Basic image processing

Vlad Shakhuro

20 September 2023

Outline

1. What is image processing?
2. Tonal correction
3. Color correction
4. Noise reduction, convolution operation
5. Fast filtering
6. Edge detection

What is image processing?

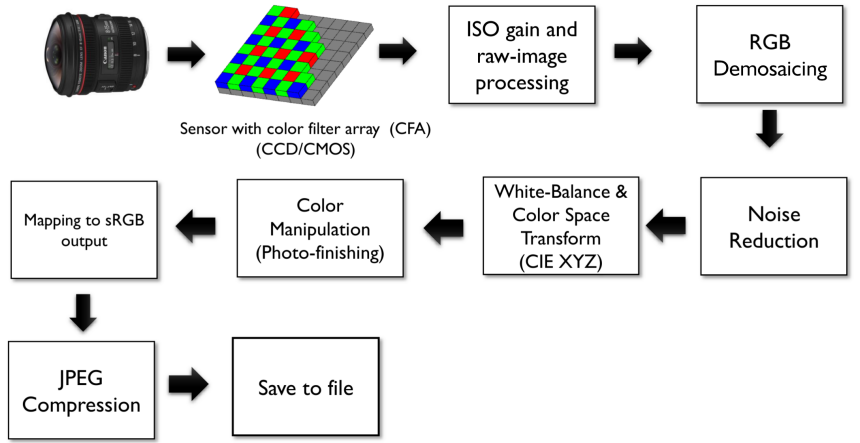
Tasks and methods where input and output are images

$$Y = f(X), \quad X \in \mathbb{R}^{H_{\text{in}} \times W_{\text{in}} \times C_{\text{in}}}$$
$$Y \in \mathbb{R}^{H_{\text{out}} \times W_{\text{out}} \times C_{\text{out}}}$$

Goals:

1. Improve image for human perception
2. Improve image for automatic recognition
3. Extracting features for further analysis
4. Conversion for technical needs
5. Entertainment (special effects)

Camera imaging pipeline



Outline

1. What is image processing?
2. Tonal correction
3. Color correction
4. Noise reduction, convolution operation
5. Fast filtering
6. Edge detection

Image histogram

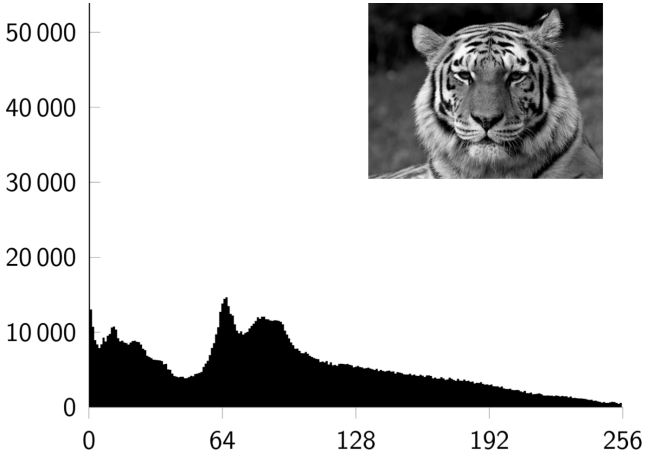


Image histogram

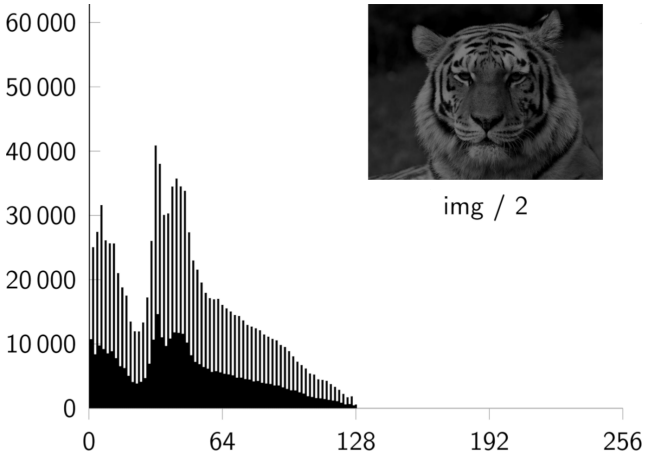
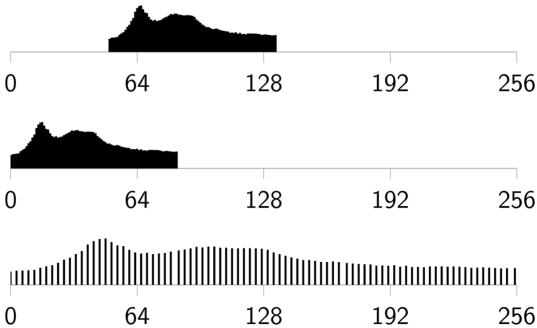


Image values don't fully use available luminance range or concentrate around certain values

Autocontrast



Point operator: pixel output value is defined only by it's own value, i.e. all pixels are processed independently. Simplest case is linear correction:

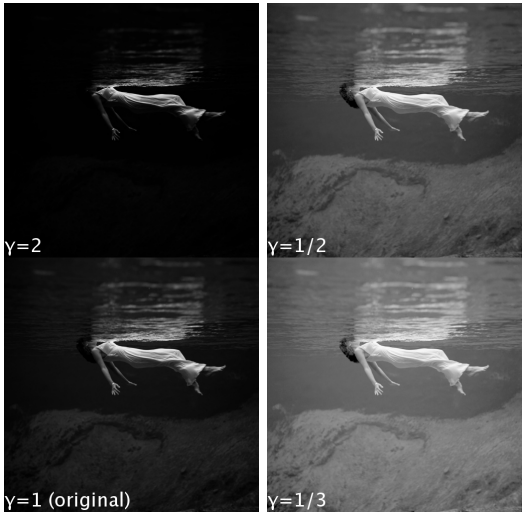
$$f(x) = (x - x_{\text{min}}) \frac{255}{x_{\text{max}} - x_{\text{min}}}$$

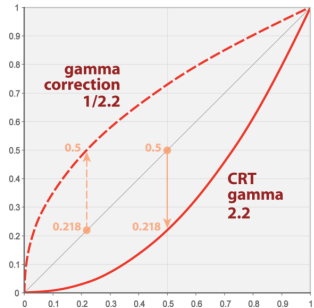
Stable autocontrast

To make autocontrast more stable to noise, drop 5% of min and max values to compute x_{min} and x_{max}

How make contrast correction for color images?

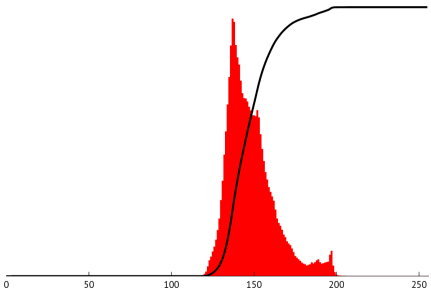
Nonlinear contrast correction

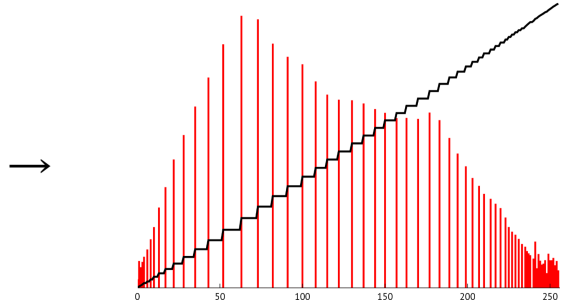




$$f(x) = c \cdot x^{\gamma}$$

Histogram equalization





$$f(x) = \text{round}\left(\frac{cdf(x) - cdf_{\text{min}}}{\#pix - 1} \cdot 255\right)$$

Histogram equalization example

Outline

1. What is image processing?
2. Tonal correction
3. Color correction
4. Noise reduction, convolution operation
5. Fast filtering
6. Edge detection

Color correction

Correcting with neutral card

Take a photo of white or gray card under desired illumination

Multiply RGB values by $1/r_w$, $1/g_w$ and $1/b_w$

What drawbacks does this method has?

Color checker

Color template for professional use (filmmaking, high-quality photography)

Possible to use more complex nonlinear models, e.g. polynomial with 24 degrees of freedom

Why use more complex models?

Gray world

Gray world assumption: average per-channel value should be equal for all three channels. Compute multipliers r_w, g_w, b_w that correct image according to this assumption

$$Avg = \frac{\bar{R} + \bar{G} + \bar{B}}{3}$$

$$r_w = \frac{\bar{R}}{Avg}, \quad g_w = \frac{\bar{G}}{Avg}, \quad b_w = \frac{\bar{B}}{Avg}$$

Gray world example

Outline

1. What is image processing?
2. Tonal correction
3. Color correction
4. Noise reduction, convolution operation
5. Fast filtering
6. Edge detection

Noise

Gaussian noise:

$$I[i,j] = I_{\text{true}}[i,j] + \varepsilon_{i,j}$$

$$\varepsilon_{i,j} \sim \mathcal{N}(\mu, \sigma)$$

Data drop-out noise:

- Salt-and-pepper: random black and white pixels
- Impulse noise: random white pixels

Image averaging

What type of noise does averaging suppress?

Image convolution

Kernel examples

$$*\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} =$$

Kernel examples

$$*\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} =$$

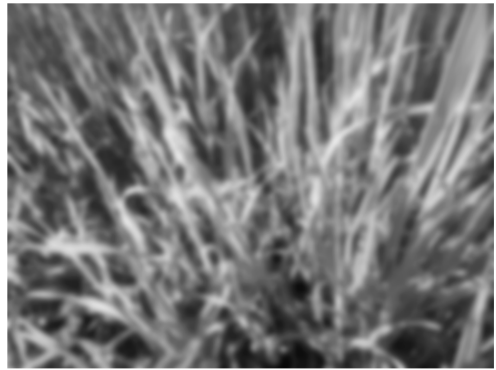
Kernel examples

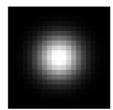
$$* \quad \frac{1}{16} \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} =$$

Kernel examples

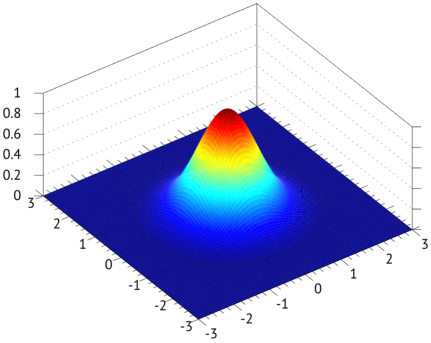
$$* \quad \frac{1}{16} \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} =$$

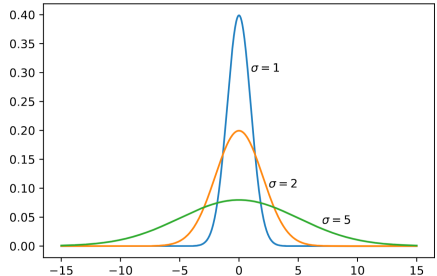
Blurring image





Gaussian kernel





Filter radius r is 3σ,
filter size is $2r + 1$

$$G_\sigma(x, y) = \frac{1}{2\pi\sigma^2} e^{\frac{-(x+y)^2}{2\sigma^2}}$$

Sharpening

We know how to blur image. How can we sharpen image?

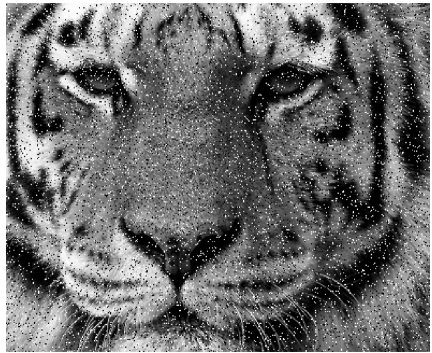
Sharpening

We know how to blur image. How can we sharpen image?

Sharpening

$$* \quad \frac{1}{10} \quad \begin{array}{|c|c|c|} \hline -1 & -2 & -1 \\ \hline -2 & 22 & -2 \\ \hline -1 & -2 & -1 \\ \hline \end{array}$$

Impulse noise



Will gaussian filter help with salt-and-pepper noise?

Median filter

10	15	20
23	90	27
33	31	30

10 15 20 23 27 30 31 33 90

Median filter

10	15	20
23	90	27
33	31	30

10 15 20 23 27 30 31 33 90

10	15	20
23	27	27
33	31	30

Gaussian and median filter comparison

Linear filters and convolution

Spatial filter f is called **linear** if two properties are fulfilled:

$$f(I_1 + I_2) = f(I_1) + f(I_2)$$
$$f(kI) = k \cdot f(I)$$

Filter f is called invariant to shift if $f(\text{shift}(I)) = \text{shift}(f(I))$

Any linear filter invariant to shift may be expressed as convolution with some kernel

Linear filters and convolution

Spatial filter f is called **linear** if two properties are fulfilled:

$$f(I_1 + I_2) = f(I_1) + f(I_2)$$
$$f(kI) = k \cdot f(I)$$

Filter f is called invariant to shift if $f(\text{shift}(I)) = \text{shift}(f(I))$

Any linear filter invariant to shift may be expressed as convolution with some kernel

Is median filter linear?

Linear filters and convolution

Spatial filter f is called **linear** if two properties are fulfilled:

$$f(I_1 + I_2) = f(I_1) + f(I_2)$$
$$f(kI) = k \cdot f(I)$$

Filter f is called invariant to shift if $f(\text{shift}(I)) = \text{shift}(f(I))$

Any linear filter invariant to shift may be expressed as convolution with some kernel

Is median filter linear? No

$$\text{med}\left(\begin{bmatrix}1&1&1\\1&1&2\\2&2&2\end{bmatrix} + \begin{bmatrix}0&0&0\\0&1&0\\0&0&0\end{bmatrix}\right) \neq \text{med}\left(\begin{bmatrix}1&1&1\\1&1&2\\2&2&2\end{bmatrix}\right) + \text{med}\left(\begin{bmatrix}0&0&0\\0&1&0\\0&0&0\end{bmatrix}\right)$$

Convolution properties

Commutativity:
$I * K = K * I$

Associativity:
$(I * K_1) * K_2 = I * (K_1 * K_2)$

Distributivity:
$I * (K_1 + K_2) = (I * K_1) + (I * K_2)$

Multiplying by scalar:
$aI * K = I * aK = a(I * K)$

Convolution with identity kernel:
$I * E = I, \quad E = [\dots, 0, 0, 1, 0, 0, \dots]$

Outline

1. What is image processing?
2. Tonal correction
3. Color correction
4. Noise reduction, convolution operation
5. Fast filtering
6. Edge detection

Fast box filter

$$* \quad \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Fast box filter

$$S[x, y] = \sum_{i=0}^{x} \sum_{j=0}^{y} I[i, j]$$

Fast box filter

$$Sum(x_1, y_1, x_2, y_2) = A + B - C - D$$
$$= S[x_2, y_2] + S[x_1 - 1, y_1 - 1] - S[x_1 - 1, y_2] - S[x_2, y_1 - 1]$$

Separable filter

Filter is separable if it can be factorized into two 1D convolutions:

$$G_\sigma = \frac{1}{2\pi\sigma^2} e^{\frac{-x^2-y^2}{2\sigma^2}} = \left(\frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{x^2}{2\sigma^2}}\right)\left(\frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{y^2}{2\sigma^2}}\right)$$

Let's filter an image with $N \times N$ pixels using a kernel with $K \times K$ elements.
Filtering complexity:

2D kernel: N^2K^2

separable kernel: N^2K

Approximating using a separable filter

Filter is separable ↔ kernel rank $= 1$

$$\mathcal{K} = \boldsymbol{v}\boldsymbol{h}^T.$$

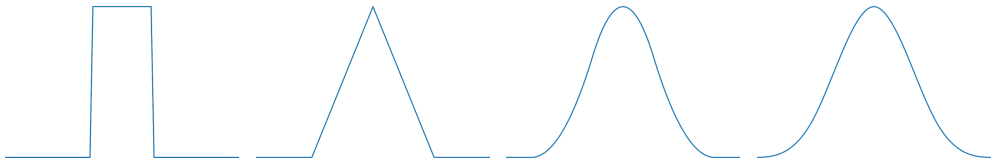
Non-separable filter may be factorized using SVD

$$\mathcal{K} = \sum_i \sigma_i \boldsymbol{u}_i \boldsymbol{v}_i^T$$

and approximated with a series of separable filters

$$\mathcal{K}_1 = \sqrt{\sigma_1}\boldsymbol{u}_1 \cdot \sqrt{\sigma_1}\boldsymbol{v}_1^T,\ \mathcal{K}_2 = \sqrt{\sigma_2}\boldsymbol{u}_2 \cdot \sqrt{\sigma_2}\boldsymbol{v}_2^T, \ldots$$

Approximating gaussian filter with a box filter



Gaussian blur with parameter σ may be approximated using N box filters with width $\sigma\sqrt{12/N}$

In practice $N = 3$ is enough. Partially-quadratic kernel approximates gaussian blur with 3% error

see also approximation in SVG standard

Fast median filter

Let's filter an image with $N \times N$ pixels using a kernel with $K \times K$ elements:

Fast median filter

Let's filter an image with $N \times N$ pixels using a kernel with $K \times K$ elements:

quick sort $N^2 K^2 \log K$

Fast median filter

Let's filter an image with $N \times N$ pixels using a kernel with $K \times K$ elements:

quick sort $N^2K^2 \log K$

partial sort N^2K^2

Fast median filter

Let's filter an image with $N \times N$ pixels using a kernel with $K \times K$ elements:

quick sort $N^2K^2 \log K$

partial sort N^2K^2

- Huang et al. 1979: N^2K

Fast median filter

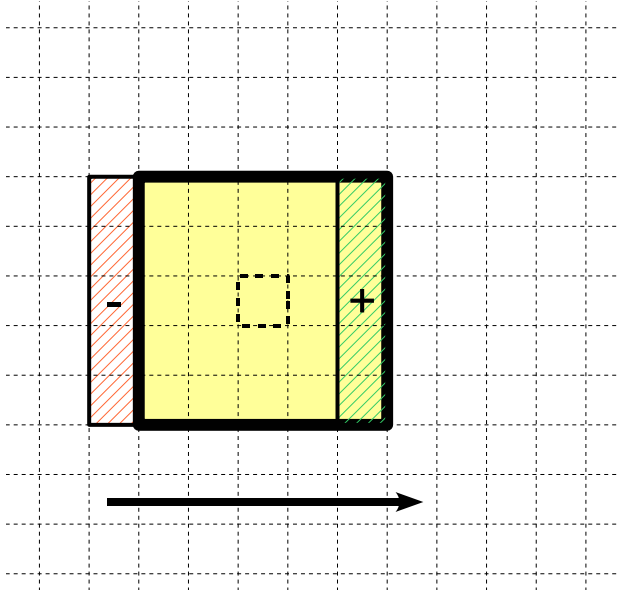
Let's filter an image with $N \times N$ pixels using a kernel with $K \times K$ elements:

quick sort $N^2K^2 \log K$

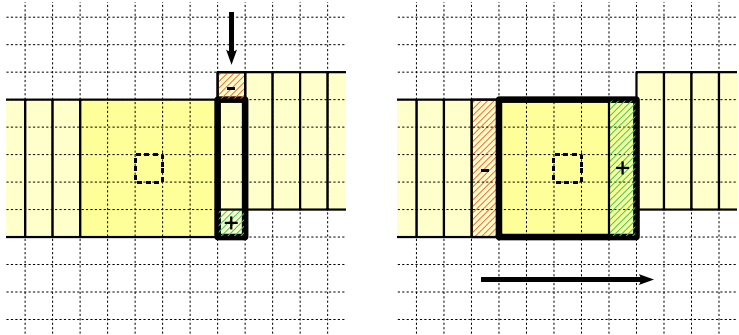
partial sort N^2K^2

- Huang et al. 1979: N^2K
- Perreault el al. 2007: N^2

Fast median filter (linear time)

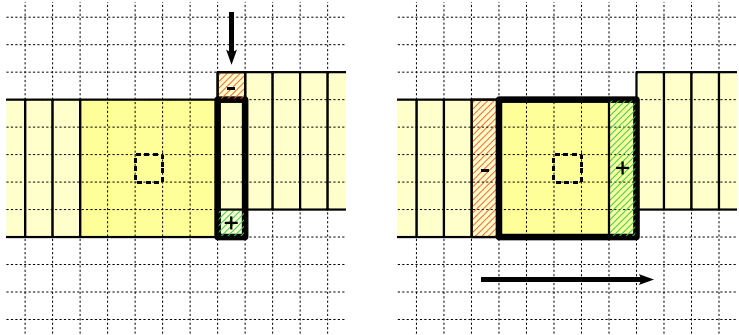


Fast median filter (constant time)



Perreault, Hebert. Median Filtering in Constant Time. 2007

Fast median filter (constant time)



- 16 bits for histogram bins, use vector operations
- split image into vertical bands for better using cache by histograms
- maintain two types of histograms: 16 and 256 bins

Perreault, Hebert. Median Filtering in Constant Time. 2007

Outline

1. What is image processing?
2. Tonal correction
3. Color correction
4. Noise reduction, convolution operation
5. Fast filtering
6. Edge detection

What is depicted on the image?

Canny edge detection algorihm

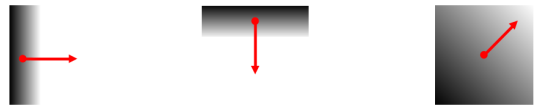
1. Blur image with gaussian filter
2. Compute gradient magnitude and direction
3. Make non-maximum suppression to make boundaries thin
4. Track edges using hysteresis

Image gradient

Gradient vector:
$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$

Gradient direction:
$\theta = \text{arctg}\left(\frac{\partial f}{\partial y} / \frac{\partial f}{\partial y}\right)$

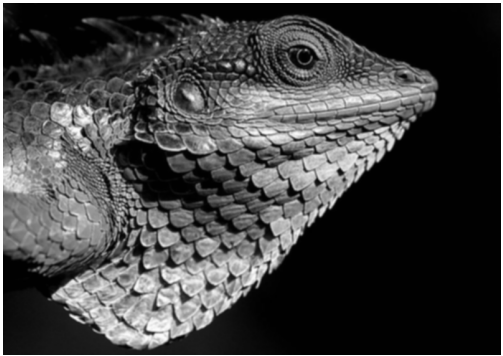
Gradient magnitude:
$\|\nabla f\| = \sqrt{\left(\frac{\partial f}{\partial y}\right)^2 + \left(\frac{\partial f}{\partial x}\right)^2}$

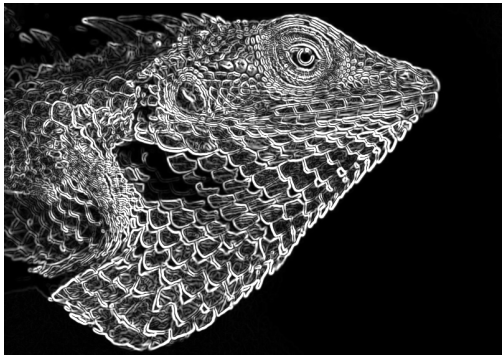


We will use Sobel 3×3 filters for computing gradient:

$$\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \quad \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

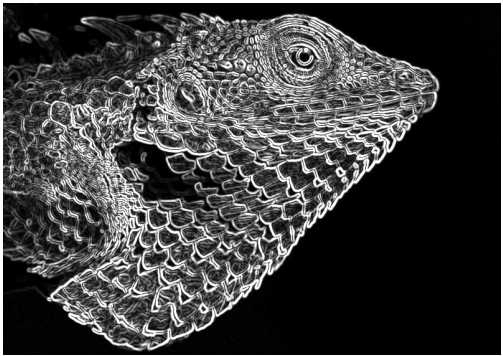
Image gradient

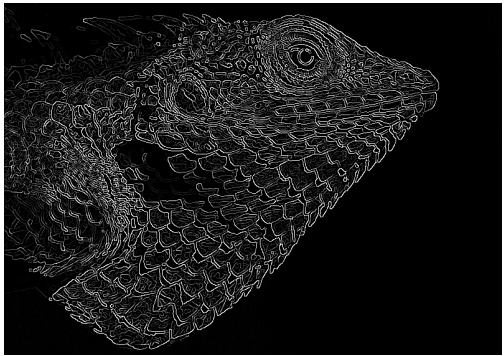




Non-maximum suppression

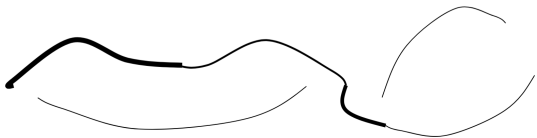
Use either 3×3 max filter or look at the gradient and antigradient neighbour pixels



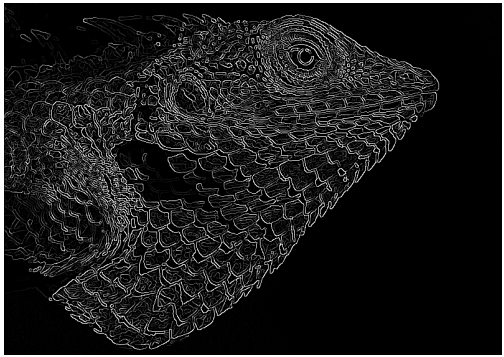


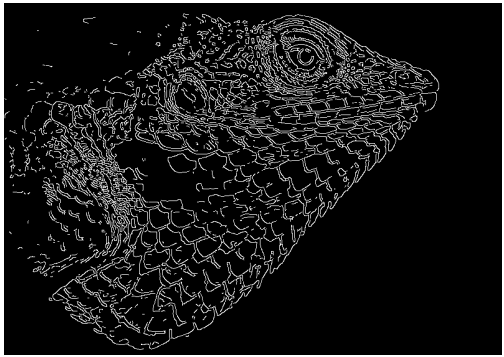
Track weak edges with hysteresis

Use two thresholds $t_{\text{strong}}, t_{\text{weak}}$ to divide into three categories: strong edges, weak edges and noise. Discard noise. Keep week edges that are connected to a strong edge



Track weak edges with hysteresis





Conclusion

We reviewed following topics:

- camera imaging pipeline
- tone and color correction
- linear filtering for simplest tasks: noise suppression, gradient estimation, image sharpening
- median filter for suppressing impulse noise
- edge detection using Canny method