26.02 Naver Boostcamp Ai tech

RE:COVER

<Live2d Album Art>
CV Final project

CV04 [Team SeeValue]
고대영 T8010
고재완 T8011
양지훈 T8119
최호준 T8211

목차

I. 프로젝트 개요

앨범 아트,

그리고

감성 한 스푼.

앨범 아트는 음반을 대표하는 시각 요소로서, 청취자가 음악의 인상을 형성하고 기억하는 데 중요한 매개체로 기능한다. 최근 Apple Music의 Motion Art, Spotify의 Canvas 등 움직이는 앨범 아트가 주요 스트리밍 플랫폼을 중심으로 확산되고 있으나, 제작 비용 및 운영 부담으로 인해 적용 범위가 제한적이며 대다수의 롱테일(Long-tail) 음원에는 도입되지 못하는 한계가 존재한다.

본 프로젝트는 이러한 격차를 해소하기 위해, 뮤직비디오와 정적인 앨범 아트 사이의 중간 영역을 지향한다. 즉, 어떤 음악이든 정적인 앨범 아트를 기반으로 Live2D 친화적(왜곡 최소/요소 고정성 확보) 루프 비디오로 재구성할 수 있도록 자동화된 생성 모델 및 서비스 구조를 설계·구현하였다.

본 프로젝트의 목표를 정리해 보면 아래와 같다.

- 텍스트 입력 또는 정적 이미지 입력만으로 **약 4초 내외의 고품질 루프 영상(Seamless Loop)**을 생성
- 반복 재생 시 이음새(Seam)가 최소화되도록 **자연스러운 순환 구조**를 확보

적용한 시스템 아키텍처를 정리하면 아래와 같다.

- **AI Core:** Wan 2.2 5B (Video Gen), Qwen (VLM), LaMa (Inpainting), Gemini API
- **Backend:** FastAPI (Async Job Queue), ComfyUI
- **External API:** iTunes Search API, YouTube Data API

II. 프로젝트 팀 구성 및 역할

[고대영] : 텍스트 - 배경 분리 알고리즘 구현, Diffusion모델 fine-tuning 실험

[고재완] : 초반 diffusion model 탐색 , Frontend , Backend

[양지훈] : ComfyUI 기반 workflow 탐색, seamless loop 구현 방식 리서치 및 적용, LoRA 학습

[최호준] : 프로젝트 제안 및 구체화, LoRA학습 모듈 구성, 데이터 캡셔닝

III. 프로젝트 수행 절차 및 방법

3-1. 핵심 아키텍처

a. OVERVIEW

	
원본 앨범(실리카겔-desert eagle)	잘못 생성된 이미지

초기에는 오브젝트마다 맞는 모션이 다르다고 보고, 오브젝트를 분리해 각각 diffusion을 적용하면 더 잘 될 것이라 판단해 InternVL3–Grounded DINO–SAM 기반 분리 파이프라인을 구성했다. 하지만 실제로는 배경/맥락 정보가 품질에 크게 기여했고, 분리보다 모델이 전체 맥락을 보고 판단하게 하는 편이 더 낫다고 결론 내려 이쪽으로 방향성을 새로 잡게 되었다.

또한 Live2D 지향 특성상 원본과의 일치가 필수였는데, 생성 과정에서 원본이 흔들리는 문제가 반복되었다. 스타일을 크게 바꾸는 대안(픽셀아트 등)도 검토했지만, 목표와 달라 제외했다.

한편 긴 영상은 품질 저하가 커서, 2~5초 내외의 짧은 클립을 고품질로 만든 뒤 루프 구성하는 방향으로 전략을 정리했다. 그리고 공개 LoRA는 인물 중심이 많아, 배경 중심 앨범아트 도메인에 그대로 적용하기 어려웠다.

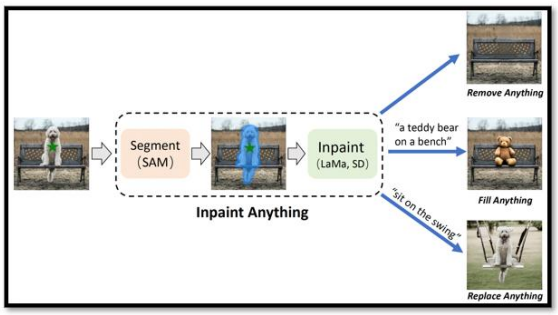
따라서 해결 과제를 다음 3가지 핵심 이슈로 정리했다.

1. Image Preprocessing (텍스트 등은 자체적으로 분류 및 처리)
2. Seamless Loop
3. Live2D-friendly LoRA 학습

diffusion base 모델로는 유저 리뷰가 어느정도 축적되어 있고, 무게가 가벼운 wan2.2 ti2v 5B 모델을 채택했다.

b. Image Preprocessing

Text layer Separation



LaMa: Resolution-robust Large Mask Inpainting with Fourier Convolutions

명도 중심 분석, 픽셀 주변의 local black과 대조하여 글자 탐지

앨범 아트의 텍스트(타이틀, 아티스트명)이 비디오 생성 시 뭉개지는 현상이 있었고, 이를 해결하기 위해 아래와 같은 방식으로 접근해 보았다.

초기에는 앨범 아트를 **배경·오브젝트·텍스트**의 3개 레이어로 완전 분리해, 텍스트는 고정(가독성 유지), 오브젝트는 미세 모션, 배경은 Ambient Motion을 부여하는 방식으로 이상적인 모션 그래픽을 목표로 했다. 그러나 **색상 기반 분리**는 텍스트와 배경 효과가 유사 색상을 공유할 때 오인식이 발생했고, Geometric rule 기반(OCR 가정)은 곡선/장식형

타이포그래피에서 텍스트 누락이 잦았다. 또한 배경과 오브젝트를 억지로 분리하는 3단 분리는 Inpainting 범위를 과도하게 키워 복원 결과가 부자연스러워지는 한계가 확인됐다.

구분	상황	사례	문제
색상 기반 분리의 한계	텍스트와 배경 오브젝트가 유사한 색상을 공유	파란색 텍스트 + 아래로 흘러내리는 파란색 물감 효과가 혼재	물감 효과를 텍스트로 오인해 지워버리거나, 반대로 텍스트를 배경으로 오인해 생성 시 뭉개짐 발생
기하학적 규칙 기반의 한계 *(Geometric Rule-based Failure)*	텍스트가 수평/수직 정렬이 아닌 디자인 요소로 배치	(잔나비 - *Monkey Hotel*) 텍스트가 곡선 리본을 따라 휘어져 배치	"텍스트는 직선"이라는 OCR 규칙이 성립하지 않아, 리본 위 글자를 텍스트로 인식하지 못하고 누락

이에 따라 전략을 단순화하여 **오브젝트는 유지하고 텍스트 처리에 집중**하는 방향으로 전환했다. 먼저 Qwen(VLM)으로 "텍스트 분리가 시각적으로 유리한지"를 판단해, 네온사인·깃발 등 배경 디자인의 일부인 경우는 분리를 생략했다. 분리가 필요한 경우에는 **Adaptive Contrast Analysis**로 텍스트의 획(stroke) 중심 특징을 추출해 복잡한 배경/곡선 배치에서도 안정적으로 텍스트 영역을 확보했다. 그리고 이렇게 텍스트가 제거된 배경은 LaMa 모델로 주변 맥락에 맞게 자연스럽게 inpainting하고, 이후 다시 Qwen에 입력하여 Diffusion모델이 텍스트 간섭 없이 최적의 프롬프트를 자동으로 생성할 수 있게 했다.

지금까지의 내용을 정리하면 아래와 같이 요약할 수 있다.

- **Detection:** Gemini API로 텍스트 영역 좌표 식별
- **Inpainting:** LaMa로 텍스트 제거 및 배경 복원 → Clean Image 생성
- **Captioning:** Clean Image를 Qwen(VLM)에 입력해 Prompt 자동 생성(positive/negative는 분위기별 조절 지시 포함

c. Seamless loop video

왼쪽 영상과 같이, 단순히 diffusion 모델로 영상을 생성하면 첫 프레임과 마지막 프레임이 자연스럽게 이어지지 않아 루프 재생 시 경계에서 튀는 현상이 발생했다. 즉 모델의 시간축 전체를 **원형(cyclic) 시퀀스**로 만들도록 강제하기 어렵기 때문에 **seamless loop**를 안정적으로 보장할 수 없었다. 이 점을 해결하기 위해 FLF2V방식과, 루프 경계 후처리 봉합 방식 총 2개의 방식을 각각 시도해보았다.

초기에는 가장 간단하게 생각해볼 수 있는 **FLF2V(First/Last Frame 기반 생성)** 방식을 처음 시도해보았다. 하지만 이 방식은 중앙 구간이 사실상 랜덤하게 생성되는 경향이 있어, 결과적으로 양 끝 프레임만 우리가 원하는 결과가 나오고, 중간 프레임들은 다른 흐름으로 붕 뜨는 문제가 있었다. 즉, 끝점 일치는 되더라도 전체 시퀀스로 보게되면 일관된 루프라고 볼 수 없어서 채택하지 않았다.

위의 FLF2V방식의 실패로 완벽하게 원형을 생성하는 방법 대신, 루프 경계(seam)를 후처리로 봉합하는 방식으로 방향을 전환하였다. 이에 따라 다음으로 시도해본 것은 루프 경계 후처리 봉합(ComfyUI내부의 custom node를 사용)방식이였다. 해당 방식의 핵심은 다음과 같다.

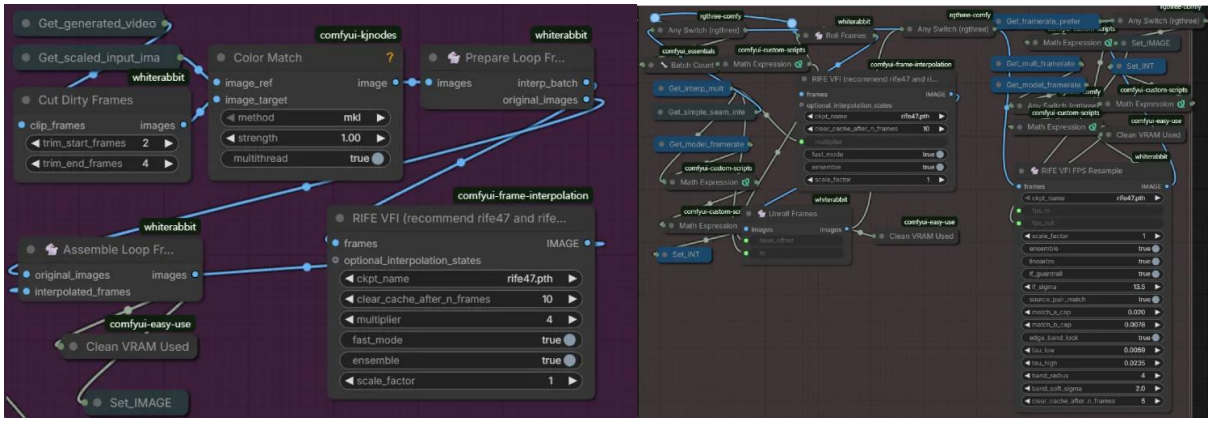
1. 경계에 사용할 **프레임을 정리**하고,
2. last -> first 사이를 **보간**으로 메우며,
3. seam이 눈에 띄지 않도록 **추가 보정**을 하는 것이다.

이를 위한 custom node들은 다음과 같다.

- 루프 생성 전역 체인: **whiterabbit** + **comfyUI-Frame-Interpolation**(RIFE)
- 제어/출력 보조: **rgthree**(Any Switch) + **VideoHelperSuite**(VHS)

위의 custom node들을 사용한 최종 파이프라인을 정리해보면, 일단 기존 방식과 동일하게 영상을 생성해준다. 이후

1. 프레임 정리 + 톤 정합
    - TrimBatchEnds로 앞 2프레임, 뒤 4프레임을 먼저 제거해 생성 과정에서 자주 나타나는 가장자리 불량 프레임을 먼저 정리한다.
    - 이후 ColorMatch로 원본 이미지 톤에 맞춰 색/밝기에 따른 반짝임(Flicker)현상을 줄인다.
2. 루프 seam 생성(봉합 과정)
    - PrepareLoopFrames로 [last, first] 2프레임 배치를 만든다.
    - RIFE VFI로 두 프레임 사이의 전이(transition) 프레임을 생성한다.
    - AssembleLoopFrames로 생성된 전이 프레임 중 중간프레임만(last/first 프레임 제외) 원래 생성된 영상 뒤에 붙여, 끝과 시작이 자연스럽게 이어지도록 한다.
3. 루프 품질 보정(더 자연스럽게)
    - RollFrames → RIFE VFI → UNrollFrames로 seam 구간을 더 부드럽게 만들고, seam이 눈에 띄는 현상을 줄인다.
    - RIFE_FPS_Resample로 재생 프레임레이트를 정리해 시간축 움직임의 균일성을 맞춘다.
    - 마지막으로 VHS_VideoCombine로 최종 루프 영상을 출력한다.



위와 같은 과정을 통해서 사용자는 끊김없이 재생되는 live2d seamless loop를 체험할 수 있게된다.

d. Live2d friendly motion

Wan 2.2 단독으로 Live2D 느낌의 영상을 생성하면 flicker, 객체 과도한 변형/난동, 원본 앨범 아트 보존 실패 등 품질 한계가 뚜렷했다. 이를 개선하기 위해 LoRA 학습과 prompt engineering 두 축으로 접근했다.

① LoRA 학습(“스타일”보다 “모션” 중심)

초기에는 다양한 앨범 아트를 모아 “album-art 도메인 LoRA”를 학습하는 방안을 검토했으나, 앨범 아트는 형식이 지나치게 다양해 학습 목표가 모호해질 가능성이 컸다. 프로젝트가 지향하는 결과물은 화려한 영상이 아니라 **원본을 보존하면서 미세하고 안정적인 움직임을 부여한 Live2D-friendly motion**이므로, 학습 목표를 **배경 중심의 모션 특성**으로 재정의했다.

사전 조사로 Civitai/Reddit/관련 논문을 탐색했으나, 사람/특정 객체 중심 LoRA가 대부분이라 배경 모션 중심 모델을 확보하기 어려웠다.따라서 내부적으로 LoRA를 직접 학습(캡션/파라미터를 달리한 2종)하고, 외부 Live2D LoRA(캐릭터 중심) 1종을 조합하여 총 3개 LoRA를 추론에 활용했다.

학습 툴로는 DiffSynth, ai-toolkit을 사용했고, (※ musubi tuner는 Wan 2.2 5B 미지원으로

제외) 학습에 사용한 데이터 셋은 Steam의 Wallpaper Engine 창작마당에서 영상 수집한 후 FFmpeg으로 **720p / 24fps / 4초(96 frames)** 정규화시켜 사용했다.

정규화된 영상 데이터와 함께 사용한 캡션 파일은 Gemini API로 자동 생성하되, 비용/토큰 최적화를 위해 전체 추론 대신 **일부 프레임 기반**으로 구성했다. 트리거 토큰은 프롬프트로 넣어주는 대신 최종 캡션 후처리로 일괄 삽입하여 존재 여부를 무조건 보장하도록 했다. 다른 프롬프트와 구분을 두기 위해 리트 형식으로 지정했다.

캡션 규칙으로는 (1) 장면/구도(단일 프레임로 파악 가능) + (2) 객체별 모션(비디오에서 관측) 을 포함하고, 논리적으로 어색한 문장은 LLM으로 후처리해 정합성을 확보했고, VRAM 제약을 고려해 **해상도 중심 LoRA**와 **프레임(시간) 중심 LoRA**로 분리 학습했다.

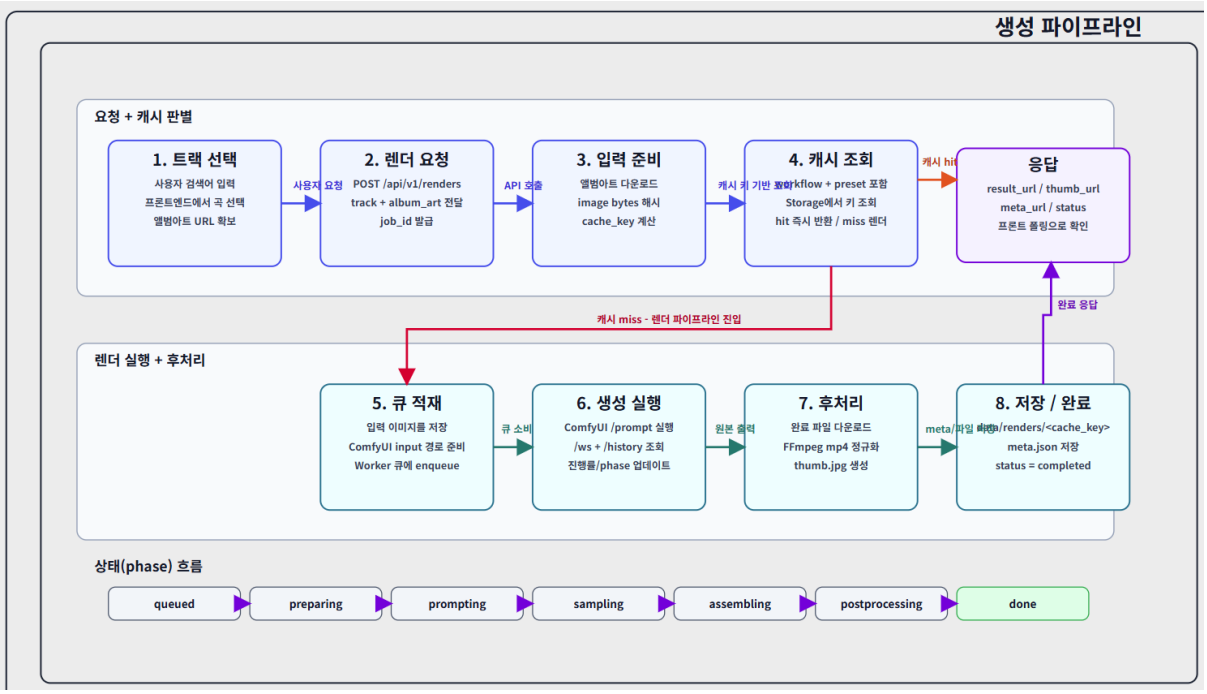
그 결과 LoRA 1~2개보다 **3개 조합**에서 안정성과 품질이 가장 좋은 것을 확인할 수 있었다.(모션 자연스러움/왜곡 억제 측면).

② Prompt Engineering(VLM 자동화 + 가드레일)

앞서 언급한 바와 같이 본 프로젝트에서는 모델 성능을 좌우하는 입력 프롬프트를 "수동 작성"에만 의존하지 않고, VLM 기반 자동 생성 + 규칙 기반 가드레일 구조로 설계했다.

- Positive Prompt 자동 생성: 텍스트가 제거된 앨범 아트 이미지를 입력으로 Qwen(VLM)을 활용해, diffusion 모델에 최적화된 positive prompt를 정해진 템플릿/규격에 맞춰 생성했다.
- Negative Prompt 설계: 앨범 아트 도메인에서 반복적으로 발생하는 아티팩트(텍스트 왜곡, 과도한 변형, 워터마크 등)를 억제하기 위해, 공통으로 적용 가능한 도메인 특화 negative prompt를 직접 설계해 일관된 품질을 확보했다.

3-2. 서비스 플로우



본 서비스는 사용자가 음악 검색 → 앨범아트 수집 → 생성 파이프라인 실행 → 루프 결과 제공으로 이어지는 비동기 처리 기반의 End-to-End 흐름을 갖는다.

사용자가 검색창에 음악을 입력하면, 백엔드는 iTunes Search API를 호출해 곡 목록과 앨범아트(이미지)를 가져온다. 사용자가 곡을 선택하면 해당 곡 정보를 기반으로 YouTube Data API를 통해 유튜브에서 재생 가능한 링크(또는 video id)를 조회해 프론트엔드에 전달한다.

이후 백엔드는 선택된 곡의 앨범아트 이미지 바이트(또는 해시) 를 기준으로 기존에 생성된 Live2D 루프 영상의 캐시 여부를 확인한다. 동일 앨범아트에 대한 결과물이 이미 저장되어 있다면, 저장소(현재는 로컬)에서 결과 영상을 즉시 로드하여 사용자에게 제공함으로써 GPU 자원과 생성 시간을 절약한다.

캐시가 없는 경우에는 백엔드가 ComfyUI와 통신하여, iTunes API를 통해 확보한 앨범아트 이미지를 입력으로 전달하고 Live2D 스타일 루프 영상 생성 파이프라인을 실행한다. 생성이 완료되면 결과 mp4(및 썸네일/메타데이터)를 저장소에 저장하고, 프론트엔드는 저장된 결과를 로드하여 사용자에게 재생/표시한다

IV. 프로젝트 수행 결과

정적인 앨범 아트를 동적인 Live2D 스타일의 루프 영상으로 변환하기 위해, 전처리-생성-후처리로 이어지는 유기적인 파이프라인을 구축했다. 앞서 언급했던 내용들을 모두 모아 정리하면 아래와 같다.

- 전처리 :
  - 문제: 이미지 전체를 Diffusion 모델에 입력 시, 앨범 아트의 텍스트(타이틀, 아티스트명)가 뭉개지는 현상 발생.
  - 해결: Gemini API로 텍스트 영역을 감지하고, LaMa Inpainting 모델로 텍스트를 제거한 'Clean Image'를 생성하여 Diffusion 모델의 입력으로 사용.
  - 프롬프트 최적화: **Qwen(VLM)**을 통해 이미지를 분석하고, Positive Prompt를 자동 생성하여 사용자 개입 없이도 고품질의 영상을 유도.
- 고품질 비디오 생성 :
  - Base Model: Wan 2.2 5B (I2V) 모델을 채택하여 고해상도 영상 생성.
  - Live2D 스타일 적용: Steam Wallpaper Engine 데이터셋으로 자체 학습한 LoRA를 적용하여, Live2D 특유의 부드럽고 몽환적인 움직임 구현.

Seamless Loop 구현 및 후처리

단순한 비디오 생성을 넘어, '앨범 아트'라는 특성에 맞춰 무한 반복 재생이 가능한 결과물을 만들었습니다.

- Loop 최적화: 오픈소스 WhiteRabbit 노드(PrepareLoopFrames → AssembleLoopFrames)를 활용하여 영상의 시작과 끝 프레임이 자연스럽게 이어지도록 설계.

- 프레임 보간 (Interpolation): RIFE VFI (Multiplier 4)를 적용하여 프레임 간 끊김을 제거하고 부드러운 움직임(Fluid Motion)을 확보.
- 규격화: FFmpeg 자동화 스크립트를 통해 웹 서비스에 최적화된 .mp4 포맷 변환 및 썸네일 추출.

V. 자체 평가 의견

잘한 점 & 성과

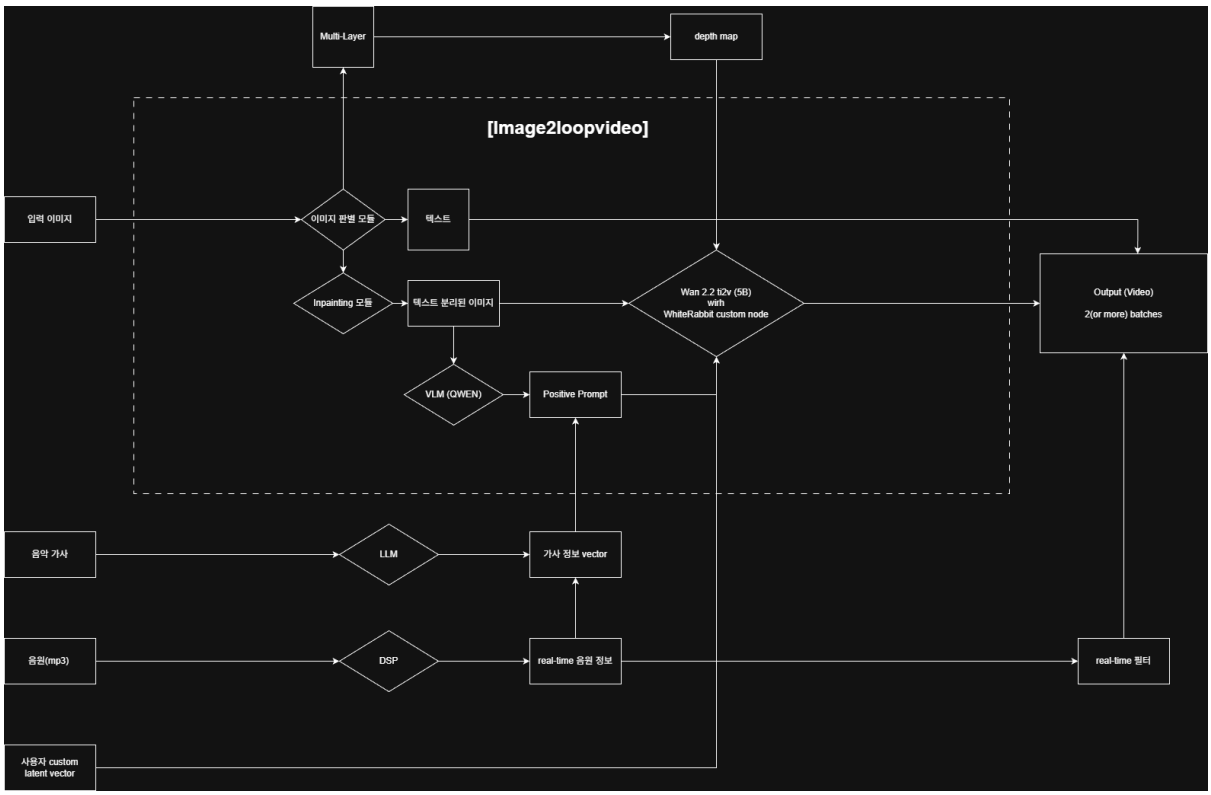
- 최신 모델 도입: Wan 2.2와 Qwen 등 최신 모델을 빠르게 파이프라인에 통합하여 고품질 결과물 산출.
- 문제 해결력: 텍스트 뭉개짐 문제를 Inpainting과 VLM의 결합으로 창의적으로 해결함.
- Loop문제 개선: 단순 ping-pong 대신 WhiteRabbit 기반 경계 보강/블렌딩으로 "끊김 없는 루프"에 집중해 사용자 체감 품질을 개선함
- LoRA학습: 외부 배포 LoRA가 캐릭터에 편향되어 있어 도메인에 필요한 배경 LoRA 확보가 어려웠고, 이를 해결하기 위해 데이터셋을 직접 수집·정제하여 LoRA 학습까지 end-to-end로 완료.

아쉬운 점 & 한계

- 리소스 제약: 고성능 모델(5B) 사용으로 인해 로컬 환경에서의 추론 속도 및 메모리 부하가 존재.
- ComfyUI 의존 구조: 워크플로우로 빠르게 검증하고 결과를 만들 수 있었지만, 핵심 기능을 .py 모듈로 분리해 독립 파이프라인을 구축하지 못해 코드 레벨 통제/확장성이 제한됨.
- T2V 루프 방법의 적용 제약: Mobius 등 루프 관련 접근을 조사했으나, T2V 전제를 강하게 갖는 방식은 입력 이미지 보존이 중요한 TI2V 파이프라인에 그대로 적용하기 어려워 실사용 단계로 연결하지 못함.

- 해상도 제한: 현재 1:1 비율에 최적화되어 있어, 16:9 등 다양한 디스플레이 규격 대응이 부족함.
- 정량 평가 체계 부족: 정량 평가 체계(예: human score 기반 품질평가, seam 정량화)의 부족으로 개선 효과를 수치로 강하게 주장하기 어려움

향후 계획 (Future Work)



- Multi-modal로 확장

앨범 아트만 입력으로 받는 것 대신, 음원(mp3), 가사(text)등의 정보도 입력으로 받을 수 있게 하여 보다 능동적인 loop을 생성할 수 있도록 합니다.

- 16:9 해상도 출력 지원

  비디오 출력 해상도를 1:1로 출력하는 것을 넘어서서, 대부분의 디스플레이의 규격에 따른 output도 지원할 수 있도록 합니다.

- ComfyUI 의존성 줄이기.

- 모델 경량화 및 최적화를 통해 비디오 생성 시간 절감하기.

- 보다 일반적인 성능을 기대할 수 있도록, 파이프라인 및 모델 고도화

  - 입력받은 앨범 아트의 카테고리를 구분짓고, 적절한 depth map로 쪼개서 입력받아 각각 diffusion 적용.
  - more expert LoRA (for worst case) 또는 agent와 함께하는 자동 학습 모델로 개선
  - human score을 통한 모델 성능 개선, 더 나아가 사용자 기반의 출력물 생성.
- transition effect
- DB 구축 및 데이터베이스화
- 유저 친화적인 프론트앤드 및 백앤드 보완.

  로그인된 스트리밍 사이트로부터 현재 재생중인 재생목록 불러오기. 또는 재생중인 음악 감지 후 어떤음악인지 확인. 이후 별도의 조작 없이 자동으로 display

VI. 개인 회고

고대영

1. 나는 내 학습목표를 달성하기 위해 무엇을 어떻게 했는가?

   처음 시도에서는 어떻게서든 깔끔한 레이어를 만들기 위해 우선적으로 배경, 오브젝트, 텍스트 이렇게 3개의 레이어 분리를 시도했습니다. 그러나 오브젝트의 크기가 너무 크면 배경을 impainting할 시에 모델이 오작동하여 의도대로 채워지지 않고 이상한 오브젝트가 채워지는 등의 문제가 있었습니다. Diffusion 모델이 글자를 뭉개는 현상을 막기 위해, 단순한 색상 분리부터 기하학적 규칙 적용, 그리고 최종적으로 적응형 명암 분석에 이르기까지 다양한 기법을 실험하고 검증했습니다.

2. 전과 비교해서, 내가 새롭게 시도한 변화는 무엇이고, 어떤 효과가 있었는가? (2 번째 프로젝트부터 해당)

   지난 프로젝트에서는 성능이 좋은 모델을 찾는데 급급하였고, 디테일한 부분을 보지 못한채 그저 모델의 순수 능력에 의존했습니다. 그러나 이번 최종 프로젝트에서는 무조건적인 분리가 아닌, VLM에게 앨범을 보고 자율적으로 판단하게 하는로직을 워크플로우에 심었습니다. 덕분에 예외 케이스를 효과적으로 처리하여 결과물의 자연스러움을 극대화했습니다.

3. 마주한 한계는 무엇이며, 아쉬웠던 점은 무엇인가?

   1. Inpainting의 한계 텍스트를 성공적으로 분리하더라도, 그 빈자리를 채우는 Lama inpainting에서 배경의 복잡한 패턴을 완벽하게 복원하지 못해 미세한 블러 자국이 남는 경우가 있었습니다. 텍스트 분리에 집중하느라Inpainting 후처리 기술까지는 깊게 파고들지 못한 점에 아쉬움이 남습니다.
   2. Workflow 복잡도와 연산 비용 완벽한 결과를 위해 ComfyUI에서 무거운 Wan 모델과 LoRA, text encoder 등 여러 실험을 병렬적으로 수행하느라 워크플로우가 매우 무거워졌습니다. 실시간 서비스 관점에서는 Latency를 획기적으로 줄이지 못한 점이 아쉬움으로 남습니다.

4. 한계/교훈을 바탕으로 다음 프로젝트에서 시도해보고 싶은 점은 무엇인가?

   '텍스트를 분리하는 방식' 또한 VLM에게 1차적으로 판단시켜, VLM을 총 3번 사용하는 (텍스트 분리 여부 - 분리한다면 어떤 방식으로 분리할지 - 분리한 후의 배경을 보고 프롬포트 판단) 더욱 구체적인 체계를 잡고 싶습니다.

   또한 ComfyUI의 의존성을 줄이고 언제든지 보수유지가 가능한 완전 코드화가 목표입니다. 이는 추후 개발에서 반드시 시도해보고 싶은 영역입니다.

고재완

1. 나는 내 학습목표를 달성하기 위해 무엇을 어떻게 했는가?

- 모델 검증: CogVideoX-5B-I2V와 Wan2.1-I2V-14B 등 최신 모델을 ComfyUI 환경에서 구동하며, 기본 추론(Inference)만으로 우리가 원하는 'Live2D 스타일'을 구현할 수 있는지 실험했습니다.
- 데이터 파이프라인 연구: 앨범 아트 자동 수집을 위해 Musicbrainz API 연동을 시도하며, 대규모 메타데이터를 서비스에 통합하는 방법을 모색했습니다.

2. 전과 비교해서, 내가 새롭게 시도한 변화는 무엇이고, 어떤 효과가 있었는가?

- 시도: VRAM 요구량이 높은 Wan2.1-14B 모델을 최적화하여 ComfyUI 워크플로에 통합하고 추론 테스트를 진행했습니다.
- 효과: 본 모델만으로는 원하는 퀄리티(Live2D 느낌)가 나오지 않음을 확인했지만, 이 과정을 통해 단순 프롬프트 엔지니어링만으로는 한계가 있으며, 반드시 LoRA 파인튜닝이 필요하다는 기술적 결론을 빠르게 도출할 수 있었습니다.

3. 마주한 한계는 무엇이며, 아쉬웠던 점은 무엇인가?

API 데이터 품질 이슈: Musicbrainz API를 연동했으나, 정제된 앨범 아트뿐만 아니라 'CD를 들고 찍은 사진'이나 '스캔 노이즈가 심한 이미지' 등이 섞여 들어오는 문제를 발견했습니다. 이를 프로그래매틱하게 필터링할 방법을 시간 내에 찾지 못해 결국 해당 API 사용을 포기했습니다.

4. 한계/교훈을 바탕으로 다음 프로젝트에서 시도해보고 싶은 점은 무엇인가?

- Robust Data Pipeline: 다음 프로젝트에서는 외부 API의 노이즈 데이터도 잘 분류할수있는 방법을 생각해서 쓸모없는 데이터를 걸러내는 전처리 파이프라인을 직접 구축해보고 싶습니다.
- Efficient Fine-tuning 마스터: 단순히 모델을 가져다 쓰는 것을 넘어, LoRA나 Dreambooth 등 경량화 학습 기법을 심도 있게 연구하여, 도메인에 맞는 모델을 직접 튜닝하고 최적화 하고 싶습니다.

양지훈

1. 나는 내 학습목표를 달성하기 위해 무엇을 어떻게 했는가?

ComfyUI 기반 생성 과정을 재현 가능한 워크플로우로 정리하고, seamless loop 품질을 개선하는 것을 학습목표로 잡았다. 앨범아트 입력부터 결과 저장/서빙까지 이어지는 흐름을 기준으로 워크플로우 후보를 비교·실험하며 최종 파이프라인을 수렴했다. 루프 구현 방법을 조사하던 중 WhiteRabbit 커스텀 노드를 찾아 적용했고, last-first frame seam 구간 보강 방식으로 끊김 체감을 줄였다. LoRA 학습도 직접 수행해 보았으며, 학습 안정성과 재현성을 위해 비디오 데이터 규격을 24fps, 96frames로 통일했다.

2. 전과 비교해서, 내가 새롭게 시도한 변화는 무엇이고, 어떤 효과가 있었는가?

이전에는 대회 제공 데이터로 실험을 반복하는 경험이 대부분이었다. 이번에는 LoRA 학습을 위해 데이터셋을 직접 수집·정제했고, 단순 수집을 넘어 학습 가능한 형태로 규격화(24fps/96frames) 했다. 그 결과 실험 조건이 통일되어 재현성과 비교 가능성이 좋아졌고, 그 결과 입력 길이/프레임 수가 고정되면서 학습 설정을 잡기가 수월했고, 시행착오를 줄이는 데 도움이 됐다.

3. 마주한 한계는 무엇이며, 아쉬웠던 점은 무엇인가?

결과적으로 ComfyUI 의존 구조로 남은 점이 가장 아쉬웠다. 워크플로우에서 검증한 기능을 .py 모듈로 분리해 우리만의 파이프라인을 구축하고 싶었지만, 기간 내 서비스 완성을 우선하며 코드 레벨 통제/확장성을 충분히 확보하지 못했다. 또한 최종 루프 방식이 마지막–처음 프레임 사이 보간(interpolation) 기반이라, 비가역 모션이 강한 케이스에서는 미세한 부자연스러움이 잔존했다. 루프 품질 개선을 위해 Mobius(텍스트→seamless loop

T2V) 접근도 검토했지만, 입력 이미지를 유지하는 TI2V(앨범아트 조건) 파이프라인과 전제가 달라 그대로 적용하기 어려워 채택하지 못했다.

4. 한계/교훈을 바탕으로 다음 프로젝트에서 시도해보고 싶은 점은 무엇인가?

다음 프로젝트에서는 ComfyUI에서 검증한 흐름을 참고하되, 핵심 단계를 파이썬 모듈로 분리해 우리 서비스에 맞는 독립 파이프라인을 구축해보고 싶다. 구체적으로 전처리/입력 정규화, 모델 실행 래퍼, 루프 보강, 캐시·저장 로직을 분리해 코드 레벨에서 통제 가능한 구조로 옮기고, 실험/운영 환경에서도 동일한 실행이 가능하도록 만들고 싶다. 또한 루프 품질은 마지막–처음 프레임 보간(커스텀 노드)에만 의존하지 않고, 생성 단계에서 루프가 자연스럽게 나오도록 유도하는 방향도 함께 시도해보고 싶다. 이를 위해 루프에 유리한 케이스(주기성이 있는 모션)를 데이터 수집 단계에서 더 의도적으로 선별하고, 생성 후에는 seam 구간만 메꾸는 수준을 넘어 프레임 간 일관성이 유지되도록 간단한 후처리/제약을 추가해 잔여 부자연스러움을 줄이고 싶다. 그리고 Mobius처럼 T2V 전제의 루프 방법은 TI2V에 그대로 가져오기 어려웠던 만큼, 다음에는 입력 이미지(앨범아트) 보존을 전제로 적용 가능한 형태만 골라 실험하고, 우리 파이프라인에 맞는 대안을 정리해 적용해보고 싶다.

최호준

1. 나는 내 학습목표를 달성하기 위해 무엇을 어떻게 했는가?

   프로젝트 주제를 제안하고, 구상했던 것들을 팀원과 교류하며 발전시켜 나갔다. 정보 수집 및 LoRA학습을 도맡아 진행했는데, 운이 좋게도 의도한 대로 LoRA가 나온 것 같았다. 개별적으로 좀더 검증을 할 수 있었으면 좋았을 것 같긴 하다.

2. 전과 비교해서, 내가 새롭게 시도한 변화는 무엇이고, 어떤 효과가 있었는가?

   일전에는 단순히 점수를 조금이라도 더 높일 수 있는 것에 집중했고, 어느 순간 점수가 오르지 않고 막히는 부분을 만날 때마다 답답했던 순간도 있었지만, 이번에는 그와는 다르게 자체적으로 정립한 목표를 이루기 위해 다양한 시도를 자유롭게 해볼 수 있었던 것이 재밌었던 것 같다.

3. 마주한 한계는 무엇이며, 아쉬웠던 점은 무엇인가?

VRAM의 한계와, 학습시킨 LoRA가 제대로 동작할 지 확인하기까지 시간이 매우 오래 걸렸기에 좀더 다양한 시도를 해볼 수 없었던 것이 아쉽다. 특정 motion위주가 아닌, 보다 일반적인 motion을 학습시킨 LoRA을 만들어 보고 싶었고, 이를 통해 positive prompt의 중요도를 조금 절감할 수 있기를 기대했는데 이 가능성에 대해 직접적으로 확인할 수 없던 것이 기억에 남는다.

4. 한계/교훈을 바탕으로 다음 프로젝트에서 시도해보고 싶은 점은 무엇인가?

이번 프로젝트에 이 Wrap-Up 리포트가 마침표라고 생각하지는 않는다. 좋은 팀원들을 만났고, 또 그 다음을 일단 지금으로서는 모두가 같은 목표를 바라보고 있기에, 함께 생각했던 최종 결과물을 만들어 보고 싶다. 아직 개선해야 할 부분도 많고 배워나가야 할 부분이 많지만, 지금까지의 경험을 바탕으로 본 프로젝트를 멋지게 매듭짓는 것이 목표다.

VII. Reference

1) Papers

- Bi, X., Yuan, J., Liu, B., Zhang, Y., Cun, X., Pun, C.-M., & Xiao, B. (2025). Mobius: Text to Seamless Looping Video Generation via Latent Shift. arXiv:2502.20307.
- Huang, Z., He, Y., Yu, J., Zhang, F., Si, C., Jiang, Y., … Liu, Z. (2024). VBench: Comprehensive Benchmark Suite for Video Generative Models. CVPR 2024.

---

2) Code / Frameworks (LoRA 학습·추론)

- Ostris. ai-toolkit (GitHub repository).
- ModelScope Community. DiffSynth-Studio (GitHub repository).
    - (문서) DiffSynth-Studio docs

---

3) ComfyUI Custom Nodes (워크플로 구성 요소)

- kijai. ComfyUI-WanVideoWrapper (custom node).
- 1038lab. ComfyUI-QwenVL (custom node).
- Artificial-Sweetener. comfyui-WhiteRabbit (custom node).
- Fannovel16. ComfyUI-Frame-Interpolation (custom node).
- Kosinkadink. ComfyUI-VideoHelperSuite (custom node).

그 외:

- rgthree. rgthree-comfy — https://github.com/rgthree/rgthree-comfy
- cubiq. comfyui_essentials — https://github.com/cubiq/ComfyUI_essentials
- pythongosssss. comfyui-custom-scripts — https://github.com/pythongosssss/ComfyUI-Custom-Scripts
- Smirnov75. comfyui-mxtoolkit — https://github.com/Smirnov75/ComfyUI-mxToolkit
- yolain. comfyui-easy-use — https://github.com/yolain/ComfyUI-Easy-Use
- aining2022. ComfyUI_Swwan — https://github.com/aining2022/ComfyUI_Swwan

4) Workflow 참고(외부 게시글/페이지)

- RunningHub workflow post: https://www.runninghub.ai/post/2008809599616421889
- Civitai model page(로그인 필요): https://civitai.com/models/1264662?modelVersionId=2080103