



Purposes of Image Processing

Visualization (human)

Enhancement, Restoration

Analysis (computer)

Documents, Textures, Biometrics,
Object recognition

There are fundamental differences between them



Lenna





Dr. Yoram Tal



Lenna



5



Dr. Yoram Tal



Lenna 1997

Lena Soderberg (ne Sjooblom) was last reported living in her native Sweden, happily married with three kids and a job with the state liquor monopoly. In 1988, she was interviewed by some Swedish computer related publication, and she was pleasantly amused by what had happened to her picture. That was the first she knew of the use of that picture in the computer business.



6



Dr. Yoram Tal



Sonnet for Lena



Thomas Colthurst



O dear Lena, your beauty is so vast
It is hard sometimes to describe it fast.
I thought the entire world I would impress
If only your portrait I could compress.



Alas! First when I tried to use VQ
I found that your cheeks belong to only you.
Your silky hair contains a thousand lines
Hard to match with sums of discrete cosines.



And your lips, sensual and tactual
Thirteen Crays found not the proper fractal.
And while these setbacks are all quite severe
I might have fixed them with hacks here or there
But when filters took sparkle from your eyes
I said, Damn all this, I ll just digitize.

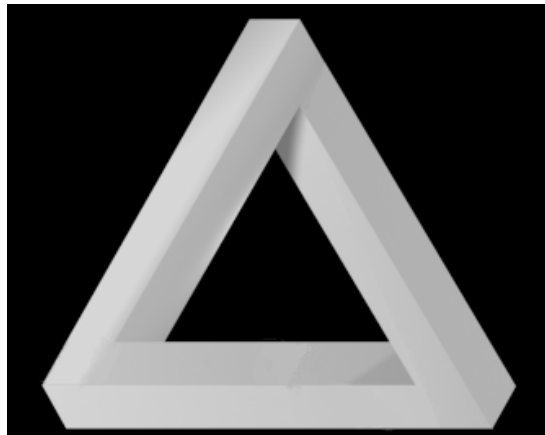
7



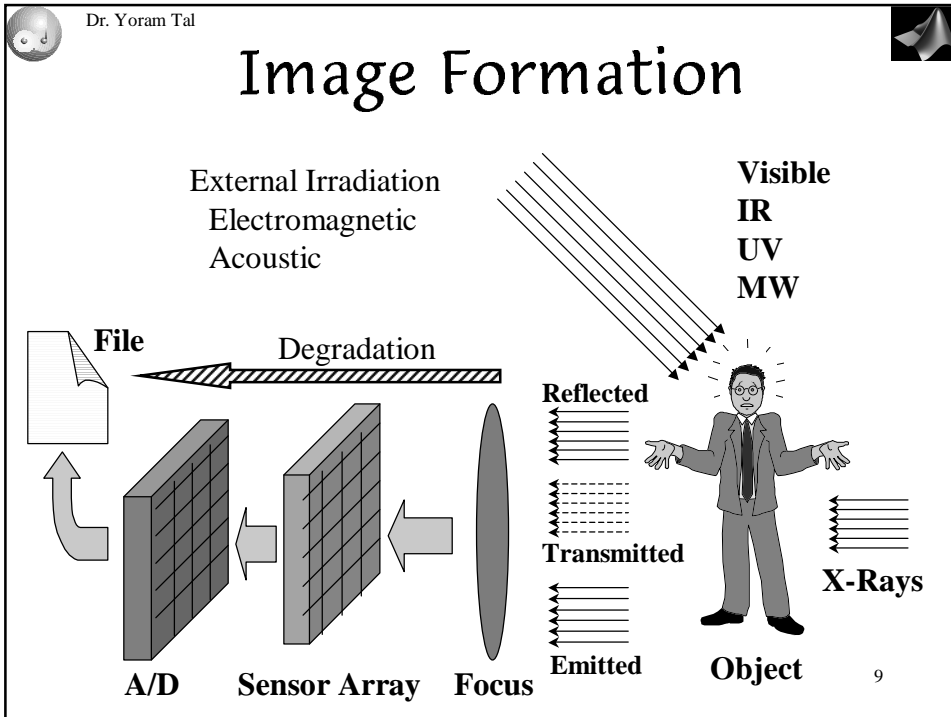
Dr. Yoram Tal



Image Basics



8



Dr. Yoram Tal

Digital Images

A **digital image** is an ordered set of pixels (picture elements, also called *pels*).

A **pixel** is actually a point that has two components: **position** and **color**. Further, pixel position has two components (X,Y) while color is represented by three components, which makes a total of five independent components.

The screen representation of an **image pixel** may contain more (sometimes less) than a single **screen point** (or **screen pixel**) per image pixel

10

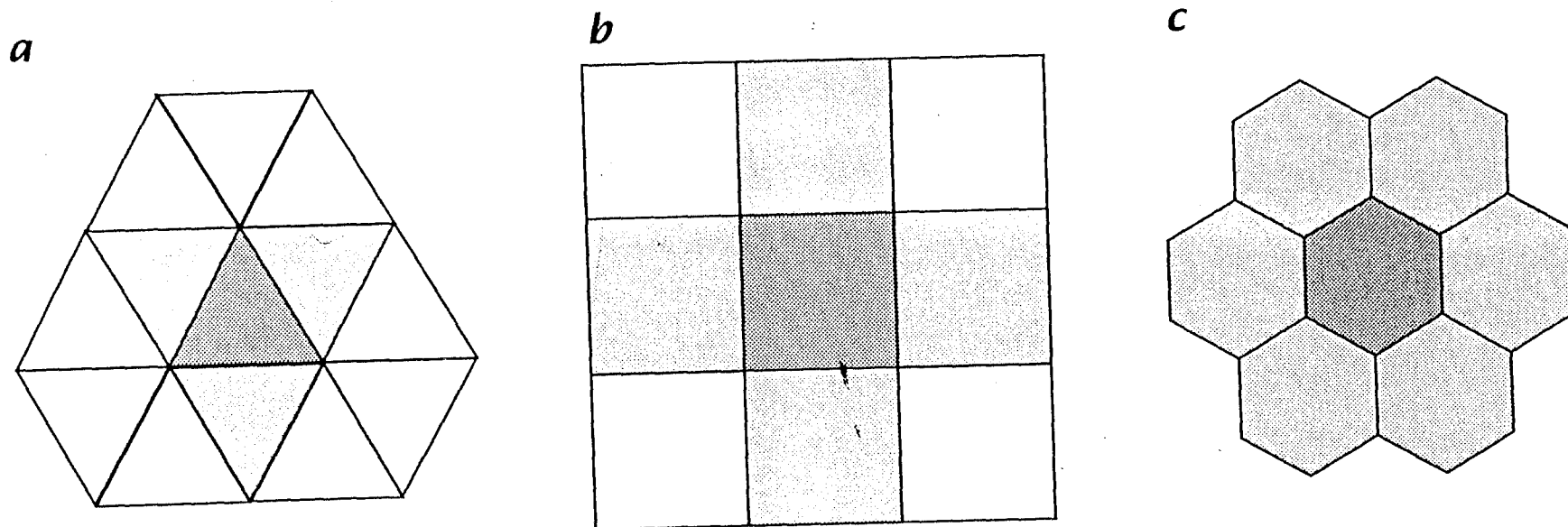


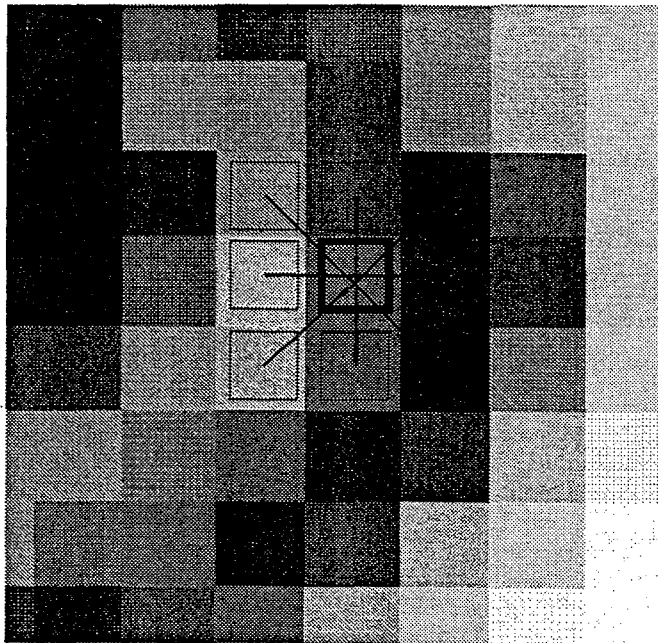
Figure 2.3: The three possible regular grids in 2-D: a triangular grid, b square grid, c hexagonal grid.

Neighborhood

4 neighborhood

8 neighborhood

*In most cases
an image is
represented by
a 2D matrix*



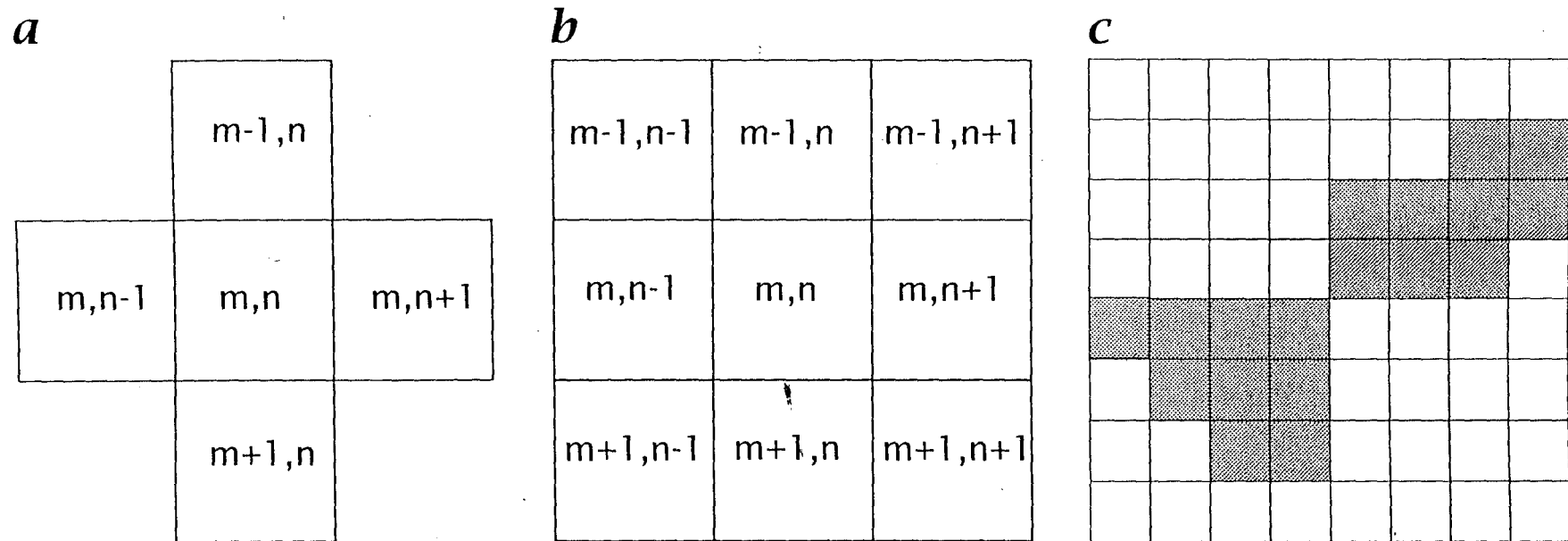


Figure 2.4: Neighborhoods on a rectangular grid: a 4-neighborhood and b 8-neighborhood. c The black region counts as one object (connected region) in an 8-neighborhood but as two objects in a 4-neighborhood.

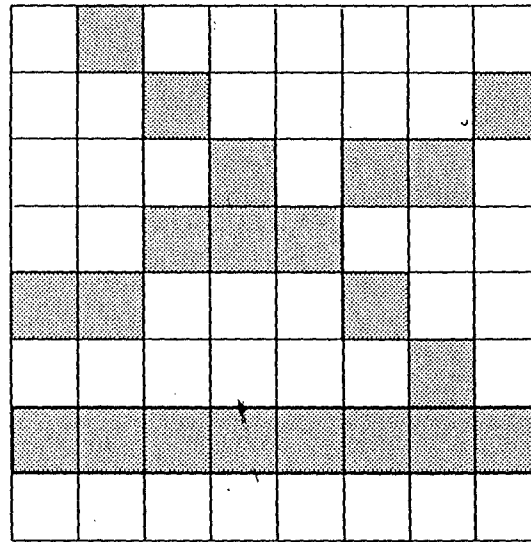


Figure 2.6: A discrete line is only well-defined in the directions of axes and diagonals. In all other directions, a line appears as a staircase-like jagged pixel sequence (exercise 2.2).

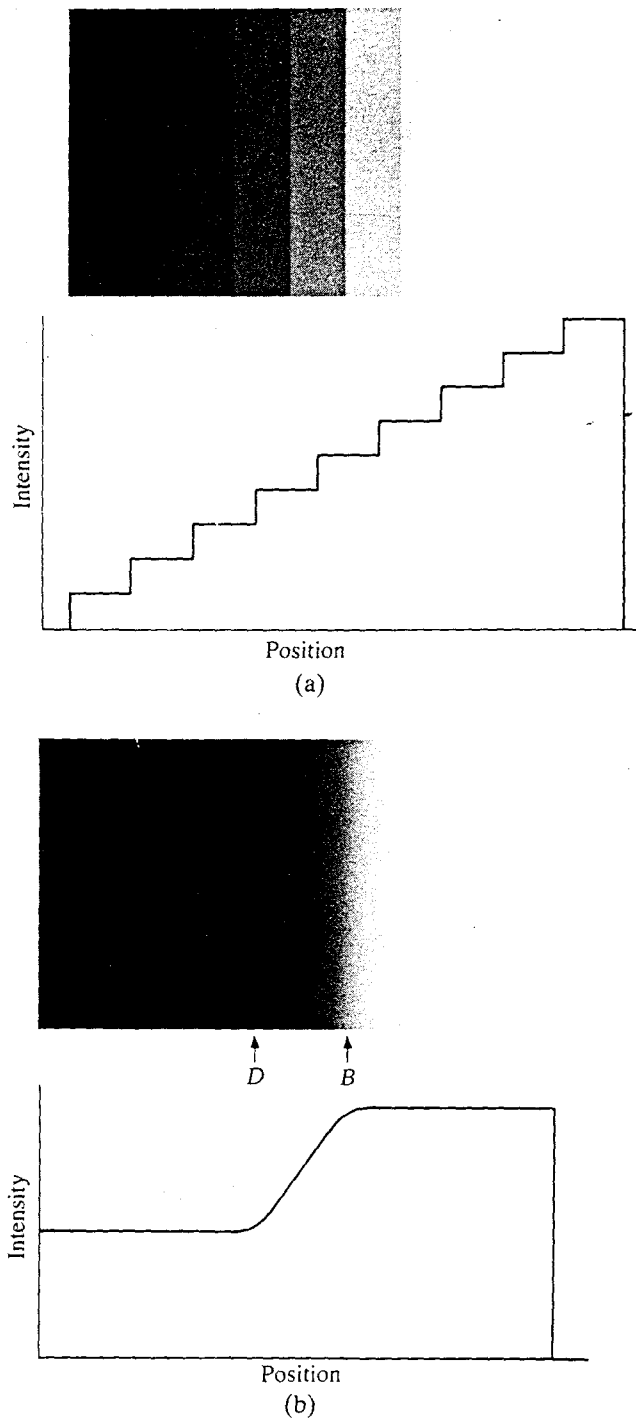
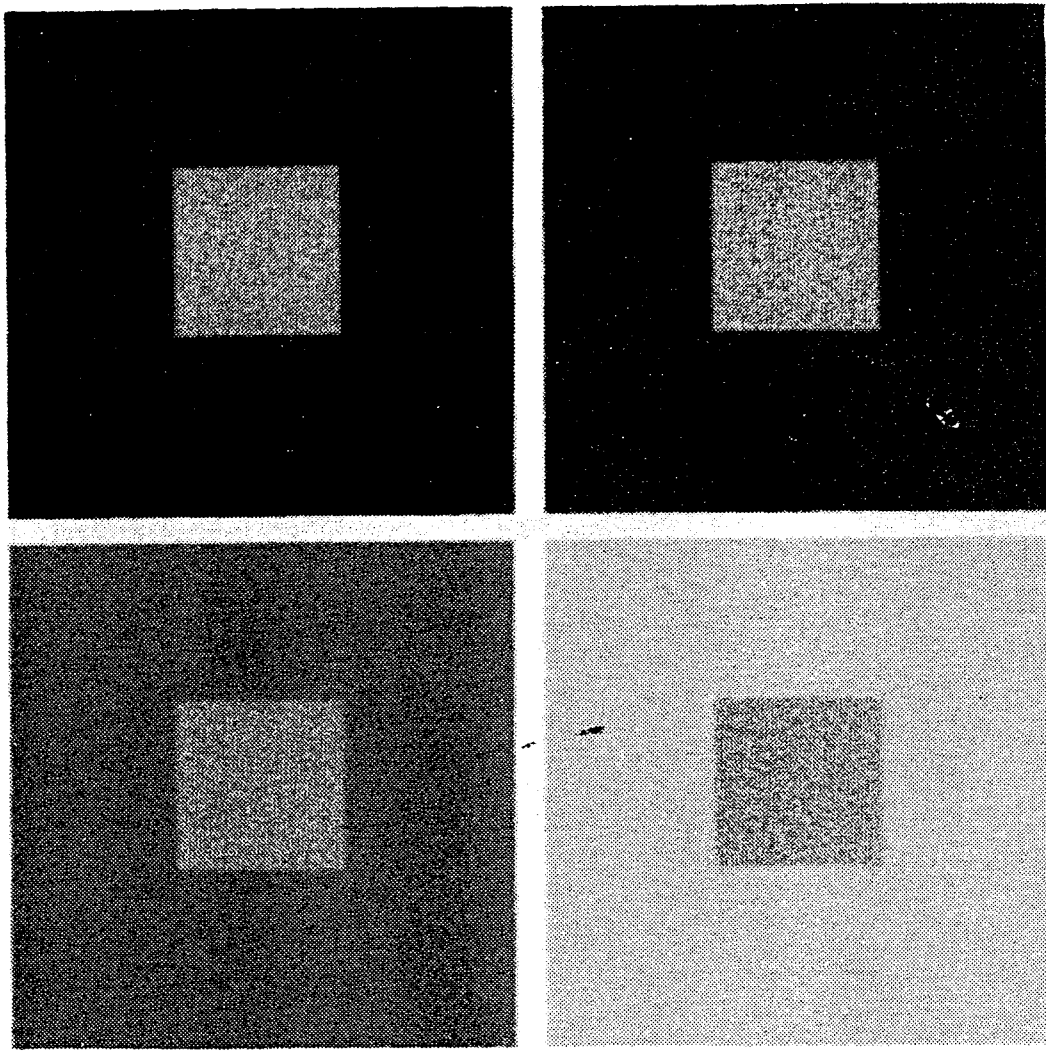


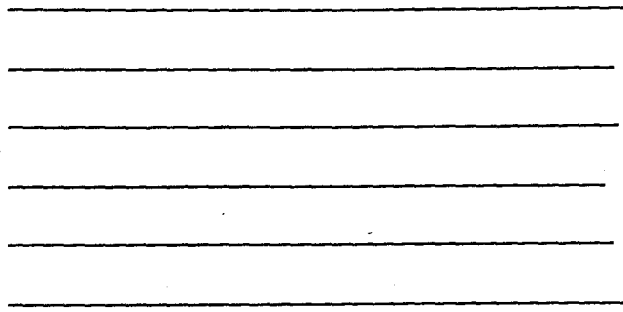
Figure 2.7 Two examples showing that perceived brightness is not a simple function of intensity. (Adapted from Cornsweet [1970].)

Digital Image Fundamentals

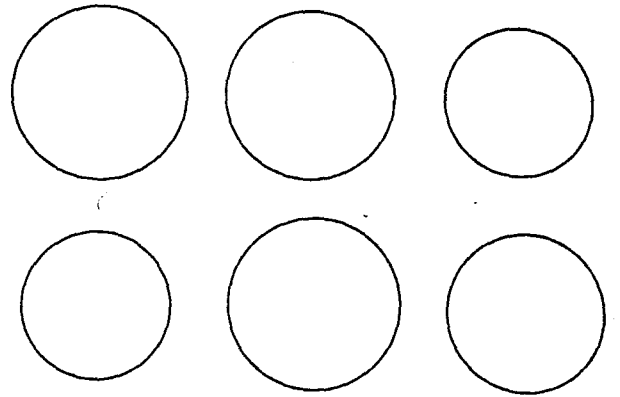


CHAPTER 1. APPLICATIONS AND T

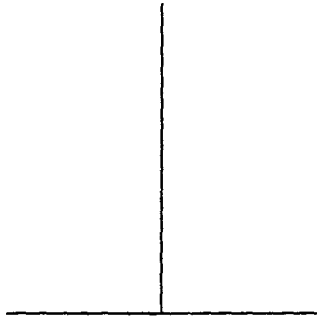
a



b



c



d

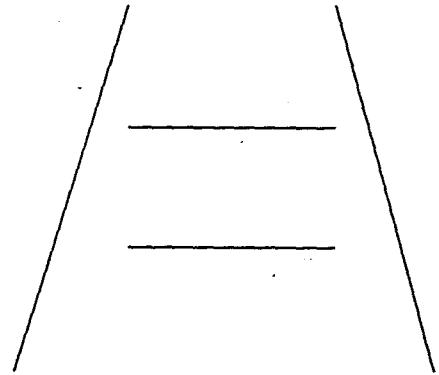


Figure 1.8: Test images for distance and area estimation: *a* parallel lines with up to 5% difference in length; *b* circles with up to 10% difference in radii. *c* A vertical line appears longer, though it has the same length as the horizontal line. *d* Distance estimation by perspective: the upper line (in the background) appears longer than the lower line (in the foreground), though both are equally long.

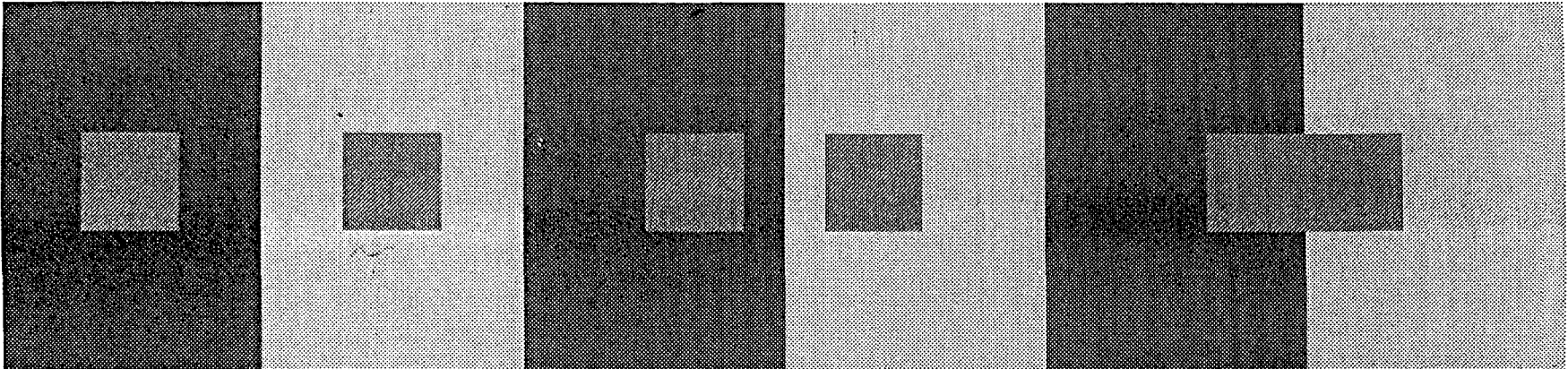
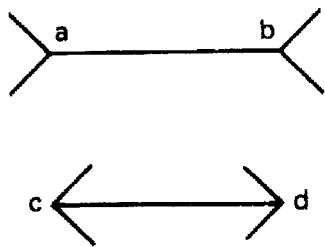


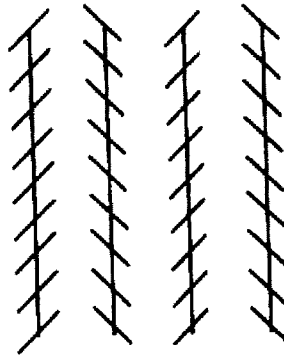
Figure 2.8: The context determines how “bright” we perceive an object to be (exercise 2.4). Both squares have the same brightness, but the square on the dark background appears brighter than the square on the light background. The two squares only appear equally bright if they touch each other.

Which horizontal line is longer?

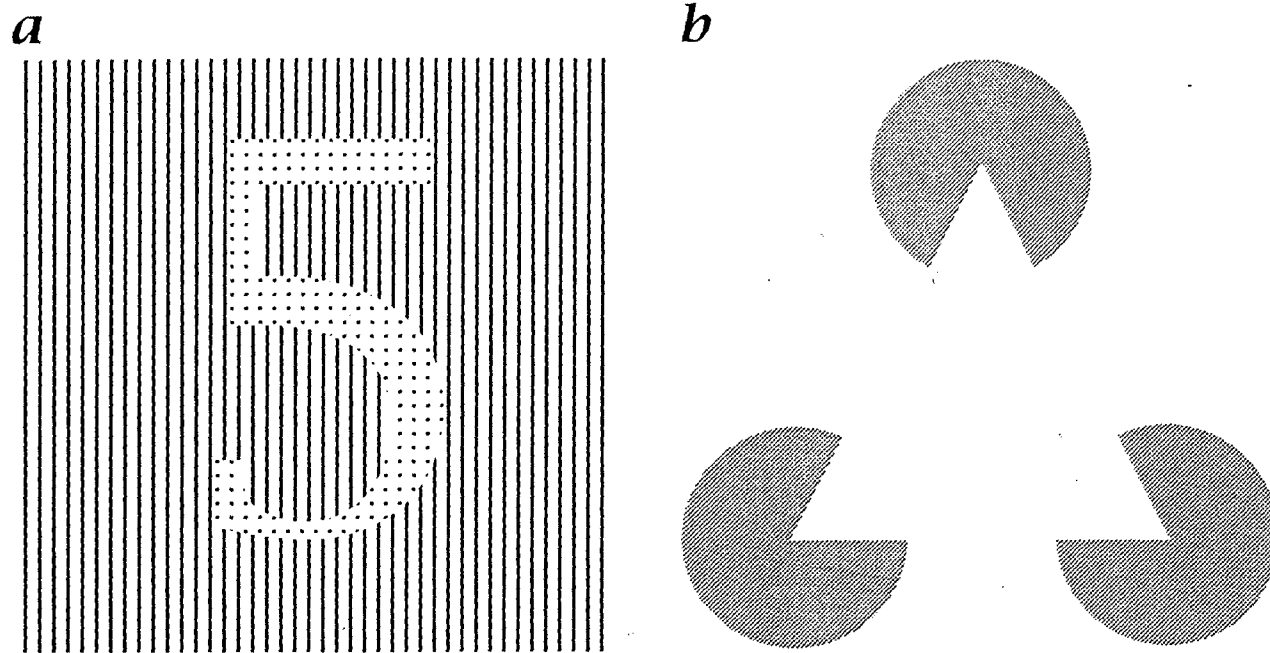


Müller-Lyer Illusion

Are the vertical lines parallel?

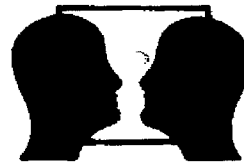
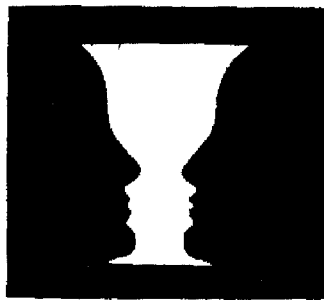


Zöllner Illusion

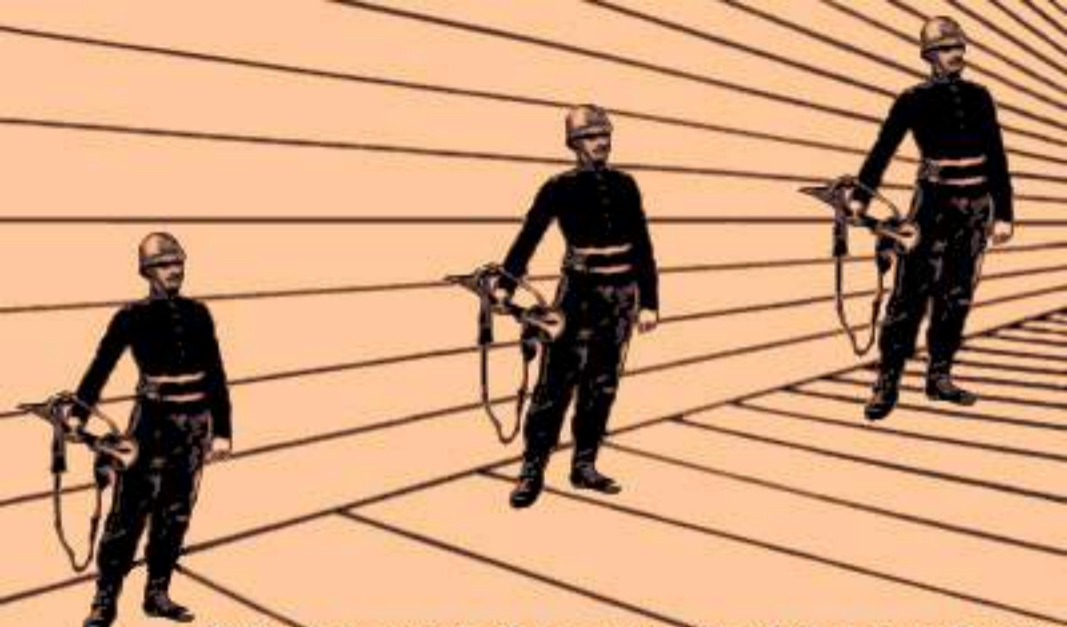


1.10: a Recognition of boundaries between textures; b "interpolation" of boundaries.

A.



Vase or Faces?



Believe it or not, these guys are all the same height

I have my own little world. But that's
OK, they know me here!

Plus, my mail is delivered here!

« [Forrest Gump goes
to Heaven](#)

[Kids on the Subject of Love](#)
»

Stretch your brain

This is not a test - just a phenomenon. All readings are explained. Read out loud the text inside the triangle below.



More than likely you said, "A bird in the bush," and if this is what you said, then you failed to see that the word THE is repeated twice! Sorry, look again.

Next, let's play with some words. What do you see?



In black you can read the word GOOD, in white the word EVIL (inside each black letter is a white letter). It's all very physiological too, because it visualizes the concept that good can't exist without evil (or the absence of good is evil). Now, what do you see?



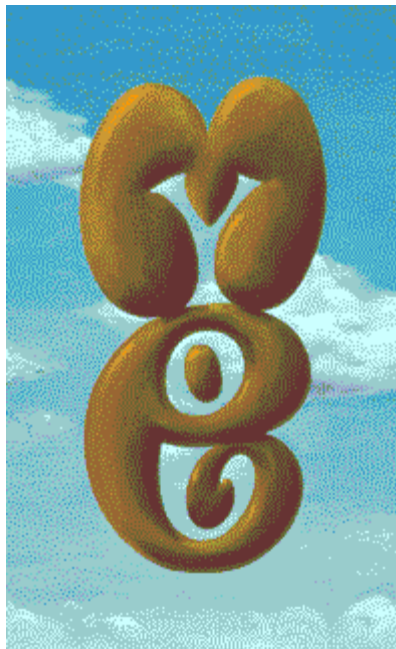
You may not see it at first, but the white spaces read the word optical, the blue landscape reads the word illusion. Look again! Can you see why this painting is called an optical illusion?

What do you see here?



This one is quite tricky! The word TEACH reflects as LEARN.

Last one. What do you see?



You probably read the word ME in brown, but when you look through ME you will see YOU! Do you need to look again?

Test your brain. This is really cool. The second one is amazing so please read all the way though.

ALZHEIMERS' EYE TEST

Count every "F" in the following text:

FINISHED FILES ARE THE RESULT OF YEARS OF SCIENTIFIC STUDY
COMBINED WITH THE EXPERIENCE OF YEARS... (SEE BELOW)

How many? Wrong! There are 6 - no joke! Read it again! Really, go Back and Try to find the 6 F's before you scroll down.

The reasoning behind is further down. The brain cannot process "OF". Incredible or what? Go back and look again! Anyone who counts all 6 "F's" on the first go is a genius. Three is normal, four is quite rare.

More Brain Stuff From Cambridge University

Olny srmata poelpe can raed tihs.

I cdnuolt blveiee taht I cluod aulacly uesdnatnrd waht I was rdanieg. The phaonmneal pweor of the hmuan mnid, aoccdrnig to a rscheearch at Cmabrigde Uinervtisy, it deosn't mtttaer in waht oredr the ltteers in a wrod are, the olny iprmoatnt tihng is taht the frist and lsat ltteer be in the rgh it pclae. The rset can be a taotl mses and you can sitll raed it wouthit a porbelm. Tihs is bcuseae the huamn mnid deos not raed ervey lteter by istlef, but the wrod as a wlohe. Amzanig huh? Yaeh and I awlyas tghuhot slpeling was ipmorantt!

Possibly related posts: (automatically generated)

- » [So you think you know everything?](#)
- » [Mind Boggles](#)
- » [Your Eye Is Slower Than Your Brain](#)
- » [O Iny Srmata Poelpe Can Raed Tihs.](#)

Tags: [brain](#), [illusions](#), [test](#)

This entry was posted on October 15, 2007 at 11:32 pm and is filed under [Random](#). You can follow any responses to this entry through the [RSS 2.0](#) feed. You can [leave a response](#), or [trackback](#) from your own site.

2 Responses to "Stretch your brain"

kaidee33 Says:

October 16, 2007 at 6:17 pm



I like the f one. I counted it a zillion times, before realising about of. lol

godslilrocker7 Says:

October 16, 2007 at 6:28 pm



Haha, me too! Actually I couldn't figure it out so I started to read down and read about "of".

Leave a Reply

You must be [logged in](#) to post a comment.

[Blog at WordPress.com](#) .
[Entries \(RSS\)](#) and [Comments \(RSS\)](#) .

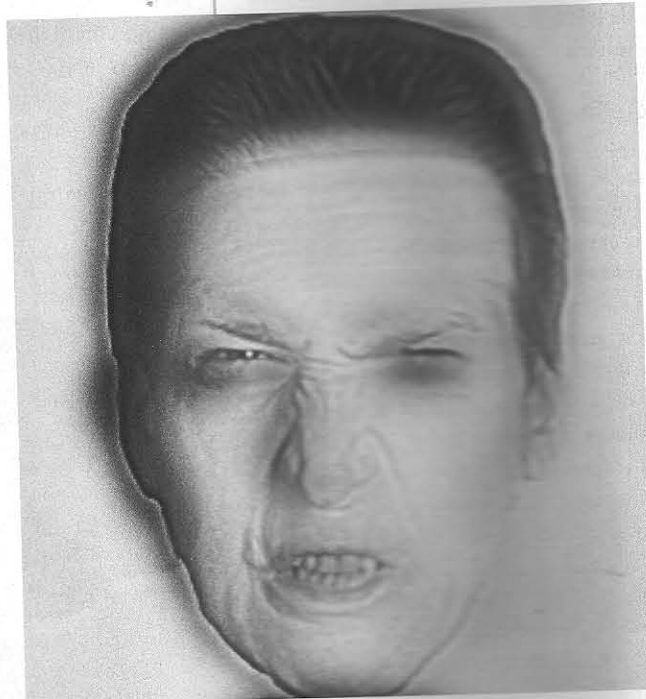
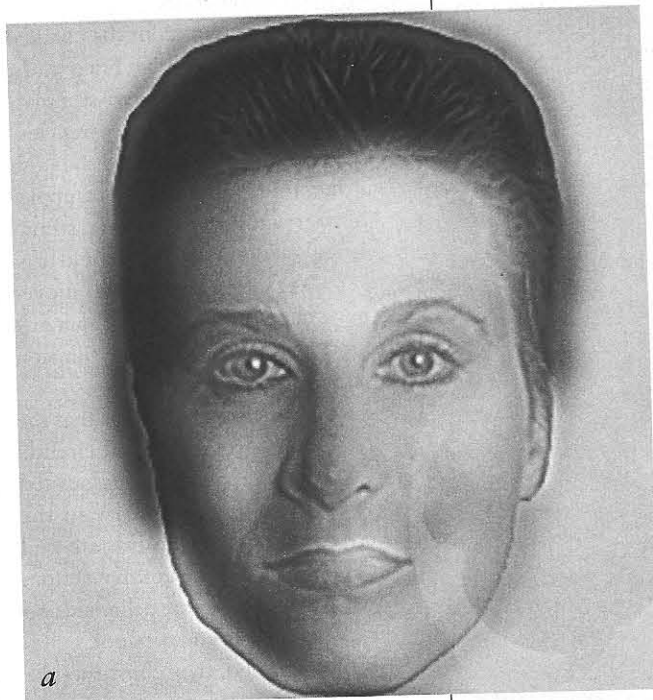


(illusions)

Cracking the da Vinci Code

What do the *Mona Lisa* and President Abraham Lincoln have in common?

BY VILAYANUR S. RAMACHANDRAN AND DIANE ROGERS-RAMACHANDRAN



SPANISH PAINTER EL GRECO often depicted elongated human figures and objects in his work. Some art historians have suggested that he might have been astigmatic—that is, his eyes' corneas or lenses may have been more curved horizontally than vertically, causing the image on the retina at the back of the eye to be stretched vertically. But surely this idea is absurd. If it were true, then we should all be drawing the world upside down, because the retinal image is upside down! (The lens flips the incoming image, and the brain interprets the image on the retina as being right-side up.) The fallacy arises from the flawed reasoning that we literally “see” a picture on the retina, as if we were scanning it with some inner eye.

No such inner eye exists. We need to think, instead, of innumerable visual mechanisms that extract information from the image in parallel and process it stage by stage, before their activity culminates in perceptual experience. As always, we will use some striking illusions to help illuminate the workings of the brain in this processing.

Angry and Calm

Compare the two faces shown in *a*. If you hold the page about nine to 12 inches away, you will see that the face on the right is frowning and the one on the left has a placid expression.

But if you move the figure, so that it is about six or eight feet away, the expressions change. The left one now

smiles, and the right one looks calm.

How is this switch possible? It seems almost magical. To help you understand it, we need to explain how the images were constructed by Philippe G. Schyns of the University of Glasgow and Aude Oliva of the Massachusetts Institute of Technology.

A normal portrait (photographic or painted) contains variations in what neuroscientists such as ourselves term “spatial frequency.” We will discuss two types of spatial frequency: The first is “high”—with sharp, fine lines or details present in the picture. The second is “low”—conveyed by blurred edges or large objects. (In fact, most images contain a spectrum of frequencies ranging from high to low, in varying ratios and

Up close, one face frowns and the other looks calm.
Viewed from farther away, the two faces change. How?

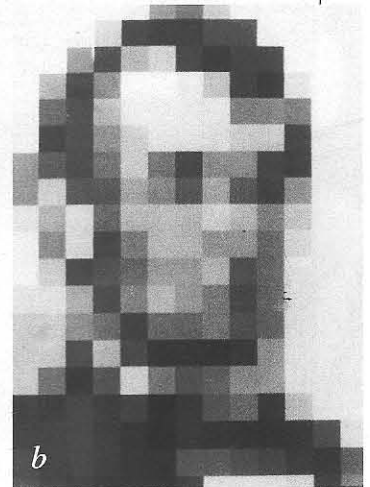
(Squint, and the image blurs, eliminating the sharp edges. Presto! Lincoln becomes instantly recognizable.)

contrasts, but that is not important for the purposes of this column.)

Using computer algorithms, we can process a normal portrait to remove either high or low spatial frequencies. For instance, if we remove high frequencies, we get a blurred image that is said to contain “low spatial frequencies in the Fourier space.” (This mathematical description need not concern us further here.) In other words, this procedure of blurring is called low-pass filtering, because it filters out the high spatial frequencies (sharp edges or fine lines) and lets through only low frequencies. High-pass filtering, the opposite procedure, retains sharp edges and outlines but removes large-scale variations. The result looks a bit like an outline drawing without shading.

These types of computer-processed images are combined together, in an atypical manner, to create the mysterious faces shown in *a*. The researchers began with normal photographs of three faces: one calm, one angry and one smiling. They filtered each face to obtain both high-pass (containing sharp, fine lines) and low-pass (blurred, so as to capture large-scale luminance variations) images. They then combined the high-pass calm face with the low-pass smiling face to obtain the left image. For the right image, they overlaid the high-pass frowning face with the low-pass calm face.

What happens when the figures are viewed close-up? And why do the expressions change when you move the page away? To answer these questions, we need to tell you two more things about visual processing. First, the image needs



Campbell and John Robson of the University of Cambridge: information from different spatial scales is extracted in parallel by various neural channels, which have wide ranges of receptive field sizes. (The receptive field of a visual neuron is the part of the visual field and corresponding tiny patch of retina to which a

stimulus needs to be presented to activate it.) It also shows that the channels do not work in isolation from one another. Rather they interact in interesting ways (for example, the sharp edges picked up by small receptive fields mask the blurred large-scale variations signaled by large receptive fields).

So when you bring the picture near, the sharp features become more visible, masking the coarse features. As a result, the face on the right looks like it is frowning and the one on the left, like it is relaxed. You simply do not notice the opposite emotions that the low spatial frequencies convey. Then, when you move the page farther away, your visual system is no longer able to resolve the fine details. So the expression conveyed by these fine features disappears, and the expression conveyed by low frequencies is unmasked and perceived.

The experiment shows vividly an idea originally postulated by Fergus

stimulus needs to be presented to activate it.) It also shows that the channels do not work in isolation from one another. Rather they interact in interesting ways (for example, the sharp edges picked up by small receptive fields mask the blurred large-scale variations signaled by large receptive fields).

Honest Abe

Experiments of this kind go back to the early 1960s, when Leon Harmon, then working at Bell Laboratories, devised the famous Abraham Lincoln effect. Harmon produced the picture of Honest Abe (*b*) by taking a regular picture and digitizing it into coarse pixels (picture elements). Even when viewed close-up, there is enough information in the blocky brightness variations to recognize Lincoln. But these data, as we noted

REPRINTED WITH PERMISSION OF ALGATEL-LUCENT/BELL LABS (b); GALA CONTEMPLATING THE MEDITERRANEAN SEA WHICH AT TWENTY METERS BECOMES THE PORTRAIT OF ABRAHAM LINCOLN (HOMAGE TO ROTHKO); MUSEO DALI/BRIDGEMAN ART LIBRARY; © 2006 SALVADOR DALI, GALA-SALVADOR DALI FOUNDATION/ARS, NEW YORK (c)
FLEXIBLY MODIFIES THE PERCEPTION OF FACES IN RAPID VISUAL PRESENTATIONS," BY P. G. SCHYNS AND A. OLIVA IN COGNITION, VOL. 69, NO. 3, 1999

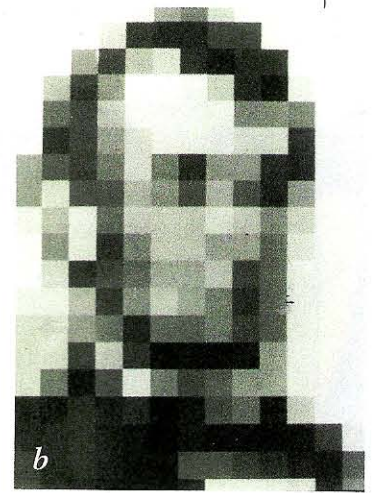
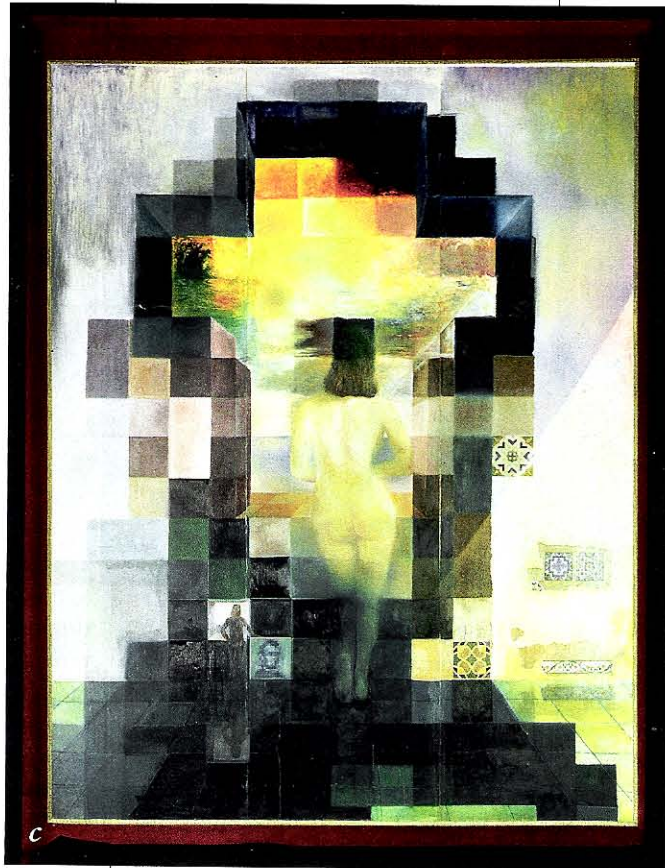
(Squint, and the image blurs, eliminating the sharp edges.)
Presto! Lincoln becomes instantly recognizable.)

contrasts, but that is not important for the purposes of this column.)

Using computer algorithms, we can process a normal portrait to remove either high or low spatial frequencies. For instance, if we remove high frequencies, we get a blurred image that is said to contain “low spatial frequencies in the Fourier space.” (This mathematical description need not concern us further here.) In other words, this procedure of blurring is called low-pass filtering, because it filters out the high spatial frequencies (sharp edges or fine lines) and lets through only low frequencies. High-pass filtering, the opposite procedure, retains sharp edges and outlines but removes large-scale variations. The result looks a bit like an outline drawing without shading.

These types of computer-processed images are combined together, in an atypical manner, to create the mysterious faces shown in *a*. The researchers began with normal photographs of three faces: one calm, one angry and one smiling. They filtered each face to obtain both high-pass (containing sharp, fine lines) and low-pass (blurred, so as to capture large-scale luminance variations) images. They then combined the high-pass calm face with the low-pass smiling face to obtain the left image. For the right image, they overlaid the high-pass frowning face with the low-pass calm face.

What happens when the figures are viewed close-up? And why do the expressions change when you move the page away? To answer these questions, we need to tell you two more things about visual processing. First, the image needs



Campbell and John Robson of the University of Cambridge: information from different spatial scales is extracted in parallel by various neural channels, which have wide ranges of receptive field sizes. (The receptive field of a visual neuron is the part of the visual field and corresponding tiny patch of retina to which a

stimulus needs to be presented to activate it.) It also shows that the channels do not work in isolation from one another. Rather they interact in interesting ways (for example, the sharp edges picked up by small receptive fields mask the blurred large-scale variations signaled by large receptive fields).

Honest Abe

Experiments of this kind go back to the early 1960s, when Leon Harmon, then working at Bell Laboratories, devised the famous Abraham Lincoln effect. Harmon produced the picture of Honest Abe (*b*) by taking a regular picture and digitizing it into coarse pixels (picture elements). Even when viewed close-up, there is enough information in the blocky brightness variations to recognize Lincoln. But these data, as we noted

to be close for you to see the sharp features. Second, sharp features, when visible, “mask”—or deflect attention away from—the large-scale objects (low spatial frequencies).

So when you bring the picture near, the sharp features become more visible, masking the coarse features. As a result, the face on the right looks like it is frowning and the one on the left, like it is relaxed. You simply do not notice the opposite emotions that the low spatial frequencies convey. Then, when you move the page farther away, your visual system is no longer able to resolve the fine details. So the expression conveyed by these fine features disappears, and the expression conveyed by low frequencies is unmasked and perceived.

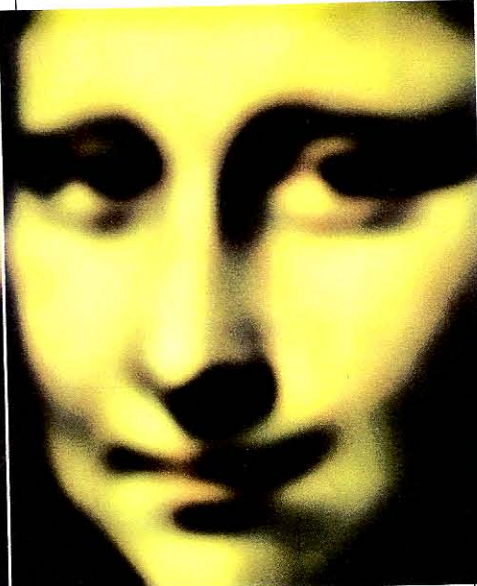
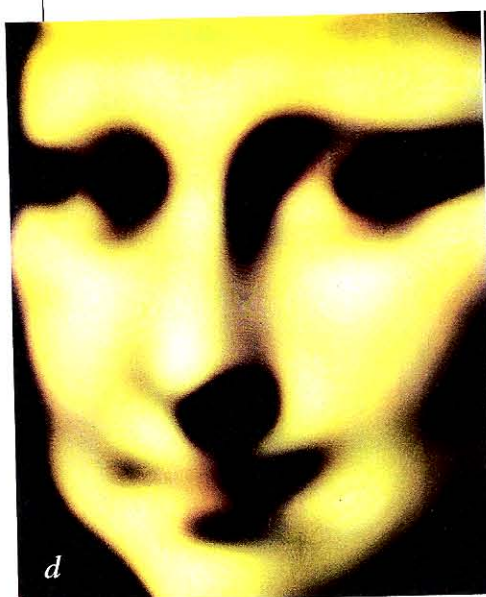
The experiment shows vividly an idea originally postulated by Fergus

REPRINTED WITH PERMISSION OF ALCATEL-LUCENT/BELL LABS (b); GALA CONTEMPLATING THE MEDITERRANEAN SEA WHICH AT TWENTY METERS BECOMES THE PORTRAIT OF ABRAHAM LINCOLN (HOMAGE TO ROTHKO); MUSEO DALI/BRIDGEMAN ART LIBRARY; © 2006 SALVADOR DALI FOUNDATION/ARF, NEW YORK (c)

FLEXIBLY MODIFIES THE PERCEPTION OF FACES IN RAPID VISUAL PRESENTATIONS." BY P. G. SCHYNS AND A. OULVA IN COGNITION, VOL. 69, NO. 3, 1999

(illusions)

The **elusive smile** can be seen only when you look away from the mouth. Attend to it out of the corner of your eye.



already, are masked by the sharp edges of the pixels. When you move far away from the photograph or squint, the image blurs, eliminating the sharp edges. Presto! Lincoln becomes instantly recognizable. The great artist Salvador Dalí was sufficiently inspired by this illusion to use it as a basis for his paintings, an unusual juxtaposition of art and science (c).

Mysterious *Mona Lisa*

Finally, consider the mysterious smile of Leonardo da Vinci's *Mona Lisa*. Philosophers and art historians who specialize in aesthetics often refer to her expression as "enigmatic" or "elusive," mainly because they do not understand it. Indeed, we wonder whether they prefer not to understand it, because they seem to resent any attempts to explain it scientifically, apparently for fear that such analysis might detract from its beauty.

But recently neurobiologist Margaret Livingstone of Harvard Medical School made an intriguing observation; she cracked the da Vinci code, you might say. She noticed that when she looked directly at Mona Lisa's mouth (d, center panel), the smile was not apparent (quite a disappointment). Yet as she moved her

gaze away from the mouth, the smile appeared, beckoning her eyes back. Looking again at the mouth, she saw that the smile disappeared again. In fact, she noted, the elusive smile can be seen only when you look away from the mouth. You have to attend to it out of the corner of your eye, rather than fixating on it directly. Because of the unique shading (placement of low spatial frequencies) at the corners of the mouth, a smile is perceived only when the low spatial frequencies are dominant—that is, when you look indirectly at the masterpiece.

To confirm this notion, she performed a low-pass filtering (left panel) and a high-pass filtering (right panel) of the *Mona Lisa*. Notice that with the low-pass (blurred) image the smile is more obvious than in the original—it can be seen even if you look directly at the mouth. With the high-pass (outlinelike) image, however, no smile is apparent, even if you look away from the mouth. Putting these two images back together restores the

original masterpiece and the elusive nature of the smile. As with the changing faces, we can now better appreciate what Leonardo seems to have stumbled on and fallen in love with—a portrait that seems alive because its fleeting expression (thanks to quirks of our visual system) perpetually tantalizes the viewer.

Taken collectively, these experiments show that there is more to perception than what meets the eye. More specifically, they demonstrate that information at different scales, such as fine versus coarse, may be extracted initially from an image by separate neural channels and recombined at different stages of processing to create the final impression of a single unified picture in your mind. **M**

VILAYANUR S. RAMACHANDRAN and DIANE ROGERS-RAMACHANDRAN are at the Center for Brain and Cognition at the University of California, San Diego. This column is reprinted from an earlier issue of *Scientific American Mind*.

(Further Reading)

◆ **Dr. Angry and Mr. Smile: When Categorization Flexibly Modifies the Perception of Faces in Rapid Visual Presentations.** Philippe G. Schyns and Aude Oliva in *Cognition*, Vol. 69, No. 3, pages 243–265; 1999.

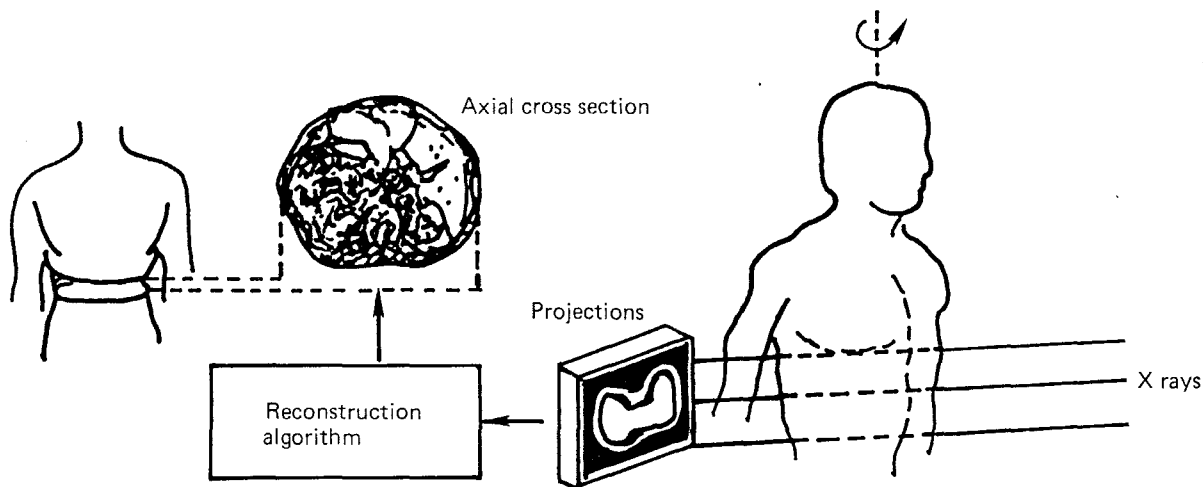


Figure 1.9 Image reconstruction using X-ray CT scanners.

1.7 IMAGE DATA COMPRESSION

The amount of data associated with visual information is so large (see Table 1.1a) that its storage would require enormous storage capacity. Although the capacities of several storage media (Table 1.1b) are substantial, their access speeds are usually inversely proportional to their capacity. Typical television images generate data rates exceeding 10 million bytes per second. There are other image sources that generate even higher data rates. Storage and/or transmission of such data require large capacity and/or bandwidth, which could be very expensive. *Image data compression techniques* are concerned with reduction of the number of bits required to store or transmit images without any appreciable loss of information. Image trans-

TABLE 1.1a Data Volumes of Image Sources
(in Millions of Bytes)

National archives	12.5×10^9
1 h of color television	28×10^3
Encyclopaedia Britannica	12.5×10^3
Book (200 pages of text characters)	1.3
One page viewed as an image	.13

TABLE 1.1b Storage Capacities
(in Millions of Bytes)

Human brain	125,000,000
Magnetic cartridge	250,000
Optical disc memory	12,500
Magnetic disc	760
2400-ft magnetic tape	200
Floppy disc	1.25
Solid-state memory modules	0.25



RGB Images

	0.2235	0.1294	Blue	0.4196	0.2588	0.1608	0.2588	0.1608	0.2588	0.1608
0.5804	0.2902	0.0627	0.2902	0.2902	0.4824	0.2235	0.2588	0.1608	0.2588	0.1608
0.5804	0.0627	0.0627	0.0627	0.2235	0.2588	0.1608	0.2588	0.1608	0.2588	0.1608
0.5176	0.1922	0.0627	Green	0.1922	0.2588	0.2588	0.1608	0.2588	0.1608	0.2588
0.5176	0.1294	0.1608	0.1294	0.1294	0.2588	0.2588	0.1608	0.2588	0.1608	0.2588
0.5176	0.1608	0.0627	0.1608	0.1922	0.2588	0.2588	0.1608	0.2588	0.1608	0.2588
0.5490	0.2235	0.5490	Red	0.7412	0.7765	0.7765	0.1608	0.2588	0.1608	0.2588
0.5490	0.3882	0.5176	0.5804	0.5804	0.7765	0.7765	0.1608	0.2588	0.1608	0.2588
0.5490	0.2588	0.2902	0.2588	0.2235	0.4824	0.2235	0.1608	0.2588	0.1608	0.2588
0.5490	0.2235	0.1608	0.2588	0.2588	0.1608	0.2588	0.1608	0.2588	0.1608	0.2588
0.5490	0.1608	0.2588	0.2588	0.2588	0.2588	0.2588	0.1608	0.2588	0.1608	0.2588



RGB images

Three intensity matrices: **R**, **G**, and **B**

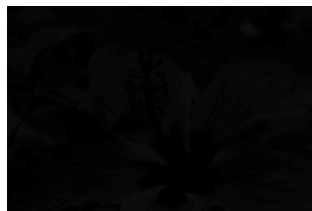
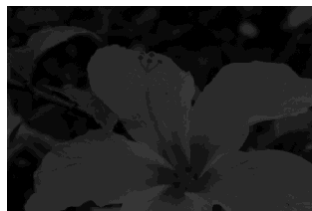




Image Display

- image** - Create and display image object.
- imagesc** - Scale data and display as image.
- imshow** - Display image.

- colorbar** - Display colorbar.
- getimage** - Get image data from axes.
- truesize** - Adjust display size of image.
- warp** - Display image as texture-mapped surface.
- zoom** - Zoom in and out of image or 2-D plot.



Image I/O

- imread** - Read image type files.
- imwrite** - Write image type files.
- imshow** - Load and display an image.
- uiimport** - Interactive loading of different files (MATLAB 6).
- iminfo** - image file info (IP toolbox 3)



Image Files Structure

.tif .bmp .jpg .png .pcx .gif

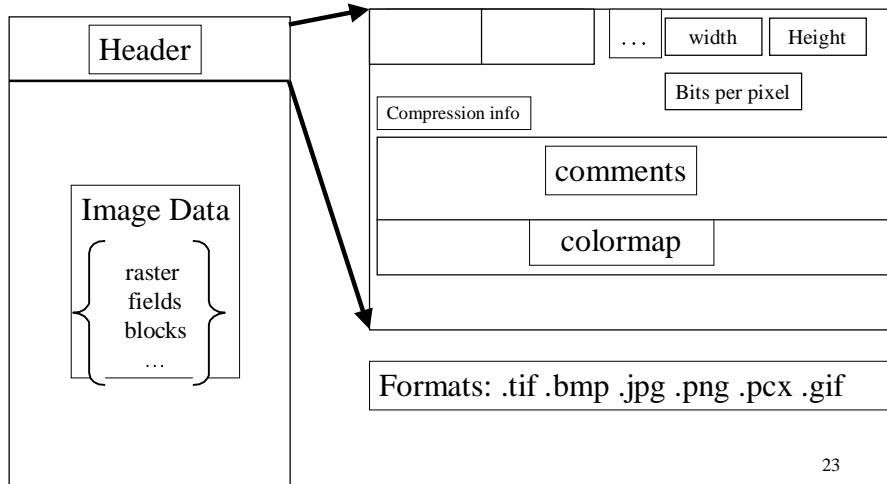
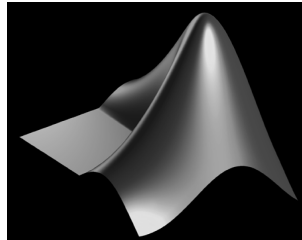


Image Conversion

- gray2ind** - intensity image to indexed image.
- im2bw** - image to binary image by thresholding.
- Im2double** - image to double precision.
- im2uint8** - image to 8-bit unsigned integers.
- Im2uint16** - image to 16-bit unsigned integers.
- ind2gray** - indexed image to intensity image.
- ind2rgb** - indexed image to RGB image.
- mat2gray** - matrix to intensity image.
- rgb2gray** - RGB image or colormap to grayscale.
- rgb2ind** - RGB image to indexed image.



MATLAB programming



MATLAB Environment





Important Concepts

Selection
Vectorization
Sparse Matrices



Selection (1)

**Matrices are stored column wise,
in contiguous memory blocks**

Matrix:

a	b	c
d	e	f
g	h	i

Memory:

a	d	g	b	e	h	c	f	i
---	---	---	---	---	---	---	---	---



Selection (2)

Arrays vs. Linear reference

$$M = \begin{array}{|c|c|c|} \hline 8 & 1 & 6 \\ \hline 3 & 5 & 7 \\ \hline 4 & 9 & 2 \\ \hline \end{array}$$

$M(1,3) = 6$

$M(3,2) = 9$

$M(5) = 5$

$M(:)$ = all elements of M

in a column vector

$M(1,:) = [8, 1, 6]$

$$M(:,2) = \begin{bmatrix} 1 \\ 5 \\ 9 \end{bmatrix}$$

Functions:
sub2ind
ind2sub



Selection (3)

Arrays vs. Linear reference

$$M = \begin{array}{|c|c|c|} \hline 8 & 1 & 6 \\ \hline 3 & 5 & 7 \\ \hline 4 & 9 & 2 \\ \hline \end{array}$$

$M([1\ 2],3) = \begin{bmatrix} 6 \\ 7 \end{bmatrix}$

$M(1:2:7) = [8, 4, 5, 6]$

$M([1\ 2],[2\ 3]) = \begin{bmatrix} 1 & 6 \\ 5 & 7 \end{bmatrix}$

$M([1\ 2],[3\ 3\ 3]) = \begin{bmatrix} 6 & 6 & 6 \\ 7 & 7 & 7 \end{bmatrix}$



Selection (4)

$$M = \begin{array}{|c|c|c|} \hline 8 & 1 & 6 \\ \hline 3 & 5 & 7 \\ \hline 4 & 9 & 2 \\ \hline \end{array}$$

$$L = \begin{array}{|c|} \hline 0 \\ \hline 22 \\ \hline 4 \\ \hline -1 \\ \hline 3.1 \\ \hline 5 \\ \hline 1 \\ \hline 3 \\ \hline 3 \\ \hline \end{array}$$

What is $R = L(M)$?

$$R = \begin{array}{|c|c|c|} \hline 3 & 0 & 5 \\ \hline 4 & 3.1 & 1 \\ \hline -1 & 3 & 22 \\ \hline \end{array}$$

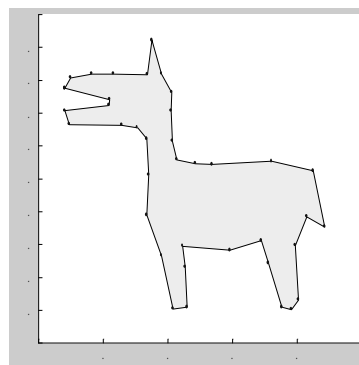
R is the result of a selection M from a Look Up table L



Vectorization (1)

Area of a polygon

	x	y
1.	0.0804	0.7751
2.	0.0986	0.8062
3.	0.1626	0.8206
4.	0.2311	0.8206
5.	0.3361	0.8182
6.	0.3498	0.9211
7.	0.3817	0.8206
8.	0.4114	0.7632
9.	0.4091	0.7081



$$2S = \sum_{i=1}^n (x_{i+1}y_i - x_i y_{i+1})$$

n vertices
 $\mathbf{x}(n+1) = \mathbf{x}(1)$
 $\mathbf{y}(n+1) = \mathbf{y}(1)$



Vectorization (2)

option #1: Loop

```

% close polygon (if not closed already)
x(end+1) = x(1);
y(end+1) = y(1);
s = 0;

% loop
for i = 1:n,
    s = s + x(i+1)*y(i) - x(i)*y(i+1);
end

s = s/2;

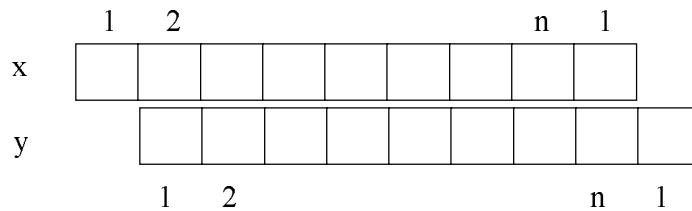
```



Vectorization (3)

option #2: element by element product

$$2S = x_2y_1 + x_3y_2 + \dots + x_ny_{n-1} - x_1y_2 - x_2y_3 - \dots - x_{n-1}y_n$$



```

s = sum(x(2:end).*y(1:end-1) - x(1:end-1).*y(2:end));
s = s/2;

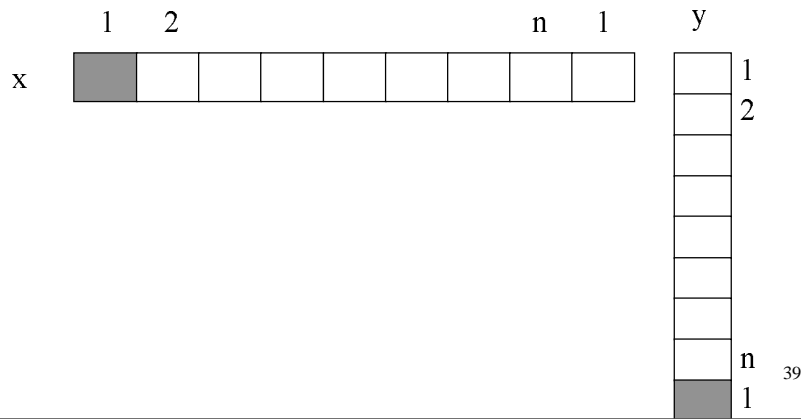
```



Vectorization (4)

option #3: inner product

```
s = x(2:end) * y(1:end-1) - x(1:end-1) * y(2:end);
s = s/2;
```



DMpoly



Sparse Matrices



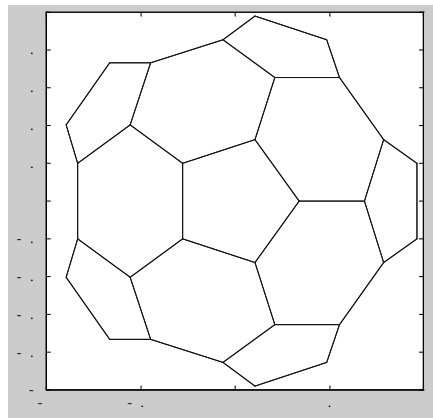
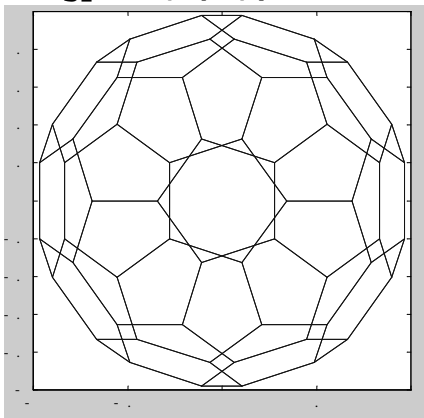
Sparse Matrices

The Bucky ball graph

(Buckminster Fuller - architect, mathematician, inventor, ...)

```
[B v] = bucky;  
gplot(B,v);
```

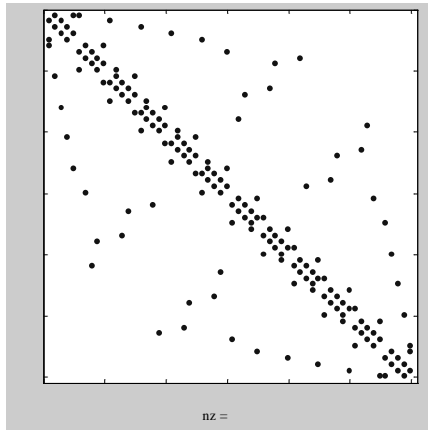
half bucky





Adjacency Matrices

B is the adjacency (connectivity) matrix of the Bucky-ball.



$$B(i, j) = \begin{cases} 1 & \text{if } i \text{ and } j \text{ are connected} \\ 0 & \text{otherwise} \end{cases}$$

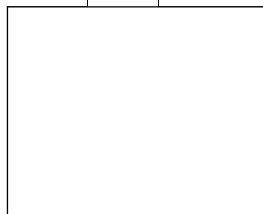
`spy(B)`

Sparsity = 5%



Sparse Matrices: Representation & Syntax

Full



Sparse

i	j	s
1	2	2
1	3	1
2	3	5
3	4	7
		.
2	10	3

Syntax

`S = sparse(A)`
`S = sparse(i,j,s, m,n)`
`S = sparse(i,j,s)`

Any elements of `s` that have duplicate values of `i` and `j` are added together.

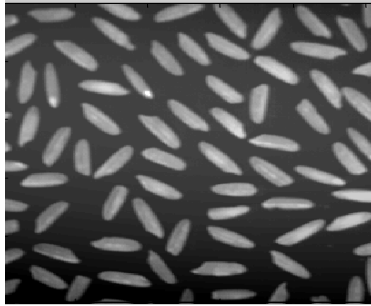
i.e., $(i=2, j=5, s=10)$ and $(i=2, j=5, s=11)$ yield $S(2,5) = 21$



Sparse Matrix Applications

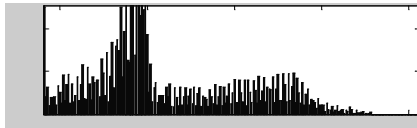
1. Histograms

im (image matrix)



The value of each pixel in this image is an integer in the range [0, 255]. Compute the number of pixels for each gray-level

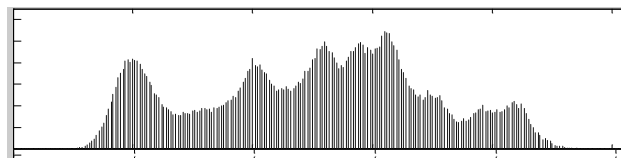
```
S = sparse(im(:),1,1);
```



bar(S)



Histograms





Histograms

Given an array of numbers:

$$A = \begin{Bmatrix} 2 & 9 & 4 & 5 & 2 & 2 & 5 & 6 & 1 \\ 6 & 9 & 3 & 7 & 3 & 5 & 4 & 2 & 0 \\ 3 & 7 & 8 & 0 & 7 & 7 & 5 & 4 & 8 \\ 9 & 4 & 0 & 6 & 6 & 5 & 3 & 0 & 1 \\ 7 & 4 & 7 & 0 & 4 & 6 & 4 & 9 & 2 \end{Bmatrix}$$

Find the frequency of each number:

Number:	0	1	2	3	4	5	6	7	8	9
Count:	5	2	5	4	7	5	5	6	2	4



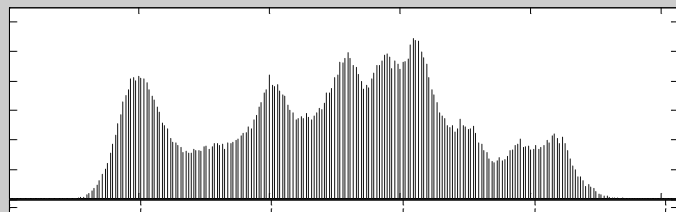
Histogram (Normal)



x image matrix (intensity)

```
imhist(x) display histogram
[count, bin] = imhist(x);
```

Count
pixels



Gray level



Histograms (cont.)

The histogram of an image contains valuable information concerning the distribution of gray levels

It does not contain any spatial information

All the following images have exactly the same histograms!



49



Histograms as Voting

A histogram is a result of voting it counts the number of supporters (i.e., pixels) of each candidate (graylevel)

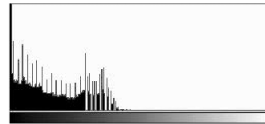
In the simple case of a binary image there are only two candidates: Mr zero and Ms one.

Voting has many useful applications in image processing (as well as in democracy).

50



Image global deformations (1)



Too dark

(a)

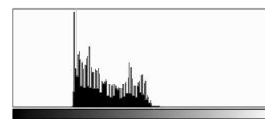


Too bright

(b)

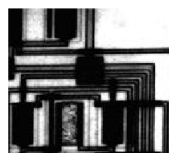


Image global deformations (2)



Low contrast

(c)



High contrast

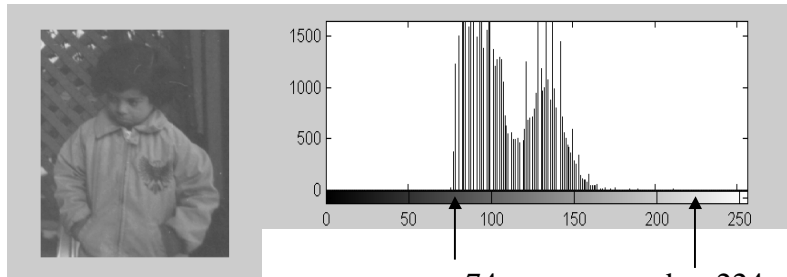
(d)



Histogram stretching

I

Histogram of I



```

I = imread('pout.tif');
a = min(I(:));
b = max(I(:));
J = 255*(I-a)/(b-a);
J = uint8(J);

```

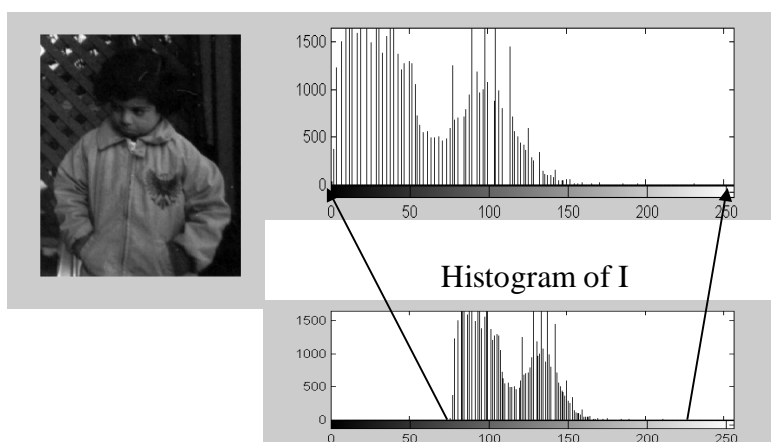
$$J = \frac{I - I_{\min}}{I_{\max} - I_{\min}} \cdot 255$$



Histogram stretching (cont.)

J

Histogram of J

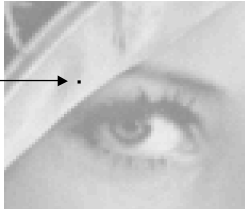




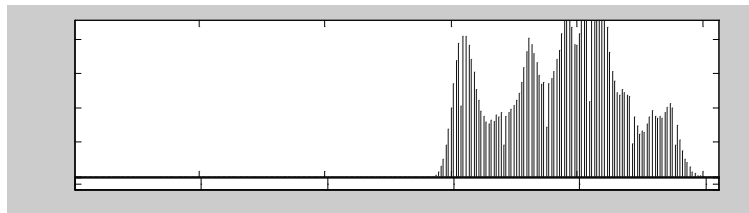
Stretching limitations



A single pixel
Change
Plus rescaling



What happened?

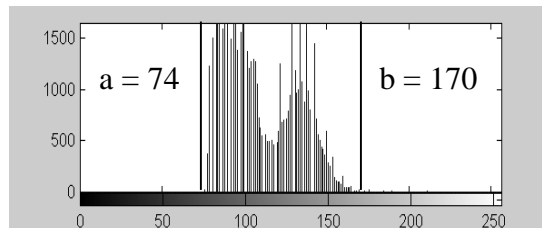


The minimum is 0 and the maximum is 255



Histogram adjustment

Histogram of I $K = \text{imadjust}(I, [0.3 \ 0.67], []);$



$$K = \begin{cases} 255 & \text{if } I_i \geq b \\ 255 \cdot (I_i - a) / (b - a) & \text{if } a \leq I_i < b \\ 0 & \text{if } I_i < a \end{cases}$$

This is a non-linear operation

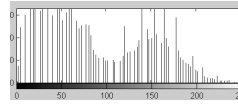
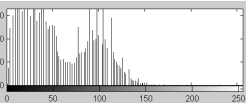
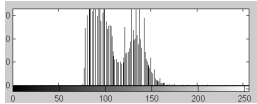
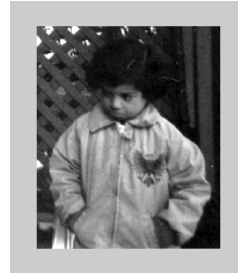
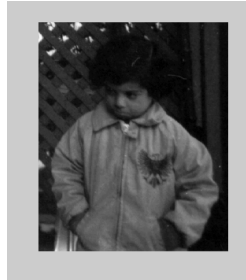


Histogram adjustment (cont.)

I

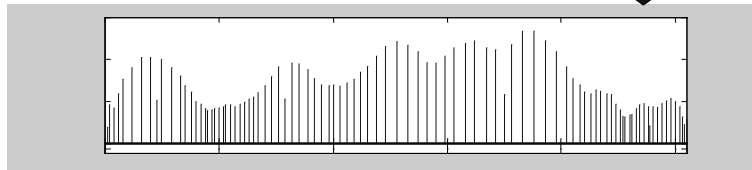
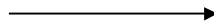
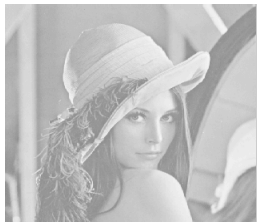
J

K



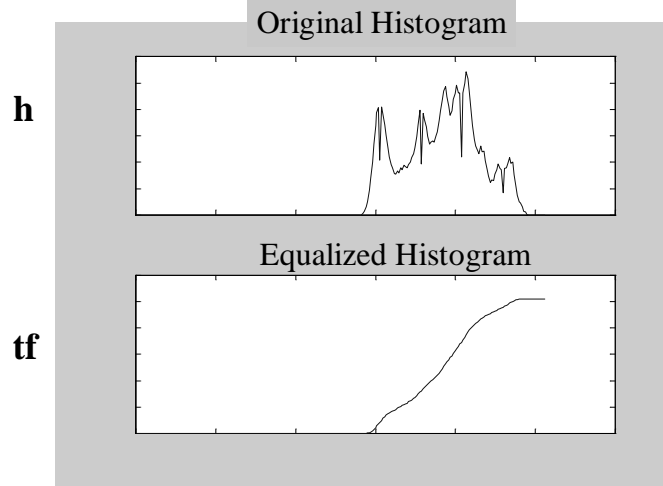
Histogram equalization

$$y = \text{histeq}(x, 256);$$





Transfer Function



```
n = prod(size(x))
tf = cumsum(255*h/n)
```

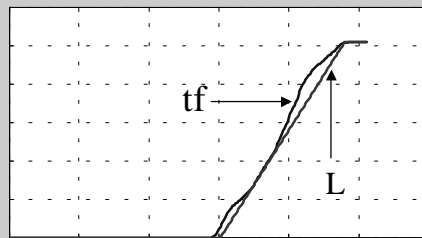


Look Up Table (LUT)



X (in [0 255])

Linear approx. of tf



```
L(1:150) = 0;
L(151:240) = linspace(0,255,90);
L(241:255) = 255;
```

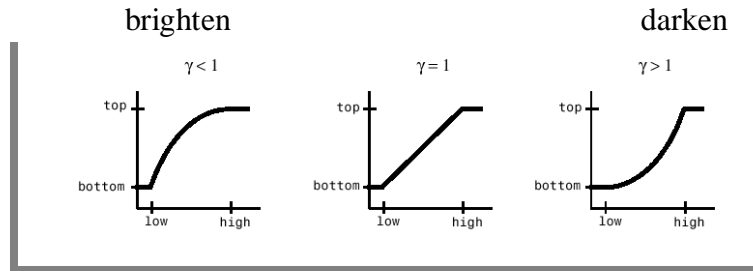
```
y = L(x+1);
```

y





Non-linear LUT (1): Gamma Correction

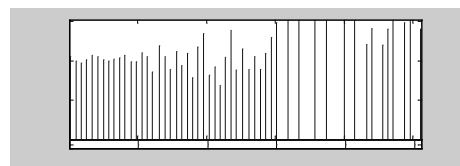
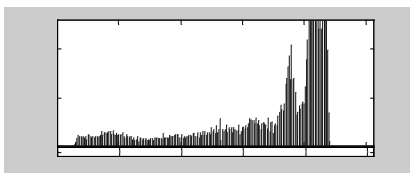
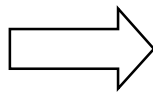
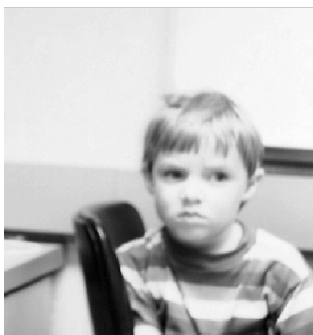


$$I_{out} = 255 \cdot \left(\frac{I - I_{min}}{I_{max} - I_{min}} \right)^\gamma$$



Histogram equalization (2)

Adam



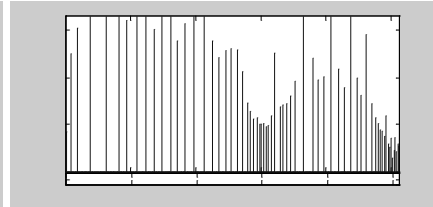
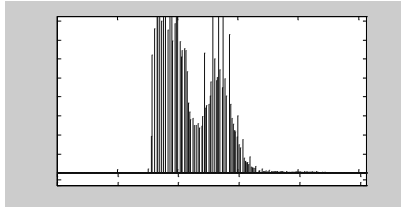
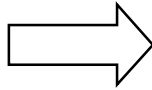


Dr. Yoram Tal



Histogram equalization (3)

Pout



Dr. Yoram Tal



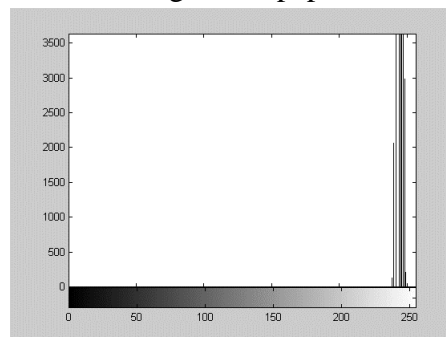
Histogram equalization (4)

Determining the grayscale precision of a scanner

paper



Histogram of paper



64

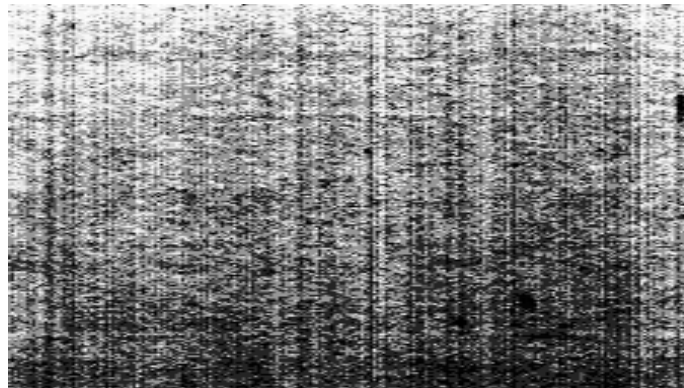


Dr. Yoram Tal

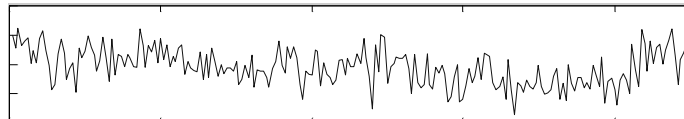


Histogram equalized "paper"

`xeq`



`plot(sum(xeq))`



Dr. Yoram Tal



Non-linear LUT (2): Quantization

Reducing the number of bits/pixel - compression

Lenna



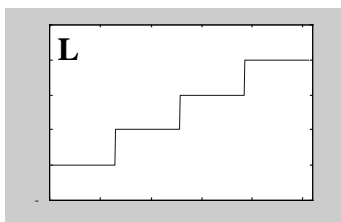
```
1) y = 1 + floor(Lenna/64);
```

```
2) L = repmat(0:3,64,1);
```

```
L = L(:);
```

```
y = L(Lenna);
```

y



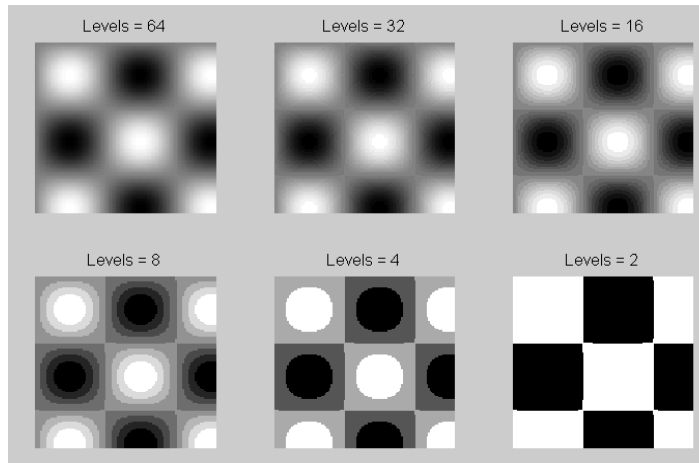
66



Quantization

The contouring effect

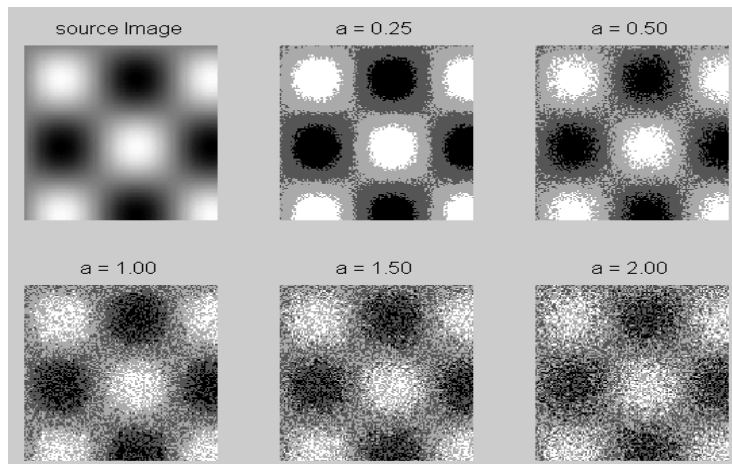
2D sine function



Quantization

Can it be made looking better?

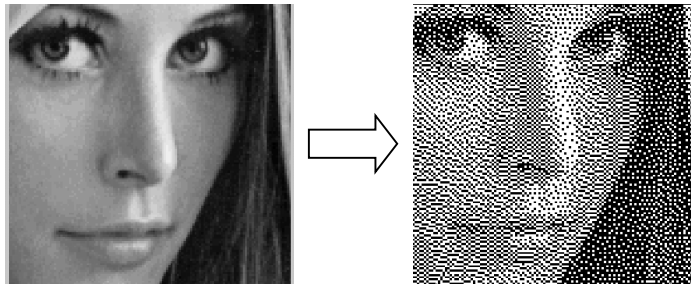
Sure, Add some noise then quantize





Graylevel/color Dithering

Optimal representation of graylevel (or color) images using a small number of graylevels (or colors)



References:

R. W. Floyd and L. Steinberg, "An Adaptive Algorithm for Spatial Gray Scale, International Symposium Digest of Technical Papers, Society for Information Displays, 36. 1975.

Spencer W. Thomas, "Efficient Inverse Color Map Computation", Graphics Gems II, (ed. James Arvo), Academic Press: Boston. 1991. (includes source code)



MATLAB Dithering

rgb2ind converts RGB images to indexed images using one of three different methods: uniform quantization, minimum variance quantization, and colormap mapping. For all of these methods, **rgb2ind** also dithers the image unless you specify 'nodither' for **dither_option**.

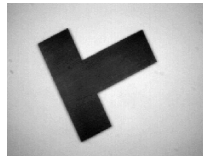
$X = \mathbf{dither}(RGB, map)$ creates an indexed image approximation of the RGB image in the array **RGB** by dithering the colors in colormap **map**. **map** can not have more than 65,536 colors.

$BW = \mathbf{dither}(I)$ converts the intensity image in the matrix **I** to the binary (black and white) image **BW** by dithering.

$[Y, newmap] = \mathbf{imapprox}(X, map, n)$ approximates the colors in the indexed image **X** and associated colormap **map** by using minimum variance quantization. **imapprox** returns indexed image **Y** with colormap **newmap**, which has at most **n** colors. (additional options)

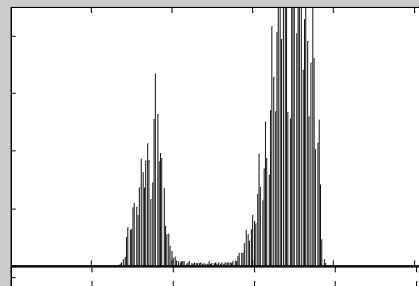
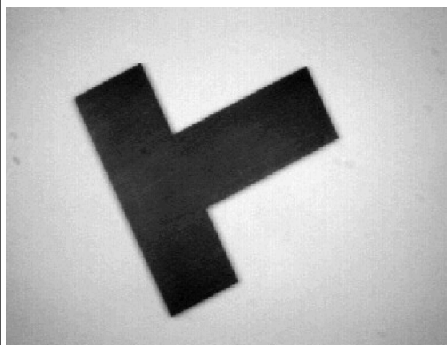


Thresholding



Thresholding

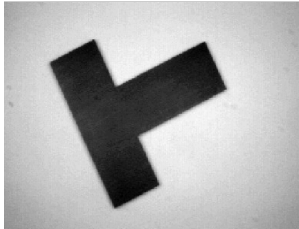
separate objects from background



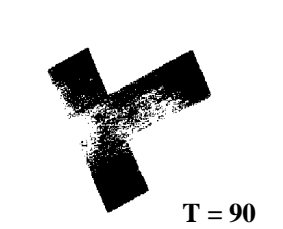
Dr. Yoram Tal

Thresholding (2)

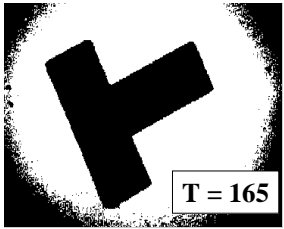
t



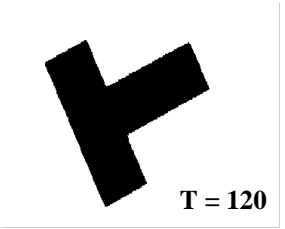
`imagesc(t > T)`



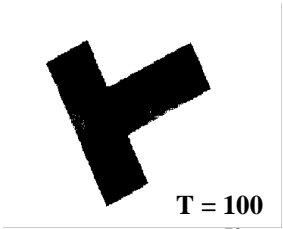
T = 90



T = 165



T = 120



T = 100

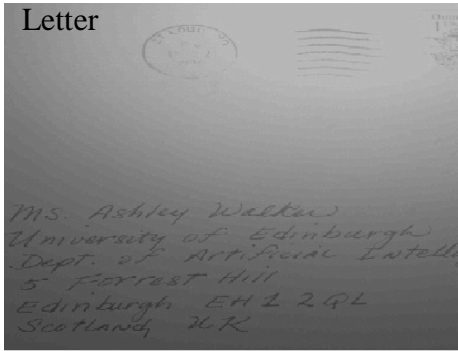
73

Dr. Yoram Tal

Background separation

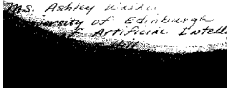
Problem: background is not uniform

Letter

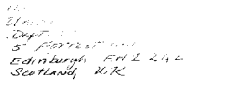
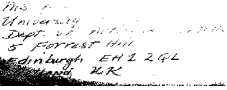
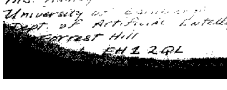


`imagesc(Letter > T)`

T = 100



T = 70 **T = 80** **T = 90**

74



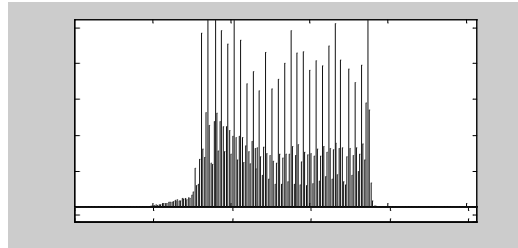
Dr. Yoram Tal



Background separation (2)

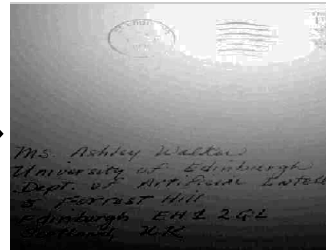
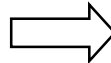
3D map of Letter

Histogram of Letter



`Meshz(Letter)`
`Colormap copper`

Histogram equalization does not
Help here!



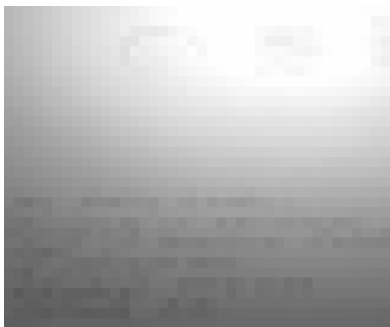
Dr. Yoram Tal



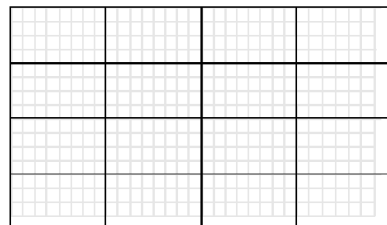
Background approximation

```
bg = blkproc(x,[10 10],'mean2(x)*ones(size(x))');
```

bg



Block processing



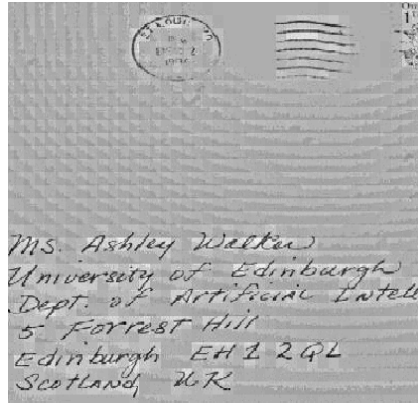
Better approximations may be used (e.g., splines)

76

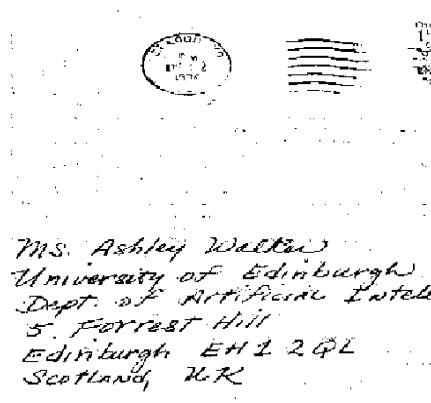


Background subtraction

Letter - bg

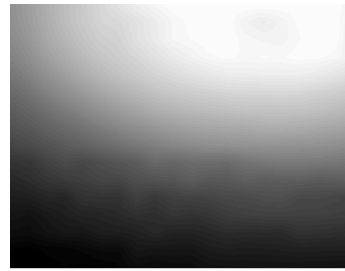
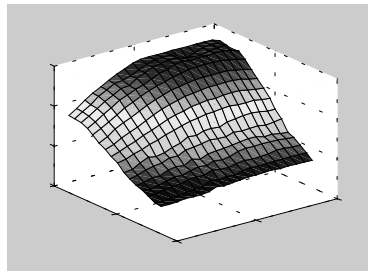


Thresholding



Background approximation (2)

```
bg = blkproc(x,[20 20],'mean2(x)');
```



```
surf(bg); axis ij
```

bg is smaller than Letter

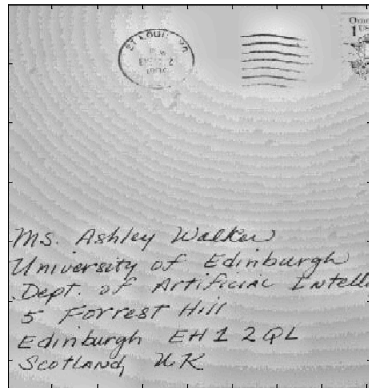
```
bge = imresize(bg,size(Letter), bilinear );
```



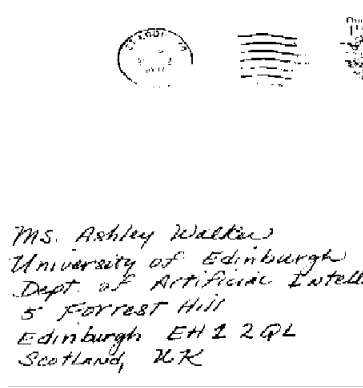


Background subtraction (2)

Letter - bge

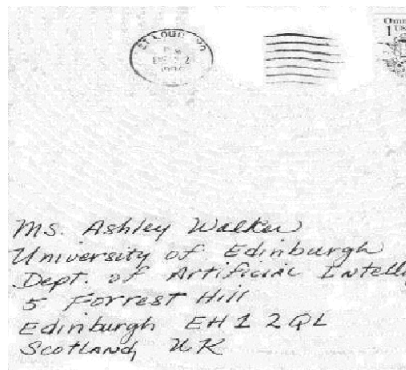


Thresholding

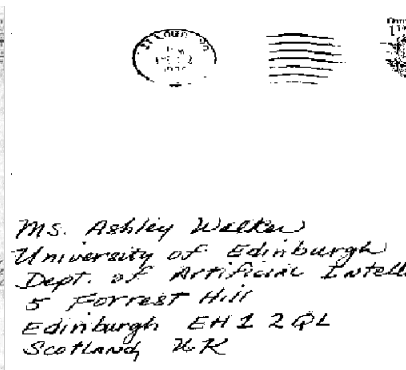


Morphological Filtering (will be discussed later)

Filtered Letter



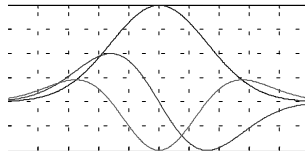
Thresholding



(Top hat filter 7x7)



Linear Filtering



MATLAB's Functions

Conv2	Perform 2-D convolution. (This is a MATLAB function)
convmtx2	Compute 2-D convolution matrix
convn	Perform N-D convolution. (This is a MATLAB function)
filter2	Perform 2-D filtering. (This is a MATLAB function.)
fspecial	Create predefined filters
imfilter	Multidimensional image filtering



Properties

Linear filtering is a transformation of a source (image) matrix into a target matrix such that each element of the target matrix is a linear combination of some elements of the source matrix.

In MATLAB, linear filtering is realized through convolution:

$$Target = kernel \otimes source$$

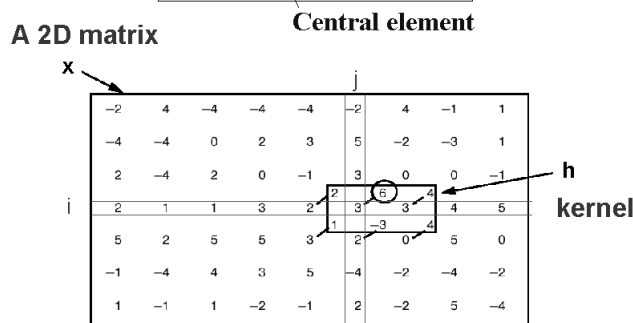
- Convolution is associative:

$$A \otimes (B \otimes C) = (A \otimes B) \otimes C$$



Linear filtering

$$h: \begin{matrix} 2 & 6 & 4 \\ 1 & -3 & 4 \end{matrix} \quad \text{A moving window}$$



$$Result(i,j) = 2*2 + 6*3 + 4*3 + 1*3 \quad 3*2 + 4*0 = 31$$



kernels

Filter2 uses a correlation kernel (the kernel as is)

Conv2 uses a convolution kernel (rotated 180)

$K_{\text{convolution}} = \text{rot90}(K_{\text{correlation}}, 2);$



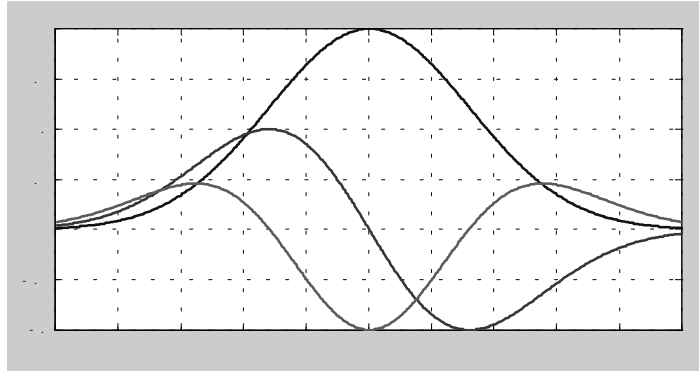
Linear Filtering: fspecial

`h = fspecial(Type, {size}, {sigma})`

- 'gaussian' Gaussian lowpass filter
- 'sobel' Sobel horizontal edge-emphasizing filter
- 'prewitt' Prewitt horizontal edge-emphasizing filter
- 'laplacian' 2D Laplacian operator filter
- 'log' Laplacian of Gaussian filter
- 'average' Averaging filter
- 'unsharp' Unsharp contrast enhancement filter



The Gaussian function



87



The Gaussian Filter

```
h = fspecial( 'gauss',
             freqz2(h)
```

h		
0.0113	0.0838	0.0113
0.0838	0.6193	0.0838
0.0113	0.0838	0.0113

Frequency response of h a lowpass filter

88



Filtering: filter2

```
y = filter2(h,x,{ shape })
```

shape { **full** , **same** , **valid** }
Defines the boundary conditions

Speedup features:

Automatically identifies separability
Realized via a mex (dll) file



Filtering: conv2

```
y = conv2(h,x,{ shape })  
y = conv2(vert,horz,x,{ shape })
```

vert is a vertical column-vector [e.g., ones(21,1)]
horz is a horizontal line-vector [e.g., ones(1,21)]

shape is the same as in filter2.
Zero padding is applied where necessary.



Border padding

When shape = { **full** , **same** }
a border padding is required

Filter2 and **conv2** use zero padding

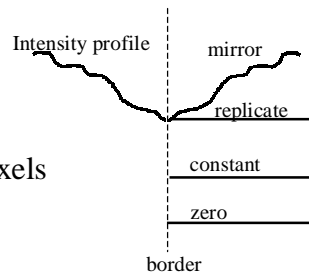
Other choices are:

Constant, non-zero value

Replications of the border pixels

Mirror image of the near border pixels

Circular (periodic) padding



Filtering: imfilter

`y = imfilter(X, H, opt1, opt2,)`

X and **H** are multidimensional

The class of **y** is the same as the class of **x**

Boundary options: zero, const. value, replicate, and symmetric

Output size: same and full

Kernel: correlation or convolution



Filtering - Separability

$$h = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Is separable into

$$h_1 = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

Followed by

$$h_2 = [1 \ 1 \ 1]$$

Therefore: `y = filter2(h,x)`

Is the same as: `tmp = filter2(h1,x)`
`y = filter2(h2,tmp)`



Filtering - Separability (cont.)

The same is true for any function which can be written as a product:

h is separable if
`h(x,y) = h(x)*h(y)`

For example, the Gaussian filter is separable since:

`exp(-x^2 - y^2) =`
`exp(-x^2)*exp(-y^2)`



Filtering - Separability (cont.)

Separability saves time!

A separable 20x20 filter requires 40 operations (multiplication-addition) per pixel

The same, non-separable filter requires 400 operations per pixel 10 times as much



Nonlinear Diffusion filtering



The Physical Model

basic equations

Fick's law: The concentration (or heat) gradient ∇u causes a flux j which aims to compensate it.

$$j = -D \cdot \nabla u \quad j \parallel \nabla u \begin{cases} \text{yes} & \text{isotropic} \\ \text{no} & \text{anisotropic} \end{cases}$$

Continuity equation: Diffusion does not create/destroys mass (heat)

$$\partial_t u = -\text{div}(j)$$



The Diffusion equation

Combining Fick's law with the continuity equation yields the **Diffusion equation:**

$$\partial_t u = \text{div}(D \cdot \nabla u)$$

with the initial condition:

$$u(\vec{x}, 0) = f(\vec{x})$$

D is the diffusion tensor (a positive definite symmetric matrix)



Gaussian convolution

The solution of this equation ($D = 1$) is given by the *Convolution Integral*:

$$u(\vec{x}, t) = \begin{cases} f(\vec{x}) & (t = 0) \\ (G_{\sqrt{2t}} \otimes f)(\vec{x}) & (t > 0) \end{cases}$$

where

$$G_{\sigma}(\vec{x}) = \frac{1}{2\pi\sigma^2} \cdot \exp\left(-\frac{|\vec{x}|^2}{2\sigma^2}\right) \quad f: \mathbf{R}^2 \rightarrow \mathbf{R}$$



Johann Carl Friedrich Gauss



Born: 30 April 1777 in Brunswick, Duchy of Brunswick (now Germany)
Died: 23 Feb 1855 in Göttingen, Hanover (now Germany)