University of Bologna

Computer Vision Laboratory

Image Formation and Acquisition (Part 2)

Luigi Di Stefano (luigi.distefano@unibo.it)

Perspective Projection REVIEW!

CAMERA REFERENCE FRAME (CRF)(3D): solidal\attached to the camera, origin on camera focal point, axis z coincidend with the optical axis!

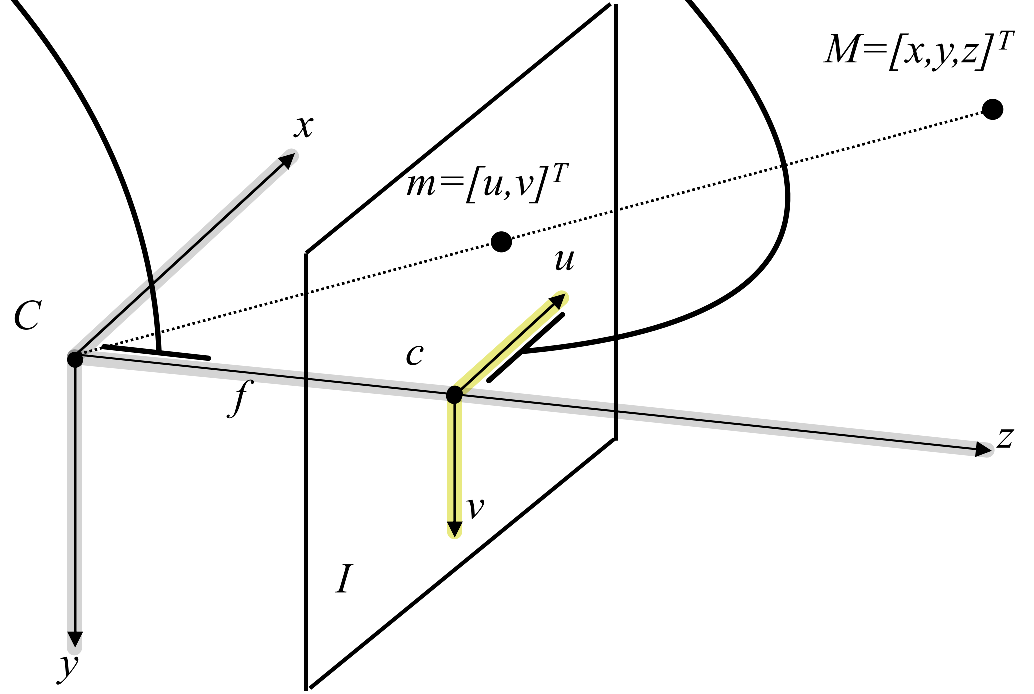
IMAGE REFERENCE FRAME (IRF)(2D): origin on the optical axis at a distance f from C, axis x and y orthogonal to the optical axis itself and paraller to ones of the CRF!

- Let us consider
  - A point in the 3D space, $M=[x,y,z]^T$, with coordinates given in the Camera Reference Frame (CRF).
  - Its projection onto the image plane I, denoted as $m=[u,v]^T$
- The **non-linear** equations providing image coordinates as a function of the 3D coordinates in the CRF are as follows:

$$\begin{cases} u = \dfrac{f}{z}x \\ v = \dfrac{f}{z}y \end{cases}$$

BE AWARE: theese NL equations hold ONLY in the euclidian space AND ONLY with points expressed in CRF and IRF!

Perspective projection equations indeed works only with points expressed in CRF and IRF!



Projective Space

- The physical space is a 3D Euclidean Space (**R**^{3}) whose points can be represented as 3D vectors in a given reference frame.
  - In this space parallel lines do not intersect, or intersect "at infinity ".
  - Points at infinity cannot be represented in this vector space.

In euclidian space we know that points at infinity exist BUT we do not have a way to represent them properly! On the contrary, in Projective Spaces this representation is possible! !

- Let's now append one more coordinate to our Euclidean triples, so that e.g.

$$\begin{pmatrix} x & y & z \end{pmatrix} \text{ becomes } \begin{pmatrix} x & y & z & 1 \end{pmatrix}$$

and assume that both vectors are correct representations of the same 3D point.

- Moreover, we do not constrain the 4th coordinate to be 1 but instead assume

$$\begin{pmatrix} x & y & z & 1 \end{pmatrix} \equiv \begin{pmatrix} 2x & 2y & 2z & 2 \end{pmatrix} \equiv \begin{pmatrix} kx & ky & kz & k \end{pmatrix} \forall k \neq 0$$

To return back to euclidian coordinates from projective ones we simply divide by k and then we get rid of the final one!

$\begin{bmatrix} x \\ y \\ z \end{bmatrix} \in \mathbb{R}^3 \quad k\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \in P^3, k \neq 0$

- In this representation a point in space is represented by an **equivalence class** of **quadruples**, wherein equivalent quadruples differ just by a multiplicative factor.
- This is the so-called **homogeneous coordinates** (a.k.a. projective coordinates) representation of the 3D point having Euclidean coordinates (*x*, *y*, *z*). The space associated with the homogeneous coordinates representation is called **Projective Space**, denoted as **P**^{3}. → the 3dimensional projective space!
- Extension to Euclidean spaces of any other dimension is straightforward (**R**^{n} → **P**^{n})

Point at infinity of a 3D line

Let's consider the parametric equation of a 3D line:

$$\mathrm{M} = M_0 + \lambda D = \begin{bmatrix} x_0 \\ y_0 \\ z_0 \end{bmatrix} + \lambda \begin{bmatrix} a \\ b \\ c \end{bmatrix} = \begin{bmatrix} x_0 + \lambda a \\ y_0 + \lambda b \\ z_0 + \lambda c \end{bmatrix}$$

! A direction vector! A very common one is the direction cosines vector, AKA vector of cosines on angles between the 3D line and the 3 main directions of the ref. frame!

and represent the generic point along the line in projective coordinates:

$$\widetilde{M} = \begin{bmatrix} M \\ 1 \end{bmatrix} = \begin{bmatrix} x_0 + \lambda a \\ y_0 + \lambda b \\ z_0 + \lambda c \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{x_0}{\lambda} + a \\ \frac{y_0}{\lambda} + b \\ \frac{z_0}{\lambda} + c \\ \frac{1}{\lambda} \end{bmatrix}$$

by taking the limit with $\lambda \to \infty$ we obtain the projective coordinates of the point at infinity of the given line:

$$\widetilde{M}_\infty = \lim_{\lambda \to \infty} \widetilde{M} = \begin{bmatrix} a \\ b \\ c \\ 0 \end{bmatrix}$$

BE AWARE: 3D euclidian lines that we bring in the projective space have each one only ONE single point at infinity!

The projective coordinates of the point at infinity of a 3D line are obtained by taking any Euclidean vector parallel to the line and appending a 0 as fourth coordinate. There exist infinitely many points at infinity in **P**^{3}, as many as the directions of the 3D lines.

Points at infinity

- The points of the 3D Projective Space **having the fourth coordinate equal to 0** , e.g. (x, y, z, 0), are the points at infinity of the 3D lines. These points cannot be represented in the 3D Euclidean Space.
- Indeed, to map such points into the Euclidean Space we would divide by the –null- fourth coordinate, so as to get (x/0, y/0, z/0), i.e. infinite coordinates, which is not a valid representation in the Euclidean Space.
- By the homogenous coordinates it is therefore possible to represent and process seamlessly both ordinary points as well as *points at infinity.*
- Point (0, 0, 0, 0) is undefined.
  - indeed, the above point is NOT the origin of the Euclidean Space (0, 0, 0), for such point is represented in homogeneous coordinates as (0, 0, 0, *k*), *k*≠0.
- It can be shown that all points at ininity of $\mathbf{P}^3$ lie on a plane, which is called the **plane at infinity**.

Recap

- Any Euclidean Space $\mathbf{R}^n$ can be extended to a corresponding **Projective Space** $\mathbf{P}^n$ by representing points in homogeneous coordinates.
- The projective representation includes one additional coordinate, referred to here as *k*, wrt the Euclidean representation:
  - $k \neq 0$ denotes point existing in $\mathbf{R}^n$, their coordinates given by x_i/k, $i = 1 \ldots n$
  - $k = 0$ denotes points at infinity (a.k.a. ideal points) in $\mathbf{R}^n$, which do not admit a representation via Euclidean coordinates.
- The Projective Space allows then to represent and process homogeneously (i.e. without introduction of exceptions or special cases) both the ordinary and the ideal points of the Euclidean Space.
- Why are we interested in Projective Spaces ?

The one big advantage of perspective coords is making prospective projection governed by LIN MATRIX relations!

Because Perspective Projection is more conveniently dealt with using projective coordinates as it becomes a linear transformation !

Perspective Projection in projective coordinates (1)

- We already know that there exist a non-linear transformation between 3D coordinates and image coordinates:

$$u = \frac{f}{z}x \qquad v = \frac{f}{z}y$$

- Let's now go back to our 3D point *M* (with coordinates expressed in the CRF) and its projection onto the image plane, *m*:

$$\boldsymbol{M} = [x,y,z\,]^T \qquad \boldsymbol{m} = [u,v\,]^T$$

and represent both points in homogenous coordinates (denoted by symbol ~)

$$\tilde{\mathbf{m}} = \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \qquad \tilde{\mathbf{M}} = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

BE AWARE: with the tilda symbol ~ we indicate quantities that are expressed in P^3 and that rely on homogeneous coords usage!

Perspective Projection in projective coordinates (2)

- In homogeneous coordinates (hence considering the mapping between projective spaces), perspective projection becomes a **linear transformation**:

$$\overbrace{\begin{bmatrix} u \\ v \\ 1 \end{bmatrix}}^{\tilde{m}} = \begin{bmatrix} f\dfrac{x}{z} \\ f\dfrac{y}{z} \\ 1 \end{bmatrix} = \begin{bmatrix} fx \\ fy \\ z \end{bmatrix} = \overbrace{\begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}}^{\tilde{P}} \overbrace{\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}}^{\tilde{M}}$$

- In matrix notation

that k means: up to a constant multiplication!

$$k\tilde{\mathbf{m}} = \tilde{\mathbf{P}}\tilde{\mathbf{M}}$$

often expressed also as below, where $\approx$ means "equal up to an arbitrary scale factor"

$$\tilde{\mathbf{m}} \approx \tilde{\mathbf{P}}\tilde{\mathbf{M}}$$

Vanishing Point of a 3D line

this is the projection of the point at infinity of parallel 3D lines (with direction a,b,c) onto the image plane!

$$\begin{bmatrix} fa \\ fb \\ c \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ 0 \end{bmatrix}$$

$$\begin{bmatrix} fa \\ fb \\ c \end{bmatrix} \rightarrow \begin{bmatrix} f\frac{a}{c} \\ f\frac{b}{c} \end{bmatrix}$$

If we want to get the euclidian coordinates of the obtained 2D point in the image plane!

This is the actual computation to do in order to compute the euclidian coords of a vanishing point!

Vanishing point: particular cases!

The vanish point of 3D lines parallel to the optical axis is the origin of the IRF, indeed:

$$\begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

The vanish point of 3D lines parallel to the image plane is onto the image plane BUT is at infinity, indeed:

$$\begin{bmatrix} fa \\ fb \\ 0 \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} a \\ b \\ 0 \\ 0 \end{bmatrix}$$

Perspective Projection Matrix (PPM)

- Matrix $\tilde{\mathbf{P}}$ represents the geometric camera model and is known as **Perspective Projection Matrix (PPM).**

Indeed, project coords. advantages are ensuring eqs. linearity AND highlighting cameras canonical behaviour!

$x_F = x/f \quad y_F = y/f \quad z_F = z/f$ (f as units of measure)

- If we assume distances to be measured in focal lenght units (f = 1), the PPM becomes

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \overbrace{[I\,|\,0]}^{\tilde{P}} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \text{ proj. coords}$$

to euclidian coords $\begin{bmatrix} x/z \\ y/z \end{bmatrix}$

$$\tilde{\mathbf{P}} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} = [\mathbf{I}\,|\,\mathbf{0}]$$

This is the true canonical nature of all cameras and of prospective projection in a nutshell: SCALING 3D coordinates ONLY depending on DEPTH! Then, we add camera-specific features such as it is, for example, the concept of focal lenght!

- This form is useful to understand the core operation carried out by perspective projection, which is indeed scaling lateral coordinates (i.e. x,y) according to the distance from the camera (z). The actual focal length just introduces an additional, fixed (i.e. independent of z) scaling factor of projected coordinates. The above form is usually referred to as ***canonical*** or ***standard*** **PPM**.

RECALL: perspective projection applyes only when considering CRF and IRF!

A more comprehensive camera model

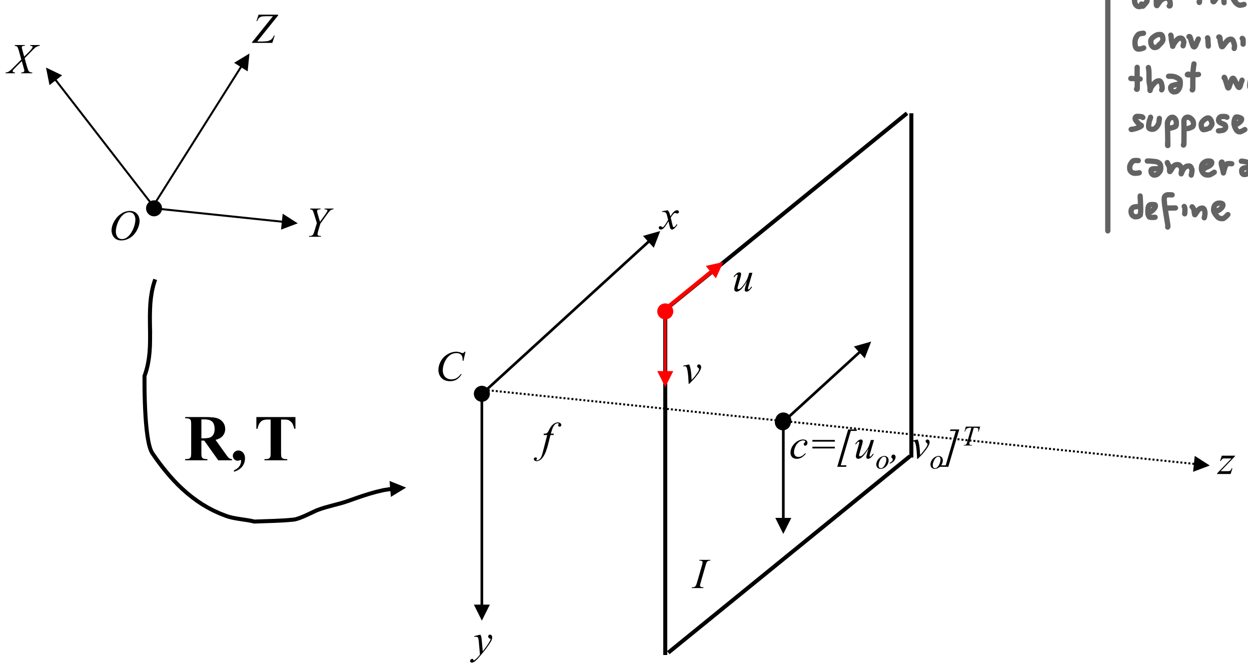
LET'S NOW IMPROVE OUR MODEL with some camera-specific properties!

- To come up with a really useful camera model we need to take into account two additional issues:
  - Image Digitization;
  - The 6 DOF (3D rotation and translation) rigid motion between the Camera Reference Frame (CRF) and the World Reference Frame (WRF).

Talking about ROTOTRASLATION!

Typically the CRF is NOT available; on the other hand, once we have conviniently defined a 3D WRF that we know well and that is supposed to be incorporated in the camera model, we can very easily define 3D points w.r.t. it!

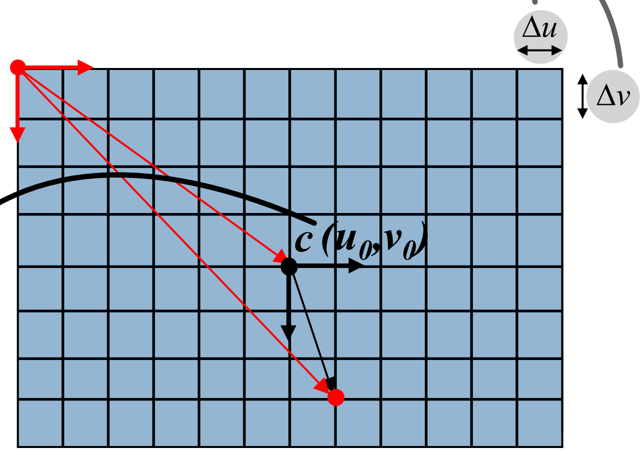
until now we always referred to image coordinates that are continous and that are defined w.r.t. the image center, meanwhile we aim to get pixel coords. referred to the topleft corner of the image!



In the end what we want is a model that goes FROM 3D WRF euclidian coords TO euclidian image 2D pixel coords (and viceversa)!

Image Digitization

in general applications, not necessary equal



u_0, v_0 are the pixel coordinates of the image center w.r.t. the top-left corner!

Δu= horizontal pixel size

Δv= vertical pixel size

OLD

$$u = \frac{f}{z}x \qquad v = \frac{f}{z}y$$

NEW

$$u = \frac{1}{\Delta u}\frac{f}{z}x = k_u \frac{f}{z}x + u_0$$

$$v = \frac{1}{\Delta v}\frac{f}{z}y = k_v \frac{f}{z}y + v_0$$

- FROM NOW, u and v are gonna be defined like that, including in their definitions the pixelization (AKA image digitalization) process!

- Digitization can be accounted for by including into the projection equations the scaling factors along the two axes due to the quantization associated with the horizontal and vertical pixel size. Morever, we need to model the translation of the image center (aka piercing point, the intersection between the optical axis and the image plane) wrt the origin of the image coordinate system (top-left corner of the image).

Intrinsic Parameter Matrix

Based on the equations in the previous slide, the PPM can be written as

In a lot of applications we do not know/estimate f, Kv, Ku BUT we are directly interested in αu and αv pure values! Indeed, the intrinsic params are considered to be four

$$\tilde{\mathbf{P}} = \begin{bmatrix} fk_u & 0 & u_0 & 0 \\ 0 & fk_v & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} fk_u & 0 & u_0 \\ 0 & fk_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} = \mathbf{A}[\mathbf{I}|\mathbf{0}]$$

incorporates pixelization process

canonic behaviour of any camera

specific intrinsic parameters of the camera

- Matrix **A**, which models the characteristics of the image sensing device, is called **Intrinsic Parameter Matrix** (or **Camera Matrix**). !
- Intrinsic parameters can be reduced in number by setting $\overbrace{\alpha_u = fk_u\ ,\ \alpha_v = fk_v}^{\text{PURE NUMBERS!}}$, such quantities representing, respectively, the focal length expressed in horizontal and vertical pixel sizes. The smallest number of intrinsic parameters is thus 4.
- A more general model would include a 5th parameter, known as *skew*, to account for possible non orthogonality between the axis of the image sensor. The *skew* would be A[1,2], but it is usually 0 (= ctg(π/2)) in practice.

Rigid motion between CRF and WRF (1)

- So far we have assumed 3D coordinates to be measured into the CRF, though this is hardly feasible in practice.
- More generally, 3D coordinates are measured into a World Reference Frame (WRF) external to the camera. The WRF will be related to the CRF by:
  - A rotation around the optical centre (e,g. expressed by a 3x3 rotation matrix $\mathbf{R}$)
  - A translation (expressed by a 3x1 translation vector $\mathbf{T}$)
- Therefore, the relation between the coordinates of a point in the two RFs is:

$$\mathbf{W} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}, \mathbf{M} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \Rightarrow \underset{\text{CRF}}{\mathbf{M}} = \mathbf{R}\underset{\text{WCF}}{\mathbf{W}} + \mathbf{T}$$

rototraslation in euclidian coordinates!

- Which can be rewritten in homogeneous coordinates as follows:

$$\tilde{W} = \begin{bmatrix} W \\ 1 \end{bmatrix} \qquad \tilde{M} = \begin{bmatrix} M \\ 1 \end{bmatrix}$$

$$\tilde{\mathbf{W}} = \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}, \tilde{\mathbf{M}} = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \Rightarrow \tilde{\mathbf{M}} = \overbrace{\begin{bmatrix} \mathbf{R} & \mathbf{T} \\ \mathbf{0} & 1 \end{bmatrix}}^{4\times 4}\tilde{\mathbf{W}} = \mathbf{G}\tilde{\mathbf{W}}$$

Again the usage of homogeneous coords has the advantage to make the equation as a simple and single matrix product!

rototraslation in homogeneous coordinates!

Rigid motion between CRF and WRF (2)

Computer Vision Laboratory

- So far So far we have seen how to map a 3D point expressed in the CRF

$$\cancel{\lambda}\tilde{\mathbf{m}} = \mathbf{A}\left[\mathbf{I} \mid \mathbf{0}\right]\tilde{\mathbf{M}}$$

- We need now to consider also the rigid motion between the WRF and the CRF :

$$\tilde{\mathbf{M}} = \mathbf{G}\tilde{\mathbf{W}} \longrightarrow \cancel{\lambda}\tilde{\mathbf{m}} = \mathbf{A}\left[\mathbf{I} \mid \mathbf{0}\right]\mathbf{G}\tilde{\mathbf{W}}$$

$$\cancel{\lambda}\tilde{\mathbf{m}} = \underbrace{\mathbf{A}\left[\mathbf{I} \mid \mathbf{0}\right]\begin{bmatrix} \mathbf{R} & \mathbf{T} \\ \mathbf{0} & 1 \end{bmatrix}}_{3\times4}\tilde{\mathbf{W}}$$

Accordingly, the general form of the PPM can be expressed as:

$$\tilde{\mathbf{P}} = \mathbf{A}\left[\mathbf{I} \mid \mathbf{0}\right]\mathbf{G} \quad \text{or also} \quad \tilde{\mathbf{P}} = \mathbf{A}\left[\mathbf{R} \mid \mathbf{T}\right]$$

Final model from WCR to pixel coords that includes BOTH 3D rototraslation and 2D pixelization (intrinsic & extrinsic parameters) of course all together with the canonic behaviour, for a total of 10 (specific) parameters!

Extrinsic Parameters

- **Matrix G**, which encodes the position and orientation of the camera with respect to the WRF, is called **Extrinsic Parameter Matrix**.

- As a rotation matrix (3x3=9 entries) has indeed only 3 independent parameters (DOF), which correspond to the rotation angles around the axis of the RF, the total number of extrinsic parameter is 6 (3 translation parameters, 3 rotation parameters).

FINAL RESULT...

- Hence, the general form of the PPM can be thought of as encoding the position of the camera wrt the world into $\mathbf{G}$, the perspective projection carried out by a pinhole camera into the canonical PPM $[\,\mathbf{I} \mid \mathbf{0}\,]$ and, finally, the actual characteristics of the camera into $\mathbf{A}$.

P as a Homography

Particular cases in which P behaves as an homography (simplification of its general behaviour!

- If the camera is imaging a **planar scene**, we can assume the z-axis of the WRF to be perpendicular to the plane such that all 3D points will have **their z-coordinate equal to 0**. Accordingly, the PPM boils down to a simpler transformation defined by a 3x3 matrix:

WRF has its origin in the said planar scene!

$$k\tilde{\mathbf{m}} = \tilde{\mathbf{P}}\,\tilde{\mathbf{W}} = \begin{bmatrix} p_{1,1} & p_{1,2} & p_{1,3} & p_{1,4} \\ p_{2,1} & p_{2,2} & p_{2,3} & p_{2,4} \\ p_{3,1} & p_{3,2} & p_{3,3} & p_{3,4} \end{bmatrix} \begin{bmatrix} x \\ y \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} p_{1,1} & p_{1,2} & p_{1,4} \\ p_{2,1} & p_{2,2} & p_{2,4} \\ p_{3,1} & p_{3,2} & p_{3,4} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \mathbf{H}\tilde{\mathbf{M}}$$

Be aware: only in the particular case of CRF and WRF beeing aligned AND with coincident axis z, then $\tilde{W}$ and $\tilde{M}$ (w.r.t. WRF and CRF) have same x,y coordinates – this is NOT true in general! Indeed, in general $\tilde{P}$ and H still include the rototranslation WRF → CRF

- Such a transformation, denoted here as $\mathbf{H}$, is known as **homography** and represents a general linear transformation between projective planes. Above, $\tilde{\boldsymbol{M}}$ represents vector $\begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$. $\mathbf{H}$ can be thought of as a simplification of $\mathbf{P}$ in case the imaged scene is planar.

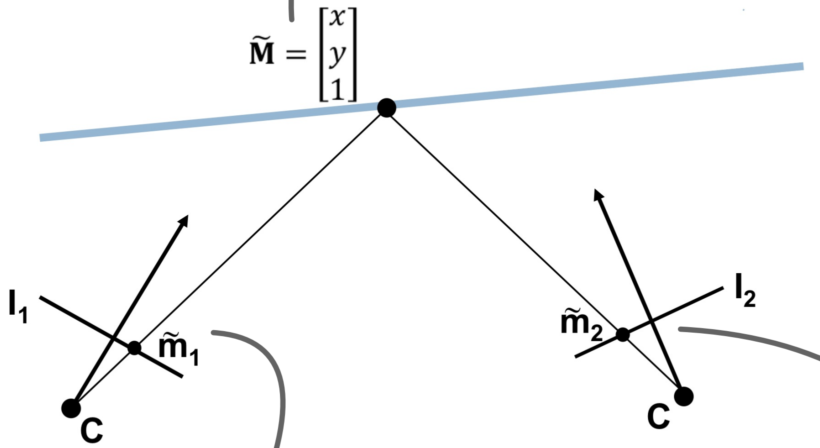
Indeed this homography is a linear transformation between the planar scene plane (in the 3D world) and the usual image plane, all of that in homogeneous coords! !

- Akin to $\mathbf{P}$, $\mathbf{H}$ is known up to an arbitrary scale factor and thus the independent elements in the 3x3 matrix are just 8. → AN HOMOGRAFY IS DEFINED UP TO SCALE → one of its element can be fixed to be one, so at most 8 of them can be independent! HOMOGRAPHY IS INDEED A PROJECTION TRANSFORMATION!

Homographies (1)

Any two images of a planar scene are related by a homography:

IMPORTANT: here we have an unique WRF with origin in the planar scene and z axis which is orthogonal to it; also, $\tilde{M}$ is (improperly) used to indicate a point in the WRF!



$$\tilde{\mathbf{m}}_1 = \mathbf{H}_1\tilde{\mathbf{M}} \qquad \tilde{\mathbf{m}}_2 = \mathbf{H}_2\tilde{\mathbf{M}}$$

$$\tilde{\mathbf{m}}_1 = \underbrace{\mathbf{H}_1\mathbf{H}_2^{-1}}_{\mathbf{H}_{21}}\tilde{\mathbf{m}}_2 \qquad \tilde{\mathbf{m}}_2 = \underbrace{\mathbf{H}_2\mathbf{H}_1^{-1}}_{\mathbf{H}_{12}}\tilde{\mathbf{m}}_1$$

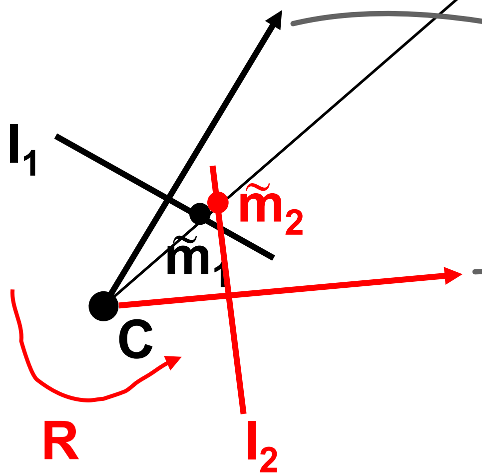
$\mathbf{H}_{21}$ ← **Homographys** → $\mathbf{H}_{12}$

- **These homographies link pixels in the first image with pixels in the second image.** They link corresponding image points, alias that are the projection (in the two images) of the same 3D point!

Homographies (2)

Any two images taken by a camera rotating about the optical center are related by a homography:

IMPORTANT: again in here $\tilde{M}$ indicates a point expressed in WRF. Also, we consider WRF coincident to CRF1 which implies for camera one [R|T]=[I|0] and for camera 2 [R|T]=[R|0] where R express rotation from CRF1 to CRF2!



$$\tilde{M} = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{M} \\ 1 \end{bmatrix}$$

$$\tilde{\mathbf{m}}_1 = \mathbf{A}[\mathbf{I}\ \mathbf{0}]\tilde{\mathbf{M}} = \mathbf{AM} \qquad \tilde{\mathbf{m}}_2 = \mathbf{A}[\mathbf{R}\ \mathbf{0}]\tilde{\mathbf{M}} = \mathbf{ARM}$$

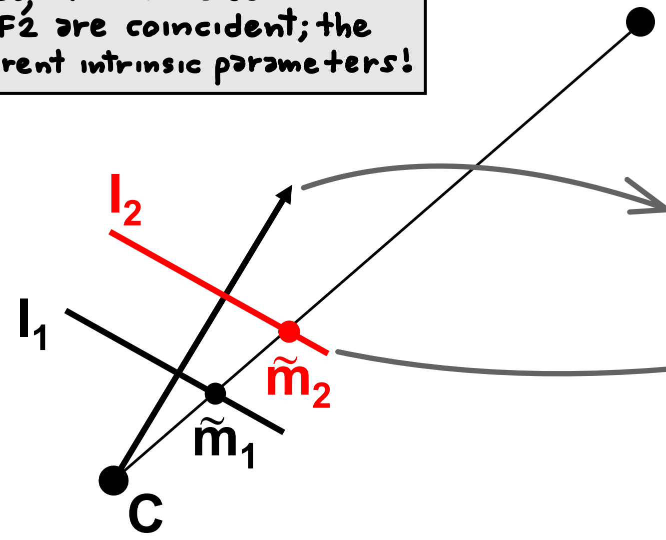
$$\tilde{\mathbf{m}}_1 = \underbrace{\mathbf{AR}^{-1}\mathbf{A}^{-1}}_{\mathbf{H}_{21}}\tilde{\mathbf{m}}_2 \qquad \tilde{\mathbf{m}}_2 = \underbrace{\mathbf{ARA}^{-1}}_{\mathbf{H}_{12}}\tilde{\mathbf{m}}_1$$

<u>Homographys</u>

Homographies (3)

Any two images taken by different cameras (i.e. different A) in a fixed pose (i.e. same CRF) are related by a homography:

IMPORTANT: again in here $\tilde{M}$ indicates a point expressed in WRF. Also, WRF coincident to CRF1; CRF1 and CRF2 are coincident; the two cameras have different intrinsic parameters!



$$\tilde{\mathbf{M}} = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{M} \\ 1 \end{bmatrix}$$

$$\widetilde{\mathbf{m}_1} = \mathbf{A}_1[\mathbf{I}\ \mathbf{0}]\tilde{\mathbf{M}} = \mathbf{A}_1\mathbf{M} \qquad \widetilde{\mathbf{m}_2} = \mathbf{A}_2[\mathbf{I}\ \mathbf{0}]\tilde{\mathbf{M}} = \mathbf{A}_2\,\mathbf{M}$$

$$\widetilde{\mathbf{m}_1} = \underbrace{\mathbf{A}_1\mathbf{A}_2^{-1}}_{\mathbf{H}_{21}}\widetilde{\mathbf{m}_2} \qquad \widetilde{\mathbf{m}_2} = \underbrace{\mathbf{A}_2\mathbf{A}_1^{-1}}_{\mathbf{H}_{12}}\widetilde{\mathbf{m}_1}$$

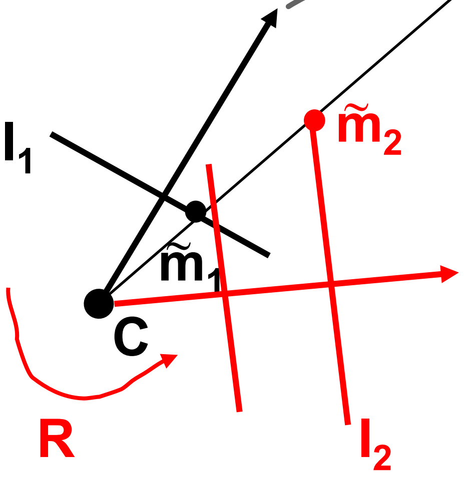
<u>Homographys</u>

Homographies (4)

Should we both rotate the camera about the optical center and change the intrinsic we still end up again with images related by a homography:

The previous two cases merged all together!

$$\tilde{\mathbf{M}} = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{M} \\ 1 \end{bmatrix}$$



$$\tilde{\mathbf{m}}_1 = \mathbf{A}_1[\mathbf{I}\ \mathbf{0}]\tilde{\mathbf{M}} = \mathbf{A}_1\mathbf{M} \qquad \tilde{\mathbf{m}}_2 = \mathbf{A}_2[\mathbf{R}\ \mathbf{0}]\tilde{\mathbf{M}} = \mathbf{A}_2\mathbf{R}\mathbf{M}$$

$$\tilde{\mathbf{m}}_1 = \underbrace{\mathbf{A}_1\mathbf{R}^{-1}\mathbf{A}_2^{-1}}_{\mathbf{H}_{21}}\tilde{\mathbf{m}}_2 \qquad \tilde{\mathbf{m}}_2 = \underbrace{\mathbf{A}_2\mathbf{R}\mathbf{A}_1^{-1}}_{\mathbf{H}_{12}}\tilde{\mathbf{m}}_1$$

<u>Homographys</u>

Lens Distortion

Still there is something very important to be taken into account in our model!

we have some NL that NEEDS to be included!

This has nothing to do with persective projection (that transforms lines in lines); the thin-lens-model, that consent to model lens with the pin-hole model and dealing on persective projection, is only ideal: in reality lens introduce distorsions that are NL deviations w.r.t. the model we've seen so far (linear under homogeneous coordinates)! This NL deviations that we've in reality are called lens distorsions!

We're gonna introduce a warping (AKA transformation) of images that compensates for the distortion issue!

- However, to explain observed images we often need to model also the effects due to the optical distortion induced by lenses, which indeed renders the pure pinhole not accurate enough a model in many applications. Lens distortion is modelled through additional parameters that do not alter the form of the PPM.

Modelling Lens Distortion

The Kind of modeling that we study includes two Kind of distorsion: radial and tangential distorsion!

we will see one of the many existing models (used by opencv) for lens distorsion!

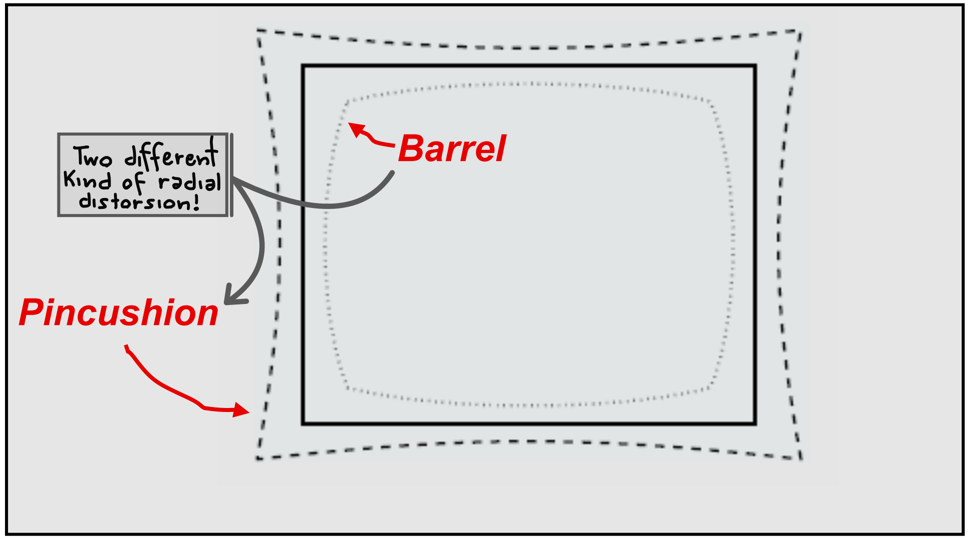
- The PPM is based on the pinhole camera model. However, real lenses introduce distortions wrt to the pure pinhole model, this being true especially for cheap and/or short focal length lenses. The most significant deviation from the ideal pinhole model is known as **radial distortion** (lens "curvature"). Second order effects are induced by **tangential distortion** ("misalignement" of optical components and/or defects).
- We adopt a model whereby lens distortion is modelled through a non-linear transformation which maps ***undistorted*** **continuous** image coordinates into ***distorted*** **continuous** image coordinates:

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \underbrace{L(r)}_{\text{RADIAL DISTORSION}} \begin{pmatrix} \tilde{x} \\ \tilde{y} \end{pmatrix} + \overbrace{\begin{pmatrix} d\tilde{x} \\ d\tilde{y} \end{pmatrix}}^{\text{TANGENTIAL DISTORSION}}$$

Be aware: this models lens distorsion on 2D continous image coordinates BEFORE the pixelization process (intrinsic params effect)

where r is the distance from the distortion centre $(\tilde{x}_c, \tilde{y}_c)$:

$$r = \sqrt{(\tilde{x} - \tilde{x}_c)^2 + (\tilde{y} - \tilde{y}_c)^2}$$



Distorsion accordingly to our modeling: RADIAL distorsion or TANGENTIAL distorsion

Radial distorsion: depends on the lens NOT beeing thin, alias its thikness not beeing infinitesimal! Among the two, it's the most present\affecting one. It's called "radial" because its entity is directly proportional to the distance from the center of distorsion AND grows with it. It may be BARREL (compressing) or PICUSHION (dilatating).

Tangential distorsion: depends on non idealities and disalignments in mounting lens in the camera!

The center of distorsion is a point of the 2D continous image coords. plane THAT ALSO is typically taken to be the center of the image itself!

Lens Distortion Parameters

- The radial distortion function $L(r)$ is defined for positive r only and such as $L(0) = 1$. This non-linear function is typically approximated by its Taylor series (up to a certain approximation order)

$$L(r) = 1 + k_1 r^2 + k_2 r^4 + k_3 r^6 + \ldots$$

$f(x) = f(0) + f'(0)x + \frac{1}{2} f''(0) x^2 + \ldots$

→ for $x \to 0$

where we want $f(0) = 1$ cause at the center of distorsion the radial distorsion has no effect (x is indeed the center of dist.). Also we do not have odd terms cause our $L(r) \equiv f(x)$ is an even positive defined function, indeed $L(r) = L(-r) > 0$!

$\Rightarrow K_1 = \frac{1}{2} f''(0)$; $K_2 = \frac{1}{4!} f^{IV}(0)$; ...

The more the terms, the more the precision!

TYPICALLY WE USE AT MOST 3 RADIAL DISTORSION PARAMETERS

K_1, K_2, K_3 to be estimated!

- The tangential distortion vector is instead approximated as follows

$$\begin{pmatrix} d\tilde{x} \\ d\tilde{y} \end{pmatrix} = \begin{pmatrix} 2 p_1 \tilde{x}\tilde{y} + p_2 \left(r^2 + 2\tilde{x}^2 \right) \\ p_1 \left(r^2 + 2\tilde{y}^2 \right) + 2 p_2 \tilde{x}\tilde{y} \end{pmatrix}$$

- The radial distortion coefficients $k_1, k_2, \ldots, k_n$, together with the distortion centre $(\tilde{x}_c, \tilde{y}_c)$ and the two tangential distortion coefficients p_1 and p_2 form the set of the lens distortion parameters, which extends the set of parameters required to build a useful and realistic camera model. Typically, for the sake of simplicity, the distortions centre is taken to coincide with the image centre (i.e. the piercing point).

Lens distortion in the image formation flow

- Lens distortion is modelled as a non-linear mapping taking place after canonical perspective projection onto the image plane. Afterwards, the intrinsic parameter matrix applies an affine transformation which maps continuous image coordinates into pixel coordinates.
- Accordingly, the image formation flow can be summarized as follows:

The final flow of image formation

1. Transformation of 3D points from the WRF to the CRF, according to extrinsic parameters:

$$\mathbf{M} = \mathbf{R}\mathbf{W} + \mathbf{T}$$

2. Canonical perspective projection (i.e. scaling by the third coordinate):

$$\tilde{x} = x / z \quad , \quad \tilde{y} = y / z$$

3. Non-linear mapping due to lens distortion:

$$\underset{\text{DISTORTED}}{\begin{pmatrix} x' \\ y' \end{pmatrix}} = L(r) \overset{\text{UNDISTORTED}}{\begin{pmatrix} \tilde{x} \\ \tilde{y} \end{pmatrix}} + \begin{pmatrix} d\tilde{x} \\ d\tilde{y} \end{pmatrix}$$

4. Mapping from continuous image coordinates to pixels coordinates according to the intrinsic parameters:

$$\tilde{\mathbf{m}} = \mathbf{A}\,(x'\ y'\ 1)^T$$