Contents

- Course Structure
- Basic concepts of Computer Vision
  - Pixels
  - Colour vs. Black&White
  - Noise and linear filtering
  - Image enhancement
- Some practical examples
  - My first programme with images
  - Image denoising
  - Image enhancement
- Basic commands matlab
Course Structure

This course is an introductory course for students with very few practical knowledge on computer vision.

Resources available for students through BCAM website

- Course material: slides, exercises and examples

What do you need for this course?

- The most important thing is your **self-motivation**!
- **Programming**: examples will be done in Matlab, and also, if needed, I will explain how to do it in OpenCV (C++)
- For those C++ or Phyton programmers, I suggest to use a proper code editor (eclipse, netbeans, …)
- Some examples of useful tools (not needed) but good to know
  - Versioning: git, SVN
  - Automatic documentation generation: Doxygen
  - Project management: Redmine
Course Structure: Contents

- Each class will last 2 hours and a half
  - 50 min Theory (9:30-10:20)
  - 10 min break
  - 50 min Theory (10:30-11:20)
  - 10 min break
  - 30 min practice (11:30-12:00)

- Each slide will contain a blue box with the command in MATLAB, in case you want to try it during the lessons.

- At the end of the slides, you will have a table with the commands used during the lessons, for a quick guide.
Some good books

  - This is the bible in this domain

- Learning OpenCV, Computer Vision in C++ with the OpenCV Library, by Adrian Kaehler, Gary Bradski. Publisher: O’Reilly Media (2013)
  - This is a really good book if you want to start learning with OpenCV.
  - OpenCV offers also dedicated tutorials online, check its website
What is Computer Vision?

- **Computer vision** is a scientific discipline that aims to extract information from images through artificial systems to produce numerical or symbolic information, e.g. in the forms of decisions.

- Acquisition, processing, analyzing, or understanding techniques are usually involved.

- The image data can take many forms, photography, video frame sequences, views from multiple cameras, or multi-dimensional data from a medical scanner, etc.
Optical illusions

Your vision can be cheated

There are plenty of examples on this, and big studies

This only shows the complexity of the field
**Image Processing:** set of mathematical operations by using any form of signal processing to **process the image:**

- Emphasize
- Segment
- Resize
- Smooth, etc)

**Computer graphics:** set of techniques to manually make images from physical models, environments and lighting (e.g. animated movies).

**Computer Vision:** high-level image processing where the system is intended to figure out the contents of an image or a video (e.g. object recognition)

**Machine Vision:** is the application of image processing within industrial environments, dealing with issues like: dust, fat, poor illumination conditions, real-time constraints, physical restrictions within the machines, etc.
Computer Vision World

Computer Science
Engineering
Physics
Maths
Biology
Medicine
Some Examples
Computer Vision System: Let's start!

- **Image acquisition** → capturing the image or video with a physical device (video, medical device, camera, mobile, etc)

- **Image processing (low level processing)** → segmentation, tracking, enhancement, feature detection.

- **Image understanding** → interpretation of those regions/objects to figure out what is actually happening in the image (e.g. object recognition, spatial relationship, etc)

- **Image recognition** → face recognition, biometrics, optical character recognition
Some Basic Concepts before start
• How does a computer “see” an image?

- **A 2D image** \((I)\) is seen as a set of pixels \(I = f(x,y)\)

- **A gray-scale image** in matlab and in openCV is considered as a matrix.

- The typical pixel value range is from \([0 \, 255]\), being 0 black and 255 white.

- If we are in **3D**, then instead of pixels we have **voxels**!

Images extracted from:
http://what-is-this.net/fr/definition/tomography; http://oivdoc95.vsg3d.com/content/111-what-volumeviz
In this course, we are going to focus only in grayscale images.

Any colour image can be easily transformed into grayscale.

2D images + RGB (Red-Green-Blue). colormap → usually green channel is selected (less noise)

But, take care since they are not actually equal!

\texttt{rgb2gray}
Neighbourhood

- 4-neighbours
- 4-diagonal neighbours
- 8-neighbours

Adjacency for pixels p and q

- 4-adjacent if they have the same value and q is in the set \( N_4(p) \).
- 8-adjacent if they have the same value and q is in the set \( N_8(p) \).
- m-adjacent if they have the same value and q is in the set \( N_4(p) \) OR q is in the set \( N_D(p) \) AND the set \( N_4(p) \cup N_4(q) \) is empty.

Connectivity: Two pixels p and q are connected if they are neighbours and they gray levels satisfy a specific condition (e.g. in binary images, they must have the same value). So, two pixels are m-connected if a m-path can be drawn between them.
Pixel Distances

- **Pixel distances**
  - **Euclidean distance:** straight line distance
  - **City-block (Manhattan) distance:** assumes that in going from one pixel to the other it is only possible to travel directly along pixel grid lines. Diagonal moves are not allowed
  - **Chessboard distance:** assumes that you can make moves on the pixel grid as if you were a King making moves in chess
  - The last ones are faster to compute than Euclidean distance
  - $D_m$ distance between two pixels is defined as the shortest $m$-path between the points.
First steps with your images

Any computer vision challenge starts in an image

Image quality is far away of your ideal

You will need to do some prior work to adjust the quality of the image to your needs

Some **challenges** you usually face:

- Poor/too bright illumination conditions
- Camera without right position
- Imaging distortion
- Dust, pollution, …
- Image artifacts
Image Preprocessing

- The **image** usually does not meet your system requirements.
- Computer Vision Engineers will need to process the image to remove, enhance or highlight all the elements they need.
- We refer to **“image preprocessing”** to all those basic image processing operations performed over the image to make it suitable for our system.
- Image preprocessing usually refers to the following aspects:
  - Geometric operations
  - Noise removal
  - Image enhancement
- This step shall provide a clean/homogeneous/normalized image to the other steps of our system, regardless of the input device or the conditions when the image was taken. Thus, minimizing errors or issues generated during the acquisition stage.
Image Processing

- Are these two images similar?
  - Geometric transformations

- I cannot see the image very well
  - Noise removal
  - Linear Filtering
  - Smoothing techniques

- The quality of the image is not the expected one. What can I do?
  - Image Enhancement

- I need to identify the object's edges
  - Thresholding

- What can I do with the edges?
  - Image Morphology
Noise
**Variety of images: Noise the common factor**

- **Image noise** is random variation of brightness or color information in images, and is usually an aspect of electronic noise that adds spurious and extraneous information. It can be produced by the sensor and circuitry of a scanner or digital camera.

- The **signal to noise ratio (SNR)** is a useful and universal way of comparing the relative amounts of signal and noise for any electronic system; high ratios will have very little visible noise whereas the opposite is true for low ratios.

\[
SNR = \frac{P_{signal}}{P_{noise}}
\]

- **Types of noise**
  - Gaussian noise
  - Salt-and-pepper noise
  - Shot noise or Poisson noise
  - Film grain
Gaussian Noise

- Gaussian noise is statistical noise which values are \textbf{Gaussian}-distributed.
- The probability density function $p$ of a Gaussian random variable $z$ is given by:

$$p_G(z) = \frac{1}{\sigma \sqrt{2\pi}} e^{-\frac{(z-\mu)^2}{2\sigma^2}}$$

where $z$ represents the grey level.

- \textbf{White Gaussian noise}: A special case where the values at any pair of times are identically distributed and statistically independent (and hence uncorrelated).

- Principal sources arise during acquisition e.g. sensor noise caused by poor illumination and/or high temperature, and/or transmission e.g. electronic circuit noise.

- \textbf{It can be reduced using a spatial filter}: mean (convolution) filtering, median filtering and Gaussian smoothing.

![Image of Gaussian noise reduction](image)
Salt & Pepper

- Salt-and-pepper noise is a form of noise that presents as sparsely occurring white and black pixels.

- This noise can, e.g., be used to model defects in the CCD or in the transmission of the image.

- An effective noise reduction method for this type of noise is a
  - median filter
  - morphological filter
Noise: Other

- **Shot noise or Poisson noise** is a type of electronic noise which can be modeled by a Poisson process.
  - Many images, such as those from radiography, contain noise that satisfies a Poisson distribution.
  - The magnitude of Poisson noise varies across the image, as it depends on the image intensity. This makes removing such noise very difficult.

- **Film grain** or granularity is the random optical texture of processed photographic film due to the presence of small particles of a metallic silver, or dye clouds, developed from silver halide that have received enough photons.
Image denoising
Image Averaging

- **Image averaging** works on the assumption that the noise in your image is truly random.

- Random fluctuations above and below actual image data will gradually even out as one averages more and more images.

- This technique is used in applications as astronomy.

Astronomical Images
http://fidabaluch.blogspot.com.es/

A series of ST-7 Dark Frame Residuals
http://www.frazmtn.com/~bwallis/drk_avg.htm

<table>
<thead>
<tr>
<th>n=5</th>
<th>n=10</th>
<th>n=20</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1.png" alt="Image" /></td>
<td><img src="image2.png" alt="Image" /></td>
<td><img src="image3.png" alt="Image" /></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>n=50</th>
<th>n=100</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image4.png" alt="Image" /></td>
<td><img src="image5.png" alt="Image" /></td>
</tr>
</tbody>
</table>
Smoothing filters compute a weighted average of the pixels under the mask.

- Drawback: smoothing reduces fine image detail

Smoothing filters use a mask or kernel.

- The convolution of the image with the kernel, obtains an image where the noise is averaged with the actual intensity of the image.

There are several types: mean, median, etc.

- It is possible and very practical to design a specific kernel
  - Tip: Always and odd number!

Example of 3x3 Mean filter

http://nullprogram.com/blog/2008/02/22/
Mean filter

- Mean filtering is a simple, intuitive and easy to implement method of smoothing images, which is often used to reduce noise in images.

- **Idea:** to replace each pixel value in an image with the mean (‘average’) value of its neighbors, including itself.

- It removes pixel values which are **unrepresentative** of their surroundings.

- Often a \(3 \times 3\) square kernel is used, although larger kernels (e.g. \(5 \times 5\) squares) can be used for more severe smoothing.

- A small kernel can be applied more than once to produce a similar but not identical effect as a single pass with a large kernel.
Median Filter (I)

- The median filter is normally used to reduce noise in an image, like the mean filter, but preserving image edges.

- **Idea:** to replace each pixel value in an image with the median value of its neighbors, including itself.

- The median is calculated by first sorting all the pixel values from the surrounding neighborhood into numerical order and then replacing the pixel being considered with the middle pixel value.

- A 3×3 square neighborhood is preferred since larger neighborhoods will produce more severe smoothing.
Median filter is adequate to remove salt & pepper noise without affecting too much to the image details.
Gaussian smooth kernel

- The effect of Gaussian smoothing is to blur an image, in a similar fashion to the mean filter, but with a different kernel that represents the shape of a Gaussian (‘bell-shaped’) hump.

- The degree of smoothing is determined by the standard deviation of the Gaussian.
Image Enhancement
The aim of image enhancement is to provide `better' input for other automated image processing techniques.

Image enhancement techniques can be divided into two broad categories:

- **Spatial domain methods**, which operate directly on pixels
  - **Histogram modeling techniques** provide a sophisticated method for modifying the dynamic range and contrast of an image by altering that image such that its intensity histogram has a desired shape.
  - Logarithm & Exponential operators to enhance specifically low/high pixel values

- **Frequency domain methods**, which operate on the Fourier transform of an image (out of the scope in this course).
Contrast stretching

- Contrast stretching (often called **normalization**) attempts to improve the contrast in an image by `stretching` the range of intensity values it contains to span a desired range of values.

- It can only apply a **linear scaling function** to the image pixel values.

- The upper and lower pixel value limits over which the image is to be normalized must be specified.

```
imadjust(I, stretchlim(I))
```
**Histogram Equalization**

- **Histogram equalization** employs a **monotonic, non-linear** mapping which re-assigns the intensity values of pixels in the input image such that the output image contains a uniform distribution of intensities (i.e. a flat histogram).

- **Advantages:**
  - Effective in detail enhancement
  - Correction of non-linear effects
Histogram based comparison

Original

Contrast Stretched

Histogram Equalization
Logarithm & Exponential Operator (I)

- **Logarithmic Operator** is used to change the dynamic range of an image, enhancing **low intensity pixel values**.
  - The logarithmic operator is a simple point processor where the mapping function is a logarithmic curve.
  - Each pixel value is replaced with its logarithm.
  - The effect is that **low intensity pixel values are enhanced**.

- **Exponential operator** is used to change the dynamic range of an image, enhancing **high intensity pixel values**.
  - The exponential operator is a point process where the mapping function is an exponential curve.
  - Each pixel intensity value in the output image is equal to a basis value raised to the value of the corresponding pixel value in the input image.
  - To enhance the visibility of a normal photograph, values just above 1 are suitable.
Logarithm & Exponential Operator (II)
Geometrical Transformations
Image Registration (I)

- Image registration is the process of transforming different sets of data into one coordinate system. Also known as image alignment.
- Types: intensity-based or feature-based
**Intensity-based methods** register entire images or sub-images.

- Centers of registered sub images are treated as corresponding feature points.

**Feature-based methods** establish a correspondence between a number of especially distinct points in images.

→ Find the geometrical transformation to match the target image to the reference images.

**Transformations**

- **Linear transformations (global)**
  - Affine transformations

- **Non rigid transformations** (allows locally warping of the target image)
  - Surface splines
Geometric transformations are widely used in **image registration** and for the **removal of geometric distortion**.

- Resize, rotate, and crop images; perform geometric transformation of multidimensional arrays.
- **Spatial Referencing**: geometric transformation of the image coordinate system.
- Typical operations: translation, similarity, euclidean, affine, projective ...
Affine Transformation

- **Definition:** Is any transformation that preserves
  - collinearity
  - ratios of distances
- Transforming and fusing the images to a large, flat coordinate system is desirable to eliminate distortion
- It is a composition of rotations, translations, magnifications and shears
- It is typically used to correct for geometric distortions or deformations that occur with non-ideal camera angles.
  - satellite imagery uses them to correct for wide angle lens distortion, panorama stitching
  - image registration to align images

\[
\begin{align*}
  u &= ax+by+c \\
  v &= dx+ey+f
\end{align*}
\]
Image scaling is the process of resizing a digital image.

Scaling is a non-trivial process that involves a trade-off between efficiency, smoothness and sharpness.

This is technically impossible!

\[
T = \begin{bmatrix}
s1 & 0 & 0 \\
0 & s2 & 0 \\
0 & 0 & 1
\end{bmatrix}
\]

http://goo.gl/qz0WnP

imresize
In Euclidean geometry, a translation is a function that moves every point a constant distance in a specified direction.

If \( \mathbf{v} \) is a fixed vector, then the translation \( T_{\mathbf{v}} \) will work as

\[
T_{\mathbf{v}}(\mathbf{p}) = \mathbf{p} + \mathbf{v}
\]

\[
T = \begin{pmatrix}
1 & 0 & 0 \\
0 & 1 & 0 \\
\mathbf{t}_x & \mathbf{t}_y & 1
\end{pmatrix}
\]
In linear algebra, a **rotation matrix** is a matrix that is used to perform a rotation in Euclidean space.

Rotation matrix:

\[
T = \begin{bmatrix}
\cos \theta & \sin \theta & 0 \\
-s\sin \theta & \cos \theta & 0 \\
0 & 0 & 1 \\
\end{bmatrix}
\]
A **shear mapping** is a linear map that displaces each point in fixed direction, by an amount proportional to its signed distance from a line that is parallel to that direction.

**Horizontal shear**

\[
T = \begin{bmatrix}
1 & 0 & 0 \\
0.5 & 1 & 0 \\
0 & 0 & 1
\end{bmatrix}
\]

This mapping takes any point with coordinates \((x,y)\) to the point \((x + 0.5y,y)\).

Note that points on opposite sides of the reference line are displaced in opposite directions.
Non-rigid transformations: MRI alignment

- TODO
- ADNI
- DTI image selection
- Neuroimaging Processing
- Selected Raw DTI
- Normalized DTI

- Non-rigid registration
- Warp-fields → deformation field

Original Brain + MN152 Template = Registered Brain

imwrap
Digital sampling of any signal, whether sound, digital photographs, or other, can result in apparent signals at frequencies well below anything present in the original.

- **moire patterns** arise in image warping & texture mapping
- **jaggies** arise in rendering

The aliasing that occurs as a result of size reduction normally appears as **stair-step patterns** (especially in high-contrast images), or as **moire** (ripple-effect) patterns in the output image.

- Tip: Use antialiasing in matlab to avoid this effect
- If you want to remove them manually, usually you will need to go to the frequency domain
Summary

- **Pixel**, neighbourhood, distances and connectivity
- Colour vs **GrayScale**
- Image Preprocessing
  - **Image enhancement**
    - Contrast stretching
    - Histogram equalization
    - Logarithm and exponential operators
  - **Noise**
    - Gaussian noise and Salt & pepper
  - **Image denoising techniques**
    - Averaging
    - Mean filter
    - Median filter
    - Gaussian smooth kernels
- **Geometric transformations**
  - Affine Transformation
  - Scale
  - Translation
  - Rotation
  - Shear
  - Image registration
  - Aliasing
<table>
<thead>
<tr>
<th>Action</th>
<th>Command Matlab</th>
</tr>
</thead>
<tbody>
<tr>
<td>Read image</td>
<td><code>I = imread('cameraman.tif');</code></td>
</tr>
<tr>
<td>Save image</td>
<td><code>imsave</code></td>
</tr>
<tr>
<td>Show image</td>
<td><code>figure(1); imshow(I);</code></td>
</tr>
<tr>
<td>Show image (2)</td>
<td><code>figure(2); imagesc(I);</code></td>
</tr>
<tr>
<td>See several images in a single figure</td>
<td><code>subplot(nmp);</code></td>
</tr>
<tr>
<td>Change colormap</td>
<td><code>Colormap('gray');</code></td>
</tr>
<tr>
<td>help</td>
<td><code>help command</code></td>
</tr>
<tr>
<td></td>
<td><code>doc command</code></td>
</tr>
<tr>
<td>Close all images and figures</td>
<td><code>close all</code></td>
</tr>
<tr>
<td>Clear all previous data</td>
<td><code>clear all</code></td>
</tr>
</tbody>
</table>
# Matlab: Commands used in the examples

<table>
<thead>
<tr>
<th>Slide no.</th>
<th>Matlab</th>
<th>OpenCv tips &amp; equivalences</th>
</tr>
</thead>
<tbody>
<tr>
<td>14</td>
<td>rgb2gray</td>
<td>cvtColor()</td>
</tr>
<tr>
<td>22,23</td>
<td>imnoise</td>
<td>-</td>
</tr>
<tr>
<td>27-28,31</td>
<td>fspecial</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>imfilter</td>
<td>-</td>
</tr>
<tr>
<td>29</td>
<td>medfilt2</td>
<td>medianBlur()</td>
</tr>
<tr>
<td>34</td>
<td>imadjust</td>
<td>-</td>
</tr>
<tr>
<td>35</td>
<td>histeq</td>
<td>equalizeHist()</td>
</tr>
<tr>
<td>36</td>
<td>imhist</td>
<td>calcHist()</td>
</tr>
<tr>
<td>37</td>
<td>log2</td>
<td>log()</td>
</tr>
<tr>
<td>37</td>
<td>exp</td>
<td>exp()</td>
</tr>
<tr>
<td>43,47</td>
<td>affine2d</td>
<td>WarpAffine</td>
</tr>
<tr>
<td></td>
<td>imtransform</td>
<td></td>
</tr>
<tr>
<td>45</td>
<td>imtranslate</td>
<td>WarpAffine</td>
</tr>
<tr>
<td>44</td>
<td>imresize</td>
<td>WarpAffine</td>
</tr>
<tr>
<td>46</td>
<td>imrotate</td>
<td>getRotationMatrix2D()</td>
</tr>
<tr>
<td>48</td>
<td>imwrap</td>
<td>warpAffine</td>
</tr>
</tbody>
</table>
Exercise 1. Download any colour image from internet (e.g. png format)

- Open the image and see independently the 3 channels in a single figure. Do you see any difference?
- Save the 3 independent channels of the image with different names.

Exercise 2. Image Denoising

- Choose one the saved images from exercise 1.
- Add to these image salt&pepper noise, and a gaussian noise. Save them separately.
- Now, try to remove the noise from both noisy images using all the techniques (mean, median, gaussian smooth – play a little with different kernel sizes).
- Visualize the original images, noisy images and smoothed ones in a single figure
  - Which one works better? What happen with the image? Do you see it equally?
- Save the result images
Exercise 3. Image enhancement

- Use any grayscale image from the previous exercises.
- Apply contrast stretching and histogram equalization, and visualize the histograms.
- Apply log and exponential operators.
- Create a figure showing original, contrast stretching, histogram equalized, log and exponential. Which one is better? Is this always true or does it depends on your system?

Exercise 4. Image registration

- Use any grayscale image from the previous exercises.
- Apply a translation, a rotation, a shear and a scale transformation and see each of the images in a single figure.
- Apply an affine transformation to the image, and visualize it together with the original one. Are you able to see how they are similar?
Any image must be treated before to be used in our computer vision system

- Images are noisy → We don't want noise!
- Exterior conditions or input devices can provide a image with a quality that does not meet our needs
  - Improve contrast, light, etc..
- Devices and people never take the same picture in the same conditions, position, etc
  - We must ensure that our images are as much similar as we can