

CS325 Artificial Intelligence

Ch. 24, Computer Vision I – Object Recognition

Cengiz Günay, Emory Univ.



Spring 2013

Computer Vision

- Done with games, except homework :)

Computer Vision

- Done with games, except homework :)



- **Vision** is one of our main perceptions
- **Computer vision** is what robots use to understand their surrounding



Computer Vision

- Done with games, except homework :)



- **Vision** is one of our main perceptions
- **Computer vision** is what robots use to understand their surrounding

3 lectures:

- 1 Object recognition (today)
- 2 3D reconstruction
- 3 Motion analysis



Exit survey: Advanced Planning

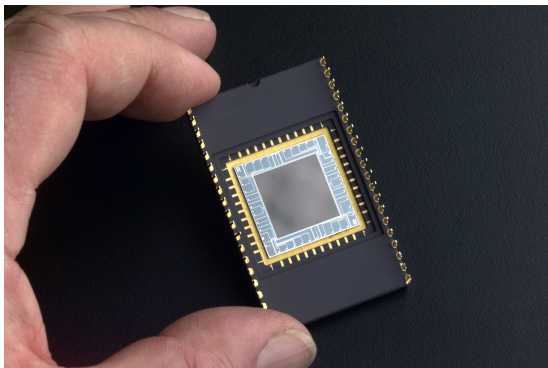
- Why isn't classical planning schema adequate for resource planning?
- What is the advantage gained in abstract plans by having *surely-reachable* versus *potentially-reachable* states?

Entry survey: Computer Vision I – Image Processing (0.25 points)

- List three specific tasks where computer vision would be desirable.
- What do you think are the major hurdles in computer vision?

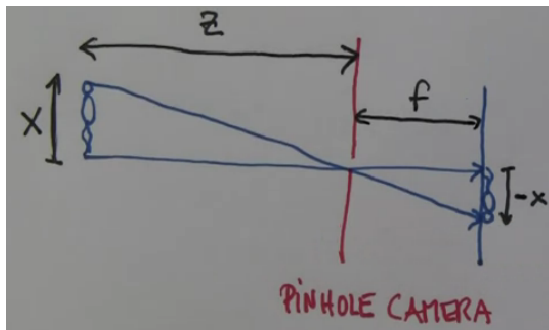
How Machines See: Cameras

A charge-coupled device (CCD) photo sensor array:



Focal Optics for Determining Distance and Size

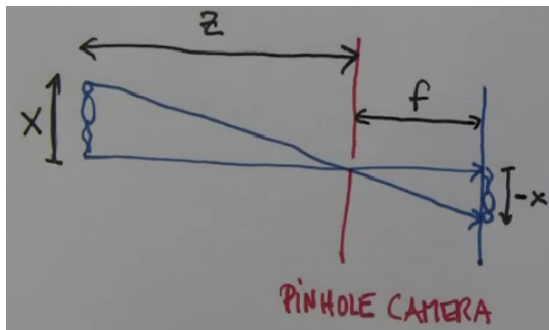
See the videos, I'll summarize:



$$\frac{X}{Z} = \frac{x}{f}$$

Focal Optics for Determining Distance and Size

See the videos, I'll summarize:

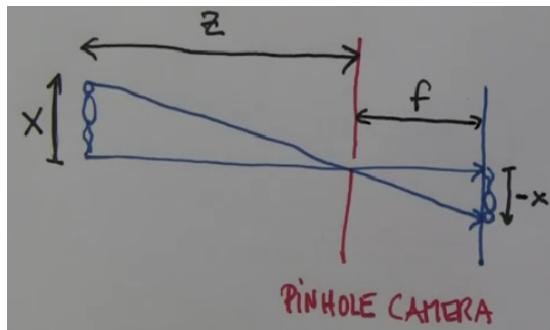


$$\frac{X}{Z} = \frac{x}{f}$$

What can we figure out from this?

Focal Optics for Determining Distance and Size

See the videos, I'll summarize:



$$\frac{X}{Z} = \frac{x}{f}$$

What can we figure out from this?

- Object's distance (Z) & height (X) based on projection height (x) and focal distance (f)

We All See a Perspective Projection

Vanishing points from parallel lines:



We All See a Perspective Projection

Vanishing points from parallel lines:



We All See a Perspective Projection

Vanishing points from parallel lines:



We All See a Perspective Projection

Vanishing points from parallel lines:



- Giant panda, or just close?

Object Recognition: How Hard Can It Be?



Object Recognition: How Hard Can It Be?



Problems?

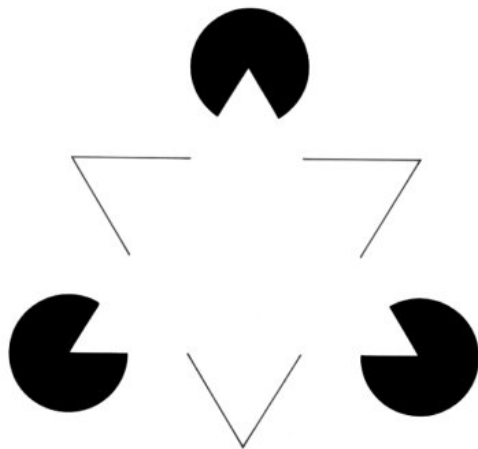
Object Recognition: How Hard Can It Be?



Problems?

- Rotation, scale, illumination, occlusion, viewpoint, deformation

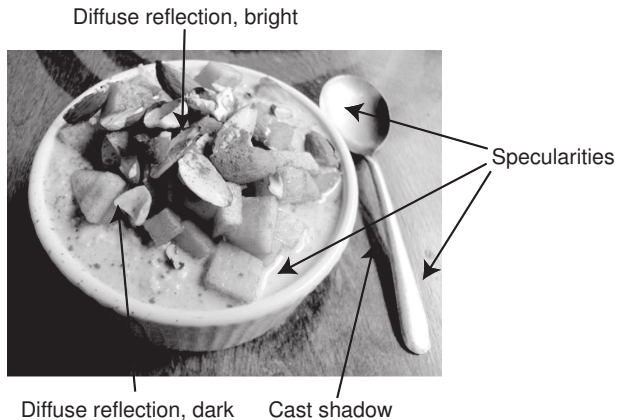
Not Hard for Us



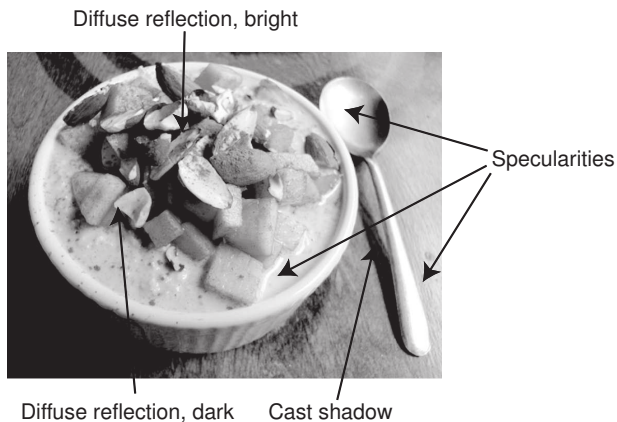
Not Hard for Us



Not Hard for Us

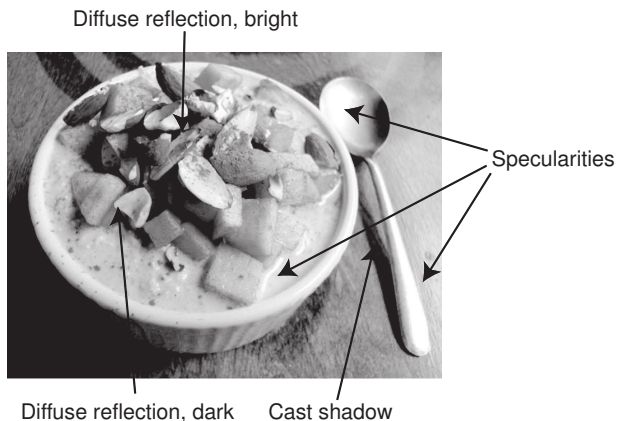


Not Hard for Us



How does our brain do it?

Not Hard for Us



How does our brain do it? Will have examples later.

Invariance is Crucial for Computer Vision

Must recognize objects **invariant** of their:

- Rotation, scale, illumination, occlusion, viewpoint, deformation

Invariance is Crucial for Computer Vision

Must recognize objects **invariant** of their:

- Rotation, scale, illumination, occlusion, viewpoint, deformation

Let's start by simplifying:

- 1 Greyscale (monochrome) images
- 2 Pixels can have values: 0...255

Even Terminator Has Monochrome Vision



ANALYSIS: SERIES
1000 TERMINATOR
PROTOTYPE

DEFENSE MODE LEVEL 69825

TARGET ACQUIRED

PRIMARY MISSION: ENSURE THE SURVIVAL OF JOHN CONNOR

DAMAGE TAKEN: 46%
RUNNING ON 53% ENERGY
21 BULLETS LEFT IN CLIP

POSSIBILITY OF T-1000 TERMINATION: 52%

VISUAL: TERMINATOR MODEL 1000

CAUTION: T-1000 CAPABLE OF KNIVES AND STABBING WEAPONS EQUIPED WITH HANDGUN

VULNERABLE TO MOLTEN STEEL AND LIQUID NITROGEN

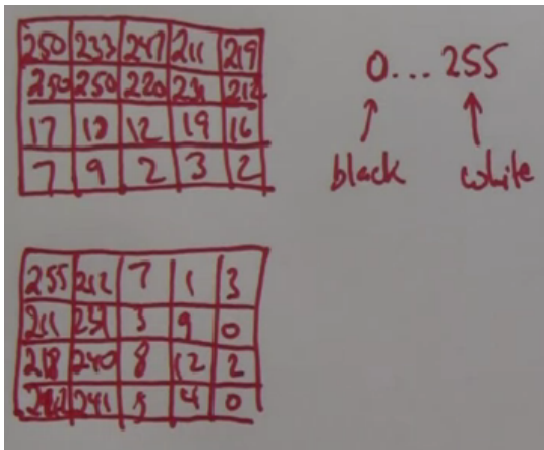
01. T-1000 STATUS
02. RATIO OF ATTACK
03. 83%
04. A1 = 84889
05. A2 = 00933
06. A3 = F8367
07. A4 = G0894
08. DISTANCE AFT
09. A5 = H0837
10. A6 = J0948
11. A7 = K6364
12. A8 = L3748
13. A9 = Z3864

14. 9846592834
15. 2094875204
16. 8764523456
17. VISUAL IN SITE
18. 873538724
19. 2345987087
20. 5883745809
21. 0987435234
22. 0876863456
23. 87560983475689347
24. 87076345665421234
25. 74657483230856723
26. 74652983745692387
27. 75747503784785747
28. 48584398344857984
29. 45786384760847508
30. 57830984763098959
31. 47560238745089763
32. 47577458868308568
33. 466375947568285975
34. 74295884558943534
35. 75345082347523049

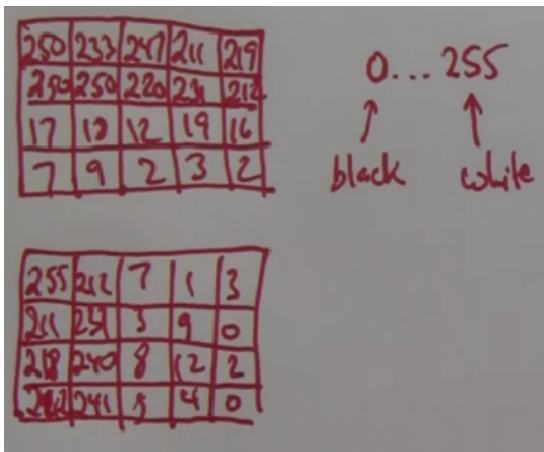
Extracting Features: Edge Detection



Extracting Features: Edge Detection

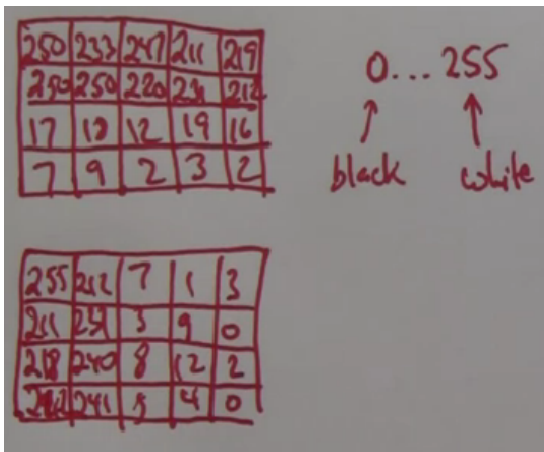


Extracting Features: Edge Detection



How to detect the vertical edge?

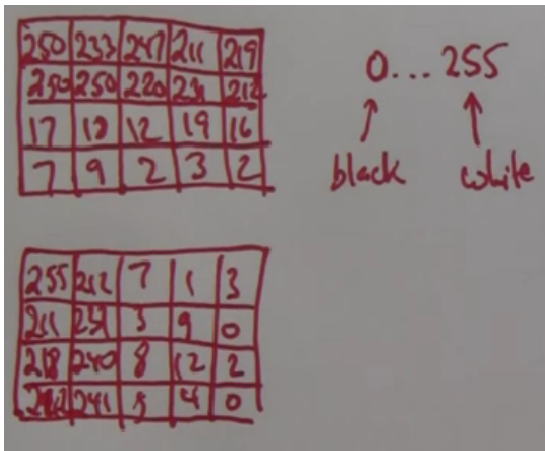
Extracting Features: Edge Detection



How to detect the vertical edge?

- 1 Spatial derivative?

Extracting Features: Edge Detection



How to detect the vertical edge?

1 Spatial derivative?

2 Filter with mask:

+1	-1
----	----

Extracting Features: Edge Detection

0... 255
↑ ↑
black white

250	233	247	211	219
250	250	220	231	212
17	12	12	19	16
7	9	2	3	2

255 211 ? 1 3
211 251 5 9 0
218 240 8 12 2
212 241 5 4 0

+1 -1 mask

43	205	6	-2
-26	234	-6	9
-22	232	-4	8
0	235	1	4

How to detect the vertical edge?

1 Spatial derivative?

2 Filter with mask:

+1	-1
----	----

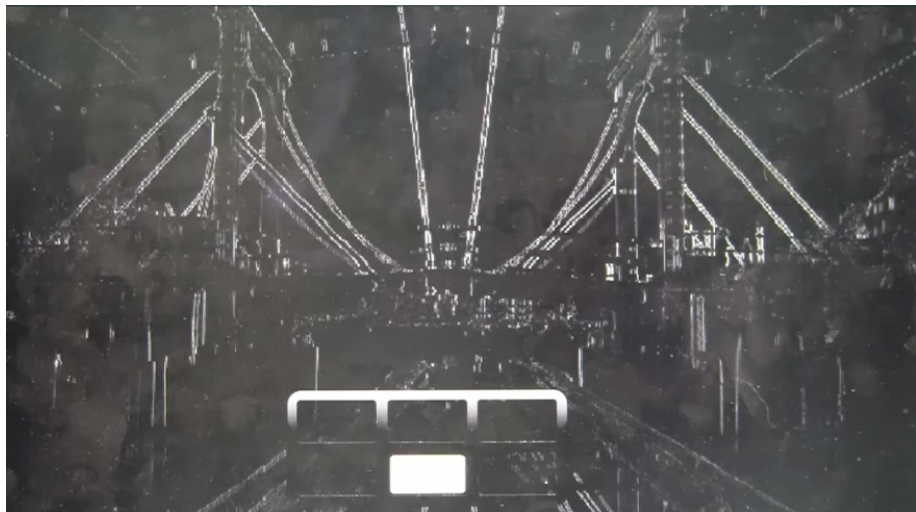
Extracting Features: Edge Detection



How to detect the vertical edge?

1 Spatial derivative?

Extracting Features: Edge Detection

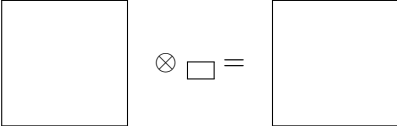


How to detect the vertical edge?

1 Spatial derivative?

Edge Detection: Linear Filter


What we did is called **convolution**:

$$I \otimes g = I'$$


The diagram shows the convolution operation. On the left, a large square represents the input image I . In the middle, a small square represents the kernel g . On the right, a large square represents the output image I' . The operation is denoted by \otimes and an equals sign.

Edge Detection: Linear Filter

What we did is called **convolution**:

$$I \otimes g = I'$$



The diagram shows the convolution operation. At the top, the equation $I \otimes g = I'$ is written. Below it, a large square on the left represents the input image I . To its right is a small square representing the kernel g . An equals sign follows, and then another large square on the right represents the output image I' . The small square g is positioned such that its top-left corner is aligned with the top-left corner of the large square I .

For each pixel, we multiply by **mask** and sum:

$$I'(x, y) = \sum_{u, v} I(x - u, y - v) g(u, v)$$

Edge Detection: Linear Filter

What we did is called **convolution**:

$$I \otimes g = I'$$


The diagram shows the convolution operation. At the top, the equation $I \otimes g = I'$ is written. Below it, a large square represents the input image I . To its right, a small square represents the kernel g . An equals sign follows, and then another large square represents the output image I' . The convolution symbol \otimes is placed between I and g , and between g and I' .


For each pixel, we multiply by **mask** and sum:

$$I'(x, y) = \sum_{u, v} I(x - u, y - v) g(u, v)$$

Does that equation look familiar?

Edge Detection: Linear Filter

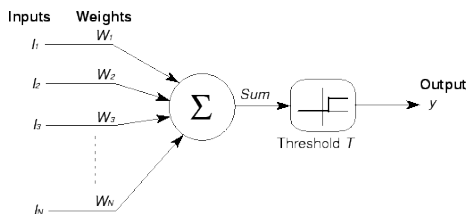
What we did is called **convolution**:

$$I \otimes g = I'$$


For each pixel, we multiply by **mask** and sum:


$$I'(x, y) = \sum_{u, v} I(x - u, y - v) g(u, v)$$

Does that equation look familiar?
Perceptron?



Edge Detection: Linear Filter

What we did is called **convolution**:

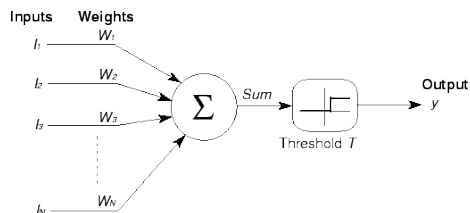
$$I \otimes g = I'$$


For each pixel, we multiply by **mask** and sum:

$$I'(x, y) = \sum_{u, v} I(x - u, y - v) g(u, v)$$


Does that equation look familiar?
Perceptron?

- What are the weights?



Edge Detection: Linear Filter

What we did is called **convolution**:

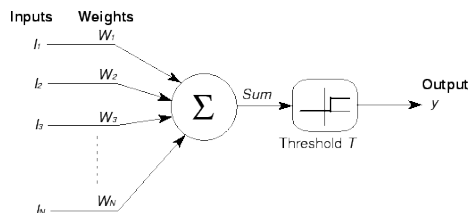
$$I \otimes g = I'$$


For each pixel, we multiply by **mask** and sum:

$$I'(x, y) = \sum_{u, v} I(x - u, y - v) g(u, v)$$


Does that equation look familiar?
Perceptron?

- What are the weights?
The mask, g .



Edge Detection: Linear Filter

What we did is called **convolution**:

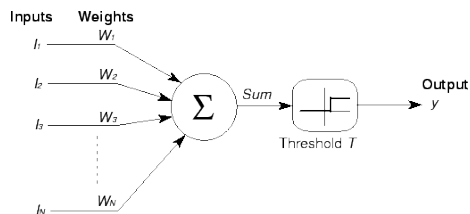
$$I \otimes g = I'$$


For each pixel, we multiply by **mask** and sum:

$$I'(x, y) = \sum_{u, v} I(x - u, y - v) g(u, v)$$


Does that equation look familiar?
Perceptron?

- What are the weights?
The mask, g .
- What's the advantage?



Edge Detection: Linear Filter

What we did is called **convolution**:

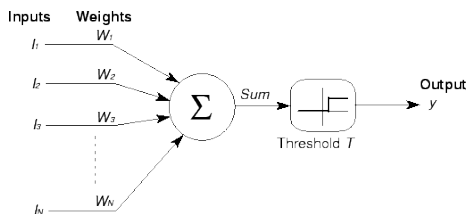
$$I \otimes g = I'$$


For each pixel, we multiply by **mask** and sum:

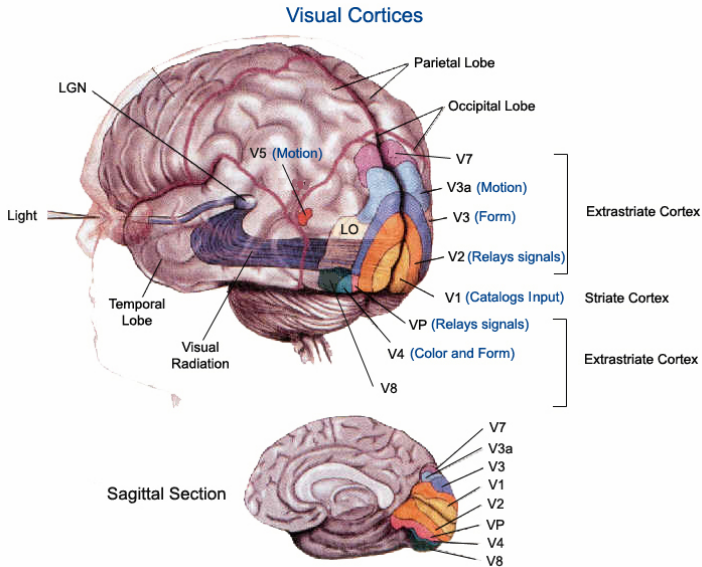
$$I'(x, y) = \sum_{u, v} I(x - u, y - v) g(u, v)$$

Does that equation look familiar?
Perceptron?

- What are the weights?
The mask, g .
- What's the advantage?
Works in parallel!



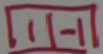
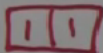
Neurons Can Do It Faster?



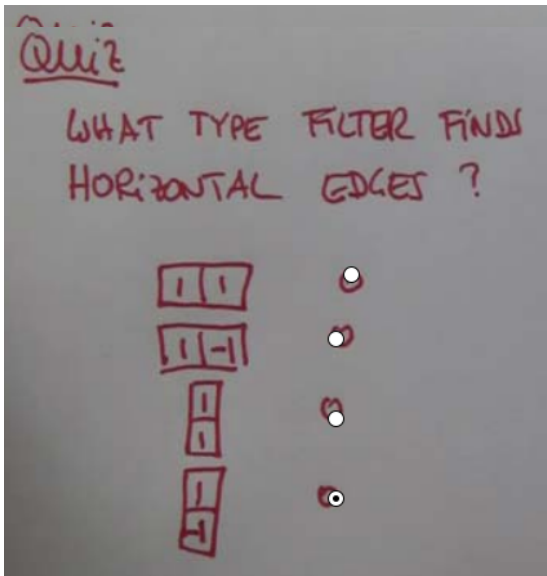
Detect Only Vertical Edges?

Quiz

WHAT TYPE FILTER FINDS
HORIZONTAL EDGES ?



Detect Only Vertical Edges?



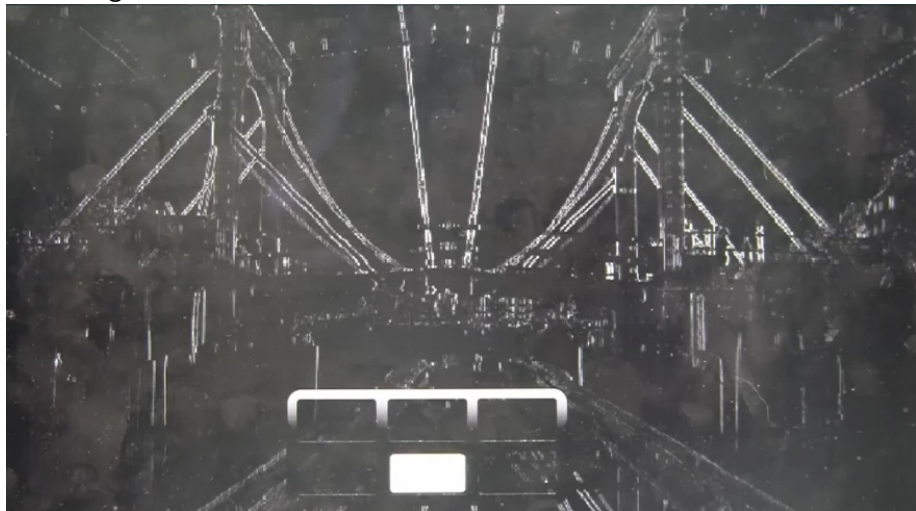
Horizontal and Vertical Gradients

Original:



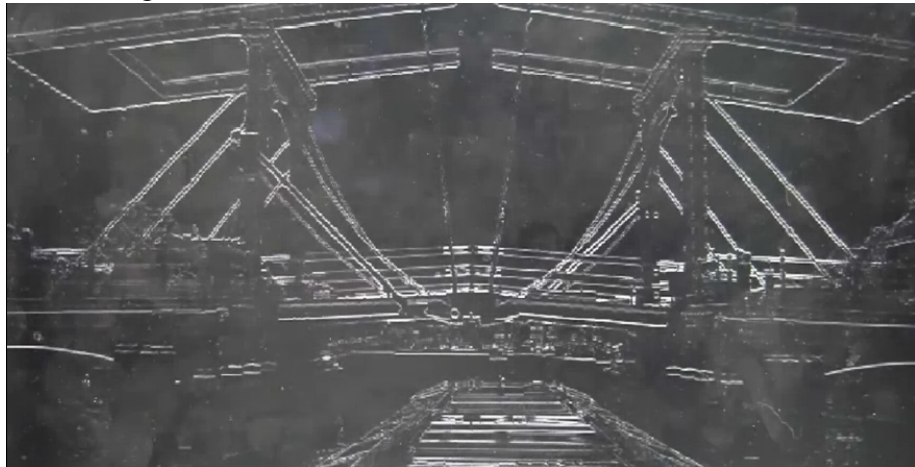
Horizontal and Vertical Gradients

Vertical gradient:



Horizontal and Vertical Gradients

Horizontal gradient:



Combining Gradients

Horizontal mask gives vertical gradient (I_x) and vice versa:

$$I_x = I \otimes \begin{bmatrix} -1 & +1 \end{bmatrix}$$

$$I_y = I \otimes \begin{bmatrix} -1 \\ +1 \end{bmatrix}$$

Combining Gradients

Horizontal mask gives vertical gradient (I_x) and vice versa:

$$I_x = I \otimes \begin{bmatrix} -1 & +1 \end{bmatrix}$$

$$I_y = I \otimes \begin{bmatrix} -1 \\ +1 \end{bmatrix}$$

How to combine them?

Combining Gradients

Horizontal mask gives vertical gradient (I_x) and vice versa:

$$I_x = I \otimes \begin{bmatrix} -1 & +1 \end{bmatrix}$$

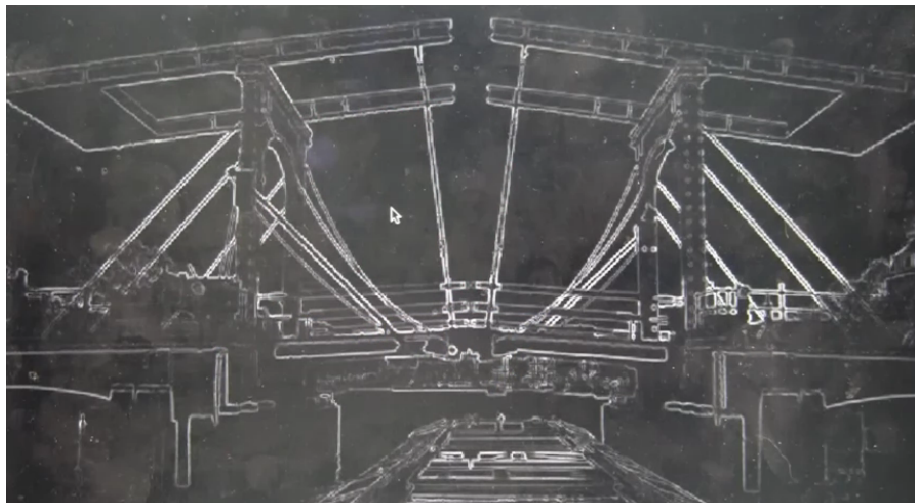
$$I_y = I \otimes \begin{bmatrix} -1 \\ +1 \end{bmatrix}$$

How to combine them?

$$E = \sqrt{I_x^2 + I_y^2}$$

Horizontal and Vertical Gradients

Combined gradients:



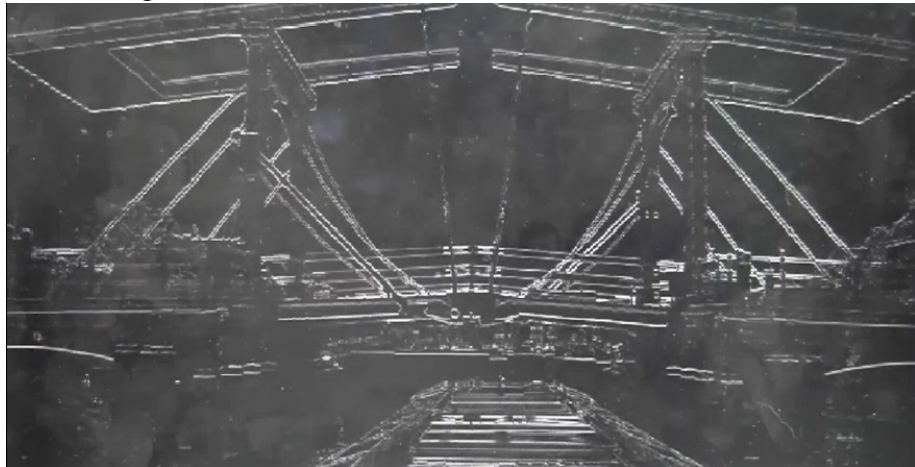
Horizontal and Vertical Gradients

Original:



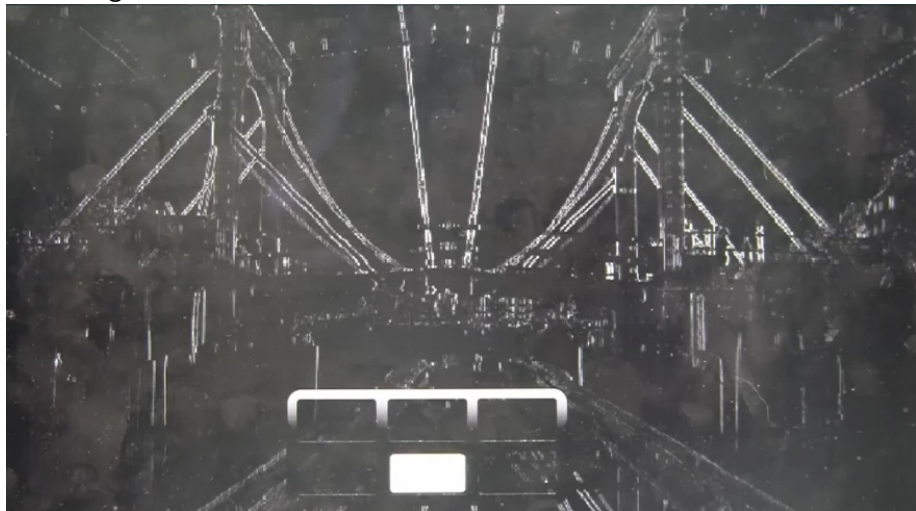
Horizontal and Vertical Gradients

Horizontal gradient:



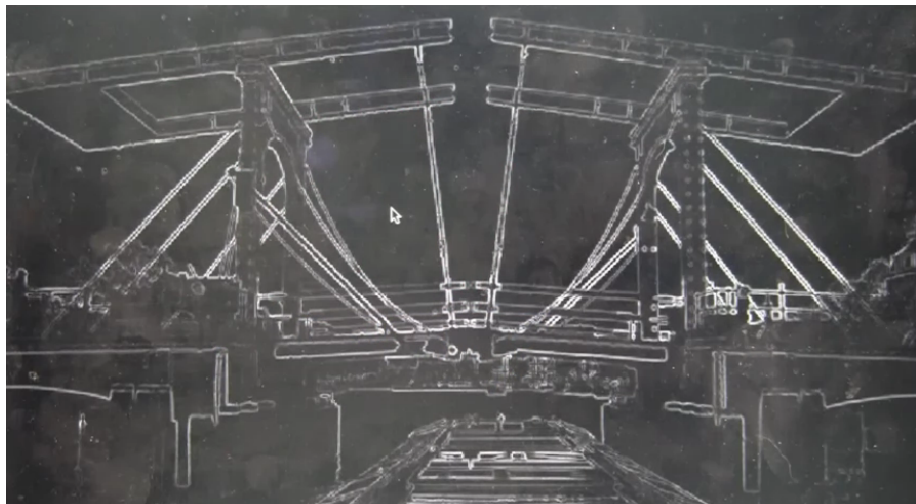
Horizontal and Vertical Gradients

Vertical gradient:



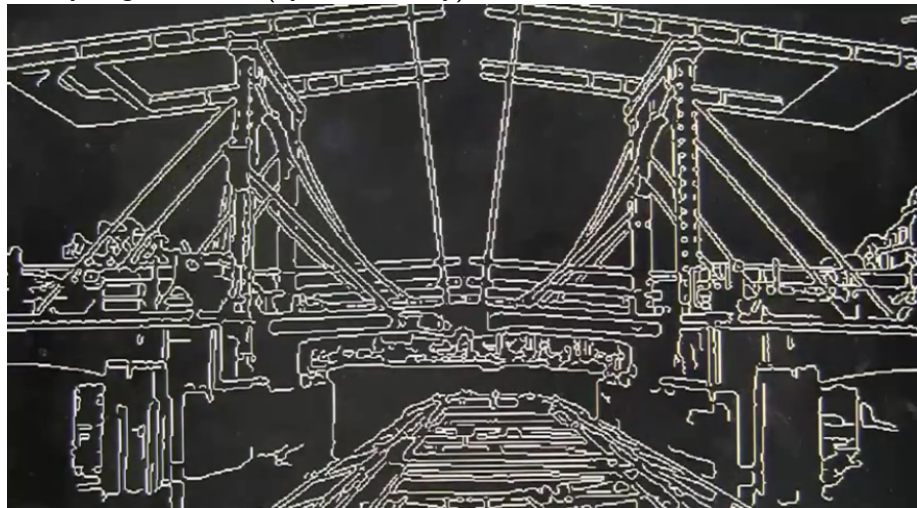
Canny Edge Detector is Uncanny!

Combined gradients:

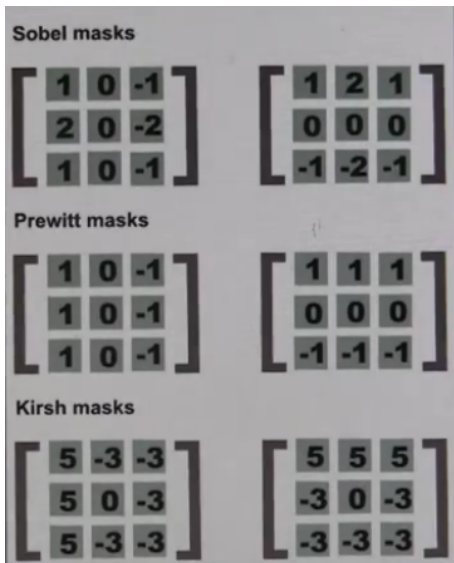


Canny Edge Detector is Uncanny!

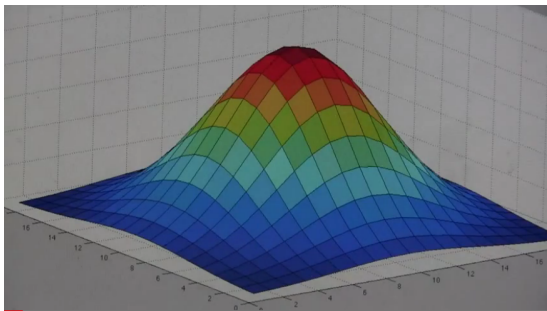
Canny edge detector (by John Canny):



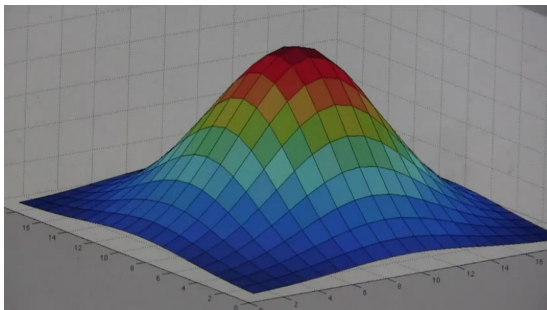
Other Edge Detection Masks



A Gaussian Mask?



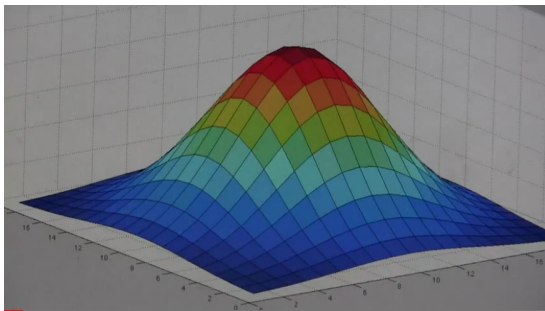
A Gaussian Mask?



What will it do?

- 1 Edge filter
- 2 Dot filter
- 3 Corner
- 4 Blur
- 5 Sharpen

A Gaussian Mask?

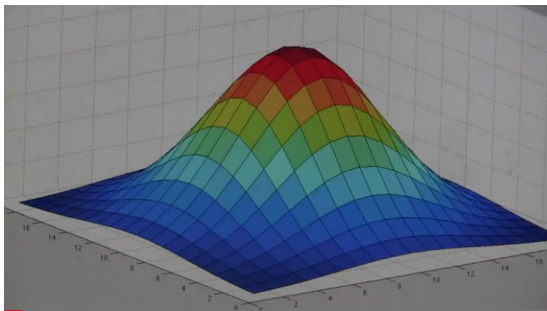


What will it do?

- 1 Edge filter
- 2 Dot filter
- 3 Corner
- 4 **Blur**
- 5 Sharpen

What's the Point of Blurring Images?

A Gaussian Mask?



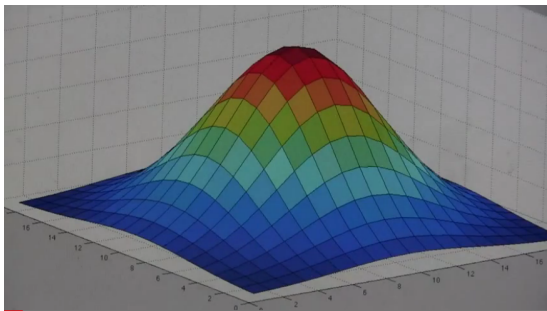
What will it do?

- 1 Edge filter
- 2 Dot filter
- 3 Corner
- 4 **Blur**
- 5 Sharpen

What's the Point of Blurring Images?

- 1 Downsampling

A Gaussian Mask?



What will it do?

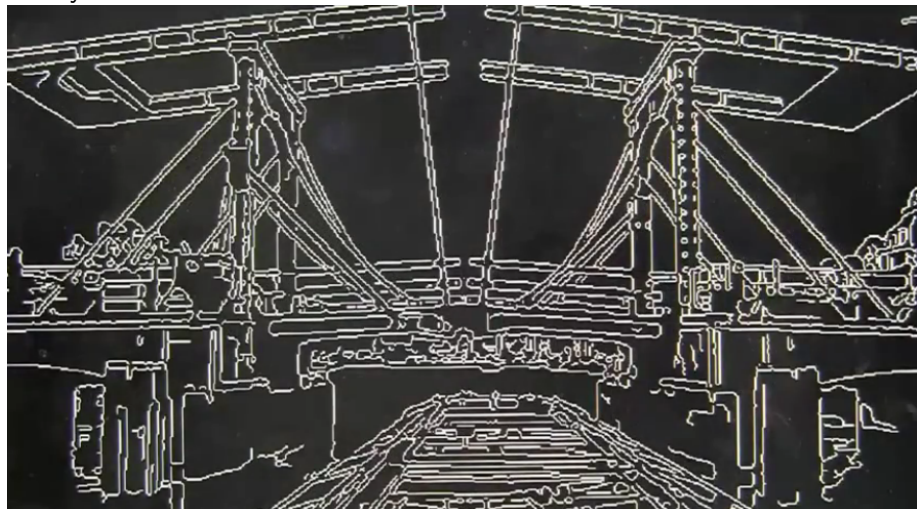
- 1 Edge filter
- 2 Dot filter
- 3 Corner
- 4 **Blur**
- 5 Sharpen

What's the Point of Blurring Images?

- 1 Downsampling
- 2 Noise reduction

Gaussian Mask in Action

Canny filter:



Gaussian Mask in Action

Canny with Gaussian:



$$l' = l \otimes f \otimes g$$

where

f is Gaussian mask and

g is gradient mask.

$$l' = l \otimes f \otimes g$$

where

f is Gaussian mask and

g is gradient mask.

Does the order matter?

Tricks with Linear Filters

$$I' = I \otimes f \otimes g$$

where

f is Gaussian mask and

g is gradient mask.

Does the order matter? **No.** Linear operations are transitive.

$$= I \otimes g \otimes f$$

Tricks with Linear Filters

$$I' = I \otimes f \otimes g$$

where

f is Gaussian mask and

g is gradient mask.

Does the order matter? **No.** Linear operations are transitive.

$$= I \otimes g \otimes f$$

Can we combine them?

Tricks with Linear Filters

$$I' = I \otimes f \otimes g$$

where

f is Gaussian mask and

g is gradient mask.

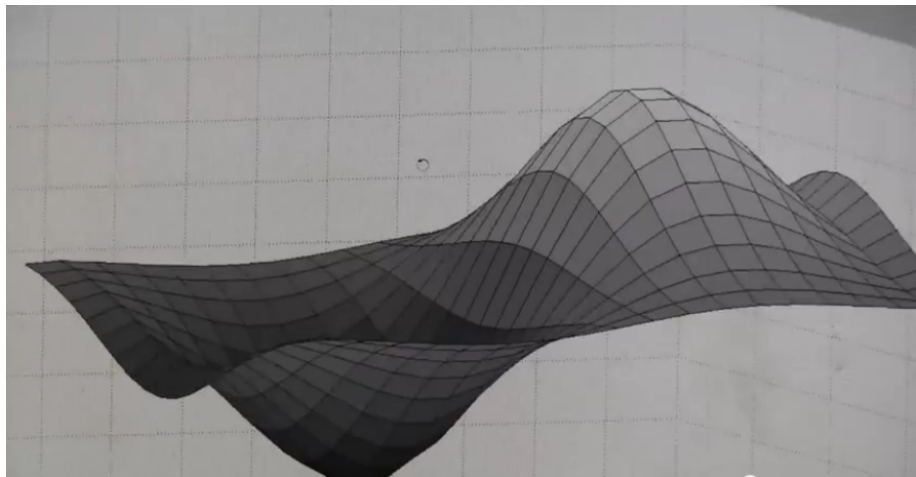
Does the order matter? **No.** Linear operations are transitive.

$$= I \otimes g \otimes f$$

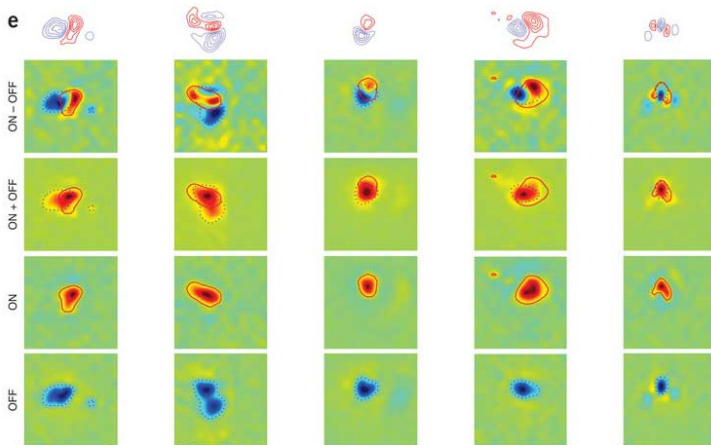
Can we combine them? **Yes.** We'll get a new linear mask/kernel.

$$= I \otimes (f \otimes g)$$

Gaussian Mask Combined with Gradient



Neurons Are Doing Exactly That!

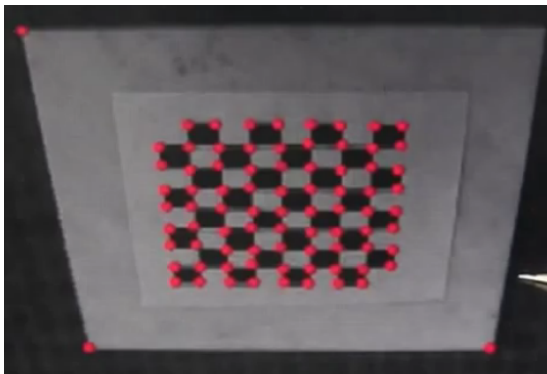


J Jin, Y Wang, HA Swadlow & JM Alonso (2011)

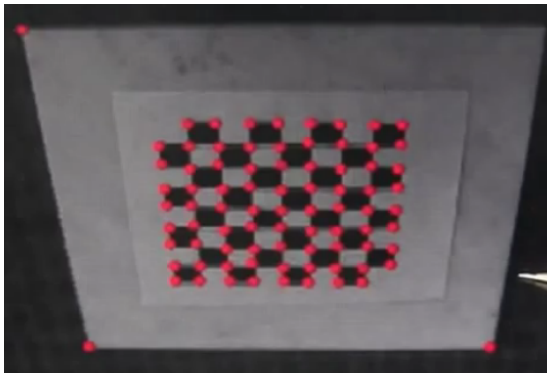
"Population receptive fields of ON and OFF thalamic inputs to an orientation column in visual cortex"

Nature Neuroscience 14(2): 232–238. doi:10.1038/nn.2729

Corner Detection



Corner Detection

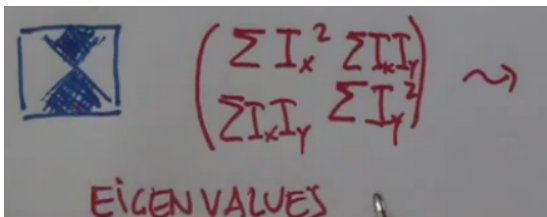
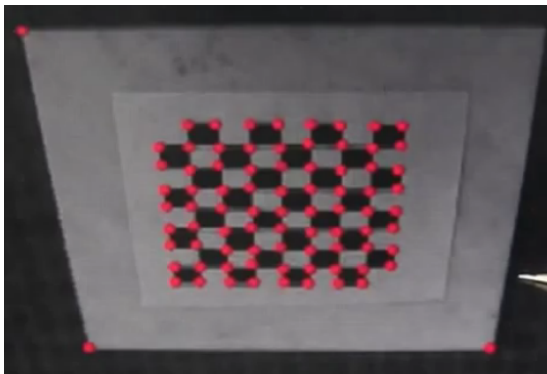


HARRIS CORNER DETECTOR



$$\left. \begin{array}{l} \sum (I_x)^2 \rightarrow \text{LARGE} \\ \sum (I_y)^2 \rightarrow \text{LARGE} \end{array} \right\} \text{CORNER}$$

Corner Detection



Modern Feature Detectors

They are:

- 1 Localizable
- 2 Unique signatures

They are:

- 1 Localizable
- 2 Unique signatures

Two major algorithms:

- 1 HOG: Histogram of Oriented Gradients
- 2 SiFT: Scale-invariant Feature Transform

SiFT in Action

