Gradient-based Editing

David Capel

These slides borrow from Bill Freeman and Frédo Durand’s PPT:

http://groups.csail.mit.edu/graphics/classes/CompPhoto06/html/lecturenotes/10_Gradient.ppt

And images from

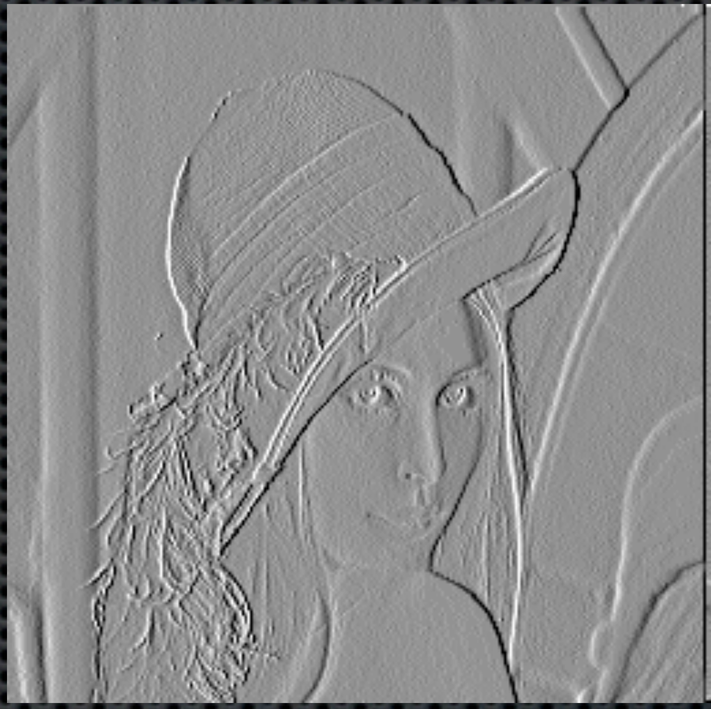
Perez et al. *“Poisson Image Editing”*
SIGGRAPH 2003

Goal: Splice together parts from several images

Naive cut-and-paste leaves nasty seams!

- Human eye is very sensitive to edges and contrast
- Edges contain much of the “salient” structure in an image
- Image gradients capture edge structure quite well

f



$\frac{df}{dx}$

$\frac{df}{dy}$

- Human eye is very sensitive to edges and contrast
- Edges contain much of the “salient” structure in an image
- Image gradients capture edge structure quite well

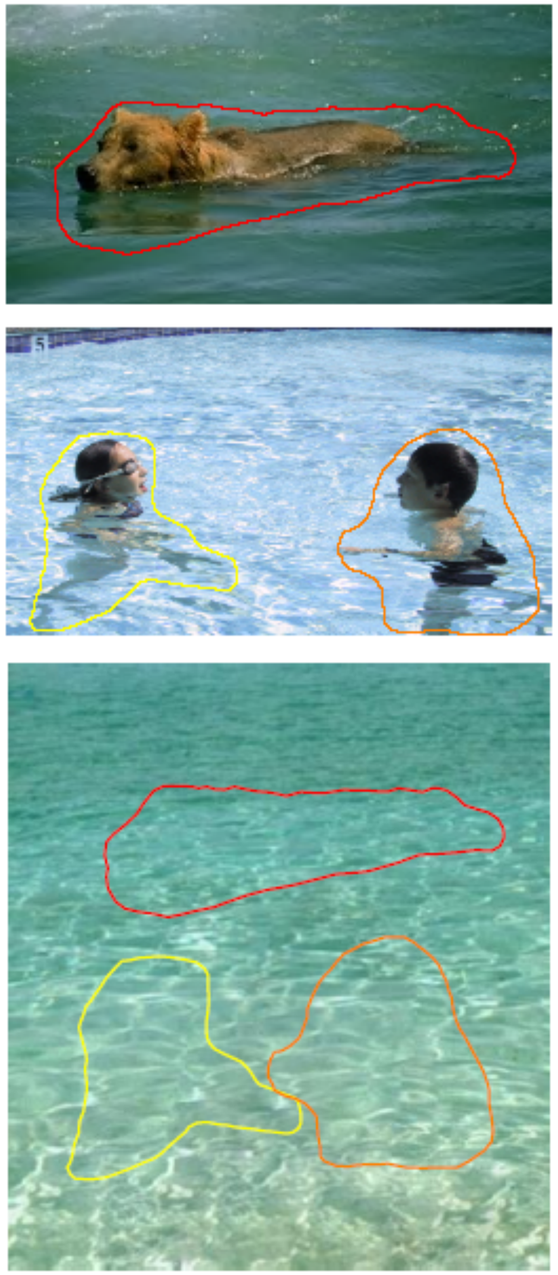
f

$\frac{df}{dx}$

$\frac{df}{dy}$

Idea!

- Do blending or splicing of the **image gradients**
- Then reconstruct image from the blended gradients



sources/destinations

pasted color gradients

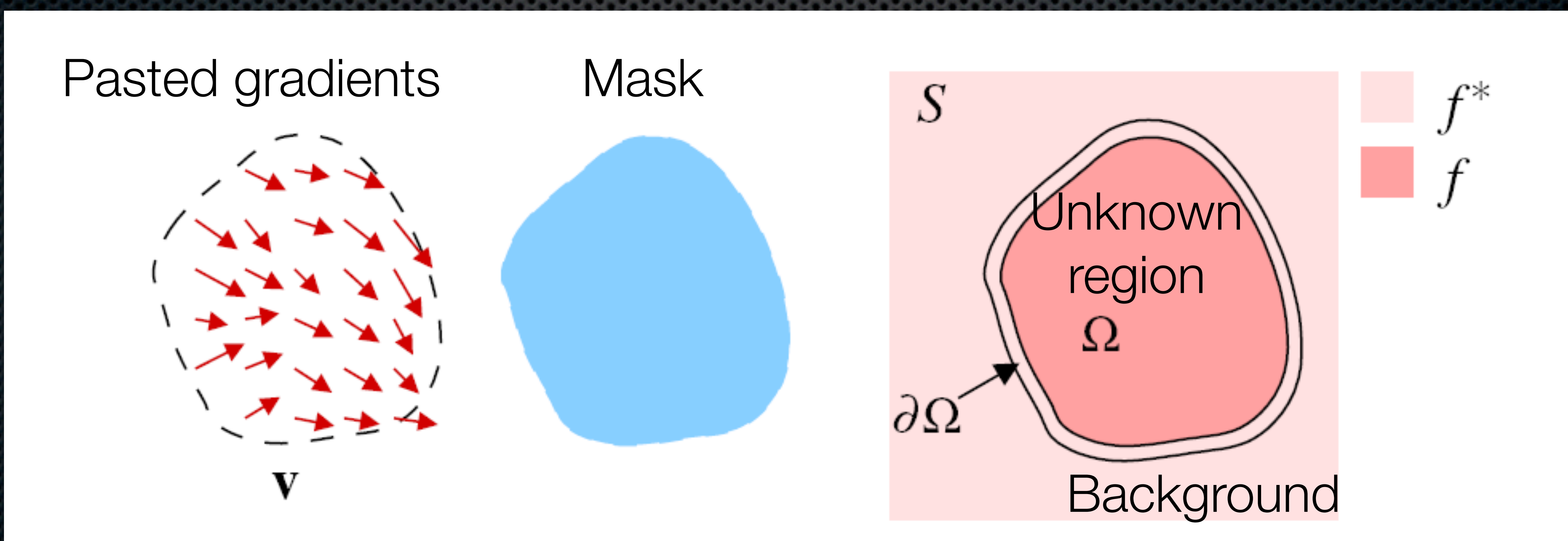
seamless cloning

Formulation as a minimization problem ...

- Compute image **f** that is consistent with pasted gradient vector field **v** in least-squares sense

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

(color equality at region boundary)



Aside for the mathematically inclined

Minimization of this energy functional ...

$$\begin{cases} \min_f \iint_\Omega \|\nabla f - \mathbf{v}\|^2 \\ f|_{\partial\Omega} = f^*|_{\partial\Omega} \end{cases}$$

... corresponds to Poisson's equation with Dirichlet boundary conditions:

$$\begin{cases} \nabla^2 f = \nabla \cdot \mathbf{v} \\ f|_{\partial\Omega} = f^*|_{\partial\Omega} \end{cases}$$

PDE for steady-state wave and heat problems

Hence the name : **Poisson Blending**

Discretization

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

$$\min_{f|\Omega} \sum_{<p,q>\cap\Omega\neq\emptyset} (f_p - f_q - v_{pq})^2 \text{ with } f_p = f_p^* \text{ for } \forall p \in \partial\Omega$$

Discrete gradient — Desired gradient — Boundary constraint

Differentiate and rearrange ...

$$\text{for } \forall p \in \Omega,\ |N_p|\, f_p - \sum_{q\in N_p\cap\Omega} f_q = \sum_{q\in N_p\cap\partial\Omega} f_q^* + \sum_{q\in N_p} v_{pq}$$

where N_p = neighbors of pixel p

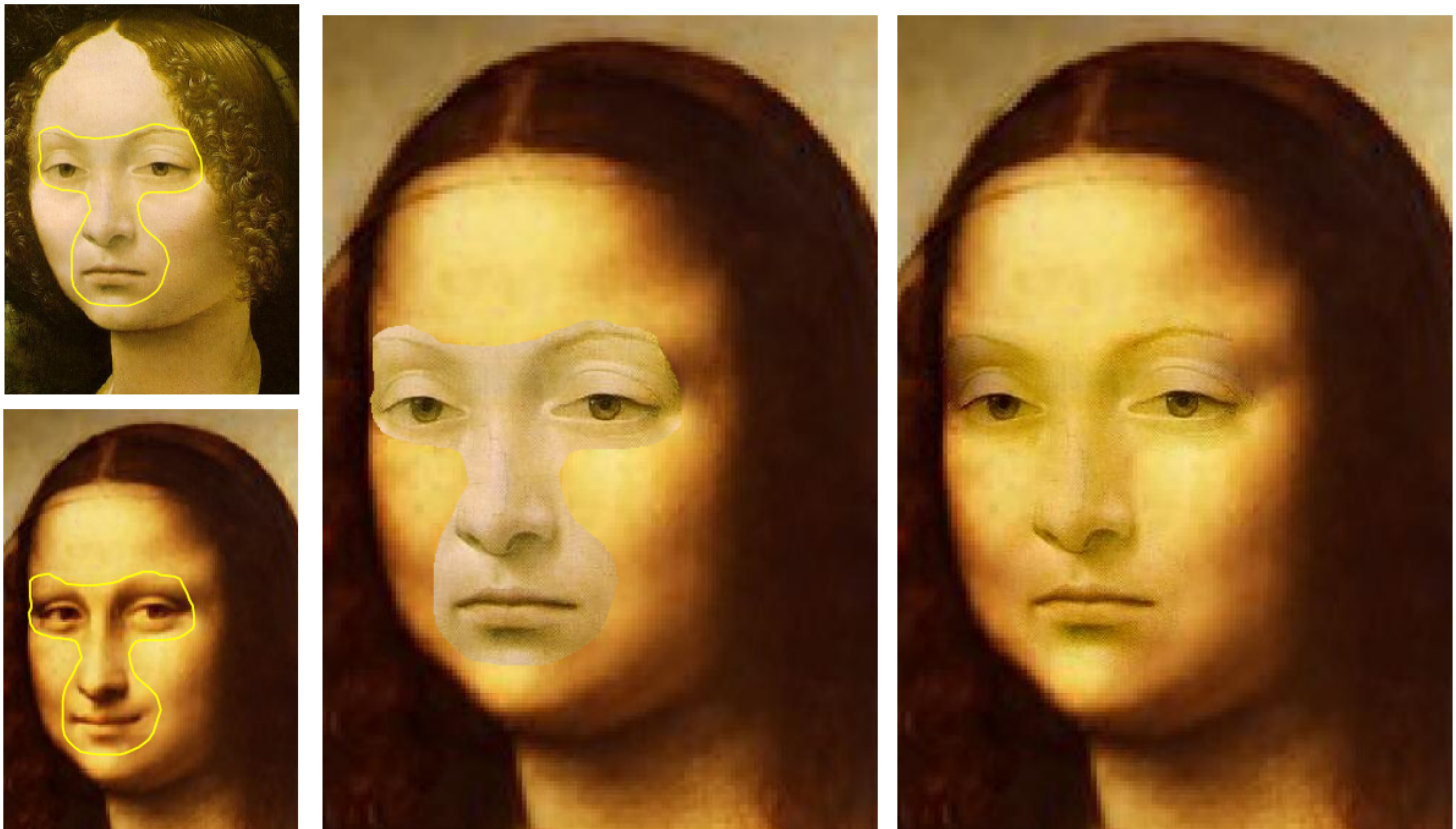
Solving for the image **f**

$$\text{for } \forall p \in \Omega,\ |N_p|\, f_p - \sum_{q \in N_p \cap \Omega} f_q = \sum_{q \in N_p \cap \partial\Omega} f_q^* + \sum_{q \in N_p} v_{pq}$$

$$\begin{bmatrix} ..\text{-}1..\text{-}1\ \ 4\ \ \text{-}1..\text{-}1.... \\ ....\text{-}1..\text{-}1\ \ 4\ \ \text{-}1..\text{-}1.. \\ \cdots \\ \cdots \end{bmatrix} \begin{bmatrix} f_0 \\ f_1 \\ . \\ . \\ f_N \end{bmatrix} = \begin{bmatrix} \sum_{q \in N_p \cap \partial\Omega} f_q^* + \sum_{q \in N_p} v_{pq} \\ \\ \cdots \\ \cdots \end{bmatrix}$$

- A large, sparse quadratic problem (5 non-zeros per row)
- Can just solve RGB channels as 3 separate problems
- Easy to solve in Matlab
  - Conjugate gradients `f = pcg(A,b)`
  - Backslash operator `f = A\b`

Poisson Blending Results



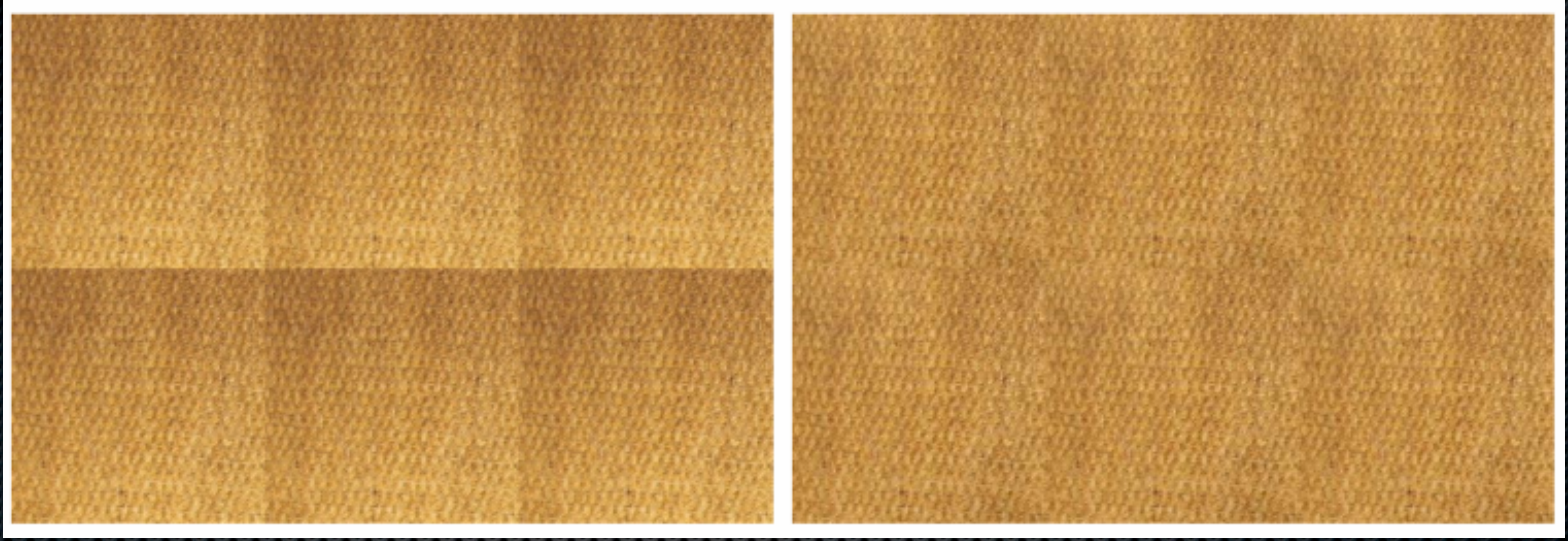
source/destination cloning seamless cloning

sources destinations cloning seamless cloning

“Clone brushing” from within the same image

Figure 2: **Concealment**. By importing seamlessly a piece of the background, complete objects, parts of objects, and undesirable artifacts can easily be hidden. In both examples, multiple strokes (not shown) were used.

Creating tileable textures



- Call original tile **g**
- Boundary conditions
  - $f^*_{north} = f^*_{north} = 0.5(g_{north} + g_{south})$
  - $f^*_{east} = f^*_{west} = 0.5(g_{east} + g_{west})$

Beyond cut-and-paste: **mixing gradients**

Idea: At each pixel, choose stronger of source or destination image gradient

$$\text{for all } \mathbf{x} \in \Omega,\ \mathbf{v}(\mathbf{x}) = \begin{cases} \nabla f^*(\mathbf{x}) & \text{if } |\nabla f^*(\mathbf{x})| > |\nabla g(\mathbf{x})| \\ \nabla g(\mathbf{x}) & \text{otherwise.} \end{cases}$$

Discretized version:

$$v_{pq} = \begin{cases} f_p^* - f_q^* & \text{if } |f_p^* - f_q^*| > |g_p - g_q| \\ g_p - g_q & \text{otherwise,} \end{cases}$$

Mixed gradient results

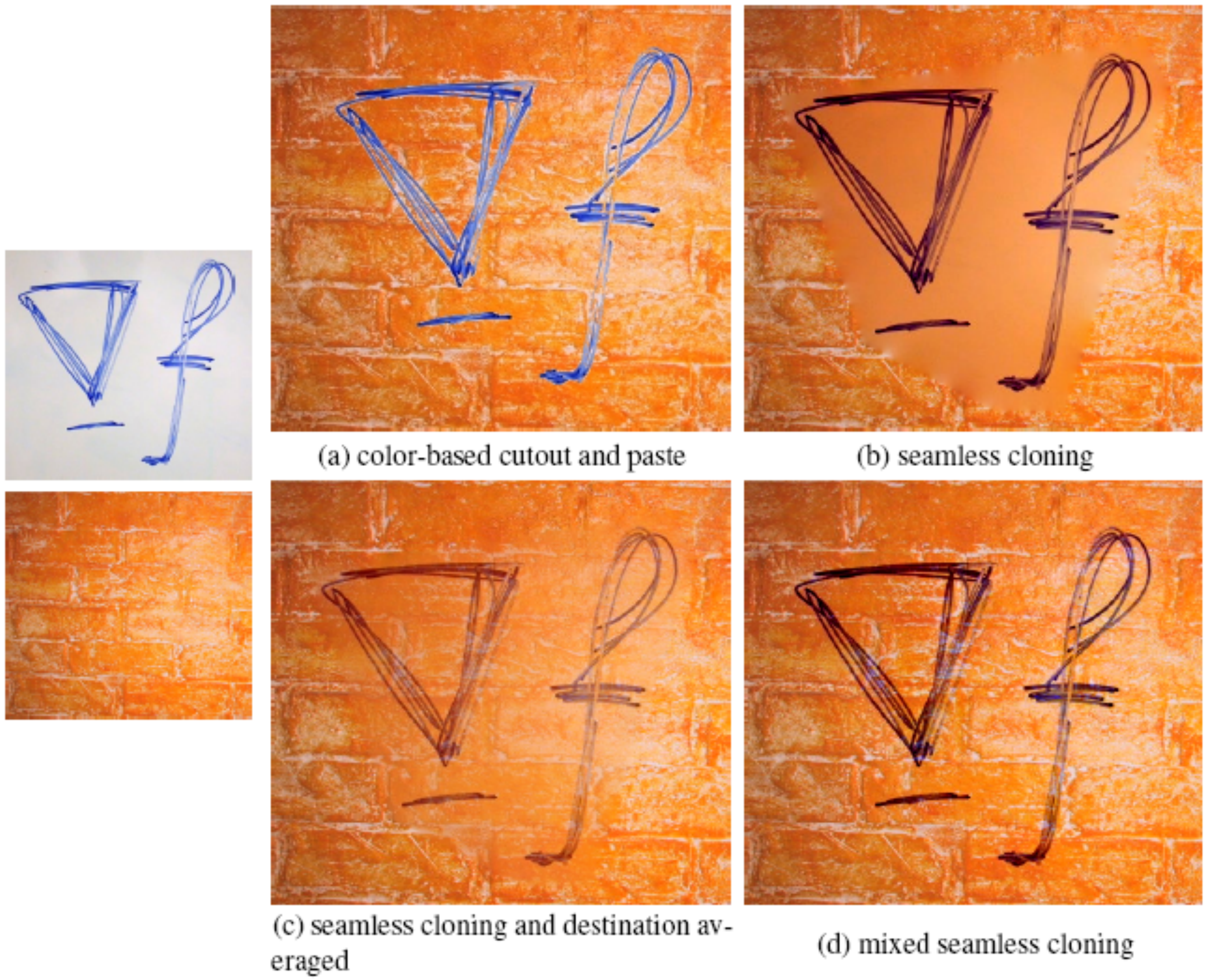


Figure 6: **Inserting objects with holes**. (a) The classic method, color-based selection and alpha masking might be time consuming and often leaves an undesirable halo; (b-c) seamless cloning, even averaged with the original image, is not effective; (d) mixed seamless cloning based on a loose selection proves effective.

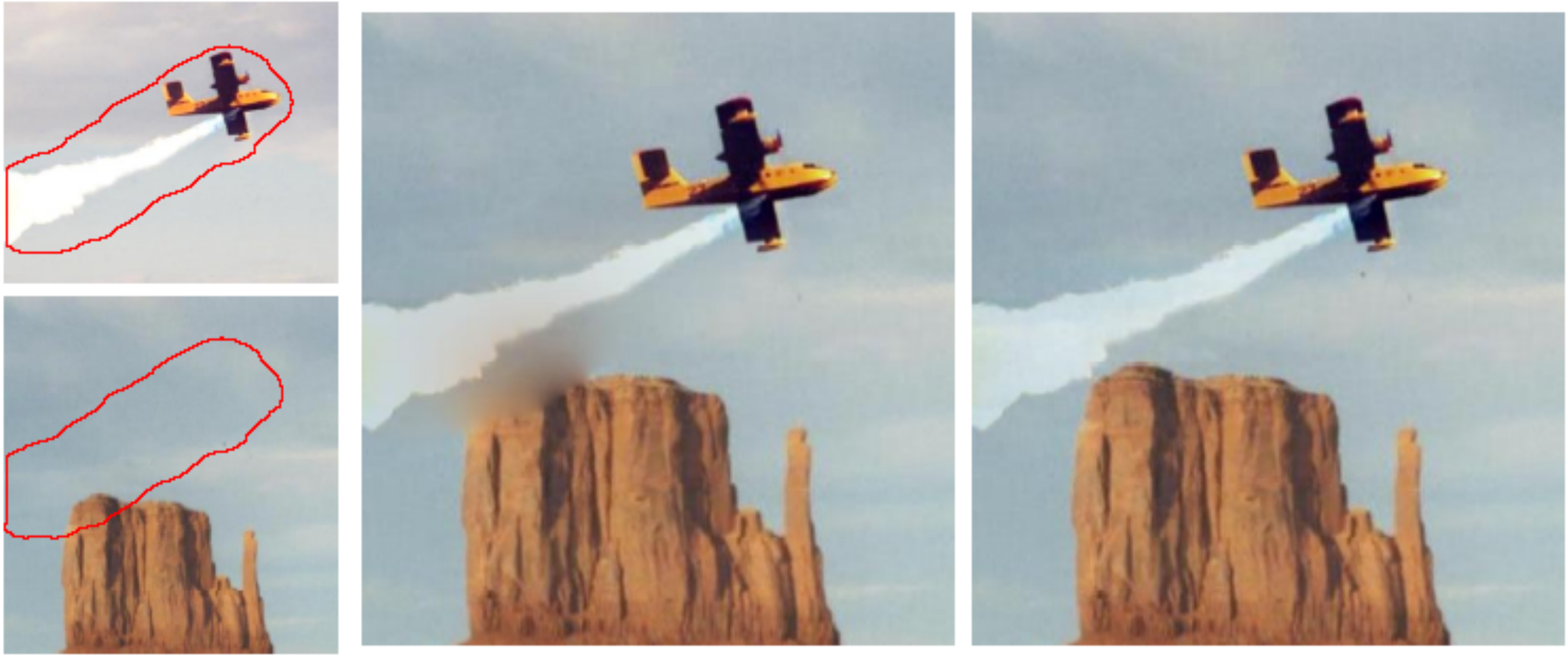


Figure 8: **Inserting one object close to another**. With seamless cloning, an object in the destination image touching the selected region Ω bleeds into it. Bleeding is inhibited by using mixed gradients as the guidance field.

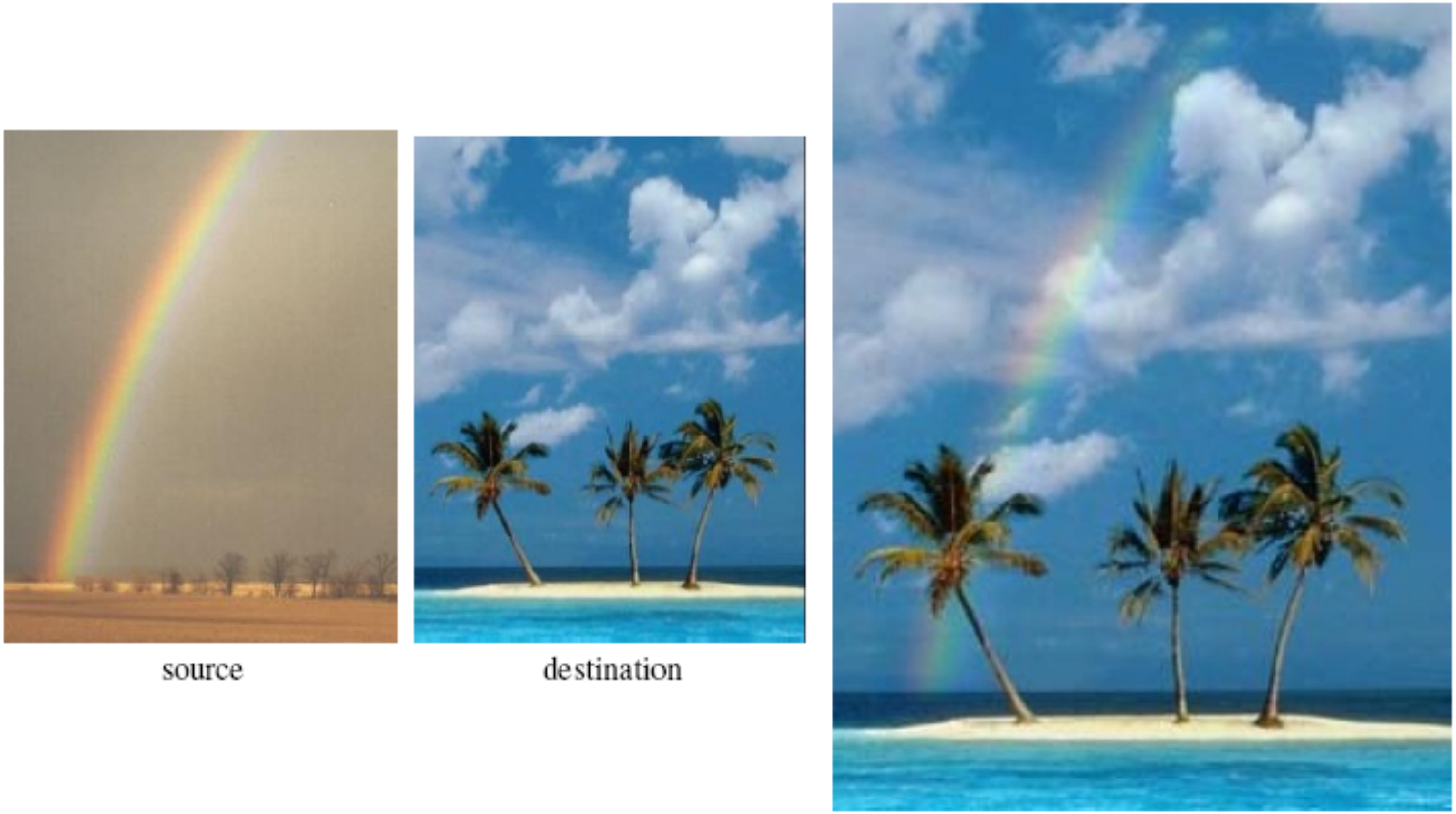


Figure 7: **Inserting transparent objects**. Mixed seamless cloning facilitates the transfer of partly transparent objects, such as the rainbow in this example. The non-linear mixing of gradient fields picks out whichever of source or destination structure is the more salient at each location.

Image flattening - nice "cartoon" effect

Idea: Eliminate gradients below a threshold

$$v_{pq} = \begin{cases} f_p - f_q & \text{if an edge lies between } p \text{ and } q \\ 0 & \text{otherwise,} \end{cases}$$

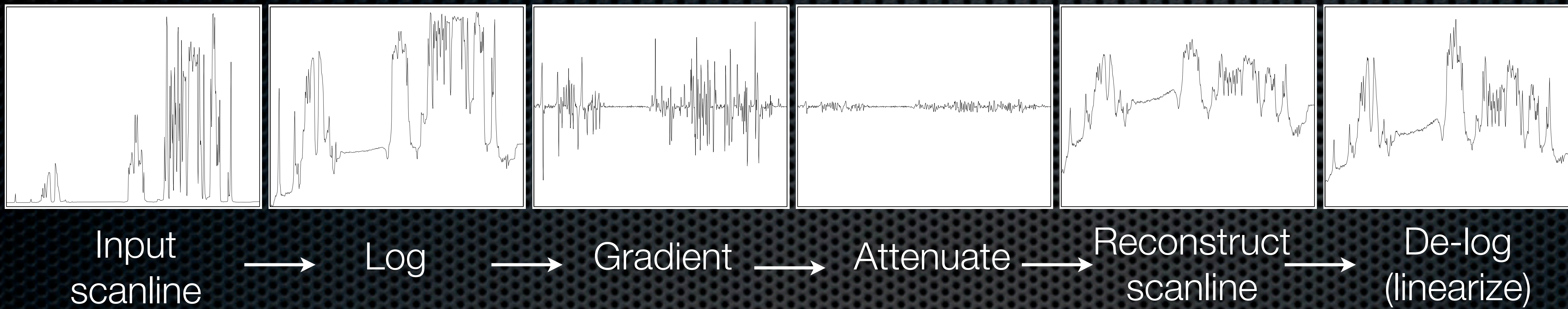
Texture transfer

Idea: Paste *luminance* gradients from one object onto another

swapped textures

“Gradient-Domain High Dynamic Range Compression”
Fattal et al. SIGGRAPH 2002

Idea: Attenuate strong gradients, reconstruct image



Attenuation factors