Lecture 1
Image Formation Model

IMAGE PROCESSING AND COMPUTER VISION – PART 2

SAMUELE SALTI

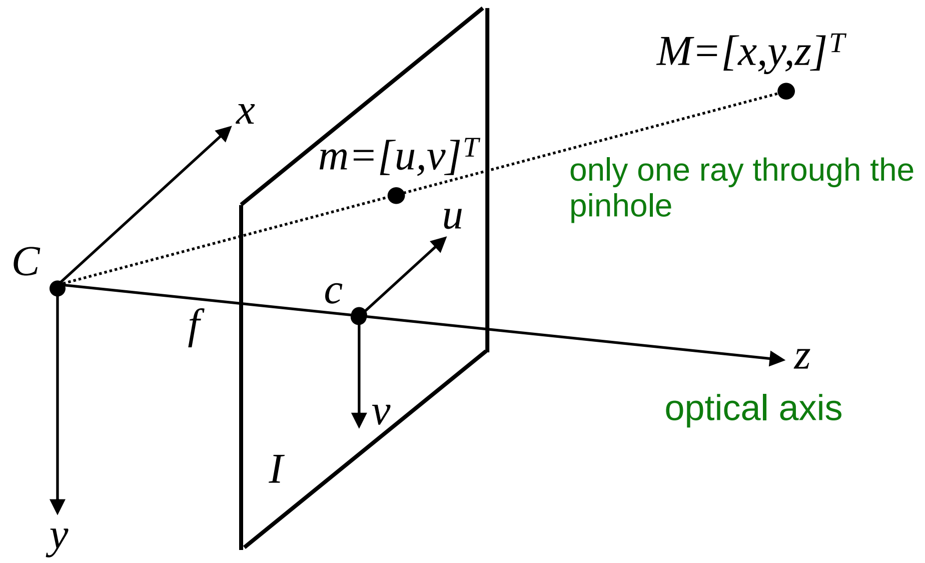
Image formation (recap)

we need a model for digital images

Images are 2D perspective projections of the 3D world.

Perspective projection model:

- given a point in 3D space, $M = [x, y, z]^T$, whose coordinates are given in the Camera Reference Frame (CRF),
- its perspective projection onto the image plane I, denoted as $m = [u, v]^T$ is given by the non-linear equations (1)



$$\begin{cases} u = \dfrac{f}{z}x \\ v = \dfrac{f}{z}y \end{cases} \quad (1)$$

non-linear operation

The Camera Reference Frame (CRF) is a 3D reference frame for which:

- the origin is the optical centre C
- x, y axis are parallel to u and v (the image horizontal and verical axis)
- z axis is the optical axis

they are all column vector but we need rows --> transpose

Why do we need an image formation model?

NeRF: Neural Radiance Fields

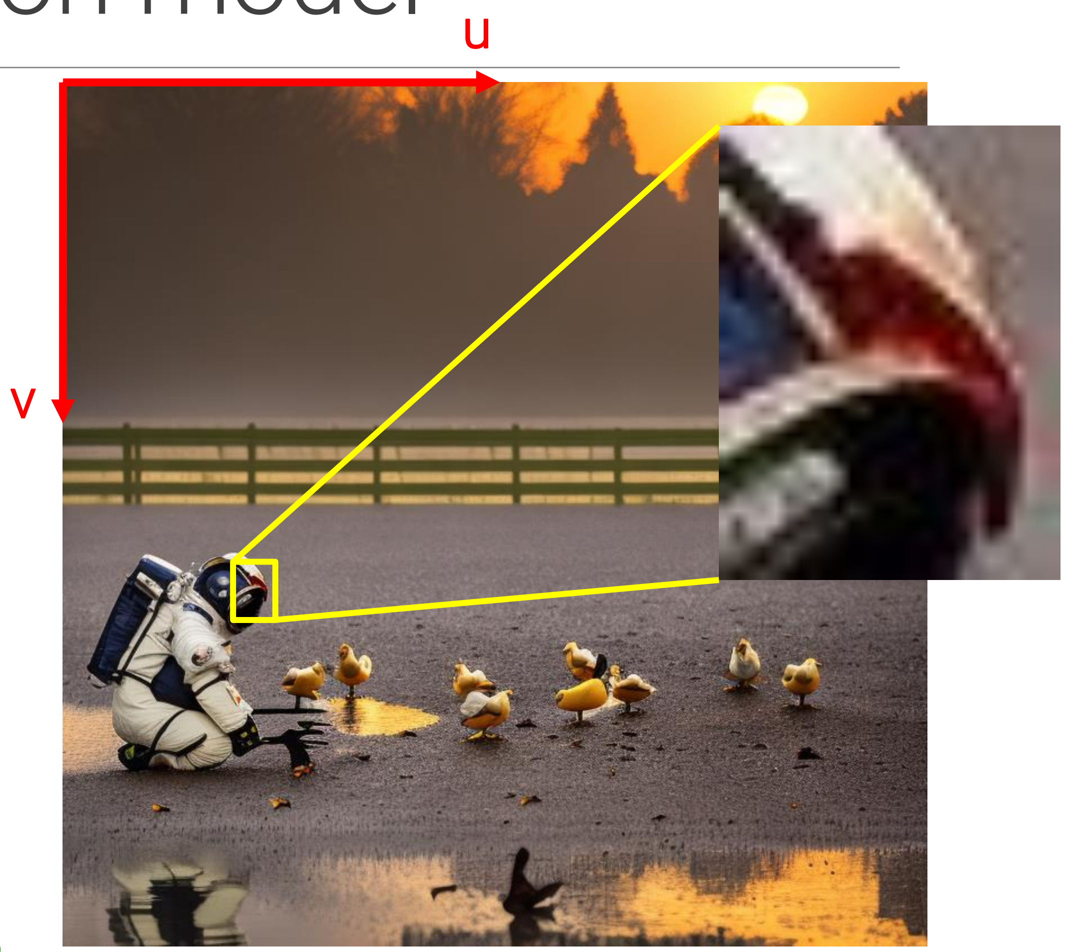
Ben Mildenhall et al., “NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis”, ECCV 2020.

A more realistic image formation model

To create a more realistic image formation model we need to consider three additional issues:

1. Image coordinates are usually expressed in an **image reference frame with the origin in the top left corner**
2. Images are **grid of pixels**, not a continuous plane (aka ***pixelization***)
3. We do not normally know or want to project back the coordinate of a 3D point in the Camera Reference Frame, but in some generic **World Reference Frame**.

the WRF is the 3d scene from which a point is captured and projected into the 2d image plane, through the CRF



https://stability.ai/blog/stable-diffusion-v2-release

Projecting Back: This refers to the theoretical process of trying to recover the 3D location of a point in the world (WRF) given only its corresponding image point

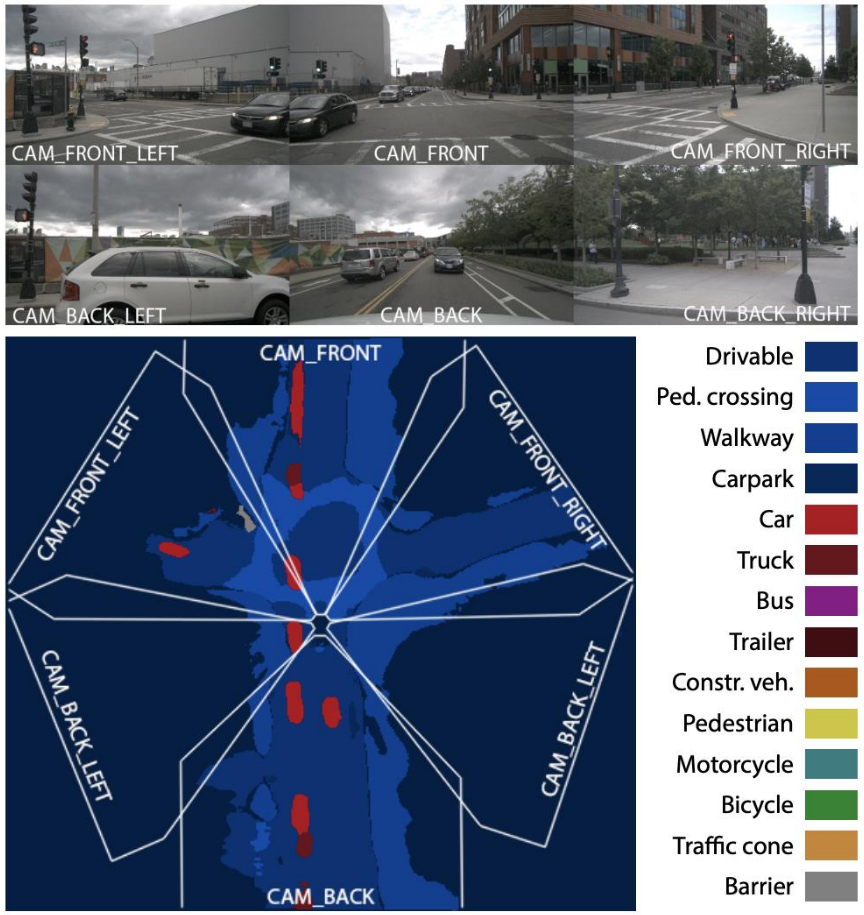
$\hookrightarrow$ but we know it is not completely possible because in the projection process we loose the depth information (z-coord) of the point

A more realistic camera model

To create a more realistic camera model we need to consider three additional issues:

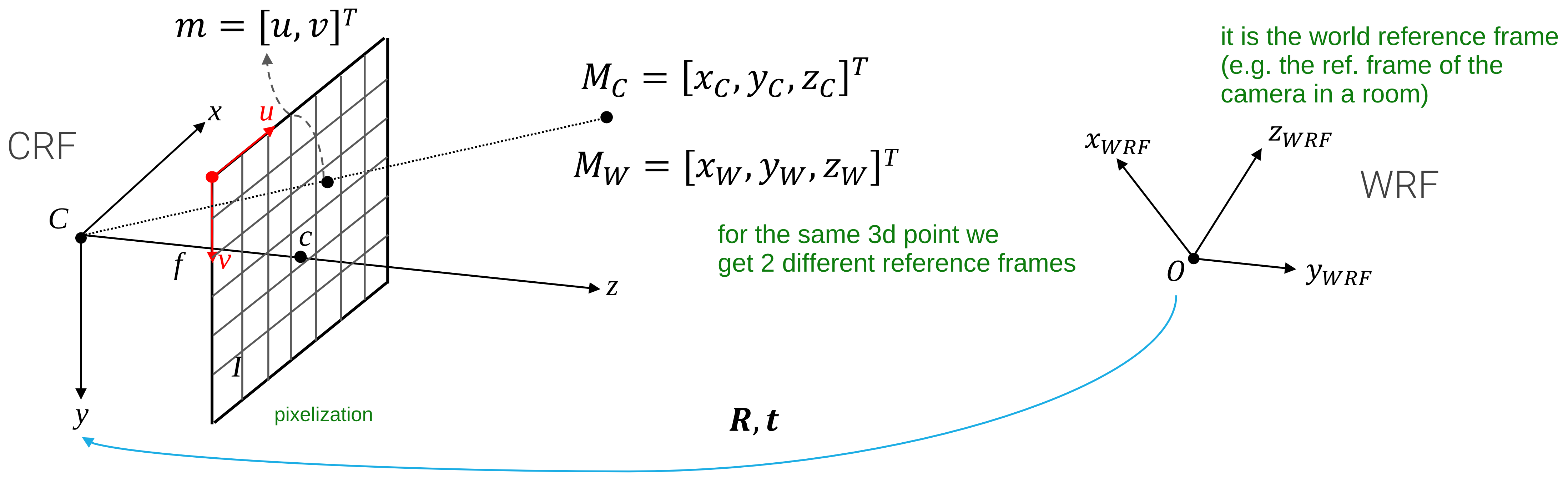
1. Image coordinates are usually expressed in an image reference frame with the origin in the top left corner
2. Images are grid of pixels, not a continuous plane (aka *pixelization*)
3. We do not normally know or want to project back the coordinate of a 3D point in the Camera Reference Frame, but in some generic World Reference Frame. thus 3d

(e.g. robot navigation, Augmented reality, etc.)



Thomas Roddick, Roberto Cipolla, “Predicting Semantic Map Representations from Images using Pyramid Occupancy Networks”, CVPR 2020.

Complete Forward Imaging model: 3D to 2D


$m = [u, v]^T$
$M_C = [x_C, y_C, z_C]^T$
$M_W = [x_W, y_W, z_W]^T$
CRF
C
x
u
v
c
f
z
y
I
pixelization
it is the world reference frame (e.g. the ref. frame of the camera in a room)
x_{WRF}
z_{WRF}
WRF
O
y_{WRF}
for the same 3d point we get 2 different reference frames
$\boldsymbol{R}, \boldsymbol{t}$

2D point [pixels]

$$m = \begin{bmatrix} u \\ v \end{bmatrix}$$

transformation

3D point in CRF [Metric, e.g. mm]

$$M_C = \begin{bmatrix} x_C \\ y_C \\ z_C \end{bmatrix}$$

transformation

3D point in WRF [Metric, e.g. mm]

meters, mm, cm, etc.

$$M_W = \begin{bmatrix} x_W \\ y_W \\ z_W \end{bmatrix}$$

Intrinsic parameters
they depend on the model

Extrinsic parameters
they depend on the position of the camera in the world

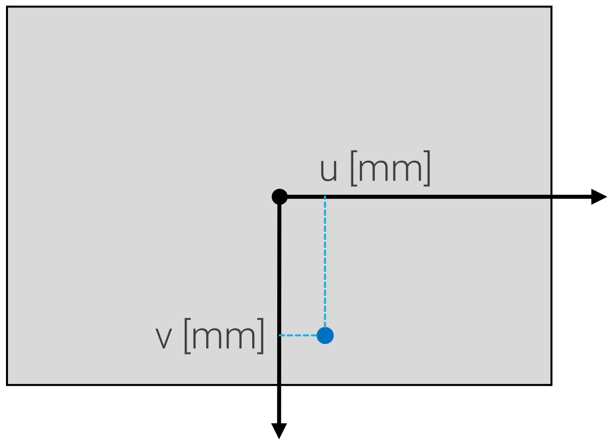
Image Pixelization

which pixel is activated? namely recevies the light?

Digitization can be accounted for by including into the projection equations the pixel size $\boldsymbol{\Delta u}$ and $\boldsymbol{\Delta v}$ along the two axes (typically they have the same value, i.e. pixels are squared, but in the general model we let them be different)

- We have a discrete grid of pixels (not a continuous plane)

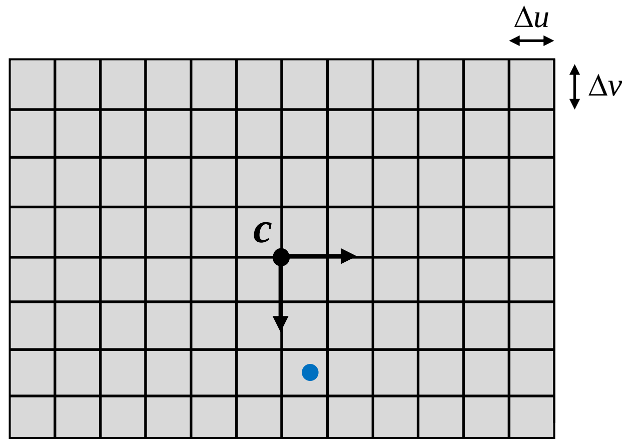
Ideal case: Image plane



$$u = \frac{f}{z_C} x_C$$

$$v = \frac{f}{z_C} y_C$$

Real case: Image sensor



it gives us a real number

$$u = \frac{1}{\Delta u} \frac{f}{z_C} x_C$$

$$v = \frac{1}{\Delta v} \frac{f}{z_C} y_C$$

we introduce the delta_u/v factor to deal with the size of pixels

Δu = horizontal pixel size in mm (quantization step)
Δv= vertical pixel size in mm (quantization step)

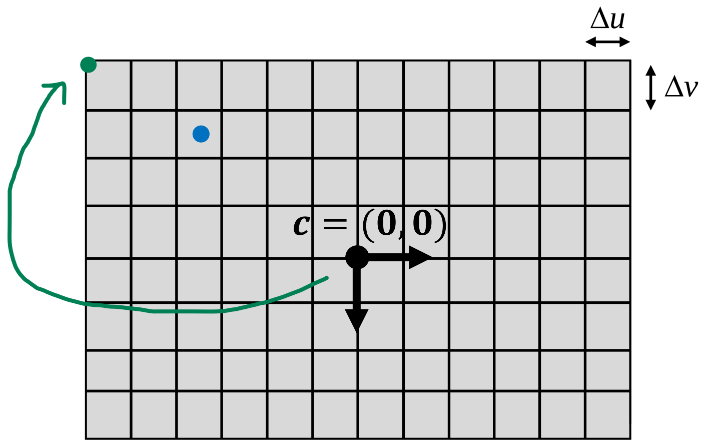
Origin of the image reference frame

We also need to model the translation of the piercing point (intersection between the optical axis and the image plane) wrt the origin of the image coordinate system (top-left corner of the image)

- We do not use negative pixel indices when accessing images

translation of the reference system

Ideal origin as piercing point

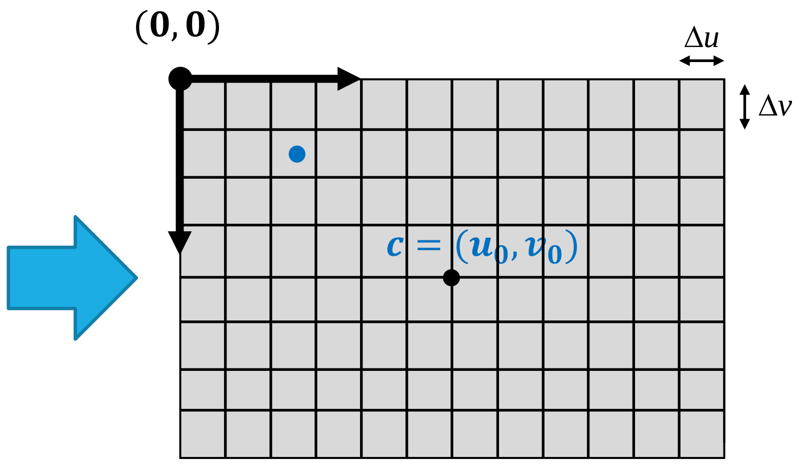


$$u = \frac{1}{\Delta u}\frac{f}{z_C}x_C$$

$$v = \frac{1}{\Delta v}\frac{f}{z_C}y_C$$

in this way all the pixels will be positive

Real origin



$$u = \frac{1}{\Delta u}\frac{f}{z_C}x_C + u_0$$

$$v = \frac{1}{\Delta v}\frac{f}{z_C}y_C + v_0$$

u_0 = horizontal coordinate of the piercing point [px]
v_0 = vertical coordinate of the piercing point [px]

Intrinsic parameters

model parameters (reference system)

The final relationship between 3D coordinates in the Camera Reference Frame and its corresponding pixel in an image is

$$\begin{cases} u = \frac{1}{\Delta u}\frac{f}{z_C}x_C + u_0 \\ v = \boxed{\frac{1}{\Delta v}\frac{f}{z_C}}y_C + v_0 \end{cases}$$

scaling factor

It is common to consider all multiplicative parameters in each equation as one parameter, i.e. to write

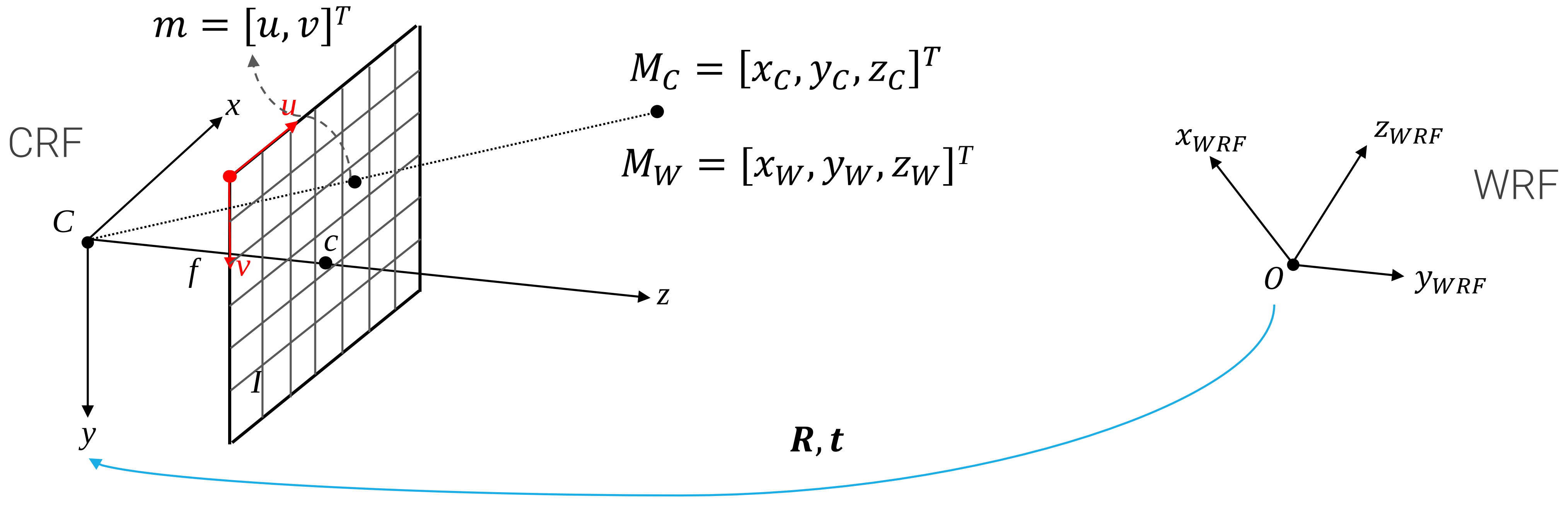
$$\begin{cases} u = f_u\frac{x_C}{z_C} + u_0 \\ v = f_v\frac{y_C}{z_C} + v_0 \end{cases}$$

f_u is the horizontal focal lenght in pixels

where $f_u = \frac{f}{\Delta u}$ is the focal length measured in horizontal pixels and $f_v = \frac{f}{\Delta v}$ is the focal length measured in vertical pixels.

The total number of intrinsic parameters is then 4: f_u, f_v and the coordinates of the piercing point c (u_0, v_0). They represent the camera geometry, which is independent of its position in the world.

Complete Forward Imaging model: 3D to 2D



$m = [u, v]^T$
CRF
x
u
C
f
v
c
I
y
z
$M_C = [x_C, y_C, z_C]^T$
$M_W = [x_W, y_W, z_W]^T$
x_{WRF}
z_{WRF}
WRF
O
y_{WRF}
$\boldsymbol{R}, \boldsymbol{t}$

2D point [pixels]		3D point in CRF [Metric, e.g. mm]		3D point in WRF [Metric, e.g. mm]
$m = \begin{bmatrix} u \\ v \end{bmatrix}$	$\Leftarrow$	$M_C = \begin{bmatrix} x_C \\ y_C \\ z_C \end{bmatrix}$	$\Leftarrow$	$M_W = \begin{bmatrix} x_W \\ y_W \\ z_W \end{bmatrix}$
	4 Intrinsic parameters		Extrinsic parameters	

Rigid motion between CRF and WRF

3D coordinates are measured into a World Reference Frame (WRF) external to the camera, related to the CRF by a *Roto-Translation*:

- A rotation around the optical centre (e.g. expressed by a 3x3 rotation matrix $\boldsymbol{R}$) corresponding axes should be aligned
- A translation (expressed by a 3x1 translation vector $\boldsymbol{t}$) to the center C (0,0,0)

Therefore, the relation between the coordinates of a point in the two RFs is:

$$\boldsymbol{M_C} = \begin{bmatrix} x_C \\ y_C \\ z_C \end{bmatrix} = \mathbf{R}\,\boldsymbol{M_W} + \boldsymbol{t} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \begin{bmatrix} x_W \\ y_W \\ z_W \end{bmatrix} + \begin{bmatrix} t_1 \\ t_2 \\ t_3 \end{bmatrix}$$

To understand how an object / a point will appear in the image, we need to know its location relative to the camera (CRF). This is achieved by roto-translation.

$\hookrightarrow$ only after this step, we can perform the 2d projection

Rotation matrix

The 3x3 matrix structure allows us to represent how the original coordinates (X, Y, Z) of a point are transformed along each axis after the rotation.

Any valid rotation matrix is an orthonormal matrix, i.e. its rows and columns are orthonormal.

Any two vectors $\boldsymbol{a}$ and $\boldsymbol{b}$ are orthonormal if and only if

$$\langle \boldsymbol{a}, \boldsymbol{b} \rangle = \boldsymbol{a}^T \boldsymbol{b} = 0 \quad \text{and} \quad \|\boldsymbol{a}\|_2 = \|\boldsymbol{b}\|_2 = 1$$

(orthogonality) (unit length)

For an orthonormal matrix $\boldsymbol{R}$ it is then true that

$$\boldsymbol{R}\,\boldsymbol{R}^T = \boldsymbol{R}^T\boldsymbol{R} = \boldsymbol{I}$$

i.e. its inverse is its transpose matrix.

What are the coordinates $\boldsymbol{C}_W$ of the optical centre $\boldsymbol{C}$ in the world reference frame?

$$\boldsymbol{0} = \boldsymbol{R}\boldsymbol{C}_W + \boldsymbol{t} \;\Rightarrow\; \boldsymbol{R}\boldsymbol{C}_W = -\boldsymbol{t} \;\Rightarrow\; \boldsymbol{C}_W = -\boldsymbol{R}^T\boldsymbol{t}$$

Extrinsic parameters

The rotation matrix has 9 entries but it has only 3 independent parameters (Degrees of Freedom), which correspond to the rotation angles around the axes of the Reference Frame

t has 3 independent entries

The total number of extrinsic parameters is then $\boxed{6}$ (3 translation parameters, 3 rotation parameters). They encode the position of the camera with respect to the WRF.

Given the nature of the rotation matrix, we have:

3 parameters for rotation around the x-axis.
3 parameters for rotation around the y-axis.
3 parameters for rotation around the z-axis.

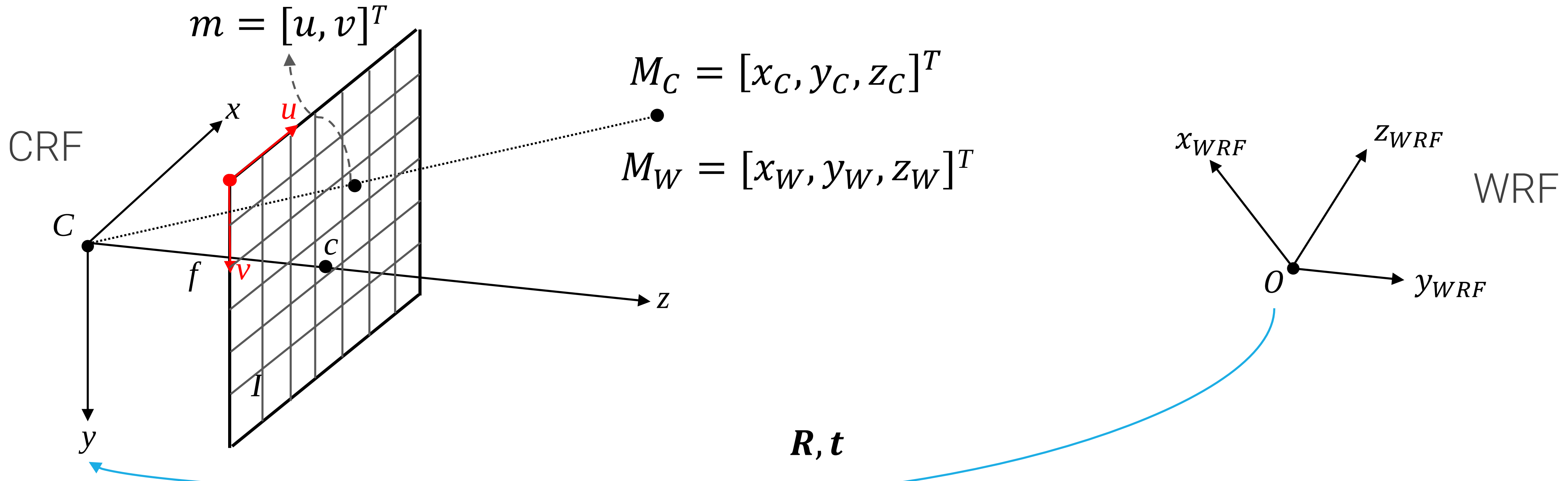
then 9 entries

However, since the length of each axis vector is constrained to be 1 and they must be orthogonal, some of the parameters are not independent. For instance:

If you know the rotation around the x-axis, the other two axes' rotations are affected due to orthogonality (1 param. needed)
If you know two of the rotations (e.g., around x and y axes), the third rotation (around the z-axis) can be determined from them due to the orthogonality constraint (2 param. needed)

then 3 indipendent params

Complete Forward Imaging model: 3D to 2D



$m = [u, v]^T$
$M_C = [x_C, y_C, z_C]^T$
$M_W = [x_W, y_W, z_W]^T$
CRF
C
x
u
v
c
f
z
y
I
x_{WRF}
z_{WRF}
WRF
O
y_{WRF}
$\boldsymbol{R}, \boldsymbol{t}$

2D point [pixels]		3D point in CRF [Metric, e.g. mm]		3D point in WRF [Metric, e.g. mm]
$m = \begin{bmatrix} u \\ v \end{bmatrix}$	$\Leftarrow$	$M_C = \begin{bmatrix} x_C \\ y_C \\ z_C \end{bmatrix}$	$\Leftarrow$	$M_W = \begin{bmatrix} x_W \\ y_W \\ z_W \end{bmatrix}$
	4 Intrinsic parameters		6 Extrinsic parameters	

Can we just combine the equations we have?

$$\begin{cases} u = f_u \dfrac{x_C}{z_C} + u_0 \\ v = f_v \dfrac{y_C}{z_C} + v_0 \end{cases}$$

Intrinsic camera model

$$\begin{bmatrix} x_C \\ y_C \\ z_C \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \begin{bmatrix} x_W \\ y_W \\ z_W \end{bmatrix} + \begin{bmatrix} t_1 \\ t_2 \\ t_3 \end{bmatrix}$$

Extrinsic roto-translation

the non linearity arises from the division

$$\begin{cases} u = f_u \dfrac{r_{11}x_W + r_{12}y_W + r_{13}z_W + t_1}{r_{31}x_W + r_{32}y_W + r_{33}z_W + t_3} + u_0 \\ v = f_v \dfrac{r_{21}x_W + r_{22}y_W + r_{23}z_W + t_1}{r_{31}x_W + r_{32}y_W + r_{33}z_W + t_3} + v_0 \end{cases}$$

non linear: no straight-line relationship between input and output

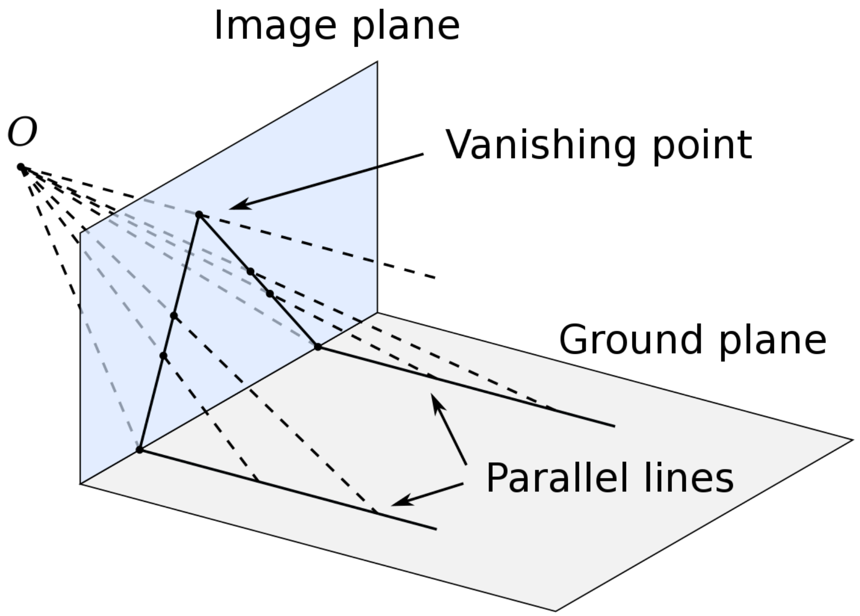
Yes, we can. But it gives us a non-linear model.

Can we make it linear?

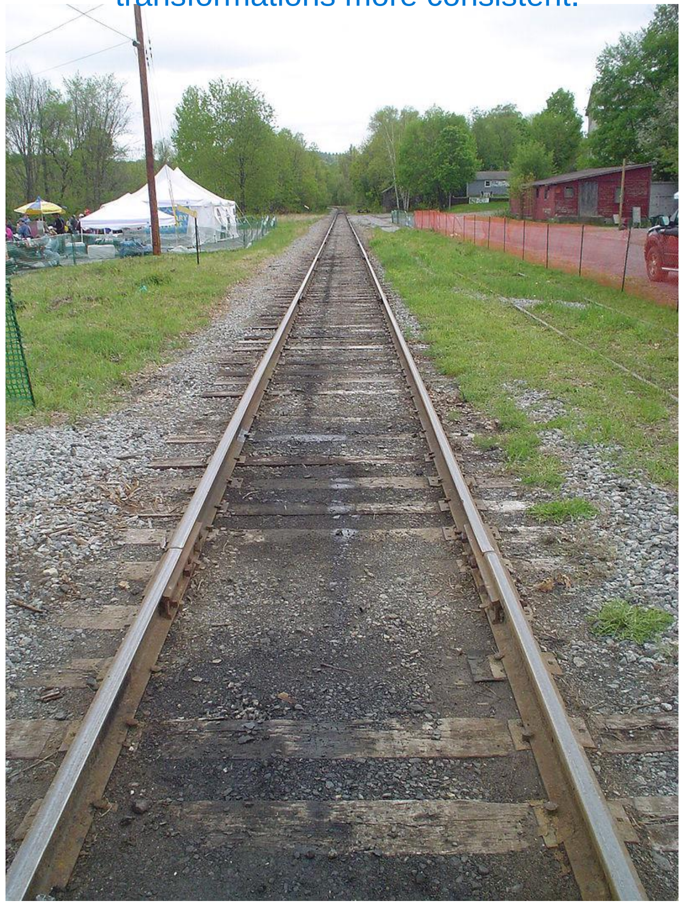
Points at infinity and vanishing point

In projective geometry, points at infinity are hypothetical points that lie "at infinity" along lines in a projective space. They are introduced to extend the concept of points to include those that are infinitely far away in a particular direction. Points at infinity help complete certain geometric figures and make projective transformations more consistent.

in perspective projection, lines that are parallel in the 3D world (such as railroad tracks) converge to points at infinity in the image plane. These points at infinity are essential for accurately representing parallel lines in a perspective view.



Vanishing points are specific points in a perspective drawing or image where parallel lines in the scene appear to converge or meet. In linear perspective, which is a geometric technique used in drawing and painting to create the illusion of depth on a flat surface, vanishing points play a crucial role.



By Mgunyho, original drawing by Niharikamaheshwari - Vectorized version of Vanishing Point.jpg, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=81515012

By User:MikKBDFJKGeMalak - Own work, Public Domain, https://commons.wikimedia.org/w/index.php?curid=805589

Projective Space

The standard 2D Euclidean plane can be represented as $\mathbb{R}^2$, i.e. by 2D vectors in a given reference frame. In this space

- parallel lines do not intersect and
- points at infinity cannot be represented.

Let's now append one more coordinate to our Euclidean vectors, so that:

m = [u ,v] and m' = [u,v,1] represents the same point

$$m = [u, v] \quad \text{becomes} \quad \tilde{m} = [u, v, 1]$$

This additional coordinate introduces the concept of homogeneous coordinates.

and assume that both vectors are equivalent representations of the same 2D point

Moreover, we do not constrain the 3rd coordinate to be 1 but instead let

$$\tilde{m} \equiv [u, v, 1] \boxed{\equiv} [2u, 2v, 2] \equiv [ku, kv, k] \; \forall \; k \neq 0$$

"equivalent", not "equal"

However, we can scale the coordinates by any non-zero factor k and still represent the same point.

Projective spaces

the projective space P^2 extends the euclidean space R^2 to include points at the infinity, providing then a more unified framework for geometry and transformations

$$\tilde{m} \equiv [u, v, 1] \equiv [2u, 2v, 2] \equiv [ku, kv, k] \; \forall \; k \neq 0$$

In this representation a point in the plane is represented by an equivalence class of triplets, wherein equivalent triplets differ by a multiplicative factor.

These are referred to as the homogeneous coordinates (a.k.a. projective coordinates) of the 2D point having Euclidean coordinates$[u, v]^T$. The space associated with the homogeneous representation is called Projective Space, denoted as $\mathbb{P}^{\mathbf{2}}$.

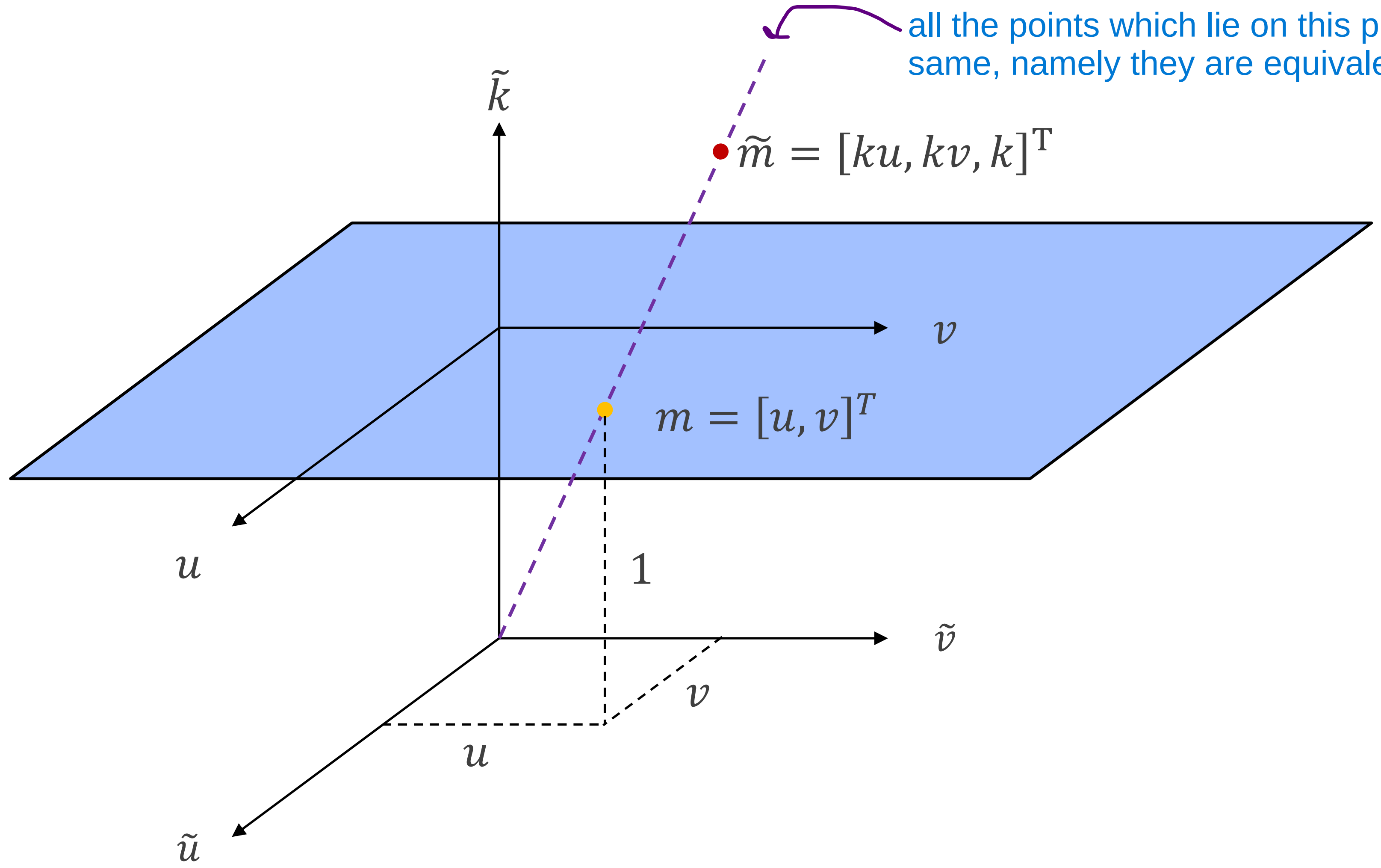
Extensions to Euclidean spaces of any other dimension is straightforward ($\mathbb{R}^{\boldsymbol{n}} \rightarrow \mathbb{P}^{\boldsymbol{n}}$), e.g. for the 3D Euclidean space $\mathbb{R}^{\mathbf{3}}$ we can convert a point $M_C \in \mathbb{R}^{\mathbf{3}}$ into projective coordinates $\widetilde{M}_C \in \mathbb{P}^{\mathbf{3}}$ by adding a 4th coordinate

$$\widetilde{M}_C \equiv [x, y, z, 1] \equiv [2x, 2y, 2z, 2] \equiv [kx, ky, kz, k] \; \forall \; k \neq 0$$

What have we done?

we have added another dimension, defined by k

all the points which lie on this purple line are the same, namely they are equivalent



Point at infinity of a 2D line

Parametric equation of a 2D line: $m = m_0 + \lambda d = \begin{bmatrix} u_0 \\ v_0 \end{bmatrix} + \lambda \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} u_0 + \lambda a \\ v_0 + \lambda b \end{bmatrix}$

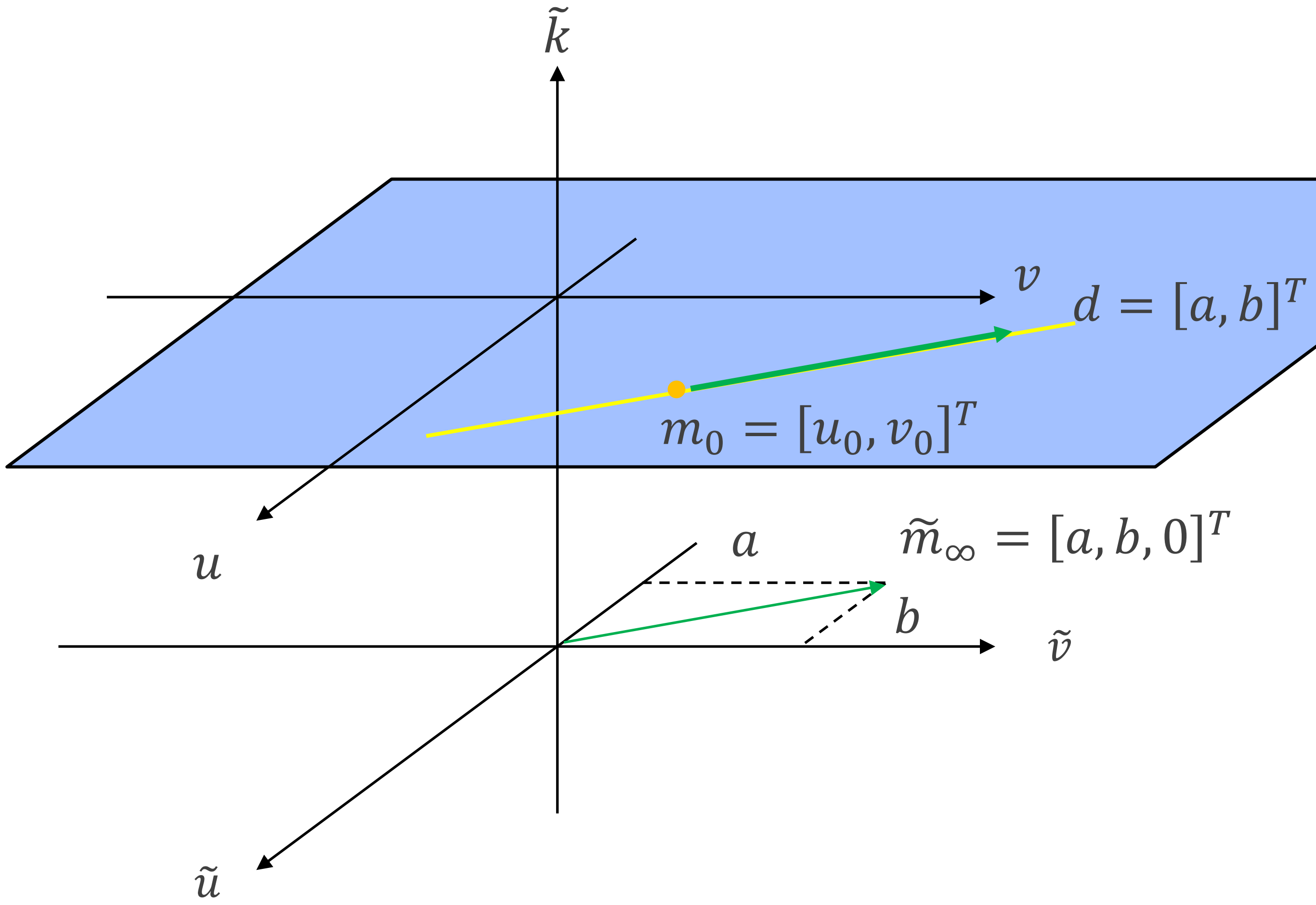
Generic point along the line in projective coordinates: $\tilde{m} \equiv \begin{bmatrix} m \\ 1 \end{bmatrix} \equiv \begin{bmatrix} u_0 + \lambda a \\ v_0 + \lambda b \\ 1 \end{bmatrix} \equiv \begin{bmatrix} \frac{u_0}{\lambda} + a \\ \frac{v_0}{\lambda} + b \\ \frac{1}{\lambda} \end{bmatrix}$

By taking the limit with λ → ∞ we obtain the projective coordinates of the point at infinity of the given line: $\tilde{m}_\infty = \lim_{\lambda \to \infty} \tilde{m} \equiv \begin{bmatrix} a \\ b \\ 0 \end{bmatrix}$

The point at infinity of a line has projective coordinates equal to the direction of the line, but with $\boldsymbol{k = 0}$

There exist infinitely many points at infinity in $\mathbb{P}^{\mathbf{2}}$, as many as the directions of the 2D lines

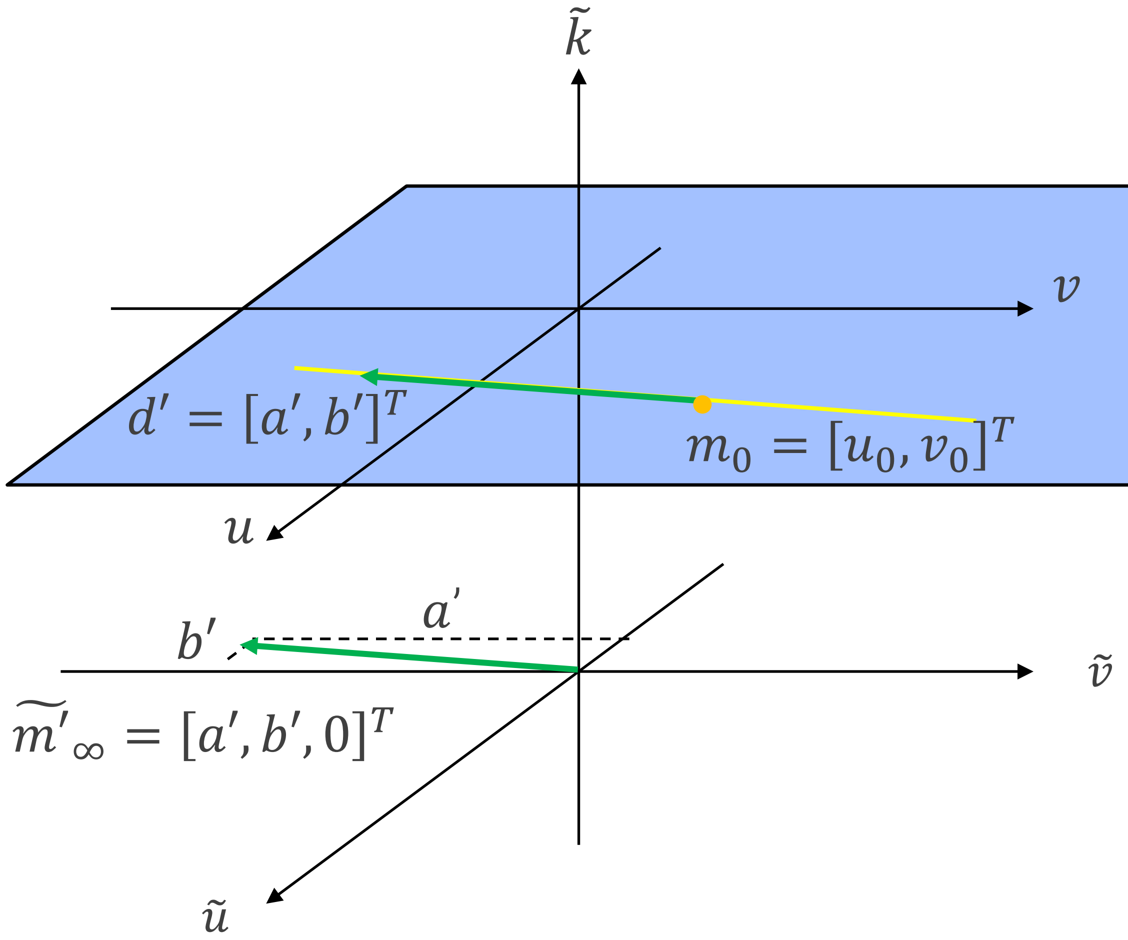
Points at infinity



All the points where $k = 0$, but (0,0,0) are valid $\mathbb{P}^2$ points!

They form the line at infinity.

Line at infinity



All the points where $k = 0$, **but (0,0,0)** are valid $\mathbb{P}^{\mathbf{2}}$ points!

They form the line at infinity.

Point at infinity of a 3D line

Parametric equation of a 3D line: $M = M_0 + \lambda D = \begin{bmatrix} x_0 \\ y_0 \\ z_0 \end{bmatrix} + \lambda \begin{bmatrix} a \\ b \\ c \end{bmatrix} = \begin{bmatrix} x_0 + \lambda a \\ y_0 + \lambda b \\ z_0 + \lambda c \end{bmatrix}$

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 λ → ∞ we obtain the projective coordinates of the

point at infinity of the given line: $\widetilde{M}_\infty = \lim_{\lambda \to \infty} \widetilde{M} = \begin{bmatrix} a \\ b \\ c \\ 0 \end{bmatrix}$

Perspective spaces: recap

- Any Euclidean Space $\mathbb{R}^n$ can be extended to a corresponding **Projective Space** $\mathbb{P}^n$ by representing points in homogeneous coordinates. adding another coordinate, k

- The projective representation features one additional coordinate, referred to here as k, wrt the Euclidean representation:
  - $k \neq 0$ denotes points existing in $\mathbb{R}^n$, their coordinates are given by $\frac{x_i}{k}$, $i = 1, \dots, n$
  - $k = 0$ denotes points at infinity (a.k.a. ideal points) in $\mathbb{R}^n$, which do not admit a representation via Euclidean coordinates.
  - Point (0, 0, 0, 0) is not part of $\mathbb{P}^3$. It is NOT the origin of the Euclidean Space (0, 0, 0). The origin is represented in homogeneous coordinates as [0, 0, 0, *k*] with *k*≠0

the origin of the Projective Space

- The Projective Space allows 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.

Perspective Projection in homogeneous coordinates

Let’s go back to the non-linear transformations between 3D coordinates in the CRF and image coordinates:

$$u = f_u \frac{x_C}{z_C} + u_0 \, , v = f_v \frac{y_C}{z_C} + v_0$$

Given a 3D point $M_C = [x_C, y_C, z_C]^T$ in the CRF and its projection onto the image plane $m = [u, v]^T$, what happens if we express this operation between projective spaces instead of Euclidean ones?

If they are expressed in homogenous coordinates (denoted by symbol ~) then the equations can be written as matrix multiplication, i.e. in linear form:

$$\tilde{m} \equiv \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \equiv \begin{bmatrix} f_u \frac{x_C}{z_C} + u_0 \\ f_v \frac{y_C}{z_C} + v_0 \\ 1 \end{bmatrix} \boxed{\equiv} \begin{bmatrix} f_u x_C + z_C u_0 \\ f_v y_C + z_C v_0 \\ z_C \end{bmatrix} \equiv \begin{bmatrix} f_u & 0 & u_0 & 0 \\ 0 & f_v & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_C \\ y_C \\ z_C \\ 1 \end{bmatrix} \equiv \boldsymbol{P_{int}} \tilde{M}_C$$

multiply all entries by z_C

Using the matrix P, we can now express the projection of the 3D point M'c to the 2D point m' in homogeneous coordinates as a linear transformation

Perspective Projection in homogeneous coordinates

In homogeneous coordinates the perspective projection becomes a linear transformation:

$$\tilde{m} \equiv \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \equiv \begin{bmatrix} ku \\ kv \\ k \end{bmatrix} \equiv \begin{bmatrix} f_u & 0 & u_0 & 0 \\ 0 & f_v & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_C \\ y_C \\ z_C \\ 1 \end{bmatrix} \equiv \boldsymbol{P_{int}} \widetilde{M}_C$$

Given $\tilde{m} \equiv \begin{bmatrix} \tilde{u} \\ \tilde{v} \\ \tilde{k} \end{bmatrix} \equiv \begin{bmatrix} u \\ v \\ 1 \end{bmatrix}$ we can recover the 2D pixel coordinates on the Euclidean sensor by dividing by the third coordinate, i.e. $u = \frac{\tilde{u}}{\tilde{k}}$, $v = \frac{\tilde{v}}{\tilde{k}}$

To emphasize that everything is equal up to an arbitrary scale factor k, the equation can also be written as $k\tilde{m} = \boldsymbol{P_{int}} \widetilde{M_C}$ or as $\tilde{m} \approx \boldsymbol{P_{int}} \widetilde{M_C}$

Points at infinity ≠ vanishing point

all the lines parallel to the image plane will not have a vanishing point because the will not converge along theh optical axis

The point at infinity is a point in space, the vanishing point is a point onto the image plane.

However, there is a very clear relationship between the two: the vanishing point for a set of 3D parallel lines is the projection of their point at infinity

Point at infinity cannot be represented in the **3D Euclidean Space:** to map such points into the Euclidean Space we would divide by the fourth coordinate (x/0, y/0, z/0), which is not a valid representation in the Euclidean Space

With homogenous coordinates it is instead possible to represent and process both *ordinary points* as well as *points at infinity*

For instance, we can easily project the point at infinity of a set of 3D parallel lines expressed in the CRF,

$$\tilde{m}_\infty \equiv \boldsymbol{P}_{int} \underbrace{\begin{bmatrix} a \\ b \\ c \\ 0 \end{bmatrix}}_{\text{M'c\_inf}} \equiv \begin{bmatrix} f_u & 0 & u_0 & 0 \\ 0 & f_v & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ 0 \end{bmatrix} \equiv \begin{bmatrix} f_u a + c u_0 \\ f_v b + c v_0 \\ c \end{bmatrix}$$

which, in Euclidean coords is $m_\infty = \begin{bmatrix} f_u \frac{a}{c} + u_0 \\ f_v \frac{b}{c} + v_0 \end{bmatrix}$

we recover them dividing by the third coord, c

Intrinsic Parameter Matrix

$$\tilde{m} \equiv \left[\begin{array}{ccc|c} \boxed{\begin{matrix} f_u & 0 & u_0 \\ 0 & f_v & v_0 \\ 0 & 0 & 1 \end{matrix}} & \begin{matrix} 0 \\ 0 \\ 0 \end{matrix} \end{array}\right] \begin{bmatrix} x_C \\ y_C \\ z_C \\ 1 \end{bmatrix} \equiv \boldsymbol{P}_{int} \widetilde{M_C} \equiv [\boldsymbol{A}|\boldsymbol{0}] \widetilde{M_C}$$

The 3x3 matrix leftmost part of $\boldsymbol{P}_{int}$ is called the intrinsic parameter matrix. It is usually referred to as $\boldsymbol{A}$ or $\boldsymbol{K}$ and it models the characteristics of the image sensing device. It contains the four parameters we already introduced.

instrinsic

It is always an upper right triangular matrix.

Intrinsic Parameter Matrix

A more general model would include a 5*th* parameter, known as skew, to account for possible non orthogonality between the axes of the image sensor.

The skew would be A[1,2], but it is usually 0 in practice

This encodes any possible skew between the sensor axes due to

- the sensor not being mounted perpendicular to the optical axis, or
- issues in the manufacturing process (in the sensor the pixel rows and columns are not perfectly perpendicular).

Extrinsic parameter matrix

If we consider the general case of 3D points expressed in a world projective space, the relationship between CRF and WRF can also be expressed with one 4x4 matrix $\boldsymbol{G}$, the extrinsic parameter matrix

$$M_C = \begin{bmatrix} x_C \\ y_C \\ z_C \end{bmatrix} = \mathbf{R}\, M_W + \boldsymbol{t} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \begin{bmatrix} x_W \\ y_W \\ z_W \end{bmatrix} + \begin{bmatrix} t_1 \\ t_2 \\ t_3 \end{bmatrix}$$

$$\Big\Downarrow$$

$$\widetilde{M}_C \equiv \begin{bmatrix} x_C \\ y_C \\ z_C \\ 1 \end{bmatrix} \equiv \begin{bmatrix} \boldsymbol{R} & \boldsymbol{t} \\ \mathbf{0} & 1 \end{bmatrix} \widetilde{M}_W \equiv \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_W \\ y_W \\ z_W \\ 1 \end{bmatrix} \equiv \boldsymbol{G}\widetilde{M}_W$$

Perspective projection matrix

Camera RF to pixel

intrinsic part

$$\begin{bmatrix} \boldsymbol{u} \\ \boldsymbol{v} \\ \boldsymbol{1} \end{bmatrix} \equiv \begin{bmatrix} f_u & 0 & u_0 & 0 \\ 0 & f_v & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_C \\ y_C \\ z_C \\ 1 \end{bmatrix}$$

$$\tilde{m} \equiv \boldsymbol{P_{int}} \widetilde{M_C}$$

World RF to Camera RF

extrinsic part

$$\begin{bmatrix} x_C \\ y_C \\ z_C \\ 1 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_W \\ y_W \\ z_W \\ 1 \end{bmatrix}$$

$$\boldsymbol{\widetilde{M_C}} \equiv \boldsymbol{G \widetilde{M_W}}$$

Putting everything together we obtain a linear model for the image formation process

$$\boxed{\tilde{m} \equiv \boldsymbol{P_{int}} \widetilde{M}_C \equiv \boldsymbol{P_{int} G} \widetilde{M}_W \equiv \boldsymbol{P} \widetilde{M}_W}$$

$\boldsymbol{P}$ is known as the perspective projection matrix (PPM).

we get a LINEAR MODEL

Canonical perspective projection

P just represents the perspective projection model at its simplest form

$\boldsymbol{P}$ is a 3x4 matrix with full rank. The most basic PPM is

does not collapse onto lower dimensions

identity matrix

$$\boldsymbol{P} \equiv [\boldsymbol{I}|\boldsymbol{0}]$$

This form is useful to understand the core operations carried out by perspective projection, i.e. scaling lateral coordinates (x, y) according to the distance from the camera (z). The above form is usually referred to as *canonical* or *standard* PPM.

A generic PPM can then be factorized as $\underline{\boldsymbol{P} \equiv \boldsymbol{A}[\boldsymbol{I}|\boldsymbol{0}]\boldsymbol{G}}$, i.e. it can be thought of as carrying out three fundamental operations one after the other:

this highlights the 3 main operations done in perspective projection

1. Convert coordinates from WRF to CRF G
2. Perform canonical perspective projection, i.e. divide by the third coordinate
3. Apply camera specific transformations A

From it, we get another useful factorization of a generic PPM as $\boldsymbol{P} \equiv \boldsymbol{A}[\boldsymbol{I}|\boldsymbol{0}]\boldsymbol{G} \equiv \boldsymbol{A}[\boldsymbol{I}|\boldsymbol{0}]\begin{bmatrix} \boldsymbol{R} & \boldsymbol{t} \\ \boldsymbol{0} & 1 \end{bmatrix} \equiv \boldsymbol{A}[\boldsymbol{R}|\boldsymbol{t}]$

the PPM is the same everywhere in the image plane

Lens distortion

the camera we use do not have a pinhole
the pinhole camera model is just an abstraction

The PPM is based on the pinhole camera model

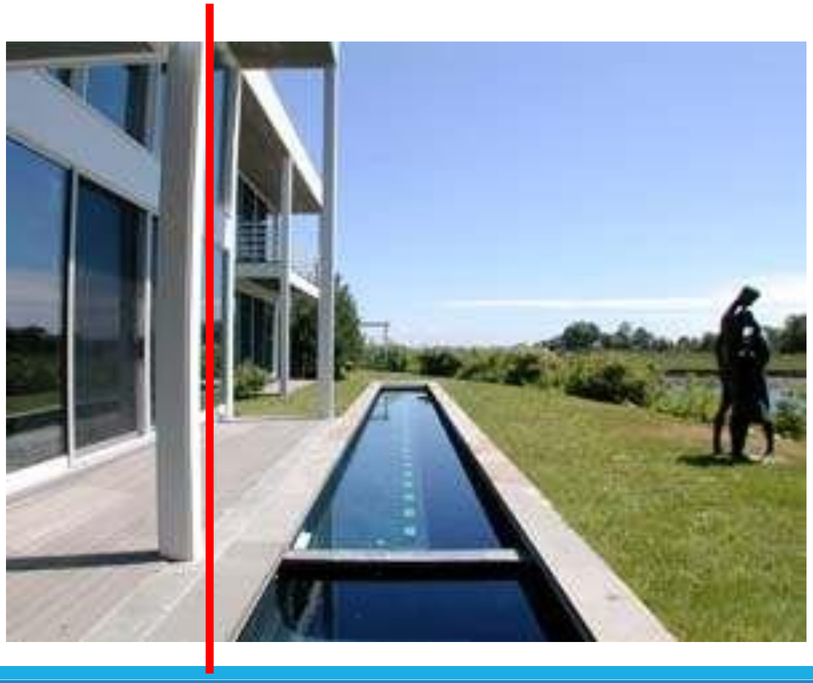
However, real lenses introduce distortions wrt to the pure pinhole model (in particular, for cheap and/or short focal length lenses)

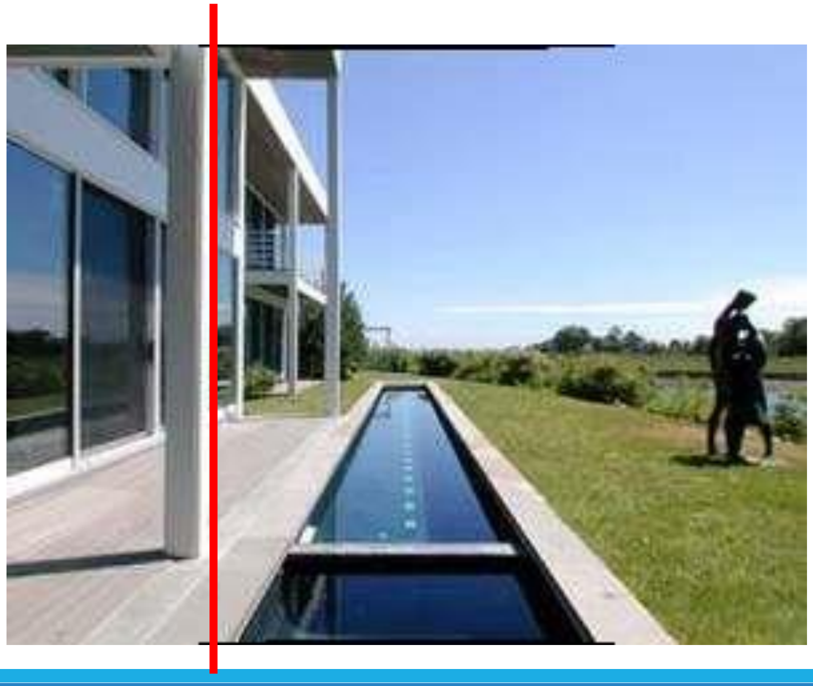
We often need to model also the effects caused by the optical distortion induced by lenses

- Lens distortion is modelled through additional parameters that do not alter the form of the PPM

lenses in real world do not preserve behavior of light, changing its path, creating a distortion

this happens because in real word lenses are not that thin, etc.





"ideal" lenses infinitely thin lenses that allow straight lines preserving light's behavior

Lens distortion

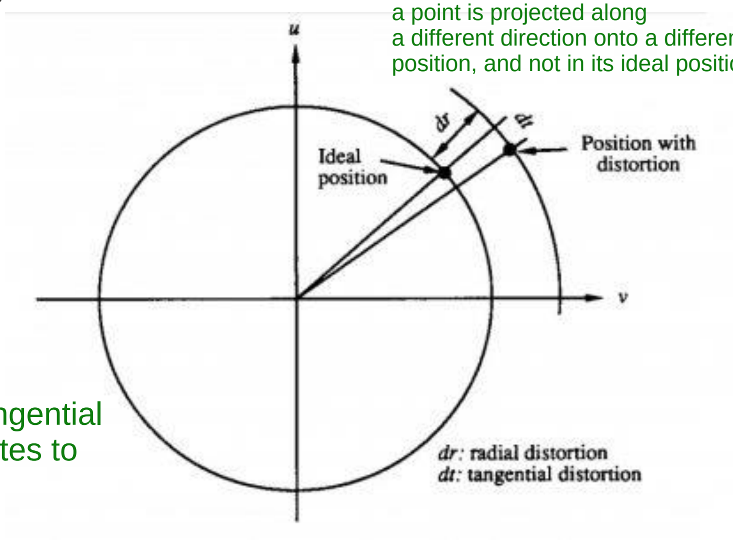
radial distortion is local, and it is stronger the farther I am from the center of the image

The most significant deviation from the ideal pinhole model is known as radial distortion (lens "curvature"):

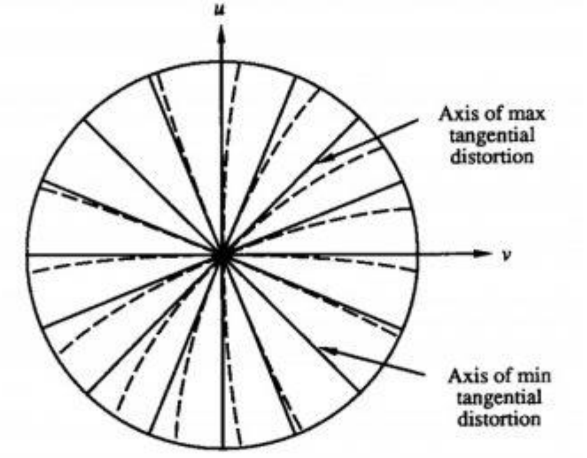
- Barrel distortion is a lens defect usually associated with wide-angle lenses, that causes straight lines to bend outwards, toward the edges of the frame
- Pincushion distortion is usually associated with telephoto lenses and it is the opposite of barrel distortion, it causes straight lines to curve inward from the edges to the centre of the frame

Second order effects are induced by tangential distortion ("misalignments" of optical components and/or defects).

a point is projected along a different direction onto a different position, and not in its ideal position

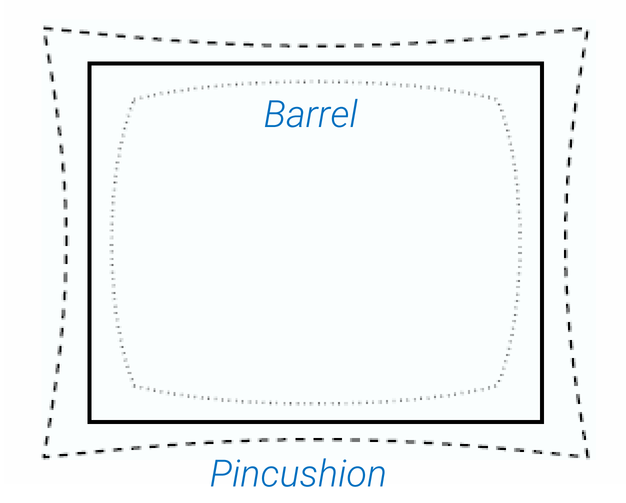


radial and tangential both contributes to distortion



u
v
Axis of max tangential distortion
Axis of min tangential distortion

the amount of tangential distortion depend on where the point is

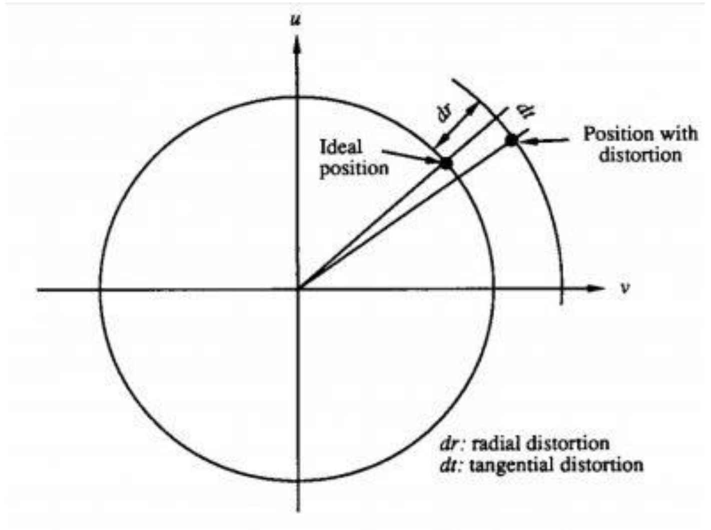


J. Weng, P. Cohen, and M. Herniou. Camera calibration with distortion models and accuracy evaluation. IEEE Trans. on PAMI, Oct. 1992.

Modelling lens distortion

Lens distortion, especially radial distortion, is inherently non-linear. The distortion causes straight lines to bend in a way that cannot be accurately captured by a linear transformation. This bending is more pronounced towards the edges of the lens and varies non-linearly with the distance from the center of the image.

Lens distortion is modelled through a non-linear transformation which maps ideal (i.e., *undistorted*) image coordinates into the observed (i.e., *distorted*) image coordinates.



ideal coords

$$\begin{bmatrix} x \\ y \end{bmatrix} = \underbrace{L(r)\begin{bmatrix} x_{undist} \\ y_{undist} \end{bmatrix}}_{\text{Radial distortion}} + \underbrace{\begin{bmatrix} dx(x_{undist}, y_{undist}, r) \\ dy(x_{undist}, y_{undist}, r) \end{bmatrix}}_{\text{Tangential distortion}}$$

influenced by the distance r from the distortion centre, which is usually assumed to correspond with the piercing point $\mathbf{c} = [0,0]^T$, $r = \sqrt{(x_{undist})^2 + (y_{undist})^2}$

the distortion becomes way bigger proportionally to the distance

J. Weng, P. Cohen, and M. Herniou. Camera calibration with distortion models and accuracy evaluation. IEEE Trans. on PAMI, Oct. 1992.

e.g. fish eye cameras introduce a lot of radial distortion

Modelling lens distortion

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

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

The tangential distortion vector is instead approximated as follows

$$\begin{bmatrix} dx(x_{undist}, y_{undist}, r) \\ dy(x_{undist}, y_{undist}, r) \end{bmatrix} = \begin{bmatrix} 2p_1 x_{undist}\, y_{undist} + p_2(r^2 + 2x_{undist}^2) \\ 2p_1(r^2 + 2y_{undist}^2) + p_2 y_{undist}\, x_{undist} \end{bmatrix}$$

The radial distortion coefficients $k_1, k_2, \dots, k_n$, together with the two tangential distortion coefficients p_1 and p_2 form the set of the lens distortion parameters, which extends the set of intrinsic parameters required to build a realistic camera model.

where lens distortion "suppression" takes place in the image formation process?

Lens distortion within the image formation process

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. discretization/pixelization

Accordingly, the image formation flow with lenses can be summarized as follows:

1. Transformation of 3D points from the WRF to the CRF, according to extrinsic parameters: $\begin{bmatrix} x_C \\ y_C \\ z_C \\ 1 \end{bmatrix} \equiv G\tilde{M}_W$

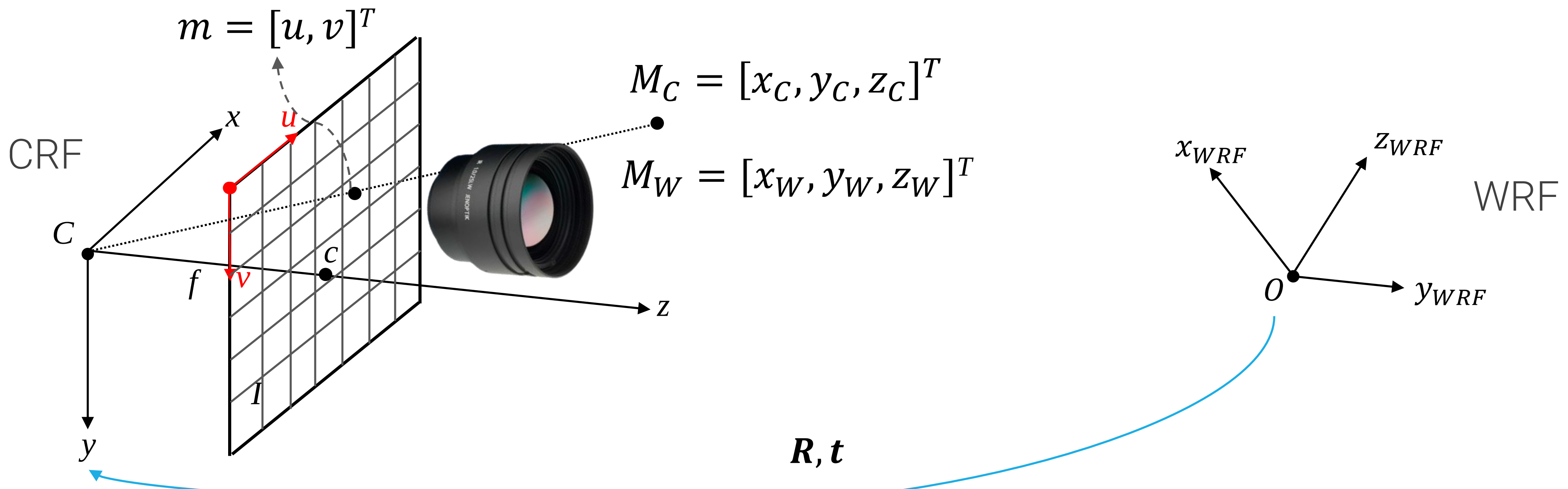
dividing by z coord.

2. Canonical perspective projection (i.e. scaling by the third coordinate): $\begin{bmatrix} x_{undist} \\ y_{undist} \end{bmatrix} = \begin{bmatrix} x_C/z_C \\ y_C/z_C \end{bmatrix}$

3. Non-linear mapping due to lens distortion: $\begin{bmatrix} x \\ y \end{bmatrix} = L(r) \begin{bmatrix} x_{undist} \\ y_{undist} \end{bmatrix} + \begin{bmatrix} dx(x_{undist}, y_{undist}, r) \\ dy(x_{undist}, y_{undist}, r) \end{bmatrix}$

4. Mapping from image coordinates to pixels coordinates according to the intrinsic parameters: $\tilde{m} \equiv \begin{bmatrix} ku \\ kv \\ k \end{bmatrix} \equiv \boldsymbol{A} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$

Complete camera model



$m = [u, v]^T$
$M_C = [x_C, y_C, z_C]^T$
$M_W = [x_W, y_W, z_W]^T$
CRF
x
u
C
c
f
v
z
I
y
x_{WRF}
z_{WRF}
WRF
O
y_{WRF}
$\boldsymbol{R}, \boldsymbol{t}$

2D point
[pixels]

$$m = \begin{bmatrix} u \\ v \end{bmatrix}$$

4 Intrinsic parameters
+ Distortion coefficients

3D point in CRF
[Metric, e.g. mm]

$$M_C = \begin{bmatrix} x_C \\ y_C \\ z_C \end{bmatrix}$$

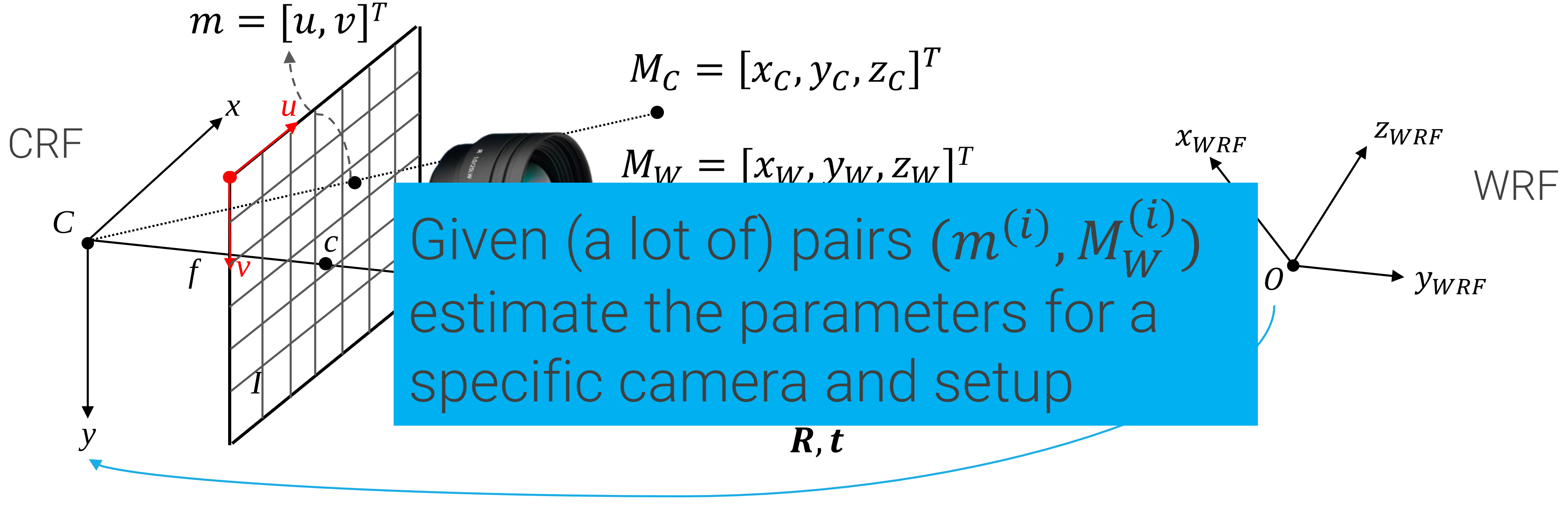
6 Extrinsic parameters

3D point in WRF
[Metric, e.g. mm]

$$M_W = \begin{bmatrix} x_W \\ y_W \\ z_W \end{bmatrix}$$

What's next? Camera calibration

intrinsic parameters: the do not change, they exist within the chosen model -> this means fixed focal length then fixed pinhole models



$m = [u, v]^T$
$M_C = [x_C, y_C, z_C]^T$
$M_W = [x_W, y_W, z_W]^T$
CRF
WRF
C
x
y
u
v
f
c
I
x_{WRF}
z_{WRF}
y_{WRF}
O
$\boldsymbol{R}, \boldsymbol{t}$

Given (a lot of) pairs $(m^{(i)}, M_W^{(i)})$ estimate the parameters for a specific camera and setup

Known

2D point [pixels]

$$m = \begin{bmatrix} u \\ v \end{bmatrix}$$

4 Intrinsic parameters
Distortion coefficients

3D point in CRF [Metric, e.g. mm]

$$M_C = \begin{bmatrix} x_C \\ y_C \\ z_C \end{bmatrix}$$

Unknown

6 Extrinsic parameters

3D point in WRF [Metric, e.g. mm]

$$M_W = \begin{bmatrix} x_W \\ y_W \\ z_W \end{bmatrix}$$

Known