# Image Editing and Compositing

**Frédo Durand**
**MIT - EECS**

# Interactive Digital Photomontage

- Aseem Agarwala, Mira Dontcheva, Maneesh Agrawala, Steven Drucker, Alex Colburn, Brian Curless, David Salesin, Michael Cohen. Interactive Digital Photomontage. ACM Transactions on Graphics (Proceedings of SIGGRAPH 2004), 2004.
- Set of aligned images of same scene
- Combine in clever ways
  - automatic or user-specified
- More about the exact combination next time.

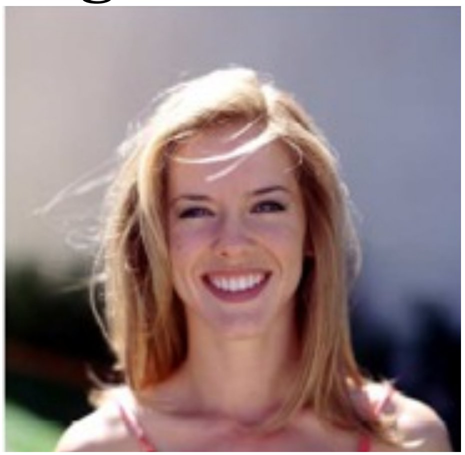
# Montage

# Digital photomontage

# Segmentation & compositing

- **Segmentation / Selection / Matting:**
  **Separate a foreground object from a background**

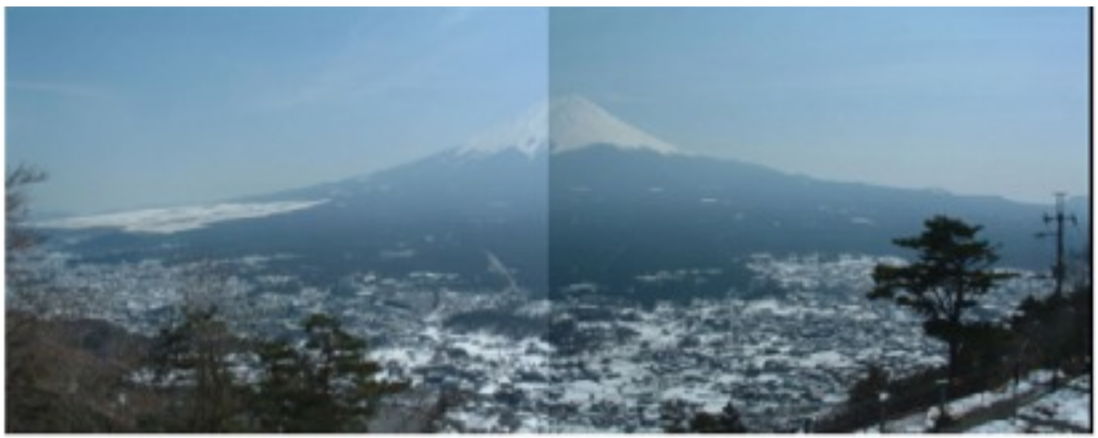

- **Compositing: Paste an image region seamlessly**




# Why segmentation & compositing?

- Special effects
- Clean background
- Wire / distractor removal
- Magazine cover
- Hide skin defects
- Panorama stitching





# Photo editing

- **Edit the background independently from foreground**

# Photo editing

- **Edit the background independently from foreground**

# Healing brush demo

# Two strategies

- **Smart extraction**
  - Blue/green screen
  - Intelligent scissors
  - Snakes
  - Graph cut
  - Matting
- **Smart compositing**
  - Poisson

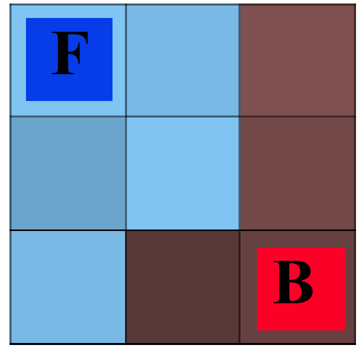
- **We focus on single-image solutions**

# Graph Cut

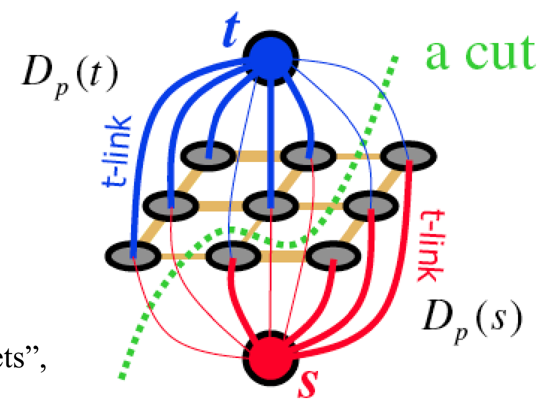
**Frédo Durand**
**MIT - EECS**

# Graph cut overview

- **Interactive image segmentation using graph cut**
- **Binary label: foreground vs. background**
- **User labels some pixels**
- **Exploit**
  - Statistics of known Fg & Bg
  - Smoothness of label
- **Turn into discrete graph optimization**
  - Graph cut (min cut / max flow)




| | | |
|---|---|---|
| F | F | B |
| F | F | B |
| F | B | B |




**Images from**
European Conference on Computer Vision 2006 : "Graph Cuts vs. Level Sets",
Y. Boykov (UWO), D. Cremers (U. of Bonn), V. Kolmogorov (UCL)

# Refs

- **Combination of**

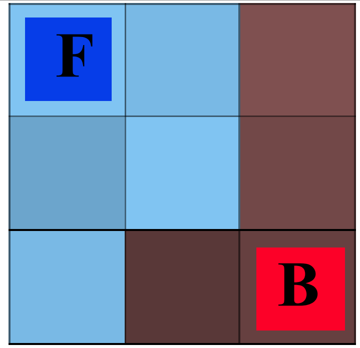
- **Yuri Boykov, Marie-Pierre Jolly**
  **Interactive Graph Cuts for Optimal Boundary & Region Segmentation of Objects in N-D Images**
  **In International Conference on Computer Vision (ICCV), vol. I, pp. 105-112, 2001**
- **C. Rother, V. Kolmogorov, A. Blake. GrabCut: Interactive Foreground Extraction using Iterated Graph Cuts. ACM Transactions on Graphics (SIGGRAPH'04), 2004**

# Cool motivation

- **The rectangle is the only user input**
- **[Rother et al.'s grabcut 2004]**

# Questions?

# Energy function

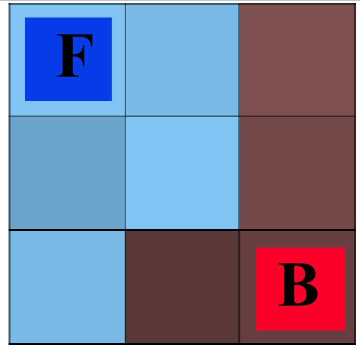



One labeling
(ok, not best)

| | | |
|---|---|---|
| F | B | B |
| F | B | B |
| F | B | B |

# Energy function

- **Labeling: one value per pixel, F or B**
- **Energy(labeling) = data + smoothness**
  - Very general situation
  - Will be minimized

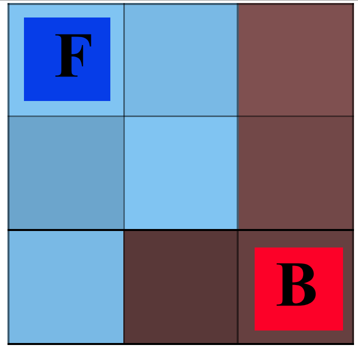



**One labeling (ok, not best)**

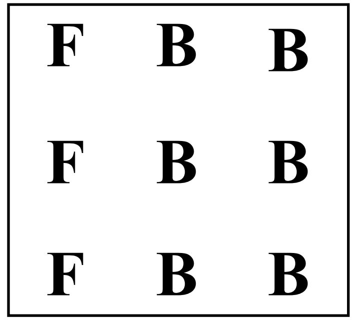
| | | |
|---|---|---|
| **F** | **B** | **B** |
| **F** | **B** | **B** |
| **F** | **B** | **B** |

# Energy function

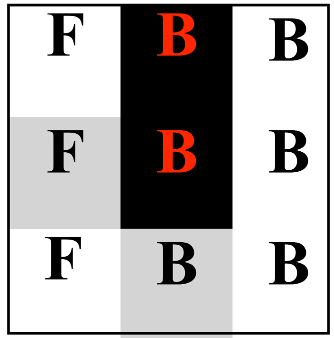
- **Labeling: one value per pixel, F or B**
- **Energy(labeling) = data + smoothness**
  - Very general situation
  - Will be minimized
- **Data: for each pixel**
  - Probability that this color belongs to F (resp. B)
  - Similar in spirit to Bayesian matting




**One labeling (ok, not best)**

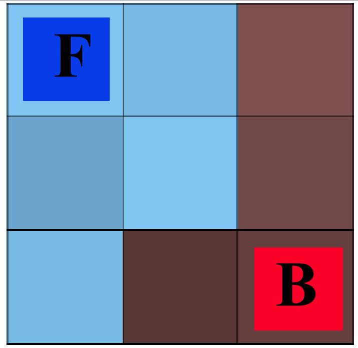



**Data**

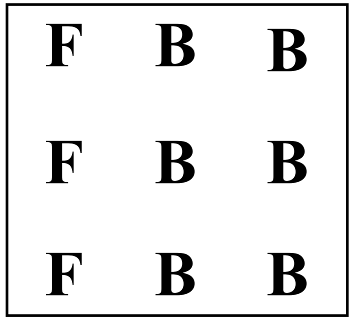


# Energy function

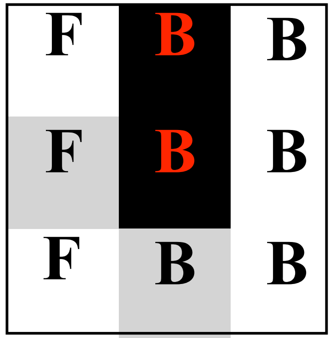
- **Labeling: one value per pixel, F or B**
- **Energy(labeling) = data + smoothness**
  - Very general situation
  - Will be minimized
- **Data: for each pixel**
  - Probability that this color belongs to F (resp. B)
  - Similar in spirit to Bayesian matting
- **Smoothness (aka regularization): per neighboring pixel pair**
  - Penalty for having different label
  - Penalty is downweighted if the two pixel colors are very different
  - Similar in spirit to bilateral filter



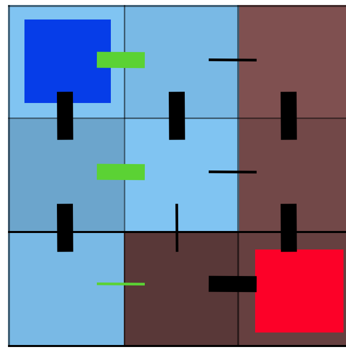

**One labeling (ok, not best)**




**Data**

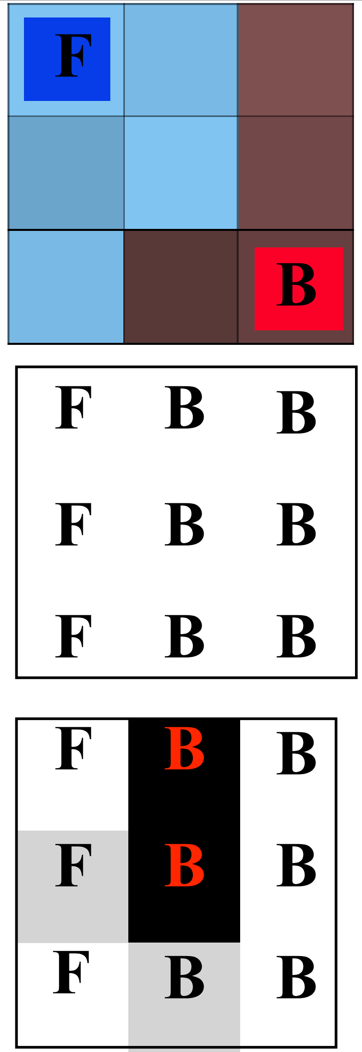



**Smoothness**

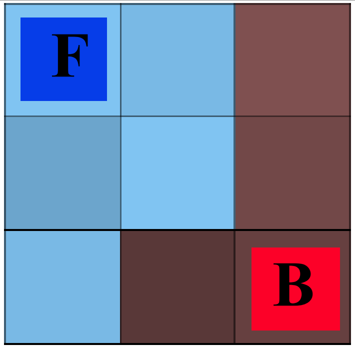


# Data term

- **A.k.a regional term**
  **(because integrated over full region)**
- **$D(L)=\sum_i -\log h[L_i](C_i)$**
- **Where *i* is a pixel**
  **$L_i$ is the label at *i* (F or B),**
  **$C_i$ is the pixel value**
  **$h[L_i]$ is the histogram of the observed Fg (resp Bg)**
- **Note the minus sign**



# Hard constraints

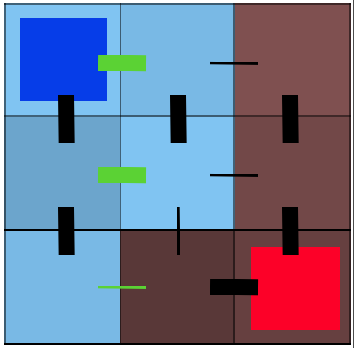



- **The user has provided some labels**
- **The quick and dirty way to include constraints into optimization is to replace the data term by a huge penalty K if not respected.**
- **D(Li)=0 if respected**
- **D(Li) = K if not respected**
  – e.g. K= - #pixels

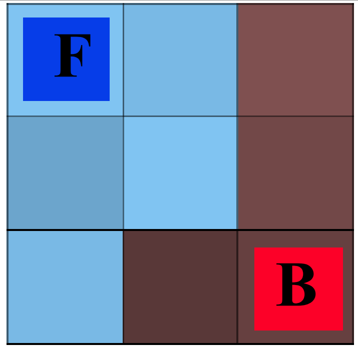
# Smoothness term

- ***a.k.a boundary term, a.k.a. regularization***
- **$S(L)=\Sigma_{\{j,\ i\}2\ N}\ B(C_i,C_j)\ \delta(L_i-L_j)$**
- **Where i,j are neighbors**
  - e.g. 8-neighborhood
    (but I show 4 for simplicity)
- **$\delta(L_i-L_j)$ is 0 if $L_i=L_j$, 1 otherwise**
- **$B(C_i,C_j)$ is high when $C_i$ and $C_j$ are similar, low if there is a discontinuity between those two pixels**
  - e.g. $\exp(-||C_i-Cj||^2/2\sigma^2)$
  - where $\sigma$ can be a constant
    or the local variance

| | | |
|---|---|---|
| F | B | B |
| F | B | B |
| F | B | B |



# Recap: Energy function

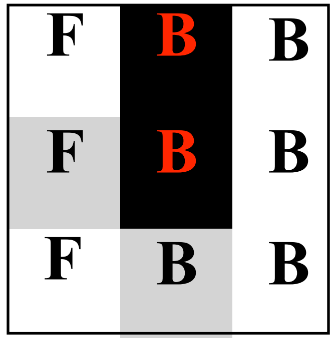



- **Labeling: one value Li per pixel, F or B**
- **Energy(labeling) = Data + Smoothness**
- **Data: for each pixel**
  - Probability that this color belongs to F (resp. B)
  - Using histogram
  - $D(L)=\Sigma_i$ -log $h[L_i](C_i)$
- **Smoothness (aka regularization): per neighboring pixel pair**
  - Penalty for having different label
  - Penalty is downweighted if the two pixel colors are very different
  - $S(L)=\Sigma_{\{j,\ i\}2\ N}\ B(C_i,C_j)\ \delta(L_i-L_j)$

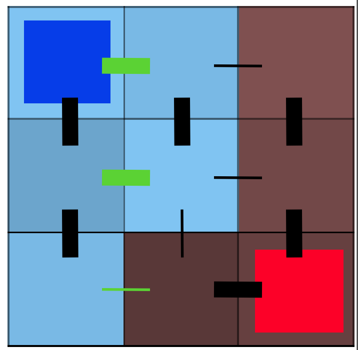
**One labeling (ok, not best)**

| | | |
|---|---|---|
| F | B | B |
| F | B | B |
| F | B | B |

**Data**




**Smoothness**

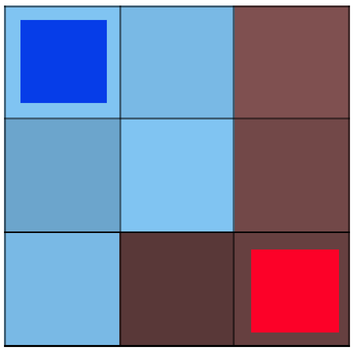


# Questions?

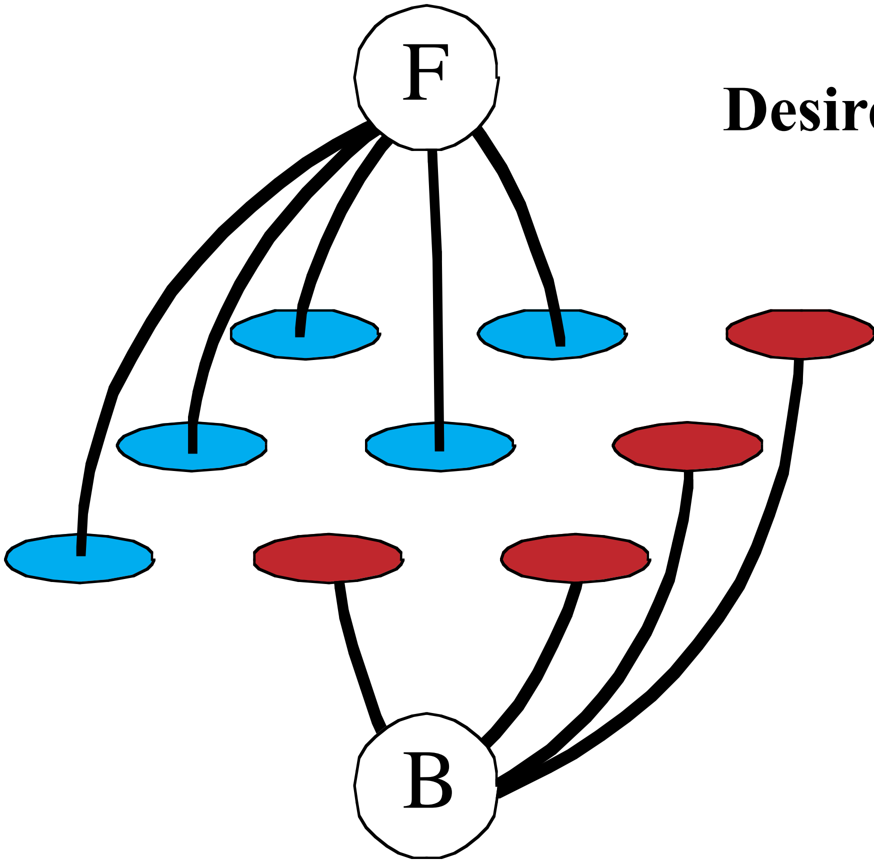
- **Recap:**
  - Labeling F or B
  - Energy(Labeling) = Data+Smoothness
  - Need efficient way to find labeling with lowest energy

# Labeling as a graph problem

- **Each pixel = node**
- **Add two label nodes F & B**
- **Labeling: link each pixel to either F or B**



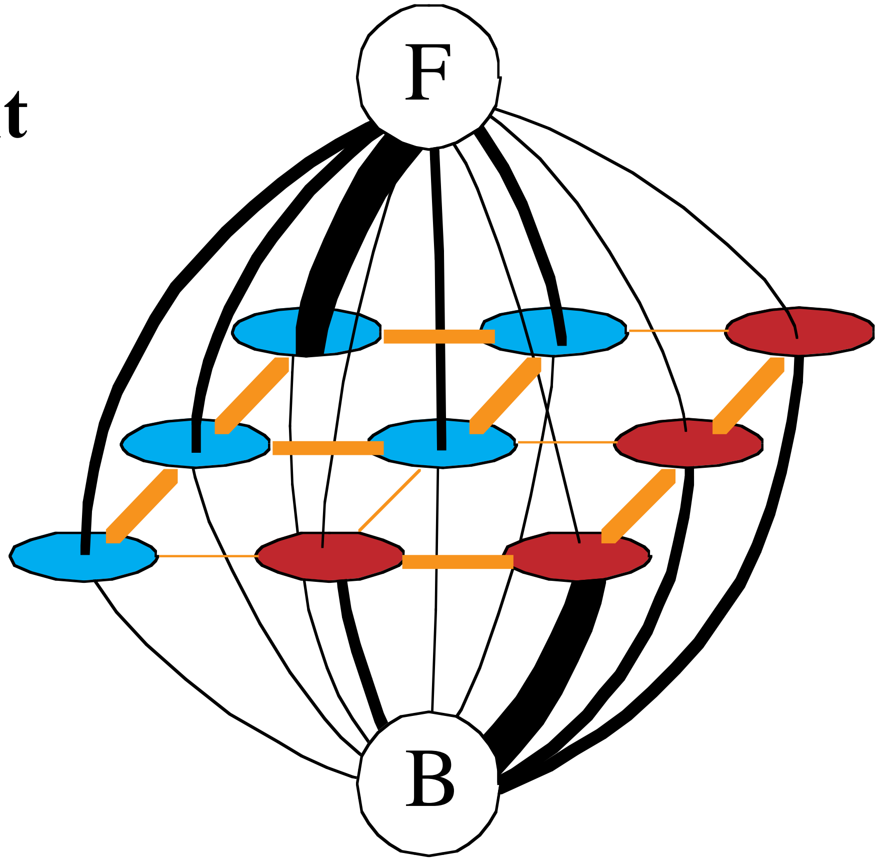
| | | |
|---|---|---|
| F | F | B |
| F | F | B |
| F | B | B |




**Desired result**

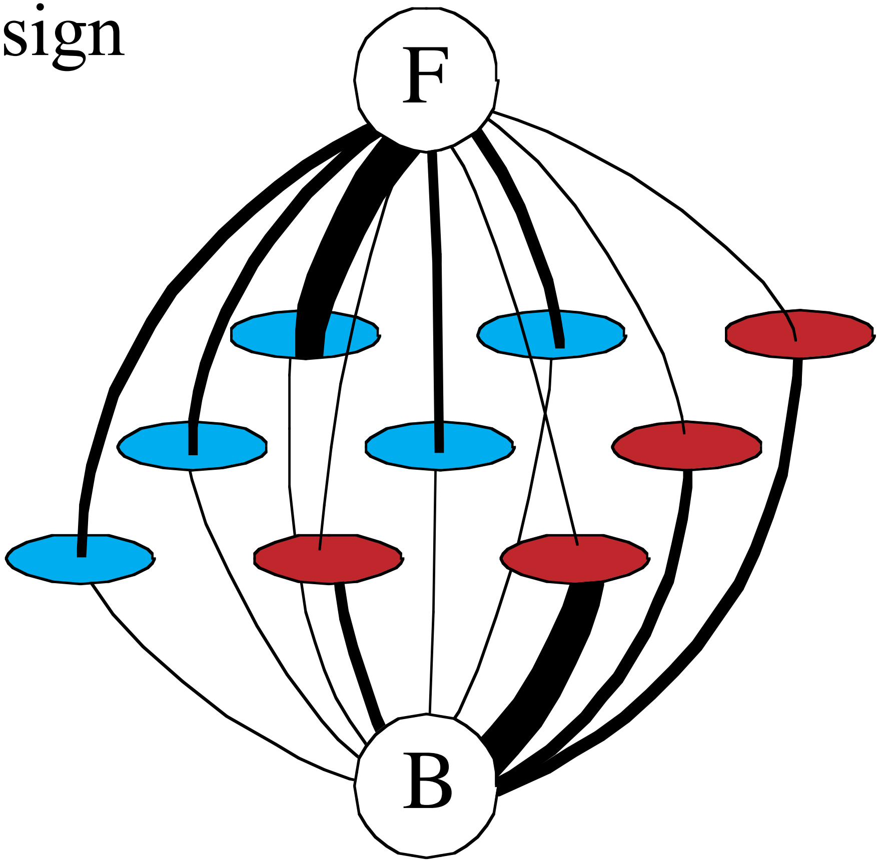
# Idea

- **Start with a graph with too many edges**
  - Represents all possible labeling
  - Strength of edges depends on data and smoothness terms
- **solve as min cut**



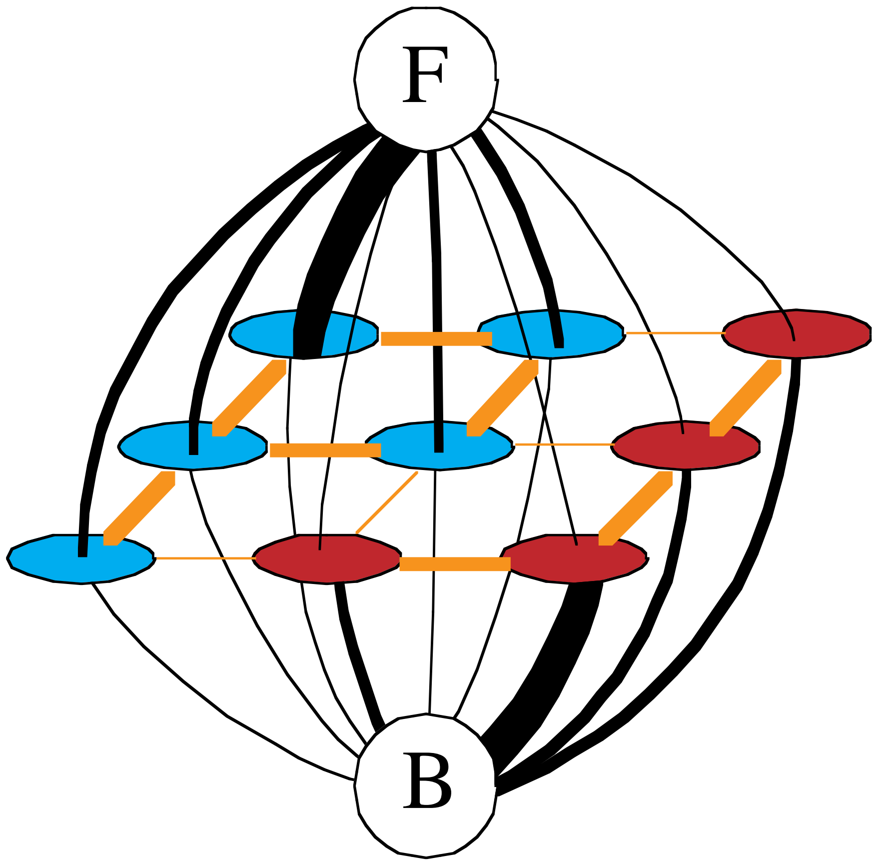
# Data term

- **Put one edge between each pixel and both F & G**
- **Weight of edge = minus data term**
  - Don't forget huge weight for hard constraints
  - Careful with sign



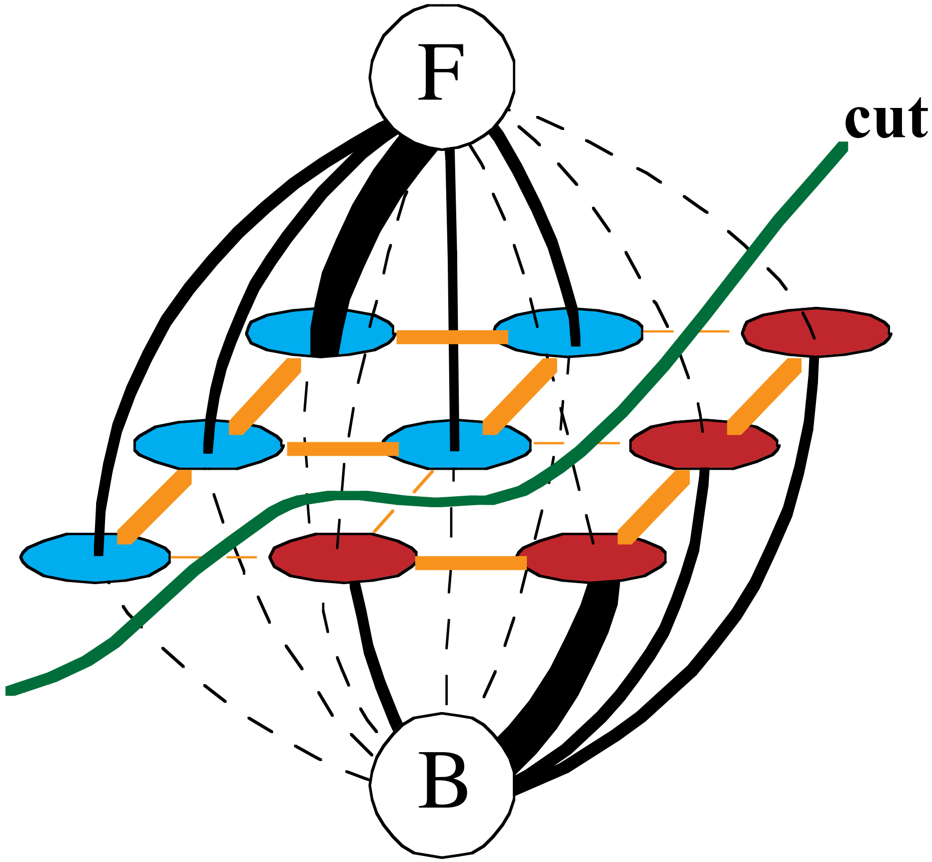
# Smoothness term

- **Add an edge between each neighbor pair**
- **Weight = smoothness term**



# Min cut

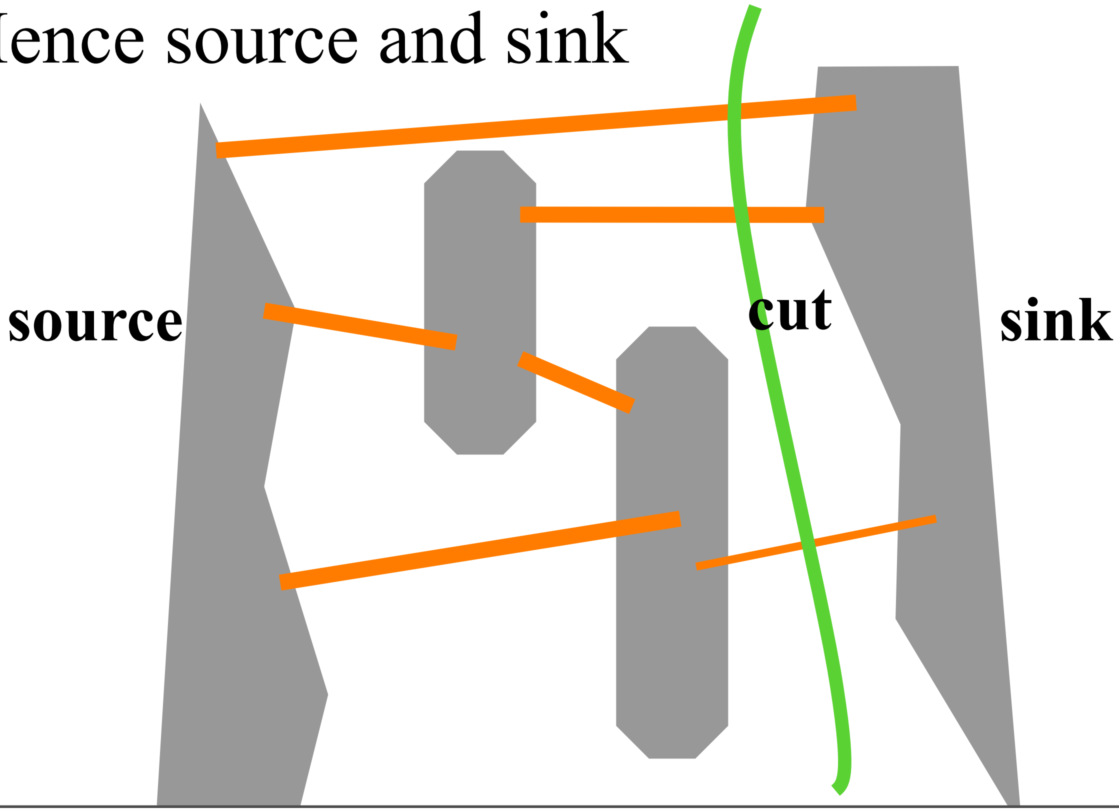
- **Energy optimization equivalent to graph min cut**
- **Cut: remove edges to disconnect F from B**
- **Minimum: minimize sum of cut edge weight**



# Questions?

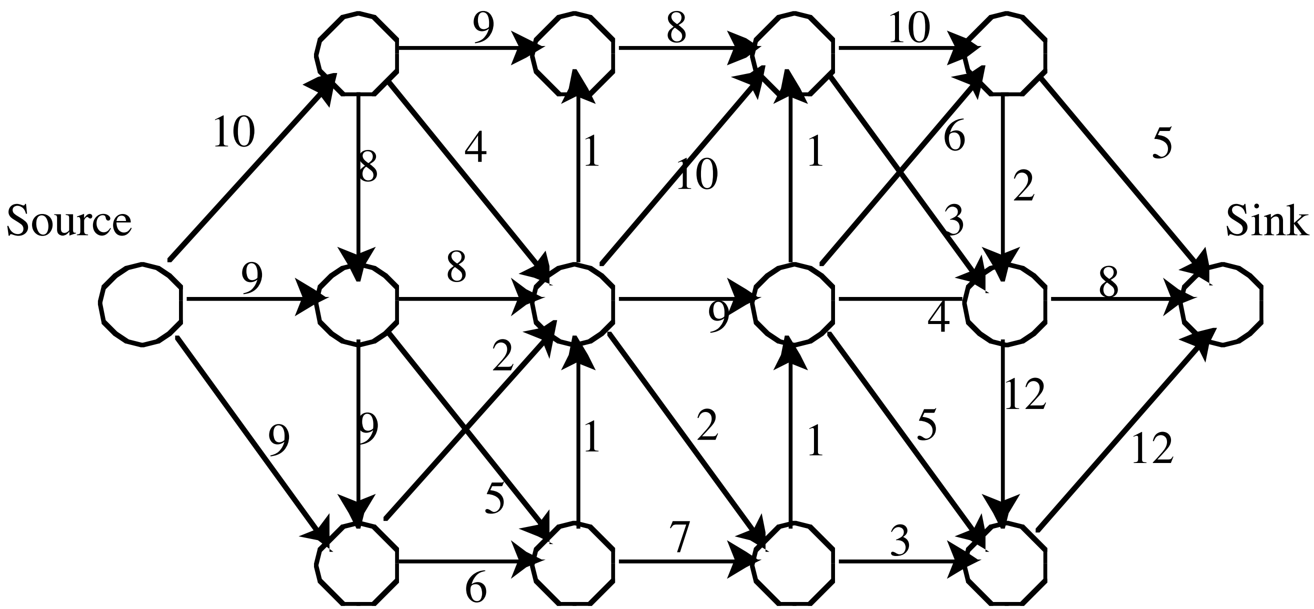
# Min cut

- **Graph with one source & one sink node**
- **Edge = bridge; Edge label = cost to cut bridge**
- **Find the min-cost cut that separates source from sink**
  - Turns out it's easier to see it as a *flow* problem
  - Hence source and sink



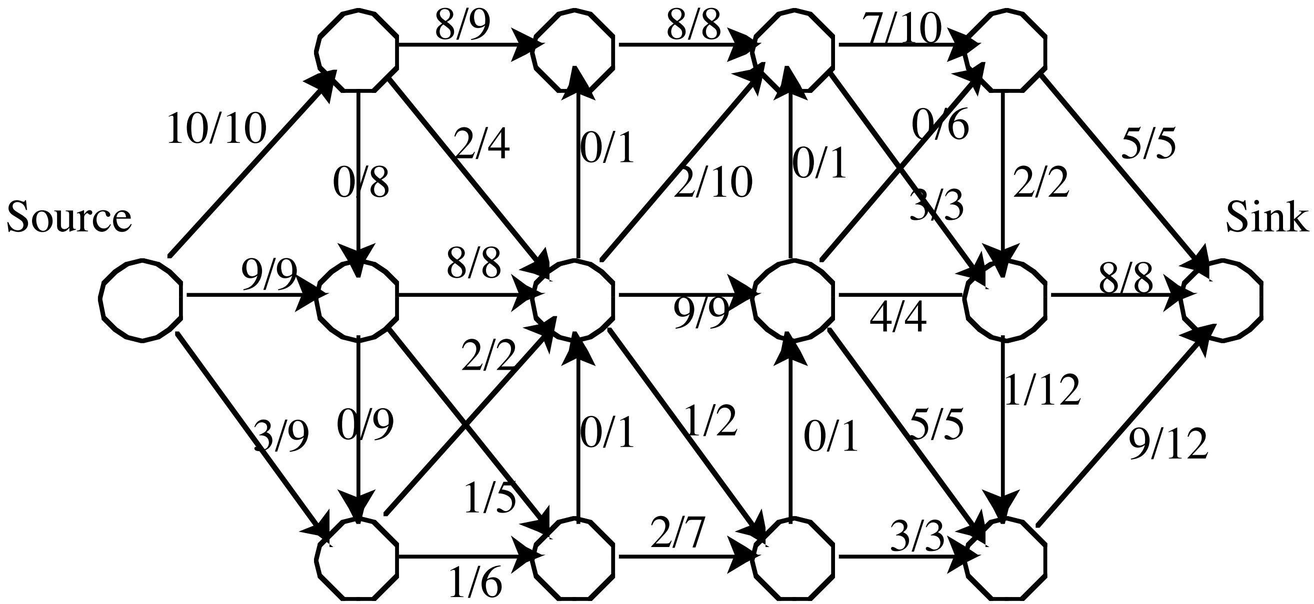
# Max flow

- **Directed graph with one source & one sink node**
- **Directed edge = pipe**
- **Edge label = capacity**
- **What is the max flow from source to sink?**



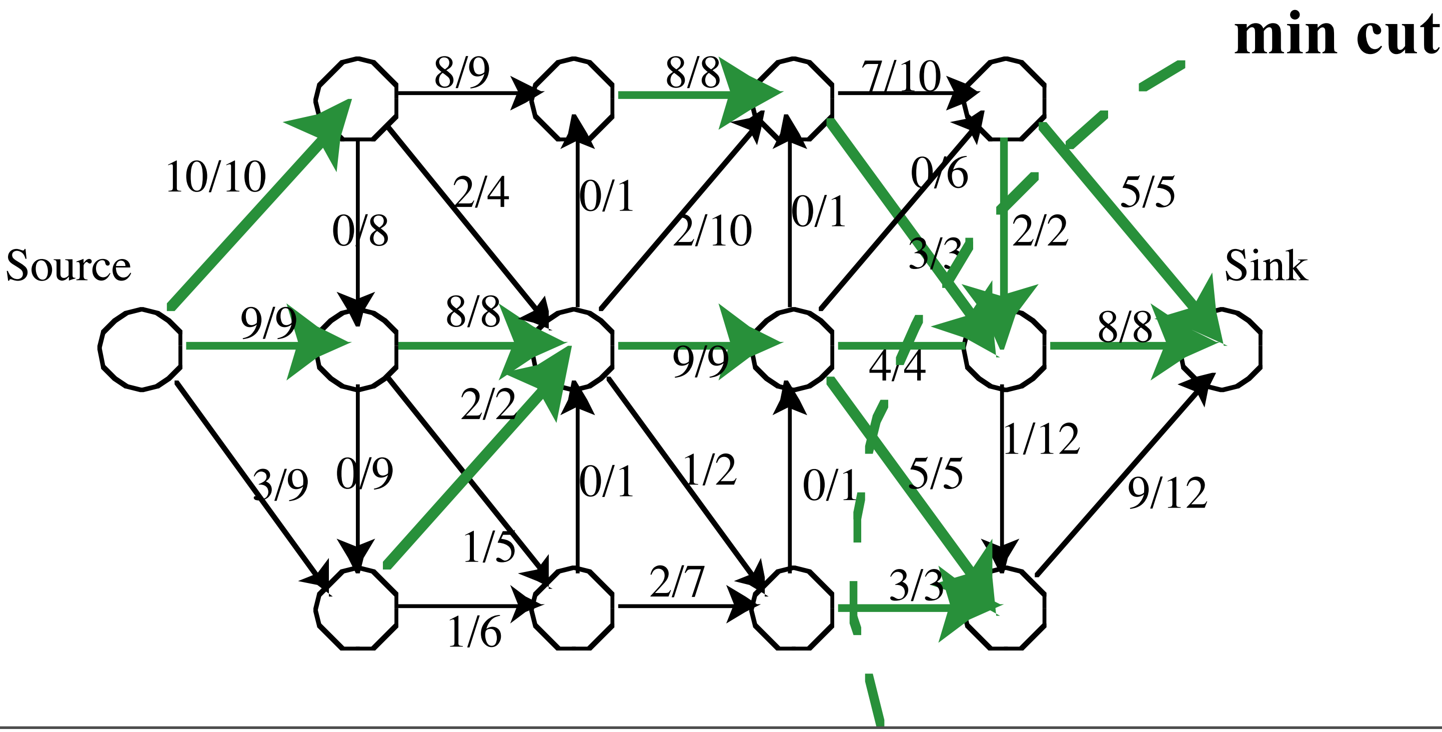
# Max flow

- **Graph with one source & one sink node**
- **Edge = pipe**
- **Edge label = capacity**
- **What is the max flow from source to sink?**



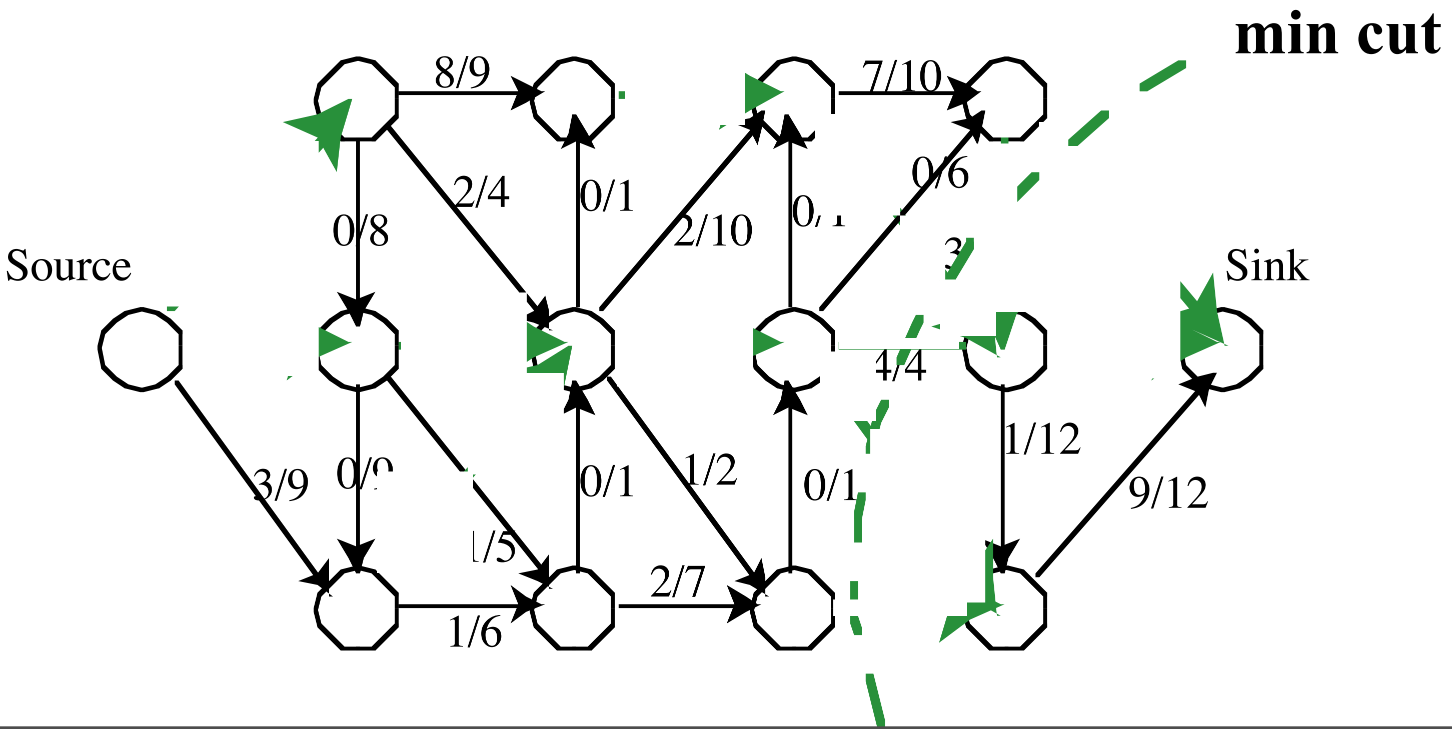
# Max flow

- **What is the max flow from source to sink?**
- **Look at residual graph**
  - remove saturated edges (green here)
  - min cut is at boundary between 2 connected components



# Max flow

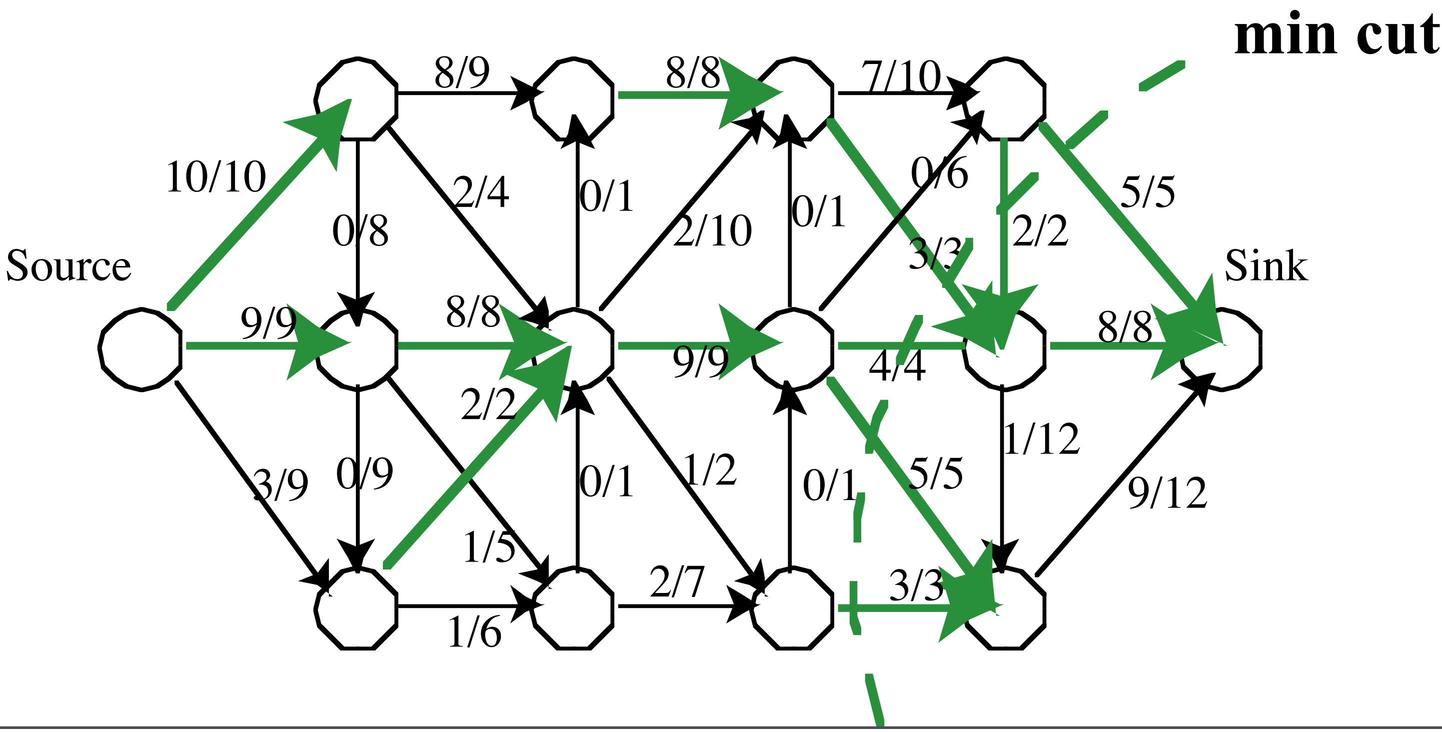
- **What is the max flow from source to sink?**
- **Look at residual graph**
  - remove saturated edges (gone here)
  - min cut is at boundary between 2 connected components



# Equivalence of min cut / max flow

**The three following statements are equivalent**

- **The maximum flow is *f***
- **The minimum cut has weight *f***
- **The residual graph for flow *f* contains no directed path from source to sink**



# Questions?

- **Recap:**
  - We have reduced labeling to a graph min cut
    - vertices for pixels and labels
    - edges to labels (data) and neighbors (smoothness)
  - We have reduced min cut to max flow

  - Now how do we solve max flow???

# Max flow algorithm

- **We will study a strategy where we keep augmenting paths (Ford-Fulkerson, Dinic)**
- **Keep pushing water along non-saturated paths**
  - Use residual graph to find such paths

# Max flow algorithm

```
Set flow to zero everywhere
Big loop
   compute residual graph
   Find path from source to sink in residual
      If path exist add corresponding flow
      Else
         Min cut = {vertices reachable from source;
           other vertices}
         terminate
```

**Animation at**
http://www.cse.yorku.ca/~aaw/Wang/MaxFlowStart.htm

# Efficiency concerns

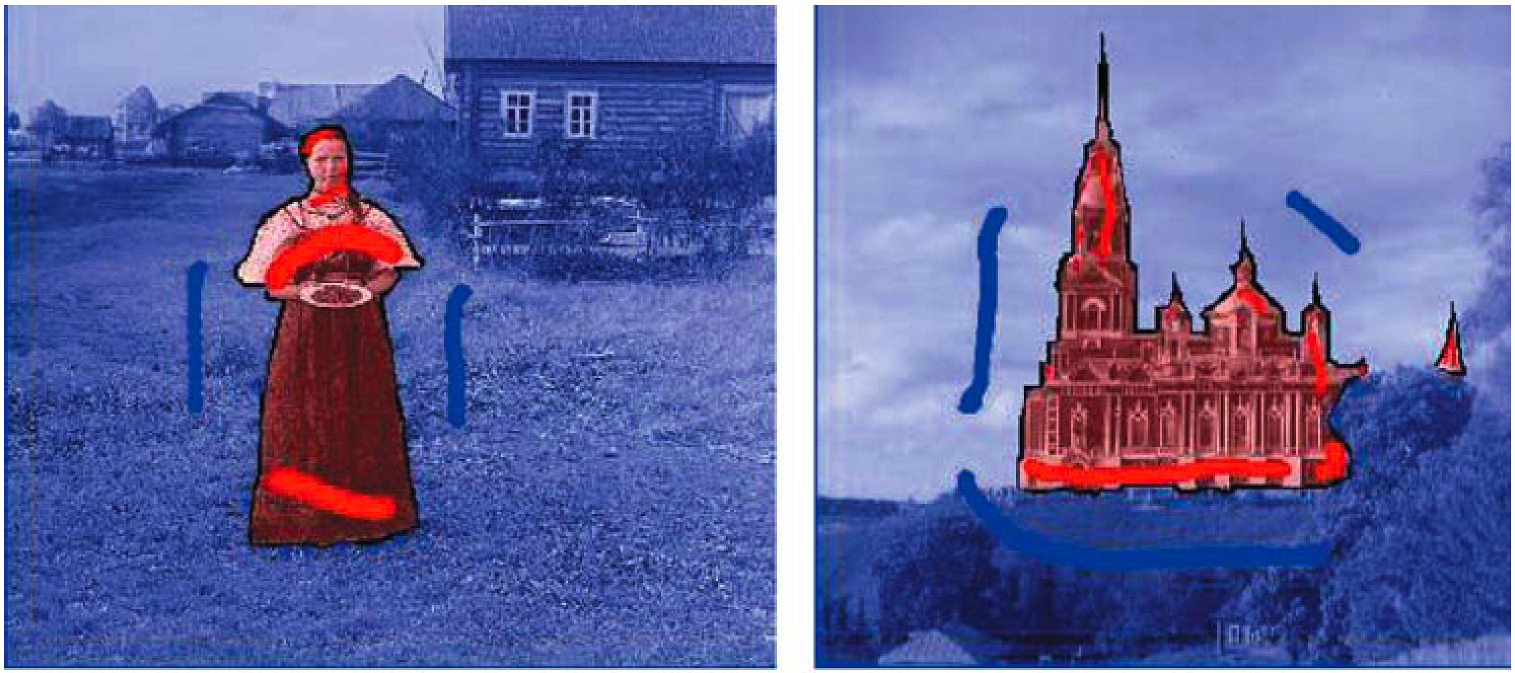
- **The search for a shortest path becomes prohibitive for the large graphs generated by images**
- **For practical vision/image applications, better (yet related) approaches exist**

  An Experimental Comparison of Min-Cut/Max-Flow Algorithms for Energy Minimization in Vision. Yuri Boykov, Vladimir Kolmogorov In IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 26, no. 9, Sept. 2004. http://www.csd.uwo.ca/faculty/yuri/Abstracts/pami04-abs.html

- **Maintain two trees from sink & source.**
- **Augment tree until they connect**
- **Add flow for connection**
- **Can require more iterations because not shortest path**
  **But each iteration is cheaper because trees are reused**

# Questions?

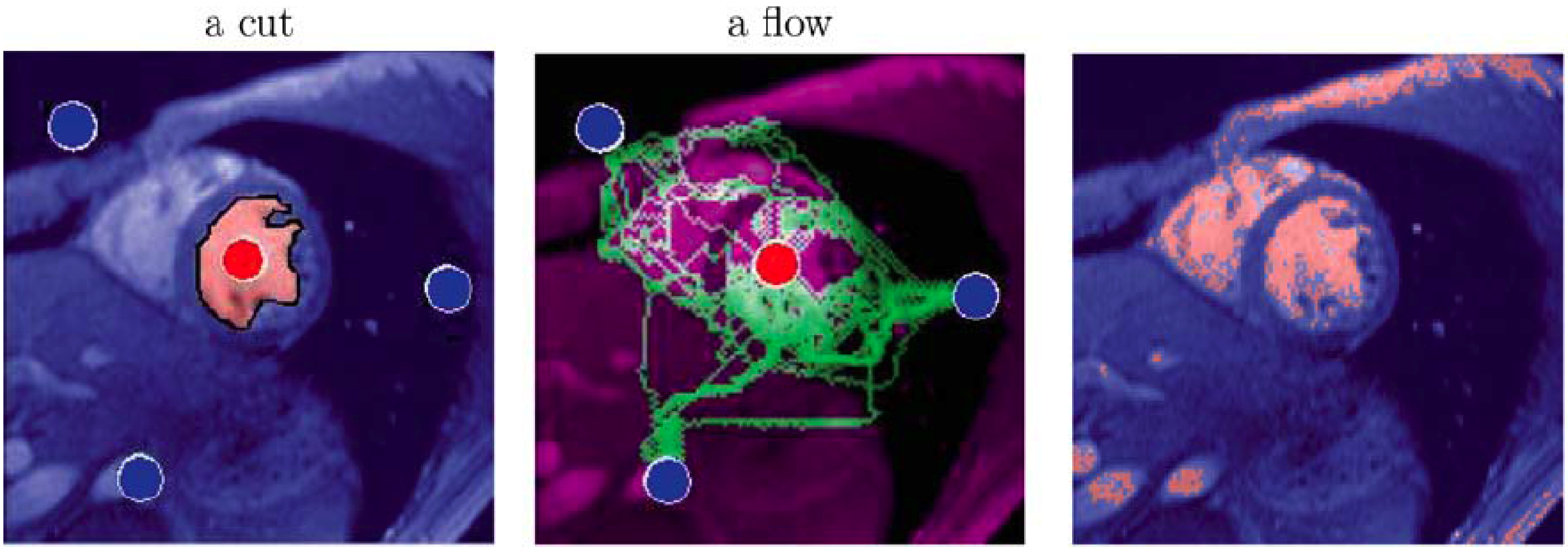
- Graph Cuts and Efficient N-D Image Segmentation
- Yuri Boykov, Gareth Funka-Lea
- In International Journal of Computer Vision (IJCV), vol. 70, no. 2, pp. 109-131, 2006 (accepted in 2004).



(a) A woman from a village　　(b) A church in Mozhaisk (near Moscow)

*Figure 8.* Segmentation of photographs (early 20th century). Initial segmentation for a given set of hard constraints (seeds) takes less than a second for most 2D images (up to 1000 × 1000). Correcting seeds are incorporated in the blink of an eye. Thus, the speed of our method for photo editing mainly depends on time for placing seeds. An average user will not need much time to enter seeds in (a) and (b).

- **Importance of smoothness**

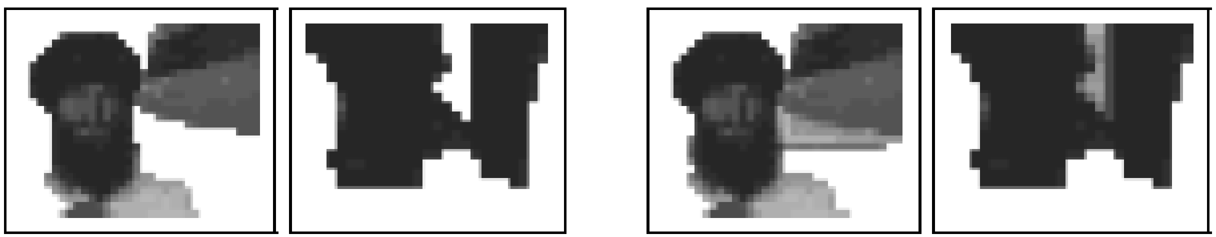



(a) Boundary cues with topological constraints (b) Region-based cues (only)

**From Yuri Boykov, Gareth Funka-Lea**

# Data (regional) term

(a) Original B&W photo (b) Segmentation results



(c) Details of segmentation with regional term (d) Details of segmentation without regional term

# Graph cut is a very general tool

- **Stereo depth reconstruction**
- **Texture synthesis**
- **Video synthesis**
- **Image denoising**

**3D model of scene**

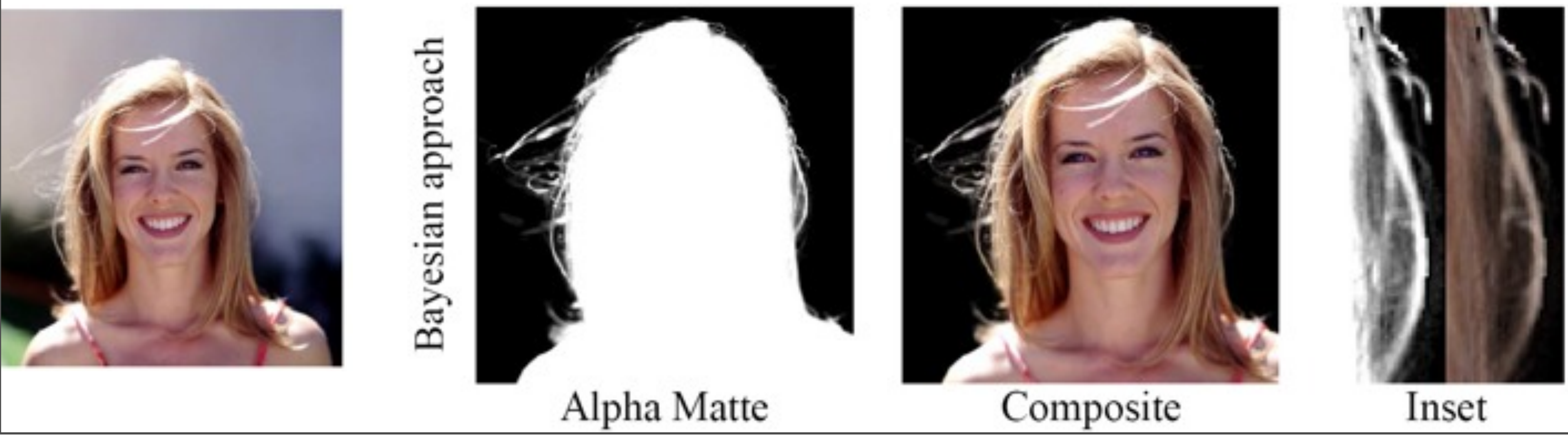
# Questions?

# Refs

- http://www.csd.uwo.ca/faculty/yuri/Abstracts/eccv06-tutorial.html
- **Interactive Graph Cuts for Optimal Boundary & Region Segmentation of Objects in N-D images.**
  **Yuri Boykov and Marie-Pierre Jolly.**
  **In International Conference on Computer Vision, (ICCV), vol. I, 2001.**
  http://www.csd.uwo.ca/~yuri/Abstracts/iccv01-abs.html
- http://www.cse.yorku.ca/~aaw/Wang/MaxFlowStart.htm
- http://research.microsoft.com/en-us/um/cambridge/projects/visionimagevideoediting/segmentation/grabcut.htm
- http://www.cc.gatech.edu/cpl/projects/graphcuttextures/
- **A Comparative Study of Energy Minimization Methods for Markov Random Fields. Rick Szeliski, Ramin Zabih, Daniel Scharstein, Olga Veksler, Vladimir Kolmogorov, Aseem Agarwala, Marshall Tappen, Carsten Rother. ECCV 2006**
  www.cs.cornell.edu/~rdz/Papers/SZSVKATR.pdf

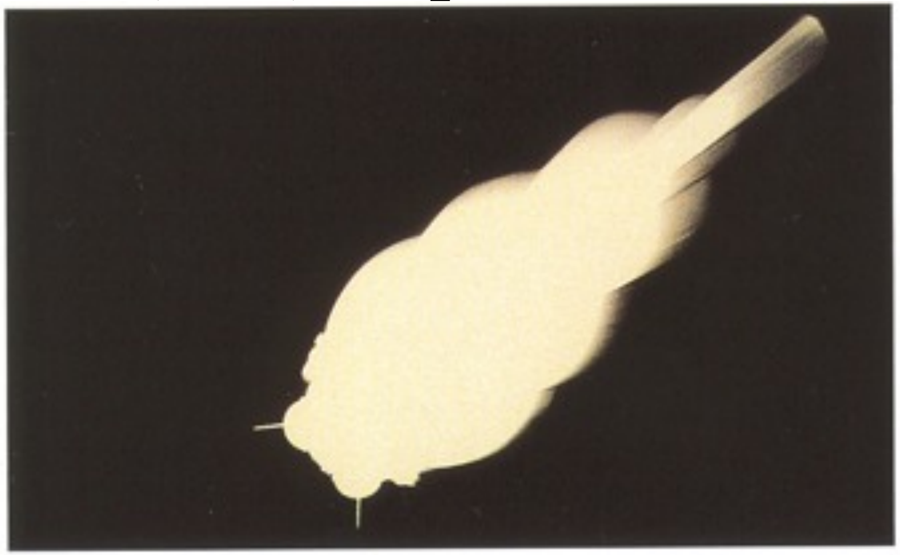
# Beyond binary segmentation

- **Matte: fractional visibility**
  - e.g. 34% foreground, 66% background
  - Critical for hair, complex boundaries, defocus/motion blur
  - Very challenging



# Why fractional alpha?

- Motion blur, small features (hair), depth of field cause partial occlusion



From Digital Domain

# With binary alpha

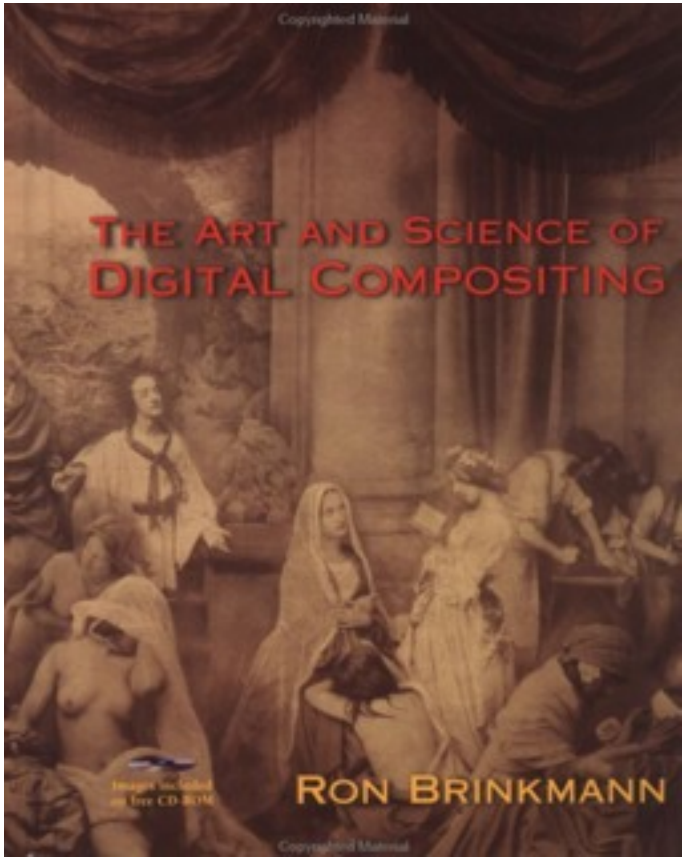
From Digital Domain

# With fractional alpha

From Digital Domain

# References

- **Smith & Blinn 1996**
  **http://portal.acm.org/citation.cfm?id=237263**
  **Formal treatment of Blue screen**
- **Ruzon & Tomasi 2000**
  **http://ai.stanford.edu/~ruzon/alpha/**
  **The breakthrough that renewed the issue**
  **(but not crystal clear)**
- **Chuang et al. 2001**
  **http://research.microsoft.com/vision/visionbasedmodeling/publications/Chuang-CVPR01.pdf**
- **Brinkman's Art & Science of Digital Compositing**
  - Not so technical , more for practitioners



# More Refs

**Matting:**

- http://graphics.cs.cmu.edu/courses/15-463/2004_fall/www/Lectures/matting.pdf
- http://www.csie.ntu.edu.tw/~cyy/publications/papers/Chuang2004Phd.pdf
- http://www.cse.ucsd.edu/classes/wi03/cse291-j/lec10-compositing.pdf
- http://graphics.stanford.edu/courses/cs248-99/comp/hanrahan-comp-excerpt.ppt

**Chroma Key**

- http://www.cs.utah.edu/~michael/chroma/

**Blue screen:**

- http://www.sut.ac.th/emdp/VisualEffect/The%20Blue%20Screen%20-%20Chroma%20Key%20Page.htm
- http://www.cs.princeton.edu/courses/archive/fall00/cs426/papers/smith95c.pdf
- http://www.seanet.com/Users/bradford/bluscrn.html
- http://en.wikipedia.org/wiki/Bluescreen
- http://www.neopics.com/bluescreen/
- http://entertainment.howstuffworks.com/blue-screen.htm
- http://www.vce.com/bluescreen.html
- http://www.pixelpainter.com/NAB/Blue_vs_Green_Screen_for_DV.pdf

**Petro Vlahos (inventor of blue screen matting)**

- http://theoscarsite.com/whoswho4/vlahos_p.htm
- http://en.wikipedia.org/wiki/Petro_Vlahos

**To buy a screen:**
http://shop.store.yahoo.com/cinemasupplies/chromkeyfab.html

**Superman & blue screen:**

- http://supermancinema.co.uk/superman1/the_production/the_crew/fx_bios/index.shtml
- http://home.utm.utoronto.ca/~kin/bluescreen.htm

Warning:
French Mathematicians inside

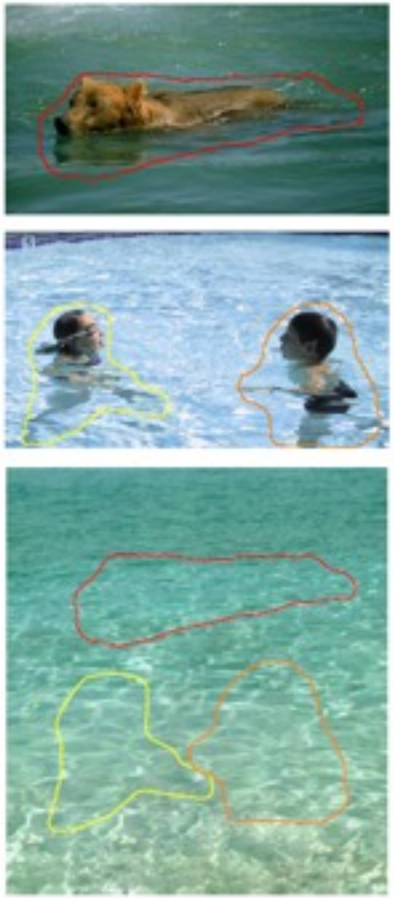
# Gradient Image Processing

**Frédo Durand**
**MIT - EECS**

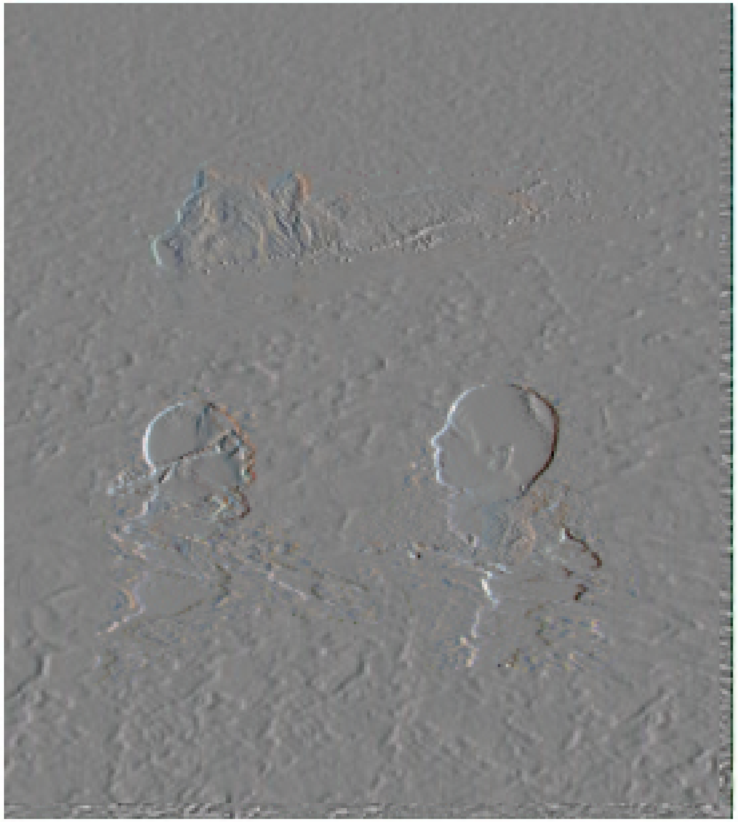
# Problems with direct copy/paste


From Perez et al. 2003

# Solution: paste gradient



sources/destinations



hacky visualization of gradient

seamless cloning

# Demo of healing brush

- **Slightly smarter version of what we learn today**
  - higher-order derivative in particular

# What is a gradient?

# What is a gradient?

- **derivative of a multivariate function**
- **for example, for f(x,y)**

$$\nabla f = \left( \frac{df}{dx}, \frac{df}{dy} \right)$$

- **For a discrete image, can be approximated with finite differences**

$$\frac{df}{dx} \approx f(x+1, y) - f(x, y)$$

$$\frac{df}{dy} \approx f(x, y+1) - f(x, y)$$

# Gradients and grayscale images

- **Grayscale image: n×n scalars**
- **Gradient:**

# Gradients and grayscale images

- **Grayscale image: n×n scalars**
- **Gradient: n×n *2D vectors***

# Gradients and grayscale images

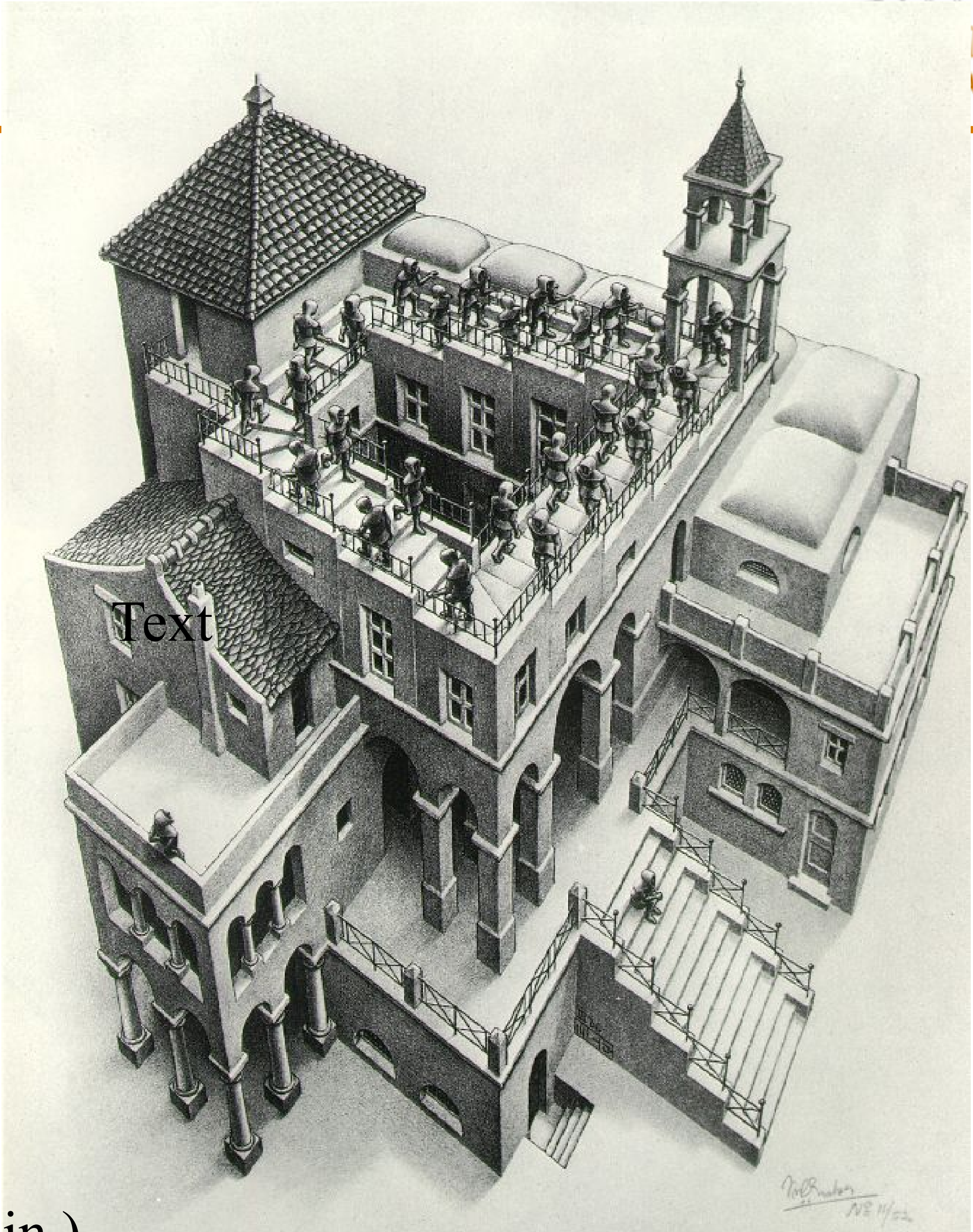
- **Grayscale image: n×n scalars**
- **Gradient: n×n *2D vectors***
- **Two many numbers!**
- **What's up with this?**

# Gradients and grayscale images

- **Grayscale image: n×n scalars**
- **Gradient: n×n *2D vectors***
- **Two many numbers!**
- **What's up with this?**
- **Not all vector fields are the gradient of an image!**

# Gradients and grayscale images

- **Grayscale image: n×n scalars**
- **Gradient: n×n *2D vectors***
- **Two many numbers!**
- **What's up with this?**
- **Not all vector fields are the gradient of an image!**
- **Only if they are curl-free (a.k.a. conservative)**
  – But we'll see it does not matter for us




Escher, Maurits Cornelis
Ascending and Descending
1960
Lithograph
35.5 x 28.5 cm (14 x 11 1/4 in.)

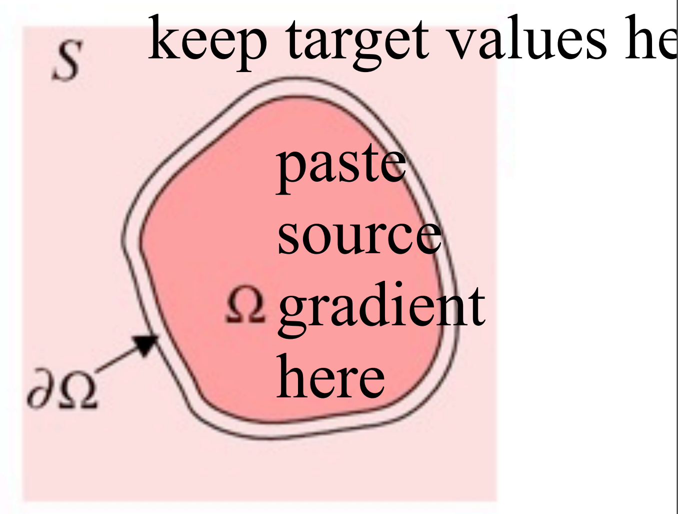
# Color images

- **3 gradients, one for each channel.**
- **We'll sweep this under the rug for this lecture**
- **In practice, treat each channel independently**

# Questions?

# Seamless Poisson cloning

- **Paste source gradient into target image inside a selected region**
- **Make the new gradient as close as possible to the source gradient while respecting pixel values at the boundary**

# Seamless Poisson cloning

- **Given vector field *v* (pasted gradient), find the value of *f* in unknown region that optimize:**

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

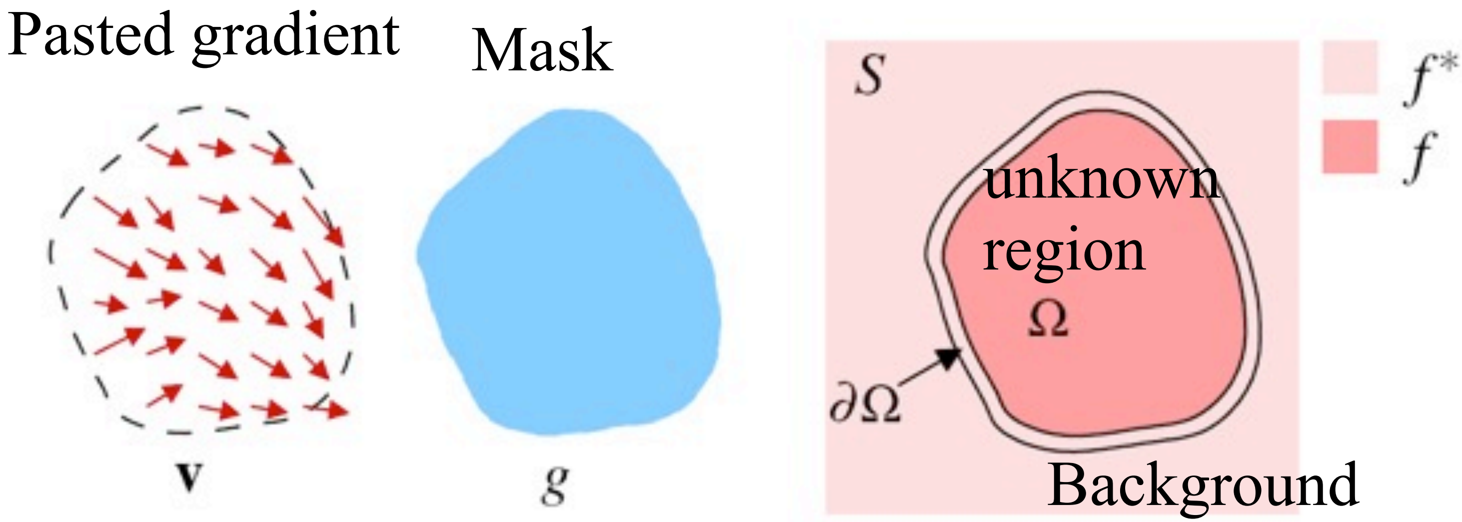



Figure 1: **Guided interpolation notations**. Unknown function $f$ interpolates in domain $\Omega$ the destination function $f^*$, under guidance of vector field $\mathbf{v}$, which might be or not the gradient field of a source function $g$.

# Seamless Poisson cloning

- **Given vector field *v* (pasted gradient), find the value of *f* in unknown region that optimize:**

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

Poisson equation with Dirichlet conditions

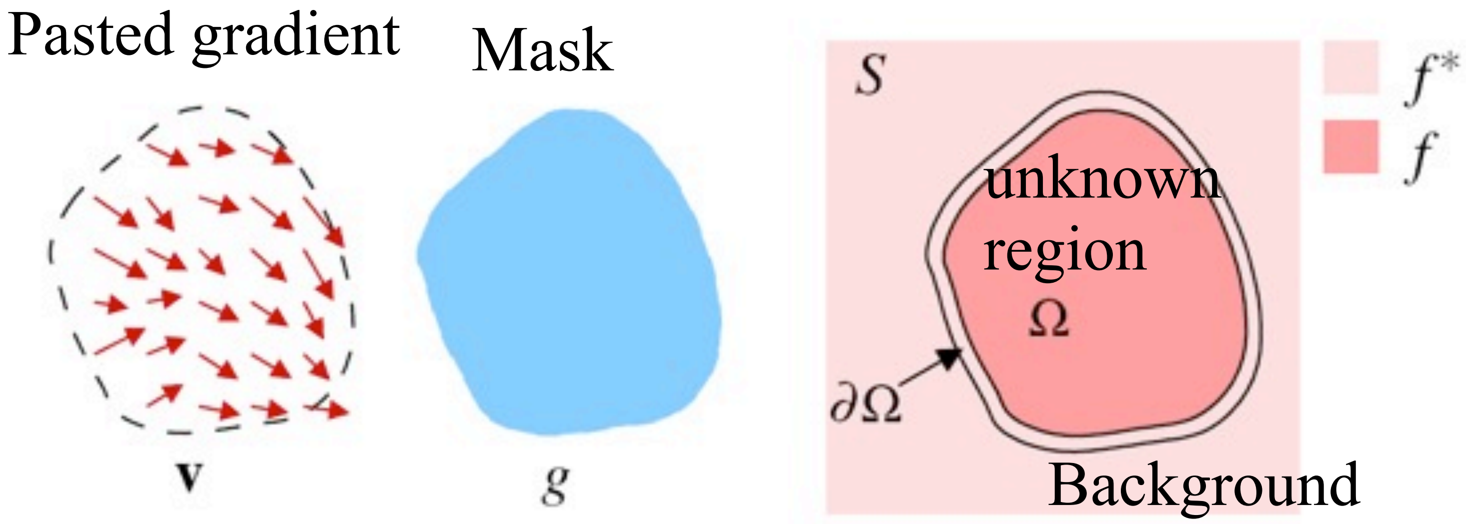
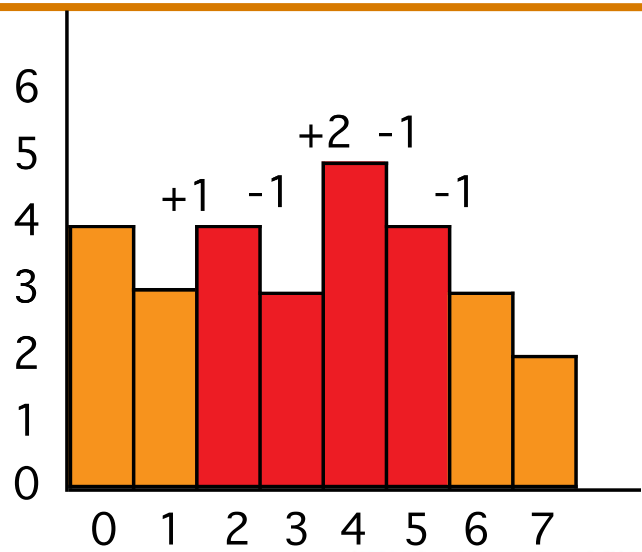


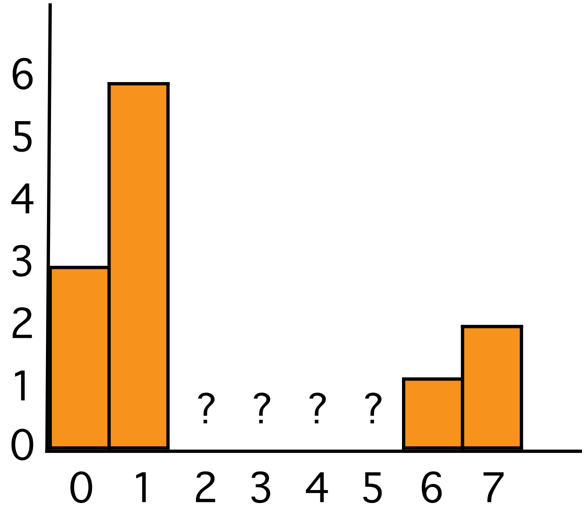

Figure 1: **Guided interpolation notations.** Unknown function $f$ interpolates in domain $\Omega$ the destination function $f^*$, under guidance of vector field $\mathbf{v}$, which might be or not the gradient field of a source function $g$.

# Discrete 1D example: minimization

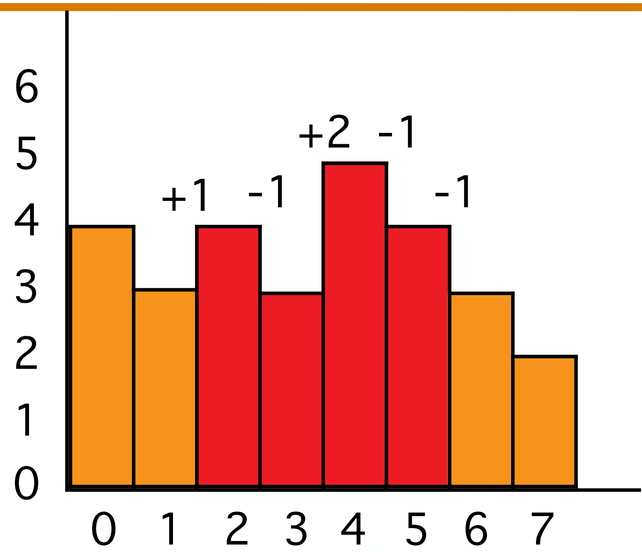
- **Copy**

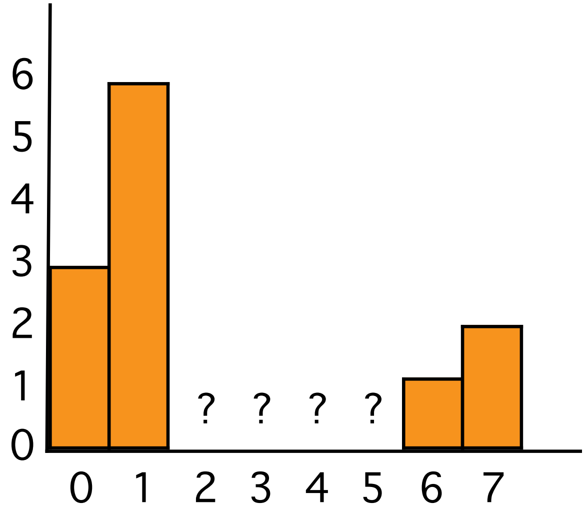



**to**




# Discrete 1D example: minimization

- **Copy** 

**to** 


**Min** $[(f_2-f_1)-1]^2$

$+ [(f_3-f_2)-(-1)]^2$

$+ [(f_4-f_3)-2]^2$

$+ [(f_5-f_4)-(-1)]^2$

$+ [(f_6-f_5)-(-1)]^2$

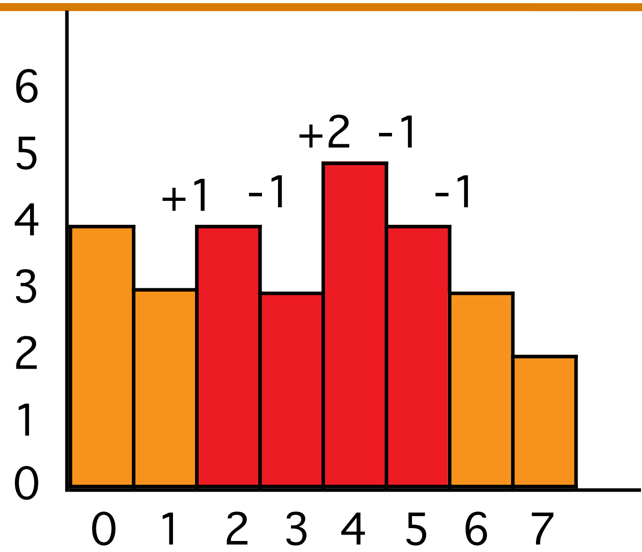
With

$f_1=6$

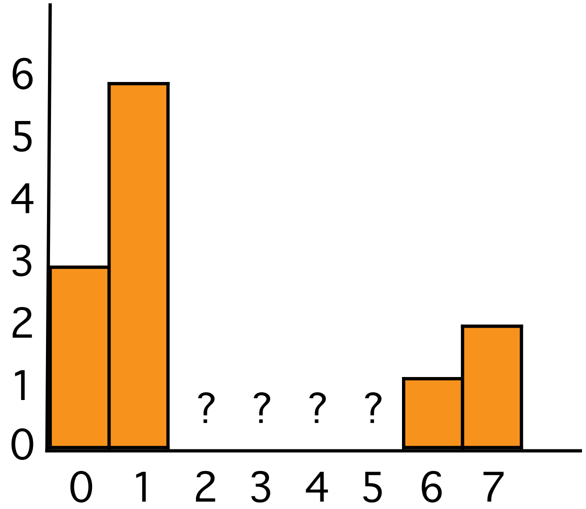
$f_6=1$

# 1D example: minimization

- **Copy**




**to**




$$\text{Min } [(f_2-f_1)-1]^2$$

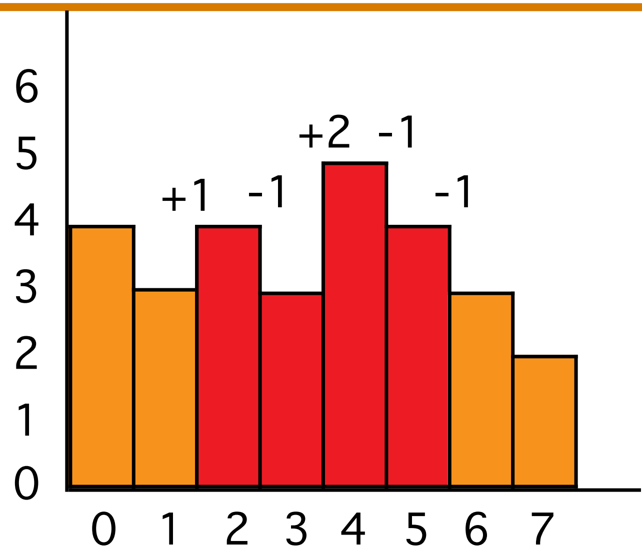
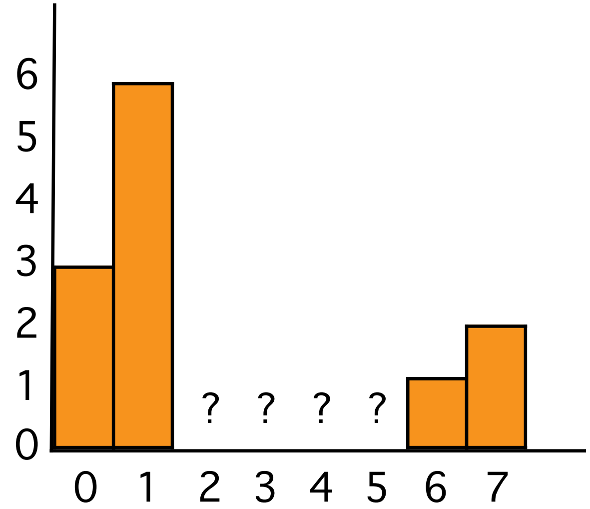
$$+ [(f_3-f_2)-(-1)]^2$$

$$+ [(f_4-f_3)-2]^2$$

$$+ [(f_5-f_4)-(-1)]^2$$

$$+ [(f_6-f_5)-(-1)]^2$$

# 1D example: minimization

- **Copy**  **to** 





$$\text{Min } [(f_2-f_1)-1]^2 \quad ==> \quad f_2^2+49-14f_2$$

$$+ [(f_3-f_2)-(-1)]^2 \quad ==> \quad f_3^2+f_2^2+1-2f_3f_2 +2f_3-2f_2$$

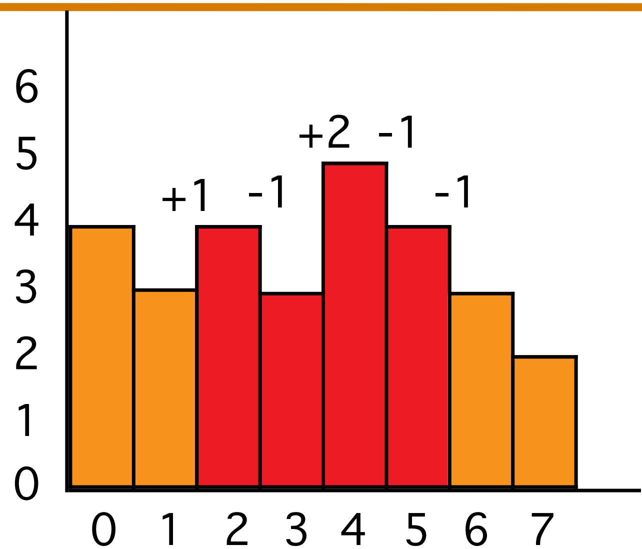
$$+ [(f_4-f_3)-2]^2 \quad ==> \quad f_4^2+f_3^2+4-2f_3f_4 -4f_4+4f_3$$

$$+ [(f_5-f_4)-(-1)]^2 \quad ==> \quad f_5^2+f_4^2+1-2f_5f_4 +2f_5-2f_4$$

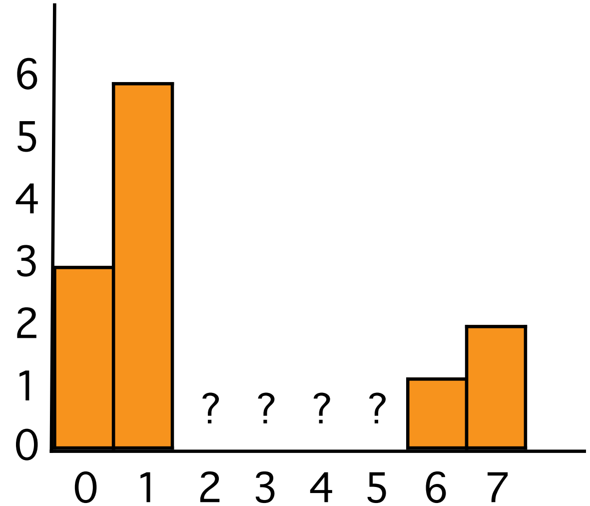
$$+ [(f_6-f_5)-(-1)]^2 \quad ==> \quad f_5^2+4-4f_5$$

# 1D example: big quadratic

- Copy




to




- Min $(f_2^2+49-14f_2$

  $+ f_3^2+f_2^2+1-2f_3f_2 +2f_3-2f_2$

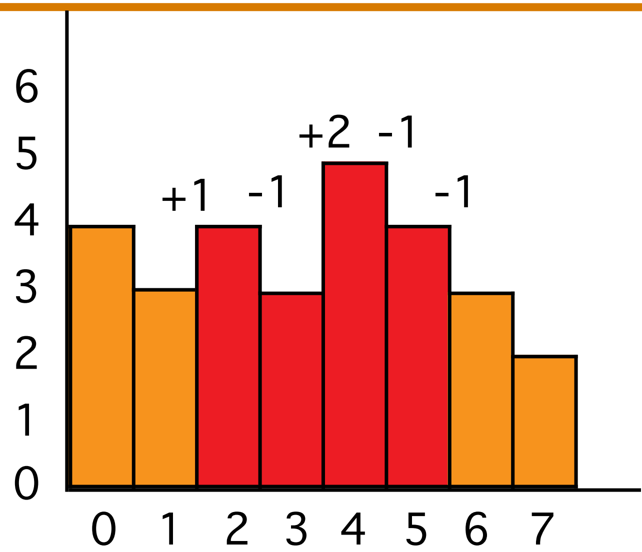
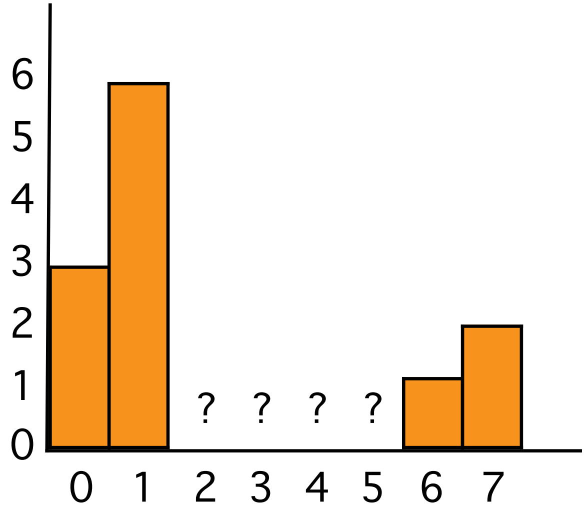
  $+ f_4^2+f_3^2+4-2f_3f_4 -4f_4+4f_3$

  $+ f_5^2+f_4^2+1-2f_5f_4 +2f_5-2f_4$

  $+ f_5^2+4-4f_5)$

  **Denote it Q**

# 1D example: derivatives

- Copy  to 



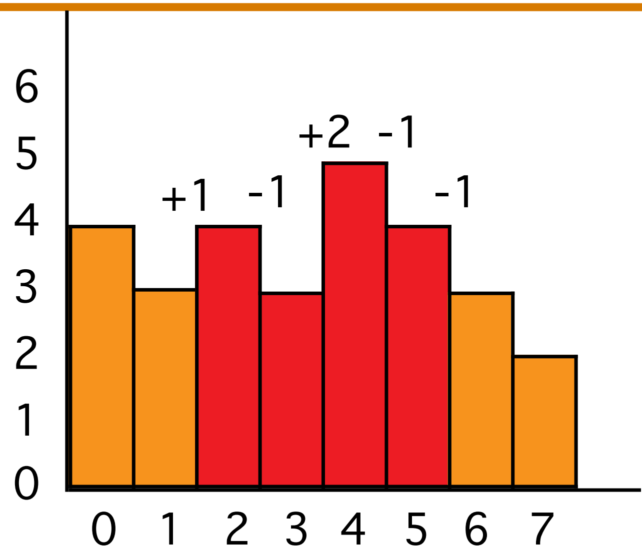


$$\begin{aligned}\text{Min }(&f_2^2+49-14f_2\\ &+ f_3^2+f_2^2+1-2f_3f_2 +2f_3-2f_2\\ &+ f_4^2+f_3^2+4-2f_3f_4 -4f_4+4f_3\\ &+ f_5^2+f_4^2+1-2f_5f_4 +2f_5-2f_4\\ &+ f_5^2+4-4f_5)\end{aligned}$$

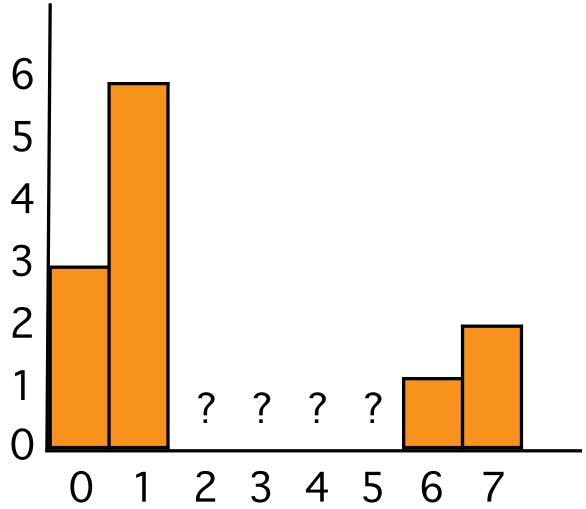
Denote it Q

# 1D example: derivatives

- **Copy**




**to**




**Min** $(f_2^2+49-14f_2$
$+ f_3^2+f_2^2+1-2f_3f_2 +2f_3-2f_2$
$+ f_4^2+f_3^2+4-2f_3f_4 -4f_4+4f_3$
$+ f_5^2+f_4^2+1-2f_5f_4 +2f_5-2f_4$
$+ f_5^2+4-4f_5)$

**Denote it Q**

$$\frac{dQ}{df_2} = 2f_2 + 2f_2 - 2f_3 - 16$$

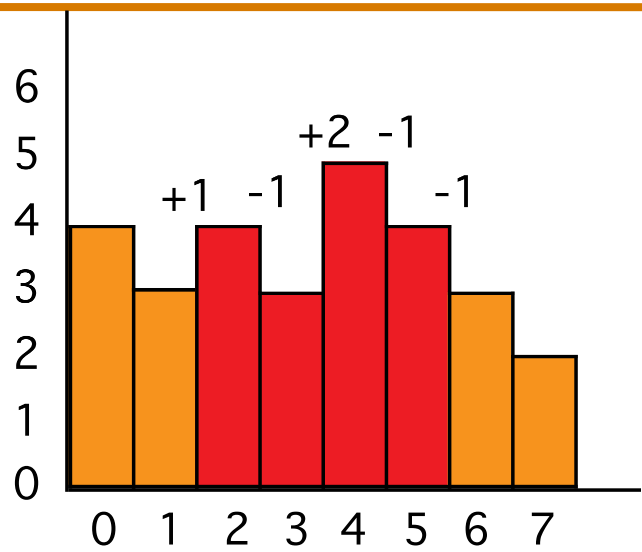
$$\frac{dQ}{df_3} = 2f_3 - 2f_2 + 2 + 2f_3 - 2f_4 + 4$$

$$\frac{dQ}{df_4} = 2f_4 - 2f_3 - 4 + 2f_4 - 2f_5 - 2$$

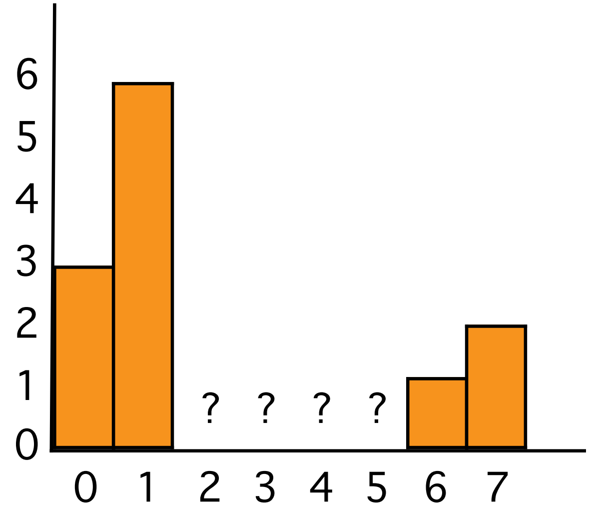
$$\frac{dQ}{df_5} = 2f_5 - 2f_4 + 2 + 2f_5 - 4$$

# 1D example: set derivatives to zero

- **Copy**




**to**



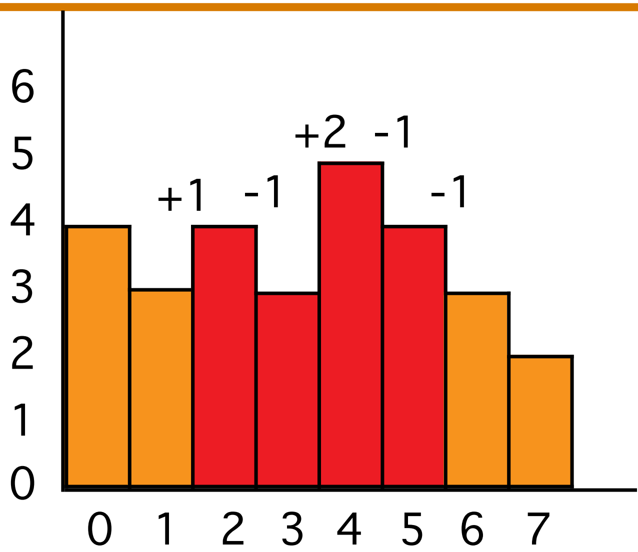
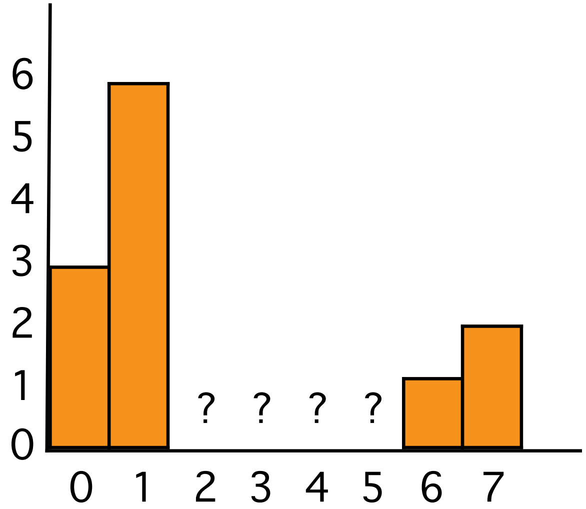

$$\frac{dQ}{df_2} = 2f_2 + 2f_2 - 2f_3 - 16$$

$$\frac{dQ}{df_3} = 2f_3 - 2f_2 + 2 + 2f_3 - 2f_4 + 4$$

$$\frac{dQ}{df_4} = 2f_4 - 2f_3 - 4 + 2f_4 - 2f_5 - 2$$

$$\frac{dQ}{df_5} = 2f_5 - 2f_4 + 2 + 2f_5 - 4$$

# 1D example: set derivatives to zero

- **Copy**  **to** 





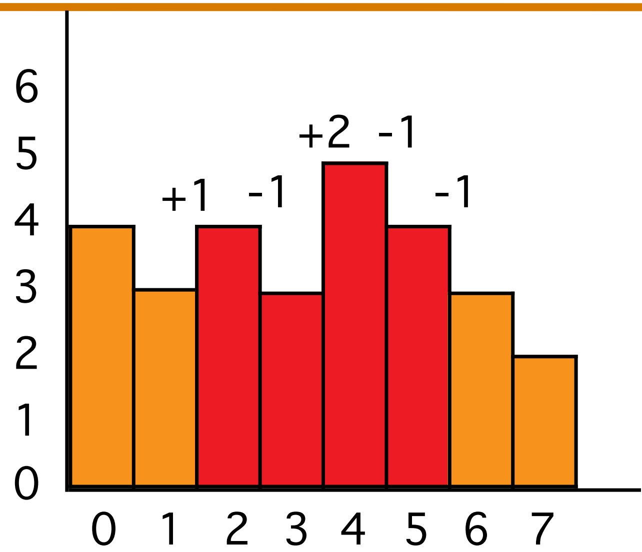
$$\frac{dQ}{df_2} = 2f_2 + 2f_2 - 2f_3 - 16 \quad = 0$$

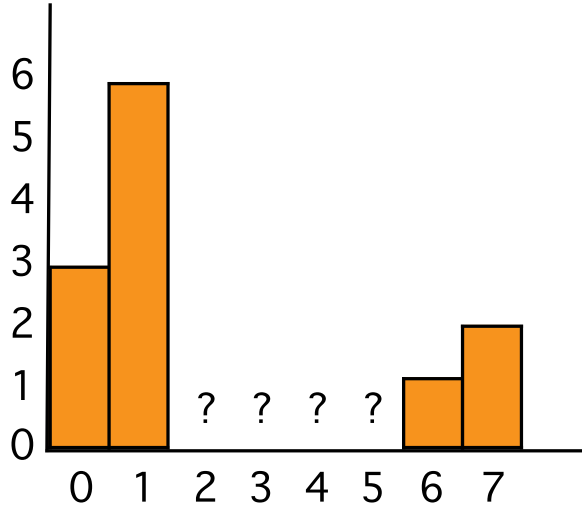
$$\frac{dQ}{df_3} = 2f_3 - 2f_2 + 2 + 2f_3 - 2f_4 + 4 \quad = 0$$

$$\frac{dQ}{df_4} = 2f_4 - 2f_3 - 4 + 2f_4 - 2f_5 - 2 \quad = 0$$

$$\frac{dQ}{df_5} = 2f_5 - 2f_4 + 2 + 2f_5 - 4 = 0$$

# 1D example: set derivatives to zero

- **Copy** 

**to** 


$$\frac{dQ}{df_2} = 2f_2 + 2f_2 - 2f_3 - 16 = 0$$

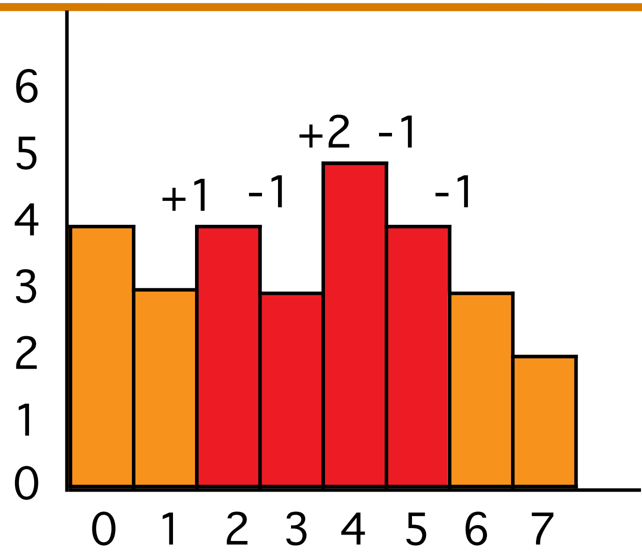
$$\frac{dQ}{df_3} = 2f_3 - 2f_2 + 2 + 2f_3 - 2f_4 + 4 = 0$$

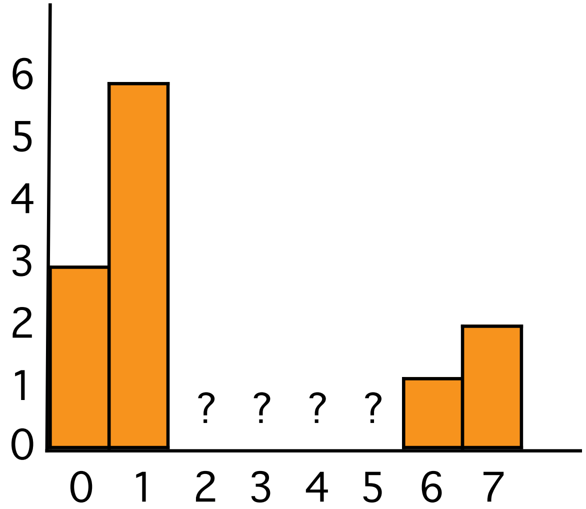
$$\frac{dQ}{df_4} = 2f_4 - 2f_3 - 4 + 2f_4 - 2f_5 - 2 = 0$$

$$\frac{dQ}{df_5} = 2f_5 - 2f_4 + 2 + 2f_5 - 4 = 0$$

$$\Longrightarrow$$

# 1D example: set derivatives to zero

- **Copy** 


**to** 


$$\frac{dQ}{df_2} = 2f_2 + 2f_2 - 2f_3 - 16 = 0$$

$$\frac{dQ}{df_3} = 2f_3 - 2f_2 + 2 + 2f_3 - 2f_4 + 4 = 0$$

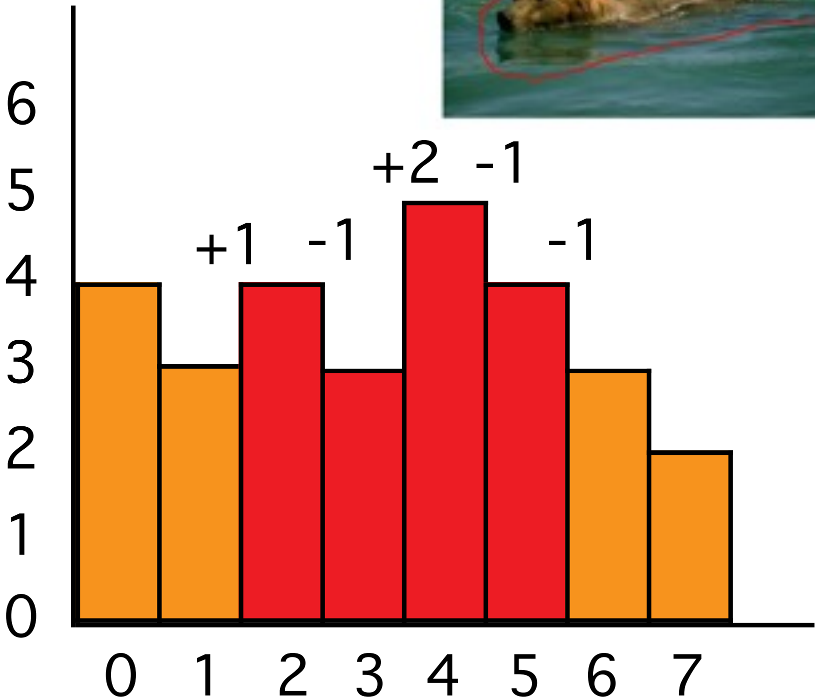
$$\frac{dQ}{df_4} = 2f_4 - 2f_3 - 4 + 2f_4 - 2f_5 - 2 = 0$$

$$\frac{dQ}{df_5} = 2f_5 - 2f_4 + 2 + 2f_5 - 4 = 0$$

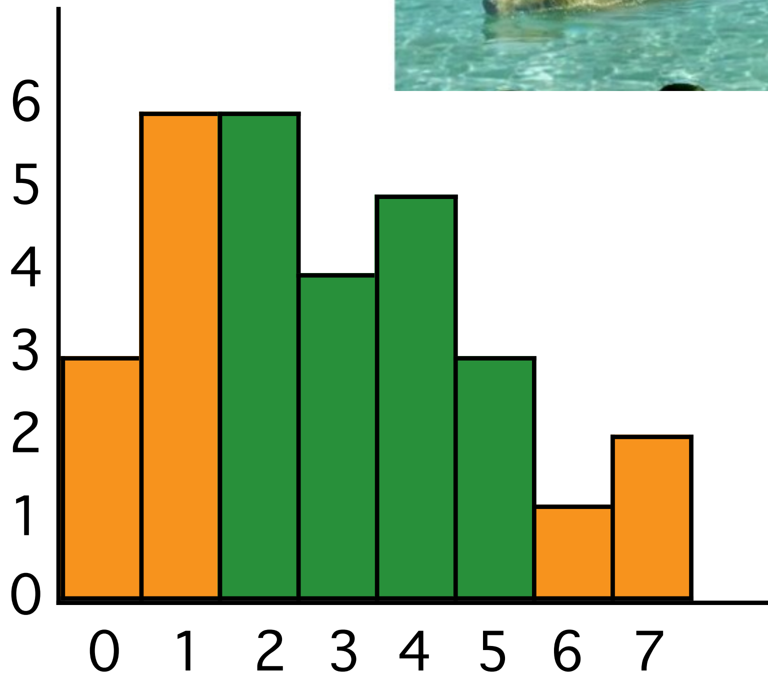
$$\Longrightarrow \begin{pmatrix} 4 & -2 & 0 & 0 \\ -2 & 4 & -2 & 0 \\ 0 & -2 & 4 & -2 \\ 0 & 0 & -2 & 4 \end{pmatrix} \begin{pmatrix} f_2 \\ f_3 \\ f_4 \\ f_5 \end{pmatrix} = \begin{pmatrix} 16 \\ -6 \\ 6 \\ 2 \end{pmatrix}$$

# 1D example recap

- **Copy**




**to**




==>

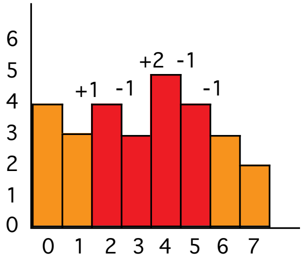
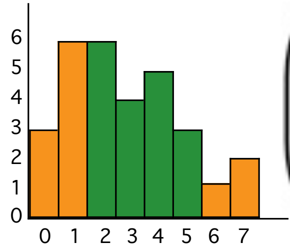
$$\begin{pmatrix} 4 & -2 & 0 & 0 \\ -2 & 4 & -2 & 0 \\ 0 & -2 & 4 & -2 \\ 0 & 0 & -2 & 4 \end{pmatrix} \begin{pmatrix} f_2 \\ f_3 \\ f_4 \\ f_5 \end{pmatrix} = \begin{pmatrix} 16 \\ -6 \\ 6 \\ 2 \end{pmatrix}$$

$$\begin{pmatrix} f_2 \\ f_3 \\ f_4 \\ f_5 \end{pmatrix} = \begin{pmatrix} 6 \\ 4 \\ 5 \\ 3 \end{pmatrix}$$

# Questions?

- **Recap:**
  - copy gradient, not pixel values
  - enforce boundary condition
  - solve linear least square:
    minimize square difference with source gradient

# 1D example: remarks

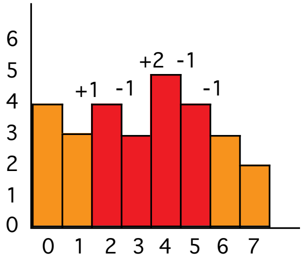
- **Copy**  **to** 

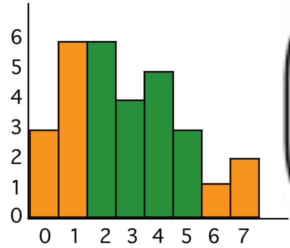




$$\begin{pmatrix} 4 & -2 & 0 & 0 \\ -2 & 4 & -2 & 0 \\ 0 & -2 & 4 & -2 \\ 0 & 0 & -2 & 4 \end{pmatrix} \begin{pmatrix} f_2 \\ f_3 \\ f_4 \\ f_5 \end{pmatrix} = \begin{pmatrix} 16 \\ -6 \\ 6 \\ 2 \end{pmatrix}$$

# 1D example: remarks

- **Copy** 

**to** 


$$\begin{pmatrix} 4 & -2 & 0 & 0 \\ -2 & 4 & -2 & 0 \\ 0 & -2 & 4 & -2 \\ 0 & 0 & -2 & 4 \end{pmatrix} \begin{pmatrix} f_2 \\ f_3 \\ f_4 \\ f_5 \end{pmatrix} = \begin{pmatrix} 16 \\ -6 \\ 6 \\ 2 \end{pmatrix}$$

- **Matrix is sparse**
  - many zero coefficients
  - because gradient only depends on neighboring pixels
- **Matrix is symmetric**
- **Everything is a multiple of 2**
  - because square and derivative of square
- **Matrix is a convolution (kernel -2 4 -2)**
  - all the rows are the same, just shifted
- **Matrix is independent of gradient field. Only RHS is**
- **Matrix is a second derivative**

# Questions?

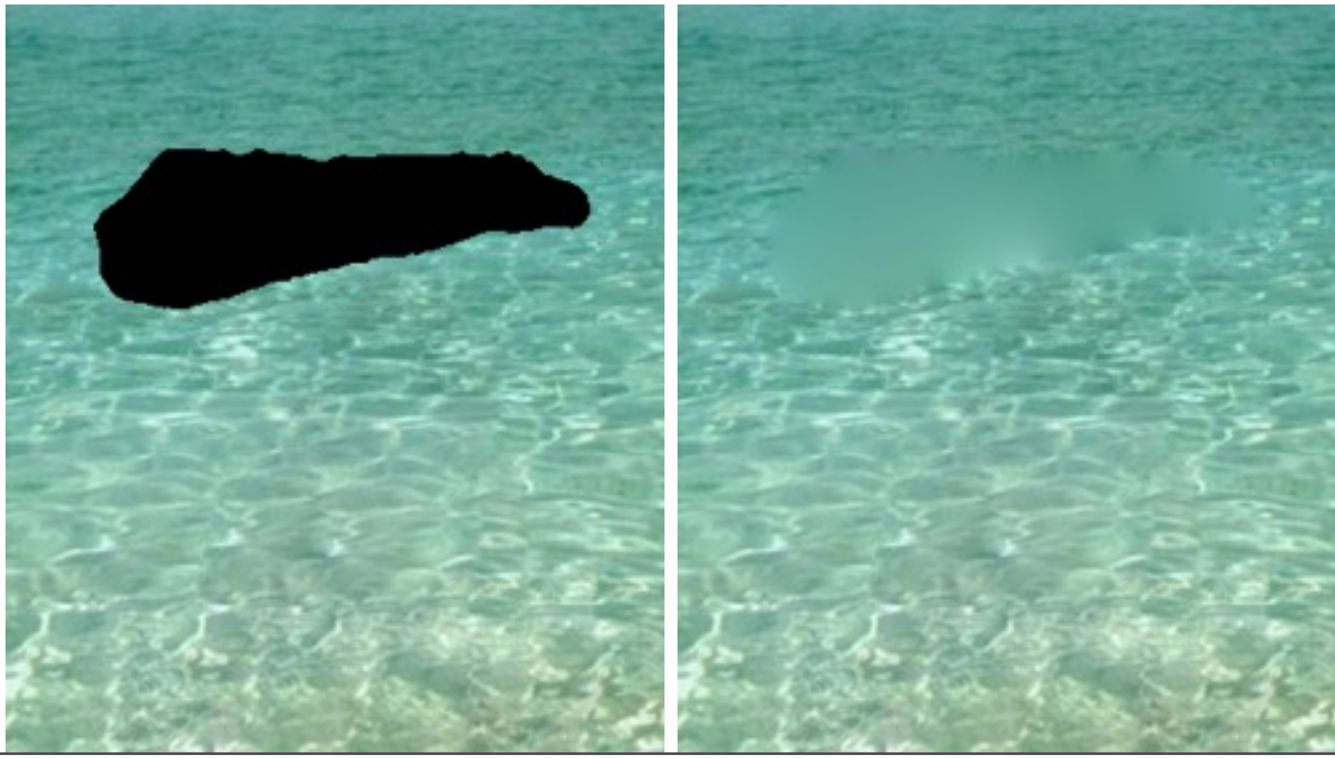
# Let's try to further analyze

- **What is a simple case?**

# Membrane interpolation

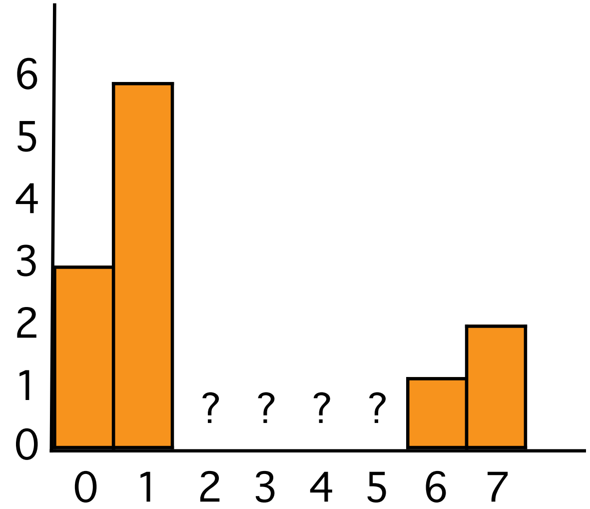
- **What if *v* is null?**
- **Laplace equation (a.k.a. membrane equation )**

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



# 1D example: minimization

- **Minimize derivatives to interpolate**




**Min $(f_2-f_1)^2$**

**$+ (f_3-f_2)^2$**

**$+ (f_4-f_3)^2$**

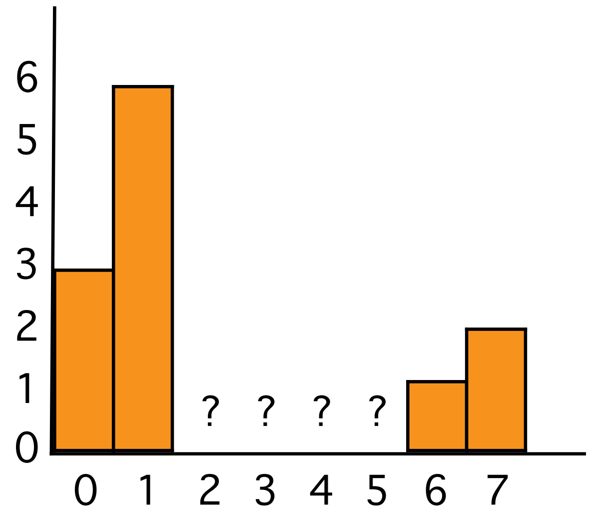
**$+ (f_5-f_4)^2$**

**$+ (f_6-f_5)^2$**

With
$f_1=6$
$f_6=1$

# 1D example: derivatives

- **Minimize derivatives to interpolate**




**Min** $(f_2^2+36-12f_2$

$+ f_3^2+f_2^2-2f_3f_2$

$+ f_4^2+f_3^2-2f_3f_4$

$+ f_5^2+f_4^2-2f_5f_4$

$+ f_5^2+1-2f_5)$

**Denote it Q**

$$\frac{dQ}{df_2} = 2f_2 + 2f_2 - 2f_3 - 12$$

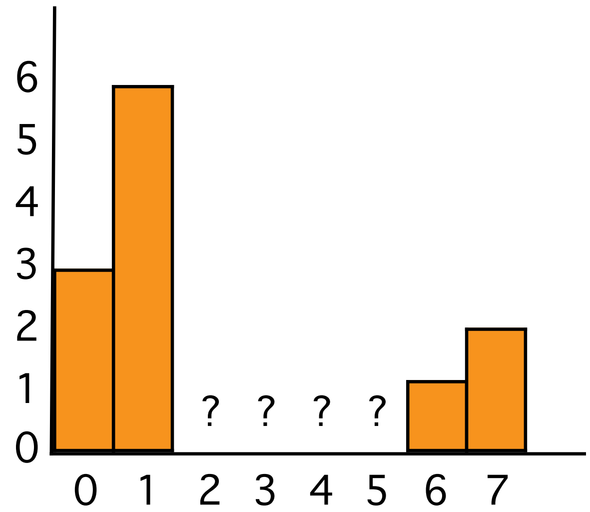
$$\frac{dQ}{df_3} = 2f_3 - 2f_2 + 2f_3 - 2f_4$$

$$\frac{dQ}{df_4} = 2f_4 - 2f_3 + 2f_4 - 2f_5$$

$$\frac{dQ}{df_5} = 2f_5 - 2f_4 + 2f_5 - 2$$

# 1D example: set derivatives to zero

- **Minimize derivatives to interpolate**




$$\frac{dQ}{df_2} = 2f_2 + 2f_2 - 2f_3 - 12$$

$$\frac{dQ}{df_3} = 2f_3 - 2f_2 + 2f_3 - 2f_4$$

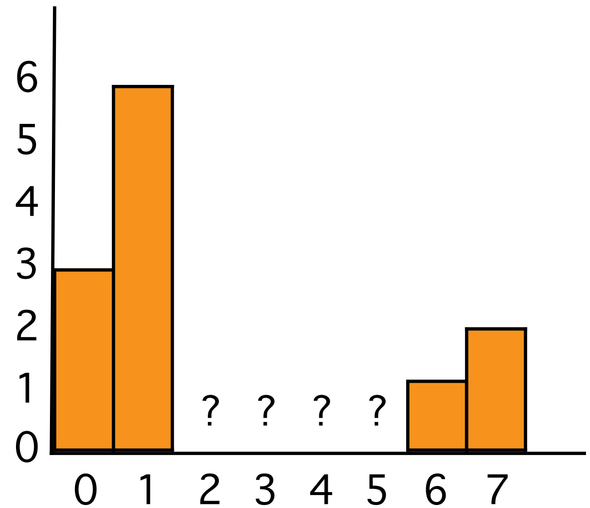
$$\frac{dQ}{df_4} = 2f_4 - 2f_3 + 2f_4 - 2f_5$$

$$\frac{dQ}{df_5} = 2f_5 - 2f_4 + 2f_5 - 2$$

==>

# 1D example: set derivatives to zero

- **Minimize derivatives to interpolate**




$$\frac{dQ}{df_2} = 2f_2 + 2f_2 - 2f_3 - 12$$

$$\frac{dQ}{df_3} = 2f_3 - 2f_2 + 2f_3 - 2f_4$$

$$\frac{dQ}{df_4} = 2f_4 - 2f_3 + 2f_4 - 2f_5$$

$$\frac{dQ}{df_5} = 2f_5 - 2f_4 + 2f_5 - 2$$

==> $$\begin{pmatrix} 4 & -2 & 0 & 0 \\ -2 & 4 & -2 & 0 \\ 0 & -2 & 4 & -2 \\ 0 & 0 & -2 & 4 \end{pmatrix} \begin{pmatrix} f_2 \\ f_3 \\ f_4 \\ f_5 \end{pmatrix} = \begin{pmatrix} 12 \\ 0 \\ 0 \\ 2 \end{pmatrix}$$

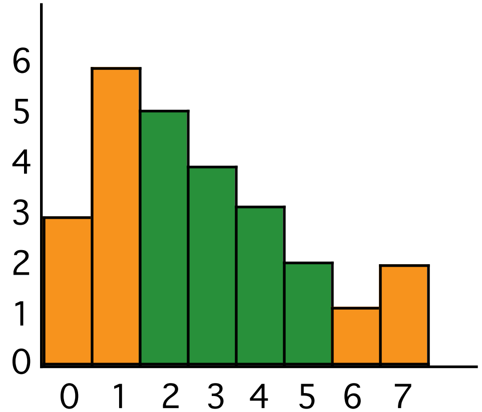
# 1D example

- **Minimize derivatives to interpolate**

- **Pretty much says that second derivative should be zero**

  (-1 2 -1)

  is a second derivative filter

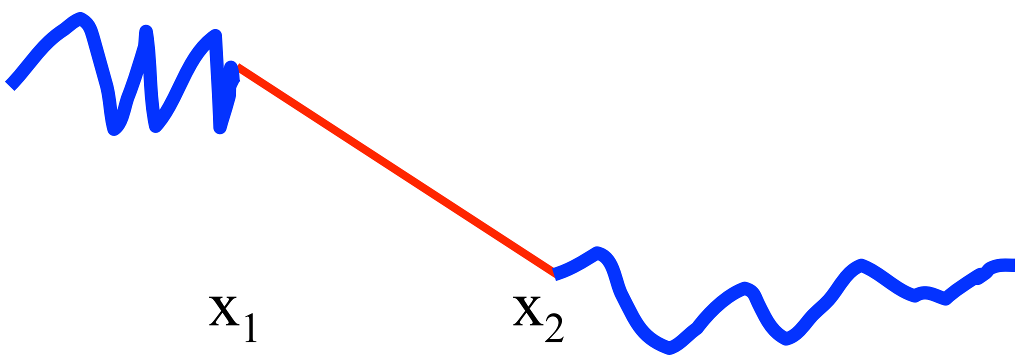



$$\begin{pmatrix} 4 & -2 & 0 & 0 \\ -2 & 4 & -2 & 0 \\ 0 & -2 & 4 & -2 \\ 0 & 0 & -2 & 4 \end{pmatrix} \begin{pmatrix} f_2 \\ f_3 \\ f_4 \\ f_5 \end{pmatrix} = \begin{pmatrix} 12 \\ 0 \\ 0 \\ 2 \end{pmatrix}$$

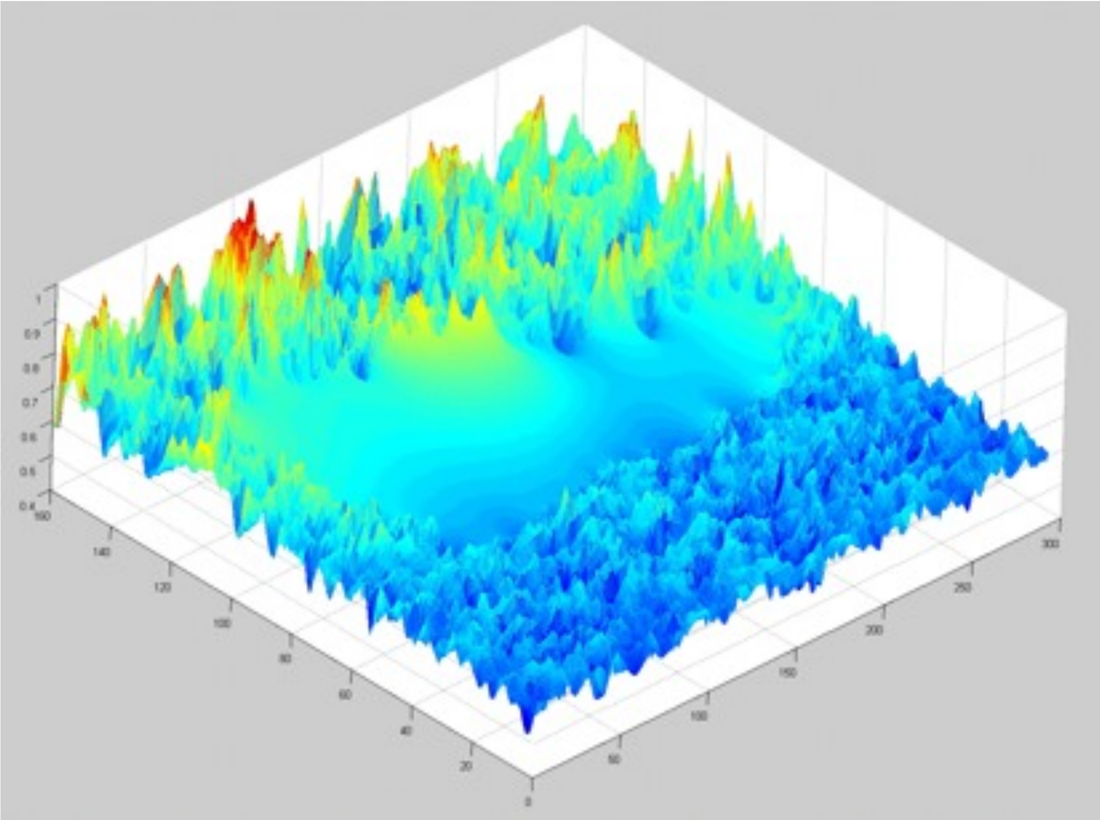
$$\begin{pmatrix} f_2 \\ f_3 \\ f_4 \\ f_5 \end{pmatrix} = \begin{pmatrix} 5 \\ 4 \\ 3 \\ 2 \end{pmatrix}$$

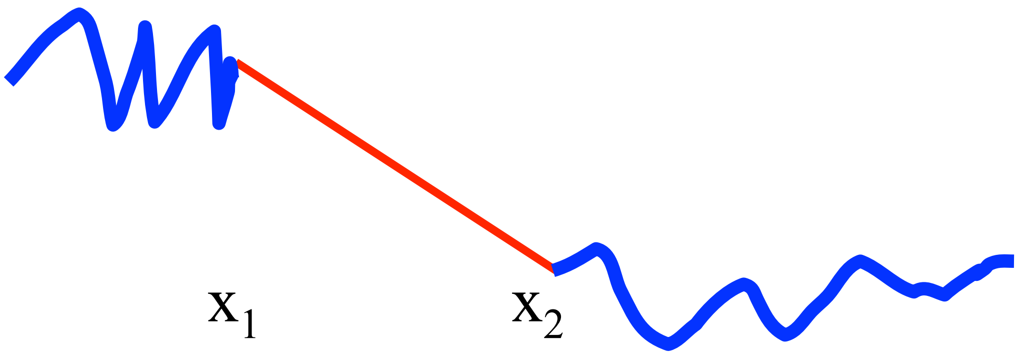
# Intuition

- **In 1D; just linear interpolation!**
- **Locally, if the second derivative was not zero, this would mean that the first derivative is varying, which is bad since we want $(\nabla f)^2$ to be minimized**
- **Note that, in 1D: by setting f'', we leave two degrees of freedom. This is exactly what we need to control the boundary condition at $x_1$ and $x_2$**

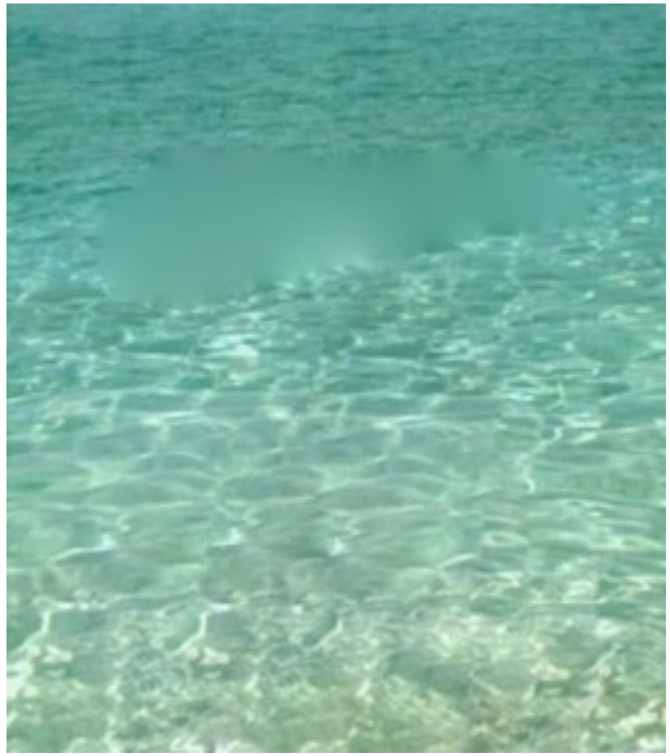


# In 2D: membrane interpolation






Not as simple



# Questions?

# What if $v$ is not null?

sources/destinations

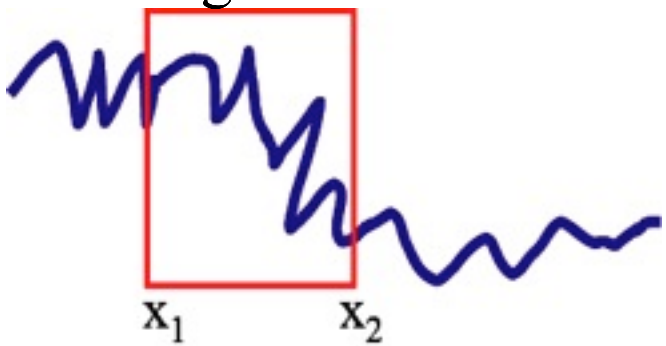
seamless cloning

# What if *v* is not null?

- **1D case**

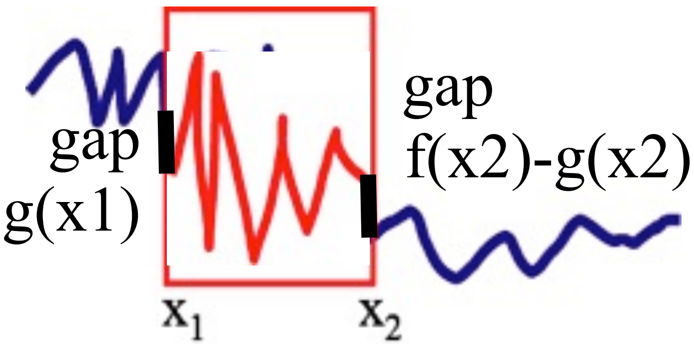
Seamlessly paste source g

onto target f*



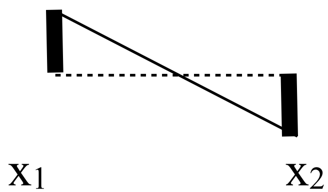

Just add a linear function so that the boundary condition is respected

gap f(x1)-g(x1)

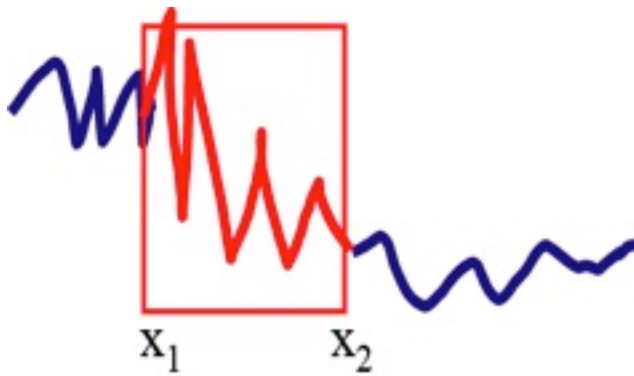



gap f(x2)-g(x2)

correction $\hat{f}$




solution $f = \hat{f} + g$



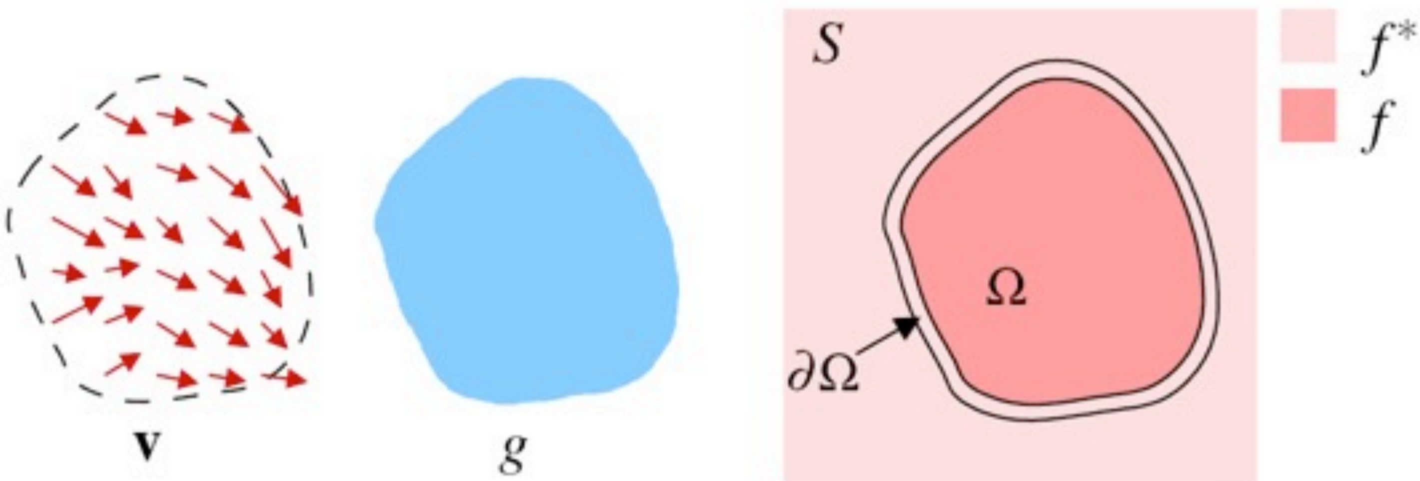
# Recap 1D case

- **Poisson clone of g into f* between x1 and x2**
- **if g is null, simple linear function**
  - $f(x) = (x2-x)/(x2-x1)f^*(x1)+(x-x1)/(x2-x1)f^*(x2)$
- **otherwise, add a correction function to g in order to linearly interpolate between $f^*(x1)-g(x1)$ and $f^*(x2)-g(x2)$**
  - $f(x)=\hat{f}(x)+g(x)$
  - where
    $\hat{f}(x) = (x2-x)/(x2-x1)(f^*(x1)-g(x1)$
    $+(x-x1)/(x2-x1)(f^*(x2)-g(x2))$
  - Note that boundary conditions are respected and the difference to g is spread uniformly

# In 2D, if *v* is conservative

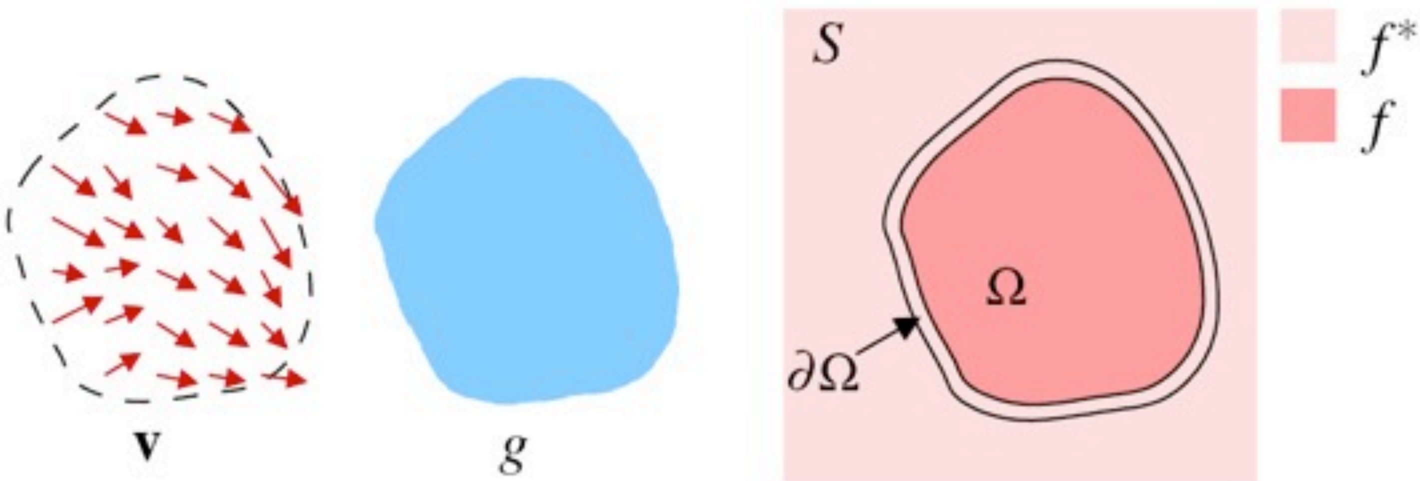
- **If *v* is the gradient of an image g (it is conservative)**
- **Correction function $\widehat{f}$ so that** $f = g + \widehat{f}$
- $\widehat{f}$ **performs membrane interpolation over Ω:**

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



# In 2D, if *v* is NOT conservative

- Also need to project the vector field v to a conservative field
- And do the membrane thing
- Of course, we do not need to worry about it, it's all handled naturally by the least square approach



# Exploited in

SIGGRAPH 2009

## Coordinates for Instant Image Cloning


Zeev Farbman, The Hebrew University
Gil Hoffer, Tel Aviv University
Yaron Lipman, Princeton University
Daniel Cohen-Or, Tel Aviv University
Dani Lischinski, The Hebrew University


(a) Source patch (b) Laplace membrane (c) Mean-value membrane

(d) Target image (e) Poisson cloning (f) Mean-value cloning

# Questions?

# Recap

- **Find image whose gradient best approximates the input gradient**
  - least square Minimization
- **Discrete case: turns into linear equation**
  - Set derivatives to zero
  - Derivatives of quadratic ==> linear
- **When gradient is null, membrane interpolation**
  - Linear interpolation in 1D

# Fourier interpretation

# Fourier interpretation

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

# Fourier interpretation

- **Least square on gradient** $\min_f \iint_\Omega |\nabla f - \mathbf{v}|^2 \text{ with } f|_{\partial\Omega} = f^*|_{\partial\Omega}$
- **Parseval anybody?**

# Fourier interpretation

- **Least square on gradient** $\min_f \iint_\Omega |\nabla f - \mathbf{v}|^2 \text{ with } f|_{\partial\Omega} = f^*|_{\partial\Omega}$
- **Parseval anybody?**
  - Integral of squared stuff is the same in Fourier and primal

# Fourier interpretation

- **Least square on gradient** $\min_f \iint_\Omega |\nabla f - \mathbf{v}|^2 \text{ with } f|_{\partial\Omega} = f^*|_{\partial\Omega}$
- **Parseval anybody?**
  - Integral of squared stuff is the same in Fourier and primal
- **What is the gradient/derivative in Fourier?**

# Fourier interpretation

- **Least square on gradient** $\min_f \iint_\Omega |\nabla f - \mathbf{v}|^2 \text{ with } f|_{\partial\Omega} = f^*|_{\partial\Omega}$
- **Parseval anybody?**
  - Integral of squared stuff is the same in Fourier and primal
- **What is the gradient/derivative in Fourier?**
  - Multiply coefficients by frequency and i

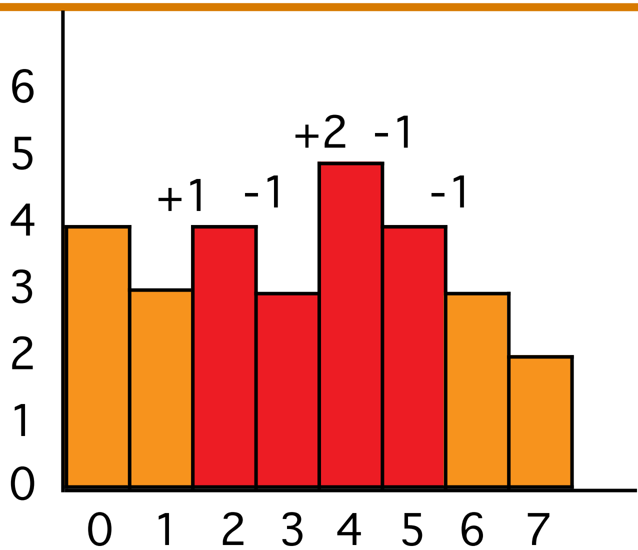
# Fourier interpretation

- **Least square on gradient** $\min_f \iint_\Omega |\nabla f - \mathbf{v}|^2 \text{ with } f|_{\partial\Omega} = f^*|_{\partial\Omega}$
- **Parseval anybody?**
  - Integral of squared stuff is the same in Fourier and primal
- **What is the gradient/derivative in Fourier?**
  - Multiply coefficients by frequency and i
- **Seen in Fourier domain, Poisson editing does a weighted least square of the image where low frequencies have a small weight and high frequencies a big weight**

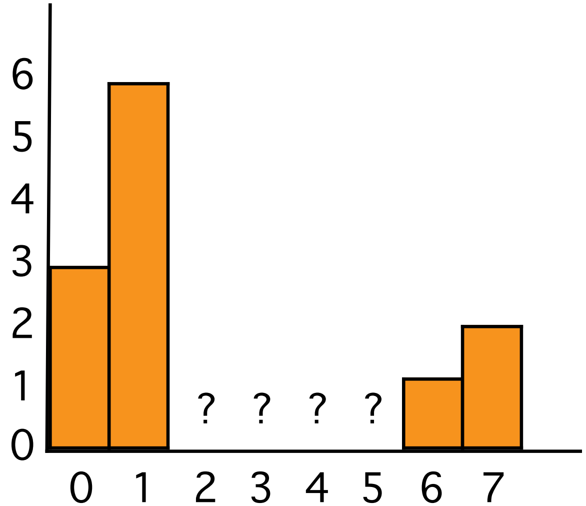
# Questions?

# Discrete solver: Recall 1D

- **Copy**




**to**




$$\frac{dQ}{df_2} = 2f_2 + 2f_2 - 2f_3 - 16$$

$$\frac{dQ}{df_3} = 2f_3 - 2f_2 + 2 + 2f_3 - 2f_4 + 4$$

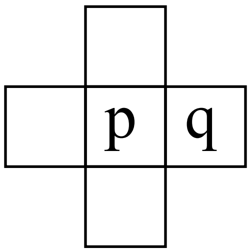
$$\frac{dQ}{df_4} = 2f_4 - 2f_3 - 4 + 2f_4 - 2f_5 - 2$$

$$\frac{dQ}{df_5} = 2f_5 - 2f_4 + 2 + 2f_5 - 4$$

==> $$\begin{pmatrix} 4 & -2 & 0 & 0 \\ -2 & 4 & -2 & 0 \\ 0 & -2 & 4 & -2 \\ 0 & 0 & -2 & 4 \end{pmatrix} \begin{pmatrix} f_2 \\ f_3 \\ f_4 \\ f_5 \end{pmatrix} = \begin{pmatrix} 16 \\ -6 \\ 6 \\ 2 \end{pmatrix}$$

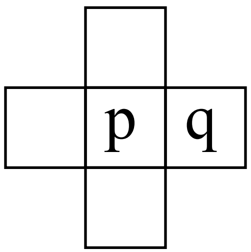
# Discrete Poisson solver

- **Minimize variational problem** $\min_f \iint_\Omega |\nabla f - \mathbf{v}|^2 \text{ with } f|_{\partial\Omega} = f^*|_{\partial\Omega},$
- **Discretize derivatives**
  - Finite differences over pairs of pixel neighbors
  - We are going to work using pairs of pixels



# Discrete Poisson solver

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



# Discrete Poisson solver

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

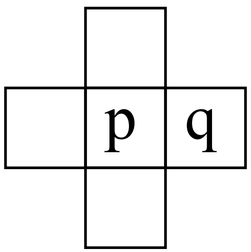
Discretized gradient

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

(all pairs that are in Ω)

Discretized v: g(p)-g(q)

Boundary condition



# Discrete Poisson solver

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

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

Discretized gradient

(all pairs that are in Ω)

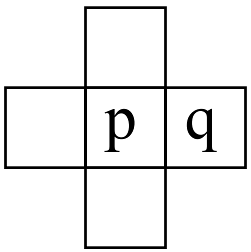
Discretized v: g(p)-g(q)

Boundary condition

- **Derive, rearrange and call $N_p$ the neighbors of p**

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

Only for boundary pixels



# Discrete Poisson solver

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

Discretized gradient

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

(all pairs that are in Ω)

Discretized v: g(p)-g(q)

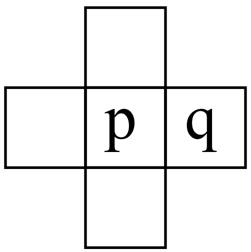
Boundary condition

- **Derive, rearrange and call $N_p$ the neighbors of p**

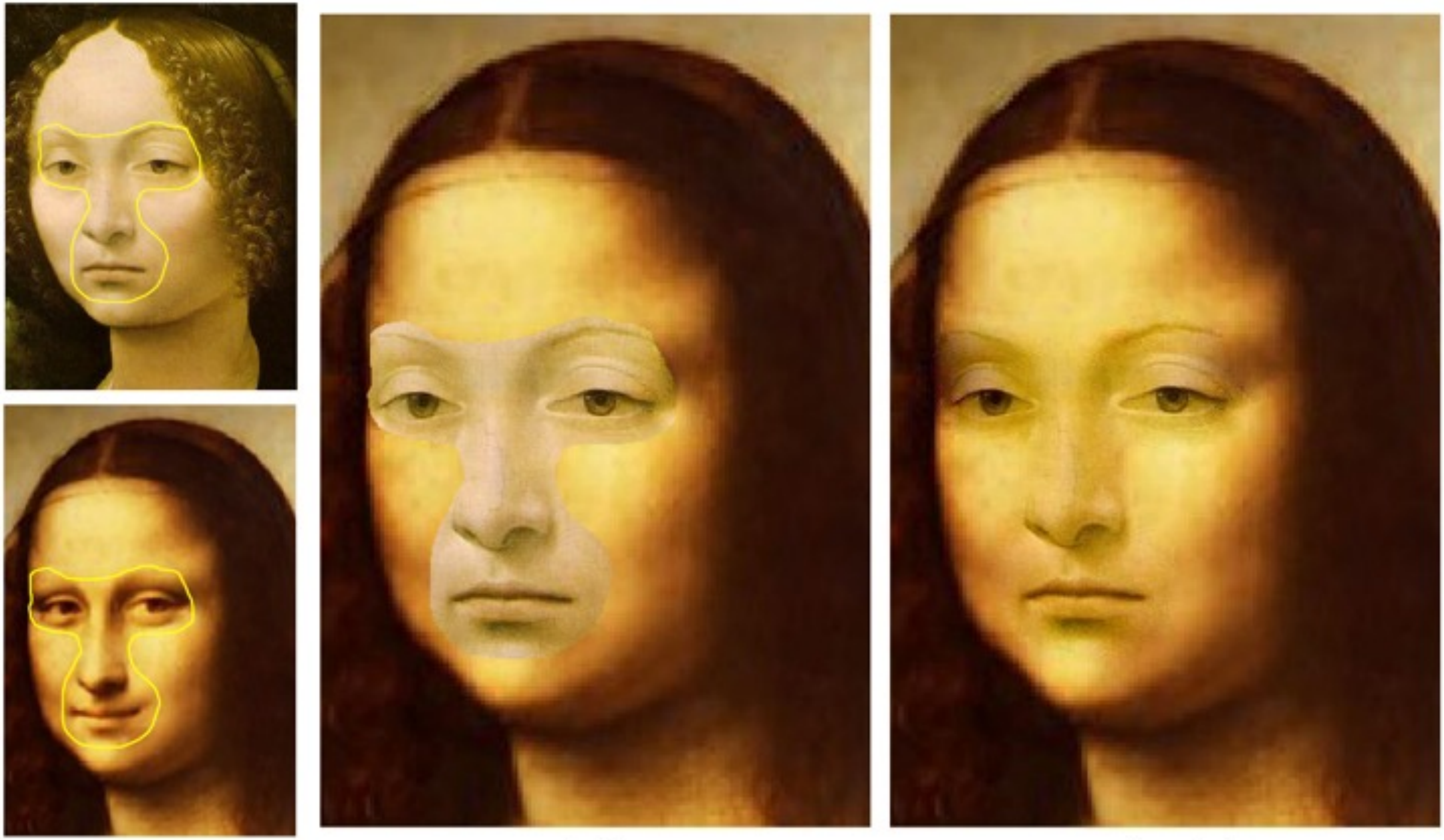
$$\text{for all } p \in \Omega, \quad |N_p| f_p - \sum_{q \in N_p \cap \Omega} f_q = \sum_{q \in N_p \cap \partial\Omega} f_q^* + \sum_{q \in N_p} v_{pq}$$

Only for boundary pixels

- **Big yet sparse linear system**



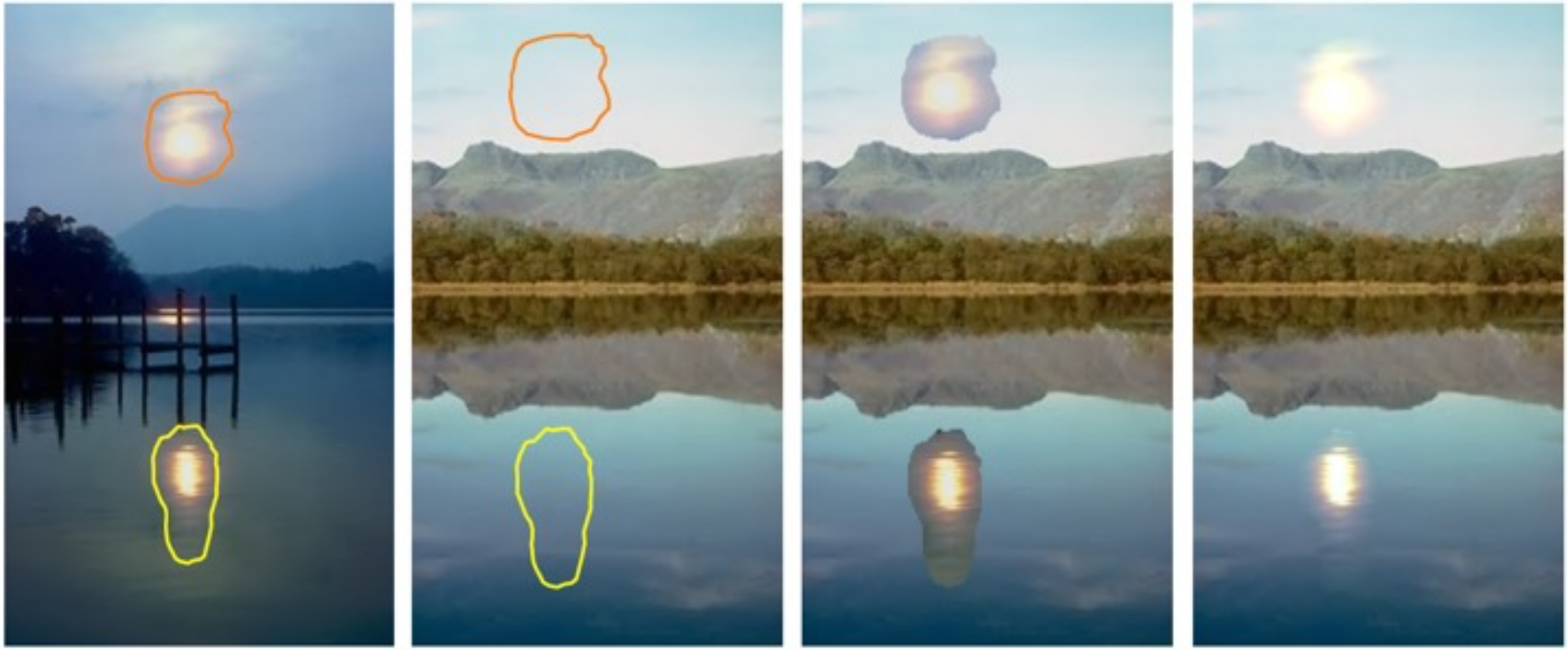
# Result (eye candy)



source/destination

cloning

seamless cloning



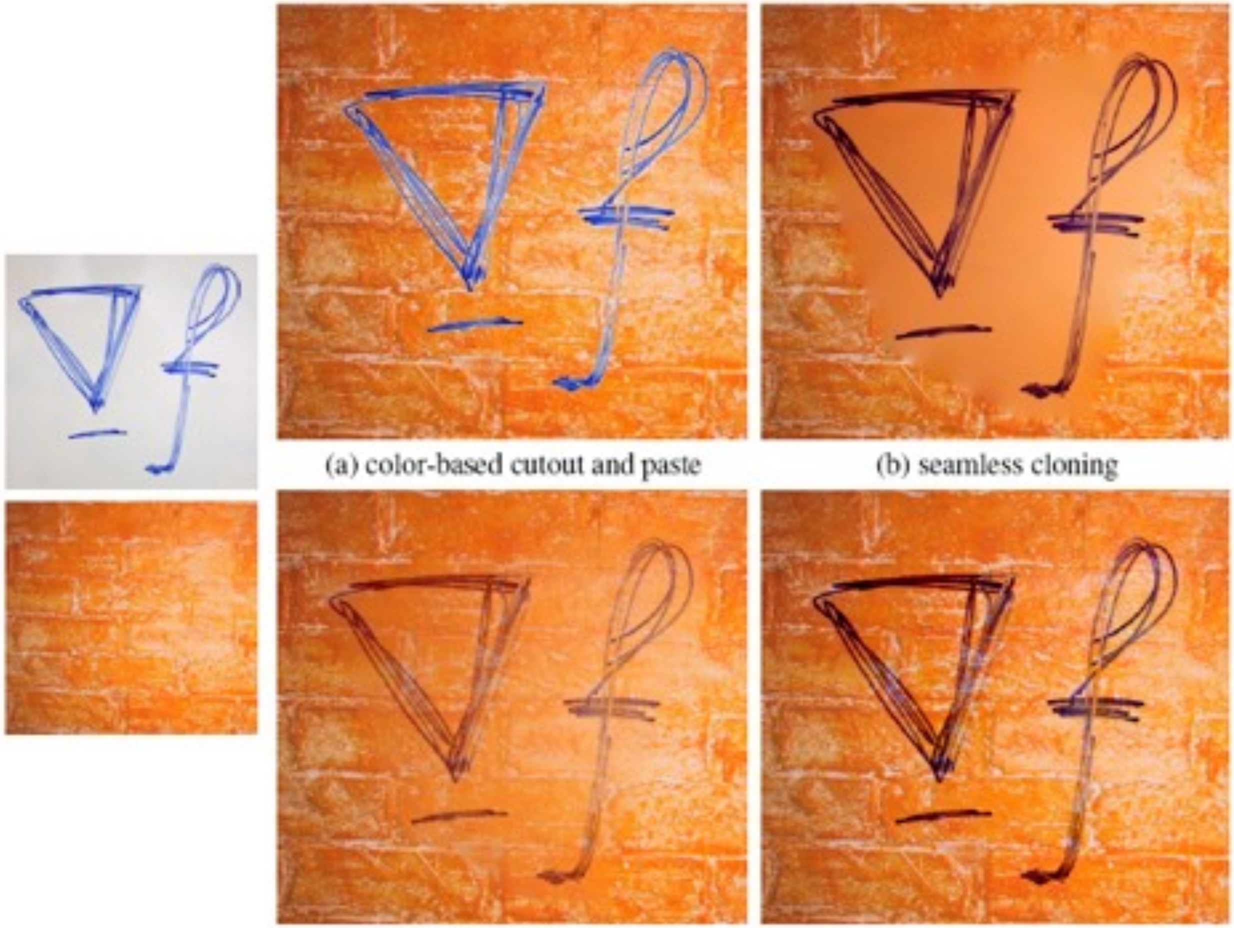
sources destinations cloning seamless cloning

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.

# Manipulate the gradient

- **Mix gradients of g & f: take the max**


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.



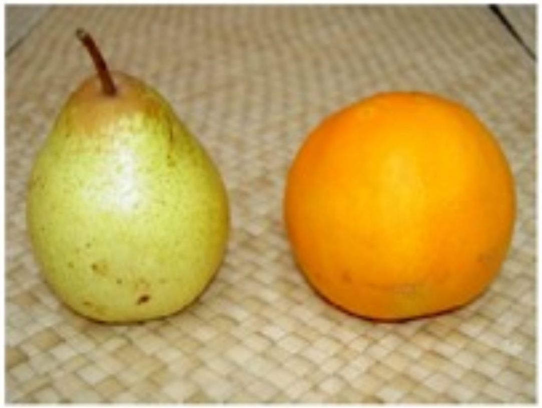
(a) color-based cutout and paste

(b) seamless cloning

(c) seamless cloning and destination averaged

(d) mixed seamless cloning

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.



swapped textures

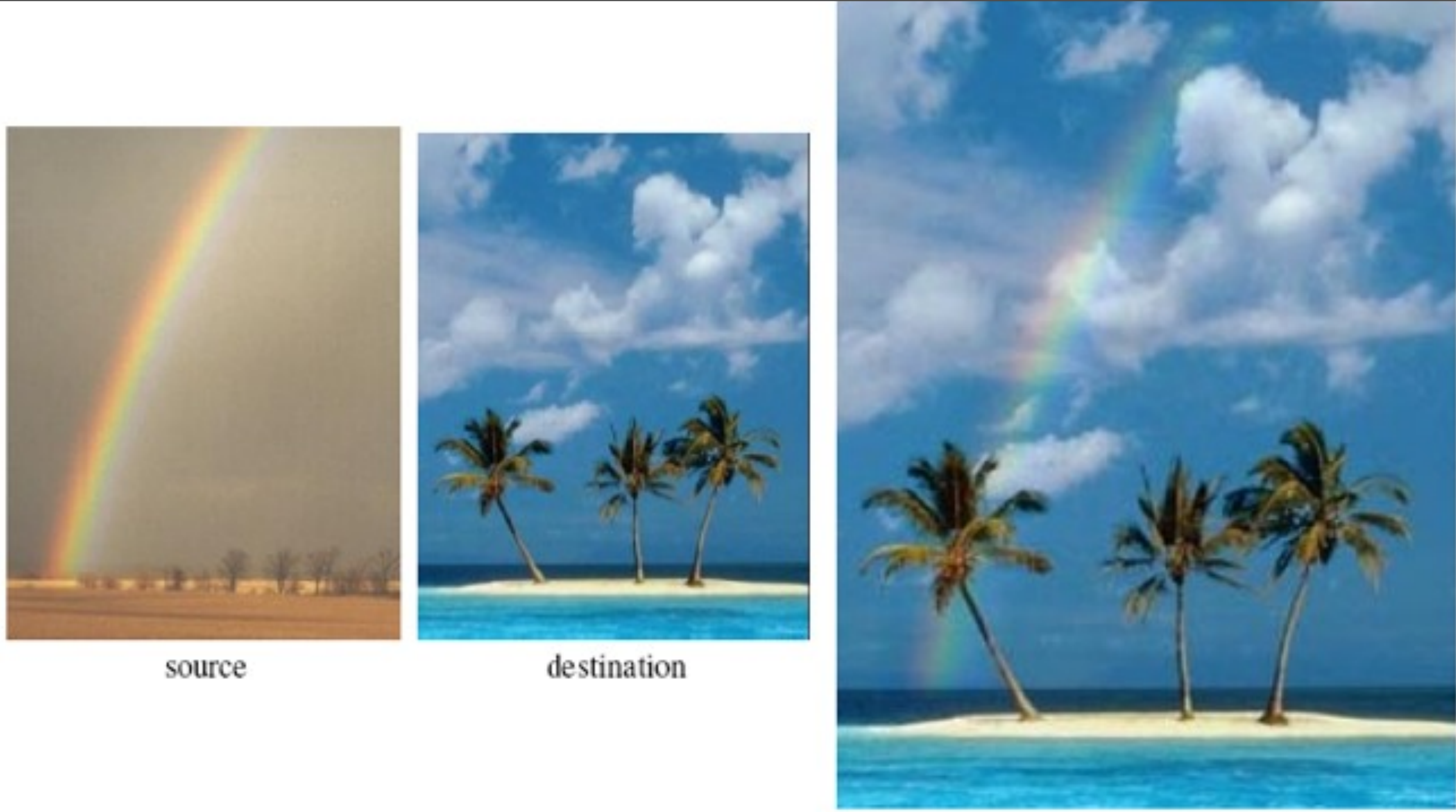



Figure 7: **Inserting transparent objects**. Mixed seamless cloning facilitates the transfer of partly transparent objects, such as the rainbow in this example. The non-linear mixing of gradient fields picks out whichever of source or destination structure is the more salient at each location.

# Questions?

# Issues with Poisson cloning

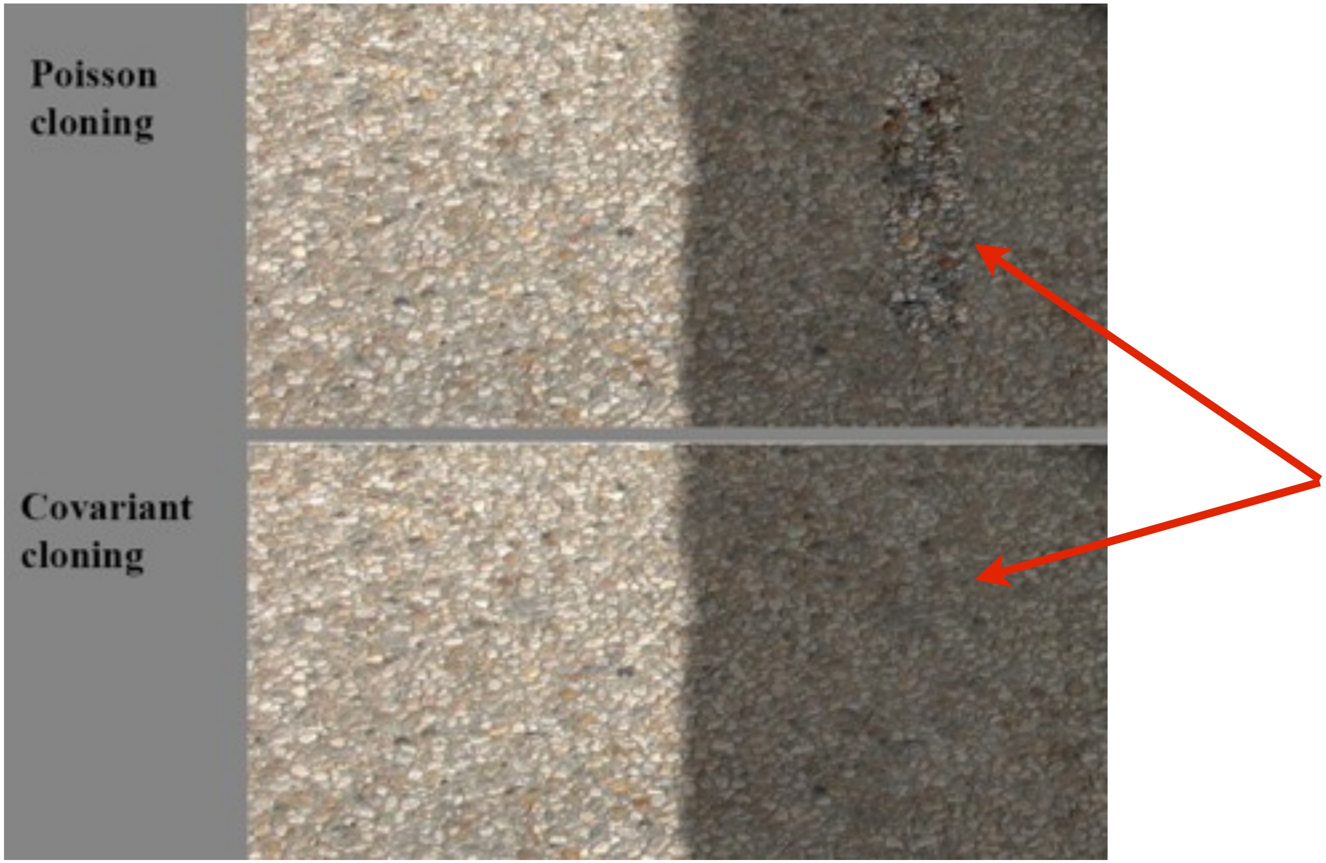
- Colors
- Contrast
- The backgrounds in f & g should be similar

# Improvement: local contrast

- **Use the log**
- **Or use covariant derivatives (next slides)**

# Covariant derivatives & Photoshop

- **Photoshop Healing brush**
- **Developed independently from Poisson editing by Todor Georgiev (Adobe)**




From Todor Georgiev's slides http://photo.csail.mit.edu/posters/todor_slides.pdf

# Seamless Image Stitching in the Gradient Domain

- Anat Levin, Assaf Zomet, Shmuel Peleg, and Yair Weiss
  http://www.cs.huji.ac.il/~alevin/papers/eccv04-blending.pdf
  http://eprints.pascal-network.org/archive/00001062/01/tips05-blending.pdf
- **Various strategies (optimal cut, feathering)**

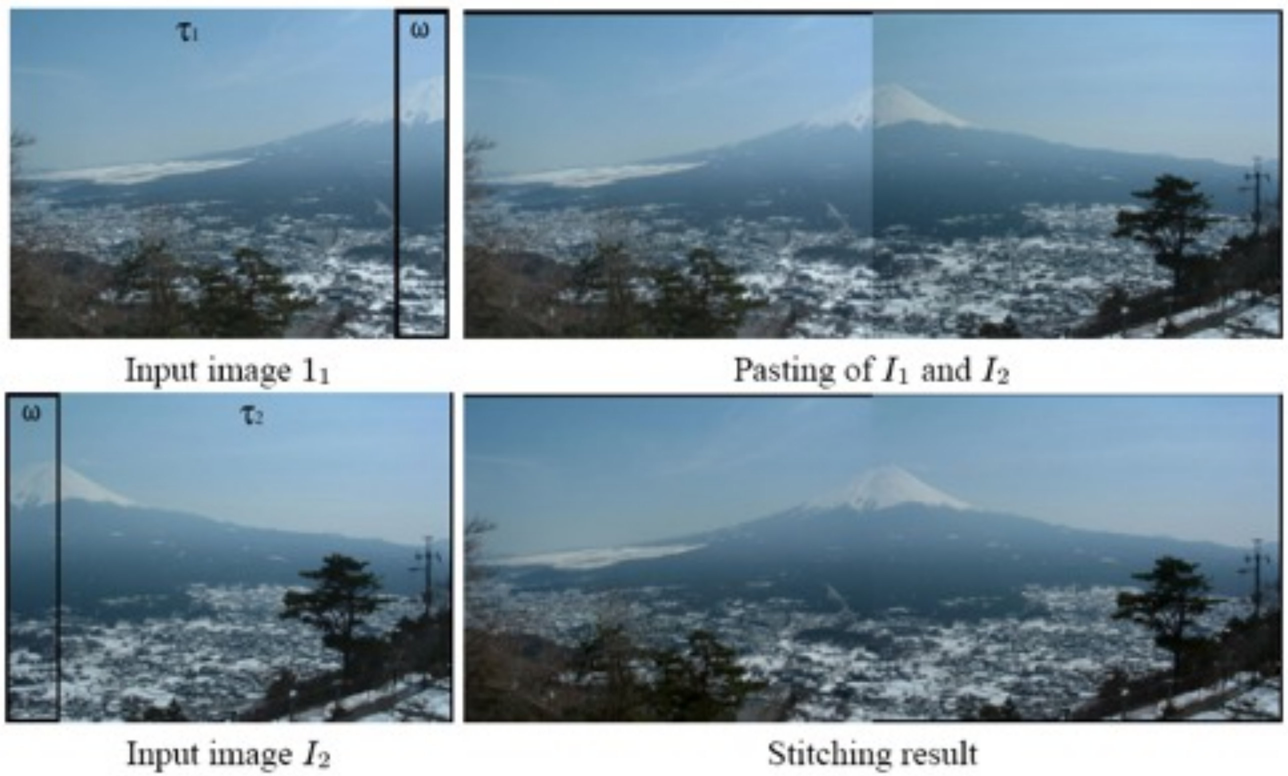



**Fig. 1.** Image stitching. On the left are the input images. $\omega$ is the overlap region. On top right is a simple pasting of the input images. On the bottom right is the result of the GIST1 algorithm.

# Photomontage

- [http://grail.cs.washington.edu/projects/photomontage/photomontage.pdf](http://grail.cs.washington.edu/projects/photomontage/photomontage.pdf)

**Figure 6** We use a set of portraits (first row) to mix and match facial features, to either improve a portrait, or create entirely new people. The faces are first hand-aligned, for example, to place all the noses in the same location. In the first two images in the second row, we replace the closed eyes of a portrait with the open eyes of another. The user paints strokes with the *designated source* objective to specify desired features. Next, we create a fictional person by combining three source portraits. Gradient-domain fusion is used to smooth out skin tone differences. Finally, we show two additional mixed portraits.

# Elder's edge representation

- http://elderlab.yorku.ca/~elder/publications/journals/ElderPAMI01.pdf

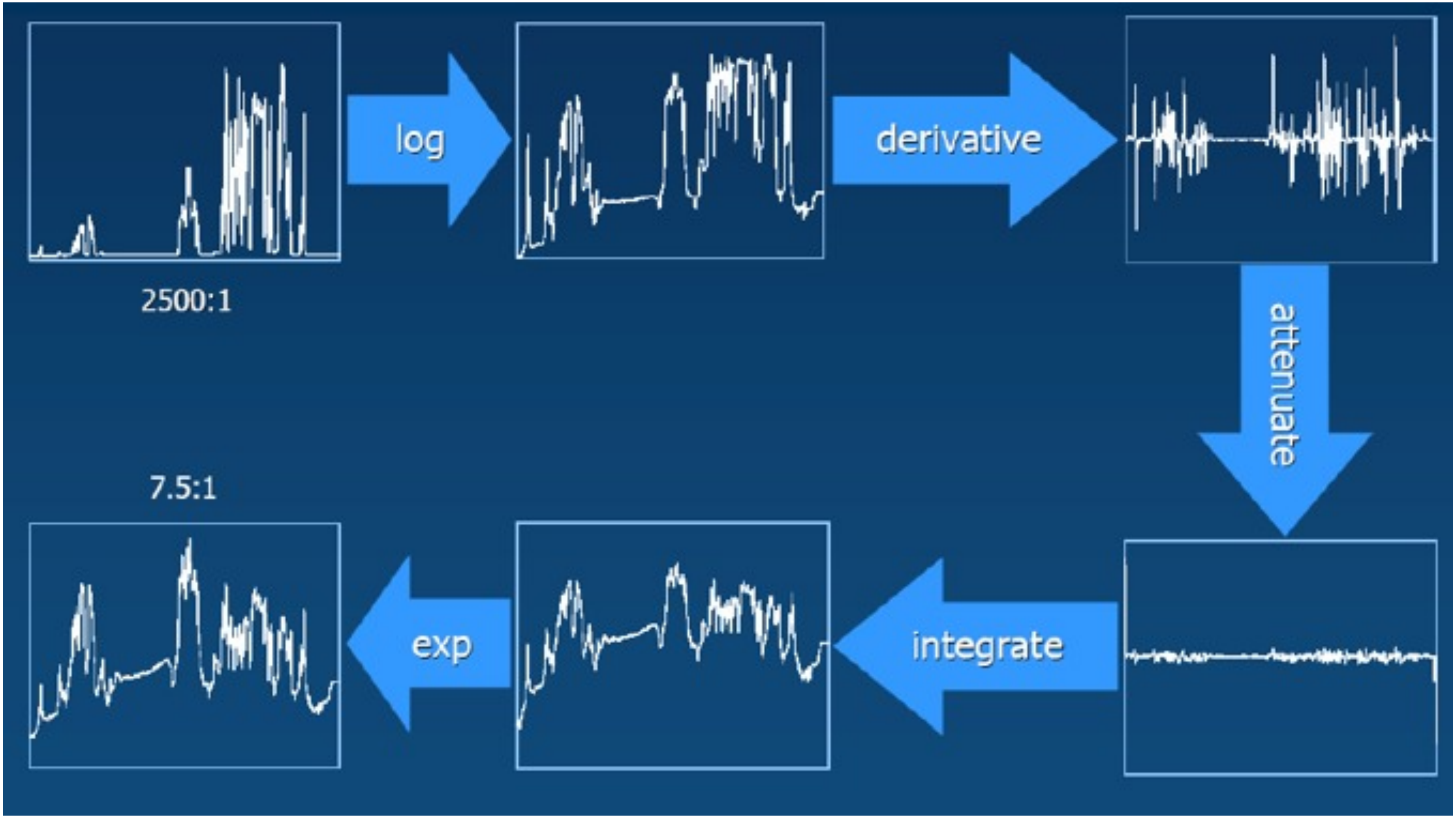
# Reduce big gradients

- **Dynamic range compression**
- **See Fattal et al. 2002**

Figure 10: **Local illumination changes**. Applying an appropriate non-linear transformation to the gradient field inside the selection and then integrating back with a Poisson solver, modifies locally the apparent illumination of an image. This is useful to highlight under-exposed foreground objects or to reduce specular reflections.

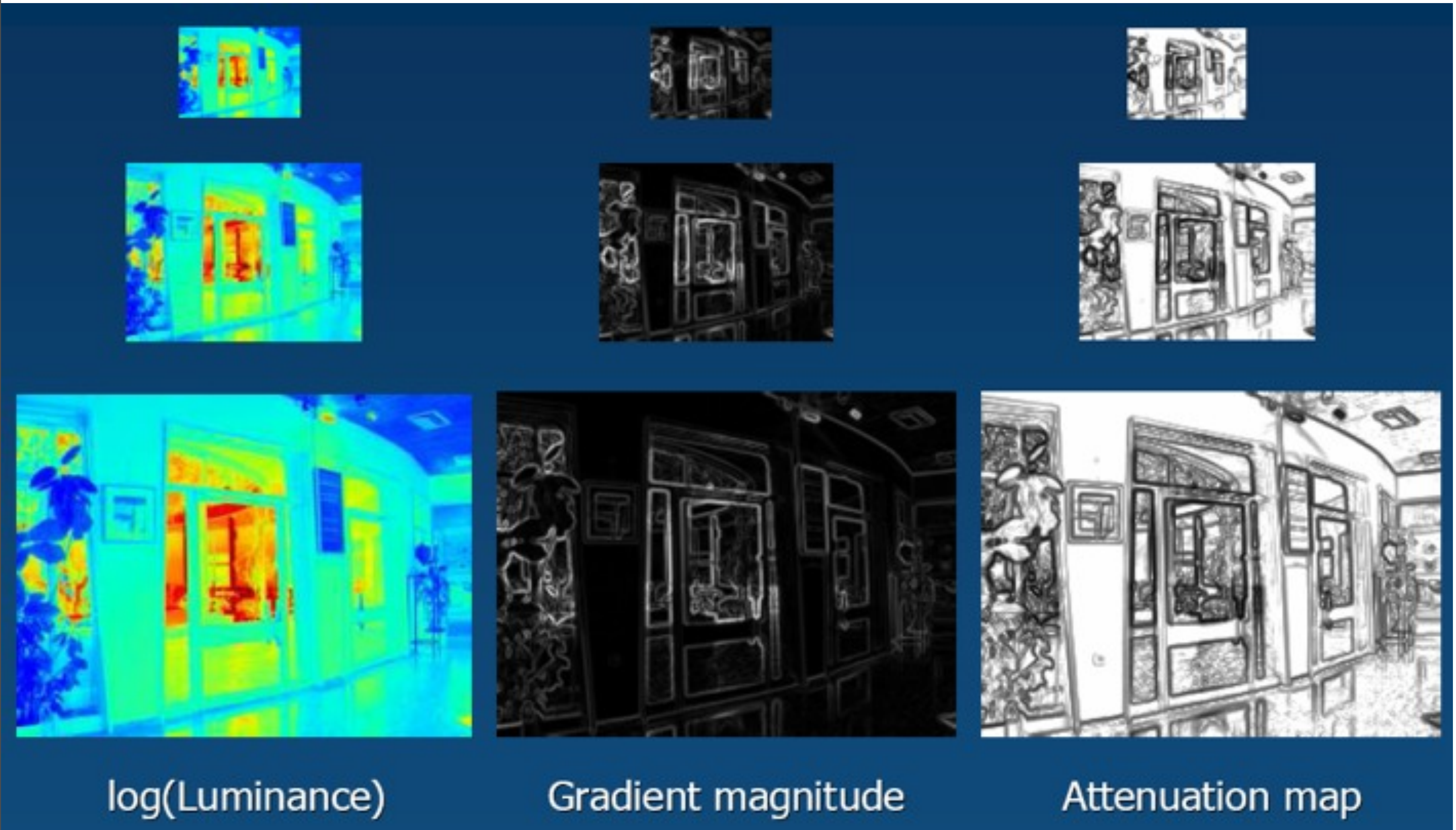
# Gradient tone mapping

- **Fattal et al. Siggraph 2002**




Slide from Siggraph 2005 by Raskar (Graphs by Fattal et al.)

# Gradient attenuation

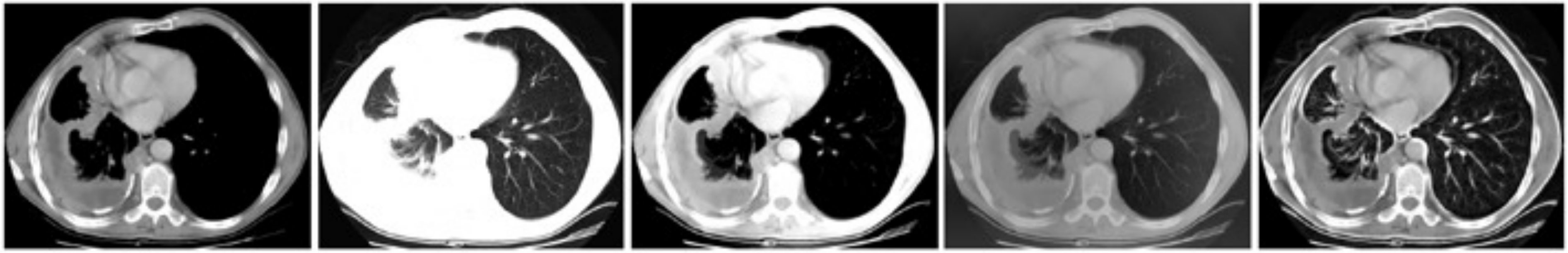



From Fattal et al.

# Fattal et al. Gradient tone mapping

# Gradient tone mapping

- **Socolinsky, D.** ***Dynamic Range Constraints in Image Fusion and Visualization*** **, in Proceedings of Signal and Image Processing 2000, Las Vegas, November 2000.**



(a) (b) (c) (d) (e)

Fig. 1. (a) Mediastinal window of thoracic CT scan. (b) Lung window of thoracic CT scan. (c) Clipped solution of equation (2) for the fusion of (a) and (b). (d) Linearly scaled solution of (2) for the fusion of (a) and (b). (e) Solution of equation (6) for the fusion of(a) and (b).

# Gradient tone mapping

- Socolinsky, D. *Dynamic Range Constraints in Image Fusion and Visualization* , in Proceedings of Signal and Image Processing 2000.

Fig. 4. Left: average of images in figure 2. Middle: rendering of the sum of the images in figure 2 through adaptive histogram compression. Right: fusion of images in figure 2 using the obstacle method.

- **Socolinsky, D. and Wolff, L.B., *A new paradigm for multispectral image visualization and data fusion*, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Fort Collins, June 1999.**

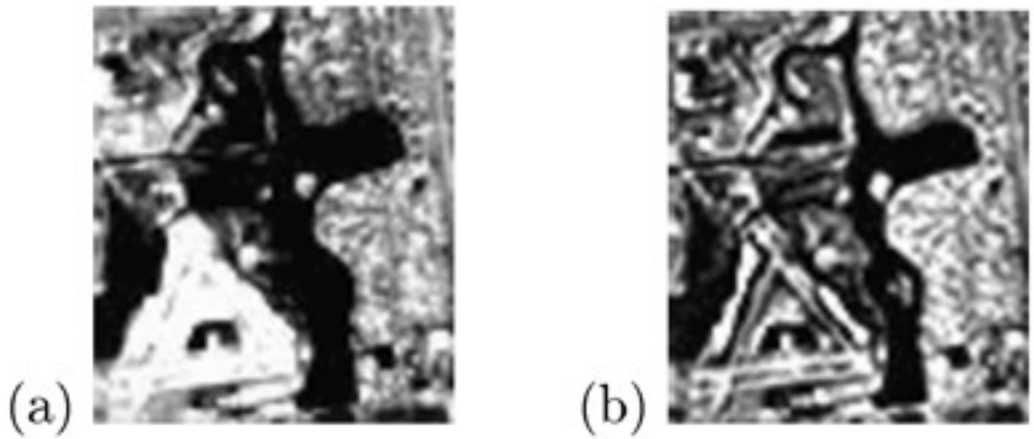
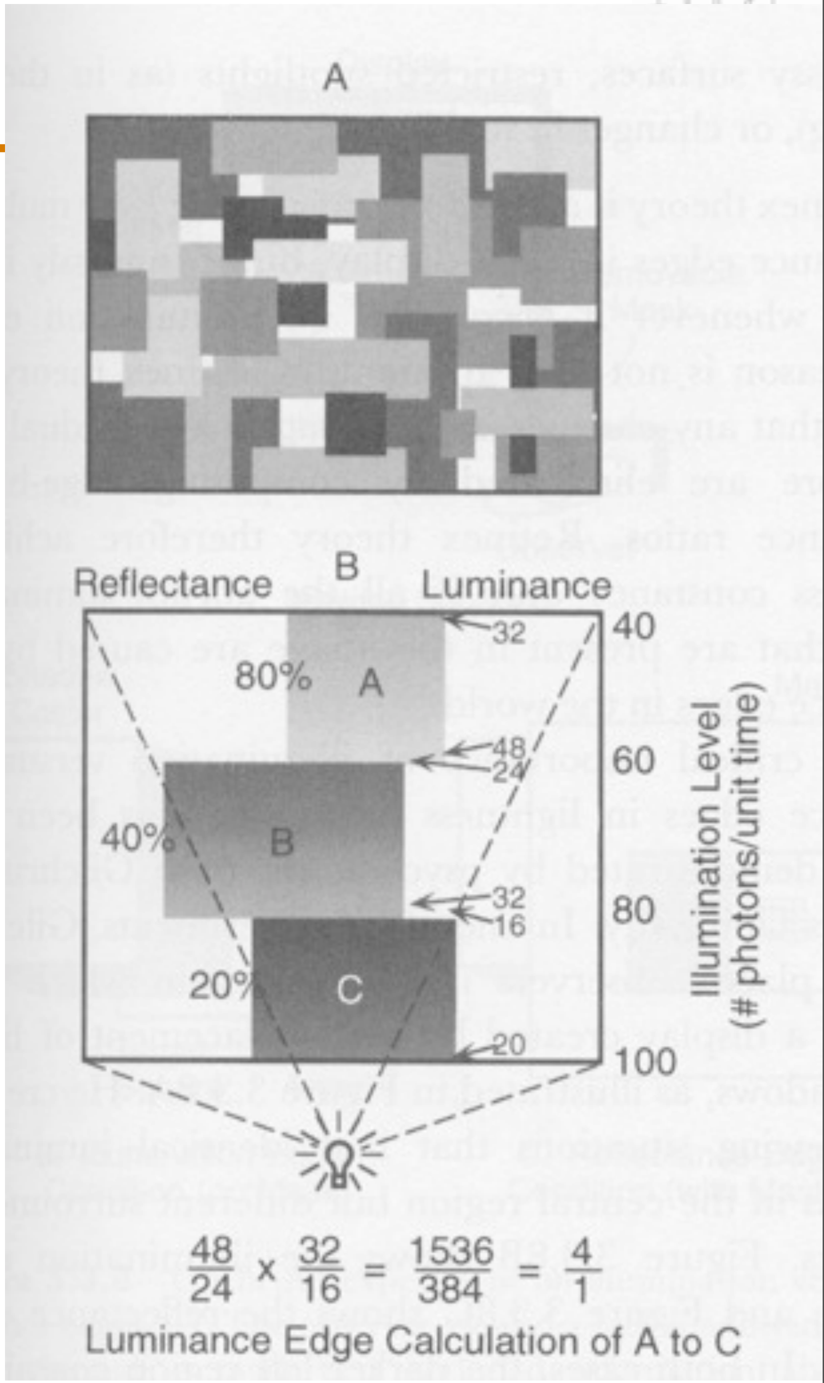



Figure 4: (a) Grayscale version of 9-band image computed through PCA. (b) Grayscale version of the same image computed through our algorithm.

# Retinex

- **Land, Land and McCann (inventor/founder of polaroid)**
- **Theory of lightness perception (albedo vs. illumination)**
- **Strong gradients come from albedo, illumination is smooth**



# Questions?

# Color2gray

- **Use Lab gradient to create grayscale images**

## Color2Gray: Salience-Preserving Color Removal

Amy A. Gooch    Sven C. Olsen    Jack Tumblin    Bruce Gooch

Northwestern University *

Figure 1: A color image (Left) often reveals important visual details missing from a luminance-only image (Middle). Our Color2Gray algorithm (Right) maps visible color changes to grayscale changes. *Image: Impressionist Sunrise by Claude Monet, courtesy of Artcyclopedia.com.*

# Gradient camera?

- **Tumblin et al. CVPR 2005** [**http://www.cfar.umd.edu/~aagrawal/gradcam/gradcam.html**](http://www.cfar.umd.edu/~aagrawal/gradcam/gradcam.html)

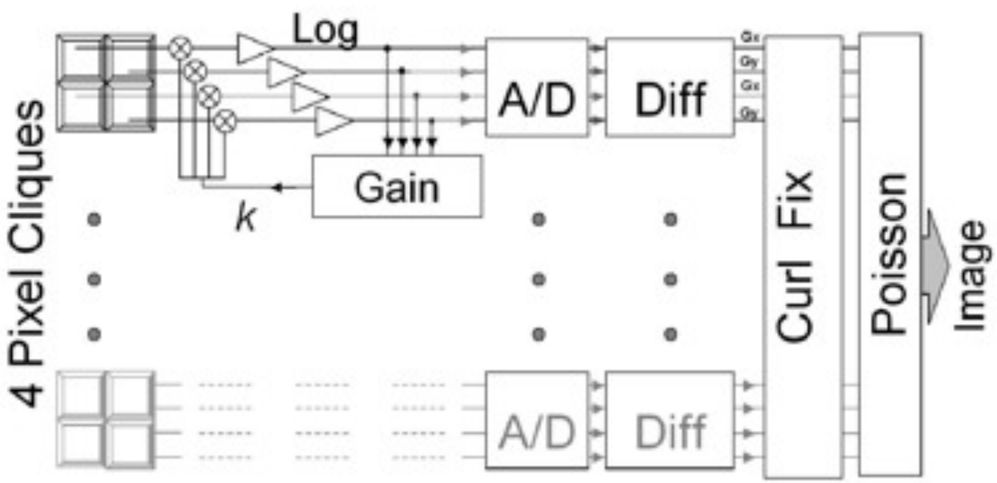



Figure 2. Log-gradient camera overview: intensity sensors organized into 4-pixel cliques share the same self-adjusting gain setting $k$, and send $\log(I_d)$ signals to A/D converter. Subtraction removes common-mode noise, and a linear 'curl fix' solver corrects saturated gradient values or 'dead' pixels, and a Poisson solver finds output values from gradients.

# Poisson-ish mesh editing

- http://portal.acm.org/citation.cfm?id=1057432.1057456
- http://www.cad.zju.edu.cn/home/xudong/Projects/mesh_editing/main.htm
- http://people.csail.mit.edu/sumner/research/deftransfer/

Figure 1: An unknown mythical creature. Left: mesh components for merging and deformation (the arm), Right: final editing result.

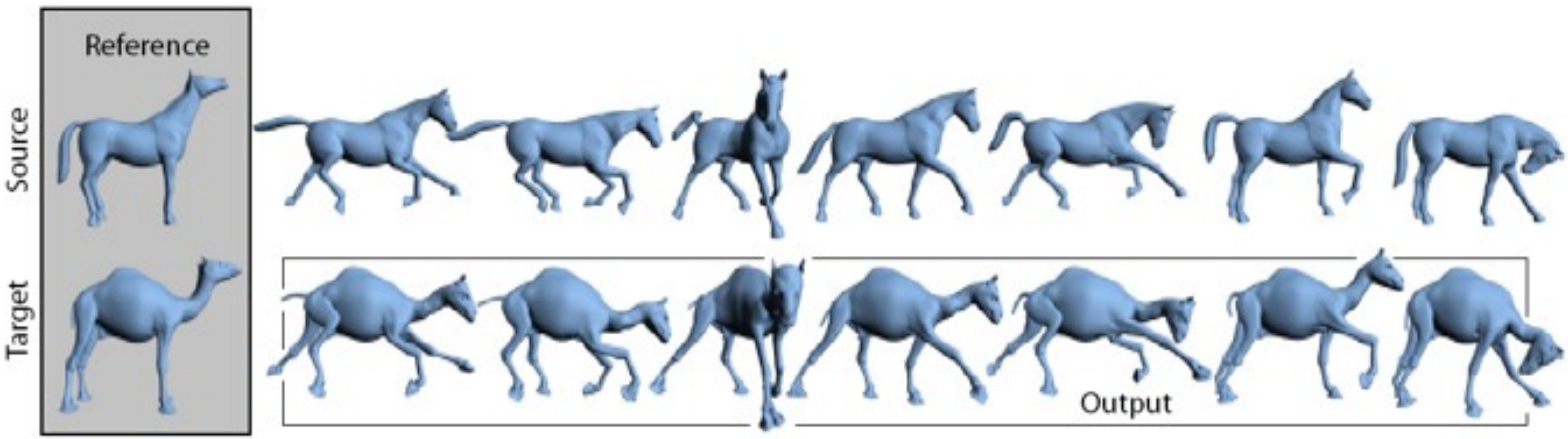



Figure 1: Deformation transfer copies the deformations exhibited by a source mesh onto a different target mesh. In this example, deformations of the reference horse mesh are transfered to the reference camel, generating seven new camel poses. Both gross skeletal changes as well as more subtle skin deformations are successfully reproduced.

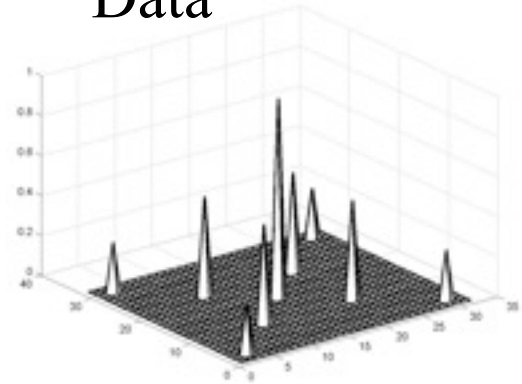
# Questions?

# Alternative to membrane

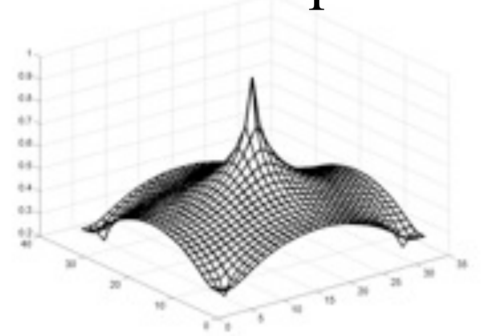
- **Thin plate:**
  **minimize *second* derivative**

$$\min_f \int\int f_{xx}^2 + 2f_{xy}^2 + f_{yy}^2 \,\mathrm{d}x\mathrm{d}y$$

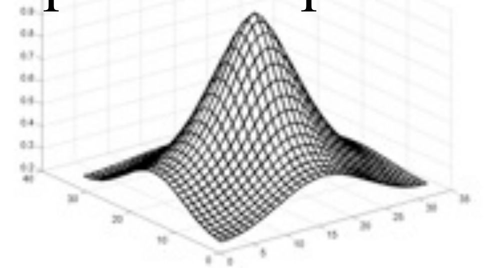
Data



Membrane interpolation



Thin-plate interpolation



# Inpainting

- **More elaborate energy functional/PDEs**
- **http://www-mount.ee.umn.edu/~guille/inpainting.htm**


# Key references

- **Socolinsky, D.** ***Dynamic Range Constraints in Image Fusion and Visualization*** **2000. http://www.equinoxsensors.com/news.html**
- **Elder, Image editing in the contour domain, 2001 http://elderlab.yorku.ca/~elder/publications/journals/ElderPAMI01.pdf**
- **Fattal et al. 2002**
  **Gradient Domain HDR Compression http://www.cs.huji.ac.il/%7Edanix/hdr/**
- **Poisson Image Editing  Perez et al. http://research.microsoft.com/vision/cambridge/papers/perez_siggraph03.pdf**
- **Covariant Derivatives and Vision,  Todor Georgiev (Adobe Systems)  ECCV 2006**

## Poisson, Laplace, Lagrange, Fourier, Monge, Parseval

- **Fourier studied under Lagrange, Laplace & Monge, and Legendre & Poisson were around**
- **They all raised serious objections about Fourier's work on Trigomometric series**
- http://www.ece.umd.edu/~taylor/frame2.htm
- http://www.mathphysics.com/pde/history.html
- http://www-groups.dcs.st-and.ac.uk/~history/Mathematicians/Fourier.html
- http://www.memagazine.org/contents/current/webonly/wex80905.html
- http://www.shsu.edu/~icc_cmf/bio/fourier.html
- http://en.wikipedia.org/wiki/Simeon_Poisson
- http://en.wikipedia.org/wiki/Pierre-Simon_Laplace
- http://en.wikipedia.org/wiki/Jean_Baptiste_Joseph_Fourier
- http://www-groups.dcs.st-and.ac.uk/~history/Mathematicians/Parseval.html

# Refs Laplace and Poisson

- http://www.ifm.liu.se/~boser/elma/Lect4.pdf
- http://farside.ph.utexas.edu/teaching/329/lectures/node74.html
- http://en.wikipedia.org/wiki/Poisson's_equation
- http://www.colorado.edu/engineering/CAS/courses.d/AFEM.d/AFEM.Ch03.d/AFEM.Ch03.pdf

# Gradient image editing refs

- http://research.microsoft.com/vision/cambridge/papers/perez_siggraph03.pdf
- http://www.cs.huji.ac.il/~alevin/papers/eccv04-blending.pdf
- http://www.eg.org/EG/DL/WS/COMPAESTH/COMPAESTH05/075-081.pdf.abstract.pdf
- http://photo.csail.mit.edu/posters/Georgiev_Covariant.pdf
- **Covariant Derivatives and Vision, Todor Georgiev (Adobe Systems) ECCV 2006**
- http://www.mpi-sb.mpg.de/~hitoshi/research/image_restoration/index.shtml
- http://www.cs.tau.ac.il/~tommer/vidoegrad/
- http://ieeexplore.ieee.org/search/wrapper.jsp?arnumber=1467600
- http://grail.cs.washington.edu/projects/photomontage/
- http://www.cfar.umd.edu/~aagrawal/iccv05/surface_reconstruction.html
- http://www.merl.com/people/raskar/Flash05/
- http://research.microsoft.com/~carrot/new_page_1.htm
- http://www.idiom.com/~zilla/Work/scatteredInterpolation.pdf

# Poisson image editing

- **Two aspects**
  - When the new gradient is conservative: Just membrane interpolation to ensure boundary condition
  - Otherwise: allows you to work with non-conservative vector fields and
- **Why is it good?**
  - More weight on high frequencies
    - Membrane tries to use low frequencies to match boundaries conditions
  - Manipulation of the gradient can be cool (e.g. max of the two gradients)
    - Manipulate local features (edge/gradient) and worry about global consistency later
- **Smart thing to do: work in log domain**
- **Limitations**
  - Color shift, contrast shift (depends strongly on the difference between the two respective backgrounds)