The perspective camera model

Thomas Opsahl

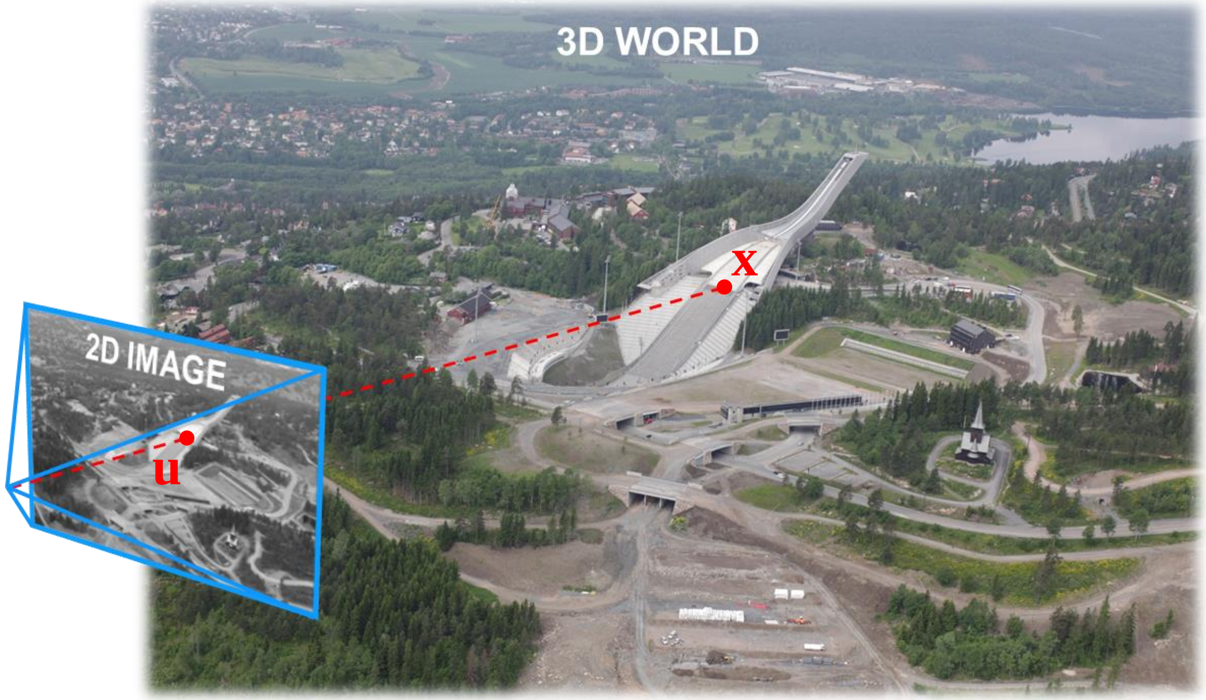
2023

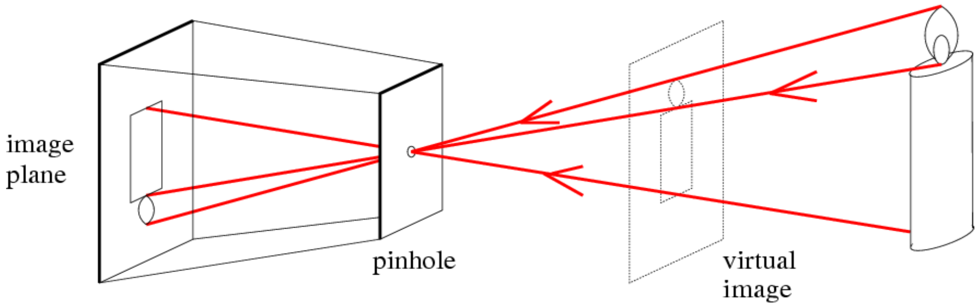
The perspective camera model

A mathematical model that with some adaptations can be used to accurately describe the viewing geometry of most cameras

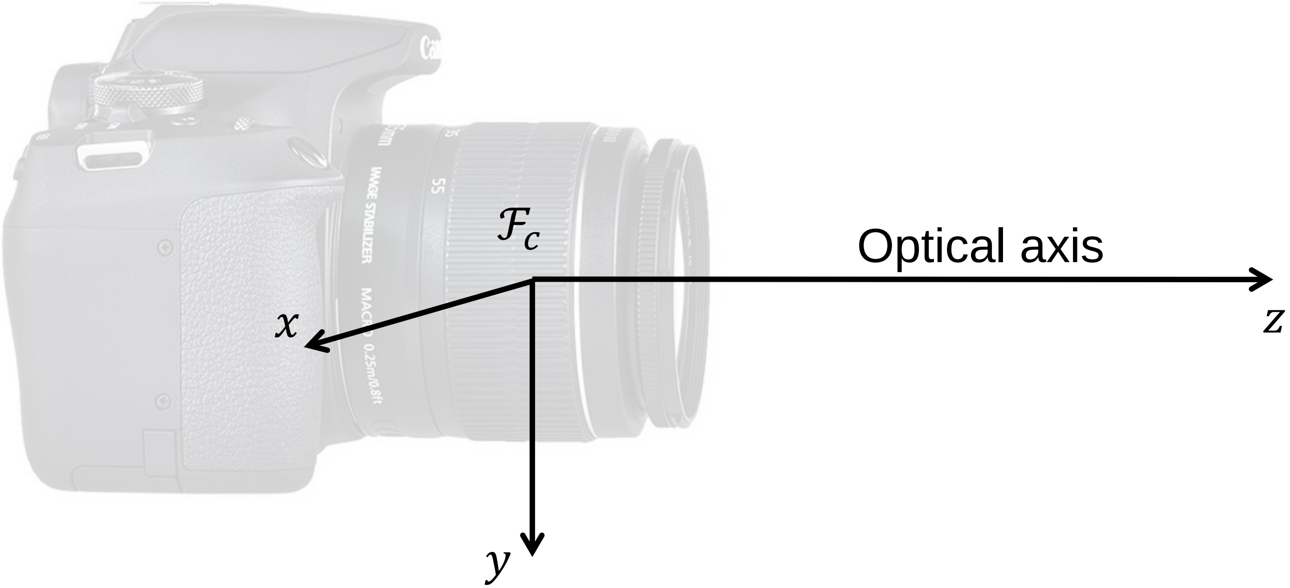
It describes how a perspective camera, i.e. a camera with pinhole geometry, maps 3D points in the world to 2D points in the image

A key characteristic of perspective cameras is that they preserve straight lines





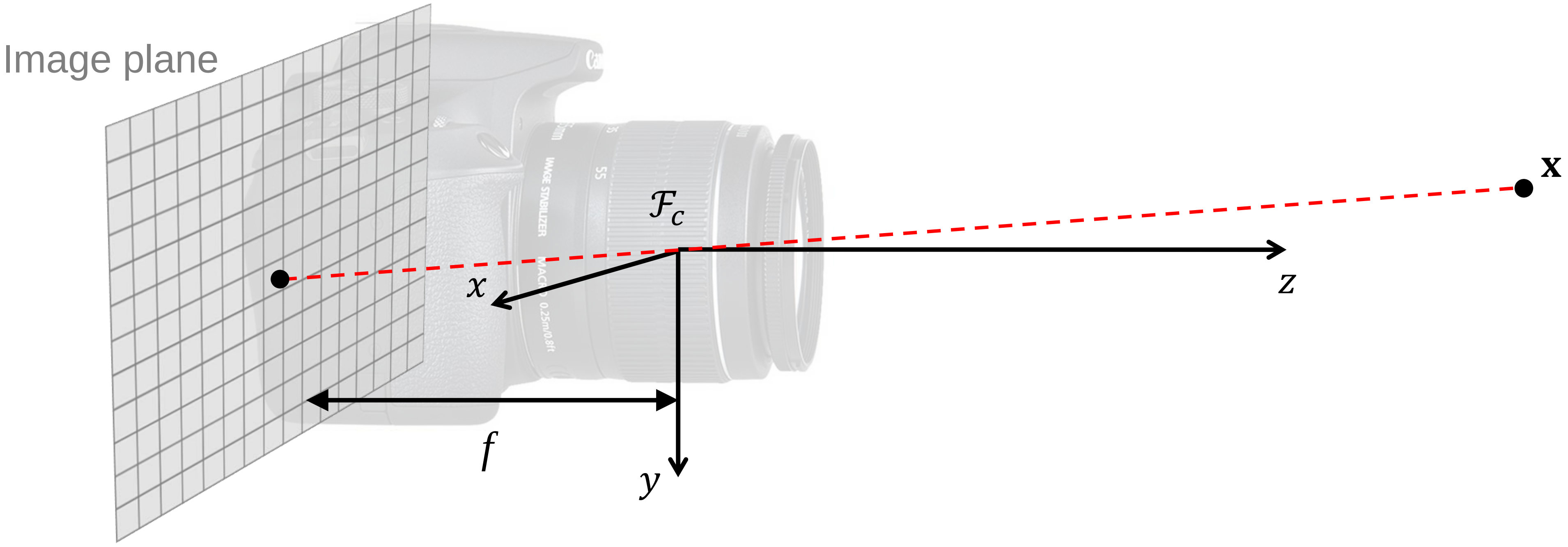
The perspective camera model



The camera is represented by a 3D frame $\mathcal{F}_c$ with its origin in the camera's projective center (pinhole), z-axis pointing forwards, x-axis to the right and y-axis pointing downwards

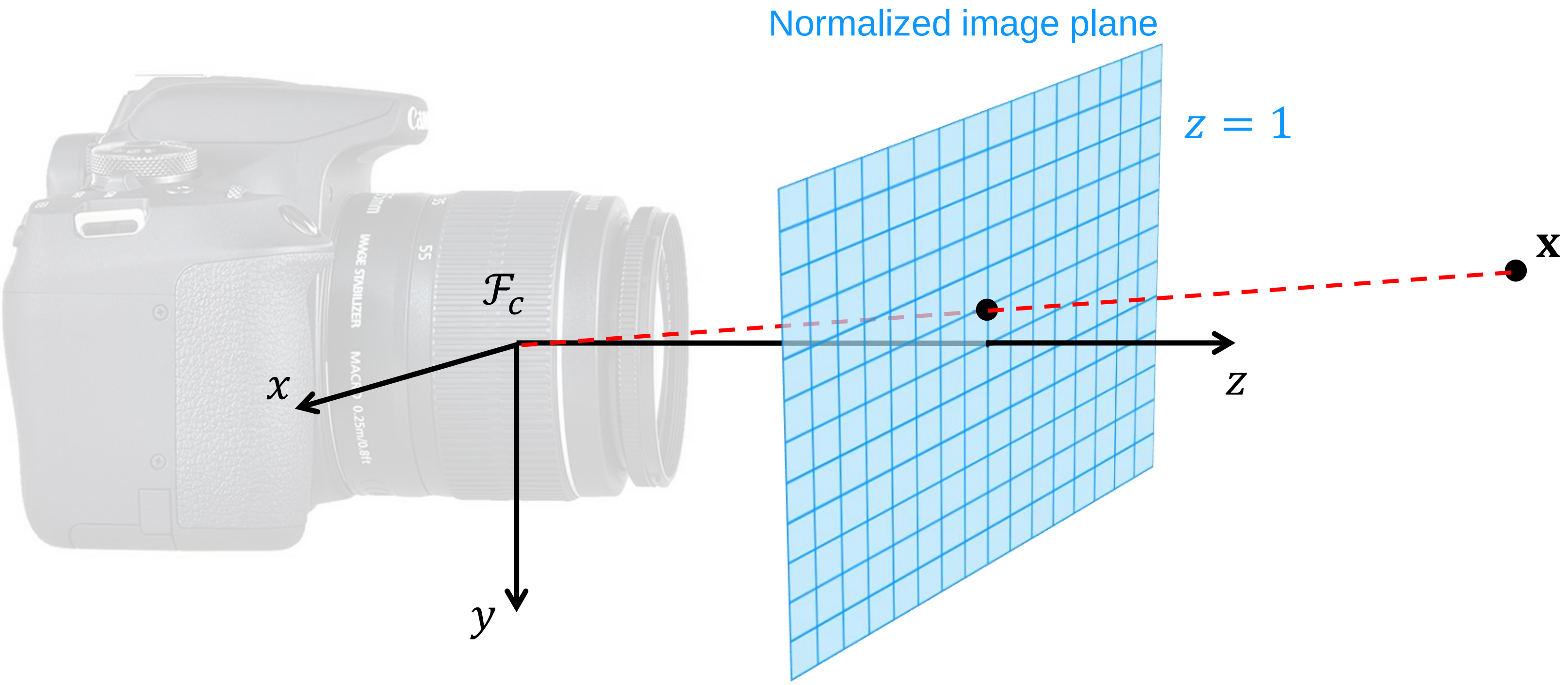
The z-axis is commonly referred to as the cameras **optical axis**

The perspective camera model



According to the pinhole geometry, the imaging process is a central projection onto the image plane a distance f (focal length) behind the pinhole

The perspective camera model

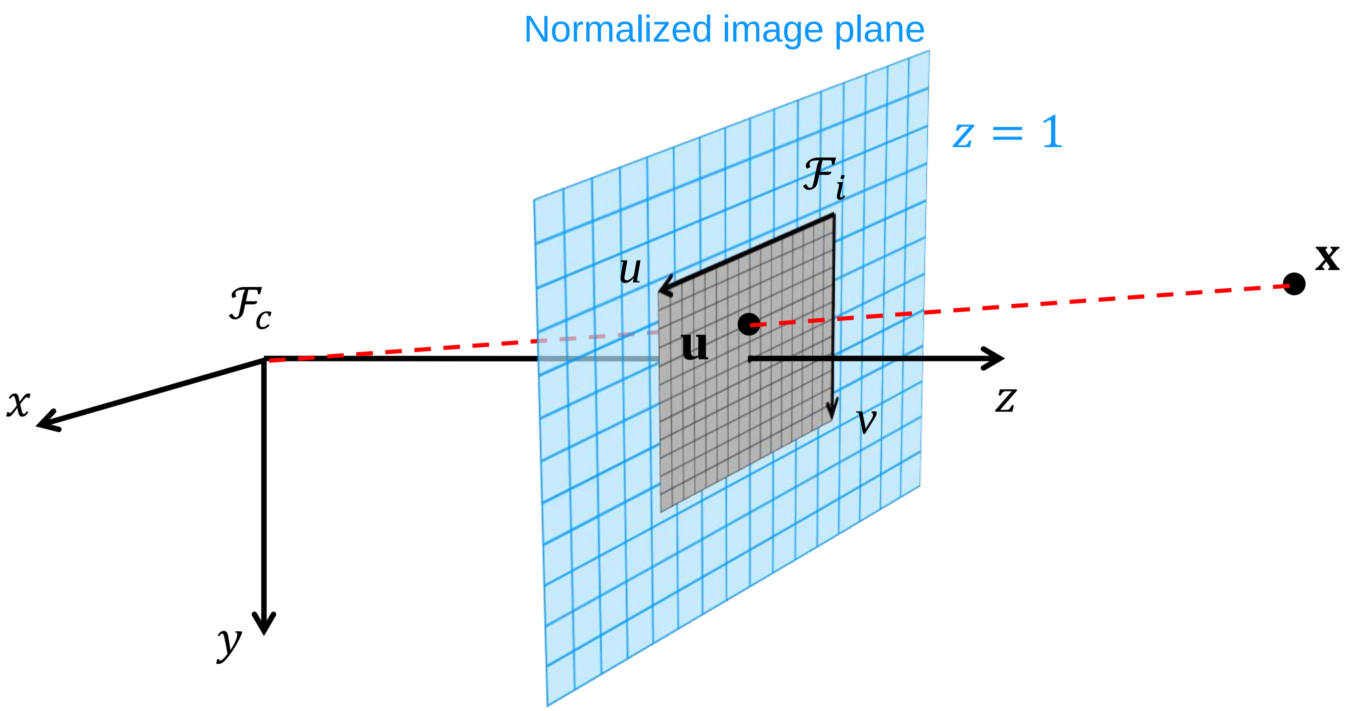


The normalized image plane is more convenient to work with than the image plane

The normalized image plane has a fixed position in $\mathcal{F}_c$ defined by $z = 1$

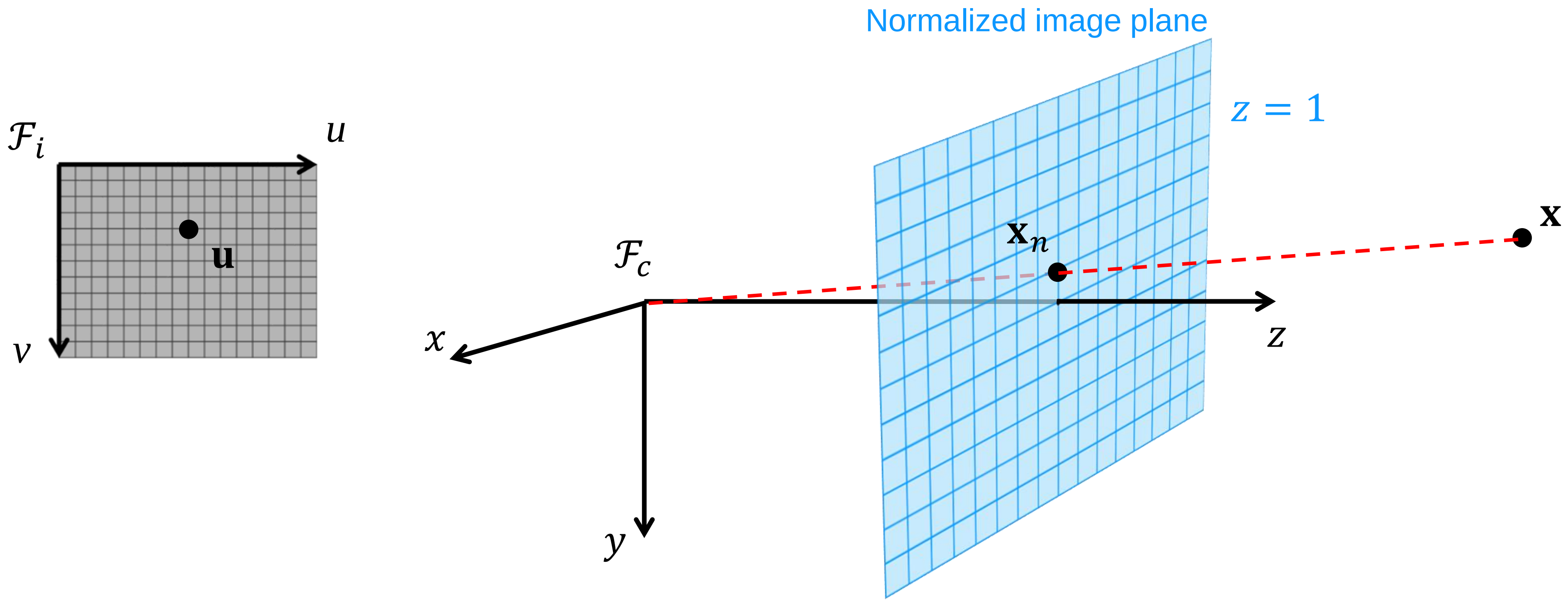
– The image plane is camera specific (not necessarily $z = -f$)

The perspective camera model



The image is represented by a 2D frame $\mathcal{F}_i$ that spans the normalized image plane

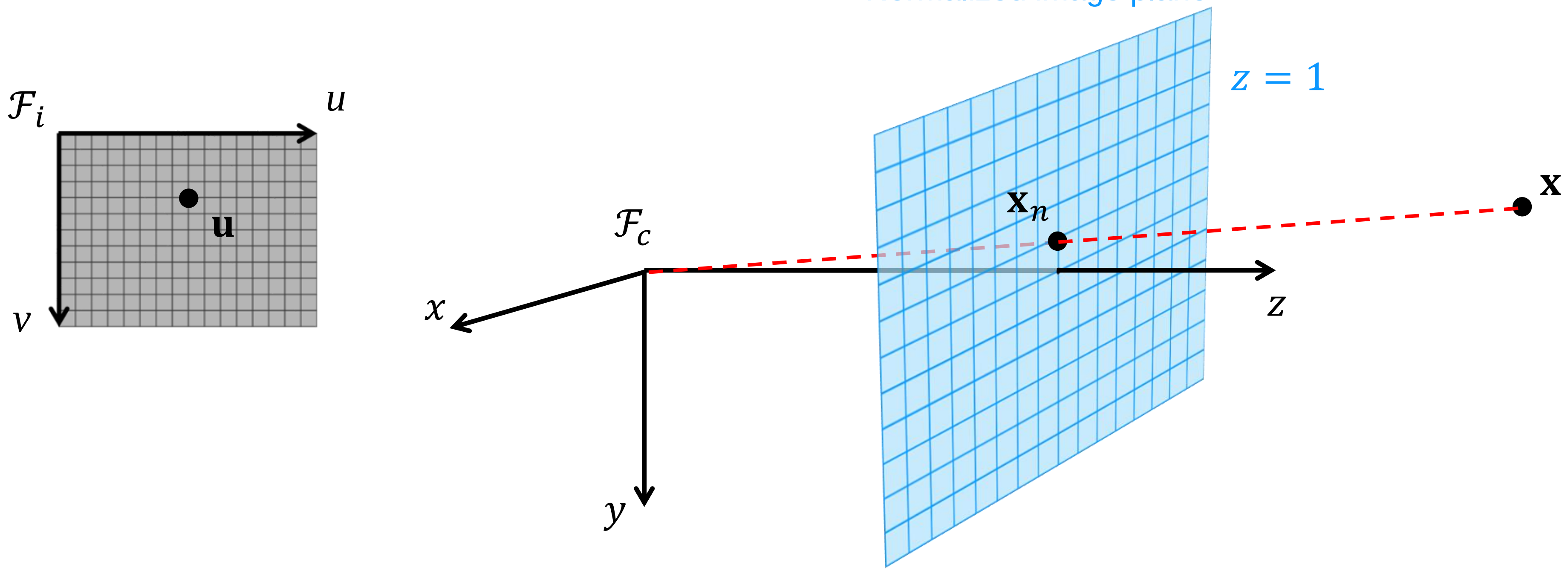
The perspective camera model



Points in the normalized image plane can be described both as 2D and 3D points

- 3D points $\mathbf{x}_n$ in $\mathcal{F}_c$
- 2D points $\mathbf{u}$ in $\mathcal{F}_i$

The perspective camera model



The perspective camera model is composed by two transformations:

- A perspective projection that maps $\mathbf{x}$ to $\mathbf{x}_n$
- A transformation of the normalized image plane, that maps $\mathbf{x}_n$ to $\mathbf{u}$

A more thorough introduction is given in another lecture

Projective geometry

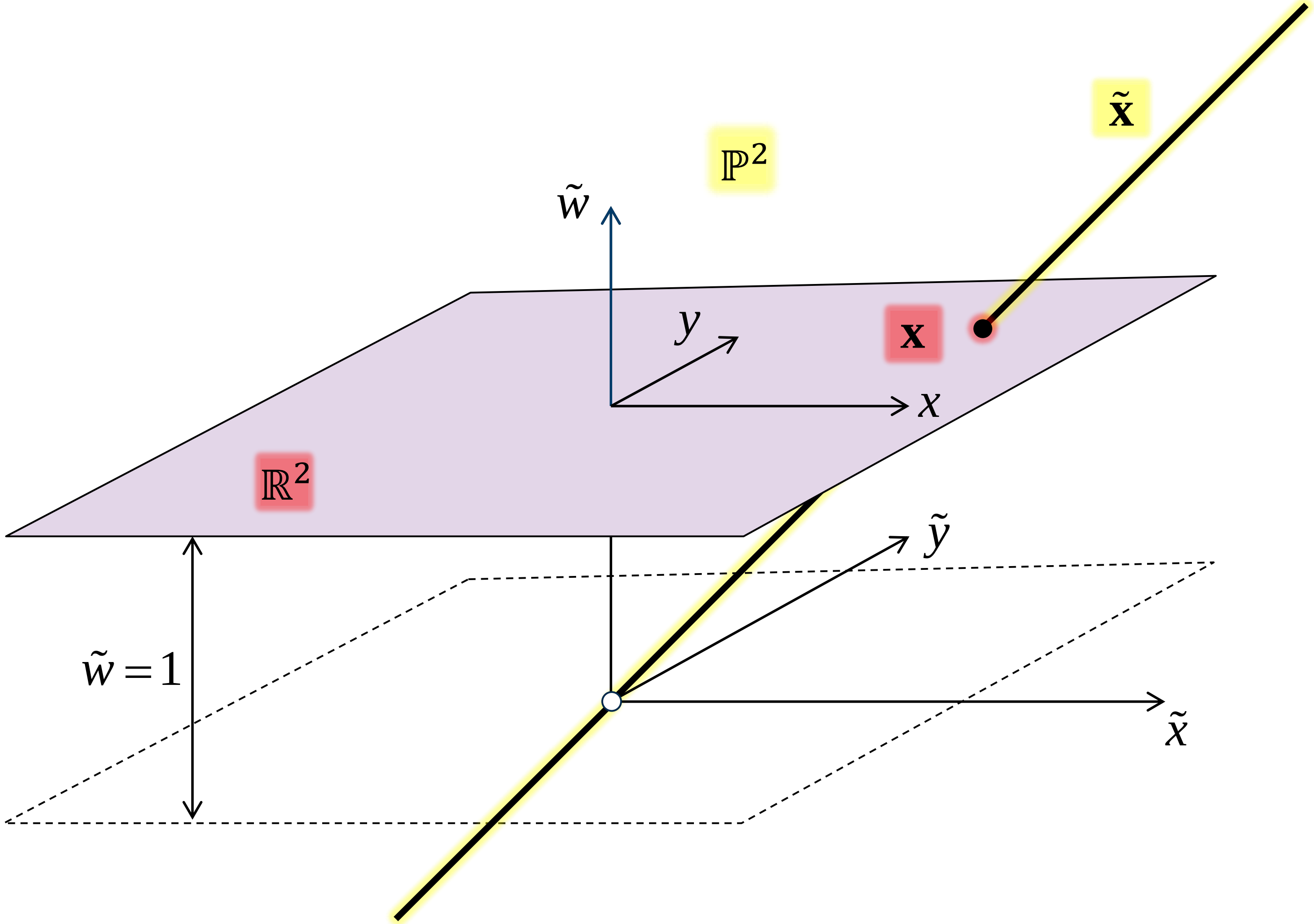
- Projective geometry is an alternative to Euclidean geometry
  - Points
  - Point transformations
  - +++

- The perspective camera model is most conveniently expressed using some of the basic notions from projective geometry

- In computer vision many results and expressions are easiest described in the projective framework

A more thorough introduction is given in another lecture

Projective geometry



Points in the plane

Euclidean geometry

- Unique representation
- Each point corresponds to a coordinate pair

$$\mathbf{x} = \begin{bmatrix} x \\ y \end{bmatrix} \in \mathbb{R}^2$$

Projective geometry

- Unique representation up to scale
- Each point corresponds to a triple of **homogeneous coordinates**

$$\tilde{\mathbf{x}} = \begin{bmatrix} \tilde{x} \\ \tilde{y} \\ \tilde{w} \end{bmatrix} \in \mathbb{P}^2$$

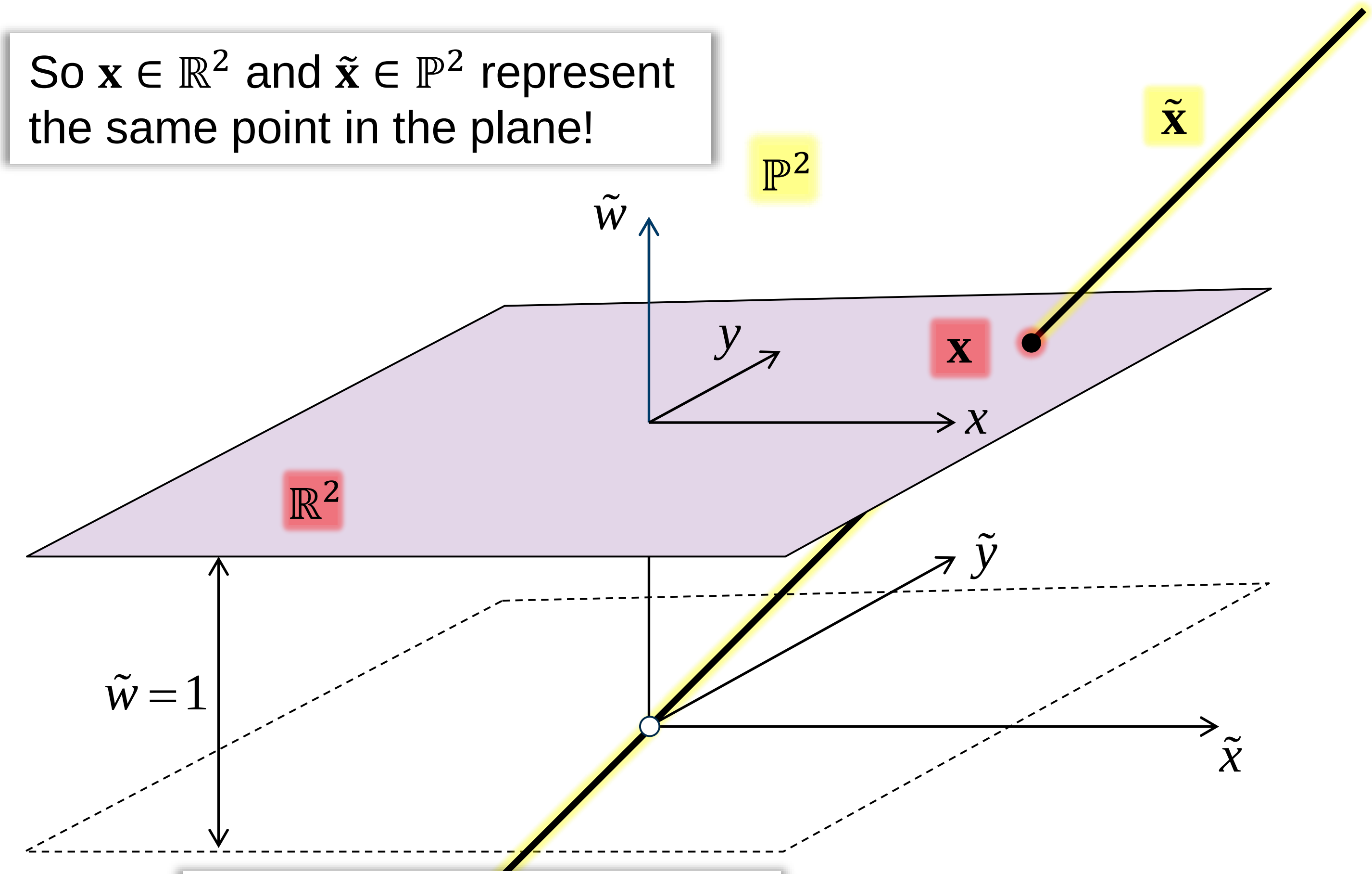
where

$$\tilde{\mathbf{x}} = \lambda\tilde{\mathbf{x}} \;\; \forall \; \lambda \in \mathbb{R}\backslash\{0\}$$

Projective geometry

A more thorough introduction is given in another lecture

So $\mathbf{x} \in \mathbb{R}^2$ and $\tilde{\mathbf{x}} \in \mathbb{P}^2$ represent the same point in the plane!



Euclidean ↔ Projective
$\tilde{w} = 1 \quad \Rightarrow \quad \tilde{x} = x, \tilde{y} = y$

Points in the plane

Euclidean geometry

- Unique representation
- Each point corresponds to a coordinate pair

$$\mathbf{x} = \begin{bmatrix} x \\ y \end{bmatrix} \in \mathbb{R}^2$$

Projective geometry

- Unique representation up to scale
- Each point corresponds to a triple of **homogeneous coordinates**

$$\tilde{\mathbf{x}} = \begin{bmatrix} \tilde{x} \\ \tilde{y} \\ \tilde{w} \end{bmatrix} \in \mathbb{P}^2$$

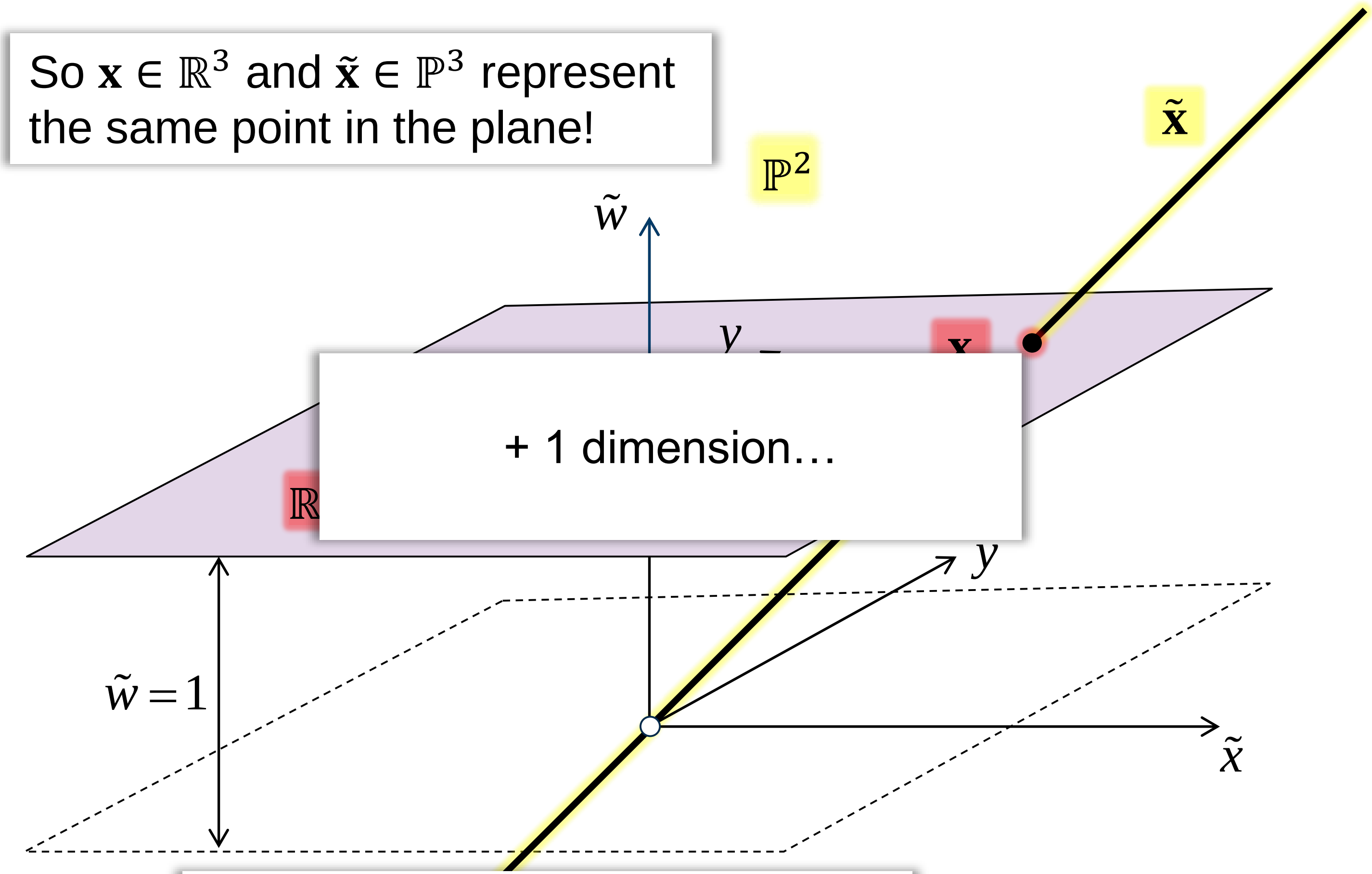
where

$$\tilde{\mathbf{x}} = \lambda \tilde{\mathbf{x}} \;\; \forall \, \lambda \in \mathbb{R} \setminus \{0\}$$

Projective geometry

A more thorough introduction is given in another lecture

So $\mathbf{x} \in \mathbb{R}^3$ and $\tilde{\mathbf{x}} \in \mathbb{P}^3$ represent the same point in the plane!



+ 1 dimension…

Euclidean ↔ Projective

$\tilde{w} = 1 \quad \Rightarrow \quad \tilde{x} = x, \tilde{y} = y, \tilde{z} = z$

Points in space

Euclidean geometry

- Unique representation

$$\mathbf{x} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \in \mathbb{R}^3$$

Projective geometry

- Unique representation up to scale

$$\tilde{\mathbf{x}} = \begin{bmatrix} \tilde{x} \\ \tilde{y} \\ \tilde{z} \\ \tilde{w} \end{bmatrix} \in \mathbb{P}^3$$

where

$$\tilde{\mathbf{x}} = \lambda \tilde{\mathbf{x}} \ \forall \ \lambda \in \mathbb{R} \backslash \{0\}$$

Projective geometry

A more thorough introduction is given in another lecture

Linear transformations

Euclidean geometry

- Linear transformations can be represented as a unique matrix

$$\begin{aligned} T\colon \quad \mathbb{R}^2 &\rightarrow \mathbb{R}^2 \\ \mathbf{x} &\mapsto \mathbf{y} = \mathbf{T}\mathbf{x} \end{aligned}$$

2x2 matrix

Projective geometry

- Linear transformations can be represented as a **homogeneous matrix** (unique up to scale)

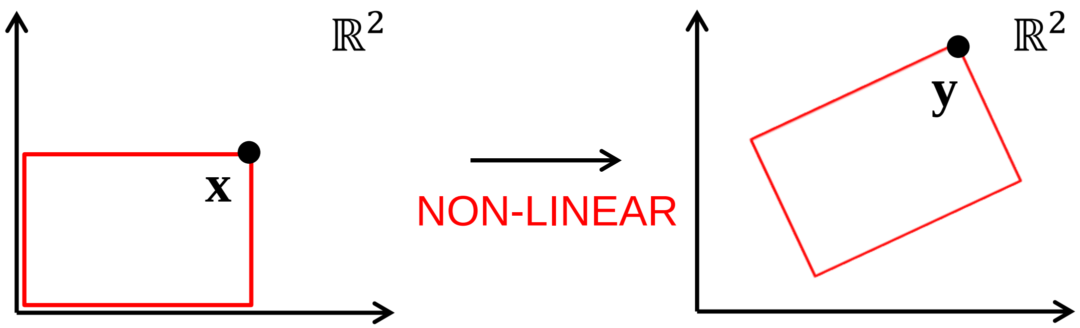
$$\begin{aligned} H\colon \quad \mathbb{P}^2 &\rightarrow \mathbb{P}^2 \\ \tilde{\mathbf{x}} &\mapsto \tilde{\mathbf{y}} = \mathbf{H}\tilde{\mathbf{x}} \end{aligned}$$

3x3 matrix

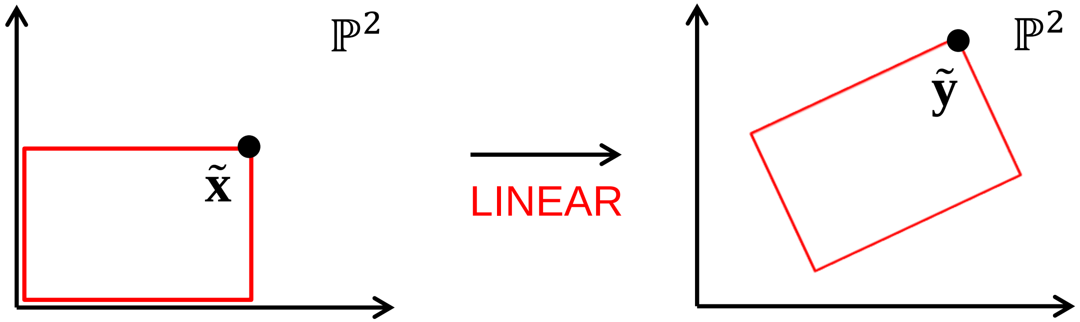
where

$$\mathbf{H} = \lambda\mathbf{H} \;\; \forall\, \lambda \in \mathbb{R}\backslash\{0\}$$

Projective geometry



Some transformations are linear in projective geometry and non-linear in Euclidean geometry



A more thorough introduction is given in another lecture

Linear transformations

Euclidean geometry

- Linear transformations can be represented as a unique matrix

$$\begin{aligned} T:\quad \mathbb{R}^2 &\rightarrow \mathbb{R}^2 \\ \mathbf{x} &\mapsto \mathbf{y} = \mathbf{T}\mathbf{x} \end{aligned}$$

2x2 matrix

Projective geometry

- Linear transformations can be represented as a **homogeneous matrix** (unique up to scale)

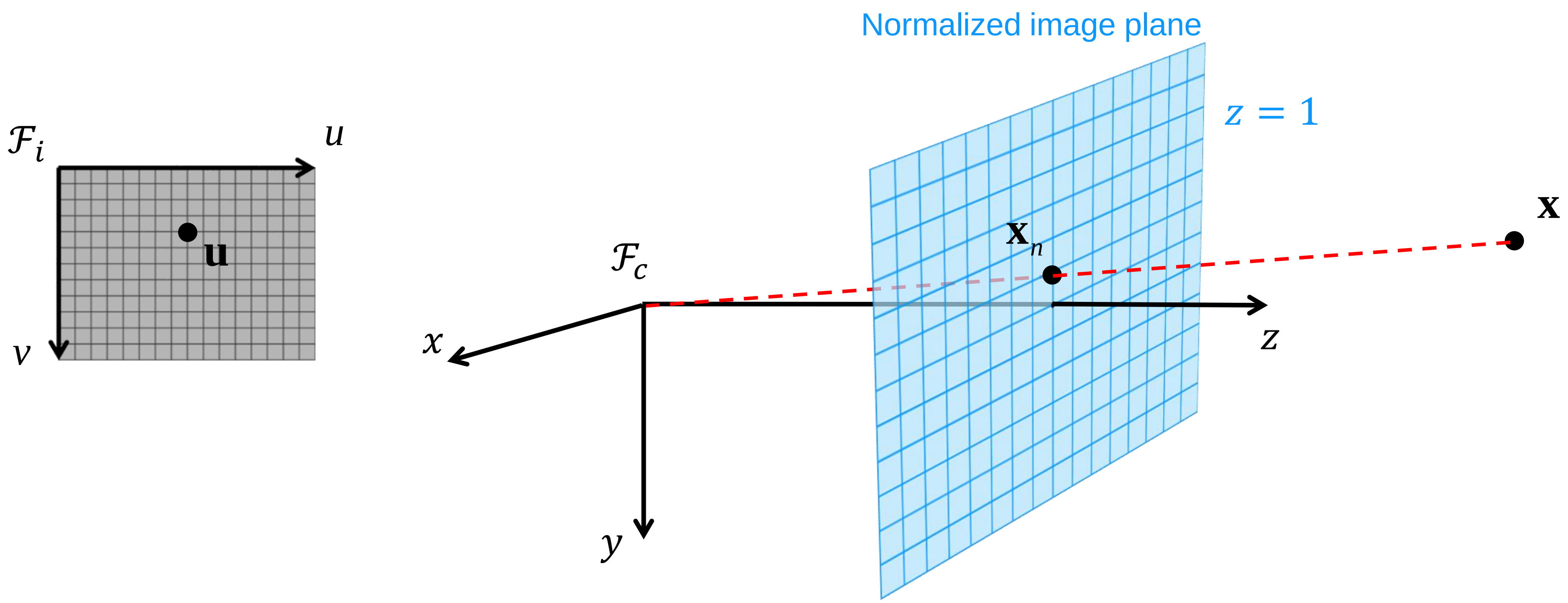
$$\begin{aligned} H:\quad \mathbb{P}^2 &\rightarrow \mathbb{P}^2 \\ \tilde{\mathbf{x}} &\mapsto \tilde{\mathbf{y}} = \mathbf{H}\tilde{\mathbf{x}} \end{aligned}$$

3x3 matrix

where

$$\mathbf{H} = \lambda\mathbf{H} \;\; \forall\, \lambda \in \mathbb{R}\backslash\{0\}$$

The perspective camera model

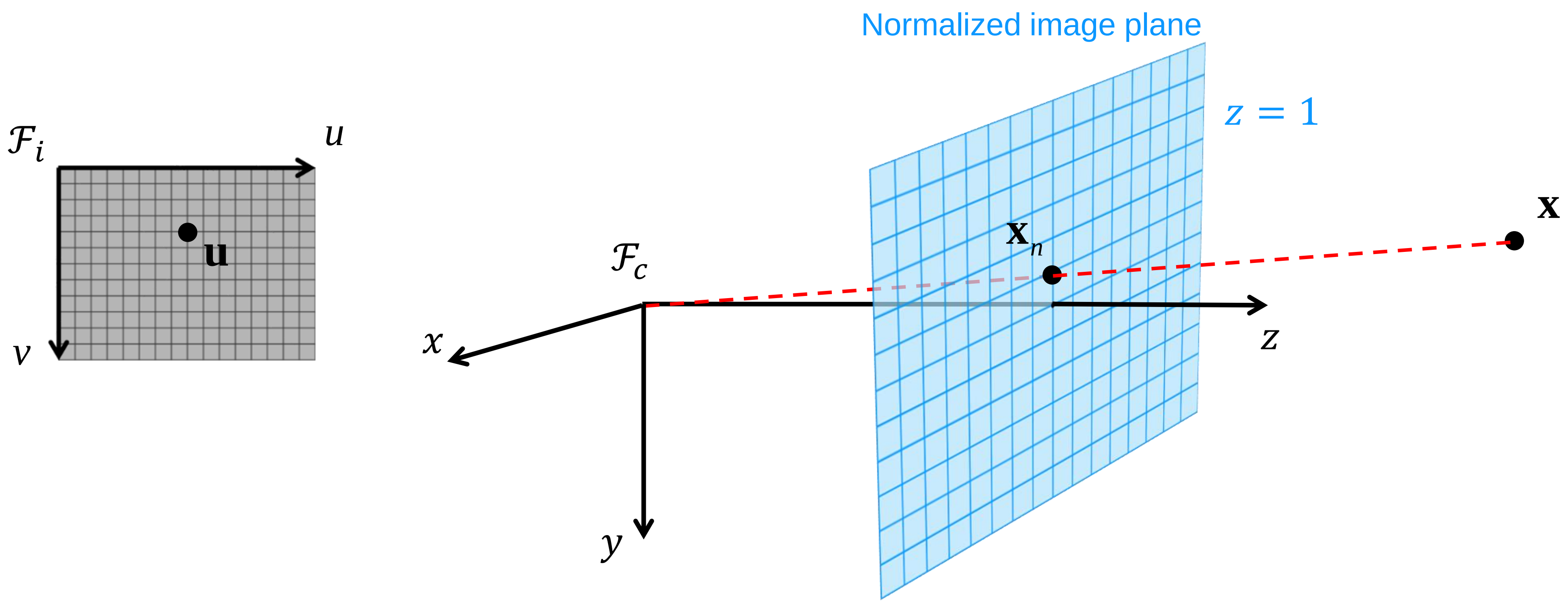


The perspective camera model is composed by two transformations:

- A perspective projection Π_0 that maps $\mathbf{x}$ to $\mathbf{x}_n$
- An affine transformation K that maps $\mathbf{x}_n$ to $\mathbf{u}$

$$\tilde{\mathbf{u}} = \underbrace{\begin{bmatrix} f_u & s & c_u \\ 0 & f_v & c_v \\ 0 & 0 & 1 \end{bmatrix}}_{\mathbf{K}} \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}}_{\mathbf{\Pi}_0} \tilde{\mathbf{x}}$$

The perspective camera model



The perspective camera model is composed by two transformations:

- A perspective projection Π_0 that maps $\mathbf{x}$ to $\mathbf{x}_n$
- An affine transformation K that maps $\mathbf{x}_n$ to $\mathbf{u}$

$$\tilde{\mathbf{u}} = \underbrace{\begin{bmatrix} f_u & s & c_u \\ 0 & f_v & c_v \\ 0 & 0 & 1 \end{bmatrix}}_{\mathbf{K}} \tilde{\mathbf{x}}_n$$

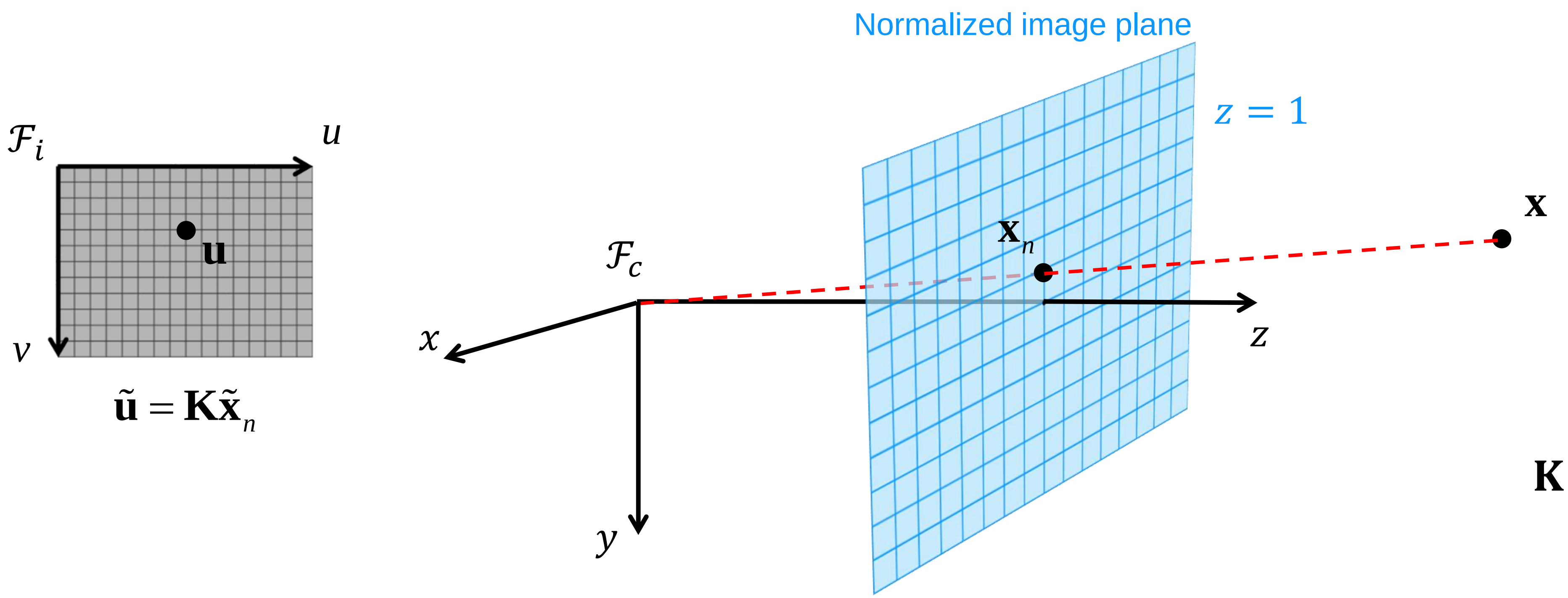
Remark on computations

Computing the image point $[u, v]^T$ for a world point $[x, y, z]^T$ is done in three steps

$$\mathbf{x} \quad \mapsto \quad \tilde{\mathbf{x}} \quad \overset{\mathbf{K\Pi}_0}{\mapsto} \quad \tilde{\mathbf{u}} \quad \mapsto \quad \mathbf{u}$$

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} \qquad \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \qquad \begin{bmatrix} \tilde{u} \\ \tilde{v} \\ \tilde{w} \end{bmatrix} \qquad \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} \frac{\tilde{u}}{\tilde{w}} \\ \frac{\tilde{v}}{\tilde{w}} \end{bmatrix}$$

The perspective camera model



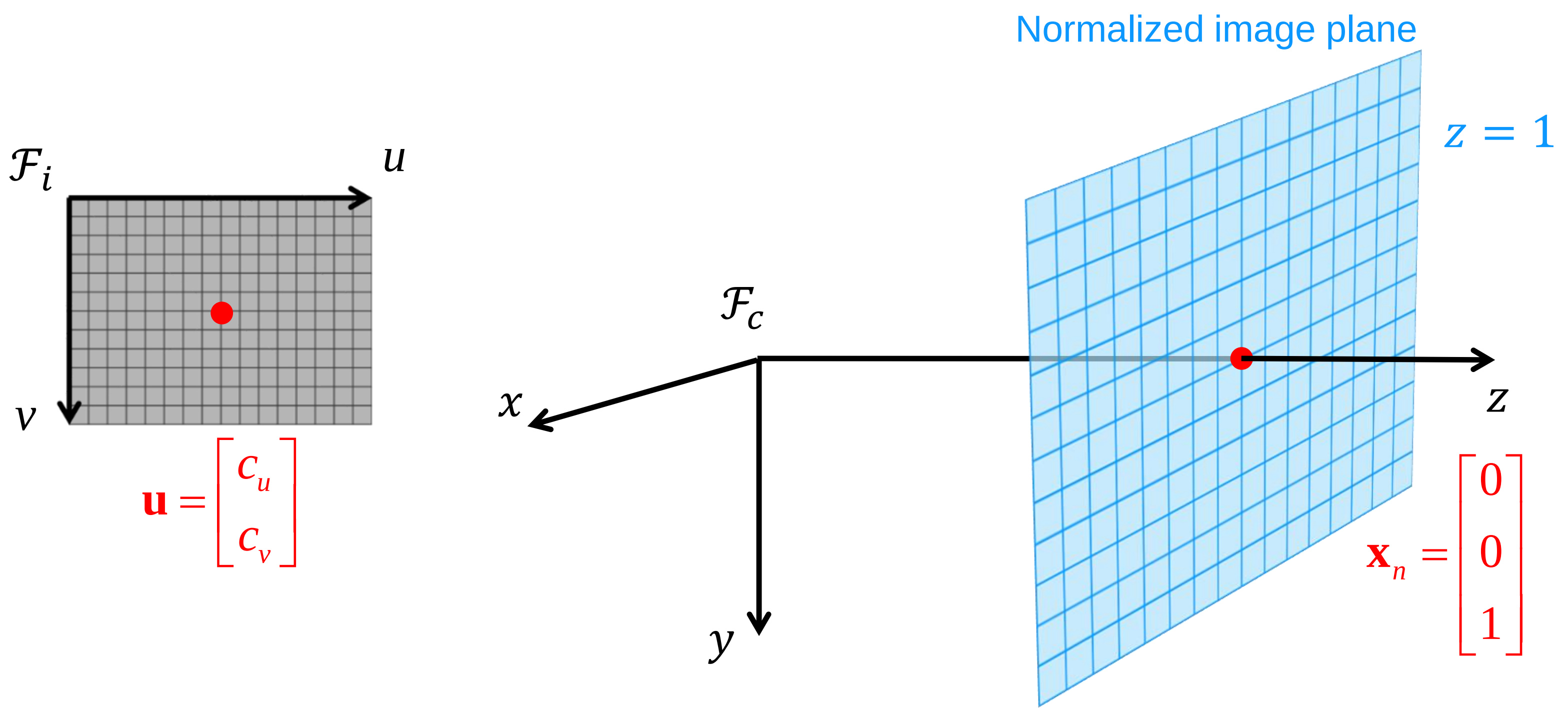
$$\tilde{\mathbf{u}} = \mathbf{K}\tilde{\mathbf{x}}_n$$

$$\mathbf{K} = \begin{bmatrix} f_u & s & c_u \\ 0 & f_v & c_v \\ 0 & 0 & 1 \end{bmatrix}$$

The affine transformation matrix $\mathbf{K}$ is the **intrinsic** part of the camera model, and it is often called the **camera calibration matrix**

The parameters are usually given in pixels

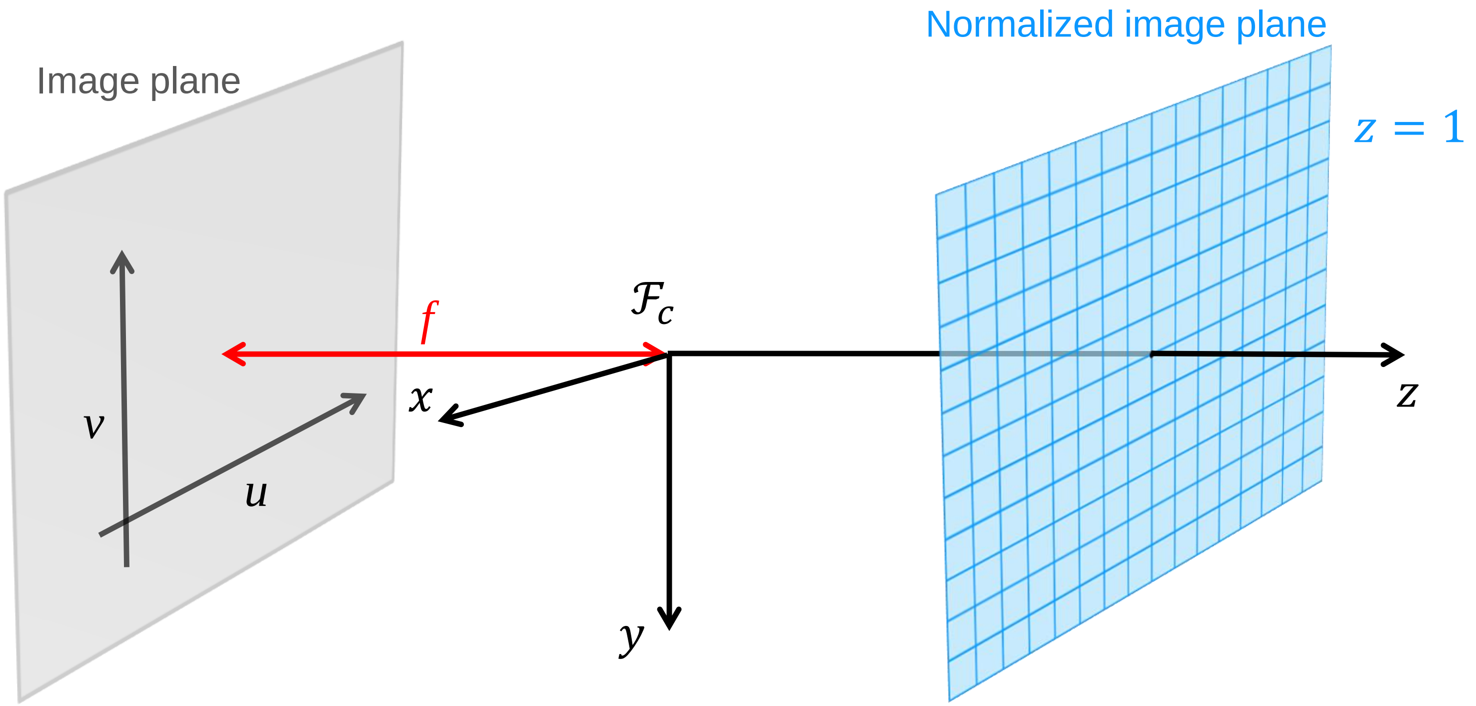
The camera calibration matrix

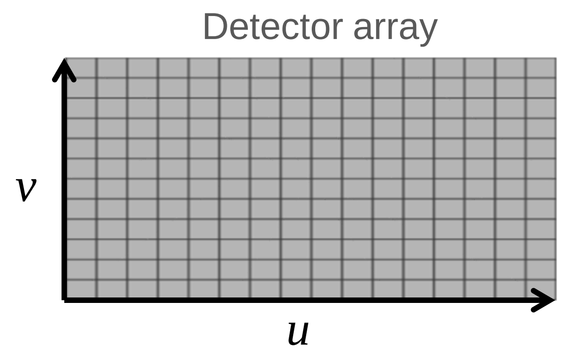


$$\mathbf{K} = \begin{bmatrix} f_u & s & c_u \\ 0 & f_v & c_v \\ 0 & 0 & 1 \end{bmatrix}$$

- The **optical center**, or **principal point**, (c_u, c_v) is where the optical axis intersects the image plane
- Often approximated by the center of the image, but the true value depends on how the detector array is aligned with the optical axis

The camera calibration matrix



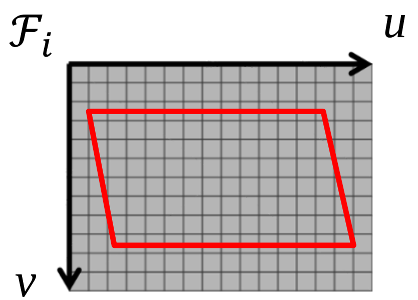


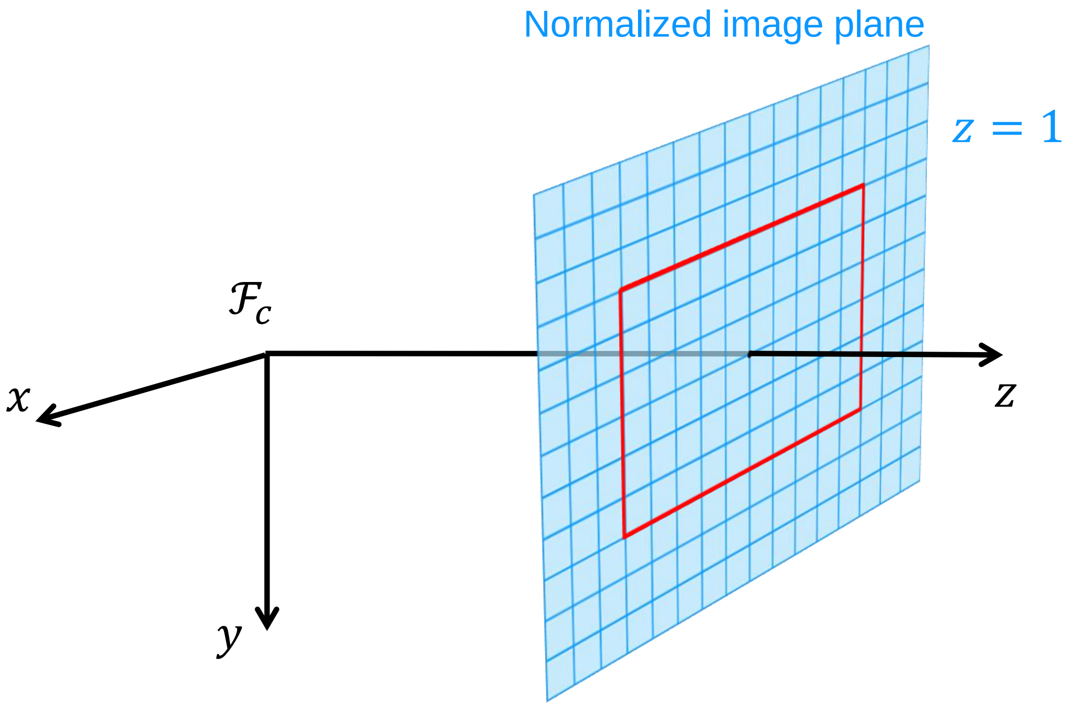
$$\mathbf{K} = \begin{bmatrix} f_u & s & c_u \\ 0 & f_v & c_v \\ 0 & 0 & 1 \end{bmatrix}$$

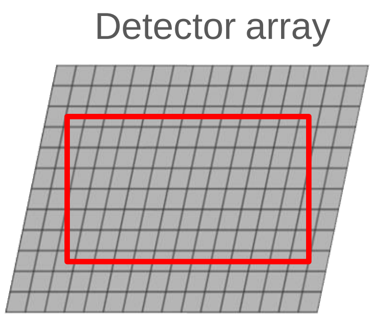
- The **focal length** f is the distance between the projective center and the image plane

- The parameters f_u and f_v are scaled versions of f reflecting that the density of detector elements can be different in the u- and v direction of the image plane

The camera calibration matrix







$$\mathbf{K} = \begin{bmatrix} f_u & s & c_u \\ 0 & f_v & c_v \\ 0 & 0 & 1 \end{bmatrix}$$

- The **skew** parameter s is required to describe cases when the detector array is not orthogonal to the optical axis

- For modern cameras this effect can typically be ignored, so it is common to set $s = 0$

The camera calibration matrix

Detector array

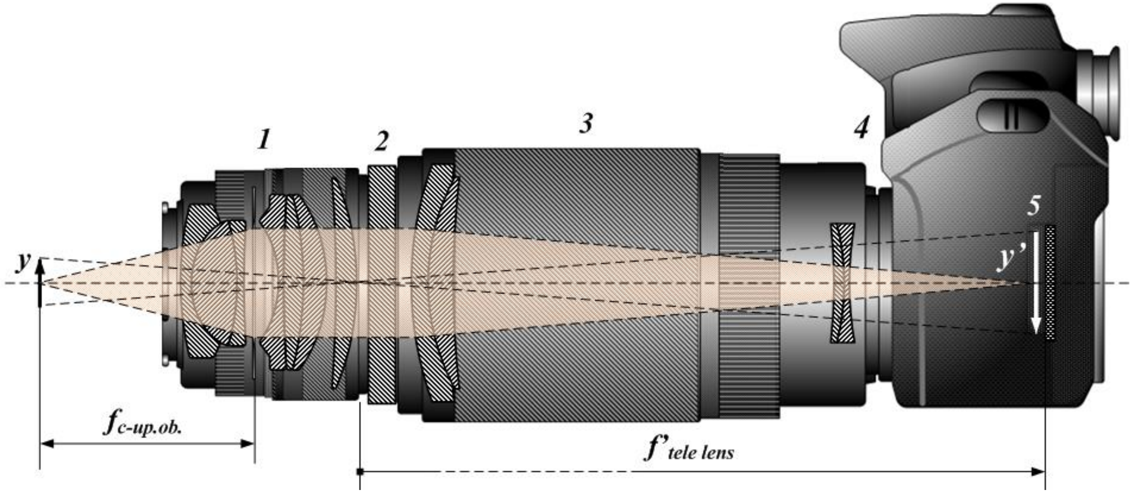
FOR THIS COURSE

$$\text{SKEW} = 0$$

$$\mathbf{K} = \begin{bmatrix} f_u & 0 & c_u \\ 0 & f_v & c_v \\ 0 & 0 & 1 \end{bmatrix}$$

FOR THIS COURSE

Non-ideal cameras



By Tamasflex [CC BY-SA 3.0 (https://creativecommons.org/licenses/by-sa/3.0) or GFDL (http://www.gnu.org/copyleft/fdl.html)], from Wikimedia Commons

- The perspective camera model describes a 3D to 2D transformation consistent with the pinhole geometry
  - Key characteristic: Preserves straight lines

- No cameras fit this model perfectly – All cameras suffer from some kind of distortion

- If we want to use images for geometrical computations we need to take this distortion into account

Non-ideal cameras

- Image from a camera with a large field of view
- Distorted – Lines are not preserved
- The perspective camera model does not apply!

Non-ideal cameras

- Image from a camera with a large field of view
- Distorted – Lines are not preserved
- The perspective camera model does not apply!

- Undistorted version of the same image
- **Undistortion** is an image transformation that removes distortion effects
- The perspective camera model applies!

Non-ideal cameras

http://www.robots.ox.ac.uk/~vgg/hzbook/

UNDISTORTED
FULL COVERAGE

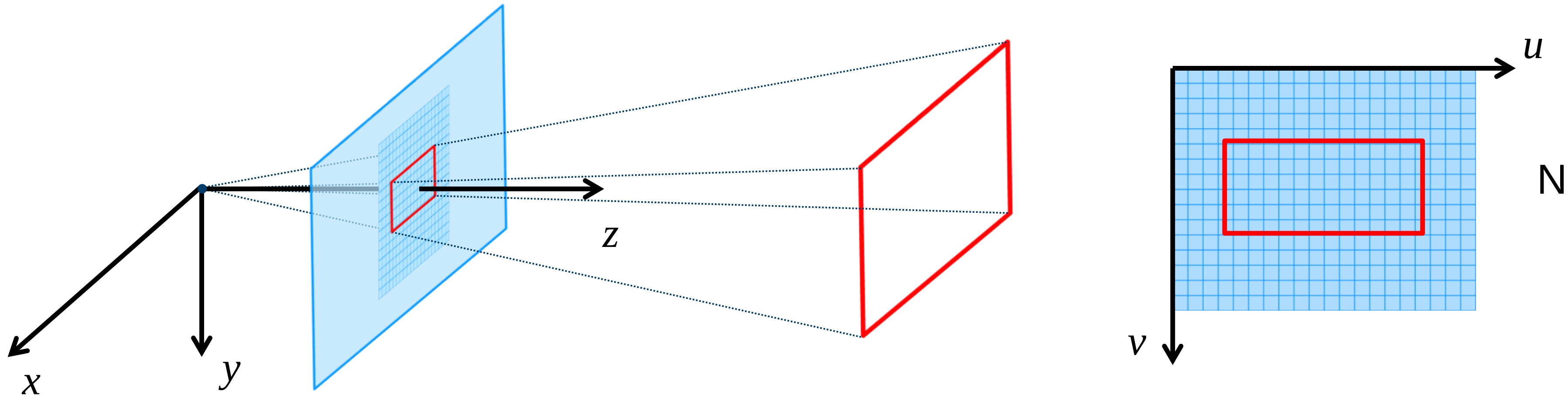
ORIGINAL

UNDISTORTED
LIMITED COVERAGE

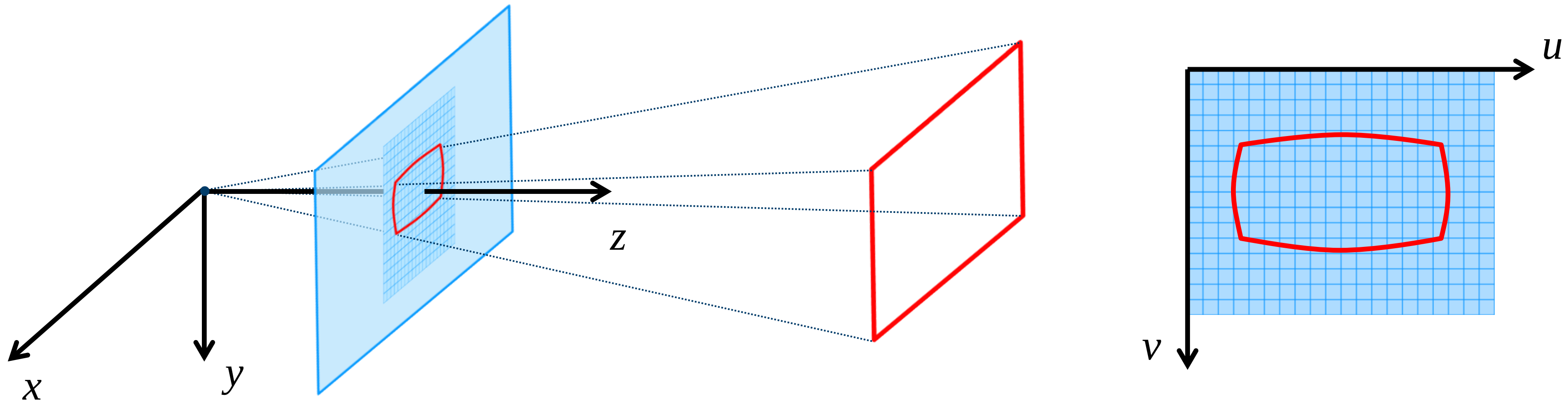
- The undistorted image has a different “footprint” than the original image
  - Images are rectangular → empty pixels

- It is common to restrict the visible part of the undistorted image to avoid empty pixels

Radial distortion

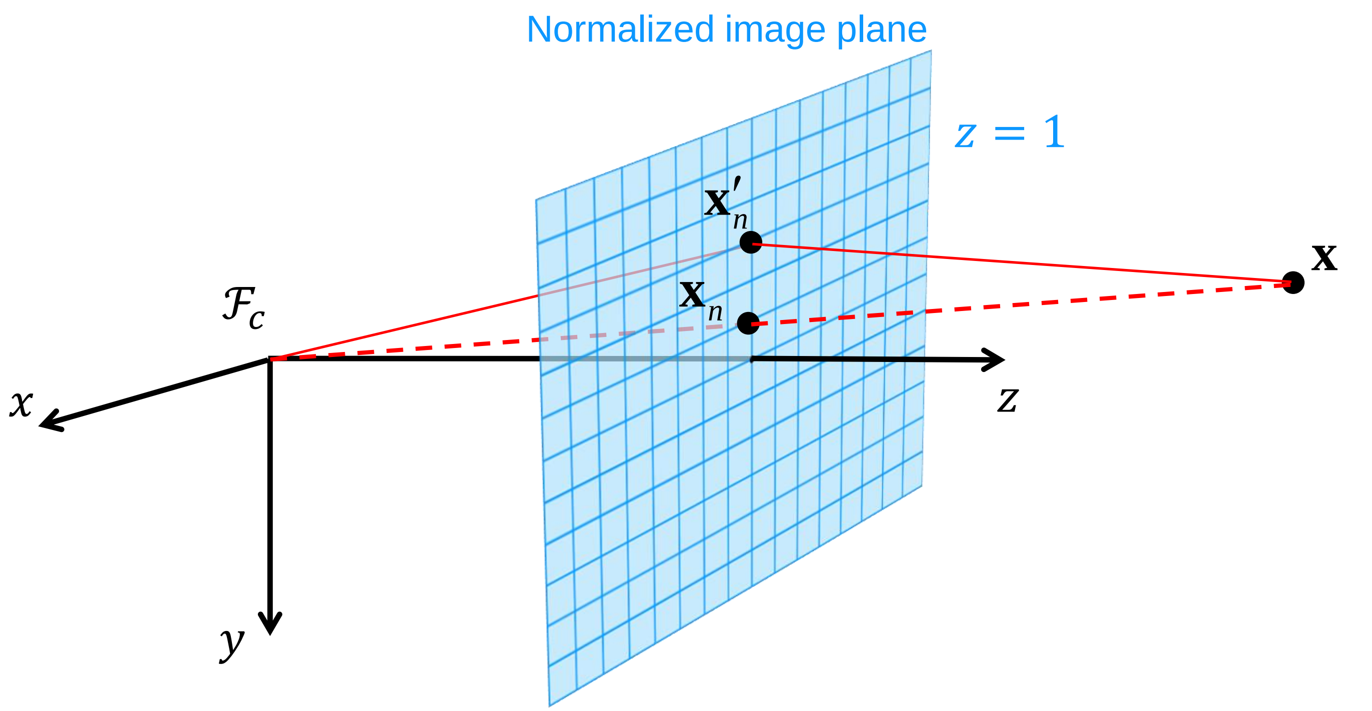


No radial distortion



Barrel distortion

Distortion model



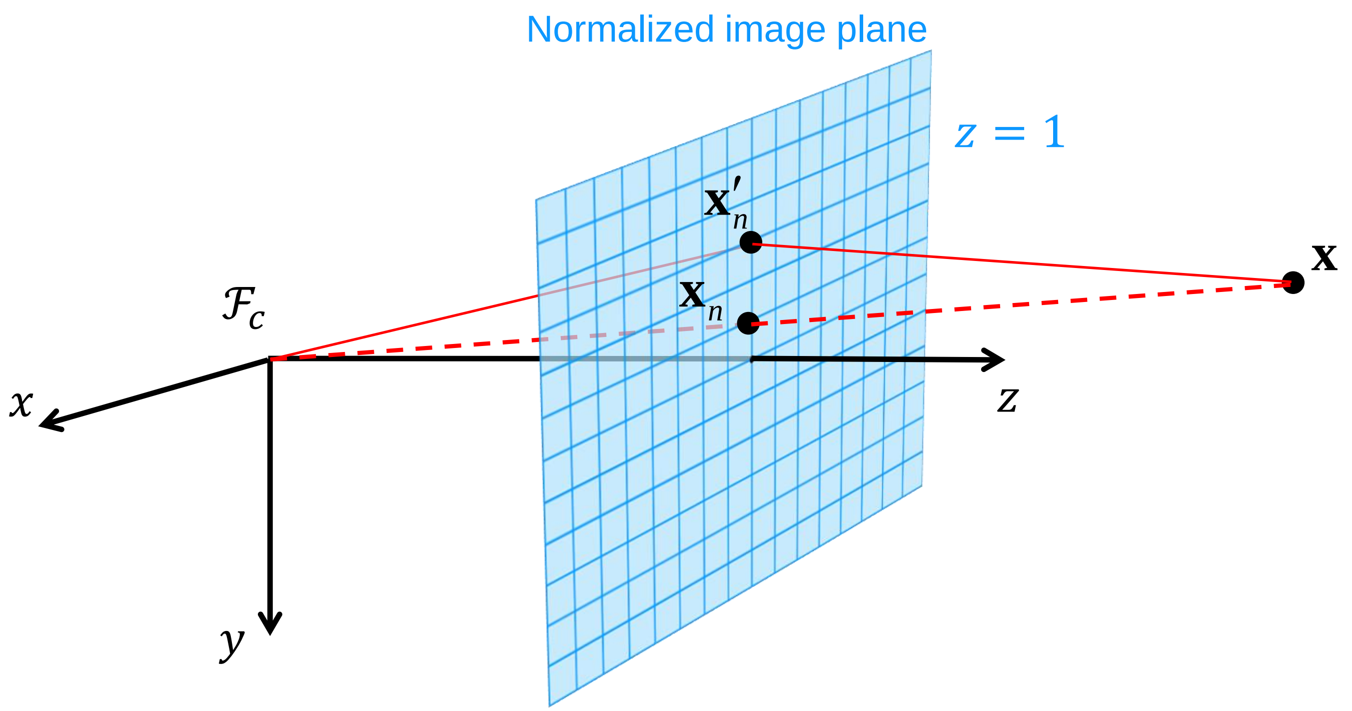
A **distortion model** describes how a camera deviates from the pinhole camera geometry

The deviation is most conveniently described in the normalized image plane as a relationship between the corrected (undistorted) points $\mathbf{x}_n$ and the true (distorted) points $\mathbf{x}'_n$

Distortion model

distorted: $\mathbf{x}'_n = \begin{bmatrix} x'_n \\ y'_n \\ 1 \end{bmatrix}$

undistorted: $\mathbf{x}_n = \begin{bmatrix} x_n \\ y_n \\ 1 \end{bmatrix}$

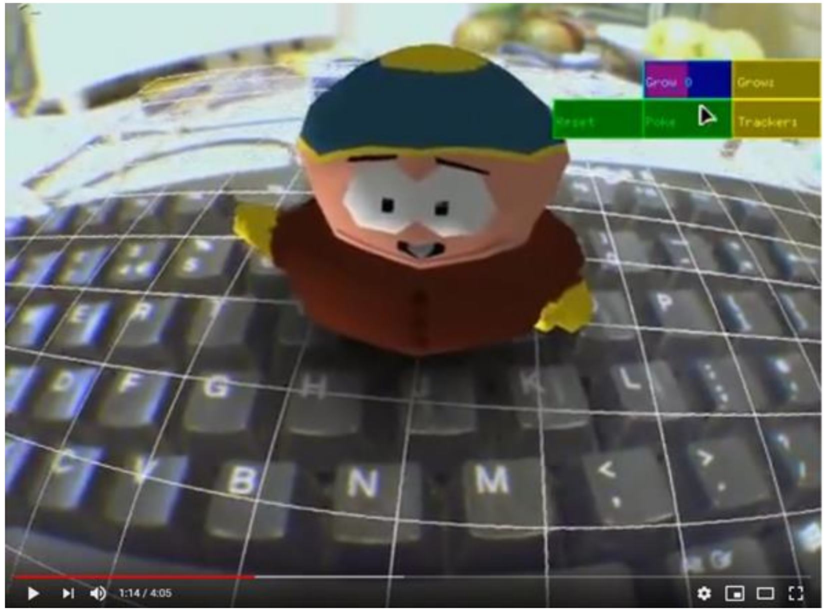


Example 2-parameter distortion model for radial distortion only

$$x'_n = x_n(1 + k_1 r_n^2 + k_2 r_n^4)$$
$$y'_n = y_n(1 + k_1 r_n^2 + k_2 r_n^4)$$

where $r_n^2 = x_n^2 + y_n^2$

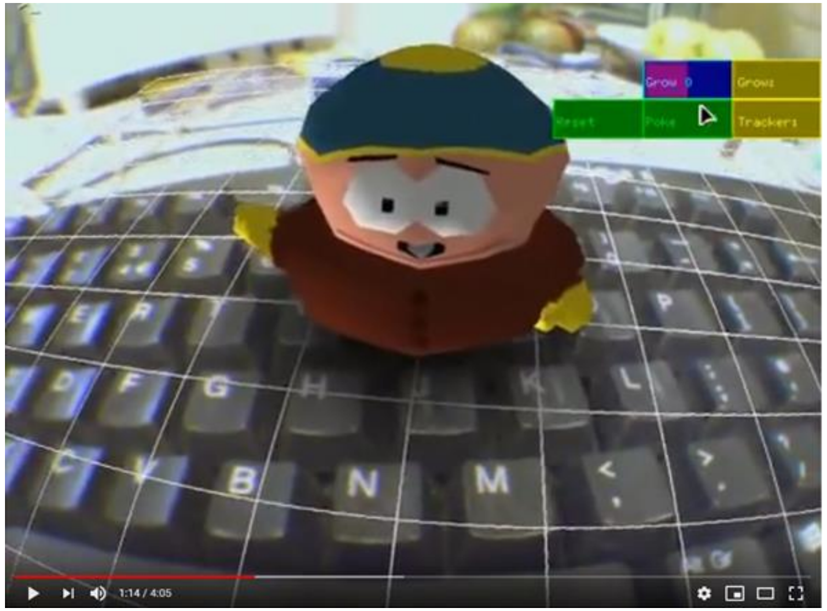
Working with images from non-ideal cameras



https://www.youtube.com/watch?v=F3s3M0mokNc

- Geometrical computations requires knowledge about the camera’s geometrical model
- For many cameras this can accurately be described by the perspective camera model combined with a distortion model

Working with images from non-ideal cameras



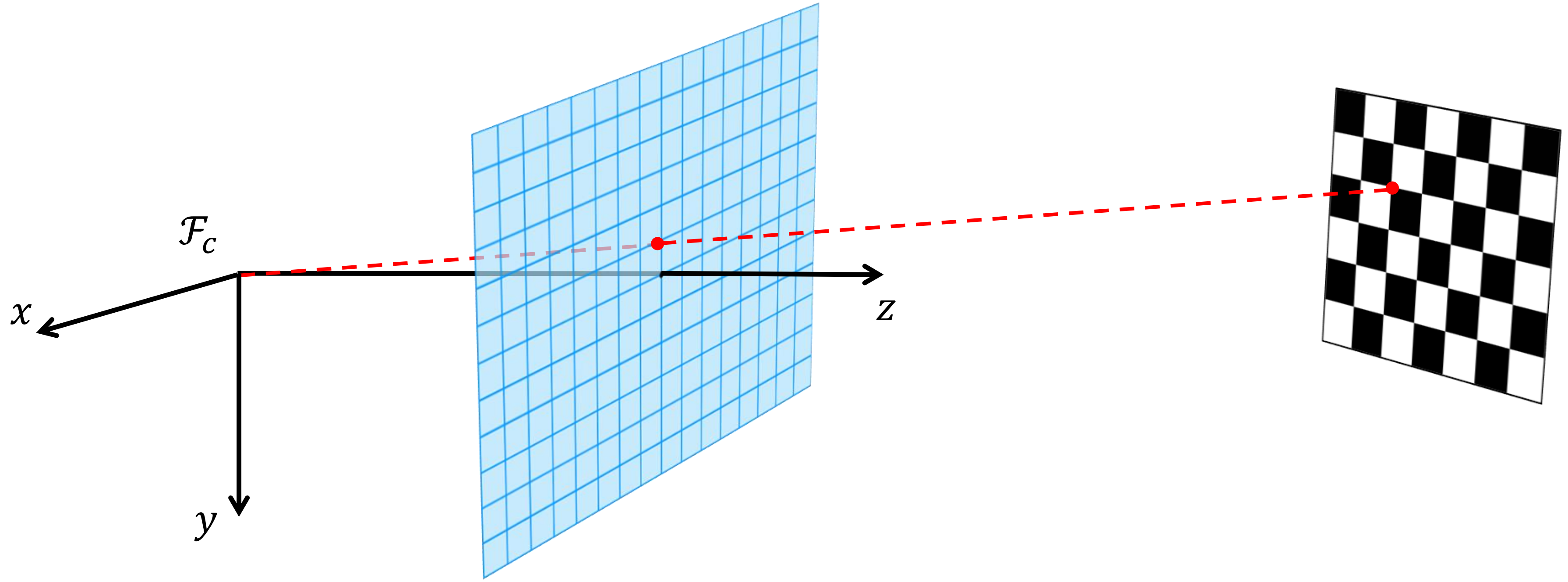
https://www.youtube.com/watch?v=F3s3M0mokNc

For geometrical computations, there are two common approaches

1. Work with undistorted images

2. Work with original images but undistort image points that are relevant for the computations

Camera calibration



- Estimates the intrinsic parameters f_u, f_v, s, c_u, c_v and the distortion parameters for a camera

- Calibration software
  - OpenCV
  - Kalibr (https://github.com/ethz-asl/kalibr)

Remark on computations with a distortion model

Computing the image point $[u, v]^T$ for a world point $[x, y, z]^T$ is done in five steps

$$\mathbf{x} \mapsto \tilde{\mathbf{x}} \overset{\mathbf{\Pi}_0}{\mapsto} \tilde{\mathbf{x}}_n \overset{\text{distort}}{\mapsto} \tilde{\mathbf{x}}'_n \overset{\mathbf{K}}{\mapsto} \tilde{\mathbf{u}} \mapsto \mathbf{u}$$

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} \quad \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \quad \begin{bmatrix} x_n \\ y_n \\ 1 \end{bmatrix} \quad \begin{bmatrix} x'_n \\ y'_n \\ 1 \end{bmatrix} \quad \begin{bmatrix} \tilde{u} \\ \tilde{v} \\ \tilde{w} \end{bmatrix} \quad \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} \frac{\tilde{u}}{\tilde{w}} \\ \frac{\tilde{v}}{\tilde{w}} \end{bmatrix}$$

Note that not all distortion models are easily invertible, so back projection of a pixel and undistortion of an image might be non-trivial

Geometric camera models

In general, we can represent a geometric camera model as a function

$$\pi\colon \mathbb{R}^3 \rightarrow \Omega$$

that projects 3D points $\mathbf{x}$ in the world to 2D points $\mathbf{u}$ in the image.

Here Ω denotes the image domain, so that

$$\mathbf{u} = \begin{bmatrix} u \\ v \end{bmatrix} \in \Omega \subset \mathbb{R}^2$$

Geometric camera models

The perspective camera model is one example – Here in Euclidean form (with zero skew)

$$\pi_p(\mathbf{x}) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \mathbf{K} \frac{1}{z} \mathbf{x} = \begin{bmatrix} f_u \dfrac{x}{z} + c_u \\ f_v \dfrac{y}{z} + c_v \end{bmatrix}$$

where $\mathbf{x} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$

Geometric camera models

The perspective camera model is one example – Here in Euclidean form (with zero skew)

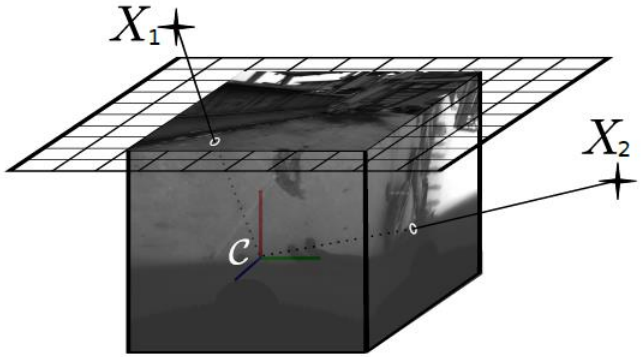
$$\pi_p(\mathbf{x}) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \mathbf{K} \frac{1}{z} \mathbf{x} = \begin{bmatrix} f_u \dfrac{x}{z} + c_u \\ f_v \dfrac{y}{z} + c_v \end{bmatrix} \qquad \text{where } \mathbf{x} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

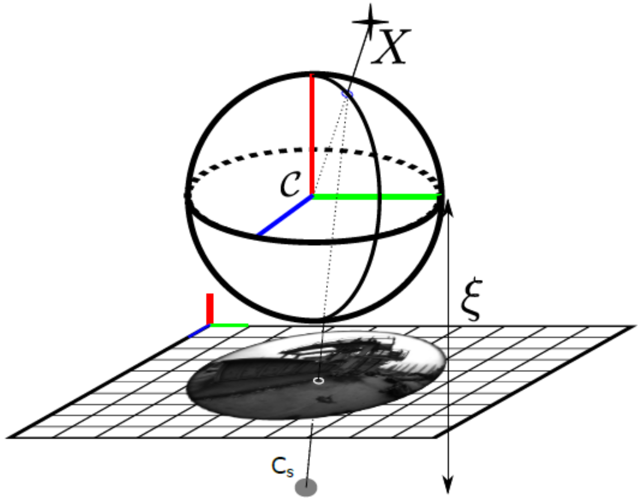
$$\Updownarrow$$

$$\tilde{\mathbf{u}} = \begin{bmatrix} f_u & 0 & c_u \\ 0 & f_v & c_v \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tilde{\mathbf{x}} \qquad \text{where } \tilde{\mathbf{x}} = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

Geometric camera models

But others exist





Array of n perspective cameras

$$\pi_{mp}(\mathbf{x}) = \left\{ \pi_{p_i}(\mathbf{R}_i \mathbf{x}) \right\}_{i=1 \ldots n}$$

Unified model

$$\pi_u(\mathbf{x}) = \begin{bmatrix} f_x \dfrac{x}{z + \|\mathbf{x}\| \xi} \\ f_y \dfrac{y}{z + \|\mathbf{x}\| \xi} \end{bmatrix} + \begin{bmatrix} c_x \\ c_y \end{bmatrix}$$

Caruso, D., Engel, J., & Cremers, D. (2015). Large-scale direct SLAM for omnidirectional cameras. In *IEEE International Conference on Intelligent Robots and Systems* (Vol. 2015–Decem, pp. 141–148). https://doi.org/10.1109/IROS.2015.7353366

Inverting the perspective camera model

Sometimes we want to backproject a 2D image point $\mathbf{u}$ to a 3D world point $\mathbf{x}$

$$\begin{bmatrix} f_u & 0 & c_u \\ 0 & f_v & c_v \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

Not invertible!

This is impossible unless we impose some restriction upon $\mathbf{x}$

One natural option is to backproject to a predefined depth z

Inverting the perspective camera model

The inverse model is then the backprojection

$$\pi_p^{-1}:\ \Omega \times \mathbb{R}^+ \rightarrow \mathbb{R}^3$$

which maps 2D image points back to 3D world points for a given depth z

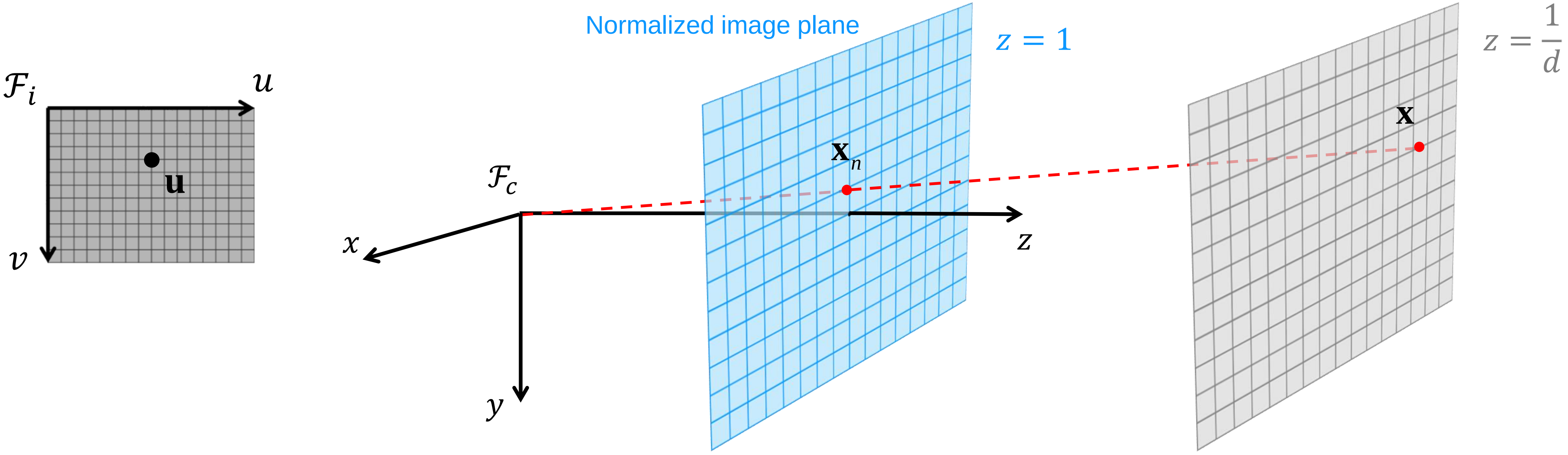
$$\pi_p^{-1}(\mathbf{u}, z) = z\mathbf{K}^{-1}\begin{bmatrix}\mathbf{u}\\1\end{bmatrix}$$

The depth is sometimes represented as **inverse depth** $d = z^{-1}$ since this parametrization is better suited when we want to model uncertainty

The backprojection model then becomes

$$\pi_p^{-1}(\mathbf{u}, d) = \frac{1}{d}\mathbf{K}^{-1}\begin{bmatrix}\mathbf{u}\\1\end{bmatrix}$$

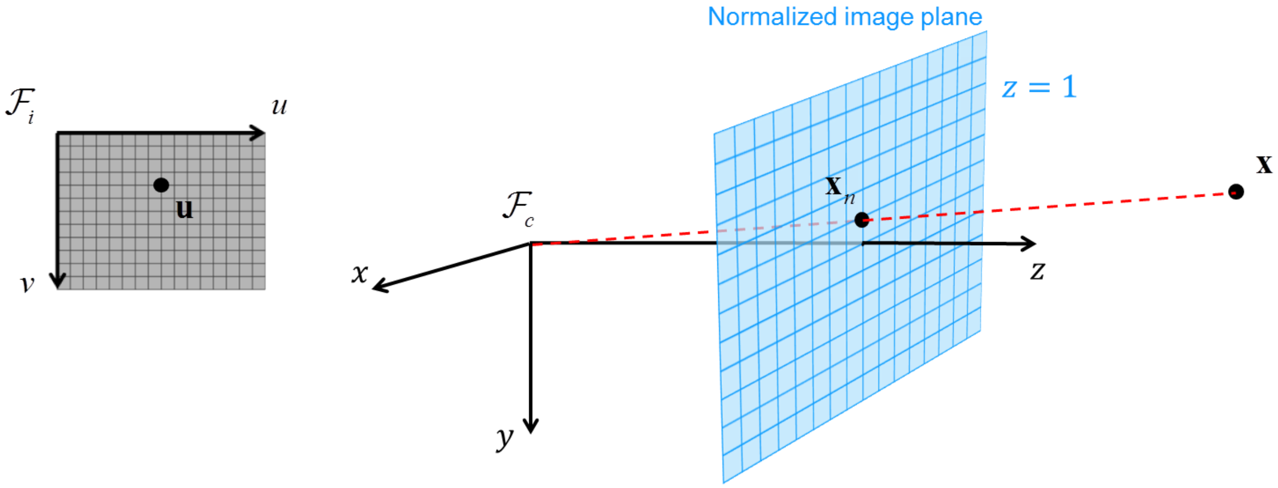
Inverting the perspective camera model



$$\mathbf{x} = z\mathbf{K}^{-1}\begin{bmatrix}\mathbf{u}\\1\end{bmatrix}$$

$$\mathbf{x} = \frac{1}{d}\mathbf{K}^{-1}\begin{bmatrix}\mathbf{u}\\1\end{bmatrix}$$

Summary



$$\tilde{\mathbf{u}} = \underbrace{\begin{bmatrix} f_u & 0 & c_u \\ 0 & f_v & c_v \\ 0 & 0 & 1 \end{bmatrix}}_{\mathbf{K}} \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}}_{\mathbf{\Pi}_0} \tilde{\mathbf{x}}$$

$$\Updownarrow$$

$$\pi_p(\mathbf{x}) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \mathbf{K} \frac{1}{z} \mathbf{x} = \begin{bmatrix} f_u \frac{x}{z} + c_u \\ f_v \frac{y}{z} + c_v \end{bmatrix}$$

The perspective camera model

- Pinhole geometry
- Preserves straight lines
- “Invertible”

$$\pi_p^{-1}(\mathbf{u}, d) = \frac{1}{d} \mathbf{K}^{-1} \begin{bmatrix} \mathbf{u} \\ 1 \end{bmatrix}$$

Non-ideal cameras

- Perspective camera model + distortion model
- Undistorted images are consistent with the perspective camera model

Supplementary material

Recommended

- *Richard Szeliski: Computer Vision: Algorithms and Applications 2nd ed*
  - Chapter 2 “Image formation”, in particular sections 2.1.4 “3D to 2D projections” and 2.1.5 “Lens distortions”