ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

CVL Computer Vision Lab

Sliced Wasserstein Generative Models for Image Generation and Enhancement

Zhiwu Huang

Computer Vision Lab @ ETH Zurich

Presented Papers

- **Sliced Wasserstein Generative Models**
  Jiqing Wu*, **Zhiwu Huang***, Dinesh Acharya, Wen Li, Janine Thoma, Danda Pani Paudel, Luc Van Gool. *(*indicates equal contributions)*
  Computer Vision and Pattern Recognition (CVPR), 2019

- **Divide-and-Conquer Adversarial Learning for High-resolution Photo Enhancement**
  Zhiwu Huang, Danda Pani Paudel, Guanju Li, Jiqing Wu, Radu Timofte, Luc Van Gool.
  Submitted to The International Conference on Learning Representations (ICLR), 2020

Real or Fake?


A


B


C


D


E


F


G


H

✔ ✖ Real or Fake?

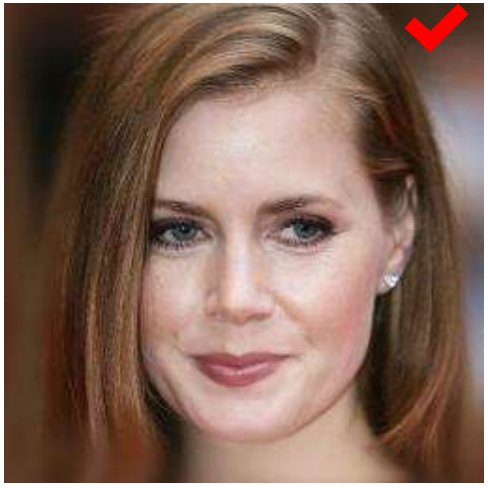
A

B

C

D

E



F

G

H

Hints for Real or Fake?

Text is uninterpretable

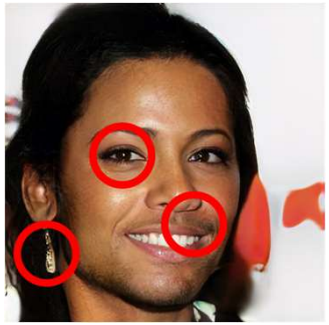
Background is surreal

Asymmetry

Weird teeth

Messy hair

Non-stereotypical gender



Images from *https://medium.com/@kcimc/how-to-recognize-fake-ai-generated-images-4d1f6f9a2842*

Real or Fake?

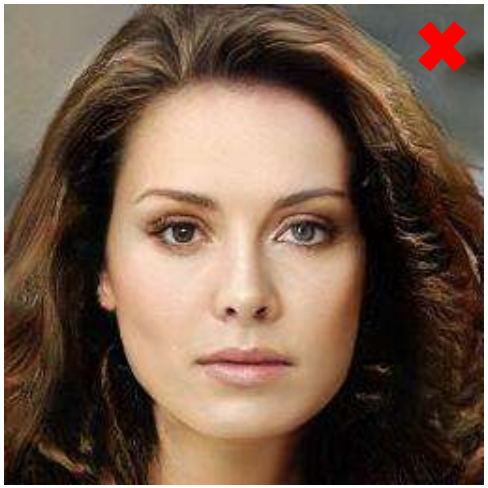
A

B

C

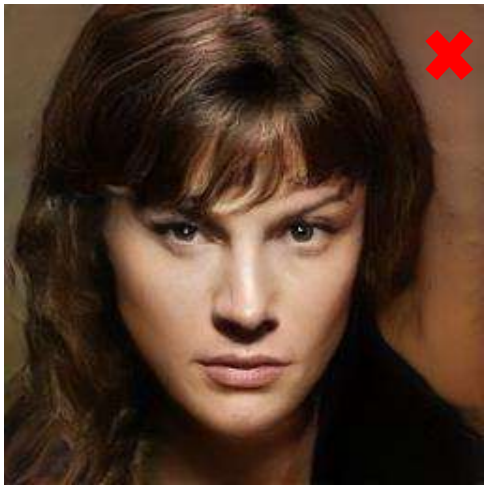
D

E



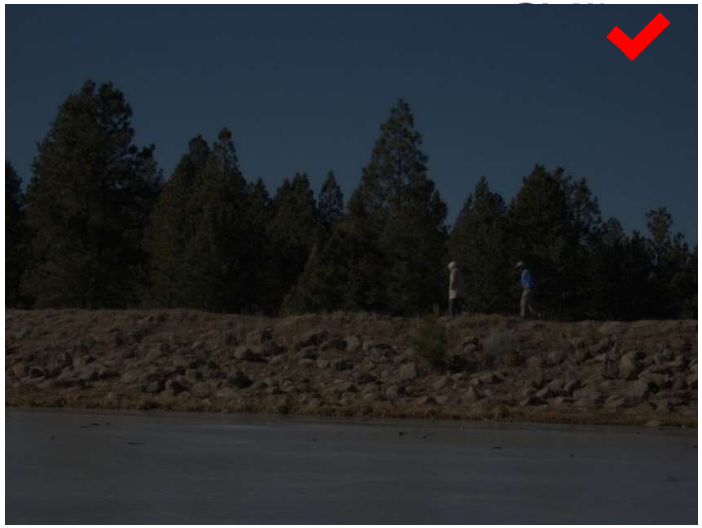
F

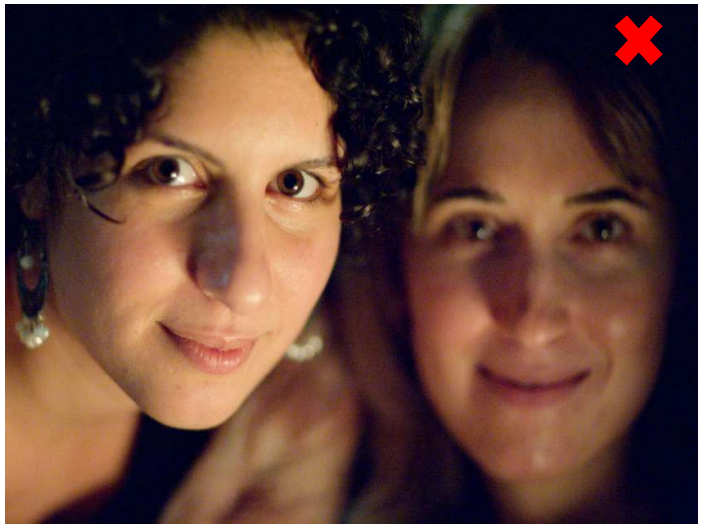
G



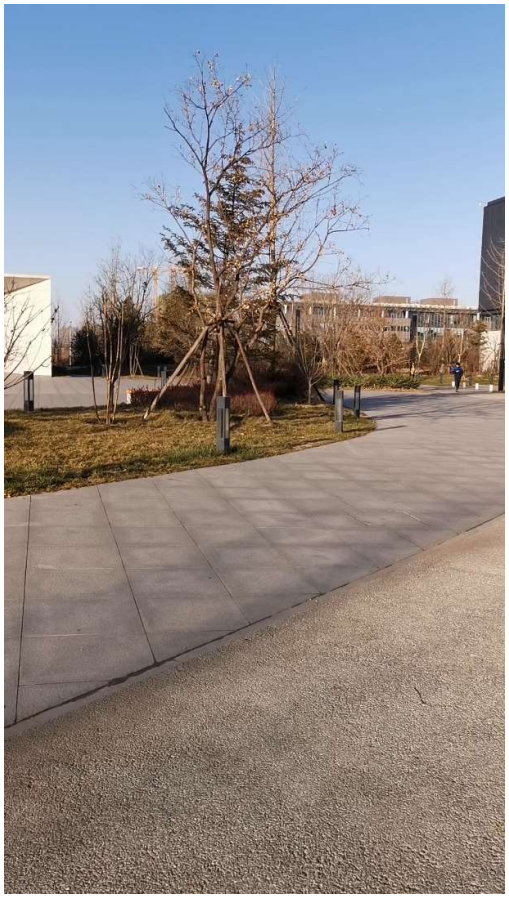
H

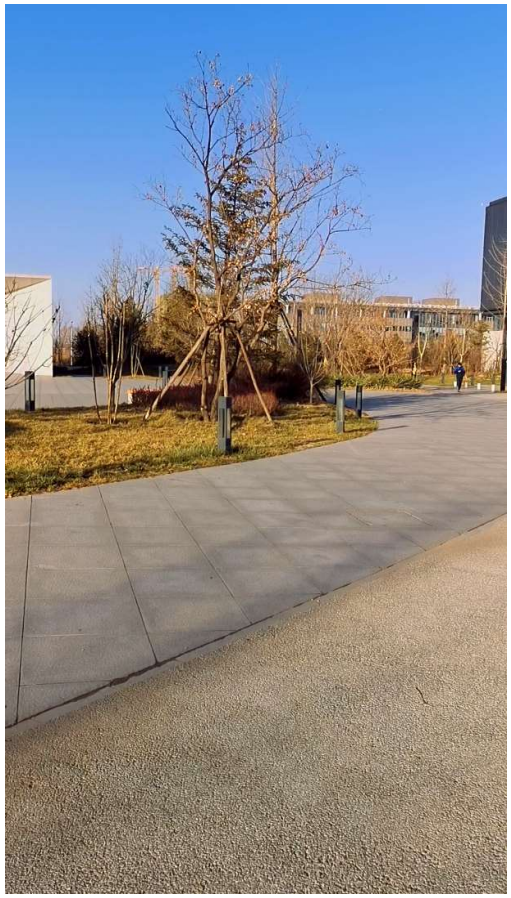
Before or After?

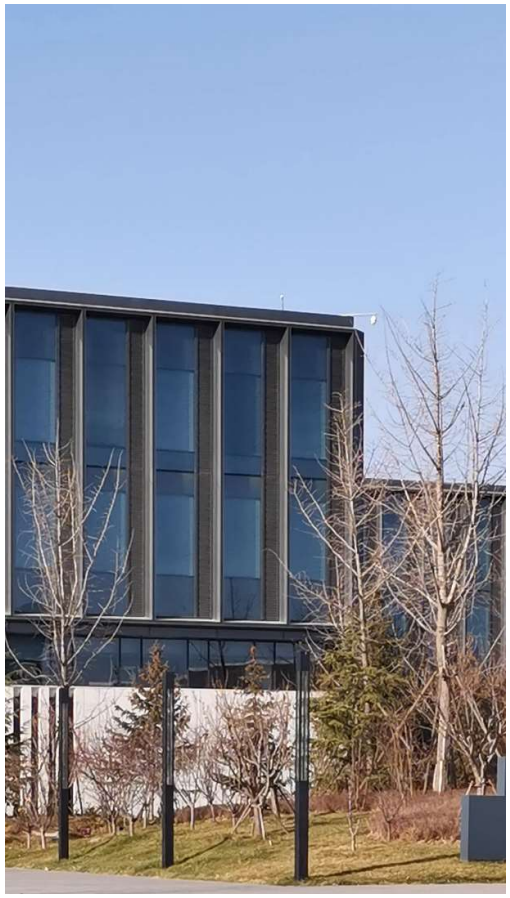


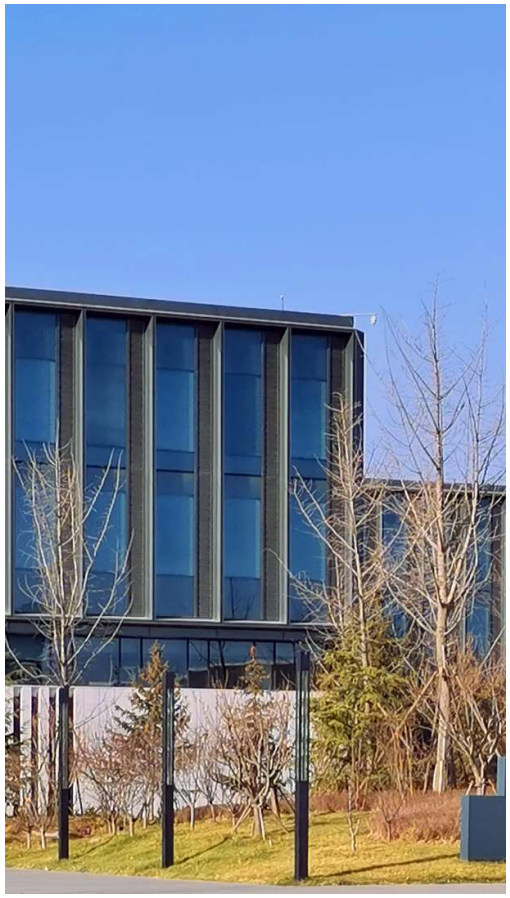


Before or After?





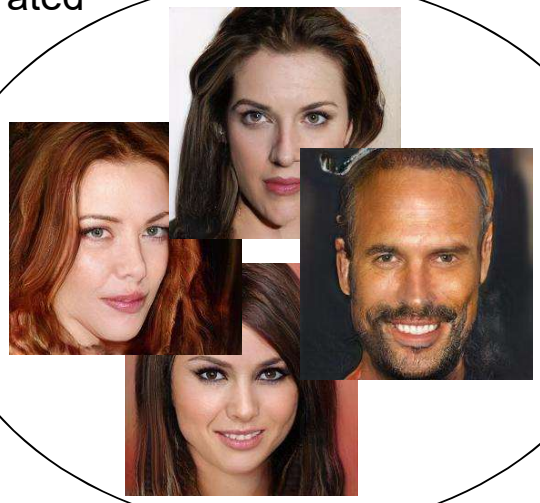




Img Generation

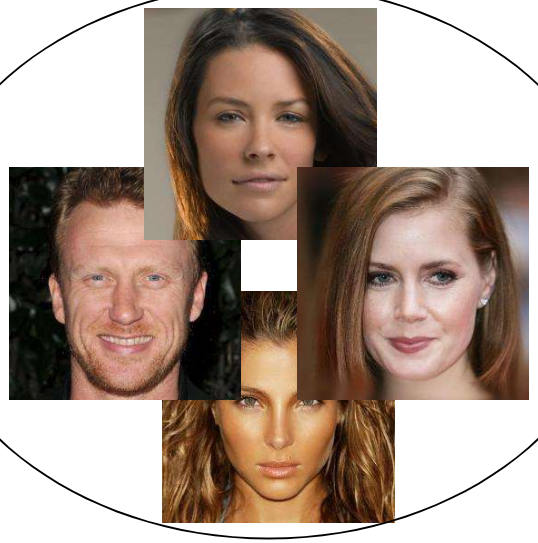
Img Enhancement

Generated



Enhanced

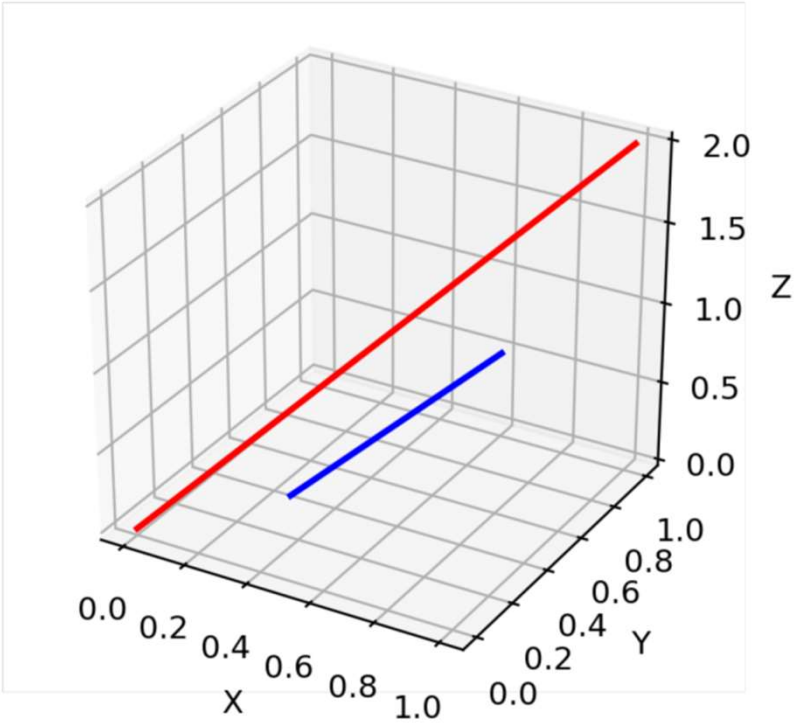
Real



Target

Distribution Distance Measurement ???

From KL and JS Divergence to Wasserstein Distance



$$D_{KL}(P\|Q) = \sum_{x=0, y\sim U(0,1)} 1 \cdot \log \frac{1}{0} = +\infty$$

$$D_{KL}(Q\|P) = \sum_{x=\theta, y\sim U(0,1)} 1 \cdot \log \frac{1}{0} = +\infty$$

$$D_{JS}(P,Q) = \frac{1}{2}\left(\sum_{x=0, y\sim U(0,1)} 1 \cdot \log \frac{1}{1/2} + \sum_{x=0, y\sim U(0,1)} 1 \cdot \log \frac{1}{1/2}\right) = \log 2$$

$$W(P,Q) = |\theta|$$ ✔

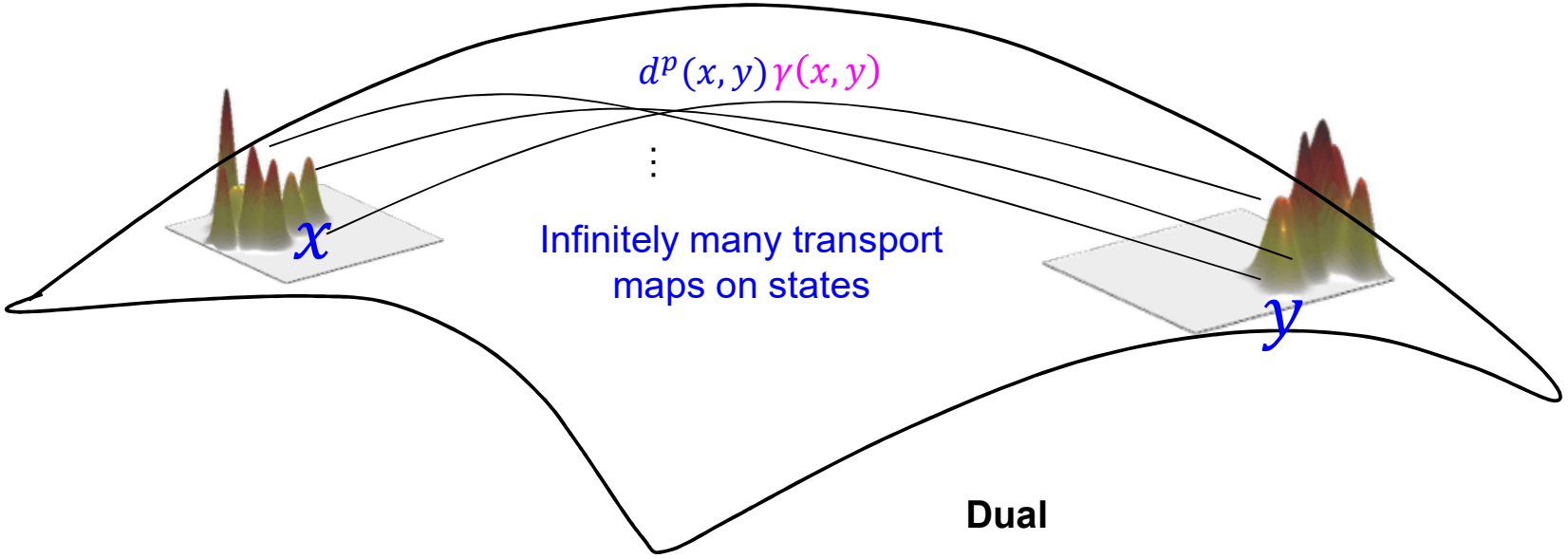
low dimensional manifolds in high dimension space hardly have overlaps

Wasserstein Distance – High Sample Complexity

Primal

cost map

$$W_p(P,Q) = \inf_{\gamma \in \Pi} < D, \Gamma >_F = \left(\inf_{\gamma \in \Pi} d^p(x,y)\gamma(x,y)\right)^{\frac{1}{p}} = \left(\inf_{\gamma \in \Pi} \mathbb{E}_{x,y\sim\gamma} d^p(x,y)\right)^{\frac{1}{p}}$$

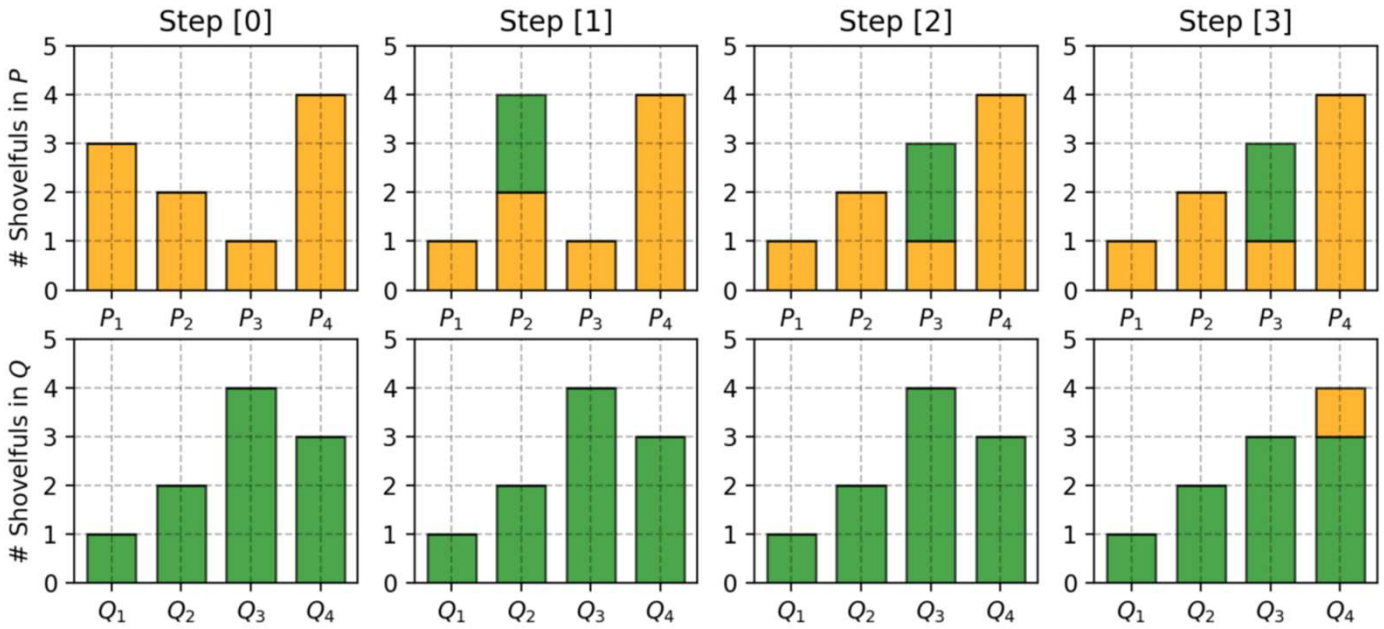


Dual

$$W_p(P,Q) = \sup_{\|f\|_L \leq 1} \mathbb{E}_{x\sim p_r}[f(x)] - \mathbb{E}_{y\sim p_\theta}[f(y)]$$

Divide and Conquer

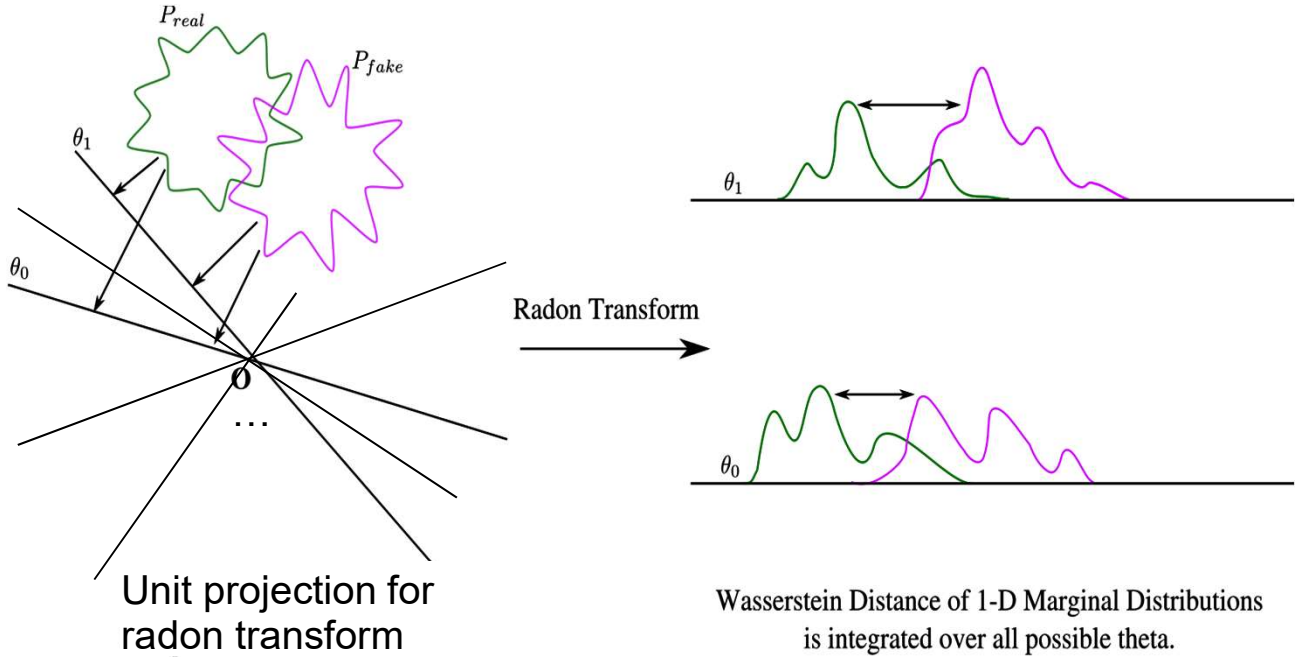
Sliced Wasserstein Distance – 1D case



- Optimal map (closed form, increasing arrangement): $\tau(x) = F_Q^{-1} \circ F_P(x)$, where F_P, F_Q is the corresponding cumulative distribution functions (CDFs)

Sliced Wasserstein Distance – High Projection Complexity

$SW_p(P,Q) = \left(\int_{\Omega} W_p(\pi_\theta^* P, \pi_\theta^* Q) d\theta\right)^{\frac{1}{p}}$, where Ω is a unit sphere, $\pi_\theta^* P = P \circ \pi_\theta$, $\pi_\theta(x) = \theta^T x$



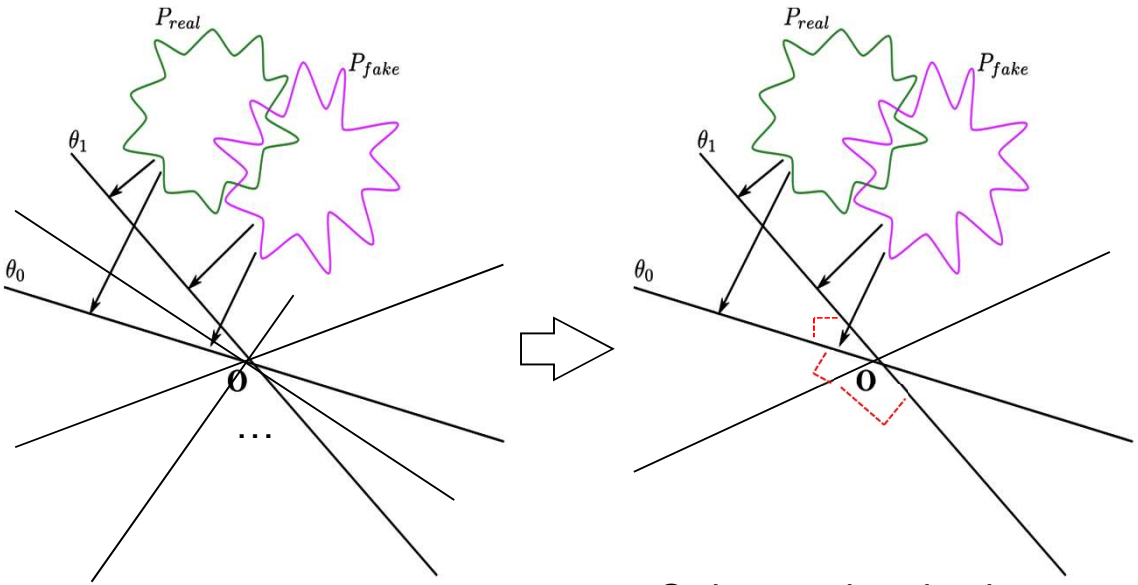
Unit projection for radon transform

Wasserstein Distance of 1-D Marginal Distributions is integrated over all possible theta.

Infinitely many random unit projections

Proposed Sliced Wasserstein Distance

- Learnable orthogonal projections for Radon Transform $\pi_\theta(x) = \theta^T x$ in a deep learning manner
  - Orthogonal projections vs. unit projections
    - More efficient to cover entire space
  - Trainable net weights vs. random weights
    - More compact



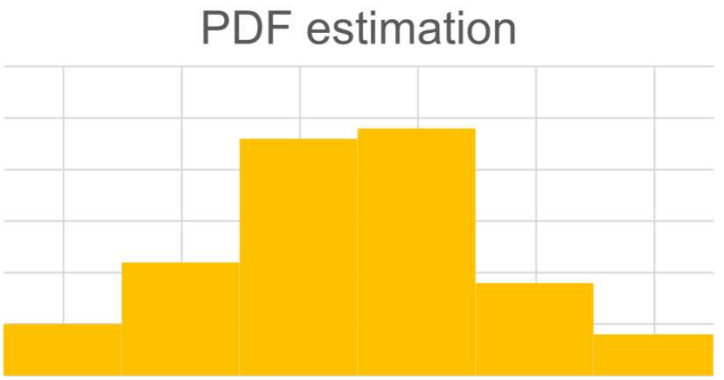
Unit projection for radon transform

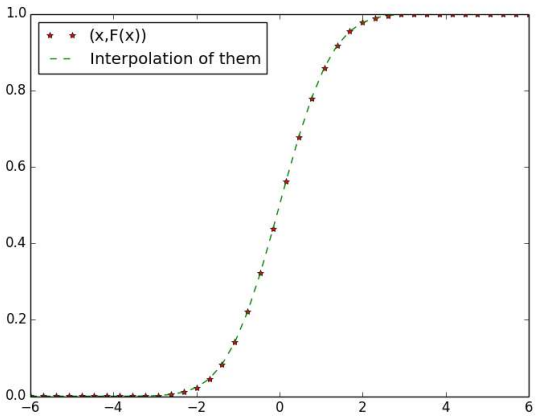
Orthogonal projection for radon transform

Proposed Sliced Wasserstein Distance

- Learnable orthogonal projections for Radon Transform $\pi_\theta(x) = \theta^T x$ in a deep learning manner

- Differetiable1D transport map $\tau_\theta = (\pi_\theta^* F_Q)^{-1} \circ \pi_\theta^* F_P$,
  - 1D PDF estimation using soft histogram assignment
    - $\frac{e^{-\alpha\|y-c_i\|^2}}{\sum_{j=1}^{l} e^{-\alpha\|y-c_j\|^2}}$ to the i-th bin, where c_i are the i-th bin center
  - (Inverse) 1D CDF estimation using linear interpolation





Proposed Sliced Wasserstein Distance

- Learnable orthogonal projections for Radon Transform $\pi_\theta(x) = \theta^T x$ in a deep learning manner

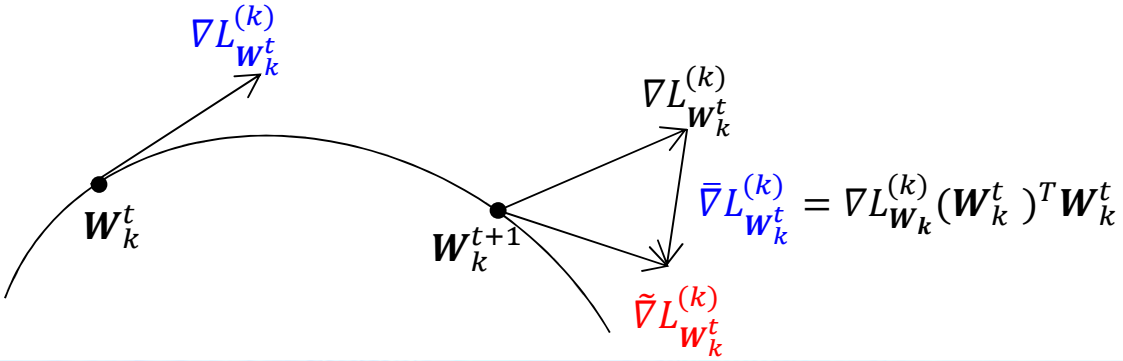
- Differetiable1D transport map $\tau_\theta = (\pi_\theta^* F_Q)^{-1} \circ \pi_\theta^* F_P$,

- Dual form (non-linearity f):

$$\int_{\mathbb{S}^{n-1}} \left(\sup_{f \in \mathrm{Lip}^k} \mathbb{E}_{X_{\boldsymbol{\theta}} \sim \pi_{\boldsymbol{\theta}}^* P_X} [f(X_{\boldsymbol{\theta}})] - \mathbb{E}_{Y_{\boldsymbol{\theta}} \sim \pi_{\boldsymbol{\theta}}^* P_Y} [f(Y_{\boldsymbol{\theta}})] \right) d\boldsymbol{\theta}$$

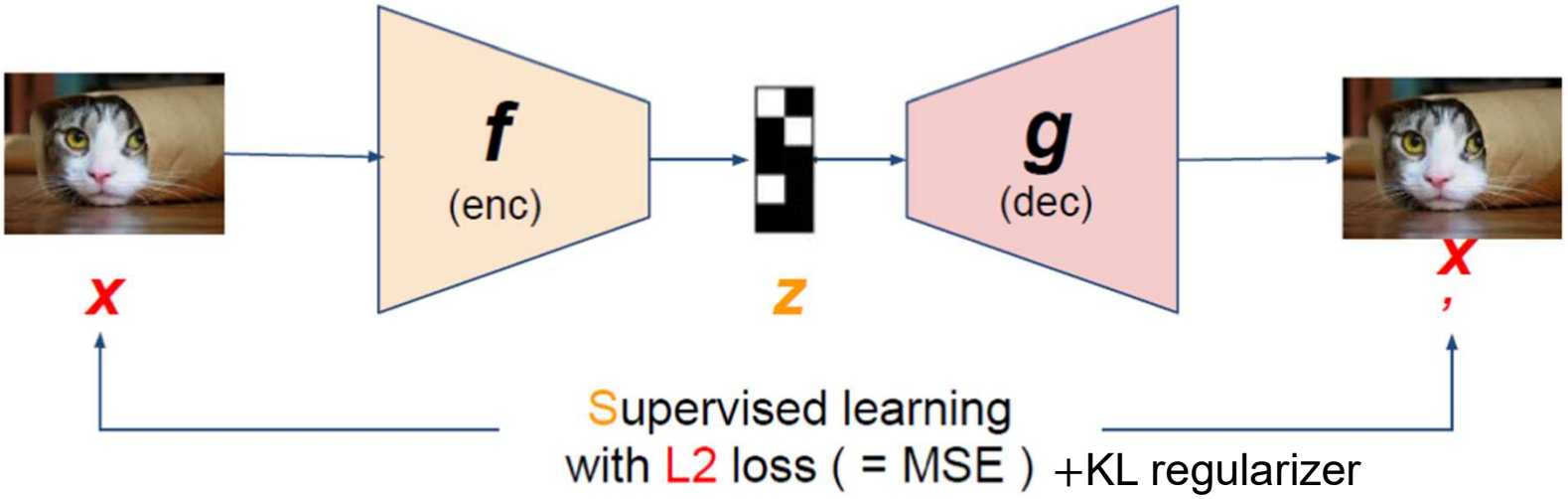
Proposed Sliced Wasserstein Distance

- Learnable orthogonal projections for Radon Transform $\pi_\theta(x) = \theta^T x$ in a deep learning manner
- Differetiable1D transport map $\tau_\theta = (\pi_\theta^* F_Q)^{-1} \circ \pi_\theta^* F_P$,
- Dual form (non-linearity f):
- Network training: updating orthogonal weights on Stiefel manifolds



How AE based generative models work?

- Reconstruction with penalization on latent variables
  - $\min_f(f, g) = \mathbb{E}_{q(z)}[\log p(x|z)] - KL(q(z)||p(z))$



Images from Namju Kim

Proposed Sliced Wasserstein Auto-Encoder (SWAE)

Algorithm 1 The proposed primal SWD block

Require: Orthogonal matrix $\boldsymbol{O_\Theta} = [\boldsymbol{\theta}_1, \dots, \boldsymbol{\theta}_r] \in \mathbb{R}^{r \times r}$, batch of latent codes $\boldsymbol{M_y} = [\boldsymbol{y}_1, \dots, \boldsymbol{y}_b] \in \mathbb{R}^{r \times b}$, batch of Gaussian noise $\boldsymbol{M_z} = [\boldsymbol{z}_1, \dots, \boldsymbol{z}_b] \in \mathbb{R}^{r \times b}$, and bin number l

Output: Batch of transferred latent codes $M_{\tilde{y}} = [\tilde{\boldsymbol{y}}_1, \dots, \tilde{\boldsymbol{y}}_b]$

for $i \leftarrow 1, r$ **do**

$\boldsymbol{y}'_i = \boldsymbol{\theta}_i^T \boldsymbol{M_y}, \boldsymbol{z}'_i = \boldsymbol{\theta}_i^T \boldsymbol{M_z}$

$\boldsymbol{y}''_i = \frac{\boldsymbol{y}'_i - \min_j\{y'_{i,j}\}}{\max_j\{y'_{i,j}\} - \min_j\{y'_{i,j}\}}, \boldsymbol{z}''_i = \frac{\boldsymbol{z}'_i - \min_j\{z'_{i,j}\}}{\max_j\{z'_{i,j}\} - \min_j\{z'_{i,j}\}}$,

$y'_{i,j}, z'_{i,j}$ are the j-th element of $\boldsymbol{y}'_i, \boldsymbol{z}'_i$ respectively.

Compute soft PDF histogram $p_{y''_i}, p_{z''_i}$ of $\boldsymbol{y}''_i, \boldsymbol{z}''_i$ with l bins

Compute CDF $F_{y''_i}, F_{z''_i}$ of $p_{y''_i}, p_{z''_i}$

Compute $F_{y''_i}(\boldsymbol{y}''_i)$ element-wise by linear interpolation

$\hat{\boldsymbol{y}}_i = (\max_j\{z'_{i,j}\} - \min_j\{z'_{i,j}\})(F_{z''_i})^{-1} F_{y''_i}(\boldsymbol{y}''_i) + \min_j\{z'_{i,j}\}$

end for

Compute $\boldsymbol{M_{\tilde{y}}} = \boldsymbol{O_\Theta} \boldsymbol{M}_{\hat{\boldsymbol{y}}}^T, \quad \boldsymbol{M_{\hat{y}}} = [\hat{\boldsymbol{y}}_1^T, \dots, \hat{\boldsymbol{y}}_r^T]$

Algorithm 2 The proposed SWAE

Require: Primal SWD block number m, batch size b, decoder G and encoder $Q = S_{p,m} \circ \dots \circ S_{p,2} \circ S_{p,1} \circ E$, training steps h, training hyperparameters, etc.

for $t \leftarrow 1, h$ **do**

Sample real data $\boldsymbol{M_x} = [\boldsymbol{x}_1, \dots, \boldsymbol{x}_b]$ from P_X

Sample Gaussian noise $\boldsymbol{M_z} = [\boldsymbol{z}_1, \dots, \boldsymbol{z}_b]$ from $\mathcal{N}(0, 1)$

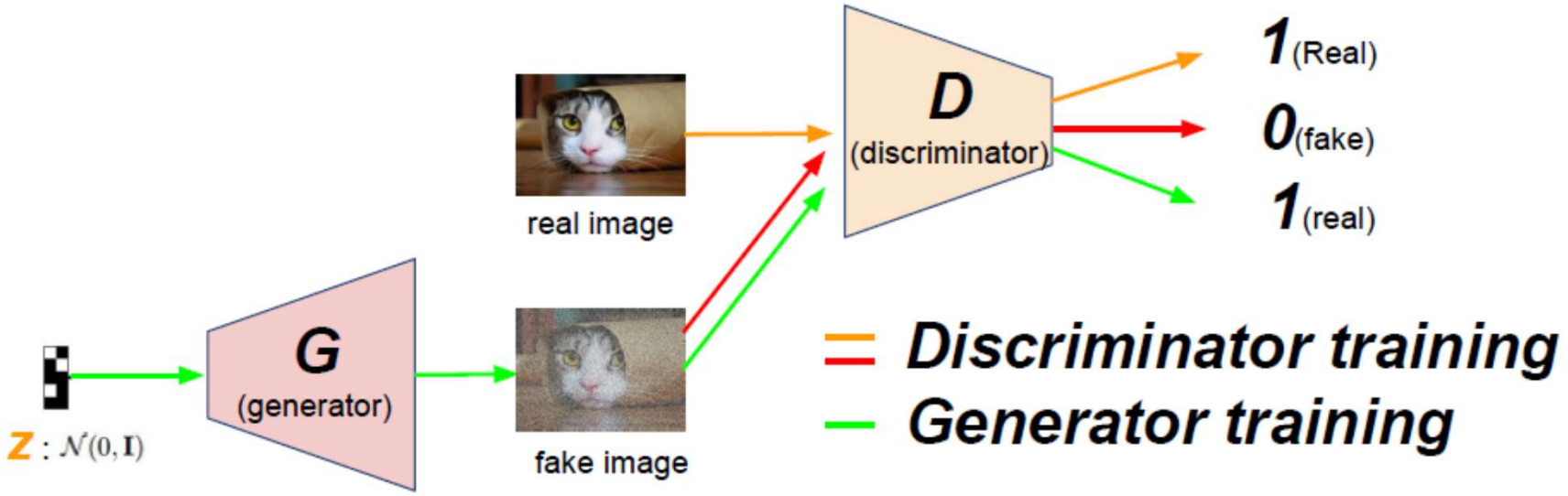
Update the weights $\boldsymbol{w}$ of Q and G by descending:

$\boldsymbol{w} \leftarrow \text{Adam}(\nabla_{\boldsymbol{w}}(\frac{1}{b}\|\boldsymbol{M_x} - G(Q(\boldsymbol{M_x}, \boldsymbol{M_z}))\|_2^2), \boldsymbol{w})$

end for

How Generative Adversarial Nets (GANs) work?

- Two-player game (min-max objective function)
  - $\min_G \max_D (D, G) = \mathbb{E}_{x \sim p_{data}(x)}[\log D(x)] + \mathbb{E}_{x \sim p_z(z)}\left[\log\left(1 - D\left(G(z)\right)\right)\right]$



Images from Namju Kim

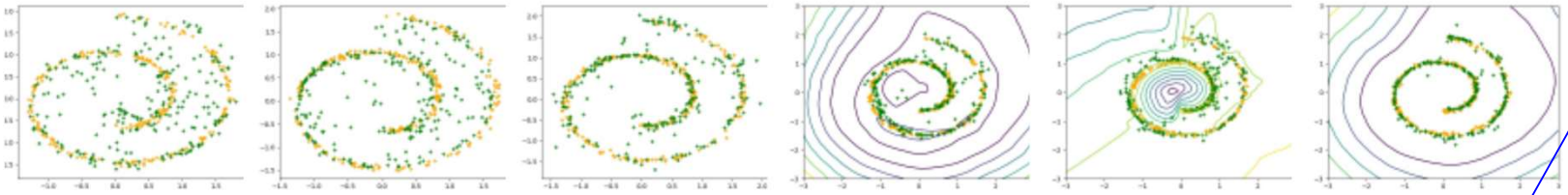
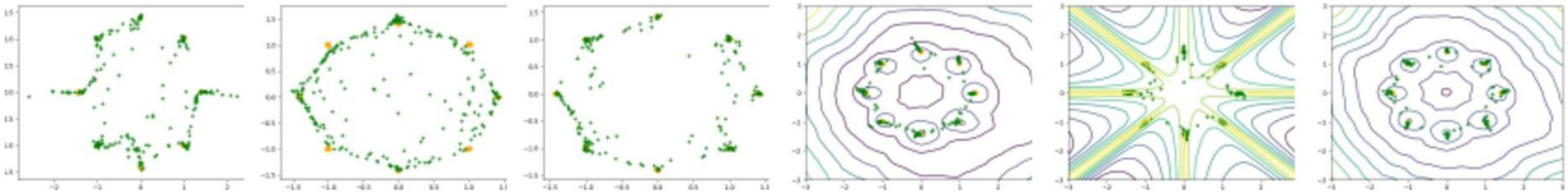
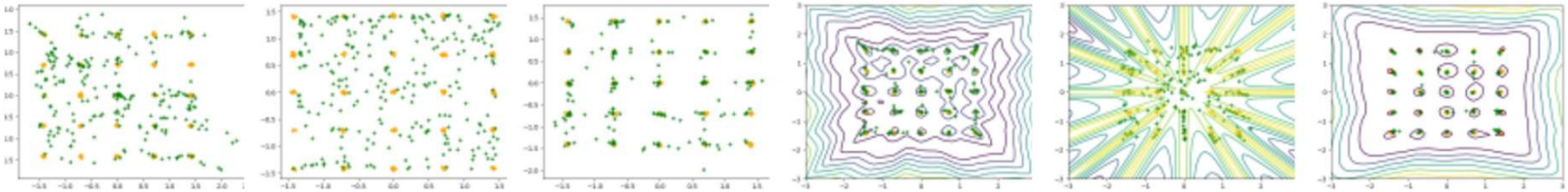
Proposed Sliced Wasserstein Generative Adversarial Nets (SWGAN)

Algorithm 3 The proposed dual SWD block

Require: Orthogonal matrix $\boldsymbol{O}_{\boldsymbol{\Theta}} = [\boldsymbol{\theta}_1, \dots, \boldsymbol{\theta}_r] \in \mathbb{R}^{r \times r}$ and batch of latent codes $\boldsymbol{M}_{\boldsymbol{y}} = [\boldsymbol{y}_1, \dots, \boldsymbol{y}_b] \in \mathbb{R}^{r \times b}$.
Output: Batch of $\tilde{\boldsymbol{y}}$ for dual SWD
for $i \leftarrow 1, r$ **do**
Compute $\boldsymbol{y}'_i = \boldsymbol{\theta}_i^T \boldsymbol{M}_{\boldsymbol{y}}$
Compute $\boldsymbol{y}''_i = F_i(\boldsymbol{y}'_i)$ element-wise, where $F = (F_1, \dots, F_r)$ are one-dimensional functions to approximate the f in Eq. 10.
end for
$\tilde{\boldsymbol{y}} = [\boldsymbol{y}''_1, \dots, \boldsymbol{y}''_b]^T$

Algorithm 4 The proposed SWGAN

Require: Number of dual SWD blocks m, batch size b, generator G and discriminator $D = [S_{d,1} \circ E, \dots, S_{d,m} \circ E]^T$, latent code dimension r, Lipschitz constant k, training steps h, training hyperparameters, etc.
for $t \leftarrow 1, h$ **do**
Sample real data $\boldsymbol{M}_{\boldsymbol{x}} = [\boldsymbol{x}_1, \dots, \boldsymbol{x}_b]$ from P_X
Sample Gaussian noise $\boldsymbol{M}_{\boldsymbol{z}} = [\boldsymbol{z}_1, \dots, \boldsymbol{z}_b]$ from $\mathcal{N}(0, 1)$
Sample two vectors $\boldsymbol{\mu_1}, \boldsymbol{\mu_2}$ from uniform distribution $U[0, 1]$ and for $l = 1, \dots, b$ calculate the elements of $\boldsymbol{M}_{\hat{\boldsymbol{x}}}, \boldsymbol{M}_{\hat{\boldsymbol{y}}}$:
$\hat{\boldsymbol{x}}_l = (1 - \mu_{1,l})\boldsymbol{x}_l + \mu_{1,l} G(\boldsymbol{z}_l)$
$\hat{\boldsymbol{y}}_l = (1 - \mu_{2,l})E(\boldsymbol{x}_l) + \mu_{2,l} E(G(\boldsymbol{z}_l))$
Update the weights $\boldsymbol{w}_G$ of G by descending:
$\boldsymbol{w}_G \leftarrow \text{Adam}(\nabla_{\boldsymbol{w}_G}(\frac{1}{b}\sum_{j,i=1}^{r \times m, b} D_{ji}(G(\boldsymbol{M}_{\boldsymbol{z}}))), \boldsymbol{w}_G)$
Update the weights $\boldsymbol{w}_D$ of D by descending:
$\boldsymbol{w}_D \leftarrow \text{Adam}(\nabla_{\boldsymbol{w}_D}(\frac{1}{b}\sum_{j,i=1}^{r \times m, b}(D_{ji}(\boldsymbol{M}_{\boldsymbol{x}}) - D_{ji}(G(\boldsymbol{M}_{\boldsymbol{z}})) + \lambda_1 \|\nabla_{\boldsymbol{M}_{\hat{\boldsymbol{x}}}} D(\boldsymbol{M}_{\hat{\boldsymbol{x}}})\|_2^2 + \lambda_2 \|\nabla_{\boldsymbol{M}_{\hat{\boldsymbol{y}}}} F(\boldsymbol{M}_{\hat{\boldsymbol{y}}}) - k \cdot \mathbf{1}\|_2^2), \boldsymbol{w}_D)$, where we compute the gradients of F element-wise.
end for

WAE	AE +100 IDT	**SWAE**	CT-GAN	SWG	**SWGAN**
					
0.04 (0.06)	0.05 (0.06)	**0.04 (0.04)**	0.01	0.03	**0.01**
					
0.07 (0.04)	0.06 (0.06)	**0.03 (0.02)**	0.03	0.05	**0.02**
					
0.05 (0.03)	0.05 (0.06)	**0.04 (0.02)**	0.02	0.05	**0.01**

Frechet Inception Distance (FID)

PG-WGAN

7.5

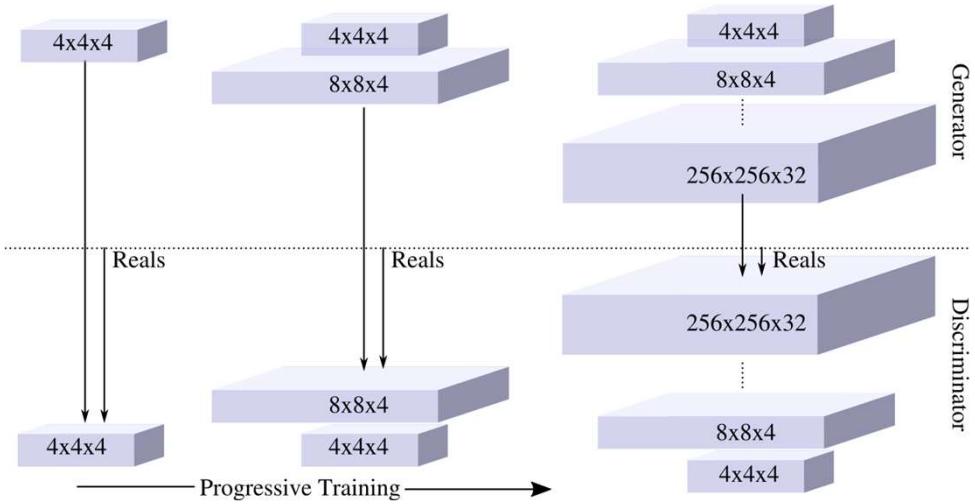
PG-SWGAN

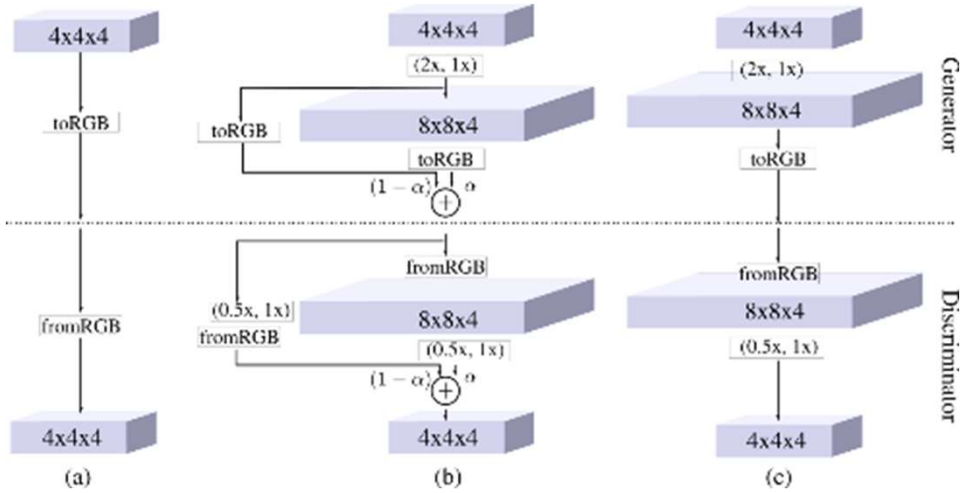
5.5

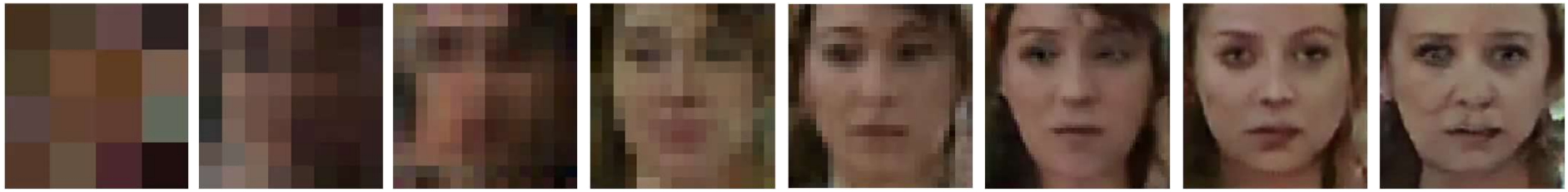
Frechet Inception Distance (FID)

	AMT Preference
PG-WGAN	0.45
PG-SWGAN	0.55

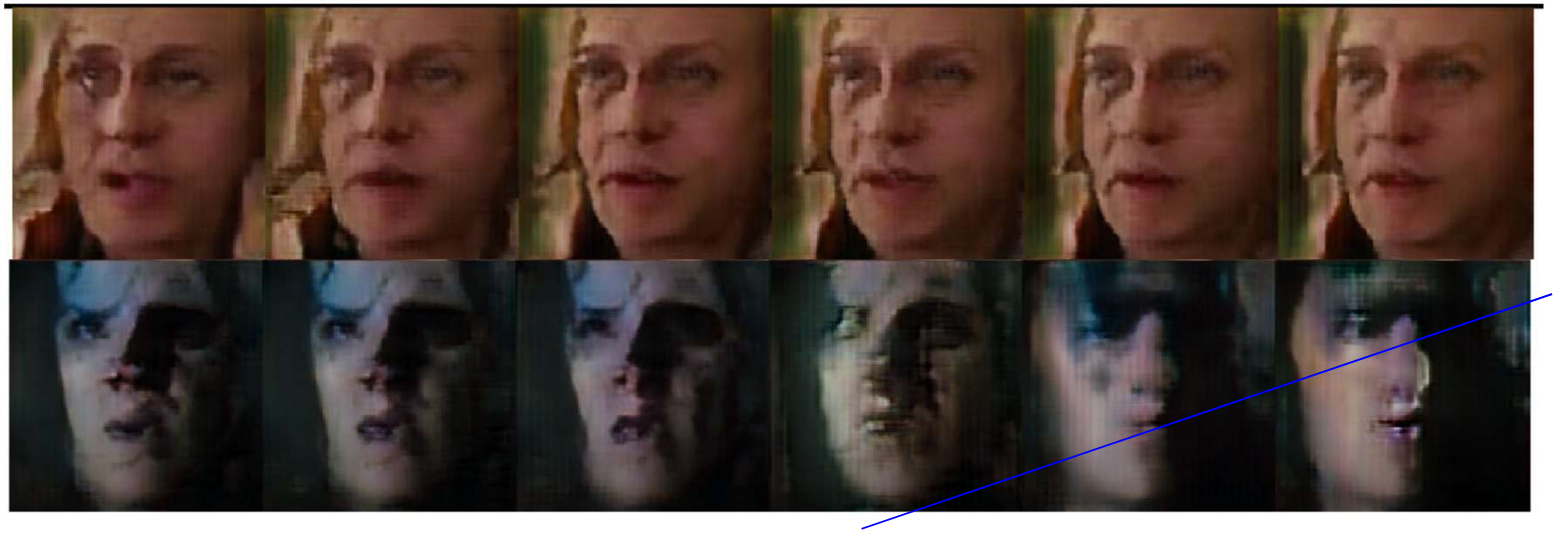
Progressive Growing Technique for Video Generation







PG-WGAN-3D



462.6

PG-SWGAN-3D

404.1

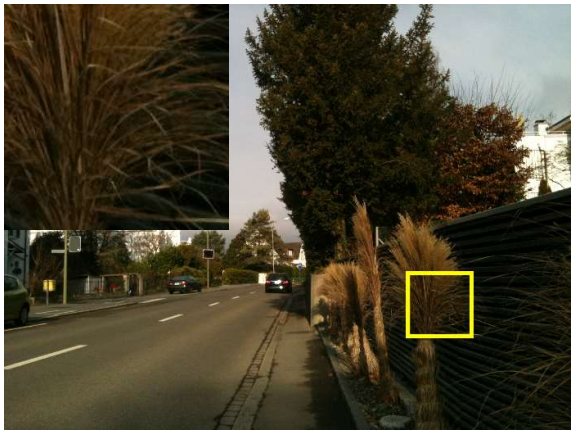
Frechet Inception Distance (FID)

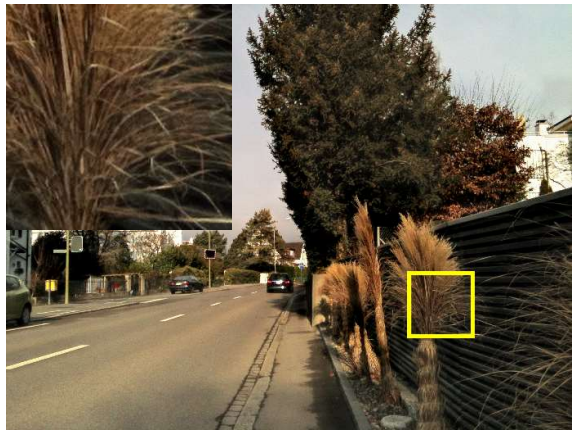
	AMT Preference
PG-WGAN	0.46
PG-SWGAN	0.54

Mixed-Perception Issue for Image Enhancement

Color Adjustment

Illumination Enhancement





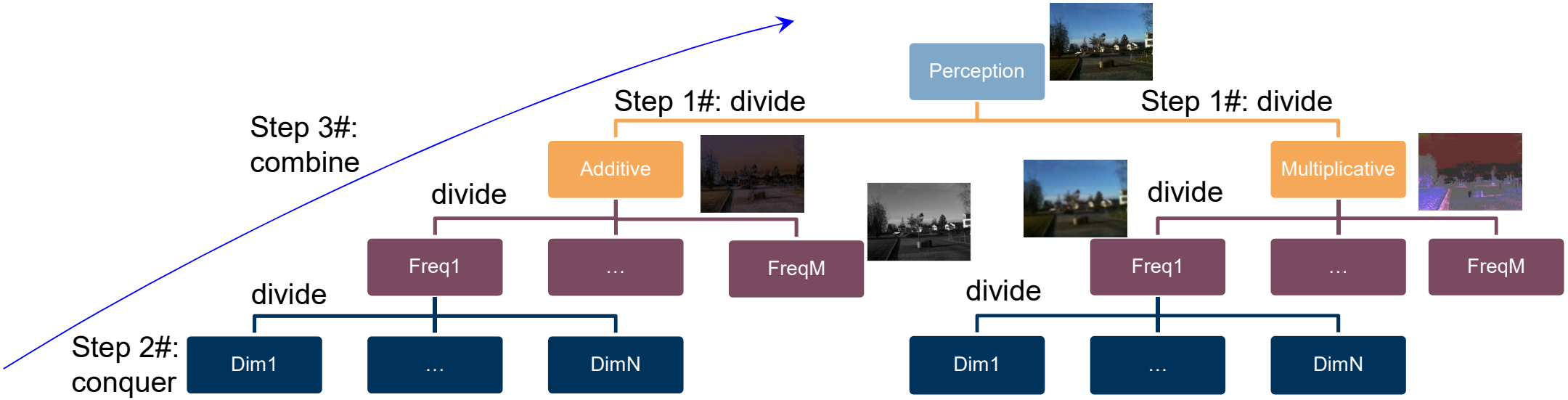
Texture Sharpening

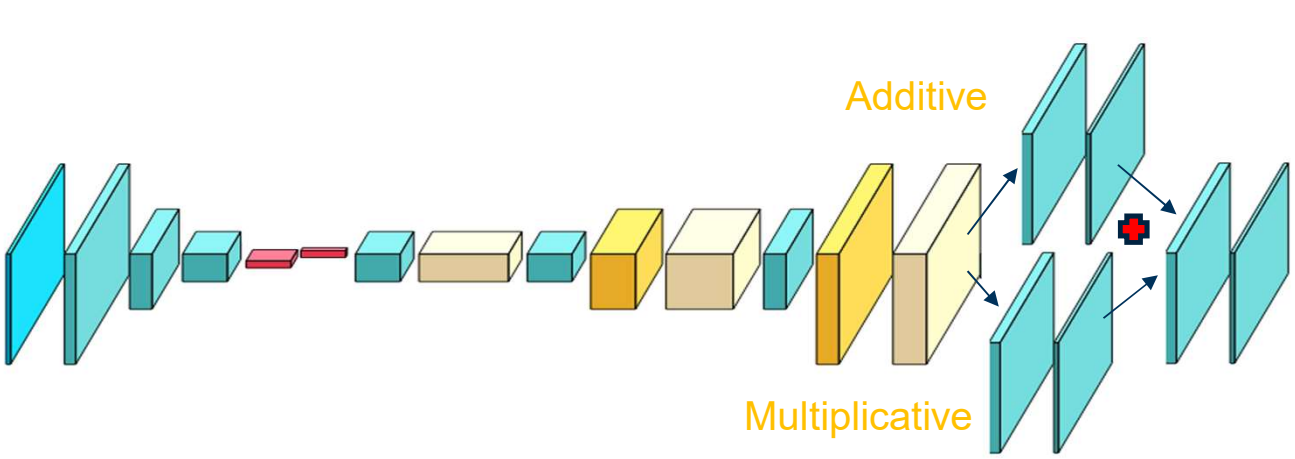
Divide and Conquer

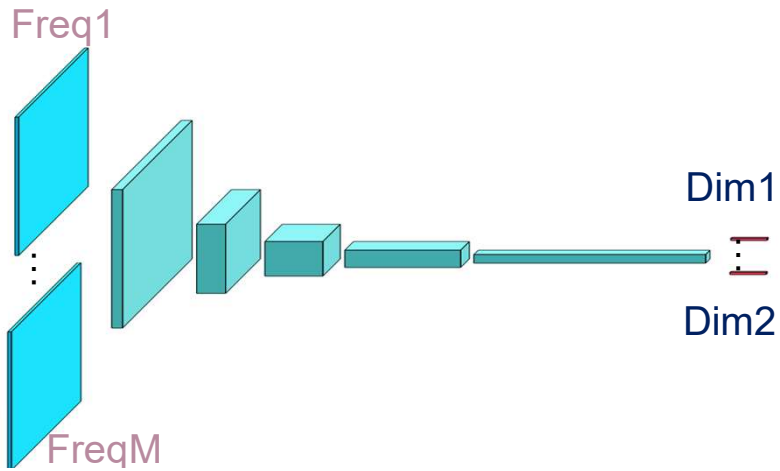
ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

CVL Computer Vision Lab

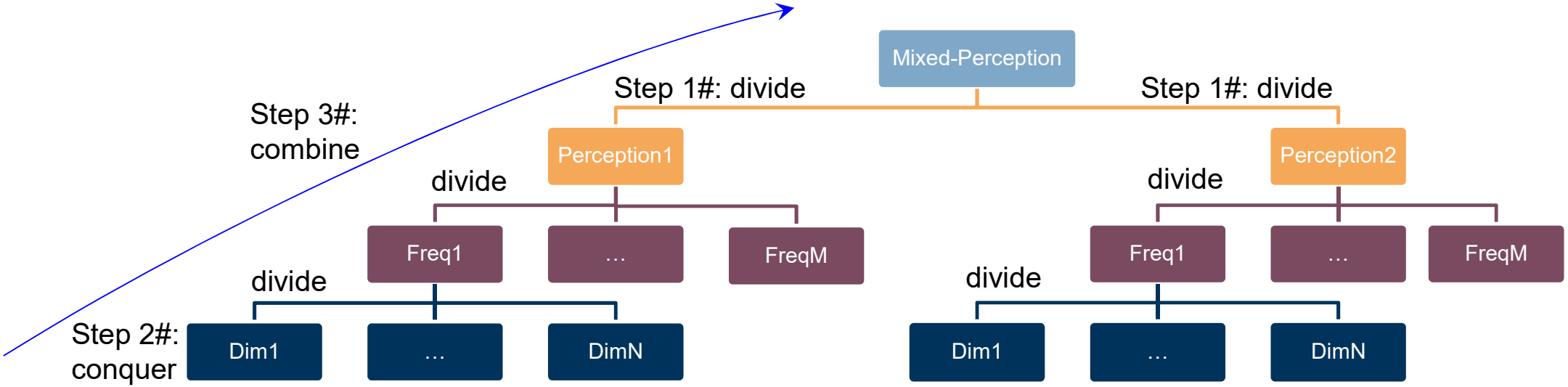
Hierarchically Multisliced Methodology

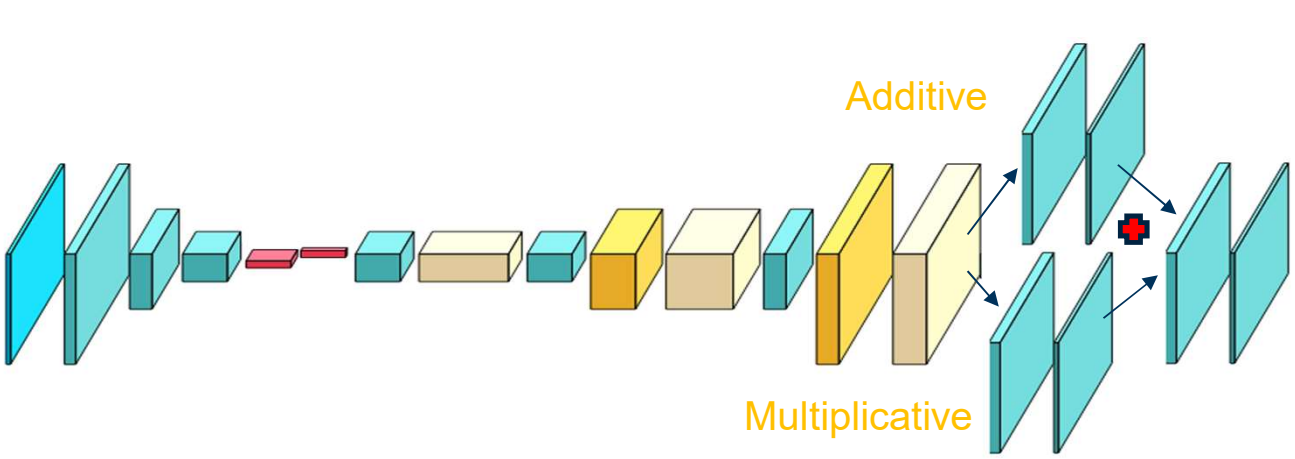


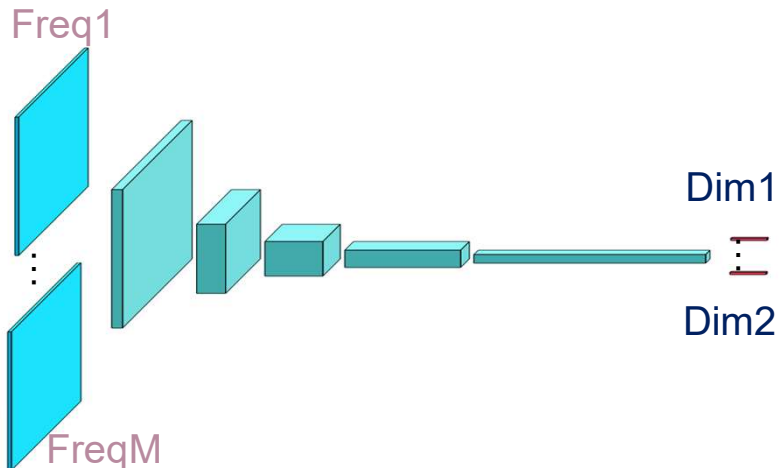




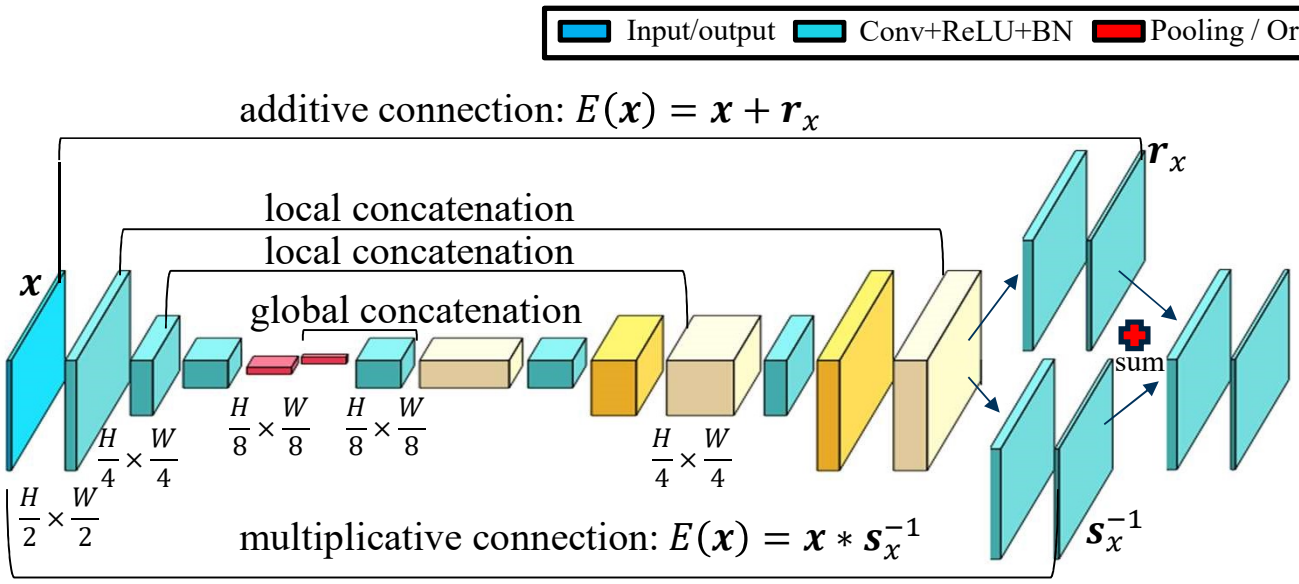
Hierarchically Multisliced Methodology



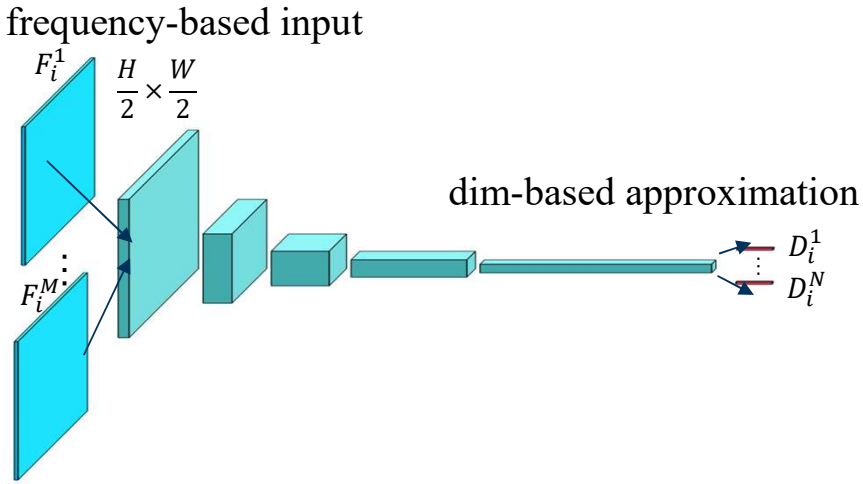




Network Design



(a) Enhancer for Peception-based Division

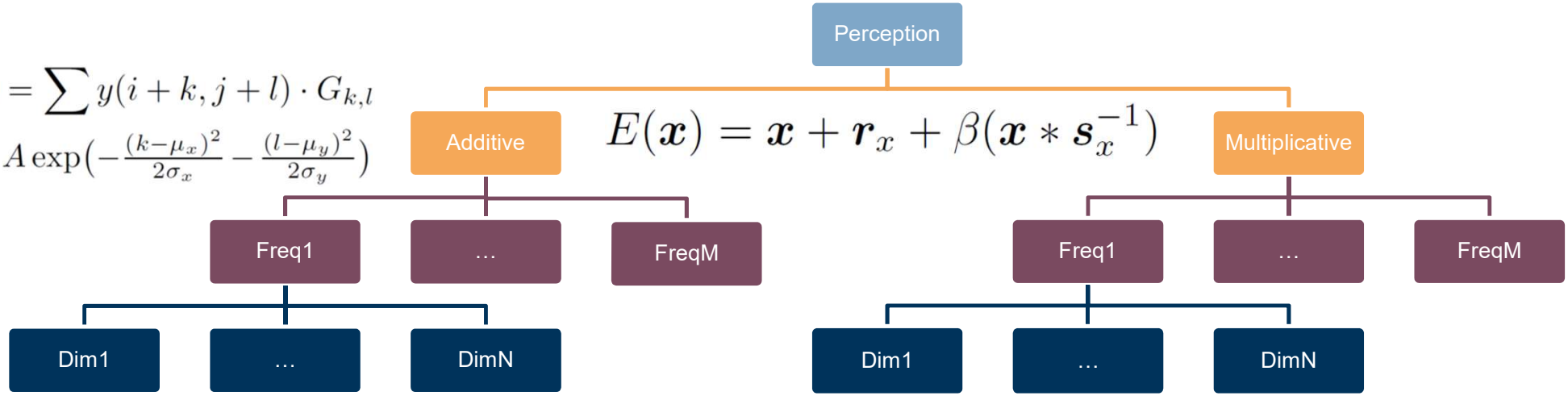


(b) Discriminator for Freq- and Dim-based Division

Hierarchically Multisliced Methodology

$$y_b(i,j) = \sum y(i+k, j+l) \cdot G_{k,l}$$

$$G_{k,l} = A \exp\left(-\frac{(k-\mu_x)^2}{2\sigma_x} - \frac{(l-\mu_y)^2}{2\sigma_y}\right)$$



New SWGAN loss

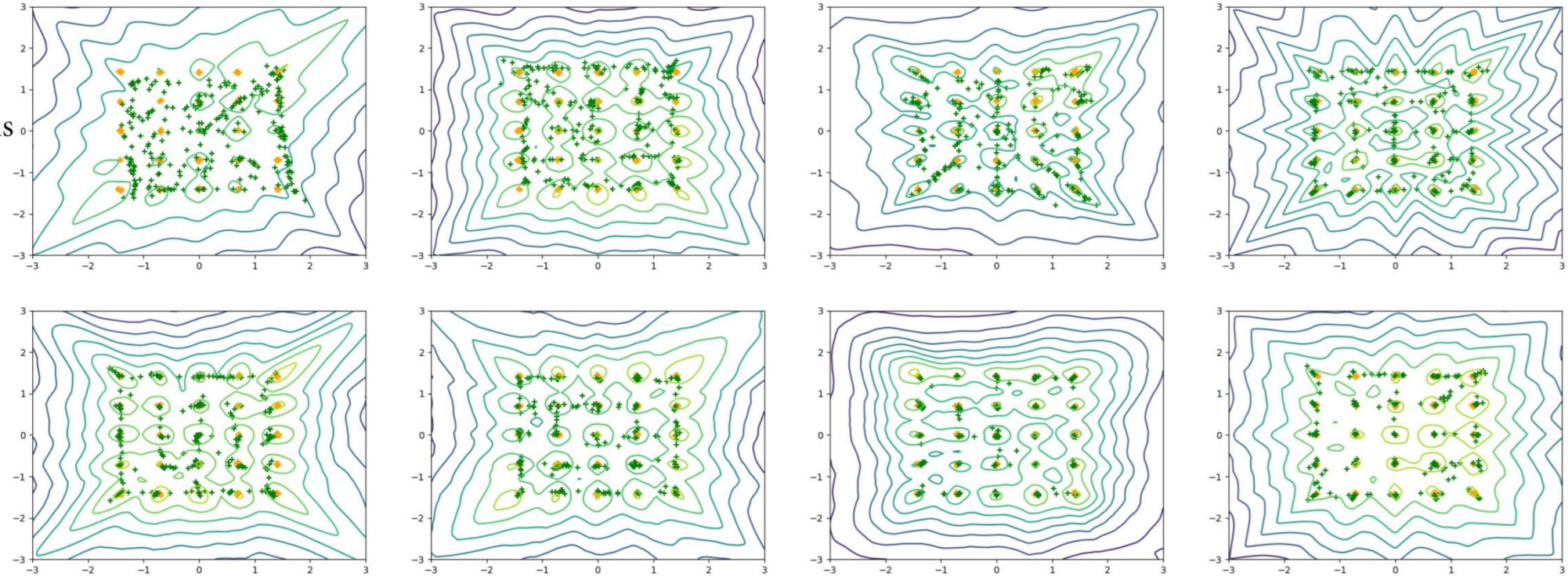
$$\min_E \max_C \int_{\theta \in \mathbb{S}^{n-1}} \left(\mathbb{E}_{\boldsymbol{y} \sim P_y}[C(\boldsymbol{y})] - \mathbb{E}_{\tilde{\boldsymbol{y}} \sim P_E}[C(E(\boldsymbol{x}))] \right) + \lambda \mathbb{E}_{\hat{\boldsymbol{y}} \sim P_{\hat{y}}}[\max(0, \|\nabla_{\hat{y}} C(\hat{\boldsymbol{y}})\|_2 - 1)]$$

$$\overline{\nabla_{\hat{y}} C(\hat{\boldsymbol{y}})} = \eta \overline{\nabla_{\hat{y}} C(\hat{\boldsymbol{y}})} + (1-\eta) \frac{\nabla_{\hat{y}} C(\hat{\boldsymbol{y}})}{\lambda}$$

Adaptive Penalty

2.5k iterations

5k iterations



(a) WGAN (b) AdaWGAN (c) SWGAN (d) Proposed AdaSWGAN

High-resolution Issue for Image Enhancement

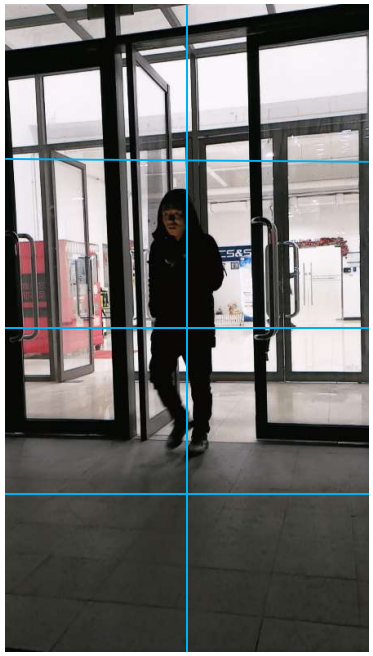
Input

✖

Downscaling
(low-res, noisy, blurry)

Deep Photo Ehancer (DPE) [Chen et al in CVPR'18]

✖



Patch-wise Enhancement
(spatial inconsistency)

Weakly Supervised Photo Enhancer (WESPE) [our work in CVPR'18 workshop

✔

coarse

fine

Multi-scale Photo Enhancement (MUSPE)

Our current work

Multi-scale Extension of DACAL for Image Enhancement

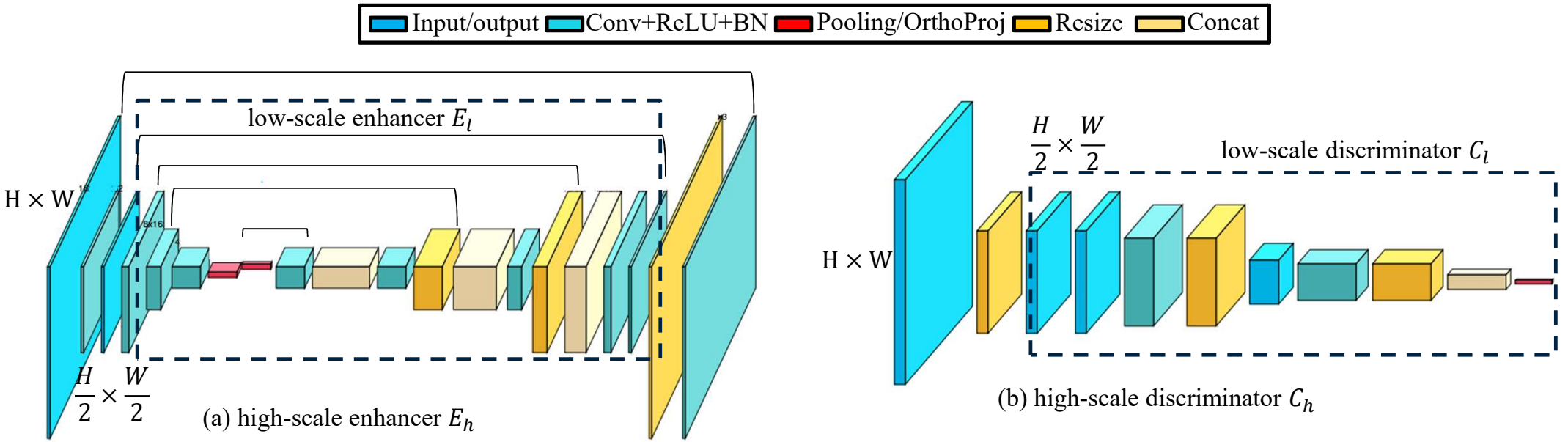
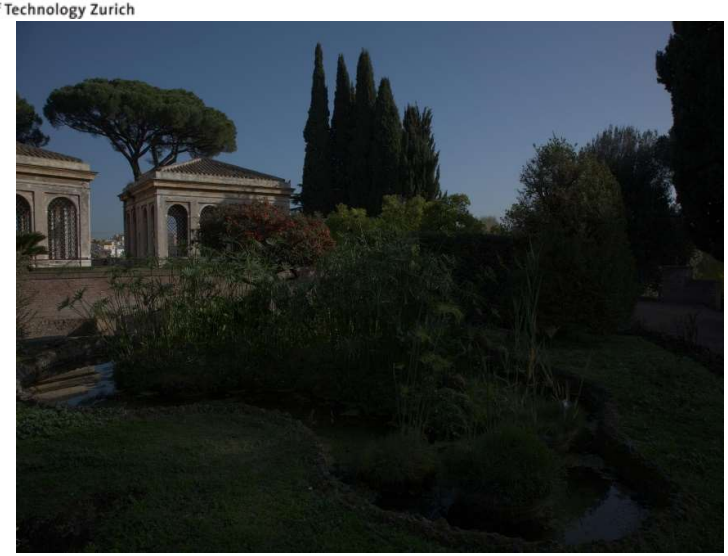


Table 1: PSNR and SSIM results for the MIT-Adobe FiveK [42] test images. Here, WB and DR indicate the White-Box and Distort-and-Recover methods, respectively. MUSPE_{l_1}, MUSPE_{l_2}, MUSPE_{l_3} and MUSPE_{l} represent the use of individual additive, individual multiplicative, multiplicative cascaded by additive, and our suggested parallel fusion (two-stream strategy), respectively. MUSPE_{h} is our higher-scale version. PSNR_d/SSIM_d and PSNR_f/SSIM_f indicate the results on downscaled images and full-resolution images, respectively.

	WB	DR	DPED	DPE	MUSPE_{l_1}	MUSPE_{l_2}	MUSPE_{l_3}	MUSPE_{l}	MUSPE_{h}
PSNR_d	18.86	21.64	21.05	22.10	22.73	22.99	23.01	23.52	**24.15**
PSNR_f	19.09	21.52	20.86	21.65	22.43	22.69	23.02	23.56	**24.07**
SSIM_d	0.928	0.936	0.922	0.947	0.958	0.942	0.949	0.959	**0.962**
SSIM_f	0.920	0.922	0.916	0.894	0.948	0.942	0.940	0.954	**0.956**

Table 2: PSNR and SSIM results for the DPED [14] test 100×100 image patches. Here, l, f, d for MUSPE represent the use of our proposed sliced-perception, sliced-frequency and sliced-dimension learning respectively. MUSPE_{h} is our higher-scale version.

	WESPE	DPE	MUSPE_{l}	MUSPE_{l+f}	MUSPE_{l+f+d}	MUSPE_{h}
PSNR_{100}	17.45	18.53	19.62	20.01	20.43	**20.90**
SSIM_{100}	0.854	0.861	0.868	0.869	0.872	**0.874**



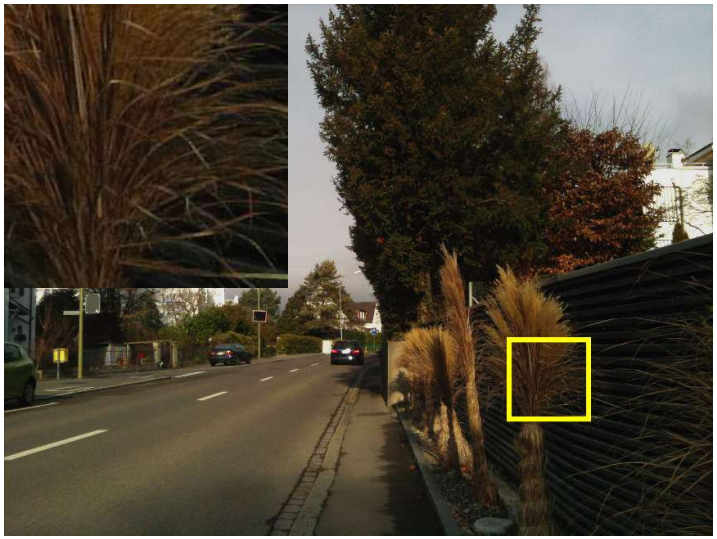
Input

WESPE [Ignatov, CVPRW'18]

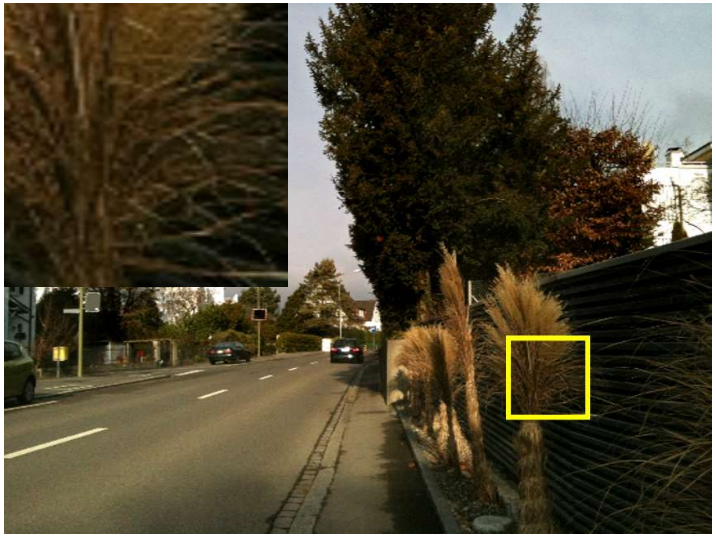
DPE [Chen, CVPR'18]

Proposed MUSPE [ICLR'20 submission]

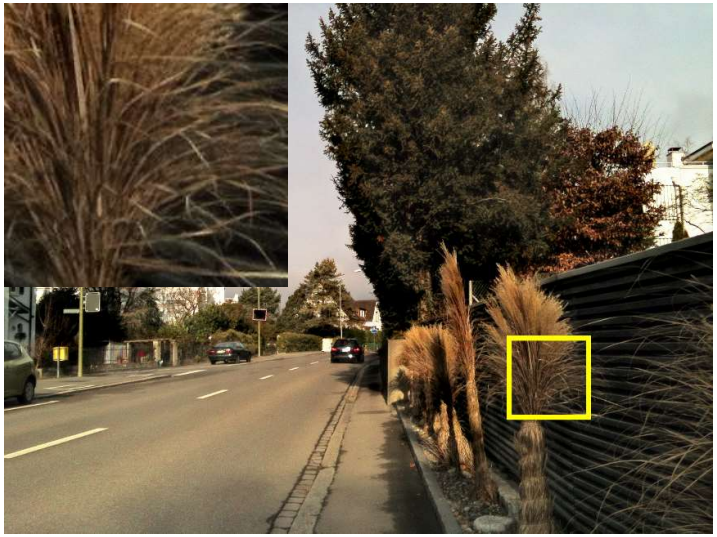
Input



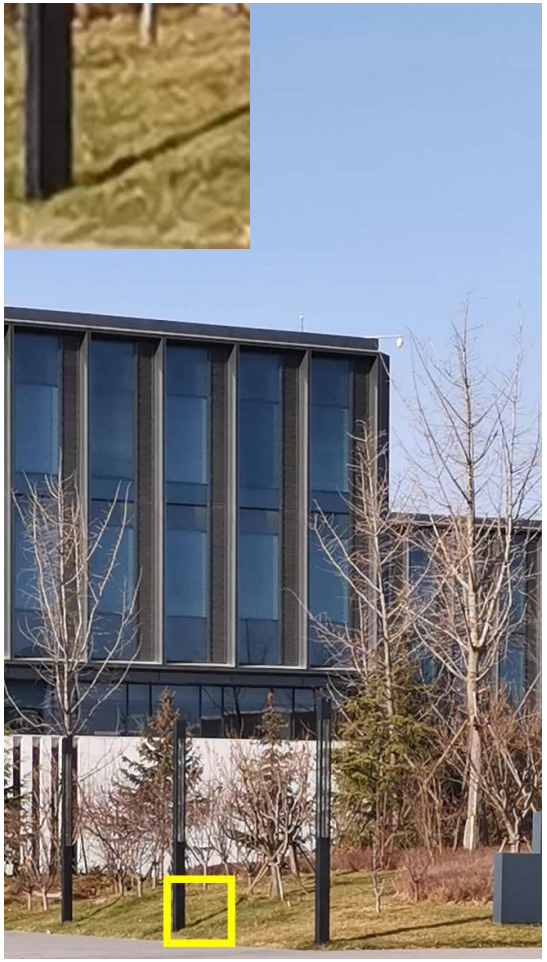
WESPE [Ignatov, CVPRW'18]



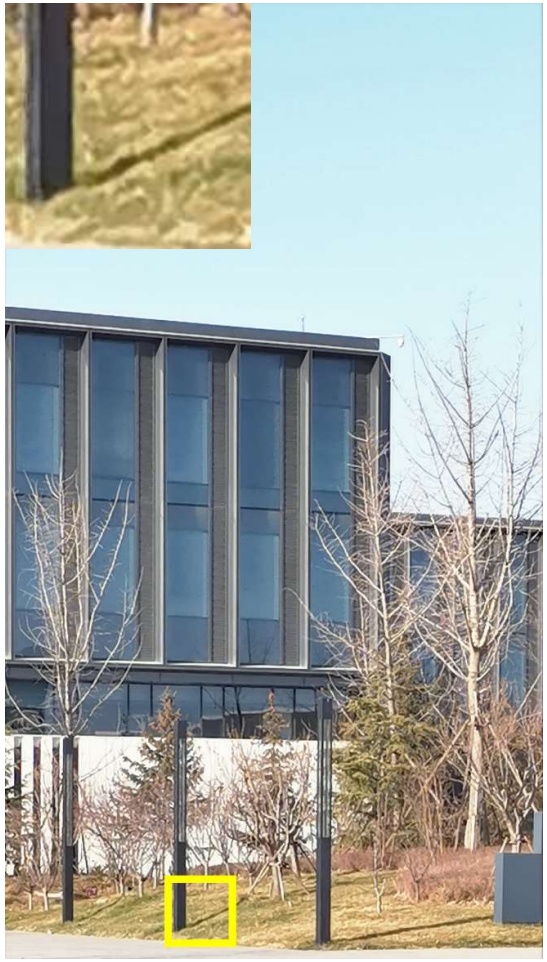
DPE [Chen, CVPR'18]



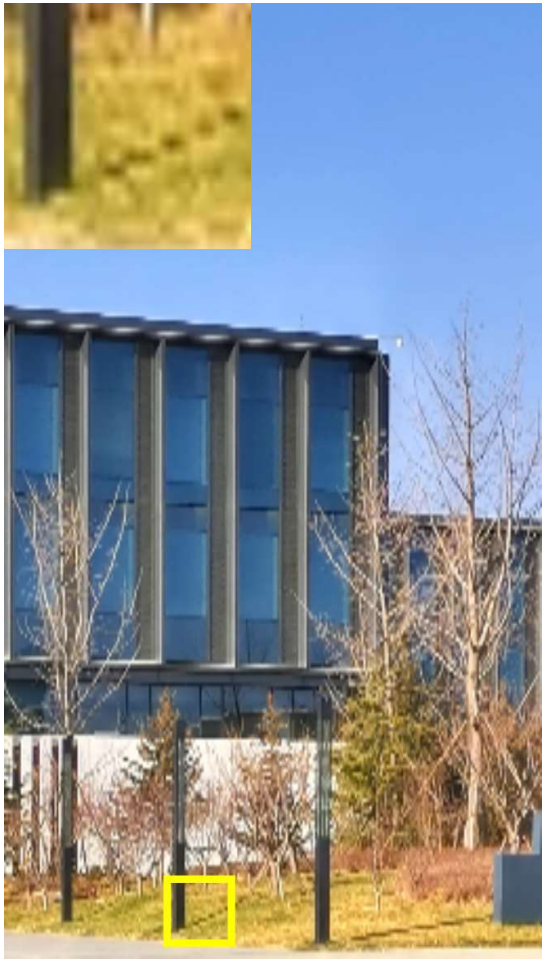
Proposed MUSPE [ICLR'20 submission]



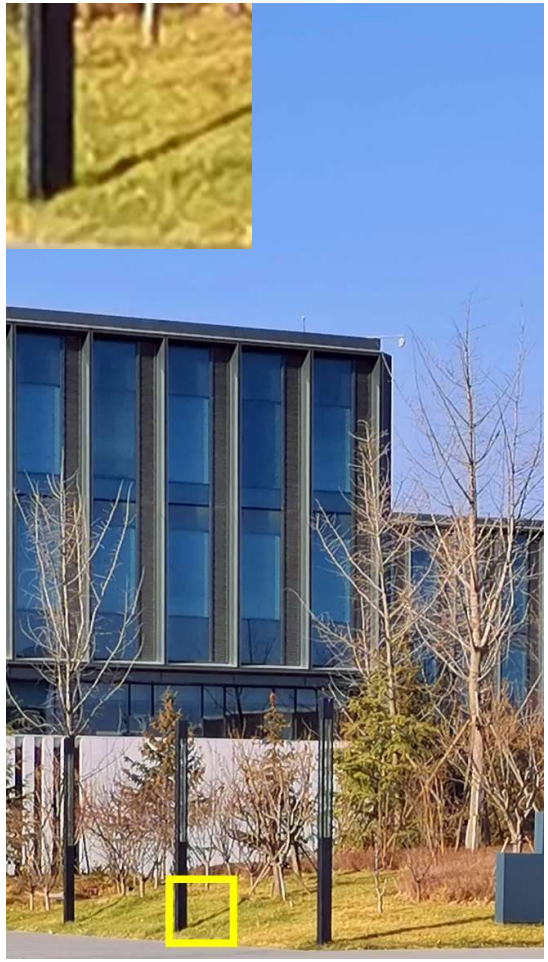
Input



WESPE [Ignatov, CVPRW'18]

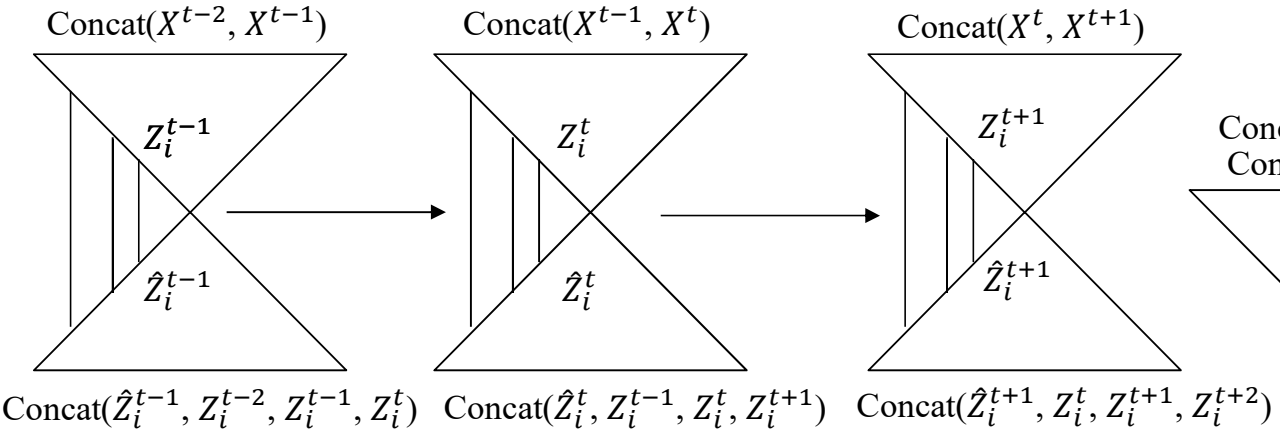


DPE [Chen, CVPR'18]

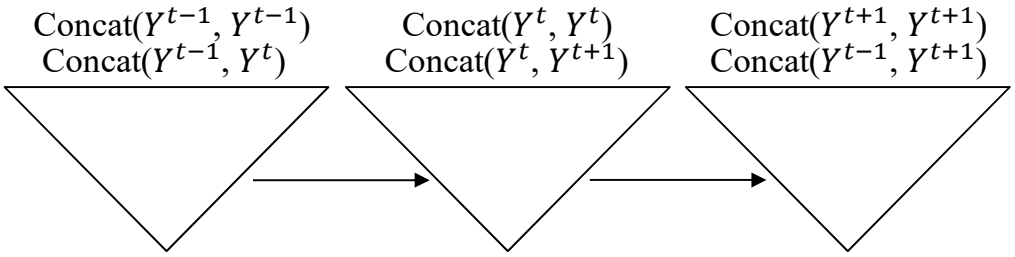


Proposed MUSPE
[ICLR'20 submission]

Recurrent Extension of DACAL for Video Enhancement



(a) recurrent extension of Generator



(b) recurrent extension of Disriminator

Perframe-DACAL

Recurrent-DACAL

Perframe-DACAL

Recurrent-DACAL (fine-tuned on Retouched&DSLR images)

Perframe-DACAL

Recurrent-DACAL

Perframe-DACAL

Recurrent-DACAL

Conclusion

- Sliced Wasserstein distance
  - Lower sample complexity
  - Lower projection complexity
- Sliced Wasserstein generative models for image & video generation
  - AE-based generative models
    - Penalization free on latent variables
    - Trained easier
  - GAN models
    - Easier dual form approximation
    - Reach state-of-the-art
- Divide-and-Conquer Adversarial Learning models for image & video enhancement
  - Hierachical decomposition of complexity
    - Adaptive sliced Wasserstein distance learning

Q&A