AI Capstone Final Project Report - Team 16
Aerial Shot Image Generation with Conditional Control

Members: 109350008 張詠哲
110550089 周冠辰
110705013 沈昱宏

1. Introduction

Motivation

With the rapidly growing field of generative AI, especially in the domain of image generation, we are curious about the technology behind it and how these methods are applied to real-world tasks. Generating aerial images, a task that produces images similar to those captured by UAV cameras, is one such application.

Training UAV autopilot models typically requires a substantial amount of paired data consisting of aerial images and corresponding flight paths, and collecting this data can be resource-intensive and time-consuming. However, if we can generate realistic images under specific conditions, these images could be used to train UAV autopilot models to learn flight paths, significantly reducing the cost of building a training dataset for autopilots.

Dataset

Our datasets are provided by AI CUP 2024 [1], a competition organized by the government. We have 2,160 pairs of training data; each pair consists of an aerial shot image and a UAV path image. The path image features a black background with white lines outlining the road boundaries and the flight path that the UAV should follow, as shown in Figure 1. Additionally, we have a test dataset containing 360 UAV path images. We will use these images as conditions to generate realistic aerial shot images, expecting that the generated images will have a distribution similar to that of the training dataset.

Figure 1. Aerial shot and the corresponding UAV path image.

2. Methods

In our experiments, we implemented and compared the previously popular conditional GAN methods with the currently trending diffusion-based conditional image generation techniques. In summary, we tried 3 architectures to complete our task, Pix2Pix, stable diffusion with LoRA, and stable diffusion with ControlNet.

Pix2Pix

This is a GAN based method with UNet as generator and PathGAN classifier as discriminator. We leverage the implementation in [5] and train the model on our own dataset.

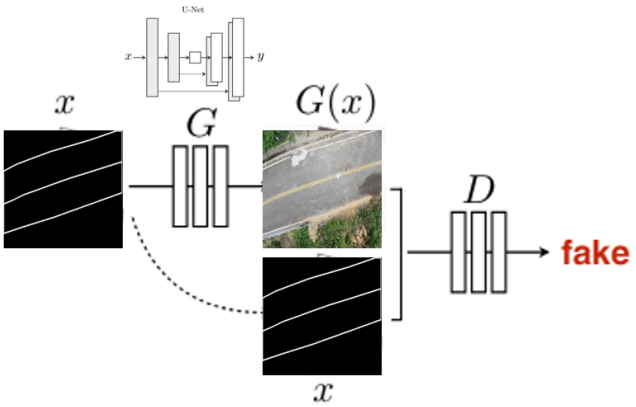


Figure 2. Pix2Pix training architecture

Stable Diffusion with LoRA

Stable Diffusion [2] is currently the most popular diffusion model. It shares the same architecture as the LDM (latent diffusion model) [3] but is pretrained on a very large dataset. Unlike the original DDPM, the latent diffusion model performs the diffusion process in latent space and can incorporate conditions in the denoising UNet. The original Stable Diffusion was pre-trained using a CLIP text encoder, which enables image generation guided by corresponding text prompts. As shown in Figure 4, we plan to replace the text encoder with a CLIP image encoder. We expect that the diffusion UNet will then be able to use UAV path image features to generate corresponding images.

Fine-tuning a latent diffusion model directly requires a significant amount of VRAM. Therefore, we plan to use LoRA [4] for fine-tuning the model. LoRA is a parameter-efficient fine-tuning method originally designed for large language models. The key concept of LoRA is to add an additional trainable low-rank matrix to each layer of the original model. During the training process, only the weights of these matrices are tuned, while the original model's weights are frozen.

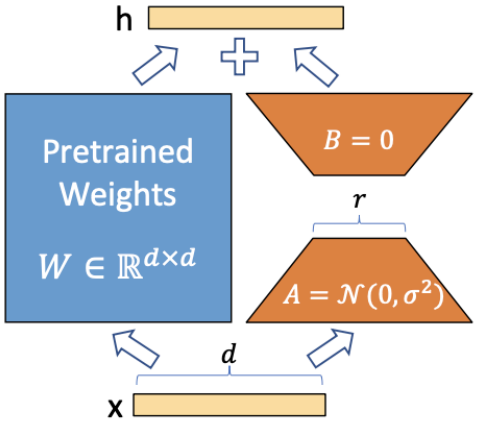


Figure 3. Concept of LoRA.

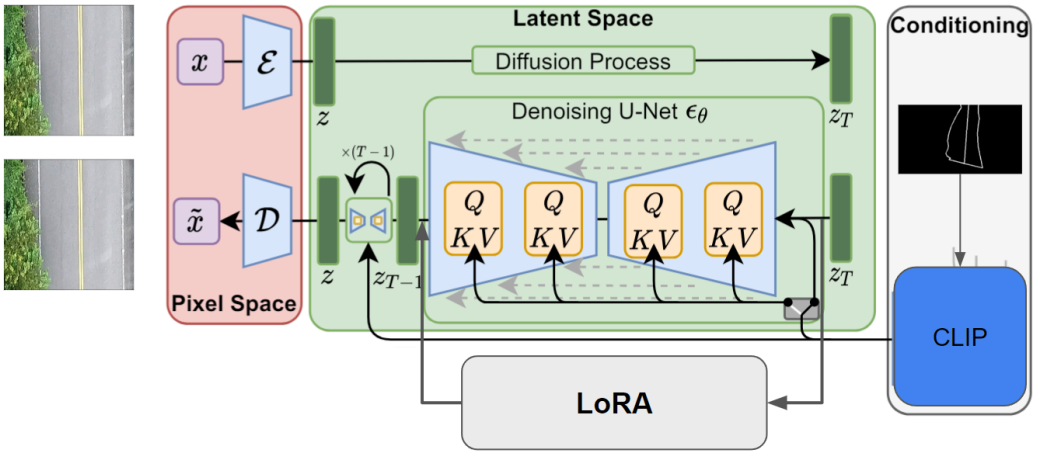


Figure 4. Our modified pipeline of Stable Diffusion with LoRA.

However, we found that our designed pipeline cannot really converge and generate reasonable results, as shown in Figure 5. We believe there are two reasons for the poor performance. Firstly, the CLIP image encoder was trained with pairs of text, which means that unlike other general image encoders, it may not easily capture the spatial features in an image. Secondly, LoRA, with too few parameters, may not effectively learn to transition from using text embedding guidance to image embedding guidance.

Figure 5. Failure case of training Stable Diffusion with LoRA.

Stable Diffusion with ControlNet

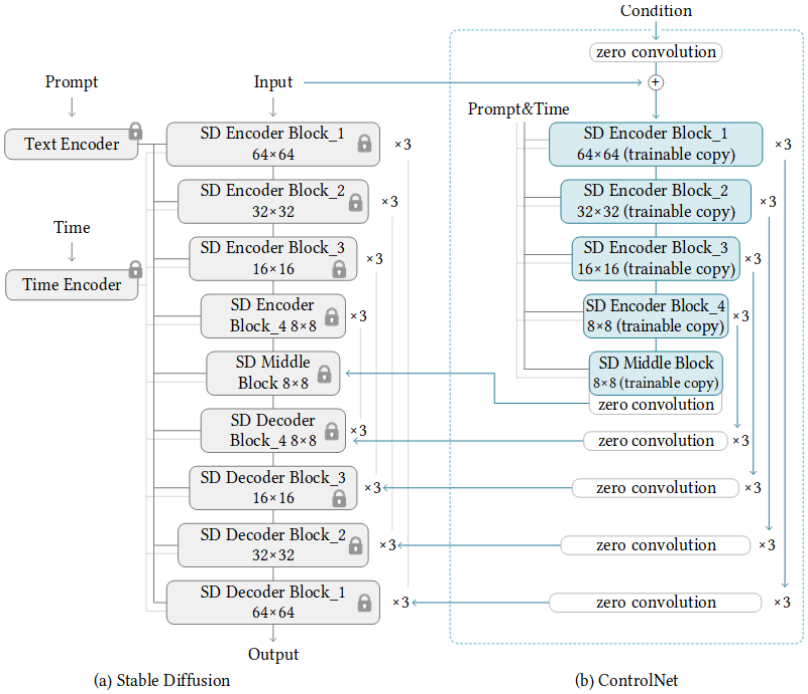


Figure 6. Architecture of Stable Diffusion with ControlNet.

To address the issues encountered with the previous method, we experimented with another popular technique for guiding the diffusion process with image conditions: ControlNet [6]. The concept of ControlNet is quite similar to LoRA, as it involves adding additional modules to learn how the conditions guide the model in generating images. Specifically, ControlNet makes a trainable copy of the encoder part of the UNet, adds a convolutional layer as the first layer, and integrates additional convolutional layers in the last few stages that connect to the UNet's decoder.

In the original paper, ControlNet was trained with all of UNet's weights locked. However, the authors also suggested that when training on a dataset with a distribution different from that of the original pre-training dataset, unlocking the decoder part of the UNet may be beneficial. The decoder in the UNet is tasked with predicting the next step of noise in the diffusion denoising process, using the features extracted by the encoder. Fine-tuning the decoder, therefore, can help ensure that the generated images more closely align with the fine-tuning dataset. In our experiments, we have compared these two methods.

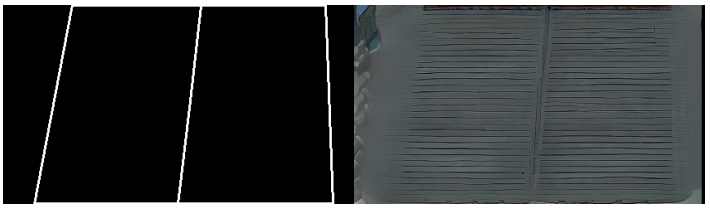


Figure 7. Failure case of training with UAV path images.

Initially, we used the original UAV path image, which consisted only of a black background with a white line. However, we found that the model struggled to learn effectively and failed to generate reasonable images, as depicted in Figure 7. We believe the main issue is that the original conditional image contains too little information for the model to accurately determine road placement. To overcome this, we converted the UAV path images into segmentation maps by coloring the areas where roads should and should not be. We then used these segmentation maps as conditions to train the ControlNet, which resulted in successful outcomes.

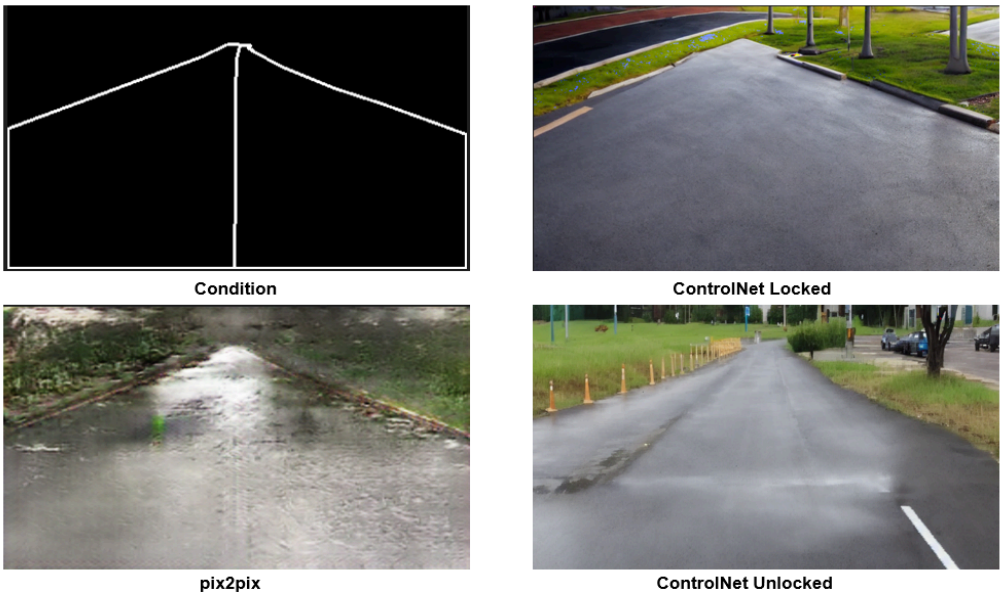
Figure 8. Transition from UAV path image to segmentation map image.

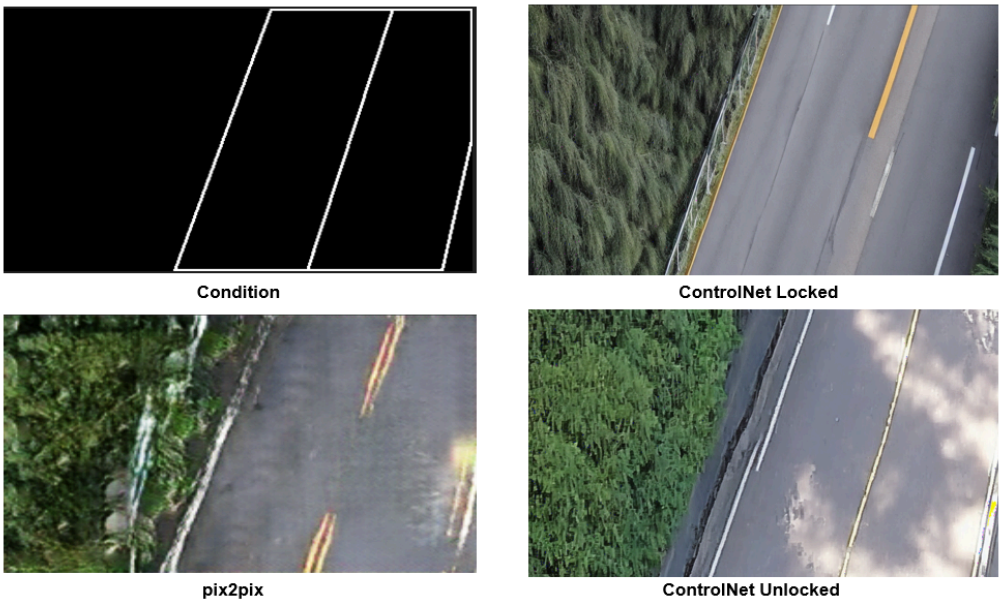
3. Experimental Results & Analysis

We compare the three successful cases: Pix2Pix, Stable Diffusion with ControlNet and decoder locked, Stable Diffusion with ControlNet and decoder unlocked.

Generated images

You can observe that the three methods all learn the condition well, while stable diffusion models will have higher image quality. When using ControlNet with a stable diffusion decoder locked, the image is not as realistic as the one with the decoder unlocked and generates an image with an animation style, meaning that unlocking the decoder does help transferring the style.





Evaluation Metrics

We use FID (Fréchet Inception Distance) as our evaluation metrics. FID measures the similarity between two sets of images by comparing the distribution of features extracted from each set using a deep learning model.

	FID-192	FID-64
pix2pix	9.717	3.915
ControlNet-lock	22.929	6.169
ControlNet-unlock	16.004	6.733

A lower FID score means that your generated image has a more similar distribution to the training data. In the table, you can see that pix2pix has a better similarity to the training data. This is quite surprising because the image generated by Controlnet looks way better than the one by pix2pix in our perspective. We took a look at the training data, and came up with a possible reason. In the training data, there are only roads and trees in the image. This makes pix2pix, which we train from scratch, only able to generate images with roads and trees. However, the stable diffusion model is pre-trained on a large amount of images, so it can generate cars, fences, traffic cones, etc. This makes the feature of generated images quite different from the original dataset, making the FID score to be higher. We consider this diversity as an improvement brought by the pretrained stable diffusion model, which makes the generated images more diverse while being reasonable.

4. Conclusion

In our experiment, we found that both pix2pix and ControlNet are capable of generating reasonable results in image generation tasks. Pix2pix requires less in terms of computational resources and can be easily trained with a small dataset, making it accessible for projects with limited resources. However, it falls short in generating high-quality images. On the other hand, training a ControlNet requires significantly more VRAM; however, it compensates for this by producing more realistic images. Moreover, by leveraging a well-trained foundation model, ControlNet can generate details that are not present in the fine-tuning dataset, enhancing the richness and authenticity of the generated images.

5. Contribution

張詠哲 - pix2pix
周冠辰 - stable diffusion + ControlNet
沈昱宏 - stable diffusion + LoRA

6. Reference

[1] https://www.aicup.tw/ai-cup-2024-competition

[2] https://github.com/CompVis/stable-diffusion

[3] https://arxiv.org/abs/2112.10752

[4] https://github.com/microsoft/LoRA

[5] junyanz/pytorch-CycleGAN-and-pix2pix

[6] https://github.com/lllyasviel/ControlNet

[7] runwayml/stable-diffusion-v1-5 · Hugging Face