Stitching and Blending

Kari Pulli
VP Computational Imaging
Light

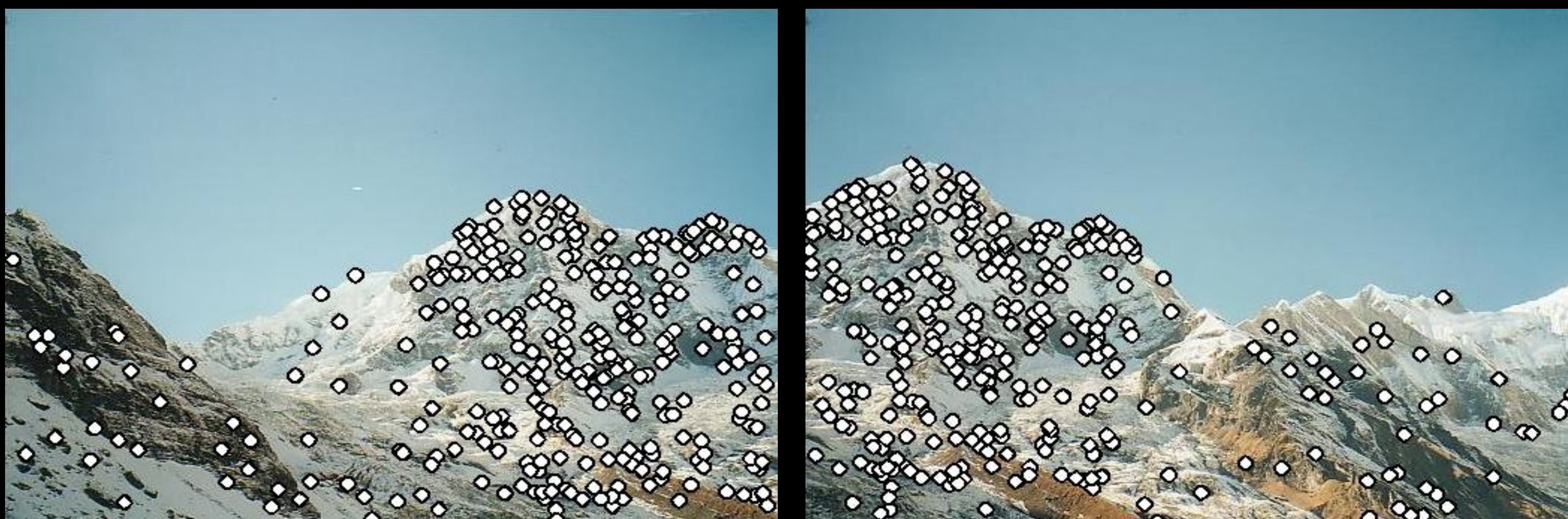
First project

- Build your own (basic) programs
  - panorama
  - HDR (really, exposure fusion)
- The key components
  - register images so their features align
  - determine overlap
  - blend

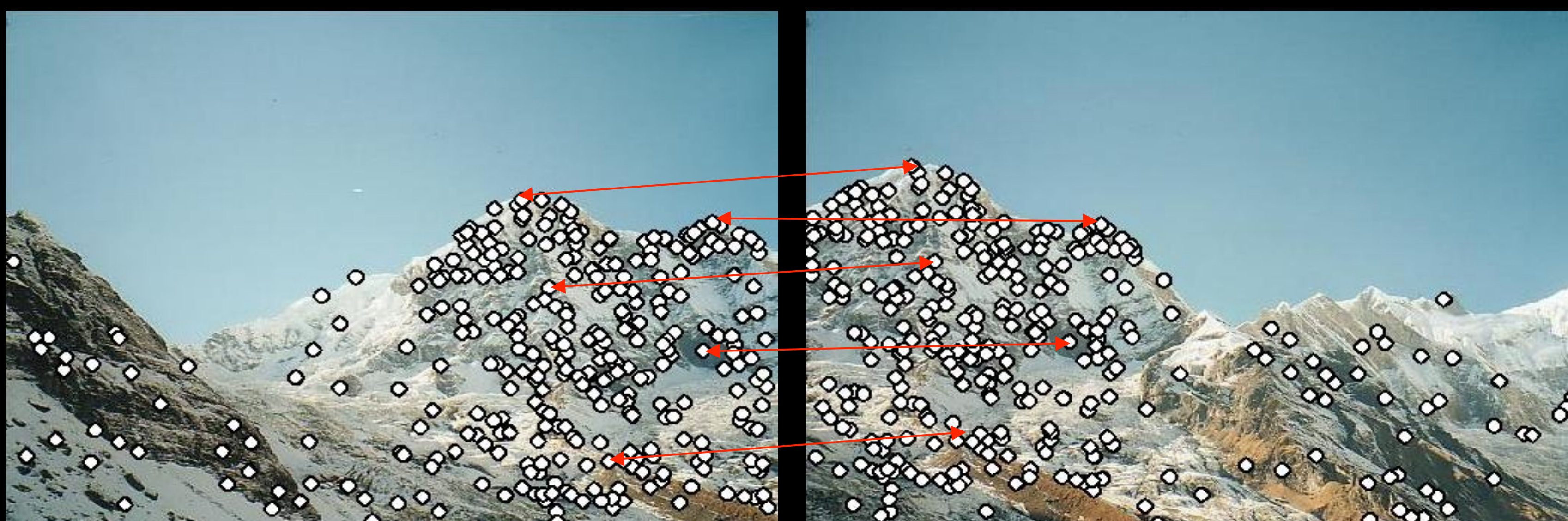
Scalado Rewind

We need to match (align) images

Detect feature points in both images



Find corresponding pairs



Use these pairs to align images

Matching with Features

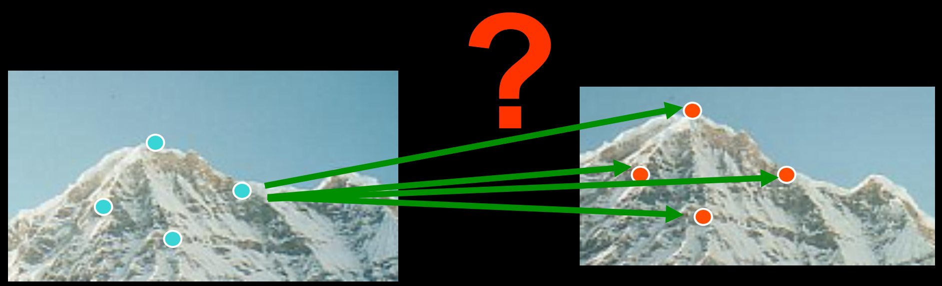
- **Problem 1:**
  - **Detect the *same* point *independently* in both images**

no chance to match!

We need a repeatable detector

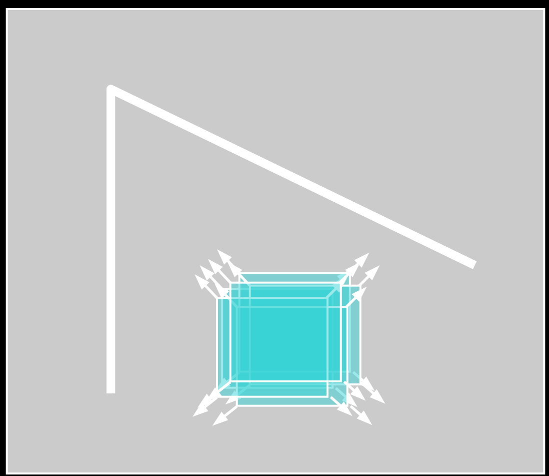
Matching with Features

- **Problem 2:**
  - **For each point correctly recognize the corresponding one**

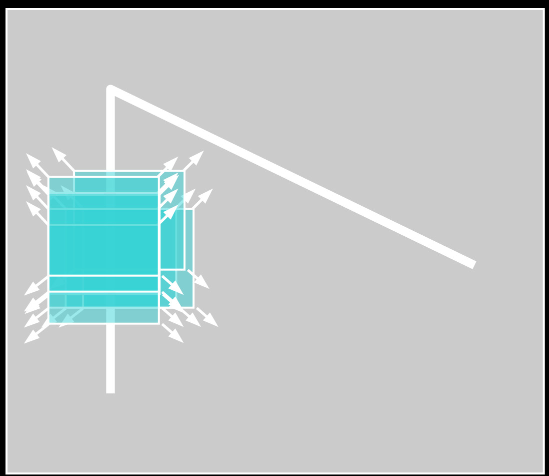


We need a reliable and distinctive descriptor

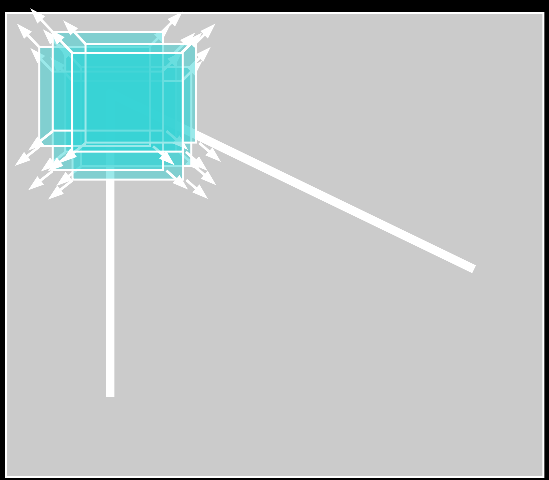
Harris Detector: Basic Idea



"flat" region:
no change in all directions



"edge":
no change along the edge direction



"corner":
significant change in all directions

Harris Detector: Mathematics

Window-averaged change of intensity for the shift [*u*,*v*]:

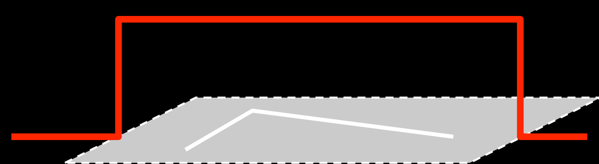
$$E(u,v) = \sum_{x,y} w(x,y)\left[I(x+u, y+v) - I(x,y)\right]^2$$

Window function

Shifted intensity

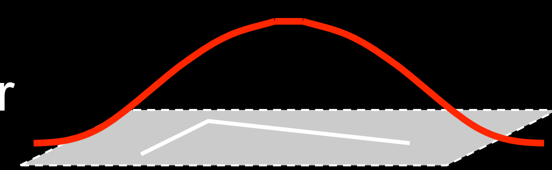
Intensity

Window function $w(x,y)$ =



1 in window, 0 outside

or



Gaussian

Harris Detector: Mathematics

Expanding *E(u,v)* in a 2nd order Taylor series expansion, we have, for small shifts $[u, v]$, a *bilinear* approximation:

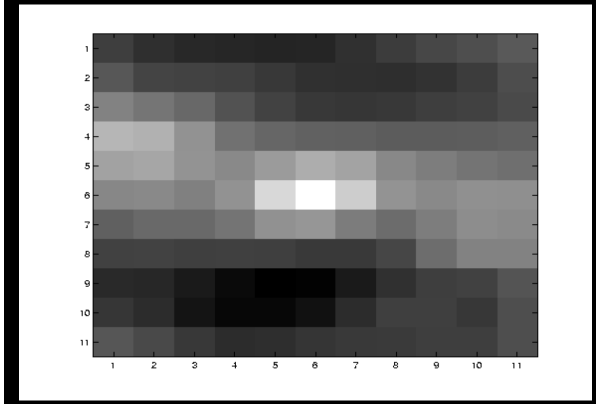
$$E(u,v) \cong [u,v] \; M \begin{bmatrix} u \\ v \end{bmatrix}$$

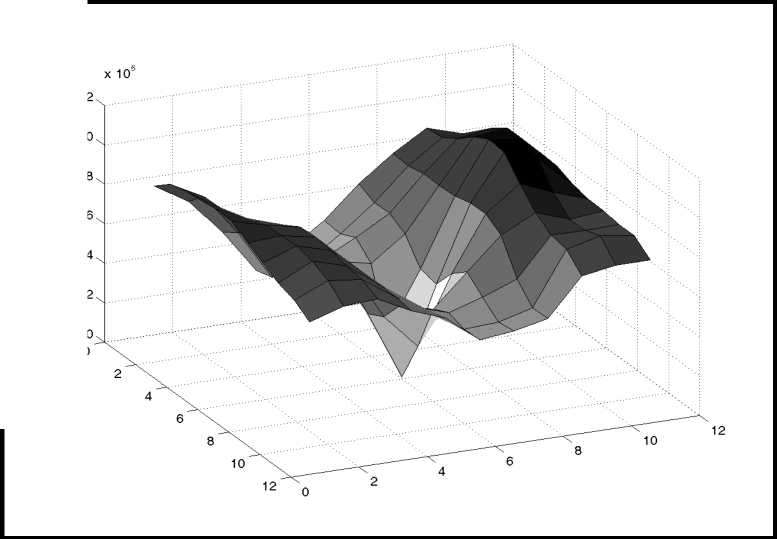
where *M* is a 2×2 matrix computed from image derivatives:

$$M = \sum_{x,y} w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

$$I_x = I(x,y) - I(x-1,y)$$

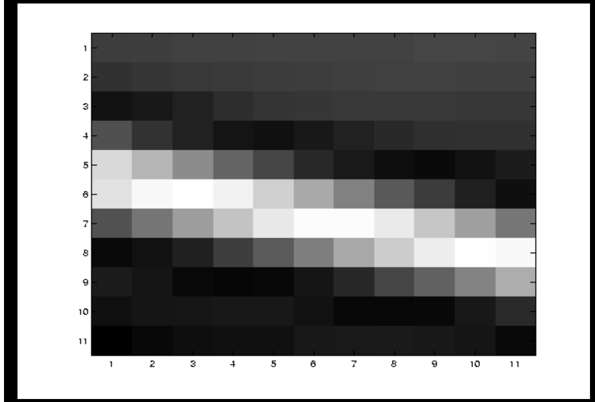
Eigenvalues λ_1, λ_2 of M at different locations

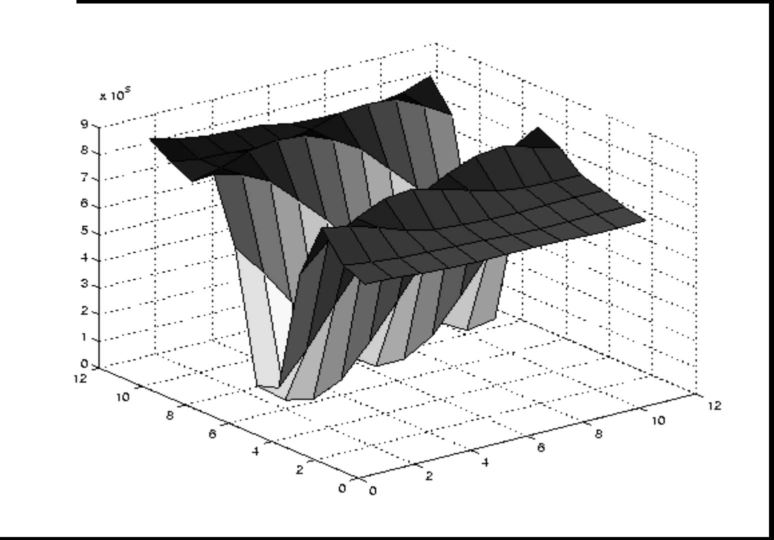




λ_1 and λ_2 are large

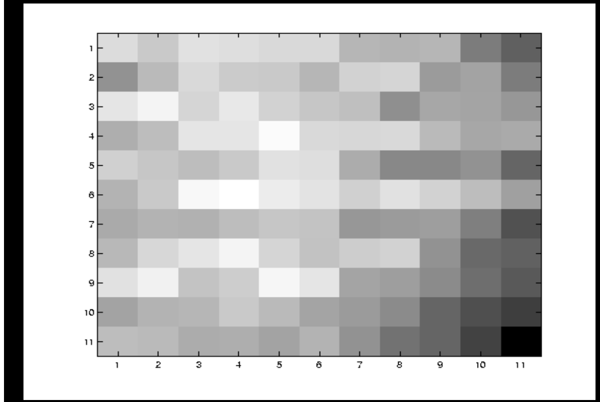
Eigenvalues λ_1, λ_2 of M at different locations

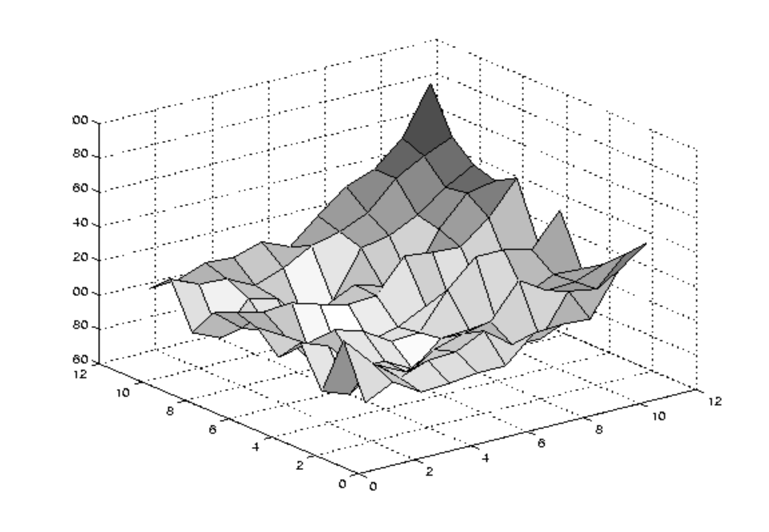




large λ_1, small λ_2

Eigenvalues λ_1, λ_2 of M at different locations





small λ_1, small λ_2

Harris Detector: Mathematics

Measure of corner response:

$$M = \sum_{x,y} w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

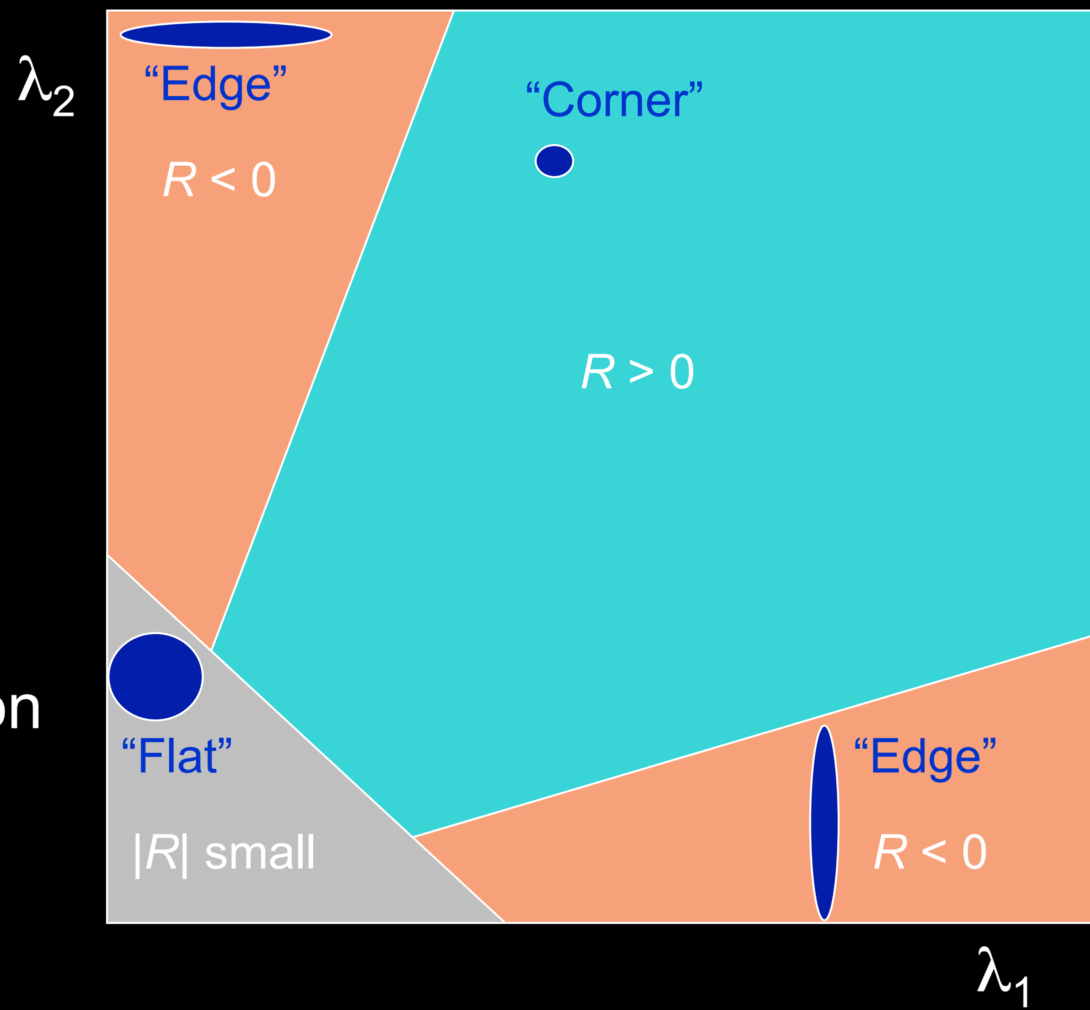
$$\det M = \lambda_1 \lambda_2$$
$$\operatorname{trace} M = \lambda_1 + \lambda_2$$

$$R = \det M - k\left(\operatorname{trace} M\right)^2$$

(*k* – empirical constant, *k* = 0.04 - 0.06)

Harris Detector: Mathematics

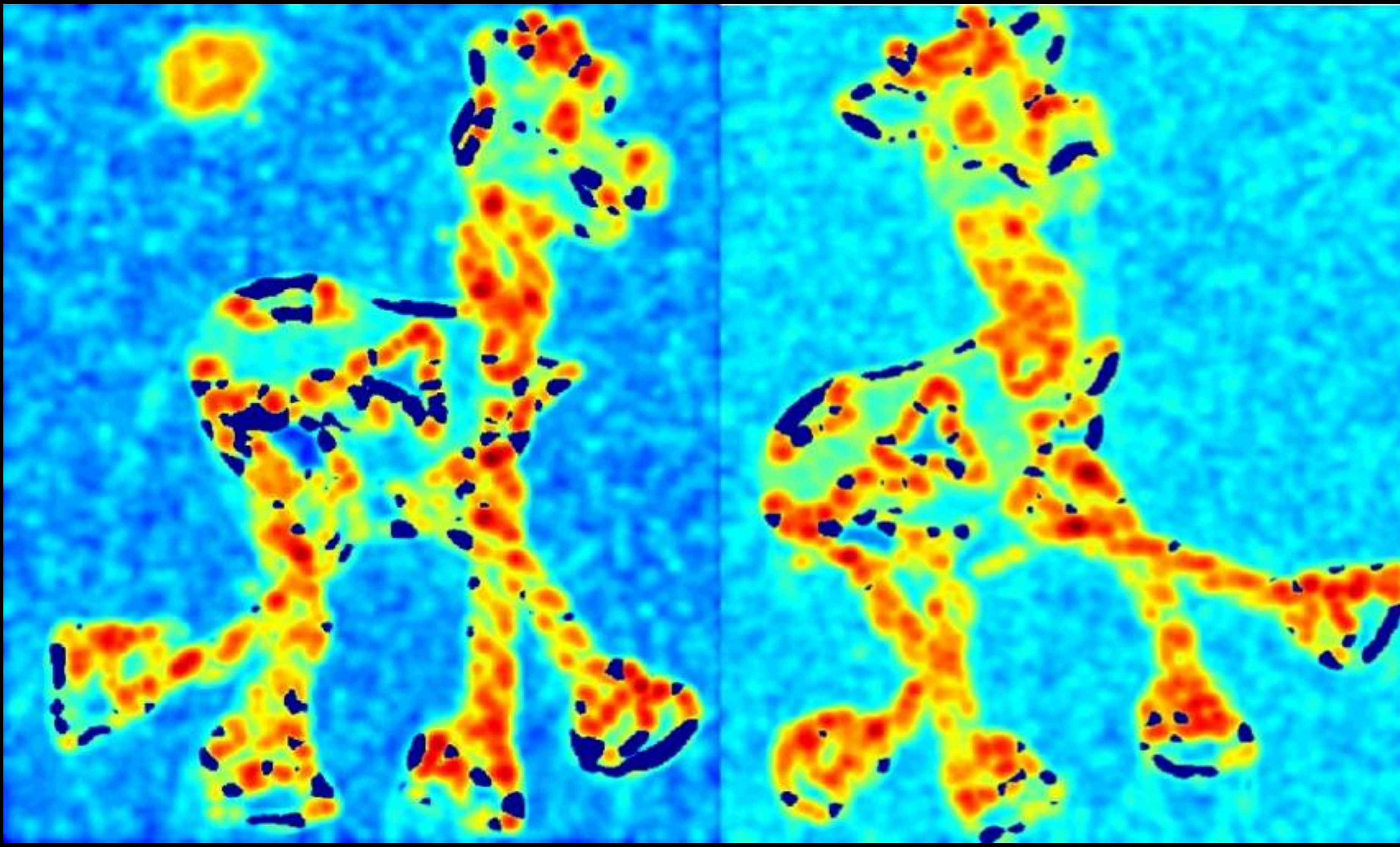
- R depends only on eigenvalues of M
- R is large for a corner
- R is negative with large magnitude for an edge
- $|R|$ is small for a flat region



Harris Detector: Workflow

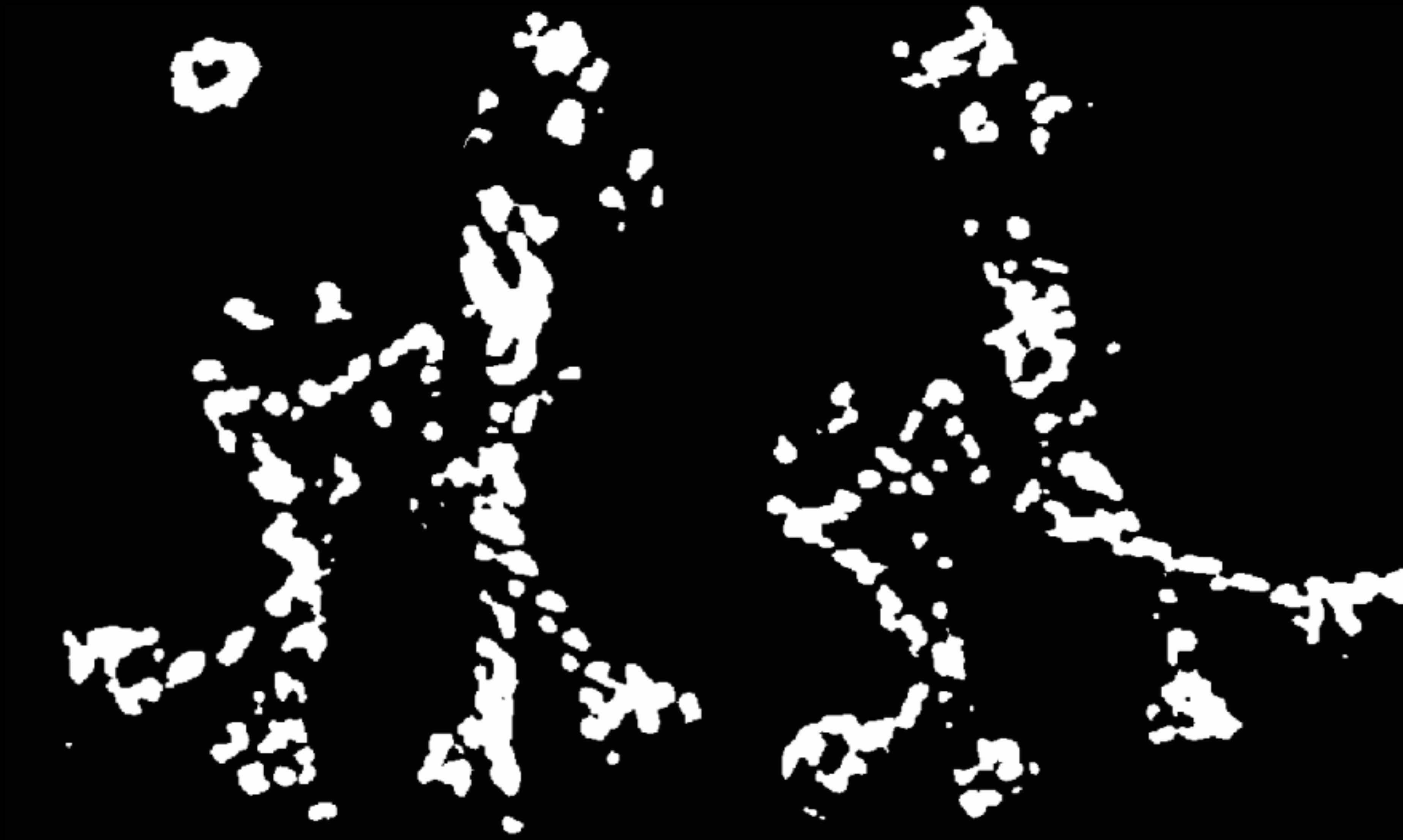
Harris Detector: Workflow

Compute corner response R



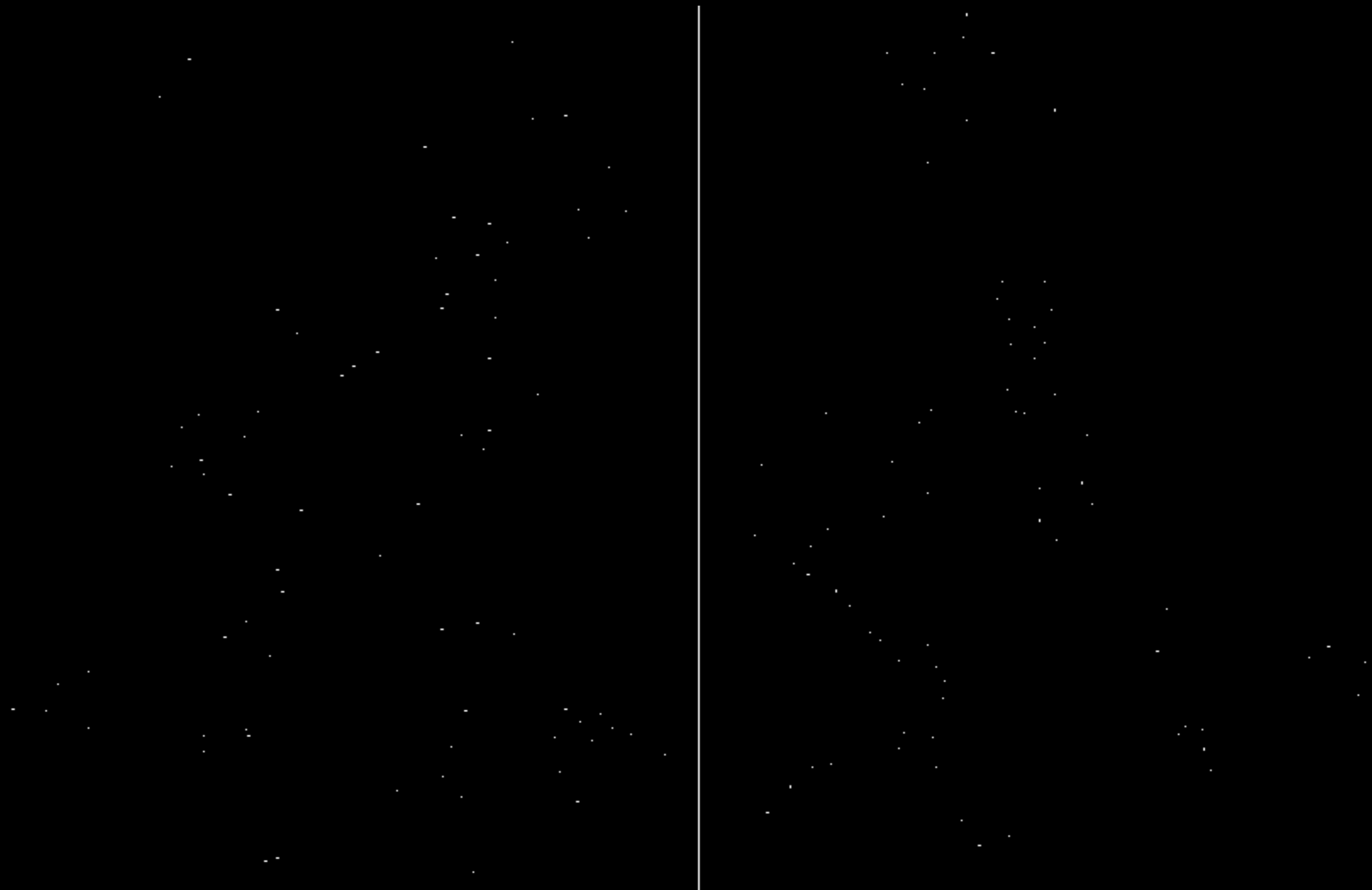
Harris Detector: Workflow

Find points with large corner response: $R >$ threshold



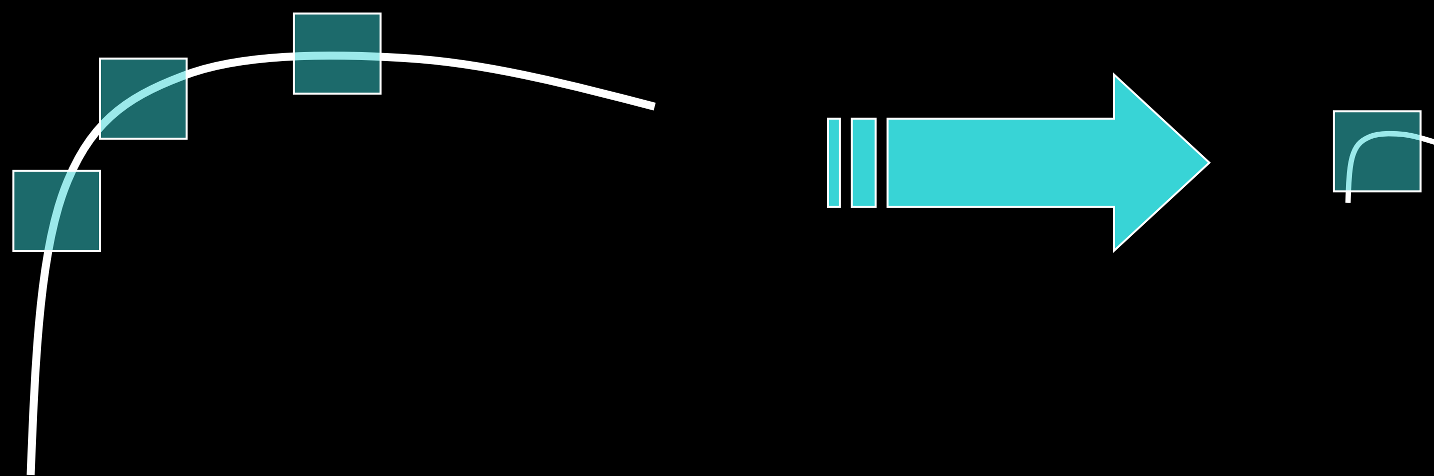
Harris Detector: Workflow

Take only the points of local maxima of R



Harris Detector: Workflow

Not invariant to image scale!

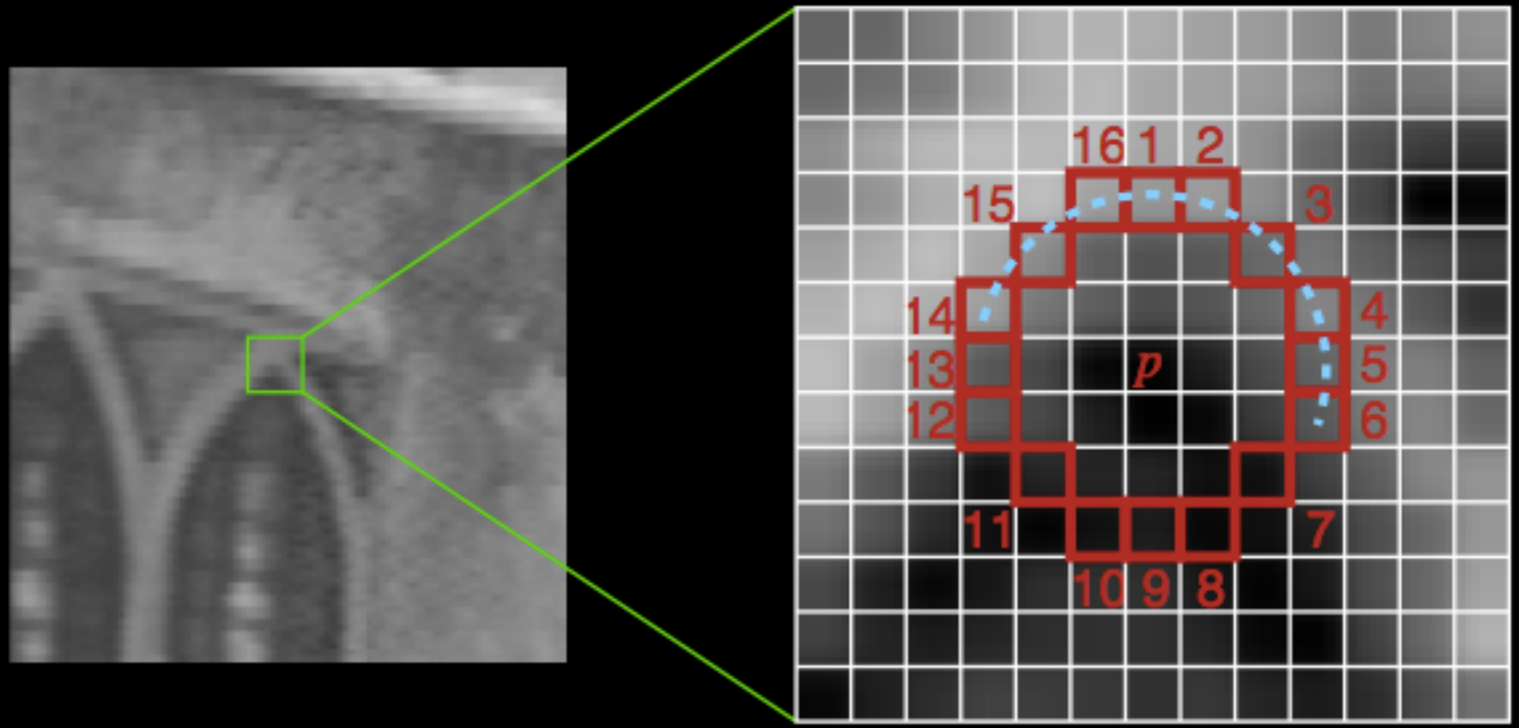


All points will be classified as edges

Corner !

FAST Corners

- Look for a contiguous arc of N pixels
  - all much darker (or brighter) than the central pixel *p*



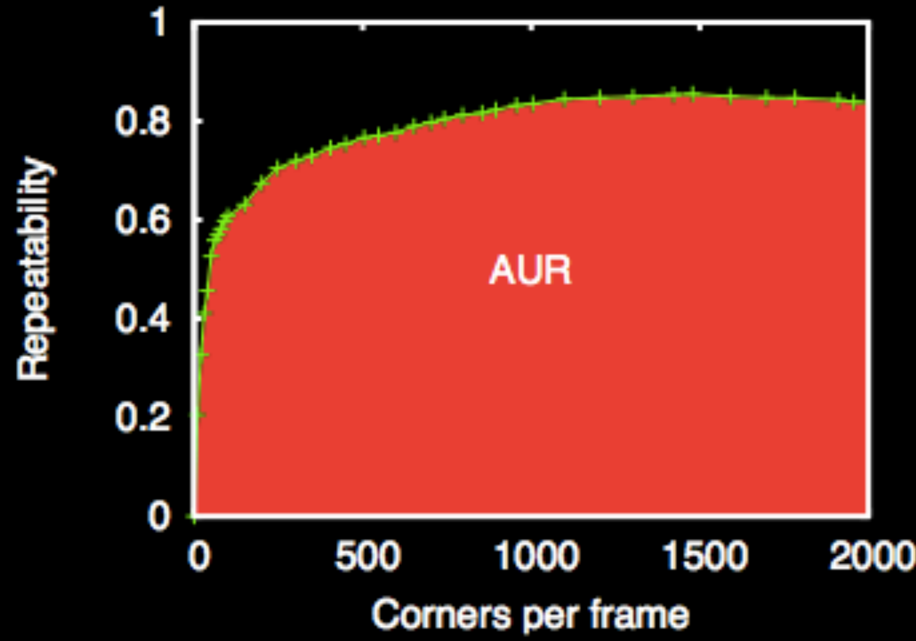
It actually is fast

Detector	Set 1 Pixel rate (MPix/s)
FAST $n = 9$	188
FAST $n = 12$	158
Original FAST ($n = 12$)	79.0
FAST-ER	75.4
SUSAN	12.3
Harris	8.05
Shi-Tomasi	6.50
DoG	4.72

And repeatable!

AUR = Area Under Repeatability curve

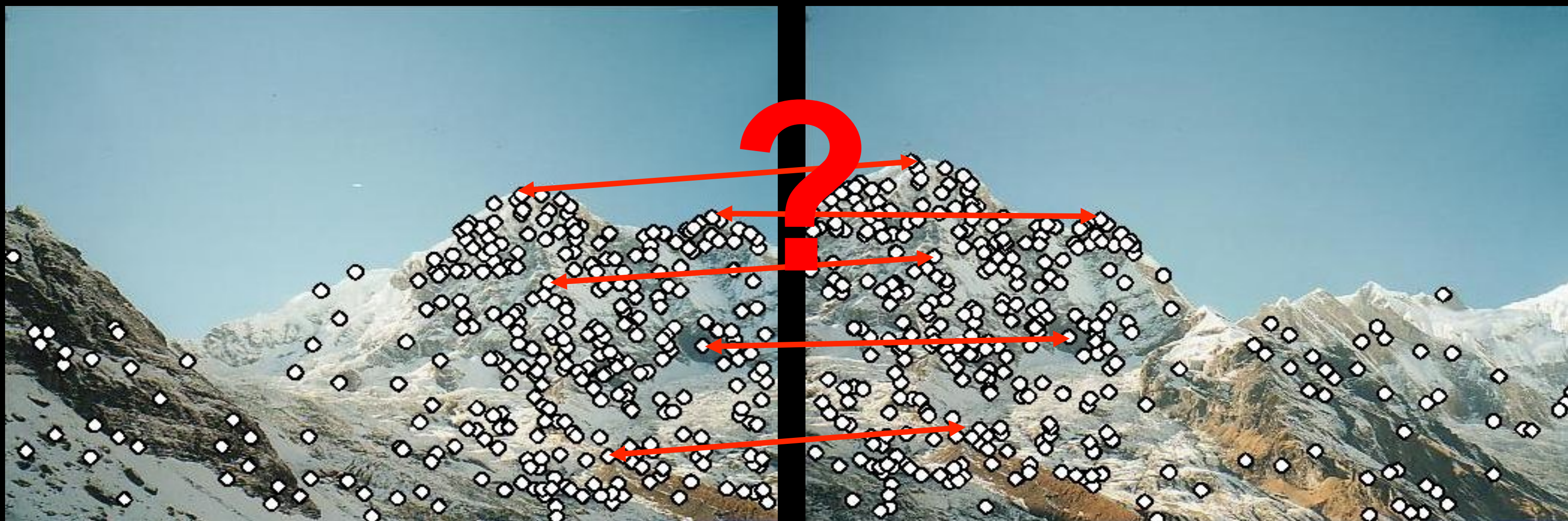
Detector	AUR
FAST-ER	1313.6
FAST-9	1304.57
DoG	1275.59
Shi & Tomasi	1219.08
Harris	1195.2
Harris-Laplace	1153.13
FAST-12	1121.53
SUSAN	1116.79
Random	271.73



Point Descriptors

- **We know how to detect points**
- **Next question:**

How to match them?



Point descriptor should be:

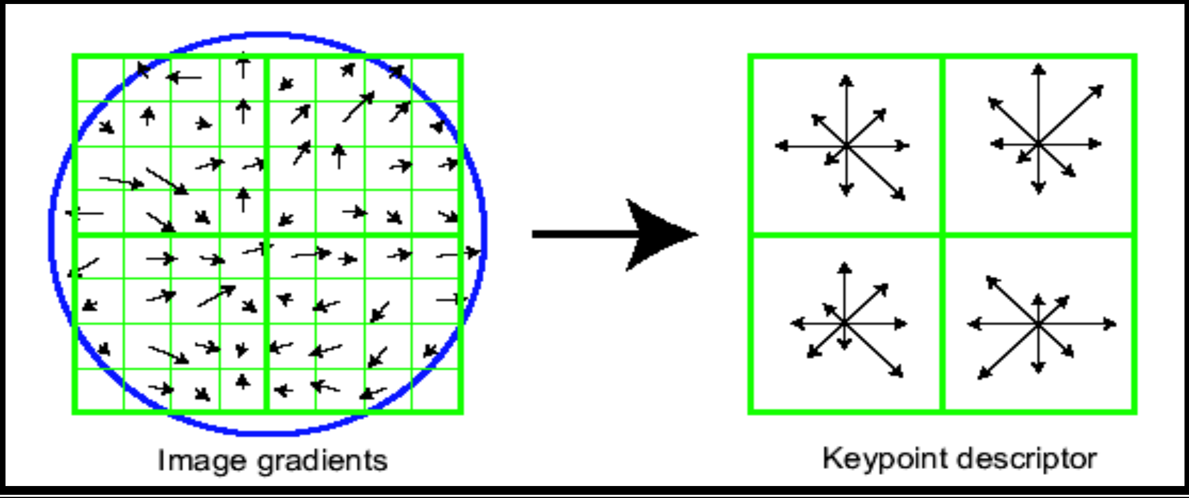
1. Invariant
2. Distinctive

SIFT – Scale Invariant Feature Transform

Figure 12: The training images for two objects are shown on the left. These can be recognized in a cluttered image with extensive occlusion, shown in the middle. The results of recognition are shown on the right. A parallelogram is drawn around each recognized object showing the boundaries of the original training image under the affi ne transformation solved for during recognition. Smaller squares indicate the keypoints that were used for recognition.

SIFT – Scale Invariant Feature Transform

- Descriptor overview:
  - Determine scale (by maximizing DoG in scale and in space), local orientation as the dominant gradient direction
  - Use this scale and orientation to make all further computations invariant to scale and rotation
  - Compute gradient orientation histograms of several small windows (128 values for each point)
  - Normalize the descriptor to make it invariant to intensity change

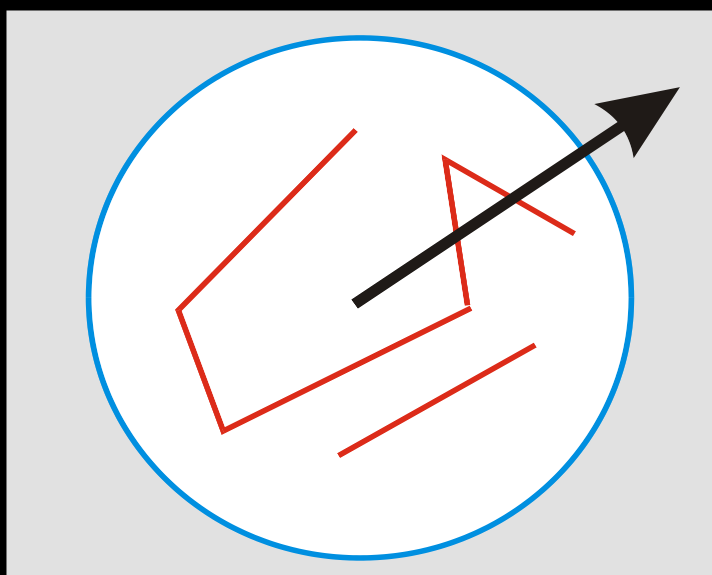


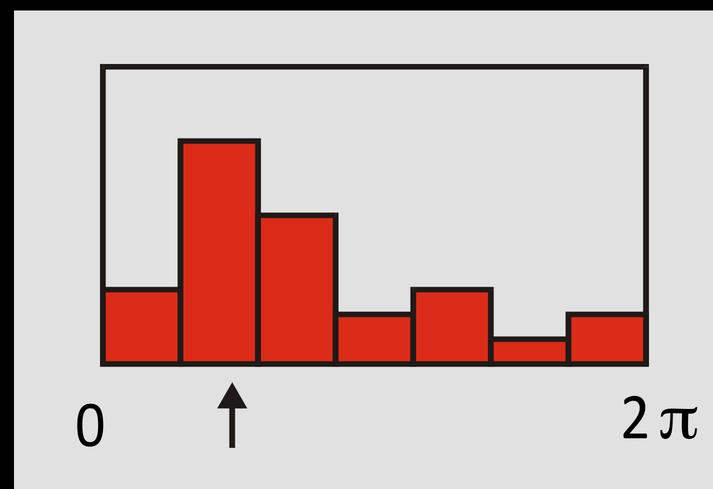
D. Lowe. "Distinctive Image Features from Scale-Invariant Keypoints" IJCV 2004

Match orientations

Determine the local orientation

- **Within the image patch**
  - **estimate dominant gradient direction**
  - **collect a histogram of gradient orientations**
  - **find the peak, rotate the patch so it becomes vertical**

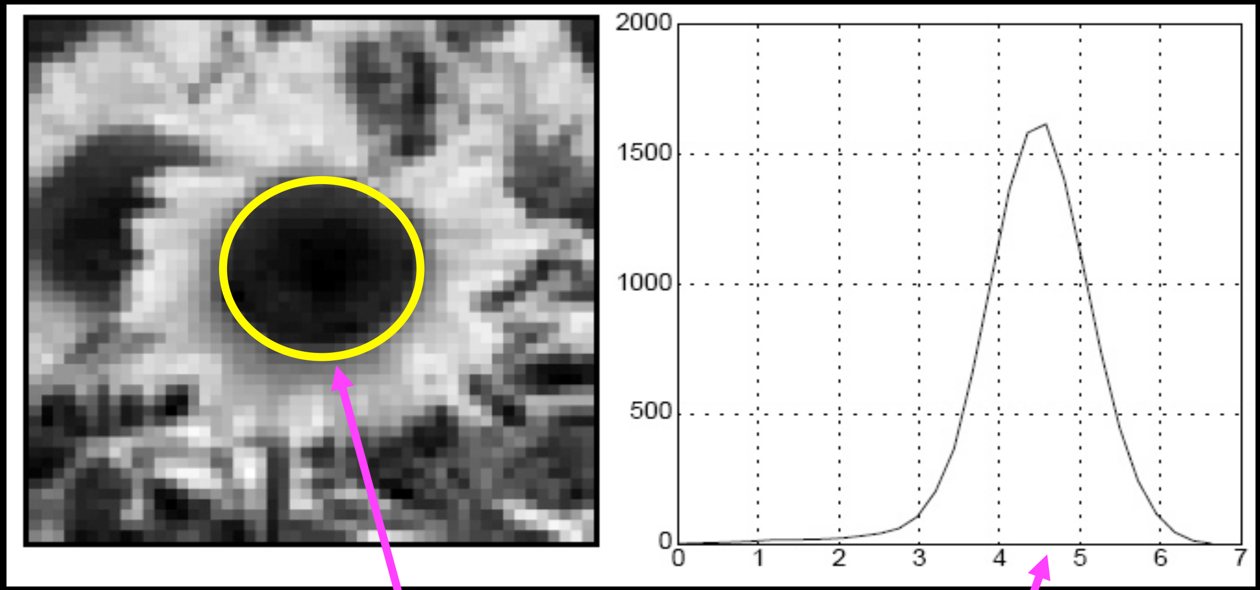




D. Lowe. "Distinctive Image Features from Scale-Invariant Keypoints" IJCV 2004

Determine the scale

- **We define the characteristic scale as the scale that produces peak of Laplacian response**

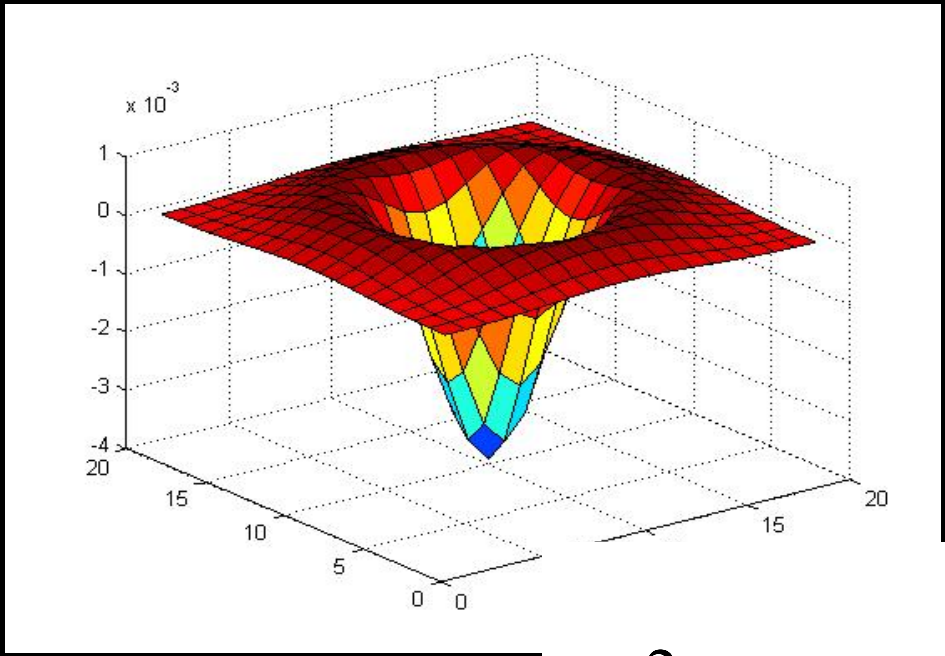


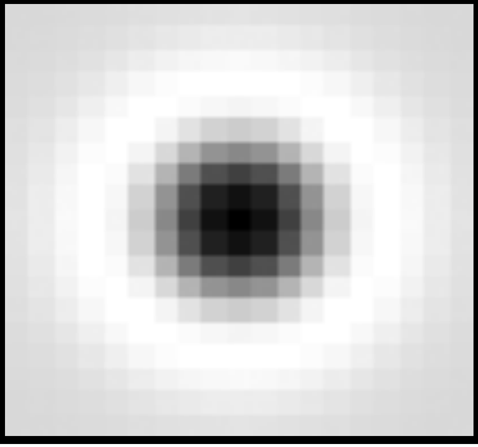
characteristic scale

T. Lindeberg (1998). Feature detection with automatic scale selection. *International Journal of Computer Vision* **30** (2): pp 77--116.

Blob detection in 2D

- **Laplacian of Gaussian: Circularly symmetric operator for blob detection in 2D**





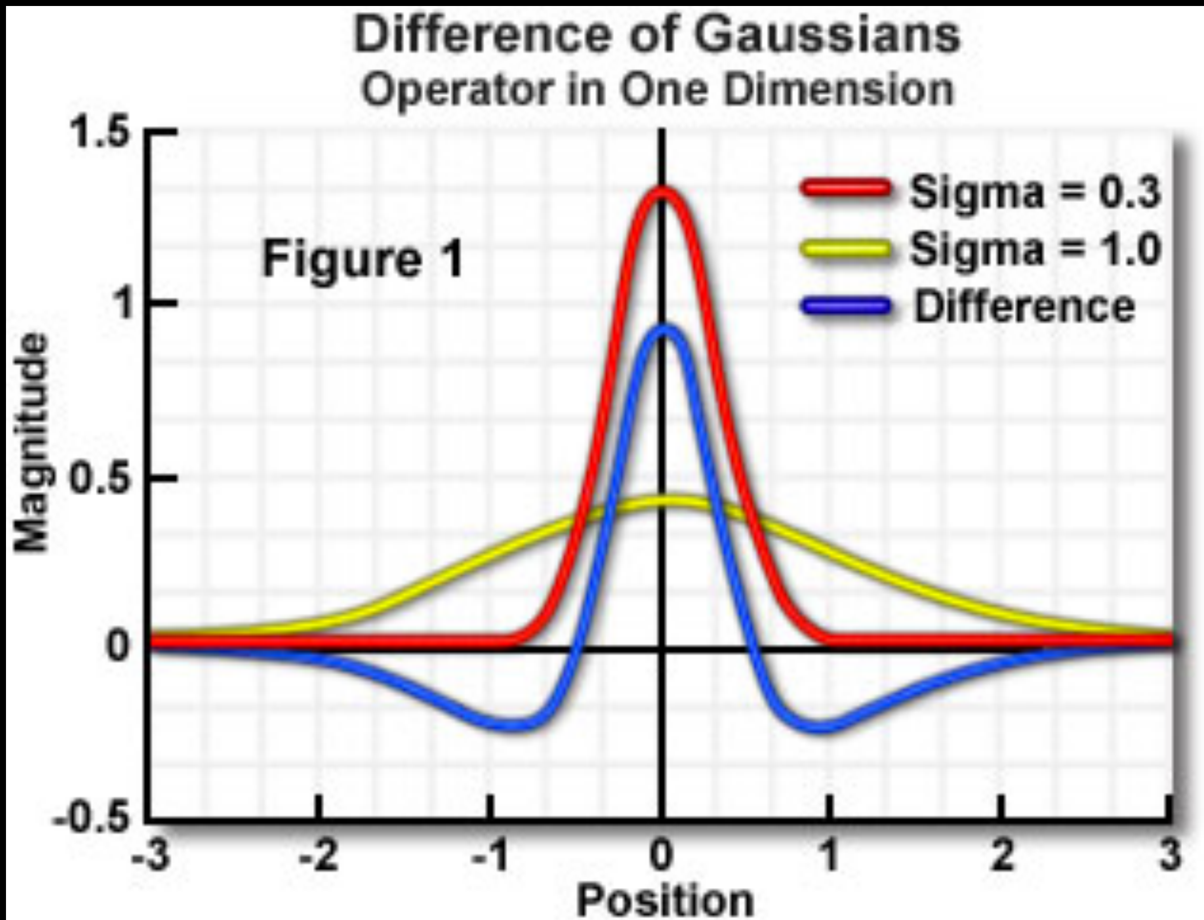
0	1	0
1	−4	1
0	1	0

$$\frac{\partial^2 g}{\partial x^2} = g(x-1,y) - 2g(x,y) + g(x+1,y)$$

$$\nabla^2_{\text{norm}} g = \sigma^2 \left(\frac{\partial^2 g}{\partial x^2} + \frac{\partial^2 g}{\partial y^2} \right)$$

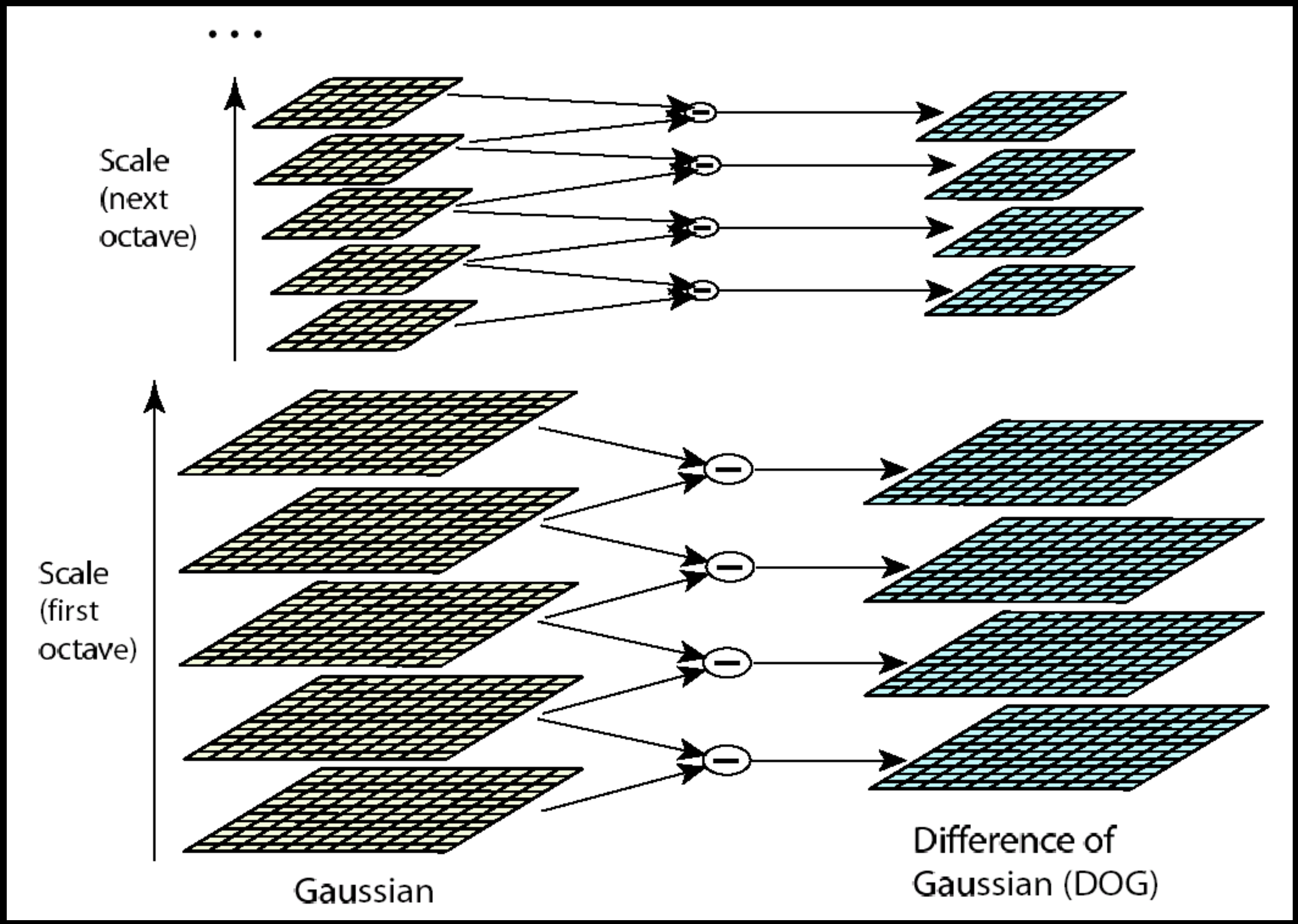
Difference of Gaussians (DoG)

- **Laplacian of Gaussian can be approximated by the difference between two different Gaussians**



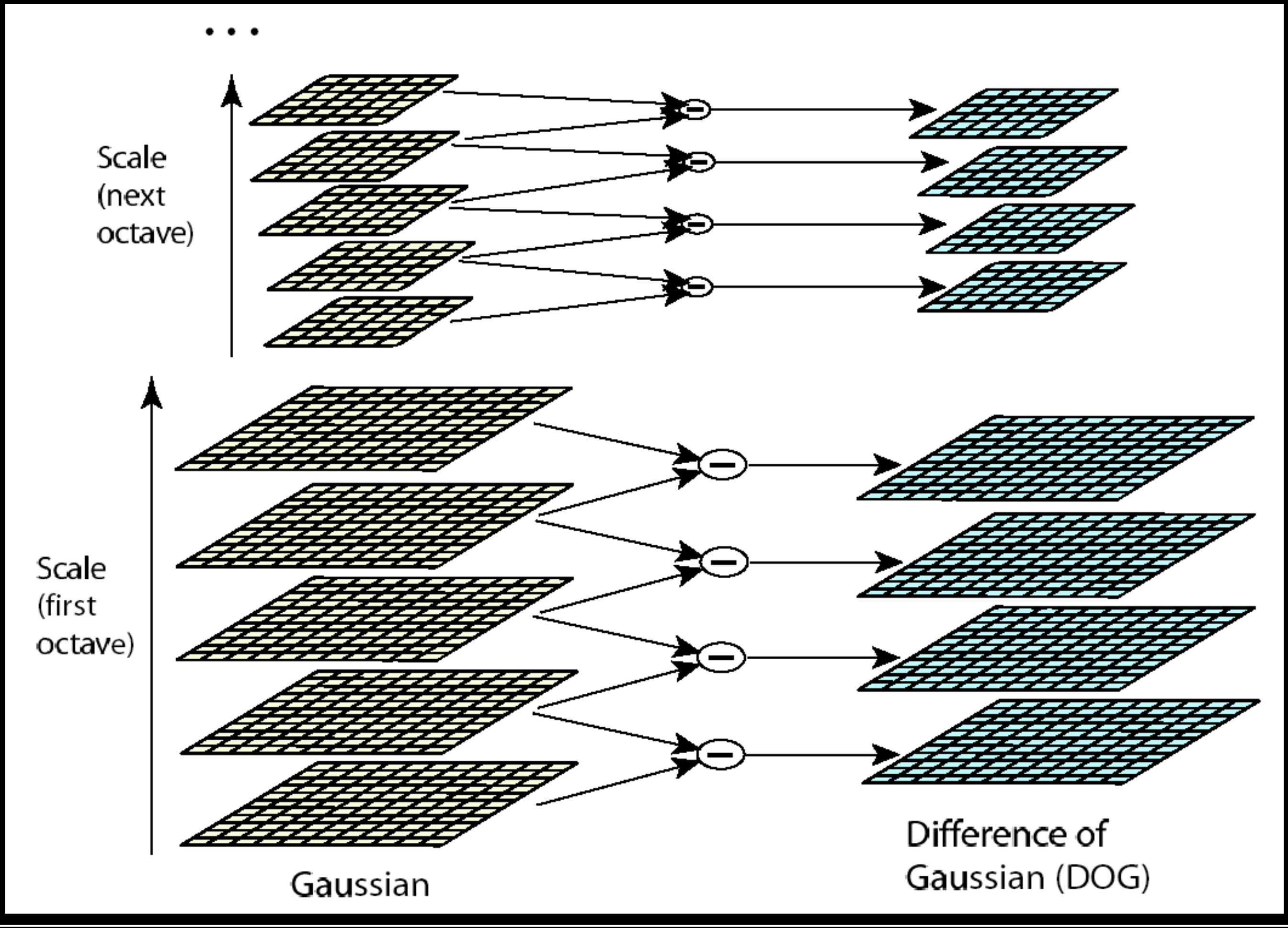
Create a pack of DoGs

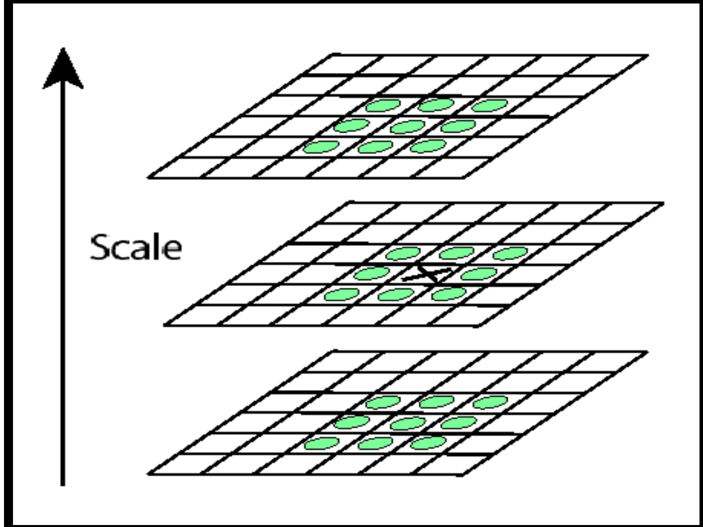
- **Fast computation, process scale space an octave at a time**



Determine the scale

- **Find a local maximum in space and scale**





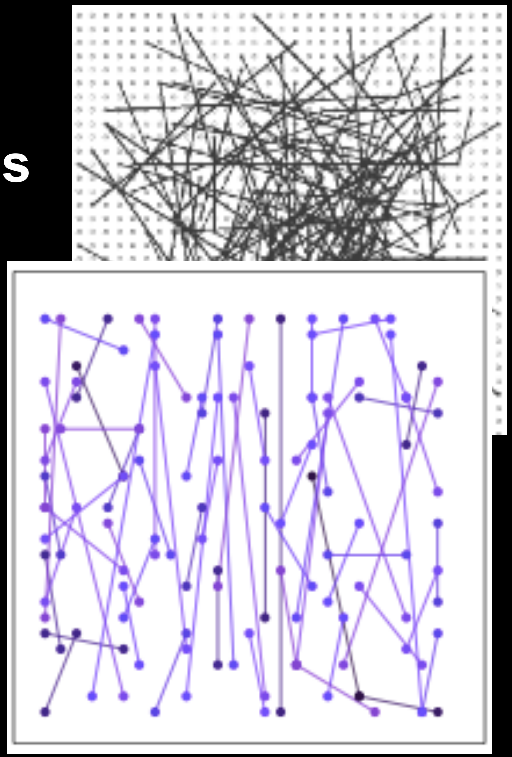
D. Lowe. "Distinctive Image Features from Scale-Invariant Keypoints" IJCV 2004

ORB (Oriented FAST and Rotated BRIEF)

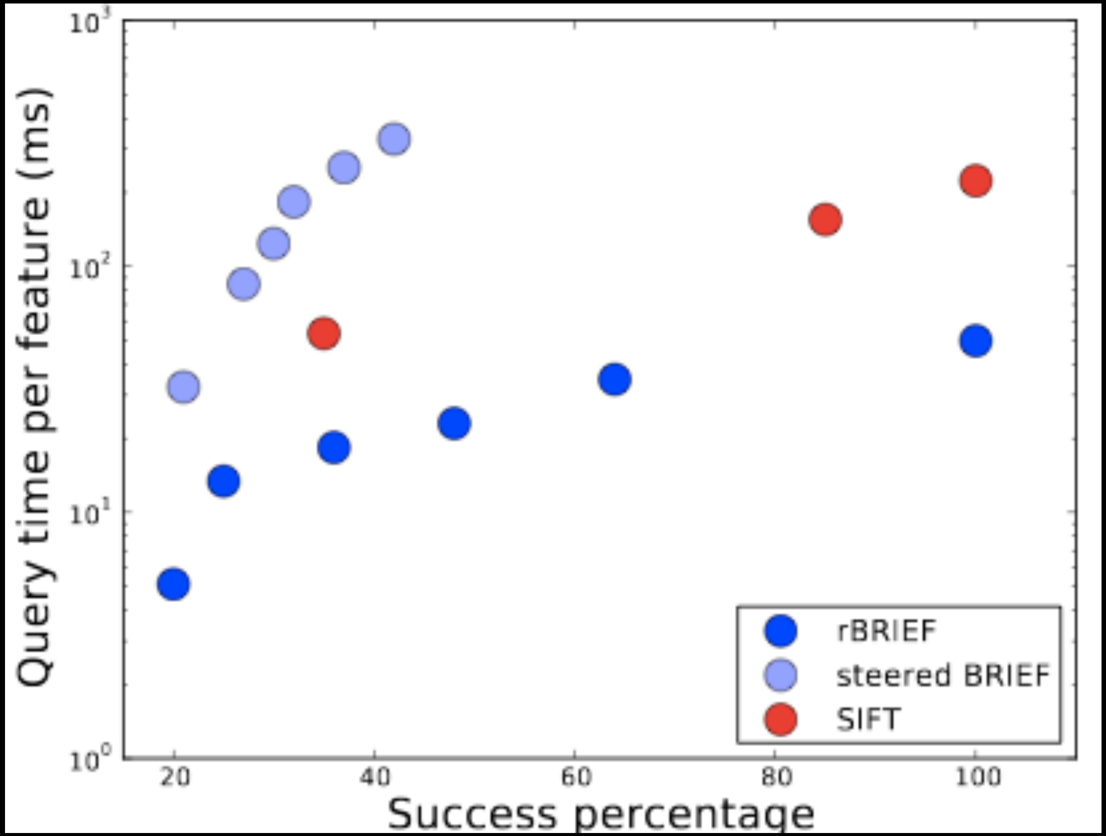
- Use FAST-9
  - use Harris measure to order them
- Find orientation
  - calculate weighted new center
  - reorient image so that gradients vary vertically

$$\left(\frac{\sum x I(x,y)}{\sum I(x,y)}, \frac{\sum y I(x,y)}{\sum I(x,y)} \right)$$

- BRIEF
  - Binary Robust Independent Elementary Features
  - choose pixels to compare, result creates 0 or 1
  - combine to a binary vector, compare using Hamming distance (XOR + pop count)
- Rotated BRIEF
  - train a good set of pixels to compare



rBRIEF vs. SIFT



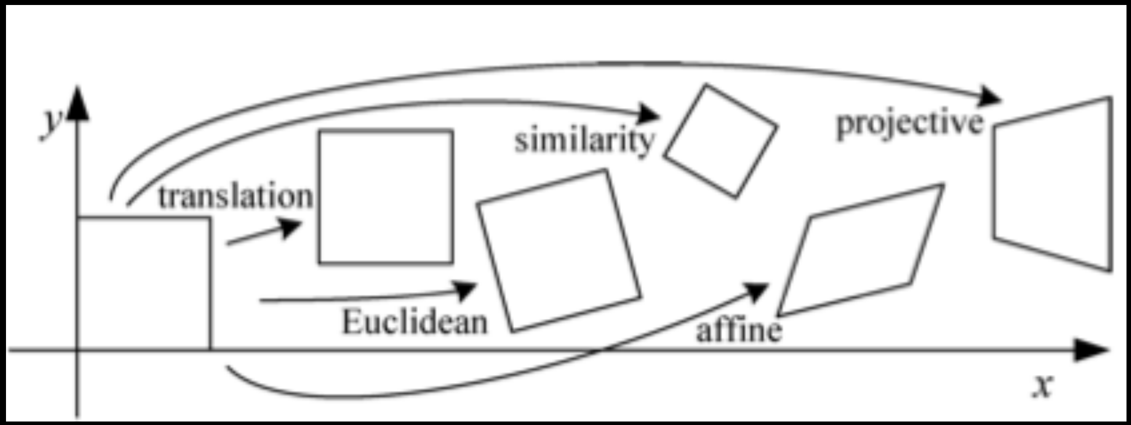
Aligning images: Translation?

left on top

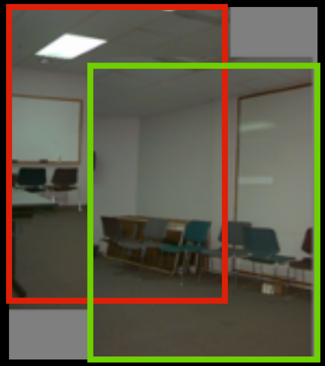
right on top

Translations are not enough to align the images

Which transform to use?

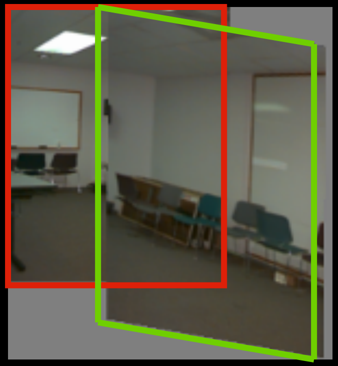


Translation



2 unknowns

Affine

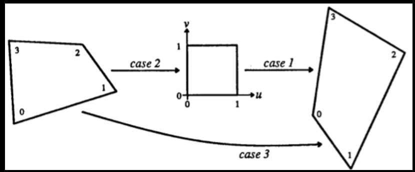


6 unknowns

Perspective

8 unknowns

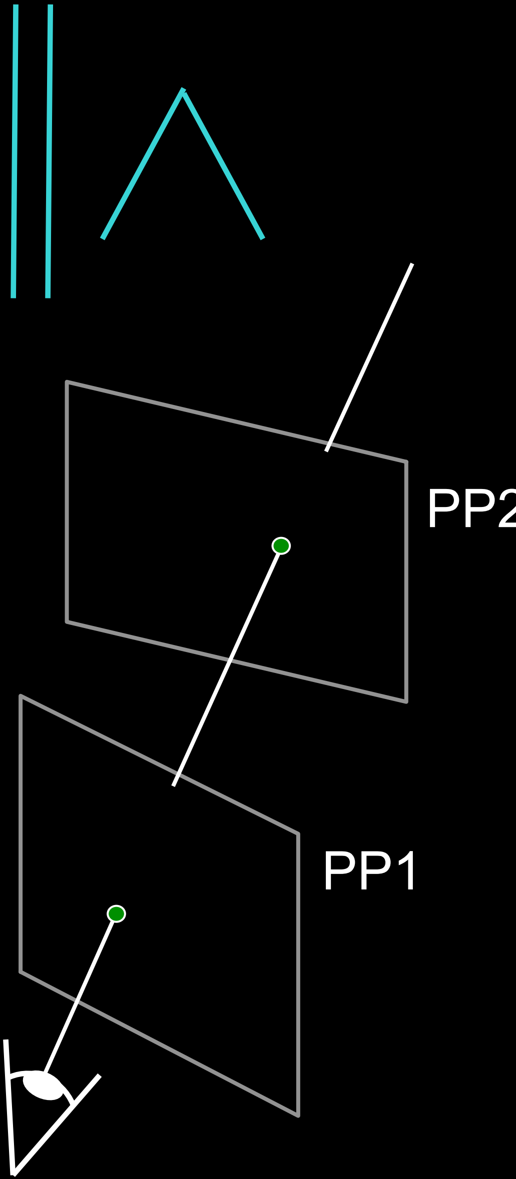
Homography



- **Projective mapping between any two PPs with the same center of projection**
  - **rectangle maps to almost arbitrary quadrilateral**
  - **parallel lines do not remain parallel**
  - **but must preserve straight lines**

is called a Homography

$$\underbrace{\begin{bmatrix} wx' \\ wy' \\ w \end{bmatrix}}_{\mathbf{p'}} = \underbrace{\begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix}}_{\mathbf{H}} \underbrace{\begin{bmatrix} x \\ y \\ 1 \end{bmatrix}}_{\mathbf{p}}$$



- **To apply a homography H**
  - **compute p' = Hp** ***(regular matric multiply)***
  - **convert p' from homogeneous to image coordinates [x', y']** ***(divide by w)***

Homography from mapping quads

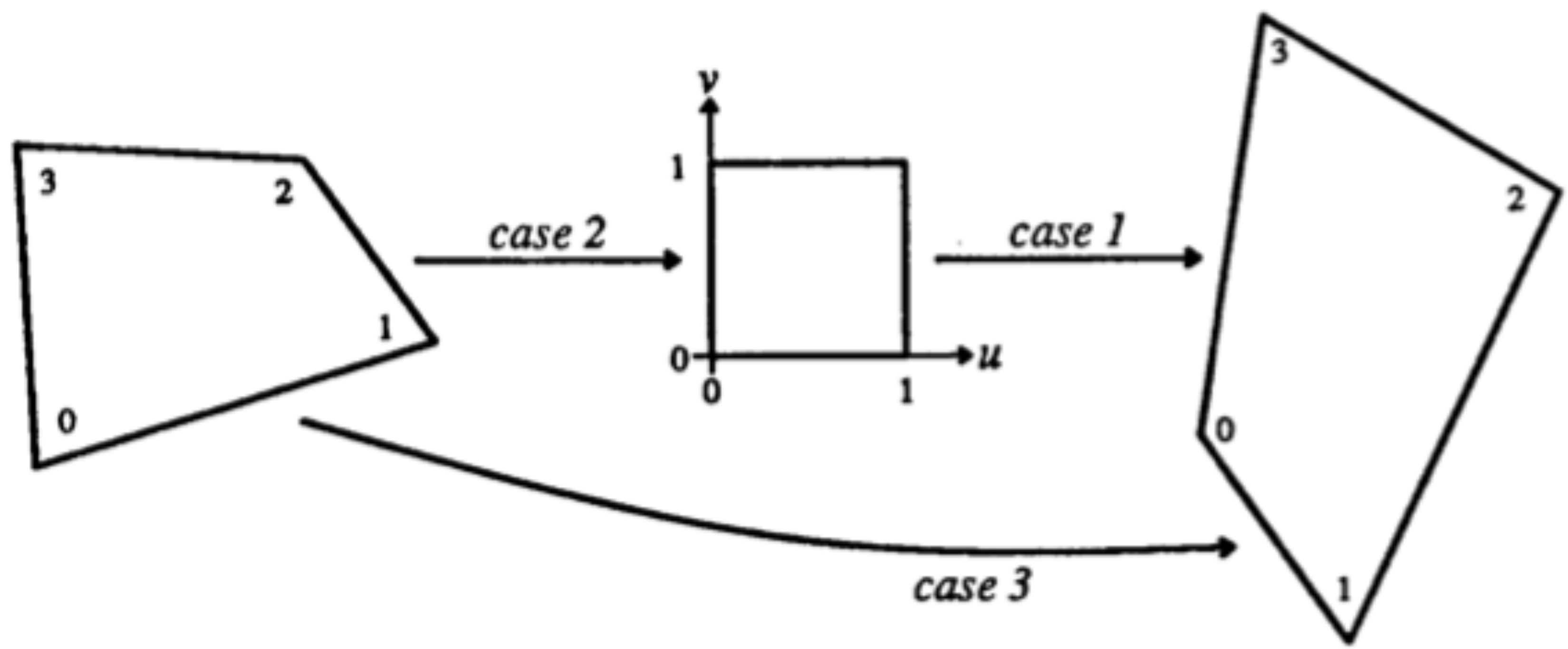


Figure 2.8: *Quadrilateral to quadrilateral mapping as a composition of simpler mappings.*

Fundamentals of Texture Mapping and Image Warping
Paul Heckbert, M.Sc. thesis, U.C. Berkeley, June 1989, 86 pp.
http://www.cs.cmu.edu/~ph/texfund/texfund.pdf

Homography from *n* point pairs (x,y ; x',y')

$$\begin{bmatrix} wx' \\ wy' \\ w \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

p' H p

- **Multiply out**
  - $wx' = h_{11} x + h_{12} y + h_{13}$
  - $wy' = h_{21} x + h_{22} y + h_{23}$
  - $w = h_{31} x + h_{32} y + h_{33}$
- **Get rid of w**
  - $(h_{31} x + h_{32} y + h_{33})x' - (h_{11} x + h_{12} y + h_{13}) = 0$
  - $(h_{31} x + h_{32} y + h_{33})y' - (h_{21} x + h_{22} y + h_{23}) = 0$
- **Create a new system Ah = 0**
  - Each point constraint gives two rows of A
    - $[-x \quad -y \quad -1 \quad 0 \quad 0 \quad 0 \quad xx' \quad yx' \quad x']$
    - $[\ 0 \quad 0 \quad 0 \quad -x \quad -y \quad -1 \quad xy' \quad yy' \quad y']$
- **Solve with singular value decomposition of $A = USV^T$**
  - solution is in the nullspace of A
  - the last column of V (= last row of V^T)

$$h = \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \\ h_{33} \end{bmatrix}$$

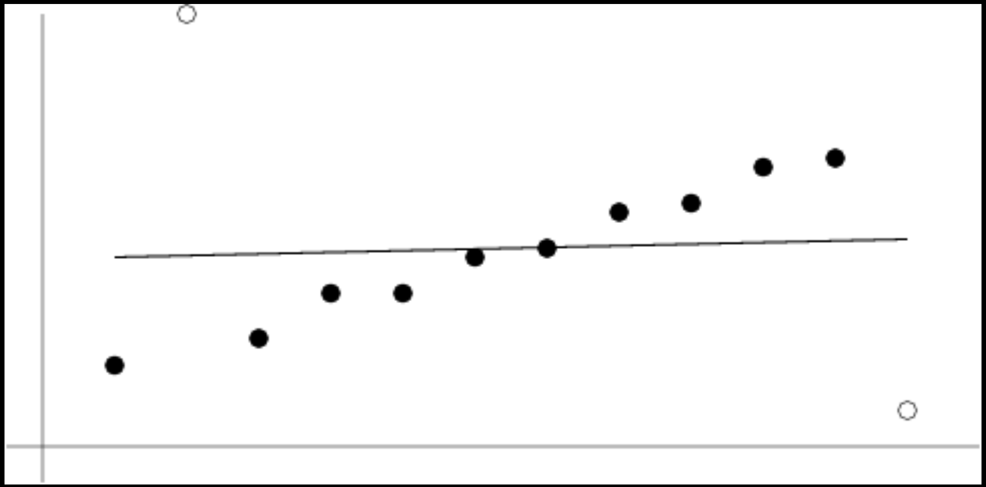
Example

perspective reprojection

Pics: Marc Levoy

What to do with outliers?

- **Least squares OK when error has Gaussian distribution**
- **But it breaks with *outliers***
  - **data points that are not drawn from the same distribution**
- **Mis-matched points are outliers to the Gaussian error distribution**
  - **severely disturbs the Homography**



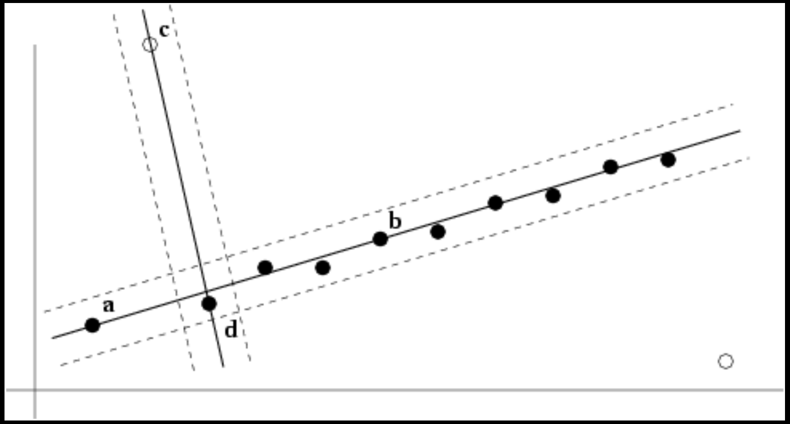
Line fitting using regression is biased by outliers

RANSAC

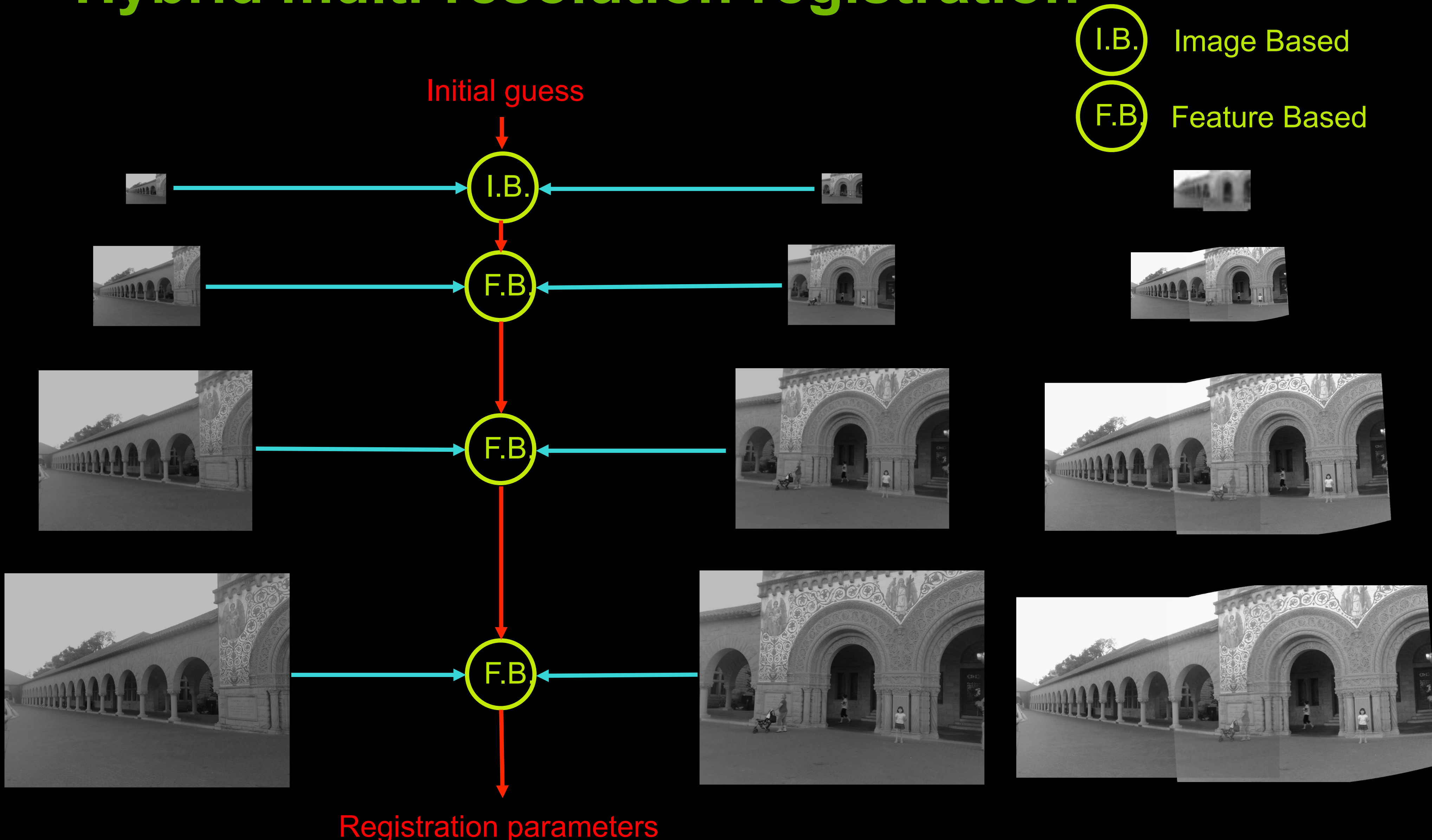
- RANdom SAmple Consensus

1. Randomly choose a subset of data points to fit model (a *sample*)
2. Points within some distance threshold **t** of model are a *consensus set*
   Size of consensus set is model's *support*
3. Repeat for **N** samples; model with the biggest support is the most robust fit
   - Points within distance **t** of best model are inliers
   - Fit final model to all inliers

Two samples
and their supports
for line-fitting

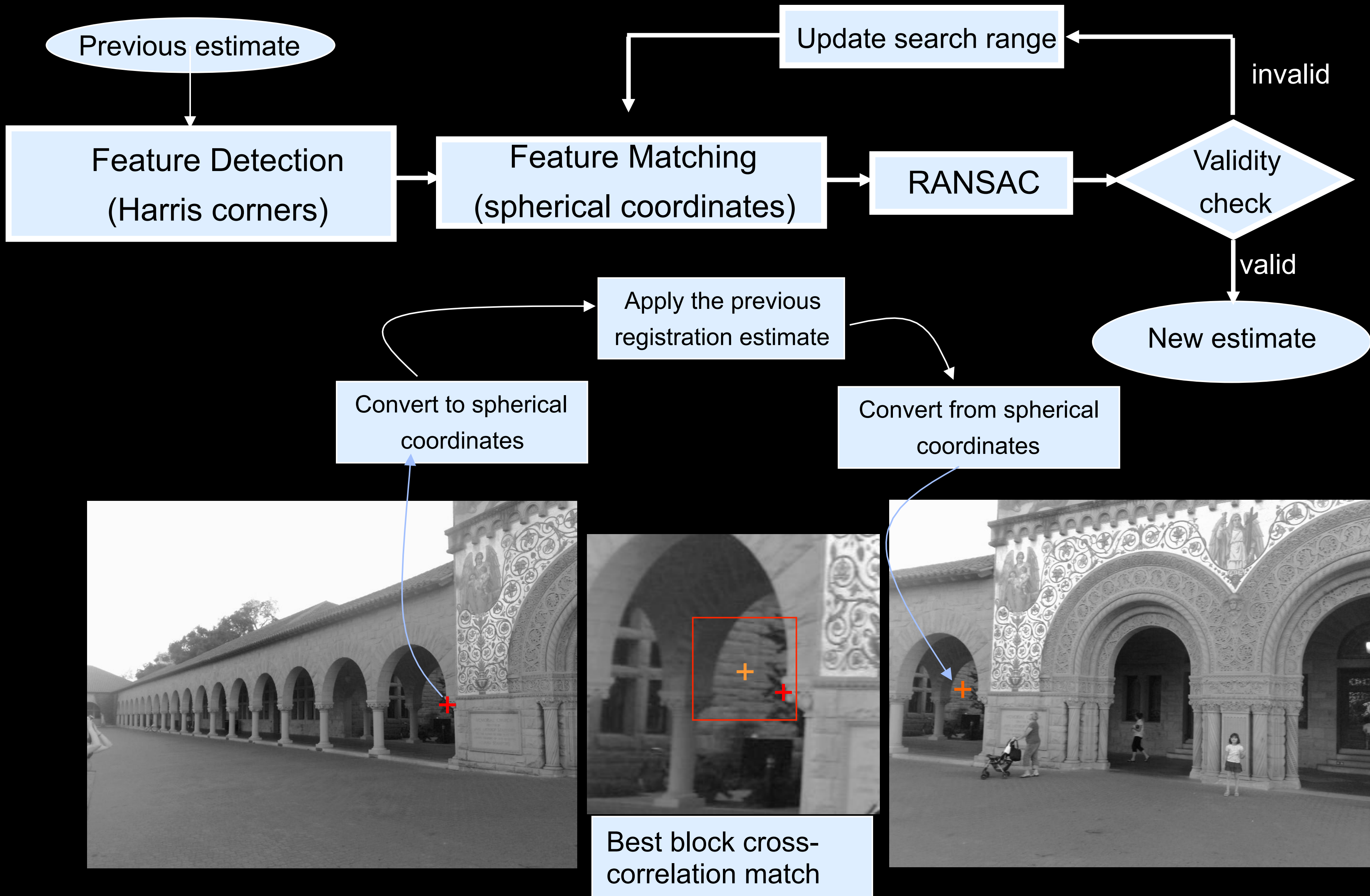


Hybrid multi-resolution registration



K. Pulli, M. Tico, Y. Xiong, X. Wang, C-K. Liang, "Panoramic Imaging System for Camera Phones", ICCE 2010

Feature-based registration



Progression of multi-resolution registration

Actual size

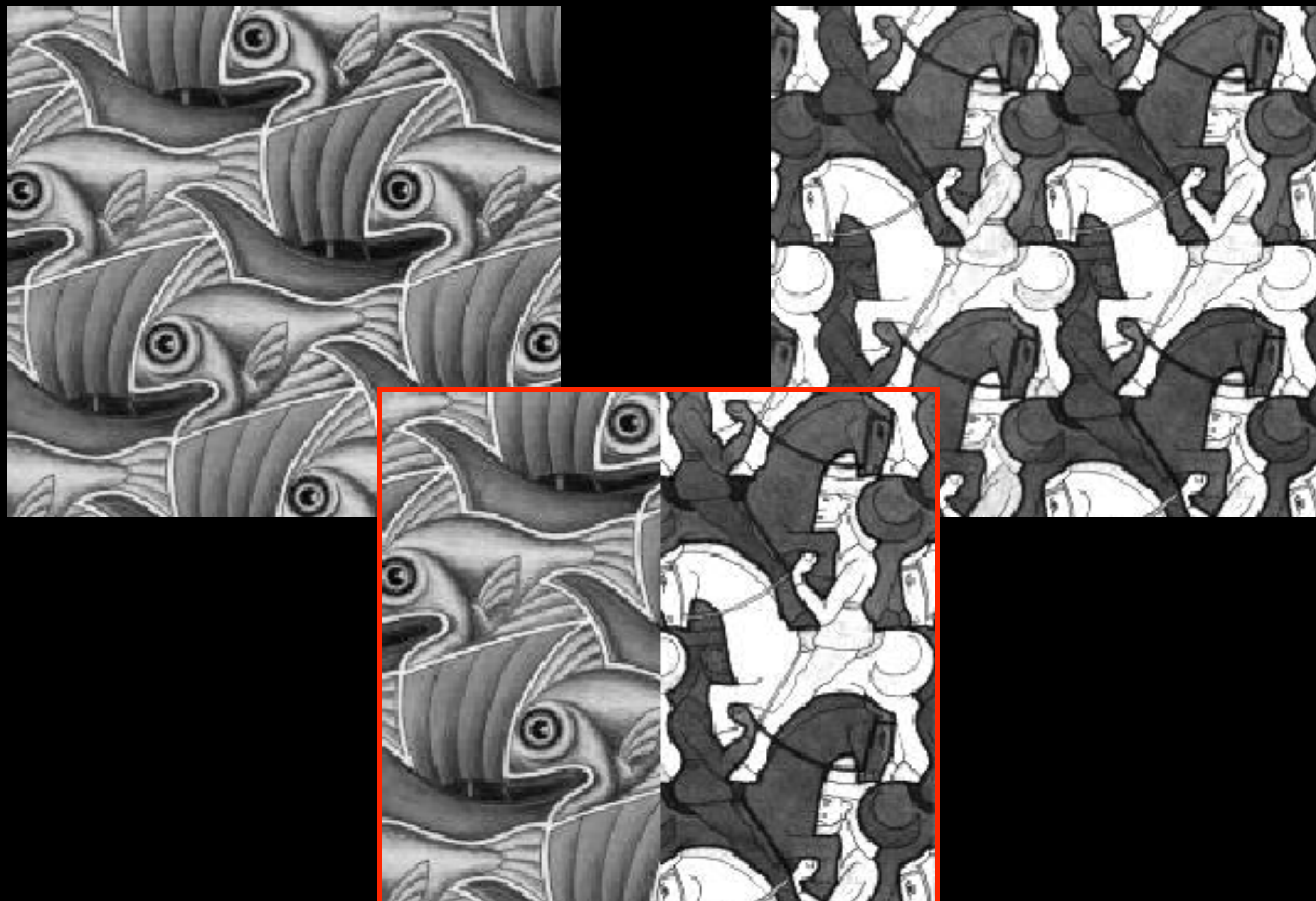
Applied to hi-res

Image blending

- **Directly averaging the overlapped pixels results in ghosting artifacts**
  - **Moving objects, errors in registration, parallax, etc.**

Photo by Chia-Kai Liang

Alpha Blending / Feathering



Alpha Blending / Feathering

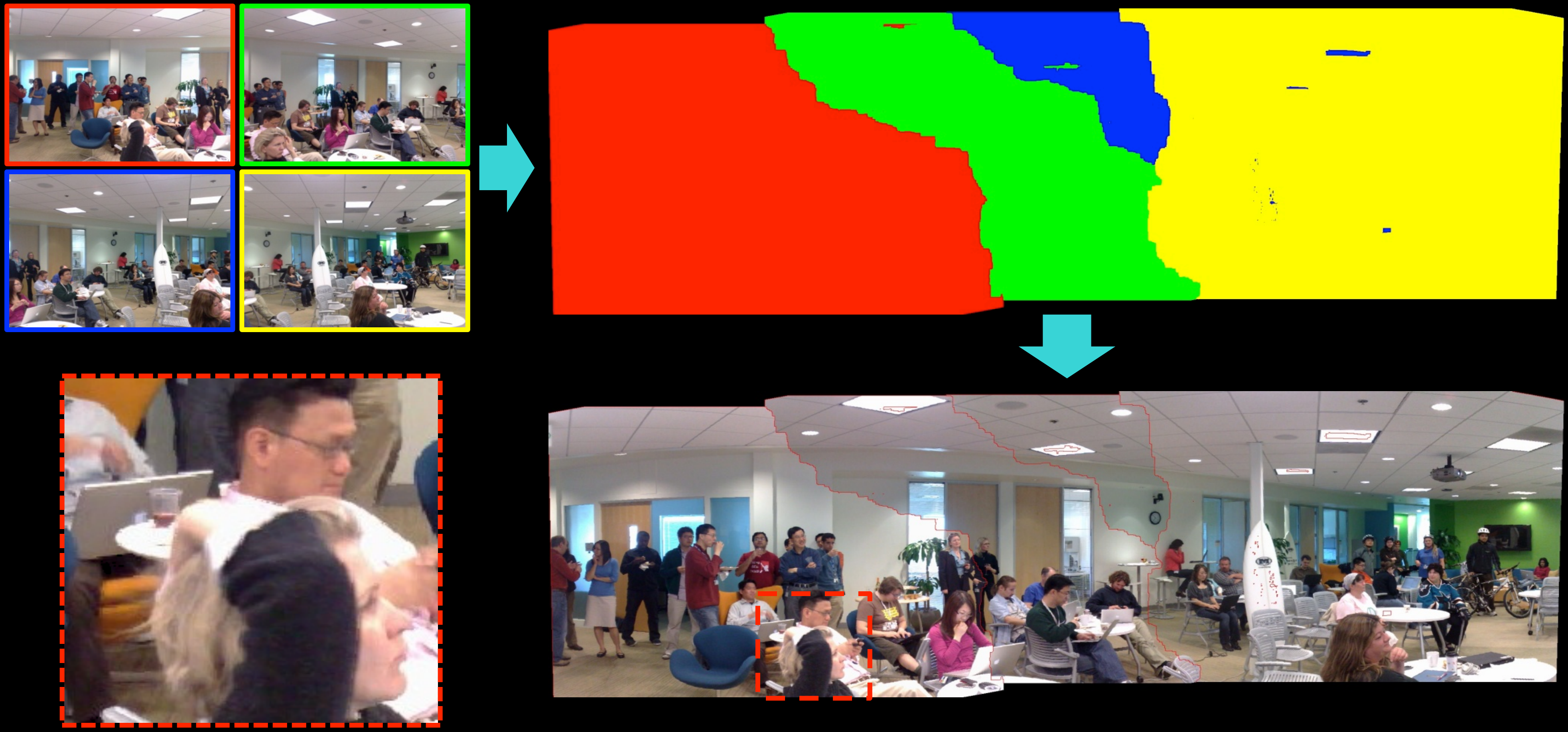
+

=

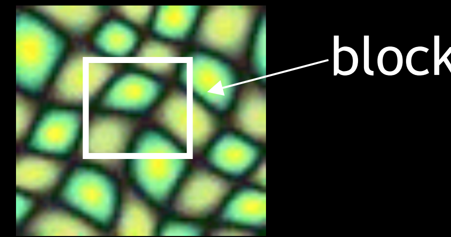
$$I_{blend} = \alpha I_{left} + (1-\alpha) I_{right}$$

Solution for ghosting: Image labeling

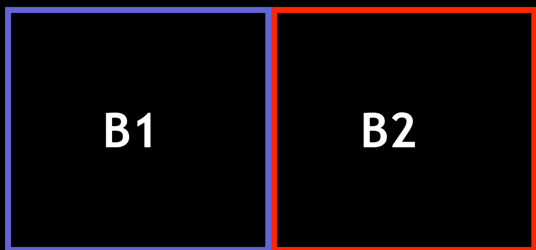
- Assign one input image to each output pixel
  - Optimal assignment can be found by graph cut [Agarwala et al. 2004]



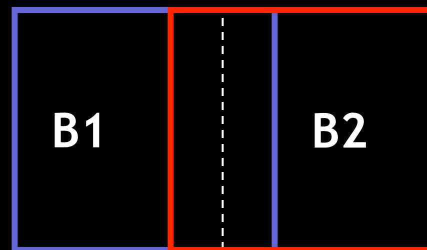
Faster solution with dynamic programming



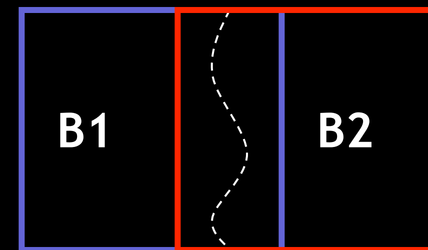
Input texture



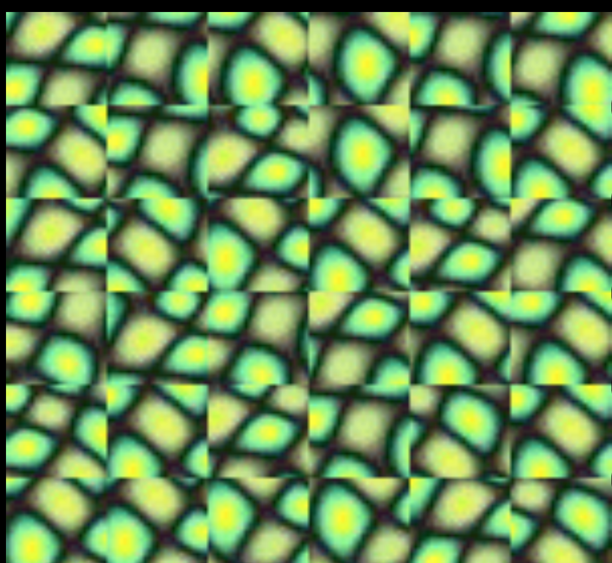
Random placement of blocks

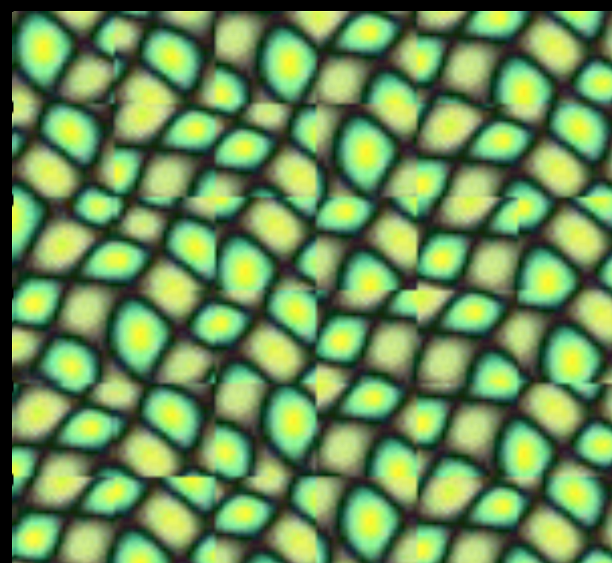


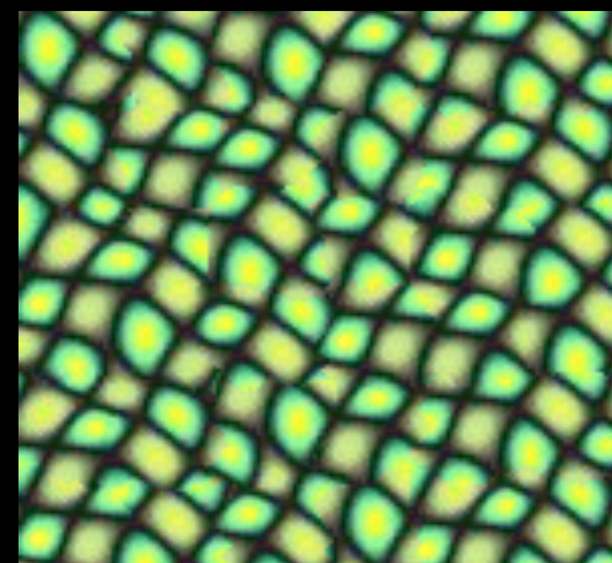
Neighboring blocks constrained by overlap



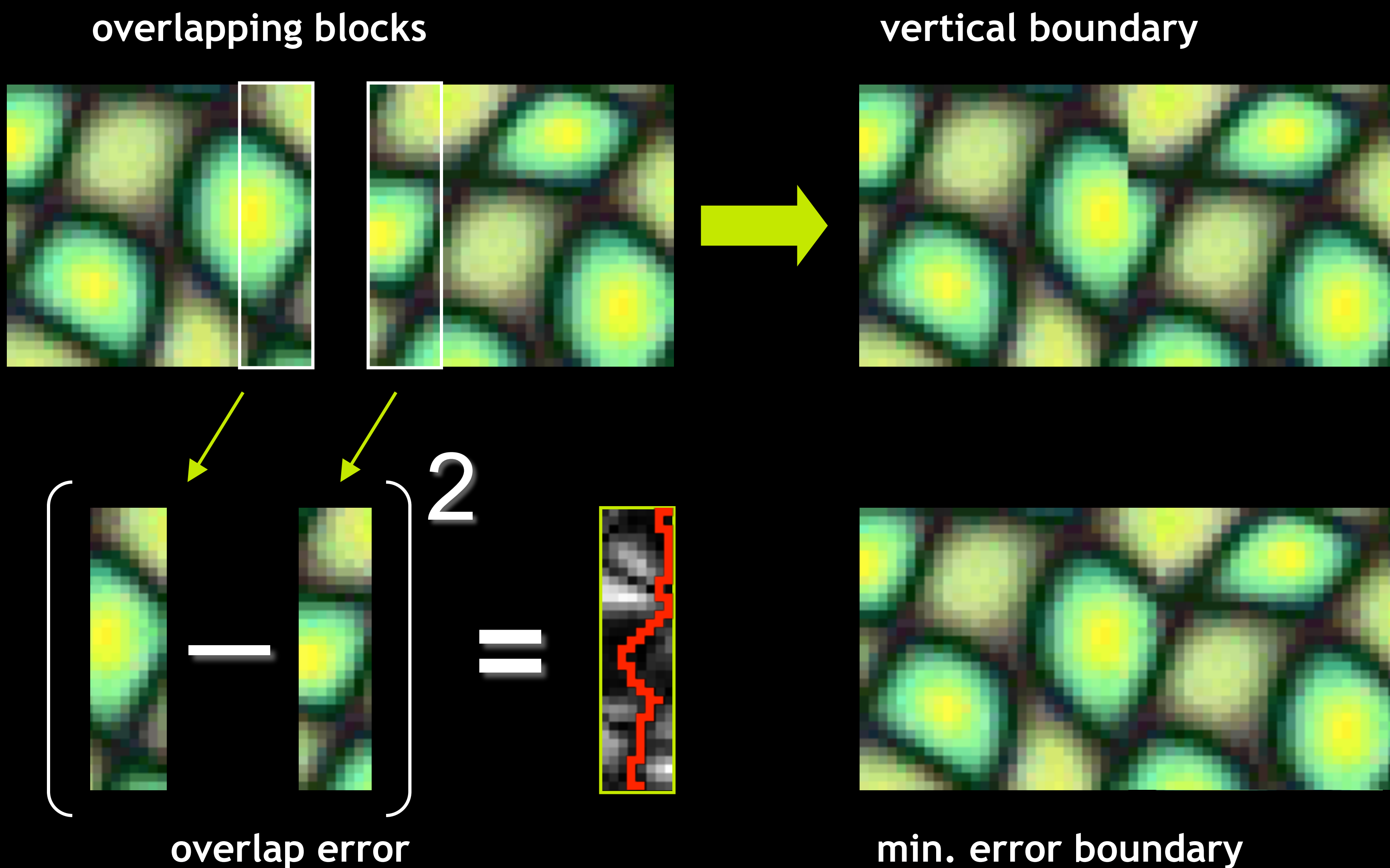
Minimal error boundary cut





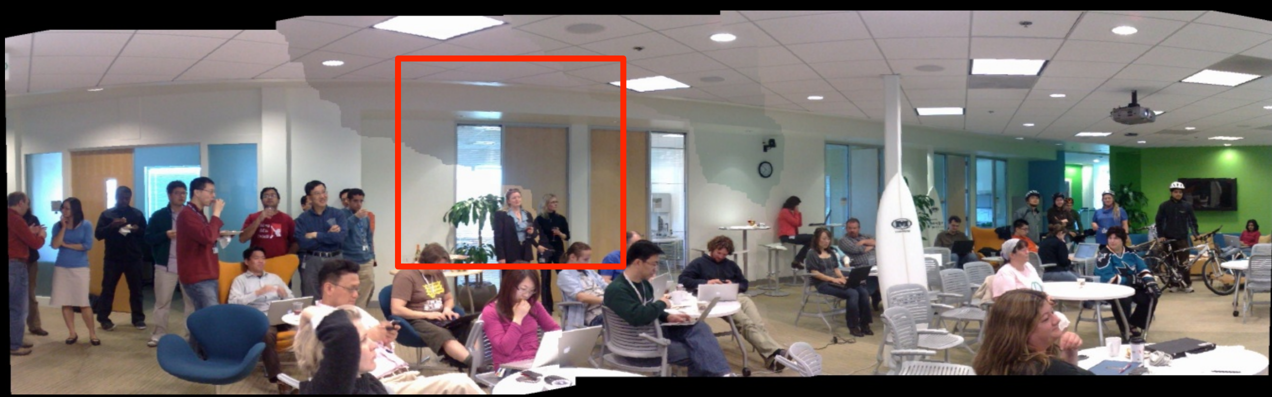


Minimal error boundary with DP



New artifacts

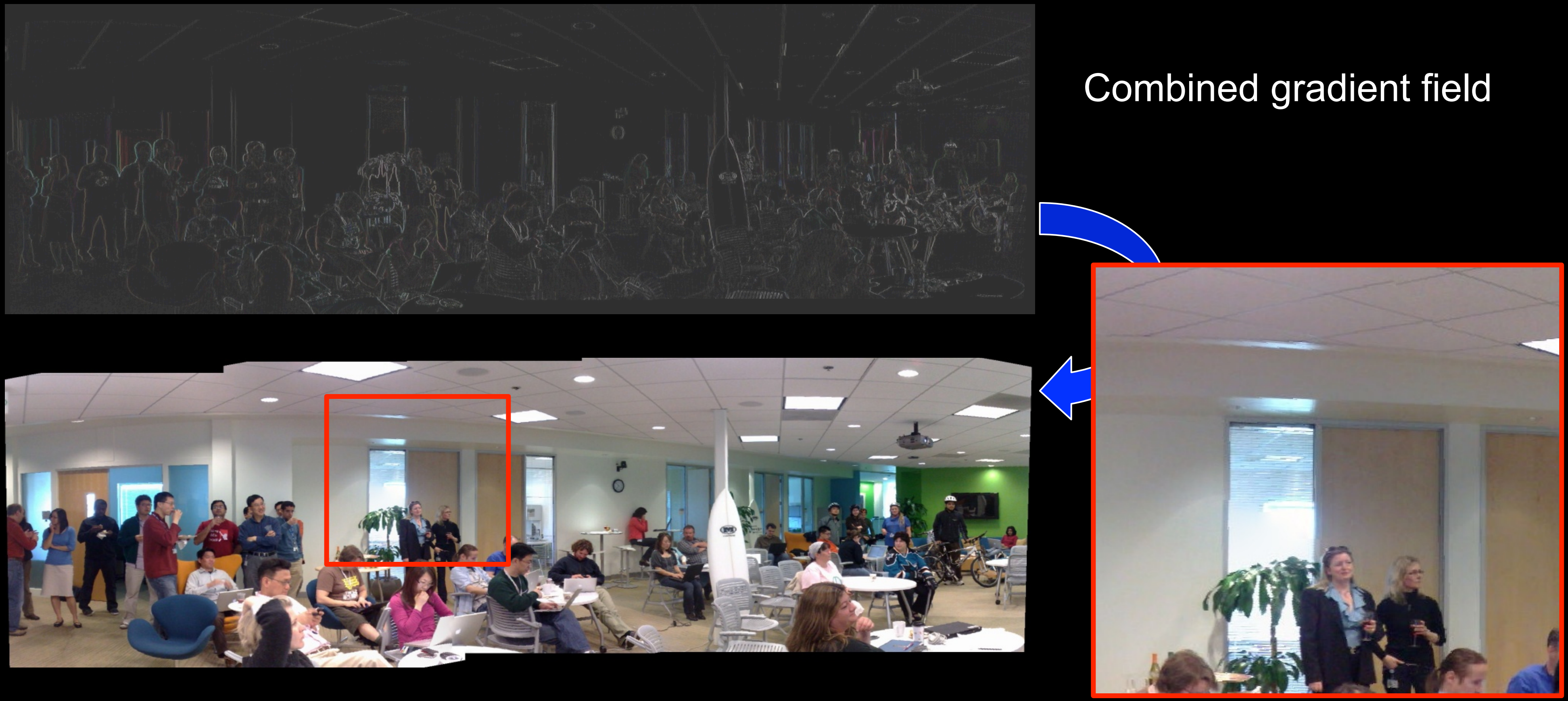
- Inconsistency between pixels from different input images
  - Different exposure/white balance settings
  - Photometric distortions (e.g., vignetting)



Solution: Poisson blending

- Copy the gradient field from the input image
- Reconstruct the final image by solving a Poisson equation

Combined gradient field

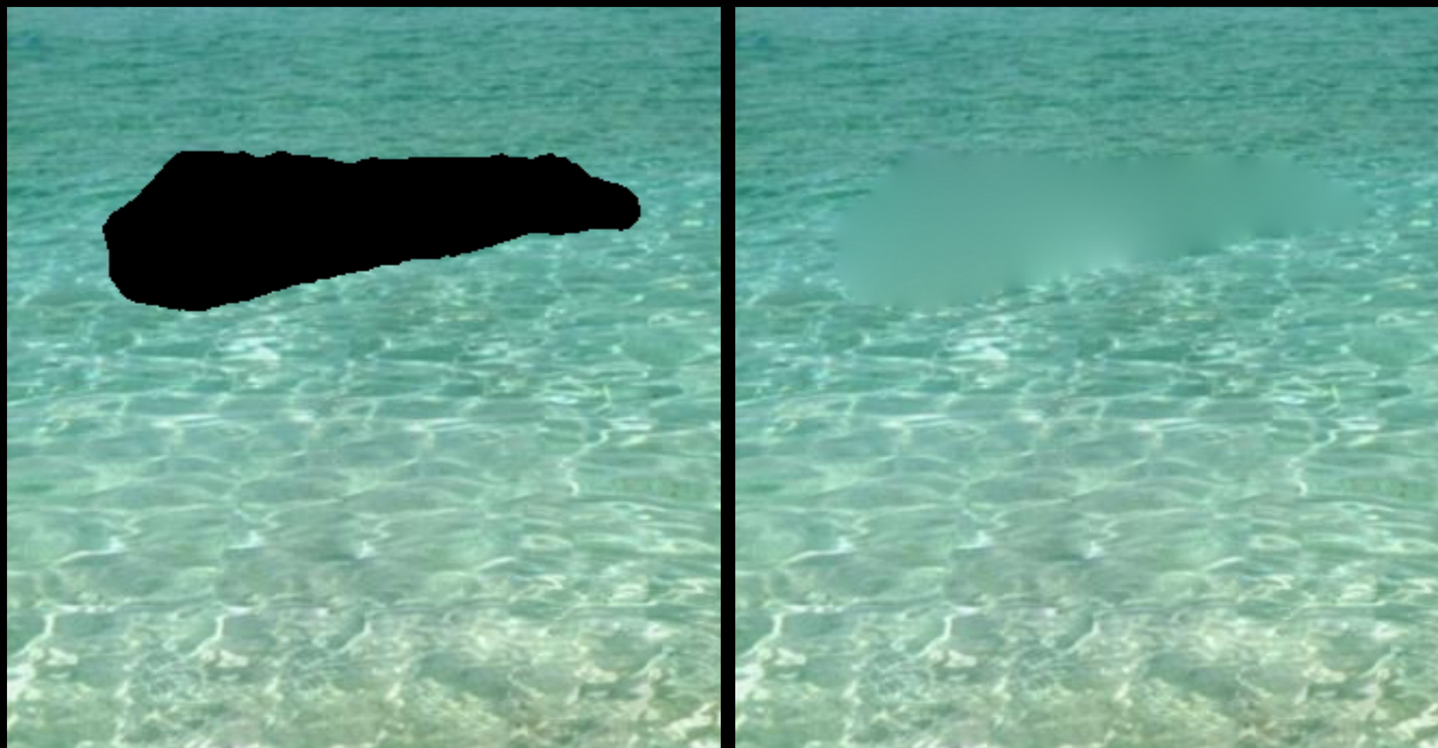


Problems with direct cloning

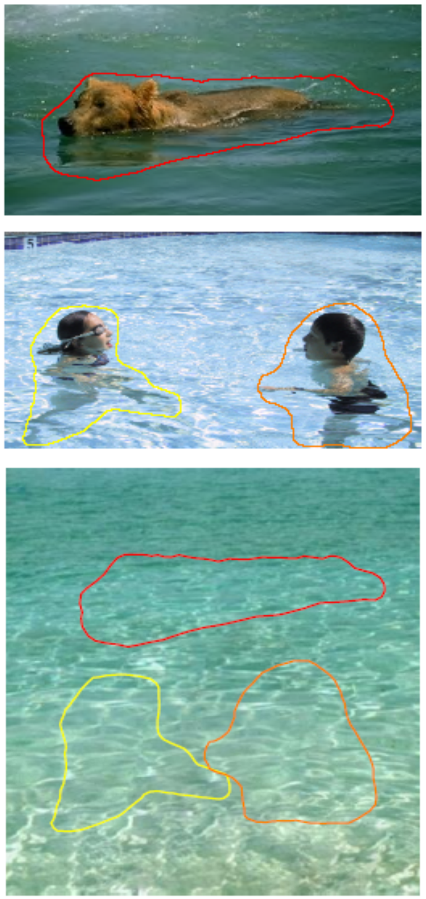
P. Pérez, M. Gangnet, A. Blake. Poisson image editing. SIGGRAPH 2003

http://www.irisa.fr/vista/Papers/2003_siggraph_perez.pdf

Membrane interpolation



Solution: clone gradient, integrate colors

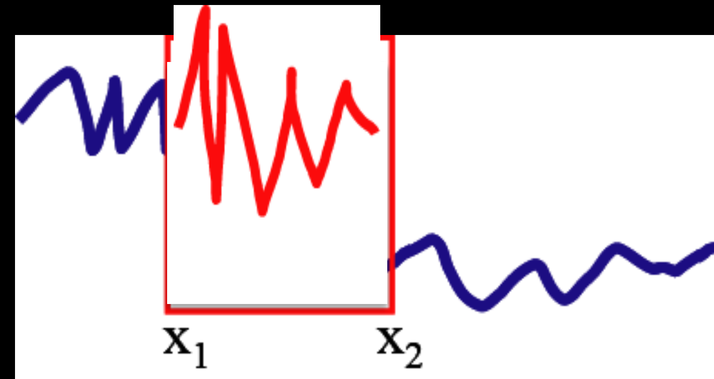


sources/destinations

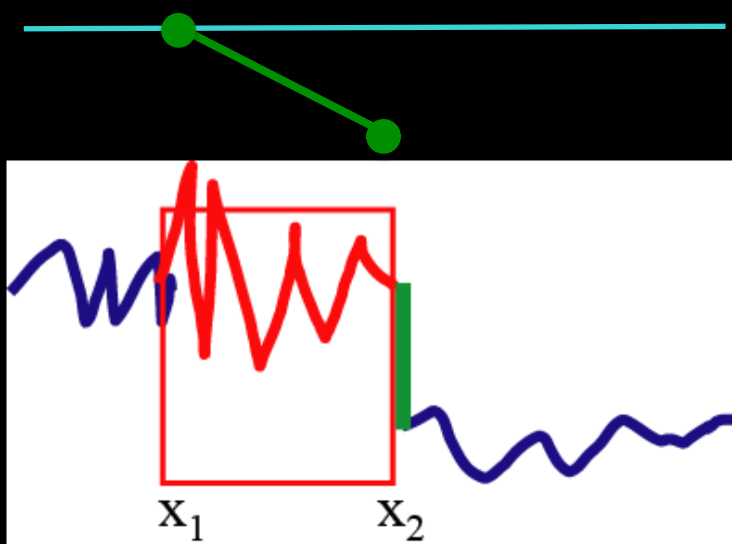
cloning

seamless cloning

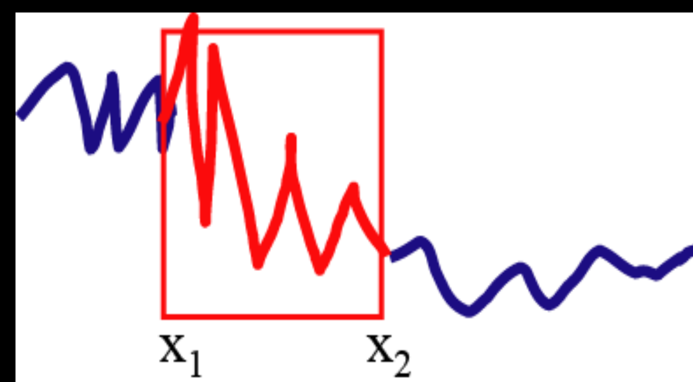
Copy the details

Seamlessly paste  onto 

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



Gradients didn't change much, and function is continuous



Coordinates for Instant Image Cloning SIGGRAPH 2009

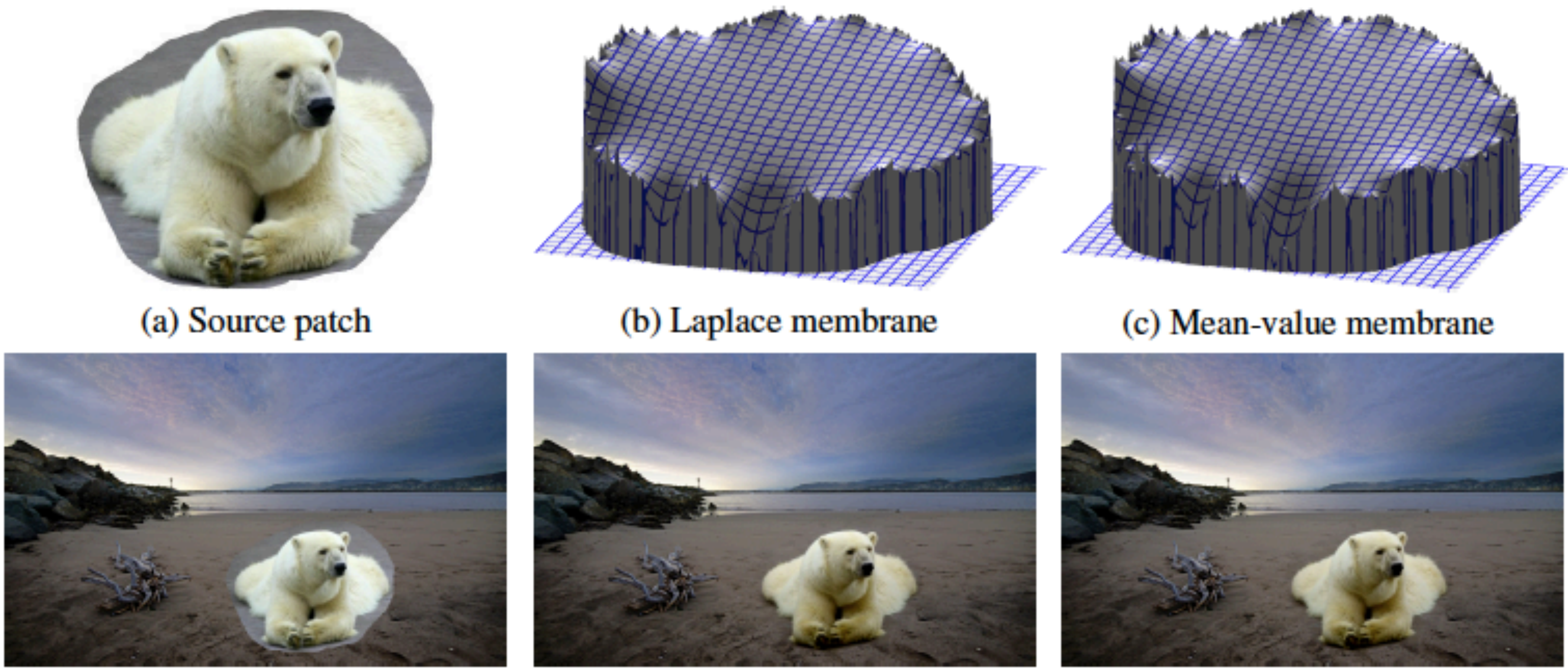
Zeev Farbman
Hebrew University

Gil Hoffer
Tel Aviv University

Yaron Lipman
Princeton University

Daniel Cohen-Or
Tel Aviv University

Dani Lischinski
Hebrew University



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

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

Figure 1: *Poisson cloning smoothly interpolates the error along the boundary of the source and the target regions across the entire cloned region (the resulting membrane is shown in (b)), yielding a seamless composite (e). A qualitatively similar membrane (c) may be achieved via transfinite interpolation, without solving a linear system. (f) Seamless cloning obtained instantly using the mean-value interpolant.*

Smooth interpolation over a triangulation

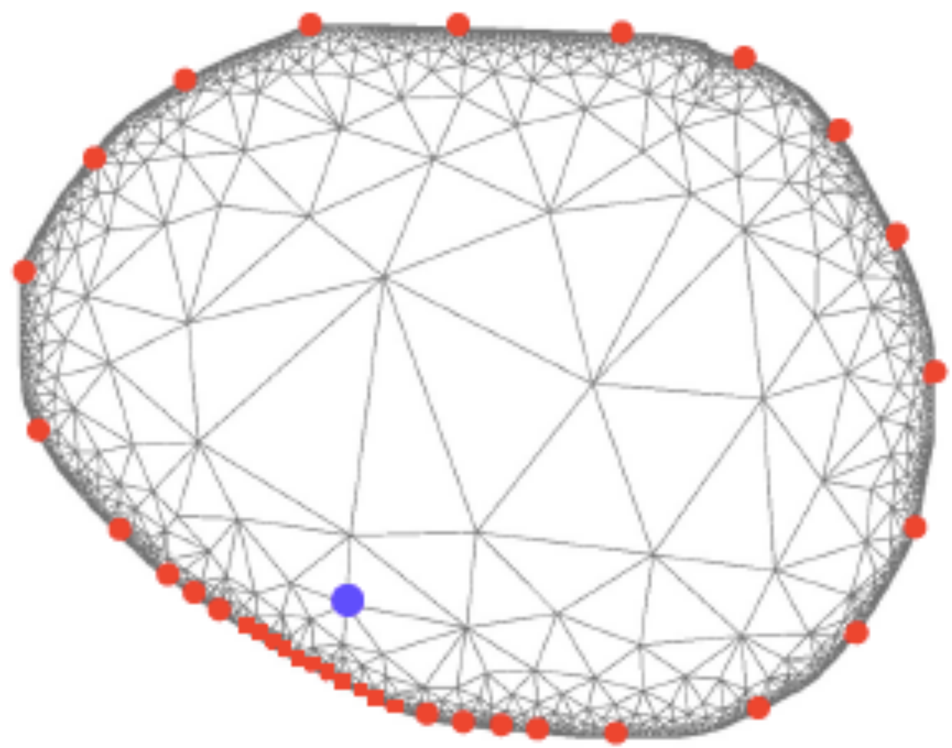


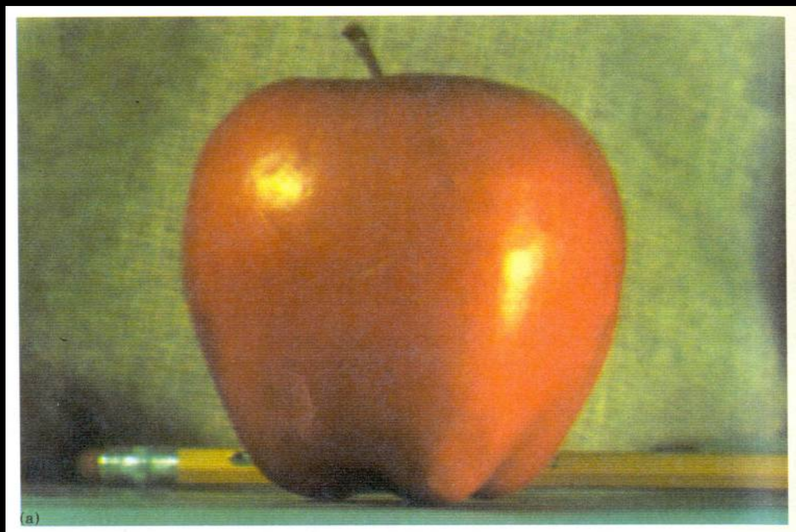
Figure 3: *An adaptive triangular mesh constructed over the region to be cloned. The red dots on the boundary show the positions of boundary vertices that were selected by adaptive hierarchical subsampling for the mesh vertex indicated in blue.*

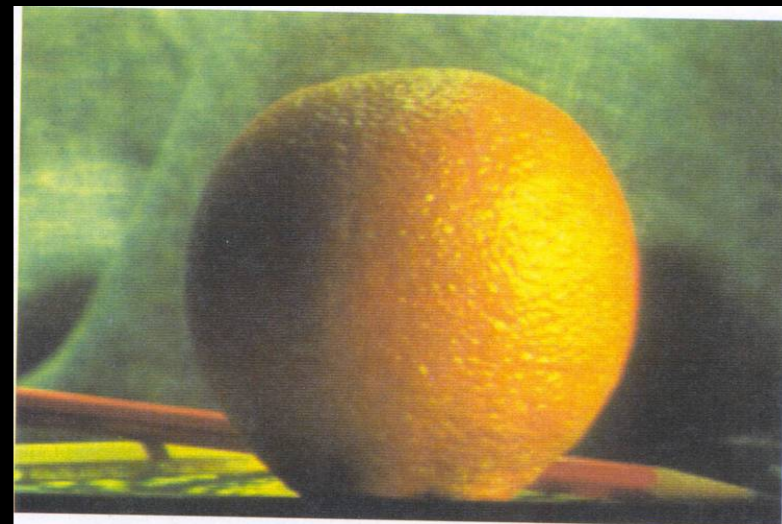
Alpha blending

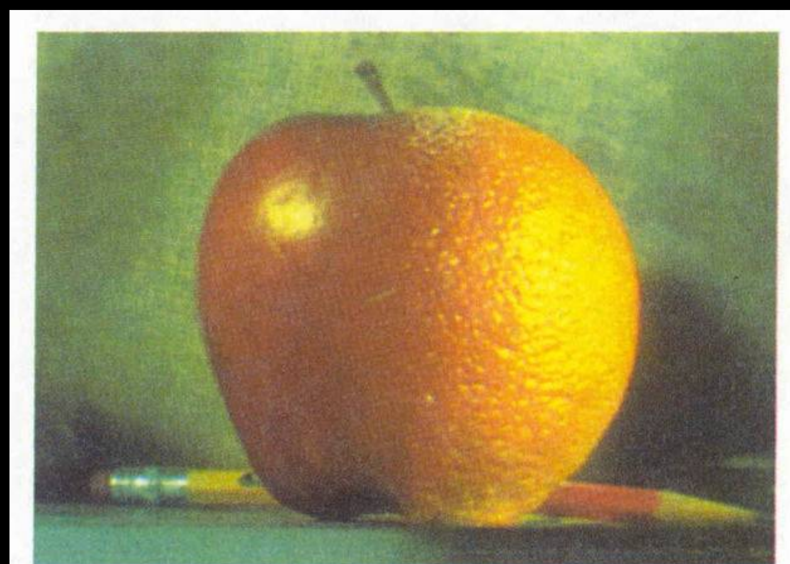
After labeling

Poisson blending

Pyramid Blending

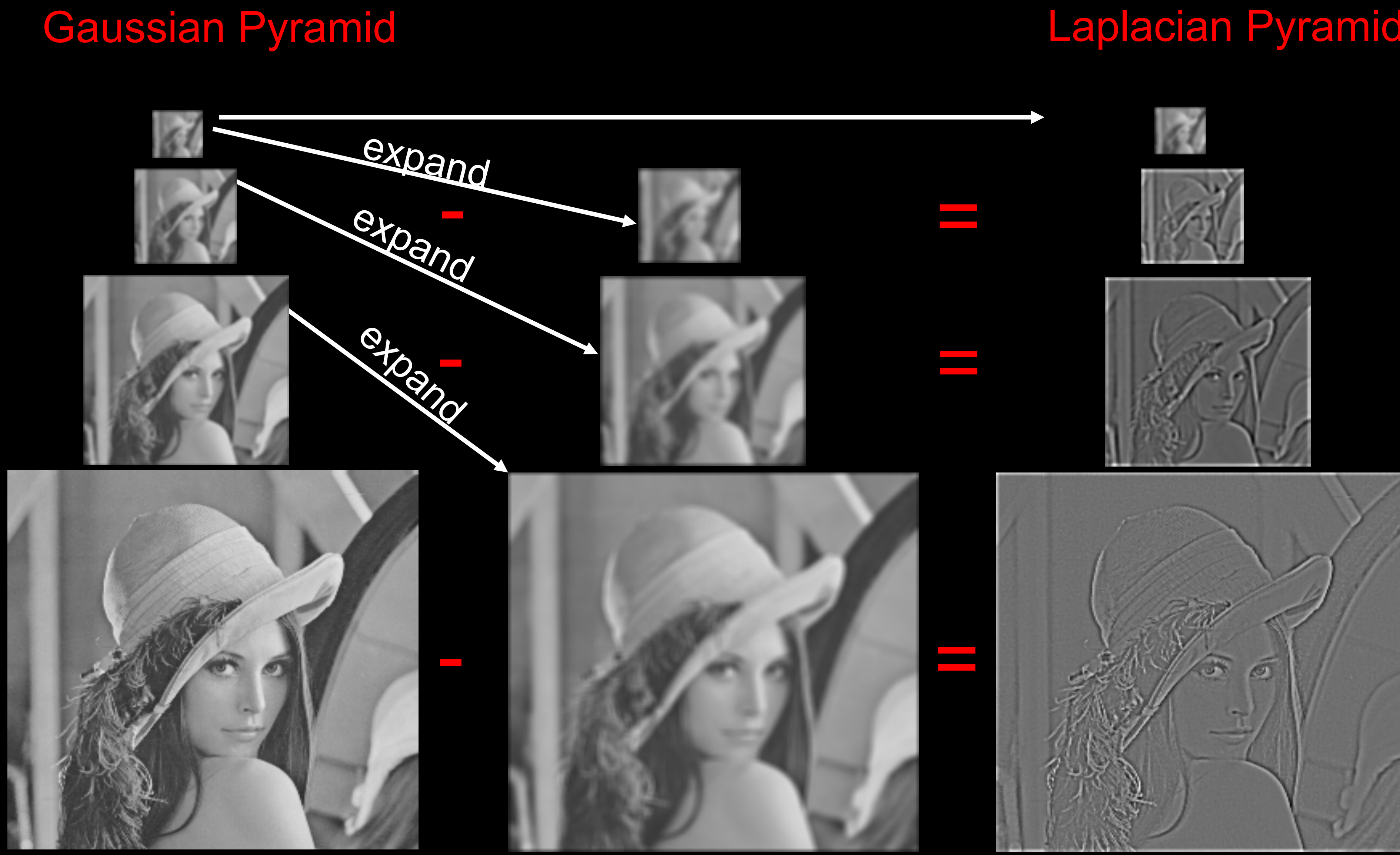






(l)

The Laplacian pyramid



Laplacian Pyramid: Blending

1. Build Laplacian pyramids *LA* and *LB* from images *A* and *B*

2. Build a Gaussian pyramid *GM* from selection mask M

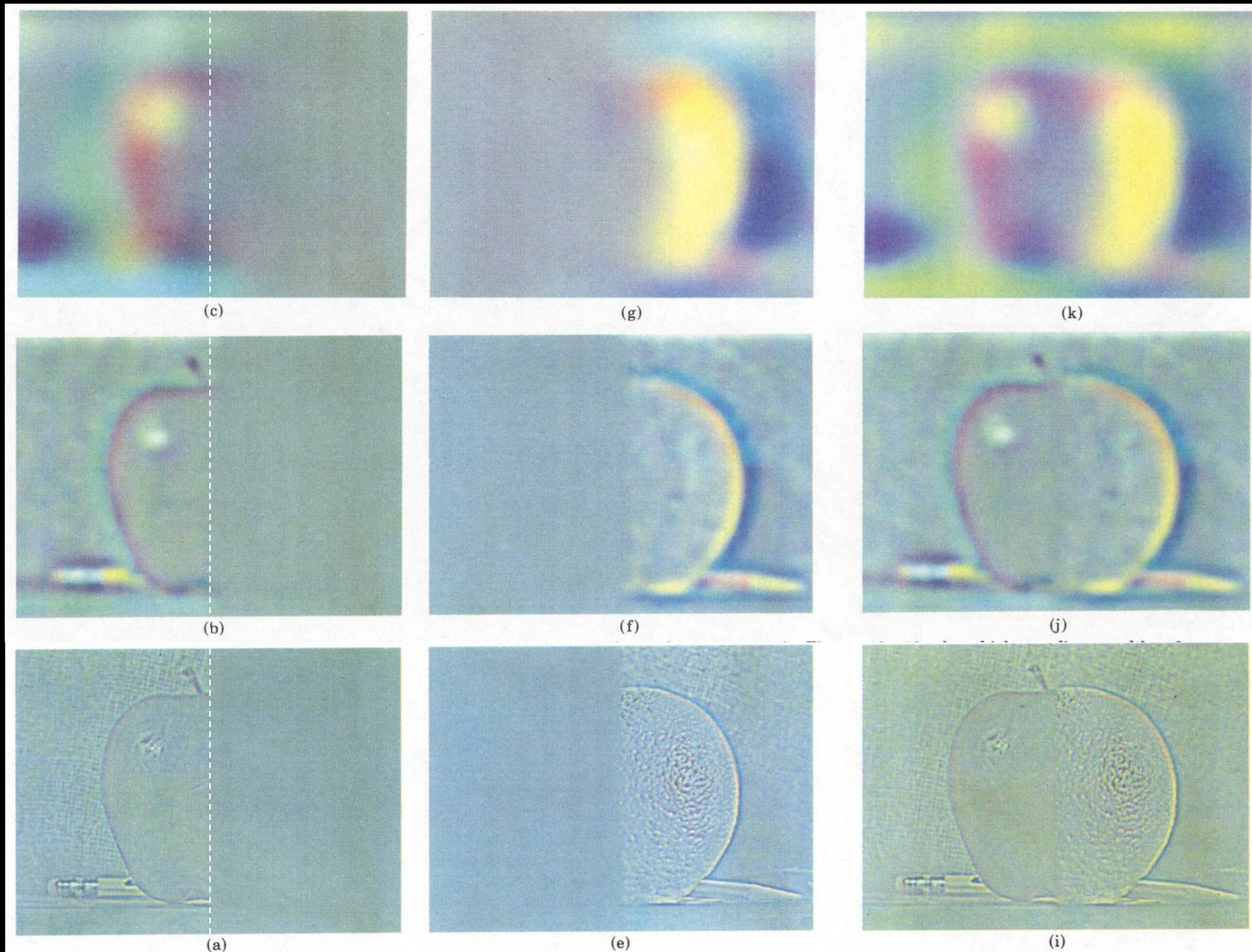
3. Form a combined pyramid *LS* from *LA* and *LB* using nodes of *GR* as weights:
   - $LS = GM * LA + (1-GM) * LB$

4. Collapse the *LS* pyramid to get the final blended image

Laplacian level 4

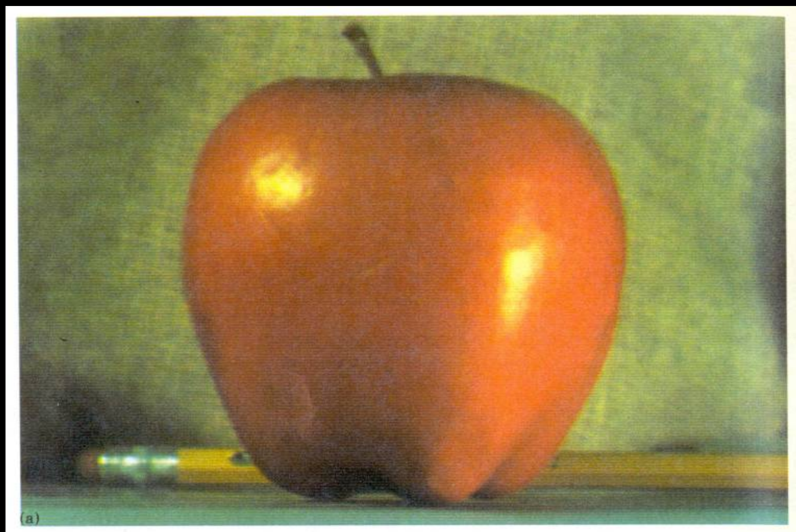
Laplacian level 2

Laplacian level 0

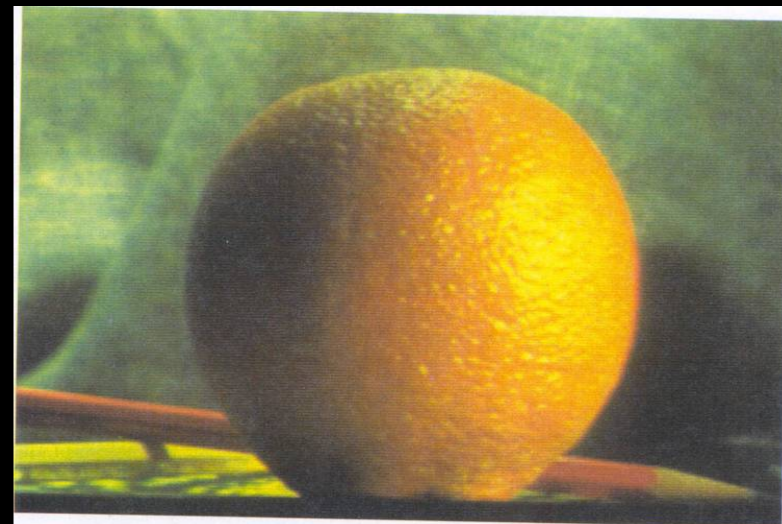


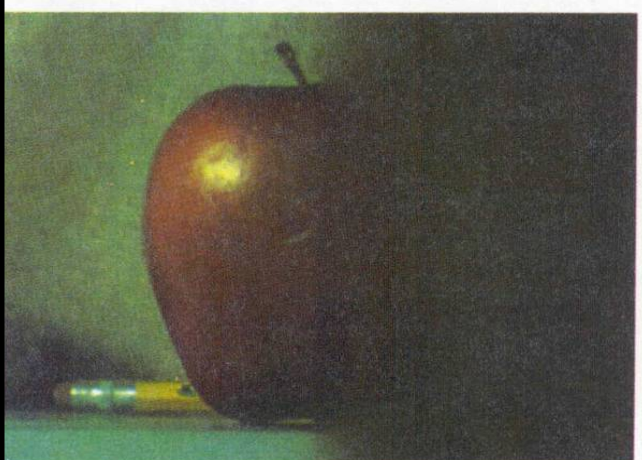
left pyramid | right pyramid | blended pyramid

Pyramid Blending

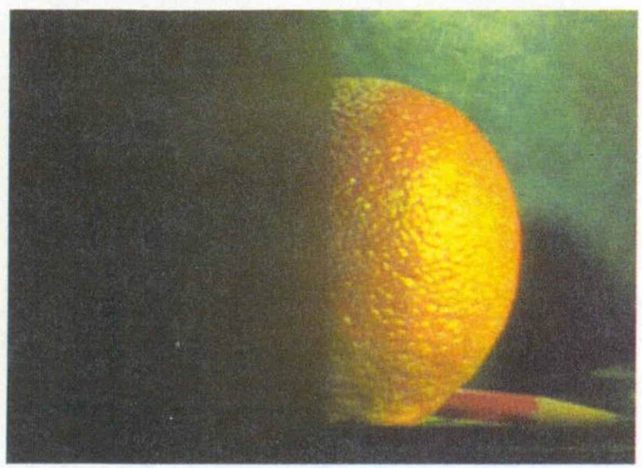


(a)

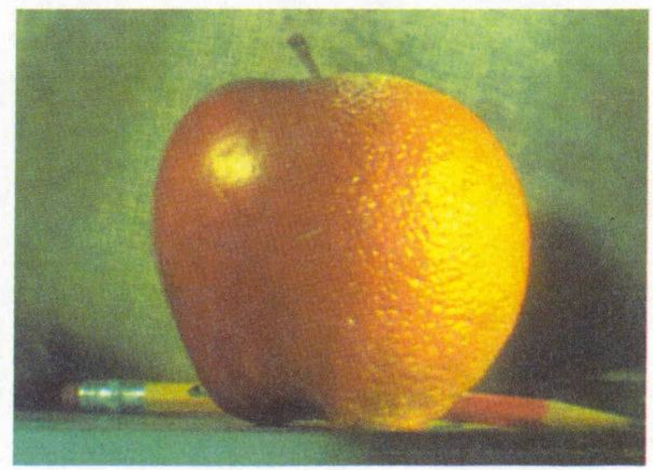




(d)

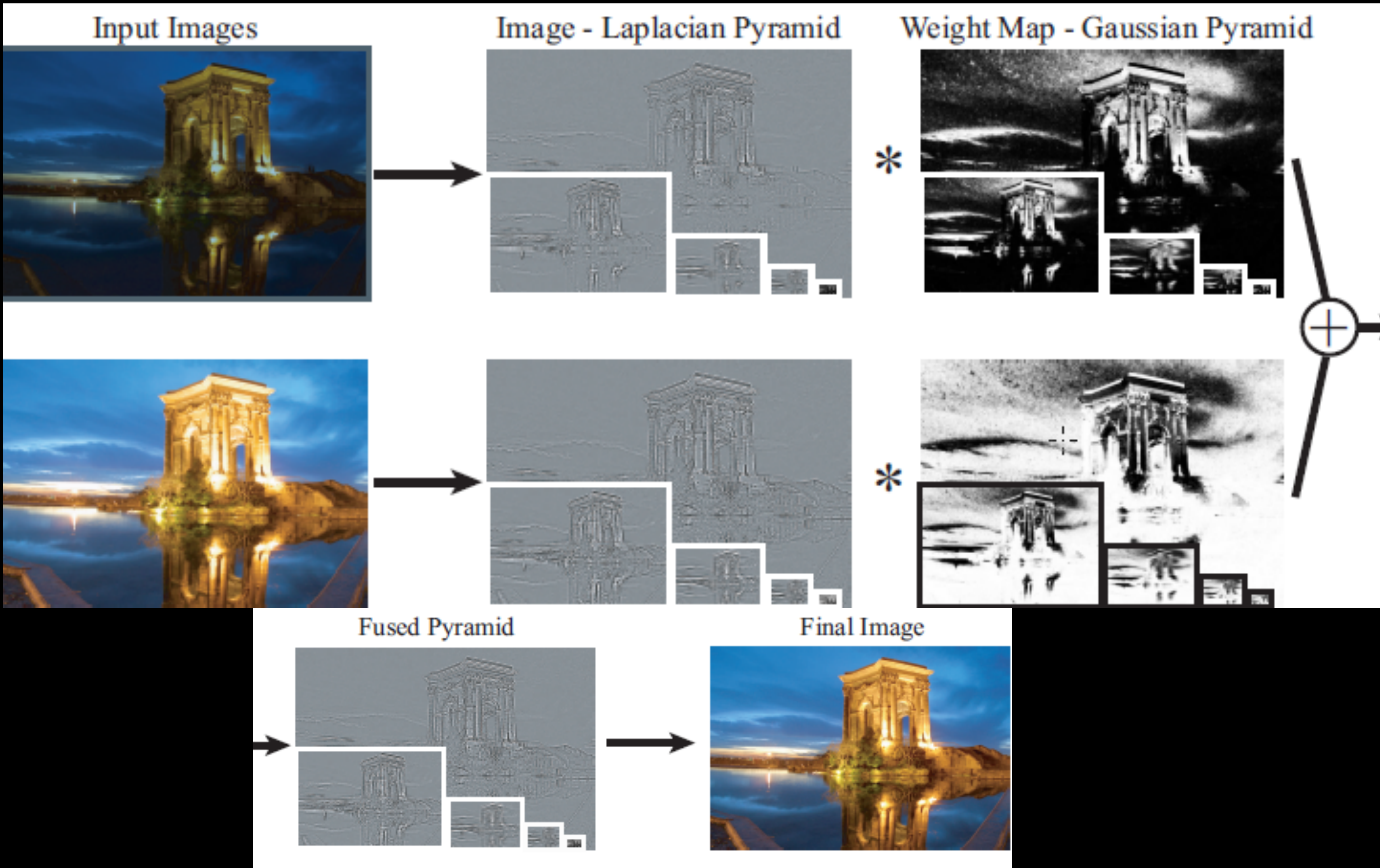


(h)



(l)

Multi-resolution fusion



Simplification: Two-band Blending

- Brown & Lowe, 2003
  - Only use two bands: high freq. and low freq.
  - Blends low freq. smoothly

2-band Blending

Low frequency (λ > 2 pixels)

High frequency (λ < 2 pixels)

Linear Blending

2-band Blending

Additional reading

- **Image Alignment and Stitching: A tutorial**
  - Richard Szeliski
  - Foundations and Trends in Computer Graphics and Vision
- **Computer Vision: Algorithms and Applications**
  - Richard Szeliski
  - http://szeliski.org/Book/
  - Chapters 4, 6, 9