Week 5: The image gradient

News:

A1 is being marked. Marks will be available on blackboard by next lecture.

A2 is out! We'll check it out during the tutorial, tonight.

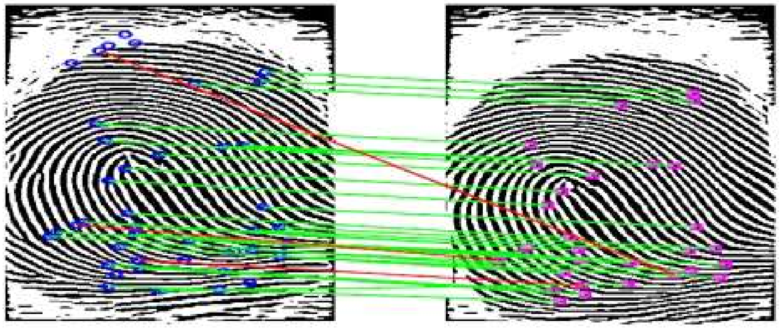
Vote for the alternative office hour.

- Link in the announcements section of the course website.

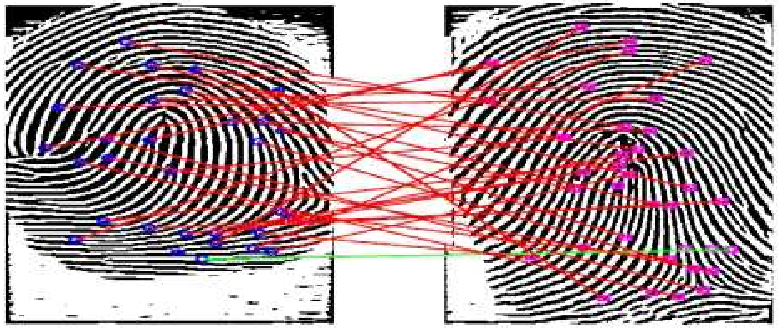
Tutorial tonight on:

- A2
- Answers to A1 Part B, including and how estimating the pseudoinverse is not relevant
- Paper on Accidental Pinhole and Pinspeck cameras (time permitting)

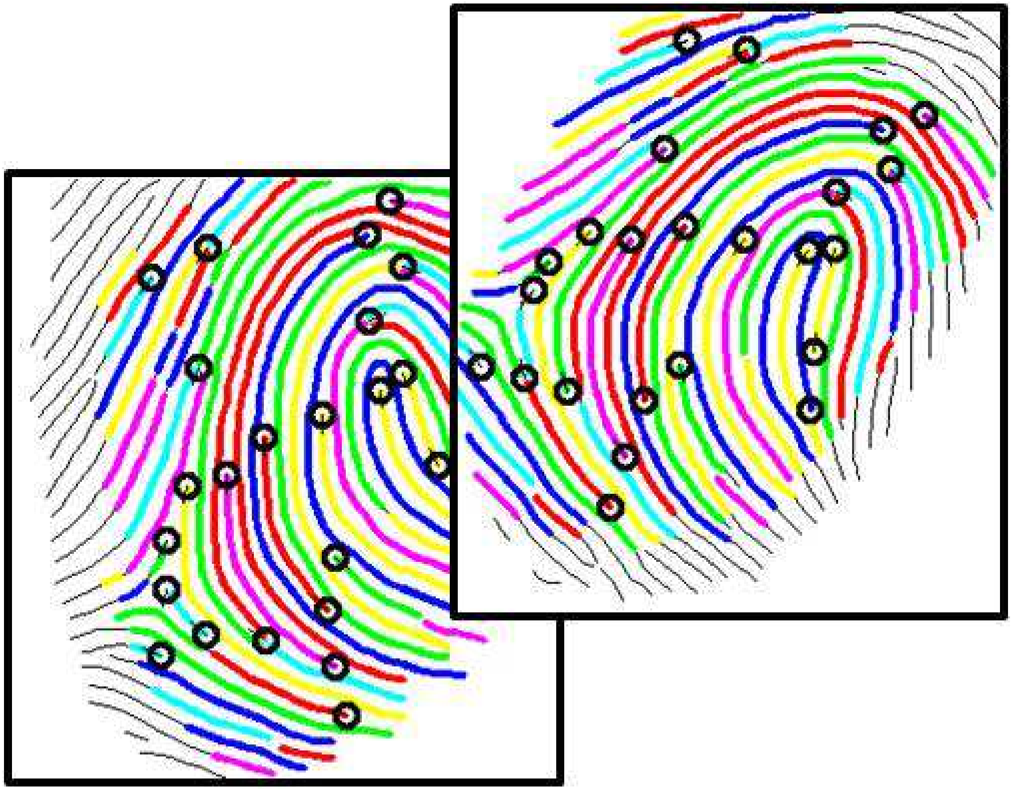
Curves applications: matching features



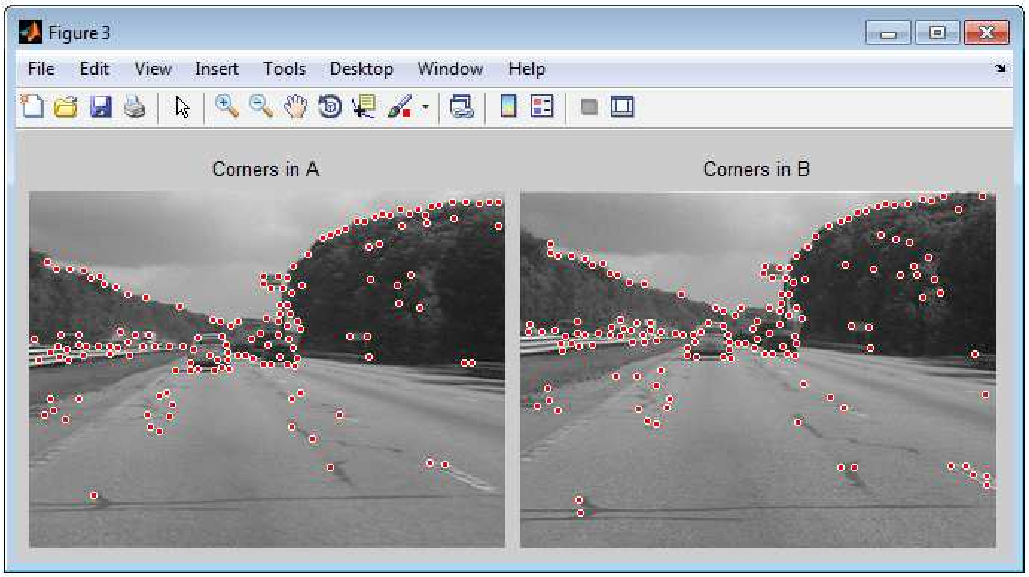
(a) Genuine: finger #31 imp. #1 & imp. #2

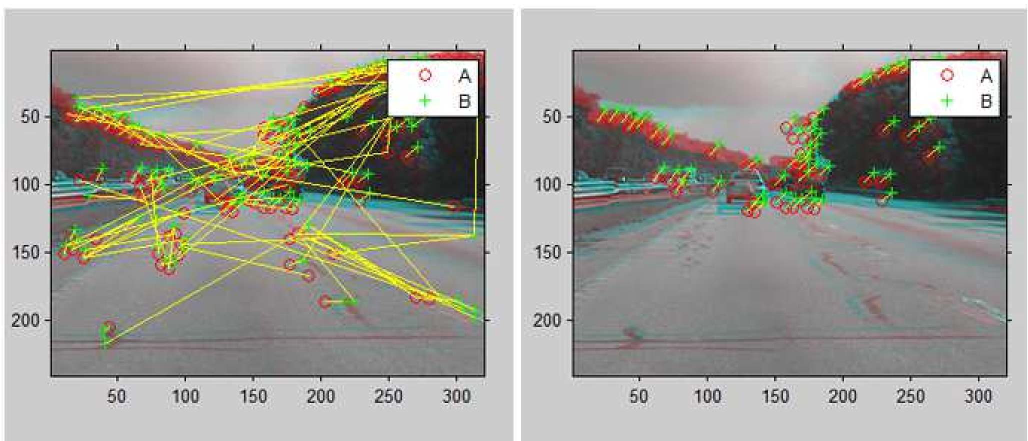


(b) Impostor: finger #31 imp. #1 & finger #11 imp. #1

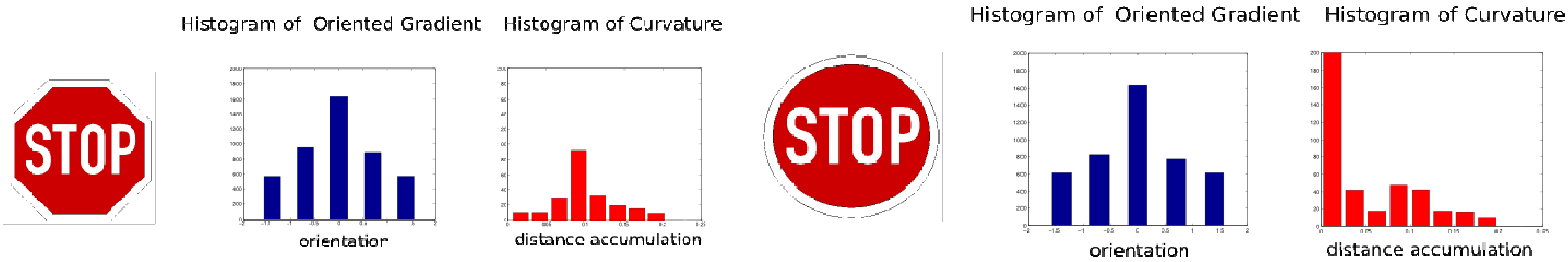


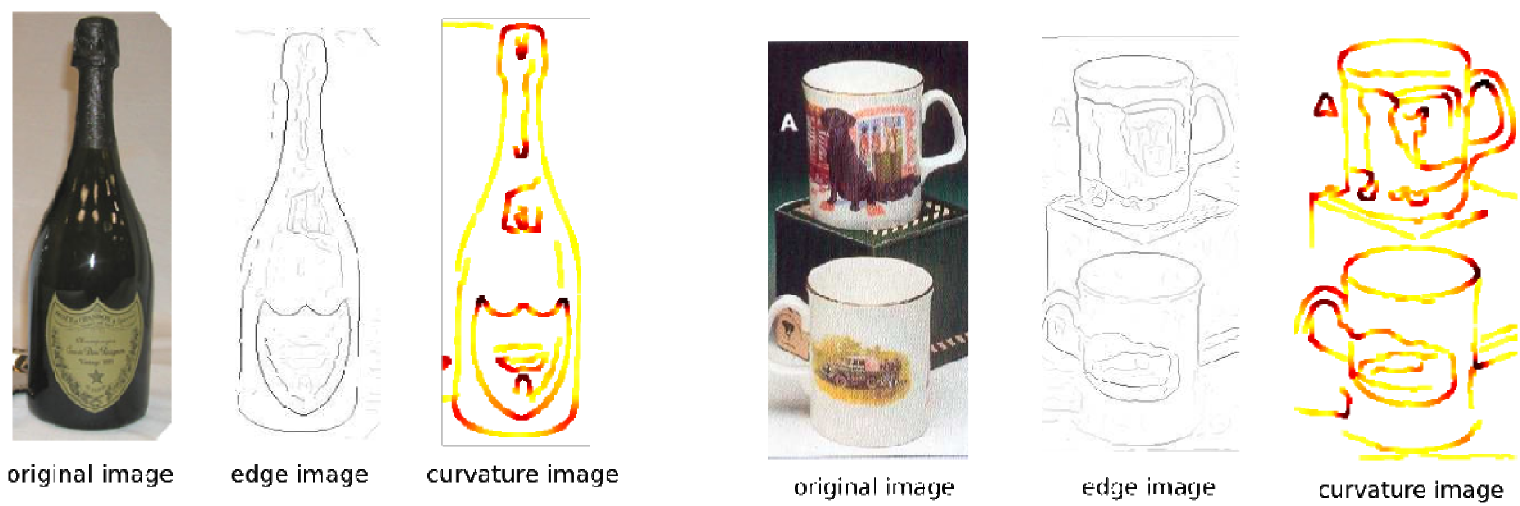
Curves applications: matching features





Curves applications: detection





From: http://hci.iwr.uni-heidelberg.de/COMPVIS/research/curvature/

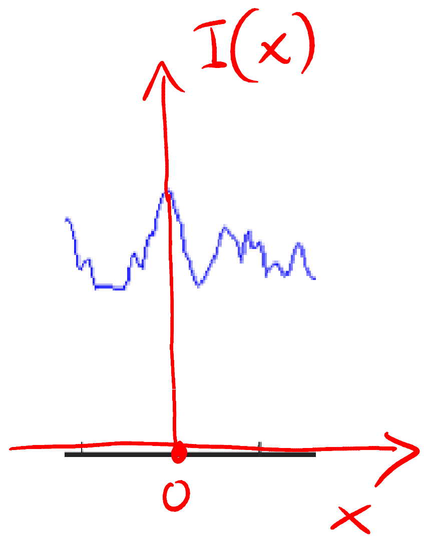
Curves: summary

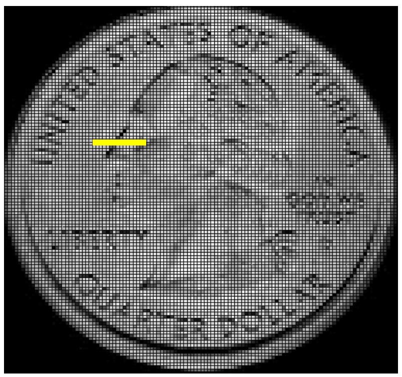
Today

Images as 3D surfaces

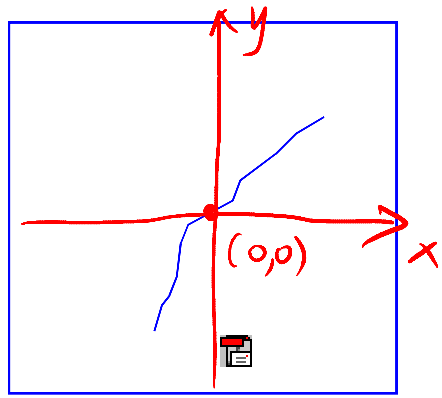
Local Analysis of Image Patches: Outline

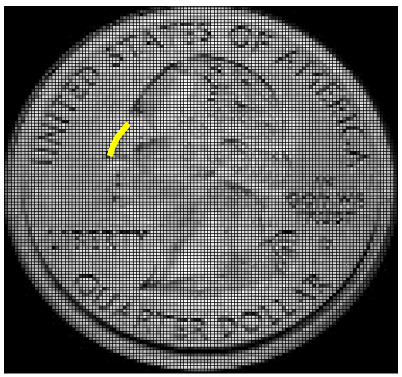
As graph in 2D



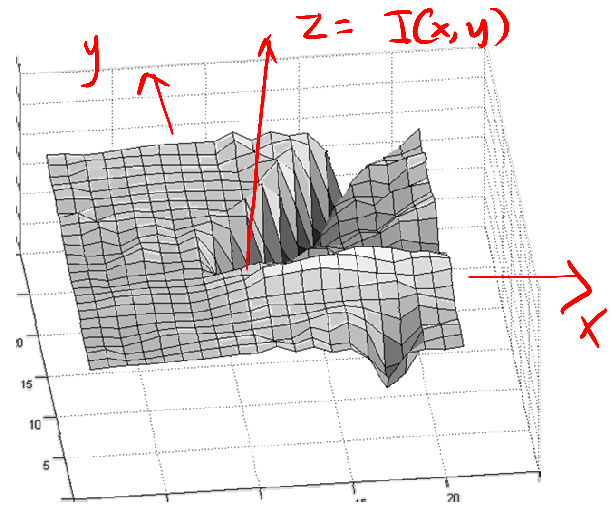


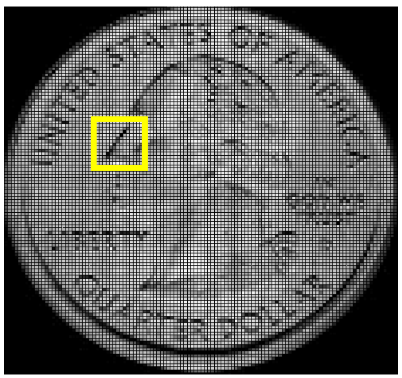
As curve in 2D





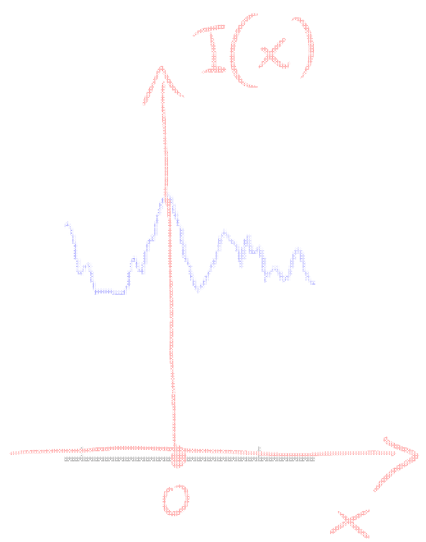
As surface in 3D

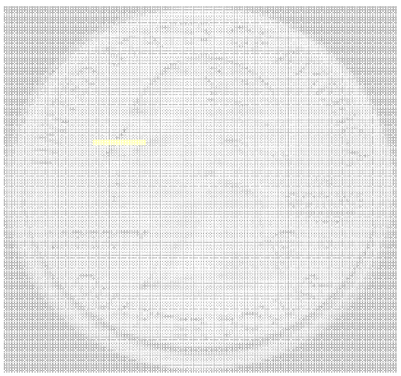




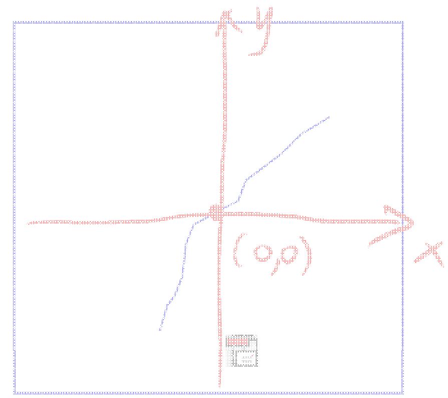
Local Analysis of Image Patches: Outline

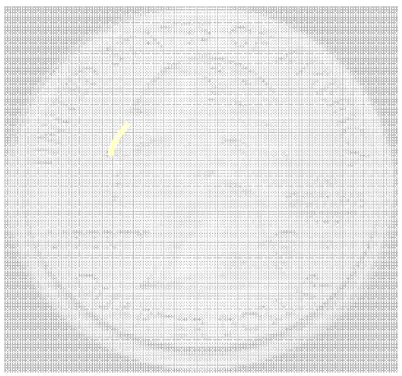
As graph in 2D



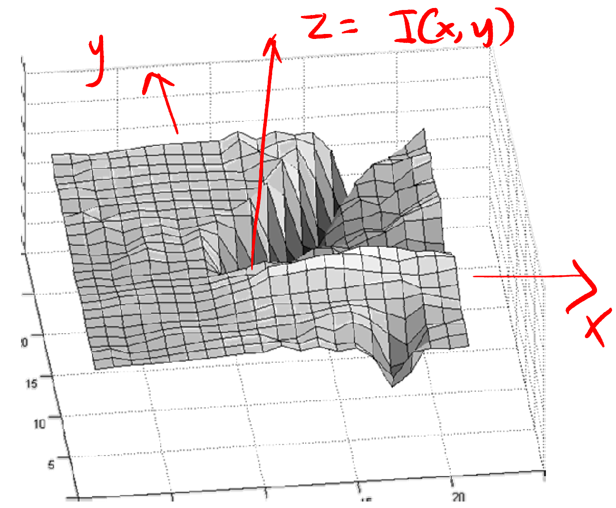


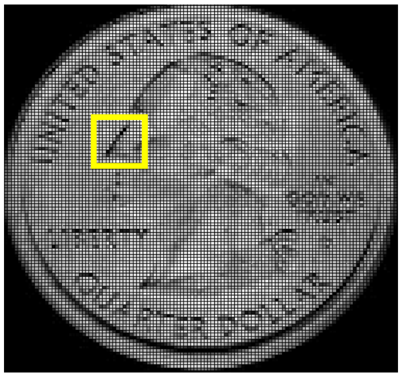
As curve in 2D





As surface in 3D





Topic 4.3:

Local analysis of 2D image patches

- Images as surfaces in 3D
- Directional derivatives
- Image Gradient
- Edge detection & localization
    - Gradient extrema
    - Laplacian zero-crossings
- Painterly rendering
- Local geometry at image extrema
- The Image Hessian
- Eigenvectors & eigenvalues
- Corner & feature detection
    - Lowe feature detector
    - Harris/Forstner detector

Image ⇔ Surface in 3D

Gray-scale image

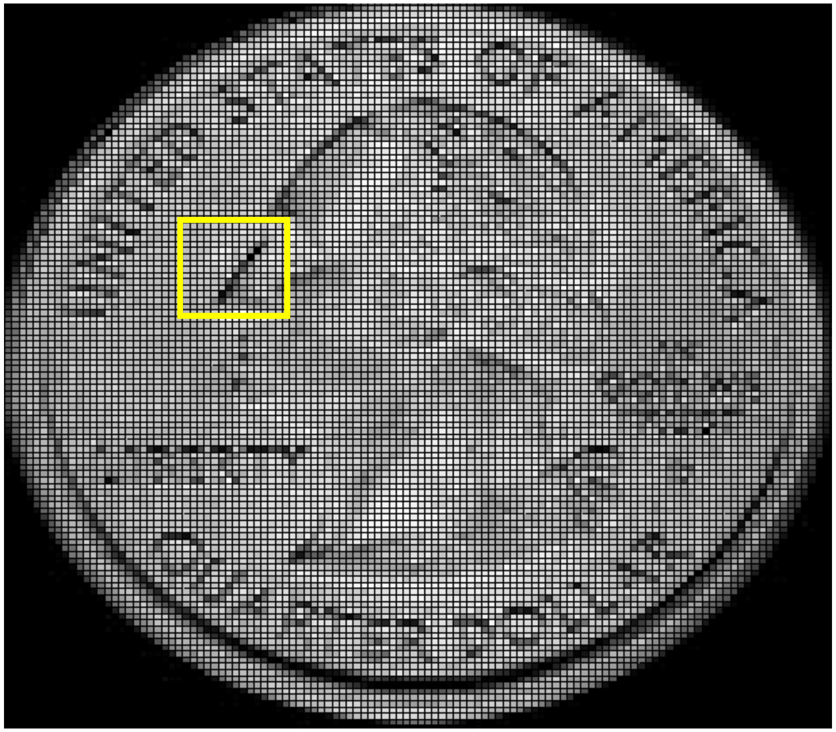


Image ⇔ Surface in 3D

Gray-scale image

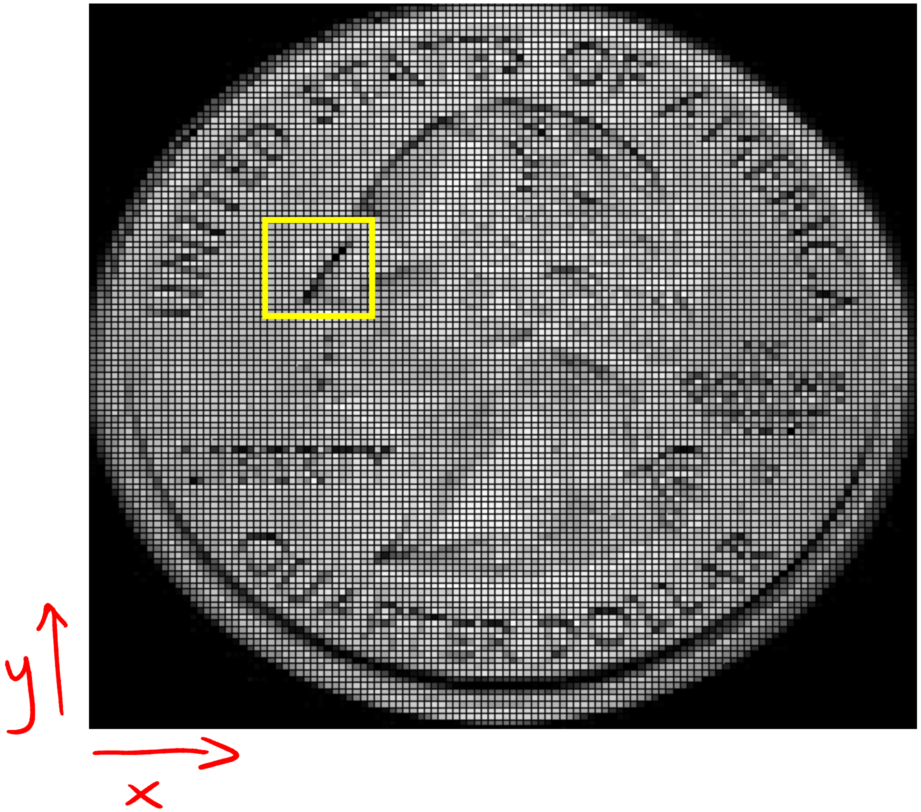


Image patch

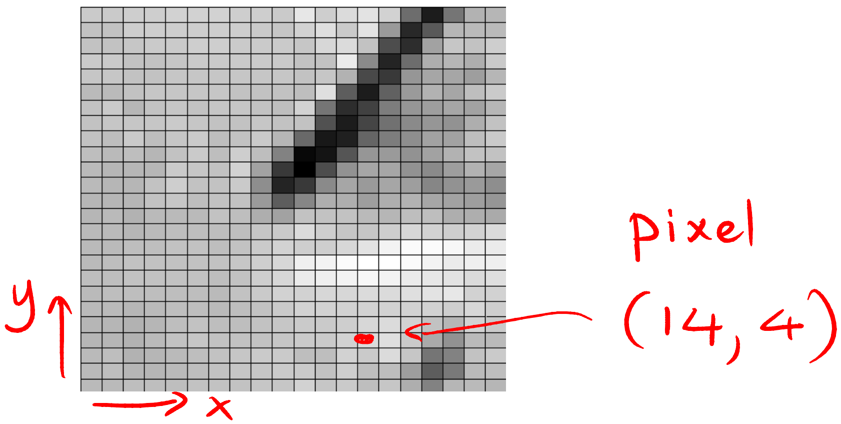
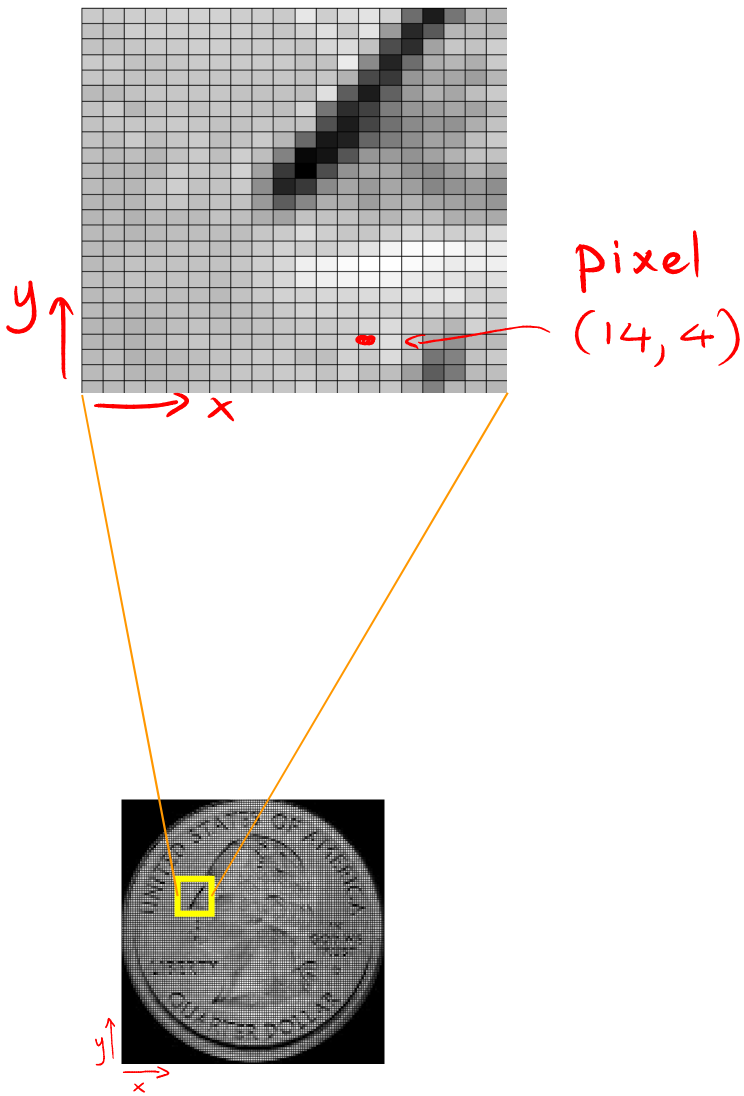
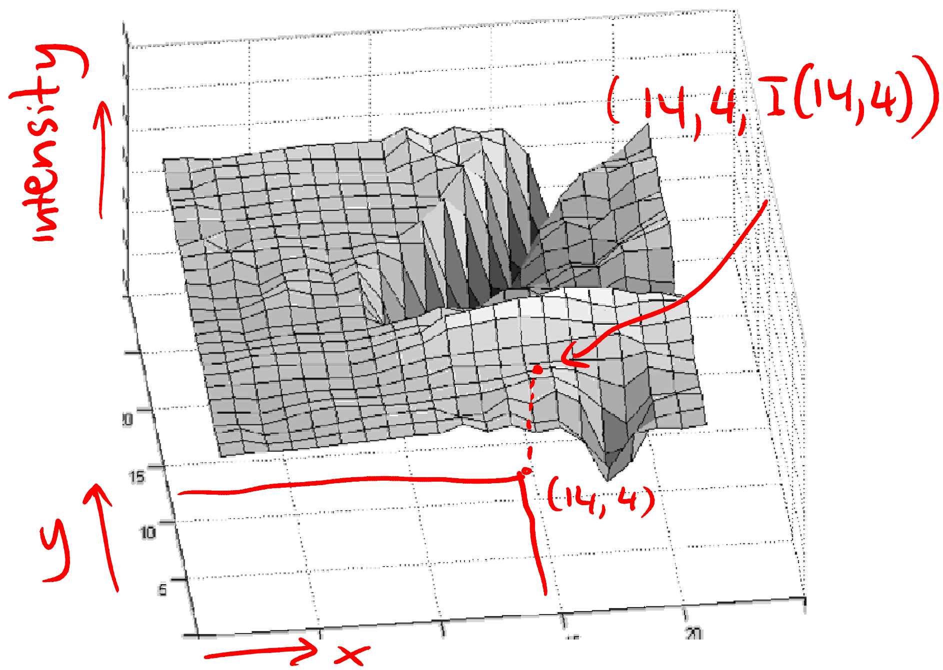


Image ⇔ Surface in 3D

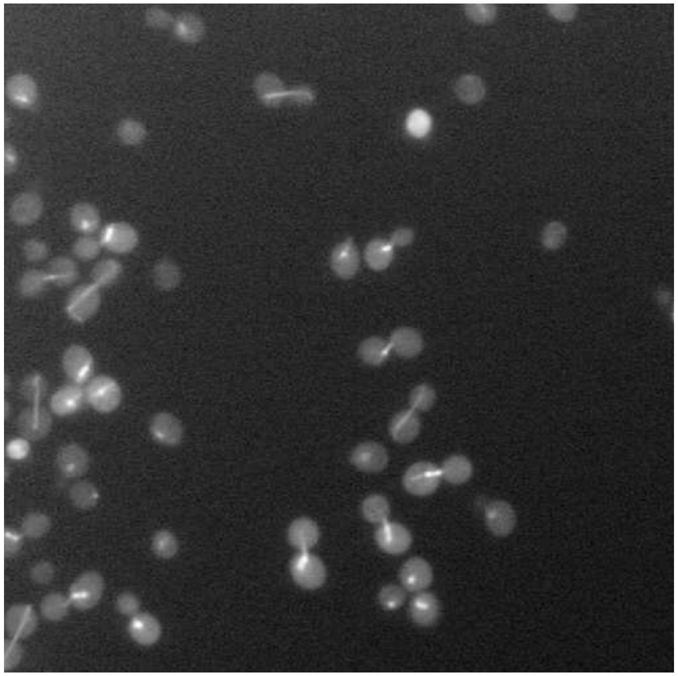
Image patch

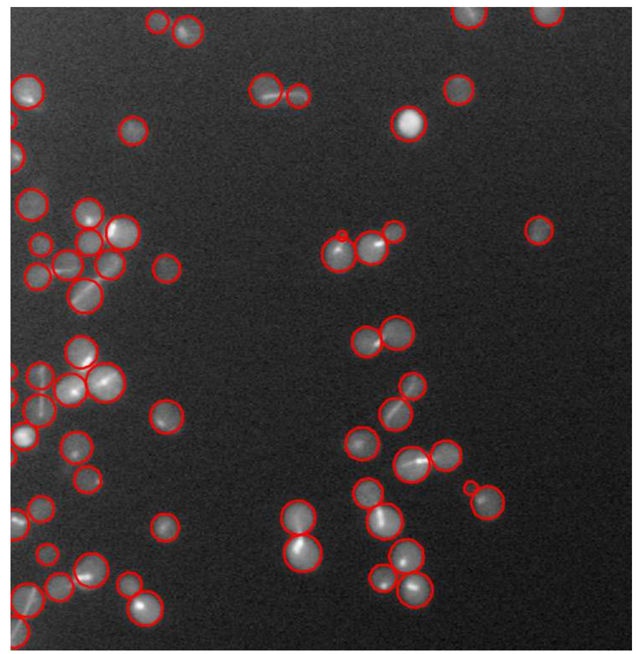


Represented as a surface:



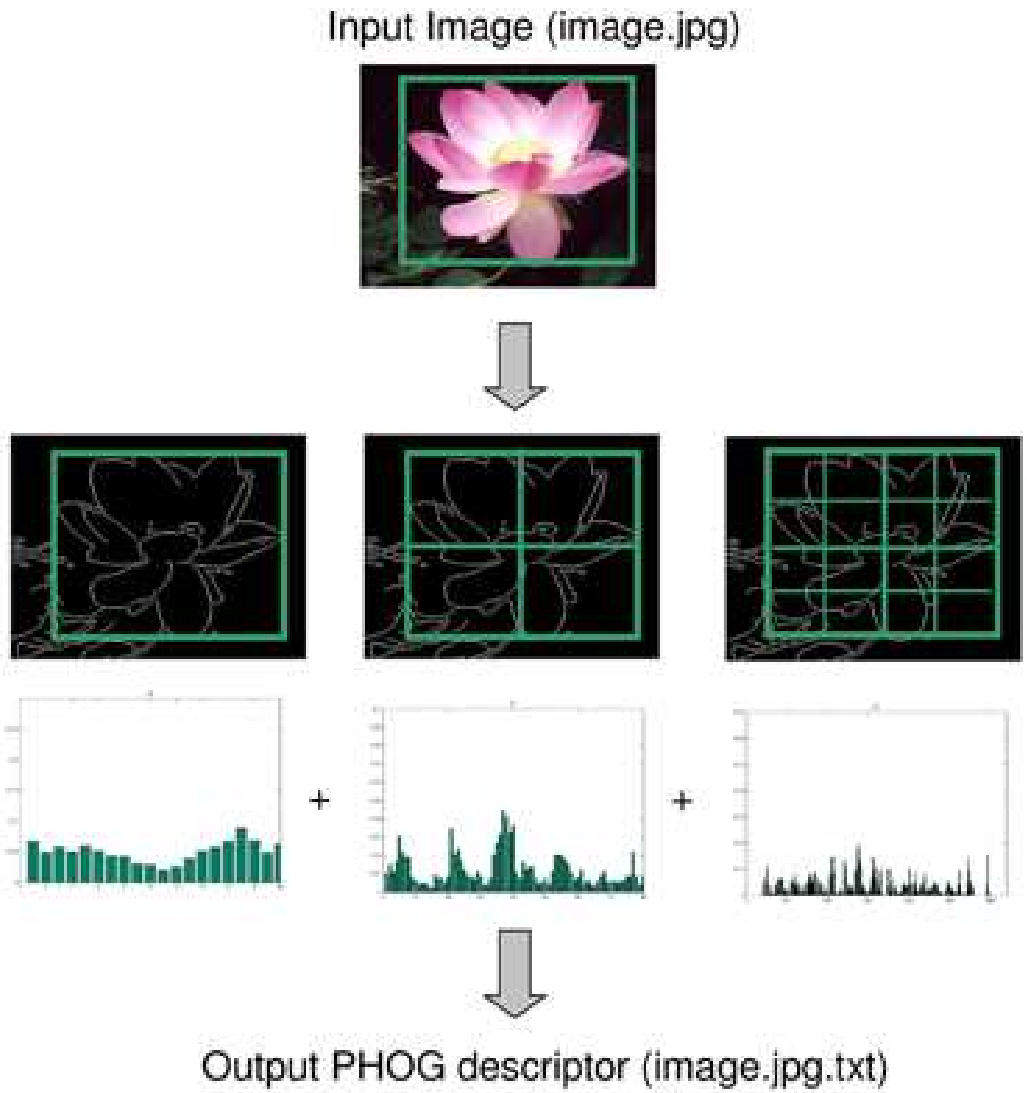
Why: detection





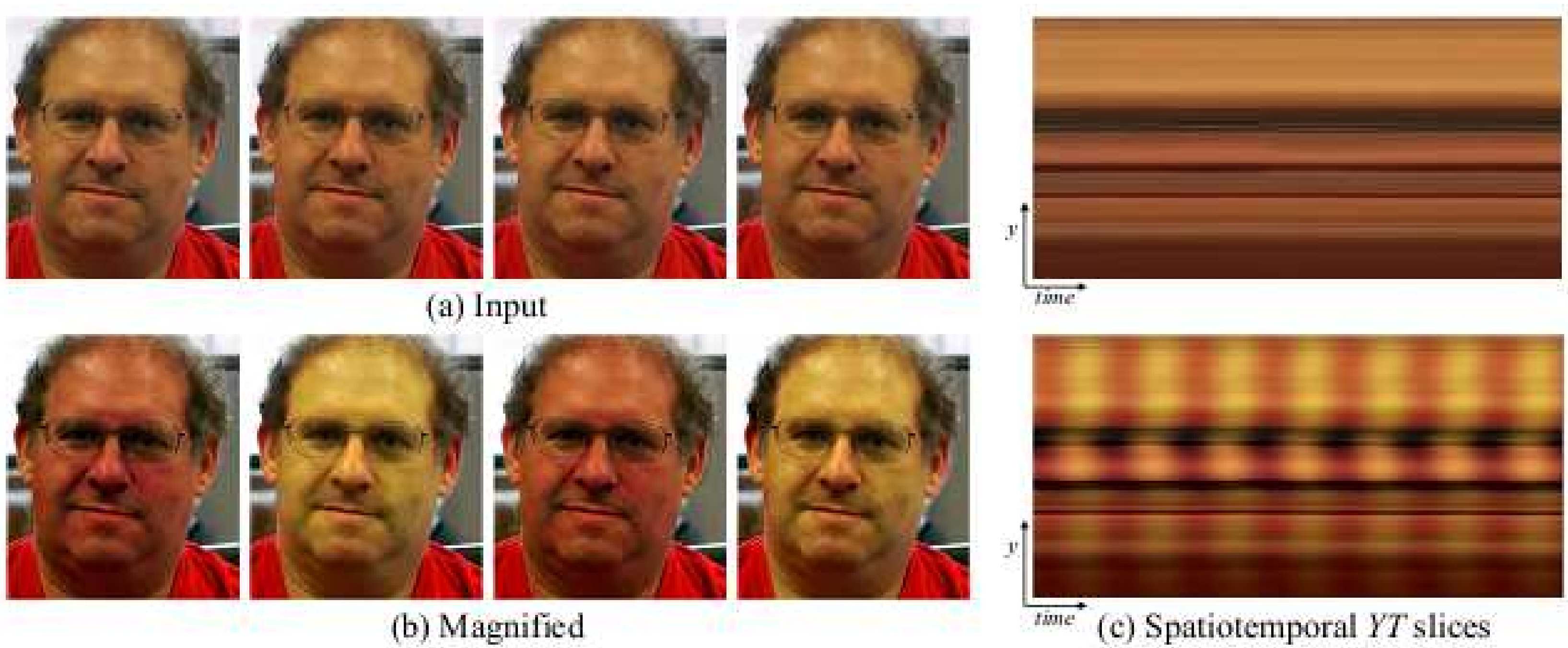
From: http://www.cs.toronto.edu/~jepson/csc420/asgn/a2_11.pdf

Why: recognition



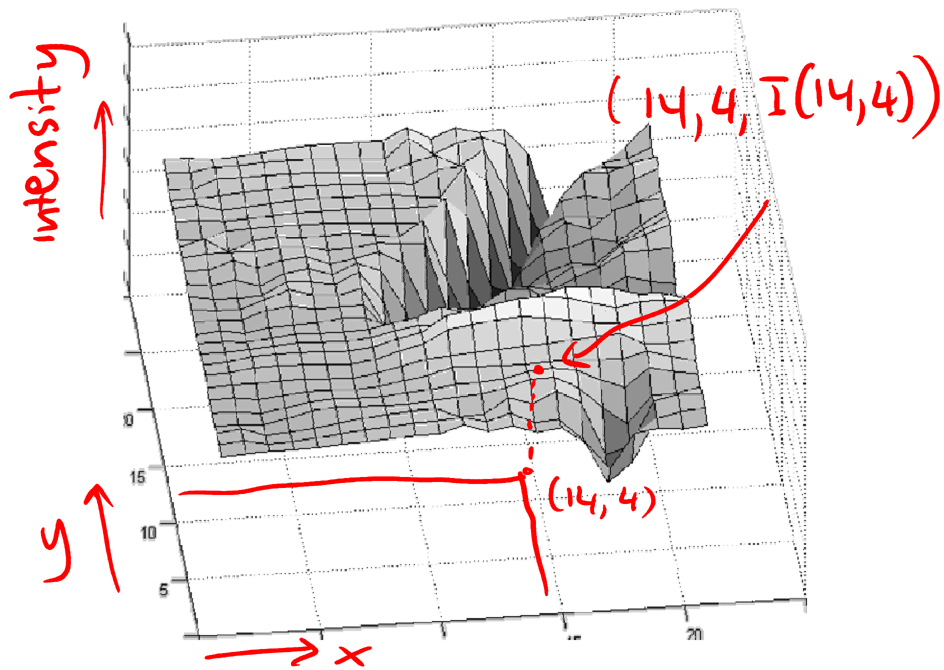
From: http://www.robots.ox.ac.uk/~vgg/research/caltech/phog.html

Why: estimation



From: “Eulerian Video Magnification for Revealing Subtle Changes in the World”, Wu et al.

Estimating I(x,y) in a neighborhood

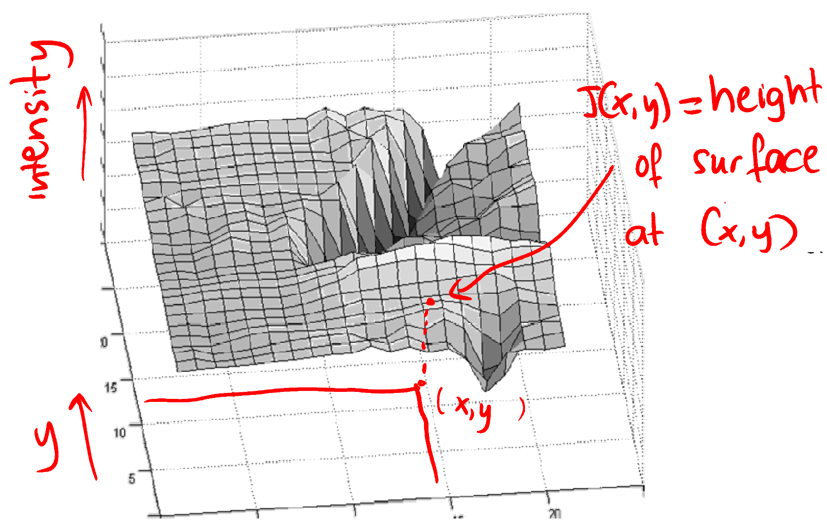


2D Taylor Series Expansion

2D Taylor series expansion near (0,0) with 3 terms:

$$I(x,y) = I(0,0) + x\frac{\partial I}{\partial x}(0,0) + y\frac{\partial I}{\partial y}(0,0) +$$

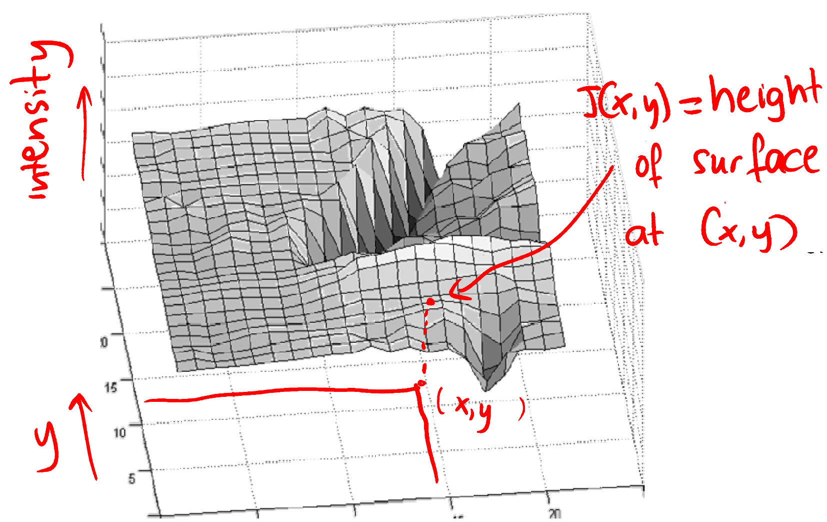
$$\frac{1}{2}\left(x^2\frac{\partial^2 I}{\partial x^2}(0,0) + y^2\frac{\partial^2 I}{\partial x}(0,0) + 2xy\frac{\partial^2 I}{\partial x\partial y}(0,0) \right) + \cdots$$



2D Taylor Series Expansion

2D Taylor series expansion near (0,0) with 3 terms:

$$I(x,y) = \boxed{I(0,0)} + x\frac{\partial I}{\partial x}(0,0) + y\frac{\partial I}{\partial y}(0,0) + \frac{1}{2}\left(x^2\frac{\partial^2 I}{\partial x^2}(0,0) + y^2\frac{\partial^2 I}{\partial x}(0,0) + 2xy\frac{\partial^2 I}{\partial x\partial y}(0,0)\right) + \dots$$

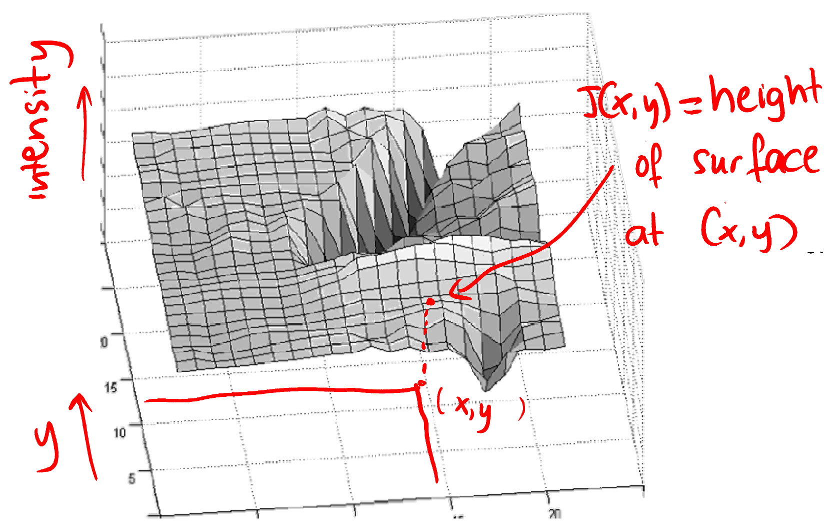


2D Taylor Series Expansion

2D Taylor series expansion near (0,0) with 3 terms:

$$I(x,y) = I(0,0) + \boxed{x\frac{\partial I}{\partial x}(0,0) + y\frac{\partial I}{\partial y}(0,0) +}$$

$$\frac{1}{2}\left(x^2\frac{\partial^2 I}{\partial x^2}(0,0) + y^2\frac{\partial^2 I}{\partial x}(0,0) + 2xy\frac{\partial^2 I}{\partial x \partial y}(0,0)\right) + \dots$$

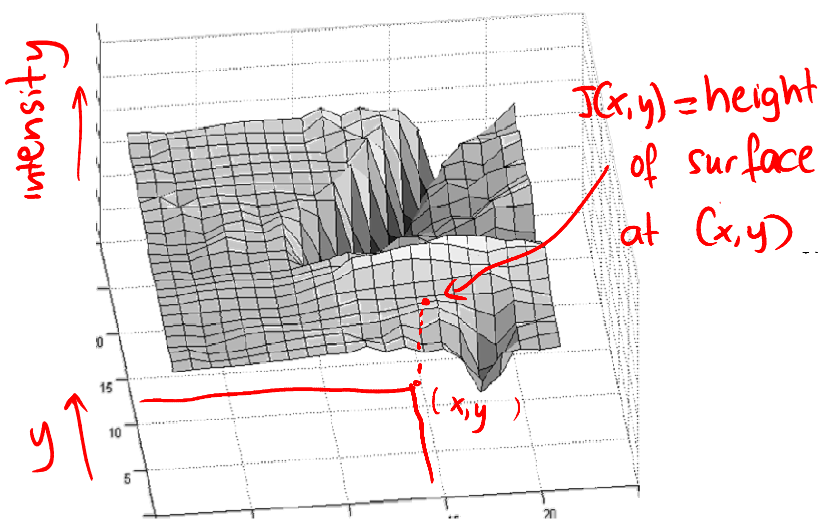


2D Taylor Series Expansion

2D Taylor series expansion near (0,0) with 3 terms:

$$I(x,y) = I(0,0) + x\frac{\partial I}{\partial x}(0,0) + y\frac{\partial I}{\partial y}(0,0) +$$

$$\frac{1}{2}\left(x^2 \frac{\partial^2 I}{\partial x^2}(0,0) + y^2 \frac{\partial^2 I}{\partial x}(0,0) + 2\,xy\,\frac{\partial^2 I}{\partial x \partial y}(0,0) \right) + \ldots$$



Topic 4.3:

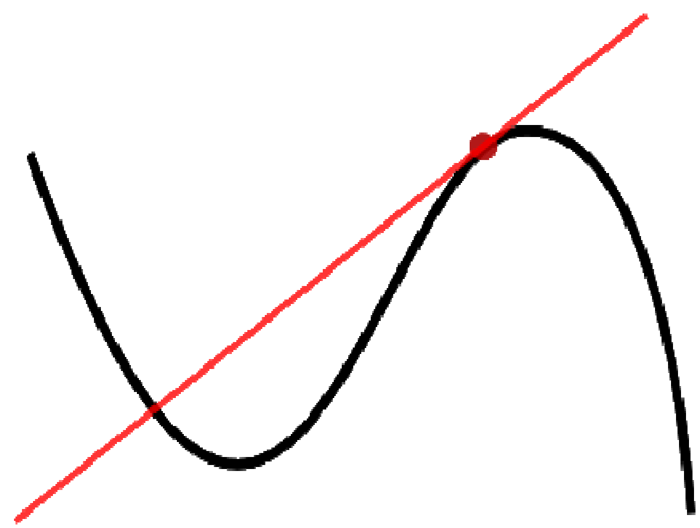
Local analysis of 2D image patches

- Images as surfaces in 3D
- Directional derivatives
- Image Gradient
- Edge detection & localization
  - Gradient extrema
  - Laplacian zero-crossings
- Painterly rendering
- Local geometry at image extrema
- The Image Hessian
- Eigenvectors & eigenvalues
- Corner & feature detection
  - Lowe feature detector
  - Harris/Forstner detector

Computing Directional Image Derivatives

1st order Taylor Series approximation

$$I(x,y) = I(0,0) + x\frac{\partial I}{\partial x}(0,0) + y\frac{\partial I}{\partial y}(0,0)$$

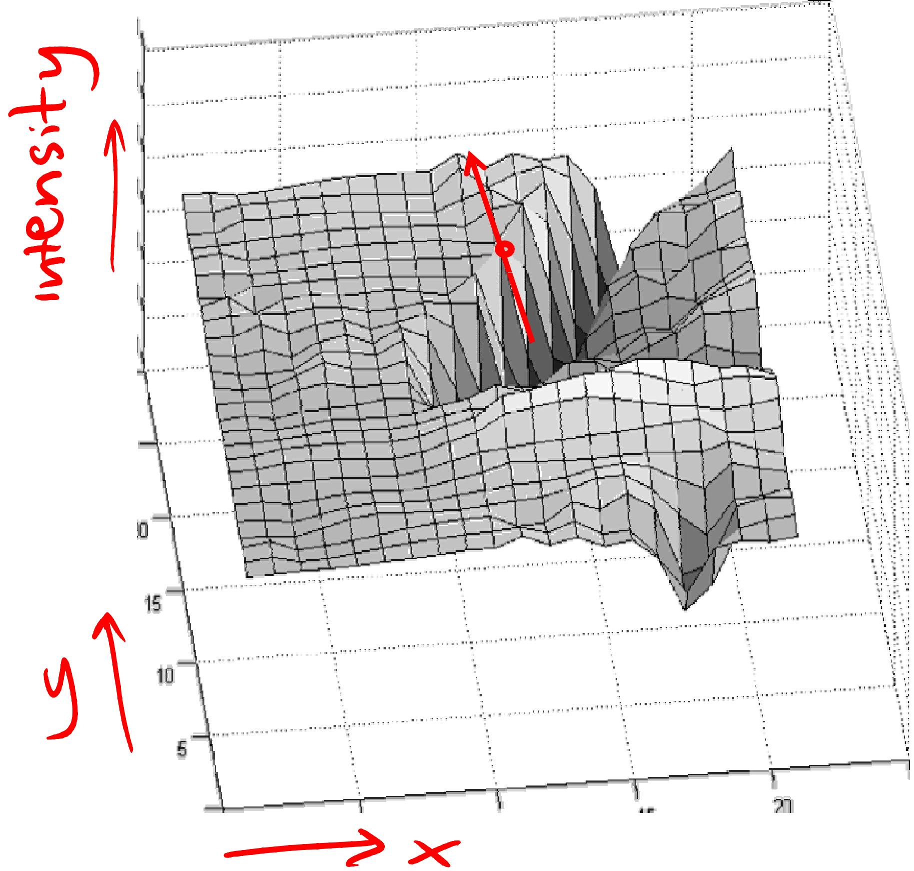


In 1-D

Computing Directional Image Derivatives

1st order Taylor Series approximation

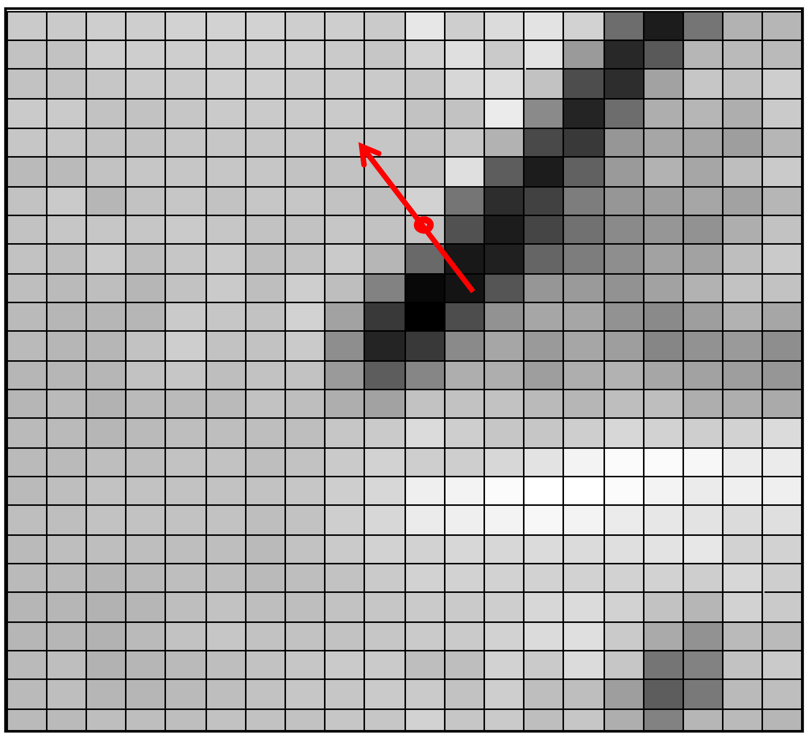
$$I(x,y) = I(0,0) + x\frac{\partial I}{\partial x}(0,0) + y\frac{\partial I}{\partial y}(0,0)$$



Computing Directional Image Derivatives

1st order Taylor Series approximation

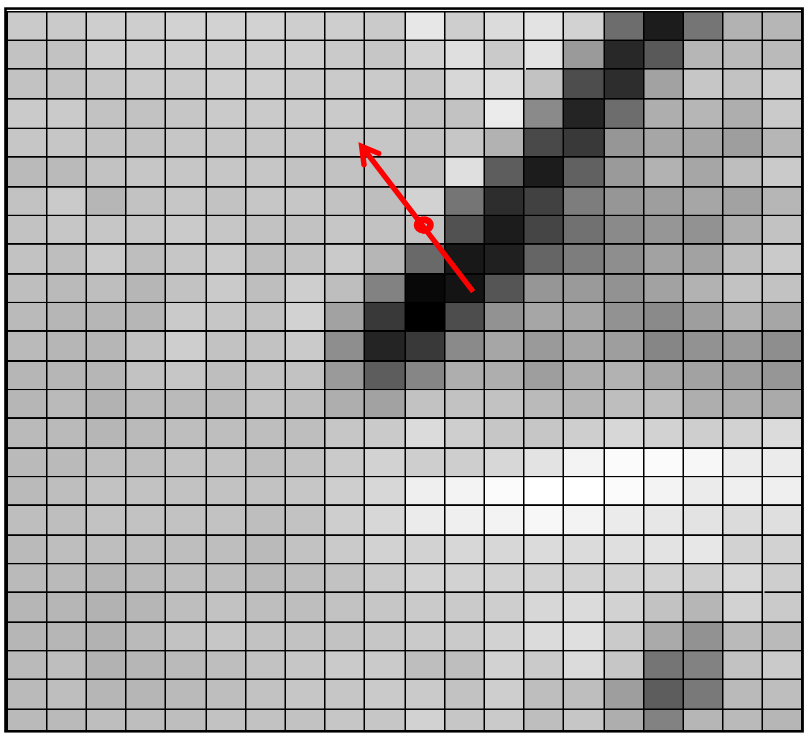
$$I(x,y) = I(0,0) + x\frac{\partial I}{\partial x}(0,0) + y\frac{\partial I}{\partial y}(0,0)$$



Computing Directional Image Derivatives

1st order Taylor Series approximation

$$I(x,y) = I(0,0) + x \frac{\partial I}{\partial x}(0,0) + y \frac{\partial I}{\partial y}(0,0)$$



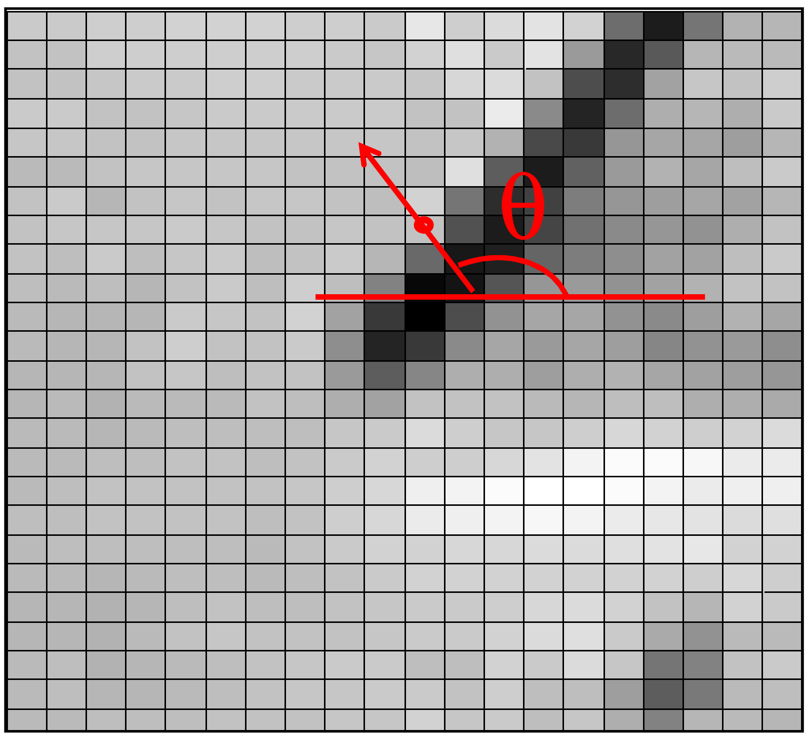
The first derivative tells us the direction of maximum change.

Its magnitude indicates the rate of change (like in 1D).

Computing Directional Image Derivatives

1st order Taylor Series approximation

$$I(x,y) = I(0,0) + x\frac{\partial I}{\partial x}(0,0) + y\frac{\partial I}{\partial y}(0,0)$$

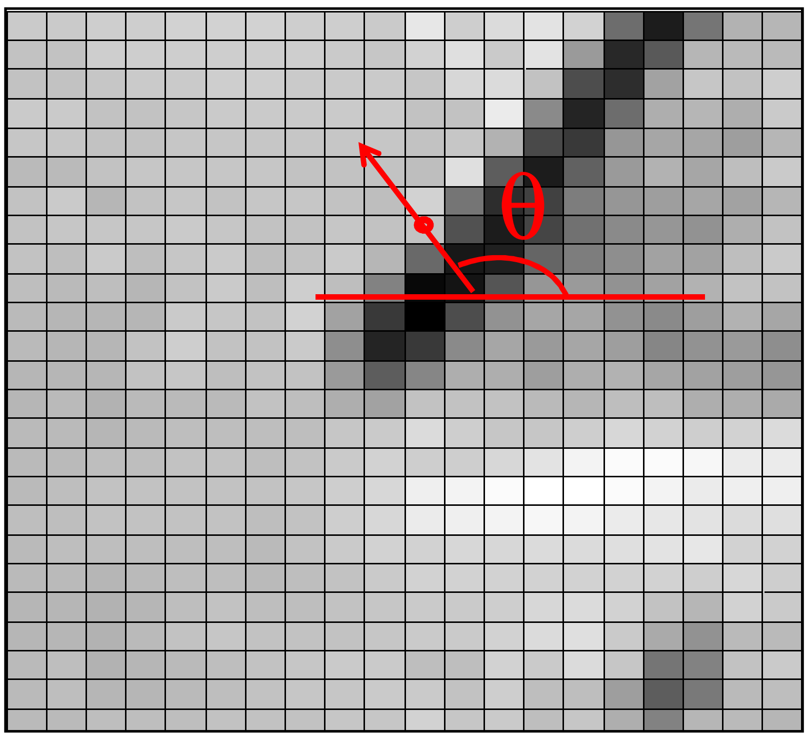


Now, if the function I(x,y) was continuous, what is the intensity I(x,y) along the direction θ?

Computing Directional Image Derivatives

1^{st} order Taylor Series approximation

$$I(x,y) = I(0,0) + x\frac{\partial I}{\partial x}(0,0) + y\frac{\partial I}{\partial y}(0,0)$$



Now, if the function I(x,y) was continuous, what is the intensity I(x,y) along the direction θ?

Walking in the direction of θ can be done by multiplying a constant times a unit vector:

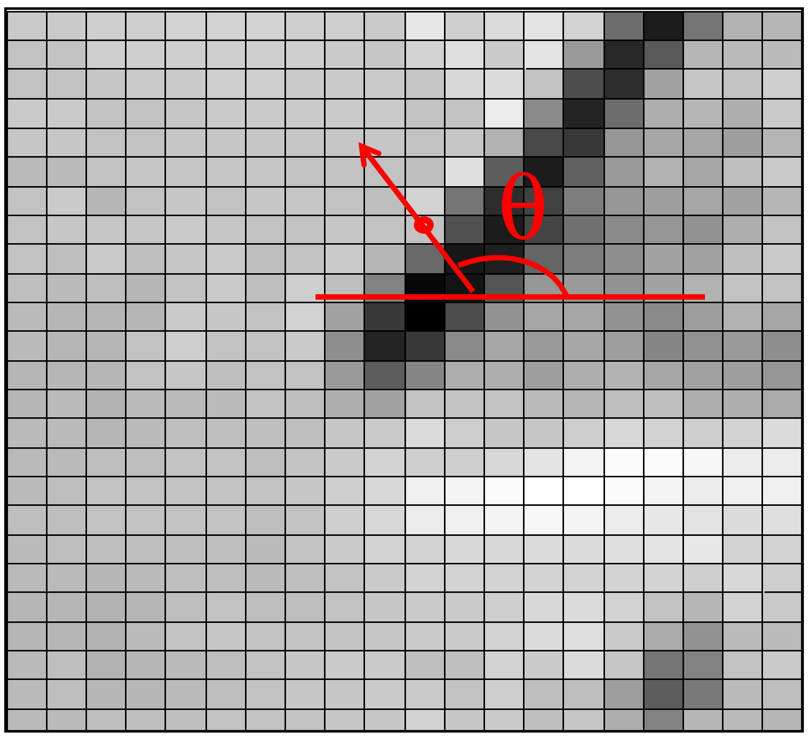
p(t) = t * [cos(θ), sin(θ)]

Unit vector!

Computing Directional Image Derivatives

1st order Taylor Series approximation

$$I(x,y) = I(0,0) + x\frac{\partial I}{\partial x}(0,0) + y\frac{\partial I}{\partial y}(0,0)$$



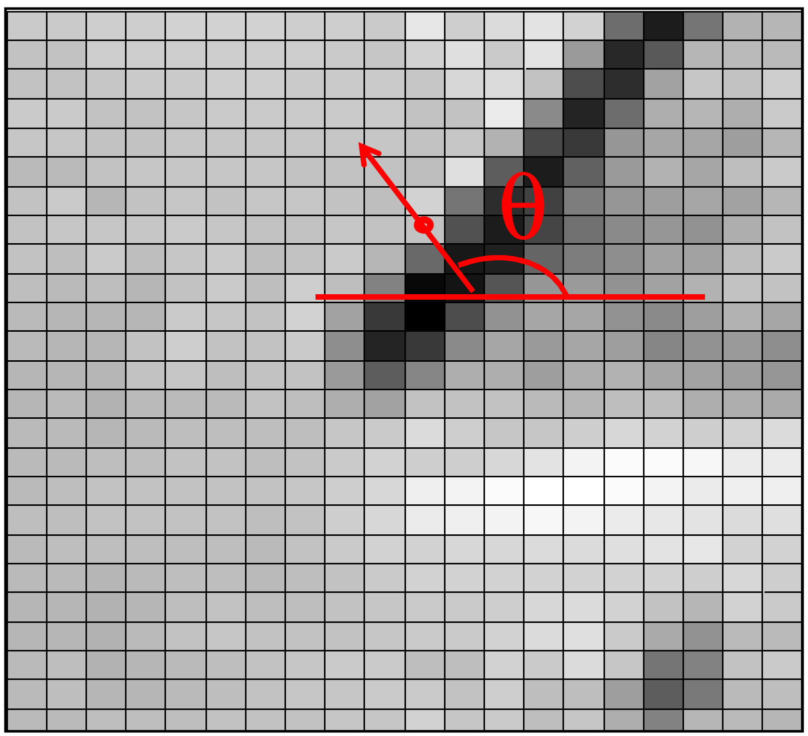
Now, if the function I(x,y) was continuous, what is the intensity I(x,y) along the direction θ?

So, we are really asking what is what is the value of:
I(t cos(θ), t sin(θ))

Computing Directional Image Derivatives

1st order Taylor Series approximation

$$I(x,y) = I(0,0) + x\frac{\partial I}{\partial x}(0,0) + y\frac{\partial I}{\partial y}(0,0)$$



Now, if the function I(x,y) was continuous, what is the intensity I(x,y) along the direction θ?

So, we are really asking what is what is the value of:
I(t cos(θ), t sin(θ))

Ask the Taylor Series approximation!

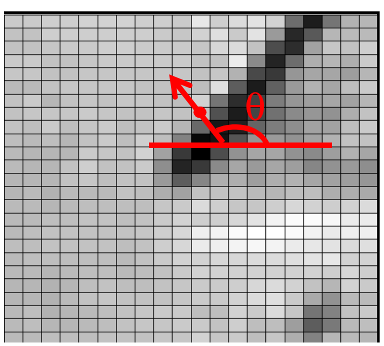
Computing Directional Image Derivatives

1st order Taylor Series approximation

$$I(x,y) = I(0,0) + x\frac{\partial I}{\partial x}(0,0) + y\frac{\partial I}{\partial y}(0,0)$$

Substituting:

$$I(t\cdot\cos\theta, t\cdot\sin\theta) = I(0,0) + t\cdot\cos\theta\cdot\frac{\partial I}{\partial x}(0,0) + t\cdot\sin\theta\,\frac{\partial I}{\partial y}(0,0)$$



Computing Directional Image Derivatives

1st order Taylor Series approximation

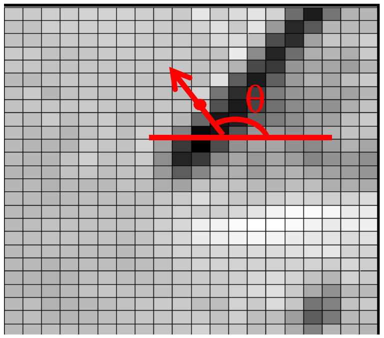
$$I(x,y) = I(0,0) + x\frac{\partial I}{\partial x}(0,0) + y\frac{\partial I}{\partial y}(0,0)$$

Substituting:

$$I(t\cdot\cos\theta, t\cdot\sin\theta) = I(0,0) + t\cdot\cos\theta\cdot\frac{\partial I}{\partial x}(0,0) + t\cdot\sin\theta\,\frac{\partial I}{\partial y}(0,0)$$

Or equivalently:

$$I(t\cdot\cos\theta, t\cdot\sin\theta) = I(0,0) + t\left(\cos\theta\cdot\frac{\partial I}{\partial x}(0,0) + \sin\theta\,\frac{\partial I}{\partial y}(0,0)\right)$$



Computing Directional Image Derivatives

1st order Taylor Series approximation

$$I(x,y) = I(0,0) + x\frac{\partial I}{\partial x}(0,0) + y\frac{\partial I}{\partial y}(0,0)$$

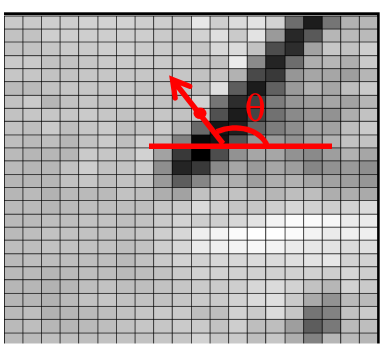
Substituting:

$$I(t\cdot\cos\theta, t\cdot\sin\theta) = I(0,0) + t\cdot\cos\theta\cdot\frac{\partial I}{\partial x}(0,0) + t\cdot\sin\theta\,\frac{\partial I}{\partial y}(0,0)$$

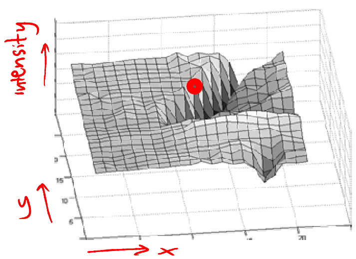
Or equivalently:

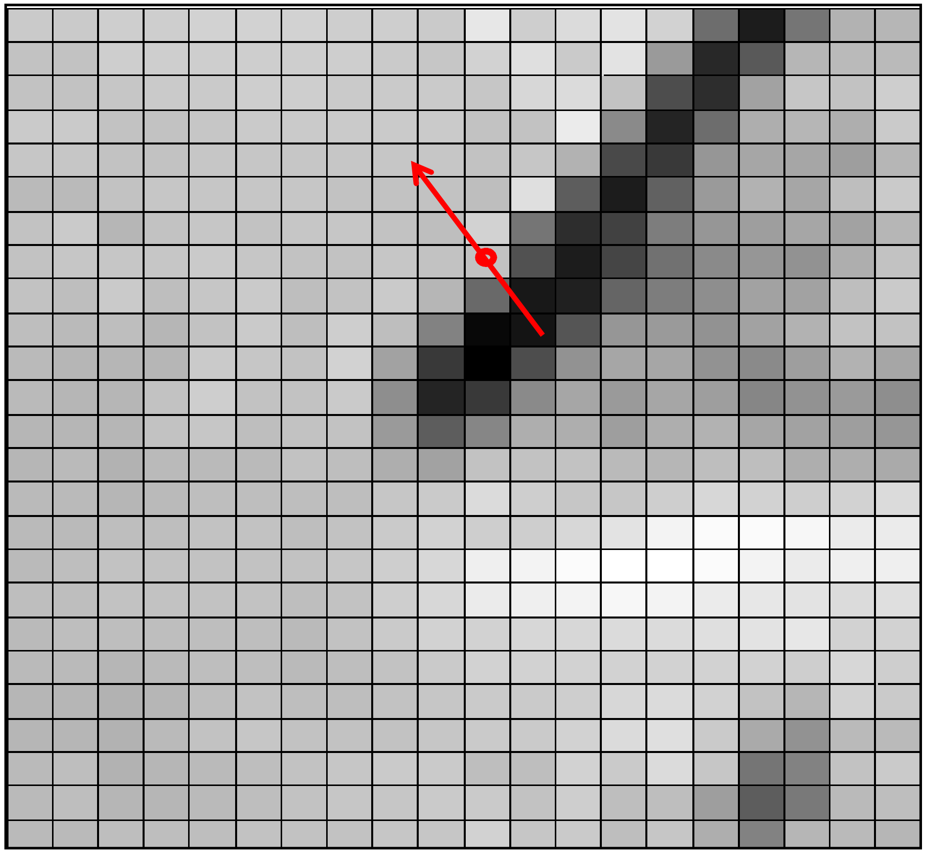
$$I(t\cdot\cos\theta, t\cdot\sin\theta) = I(0,0) + t\left(\cos\theta\cdot\frac{\partial I}{\partial x}(0,0) + \sin\theta\,\frac{\partial I}{\partial y}(0,0)\right)$$

Directional Derivative of I(x,y) in the direction of [cos(θ), sin(θ)]



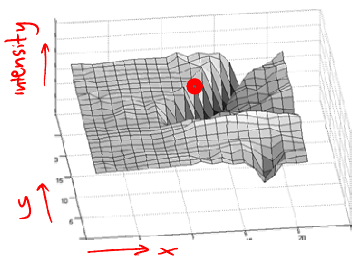
Computing Directional Image Derivatives

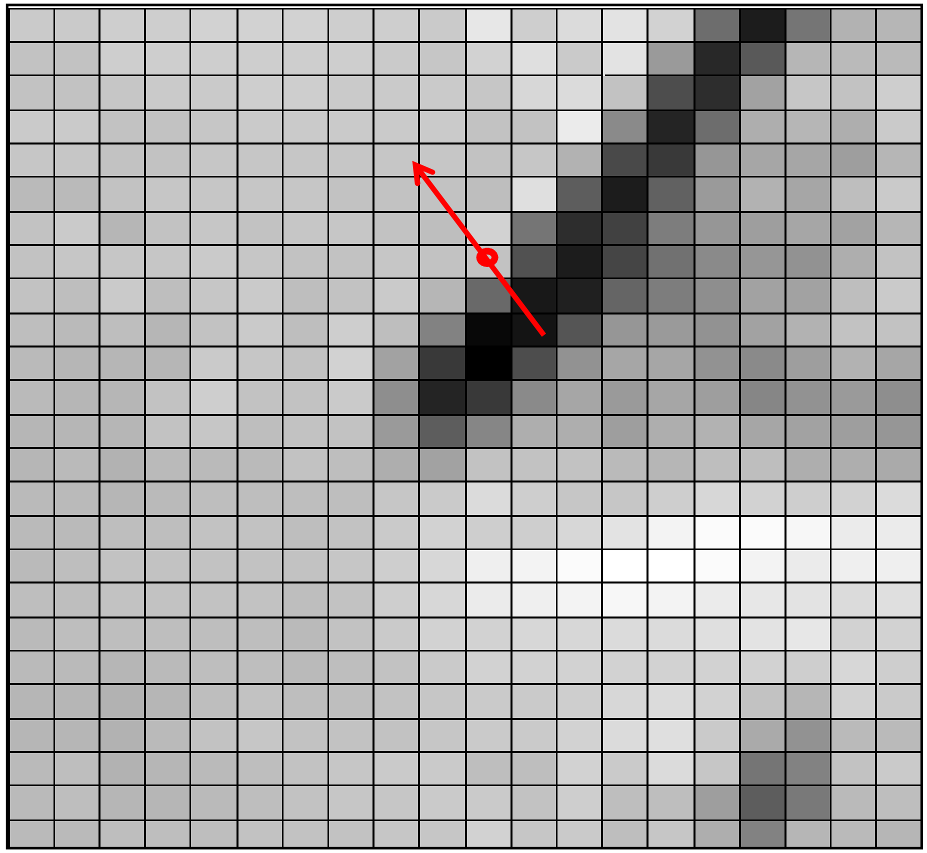




Directional derivative?

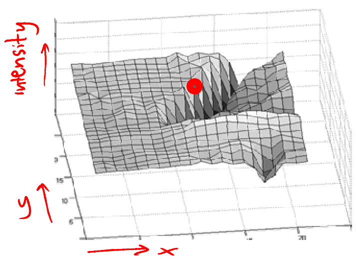
Computing Directional Image Derivatives

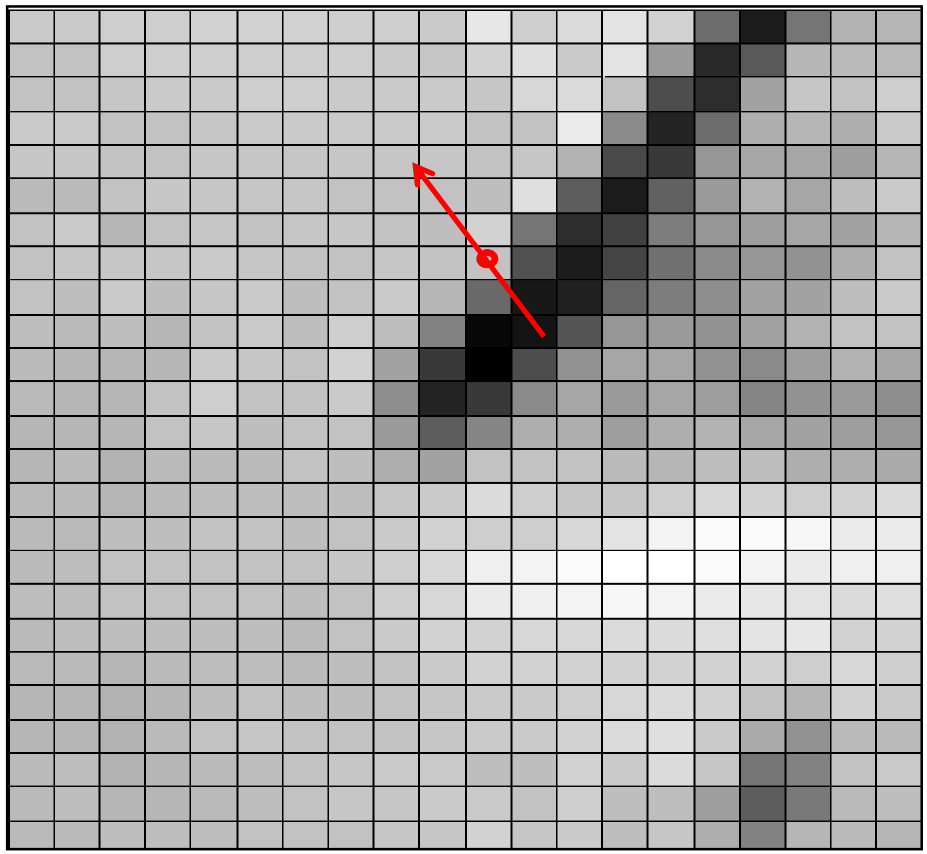




Directional derivative: rate of change in the given direction

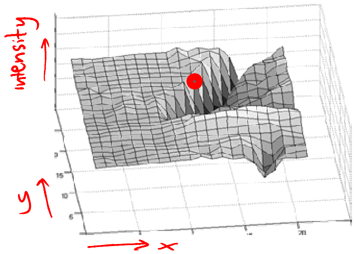
Computing Directional Image Derivatives

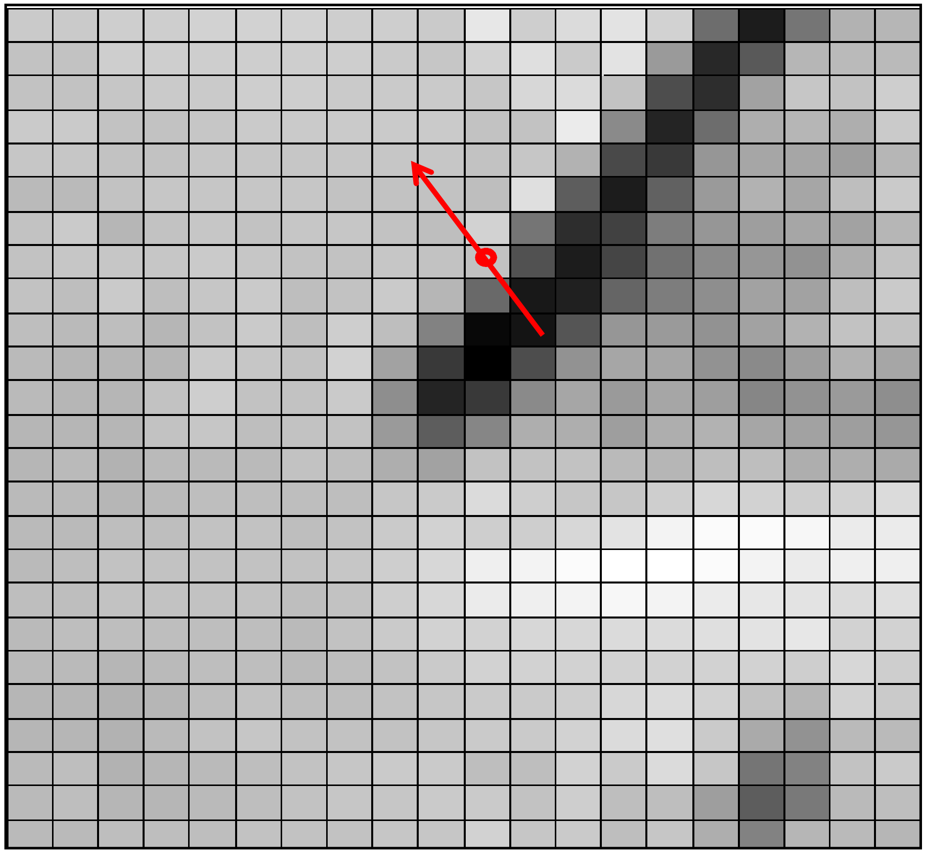




What is it for the red dot?

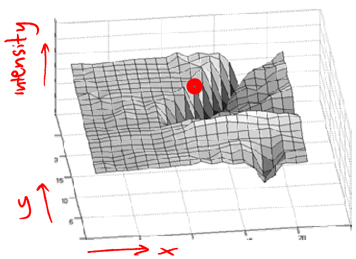
Computing Directional Image Derivatives

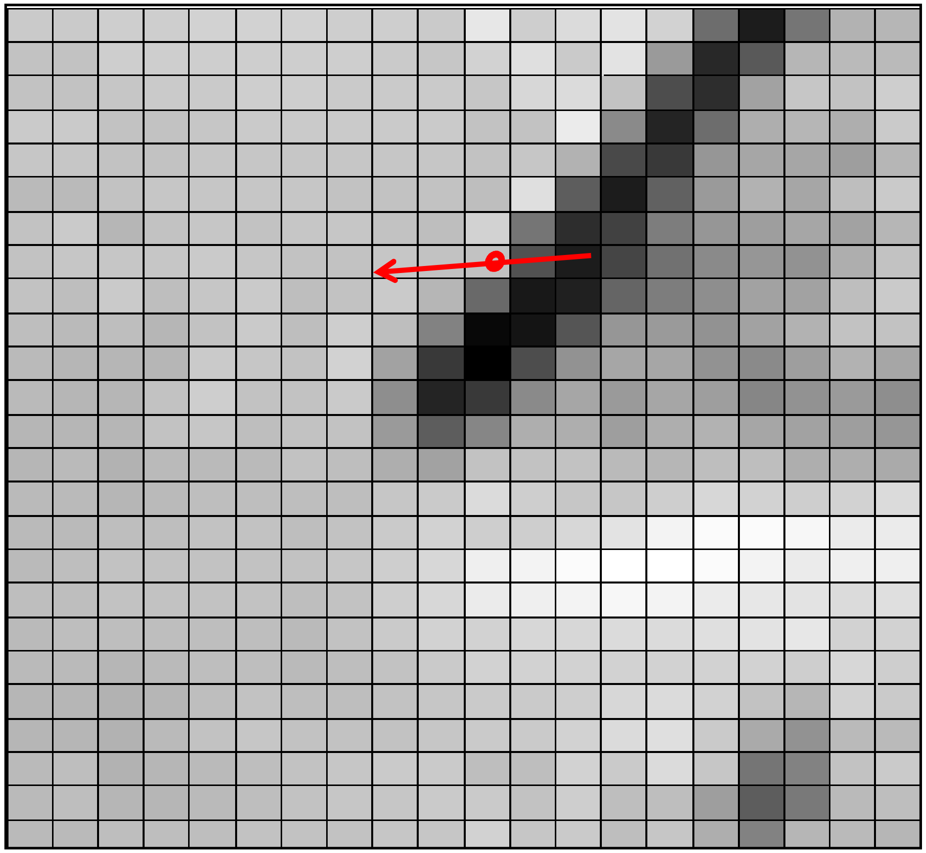




Large and positive

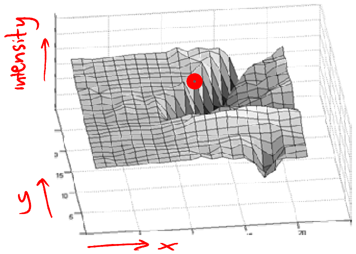
Computing Directional Image Derivatives

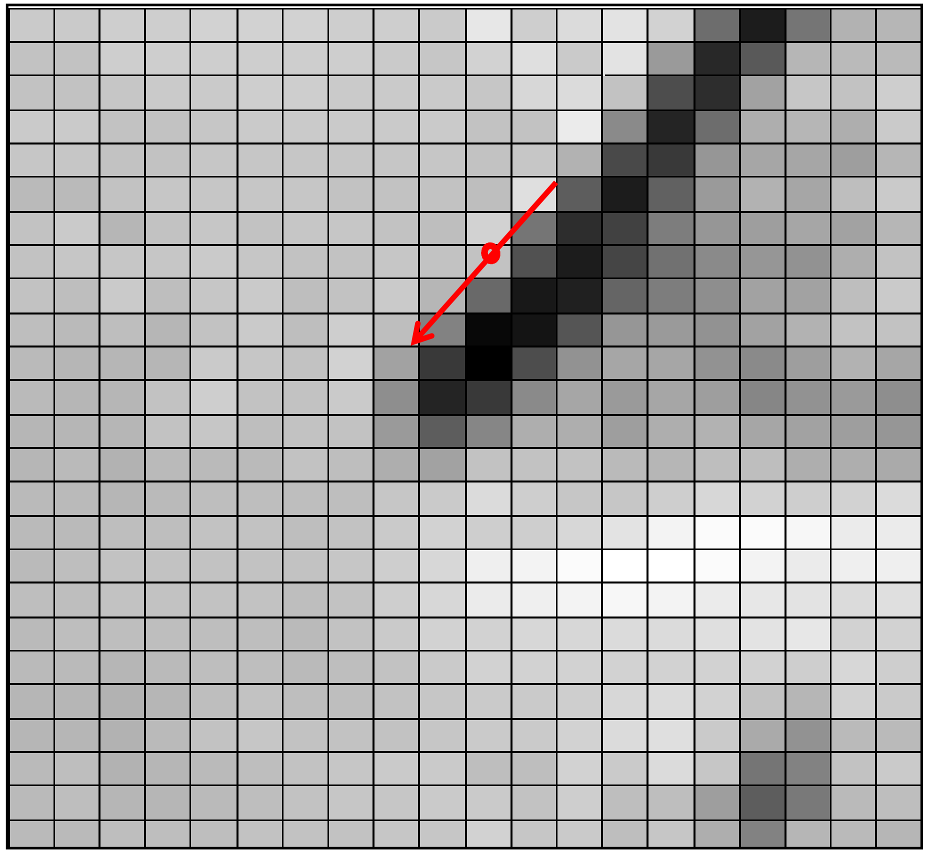




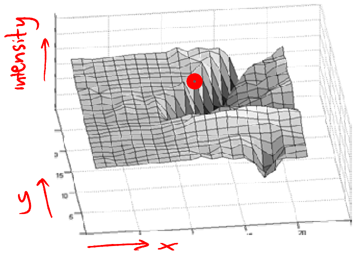
Positive

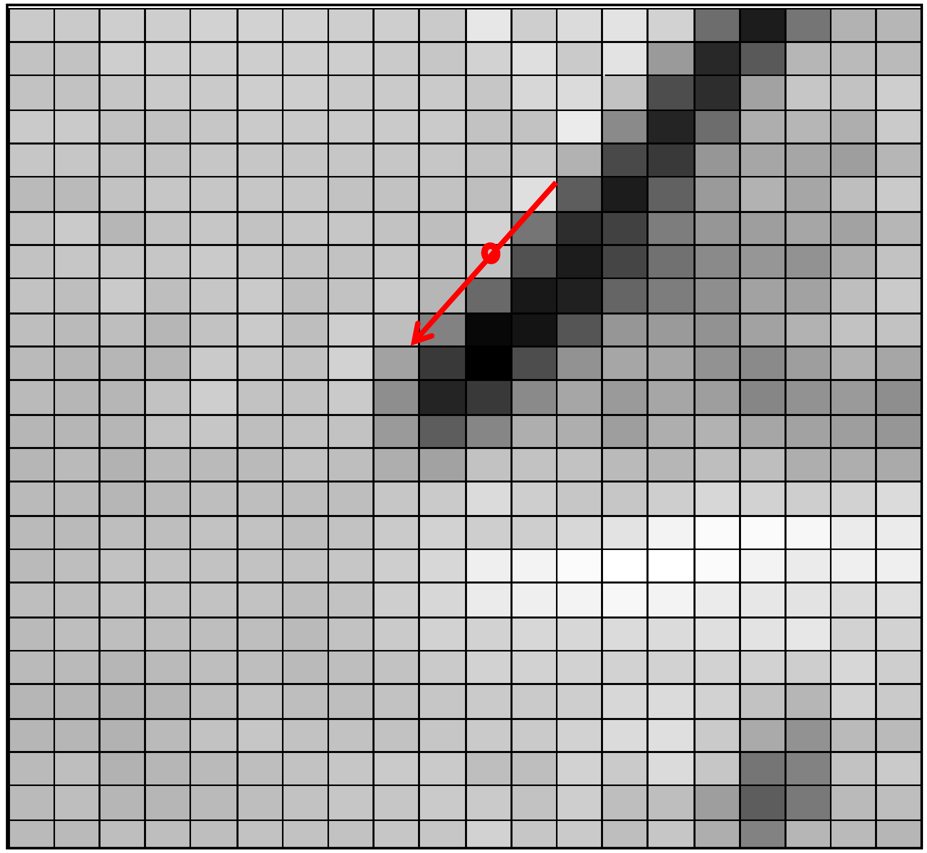
Computing Directional Image Derivatives





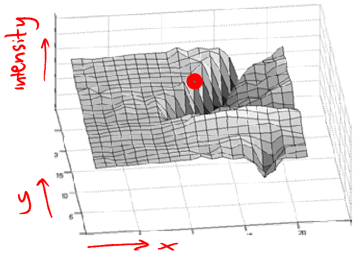
Computing Directional Image Derivatives

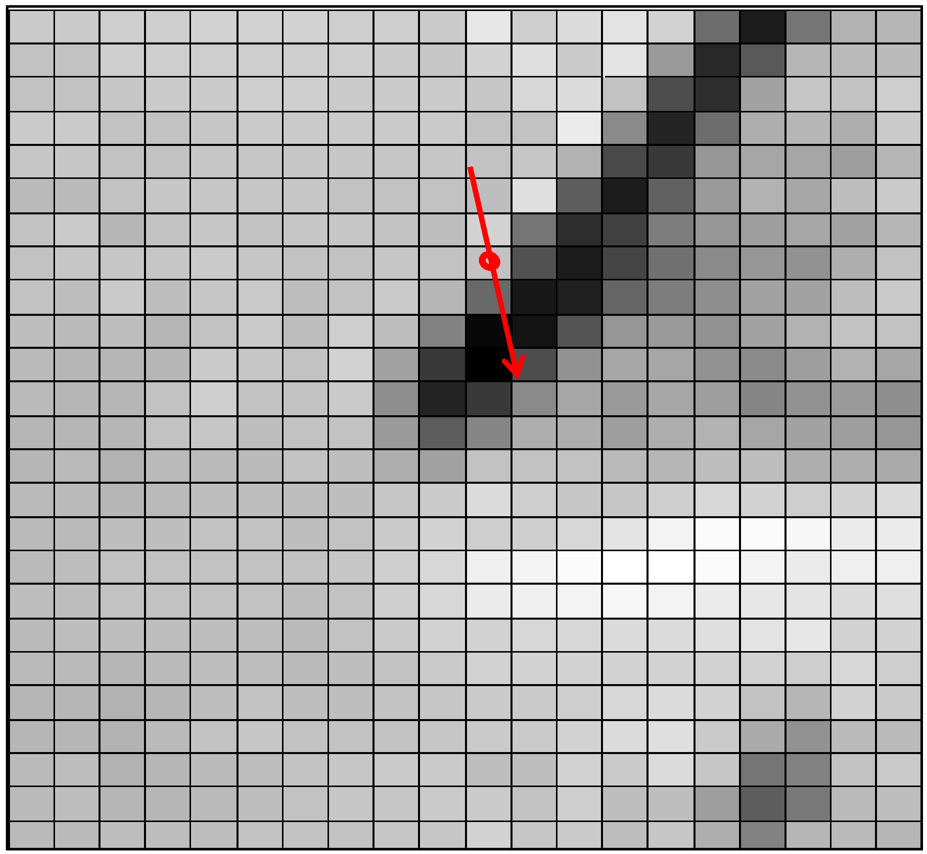




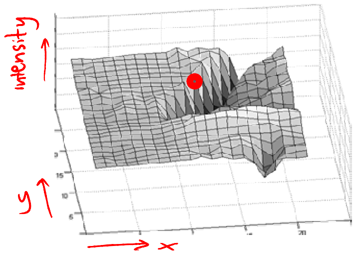
Close to zero

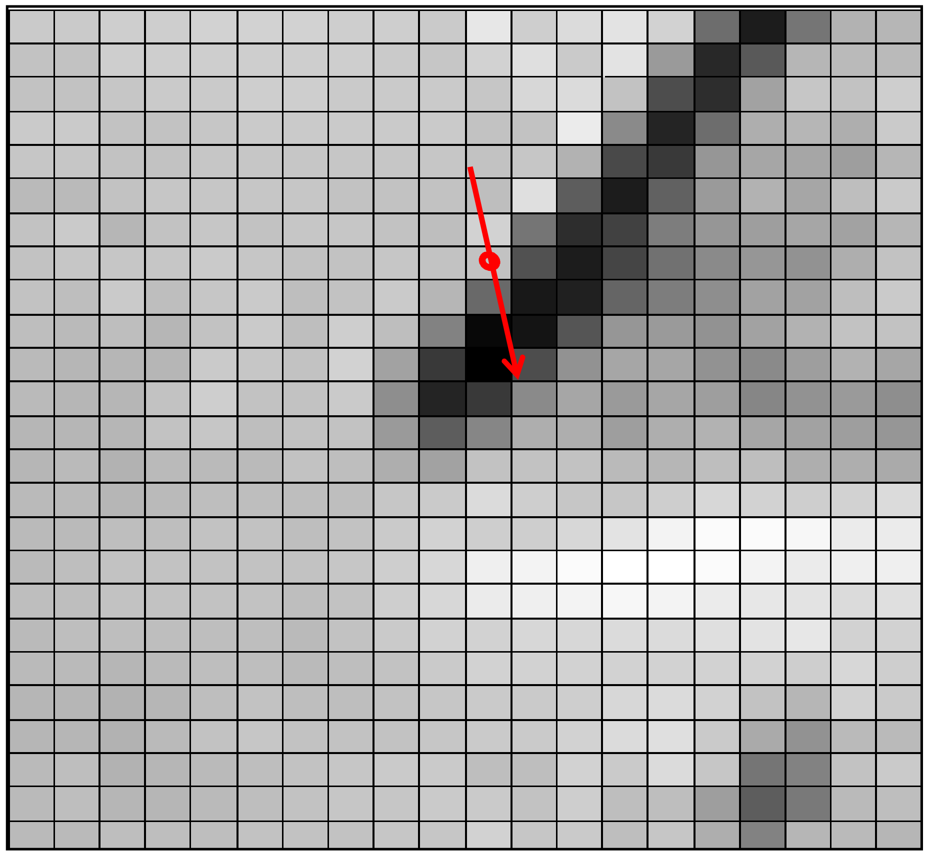
Computing Directional Image Derivatives





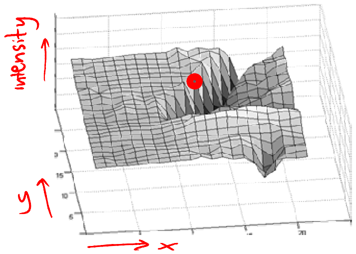
Computing Directional Image Derivatives

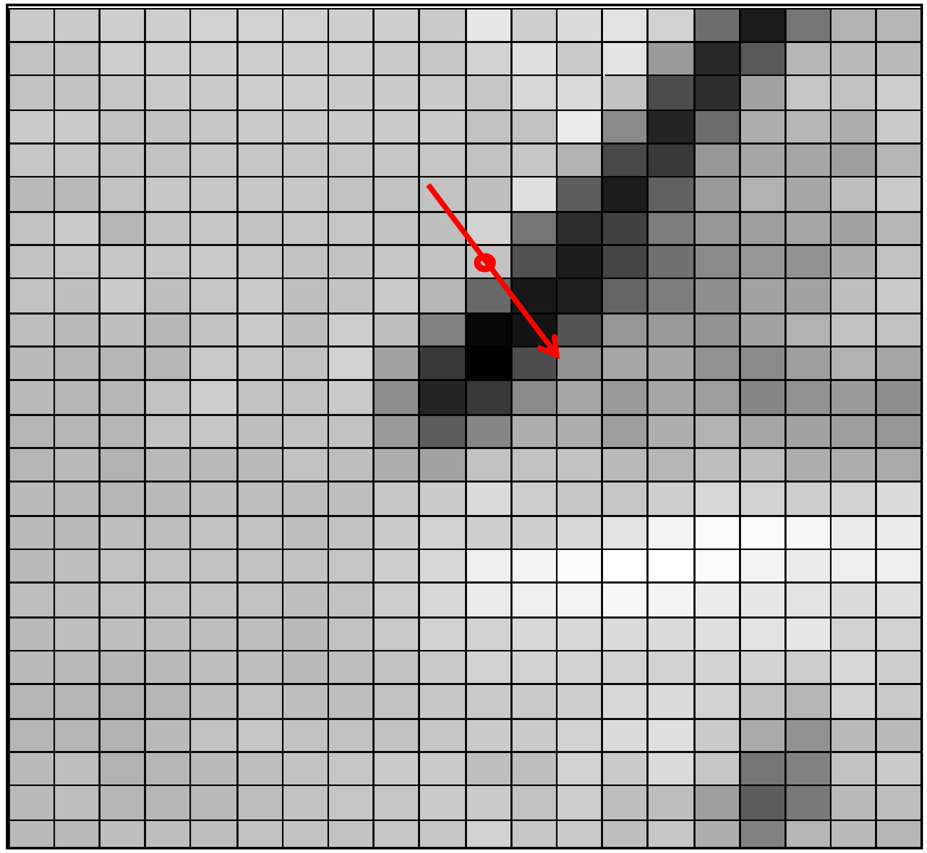




Negative

Computing Directional Image Derivatives





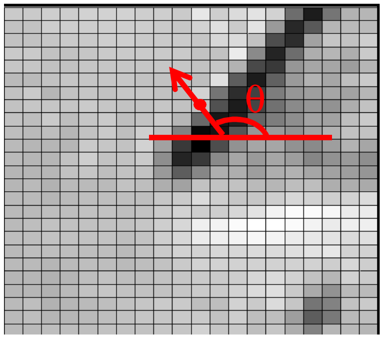
Large and negative

Computing Directional Image Derivatives

$$I(t\cdot\cos\theta, t\cdot\sin\theta) = I(0,0) + t\left(\cos\theta\cdot\frac{\partial I}{\partial x}(0,0) + \sin\theta\,\frac{\partial I}{\partial y}(0,0)\right)$$

Directional Derivative of I(x,y) in the direction of [cos(θ), sin(θ)]

Or in matrix form: $\left[\frac{\partial I}{\partial x}(0,0)\quad \frac{\partial I}{\partial y}(0,0)\right]\cdot\begin{bmatrix}\cos\theta\\ \sin\theta\end{bmatrix}$

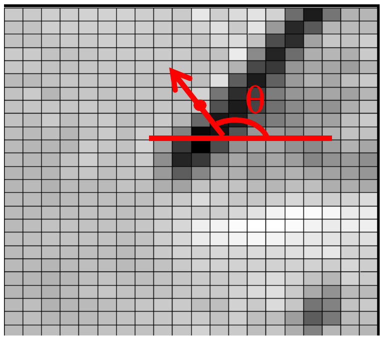


Computing Directional Image Derivatives

$$\left[\frac{\partial I}{\partial x}(0,0) \quad \frac{\partial I}{\partial y}(0,0)\right] \cdot \begin{bmatrix} \cos\theta \\ \sin\theta \end{bmatrix}$$

→ Directional derivative in the direction of [cos(θ), sin(θ)]

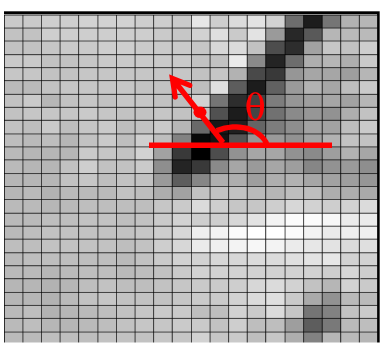
When is this maximum?



Computing Directional Image Derivatives

$$\left[\frac{\partial I}{\partial x}(0,0) \quad \frac{\partial I}{\partial y}(0,0) \right] \cdot \begin{bmatrix} \cos\theta \\ \sin\theta \end{bmatrix}$$ → Directional derivative in the direction of [cos(θ), sin(θ)]

$$[\cos\theta \quad \sin\theta] = \left[\frac{\partial I}{\partial x}(0,0) \quad \frac{\partial I}{\partial y}(0,0) \right]$$ → Maximum



Computing Directional Image Derivatives

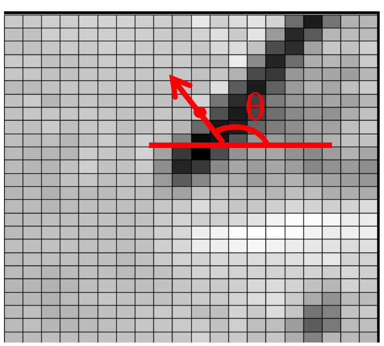
$$\left[\frac{\partial I}{\partial x}(0,0) \quad \frac{\partial I}{\partial y}(0,0)\right] \cdot \begin{bmatrix}\cos\theta \\ \sin\theta\end{bmatrix}$$

→ Directional derivative in the direction of [cos(θ), sin(θ)]

$$[\cos\theta \quad \sin\theta] = \left[\frac{\partial I}{\partial x}(0,0) \quad \frac{\partial I}{\partial y}(0,0)\right]$$

→ Maximum

When is it zero?



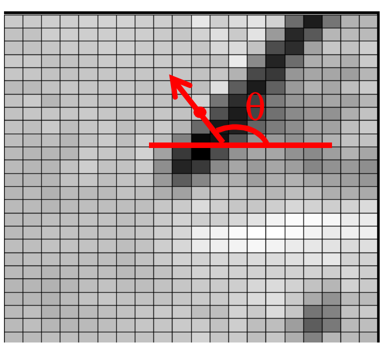
Computing Directional Image Derivatives

$$\left[\frac{\partial I}{\partial x}(0,0) \quad \frac{\partial I}{\partial y}(0,0)\right] \cdot \begin{bmatrix} \cos\theta \\ \sin\theta \end{bmatrix}$$ → Directional derivative in the direction of [cos(θ), sin(θ)]

$$[\cos\theta \quad \sin\theta] = \left[\frac{\partial I}{\partial x}(0,0) \quad \frac{\partial I}{\partial y}(0,0)\right]$$ → Maximum

orthogonal

$$[\cos\theta \quad \sin\theta] \perp \left[\frac{\partial I}{\partial x}(0,0) \quad \frac{\partial I}{\partial y}(0,0)\right]$$ → Zero



Computing Directional Image Derivatives

$$\left[\frac{\partial I}{\partial x}(0,0) \quad \frac{\partial I}{\partial y}(0,0)\right] \cdot \begin{bmatrix} \cos\theta \\ \sin\theta \end{bmatrix}$$

→ Directional derivative in the direction of [cos(θ), sin(θ)]

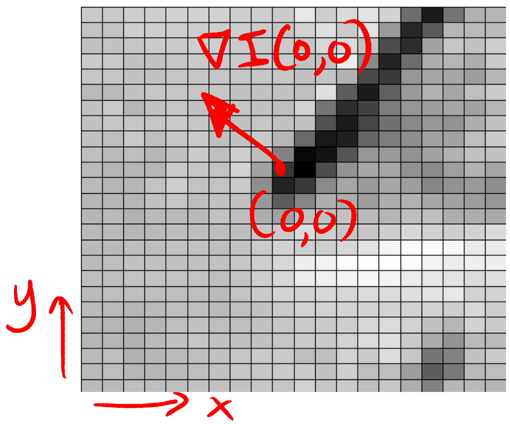
Directional Derivative in any direction can be computed from these two!

Topic 4.3:

Local analysis of 2D image patches

- Images as surfaces in 3D
- Directional derivatives
- Image Gradient
- Edge detection & localization
    - Gradient extrema
    - Laplacian zero-crossings
- Painterly rendering
- Local geometry at image extrema
- The Image Hessian
- Eigenvectors & eigenvalues
- Corner & feature detection
    - Lowe feature detector
    - Harris/Forstner detector

The Image Gradient & Its Properties



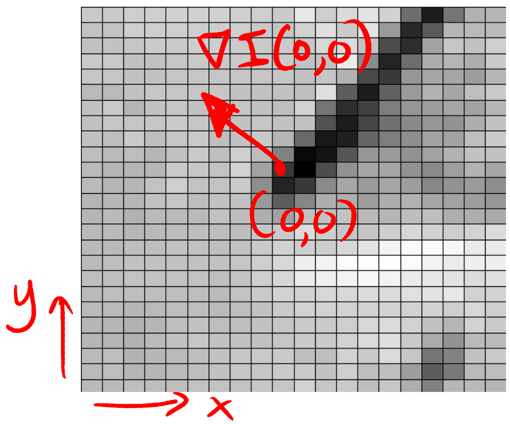
In general the Image gradient is the vector of first derivatives

$$\nabla I(x,y) = \left[\frac{\partial I}{\partial x}(x,y) \quad \frac{\partial I}{\partial y}(x,y) \right]$$

And the directional derivative along a direction vector 'v' can then be defined as:

$$D_v(x,y) = \nabla I(x,y) \cdot v$$

The Image Gradient & Its Properties



The directional derivative:

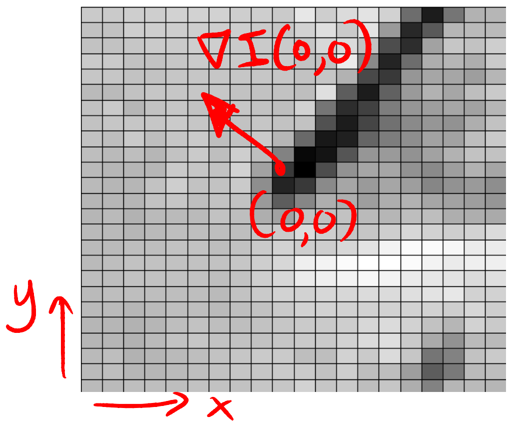
$$D_v(x,y) = \nabla I(x,y) \cdot v$$

Is maximum when

$$v = \nabla I(x,y)$$

And zero when v and $\nabla I(x,y)$ are orthogonal.

The Image Gradient & Its Properties



The directional derivative:

$$D_v(x,y) = \nabla I(x,y) \cdot v$$

Is maximum when

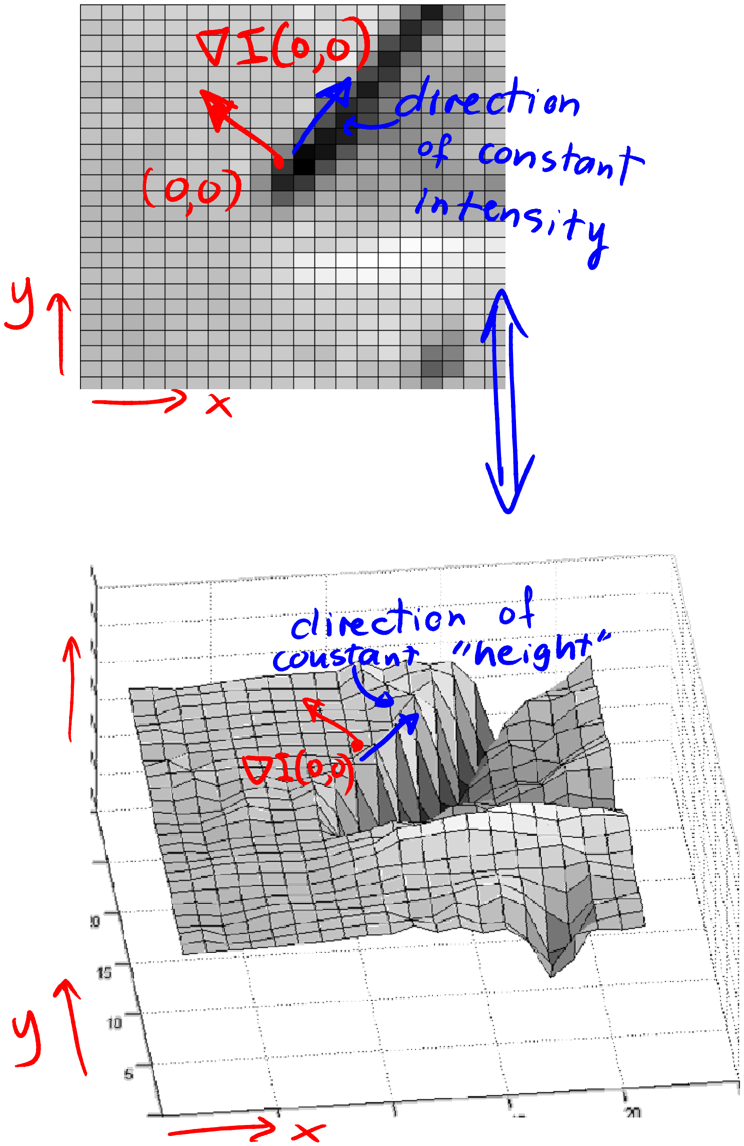
$$v = \nabla I(x,y)$$

And zero when v and $\nabla I(x,y)$ are orthogonal, in which case:

$$I(t\cdot\cos\theta, t\cdot\sin\theta) = I(0,0) + t\left(\cos\theta \cdot \frac{\partial I}{\partial x}(0,0) + \sin\theta \frac{\partial I}{\partial y}(0,0)\right)$$

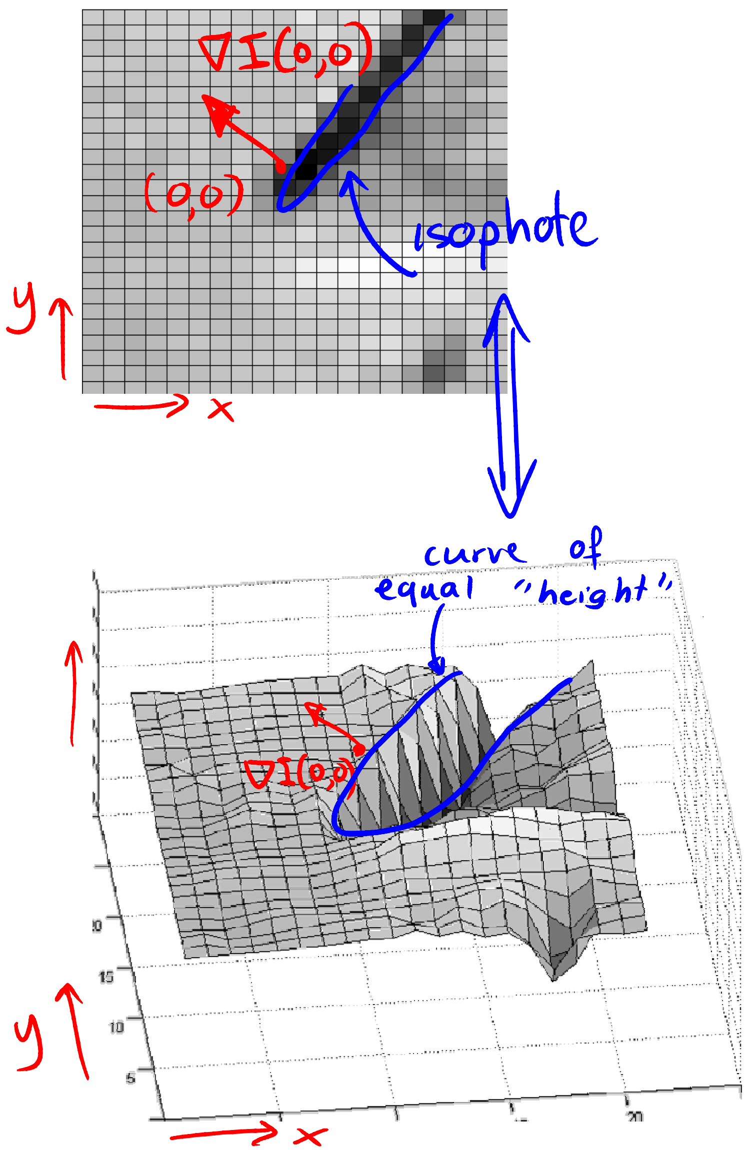
$$I(t\cdot\cos\theta, t\cdot\sin\theta) = I(0,0)$$

The Image Gradient & Its Properties



Note then how the gradient $\nabla I(x,y)$ is the normal vector of the isointensity curve (aka isophote) through pixel (x,y).

The Image Gradient & Its Properties



Note then how the gradient $\nabla I(x,y)$ is the normal vector of the isointensity curve (aka isophote) through pixel (x,y).

Great, but how do we compute $\nabla I(x,y)\cdot$ from image data?

Computing & Visualizing Gradients

Compute $\nabla I(x,y) = \left[\frac{\partial I}{\partial x}(x,y) \quad \frac{\partial I}{\partial y}(x,y) \right]$ at each pixel.

Step 1: Compute a Grayscale Image

Start by computing a one-dimensional I(x,y) (grayscale image) by doing:

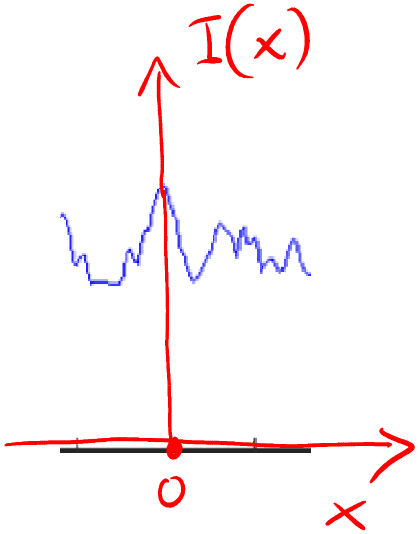
$$I(x,y) = 1/3 * (Red\ (x,y) + Green(x,y) + Blue(x,y))$$

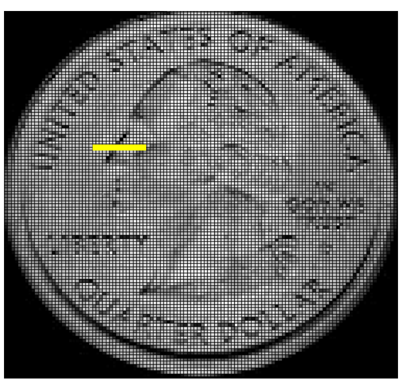
Step 2: Compute the Partial Derivative along X

Then use a 1D derivative estimation method to evaluate $\frac{\partial I}{\partial x}(x, y)$

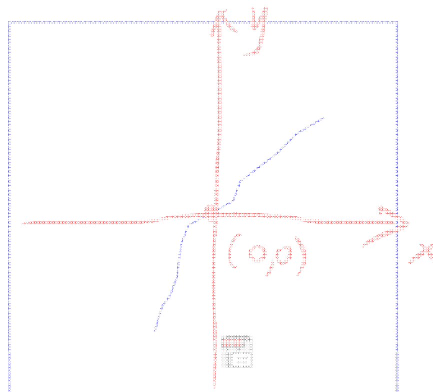
Local Analysis of Image Patches: Outline

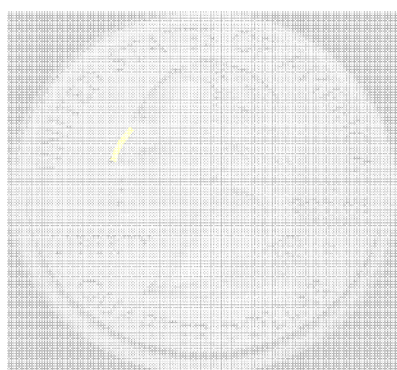
As graph in 2D



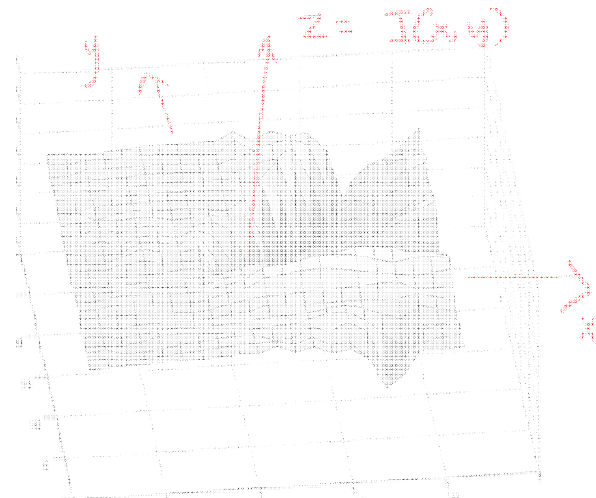


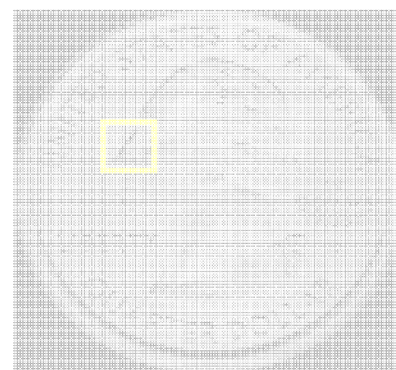
As curve in 2D





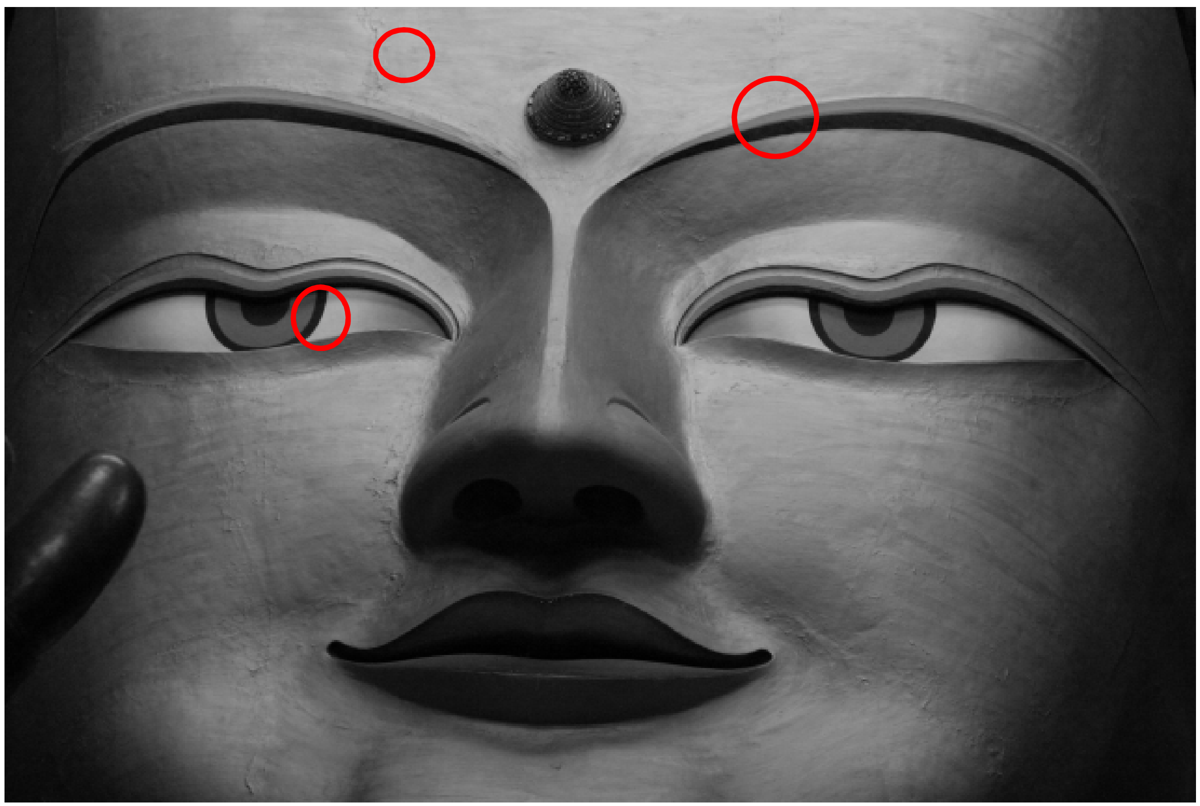
As surface in 3D





Step 2: Compute the Partial Derivative along X

How does $\left|\frac{\partial I}{\partial x}(x, y)\right|$ look for the image below?



Step 2: Compute the Partial Deriv along X

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

Step 3: Compute the Partial Derivative along Y

Repeat for $\frac{\partial I}{\partial y}(x,y)$

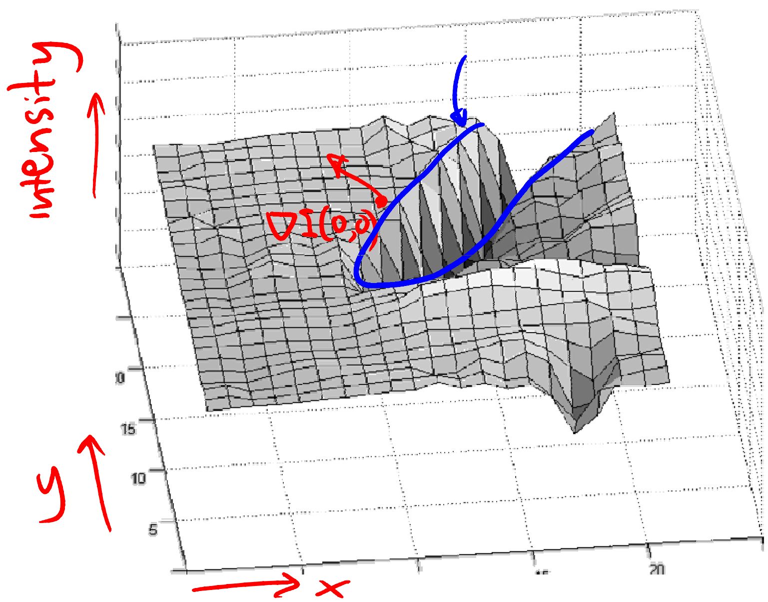
Step 2: Compute the Partial Derivative along X

How does $\left|\frac{\partial I}{\partial y}(x,y)\right|$ look for the image below?

Step 3: Compute the Partial Deriv along Y

$$\frac{\partial I}{\partial y}(x,y)$$

The Gradient Magnitude



Or the length of $\nabla I(x,y)$:

$$|\nabla I(x,y)| = \sqrt{\left(\frac{\partial I}{\partial x}(x,y)\right)^2 + \left(\frac{\partial I}{\partial y}(x,y)\right)^2}$$

Tells us how quickly intensity is changing in the neighborhood of pixel (x,y) in the direction of the gradient.

Step 4: Compute Magnitude at Each Pixel

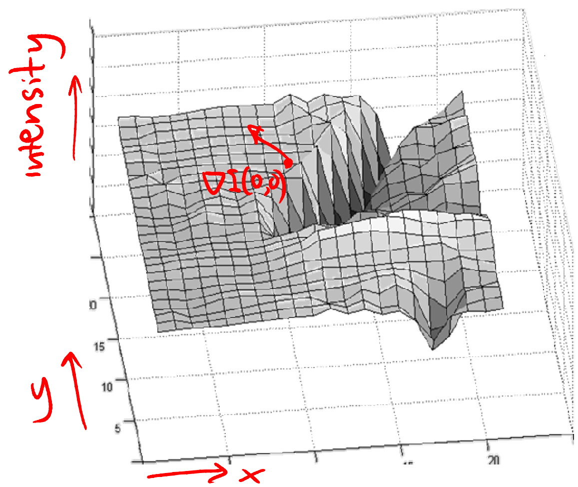
$$|\nabla I(x,y)| = \sqrt{\left(\frac{\partial I}{\partial x}(x,y)\right)^2 + \left(\frac{\partial I}{\partial y}(x,y)\right)^2}$$

The Gradient Orientation

The gradient orientation:

$$\theta = \tan^{-1}\left(\frac{\partial I}{\partial y}(x,y) \Big/ \frac{\partial I}{\partial x}(x,y) \right)$$

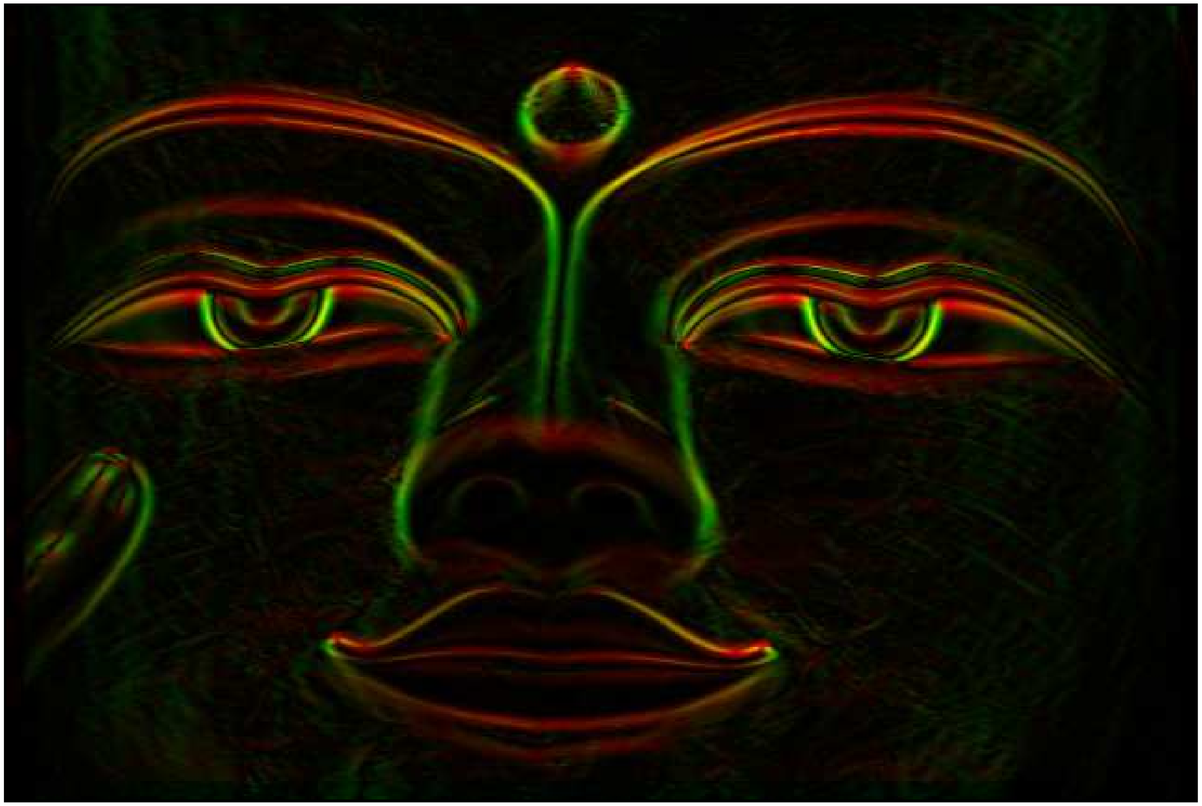
Tells us the direction of greatest intensity change in the neighborhood of pixel (x,y)



Step 5: Visualizing Magnitude & Orientation

One way of visualizing magnitude and orientation simultaneously:

$$red(x,y) = |\nabla I(x,y)| \cdot \sin\theta \qquad green(x,y) = |\nabla I(x,y)| \cdot \cos\theta \qquad blue(x,y) = 0$$



Looks like gradients are useful to find corners and edges, right?

Right

Topic 4.3:

Local analysis of 2D image patches

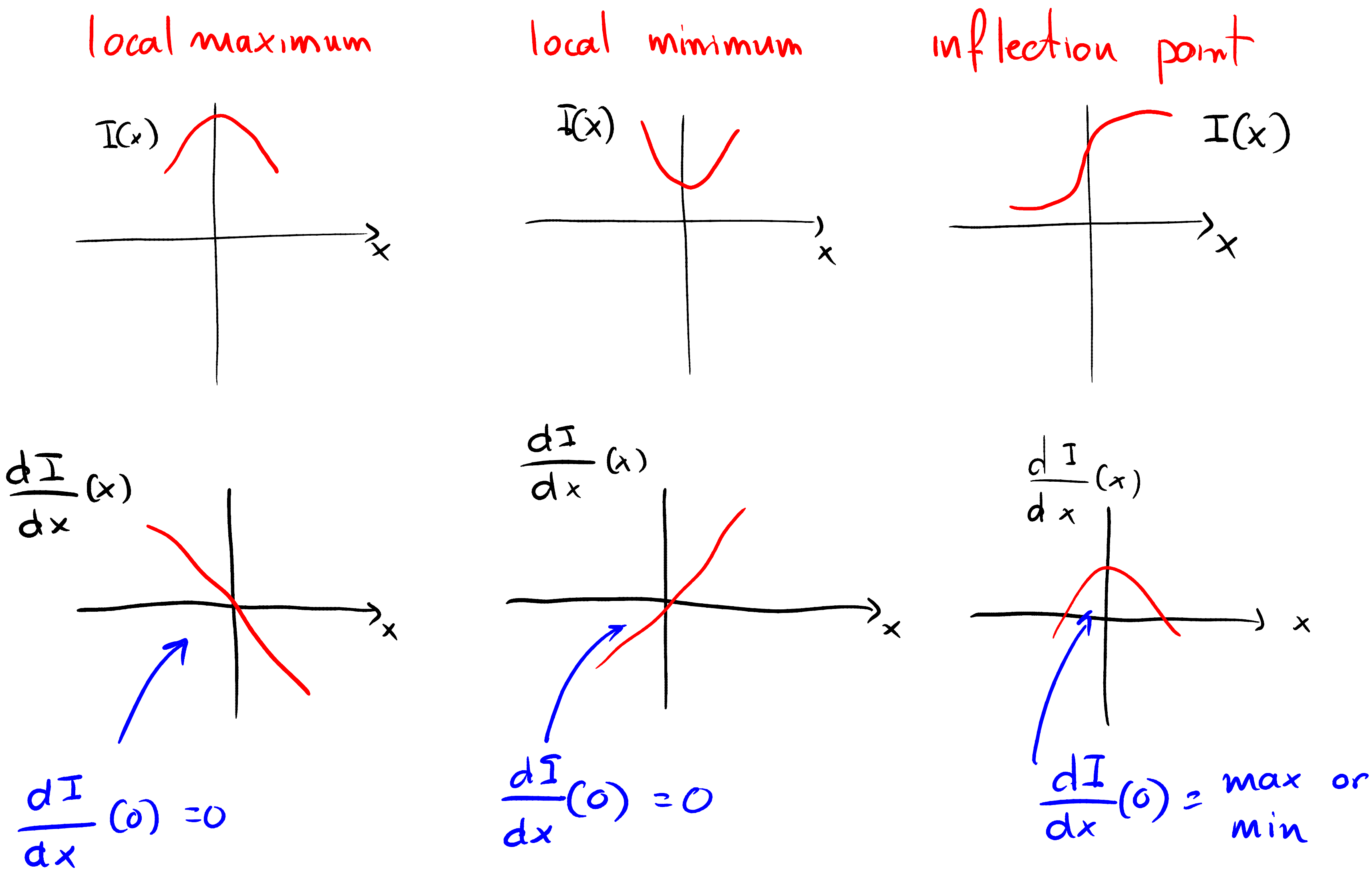
- Images as surfaces in 3D
- Directional derivatives
- Image Gradient
- Edge detection & localization
  - Gradient extrema
  - Laplacian zero-crossings
- Painterly rendering
- Local geometry at image extrema
- The Image Hessian
- Eigenvectors & eigenvalues
- Corner & feature detection
  - Lowe feature detector
  - Harris/Forstner detector

Analysing Special 2D Image Patches

How do we mathematically characterize local image patches as corners or edges?

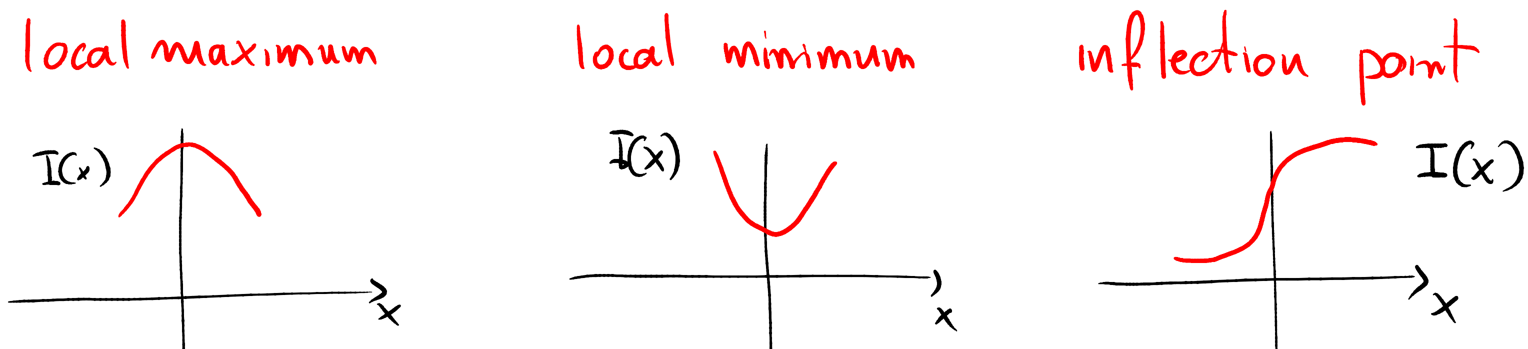
Special Patches in 1D

3 special 1D patches

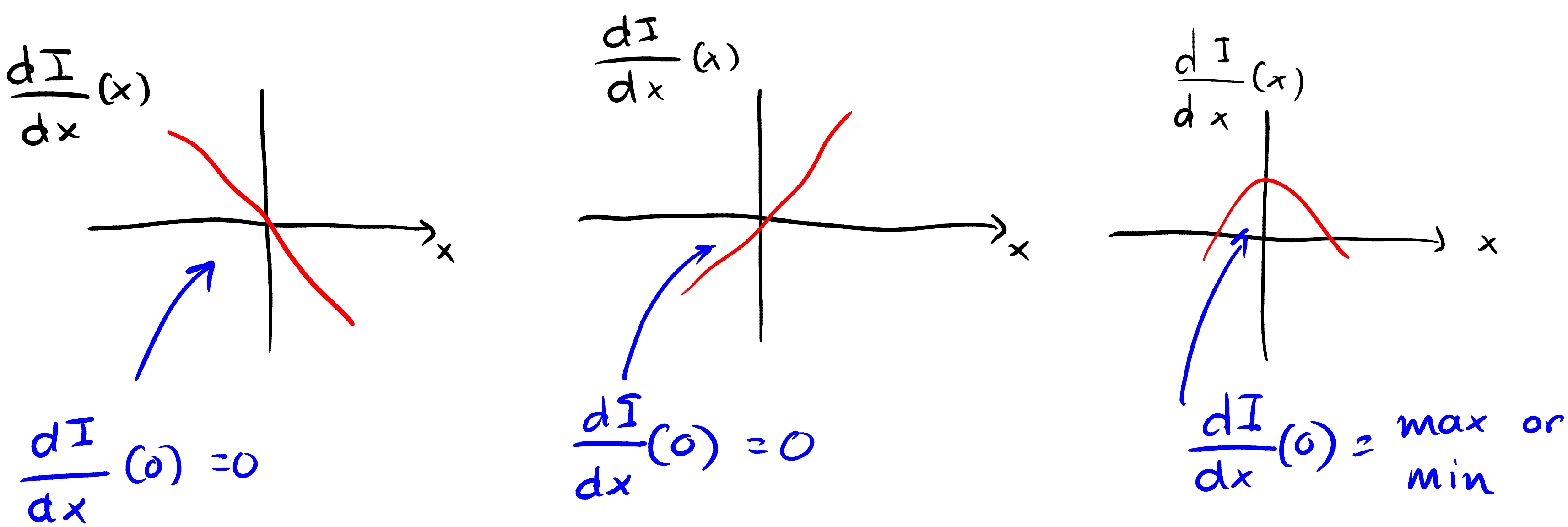


Special Patches in 1D

3 special 1D patches

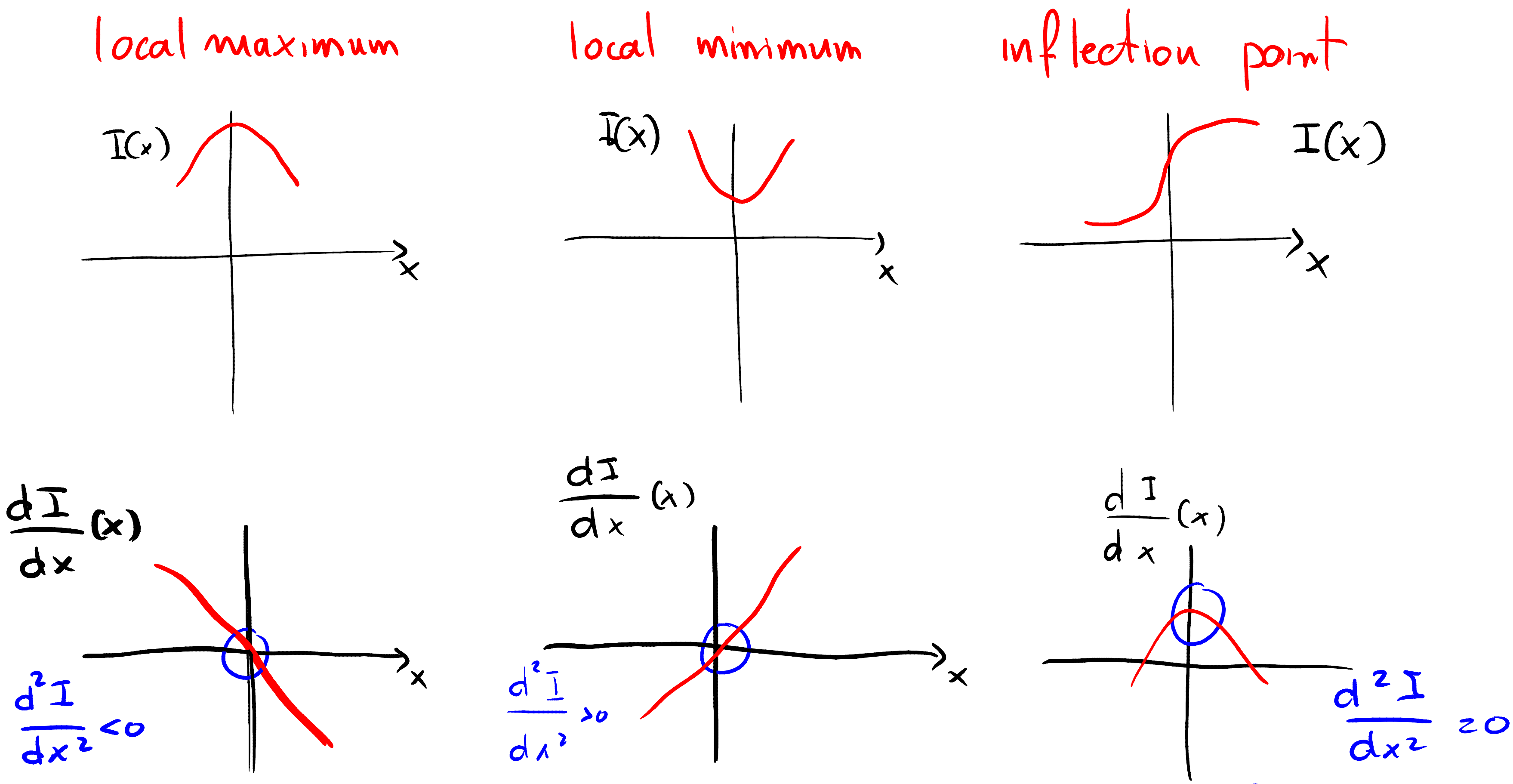


Can we tell between these three?



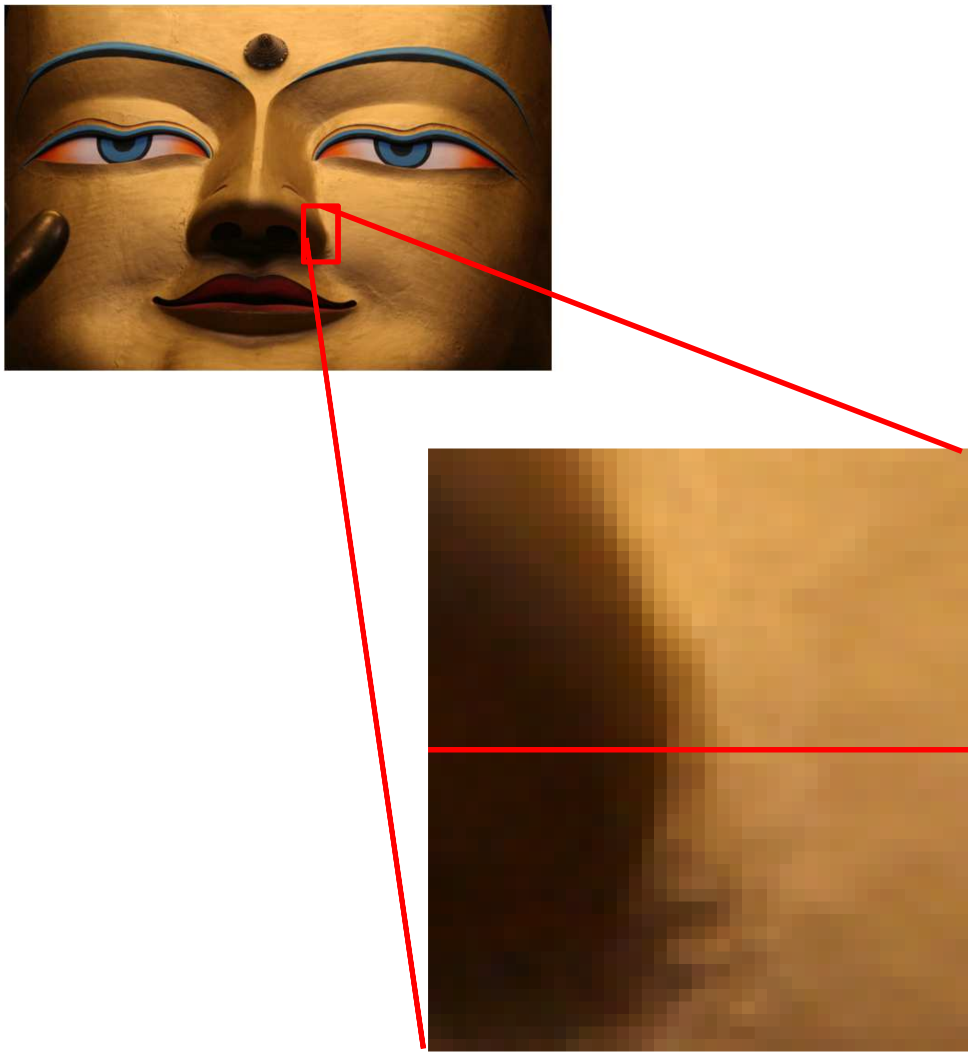
Special Patches in 1D

3 special 1D patches



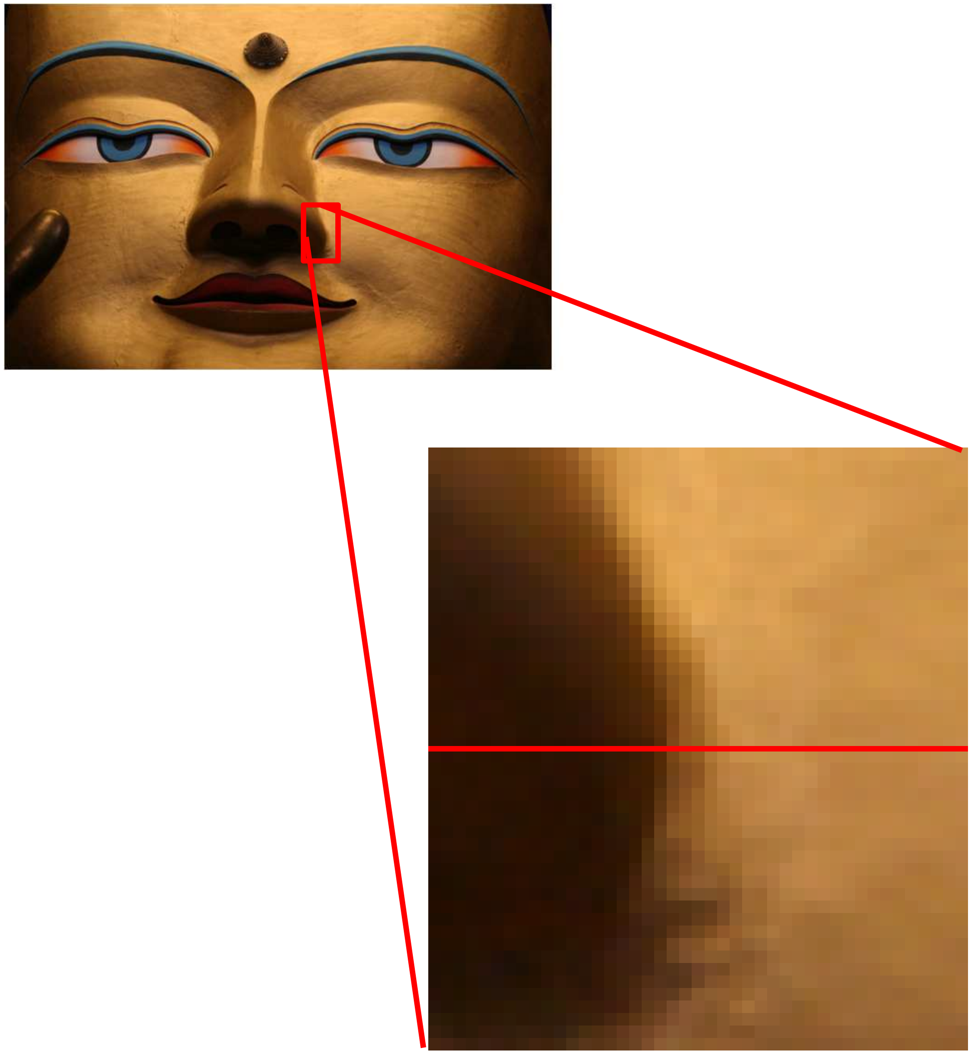
$\Rightarrow$ Types are distinguished by sign of $\frac{d^2I}{dx^2}(x)$

Special Patches in 1D

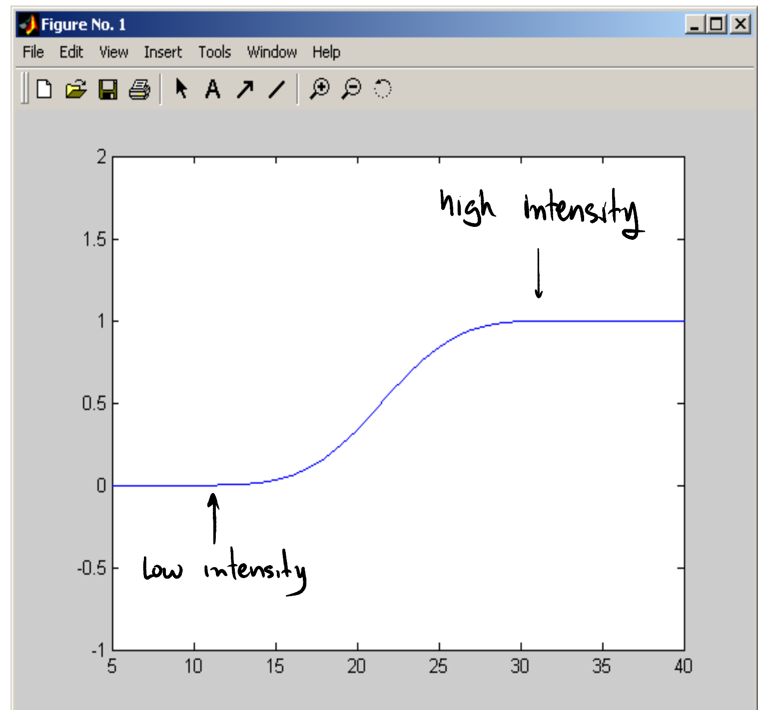


How does an edge look in 1D?

Special Patches in 1D

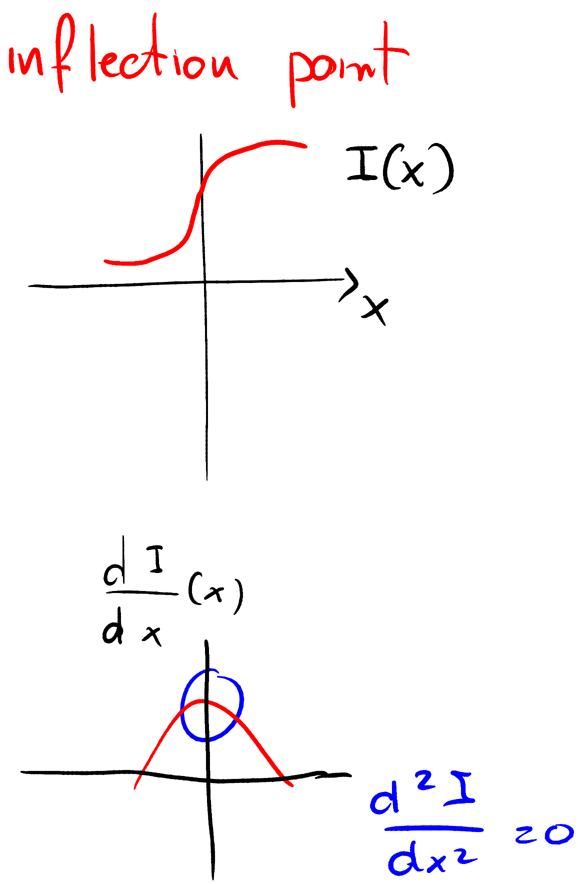


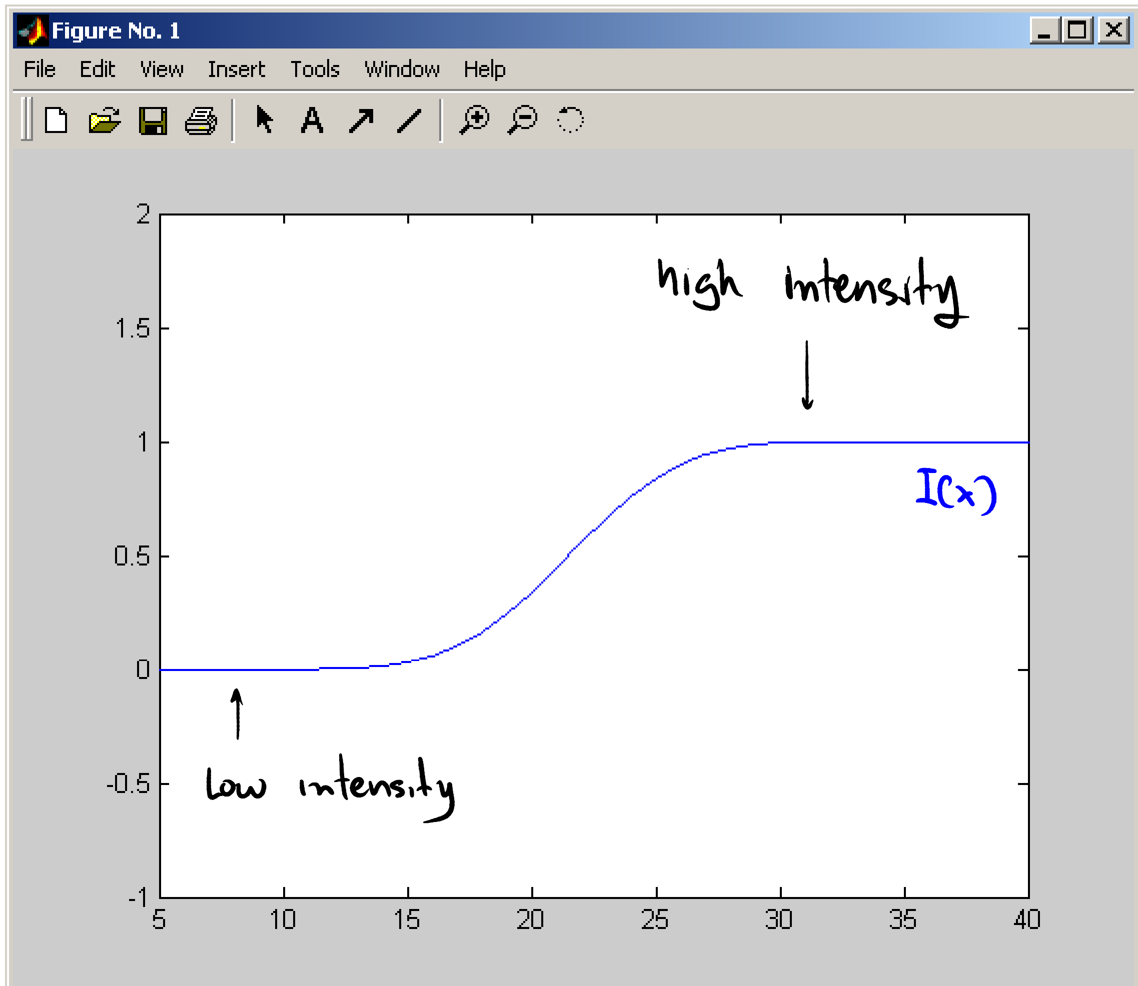
How does an edge look in 1D?



Detecting & Localizing 1D Edge Patches

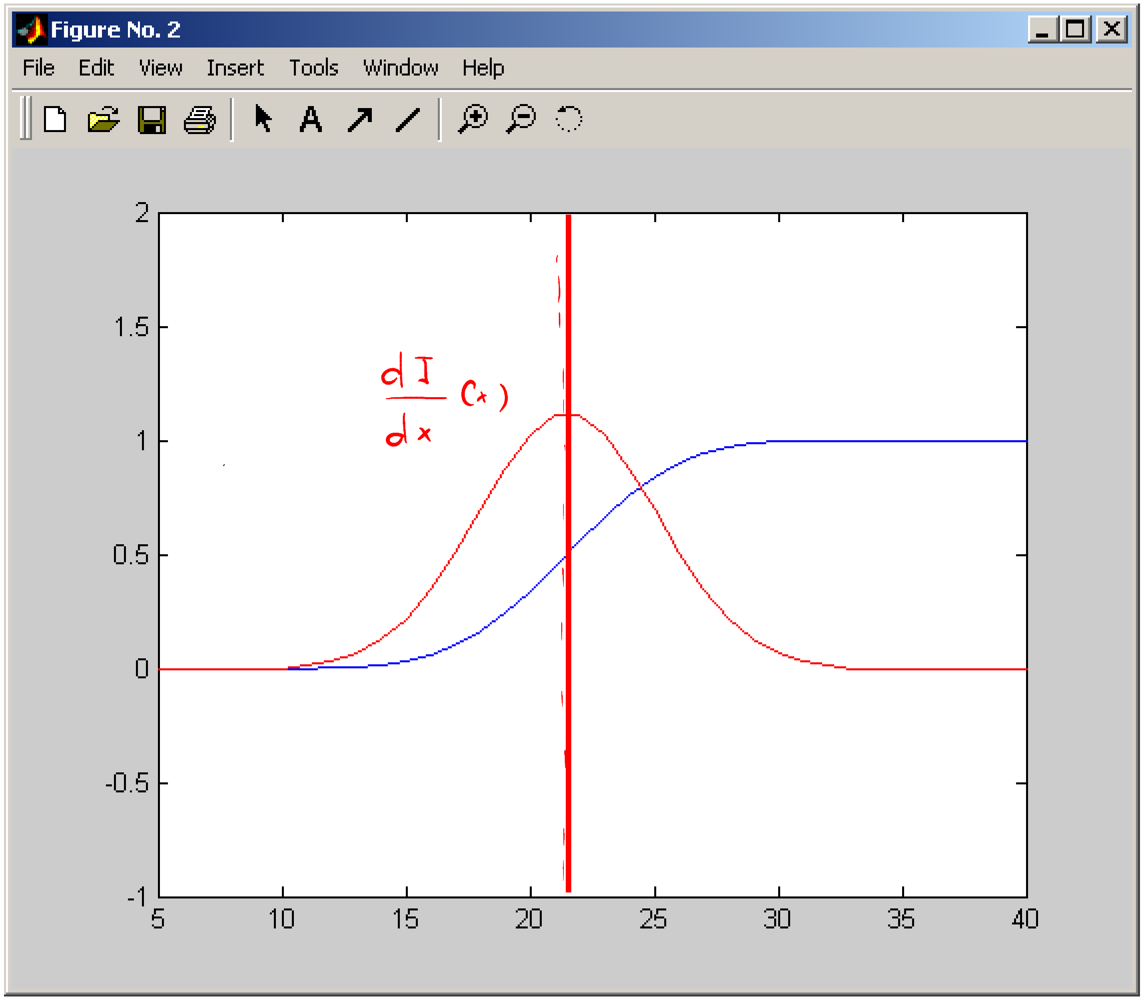
The ideal edge can be modeled as a smooth step function (which looks like an inflection point!)





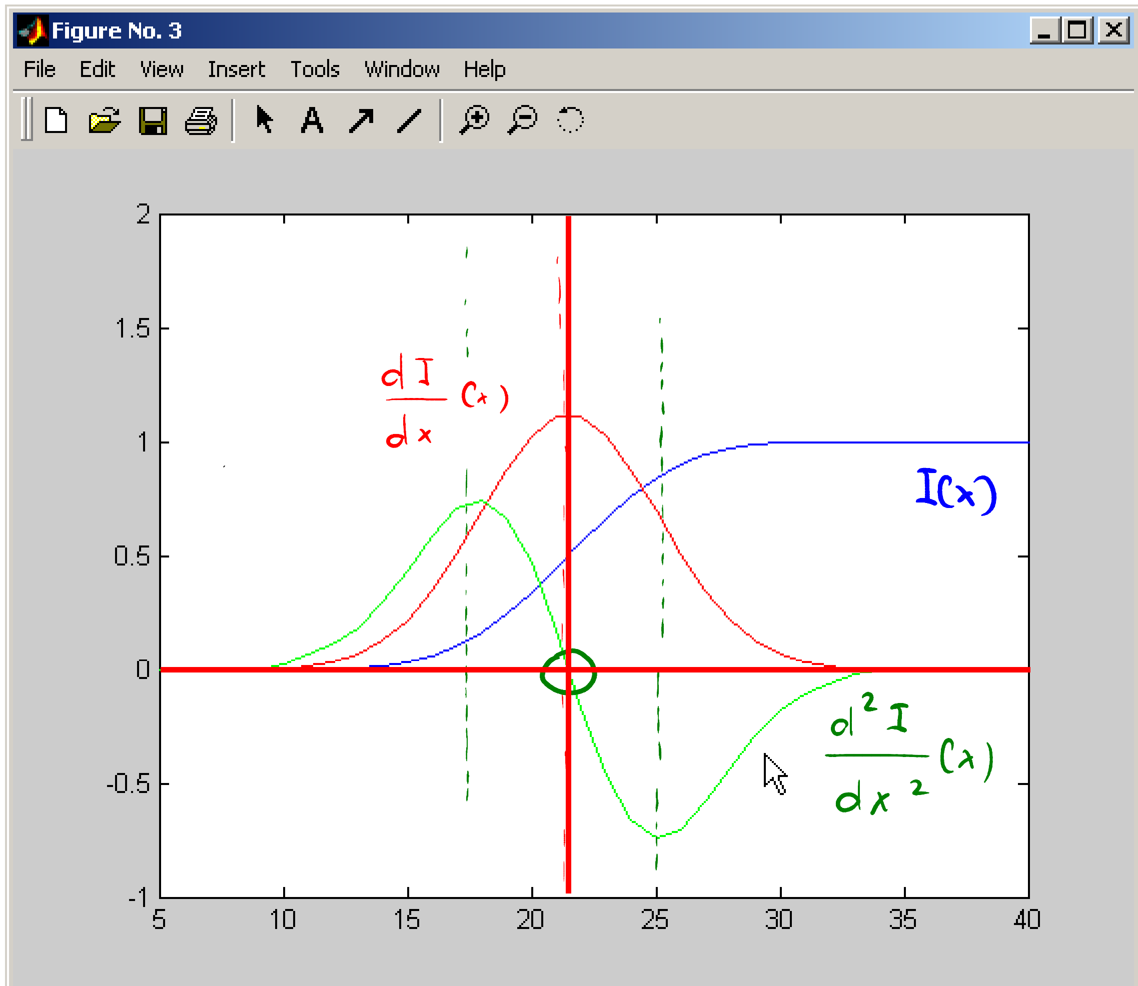
Detecting & Localizing 1D Edge Patches

The location of an edge is the same as the location of the max (or min) of $\frac{dI}{dx}$



Detecting & Localizing 1D Edge Patches

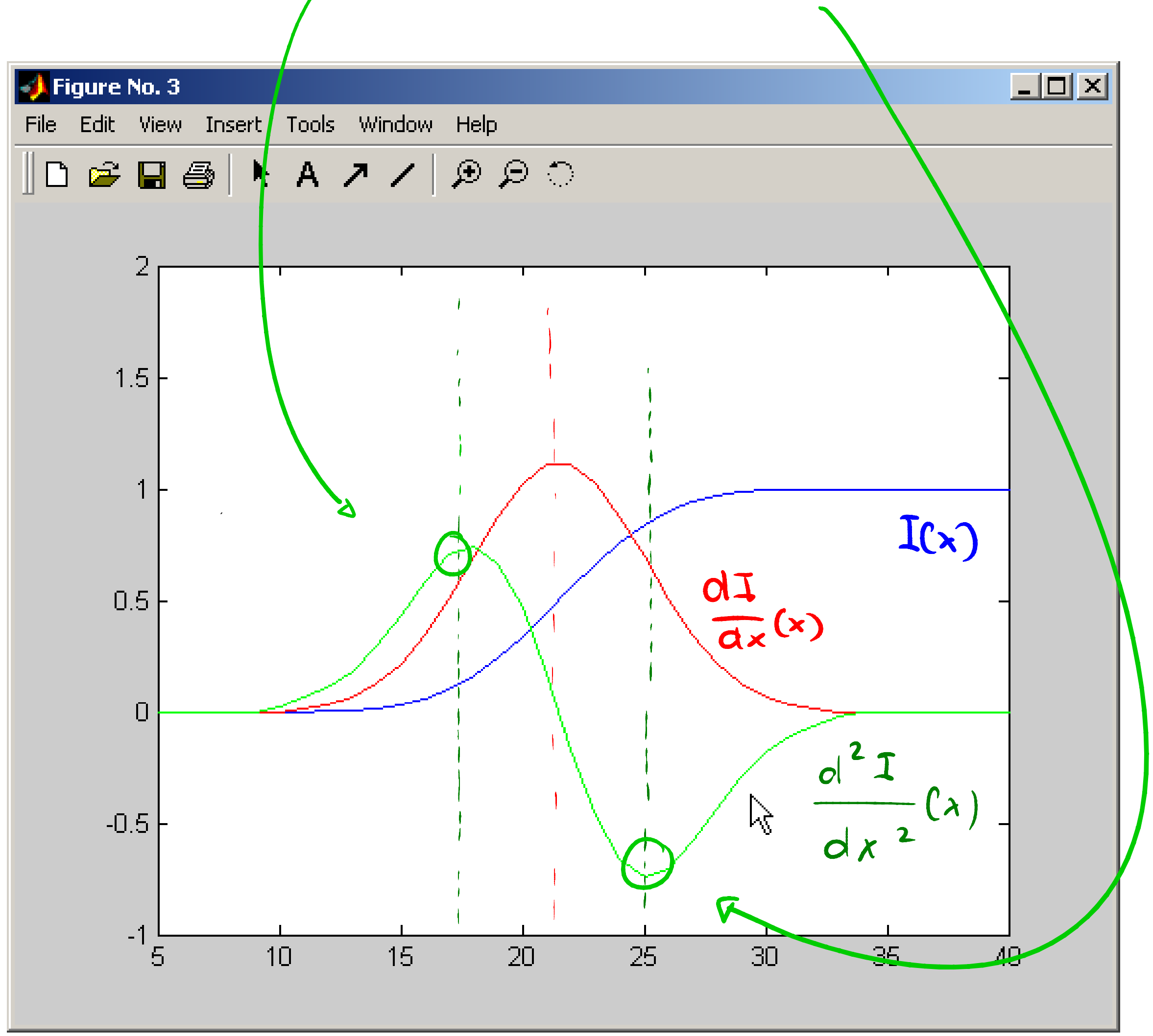
Or equivalently, the location of the zero-crossing of $\frac{d^2}{dx^2} I(x)$



Detecting & Localizing 1D Edge Patches

A third option is to find maximum and minimum in $\frac{d^2}{dx^2} I(x)$

Pairs of extrema determine the “beginning” and the “end” of an edge.



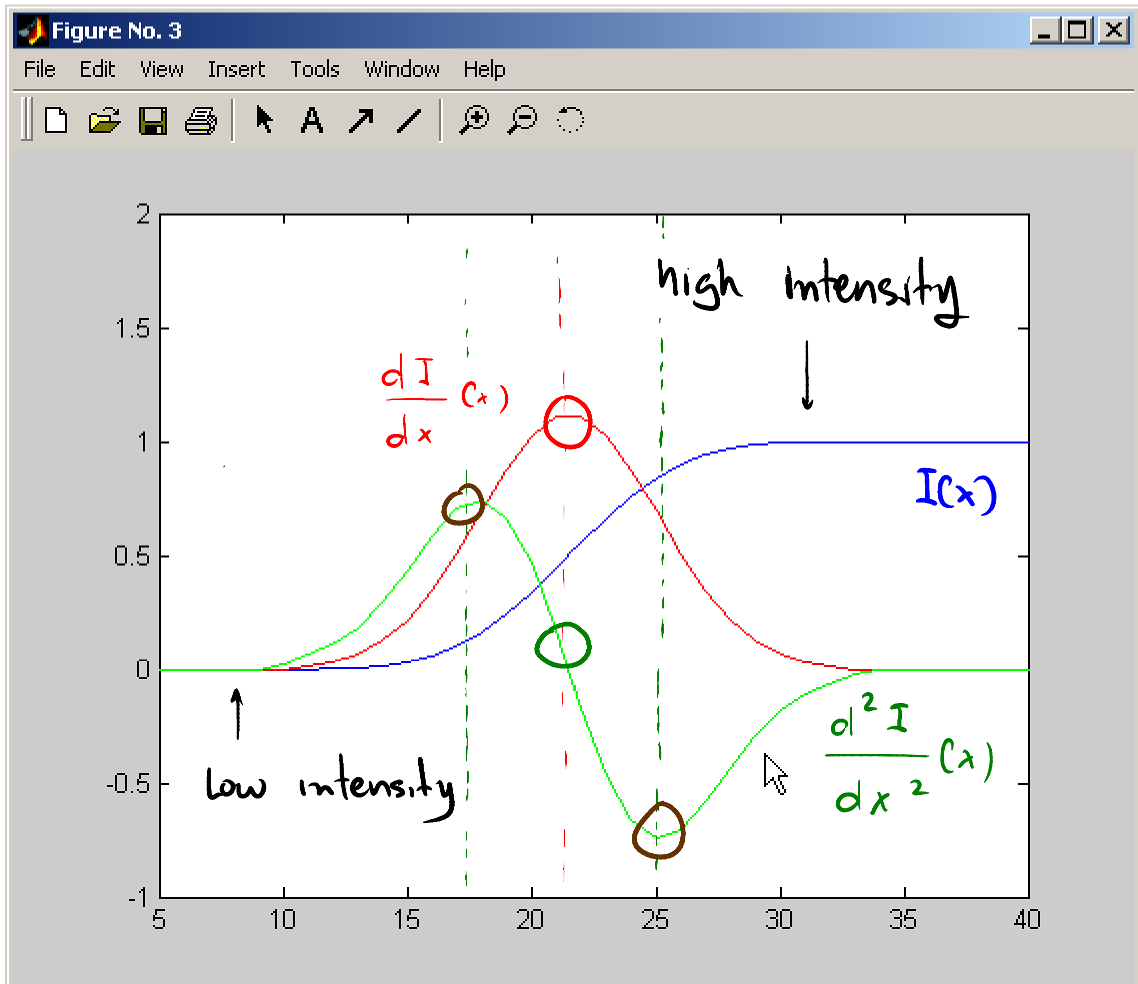
Detecting & Localizing 1D Edge Patches

In summary, to identify an edge (or an inflection point) one can:

Find maxima or minima of $\frac{dI}{dx}$

Find zero crossings of $\frac{d^2I}{dx^2}$

Find maxima and minima of $\frac{d^2I}{dx^2}$



Alright, lets find some edges!

Algorithm #1

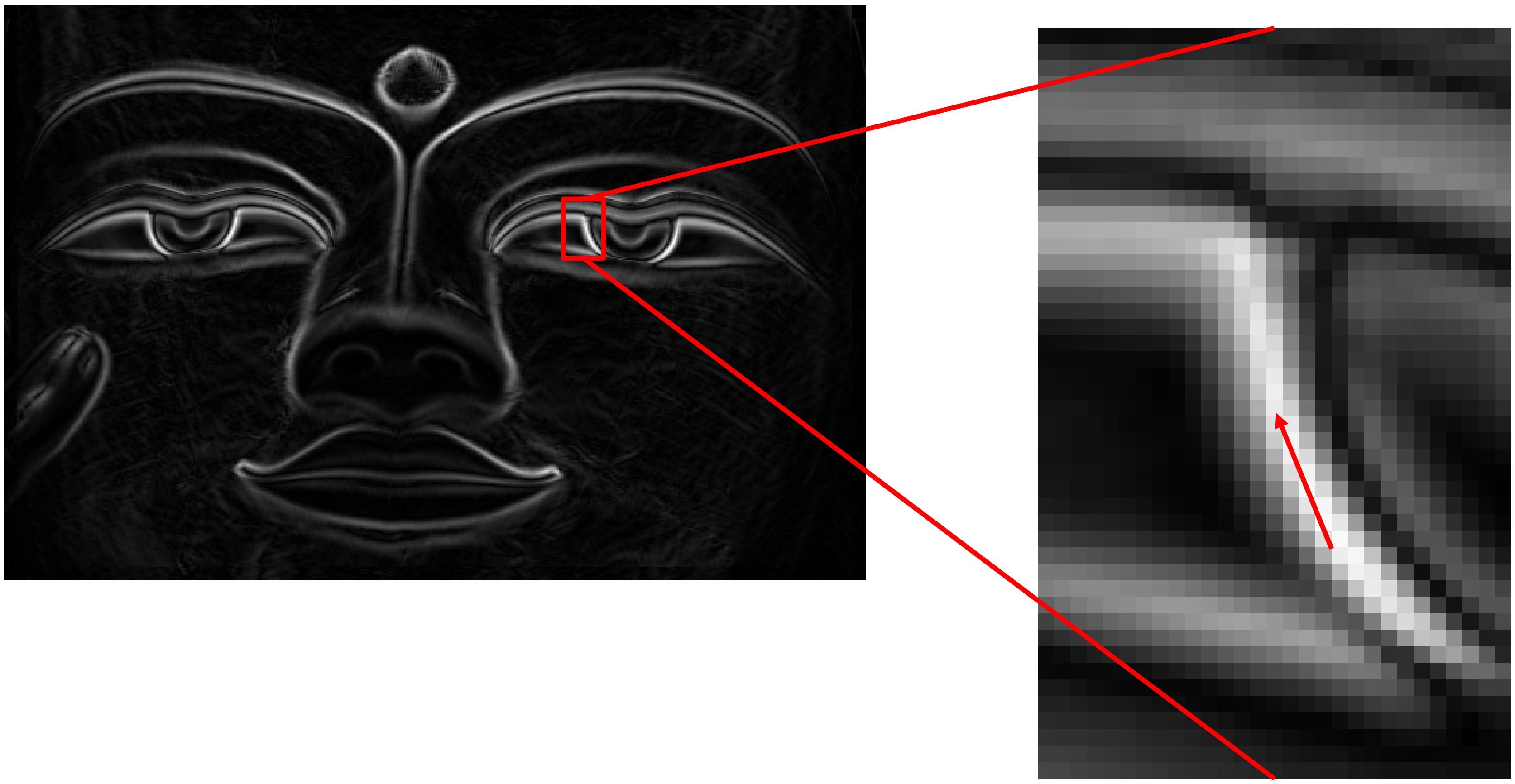
Pixels with maximum Gradient (magnitude)

Topic 4.3:

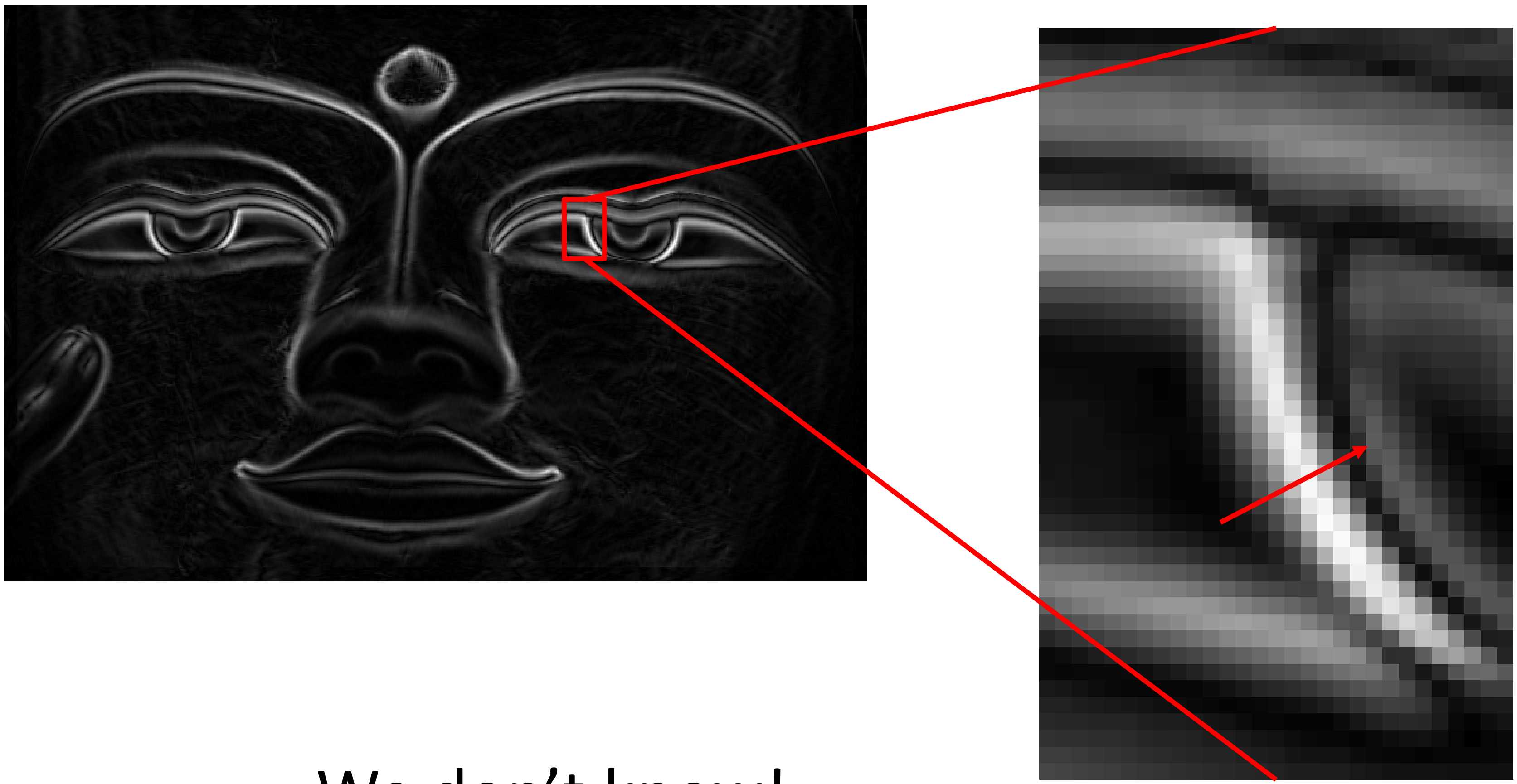
Local analysis of 2D image patches

- Images as surfaces in 3D
- Directional derivatives
- Image Gradient
- Edge detection & localization
  - Gradient extrema
  - Laplacian zero-crossings
- Painterly rendering
- Local geometry at image extrema
- The Image Hessian
- Eigenvectors & eigenvalues
- Corner & feature detection
  - Lowe feature detector
  - Harris/Forstner detector

Maxima? In which direction?



Maxima? In which direction?



We don’t know!

But not all is lost.

Let’s simply use large magnitude gradients

Step #1: Compute Gradient Magnitude

Using a gradient magnitude image $|\nabla I(x,y)|$

Step #2: Find Pixels with High Gradient Mag

Mark all the pixels with $|\nabla I| > 5$ as edges.

Step #2: Find Pixels with High Gradient Mag

Trivial, works, but:

Edges are not well-localized (i.e. they are thick)

We have to choose a threshold (how?)

Image where all pixels with $|\nabla I| > 5$ are shown in white

threshold

Step #2: Find Pixels with High Gradient Mag

Can we do better?

Step #2: Find Pixels with High Gradient Mag

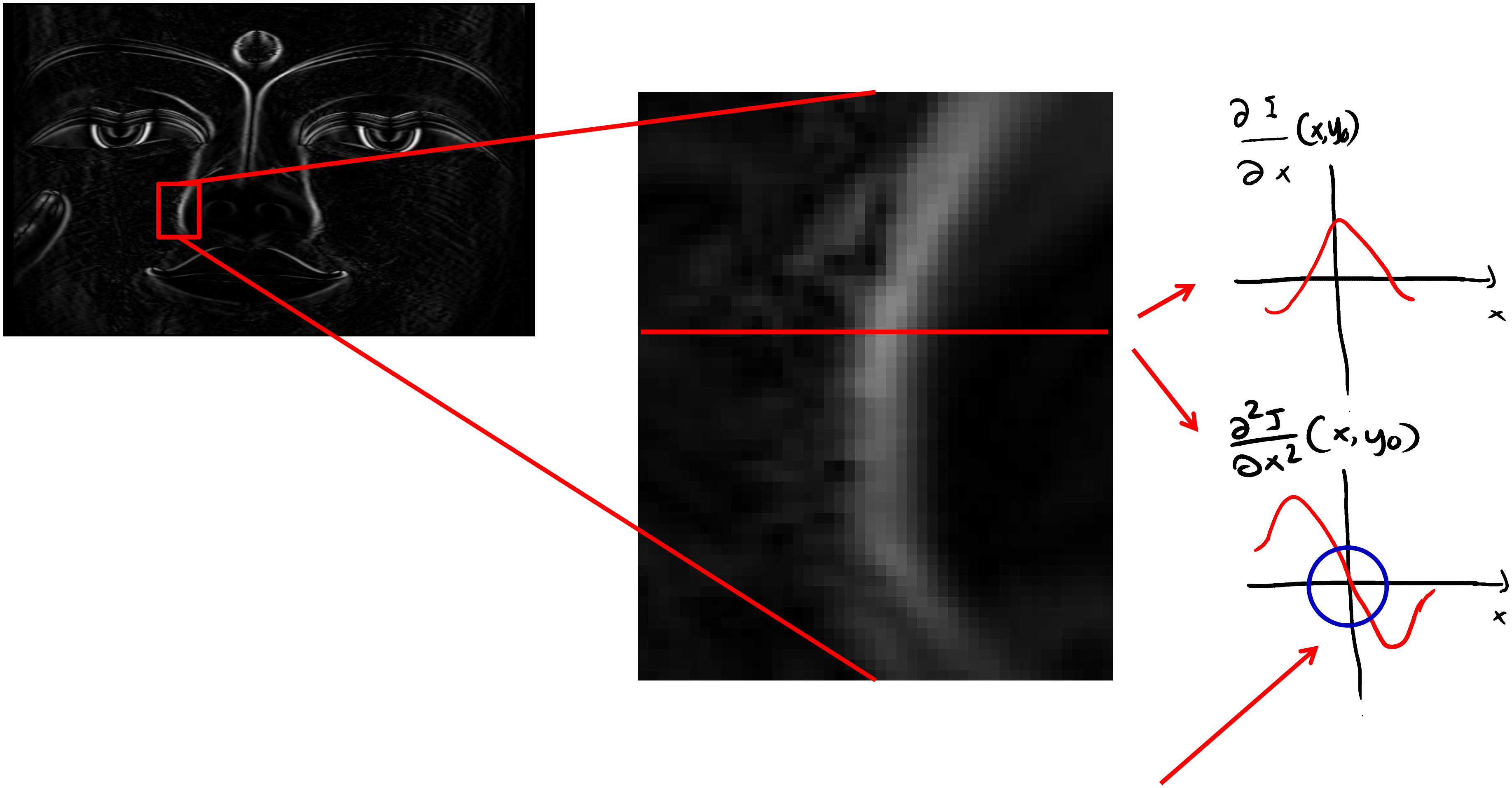
Can we do better? How about zero crossings from the second derivative?

Topic 4.3:

Local analysis of 2D image patches

- Images as surfaces in 3D
- Directional derivatives
- Image Gradient
- Edge detection & localization
    - Gradient extrema
    - Laplacian zero-crossings
- Painterly rendering
- Local geometry at image extrema
- The Image Hessian
- Eigenvectors & eigenvalues
- Corner & feature detection
    - Lowe feature detector
    - Harris/Forstner detector

Algorithm #2: Find Extrema of 1st Derivative



Here! Look! Extrema!

Step 1: Compute 2nd order Image Derivative

Compute the 2nd order derivative $\frac{\partial^2 I}{\partial x^2}$

$\left|\frac{\partial^2 I}{\partial x^2}\right| \rightarrow$

Step 2: Compute 2nd order Image Derivative

Compute the 2nd order derivative $\frac{\partial^2 I}{\partial y^2}$

$\left|\frac{\partial^2 I}{\partial y^2}\right| \rightarrow$

Step 3: Compute The Image Laplacian

Form the Laplacian $\nabla^2 I = \frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 I}{\partial y^2}$

Laplacian: scalar, analog to second derivative

$|\nabla^2 I|$ →

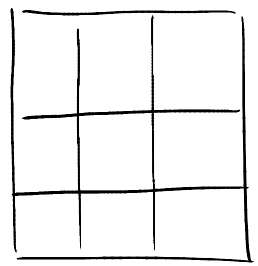
Step 4: Find the Laplacian Zero Crossings

Finding zero crossings is much easier than finding extrema because…

Step 4: Find the Laplacian Zero Crossings

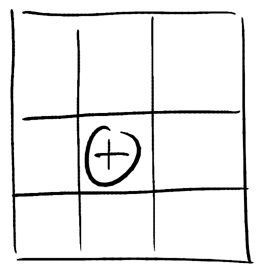
Finding zero crossings is much easier than finding extrema because it's a local property!

Consider a 3x3 patch:



Step 4: Find the Laplacian Zero Crossings

Finding zero crossings is much easier than finding extrema because it's a local property!

Consider a 3x3 patch: 

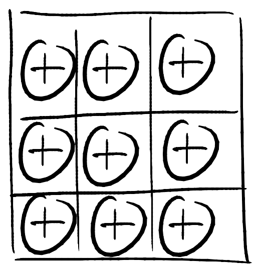
assume

how can we tell if there was a zero crossing in the patch?

Step 4: Find the Laplacian Zero Crossings

Finding zero crossings is much easier than finding extrema because it's a local property!

Consider a 3x3 patch:

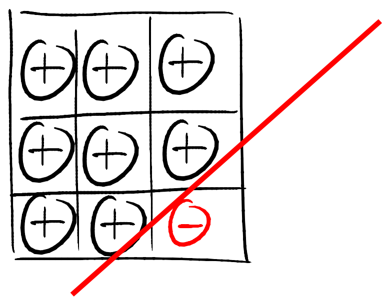


no zero crossing

Step 4: Find the Laplacian Zero Crossings

Finding zero crossings is much easier than finding extrema because it's a local property!

Consider a 3x3 patch:

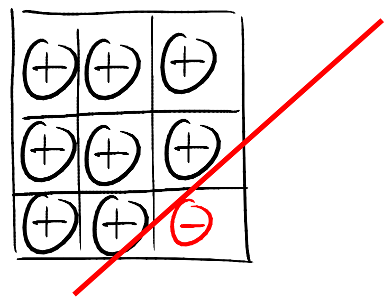


zero crossing!

Step 4: Find the Laplacian Zero Crossings

Finding zero crossings is much easier than finding extrema because it's a local property!

Consider a 3x3 patch:

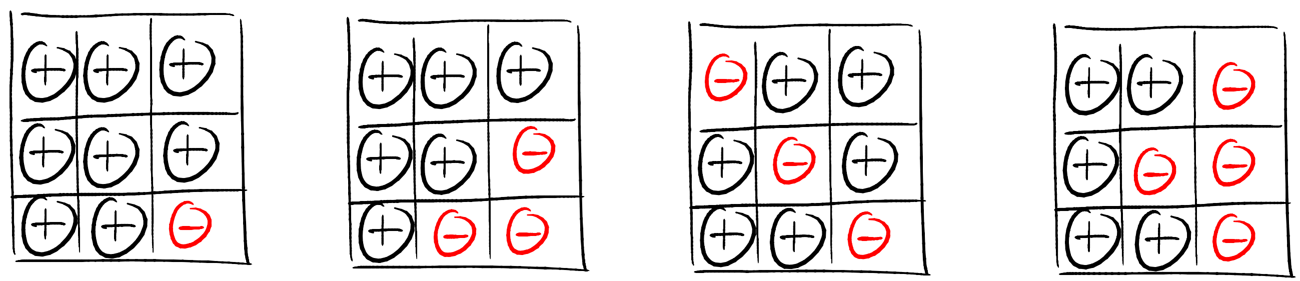


zero crossing!

If at least one pixel has a Laplacian of different sign than the Laplacian of the center pixel, then a zero crossing occurred!

Step 4: Find the Laplacian Zero Crossings

Finding zero crossings is much easier than finding extrema because it's a local property!

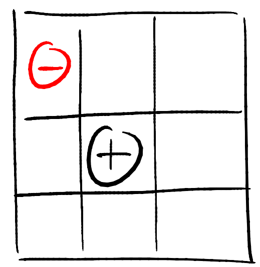


Other examples.

If at least one pixel has a Laplacian of different sign than the Laplacian of the center pixel, then a zero crossing occurred!

Step 4: Find the Laplacian Zero Crossings

Not all zero crossings are created equal!



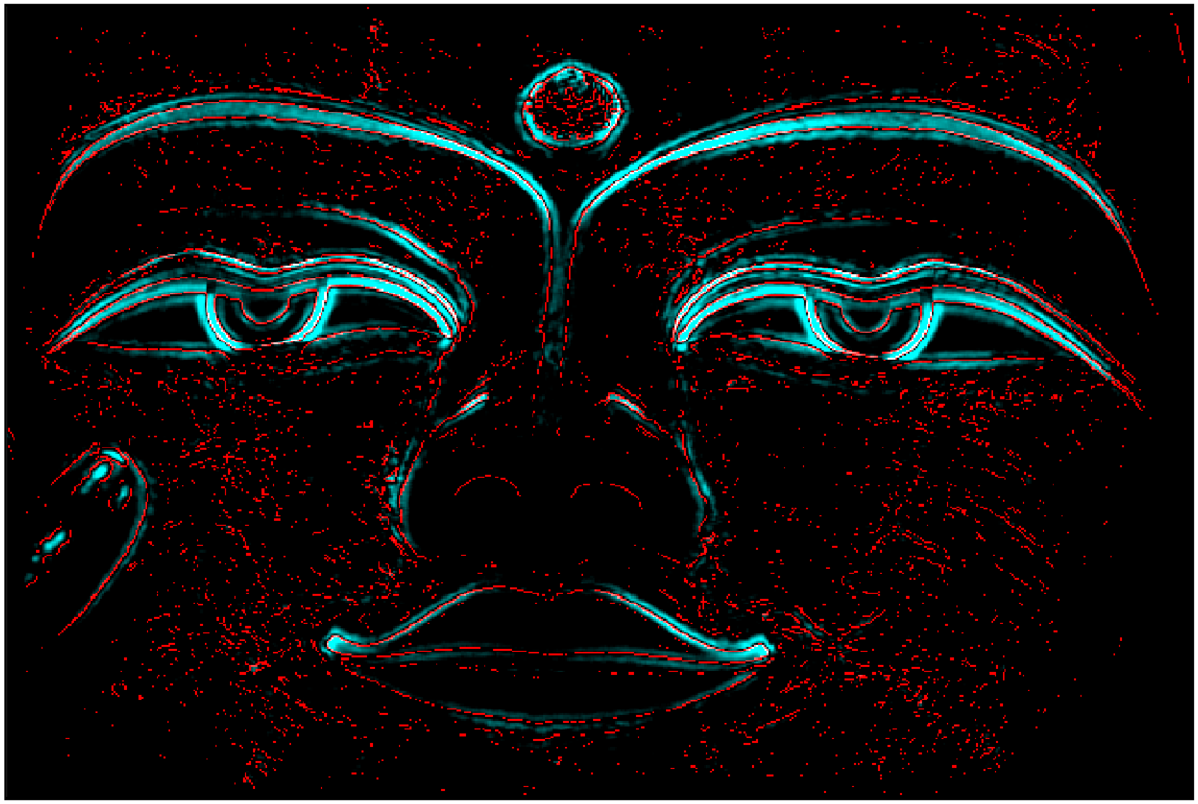
The strength of the zero crossing can be defined as the difference between the ⊕ and the ⊖ values.

Step 4: Find the Laplacian Zero Crossings

Zero-crossings whose strength is greater than a threshold.

Step 4: Find the Laplacian Zero Crossings

Laplacian with zero-crossings overlaid



Topic 4.3:

Local analysis of 2D image patches

- Images as surfaces in 3D
- Directional derivatives
- Image Gradient
- Edge detection & localization
  - Gradient extrema
  - Laplacian zero-crossings
- Painterly rendering
- Local geometry at image extrema
- The Image Hessian
- Eigenvectors & eigenvalues
- Corner & feature detection
  - Lowe feature detector
  - Harris/Forstner detector

Giving Photos a "Painted" Look

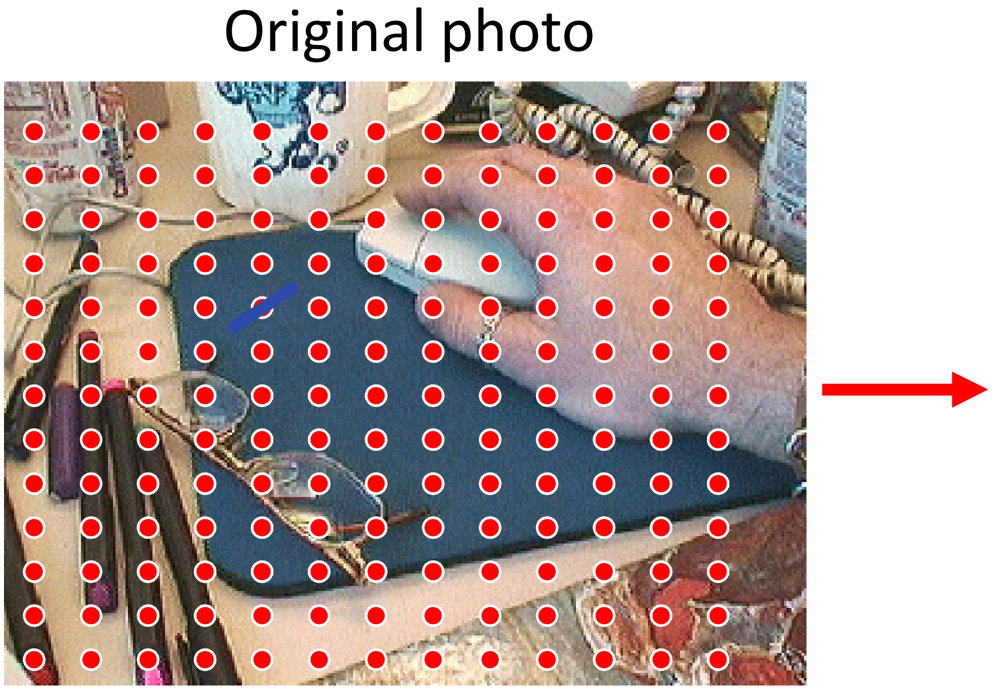
Case study: From P. Litwinowicz's SIGGRAPH'97 paper
"Processing Images and Videos for an
Impressionist Effect"

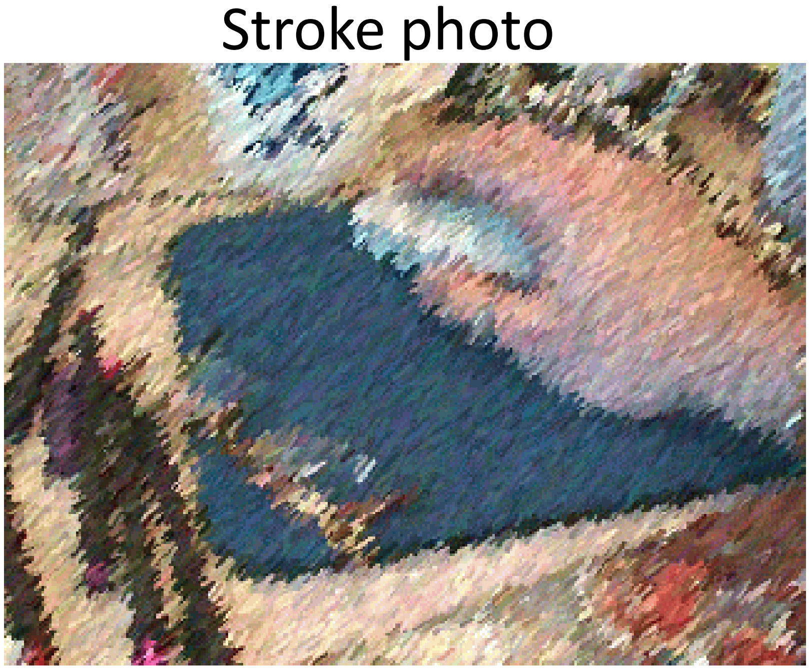
Giving Photos a “Painted” Look

How would you do it?

Original photo

Step 1: Stroke Scan-Conversion

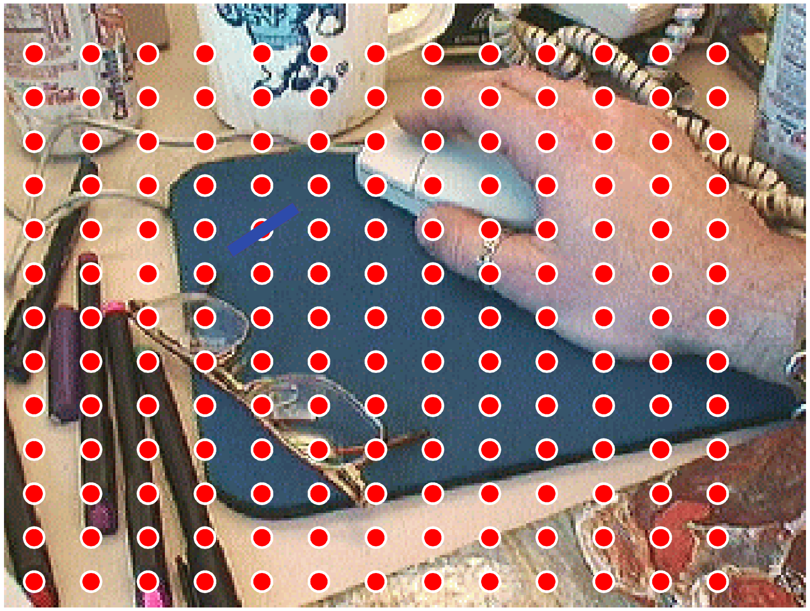




- Stroke: A short line drawn over the photo
- Strokes are drawn every k pixels
- Strokes drawn at a fixed angle (45 deg.)
- Strokes take color of their origin pixel
- Stroke length is chosen at random

Step 1: Stroke Scan-Conversion

Original photo



Stroke photo

Cool, but jagged edges not cool

Step 2: Edge Detection

Original photo

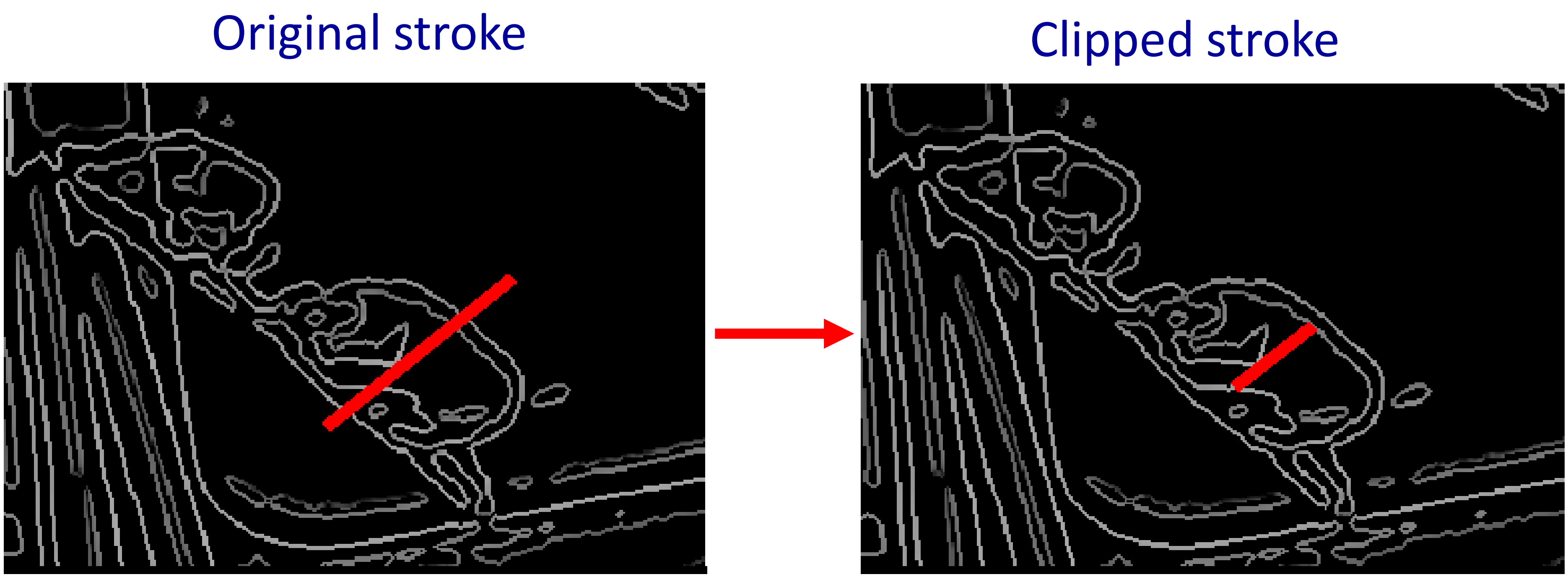
Edge image

<u>Edge detection step</u>: For every pixel in original photo

- Compute image gradient at the pixel
- Compute gradient magnitude (in the range 0-255)
- If magnitude > threshold, label pixel as an “edge pixel”
- Compute gradient orientation
- Compute the vector v perpendicular to pixel’s gradient

Step 3: Stroke Clipping

Motivation: To avoid “spill-over” artifacts, strokes are clipped at edges detected in the image (i.e., a stroke should not cross an edge pixel)



Step 3: Stroke Clipping Results

Original Stroke Photo

Clipped Stroke Photo

Cooler, but still not van Gogh!

Strokes are all oriented: boring

Step 4: Incorporating Edge Orientation

Toss the 45-degree angle strokes

Draw strokes in the direction normal to the gradient!

Clipped Stroke Photo

Oriented Stroke Photo

v.G. would be proud!