AI-Based Sketch-to-Anime Transformation Using ControlNet and Stable Diffusion XL.

1. Introduction

In recent years, AI- aided image generation has brought a revolution in the field of digital art. For example the Ghibli art trend that went viral where people used various AI models to generate their animated images mainly using OpenAI's GPT-4o. Our project, **Magic Pencil – Anime**, works on a similar genre in anime-style image creation by converting rough sketches into high-quality anime artworks.
Our application uses **ControlNet**, a group of neural networks refined using Stable Diffusion, which empowers precise artistic and structural control in generating images. It improves default Stable Diffusion models by incorporating task-specific conditions combined with **Stable Diffusion XL (SDXL)** for high-fidelity image synthesis.

2. Objectives

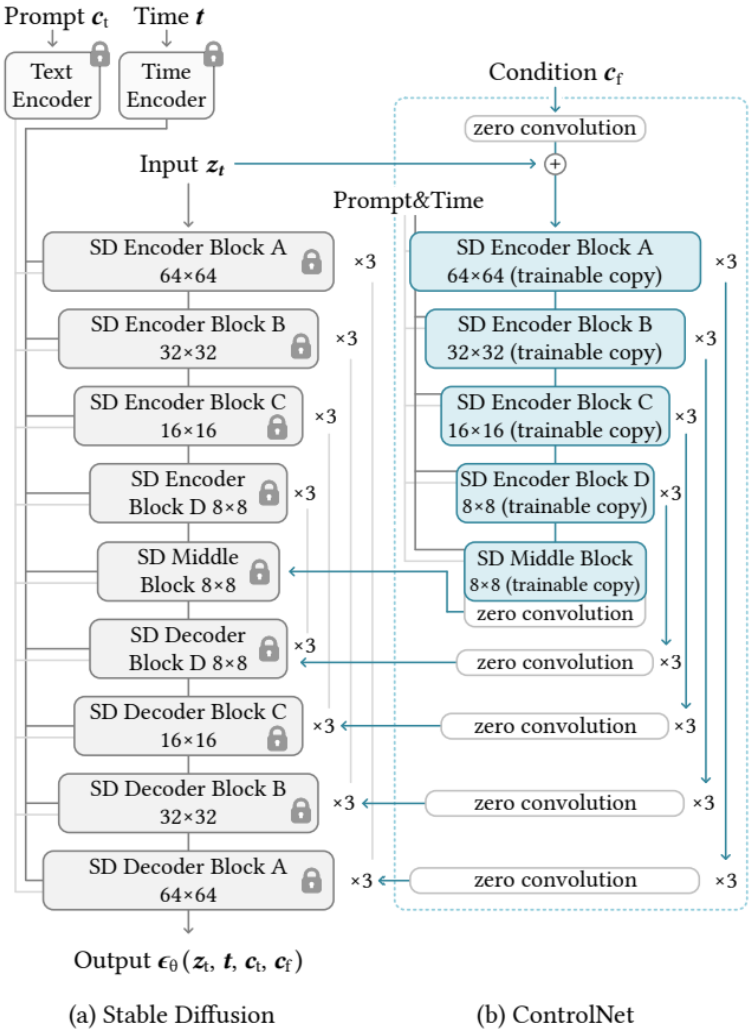
The main objectives of our project is:

- Develop a user-friendly web interface that transforms sketches into anime-style art.
- Use **ControlNet** and **Stable Diffusion XL** for high-quality and controllable image generation.
- Allow users to customize generation parameters such as prompt strength, inference steps, and sketch control strength.
- Demonstrate the practical use of **diffusion models** in creative digital applications.

Our Webapp converts hand drawn rough images and digital sketches into anime style pictures, with colors customized as per the user requirements.It leverages a ControlNet model tuned to anime line art **(xinsir/anime-painter)** as an input to a Stable Diffusion XL pipeline **(gsdf/CounterfeitXL)** using the **Hugging Face Diffusers framework**.

3. Literature Review

Diffusion models are a kind of generative AI that creates completely new data like images, audio or video by starting with random noise and gradually turning it into something meaningful. They work by simulating a **diffusion process** in which the data is slowly corrupted by noise during training and then learning to reverse this process step by step. By doing this the model learns how to generate high quality samples of images or videos from scratch. **Stable Diffusion XL (SDXL)** is a large-scale diffusion model capable of generating detailed, photorealistic or stylized images from text prompts.



ControlNet, proposed by Lvmin Zhang and Maneesh Agrawala (2023), extends diffusion models by conditional generation, meaning guiding the model on what to generate rather than letting it generate an output randomly, constraints/inputs such as edges, sketches or poses the user desires.It reuses the large-scale pretrained layers of source models to build a deep and strong encoder to learn specific conditions. The original model and trainable copy are connected via “zero convolution” layers that eliminate harmful noise during training.

4. System Architecture

4.1 Architecture Overview

Our project's architecture contains 3 main components:

1. **Frontend (Streamlit):**
    - Handles UI.
    - Allows image upload, input prompt, and adjusting parameters for image generation.
    - Displays the generated image and provides downloading options.
2. **Model Backend (Diffusers Framework):**
    - Integrates ControlNet and Stable Diffusion XL.
    - Performs inference using GPU acceleration when available.
3. **Processing Layer (image preprocessing):**
    - Converts uploaded sketch into a black and white line art by inverting colors.

4.2 Workflow

1. The user uploads a **rough sketch of anime character**.
2. The image is **preprocessed** (converted to grayscale, thresholded, blurred, and inverted).
3. A **ControlNet model (`xinsir/anime-painter`)** puts constraints on the sketch specified by the user.
4. The **Stable Diffusion XL (`gsdf/CounterfeitXL`)** model generates the final image.
5. A downloadable image output is displayed .

5. Technologies Used

Component	Technology/Library	Description
Frontend	Streamlit	Web-based Python framework for rapid UI development
Backend	Diffusers (Hugging Face)	Provides Stable Diffusion and ControlNet integration
Model 1	ControlNet (xinsir/anime-painter)	Guides image generation using sketch input
Model 2	Stable Diffusion XL (gsdf/CounterfeitXL)	Core image generation engine
Processing	OpenCV, NumPy, Pillow	Used for sketch preprocessing and image manipulation
Hardware	CUDA GPU (optional)	Accelerates inference using PyTorch

6. Implementation

6.1 Sketch Preprocessing

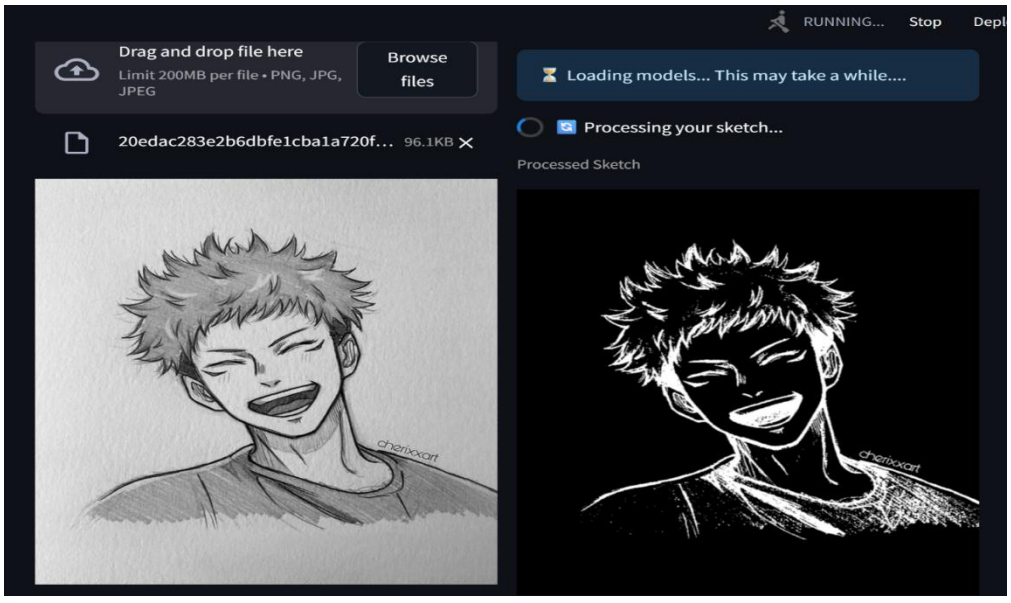
We are converting the sketch uploaded by the user to a clean line map *(grayscale → threshold → blur → optional inversion)* then sent to the ControlNet model.

preprocess.py

```python
def preprocess_sketch(image_pil):
    # Convert to grayscale
    image_gray = image_pil.convert('L')

    # Apply threshold to create binary image
    image_array = np.array(image_gray)
    _, binary = cv2.threshold(image_array, 127, 255, cv2.THRESH_BINARY)

    # Apply slight blur to smooth lines
    binary = cv2.GaussianBlur(binary, (3, 3), 0)

    # Invert
    if np.mean(binary) > 127:
        binary = 255 - binary

    return Image.fromarray(binary)
```

The function *preprocess_sketch* ensures the sketch has clear light strokes on a dark background or inverted to that form. Gaussian blur function from OpenCV library smooths rough/sharp strokes so ControlNet sees clean smoothed out edges as input.

Output:



6.2 Model Loading and Optimization

Loading ControlNet, VAE, Scheduler, and the Stable Diffusion XL + ControlNet pipeline, we use GPU optimizations if GPU is available on the system and cache the pipeline for future executions.

Load_models.py

```python
# gpu
if torch.cuda.is_available():
    device = "cuda"
    dtype = torch.float16
else:
    device = "cpu"
    dtype = torch.float32
# print(device)


# Cache models
@st.cache_resource
def load_models():
    try:
        st.info("⏳ Loading models... This may take a while....")

        # ControlNet
        controlnet = ControlNetModel.from_pretrained(
            "xinsir/anime-painter", torch_dtype=dtype
        )

        # Load VAE
        vae = AutoencoderKL.from_pretrained(
            "gsdf/CounterfeitXL", subfolder="vae", torch_dtype=dtype
        )

        # scheduler
        scheduler = EulerAncestralDiscreteScheduler.from_pretrained(
            "gsdf/CounterfeitXL", subfolder="scheduler"
        )

        # Load pipeline
        pipe = StableDiffusionXLControlNetPipeline.from_pretrained(
            "gsdf/CounterfeitXL",
            controlnet=controlnet,
            vae=vae,
            scheduler=scheduler,
            safety_checker=None,
            torch_dtype=dtype,
        )

        # Optimize pipeline
        if device == "cuda":
            pipe.enable_xformers_memory_efficient_attention()
            pipe.enable_model_cpu_offload()

        return pipe

    except Exception as e:
        st.error(f"Error loading models: {str(e)}")
        return None
```

`@st.cache_resource` caches the model objects so they are not required to be reloaded on every interaction. When CUDA is available, it uses `float16` and enables memory optimizations.

6.3 Generation (ControlNet-conditioned Inference)

Resize input to multiples of 8 and run the pipeline with ControlNet conditioning then return the generated image.

Load_models.py

```python
def generate_anime(pipe, sketch_image, prompt, negative_prompt, num_steps,
guidance_scale, controlnet_scale):
    try:
        width, height = sketch_image.size
        #dimensions multiples of 8
        width =(width // 8) * 8
        height= (height //8)* 8
        sketch_image = sketch_image.resize((width, height))

        # Generate with ControlNet
        with torch.no_grad():
            output = pipe(
                prompt=prompt,
                image=sketch_image,
                negative_prompt=negative_prompt,
                num_inference_steps=num_steps,
                guidance_scale=guidance_scale,
                controlnet_conditioning_scale=controlnet_scale,
                height=height,
                width=width
            )

        return output.images[0]

    except Exception as e:
        st.error(f"Error during generation: {str(e)}")
        return None
```

This function handles the image size constraints and calls the [StableDiffusionXLControlNetPipeline] with the user specified parameters and the processed sketch that was uploaded.

6.4 UI: Upload, Controls, and Downloads

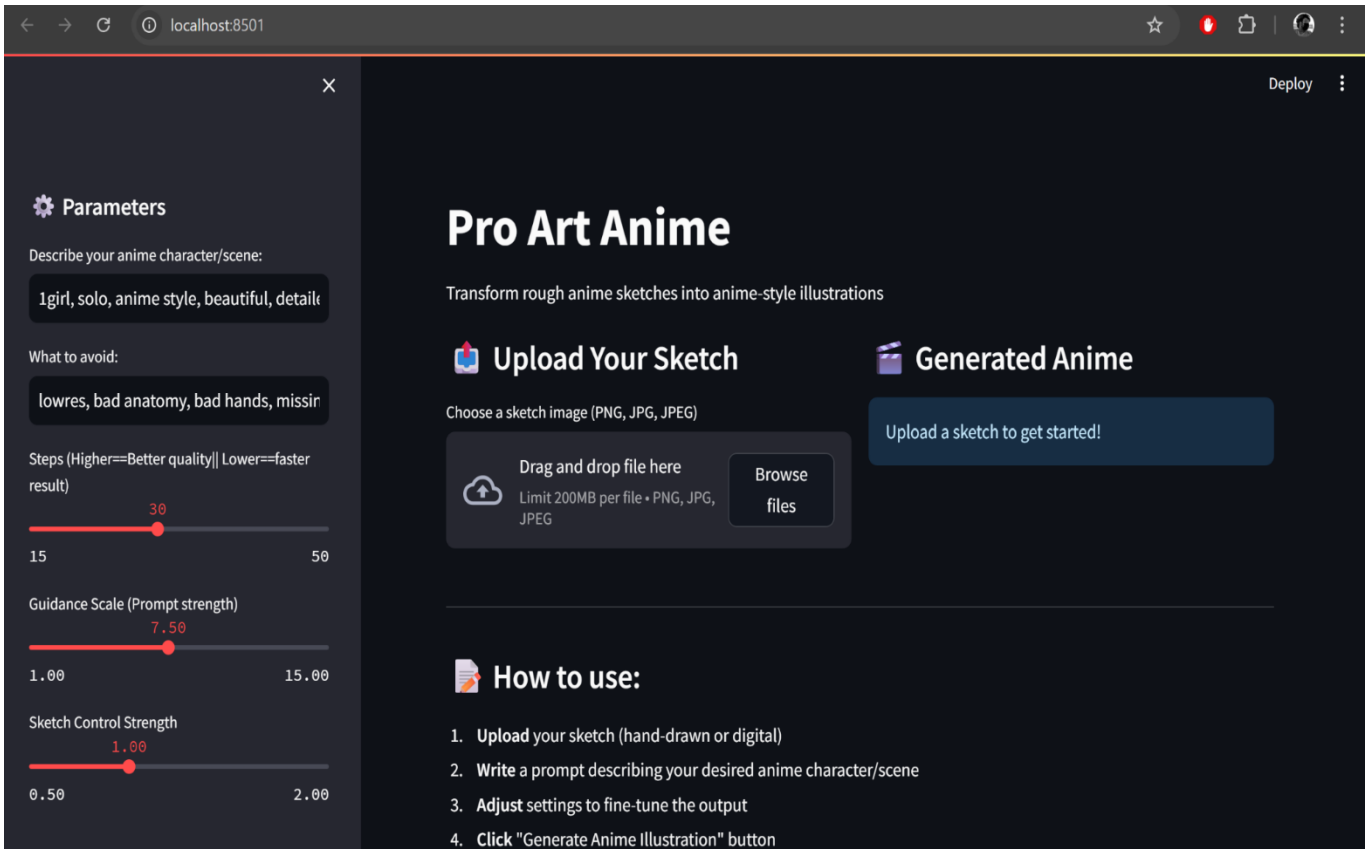
Controls the user interface supports uploading, parameter controls in the sidebar, handles the generation flow and image download.

app.py

```python
st.set_page_config(
    page_title="Magic Pencil --Anime",
    page_icon="🎨",
    layout="wide",
    initial_sidebar_state="expanded",
)

st.title("Magic Pencil --Anime")
st.markdown("Transform rough anime sketches into anime-style illustrations")

# Sidebar controls
st.sidebar.header("⚙️ Parameters")
prompt = st.sidebar.text_input(
    "Describe your anime character/scene:",
    value="1girl, solo, anime style, beautiful, detailed, high quality,
masterpiece",
)

negative_prompt = st.sidebar.text_input(
    "What to avoid:",
    value="lowres, bad anatomy, bad hands, missing fingers, worst quality, low
quality",
)

num_steps = st.sidebar.slider(
    "Steps (Higher==Better quality|| Lower==faster result)", 15, 50, 30
)
guidance_scale = st.sidebar.slider("Guidance Scale (Prompt strength)", 1.0, 15.0,
7.5)
controlnet_scale = st.sidebar.slider("Sketch Control Strength", 0.5, 2.0, 1.0)

# Main UI
col1, col2 = st.columns(2)

with col1:
    st.subheader("📤 Upload Your Sketch")
    uploaded_file = st.file_uploader(
        "Choose a sketch image (PNG, JPG, JPEG)", type=["png", "jpg", "jpeg"]
    )

    if uploaded_file is not None:
        sketch = Image.open(uploaded_file)
        st.image(sketch, caption="Original Sketch", use_column_width=True)

with col2:
```

```python
with col2:
    st.subheader("🎬 Generated Anime")
    if uploaded_file is not None:
        if st.button("✨ Generate Anime Illustration", key="generate"):
            # Load models
            pipe = load_models()

            if pipe is not None:
                with st.spinner("🔄 Processing your sketch..."):
                    # Preprocess sketch
                    sketch_pil = Image.open(uploaded_file)
                    processed_sketch = preprocess_sketch(sketch_pil)

                    # Show processed sketch
                    st.caption("Processed Sketch")
                    st.image(processed_sketch, use_column_width=True)

                    # Generate anime
                    result = generate_anime(
                        pipe,
                        processed_sketch,
                        prompt,
                        negative_prompt,
                        num_steps,
                        guidance_scale,
                        controlnet_scale,
                    )

                    if result is not None:
                        st.success("✅ Generation complete!")
                        st.image(
                            result, caption="Generated Anime", use_column_width=True

                        )

                        # Download button
                        buf = io.BytesIO()
                        result.save(buf, format="PNG")
                        buf.seek(0)
                        st.download_button(
                            label="⬇️ Download Image",
                            data=buf,
                            file_name="anime_output.png",
                            mime="image/png",
                        )
    else:
        st.info("Upload a sketch to get started!")
```

Output:


localhost:8501
Deploy
Parameters
Describe your anime character/scene:
1girl, solo, anime style, beautiful, detail
What to avoid:
lowres, bad anatomy, bad hands, missir
Steps (Higher==Better quality|| Lower==faster result)
30
15
50
Guidance Scale (Prompt strength)
7.50
1.00
15.00
Sketch Control Strength
1.00
0.50
2.00
Pro Art Anime
Transform rough anime sketches into anime-style illustrations
Upload Your Sketch
Choose a sketch image (PNG, JPG, JPEG)
Drag and drop file here
Limit 200MB per file • PNG, JPG, JPEG
Browse files
Generated Anime
Upload a sketch to get started!
How to use:
1. Upload your sketch (hand-drawn or digital)
2. Write a prompt describing your desired anime character/scene
3. Adjust settings to fine-tune the output
4. Click "Generate Anime Illustration" button

6.5 Generation Parameters

Parameter	Description	Default
Prompt	Text describing the anime scene	“1girl, anime style, masterpiece”
Negative Prompt	Avoids unwanted features	“lowres, bad anatomy, low quality”
Steps	Number of denoising iterations	30
Guidance Scale	Strength of prompt influence	7.5
ControlNet Scale	Strength of sketch influence	1.0

7. User Interface (Streamlit)

The interface has two main panels:

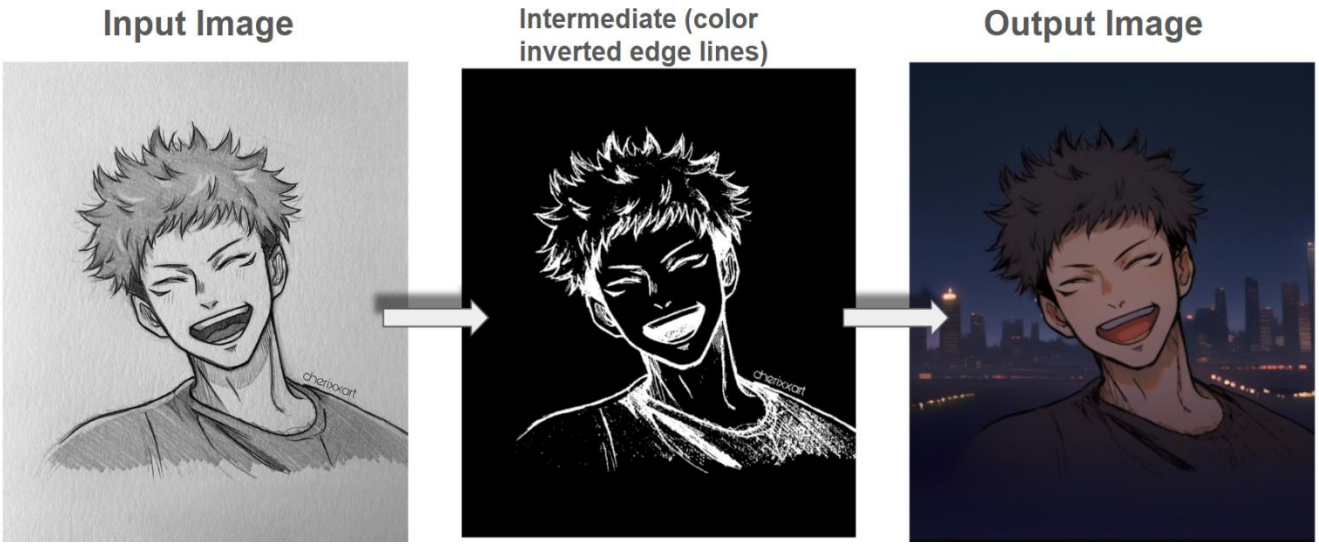
Section	Function
Left Panel	Upload sketch, prompts, adjust parameters
Right Panel	Processed sketch and generated anime image

After clicking **"Generate Anime Illustration"** the app will process the sketch and display both intermediate and final results which are downloadable.

8. Results

8.1 Sample Outputs

- **Input:** Rough black and white sketch
- **Output:** Colored anime style image consistent with the sketch structure



8.2 Observations

- Higher inference steps improve quality but also increase generation time.
- Increasing the guidance scale strengthens the bias to text prompts.
- Adjusting the ControlNet scale allows balancing between the original sketch and model creativity.

9. Conclusion

The **Pro Art Anime** project successfully demonstrates how generative AI models can assist art enthusiasts in enhancing sketches into professional artworks.
Using **ControlNet** with **Stable Diffusion XL** our project achieves controlled, higher quality anime style image generation.
The Web app provides an intuitive and accessible way for anyone to experiment with creative AI and produce professional grade anime images.

10. References

1. Lvmin Zhang & Maneesh Agrawala. *ControlNet: Adding Conditional Control to Text-to-Image Diffusion Models*. 2023.
2. Hugging Face Diffusers Library: https://huggingface.co/docs/diffusers
3. Stable Diffusion XL Model: https://huggingface.co/gsdf/CounterfeitXL
4. ControlNet Model (Anime Painter): https://huggingface.co/xinsir/anime-painter

Contributors:

NAME	%AGE OF CONTRIBUTION
AYANAVA DUTTA (2511AI61)	10.0
MANSI VASHISTHA (2511AI60)	10.0
SASIKUMAR R (2511CS14)	10.0
ARVIND KUMAR (2511CS11)	10.0
UTKARSH PANDEY (2511AI63)	10.0
SAMYAKA BHAVSAGAR (2511AI46)	10.0
ABDUL KHAZIMUDDIN (2511CS06)	10.0
ABHYUDAYA YADAV (2511AI04)	10.0
KONAPALA TARAKANANADA (2511AI41)	10.0
TARUN VIJAY (2511CS20)	10.0