Gradient image processing

Announcement

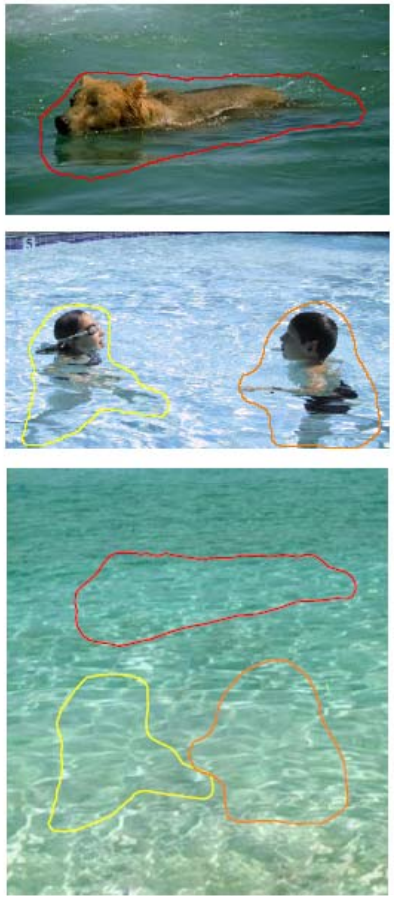
- Assignment 1 is out
- 3 programming questions:
  - Poisson Image Editing (Topic in this class)
  - Closed Form Matting (Topic in next class)
  - Lazy Snapping (Topic in Interactive Image Seg. and MRF)
- Today's topics:
  - Poisson Image Editing, Siggraph'03
  - Interactive Digital Photomontage, Siggraph'04
  - Drag and Drop Pasting, Siggraph'06
- Extra reading materials:
  - Real-Time Gradient-Domain Painting, Siggraph'08
  - Moving Gradients: A Path-Based Method for Plausible Image Interpolation, Siggraph'09

Gradient manipulation

Idea:

- Human visual system is very sensitive to gradient, e.g. edges, discontinuities
- Gradient encode edges and local contrast quite well
- Do your editing in the gradient domain
- Reconstruct image from gradient

Problems with direct cloning



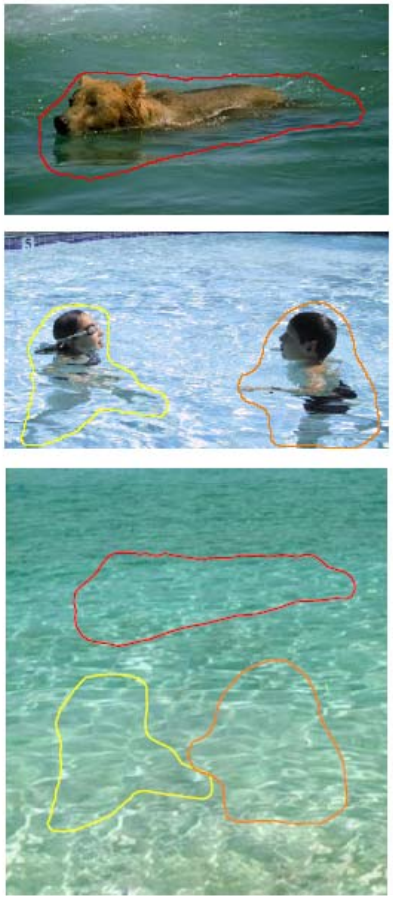
sources/destinations

cloning

Discontinuities introduced at region boundaries.

From Perez et al. 2003

Solution: clone gradient



sources/destinations

seamless cloning

Instead of copying pixel intensity, we copied image gradients

Gradients and grayscale images

- Grayscale image: N x N scalars
- Gradient: dx, dy values for each pixel
- We store 2 values for gradient, but we store 1 values for intensity
- This is Over-complete information!
- What's up with this?
  - We can reconstruct images from gradients

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 boundary conditions

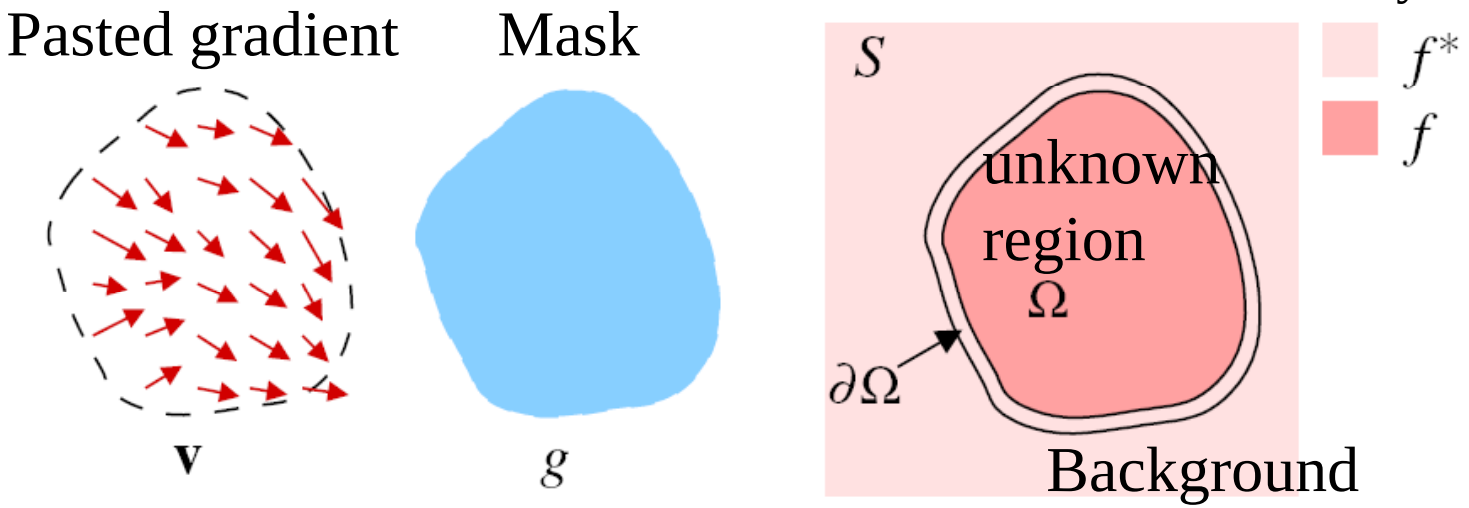


Figure 1: **Guided interpolation notations**. Unknown function f interpolates in domain Ω the destination function f^*, under guidance of vector field $\mathbf{v}$, which might be or not the gradient field of a source function g.

Poisson Image Editing

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

Gradient of *f*, should be as close as possible to **v**.

Boundaries of *f* and *f** should match. *f** are from the original image.

This is the Poisson constraint:

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

2nd Derivate of ***f***

Divergence of "***v***" (derivate of ***v***)

with boundary constraint.

Practical Implementation

- There are many way to solve this equation
- In this class, I will teach you the method to solve Poisson equation by solving a least square linear equation system:

$$Ax=b$$

- You will learn that later, many problems can be transformed into a problem of solving Ax = b
- This is a useful technique

Discrete Poisson solver

- Minimization problem

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

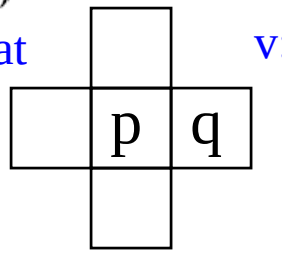
Discretized gradient

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

(all pairs that are in Ω)

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

Boundary condition



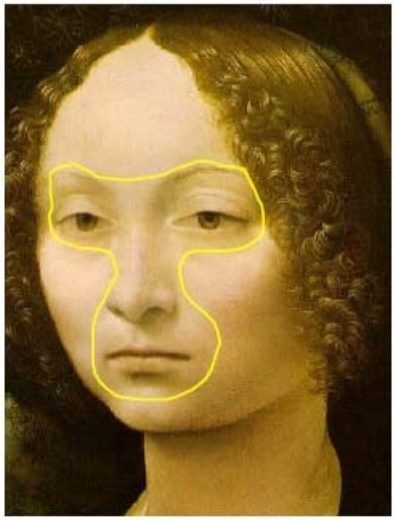
- Rearrange and call N_p the neighbors of p

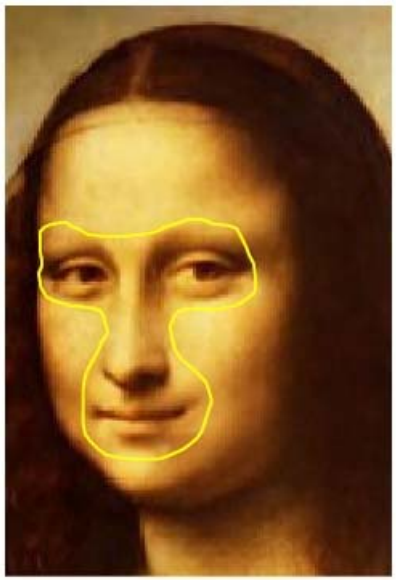
$$\text{for all } p \in \Omega, \quad |N_p| f_p - \sum_{q \in N_p \cap \Omega} f_q = \sum_{q \in N_p \cap \partial\Omega} f_q^* + \sum_{q \in N_p} v_{pq}$$

Only for boundary pixels

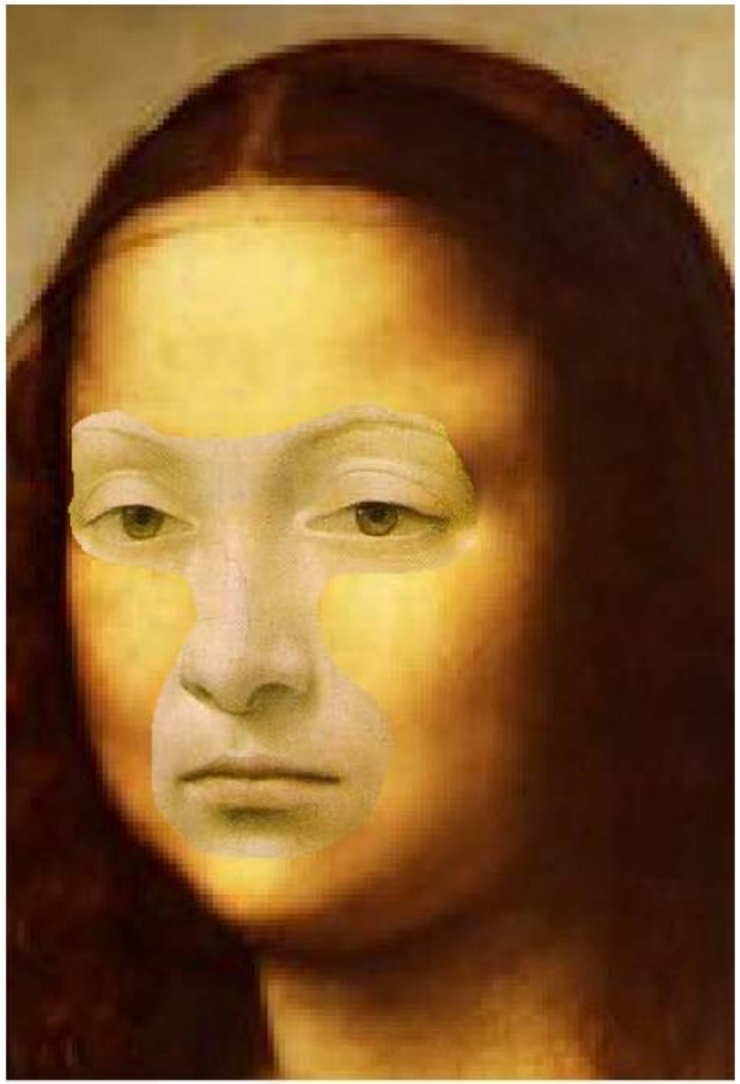
- Big yet sparse linear system
- Reminder, f is solutions

Result (eye candy), Questions ?

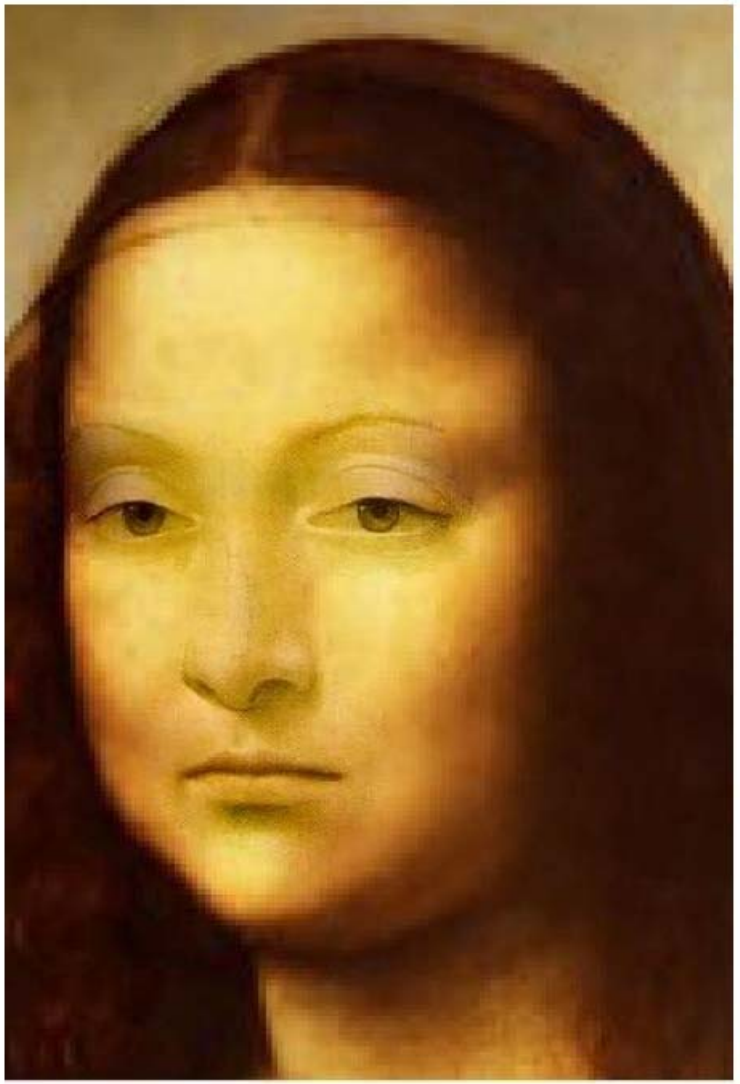




source/destination



cloning



seamless cloning

Solving big matrix systems

- Ax=b
- You can use Matlab's \
  - But not very scalable
- There is also sparse matrix library in C\C++, e.g. TAUCS, that provides routine for solving this sparse linear system
- Good News !
  You can use existing library to avoid the ``trouble'' implementation of linear equation solver
- But, you need to understand what is happening within the linear solver

Conjugate gradient

An Introduction to
the Conjugate Gradient Method
Without the Agonizing Pain
Edition $1\frac{1}{4}$

Jonathan Richard Shewchuk
August 4, 1994

- "The Conjugate Gradient Method is the most prominent iterative method for solving sparse systems of linear equations. Unfortunately, many textbook treatments of the topic are written with neither illustrations nor intuition, and their victims can be found to this day babbling senselessly in the corners of dusty libraries. For this reason, a deep, geometric understanding of the method has been reserved for the elite brilliant few who have painstakingly decoded the mumblings of their forebears. Nevertheless, the Conjugate Gradient Method is a composite of simple, elegant ideas that almost anyone can understand. Of course, a reader as intelligent as yourself will learn them almost effortlessly."

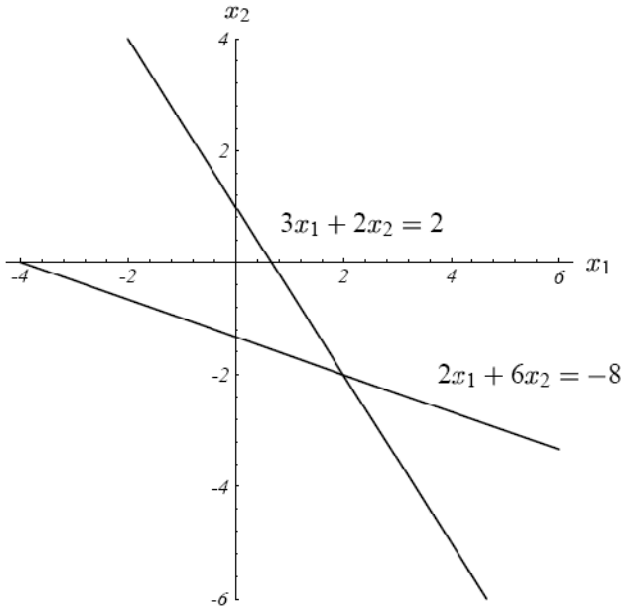
Ax=b

- A is square, symmetric and positive-definite
- When the A is dense, you're stuck, use backsubstitution
- When A is sparse, iterative techniques (such as Conjugate Gradient) are faster and more memory efficient

- Simple example:

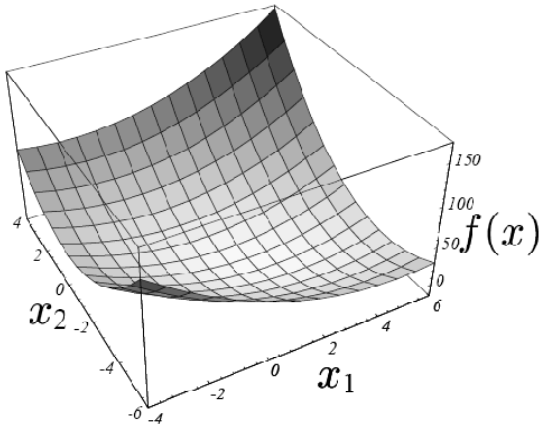
$$\begin{bmatrix} 3 & 2 \\ 2 & 6 \end{bmatrix} x = \begin{bmatrix} 2 \\ -8 \end{bmatrix}$$

(Yeah yeah, it's not sparse)

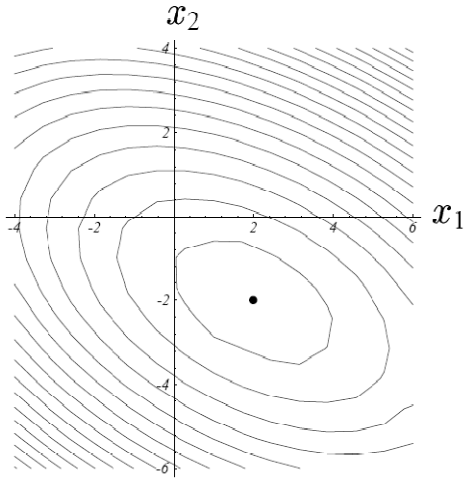


Turn Ax=b into a minimization problem

- Minimization is more logical to analyze iteration (gradient ascent/descent)
- Quadratic form $f(x) = \frac{1}{2}x^T Ax - b^T x + c$
  - c can be ignored because we want to minimize
- Intuition:
  - the solution of a linear system is always the intersection of n hyperplanes
  - Take the square distance to them
  - A needs to be positive-definite so that we have a nice parabola



Graph of quadratic form $f(x) = \frac{1}{2}x^T Ax - b^T x + c$. The minimum point of this surface is the solution to $Ax = b$.



Contours of the quadratic form. Each ellipsoidal curve has constant $f(x)$.

Gradient of the quadratic form

$$f'(x) = \begin{bmatrix} \frac{\partial}{\partial x_1} f(x) \\ \frac{\partial}{\partial x_2} f(x) \\ \vdots \\ \frac{\partial}{\partial x_n} f(x) \end{bmatrix}$$

- Not our image gradient!
- Multidimensional gradient (as many dim as rows in matrix)

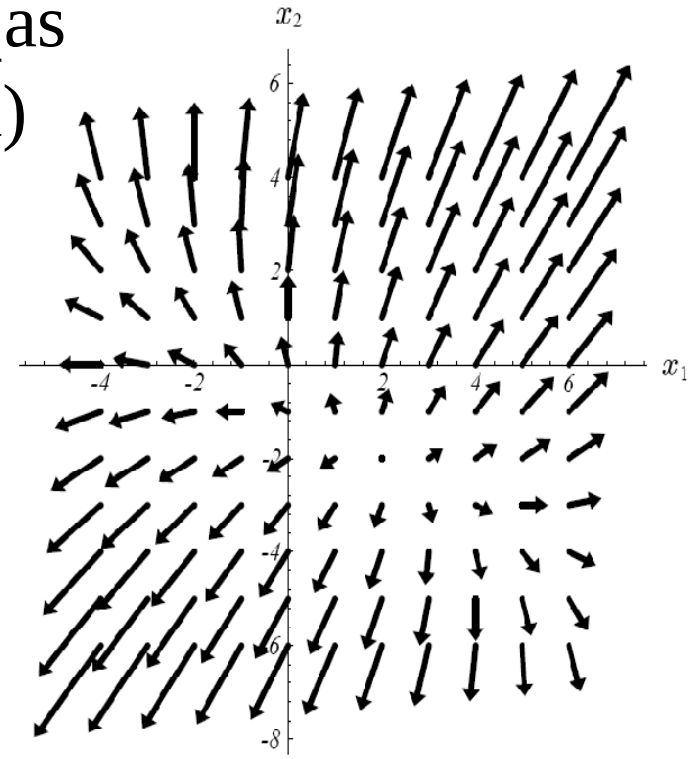
since $$f(x) = \frac{1}{2} x^T A x - b^T x + c$$

$$f'(x) = \frac{1}{2} A^T x + \frac{1}{2} A x - b.$$

And since A is symmetric

$$f'(x) = Ax - b$$

Not surprising: we turned Ax=b into the quadratic minimization

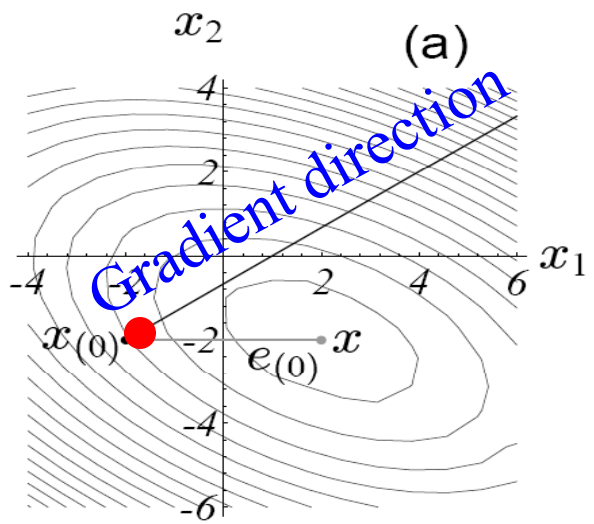


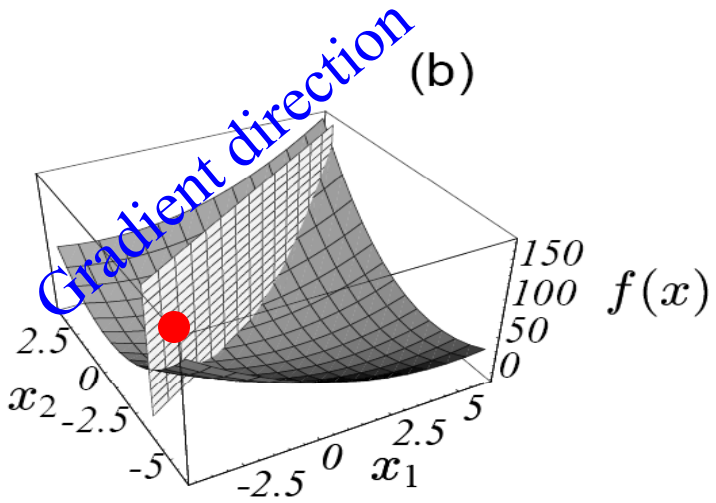
Gradient $f'(x)$ of the quadratic form. For every x, the gradient points in the direction of steepest increase of $f(x)$, and is orthogonal to the contour lines.

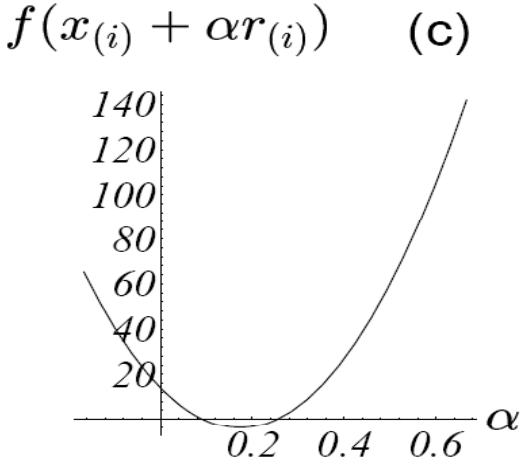
(if A is not symmetric, conjuagte gradient finds solution for $\frac{1}{2}(A^T + A)x = b.$

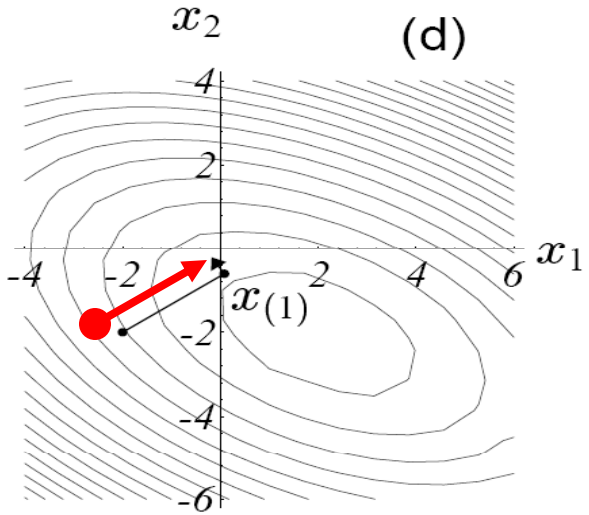
Steepest descent/ascent

- Pick gradient direction
- Find optimum in this direction









Energy along the gradient

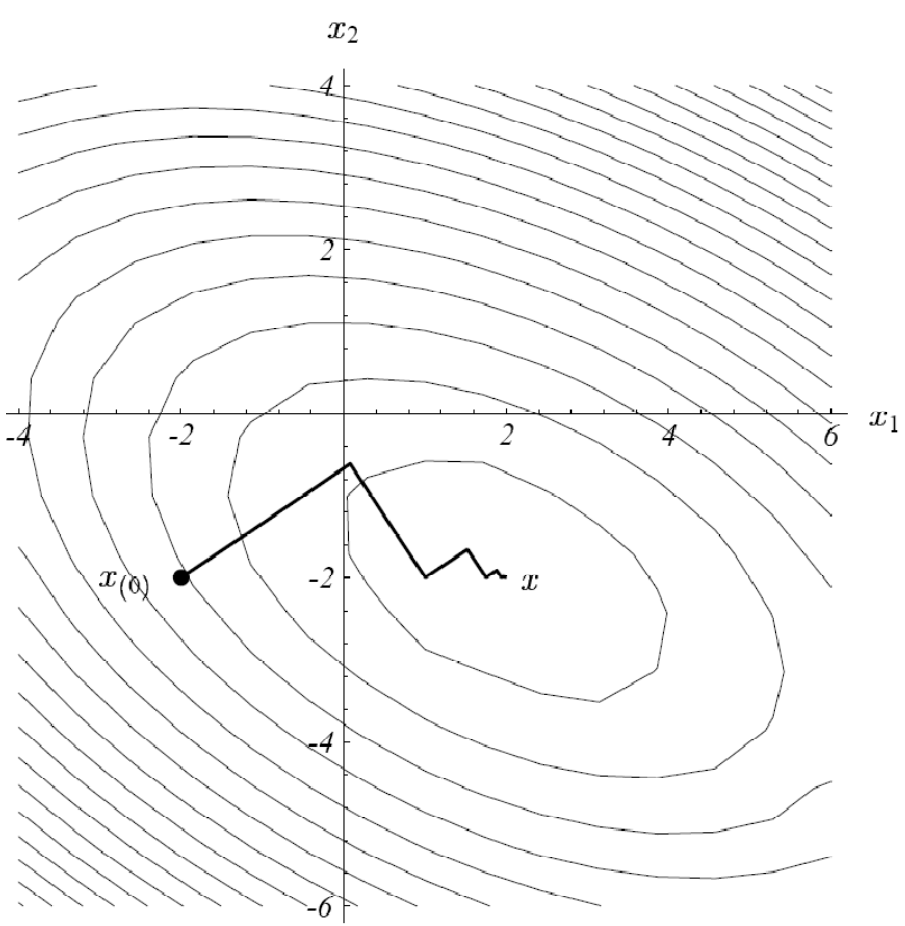
The method of Steepest Descent.

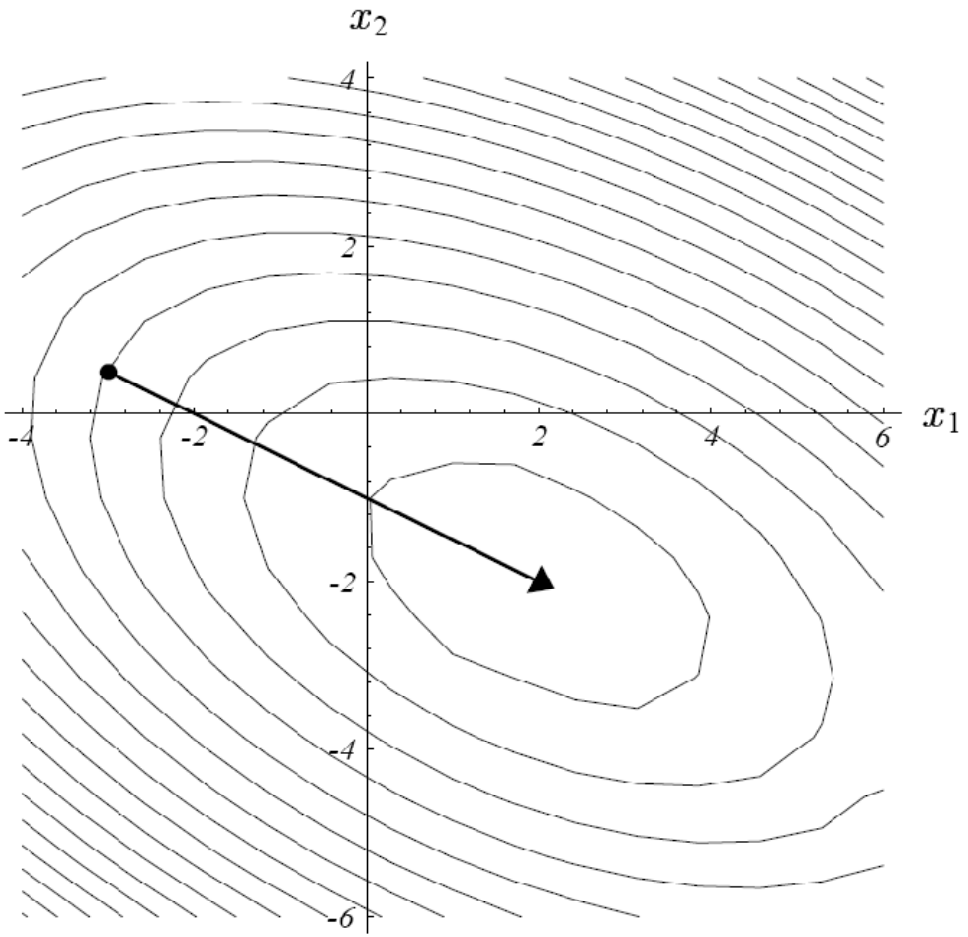
Residual

- At iteration i, we are at a point x(i)
- Residual r(i)=b-Ax(i)
- Cool property of quadratic form:
  residual = - gradient

Behavior of gradient descent

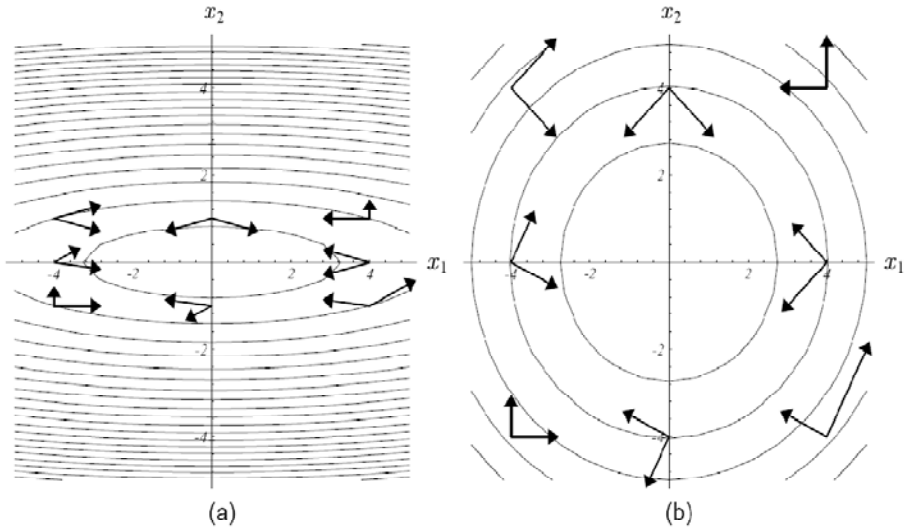
- Zigzag or goes straight depending if we're lucky
  - Ends up doing multiple steps in the same direction





Conjugate gradient

- Smarter choice of direction
  - Ideally, step directions should be orthogonal to one another (no redundancy)
  - But tough to achieve
  - Next best thing: make them A-orthogonal $d_{(i)}^T A d_{(j)} = 0$



transformed by A:

Figure 22: These pairs of vectors are A-orthogonal . . . because these pairs of vectors are orthogonal.

Conjugate gradient

- For each step:
  - Take the residual (gradient)
  - Make it A-orthogonal to the previous ones
  - Find minimum along this direction
- Plus life is good:
  - In practice, you only need the previous one
  - You can show that the new residual r(i+1) is already A-orthogonal to all previous directions p but p(i)

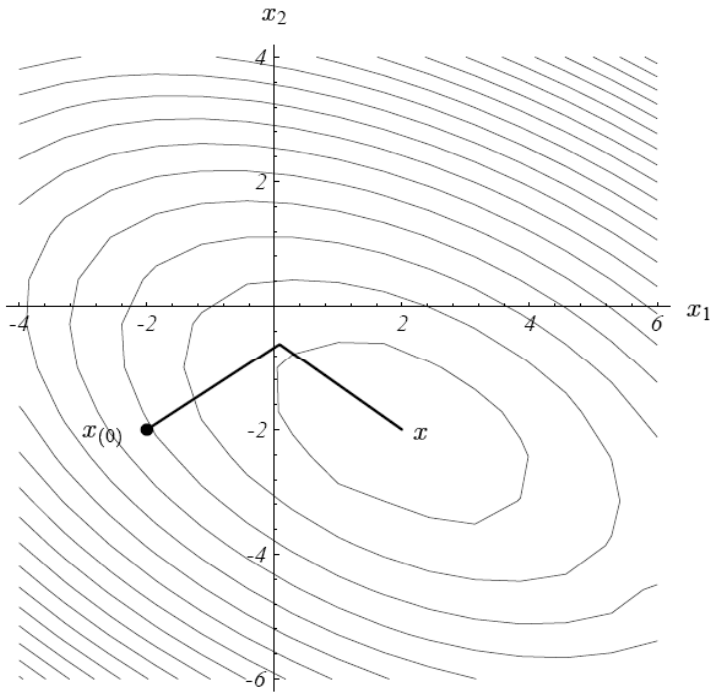


Figure 30: The method of Conjugate Gradients.

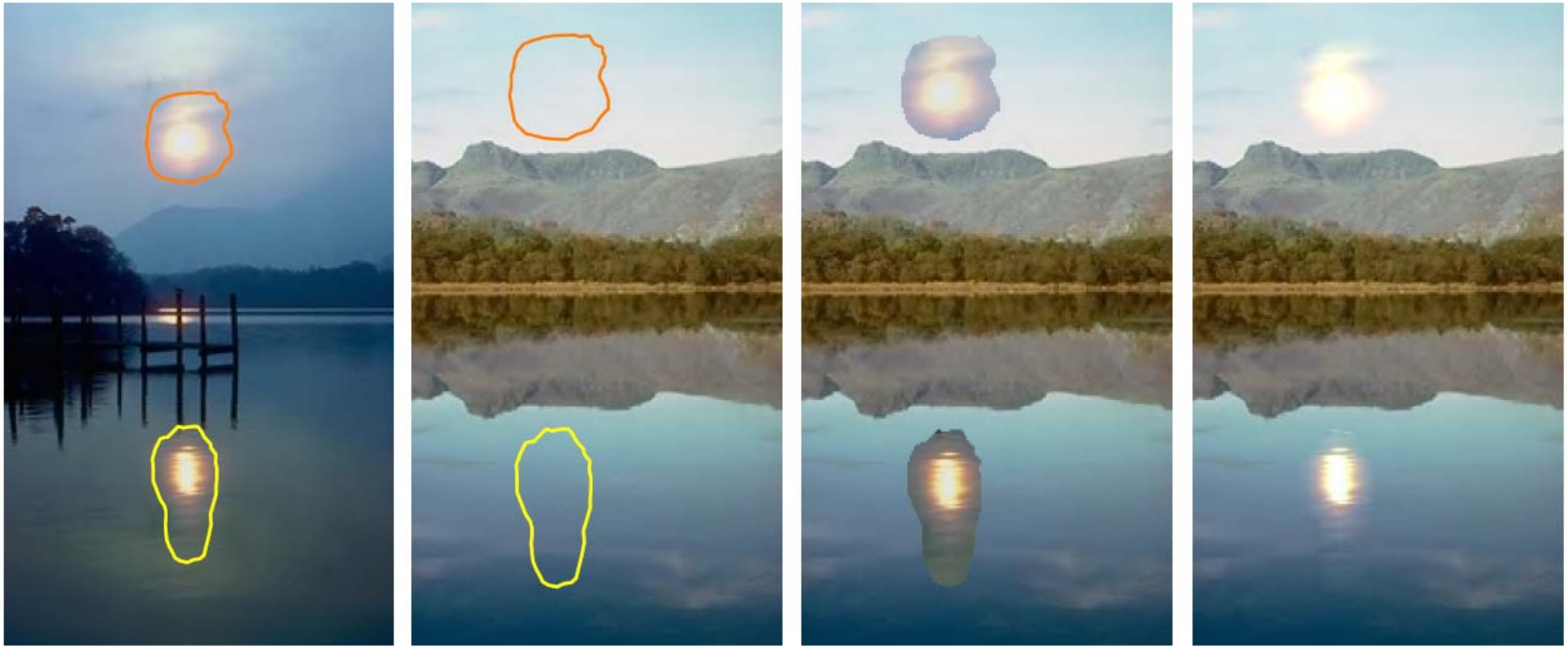
Recap

- Poisson image cloning: paste gradient, enforce boundary condition

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

- Discretize version, leads to big but sparse linear system, Ax = b
- Conjugate gradient is a smart iterative technique to solve it
- Questions ?

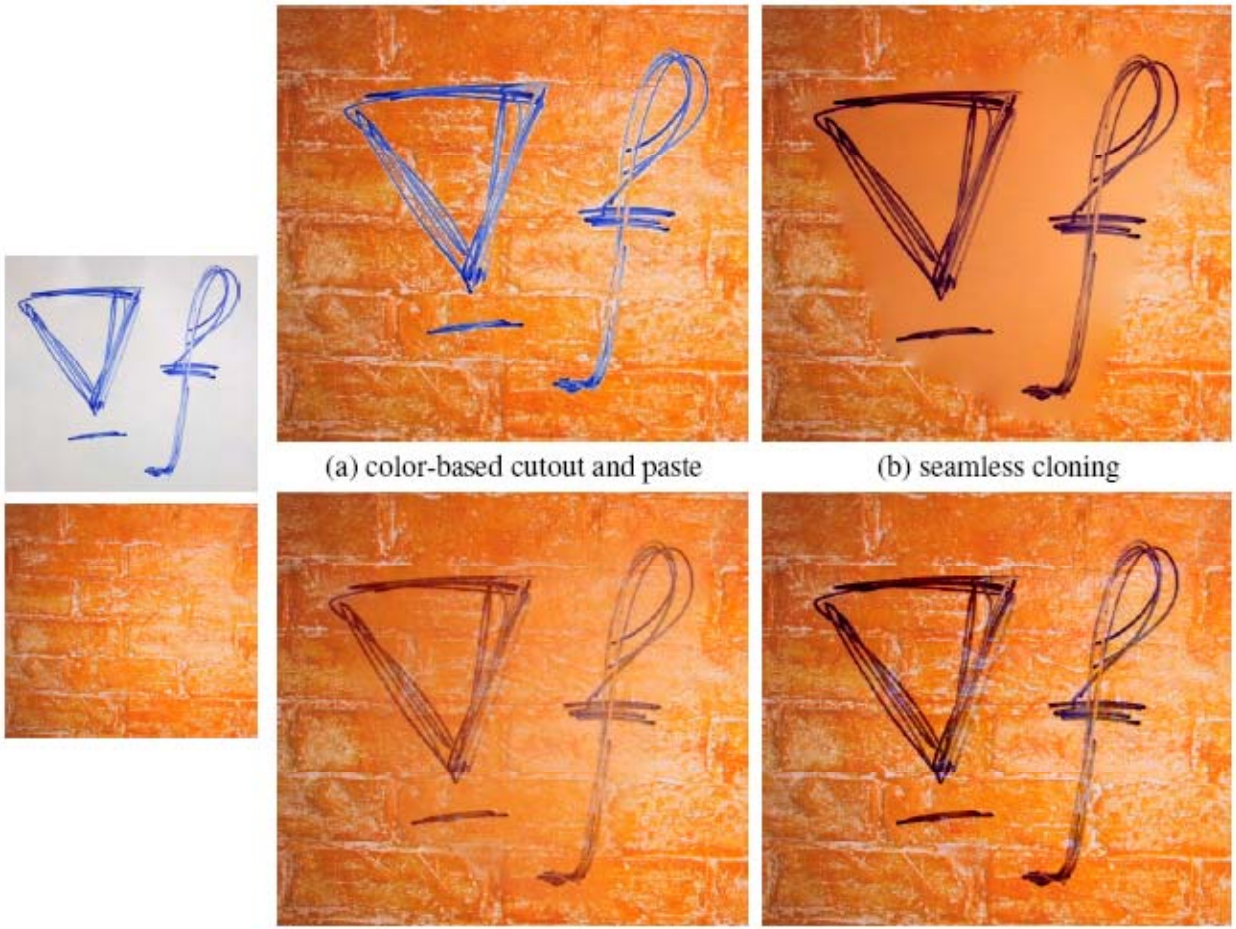
Figure 2: **Concealment**. By importing seamlessly a piece of the background, complete objects, parts of objects, and undesirable artifacts can easily be hidden. In both examples, multiple strokes (not shown) were used.



sources destinations cloning seamless cloning

Manipulate the gradient

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

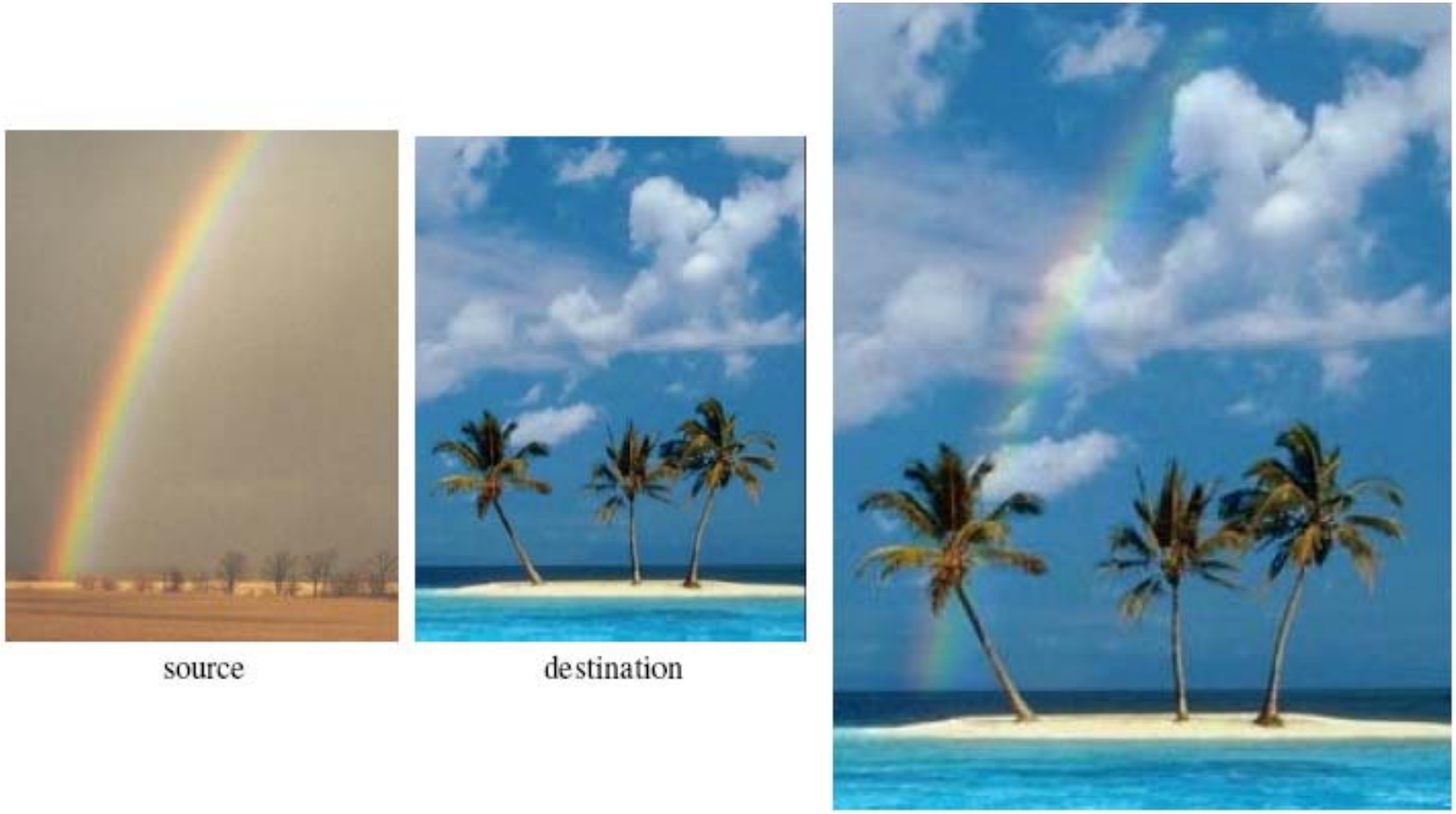


Figure 7: **Inserting transparent objects**. Mixed seamless cloning facilitates the transfer of partly transparent objects, such as the rainbow in this example. The non-linear mixing of gradient fields picks out whichever of source or destination structure is the more salient at each location.

Issues with Poisson cloning

- The backgrounds in f & g should be similar.
- The numerical values of linear solution does not necessary fall within intensity range.
- If boundary condition is not provided, the solution has freedom to shift in intensity level

Interactive Digital Photomontage

- http://grail.cs.washington.edu/projects/photo

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.

Idea

- Use interactive image segmentation together with gradient blending in image stitching
- Stitch multiple photos of same scene / similar object

Input

- Multiple photos of same scene.

Problems: Not all of them are looking at camera

User Interaction

- Select the part of photos you want to merge
- Is it a very familiar user interface ?

Graph-cut segmentation

- This is again MRF segmentation problem, I hope you still remember

$$C(L) = \underbrace{\sum_{p} C_d(p, L(p))}_{\text{Data term}} + \underbrace{\sum_{p,q} C_i(p, q, L(p), L(q))}_{\text{Neighborhood term}}$$

- For more details, please see the paper. But, the definition of data term and neighborhood term is very similar to the one we learnt in last lesson
- You can derived/defined your own data term and neighborhood term for different problem with different inputs

Results

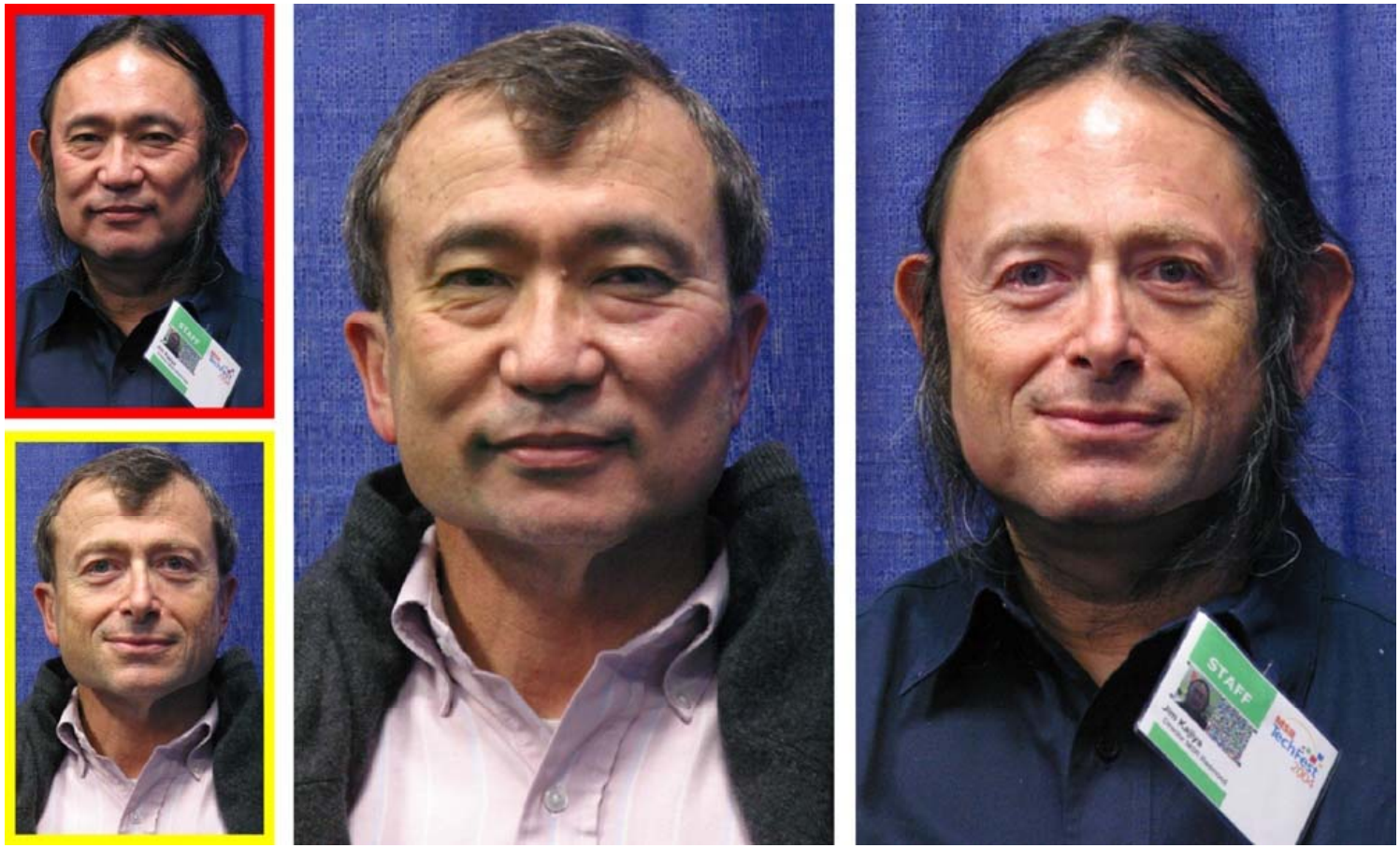
Now they are all looking at camera nicely.

Results

- Video summaries

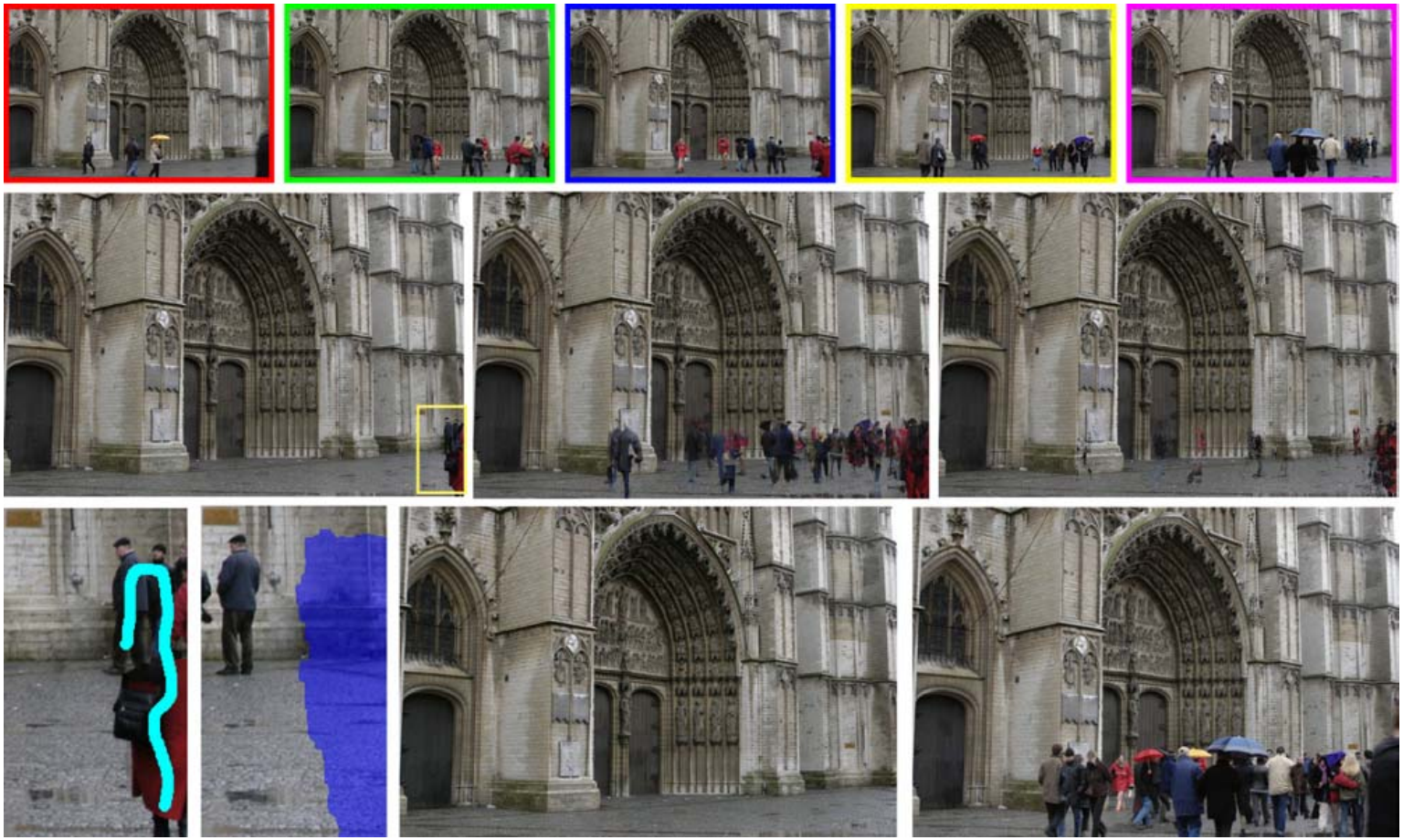
Results

- Create a new face



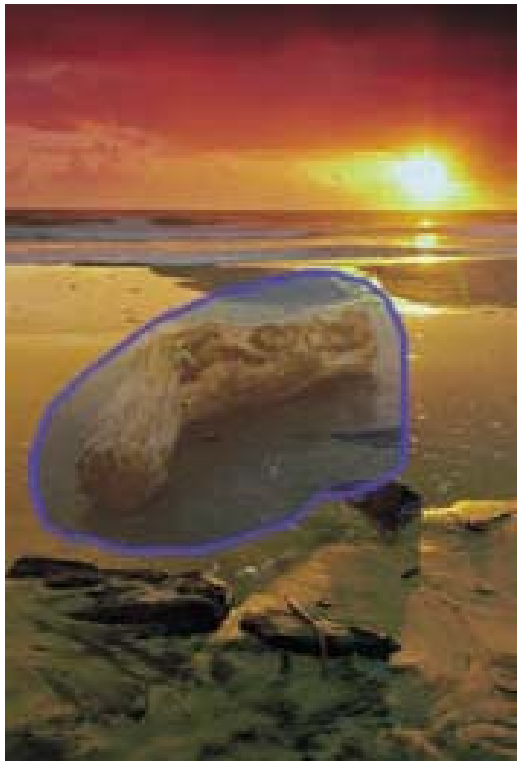
Results

- Taking ``impossible'' photos



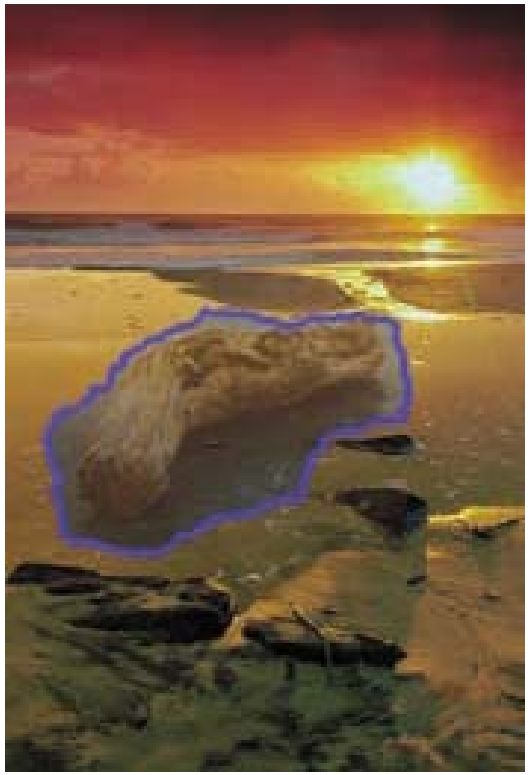
Drag-and-drop pasting

- Let's go back to Poisson Image Editing problem
- Problems:



Drag-and-drop pasting

- How about this ?



It looks good !

Problem of Poisson image editing

- Specifying an ``optimal'' boundary for seamless cloning is not easy
- Users have no idea on what is ``optimal'' boundary
- Is there any a easy way to refine the boundary?
  - Drag-and-drop pasting.
- Idea : find the ``optimal'' boundary for users

Inputs

- Source and target image
- Select a region from source, and then drag-and-drop it to target



Finding optimal boundary

- Shortest path solution, see paper for details

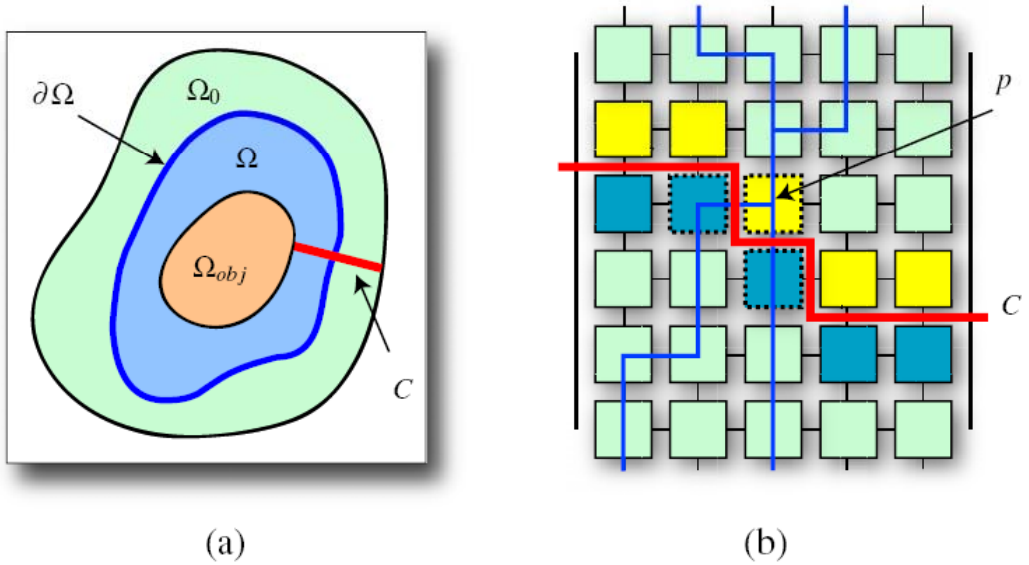


Figure 3: Boundary optimization. (a) The region of interest Ω_0 is pasted onto the target image, which completely encloses the object of interest Ω_{obj}. The optimized boundary $\partial\Omega$ lies inside $\Omega_0 \backslash \Omega_{obj}$. The cut C is shown in red. (b) Zoom-in view of the cut C. The yellow and blue pixels are on different sides of C. The dashed yellow pixel p is adjacent to two blue pixels. Two shortest paths, shown as blue lines, are simultaneously computed.

Notes: This problem can also be again formulated into MRF segmentation problem, the results are similar

Results

Results, Questions?

Other applications: Color2gray

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.*

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)

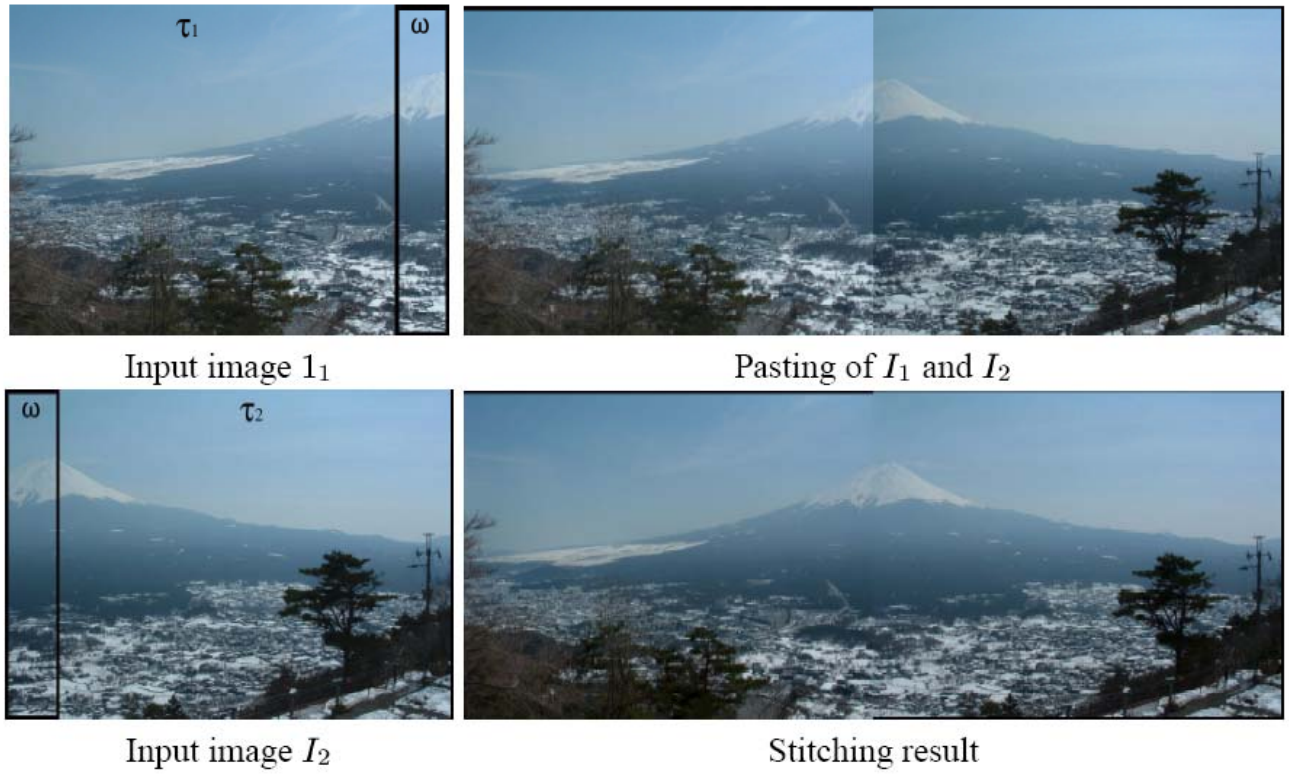


Fig. 1. Image stitching. On the left are the input images. ω is the overlap region. On top right is a simple pasting of the input images. On the bottom right is the result of the GIST1 algorithm.

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.

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/

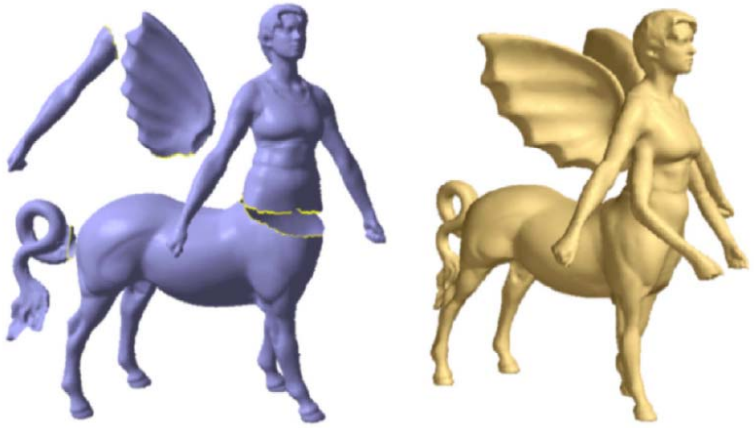


Figure 1: An unknown mythical creature. Left: mesh components for merging and deformation (the arm), Right: final editing result.

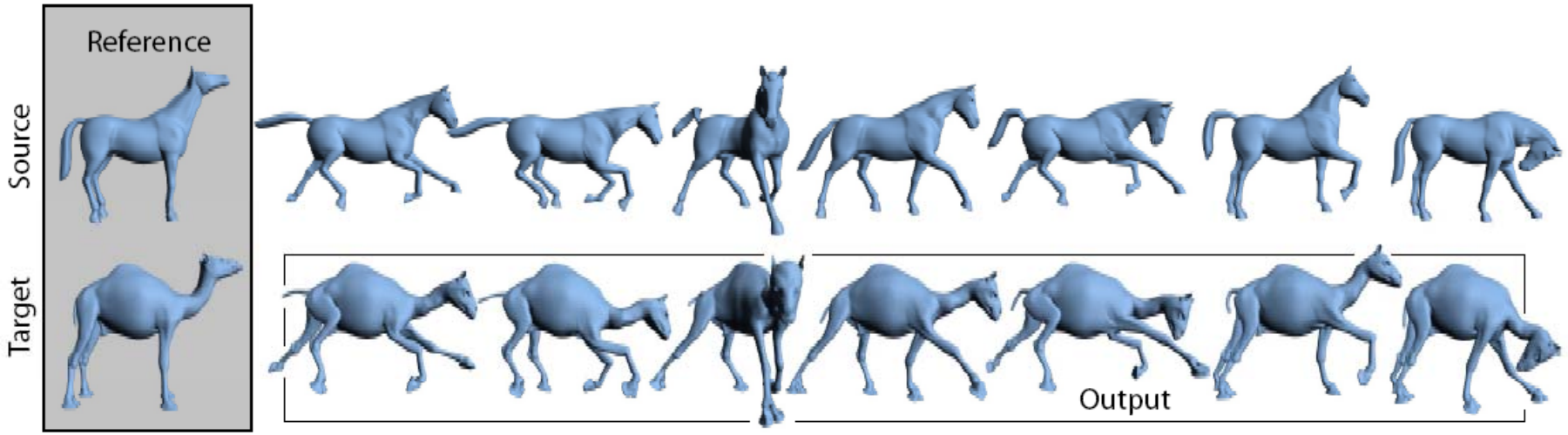


Figure 1: Deformation transfer copies the deformations exhibited by a source mesh onto a different target mesh. In this example, deformations of the reference horse mesh are transfered to the reference camel, generating seven new camel poses. Both gross skeletal changes as well as more subtle skin deformations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

Summary

- We see the power of gradient image processing
- Most later works are based on the Poisson Image Editing, so this is very useful tool
  - You will have your own after the assignment
- Suggested Extra reading:
  - Real-Time Gradient-Domain Painting, Siggraph'08
  - Moving Gradients: A Path-Based Method for Plausible Image Interpolation, Siggraph'09

Importance of good writing

- What is the use of creating the best innovative ideas if nobody else can understand them?

- See Fredo's slides "How to write a bad paper" http://people.csail.mit.edu/fredo/FredoBadWriting.pdf

  useful links http://people.csail.mit.edu/fredo/student.html
- Bill's slides and links:

  http://www.ai.mit.edu/courses/6.899/doneClasses.html

  (Talk on "How to write a conference paper", April 10)