IMAGE FORMATION

Antonino Furnari
furnari@dmi.unict.it
http://dmi.unict.it/~furnari
VISUAL PERCEPTION

“the ability to interpret the surrounding environment by processing information that is contained in visible light”
The visual system allows individuals to assimilate information from their surroundings. The act of seeing passes through the following steps:

- The lens of the eye focuses an image of the scene onto the retina, which is actually part of the brain that is isolated to serve as a transducer for the conversion of patterns of light into neuronal signals;
- The retina detects the photons of light and produces neural impulses;
- The signals are processed by different parts of the brain in a hierarchical fashion.

VISUAL SYTEM OF MAMMALS
THE EYES

The eyes are an important part of the visual system and are responsible for the initial representation (arrays of intensity values) of the scene which our brain manipulates to enable us to see the world.

Understanding the optical process through which an image is formed on the retina is the first step to model visual perception.
SCOPE OF THIS UNIT

We will concentrate on the Computer Vision methodologies to extract information related to the 3D arrangement of the physical surfaces present in the real scene starting from the 2-dimensional projection of the scene to an image.

Understanding the 3D structure of a scene given a 2-dimensional representation is an inverse problem and indeed it is a difficult one!

Most of the information present in the 3D scene is lost in the 2-d projection process. However, it is possible to "recover" part of this information using constraints derived from our a-priori knowledge of the real world.

To understand and model an inverse problem, it is essential to study its forward counterpart, which is the scope of part of this unit.
PROJECTION OF A 3D SCENE TO A 2D IMAGE
AGENDA

- Introduction
- Geometric Primitives & Transformations
- Image Formation
- From the World to The Camera
GEOMETRIC PRIMITIVES

3D and 2D geometric primitives are the basic building blocks used to describe three-dimensional shapes. We will discuss the following primitives:

- 2D and 3D points;
- 2D lines;
- 3D planes.

We will moreover discuss the homogeneous coordinate system.
2D points are usually denoted by \( x = (x, y) \in \mathbb{R}^2 \) or alternatively:

\[
\mathbf{x} = \begin{bmatrix} x \\ y \end{bmatrix} \quad \tilde{\mathbf{x}} = \begin{bmatrix} x \\ y \\ w \end{bmatrix}
\]

Similarly, 3D points are denoted by \( x = (x, y, z) \in \mathbb{R}^2 \) or alternatively:

\[
\mathbf{x} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \quad \tilde{\mathbf{x}} = \begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix}
\]
Homogeneous coordinates or projective coordinates are a system of coordinates used in projective geometry; Homogeneous coordinates have a range of applications, including computer graphics and 3D computer vision, where they allow affine transformations and, in general, projective transformations to be easily represented by a matrix; The points are projected from a space of dimensions $n$ to a space of dimensions $n+1$: $P^n = \mathbb{R}^{n+1} - \vec{0}$; In the 2D case: $x = (x, y) \in \mathbb{R}^2 \rightarrow \bar{x} = (\tilde{x}, \tilde{y}, \tilde{w}) \in P^2$; In the 3D case: $x = (x, y, z) \in \mathbb{R}^3 \rightarrow \bar{x} = (\tilde{x}, \tilde{y}, \tilde{z}, \tilde{w}) \in P^3$; The conversion from homogeneous coordinates to Cartesian ones is possible dividing the homogeneous coordinates by $w$. 
EXAMPLE

(cartesian point)

(projective points)

(scale factor (if equals to zero, the point is at infinity)

all points are equivalent)

\[ x_i = \omega \times x \]
\[ y_i = \omega \times y \]

Example:

\[(2, 6) \quad (3, 9, 1.5) \quad (1, 3, 0.5) \quad \ldots\]

\[\xrightarrow{1.5} \quad \xrightarrow{0.5} \quad \xrightarrow{2} \quad \xrightarrow{2} \quad \ldots\]
**AUGMENTED VECTORS VS HOMOGENEOUS VECTORS**

Given the 2D point $\mathbf{x} = (x, y)$, the easiest way to obtain an equivalent homogenous point, is to add a 1 as the third value. We refer to this vector as **augmented vector**:

$$\mathbf{x'} = (x, y, 1)$$

Moreover, we denote an homogeneous point equivalent to point $\mathbf{x}$ with the notation:

$$\tilde{\mathbf{x}} = (\tilde{x}, \tilde{y}, \tilde{w})$$

Similar considerations hold for 3D points.
A line in is represented in cartesian coordinates by its canonical equation:

\[ ax + by + c = 0 \]

or alternatively in homogeneous coordinates as: \( \vec{l} = (a, b, c) \).

Lines can also be represented using the normalized form \( \vec{l} = (\hat{n}, d) \), where \( \hat{n} = (\cos \theta, \sin \theta) \) is the normal vector perpendicular to the line and \( d \) is the distance from the origin.

\[ ||\hat{n}|| = 1 \]

\( \hat{n} = (\cos \theta, \sin \theta) \)
A plane in the 3D space is represented in cartesian coordinates as:

$$ax + by + cz + d = 0$$

or alternatively in homogenous coordinates as:

$$\overline{m} = (a, b, c, d)$$

The 3D planes can also be represented using the normalized form

$$\overline{m} = (\hat{n}, d)$$

where $\hat{n}$ is the normal vector to the plane and $d$ is its distance from the origin.
TRANSFORMATIONS

- **Translation**: preserves orientation;
- **Euclidean**: preserves lengths;
- **Similarity**: preserves angles;
- **Affine**: preserves parallelism;
- **Projective**: preserves straight lines.
A 2D point $x$ is translated by vector $t$ as:

$$x' = x + t$$

This is equivalent to:

$$x' = [I \ t][\bar{x}]$$

where $I$ is the $(2 \times 2)$ identity matrix and $\bar{x}$ is the original point in homogeneous coordinates. If we wish to obtain an homogeneous result:

$$\bar{x}' = \begin{bmatrix} I & t \\ 0 & 1 \end{bmatrix} \bar{x}$$

where $0$ is the $(2 \times 1)$ zero vector. The translations preserve line orientations.
2D ROTATIONS + TRANSLATIONS (EUCLIDEAN)

The transformation can be written as:

\[ R = \begin{bmatrix} \cos \theta & -\sin \theta & dx \\ \sin \theta & \cos \theta & dy \\ 0 & 0 & 1 \end{bmatrix} \]

\[ x' = Rx + t \]

where \( R = \begin{pmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{pmatrix} \) and \( \theta \) is the angle by which the points are rotated. In matrix form:

\[ x' = [R \quad t]\bar{x} \]

\[ \bar{x}' = \begin{bmatrix} R & t \\ 0^T & 1 \end{bmatrix}\bar{x} \]

The euclidean transformations preserve lengths.
The transformation can be written as:

\[ \mathbf{x}' = s \mathbf{R} \mathbf{x} + \mathbf{t} \]

where \( \mathbf{R} = \begin{pmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{pmatrix} \) and \( \theta \) is the angle by which the points are rotated. In matrix form:

\[ \mathbf{x}' = [s \mathbf{R} \quad \mathbf{t}] \bar{\mathbf{x}} \]

\[ \bar{\mathbf{x}}' = [s \mathbf{R}^T \quad \mathbf{t}] \bar{\mathbf{x}} \]

The similarity transformations preserve angles between lines.
2D AFFINE TRANSFORMATION

The affine transformation is written as:

\[ \mathbf{x}' = A\mathbf{x} \]

where \( A \) is an arbitrary \((2 \times 3)\) matrix, such as:

\[ A = \begin{bmatrix} a_{00} & a_{01} & a_{02} \\ a_{10} & a_{11} & a_{12} \end{bmatrix} \]

Parallel lines remain parallel under affine transformations.
2D PROJECTIVE TRANSFORMATIONS

This transformation, also known as perspective transformation or homography, operates solely on homogeneous coordinates:

\[ \tilde{x}' = \tilde{H} \tilde{x} \]

Where \( \tilde{H} \) is an arbitrary \((3 \times 3)\) matrix. \( \tilde{H} \) is an homogeneous matrix, which means that it is defined up to a scale and two \( \tilde{H} \) matrices with different scales are equivalent. Note that \( \tilde{H} \) operates only on homogeneous coordinate.
SAMPLE TRANSFORMATION MATRICES

\[ T = \begin{bmatrix} 1 & 0 & dx \\ 0 & 1 & dy \\ 0 & 0 & 1 \end{bmatrix} \]  
(translation)

\[ R = \begin{bmatrix} \cos \theta & -\sin \theta & dx \\ \sin \theta & \cos \theta & dy \\ 0 & 0 & 1 \end{bmatrix} \]  
(rigid)

\[ R = \begin{bmatrix} s \cos \theta & -s \sin \theta & dx \\ s \sin \theta & s \cos \theta & dy \\ 0 & 0 & 1 \end{bmatrix} \]  
(similarity)
EXAMPLES

Given a point \( \mathbf{x} \) and a transformation matrix \( \mathbf{A} \), the projection of the point is performed through the product: \( \mathbf{x}' = \mathbf{Ax} \):

\[
\mathbf{Ax} = \begin{bmatrix}
1 & 0 & dx \\
0 & 1 & dy \\
0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
x \\
y \\
1
\end{bmatrix}
= \begin{bmatrix}
x + dx \\
y + dy \\
1
\end{bmatrix}
\]

(translation)

\[
\mathbf{Ax} = \begin{bmatrix}
\cos \theta & -\sin \theta & dx \\
\sin \theta & \cos \theta & dy \\
0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
x \\
y \\
1
\end{bmatrix}
= \begin{bmatrix}
x \cos \theta - y \sin \theta + dx \\
x \sin \theta + y \cos \theta + dy \\
1
\end{bmatrix}
\]

(rigid)
2D TRANSFORMATIONS

The basic transformations can be easily handled as the product between a transformation matrix and the homogenous point vector.

The $2\times3$ matrices in the diagram can be converted to $3\times3$ matrices adding a $[0 \ 0 \ 1]$ row at the bottom in order to obtain homogeneous results.

$$R(\theta) = \begin{pmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{pmatrix}$$
A 3D point $x$ is translated by vector $t$ as:

$$x' = x + t$$

This is equivalent to:

$$x' = [I \quad t] \bar{x}$$

where $I$ is the $(3 \times 3)$ identity matrix and $\bar{x}$ is the original point in homogeneous coordinates. If we wish to obtain an homogeneous result:

$$\bar{x}' = \begin{bmatrix} I & t \\ 0^T & 1 \end{bmatrix} \bar{x}$$

where $0$ is the $(3 \times 1)$ zero vector. The translations preserve line orientations.
3D ROTATIONS + TRANSLATIONS (EUCLIDEAN)

The transformation can be written as:

\[ x' = Rx + t \]

where \( R \) is a \( (3 \times 3) \) rotation matrix defined as shown in the next slide. In matrix form:

\[ x' = [R \quad t] \bar{x} \]

\[ \bar{x}' = \begin{bmatrix} R & t \\ 0^T & 1 \end{bmatrix} \bar{x} \]
### 3D Rotations

The rotations of 3D points can be performed multiplying by $3 \times 3$ rotation matrices:

- **Rotation about the x axis**
  $$R_x(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \theta & -\sin \theta \\ 0 & \sin \theta & \cos \theta \end{bmatrix}$$

- **Rotation about the y axis**
  $$R_y(\theta) = \begin{bmatrix} \cos \theta & 0 & \sin \theta \\ 0 & 1 & 0 \\ -\sin \theta & 0 & \cos \theta \end{bmatrix}$$

- **Rotation about the z axis**
  $$R_z(\theta) = \begin{bmatrix} \cos \theta & -\sin \theta & 0 \\ \sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

It is also possible to rotate a point by angle $\theta$ about the unit vector $\mathbf{u} = (u_x, u_y, u_z)$:

$$R = \begin{bmatrix} \cos \theta + u_x^2(1 - \cos \theta) & u_xu_y(1 - \cos \theta) - u_z\sin \theta & u_xu_z(1 - \cos \theta) + u_y\sin \theta \\ u_yu_x(1 - \cos \theta) + u_z\sin \theta & \cos \theta + u_y^2(1 - \cos \theta) & u_yu_z(1 - \cos \theta) - u_x\sin \theta \\ u_zu_x(1 - \cos \theta) - u_y\sin \theta & u_zu_y(1 - \cos \theta) + u_x\sin \theta & \cos \theta + u_z^2(1 - \cos \theta) \end{bmatrix}$$
2D SCALED ROTATION (SIMILARITY)

The transformation can be written as:

\[ x' = sRx + t \]

In matrix form:

\[
\begin{bmatrix}
  x' \\
  \bar{x}'
\end{bmatrix}
= \begin{bmatrix}
  sR & t \\
  0 & 1
\end{bmatrix}
\begin{bmatrix}
  x \\
  1
\end{bmatrix}
\]
The affine transformation is written as:

\[ \mathbf{x}' = A \mathbf{x} \]

where \( A \) is an arbitrary \((3 \times 4)\) matrix, such as:

\[
A = \begin{bmatrix}
a_{00} & a_{01} & a_{02} & a_{03} \\
a_{10} & a_{11} & a_{12} & a_{13} \\
a_{20} & a_{21} & a_{22} & a_{23}
\end{bmatrix}
\]
3D PROJECTIVE TRANSFORMATIONS

This transformation, also known as perspective transformation or homography, operates solely on homogeneous coordinates:

\[ \tilde{x}' = \tilde{H}\tilde{x} \]

Where \( \tilde{H} \) is an arbitrary \((4 \times 4)\) matrix. \( \tilde{H} \) is an homogeneous matrix, which means that it is defined up to a scale and two \( \tilde{H} \) matrices with different scales are equivalent. Note that \( \tilde{H} \) operates only on homogeneous coordinate.
# 3D Transformations

<table>
<thead>
<tr>
<th>Transformation</th>
<th>Matrix</th>
<th># DoF</th>
<th>Preserves</th>
<th>Icon</th>
</tr>
</thead>
<tbody>
<tr>
<td>translation</td>
<td>$\begin{bmatrix} I &amp; t \end{bmatrix}_{3\times4}$</td>
<td>3</td>
<td>orientation</td>
<td></td>
</tr>
<tr>
<td>rigid (Euclidean)</td>
<td>$\begin{bmatrix} R &amp; t \end{bmatrix}_{3\times4}$</td>
<td>6</td>
<td>lengths</td>
<td></td>
</tr>
<tr>
<td>similarity</td>
<td>$\begin{bmatrix} sR &amp; t \end{bmatrix}_{3\times4}$</td>
<td>7</td>
<td>angles</td>
<td></td>
</tr>
<tr>
<td>affine</td>
<td>$\begin{bmatrix} A \end{bmatrix}_{3\times4}$</td>
<td>12</td>
<td>parallelism</td>
<td></td>
</tr>
<tr>
<td>projective</td>
<td>$\begin{bmatrix} \tilde{H} \end{bmatrix}_{4\times4}$</td>
<td>15</td>
<td>straight lines</td>
<td></td>
</tr>
</tbody>
</table>
IMAGE FORMATION

Pinhole Model
Perspective Camera
Wide Angle Cameras
The image is formed when the light rays hit the objects and are reflected to a photosensitive surface:

Each light ray is refracted everywhere on the surface.
The image is formed when the light rays hit the objects and are reflected to a photosensitive surface:
PINHOLE MODEL

The pinhole model allows to "control" the direction of the light rays.

However, it is impractical for real applications: no zoom, no focus, requires high sensitivity.
NATURAL PINHOLE

The effect of pinhole cameras can be observed at this wall opposite some balistrarias in the Castelgrande of Bellinzona. You can see the red roofs of the houses and the green trees that lie behind the balistrarias being projected onto the wall.
THIN LENSES

A thin lens is a lens which thickness is negligible compared to the radii of curvature of the two surfaces of the lens:

Each thin lens is characterized by:
• the focal length \( f \);
• the two focii \( F_1 \) and \( F_2 \).

\[
d << R_1 \land d << R_2
\]
THIN LENSES

The thin lenses implement the pinhole model and can be used to "control" the direction of the incoming light rays in order to form a sharp image on the sensor, the same way the eye lens allows to obtain a sharp image on the retina.
PROPERTIES OF THIN LENSES

The thin lenses have the following properties:

- Any ray that enters parallel to the axis on one side of the lens proceeds towards the focal point F on the other side.

- Any ray that arrives at the lens after passing through the focal point on the front side, comes out parallel to the axis on the other side.

- Any ray that passes through the center of the lens will not change its direction.
The thin lenses equation can be derived directly from the three properties already discussed.

\[
\frac{1}{\tilde{Z}'} + \frac{1}{\tilde{z}'} = \frac{1}{f}
\]

The equation is proved considering similar triangles.
CONSIDERATIONS

\[ \frac{1}{\tilde{Z}} + \frac{1}{\tilde{z}} = \frac{1}{f} \]

- The "level of zoom" depends on the focal length;
- The image focus depends on the distance between the points in the real world and the lens:
  - only points which distance is equals to \( \tilde{Z} \) will be focused.
In the considered models (pinhole cameras, thin lenses and human eyes), the image is formed through an optical process which is determined by the physical properties of the model;

The image formation process can be simply modeled as the projection of a 3D scene to a 2D plane \( \pi \);

Hence we consider a \textit{simplified projection model} (the perspective camera model) where the 3D coordinate system of the scene and the 2D coordinate system of the camera are somehow aligned. In particular we assume that:

- the origin of both the camera and the real world coordinate systems are in the same point, which we call the \textit{projection center};
- the \textit{image plane} \( \pi \) is perpendicular to the Z axis (parallel to the XY plane);
- the distance between \( \pi \) and the origin of the Cartesian system is equal to the focal length of the lens \( f \);

We will study how to model the mapping between a 3D point \( P \equiv (X,Y,Z) \) in the scene and a 2D point \( p \equiv (x,y) \) in the image plane.
**PERSPECTIVE CAMERA MODEL**

- **Projection Center**: The point from which the projection occurs.
- **Image Plane**: The plane onto which the projection is made.
- **Image Point**: The point on the image plane corresponding to a real world point.
- **Real World Point**: The point in the 3D space being projected.

Mathematically:
- \( p \equiv (x, y) \) represents the image point.
- \( P \equiv (X, Y, Z) \) represents the real world point.

The projection process maps real world points to image points using a perspective projection matrix.
The fundamental equations of the perspective camera are derived considering similar triangles:

\[ x = f \cdot \frac{X}{Z} \quad y = f \cdot \frac{Y}{Z} \]
WEAK PERSPECTIVE CAMERA MODEL

For different 3D points, the foundamental equations \( x = f \cdot \frac{X}{Z} \) and \( y = f \cdot \frac{Y}{Z} \) are not linear, since they refer to different \( Z \) values.

If the 3D points of the scene are far enough from the camera, however, their \( Z \) values are large and their differences are negligible, so they can be approximated by the average value \( \tilde{Z} \) in order to obtain linear equations:

\[
x \approx f \cdot \frac{X}{\tilde{Z}} \quad \quad y \approx f \cdot \frac{Y}{\tilde{Z}}
\]

where \( \tilde{Z} \approx Z \)
POLAR AND SPHERICAL COORDINATES

It is often convenient to consider the projections in polar form. To do this, the 2D points in the image have to be expressed in polar coordinates and the 3D points of the real world have to be expressed in spherical coordinates.
POLAR COORDINATES

The 2D point in cartesian coordinates \((x, y)\) is mapped to the point in the polar coordinates \((\rho, \varphi)\), where \(\rho = \sqrt{x^2 + y^2}\) and \(\varphi = \text{atan2}(y, x)\).

\[
\text{atan2}(y, x) = \begin{cases} 
\arctan \frac{y}{x} & x > 0 \\
\arctan \frac{y}{x} + \pi & y \geq 0, x < 0 \\
\arctan \frac{y}{x} - \pi & y < 0, x < 0 \\
\frac{\pi}{2} & y > 0, x = 0 \\
-\frac{\pi}{2} & y < 0, x = 0 \\
\text{undefined} & y = 0, x = 0
\end{cases}
\]
SPHERICAL COORDINATES

The 3D point in cartesian coordinates \((x, y, z)\) is mapped to the point in the spherical coordinates \((\rho, \theta, \varphi)\), where

\[
\rho = \sqrt{x^2 + y^2 + z^2}, \\
\varphi = \text{atan2}(y, x) \\
\theta = \arccos\left(\frac{z}{\sqrt{x^2 + y^2 + z^2}}\right).
\]

\[
\text{atan2}(y, x) = \begin{cases} 
\arctan \frac{y}{x} & x > 0 \\
\arctan \frac{y}{x} + \pi & y \geq 0, x < 0 \\
\arctan \frac{y}{x} - \pi & y < 0, x < 0 \\
+\frac{\pi}{2} & y > 0, x = 0 \\
-\frac{\pi}{2} & y < 0, x = 0 \\
\text{undefined} & y = 0, x = 0 
\end{cases}
\]
The fundamental equations of the perspective camera model can be rewritten considering the polar and spherical coordinate systems. In this context, a 3D point $P \equiv (\rho, \theta, \varphi)$ is projected to the 2D point $p \equiv (\rho', \varphi)$, where:

$$\rho' = f \cdot \tan(\theta)$$


**PERSPECTIVE CAMERA MODEL (POLAR 3D VERSION)**

\[ (\rho, \theta, \varphi) \to (\rho', \varphi) \quad \text{where} \quad \rho' = f \cdot \tan(\theta) \]
FIELD OF VIEW (FOV)

Given a camera, it is possible to measure its field of view, which is the solid angle through which it is sensitive to electromagnetic radiation. It can be measured in three directions: horizontal, vertical, and diagonal.
Despite its simplicity, the perspective camera model can be successfully used to model the image formation process of the class of the perspective cameras. This class includes most conventional cameras, e.g., consumer digital (and analogical) cameras, mobile phone cameras, webcams, etc.;

However, the perspective camera model is suitable to describe only narrow FOV cameras: theoretically under 90°, but up to a maximum of 140° with some deviation from the model (i.e., image distortion);

It should be noted the the human binocular FOV is approximately 180° horizontally and 120° vertically;

Many real world applications fields (e.g., surveillance, automotive, robotics) require the use of wide angle cameras with a large FOV up to 180°.
WIDE ANGLE CAMERAS

Since the perspective projection is not suitable to model wide angle cameras (it would project points forming an angle with the optical axis close to 180° at infinity!), the manufacturers usually follow different projection schemes. Two main approaches are possible:

- Using a different lens (e.g., fisheye lenses);
- Using a curved mirror in order to deviate the light rays prior to acquiring them with a perspective camera.

The former approach defines the class of dioptric systems, while the latter defines the class of catadioptric systems.
WIDE ANGLE CAMERAS: DESIGNS

- catadioptric
- dioptic (fisheye)
Since it is not possible to form an image of an emispheric field on a plane without distortion, all wide angle system have inherent distortion. The distortion have the effect to compress the space non-linearly at the borders, ending up in a radial distortion symmetric with respect to the principal point (the interception of the optical axis with the image plane).
FISHEYE PROJECTION FUNCTIONS

\[ \rho = f \tan \theta \]  
(perspective projection)

\[ \rho = 2f \tan(\theta/2) \]  
(stereographic projection)

\[ \rho = f \theta \]  
(equidistance projection)

\[ \rho = 2f \sin(\theta/2) \]  
(equisolid angle projection)

\[ \rho = f \sin \theta \]  
(orthogonal projection)
WIDE ANGLE VS PERSPECTIVE PROJECTION

position according to perspective projection

“distorted” position according to a wide angle projection
DISTORTION COMPENSATION

If the projection function of a given wide angle camera is known (in the form of one of the functions shown in the previous slides), it is possible to compensate the geometrical distortion undergone by the acquired image, through a process often called "rectification".

The output image has the properties of an image acquired by a perspective camera (i.e., straight lines in the real world are mapped to straight lines in the image) but an interpolation process is necessary to recover the missing information.

The rectification is performed defining a mapping between the distorted points \((x, y)\) and their undistorted counterparts \((u, v)\). In practice the points \((x, y)\) are "moved" to their original position in the undistorted image plane.

Since the distortion is radially symmetric with respect to the centre of distortion, the distorted point \((\rho, \varphi)\) should be mapped to a point laying on the same direction \(\varphi\) but having larger radius \(\rho'\):

\[(\rho, \varphi) \mapsto (\rho', \varphi)\]
DISTORTION COMPENSATION (2)

The mapping between $\rho$ and $\rho'$ is obtained solving the projection function of the wide angle camera and the function of the standard perspective projection for $\theta$. For instance, if our wide angle camera follows the equidistance projection, then we have to solve the following two equations for $\theta$:

\[ \rho' = f' \cdot \tan \theta \quad (1) \text{ (perspective)} \]
\[ \rho = f \cdot \theta \quad (2) \text{ (equidistance)} \]

Solving (2) for $\theta$ we obtain: $\theta = \frac{\rho}{f}$. Substituting this expression in (1), we get:

\[ \rho' = f' \cdot \tan \left( \frac{\rho}{f} \right) \]

The mapping function is defined as:

\[ (\rho, \varphi) \mapsto (f' \cdot \tan \left( \frac{\rho}{f} \right), \varphi) \]
DISTORTION COMPENSATION (3)

The expression in the previous slide is equivalent to a vectorial mapping function $f$ between the distorted points $(x, y)$ and their undistorted counterparts $(u, v)$:

$$(u, v) = f(x, y)$$

Once this function is known, the undistorted image $I^u$ is obtained from the distorted image $I$ remapping the point coordinates:

$$I^u(u, v) = I(f(u, v))$$

Due to the effects of the distortion, a regular grid of distorted $(x, y)$ coordinates does not correspond to a regular grid of undistorted $(u, v)$ coordinates and hence the interpolation is required.

In the attempt to "recover" the lost information, the interpolation process introduces some artifacts, in the form of blurred regions.
DISTORTION COMPENSATION (4)

wide angle image

rectified image
DISTORTION COMPENSATION (4)

wide angle image  

rectified image  
(without interpolation)
EXERCISE

Write a MATLAB function that takes in input a rectilinear image $I$ and returns a distorted image $\hat{I}$ according to one of the discussed models.

Hints:

- Establish a mapping between the distorted and the undistorted coordinates;
- Consider converting cartesian coordinates to polar ones;
- Set $f = f'$ to an arbitrary value;
- Use meshgrid to speed up the computation;
- Use function "interp2" to perform the interpolation;
FROM THE WORLD TO THE CAMERA

Coordinate Systems
Extrinsic Parameters
Intrinsic Parameters
Coordinate Mapping
So far we have assumed that the real world coordinate system is aligned with the camera coordinate system;

Accordingly we have modeled the mapping between the real world 3D points and the corresponding 2D points in the image plane as a simple projection;

In most of the applications it is convenient to refer to the 3D points in a coordinate system independent from the one of the observer. So we need to know how to project the real world 3D points in metric coordinates to the 2D image points in pixel coordinates;

The projection function depends on many parameters which are specific to the used camera and to the scene. The parameters can be found through a process called geometric camera calibration;

In the next slides we will define the projection function and discuss its parameters.
We consider four coordinate systems (CS):

1. The 3D coordinate system of the **real world** in metric units (e.g., millimeters);

2. The 3D coordinate system of the **camera** in metric units (the observer’s viewpoint);

3. The 2D coordinate system of the **image plane** (**ideal camera plane**) in metric units;

4. The 2D coordinate system of the **camera plane** (the final image) in pixles;
1) THE REAL WORLD CS
2) THE CAMERA CS (STILL 3D!)
THE CAMERA VIEW
THE FINITE IMAGE
FINITE IMAGE + 1) REAL WORLD CS
3) IDEAL IMAGE (IDEAL CAMERA) CS
4) THE FINAL IMAGE CS
A 3D point in the scene is mapped to a 2D point in the image in three steps:

I. The 3D point is projected from the coordinate system of the real world to the 3D coordinate system of the camera;

II. The 3D point in the coordinate system of the camera is projected to the 2D coordinate system of the image plane;

III. The 2D point is projected from the image plane in metric units to the camera plane in pixels.
STEP 1: ALIGNING THE COORDINATE SYSTEMS

ESTRINSIC PARAMETERS

In order to “align” the coordinate system of the **real world** to the one of the **image plane**, we need to perform a **translation** followed by a **rotation**. So we need:

- a rotation matrix $R$ (3 angles = 3 DoF);
- a translation vector $T$ (3 components = 3 DoF);

The 3D points $P_W$ in the real world, are hence mapped to the 3D points $P_C$ in the coordinate system of the camera using the equation:

$$P_C = R(P_W - T)$$

The 6 degrees of freedom of the matrices $R$ and $T$ constitute the **extrinsic parameters** of the camera. Such parameters actually depend on the coordinate system we chose in the real world and hence are usually considered "**scene dependent**" parameters.
STEP 2: PROJECTION TO THE IDEAL IMAGE PLANE

Now that the real world points are expressed in the coordinate system of the camera, we can apply the perspective camera model and project the points using the fundamental equations of the perspective camera:

\[ x = f \cdot \frac{X}{Z}; \quad y = f \cdot \frac{Y}{Z} \]

For instance, given the point in the camera coordinate system \( P_C \equiv (X, Y, Z) \), we get the 2D point:

\[ p \equiv (x, y) = (f \cdot \frac{X}{Z}, f \cdot \frac{Y}{Z}) \]
IDEAL IMAGE PLANE VS CAMERA PLANE

The differences between the **ideal image plane** and the **camera plane** are the following:

<table>
<thead>
<tr>
<th>Plane</th>
<th>Units</th>
<th>Centre of the CS</th>
<th>Distortion</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ideal Image Plane</td>
<td>metric (e.g., millimeters)</td>
<td>principal point</td>
<td>no distortion</td>
</tr>
<tr>
<td>Image Plane</td>
<td>pixels</td>
<td>top left corner</td>
<td>radial distortion</td>
</tr>
</tbody>
</table>

![Image Diagram]
The **intrinsic parameters** are characteristic of the camera and define how to perform the projection from the real world to the final image and how to map the **ideal image plane** to the **camera plane** (the sensor array!). The intrinsic parameters are:

- **focal length** $f$;
- coordinates of the **principal point** in the camera plane coordinate system: $o_x$ and $o_y$;
- **scale factors** $s_x$ and $s_y$ which express the relationship between the metric units (e.g., millimeters) and the pixels. These parameters represent the size of a pixel in metric units;
- **radial distortion** parameters $k_1$ and $k_2$ to model the deviation from the perspective camera model.
STEP 3: PROJECTION TO THE CAMERA PLANE

Let \((x_{im}, y_{im})\) be the pixel coordinates (camera plane) of point \((x, y)\) in the image plane. The following equations hold:

\[
x_{im} = \frac{x}{s_x} + o_x \Rightarrow x = -(x_{im} - o_x) \cdot s_x
\]

\[
y_{im} = \frac{y}{s_y} + o_y \Rightarrow y = -(y_{im} - o_y) \cdot s_y
\]

Real cameras are subject to distortion caused by the deviation from the ideal model. The distortion is modeled with the following formulas:

\[
x = x_d (1 + k_1 r^2 + k_2 r^4)
\]

\[
y = y_d (1 + k_1 r^2 + k_2 r^4)
\]

where \(r^2 = x_d^2 + y_d^2\) are the distorted coordinates corresponding to point \((x, y)\) and \(r^2 = x_{im}^2 + y_{im}^2\).
FROM THE WORLD TO THE CAMERA: PARAMETERS

The projection from the world to the camera finally depends on the following parameters:

**extrinsic**
- T: translation vector;
- R: rotation matrix;

**intrinsic**
- f: focal length;
- \( o_x, o_y \): coordinates of the principal point in the camera plane;
- \( s_x, s_y \): dimensions of the pixels in millimeters;
- \( k_1, k_2 \): distortion parameters.
The final mapping can be defined in formulas using equations:

\[ x = f \cdot \frac{X}{Z}, y = f \cdot \frac{Y}{Z} \quad (1) \]
\[ P_c = R(P_W - T) \quad (2) \]
\[ x = -(x_{im} - o_x)s_x \quad (3) \]
\[ y = -(y_{im} - o_y)s_y \quad (3) \]

Equating (1) and (3) we get:

\[ -(x_{im} - o_x)s_x = f \frac{X}{Z} \quad (4) \]
\[ -(y_{im} - o_y)s_y = f \frac{Y}{Z} \]

where \( P_c \equiv (X, Y, Z) \).
The rotation matrix $R$ in equation (2) can be denoted as:

$$R = \begin{pmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{pmatrix} = \begin{pmatrix} R_1 \\ R_2 \\ R_3 \end{pmatrix}$$

where $R_i$ is the $i$-th row of the matrix. Equation (2) can be then expressed as:

$$p_c = \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \begin{pmatrix} R_1(P_W - T)^T \\ R_2(P_W - T)^T \\ R_3(P_W - T)^T \end{pmatrix}$$

Substituting such expression for $X$, $Y$ and $Z$ equation (4), we get:

$$-(x_{im} - o_x)s_x = f \frac{R_1(P_W - T)^T}{R_3(P_W - T)^T}$$

$$-(x_{im} - o_y)s_y = f \frac{R_2(P_W - T)^T}{R_3(P_W - T)^T}$$
If we know all the parameters, we can perform the projection (apart from distortion) using the formulas:

\[ x_{im} = o_x - \frac{f}{s_x} \frac{R_1 (P_W - T)^T}{R_3 (P_W - T)^T}, \]

\[ y_{im} = o_y - \frac{f}{s_y} \frac{R_2 (P_W - T)^T}{R_3 (P_W - T)^T}. \]
COORDINATE MAPPING: MATRIX FORM

Let us define the matrices of the intrinsic and extrinsic parameters:

\[
P_C = R (P_W - T)
\]

\[
M_{int} = \begin{pmatrix}
-\frac{f}{s_x} & 0 & o_x \\
0 & -\frac{f}{s_y} & o_y \\
0 & 0 & 1
\end{pmatrix}; M_{ext} = \begin{pmatrix}
r_{11} & r_{12} & r_{13} & -R_{1T}^T \\
r_{21} & r_{22} & r_{23} & -R_{2T}^T \\
r_{31} & r_{32} & r_{33} & -R_{3T}^T
\end{pmatrix}
\]

The mapping operation is defined in matrix form as:

\[
\begin{pmatrix}
x \\
y \\
w
\end{pmatrix} = M_{int} M_{ext} \begin{pmatrix}
X_W \\
Y_W \\
Z_W
\end{pmatrix}
\]

\[
M = M_{int} M_{ext}
\]

is known as "camera matrix".
QUESTION TIME
CONTACTS

For any doubts feel free to contact me:

- furnari@dmi.unict.it
- Room 30;

Slides available at:

- My personal page:
  - http://www.dmi.unict.it/~furnari
- Studium course page