Topic 06:

Edge Detection Basics

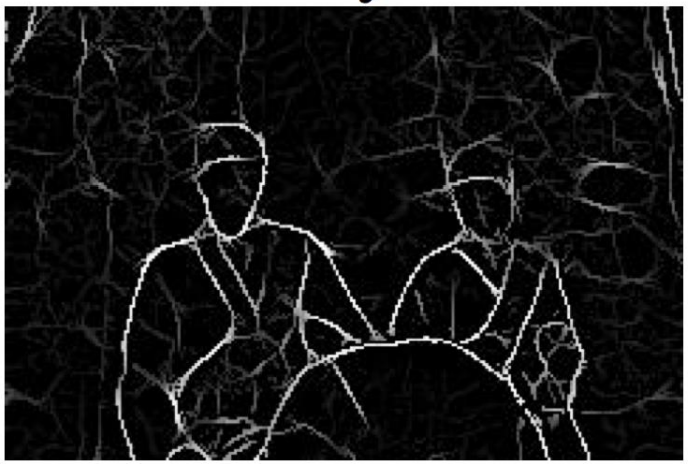
- Edge detection in 2D
- Sub-pixel edge localization
- The Canny edge detector
- Gradient-domain image processing

edge detection

Image

Canny

Boundary Prob.

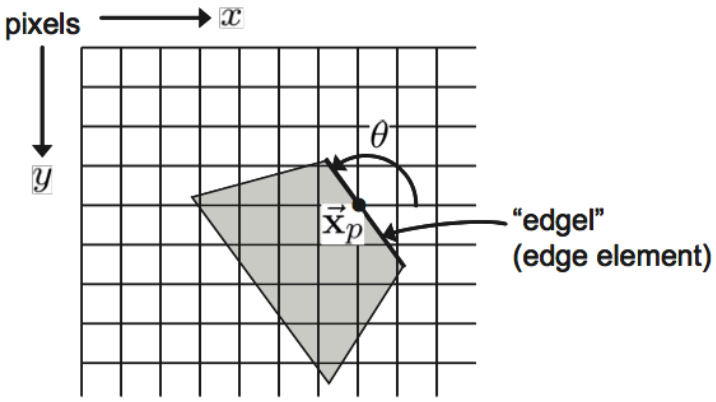


Human

edge detection

Motivation:

- Significant, often sharp, contrast variations in images caused by illumination, surface markings (albedo), and surface boundaries. These are useful for scene interpretation.
- **Edgels (edge elements):** significant local variations in image brightness, characterized by the position $\vec{\mathbf{x}}_p$ and the orientation θ of the brightness variation. (Usually $\theta \bmod \pi$ is sufficient.)



- **Edge:** a sequence of edgels forming a smooth curve

Two Problems:

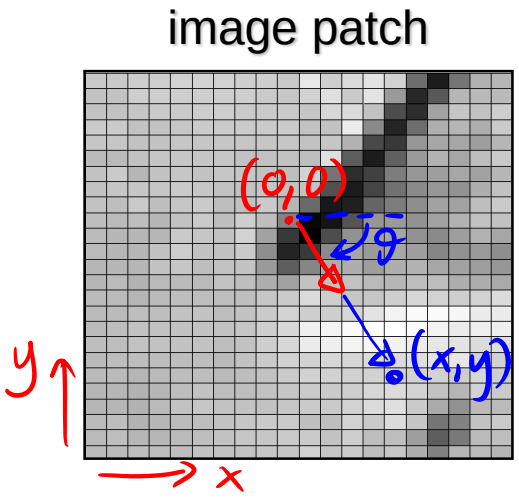
1. estimating edgels
2. grouping edgels into edges

Topic 06:

Edge Detection Basics

- Edge detection in 2D
- Sub-pixel edge localization
- The Canny edge detector
- Gradient-domain image processing

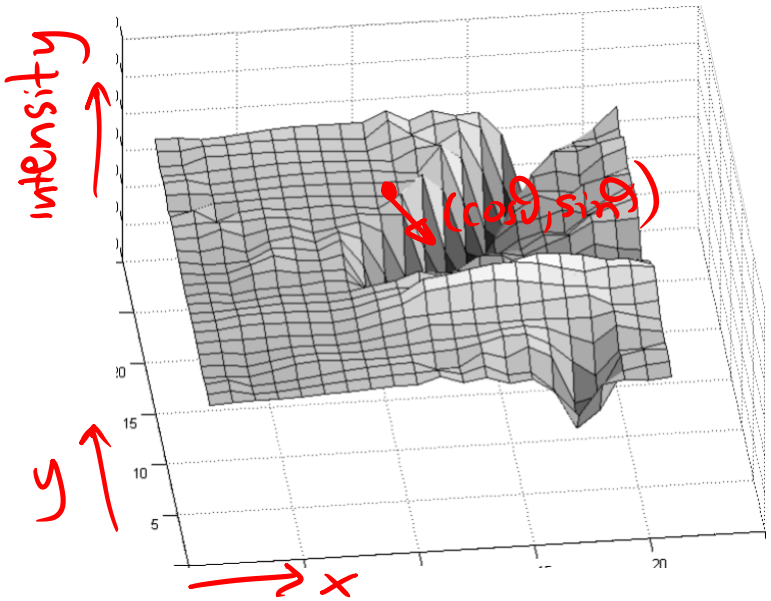
computing directional image derivatives



1st order Taylor series approximation

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

What is the derivative of the image intensity along a direction θ?



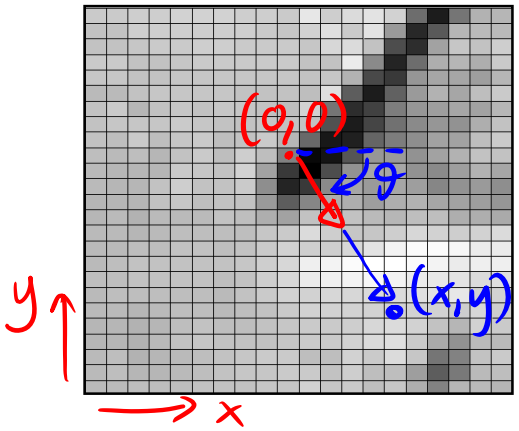
- Tells us how quickly intensity changes in a given direction
- Unit vector in direction θ:

$$v = (\cos\theta, \sin\theta)$$

- Intensity function along v:

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

computing directional image derivatives

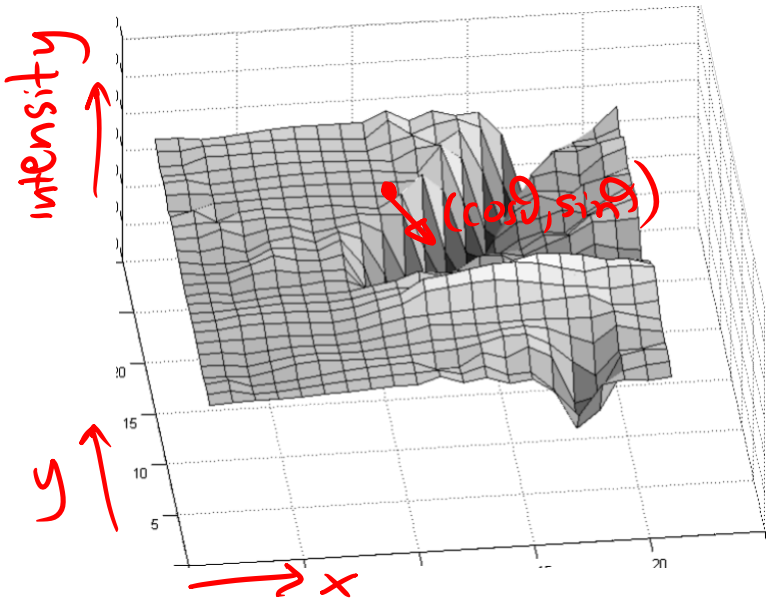


1st order Taylor series approximation

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

substitute $x \to t\cos\theta$ $y \to t\sin\theta$

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



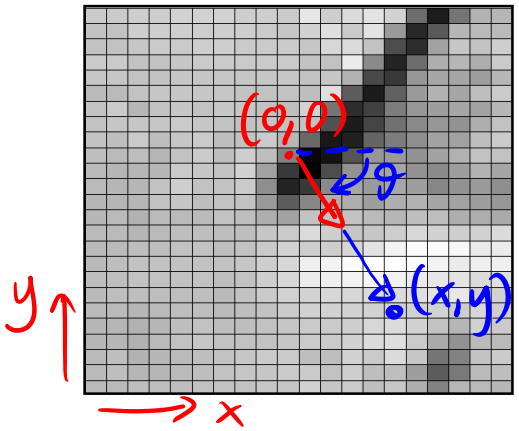
- Unit vector in direction θ:

$$v = (\cos\theta, \sin\theta)$$

- Intensity function along v:

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

computing directional image derivatives



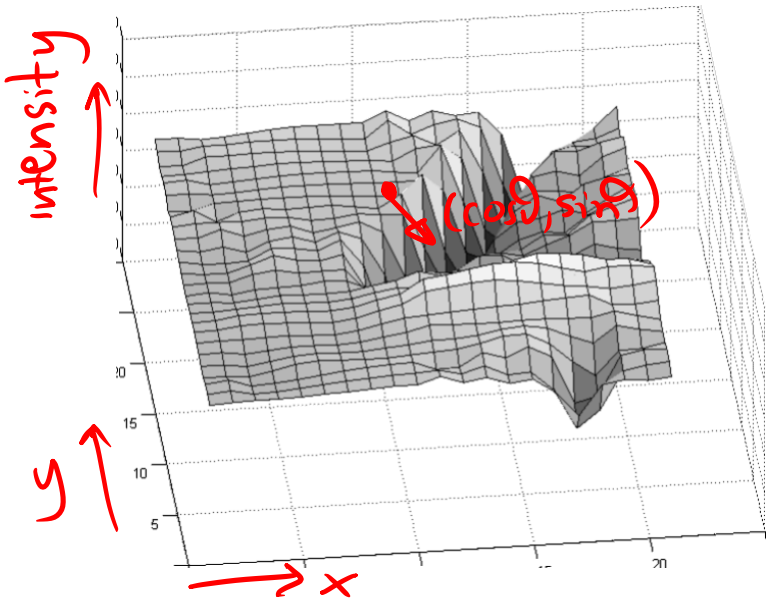
1st order Taylor series approximation

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

substitute $x \to t\cos\vartheta$ $\quad y \to t\sin\vartheta$

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

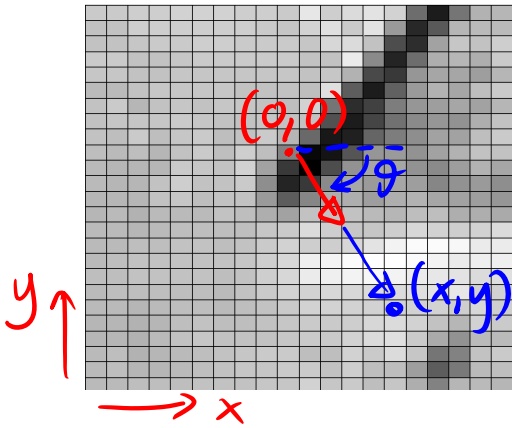
Directional derivative of I in direction $(\cos\vartheta, \sin\vartheta)$



in matrix notation:

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

the directional derivative



Directional derivative of I in direction $v = (\cos\theta, \sin\theta)$

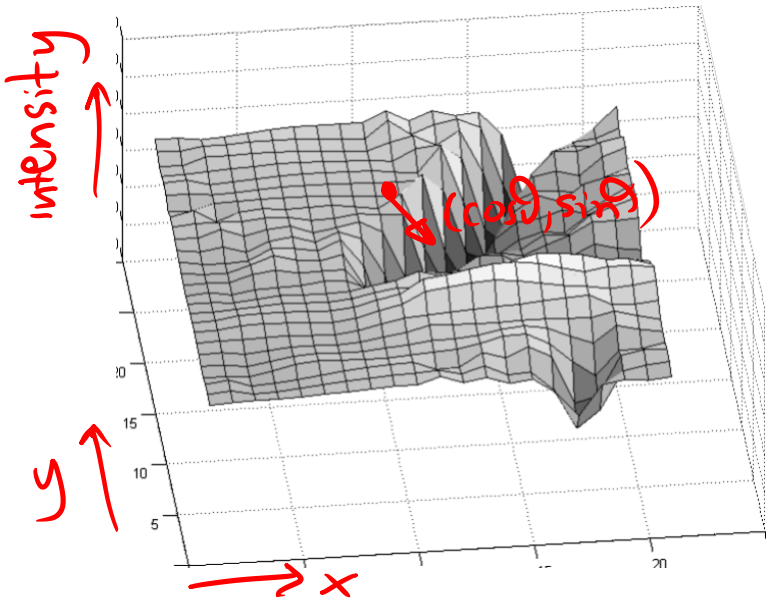
$$\left[\frac{\partial I}{\partial x}(0,0) \quad \frac{\partial I}{\partial y}(0,0)\right] \cdot \begin{bmatrix}\cos\theta \\ \sin\theta\end{bmatrix}$$

- Maximized when

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

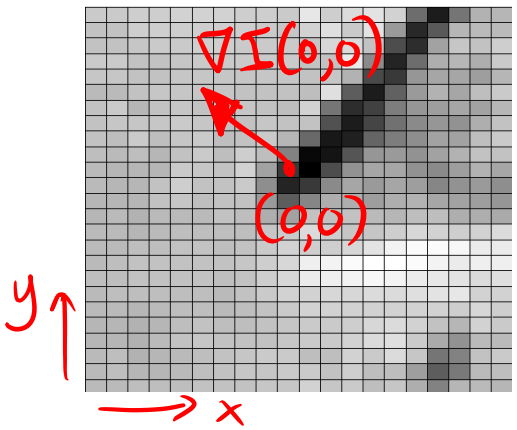
- 0 when (orthogonal)

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



- Derivatives along ANY other direction can be computed from these two!!

the image gradient



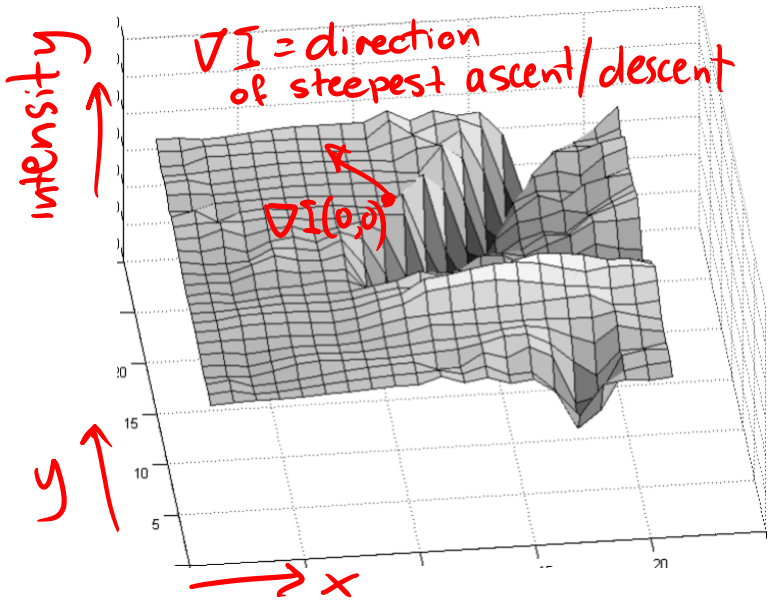
The Image Gradient

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

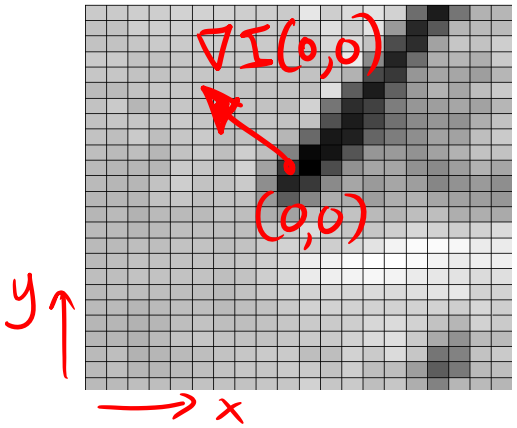
Directional derivative along v

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

- Maximized when $v = \nabla I(x,y)$ — ∇I = direction of greatest intensity change
- Zero when v and $\nabla I(x,y)$ are orthogonal



the image gradient

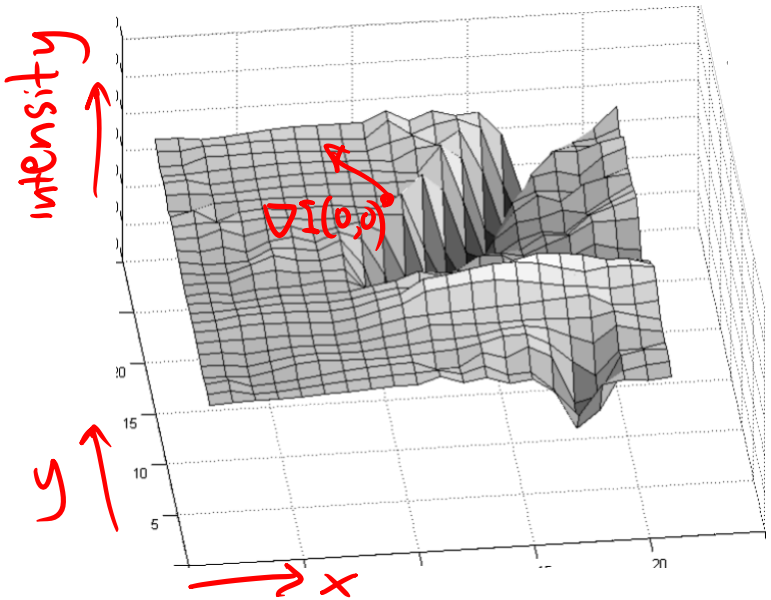


The Image Gradient

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

Directional derivative along v

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



- Zero when

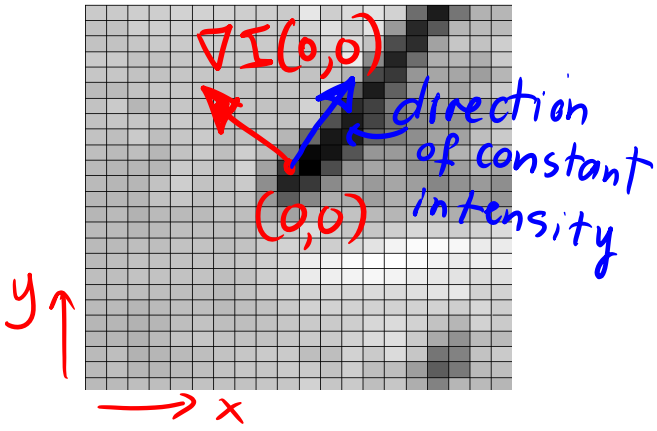
v, $\nabla I(0,0)$ orthogonal

In that case,

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

$$= I(0,0) = \text{constant}$$

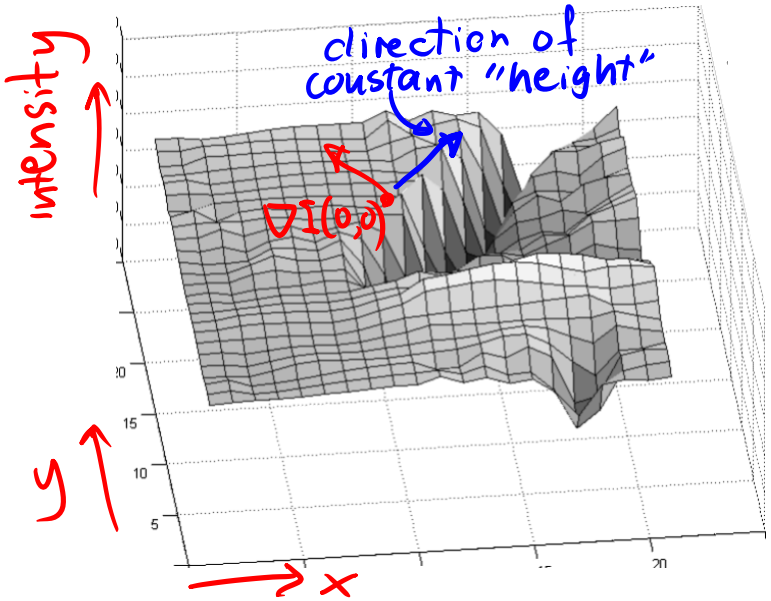
the image gradient



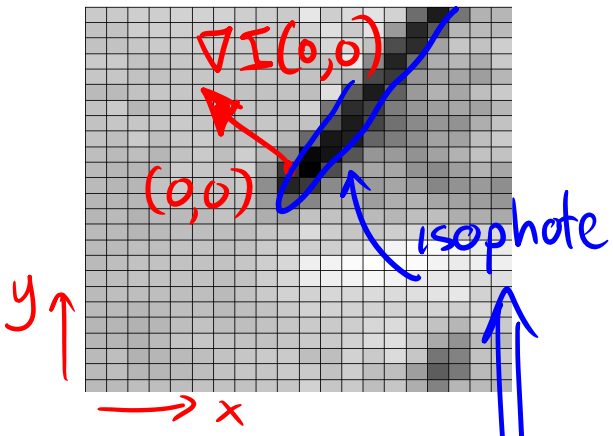
The Image Gradient

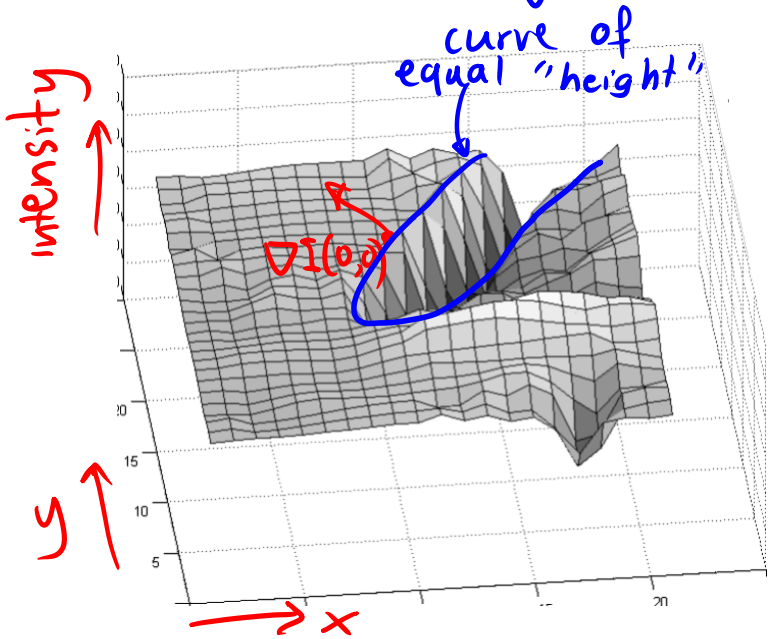
$$\nabla I(x,y) = \left[\frac{\partial I}{\partial x}(x,y) \quad \frac{\partial I}{\partial y}(x,y) \right]$$

- ∇I is orthogonal to the direction of constant intensity
- Zero when
  $\nabla I(0,0)$ orthogonal
- $\nabla I(x,y)$ is the <u>normal</u> vector of the isointensity curve (a.k.a. isophote) through pixel (x,y)



the image gradient



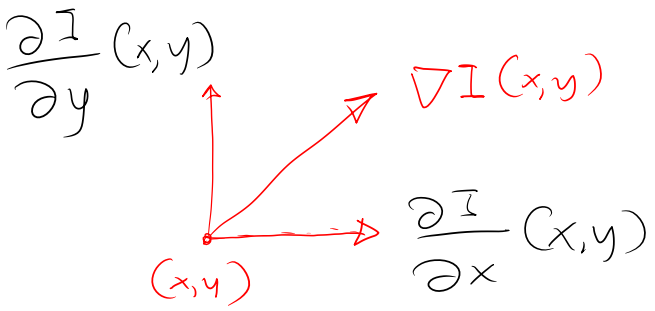


The Image Gradient

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

- ∇I is orthogonal to the direction of constant intensity
- Zero when $\nabla I(0,0)$ orthogonal
- $\nabla I(x,y)$ is the <u>normal</u> vector of the isointensity curve (a.k.a. isophote) through pixel (x,y)

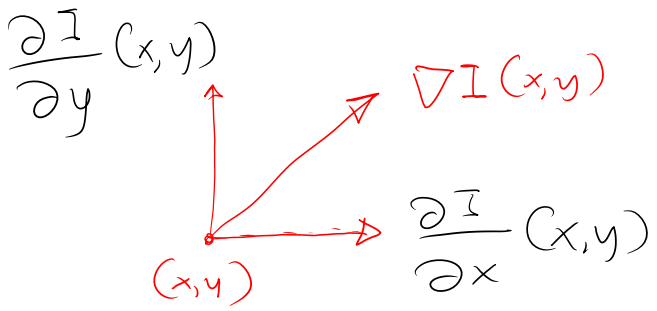
visualizing gradients



The Image Gradient

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

visualizing gradients: compute grayscale image



The Image Gradient

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

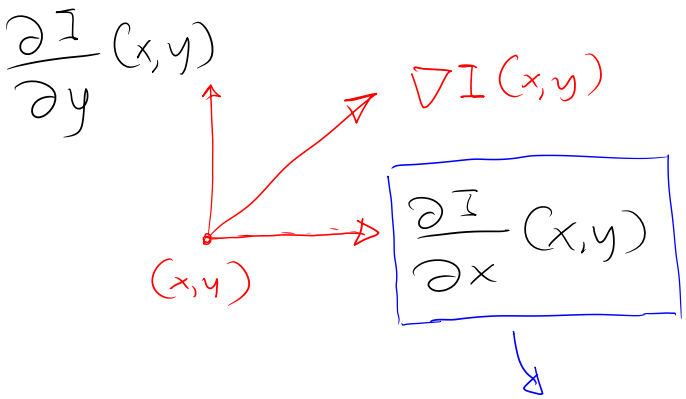
$I(x,y) =$

$\frac{1}{3}[\text{red}(x,y) +$

$\text{green}(x,y) +$

$\text{blue}(x,y)]$

visualizing gradients: compute partial along x



The Image Gradient

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

(computed using the 1D polydemo routines)

visualizing gradients: compute partial along y

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

$\nabla I(x,y)$

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

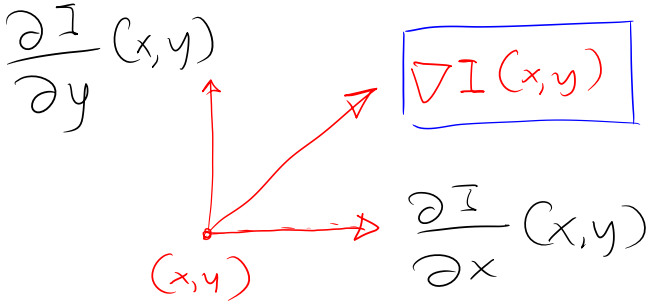
(x,y)

The Image Gradient

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

(computed using the 1D poly demo routines)

visualizing gradients: compute gradient magnitude



The Image Gradient

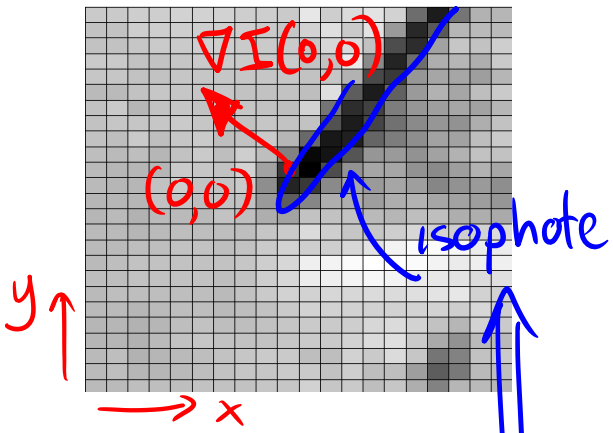
$$\nabla I(x,y) = \left[\frac{\partial I}{\partial x}(x,y) \quad \frac{\partial I}{\partial y}(x,y) \right]$$

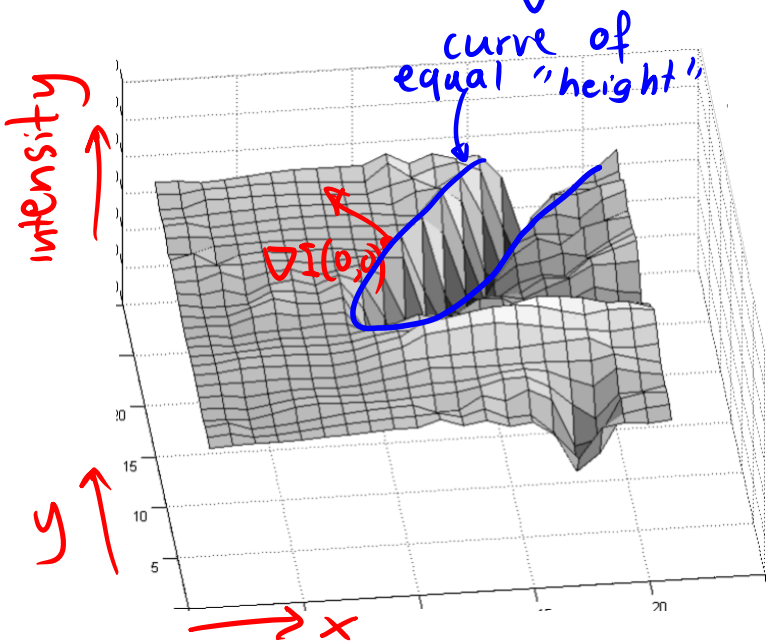
Gradient magnitude image

$|\nabla I(x,y)|$

= length of vector $\nabla I(x,y)$

isophote curves





The Image Gradient

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

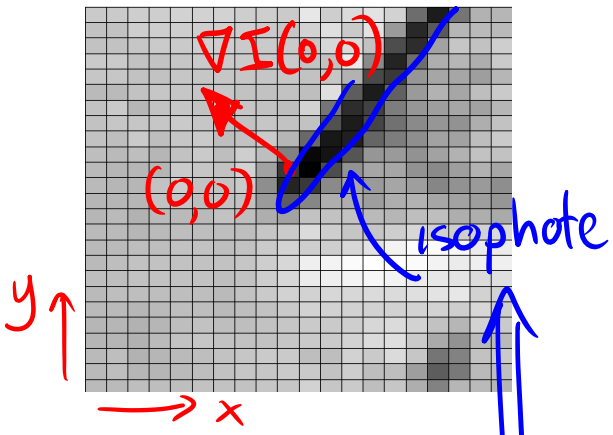
The Gradient Magnitude

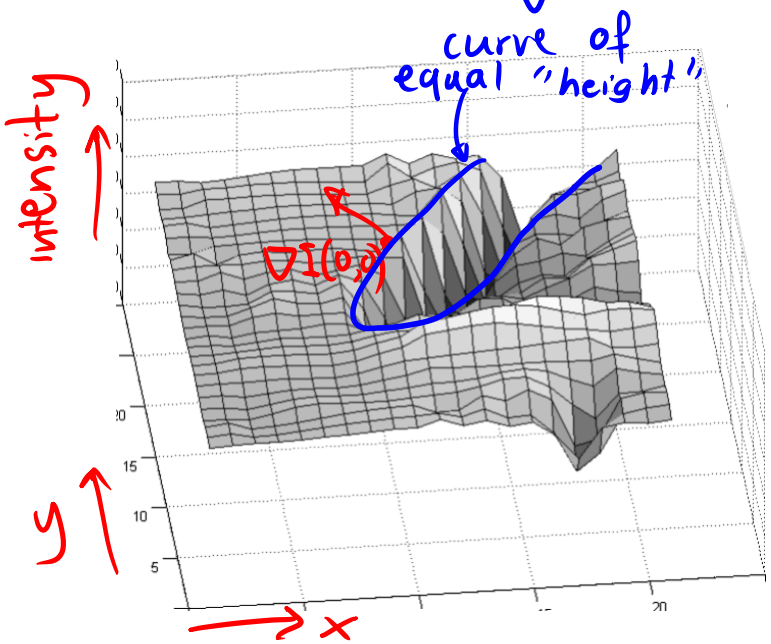
(= length of vector ∇I)

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

⇒ Tells us how quickly intensity changes in the neighborhood of pixel (x,y)

gradient orientation





The Image Gradient

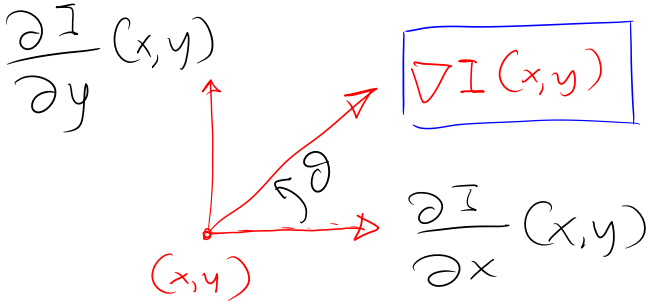
$$\nabla I(x,y) = \left[\frac{\partial I}{\partial x}(x,y) \quad \frac{\partial I}{\partial y}(x,y) \right]$$

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)

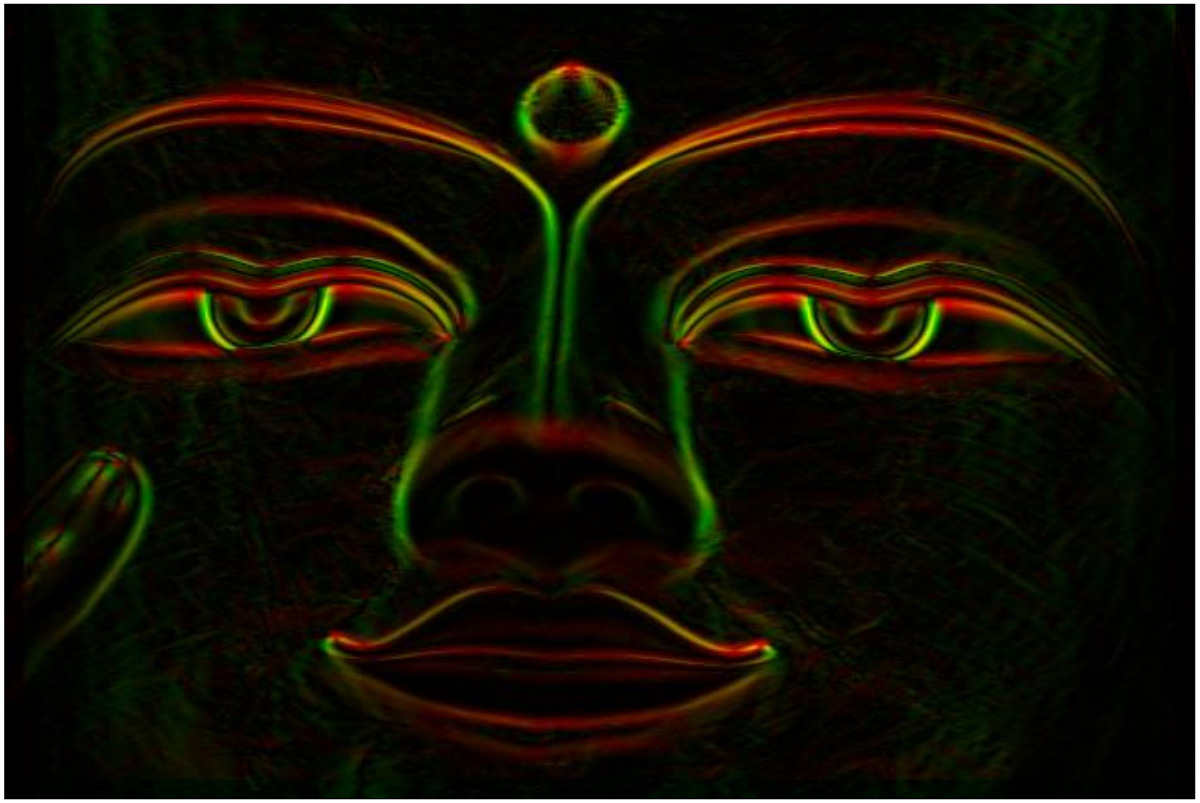
visualizing gradient magnitude & orientation



The Image Gradient

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

Visualizing magnitude & orientation



$red(x,y) = |\nabla I(x,y)| \cdot \sin\theta$

$green(x,y) = |\nabla I(x,y)| \cdot \cos\theta$

$blue(x,y) = 0$

gradient estimation using Gaussian filters

Let $\vec{\mathbf{n}} = (\cos\theta, \sin\theta)$ be the unit normal to the edge orientation.

The directional derivative of a 2D isotropic Gaussian, $G(\vec{\mathbf{x}}; \sigma^2) \equiv \frac{1}{2\pi\sigma^2} \exp\left(\frac{-(x^2+y^2)}{2\sigma^2}\right)$, is given by

$$\begin{aligned} \frac{\partial}{\partial \vec{\mathbf{n}}} G(\vec{\mathbf{x}};\, \sigma^2) &= \nabla G(\vec{\mathbf{x}};\, \sigma^2) \cdot \vec{\mathbf{n}} \\ &= G_x(\vec{\mathbf{x}};\, \sigma^2) \cos\theta \;+\; G_y(\vec{\mathbf{x}};\, \sigma^2) \sin\theta \end{aligned}$$

where $G_x \equiv \frac{\partial G}{\partial x}$, $G_y \equiv \frac{\partial G}{\partial y}$, and $\nabla G \equiv (G_x,\, G_y)$.

The direction of steepest ascent/descent at each pixel is given by the direction of the image gradient:

$$\vec{\mathbf{R}}(\vec{\mathbf{x}}) \;=\; \nabla G(\vec{\mathbf{x}};\, \sigma^2) \;*\; I(\vec{\mathbf{x}})\;.$$

The unit edge normal is then

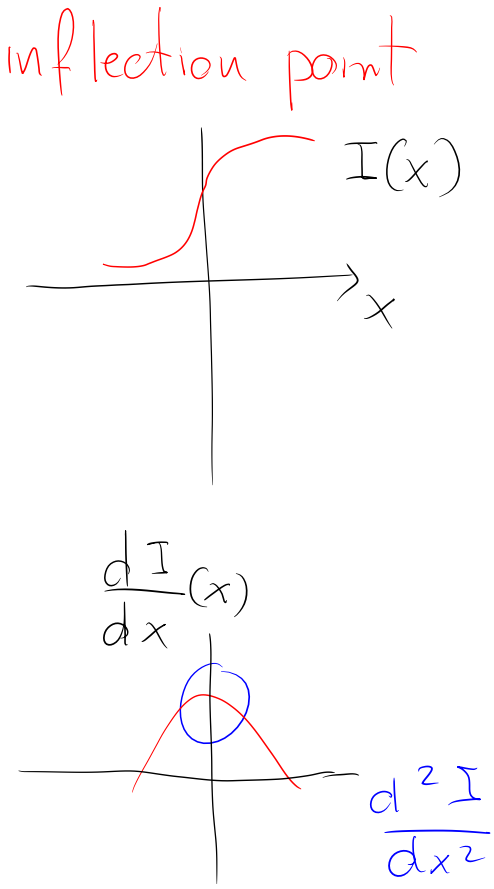
$$\vec{\mathbf{n}}(\vec{\mathbf{x}}) \;=\; \frac{\vec{\mathbf{R}}(\vec{\mathbf{x}})}{|\vec{\mathbf{R}}(\vec{\mathbf{x}})|}$$

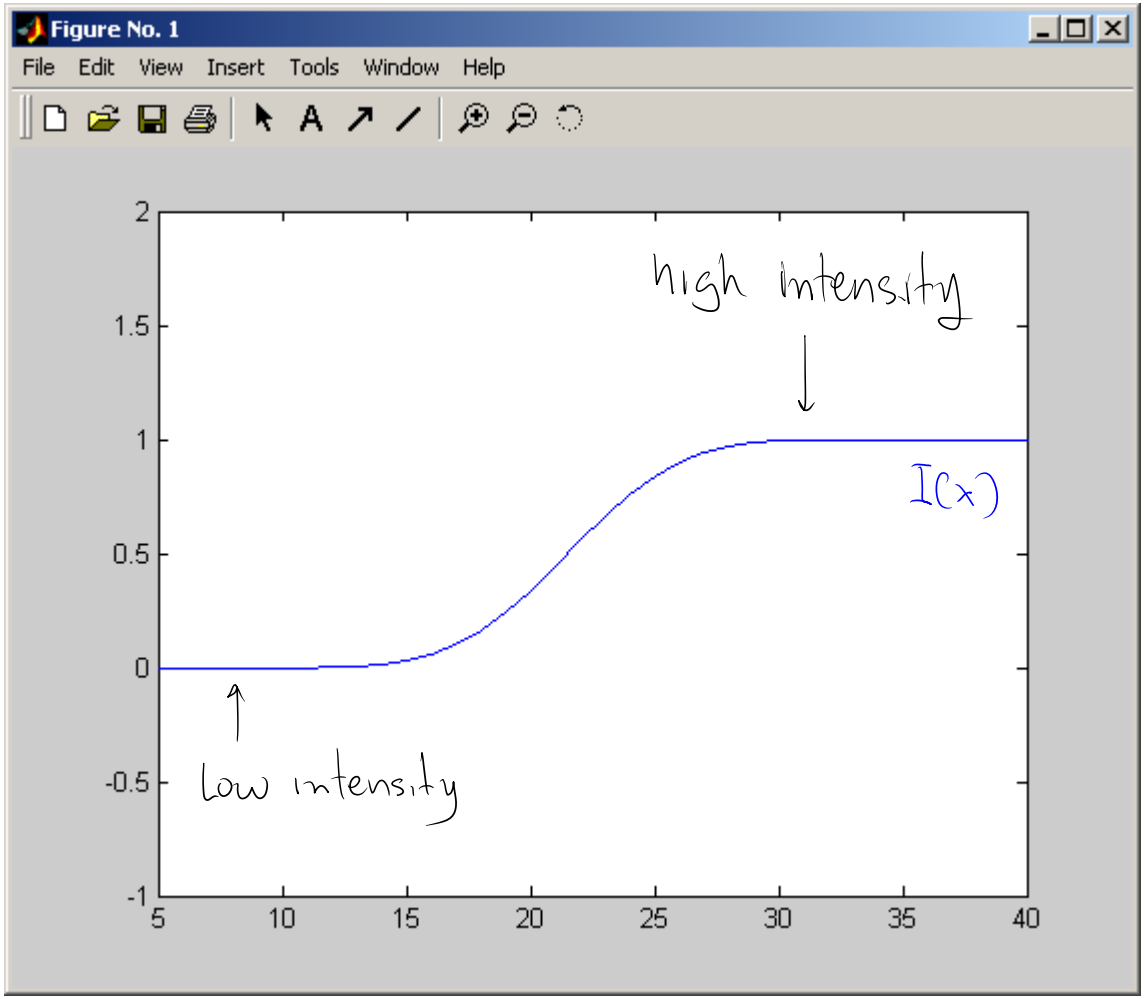
Topic 06:

Edge Detection Basics

- Edge detection in 2D
- Sub-pixel edge localization
- The Canny edge detector
- Gradient-domain image processing

1st deriv extrema vs. zero-crossings of 2nd deriv

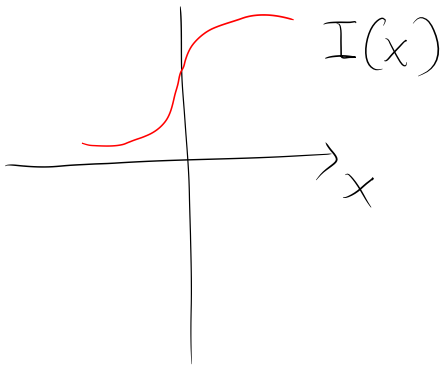


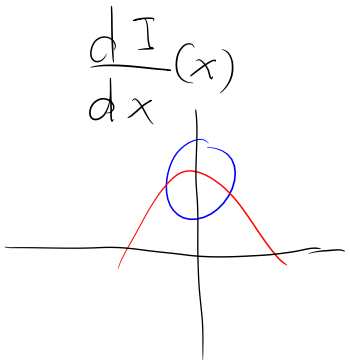


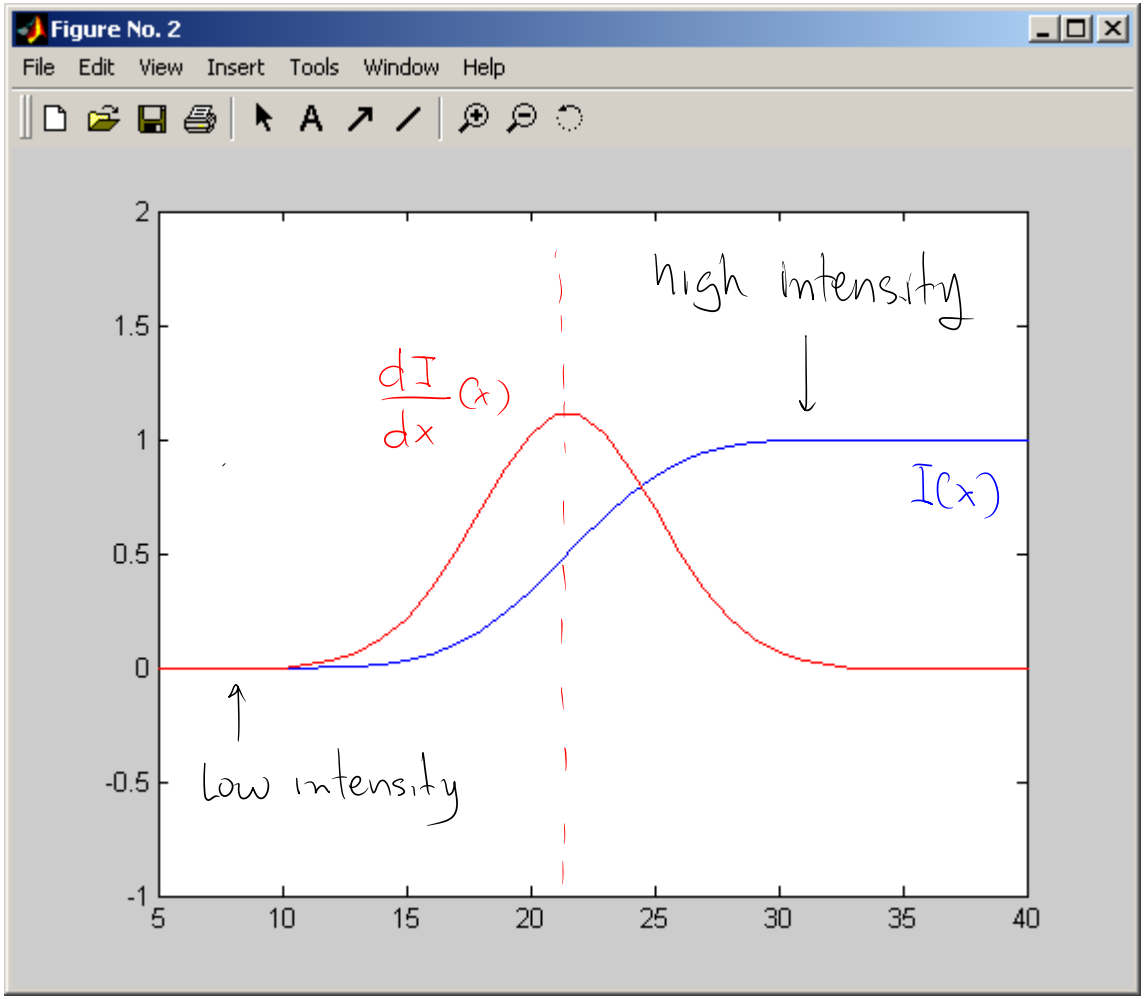
1st deriv extrema vs. zero-crossings of 2nd deriv

Location of edge = location of maximum / minimum of $\frac{dI}{dx}$

inflection point



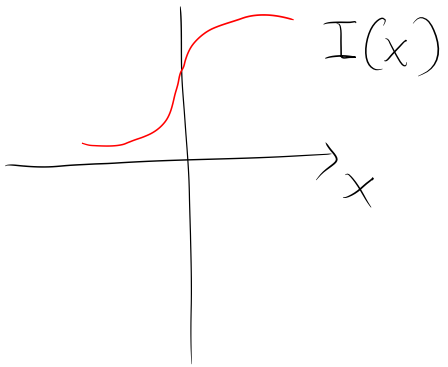


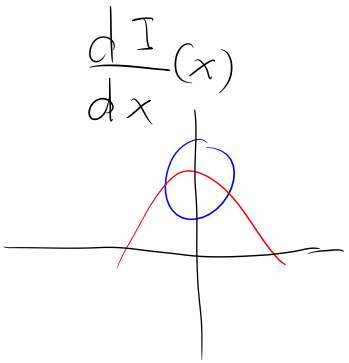


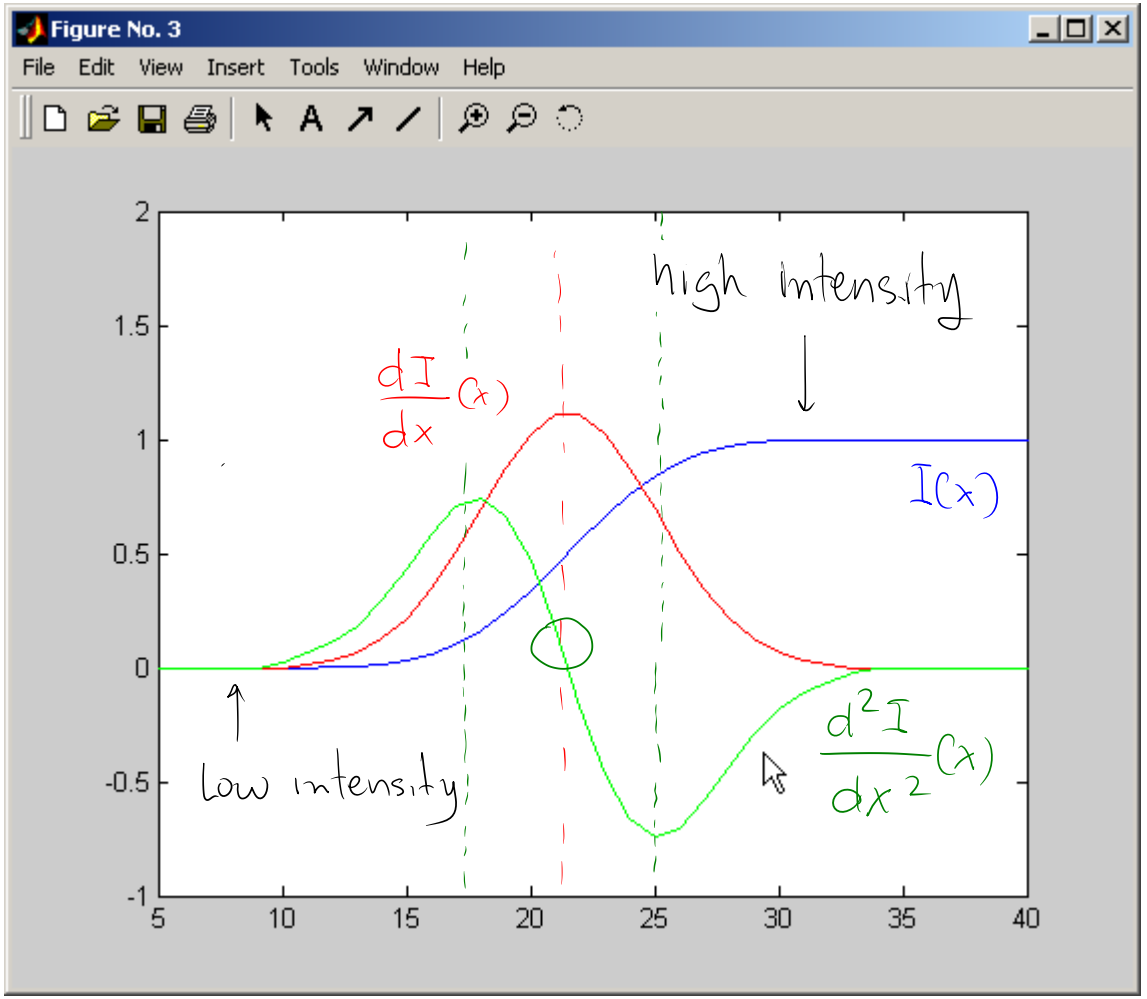
1st deriv extrema vs. zero-crossings of 2nd deriv

Location of edge = Zero-crossing of $\frac{d^2}{dx^2} I(x)$

inflection point







subpixel localization using Gaussian filters

So, given a local maxima and its normal, $\vec{\mathbf{n}} = (\cos\theta, \sin\theta)$, we can compute the 2^{nd}-order directional derivative in the local region:

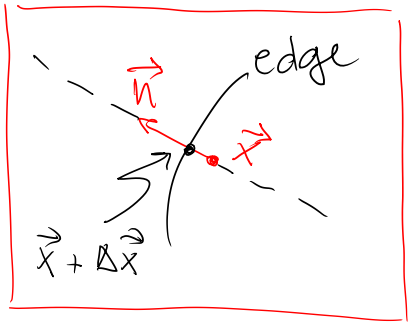
$$\begin{aligned} \frac{\partial^2}{\partial \vec{\mathbf{n}}^2} G(\vec{\mathbf{x}}) * I(\vec{\mathbf{x}}) \;\; = \;\; & \cos^2\theta \, G_{xx}(\vec{\mathbf{x}}) * I(\vec{\mathbf{x}}) \; + \\ & 2 \, \cos\theta \sin\theta \, G_{xy}(\vec{\mathbf{x}}) * I(\vec{\mathbf{x}}) \; + \\ & \sin^2\theta \, G_{yy}(\vec{\mathbf{x}}) * I(\vec{\mathbf{x}}) \, , \end{aligned} \tag{1}$$

where G is a Gaussian. Note that the three filters, $G_{xx} \equiv \frac{\partial^2 G}{\partial x^2}$, $G_{xy} \equiv \frac{\partial^2 G}{\partial x \partial y}$ and $G_{yy} \equiv \frac{\partial^2 G}{\partial y^2}$ can be applied to the image independent of $\vec{\mathbf{n}}$.

subpixel localization using Gaussian filters: steps

① 2nd order Taylor series expansion of $D = G * I$

In vector notation



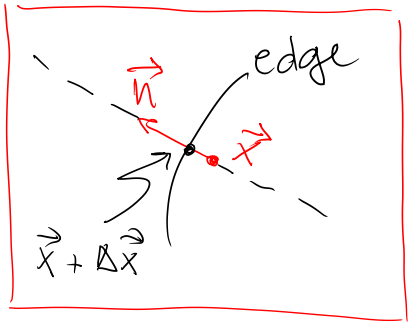
$$\vec{x} = \begin{bmatrix} x \\ y \end{bmatrix} \quad \Delta\vec{x} = \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix}$$

$$D\left(\vec{x} + \Delta\vec{x}\right) = D(\vec{x}) + \left(\frac{\partial D}{\partial \vec{x}}\right)^{T} \cdot \Delta\vec{x}$$

$$+ \frac{1}{2}\left(\Delta\vec{x}\right)^{T} \cdot \frac{\partial^2 D}{\partial \vec{x}^2} \cdot \left(\Delta\vec{x}\right)$$

subpixel localization using Gaussian filters: steps

① 2nd order Taylor series expansion of $D = G * I$



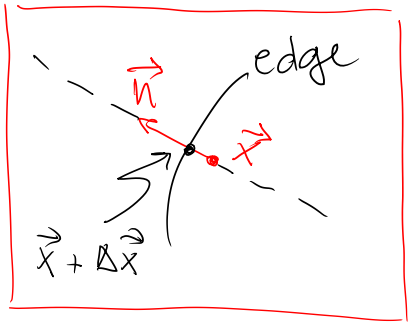
$$D(\vec{x} + \Delta\vec{x}) = D(\vec{x}) + \left(\frac{\partial D}{\partial \vec{x}}\right)^T \cdot \Delta\vec{x} + \frac{1}{2}(\Delta\vec{x})^T \cdot \frac{\partial^2 D}{\partial \vec{x}^2} \cdot (\Delta\vec{x})$$

② Take derivatives with respect to $\Delta\vec{x}$

$$\frac{\partial D}{\partial(\Delta\vec{x})} = \left(\frac{\partial D}{\partial \vec{x}}\right)^T + (\Delta\vec{x})^T \left(\frac{\partial^2 D}{\partial \vec{x}^2}\right)^T$$

subpixel localization using Gaussian filters: steps

① 2nd-order Taylor series expansion of $D = G * I$



$$D(\vec{x} + \Delta\vec{x}) = D(\vec{x}) + \left(\frac{\partial D}{\partial \vec{x}}\right)^T \Delta\vec{x} + \frac{1}{2}(\Delta\vec{x})^T \cdot \frac{\partial^2 D}{\partial \vec{x}^2} \cdot (\Delta\vec{x})$$

② Take derivatives with respect to $\Delta\vec{x}$

$$\frac{\partial D}{\partial(\Delta\vec{x})} = \left(\frac{\partial D}{\partial \vec{x}}\right)^T + (\Delta\vec{x})^T \left(\frac{\partial^2 D}{\partial \vec{x}^2}\right)^T$$

③ Extremum of $\frac{\partial D}{\partial(\Delta\vec{x})}$ along $\vec{n}$ $\Leftrightarrow$ its derivative is zero

$\Rightarrow$ solve for $\Delta\vec{x}$ where this occurs.

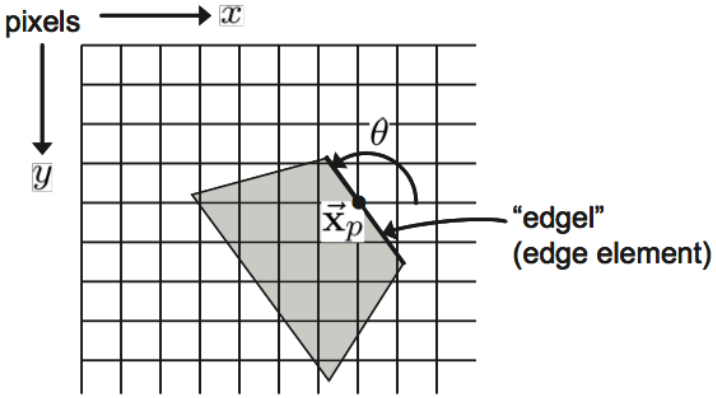
Topic 06:

Edge Detection Basics

- Edge detection in 2D
- Sub-pixel edge localization
- The Canny edge detector
- Gradient-domain image processing

edge detection in discrete images

- **Edgels (edge elements):** significant local variations in image brightness, characterized by the position $\vec{\mathbf{x}}_p$ and the orientation θ of the brightness variation. (Usually $\theta \bmod \pi$ is sufficient.)



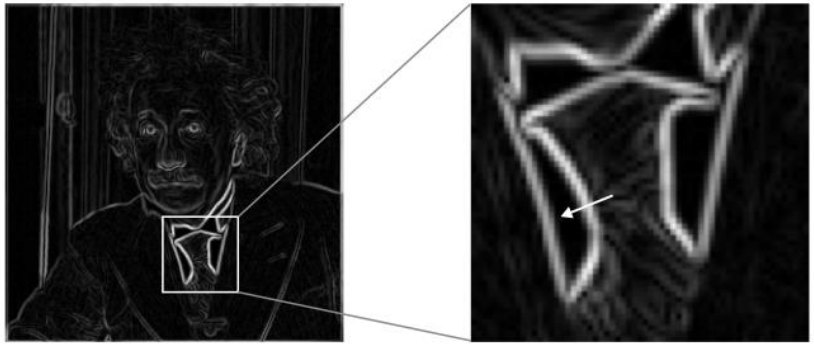
- **Edge:** a sequence of edgels forming a smooth curve

Two Problems:

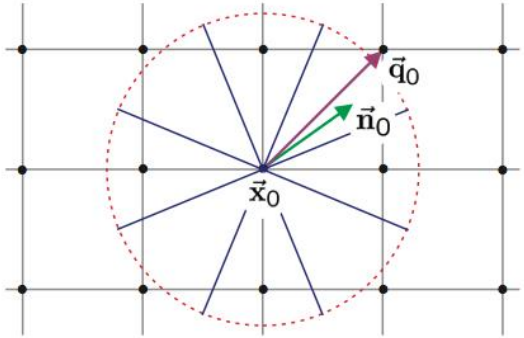
1. estimating edgels
2. grouping edgels into edges

non-maximum suppression

Search for local maxima of gradient magnitude $S(\vec{\mathbf{x}}) = |\vec{\mathbf{R}}(\vec{\mathbf{x}})|$, in the direction normal to local edge, $\vec{\mathbf{n}}(\vec{\mathbf{x}})$, suppressing all responses except for local maxima (called non-maximum suppression).



In practice, the search for local maxima of $S(\vec{\mathbf{x}})$ takes place on the discrete sampling grid. Given $\vec{\mathbf{x}}_0$, with normal $\vec{\mathbf{n}}_0$, compare $S(\vec{\mathbf{x}}_0)$ to nearby pixels closest to the direction of $\pm\vec{\mathbf{n}}_0$, e.g., pixels at $\vec{\mathbf{x}}_0 \pm \vec{\mathbf{q}}_0$, where $\vec{\mathbf{q}}_0$ is $\frac{1}{2\sin(\pi/8)}\vec{\mathbf{n}}_0$ with its elements rounded to the nearest integer.



Canny edge detection algorithm

1. Convolve with gradient filters (at multiple scales)

$$\vec{\mathbf{R}}(\vec{\mathbf{x}}) \equiv (R_x(\vec{\mathbf{x}}),\, R_y(\vec{\mathbf{x}})) = \nabla G(\vec{\mathbf{x}};\, \sigma^2) * I(\vec{\mathbf{x}})\,.$$

2. Compute response magnitude, $S(\vec{\mathbf{x}}) = \sqrt{R_x^2(\vec{\mathbf{x}}) + R_y^2(\vec{\mathbf{x}})}$.

3. Compute local edge orientation (represented by unit normal):

$$\vec{\mathbf{n}}(\vec{\mathbf{x}}) = \begin{cases} (R_x(\vec{\mathbf{x}}),\, R_y(\vec{\mathbf{x}}))/S(\vec{\mathbf{x}}) & \text{if } S(\vec{\mathbf{x}}) > threshold \\ 0 & \text{otherwise} \end{cases}$$

4. Peak detection (non-maximum suppression along edge normal)

5. Non-maximum suppression through scale, and hysteresis thresholding along edges (see Canny (1986) for details).

Implementation Remarks:

Separability: Partial derivatives of an isotropic Gaussian:

$$\frac{\partial}{\partial x} G(\vec{\mathbf{x}};\, \sigma^2) = -\frac{x}{\sigma^2}\, G(x;\, \sigma^2)\, G(y;\, \sigma^2)\,.$$

Filter Support: In practice, it's good to sample the impulse response so that the support radius $K \geq 3\sigma_r$. Common values for K are 7, 9, and 11 (i.e., for $\sigma \approx 1, 4/3$, and $5/3$).

Canny edge detection example

Image `three.pgm`

Gaussian Blur $\sigma = 1.0$

Gradient in x

Gradient in y

Canny edge detection example

Gradient Strength

Gradient Orientations

Canny Edgels

Edgel Overlay

Colour gives gradient direction (red – 0°; blue – 90°; green – 270°)

Topic 06:

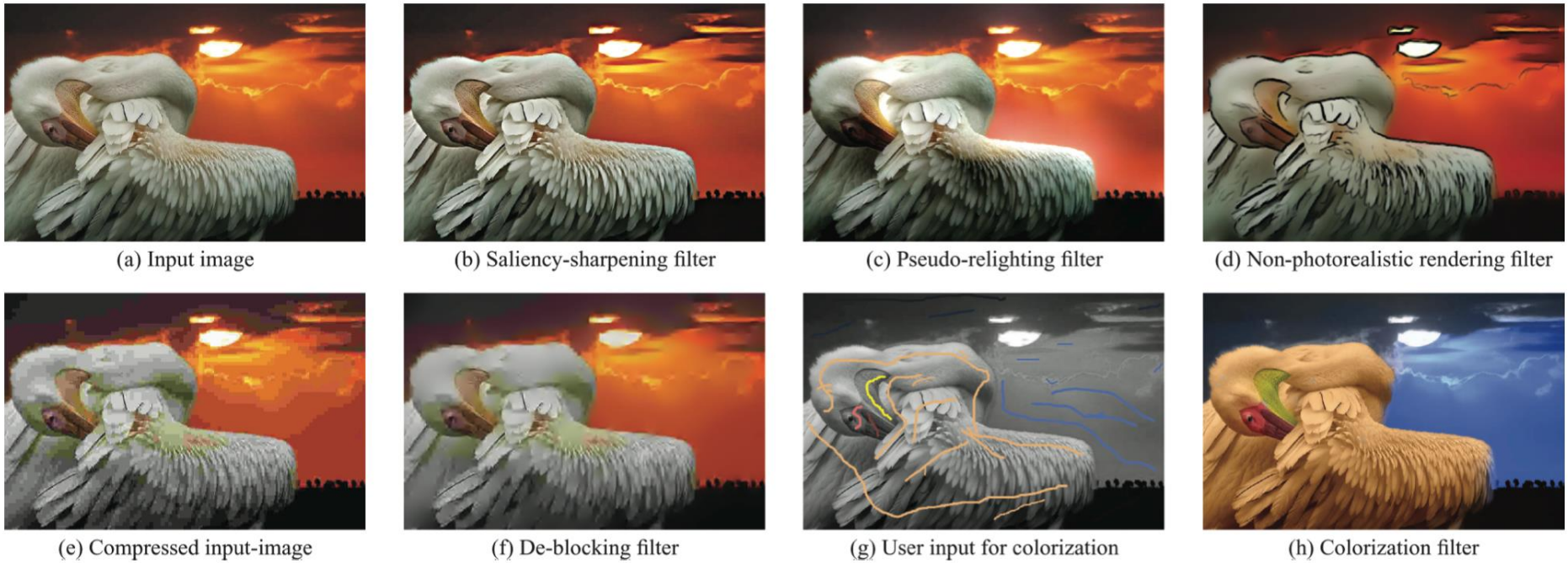
Edge Detection Basics

- Edge detection in 2D
- Sub-pixel edge localization
- The Canny edge detector
- Gradient-domain image processing

manipulating images in the gradient domain

Basic idea:

1. Compute gradient of input image
2. Manipulate gradient by applying (nonlinear) filters
3. Reconstruct image by solving the Poisson Equation



(a) Input image (b) Saliency-sharpening filter (c) Pseudo-relighting filter (d) Non-photorealistic rendering filter

(e) Compressed input-image (f) De-blocking filter (g) User input for colorization (h) Colorization filter

Fig. 1. The figure shows some of the image enhancement filters we have created using the GradientShop optimization framework. GradientShop has been designed to allow applications to explore gradient-domain solutions for various image processing problems.

(Bhat et al, 2010)

manipulating images in the gradient domain

Basic idea:

1. Compute gradient of input image

$$\nabla I = (g_x, g_y)$$

2. Manipulate gradient by applying (nonlinear) filters

$$(g_x, g_y) \longrightarrow (h_x, h_y)$$

3. Reconstruct image by solving the Poisson Equation

find image J that minimizes

$$\left\| \left(\frac{\partial J}{\partial x} - h_x\right)^2 + \left(\frac{\partial J}{\partial y} - h_y\right)^2 \right\|$$

manipulating images in the gradient domain

Basic idea:

① Compute gradient of input image

$$\nabla I = (g_x, g_y)$$

② Manipulate gradient by applying (nonlinear) filters

$$(g_x, g_y) \longrightarrow (h_x, h_y)$$

③ Reconstruct image by solving the Poisson Equation

- image must satisfy $\nabla^2 I = \mathrm{div}(h_x, h_y)$
- usually solved by weighted least squares
- see Bhat et al (2010) for an example implementation & application of this idea

manipulating images in the gradient domain

Basic idea:

1. Compute gradient of input image

$$\nabla I = (g_x, g_y)$$

2. Manipulate gradient by applying (nonlinear) filters

$$(g_x, g_y) \longrightarrow (h_x, h_y)$$

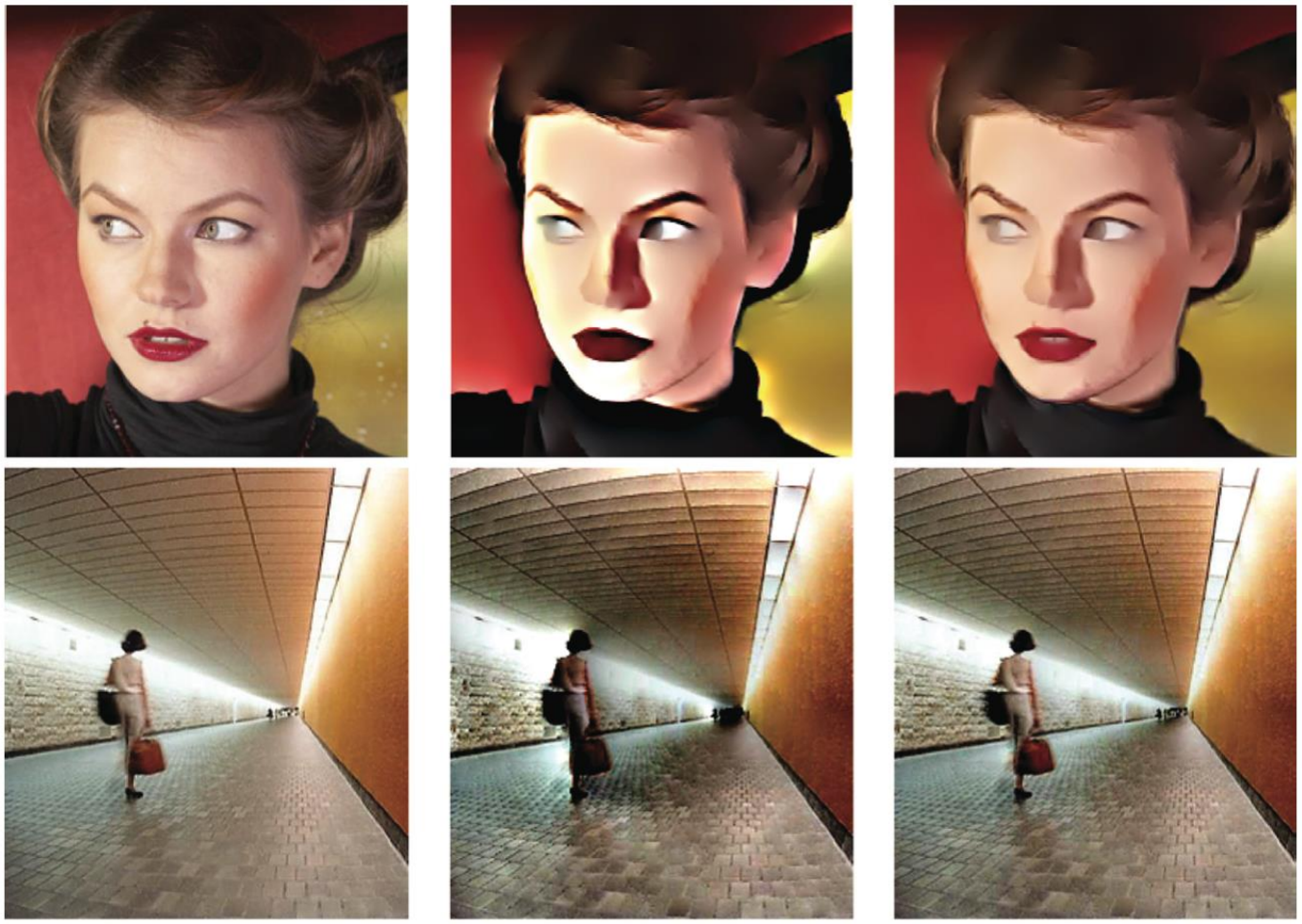
3. Reconstruct image by solving the Poisson Equation

image must satisfy $\nabla^2 I = \operatorname{div}(h_x, h_y)$

$$\frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 I}{\partial y^2}$$

$$\frac{\partial h_x}{\partial x} + \frac{\partial h_y}{\partial y}$$

manipulating images in the gradient domain



Original image | Uniform weighting result | Robust weighting result

(Bhat et al, 2010)

further readings (in addition to those in notes)

Hwang et al, Difference-based image noise modeling using Skellam distribution, IEEE Trans PAMI, v34, n7, 2012

Witkin, Scale-space filtering, Proc. ICASSP, 1984

Baband et al, Uniqueness of the Gaussian kernel for scale-space filtering, IEEE Trans. PAMI, v.1, 1986

Bhat et al, Gradient-shop: A gradient-domain optimization framework for image and video filtering, ACM TOG v.29, n.2, 2010