EmoGE'T – **Emoticon Ge**neration by **T**obig's

투빅티콘

Image2Video 기반 나만의 움직이는 이모티콘 생성

신민정 이예지 이유민 최혜빈
김상현 김민경 정재윤 한유진

CONTENTS

00. Intro

이모티콘

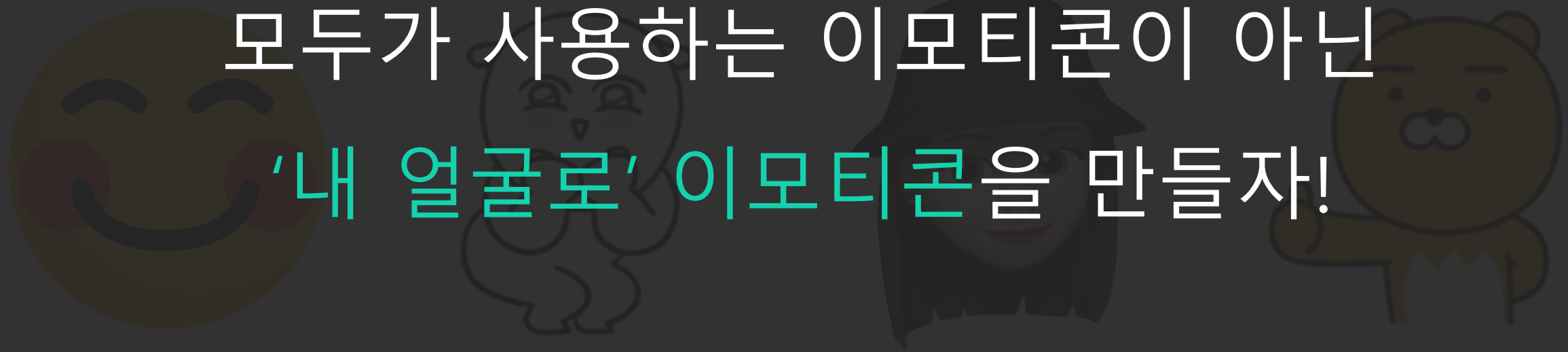
: 감정이나 느낌을 전달하기 위해 사용하는 문자&그림




00. Intro

이모티콘

: 감정이나 느낌을 전달하기 위해 사용하는 문자&그림



모두가 사용하는 이모티콘이 아닌

'내 얼굴로' 이모티콘을 만들자!

EmoGE'T – **Emoticon Ge**neration by **T**obig's

01

Tobigticon

01. Tobigticon

step1. 사용자 얼굴 이미지 업로드

Input your own face

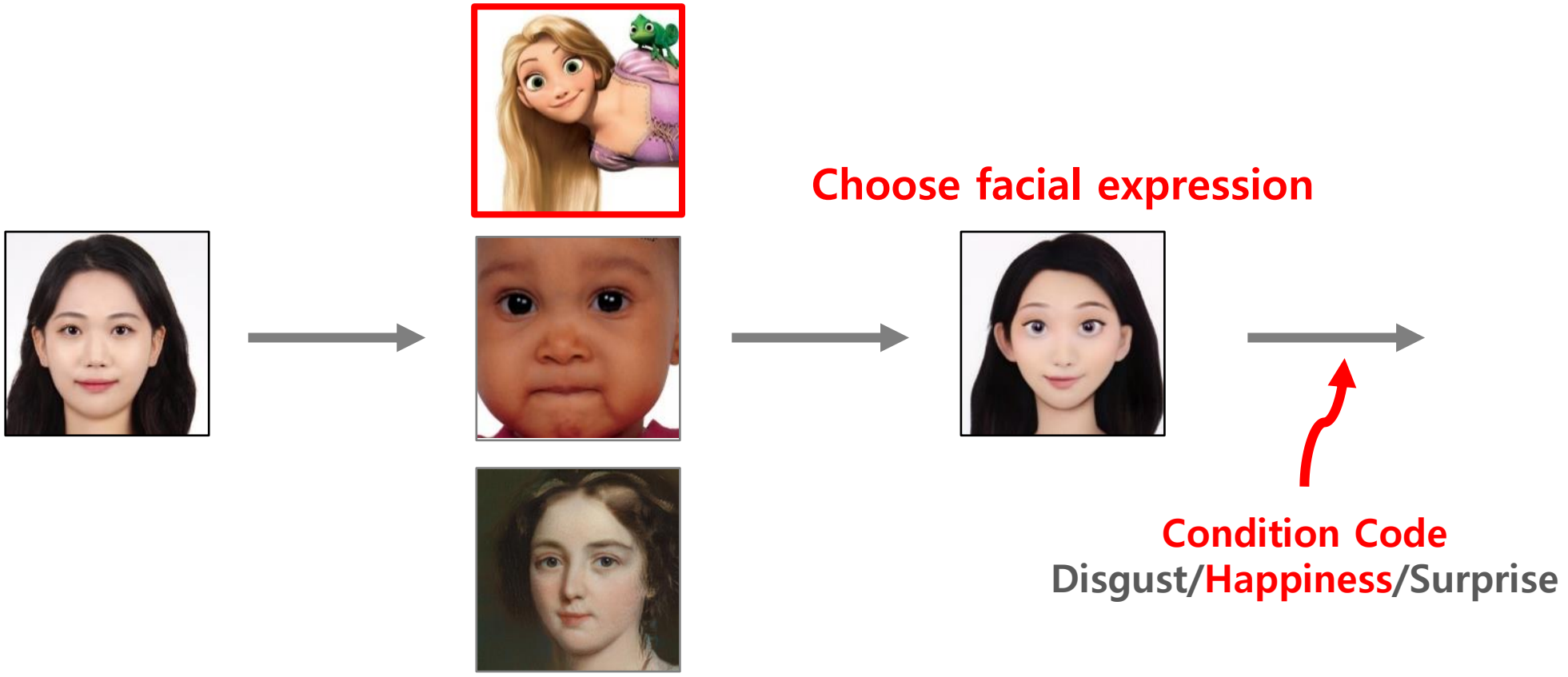
01. Tobigticon

step2. emoticon style 선택

Choose an emoticon style

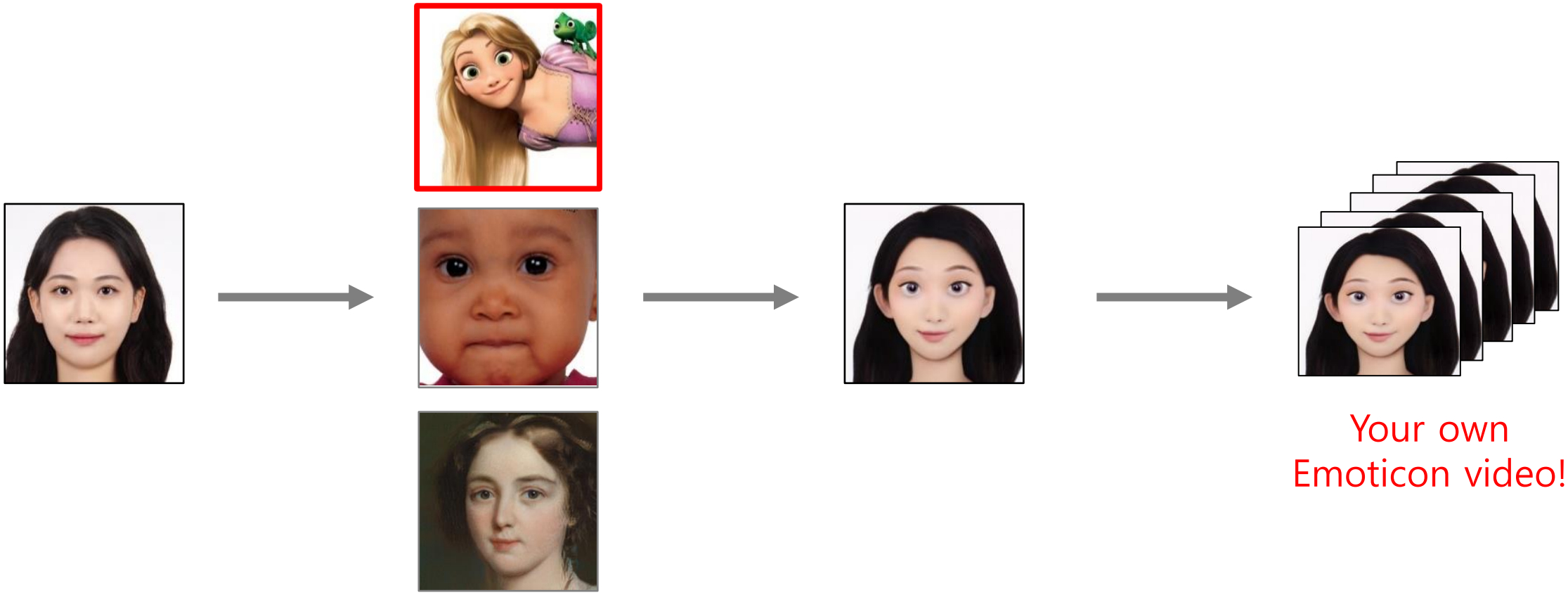
01. Tobigticon

step3. facial expression 선택



01. Tobigticon

step4. 이모티콘 비디오 생성



Your own
Emoticon video!

EmoGE'T – **Emoticon Ge**neration by **T**obig's

02

Mode

02. Model

Model

: 주어진 이미지에서 스타일을 입힌 비디오를 생성하는 모델 구조를 2단계로 제안

face2emoticon

1. Conditional StyleGAN2
2. Network Blending
3. GAN Inversion

이미지에 스타일 적용

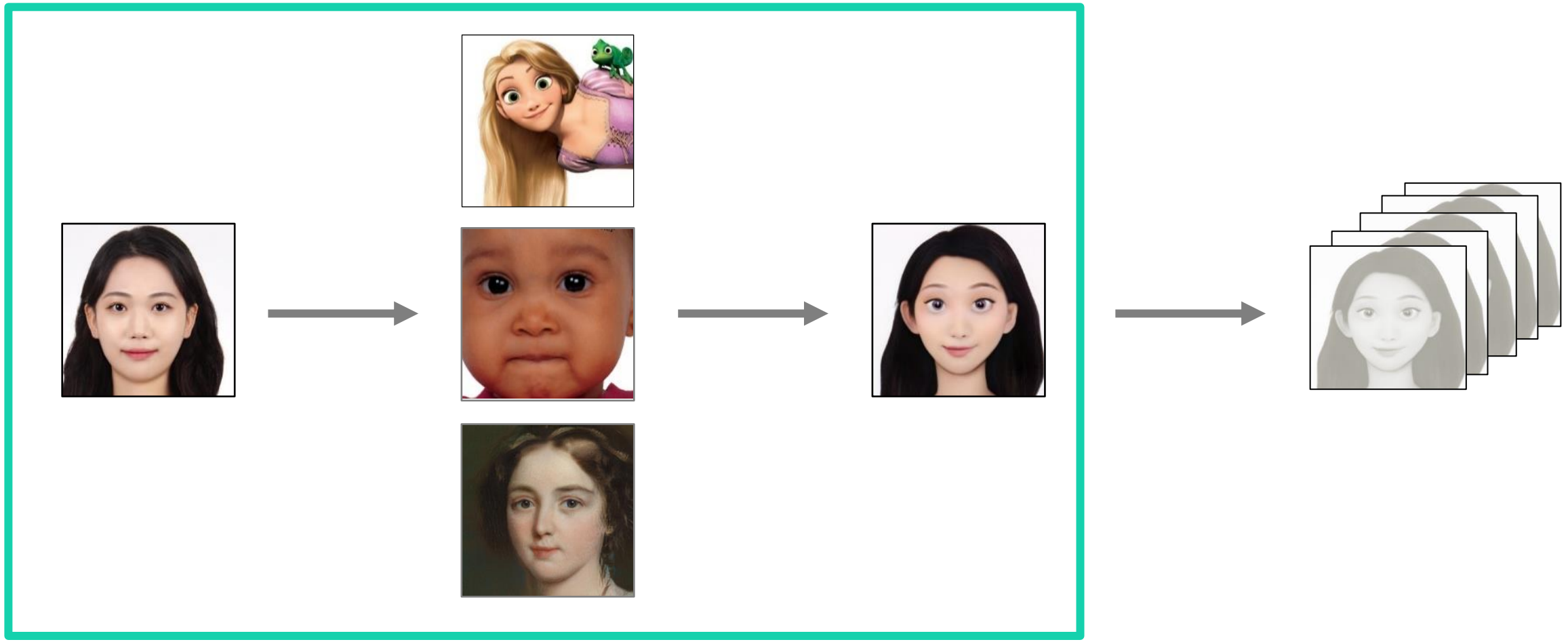
Video generation

1. Imge2Video Model
2. Landmark Generation

이미지에서 비디오 생성

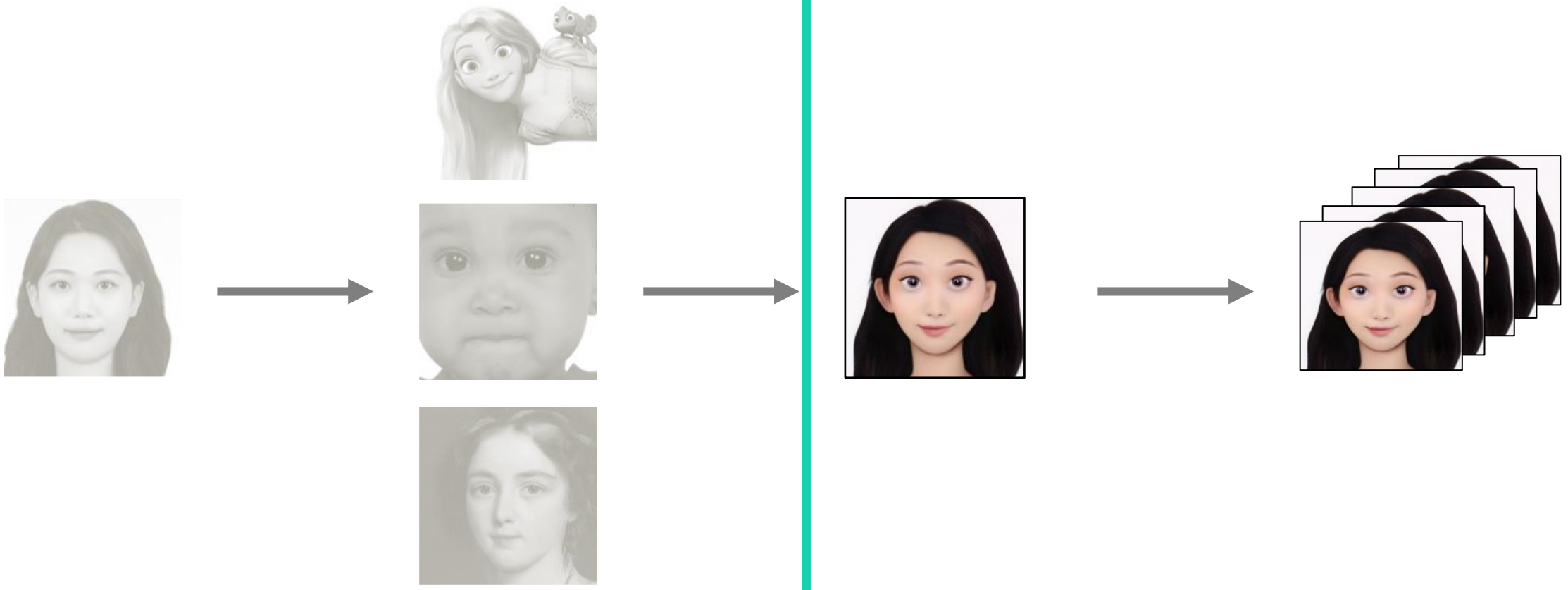
02. Model

1. face2emoticon



02. Model

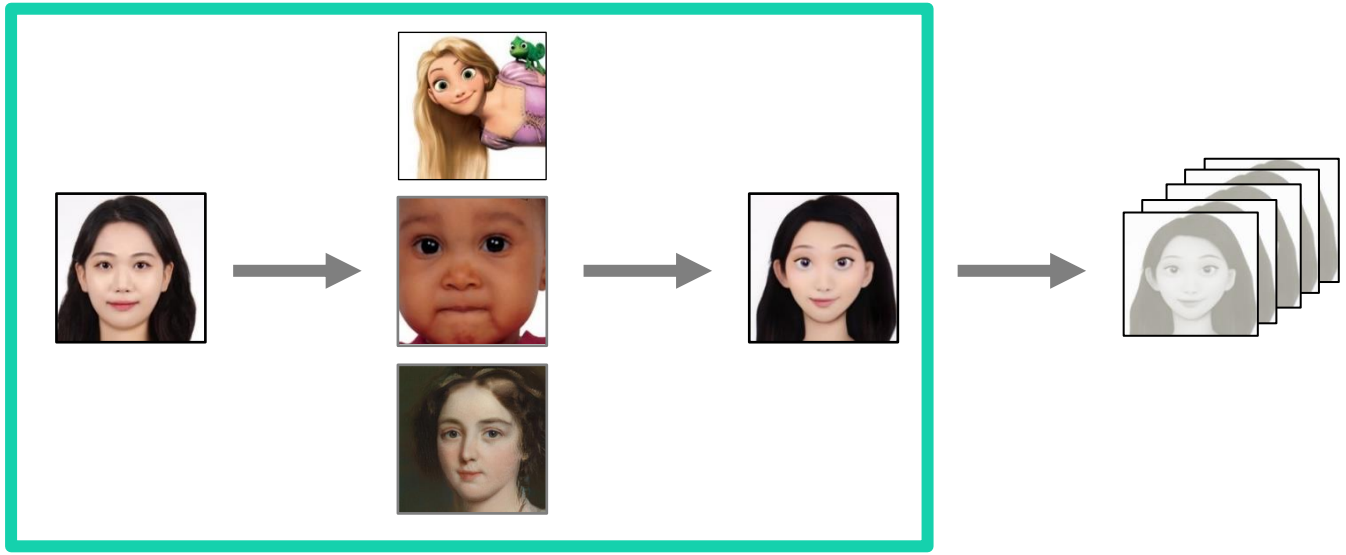
2. Video generation



02. Model – face2emoticon

< Face to emoticon >

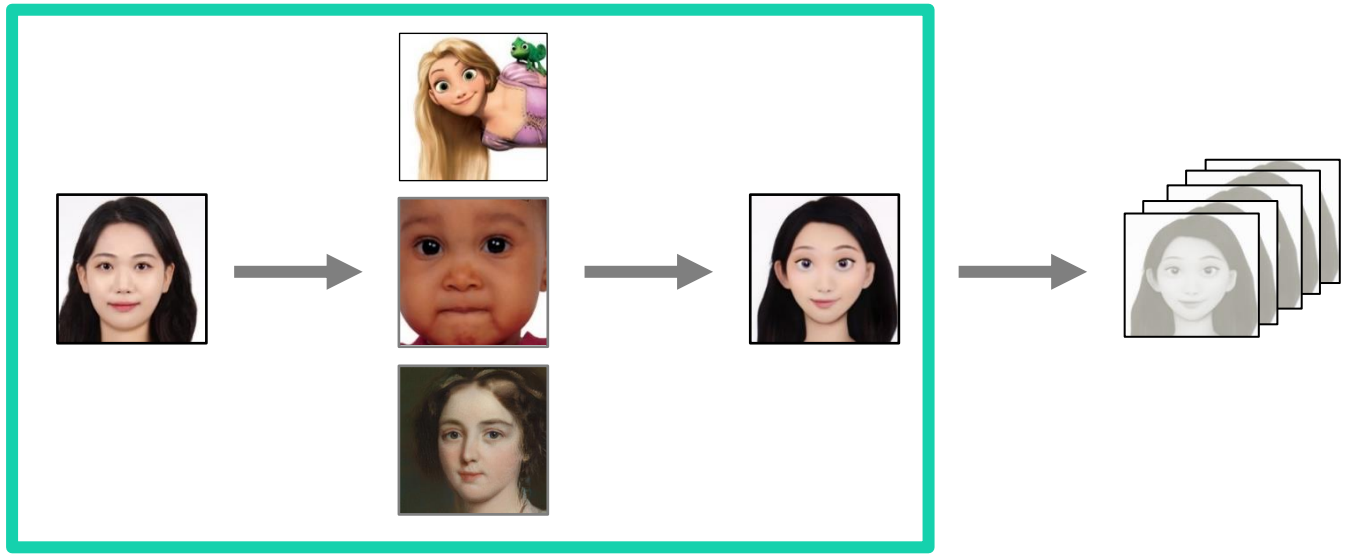
1. Conditional StyleGAN2
2. Network Blending
3. GAN Inversion



02. Model – face2emoticon

< Face to emoticon >

1. Conditional StyleGAN2
2. Network Blending
3. GAN Inversion



02. Model – face2emoticon

1. Conditional styleGAN2

: 여러 style을 주고자 CGAN 구조를 차용하여 stylegan2를 변형

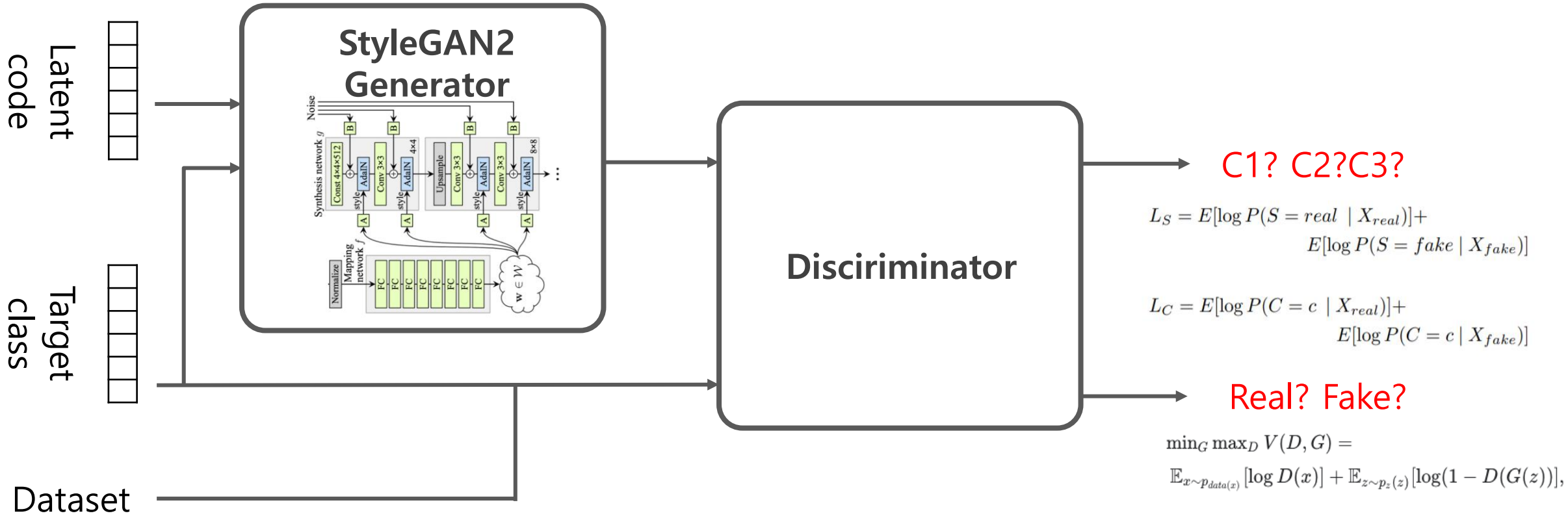
: Discriminator에 auxiliary를 추가하여 생성된 이미지의 class를 구분하게 함

-> 즉, class를 embedding 하는 과정을 추가하여 class vector를 만들어 줌

Input으로 latent vector와 class vector를 함께 넣음으로써 클래스별 분포를 학습하게 함!

02. Model – face2emoticon

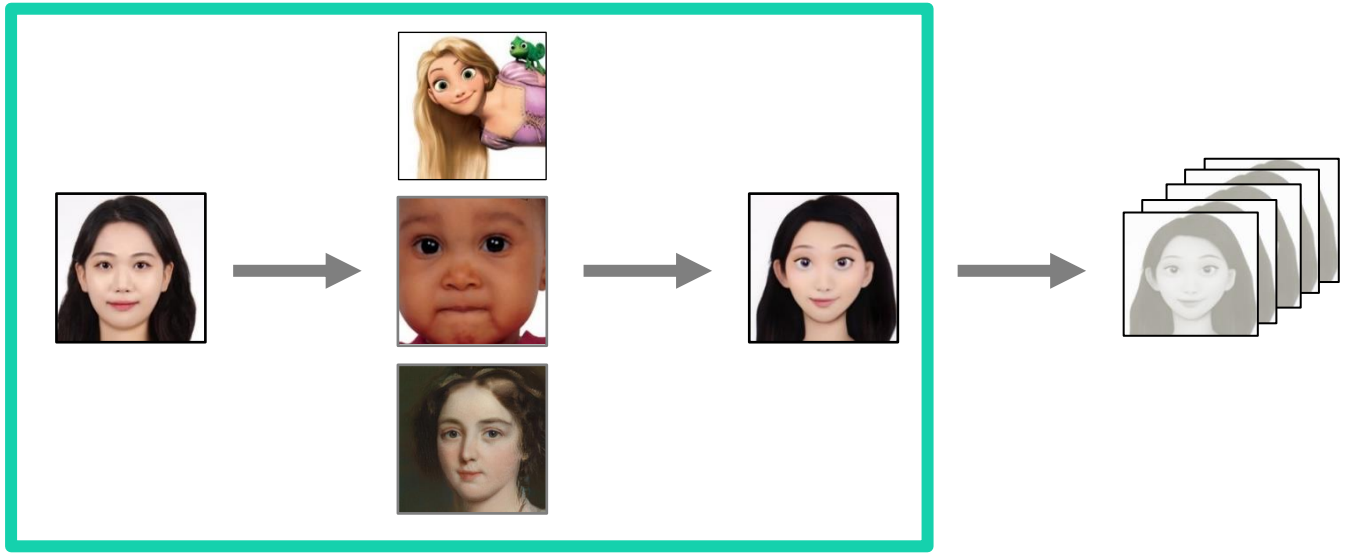
1. Conditional styleGAN2



02. Model – face2emoticon

< Face to emoticon >

1. Conditional StyleGAN2
2. Network Blending
3. GAN Inversion



02. Model – face2emoticon

2. Network blending

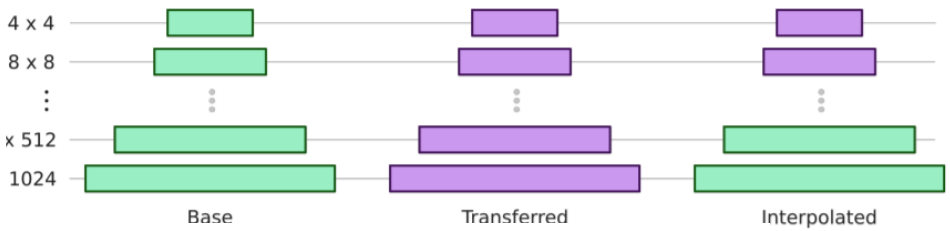
: 파라미터를 섞기 위해 interpolation을 사용한 것

: 앞쪽에서는 물체의 엣지를 잡아내고, 뒤쪽에서는 구체적인 특징(눈 생김새, 코, 귀 등)을 살려 냄

StyleGAN2 + Network blending + Distillation

Base model → Flickr-Faces-HQ Dataset (FFHQ)

Transfer model → x Disney



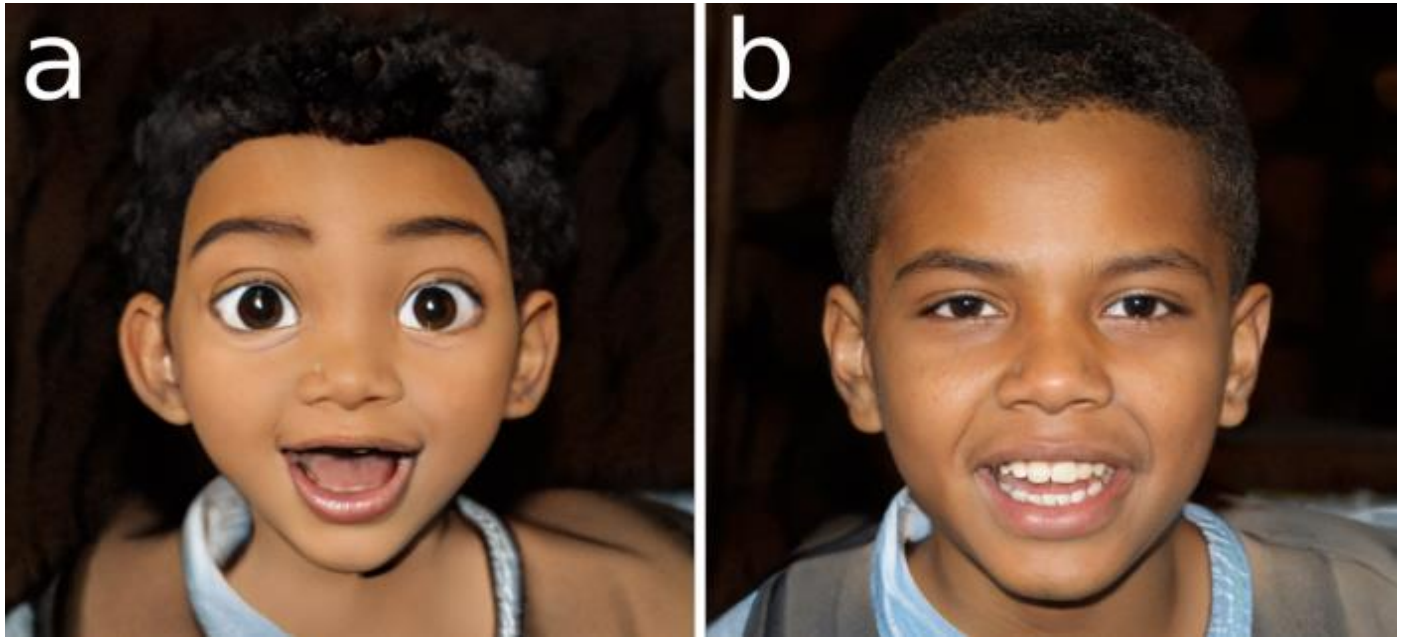
$$\theta_{interp} = \alpha_1\theta_1 + \alpha_2\theta_2 + \ldots + \alpha_N\theta_N$$

02. Model – face2emoticon

2. Network blending

: 네트워크 블렌딩 기법 사용시 전반적으로 사람의 특징을 살리면서,

동시에 애니메이션의 특징인 과장된 표정과 디테일한 특징(눈, 코, 입 등)을 가진 이미지 생성이 가능!



02. Model – face2emoticon

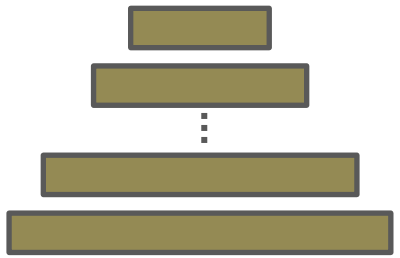
2. Network blending

0) DATASET

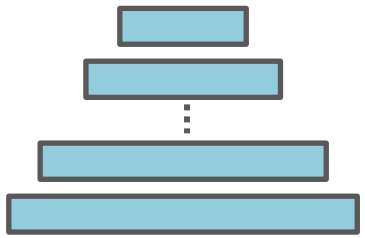
- BASE: FFHQ (일반인 얼굴 사진 데이터셋)
- TRANSFERRED: DISNEY, BABY, PAINTINGS **(이모티콘화를 위한 데이터셋 – 애니메이션, 아기 얼굴 사진, 초상화)**

1) FFHQ로 **pretraining**을 하여 $\boldsymbol{M_{base}}$를 생성 → 2) $\boldsymbol{M_{base}}$를 **finetuning** 하여 $\boldsymbol{M_{transferred}}$를 생성

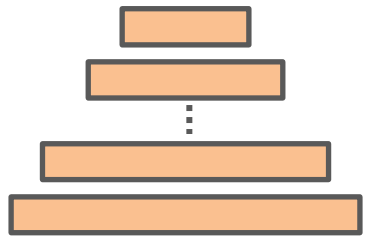
FFHQ



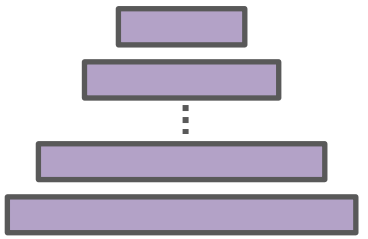
M_{base}



$M_{transferred_1}$



$M_{transferred_2}$



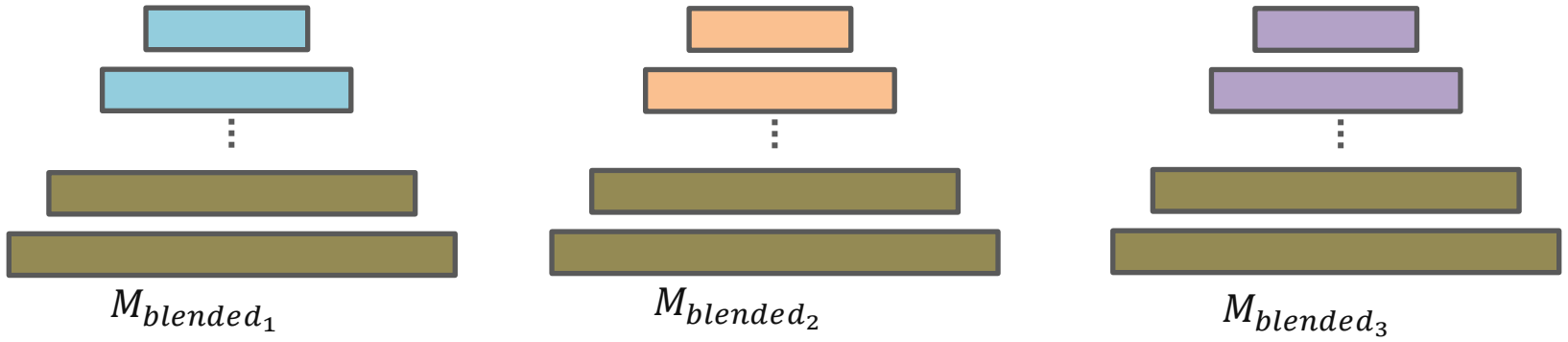
$M_{transferred_3}$

02. Model – face2emoticon

2. Network blending

→ 3) 각 네트워크 블렌딩하여 각 파라미터 별 데이터를 생성 -> Conditional StlyeGAN2의 data로 사용

(* 데이터셋마다 블렌딩 기준 resolution은 다를 수 있음)



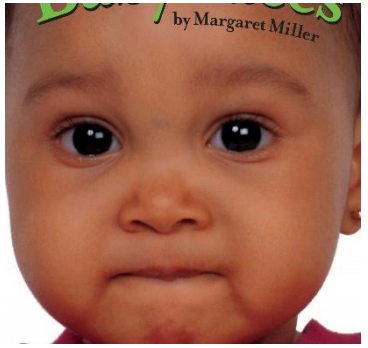
02. Model – face2emoticon

2. Network blending – Result

FFHQ

X

DISNEY



BABY

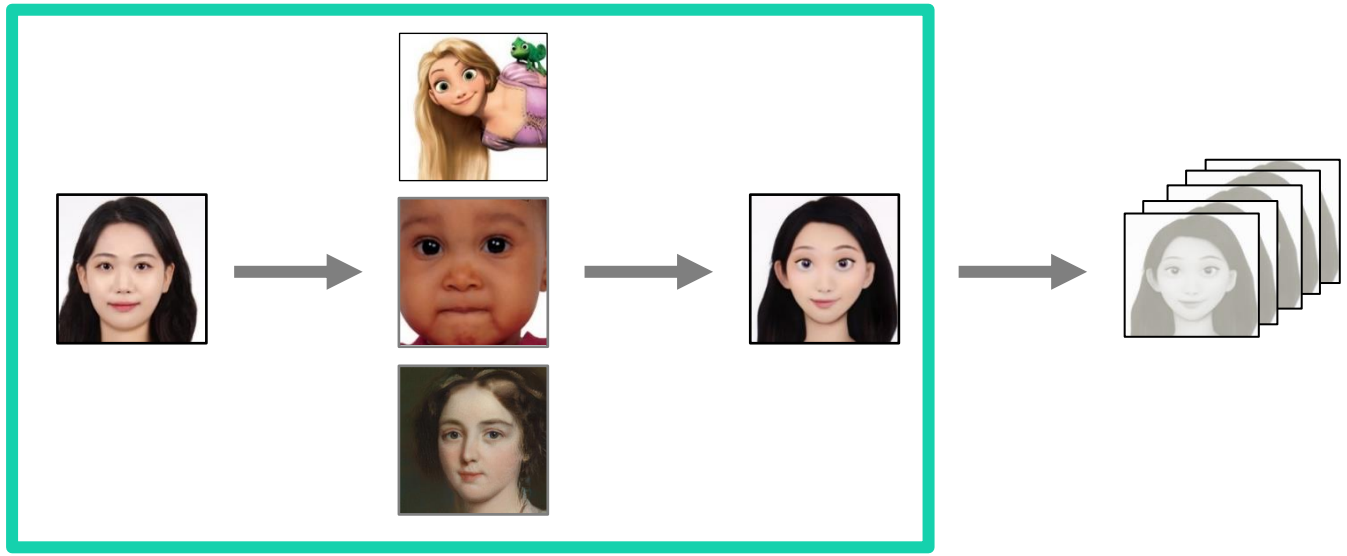
Paintings

<DATASET>
- BASE: FFHQ
- TRANSFERRED
: DISNEY, BABY, PAINTINGS

02. Model – face2emoticon

< Face to emoticon >

1. Conditional StyleGAN2
2. Network Blending
3. GAN Inversion



02. Model – face2emoticon

3. GAN Inversion

: GAN은 학습 전에 가정한 분포에서 랜덤하게 latent code z를 input으로 함

따라서 원하는 이미지로의 변형이 어려움

-> input image를 복원할 수 있는 가장 적절한 latent code를 찾아내는 **" Inversion"** 을 적용하자

: GAN Inversion

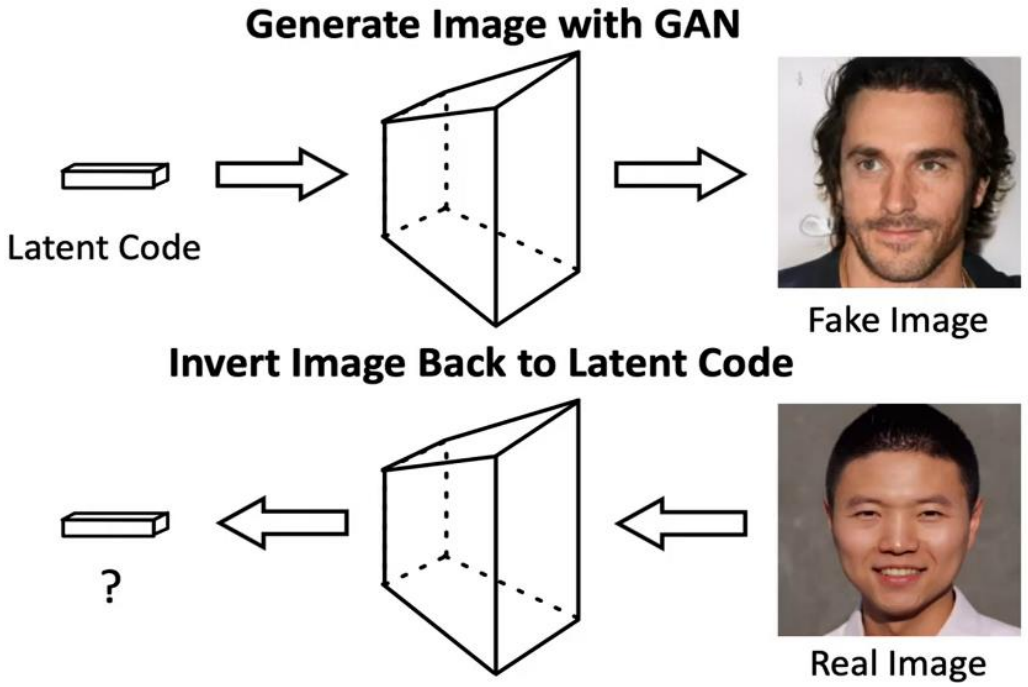
내가 원하는 이미지를 생성하기 위해 latent code를 어떻게 잡으면 좋을지를 찾아 냄

02. Model – face2emoticon

3. GAN inversion

: MSE loss를 사용하는 **Optimization based** 접근 방식 선택

: 실제 이미지(x)와 모델이 만들어내는 이미지 (G(z)) 간 차이를 가장 작게 하는 z를 찾음



- **Optimization based (we used)**
- Learning based

- They either learn **an extra encoder** beyond the GAN [23, 36, 4] or directly **optimize the latent code** for an individual image [22,24,8]

$$\mathcal{L}_L(E,G) \equiv \mathbb{E}_{\mathbf{z}}[d_L(\mathbf{z}, E(G(\mathbf{z})))]$$
$$\mathcal{L}_R(E,G) \equiv \mathbb{E}_{\mathbf{z}}[d_I(\mathbf{x}, G(E(\mathbf{x}))) \mid \mathbf{x} = G(\mathbf{z})]$$
$$E_{(b)} = \arg\min_E (\mathcal{L}_L(E,G) + \lambda \mathcal{L}_R(E,G))$$

$$\theta_P^* = \arg\min_{\theta_P} \sum_n \mathcal{L}(G(P(x_n^R; \theta_P)), x_n^R),$$

$$\mathbf{z}^* = \arg\min_{\mathbf{z}} d_I(\mathbf{x}, G(\mathbf{z}))$$

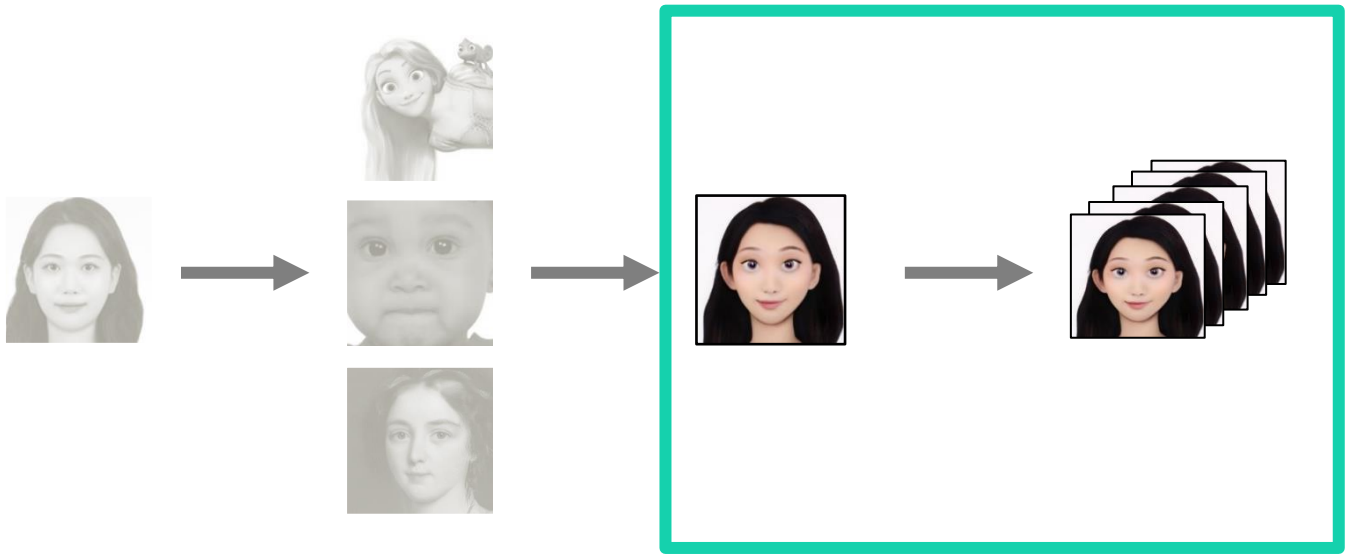
$$\min_{z'} ||\phi(z) - \phi(z')||_2^2.$$

02. Model – Video Generation

< Video Generation>

1. Imge2Video Model

2. Landmark Generation

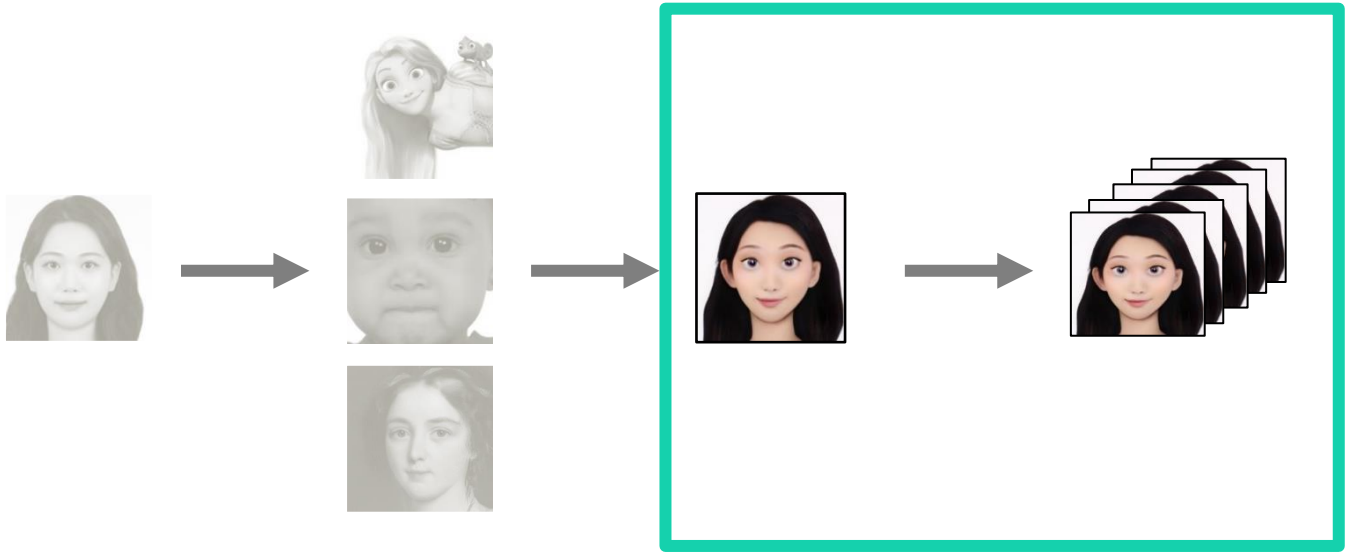


02. Model – Video Generation

< Video Generation>

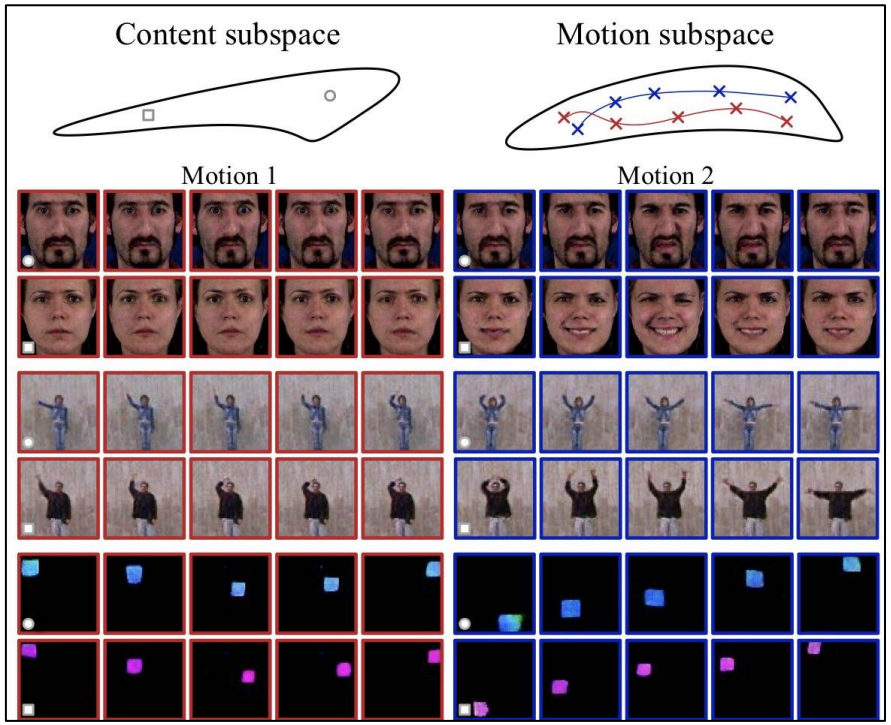
1. Imge2Video Model

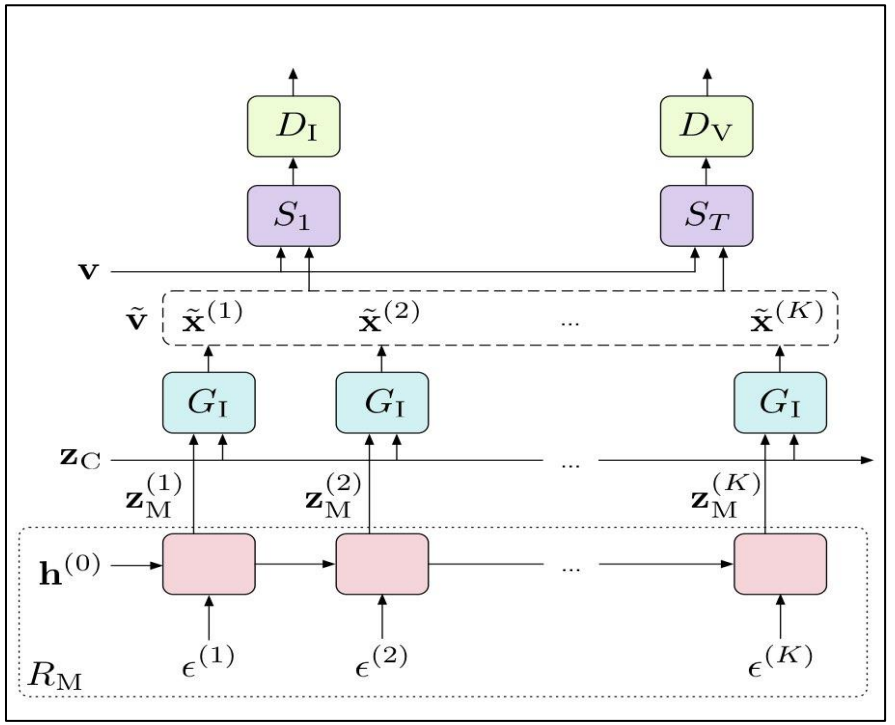
2. Landmark Generation



02. Model – Video Generation

1. Image2Video - MoCoGAN

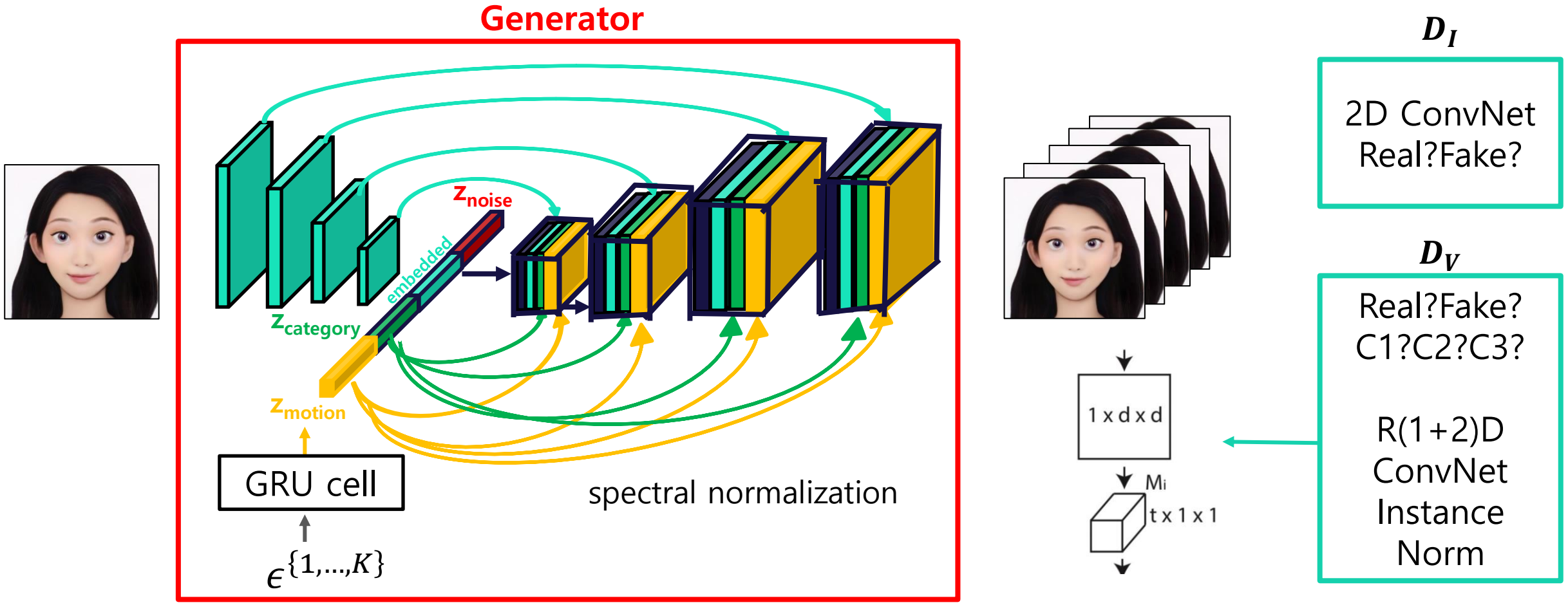




Video = Content + Motion

02. Model – Video Generation

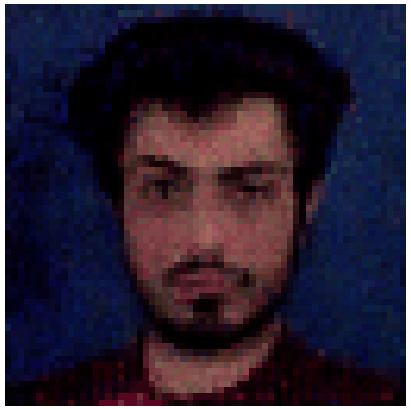
1. Image2Video Model – Our model



02. Model – Video Generation

1. Image2Video Model – Our model






문제점

1. 낮은 화질
2. 데이터셋 편향

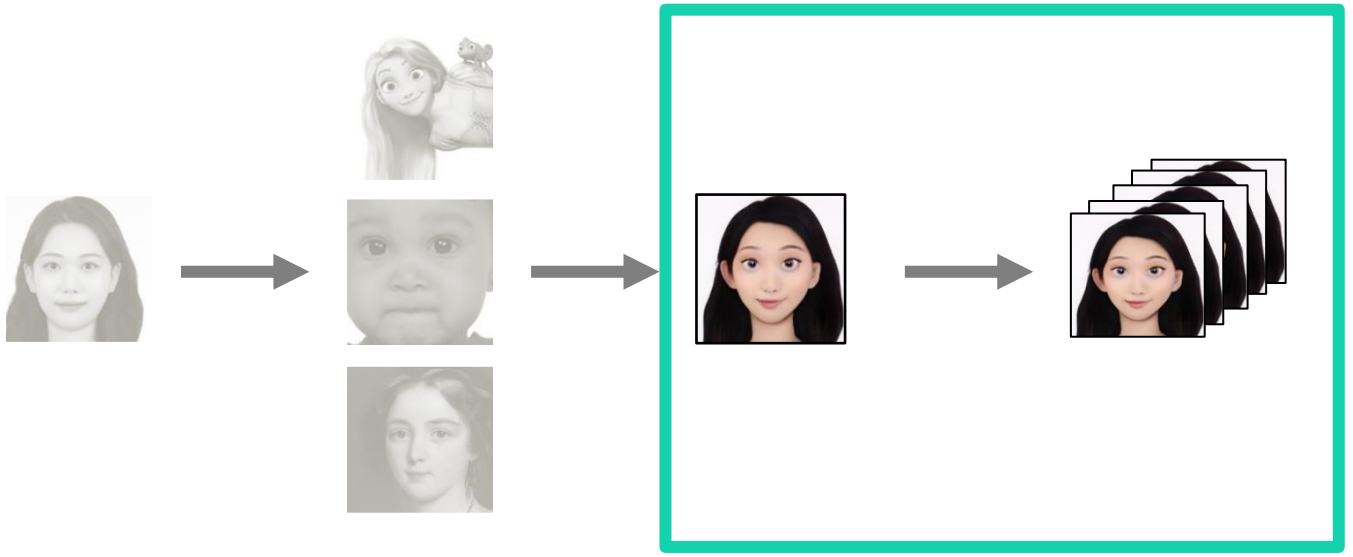
Landmark Generation !

02. Model – Video Generation

< Video Generation>

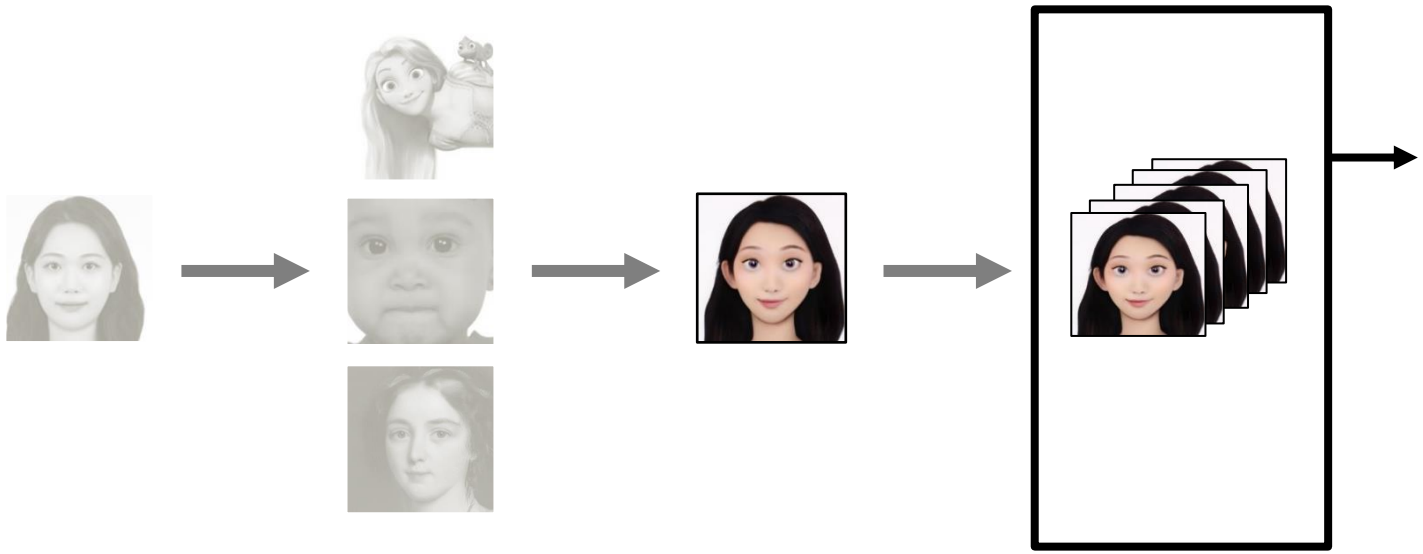
1. Imge2Video Model

2. Landmark Generation



02. Model – Video Generation

2. Landmark Generation



이모티콘의 Facial expression 동영상 GT X

이모티콘의 얼굴 ≠ 실제 사람의 얼굴

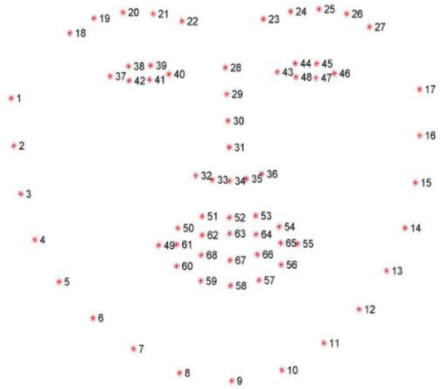
⟶이모티콘의 얼굴 landmark ≅ 실제 사람의 얼굴 landmark

02. Model – Video Generation

2. Landmark Generation

l_t: t시점의 $landmark$
l_0^i: 0시점의 $landmark$가 $class\ embedding\ vector$와 함께 $linear\ layer$를 통과하여 각 시점에 맞게 변형된 형태

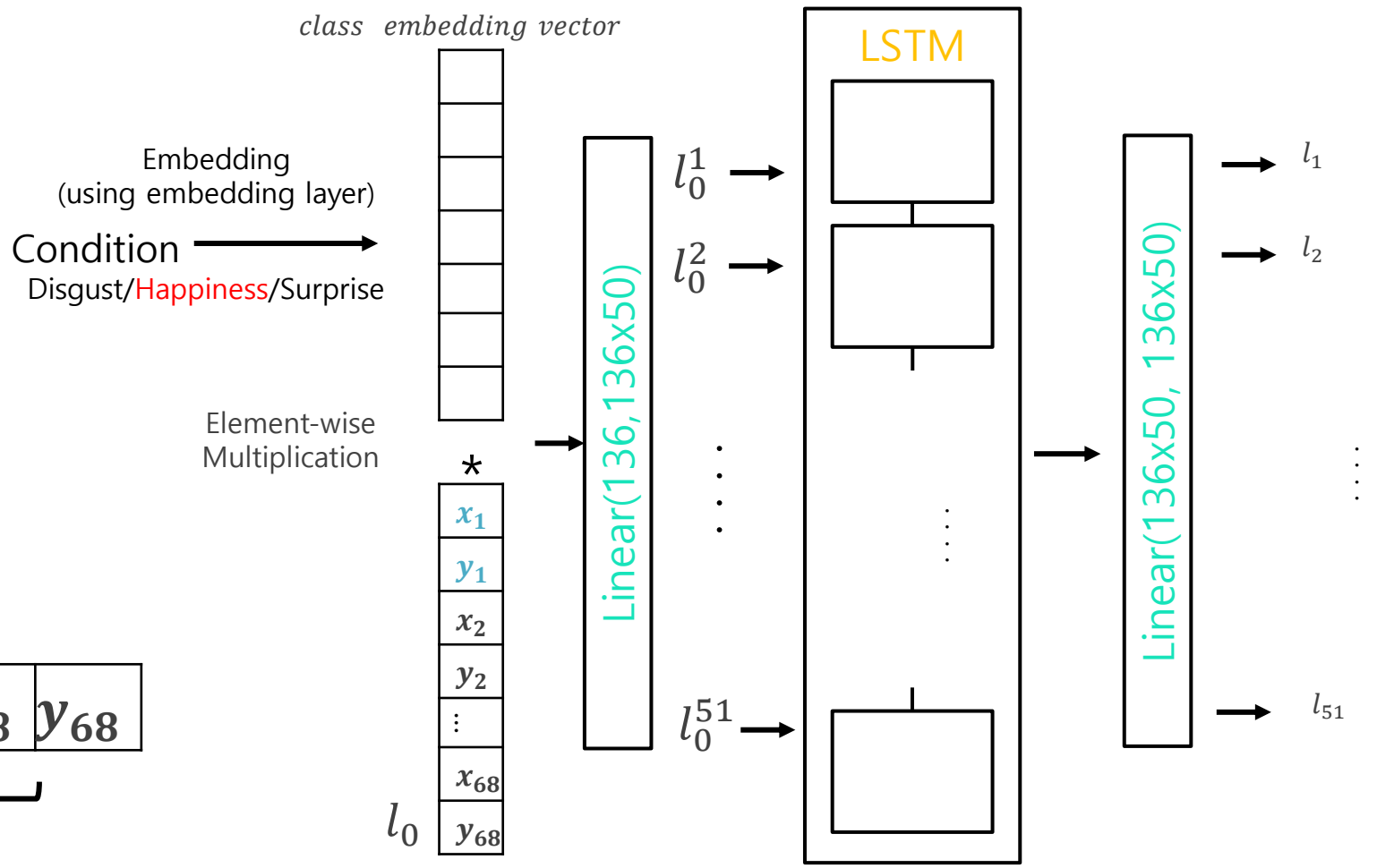
Facial Landmark



1번 landmark의 좌표

x_1	y_1	x_2	y_2	…	x_{68}	y_{68}

(1,136)



02. Model – Video Generation

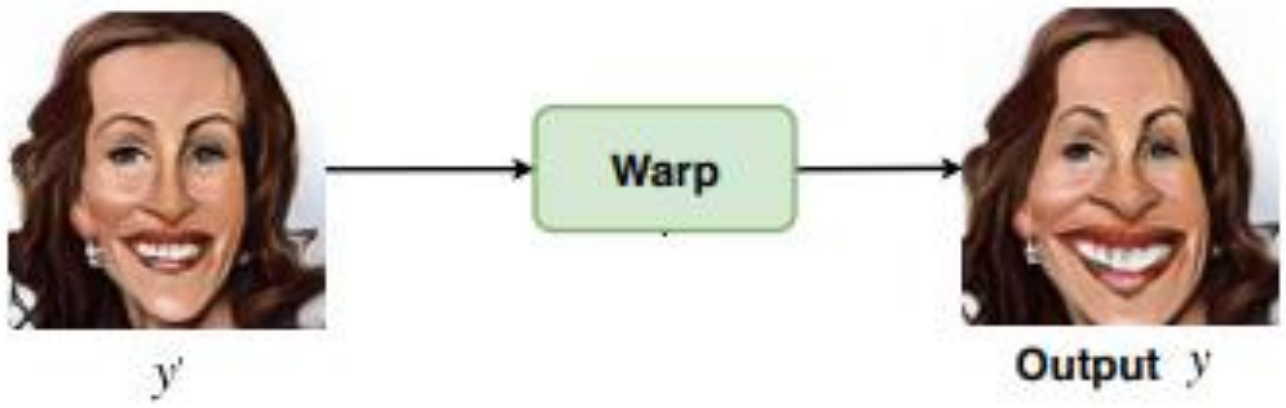
2. Landmark Generation- warp

Image warp

: 인공위성으로부터 전송된 영상이 렌즈의 변형, 신호의 왜곡 등의 이유로 일그러질 때 이를 복구하는 데 사용

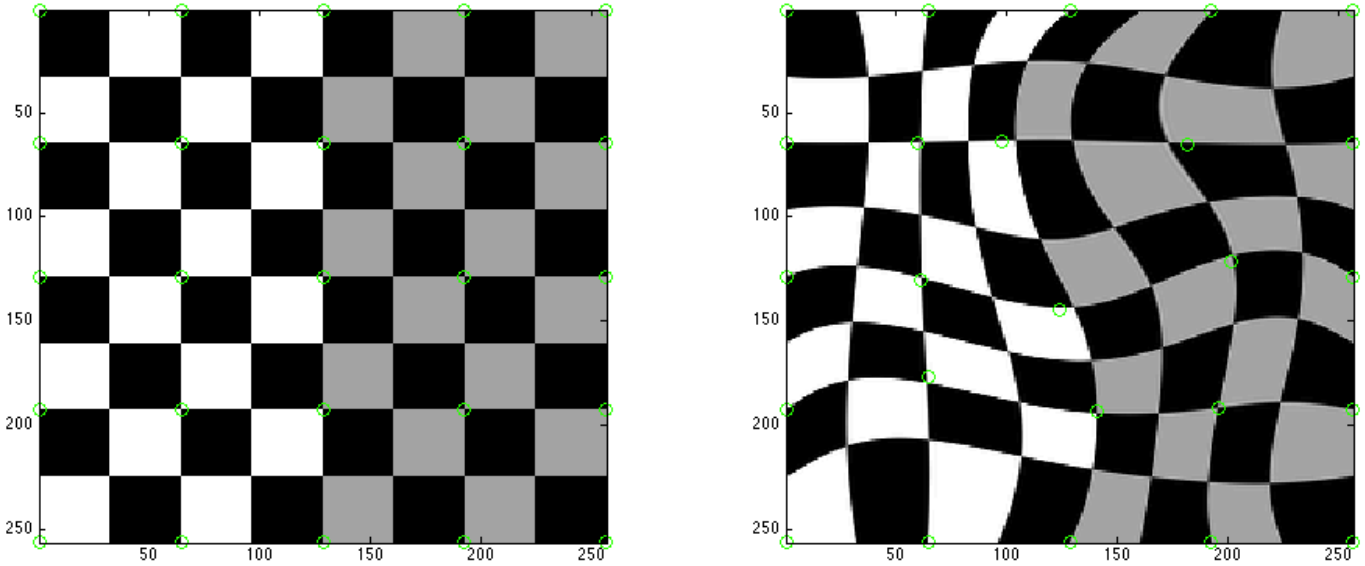
: (x, y) 에서 (x', y')로 픽셀을 대응시키는 작업

: 즉, 영상을 찌그러.



02. Model – Video Generation

2. Landmark Generation



(x, y) 좌표의 Pixel of image I
$I(x, y)$

→

미리 지정된 위치변환 함수
$U_x(x, y)$: X방향 위치변환 함수 $U_y(x, y)$:Y방향 위치변환 함수

→

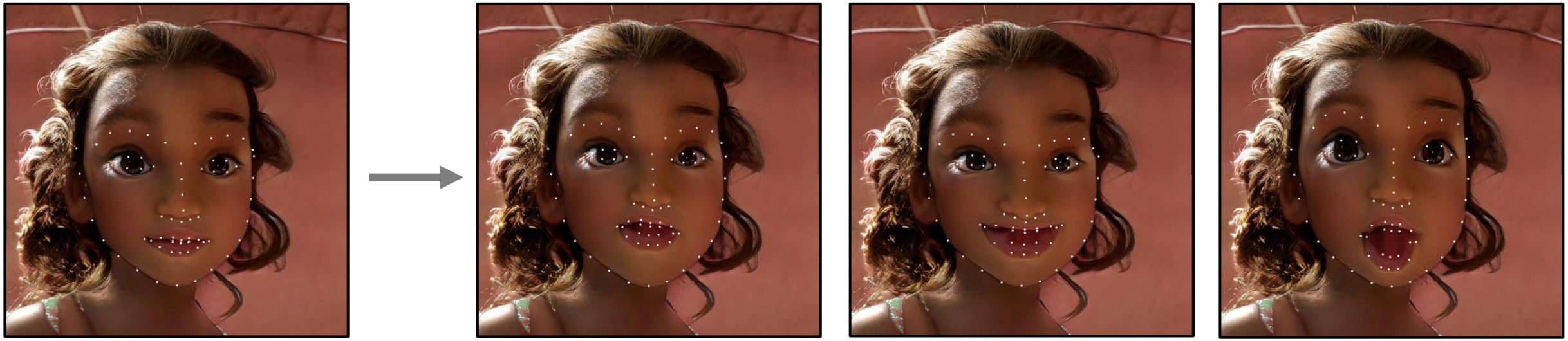
Warping된 image I의 pixel 좌표
$I\left(U_x(x, y), U_y(x, y)\right) = \mathbf{I}(\mathbf{x}', \mathbf{y}')$

02. Model – Video Generation

2. Landmark Generation – warp

: 원본 이미지의 얼굴 랜드마크에서 새롭게 생성된 얼굴 랜드마크로의 위치변환

-> 자연스러운 얼굴 변형



Original Disgust Happiness Surprise

EmoGE'T – **Emoticon Ge**neration by **T**obig's

03

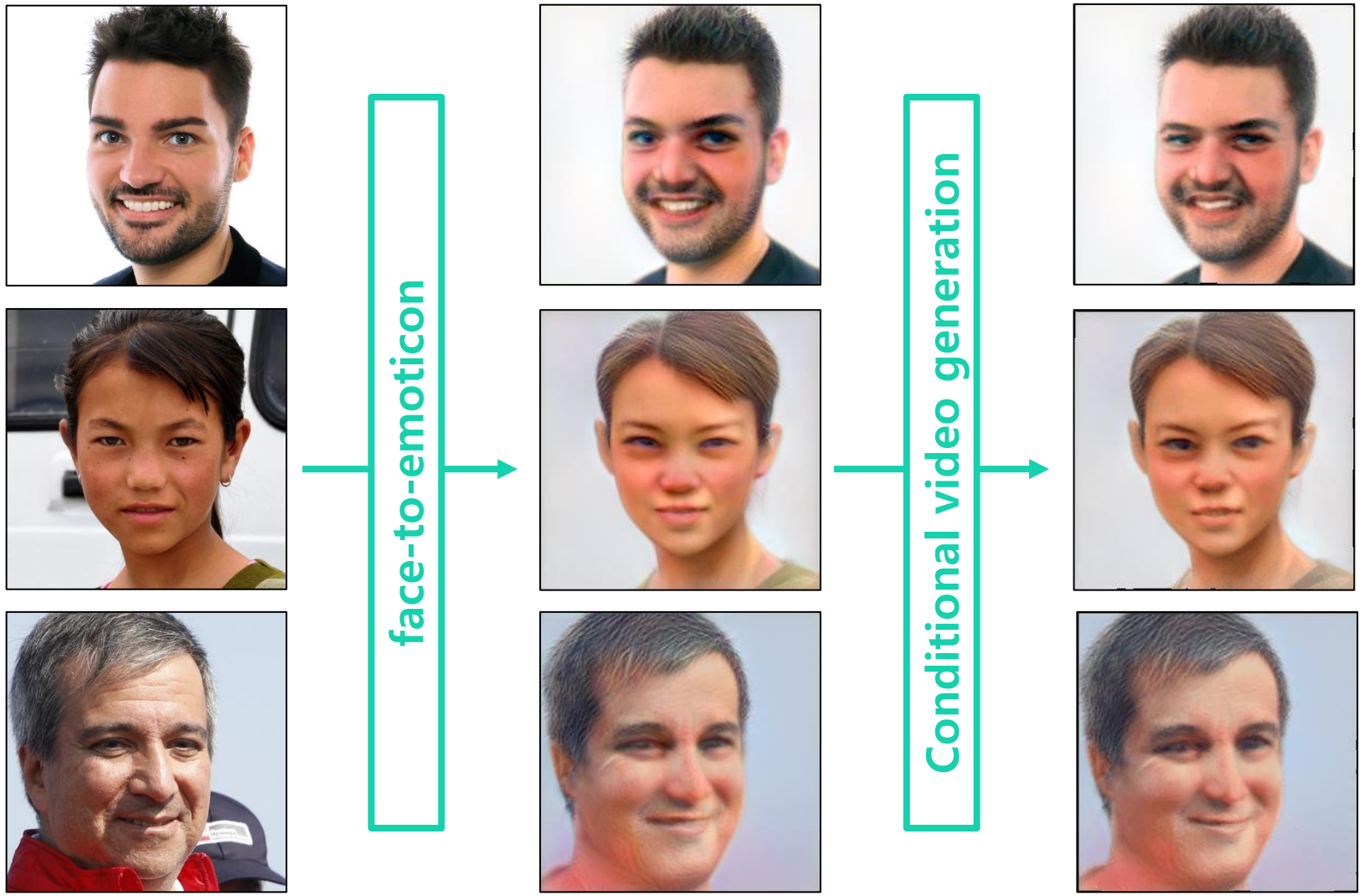
Result

03. Result

1. Style - Disney

Condition

- Disgust
- Surprise
- Happiness

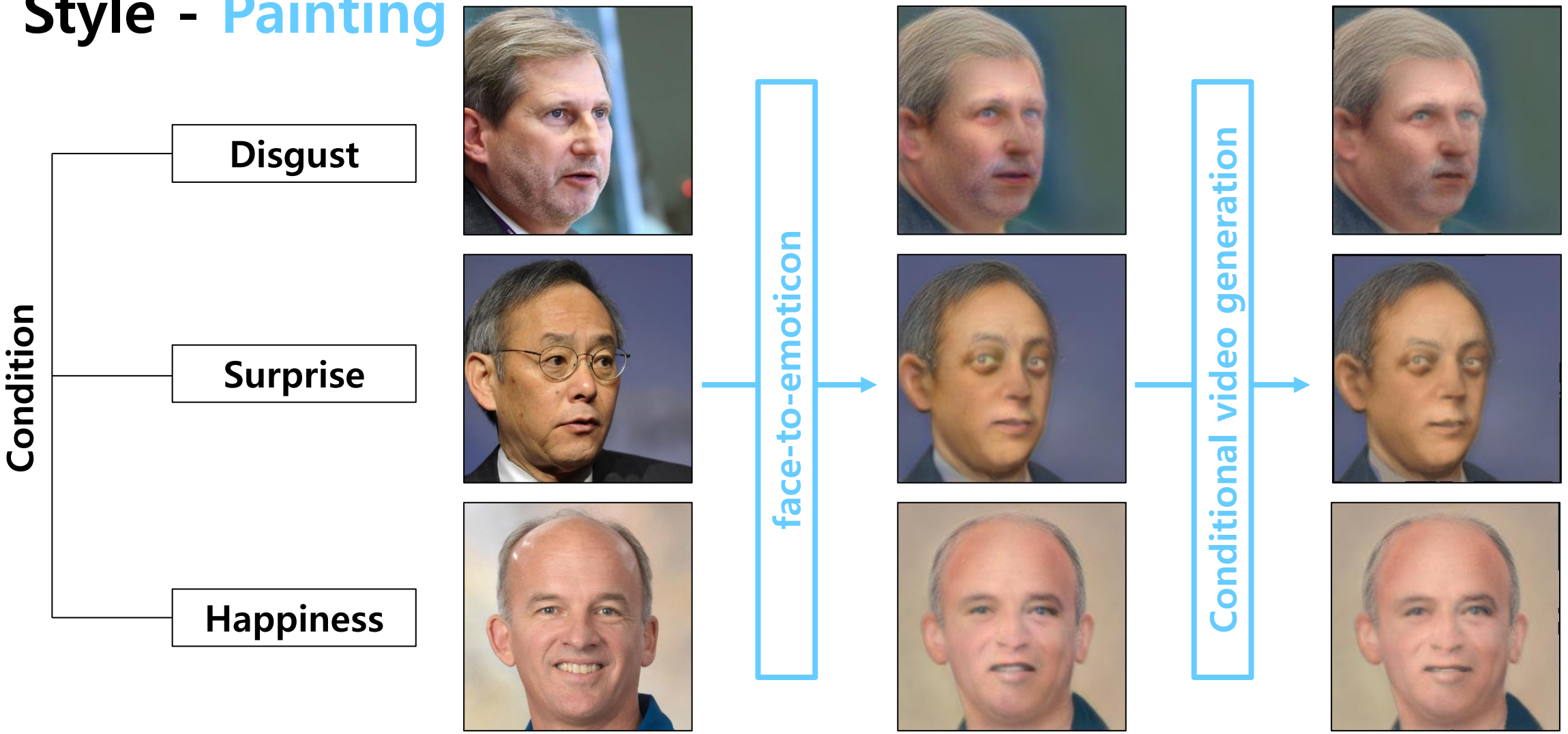


03. Result

2. Style - Baby

03. Result

3. Style - Painting

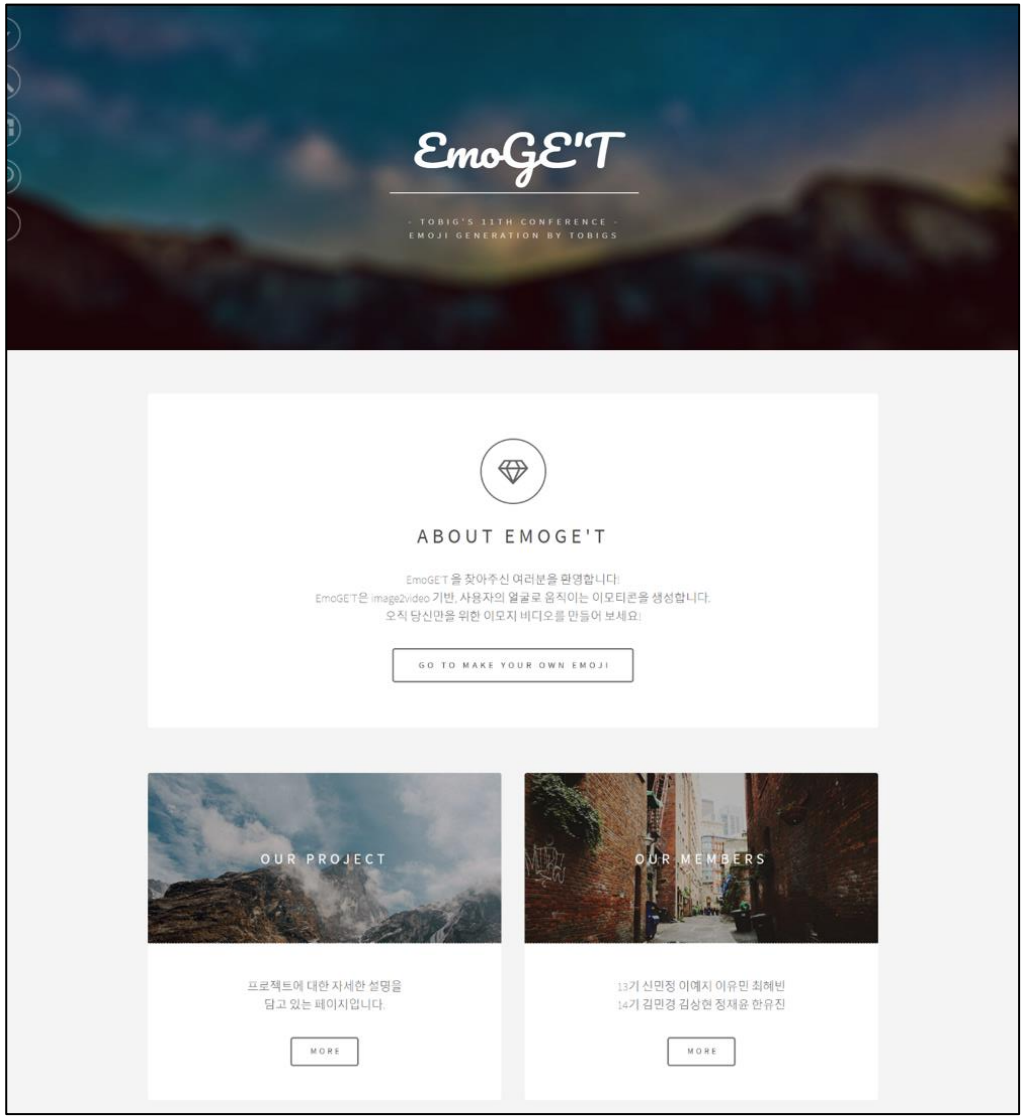


EmoGE'T – **Emoticon Ge**neration by **T**obig's

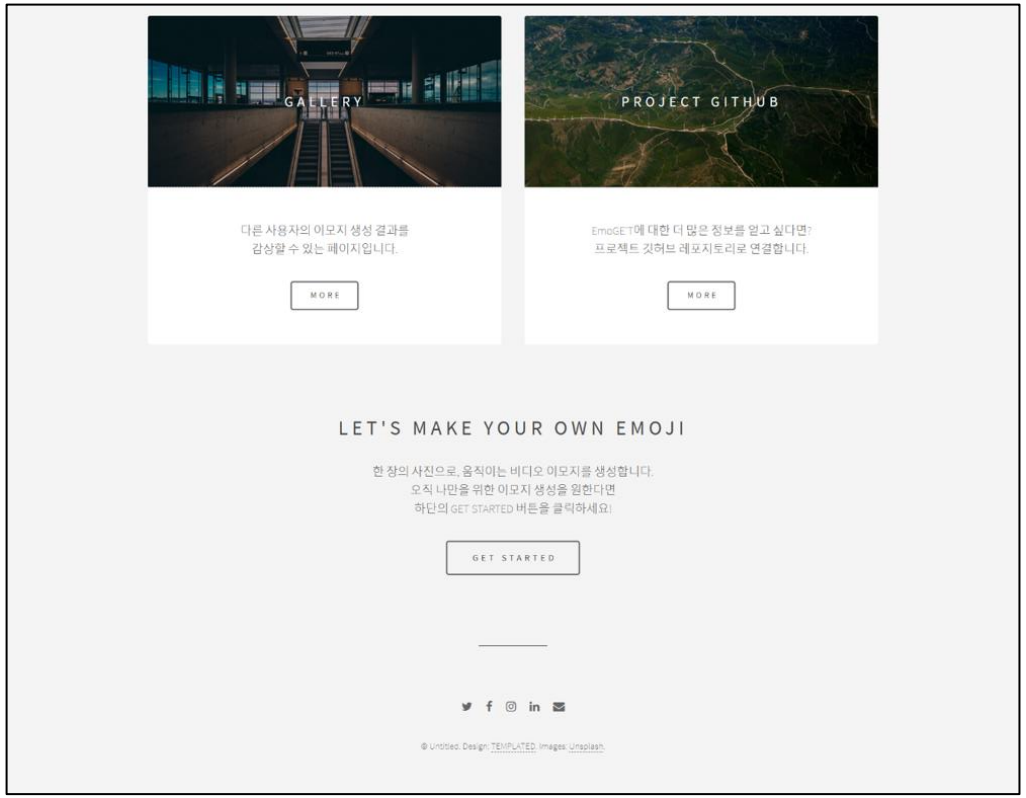
04

Demo

04. Demo



Let's go to our webpage!



EmoGE'T – **Emoticon Ge**neration by **T**obig's

05

Conclusion

05. Conclusion

1. 의의

- 오직 한 장의 이미지만으로 비디오(gif) 생성
- 이미지에 원하는 스타일 적용이 가능하게 함

2. 한계점 및 개선 방향

- 얼굴 특정부위가 찌그러지는 경우 발생
- 모델이 무거움
- 추후 성능 개선시 보다 자연스러운 비디오 생성 기대

Members

13기 신민정

GAN장공장공장장

13기 이예지

15기 도망GAN

13기 이유민

오류 발생시 GAN톡 주세용

13기 최혜빈

GAN질GAN질 (❁´◡`❁)

Members

14기 김민경

GAN신히 끝냈다

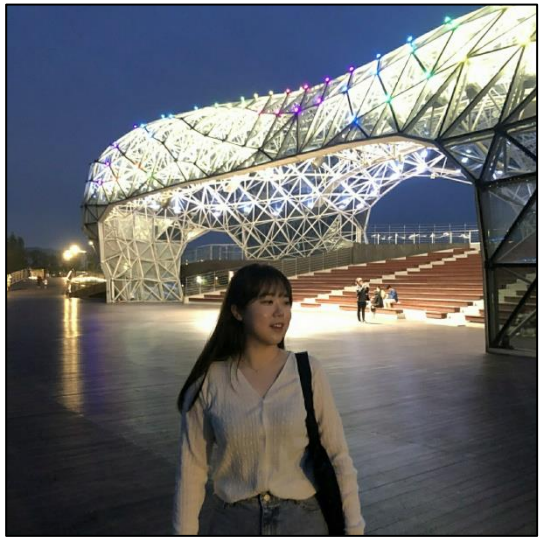
휴 (ಥ _ ಥ)

14기 김상현

파리GAN 남자

14기 정재윤

GAN 떨어지는 컨퍼



14기 한유진

GAN린이의 성장일기

끝~?

Reference

도움주신 분

12기 김수아님

참고자료 및 참고문헌

CariGANs https://arxiv.org/abs/1811.00222

Rosinality, stylegan2-pytorch, 2019, https://github.com/rosinality/stylegan2-pytorch
PieraRiccio, stylegan2-pytorch, 2019, https://github.com/PieraRiccio/stylegan2-pytorch
justinpinkney, toonify, 2020, https://github.com/justinpinkney/tonify
marsbroshok, face-replace, 2016, https://github.com/marsbroshok/face-replace
sergeytulyakov, mocogan, 2017, https://github.com/sergeytulyakov/mocogan
Yaohui Wang, Piotr Bilinski, Francois Bremond, Antitza Dantcheva. ImaGINator: Conditional Spatio-Temporal GAN for Video Generation. 2019.

https://paeton.tistory.com/entry/%EC%9D%B4%EB%AF%B8%EC%A7%80-%EC%9B%8C%ED%95%91-image-Warping

https://m.blog.naver.com/PostView.nhn?blogId=infoefficien&logNo=220820421779&proxyReferer=https:%2F%2Fwww.google.com%2F

https://www.instiz.net/name/33250372

https://www.pngegg.com/ko/png-eigow

EmoGE'T – **Emoticon Ge**neration by **T**obig's

감사합니다