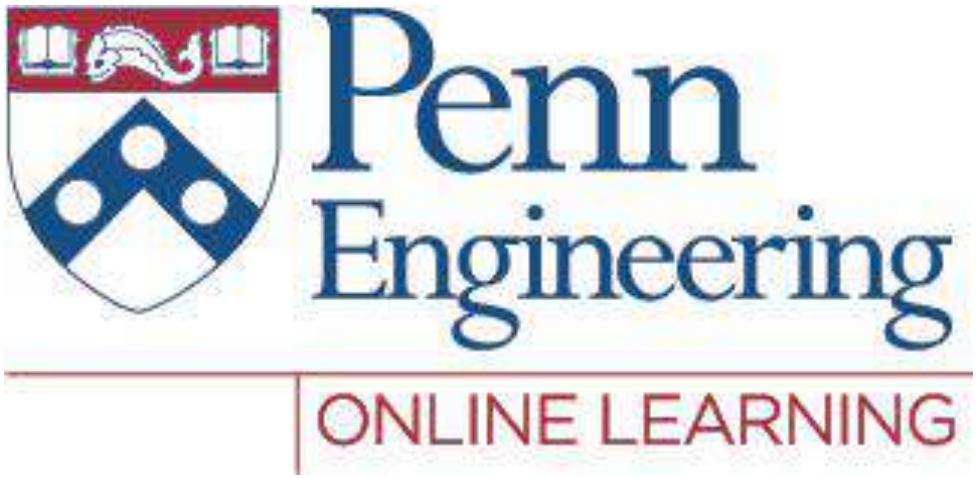
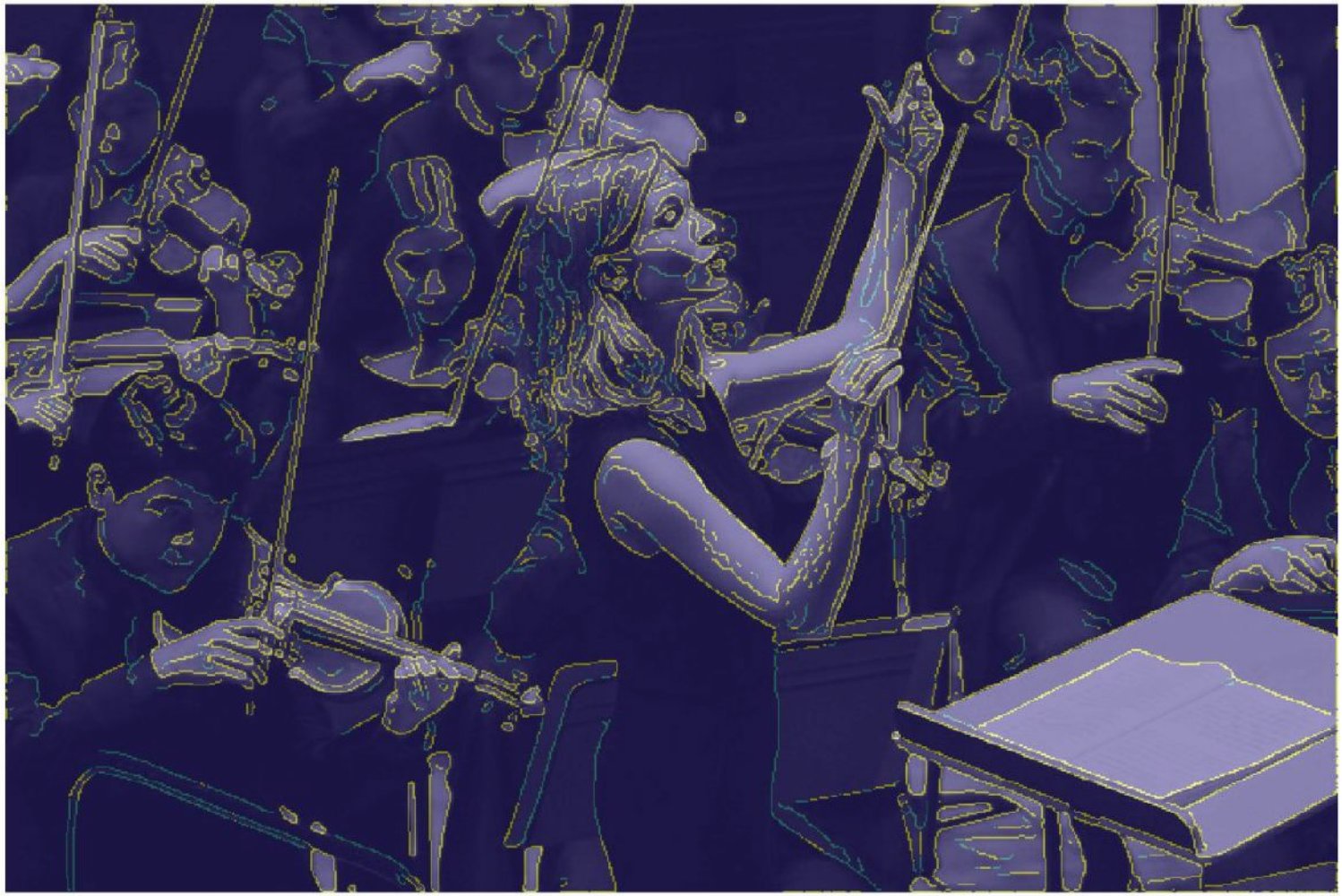
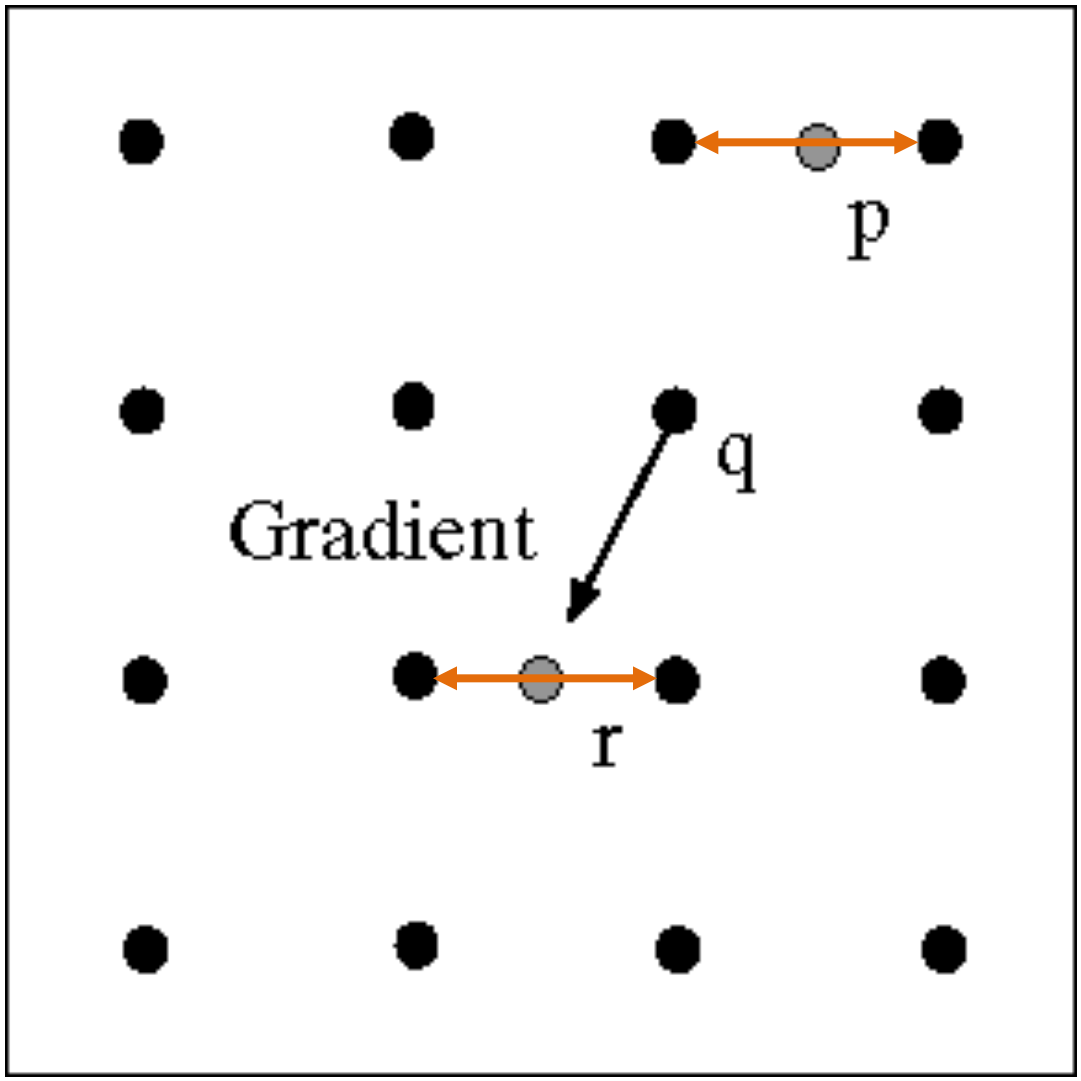


Video 3.1
Jianbo Shi



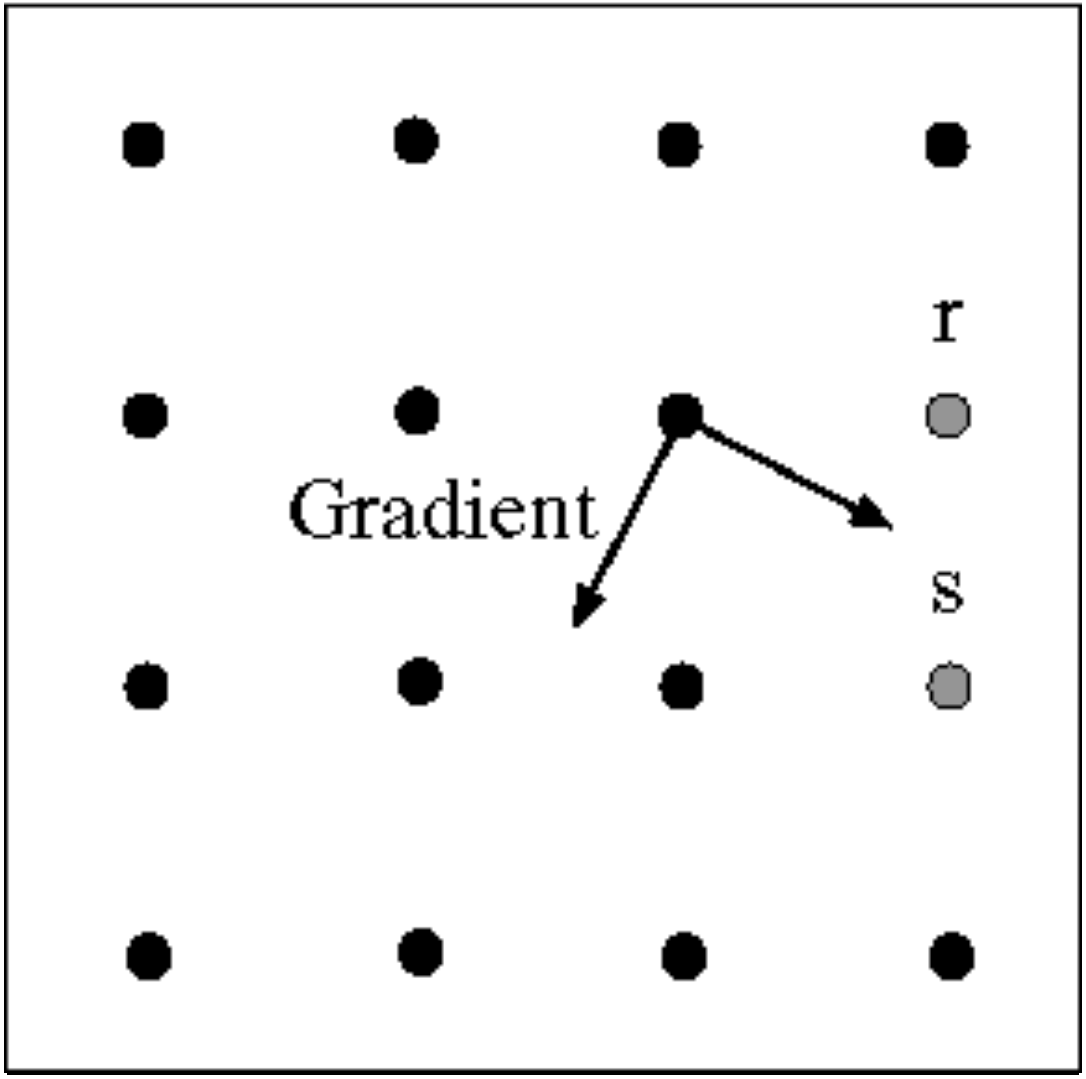
No intensity values at r and p:
Interpolate these intensities using neighbor pixels.

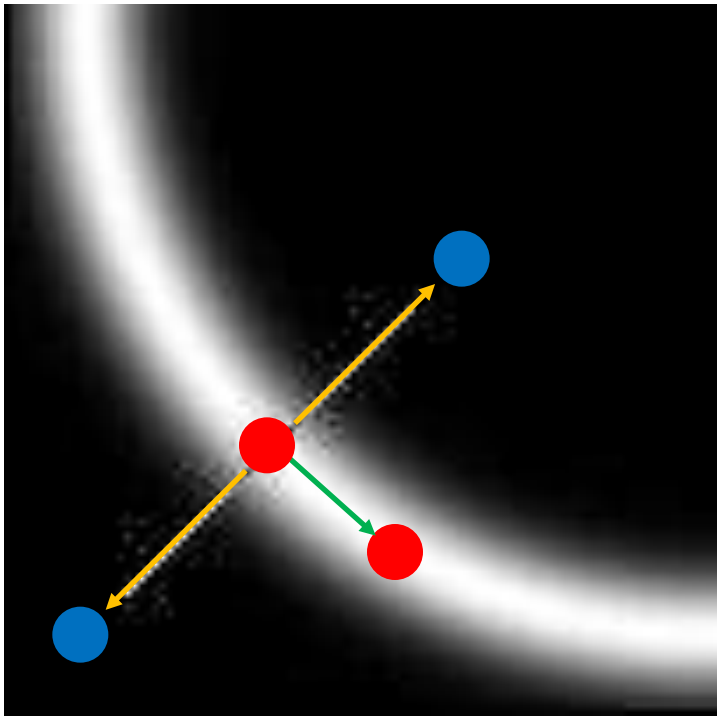


Where is next edge point?

Where is next edge point?

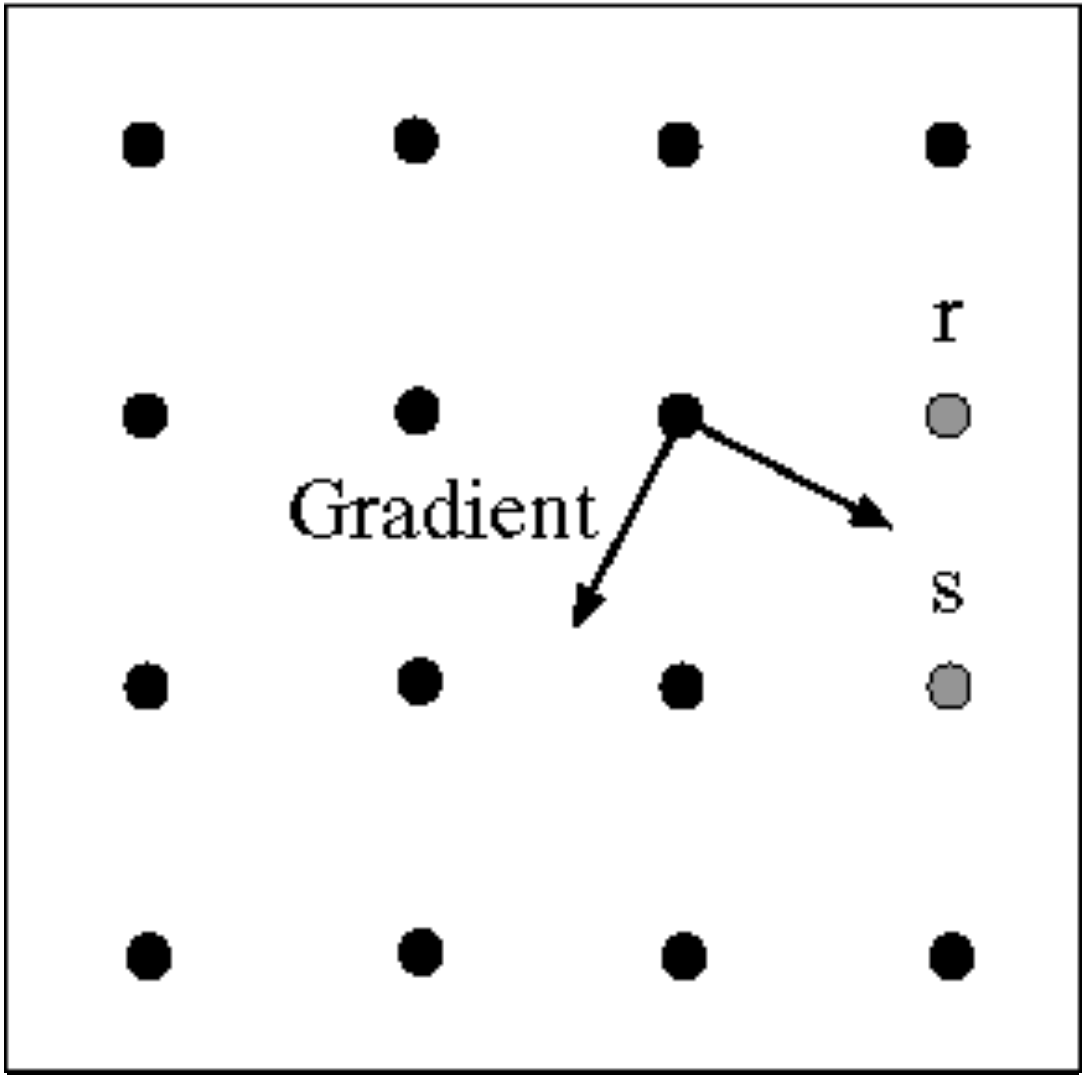
we construct the tangent to the edge curve (which is normal to the gradient at that point) and use this to predict the next points

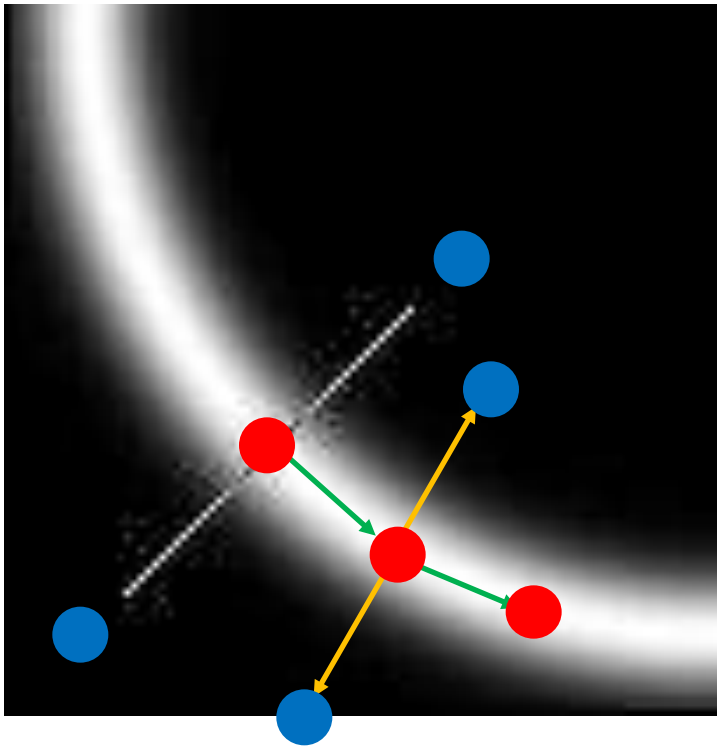




Where is next edge point?

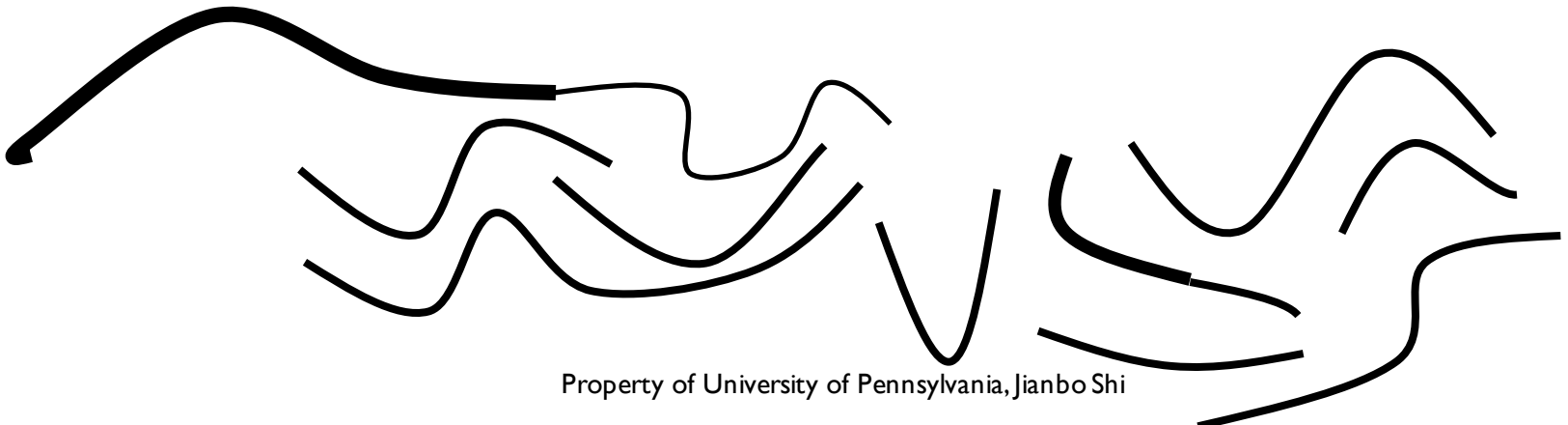
we construct the tangent to the edge curve (which is normal to the gradient at that point) and use this to predict the next points





Edge Linking: Hysteresis

- Check that maximum value of gradient value is sufficiently large
  - drop-outs? use **hysteresis**
    - use a high threshold to start edge curves and a low threshold to continue them.



Edge Linking: Hysteresis

- Check that maximum value of gradient value is sufficiently large
  - drop-outs? use **hysteresis**
    - use a high threshold to start edge curves and a low threshold to continue them.

threshold_high

0	1	0	0
0	1	0	1
0	0	0	0
0	1	0	0

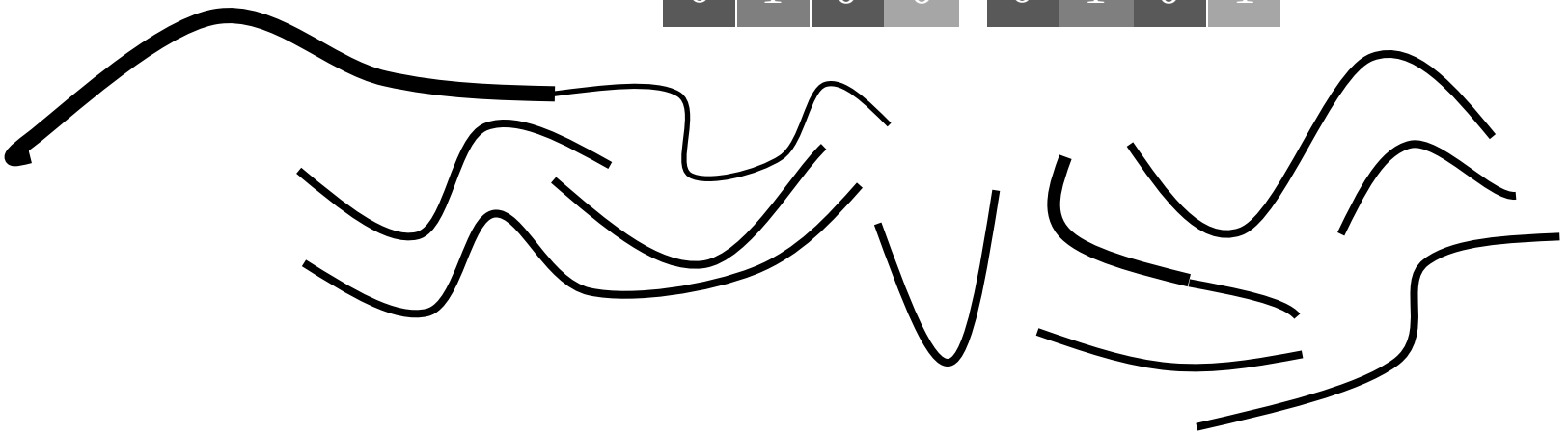
Edge Linking: Hysteresis

threshold_high

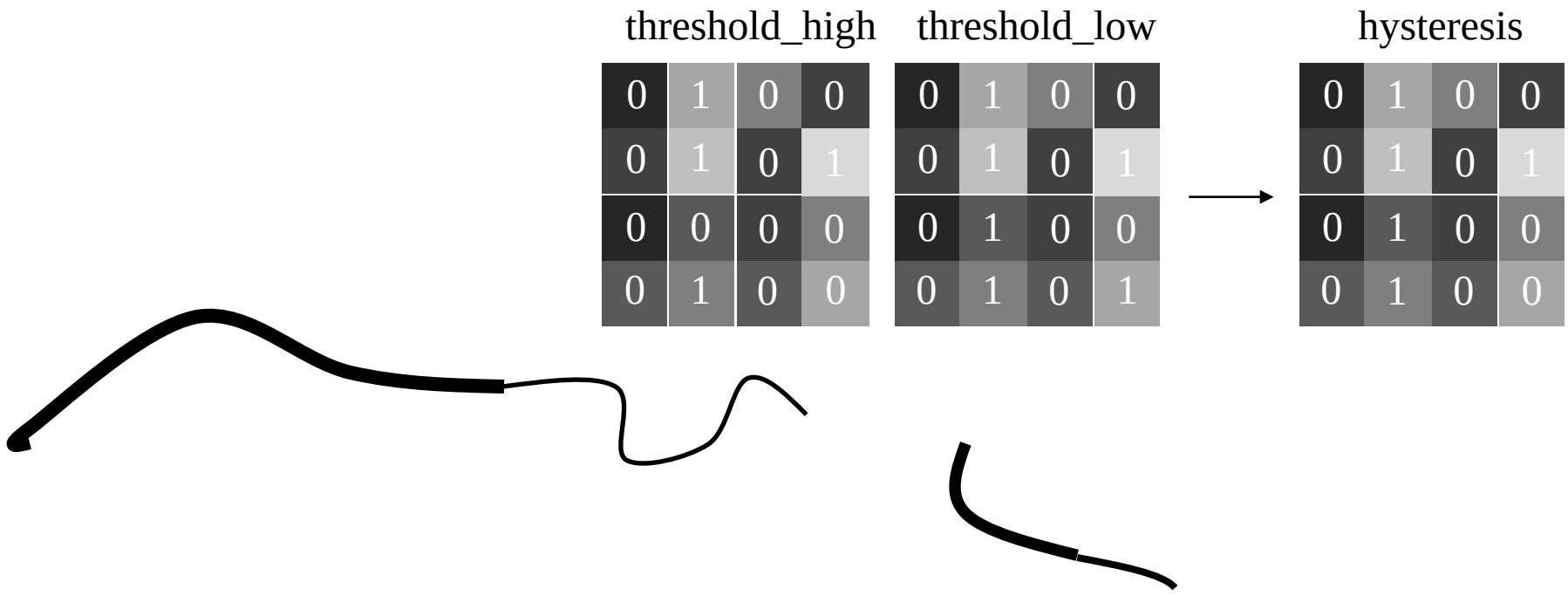
0	1	0	0
0	1	0	1
0	0	0	0
0	1	0	0

threshold_low

0	1	0	0
0	1	0	1
0	1	0	0
0	1	0	1

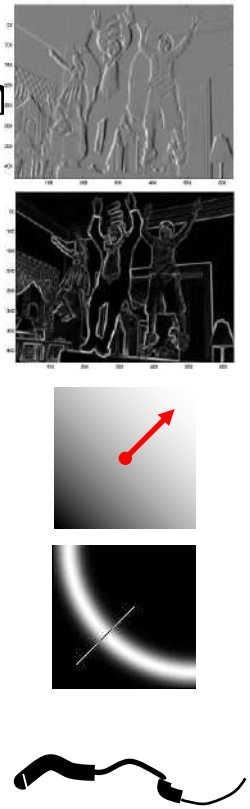


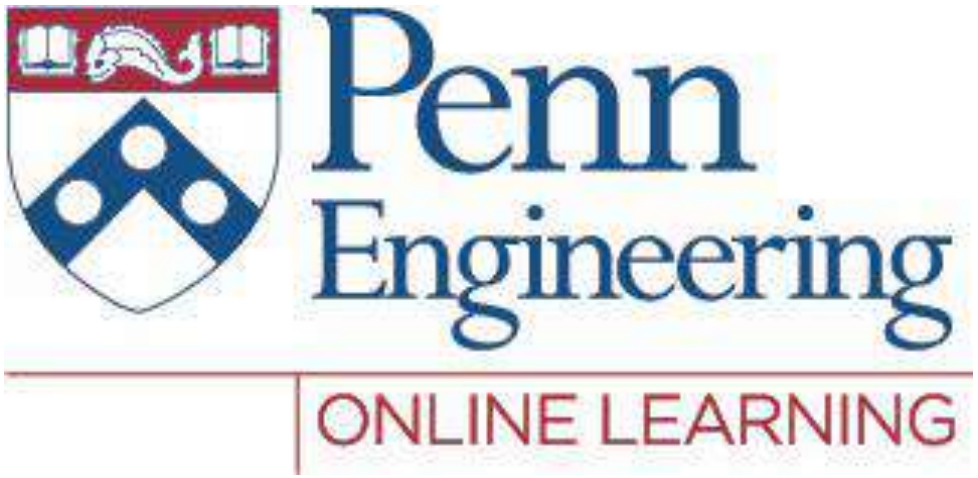
Edge Linking: Hysteresis



Canny Edge Detection

1. Filter image by derivatives of Gaussian
2. Compute magnitude of gradient
3. Compute edge orientation
4. Detect local maximum
5. Edge linking

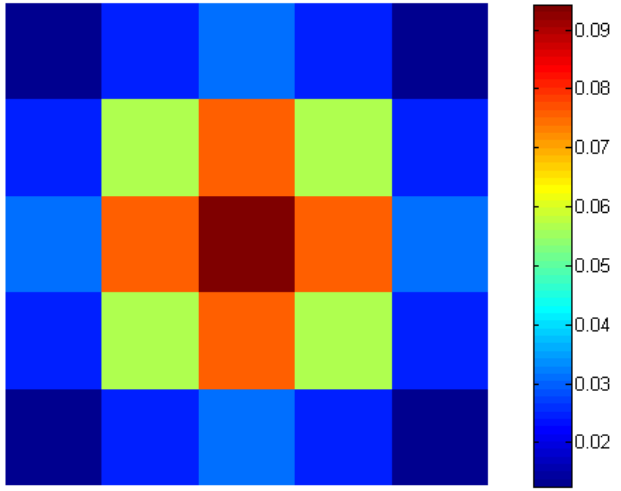




Video 3.2
Jianbo Shi

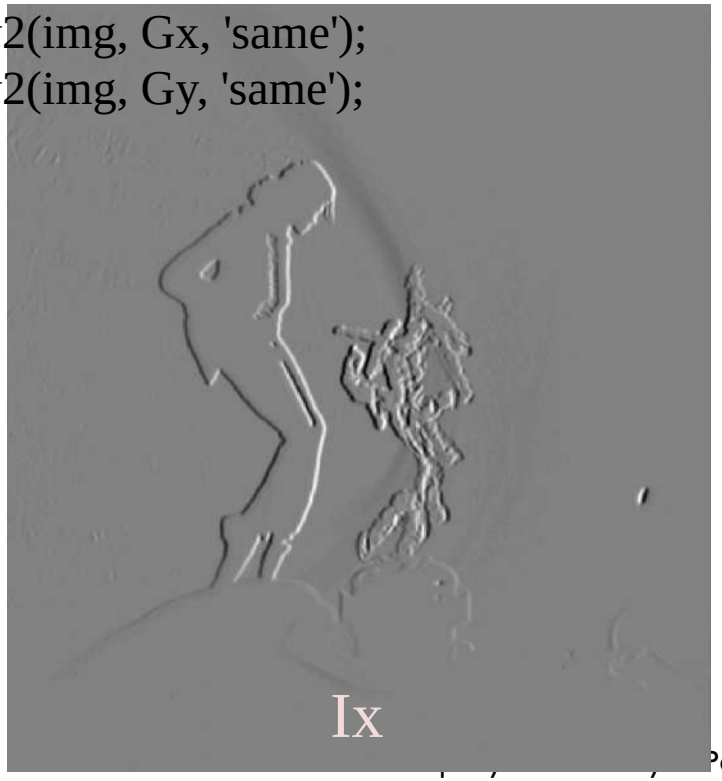
Canny Edge Implementation

```
img = imread (‘image.png');
img = rgb2gray(img);
img = double (img);

% Value for high and low thresholding
threshold_low = 0.035;
threshold_high = 0.175;

%% Gaussian filter definition (https://en.wikipedia.org/wiki/Canny_edge_detector)
G = [2, 4, 5, 4, 2; 4, 9, 12, 9, 4;5, 12, 15, 12, 5;4, 9, 12, 9, 4;2, 4, 5, 4, 2];
G = 1/159.* G;

%Filter for horizontal and vertical direction
dx = [1 -1];
dy = [1; -1];
```

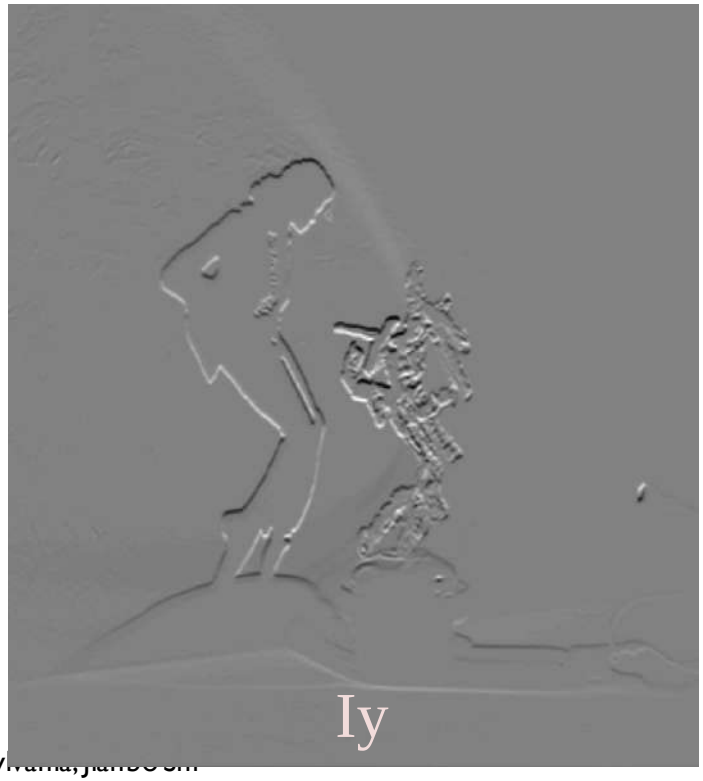


Canny Edge Implementation

```
% % Convolution of image with
Gaussian
Gx = conv2(G, dx, 'same');
Gy = conv2(G, dy, 'same');

% Convolution of image with Gx and
Gy
Ix = conv2(img, Gx, 'same');
Iy = conv2(img, Gy, 'same');
```



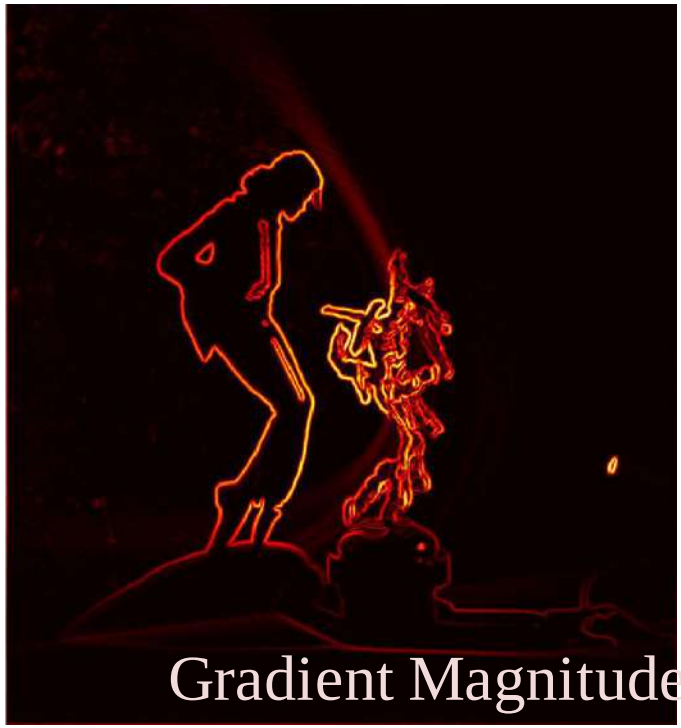
Ix

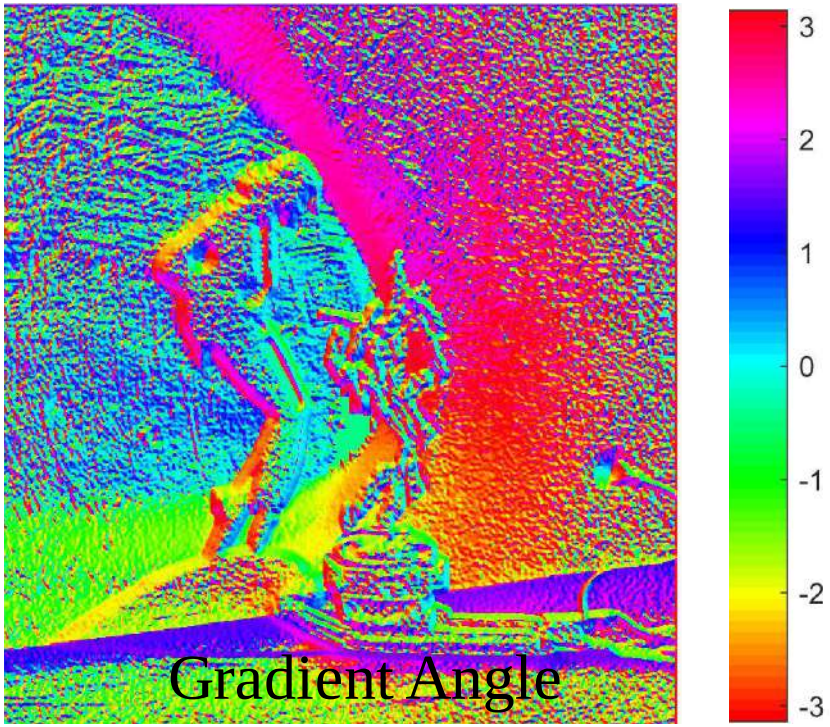


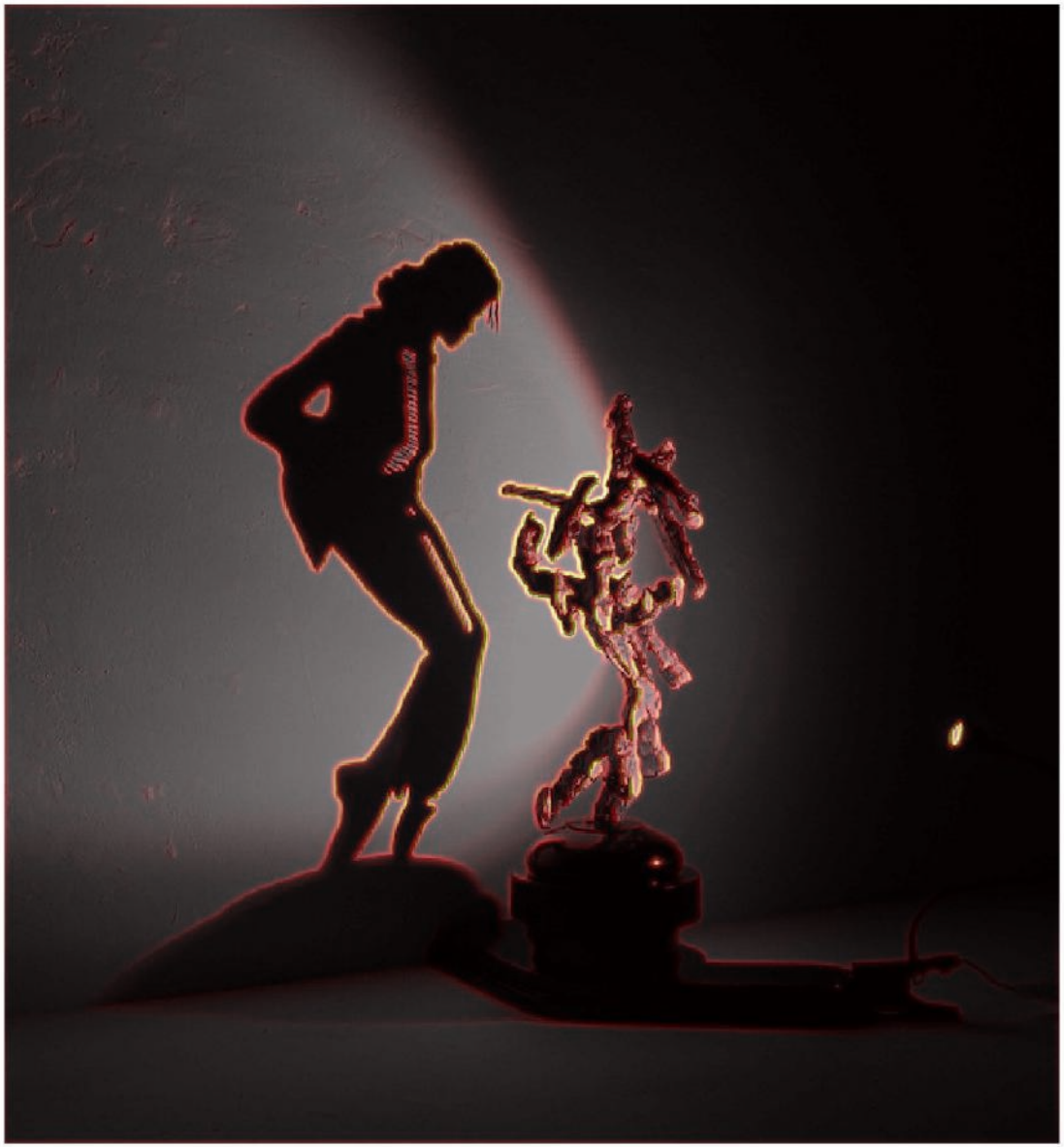
Iy

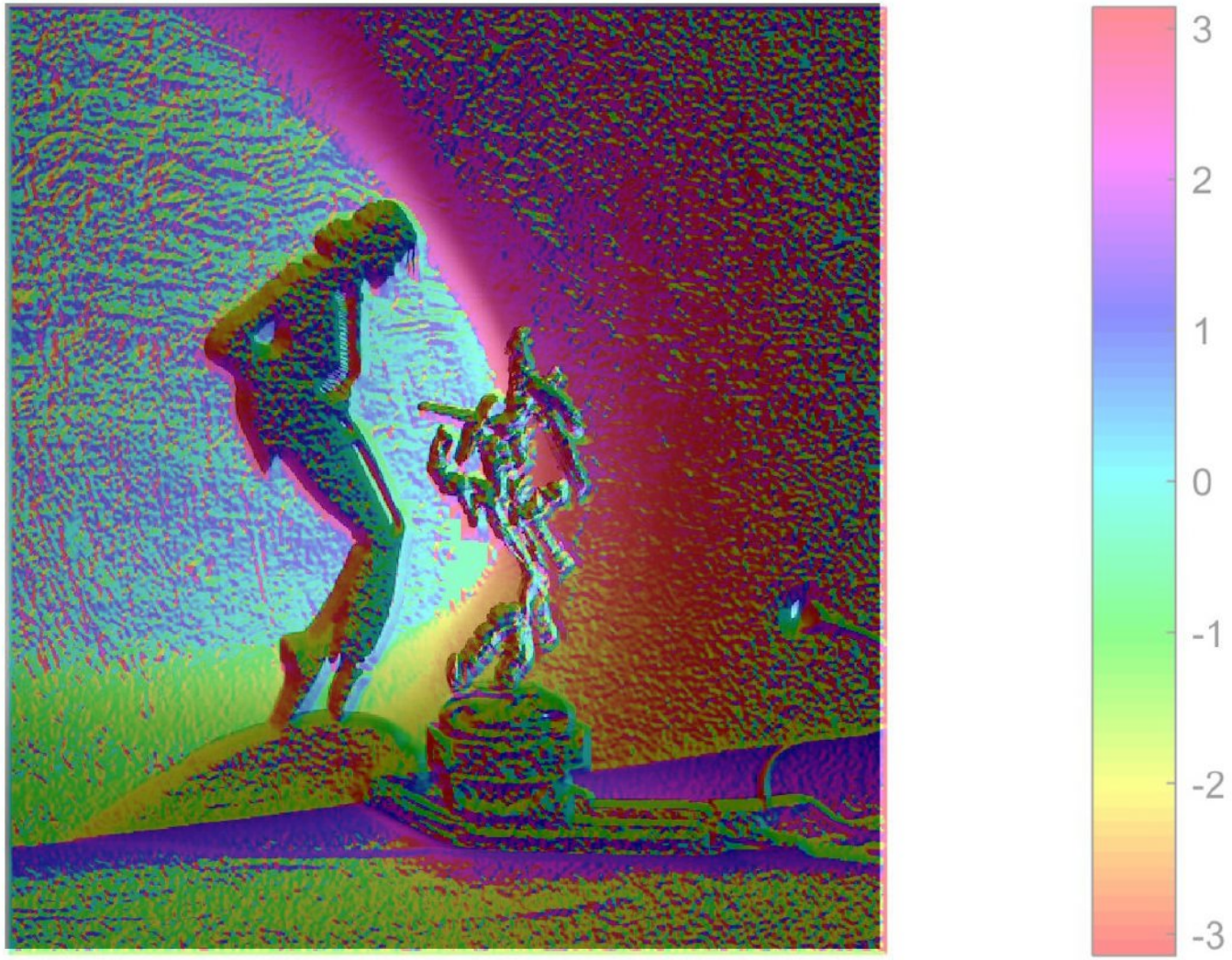
Canny Edge Implementation

angle = atan2(Iy, Ix);
mag = sqrt(Iy.^2 + Ix.^2);





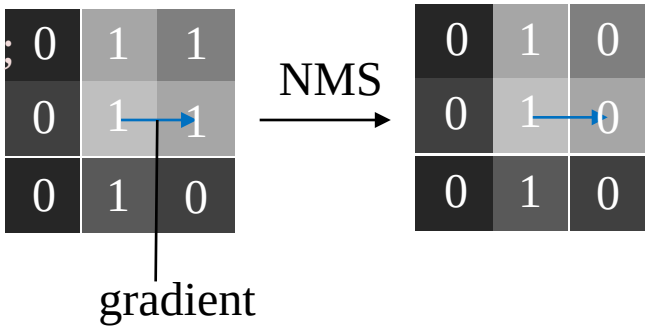


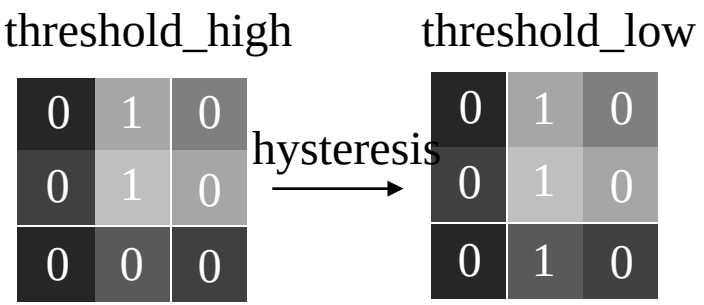


Canny Edge Implementation

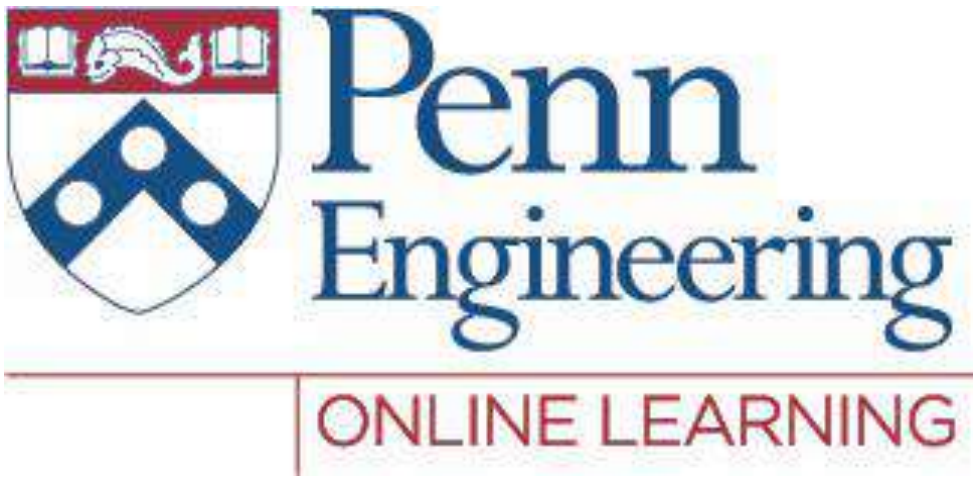
%% Non-Maximum Supression
edge = non_maximum_suppression(magnitude, angle, edge);

low = threshold_low * max(edge(:));
high = threshold_high * max(edge(:));
linked_edge = hysteresis_thresholding(low, high);



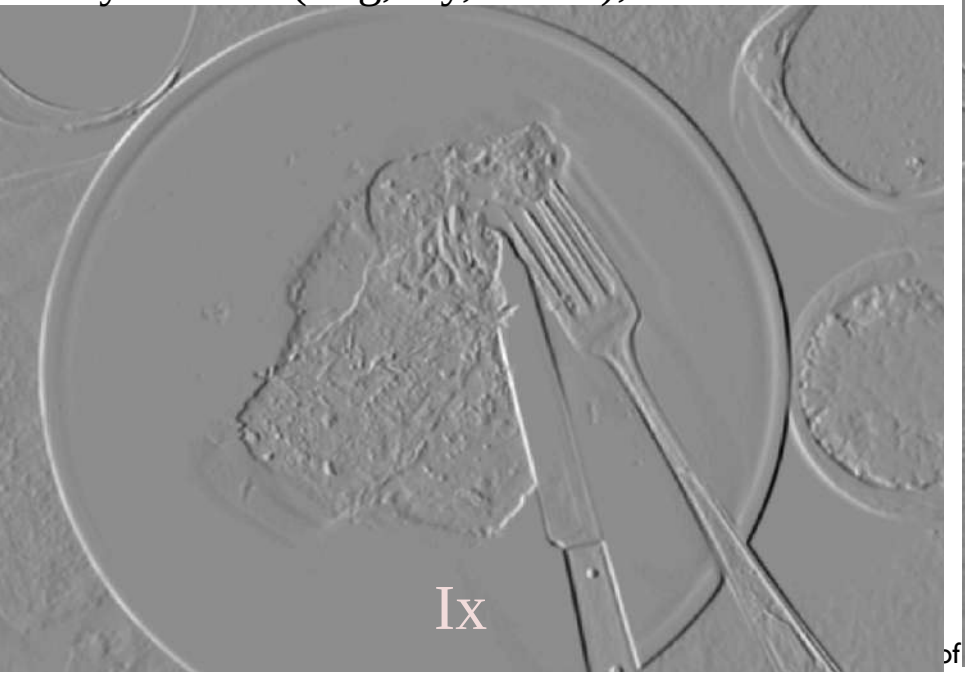


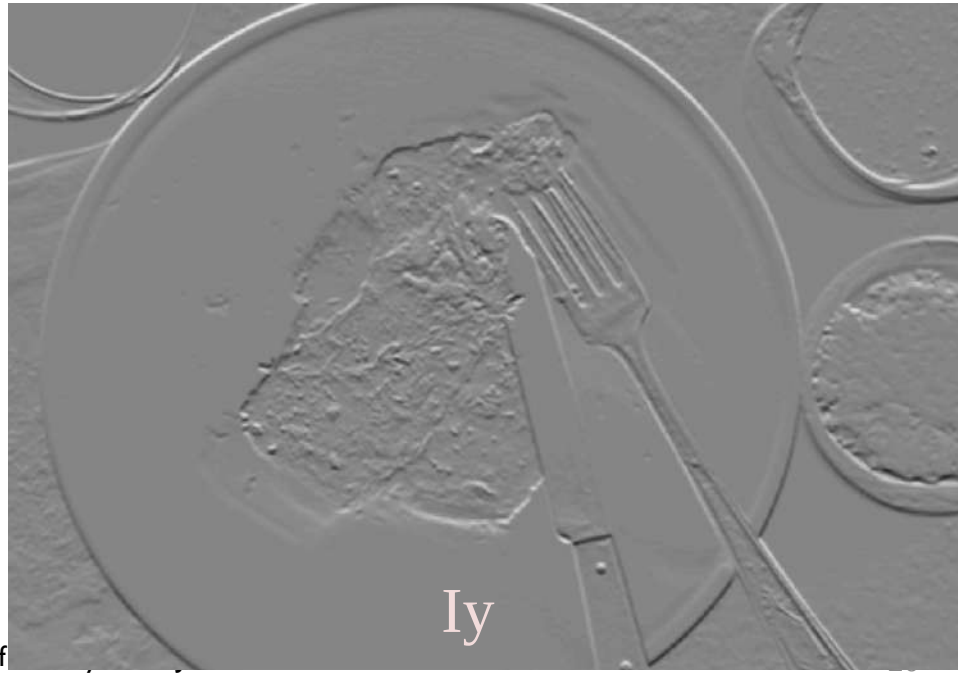
Localized edge



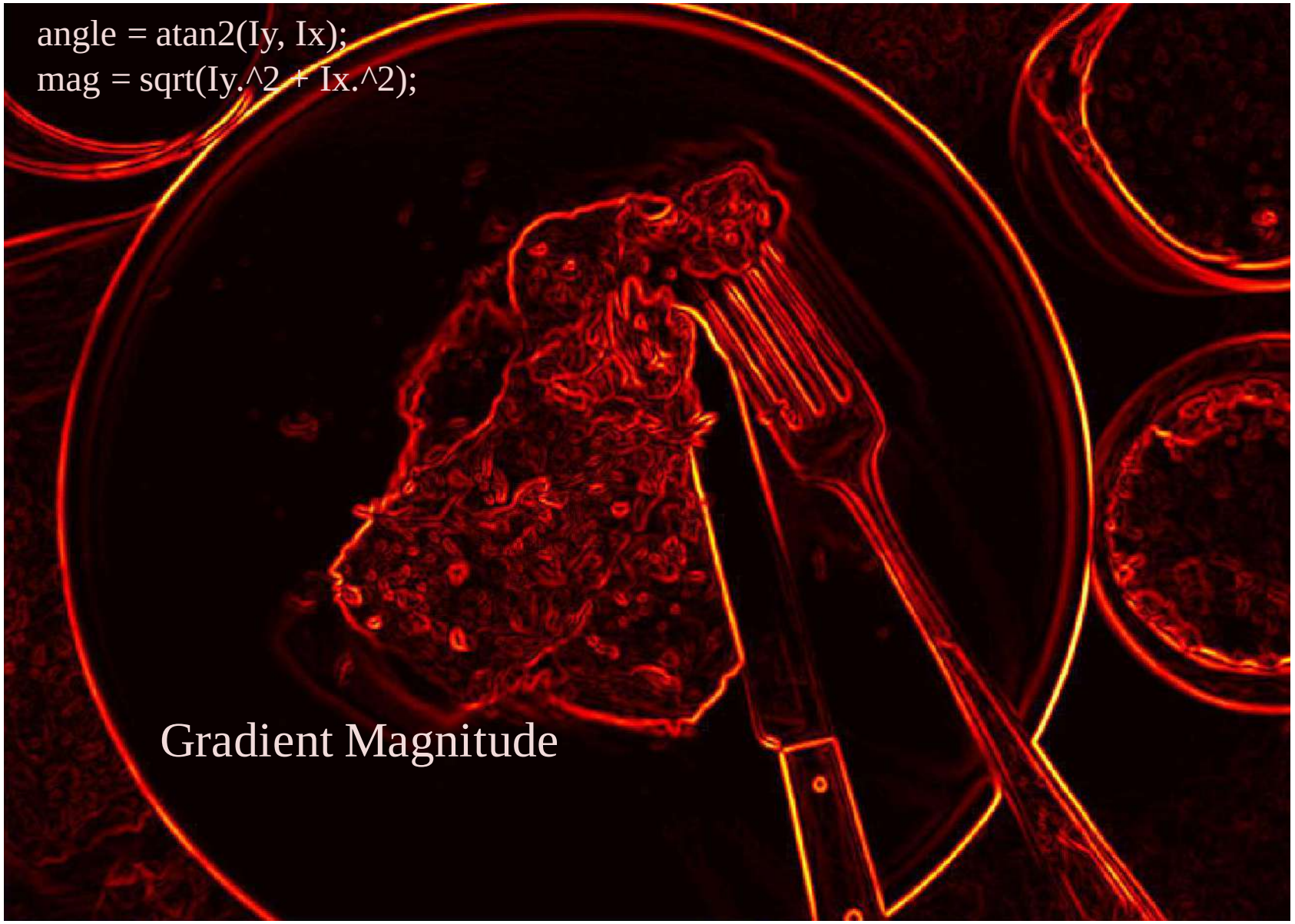
Video 3.3
Jianbo Shi

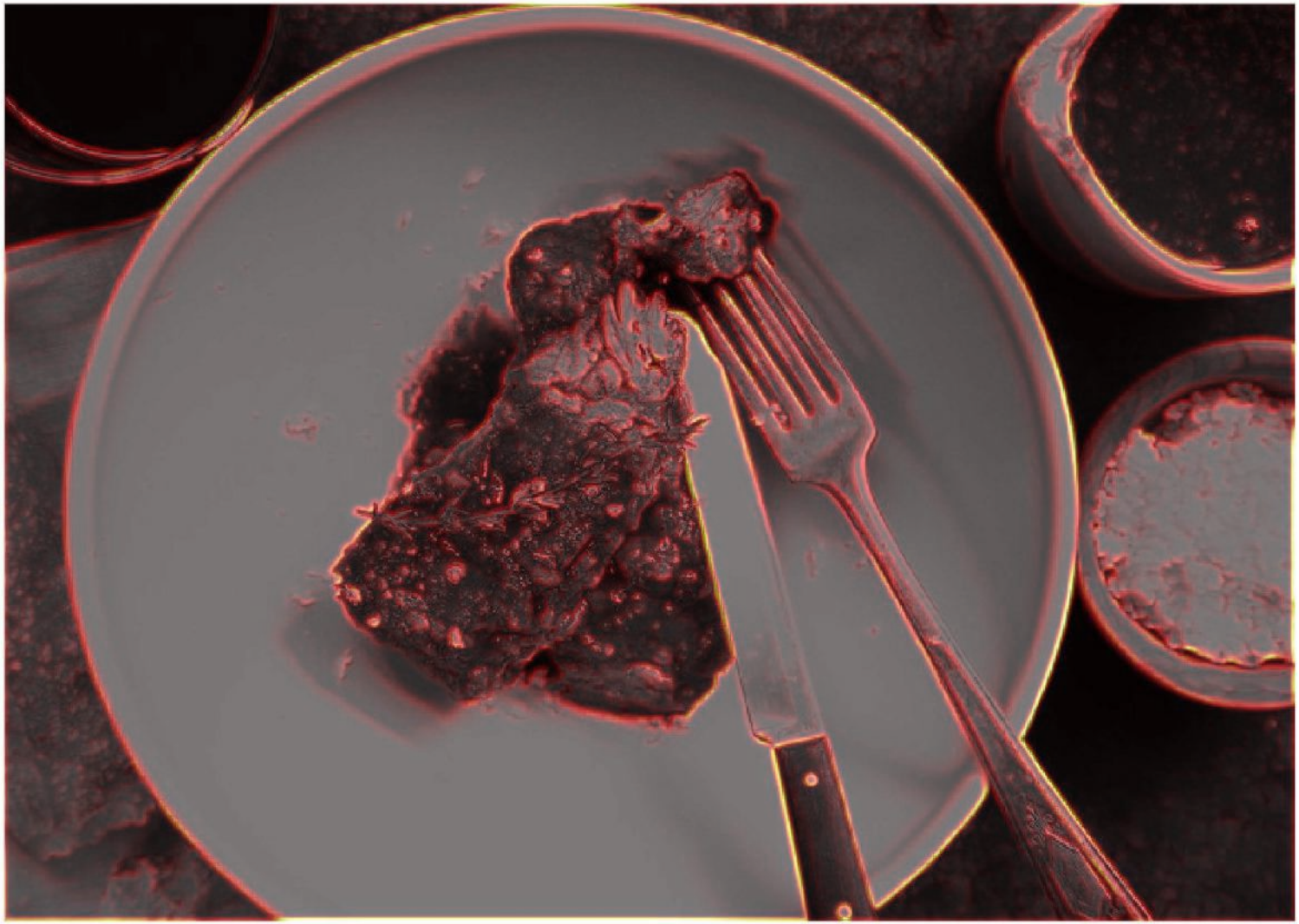
```
% % Convolution of image with
Gaussian
Gx = conv2(G, dx, 'same');
Gy = conv2(G, dy, 'same');

% Convolution of image with Gx and
Gy
Ix = conv2(img, Gx, 'same');
Iy = conv2(img, Gy, 'same');
```





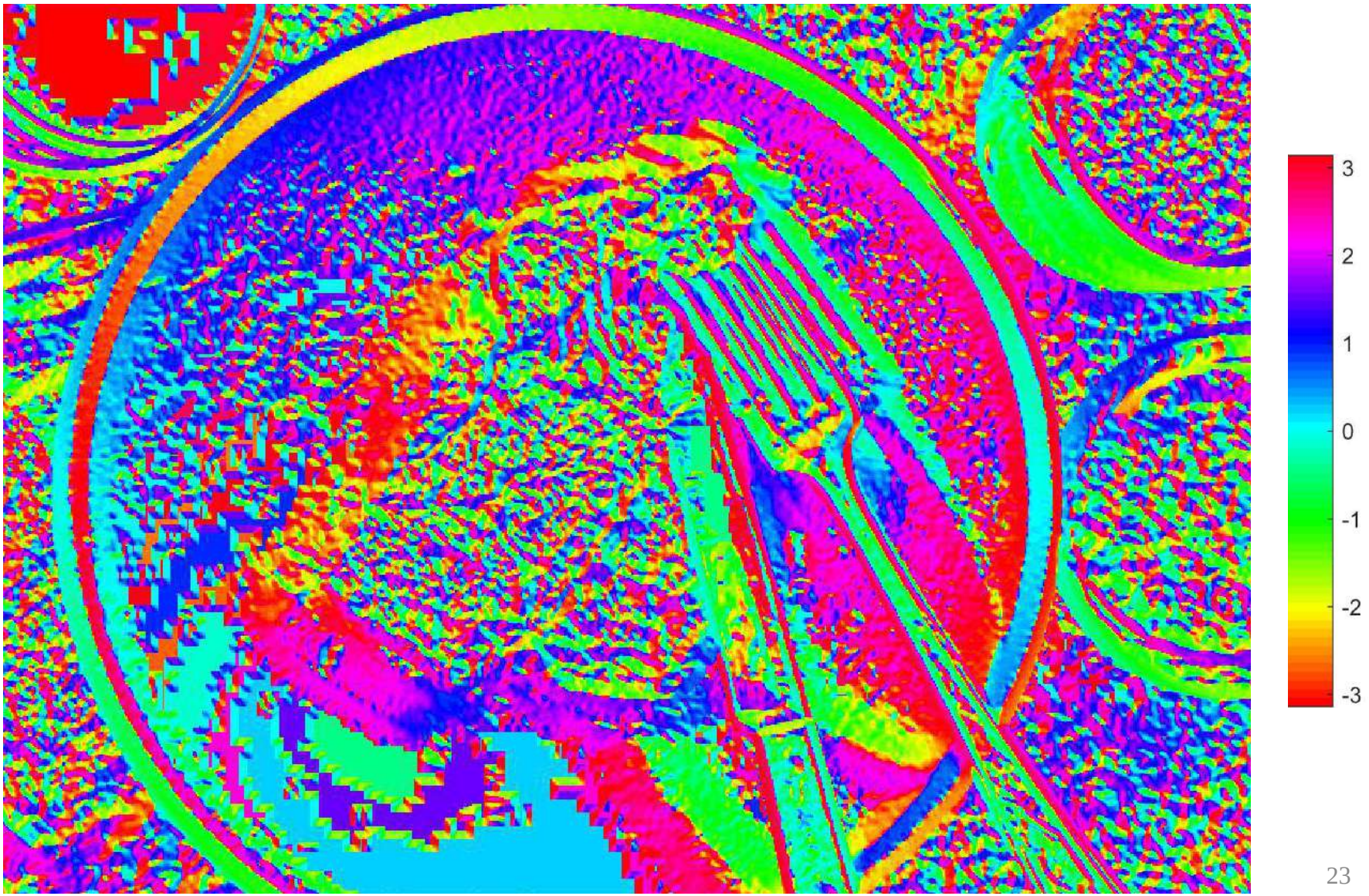
Canny Edge Implementation

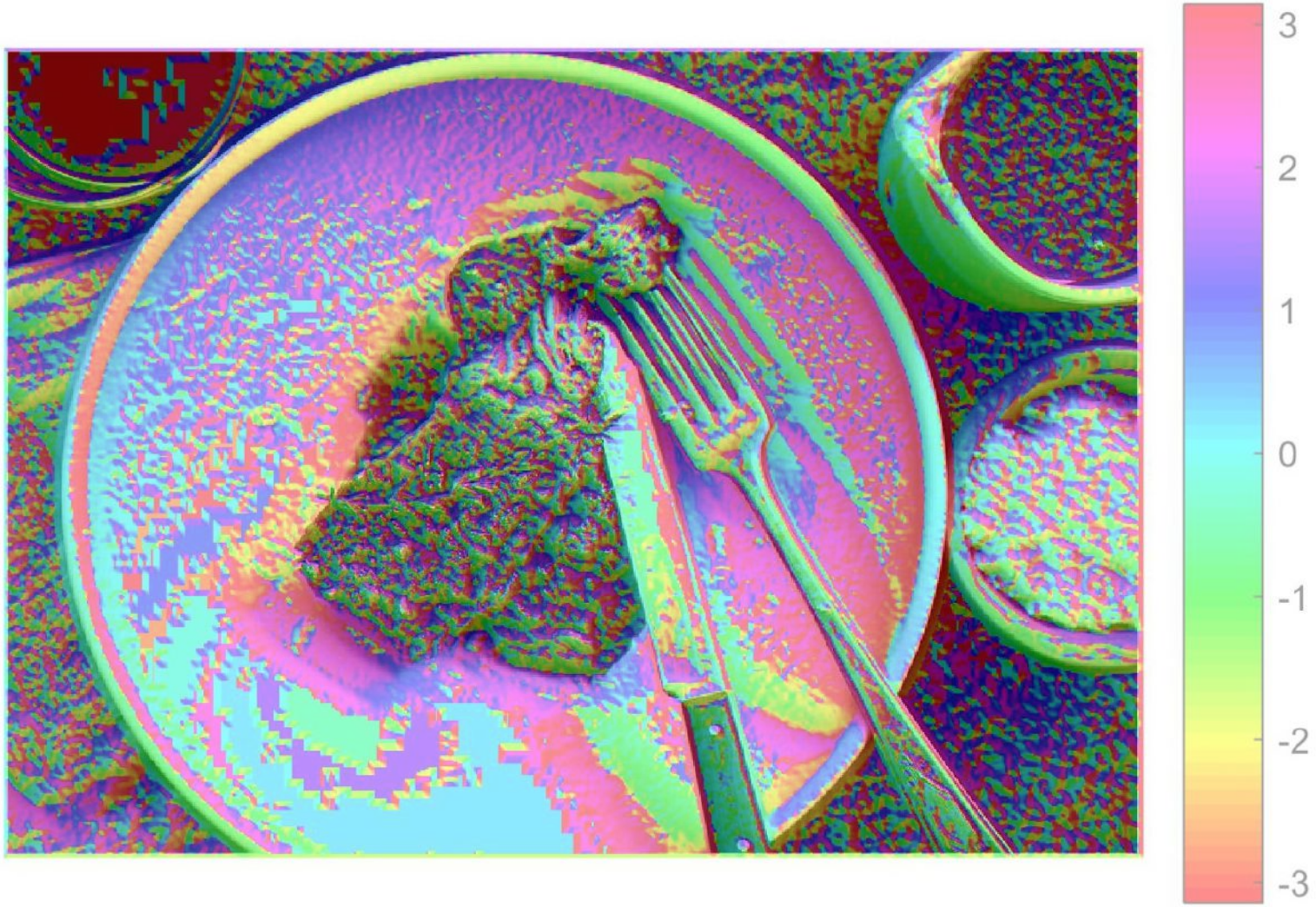




Canny Edge Implementation

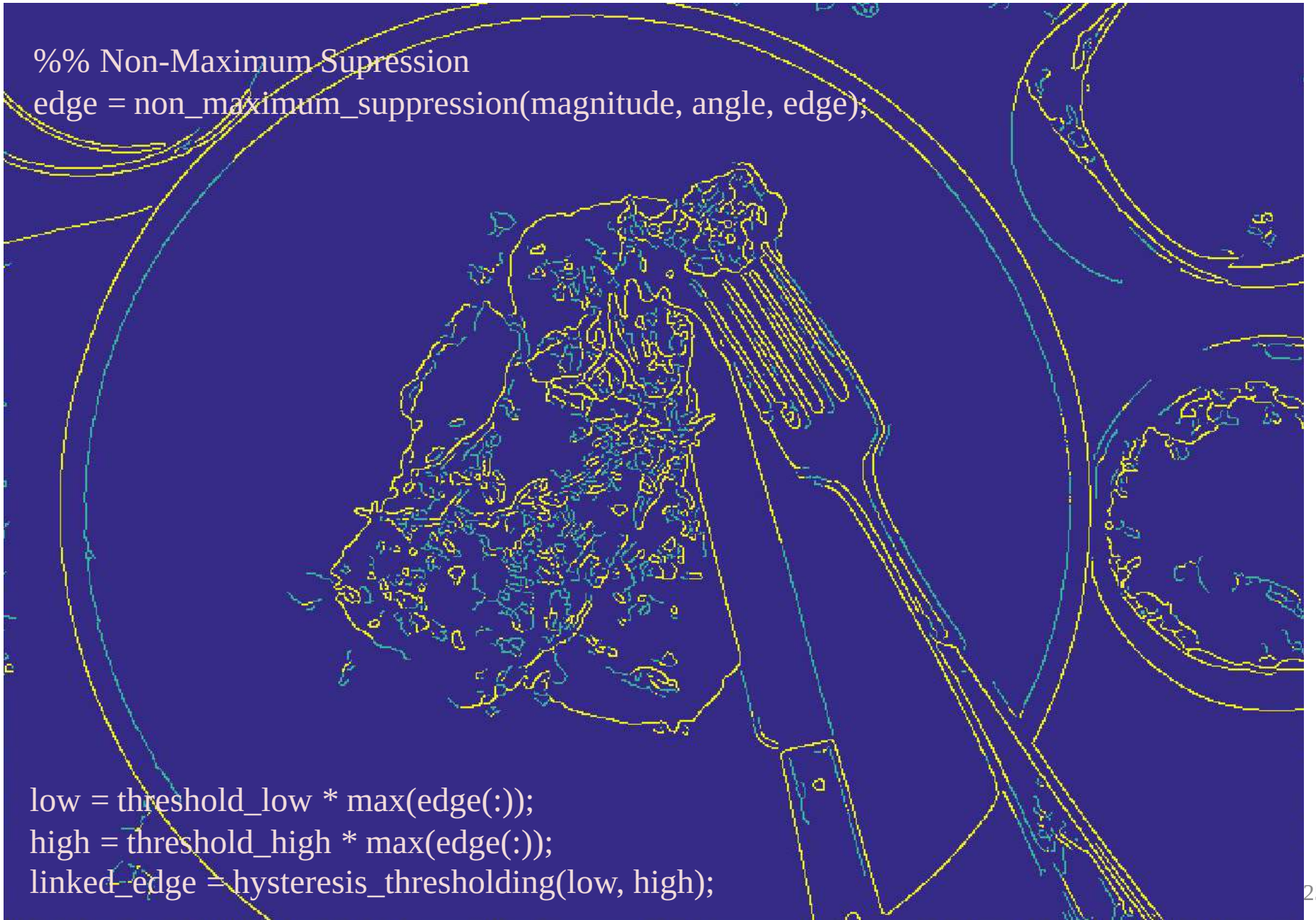
```
angle = atan2(Iy, Ix);
mag = sqrt(Iy.^2 + Ix.^2);
```

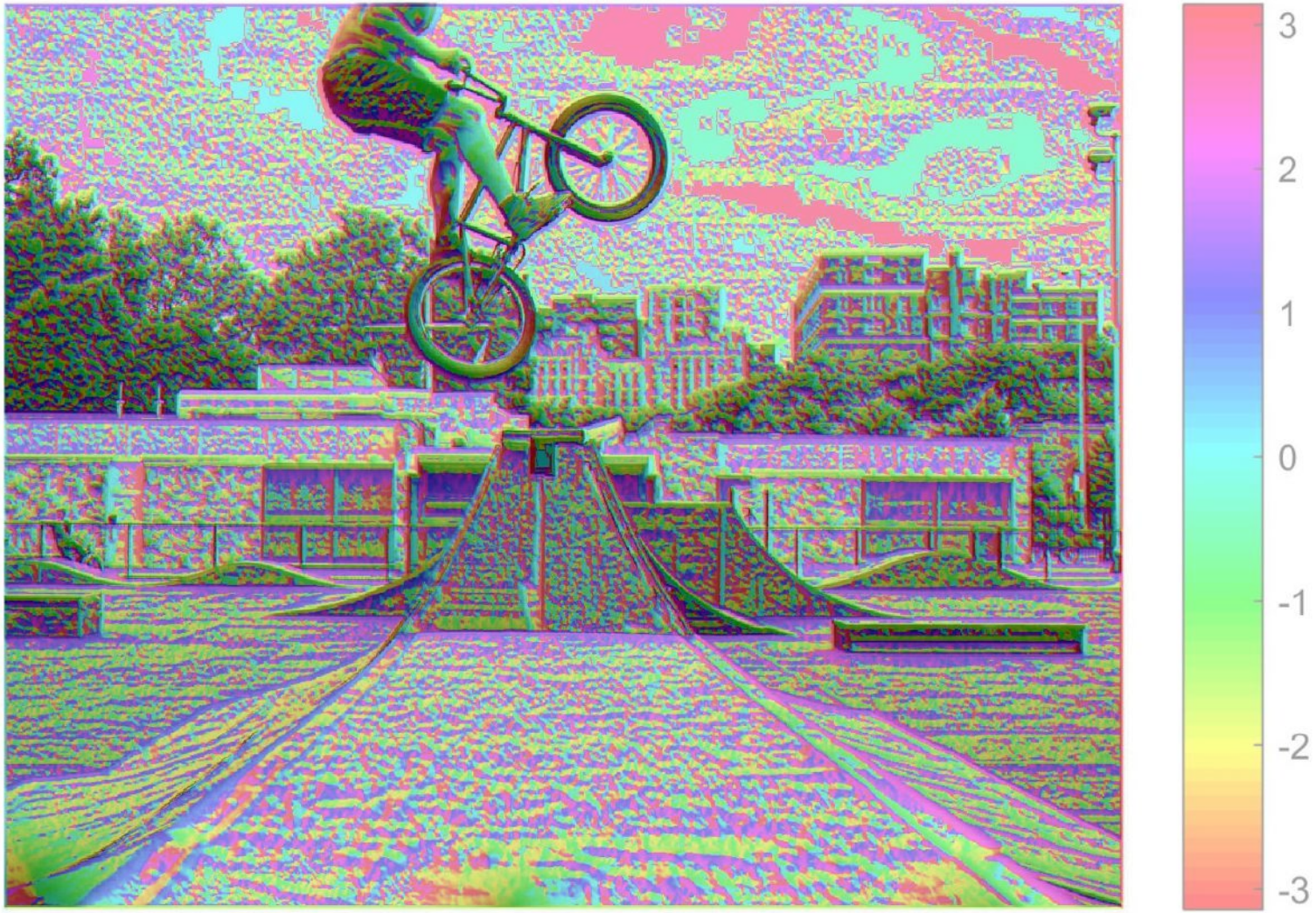


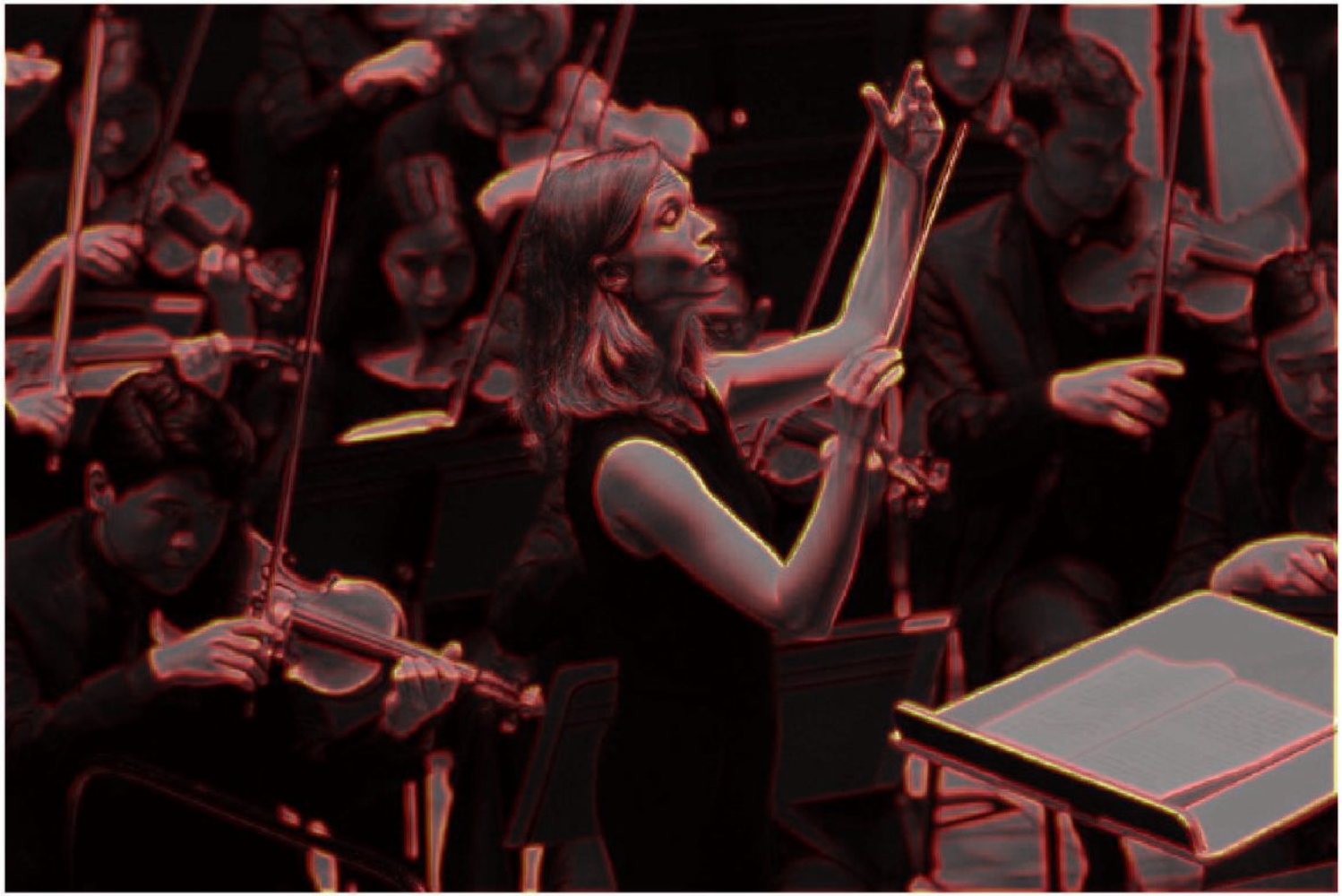


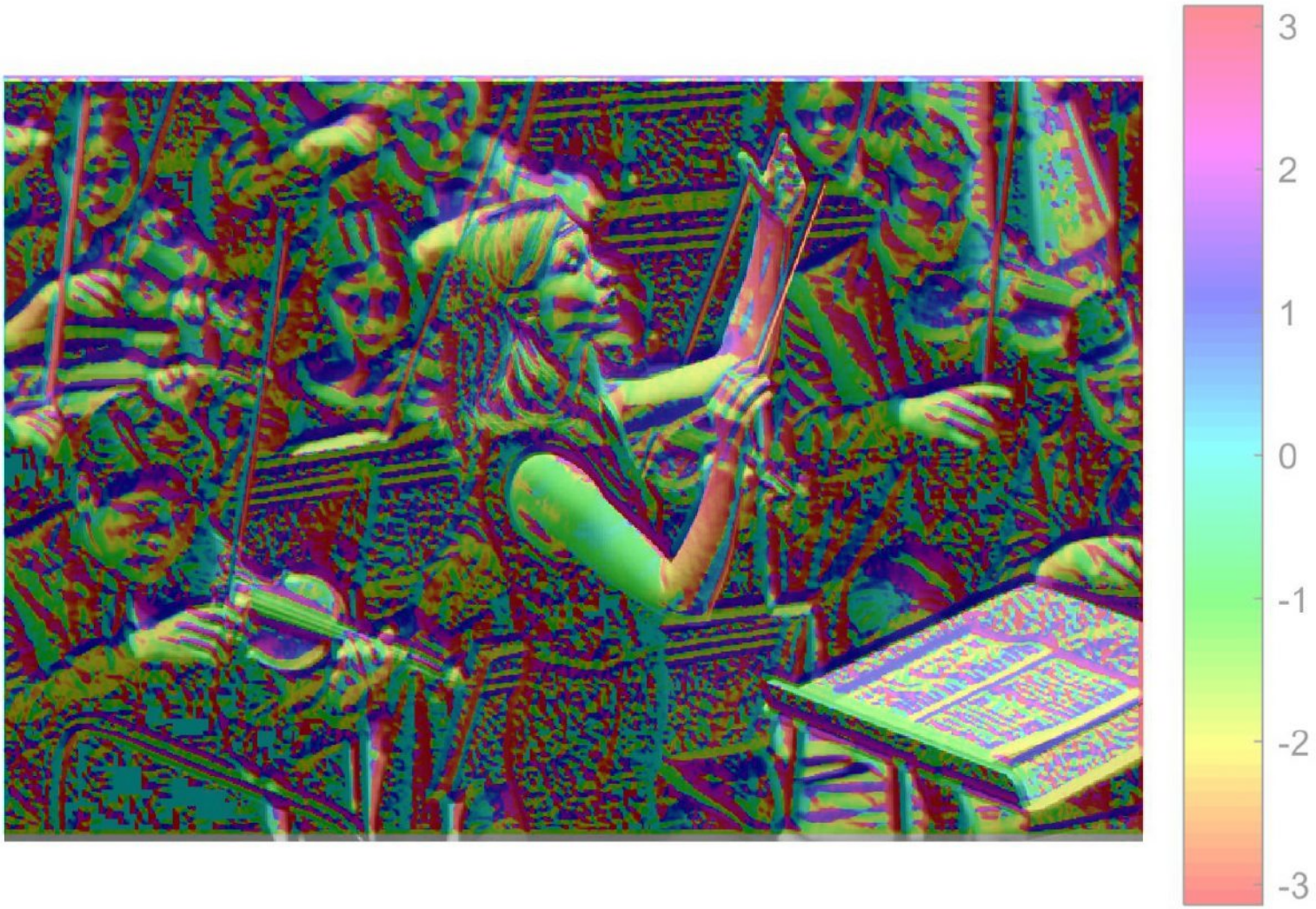
Canny Edge Implementation

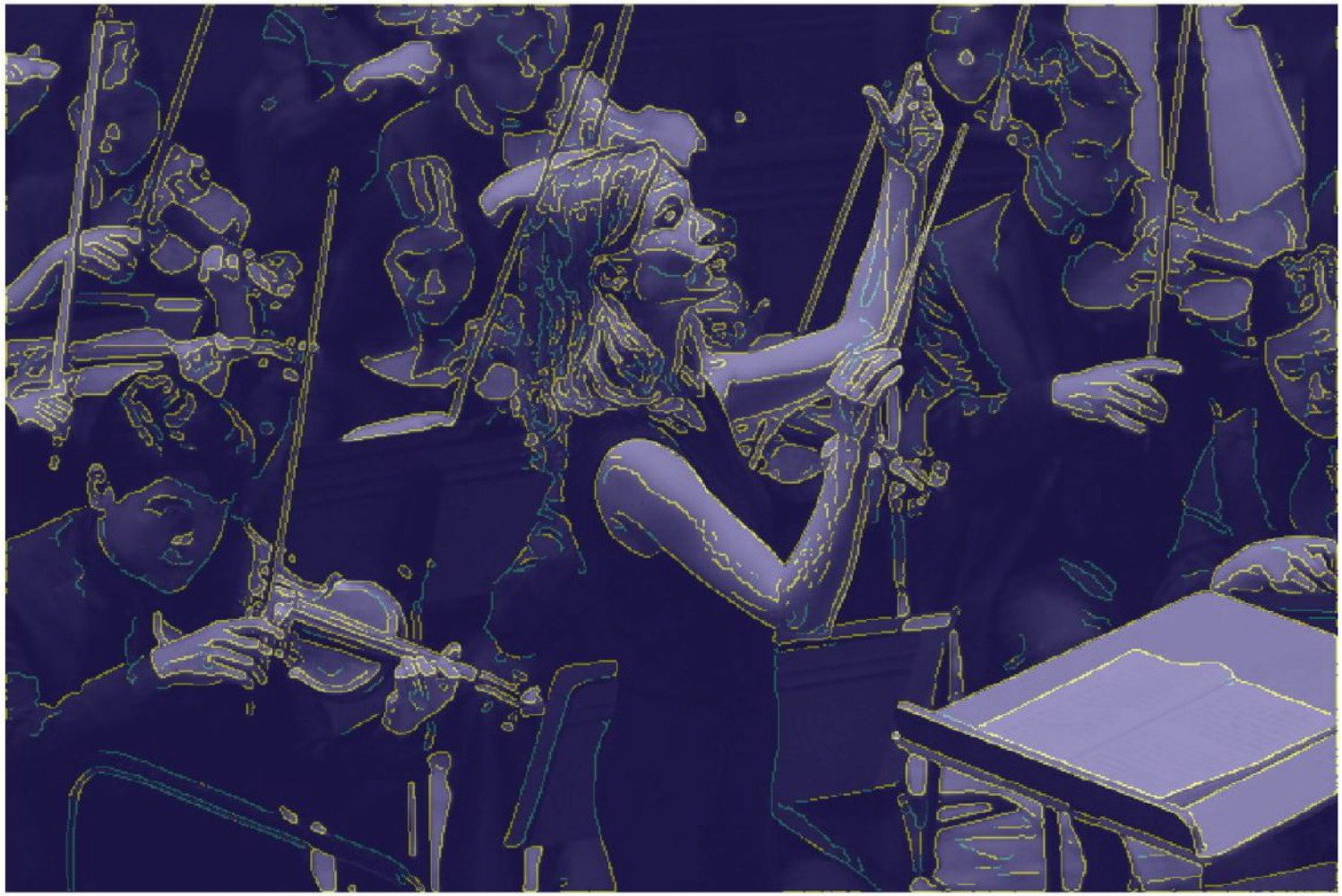
Localized edge

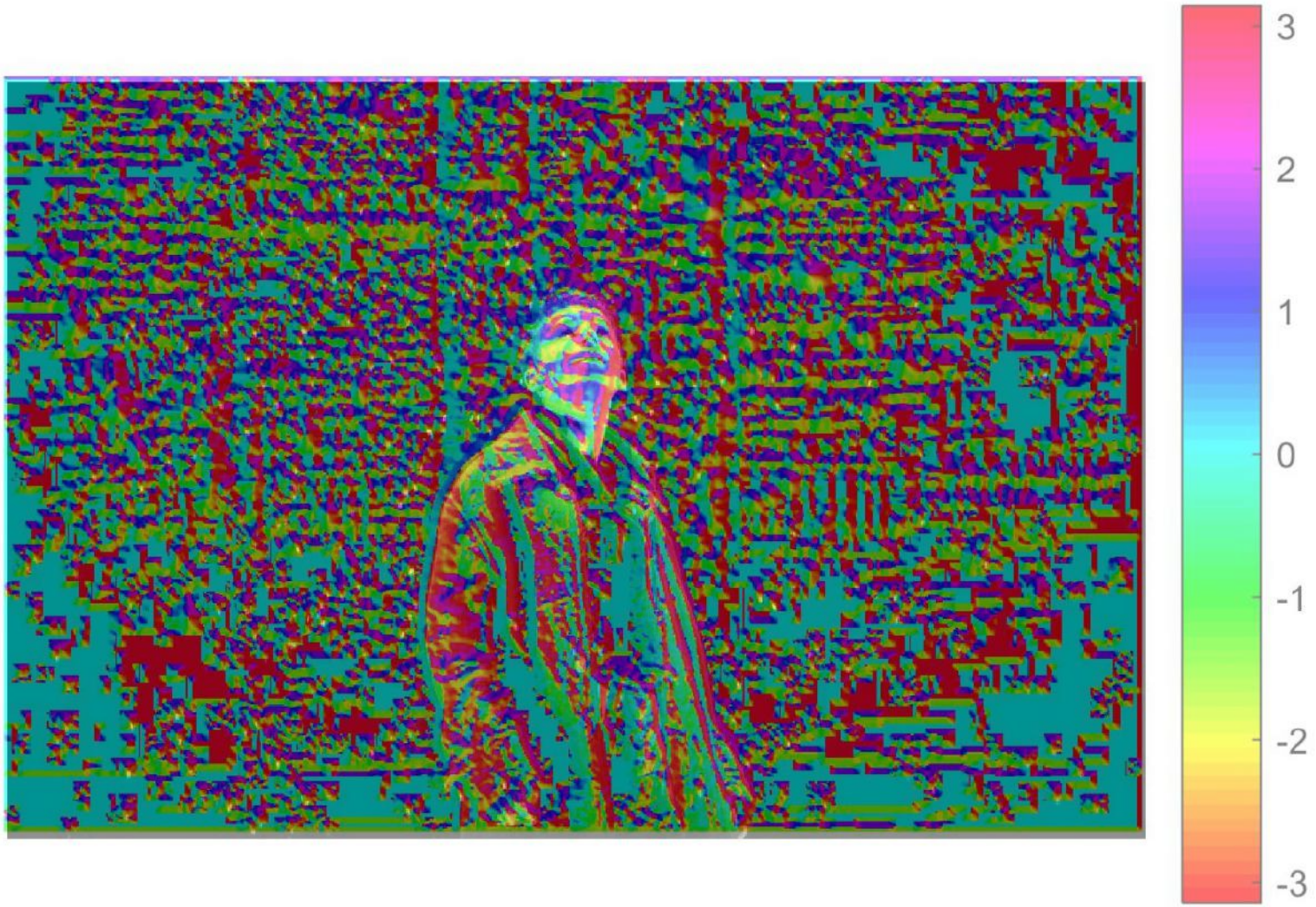


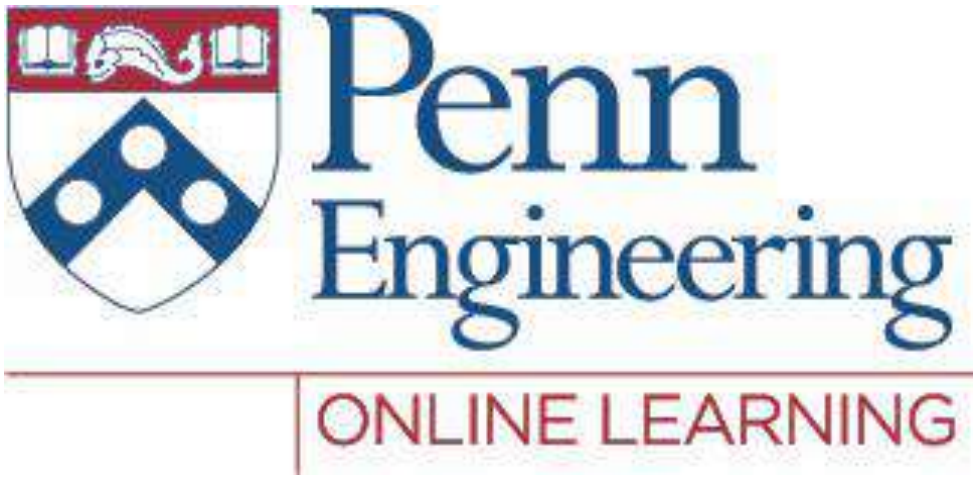






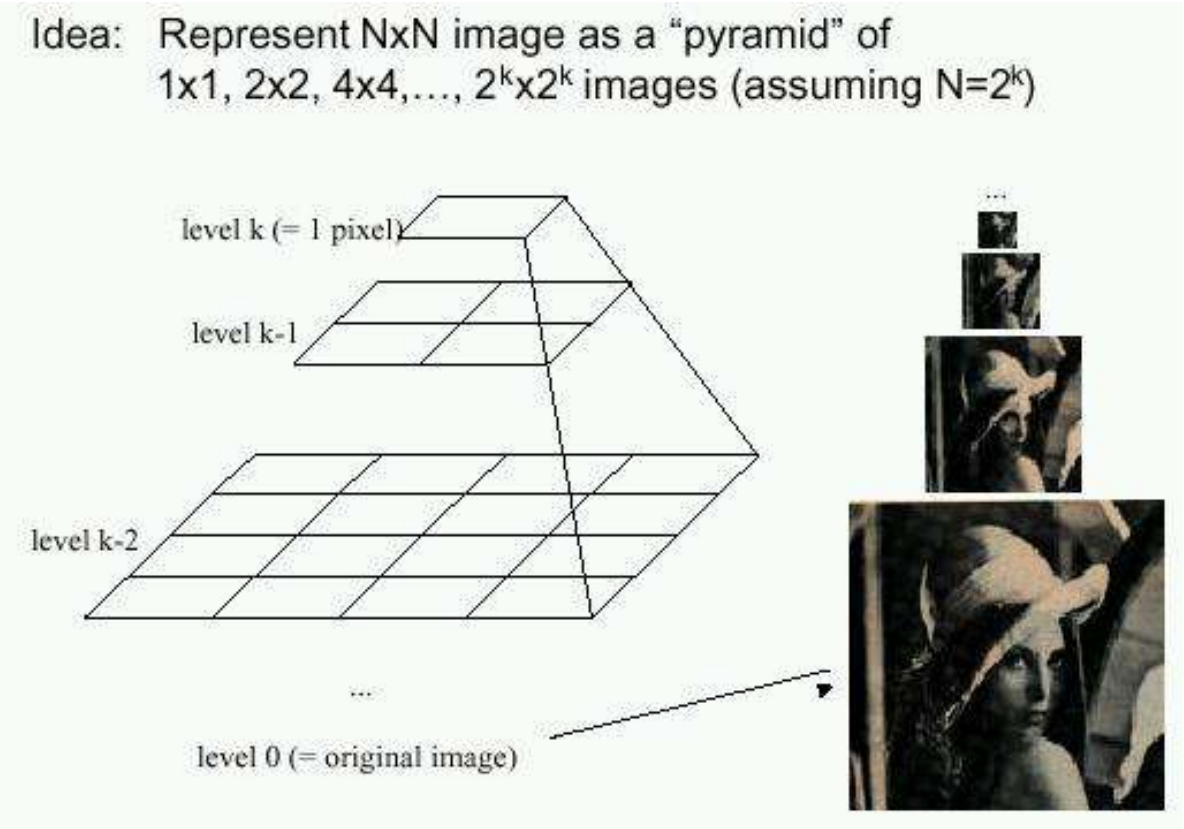






Video 3.4
Jianbo Shi

Image Pyramids



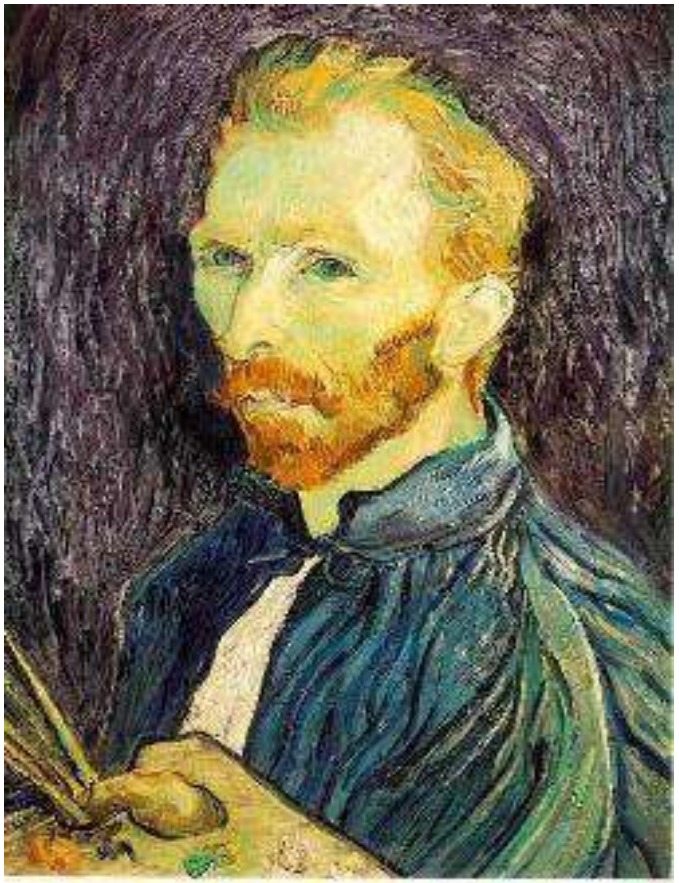
Known as a Gaussian Pyramid [Burt and Adelson, 1983]

- In computer graphics, a *mip map* [Williams, 1983]
- A precursor to *wavelet transform*

512 256 128 64 32 16 8

Figure from David Forsyth

Image sub-sampling



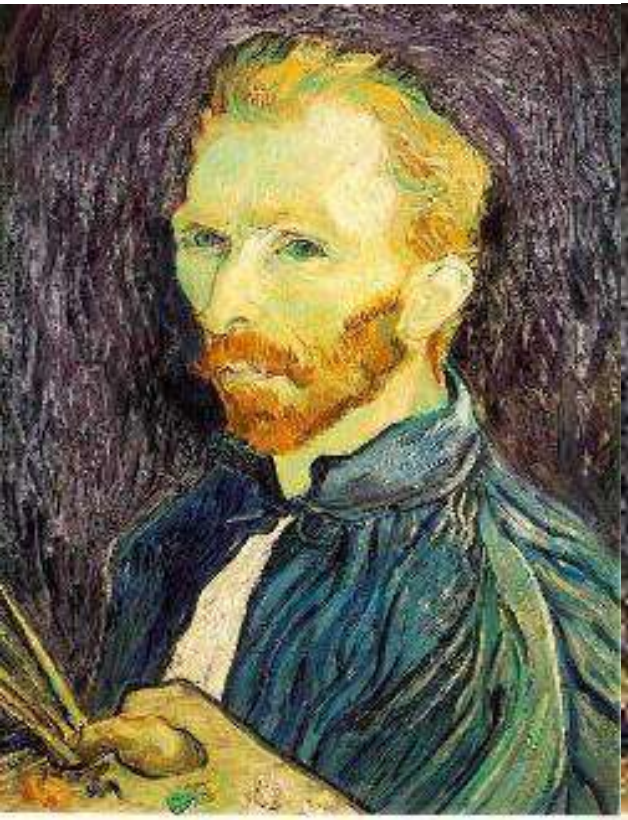
1/4

1/8

Throw away every other row and column to create a 1/2 size image

- called *image sub-sampling*

Image sub-sampling



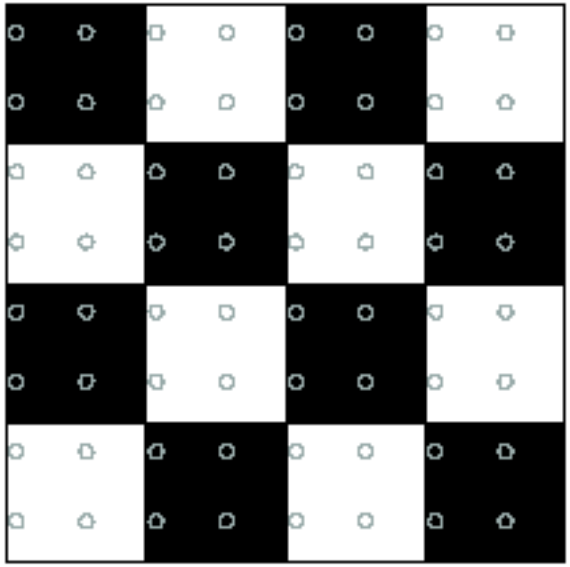
1/2

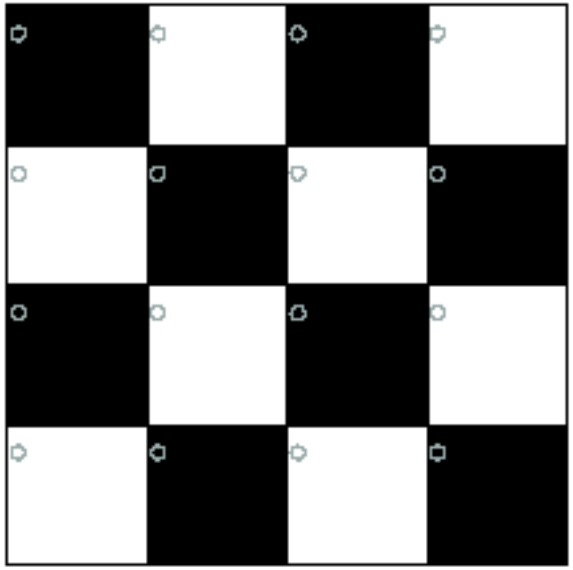
1/4 (2x zoom)

1/8 (4x zoom)

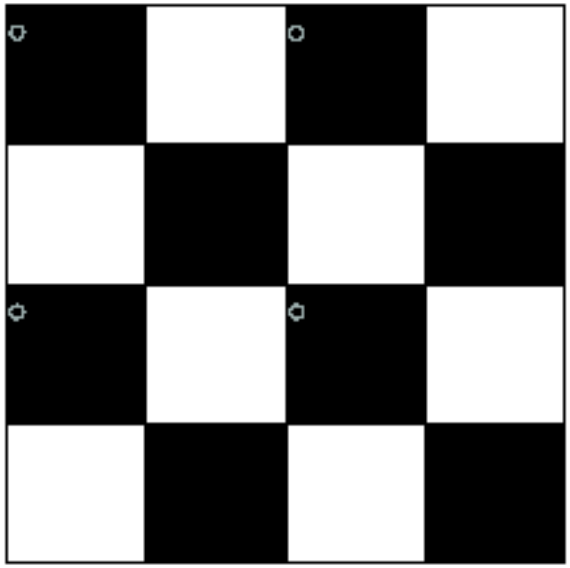
Why does this look so bad?

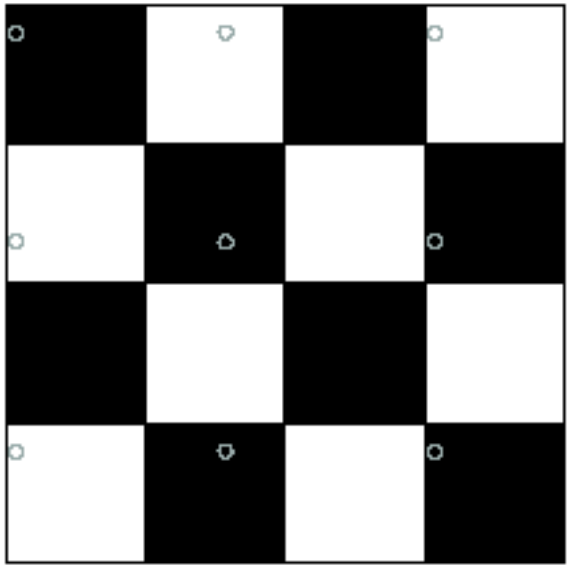
Sampling





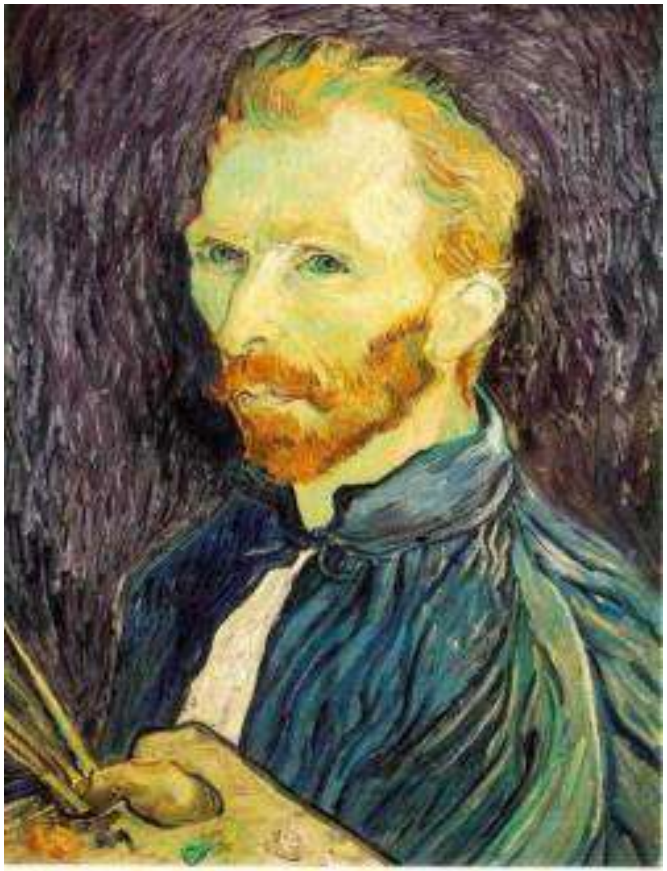
Good sampling:
- Sample often or,
- Sample wisely





Bad sampling:
- see aliasing in action!

Gaussian pre-filtering



Gaussian 1/2

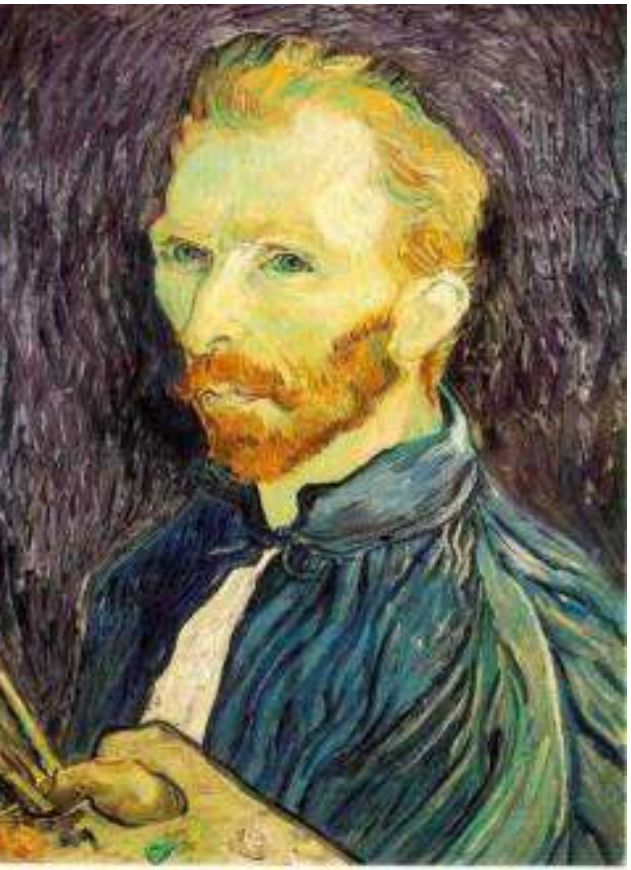
G 1/4

G 1/8

Solution: filter the image, *then* subsample

- Filter size should double for each ½ size reduction. Why?

Subsampling with Gaussian pre-filtering



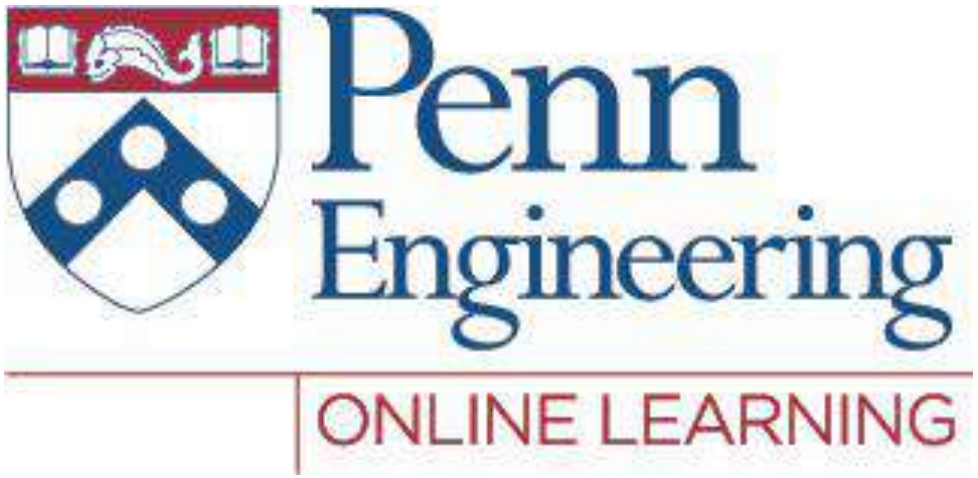
Gaussian 1/2

G 1/4

G 1/8

Solution: filter the image, *then* subsample

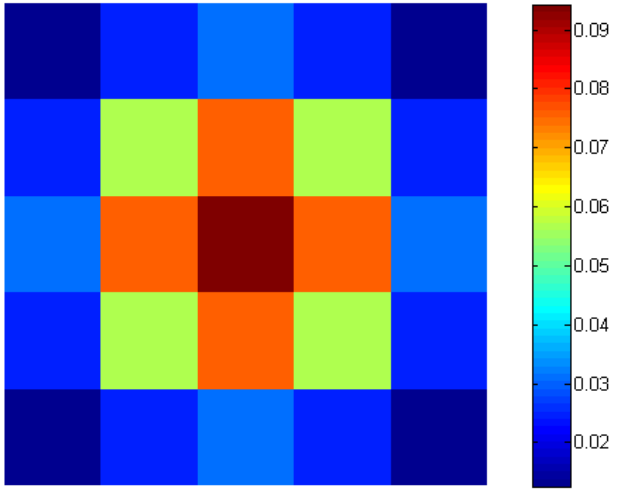
- Filter size should double for each ½ size reduction. Why?
- How can we speed this up?



Video 3.5
Jianbo Shi

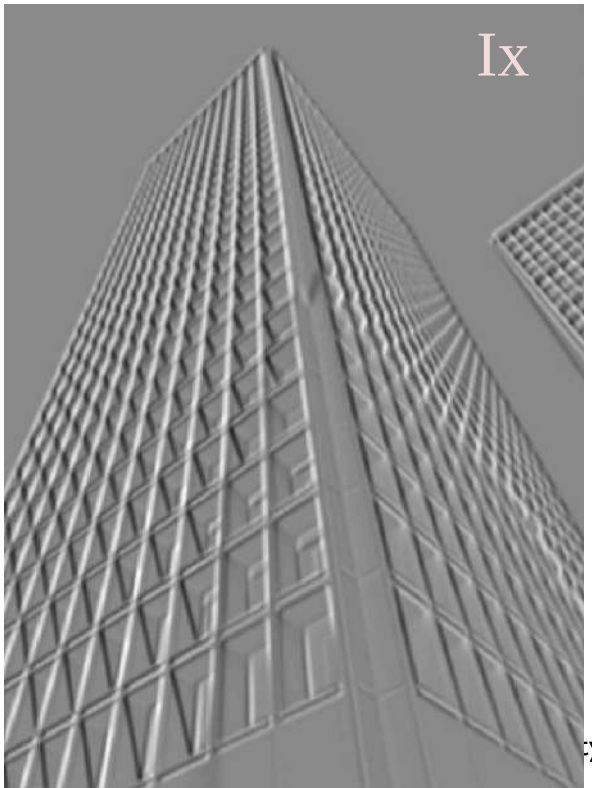
Canny Edge Implementation

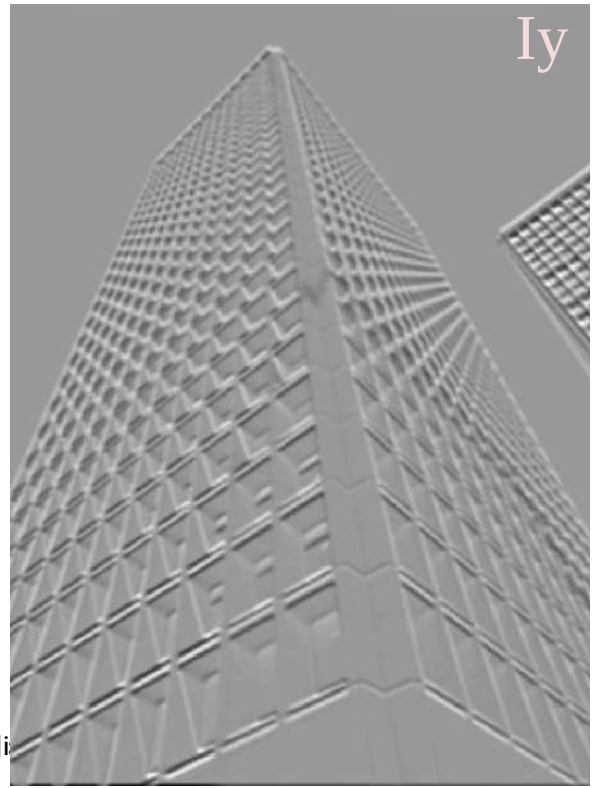
```
img = imread (‘image.png');
img = rgb2gray(img);
img = double (img);

% Value for high and low thresholding
threshold_low = 0.035;
threshold_high = 0.175;

%% Gaussian filter (https://en.wikipedia.org/wiki/Canny_edge_detector)
G = [2, 4, 5, 4, 2; 4, 9, 12, 9, 4;5, 12, 15, 12, 5;4, 9, 12, 9, 4;2, 4, 5, 4, 2];
G = 1/159.* G;

%Filter for horizontal and vertical direction
dx = [1 -1];
dy = [1; -1];
```



Canny Edge Implementation

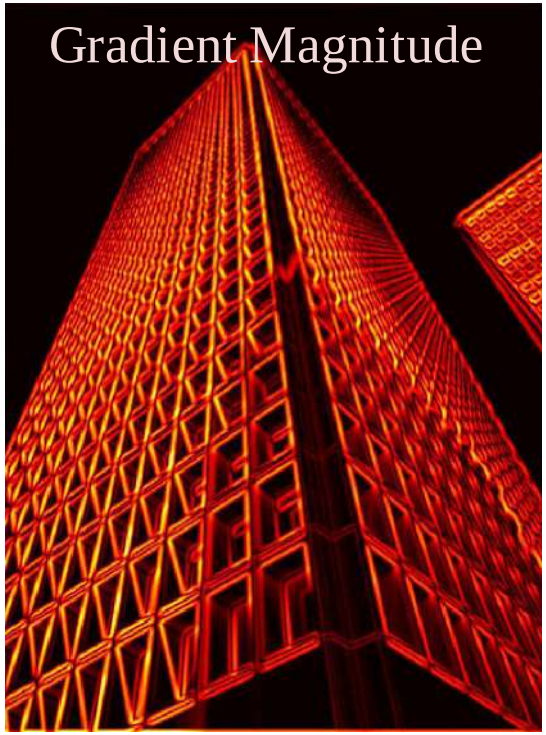
```
% % Convolution of image with Gaussian
Gx = conv2(G, dx, 'same');
Gy = conv2(G, dy, 'same');

% Convolution of image with Gx and Gy
Ix = conv2(img, Gx, 'same');
Iy = conv2(img, Gy, 'same');
```

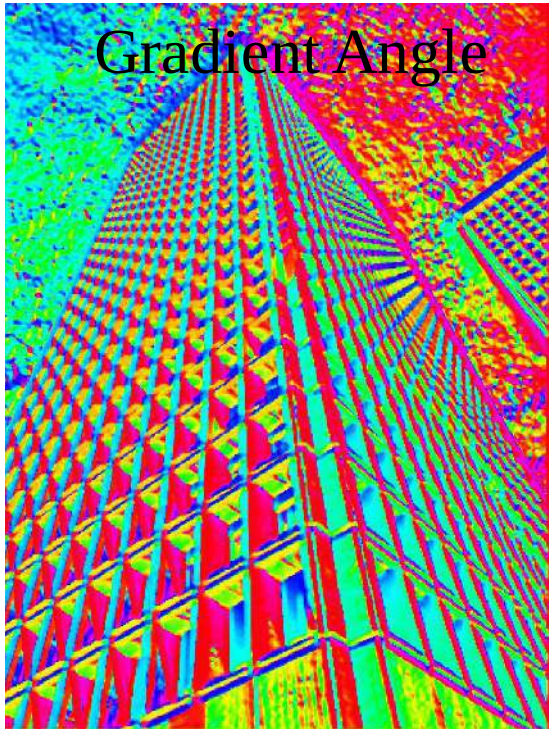


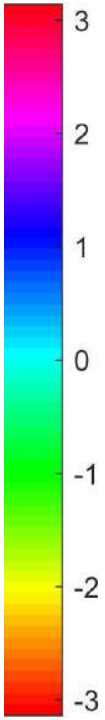


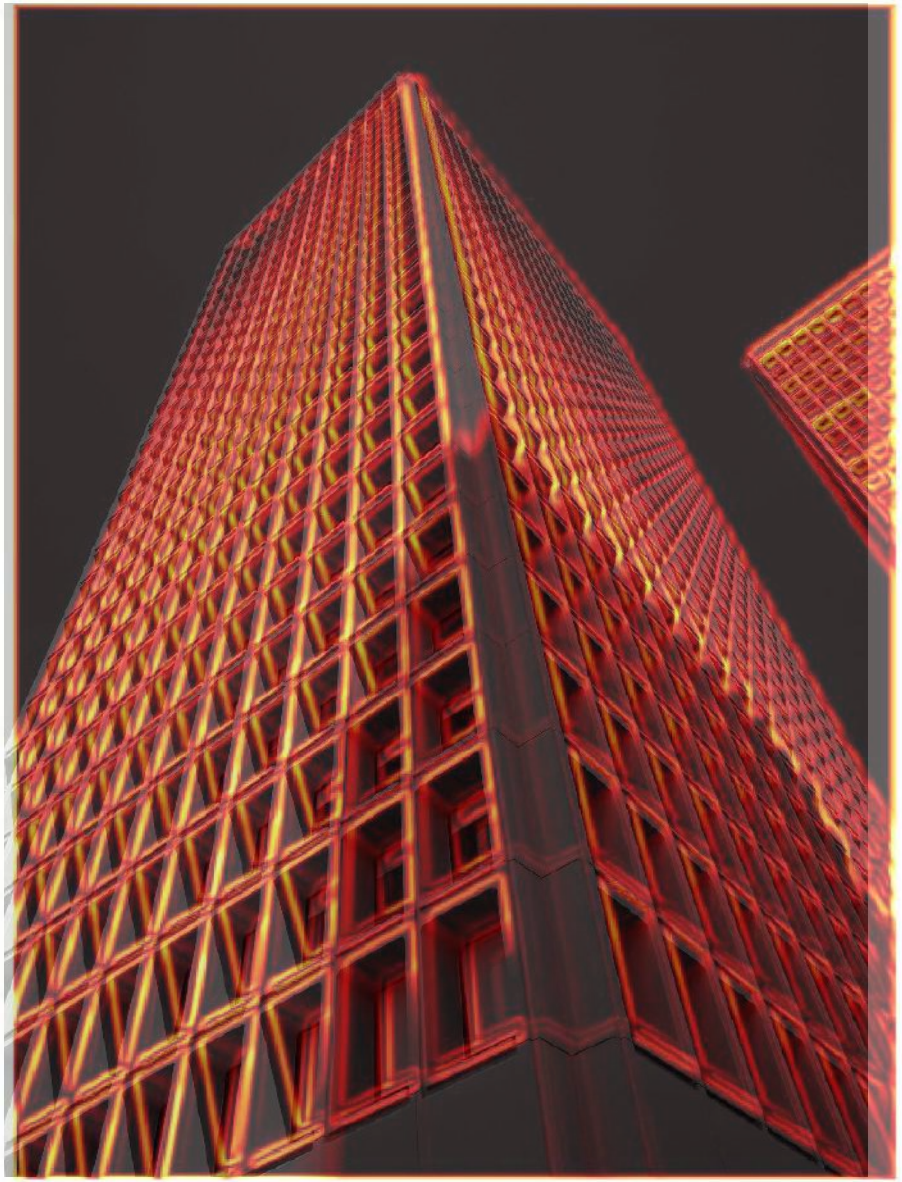
Canny Edge Implementation

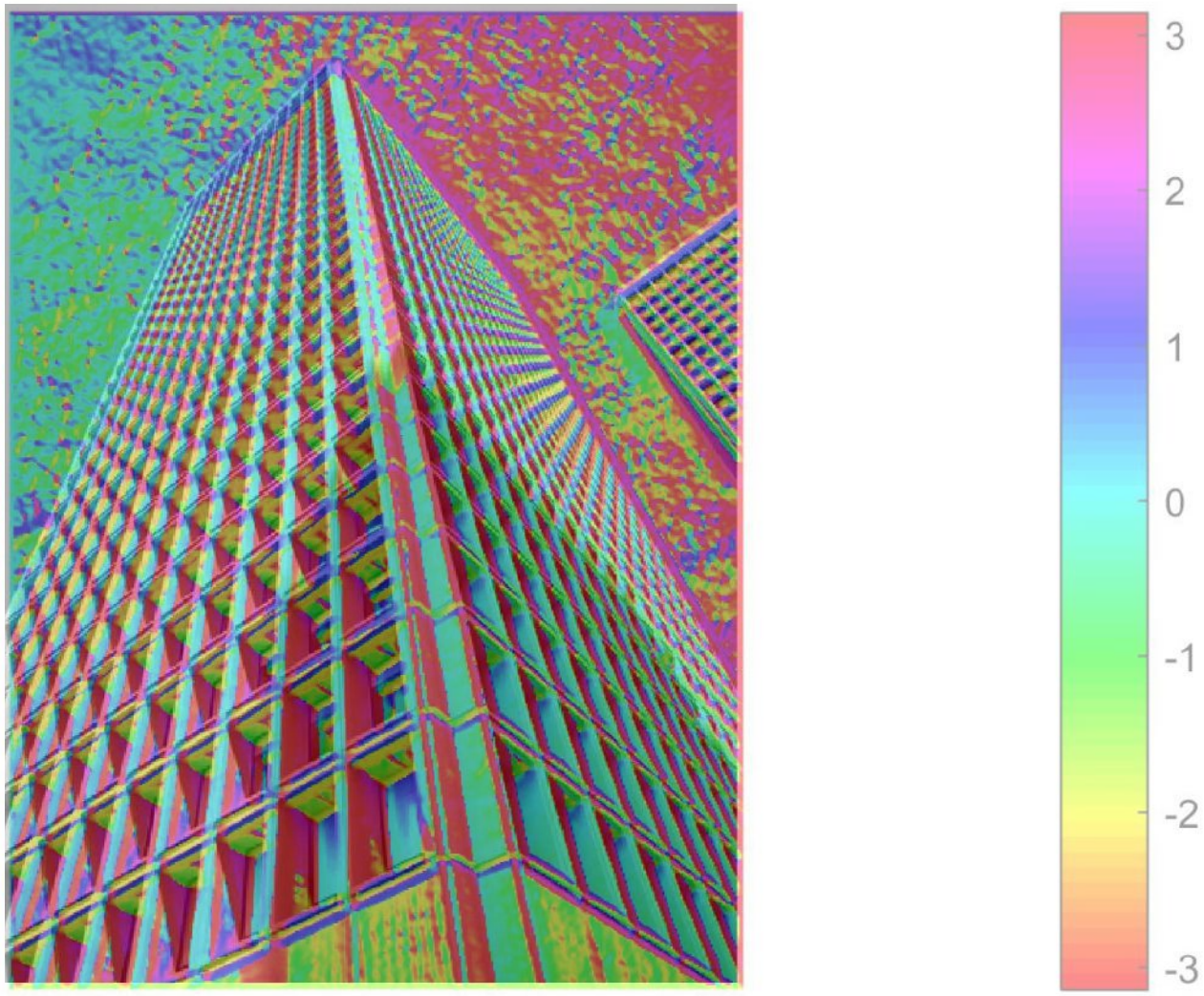
angle = atan2(Iy, Ix);
mag = sqrt(Iy.^2 + Ix.^2);



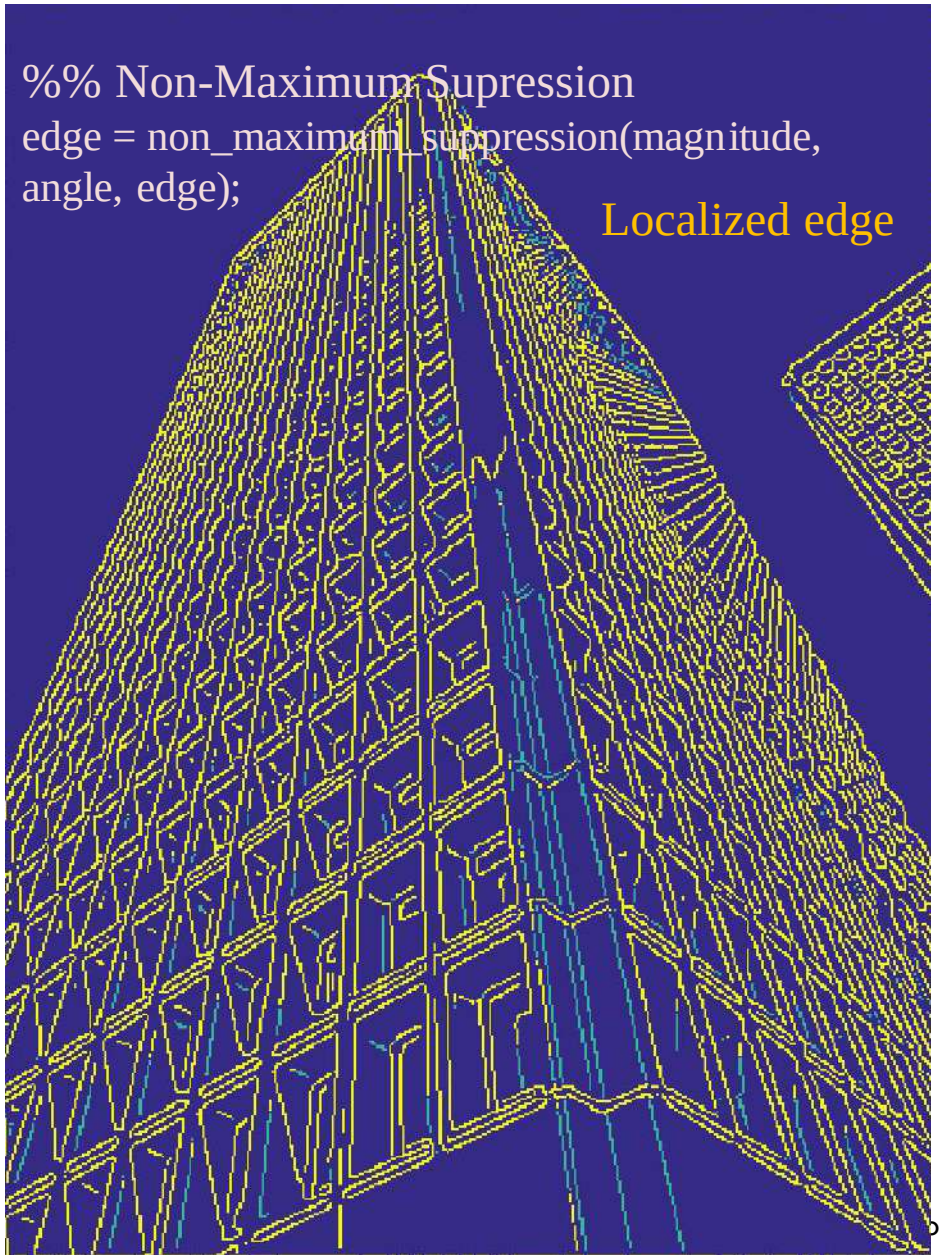


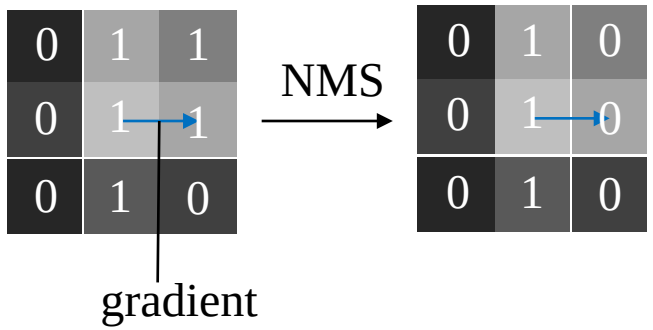


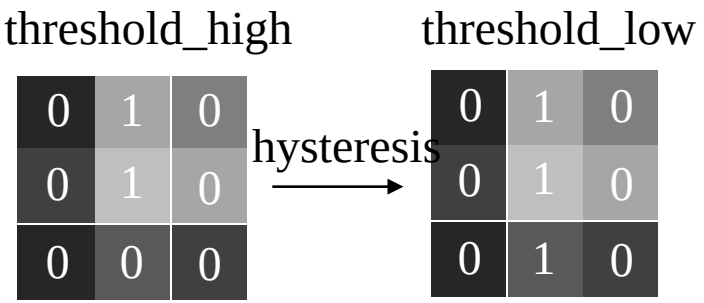




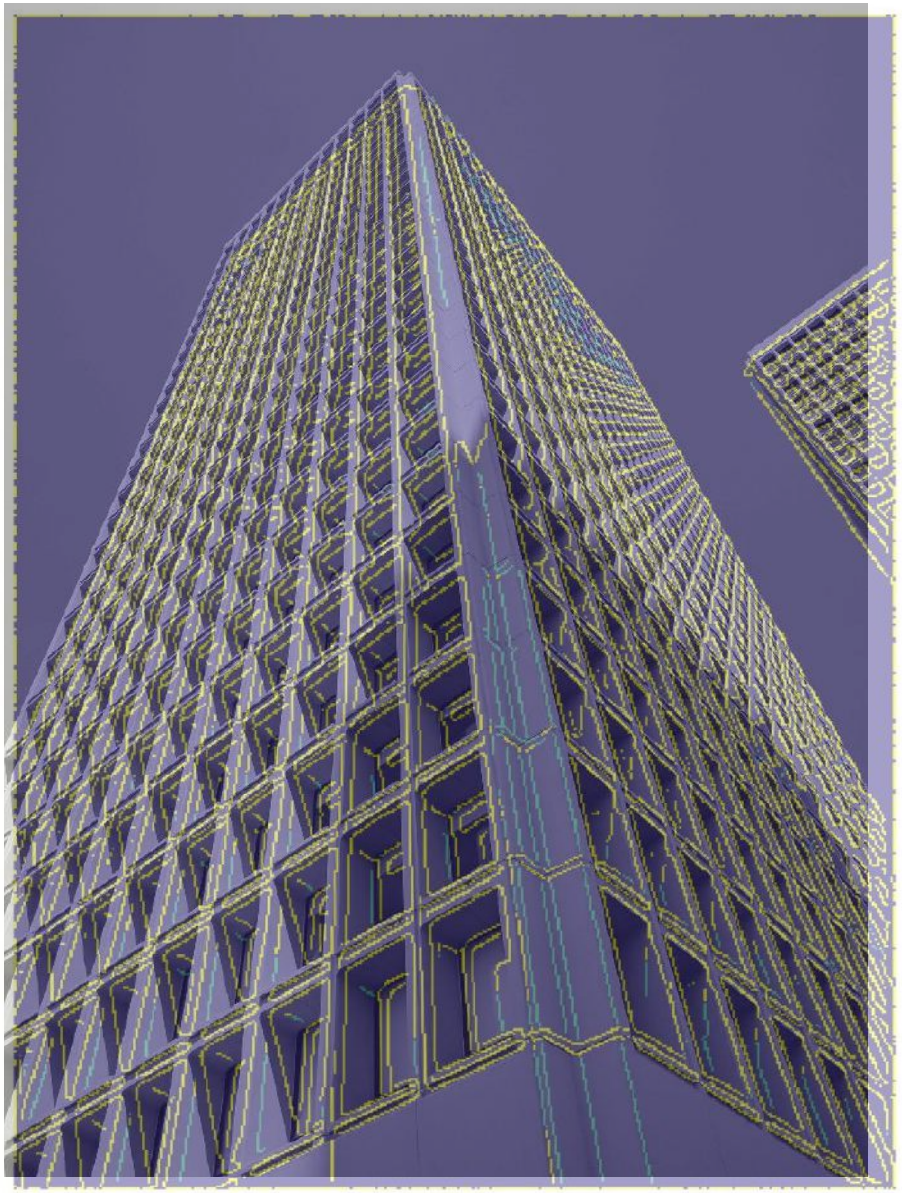
Canny Edge Implementation



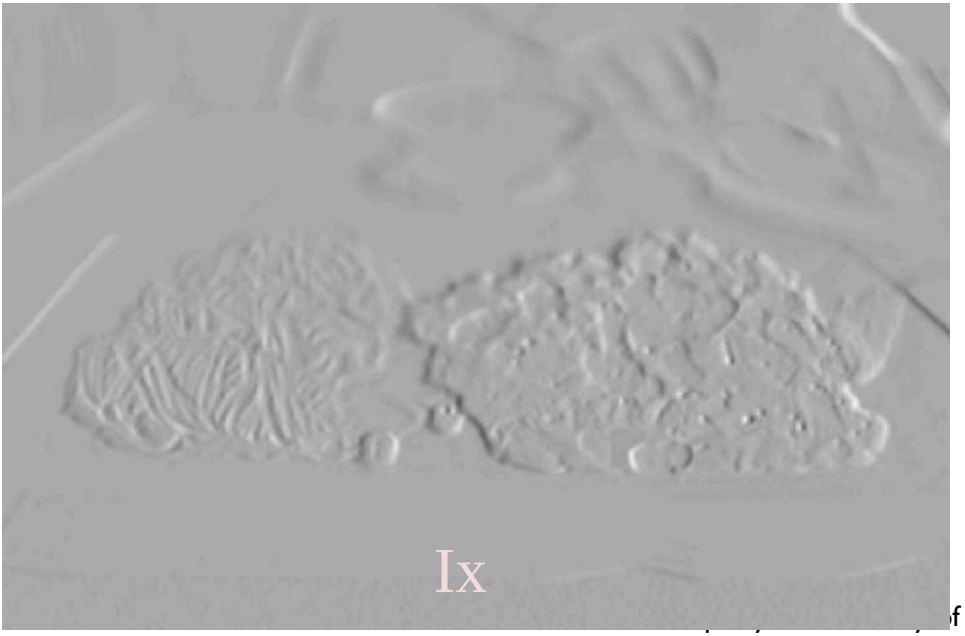


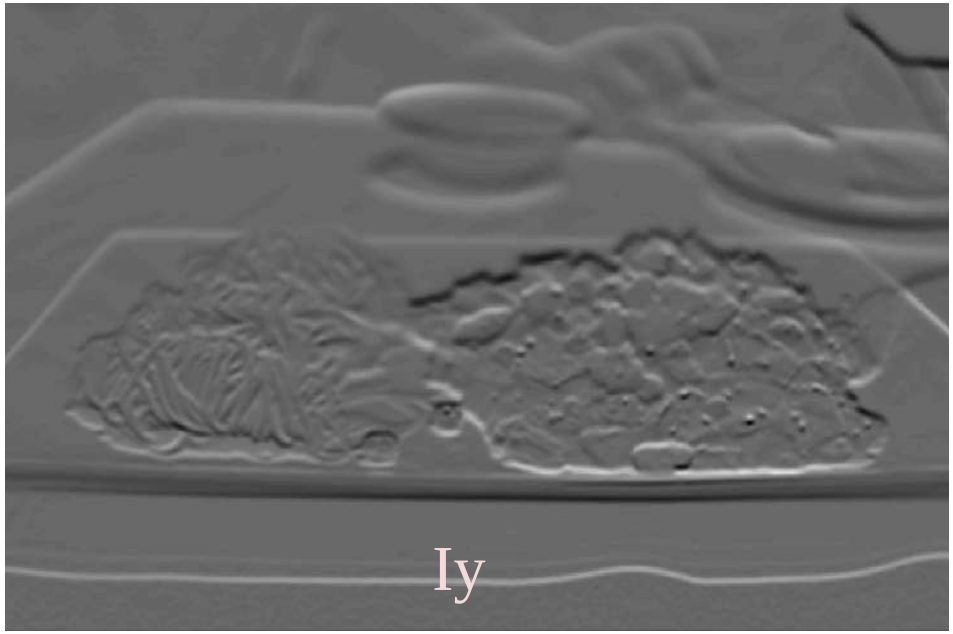



```
low = threshold_low * max(edge(:));
high = threshold_high * max(edge(:));
linked_edge = hysteresis_thresholding(low, high);
```

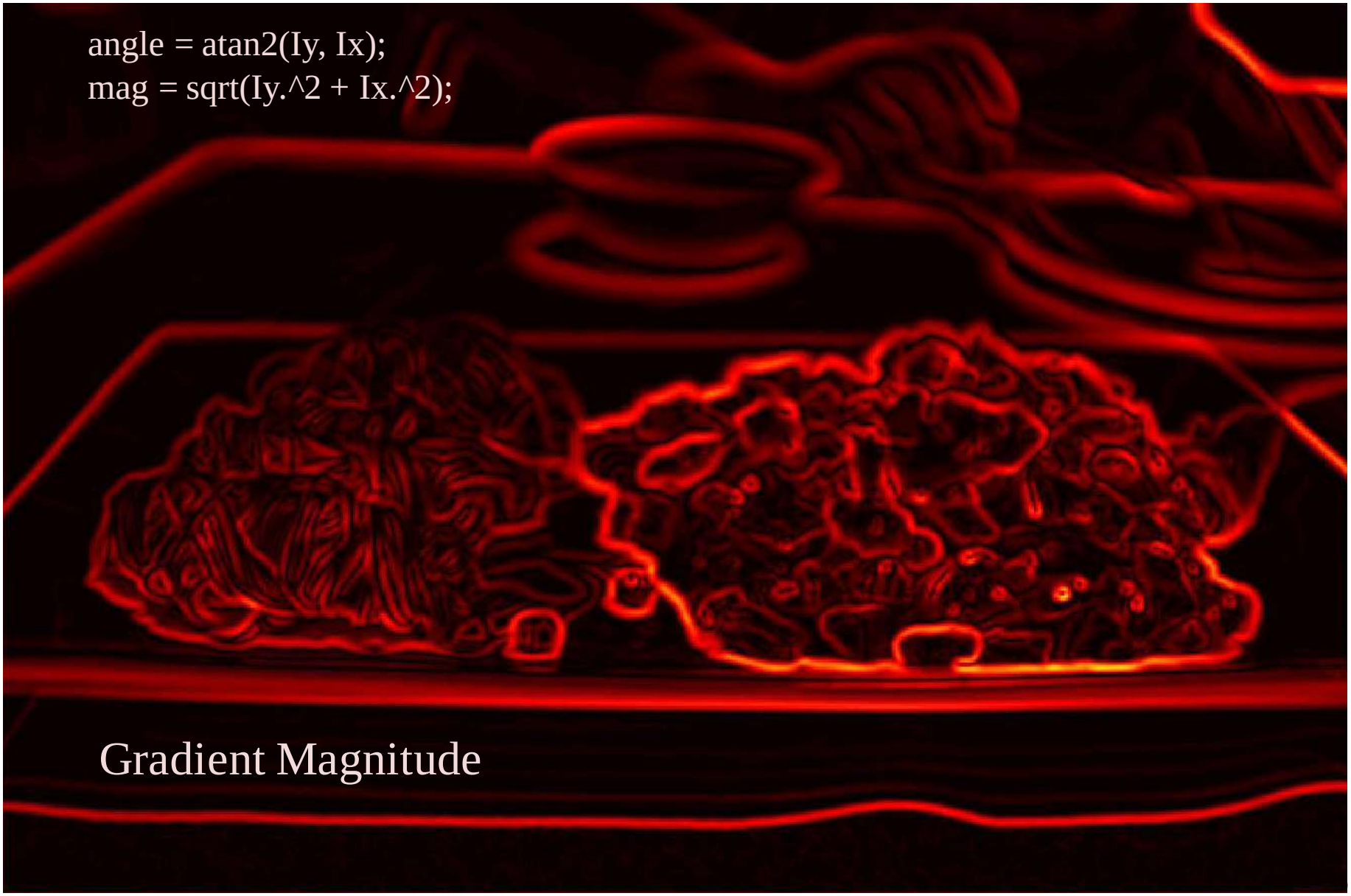


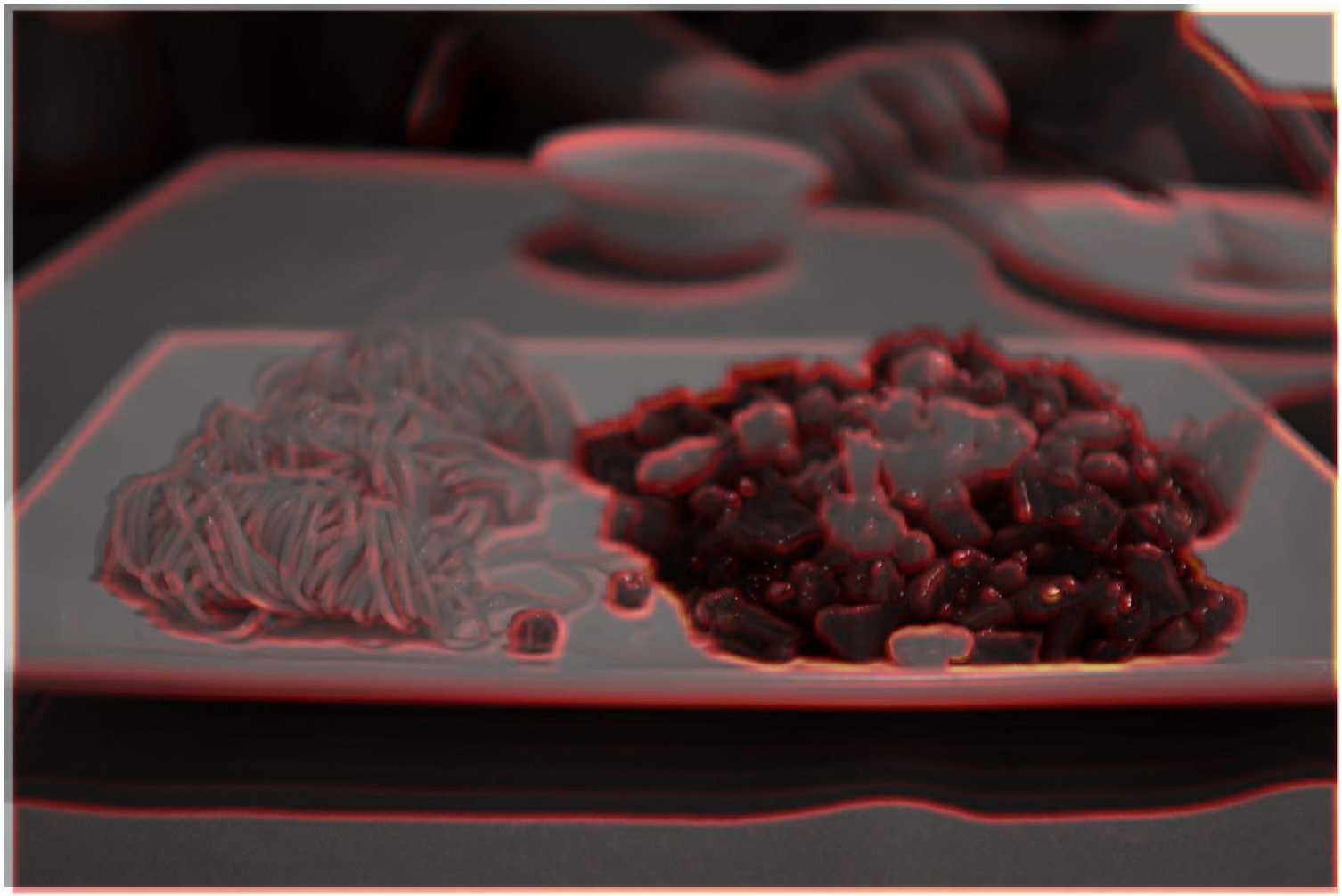
```
% % Convolution of image with Gaussian
Gx = conv2(G, dx, 'same');
Gy = conv2(G, dy, 'same');

% Convolution of image with Gx and Gy
Ix = conv2(img, Gx, 'same');
Iy = conv2(img, Gy, 'same');
```



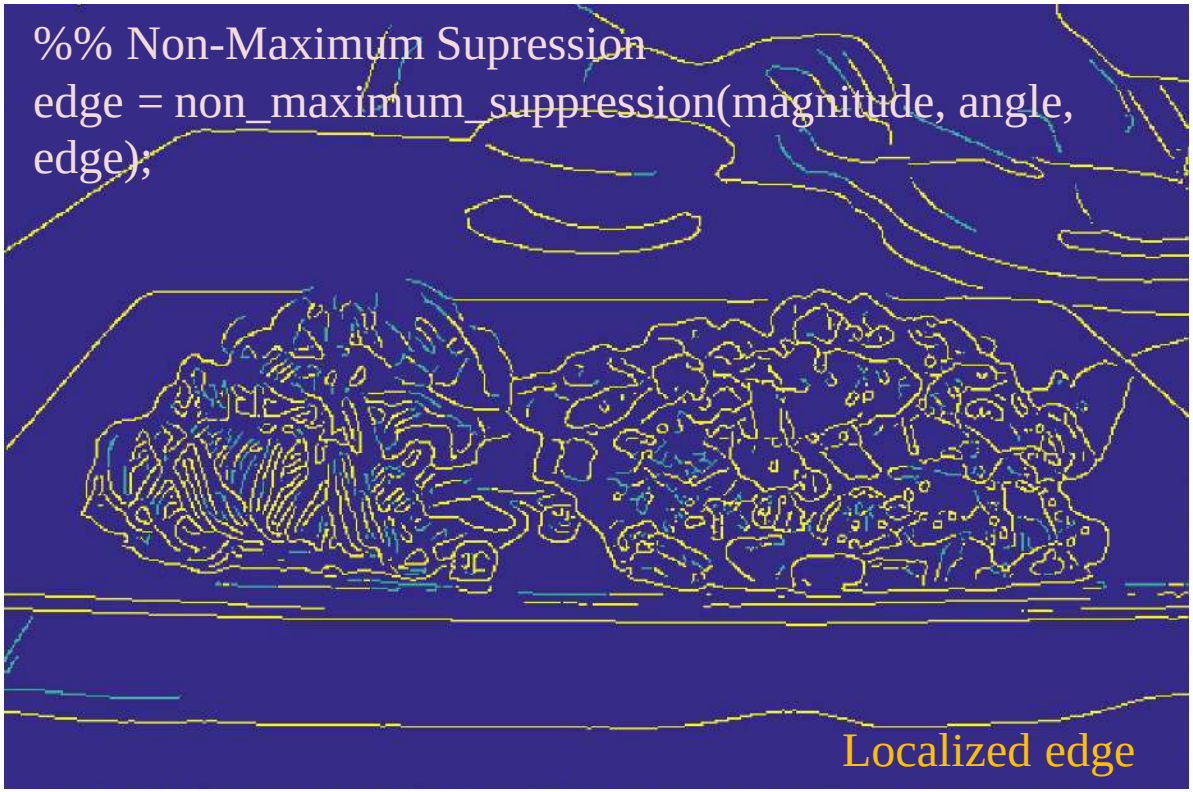


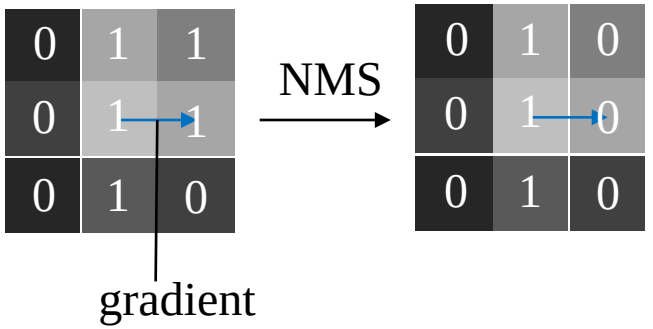
Canny Edge Implementation

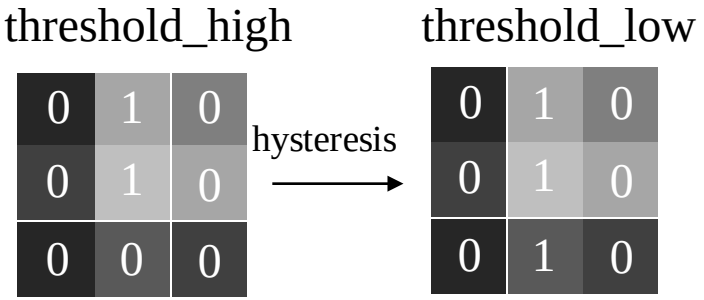




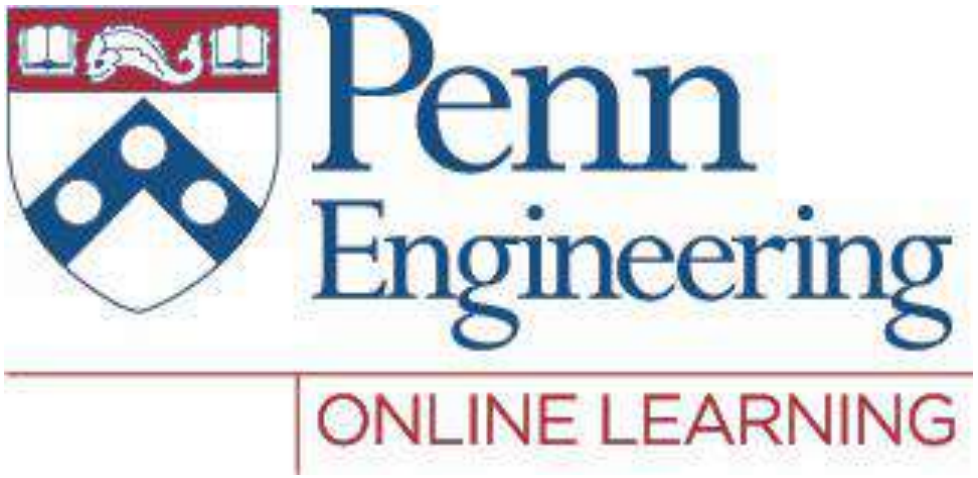
Canny Edge Implementation








```
low = threshold_low * max(edge(:));
high = threshold_high * max(edge(:));
linked_edge = hysteresis_thresholding(low, high);
```

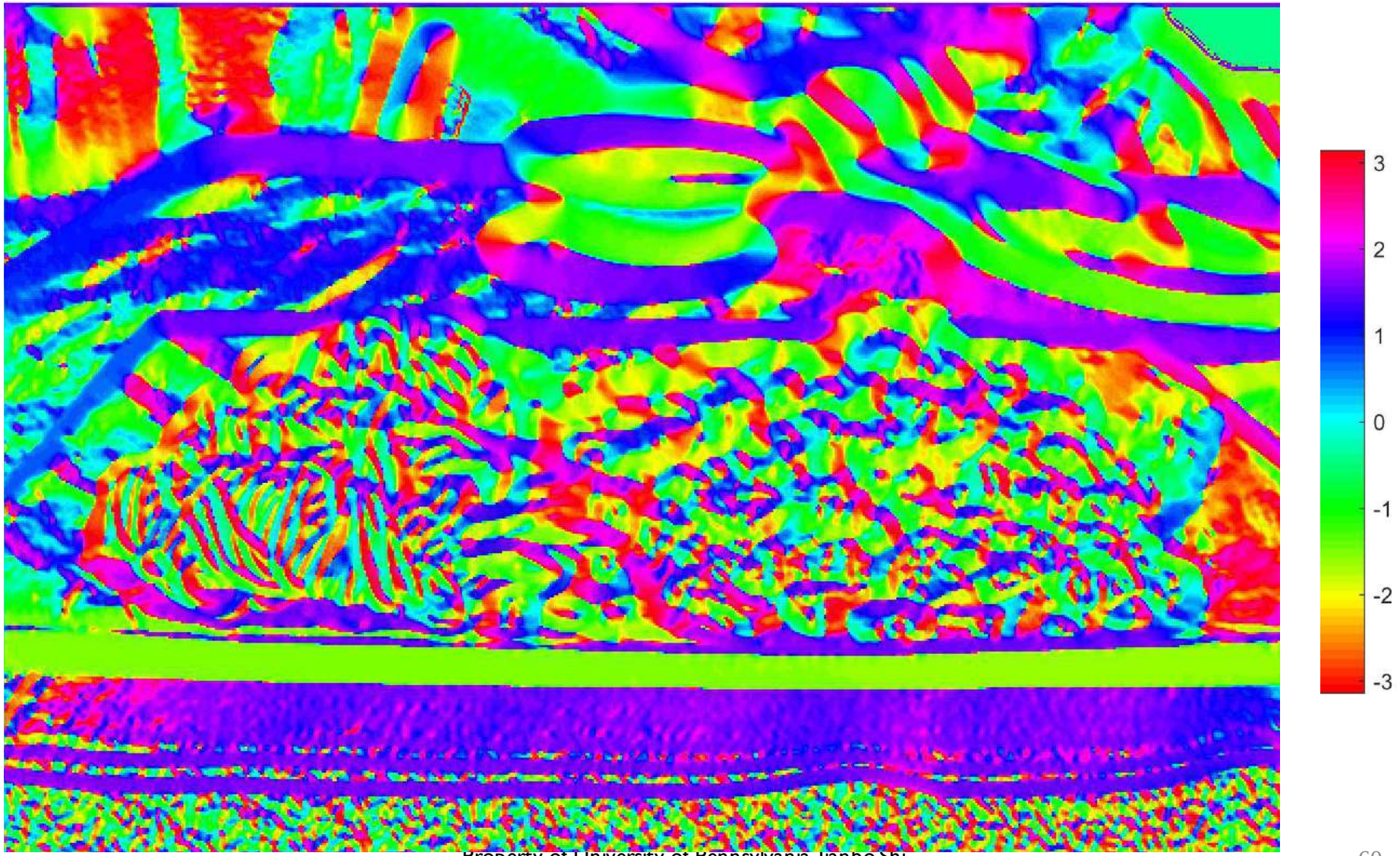


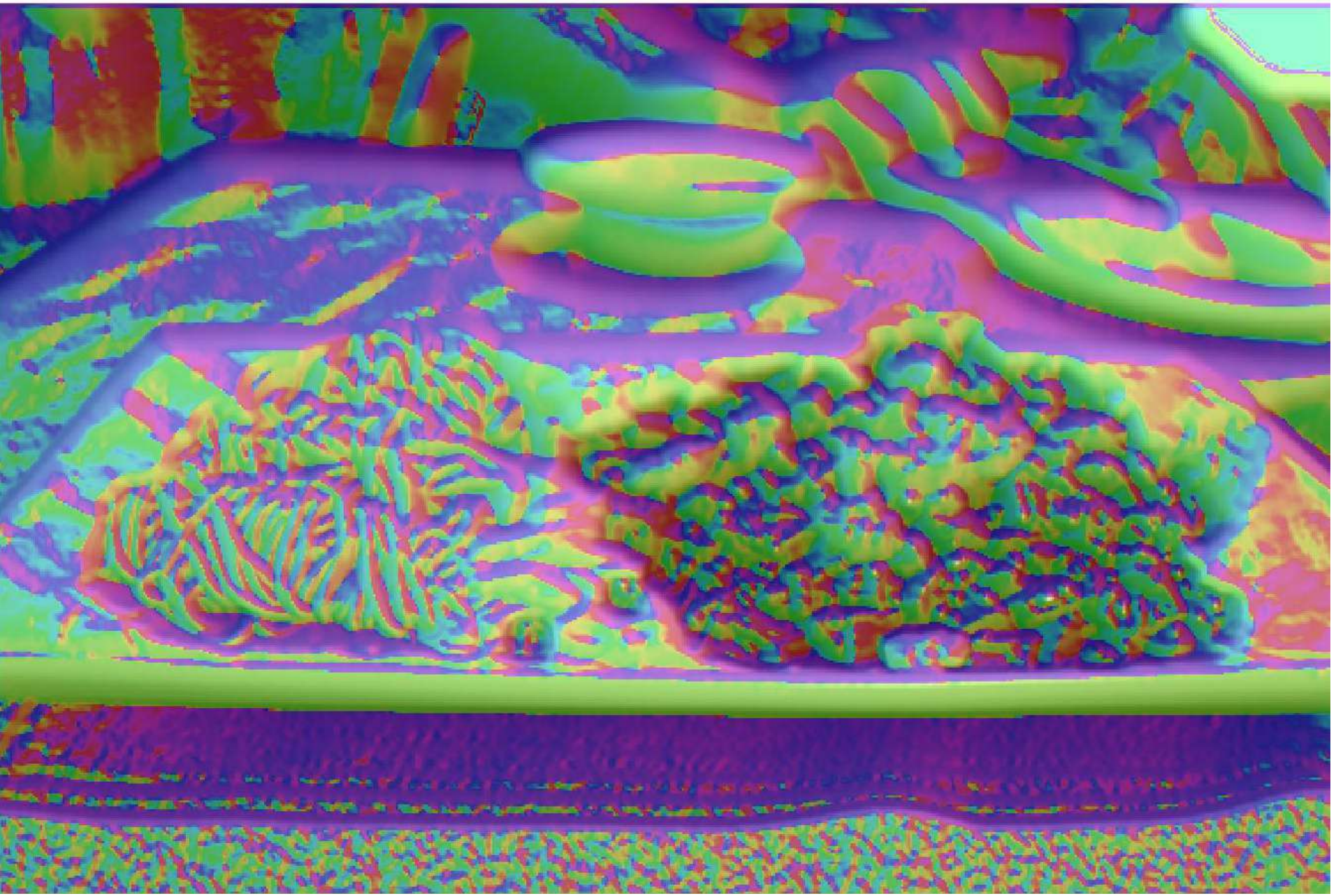
Video 3.6
Jianbo Shi

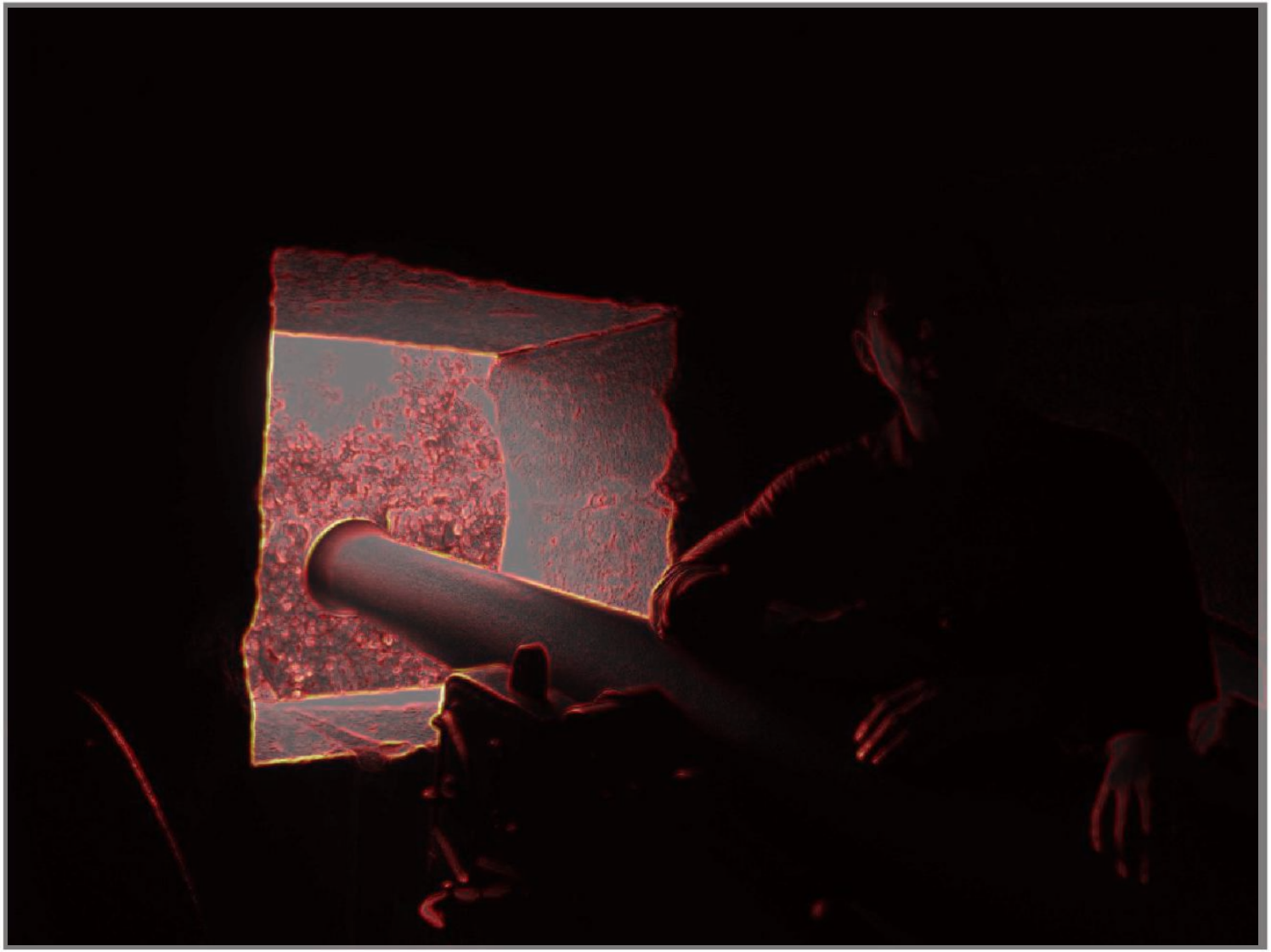
Edge Implementation

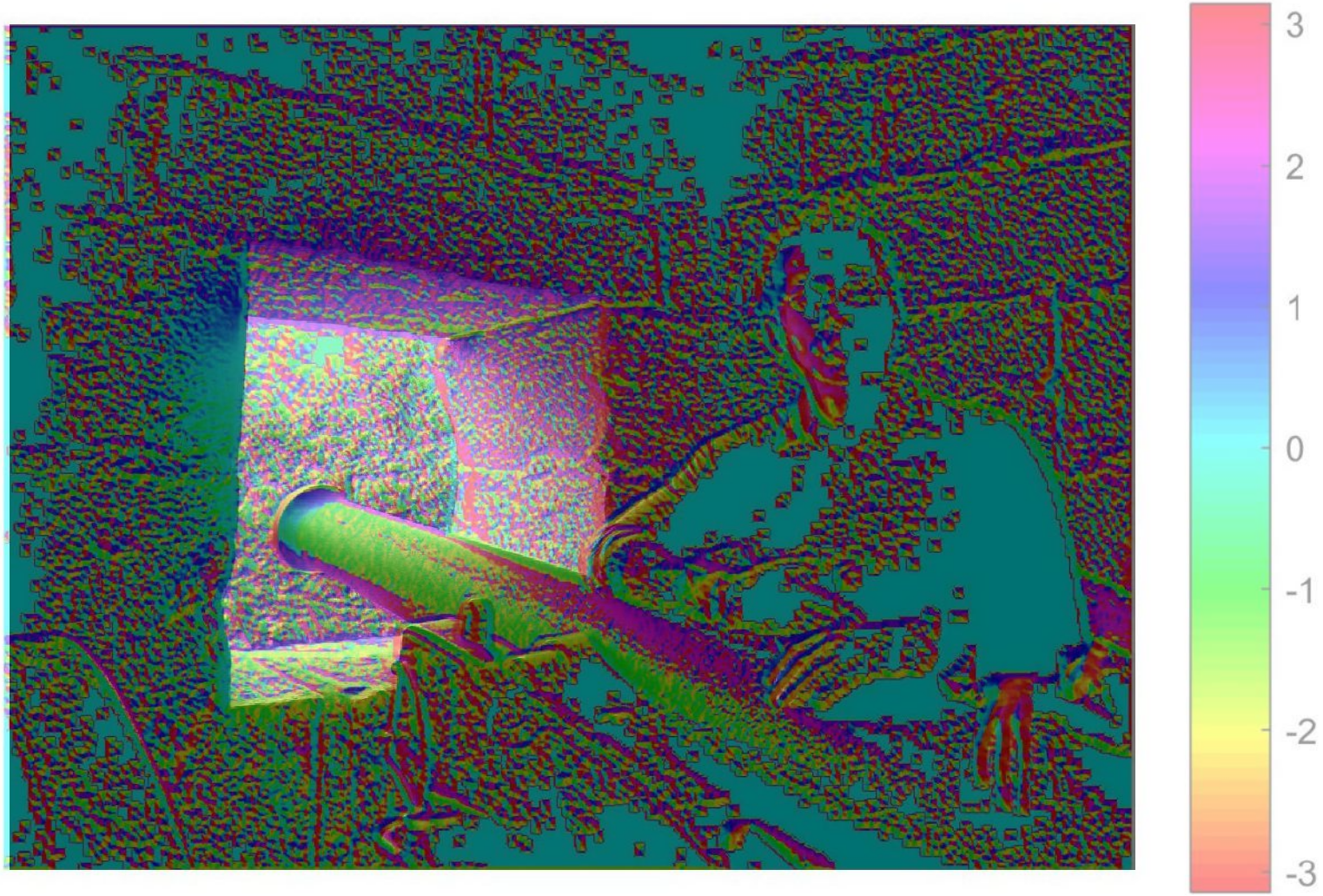
```
angle = atan2(Iy, Ix);
mag = sqrt(Iy.^2 + Ix.^2);
```

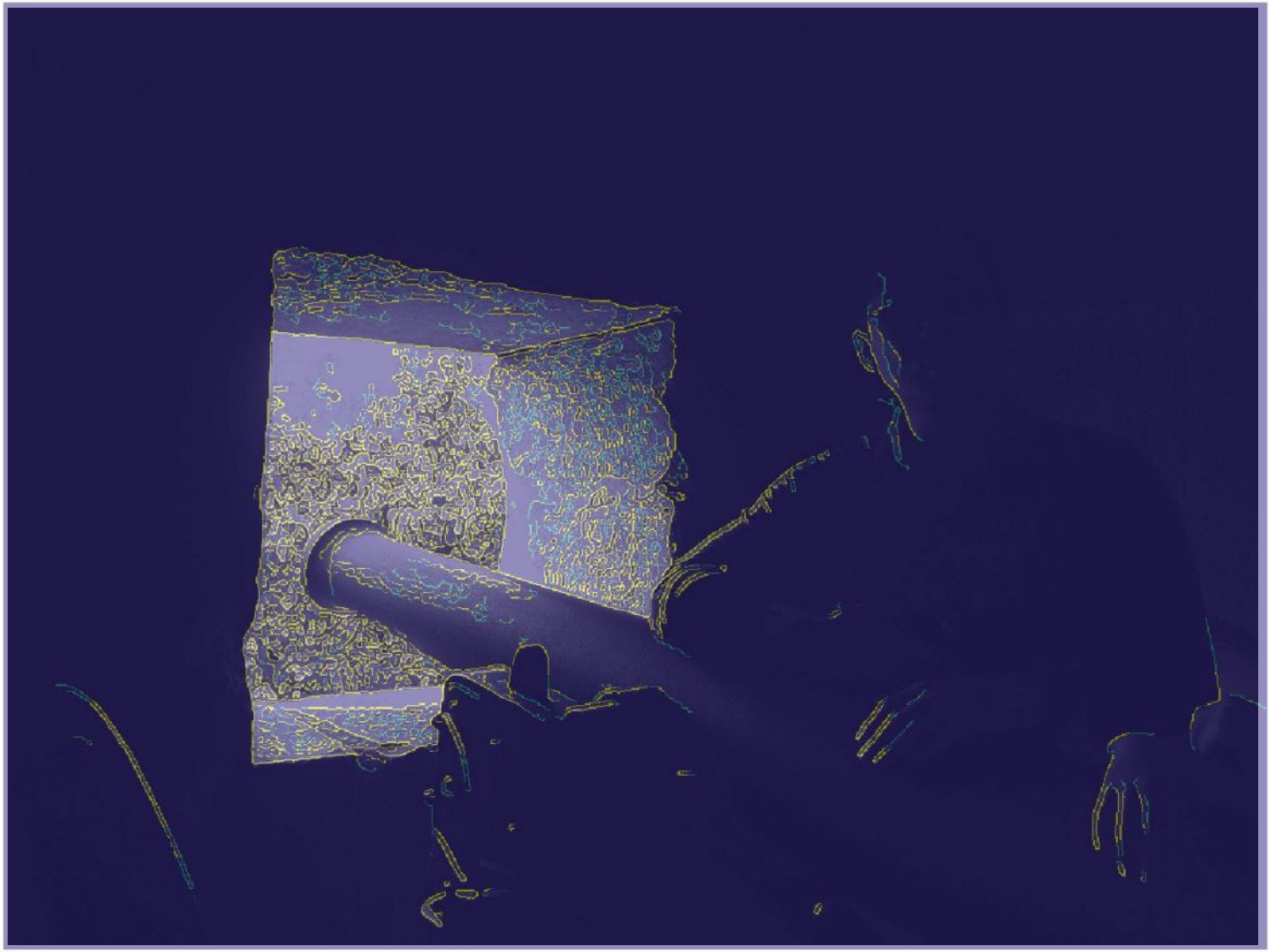
Gradient Angle



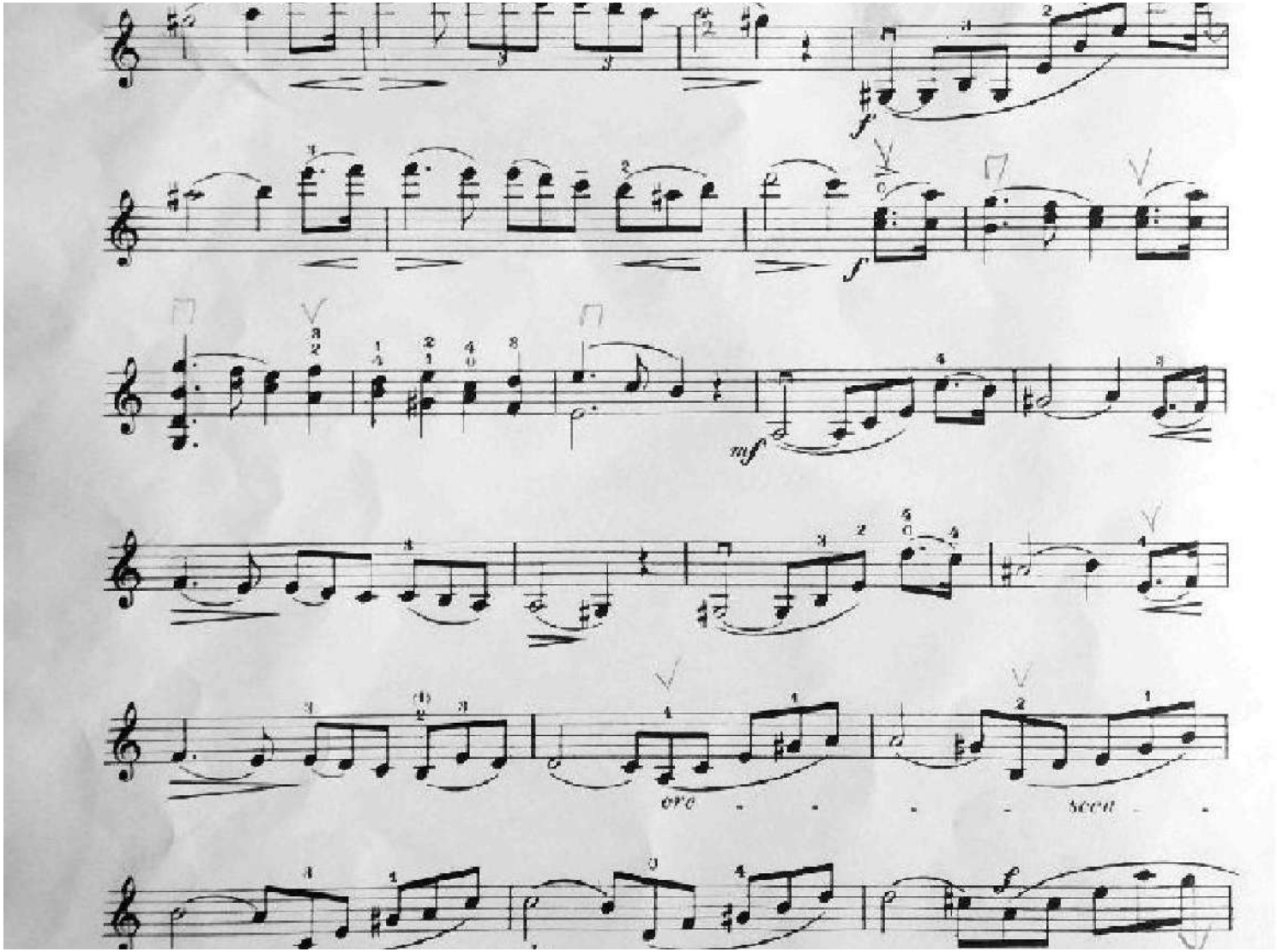




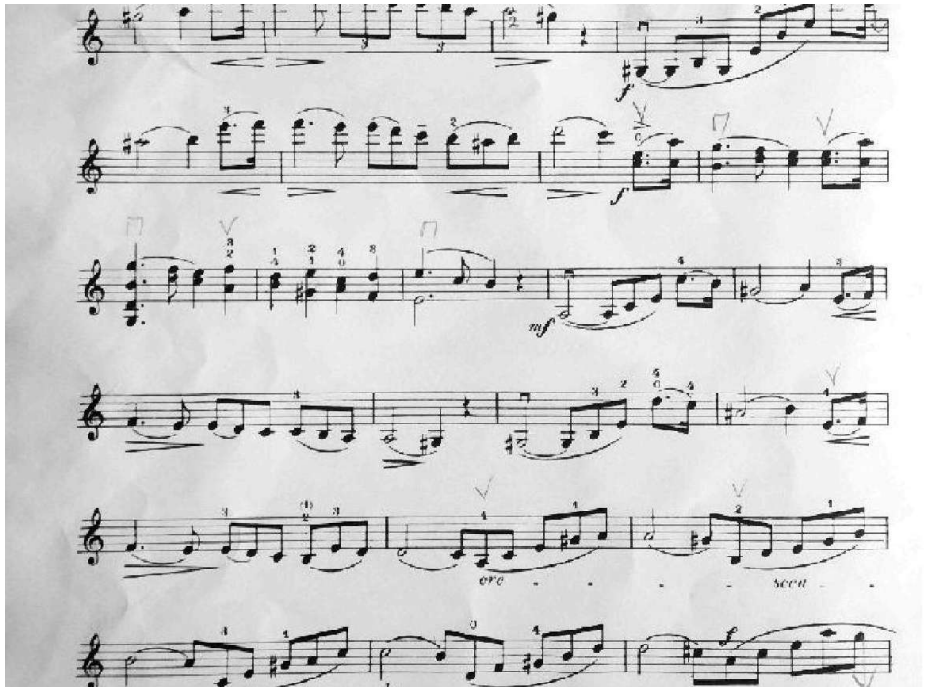


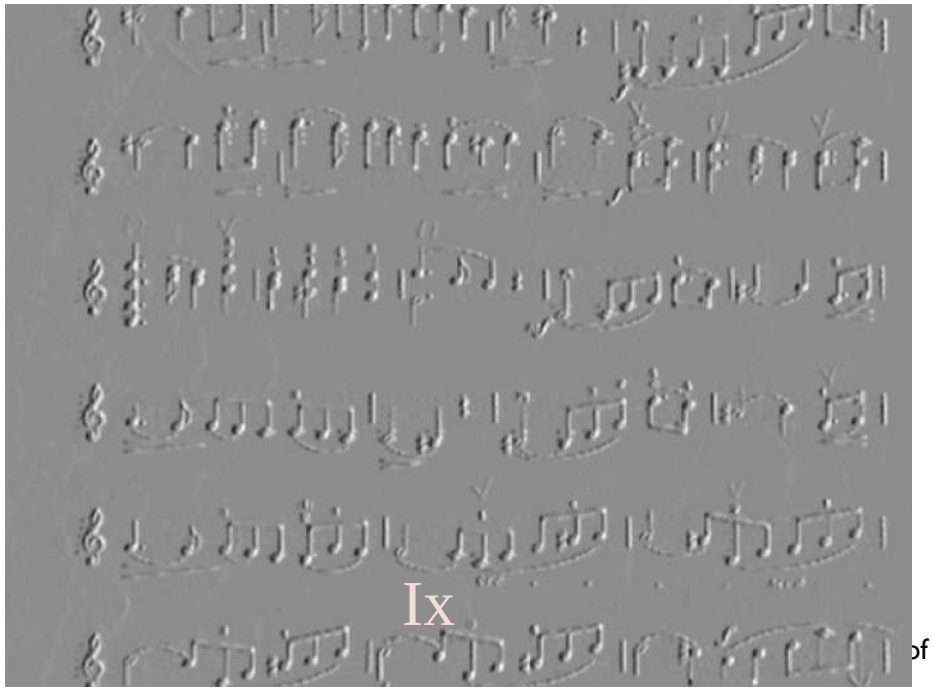


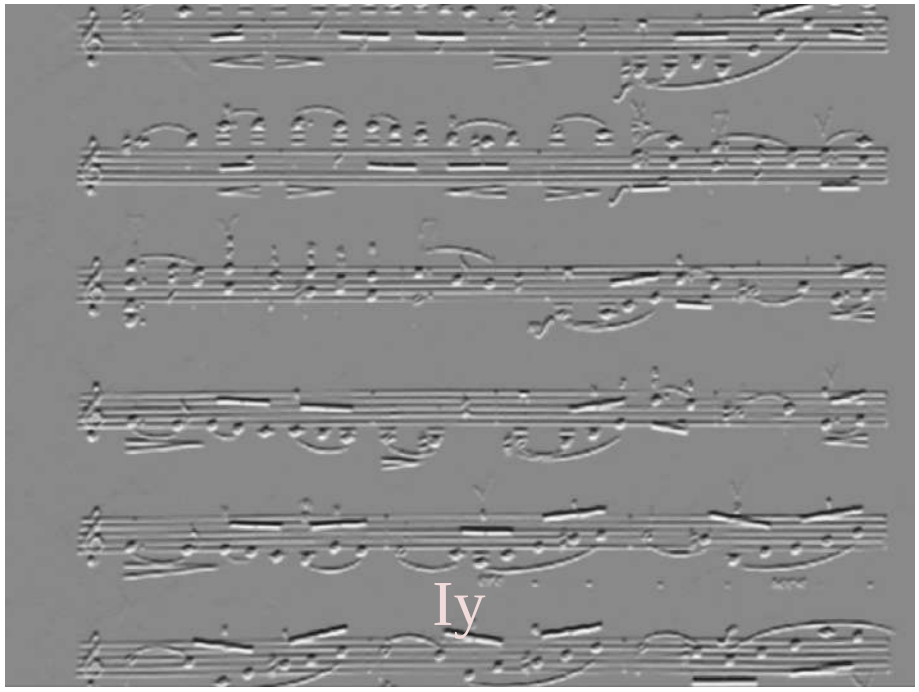
Ix Iy

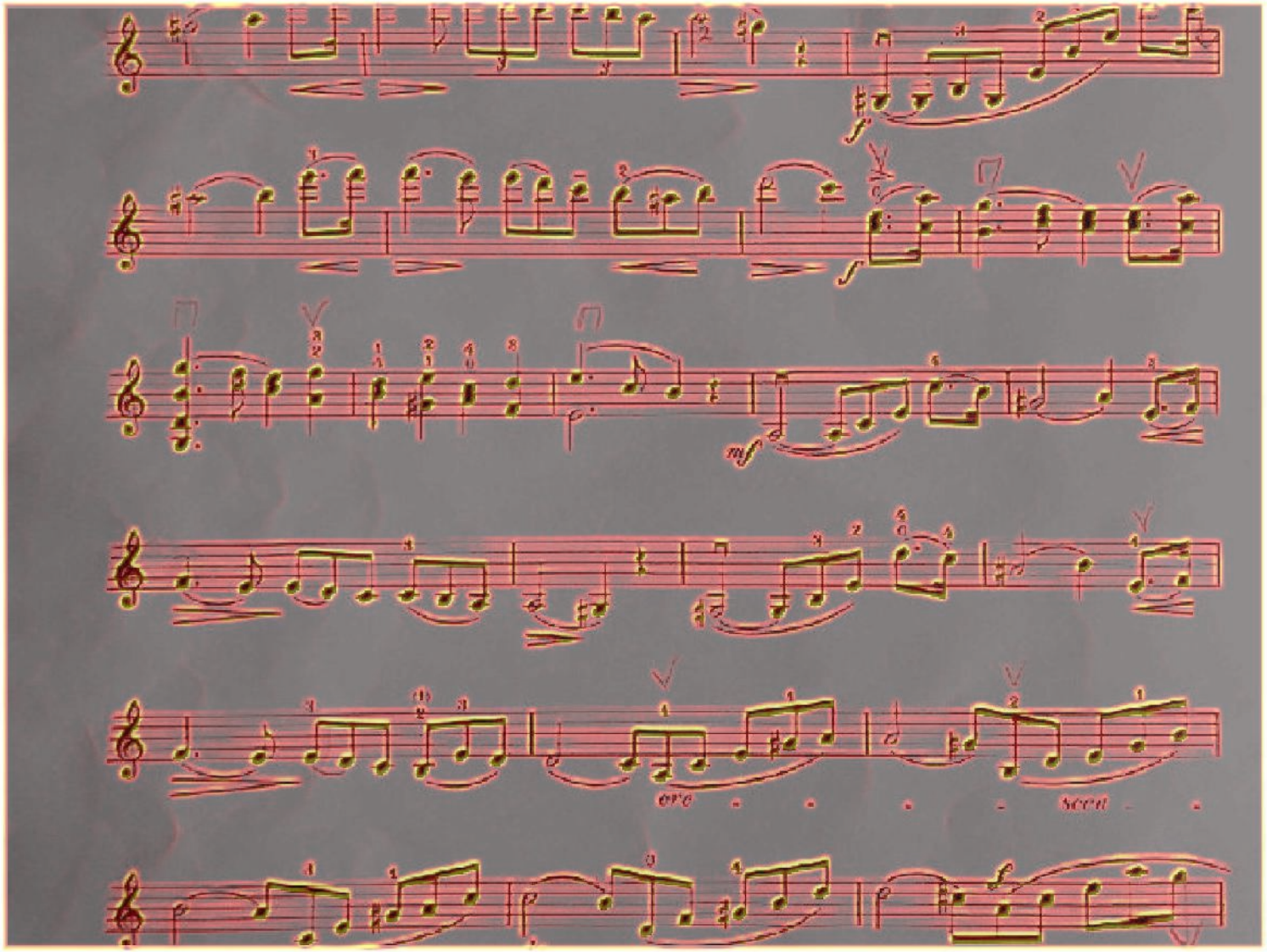


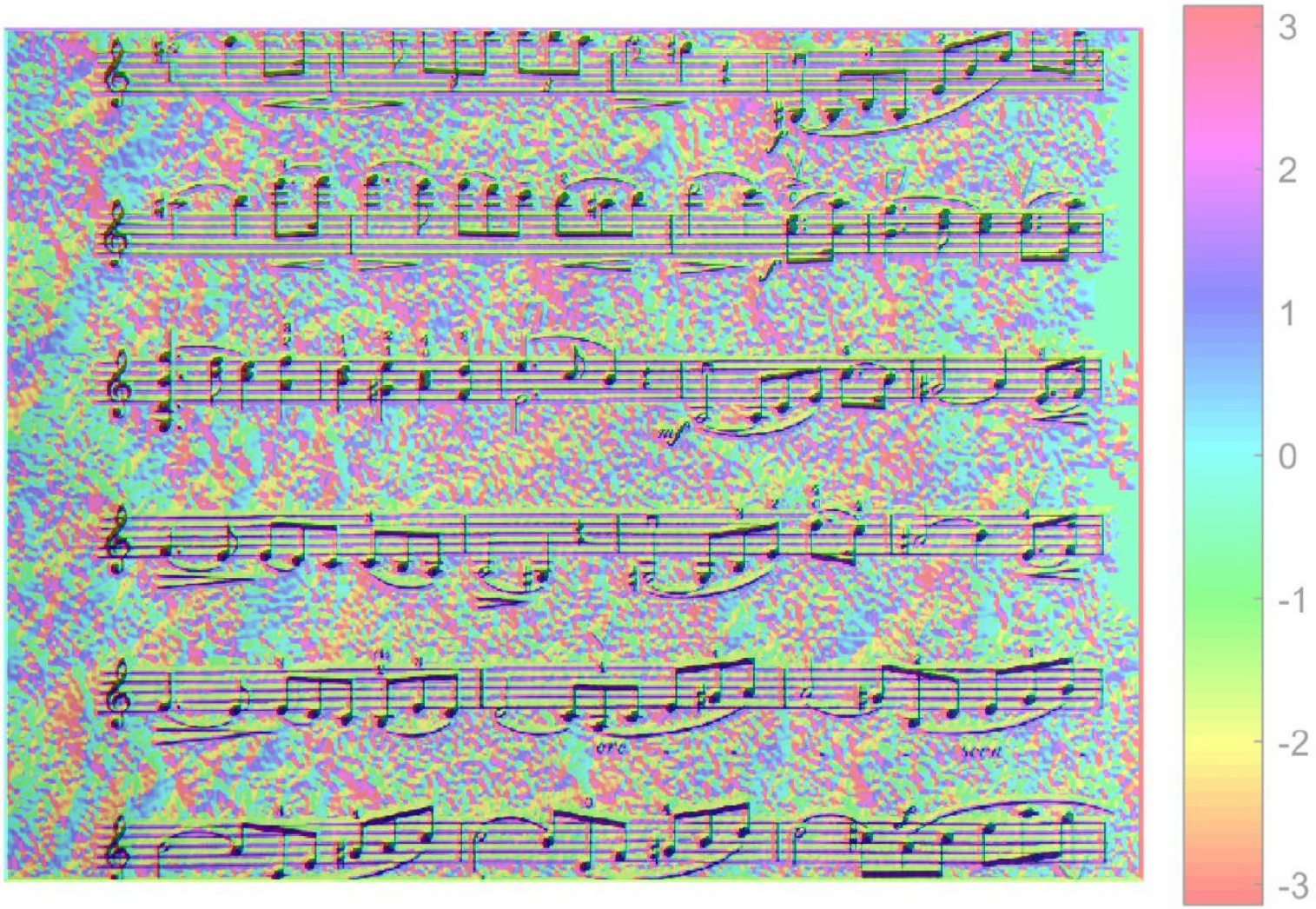
```
% % Convolution of image with Gaussian
Gx = conv2(G, dx, 'same');
Gy = conv2(G, dy, 'same');

% Convolution of image with Gx and Gy
Ix = conv2(img, Gx, 'same');
Iy = conv2(img, Gy, 'same');
```

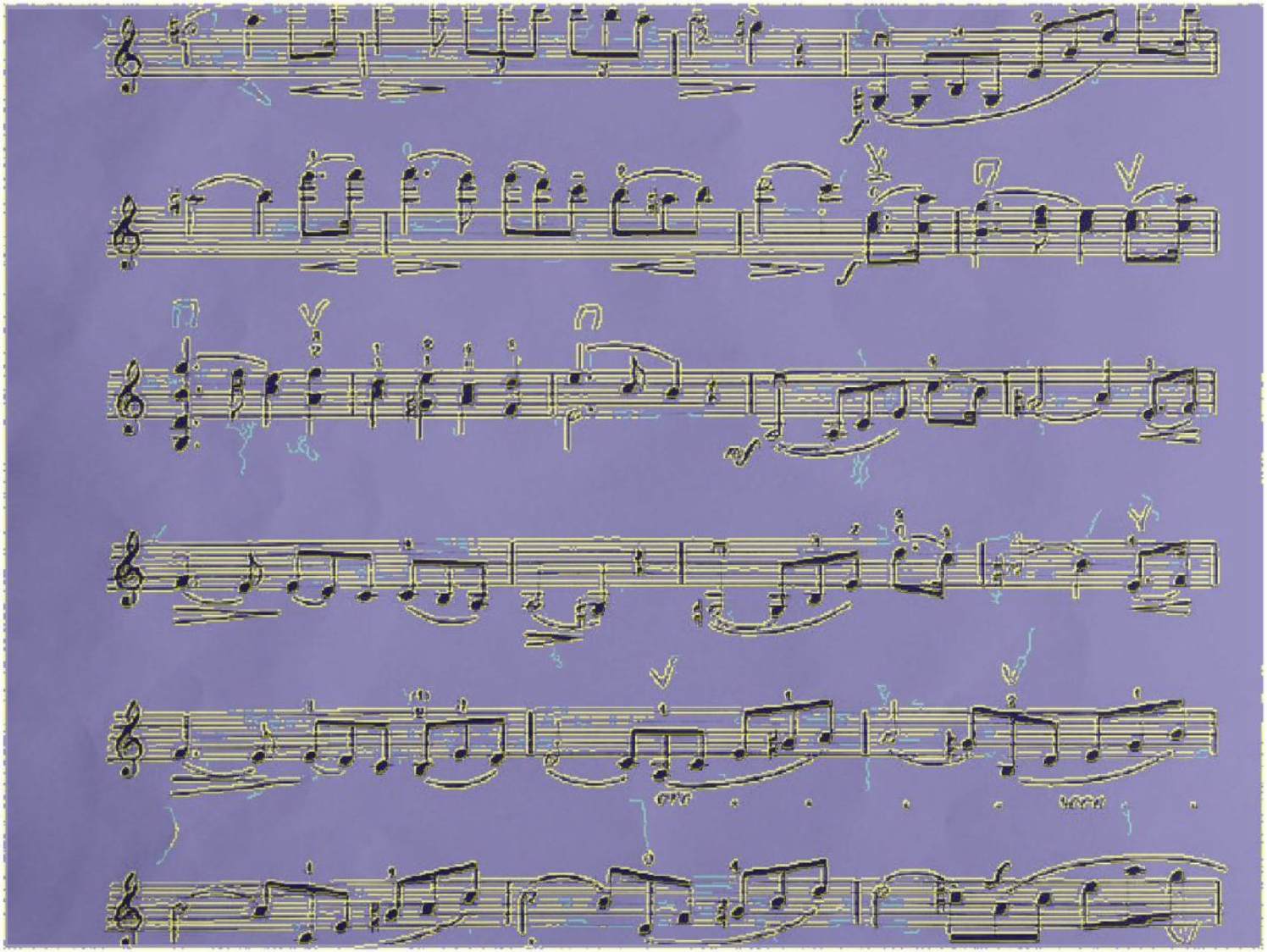




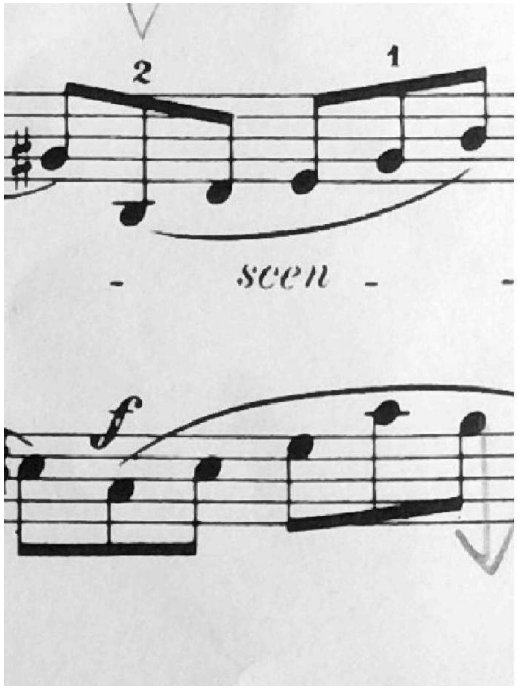


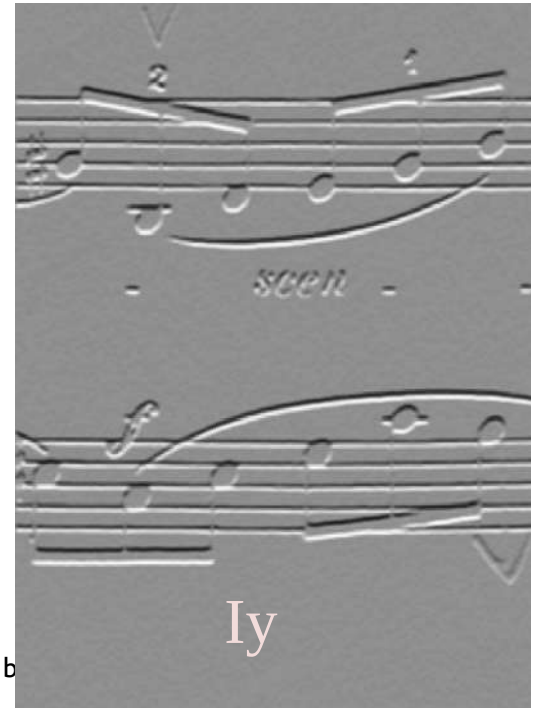


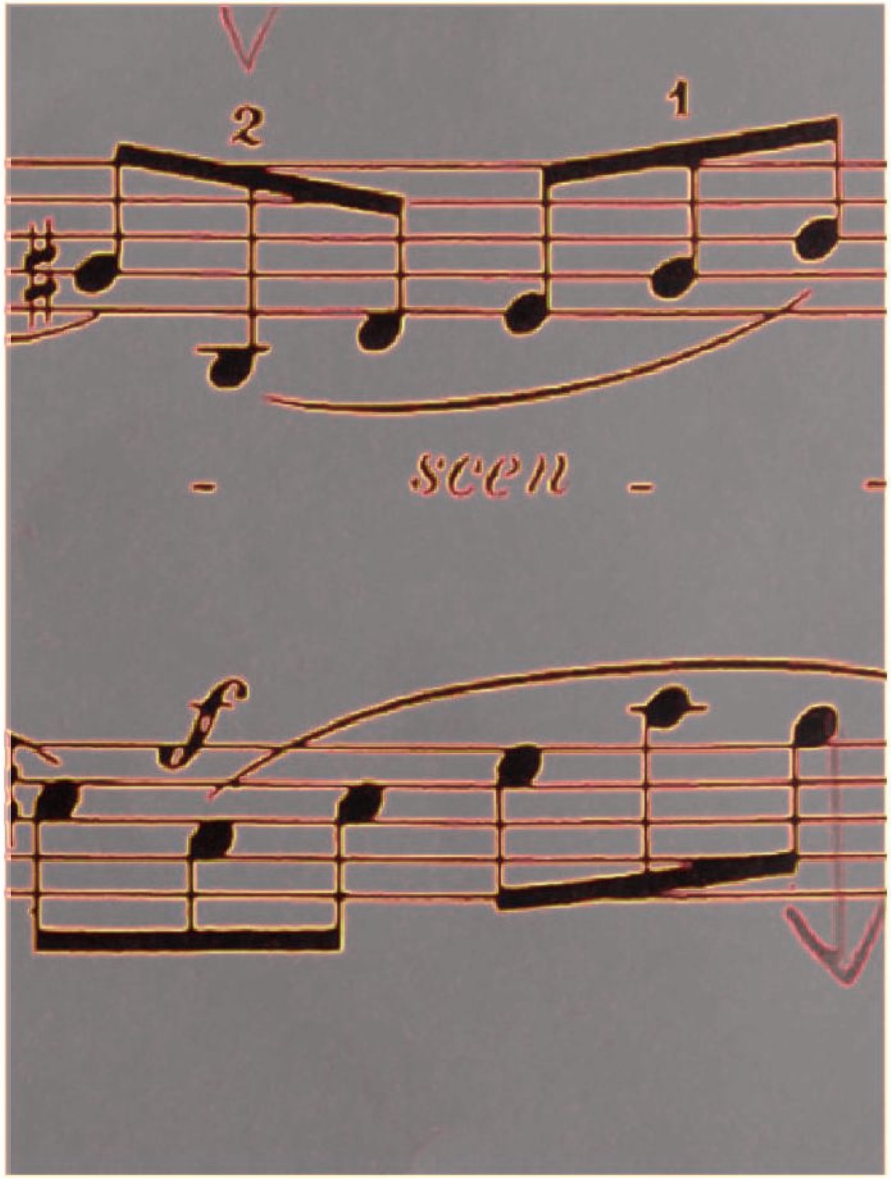


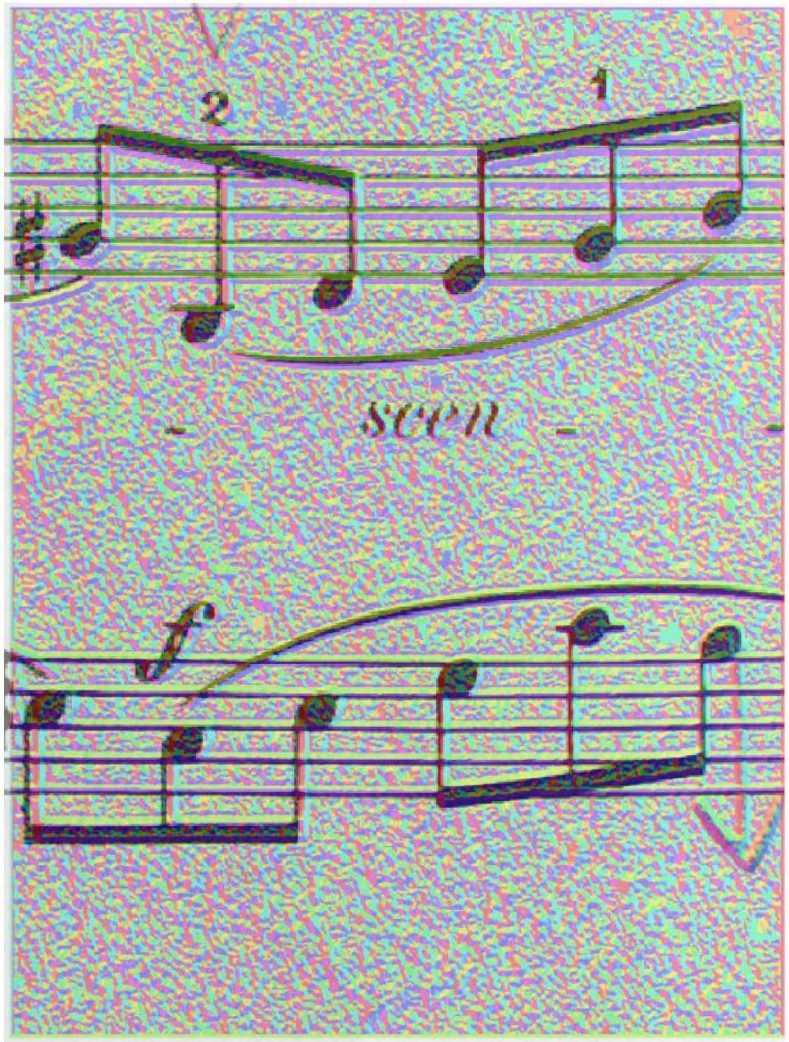


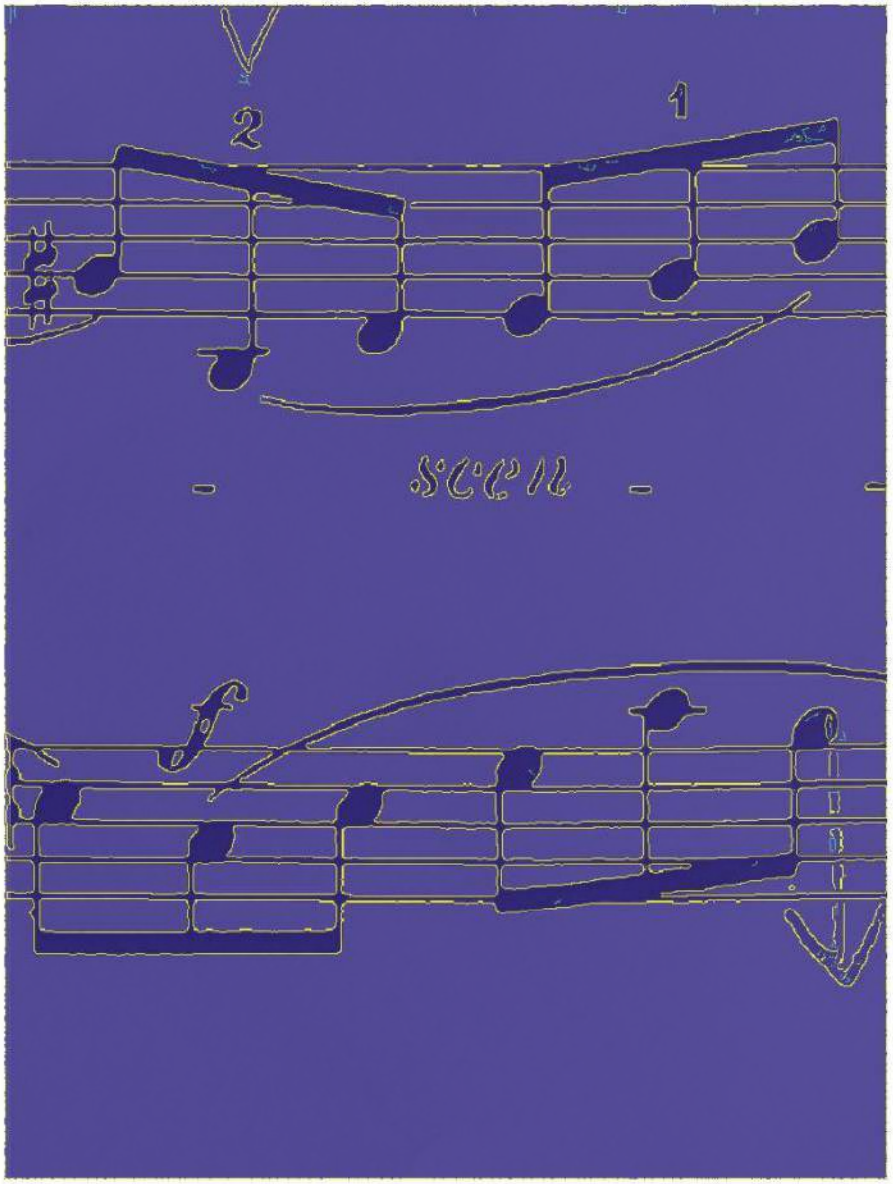
```
% % Convolution of image with Gaussian
Gx = conv2(G, dx, 'same');
Gy = conv2(G, dy, 'same');

% Convolution of image with Gx and Gy
Ix = conv2(img, Gx, 'same');
Iy = conv2(img, Gy, 'same');
```

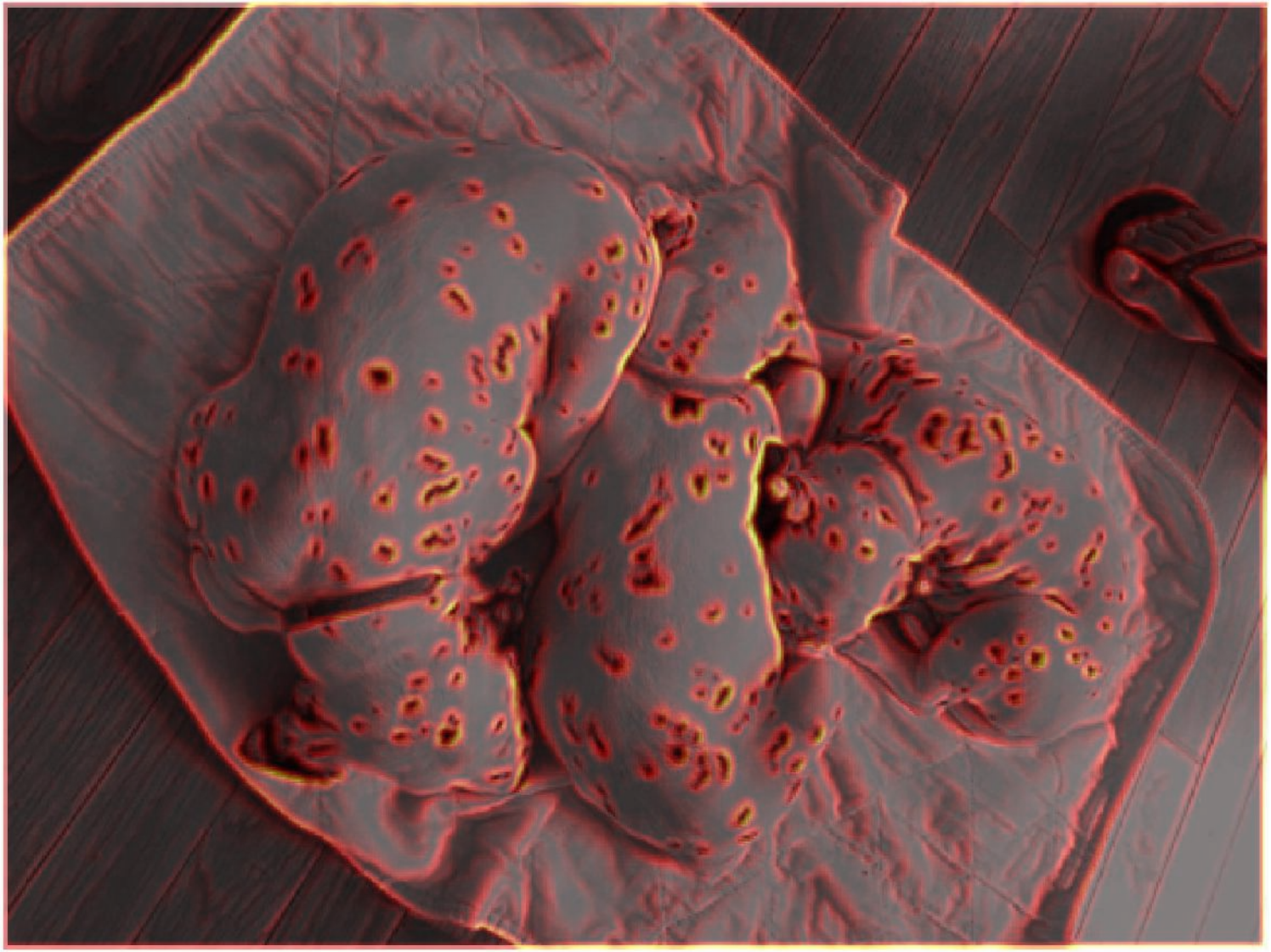


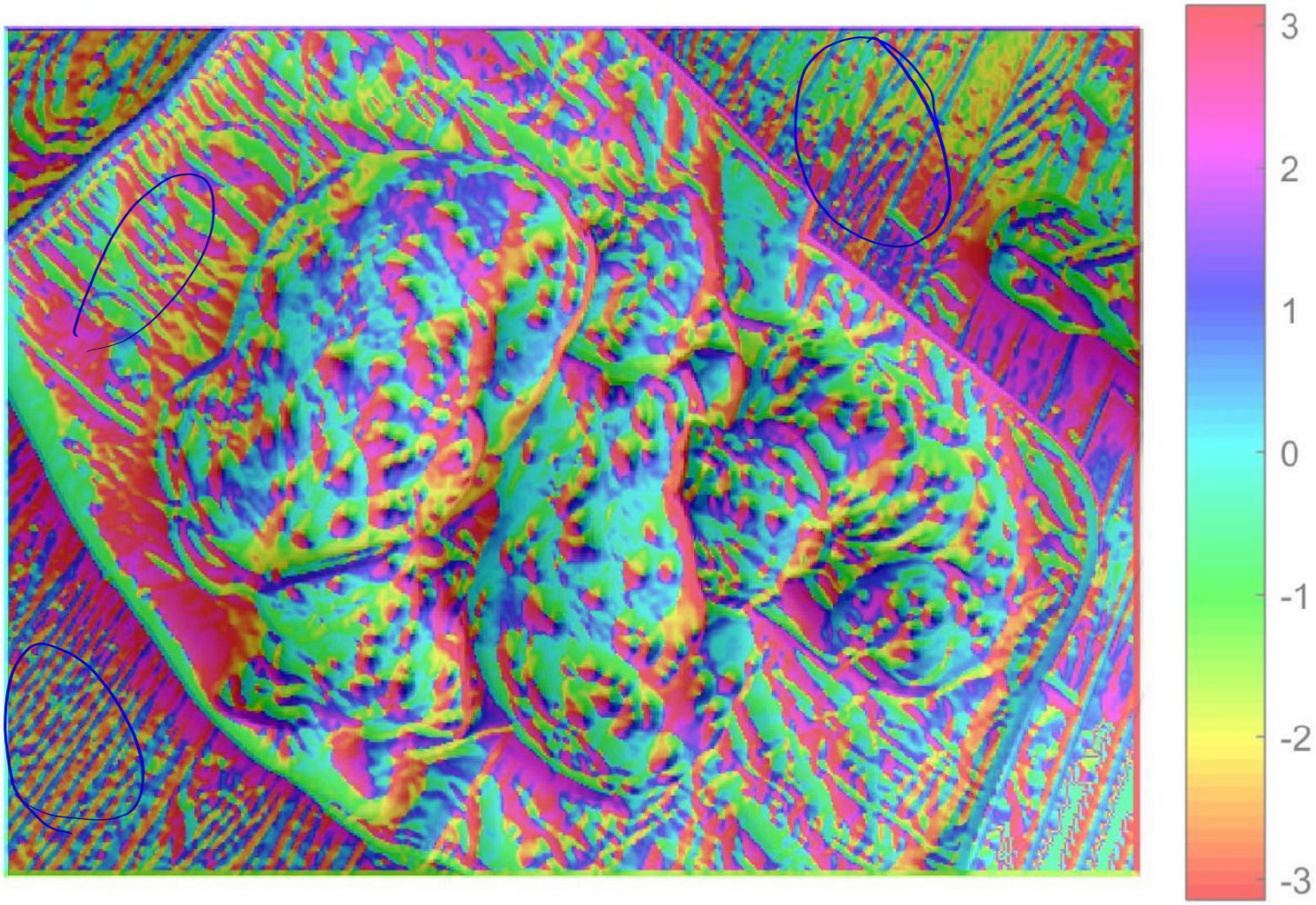


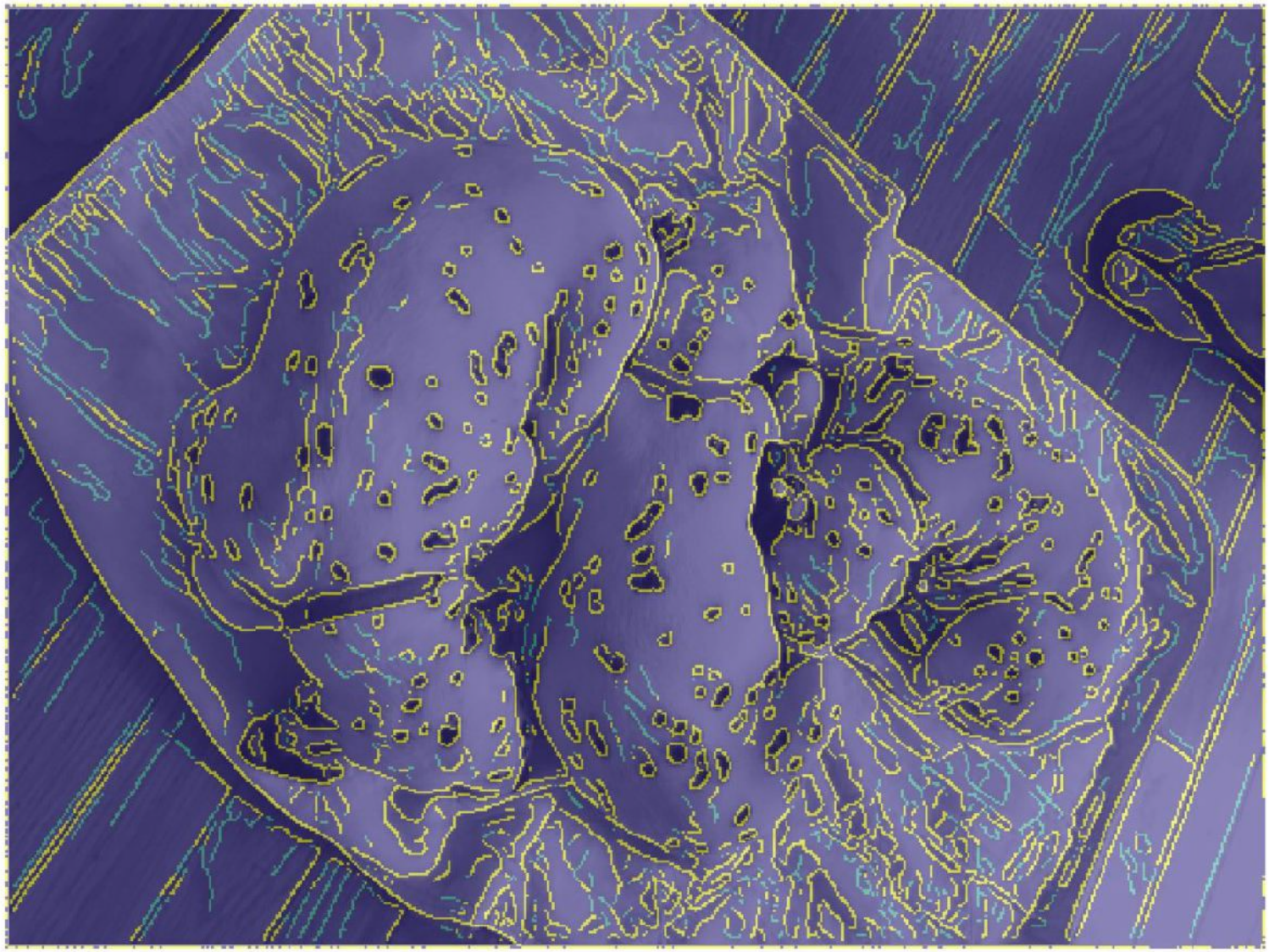


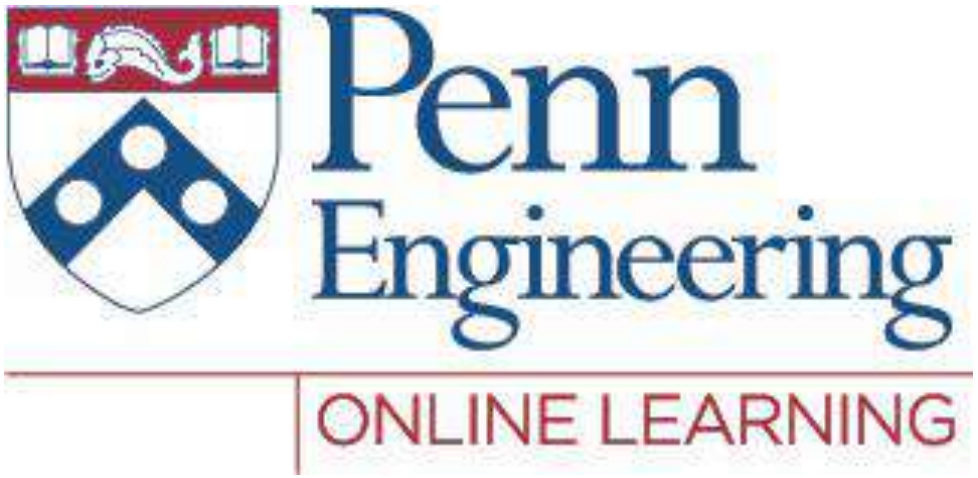






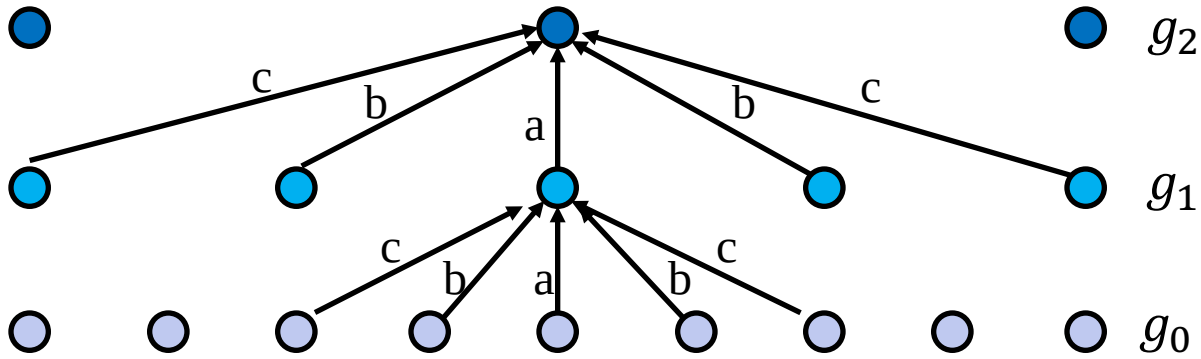






Video 3.7
Jianbo Shi

Image Reduce

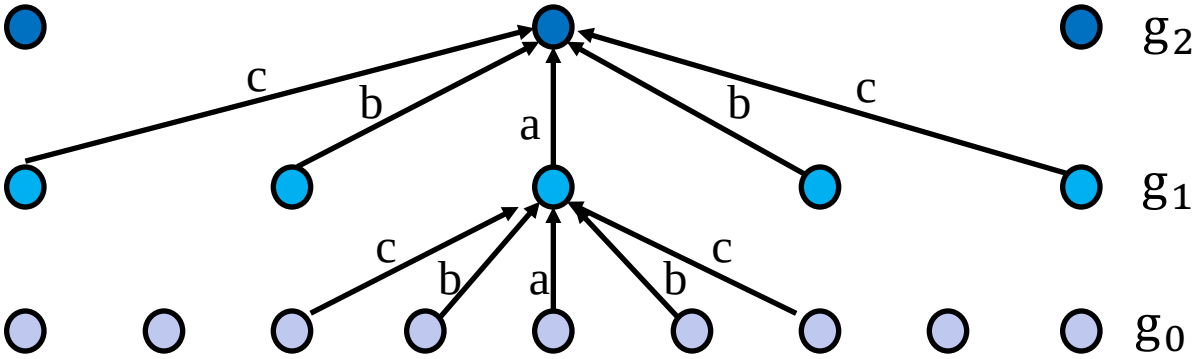


$$g_0 = \text{Image}$$

$$g_l = \text{REDUCE}[g_{l-1}]$$

[Burt & Adelson, 1983]

Image Reduce



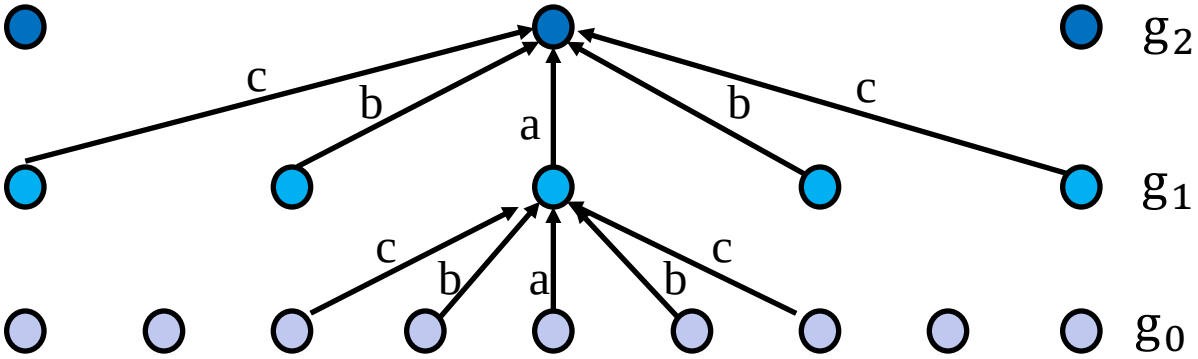
$$g_0 = \text{Image}$$

$$g_l = \text{REDUCE}[g_{l-1}]$$

$$g_l(i,j) = \sum_{m=-2}^{2} \sum_{n=-2}^{2} w(m,n)\, g_{l-1}(2i+m, 2j+n)$$

$$w(m,n) = \hat{w}(m)\hat{w}(n) \qquad \sum_{m}^{2} \hat{w}(m) = 1 \qquad \hat{w}(m) = \hat{w}(-m)$$

Image Reduce



$$g_0 = \text{Image}$$

$$g_l = \text{REDUCE}[g_{l-1}]$$

$$g_l(i,j) = \sum_{m=-2}^{2} \sum_{n=-2}^{2} w(m,n)\, g_{l-1}(2i+m, 2j+n)$$

$$g_l = [g_{l-1} \otimes w] \downarrow 2$$

Choice in weighting function

$$\hat{w}(0) = a$$

$$\hat{w}(1) = \hat{w}(-1) = {}^{1}/_{4}$$

$$\hat{w}(1) = \hat{w}(-1) = {}^{1}/_{4} - {}^{a}/_{2}$$

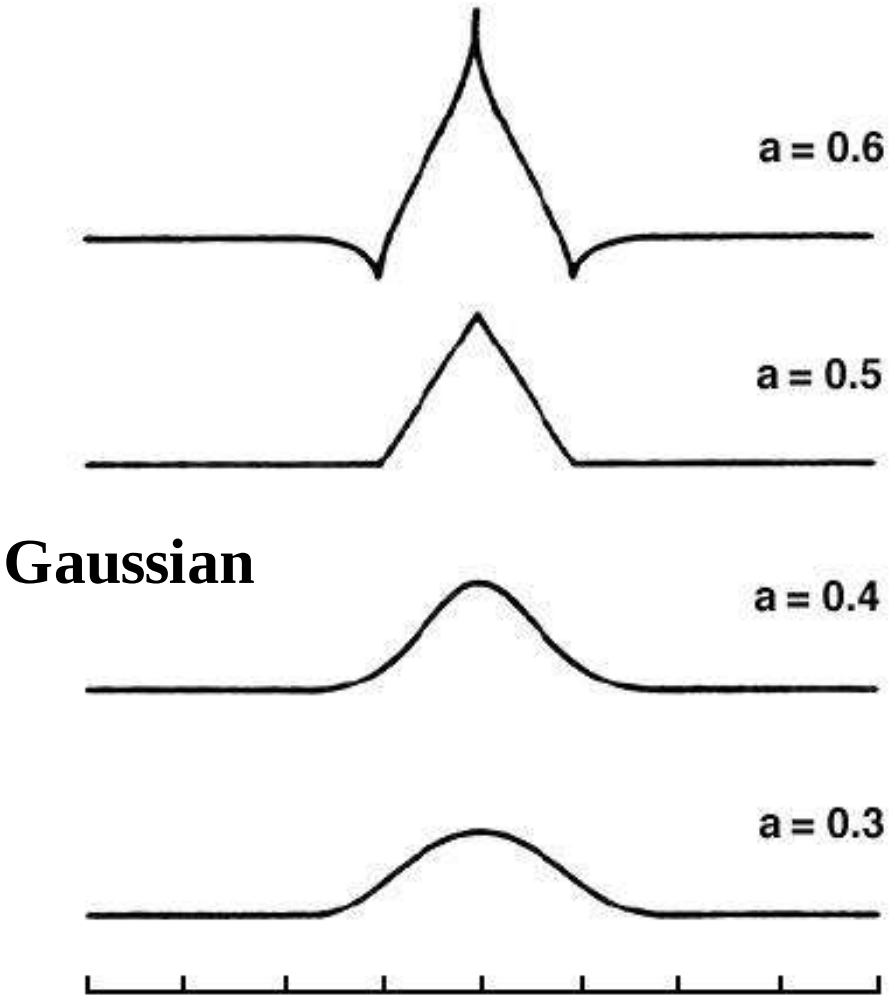
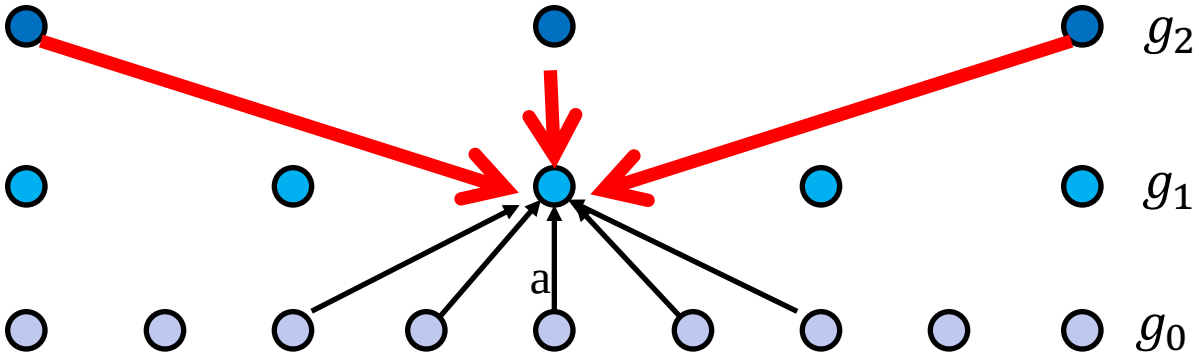
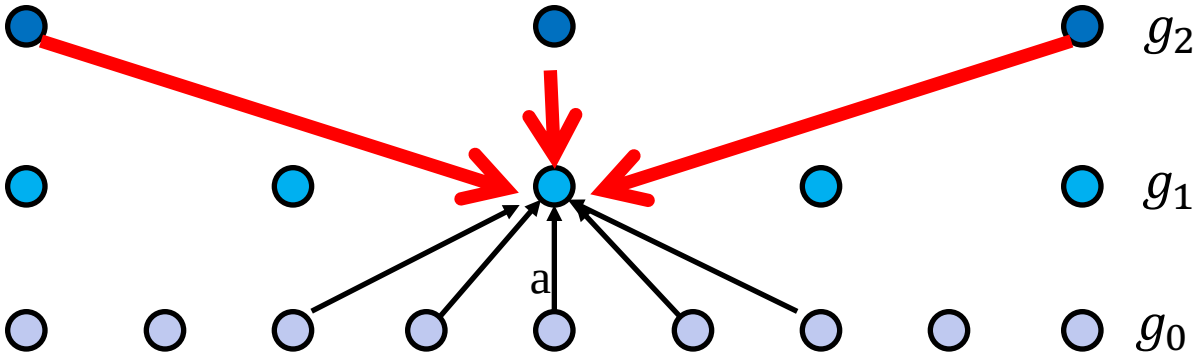


Image Expansion



$$g_{l-1} = \text{EXPAND}[g_l]$$

Image Expansion

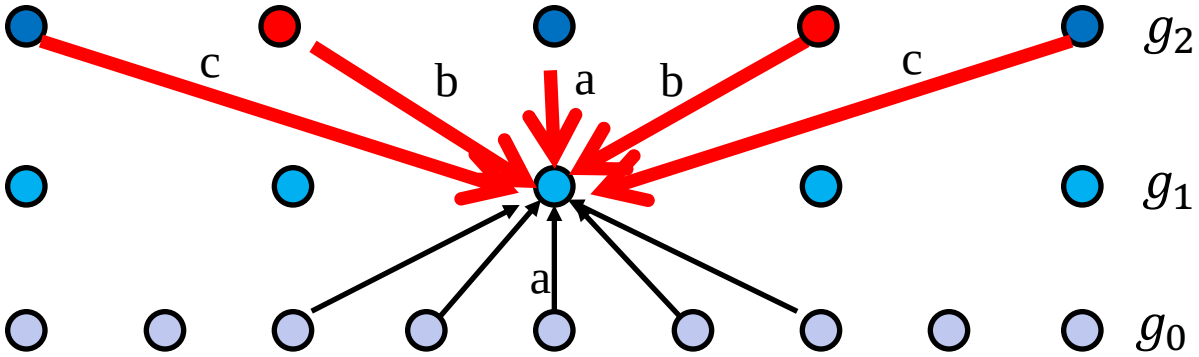


$$g_{l-1} = \text{EXPAND}[g_l]$$

$$g_l(i,j) = 4\sum_{m=-2}^{2}\sum_{n=-2}^{2} w(m,n)\cdot g_{l+1}\left(\frac{i-m}{2},\frac{j-n}{2}\right)$$

$^{(i-m)}/_{2}$ and $^{(j-n)}/_{2}$ are integers

Image Expansion

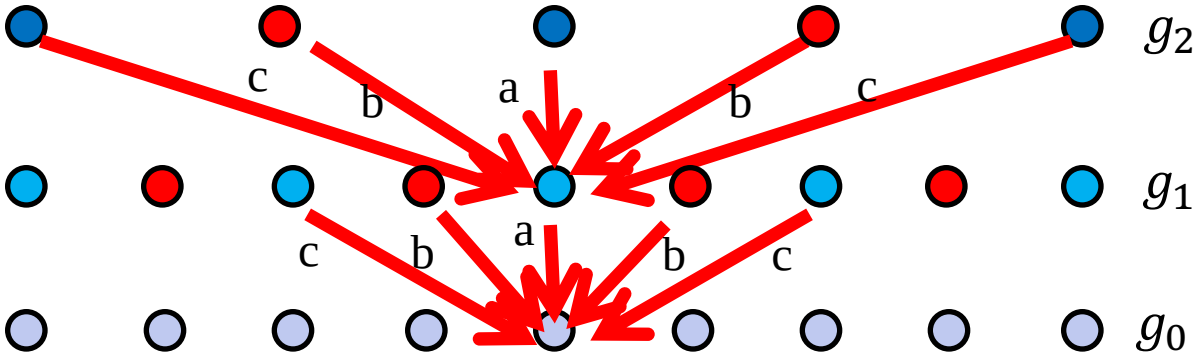


$$g_{l-1} = \text{EXPAND}[g_l]$$

$$g_l(i,j) = 4 \sum_{m=-2}^{2} \sum_{n=-2}^{2} w(m,n) \cdot g_{l+1}\left(\frac{i-m}{2}, \frac{j-n}{2}\right)$$

$^{(i-m)}/_{2}$ and $^{(j-n)}/_{2}$ are integers

Image Expansion

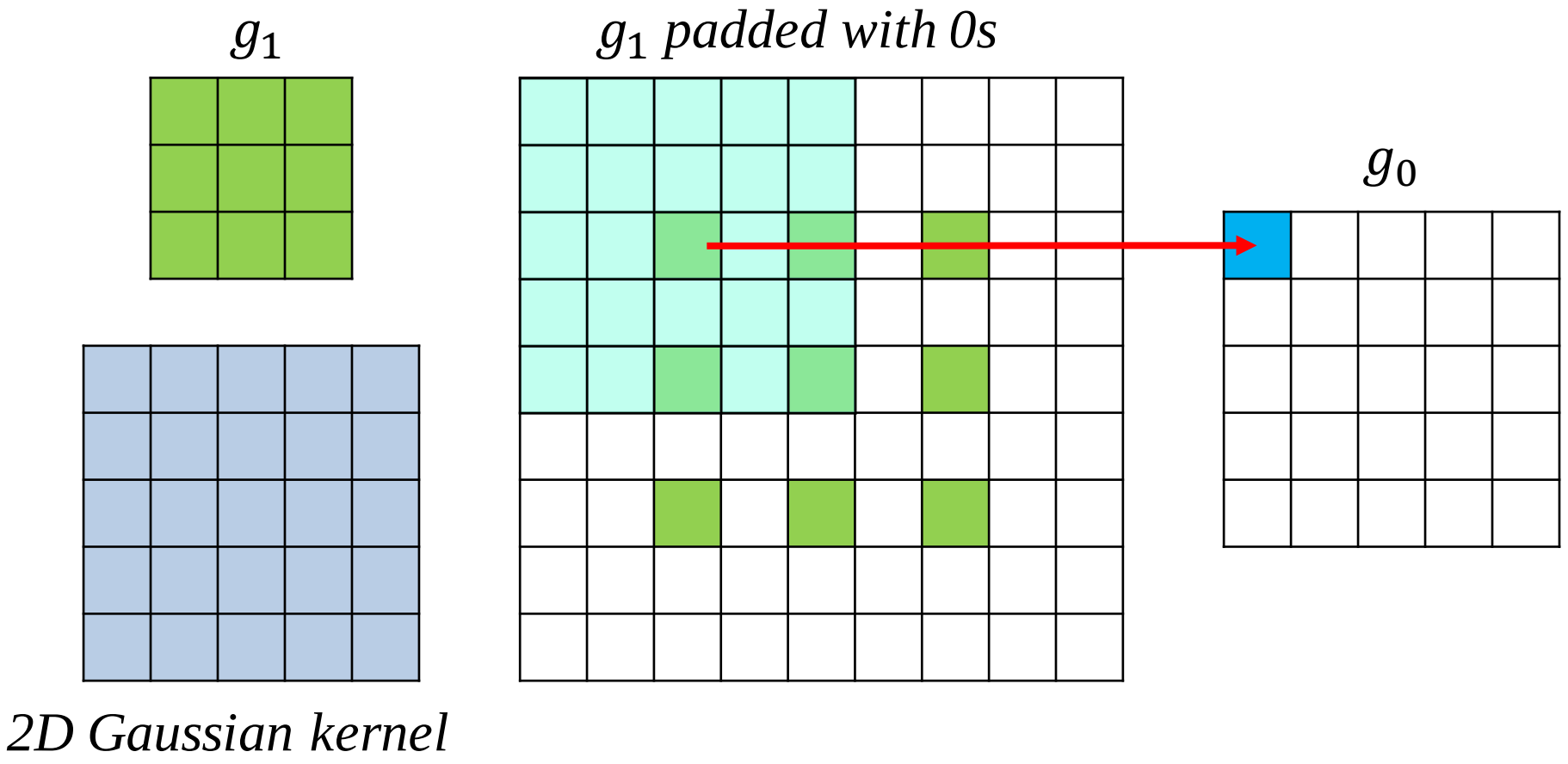


$$g_{l-1} = \text{EXPAND}[g_l]$$

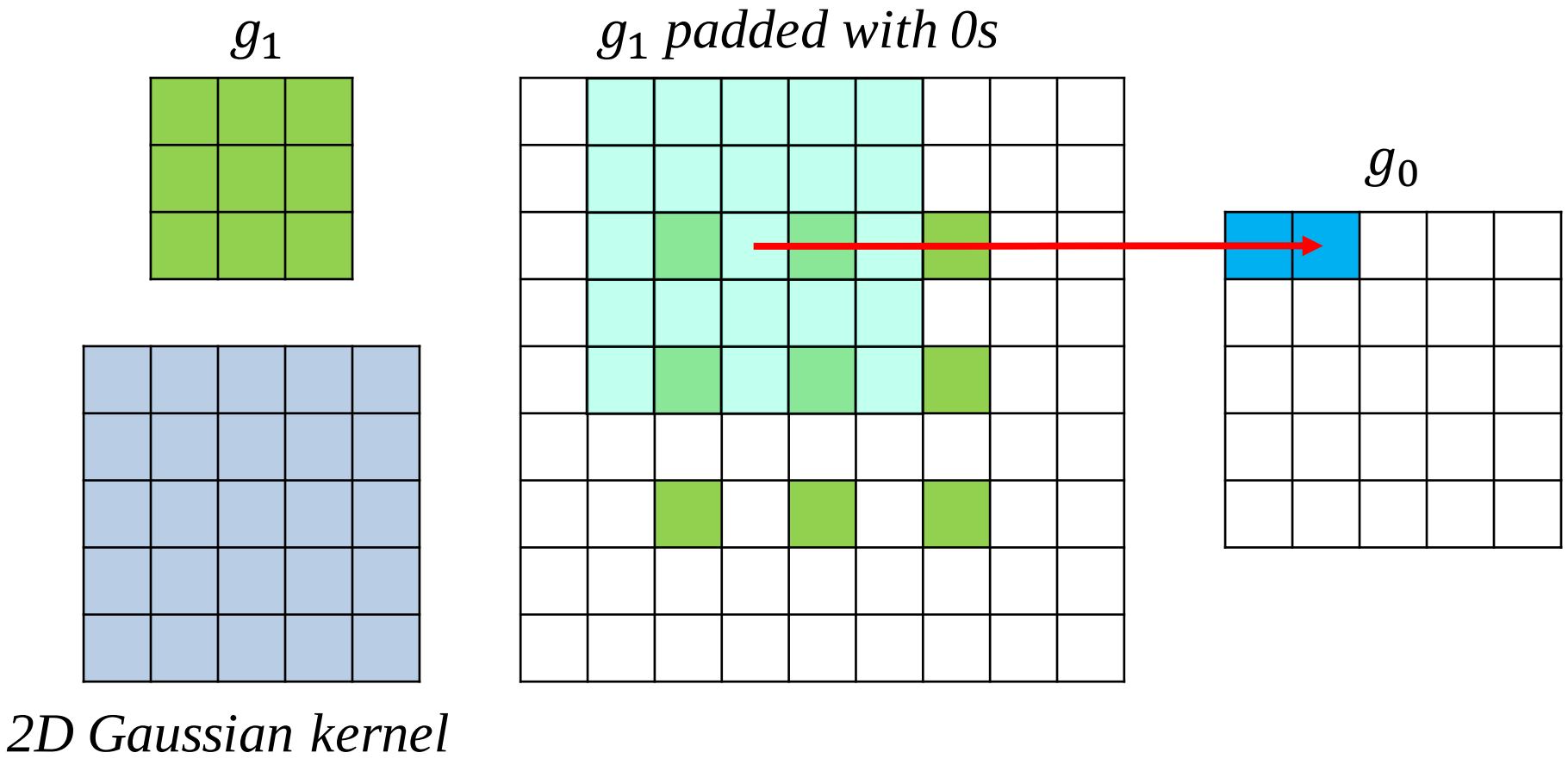
$$g_l(i,j) = 4 \sum_{m=-2}^{2} \sum_{n=-2}^{2} w(m,n) \cdot g_{l-1}\left(\frac{i-m}{2}, \frac{j-n}{2}\right)$$

$^{(i-m)}/_{2}$ and $^{(j-n)}/_{2}$ are integers

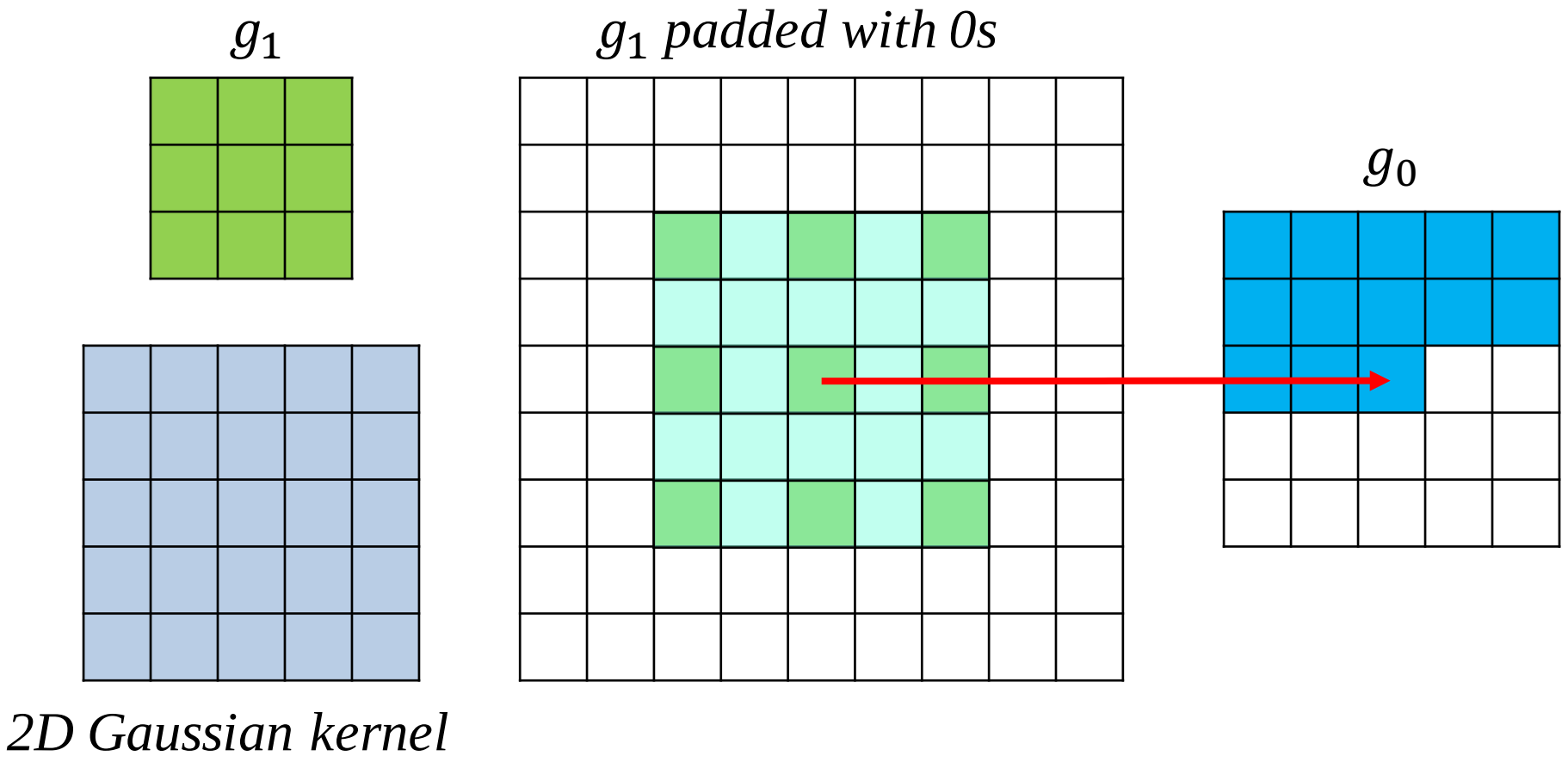
2D Image Expansion (part1)



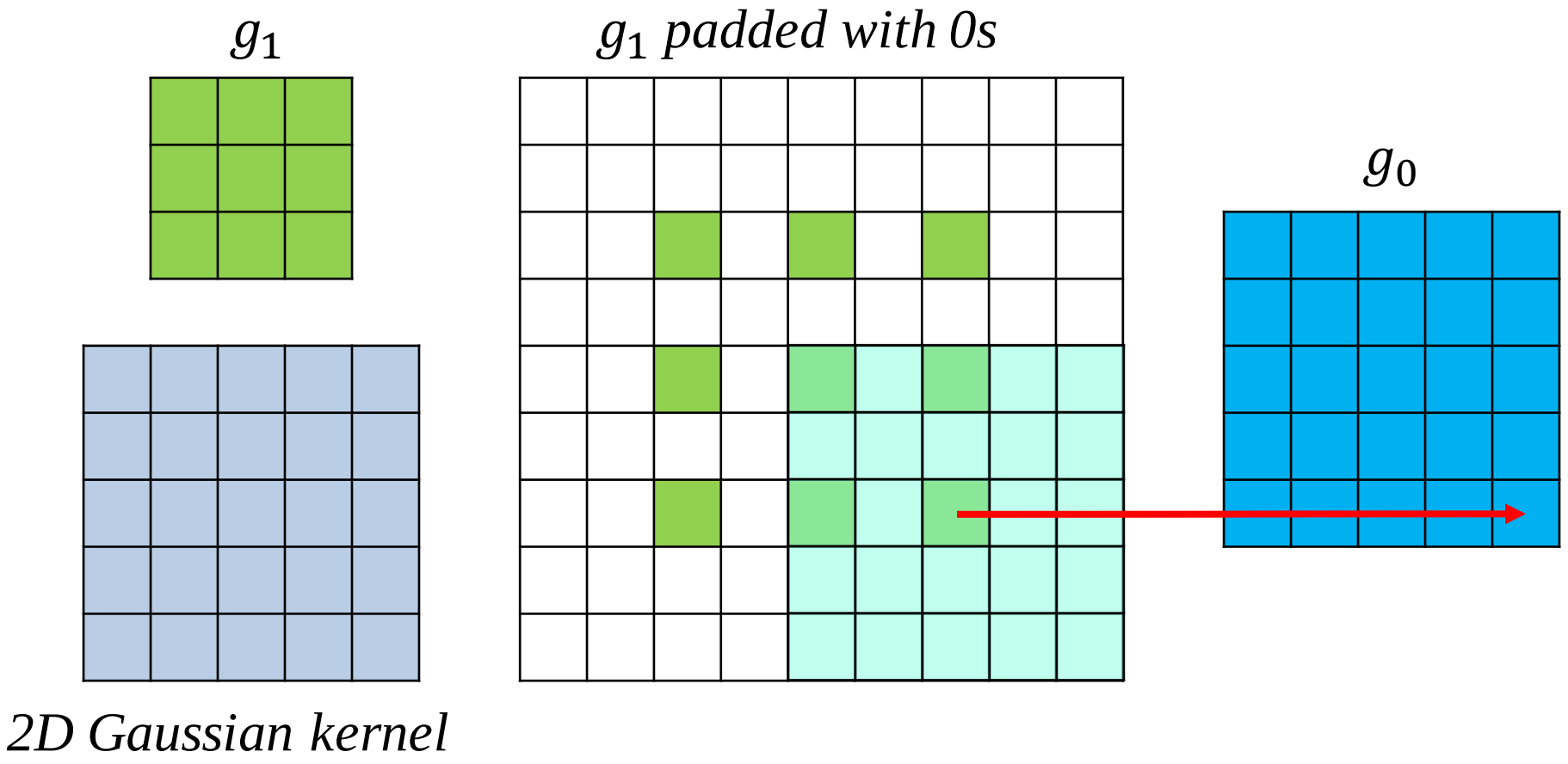
2D Image Expansion (part2)

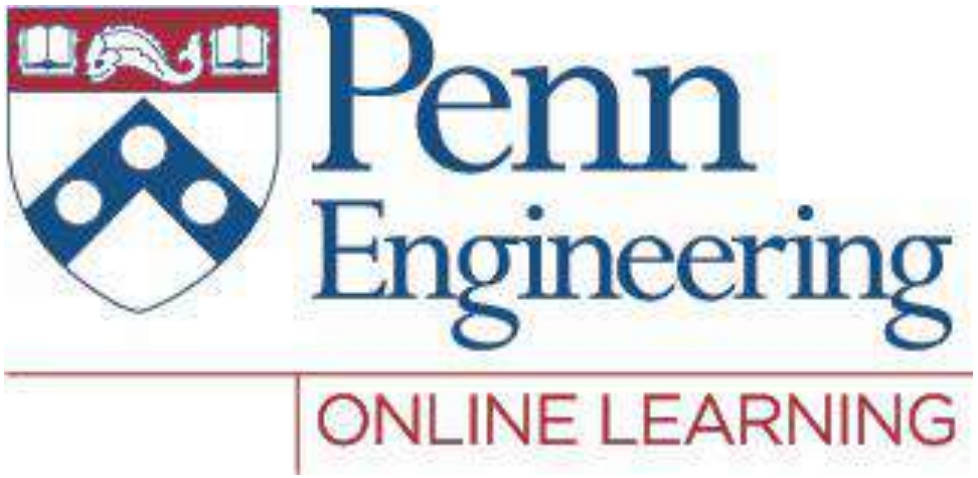


2D Image Expansion (part3)



2D Image Expansion (part4)



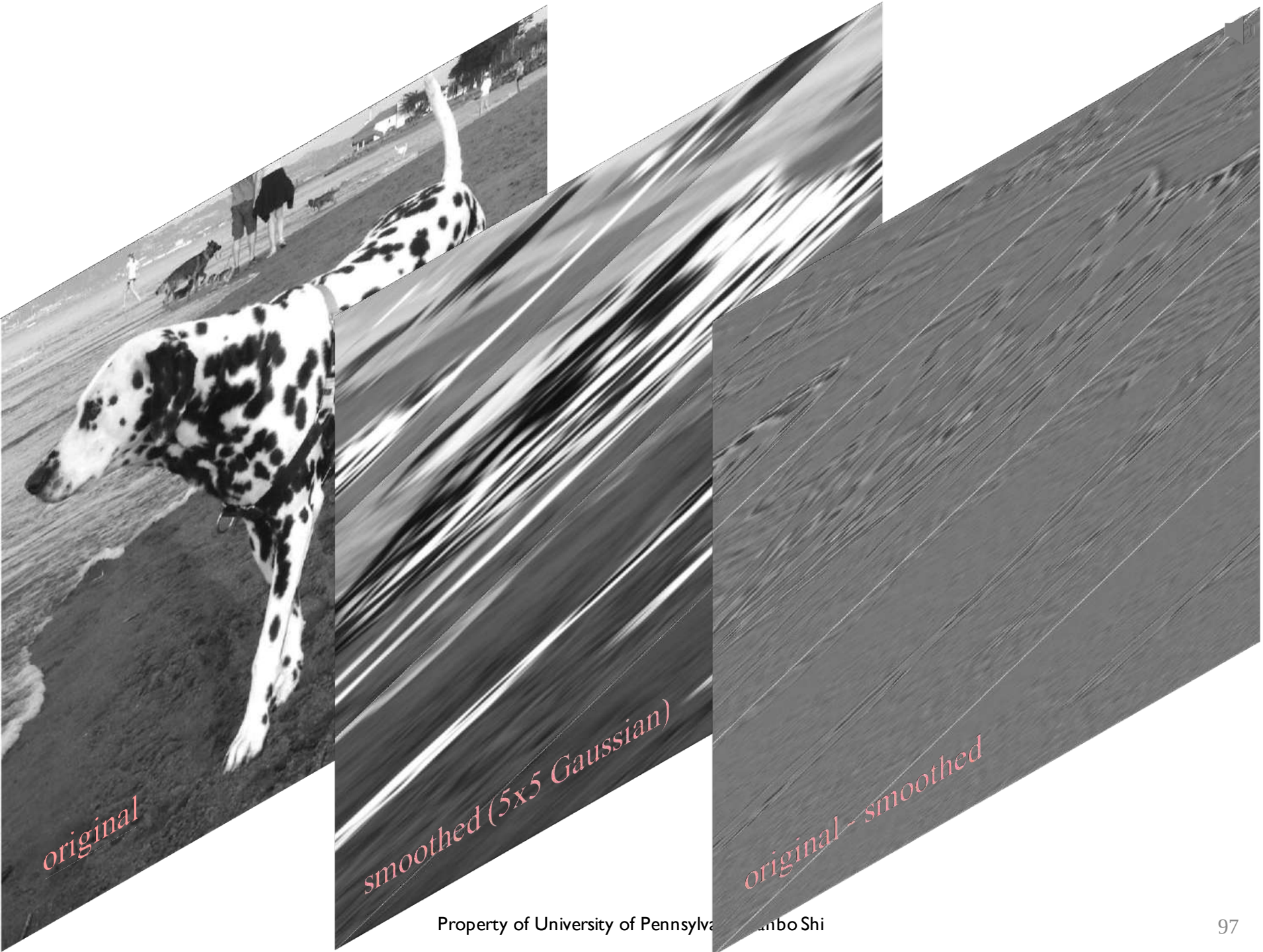


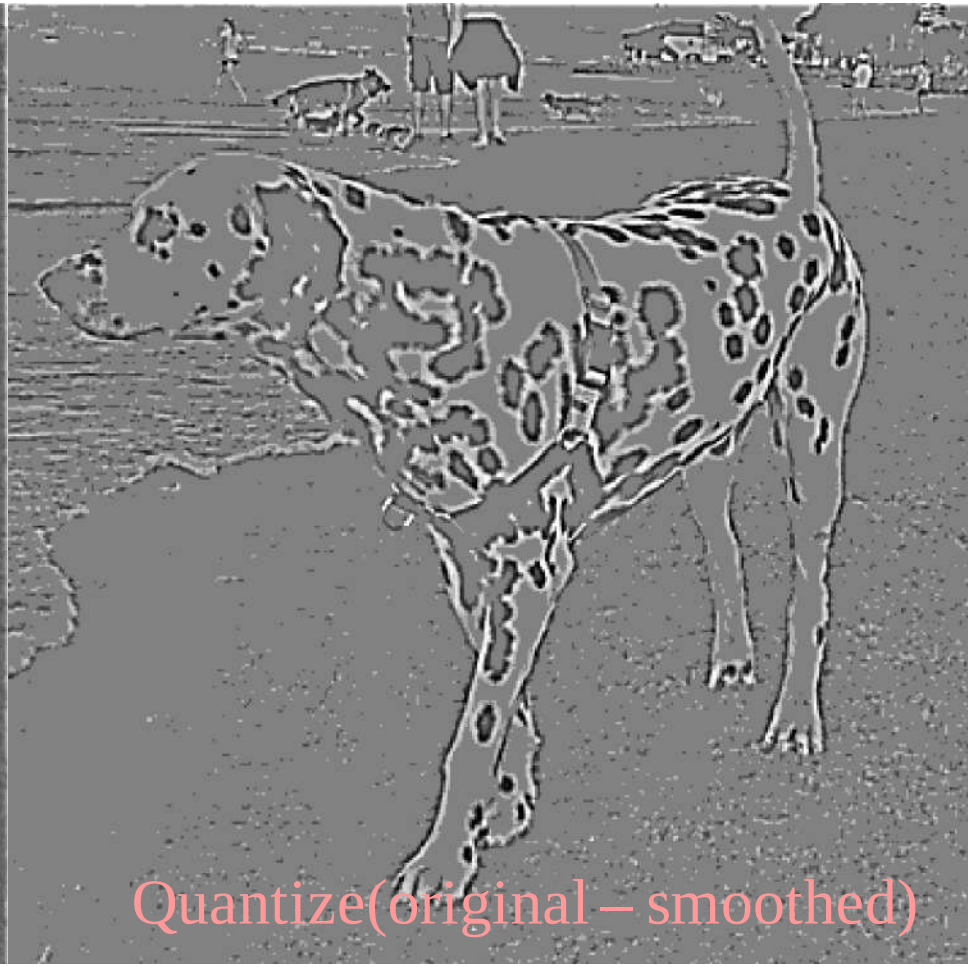
Video 3.8
Jianbo Shi

What does blurring take away?

What does blurring take away?

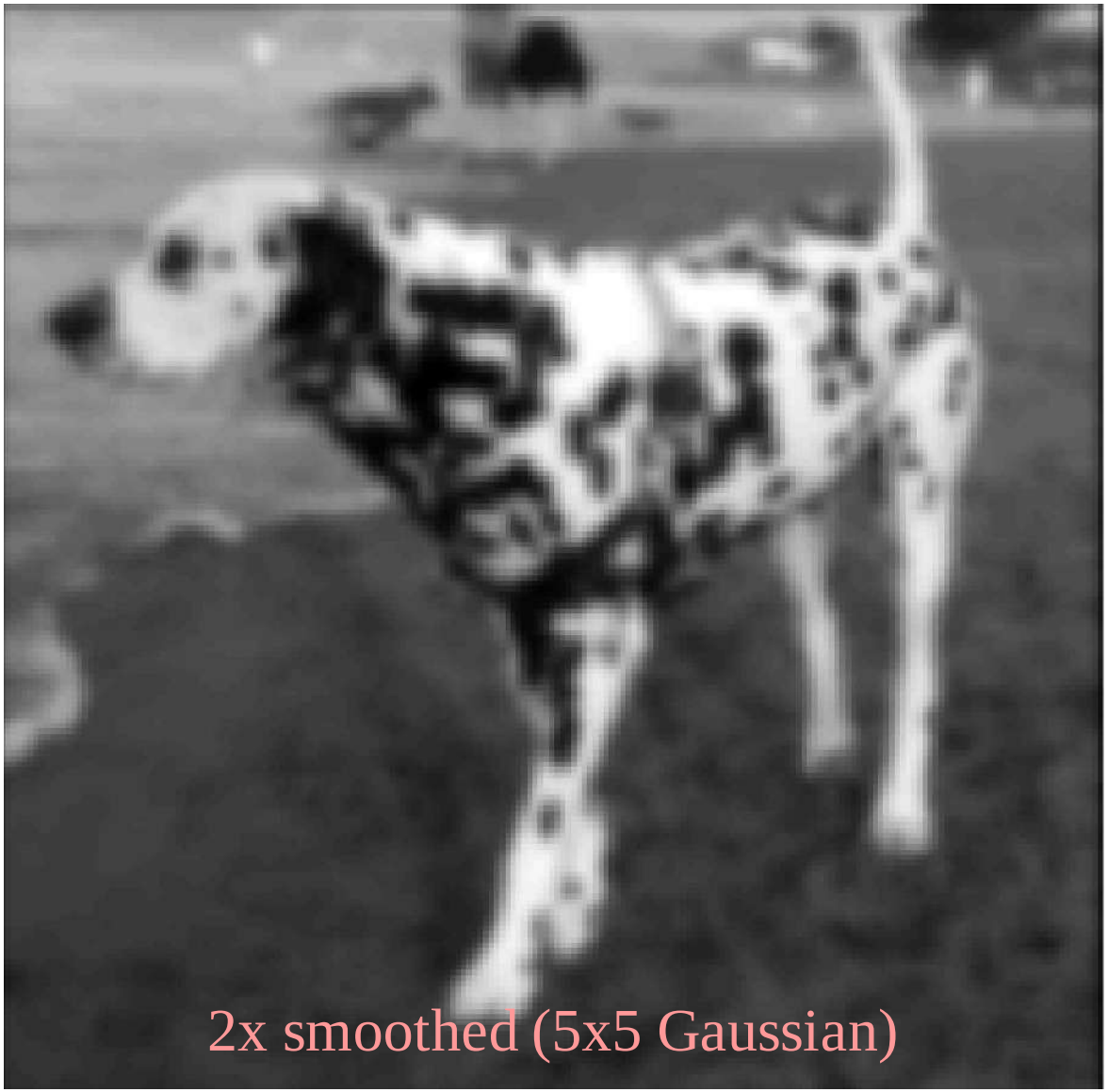
Difference as result of smoothing


original
smoothed (5x5 Gaussian)
original - smoothed

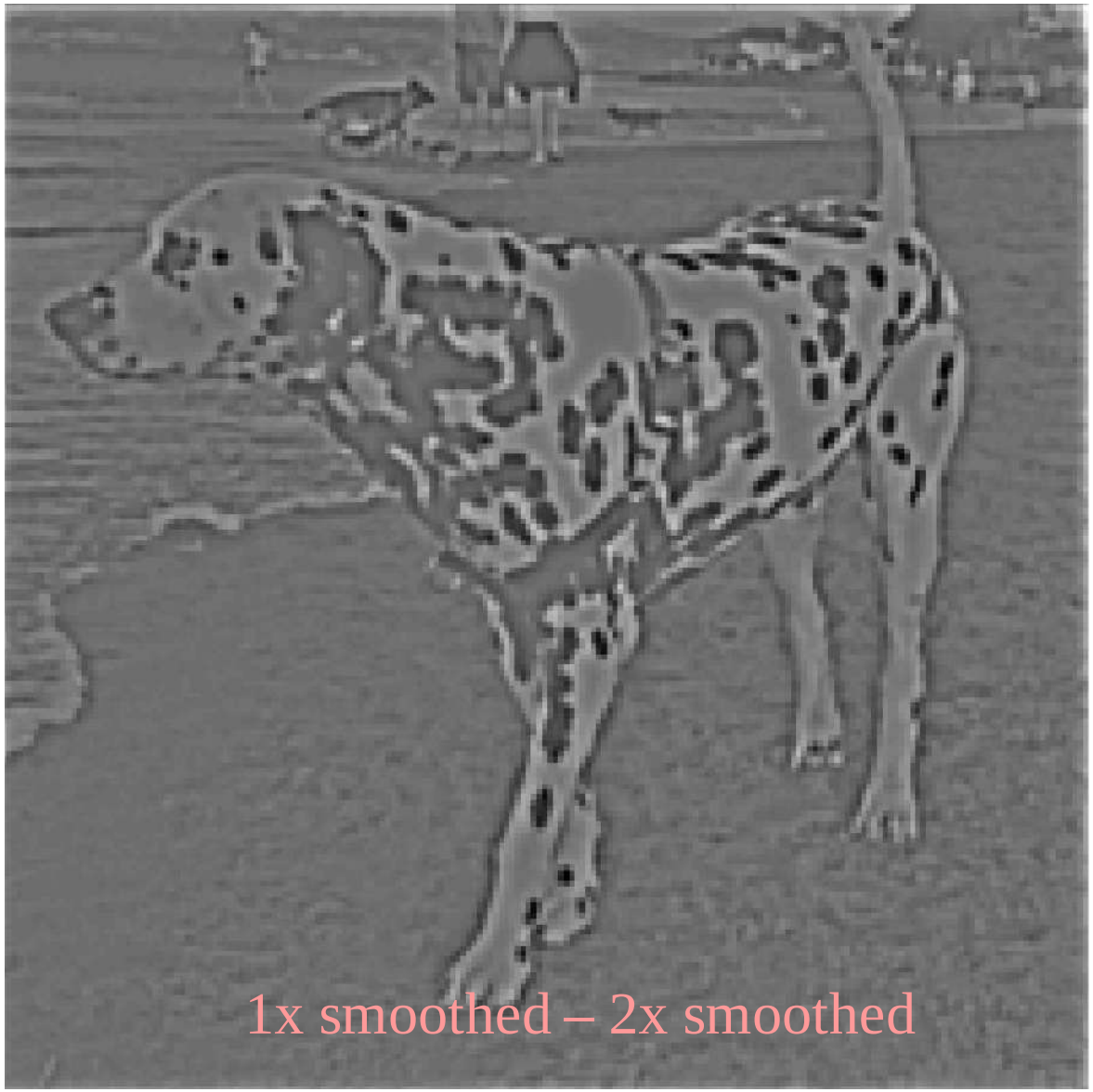


What does blurring take away?

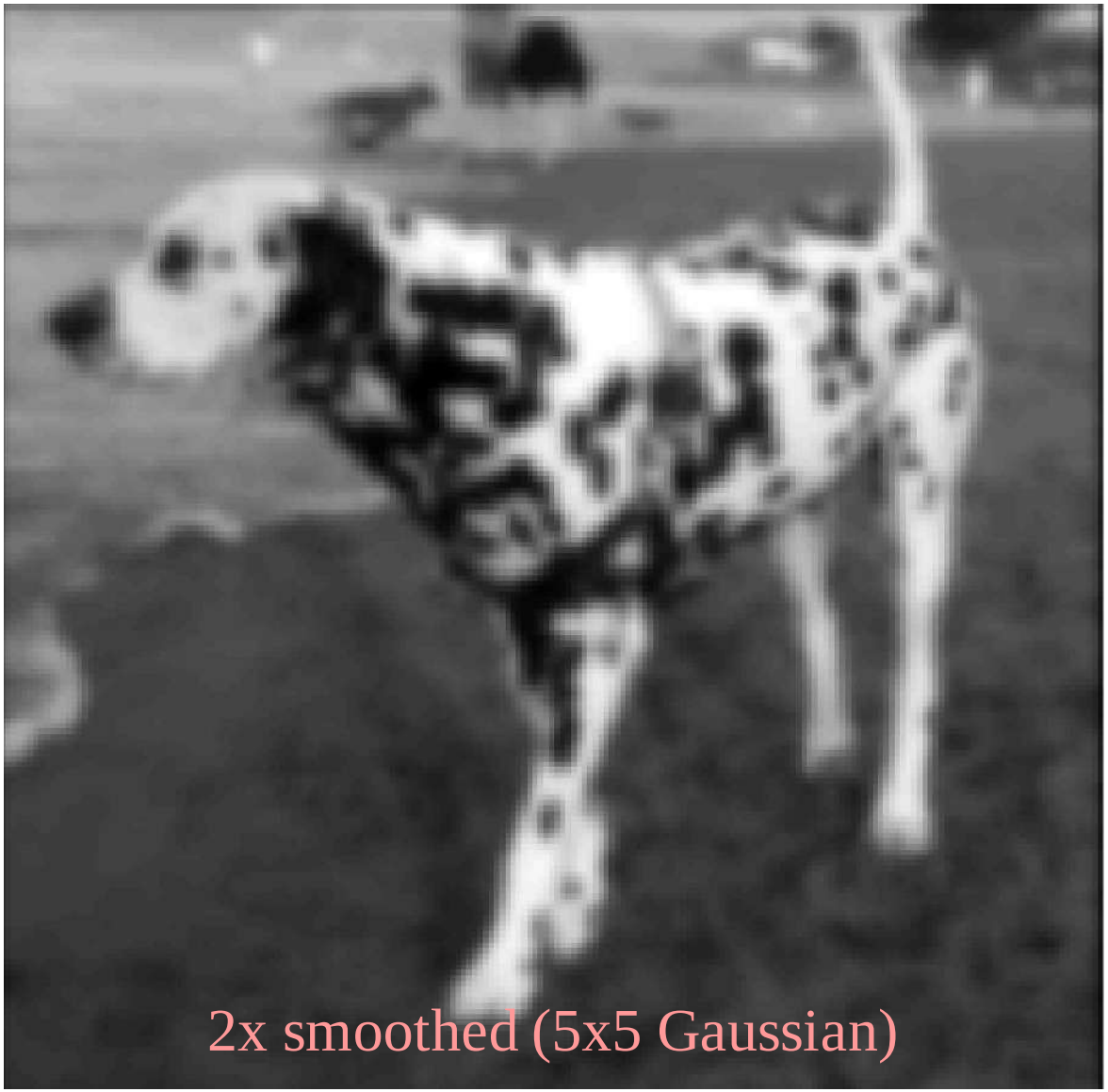
What does blurring take away?

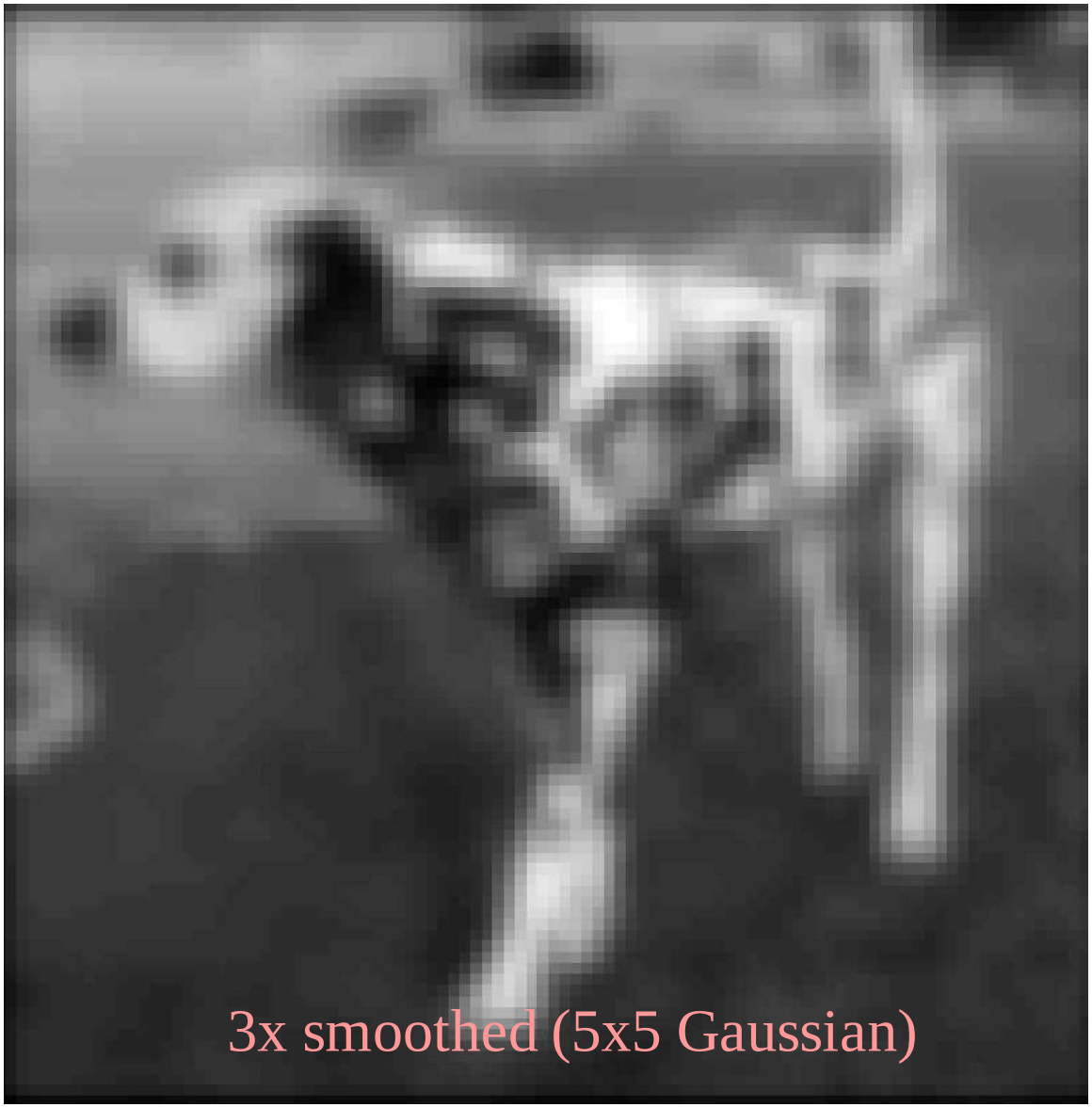


Difference of Gaussian

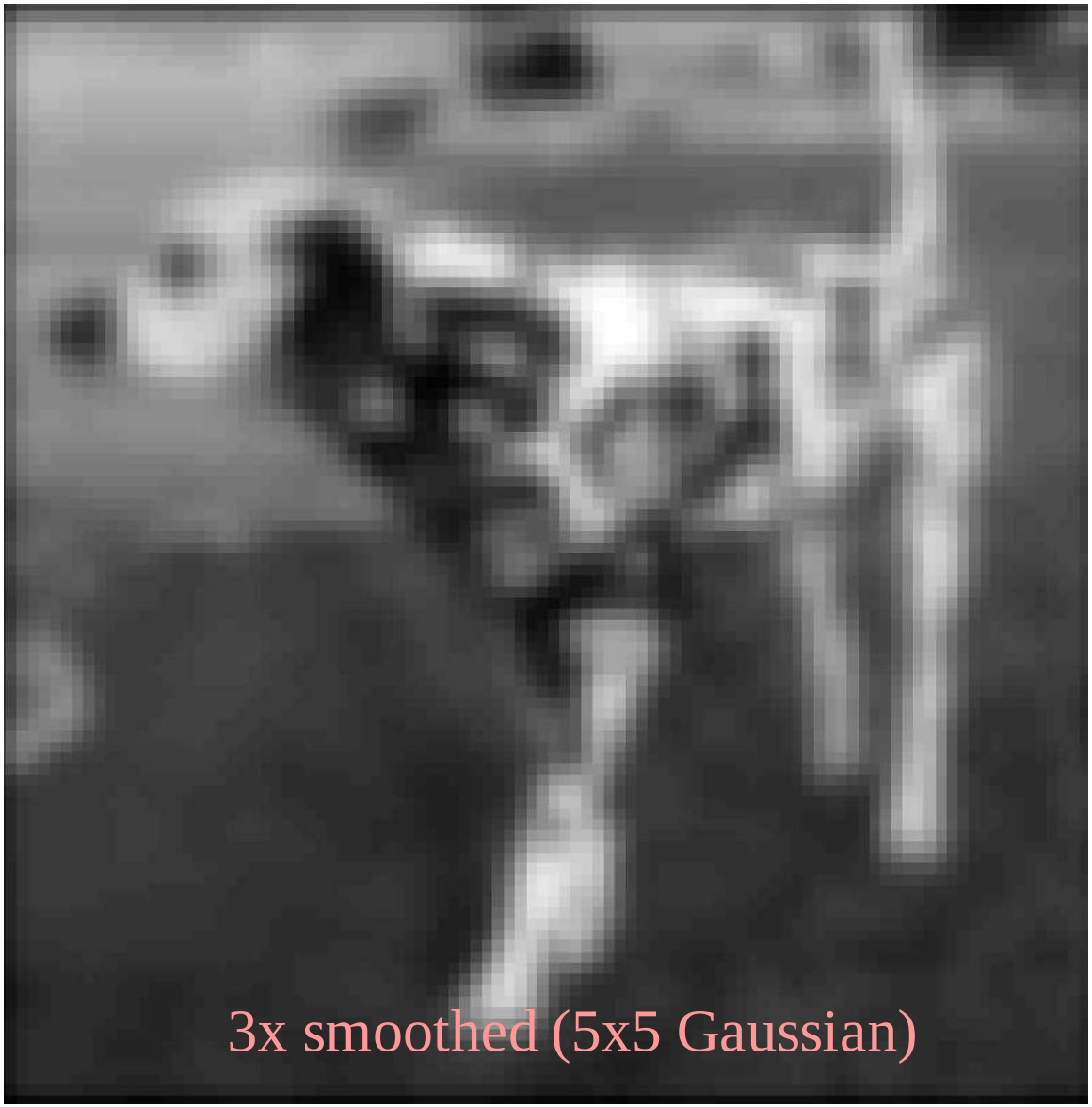


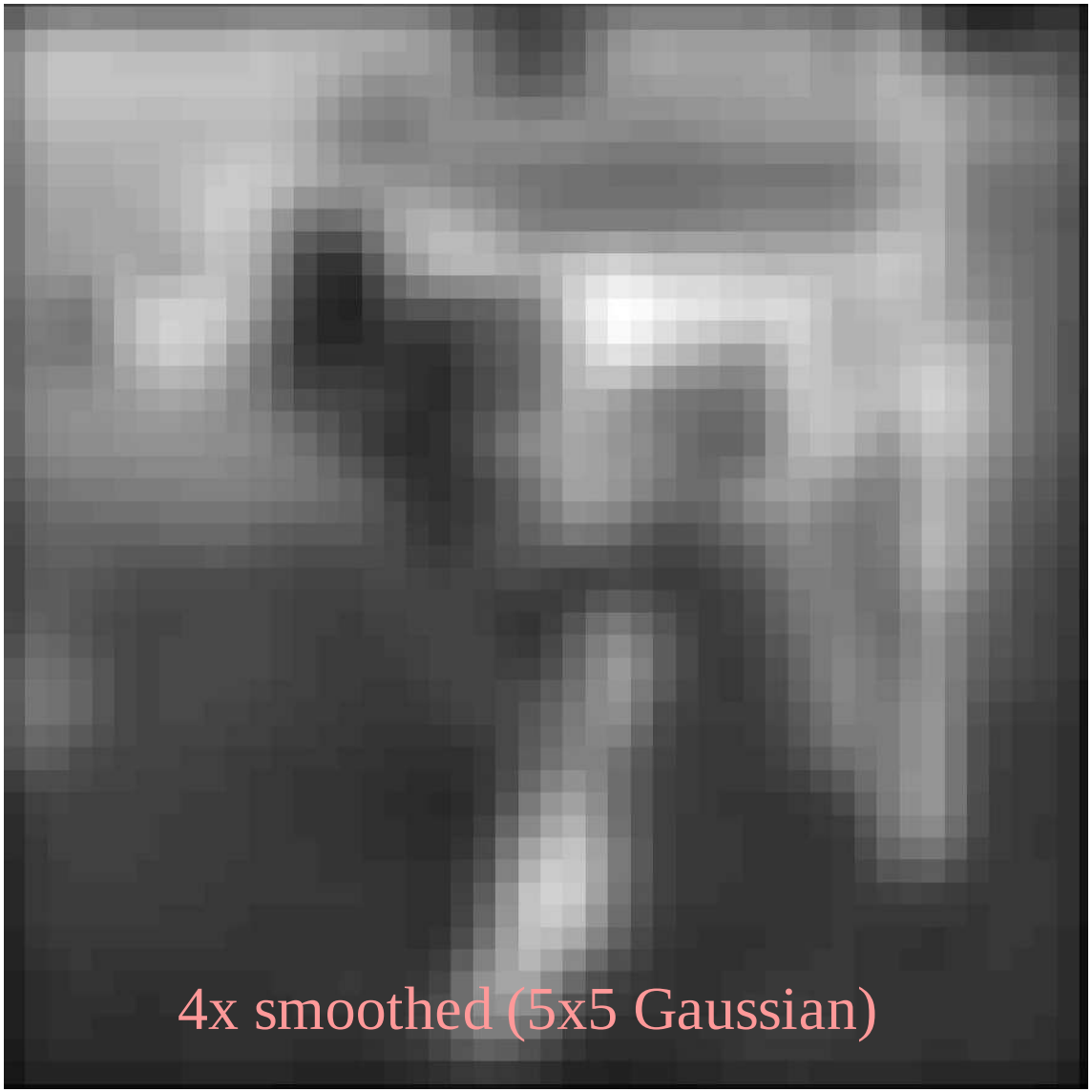
What does blurring take away?

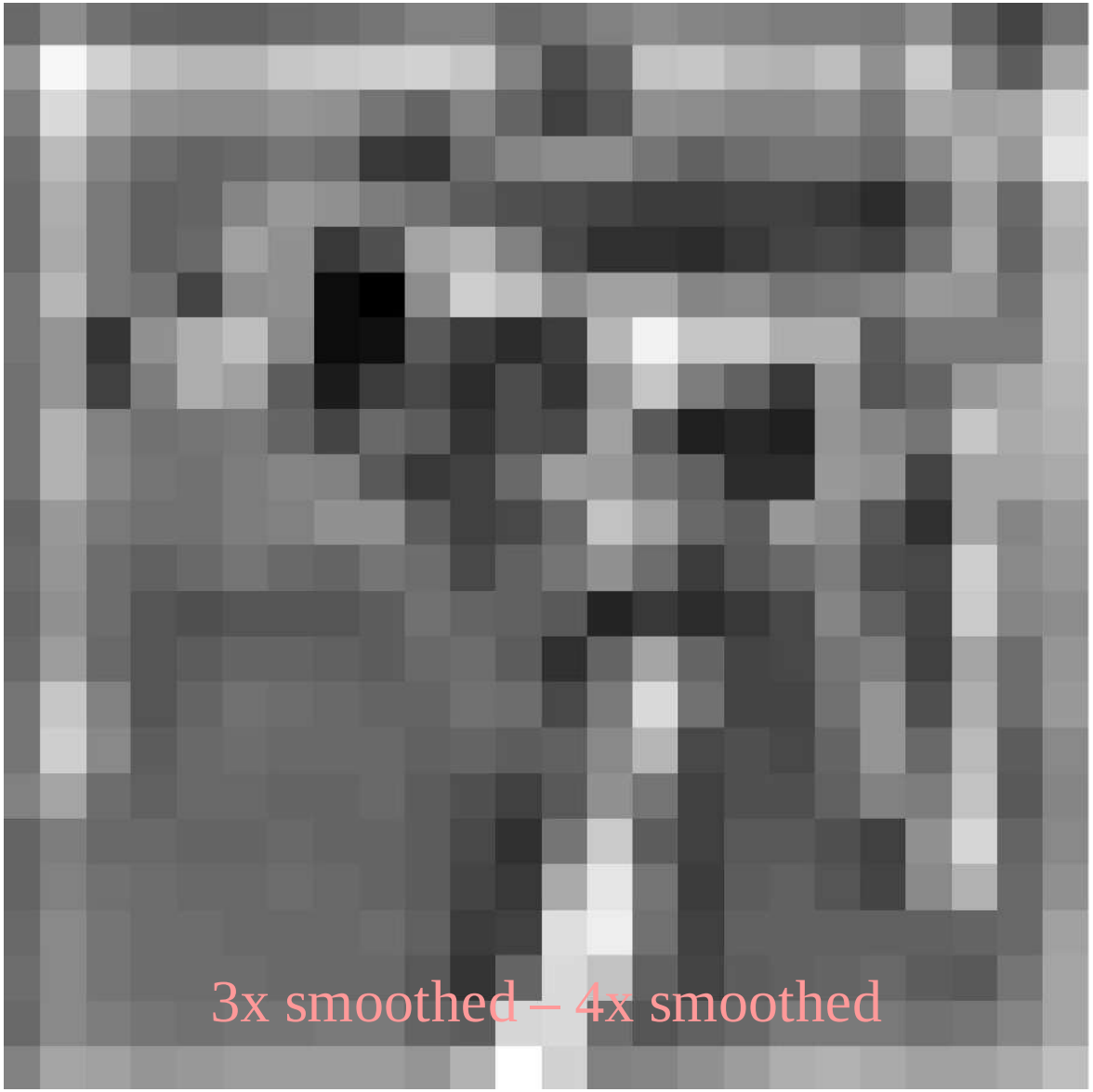




Difference of Gaussian

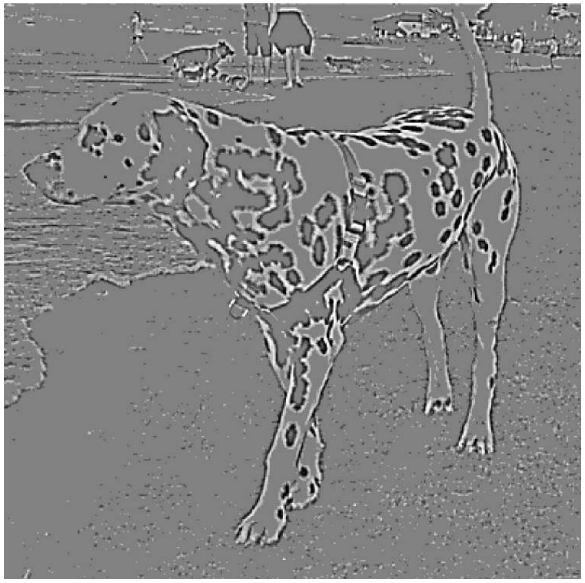





3x smoothed – 4x smoothed

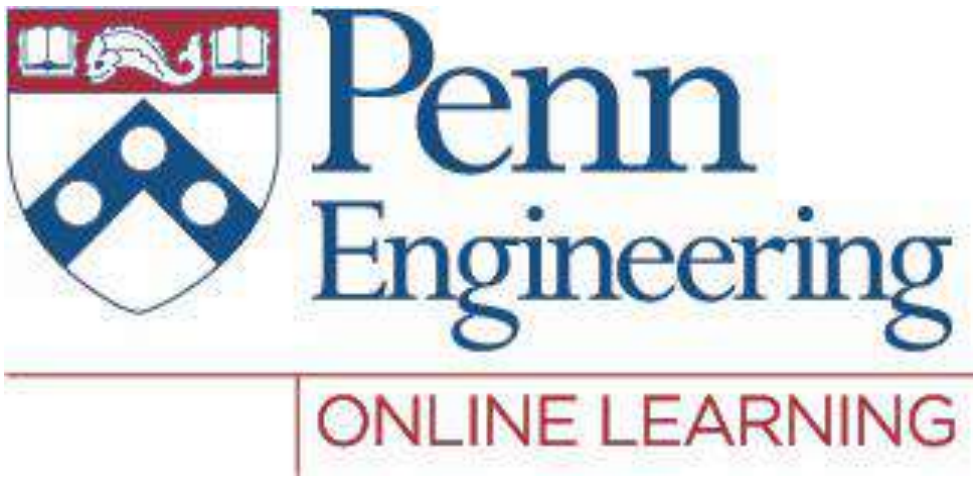
+

-



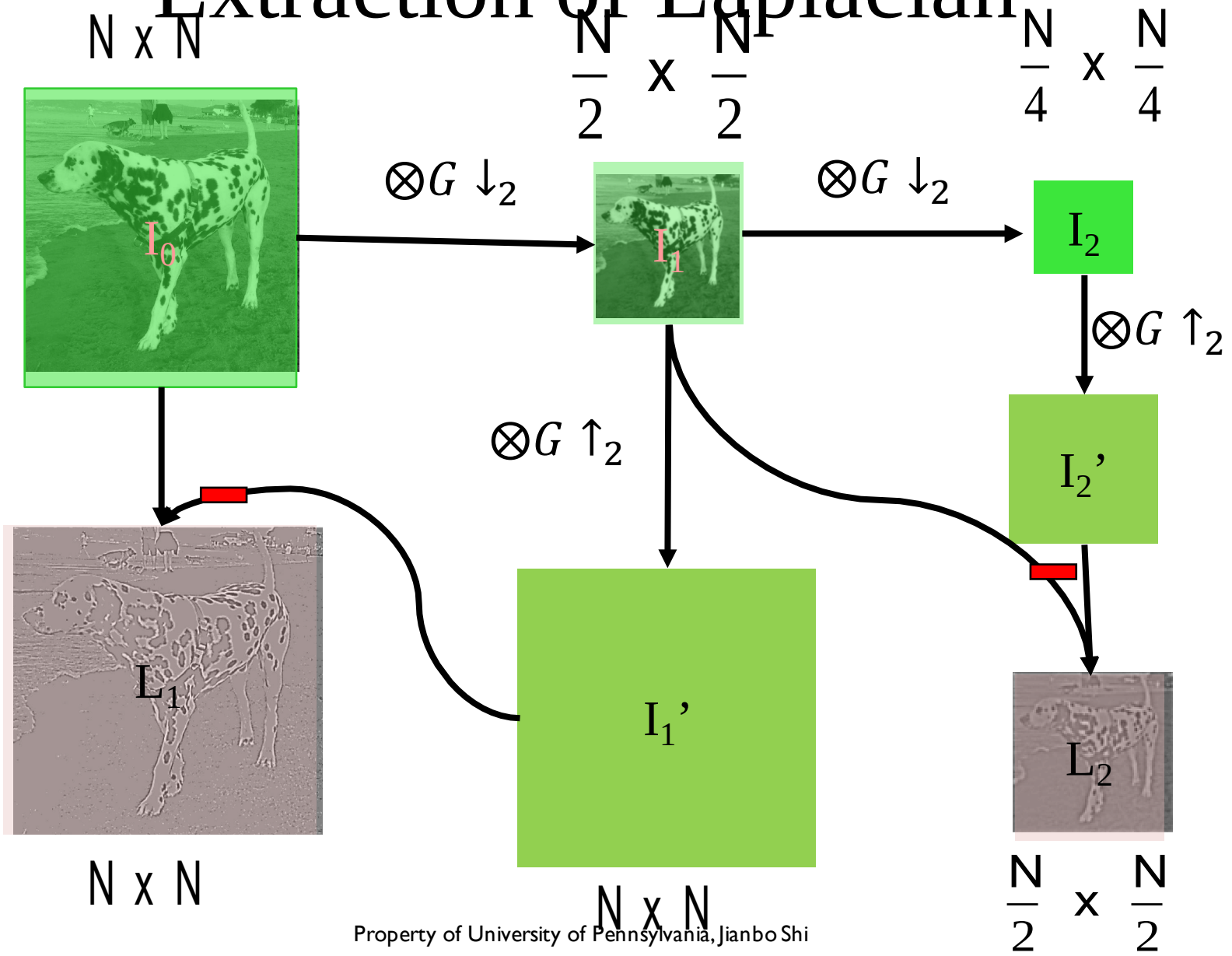
Laplacian Image

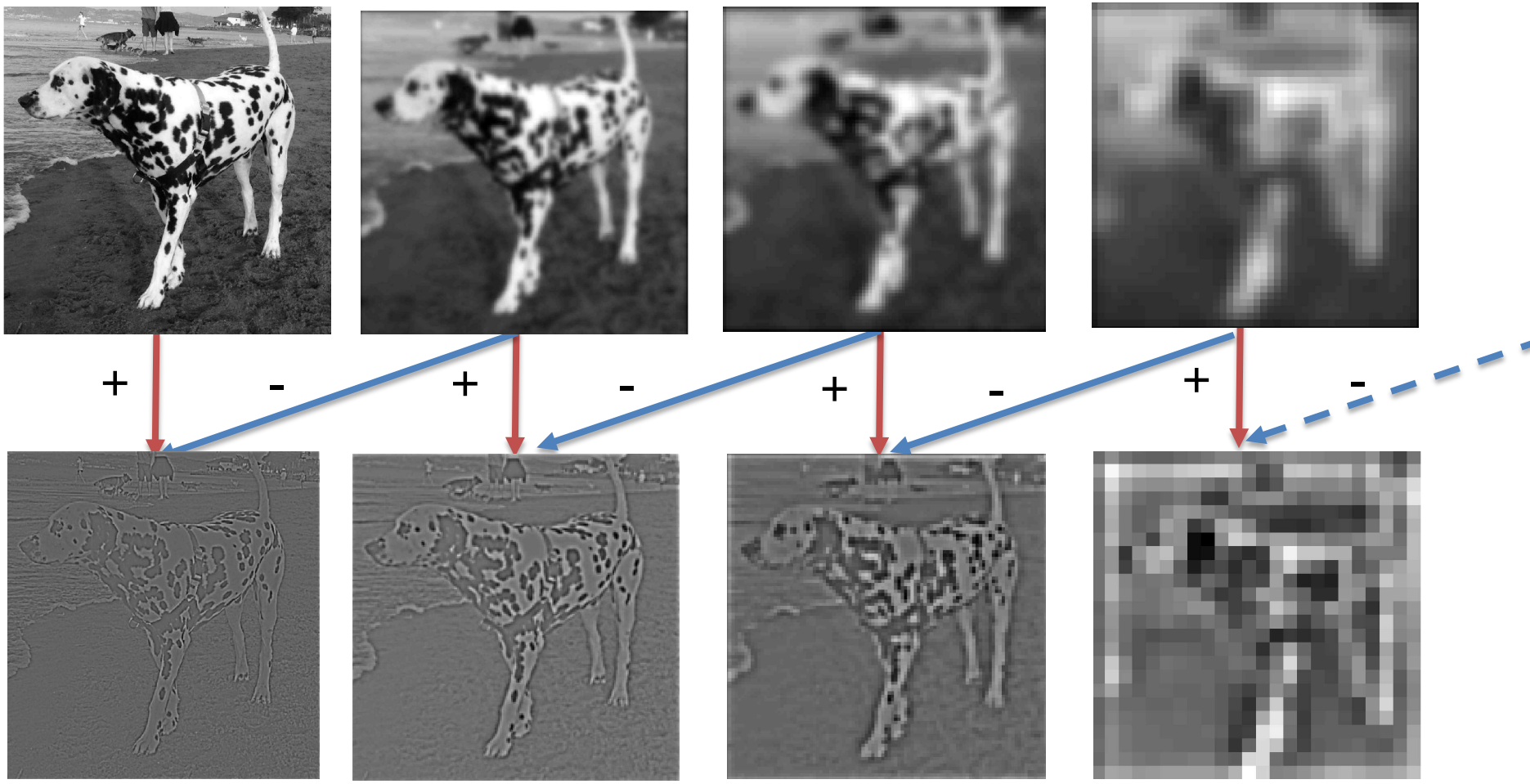
$$L_l = g_l - \text{EXPAND}[g_{l+1}]$$



Video 3.9
Jianbo Shi

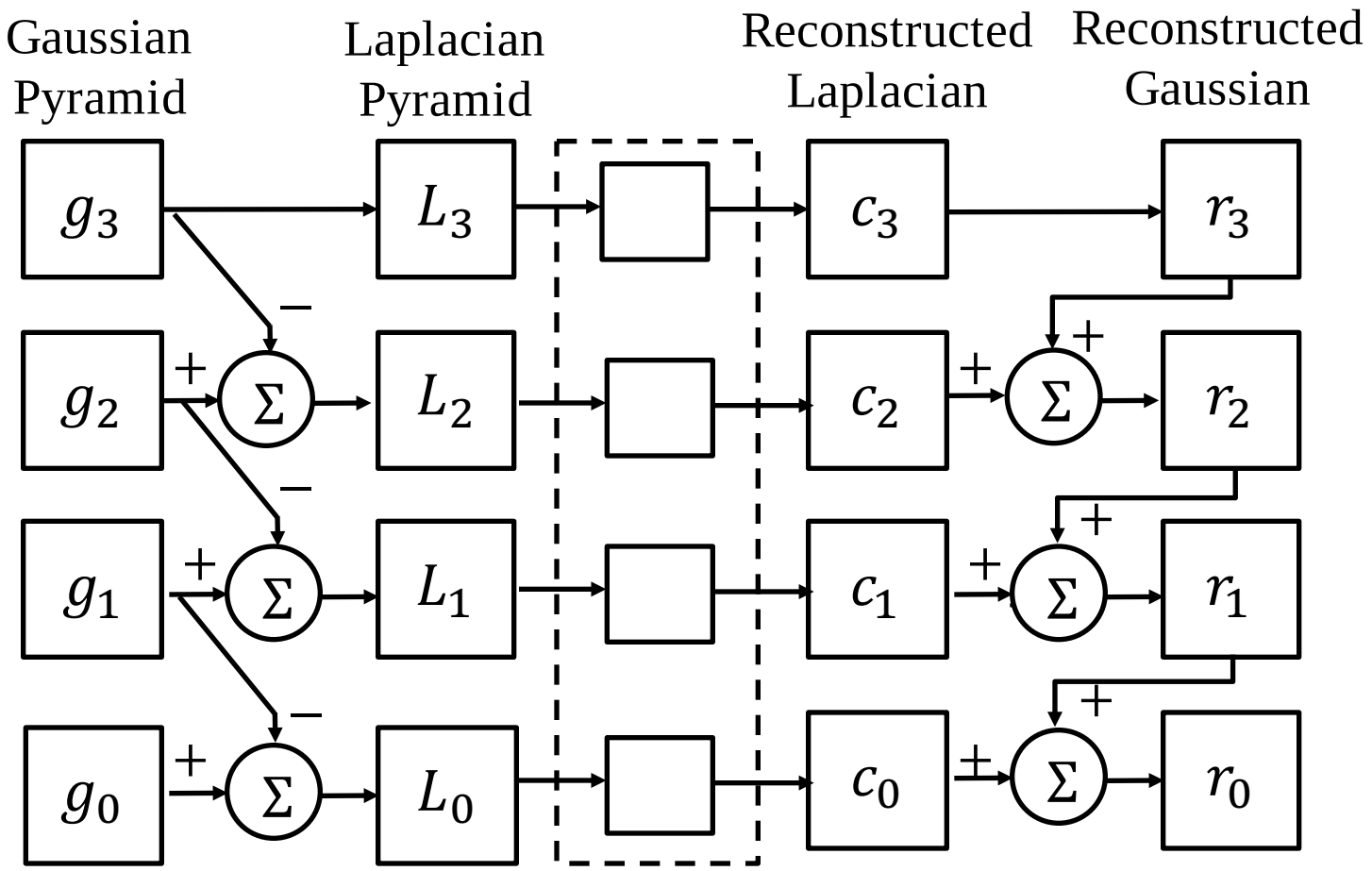
Extraction of Laplacian





Save only the last picture in Gaussian pyramid, and the entire Laplacian

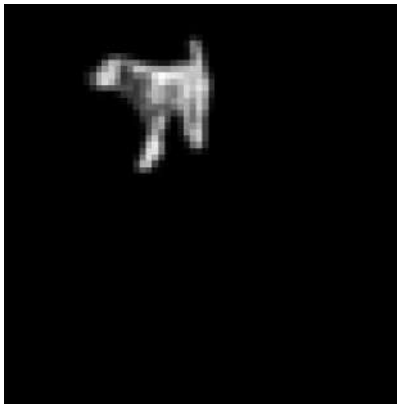
Laplacian pyramid has narrow band of frequency=> compressed



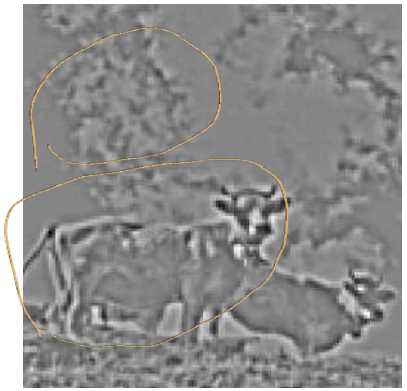
$$g_N = L_N \qquad g_l = L_l + \mathrm{EXPAND}[g_{l+1}]$$

Pyramid Blending

laplacian level 4



laplacian level 2

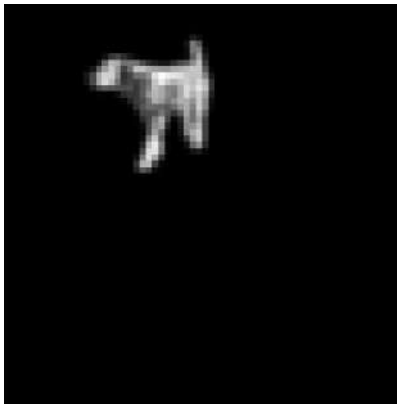


laplacian level 0

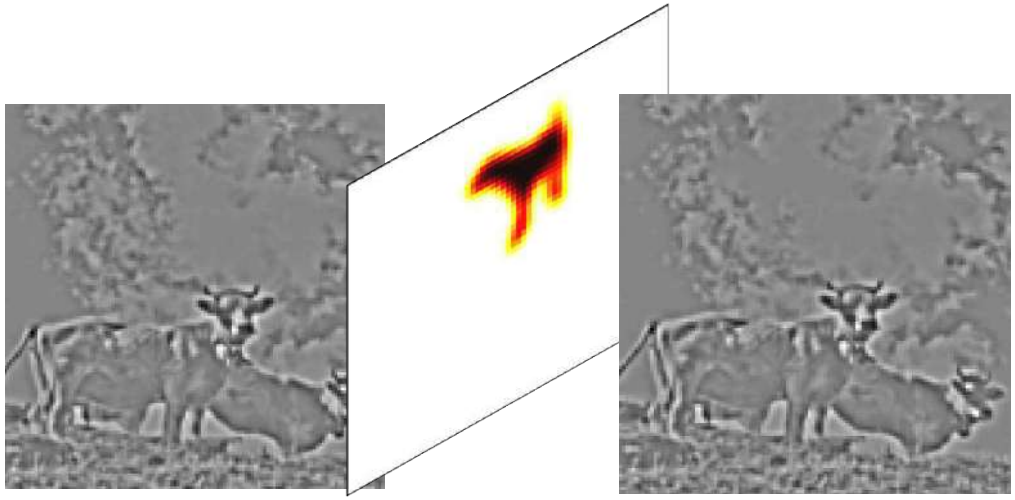
top pyramid

bottom pyramid

laplacian level 4



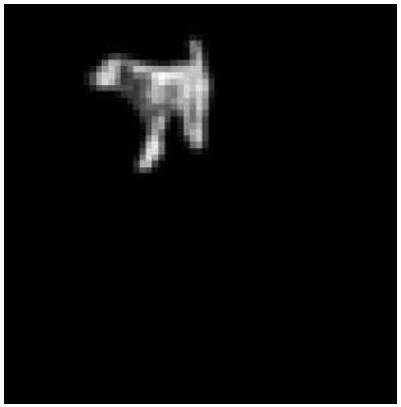
laplacian level 2



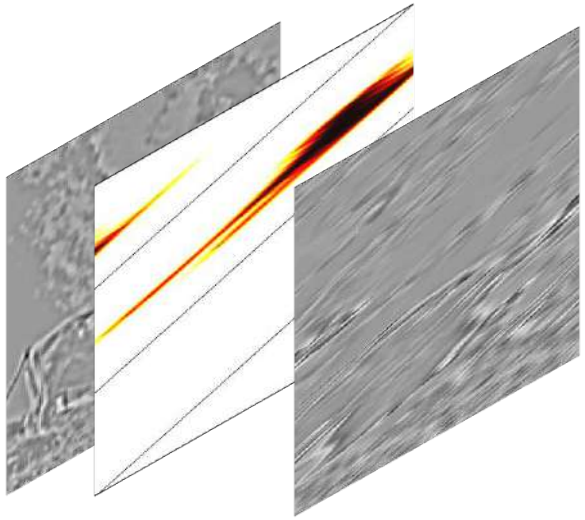
laplacian level 0

top pyramid bottom pyramid

laplacian level 4



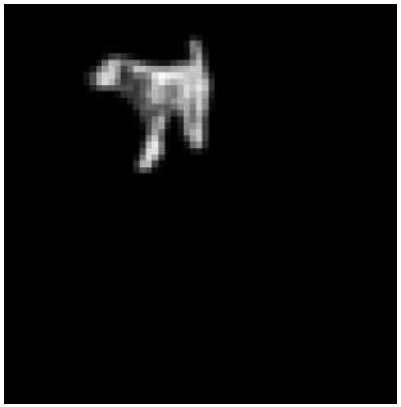
laplacian level 2



laplacian level 0

top pyramid bottom pyramid

laplacian level 4

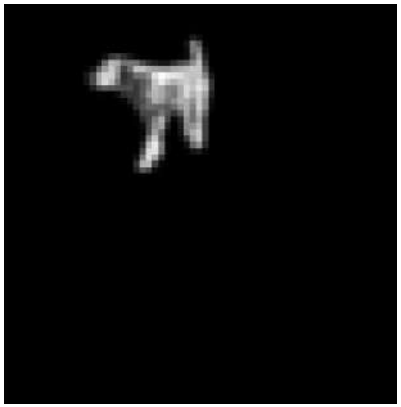


laplacian level 2

laplacian level 0

top pyramid bottom pyramid blended pyramid

laplacian level 4



laplacian level 2

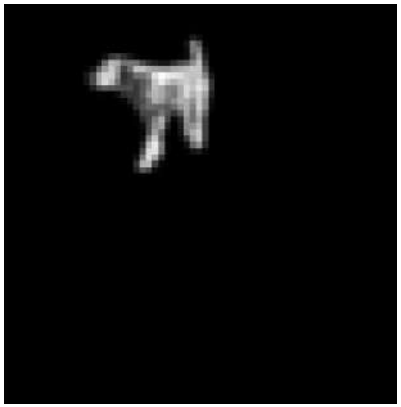
laplacian level 0

top pyramid

bottom pyramid

blended pyramid

laplacian level 4



laplacian level 2

laplacian level 0

top pyramid

bottom pyramid

blended pyramid

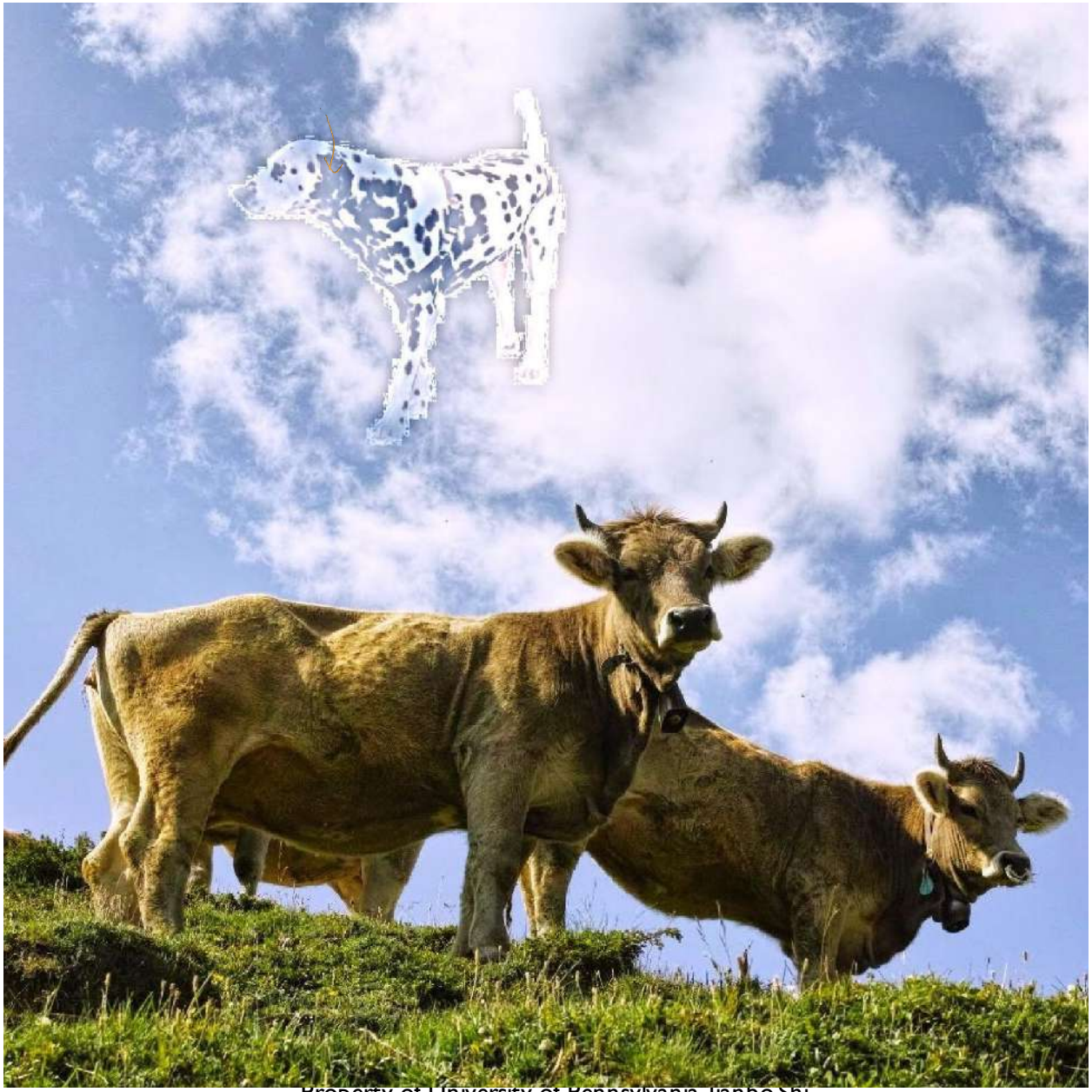
Laplacian Pyramid: Blending

General Approach:

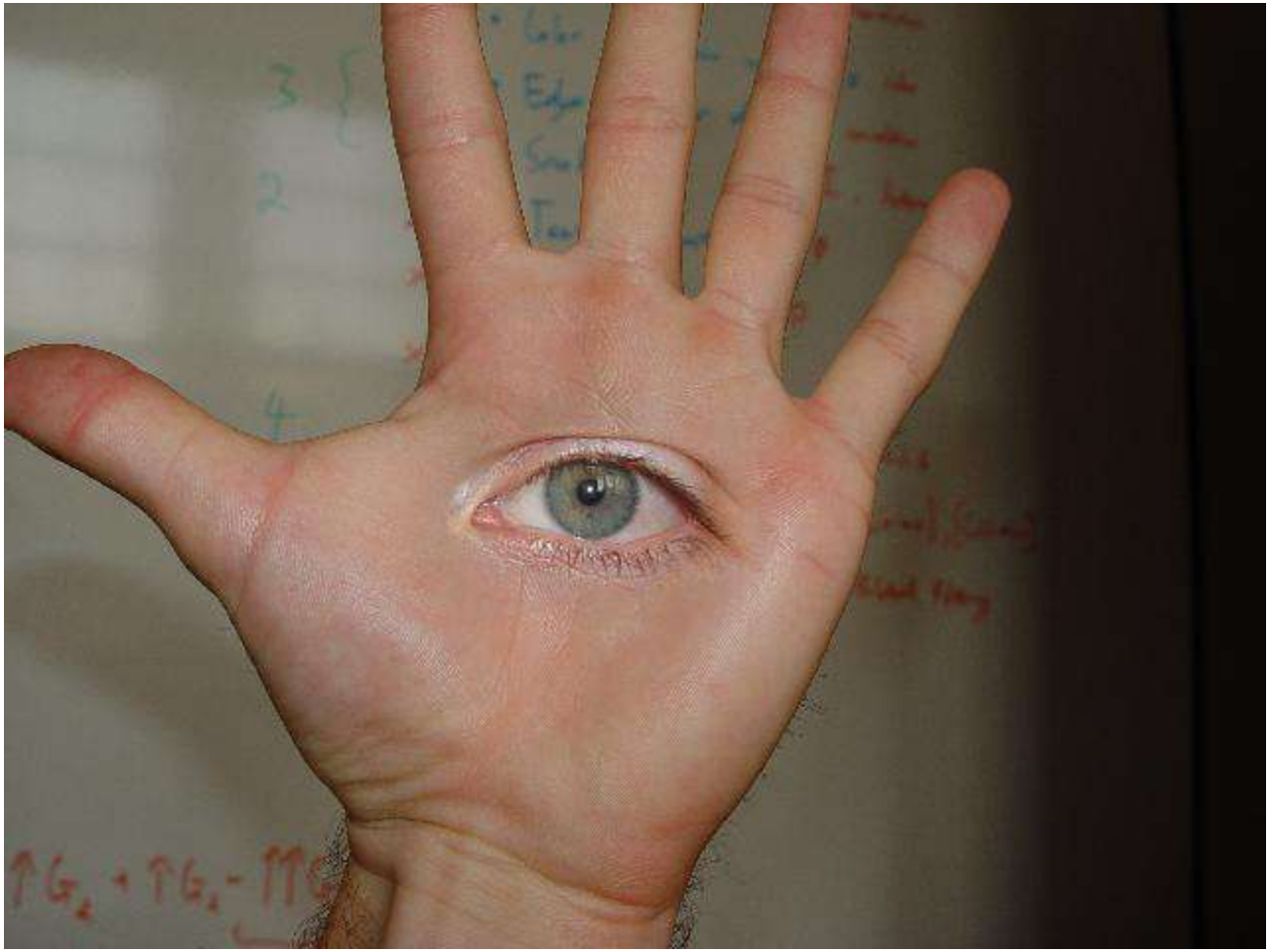
1. Build Laplacian pyramids LA and LB from images A and B
2. Build a Gaussian pyramid $MASK$ from selected region R
3. Form a combined pyramid LS from LA and LB using nodes of GR as weights:

   $$LS(i,j) = MASK(i,j) * LA(i,j) + (1 - MASK(i,j)) * LB(i,j)$$

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



Horror Photo



© prof. dmartin