Computational Photography

* Study the basics of computation and its impact on the entire workflow of photography, from capturing, manipulating and collaborating on, and sharing photographs.

Georgia Tech College of Computing

Digital Images: Into the Frequency Domain

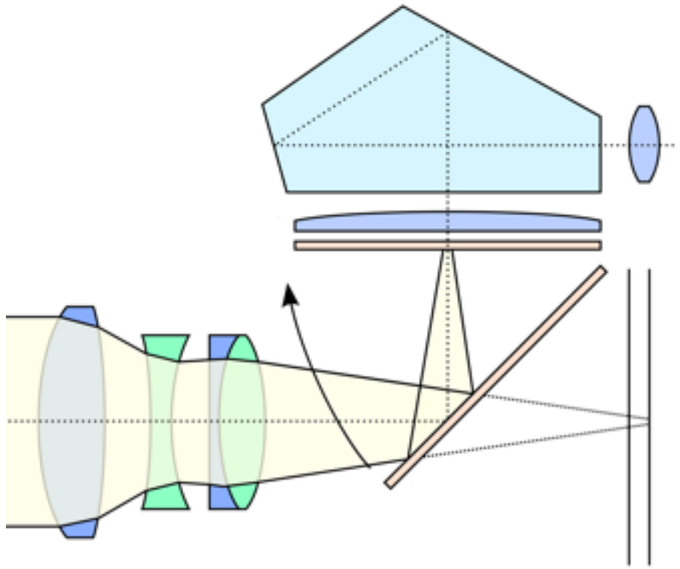
* Images are samples of light information stored in array of pixels.

Georgia Tech College of Computing

Lesson Objectives

1. Using sines and cosines to reconstruct a signal
2. The Fourier Transform
3. Frequency Domains for a Signal
4. Three properties of Convolution relating to Fourier Transform

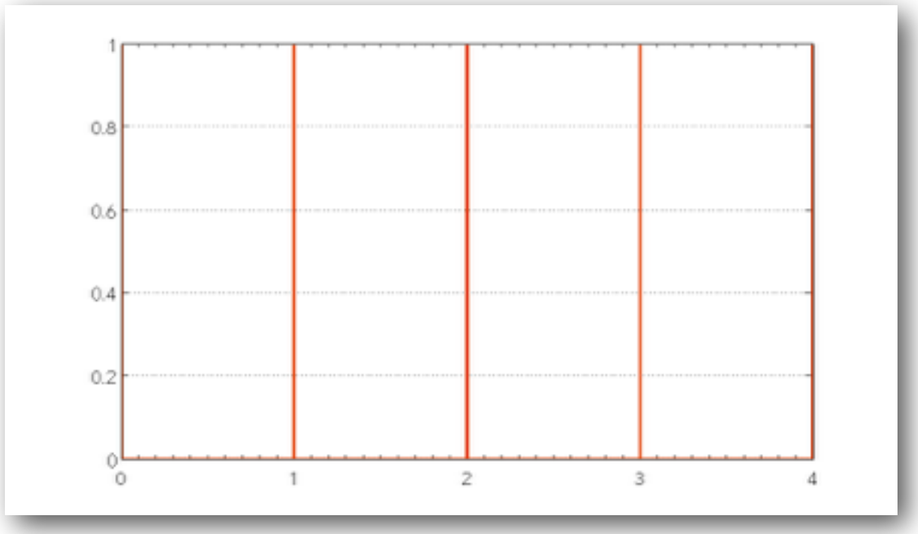
Recall: Images and Camera



* Rays of Light go through a Camera / Optics / Lens
* Aperture, Shutter, and Film Sensitivity
* Sensor to Convert Light to Digital Information
* Process the Image

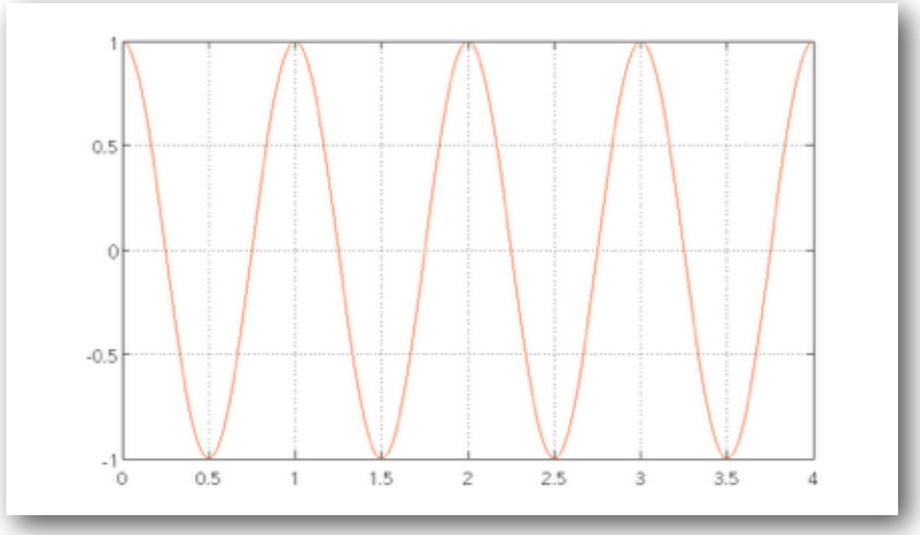
Reconstructing a Signal

Target Signal: $f^T(t)$



Repeating Impulse Function

f_1



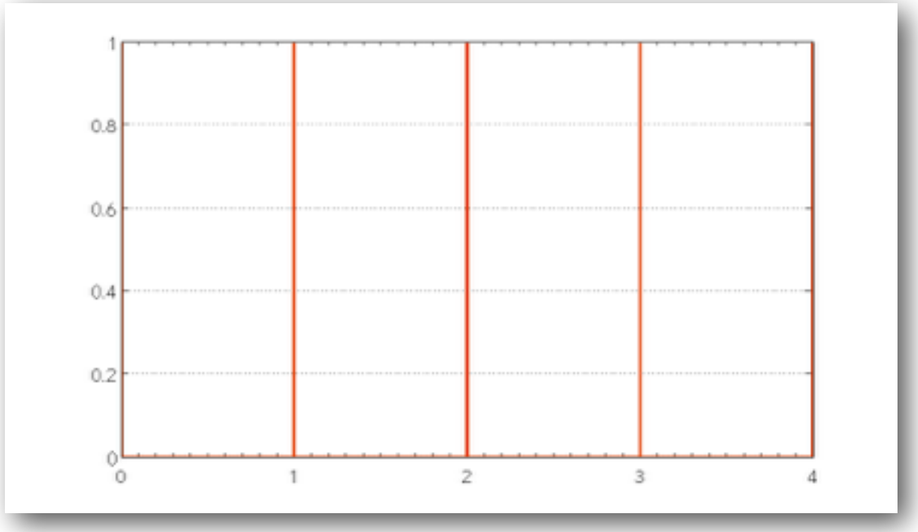
* Basic building block

* A: Amplitude, w: frequency

$$f(t) = A\cos(n\omega t)$$

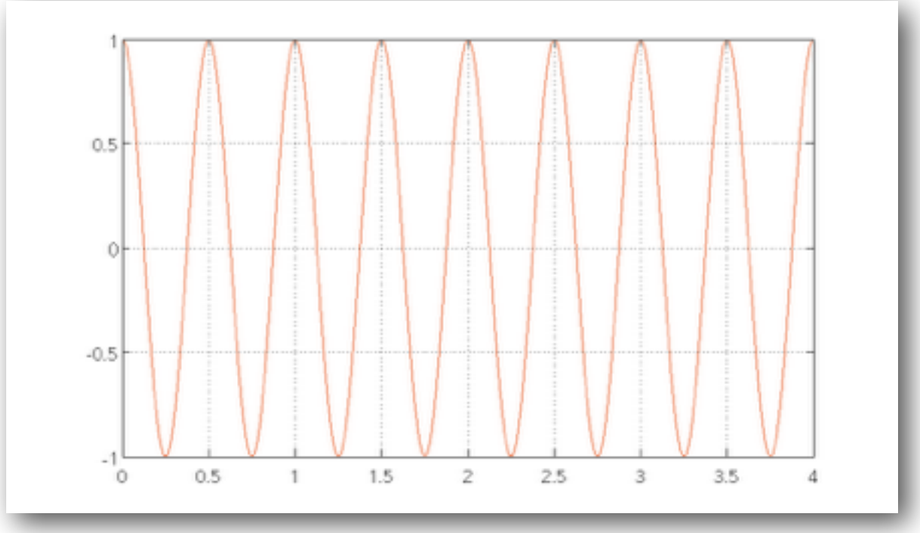
Reconstructing a Signal

Target Signal: $f^T(t)$



Repeating Impulse Function

f_2

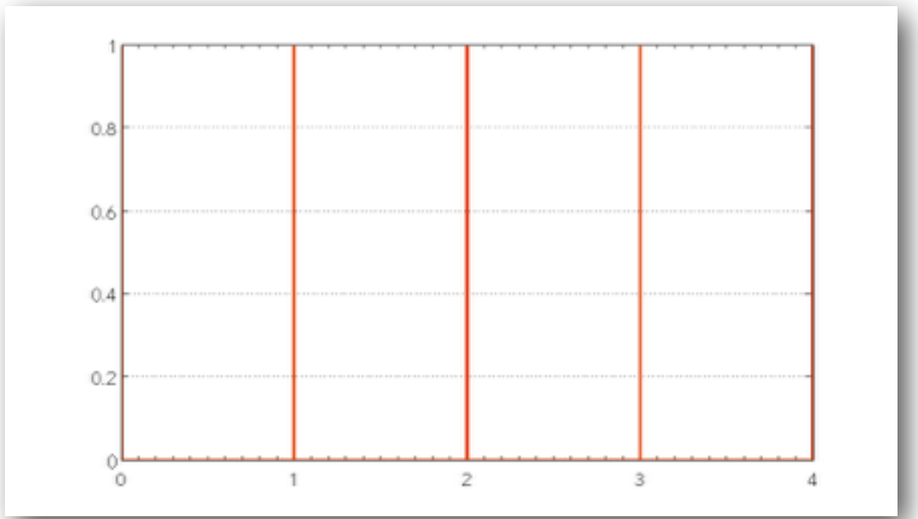


* Basic building block
  * A: Amplitude, w: frequency

$$f(t) = A\cos(n\omega t)$$

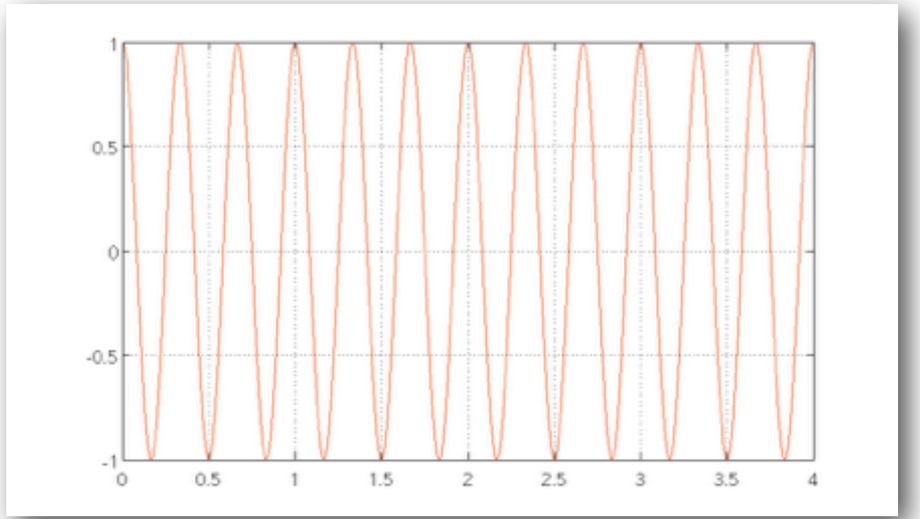
Reconstructing a Signal

Target Signal: $f^T(t)$



Repeating Impulse Function

f_3

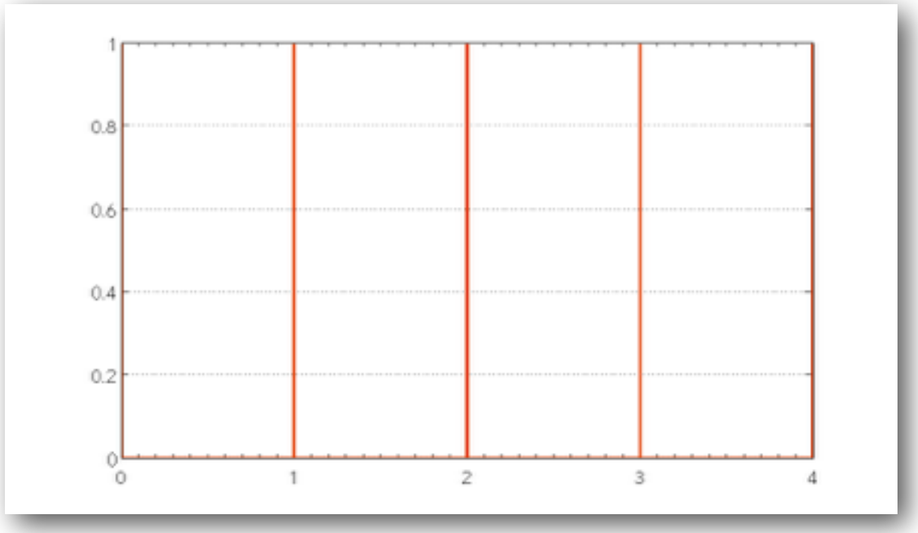


* Basic building block
  * A: Amplitude, w: frequency

$$f(t) = A\cos(n\omega t)$$

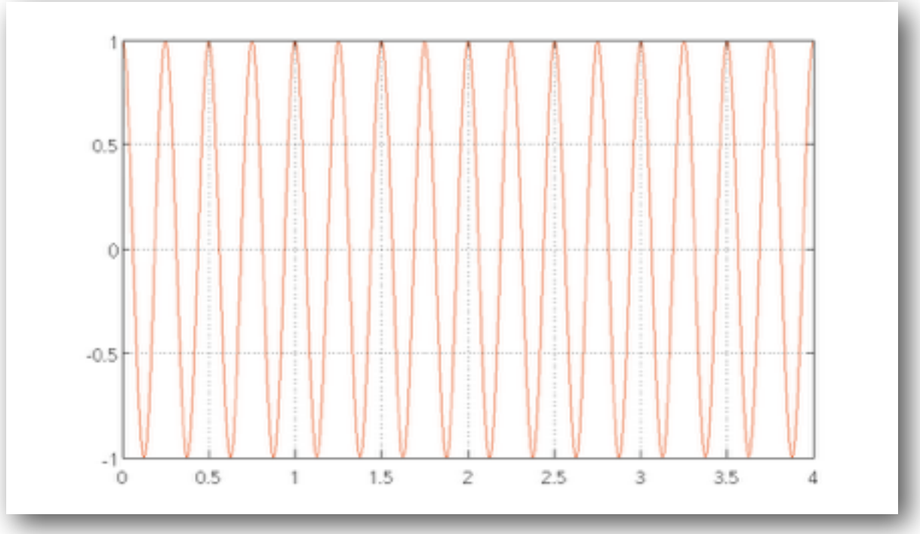
Reconstructing a Signal

Target Signal: $f^T(t)$



Repeating Impulse Function

f_4

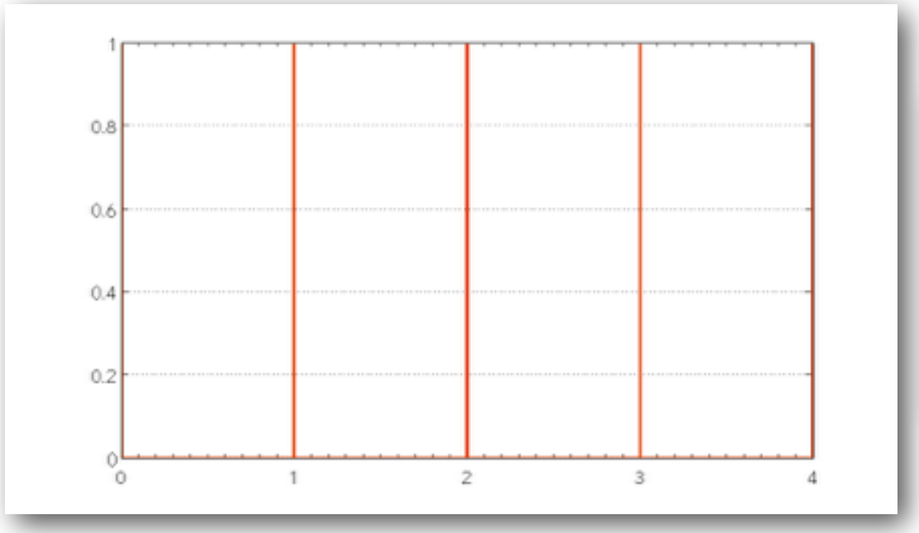


* Basic building block
* A: Amplitude, w: frequency

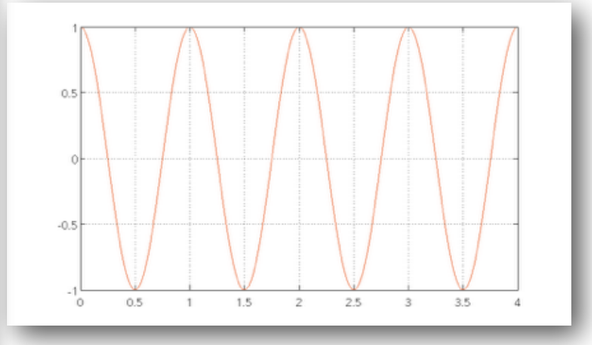
$$f(t) = A\cos(n\omega t)$$

Reconstructing a Signal

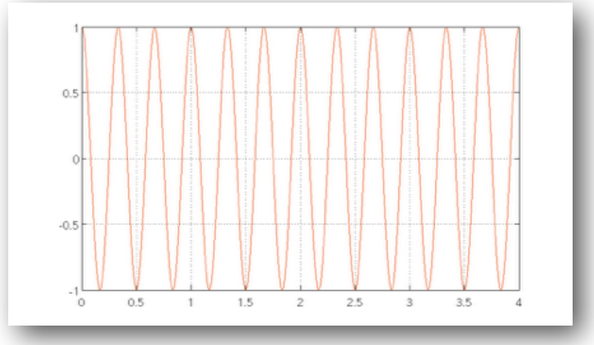
Target Signal: $f^T(t)$



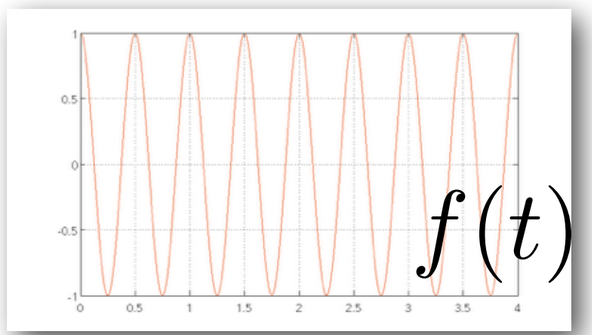
Repeating Impulse Function



f_1

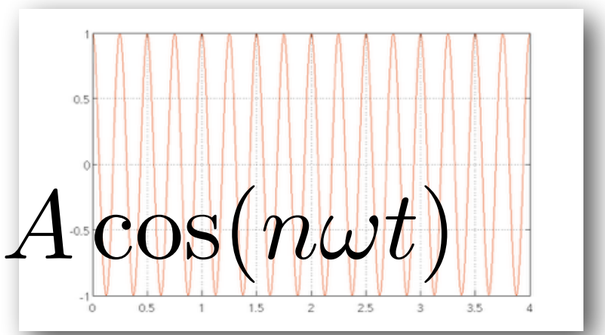


f_3



f_2

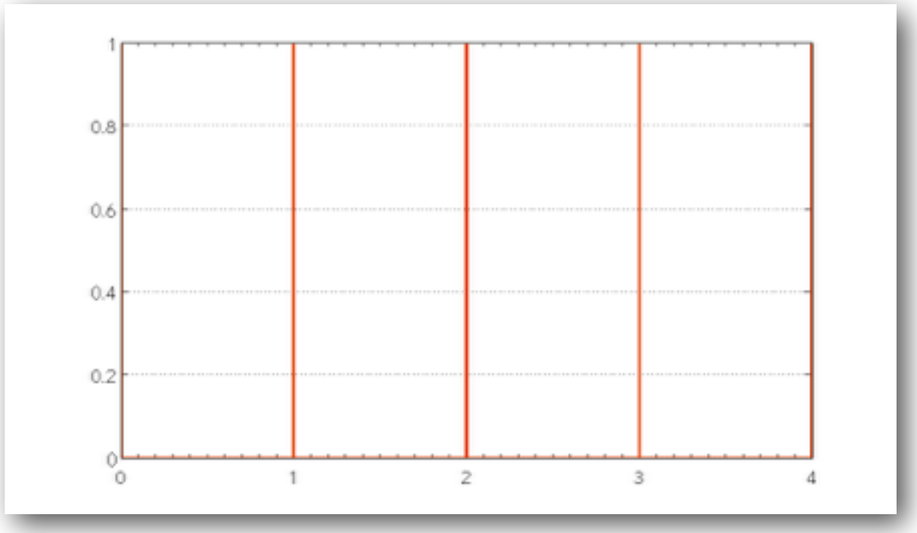
$$f(t) = A\cos(n\omega t)$$



f_4

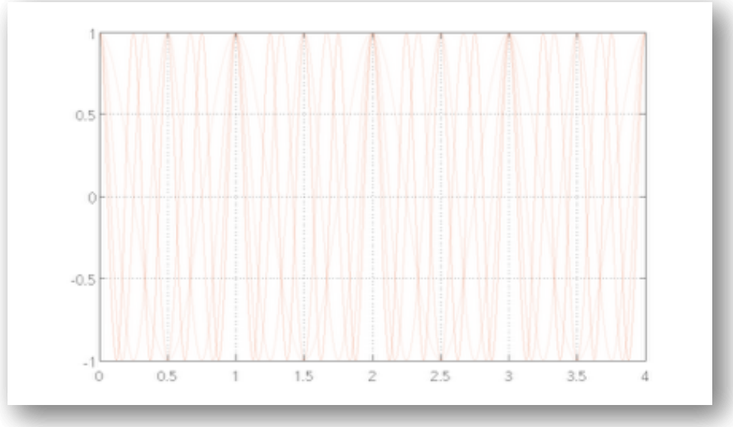
Sum of Cosines

Target Signal: $f^T(t)$



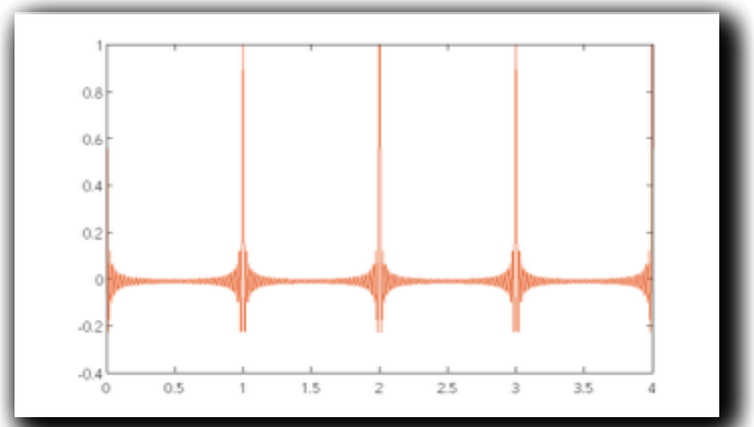
Repeating Impulse Function

f_1 f_2 f_3 f_4



N=10

N=50



$f_1 + f_2 + f_3 + f_4$

* Adding Cosines (N of them)

$$f(t) = A\cos(n\omega t)$$

$$f^T(t) = \sum_{n=1}^{N} A\cos(n\omega t)$$

A Fourier Transform

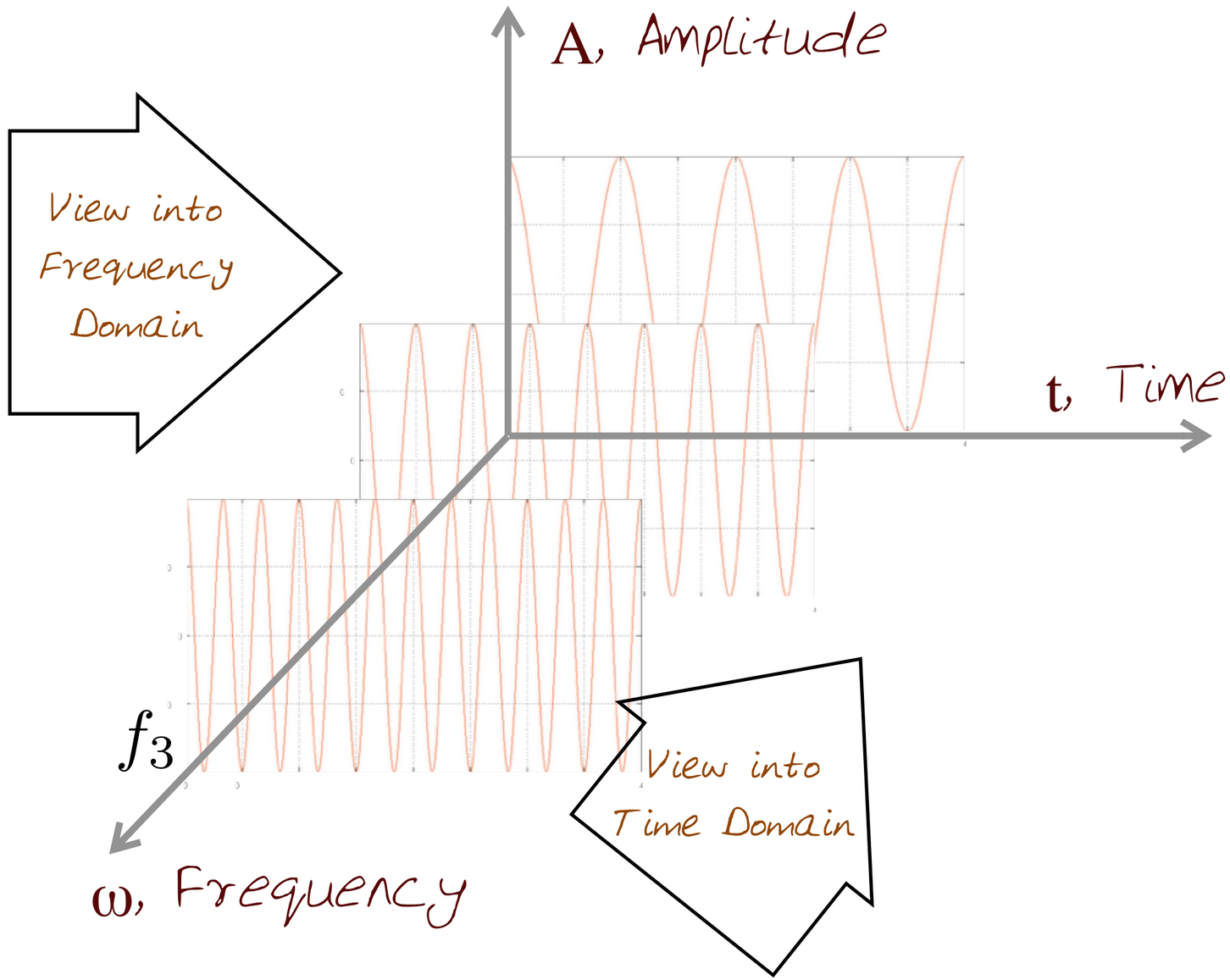
Jean Baptiste Joseph Fourier (1768-1830)

* Periodic function ⇒ a weighted sum of sines and cosines of different frequencies
* Transforms $\mathbf{f(t)}$, into a $\mathbf{F(\omega)}$
* Frequency spectrum of the function $\mathbf{f}$
* A reversible operation
* For every ω from 0 to ∞ (infinity) $\mathbf{F(\omega)}$ holds the Amplitude $\mathbf{A}$ and phase $\mathbf{\phi}$ of a sine function

$$F(\omega) = A\cos(\omega t + \phi)$$

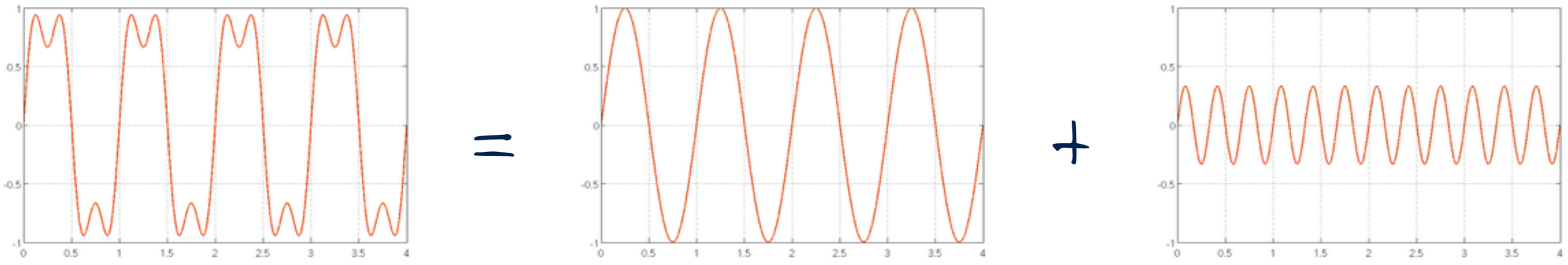
http://en.wikipedia.org/wiki/File:Fourier2.jpg

Frequency Domain of a Signal

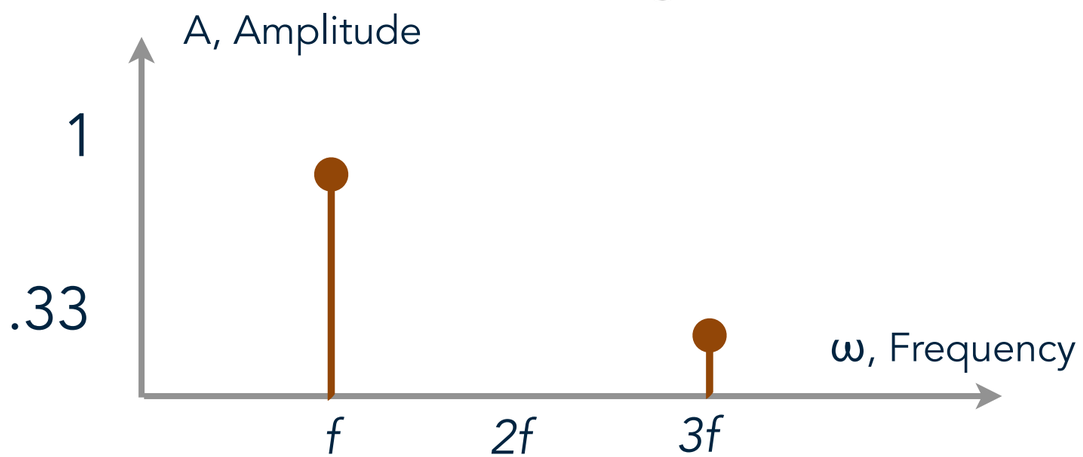


* How many N?
* What does each control?
* Coarse vs. fine structure

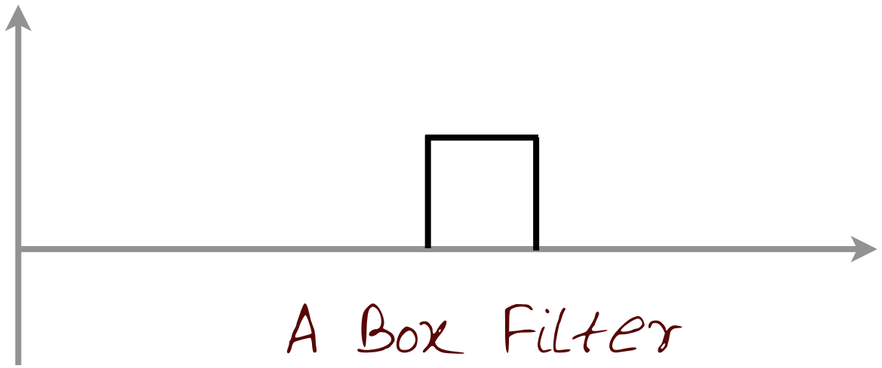
Time, Frequency, and Frequency Spectra

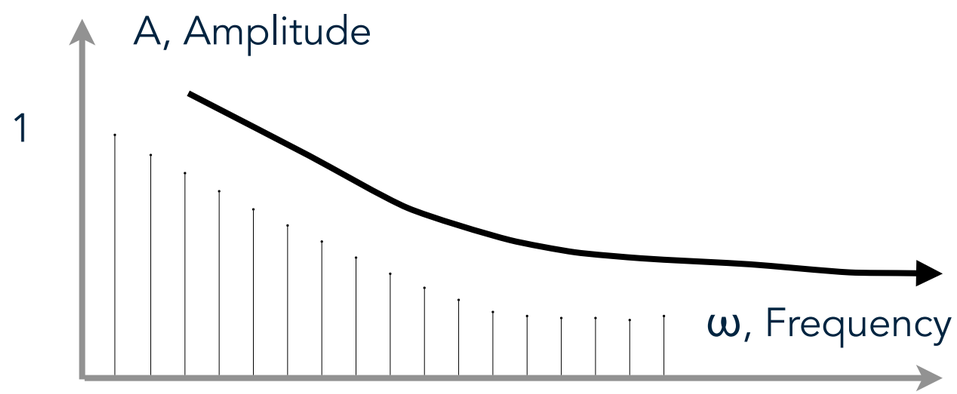


$$g(t) = \sin(2p\omega t) + \frac{1}{3}\sin(2p(3\omega)t)$$



Frequency Spectra





$$A \sum_{k=1}^{\infty} \frac{1}{k} \sin(2\pi k t)$$

Convolution Theorem and the Fourier Transform

* **Fourier transform** of a convolution of two functions = product of their Fourier transforms

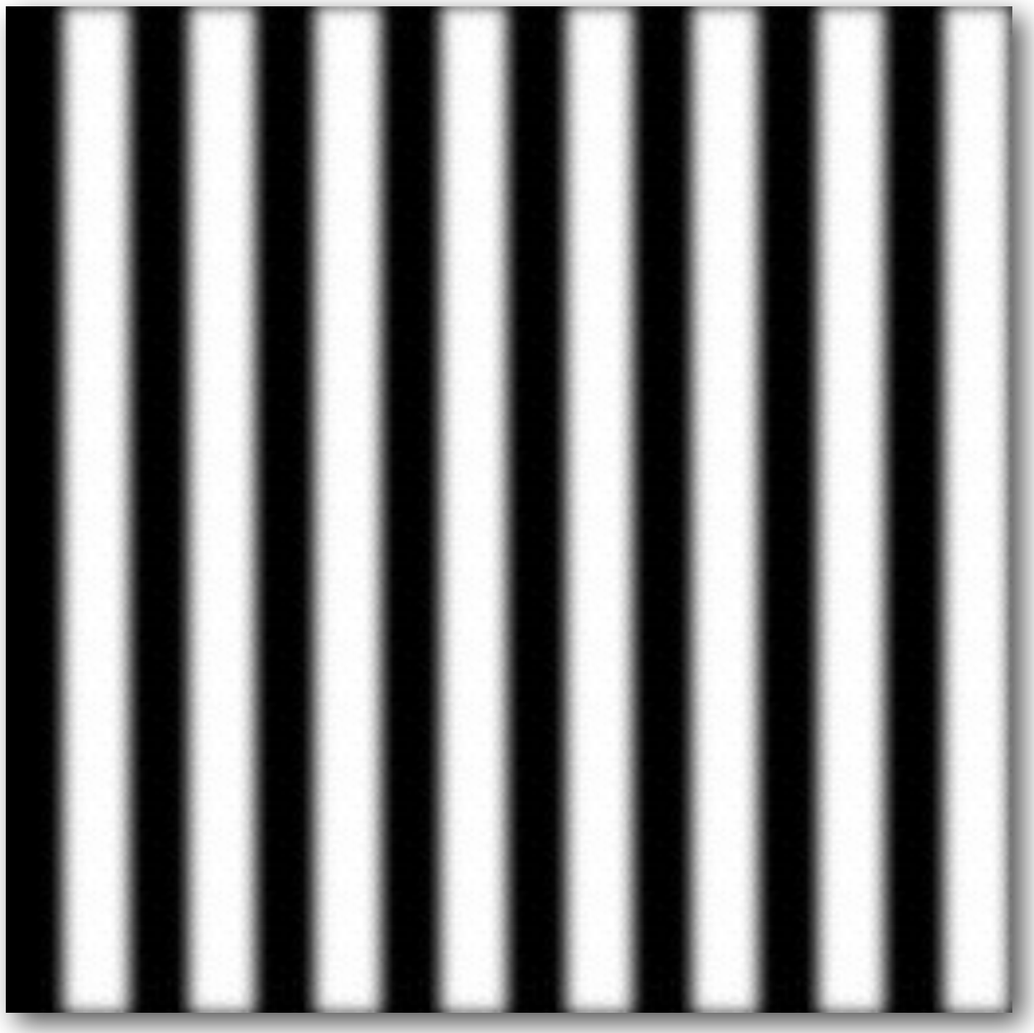
$$F[g * h] = F[g]F[h]$$

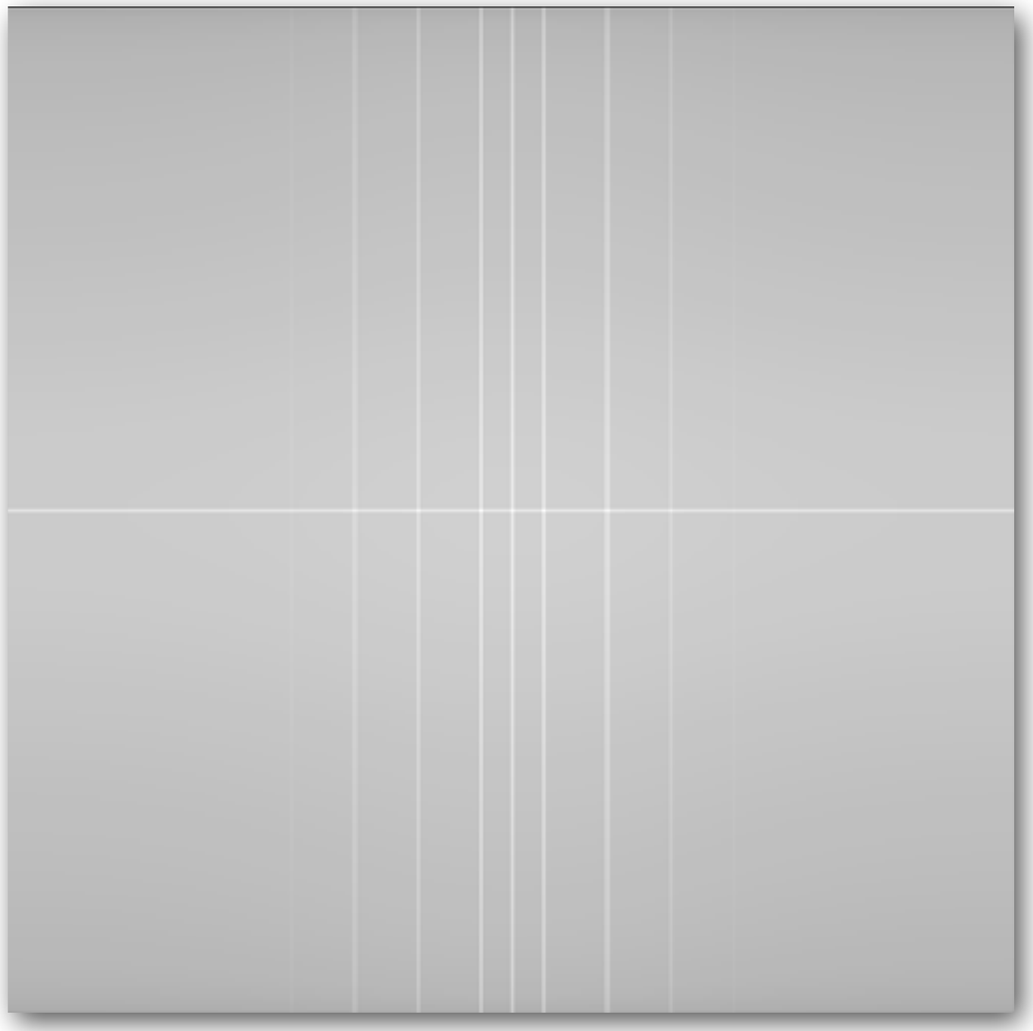
* **Inverse Fourier transform** of the product of two Fourier transforms = convolution of the two inverse Fourier transforms

$$F^{-1}[gh] = F^{-1}[g] * F^{-1}[h]$$

* Convolution in spatial domain is equivalent to multiplication in frequency domain!

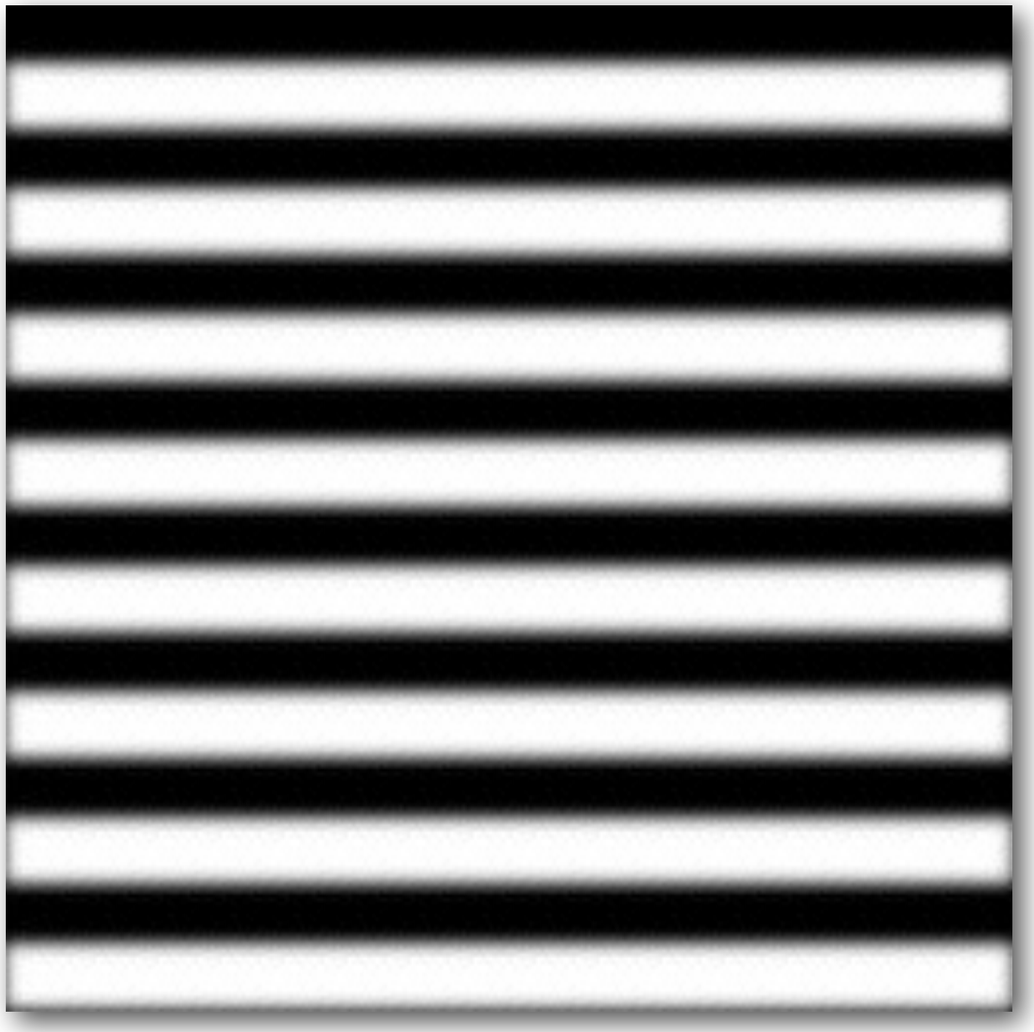
Now, Frequency Spectra for Images

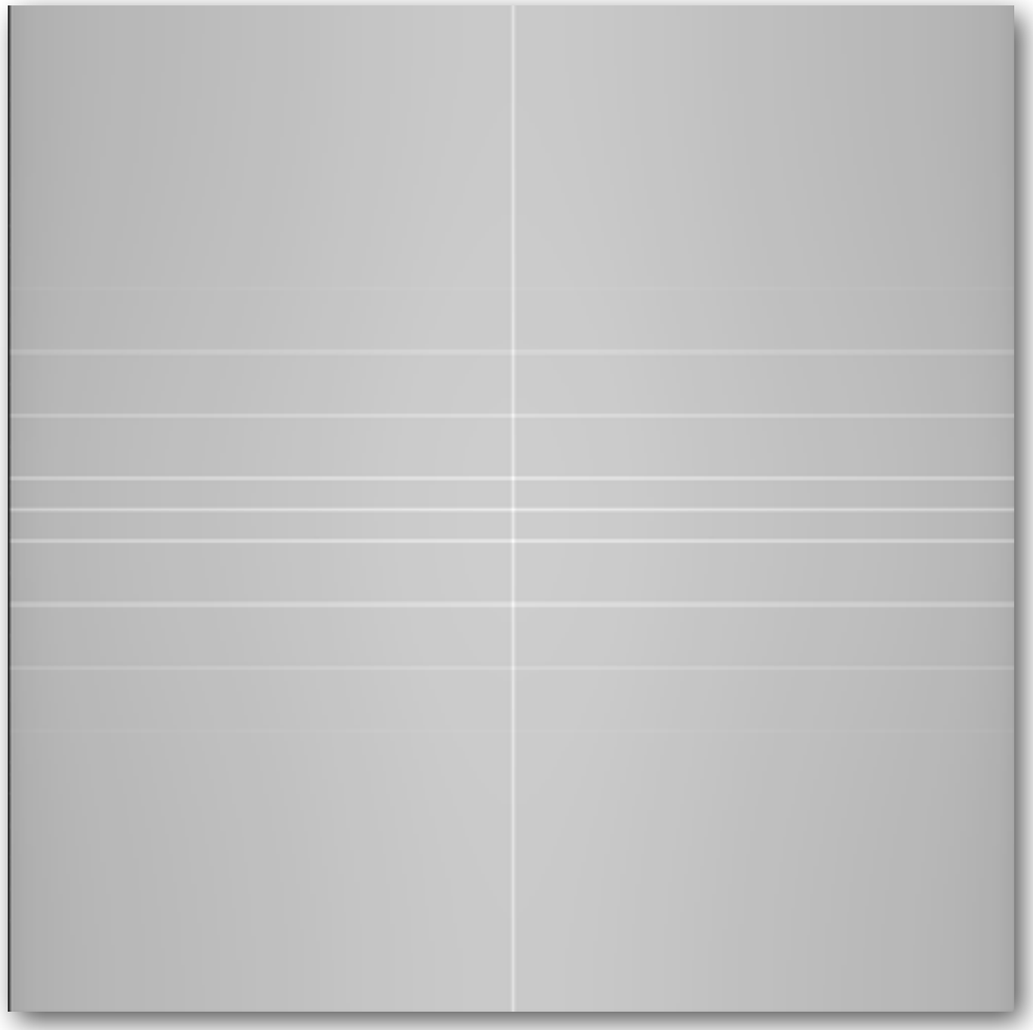




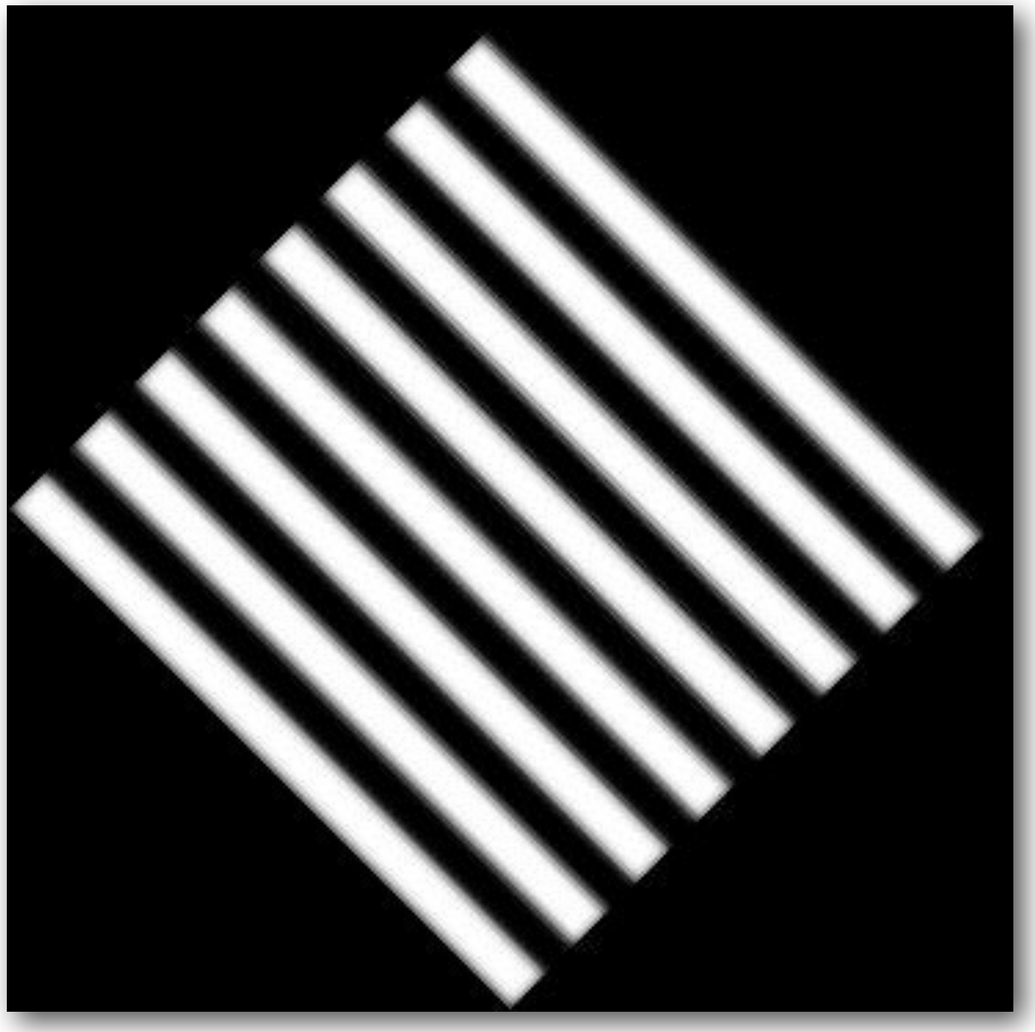
ω, Frequency

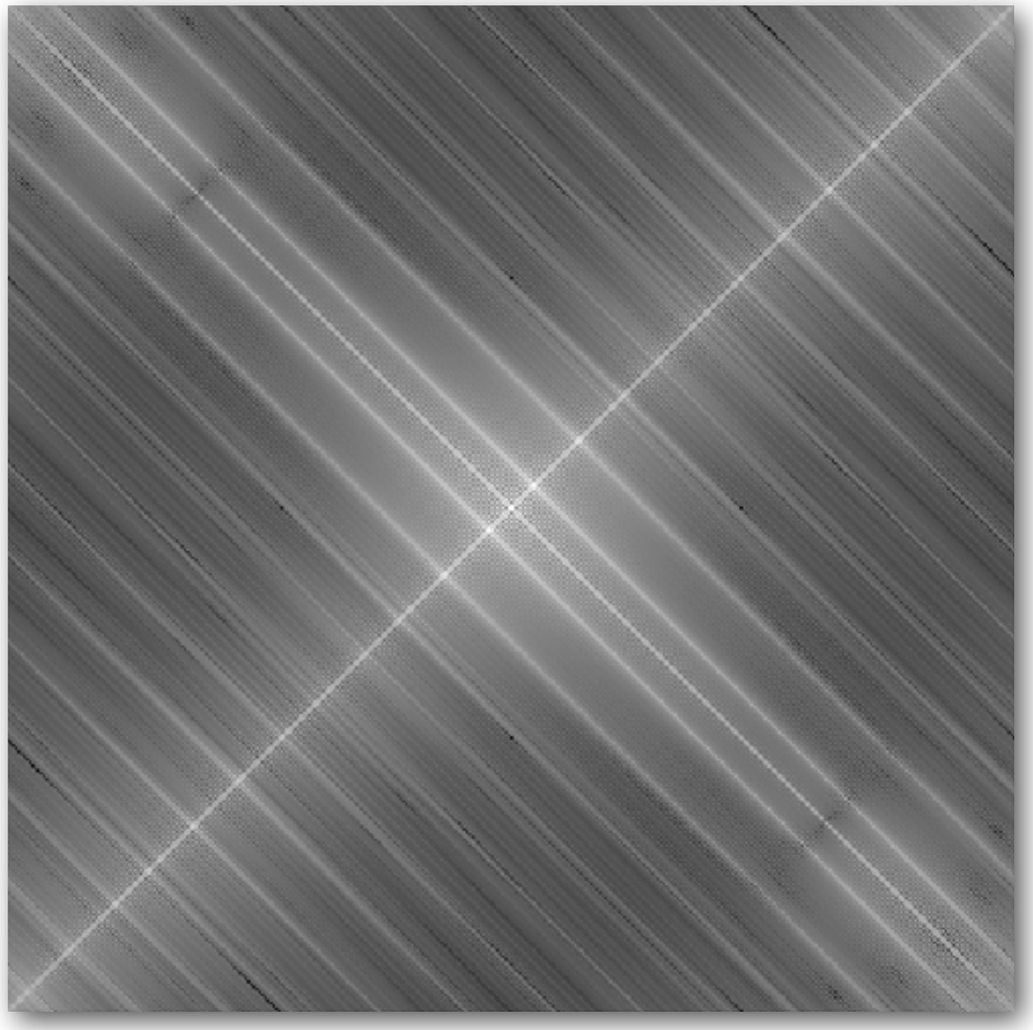
Now, Frequency Spectra for Images



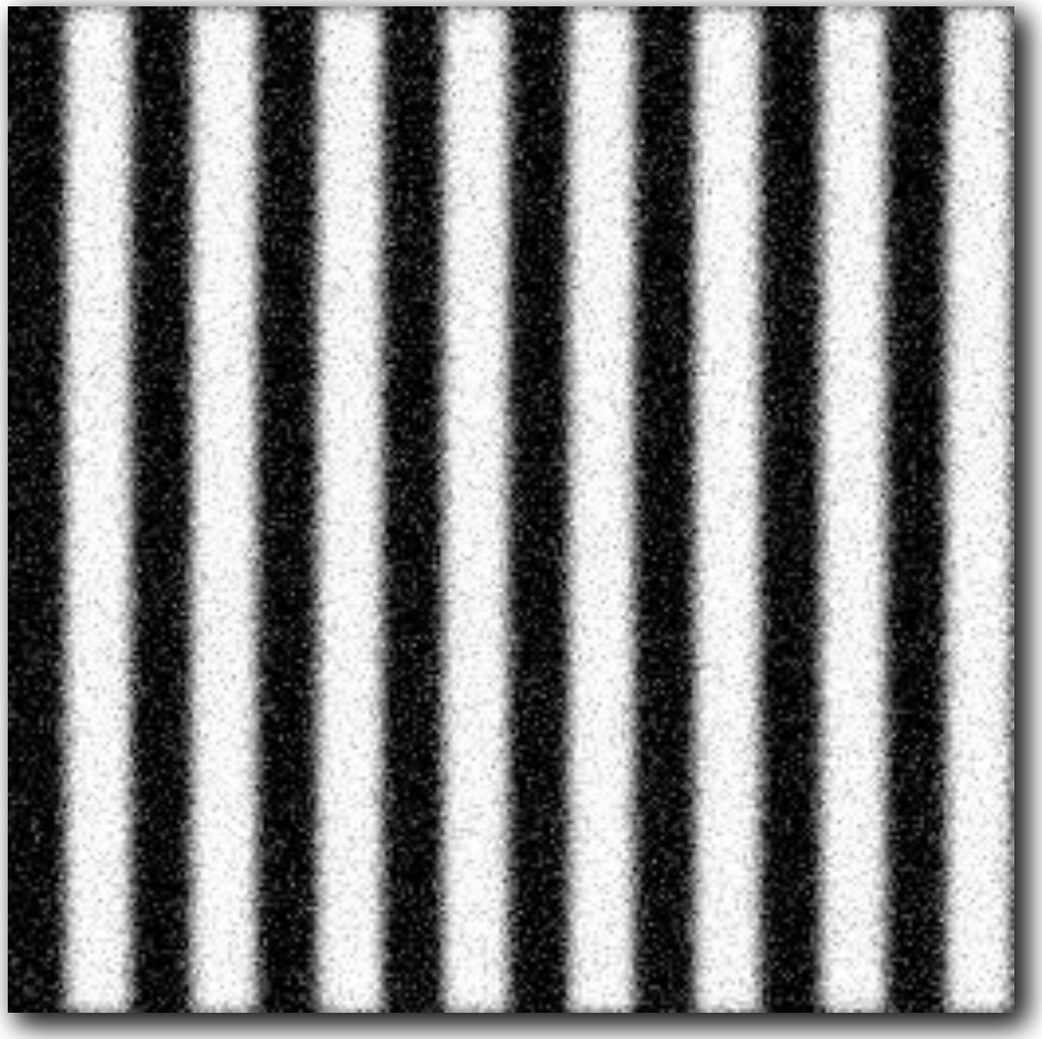


Now, Frequency Spectra for Images

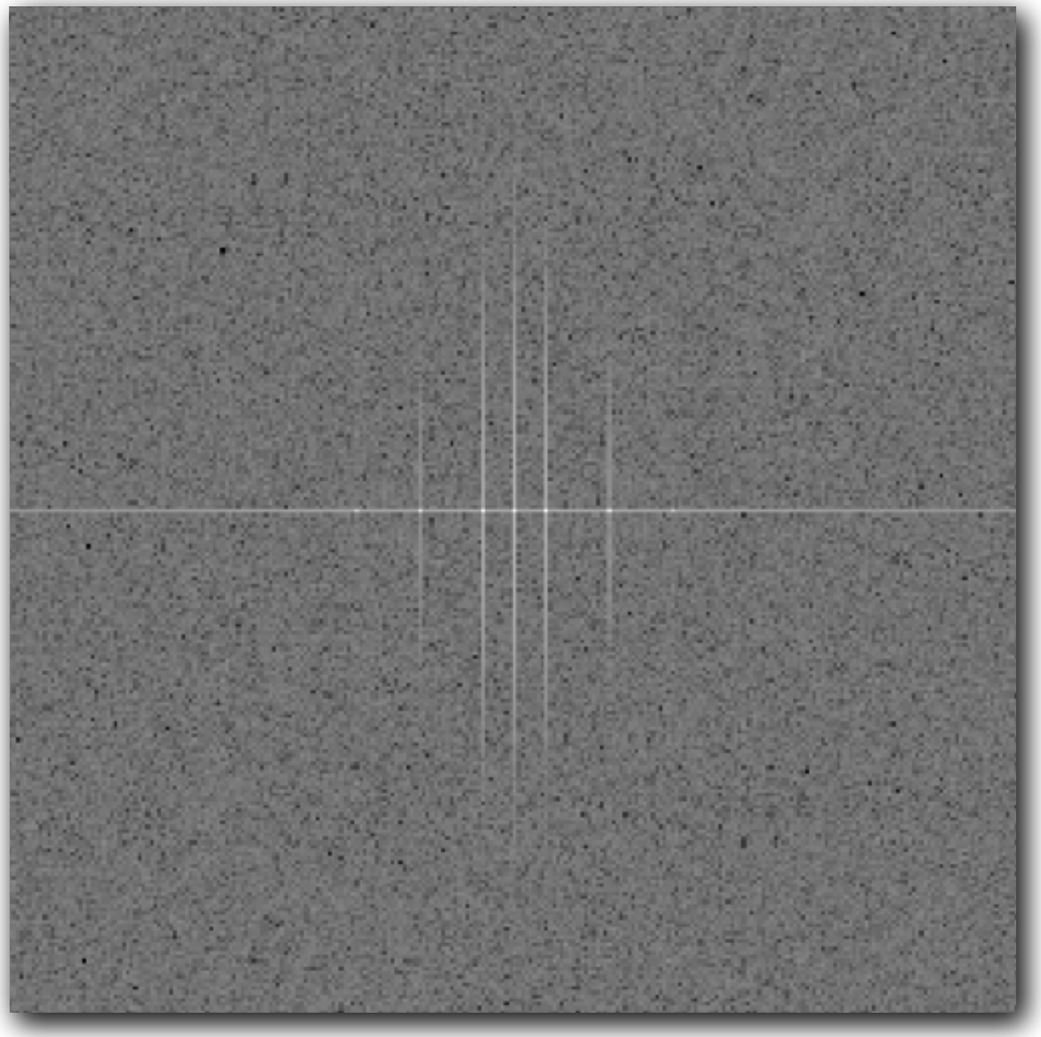




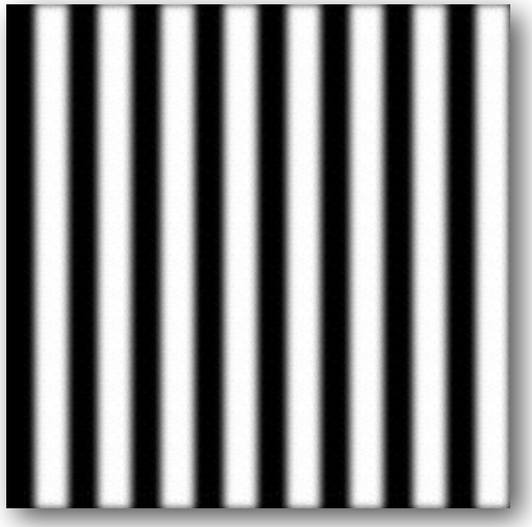
Now, Frequency Spectra for Images

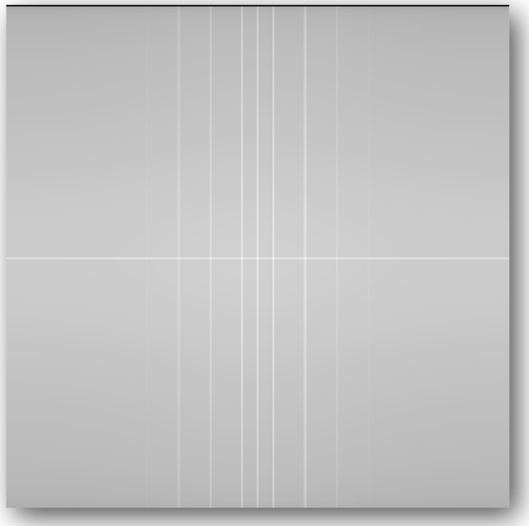


Noise Added

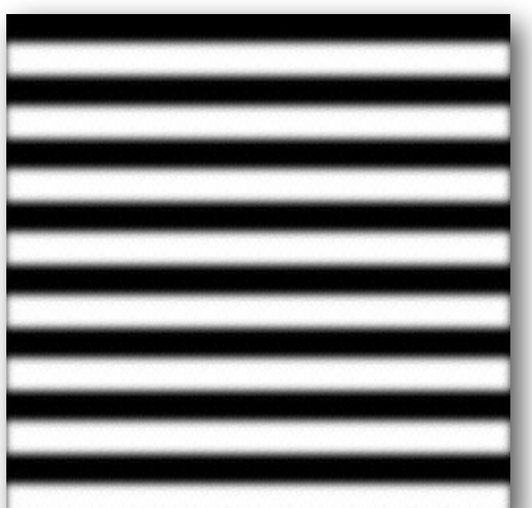


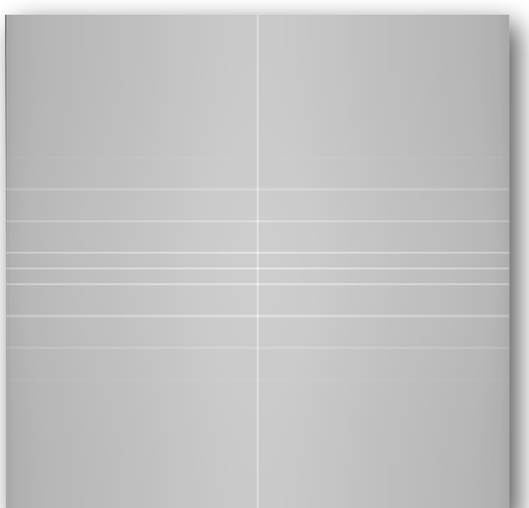
Now, Frequency Spectra for Images

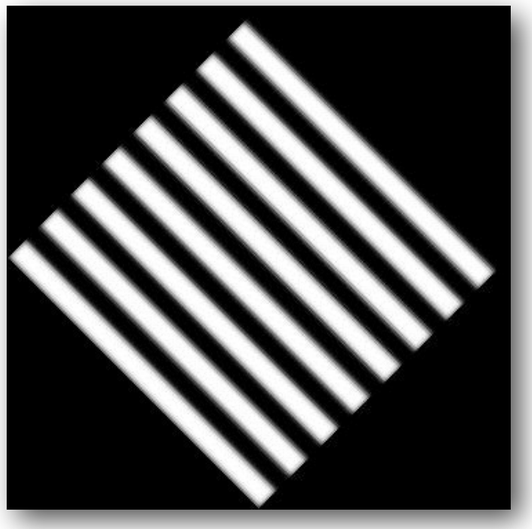


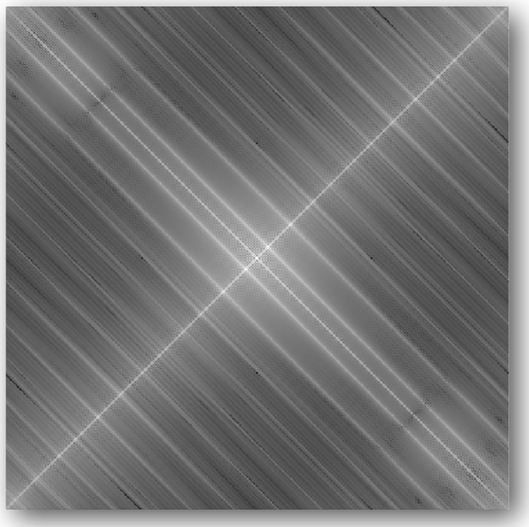


Noise Added

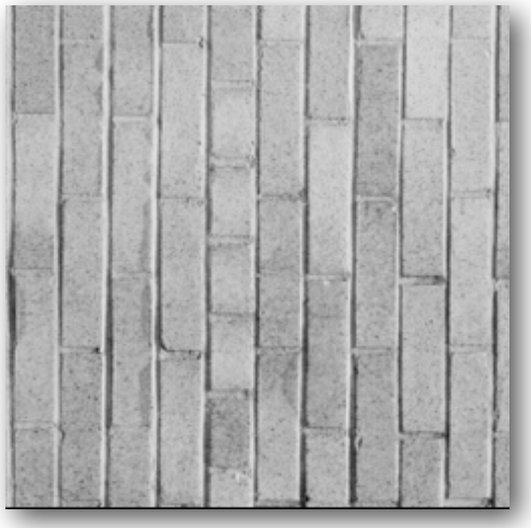


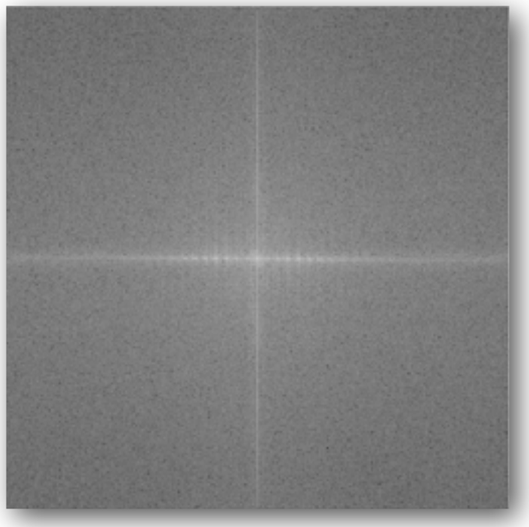


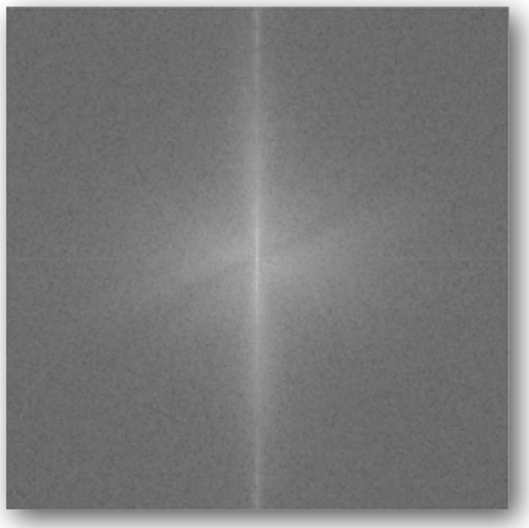


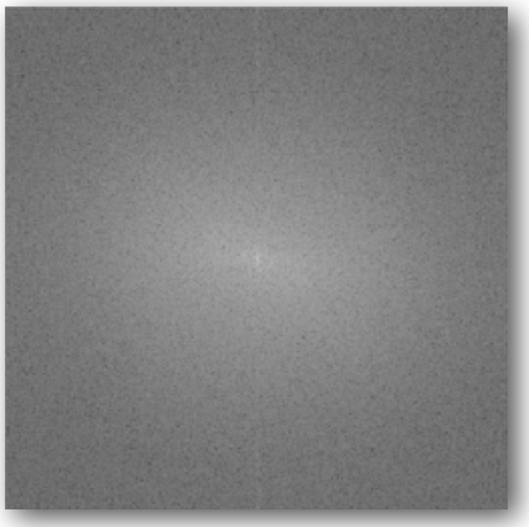


Frequency Spectra for Real Images









Fourier Transform: Some Observations

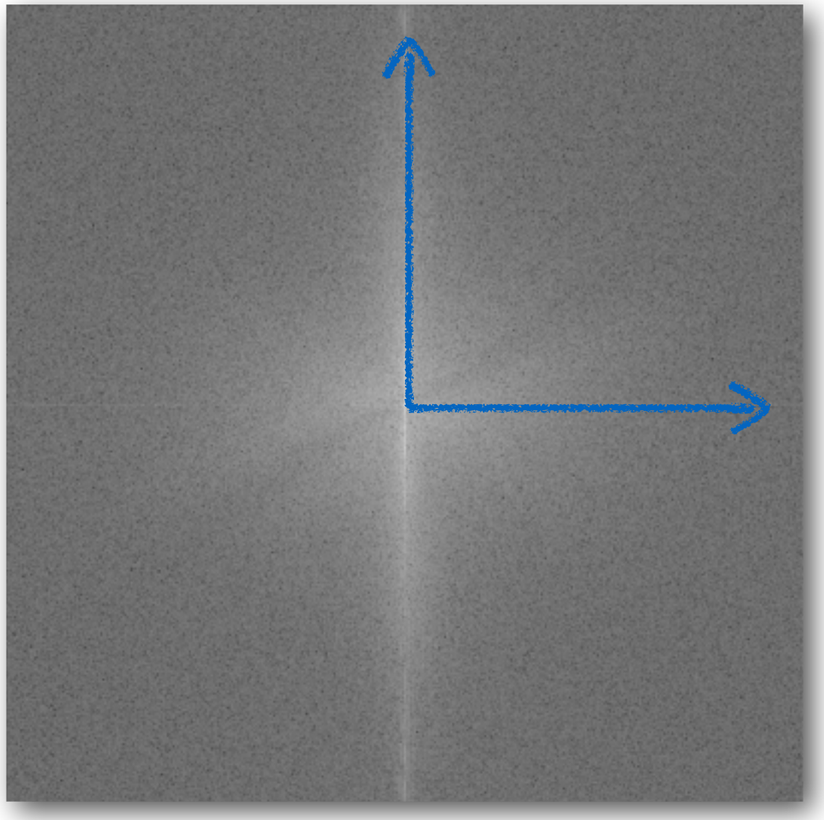
* For every ω from 0 to ∞ (infinity) $F(\omega)$ holds the Amplitude A and phase ϕ of a sine function.
* It uses Real and Complex Numbers to achieve this

$$A\sin(\omega t + \phi)$$

$$F(\omega) = R(\omega) + jI(\omega)$$

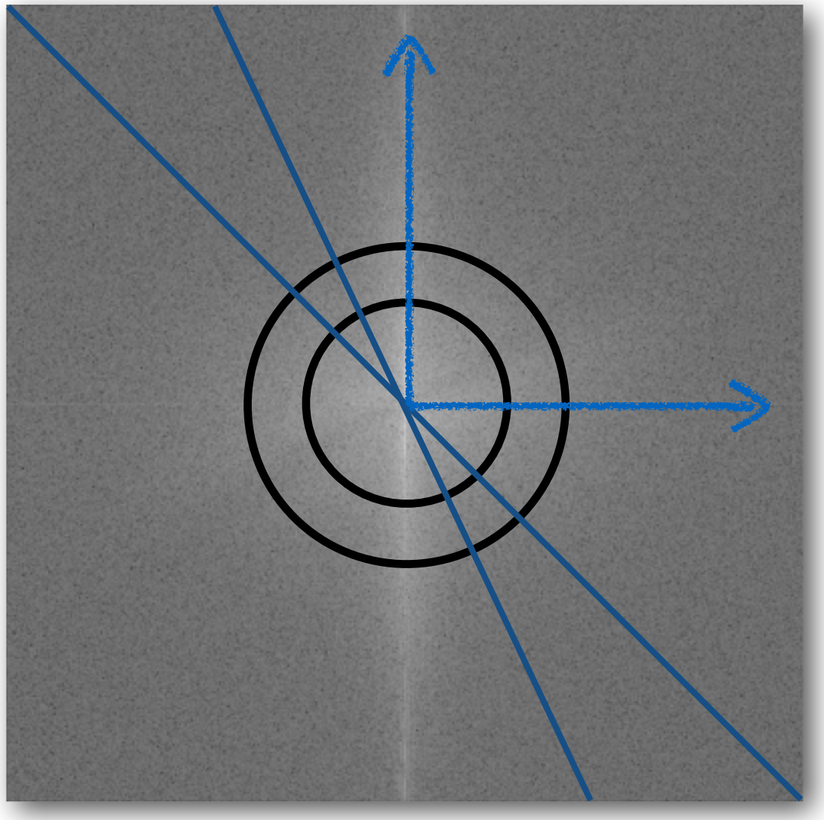
$$A = \pm\sqrt{R(\omega)^2 + I(\omega)^2}$$

$$\phi = \tan^{-1}\frac{I(\omega)}{R(\omega)}$$



Using the Frequency Spectra

* Low-pass,
* High-pass,
* Band-pass
  Filtering
* Change the spectrum and reconstruct



Blurring and Frequencies

Original Image

Gaussian 5x5 Blur

Smooth - Original

Summary

* Introduced how sines and cosines can be used to reconstruct a signal
* Characterized the Fourier Transform using Frequency, Amplitude and Phase
* Introduced the Frequency Domains for a Signal
* Identified the three properties of Convolution as it is associated with the Fourier Transform

Next Class

* Merging and Blending of Images

Credits

- For more information, see
  - Richard Szeliski (2010) Computer Vision: Algorithms and Applications, Springer
  - Forsyth & Ponce (2012), Computer Vision: A Modern Approach, Pearson
- Some concepts in slides motivated by similar slides by A. Efros and J. Hays
- Some images retrieved from
  - http://commons.wikimedia.org/
  - List will be available on website

Computational Photography

* Study the basics of computation and its impact on the entire workflow of photography, from capturing, manipulating and collaborating on, and sharing photographs.

Georgia Tech College of Computing

Computational Photography

* Study the basics of computation and its impact on the entire workflow of photography, from capturing, manipulating and collaborating on, and sharing photographs.

Georgia Tech College of Computing

Digital Images: Merging and Blending Images

* Different methods for Combing Multiple Images to Generate a Novel Image

Georgia Tech College of Computing

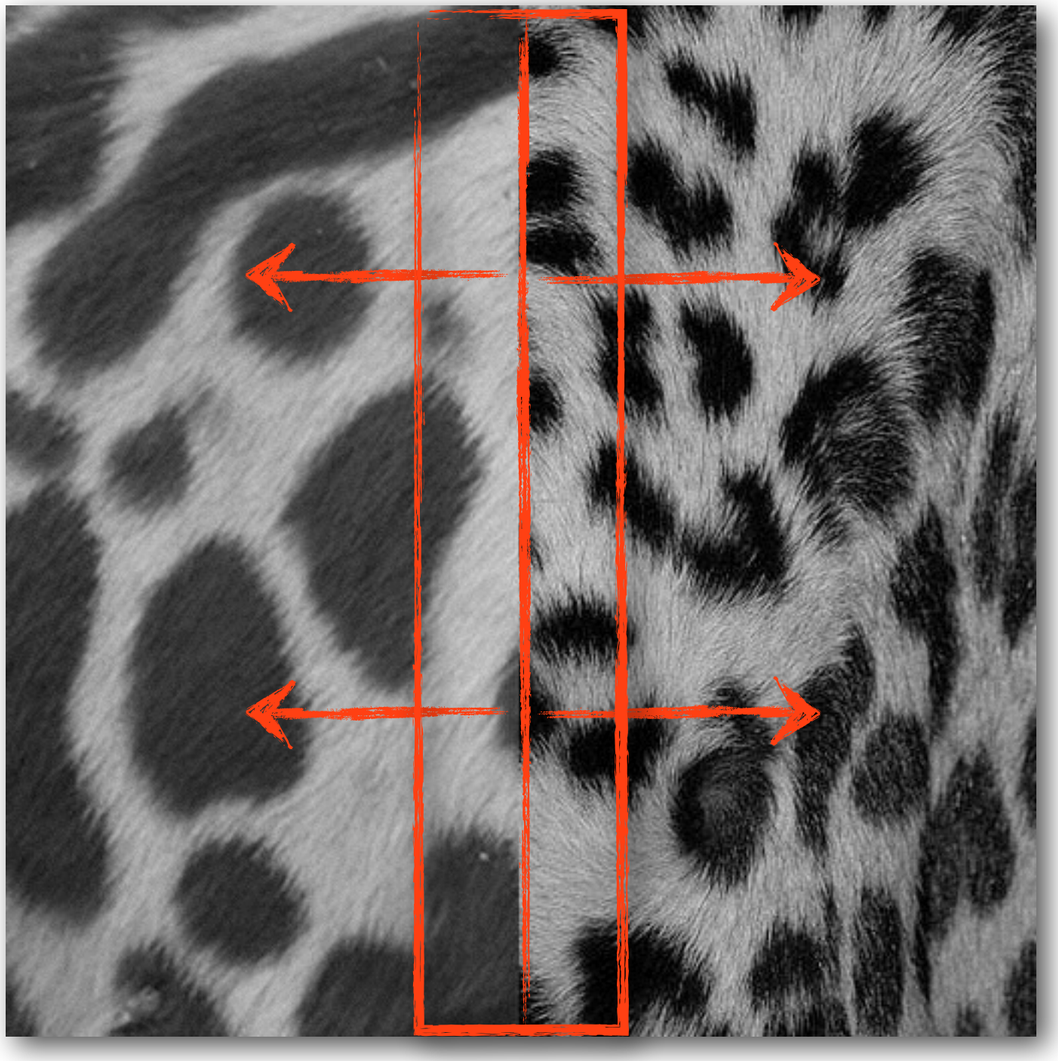
* merging two images
* Window sizes used for merging images
* Advantages of a using the Fourier Domain

Recall: Combine, Merge, Blend Images

Merging Two Images

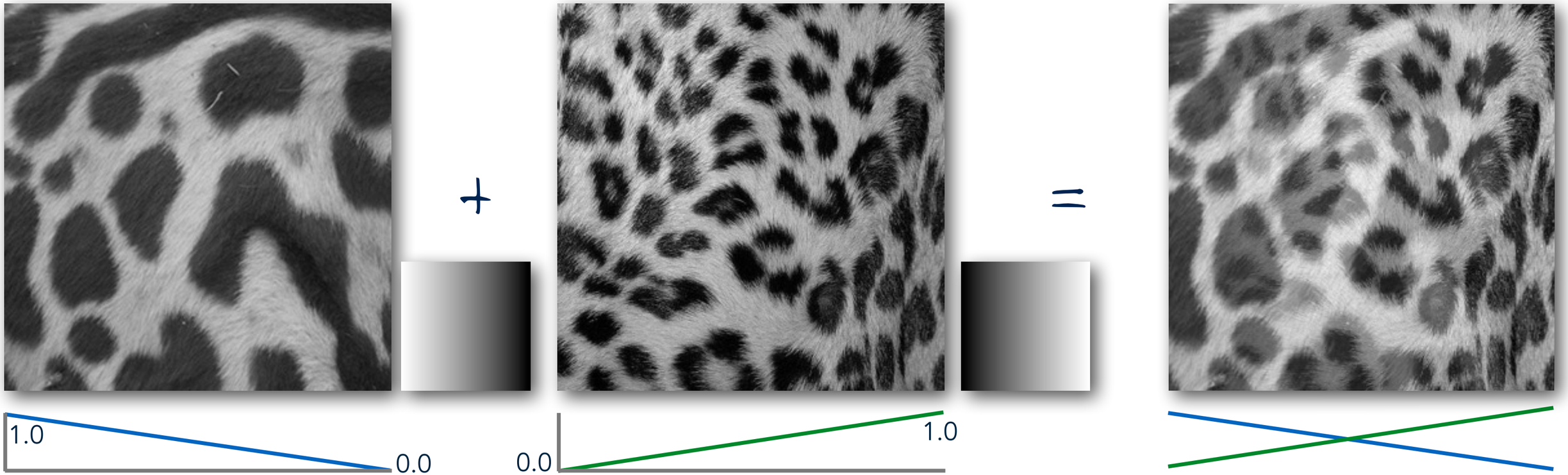
Merging Two Images

Merging Two Images

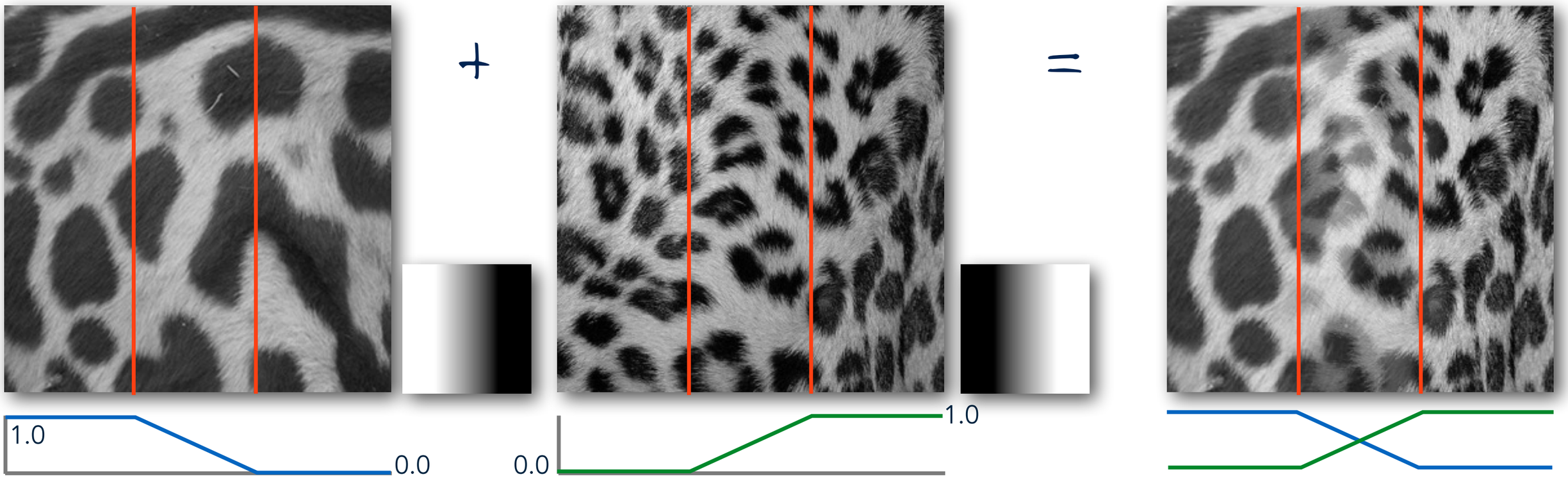


Cross-Fading Two Images

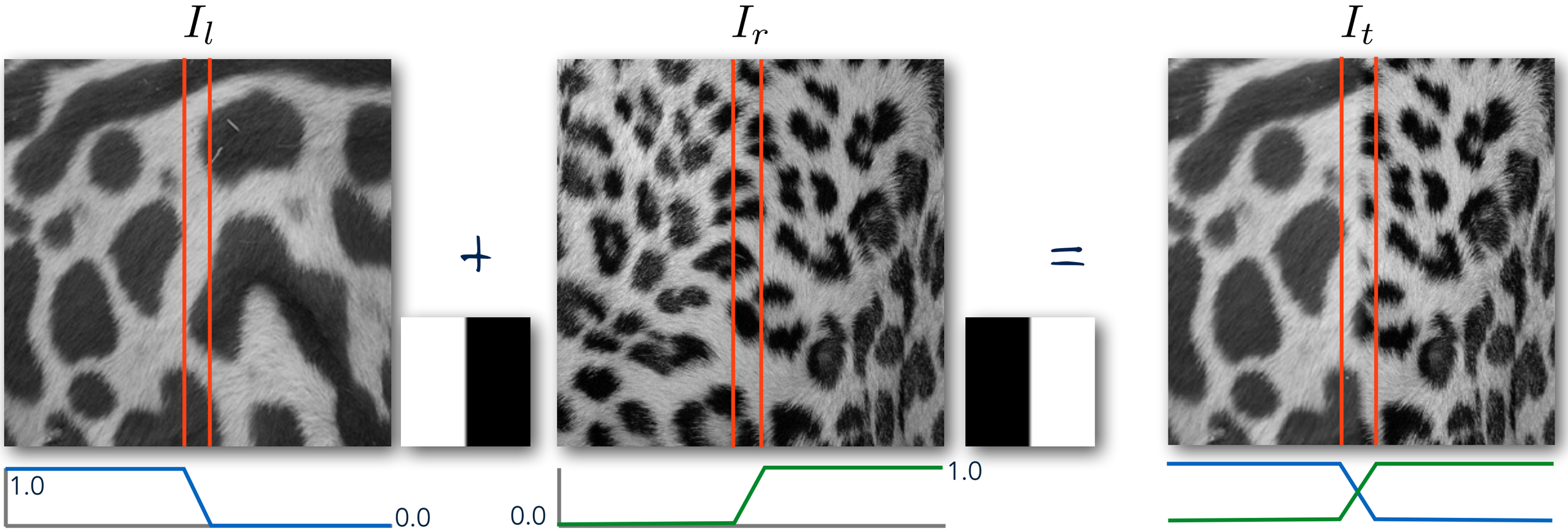
I_l I_r I_t



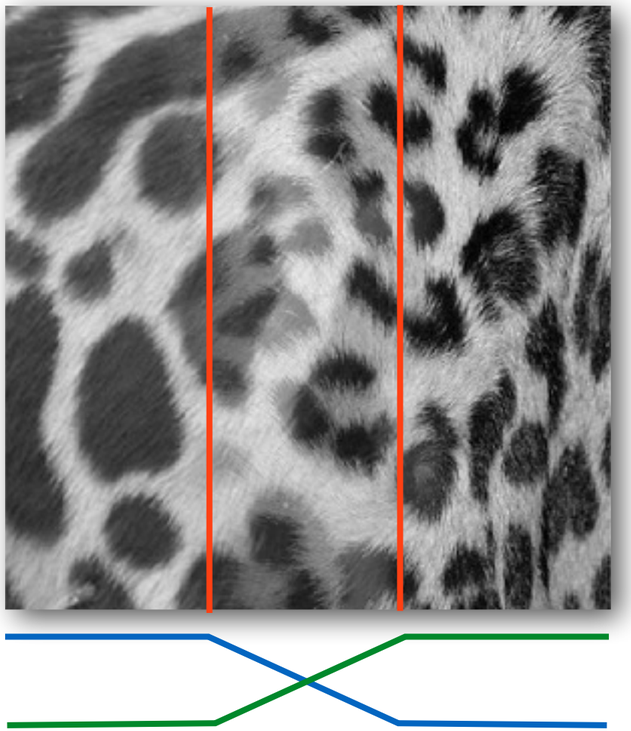
Cross-Fading Two Images

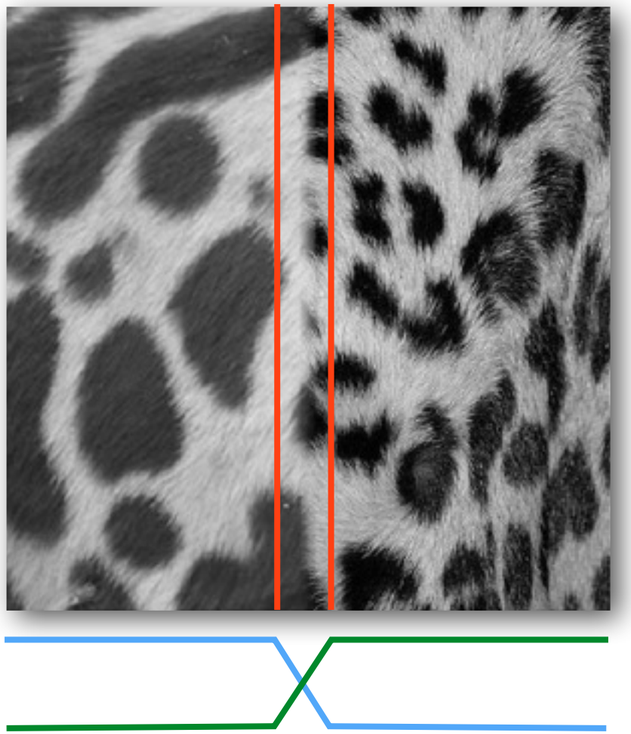


Cross-Fading Two Images

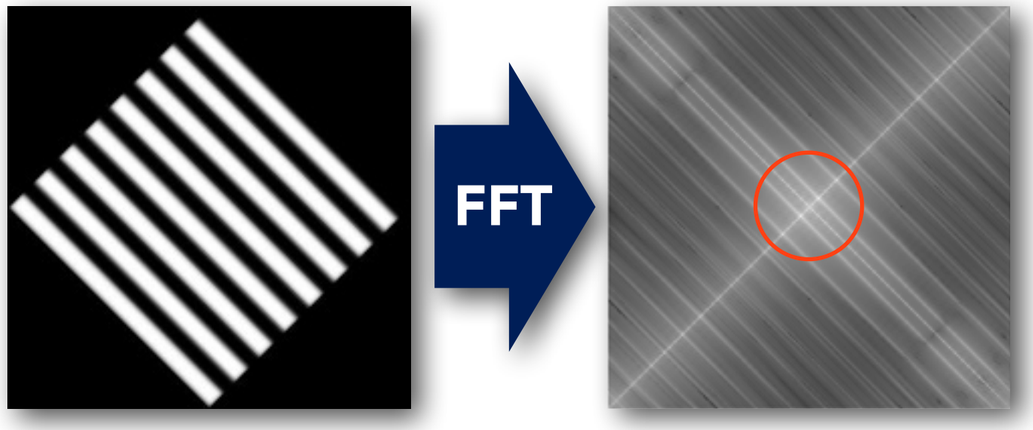


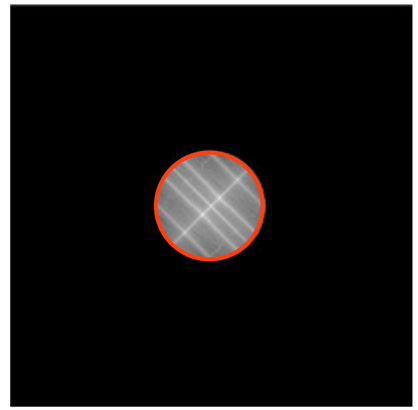
Cross-Fading Window Size

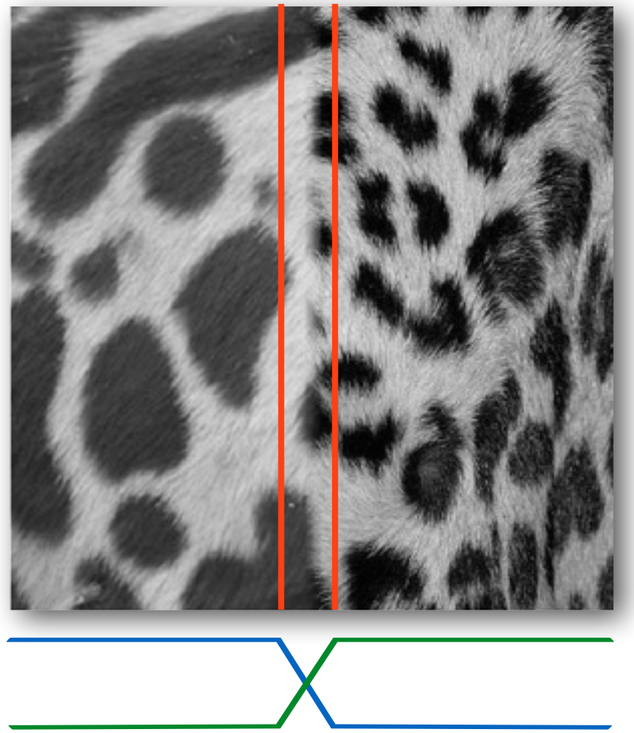




Factors for Optimal Window Size

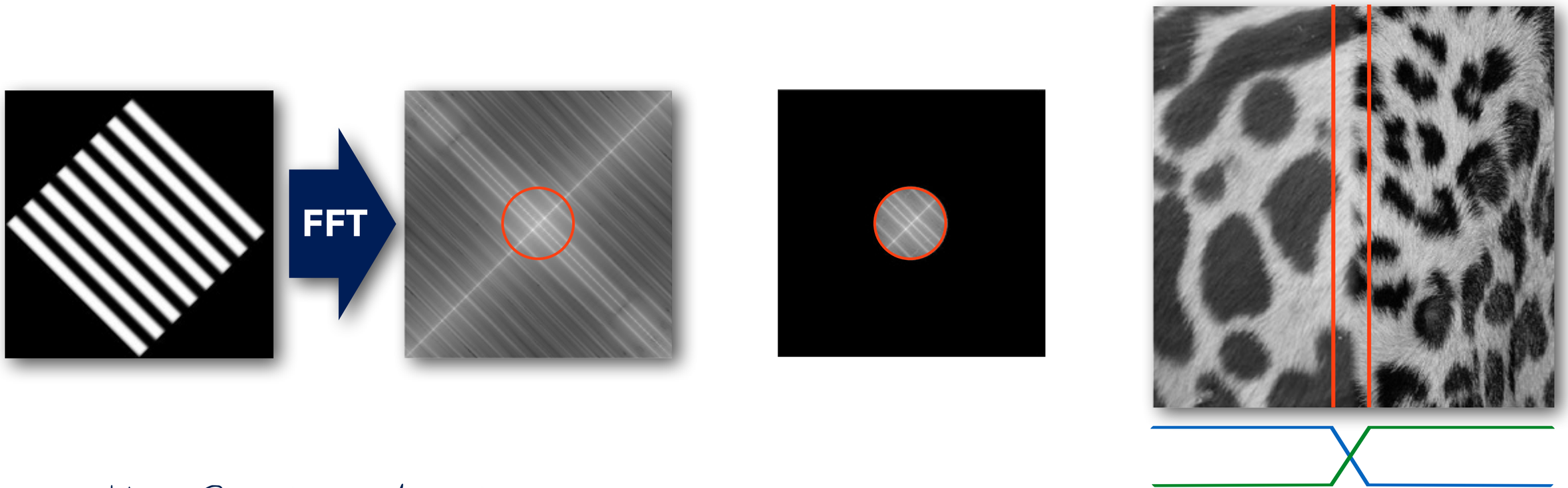






* To avoid seams: Window = size of largest prominent "feature"
* To avoid ghosting: Window $\leq 2\times$ size of smallest prominent "feature"

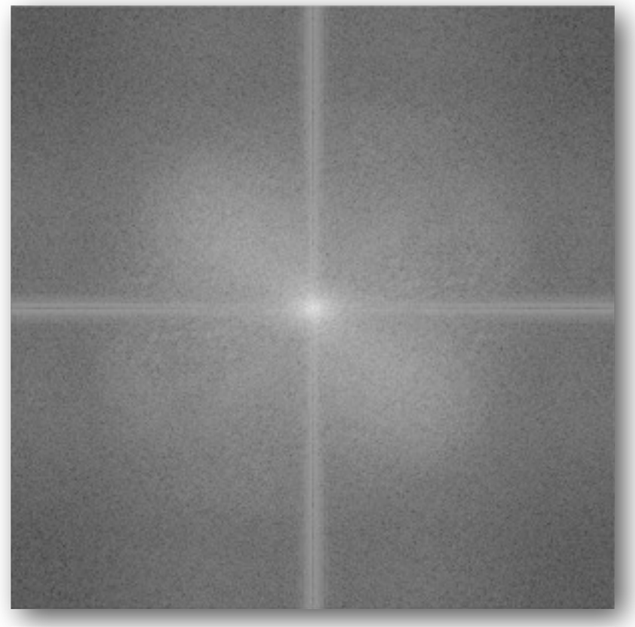
Factors for Optimal Window Size

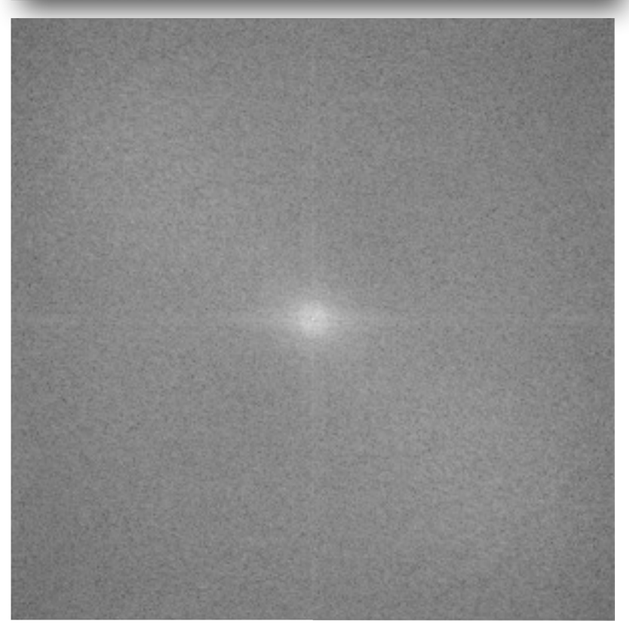


Use Fourier domain

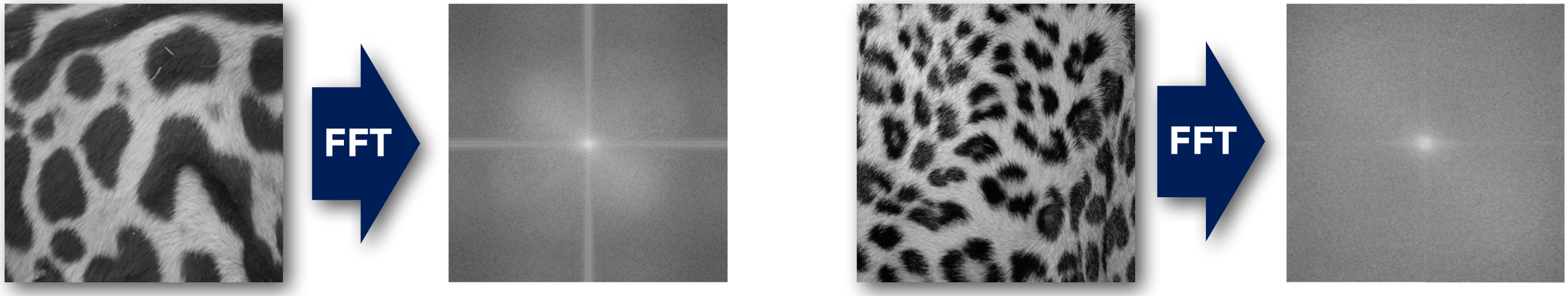
* Largest frequency $\leq 2 \times$ size of smallest frequency
* Image frequency content should occupy one "octave"
  * (power of 2)

Frequency Spread needs to be Modeled





Frequency Spread needs to be Modeled

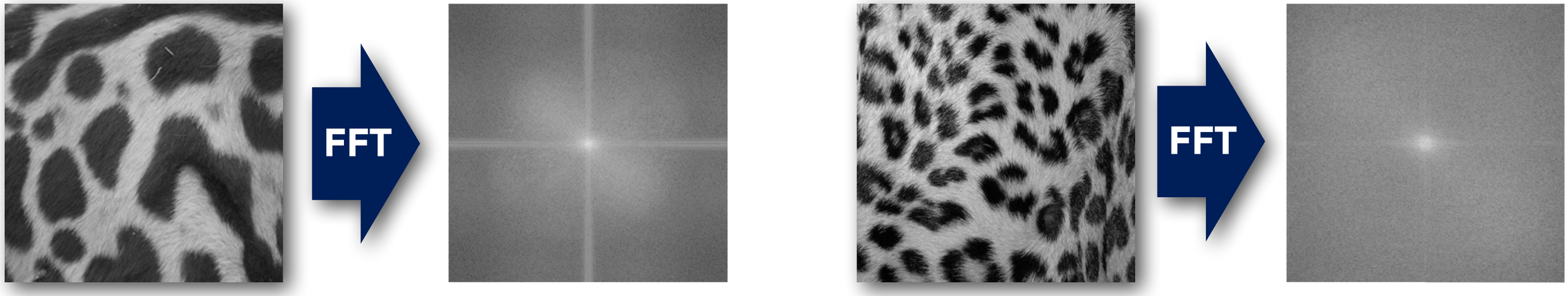


* Compute: $FFT(I_l) \Rightarrow F_l, \quad FFT(I_r) \Rightarrow F_r$

* Decompose Fourier image into octaves (bands)

$$F_l = F_l^1 + F_l^2 + F_l^3 + \ldots, \quad F_r = F_r^1 + F_r^2 + F_r^3 + \ldots$$

Frequency Spread needs to be Modeled



* "Feather" corresponding octaves of: F_l F_r
* Compute inverse FFT and feather in spatial domain
* Sum feathered octave images in frequency domain

Burt and Adelson (1983)

Feathering

* "Blur" the edges before applying the blend operations.
* Makes the blend, smoother.

Summary

* Merging two images
* Presented the two issues caused by not being able to determine the window used for merging images
* Presented the two advantages of a using the Fourier Domain

Next Class

* Merging and Blending of Images: Use of Pyramids

Credits

- For more information, see
  - Szeliski (2010) Computer Vision: Algorithms and Applications, Springer.
  - Burt and Adelson (1983) "The Laplacian Pyramid as a Compact Image Code", In IEEE Transactions on Communications, 31 (4). p 532-540. 1983 (DOI)
  - Burt and Adelson (1983) "A multiresolution spline with application to image mosaics". In ACM Transactions on Graphics, 2 (4). 1983 (DOI)
- Some concepts in slides motivated by similar slides by A. Efros and J. Hays.
- Some images retrieved from
  - http://commons.wikimedia.org/.
  - List will be available on website
  - by Irfan Essa

Computational Photography

* Study the basics of computation and its impact on the entire workflow of photography, from capturing, manipulating and collaborating on, and sharing photographs.

Georgia Tech College of Computing

Computational Photography

* Study the basics of computation and its impact on the entire workflow of photography, from capturing, manipulating and collaborating on, and sharing photographs.

Georgia Tech College of Computing

Digital Images: Merging and Blending Images using Image Pyramids

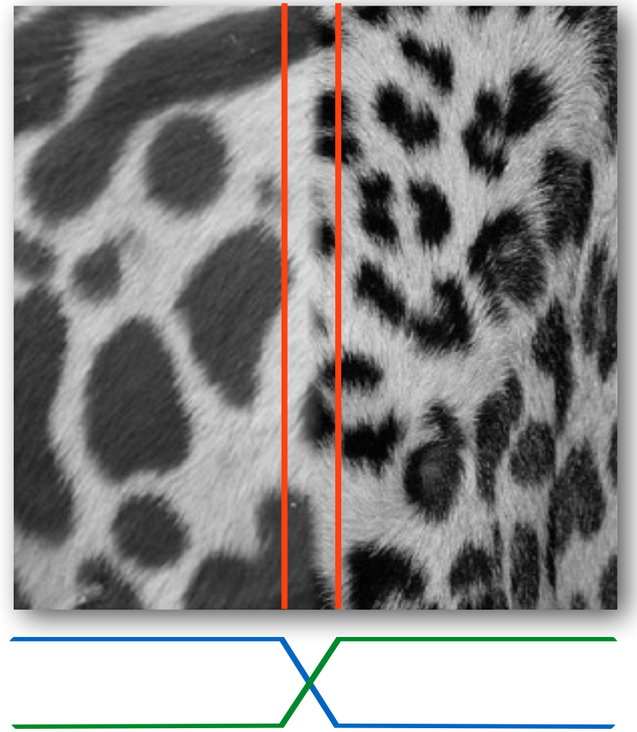
* Combing Multiple Images to Generate a Novel Image: Image Pyramids

Georgia Tech College of Computing

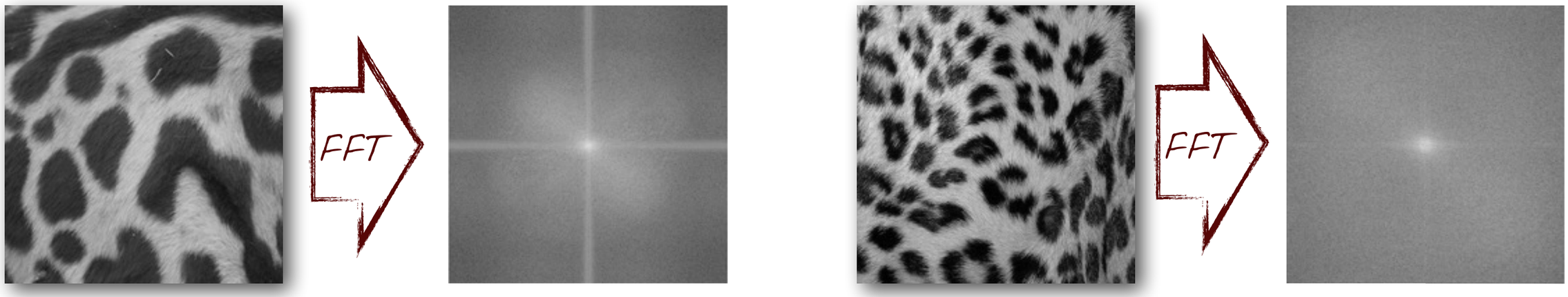
1. Gaussian and the Laplacian Pyramids
2. Use of Pyramids to encode the Frequency domain
3. Compute a Laplacian Pyramid form a Gaussian Pyramid
4. Blend two images using Pyramids

Recall: Optimal Window Size

* Avoid seams: Window = size of largest prominent "feature"
* Avoid ghosting: Window $\leq 2 \times$ size of smallest prominent "feature"
* Use Fourier domain
  * Largest frequency $\leq 2 \times$ size of smallest frequency
  * Image frequency content should occupy one "octave" (power of two)



Recall: Frequency Spread needs to be Modeled

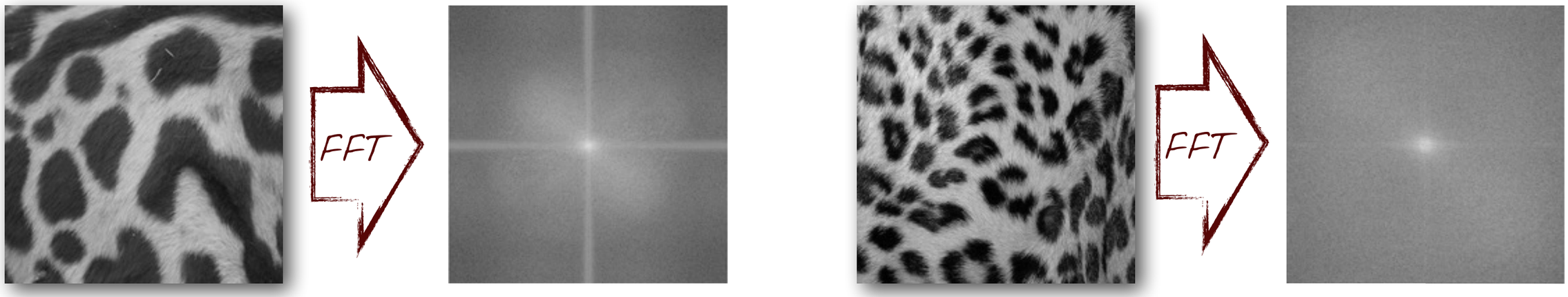


* Compute: $FFT(I_l) \Rightarrow F_l, \quad FFT(I_r) \Rightarrow F_r$

* Decompose Fourier image into octaves (bands)

$$F_l = F_l^1 + F_l^2 + F_l^3 + \ldots, \quad F_r = F_r^1 + F_r^2 + F_r^3 + \ldots$$

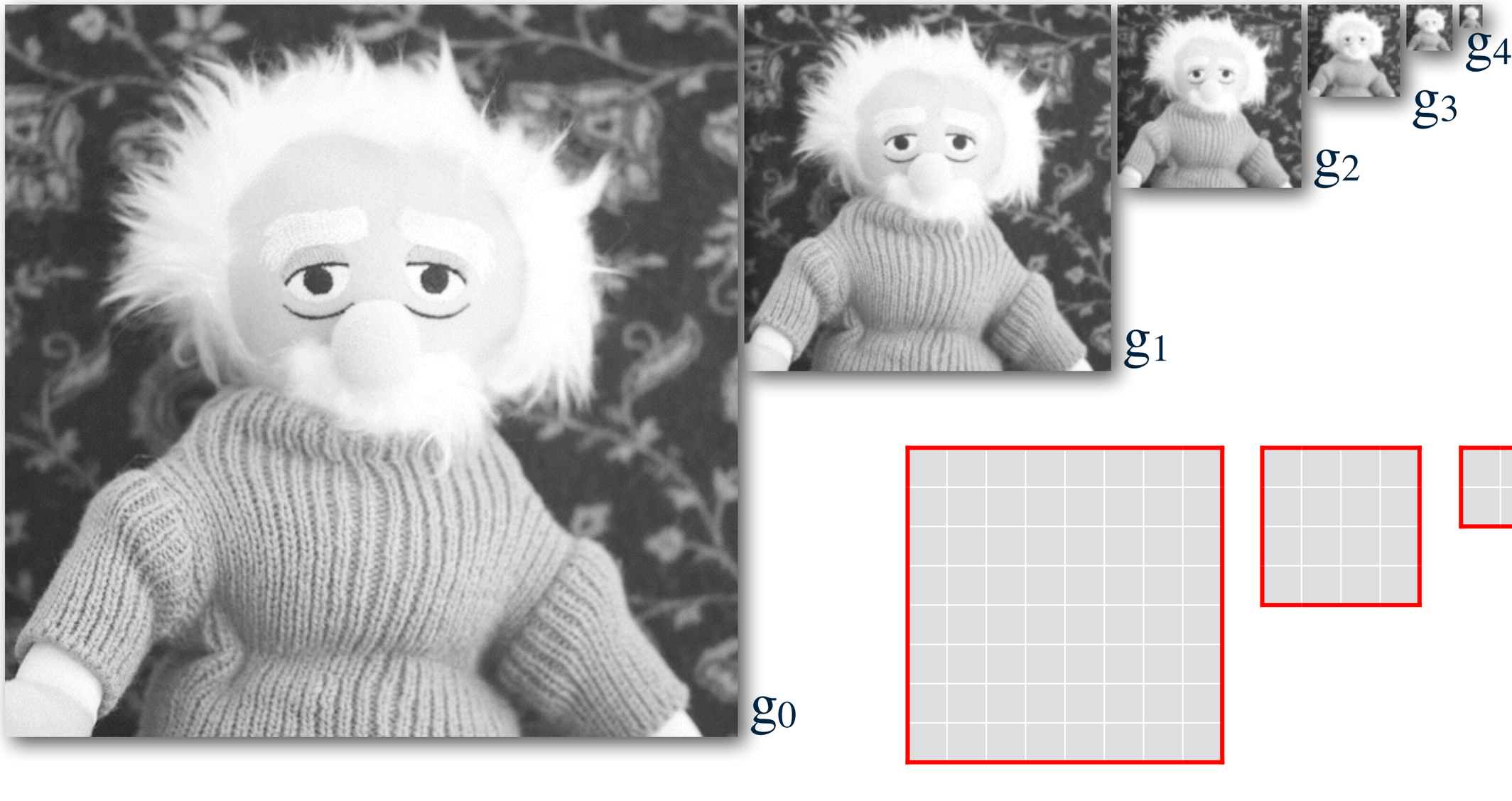
Recall: Frequency Spread needs to be Modeled



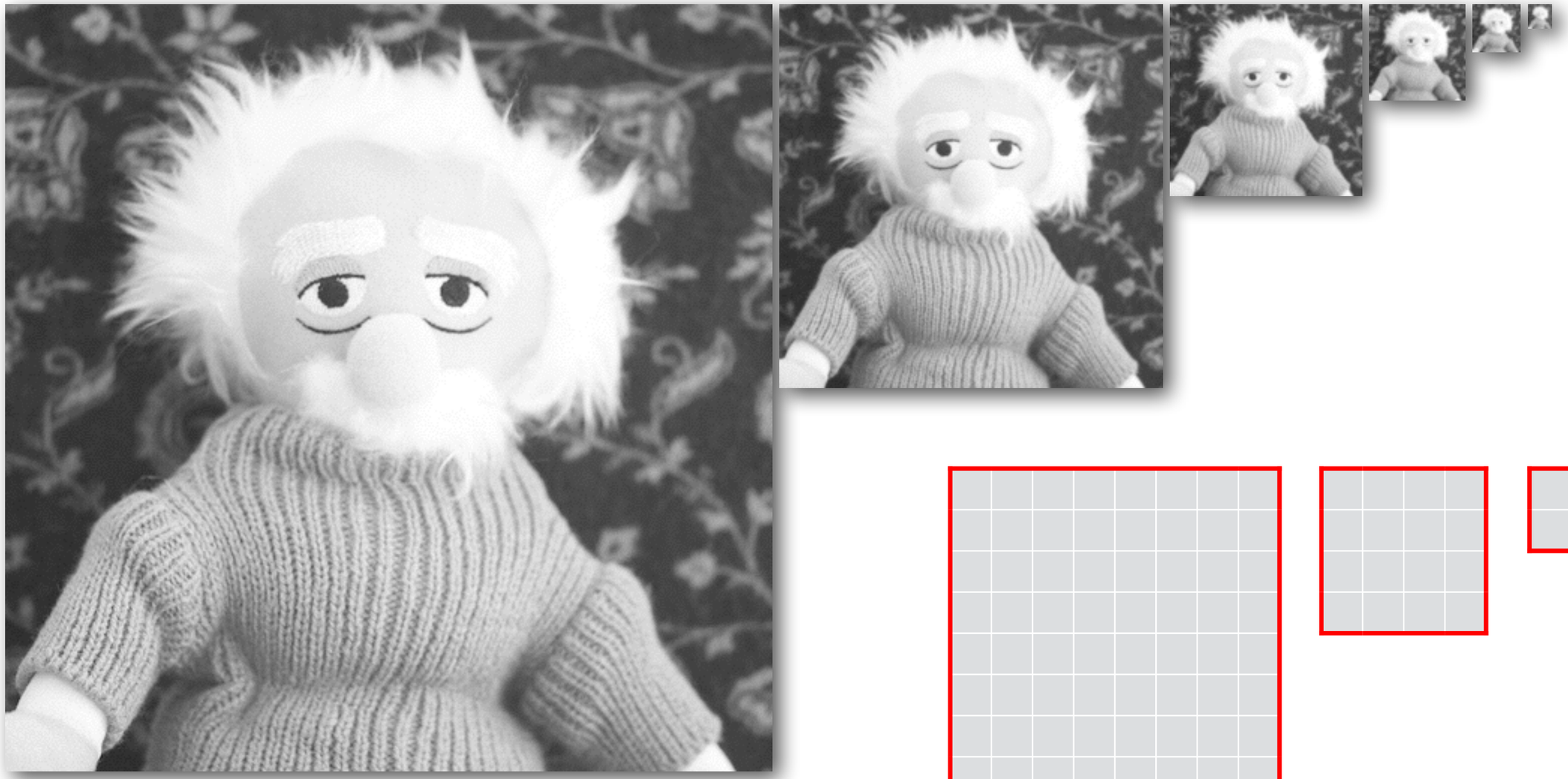
* "Feather" corresponding octaves of: F_l F_r
* Compute inverse FFT and feather in spatial domain
* Sum feathered octave images in frequency domain

Burt and Adelson (1983)

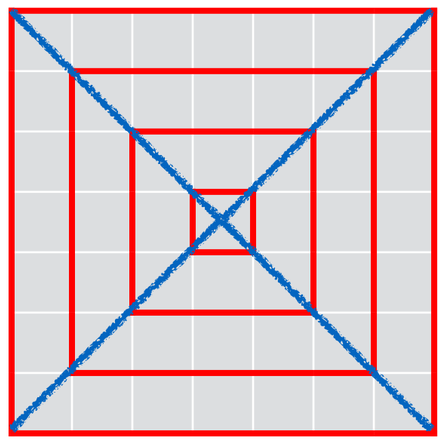
Pyramid Representation: A Gaussian Pyramid

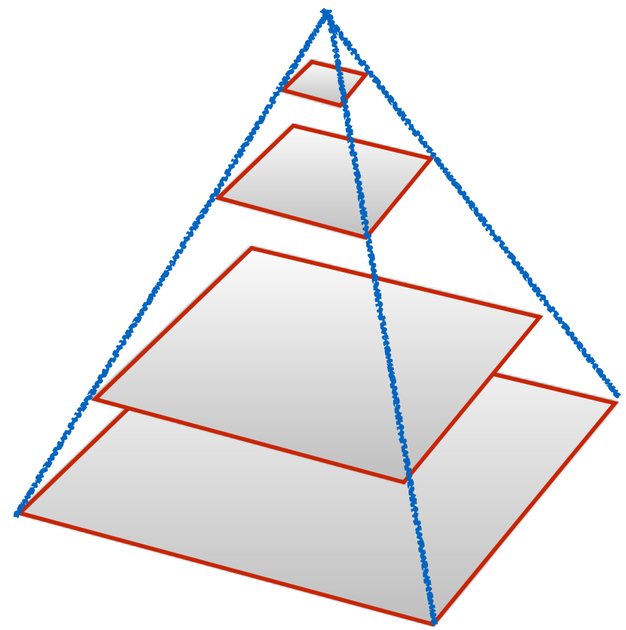


Pyramid Representation: A Gaussian Pyramid



Pyramid Representation: A Gaussian Pyramid





Pyramid Representation of Images (A Gaussian Pyramid)

g_0

g_1

g_2 g_3 g_4

Burt and Adelson (1983)

$$a = 0.3 - 0.6\ (.38)$$

$$h = \omega_h \star \omega_v$$

$\omega_v =$
1/4 - a/2
1/4
a
1/4
1/4 - a/2

$\omega_h =$				
1/4 - a/2	1/4	a	1/4	1/4 - a/2

Pyramid Representation of Images (A Gaussian Pyramid)

g_1 g_2 g_3 g_4 g_5

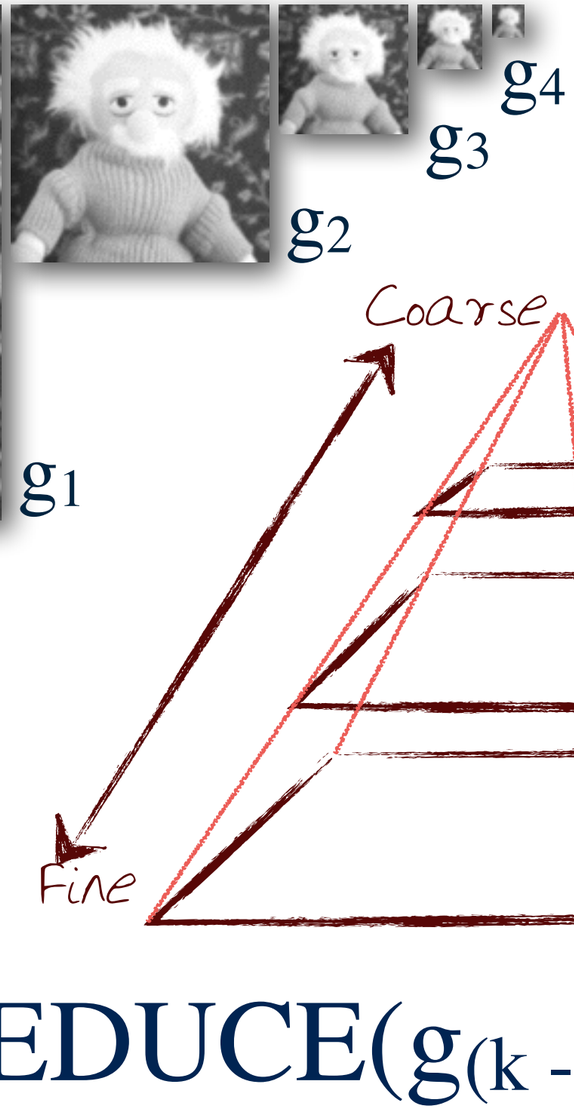
$$g_k = h \star g_{(k-1)}$$

$$g_k = \text{REDUCE}(g_{(k-1)})$$

Pyramid Representation of Images (A Gaussian Pyramid)

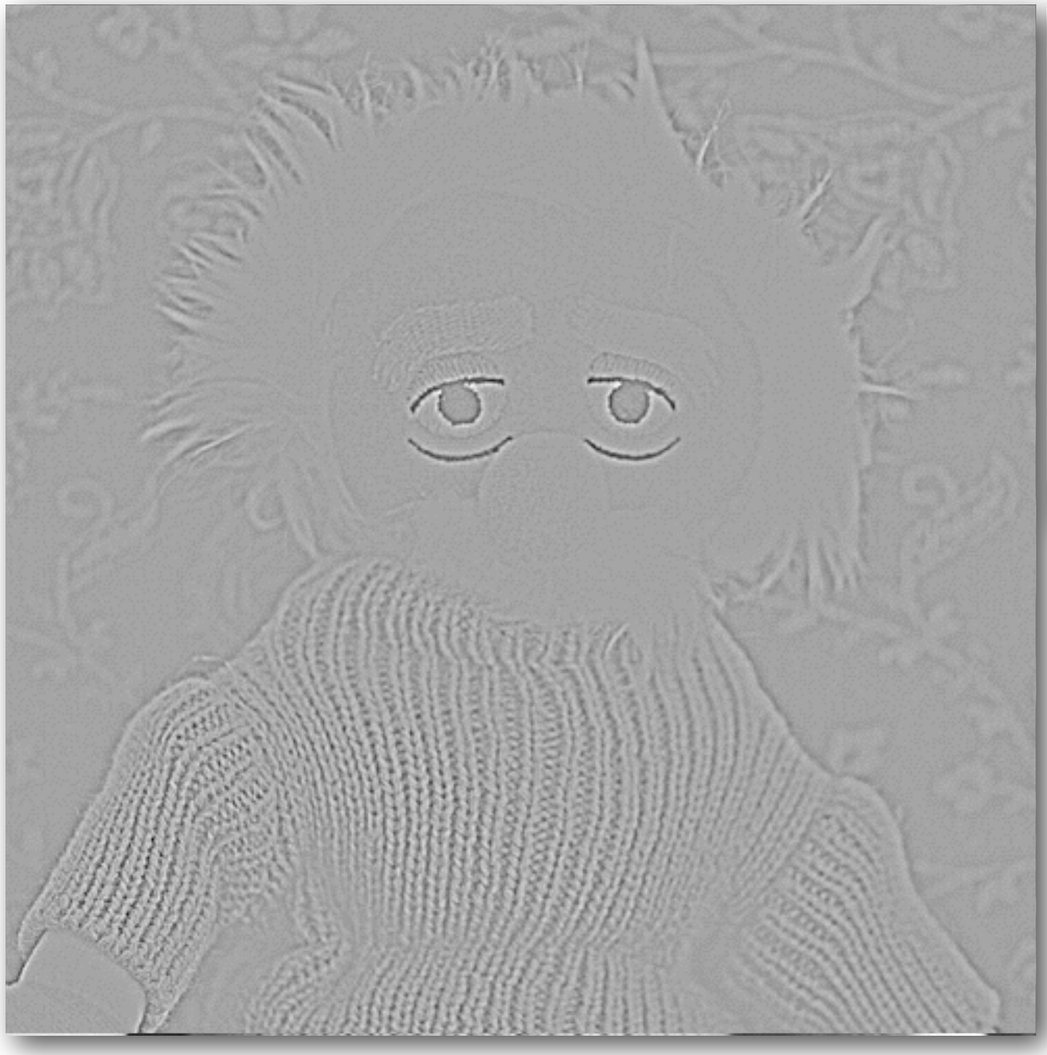
g_0

g_1



$$g_k = \text{REDUCE}(g_{(k-1)})$$

Pyramid Representation of Images (A Gaussian Pyramid)



g_0

$g_{0,1}$

$g_0 - g_{0,1}$

g_1

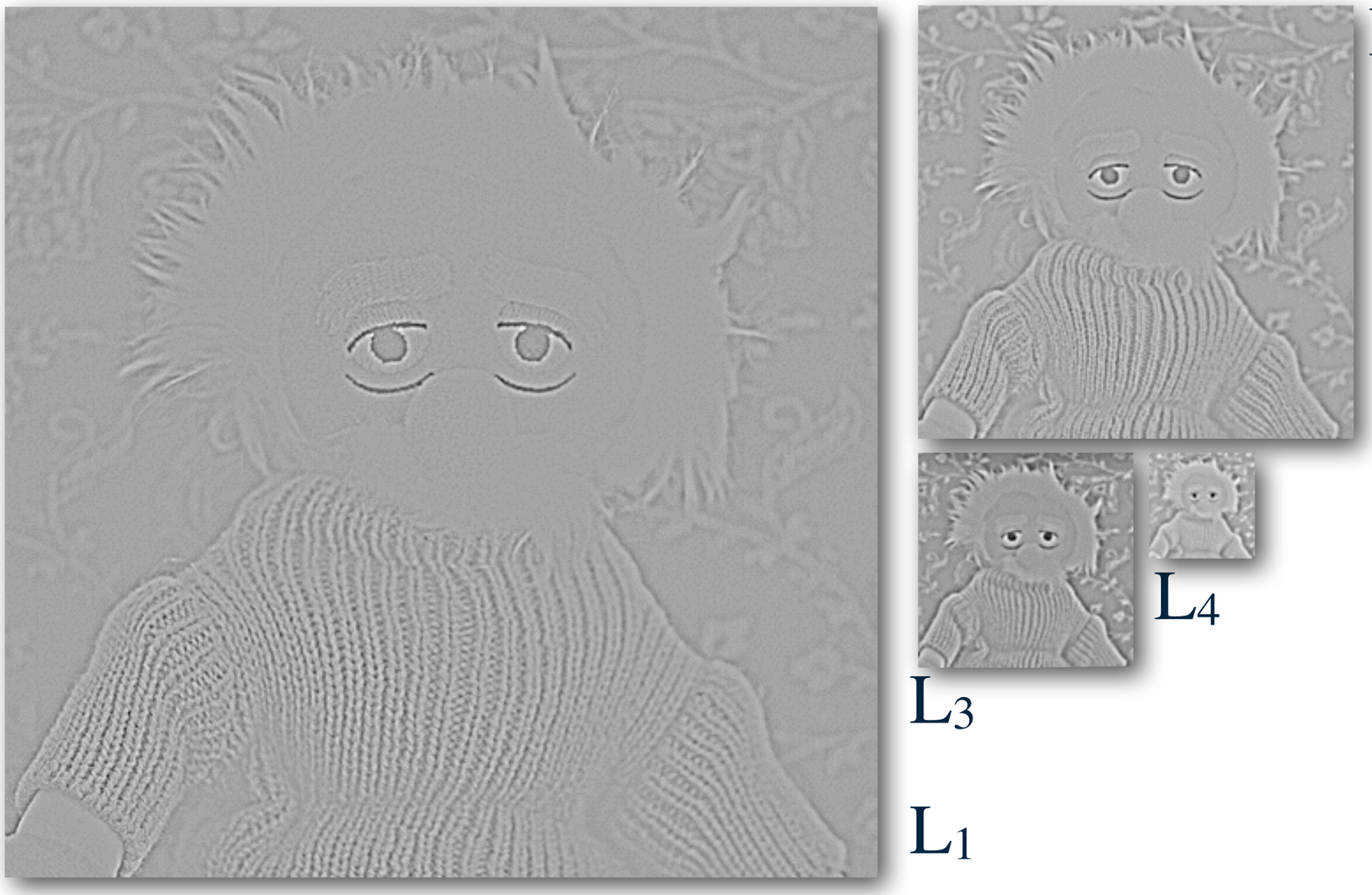
$$g_k = \text{REDUCE}(g_{k-1})$$

$$g_{j,n} = \text{EXPAND}(g_{j,n-1})$$

EXPAND is inverse of REDUCE, as it seeks to add new values in between knowns ones.

$g_{j,n}$ is g_j expanded **n** times

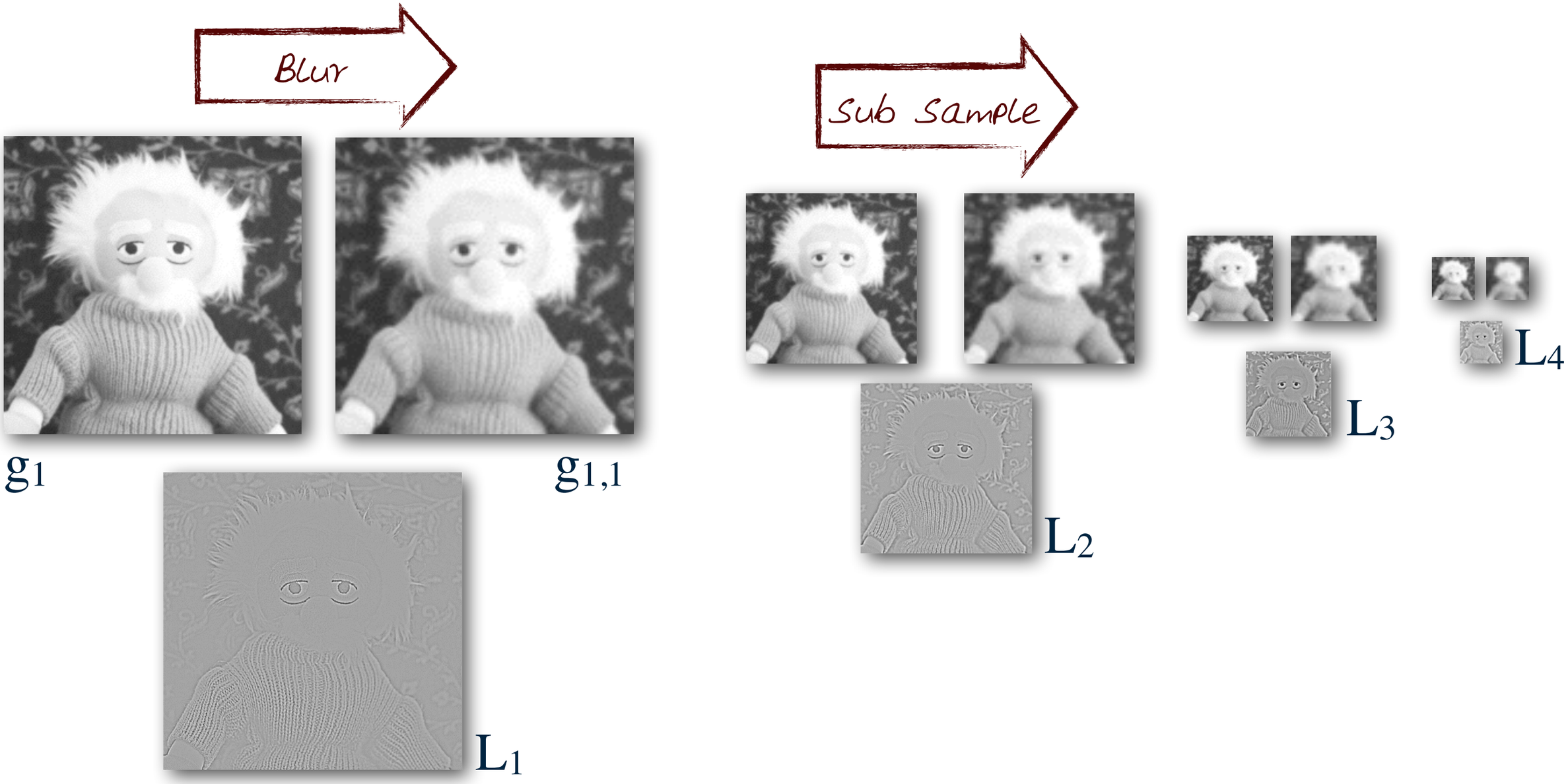
Pyramid Representation of Images (A Laplacian Pyramid)



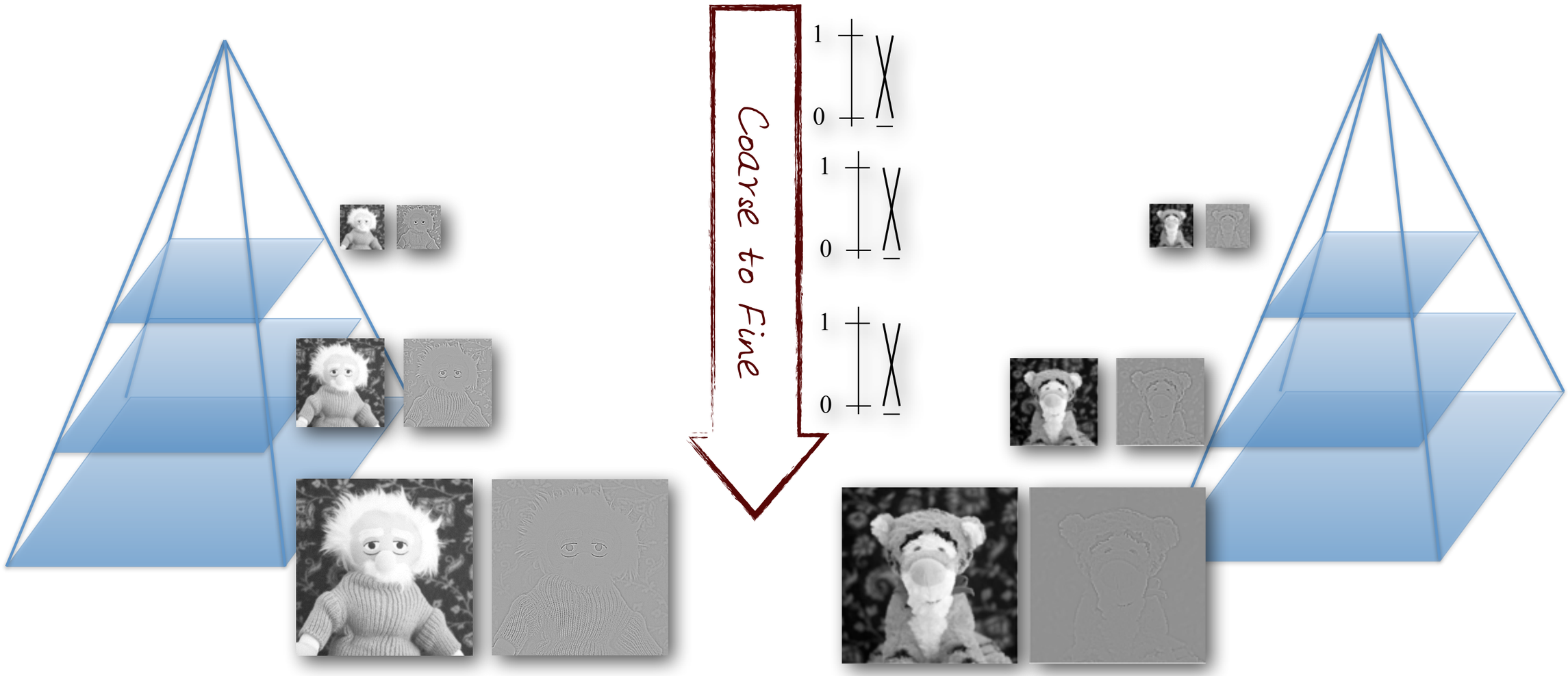
$$L_l = g_l - \text{EXPAND}(g_{l+1})$$

* A series of "error" images,
* A difference between two levels of a Gaussian Pyramid

Computing Gaussian and Laplacian Pyramids



Pyramid Blending



Blend 1

Pyramid Blending Process

A

B

R

* Build Laplacian pyramids
* Build a Gaussian pyramid from selected region R
* Form a combined pyramid using $\mathbf{G_R}$ as weights:
  * $L_O(i,j) = G_R(i,j,)* L_A(i,j) + (1-G_R(i,j))* L_B(i,j)$
* Collapse the L_O pyramid to get the final blended image

Blend 2

Summary

* Merge two images leveraging the Frequency Domain
* Gaussian and the Laplacian Pyramids
* mathematical formulation to compute a Laplacian
* Blend two images using Pyramids

Next Class

* Merging and Blending of Images:
  Cutting Images

Further Reading

- Richard Szeliski (2010) Computer Vision: Algorithms and Applications, Springer.
- Burt and Adelson (1983) "The Laplacian Pyramid as a Compact Image Code", In IEEE Transactions on Communications, 31 (4). p 532-540. 1983 (DOI)
- Burt and Adelson (1983) "A multiresolution spline with application to image mosaics". In ACM Transactions on Graphics, 2 (4). 1983 (DOI)
- Look for "pyramids" on OpenCV and matlab sites

Credits

- For more information, see
  - Richard Szeliski (2010) Computer Vision: Algorithms and Applications, Springer.
- Some concepts in slides motivated by similar slides by A. Efros and J. Hays.
- Some images retrieved from
  - List will be available on website.
  - by Irfan Essa

Computational Photography

* Study the basics of computation and its impact on the entire workflow of photography, from capturing, manipulating and collaborating on, and sharing photographs.

Georgia Tech College of Computing

Computational Photography

* Study the basics of computation and its impact on the entire workflow of photography, from capturing, manipulating and collaborating on, and sharing photographs.

Georgia Tech College of Computing

Digital Images: Cutting Images for Merging

* Methods for Combing Multiple Images by Cutting to Generate a Novel Image

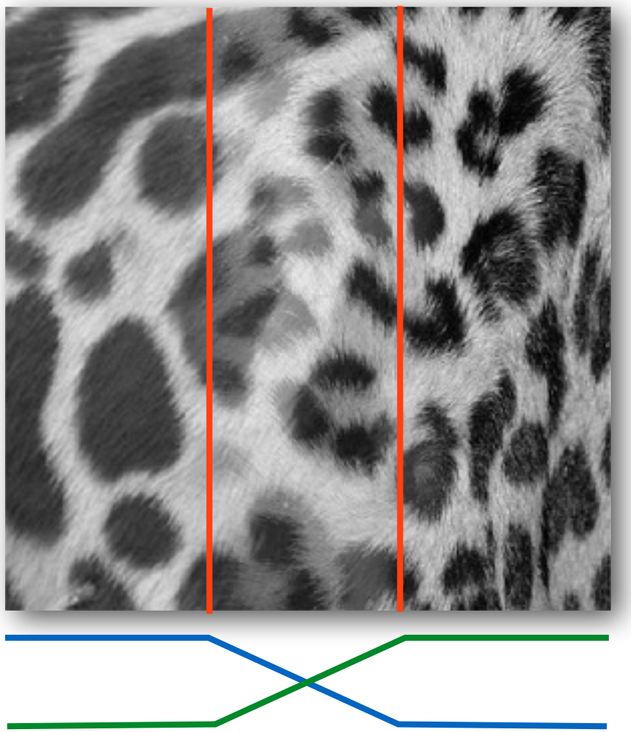
Georgia Tech College of Computing

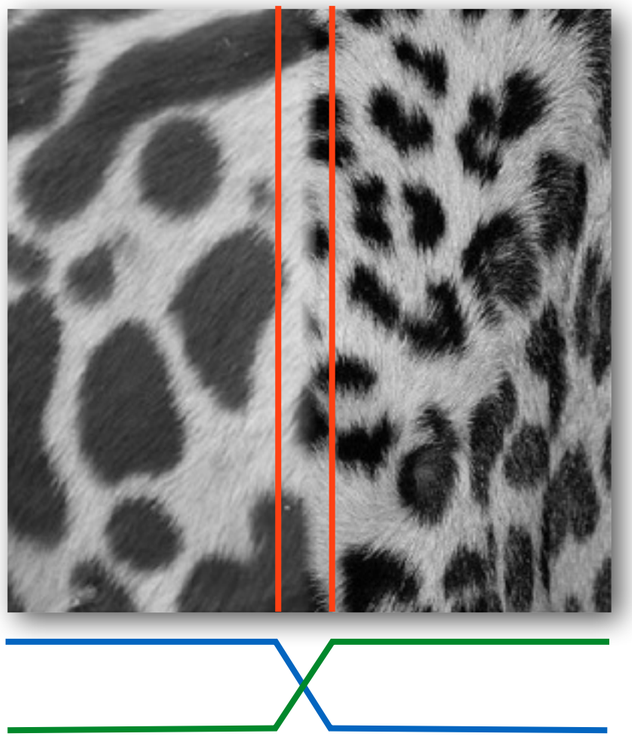
Lesson Objectives

1. An additional method for merging images besides blending
2. Finding seams in images
3. Benefits of cutting images over blending images

Recall: Combine, Merge, Blend Images

Recall: Cross-Fading Window Size



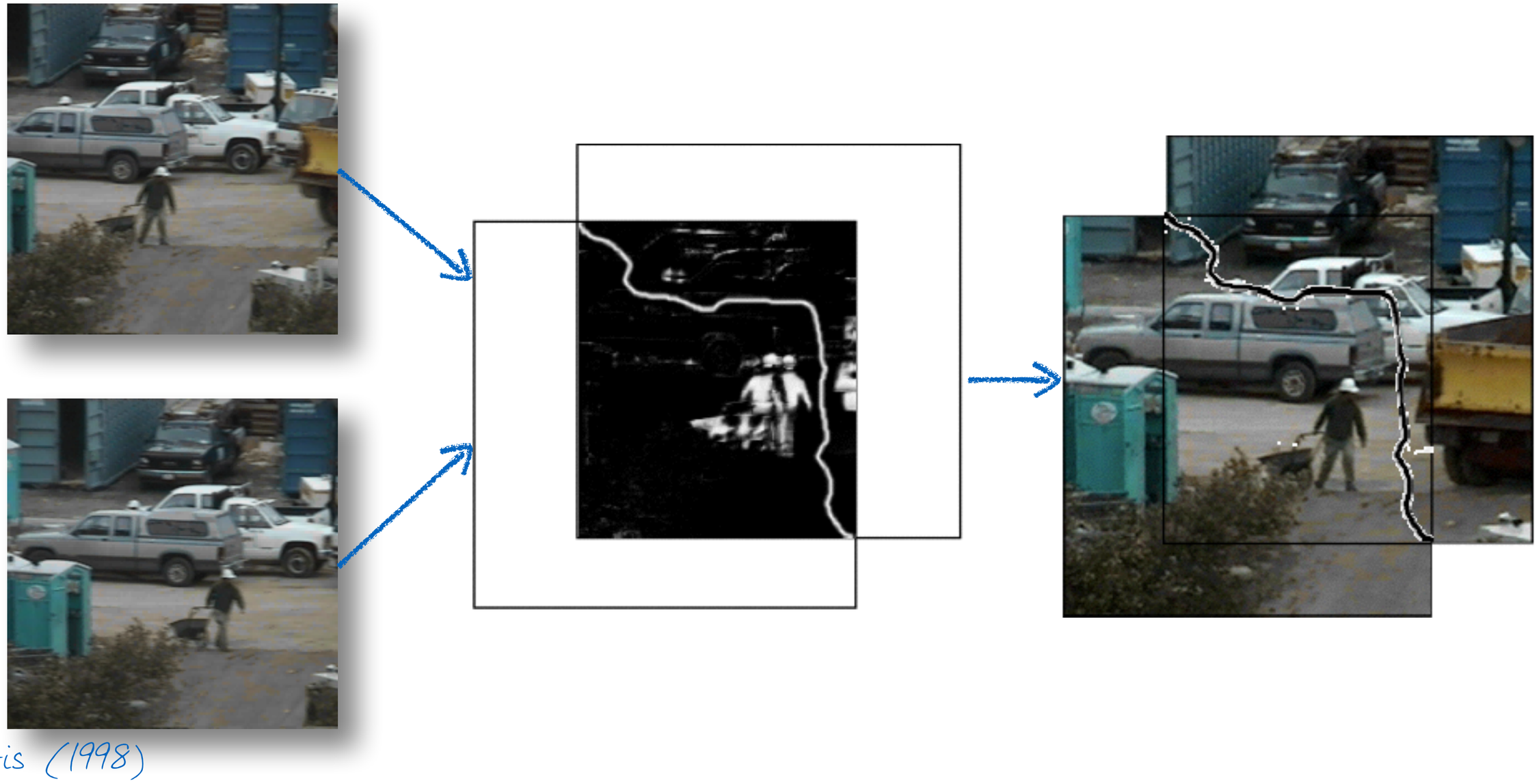


Cut, Don't Blend!

Cut, Don't Blend

Davis (1998)

Cut, Don't Blend

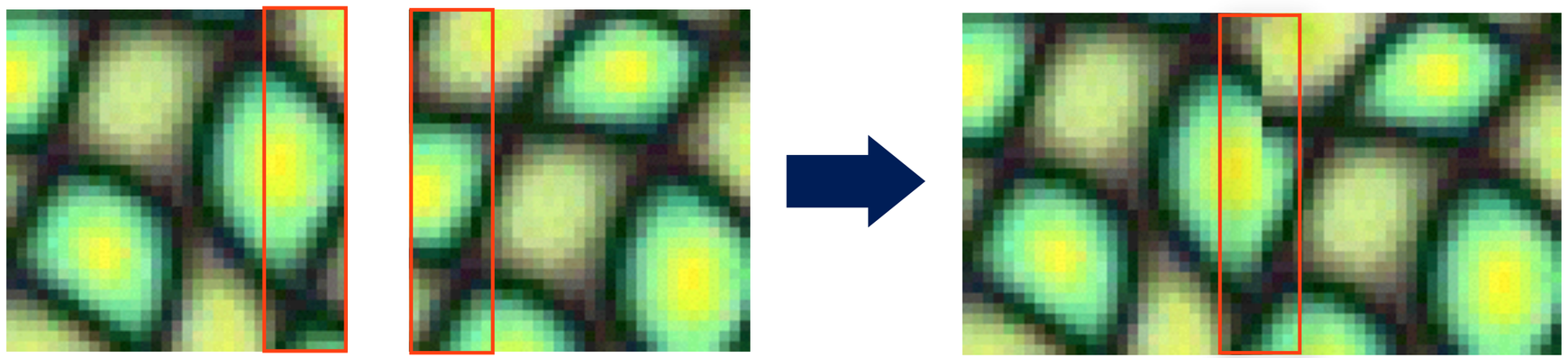


Davis (1998)

Davis (1998)

- Moving objects cause "ghosting"
- Find an optimal seam as opposed to blend between images
- Final has exact pixels from an Image

Finding the Seams

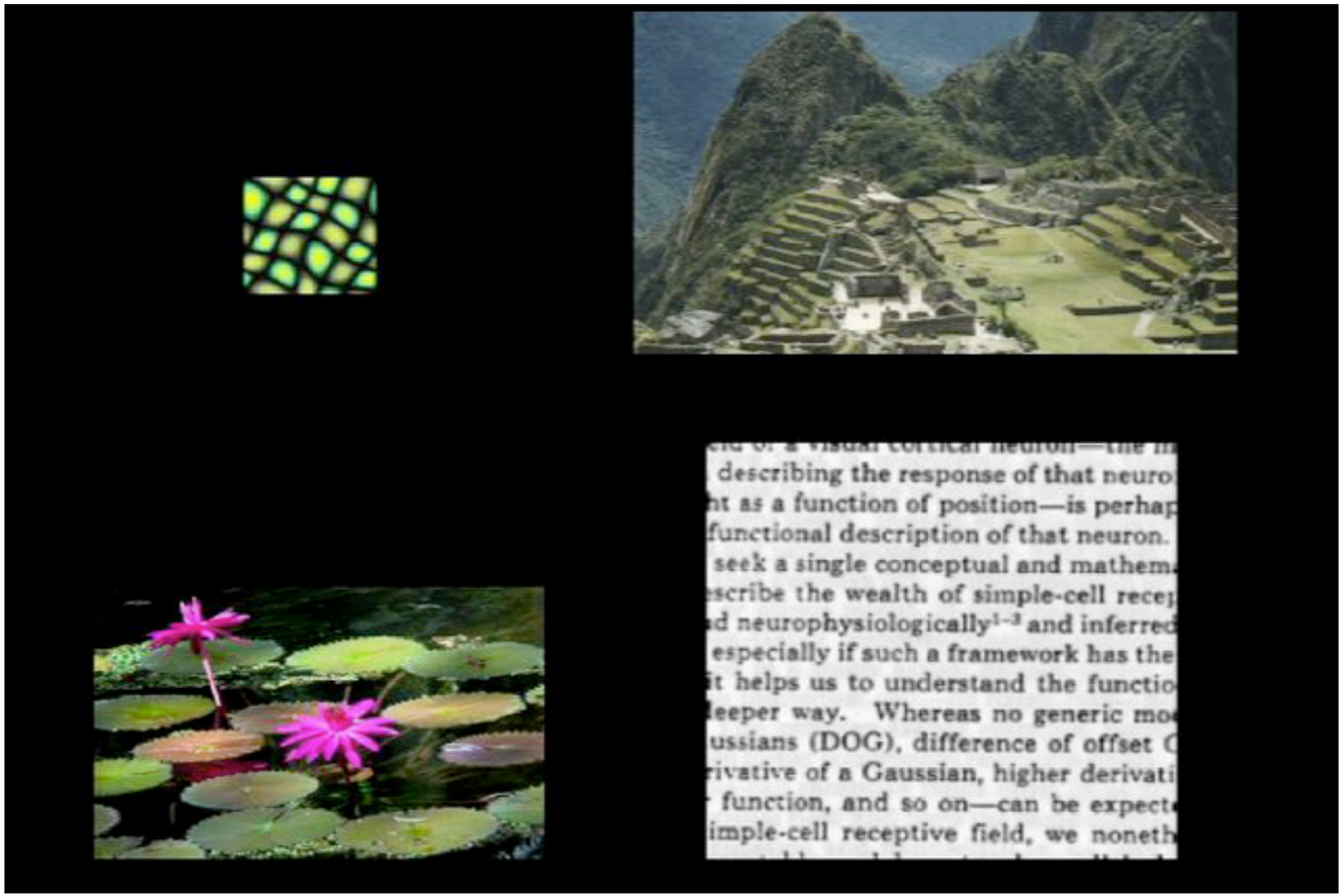


Efros and Freeman (2001)

Finding Seams

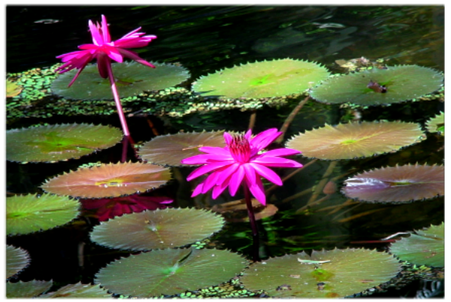
Kwatra et al. (2003)

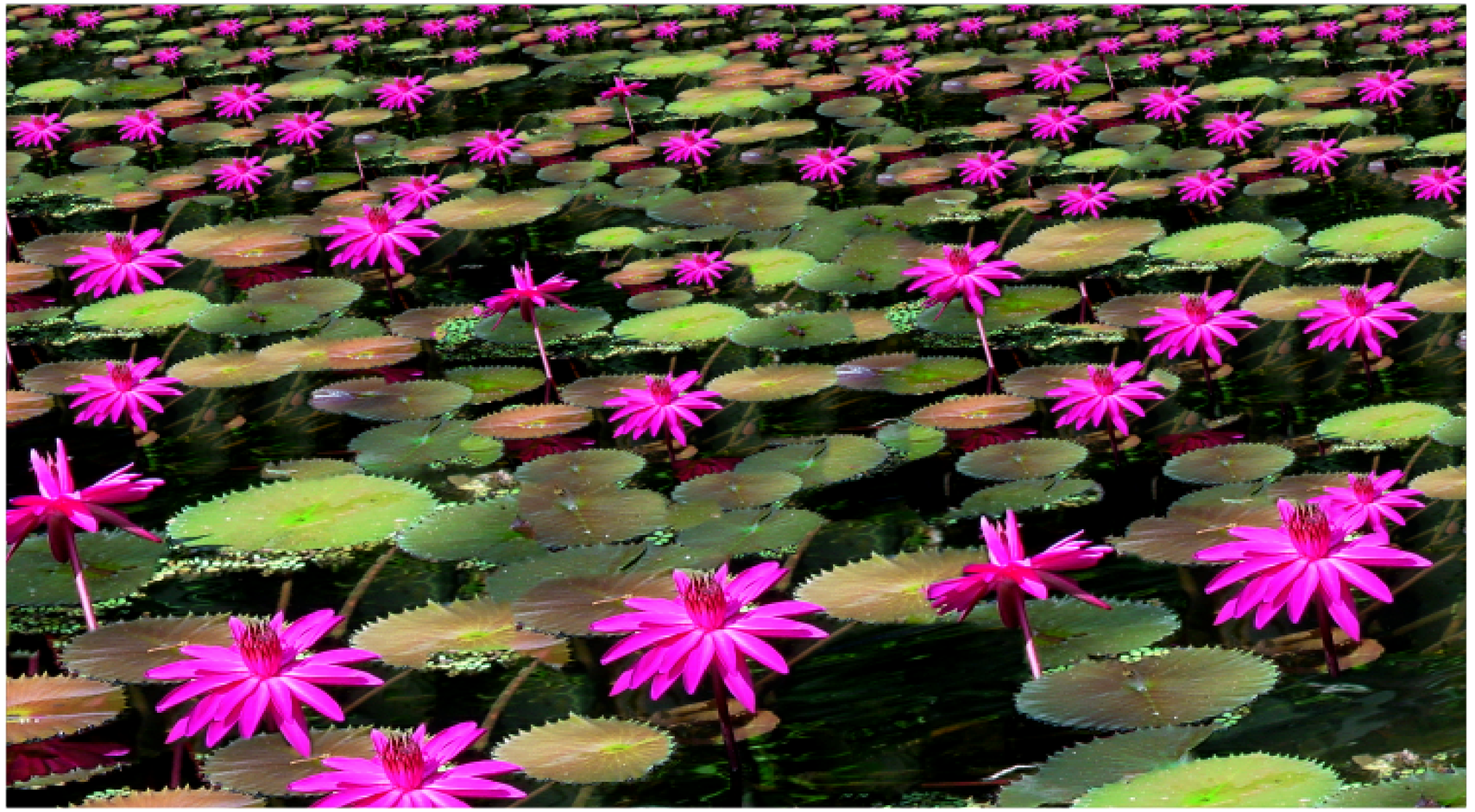
Generating Novel Images



Kwatra et al. (2003)

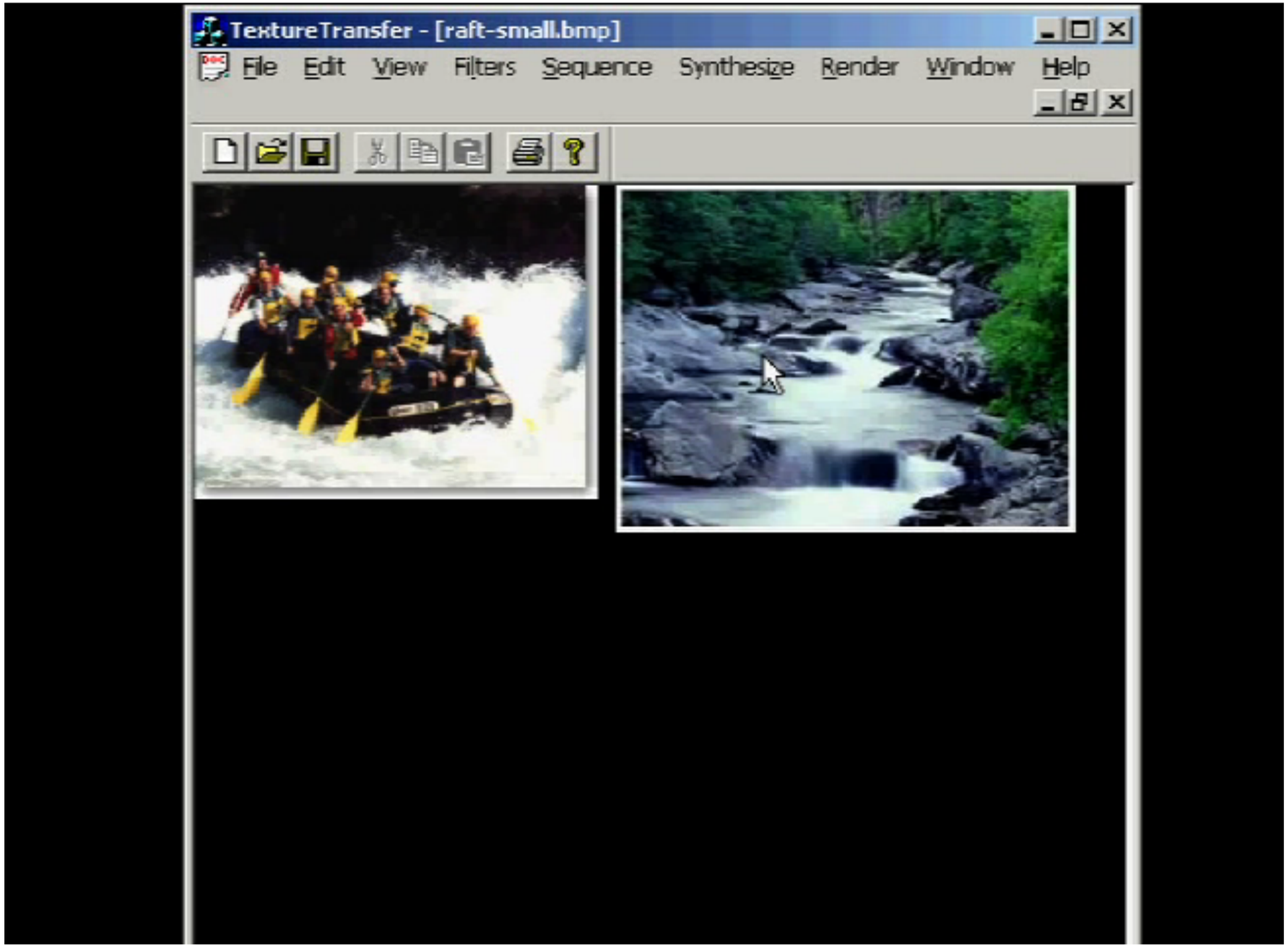
Extending Images





Kwatra et al. (2003)

Editing Images

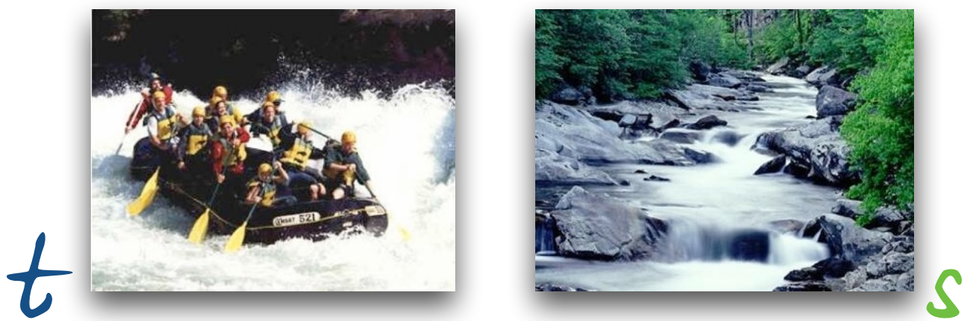


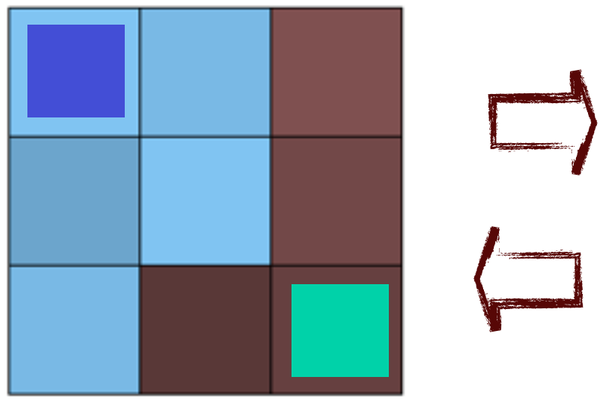
Kwatra et al. (2003)

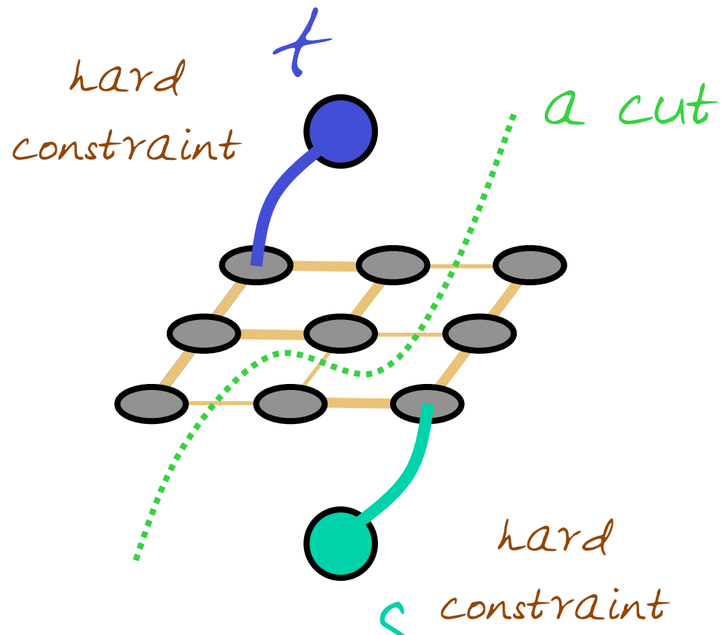
Editing Images

Kwatra et al. (2003)

Seam Finding using Graph-cuts





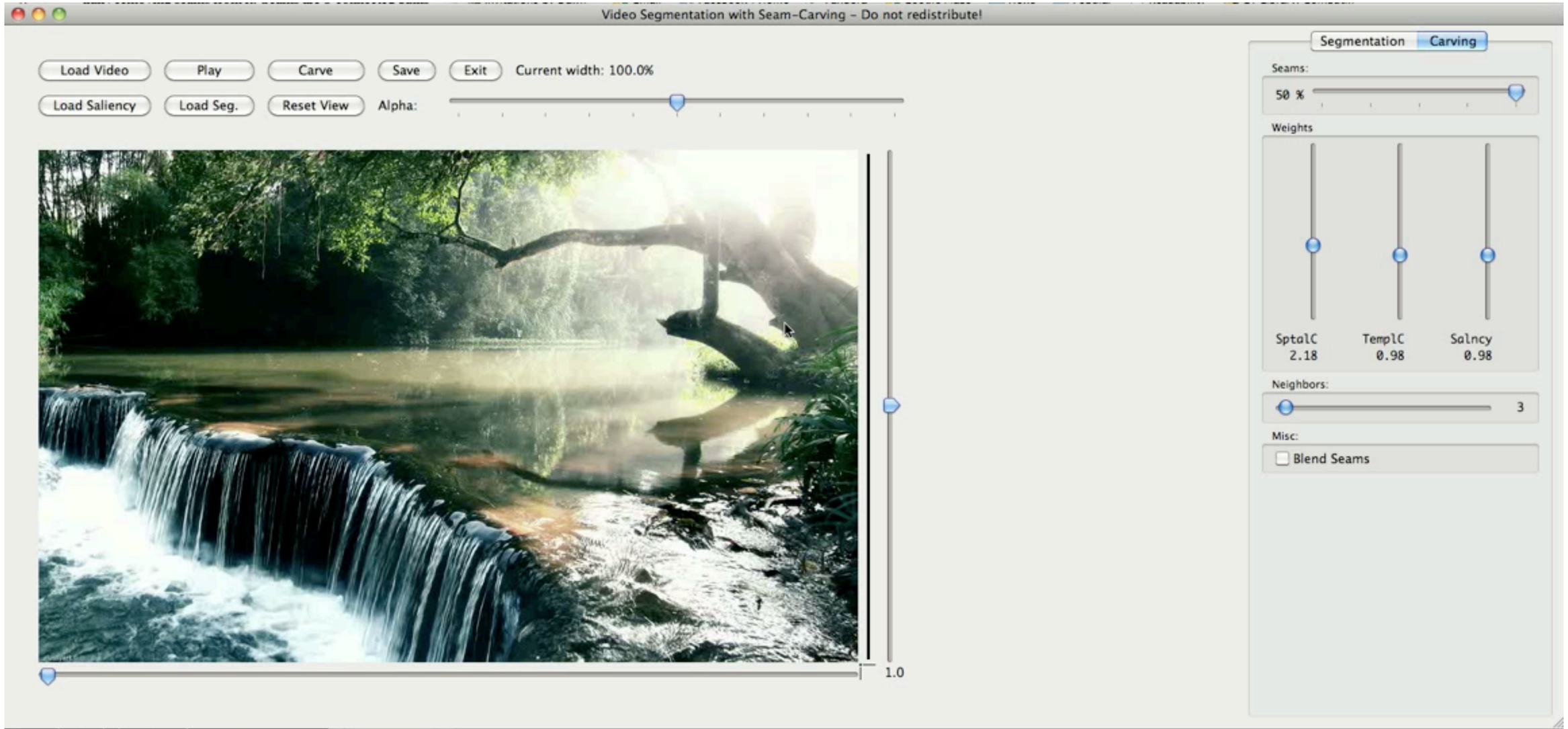


Minimum cost cut can be computed in polynomial time (max-flow/min-cut algorithms)

Another approach is to use Dynamic Programing to find seams

Boykov and Jolly (2001) & Kwatra et al. (2003)

Seam Carving



Grundmann et al. (2010), Avidan and Shamir (2007)

Summary

* Introduced an additional method for merging images besides blending
* Described how seams are found in images
* Discussed the benefits of cutting images over blending images

Further Reading

- Davis (1998), "Mosaics of Scenes with Moving Objects" IEEE Computer Vision and Pattern Recognition (CVPR),1998.
- Efros and Freeman (2001), "Image Quilting for Texture Synthesis and Transfer" SIGGRAPH 2001
- Kwatra, Schödl, Essa, Turk, Bobick (2003), "Graphcut textures: image and video synthesis using graph cuts" SIGGRAPH 2003
- Boykov and Jolly (2001), "Interactive Graph Cuts for Optimal Boundary & Region Segmentation of Objects in N-D images, ICCV 2001.
- Avidan and Shamir (2007), "Seam carving for content-aware image resizing", SIGGRAPH 2007.
- Agarwala, Dontcheva, Agrawala, Drucker, Colburn, Curless, Salesin, Cohen (2004), "Interactive Digital Photomontage" SIGGRAPH 2004.

Next Class

* Feature Detection and matching

Credits

- For more information, see
  - Richard Szeliski (2010) Computer Vision: Algorithms and Applications, Springer.
- Some concepts in slides motivated by similar slides by A. Efros and J. Hays.
- Some images retrieved from
  - http://commons.wikimedia.org/.
  - List will be available on website.

Computational Photography

* Study the basics of computation and its impact on the entire workflow of photography, from capturing, manipulating and collaborating on, and sharing photographs.

Georgia Tech College of Computing

Computational Photography

* Study the basics of computation and its impact on the entire workflow of photography, from capturing, manipulating and collaborating on, and sharing photographs.

Georgia Tech College of Computing

Feature Detection and Matching

* Methods for Detecting Features in Images that can be matched for Registration and Alignment.

Georgia Tech College of Computing

1. Benefits of Feature Detection and matching in images
2. Characteristics of Good Features
3. Corners are Good Features
4. Harris Corner Detector Algorithm
5. Stages of a SIFT detector

Recall: Detection and Matching

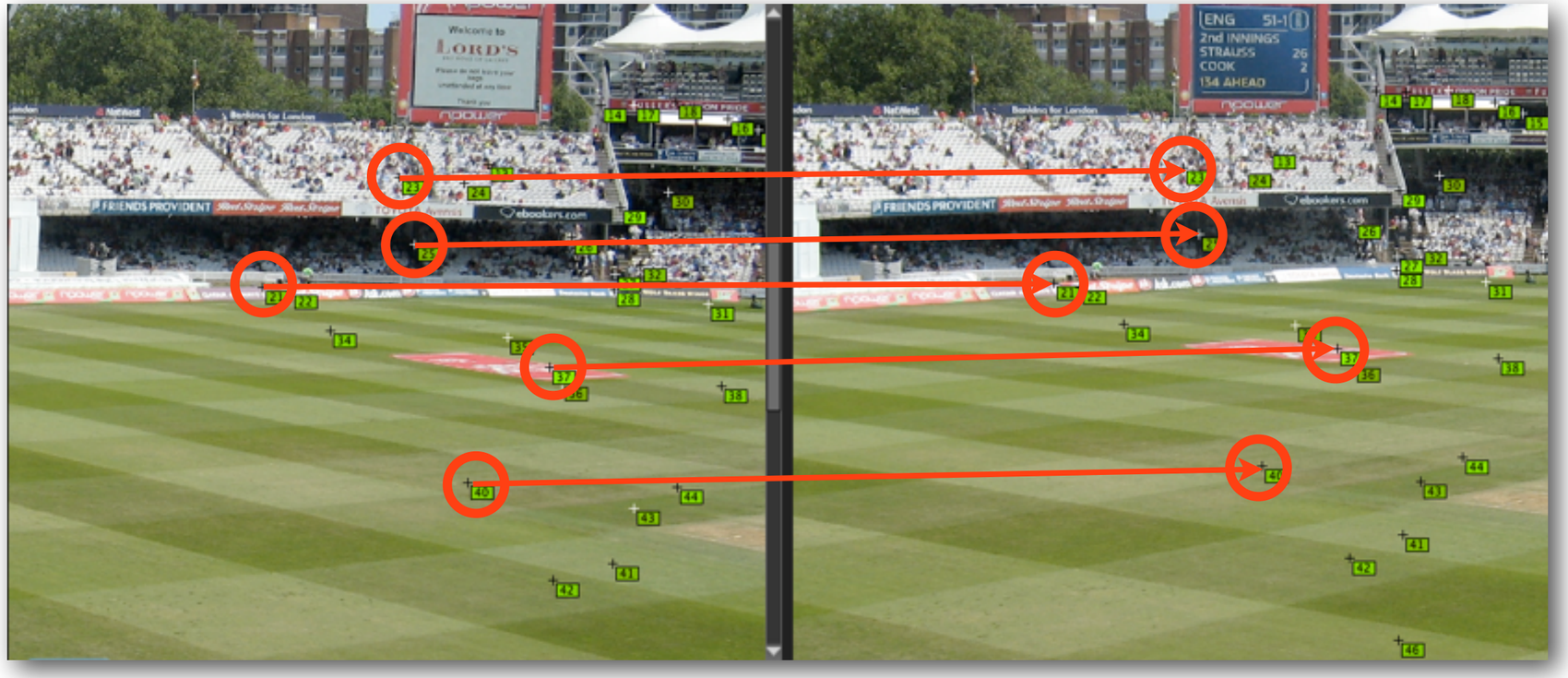
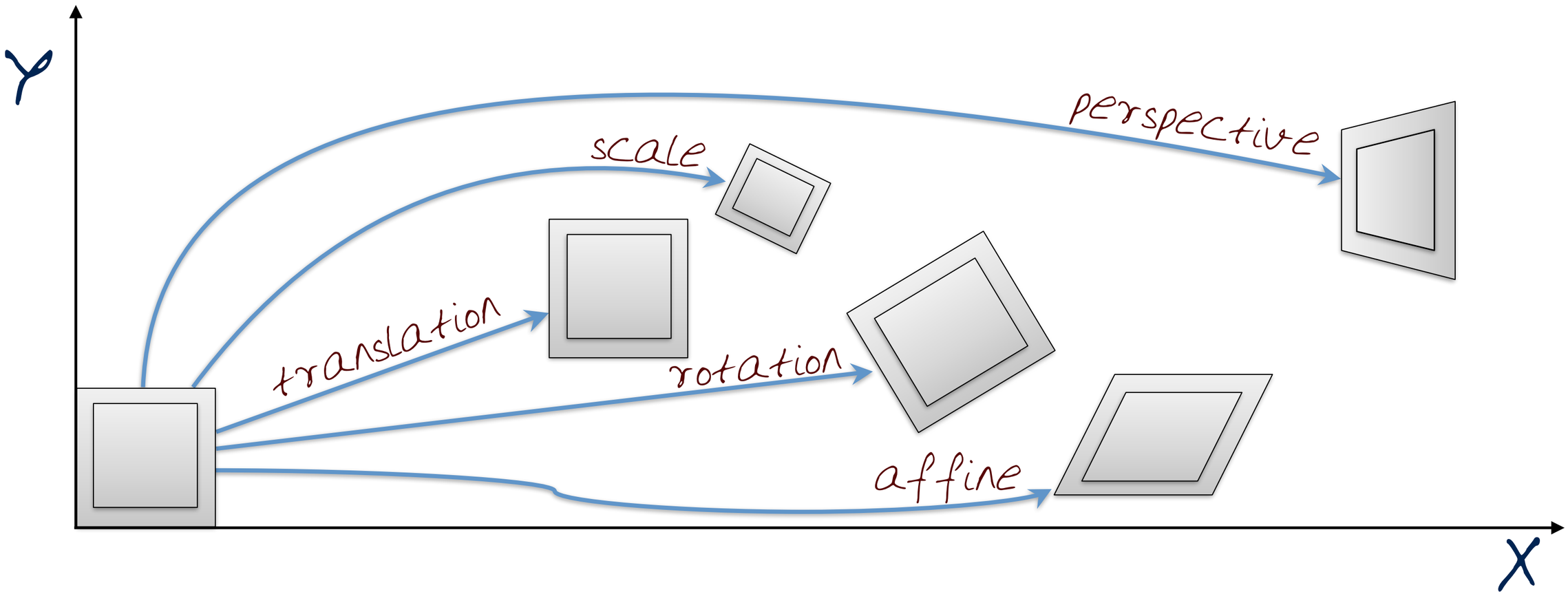
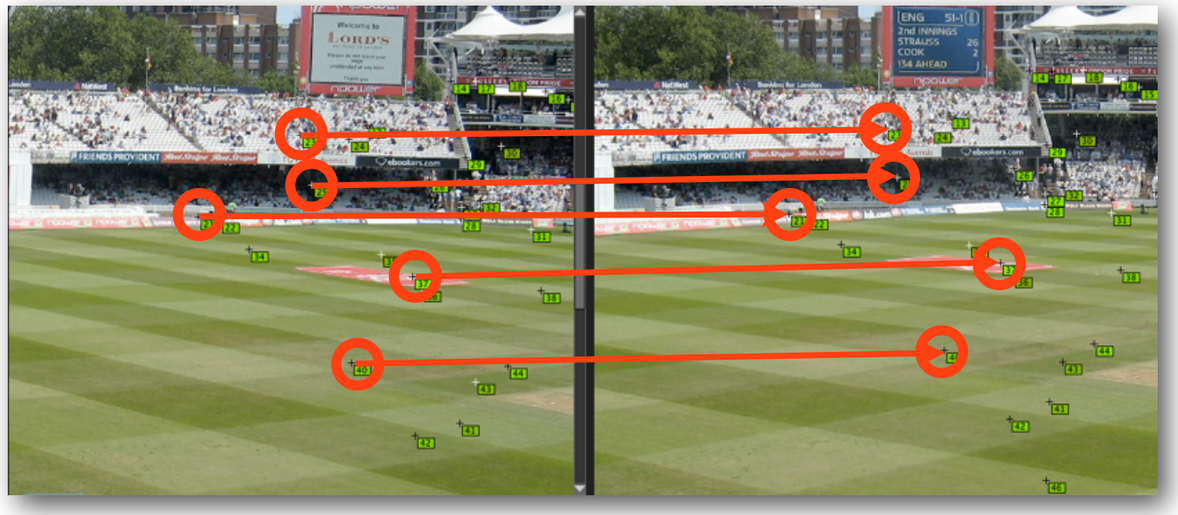


Image Matching



Finding Features

* Goal - Find points in an image that can be:
  * Found in other images
  * Found precisely - well localized
  * Found reliably - well matched



Characteristics of Good Features

* Repeatability/Precision
* Saliency/Matchability
* Compactness and efficiency
* Locality

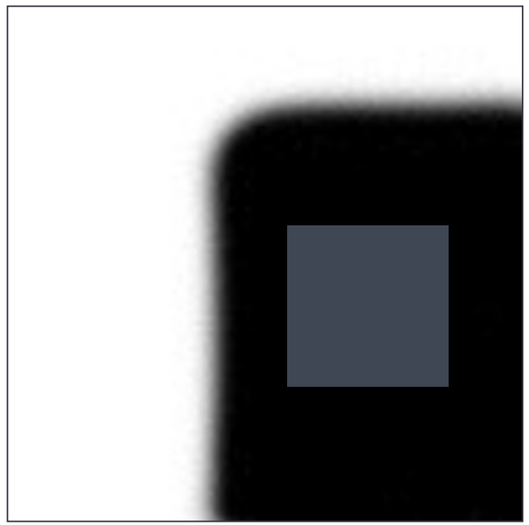
Slide motivated by Aaron Bobick

Find Corners

* Key property: In the region around a corner, image gradient has two or more dominant directions
* Corners are repeatable and distinctive

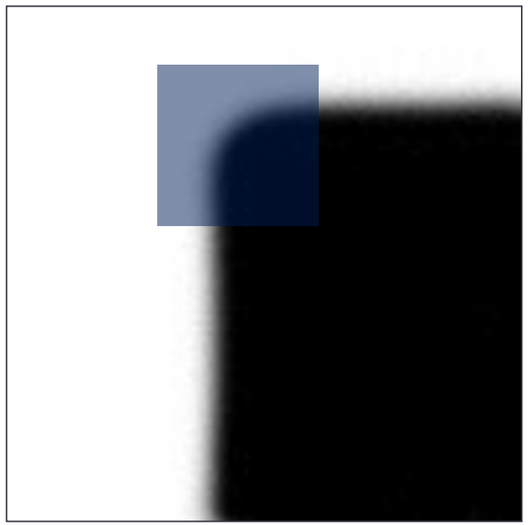
Harris and Stephens (1988)

Corner Detection: The Basics



"flat"
No change in any directions

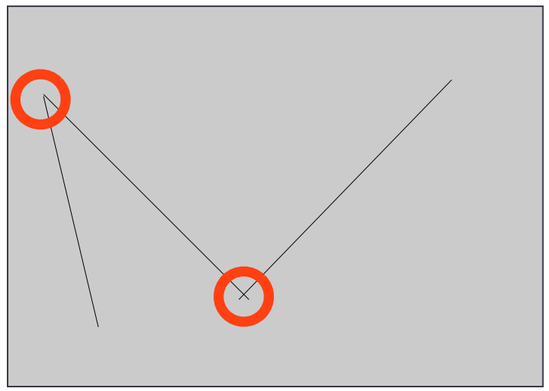
"edge"
No change along the edge direction

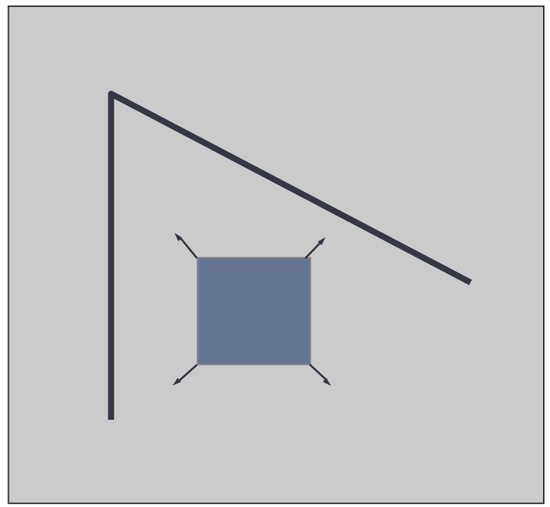


"corner"
Change in all directions

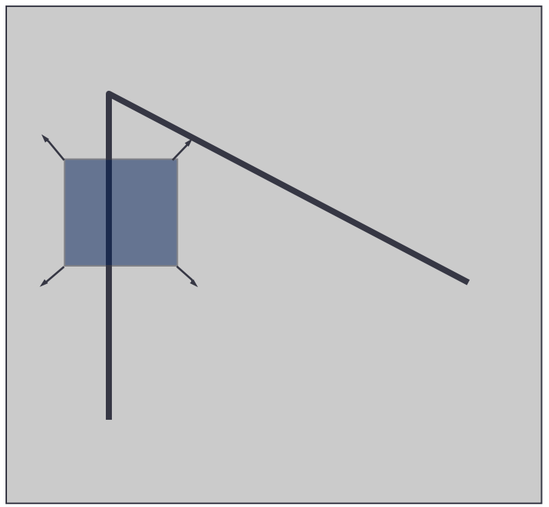
* Recognize a point by looking through a small window
* Shifting a window in any direction causes a large change in intensity

Corner Detection: The Basics

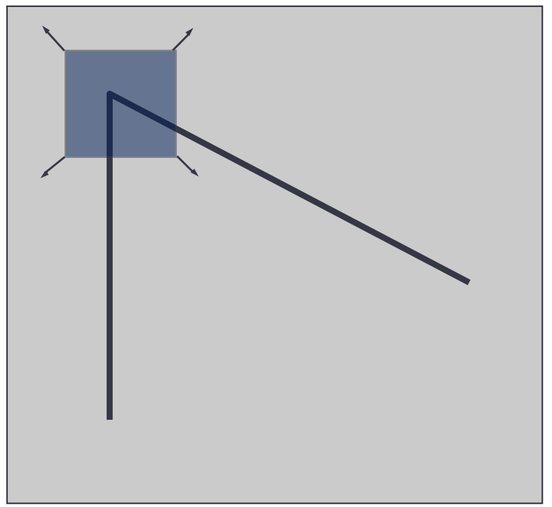




"flat"
No change in any directions



"edge"
No change along the edge direction



"corner"
Change in all directions

Harris and Stephens (1988)

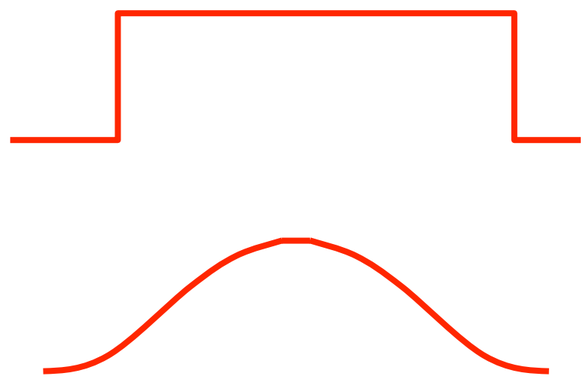
Corner Detection: Mathematics

Compute the change in appearance by shifting the window by u,v:

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

- $E(u,v)$: change in appearance
- $w(x,y)$: window function
- $I(x+u, y+v)$: shifted intensity
- $I(x,y)$: intensity

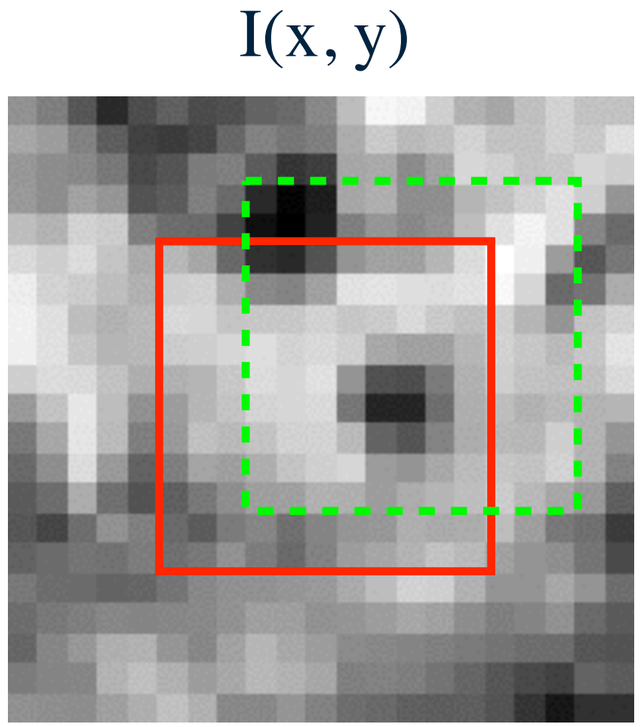
$w(x,y)$ {
- box function
- a Gaussian

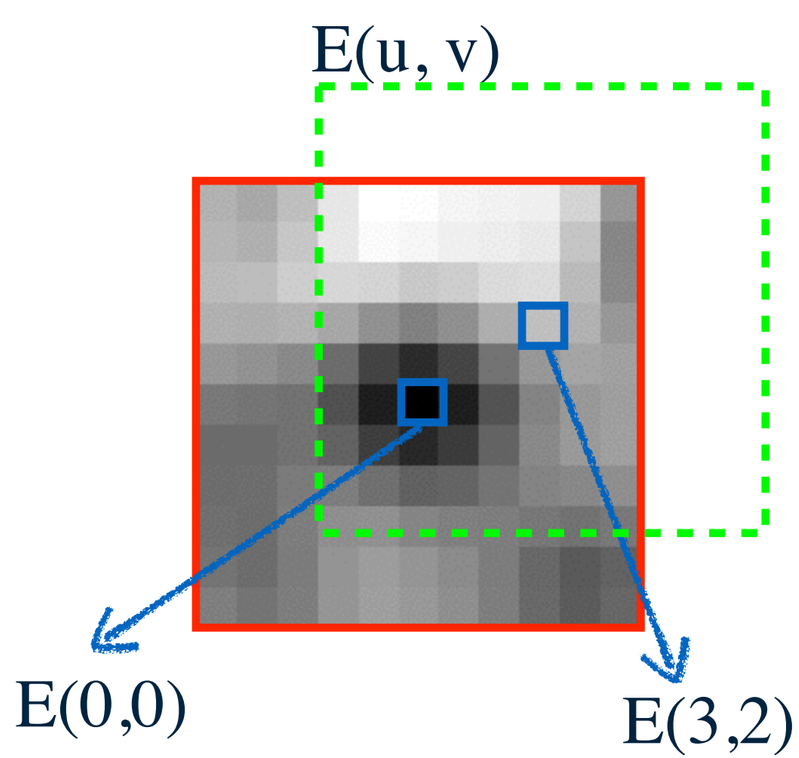


Corner Detection: Mathematics

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

Computation of the change in appearance by shifting the window by u,v:





Slide motivated by Alyosha Efros

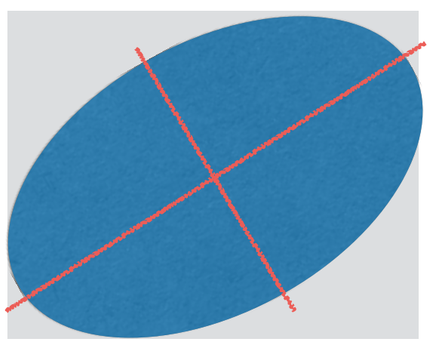
Corner Detection: Mathematics

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

The quadratic approximation, following Taylor Expansion, simplifies to:

$$E(u,v) \approx \begin{bmatrix} u & v \end{bmatrix} M \begin{bmatrix} u \\ v \end{bmatrix}$$

where $\boldsymbol{M}$ is a second moment matrix computed from image derivatives $\boldsymbol{I_x}$ and $\mathbf{I_y}$:



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

Corner Detection: Mathematics

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

$$E(u,v) = \sum_{x,y} \omega(x,y) \left(u^2 I_x^2 + 2uv I_x I_y + v^2 I_y^2\right)$$

The surface E(u,v) is locally approximated by a quadratic form

Corner Detection: Mathematics

$$E(u,v) = \sum_{x,y} \omega(x,y) \left(u^2 I_x^2 + 2uvI_xI_y + v^2 I_y^2 \right)$$

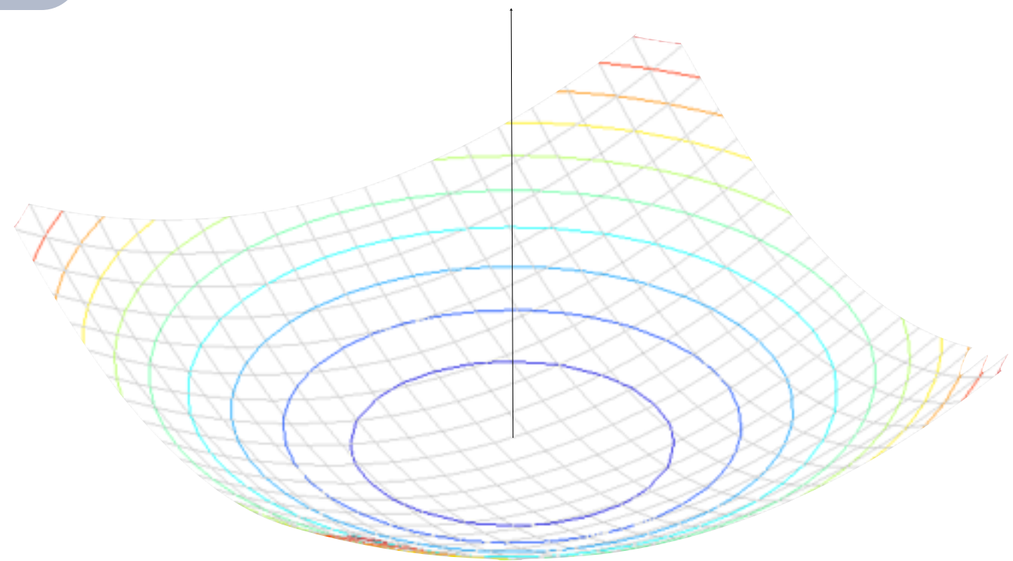
The surface E(u,v) is locally approximated by a quadratic form.

$$E(u,v) = \sum_{x,y} \omega(x,y) \left(u^2 I_x^2 + 2uv I_x I_y + v^2 I_y^2\right)$$

consider a "slice" :

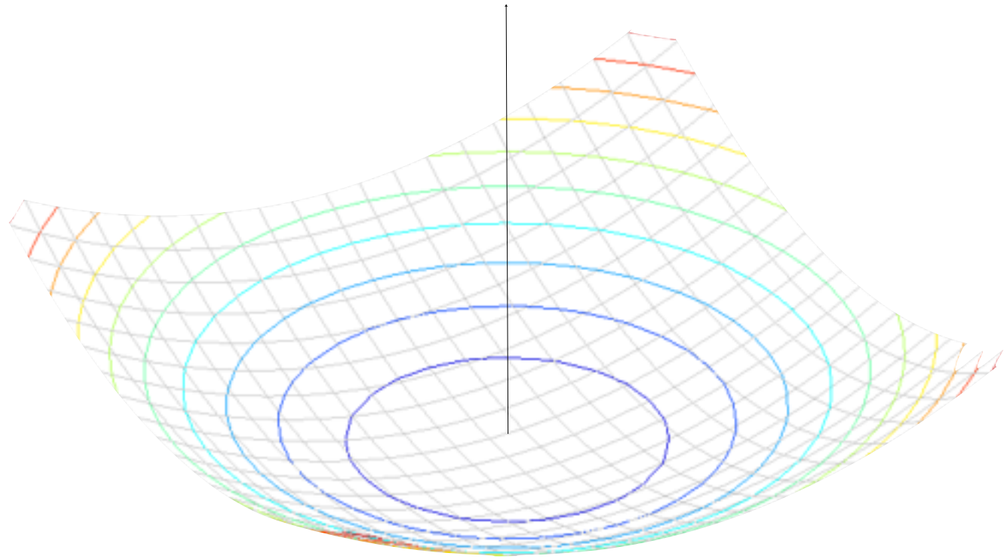
$$u^2 I_x^2 + 2uv I_x I_y + v^2 I_y^2 = k$$

which is an equation of an ellipse



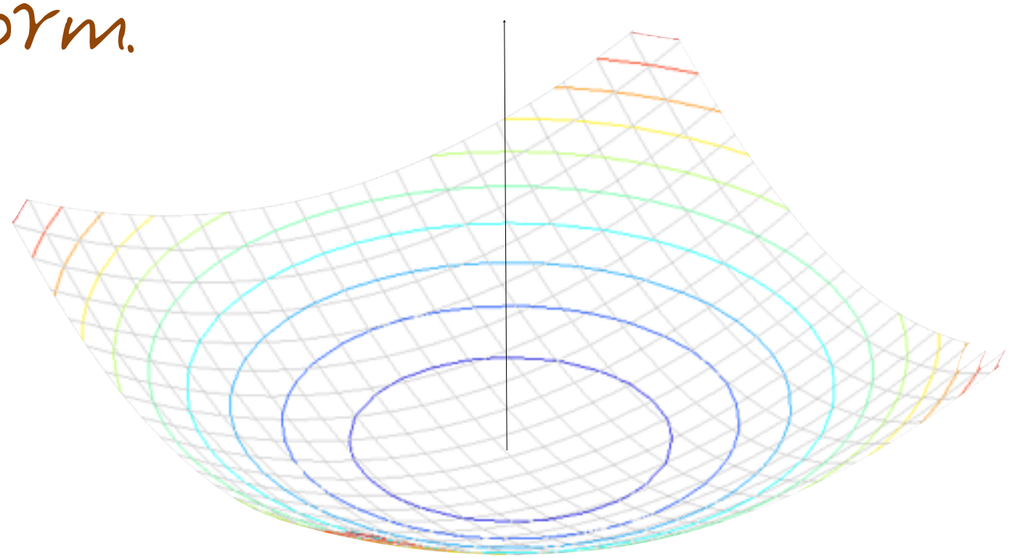
The surface E(u,v) is locally approximated by a quadratic form.

$$u^2 I_x^2 + 2uvI_xI_y + v^2 I_y^2 = k$$



The surface E(u,v) is locally approximated by a quadratic form.

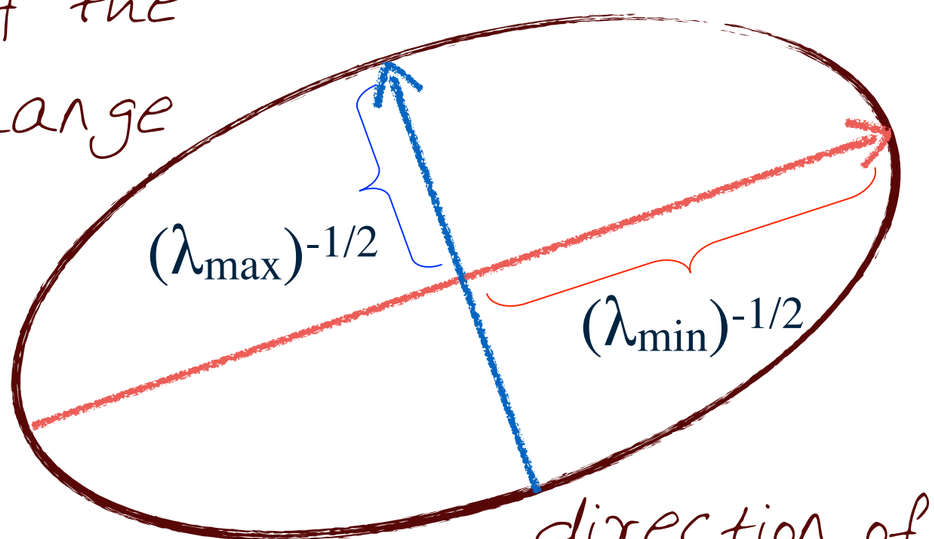
$$u^2 I_x^2 + 2uvI_xI_y + v^2 I_y^2 = k$$



Eigenvalue Analysis:

- Axis lengths ⇒ Eigenvalues
- Orientation ⇒ Eigenvectors

direction of the fastest change



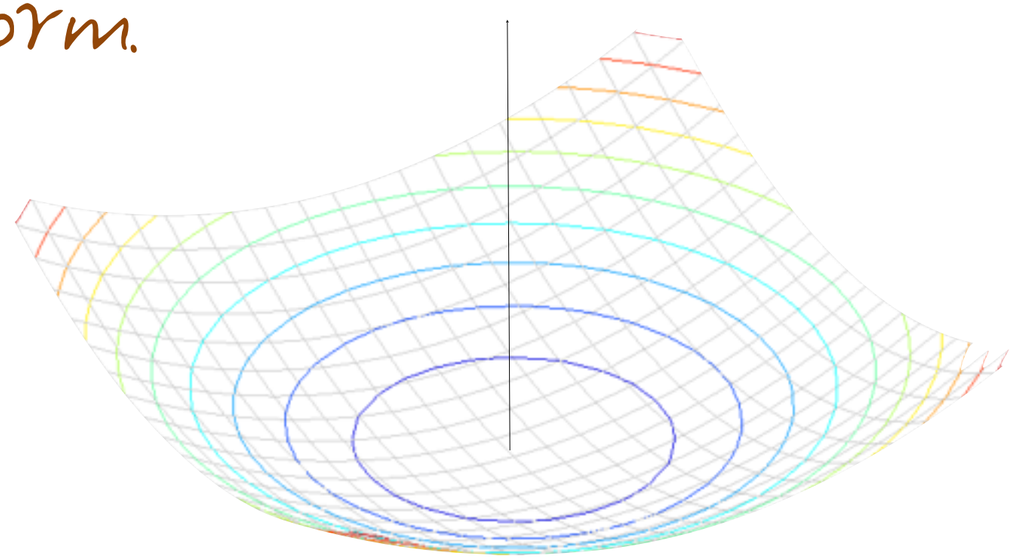
direction of the slowest change

Slide motivated by Aaron Bobick

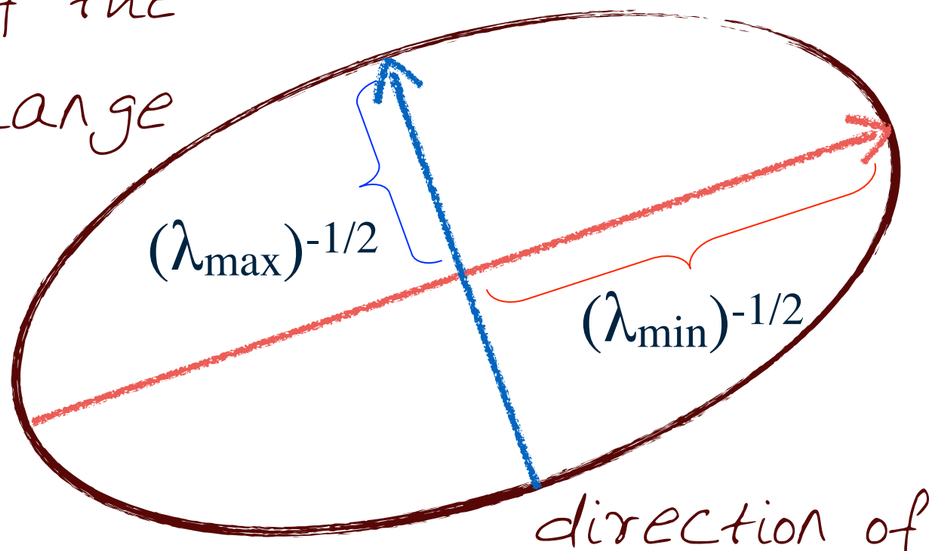
The surface E(u,v) is locally approximated by a quadratic form.

$$u^2 I_x^2 + 2uvI_xI_y + v^2 I_y^2 = k$$

$$\begin{bmatrix} u & v \end{bmatrix} M \begin{bmatrix} u \\ v \end{bmatrix} = k$$



direction of the fastest change



direction of the slowest change

M is a diagonal matrix

$$M = R^{-1} \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} R$$

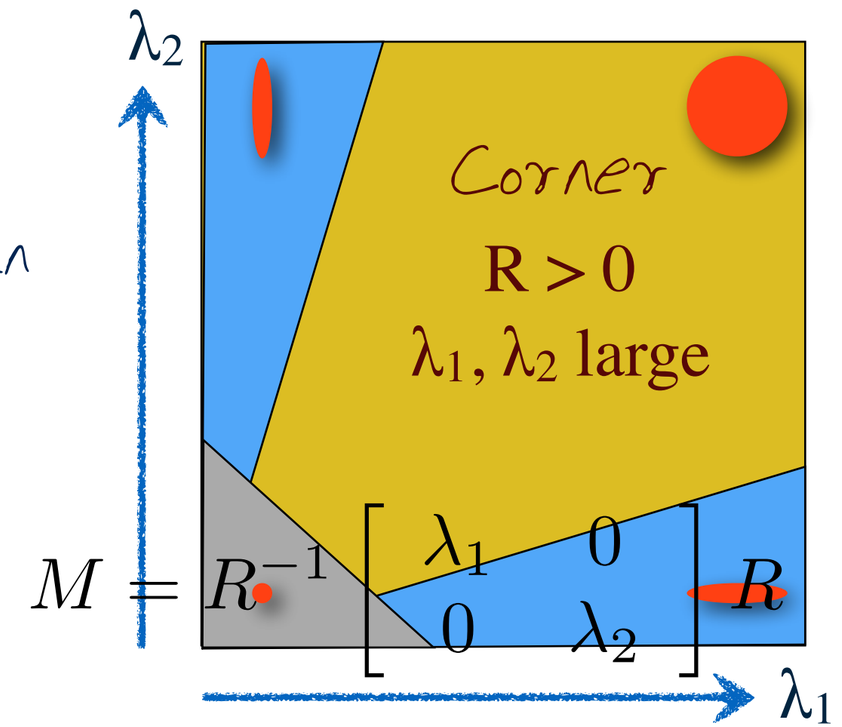
Slide motivated by Aaron Bobick

Eigenvalues

$$R = \det(M) - \alpha\ \mathrm{trace}(M)^2 = \lambda_1\lambda_2 - \alpha(\lambda_1 + \lambda_2)^2$$

α: constant (0.04 to 0.06)

* R depends only on eigenvalues of M
* $R \Rightarrow$ large for a corner
* $R \Rightarrow$ negative with large magnitude for an edge
* $|R| \Rightarrow$ small for a flat region
* Note: No explicit computation of eigenvalues required



Harris Detector Algorithm (Preview)

* Compute Gaussian derivatives at each pixel
* Compute second moment matrix $\boldsymbol{M}$ in a Gaussian window around each pixel
* Compute corner response function $\boldsymbol{R}$
* Threshold $\boldsymbol{R}$
* Find local maxima of response function (non-maximum suppression)

Properties of the Harris Detector

- Rotation Invariant?
  - Ellipses rotate, but shape (eigenvalues) remain same
  - Corner Response R is invariant
- Intensity Invariant?
  - Partial invariance to additive and multiplicative intensity changes (threshold issue for multiplicative)
  - Only Image derivatives are used
- Scale Invariant
  - NO! Dependent on Window Size!
  - USE Pyramids (or Frequency Domain!)

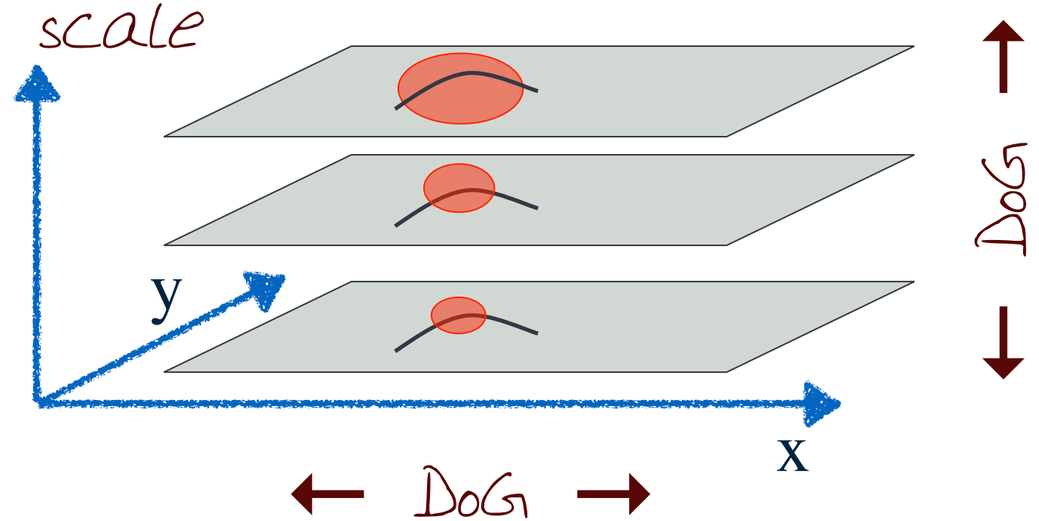
Scale Invariant Detectors

* Harris-Laplacian
* Find local maximum of:
  * Harris corner detector in space (image coordinates)
  * Laplacian in scale

(Mikolajczyk and Schmid, 2001)

Scale Invariant Detectors

* SIFT (Lowe, 2004)
* Find local maximum of:
  * Difference of Gaussians (DoG) in space and scale
  * DoG is simply a pyramid of the difference of Gaussians within each octave



Lowe 2004

SIFT (Scale-Invariant Feature Transform)

* Orientation assignment
  * Compute best orientation(s) for each keypoint region.
* Keypoint description
  * Use local image gradients at selected scale and rotation to describe each keypoint region.

http://en.wikipedia.org/wiki/File:Sift_keypoints_filtering.jpg

Invariant Local Features

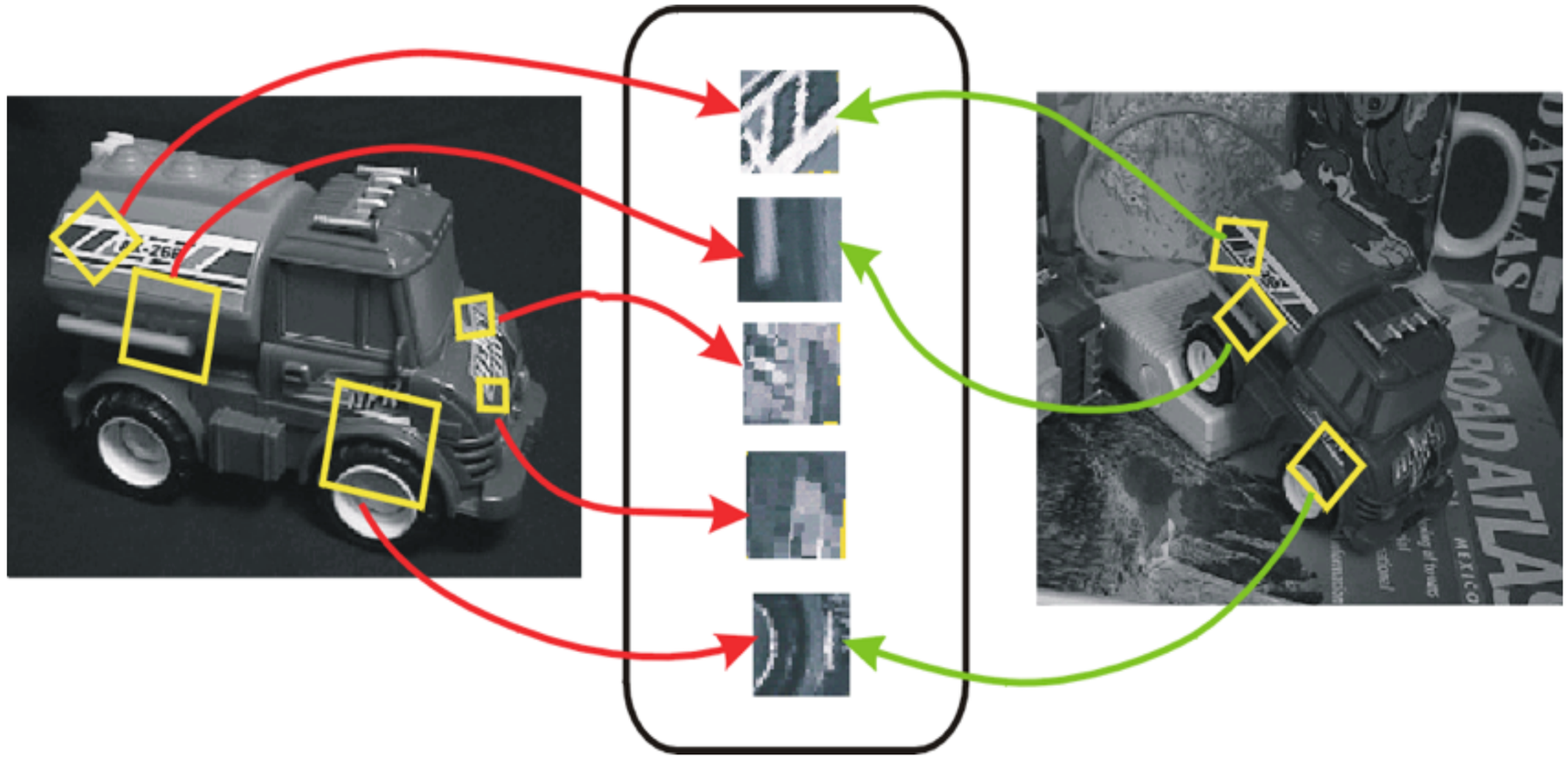


Image content is transformed into local feature coordinates that are invariant to translation, rotation, scale, and other imaging parameters

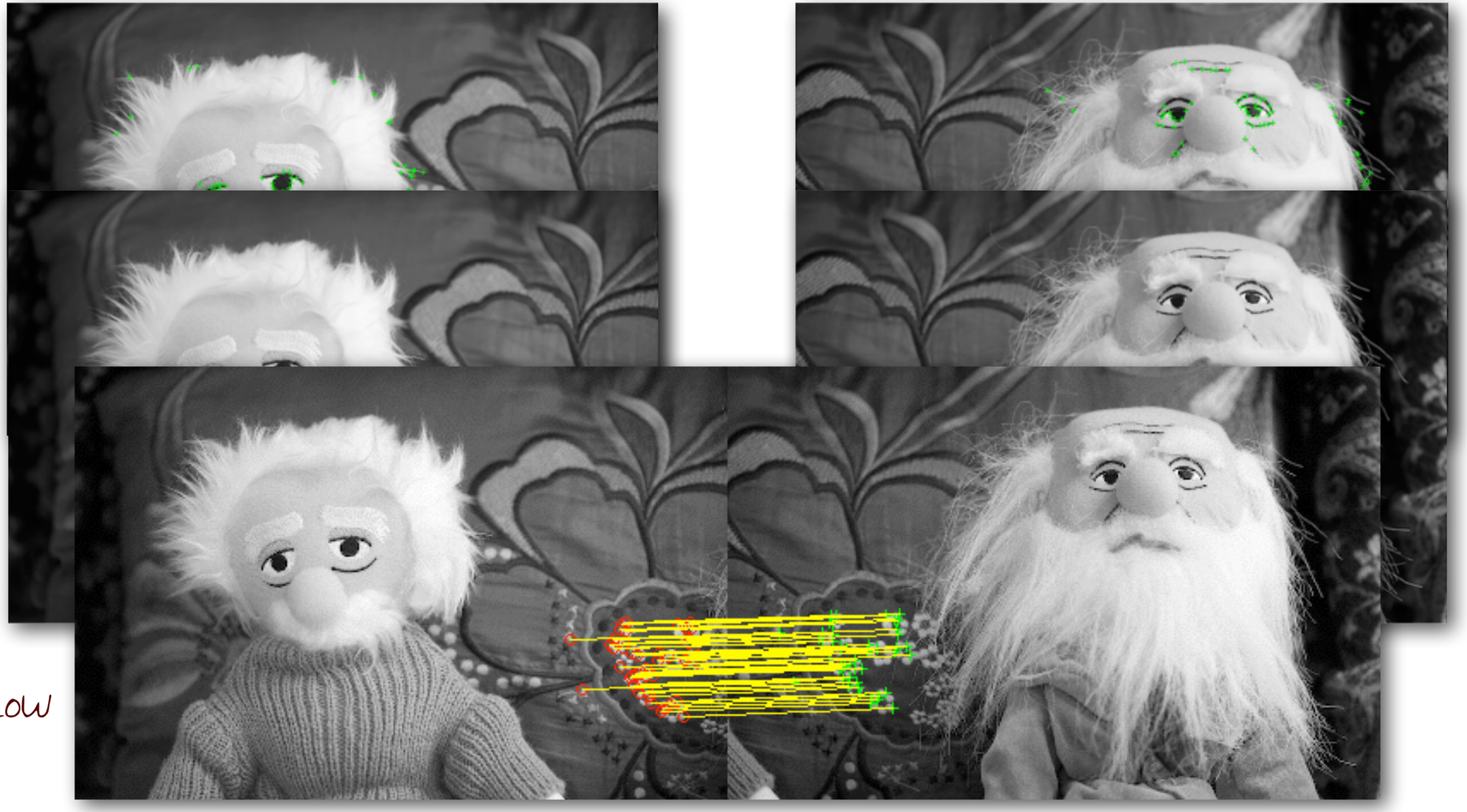
Lowe 2004

Results

Detect

match

Show



Summary

* Introduced Feature Detection and matching for images
* Discussed the four (4) Characteristics of Good Features
* Introduced the Harris Corner Detector Framework
* Introduced the SIFT detector

Further Reading

- Harris and Stephens (1988) "A Combined Corner and Edge Detector." Proceedings of the 4th Alvey Vision Conference, 1988, [PDF][DOI]
- Mikolajczyk and Schmid (2001). "Indexing Based on Scale Invariant Interest Points". ICCV 2001
- Lowe (2004) "Distinctive Image Features from Scale-Invariant Keypoints". IJCV 2004
- Search for "Features" on OpenCV site

Next Class

* More details on SIFT and Harris Corner Detectors!

Credits

* For more information, see:
  * Richard Szeliski (2010) Computer Vision: Algorithms and Applications, Springer
* Some concepts in slides motivated by similar slides by A. Efros and J. Hays
* Some images retrieved from
  * http://commons.wikimedia.org/
  * List will be available on website

Computational Photography

* Study the basics of computation and its impact on the entire workflow of photography, from capturing, manipulating and collaborating on, and sharing photographs.

Georgia Tech College of Computing

Computational Photography

* Study the basics of computation and its impact on the entire workflow of photography, from capturing, manipulating and collaborating on, and sharing photographs.

Georgia Tech College of Computing

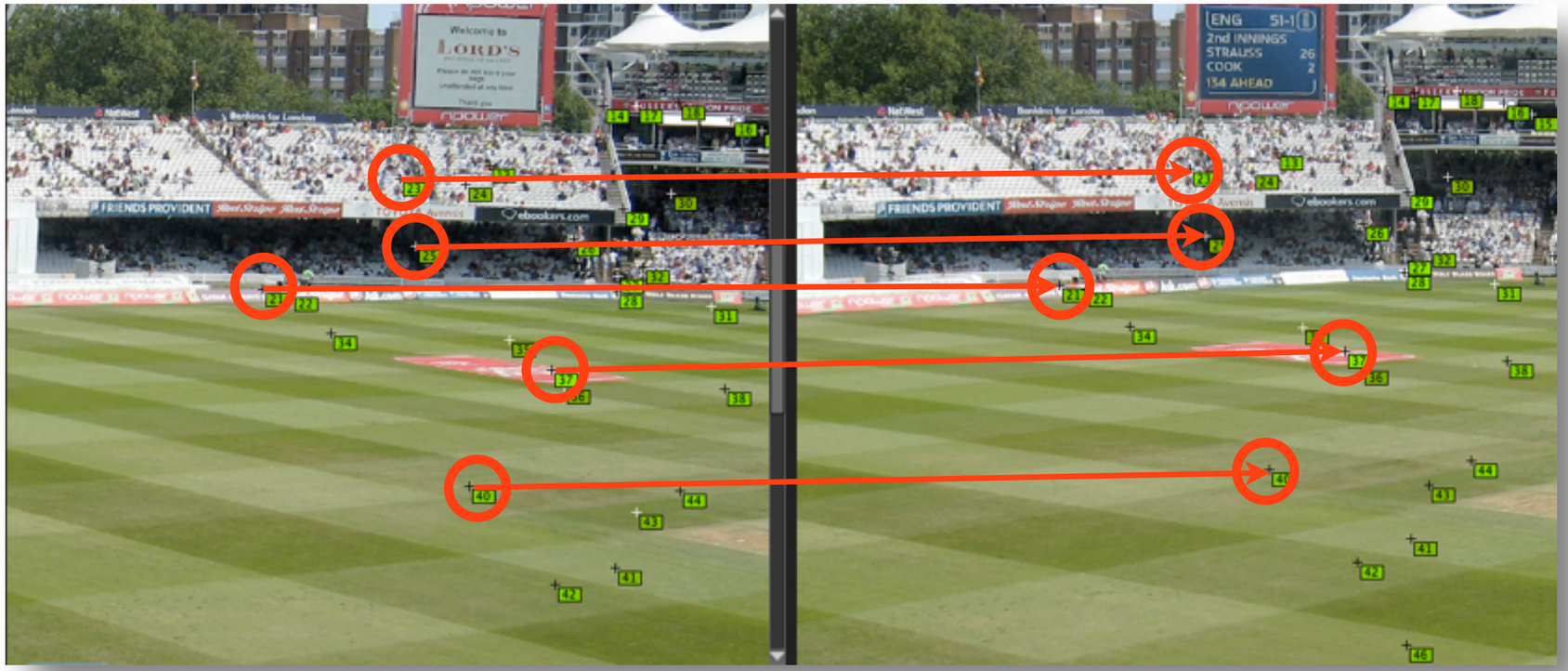
Feature Detection and Matching

* More details on Harris and SIFT Features

1. Harris Corner Detector Algorithm

2. SIFT

Recall: Detection and Matching



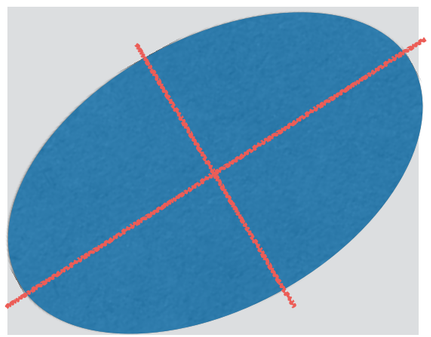
Recall: Corner Detection: Mathematics

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

The quadratic approximation, following Taylor Expansion, simplifies to:

$$E(u,v) \approx \begin{bmatrix} u & v \end{bmatrix} M \begin{bmatrix} u \\ v \end{bmatrix}$$

where $\boldsymbol{M}$ is a second moment matrix computed from image derivatives $\boldsymbol{I_x}$ and $\mathbf{I_y}$:



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

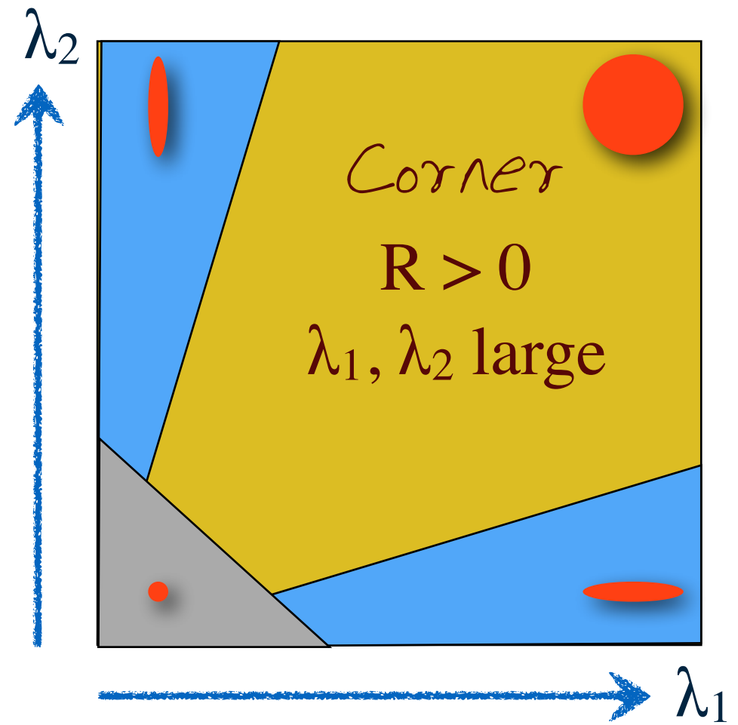
Recall: Harris Corner Response Function

$$R = \det(M) - \alpha\ \mathrm{trace}(M)^2 = \lambda_1\lambda_2 - \alpha(\lambda_1 + \lambda_2)^2$$

α: constant (0.04 to 0.06)

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

* ***R*** depends only on eigenvalues of ***M***
* ***R*** $\Rightarrow$ large for a corner
* ***R*** $\Rightarrow$ negative with large magnitude for an edge
* ***\|R\|*** $\Rightarrow$ small for a flat region
* Note: No explicit computation of eigenvalues required



Harris Detector: Step by Step

1. Compute horizontal and vertical derivatives of the image (convolve with derivative of Gaussians)
2. Computer outer products of gradients $\boldsymbol{M}$
3. Convolve with larger Gaussian
4. Compute scalar interest measure $\boldsymbol{R}$
5. Find local maxima above some threshold, detect features!

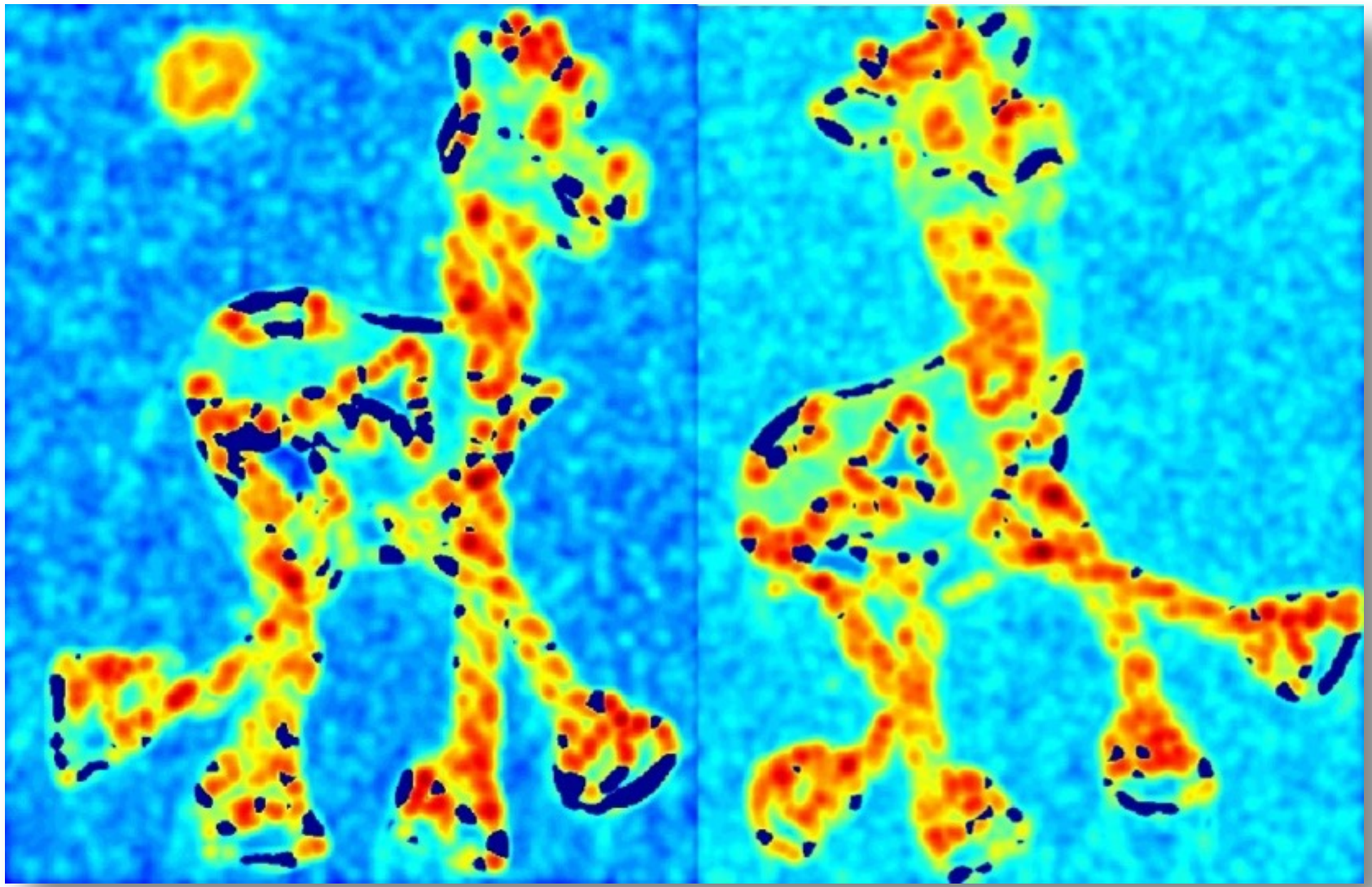
Harris Detector: Workflow

Image Pair

Slides adapted from Aaron Bobick, Alysosha Efros, etc.

Harris Detector: Workflow

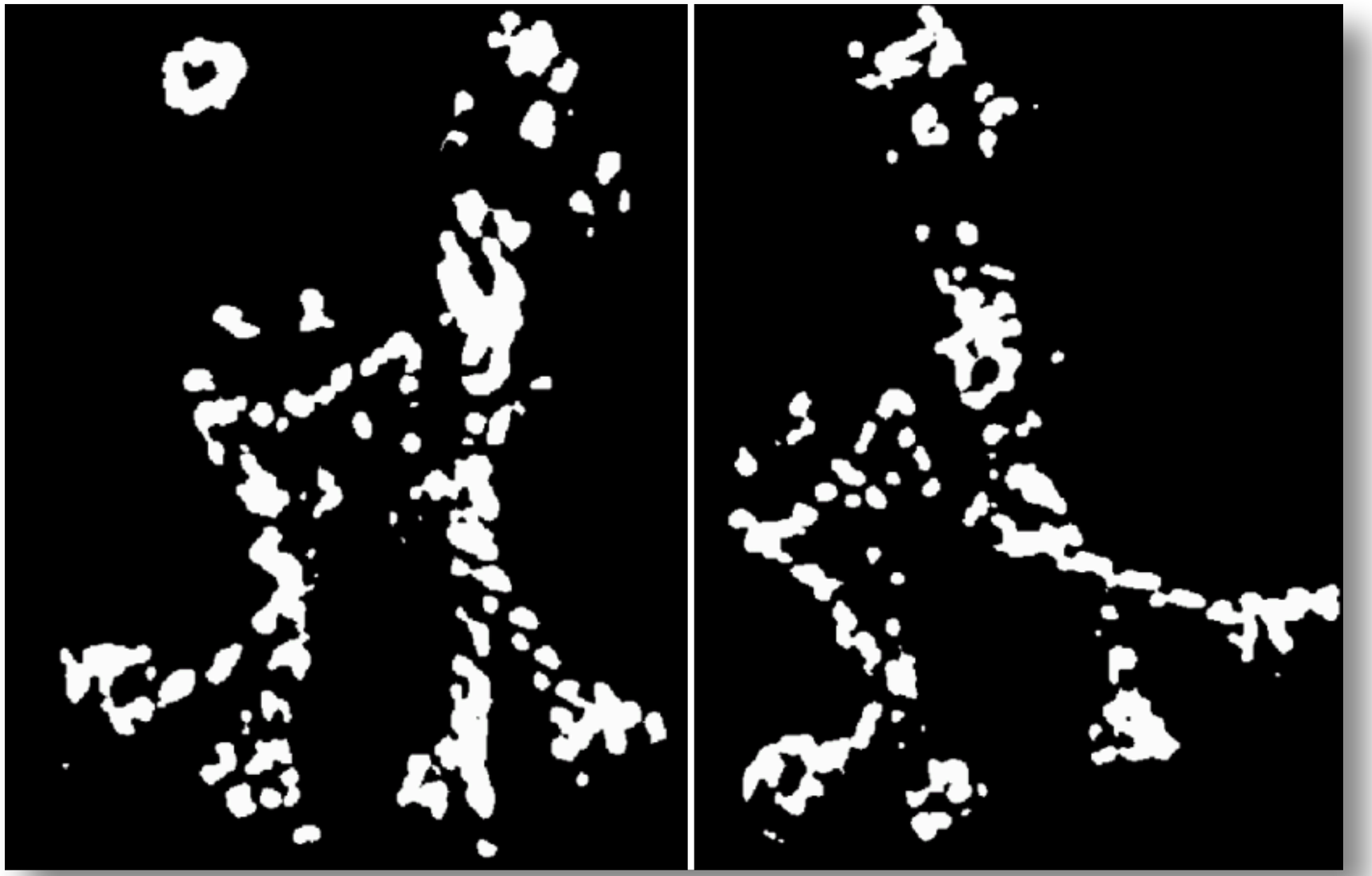
Compute corner response R



Slides adapted from Aaron Bobick, Alysosha Efros, etc.

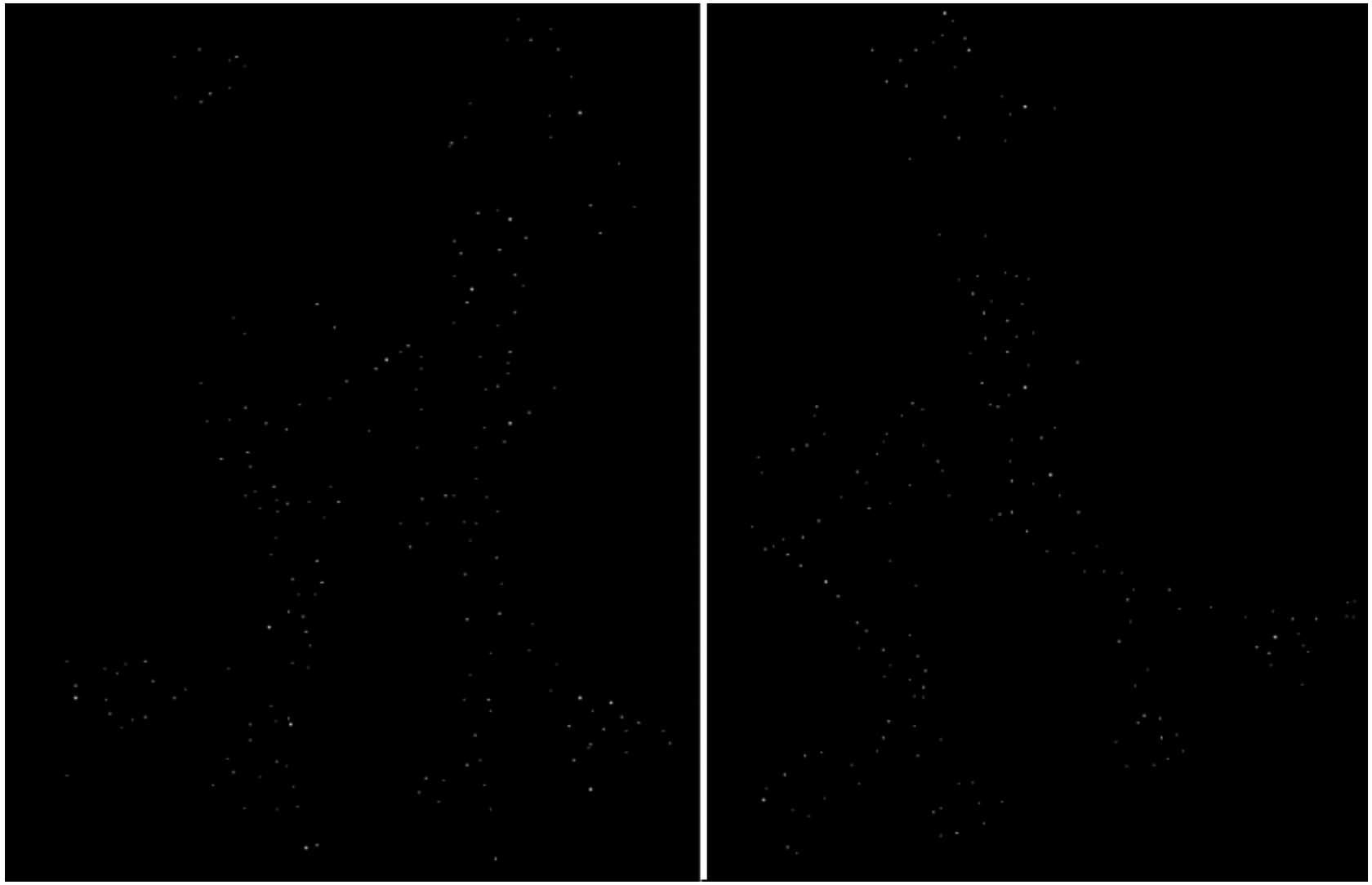
Harris Detector: Workflow

Find points with large corner response: R>threshold



Slides adapted from Aaron Bobick, Alysosha Efros, etc.

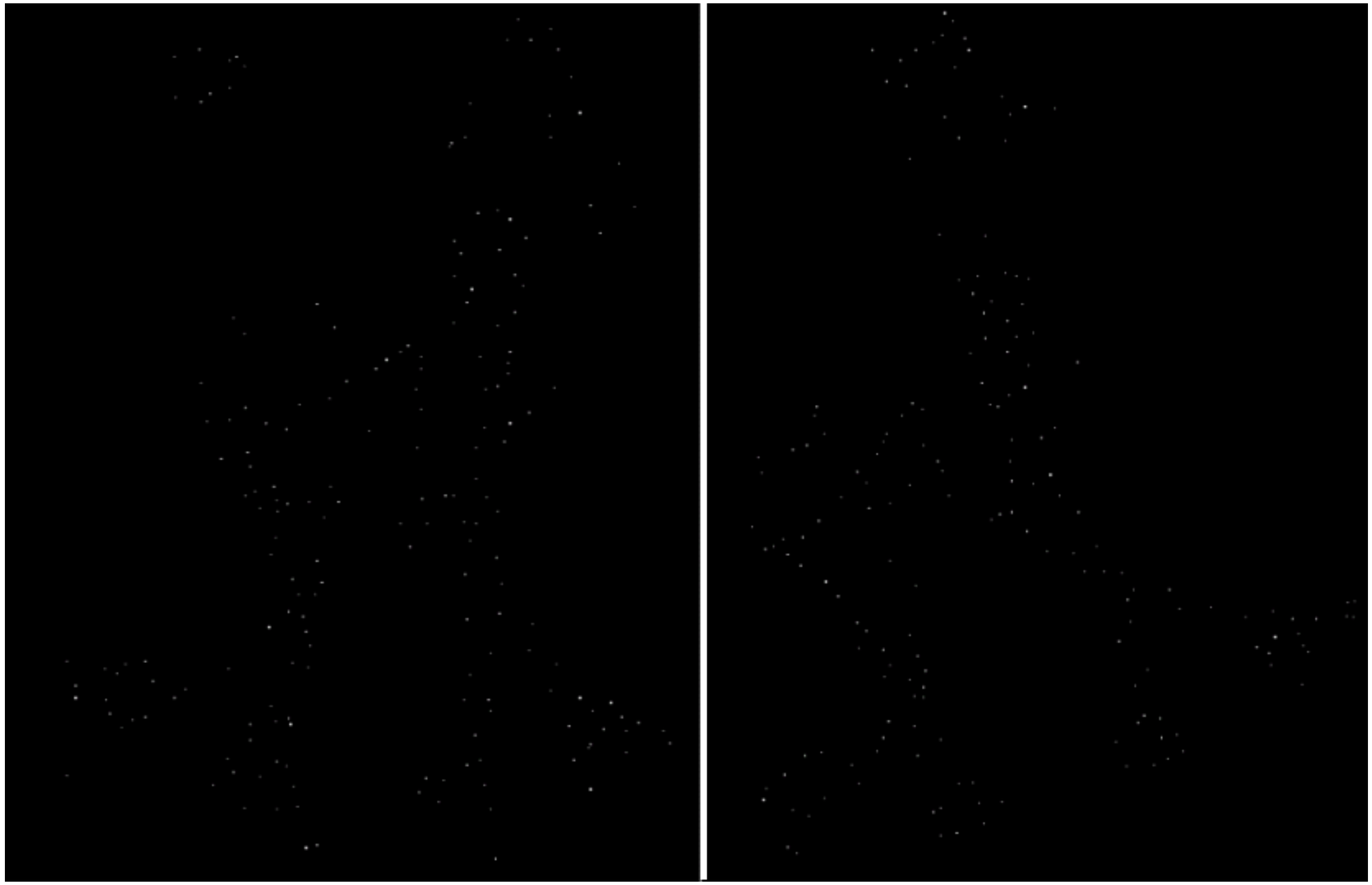
Harris Detector: Workflow



Take only the points of local maxima of R

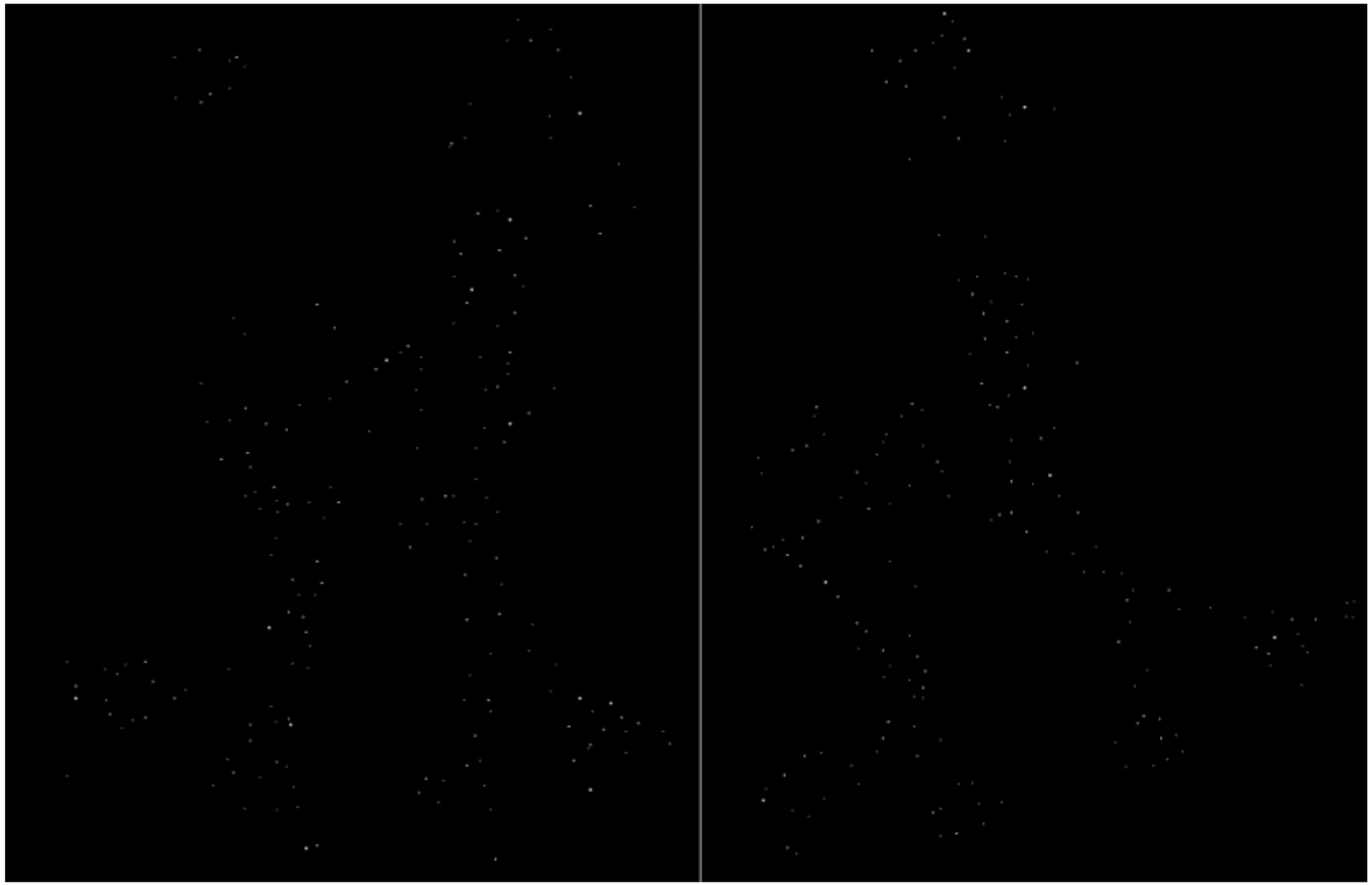
Slides adapted from Aaron Bobick, Alysosha Efros, etc.

Harris Detector: Workflow



Take only the points of local maxima of R

Slides adapted from Aaron Bobick, Alysosha Efros, etc.



Harris Detector: Workflow

Output

Harris Detector Algorithm (Preview)

* Compute Gaussian derivatives at each pixel
* Compute second moment matrix $\boldsymbol{M}$ in a Gaussian window around each pixel
* Compute corner response function $\boldsymbol{R}$
* Threshold $\boldsymbol{R}$
* Find local maxima of response function (non-maximum suppression)

Harris Detector: Some Properties

* Invariant to Rotation?

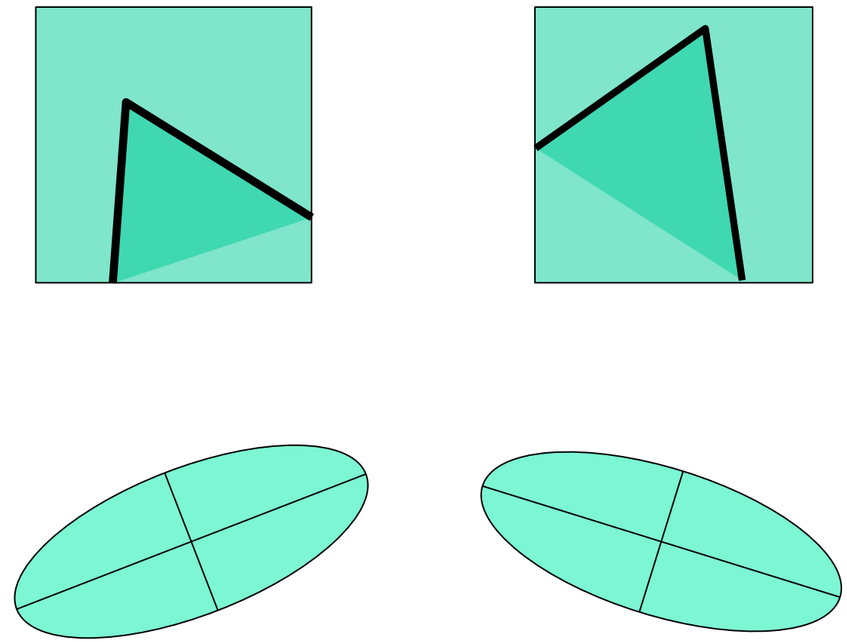
* Invariance to image intensity?

* Invariant to image scale?

Slides adapted from Aaron Bobick

Harris Detector: Some Properties

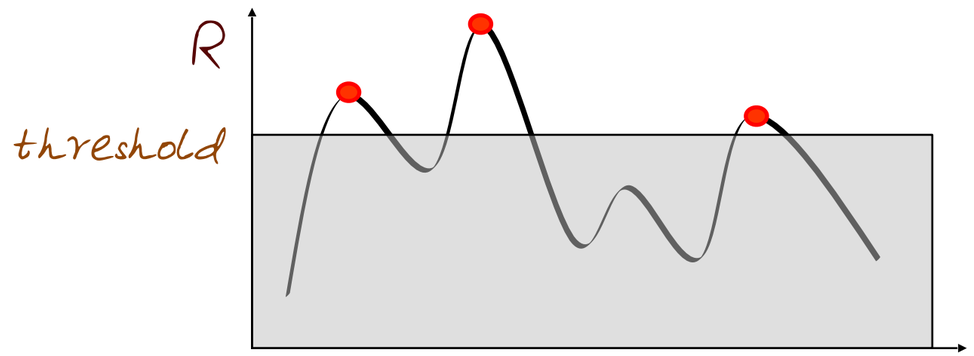
* Invariant to Rotation?
* Ellipse rotates but its shape (i.e. eigenvalues) remains the same
* Corner response R is invariant to image rotation

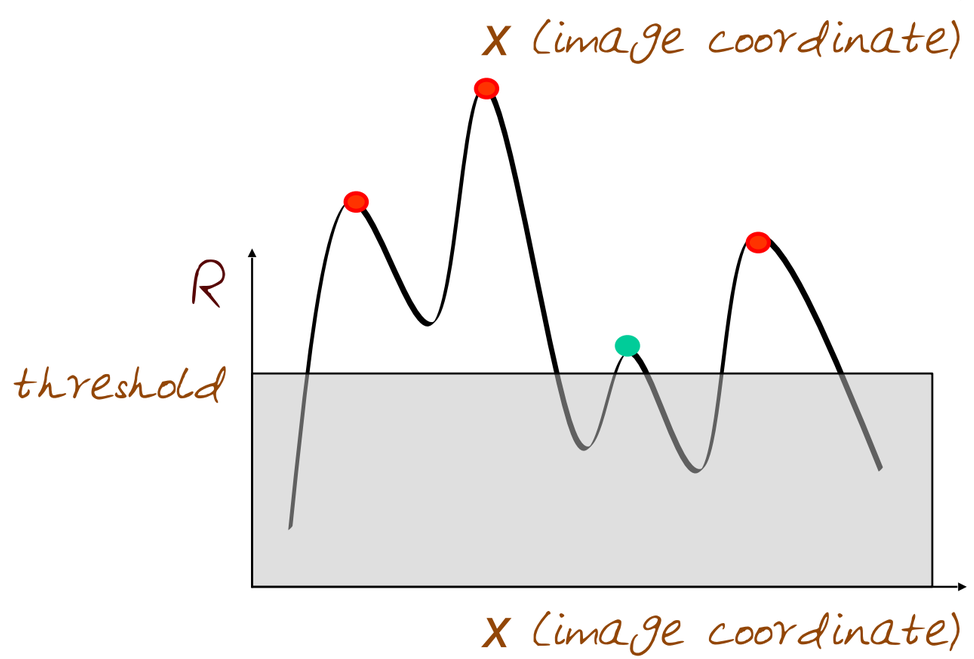


Slides adapted from Aaron Bobick

Harris Detector: Some Properties

- Invariance to image intensity?
- Mostly invariant to additive and multiplicative intensity changes
  - Only derivatives are used
  - Invariance to intensity shift:
    - $I \rightarrow I + b$
  - Intensity scale:
    - $I \rightarrow a I$





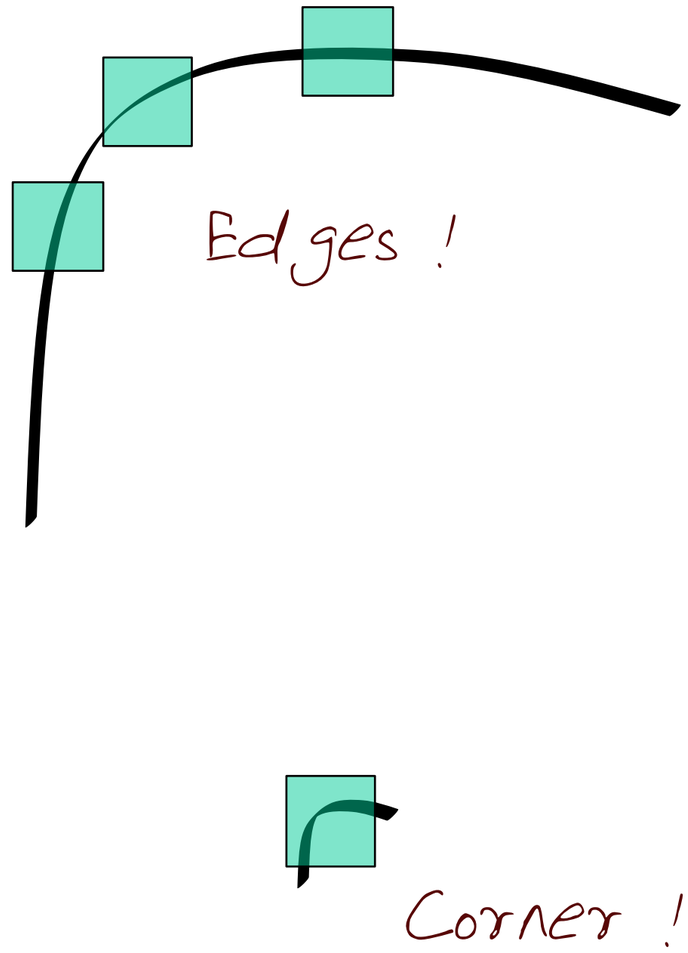
Slides adapted from Aaron Bobick

Harris Detector: Some Properties

* Invariant to image scale?

* Not Invariant to image scale,

* But can we do something about this?

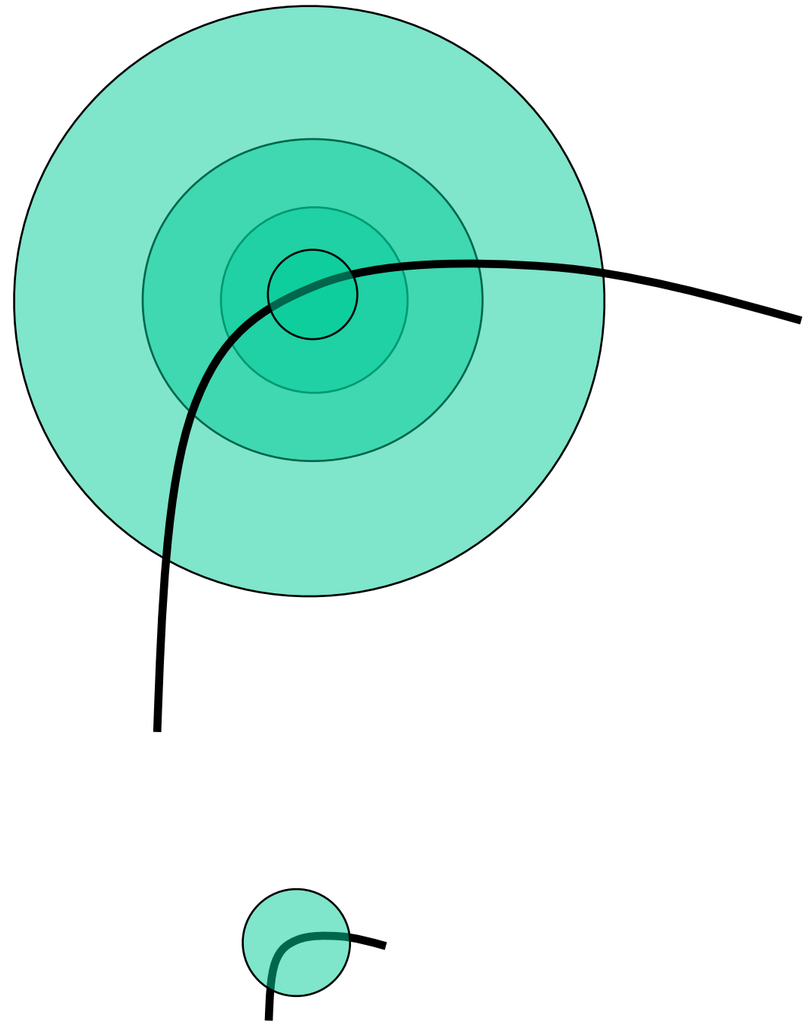


Slides adapted from Aaron Bobick

Scale Invariant Detection

* Consider regions (e.g. circles) of different sizes around a point

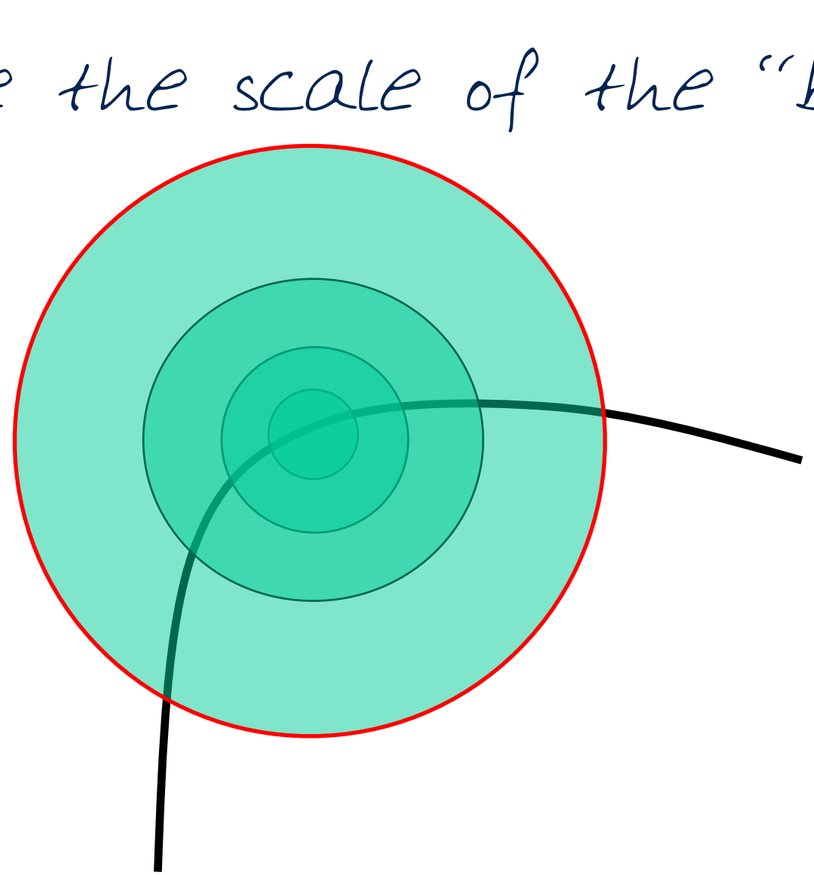
* Regions of corresponding sizes will look the same in both images

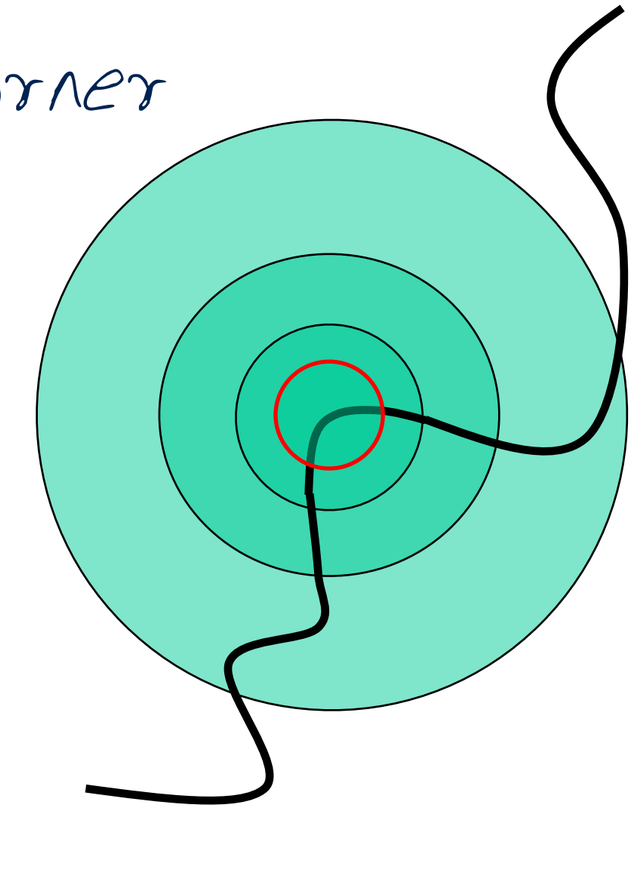


Slides adapted from Aaron Bobick

Scale Invariant Detection

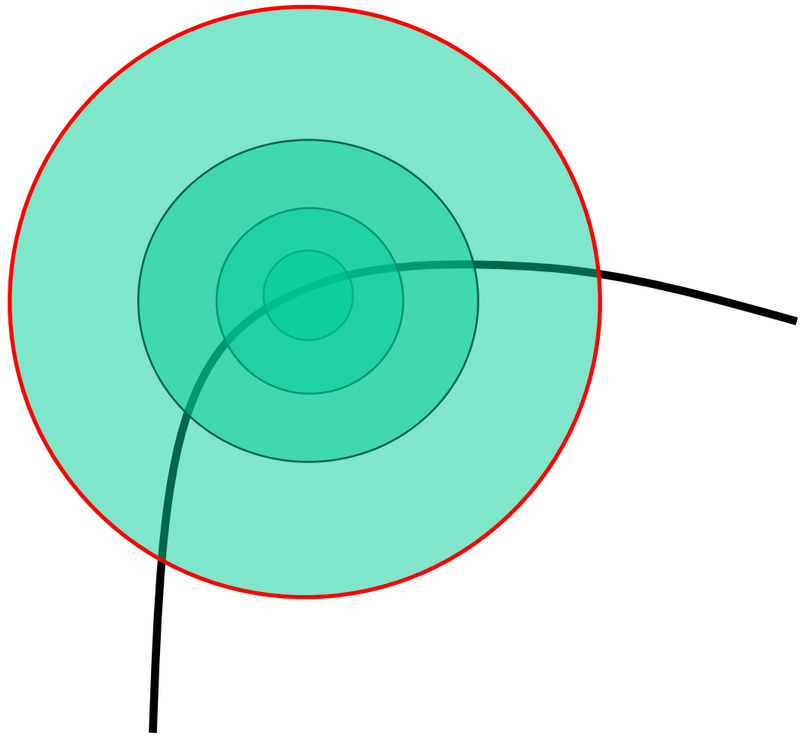
* The problem: how do we choose corresponding circles independently in each image?
* Choose the scale of the "best" corner

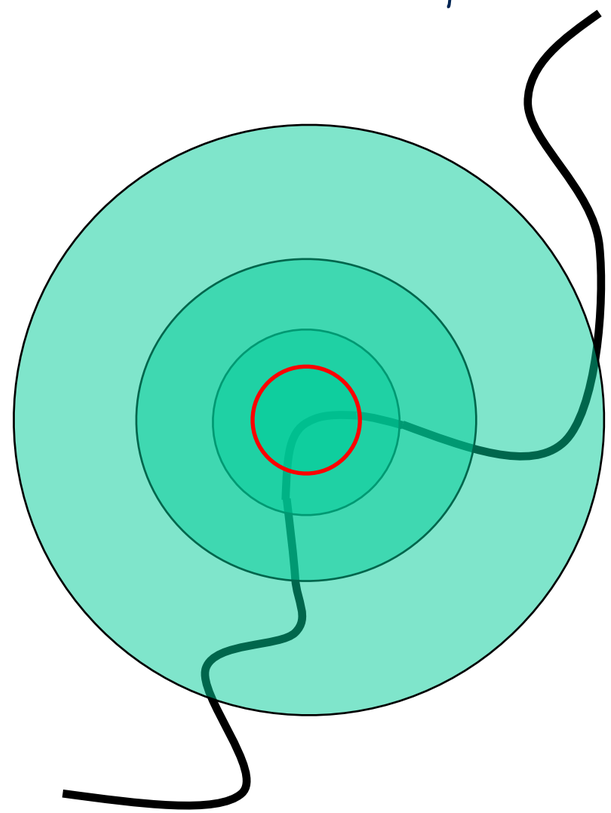




Scale Invariant Detection

* A region (circle), which is "scale invariant"
* Not affected by the size but will be the same for "corresponding regions"





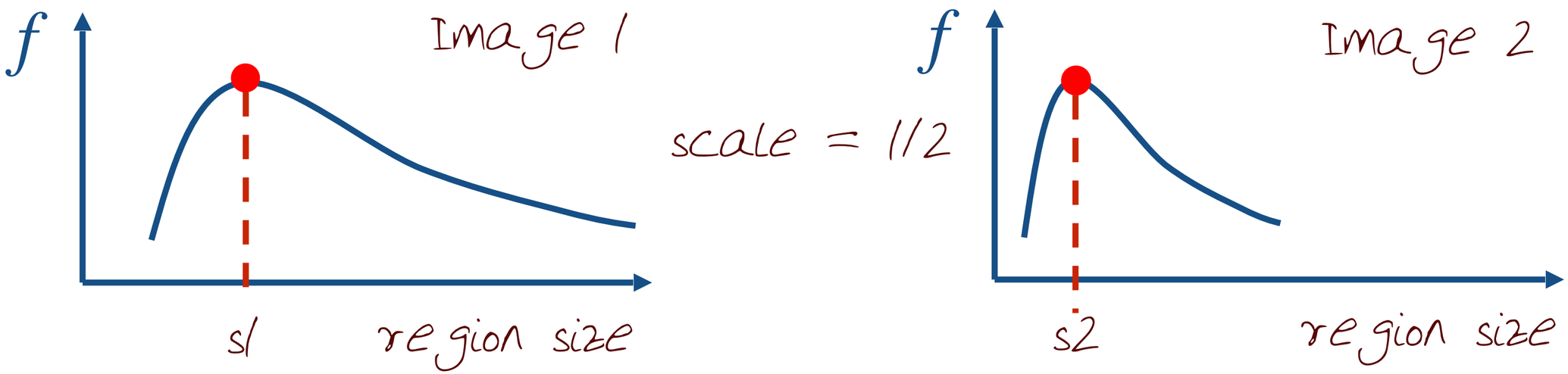
Scale Invariant Detection

* A region (circle), which is "scale invariant"
* Not affected by the size but will be the same for "corresponding regions
* Example: Average intensity. For corresponding regions (even of different sizes) it will be the same

Slides adapted from Aaron Bobick

Scale Invariant Detection

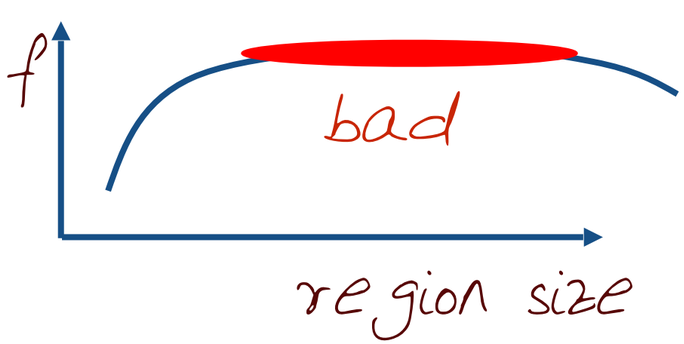
* At a point, compute the scale invariant function over different size neighborhoods (different scales)
* Choose the scale for each image at which the function is a maximum

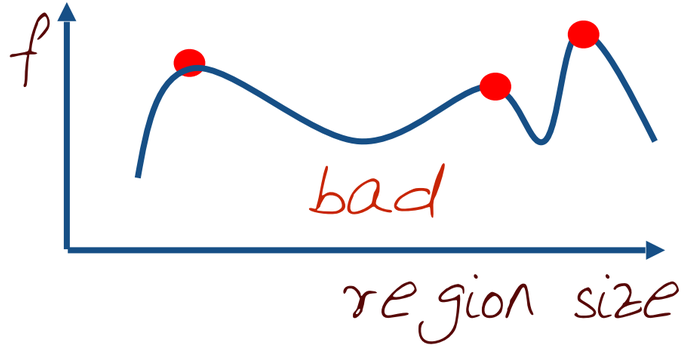


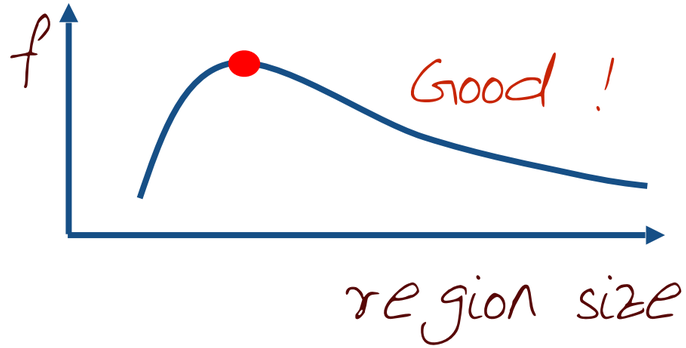
Slides adapted from Aaron Bobick

Scale Invariant Detection

* A "good" function for scale detection has one stable sharp peak

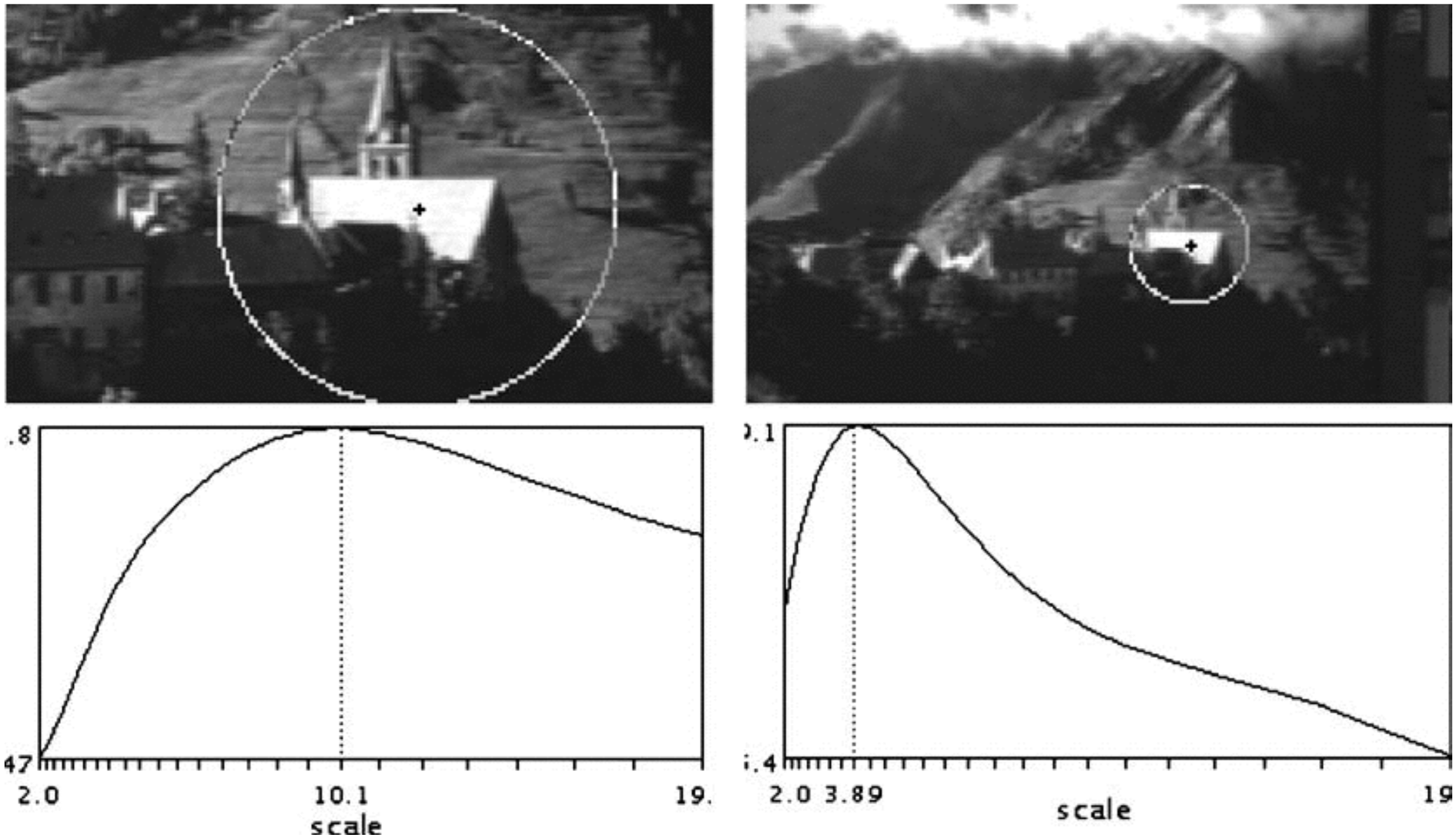






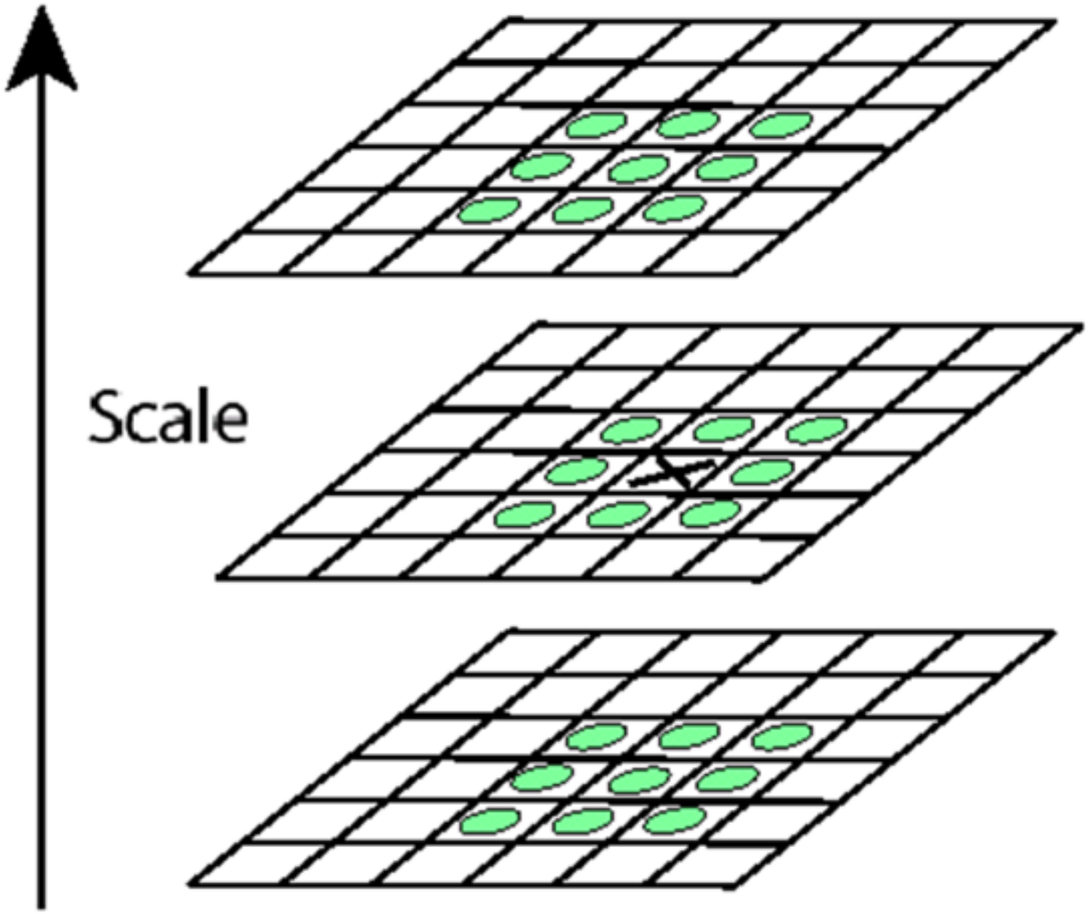
* For usual images: a good function would be a one which responds to contrast (sharp local intensity change)

Scale Sensitive Response



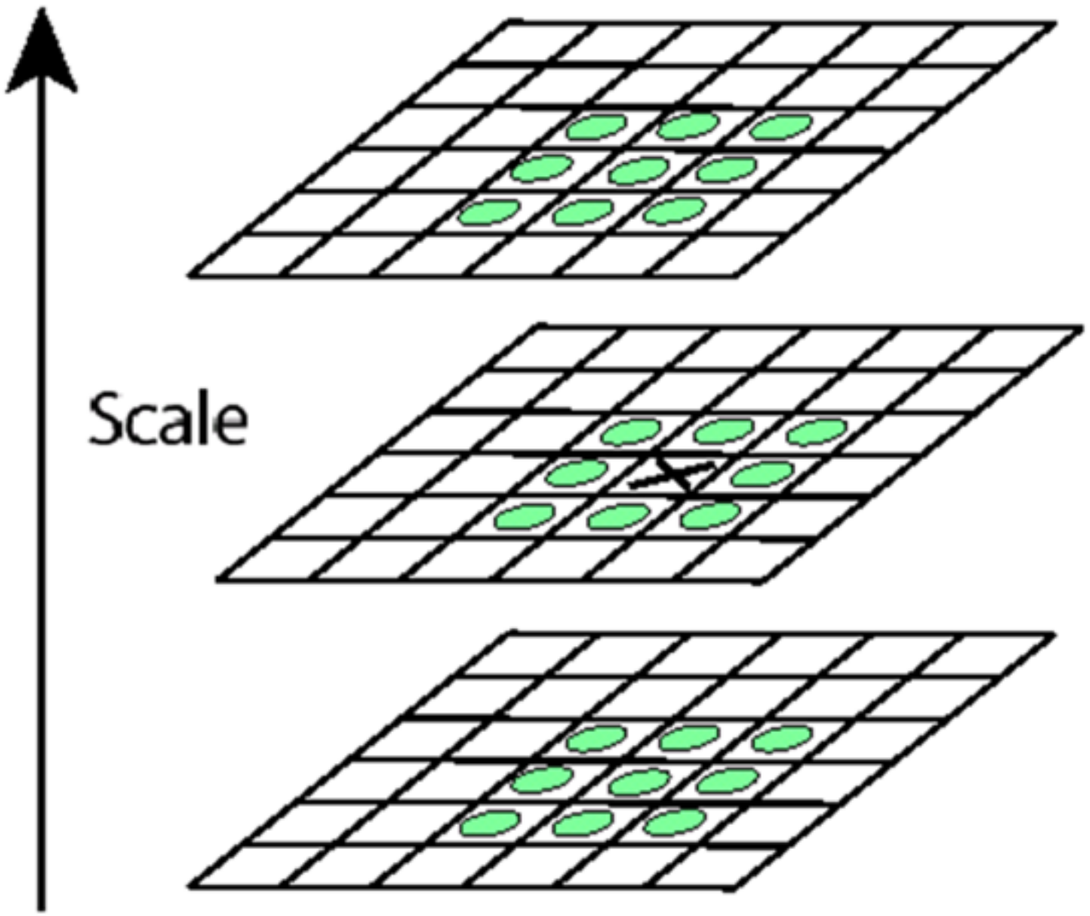
Key Point Localization in Space

* Find robust extremum (maximum or minimum) both in space and in scale



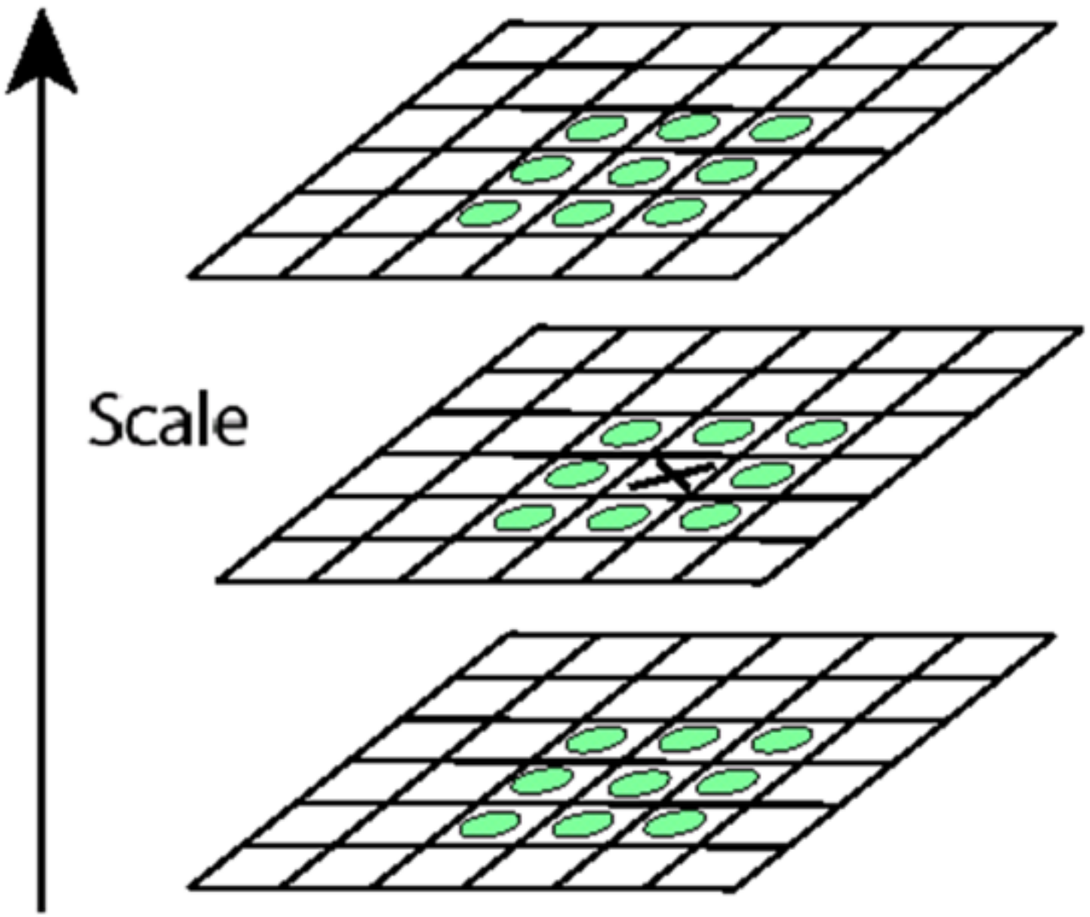
Key Point Localization

* SIFT: Scale Invariant Feature Transform
* Specific suggestion: use pyramid to find maximum values (remember edge detection?) – then eliminate "edges" and pick only corners

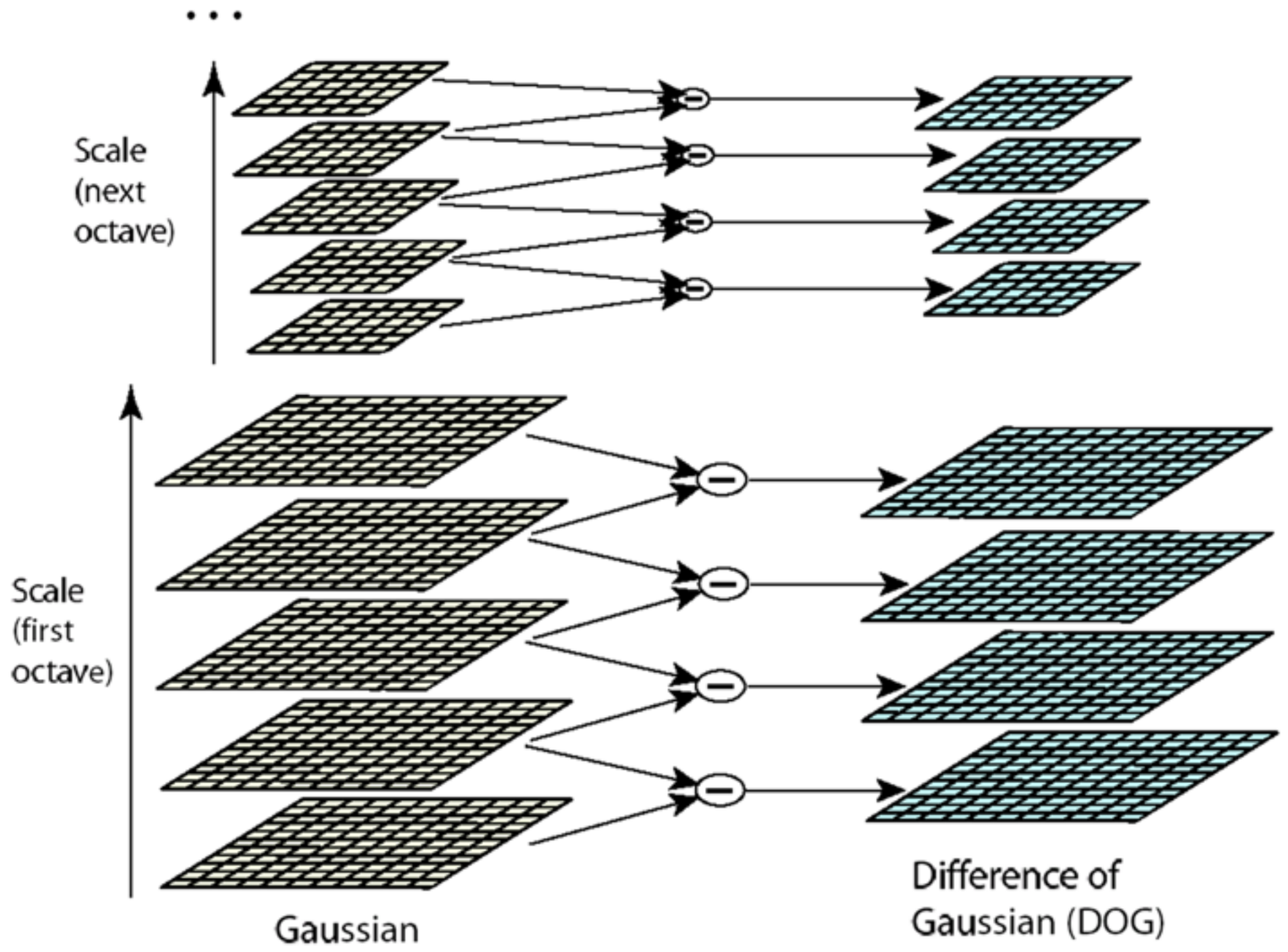


Key Point Localization

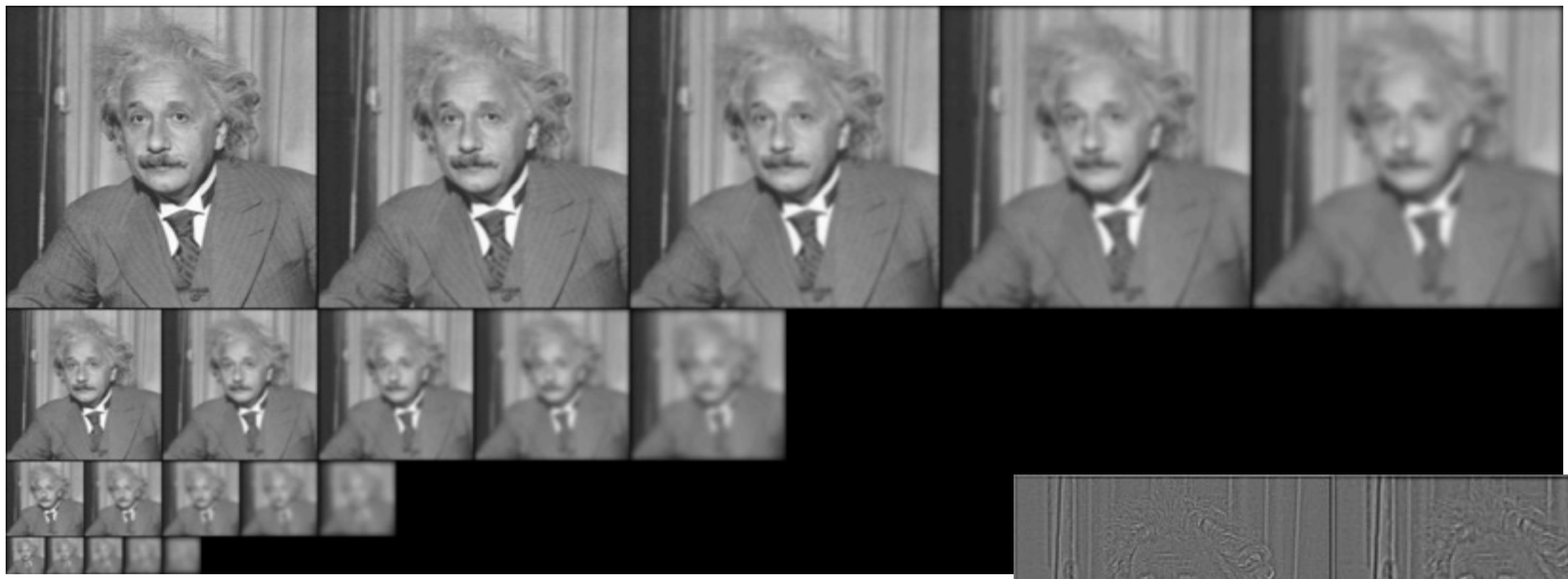
* Each point is compared to its 8 neighbors in the current image and 9 neighbors each in the scales above and below)



Scale Space Processed One Octave at a Time

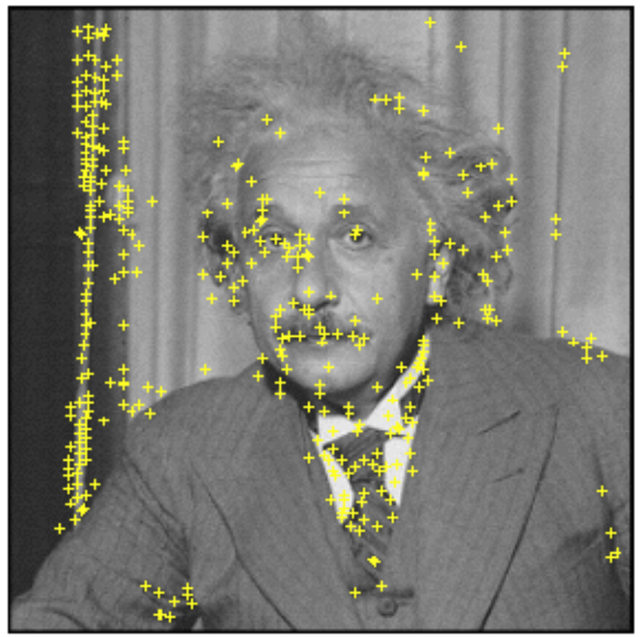


Extrema at Different Scales

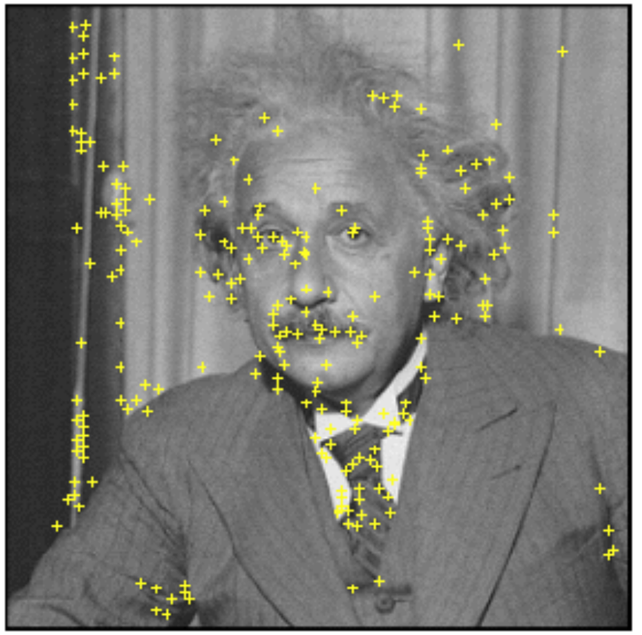


Remove Low Contrast, Edge Bound

Extrema points



Contrast > C



Not on edge

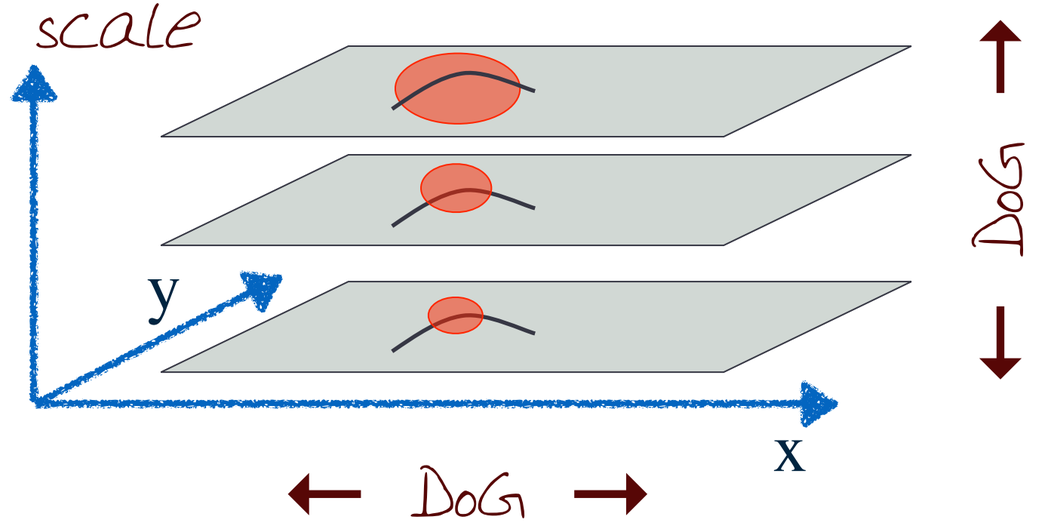
Scale Invariant Detectors

* Harris-Laplacian
* Find local maximum of:
  * Harris corner detector in space (image coordinates)
  * Laplacian in scale

(Mikolajczyk and Schmid, 2001)

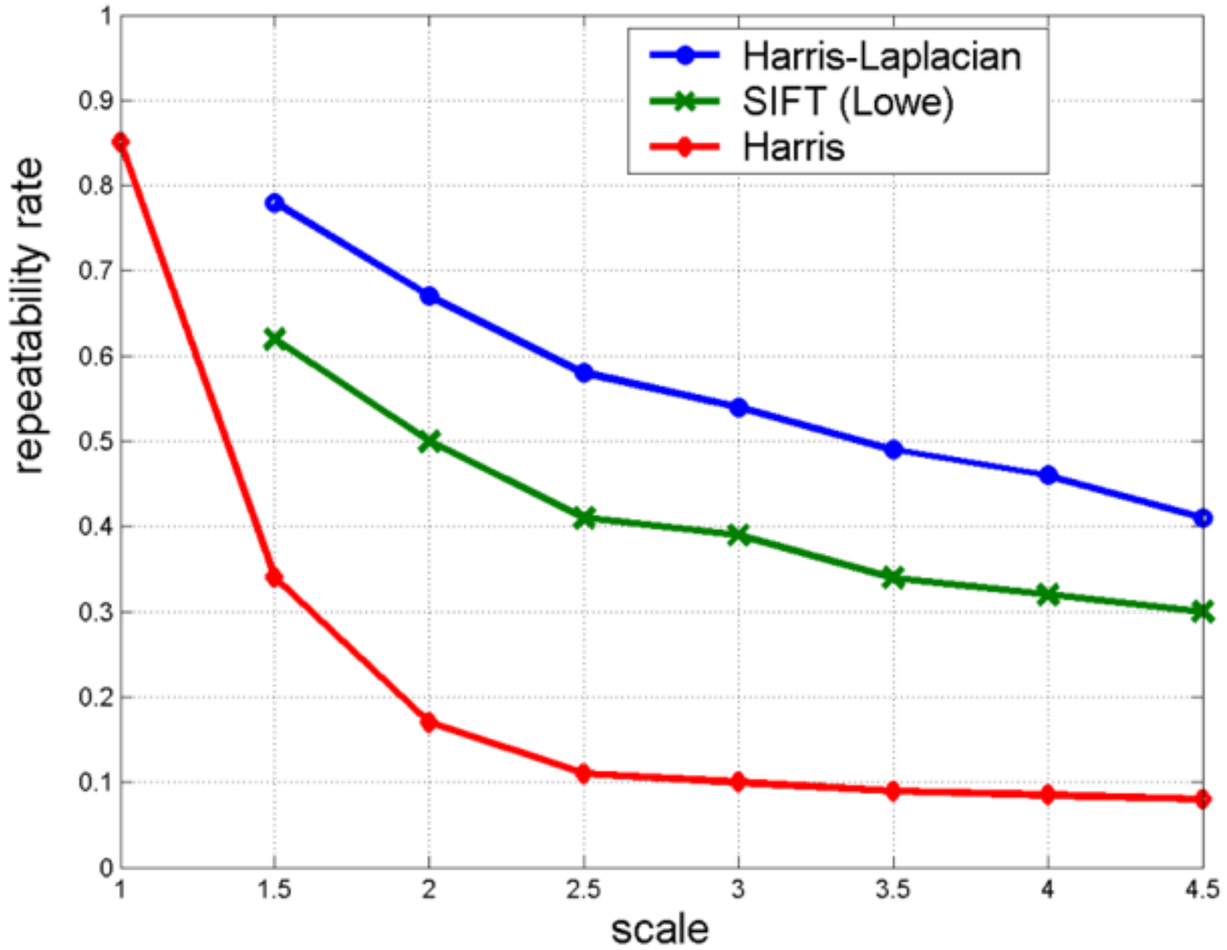
Scale Invariant Detectors

* SIFT (Lowe, 2004)
* Find local maximum of:
  * Difference of Gaussians (DoG) in space and scale
  * DoG is simply a pyramid of the difference of Gaussians within each octave



Lowe 2004

Scale Invariant Detectors



K. Mikolajczyk, C. Schmid (2001) "Indexing Based on Scale Invariant Interest Points". ICCV 2001

SIFT (Scale-Invariant Feature Transform)

* Orientation assignment
  * Compute best orientation(s) for each keypoint region.
* Keypoint description
  * Use local image gradients at selected scale and rotation to describe each keypoint region.

http://en.wikipedia.org/wiki/File:Sift_keypoints_filtering.jpg

Invariant Local Features

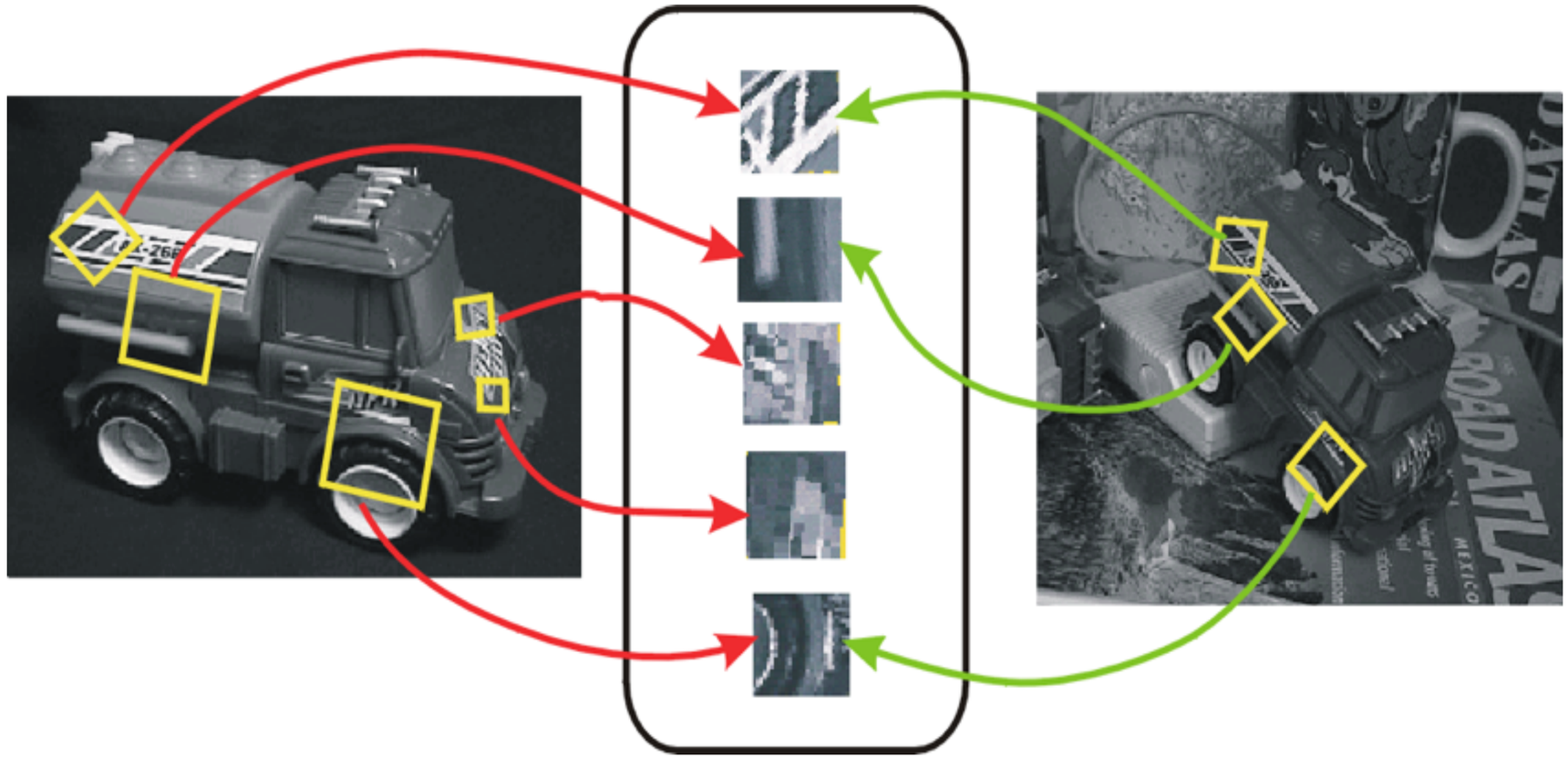


Image content is transformed into local feature coordinates that are invariant to translation, rotation, scale, and other imaging parameters

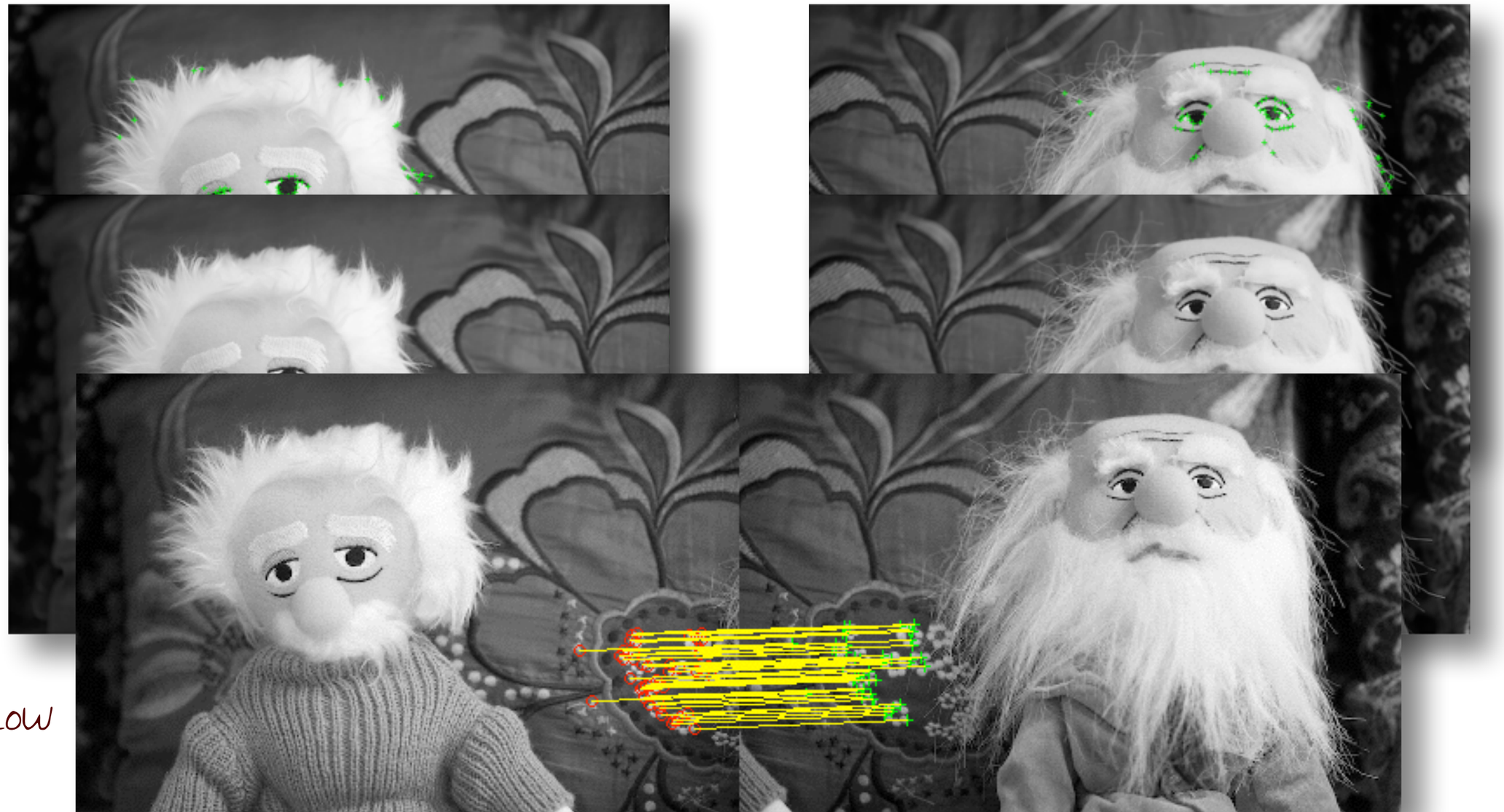
Lowe 2004

Results

Detect

match

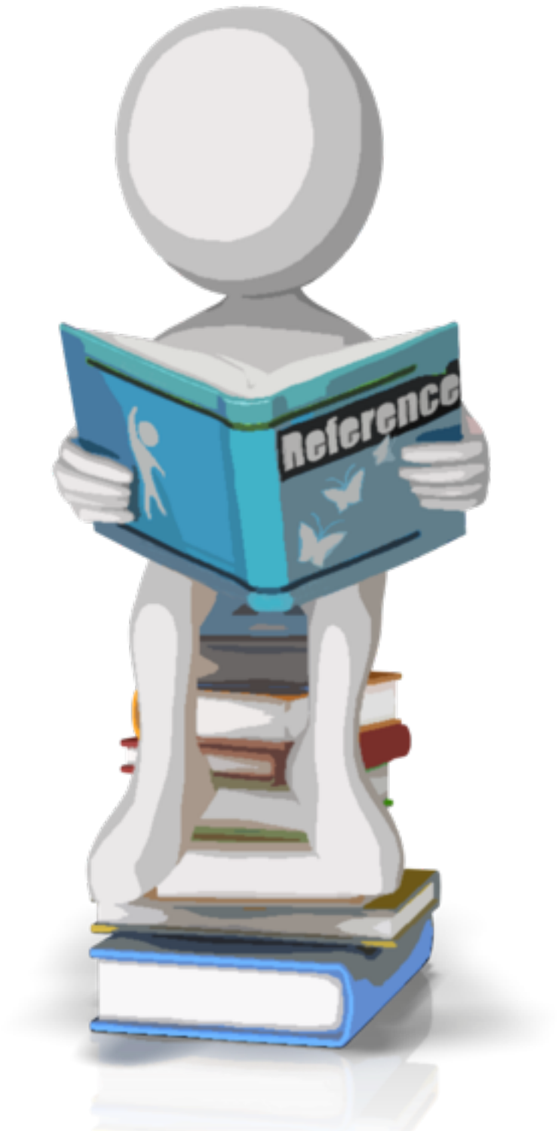
Show



Summary

* Discussed invariants for feature detection
* Harris Corner Detector Framework
* SIFT detector

Further Reading



* Harris and Stephens (1988) "A Combined Corner and Edge Detector." Proceedings of the 4th Alvey Vision Conference, 1988,
* Mikolajczyk and Schmid (2001). "Indexing Based on Scale Invariant Interest Points". ICCV 2001
* Lowe (2004) "Distinctive Image Features from Scale-Invariant Keypoints". IJCV 2004
* Search for "Features" on OpenCV site

Next Class

* Moving on to more advanced topics

Credits

* For more information, see:
  * Richard Szeliski (2010) Computer Vision: Algorithms and Applications, Springer
* Some concepts in slides motivated by similar slides by Aaron Bobick, Alysoha Efros and Greg Turk
* Additional list will be available on website

Computational Photography

* Study the basics of computation and its impact on the entire workflow of photography, from capturing, manipulating and collaborating on, and sharing photographs.

Georgia Tech College of Computing