Biometrics

Khalid Saeed, DSc, PhD, MSc, BSc Eng.

full profesor at
Bialystok University of Technology, Poland
and **visiting profesor at**
CORPORACIÓN UNIVERSIDAD DE LA COSTA-CUC
BARRANQUILLA, COLOMBIA

previously with
- **Warsaw University of Technology, Poland** (*full professor*),
- **AGH Krakow, Poland** (*professor*),

and
- **Hanbat University, Daejeon, South Korea** (*visiting professor*)

Image Analysis and Processing

SPINAKER
Intensywne Międzynarodowe Programy Kształcenia (nabór 2021) - BPI/SPI/2021/1/00066/U/00001

Some Activities

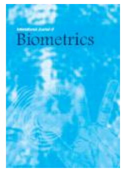
* Head

- **Department of Biometrics and Signal Processing at PB**

*** > 250 publications**

* Editor-in-Chief:

- **International Journal Biometrics, Inderscience, UK (since 2007)**
- *(JCR journal - WoS, SCOPUS)*



* CISIM General Chair and PC Chair:
International Conference on Computer Information Systems and Industrial Management (22 editions)

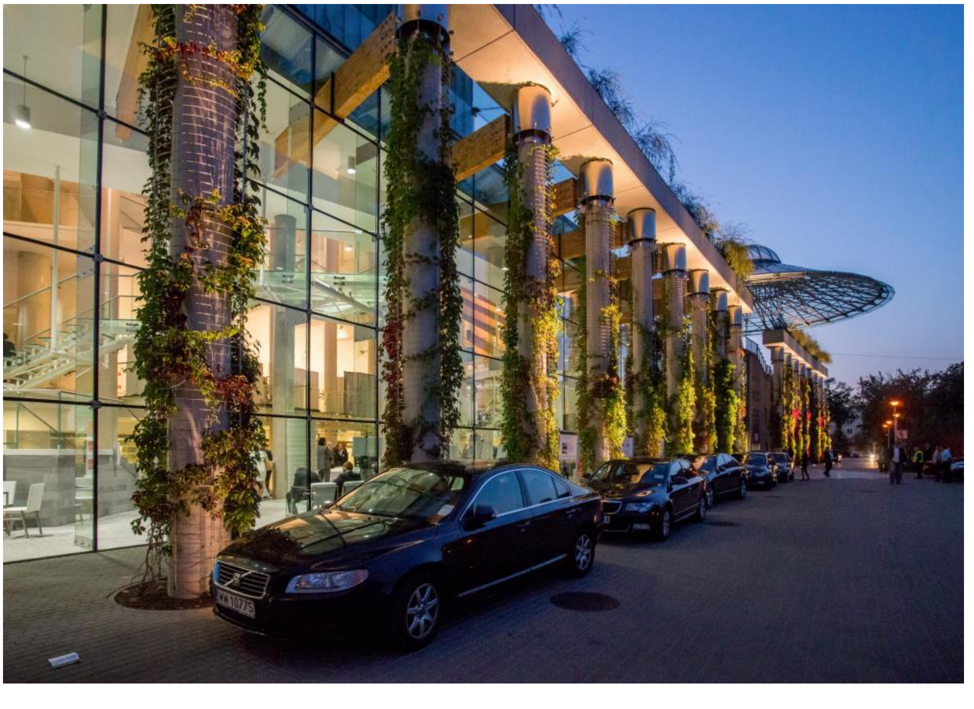
Selected Books

Bialystok


Odense
Kiel
Hamburg, Freie und Hansestadt
Oldenburg (Oldenburg)
Bremen, Hansestadt
Lueneburg
Hannover
Potsdam
Berlin
Magdeburg
Detmold
Dessau
Arnsberg
Kassel
Halle (Saale)
Erfurt
Dresden
Chemnitz
Giessen
Darmstadt
Karlovy Vary
Kladno
Praha
Bayreuth
Plzen
Hradec Kralove
Olomouc
Karlsruhe
Regensburg
Ceske-Budejovice
Brno
Zlin
Trencin
Frydek-Mistek
Presov
Slupsk
Koszalin
Gdańsk
Olsztyn
Szczecin
Pila
Toruń
Ciechanów
Plock
Poznań
Konin
Warszawa
Zielona Góra
Kalisz
Łódź
Piotrków Trybunalski
Radom
Lublin
Wrocław
Opole
Kielce
Zamość
Kraków
Rzeszów
Nowy Sącz
Biała Podlaska
Suwałki
Łomża
Białystok
Kaunas
Marijampole
Alytus
Vilnius
Minsk
Baranowichi
Kovel
Luts'k
Rivne
Zhytomyr
Ternopil
Khmelnitsky
Ivano-Frankivs'k


PAWELTADEJKO.COM

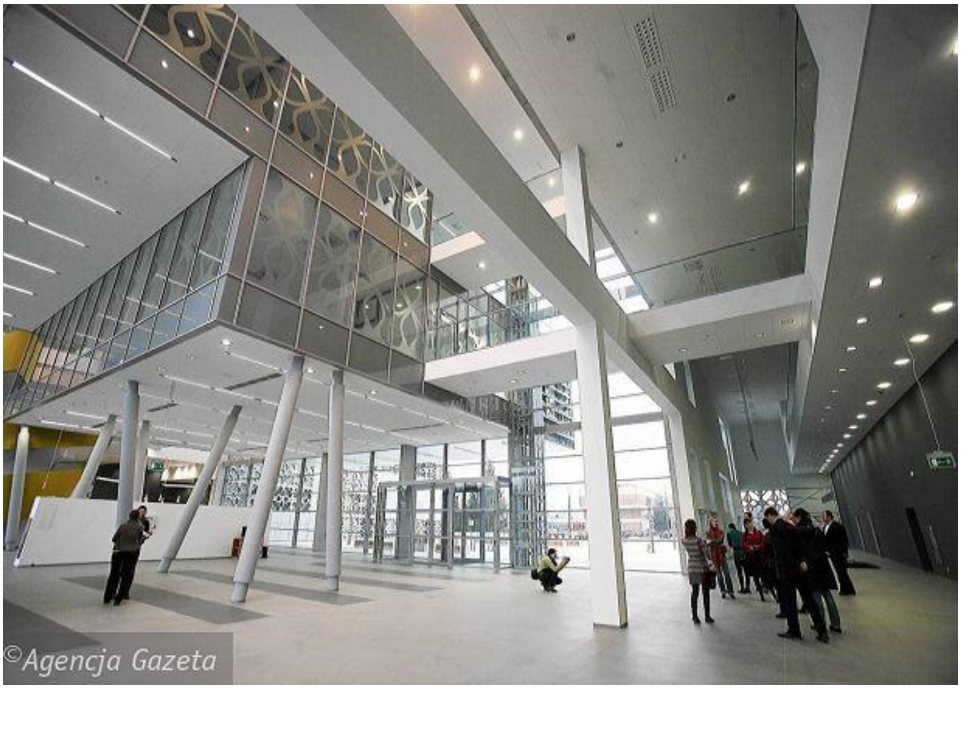

P



Bialystok University of Technology

Faculty of Computer Science

SPINAKER
Intensywne Międzynarodowe Programy Kształcenia (nabór 2021) - BPI/SPI/2021/1/00066/U/00001


©Agencja Gazeta

A word about BUT:

Bialystok University of Technology

- 7 faculties
- 8 000 students
- 680 academic teachers
- 26 first degree courses
- 19 second degree courses
- 8 rights to confer doctoral degrees
- 5 rights to confer post-doctoral degrees
- ~ 90 bilateral scientific agreements with academic centres from 37 countries:
- Japan (Tokyo), Korea Daejeon and Seul), India Kolkata, Czech Republic, China, …
- and Colombia.

Biometrics …..

What is it?

Pattern Recognition?!

What is Biometrics?

Biometrics means Biological Measurements
The name comes from Greek *BIOS* - life
and *METRICOS* – measuring.

Since when have we known Biometrics?

1885-1913 B.C. (Mesopotamia)
Thumbprints were found on ancient Babylonian clay tablets, seals, and pottery. People impressed their fingerprints into the clay tablet on which the legal business transaction contract had been written to protect it against forgery.

Since when?

A transaction deal (left) with the thumbprint on the other side of the clay seal.

Since when?

(*seen in **Museum of London***)

More,
by 246 B.C.E., Chinese officials impressed their fingerprints in clay seals, to stamp documents.

In the 14th century, also in China - hand and foot prints of children were stamped on paper with ***ink*** to distinguish one child from another.

The school will comprise three parts:

1. Introduction to Image Processing
2. Introduction to Biometrics
3. Implemenationand Practical Use of Biometrics Methods

Hence, we will start with Image Processing and Analysis and then go to BM.

Introduction to Analysis and Processing
of Biometric Images

Universal system for
Image Analysis and Processing

Block Diagram

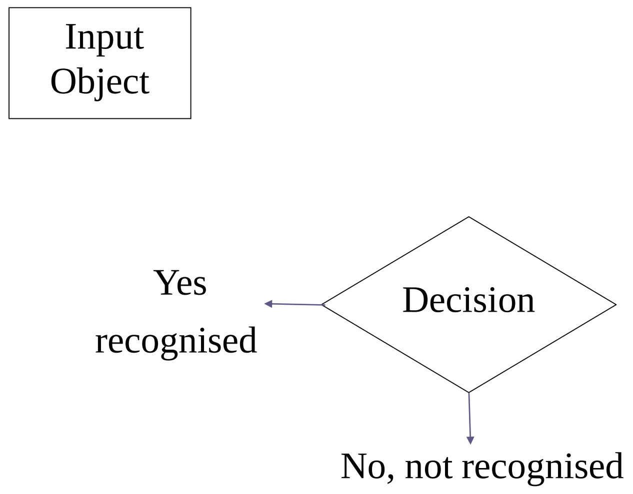
Input Object

Block Diagram

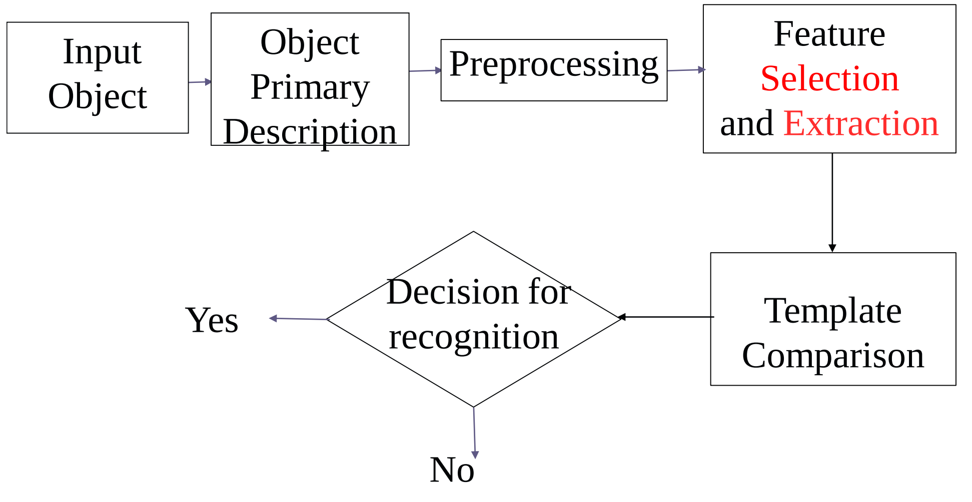
Input Object

For the sake of recognition

Block Diagram



For the sake of recognition



For the sake of recognition

What is Processing,
and what is Analysis?

Processing

Processor

SPINAKER
Intensywne Międzynarodowe Programy Kształcenia (nabór 2021) - BPI/SPI/2021/1/00066/U/00001

Processing

DSP

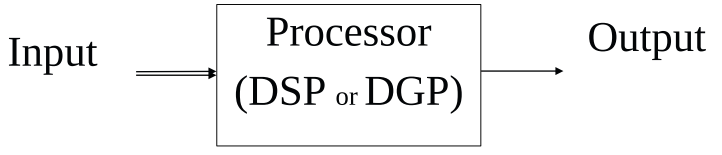
Or,

DGP

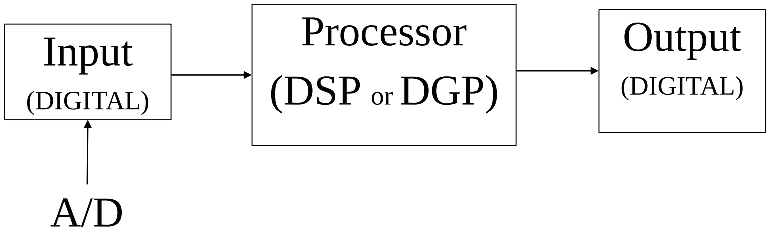
Processing

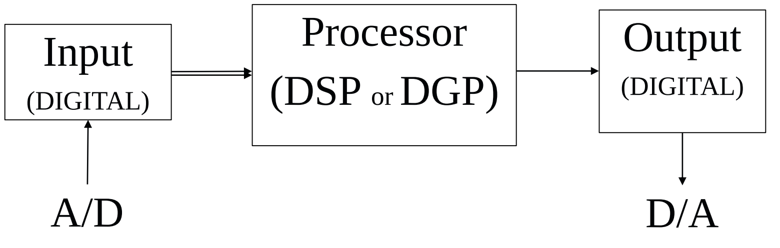
Input → Processor (DSP or DGP)

Processing



Processing


Input
(DIGITAL)
Processor
(DSP or DGP)
Output
(DIGITAL)
A/D


Input
(DIGITAL)
Processor
(DSP or DGP)
Output
(DIGITAL)
A/D
D/A

… and **ANALYSIS**?

Image Analysis
for Object Recognition

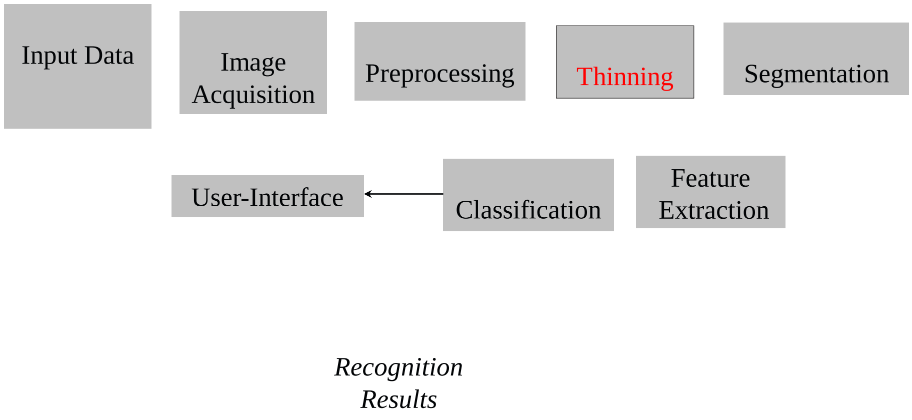
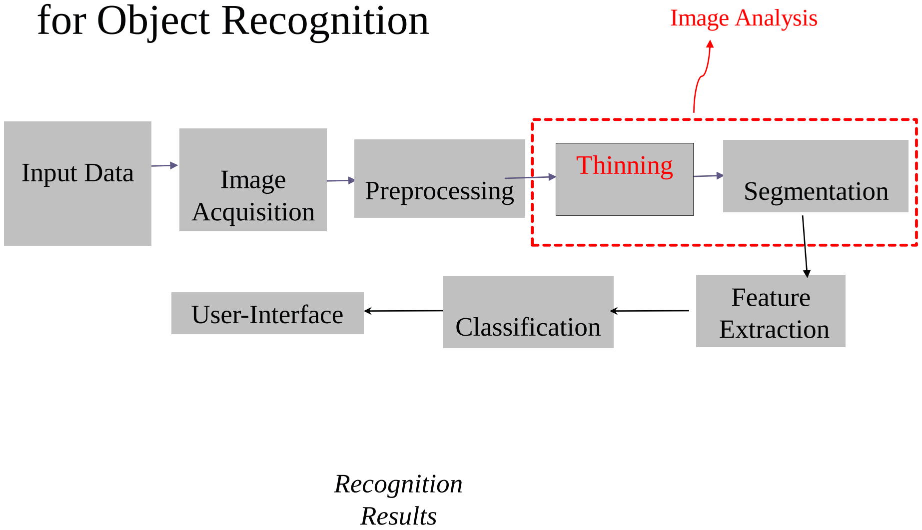
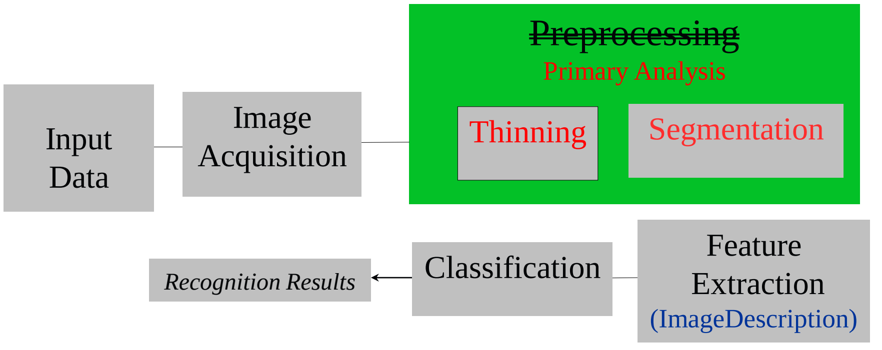


Image Analysis
for Object Recognition



RECALL



Hence, Analysis is the process of:

Image Segmentation and Thinning

Now, we will consider Image Processing and then come back to Image Segmentation and Thinnning.

SPINAKER
Intensywne Międzynarodowe Programy Kształcenia (nabór 2021) - BPI/SPI/2021/1/00066/U/00001

Image Processing

Image Processing, *in Some Understanding, is the process of* **Image Transformation**.

The following study will let us understand the ways of image PROCESSING, that is: changing/transformation/adapting/converting/altering depending on the aim of the development. We will pay special attention to the OBJECT RECOGNITION

Methods of Image Transformation (PROCESSING)

1. **Methods of Point-Wise Operations – LUT (Look Up Table)**
2. **Geometric Methods (Image Position)**
3. **Processing Methods by Image Filtering**

Methods of Image Transformation (PROCESSING)

1. Methods of Point-Wise Operations
LUT (Look Up Table)

Operations on special parts of the the image.

The most populaer method of processing: VISION EFFECTS like Brightness and Contrast, Autoscaling, … etc.

The operations are done on monochromatic versions.
For the case of colored images, the operaions should be done on each channel independently.

Methods of Image Transformation (PROCESSING)

1. Methods of Point-Wise Operations

LUT (Look Up Table)

Operations on special parts of the the image.

The most popular methods of processing are: VISION EFFECTS like Brightness and Contrast, Autoscaling, … etc.

The operations are done on monochromatic versions.
For the case of colored images, the operaions should be done on each channel independently.

Methods of Image Transformation (PROCESSING)

2. Geometric Methods (Image Position)

In particular they concern the SENSOR ERROR correction resulting after image capturing, aquiring and then forwarding to the system for further processing.

Examples of such sensors are the camera, scanner, … *which often cause image shifting, rotation*, .. etc.

Methods of Image Transformation (PROCESSING)

3a. Processing Methods by Image Filtering

* **Spectral Filters** (Whole-Image Filtering). Fourier Transform is a good example.

Input Image → FT → Image Spectrum → Removal of unnecessary (e.g., low or high frequencies → Inverse FT to get the output image as the ORIGINAL one BUT without the unwanted parts.

Methods of Image Transformation (PROCESSING)

3b. Processing Methods by Image Filtering

* **Context Filters** (filtering some selected parts or regions of an image and here may go all known popular filters)

Methods of Image Transformation (PROCESSING)

3c. Processing Methods by Image Filtering

* **morphological** (conditonal filtering – conditions should be satisfied)

Methods of Image Transformation (PROCESSING)

1. Methods of Point-Wise Operations – LUT (Look Up Table)
Operations on special parts of the the image.
The most populaer method of processing: VISION EFFECTS like Brightness and Contrast, Autoscaling, … etc.

2. Geometric Methods (Image Position)
In particular they concern the SENSOR ERROR correction resulting after image capturing, aquiring and then forwarding to the system for further processing.
Examples of such sensors are the camera, scanner, … which often cause image shifting, rotation, .. etc.

3. Processing Methods by Image Filtering

* **Spectral Filters** (Whole-Image Filtering). Fourier Transform is a good example.

Input Image → FT → Image Spectrum → Removal of unnecessary (e.g., low or high frequencies → Inverse FT to get the output image as the ORIGINAL one BUT without the unwanted parts.

* **Context Filters** (filtering selected parts or regions of an image

Here may go all known popular filters

* **morphological** (conditonal filtering – conditions should be satisfied)

1. Look Up Table - LUT

(Point-Wise Operations)

(Point Processing of Images)

The most popular LUT operations on Images are the *Brightness* and *Contrast*.

1) BRIGHTNESS
(Linear Correction /Linear Brightness Enhancement)

Brightness (or also Darkness) is a linear operation:

$$W(x) = ax + b$$

Assume a = 1, then $W(x) = x + b$, with x to represent intensity (intensity value) in [0,1].

Now,

if b > 0, then we have *brightening*,
if b < 0, then we have *darkening*.

The most popular LUT operations on Images are the *Brightness* and *Contrast*.

1) BRIGHTNESS
(Linear Correction /Linear Brightness Enhancement)

Brightness (or also Darkness) is a linear operation:

$$W(x) = ax + b$$

Assume a = 1, then $W(x) = x + b$, with x to represent intensity (intensity value) in [0,1].

Now,
if b > 0, then we have *brightening*,
if b < 0, then we have *darkening*.

The most popular LUT operations on Images are the *Brightness* and *Contrast*.

1) BRIGHTNESS
(Linear Correction /Linear Brightness Enhancement)

Brightness (or also Darkness) is a linear operation:

$$W(x) = ax + b$$

Assume a = 1, then $W(x) = x + b$, with x to represent intensity (intensity value) in [0,1].

Now,
if b > 0, then we have *brightening*,
if b < 0, then we have *darkening*.

The most popular LUT operations on Images are the *Brightness* and *Contrast*.

1) BRIGHTNESS
(Linear Correction /Linear Brightness Enhancement)

Brightness (or also Darkness) is a linear operation:

$$W(x) = ax + b$$

Assume a = 1, then $W(x) = x + b$, with x to represent intensity (intensity value) in [0,1].

Now,

if b > 0, then we have *brightening*,

if b < 0, then we have *darkening*.

The most popular LUT operations on Images are the *Brightness* and *Contrast*.

1) BRIGHTNESS
(Linear Correction /Linear Brightness Enhancement)

Brightness (or also Darkness) is a linear operation:

$$W(x) = ax + b$$

Assume a = 1, then $W(x) = x + b$, with x to represent intensity (intensity value) in [0,1].

Now,

if b > 0, then we have *brightening*,

if b < 0, then we have *darkening*.

The most popular LUT operations on Images are the *Brightness* and *Contrast*.

1) BRIGHTNESS
(Linear Correction /Linear Brightness Enhancement)

Brightness (or also Darkness) is a linear operation:

$$W(x) = ax + b$$

Assume a = 1, then $W(x) = x + b$, with x to represent intensity (intensity value) in [0,1].

Now,

if b > 0, then we have *brightening*,
if b < 0, then we have *darkening*.

2) CONTRAST – Linear Contrast Enhancement
The second important function performed by LUT.

(**contrast** means to set in opposition in order to emphasize differences: city and country life, for example B&W in LEDded LCD.

In W = ax + b, for different values of a and b,
If b > 0, then low contrast,
if b < 0, then high contrast.

When a = -1, b = +1, then image will be inversed, white → black, and black → white

2) CONTRAST – Linear Contrast Enhancement

The second important function performed by LUT.

(**contrast -** opposition in order to emphasize differences: for example city and country life, B&W in LEDded LCD).

In W = a*x* + b, for different values of a and b,

If b > 0, then low contrast,

if b < 0, then high contrast.

When a = -1, b = +1, then image will be inversed, white → black, and black → white

2) CONTRAST – Linear Contrast Enhancement

The second important function performed by LUT.

(**contrast -** opposition in order to emphasize differences: for example city and country life, B&W in LEDded LCD).

In W = ax + b, for different values of a and b,

If b > 0, then low contrast,

if b < 0, then high contrast.

When a = -1, b = +1, then image will be inversed, white → black, and black → white

2) CONTRAST – Linear Contrast Enhancement

The second important function performed by LUT.

(**contrast** - opposition in order to emphasize differences: for example city and country life, B&W in LEDded LCD).

In W = a*x* + b, for different values of a and b,

If b > 0, then low contrast,

if b < 0, then high contrast.

When a = -1, b = +1, then image will be inversed, white → black, and black → white

2) CONTRAST – Linear Contrast Enhancement

The second important function performed by LUT.

(**contrast** - opposition in order to emphasize differences: for example city and country life, B&W in LEDded LCD).

In W = a*x* + b, for different values of a and b,

If b > 0, then low contrast,

if b < 0, then high contrast.

When a = -1, b = +1, then image will be inversed, white → black, and black → white

2) CONTRAST – Linear Contrast Enhancement

The second important function performed by LUT.

(**contrast** - opposition in order to emphasize differences: for example city and country life, B&W in LEDded LCD).

In W = a*x* + b, for different values of a and b,

If b > 0, then low contrast,

if b < 0, then high contrast.

When a = -1, b = +1, then image will be inverstd, white → black, and black → white

LUT System:

input *x* → LUT → output *W*(*x*)

LUT System:

input x → LUT → output $W(x)$

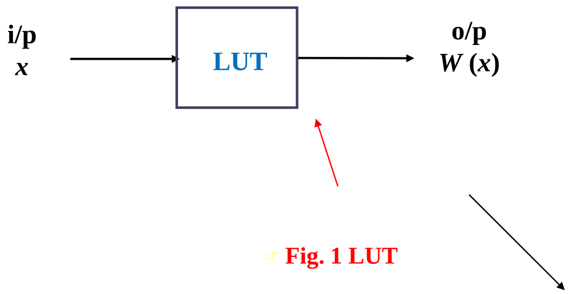
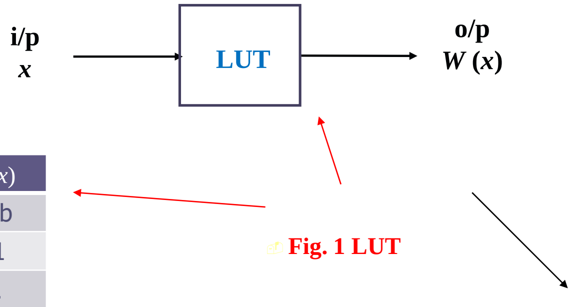


Fig. 1 LUT

LUT System:

input x → LUT → output $W(x)$

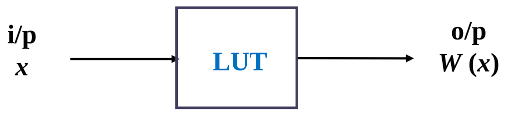


x	$W(x)$
aaa	bbb
a1	b1
…	…

Fig. 1 LUT

LUT System:

input x → LUT → output $W(x)$



x	$W(x)$
aaa	bbb
a1	b1
…	…

Fig. 1 LUT

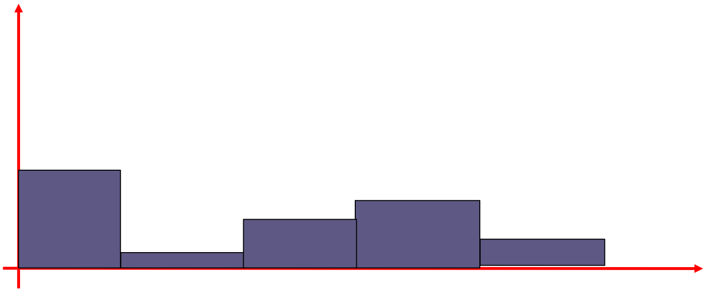
- **For example, information about the image pixels as an array or a matrix $y = W(x)$, *with* x to be the tested pixel and y its value in (0-255).**
  Thus arises what is called HISTOGRAM

HISTOGRAM

HISTOGRAM

(Information about image pixel distribution)

4	2	4	4	4	4	4	4	4	4
2	2	2	4	4	1	4	4	4	4
4	2	4	4	1	1	1	4	4	4
4	4	4	1	1	1	1	1	4	4
4	4	1	1	1	1	1	1	1	4
4	1	1	1	1	1	1	1	1	1
3	3	1	1	1	1	5	5	1	3
3	3	1	1	1	1	5	5	1	3
3	3	1	1	1	1	5	5	1	3
3	3	1	1	1	1	5	5	1	3



Histogram

4	2	4	4	4	4	4	4	4	4
2	2	2	4	4	1	4	4	4	4
4	2	4	4	1	1	1	4	4	4
4	4	4	1	1	1	1	1	4	4
4	4	1	1	1	1	1	1	1	4
4	1	1	1	1	1	1	1	1	1
3	3	1	1	1	1	5	5	1	3
3	3	1	1	1	1	5	5	1	3
3	3	1	1	1	1	5	5	1	3
3	3	1	1	1	1	5	5	1	3

LUT Extension

If the input *x* is the value of the pixel defined in the domain from 0 to ($2^B - 1$), where B = number of bits representing one image point (pixel value),

then at the output (the display) as an EFFECT, *W* can be described according to the user's need or the application:

- $W(x) = a\,x^n$, $n > 0$

- $n = 1$ $\quad n < 1$ $\quad n > 1$
- $W(x) = ax$; $\quad a\sqrt{x}$; $\quad ax^2$ >>> see Fig. 2

LUT Extension

If the input *x* is the value of the pixel defined in the domain from 0 to ($2^B – 1$), where B = number of bits representing one image point (pixel value),

then at the output (the display) as an EFFECT,
***W* can be described according to the user's need or the application:**

- $W(x) = a\,x^n$, $n > 0$

$n = 1$	$n < 1$	$n > 1$
$W(x) = ax$;	$a\sqrt{x}$;	ax^2 >>> see Fig. 2

LUT Extension

If the input *x* is the value of the pixel defined in the domain from 0 to ($2^B – 1$), where B = number of bits representing one image point (pixel value),

then at the output (the display) as an EFFECT,
***W* can be described according to the user's need or the application:**

- $W(x) = a\,x^n$, $n > 0$

$n = 1$	$n < 1$	$n > 1$
$W(x) = ax$;	$a\sqrt{x}$;	ax^2 >>> see Fig. 2

LUT Extension

If the input x is the value of the pixel defined in the domain from 0 to $(2^B – 1)$, where B = number of bits representing one image point (pixel value),

then at the output (the display) as an EFFECT, W can be described according to the user's need or the application:

- $W(x) = a\, x^n , n > 0$

- $n = 1$ | $n < 1$ | $n > 1$
- $W(x) = ax$; | $a\sqrt{x}$; | ax^2 >>> see Fig. 2

LUT Extension

If the input x is the value of the pixel defined in the domain from 0 to $(2^B – 1)$, where B = number of bits representing one image point (pixel value),

then at the output (the display) as an EFFECT, W can be described according to the user's need or the application:

- $W(x) = a\, x^n , n > 0$

input x → LUT → output $W(x)$

If the input x is the value of the pixel defined in the domain from 0 to ($2^B - 1$), where B = number of bits representing one image point (pixel value),

then at the output (the display) as an EFFECT,
W **can be described according to the user's need or the application:**

- $W(x) = a\,x^n$, $n > 0$

$n = 1$	$n < 1$	$n > 1$
$W(x) = ax$;	$a\sqrt{x}$;	ax^2 >>> see Fig. 2

LUT Extension

If the input x is the value of the pixel defined in the domain from 0 to ($2^B - 1$), where B = number of bits representing one image point (pixel value),

then at the output (the display) as an EFFECT,
W **can be described according to the user's need or the application:**

- $W(x) = a\,x^n$, $n > 0$

$n = 1$	$n < 1$	$n > 1$
$W(x) = ax$;	$a\sqrt{x}$;	ax^2 >>> see Fig. 2

LUT Extension

If the input x is the value of the pixel defined in the domain from 0 to $(2^B - 1)$, where B = number of bits representing one image point (pixel value),

then at the output (the display) as an EFFECT,
W can be described according to the user's need or the application:

- $W(x) = a\,x^n$, $n > 0$

$n = 1$	$n < 1$	$n > 1$
$W(x) = ax$;	$a\sqrt{x}$;	ax^2 >>> see Fig. 2

LUT Extension

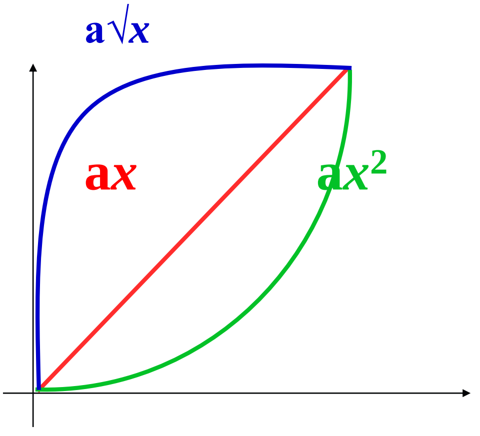
If the input x is the value of the pixel defined in the domain from 0 to $(2^B - 1)$, where B = number of bits representing one image point (pixel value),

then at the output (the display) as an EFFECT,
W can be described according to the user's need or the application:

- $W(x) = a\,x^n$, $n > 0$

$n = 1$	$n < 1$	$n > 1$
$W(x) = ax$;	$a\sqrt{x}$;	ax^2 >>> see Fig. 2

SPINAKER
Intensywne Międzynarodowe Programy Kształcenia (nabór 2021) - BPI/SPI/2021/1/00066/U/00001



- **Consider the case when a = 1 in $W(x)$ = ax, i.e., $W(x) = x$, which means that the otput image is identical to the input one and hence LUT does not change anything.**
- **Thus, it is said *x is directly mapped into $W(x)$*.**

- **Now, if x represents the Image Intensity, B = 8 → 0 - 255,**
- **then, as an effect, the brightness $W(x)$ will vary between black (at x = 0) and white (at x = 255).**

>>>>>>> See Fig. 3

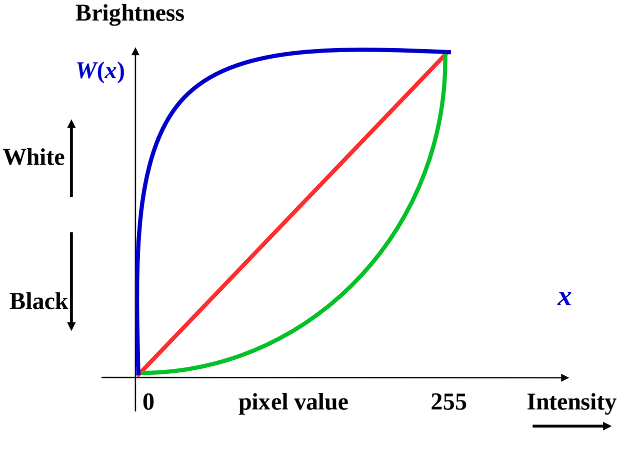


Fig. 3. Visual Effects – linear and nonlinear

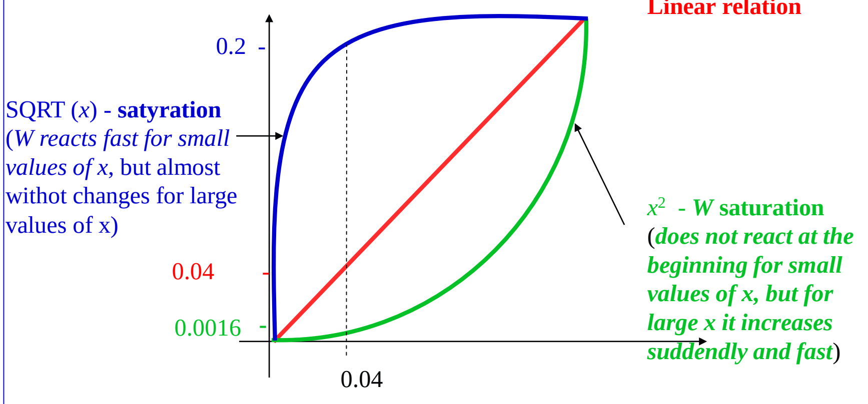


Fig. 4: x, SQRT(x), x^2 for a=1

Hence, we can get varieties of visual effects by LUT.

Consider now the graphical representation for BRIGHTNESS and CONTRAST aganist pixel intensity:

(*LINEAR Brightness Enhancement*)

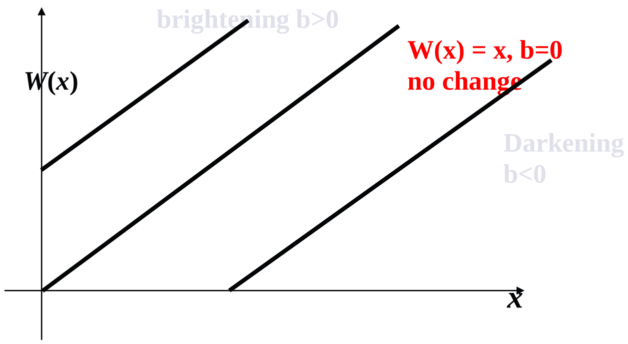


Fig.. 4 Brightening and Darkening

(*LINEAR Brightness Enhancement*)

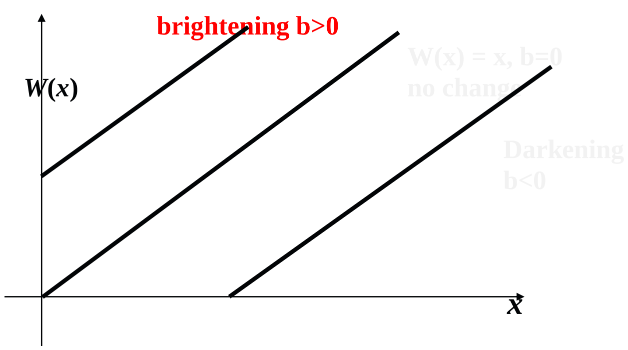


Fig.. 4 Brightening and Darkening

(*LINEAR Brightness Enhancement*)

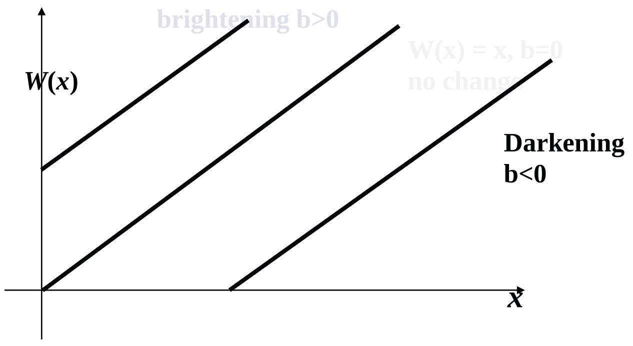


Fig.. 4 Brightening and Darkening

2) CONTRAST – Linear Contrast Enhancement

The second important function performed by LUT.

(**contrast** means to set in opposition in order to emphasize differences: city and country life, for example B&W in LEDded LCD.

In W = a*x* + b, for different values of a and b,

If b > 0, then low contrast,

if b < 0, then high contrast.

When a = -1, b = +1, then image will be inverstd, white → black, and black → white

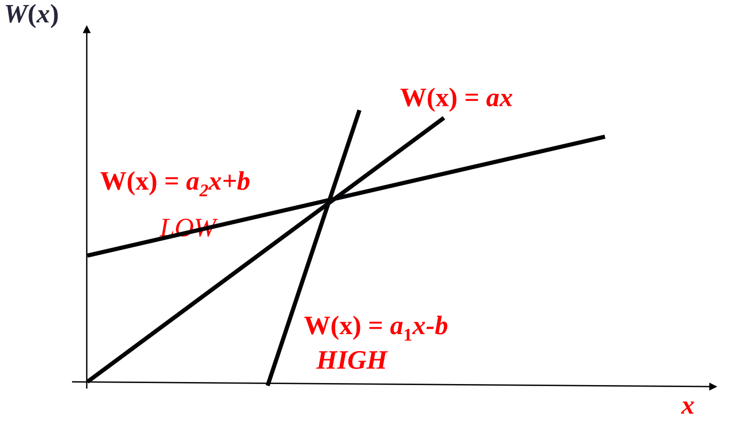


Fig. 5 Contrast – Changing in *b*

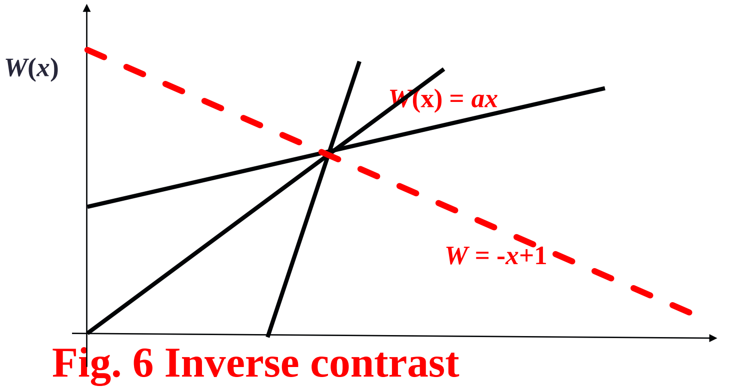


Fig. 6 Inverse contrast

Hence, ***contrast and brightnes are changed by only modifying the dynamic intensity wherever required and/or desired.***

Drawbacks of linear contrast correction

The enhancement or correction will usually lead to saturation.

>>> Fig. 7

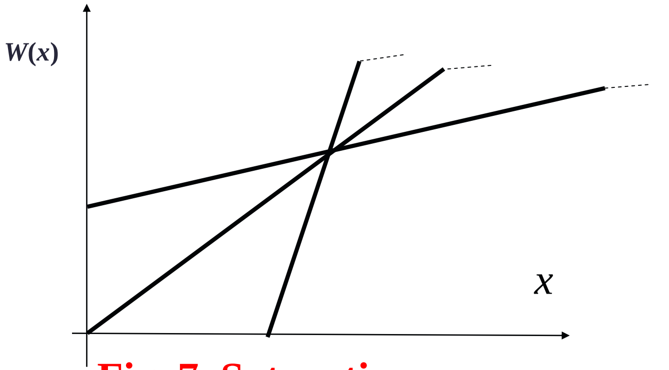


Fig. 7 Saturation

Problem solution:

1. Autoscaling

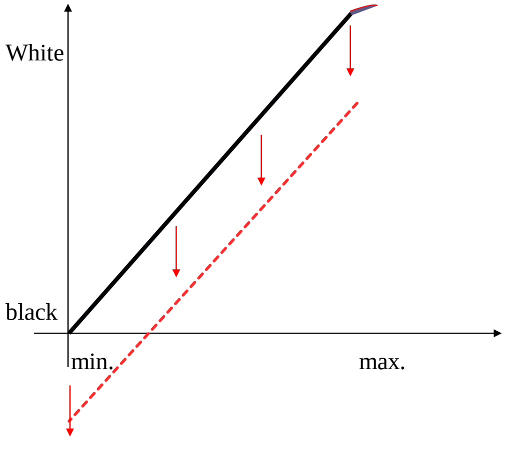


Fig. 8 Autoscaling – *Curve shifting, that is changing b value*

- **2. Intensity values sorting in increasing order and then mapping 1% → black, 99% → white.**
- **This leads to smaller amount of saturation, but would lead to a good contrast.**

2. Problem solving: *curve cut*

- **White:100% → 99%**
- **Black: 0% → 1%**

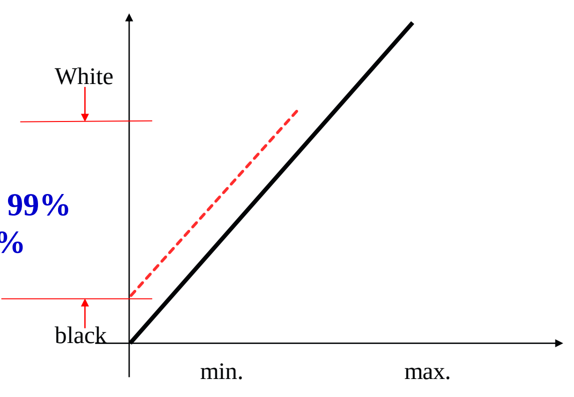


Fig. 8 Curve cutting

- **3. Problem Solution:**
- **Gamma Correction >>> Fig. 9**

- **Gamma Correction**
- **Recall**

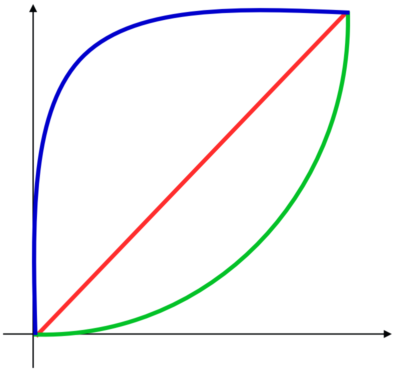
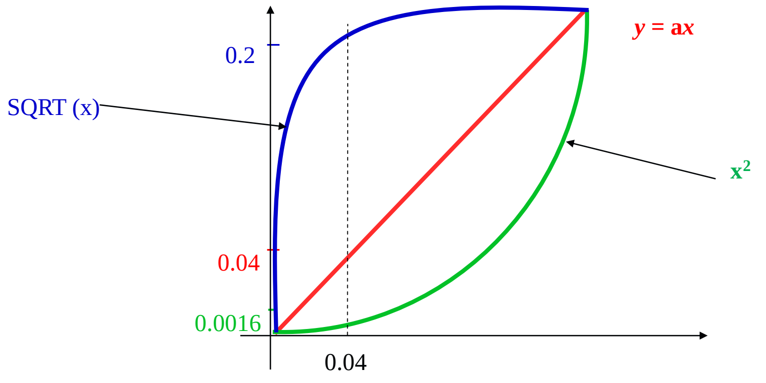


Fig. 9



Rys. 2. x, $\sqrt{x}$, x^2 dla a=1

- **Gamma Correction would give**
- **more balanced correction, after scaling -- 2 degrees of freedom**
- **>>> see Rys. 10**

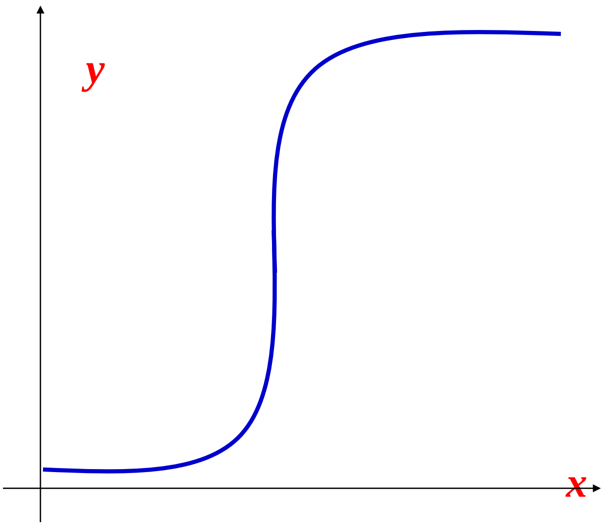
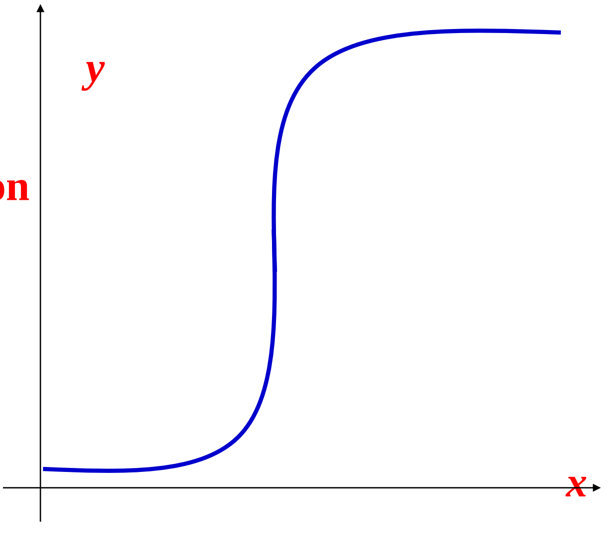
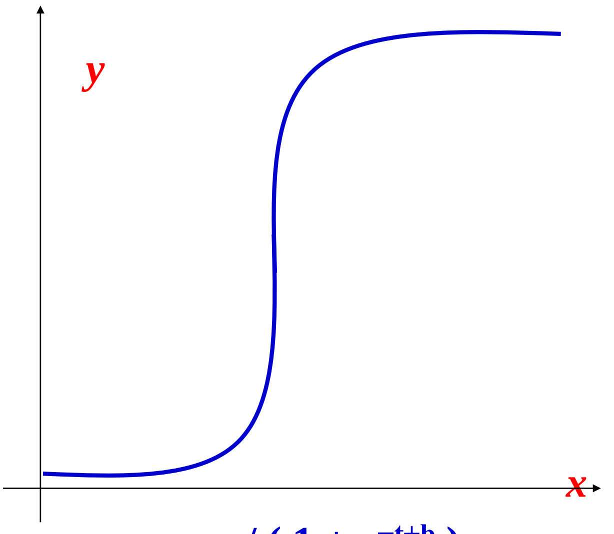


Fig. 10 Contrast Sigmoid – gamma correction



Define *y* as a function of *x* !!!

Fig. 10 Contrast Sigmoid – gamma correction



$a / (1 + e^{-t+b})$

Fig. 10 Contrast Sigmoid – gamma correction

Quantize and Threshold

Quantize **(2^b with *b* being the bit value)**

Threshold **- also 2^b) but *b*=1, that is one bit value to get 2 intensity values)**

Quantize

Quanizing with 5 step function:

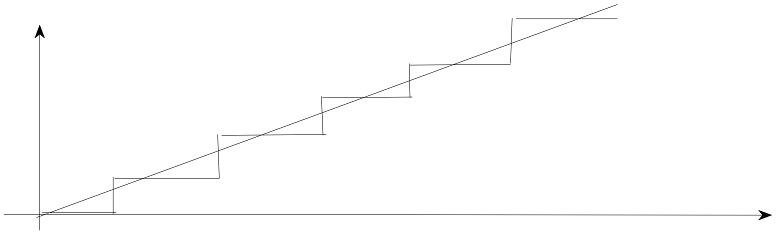


Fig. 11 Step function with 5 intesity values

Threshold

Thresholding with a 2-step function

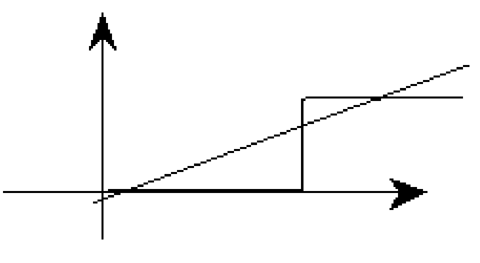


Fig. 12 Step function with 2 intesity values

Methods of Image Transformation (PROCESSING)
Part II

Methods of Image Transformation (PROCESSING)
Part II

1. Methods of Point-Wise Operations – LUT (Look Up Table)
Operations on special parts of the the image.
The most populaer method of processing: VISION EFFECTS like Brightness and Contrast, Autoscaling, … etc.

2. Geometric Methods (Image Position)
In particular they concern the SENSOR ERROR correction resulting after image capturing, aquiring and then forwarding to the system for further processing.
Examples of such sensors are the camera, scanner, … which often cause image shifting, rotation, .. etc.

3. Processing Methods by Image Filtering

* **Spectral Filters** (Whole-Image Filtering). Fourier Transform is a good example.
Input Image → FT → Image Spectrum → Removal of unnecessary (e.g., low or high frequencies → Inverse FT to get the output image as the ORIGINAL one BUT without the unwanted parts.

* **Context Filters** (filtering selected parts or regions of an image
Here may go all known popular filters

* **morphological** (conditonal filtering – conditions should be satisfied)

Methods of Image Transformation (PROCESSING)
Part II

1. Methods of Point-Wise Operations – LUT (Look Up Table)
Operations on special parts of the the image.
The most populaer method of processing: VISION EFFECTS like Brightness and Contrast, Autoscaling, … etc.

2. Geometric Methods (Image Position)
In particular they concern the SENSOR ERROR correction resulting after image capturing, aquiring and then forwarding to the system for further processing.
Examples of such sensors are the camera, scanner, … which often cause image shifting, rotation, .. etc.

3. Processing Methods by Image Filtering
* **Spectral Filters** (Whole-Image Filtering). Fourier Transform is a good example.
Input Image → FT → Image Spectrum → Removal of unnecessary (e.g., low or high frequencies → Inverse FT to get the output image as the ORIGINAL one BUT without the unwanted parts.
* **Context Filters** (filtering selected parts or regions of an image
Here may go all known popular filters
* **morphological** (conditonal filtering – conditions should be satisfied)

2. Geometric Methods (Image Position)

In particular they concern the SENSOR ERROR correction resulting after image capturing, aquiring and then forwarding to the system for further processing.
Examples of such sensors are the camera, scanner, … which often cause image shifting, rotation, .. etc.

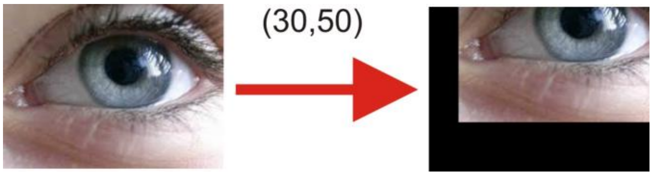
Geometric Methods

- Shifting,
- Rotation
- Reflection (mirror effect)
- Deformation
- Linear translation

Shifting

- Input Image $Z(x,y)$
- Output Image $D(m,n)$
- Vector (p,q)

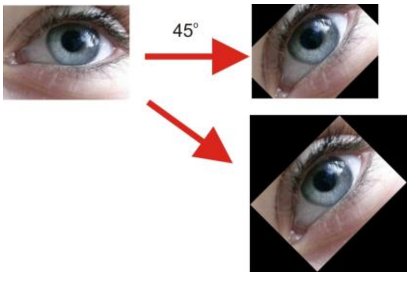
$$m=x+p,\ n=y+q$$



Rotation

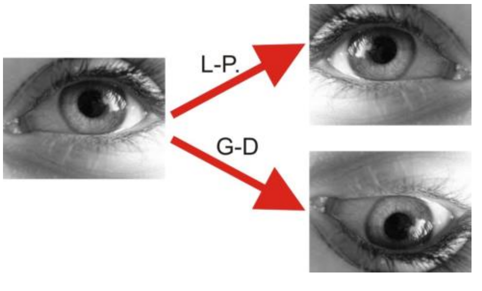
- Input Image Z(x,y)
- Output Image D(m,n)
- Angle α

$m = x \cos \alpha - y \sin \alpha$
$n = x \cos \alpha + y \sin \alpha$

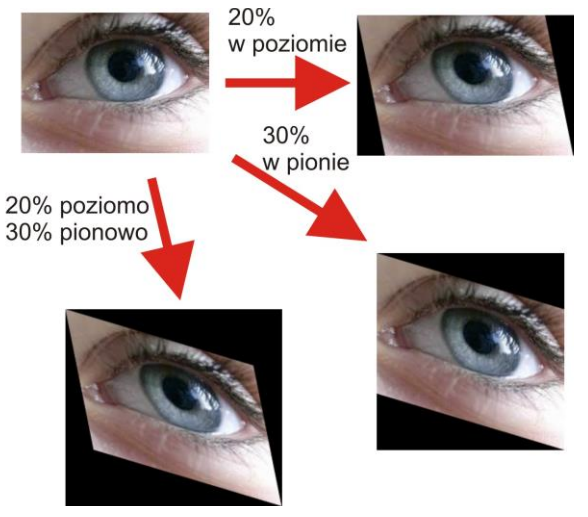


Reflection

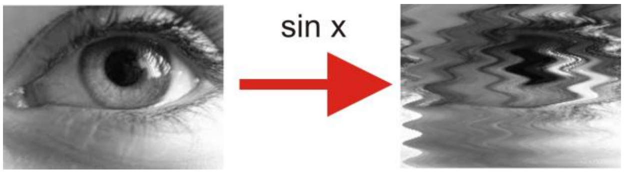
- Horizontal (L-P)
- Vertical (G-D)



Translation



Deformation



Part III

Filters

Image Processing by Filtering

Methods of Image Transformation (PROCESSING)
Part III

1. Methods of Point-Wise Operations – LUT (Look Up Table)
Operations on special parts of the the image.
The most populaer method of processing: VISION EFFECTS like Brightness and Contrast, Autoscaling, … etc.

2. Geometric Methods (Image Position)
In particular they concern the SENSOR ERROR correction resulting after image capturing, aquiring and then forwarding to the system for further processing.
Examples of such sensors are the camera, scanner, … which often cause image shifting, rotation, .. etc.

3. Processing Methods by Image Filtering
* **Spectral Filters** (Whole-Image Filtering). Fourier Transform is a good example.
Input Image → FT → Image Spectrum → Removal of unnecessary (e.g., low or high frequencies → Inverse FT to get the output image as the ORIGINAL one BUT without the unwanted parts.
* **Context Filters** (filtering selected parts or regions of an image
Here may go all known popular filters
* **morphological** (conditonal filtering – conditions should be satisfied)

Methods of Image Transformation (PROCESSING)
Part II

1. Methods of Point-Wise Operations – LUT (Look Up Table)
Operations on special parts of the the image.
The most populaer method of processing: VISION EFFECTS like Brightness and Contrast, Autoscaling, … etc.

2. Geometric Methods (Image Position)
In particular they concern the SENSOR ERROR correction resulting after image capturing, aquiring and then forwarding to the system for further processing.
Examples of such sensors are the camera, scanner, … which often cause image shifting, rotation, .. etc.

3. Processing Methods by Image Filtering
* **Spectral Filters** (Whole-Image Filtering). Fourier Transform is a good example.
Input Image → FT → Image Spectrum → Removal of unnecessary (e.g., low or high frequencies → Inverse FT to get the output image as the ORIGINAL one BUT without the unwanted parts.
* **Context Filters** (filtering selected parts or regions of an image
Here may go all known popular filters
* **morphological** (conditonal filtering – conditions should be satisfied)

Methods of Image Transformation

3. Image Filtering – divided into:

* Spectral (**whole image filtration**).

Example of these methods is Fourier Transform:

Input image FT → Image Spectrum → removing the unwanted harmonics (e.g., low or high frequencies → Inverse Fourier Transform (to obtain an image with the required conditions).

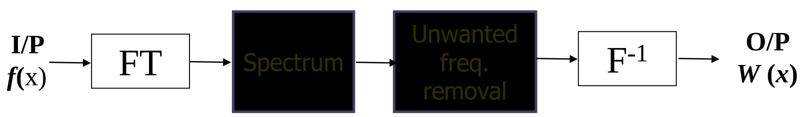
Methods of Image Transformation

3. Image Filtering – divided into:

* Spectral (**whole image filtration**).

Example of these methods is Fourier Transform:

Input image FT → Image Spectrum → removing the unwanted harmonics (e.g., low or high frequencies → Inverse Fourier Transform (to obtain an image with the required conditions).



Methods of Image Transformation (PROCESSING)

3. **Processing Methods by Image Filtering**

* **Spectral Filters** (Whole-Image Filtering). Fourier Transform is a good example.

Input Image → FT → Image Spectrum → Removal of unnecessary (e.g., low or high frequencies → Inverse FT to get the output image as the ORIGINAL one BUT without the unwanted parts.

* **Context Filters** (filtering selected parts or regions of an image.

Here may go all known popular filters

* **morphological** (conditonal filtering – conditions should be satisfied)

Methods of Image Transformation (PROCESSING)

3. **Processing Methods by Image Filtering**

* **Spectral Filters** (Whole-Image Filtering). Fourier Transform is a good example.

Input Image → FT → Image Spectrum → Removal of unnecessary (e.g., low or high frequencies → Inverse FT to get the output image as the ORIGINAL one BUT without the unwanted parts.

* **Context Filters** (filtering selected parts or regions of an image

Here may go all known popular filters

* **morphological** (conditonal filtering – conditions should be satisfied).

This is rather a LOGICAL Operation.

Filter Selection

The choice of the filter is often determined by the nature of the task and the type and behaviour of the DATA.

Hence,
- Noise,
- dynamic range,
- color accuracy,
- optical artifacts,
and many more details …

These factors will always affect the outcome of the filter functions in image processing.

Filter Classification is a difficult task ☺ !!

POPULAR METHODS:

There are two major catagories:

Linear
&
Nonlinear Filters

Linear Filters

Several principles define a linear system.

The first two are the basic definitions of linearity.

If a system is defined to have an input as $x[n] = ax[n_1] + bx[n_2]$, then the linear system response is $y[n] = ay[n_1] + by[n_2]$. This is known as the **superposition property**, and is fundamental to linear system design.

The second property is **shift invariance**. If $y[n]$ is the response to a linear, shift-invariant system with input $x[n]$, then $y[n-n_0]$ is the response to the system with input x[n-n0].

- In addition, two extra conditions are imposed, **causal and stable**. The **causal** condition is needed when considering systems in which future values are not known (for example, in video streaming). It is possible to consider a system that is not causal when looking at captured images with samples before and after the target location (for example, in a buffered version of an image frame). **Stability** is imposed to keep a filter's output from exceeding a finite limit, given an input that also does not exceed a finite limit. This is called the Bounded-Input Bounded-Output (BIBO) condition.

Linear Filters

Several principles define a linear system.

The first two are the basic definitions of linearity.

If a system is defined to have an input as $x[n] = ax[n_1] + bx[n_2]$, then the linear system response is $y[n] = ay[n_1] + by[n_2]$. This is known as the **superposition property**, and is fundamental to linear system design.

The second property is **shift invariance**. If $y[n]$ is the response to a linear, shift-invariant system with input $x[n]$, then $y[n-n_0]$ is the response to the system with input x[n-n0].

- In addition, two extra conditions are imposed, **causal and stable**. The **causal** condition is needed when considering systems in which future values are not known (for example, in video streaming). It is possible to consider a system that is not causal when looking at captured images with samples before and after the target location (for example, in a buffered version of an image frame). **Stability** is imposed to keep a filter's output from exceeding a finite limit, given an input that also does not exceed a finite limit. This is called the Bounded-Input Bounded-Output (BIBO) condition.

Linear Filters

Several principles define a linear system.
The first two are the basic definitions of linearity.
If a system is defined to have an input as $x[n] = ax[n_1] + bx[n_2]$, then the linear system response is $y[n] = ay[n_1] + by[n_2]$. This is known as the **superposition property**, and is fundamental to linear system design.

The **second property** is **shift invariance**. If y[n] is the response to a linear, shift-invariant system with input x[n], then $y[n-n_0]$ is the response to the system with input x[n-n0].

- In addition, two extra conditions are imposed, **causal and stable**. The **causal** condition is needed when considering systems in which future values are not known (for example, in video streaming). It is possible to consider a system that is not causal when looking at captured images with samples before and after the target location (for example, in a buffered version of an image frame). **Stability** is imposed to keep a filter's output from exceeding a finite limit, given an input that also does not exceed a finite limit. This is called the Bounded-Input Bounded-Output (BIBO) condition.

Linear Filters

Several principles define a linear system.
The first two are the basic definitions of linearity.
If a system is defined to have an input as $x[n] = ax[n_1] + bx[n_2]$, then the linear system response is $y[n] = ay[n_1] + by[n_2]$. This is known as the **superposition property**, and is fundamental to linear system design.

The **second property** is **shift invariance**. If y[n] is the response to a linear, shift-invariant system with input x[n], then $y[n-n_0]$ is the response to the system with input x[n-n0].

- In addition, two extra conditions are imposed, **causal and stable**. The **causal** condition is needed when considering systems in which future values are not known (for example, in video streaming). It is possible to consider a system that is not causal when looking at captured images with samples before and after the target location (for example, in a buffered version of an image frame). **Stability** is imposed to keep a filter's output from exceeding a finite limit, given an input that also does not exceed a finite limit. This is called the Bounded-Input Bounded-Output (BIBO) condition.

Nonlinear Filters

For nonlinear filters, the filter output or response of the filter does not obey the principles outlined earlier, **particularly scaling and shift invariance.**

Moreover, a nonlinear filter can produce results that vary in a non-intuitive manner.

The simplest nonlinear filter to consider is the **median or rank-order** filter. In the median filter, filter output depends on the ordering of input values, usually ranked from smallest to largest or vice versa. A filter support range with an odd number of values is used, making it easy to select the output.

Median filter is called NONLINEAR only because, its response is NOT LINEAR – it is dependent on the input values sorting.

REMARK

Each of these filter types can be parameterized to work one way under certain circumstances and another way under a different set of circumstances using adaptive filter rule generation.

Nonlinear Filters

For nonlinear filters, the filter output or response of the filter does not obey the principles outlined earlier, **particularly scaling and shift invariance.**

Moreover, a nonlinear filter can produce results that vary in a non-intuitive manner (**w nie wyczuwalny sposób**).

The simplest nonlinear filter to consider is the **rank-order (median)** filter. In the median filter, filter output depends on the ordering of input values, usually ranked from smallest to largest or vice versa. A filter support range with an **odd number** of values is used, making it easy to select the output.

Median filter is called NONLINEAR only because, its response is NOT LINEAR – it is dependent on the input values sorting.

REMARK

Each of these filter types can be parameterized to work one way under certain circumstances and another way under a different set of circumstances using adaptive filter rule generation.

Nonlinear Filters

For nonlinear filters, the filter output or response of the filter does not obey the principles outlined earlier, **particularly scaling and shift invariance.**

Moreover, a nonlinear filter can produce results that vary in a non-intuitive manner (**w nie wyczuwalny sposób**).

The simplest nonlinear filter to consider is the **rank-order (median)** filter. In the median filter, filter output depends on the ordering of input values, usually ranked from smallest to largest or vice versa. A filter support range with an **odd number** of values is used, making it easy to select the output.

Median filter is called NONLINEAR only because, its response is NOT LINEAR – it is dependent on the input values sorting.

REMARK

Each of these filter types can be parameterized to work one way under certain circumstances and another way under a different set of circumstances using adaptive filter rule generation.

Nonlinear Filters

For nonlinear filters, the filter output or response of the filter does not obey the principles outlined earlier, **particularly scaling and shift invariance.**

Moreover, a nonlinear filter can produce results that vary in a non-intuitive manner (**w nie wyczuwalny sposób**).

The simplest nonlinear filter to consider is the **rank-order (median)** filter. In the median filter, filter output depends on the ordering of input values, usually ranked from smallest to largest or vice versa. A filter support range with an **odd number** of values is used, making it easy to select the output.

Median filter is called NONLINEAR only because, its response is NOT LINEAR – it is dependent on the input values sorting.

REMARK

Each of these TWO filter types can be parameterized to **work one** way under certain circumstances **and another way** under a different set of circumstances **using adaptive filter** rule generation.

Contextual Methods of Image Filtering

Linear >>> to remove noise that has been introduced in an additive fashion

- Convolution
- LP Filters
- HP Filters
- Gauss Filter (Averaging)
- Edge Detection, derivative, steerable, Wiener

Nonlinear Filters >>> here the noise is a small number of pixels that are corrupted (randomly take a value of white or black – salt&pepper) due to, for example, a faulty transmission line.

- Median
- Extreme (max and min)
- Adaptive

Low-Pass Filters

Origin/source image

Low-Pass Filters

Origin/source image

In this filter, when all the elements are ones ‚1’
then the filter is of AVERAGING character

Low-Pass Filters

Origin/source image

In this filter, when all the elements are ones ‚1’
then the filter is of AVERAGING character,

as here →

$$w(i,j) = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

SPINAKER
Intensywne Międzynarodowe Programy Kształcenia (nabór 2021) - BPI/SPI/2021/1/00066/U/00001

Low-Pass Filters

Origin/source image

In this filter, when all the elements are ones ‚1' then the filter is of AVERAGING character,

as here →

$$w(i,j)=\begin{bmatrix}1&1&1\\1&1&1\\1&1&1\end{bmatrix}$$

Now, if the mid element is >1, (for example ‚2' as in the image here →) then the aim is to enhance and strenthen the central pixel.

$$w(i,j)=\begin{bmatrix}1&1&1\\1&2&1\\1&1&1\end{bmatrix}$$

Low-Pass Filters, more …

If the elements a_{12}, a_{21}, a_{23}, a_{32} są >1, and a_{22} isw their square, then we call it GAUSS Filter

Low-Pass Filters, more ...

If the elements a_{12}, a_{21}, a_{23}, a_{32} są >1, and a_{22} isw their square, then we call it GAUSS Filter

$$w(i,j)=\begin{bmatrix}1 & 2 & 1\\ 2 & 4 & 2\\ 1 & 2 & 1\end{bmatrix}$$

Low-Pass Filters, more ...

If the elements a_{12}, a_{21}, a_{23}, a_{32} są >1, and a_{22} isw their square, then we call it GAUSS Filter

$$w(i,j)=\begin{bmatrix}1 & 2 & 1\\ 2 & 4 & 2\\ 1 & 2 & 1\end{bmatrix}$$

However, if the central element is ZERO, then the aim is to enhance or strenthen the surrounding pixels.

Low-Pass Filters, more …

If the elements a_{12}, a_{21}, a_{23}, a_{32} są >1, and a_{22} isw their square, then we call it GAUSS Filter

$$w(i,j)=\begin{bmatrix}1 & 2 & 1\\ 2 & 4 & 2\\ 1 & 2 & 1\end{bmatrix}$$

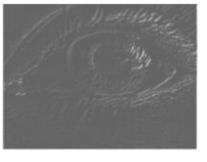
However, if the central element is ZERO, then the aim is to enhance or strenthen the surrounding pixels.

$$w(i,j)=\begin{bmatrix}1 & 1 & 1\\ 1 & 0 & 1\\ 1 & 1 & 1\end{bmatrix}$$

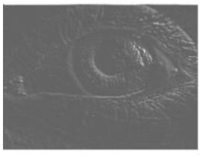
High-Pass Filters – Edge Detecting Filters

Roberts Gradient

$$w(i,j)=\begin{bmatrix}0 & 0 & 0\\ -1 & 0 & 0\\ 0 & 1 & 0\end{bmatrix}$$

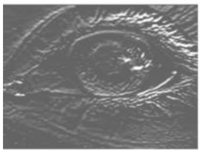


$$w(i,j)=\begin{bmatrix}0 & 0 & 0\\ 0 & 0 & -1\\ 0 & 1 & 0\end{bmatrix}$$



Prewitt mask/filter

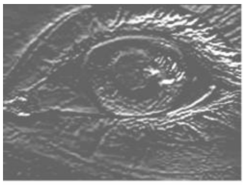
$$w(i,j)=\begin{bmatrix}-1 & -1 & -1\\ 0 & 0 & 0\\ 1 & 1 & 1\end{bmatrix}$$



High-Pass Filter

- Sobel mask

$$w(i,j)=\begin{bmatrix}-1 & -2 & -1\\ 0 & 0 & 0\\ 1 & 2 & 1\end{bmatrix}$$



- Directional mask – Corner detection

$$w(i,j)=\begin{bmatrix}1 & 1 & 1\\ 1 & -2 & -1\\ 1 & -1 & -1\end{bmatrix}$$

Convolution

- Mathematically:

$$g(x)=(f\times h)(x)=\int_{-\infty}^{+\infty} f(x-t)h(t)dt$$

Convolution

- Mathematically:

$$g(x) = (f \times h)(x) = \int_{-\infty}^{+\infty} f(x-t)h(t)dt$$

- Discrete case:

$$L'(m,n) = (w \times L)(m,n) = \sum_{i,j \in K} L(m-i, n-j)w(i,j)$$

w(*i*,*j*) – convolution mask

Convolution

- Mathematically:

$$g(x) = (f \times h)(x) = \int_{-\infty}^{+\infty} f(x-t)h(t)dt$$

- Discrete case:

$$L'(m,n) = (w \times L)(m,n) = \sum_{i,j \in K} L(m-i, n-j)w(i,j)$$

w(*i*,*j*) – convolution mask

In practice, normalization is used:

$$L'(m,n) = (w \times L)(m,n) = \frac{1}{\sum_{i,j \in MK} w(i,j)} \sum_{i,j \in K} L(m-i, n-j)w(i,j)$$

SPINAKER
Intensywne Międzynarodowe Programy Kształcenia (nabór 2021) - BPI/SPI/2021/1/00066/U/00001

How the convolution filters work !!

17	35	91
20	48	82
78	43	70

$$w(i,j)=\begin{bmatrix} w_1 & w_2 & w_3 \\ w_4 & w_5 & w_6 \\ w_7 & w_8 & w_9 \end{bmatrix}=\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

How the convolution filters work !!

17	35	91
20	48	82
78	43	70

$$w(i,j)=\begin{bmatrix} w_1 & w_2 & w_3 \\ w_4 & w_5 & w_6 \\ w_7 & w_8 & w_9 \end{bmatrix}=\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

- Input pixel - 48

How the convolution filters work !!

17	35	91
20	48	82
78	43	70

$$w(i,j)=\begin{bmatrix} w_1 & w_2 & w_3 \\ w_4 & w_5 & w_6 \\ w_7 & w_8 & w_9 \end{bmatrix}=\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

- Input pixel - 48
- Output value:

$$L_x = 17\cdot w_1 + 35\cdot w_2 + 91\cdot w_3 + 20\cdot w_4 + 48\cdot w_5 + 82\cdot w_6 + 78\cdot w_7 + \\ + 43\cdot w_8 + 70\cdot w_9 = 808$$

How the convolution filters work !!

17	35	91
20	48	82
78	43	70

$$w(i,j)=\begin{bmatrix} w_1 & w_2 & w_3 \\ w_4 & w_5 & w_6 \\ w_7 & w_8 & w_9 \end{bmatrix}=\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

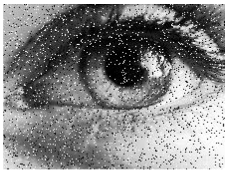
- Input pixel - 48
- Output value:

$$L_x = 17\cdot w_1 + 35\cdot w_2 + 91\cdot w_3 + 20\cdot w_4 + 48\cdot w_5 + 82\cdot w_6 + 78\cdot w_7 + \\ + 43\cdot w_8 + 70\cdot w_9 = 808$$

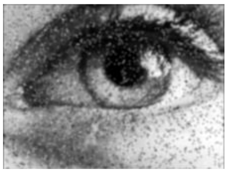
- After normalization:

$$L'_x = \frac{L_x}{\sum_{i,j} w(i,j)} = \frac{808}{16} = 50.5 \approx 51$$

Gauss Filter



Noise „Pepper and salt”



Gauss Filter (a_{22}>1) does not solve the problem of this kind of noise

Nonlinear Filters

Here the noise is when a small number of pixels are corrupted (randomly, they take a value of white or black – salt & pepper) due to, for example, a faulty transmission line.

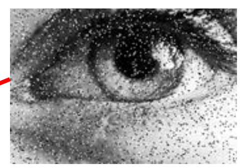
Nonlinear Filters!

Why NOT Linear?

Noise „Pepper and salt”

Median Filter

Nonlinear Filters

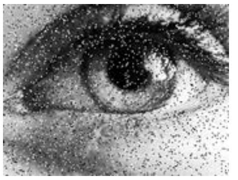


Averaging filter

For comparison, use a linear filter)

Gauss Filter – additive noise

No difference !!



Nonlinear Filters

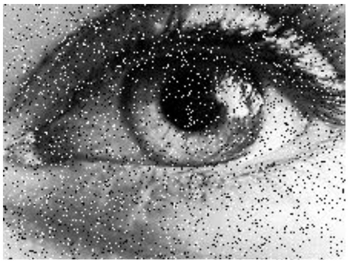
Actually, trying to average out the noise in this fashion is equivalent to asking for the average salary of a group of 8 workers and their main boss - a millionaire, say.

The boss's salary will skew (**deform**) the average salary.

And so does each noise pixel when its value is **so disparate/dissimilar** from its neighbours.

In such cases the mean is replaced with the MEDIAN-the centre pixel of each 3x3 neighborhood is replaced with the median of the nine pixel values in that neighborhood.

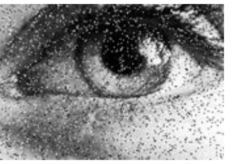
Median Filtr



Median Filter

Resulting image
WITHOUT noise

Summary



Averaging filter

For comparison, use a linear filter)

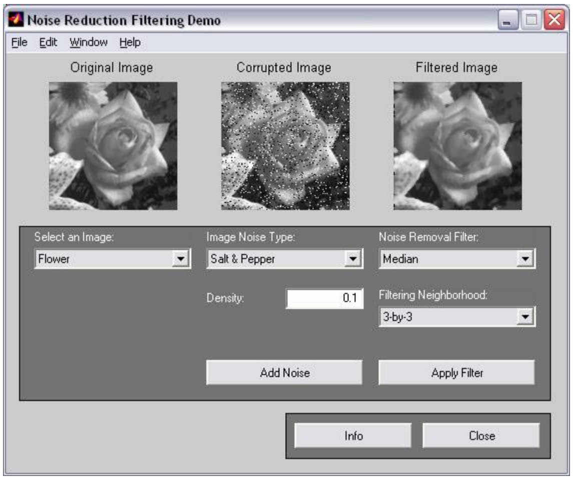
Gauss Filter – additive noise

Median filter

No difference,
Meaningless use

No noise

Take other examples…..



Nonlinear filters…. More

Applying a 15x15 median filter will result in losing many internal details, but retaining the boundary contours (edges), what is called →

POSTERIZATION.

POSTERIZATION

Median and Average

Applying a 5x5 averaging filter >>> to compare with the 15x15 median filter – see the edges !!

Median and Average

Applying a 15x15 median filter will result in losing many internal details, but retaining the boundary contours (edges), what is called POSTERIZATION. **15x15 Fussy**

This effect could never be achieved with an averaging filter which would indiscriminitely smooth over all image structures.

Min, Max and Median – their work:

91	35	70
20	78	43
17	82	48

Min, Max and Median

91	35	70
20	78	43
17	82	48

Sort the pixel values
[17,20,35,43,48,70,78,82,91]

Min, Max and Median

91	35	70
20	78	43
17	82	48

Sort the pixel values
[17,20,35,43,48,70,78,82,91]
Replace the central value with:

- - 17 for Min filter

Min, Max and Median

91	35	70
20	78	43
17	82	48

Sort the pixel values
[17,20,35,43,48,70,78,82,91]
Replace the central value with:

- - 17 for Min filter
- - 48 for Median filter

Min, Max and Median

91	35	70
20	78	43
17	82	48

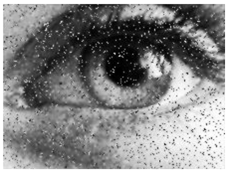
Sort the pixel values
[17,20,35,43,48,70,78,82,91]
Replace the central value with:

- \- 17 for Min filter
- \- 48 for Median filter
- \- 91 for Max filter

More examples of applications

Min

Max



Adaptive

Spectral Filters

- Fourier Transform

Discrete Fourier Transform for digital images

$$F(i,k)=\beta_L \sum_{m=0}^{M-1}\sum_{n=0}^{N-1} L(m,n) e^{\frac{-j2\Pi mi}{M}} e^{\frac{-j2\Pi nk}{N}},$$

$$i=0,\dots,M-1; k=0,\dots,N-1$$

Spectral Filters

- Fourier Transform

Discrete Fourier Transform for digital images

$$F(i,k)=\beta_L \sum_{m=0}^{M-1}\sum_{n=0}^{N-1} L(m,n) e^{\frac{-j2\Pi mi}{M}} e^{\frac{-j2\Pi nk}{N}},$$

$$i=0,\dots,M-1; k=0,\dots,N-1$$

Inverse FT:

$$L(m,n)=\beta_F \sum_{i=0}^{M-1}\sum_{k=0}^{N-1} F(i,k) e^{\frac{j2\Pi mi}{M}} e^{\frac{j2\Pi nk}{N}},$$

$$m=0,..,M-1; n=0,\dots,N-1$$

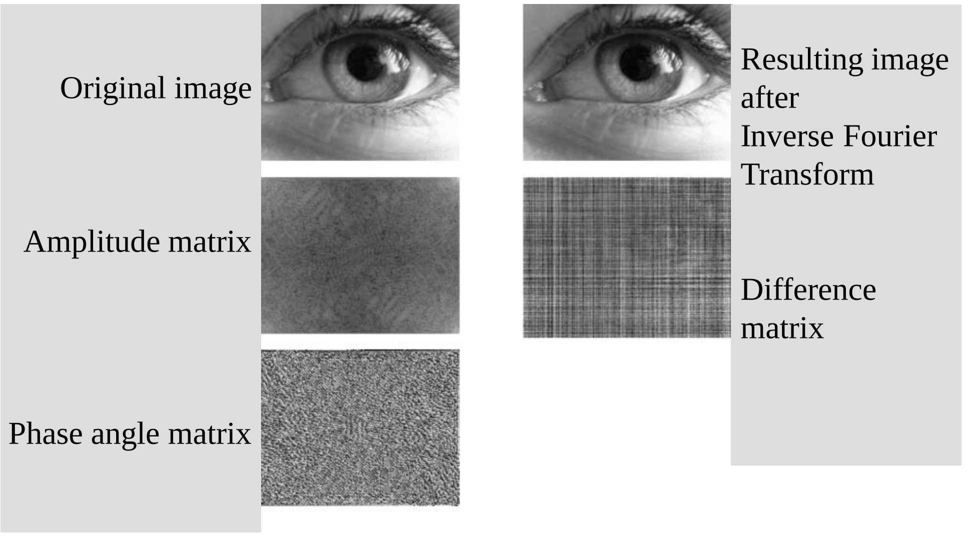
$$\beta_L \cdot \beta_F = \frac{1}{M \cdot N}$$

Filtering by FT for digital images

The procedure is similar to the analogue application:

1. - Two dimensional specrum is calculated,
2. - modified,
3. - and the resulting sepctrum is reconstructed
4. by inverse FT

Fourier Transform



Morphologial Methods

- Erosion and Dilation
- Opening and Closing
- Thinning

Erosion

Using different structural templates

- 3x3

- 5x5

- 7x7

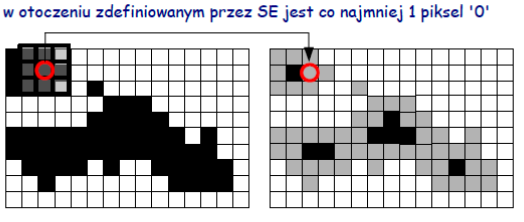
Dilation

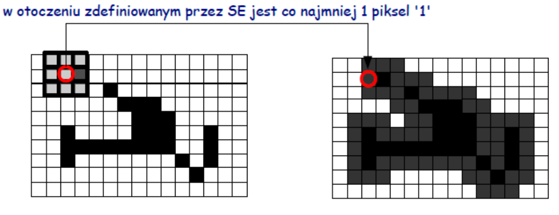
- Using different structural templates
- 3x3

- 5x5

- 7x7

Erosion and Dilation – their work:





Open and Close

- Open

  (Dilation of Erosion)

- Closing

  (Erosion of Dilation)

Thank you

LECTURE 2
Introduction to Biometrics

Human Recognition and Authentication

SPINAKER
Intensywne Międzynarodowe Programy Kształcenia (nabór 2021) - BPI/SPI/2021/1/00066/U/00001

Human Authentication

Human Authentication

- Traditional means of automatic authentication:

Human Authentication

- Traditional means of automatic authentication are:
  - **Possession-based** (credit card, smart card, passport, ID card, …)

Human Authentication

- Traditional means of automatic authentication are:
  - **Possession-based** (credit card, smart card, passport, ID card, …)

    - *Uses "something that you have"*

Human Authentication

- Traditional means of automatic authentication are:
  - **Possession-based** (credit card, smart card, passport, ID card, …)
    - *Uses "something that you have"*
  - **Knowledge-based** (password, PIN)

Human Authentication

- Traditional means of automatic authentication are:
  - **Possession-based** (credit card, smart card, passport, ID card, …)
    - *Uses "something that you have"*
  - **Knowledge-based** (password, PIN)

    - *Uses "something that you know"*

Biometrics Authentication

- Traditional means of automatic authentication are:
  - **Possession-based** (credit card, smart card, passport, ID card, …)
    - *Uses "something that you have"*
  - **Knowledge-based** (password, PIN)
    - *Uses "something that you know"*
  - **Biometrics-based** (biometric identifier)

Biometrics Authentication

- Traditional means of automatic authentication are:
  - **Possession-based** (credit card, smart card, passport, ID card, …)
    - *Uses "something that you have"*
  - **Knowledge-based** (password, PIN)
    - *Uses "something that you know"*
  - **Biometrics-based** (biometric identifier)
    - *Uses something that relies on "what you are"*

Human Authentication

- **Possession-based** (credit card, smart card, passport, ID card, …)
  - *Uses "something we have"*
- **Knowledge-based** (password, PIN)
  - *Uses "something that we know"*
- **Biometrics-based** (biometric identifier)
  - *Uses something that relies on "what we are"*

Problems with traditional authentication

- Tokens may be lost, stolen or forgotten
- Passwords or PINs may be forgotten
- or easily guessed by the imposters
- of people seem to write their PIN on their ATM card
-
- The *traditional approaches* are **unable to differentiate between an authorized person and an impostor** (the person pretending to be somebody he/she is not)

This is a good evidence that we still need a safer system for our identification or authentication

Hence, what is the alternative solution to avoid imposting?

The ***Biometric features*** seems to be the solution.

Biometric systems mainly consist of four parts:

1. Scanning Hardware (camera, microphone, text scanner, X-Ray and Magnetic Resonance Machines, …)
- to scan the human anatomy being under test.

2. Analog-to-Digital Converter - in order to gather the information and convert it into a digital form.

3. An appropriate Software to manupolate digital data for the DSP or DGP

4. Database to save the classified image and compare it with the already stored image in the database for person identification and/or verification.

Biometric systems mainly consist of four parts:

1. Scanning Hardware (camera, microphone, text scanner, X-Ray and Magnetic Resonance Machines, …)
- to scan the human anatomy being under test.

2. Analog-to-Digital Converter - in order to gather the information and convert it into a digital form.

3. An appropriate Software to manupolate digital data for the DSP or DGP

4. Database to save the classified image and compare it with the already stored image in the database for person identification and/or verification.

Biometric systems mainly consist of four parts:

1. Scanning Hardware (camera, microphone, text scanner, X-Ray and Magnetic Resonance Machines, …)
- to scan the human anatomy being under test.

2. Analog-to-Digital Converter - in order to gather the information and convert it into a digital form.

3. An appropriate Software to manupolate digital data for the DSP or DGP

4. Database to save the classified image and compare it with the already stored image in the database for person identification and/or verification.

Biometric systems mainly consist of four parts:

1. Scanning Hardware (camera, microphone, text scanner, X-Ray and Magnetic Resonance Machines, …)
- to scan the human anatomy being under test.

2. Analog-to-Digital Converter - in order to gather the information and convert it into a digital form.

3. An appropriate Software to manupolate digital data for the DSP or DGP

4. Database to save the classified image and compare it with the already stored image in the database for person identification and/or verification.

Biometric systems mainly consist of four parts:

1. Scanning Hardware (camera, microphone, text scanner, X-Ray and Magnetic Resonance Machines, …)
- to scan the human anatomy being under test.

2. Analog-to-Digital Converter - in order to gather the information and convert it into a digital form.

3. An appropriate Software to manupolate digital data for the DSP or DGP

4. Database to save the classified image and compare it with the already stored image in the database for person identification and/or verification.

Biometrics Categories:

1. Physiological (*Cognitive Biometrics*)

Physiological biometrics measure the *distinct traits that people have*, usually (but not always or entirely) dictated by their genetics. They are based on measurements and data derived from direct measurement of a part of the human body.

Examples

Fingerprints	Iris image (coloured part of the eye)
Face	Odor
Retina	Ear
Vascular pattern	Lips
Hand geometry	DNA

2. Behavioral (*Behaviometrics*)

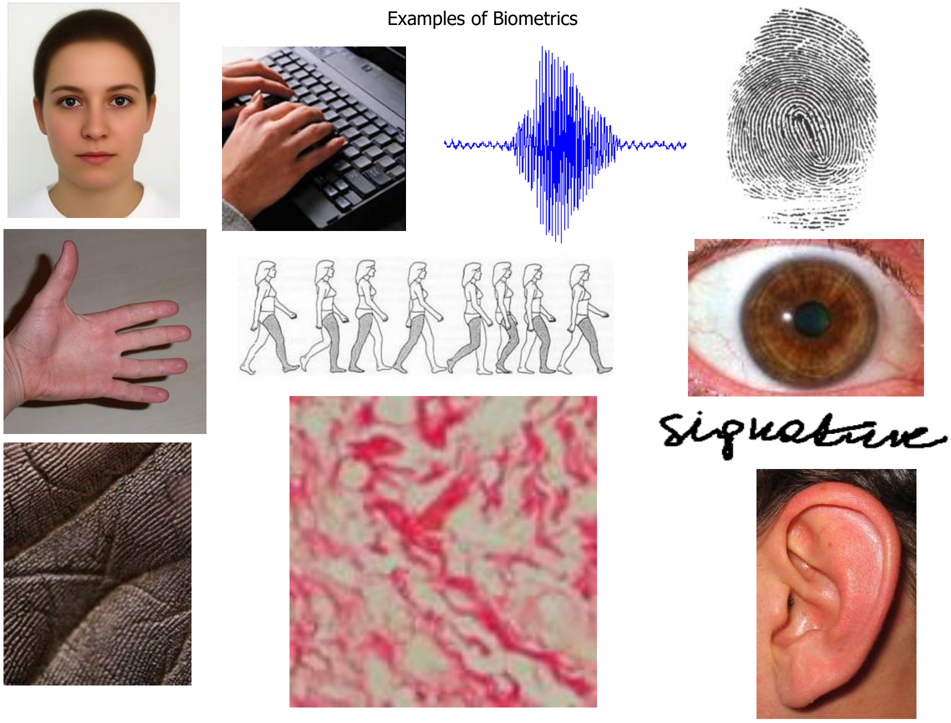
Behavioral biometrics measure the *distinct actions that humans take*, which are generally very hard to copy from one person to another. They measure characteristics of the human body indirectly.

Examples:

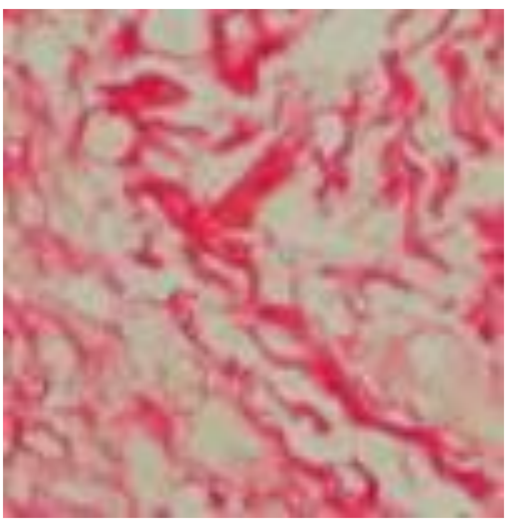
Speaker Recognition
Signature
Keystroke
Mouse dynamics
Gait (way of walking)

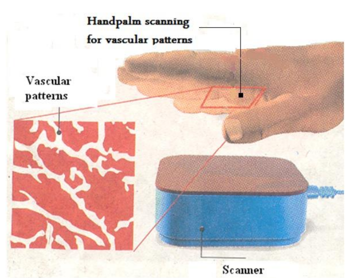
Affective Systems
Emotion and Intention Detection
>>> Kansei Engg (Japan)

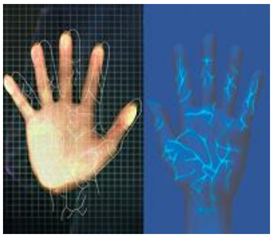
Examples of Biometrics

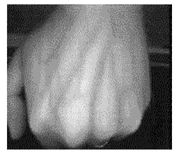


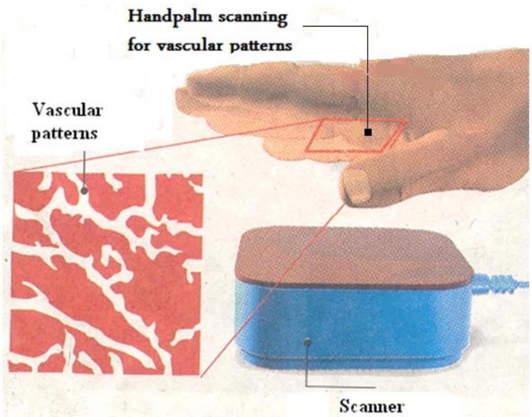
Vascular Pattern











Problems with Biometrics

Biometrics cannot be recreated,
This is a fact, but from the other side, they can be stolen.

This is called …. SPOOFING

… Spoofing !!!

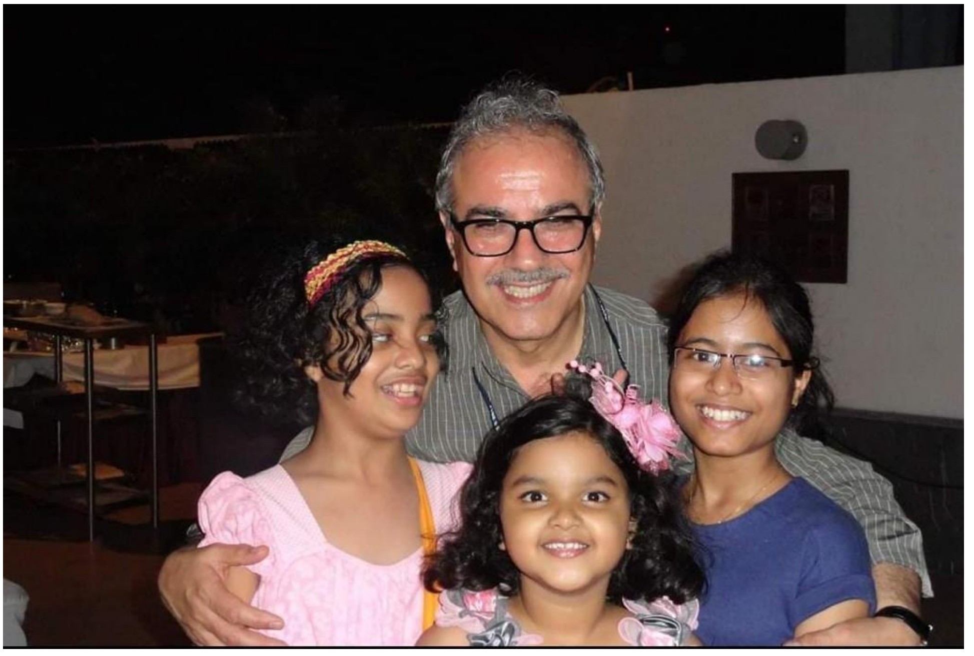
Most Biometric features can be stolen:
Shaking hands, Face or Iris image by a good camera, …
All can be taken easily and shown to ATM's !

K. Saeed (EiC), "Spoofing and Anti-Spoofing in Biometrics", IJBM-International Journal of Biometrics, vol. 1, no. 2, Inderscience Publishers, UK, 2008.

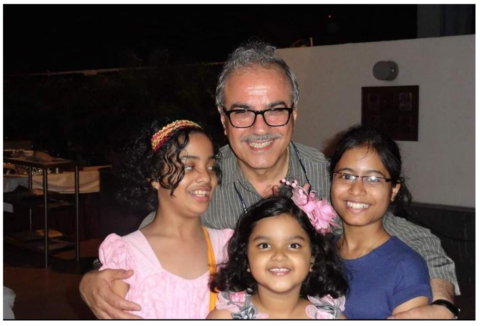
Other problems with Biometrics

>>>Aging<<<

>>>2014<<<



>>>2022<<<



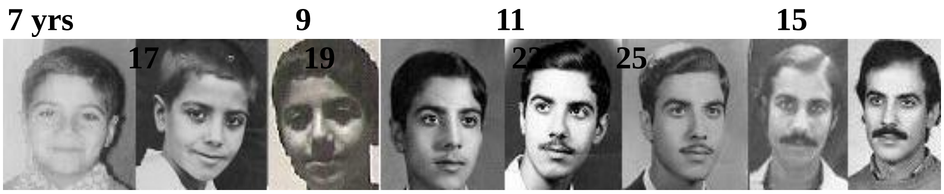
Example

Who is this man?

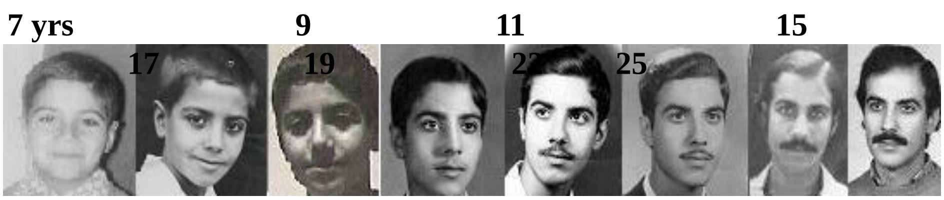
Maybe you would know this man?

If not, then I am sure you know this man!!

And this man, do you know him?



Some people never change ☺

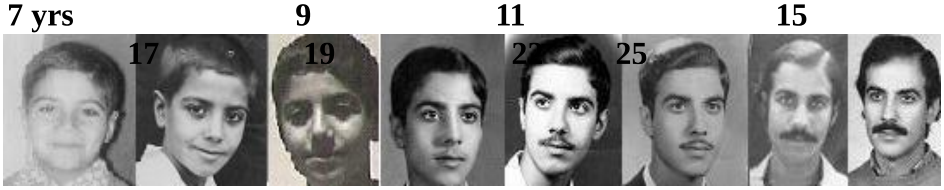


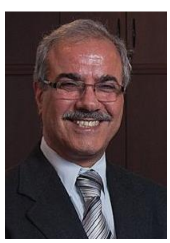
Some people never change ☺

7 yrs 9 11 15

Some people never change ☺

7 yrs 9 11 15

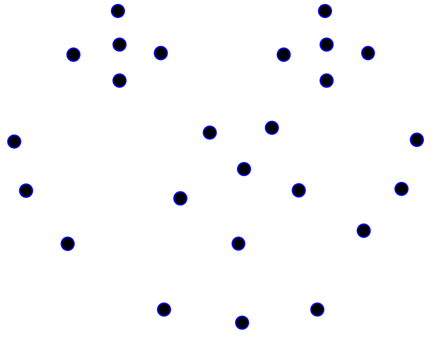




Mathematical Aspects

Challenges for engineers

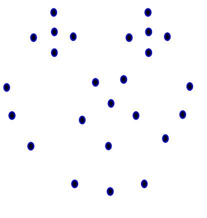
Biometrics Image Description:

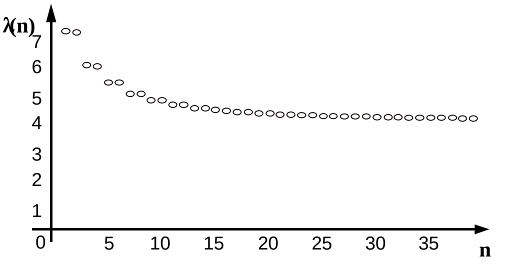


This collection of points can be the feature points of any image in BIOMETRICS, a face, for example, can be transformed to a nonincreasing series, by Toeplitz matrices

The mathematical model of **Toeplitz-Caratheodory*** character, simply deals with such points for ***image description***.

According to this theory the collection of points is transformed into a stable distribution, simply a REGULAR SEQENCE of points:





In all of the biometric images, the characteristic points would form in a way or another Toeplitz matrices:

$$C = \begin{bmatrix} c_0 & c_{-1} & c_{-2} & \cdots & c_{-n+1} \\ c_1 & c_0 & c_{-1} & \ddots & \vdots \\ c_2 & c_1 & c_0 & \ddots & c_{-2} \\ \vdots & \ddots & \ddots & \ddots & c_{-1} \\ c_{n-1} & \cdots & c_2 & c_1 & c_0 \end{bmatrix}$$

with

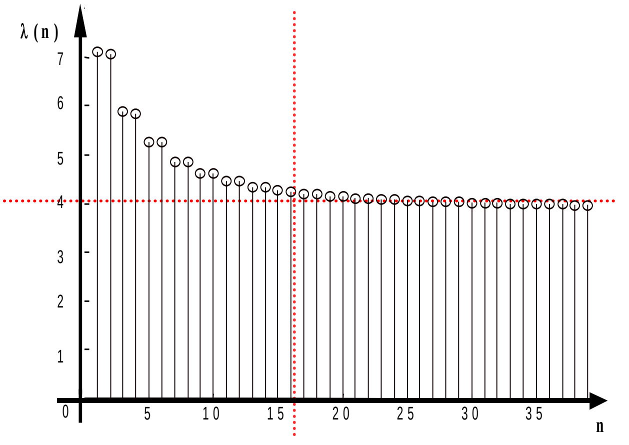
$$C_{i,j} = C_{i-1,\,j-1}$$

The minimal eigenvalues of these forms have a special unique behavior which was used successfully in Image Analysis for Object Recognition.

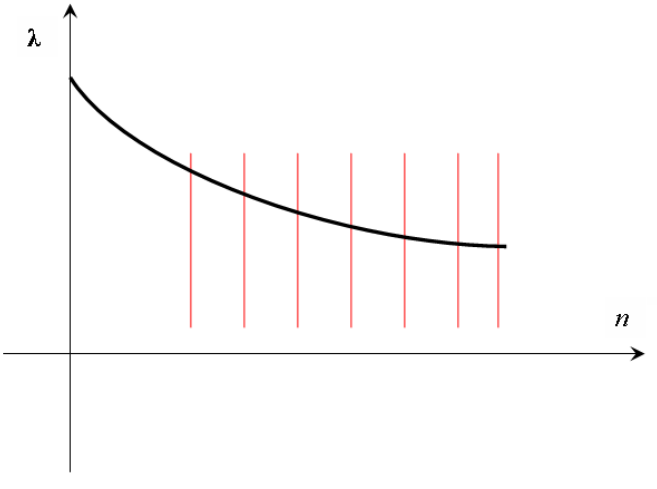
$$\lambda_0 \geq \ldots \geq \lambda_i \geq \ldots \geq \lambda_n$$

This fact was proved both theoretically and experimentally.

Töeplitz minimal eigenvalues behavior

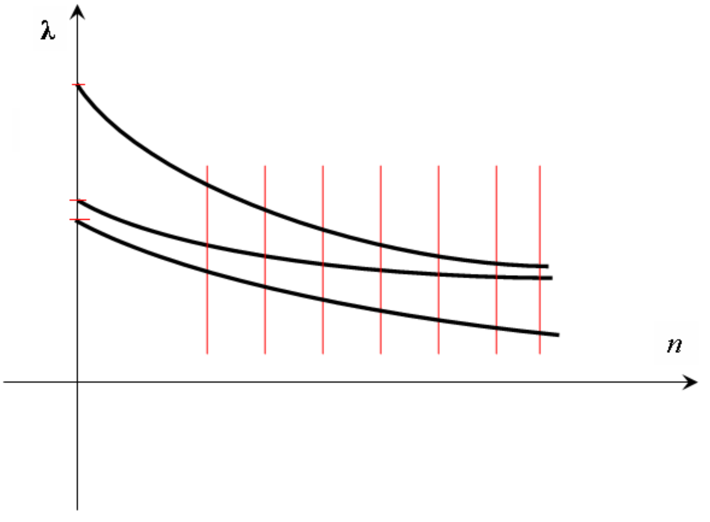


- Data reduction, optimising minimal requirements for image annotation.
The main advantage of the minimal eigenvalues comprises the
RELATION BETWEEN VECTOR COMPONENTS
and NOT THE GEOMETRIC NATURE of the features.



Each part or interval CARRIES THE SAME monotonic characteristics.

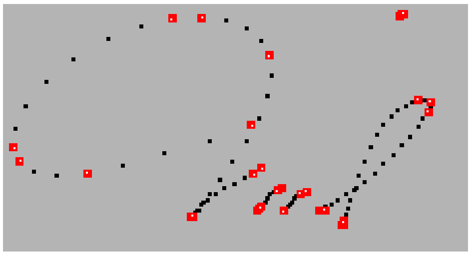
The same information is obtained from all the lines representing the same image, but <u>differing from one image to another in a class.</u>

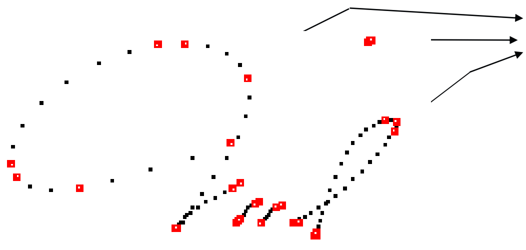


This has its applications in many science aspects (not only Biometrics).

How to construct Töeplitz matrices from plot-image characteristics ?

Recalling for those who have not dealt with TM





x_0	…	x_i	…	x_{n-1}
y_0	…	y_i	…	y_{n-1}

x_0	…	x_i	…	x_{n-1}
y_0	…	y_i	…	y_{n-1}

$$f(s)=\frac{P(s)}{Q(s)}=\frac{x_0+x_1s+x_2s^2+x_3s^3+x_4s^4+\dots}{y_0+y_1s+y_2s^2+y_3s^3+y_4s^4+\dots}$$

$$T(s)=c_0+c_1s+c_2s^2+\dots+c_is^i+\dots$$

Töeplitz Matrices and their minimal eigenvalues

$$C=\begin{bmatrix} c_0 & c_1 & c_2 & \cdots & c_{n-1} \\ c_1 & c_0 & c_1 & \ddots & \vdots \\ c_2 & c_1 & c_0 & \ddots & c_2 \\ \vdots & \ddots & \ddots & \ddots & c_1 \\ c_{n-1} & \cdots & c_2 & c_1 & c_0 \end{bmatrix} \longrightarrow \begin{matrix} \lambda_0 \\ \lambda_1 \\ \vdots \\ \lambda_i \end{matrix}$$

$$T(p)=c_0+c_1p+c_2p^2+\ldots+c_ip^i+\ldots$$

Circular Toeplitz Matrices and their minimal eigenvalues

$$C=\begin{bmatrix} c_0 & c_{n-1} & c_{n-2} & \cdots & c_1 \\ c_1 & c_0 & c_{n-1} & \ddots & \vdots \\ c_2 & c_1 & c_0 & \ddots & c_{n-2} \\ \vdots & \ddots & \ddots & \ddots & c_{n-1} \\ c_{n-1} & c_{n-2} & \cdots & c_1 & c_0 \end{bmatrix} \longrightarrow \begin{matrix} \lambda_0 \\ \lambda_1 \\ \vdots \\ \lambda_i \end{matrix}$$

$$T(p)=c_0+c_1p+c_2p^2+\ldots+c_ip^i+\ldots$$

This is done in such a way to find the minimal eigenvalue of each submatrix:

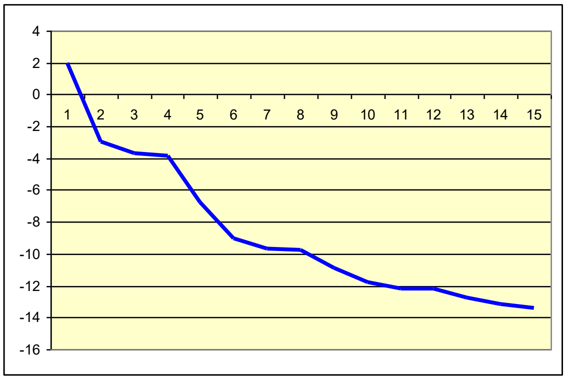
$$C = \begin{bmatrix} c_0 & c_1 & c_2 & \cdots & c_{n-1} \\ c_1 & c_0 & c_1 & \ddots & \vdots \\ c_2 & c_1 & c_0 & \ddots & c_2 \\ \vdots & \ddots & \ddots & \ddots & c_1 \\ c_{n-1} & \cdots & c_2 & c_1 & c_0 \end{bmatrix} \qquad \begin{matrix} \lambda_0 \\ \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_n \end{matrix}$$

$$T(p) = c_0 + c_1 p + c_2 p^2 + \ldots + c_i p^i + \ldots$$

As shown above, the sequence of the minimal eigenvalues of Toeplitz matrices decreases monotonically with the increase of matrix size to reach a definite value as a limit at m, where $m \geq n$,

and n is the number of characteristic points.

$$\lambda_0 \geq \ldots \geq \lambda_i \geq \ldots \geq \lambda_n$$



$$\lambda_0$$
$$\lambda_1$$
$$\vdots$$
$$\lambda_n$$

$$FV = [\lambda_0, \lambda_1, ..., \lambda_n]$$

$$FV = [\lambda_0, \lambda_1, ..., \lambda_n]$$

It could be shown that there exists
a perfect relation between these minimal eigenvalues
furnishing an easy-to-implement way of
object image description.

Thank you

Other ways of data feeding to Töeplitz forms:

1) $$c_i = \sqrt{x_i^2 + y_i^2}$$

2) $$c_0 = |r_0| - |r_i|, \quad c_1 = |r_1| - |r_2|, \ldots, c_n = |r_n|$$

3) $$c_i = r_i \, e^{j\varphi_i}$$ where, $$\begin{cases} |r_i| = \sqrt{x_i^2 + y_i^2} \\ \varphi_i = \tan^{-1} \dfrac{y_i}{x_i} \end{cases}$$

In a given class-group, each element has its own series of minimal eigenvalues forming its FEATURE VECTOR – the *image code*

$$FV = [\lambda_0, \lambda_1, \ldots, \lambda_n]$$

TM and Artificial Intelligence

AI plays an essential role in the classification stage:

TM eigenvalues give effective classification results with Neural Networks*.

---

* Saeed K., Tabędzki M.: Intelligent Feature Extract System for Cursive-Script Recognition. *Proc.* IEEE-WSTST'05, Muroran, Japan, 2005. Advances in Soft Computing, Springer-Verlag Berlin Heidelberg, Germany, 2005, 192-201

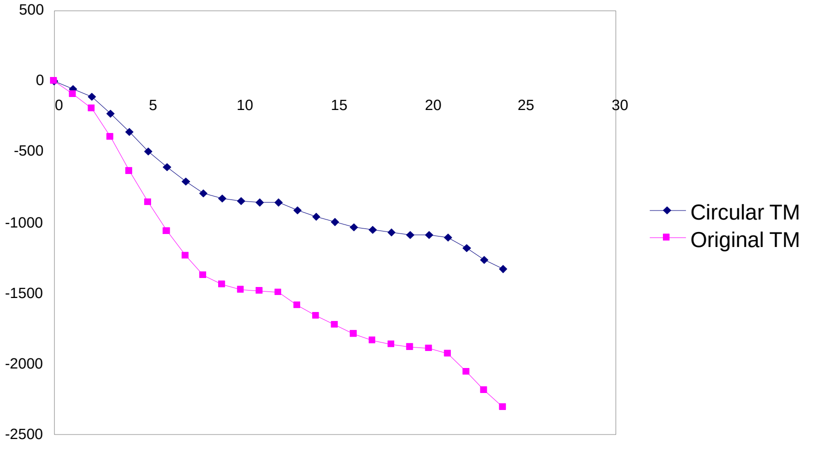
What is new?

*The use of **Circulant Toeplitz Matrices** is introduced.*

Circular Toeplitz Matrices furnish minimal eigenvalues series of the same character, but **converging** to a limit much **faster than the traditional ones**

$$C = \begin{bmatrix} c_0 & c_{n-1} & c_{n-2} & \cdots & c_1 \\ c_1 & c_0 & c_{n-1} & \ddots & \vdots \\ c_2 & c_1 & c_0 & \ddots & c_{n-2} \\ \vdots & \ddots & \ddots & \ddots & c_{n-1} \\ c_{n-1} & c_{n-2} & \cdots & c_1 & c_0 \end{bmatrix} \longrightarrow \begin{matrix} \lambda_0 \\ \lambda_1 \\ \vdots \\ \lambda_i \end{matrix}$$

The following slide shows the newest results of the research on CIRCULANT Toeplitz Matrices minimal eigenvalues



$$FV = [\lambda_0, \lambda_1, \ldots, \lambda_n]$$

Additional important property is that the eigenvalues of a circulant matrix can be readily calculated by Fast Fourier Transform and that they are related to Discrete Cosine Transform.

Additional important property is that the eigenvalues of a circulant matrix can be readily calculated by Fast Fourier Transform and that they are related to Discrete Cosine Transform.

This opens wide possibilities for research seekers !

If no time, I can skip some of the following slides ☹

If ok, then ….

I would, however, show the mathematics beyond these modles – *the descriptor models*:

Mathematics of Descriptors

The origin of the mathematical model goes back to the days of Caratheodory, Toeplitz and Schur between 1911 and 1917. Then Brune (1931) worked on similar classes of functions

Carathéodory function C(p)

Definitions:

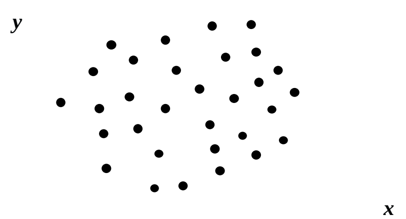
The function $C(p)$ is Carathéodory function if:

1. C(p) is analytic for
2. $Re\ C(p) > 0$ for

Given real numbers

$$a_i \quad i = 1, 2, \ldots, n$$

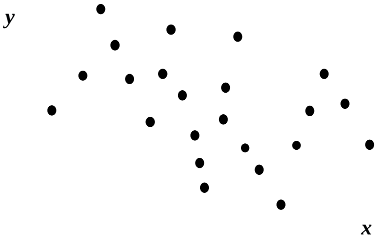
distributed in the x-y plane randomly

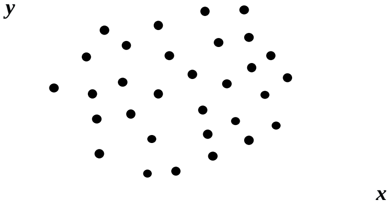


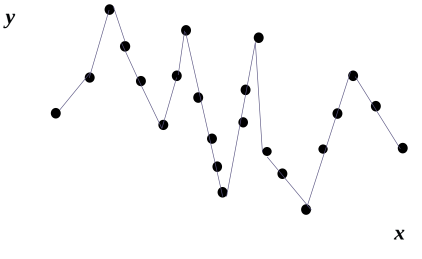
Given real numbers

$$a_i \;\; i = 1, 2, \ldots, n$$

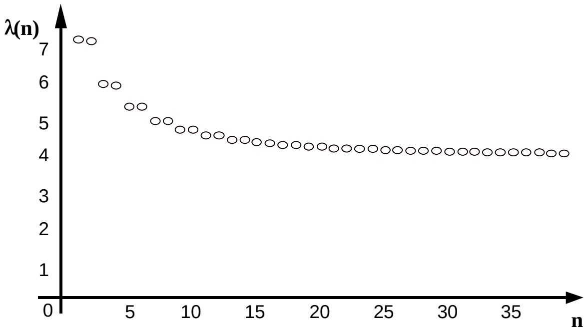
distributed in the *x*-*y* plane randomly
or according to a given criterion.







These points can be redistributed in a way to furnish the sequence illustrated below.



This is done according to the transformation:

$$(x_i, y_i) \Rightarrow \lambda_i$$

and the (x_i, y_i) can be the geometric points of an object image.

The monotonically decreasing quantity representation is useful for Image Representation to take the place of traditional methods of image escription for classification within the Biometric Systems of people Authentication.

It has proved its use in Image description, particularly as an IMAGE CODE in Biometric Image Description.

Application of the Theory to Signal & Image Processing and *Biometrics*

has already been shown above