

# Lecture 1

## Image Formation Model

---

IMAGE PROCESSING AND COMPUTER VISION – PART 2

SAMUELE SALTI

# Image formation (recap)

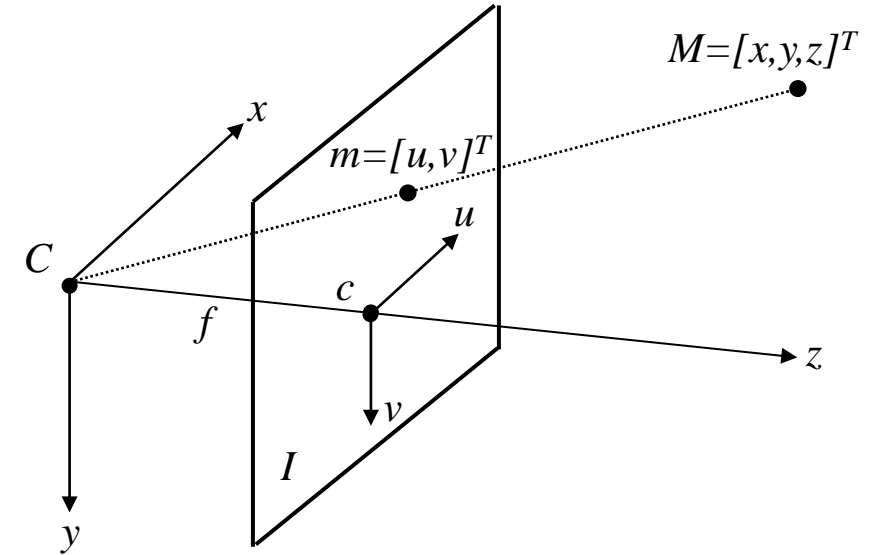
Images are 2D **perspective projections** of the 3D world.

**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)

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 vertical axis)
- $z$  axis is the optical axis



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

# Why do we need an image formation model?

---



# NeRF: Neural Radiance Fields

---

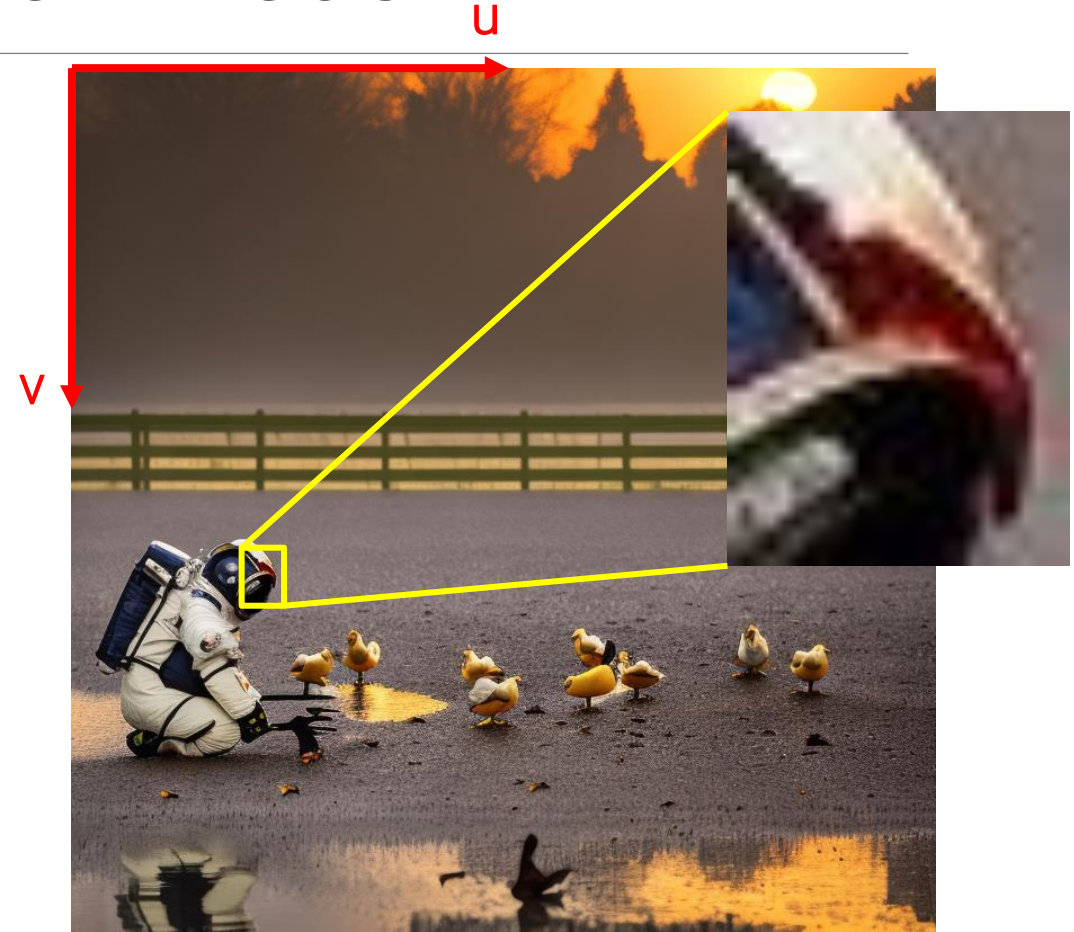


Ben Mildenhall et al., "NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis", ECCV 2020.

# 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**.

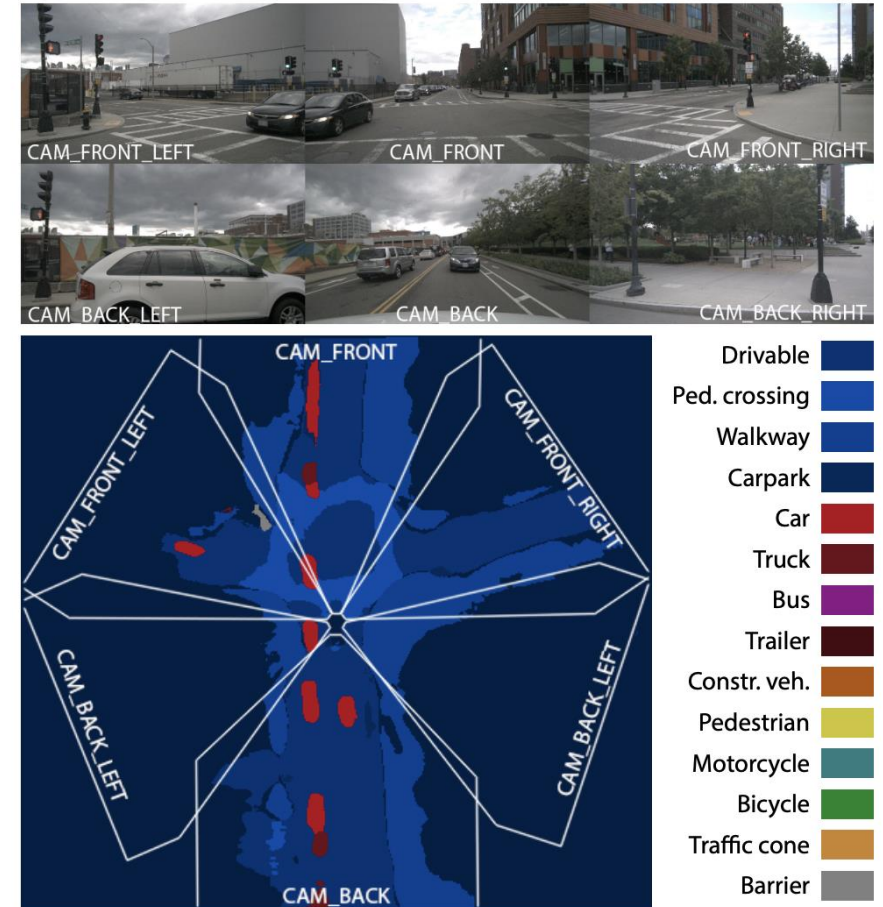


<https://stability.ai/blog/stable-diffusion-v2-release>

# A more realistic camera model

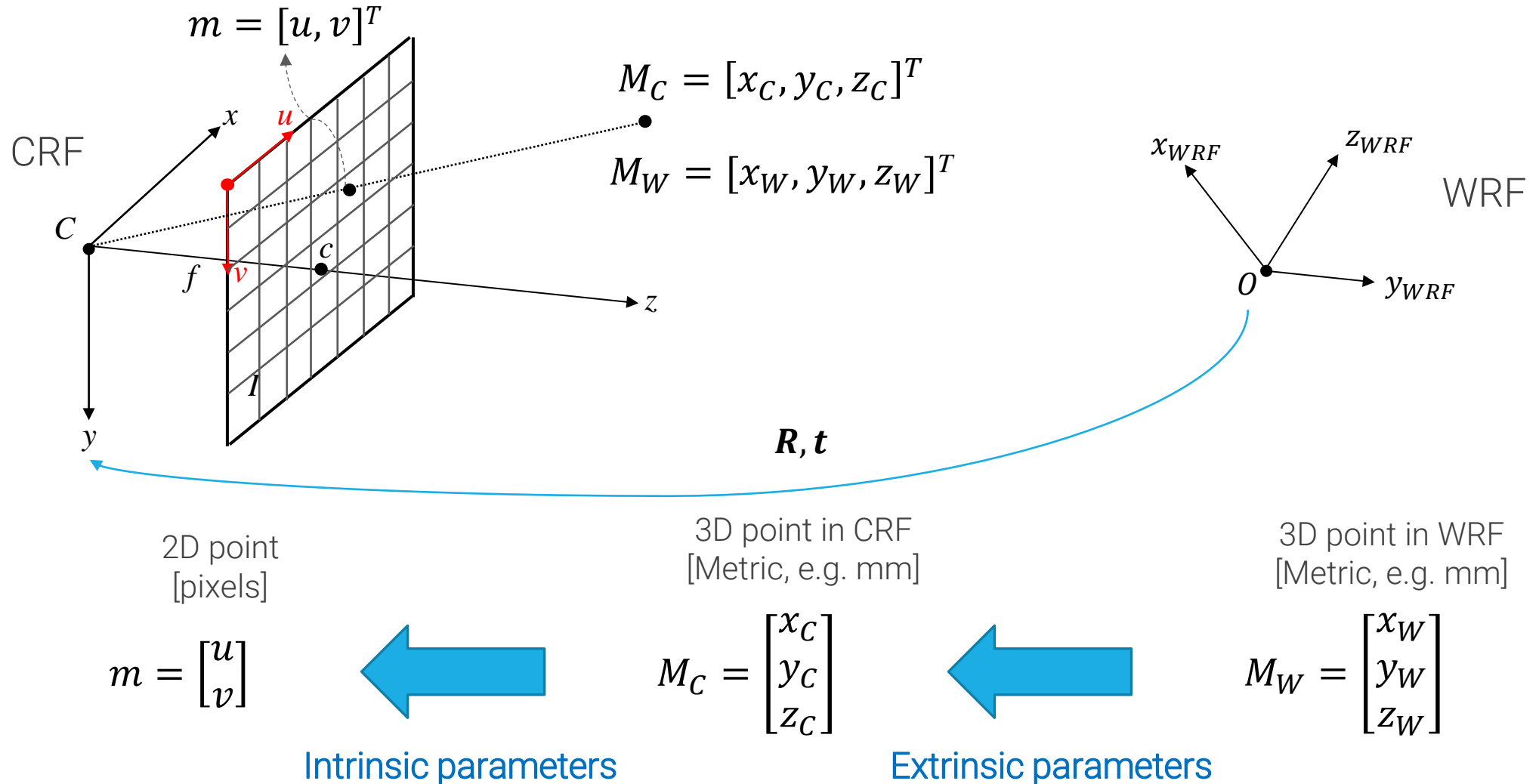
To create a more realistic camera model we need to consider three additional issues:

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**.



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

# Complete Forward Imaging model: 3D to 2D

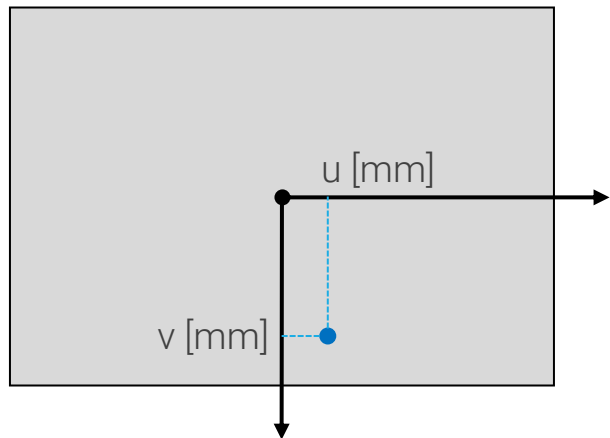


# Image Pixelization

Digitization can be accounted for by including into the projection equations **the pixel size  $\Delta u$  and  $\Delta v$**  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

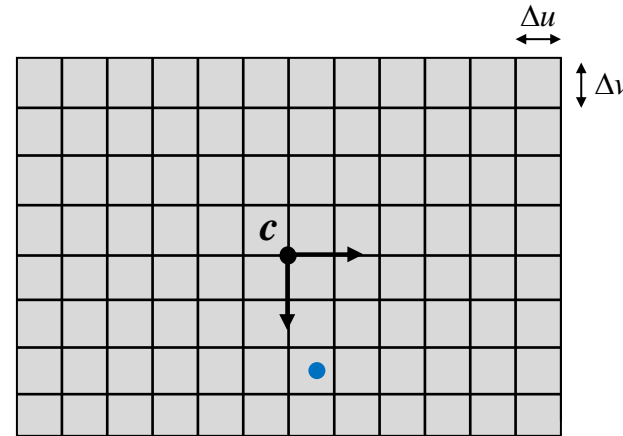


$$u = \frac{f}{z_C} x_C$$

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



Real case: Image **sensor**



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

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

$\Delta u$  = horizontal pixel size in mm (quantization step)

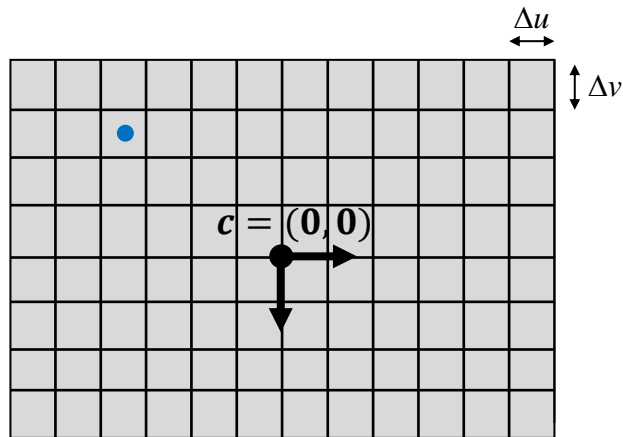
$\Delta v$  = vertical pixel size in mm (quantization step)



# Origin of the image reference frame

Since we do not use negative pixel indices when accessing images, we also need to model the translation of the piercing point (intersection between the optical axis and the image plane) with respect to the origin of the image coordinate system (top-left corner of the image)

Ideal origin as piercing point

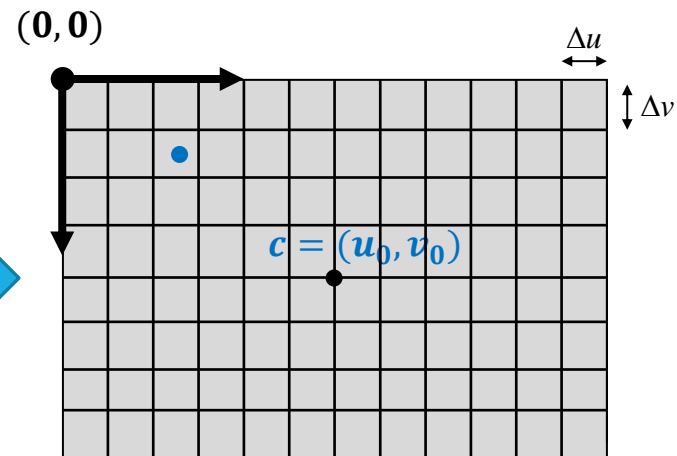


$$u = \frac{1}{\Delta u} \frac{f}{z_C} x_C$$

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



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

---

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

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

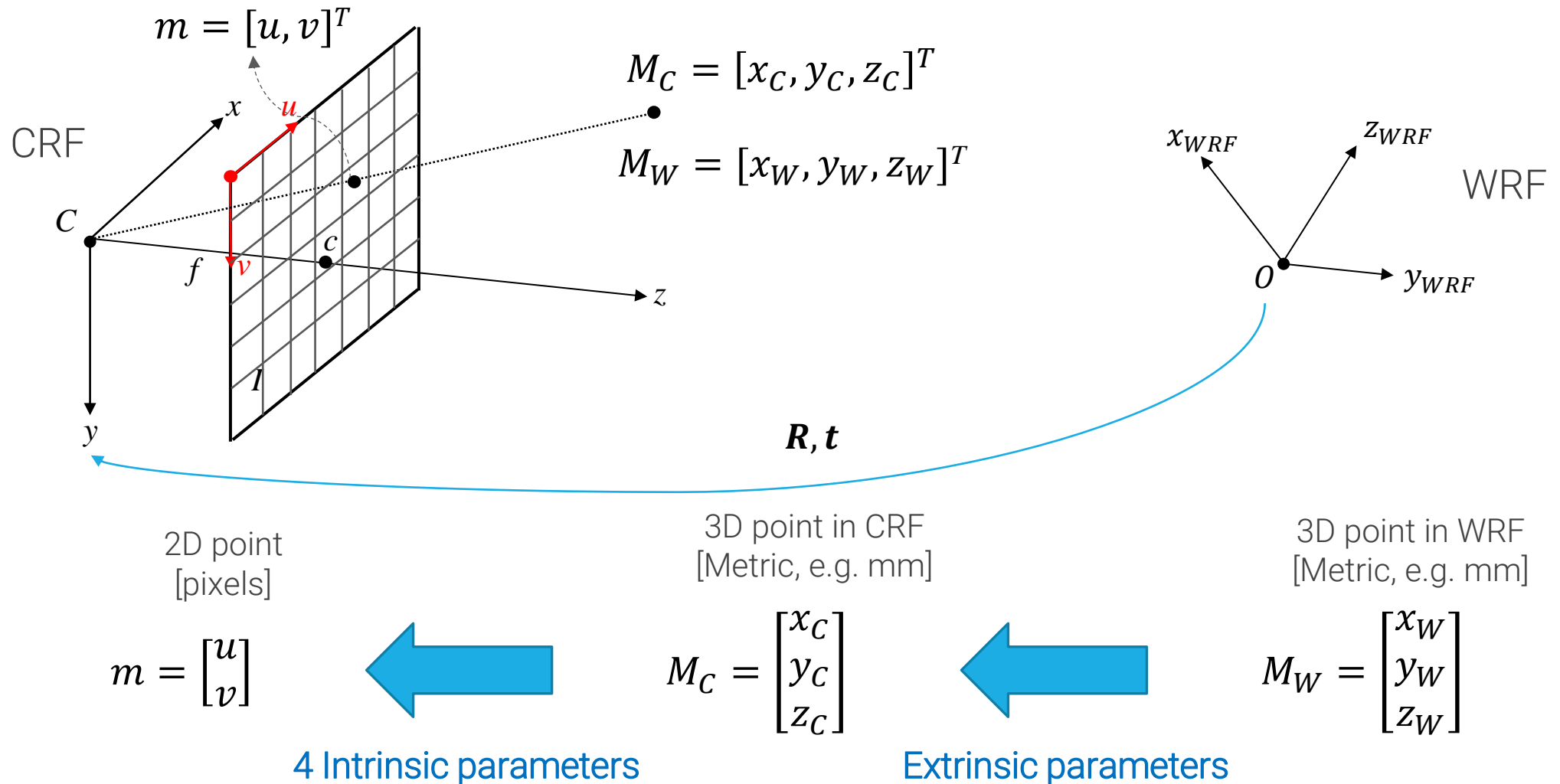
It is common to consider all multiplicative parameters in each equation as one parameter, i.e. to write

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

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



# 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  $\mathbf{R}$ )
- A translation (expressed by a 3x1 translation vector  $\mathbf{t}$ )

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

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

# Rotation matrix

---

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

Any two vectors ***a*** and ***b*** are orthonormal if and only if

$$\begin{array}{ll} \langle \mathbf{a}, \mathbf{b} \rangle = \mathbf{a}^T \mathbf{b} = 0 & \text{and } \|\mathbf{a}\|_2 = \|\mathbf{b}\|_2 = 1 \\ \text{(orthogonality)} & \text{(unit length)} \end{array}$$

For an orthonormal matrix ***R*** it is then true that

$$\mathbf{R} \mathbf{R}^T = \mathbf{R}^T \mathbf{R} = \mathbf{I}$$

i.e. its inverse is its transpose matrix.

What are the coordinates ***C<sub>W</sub>*** of the optical centre ***C*** in the world reference frame?

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

# Extrinsic parameters

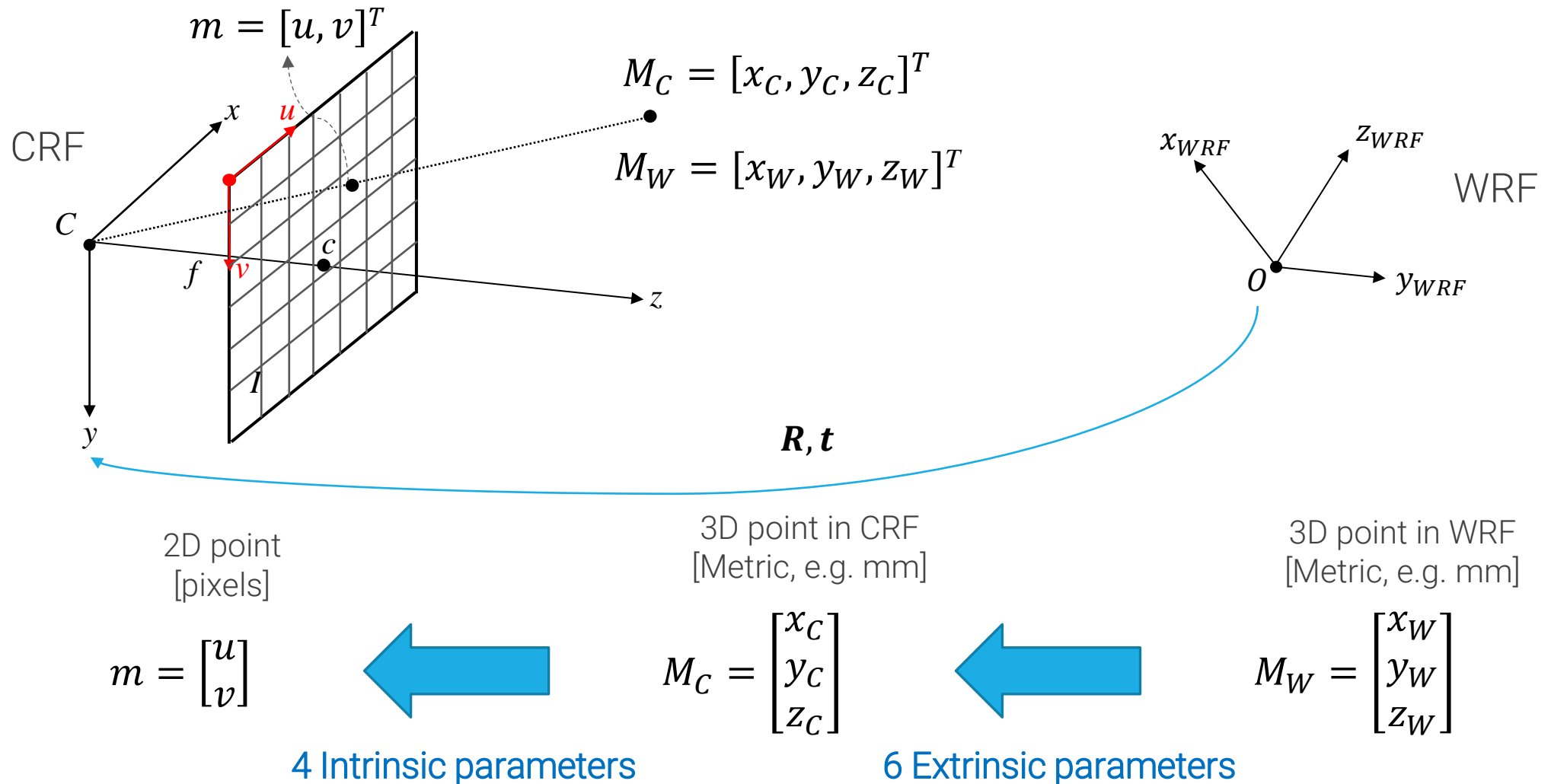
---

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 6** (3 translation parameters, 3 rotation parameters). They encode the position of the camera with respect to the WRF.

# Complete Forward Imaging model: 3D to 2D



# Can we just combine the equations we have?

---

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

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



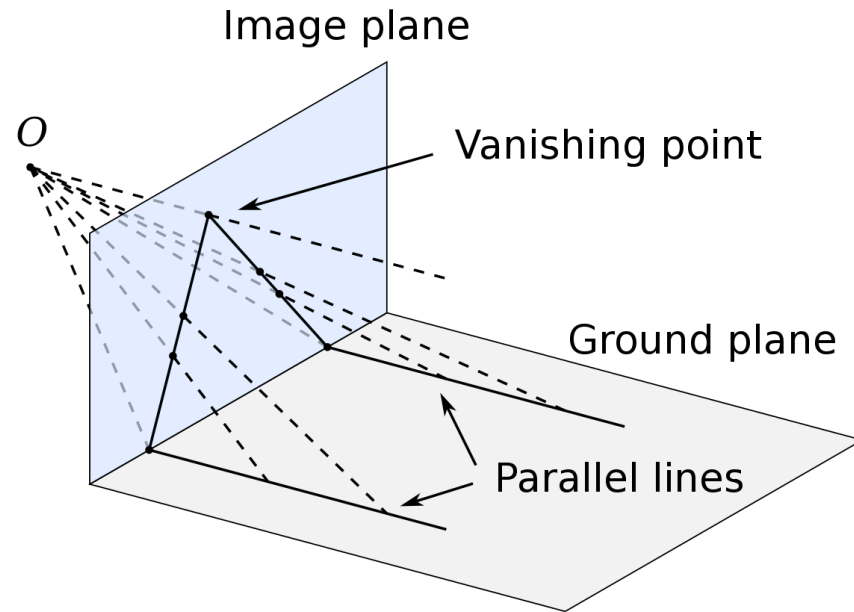
$$\begin{cases} u = f_u \frac{r_{11}x_W + r_{12}y_W + r_{13}z_W + t_1}{r_{31}x_W + r_{32}y_W + r_{33}z_W + t_3} + u_0 \\ v = f_v \frac{r_{21}x_W + r_{22}y_W + r_{23}z_W + t_2}{r_{31}x_W + r_{32}y_W + r_{33}z_W + t_3} + v_0 \end{cases}$$

Yes, we can. But it gives us a **non-linear model**, which is cumbersome to use and to estimate.

Goal #1: Can we make it linear?



# Points at infinity and vanishing point



The **vanishing point** is the **2D point** on the image where the **3D points at infinity of a set of parallel lines** are projected by the camera.

**Goal #2:** Can we have a model that can handle and project points at infinity?



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 (they “meet in points at infinity”) and
- points at infinity cannot be represented.

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

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

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

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

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

“equivalent”, not “equal”

# Projective spaces

---

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

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

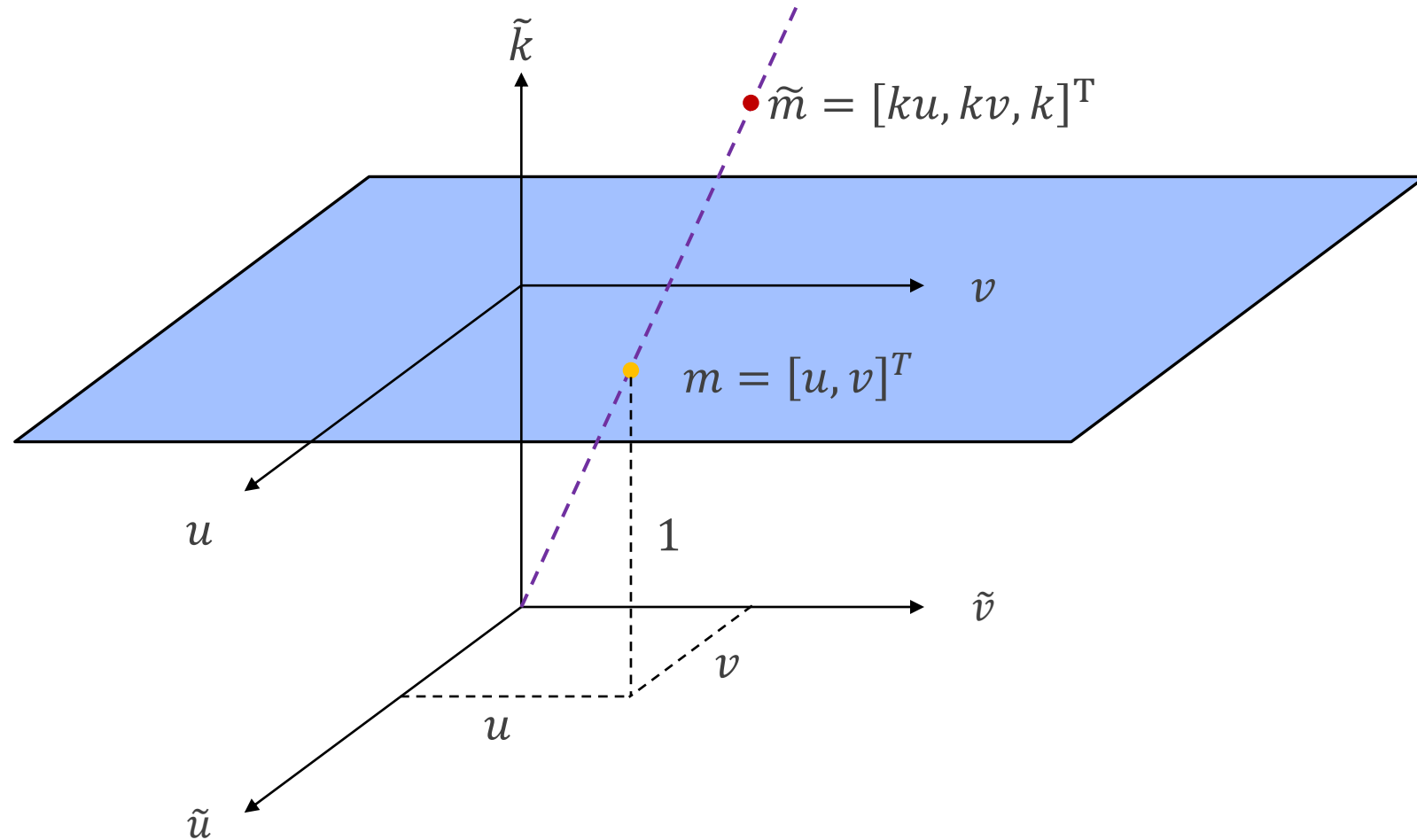
These are referred to as the **homogeneous coordinates** (a.k.a. projective coordinates) of the 2D point having Euclidean coordinates  $[u, v]^T$ . The space associated with the homogeneous representation is called **Projective Space**, denoted as  $\mathbb{P}^2$ .

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

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

# What have we done?

---



# 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:  $\tilde{m} \equiv \begin{bmatrix} m \\ 1 \end{bmatrix} \equiv \begin{bmatrix} u_0 + \lambda a \\ v_0 + \lambda b \\ 1 \end{bmatrix} \equiv \begin{bmatrix} \frac{u_0}{\lambda} + a \\ \frac{v_0}{\lambda} + b \\ \frac{1}{\lambda} \end{bmatrix}$

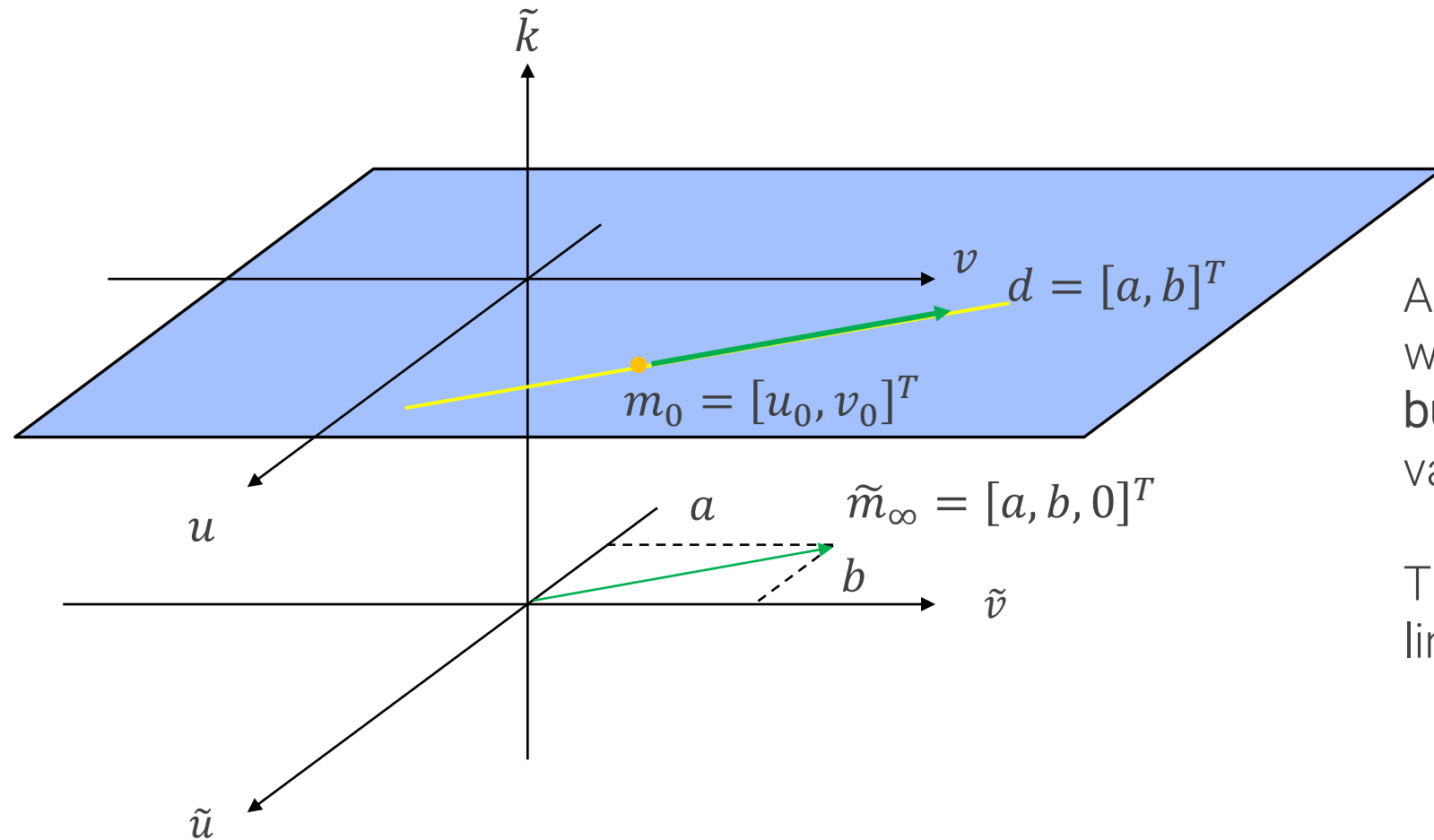
By taking the limit with  $\lambda \rightarrow \infty$  we obtain the projective coordinates of the

point at infinity of the given line:  $\tilde{m}_\infty = \lim_{\lambda \rightarrow \infty} \tilde{m} \equiv \begin{bmatrix} a \\ b \\ 0 \end{bmatrix}$

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

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

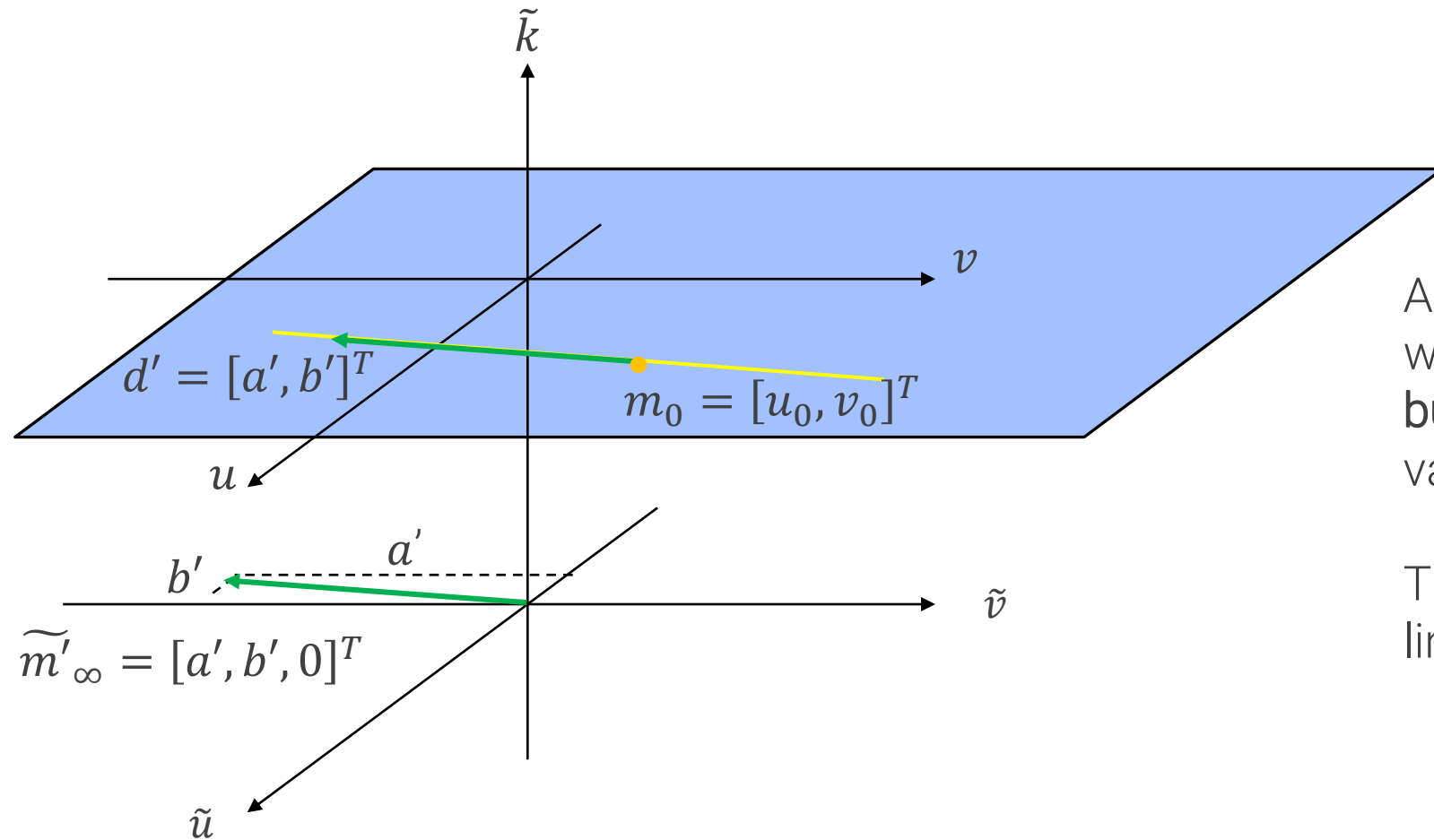
# Points at infinity



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  $\mathbb{P}^2$  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:  $\tilde{M} = \begin{bmatrix} M \\ 1 \end{bmatrix} = \begin{bmatrix} x_0 + \lambda a \\ y_0 + \lambda b \\ z_0 + \lambda c \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{x_0}{\lambda} + a \\ \frac{y_0}{\lambda} + b \\ \frac{z_0}{\lambda} + c \\ \frac{1}{\lambda} \end{bmatrix}$

By taking the limit with  $\lambda \rightarrow \infty$  we obtain the projective coordinates of the

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



# 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.
- The projective representation features one additional coordinate, referred to here as  $k$ , with respect to 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  $k \neq 0$
- Projective Spaces allow to represent and process both the ordinary and the ideal points of Euclidean Spaces.

# Perspective Projection in homogeneous coordinates

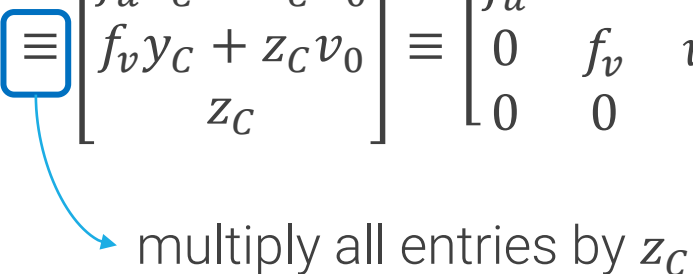
Let's go back to the non-linear transformations between 3D coordinates in the CRF and image coordinates:

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

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  $\sim$ ) then the equations can be written as matrix multiplication, i.e. [in linear form](#):

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

 multiply all entries by  $z_C$

# Perspective Projection in homogeneous coordinates

---

In homogeneous coordinates the perspective projection becomes a **linear transformation**:

$$\tilde{m} \equiv \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \equiv \begin{bmatrix} ku \\ kv \\ k \end{bmatrix} \equiv \begin{bmatrix} f_u & 0 & u_0 & 0 \\ 0 & f_v & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix} \equiv \mathbf{P}_{int} \tilde{M}_C$$

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  $k\tilde{m} = \mathbf{P}_{int}\tilde{M}_C$  or as  $\tilde{m} \approx \mathbf{P}_{int}\tilde{M}_C$

# Points at infinity $\neq$ vanishing point

---

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, i.e. compute their vanishing point, as

$$\tilde{m}_{\infty} \equiv \mathbf{P}_{int} \begin{bmatrix} a \\ b \\ c \\ 0 \end{bmatrix} \equiv \begin{bmatrix} f_u & 0 & u_0 & 0 \\ 0 & f_v & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ 0 \end{bmatrix} \equiv \begin{bmatrix} f_u a + cu_0 \\ f_v b + cv_0 \\ c \end{bmatrix} \text{ which, in Euclidean coords is } m_{\infty} = \begin{bmatrix} f_u \frac{a}{c} + u_0 \\ f_v \frac{b}{c} + v_0 \end{bmatrix}$$

# Intrinsic Parameter Matrix

---

$$\tilde{m} \equiv \begin{bmatrix} f_u & 0 & u_0 & 0 \\ 0 & f_v & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix} \equiv \mathbf{P}_{int} \widetilde{\mathbf{M}}_C \equiv [\mathbf{A} | \mathbf{0}] \widetilde{\mathbf{M}}_C$$

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

It is always an upper right triangular matrix.

# Intrinsic Parameter Matrix

---

A more general model would include a 5<sup>th</sup> 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  **$\mathbf{G}$** , the **extrinsic parameter matrix**

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



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

# Perspective projection matrix

Camera RF to pixel

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

$$\tilde{m} \equiv \mathbf{P}_{int} \tilde{M}_C$$

World RF to Camera RF

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

$$\tilde{M}_C \equiv \mathbf{G} \tilde{M}_W$$

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

$$\tilde{m} \equiv \mathbf{P}_{int} \tilde{M}_C \equiv \mathbf{P}_{int} \mathbf{G} \tilde{M}_W \equiv \mathbf{P} \tilde{M}_W$$

The 3x4 matrix  $\mathbf{P}$  is known as the perspective projection matrix (PPM).



# Canonical perspective projection

---

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

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

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

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

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

# Lens distortion

---

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

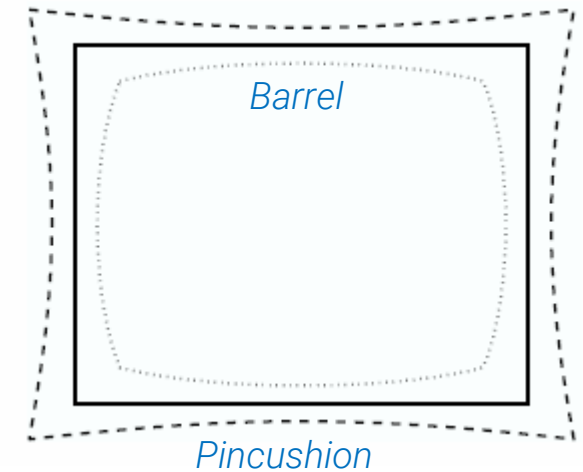
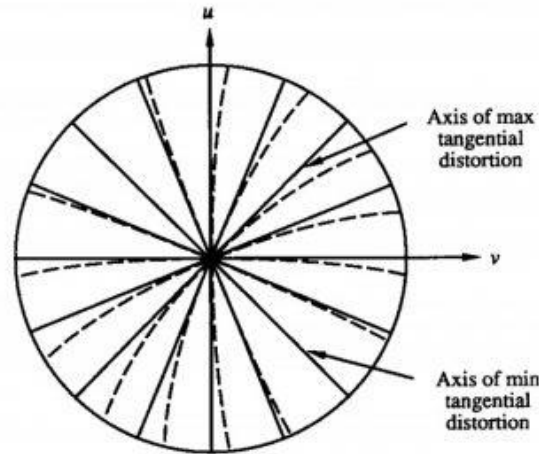
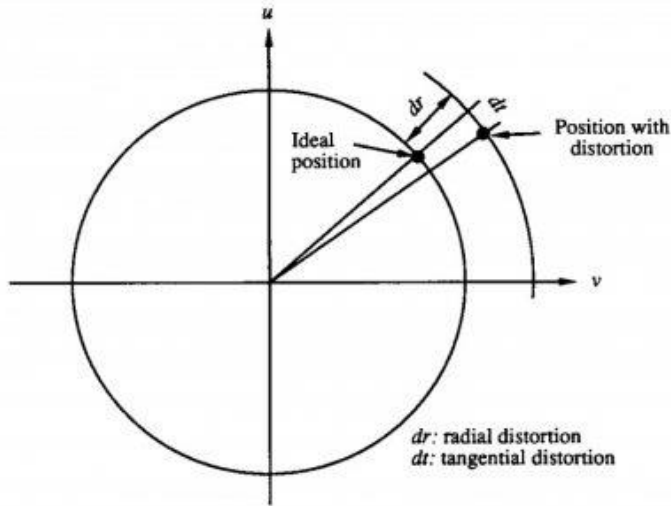


# Lens distortion

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

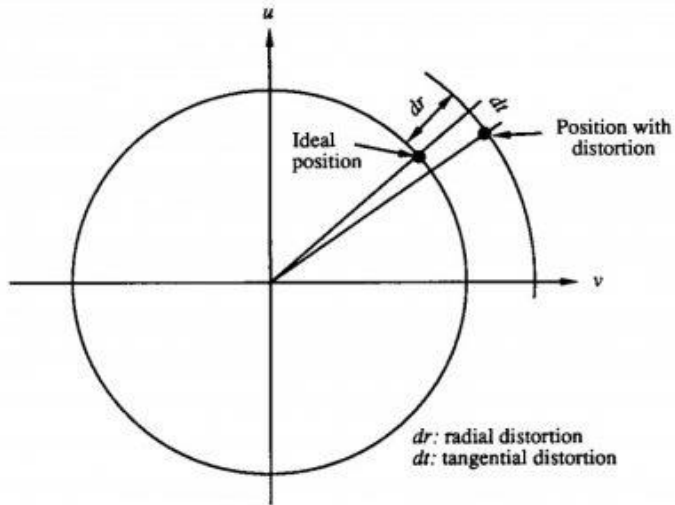
Second order effects are induced by **tangential distortion** (“misalignments” of optical components and/or defects).



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



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

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

# 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

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

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.

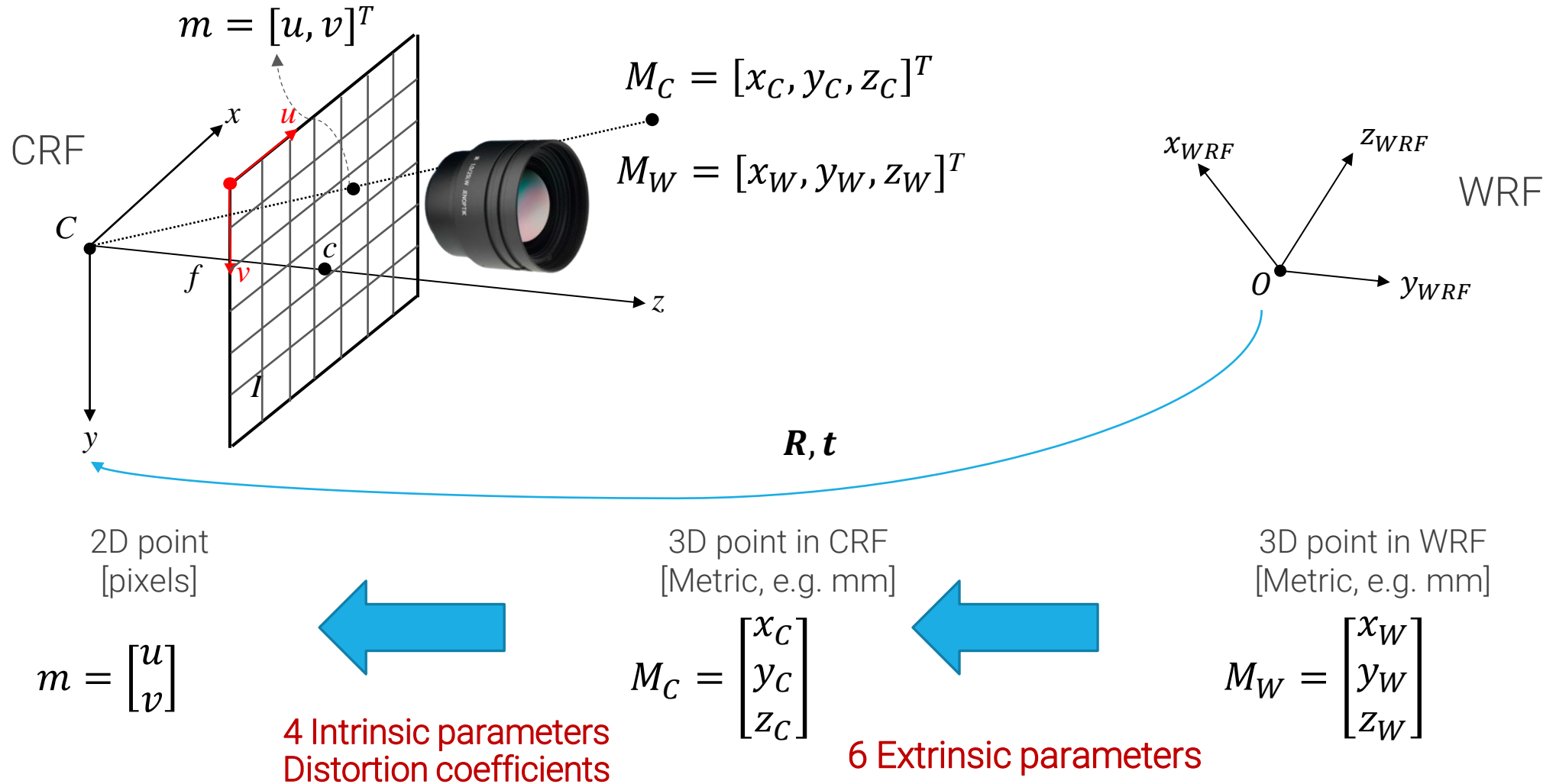
# Lens distortion within the image formation process

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.

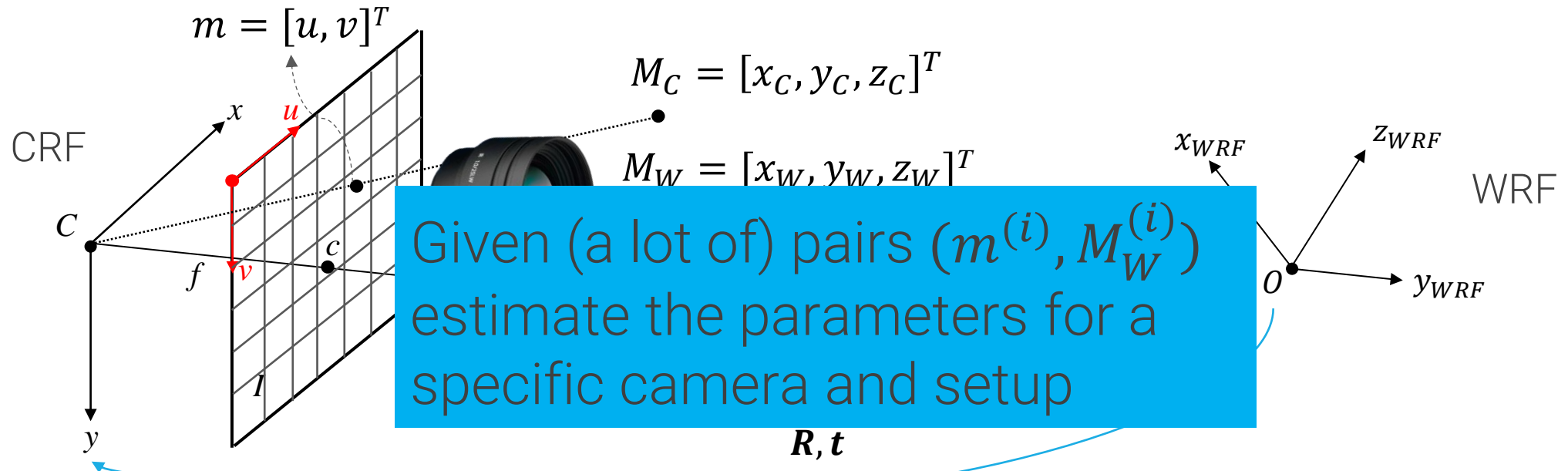
Accordingly, the image formation process with lenses can be summarized as follows:

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

# Complete camera model



# What's next? Camera calibration



Known

2D point  
[pixels]

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

4 Intrinsic parameters  
Distortion coefficients

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

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