UNIVERSIDADE FEDERAL DO PARANÁ – UFPR

Departamento de Informática

Computer Vision and Perception

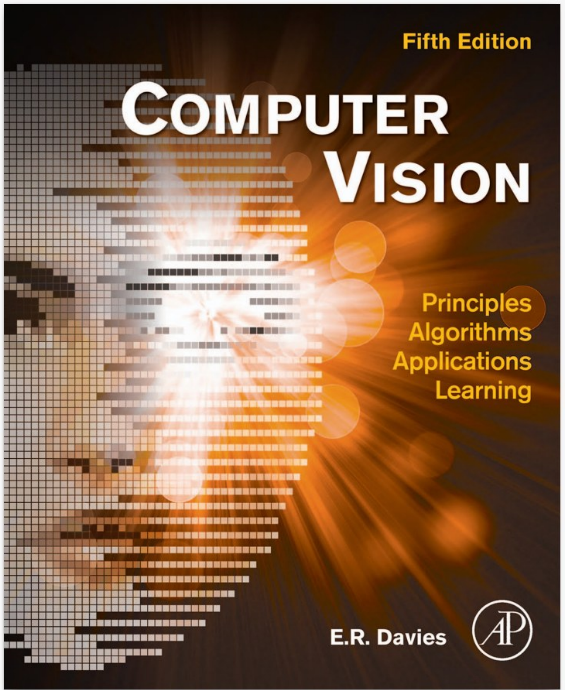
Introduction

Prof. Eduardo Todt
2023

Sumary

- Text book
- Introduction
  - recognition problem
  - image basics

Textbook



Computer Vision

Principles, Algorithms, Applications, Learning

5th Edition - November 14, 2017

Author: E. R. Davies

Hardback ISBN: 9780128092842

9 7 8 - 0 - 1 2 - 8 0 9 2 8 4 - 2

eBook ISBN: 9780128095751

Journal


ScienceDirect
Journals & Books
Computer Vision and Image Understanding
Computer Vision and Image Understanding
Supports open access

The process of recognition

5x5 characters

NxN image

window sliding: 5^2 x N^2 ops

orientation: x 360 ?

2^{N^2} possible patterns

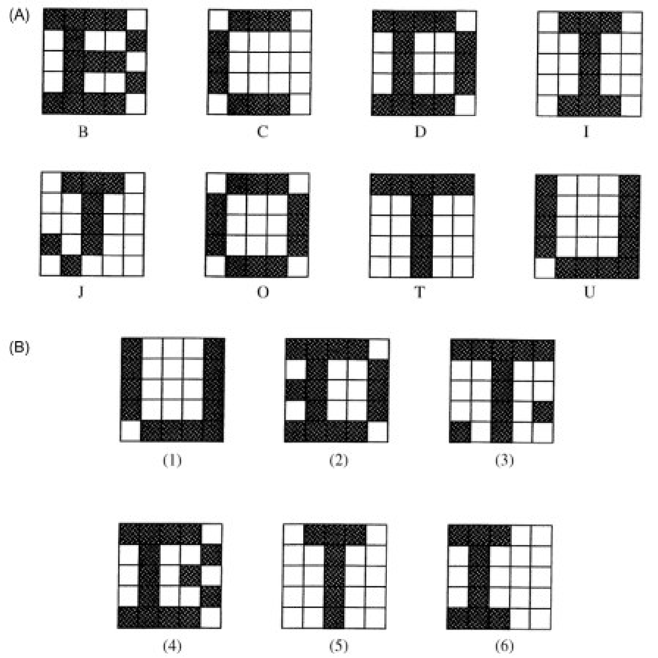


FIGURE 1.1

Some simple 25-bit patterns and their recognition classes used to illustrate some of the basic problems of recognition: (A) training set patterns (for which the known classes are indicated); (B) test patterns.

Trying to generalize

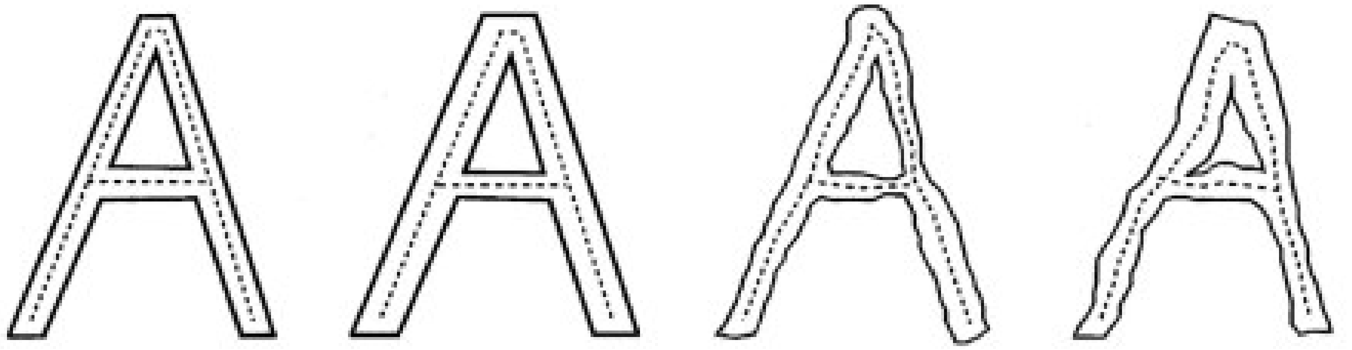


FIGURE 1.2

Use of thinning to regularize character shapes. Here character shapes of different limb widths—or even varying limb widths—are reduced to stick figures or skeletons. Thus irrelevant information is removed and at the same time recognition is facilitated.

Basic pipeline

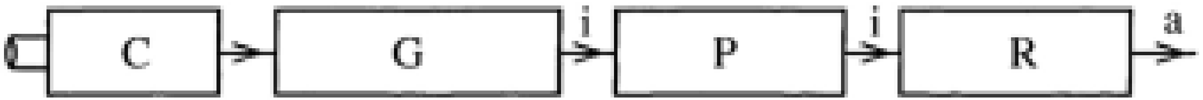


FIGURE 1.3

The two-stage recognition paradigm: C, input from camera; G, grab image (digitize and store); P, preprocess; R, recognize (i, image data; a, abstract data). The classical paradigm for object recognition is that of (1) preprocessing (image processing) to suppress noise or other artefacts and to regularize the image data and (2) applying a process of abstract (often statistical) pattern recognition to extract the very few bits required to classify the object.

Object location

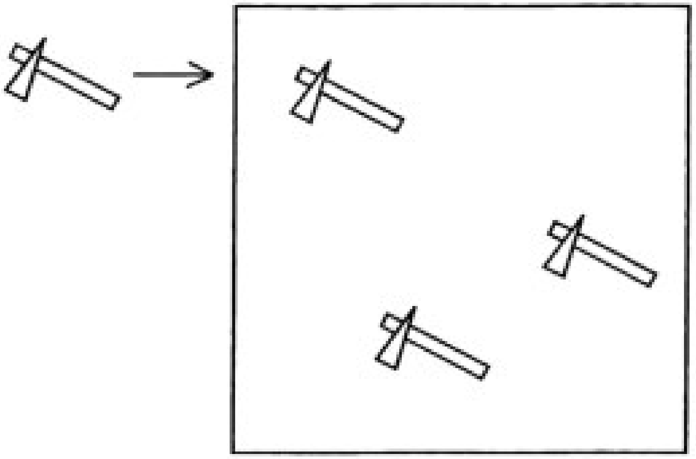


FIGURE 1.4

Template matching, the process of moving a suitable template over an image to determine the precise positions at which a match occurs, hence revealing the presence of objects of a particular type.

Generalized Hamming distance:

$$\mathcal{D} = \sum_i |I_i - I_t|$$

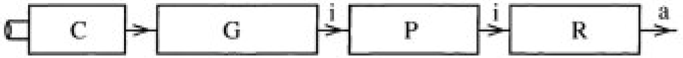
For a 30x30 template and 256x256 image
~60×10^6 ops

Without considering orientation and scale ...

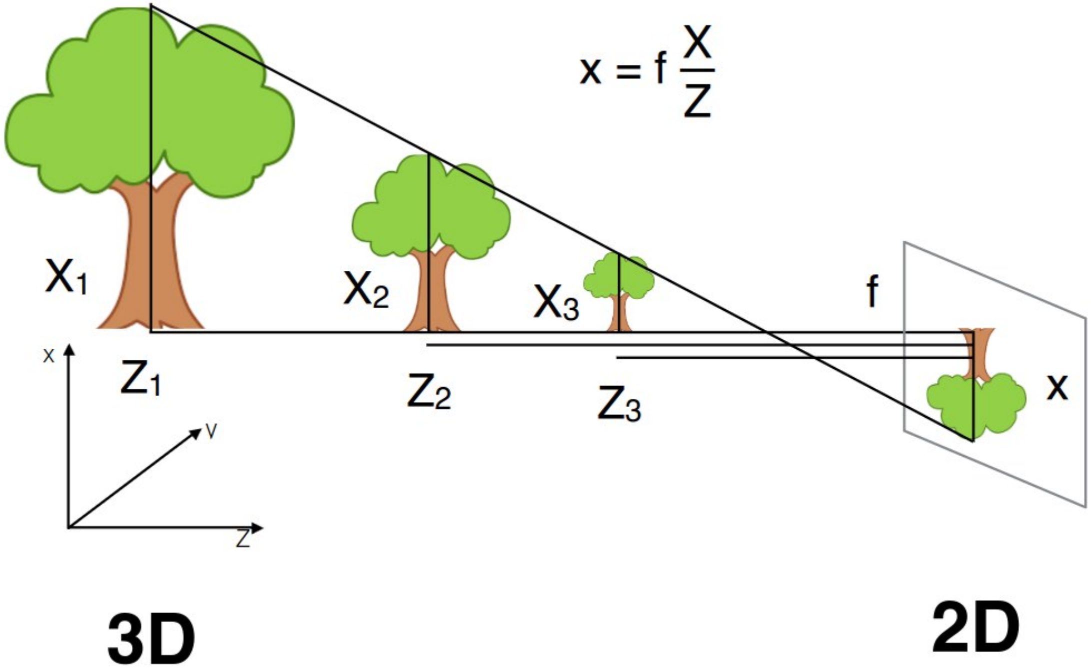
Information reduction

Two-stage template matching

- instead of template uses features
  - e.g. lines, corners,
- the problem of feature size



Vision as inverse graphics



CG: N to 1
CV: 1 to N

N can be infinite ...

https://www.cs.cmu.edu/~katef/papers/NeuralSLAMJuly2018.pdf

Images and operations

Binary images

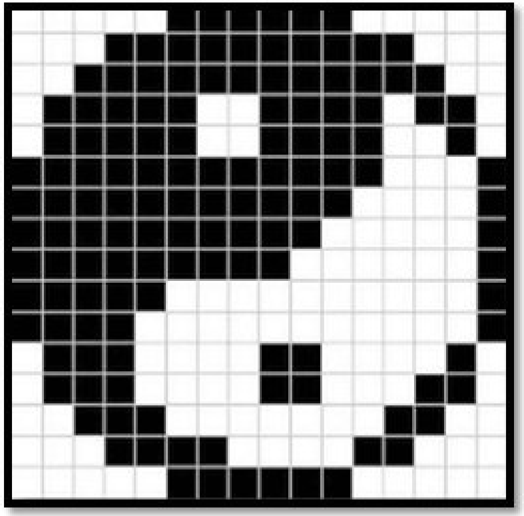


Fig. Binary image

https://www.javatpoint.com/dip-types-of-images

Images and operations

Gray-scale images

Fig. Gray-scale image

Typical: 8 bits/pixel

https://www.javatpoint.com/dip-types-of-images

Pixel intensity value

$$\underbrace{f(1,1)}_{\text{Pixel location}} = 103$$

$$f(\underbrace{645:650}_{\text{rows}},\underbrace{1323:1328}_{\text{columns}}) =$$

$$\begin{matrix} 83 & 82 & 82 & 82 & 82 & 82 \\ 82 & 82 & 82 & 81 & 81 & 81 \\ 82 & 82 & 81 & 81 & 80 & 80 \\ 82 & 82 & 81 & 80 & 80 & 79 \\ 80 & 79 & 78 & 77 & 77 & 77 \\ 80 & 79 & 78 & 78 & 77 & 77 \end{matrix}$$

$f(2724,2336) = 88$

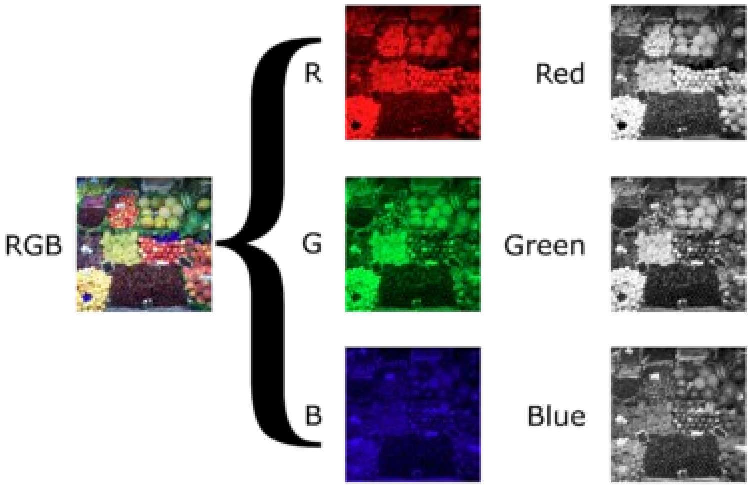
Consider the following image (2724x2336 pixels) to be 2D function or a matrix with rows and columns

In 8-bit representation
Pixel intensity values change between 0 (Black) and 255 (White)

https://sisu.ut.ee/imageprocessing/book/1

Images and operations

color images



https://en.wikipedia.org/wiki/Grayscale

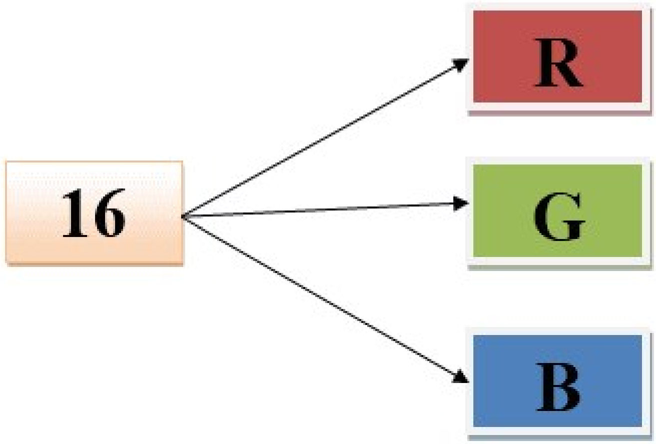
Images and operations

8-bit color images

https://www.javatpoint.com/dip-types-of-images

Images and operations

16-bit color images



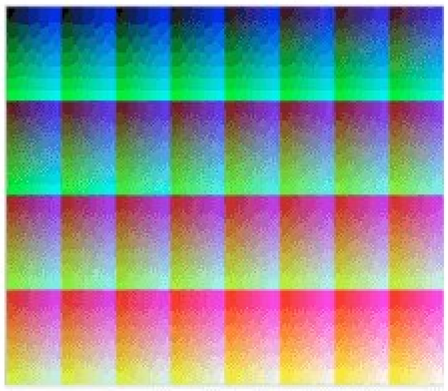


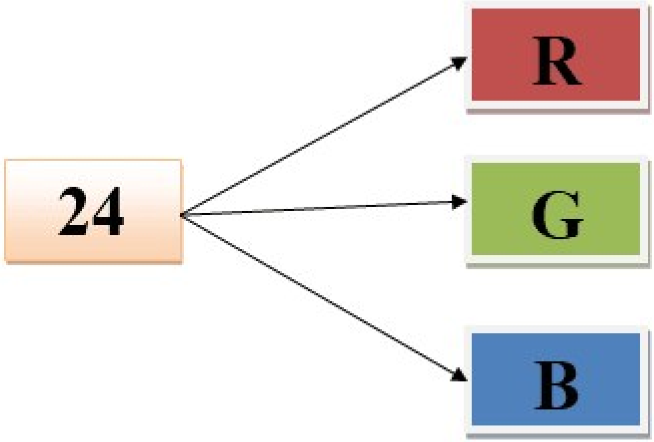
Fig: RGB 16 bits palette

R 5 bits
G 6 bits
B 5 bits

https://www.javatpoint.com/dip-types-of-images

Images and operations

24-bit color images



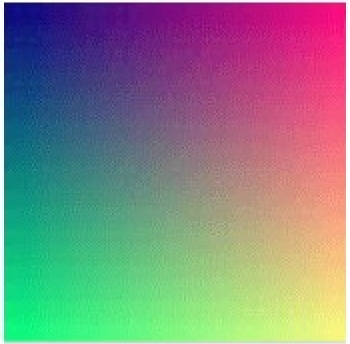


Fig: 16,777,216 colors

R 8 bits
G 8 bits
B 8 bits

https://www.javatpoint.com/dip-types-of-images

OpenCV image container

Early times: *IplImage*

https://docs.opencv.org/3.4/d6/d5b/s

Open Source Computer Vision

Main Page | Related Pages | Modules | Namespaces | Classes

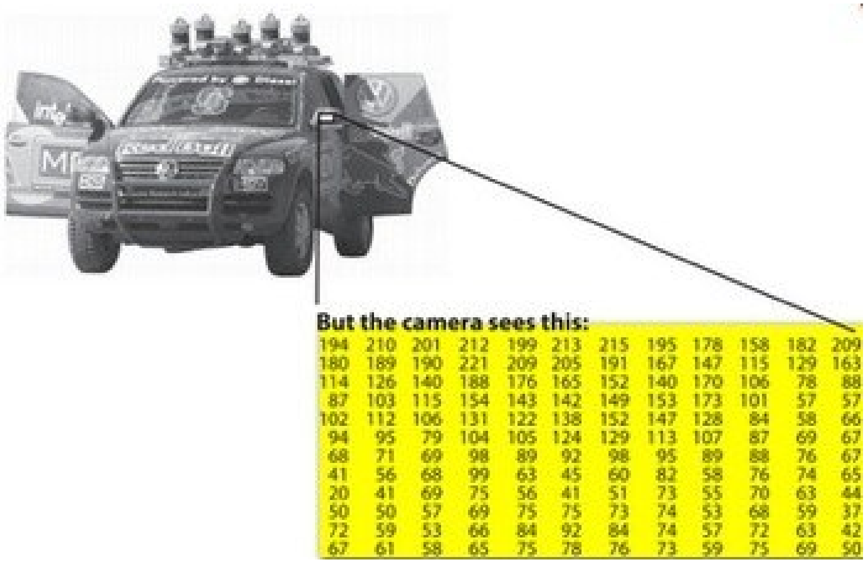
IplImage Struct Reference

Core functionality » C structures and operations

```
#include <opencv2/core/types_c.h>
```

Public Attributes

int	align
int	alphaChannel
int	BorderConst [4]
int	BorderMode [4]
char	channelSeq [4]
char	colorModel [4]
int	dataOrder
int	depth
int	height
int	ID
char *	imageData
char *	imageDataOrigin
void *	imageId
int	imageSize
struct _IplImage *	maskROI
int	nChannels
int	nSize
int	origin
struct _IplROI *	roi
struct _IplTileInfo *	tileInfo
int	width
int	widthStep



194	210	201	212	199	213	215	195	178	158	182	209
180	189	190	221	209	205	191	167	147	115	129	163
114	126	140	188	176	165	152	140	170	106	78	88
87	103	115	154	143	142	149	153	173	101	57	57
102	112	106	131	122	138	152	147	128	84	58	66
94	95	79	104	105	124	129	113	107	87	69	67
68	71	69	98	89	92	98	95	89	88	76	67
41	56	68	99	63	45	60	82	58	76	74	65
20	41	69	75	56	41	51	73	55	70	63	44
50	50	57	69	75	75	73	74	53	68	59	37
72	59	53	66	84	92	84	74	57	72	63	42
67	61	58	65	75	78	76	73	59	75	69	50

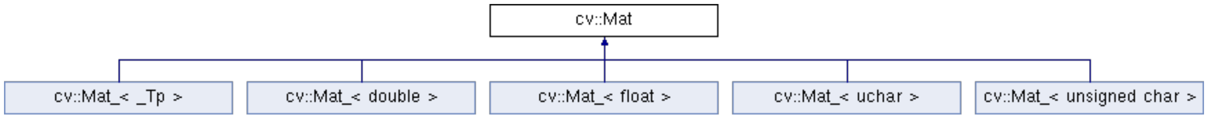
https://docs.opencv.org/4.x/d6/d6d/tutorial_mat_the_basic_image_container.html

OpenCV image container

Currently: *Mat*

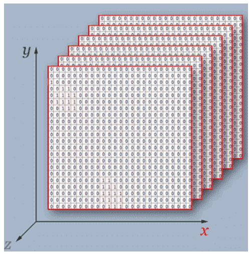
n-dimensional dense array

```
//including all the necessary headers
#include "opencv2/core.hpp"
#include <iostream>
#include <opencv2/opencv.hpp>
//defining the namespace std and cv
using namespace std;
using namespace cv;
void main()
{
//creating a matrix using mat function and
displaying the matrix as the output on the screen
Mat Mvalue(4, 4, CV_8UC3, Scalar(1, 0, 1));
cout<<"The resulting matrix is:\n";
cout << "Mvalue = " << endl << " " << Mvalue <<
endl << endl;
}
```




```
The resulting matrix is:

Mvalue =
[  1,   0,   1,   1,   0,   1,   1,   0,   1,   1,   0,   1;
   1,   0,   1,   1,   0,   1,   1,   0,   1,   1,   0,   1;
   1,   0,   1,   1,   0,   1,   1,   0,   1,   1,   0,   1;
   1,   0,   1,   1,   0,   1,   1,   0,   1,   1,   0,   1]
```



https://docs.opencv.org/4.x/d6/d6d/tutorial_mat_the_basic_image_container.html

https://docs.opencv.org/4.x/d3/d63/classcv_1_1Mat.html#details

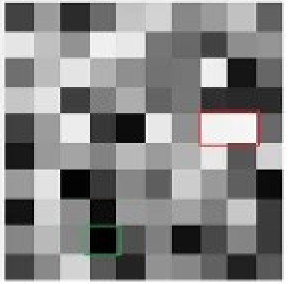
https://docs.opencv.org/4.x/d6/d6d/tutorial_mat_the_basic_image_container.html

Image processing basics

quantization, up sample, down sample

grayscale range

intensity



```
[[ 66 156  43 106 195 209 136 155 193  97]
 [229 192 146 239 231 118 103  66 142 149]
 [112 187 229 177 143 114 117 237  22 103]
 [197 226  62 125 173  99 118  47  41  43]
 [ 64 156 236  54  12 231 144 242 246 105]
 [ 25 152 164 130 184 155 178 230  87 210]
 [152 226   5  56 135  96 205 154  96  11]
 [ 19 209 138  12 153 147 162 124 198  59]
 [134 184 133   0  80 128  18  72 131  58]
 [ 65 137 212  87  35 146 135  97  33  89]]
```

http://www.linkedin.com/in/ya%C4%9Fmur-cigdem-aktas

Gray level transformations

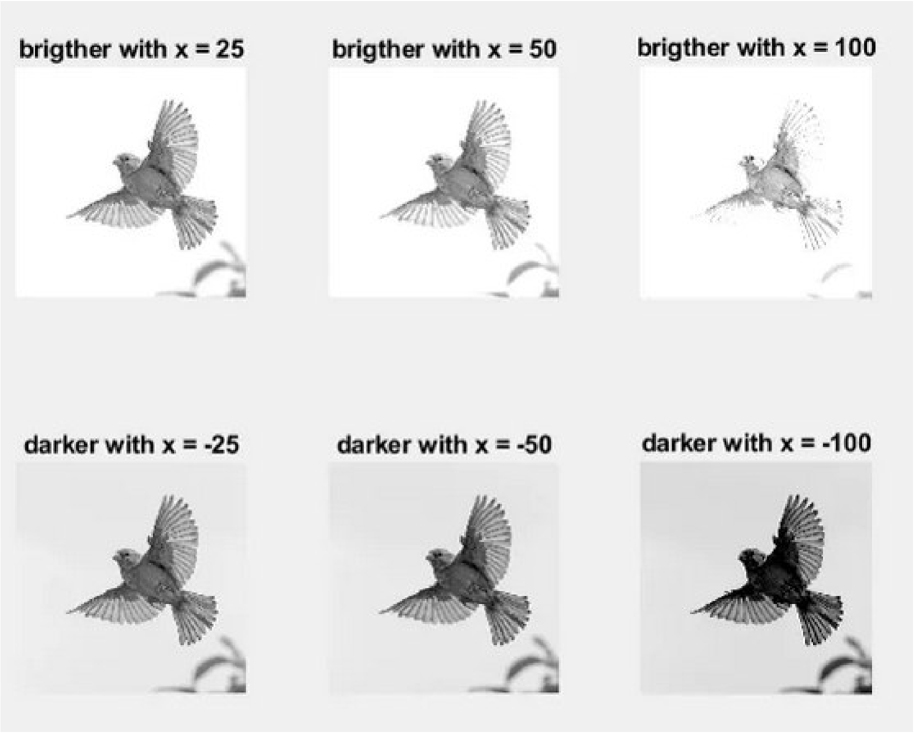
Applied to whole image

Linear transformations

s = cr + x

ex. c=1; x=25, 50, …

r initial pixel value
s transformed pixel value
x offset
c coefficient

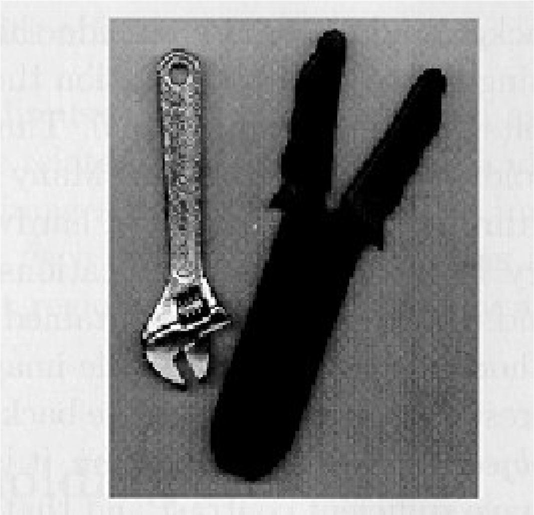


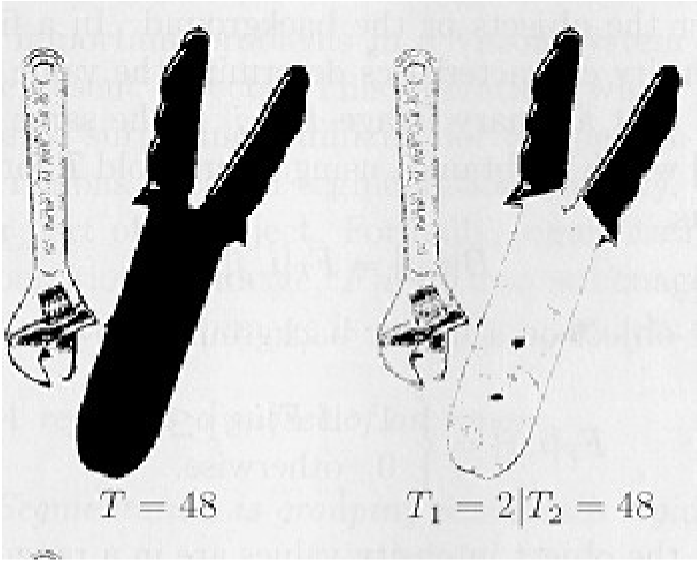
http://www.linkedin.com/in/ya%C4%9Fmur-cigdem-aktas

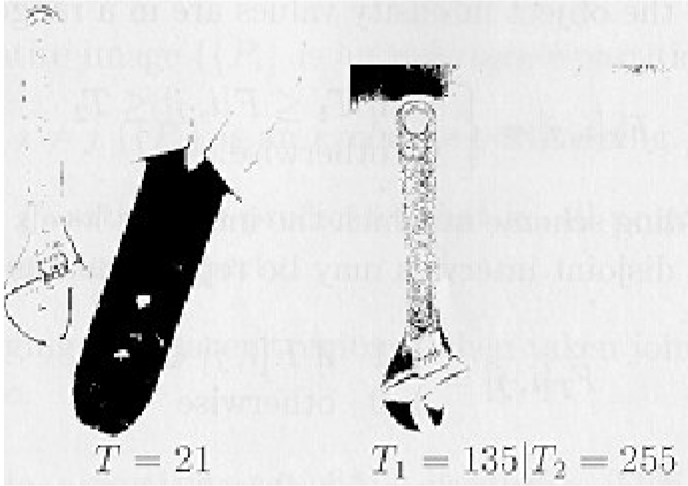
Thresholding

The simplest approach to segment an image is thresholding

If f (x, y) > T then f (x, y) = 0 else f (x, y) = 255







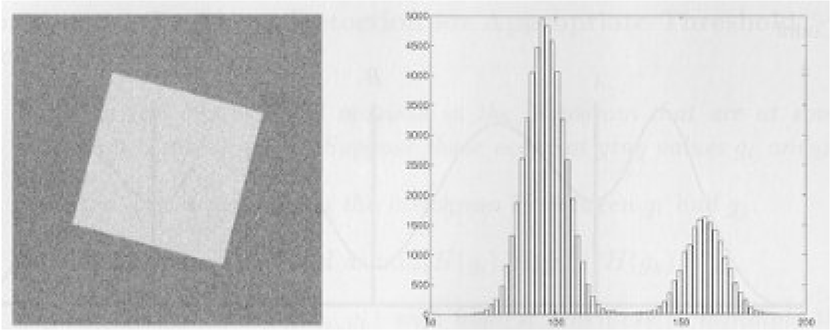
https://www.cse.unr.edu/~bebis

Thresholding

https://www.cs.cornell.edu/courses/cs5670/2017sp/lectures/lec01_filter.pdf

Automatic Thresholding

Regions with uniform intensity give rise to strong peaks in the histogram



https://www.cse.unr.edu/~bebis

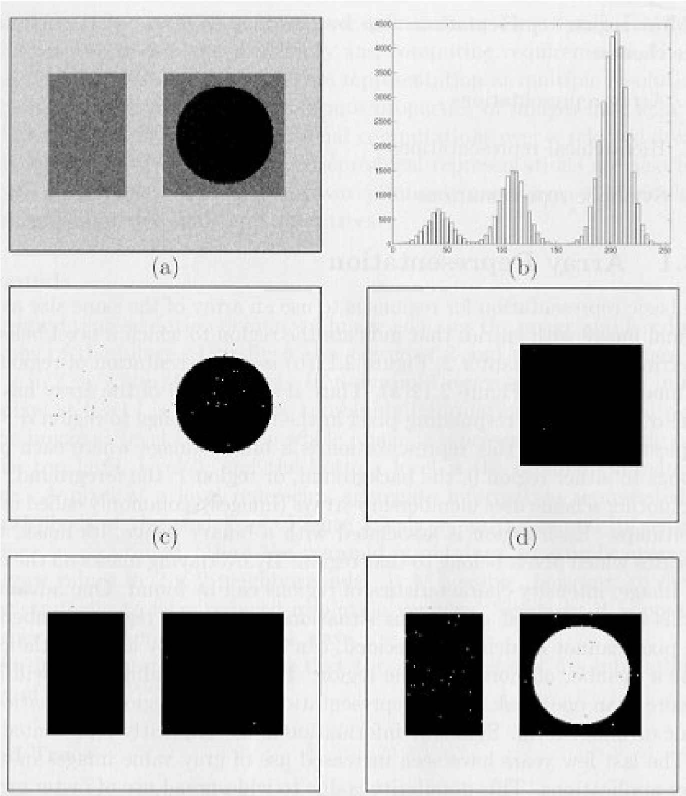
Automatic Thresholding

Multilevel thresholding is also possible

If f (x, y) < T1 then f (x, y) = 255

else if T1 ≤ f (x, y) < T2 then f (x, y) = 128

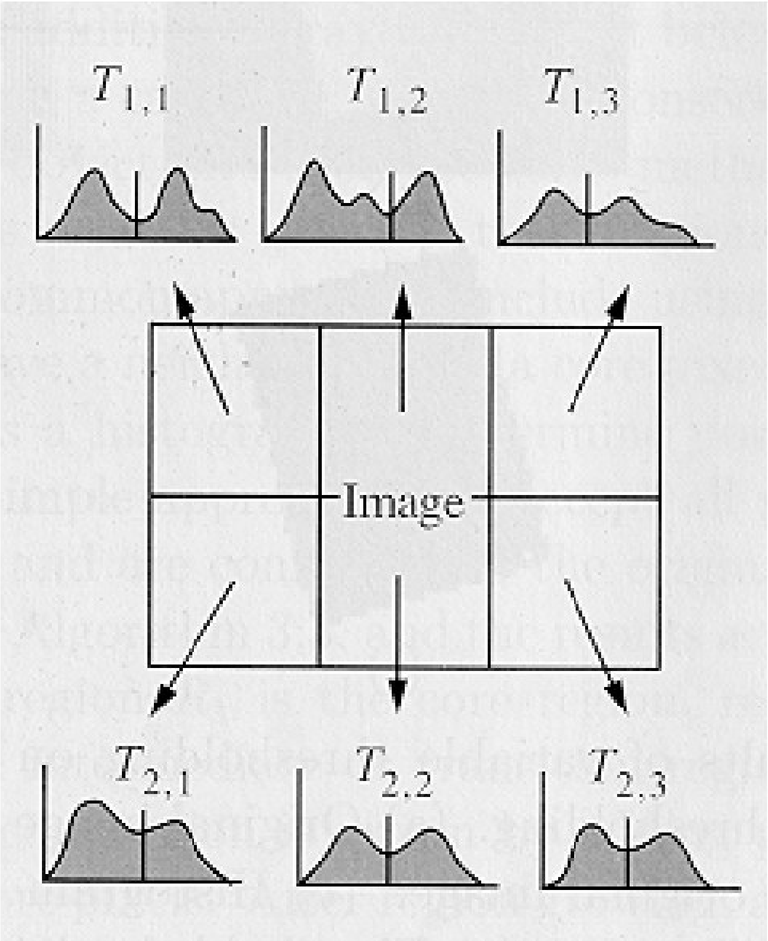
else f (x, y) = 0



https://www.cse.unr.edu/~bebis

Local Thresholding

A single threshold will not work well when we have uneven illumination due to shadows or due to the direction of illumination



https://www.cse.unr.edu/~bebis

Gray level transformations

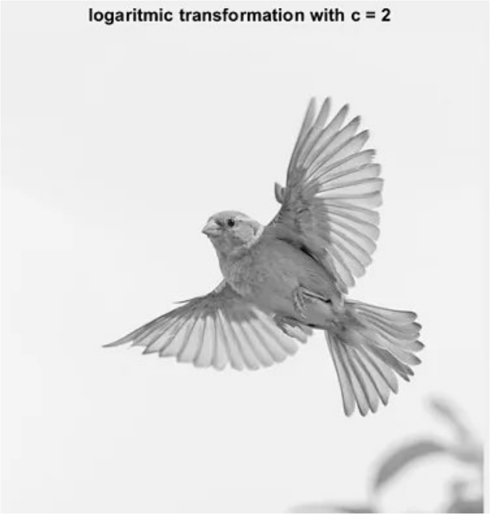
Applied to whole image

Log transformations

s = c log(1 + r)

ex. c=1; x=25, 50, …

r initial pixel value
s transformed pixel value
c coefficient



http://www.linkedin.com/in/ya%C4%9Fmur-cigdem-aktas

Gray level transformations

Applied to whole image

Power law transformations

Gamma correction

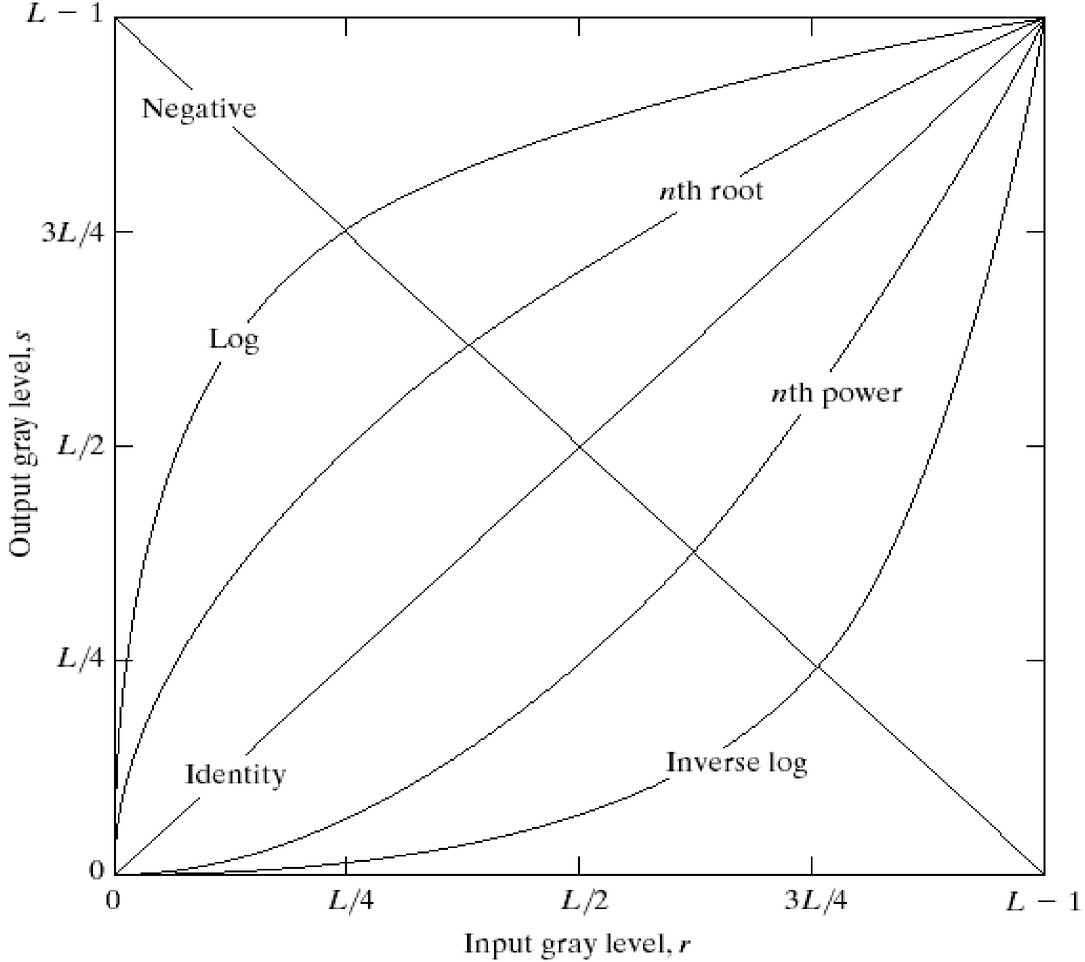
$s = c\, r^{\gamma}$

http://www.linkedin.com/in/ya%C4%9Fmur-cigdem-aktas

Gray level transformations

Applied to whole image

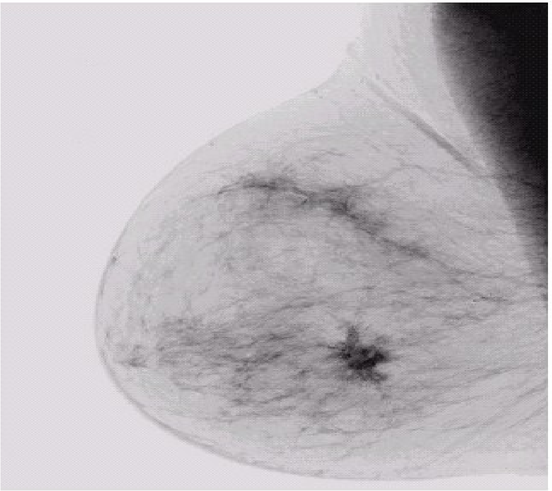
Generalized



https://sisu.ut.ee/imageprocessing/book/5

Image Negatives

s= (L-1)-r



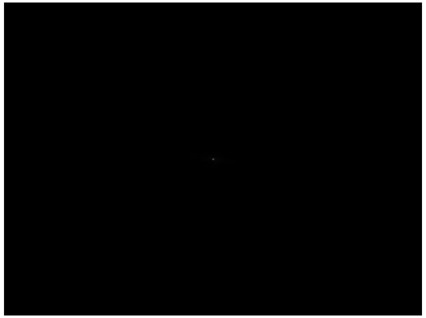
(left) Original digital mammogram. (right) Negative image obtained using the negative transformation

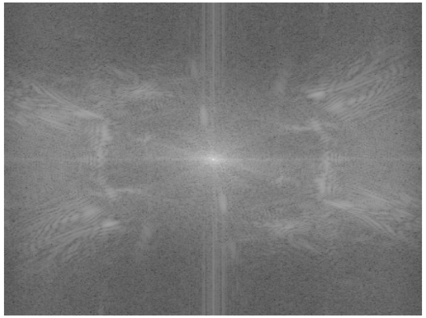
L = 2^k   k=número de bits

© 2002 R. C. Gonzalez & R. E. Woods

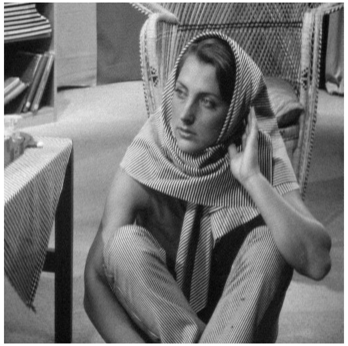
Logarithmic Transformations

$$s= c \log(1+r)$$

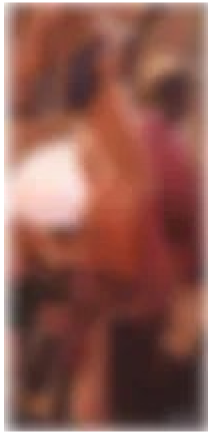




(left) Fourier spectrum of Barbara’s image. (right) Result of applying the log transformation



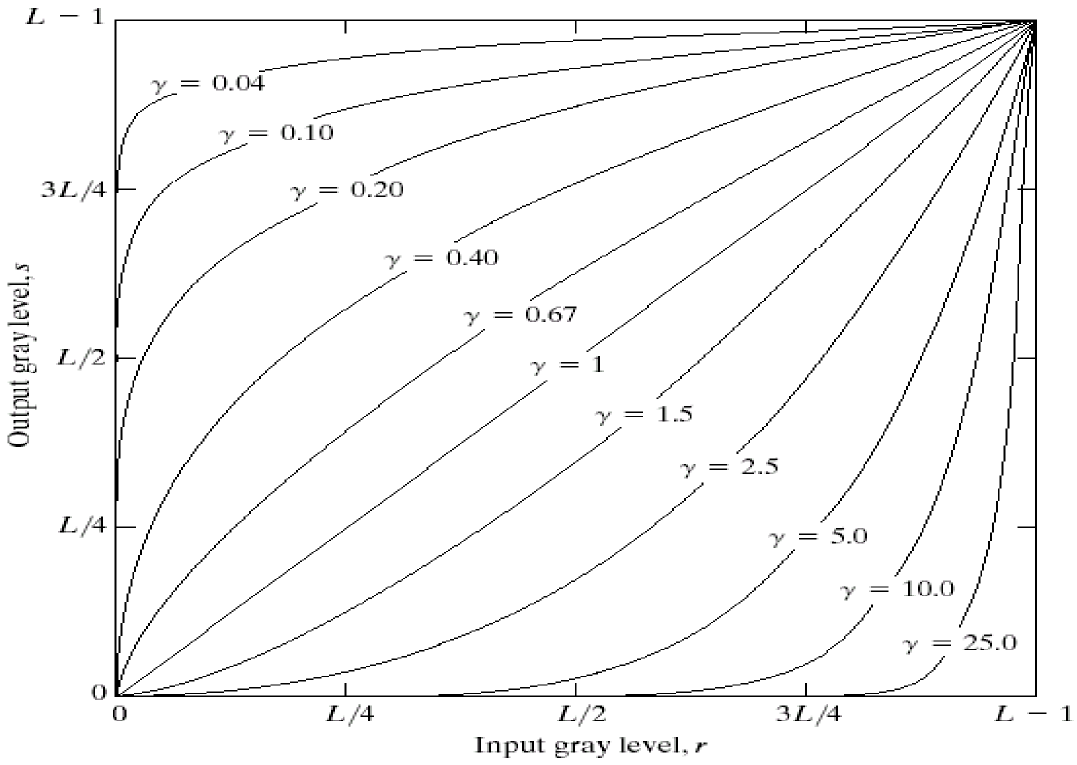
You should know Lena



Capa playboy 1973

https://en.wikipedia.org/wiki/Lenna

Logarithmic Transformations



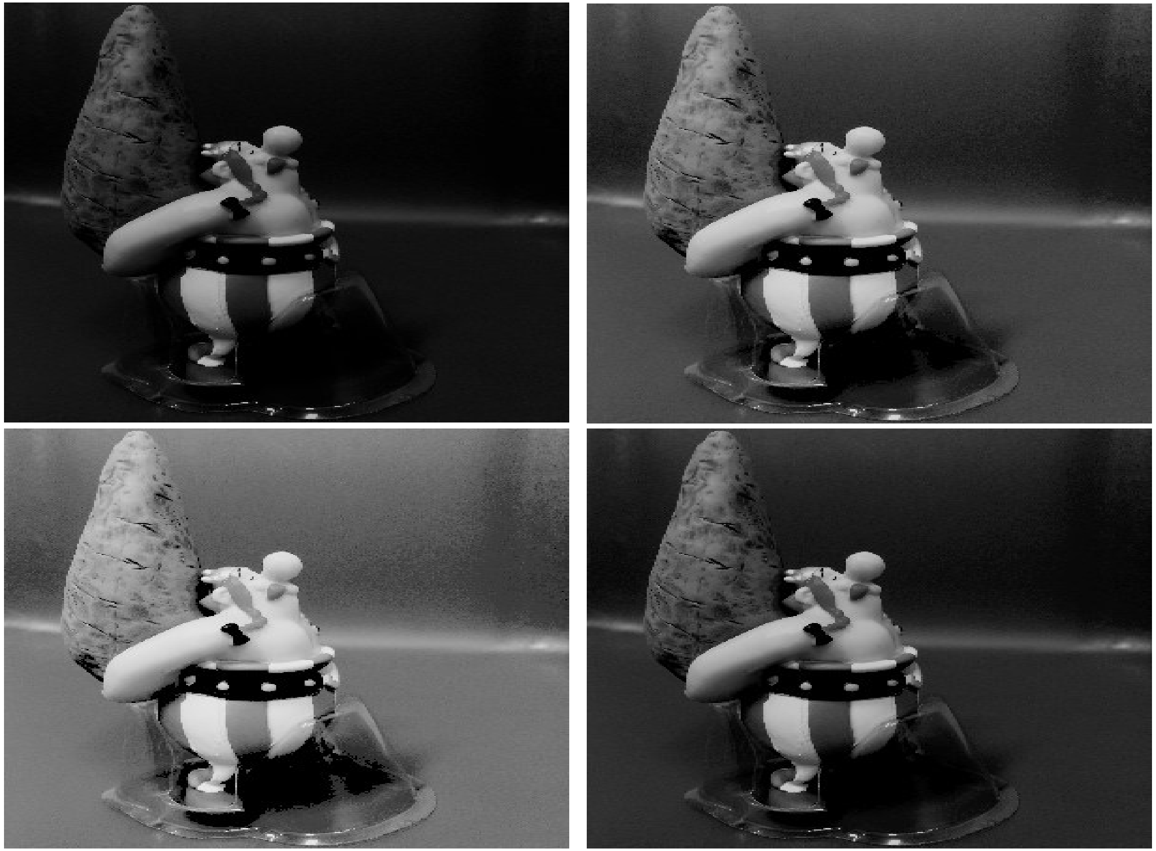
$s = c\, r^{\gamma}$

s is the pixel value of the output image and r is the pixel value of the input image. ($\gamma \geq 0$ and $0 \leq r \leq 1$)

Plots for various values of γ (c=1)

© 2002 R. C. Gonzalez & R. E. Woods

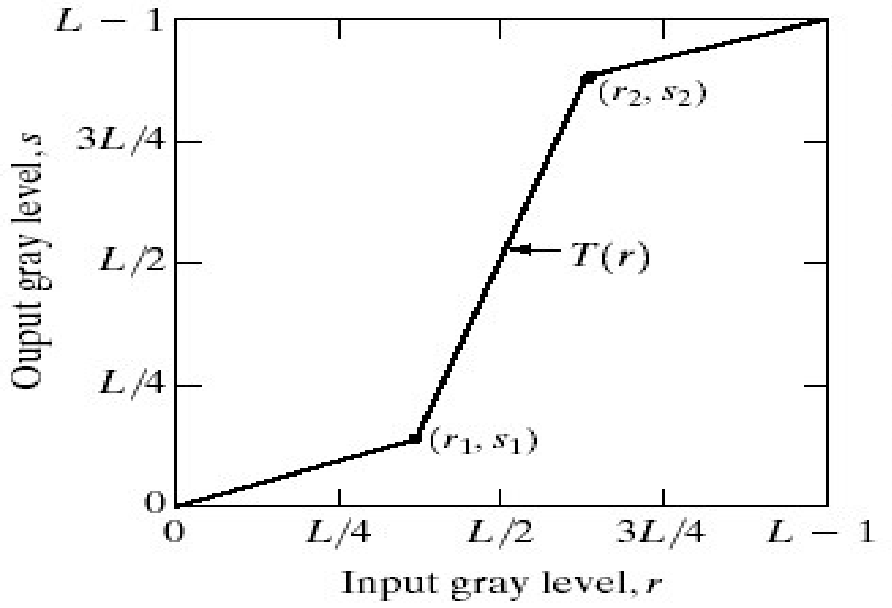
Logarithmic Transformations



a	b
c	d

(a) original image
(b) $\gamma = 0.5$
(c) $\gamma = 0.3$
(d) $\gamma = 0.7$

Piecewise-Linear Transformations

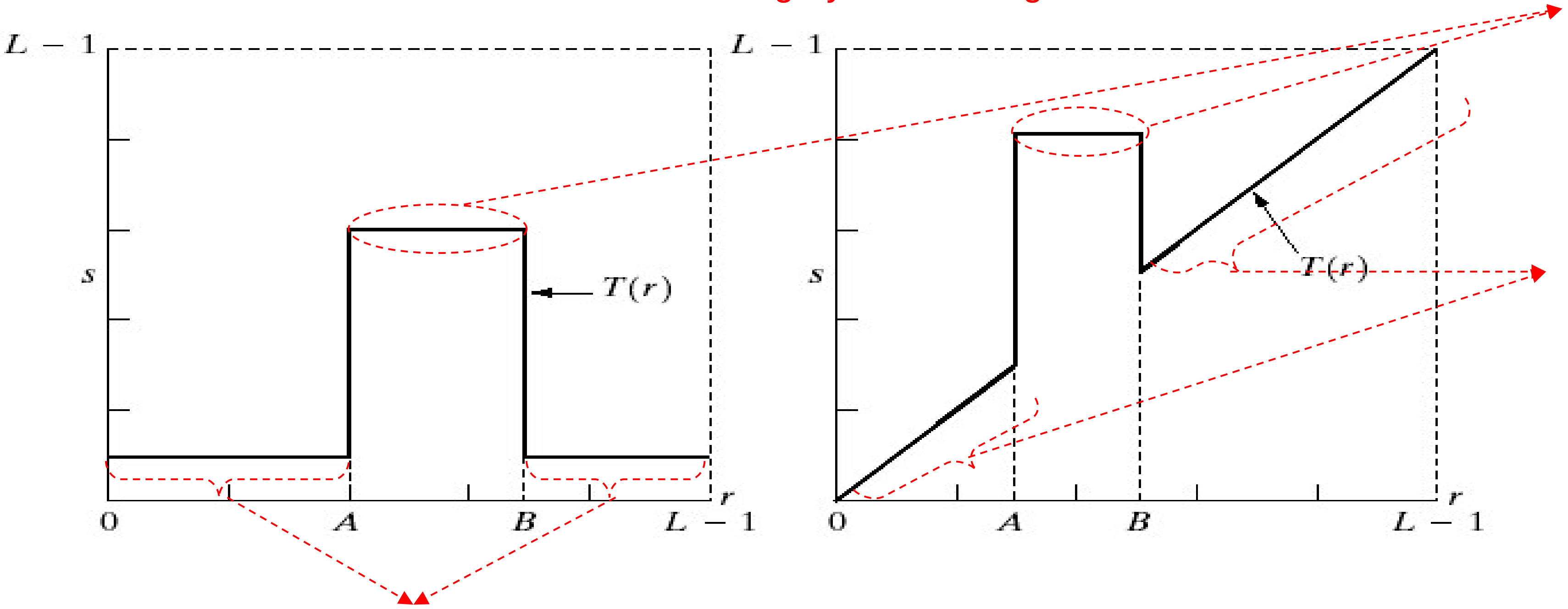


An example of piecewise linear transformation function

© 2002 R. C. Gonzalez & R. E. Woods

Piecewise-Linear Transformations

grey-level slicing



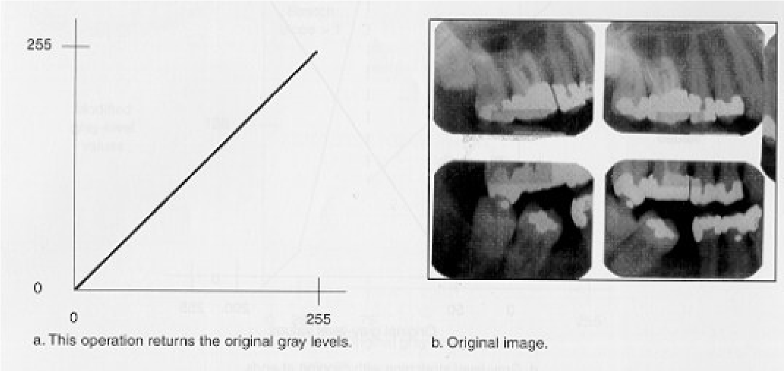
Pixel values between [A,B] are highlighted

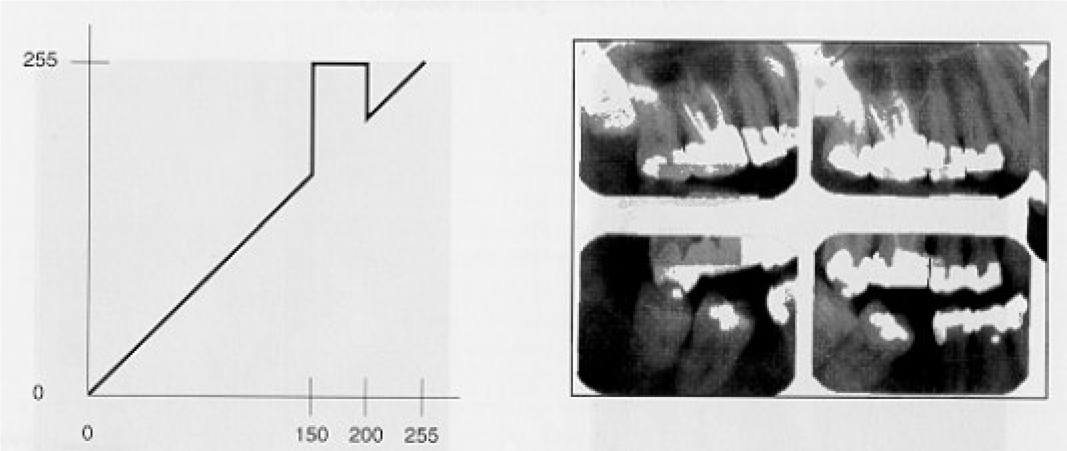
Other pixel values are preserved

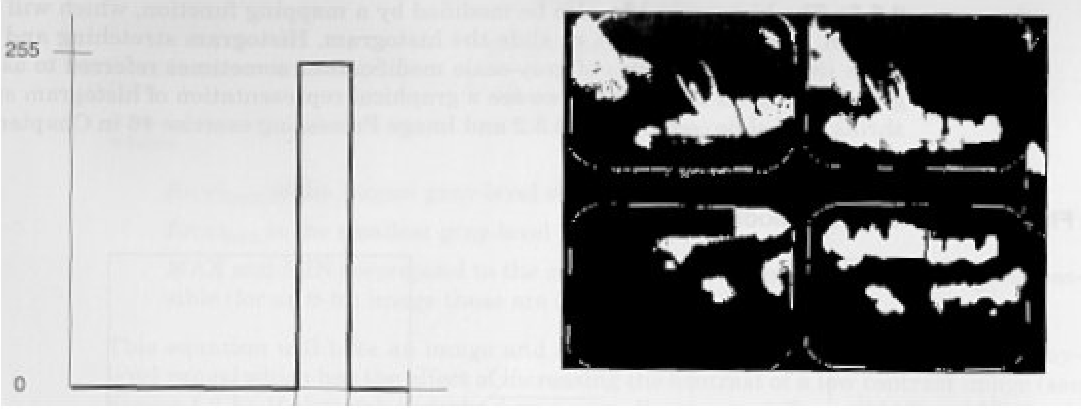
Other pixel values are darkened

© 2002 R. C. Gonzalez & R. E. Woods

Piecewise-Linear Transformations



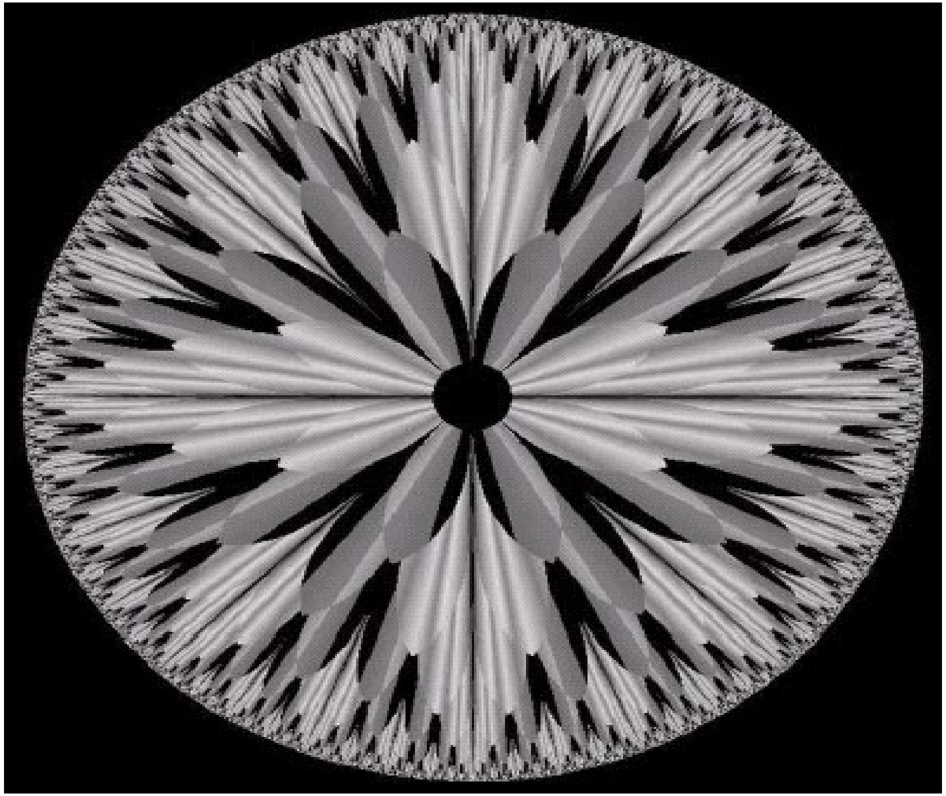




https://www.cse.unr.edu/~bebis

Piecewise-Linear Transformations

Bit Plane slicing



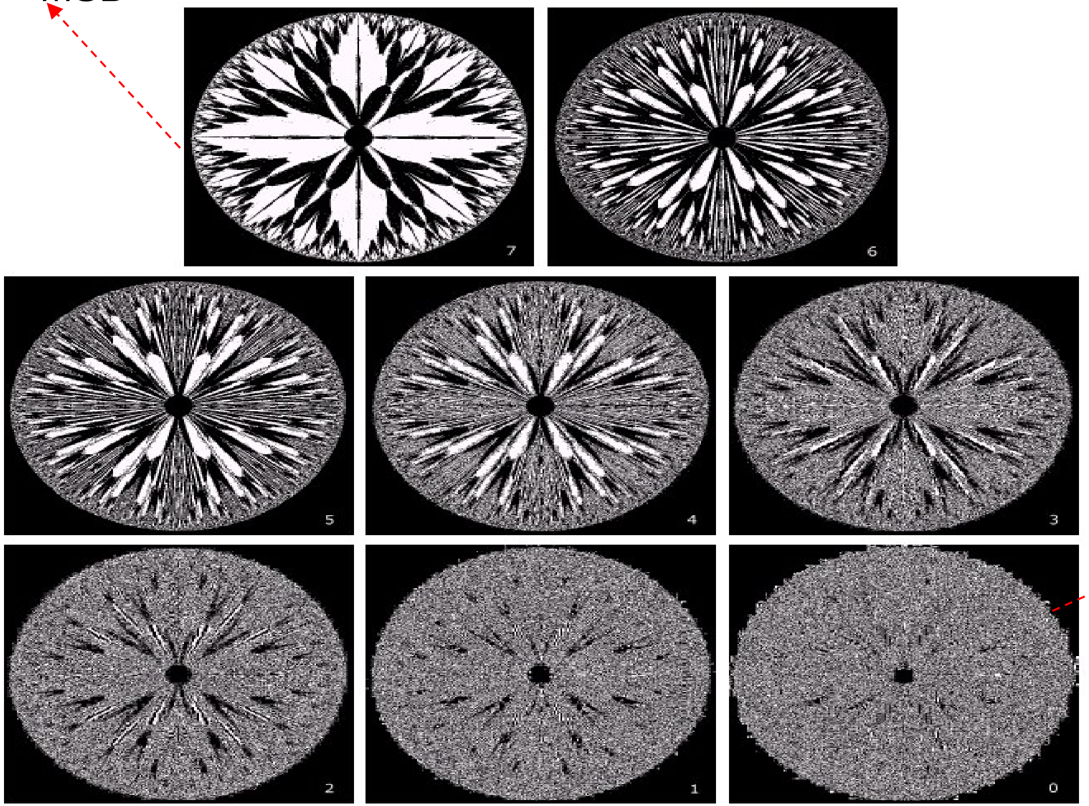
An 8-bit fractal image

© 2002 R. C. Gonzalez & R. E. Woods

Piecewise-Linear Transformations

Bit Plane slicing

MSB



LSB

© 2002 R. C. Gonzalez & R. E. Woods

Histogram Processing

Histogram : is the discrete function $h(r_k)=n_k$, where r_k is the k^{th} gray level in the range of [0, L-1] and n_k is the number of pixels having gray level r_k.

Normalized histogram : is $p(r_k)=n_k/n$, for $k=0,1,\ldots,L-1$ and $p(r_k)$ can be considered to give an estimate of the probability of occurrence of gray level r_k.

Histogram Processing

0	0	1	0	2	0
1	0	7	7	7	0
0	7	0	0	7	0
1	0	0	7	2	0
0	0	7	1	0	1
1	0	7	7	7	0

frequencies

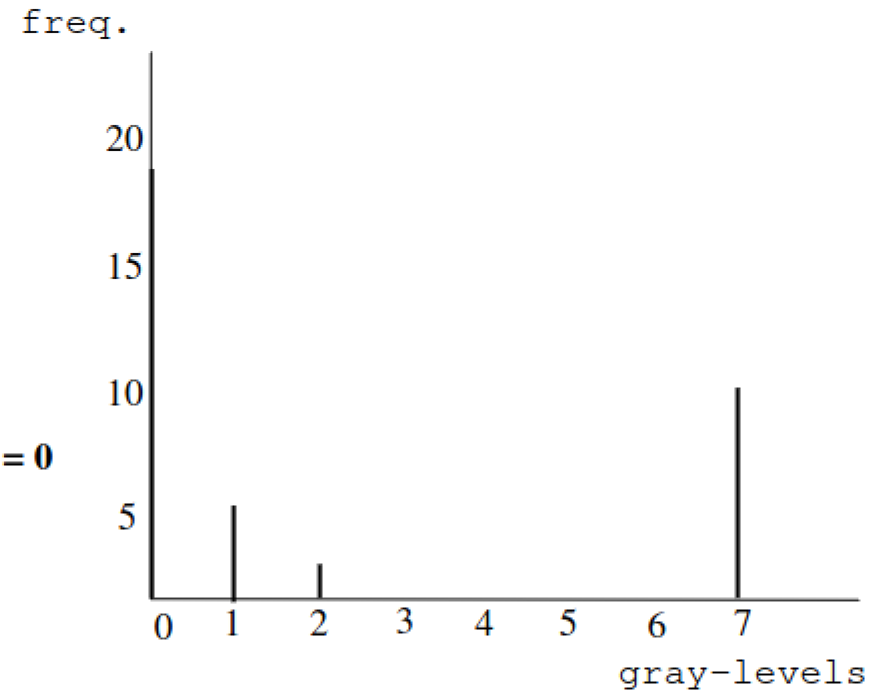
f(0)= 18

f(1)= 6

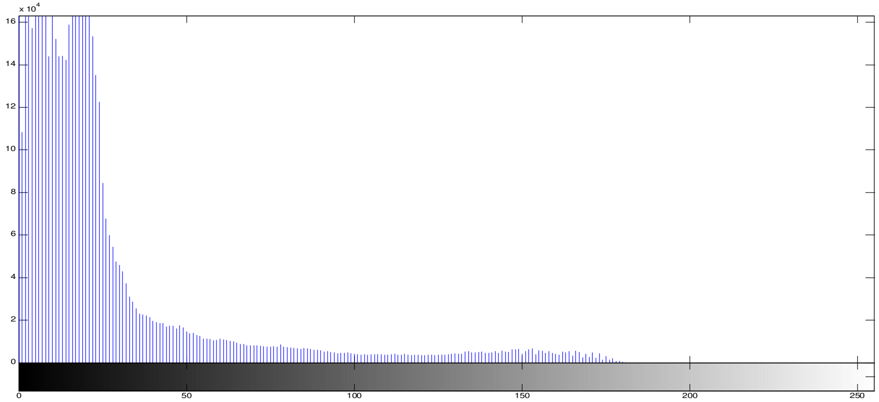
f(2)= 2

f(3)=f(4)=f(5)=f(6) = 0

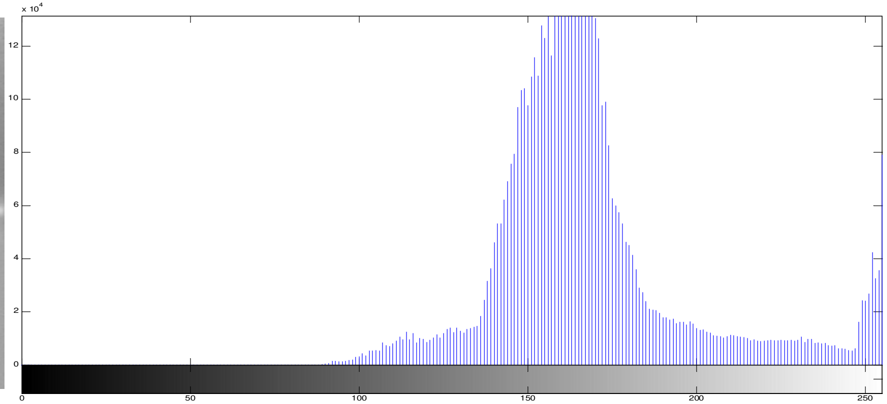
f(7) = 10



Histogram of 4 basic grey-level characteristics



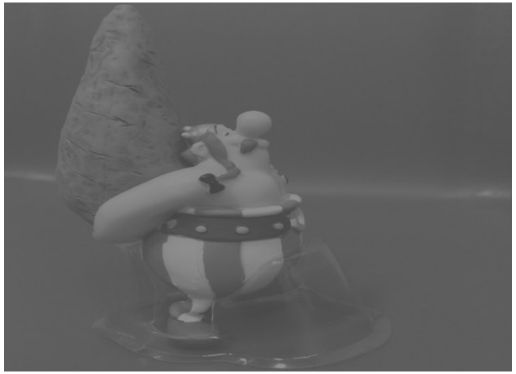
Dark image

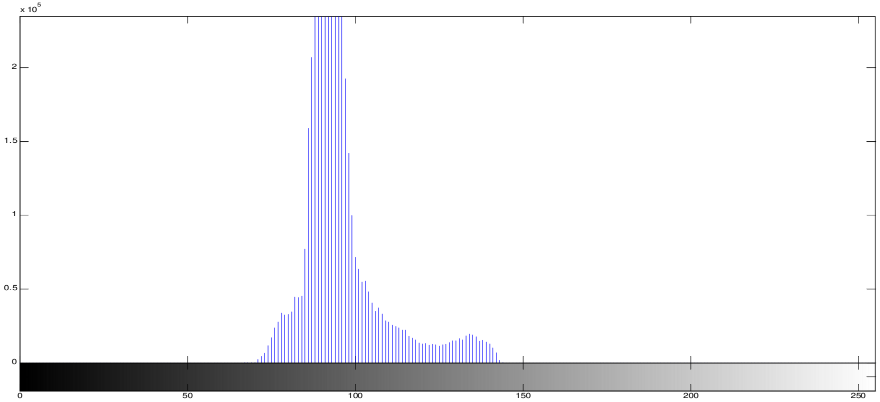


Bright image

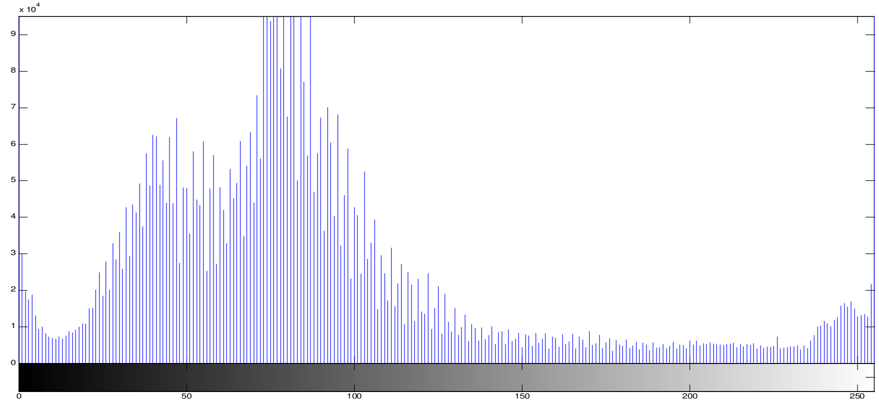
© 2002 R. C. Gonzalez & R. E. Woods

Histogram of 4 basic grey-level characteristics





Low contrast image



High contrast image

© 2002 R. C. Gonzalez & R. E. Woods

Histogram Equalization

Histogram equalization : is a method which increases the dynamic range of the gray-levels in a low-contrast image to cover full range of gray-levels.

How-to-Do: is achieved by having a transformation function which is the Cumulative Distribution Function (CDF) of a given PDF of gray-levels in a given image.

https://sisu.ut.ee/imageprocessing/book/6

Histogram Equalization

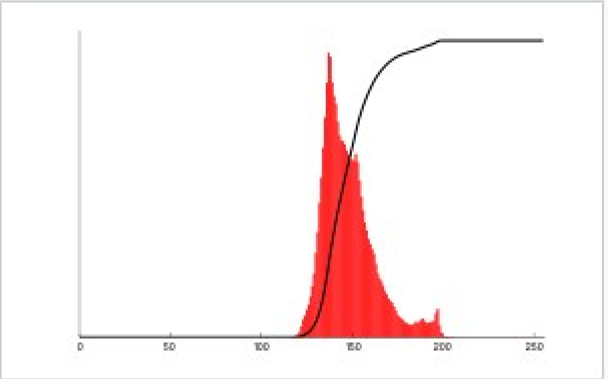
Histogram equalization

The new intensity value of pixel x is calculated by:

$$I(x) = \mathrm{round}\left(\frac{cdf(x) - \min cdf}{1 - \min cdf} \times (L - 1) \right)$$

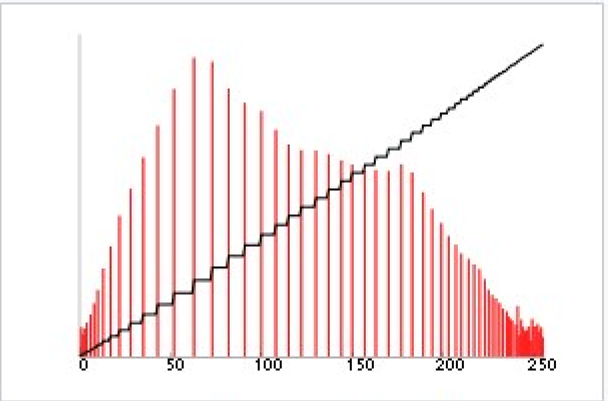
					256		
	original H(x)	original h'(x)	original cdf(h'(x))		new H(x)	new(h'(x))	new(cdf'(x))
1	5	0,125	0,125		32	0,024	0,024
2	3	0,075	0,200		51	0,038	0,061
3	9	0,225	0,425		108	0,080	0,142
4	16	0,400	0,825		210	0,156	0,297
5	0	0,000	0,825		210	0,156	0,453
6	4	0,100	0,925		236	0,175	0,627
7	2	0,050	0,975		249	0,184	0,811
8	1	0,025	1,000		255	0,189	1,000
total	40	1,000			1351,5	1,000	

Before Histogram Equalization



Corresponding histogram (red) and cumulative histogram (black)

After Histogram Equalization



Corresponding histogram (red) and cumulative histogram (black)

https://en.wikipedia.org/wiki/Histogram_equalization

Computational Vision and Perception

Prof. Eduardo Todt

todt@inf.ufpr.br

BY NC SA