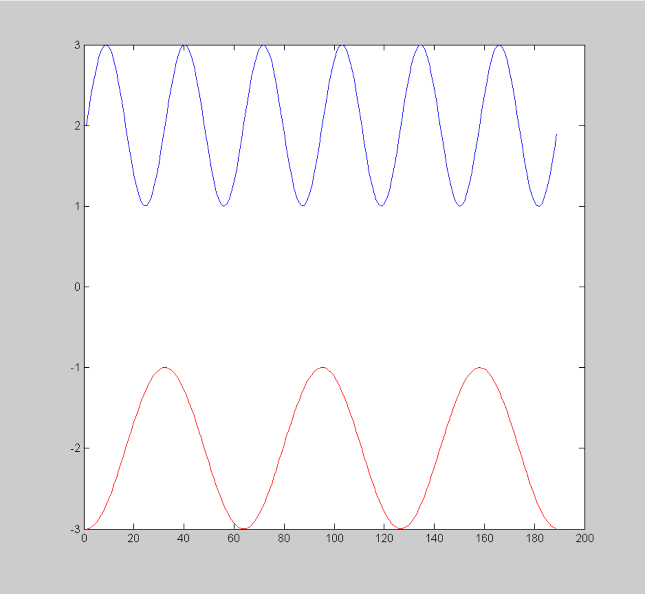
Gradient Domain blending (1D)

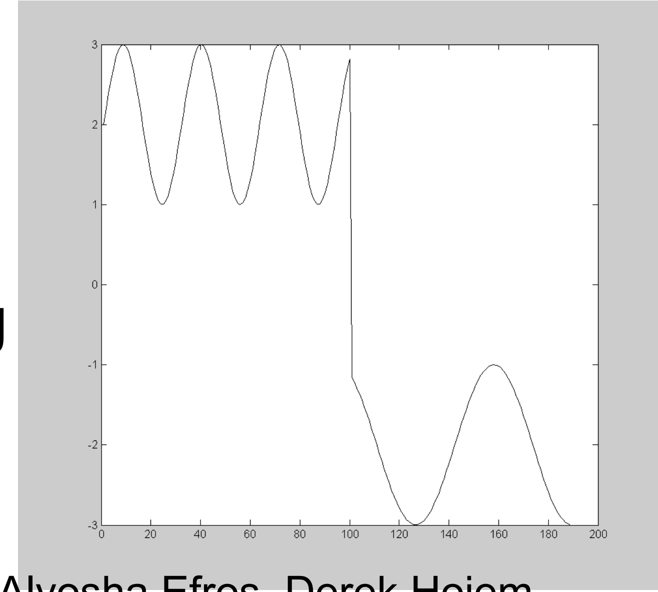
Two signals

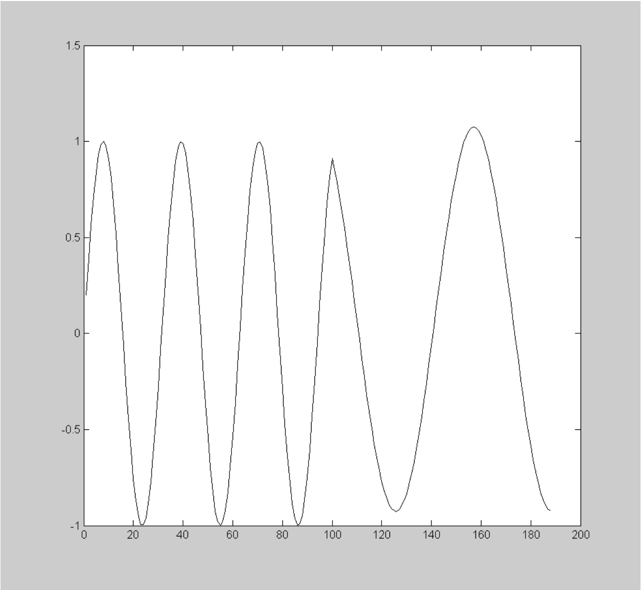


bright

dark

Regular blending

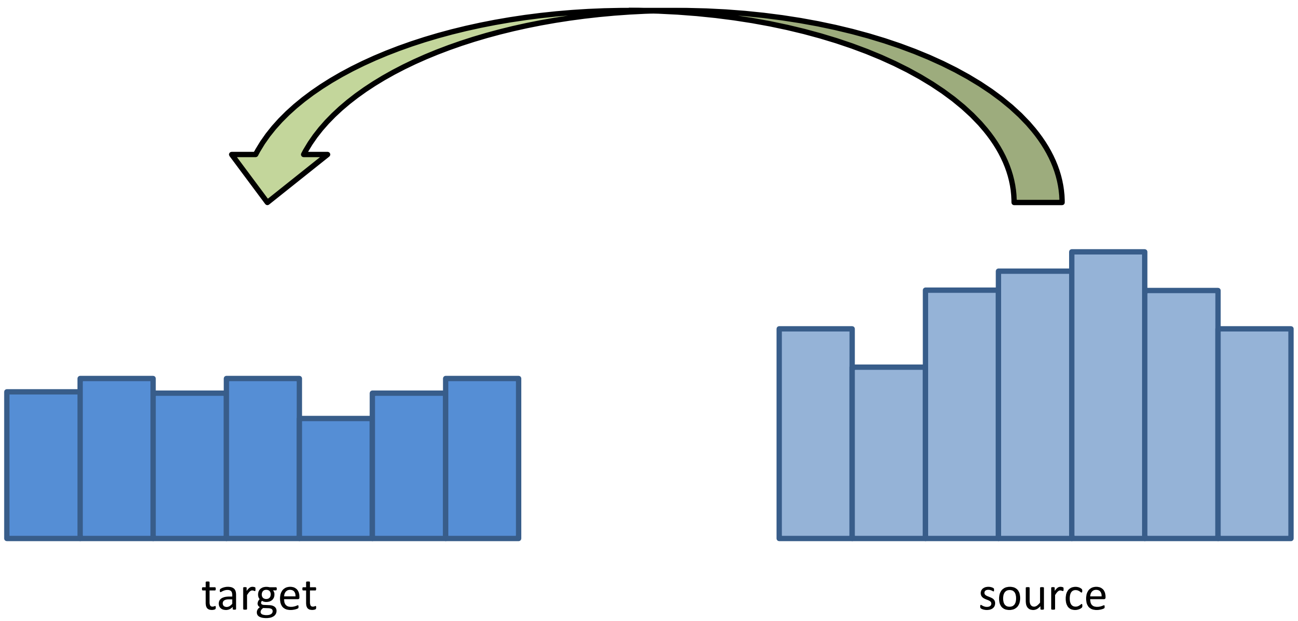




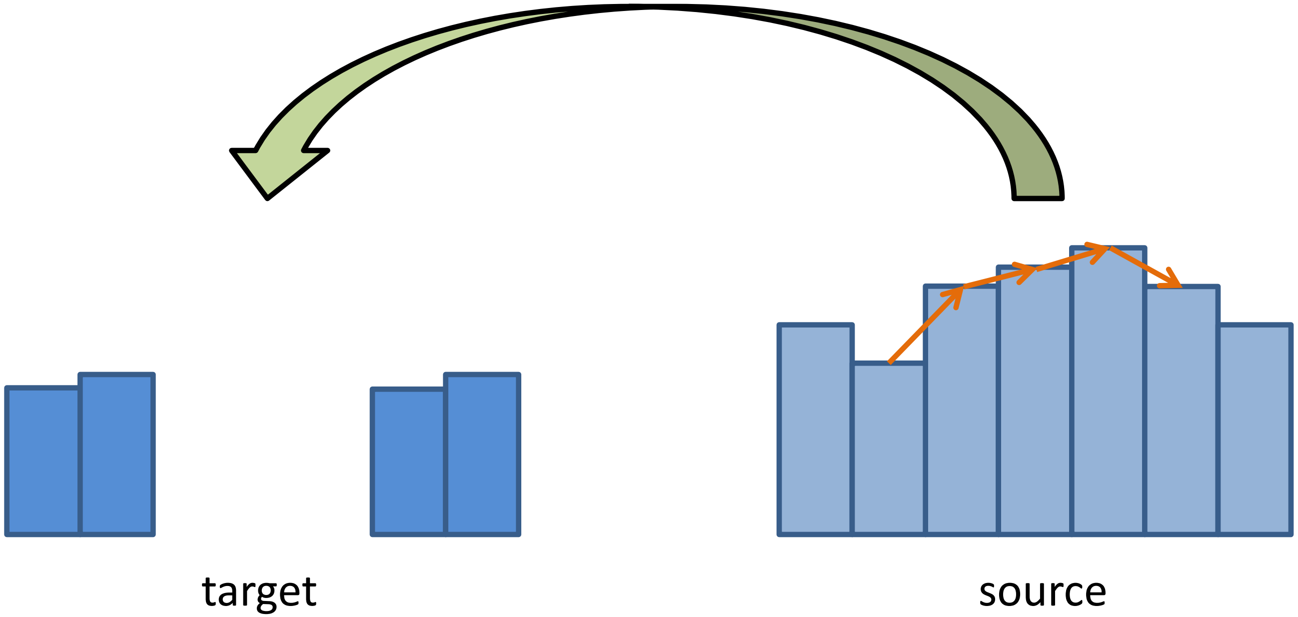
Blending derivatives

Slide credit: Alyosha Efros, Derek Hoiem

Gradient hole-filling (1D)



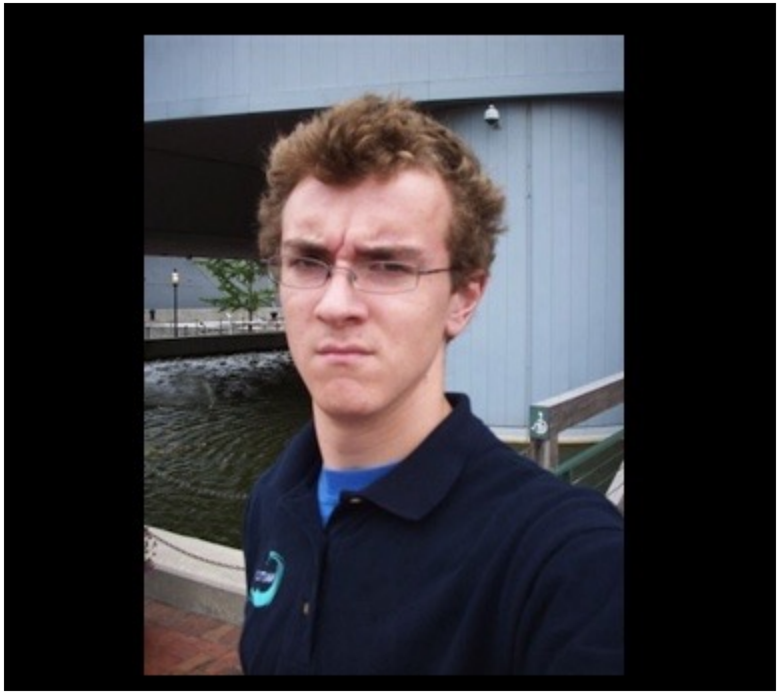
Slide credit: Alyosha Efros, Derek Hoiem



It is impossible to faithfully preserve the gradients

Slide credit: Alyosha Efros, Derek Hoiem

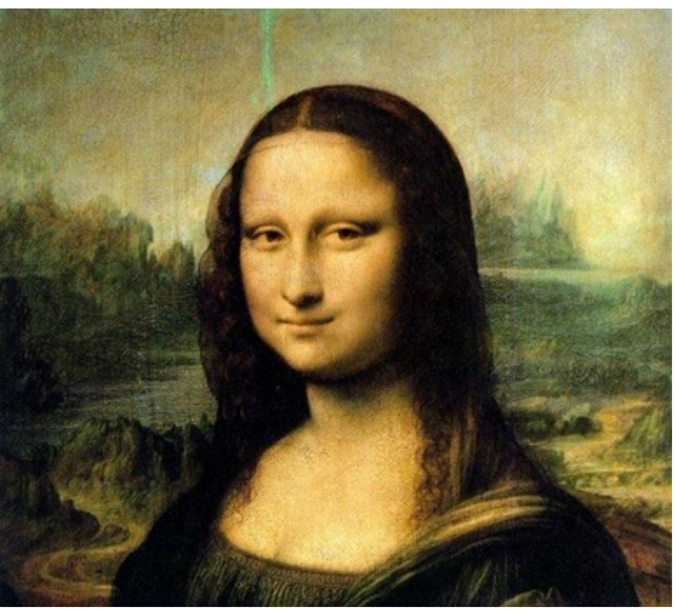
Example



Gradient Visualization

Slide credit: Alyosha Efros, Derek Hoiem

Source: Evan Wallace



Specify object region

Slide credit: Alyosha Efros, Derek Hoiem

Source: Evan Wallace

Poisson Blending Algorithm

A good blend should preserve gradients of source region without changing the background

Treat pixels as variables to be solved

- Minimize squared difference between gradients of foreground region and gradients of target region
- Keep background pixels constant

Target (background)

$$\mathbf{v} = \arg\min_{\mathbf{v}} \sum_{i \in S, j \in N_i \cap S} ((v_i - v_j) - (s_i - s_j))^2 + \sum_{i \in S, j \in N_i \cap \neg S} ((v_i - t_j) - (s_i - s_j))^2$$

Output

Source (foreground)

i	current pixel's index
N_i	Current pixel's neighbors
j	neighbor pixel index
$S \quad \neg S$	foreground/background mask

v: output pixels

s: source pixels

t: background (target) pixels

Slide credit: Alyosha Efros, Derek Hoiem

Perez et al. 2003

Examples

Gradient domain processing

Target (background)

$$\mathbf{v} = \arg\min_{\mathbf{v}} \sum_{i \in S, j \in N_i \cap S} ((v_i - v_j) - (s_i - s_j))^2 + \sum_{i \in S, j \in N_i \cap \neg S} ((v_i - t_j) - (s_i - s_j))^2$$

Output

Source (foreground)

source image

1 **20**	5 **20**	9 **20**	13 **20**
2 **20**	6 **80**	10 **20**	14 **20**
3 **20**	7 **20**	11 **80**	15 **20**
4 **20**	8 **20**	12 **20**	16 **20**

background image

1 **10**	5 **10**	9 **10**	13 **10**
2 **10**	6 **10**	10 **10**	14 **10**
3 **10**	7 **10**	11 **10**	15 **10**
4 **10**	8 **10**	12 **10**	16 **10**

target image

1 **10**	5 **10**	9 **10**	13 **10**
2 **10**	6 $\mathbf{v_1}$	10 $\mathbf{v_3}$	14 **10**
3 **10**	7 $\mathbf{v_2}$	11 $\mathbf{v_4}$	15 **10**
4 **10**	8 **10**	12 **10**	16 **10**

e.g., pixel v_1

left, top

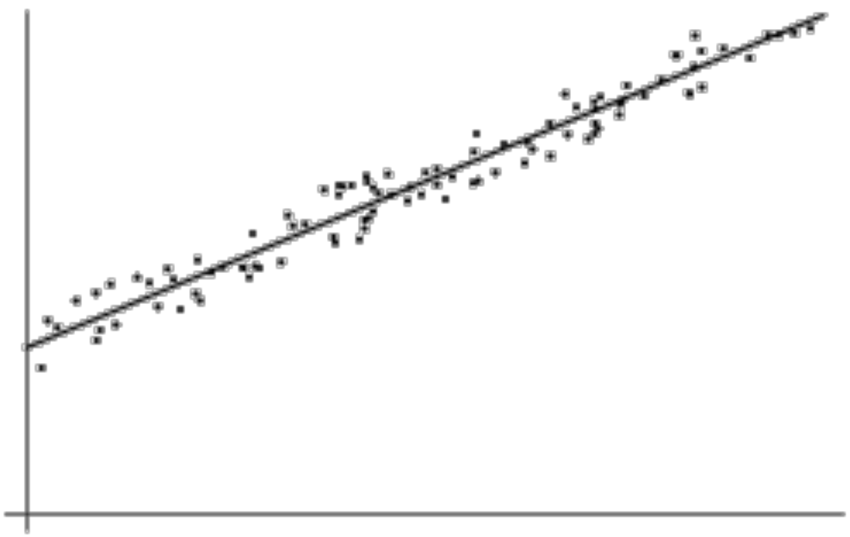
$$\big((v_1 - 10) - (80 - 20)\big)^2 + \big((v_1 - 10) - (80 - 20)\big)^2$$

right, bottom

$$((v_1 - v_3) - (80 - 20))^2 + ((v_1 - v_2) - (80 - 20))^2$$

Gradient-domain editing

Creation of image = least squares problem in terms of: 1) pixel intensities; 2) differences of pixel intensities



Least squares line fitting in 2 Dimensions

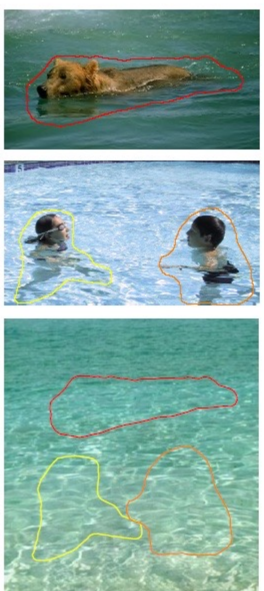
$$\hat{\mathbf{v}} = \arg\min_{\mathbf{v}} \sum_i \left(\mathbf{a}_i^T \mathbf{v} - b_i\right)^2$$

$$\hat{\mathbf{v}} = \arg\min_{\mathbf{v}} \left(\mathbf{A}\mathbf{v} - \mathbf{b}\right)^2$$

Use sparse linear equation solver in Python and MATLAB

Perez et al., 2003

sources destinations cloning seamless cloning

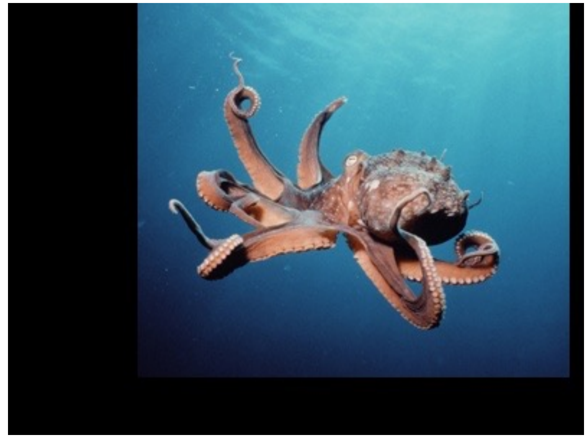


sources/destinations

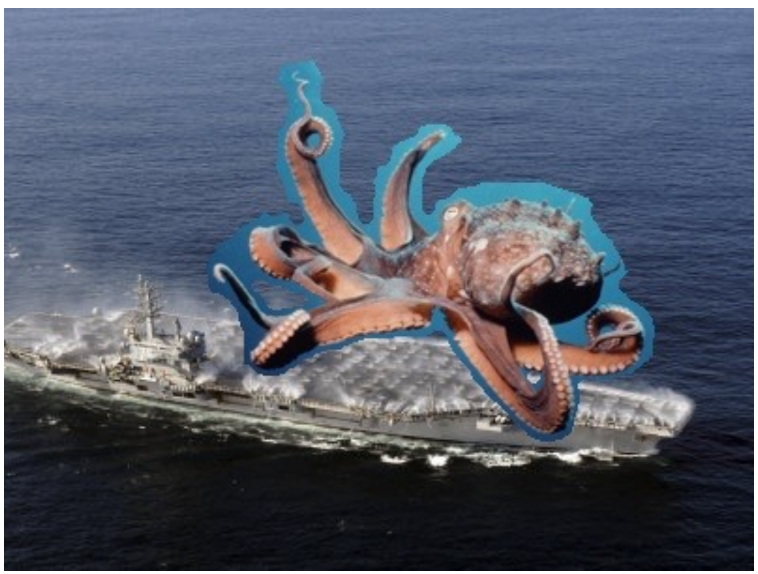
cloning

seamless cloning


target


source


mask

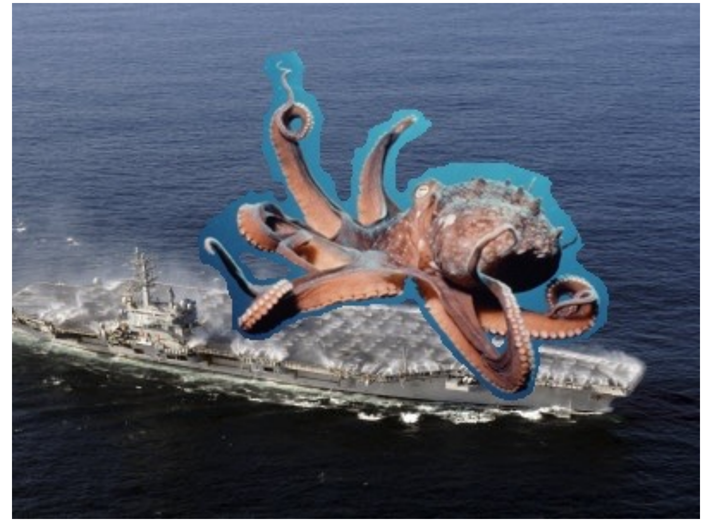

no blending


gradient domain blending

Slide by Mr. Hays

What's the difference?

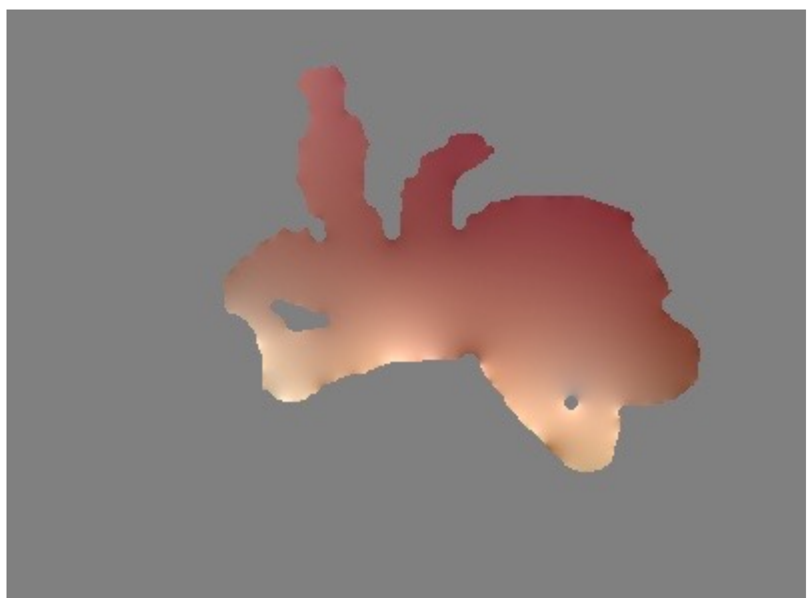
-



=

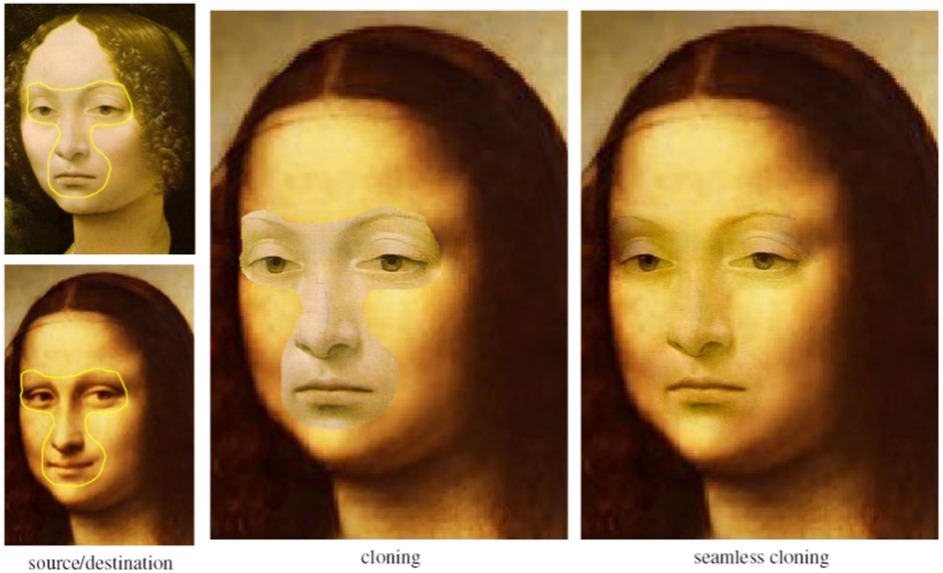
gradient domain blending

no blending



Slide by Mr. Hays

Perez et al, 2003



Local color changes

Limitations:

- Can’t do contrast reversal (gray on black -> gray on white)
- Colored backgrounds “bleed through”
- Images need to be very well aligned

Drawing in Gradient Domain

Real-Time Gradient-Domain Painting

James McCann*
Carnegie Mellon University

Nancy S. Pollard†
Carnegie Mellon University

James McCann & Nancy Pollard

Real-Time Gradient-Domain Painting,

SIGGRAPH 2009

(CMU paper)

Drawing in Gradient Domain

James McCann & Nancy Pollard

Real-Time Gradient-Domain Painting,

SIGGRAPH 2009