14/11/2023

connection between 3D space and 2D geometry of image plane

⇓

study scene-image relationship to understand usefull aspect for 3D shape reconstruction from 2D images

Camera Geometry and single-view Geometry

in principle single-view is NOT sufficent to reconstruct a genral 3D scene...
we can try to reconstruct part of 3D scene by single-view, using additional scene information!

divided in 2 parts:

1) single-view camera geometry
2) multi-view, geometry when at least 2 cameras observe scene (3D reconstruction)

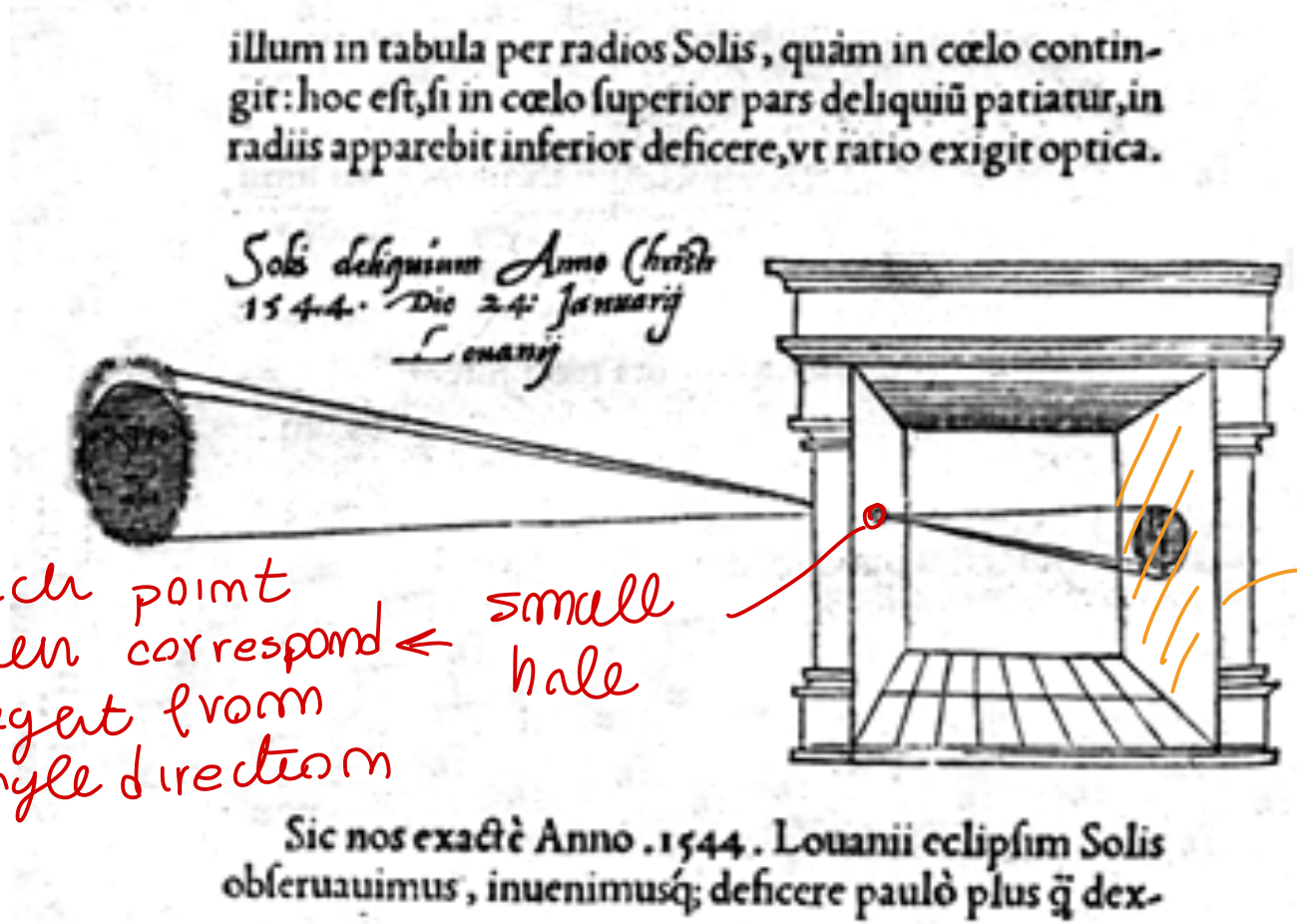
Camera Geometry

camera projective model

old camera

Pinhole camera

properties of single camera observing 3D scene



As we seen, the camera model is simplyfied as old camera pinhole model

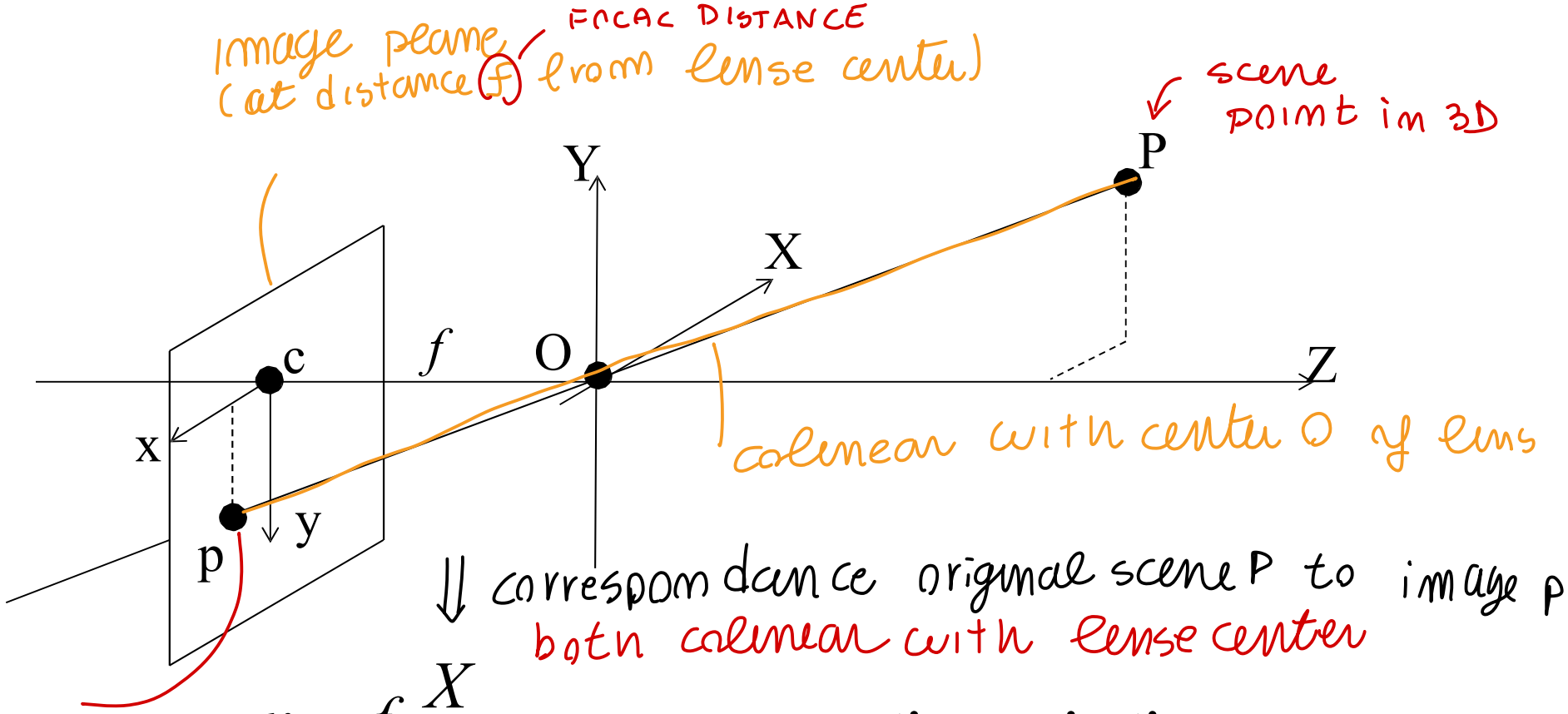
small hole

so each point on scren correspond to light from a single direction

sensitive element to light

same as lens system camera model

model: CAMERA



$\Downarrow$ correspondance orignal scene P to image p

both colinear with lense center

focalized rays emitted by point P

$$x = f\frac{X}{Z}$$

perspective projection

$$y = f\frac{Y}{Z}$$

-nonlinear
-not shape-preserving
-not length-ratio preserving

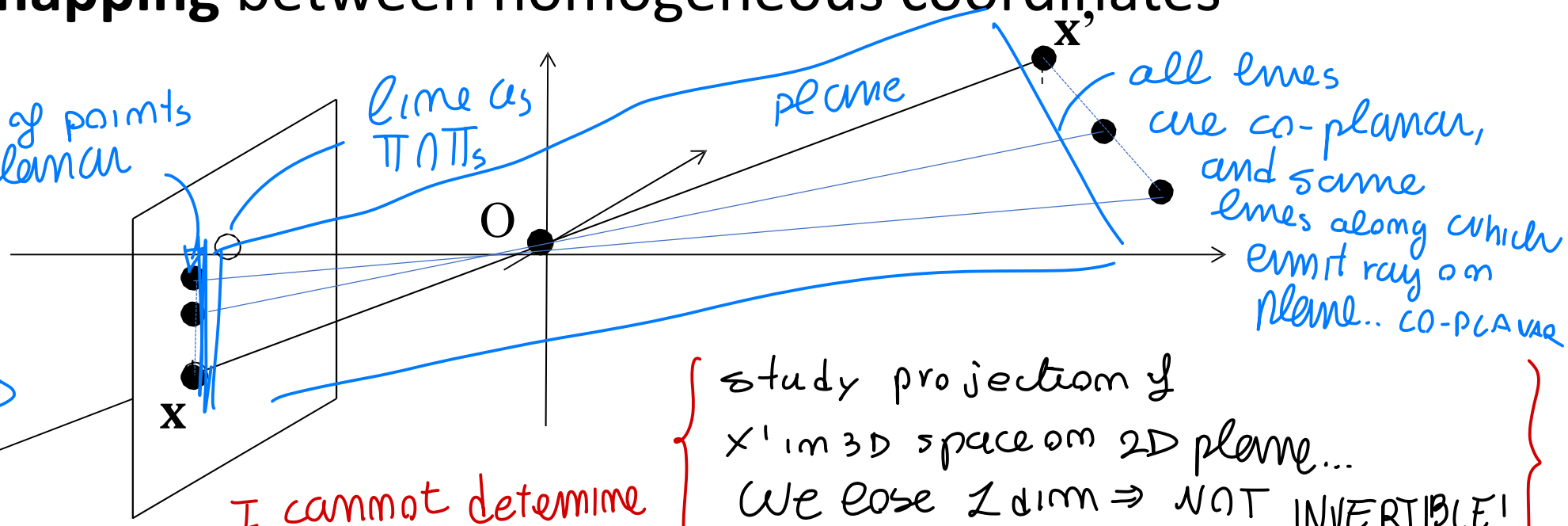
since, we saw projective transformation ≃ invertible mapping preserving co-linearity

Scene-to-image projection

Colinear points are projected onto colinear image points

→ colinearity is preserved

→ **linear mapping** between homogeneous coordinates

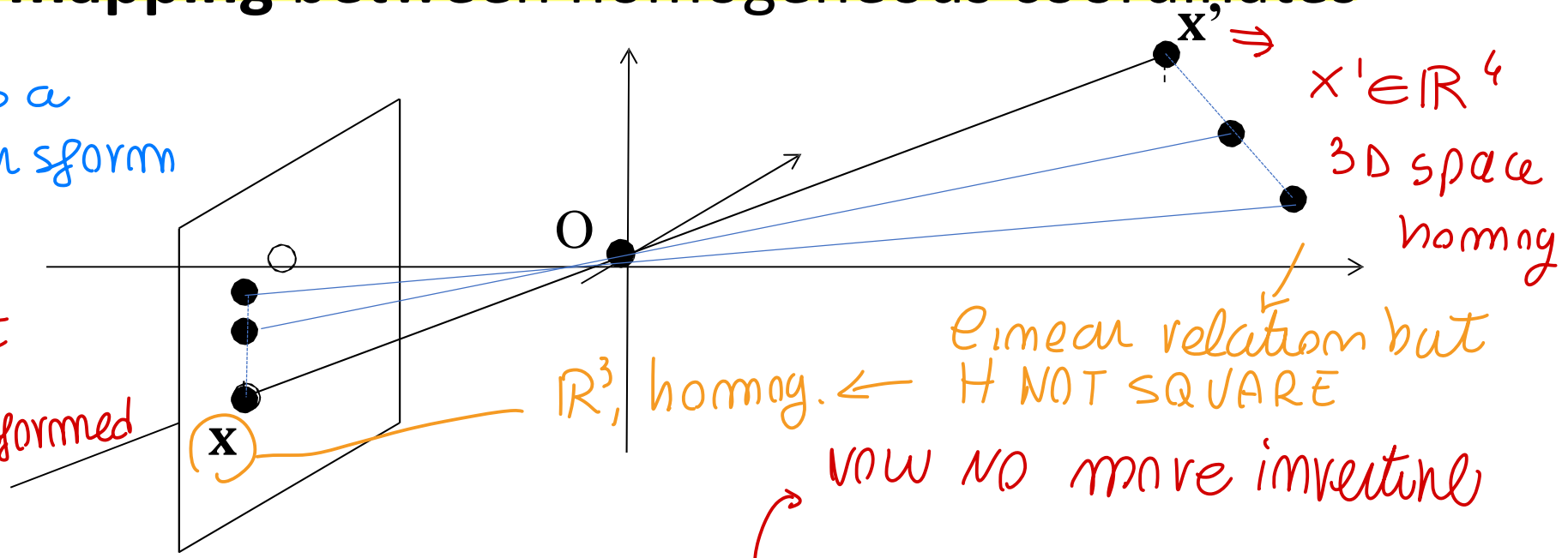


Scene-to-image projection

Colinear points are projected onto colinear image points

→ colinearity is preserved

→ **linear mapping** between homogeneous coordinates



x to x' is a linear transform

$x' = \begin{bmatrix} f & & \\ & f & \\ & & 1 \end{bmatrix} X$

+ point to be transformed

invertible?

until now

$x' = HX$ with H invertible?

$x' \Rightarrow$ $x' \in \mathbb{R}^4$ 3D space homog

linear relation but H NOT SQUARE

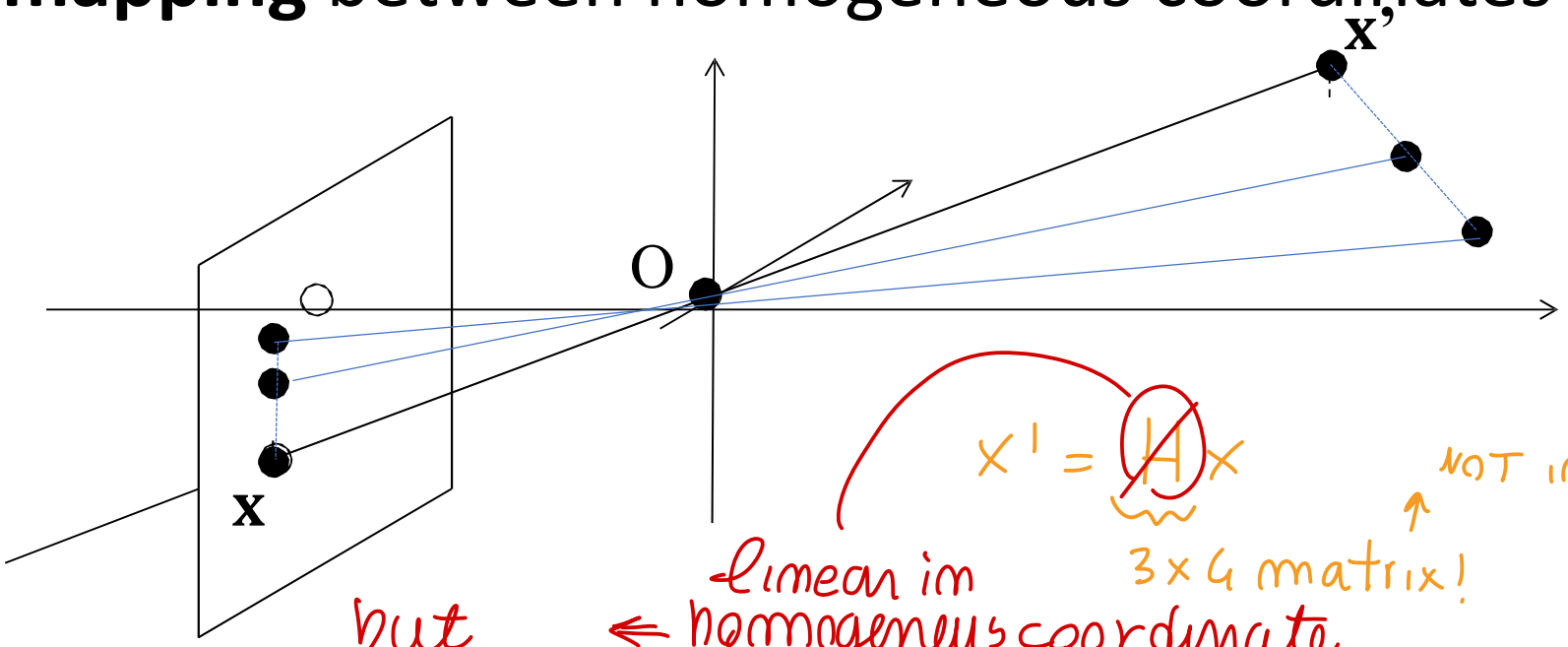
$\mathbb{R}^3$, homog. ← now no more invertible

Scene-to-image projection

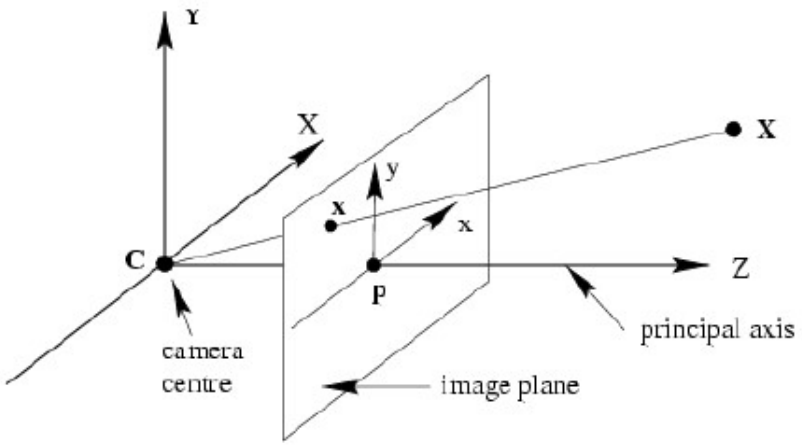
Colinear points are projected onto colinear image points

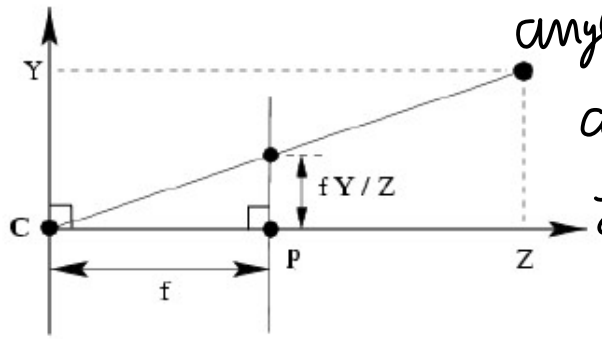
→ colinearity is preserved

→ **linear mapping** between homogeneous coordinates



CAMERA GEOMETRY





In a physical real camera with hyp of thin lens, small angles → $M_{3\times3}$ is a NON singular full rank matrix.. ∃ M^{-1} (invertible!)

colinearity is preserved → linear relation among homogeneous coords

Point on 3D scene to image relation ⇒

$$\mathbf{X} = \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \mapsto \mathbf{u} = \begin{pmatrix} u \\ v \\ w \end{pmatrix} = \mathbf{P}_{3x4} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} = \mathbf{P}_{3x4}\mathbf{X} = \left| \mathbf{M}_{3x3} \quad \mathbf{m}_{3x1} \right| \mathbf{X}$$

camera projection matrix

decompose matrix!

3D space

image

invertible sub-matrix

(column vector, NO assumption on this

- SCENE
- CAMERA ← internal part of camera

let's explore some aspects...

we can prove Null(P) is camera center O

what can we do anytime we have a non invertible 3x4 matrix

SCENE: viewing ray from image point

we can see what is Null(P)... it is JUST a 3D point

null-space of camera projection matrix **P** $\Rightarrow$ $\mathbf{PO} = 0$

(it can be proven...

a point **Y** on the line **X, O** — Y coeimcants x, O

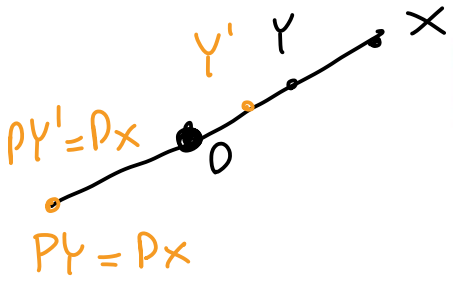
$$\mathbf{Y} = \alpha\mathbf{X} + \beta\mathbf{O}$$

in algebra ∞ solutions, but $\mathbb{P}^1$sal, 1D vector space = all multiples of common vector = one single point homogeneus

If we comput img projection

its image $\mathbf{u} = \mathbf{PY} = \alpha\mathbf{PX} + \beta\mathbf{PO} = \mathbf{PX}$

$\alpha X = X$ homogeneus



all slove same image!

$\Downarrow$

all points Y on (X,O) project on image of X,

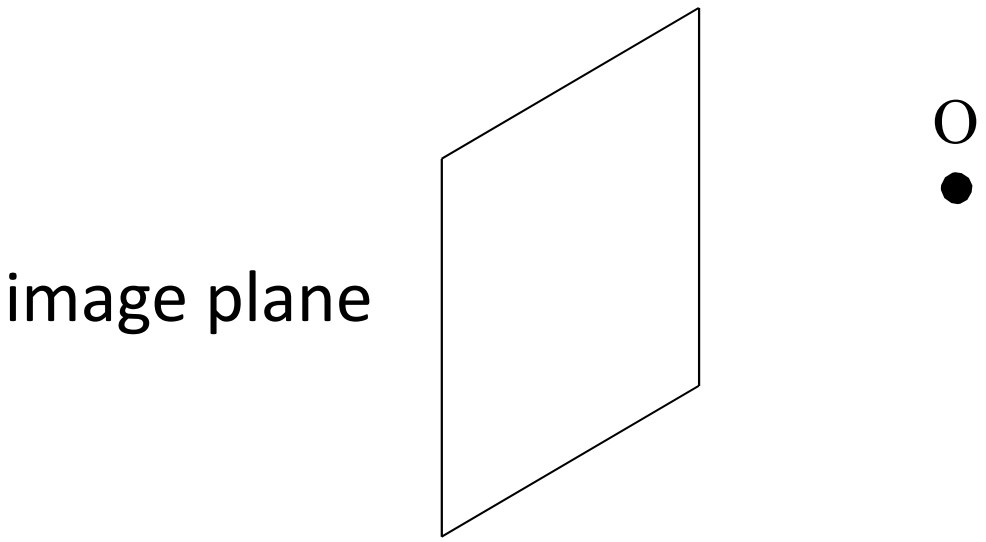
→ **O** is camera center

Image of camera center is $(0,0,0)^T$, i.e. undefined

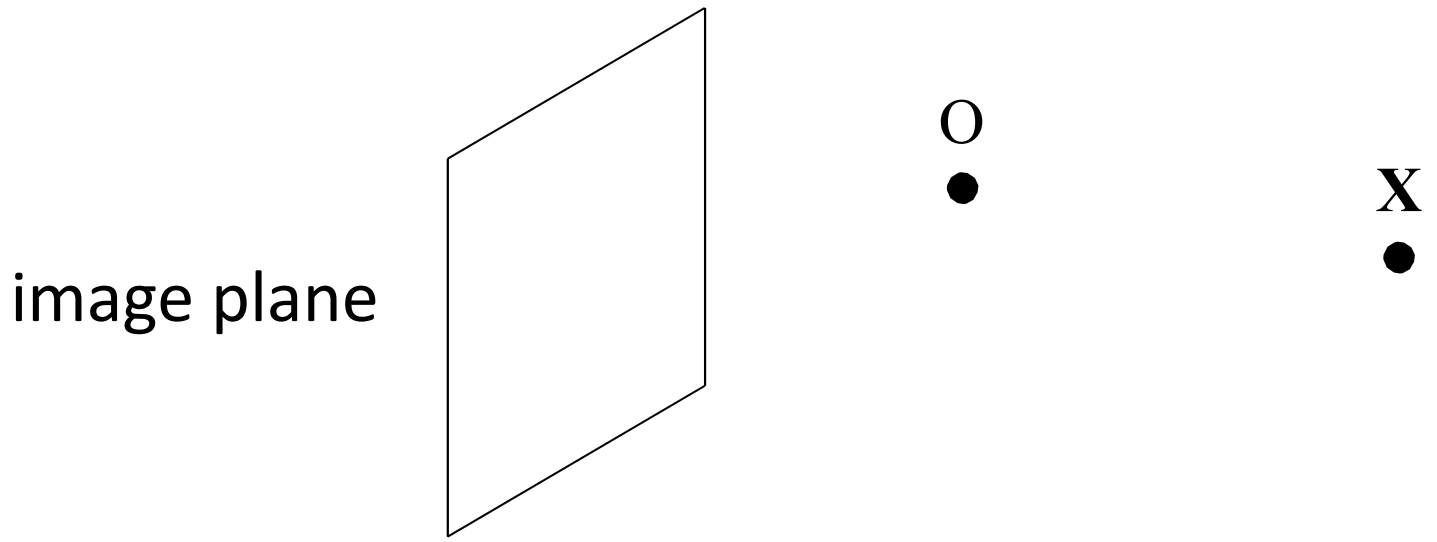
Finite cameras: $\mathbf{O} = \begin{pmatrix} \mathbf{o} \\ 1 \end{pmatrix} = \begin{pmatrix} -\mathrm{M}^{-1}\mathbf{m} \\ 1 \end{pmatrix}$

in fact ...

$$O = RNS(P): \; PO = 0$$

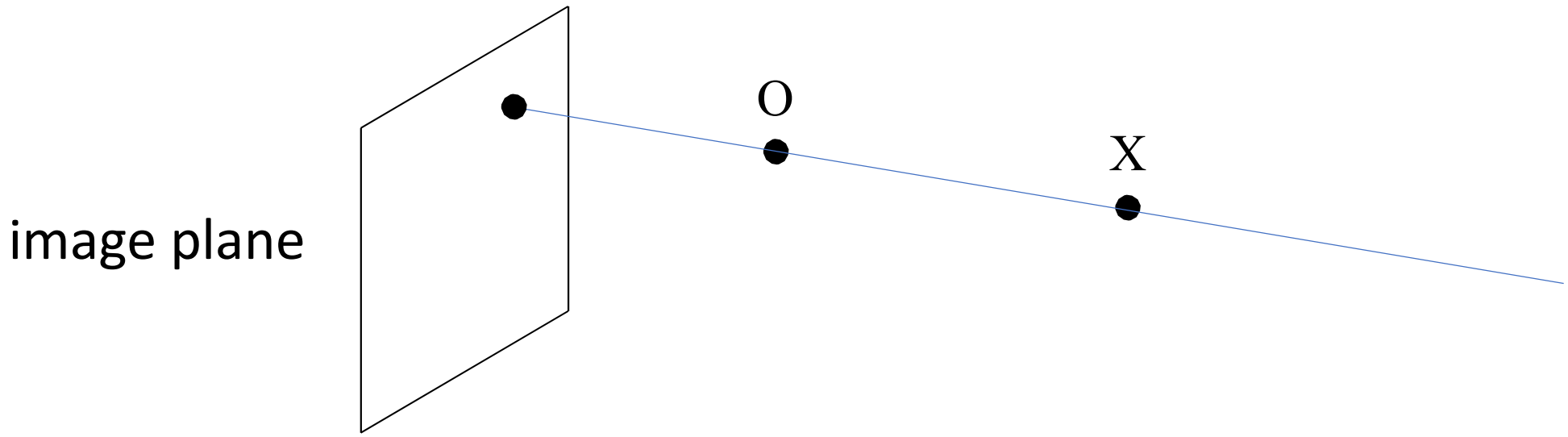


$$\mathrm{O} = RNS(P)\colon\ P\mathrm{O} = 0$$



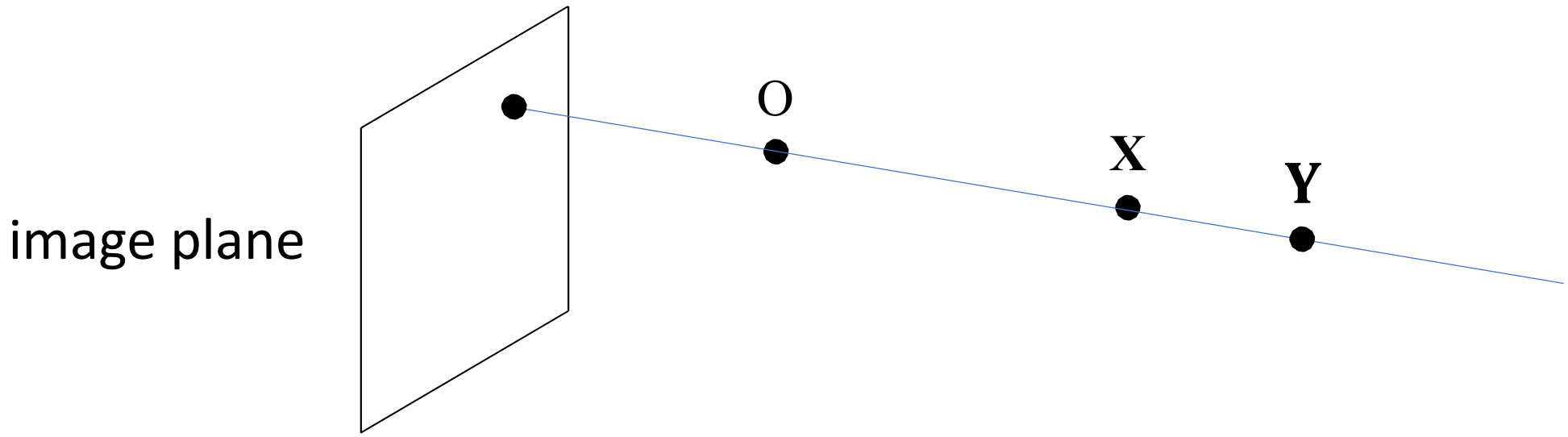
A point $\mathbf{X}$

$$\mathrm{O} = RNS(P): \; P\mathrm{O} = 0$$



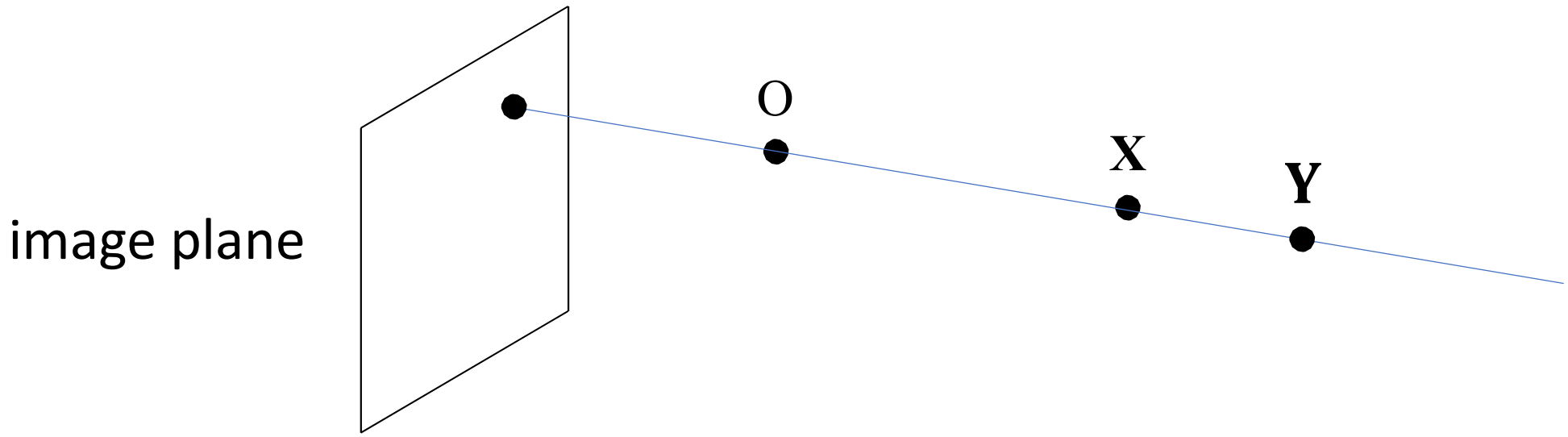
A point $\mathbf{X}$ and its image

$$\mathrm{O} = RNS(P): \; PO = 0$$



A point $\mathbf{X}$ and its image;
any point $\mathbf{Y}$ on line($\mathbf{O}, \mathbf{X}$):

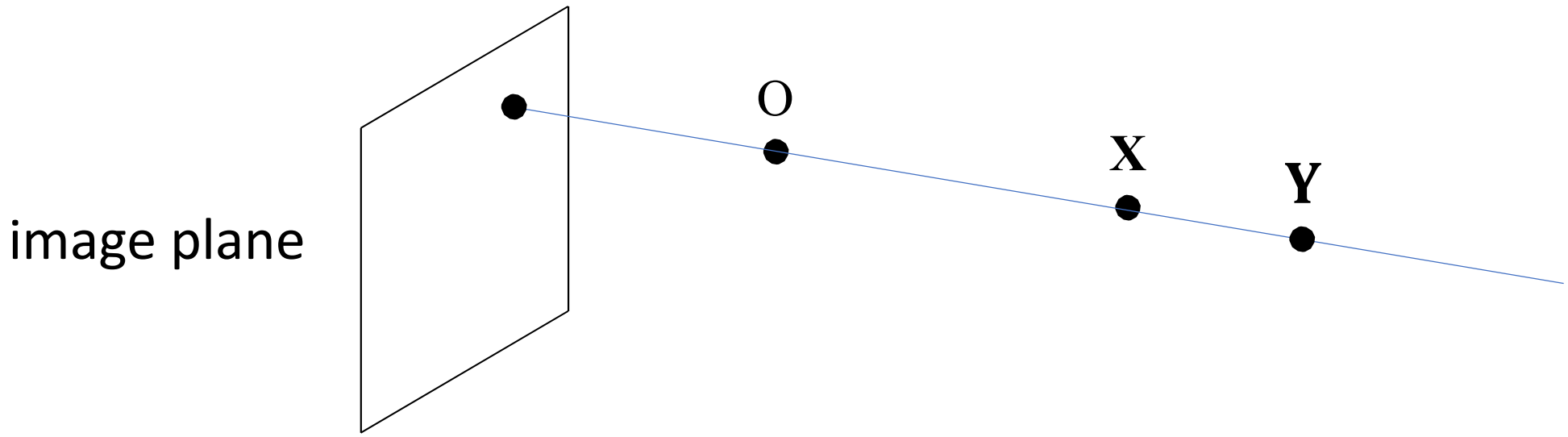
$$\mathrm{O} = RNS(P): \; PO = 0$$



A point $\mathbf{X}$ and its image;
any point $\mathbf{Y}$ on line$(\mathbf{O}, \mathbf{X})$:

$$\mathbf{Y} = \boldsymbol{\alpha}\mathbf{X} + \boldsymbol{\beta}\mathbf{O}$$

$$\mathrm{O} = RNS(P): \; P\mathrm{O} = 0$$



A point $\mathbf{X}$ and its image;

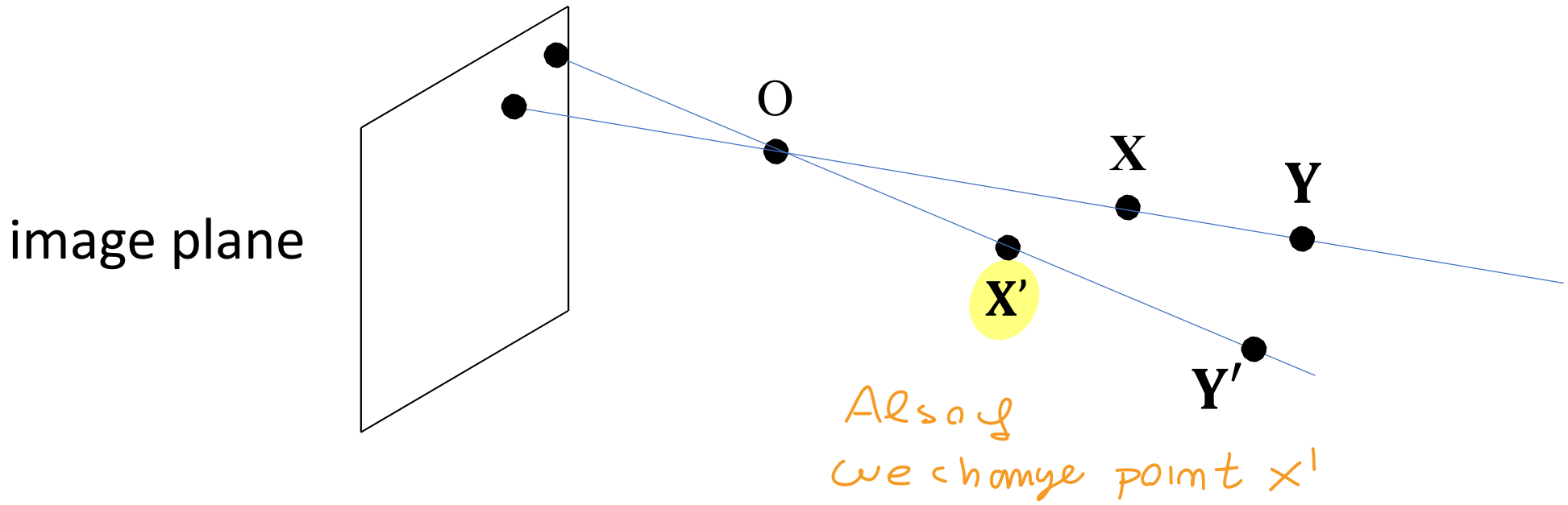
any point $\mathbf{Y}$ on line$(\mathbf{O}, \mathbf{X})$:

$$\mathbf{Y} = \boldsymbol{\alpha}\mathbf{X} + \boldsymbol{\beta}\mathbf{O}$$

also projects onto the image of $\mathbf{X}$

$$P\mathbf{Y} = \boldsymbol{\alpha}P\mathbf{X} + \boldsymbol{\beta}P\mathrm{O} = \boldsymbol{\alpha}P\mathbf{X} \approx P\boldsymbol{X}$$

the same for any other point



A point $\mathbf{X}$ and its image;

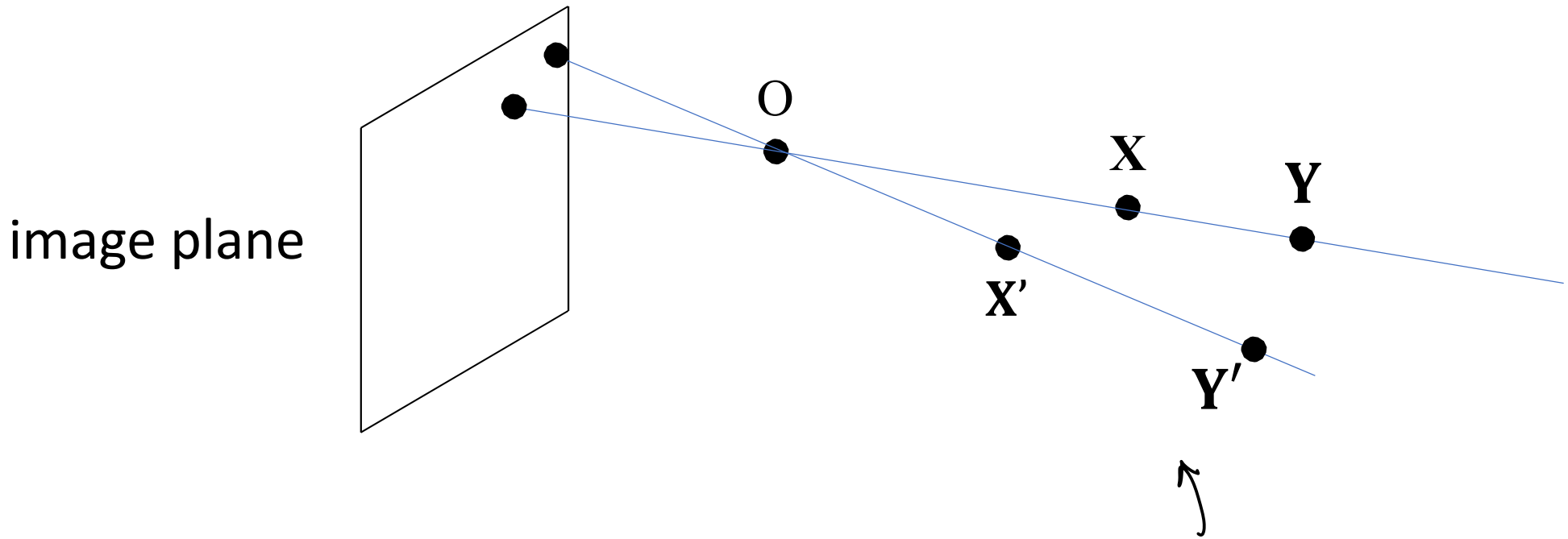
any point $\mathbf{Y}$ on line$(\mathbf{O}, \mathbf{X})$:

$$\mathbf{Y} = \boldsymbol{\alpha}\mathbf{X} + \boldsymbol{\beta}\mathbf{O}$$

also projects onto the image of $\mathbf{X}$

$$P\mathbf{Y} = \boldsymbol{\alpha}P\mathbf{X} + \boldsymbol{\beta}P\mathrm{O} = \boldsymbol{\alpha}P\mathbf{X} \approx P\boldsymbol{X}$$

the same for any other point



all viewing rays go through $\mathbf{O}$

A point $\mathbf{X}$ and its image;

any point $\mathbf{Y}$ on line$(\mathbf{O}, \mathbf{X})$:

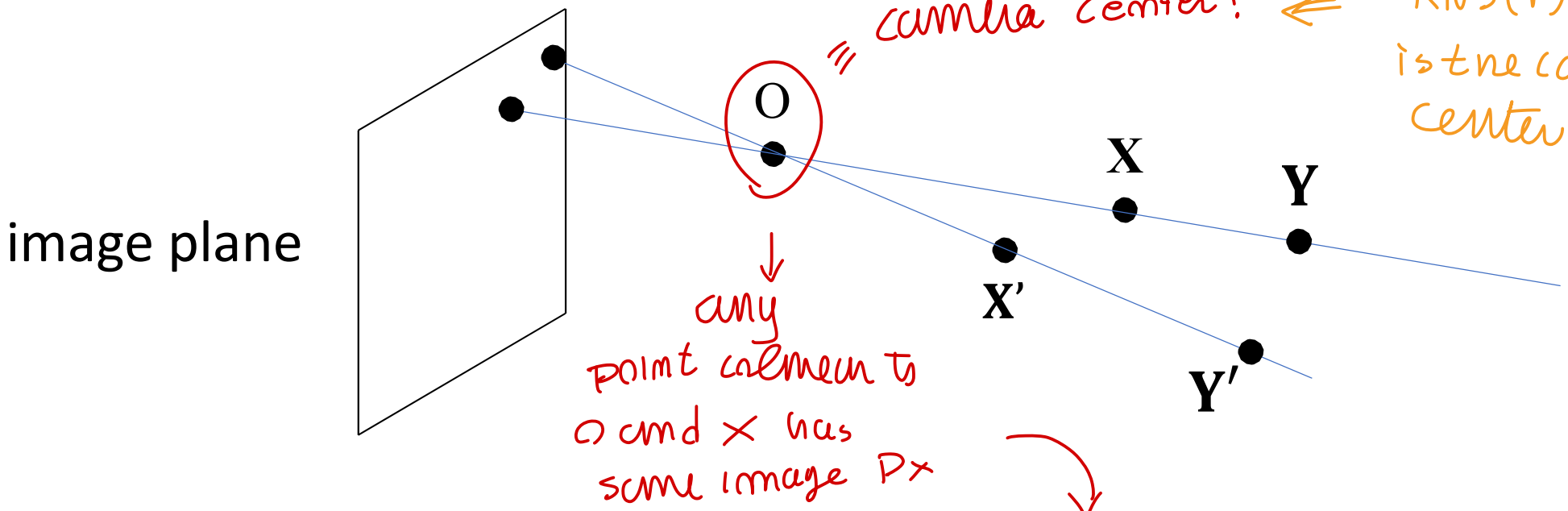
$$\mathbf{Y} = \boldsymbol{\alpha}\mathbf{X} + \boldsymbol{\beta}\mathbf{O}$$

also projects onto the image of $\mathbf{X}$

$$P\mathbf{Y} = \boldsymbol{\alpha}P\mathbf{X} + \boldsymbol{\beta}P\mathrm{O} = \boldsymbol{\alpha}P\mathbf{X} \approx P\boldsymbol{X}$$

the same for any other point

Given P, we compute RNS(P) = O is the camera center!



A point $\mathbf{X}$ and its image;

any point $\mathbf{Y}$ on line$(\mathbf{O}, \mathbf{X})$:

$$\mathbf{Y} = \boldsymbol{\alpha}\mathbf{X} + \boldsymbol{\beta}\mathbf{O}$$

also projects onto the image of $\mathbf{X}$

$$P\mathbf{Y} = \boldsymbol{\alpha}P\mathbf{X} + \boldsymbol{\beta}P\mathrm{O} = \boldsymbol{\alpha}P\mathbf{X} \approx P\boldsymbol{X}$$

all viewing rays go through **O**

O is the camera lens center

≃ pinhole of old camera model!

the same for any other point

IF $PO = 0$

$[M \quad m]\begin{bmatrix} O \\ 1 \end{bmatrix} = 0$

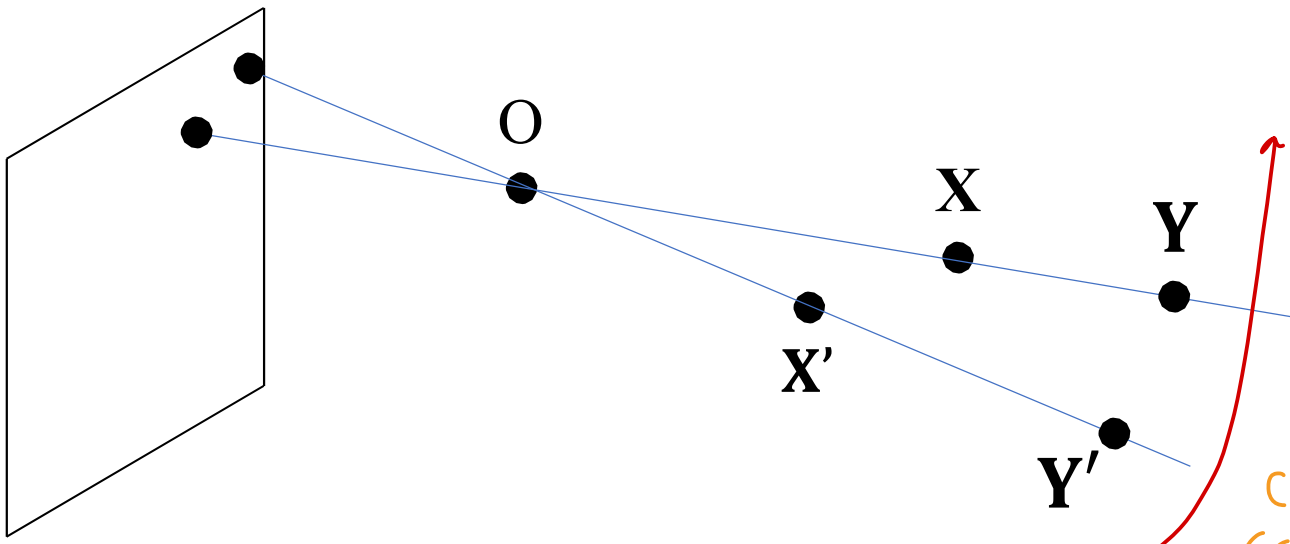
$MO + m = 0$

$O = -M^{-1}m$

When computing RNS, decomposing O

⇐ as 4 homogeneous coord $O = [O \quad 1]$ (NOT at ∞ POINT; $W \neq 0$!)

↑ cartesian coordinate of camera center



from P, we can identify camera center in cartesian coord.

A point $\mathbf{X}$ and its image;

any point $\mathbf{Y}$ on line$(\mathbf{O}, \mathbf{X})$:

$$\mathbf{Y} = \boldsymbol{\alpha}\mathbf{X} + \boldsymbol{\beta}\mathbf{O}$$

also projects onto the image of $\mathbf{X}$

$$P\mathbf{Y} = \boldsymbol{\alpha}P\mathbf{X} + \boldsymbol{\beta}P\mathrm{O} = \boldsymbol{\alpha}P\mathbf{X} \approx P\boldsymbol{X}$$

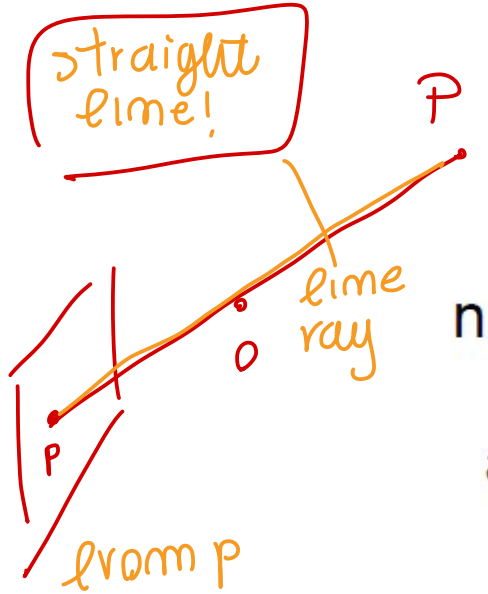
all viewing rays go through $\mathbf{O}$

$\mathbf{O}$ is the camera lens center

$$PO = [\mathbf{M} \mid \mathbf{m}]\begin{bmatrix} \mathbf{O} \\ 1 \end{bmatrix} = 0$$

$$\rightarrow \mathbf{o} = -\mathbf{M}^{-1}\mathbf{m}$$

cartesian coordinates of $\mathbf{O}$



↳ What is capability of identify viewing ray of any image point

SCENE: viewing ray from image point

null-space of camera projection matrix **O** $\quad \mathbf{PO} = 0$

a point **Y** on the line **X**, **O** $\quad \mathbf{Y} = \alpha\mathbf{X} + \beta\mathbf{O}$

its image $\quad \mathbf{u} = \mathbf{PY} = \alpha\mathbf{PX} + \beta\mathbf{PO} = \mathbf{PX}$

all points Y on (X,O) project on image of X, → **all points on a line through O share the same image**

→ **O** is camera center

Image of camera center is $(0,0,0)^T$, i.e. undefined

Finite cameras: $\mathbf{O} = \begin{pmatrix} \mathbf{o} \\ 1 \end{pmatrix} = \begin{pmatrix} -\mathrm{M}^{-1}\mathbf{m} \\ 1 \end{pmatrix}$

from p image point ↓ where is the ray in the space? ⇓ all set of 3D points projecting on point p

↳ this is a line because light straight line for our hypotesis

here we assume ray linear (straight) in HOMOGENEOUS MEDIUM ⇑ all changes when putting camera in fluid medium with non homog density! there light ray deviate, no straight line

→ Given an image point u, what is the line containing all the points who's image is u

↳ "viewing ray" associated to an image point u

$\Leftrightarrow$

set of all points X in 3D space projecting in u

IF we know direction, we can uniquely identify it ray

we can derive it cleverly

(we miss DIRECTION..)

Viewing ray associated to image point u:

↕ being a straight line ⇒ it has a point at infinity!

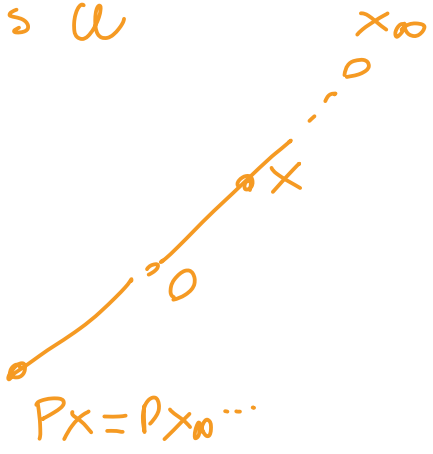
Also image of X_∞ of this is u

image point **u** is image of **X** if **X** is on a certain line through **O**

viewing ray associated to u

Consider the point at the infinity along this line $\mathbf{X} = \begin{vmatrix} \mathbf{d} \\ 0 \end{vmatrix}$

$$\mathbf{u} = \begin{vmatrix} \mathbf{M} & \mathbf{m} \end{vmatrix} \begin{vmatrix} \mathbf{d} \\ 0 \end{vmatrix} = \mathbf{M} \cdot \mathbf{d}$$



we know that all viewing rays goes through O point

⇓

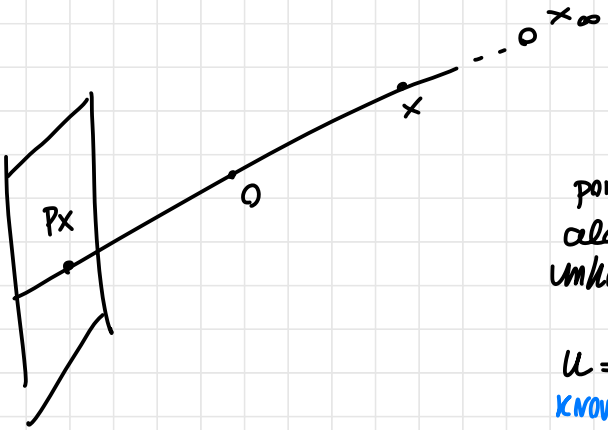
we JUST miss direction information...

The locus of the points **x** *whose image is* **u**

is a straight line through **o** *having direction* $\mathbf{d} = \mathbf{M}^{-1} \cdot \mathbf{u}$

o is the camera viewpoint (perspective projection center)

line(**o**, **d**) = *viewing ray associated to image point* **u**



IF we select x_∞ along my viewing ray

ray

$x_\infty = \begin{bmatrix} d \\ 0 \end{bmatrix}$ direction of line UNKNOWN

point at ∞

point at ∞ along this unknown ray, impose

$u = Px_\infty = [M \quad m]\begin{bmatrix} d \\ 0 \end{bmatrix} = Md$

u KNOWN, d unk, M KNOWN

INVERTIBLE

$d = M^{-1}u$

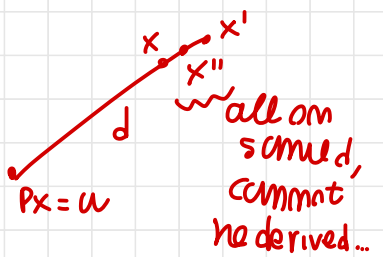
IF we KNOW M, d is easy to find

d: $\mathbb{R}^3$ vector with viewing ray direction parameters

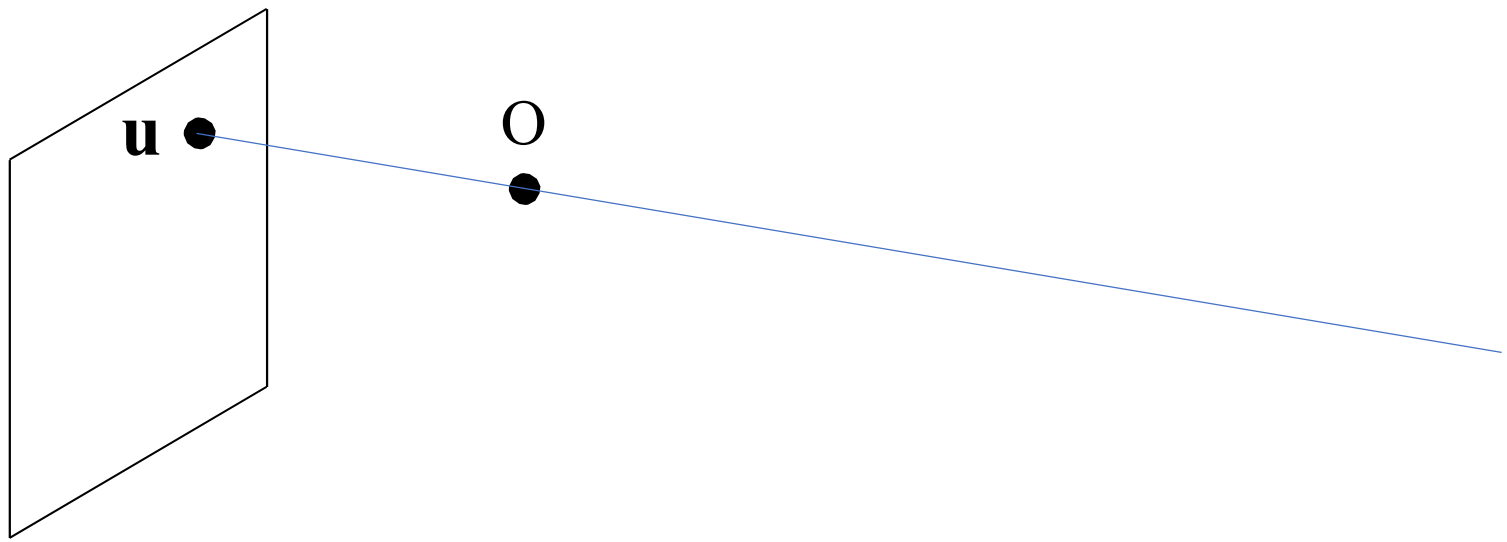
Therefore, KNOWING M, m projection matrix (P) of the camera

- we can both identify the viewpoint (camera center O)
- and $\forall$ u image point → we can derive d, the viewing ray line of it!

this line is the most information that can be extracted from 2D represent. of 3D scene (you lose info on 3D scene, you have 2D)



Viewing ray associated to an image point **u**: the «backprojection» of **u**



$$\mathrm{O} = RNS(P)\colon\ PO = 0$$

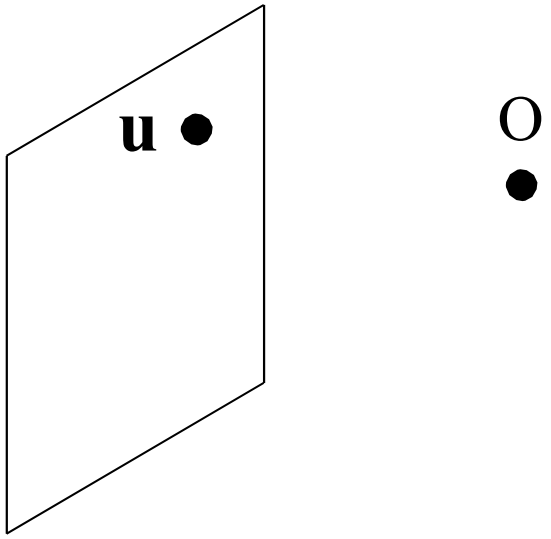
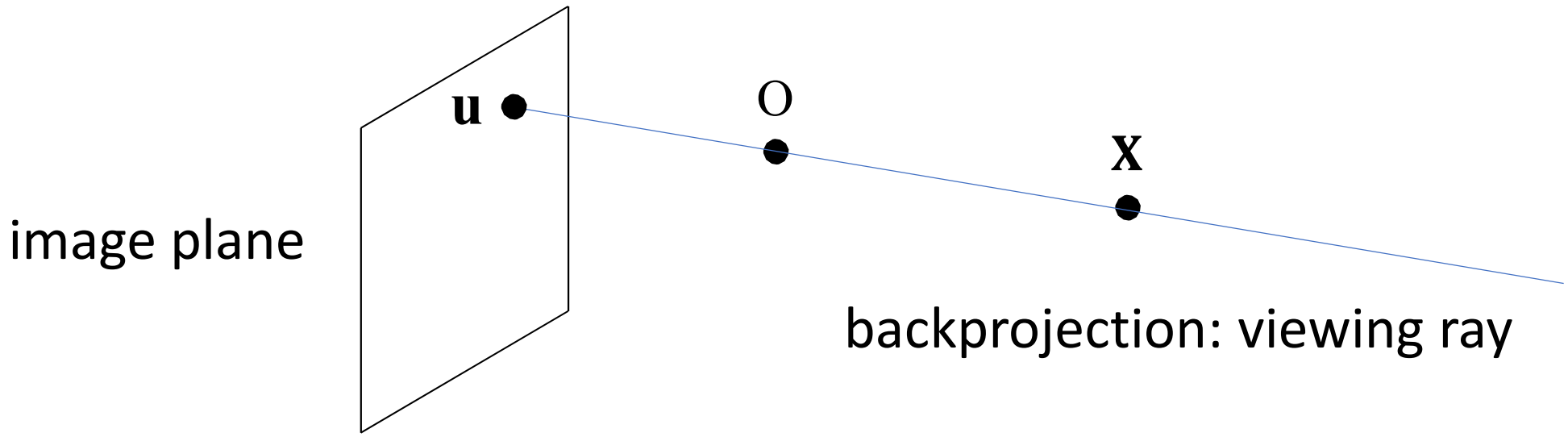


image plane

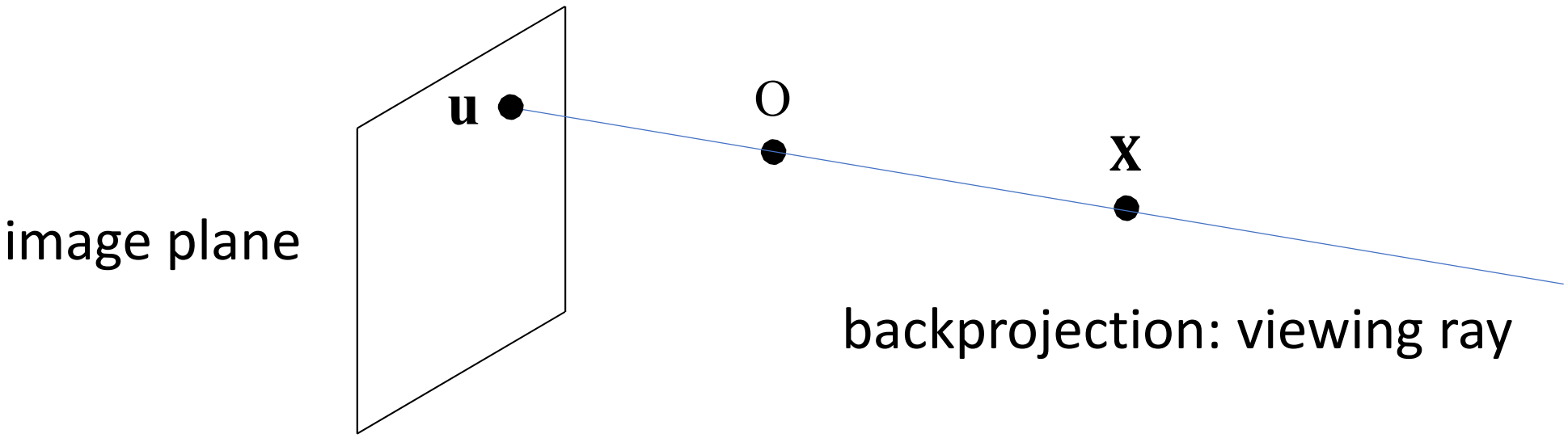
An image point **u**

$$\mathrm{O} = RNS(P): \; PO = 0$$



An image point **u** and its backprojection: set of 3D points **X** projecting onto **u**

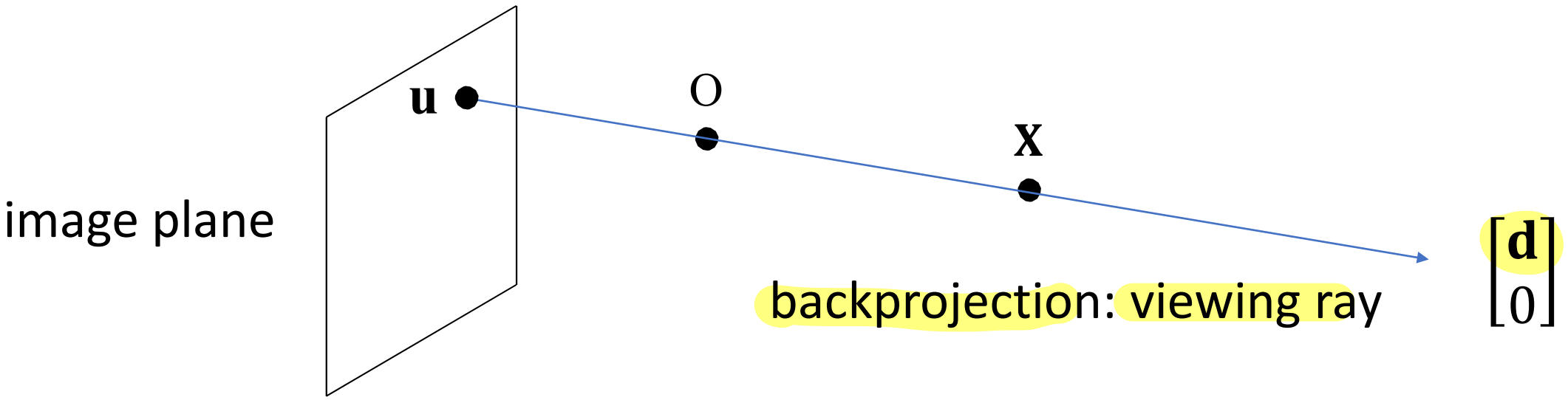
$$\mathrm{O} = RNS(P): \; P\mathrm{O} = 0$$



An image point $\mathbf{u}$ and its backprojection: set of 3D points $\mathbf{X}$ projecting onto $\mathbf{u}$

(i) a line through O

$$\mathrm{O} = RNS(P): \; PO = 0$$

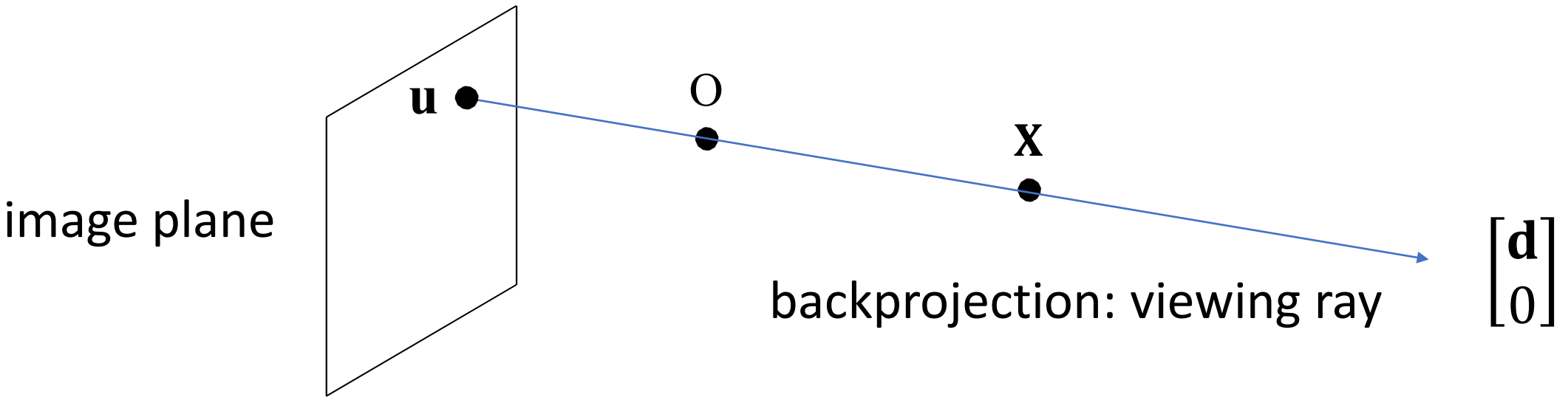


An image point $\mathbf{u}$ and its backprojection: set of 3D points $\mathbf{X}$ projecting onto $\mathbf{u}$

(i) a line through O

(ii) whose direction $\mathbf{d}$ …

$$\mathrm{O} = RNS(P): \; PO = 0$$

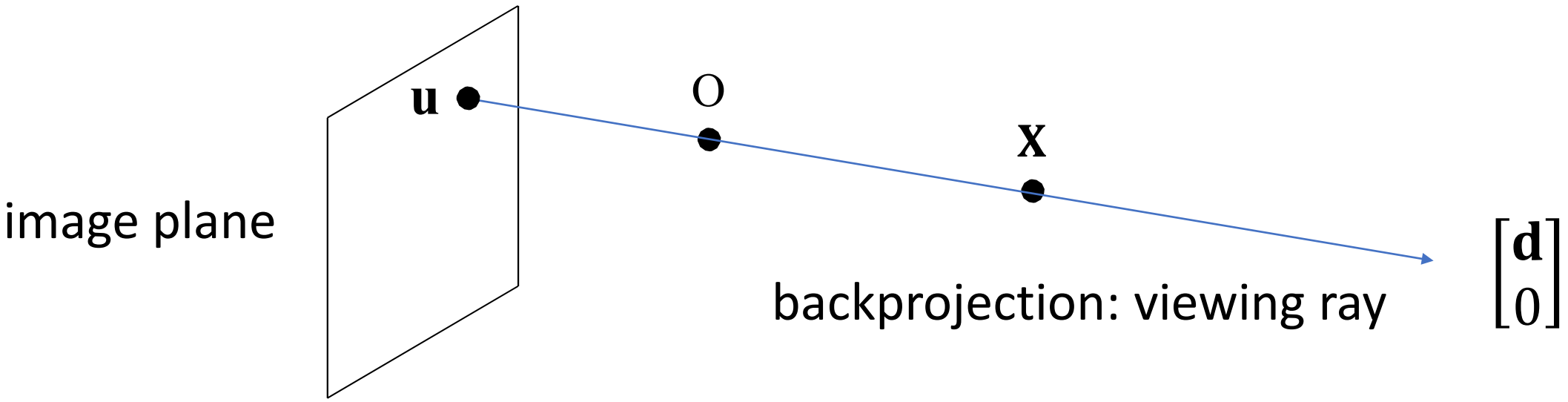


An image point $\mathbf{u}$ and its backprojection: set of 3D points $\mathbf{X}$ projecting onto $\mathbf{u}$

(i) a line through O

(ii) whose direction $\mathbf{d}$ satisfies $\mathbf{u} = [\mathbf{M} \quad \mathbf{m}] \begin{bmatrix}\mathbf{d}\\0\end{bmatrix} = \mathbf{Md}$

$$\mathrm{O} = RNS(P): \; PO = 0$$



An image point $\mathbf{u}$ and its backprojection: set of 3D points $\mathbf{X}$ projecting onto $\mathbf{u}$

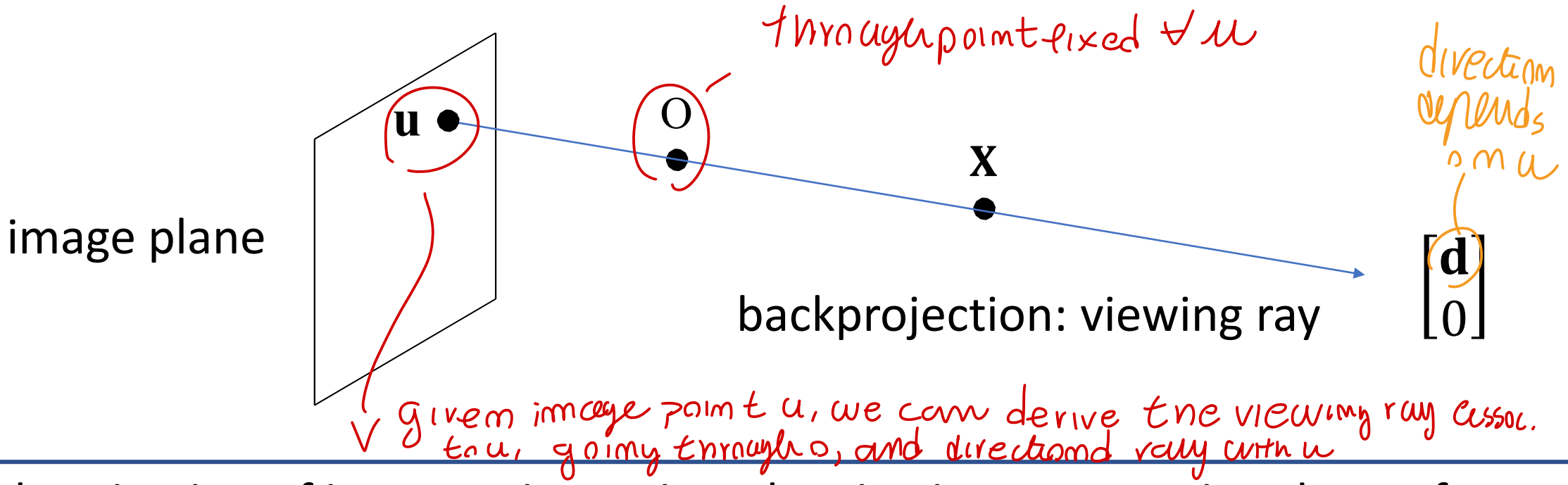
(i) a line through O

(ii) whose direction $\mathbf{d}$ satisfies $\mathbf{u} = [\mathbf{M} \quad \mathbf{m}] \begin{bmatrix}\mathbf{d}\\0\end{bmatrix} = \mathbf{M}\mathbf{d}$

→

$$\mathbf{d} = \mathbf{M}^{-1}\mathbf{u}$$

the viewing ray associated to an image point $\mathbf{u}$



The backprojection of image point $\mathbf{u}$, i.e., the viewing ray associated to $\mathbf{u}$, for the camera $P = [\mathbf{M} \quad \mathbf{m}]$, is the straight line

(i) through $\mathrm{O} = RNS(P)$

(ii) whose direction $\mathbf{d}$ is $\mathbf{d} = \mathbf{M}^{-1}\mathbf{u}$

- SCENE
- **CAMERA**

Now we
wanna to
study internal
aspects of camera

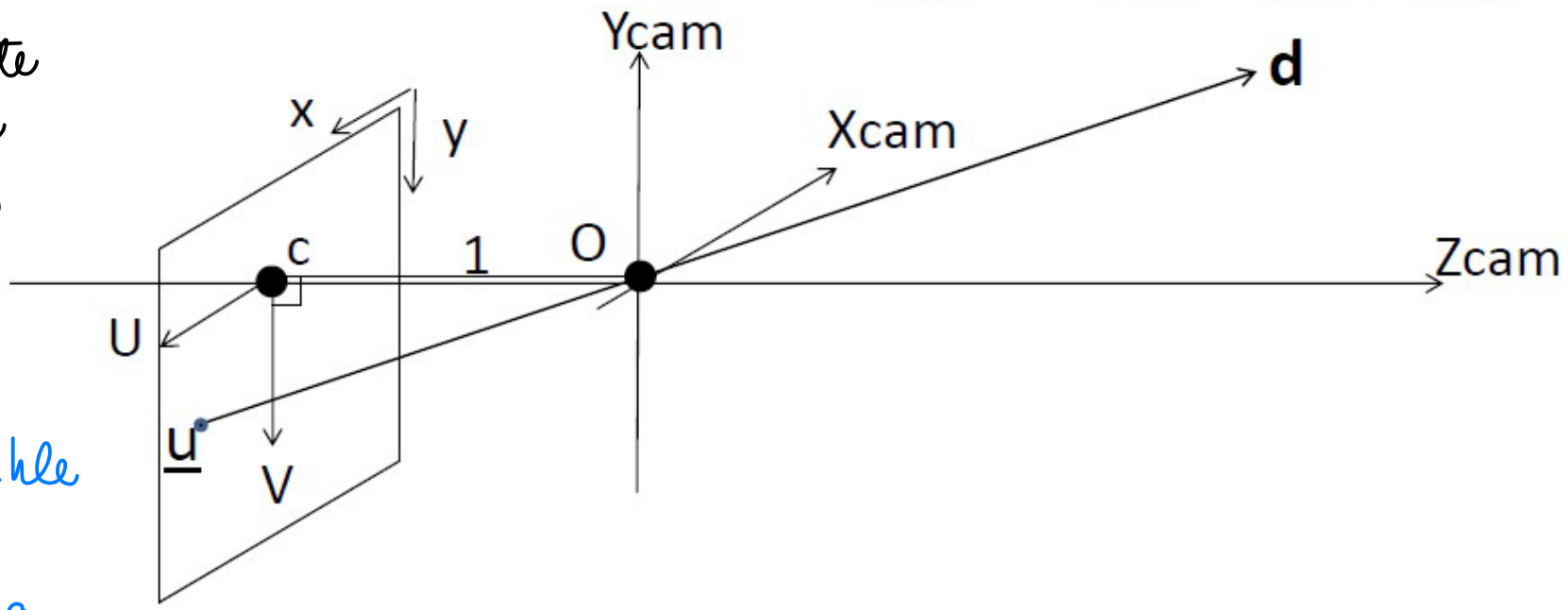
up to now we
use external information of
camera by P matrix (numbers)

We wanna
understand
what is
inside

we already define relationship between coordinate of point in space and its image on image plane

CAMERA

$$\mathbf{d}_{cam} = \mathbf{R}_{cam->world}\mathbf{d}_{world}$$



↓

We use comfortable coordinates

We defined camera reference on O camera centre

geometric coordinates

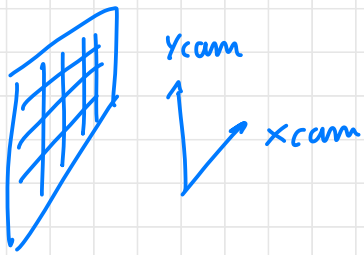
$$\begin{vmatrix} U \\ V \\ 1 \end{vmatrix} = \begin{vmatrix} X \\ Y \\ -1 \end{vmatrix} \propto \mathbf{d}_{cam}$$

with axis ⊥ to image plane

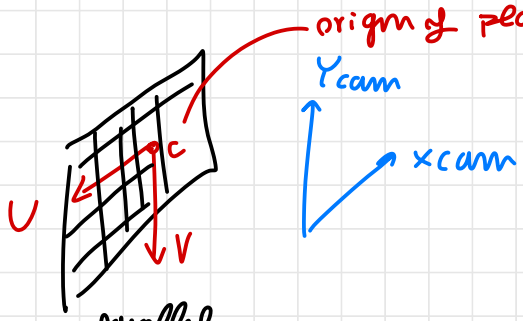
with x, y axis // to image plane

(good practice to chose x, y cam directed such that Xcam // pixel side

Chose x_{cam}, y_{cam} such that // to pixels matrix (as rectang. pixel)



+ defined reference on image plane s.t x, y same but opposite direct

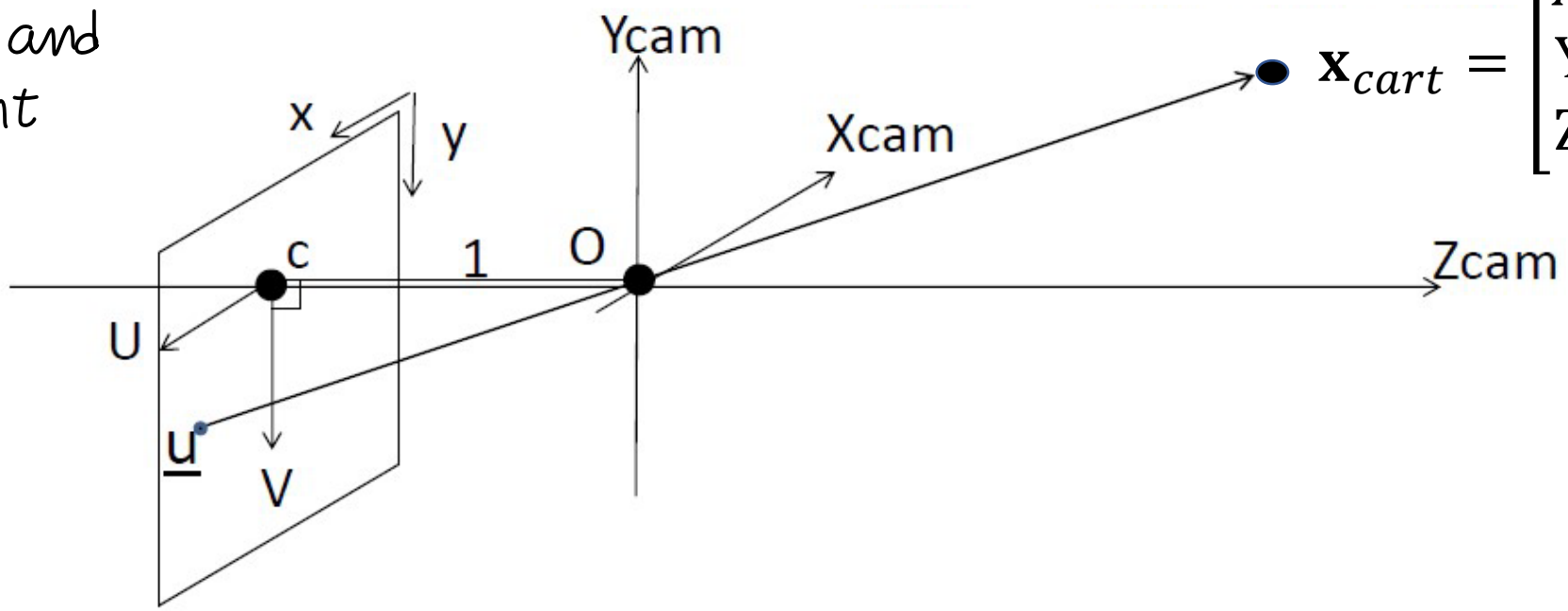


⇓ orthogonal projection of camera center on image plane

⇓ NATURAL system, simple !

camera

we have to use co-linearity between scene point and image point with view point

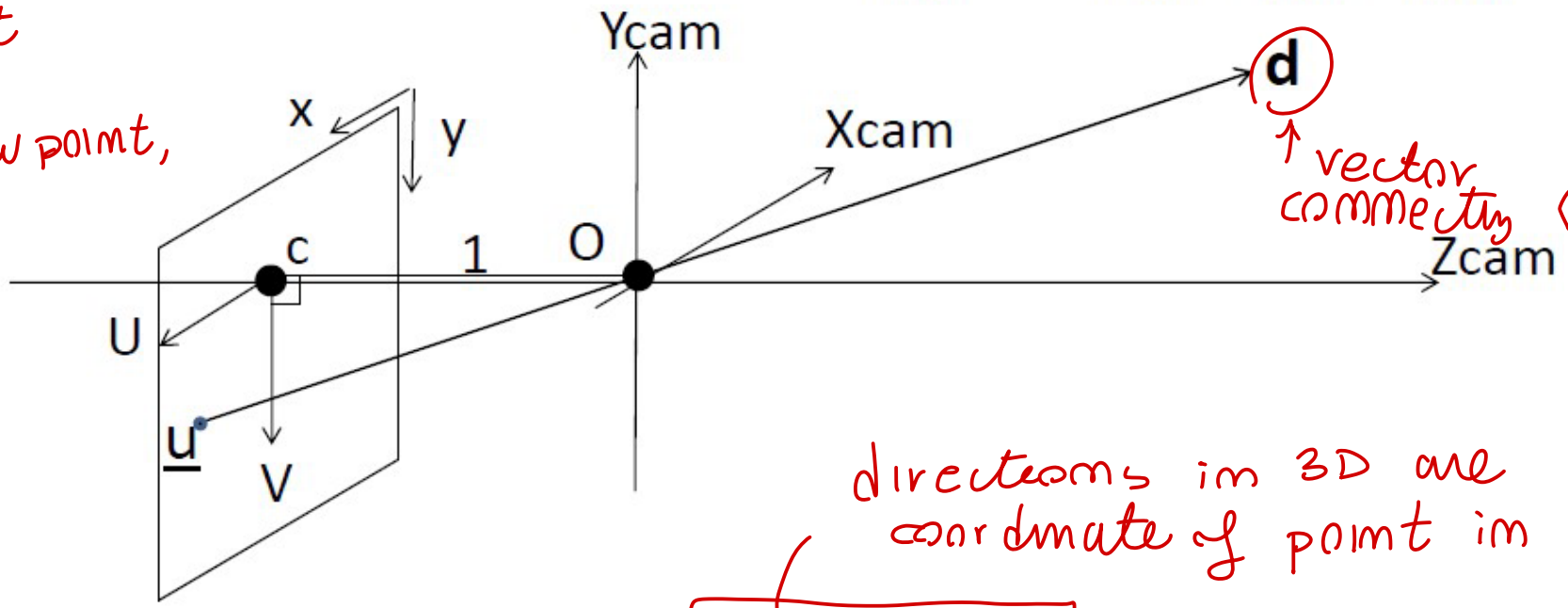


$$\mathbf{x}_{cart} = \begin{bmatrix} \mathrm{X}_{cam} \\ \mathrm{Y}_{cam} \\ \mathrm{Z}_{cam} \end{bmatrix}$$

geometric coordinates $\begin{bmatrix} U \\ V \\ 1 \end{bmatrix} \propto \begin{bmatrix} \mathrm{X}_{cam} \\ \mathrm{Y}_{cam} \\ \mathrm{Z}_{cam} \end{bmatrix} = \mathbf{x}_{cart}$

CAMERA

$$\mathbf{d}_{cam} = \mathbf{R}_{cam->world}\mathbf{d}_{world}$$



to use colinearity between u img point P scene and O view point,

d direction and uO is same

⇓

COLINEARITY

Write parameter to describe direction of vector connecting

Ou are $[U \ V \ 1]^T$ in 3D in this reference

this is same direction connecting O with 3D point

directions in 3D are coordinate of point in ref frame

$$\begin{vmatrix} U \\ V \\ 1 \end{vmatrix} = \begin{vmatrix} X \\ Y \\ -1 \end{vmatrix} \propto \mathbf{d}_{cam}$$

cartesian coordinate of point

vector connecting O and u

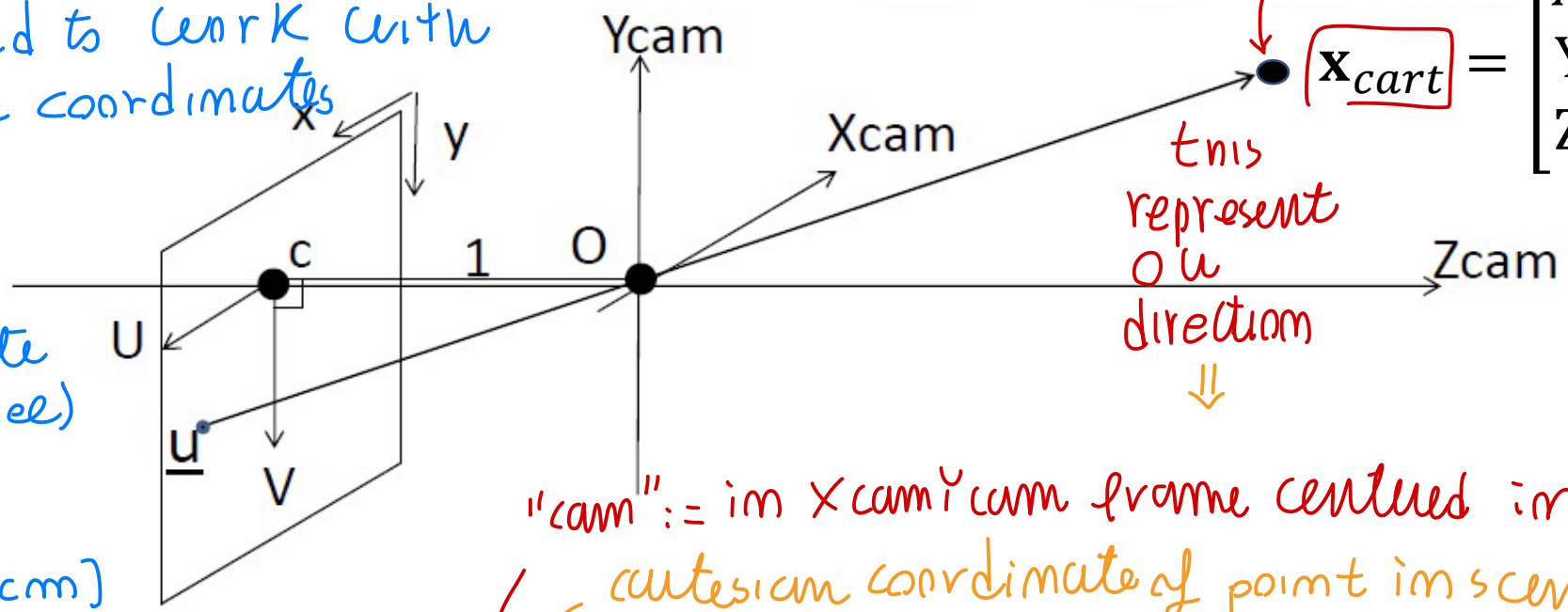
geometric coordinates

camera

to work with digital images stored in memory of PC, we need to work with pixel coordinates

we used geometric real coordinate axis (NOT pixel)

unit in [cm] as spatial! in pixel usually is in [µm]



cartesian coord of point

$$\mathbf{x}_{cart} = \begin{bmatrix} X_{cam} \\ Y_{cam} \\ Z_{cam} \end{bmatrix}$$

this represent ou direction

"cam" := in $X_{cam} Y_{cam}$ frame centred in O

cartesian coordinate of point in scene and due to co-linearity, this is co-linear with [U V 1] proportional!

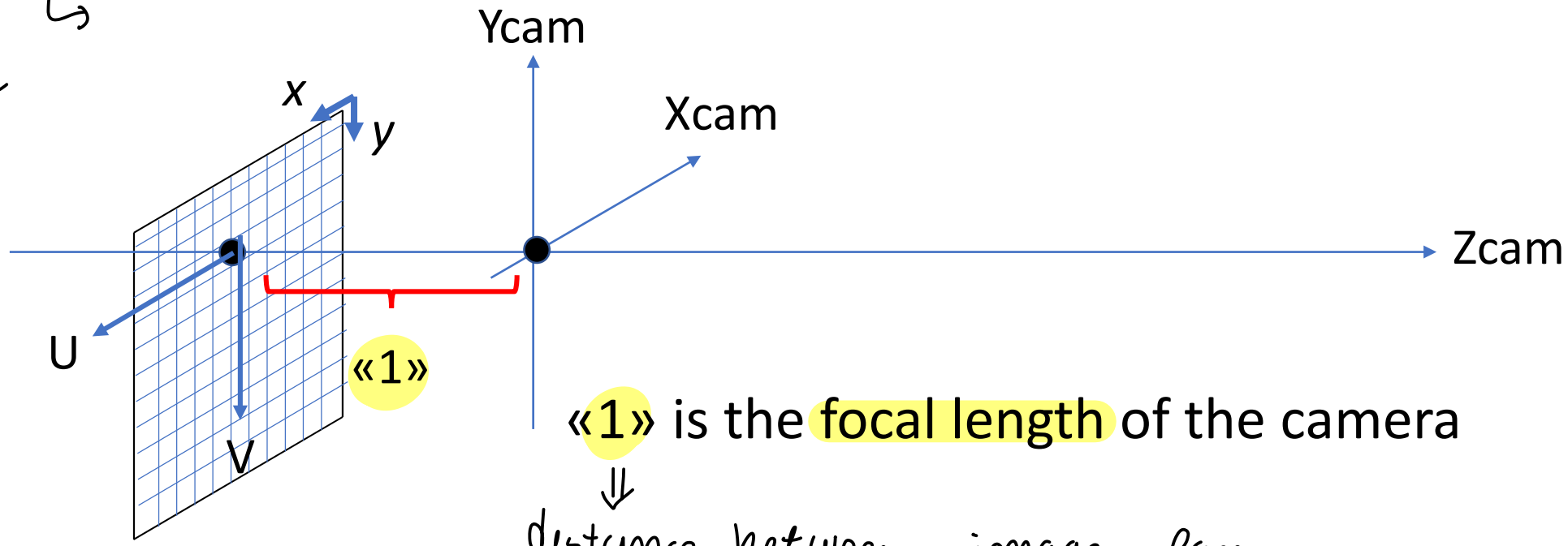
geometric coordinates $\begin{bmatrix} U \\ V \\ 1 \end{bmatrix} \propto \begin{bmatrix} X_{cam} \\ Y_{cam} \\ Z_{cam} \end{bmatrix} = \mathbf{x}_{cart}$

we have do deal with this

geometric coordinates vs pixel coordinates ??

up to now we use geometric coord.
we need PIXEL COORD
↳
cost of difference

pixel coordinates



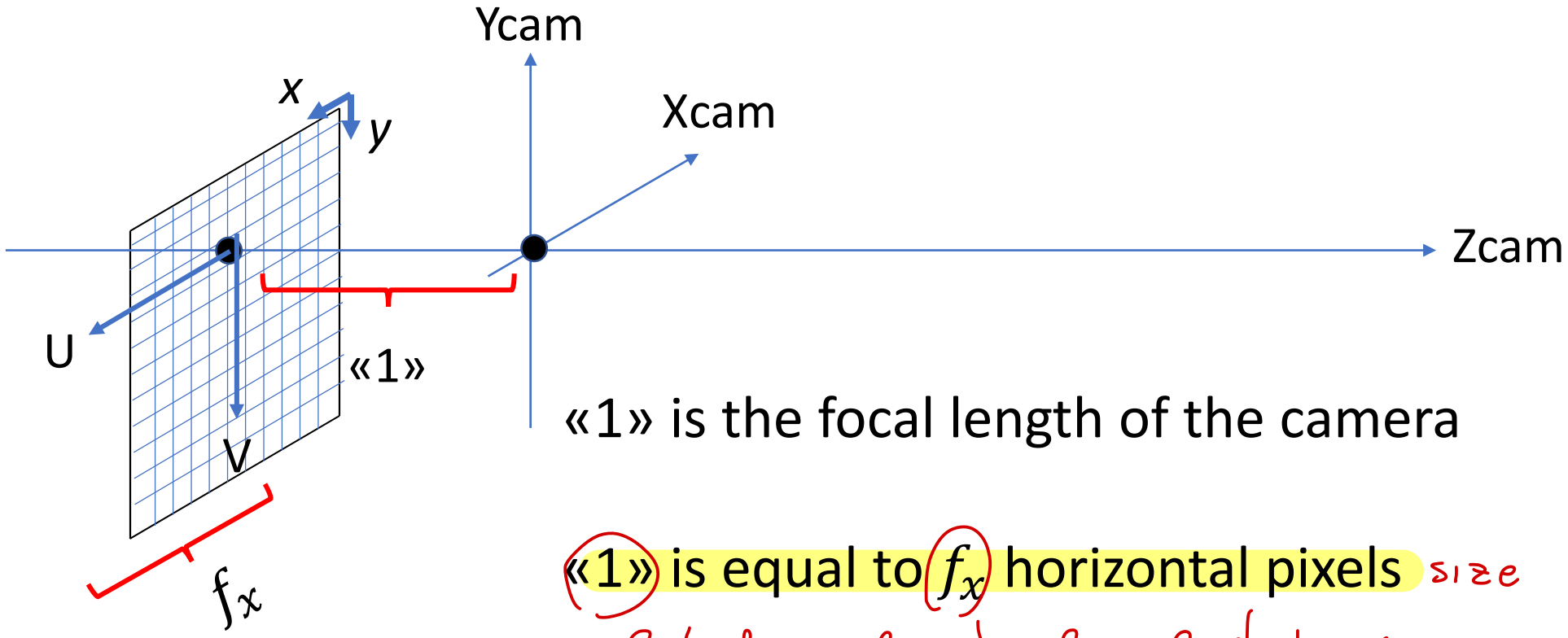
«1» is the focal length of the camera

⇓

distance between image plane and camera center in our unit

(NOT 1 in pixel coord)

pixel coordinates



«1» is the focal length of the camera

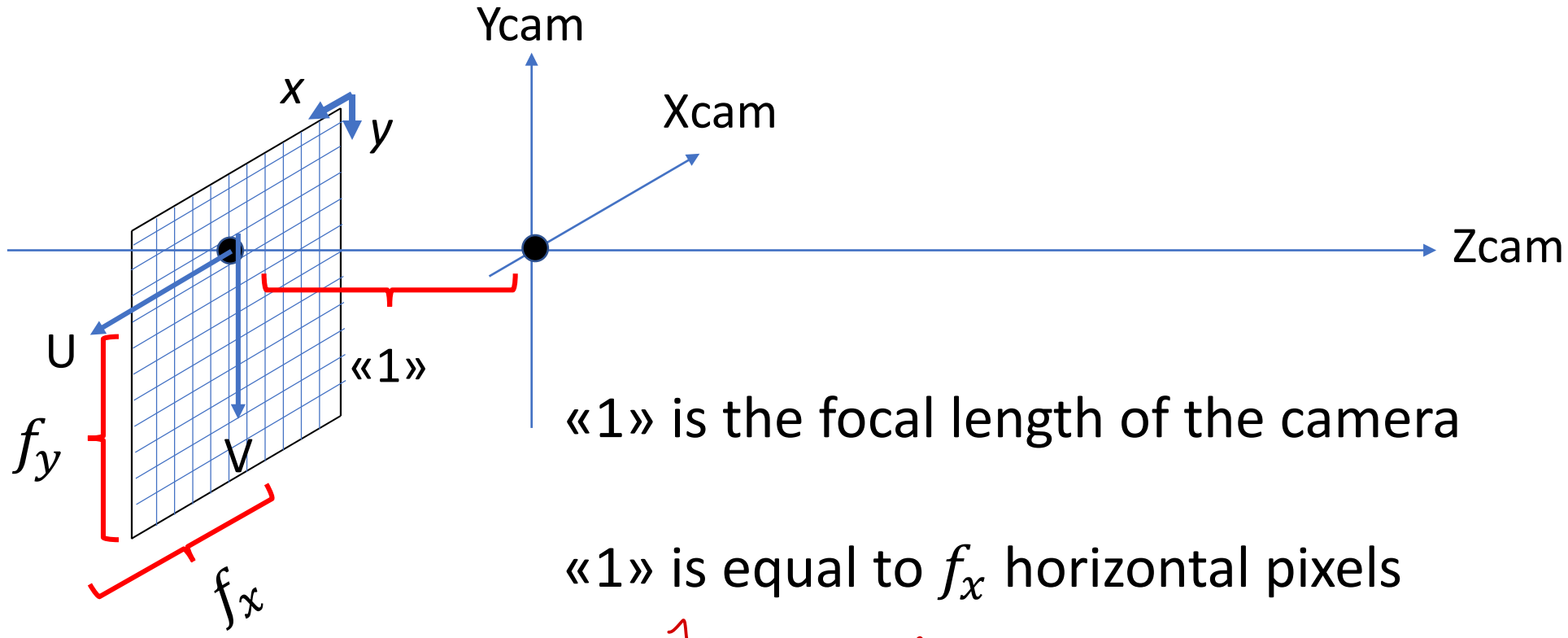
«1» is equal to f_x horizontal pixels

size

«1» ← unit of pixels

f_x ← focal distance camera-plane in pixels

pixel coordinates



«1» is the focal length of the camera

«1» is equal to f_x horizontal pixels

↕ in general pixels are rectangular, NOT squared

«1» is equal to f_y vertical pixels

$\neq f_x$

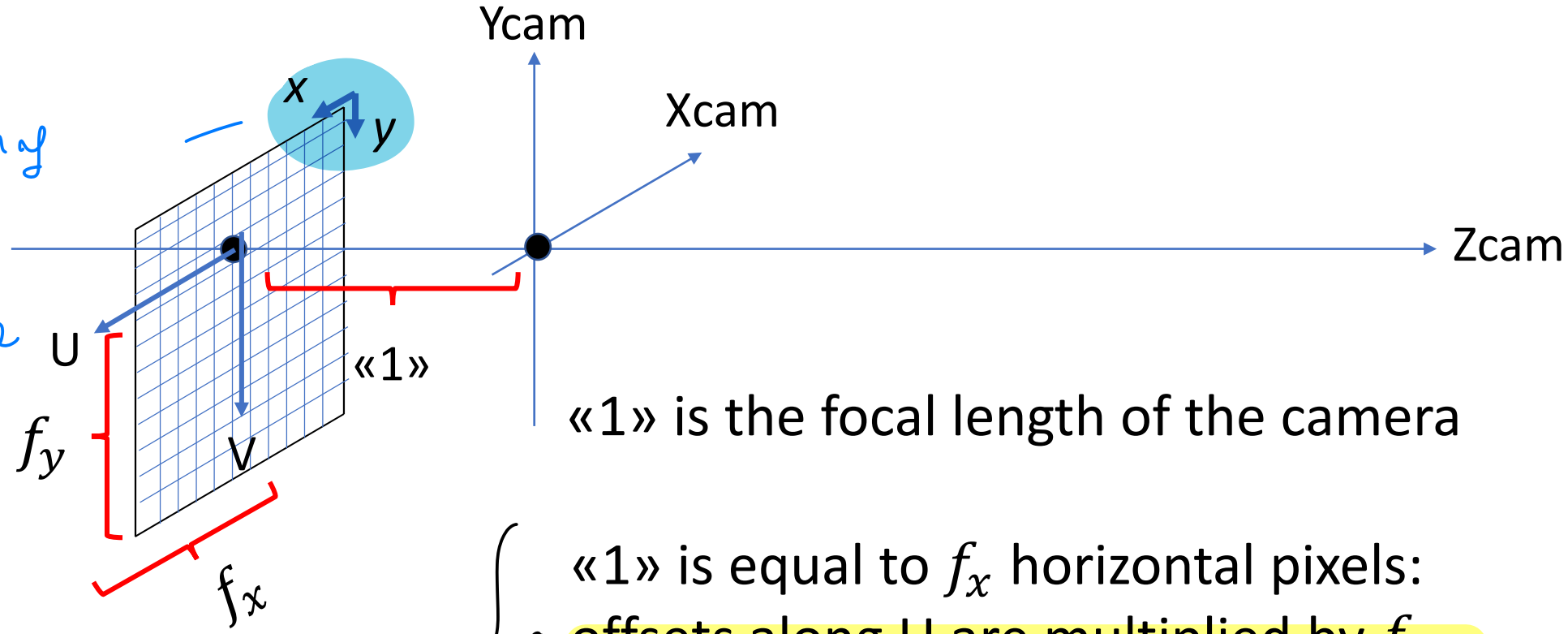
{ sometimes squared pixels, $f_x/f_y = 1$, is called NATURAL CAMERA }

⇓

$\frac{f_y}{f_x}$:= Aspect ratio of pixels ~ 4/3 often

Geometric coord
≠
pixel coord

pixel coordinates



the origin of pixel is in a VERTEX of the image plane rectangular matrix

«1» is the focal length of the camera

«1» is equal to f_x horizontal pixels:

- offsets along U are multiplied by f_x (different scaling)

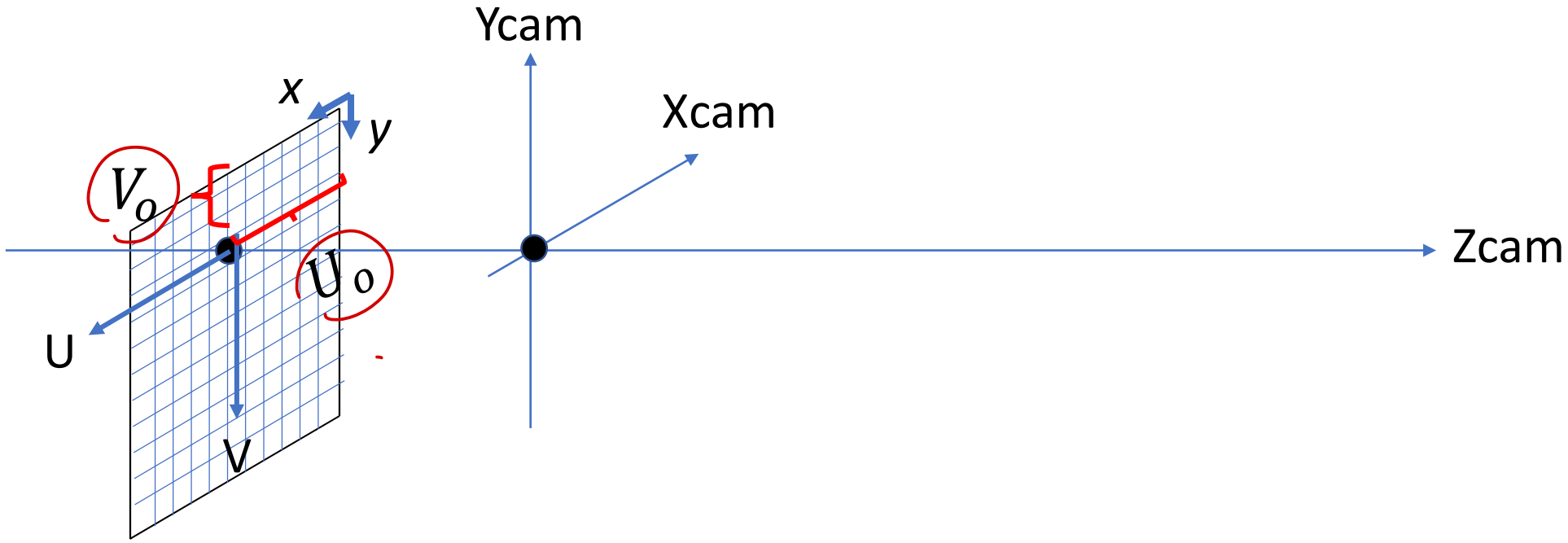
«1» is equal to f_y vertical pixels:

- offsest along V are mutiplied by f_y (scaling)

Geometric ← pixel

1) scaling factor U, V scaled by fx, fy AFFINE
2) translation of origin

pixel coordinates



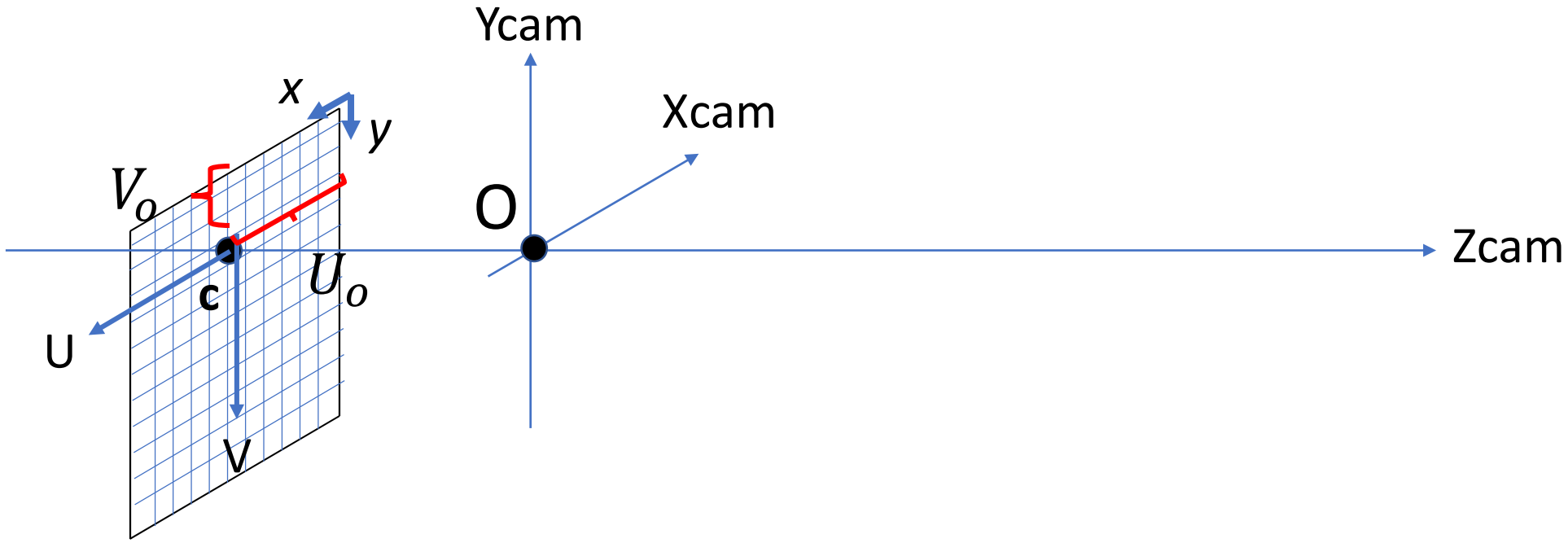
Camera **principal point**: projection of the camera center onto image plane

Pixel coordinates of the principal point:

(U_o, V_o)

to change coord,
translation + scaling

pixel coordinates

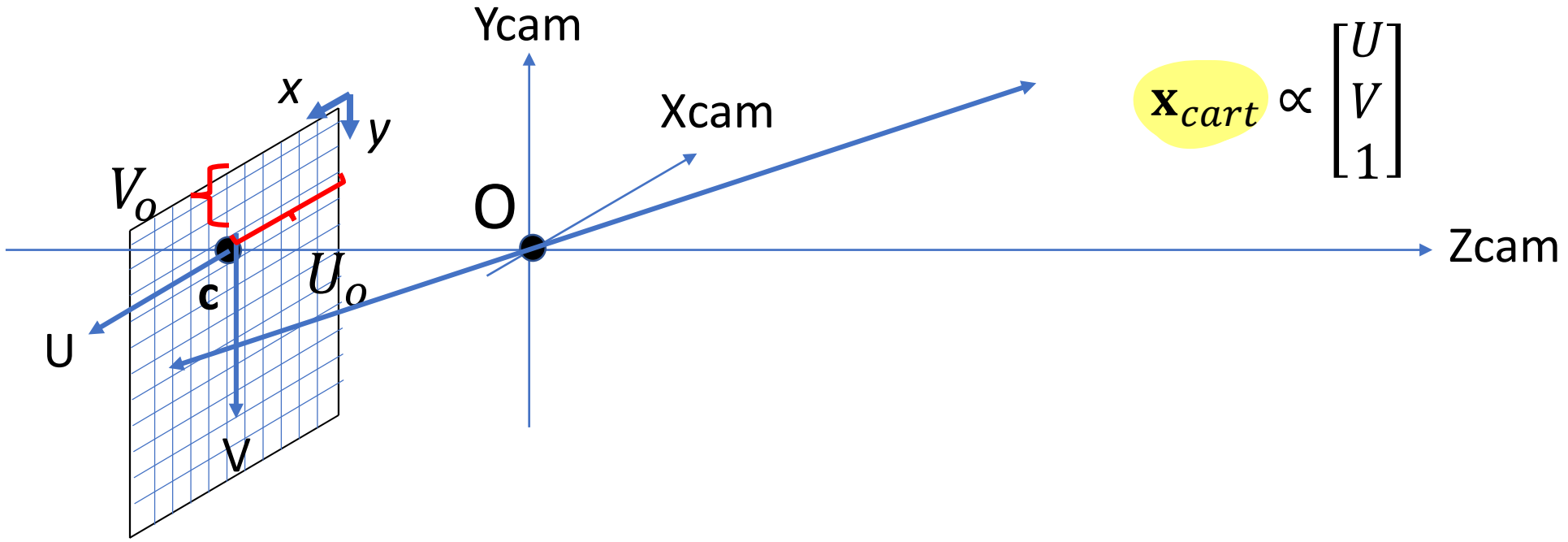


Camera **principal point**: projection of the camera center O onto image plane

Pixel coordinates of the principal point:

$$\mathbf{c} = (U_o, V_o)$$

components of $\mathbf{d}$

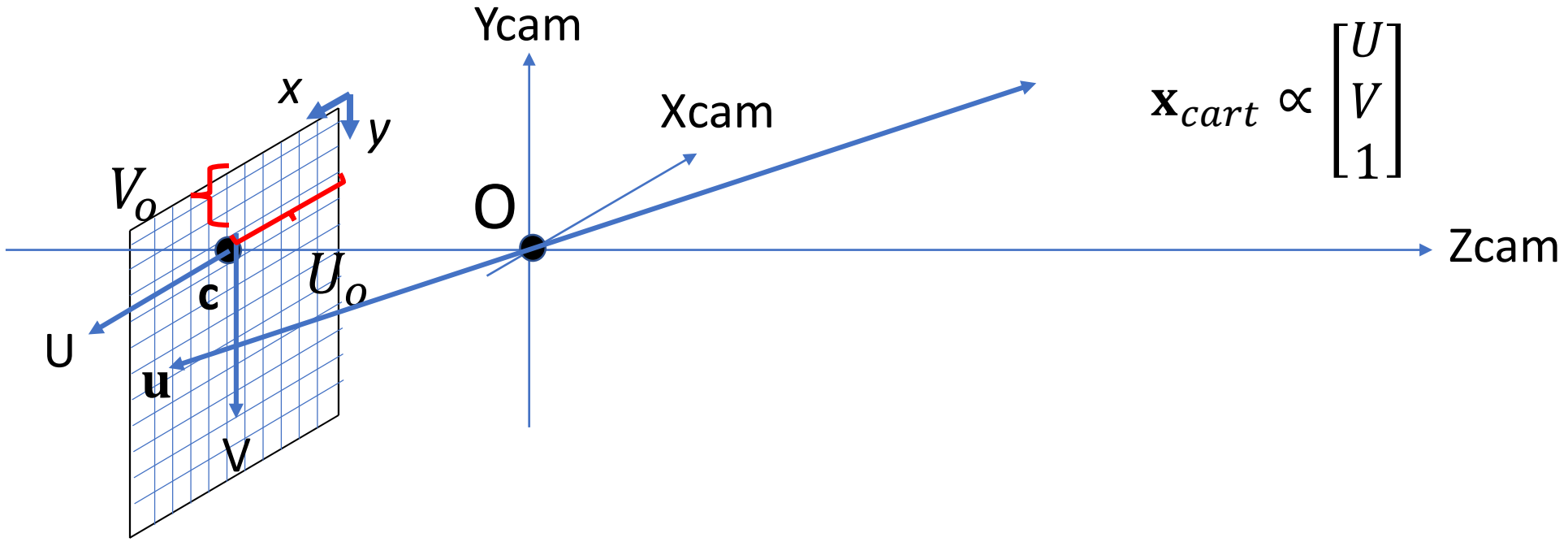


Camera **principal point**: projection of the camera center O onto image plane

Pixel coordinates of the principal point:

$$\mathbf{c} = (U_o, V_o)$$

geometric coordinates of the image point: (U, V)

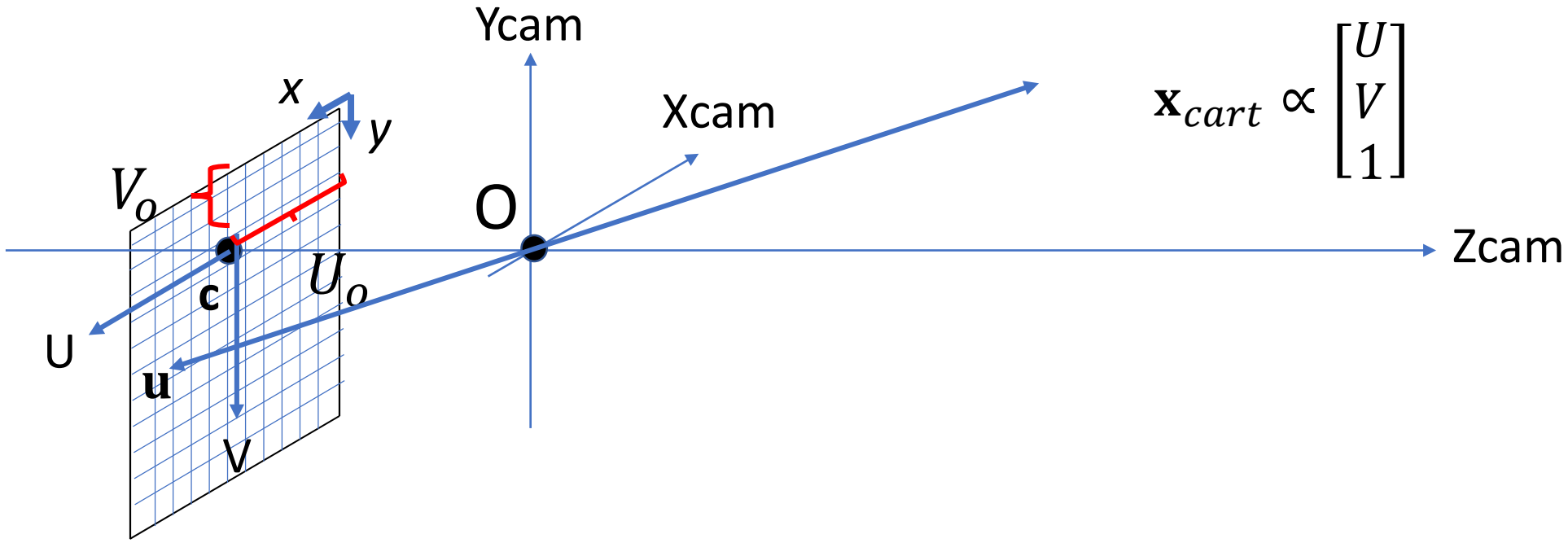


Camera **principal point**: projection of the camera center O onto image plane

Pixel coordinates of the principal point:

$$\mathbf{c} = (U_o, V_o)$$

Cartesian pixel coordinates of the image point: (x, y)

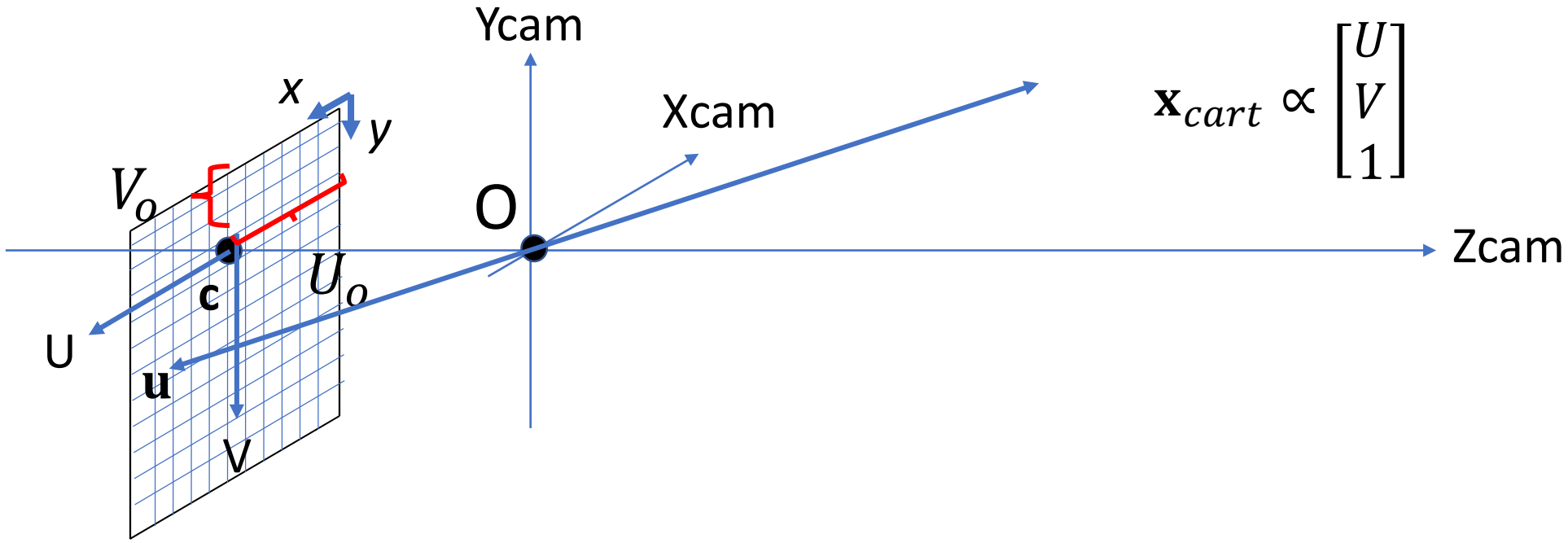


Camera **principal point**: projection of the camera center O onto image plane

Pixel coordinates of the principal point:

$$\mathbf{c} = (U_o, V_o)$$

Homogeneous pixel coordinates of image point: $\mathbf{u} = [x, y, 1]^T$

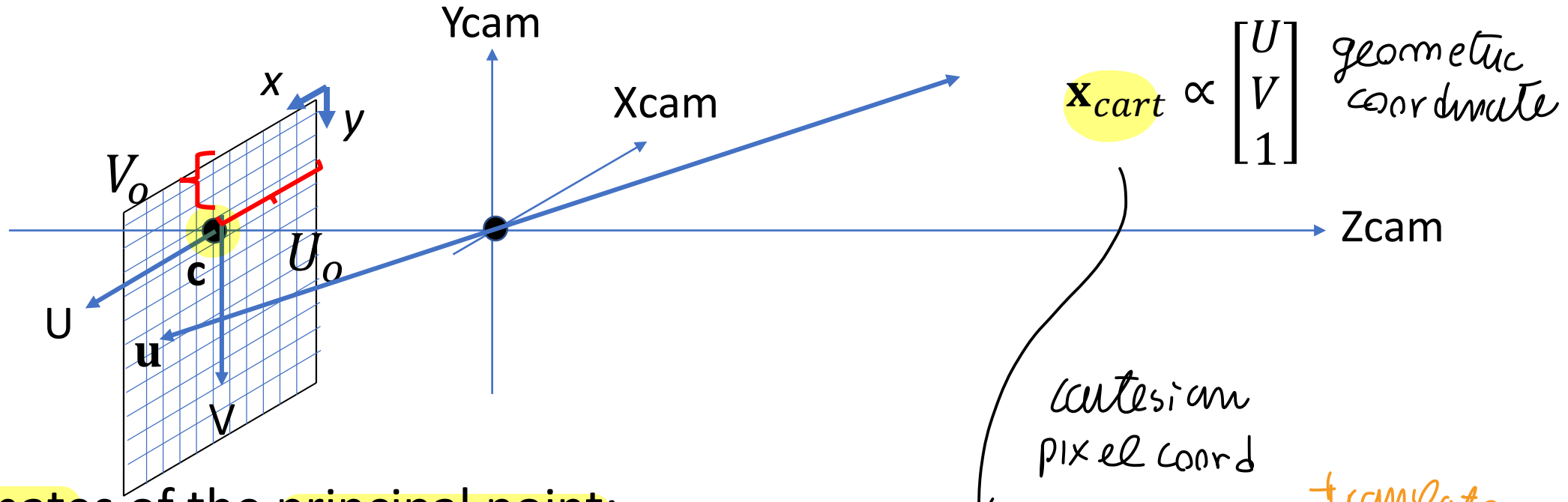


Camera **principal point**: projection of the camera center O onto image plane

Pixel coordinates of the principal point:

$$\mathbf{c} = (U_o, V_o)$$

homogeneous pixel coordinates of image point: $\mathbf{u} = [x, y, 1]^T$



$\mathbf{x}_{cart} \propto \begin{bmatrix} U \\ V \\ 1 \end{bmatrix}$ geometric coordinate

Pixel coordinates of the principal point:

$$\mathbf{c} = (U_o, V_o)$$

«1» is equal to f_x horizontal pixels:
offsets along U are multiplied by f_x

«1» is equal to f_y vertical pixels:
offsest along V are mutiplied by f_y

cartesian pixel coord

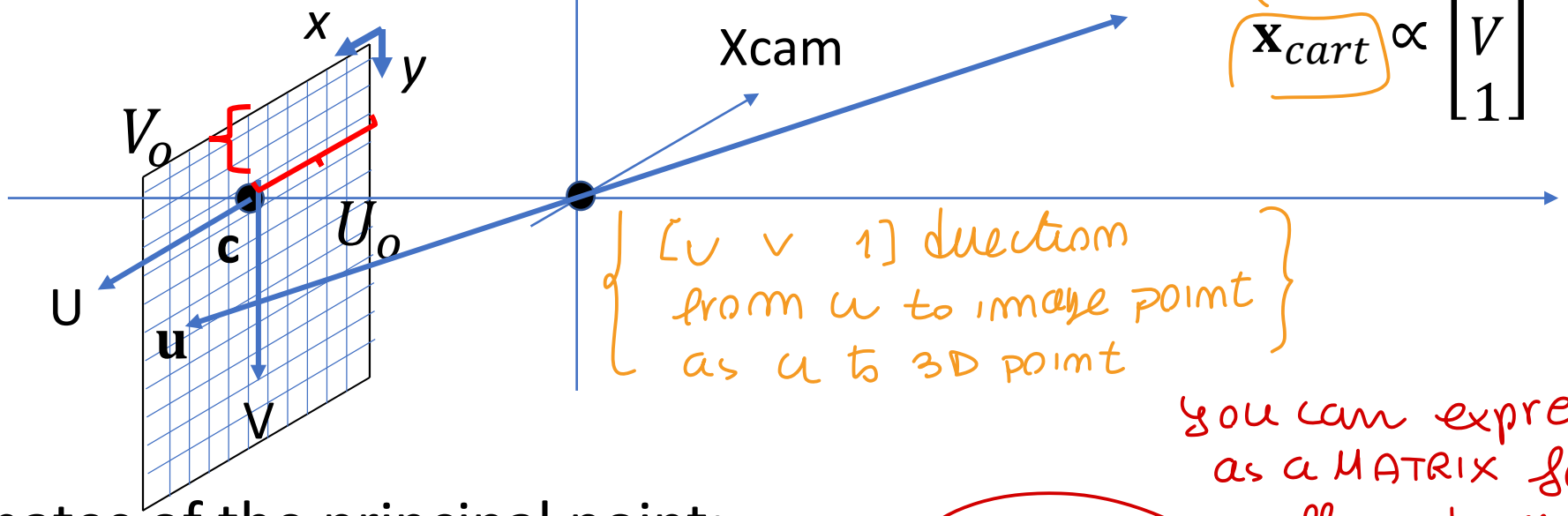
scale translate

$$\mathbf{u} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f_x U + U_o \\ f_y V + V_o \\ 1 \end{bmatrix}$$

homogeneous coord in pixel

↑ cartesian coordinate

homogeneous pixel coordinates of image point: $\mathbf{u} = [x, y, 1]^T$



$\underline{u} \sim$ point cartesian coord in 3D

$\mathbf{x}_{cart} \propto \begin{bmatrix} U \\ V \\ 1 \end{bmatrix}$

direction parameter of vector

[U V 1] direction from u to image point as u to 3D point

geometric in U V

Pixel coordinates of the principal point:

$$\mathbf{c} = (U_o, V_o)$$

«1» is equal to f_x horizontal pixels: offsets along U are multiplied by f_x

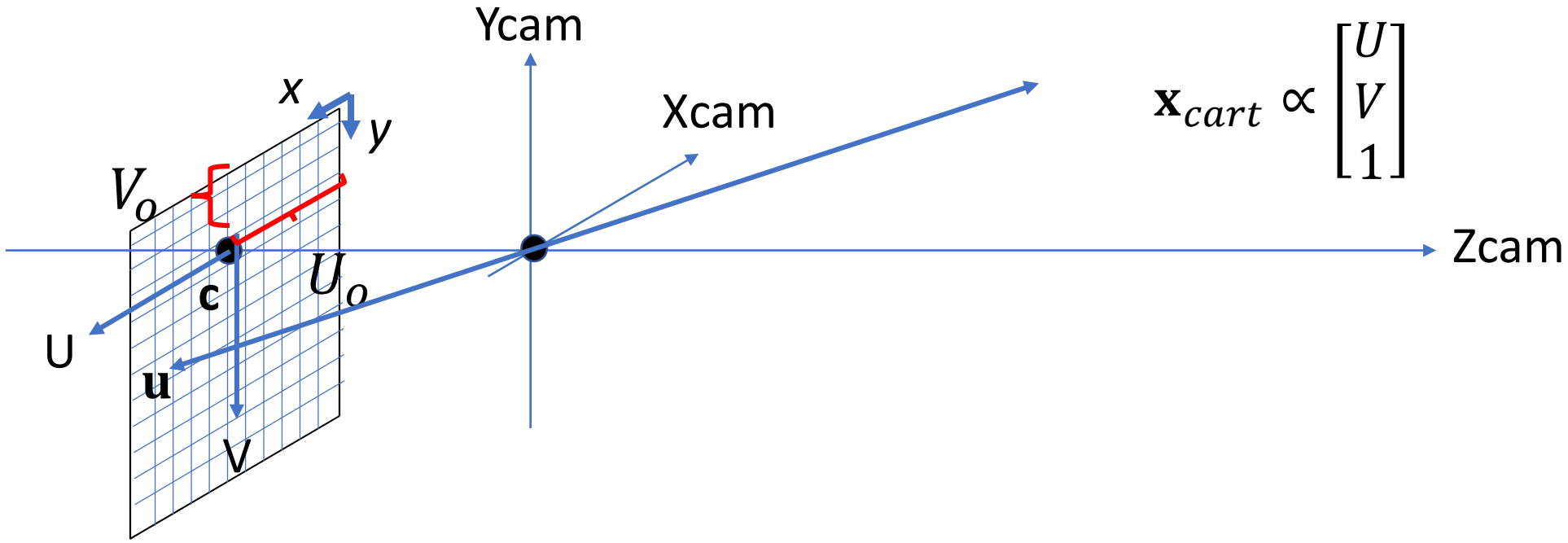
«1» is equal to f_y vertical pixels: offsest along V are mutiplied by f_y

you can express this as a MATRIX form affine transformation between geometric and pixel coordinate

$$\mathbf{u} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f_x U + U_o \\ f_y V + V_o \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & U_o \\ 0 & f_y & V_o \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} U \\ V \\ 1 \end{bmatrix}$$

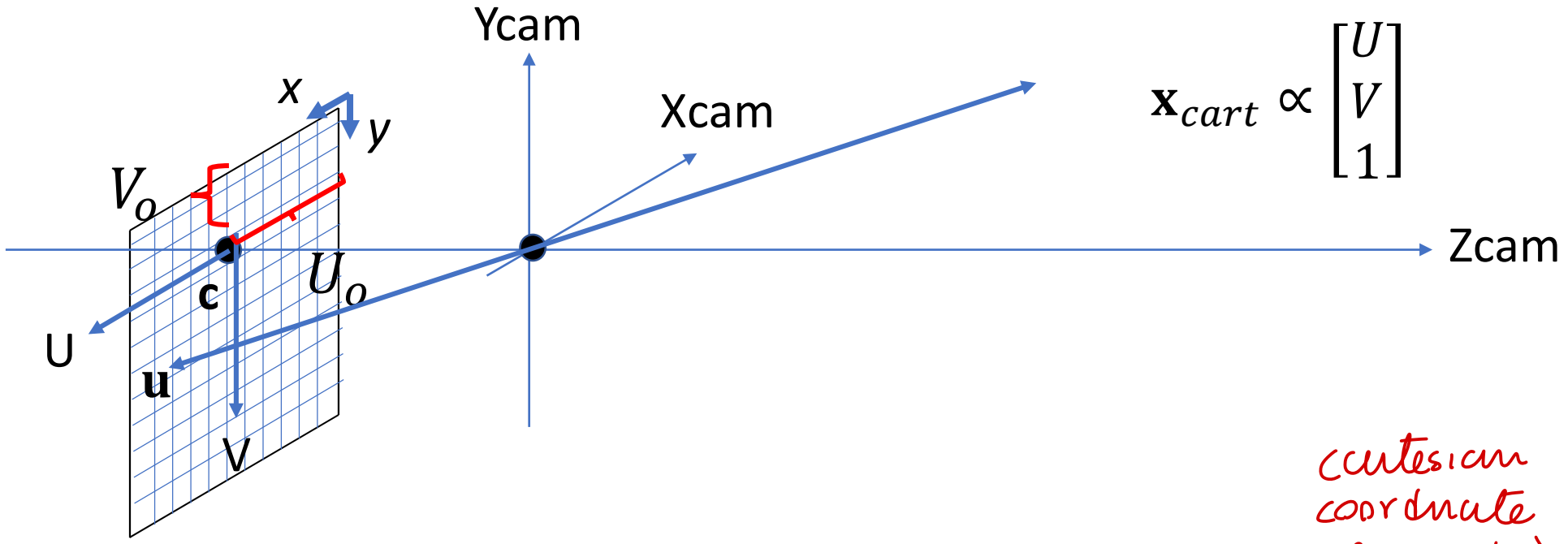
homogeneous coordinate of image on pixel

homogeneous pixel coordinates of image point: $\mathbf{u} = [x, y, 1]^T$



$$\mathbf{u} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f_x U + U_o \\ f_y V + V_o \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & U_o \\ 0 & f_y & V_o \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} U \\ V \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & U_o \\ 0 & f_y & V_o \\ 0 & 0 & 1 \end{bmatrix} \mathbf{x}_{cart}$$

homogeneous pixel coordinates of image point: $\mathbf{u} = [x, y, 1]^T$

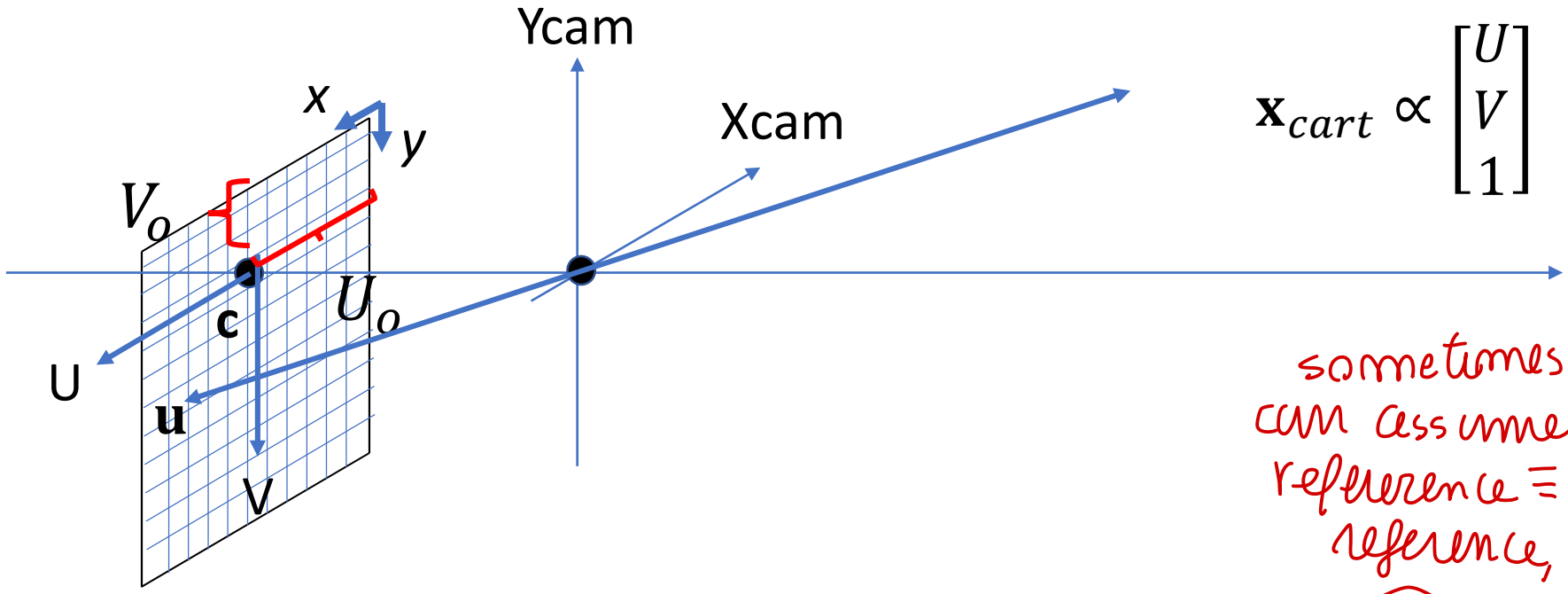


$$\mathbf{u} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f_x U + U_o \\ f_y V + V_o \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & U_o \\ 0 & f_y & V_o \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} U \\ V \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & U_o \\ 0 & f_y & V_o \\ 0 & 0 & 1 \end{bmatrix} \mathbf{x}_{cart} = \mathbf{K}\,\mathbf{x}_{cart}$$

cartesian coordinate of point in 3D wrt camera reference

$\mathbf{K}$

homogeneous pixel coordinates of image point: $\mathbf{u} = [x, y, 1]^T$

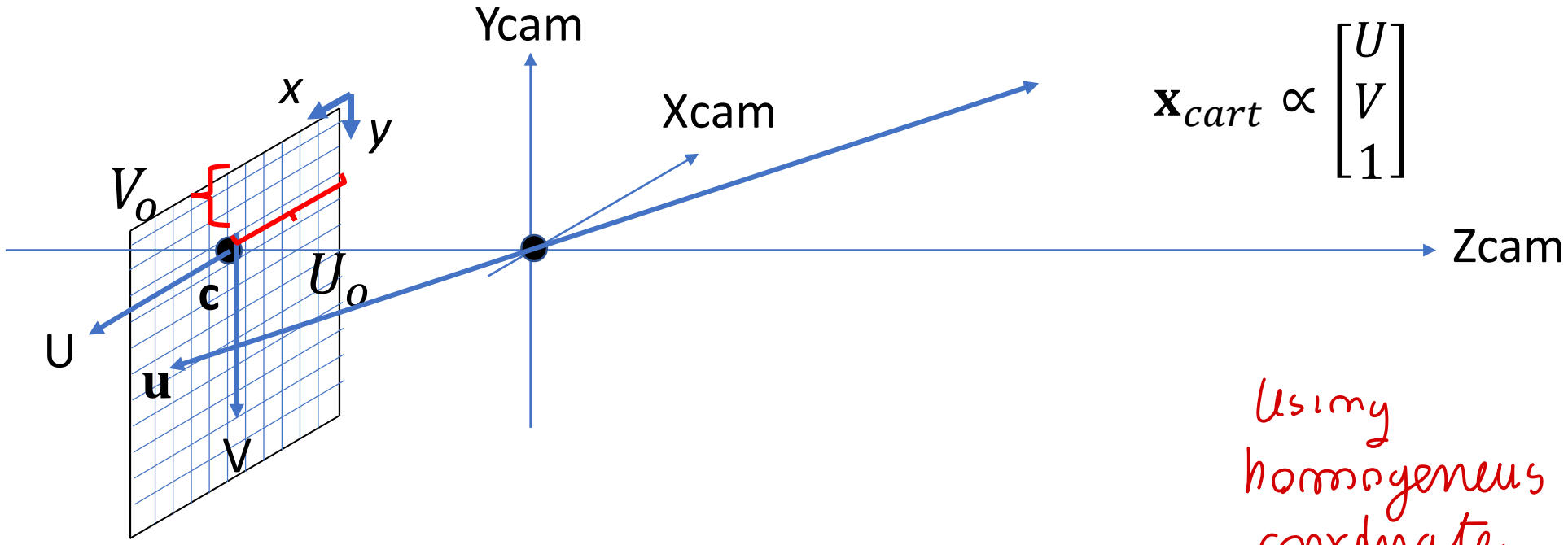


$$\mathbf{x}_{cart} \propto \begin{bmatrix} U \\ V \\ 1 \end{bmatrix}$$

sometimes we can assume WORLD reference = camera reference, NOT always

$$\mathbf{u} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f_x U + U_o \\ f_y V + V_o \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & U_o \\ 0 & f_y & V_o \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} U \\ V \\ 1 \end{bmatrix} = \underbrace{\begin{bmatrix} f_x & 0 & U_o \\ 0 & f_y & V_o \\ 0 & 0 & 1 \end{bmatrix}}_{\mathbf{K}} \mathbf{x}_{cart} = \mathbf{K} \begin{bmatrix} \mathrm{X}_{cam} \\ \mathrm{Y}_{cam} \\ \mathrm{Z}_{cam} \end{bmatrix}$$

homogeneous pixel coordinates of image point: $\mathbf{u} = [x, y, 1]^T$



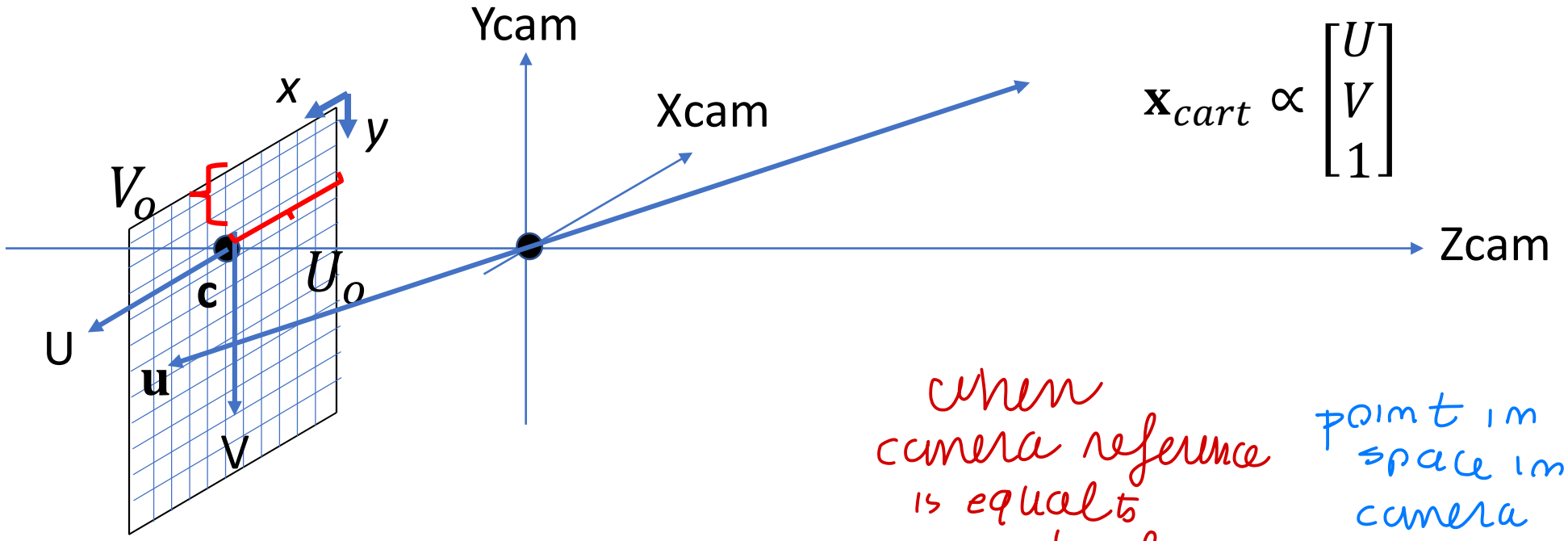
$$\mathbf{x}_{cart} \propto \begin{bmatrix} U \\ V \\ 1 \end{bmatrix}$$

$$\mathbf{u} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \underbrace{\begin{bmatrix} f_x & 0 & U_o \\ 0 & f_y & V_o \\ 0 & 0 & 1 \end{bmatrix}}_{\mathbf{K}} \mathbf{x}_{cart} = \mathbf{K} \begin{bmatrix} \mathrm{X}_{cam} \\ \mathrm{Y}_{cam} \\ \mathrm{Z}_{cam} \end{bmatrix} = \begin{bmatrix} \mathbf{K} & \begin{matrix} 0 \\ 0 \\ 0 \end{matrix} \end{bmatrix} \begin{bmatrix} \mathrm{X}_{cam} \\ \mathrm{Y}_{cam} \\ \mathrm{Z}_{cam} \\ 1 \end{bmatrix}$$

using homogeneous coordinate

add 0_3 to help valid relation

homogeneous pixel coordinates of image point: $\mathbf{u} = [x, y, 1]^T$



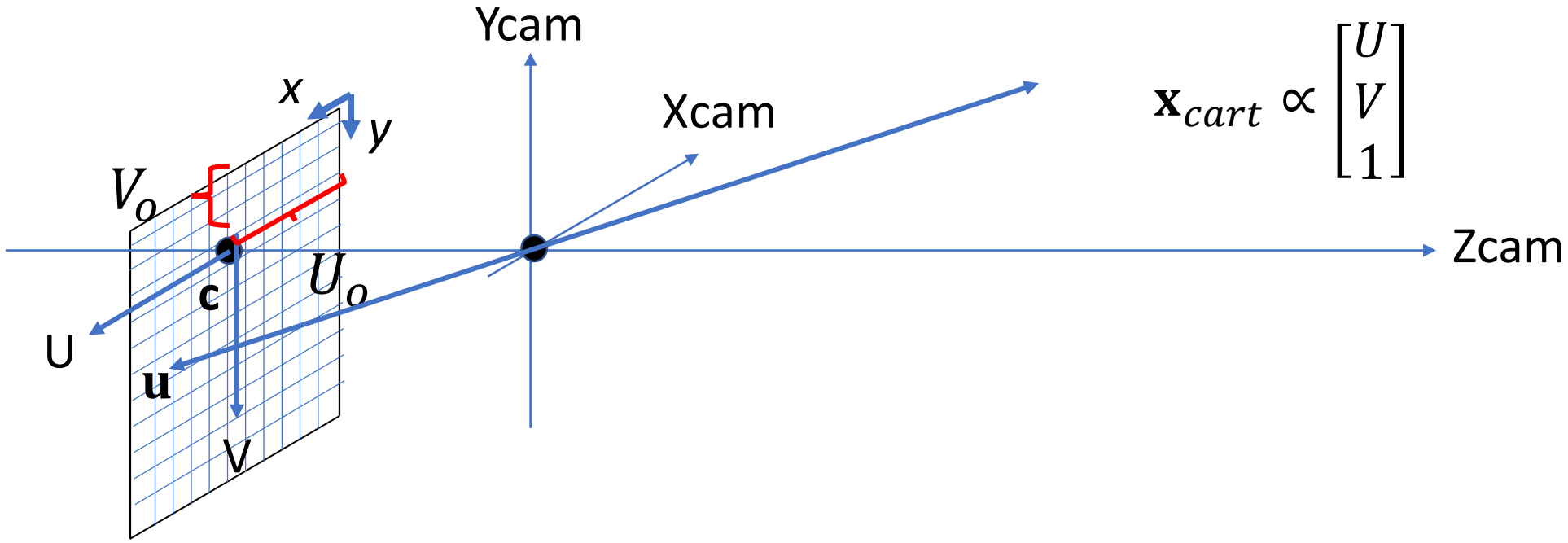
when camera reference is equal to world reference

point in space in camera reference

$$\mathbf{u} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \underbrace{\begin{bmatrix} f_x & 0 & U_o \\ 0 & f_y & V_o \\ 0 & 0 & 1 \end{bmatrix}}_{\mathbf{K}} \mathbf{x}_{cart} = \begin{bmatrix} \mathrm{X}_{cam} \\ \mathrm{Y}_{cam} \\ \mathrm{Z}_{cam} \end{bmatrix} = \begin{bmatrix} \mathbf{K} & \begin{matrix} 0 \\ 0 \\ 0 \end{matrix} \end{bmatrix} \mathbf{X}_{cam} = [\mathbf{K} \quad \mathbf{0}] \mathbf{X}_{cam}$$

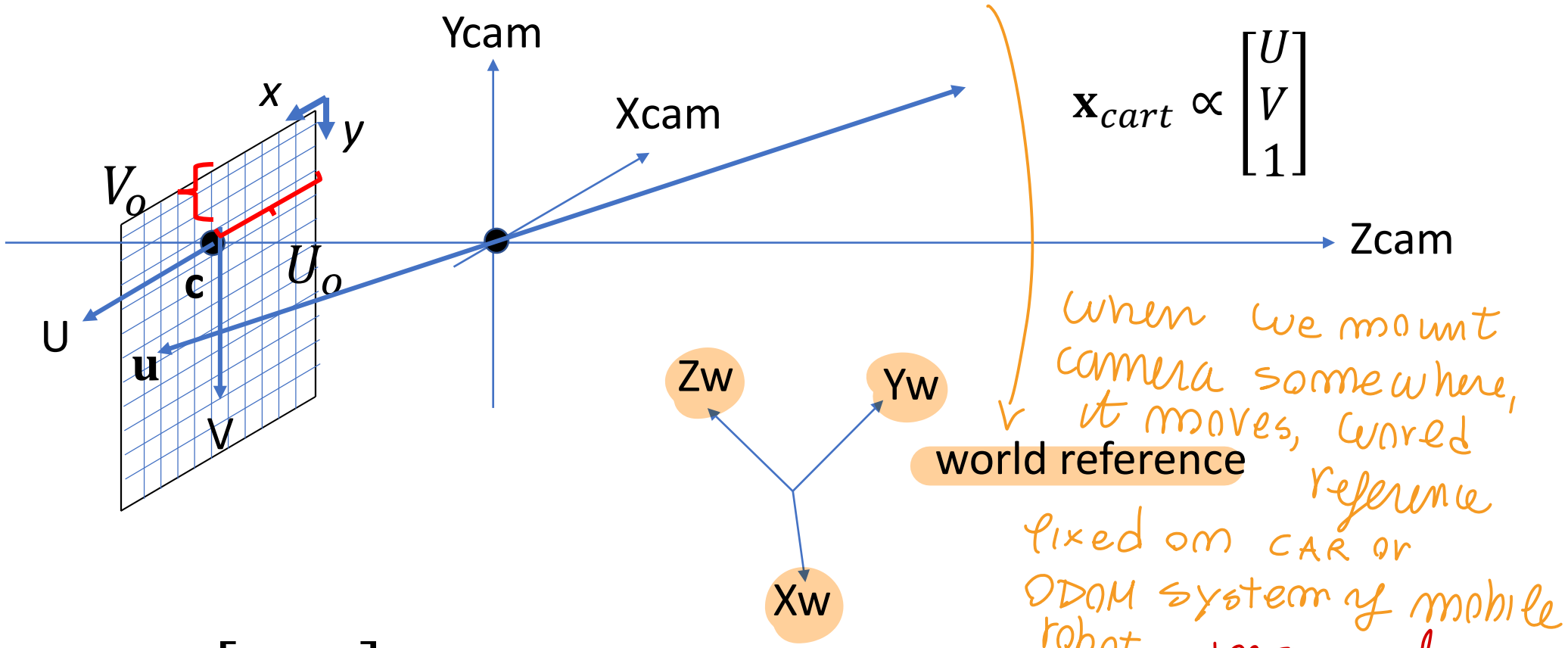
simple projection matrix

homogeneous pixel coordinates of image point: $\mathbf{u} = [x, y, 1]^T$



$$\mathbf{u} = \mathbf{K}\,\mathbf{x}_{cart} = \mathbf{K}\begin{bmatrix} \mathrm{X}_{cam} \\ \mathrm{Y}_{cam} \\ \mathrm{Z}_{cam} \end{bmatrix} = \begin{bmatrix} & & 0 \\ \mathbf{K} & & 0 \\ & & 0 \end{bmatrix}\mathbf{X}_{cam} = [\mathbf{K} \quad \mathbf{0}]\mathbf{X}_{cam} = \mathbf{K}[\mathrm{I} \quad 0]\mathbf{X}_{cam}$$

In general, world reference is ≠ camera reference

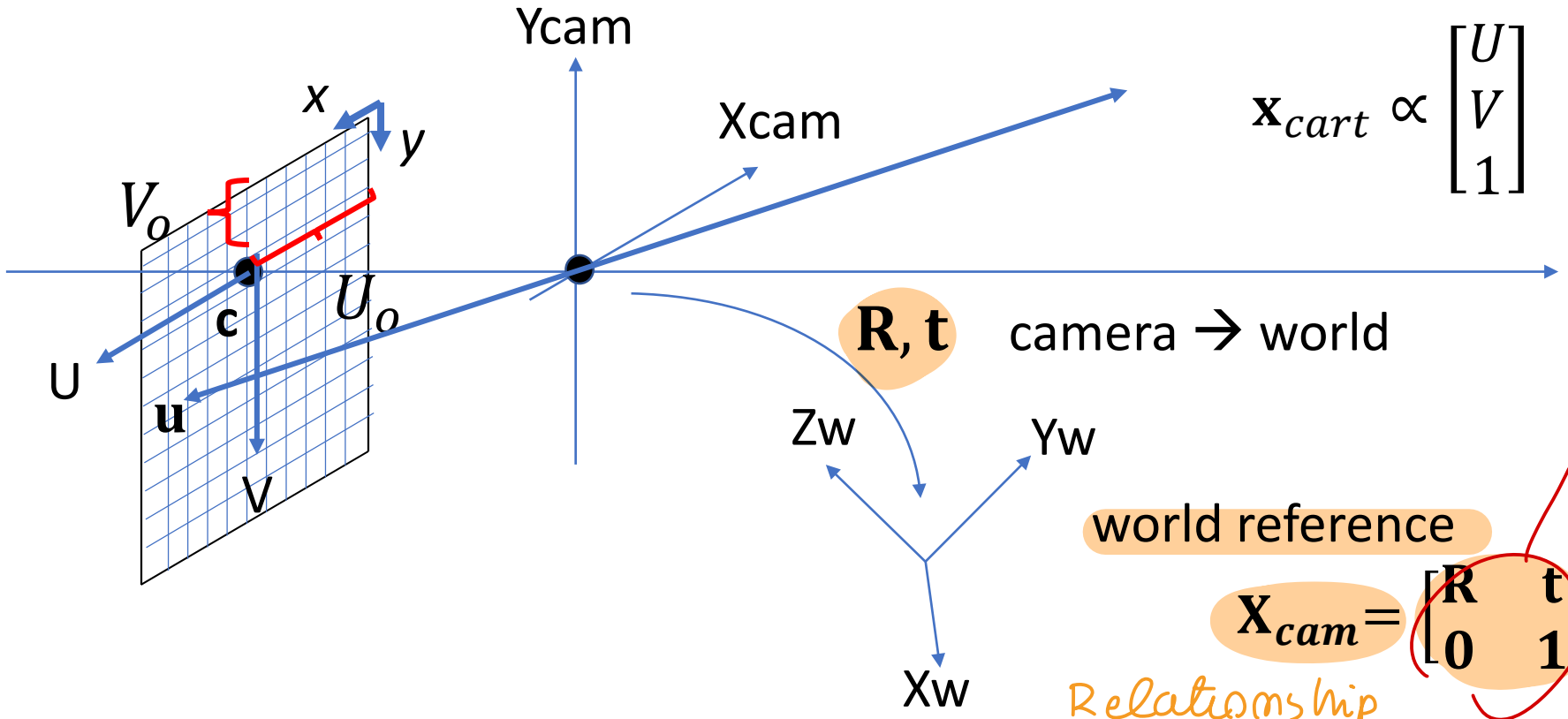


When we mount camera somewhere, it moves, cured reference fixed on CAR or ODOM system of mobile robot... ingenual

$$\mathbf{u} = \mathbf{K}\,\mathbf{x}_{cart} = \mathbf{K}\begin{bmatrix} \mathrm{X}_{cam} \\ \mathrm{Y}_{cam} \\ \mathrm{Z}_{cam} \end{bmatrix} = \begin{bmatrix} \mathbf{K} & \begin{matrix} 0 \\ 0 \\ 0 \end{matrix} \end{bmatrix} \mathbf{X}_{cam} = [\mathbf{K} \quad \mathbf{0}]\mathbf{X}_{cam} = \mathbf{K}[\mathrm{I} \quad 0]\mathbf{X}_{cam}$$

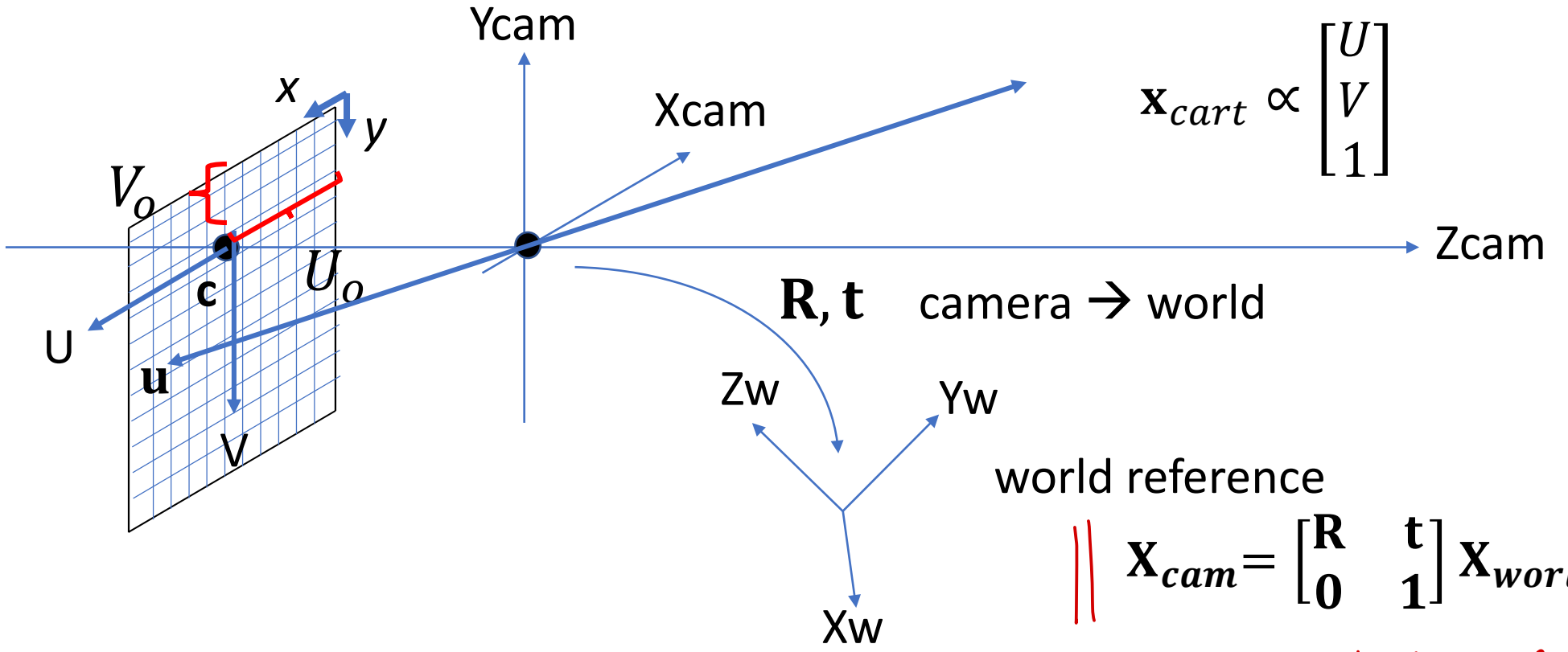
Cured ref different from camera

In general, world reference is ≠ camera reference



$$\mathbf{u} = \mathbf{K}\,\mathbf{x}_{cart} = \mathbf{K}\begin{bmatrix} \mathrm{X}_{cam} \\ \mathrm{Y}_{cam} \\ \mathrm{Z}_{cam} \end{bmatrix} = \begin{bmatrix} \mathbf{K} & \begin{matrix} 0 \\ 0 \\ 0 \end{matrix} \end{bmatrix} \mathbf{X}_{cam} = [\mathbf{K} \quad \mathbf{0}]\mathbf{X}_{cam} = \mathbf{K}[\mathrm{I} \quad 0]\mathbf{X}_{cam}$$

In general, world reference is ≠ camera reference



$$\mathbf{x}_{cart} \propto \begin{bmatrix} U \\ V \\ 1 \end{bmatrix}$$

$\mathbf{R}, \mathbf{t}$ camera → world

world reference

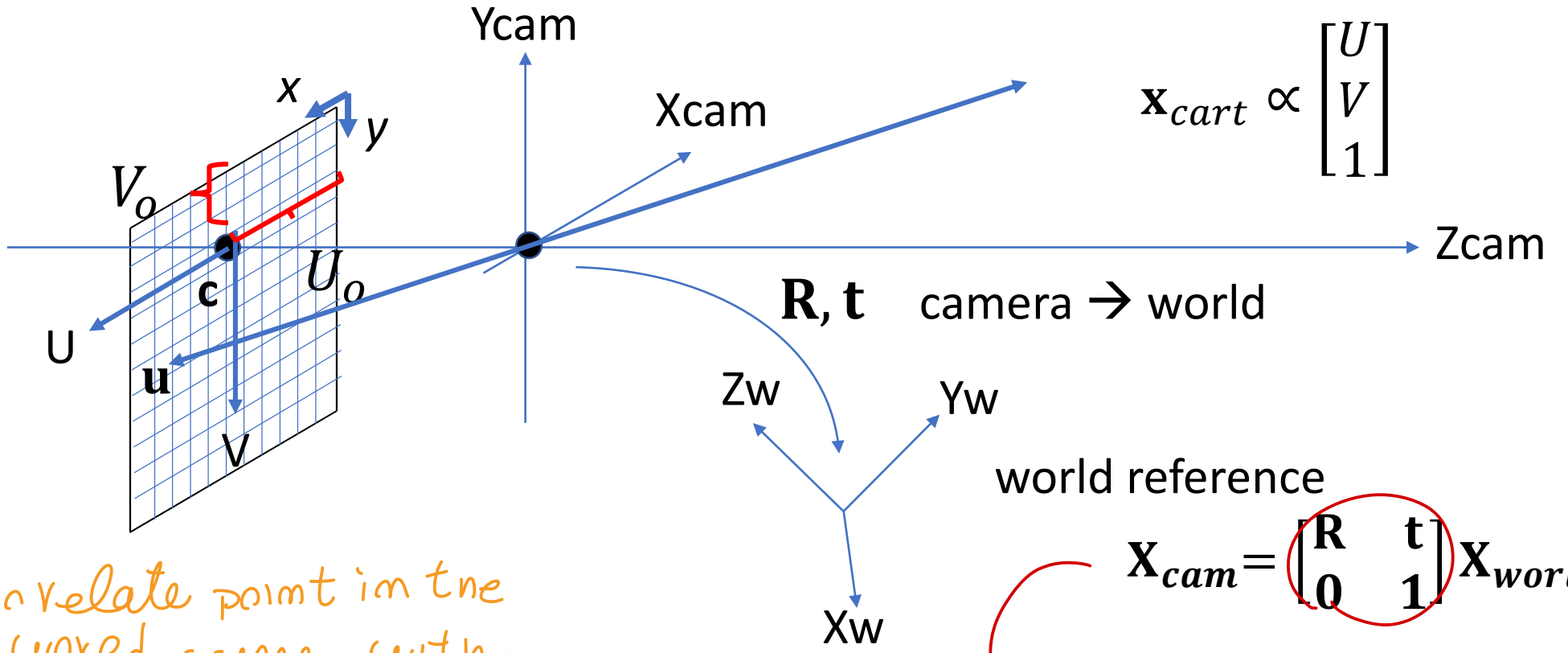
$$\mathbf{X}_{cam} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0} & \mathbf{1} \end{bmatrix} \mathbf{X}_{world}$$

$$\mathbf{u} = \mathbf{K}[\mathrm{I} \quad 0]\mathbf{X}_{cam} = \mathbf{K}[\mathrm{I} \quad 0] \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0} & \mathbf{1} \end{bmatrix} \mathbf{X}_{world}$$

to relate point on image plane with 3D point in world reference

we use rototranslation transformation

In general, world reference is ≠ camera reference



$$\mathbf{x}_{cart} \propto \begin{bmatrix} U \\ V \\ 1 \end{bmatrix}$$

$\mathbf{R}, \mathbf{t}$ camera → world

world reference

$$\mathbf{X}_{cam} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0} & \mathbf{1} \end{bmatrix} \mathbf{X}_{world}$$

to relate point in the world scene with pixel coordinate

$$\mathbf{u} = \mathbf{K}[\mathbf{R} \quad \mathbf{t}]\, \mathbf{X}_{world} = [\mathbf{KR} \quad \mathbf{Kt}]\, \mathbf{X}_{world}$$

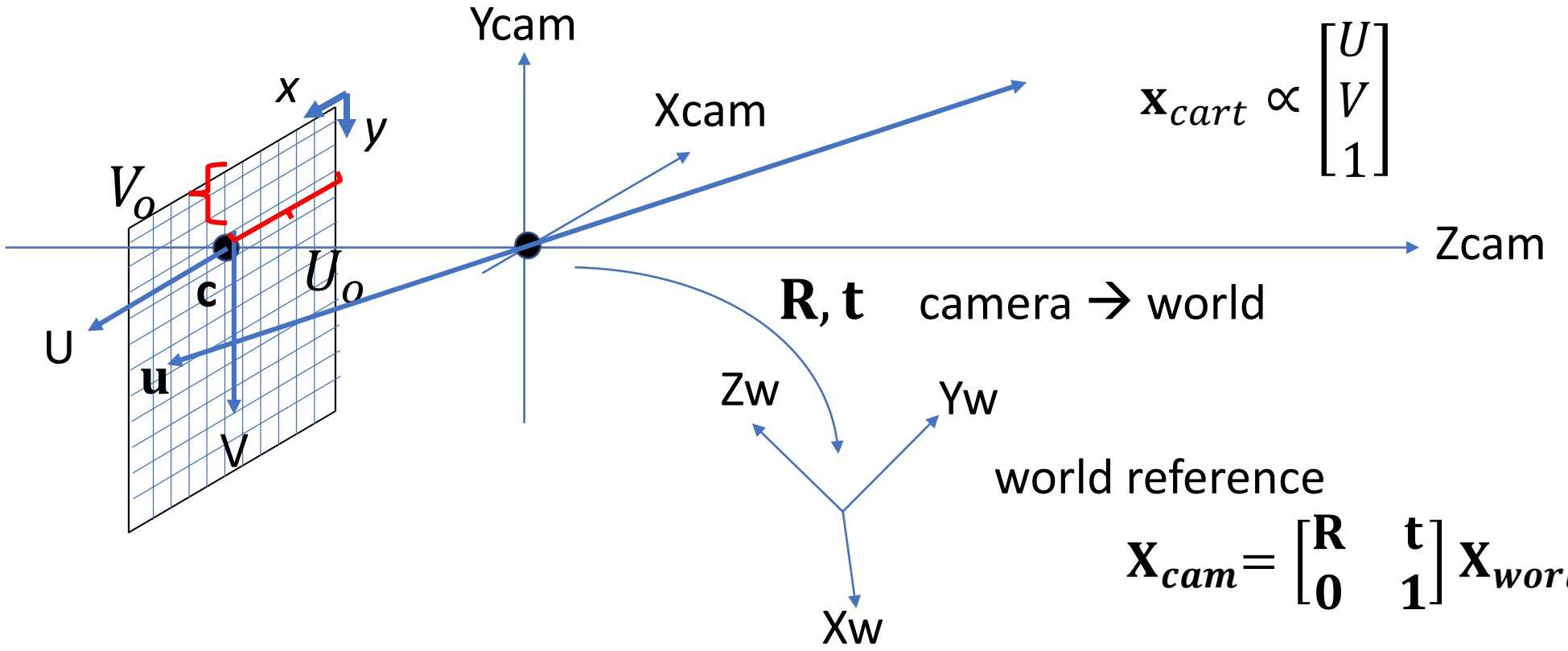
image homogeneus

$u = (KR \quad Kt)\, X_{world}$

the last row disappear!

$[KR \quad kt]$ (3×3) (3×1)

In general, world reference is ≠ camera reference



$$\mathbf{x}_{cart} \propto \begin{bmatrix} U \\ V \\ 1 \end{bmatrix}$$

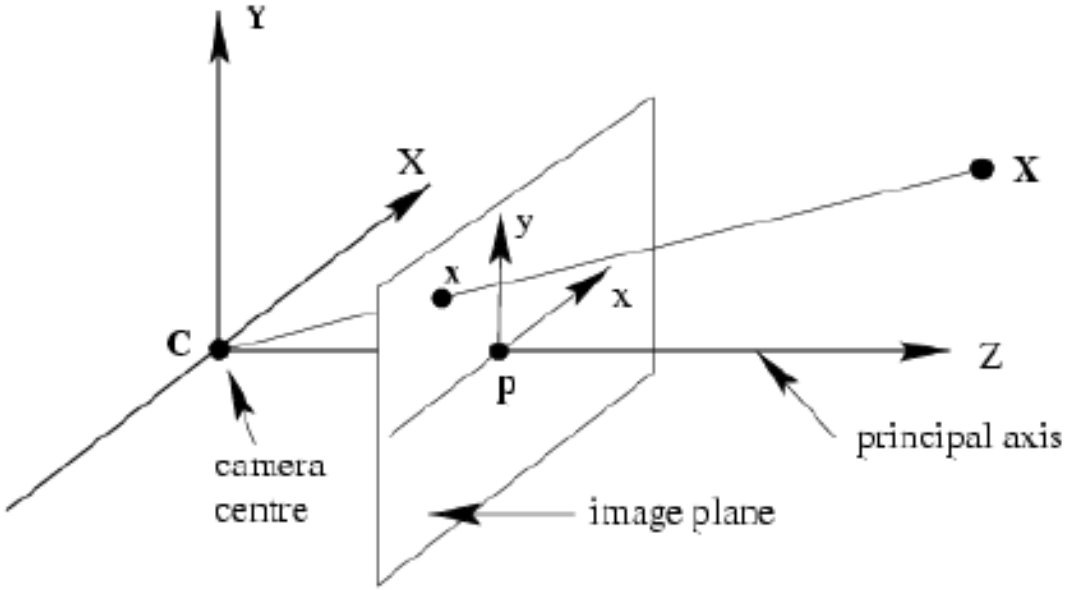
$\mathbf{R}, \mathbf{t}$ camera → world

world reference

$$\mathbf{X}_{cam} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0} & \mathbf{1} \end{bmatrix} \mathbf{X}_{world}$$

$$\mathbf{u} = \mathbf{K}[\mathbf{R} \quad \mathbf{t}]\ \mathbf{X}_{world} = [\mathbf{KR} \quad \mathbf{Kt}]\ \mathbf{X}_{world}$$

remember …



$$\mathbf{X} = \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \mapsto \mathbf{u} = \begin{pmatrix} u \\ v \\ w \end{pmatrix} = \mathbf{P}_{3\times4} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} = \mathbf{P}_{3\times4}\mathbf{X} = \left| \mathbf{M}_{3\times3} \quad \mathbf{m}_{3\times1} \right| \mathbf{X}$$

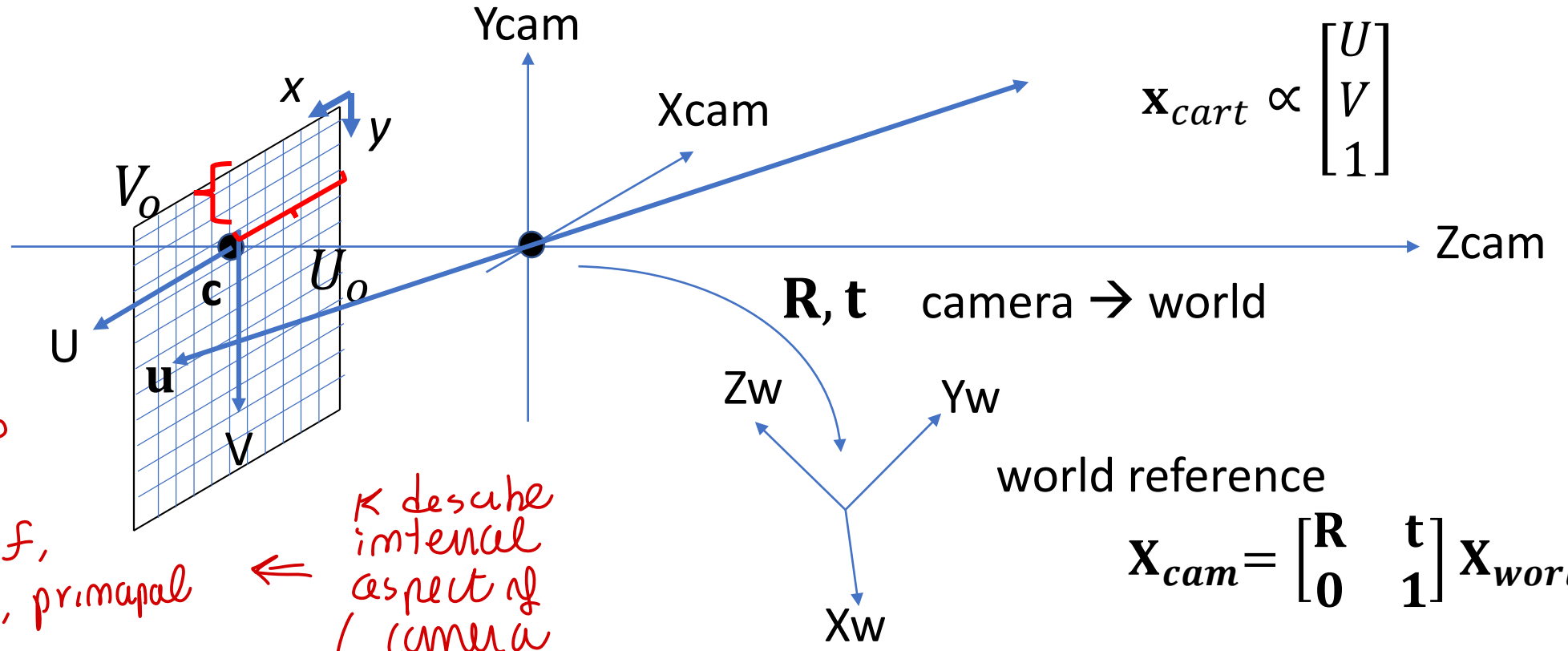
camera projection matrix

invertible

It's same found before

$$\mathbf{M} = \mathbf{K}\mathbf{R}_{cam->world}$$

In general, world reference is ≠ camera reference



$$\mathbf{x}_{cart} \propto \begin{bmatrix} U \\ V \\ 1 \end{bmatrix}$$

$\mathbf{R}, \mathbf{t}$ camera → world

world reference

$$\mathbf{X}_{cam} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0} & \mathbf{1} \end{bmatrix} \mathbf{X}_{world}$$

INTRINSIC PARAMETERS of camera ← like f, f_x, f_y, principal points, ... INTERNAL PARTS

← describe internal aspect of camera

$$\mathbf{u} = \mathbf{K}[\mathbf{R} \quad \mathbf{t}]\,\mathbf{X}_{world} = [\mathbf{KR} \quad \mathbf{Kt}]\,\mathbf{X}_{world}$$

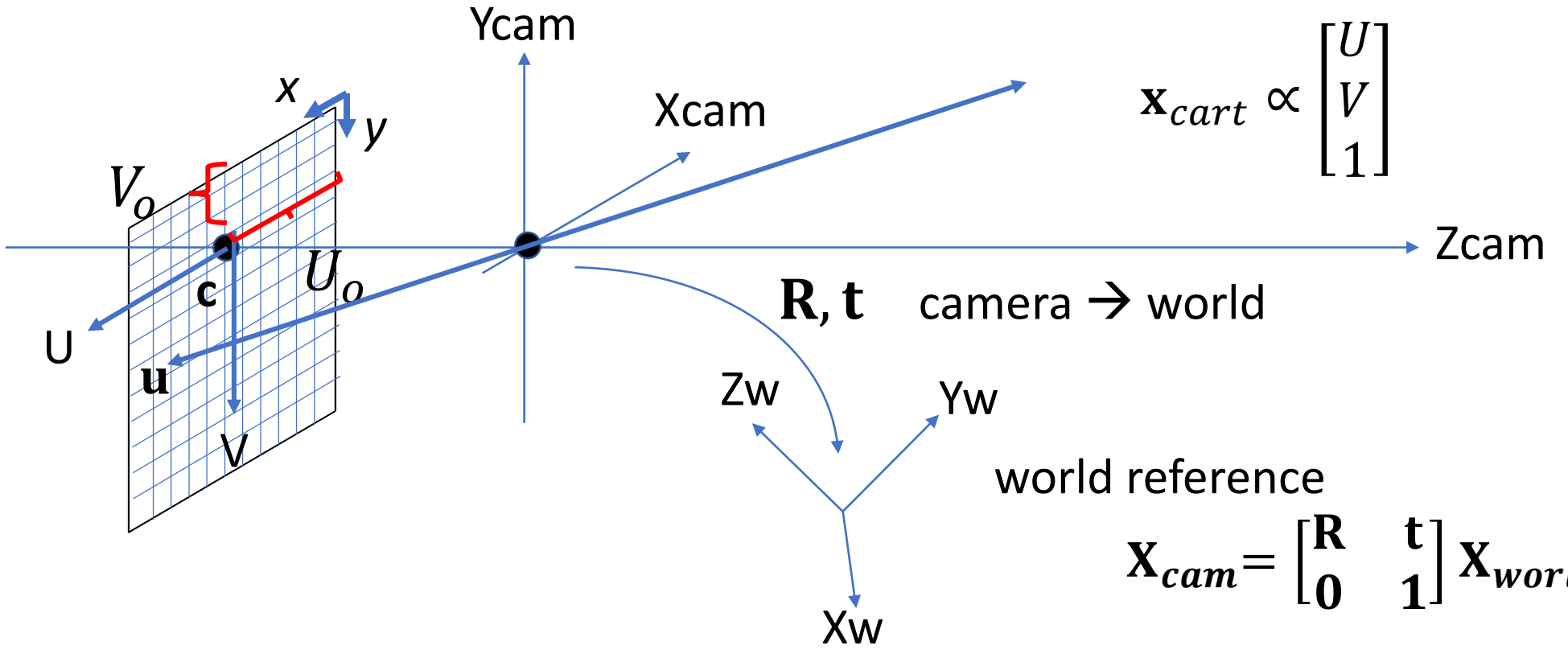
remember $\mathbf{u} = \mathbf{P}\,\mathbf{X}_{world}$

$$= [\mathbf{M} \quad \boldsymbol{m}]\,\mathbf{X}_{world}$$

as name given before!

EXTRINSIC PARAMETERS of camera ← this called information about environment as where is the camera

In general, world reference is ≠ camera reference



$$\mathbf{u} = \mathbf{K}[\mathbf{R} \quad \mathbf{t}]\ \mathbf{X}_{world} = [\mathbf{KR} \quad \mathbf{Kt}]\ \mathbf{X}_{world}$$

remember $\mathbf{u} = \mathbf{P}\ \mathbf{X}_{world} = [\mathbf{M} \quad \mathbf{m}]\ \mathbf{X}_{world}$

INTRINSIC * ROTATION

→ $\mathbf{M = KR}$ and

INTRINSIC * TRANSLATION

$\mathbf{m = Kt}$

$$\mathbf{u} = \mathbf{K}[\mathbf{R} \quad \mathbf{t}]\, \mathbf{X}_{world} = [\mathbf{KR} \quad \mathbf{Kt}]\, \mathbf{X}_{world}$$

but $$\mathbf{u} = \mathbf{P}\, \mathbf{X}_{world} = [\mathbf{M} \quad \mathbf{m}]\, \mathbf{X}_{world}$$

decompose M using as upper triangular * orthogonal R → K, R

for this specific case, this matrix operation allow to decompose it in

→ $\mathbf{M} = \mathbf{KR}$ and $\mathbf{m} = \mathbf{Kt}$

↓ how to decompose?

given **M**, find **K** and **R** by Q-R matrix decomposition of inv(**M**)

$\mathbf{M} = \mathbf{KR}_{cam->world}$

in general difficult to decompose...

being K upper triangular

$K = \begin{bmatrix} * & * & * \\ 0 & * & * \\ 0 & 0 & * \end{bmatrix}$

while R orthogonal (NOT generic matrix)

this Q-R decomposition, decompose as orthogonal * triangular, you consider inverse of M... so you get K^{-1}, R^{-1}

work with $M^{-1} = R^{-1}K^{-1}$ to get $R^{-1}K^{-1}$ by Q-R

$$\mathbf{u} = \mathbf{K}[\mathbf{R} \quad \mathbf{t}]\ \mathbf{X}_{world} = [\mathbf{KR} \quad \mathbf{Kt}]\ \mathbf{X}_{world}$$

but $$\mathbf{u} = \mathbf{P}\ \mathbf{X}_{world} = [\mathbf{M} \quad \mathbf{m}]\ \mathbf{X}_{world}$$

$$\rightarrow \mathbf{M} = \mathbf{KR} \quad \text{and} \quad \mathbf{m} = \mathbf{Kt}$$

$$\mathbf{M} = \mathbf{KR}_{cam->world}$$

given **M**, find **K** and **R** by Q-R matrix decomposition of inv(**M**)

once you have M

from $\mathbf{m} = -\mathbf{Mo}$ → $\mathbf{u} = [\mathbf{M} \quad \mathbf{m}]\ \mathbf{X}_{world} = \mathbf{M}[\mathbf{I} \quad -\boldsymbol{o}]\ \mathbf{X}_{world}$

$$\mathbf{u} = \mathbf{KR}[\mathbf{I} \quad -\boldsymbol{o}]\ \mathbf{X}_{world}$$

camera calibration

Intrinsic camera **calibration**:
estimation of matrix $\mathbf{K}$
- focal distance f_X
- focal distance f_Y
- principal point (U_o, V_o)
- skew factor (in old cameras)
aspect ratio $a = f_X / f_Y$:
ratio between pixel width and height

intrinsic camera parameters don't vary
under camera displacement

Extrinsic camera **calibration**:
estimation of matrix $\mathbf{R}$ and vector $\boldsymbol{t}$
- camera rotation $\mathbf{R}$
- camera translation $\boldsymbol{t}$

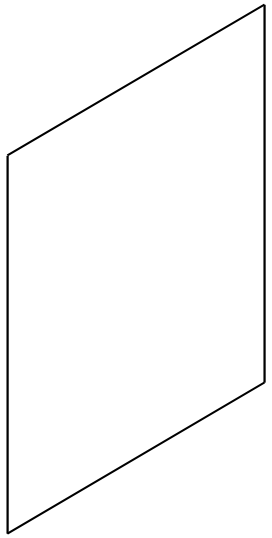
extrinsic camera parameters vary
under camera displacement

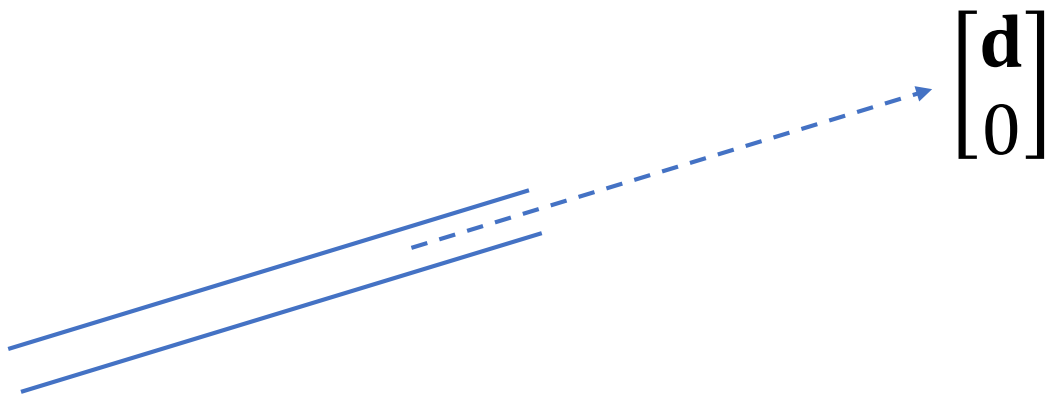
$$x = x_o + (x_o - c_x)(K_1 r^2 + K_2 r^4 + \ldots)$$
$$y = y_o + (y_o - c_y)(K_1 r^2 + K_2 r^4 + \ldots)$$

$$r = (x_o - c_x)^2 + (y_o - c_y)^2 \ .$$

vanishing point of a direction

parallel lines concur at the point at the infinity of their common direction $\mathbf{d}$





vanishing points

V image of the point at the ∞ along direction **d**

$$\mathbf{u}_V = \left| \mathbf{M} \,\middle|\, \mathbf{m} \right| \cdot \left| \begin{array}{c} \mathbf{d} \\ \hdashline 0 \end{array} \right| = \mathbf{M} \cdot \mathbf{d}$$

vanishing points

V image of the point at the ∞ along direction **d**

$$\mathbf{u_V} = \left| \mathbf{M} \,\middle|\, \mathbf{m} \right| \cdot \left| \begin{array}{c} \mathbf{d} \\ \hdashline 0 \end{array} \right| = \mathbf{M} \cdot \mathbf{d}$$

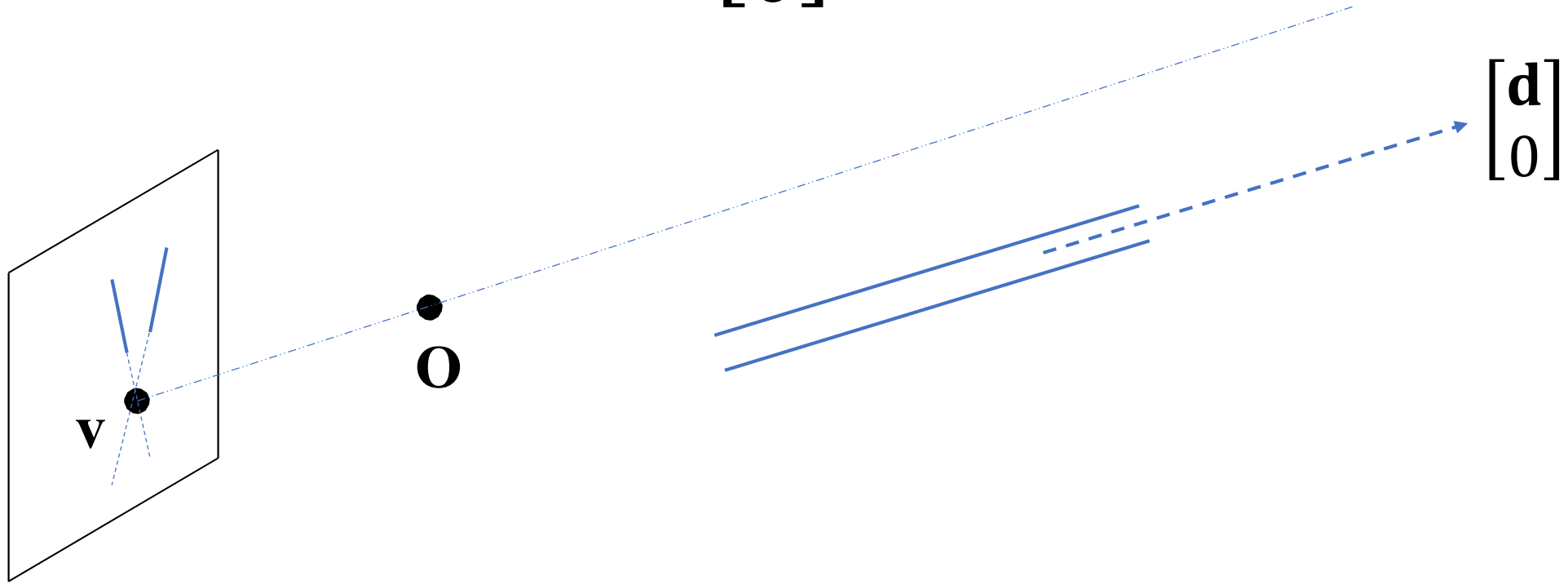
Remember: the direction of the backprojection of image point $\mathbf{V}$ (viewing ray associated to $\mathbf{V}$) is $\mathbf{M^{-1}u_V} = \mathbf{M^{-1}\,Md} = \mathbf{d}$

→

Vanishing Point Theorem:

The viewing ray associated to the vanishing point $\mathbf{V}$ of a direction $\mathbf{d}$ is parallel to $\mathbf{d}$

The backprojection (viewing ray) of vanishing point $\mathbf{v}$ goes through point at the infinity $\begin{bmatrix}\mathbf{d}\\0\end{bmatrix}$ (since $\mathbf{v}$ is its image)



→ the viewing ray of $\mathbf{v}$ is parallel to the direction $\mathbf{d}$

mobile robot navigation

vehicle driving

If $\mathbf{M}$ is known, find $\mathbf{V}$ and follow the direction of the corridor:

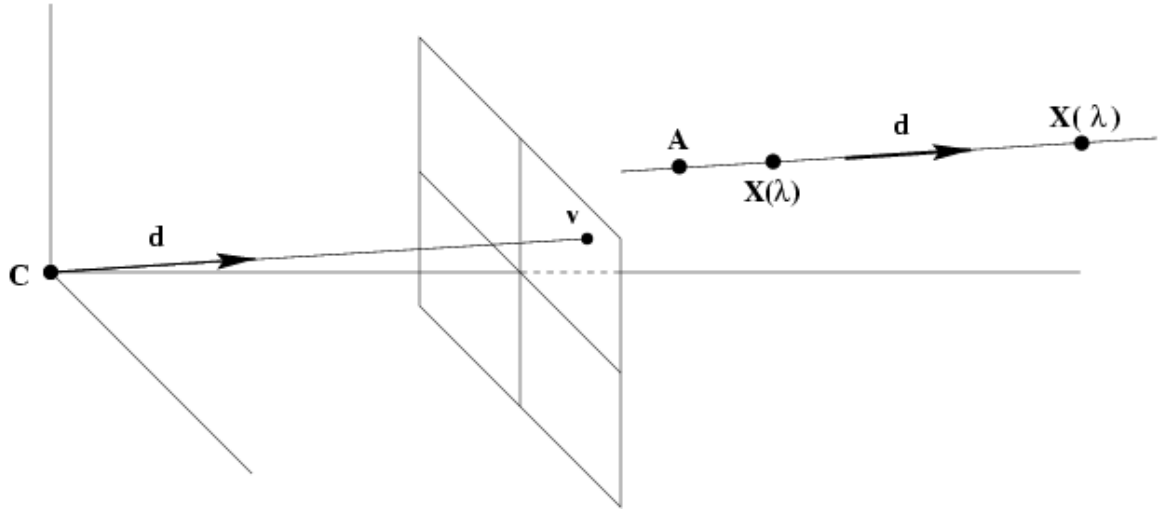
$$\mathbf{d} = \mathbf{M}^{-1}\mathbf{V}$$

Vanishing points (d measured wrt camera reference)

$$x(\lambda) = PX(\lambda) = PA + \lambda PD = a + \lambda Kd$$

$$v = \lim_{\lambda \to \infty} x(\lambda) = \lim_{\lambda \to \infty} (a + \lambda Kd) = Kd$$

$$v = PX_{\infty} = Kd$$



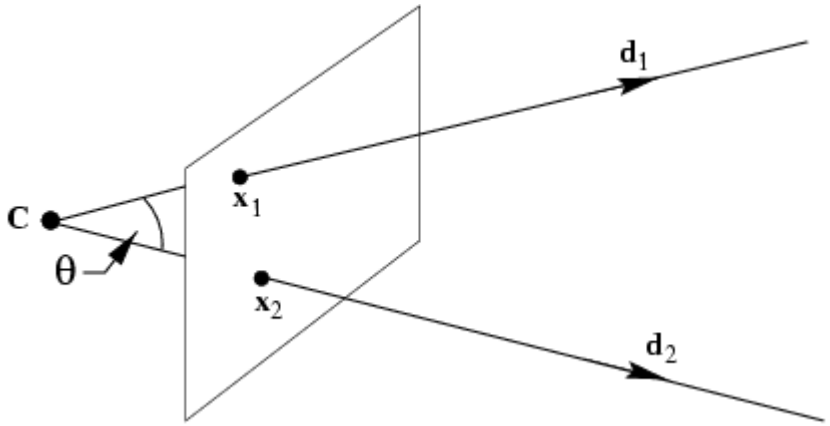
Vanishing points (d measured wrt world reference)

$$v = PX_{\infty} = KRd_w$$

What does INTRINSIC calibration give?

$$x = K[I \mid 0]\begin{bmatrix} d \\ 0 \end{bmatrix}$$

$$d = K^{-1}x$$



$$\cos\theta = \frac{d_1^{\mathrm{T}} d_2}{\sqrt{\left(d_1^{\mathrm{T}} d_1\right)\left(d_2^{\mathrm{T}} d_2\right)}} = \frac{x_1^{\mathrm{T}} (K^{-\mathrm{T}} K^{-1}) x_2}{\sqrt{\left(x_1^{\mathrm{T}} (K^{-\mathrm{T}} K^{-1}) x_1\right)\left(x_2^{\mathrm{T}} (K^{-\mathrm{T}} K^{-1}) x_2\right)}}$$

An image l defines a plane through the camera center with normal $n=K^{T}l$ measured in the camera's Euclidean frame

→ Relative position of viewing rays associated to different image points

The image ω of the absolute conic Ω_∞ is **independent** of $\mathbf{R}$ and $\mathbf{t}$: it only depends on intrinsic parameters

$$x = PX_\infty = KR[I \mid -\tilde{C}]\begin{pmatrix} d \\ 0 \end{pmatrix} = KRd$$

mapping between π_∞ to an image is given by the planar homogaphy u= Hd, with $H = \mathbf{M} = \mathbf{K}\,\mathbf{R}$

image of the absolute conic (IAC) ω $= H^{-T}\Omega_\infty H^{-1} = K^{-T}RI_3R^{-1}K^{-1}$

$$\omega = \left(KK^T\right)^{-1} = K^{-T}K^{-1} \qquad \left(C \mapsto H^{-T}CH^{-1}\right)$$

(i) IAC depends only on intrinsics
(ii) angle between two rays
(iii) DIAC=ω^*=KK^T
(iv) $\omega \Leftrightarrow$ K (cholesky factorisation)
(v) IAC contains image of circular points

$$\cos\theta = \frac{x_1^T \omega x_2}{\sqrt{\left(x_1^T \omega x_1\right)\left(x_2^T \omega x_2\right)}}$$

summary

homography from $\boldsymbol{\pi_\infty}$ to image plane: $\mathbf{H} = \mathbf{M} = \mathbf{K}\,\mathbf{R}$

→ vanishing points = image of points at the ∞: $\mathbf{v}_d = \mathbf{H}\,\mathbf{d} = \mathbf{K}\mathbf{R}\,\mathbf{d}$

image of the absolute conic $\boldsymbol{\omega} = \mathbf{H}^{-\mathrm{T}}\Omega_\infty\mathbf{H}^{-1} = \mathbf{K}^{-\mathrm{T}}\mathbf{R}^{-\mathrm{T}}\mathrm{I}_3\mathbf{R}^{-1}\mathbf{K}^{-1} = \mathbf{K}^{-\mathrm{T}}\mathbf{K}^{-1}$

→ IAC $\boldsymbol{\omega}$ independent of $\mathbf{R}$ and t; it only depends on intrinsic $\mathbf{K}$

from IAC $\boldsymbol{\omega}$ to calibration matrix $\mathbf{K}$: Cholesky factorisation of DIAC $\omega^{-1} = \mathbf{K}\mathbf{K}^{\mathrm{T}}$

absolute conic $\Omega_\infty = \cup_\pi\{\mathbf{I}_\pi, \mathbf{J}_\pi\}$ made of circular points

→ IAC $\boldsymbol{\omega} = \cup_\pi\{\mathbf{I'}_\pi, \mathbf{J'}_\pi\}$ made of imaged circular points $\mathbf{I'}_\pi = \mathbf{H}\mathbf{I}_\pi$, $\mathbf{J'}_\pi = \mathbf{H}\mathbf{J}_\pi$

constraint on $\boldsymbol{\omega}$ from known angle btw. two directions, and their vanishing points

$$\cos\theta = \frac{\mathbf{v_1}^{\mathrm{T}}\omega\mathbf{v_2}}{\sqrt{(\mathbf{v_1}^{\mathrm{T}}\omega\mathbf{v_1})(\mathbf{v_2}^{\mathrm{T}}\omega\mathbf{v_2})}}$$

Scenario n. 1

Known: vanishing points and angle θ between directions

$$\cos\theta = \frac{\mathbf{d_1}^{\mathrm{T}}\mathbf{d_2}}{\sqrt{(\mathbf{d_1}^{\mathrm{T}}\mathbf{d_1})(\mathbf{d_2}^{\mathrm{T}}\mathbf{d_2})}} = \frac{\mathbf{v_1}^{\mathrm{T}}\omega\mathbf{v_2}}{\sqrt{(\mathbf{v_1}^{\mathrm{T}}\omega\mathbf{v_1})(\mathbf{v_2}^{\mathrm{T}}\omega\mathbf{v_2})}}$$

→

constraint on the IAC ω

(linear if directions are orthogonal)

Scenario n. 2

Known: vanishing points and IAC ω

$$\cos\,\theta = \frac{\mathbf{d_1}^{\mathrm{T}}\mathbf{d_2}}{\sqrt{(\mathbf{d_1}^{\mathrm{T}}\mathbf{d_1})(\mathbf{d_2}^{\mathrm{T}}\mathbf{d_2})}} = \frac{\mathbf{v_1}^{\mathrm{T}}\omega\mathbf{v_2}}{\sqrt{(\mathbf{v_1}^{\mathrm{T}}\omega\mathbf{v_1})(\mathbf{v_2}^{\mathrm{T}}\omega\mathbf{v_2})}}$$

→

compute angle θ between directions

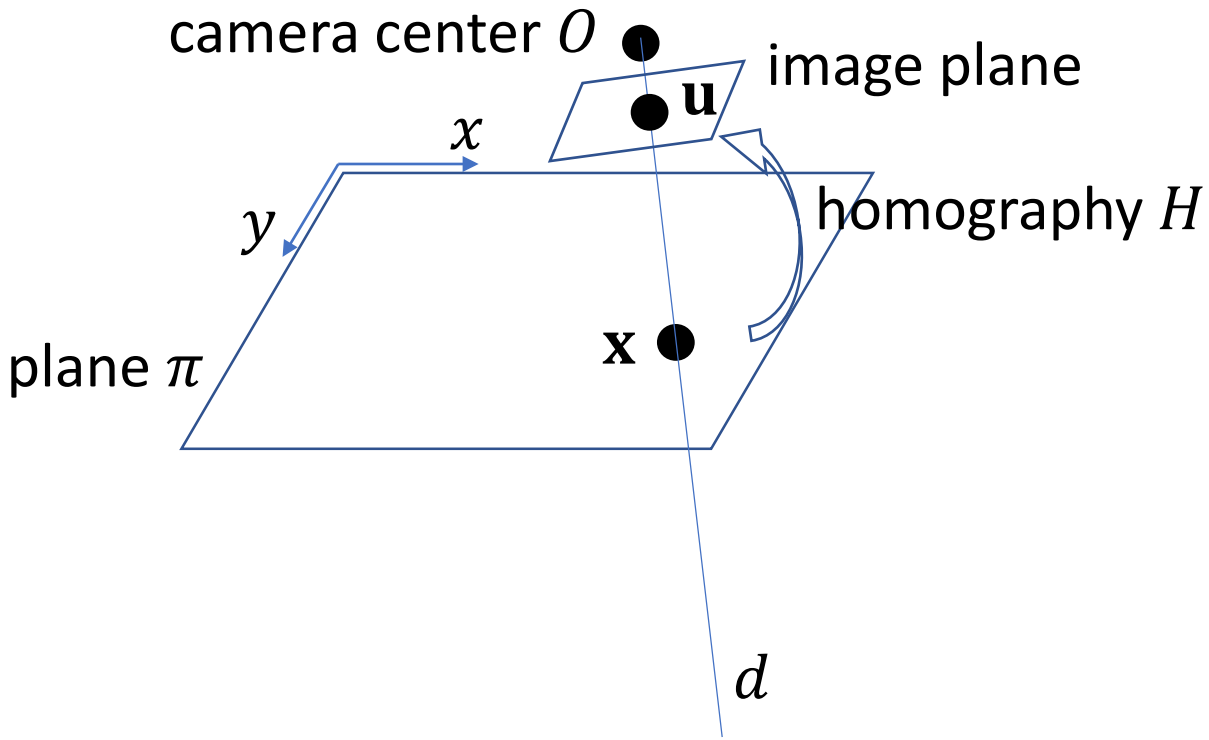
→ reconstruction of the shape

Camera calibration
from a single image of a known planar shape and known camera center position

Calibration from known plane π and camera center O

Let us refer the coordinates to a reference attached to a knwon plane π : a generic point on this plane has homogeneous coordinates

$\mathbf{x} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$, while $O = \begin{bmatrix} \mathrm{x}_o \\ \mathrm{y}_o \\ \mathrm{z}_o \end{bmatrix}$ are the known cartesian coordinates of the camera center,



Calibration from known plane π and camera center O

1) Estimate homography H from known points on the plane π and their images

2) Call M_o the matrix relating any point $\mathbf{x}$ on π to the direction $\mathbf{d}$ of a ray from O to $\mathbf{x}$:

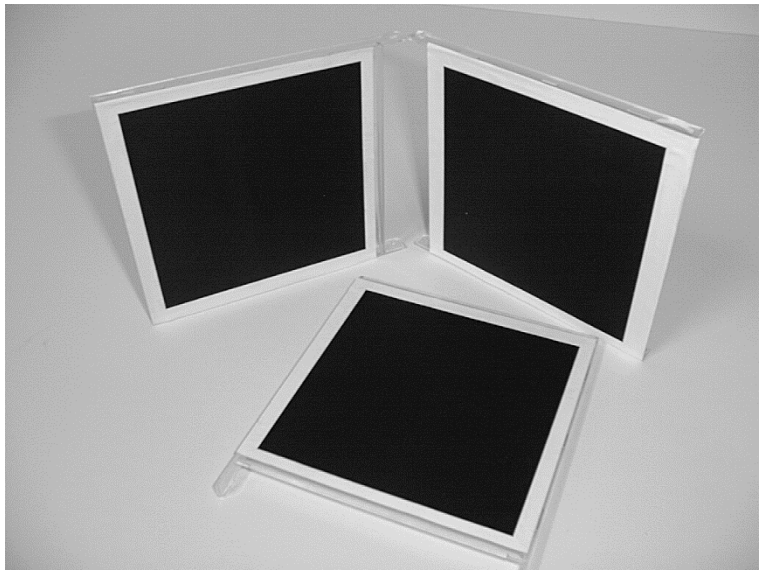
$$\mathbf{d} = \begin{bmatrix} x - \mathrm{x}_o \\ y - \mathrm{y}_o \\ -\mathrm{z}_o \end{bmatrix} = \begin{bmatrix} 1 & 0 & -\mathrm{x}_o \\ 0 & 1 & -\mathrm{y}_o \\ 0 & 0 & -\mathrm{z}_o \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \rightarrow {M_o}^{-1} = \begin{bmatrix} 1 & 0 & -\mathrm{x}_o \\ 0 & 1 & -\mathrm{y}_o \\ 0 & 0 & -\mathrm{z}_o \end{bmatrix}$$

3) Compute the matrix M relating any image point $\mathbf{u}$ to the direction $\mathbf{d}$ of its viewing ray from $\mathbf{u} = H\mathbf{x}$, is $\mathbf{d} = {M_o}^{-1}\mathbf{x} = {M_o}^{-1}H^{-1}\mathbf{u} \rightarrow M^{-1} = {M_o}^{-1}H^{-1}$

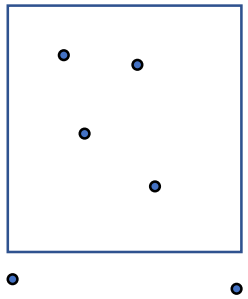
4) Q-R decompose matrix M^{-1} as $M^{-1} = R^{-1}K^{-1}$, where K is the camera intrinsic calibration matrix and R^{-1} is the rotation matrix from the world (reference attached to π) to the camera

Camera calibration
from images of known planar shapes
(Zhang method)

A simple calibration device (Zhang 2000)

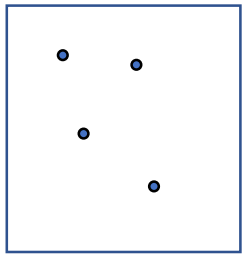


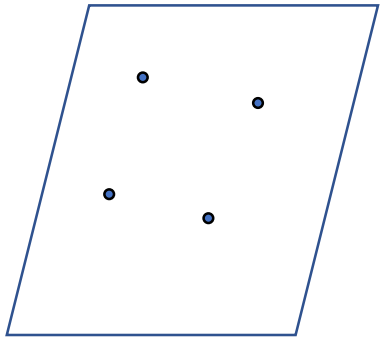
(i) compute H for each square
(corners □ (0,0),(1,0),(0,1),(1,1))
(ii) compute the imaged circular points $H(1,\pm i,0)^T$
(iii) fit a conic to 6 imaged circular points
(iv) compute K from ω through Cholesky factorization



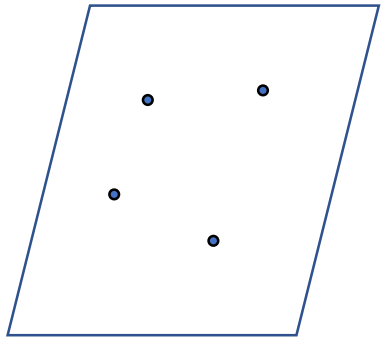
I=[1,i,0] J=[1,-i,0]

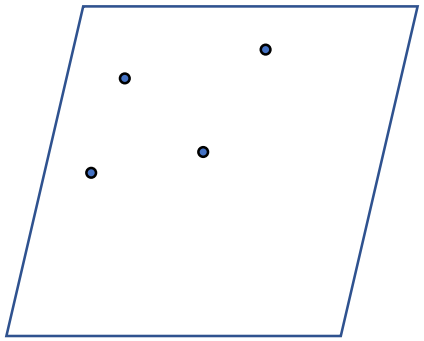
known planar scene

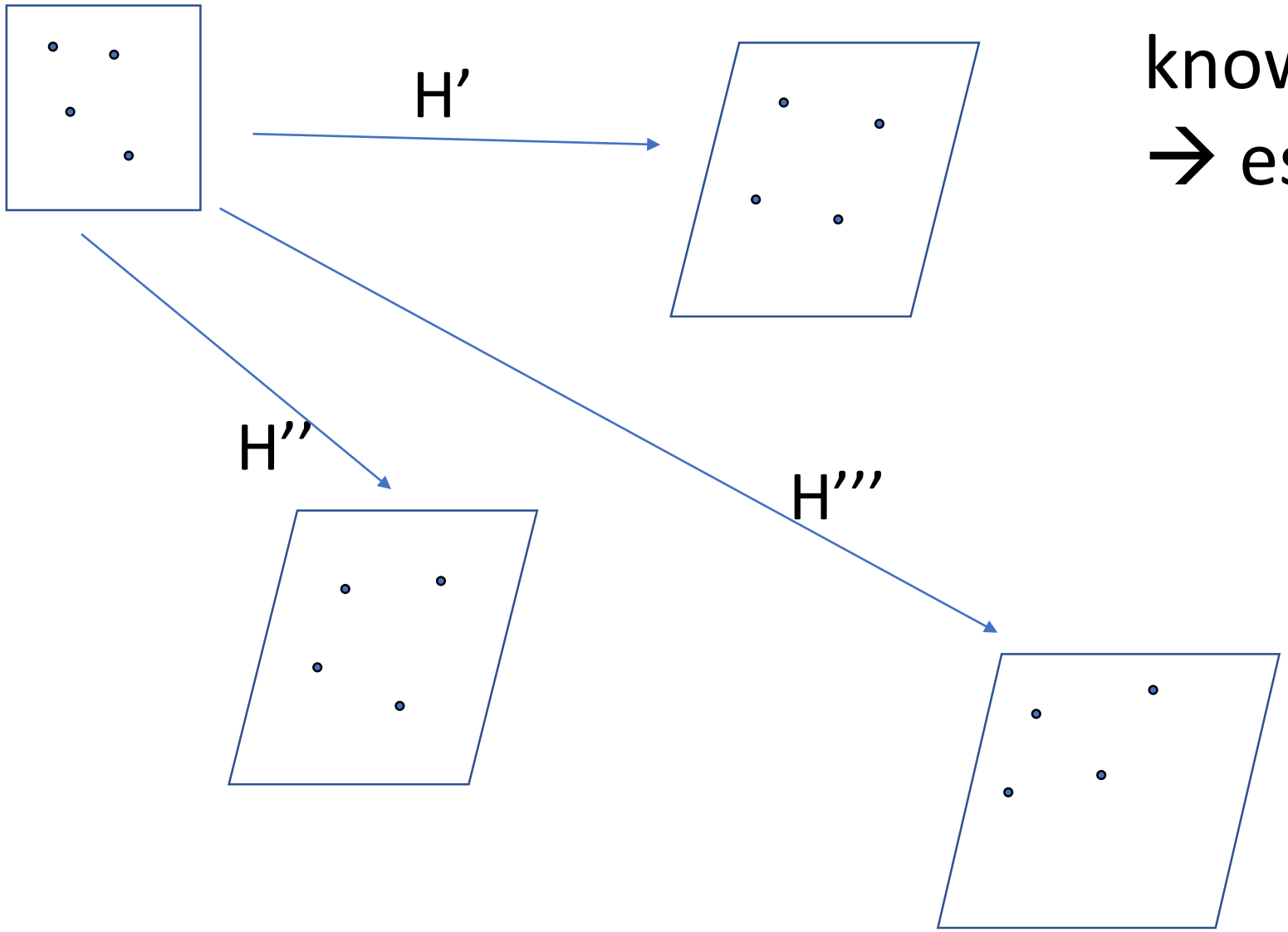




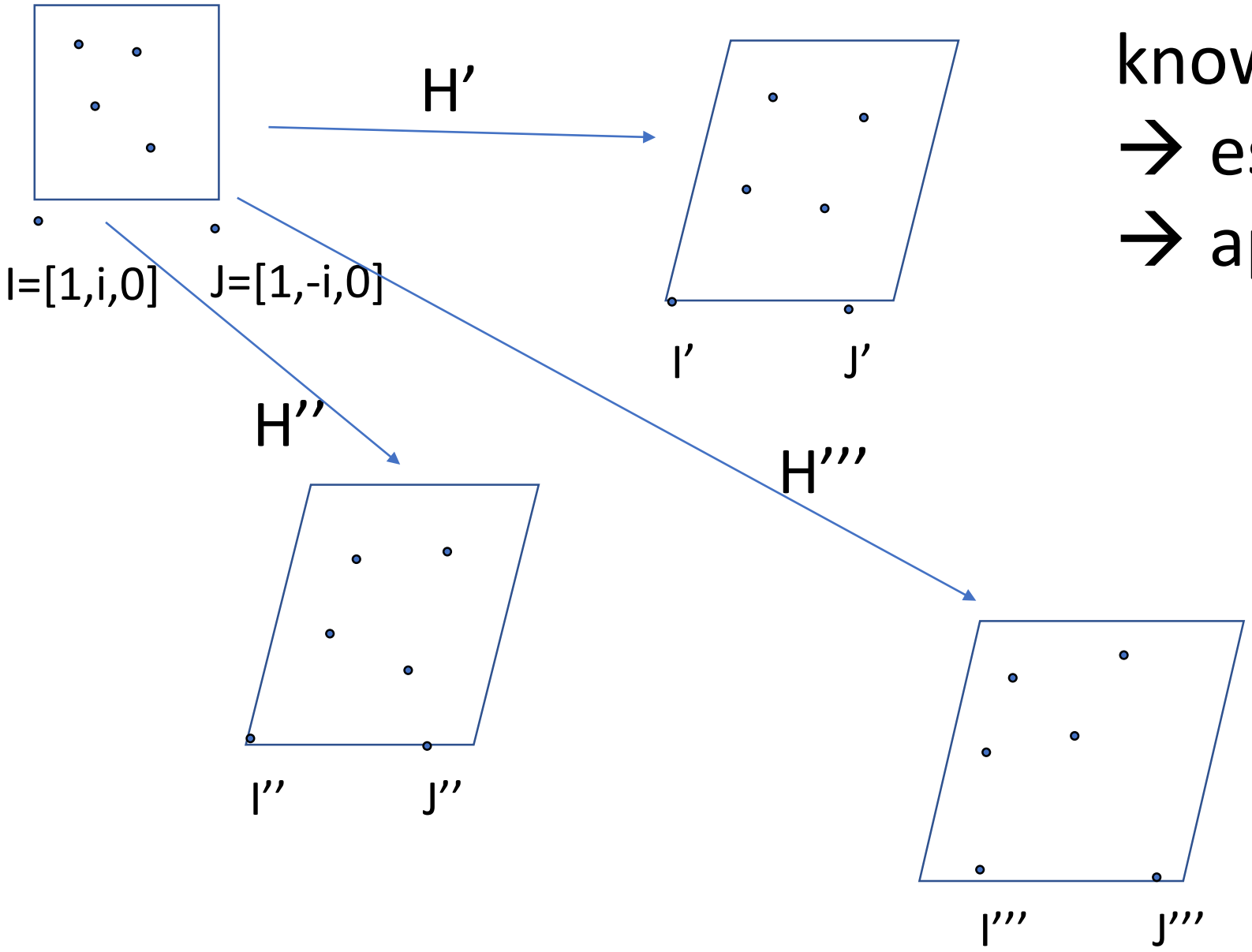
known planar scene:
take at least 3 images with
the same camera (constant $\mathbf{K}$)







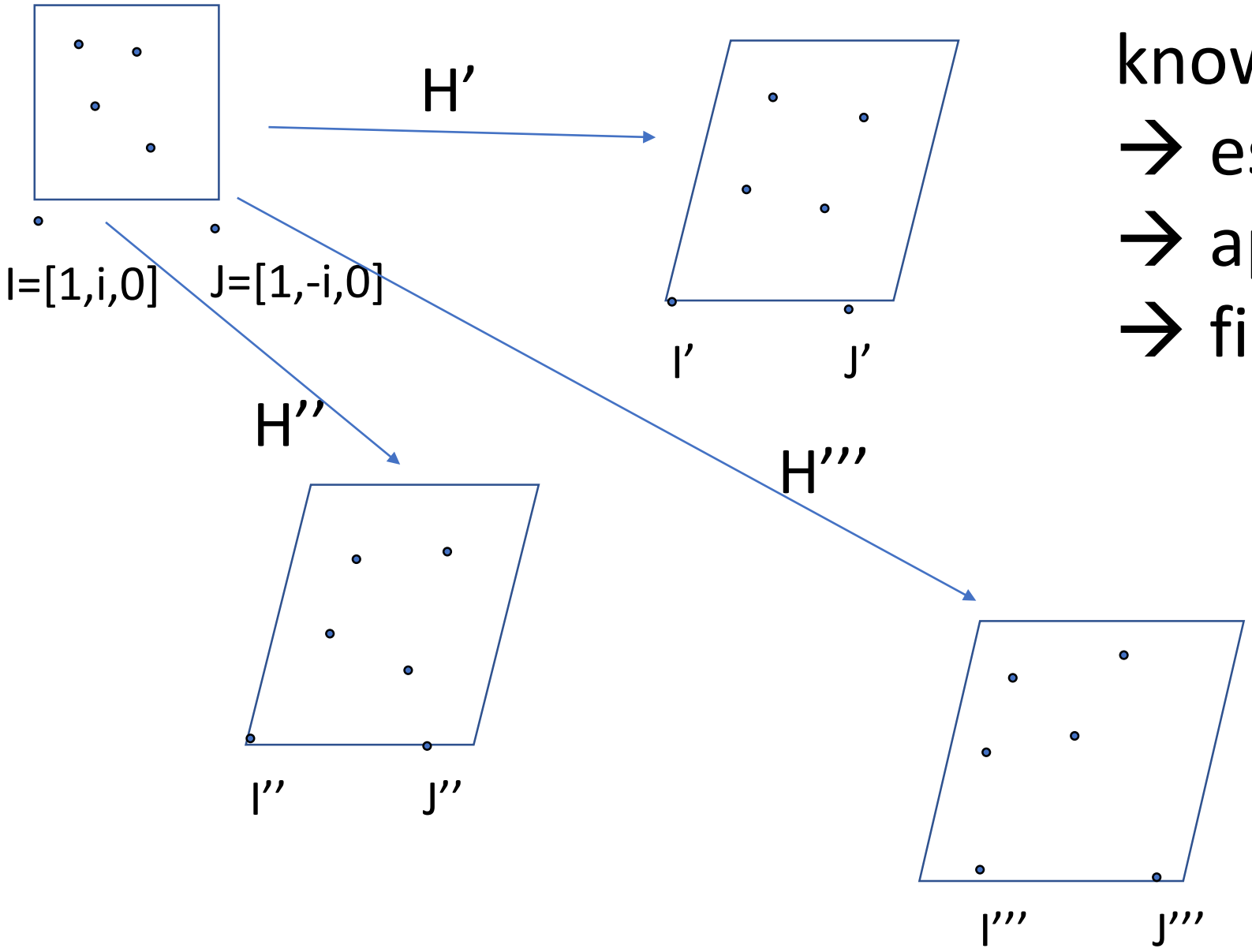
known planar scene
→ estimate homographies



known planar scene

→ estimate homographies

→ apply them to **I** and **J**

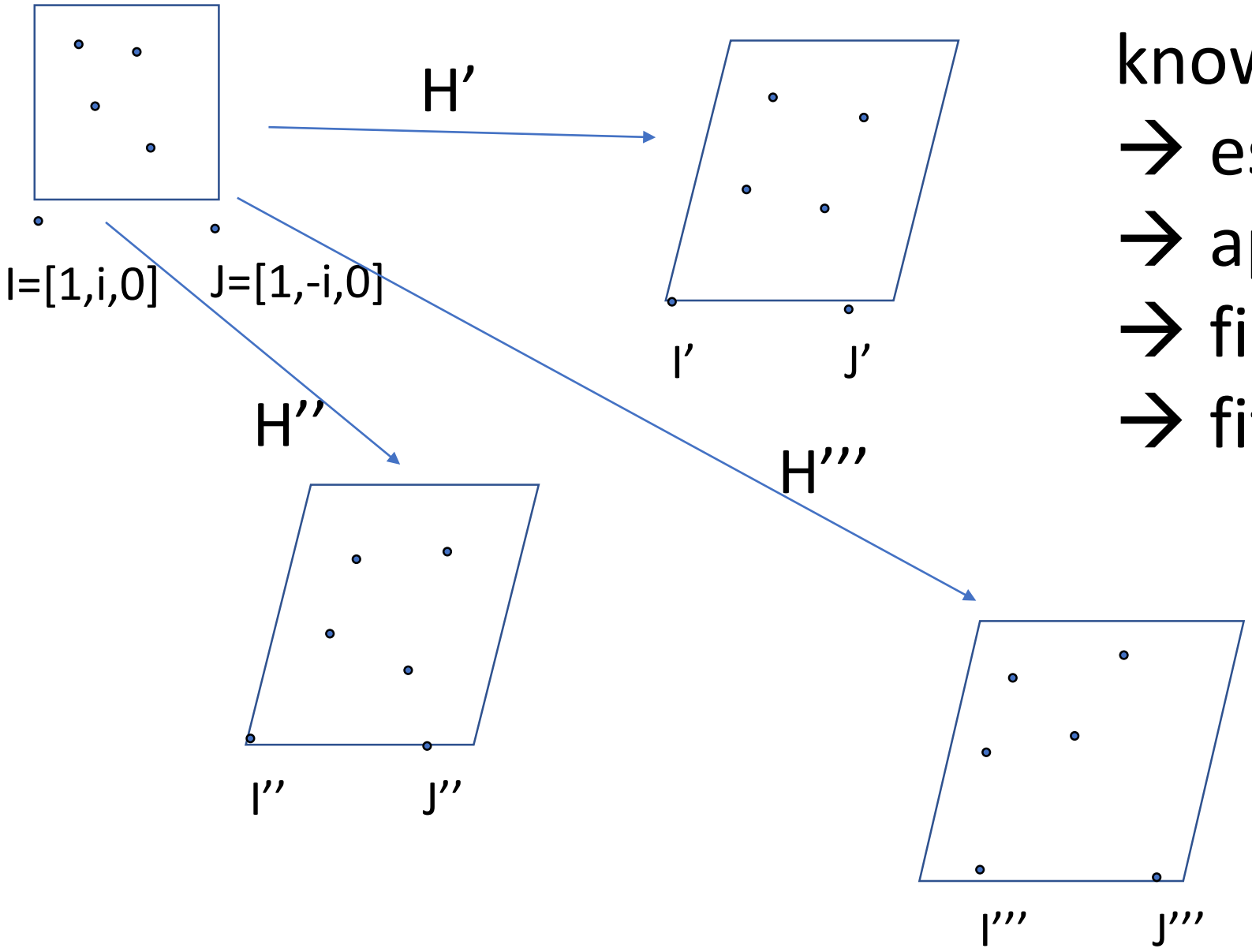


known planar scene

→ estimate homographies

→ apply them to **I** and **J**

→ find **I’**, **J’**, **I’’**, **J’’**, **I’’’**, **J’’’**



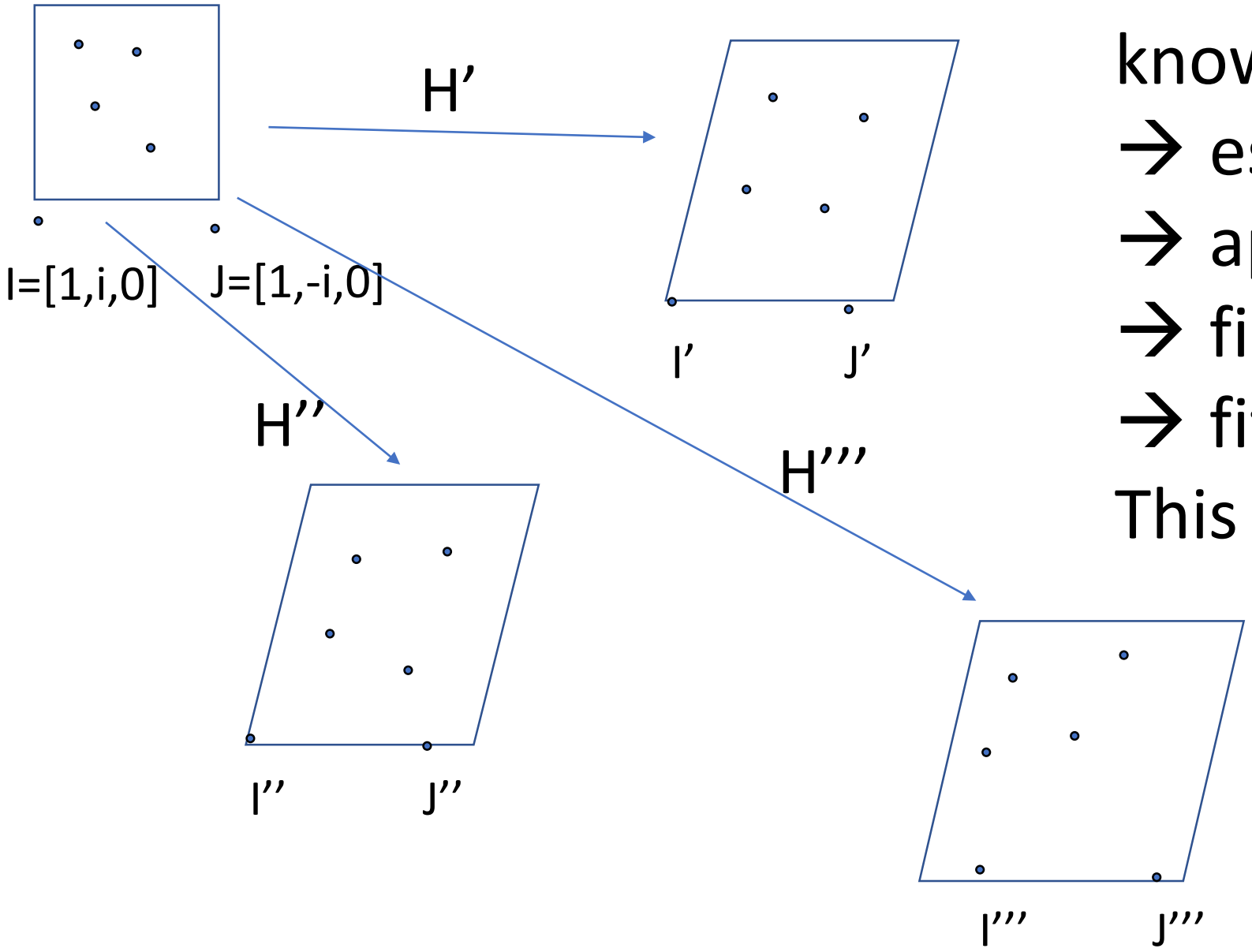
known planar scene

→ estimate homographies

→ apply them to **I** and **J**

→ find **I’**, **J’**, **I’’**, **J’’**, **I’’’**, **J’’’**

→ fit a conic ω to them:



known planar scene

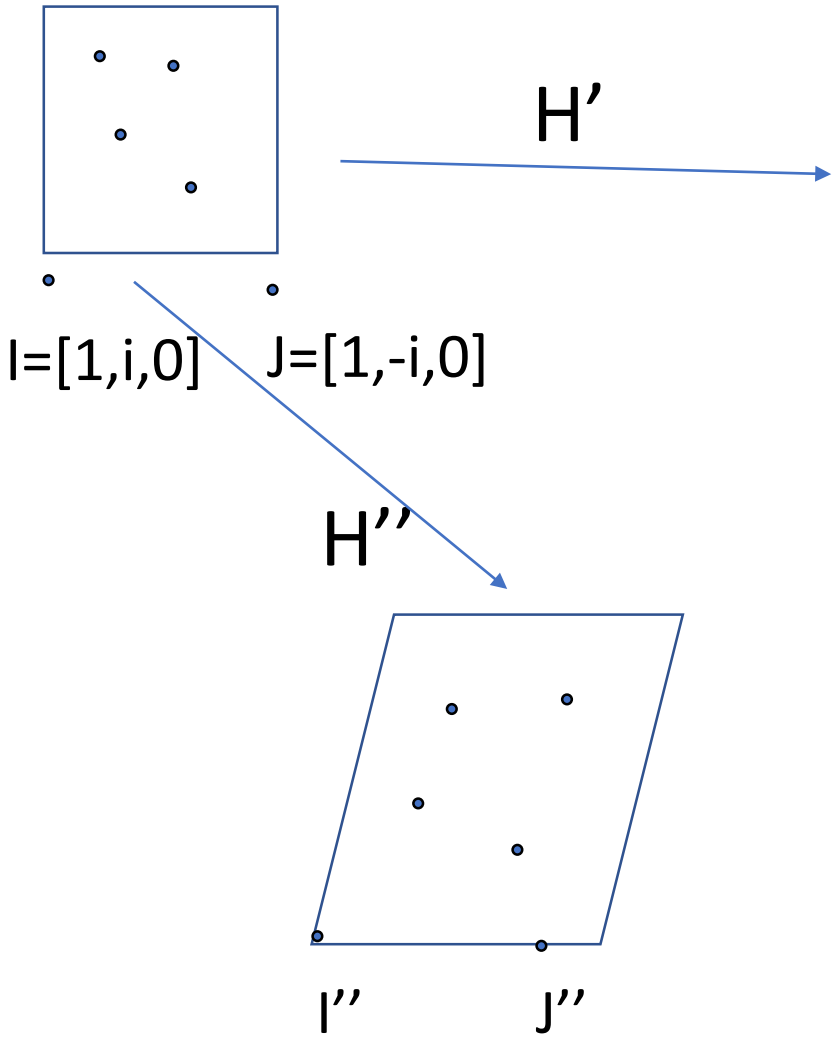
→ estimate homographies

→ apply them to **I** and **J**

→ find **I’**, **J’**, **I’’**, **J’’**, **I’’’**, **J’’’**

→ fit a conic to them:

This conic is the IAC ω !!



known planar scene

→ estimate homographies

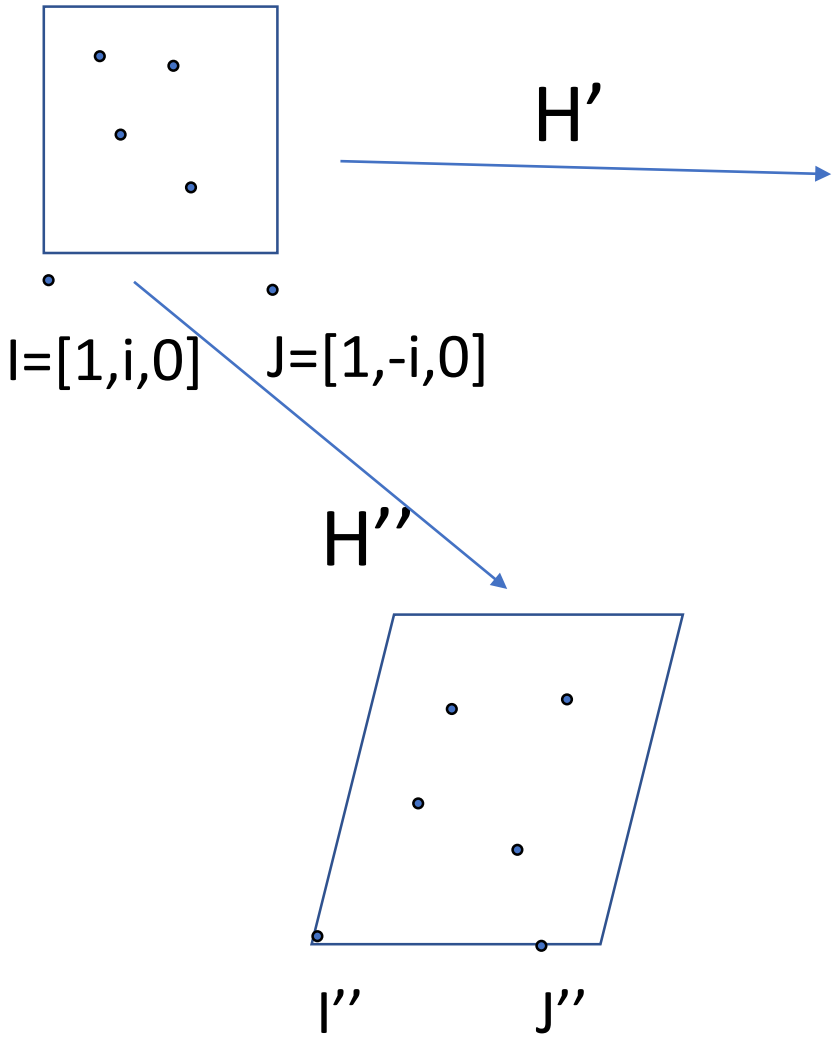
→ apply them to $\mathbf{I}$ and $\mathbf{J}$

→ find $\mathbf{I}'$, $\mathbf{J}'$, $\mathbf{I}''$, $\mathbf{J}''$, $\mathbf{I}'''$, $\mathbf{J}'''$

→ fit a conic to them:

→ This conic is the IAC ω !!

→ then take the DIAC ω^{-1}



known planar scene

→ estimate homographies

→ apply them to $\mathbf{I}$ and $\mathbf{J}$

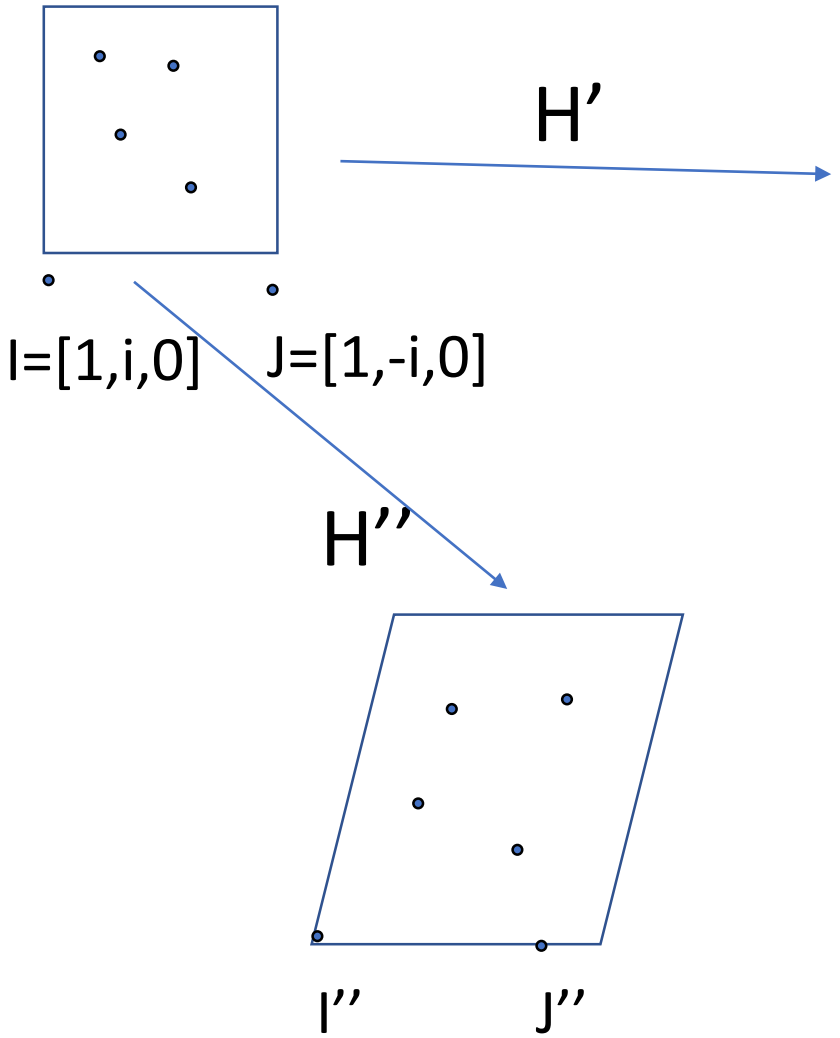
→ find $\mathbf{I}'$, $\mathbf{J}'$, $\mathbf{I}''$, $\mathbf{J}''$, $\mathbf{I}'''$, $\mathbf{J}'''$

→ fit a conic to them:

→ This conic is the IAC ω !!

→ then take the DIAC ω^{-1}

→ find $\mathbf{K}$ by Cholesky factorisation of $\omega^{-1} = \mathbf{K}\mathbf{K}^{\mathrm{T}}$



known planar scene

→ estimate homographies

→ apply them to $\mathbf{I}$ and $\mathbf{J}$

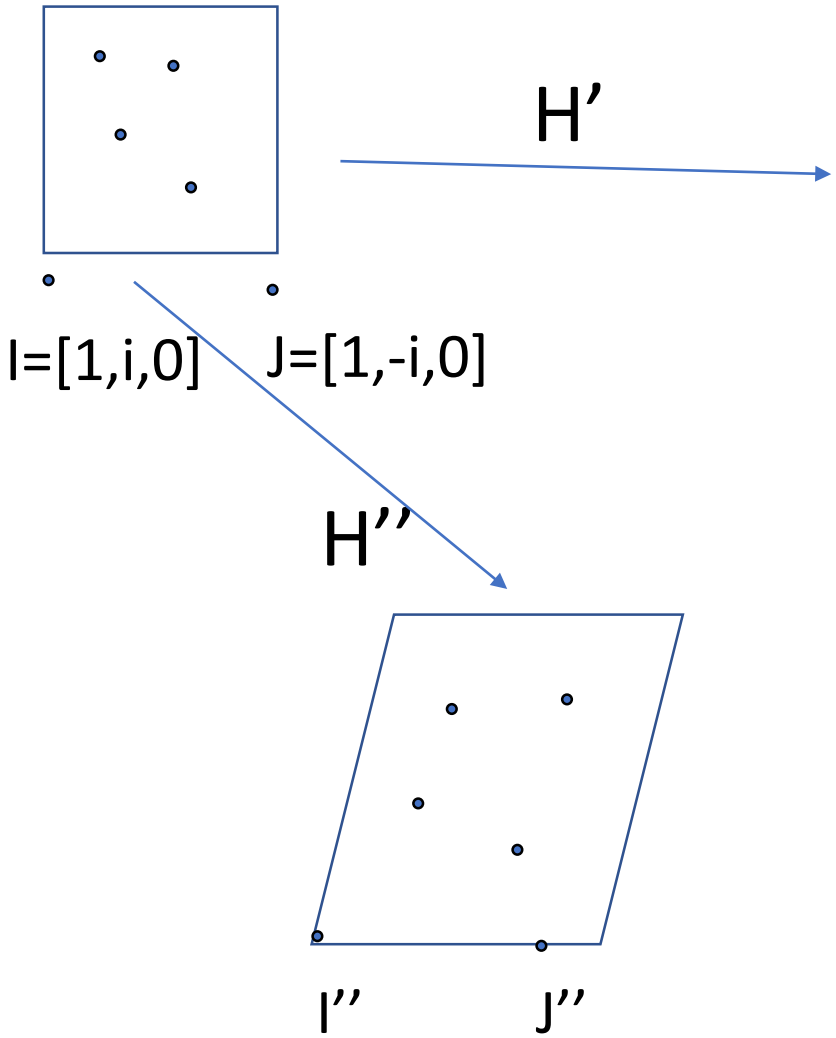
→ find $\mathbf{I}'$, $\mathbf{J}'$, $\mathbf{I}''$, $\mathbf{J}''$, $\mathbf{I}'''$, $\mathbf{J}'''$

→ fit a conic to them: (HOW?)

→ This conic is the IAC ω !!

→ then take DIAC ω^{-1}

→ find $\mathbf{K}$ by Cholesky factorisation of $\omega^{-1} = \mathbf{K}\mathbf{K}^{\mathrm{T}}$



known planar scene

→ estimate homographies

→ apply them to $\mathbf{I}$ and $\mathbf{J}$

→ find $\mathbf{I}'$, $\mathbf{J}'$, $\mathbf{I}''$, $\mathbf{J}''$, $\mathbf{I}'''$, $\mathbf{J}'''$

→ fit a conic to them: (HOW?)

→ This conic is the IAC ω !! (WHY?)

→ then take DIAC ω^{-1}

→ find $\mathbf{K}$ from Cholesky factorisation of $\omega^{-1} = \mathbf{K}\mathbf{K}^{\mathrm{T}}$

HOW? Fitting ω to images of circular points

$$I' = H'I = [h_1, h_2, h_3]I = [h_1, h_2, h_3]\begin{bmatrix}1\\ i\\ 0\end{bmatrix} = h_1 + ih_2 \in \omega$$

$$(h_1 + ih_2)^T \omega (h_1 + ih_2) = 0$$

Real part = 0 and imaginary part = 0

$$h_1{}^T \omega h_2 = 0$$

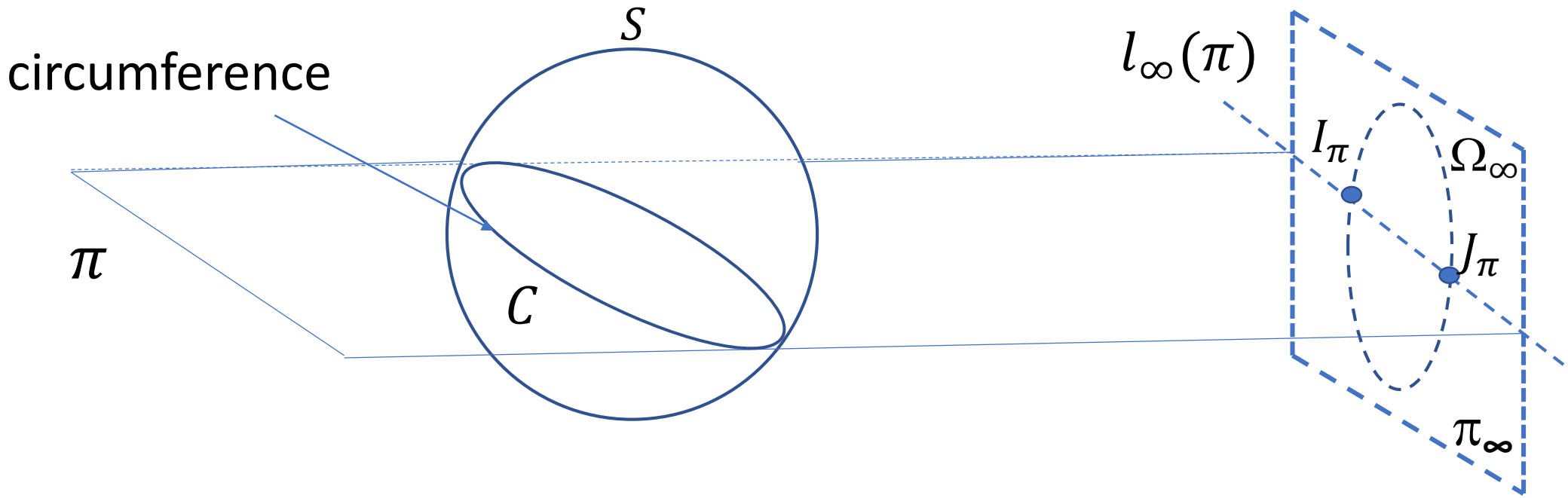
$$h_1{}^T \omega h_1 - h_2{}^T \omega h_2 = 0$$

2 **real** eqns for each image:

→ at least 3 images needed

WHY does IAC ω contain images of circular points?

1. Does the absolute conic Ω_∞ contain all the circular points (i.e. the circular points I_π, J_π of any plane belong to absolute conic Ω_∞) ?

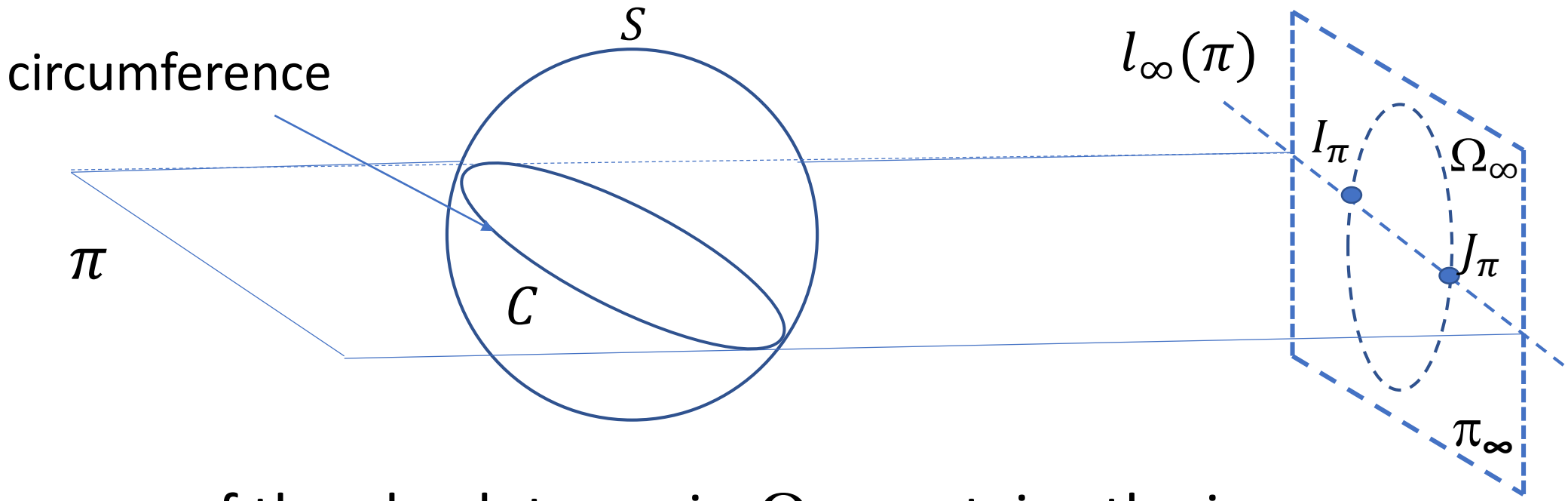


C crosses $l_\infty(\pi)$ at circular points I_π, J_π but C c S, thus $I_\pi, J_\pi \in S$

In addition, $I_\pi, J_\pi \in \pi_\infty \rightarrow I_\pi, J_\pi \in \pi_\infty \cap S = \Omega_\infty$

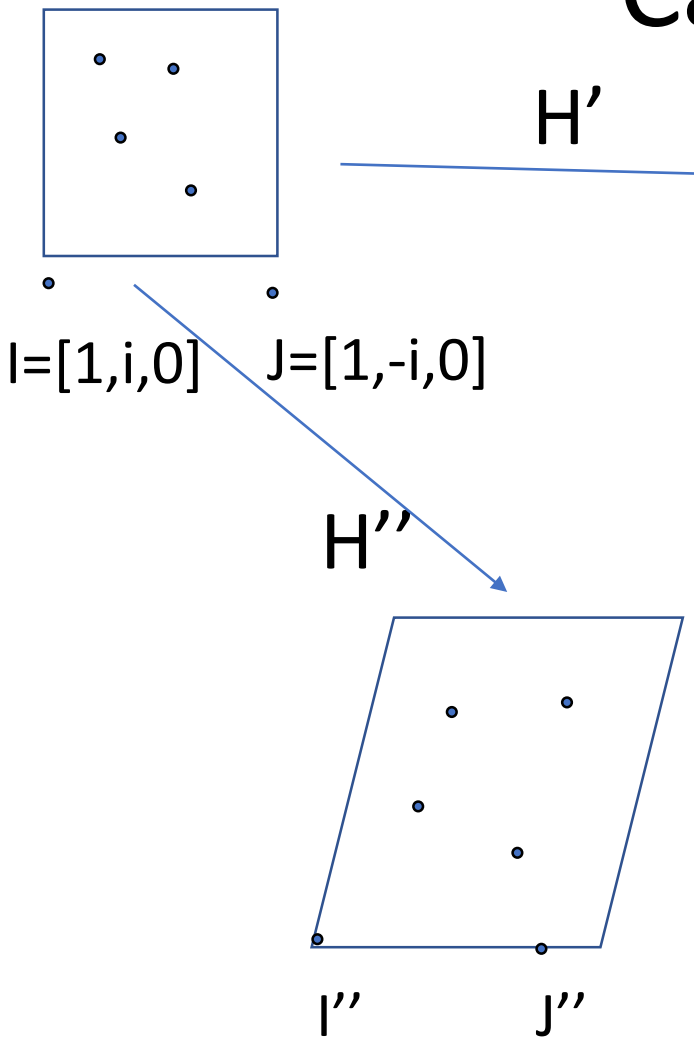
WHY does IAC ω contain images of circular points?

1. Yes: the absolute conic Ω_∞ contains all the circular points (i.e. the circular points I_π, J_π of any plane belong to absolute conic Ω_∞) !!



→ The image ω of the absolute conic Ω_∞ contains the image I'_π, J'_π of all the circular points I_π, J_π

Camera calibration from images of a known scene



Take >= 3 images

→ estimate homographies

→ apply them to $\mathbf{I}$ and $\mathbf{J}$

→ find $\mathbf{I}'$, $\mathbf{J}'$, $\mathbf{I}''$, $\mathbf{J}''$, $\mathbf{I}'''$, $\mathbf{J}'''$

→ fit a conic to them:

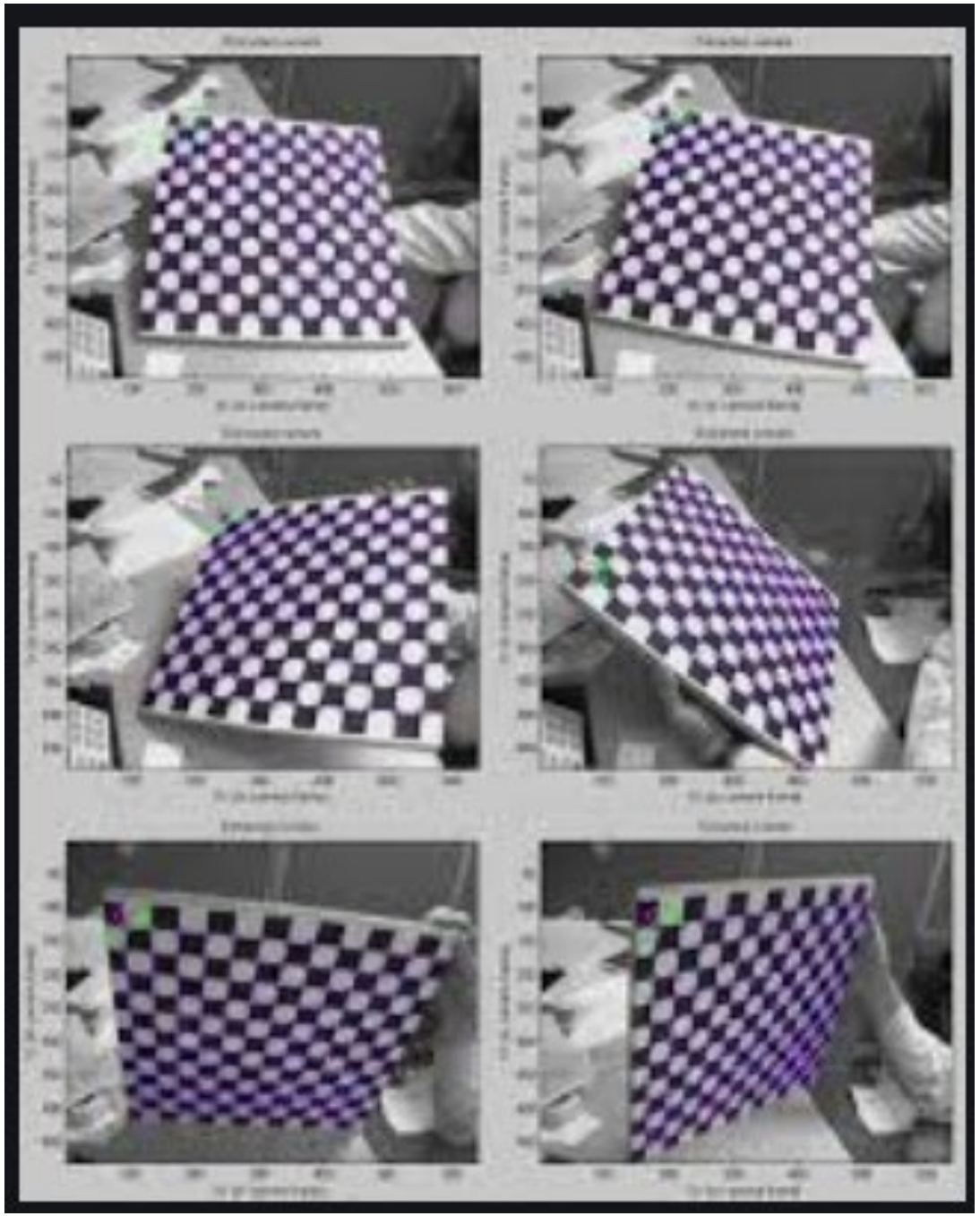
→ This conic is the IAC ω !!

→ then take the DIAC ω^{-1}

→ find $\mathbf{K}$ by Cholesky factorisation of $\omega^{-1} = \mathbf{K}\mathbf{K}^{\mathrm{T}}$

Camera calibration toolbox

- implements Zhang method
- based on the IAC ω
- planar target (easily printable)
- several images ($\sim$ 20) to cope with noise
- also estimates distortion param.
- provides accurate calibration

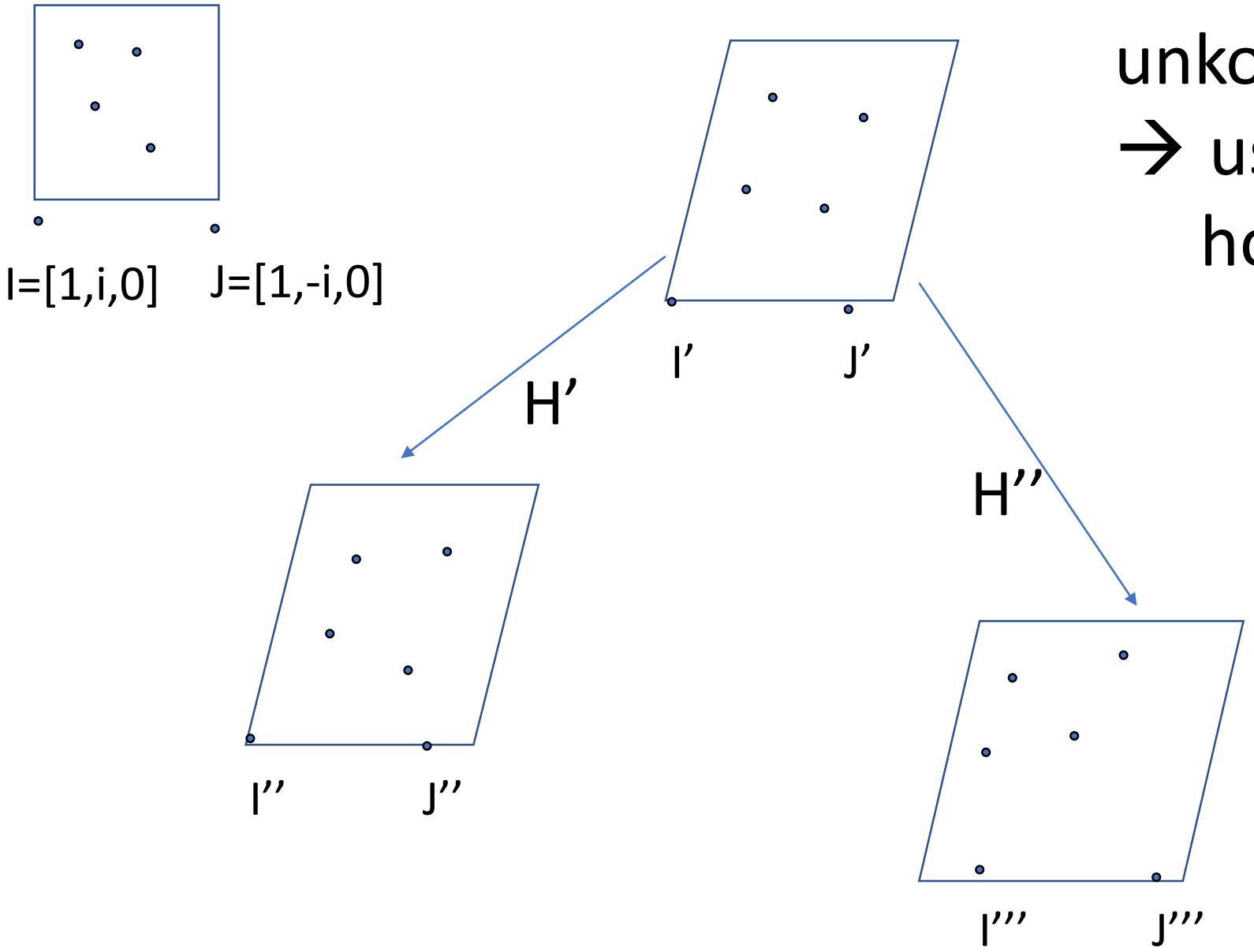


Matlab Calibration Toolbox also contains estimation of distortion parameters

$$x = x_o + (x_o - c_x)(K_1 r^2 + K_2 r^4 + \ldots)$$
$$y = y_o + (y_o - c_y)(K_1 r^2 + K_2 r^4 + \ldots)$$

$$r = (x_o - c_x)^2 + (y_o - c_y)^2 \ .$$

Camera calibration
from images of an **unknown** planar scene



unkown planar scene
→ use image-to-image homographies

$$I'' = H'I'$$
$$I'^T \omega I' = 0$$
$$I''^T \omega I'' = 0$$

$$I'^T H'^T \omega H' I' = 0$$

Unknowns: I' and J' and ω → at least 5 images
(each nonlinear eqn leads to 2 constraints: Re and Im part)

$$I'' = H'I'$$

$$I'^{T}\omega I' = 0$$

$$I''^{T}\omega I'' = 0$$

$$I'^{T}H'^{T}\omega H'I' = 0$$

Unknowns: I' and J' and ω → at least 5 images

(each nonlinear eqn leads to 2 constraints: Re and Im part)

Camera calibration
from images of an **unknown** planar scene

- take images of a planar scene with camera with constant $\mathbf{K}$
- estimate image-image homographies
- formulate equations

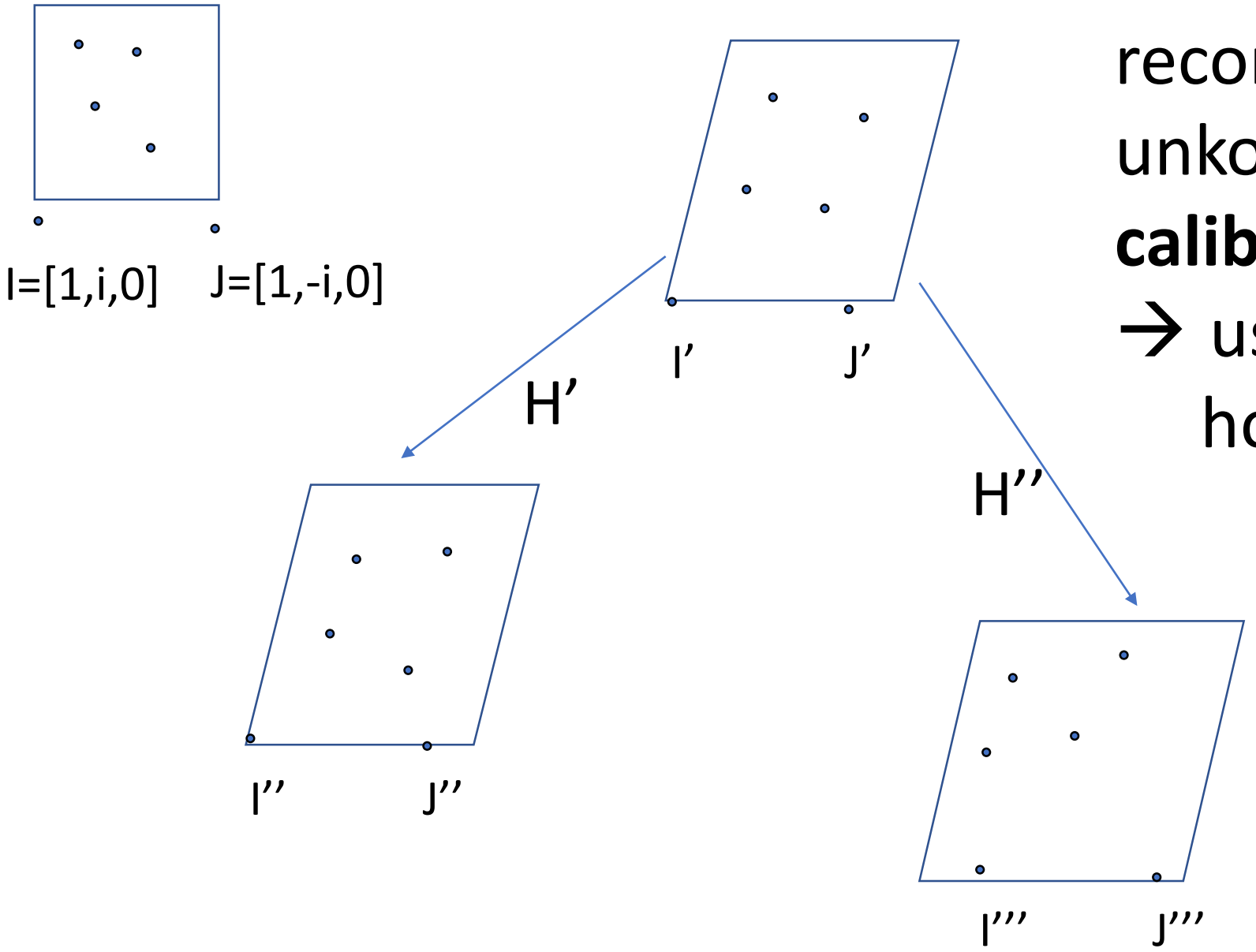
$$I'^{T}\omega I' = 0$$

$$I'^{T}H'^{T}\omega H' I' = 0$$

- solve them for ω and I'
- then take the DIAC ω^{-1}
- find $\mathbf{K}$ by Cholesky factorisation of $\omega^{-1} = \mathbf{K}\mathbf{K}^{\mathrm{T}}$

USUALLY LESS ACCURATE THAN ZHANG METHOD

Reconstruction of an unknown planar scene from calibrated images (i.e. images taken by calibrated cameras)



reconstruction of an unkown planar scene, but **calibrated camera**

→ use image-to-image homographies

same calibrated camera

$$I'' = H'I'$$

$$I'^{T}\omega I' = 0$$

$$I''^{T}\omega I'' = 0$$

$$I'^{T}H'^{T}\omega H'I' = 0$$

$\omega = \left(\mathrm{K}\mathrm{K}^{\mathrm{T}}\right)^{-1} = \mathrm{K}^{-\mathrm{T}}\mathrm{K}^{-1}$ is known after calibration

just 4 unknowns I' complex coordinates→ at least 2 images
(each eqn leads to 2 constraints: Re and Im part)

two different calibrated cameras

$$I'' = H'I'$$

$$I'^T \omega' I' = 0$$

$$I''^T \omega'' I'' = 0$$

$$I'^T H'^T \omega'' H' I' = 0$$

ω', ω'' known after calibration

just 4 unknowns I' complex coordinates→ at least 2 images
(each eqn leads to 2 constraints: Re and Im part)

$$I'^T \omega' I' = 0$$
$$I'^T H'^T \omega'' H' I' = 0$$

reduces to intersection of two conics:
ω' and $H'^T \omega'' H'$
→
two resulting pairs of
imaged circular points
→
selection based on reprojection
or on an additional (third) image

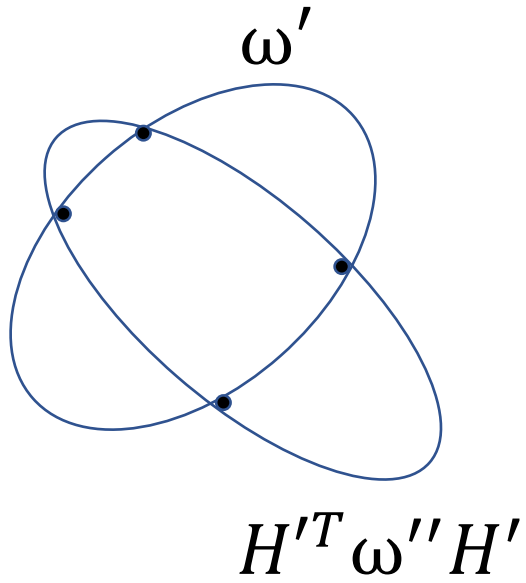


image of circular points: {I', J'} = $l'_\infty \cap \omega$

- Image of the circular points →Image of the conic dual to the circular points

$$C_\infty^{*\prime} = I'J'^T + J'I'^T$$

- Singular value decomposition

$$\text{svd}(C_\infty^{*\prime}) = U \begin{vmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{vmatrix} U^T = H_{rect}^{-1} C_\infty^* H_{rect}^{-T}$$

- Rectifiying homography (from svd output U)

$$H_{rect} = U^T$$

In practice …

- Image of the circular points →Image of the conic dual to the circular points

$$C_\infty^{*\prime} = I'J'^T + J'I'^T$$

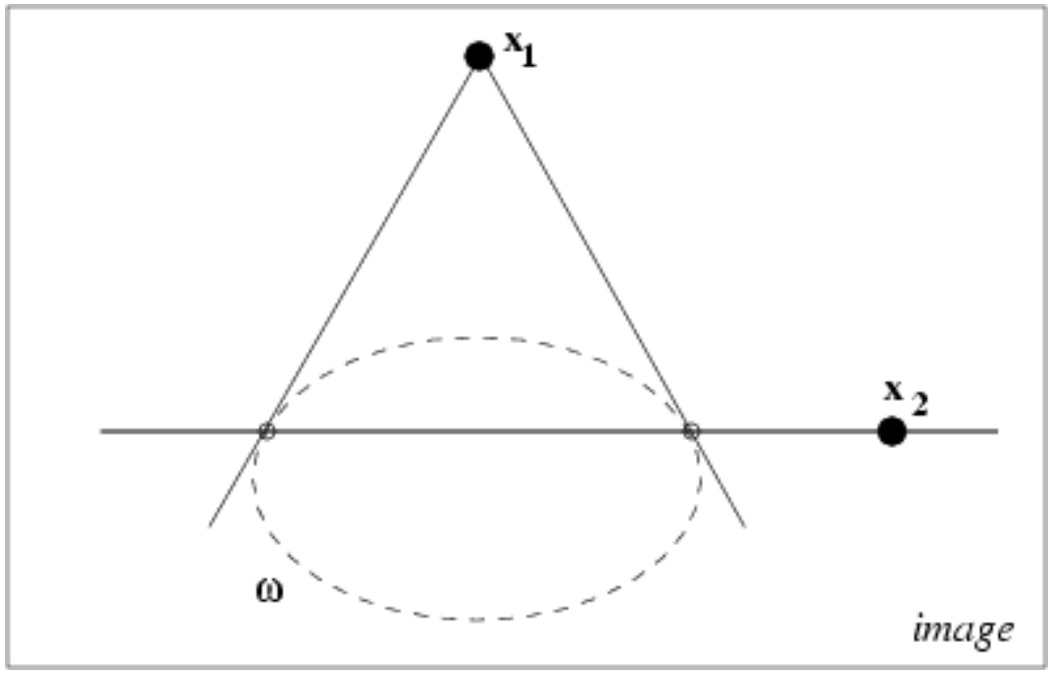
- Singular value decomposition

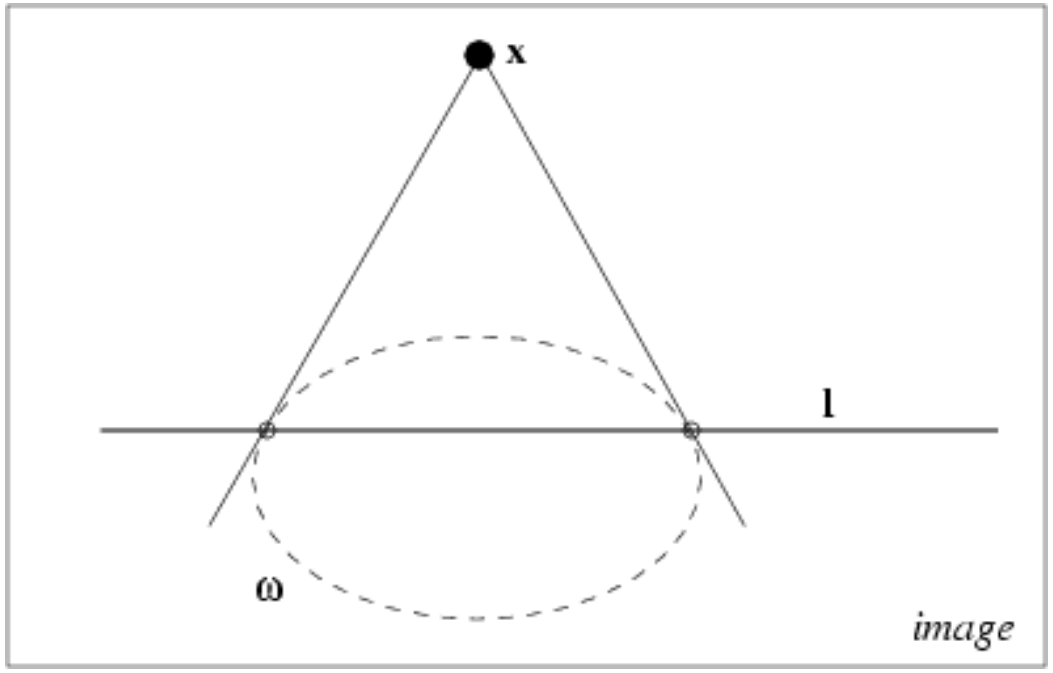
$$\text{svd}(C_\infty^{*\prime}) = U \begin{bmatrix} s_1 & 0 & 0 \\ 0 & s_2 & 0 \\ 0 & 0 & \varepsilon \end{bmatrix} U^T$$

- Rectifiying homography (from svd output U)

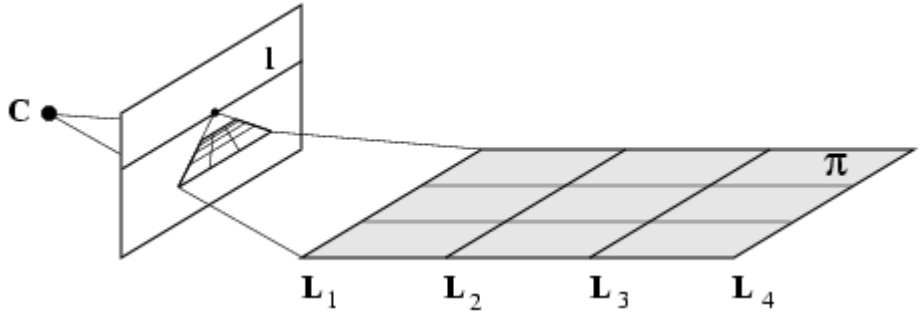
$$H_{rect} = (U \begin{bmatrix} \sqrt{s_1} & 0 & 0 \\ 0 & \sqrt{s_2} & 0 \\ 0 & 0 & 1 \end{bmatrix})^{-1} = \begin{bmatrix} \sqrt{s_1{}^{-1}} & 0 & 0 \\ 0 & \sqrt{s_2{}^{-1}} & 0 \\ 0 & 0 & 1 \end{bmatrix} U^T$$

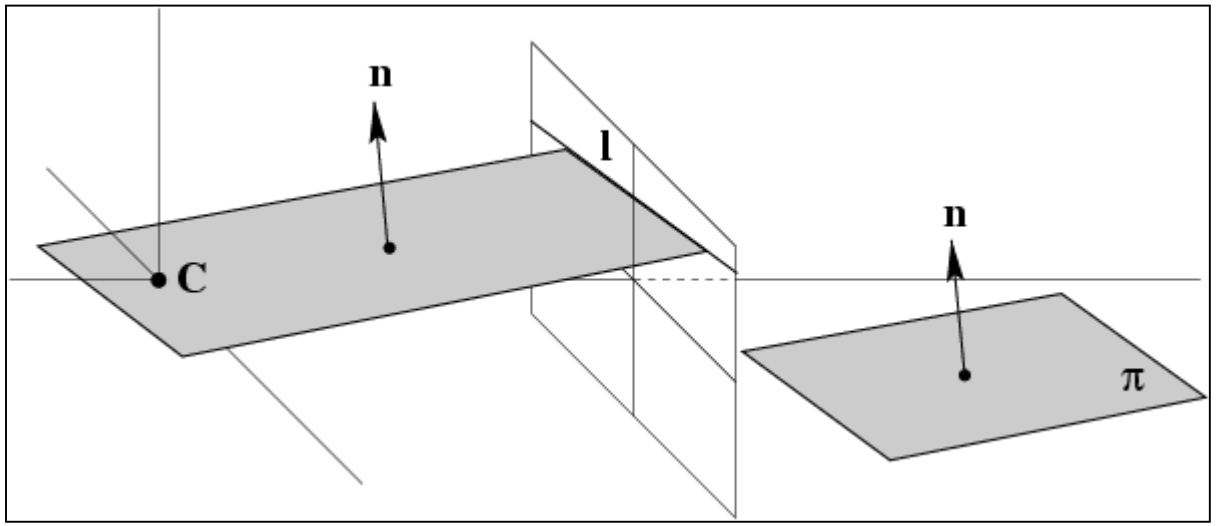
Orthogonality = pole-polar w.r.t. IAC

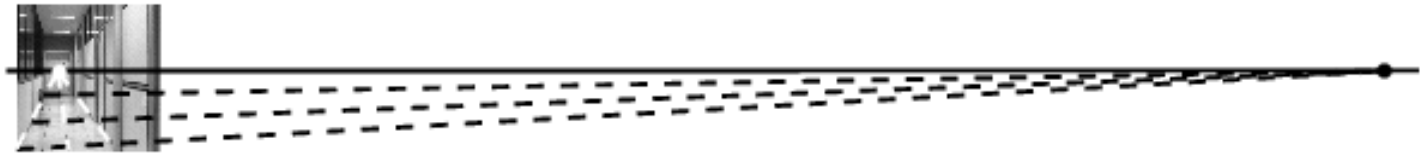




Vanishing lines







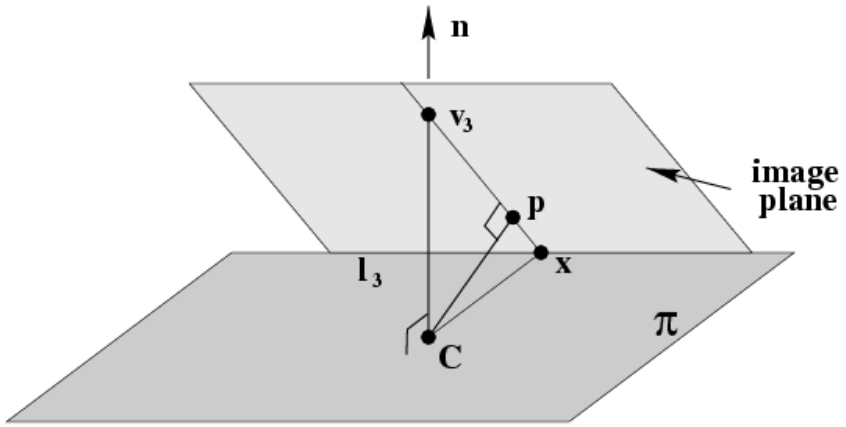
vanishing line: line through two vanishing points

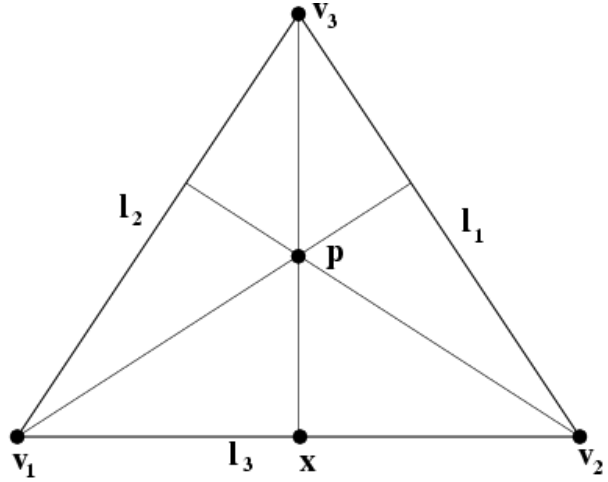
Orthogonality relation: nonlinear equation on ω reduces to linear for orthogonal directions

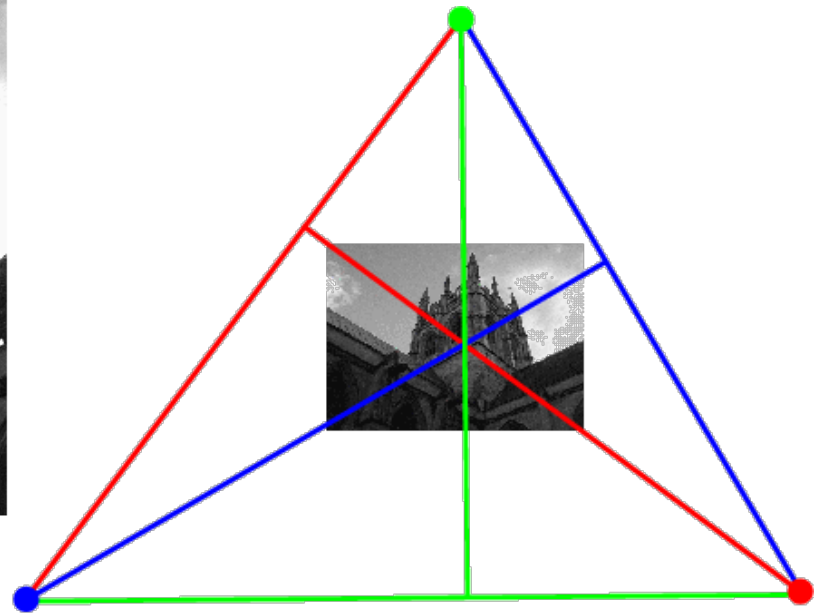
$$\cos\theta = \frac{\mathbf{v}_1^{\mathrm{T}}\omega\mathbf{v}_2}{\sqrt{\left(\mathbf{v}_1^{\mathrm{T}}\omega\mathbf{v}_1\right)\left(\mathbf{v}_2^{\mathrm{T}}\omega\mathbf{v}_2\right)}}$$

$$\mathbf{v}_1^{\mathrm{T}}\omega\mathbf{v}_2 = 0$$

Calibration of NATURAL CAMERAS from vanishing points of orthogonal directions







Orthogonality relation

$$\cos\theta = \frac{{v_1}^{T}\omega v_2}{\sqrt{({v_1}^{T}\omega v_1)({v_2}^{T}\omega v_2)}} = 0$$

$${v_1}^{T}\omega v_2 = 0$$
$${v_2}^{T}\omega v_3 = 0$$
$${v_3}^{T}\omega v_1 = 0$$

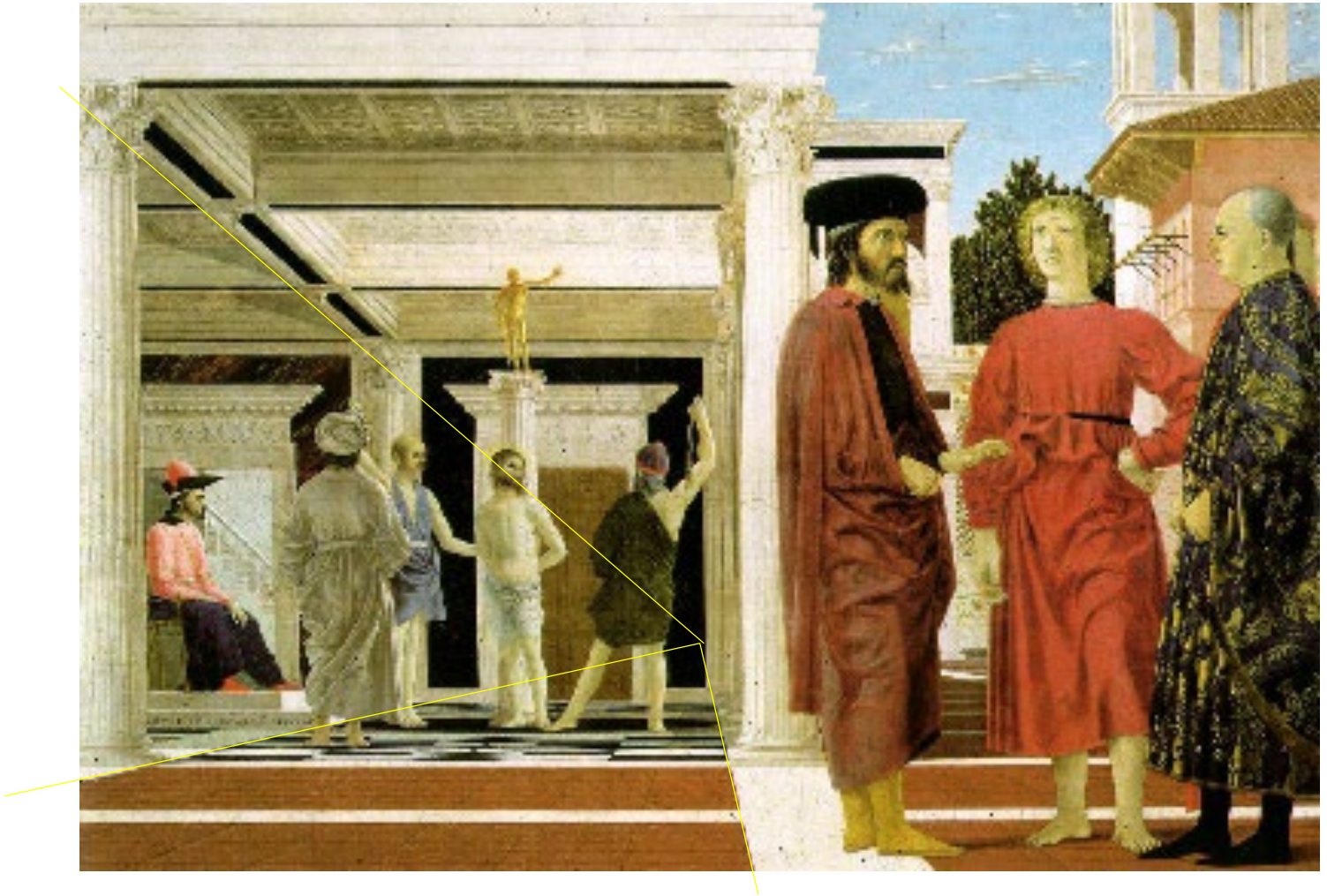
The painter’s eye

An Example of Camera calibration

where was the painter’s eye (wrt to the painting)?

Assumptions: - the floor is horizontal
- floor tiles are square

Vanishing point of «x» direction: V_x



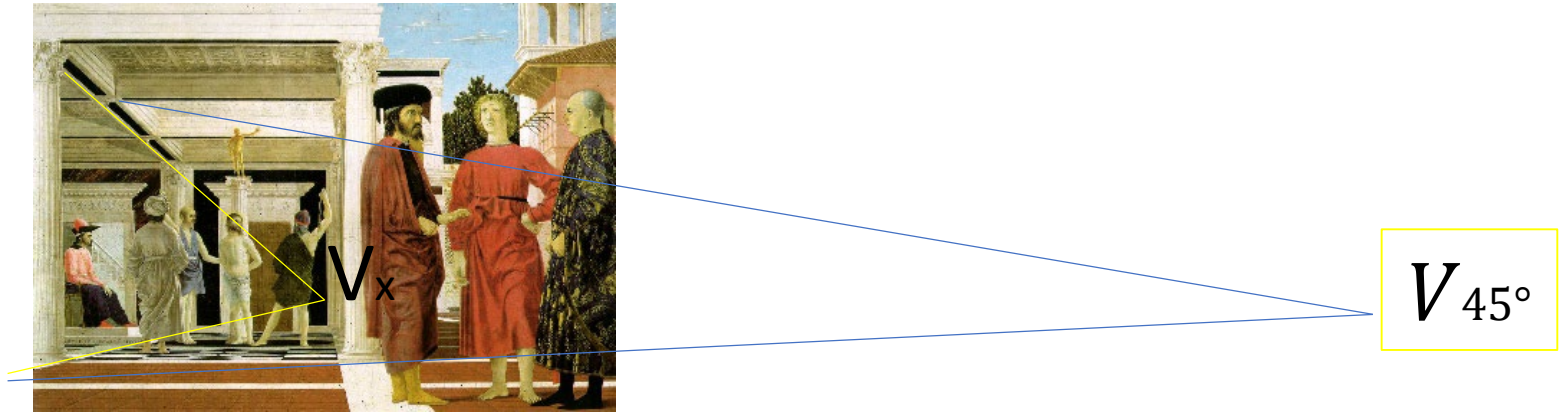
Vanishing point of «y» direction

Theorem: viewing ray associated to the vanishing point of a direction d is parallel to the direction **d**

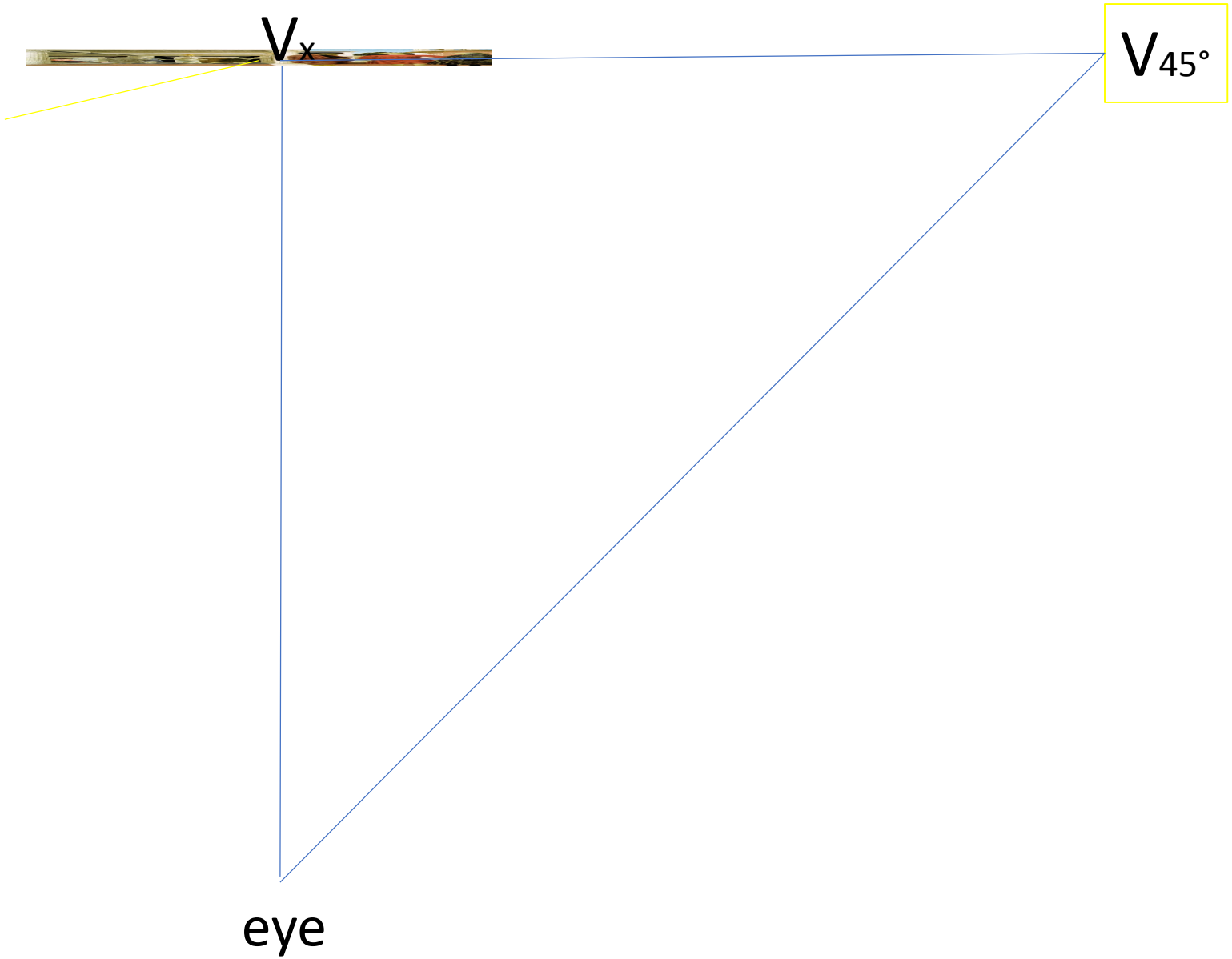
- Floor is horizontal → line (V_x, eye) is horizontal
- Vy is at the infinity → line (eye, V_∞) is parallel to the painting
- Theorem → line (V_∞, eye) is orthogonal to line (V_x, eye)
- → line (eye, V_x) is horizontal and perpendicular to y axis
- … eye is on the normal to the y-axis through the point V_x

- How far?

Tiles are square → diagonals form a 45° angle with x-direction



Theorem → line (eye, $V_{45\circ}$) forms a 45° angle with line (eye, V_x)
floor horizontal → line (eye, $V_{45\circ}$) is also horizontal


V_x
$V_{45°}$
eye

Camera calibration from

- a planar object of known (reconstructed) shape and
- a vanishing point of the normal to the plane

calibration after reconstruction of a planar face

Shape of the face:
known after metric rectificaton

How much additional information is needed to calibrate the camera?

- Vanishing point of the direction normal to the reconstructed face

Vanishing point of vertical direction

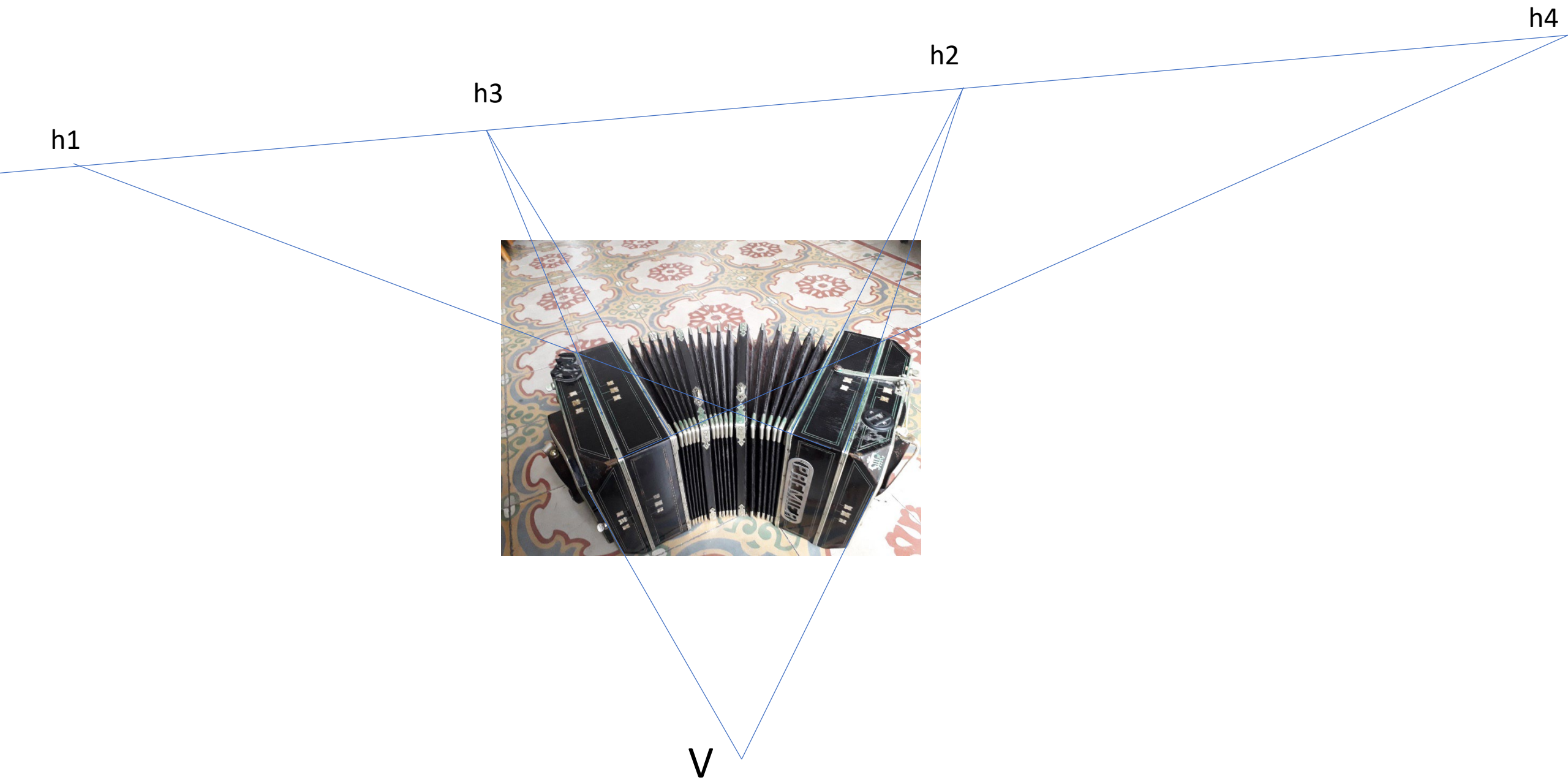
V

Camera calibration (assume skew = 0)

- Only four unknowns:

$$\omega = (KK^T)^{-1} = \begin{vmatrix} a^2 & 0 & -u_0 a^2 \\ * & 1 & -v_0 \\ * & * & f_Y^2 + a^2 u_0^2 + v_0^2 \end{vmatrix}$$

→ only four equations are needed


h1
h3
h2
h4
V

Calibration: rectified planar face plus vanishing point of the direction normal to the face

$$h_1^T \omega h_2 = 0$$
$$h_3^T \omega h_4 = 0$$
$$v^T \omega h_1 = 0$$
$$v^T \omega h_2 = 0$$

3° and 4° equations are linearly independent,

but there are no further ones (why?)

- → solve for ω
- → find $\mathbf{K}$ (by Cholesky factorisation of $\omega^{-1} = \mathbf{K}\mathbf{K}^{\mathrm{T}}$)

Calibration from rectified face plus orthogonal vanishing point

direct method:
independent of the chosen pairs of
mutually orthogonal vanishing points

from

- the reconstructing homography H_R from given img to rectified img
- the image of the line at the infinity $\boldsymbol{l}'_{\infty}$
- the vanishing point $\mathbf{v}$ along the direction orthogonal to the face

Calibration from rectified face plus orthogonal vanishing points

- from $\mathbf{h}_1^T \omega \mathbf{v} = 0$ and $\mathbf{h}_2^T \omega \mathbf{v} = 0$, and $\boldsymbol{l}'_\infty = \mathbf{h}_1 \times \mathbf{h}_2$ →

$$\boldsymbol{l}'_\infty = \omega \mathbf{v} \text{ (2 eqns)}$$

- from $I' = H_R^{-1} I = H_R^{-1} \begin{bmatrix} 1 \\ i \\ 0 \end{bmatrix} = \begin{bmatrix} h_1 & h_2 & h_3 \end{bmatrix} \begin{bmatrix} 1 \\ i \\ 0 \end{bmatrix} = h_1 + i h_2,$

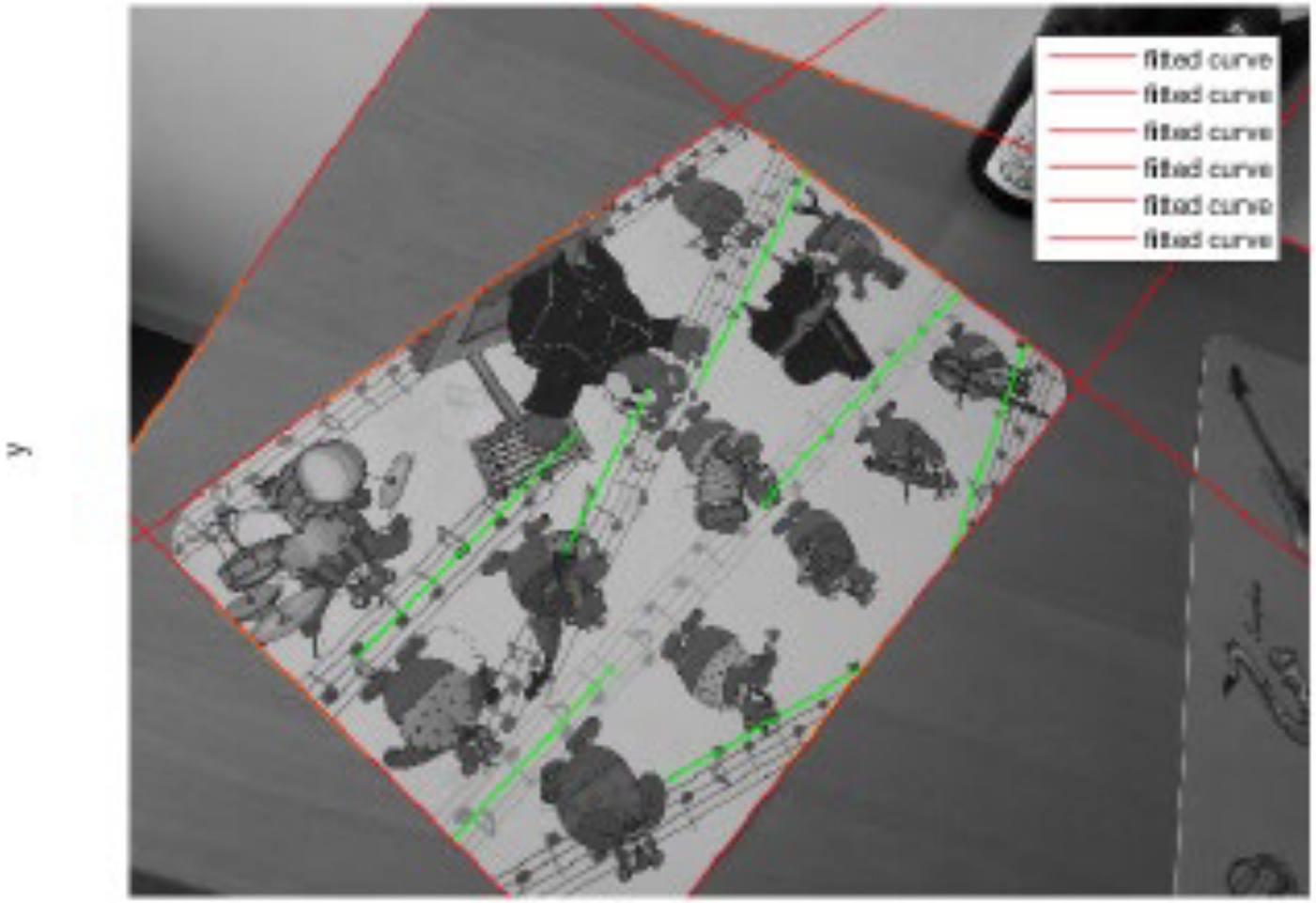
and $(h_1 + i h_2)^T \omega (h_1 + i h_2) = 0$ →

$$h_1{}^T \omega h_2 = 0$$
$$h_1{}^T \omega h_1 - h_2{}^T \omega h_2 = 0$$

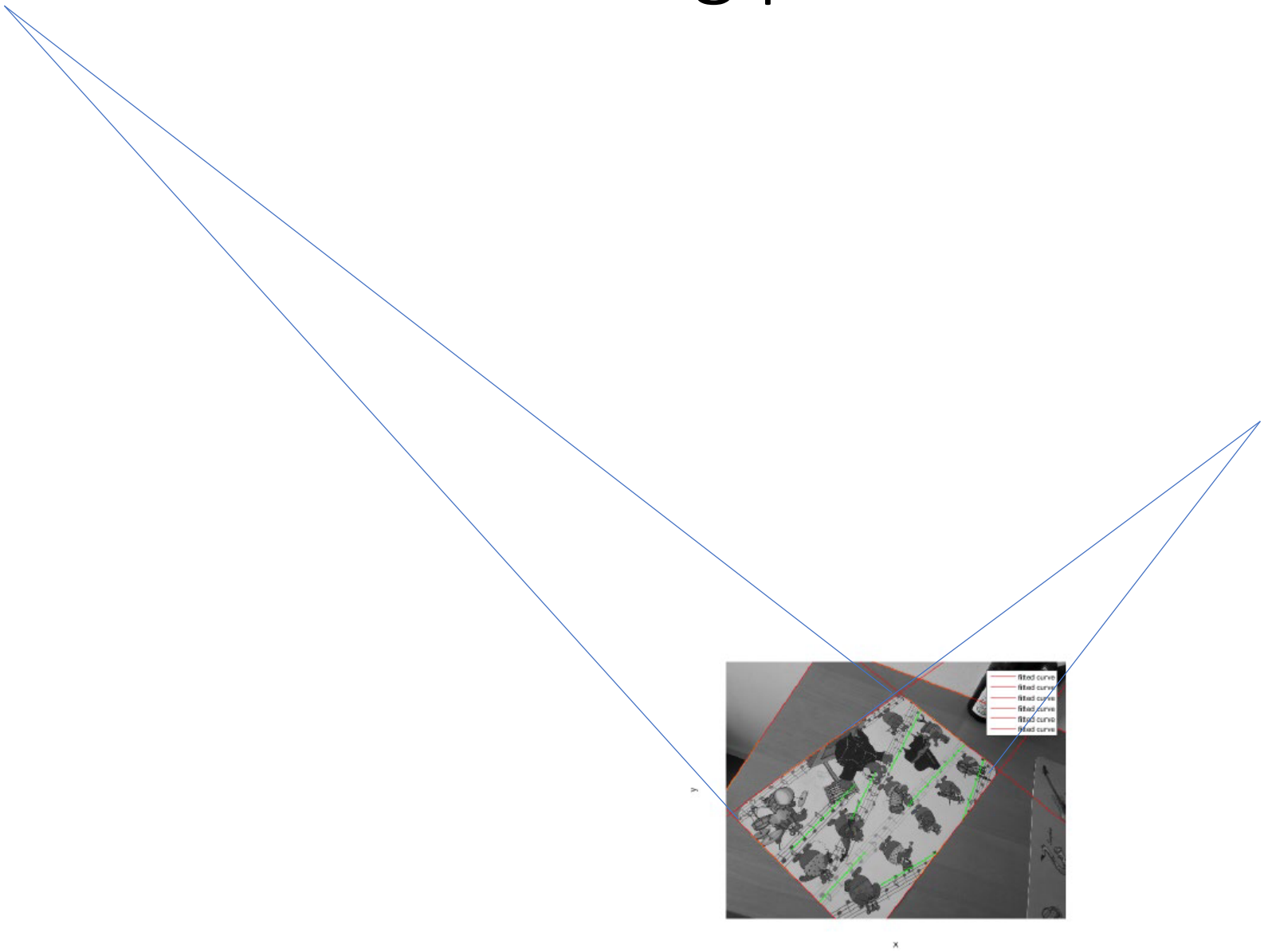
Rectification of a single calibrated image from vanishing points


Vallée d'Aoste
DONNAS
2003
COLLIO
CABERNET FRANC
Saxophone

Images of pairs of parallel lines



vanishing points



Vanishing line l'_∞

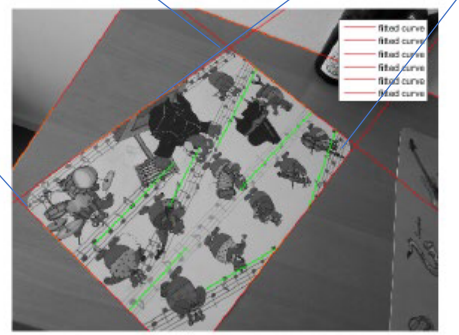


Image of the absolute conic $\omega = (\mathrm{K}\mathrm{K}^{\mathrm{T}})^{-1}$

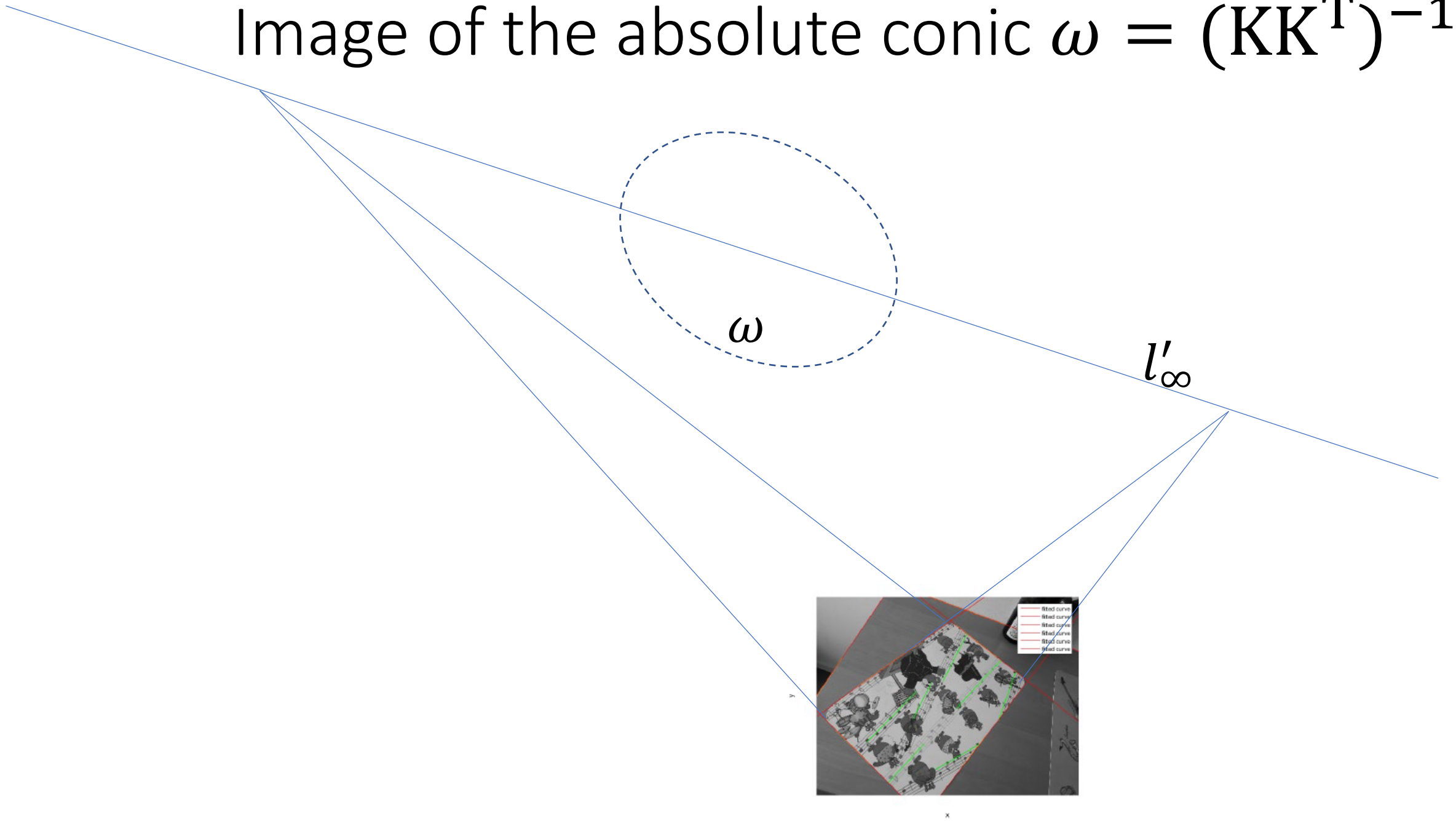


Image of the circular points $l'_\infty \cap \omega$

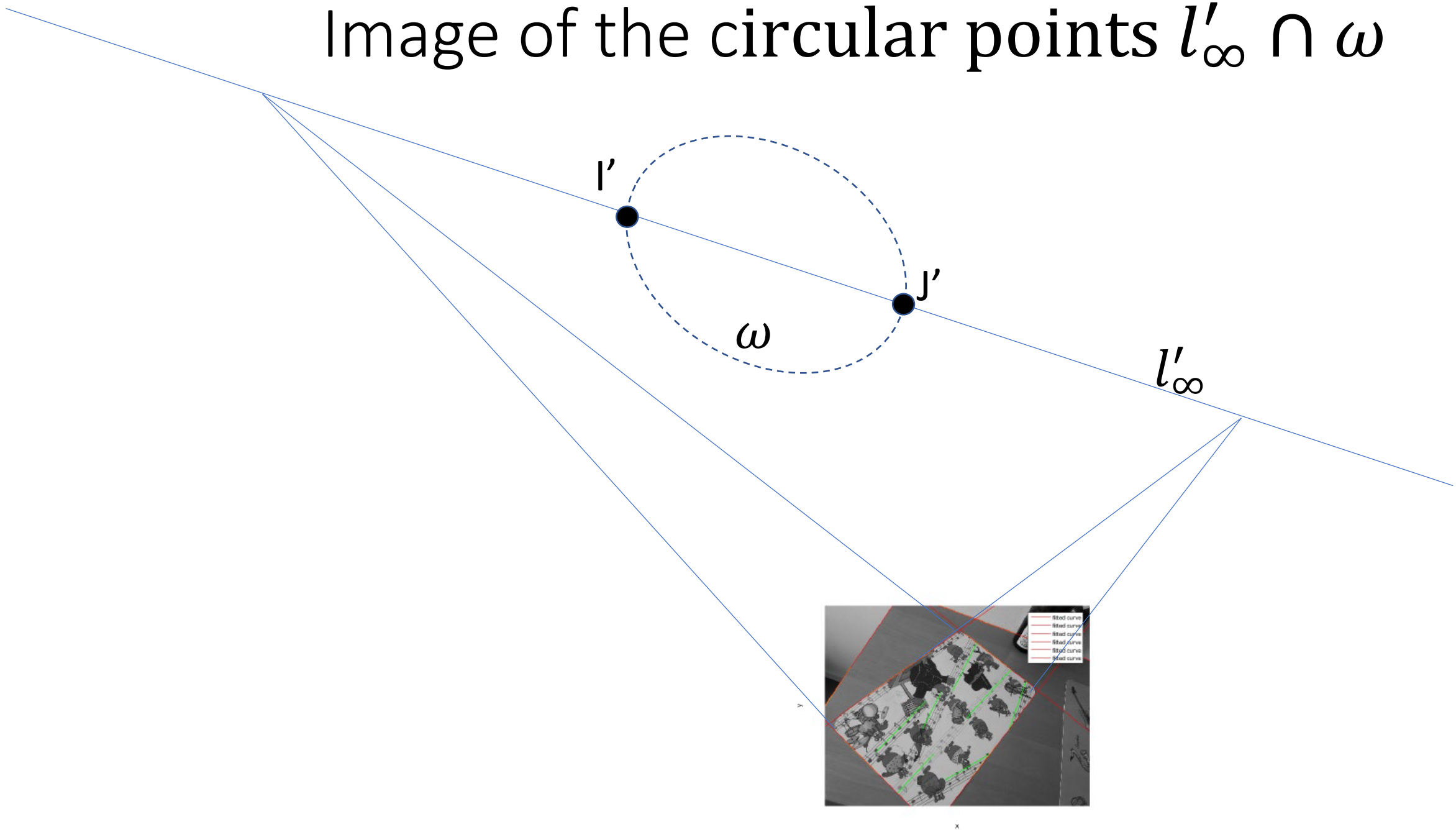
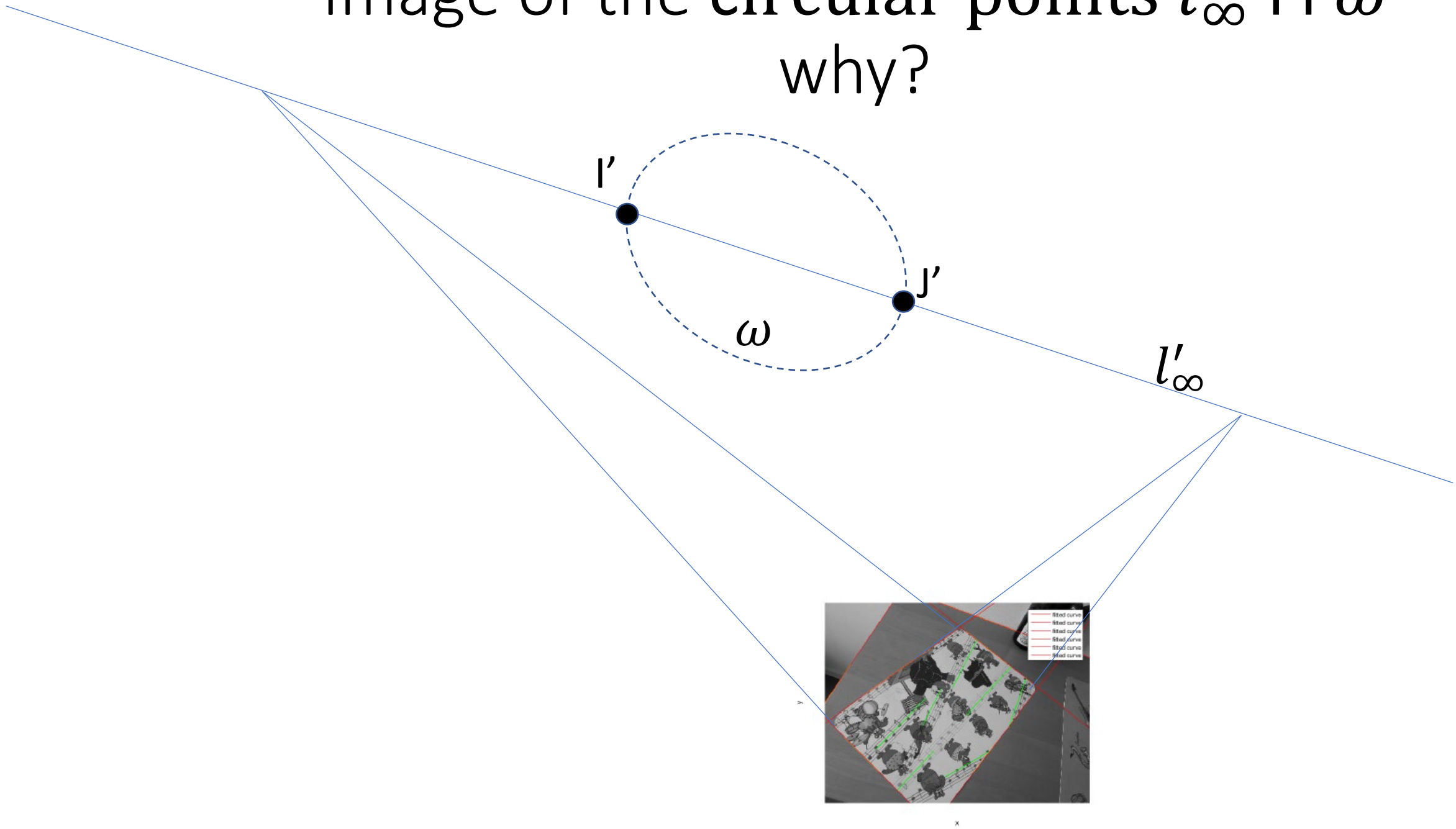
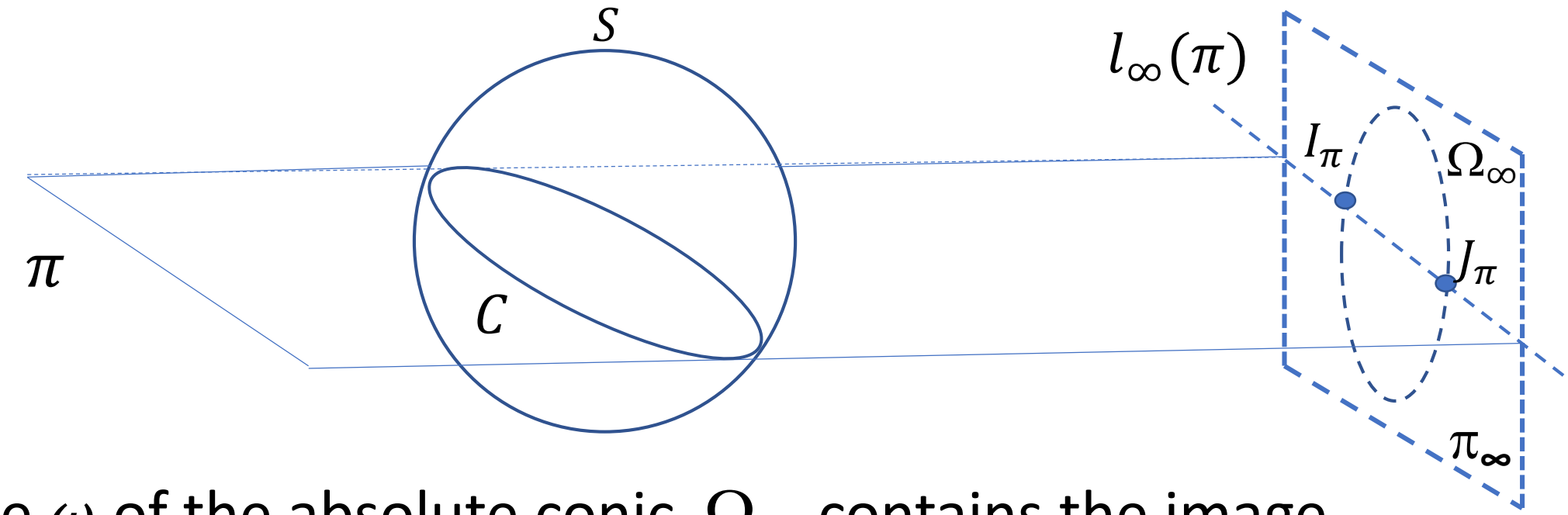


Image of the **circular points** $l'_\infty \cap \omega$ why?



remember: IAC ω contain images of circular points

1. The absolute conic Ω_∞ contains all the circular points (i.e. the circular points I_π, J_π of any plane belong to absolute conic Ω_∞) !!



2. The image ω of the absolute conic Ω_∞ contains the image I'_π, J'_π of all the circular points I_π, J_π

reconstruction of the planar scene

image of circular points: {I', J'} = $l'_\infty \cap \omega$

- Image of the circular points →Image of the conic dual to the circular points

$$C_\infty^{*\prime} = I'J'^T + J'I'^T$$

- Singular value decomposition

$$\text{svd}(C_\infty^{*\prime}) = U \begin{vmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{vmatrix} U^T = H_{rect}^{-1} C_\infty^* H_{rect}^{-T}$$

- Rectifiying homography (from svd output U)

$$H_{rect} = U^T$$

In practice …

- Image of the circular points →Image of the conic dual to the circular points

$$C_\infty^{*\prime} = I'J'^T + J'I'^T$$

- Singular value decomposition

$$\text{svd}(C_\infty^{*\prime}) = U \begin{bmatrix} s_1 & 0 & 0 \\ 0 & s_2 & 0 \\ 0 & 0 & \varepsilon \end{bmatrix} U^T$$

- Rectifiying homography (from svd output U)

$$H_{rect} = (U \begin{bmatrix} \sqrt{s_1} & 0 & 0 \\ 0 & \sqrt{s_2} & 0 \\ 0 & 0 & 1 \end{bmatrix})^{-1} = \begin{bmatrix} \sqrt{s_1{}^{-1}} & 0 & 0 \\ 0 & \sqrt{s_2{}^{-1}} & 0 \\ 0 & 0 & 1 \end{bmatrix} U^T$$

Euclidean reconstruction = Image rectification

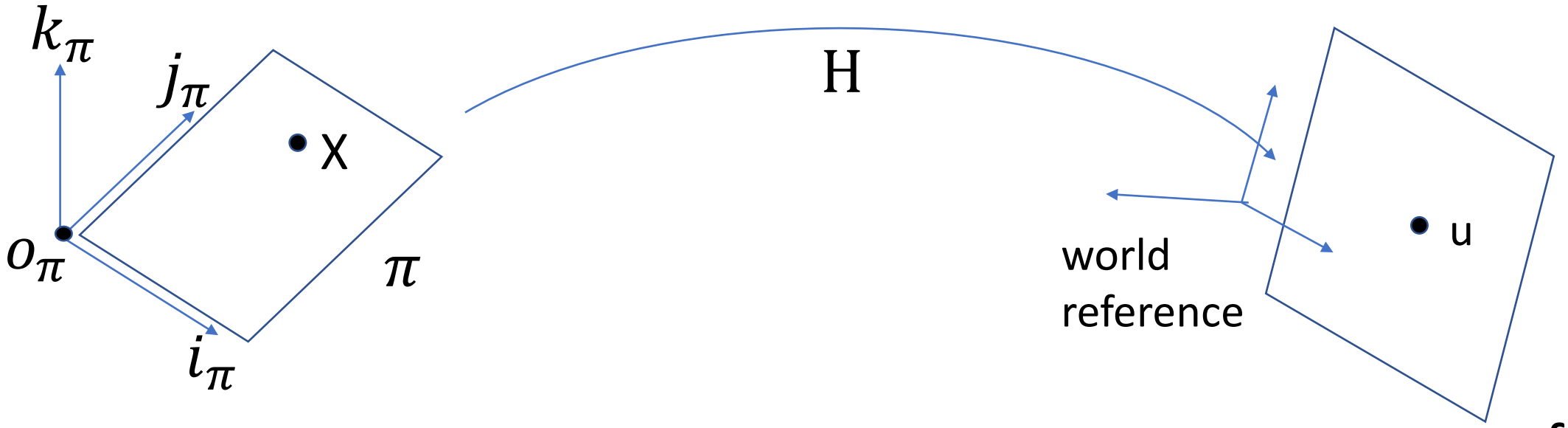
Localization of known (e.g. reconstructed) planar objects from a single calibrated image

Image of the upper face

known shape of the upper face (e.g. after reconstruction)

known planar object

image



object reference

camera reference

world reference ≡ camera reference

$$\mathbf{X}_\pi = \begin{bmatrix} x \\ y \\ 0 \\ w \end{bmatrix}$$

$$\mathbf{X}_\mathbf{w} = \begin{bmatrix} \mathrm{i}_\pi & \mathrm{j}_\pi & \mathrm{k}_\pi & \mathrm{o}_\pi \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 0 \\ w \end{bmatrix}$$

$$\mathbf{X_w} = \begin{bmatrix} \mathbf{i}_\pi & \mathbf{j}_\pi & \mathbf{k}_\pi & \mathbf{o}_\pi \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 0 \\ w \end{bmatrix} = \begin{bmatrix} \mathbf{i}_\pi & \mathbf{j}_\pi & \mathbf{o}_\pi \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ w \end{bmatrix} = \begin{bmatrix} \mathbf{i}_\pi & \mathbf{j}_\pi & \mathbf{o}_\pi \\ 0 & 0 & 1 \end{bmatrix} \mathbf{x}_{\boldsymbol{\pi}}$$

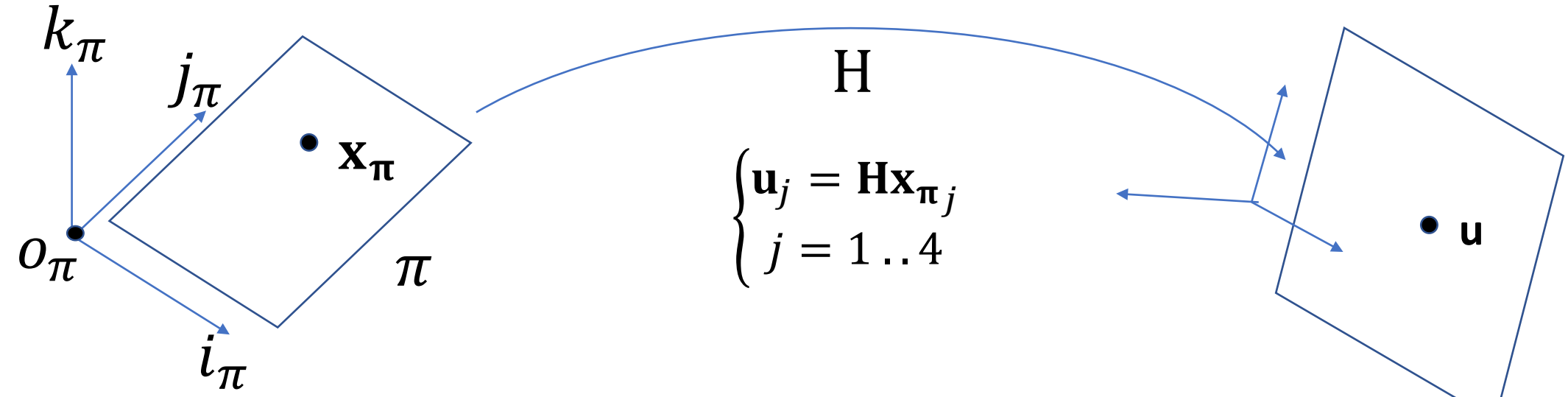
world reference on camera reference: $\mathbf{R} = \mathbf{I}_3, \mathbf{t} = \mathbf{0}$

$$\mathrm{P} = [\mathbf{KR} \quad \mathbf{KRt}] = [\mathbf{K} \quad \mathbf{0}]$$

$$\mathbf{u} = \mathrm{P}\mathbf{X_w} = [\mathbf{K} \quad \mathbf{0}] \begin{bmatrix} \mathbf{i}_\pi & \mathbf{j}_\pi & \mathbf{o}_\pi \\ 0 & 0 & 1 \end{bmatrix} \mathbf{x}_{\boldsymbol{\pi}} = \mathbf{K}[\mathbf{i}_\pi \quad \mathbf{j}_\pi \quad \mathbf{o}_\pi]\mathbf{x}_{\boldsymbol{\pi}}$$

plane π to image homography $\mathbf{H} = \mathbf{K}[\mathbf{i}_\pi \quad \mathbf{j}_\pi \quad \mathbf{o}_\pi]$

H known from image and object model
(4 pairs of corresponding points)



planar object

image

pose of the planar object relative to the camera

$$[\mathbf{i}_\pi \quad \mathbf{j}_\pi \quad \mathbf{o}_\pi] = \mathbf{K}^{-1}\mathrm{H}$$

$$(\mathbf{k}_\pi = \mathbf{i}_\pi \times \mathbf{j}_\pi)$$

Localization of a reconstructed face wrt camera

Localization from homography H
between the planar face and its image

Localization localization of the upper face

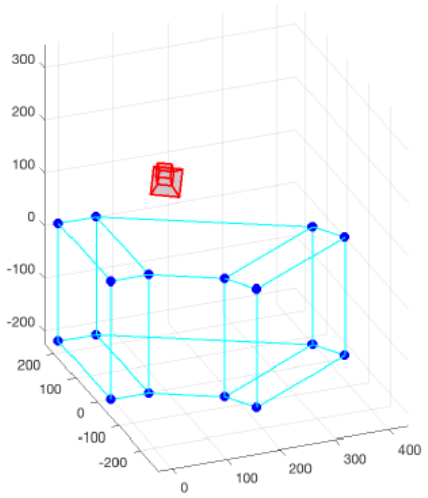
Planar object (upper planar face) wrt camera

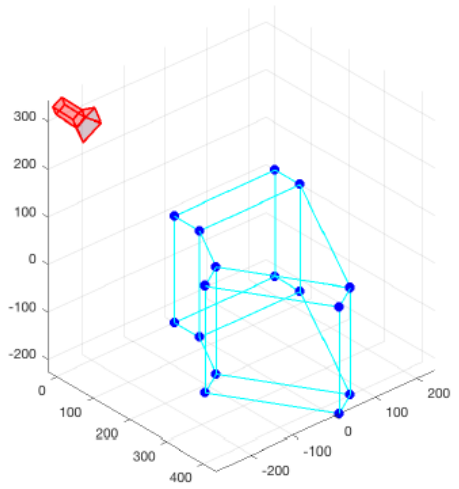
$$[i_\pi \quad j_\pi \quad o_\pi] = \mathbf{K}^{-1}\mathbf{H}$$

where $\mathbf{K}$ is the calibration matrix of the camera,
while
$\mathbf{H}$ is the homography from the planar face to its image

Localize camera wrt upper planar face:

$$\begin{bmatrix} i_\pi & j_\pi & i_\pi \times j_\pi & 0_\pi \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1}$$





$$\mathbf{u} = \mathbf{K}[\mathbf{R} \quad \mathbf{t}]\ \mathbf{X}_{world} = [\mathbf{KR} \quad \mathbf{Kt}]\ \mathbf{X}_{world}$$

remember $\mathbf{u} = \mathbf{P}\ \mathbf{X}_{world} = [\mathbf{M} \quad \mathbf{m}]\ \mathbf{X}_{world}$

$$\rightarrow \mathbf{M} = \mathbf{KR} \quad \text{and} \quad \mathbf{m} = \mathbf{Kt}$$

$$\mathbf{M} = \mathbf{KR}_{cam->world}$$

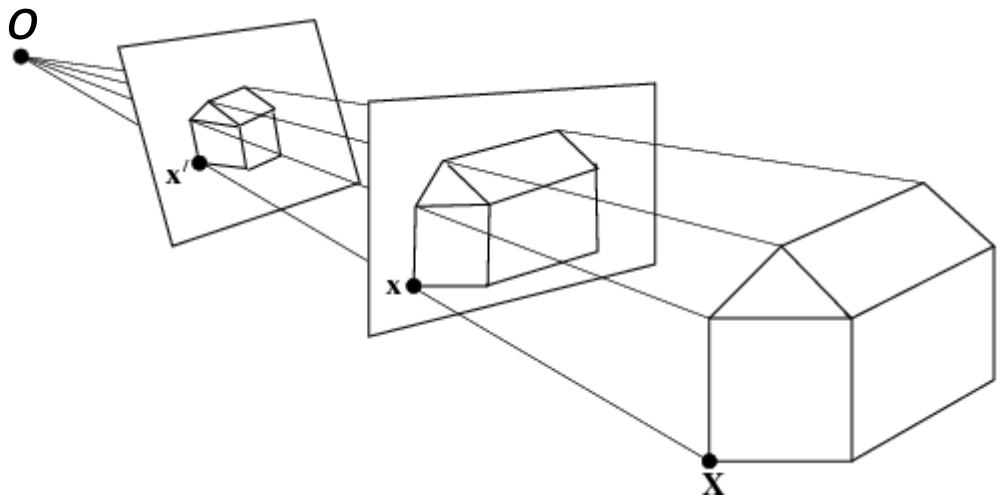
given **M**, find **K** and **R** by Q-R matrix decomposition of inv(**M**)

from $\mathbf{m} = -\mathbf{Mo} \rightarrow \mathbf{u} = [\mathbf{M} \quad \mathbf{m}]\ \mathbf{X}_{world} = \mathbf{M}[\mathbf{I} \quad -\boldsymbol{o}]\ \mathbf{X}_{world}$

$$\mathbf{u} = \mathbf{KR}[\mathbf{I} \quad -\boldsymbol{o}]\ \mathbf{X}_{world}$$

$$\rightarrow \boxed{\mathbf{P} = \mathbf{KR}[\mathbf{I} \quad -\boldsymbol{o}]}$$

Image transformations with fixed camera center



$$P = KR[I|-o] \ , P' = K'R'[I|-o]$$

$$P' = K'R'(KR)^{-1}P$$

$$x' = P'X = K'R'(KR)^{-1}PX = K'R'(KR)^{-1}x$$

$$x' = Hx \text{ with } H = K'R'(KR)^{-1} = K'R'R^{-1}K^{-1}$$

Translating the image plane: zooming (assuming fixed principal point)

$$\mathrm{x}' = \mathrm{Hx} = \mathrm{K}'\mathrm{R}'\mathrm{R}^{-1}\mathrm{K}^{-1}\mathrm{x} \text{ but, since } \mathrm{R}' = \mathrm{R}$$

$$\mathrm{x}' = \mathrm{K}'(\mathrm{K})^{-1}\mathrm{x}$$

$$\mathrm{H} = \mathrm{K}'\left(\mathrm{K}\right)^{-1} = \begin{bmatrix} \mathrm{kI} & (1-\mathrm{k})\tilde{\mathrm{x}}_0 \\ 0^{\mathrm{T}} & 1 \end{bmatrix} \qquad k = f \,/\, f'$$

$$\mathrm{K}' = \begin{bmatrix} \mathrm{kI} & (1-\mathrm{k})\tilde{\mathrm{x}}_0 \\ 0^{\mathrm{T}} & 1 \end{bmatrix} \mathrm{K}$$

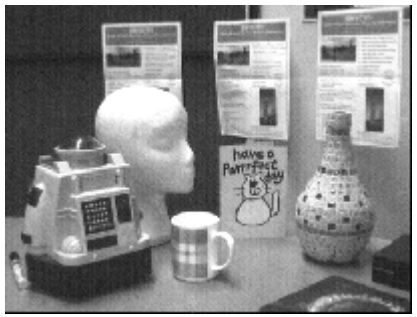
$$\rightarrow$$

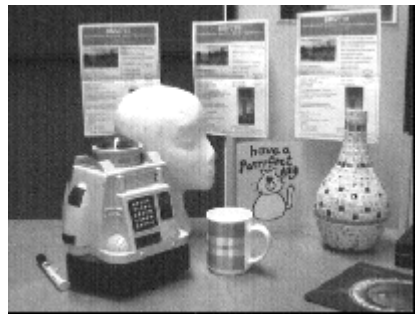
$$\mathrm{K}' = \begin{bmatrix} \mathrm{kI} & (1-\mathrm{k})\tilde{\mathrm{x}}_0 \\ 0^{\mathrm{T}} & 1 \end{bmatrix} \begin{bmatrix} \mathrm{A} & \tilde{\mathrm{x}}_0 \\ 0^{\mathrm{T}} & 1 \end{bmatrix} = \begin{bmatrix} \mathrm{kA} & \tilde{\mathrm{x}}_0 \\ 0^{\mathrm{T}} & 1 \end{bmatrix} = \mathrm{K} \begin{bmatrix} \mathrm{kI} & 0 \\ 0^{\mathrm{T}} & 1 \end{bmatrix}$$

Camera rotation around camera center

$$x = K\,[I|0]\,X$$
$$x' = K'[R|0]\,X = K'R\,[I|0]\,X = K'\,RK^{-1}K[I|0]\,X = K'RK^{-1}x$$

$$H = K'RK^{-1}$$





Camera rotation around camera center with fixed intrinsic parameters

A camera is rotated about its centre with no change in the internal parameters..
Algebraically, if x, x' are the images of a point X before and after the pure rotation

$$\mathrm{x} = \mathrm{K}\,[\mathrm{I} \mid 0]\,\mathrm{X}$$
$$\mathrm{x}' = \mathrm{K}\,[\mathrm{R} \mid 0]\,\mathrm{X} = \mathrm{KR}\,[\mathrm{I} \mid 0\,]X = \mathrm{KRK}^{-1}\,\mathrm{K}[\mathrm{I} \mid 0]\mathrm{X} = \mathrm{K\,R\,K}^{-1}\mathrm{x}$$

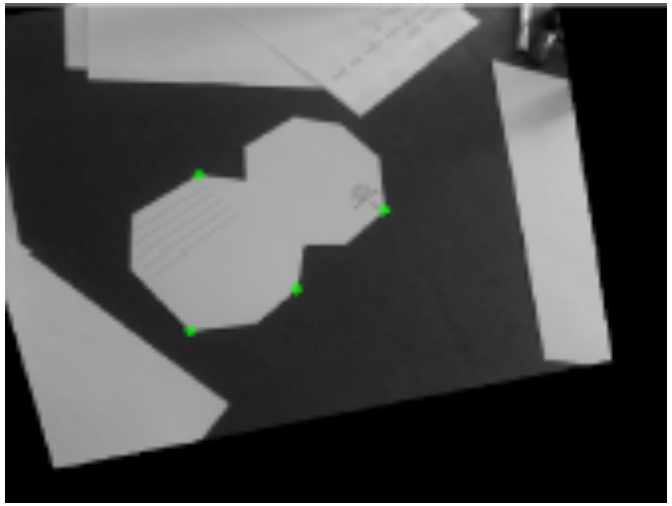
so that $\mathrm{x}' = H\,\mathrm{x}$ with $H = \mathrm{KRK}^{-1}$: conjugate rotation

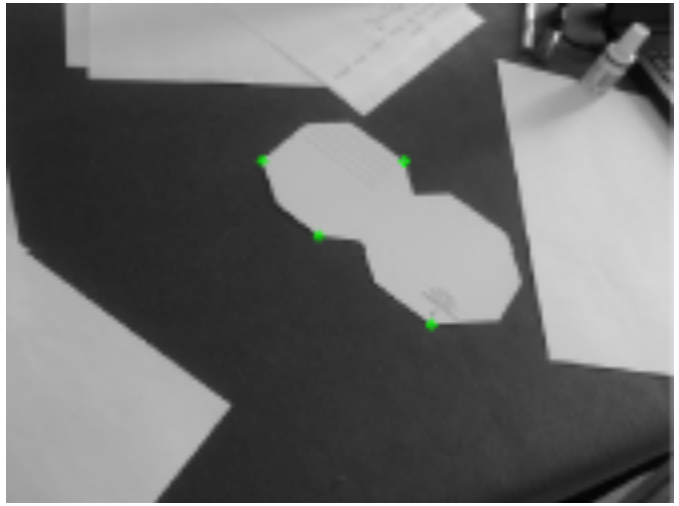
eigenvalues of $H = \mathrm{KRK}^{-1}$: $\{\mu, \mu e^{i\theta}, \mu e^{-i\theta}\}$ = the same eigenvalues of R

where μ is an unknown scale factor (if H scaled such that $\det H = 1$, then $\mu = 1$).
Hence, the rotation angle between views may be computed directly from the phase of the complex eigenvalues of H.
Similarly, it can be shown (exercise) that the eigenvector of H corresponding to the real eigenvalue is the vanishing point of the rotation axis.

Reminds to planar motion

- find eigenvector-eigenvalues of H :
- eigenvalues are proportional to $\lambda' = 1, \lambda'' = e^{i\theta}, \lambda'' = e^{-i\theta}$
- eigenvector e' associated to $\lambda' = 1$ is the image of the C.O.R. $\boldsymbol{O}$
- angle θ is the rotation angle

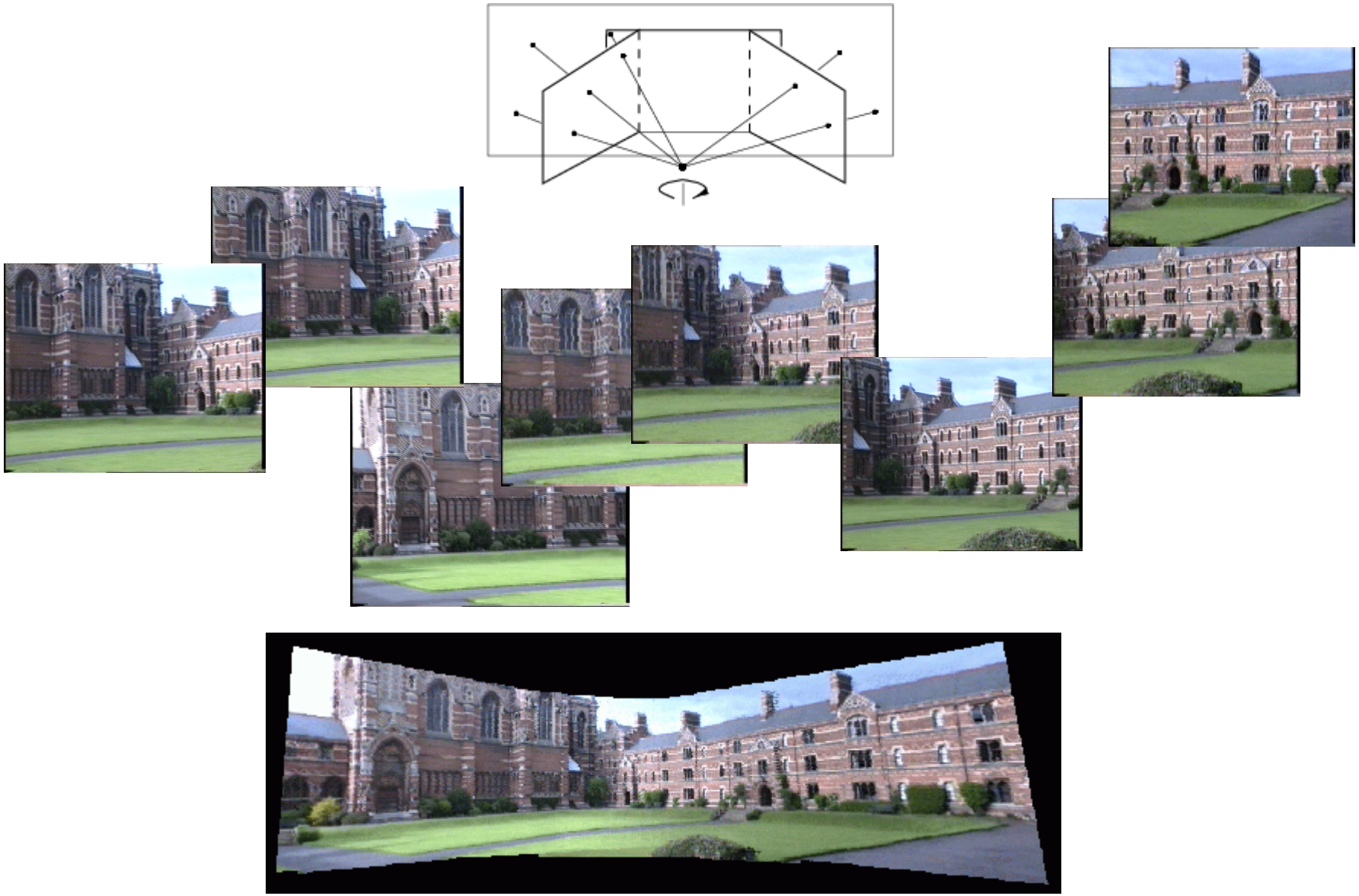




Common aspects and differences between them

- in planar motion, rotation does not need to be about the camera center
- in planar motion, rotation axis is perpendicular to the plane

Planar homography mosaicing



Planar homography mosaicing

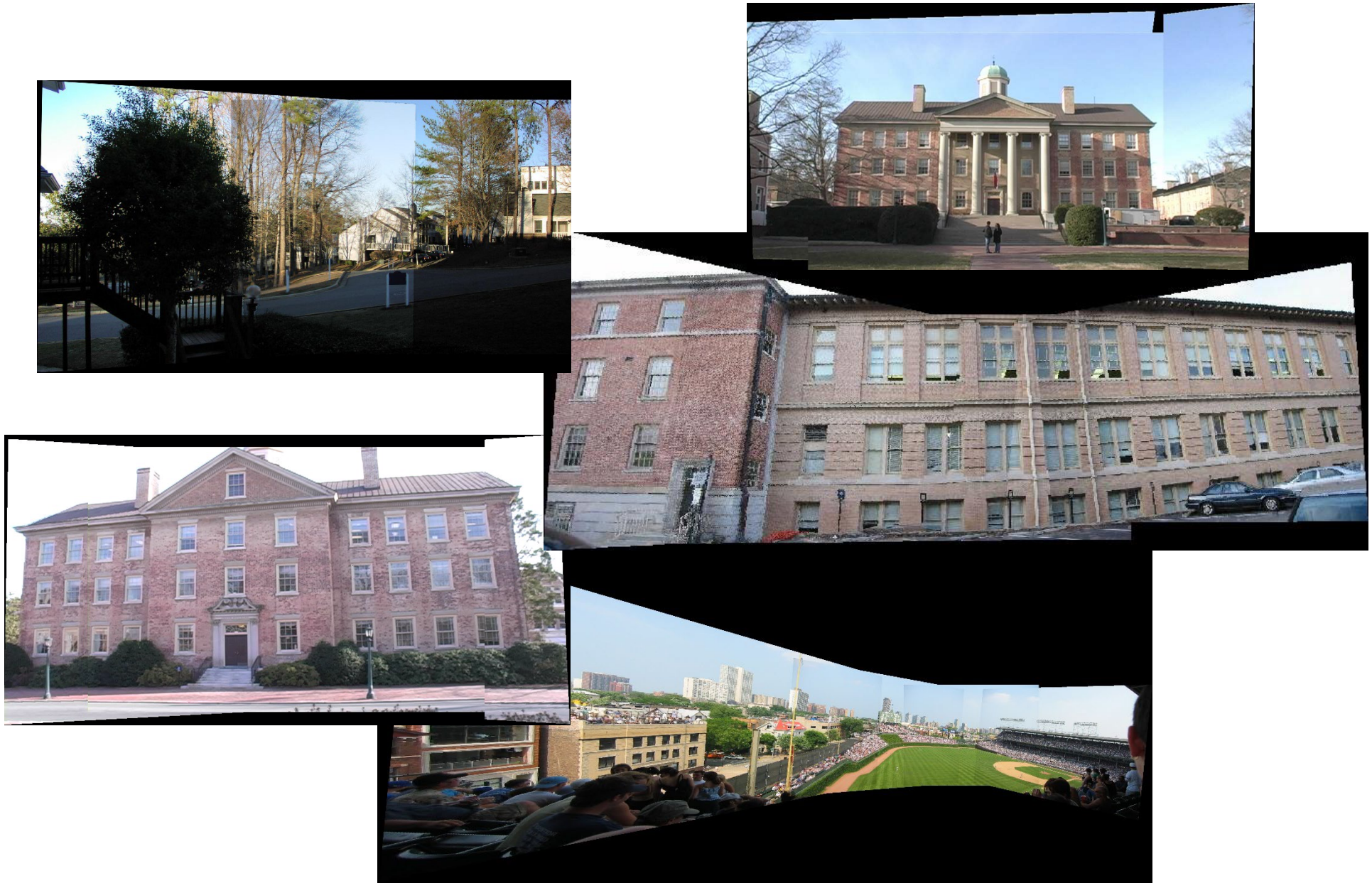
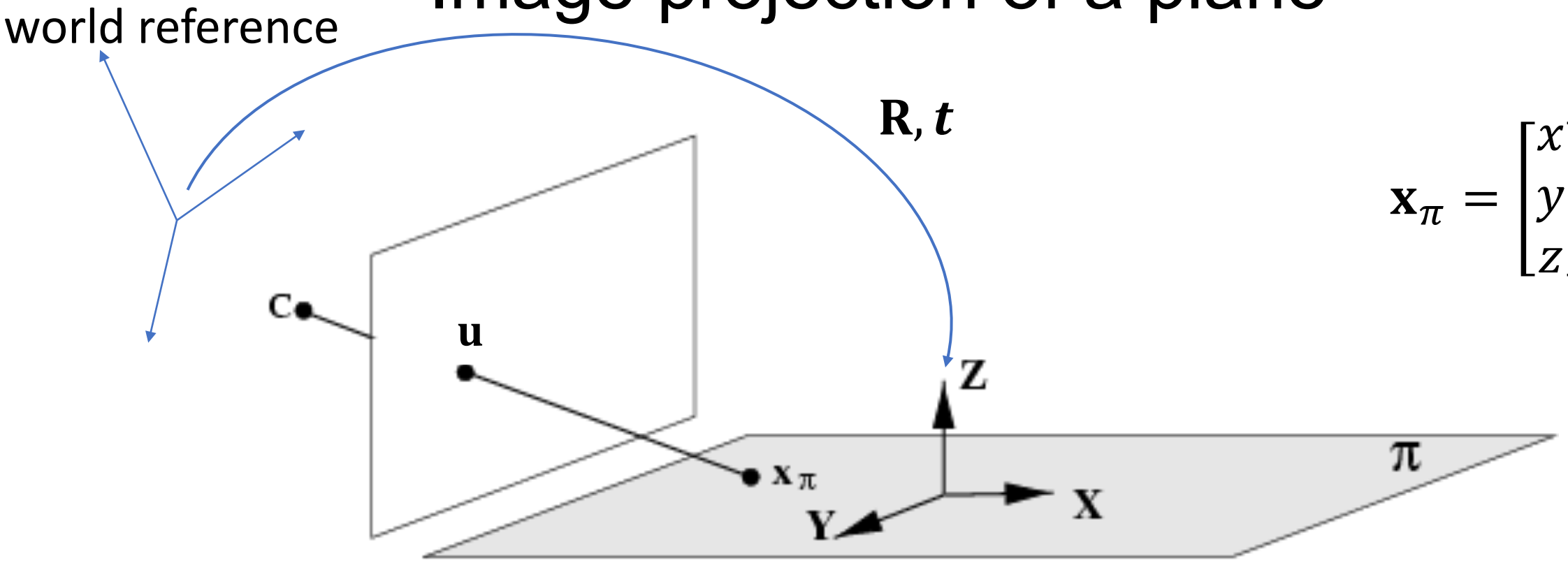


Image: projection and back-projection

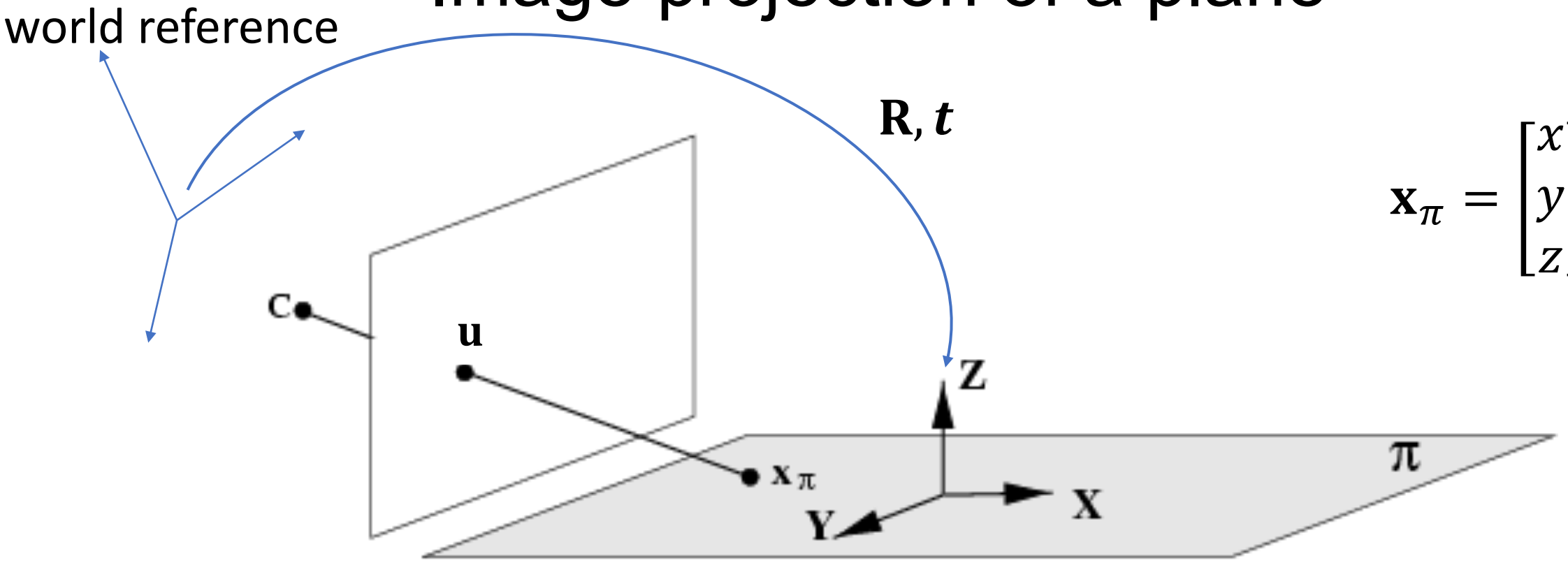
Image projection of a plane



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

$$X_w = \begin{bmatrix} \mathbf{R} & \boldsymbol{t} \\ 0 & 1 \end{bmatrix} X_\pi$$

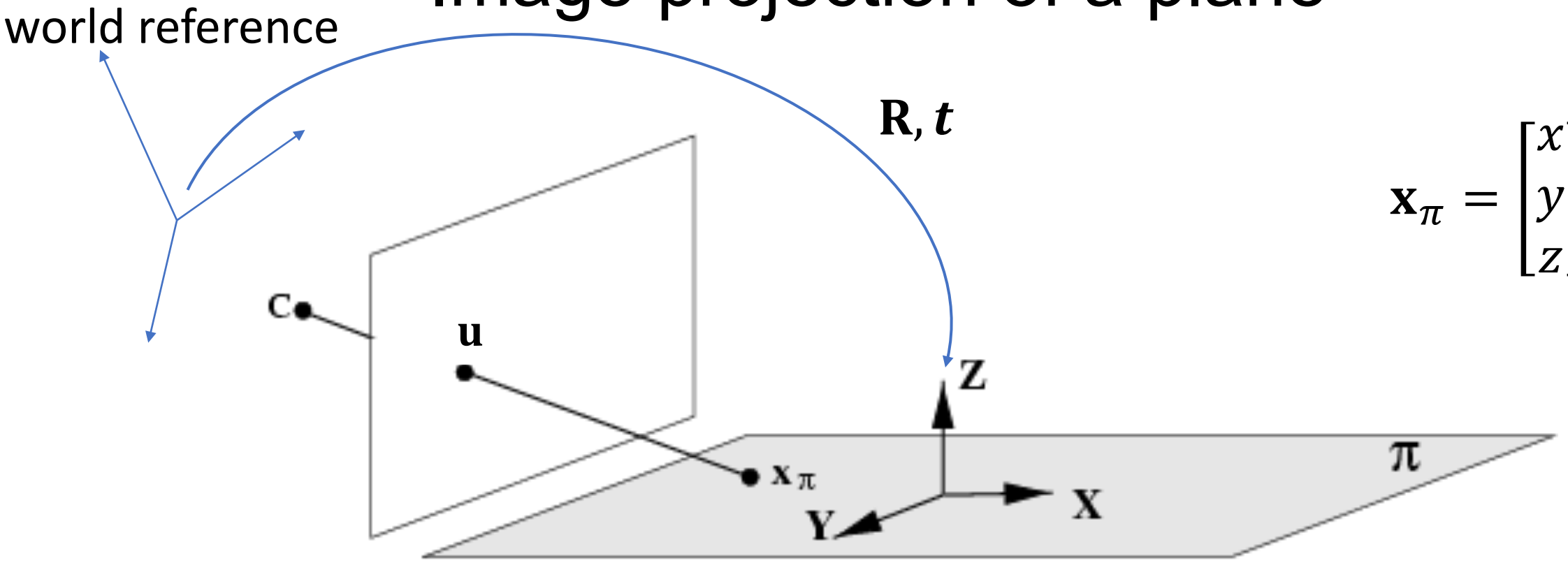
Image projection of a plane



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

$$\mathbf{u} = \mathbf{P}X_w = [\mathbf{M} \quad \mathbf{m}]X_w = [\mathbf{M} \quad \mathbf{m}]\begin{bmatrix} \mathbf{R} & \boldsymbol{t} \\ 0 & 1 \end{bmatrix} X_\pi = [\mathbf{MR} \quad \mathbf{Mt} + \mathbf{m}]X_\pi$$

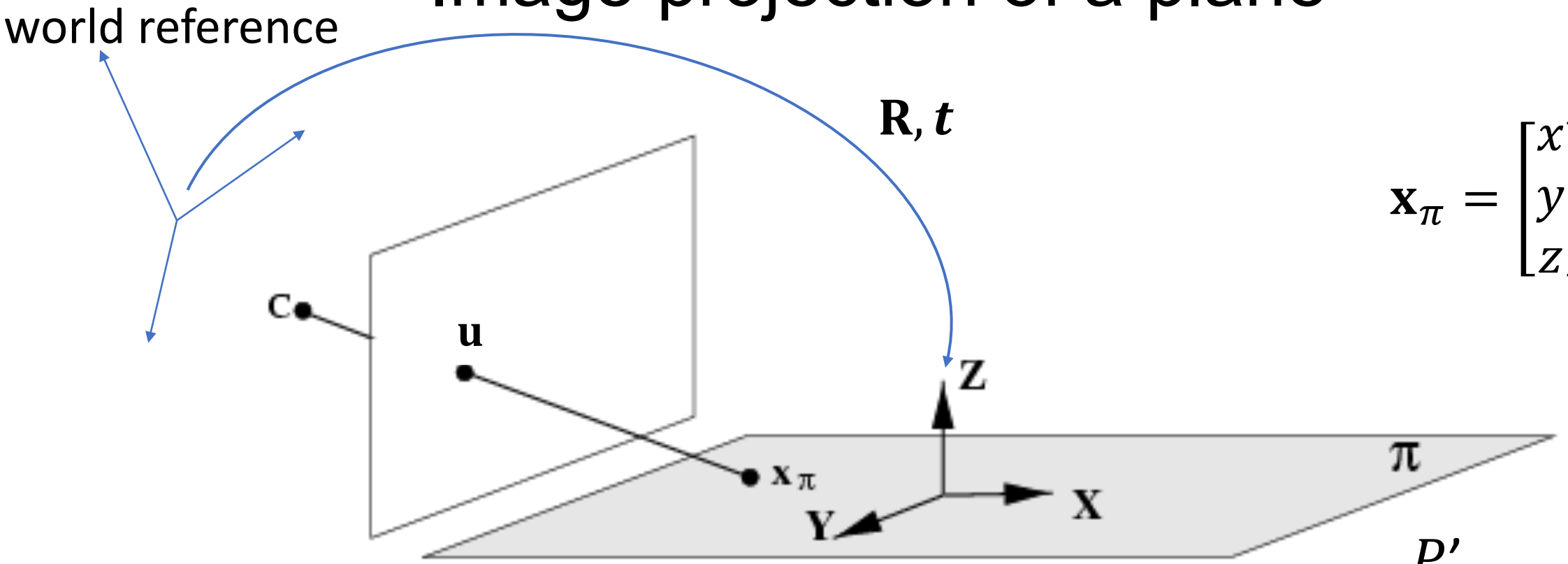
Image projection of a plane



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

$$\mathbf{u} = \mathbf{P}X_w = [\mathbf{M} \quad \mathbf{m}]X_w = [\mathbf{M} \quad \mathbf{m}]\begin{bmatrix} \mathbf{R} & \boldsymbol{t} \\ 0 & 1 \end{bmatrix} X_\pi = \underbrace{[\mathbf{MR} \quad \mathbf{Mt} + \mathbf{m}]}_{\mathbf{P}'} \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{y} \\ \mathbf{0} \\ \boldsymbol{w} \end{bmatrix}$$

Image projection of a plane



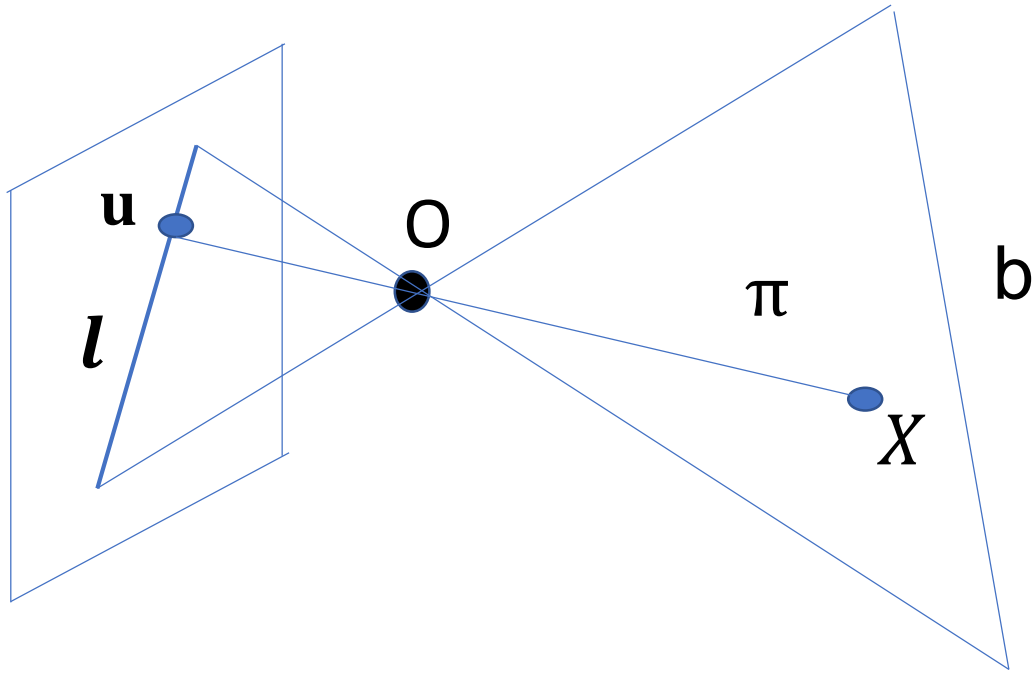
$$\mathbf{x}_\pi = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

$$\mathbf{u} = PX_w = [\mathbf{M} \quad \mathbf{m}]X_w = [\mathbf{M} \quad \mathbf{m}]\begin{bmatrix} \mathbf{R} & \boldsymbol{t} \\ 0 & 1 \end{bmatrix} X_\pi = \overbrace{[\mathbf{MR} \quad \mathbf{Mt} + \mathbf{m}]}^{P'} \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{y} \\ \mathbf{0} \\ \boldsymbol{w} \end{bmatrix}$$

$$= [\mathrm{p'}_1 \quad \mathrm{p'}_2 \quad \mathrm{p'}_3 \quad \mathrm{p'}_4][\boldsymbol{x} \quad \boldsymbol{y} \quad \mathbf{0} \quad \boldsymbol{w}]^T = [\mathrm{p'}_1 \quad \mathrm{p'}_2 \quad \mathrm{p'}_4]\mathbf{x}_\pi$$

Back-projection of an image line

set of space points X, whose image projection $\mathbf{u} = \mathbf{P}X$ is on image line $\boldsymbol{l}$



back-projection

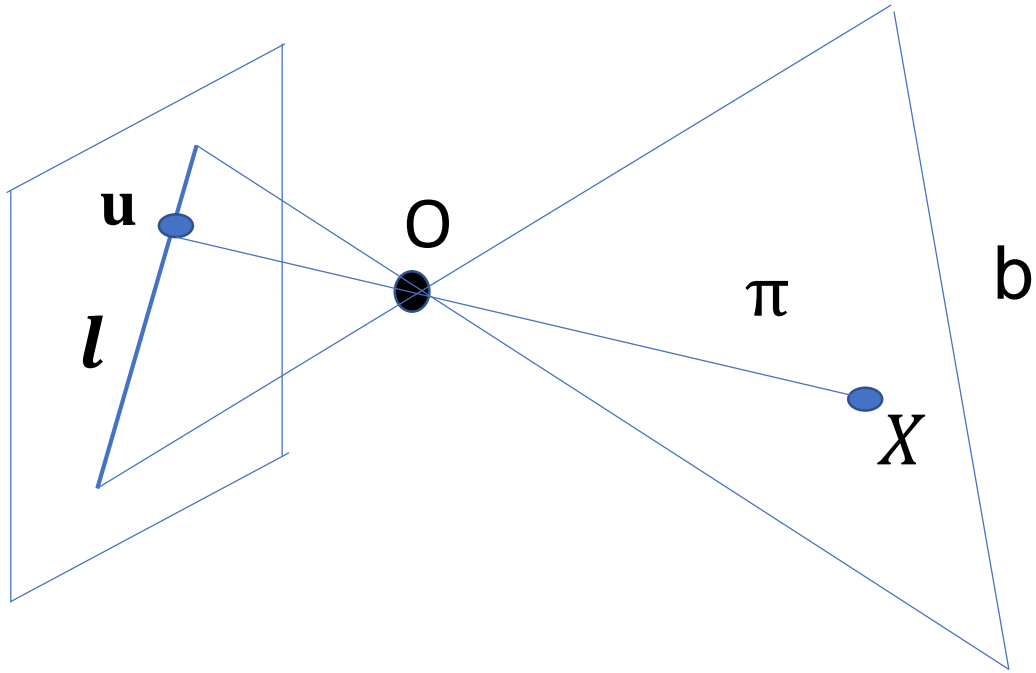
$$l^{\mathrm{T}}\mathbf{u} = l^{\mathrm{T}}\mathbf{P}X = 0$$

$$\boldsymbol{\pi}^{T}\mathrm{X} = 0$$

back-projection of $\boldsymbol{l}$ is plane $\boldsymbol{\pi} = \mathbf{P}^{\mathrm{T}}\boldsymbol{l}$

Back-projection of an image line

set of space points X, whose image projection $\mathbf{u} = \mathbf{P}X$ is on image line $\boldsymbol{l}$



back-projection

$$l^{\mathrm{T}}\mathbf{u} = l^{\mathrm{T}}\mathbf{P}X = 0$$

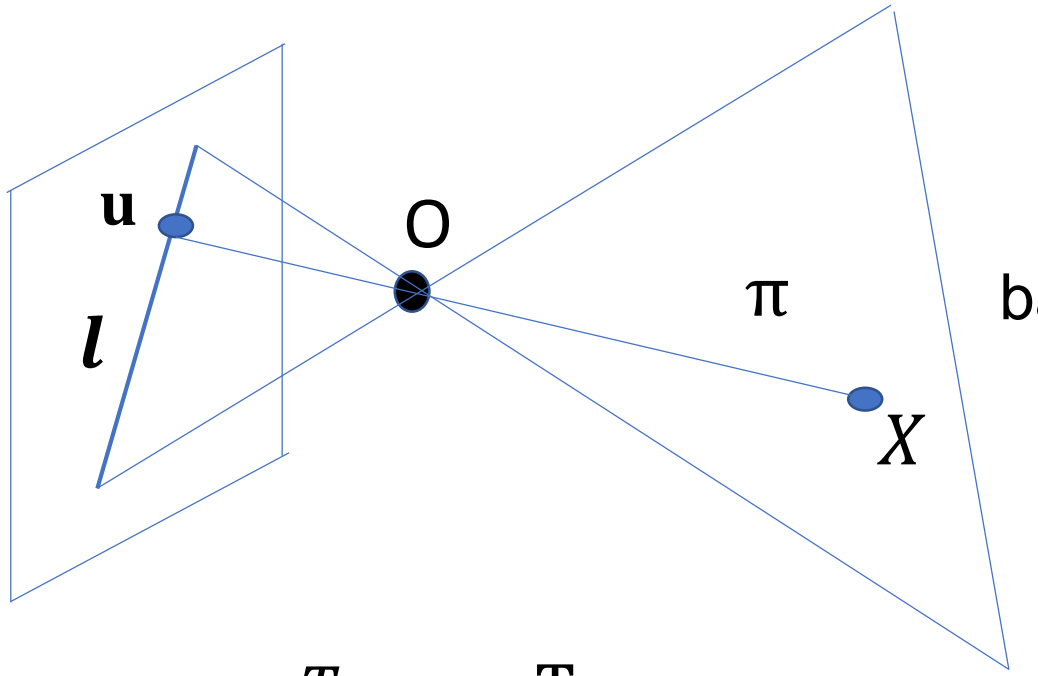
$$\boldsymbol{\pi}^{T}\mathrm{X} = 0$$

back-projection of $\boldsymbol{l}$ is plane $\boldsymbol{\pi} = \mathbf{P}^{\mathrm{T}}\boldsymbol{l}$ through O, in fact, since $\mathrm{O} = RNS(\mathbf{P})$

$$\boldsymbol{\pi}^{T}\mathrm{O} = \boldsymbol{l}^{\mathrm{T}}\mathbf{P}\,\mathrm{O} = \boldsymbol{l}^{\mathrm{T}}0 = 0$$

Back-projection of an image line

set of space points X, whose image projection is on image line $\boldsymbol{l}$



back-projection of $\boldsymbol{l}$ is $\boldsymbol{\pi} = \boldsymbol{P}^{\mathrm{T}}\boldsymbol{l}$

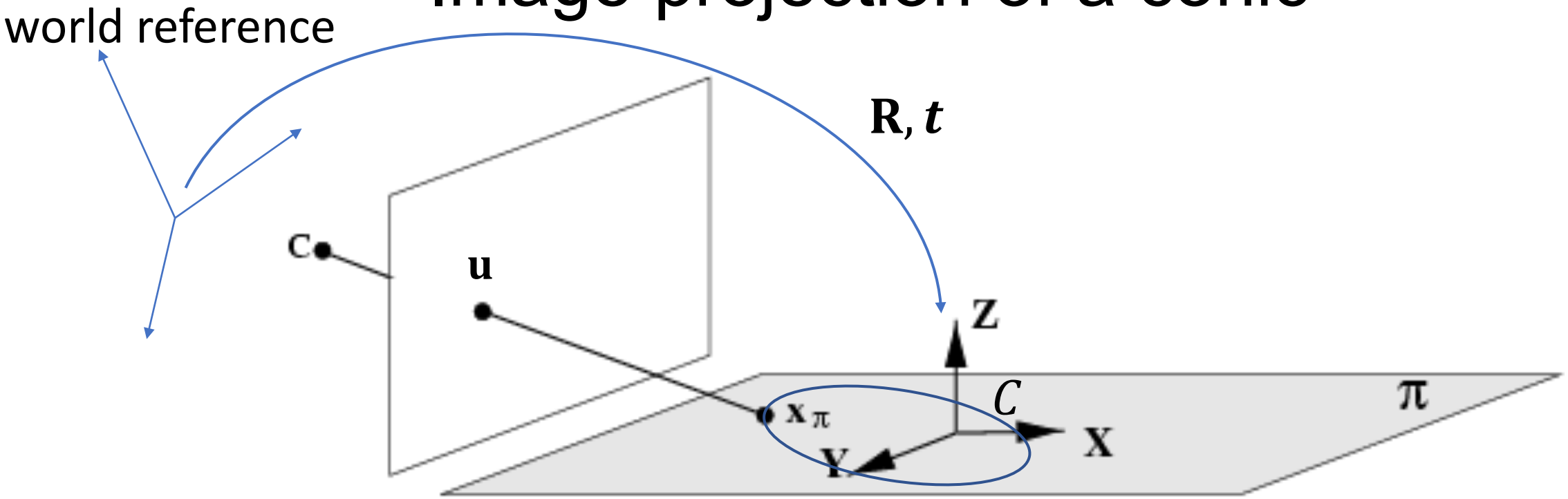
$$\boldsymbol{\pi}^{T}\mathrm{X} = l^{\mathrm{T}}PX = 0$$

try with $X = O = RNS(P)$:

$$\boldsymbol{\pi}^{T}\mathrm{O} = l^{\mathrm{T}}P\mathrm{O} = l^{\mathrm{T}}0 = 0$$

The backprojection of $\boldsymbol{l}$ is a plane $\boldsymbol{\pi} = \boldsymbol{P}^{\mathrm{T}}\boldsymbol{l}$ through O

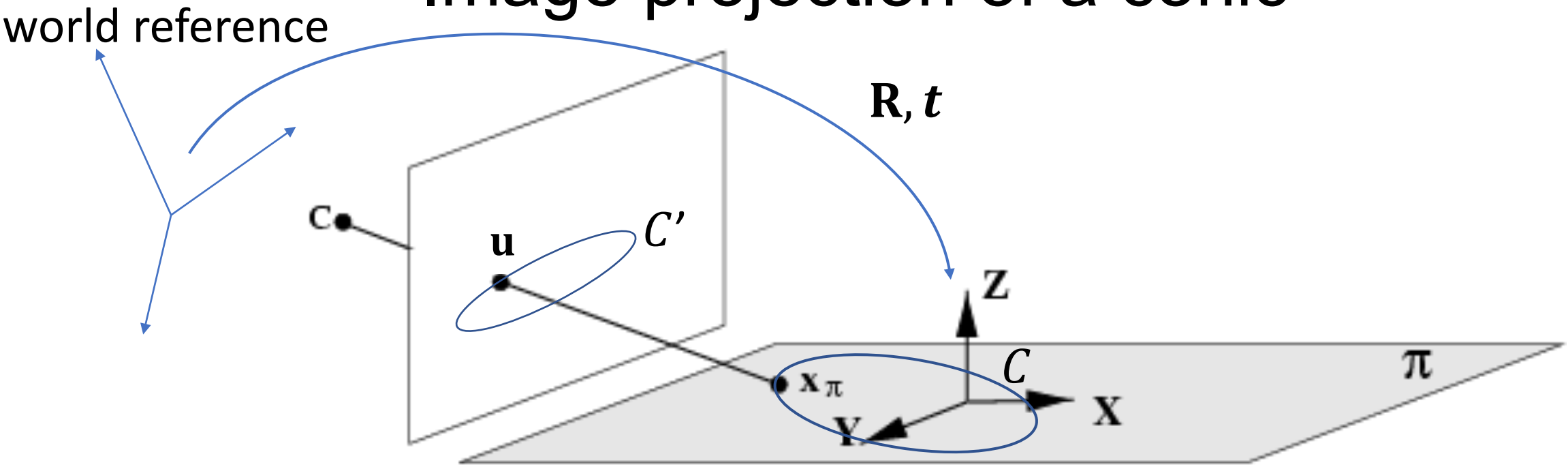
Image projection of a conic



$$\mathbf{u} = PX_w = [\mathbf{M} \quad \mathbf{m}]X_w = [\mathbf{M} \quad \mathbf{m}]\begin{bmatrix}\mathbf{R} & \boldsymbol{t} \\ 0 & 1\end{bmatrix}X_\pi = [\underbrace{\mathbf{MR} \quad \mathbf{Mt} + \mathbf{m}}_{P'}]\begin{bmatrix}\boldsymbol{x} \\ \boldsymbol{y} \\ \mathbf{0} \\ \boldsymbol{w}\end{bmatrix}$$

$$= [\mathrm{p'_1} \quad \mathrm{p'_2} \quad \mathrm{p'_3} \quad \mathrm{p'_4}][\boldsymbol{x} \quad \boldsymbol{y} \quad \mathbf{0} \quad \boldsymbol{w}]^T = [\mathrm{p'_1} \quad \mathrm{p'_2} \quad \mathrm{p'_4}]\mathbf{x}_\pi = P'_\pi \mathbf{x}_\pi$$

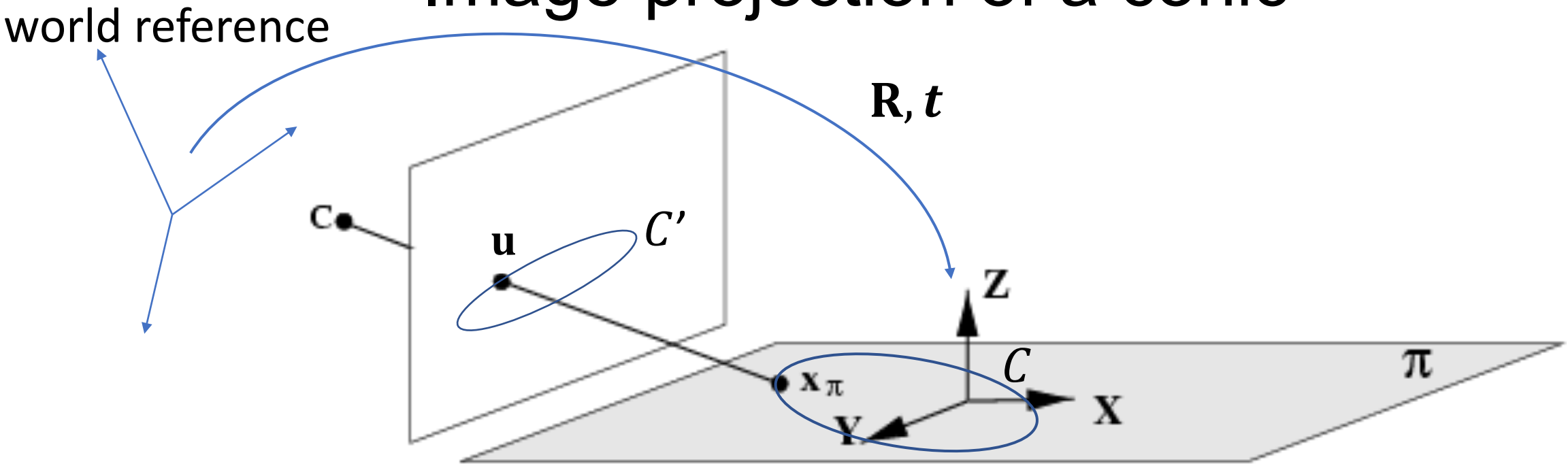
Image projection of a conic



$$\mathbf{x_\pi}^T \boldsymbol{C}\ \mathbf{x_\pi} = 0$$

$$\mathbf{u} = [\mathrm{p'}_1 \quad \mathrm{p'}_2 \quad \mathrm{p'}_4]\ \mathbf{x_\pi} = {P'}_\pi\, \mathbf{x_\pi} \rightarrow\ \mathbf{x_\pi} = {P'}_\pi^{-1}\mathbf{u}$$

Image projection of a conic



$$\mathbf{x_\pi}^T \boldsymbol{C}\ \mathbf{x_\pi} = 0$$

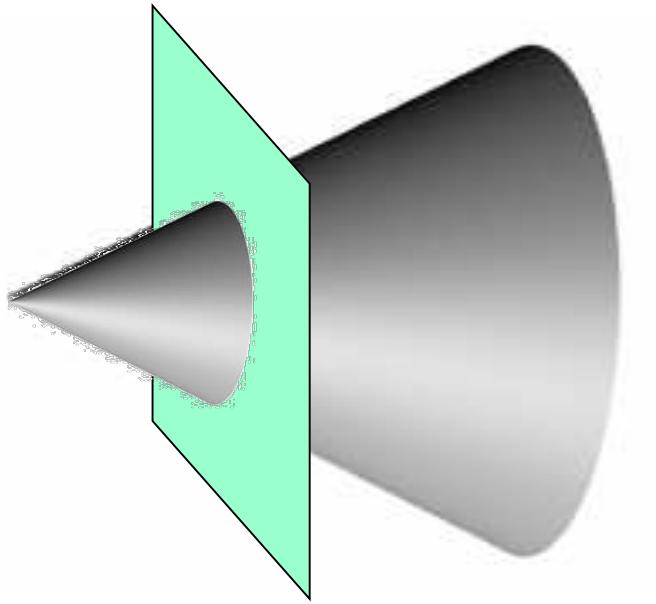
$$\mathbf{u} = [\mathrm{p'}_1 \quad \mathrm{p'}_2 \quad \mathrm{p'}_4]\ \mathbf{x_\pi} = {P'}_\pi\, \mathbf{x_\pi} \rightarrow\ \mathbf{x_\pi} = {P'}_\pi^{-1}\mathbf{u}$$

$$\rightarrow$$

$$\mathbf{u}^T {P'}_\pi^{-T}\, \boldsymbol{C}\, {P'}_\pi^{-1}\mathbf{u} = 0$$

$$\mathbf{u}^T \boldsymbol{C'}\, \mathbf{u} = 0 \text{ with } \boldsymbol{C'} = {P'}_\pi^{-T}\, \boldsymbol{C}\, {P'}_\pi^{-1}$$

Back-projection of an image conic



$$\mathrm{u} = PX$$

$$\mathrm{u}^{\mathrm{T}}\mathrm{C}\mathrm{u} = X^{\mathrm{T}}P^{\mathrm{T}}\mathrm{C}PX = X^{\mathrm{T}}\mathrm{Q}_{\mathrm{co}}X = 0$$

back-projection of an image conic is a rank-3 quadric, $\mathrm{Q}_{\mathrm{co}} = P^{\mathrm{T}}\mathrm{C}\,P$

→ a **cone**

example: $$\mathrm{Q}_{\mathrm{co}} = \begin{bmatrix} \mathrm{K}^{\mathrm{T}} \\ 0 \end{bmatrix} \mathrm{C}^{\mathrm{T}} \begin{bmatrix} \mathrm{K} \mid 0 \end{bmatrix} = \begin{bmatrix} \mathrm{K}^{\mathrm{T}}\mathrm{C}\mathrm{K} & 0 \\ 0 & 0 \end{bmatrix}$$

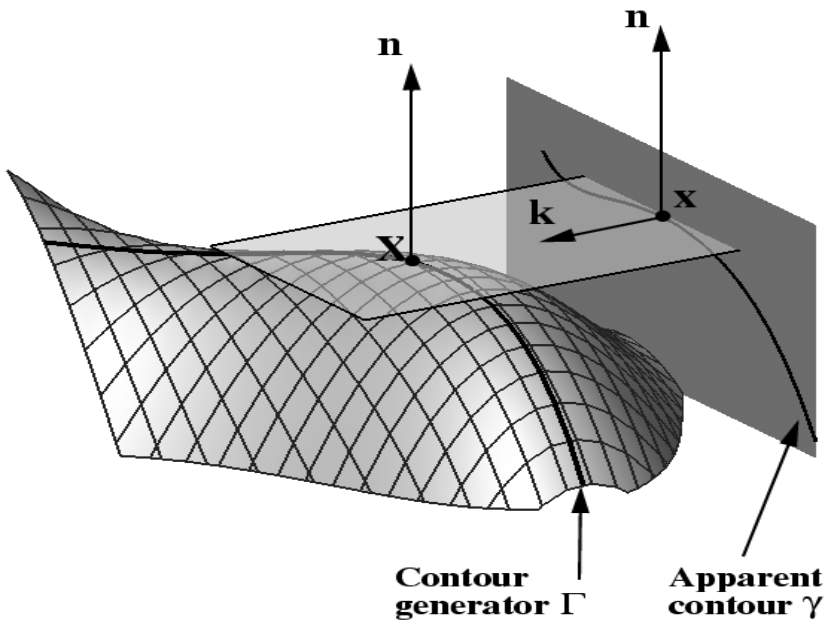
Apparent contour of a smooth surface

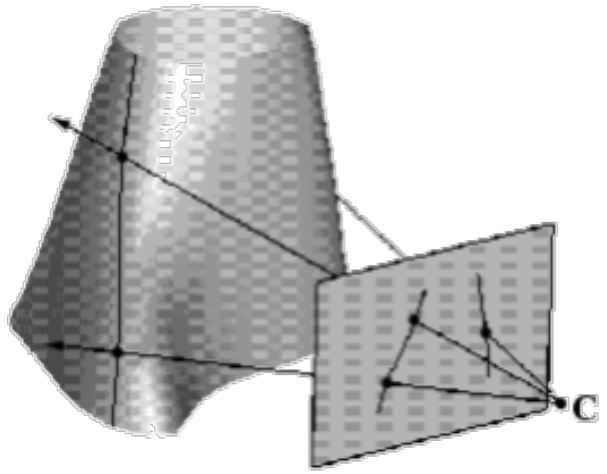
Image projection of smooth surfaces

The **contour generator** Γ is the set of points X on a surface S, whose viewing rays are tangent to S.

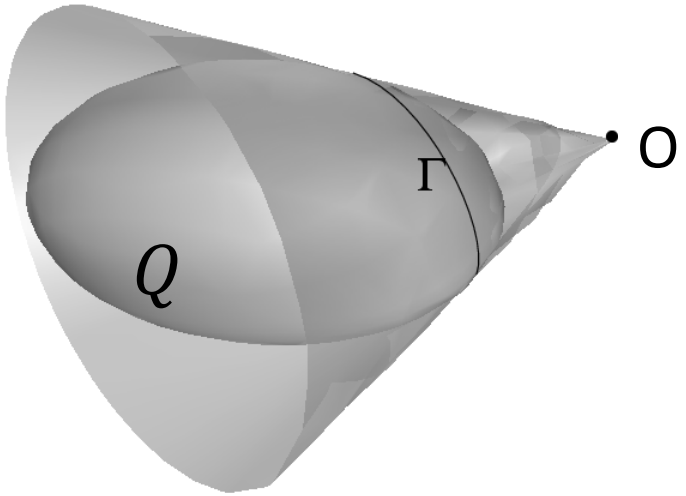
The corresponding **apparent contour** is the image of Γ

Γ depends only on position of projection center O,
Γ depends also on rest of the projection matrix P



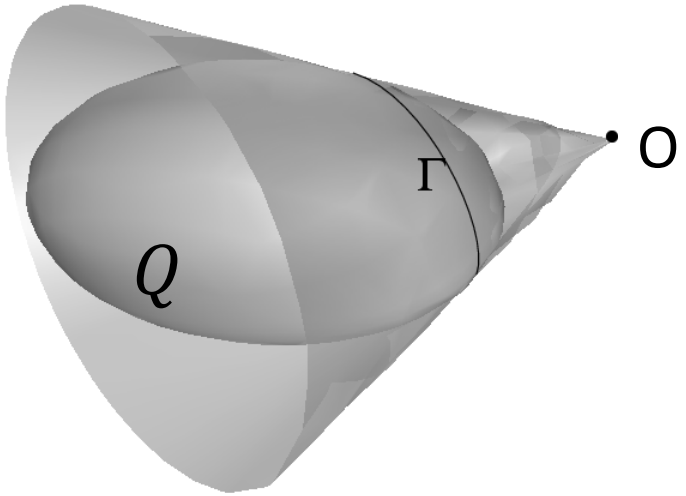


The apparent contour of a quadric



select those planes, tangent to the quadric Q, which go through camera center O

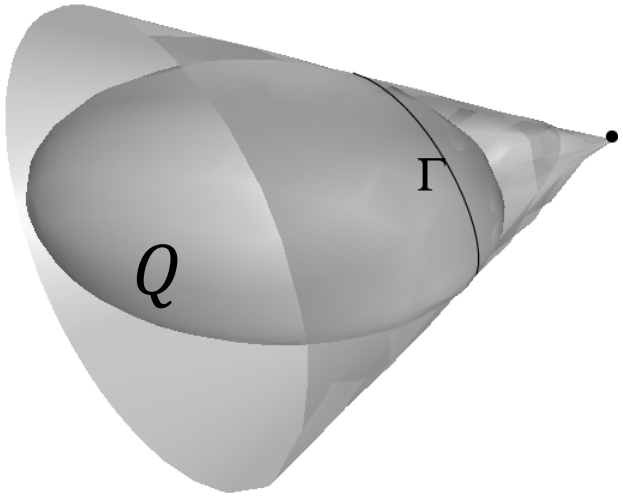
The apparent contour of a quadric



select those planes, tangent to the quadric Q,
i.e. belonging to dual quadric $Q^* = Q^{-1}$
which go through camera center O

$$\Pi^{\mathrm{T}} \mathrm{Q}^* \Pi = 0$$

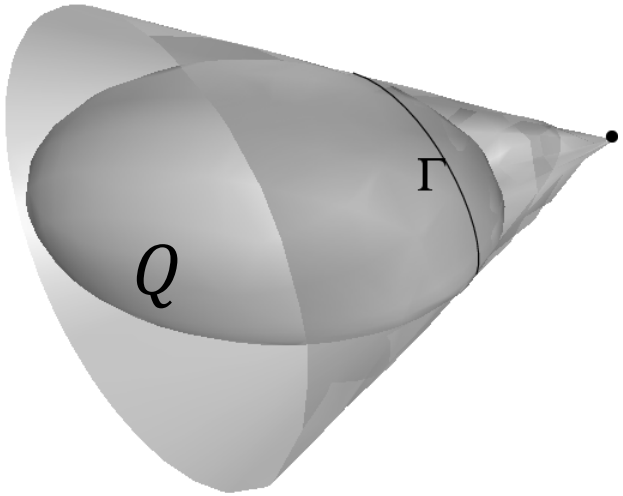
The apparent contour of a quadric



select those planes, tangent to the quadric Q,
i.e. belonging to dual quadric $Q^* = Q^{-1}$
which go through camera center O
.e., that are backprojections $\mathrm{P}^{\mathrm{T}}\boldsymbol{l}$ of some image lines $\boldsymbol{l}$

$$\Pi^{\mathrm{T}}\mathrm{Q}^*\Pi = l^{\mathrm{T}}\mathrm{P}\mathrm{Q}^*\mathrm{P}^{\mathrm{T}}l = 0$$

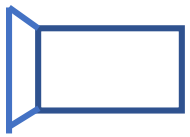
The apparent contour of a quadric



select those planes, tangent to the quadric Q,
i.e. belonging to dual quadric $Q^* = Q^{-1}$
which go through camera center O
.e., that are backprojections $\mathrm{P}^{\mathrm{T}}\boldsymbol{l}$ of some image lines $\boldsymbol{l}$

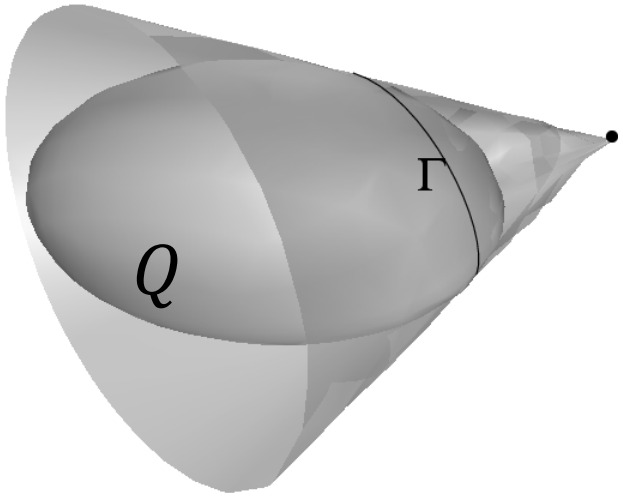
$$\Pi^{\mathrm{T}}\mathrm{Q}^*\Pi = l^{\mathrm{T}}\mathrm{P}\mathrm{Q}^*\mathrm{P}^{\mathrm{T}}l = 0$$

these planes Π are tangent
to the contour generator Γ



these lines $\boldsymbol{l}$ are image of planes Π
→ $\boldsymbol{l}$ are tangent to image of Γ
→ $\boldsymbol{l}$ are tangent to apparent contour γ

The apparent contour of a quadric

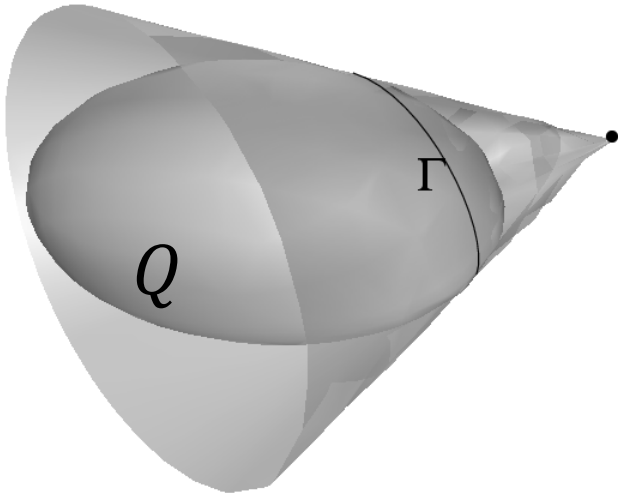


select those planes, tangent to the quadric Q,
i.e. belonging to dual quadric $Q^* = Q^{-1}$
which go through camera center O
i.e., that are backprojections $\mathrm{P}^{\mathrm{T}}\boldsymbol{l}$ of some image lines $\boldsymbol{l}$

$$\Pi^{\mathrm{T}}\mathrm{Q}^*\Pi = l^{\mathrm{T}}\mathrm{P}\mathrm{Q}^*\mathrm{P}^{\mathrm{T}}l = 0$$

these image lines $\boldsymbol{l}$ satisfy a quadratic equation
→ they belong to a dual conic $\mathrm{C}^* = \mathrm{P}\mathrm{Q}^*\mathrm{P}^T$

The apparent contour of a quadric

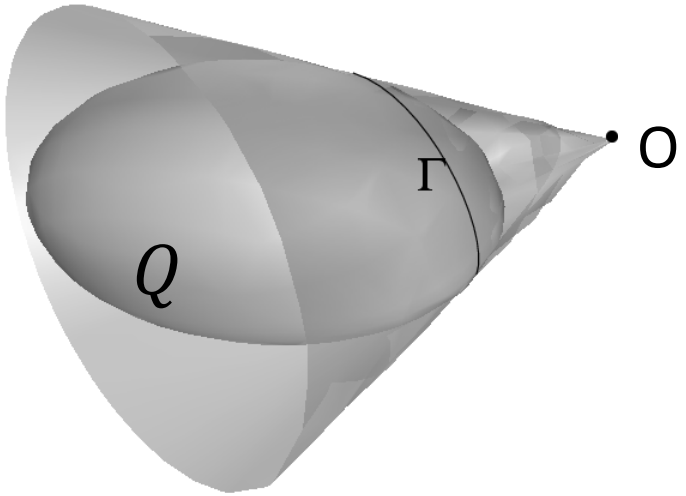


select those planes, tangent to the quadric Q,
i.e. belonging to dual quadric $Q^* = Q^{-1}$
which go through camera center O
i.e., that are backprojections of some image lines $\boldsymbol{l}$

$$\Pi^{\mathrm{T}} \mathrm{Q}^* \Pi = l^{\mathrm{T}} \mathrm{P} \mathrm{Q}^* \mathrm{P}^{\mathrm{T}} l = 0$$

these image lines $\boldsymbol{l}$ satisfy a quadratic equation
→ they belong to a dual conic $\mathrm{C}^* = \mathrm{P} \mathrm{Q}^* \mathrm{P}^T$
→ they are tangent to a conic $\mathrm{C} = \mathrm{C}^{*-1} = (\mathrm{P} \mathrm{Q}^* \mathrm{P}^T)^{-1}$

The apparent contour of a quadric



select those planes, tangent to the quadric Q,
i.e. belonging to dual quadric $Q^* = Q^{-1}$
which go through camera center O
i.e., that are backprojections of some image lines $\boldsymbol{l}$

$$\Pi^{\mathrm{T}} \mathrm{Q}^* \Pi = l^{\mathrm{T}} \mathrm{P} \mathrm{Q}^* \mathrm{P}^{\mathrm{T}} l = 0$$

these image lines $\boldsymbol{l}$ satisfy a quadratic equation
→ they belong to a dual conic $\mathrm{C}^* = \mathrm{P} \mathrm{Q}^* \mathrm{P}^T$
→ they are tangent to a conic $\mathrm{C} = \mathrm{C}^{*-1} = (\mathrm{P} \mathrm{Q}^* \mathrm{P}^T)^{-1}$

But $\boldsymbol{l}$ are tangent to conic $\mathrm{C} = \mathrm{C}^{*-1} = (\mathrm{P} \mathrm{Q}^* \mathrm{P}^T)^{-1}$ → a.c γ is the conic C

Exercise: apparent contour of a cone?

Exercise: apparent contour of a cone?

- contour generator Γ: set of tangency points from O
→ two straight lines through the vertex V

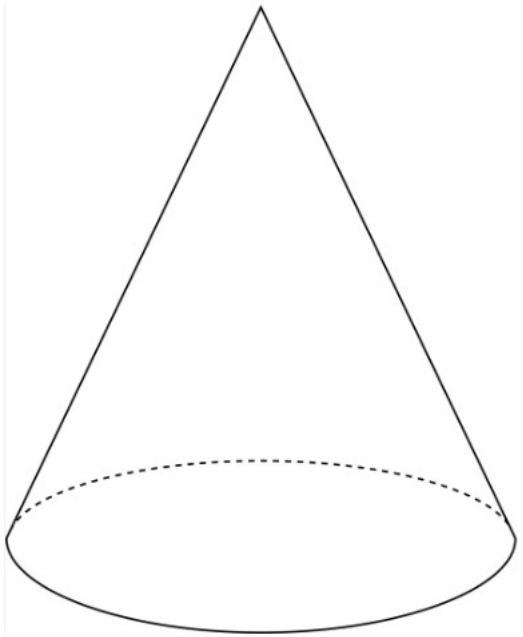
- apparent contour γ : image of Γ
→ two image lines: i.e, a degenerate conic

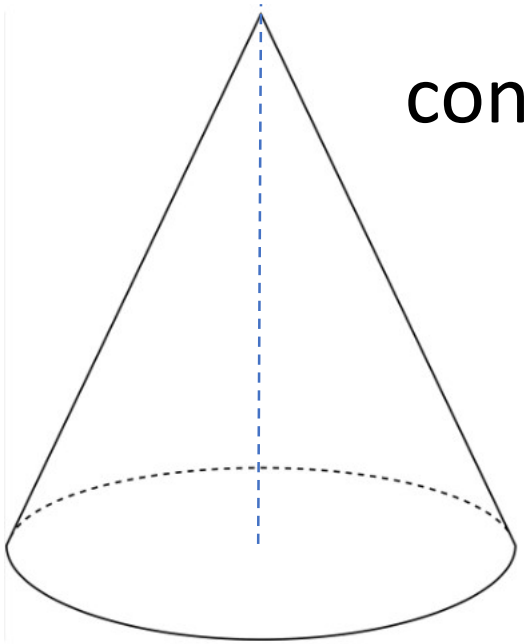
Example: contour generator of a **right** cone?

- contour generator Γ: set of tangency points from O
→ two straight lines through the vertex V

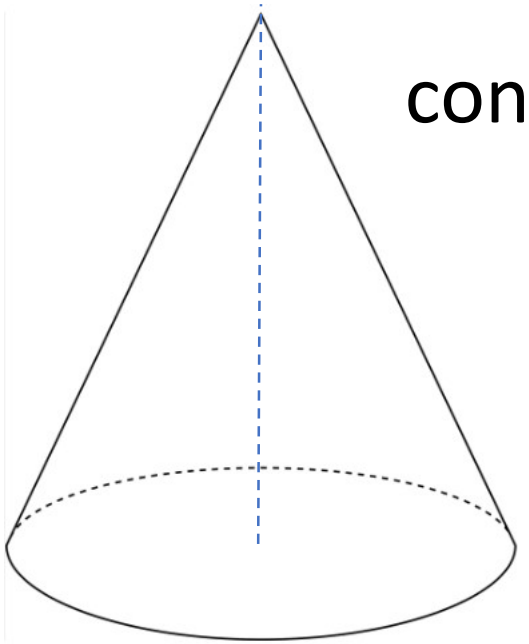
- **right** cone alone is **symmetric** wrt to its axis
→ right cone + viewpoint O : «less» symmetry,
which symmetry?

a right cone



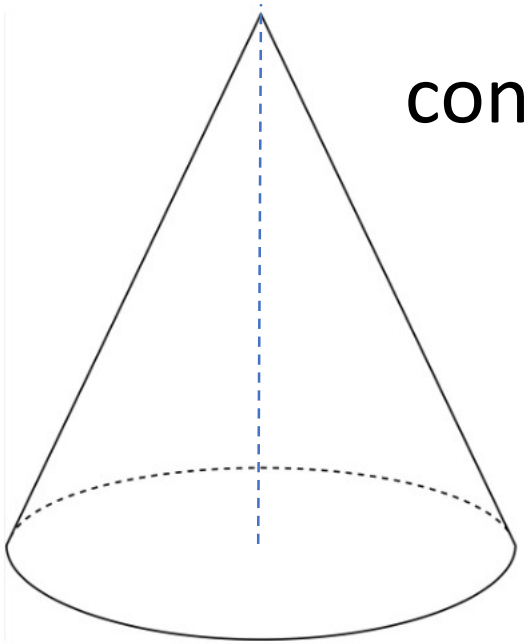


cone axis



cone axis

a circular cross section

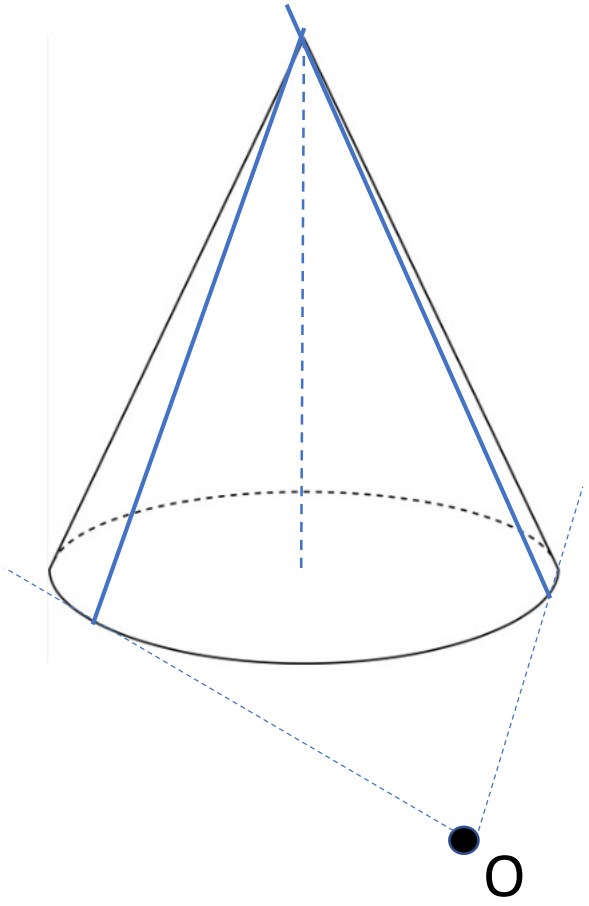


cone axis

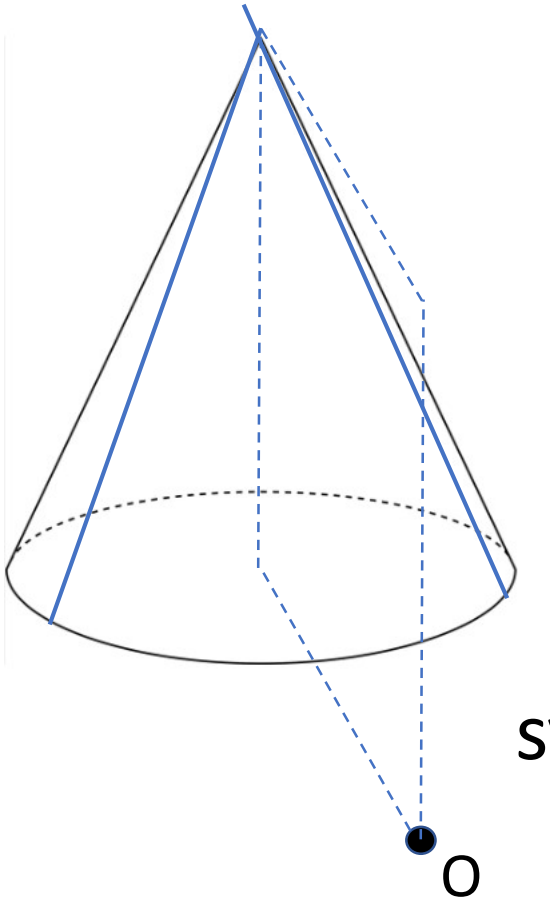
a circular cross section

o

camera viewpoint

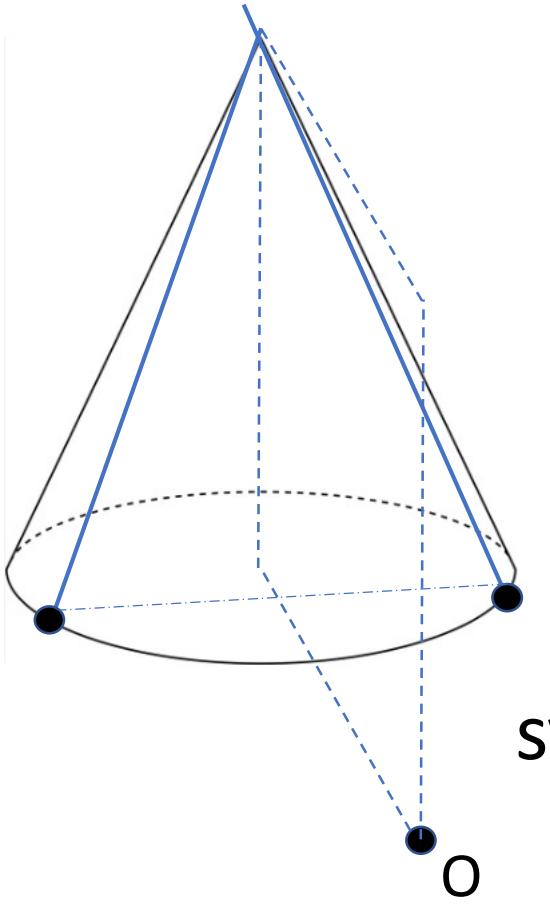


contour generator



contour generator

symmetry plane



contour generator

points are symmetric wrt symmetry plane

symmetry plane

Example: contour generator of a **right** cone?

- contour generator Γ: set of tangency points from O
→ two straight lines through the vertex V

- **right** cone e alone is **symmetric** wrt to its axis
→ right cone + viewpoint O : «less» symmetry,
which symmetry?
PLANAR SYMMETRY wrt plane through axis and O

Example: contour generator of a **right** cone?

- contour generator Γ: set of tangency points from O
→ two straight lines through the vertex V

- **right** cone alone is **symmetric** wrt to its axis
→ right cone + viewpoint O : «less» symmetry,
which symmetry?
PLANAR SYMMETRY wrt plane through axis and O
namely, wrt backprojection plane of the imaged axis

natural camera calibration from 3 images of a rectangle

Classical approach

Three equations from vanishing points of three mutually orthogonal directions:

Drawbacks: estimation of vanishing points might be inaccurate

Alternative approach

- Mutuated from Zhang: images of a planar object of known shape

→

- Here the object shape has a unique unknown parameter: the aspect ratio *a*, i.e., the ratio between «horizontal» and «vertical» lenght

Given a rectangle R of unknown aspect ratio a

- Take a first image of the rectangle: let r_1 be the extracted image
- Rectify this image by mapping it onto a rectangle R_1 by means of a homography H_1
- The aspect ratio of this reconstructed rectangle is $a_1 = sa$, where s is an unknown scale factor
- Therefore, the circular points of the plane containing the real rectangle R have been mapped onto two points (I_1, J_1) given by

$$(I_1, J_1) = \begin{bmatrix} s & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} (I, J)$$

- The first image of these circular points is therefore

$$(I_1', J_1') = H_1^{-1}(I_1, J_1) = H_1^{-1} \begin{bmatrix} s & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} (I, J)$$

- Consider now the second image R_2 of the rectangle: once it is partially rectified to a rectangle r_2, compute its aspect ratio $a_2 = s_2 a$. This time, the scale factor s_2 can be expressed in terms of s as

$$s_2 = \lambda s$$

with $\lambda = \frac{a_2}{a_1}$ known after the observation of the two aspect ratios

- If H_2 is the homography used to rectify the second image R_2 of the rectangle to the partially rectified rectangle r_2, then the second image of the circular points (I, J) is given by

$$(I_2', J_2') = H_2^{-1}(I_2, J_2) = H_2^{-1} \begin{bmatrix} \lambda s & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} (I, J)$$

and, similarly, the third image of the same circular points is

$$(I_3', J_3') = H_3^{-1}(I_3, J_3) = H_3^{-1} \begin{bmatrix} \mu s & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} (I, J)$$

Now we have three equations like in Zhang, but also an additional unknown scaling factor s

A further equation comes from the natural camera constraint:

An additional pair of imaged circular points is

$$(I, J) = \left(\begin{bmatrix} 1 \\ i \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ -i \\ 0 \end{bmatrix}\right)$$

in fact,

they are images of circular points of any plane parallel to the image plane

Therefore we have to solve

$$I_1'^T \omega I_1' = 0$$
$$I_2'^T \omega I_2' = 0$$
$$I_3'^T \omega I_3' = 0$$

and

$$I^T \omega I = 0$$

for both ω and s, with

$$\lambda = \frac{a_2}{a_1}$$

and

$$\mu = \frac{a_3}{a_1}$$