

Poisson Image Editing, GP-GAN

Paper Title

- Poisson Image Editing (2003) - 3175회 인용
- GP-GAN: Towards Realistic High-Resolution Image Blending (2017, ACMMM 2019 Oral) - 202회 인용

1. Traditional Image Editing

2. Poisson Image Editing

3. GP-GAN

1. Traditional Image Editing

Image Editing (Image Composition)이란?

“Source 이미지를 Target 이미지에 합성”

- 그림 툴을 이용 → 시간과 인력 소모
- 단순히 이미지를 붙여넣는 Copy & Paste → 이미지가 서로 어울리지 않으면 부자연스럽다.

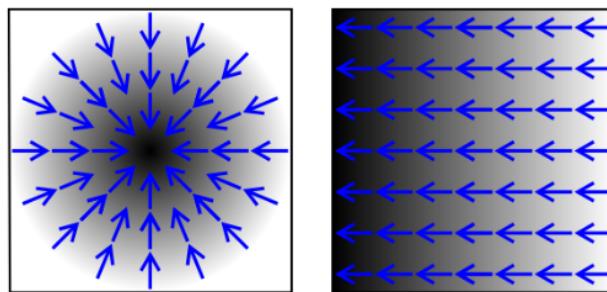
$$x = x_{src} * x_{mask} + x_{dst} * (1 - x_{mask}),$$

기본적으로 사용되는 합성 방법

- Laplacian Pyramid를 이용하여 Source 이미지의 gradient를 강조하여 Mask로 만들고, Target 이미지에 합성 → Target 이미지의 빛, 색상을 고려하지 않아 부자연스러움.



Laplacian Pyramid



2. Poisson Image Editing

Poisson Equation을 이용하는 이유

- image intensity gradient를 작게하여 두 이미지를 자연스럽게 겹치게 만들 수 있다.

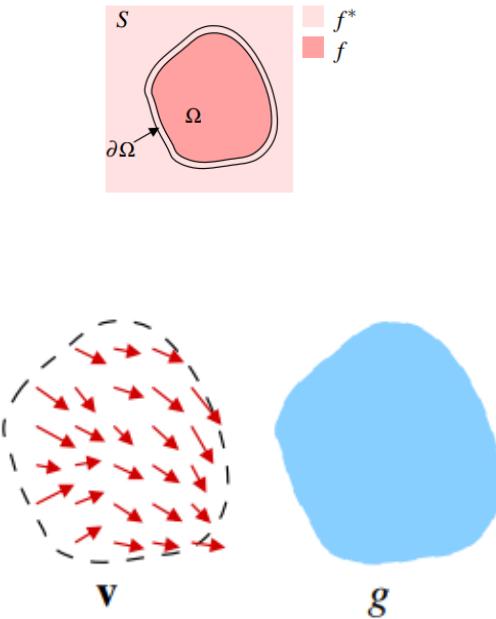
- Domain Boundary는 Domain 내부 Laplacian값으로 유일해로 정의되어 Poisson Equation에서도 유일해를 가질 수 있어 알고리즘적으로 안정적

- 전반적인 컨셉

Target 이미지 안에 Source 이미지를 합성(interpolation)한다고 했을 때,

합성 이미지는 Source 이미지의 Gradient 와 최대한 비슷하게 보이도록 하면서, 경계 $\partial\Omega$ 는 Target 이미지와 같도록 해야합니다. (Reduce Color Mismatch)

Guided Interpolation



Ω : domain,

$\partial\Omega$: boundary domain,

f^* : Destination Function

f : Unknown Function

v : source function g 에 대한 gradient field

본래 Target 이미지내에서 Source 이미지가 들어갈 영역에 대한 부분을 수식으로 표현한다면..

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

Membrane Interpolation, 경계내 픽셀값 정의

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

Laplace's equation with Dirichlet boundary conditions

∇ : gradient operator

Δ : laplacian operator

보간 함수 방법인 (1)에서 오일러-라그랑주 방정식에 따라 정리하면 $\Delta f = 0$ 이 되고,

(2) 처럼 Laplace's equation을 정의할 수 있다.

위의 (1)는 domain이 다른 source 이미지를 넣으려고 할 때를 가정하면 다음처럼 변형됩니다.

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

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

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

(3)에서 \mathbf{v} 는 Source Function에 대한 Gradient Field이고, $\nabla f = \mathbf{v}$ 가 되도록 (3)을 최소한으로 하도록 만족 시킬 수 있습니다.

(3)에 대하여 variation을 계산하면 (4)처럼 Poisson Equation으로 정리됩니다.

(4)는 채널별로 독립적으로 계산되어질 수 있어서, RGB 3채널별로 계산 할 수 있습니다.

\mathbf{v} 는 Source Image의 gradient field이고, g 는 \mathbf{v} 특성을 가진 Source Function일 때,

correction function \tilde{f} 에 대하여 $f = g + \tilde{f}$ 로 치환하여 정리하면 (5)의 Laplace's Equation 형태로 바꿀 수 있습니다.

(5) 수식은 잘 생각해보면 Target Domain 내에서 Source Domain 인지 체크하는 과정

Discrete Poisson Solver

위의 수식들을 이미지내에서는 Pixel 간의 관계로 생각하는 Discrete 문제라고 생각하면

영역 내 픽셀값은 neighbor 픽셀에 의해 Boundary인지 체크가 가능합니다.

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

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

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

수식이 매우 복잡해보이지만... 위의 식들을 Discrete하게 설명하게 바꿨다고 생각하면 됩니다.

f : 픽셀값

p, q : 픽셀,

v_{pq} : p 와 q 의 Gradient 값

$|N_p|$: 픽셀 p 에 대한 4-neighbor 픽셀들

(3)을 변형하면 (6)처럼 정리 할 수 있고,

(8)은 주변에 Boundary가 없는 픽셀에 대하여 정리한 수식

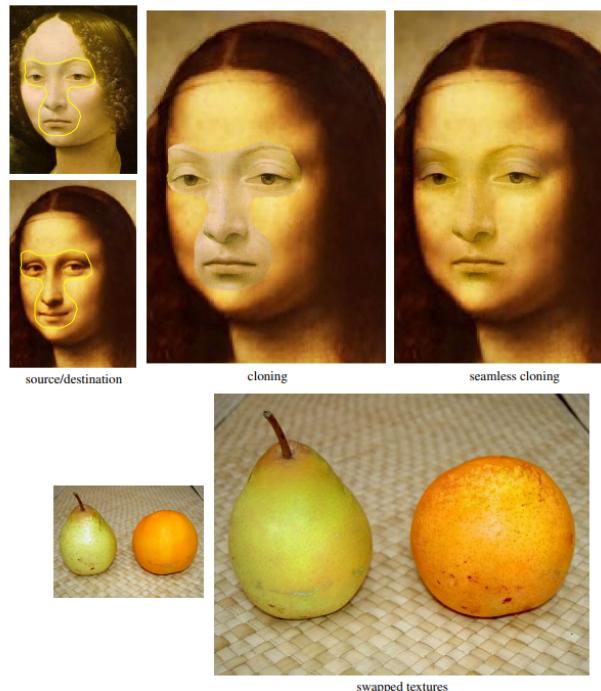
(7)은 주변에 Boundary가 있는 경우, 해당 픽셀에 대하여 4방향 픽셀들을 계산하여 정리한 수식입니다.

결론적으로는 (7), (8)을 프로그램상으로 모든 픽셀에 대해 적용하면, 원하는 합성 결과를 얻을 수 있다고 합니다.

Gradient Field v 를 가지는 Source 이미지를 가지고, 위의 수식을 다양한 변형을 통해 효과를 바꿀 수 있습니다.

Seamless Cloning

(4)에서 v 는 주어진 ∇g 로 치환하여 $v = \nabla g$ 로 바꿀 수 있습니다.

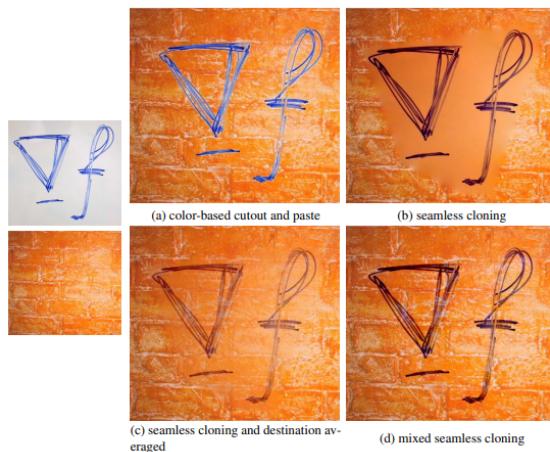


Mixing Gradients

Domain 내에서 Source Domain, Target Domain 값에 대하여 threshold를 줄 수 도 있습니다.

실제 산불 이미지 합성에서도 유의미하게 사용 가능한 부분

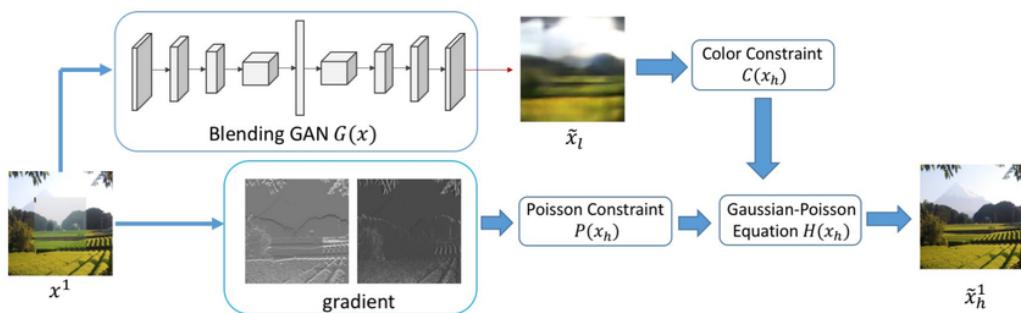
Figs. 6 and 7.





이후로는 색상 변경, 마스크 따는 법 등의 간단한 테크닉들을 소개 했으므로 생략..

3. GP-GAN: Towards Realistic High-Resolution Image Blending



이미지를 생성하면서 gradient filter를 이용하여 색상, 경계를 보정

입력값 x^1 은 Laplacian Pyramid 를 이용해 얻은 Coarest Scale 이미지

GAN 부분은 Conditional GAN에서 컨셉을 가져왔다.

GAN을 이용하여 블러 처리 된 이미지로 Color Constraint 와

Poisson Equation을 이용한 Gradient Constraint 를 더하고 있습니다..

결론적으로는 Gaussian-Laplacian Pyramid 로 치환하여 업샘플링된 이미지로 복원하는 과정으로 요약할 수 있습니다.

(1) Blending GAN

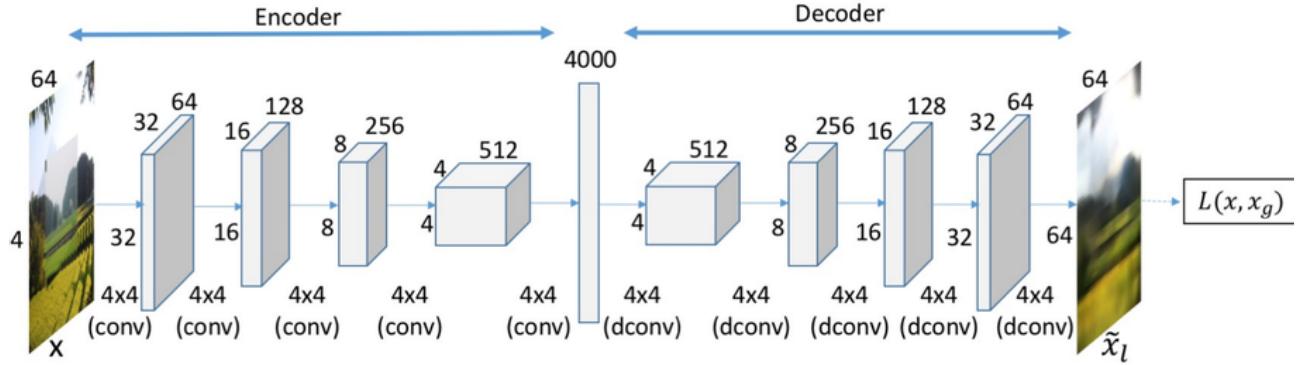


Figure 3: Network architecture of Blending GAN $G(x)$.

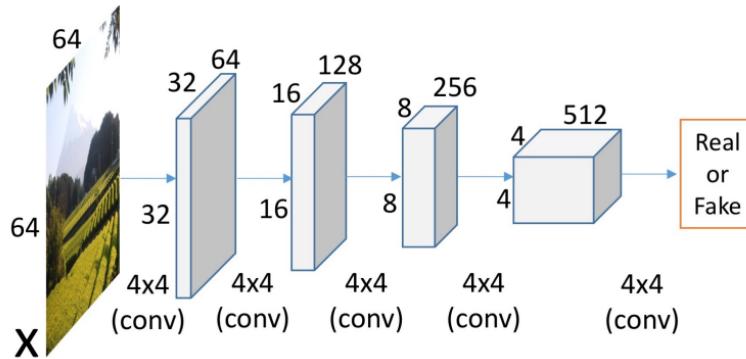


Figure 4: The architecture for the discriminator of Blending GAN.

GAN에서 L1 loss, L2 loss들은 블러링된 이미지를 생성하는 경향이 있고,

perceptual loss는 High Quality 이미지들을 만들 수 있지만 시간이 오래걸리고, 연산량이 크다.

→ L2 Loss 사용 (Loss 별 실험 결과 없음..)

$$L(x, x_g) = \lambda L_{l_2}(x, x_g) + (1 - \lambda) L_{adv}(x, x_g),$$

$$L_{l_2}(x, x_g) = \|G(x) - x_g\|_2^2,$$

$$L_{adv}(x, x_g) = \max_n E_{x \in \chi} [D(x_g) - D(G(x))].$$

Loss 함수 정의 자체는 기존의 GAN 논문들과 다를게 없는 듯 합니다.

(2) Gaussian-Poisson Equation

Blending GAN은 결국 64x64 Low-Resolution 이미지가 생성..

Two Constraint 를 이용하여 High-Resolution 이미지로 변환하고자 함

- Color Constraints
- Gradient Constraints

$$H(x_h) = P(x_h) + \beta C(x_h).$$

$$P(x_h) = \int_T \|\mathbf{div} v - \Delta x_h\|_2^2 dt,$$

$$C(x_h) = \int_T \|g(x_h) - \tilde{x}_l\|_2^2 dt,$$

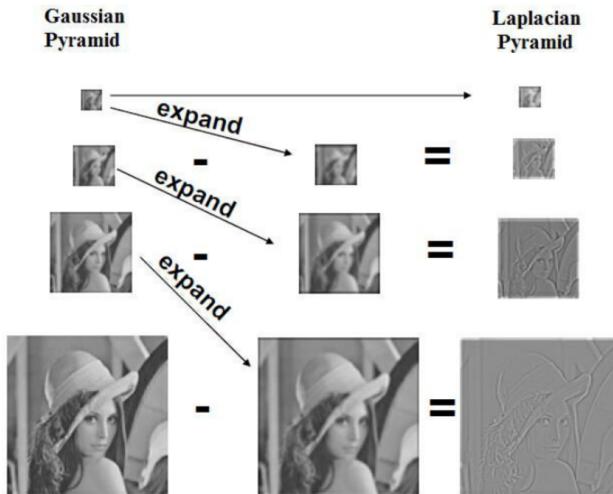
$$v^i = \begin{cases} \nabla x_{src}^i & \text{if } x_{mask}^i = 1 \\ \nabla x_{dst}^i & \text{if } x_{mask}^i = 0 \end{cases},$$

$$H(x_h) = \|u - LX_h\|_2^2 + \lambda \|GX_h - \tilde{X}_l\|_2^2,$$

마지막 수식만 봤을 때,

앞서 Poisson Image Editing에서 나온 것처럼 $P(x_h)$ 는 결국 Laplaces' Equation으로 정리되어

Laplacian Pyramid에 Gaussian Filter를 적용한 이미지를 더해줘서 Upsampling 된 이미지로 복원하는 과정을 표현



Algorithm 1: High-Resolution Image Blending Framework
GP-GAN

Input : Source image x_{src} , destination image x_{dst} , mask image x_{mask} and trained Blending GAN $G(x)$

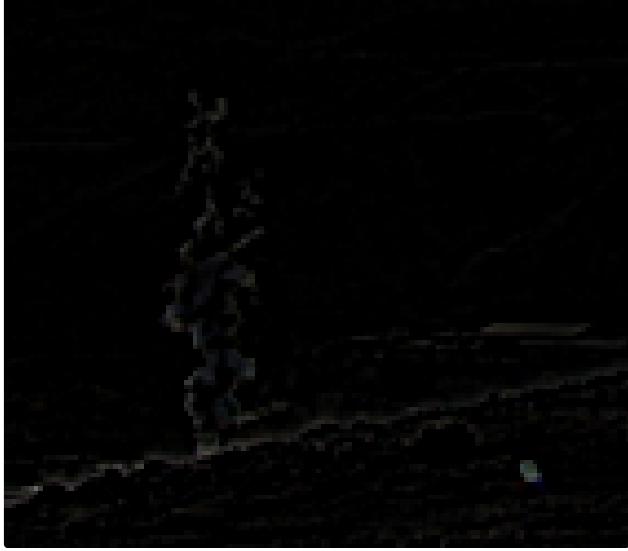
- 1 Compute Laplacian Pyramid for x_{src} , x_{dst} and x_{mask}
- 2 Compute \tilde{x}_l using $G(x)$
- 3 **for** $s \in [1, 2, \dots, S]$ **do**
- 4 Updating x_h^s by optimizing Equation 9 using the closed form solution given x_{src}^s , x_{dst}^s , x_{mask}^s and \tilde{x}_l
- 5 Set \tilde{x}_l to be upsampled x_h^s
- 6 **end**
- 7 Return x_h^S

구현에 사용되는 알고리즘.. 일반적인 Upsample Pyramid를 구하는 알고리즘입니다.

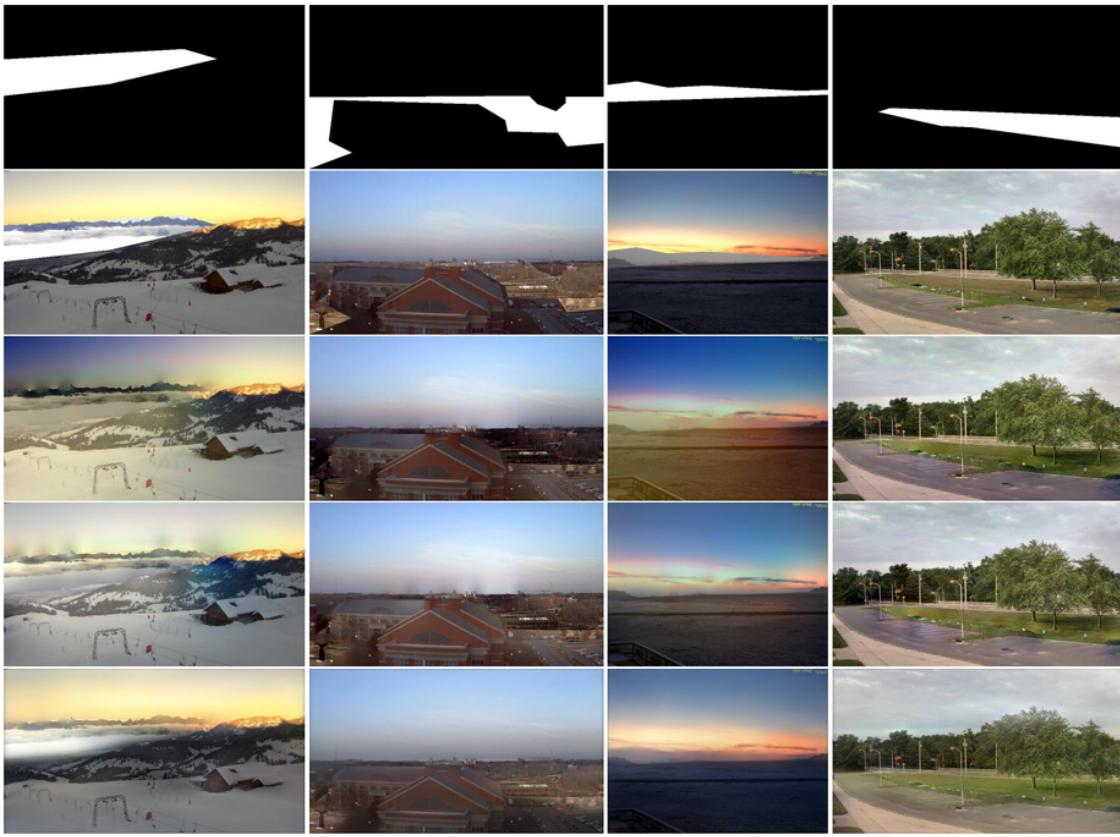


원본





산불 데이터 Laplacian Pyramid 결과



mask, multi-spline blending, Poisson Image Editing, GP-GAN 순서

Table 1: Realism scores for our method and the baselines (higher is better). GP-GAN outperforms all the baselines.

Method	Input	PB[23]	MPB[30]	MSB[28]	Ours
Score	-0.696	-0.192	-0.151	-0.140	-0.069

Table 2: User study result. 4 image blending algorithms are compared on Amazon Mechanical Turk. Our method GP-GAN obtains most votes, which is consistent with the result of the realism scores.

Method	Total votes	Average votes	Std.
PB[23]	527	1.054	1.065
MPB[30]	735	1.470	1.173
MSB[28]	770	1.540	1.271
GP-GAN	947	1.894	1.311

평가 방법은!!! 놀랍게도... Vote 방식!!!!

- 데이터셋

Transient Attributes Database

101개 웹캠으로 부터 얻은 8,571장

2장씩 랜덤으로 골라서 GP-GAN 적용

해당 이미지들로부터 생성한 이미지는 150K

GAN 평가 지표

해당 논문에서는 적용하지 않았지만,

GAN에서 사용되는 평가 지표로는 IS (Inception Score), FID (Frechet Inception distance) 가 있습니다.

생성 이미지가 잘 생성 되었는지 판단하려면

1. 충실도(Fidelity) : 고품질의 데이터여야 한다.
2. 다양성(Diversity) : 학습 데이터들의 특성을 반영하여 잘 생성해야 한다.

이 두 가지를 평가하기 위한 지표는 두 평가 지표 모두 원본, 생성 이미지간 분포의 거리를 계산하여 구합니다.

두 평가 지표는 Inception Model로 구현하여 Score를 얻습니다.

IS

→ K-L D, 주로 품질에 대한 평가

FID

→ Frechet Distance, noise robust하고, 단일 클래스보다는 다양한 클래스에서 효과적으로 평가