Video Conversion from SD to HD Using Stable Diffusion Model

Introduction

The objective of this project was to develop a prototype for converting SD resolution videos (640 x 480px) to HD resolution (1280 x 720px) using a diffusion type model. The conversion process aimed to preserve the context of the video while efficiently filling in the blank areas on the left and right sides of the video with relevant pixels or image parts.

Design and Implementation

For the above objective we proposed the design mentioned below in fig 1, where we are first extracting the low-resolution frames from the SD video and then using a stable diffusion upscaling model by hugging face to upscale the given frames, and finally we are recomposing the generated upscaled frames to get our required HD video.

Fig 1: Design of Video Conversion

1. **Frame Extraction:** This is the first step in our video conversion process where we extracted individual frames from the input SD video. It was accomplished with the help of "*cv2.VideoCapture*" function from the OpenCV library. Each frame then was saved to the designated library for further processing. Mentioned below is the code for frame extraction:

```
import os
import cv2
SDvideopath = 'SDvideo/641.mp4'
SDvideoframepath = 'SDvideo/frames'
os.makedirs(SDvideoframepath, exist_ok=True)

def extract_frames(video_path, frame_path):
    video = cv2.VideoCapture(video_path)
    frame_count = 0
    while True:
        ret, frame = video.read()
        if not ret:
```

```
            break
        frame_name = os.path.join(frame_path, f'frame_{frame_count:04d}.png')
        cv2.imwrite(frame_name, frame)
        frame_count += 1
    video.release()
    print("Frame extraction completed.")
extract_frames(SDvideopath, SDvideoframepath)
```

2. **Model loading and setup:** After frame extraction the next step was to load our pretrained Stable diffusion upscale model "*StableDiffusionUpscalePipeline*" by diffusers. The *StableDiffusionUpscalePipeline* can be used to enhance the resolution of input images by a factor of 4 . This is present in *Hugging Face* library. We have used it for upscaling the frames extracted from our SD video and then converted that upscaled frames to video.
   This model uses a prompt to perform the task. Mentioned below is the code for model loading and setup:

```
import torch
from diffusers import StableDiffusionUpscalePipeline
device = 'cuda' if torch.cuda.is_available() else 'cpu'
print(f"Using device: {device}")
model_id = "stabilityai/stable-diffusion-x4-upscaler"
pipe = StableDiffusionUpscalePipeline.from_pretrained(model_id, torch_dtype=torch.float16)
pipe = pipe.to(device)
```

3. **Frame Upscaling:** Now, each extracted frames from SDvideo is processed through the Stable Diffusion Upscale model. We designed our prompt in such a way so the context of video can be preserved as well as we can upscale our video frames. These upscaled frames were saved in a separate frame directory in HD video folder. Mentioned below is the code for frame upscaling:

```
from PIL import Image

prompt = "Upscale video frames"
HDvideoframepath = 'HDvideo/frames'
os.makedirs(HDvideoframepath, exist_ok=True)

def upscale_frames(input_path, output_path):
    for image_name in os.listdir(input_path):
        if image_name.endswith('.png'):
            file_path = os.path.join(input_path, image_name)
            image = Image.open(file_path).convert("RGB")
            with torch.no_grad():
                result = pipe(prompt=prompt, image=image)
                upscaled_image = result.images[0]

            HD_frame=
            os.path.join(output_path,f"upscaled_{image_name}")
            upscaled_image.save(HD_frame)
    print("Frame up-scaling completed.")
upscale_frames(SDvideoframepath, HDvideoframepath)
```

4. **Video Recomposing:** The final step of our conversion video was to recompose the upscaled frames to a mp4 video. This was accomplished by “cv2.videoWriter” function from OpenCV. We have use ‘.mp4v’ codec to convert our frames to a mp4 format video. Mentioned below is the code for video recomposing:

```
def create_video_from_frames(frame_path, output_video, frame_rate=60):
  frames = [frame for frame in sorted(os.listdir(frame_path)) if frame.endswith('.png')]
  first_frame = cv2.imread(os.path.join(frame_path, frames[0]))
  height, width, channels = first_frame.shape
  frame_size = (width, height)

  fourcc = cv2.VideoWriter_fourcc(*'mp4v')
   video_writer = cv2.VideoWriter(output_video, fourcc, frame_rate, frame_size)

  for frame in frames:
  video_frame = cv2.imread(os.path.join(frame_path, frame))
  video_writer.write(video_frame)

  video_writer.release()
  print("Video up-sampling completed.")
HDvideo = 'HDvideo/1080.mp4'
create_video_from_frames(HDvideoframepath, HDvideo)
```

Evaluation and Result

Here mentioned below fig2 are the results of video conversion of a SD video frame to an HD video Frame.

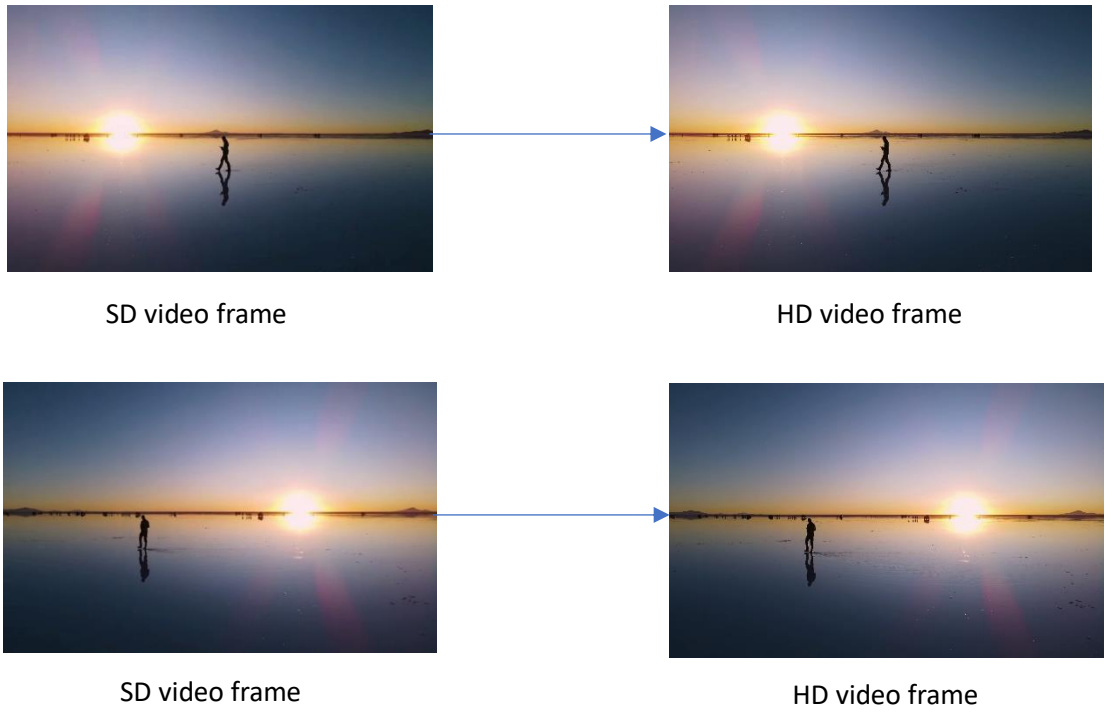

Fig 2: Result of Video Conversion

Conclusion

Our prototype can successfully change SD resolution videos to HD resolution by using a diffusion type model. Here, frames are efficiently extracted, pre-trained model is used to upscale them and later recompose them as HD video. The end result of this project would not only maintain high quality but also retain the meaning of original video. This approach just shows how diffusion models may be utilized for upscaling of video and inpainting purposes.

References

- Stable Diffusion Upscale Pipeline by Hugging Face
- OpenCV