16. Deep Image Blending

paper : https://arxiv.org/abs/1910.11495 (WACV 2020; IEEE)

Lingzhi Zhang Tarmily Wen Jianbo Shi
University of Pennsylvania

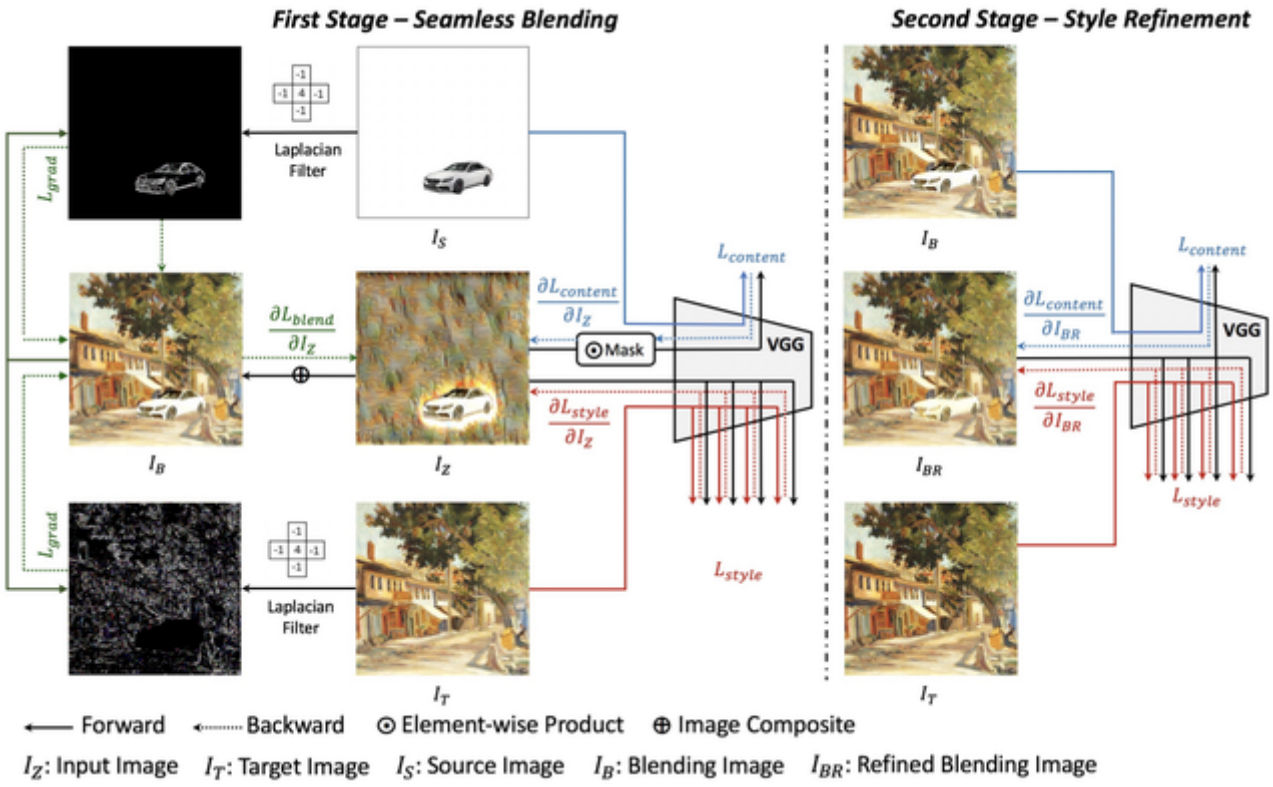
Reference

Poisson Image Edit (2003)

Image Style Transfer Using Convolutional Neural Networks (CVPR 2016)

Summary : Style Transfer + Poisson Image Edit을 동시에

Deep Image Blending (WACV 2020; IEEE)



Algorithm 1 First Stage - Seamless Blending

Input: source image I_S, blending mask M, target image I_T max iteration T, loss weights $\lambda_{grad}, \lambda_{cont}, \lambda_{style}, \lambda_{hist}, \lambda_{tv}$
Given: a gradient operator ∇, a pretrained VGG network F
Output: blending image I_B
for $i \in$ *[1:T]* **do**
    $I_B = I_Z \odot M + I_T \odot (1 - M)$
    $\mathcal{L}_{grad} = GradientLoss(I_B, I_S, I_T, \nabla)$ by Eq. 6
    $\mathcal{L}_{cont} = ContentLoss(I_Z, M, I_S, F)$ by Eq. 7
    $\mathcal{L}_{style} = StyleLoss(I_B, I_T, F)$ by Eq. 8
    $\mathcal{L}_{hist} = HistogramLoss(I_B, I_T, F)$ by Eq. 11
    $\mathcal{L}_{tv} = TVLoss(I_B)$ by Eq. 12
    $L_{total} = \lambda_{grad} * L_{grad} + \lambda_{cont} * L_{cont} + \lambda_{style} * L_{style} + \lambda_{hist} * L_{hist} + \lambda_{tv} * L_{tv}$
    $I_Z \leftarrow L\text{-}BFGS\_Solver(L_{total}, I_Z)$
end
$I_B = I_Z \odot M + I_T \odot (1 - M)$

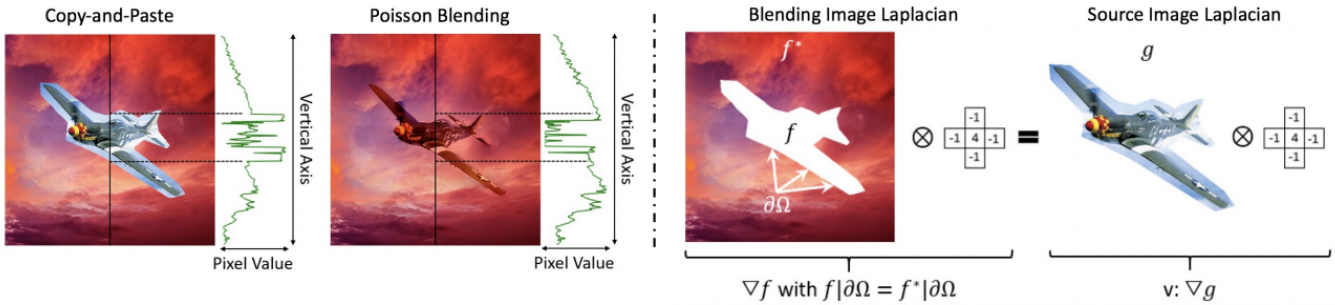
Algorithm 2 Second Stage - Style Refinement

Input: blending image I_B, target image I_T
max iteration T, loss weights $\lambda_{cont}, \lambda_{style}, \lambda_{hist}, \lambda_{tv}$
Given: a pretrained VGG network F
Output: refined blending image I_{BR}

$I_{BR} = copy(I_B)$
for $i \in [1{:}T]$ **do**
  $\mathcal{L}_{cont} = ContentLoss(I_{BR}, I_B, F)$ by Eq. 9
  $\mathcal{L}_{style} = StyleLoss(I_{BR}, I_T, F)$ by Eq. 10
  $\mathcal{L}_{hist} = HistogramLoss(I_{BR}, I_T, F)$ by Eq. 11
  $\mathcal{L}_{tv} = TVLoss(I_{BR})$ by Eq. 12
  $L_{total} = \lambda_{grad} * L_{grad} + \lambda_{cont} * L_{cont} + \lambda_{style} * L_{style} + \lambda_{hist} * L_{hist} + \lambda_{tv} * L_{tv}$
  $I_{BR} \leftarrow L\text{-}BFGS\_Solver(L_{total}, I_{BR})$
end

Loss Function

(1) gradient loss



Poisson Image Editing equation (3)에서 **guidance vector field는 source, target 각각 얻은 gradient.**

Blending Image 와 Source Image + Target Image 의 gradient를 동일하게 하는 Loss로 정의

$$\mathcal{L}_{grad} = \frac{1}{2HW} \sum_{m=1}^{H} \sum_{n=1}^{W} [\nabla f(I_B) - (\nabla f(I_S) + \nabla f(I_T))]_{mn}^2 \quad (6)$$

(2) content loss & style loss

$$\mathcal{L}_{cont} = \sum_{l=1}^{L} \frac{\alpha_l}{2N_l M_l} \sum_{i=1}^{N_l} \sum_{k=1}^{M_l} (F_l[I_Z] \odot M - F_l[I_S])_{ik}^2 \quad (7)$$

$$\mathcal{L}_{style} = \sum_{l=1}^{L} \frac{\beta_l}{2N_l^2} \sum_{i=1}^{N_l} \sum_{j=1}^{N_l} (G_l[I_Z] - G_l[I_T])_{ij}^2 \quad (8)$$

$$\mathcal{L}_{cont} = \sum_{l=1}^{L} \frac{\alpha_l}{2N_l M_l} \sum_{i=1}^{N_l} \sum_{k=1}^{M_l} (F_l[I_{BR}] - F_l[I_B])_{ik}^2 \quad (9)$$

$$\mathcal{L}_{style} = \sum_{l=1}^{L} \frac{\beta_l}{2N_l^2} \sum_{i=1}^{N_l} \sum_{j=1}^{N_l} (G_l[I_{BR}] - G_l[I_T])_{ij}^2 \quad (10)$$

L : Layer

F(*) : Layer에서 Activation Function

N : the number of channel in Activation

M_l: the number of flattened activation values in each channel

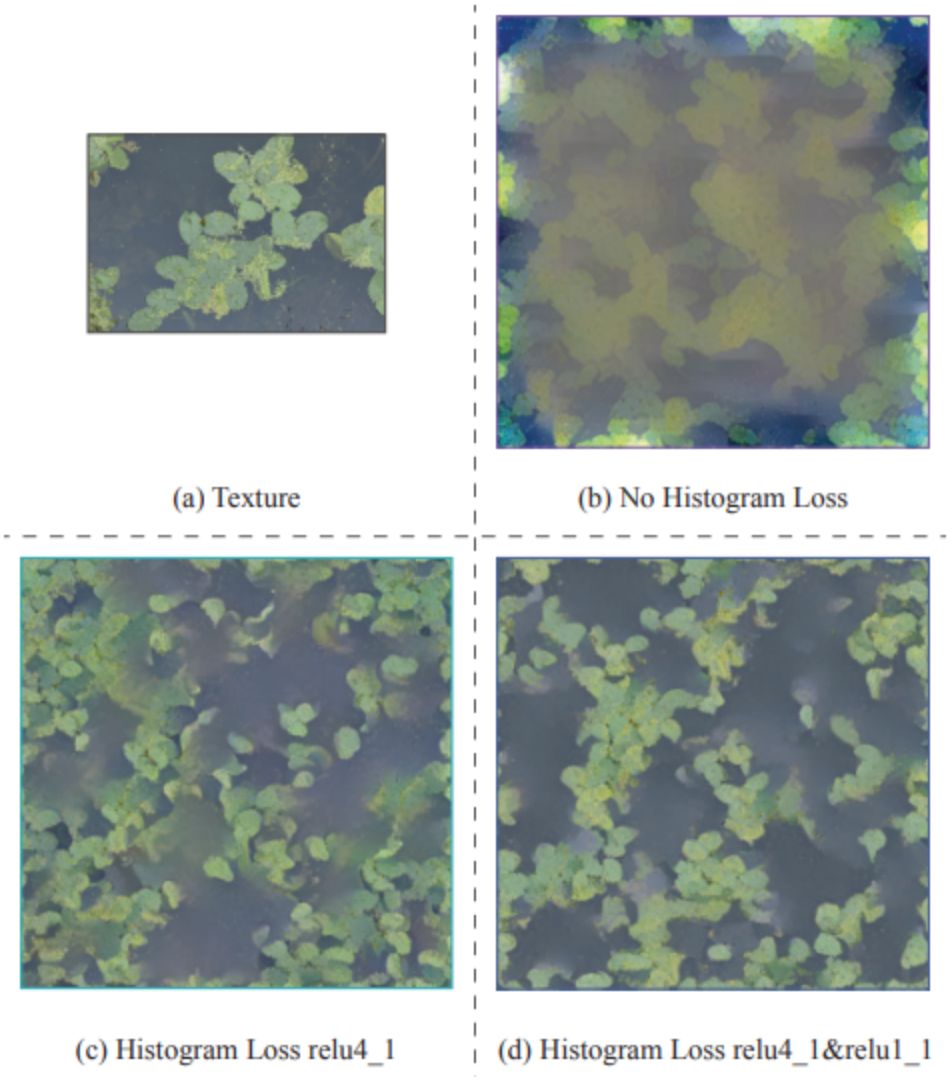
G : Gram Matrix

a & b : control the influence of each layer weights

(3) regularization loss

- histogram loss

Gram Matrix로 texture 정보는 뽑을 수 있지만, 모든 channel에서 mean or variance가 고르지 않을 수 있다.



(a) Texture (b) No Histogram Loss

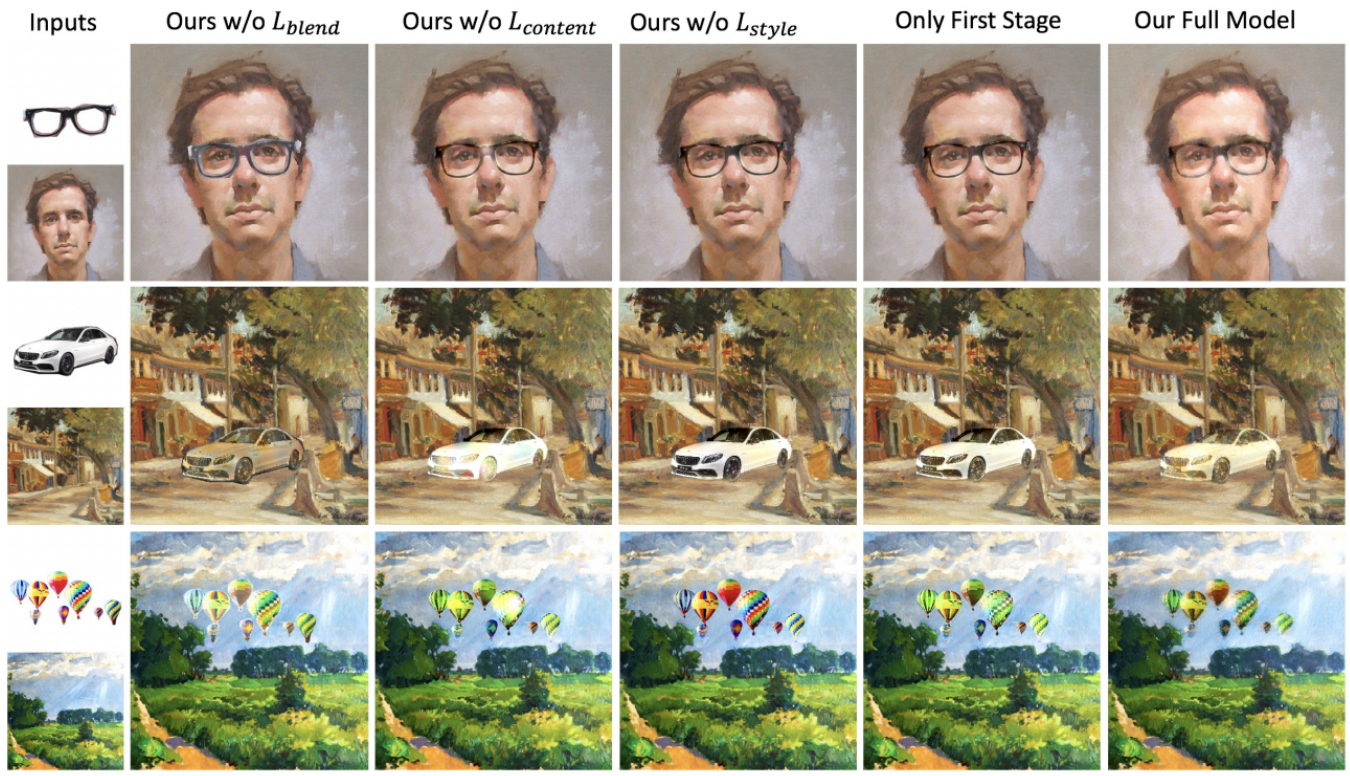
(c) Histogram Loss relu4_1 (d) Histogram Loss relu4_1&relu1_1

$$\mathcal{L}_{hist} = \sum_{l=1}^{L} \gamma_l \left\| F_l(I_B) - R_l(I_B) \right\|_{\mathrm{F}}^2 \qquad (11)$$

R : blend image 와 target image의 histogram matching 결과

- total variation loss

$$\mathcal{L}_{tv} = \sum_{m=1}^{H} \sum_{n=1}^{W} |I_{m+1,n} - I_{m,n}| + |I_{m,n+1} - I_{m,n}| \quad (12)$$



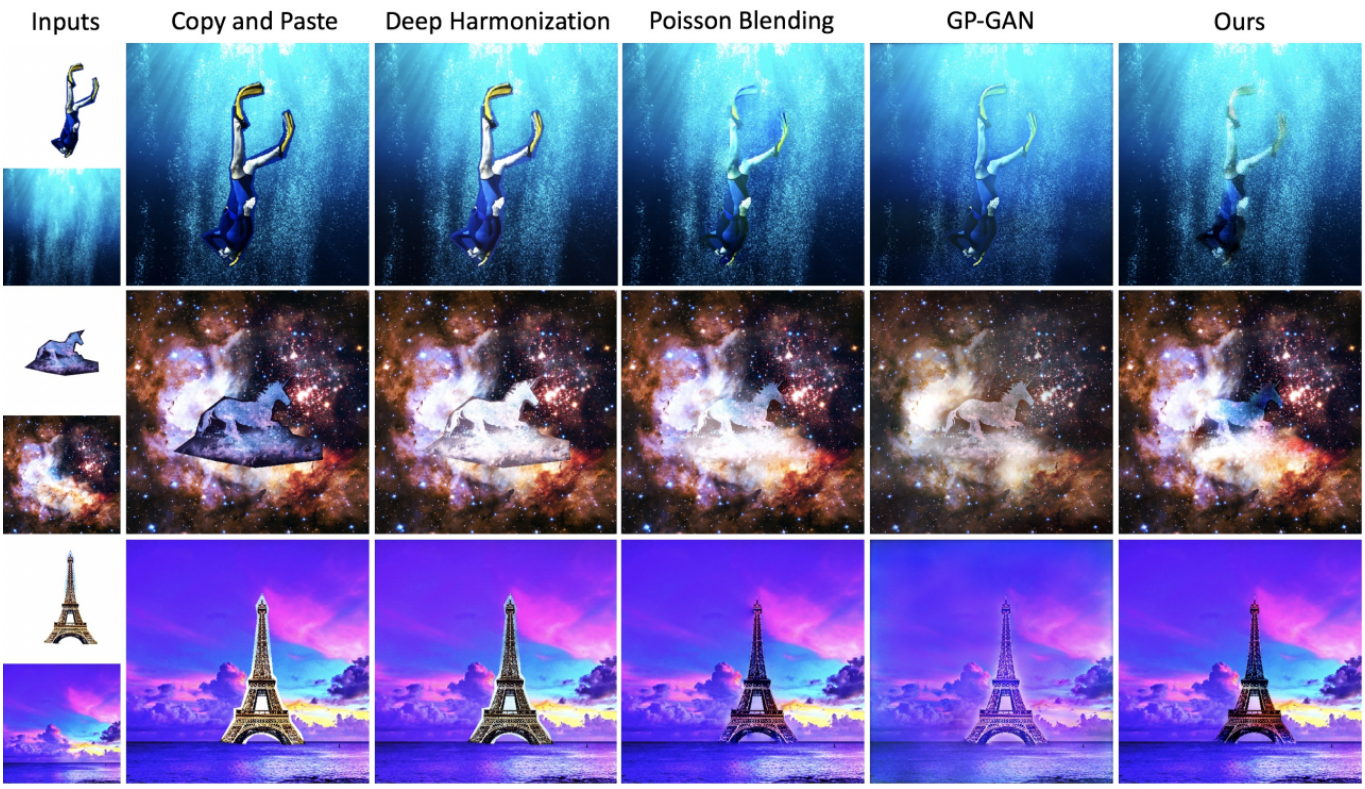
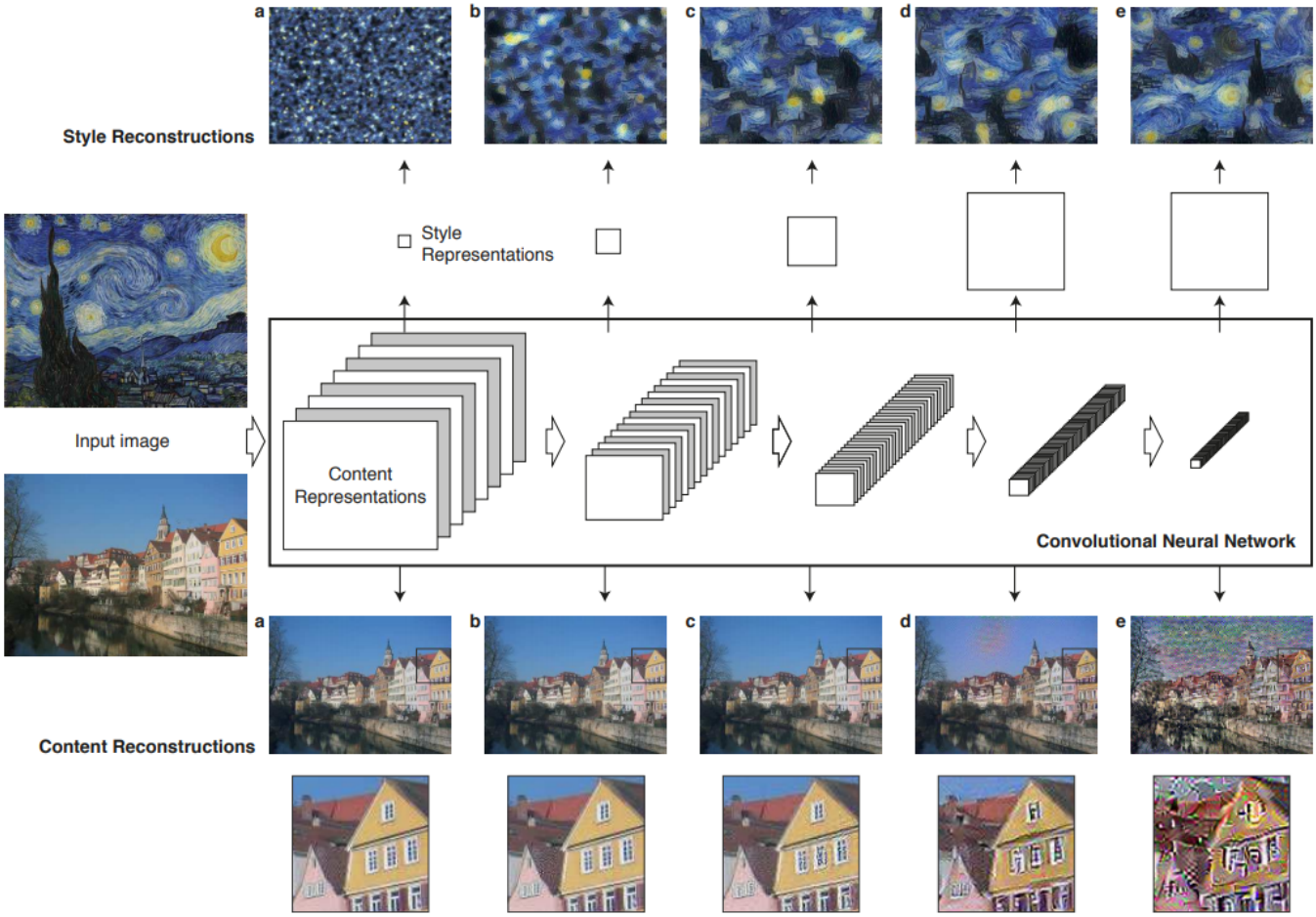
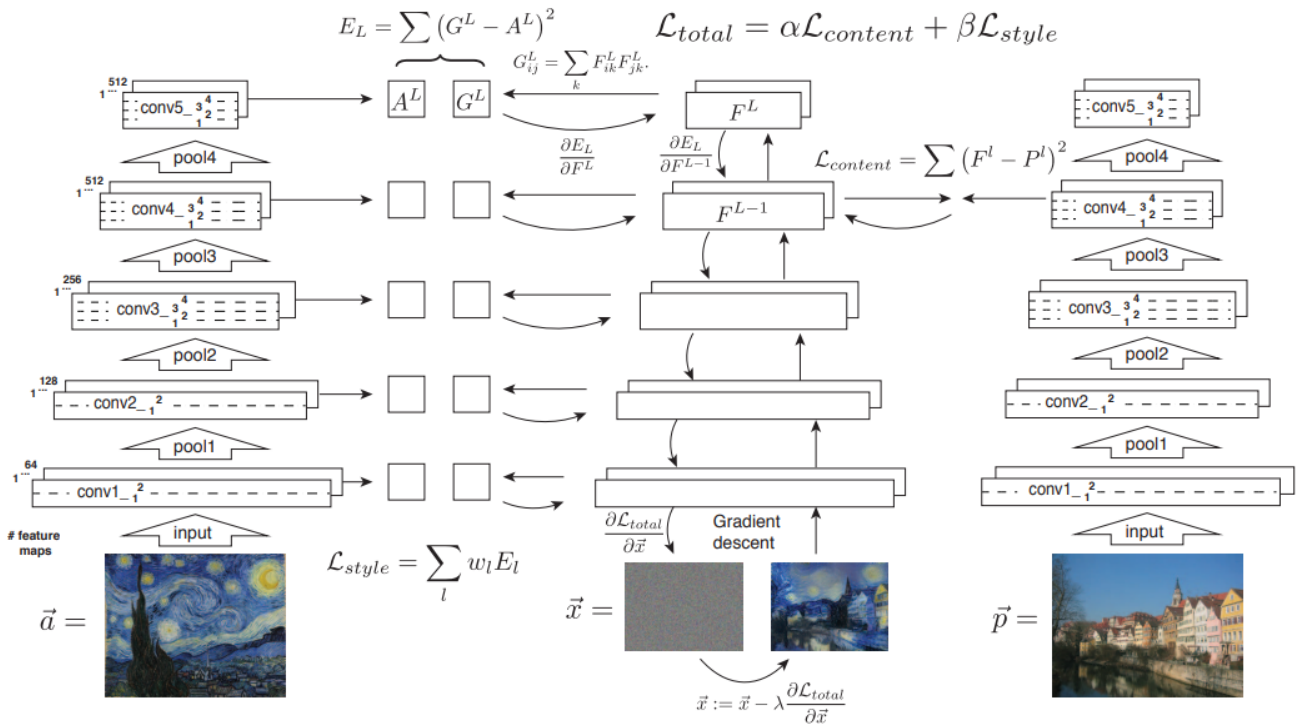


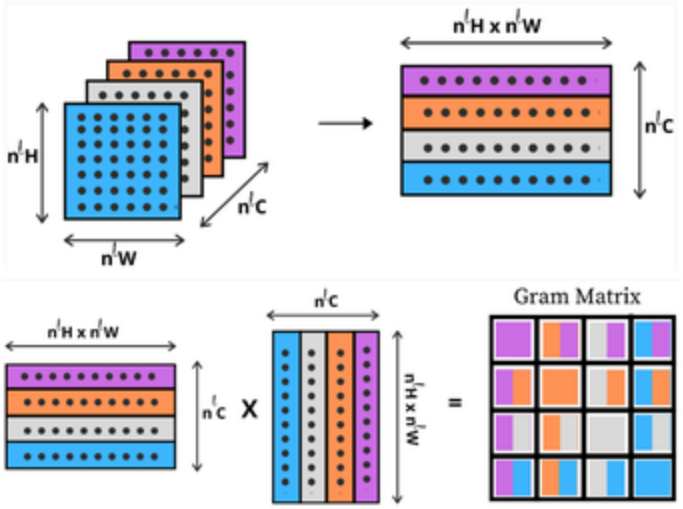
Image Style Transfer Using Convolutional Neural Networks (CVPR 2016)

style transfer = texture transfer





texture = Gram Matrix

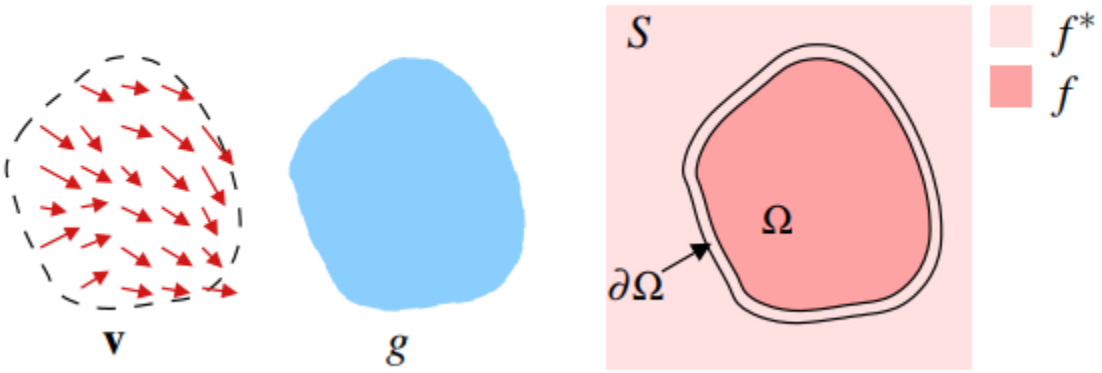


Feature Map 에서 channel 간 correlation 계산

target image Gram Matrix 와 generated image Feature Map Gram Matrix 의 차이를 최소화

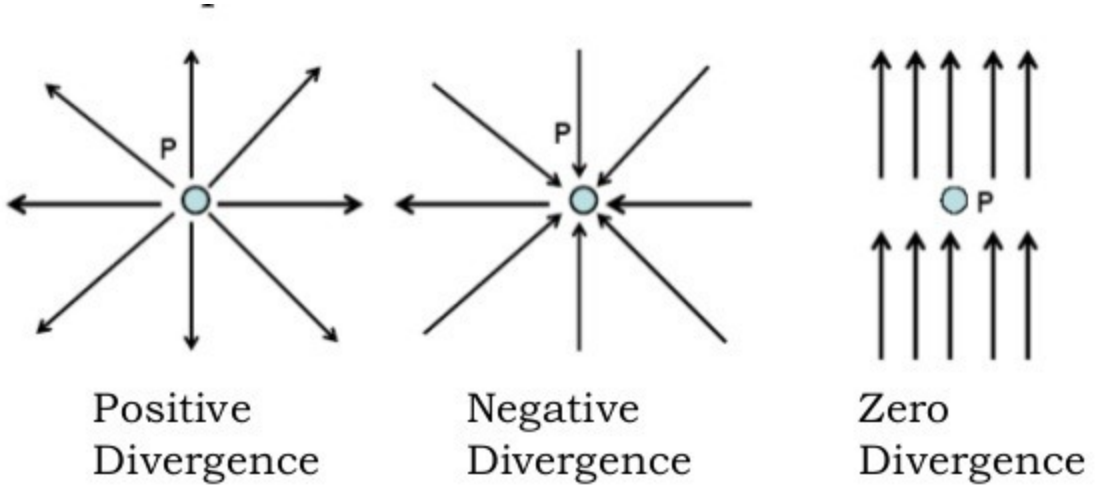
Poisson Image Edit (2003)

이전 세미나 리뷰: 9. Poisson Image Editing, GP-GAN



- unknown source's interpolation function (f)
- target (destination)'s function (f)
- guidance of vector field (v)
- gradient field or not source function (g)
- source image가 target image 의 원하는 영역 (Ω)에 interpolation 하고 싶을 때, 발생하는 문제를 해결 하고자 함.

membrane interpolation (pixel - pixel)



- interpolation function은 경계에서 서로 차이가 없도록 최소화 하는 것을 목표
- Laplacian Operator = Divergence of gradient
  - Divergence of gradient = 0 으로 만들어주면, source - target의 경계가 자연스럽게 모호해진다.

$$\min_f \iint_\Omega |\nabla f|^2 \text{ with } f|_{\partial\Omega} = f^*|_{\partial\Omega}, \tag{1}$$

$$\Delta f = 0 \text{ over } \Omega \text{ with } f|_{\partial\Omega} = f^*|_{\partial\Omega}, \tag{2}$$

"Image Inpaint" (2000) 논문에서 언급, Equation (1), (2)를 그대로 적용하면 gradient가 사라질 수도 있어 guidance 추가 (guidance of vector field)

$$\min_f \iint_\Omega |\nabla f - \mathbf{v}|^2 \text{ with } f|_{\partial\Omega} = f^*|_{\partial\Omega}, \tag{3}$$

$$\Delta f = \text{div}\mathbf{v} \text{ over } \Omega, \text{ with } f|_{\partial\Omega} = f^*|_{\partial\Omega}, \tag{4}$$

$$\Delta \tilde{f} = 0 \text{ over } \Omega, \ \tilde{f}|_{\partial\Omega} = (f^* - g)|_{\partial\Omega}. \tag{5}$$

$$\min_{f|_{\Omega}} \sum_{\langle p,q\rangle \cap \Omega \neq \emptyset} (f_p - f_q - v_{pq})^2, \text{ with } f_p = f_p^*, \text{for all } p \in \partial\Omega, \quad (6)$$

$$\text{for all } p \in \Omega, \quad |N_p| f_p - \sum_{q \in N_p \cap \Omega} f_q = \sum_{q \in N_p \cap \partial\Omega} f_q^* + \sum_{q \in N_p} v_{pq}. \quad (7)$$

$$|N_p| f_p - \sum_{q \in N_p} f_q = \sum_{q \in N_p} v_{pq}. \quad (8)$$

실제 구현은 주변 픽셀을 4-connected neighbor pixel 로 이용합니다.

v pq 는 기본 적으로 guidance field 의 gradient 이지만, seamless cloning 기법을 고려하면 source image - target image 에서 더 큰 픽셀 값을 선택하여 오브젝트가 잘 녹아들게 변경 합니다.

source destination