Lecture 6 Edge Detection

COMP3204 & COMP6223 Computer Vision

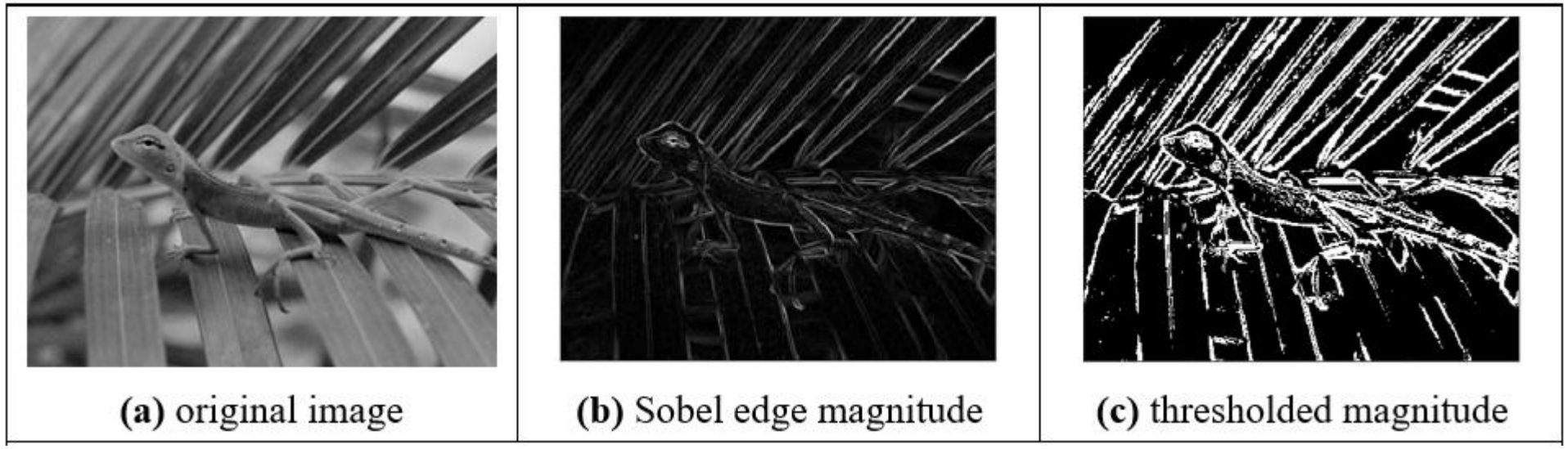
What are edges and how do we find them?

Book pp 140 - 153

Department of Electronics and Computer Science

UNIVERSITY OF Southampton
School of Electronics and Computer Science

Edge detection

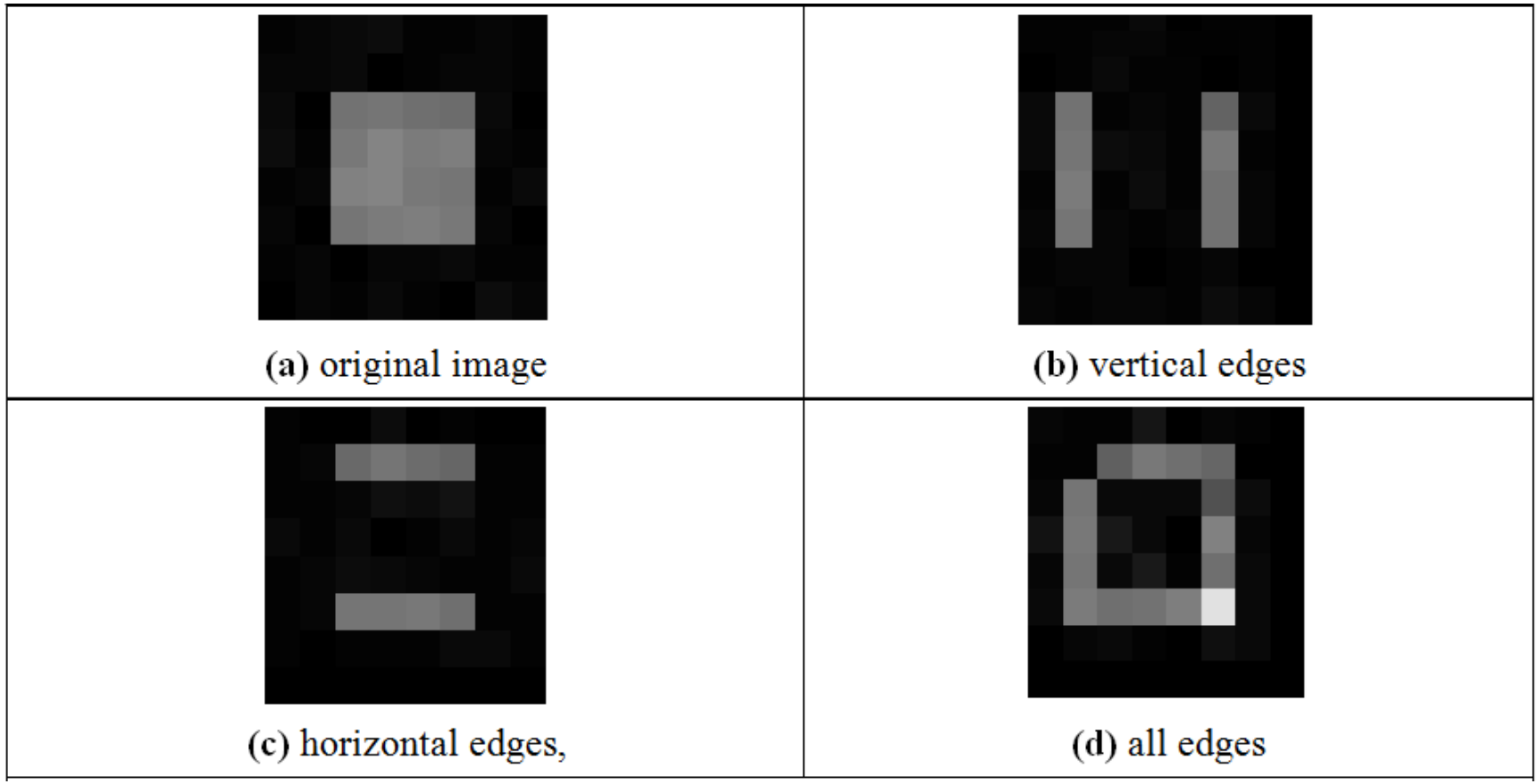


(a) original image | **(b)** Sobel edge magnitude | **(c)** thresholded magnitude

U2’s Edge can’t detect edges

http://metro.co.uk/2015/05/15/the-edge-falls-off-the-edge-of-the-stage-in-spectacular-style-during-u2s-world-tour-5199503/

First order edge detection



(a) original image

(b) vertical edges

(c) horizontal edges,

(d) all edges

First order edge detection

- vertical edges, **Ex** $\mathbf{Ex}_{x,y} = \left|\mathbf{P}_{x,y} - \mathbf{P}_{x+1,y}\right|$
- horizontal edges, **Ey** $\mathbf{Ey}_{x,y} = \left|\mathbf{P}_{x,y} - \mathbf{P}_{x,y+1}\right|$
- vertical and horizontal edges $\mathbf{E}_{x,y} = \left|2\times\mathbf{P}_{x,y} - \mathbf{P}_{x+1,y} - \mathbf{P}_{x,y+1}\right|$

First order edge detection

Template

2	-1
-1	0

Code

```
function edge = basic_difference(image)
for x = 2:cols-1 %address all columns except border
  for y = 2:rows-1 %address all rows except border
    edge(y,x)=abs(2*image(y,x)-image(y-1,x)-image(y,x-1));
  end
end
```

Edge detection maths

Taylor expansion for $f(x+\Delta x)$ $f(x+\Delta x) = f(x) + \Delta x \times f'(x) + \frac{\Delta x^2}{2!} \times f''(x) + O(\Delta x^3)$ **A**

By rearrangement $f'(x) = \frac{f(x+\Delta x) - f(x)}{\Delta x} - O(\Delta x)$

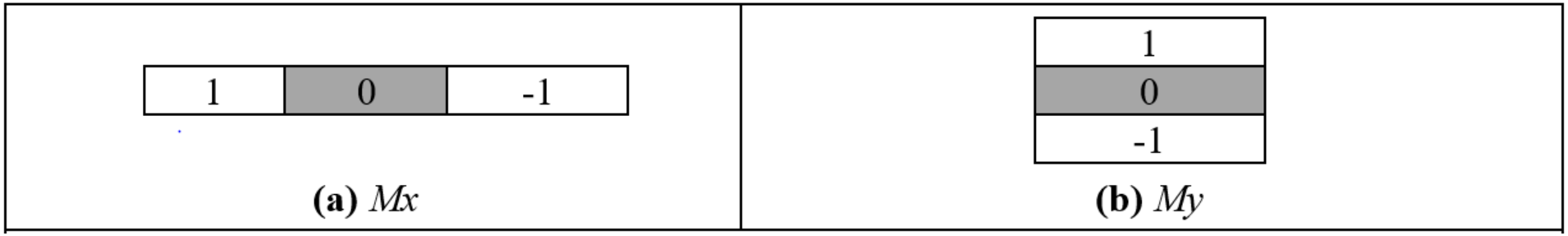
This is equivalent to $\mathbf{Exx}_{x,y} = \left|\mathbf{P}_{x+1,y} - \mathbf{P}_{x-1,y}\right|$

Expand $f(x-\Delta x)$ $f(x-\Delta x) = f(x) - \Delta x \times f'(x) + \frac{\Delta x^2}{2!} \times f''(x) - O(\Delta x^3)$ **B**

A – B $f'(x) = \frac{f(x+\Delta x) - f(x-\Delta x)}{2\Delta x} - O(\Delta x^2)$

If $\Delta x < 1$, this error is clearly smaller

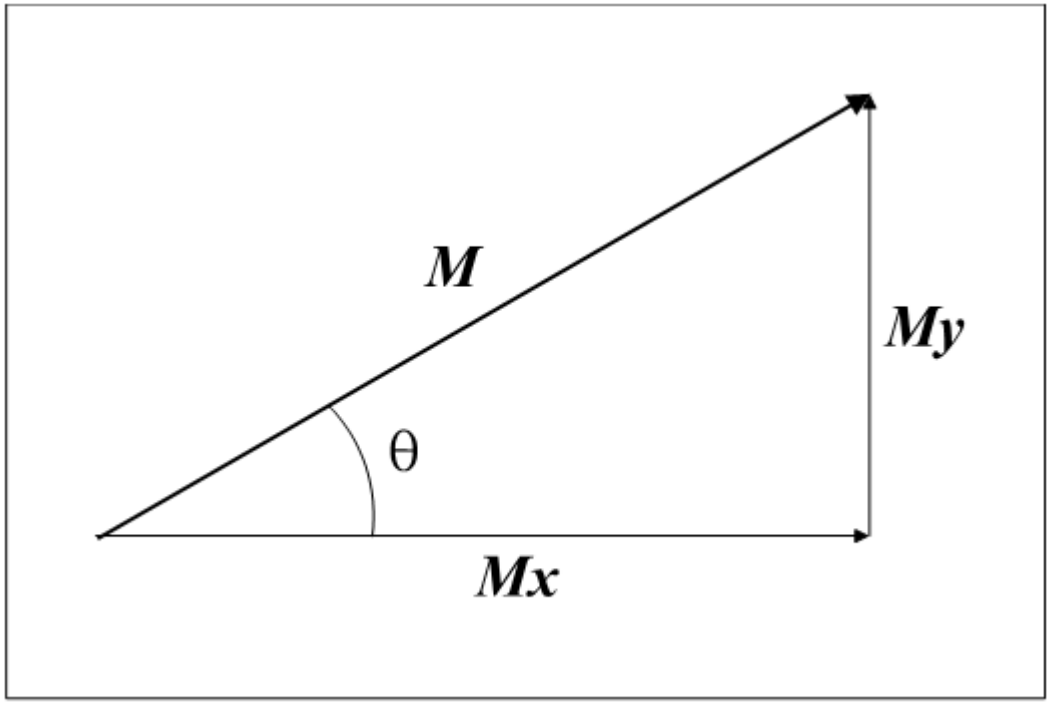
Templates for improved first order difference



Edge Detection in Vector Format

M = magnitude = $\sqrt{M_x^2+M_y^2}$

θ = direction = $tan^{-1}\left(\frac{M_y}{M_x}\right)$



Templates for Prewitt operator

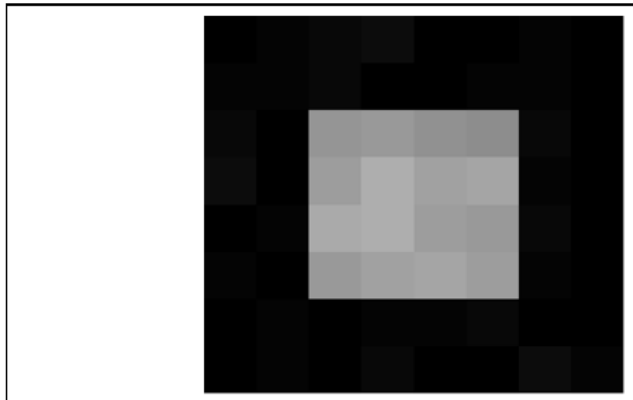
1	0	-1
1	0	-1
1	0	-1

(a) *Mx*

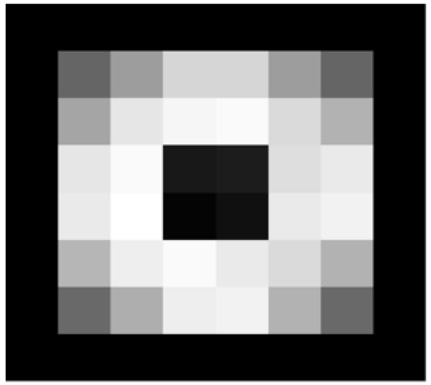
1	1	1
0	0	0
-1	-1	-1

(b) *My*

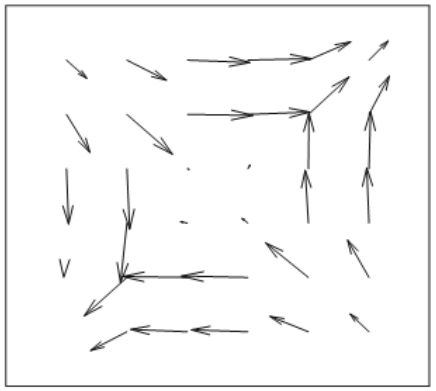
Applying the Prewitt Operator



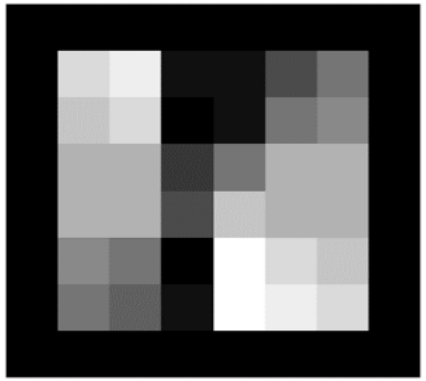
(a) original image



(b) edge magnitude



(c) vector format



(d) edge direction

Templates for Sobel operator

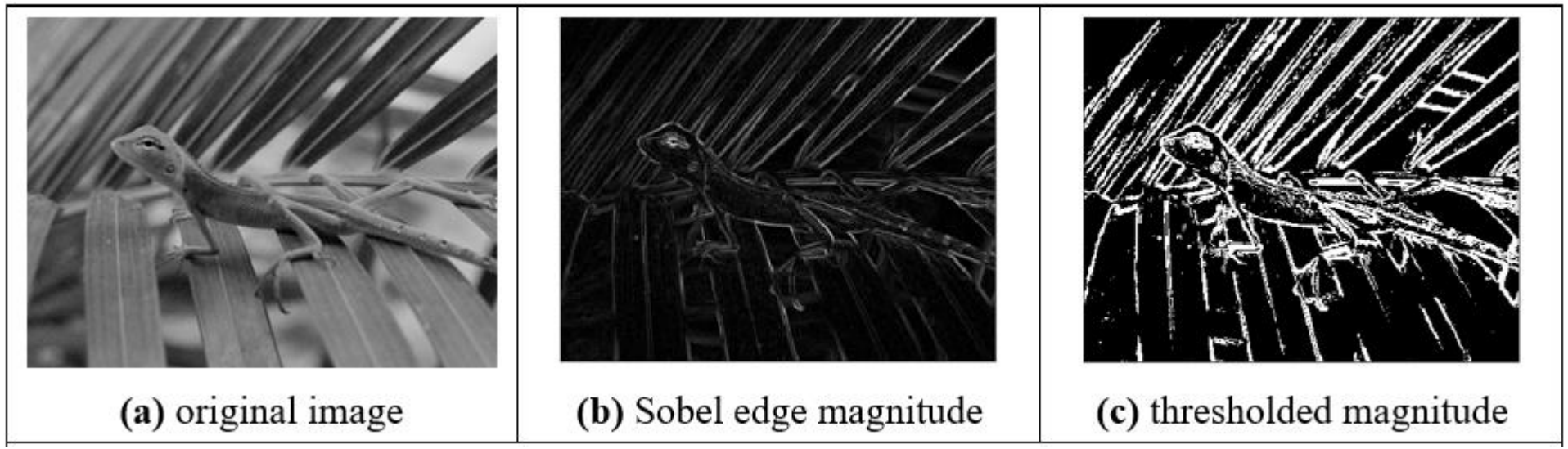
1	0	-1
2	0	-2
1	0	-1

(a) *Mx*

1	2	1
0	0	0
-1	-2	-1

(b) *My*

Applying Sobel operator



(a) original image

(b) Sobel edge magnitude

(c) thresholded magnitude

Generalising Sobel

- Averaging

Window size									
2				1		1			
3			1		2		1		
4		1		3		3		1	
5	1		4		6		4		1

- Differencing

Window size									
2				1		-1			
3			1		0		-1		
4		1		1		-1		-1	
5	1		2		0		-2		-1

Generalised Sobel

Generated by: `averaging*(differencing)`^{T}

```
>> s=Sobel_templates(5)
s(:,:,1) =
     1     2     0    -2    -1
     4     8     0    -8    -4
     6    12     0   -12    -6
     4     8     0    -8    -4
     1     2     0    -2    -1
```

COURSEWORK!!!!