshold to creation. On the other hand, the work in Chapter 4 is of limited value to artists, but provides valuable insight into the task of generating images from sketches and palettes and what challenges are present.

5.1.2 Risks

The work put forth in this thesis is done in good faith, but it is worth discussing issues which it may exacerbate. The automation of artistic processes has been criticized by artists. Firstly for the use of data which was misappropriated through online search engines without artists' consent [45]. Additionally, artists fear job displacement due to the emergence of tools which can create high-fidelity artwork. This effect is already being reported [59] and while the contributions of this thesis aim to mesh with existing artist workflows than existing AI systems, the contributions are ultimately intended to lessen the amount of work needed. Therefore this could decrease the number of available jobs due to the lower labor requirement.

5.2 Conclusions

In this thesis we have presented two new problem formulations for animation automation which adhere more closely to artist workflows than any existing work to our knowledge. We have additionally proposed systems to perform this automation through the use of VQ-VAE models.

Firstly, we proposed SketchBetween, which we showed outperformed a strong baseline. Additionally, we demonstrate that our problem formulation consists of valuable information for our system through an ablation study of the input to the system. Furthermore, we perform a case study of the system's performance on data from artists which provided valuable insight into biases and limitations of the system.

We additionally examine the related task of generating images conditioned on sketches and palettes. This task is compatible with the ideation phase of certain artist workflows, and existing research does not sufficiently address this step of the creative process. We provide a preliminary system to perform the task, and analyze the outputs as well as suggest future work based on limitations we identified.

References

[1] S. Anwar, M. Tahir, C. Li, A. Mian, F. S. Khan, and A. W. Muzaffar, *Image colorization: A survey and dataset*, 2022. arXiv: 2008.10774 [cs.CV].

[2] D. Bank, N. Koenigstein, and R. Giryes, *Autoencoders*, 2021. arXiv: 2003.05991 [cs.LG].

[3] W. Bao, W. Lai, X. Zhang, Z. Gao, and M. Yang, "Memc-net: Motion estimation and motion compensation driven neural network for video interpolation and enhancement," *CoRR*, vol. abs/1810.08768, 2018. arXiv: 1810.08768. [Online]. Available: http://arxiv.org/abs/1810.08768.

[4] J. Canny, "A computational approach to edge detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-8, no. 6, pp. 679–698, 1986. DOI: 10.1109/TPAMI.1986.4767851.

[5] F. Chollet *et al.*, *Keras*, https://keras.io, 2015.

[6] Y. Ci, X. Ma, Z. Wang, H. Li, and Z. Luo, "User-guided deep anime line art colorization with conditional adversarial networks," in *Proceedings of the 26th ACM International Conference on Multimedia*, ser. MM '18, Seoul, Republic of Korea: Association for Computing Machinery, 2018, pp. 1536–1544, ISBN: 9781450356657. DOI: 10.1145/3240508.3240661. [Online]. Available: https://doi.org/10.1145/3240508.3240661.

[7] P. Dhariwal and A. Nichol, *Diffusion models beat gans on image synthesis*, 2021. arXiv: 2105.05233 [cs.LG].

[8] C. A. Empire. [Online]. Available: https://conceptartempire.com/intro-to-thumbnail-sketching/.

[9] P. Esser, R. Rombach, and B. Ommer, *Taming transformers for high-resolution image synthesis*, 2021. arXiv: 2012.09841 [cs.CV].

[10] A. Gonzalez, M. Guzdial, and F. Ramos, "Generating gameplay-relevant art assets with transfer learning," 2020. DOI: 10.48550/ARXIV.2010.01681. [Online]. Available: https://arxiv.org/abs/2010.01681.

[11] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, *et al.*, *Generative adversarial networks*, 2014. arXiv: 1406.2661 [stat.ML].

[12] M. Grogan, M. Hudon, D. Mccormack, and A. Smolic, "Automatic palette extraction for image editing," Aug. 2018.

[13] L. Huang, D. Chen, Y. Liu, Y. Shen, D. Zhao, and J. Zhou, *Composer: Creative and controllable image synthesis with composable conditions*, 2023. arXiv: 2302.09778 [cs.CV].

[14] S. Ioffe and C. Szegedy, *Batch normalization: Accelerating deep network training by reducing internal covariate shift*, 2015. DOI: 10.48550/ARXIV.1502.03167. [Online]. Available: https://arxiv.org/abs/1502.03167.

[15] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, *Image-to-image translation with conditional adversarial networks*, 2016. DOI: 10.48550/ARXIV.1611.07004. [Online]. Available: https://arxiv.org/abs/1611.07004.

[16] A. Jain, J. Mao, and K. Mohiuddin, "Artificial neural networks: A tutorial," *Computer*, vol. 29, no. 3, pp. 31–44, 1996. DOI: 10.1109/2.485891.

[17] M. Jin, D. Gopstein, Y. Gingold, and A. Nealen, "Animesh: Interleaved animation, modeling, and editing," *ACM Trans. Graph.*, vol. 34, no. 6, Oct. 2015, ISSN: 0730-0301. DOI: 10.1145/2816795.2818114. [Online]. Available: https://doi.org/10.1145/2816795.2818114.

[18] D. P. Kingma and M. Welling, *Auto-encoding variational bayes*, 2022. arXiv: 1312.6114 [stat.ML].

[19] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2015.

[20] R. G. Kuehni, "Color space and its divisions," *Color Research & Application: Endorsed by Inter-Society Color Council, The Colour Group (Great Britain), Canadian Society for Color, Color Science Association of Japan, Dutch Society for the Study of Color, The Swedish Colour Centre Foundation, Colour Society of Australia, Centre Français de la Couleur*, vol. 26, no. 3, pp. 209–222, 2001.

[21] M. Li, Z. Lin, R. Měch, E. Yumer, and D. Ramanan, "Photo-sketching: Inferring contour drawings from images," in *WACV*, 2019.

[22] S. Li, S. Zhao, W. Yu, *et al.*, "Deep animation video interpolation in the wild," *CoRR*, vol. abs/2104.02495, 2021. arXiv: 2104.02495. [Online]. Available: https://arxiv.org/abs/2104.02495.

[23] X. Li, M. Cao, Y.-P. Tang, *et al.*, "Video frame interpolation via structure-motion based iterative fusion," *ArXiv*, vol. abs/2105.05353, 2021.

[24] S. Lloyd, "Least squares quantization in pcm," *IEEE Transactions on Information Theory*, vol. 28, no. 2, pp. 129–137, 1982. DOI: 10.1109/TIT.1982.1056489.

[25] D. Loftsdottir and M. Guzdial, "Sketchbetween: Video-to-video synthesis for sprite animation via sketches," in *Proceedings of the 17th International Conference on the Foundations of Digital Games*, 2022, pp. 1–7.

[26] A. Mallya, T.-C. Wang, K. Sapra, and M.-Y. Liu, "World-consistent video-to-video synthesis," in *ECCV*, 2020.

[27] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *The bulletin of mathematical biophysics*, vol. 5, no. 4, pp. 115–133, 1943.

[28] T. M. Mitchell, *Machine learning*. McGraw-hill New York, 1997, vol. 1.

[29] V. Nair and G. Hinton, "Rectified linear units improve restricted boltzmann machines vinod nair," vol. 27, Jun. 2010, pp. 807–814.

[30] A. Nichol, P. Dhariwal, A. Ramesh, *et al.*, *Glide: Towards photorealistic image generation and editing with text-guided diffusion models*, 2021. DOI: 10.48550/ARXIV.2112.10741. [Online]. Available: https://arxiv.org/abs/2112.10741.

[31] K. O'Shea and R. Nash, *An introduction to convolutional neural networks*, 2015. arXiv: 1511.08458 [cs.NE].

[32] A. van den Oord, O. Vinyals, and K. Kavukcuoglu, "Neural discrete representation learning," *CoRR*, vol. abs/1711.00937, 2017. arXiv: 1711.00937. [Online]. Available: http://arxiv.org/abs/1711.00937.

[33] C. Paul, *Digital art*. Thames & Hudson, 2023.

[34] F. Pedregosa, G. Varoquaux, A. Gramfort, *et al.*, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[35] R. Rakhimov, D. Volkhonskiy, A. Artemov, D. Zorin, and E. Burnaev, "Latent video transformer," in *VISIGRAPP*, 2021.

[36] A. Ramesh, P. Dhariwal, A. Nichol, C. Chu, and M. Chen, *Hierarchical text-conditional image generation with clip latents*, 2022. arXiv: 2204.06125 [cs.CV].

[37] A. Ramesh, M. Pavlov, G. Goh, *et al.*, *Zero-shot text-to-image generation*, 2021. DOI: 10.48550/ARXIV.2102.12092. [Online]. Available: https://arxiv.org/abs/2102.12092.

[38] S. E. Reed, Y. Zhang, Y. Zhang, and H. Lee, "Deep visual analogy-making," in *Advances in Neural Information Processing Systems*, C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, Eds., vol. 28, Curran Associates, Inc., 2015. [Online]. Available: https://proceedings.neurips.cc/paper/2015/file/e07413354875be01a996dc560274708e-Paper.pdf.

[39] P. Sangkloy, J. Lu, C. Fang, F. Yu, and J. Hays, *Scribbler: Controlling deep image synthesis with sketch and color*, 2016. arXiv: 1612.00835 [cs.CV].

[40] A. Saravanan and M. Guzdial, *Pixel vq-vaes for improved pixel art representation*, 2022. DOI: 10.48550/ARXIV.2203.12130. [Online]. Available: https://arxiv.org/abs/2203.12130.

[41] J. Schmidhuber, "Deep learning in neural networks: An overview," *CoRR*, vol. abs/1404.7828, 2014. arXiv: 1404.7828. [Online]. Available: http://arxiv.org/abs/1404.7828.

[42] A. Siarohin, S. Lathuilière, S. Tulyakov, E. Ricci, and N. Sebe, "Animating arbitrary objects via deep motion transfer," *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2372–2381, 2019.

[43] A. Siarohin, S. Lathuilière, S. Tulyakov, E. Ricci, and N. Sebe, "First order motion model for image animation," *CoRR*, vol. abs/2003.00196, 2020. arXiv: 2003.00196. [Online]. Available: https://arxiv.org/abs/2003.00196.

[44] G. Somepalli, V. Singla, M. Goldblum, J. Geiping, and T. Goldstein, *Diffusion art or digital forgery? investigating data replication in diffusion models*, 2022. arXiv: 2212.03860 [cs.LG].

[45] A. M. Strowel, "Chatgpt and generative ai tools: Theft of intellectual labor?" *IIC - International Review of Intellectual Property and Competition Law*, vol. 54, pp. 491–494, 2023. [Online]. Available: https://api.semanticscholar.org/CorpusID:257966567.

[46] F. Thomas and O. Johnston, *The Illusion of Life: Disney Animation*. Disney Editions, 1981. [Online]. Available: https://books.google.ca/books?id=k5TMoAEACAAJ.

[47] A. Vaswani, N. Shazeer, N. Parmar, *et al.*, *Attention is all you need*, 2017. arXiv: 1706.03762 [cs.CL].

[48] J. Walker, A. Razavi, and A. van den Oord, "Predicting video with vqvae," *ArXiv*, vol. abs/2103.01950, 2021.

[49] T.-C. Wang, M.-Y. Liu, A. Tao, G. Liu, J. Kautz, and B. Catanzaro, "Few-shot video-to-video synthesis," *ArXiv*, vol. abs/1910.12713, 2019.

[50] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, *et al.*, "Video-to-video synthesis," in *NeurIPS*, 2018.

[51] Z. Wang, S. Qiu, N. Feng, H. Rushmeier, L. McMillan, and J. Dorsey, "Tracing versus freehand for evaluating computer-generated drawings," *ACM Trans. Graph.*, vol. 40, no. 4, Jul. 2021, ISSN: 0730-0301. DOI: 10.1145/3450626.3459819. [Online]. Available: https://doi.org/10.1145/3450626.3459819.

[52] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, 2004. DOI: 10.1109/TIP.2003.819861.

[53] L. Yang, Z. Zhang, Y. Song, *et al.*, *Diffusion models: A comprehensive survey of methods and applications*, 2023. arXiv: 2209.00796 [cs.LG].

[54] J. S. Yoon, L. Liu, V. Golyanik, K. Sarkar, H. S. Park, and C. Theobalt, *Pose-guided human animation from a single image in the wild*, 2020. DOI: 10.48550/ARXIV.2012.03796. [Online]. Available: https://arxiv.org/abs/2012.03796.

[55] Q. Yu, F. Liu, Y.-Z. Song, T. Xiang, T. M. Hospedales, and C. C. Loy, "Sketch me that shoe," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 799–807. DOI: 10.1109/CVPR.2016.93.

[56] L. Zhang and M. Agrawala, *Adding conditional control to text-to-image diffusion models*, 2023. arXiv: 2302.05543 [cs.CV].

[57] M. R. Zhang, J. Lucas, G. E. Hinton, and J. Ba, "Lookahead optimizer: K steps forward, 1 step back," in *NeurIPS*, 2019.

[58] Q. Zhang, C. Xiao, H. Sun, and F. Tang, "Palette-based image recoloring using color decomposition optimization," *IEEE Transactions on Image Processing*, vol. PP, pp. 1–1, Feb. 2017. DOI: 10.1109/TIP.2017.2671779.

[59] V. Zhou, V. Z. is a reporter at Rest of World., M. Mutaher, V. Bansal, and D. Dib, *Ai is already taking video game illustrators' jobs in china*, Apr. 2023. [Online]. Available: https://restofworld.org/2023/ai-image-china-video-game-layoffs/.