Artificial Vision

Main aspects of artificial images

All images have two levels of approximation:

- Discretization
- Quantization

Discretization

The images, in theory, are constituted by infinity^2 points, with an infinite number of rows or columns, each one constituted by infinite points. The transmission time for the calculation system of this kind of images it's infinite. It's impossible to acquire this kind of images, so they need to have a finite number of points:

$$n_{points} = \frac{Area\ image}{Area\ pixel}$$

The single point is calling pixel that means picture element: its single value represents the average between all the infinity^2 points that constitutes that area. This kind of operation is called discretization of the image, that can be represented by a grid of pixels.

Quantization

Quantization is a technique that translates an infinite range of data into discrete one.

Quantization allows us to simplify the information in each pixel, that can be:

1. Black and white image => this kind of image doesn't have any kind of information about the colour, but only the information of the luminosity of each point. The luminosity is represented by using a scale of grey from white to black. The light is quantized in photons: lighter is a pixel, more photons have hit that pixel. A point on the image can assume a colour between black and white, so infinite gradations of colours: we should consider only a certain number of levels that represent all the infinite levels they are composed. They are represented by only one level that is the average of all the levels. So, we have to decide how many levels we want to consider, so the quantization level (medical image maybe have to be quantized with higher level of quantization).

2. Colour image => the fundamental colour that compose an image are blue, red and green, that belong to the RGB (red, green, blue) model. Each physical colour is split into 3 components of red, green and blue and it can assume a value between 0 and 255. The convention that we apply is:
   - ⇨ Red: 255, 0,0
   - ⇨ Green: 0, 255, 0
   - ⇨ Blue: 0, 0, 255
   - ⇨ White: 255, 255, 255
   - ⇨ Black: 0, 0, 0

The sensors

The human eye

Our eyes are composed, in the back area, by a matrix of sensors, cones and rods, that are spread in the retina and that are responsible for the conversion of light into electrical energy, that is then conveyed to the neurons of the brain. Cones are responsible for the colour vision, so they are red, green or blue depending on the light that they can acquire; they respond mainly to bright light and they are responsible for the sharpness of vision. Rods are responsible for monochromatic vision, they respond mainly to poor light and they are more than cones.

The pupil is a clear part of the eye that allows the light to entry and reach some sensors; it's sensible to the intensity of the light, so it allows to regulate the intensity of light by reducing o increasing the opening of our eyes.

The scene that we see is reflected on the back area of our eyes.

The sensitivity graph represents, for each cone and rod, the curve of sensitivity respect to the wavelength of the light.

Matrix sensors

Each sensor of the matrix has a behaviour similar to the cones of human eye: they transform light energy into electrical energy and then in power, then quantized and converted in bit. The sensor of a camera can acquire the light in two different ways:

- **Global shutter** => the content information of each pixel is read at the same time during the so called shutter time or exposed time;
- **Rolling shutter** => the content information of each pixel is read at different time: the sensor activates one row of pixel at a time to acquire the light.

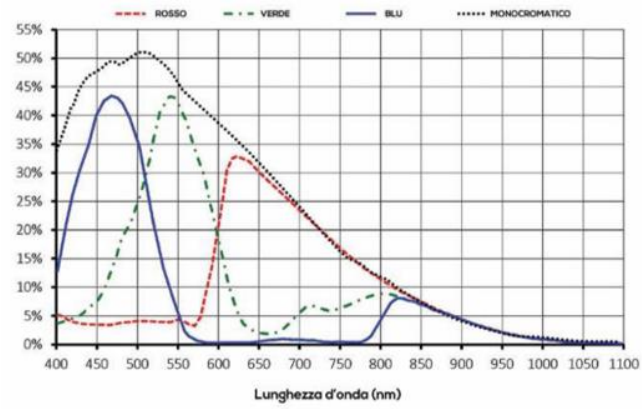
So, the **shutter time** or **exposed time** is the slot of time in which the pixel acquires the light.

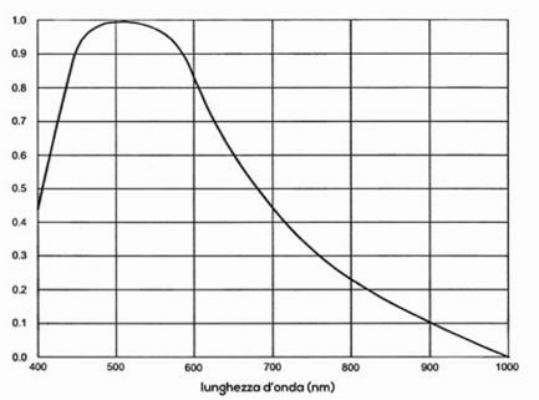
The lens of a camera has the goal to convey the best quantity of light on the sensor, so it's important that the diameter of lens is larger than the diagonal of the sensor in order to completely cover its area.

So, for this reason, we can have sensors with different diagonal sizes and different number of pixels, that change the resolution of the image they can product.

Two sensor, one smaller than the other one, with the same number of pixels have the same resolution, but the dimension of each pixel is different between the larger sensor and the smaller sensor. The big one has a bigger pixel that can receive more quantity of energy at the same exposure time, instead of the smaller sensor that has a smaller pixel that, at the same exposure time (of the big one), can acquire less quantity of light. So, to obtain the same energy in the smaller one, we have to increase the exposure time.

Each sensor, green, red, blue or monochromatic, has a different sensitivity on wavelengths like the cones and rods of human eye, so even if they are hit by the same energy, we obtain different quantized numbers. It's important to note that the sensitivity chart of monochromatic sensors is always better than the other ones: they can reach any kind of light.

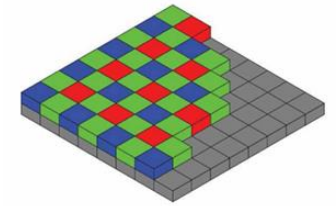




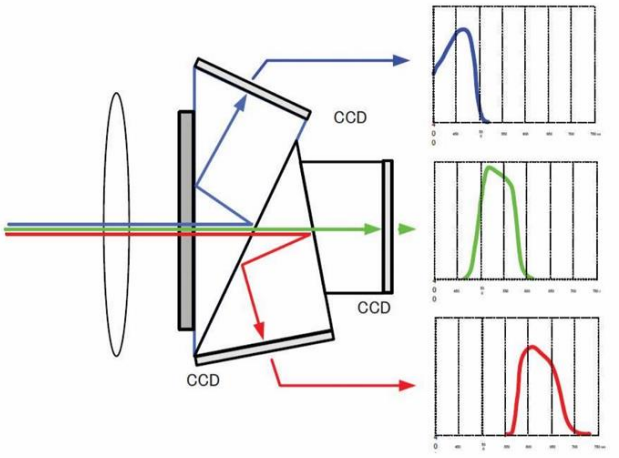
So, in order to obtain a better performance of the sensor, the sensor is often composed by a matrix of monochromatic pixels and above each of them there are different filters, red, green or blue that block all the other components except for their respective wavelength. This technique is called coated and permit to increase the sensitivity of each pixel.

In this way, we can obtain only one RGB component from each pixel, but we need 3 colour components: the other two components are calculated by **interpolation**. We consider the other pixels around the pixel that we are considering, and we compute the average of them.

One of the most widely used patterns today is Bayer's, which considers the greater sensitivity of human vision to green compared to red and blue.



Another different method, that doesn't need to use interpolation but it's more expensive, is to use a **prism camera**: it uses 3 different matrix sensors, one for each colour, and in front of each of them there is a prism that conveys only the specific wavelength of that sensor, while the other components are reflected. The sensors are monochromatic, and they are covered, in all their area, by a specific filter red, green or blue. In this way we obtain 3 images of the same scene on each sensor.



Multi-sensor devices: they are useful to avoid problems of overexposed or underexposed images, by setting the exposure time to a correct value for that scene. One example is HDR, High Dynamic Range, that we find in our telephone, and that acquires different images of the same scene with different shutter times and then makes the interpolation between them to obtain the final image (the problem of this technique is that the images are acquired in different times).

Line scan camera

This kind of camera can acquire a single line of the scene so, moving the camera or the object, we can obtain all the lines of a long object or a long scene with the resolution that we desire.

The motion of the camera has to be synchronized with the motion of the object, so we have to consider:

- The speed of the object or of the camera.
- The highest frequency of the camera: how many frames per second can my camera acquire? I must respect the minimum speed to acquire the scene, so if we want to go slower we can't.
  So, if we have to acquire a certain number of line (1 pixel = 1 line), the minimum frame rate can be calculated as:

$$\frac{line}{sec} = \frac{speed\ m/s}{\frac{1}{resolution\ pixel/m}}$$

- The resolution.

The band of the camera, that depends on the protocol of the camera, is the quantity of data that I can acquire (byte/sec). Cameras that have the same value of band belong to the same family.

The camera can capture images in 2 ways:

- Free Run: the camera captures at a certain frequency given by a clock.
- External Trigger: the camera acquires when it receives a signal from an encoder on the GPIO port.

If we use two line scan cameras, the 1st will work in free run and will send a message on its output port to the other camera (as a trigger generator) to start its acquisition (external trigger). In this way you don't need an external sync device.

If I want a resolution of 1pix/mm, I have to move the object 1mm and take a picture every time I move. If the object is moving at a speed of 1m/s, I need to acquire 1000 lines/s.

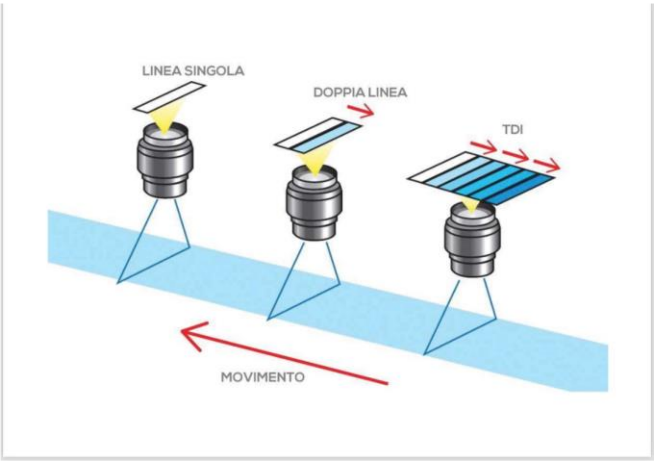
If I have an object 2m long and I want a resolution of 5pix/mm, I need a sensor of 5pix/mm.

If an object has a speed of 5m/s and I want a resolution of 5pix/mm, I need to capture:

$$line/sec = 5mm/s \times 10^3 \cdot 5pix/mm = 25000\frac{line}{sec}$$

If I have two cameras with the following parameters:
A —> 20,000 lps; 4000 pix
B —> 40,000 lps; 2000 pix; 80 MB/s
I have to use the B because I need to acquire 25,000 lps. The two cameras have the same bandwidth because the product between frame rate and resolution is the same.

If we have to acquire in T time 1000 lines with a resolution of 1 pixel/mm, we have to maintain a high speed and also an high line/sec; but T/1000 is greater than the shutter time, so the image will be black. This problem (high line rate => low exposition time) can be resolved by TDI (Time Delay Integration), in which the shutter time is "multiplied" for each line that we have to acquire.



The TDI moves the charges accumulated from one line to the next one, in synchronization with the motion of the object, to capture multiple shots of the same line and add them, thus obtaining a line with amplified sensitivity.

To obtain more sensitivity, we can also use the **binning** technique that consists in grouping adjacent pixels in only 1 pixel: instead of taking care of the energy that only one pixel acquires, we can consider

the sum of the voltages of each pixel of the group and then quantize it. This method can be applied also in matrix cameras by grouping square areas of pixel.

Acquisition by line scan camera

I need some regularity in acquiring lines: they have to be equally spaced in frequency. But obviously I cannot be sure that the time slice is always the same and the speed of the camera or of the object is perfectly constant during the time. So, I have to synchronize the motion of the object with the acquisition of the camera.

The **encoders** are external hardware that can send a signal when the object is moving of a certain distance (according to the resolution of the encoder, i.e. the minimum distance that the encoder can detect).

Now suppose that we are moving the camera: if the motion is not synchronized with the acquisition, the slope of the image could be distorted, so we need to synchronize the motion and the acquisition. When I consider the resolution of this set up, I have to consider the speed of the object and the line rate (line per second) of the camera, but also the number of pixels in the sensor (is usually composed by one line). A camera with a higher frequency can reach the same resolution in less time respect to another camera with a small value of line rate (this also represents a constraint for the speed of the object).

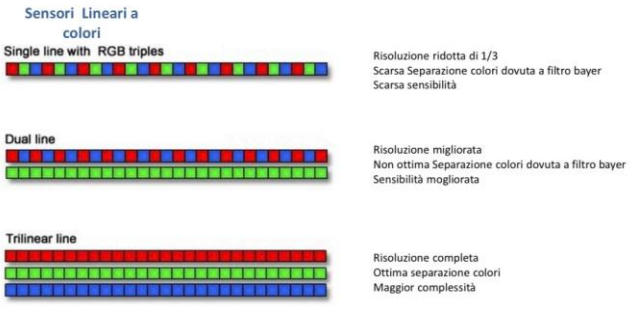
With devices that belong to the same family, what will happen is that the resolution of two cameras could be different or they have a different value of line rate, because we are using the same protocol and also the same bandwidth (they come from the same family).

Some characteristics of this kind of systems:

- Higher space resolution => lower line rate ???
- The FOV of these cameras is a line (but we can have sensors with more than one line)
- Each line has to be acquired using a shutter time < line rate
- The binning is a technique that takes the contribute of 4 pixels and considers them like one single pixel: in this way we obtain a higher sensitivity, because we have a larger pixel, but less resolution.

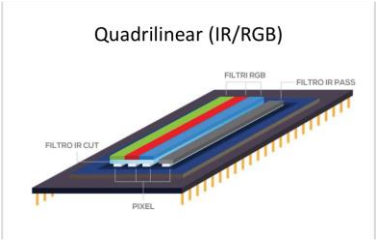
Colour Line Scan Camera

With a line scan camera, we have a sensor constituted by 1 line. For a better interpolation of pixels (to obtain all the colour components for each pixel) we can use a couple of lines. In this case, we have to consider the motion of the object also in the interpolation, because I consider the other components that I want to calculate, that are acquired in a different time respect to the line that I am considering**.**

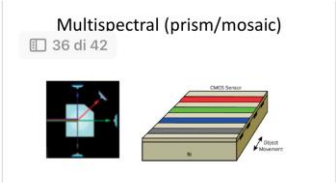


Materials signature and sensors

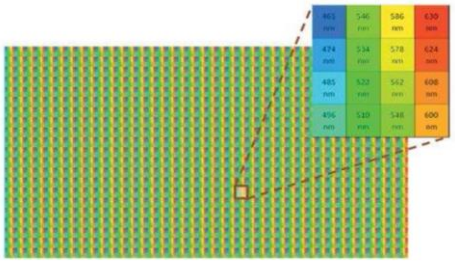
Each object reflects specific wavelengths, so we can analyse the spectrum product and understand the material that compose the object. Some sensors have a better sensitivity also in the infrared wavelengths, like the quadrilinear sensor, that is sensible to infrared band and is able to analyse crack into objects.



Multispectral camera



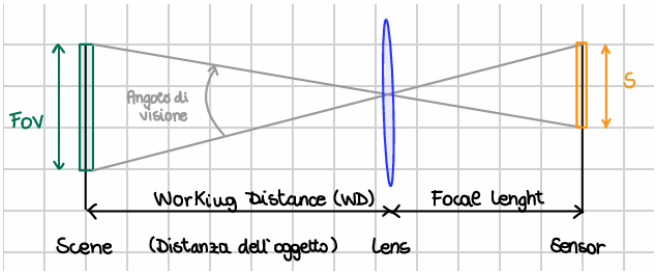
In this kind of sensors, pixels are divided into groups and each pixel of the group is sensible to a different sub band.

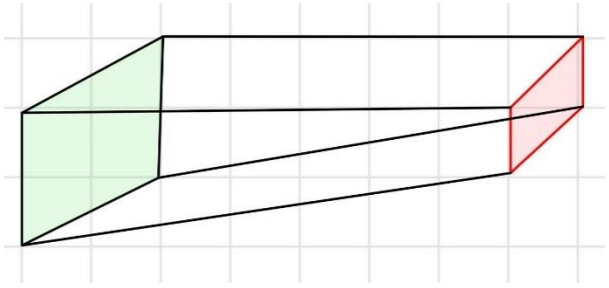


Lens

Basics characteristic of lens

The main goal of the lens is to convey the highest part of energy over the sensor. We can describe all the main characteristics of a system of lens:





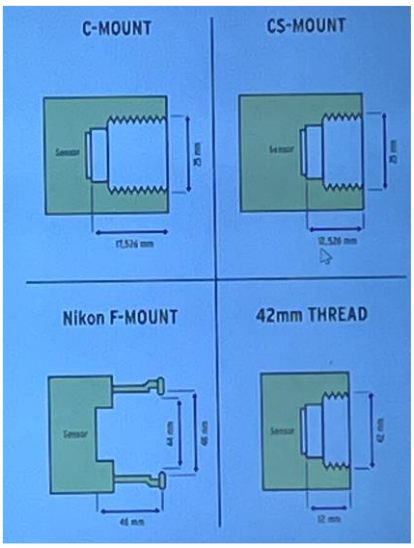
1) Field of view (FOV): area that we want to acquire;
2) Working distance: distance between FOV and sensor;
3) Focal length: distance between the sensor and the lens, related to the lens:
   - ⇨ Zoom lens have a variable focal length but they are more conservative in terms of energy, light
   - ⇨ Constant lens

$$FL = WD * \frac{SensorSize}{FOV}$$

An important thing that we have to consider before we buy a lens for our camera is the mechanical connection between lens and sensor: we have to match the lens **mount** and so consider the same value of it.

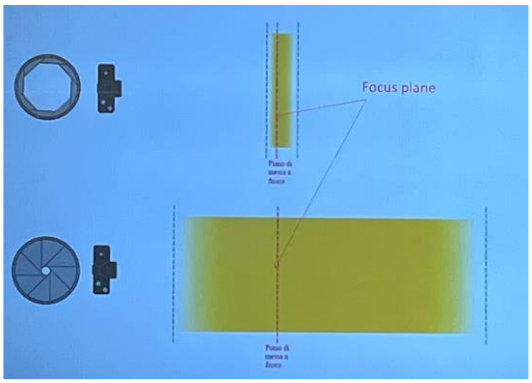
We can have different kinds of mount like those ones shown in the image.

Another important parameter is the **maximum sensor format,** that is the highest measure of diagonal of the sensor.



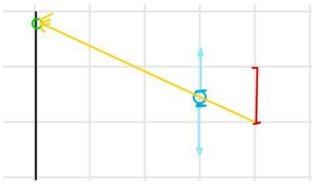
The **depth of field** is the range in which the object is focused on our lens. This parameter depends on the opening of the "diaframma" of our camera:

- Smaller the opening => larger the depth of field
- Larger the opening => smaller the depth of field



With a larger opening the sensor receives a lot of light so, some parts of the image are acquired in different pixels (not only in one specific) and this creates an overlap of information between different pixels.

The **frame rate** measures how many frames I can acquire in one second.



Exercise

Es. Sensor A: 1936 (resolution) × 5.86 μm (pixel size) ≈ 11.300 mm; 1216 × 5.86 μm = 7100 mm → 11.3 × 7.1 mm² → d = 13,3 mm (diagonale)

Sensor B: 1920 × 3.45 μm = 6.6 mm; 1200 × 3.45 μm = 4.1 mm → 6.6 × 4.1 mm² → d = 7.6 mm

Sensor C: 2048 × 3.45 μm = 7.0 mm; 1536 × 3.45 μm = 5.3 mm → 7.0 × 5.3 mm² → d = 8.9 mm

Ogni lente è caratterizzata da una propria lunghezza focale e da un diametro che deve essere maggiore della diagonale del sensore:

→ d = 2/3" ≈ 16.9 mm → max sensor diagonal

→ Focal Lenghts disponibili: 12 - 25 - 50 mm

Consideriamo un FOV = 80 cm × 70 cm e scegliamo all'inizio FL = 25 mm.

A) $WD = \frac{\text{Focal Lenght}}{\text{Sensor A wide}} \times FOV = \frac{25\text{ mm}}{11.3\text{ mm}} \times 80\text{ cm} = 177\text{ cm}$

$\text{Sensor Size (A)} = \frac{FOV}{WD} \times \text{Focal Lenght} = \frac{70\text{ cm}}{177\text{ cm}} \times 25\text{ mm} = 9.9\text{ mm} > 7.1\text{ mm}$

$FOV = \frac{7.1\text{ mm}}{25\text{ mm}} \times 177\text{ cm} = 50\text{ cm} < 70\text{ cm}$ → L'immagine viene tagliata, quindi non possiamo lavorare a una WD = 177 cm perché è insufficiente per rappresentare un FOV = 80 × 70 cm.

$WD = \frac{25\text{ mm}}{7.1\text{ mm}} \times 70\text{ cm} = 246\text{ cm}$

B) $WD = \frac{25\text{ mm}}{4.1\text{ mm}} \times 70\text{ cm} = 426\text{ cm}$

$FOV = \frac{6.6\text{ mm}}{25\text{ mm}} \times 426\text{ cm} = 112.7\text{ cm} > 80\text{ cm}$ ok!

C) $WD = \frac{25\text{ mm}}{5,3\text{ mm}} \times 70\text{ cm} = 330\text{ cm}$ (Per ridurre devo scegliere una lunghezza focale inferiore (12mm))

Optical quality of glass of lens

When a light hits a surface, some parts are reflected, and other parts are transmitted.

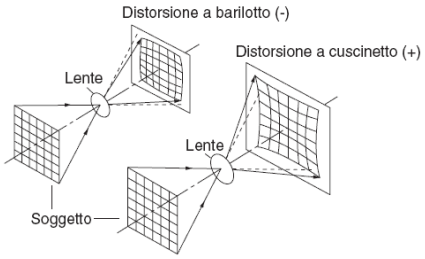
We consider the space frequency, i.e. how many transmissions between black and white we have into a given lens. 1 cm on 10 transmissions are all correctly acquired, but if we increase the number of transmissions, we don't see more back and white separately, but we see grey. It depends on the **modulation transfer function (MTF)** that depends on the quality of the lens.

A lens's MTF is a measure of its ability to transfer contrast at a particular resolution from the object to the image. As the line spacing decreases (i.e., the frequency increases), it becomes increasingly difficult for the lens to efficiently transfer this contrast.

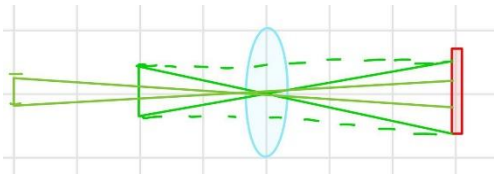
The quality of the lens is equal for each angular position. But what does it happen on a matrix sensor that is rectangular instead of the circle of the lens? We can note that a lens has different widths along its surface, so each of them follows a different optical behaviour.

We can have two kinds of distortions:

- Pillow distortion => this evolves into the diagonal direction and it happens with larger focal length
- Barrel distortion => it appears with shorter focal length and it's opposite to pillow distortion, from diagonal to the centre



Telecentric lens



One of the most popular problems is the different sizes of the image that we obtain, depending on the distance between the image and the lens. Telecentric lens resolves this problem, supposing to have an infinite focal length with parallel behaviour. In this case the depth of field increases slowly, so we have to increase the shutter time in order to have more light that hit the sensor surface. FOV is constant also with a different working distance.

Protocols for the connection between camera and PC

The image that we produce with our camera, usually has to be transmitted, processed and exc. In order to do all the possible processing steps, we have to connect our camera to a digital technology like a PC. The main parameters that we have to consider when we want to connect our camera to a PC is the Band/Bandwidth because, for several applications, it could be a problem.

At the start, the image is represented by an analogue electrical signal that varies over time, so it is necessary to convert it into a digital signal: for each line of pixel we can acquire the corresponding voltage, this will be quantized and then each value is mapped into the corresponding bits. For this phase an acquisition card inside the PC is required.

There are many different protocols for the connection:

1. Firewire

Standard introduced in 1995 by Apple to allow the connection between PC and different peripherals.

In 2000 it was used as a standard interface for machine vision such as Firewire (IEEE1394a).

The video data band was about 400 Mb/sec, in 2002 an evolution into firewire B (IEEE1394b) led to a maximum bandwidth of 800 Mb/sec.

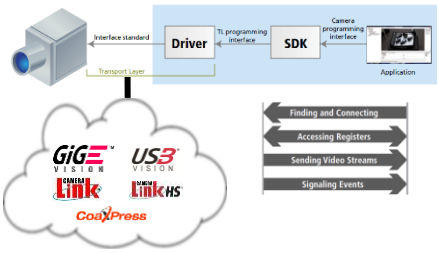
2. Universal Serial Bus USB

Introduced by Intel, Microsoft Compaq and IBM with the aim of allowing communication between different devices and the PC.

The recent version is USB 3.0 in which the data band for image transfer approx. 350MB / sec and use the standard GenICam: we have the possibility to connect each camera in each port of the PC otherwise we can use a hub in which we connect all the cameras but in this case the sum of all this stream cannot exceed the band for that port. So, we can't use all cameras at the same power in order to not saturate the band because they are sharing the bandwidth of the port.

GenICam

GenICam (Generic Interface for Cameras) is a generic programming interface for machine vision applications. The objective of the standard is to separate the technology used for the physical interface of the camera from the application interface presented to the user (API).



It is a layer devoted to creating the separation between protocol communication and the vision software system. Using this API, we have to carefully consider the bandwidth and the computing power that we need to analyze our data. The software requires a certain number of frames per second for a given resolution: I need to balance the computing power that I need and the hardware components that I really have.

GenICam is composed of modules that help to carry out the main tasks in the field of machine vision in a generic way:

- **GenTL** Standardizes programming interface for TLs (transport levels) => Makes TLs interchangeable
- **GenApi** Standardizes feature description language (XML) => Maps features to register
- **SFNC** Standard feature naming convention (~1.700 entries) => Exhaustive list of standardized camera features
- **PFNC** Pixel format naming convention
- **GenCP** Generic Control Protocol

Some basic functions are:

- Configure the camera: this function can support a series of camera parameters such as the frame size, the acquisition speed, the pixel format, the gain, the offset, etc.
  - ⇨ We can change the frame size in our software maybe because we want to care about only into a specific part of the sensor. To do this we can:
    I. Receive the total image and then we ignore and neglect the part in which we are not interested in.
    II. The camera can transmit only the interested part with a neglected mode of the camera, in this way the other lines are discard before the transmission. If we decrease the quantity of data, we can increase the frame rate, but we have always to consider a constant bandwidth in this way we reduce the traffic on the mother board. This reduction happens only when we remove rows from our images, but we don't see a reduction if we remove columns. In order to preserve the quality of the image, is better to discard a bottom and top line at the same time to avoid distortion of the image (because the lens works better in the centre part).

- Image acquisition
- Graphical interface
- Extra data transmission
- Event management

3. GigE Vision

Standard introduced in 2006 by AIA: Data band 100 MB / sec with single port, 200 MB / Sec with link aggregation; Cable length up to 100 m in copper plus with buckle; Power over GigE: is an additional feature that permit to send power to the camera with an ethernet cable.

4. Camera link

This was the fastest standard in the market and require devoted interfaces to use it. Camera can employ multiple connection so require more cable to send the information: we have a camera with a high frame rate so we have to use more cable if our PC band is lower respect the frame rate that we need. High communication speed and data bandwidth up to 850 MB/sec.

5. Camera link High Speed HS

Very high-speed cameras. Evolution of the CL protocol by improving cables and connectors. Gen <i> Cam with XML inside the camera. 3.125 Gb / sec for copper or optical fiber channel.

6. CoaXPress

6.25Gb / sec per channel (twice the CLHS). Supports link aggregation of up to 6 channels.

The frame grabber

The frame grabber is an object that grabs a frame from the camera. In our PC we have:

- The logic for coding the data, so the hardware that makes the coding, that could be DSP or FPGA;
- The buffer: data are sent to the buffer, in case the PC bus is full, that works with a FIFO politics.

To process our data, in general, we follow 3 steps:

1. Pre-processing => procedure that are almost the same as to identify the region of interest (ROI) and extract it from the image (cutting, filtering operations);
2. Processing => main algorithm to detect images;
3. Post-processing => make a suitable presentation of the results for the user.

An example of pre-processing operation could be the **filtering** with a low pass filter, that cuts the highest frequencies and maintain the lower frequencies. This method is used to analyse an image, for example to find a specific characteristic inside it and it is a basic way to remove noise: we consider a group of closed pixels with homogeneous values, and we suppose that the average of the noise is zero.
I consider the central pixel as the average of the others, and I assign to the pixels of the considered group a weighted value that decreases as the distance from the central pixel increases.
So, the average value V_{pix} is:

$$V_{pix} = \sum \frac{(V_i * w_i)}{N}$$

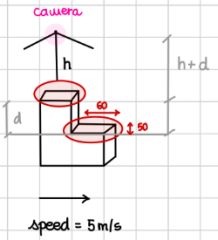
This value is an estimation of the pixel value without noise. We remove the error from noisy pixels if the average error is zero:

$$V = V_{pix} + \varepsilon_{error}$$

Larger is the image respect to the noisy pixel, more easily we can assume that the average is zero.

Another pre-processing operation is the **Edge Extraction** with a high pass or low pass filter.
In order to pre-process the data over the stream we need an additional hardware.

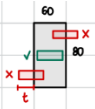
Exercise 1



FOV = (60 × 80) cm ⟶ Devo accertarmi che l'oggetto rientri in queste dimensioni

frame rate = 50 fps per una corretta acquisizione dell'immagine.

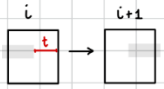
$t = \frac{1}{50\,fps} = 20$ ms (tempo fra due acquisizioni consecutive)



⟶ Poiché è impossibile avere un'immagine con queste dimensioni esatte, bisogna fissare una certa tolleranza t.

Resolution = 20 pix/cm

Working distance = 1,2 m



Entrambe le acquisizioni non sono in grado di catturare l'intero oggetto, che in 20 ms avrà percorso:

$t = \text{speed} \cdot t = 5\,\frac{m}{s} \cdot 20 \cdot 10^{-3}\,s = 10$ cm ⟶ tolleranza ⇒ Il FOV deve essere quindi (60+10) cm × 80 cm.

↑ pixel più piccolo che vogliamo acquisire

Siamo interessati ad acquisire un "difetto" di 1 mm, quindi la risoluzione minima deve essere 1 pix/mm = 10 pix/cm. Questa risoluzione, però, potrebbe non essere sufficiente perché le immagini devono essere analizzate da software che applicano dei filtri. Supponiamo di avere un filtro 5*5 e decidiamo di avere una risoluzione finale di 3 pix/mm:

(60 × 80) cm × $3\,\frac{pix}{mm}$ = 1800 × 2400 pixel.

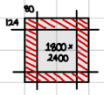
Un approccio alternativo può essere quello di utilizzare 2 camere con ris. 1936 × 1216 px con frame rate = 48 fps.

Scegliamo di utilizzarne una sola con le seguenti caratteristiche:

Camera: res = 2048 × 2560 px » 1800 × 2400 necessari ⟶ Posso decidere di usare anche solo una parte del sensore, escludendo alcune righe o colonne.

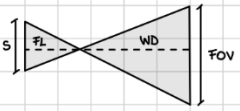
fps = 107 ⟶ t = 5 ms

WD < 200 cm = 2 m



Sensor size: (5×5) μm ⟶ (1800×2400) × 5 μm = (12×9) mm ⟶ diagonale = 15 mm = 0.59" (1 pollice = 25.4 mm)

Lente: 7.5 Mpx con C-Mount



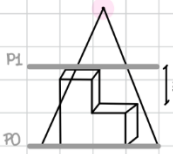
WD FOV s

120 : 60 = FL : 9 mm ⟶ FL = 18 mm ⟶ La lente per costruzione può avere o FL = 16 mm o 25 mm;

1.2 m

WD = 1,06 m WD = 1,66 m < 2 m Ok!

noi scegliamo quella con WD maggiore.



depth of field = 30 cm (Per avere questa profondità di campo è necessaria una certa apertura delle iride.)

Per avere un'immagine ancora migliore possiamo:

① Utilizzare una sorgente di luce aggiuntiva (costo maggiore)

② Impostare il guadagno della camera per aumentare l'intensità (introducendo però del rumore)

③ Aumentare lo shutter time ⟶ Maggiore energia in ingresso ⟶ Maggiore intensità luminosa (ma potrebbe introdurre il MOTION BLUR)

↓

Supponiamo di avere un frame rate = 50 fps e uno shutter time = 500 ns, che è buono.

Ora devo verificare che l'immagine non venga acquisita in movimento: questo accade quando un punto dell'oggetto si trova distribuito su più pixels dell'immagine.

v = 5 m/s ⟶ $d = v \cdot t = 2.5 \cdot 10^{-6}$ m ⟶ Se d < 1 pix, l'immagine risulterà in movimento

t = 500 ns

↳ non devo considerare le dim. del sensore fisico (5 μm)

ma il FOV mappato su ogni pixel (3 pix/mm = 1/3 mm/pix = 333 μm)

Exercise 2

$v = 80\ m/s$
$res = 1\ pix/mm$
$WD = 200\ cm = 2\ m$
Camera size = 1936×1212
$fps = 48$

$FOV = (30 + t) \times 130\ cm^2 = 190 \times 130\ cm^2 \xrightarrow{1\ pix/mm} 1900 \times 1300\ pix \longrightarrow$ La camera scelta non va bene
(t: tolleranza)

$t = \frac{1}{48}\ s \times 80\ \frac{m}{s} = 160\ cm$

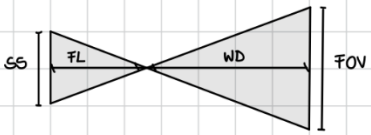
- Scegliamo una camera 2560×2048 con $fps = 107$

  $t = \frac{1}{107}\ s \times 80\ m/s \simeq 80\ cm \Rightarrow FOV = (30 + 80)\ cm \times 130\ cm = 110 \times 130\ cm^2 \longrightarrow 1100 \times 1300\ pix$ ✓

Poiché la camera è molto più grande del FOV richiesto, scelgo una camera 1500×1500 con lo stesso frame rate.

Sensor size $\begin{cases} 5\ \mu m \times 1500 = 7.5\ mm \longrightarrow \text{diagonale} = 10.6\ mm = 0.4'' \\ 5\ \mu m \times 1500 = 7.5\ mm \qquad S = (7.5 \times 7.5)\ mm^2 \end{cases}$

$WD : FOV = FL : Ss \longrightarrow \underset{(cm)}{200} : \underset{(150\,cm)}{1500} = FL : \underset{(mm)}{7,5} \longrightarrow FL = 10\ mm \longrightarrow$ In commercio ho lenti con lunghezza focale 8.5 mm o 12 mm. (*)

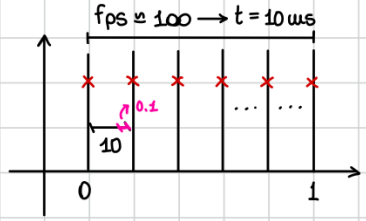


(*) Calcolo la WD utilizzando le due lenti:

1) $WD : 1500 = 8,5 : 7,5 \longrightarrow WD = 1,7\ m$
2) $WD : 1500 = 12 : 7,5 \longrightarrow WD = 2,4\ m$

- Quanto tempo richiede il sistema per shiftare di 1 mm? (1 pix in digital image corrisponde a 1 mm) $\longrightarrow t = \frac{1\ mm}{80\ m/s} = 12,5\ \mu s$

  Shutter time deve essere minore di 9.9 ms per la mia camera, quindi t va bene.



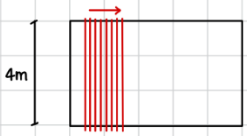
$10 - 0,1 = 9,9\ ms$
↳ tempo necessario per eseguire operazioni computazionali

Exercise Line Scan Camera

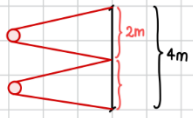
res = 2 pix/mm

n. pix = 4m · 2 pix/mm = 8K

v = 100 m/s



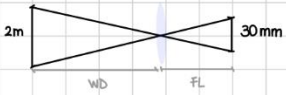
Vogliamo acquisire l'immagine di un oggetto lungo 4 m con una line scan camera. La camera deve essere quindi posizionata sul lato corto dell'oggetto.
La camera deve avere una frequenza minima pari a $\frac{100\ m/s}{2\ pix/mm}$ = 200 K lines/s = 200 KHz. Ma in commercio ci sono una camera da 8K pixel e 1ps = 100 kHz e una da 4K pixel e 200 KHz, quindi o devo ridurre la velocità dell'oggetto oppure devo scegliere di rilevare un difetto di 2mm nella direzione verticale e 4 mm in quella orizzontale. Se nessuna delle due opzioni è accettabile per il cliente devo cambiare dispositivo.
Scegliamo di usarne 2 da 4K per soddisfare le nostre specifiche. La 2ª camera riceve il segnale di trigger dallo stesso dispositivo che controlla il movimento dell'oggetto. Infatti, se vogliamo mantenere la dimensionalità dell'oggetto, dobbiamo sincronizzare la sua velocità con la frequenza di acquisizione delle lines della camera.



Tick to compare		Product Line	Model	Type	Color / Mono	Light Spectrum	Resolution (MP)	Resolution (WxH)	Frame rate / Line rate	ROI	Interface	Sensors	Sensor Name	Cell Size (WxH)
	Quick view	Sweep Series	SW-16000M-CXP4A	Line Scan	Mono	Visible	N/A	16384 x 1	277 kHz	Yes	CoaXPress-4-Lanes	1xCMOS	GL5016	5.0 x 5.0 µm
	Quick view	Sweep Series	SW-2005M-5GE	Line Scan	Mono	Visible	N/A	2048 x 1	172 kHz	Yes	5 Gbps GigE Vision (PoE)	1xCMOS	GL3504	7.0 x 7.0 µm
	Quick view	Sweep Series	SW-2005M-CXP	Line Scan	Mono	Visible	N/A	2048 x 1	172 kHz	Yes	CoaXPress-1-Lane (PoCXP)	1xCMOS	GL3504	7.0 x 7.0 µm
	Quick view	Sweep Series	SW-4000M-PMCL	Line Scan	Mono	Visible	N/A	4096 x 1	200 kHz	N/A	Mini Camera Link (PoCL)	1xCMOS	Custom	7.5 x 7.5 µm
	Quick view	Sweep Series	SW-8000M-PMCL	Line Scan	Mono	Visible	N/A	8192 x 1	100 kHz	N/A	Mini Camera Link (PoCL)	1xCMOS	Custom	3.75 x 5.78 µm

La dimensione del pixel è 7.5 µm, quindi il sensore è grande 4096 × 7.5 µm ≈ 3 cm.
Serve quindi una lente di 30 mm o 3/4".

Utilizziamo una lente di 30 mm e lunghezza focale 35 mm:



FL : Ss = WD : FOV ⟶ 35 mm : 30 mm = WD : 2m

$WD = \frac{35\ mm \cdot 2000\ mm}{30\ mm} = 2{,}33\ m$

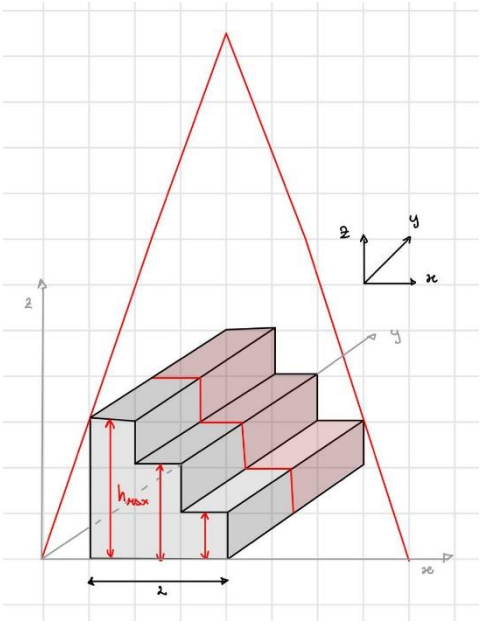
Servono inoltre:

- Frame Gabber (4 Camera Link ports)
- 4 Camera Link cables (2 per ogni camera)
- Poichè ogni acquisizione avviene ogni $t = \frac{1}{200\ kHz} = 5\ \mu s$ (shutter time) che è un valore molto basso, è necessario fornire più luce con un sistema di illuminazione aggiuntivo che va sempre sincronizzato con la camera.
- space = v·t = 100 m/s · 5 µs = 500 µm = 0.5 mm (spazio percorso dall'oggetto) ⟶ volendo ridurre la dimensione di ogni acquisizione, bisogna ridurre lo shutter time o aumentare il guadagno della camera (ma aumenta il rumore).

The Profilometry

The profilometry is the discipline that has the goal to obtain a tridimensional profile of an object, starting with a bidimensional acquisition.

We can consider a laser, positioned above the object, that creates a line that intersects the object; we also have a camera that looks the scene at different quotes: in this way we obtain the profile of the object. The camera detects only the profiles.

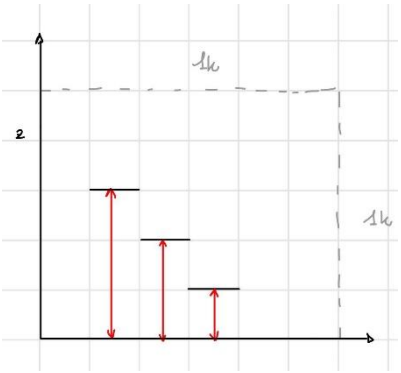


Camera and Laser are not in the same plane because, if they will be in the same plane, I will see with the camera only the line of the laser and I cannot reconstruct the object.

To reconstruct all the object, I have to collect different acquisitions of the profile by moving the object or by a linear system that moves along the horizontal axis with respect to the object that we are acquiring.

There are different possible configurations, called geometries, in which we can put camera and laser; each of them has different advantages to reconstruct the object.

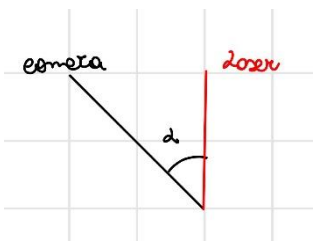
Now we are focusing on an image of one profile:

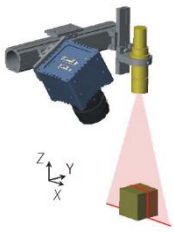


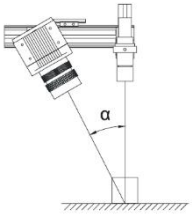
1. Detect in the image the points that are lighted by the laser, using a bandwidth filter that allows us to completely attenuate the other lights and maximize the sensitivity of the wavelength of the laser. Suppose we have to acquire an image composed by 1000 columns and 1000 rows: the horizontal resolution will be 1K/L and the vertical resolution will be 1K/h_{max}. If the object has $h_{max} = L = 1m$, we will have a resolution of 1 pixel per mm in both the dimensions.
2. Each point of the profile (the points that we acquired with the camera at the first step), in the reality, (the image that the camera acquires is bidimensional) is characterized by 3 coordinates (x, y, z):
   - ⇨ Coordinate x => each column of the image is related to a different value of x
   - ⇨ Coordinate y => y coordinate is constant for all the points in the same image: moving the camera, we acquire another different profile with a different y value, that is constant for each point of the same image. The total value of y is not represented in the image, so it is evaluated by the sum of more profiles.
   - ⇨ Coordinate z => each column of the image is related to z component, and each row will be associated to a certain quote.

Now we consider different possible setups, in which we can put our camera and the laser:

1) Laser perpendicular to the object and camera positioned with a certain angle α:







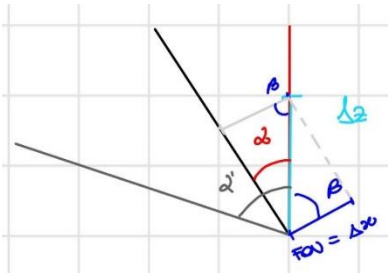
In this case the resolution in the vertical axis will be approximate with:

$$\Delta Z = \Delta X / \sin \alpha$$

The major part of the light is reflected in the same plane of the laser and only one part of the light reaches the other directions. So, smaller is alpha, larger will be the quantity of light detected by camera, but larger is the error in the acquisition.

The formula of Δz is evaluated considering this image:

$$\beta = 90° - \alpha$$
$$\Delta x = \Delta z \cos\beta = \sin\alpha\, \Delta z$$
$$\Delta z = \frac{\Delta x}{\sin\alpha}$$



In this configuration different values of z are always hit by the laser.

Lower the number of rows, shorter the dimension of the sensor with the same configuration so, when we reduce the number of rows, the side effect is:

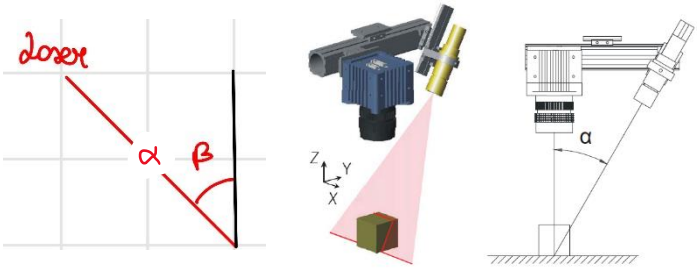
- ⇨ Increasing the frame rate (we have reduced the number of data)
- ⇨ Reducing the FOV
- ⇨ Increasing the speed of the object along the y direction

We divide the height over the Z direction dividing the different rows of the sensor over that dimension. In the image above, if we consider the angle α' that is larger than α, it creates higher noisy results because the energy that hits the sensor is lower.

The best resolution will be obtained if we maximize $\sin\alpha = 1$, so the resolution on the Z and X axes is the same, but if alpha is equal to 90°, our camera cannot acquire the profile and we don't see anything because the image will be black.

If we want to increase the light acquired by the camera without decreasing so much the angle, we can increase the power of the laser or evaluate the reflectance of the object.

2) Laser positioned with a certain angle α and camera perpendicular to the object:

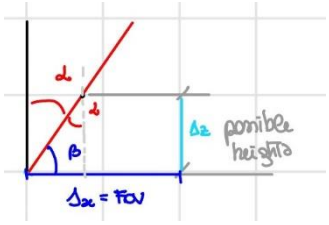


In this case the vertical resolution will be:

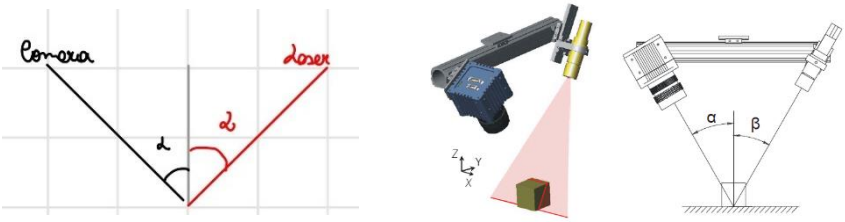
$$\Delta Z = \Delta X / \tan\alpha$$

The formula of Δz is evaluated considering this image:

$$\beta = 90° - \alpha$$
$$\Delta x = \Delta z \tan\alpha$$
$$\Delta z = \frac{\Delta x}{\tan\alpha}$$



3) Laser positioned behind the object with a certain angle β and camera positioned in front of the object with a certain angle α:



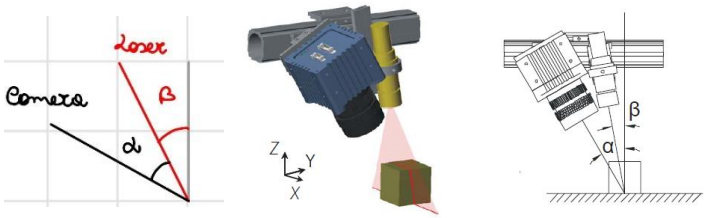
In this case the vertical resolution will be:

$$\Delta Z = \Delta X \cdot \cos\beta / (\sin\alpha + \beta)$$

If α=β: $\Delta Z = \Delta X / 2\sin\alpha$

We obtain the best acquisition, but it isn't the best configuration if we have an object with a lot of irregularities. This kind of situation guarantees to receive the highest part of the energy on the camera surface. This is the best configuration that maximize the reflected energy over the sensor.

4) **Laser positioned with an angle β and camera positioned with an angle α, both of them in front of the object:**



In this case the vertical resolution will be:

$$\Delta Z = \frac{\Delta X \cos\beta}{\sin(\alpha - \beta)}$$

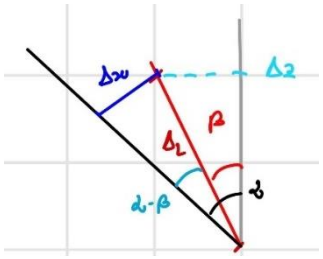
In this case, the laser that I see with my camera is only the laser reflected by the object, so only a small part of the energy of the light will reach the camera.

The Δz formula is evaluated considering this image:

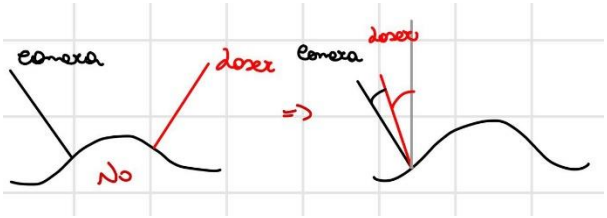
$$\Delta L = \frac{\Delta z}{\cos\beta}$$

$$\Delta x = \Delta L \sin(\alpha - \beta) = \frac{\Delta z}{\cos\beta}\sin(\alpha - \beta)$$

$$\Delta z = \Delta L \cos\beta = \frac{\Delta X \cos\beta}{\sin(\alpha - \beta)}$$



The advantage of this configuration is only visible if the object has an irregular surface.



Custom Profilometer

To personalize a profilometer we have to consider all the components of the system, the camera, the laser and also a software for the synchronization of the movement of the object and the acquisition of the profiles. An important issue is the number of profiles that I can reconstruct which depends on the time of the acquisition.

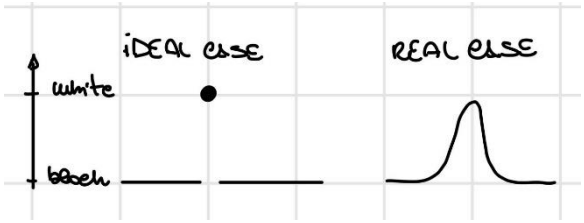
How can we reconstruct an object in 1 second? We have to acquire all the possible profiles in one second and, if we suppose that we can acquire 1 thousand profiles in a second, we have an acquisition of 1 millimetre per second. How much does the object move in y direction in this fraction of time?

Faster is the camera, higher will be the resolution in the Y axis (distance between different profiles), faster is the object, lower is the resolution in the Y axis.

In the ideal case we want a camera with a high resolution and a high frequency of acquisition, so probably it will be necessary to have a frame grabber (like with the line scan camera).

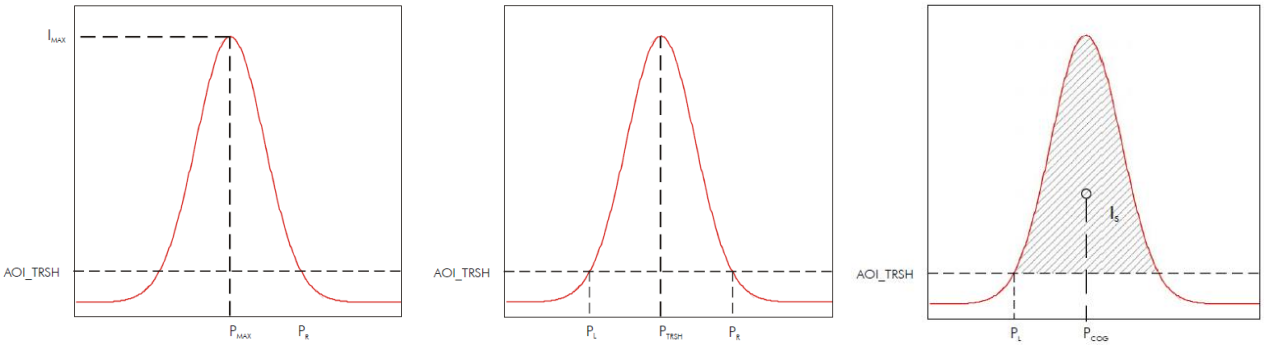
Another thing that we have to consider is that the filter we used in front of the lens has to be chosen according to the wavelength of the laser that we are using.
When we start to move the object and acquire several images, for each of them, for each column we reconstruct the coordinate Z value (each row of the column is related to a z value) and for each row we reconstruct the y coordinate. In the ideal case, we suppose to have a column composed by all black

spots and only one white spot, that represents our point on the profile, but this doesn't happen in the reality. Several pixels will be higher than zero, but not exactly one pixel white and the others black:



Now the problem is to detect a value that will be representative of that row, the z value that we are evaluating in that column. To do this, we can consider different solutions:

1) **Software solution**:
   - ⇨ consider the highest value of this shape (Maximum Intensity Profile Mode – MAX) → (1)
   - ⇨ make some interpolation of the final value;
   - ⇨ use a threshold and consider only the values higher than this threshold, then determine the z value by computing the maximum of these values (Threshold Mode – TRSH) → (2)
   - ⇨ compute the centre of gravity (centro di massa) of these points and determine a value that is not related to a single row: consider a threshold and then the average value of the points above the threshold. But this number could be not an integer, so it's necessary to choose the number of significant digits, i.e. the accuracy that we want to get. The accuracy depends on the instrument that we are using, but how can we understand the accuracy of the instrument? We can do several acquisitions and get the number of significant digits. This solution is better when we have highly reflective surfaces (instead of using the maximum). → (3)



(1) (2) (3)

2) **Hardware solution** => The hardware can do easily the computation of the final value of z. We can have an hardware embedded in our camera, according to the setting, that can use different strategies to reconstruct the value of z. We can choose to send a vector that contains all the values of the z coordinates of each column of the images that we acquired. The vector is exactly the profile that I need. This approach has two advantages:
   - ⇨ not do the computation statements: in this way it is not more necessary to analyse the image, fix a threshold, associate a value for z and then convert the value from pixel to mm.
   - ⇨ strongly reduction of the number of the data that the camera has to send in output. I can optimize the bandwidth of my camera with the computation system, so I can increase the number of profiles that I can send per second.

To increase the speed of the acquisition, i.e. to increase the frequency of the acquisition, one possible solution is to neglect some rows in order to acquire only a specific area of interest of the sensor: for example, we can acquire only 3 rows that correspond to 3 different areas of the sensor, each of them with its threshold. In this way we select the areas of interest based on the number of rows.

The dimension of the sensor is higher than in the previous case that we see, because we need a higher sensitivity and it also depends on the wavelength of the laser. The wavelength of the laser is chosen by considering different aspects:

- Reflectance of the surface at that frequency
- Danger provided by the laser: higher the class of the laser, higher the danger.

Exercise 1

C3-1280 Sensor

Parameters	Specifications
Sensitivity at peak response	1600 LSB / lux s @ 550 nm corresponds to 10695 LSB / µJ / cm²
Resolution	1280 x 1024
Pixel Size	12µm x 12µm
Sensor Size	15.4mm x 12.3mm, diagonal: 19.7mm
Optics	1 inch (C-Mount)
Sensor ADC Resolution	10 bit
Sensor Dynamic Range	59dB
Max. Internal Full-Frame Rate	450fps
Max. External Full-Frame Rate	61fps (40MHz-CameraLink, 2Tap-Mode) 91fps (60MHz-CameraLink, 2Tap-Mode)
Max. Internal Row Frequency at 1280 Pixels/Row	139kHz
Max. Profile Rate at Max. Row Length = Max. Internal Row Frequency / Number of Rows	28800 Hz (16 rows) → 24 14400 Hz (32 rows) → 40 7200 Hz (64 rows) 3600 kHz (128 rows) 1800 Hz (256 rows) 900 Hz (512 rows) 450 Hz (1024 rows)

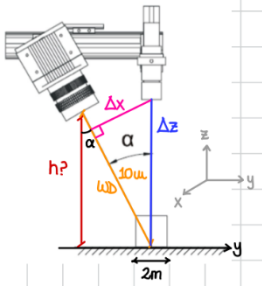
Devo ricostruire l'immagine di un oggetto acquisita con questa camera tramite un custom profilometer.

$\Delta z = 20$ cm

$\alpha = 30°$

$\lambda = 28$ mm (lunghezza focale della lente)

$\Delta y = 30$ cm



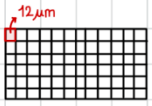
Ho bisogno di un FOV $\Delta x = \Delta z \cdot \sin\alpha = 10$ cm

Vogliamo trovare l'altezza della camera h.

Supponiamo che l'oggetto sia lungo 2m e che la camera acquisisca 2 oggetti al secondo e 5 profili/mm, quindi ha una velocità di 4 m/s:

$4\,m/s \cdot 5\,prof./mm = 20.000$ profili/s $\longrightarrow \sim 24$ righe

$SS = 24 \cdot 12\,\mu m = 288\,\mu m$



$WD : FOV = FL : SS$

$WD : 10\,cm = 28\,mm : 0.288\,mm \longrightarrow WD = 10$ m

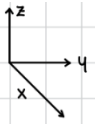
$\Rightarrow h = WD \cdot \cos\alpha = 10\,m \cdot \frac{\sqrt{3}}{2} = 8.66$ m

Ora dobbiamo verificare se riusciamo a ricostruire il profilo nella direzione di FOV = 30 cm.

$WD : FOV = FL : SS \longrightarrow 10\,m : 30\,cm = 28\,mm : SS \longrightarrow SS = 840\,\mu m \longrightarrow$ richiede n. pixel $= \frac{840\,\mu m}{12\,\mu m} \simeq 70 < 1280$ ✓

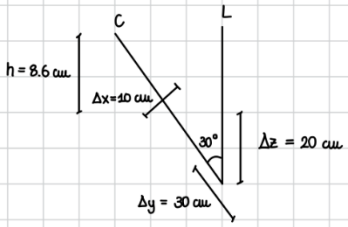
Sarà necessario quindi parzializzare il sensore, usando solo una parte delle righe e delle colonne: in questo caso migliora il throuput.

→ Qual è la risoluzione lungo x, y e z?



(con questo riferimento)

- Lungo z: $\frac{24\ righe}{20\,cm} = \frac{24\ pixel}{20\,cm} = 1\ pix / 0.83\,cm$
- Lungo y: $5\ pix/mm$ (dato in input) $= 1\ pix / 0.02\,cm$
- Lungo x: $\frac{1\ pix}{0.83 \sin\alpha} = \frac{40\ pix}{30\,cm} = 1\ pix / 0.43\,cm$



Supponiamo di voler aumentare la risoluzione lungo z a 1 pix/0.5 cm. Una soluzione potrebbe essere quella di aumentare il numero delle righe ①, ma il side effect è che si riduce la frequenza. Il numero di righe sarà $20\,cm \cdot \frac{1\ pix}{0.5\,cm} = 40$ (~ 12.000 Hz) $\longrightarrow +30\%$ di righe $\longrightarrow -30\%$ frequenza : 10.000 Hz.

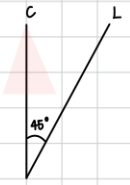
Inoltre la velocità dell'oggetto si dimezzerà perché saranno acquisiti 10.000 profili/s.

La dimensione del sensore sarà di 40 righe · 12 µm = 480 µm e $WD = \frac{10\,cm \cdot 28\,mm}{480\,\mu m} = 583$ cm $\longrightarrow h = WD \cdot \cos\alpha = 5.04$ m

Tuttavia ridurre la velocità dell'oggetto non è una soluzione accettabile. Si potrebbero eventualmente suddividere le linee. ②

Anche cambiando l'angolo, si avrebbero sempre 24 righe. Tuttavia nei casi reali, riducendo l'angolo, si hanno prestazioni migliori nell'interpolazione dei subpixel. ③

Exercise 2

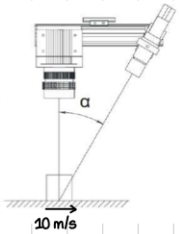


$\Delta z = \frac{\Delta x}{tg\alpha}$

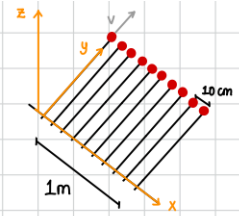
$\lambda = 28$ mm

1 pix / 0,02 mm (res. lungo z)

$v = 10$ m/s

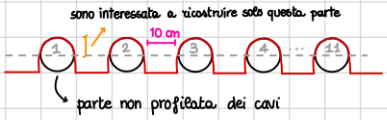


Geometria 2



11 cavi con

diametro = 0.8 cm



C3-2350 Sensor

Parameters	Specifications
Sensitivity at peak response	2500 LSB / lux s @ 610 nm
Resolution	2352 x 1728
Pixel Size	7µm x 7µm
Sensor Size	16.46mm x 12.10mm, diagonal: 20.43mm
Optics	1" C-Mount and F-Mount
Sensor ADC Resolution	10 bit
Sensor Dynamic Range	59dB
Max. Internal Full-Frame Rate	190fps
Max. External Full-Frame Rate	20fps (40MHz-CameraLink, 2Tap-Mode) 30fps (60MHz-CameraLink, 2Tap-Mode)
Max. Internal Row Frequency at 2352 Pixels/Row	115kHz
Max. Profile Rate at Max. Row Length = Max. Internal Row Frequency / Number of Rows	23450 Hz (14 rows) 12160 Hz (27 rows) 3040 Hz (108 rows) 1520 Hz (216 rows) 760 Hz (432 rows) → 200 380 Hz (864 rows) 190 Hz (1728 rows)

1 pix / 0.02 mm = 50 pix/mm ⟶ 50 pix/mm · 8 mm = 400 pix = 400 righe ⟶ ~ 820 Hz

f = 820 Hz, v = 10 m/s ⟶ ogni profilo dista dall'altro $\frac{10\ m}{820} = 0{,}012\ m = 12$ mm

Tuttavia a noi interessa profilare solo metà circonferenza, quindi 4 mm ($\Delta z = 4$ mm):

1 pix / 0.02 mm · 4 mm = 200 righe ⟶ ~ 1550 Hz

$\Delta x = \Delta z \cdot tg\alpha = \Delta z \cdot tg\,45° = 4$ mm

SS = 200 pix · 7 µm = 1400 µm = 1.4 mm

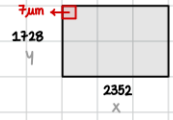
WD : FOV = FL : SS ⟶ WD : 4 mm = 28 mm : 1.4 mm ⟶ WD = 80 mm

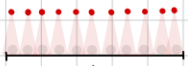
La distanza fra due profili consecutivi è Δy. Il periodo di acquisizione della camera è $\frac{1}{1550\ Hz} = 0{,}6$ ms, la velocità della camera è 10 m/s, quindi:

$\Delta y = T \cdot v = 0{,}6$ ms $\cdot$ 10 m/s = 6 mm

Ora dobbiamo verificare se la camera riesce a ricostruire il FOV = 1m: WD : FOV = FL : SS ⟶ 80 mm : 1m = 28 mm : SS ⟶ SS = 350 mm

Sono necessari quindi n. pixels = $\frac{350\ mm}{7\ \mu m} = 50.000 < 2352$ ⟶ Una camera non è sufficiente: $\frac{50.000}{2352} = 22$ camere.





Come posso quindi ridurre il costo di questo sistema?

⟶ Posso usare 11 camere, 1 per ogni cavo, con 400 pixel, dato che non ho bisogno di ricostruire tutto il FOV di 1m.