종합설계 완료보고서

1. 수행 결과보고

세부과제명	Video Inpainting을 활용한 동영상 편집 기술 개발

□ 세부과제 개요 및 필요성

본 과제는 기억하고 싶은 순간을 동영상으로 촬영했는데 원치 않는 사물이나 사람이 같이 찍히는 경우 그러한 물체 즉, 동영상 속 원하는 객체를 지정해 Masking을 해 손상시켜준다. 그리고 GAN을 통해 손상된 이미지를 메꾼 생성된 각 frame별 이미지를 제공해 동영상에 대한 만족도, 초상권 문제 회피, 동영상 결함 복구 등의 문제를 해결하고자 한다.

이 기술을 통해 Image에만 적용되던 AI 지우개와 같은 편집 기술과 초상권을 회피하기 위해 video 위에 모자이크 처리를 하는 등의 편집 기술을 대신해 해당 동영상에서 원하는 물체를 Masking을 통해 손상시킨 후 채워 넣는 형식인 Inpainting을 돌입해 원본에 손상을 입히는 편집 기술이 아닌 생성하는 형태가 적용된다.

□ 세부과제 개발 방법 및 수행 과정

- 세부과제 개발 방법

1. Inpainting을 할 Video를 불러온다.
2. 동영상 각각의 frame을 나누어준다.
3. 원하는 물체를 지정하는 방법으로 하나의 frame에 직접 Masking을 한다.
4. Masking을 된 물체를 다른 frame마다 Tracking 후 Masking을 하는 모델 PCVOS를 활용한다.
5. PCVOS의 단점인 첫 frame에서만 적용이 되는 것을 역재생 개념을 적용해 순방향 역방향 2번 인식하게 코드를 수정한다.
6. 원본 동영상 frame들과 PCVOS에서 결과로 나온 Masking이 된 동영상 frame들을 받아 Vision transformer의 encoder에 적용해 이미지의 정보를 추출 해낸다.
7. 이때 이미지를 넣을 때 이미지들이 서로 겹치게 넣어서 Vision transformer의 단점인 각 이미지의 patch간의 연관성이 낮은 것을 해결한다.
8. Vision transformer에서 Linear Classifier Head 부분을 삭제하고 GAN을 연결해 각각의 patch들의 토큰들을 입력해준다.
9. 각각의 토큰들이 GAN을 통해 patch단위 이미지를 생성한다.
10. 생성된 이미지 또한 초기에 넣을 때와 마찬가지로 겹치게 나오기 때문에 겹치는 부분은 평균으로 생성한다.
11. Decoder를 통해 원본 이미지와 같은 형식으로 생성된 patch단위 이미지들을 합쳐준다.
12. 위 과정을 진행할 때 Deep VideoInpainting에서 시간의 축에 대한 문제 해결 방법인 이미지를 생성할 때 5개의 frame을 넣는 것을 여기에도 똑같이 적용한다.
13. Video를 불러와 각각 frame으로 이미지를 생성한 후 수정된 PCVOS를 통해 Masking 된 이미지를 생성해주고 Inpainting을 적용해 결과 동영상을 출력해준다.
14. UI는 크게 영상 불러오기, 1개의 frame에 직접 그려서 Masking 하는 것, PCVOS를 통하는 연결, Inpainting이 적용되는 것 이렇게 4개의 버튼이 필요하다.

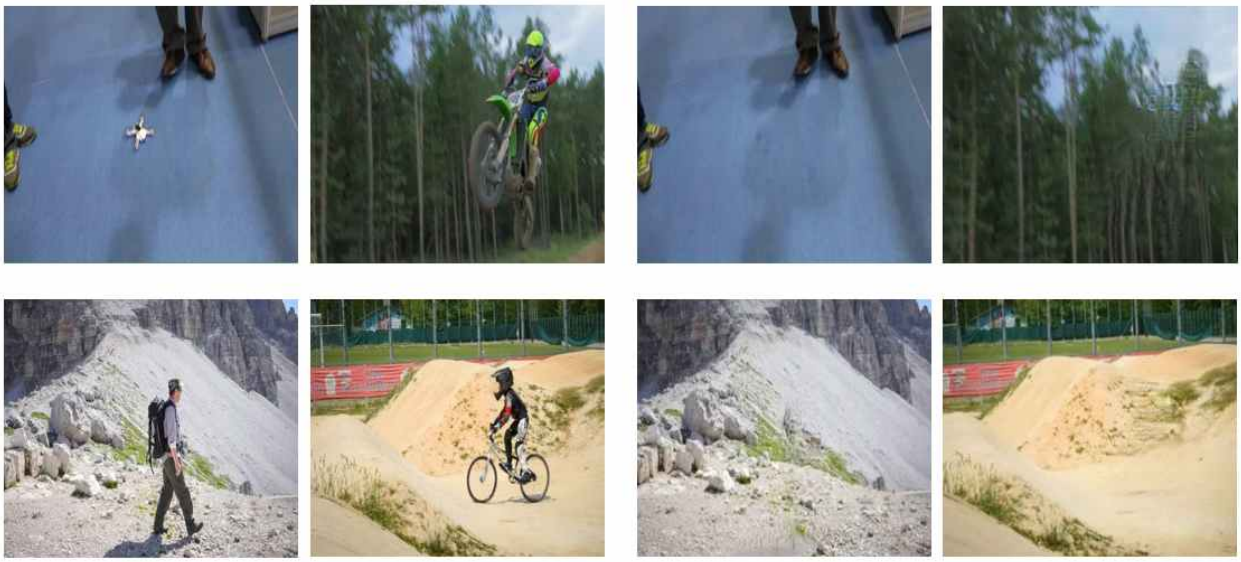
- 수행 과정

1. UI 담당자는 PyQt를 활용해 구조를 형성했다.
2. 이때 PyQt version에 따라 지원하는 함수가 다양하므로 PCVOS와 Inpainting에 쓰는 version과 최적화를 진행했다.
3. Masking 담당자는 OpenCV를 활용해 Image 위에 그림을 그려 Masking 파일을 생성하는 것을 제작했다.
4. PCVOS를 Window에서 실행이 안돼, Linux 환경에서 모든걸 다시 시작했다.
5. PCVOS 코드를 수정해 역재생 개념을 추가해 frame 어디서든 Tracking을 할 수 있게 했다.
6. Masking 파일을 생성하고 PCVOS에 인식이 안되는 문제를 해결했다.
7. PCVOS에서 사람 한명이 나왔을 때는 그 사람만 인식하는데 사람 1이 지나가고 사람 2가 새로 나오면 그것 또한 Masking을 진행해서 Classifier가 진행되게 코드를 수정했으나 해결하지 못했다.
8. Inpainting 담당자는 Vision Transformer 모델에 GAN이 포함된 TransGAN을 활용한 STTN 모델을 직접 수정한다.
9. 위에서 설명했듯이 시간의 축을 해결하기 위해 5개의 frame을 한번에 Vision Transformer에 입력하고 각 patch간의 연관성을 만들기 위해 겹치는 부분이 있게 patch를 생성했다.

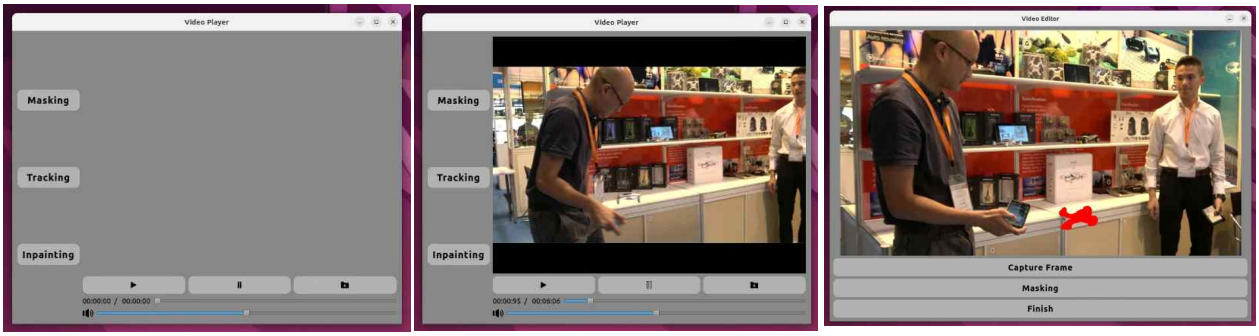
10. 각 patch의 크기와 겹치는 부분을 여러 번 시도 끝에 patch 크기는 (7,7) stride는 (3,3)으로 진행하였다.
11. Vision Transformer에 Encoder를 할 때 Conv2d를 통해 3->64->64->128->256->512로 진행했다.
12. Train에서 훈련 반복 횟수는 4만번으로 적용했고 batch_size는 컴퓨터 성능으로 인해 1로 진행했다.
13. 훈련 과정 총 반복은 5.3만번을 진행했다.

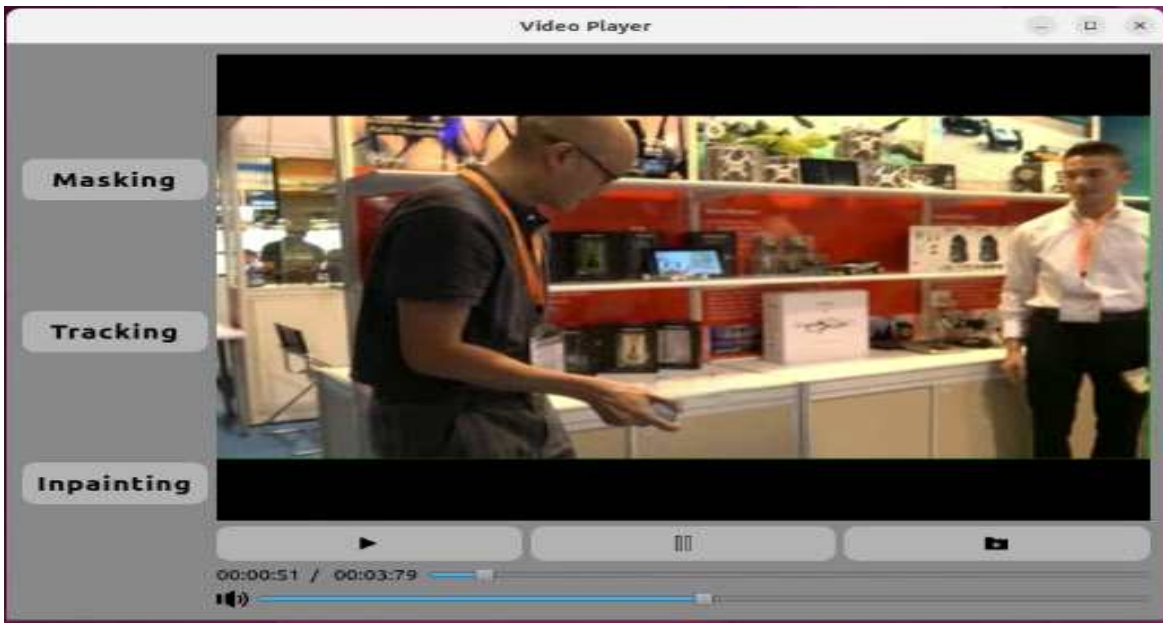
□ 결과

다른 Video Inpainting들과 비교하는 과정은 시간 관계상 생략되었고 UI 구조에 맞춰 기능들을 제공하고 서로 다른 호환 때문에 몇몇 기능들은 생략되었지만 핵심 부분들은 구현되었다.



Video에서 Masking이 차지하는 비율이 크다면 각 patch 크기 모양의 패턴이 생겨서 차지하는 비율을 최소화하는 것부터가 중요하다. 한 번에 모든 것을 지우기보다 여러 단계를 걸쳐서 적용하는 것 또한 좋은 방법이다.





2. 활용방안 및 기대효과

□ 활용방안

현재 갤럭시 휴대폰의 갤러리에는 AI 지우개를 제공하고 있다. 현재로선 이미지에만 적용이 되고 있는데 이 프로그램은 이미지가 아닌 Video로 제공된다. 새로운 영상 편집 기술로 인해 여태 활용하지 못했던 부분들에 활용을 할 수 있게 된다.

예시를 들자면
1) 원치 않는 사물이나 사람이 같이 찍히는 경우 그러한 물체 즉, 동영상 속 원하는 객체를 제거할 수 있게 된다.
2) 동영상 원본에 손상을 입히는 기술인 모자이크와 같은 것들을 대체할 수 있다.
3) 의료 관련 동영상과 같은 동영상이 손상된 경우 그 부분을 복원할 수 있다.

□ 기대효과

1) 모자이크와 같은 기능을 이 기술로 대체 가능하다.
2) 동영상을 원하는 대로 수정해 만족도를 높일 수 있다.
3) 초상권과 같은 문제를 해결할 수 있다.
4) 초상권에 민감한 영상을 다루는 사람들이 이 기술을 쓸 것을 예측할 수 있다.

□ 성능 개선점

현재의 문제가 가장 큰 것은 Image에 Masking이 어느 부분 비율을 차지하게 된다면 패턴 형식의 이미지가 생성되어 오히려 모자이크와 같은 효과가 발생하는데 kernel 크기를 (7,7)로 해서 Masking 된 부분들만 있는 patch들이 생긴다. 학습을 통해 조금씩 채워 가지만 아예 정보가 없는 patch인 경우 채워 넣는 것이 안 되어서 패턴이 생성된다. 이러한 문제점을 해결하기 위해서 (7,7)로 학습을 한 뒤 (10,10) 크기로 한번 더 학습을 시킨 후 다시 (7,7)로 학습을 하는 것이다. 그러면 패턴을 없애고 더욱 성능이 좋게 결과가 나올 것이다.

PCVOS 부분에서 Classifier이 제대로 작동되지 않아 여러 물체가 순서대로 나오면 동일한 물체로 인식되는 문제를 해결하면 원하는 물체만 제거하는데 더욱 효과적일 것이다. PCVOS가 아닌 다른 Tracking & Masking 기술을 대체해 활용할 수 있을 거라 본다.

□ 코드 작성 부분

PCVOS를 통해 구현한 Masking 수정 부분은 파일을 불러올 때 00024.png면 다음 파일이 00025.png를 불러오고 그 후 끝 까지 갔을 경우 00000.png or 00023.png로 역재생과 동일하게 가게끔 코드를 수정했다.

Video Inpainting에서 비중이 제일 큰 부분은 STTN을 기반으로 했다. 기존에 있는 코드들을 그대로 썼다. GAN을 활용한 이미지 생성 부분의 코드를 수정했으며 model 쪽 ViVE.py에서 patch들을 합치는 부분을 추가했고 각각의 train, feature 추출 부분을 새로 정의 하였다.

제일 많은 시간을 투자 했던 부분은 model이며 위 성능 개선점에 말했듯 kernel 크기를 7,7 10,10, 7,7 이렇게 3번 학습을 시키는 것을 첫 번째 목표로 하고 코드를 수정했으나 10,10 모형에서 GAN을 연결하는 부분에 계산 오류가 발생 및 시간 부족으로 정지하게 되었다.

Train을 53회 진행 하였고 RTX 3060 기준으로는 1회당 2시간이 걸렸다.
폴더 안에 있는 .pth는 성능의 차이가 있을 것이다. 학습량 조절 및 시간 조절을 위해서 수정을 했다.

PCVOS에 input data를 넣기 위해 OpenCV를 활용해 user가 직접 masking을 하는 파일을 생성했다.
전반적인 UI 및 파일들은 직접 구현했다.

ViVE 시연 영상 : https://youtu.be/BL2Myew0cjo

□ 참고 문서

1. PCVOS 참고 : https://github.com/pkyong95/PCVOS
2. TransGAN 참고 : https://github.com/VITA-Group/TransGAN
3. ICT 참고 : https://github.com/raywzy/ICT
4. STTN 참고 : https://github.com/researchmm/STTN
5. CNN Video Inpainting 참고 : https://github.com/mcahny/Deep-Video-Inpainting
6. Deep Featuer learning 참고 : https://scholar.google.com/citations?view_op=view_citation&hl=ko&user=qnbpG6gAAAAJ&citation_for_view=qnbpG6gAAAAJ:HDshCWvjkbEC