UNIVERSITY OF TWENTE.
RAM Robotics and Mechatronics

3D Computer Vision for Medical Applications

INTRODUCTION PROJECTIVE GEOMETRY AND CAMERA MODELS

F.vanderHeijden@utwente.nl

contents – camera models and projective space

- some mathematical preamble
  - inner product
  - cross product
- projective geometry
  - 2D lines and points in homogeneous representations
  - vanishing points
  - homographies in 2D images
- camera models
- virtual rotation of a camera

UNIVERSITY OF TWENTE.

RAM Robotics and Mechatronics

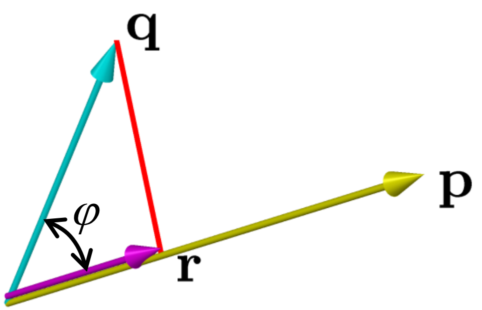
mathematical preamble

F.vanderHeijden@utwente.nl

math refresher I: Inner product, projection and orthogonality

inner product (= vector dot product):

- $(\mathbf{p},\mathbf{q}) = \|\mathbf{p}\|\|\mathbf{q}\|\cos\varphi$
- if $\mathbf{p}$ and $\mathbf{q}$ are column vectors: $(\mathbf{p},\mathbf{q}) = \mathbf{p}^T\mathbf{q}$



projection of $\mathbf{q}$ on $\mathbf{p}$:

- definition: $\mathbf{r} = \alpha\mathbf{p}$ such that $(\mathbf{p},\mathbf{r}) = (\mathbf{p},\mathbf{q})$
- $(\mathbf{p},\mathbf{r}) = (\mathbf{p},\mathbf{q}) \Rightarrow \alpha\|\mathbf{p}\|^2 = (\mathbf{p},\mathbf{q}) \Rightarrow \alpha = \dfrac{(\mathbf{p},\mathbf{q})}{\|\mathbf{p}\|^2} = \dfrac{\mathbf{p}^T\mathbf{q}}{\mathbf{p}^T\mathbf{p}}$

Matlab :

```
p'*q   or:  dot(p,q)
```

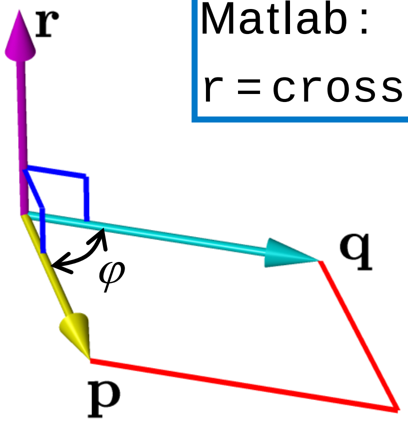
orthogonality :

- symbol: $\mathbf{p} \perp \mathbf{q}$
- definition: $\mathbf{p} \perp \mathbf{q}$ iff $(\mathbf{p},\mathbf{q}) = 0$
- equivalent with: $\varphi = 90^0$, $\cos\varphi = 0$, and $\mathbf{p}^T\mathbf{q} = 0$

math refresher II: cross product

cross product : $\mathbf{r} = \mathbf{p} \times \mathbf{q}$

- length of $\mathbf{r}$: $\|\mathbf{r}\| = \|\mathbf{p}\| \|\mathbf{q}\| \sin\varphi$ =area parallelogram
- direction of $\mathbf{r}$: $\mathbf{r} \perp \mathbf{p}$ and $\mathbf{r} \perp \mathbf{q}$ according to right hand rule



Matlab:
```
r = cross(p,q)
```

determinant form :

if
$$\mathbf{p} = p_x\mathbf{x} + p_y\mathbf{y} + p_z\mathbf{z}$$
$$\mathbf{q} = q_x\mathbf{x} + q_y\mathbf{y} + q_z\mathbf{z}$$

then
$$\mathbf{p} \times \mathbf{q} = \det\left(\begin{bmatrix} \mathbf{x} & \mathbf{y} & \mathbf{z} \\ p_x & p_y & p_z \\ q_x & q_y & q_z \end{bmatrix}\right)$$
$$= \left(p_y q_z - p_z q_y\right)\mathbf{x} - \left(p_x q_z - p_z q_x\right)\mathbf{y} + \left(p_x q_y - p_y q_x\right)\mathbf{z}$$

math refresher II: cross product

cross product : $\mathbf{r} = \mathbf{p} \times \mathbf{q}$

- length of $\mathbf{r}$: $\|\mathbf{r}\| = \|\mathbf{p}\|\|\mathbf{q}\|\sin\varphi$ =area parallelogram
- direction of $\mathbf{r}$: $\mathbf{r} \perp \mathbf{p}$ and $\mathbf{r} \perp \mathbf{q}$ according to right hand rule

skew - symmetric matrix form :

if $\mathbf{p} = p_x\mathbf{x} + p_y\mathbf{y} + p_z\mathbf{z}$

$\mathbf{q} = q_x\mathbf{x} + q_y\mathbf{y} + q_z\mathbf{z}$

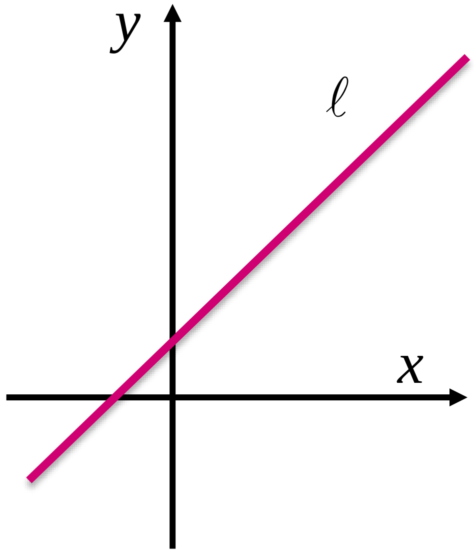
then $\mathbf{p} \times \mathbf{q} = \begin{bmatrix} 0 & -p_z & p_y \\ p_z & 0 & -p_x \\ -p_y & p_x & 0 \end{bmatrix} \mathbf{q}$

shorthand notation: $\mathbf{p} \times \mathbf{q} = [\mathbf{p}]_\times \mathbf{q}$ with $[\mathbf{p}]_\times \overset{def}{=} \begin{bmatrix} 0 & -p_z & p_y \\ p_z & 0 & -p_x \\ -p_y & p_x & 0 \end{bmatrix}$

introduction projective geometry

F.vanderHeijden@utwente.nl

representation of a line in 2D space: projective spaces



general representation of line ℓ:

$$ax + by + c = 0$$

thus, each line can be defined by a 3D vector: $\underline{\mathbf{l}} = \begin{bmatrix} a \\ b \\ c \end{bmatrix}$

equivalence of line representation:

$$\underline{\mathbf{l}} = \begin{bmatrix} a \\ b \\ c \end{bmatrix} \equiv \alpha \begin{bmatrix} a \\ b \\ c \end{bmatrix} \text{ for any } \alpha \neq 0$$

projective space:

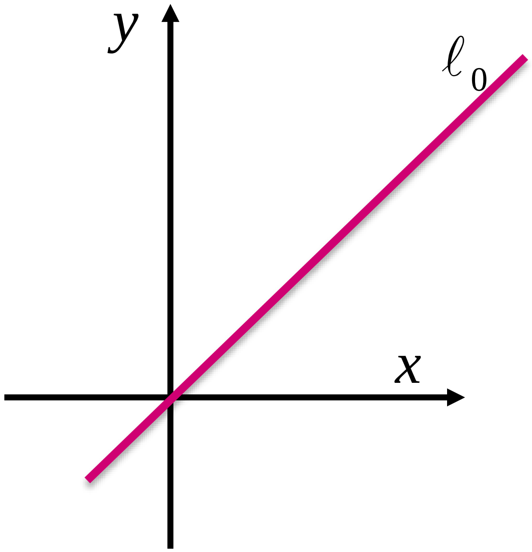
$\underline{\mathbf{l}} \in \mathbb{P}^2$ where $\mathbb{P}^2 \equiv \mathbb{R}^3 - \mathbf{0}$ i.e. 3D space without origin

representation of a 2D line crossing the origin

general representation of line ℓ_0 passing the origin:

ℓ_0 is defined by: $\begin{array}{l} x = \alpha x_0 \\ y = \alpha \end{array}$ where $\alpha \in \mathbb{R}$

thus, this line can be defined by a 2D vector: $\underline{\mathbf{l}}_0 = \begin{bmatrix} x_0 \\ 1 \end{bmatrix}$



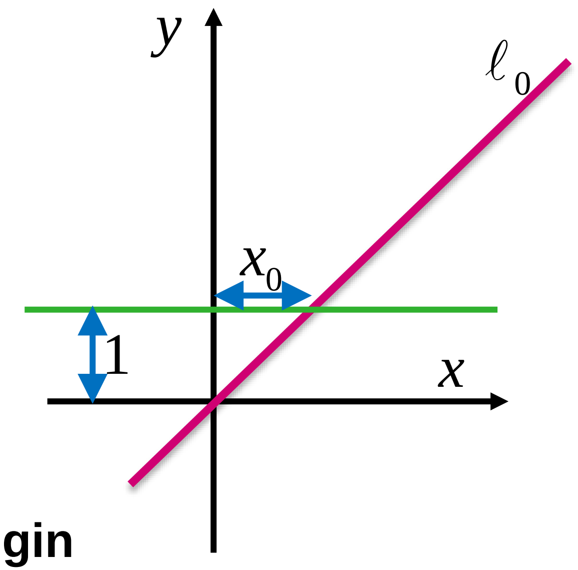
equivalence of line representation:

$\underline{\mathbf{l}}_0 = \begin{bmatrix} x_0 \\ 1 \end{bmatrix} \equiv \alpha \begin{bmatrix} x_0 \\ 1 \end{bmatrix}$ for any $\alpha \neq 0$

projective space:

$\underline{\mathbf{l}}_0 \in \mathbb{P}^1$ where $\mathbb{P}^1 \equiv \mathbb{R}^2 - \mathbf{0}$ i.e. 2D space without origin

homogeneous representation of a real number in a projective space



consider a real number x_0:

- x_0 **can be associated with a 2D line** ℓ_0 **crossing the origin**
- x_0 is the abscissa x of the line if the ordinate $y = 1$
- ℓ_0 **can be defined in a projective space by a 2D vector :** $\underline{\mathbf{l}}_0 = \alpha \begin{bmatrix} x_0 \\ 1 \end{bmatrix}$ **for any** $\alpha \neq 0$

homogeneneous representation of x_0: $\underline{x}_0 \overset{def}{=} \underline{\mathbf{l}}_0 = \alpha \begin{bmatrix} x_0 \\ 1 \end{bmatrix}$

2D points in a projective space

- cartesian representation

  of point in 2D image plane: $\mathbf{p}_0 = \begin{bmatrix} x_0 \\ y_0 \end{bmatrix}$

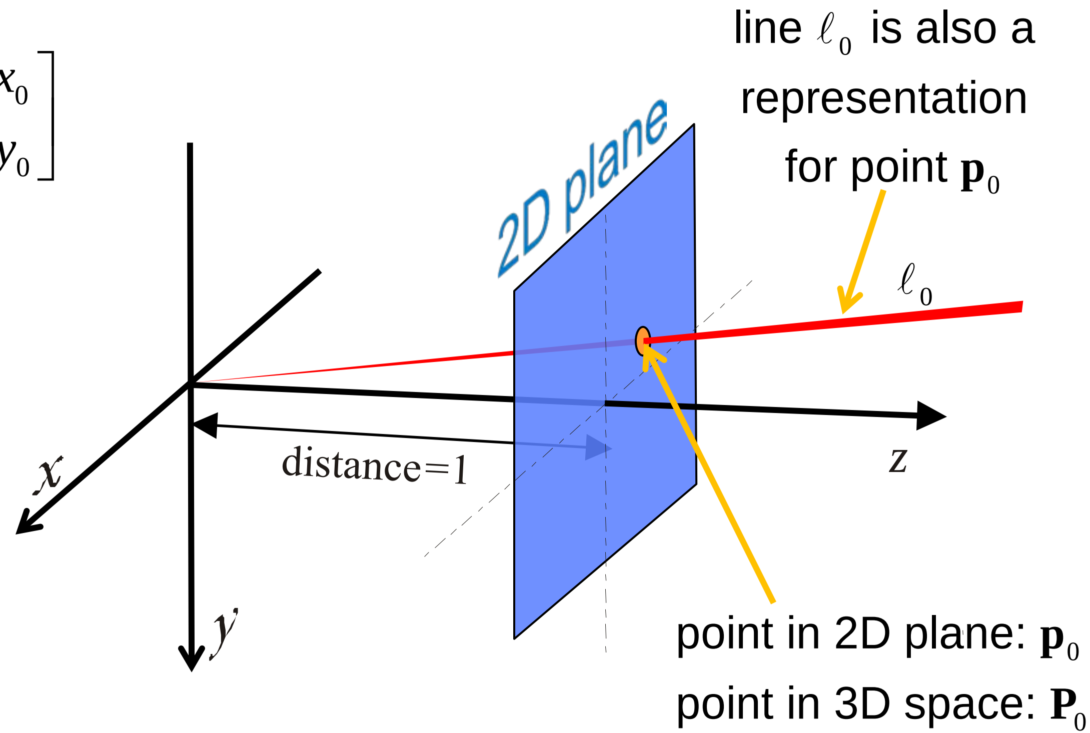
- same point in 3D: $\mathbf{P}_0 = \begin{bmatrix} x_0 \\ y_0 \\ 1 \end{bmatrix}$

- consider line ℓ_0 that intersects

  the origin and the point $\mathbf{P}_0$:

  line ℓ_0 is $\begin{cases} x = \alpha x_0 \\ y = \alpha y_0 \\ z = \alpha \end{cases}$ or: $\mathbf{l}_0 = \alpha \begin{bmatrix} x_0 \\ y_0 \\ 1 \end{bmatrix}$

  $\mathbf{l}_0$ represents ℓ_0, but so thus any $\alpha \mathbf{l}_0$ if $\alpha \neq 0$



- homogeneous representation $\mathbf{p}_0$: $\underline{\mathbf{p}}_0 = \mathbf{l}_0$

cart2hom and hom2cart

from cartersian to homogeneous coordinates:

$$\mathbf{p}_0 = \begin{bmatrix} x_0 \\ y_0 \end{bmatrix} \quad \Rightarrow \quad \underline{\mathbf{p}}_0 = \begin{bmatrix} x_0 \\ y_0 \\ 1 \end{bmatrix}$$

from homogenous to cartesian:

$$\underline{\mathbf{p}}_0 = \begin{bmatrix} a \\ b \\ c \end{bmatrix} \quad \Rightarrow \quad \mathbf{p}_0 = \begin{bmatrix} a/c \\ b/c \end{bmatrix}$$

MATLAB (robotics toolbox):

```
p0h = cart2hom(p0);
p0 = hom2cart(p0h);
```

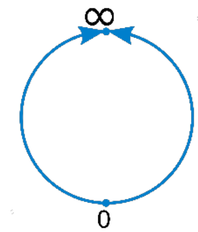
note:

matlab uses transposed form

point at infinity

theorem 1:

the points $\underline{\mathbf{p}}_1 = \begin{bmatrix} a \\ b \\ c \end{bmatrix}$ and $\underline{\mathbf{p}}_2 = \begin{bmatrix} -a \\ -b \\ -c \end{bmatrix}$ represent the same point $\mathbf{p} = \begin{bmatrix} a/c \\ b/c \end{bmatrix}$



theorem 2:

the point $\underline{\mathbf{p}}_1 = \begin{bmatrix} a \\ b \\ \varepsilon \end{bmatrix}$ with $\varepsilon \downarrow 0$ is at an infinite distance in the direction $\begin{bmatrix} a \\ b \end{bmatrix}$

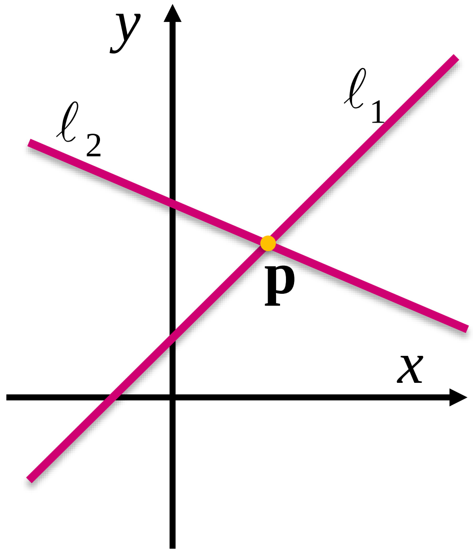
the point $\underline{\mathbf{p}}_2 = \begin{bmatrix} -a \\ -b \\ \varepsilon \end{bmatrix}$ with $\varepsilon \downarrow 0$ is at an infinite distance in the direction $\begin{bmatrix} -a \\ -b \end{bmatrix}$

corollary:

if $\varepsilon = 0$, i.e. **the point at infinity** $\begin{bmatrix} a \\ b \\ 0 \end{bmatrix}$, is the end point of a line (from either side, as on a circle)

the intersection point of two lines in 2D

given two lines ℓ_1 and ℓ_2, what is their intersection point?



- representation of ℓ_1 is: $\mathbf{l}_1 = [a_1 \quad b_1 \quad c_1]^\mathsf{T}$
- representation of ℓ_2 is: $\mathbf{l}_2 = [a_2 \quad b_2 \quad c_2]^\mathsf{T}$
- representation of $\mathbf{p}$ is: $\underline{\mathbf{p}} = [A \quad B \quad C]^\mathsf{T}$

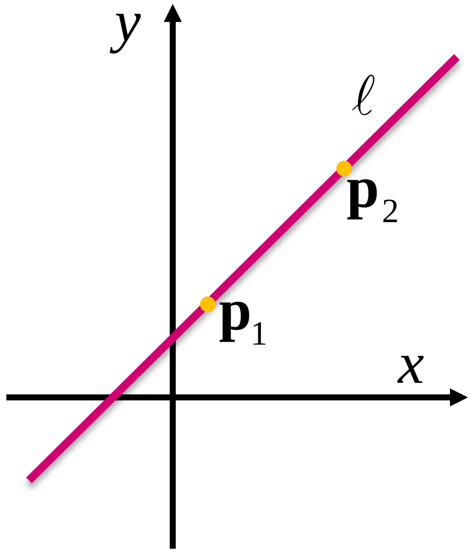
$\mathbf{p}$ is on ℓ_1, thus: $a_1 A + b_1 B + c_1 C = 0$ or: $\mathbf{l}_1^\mathsf{T} \underline{\mathbf{p}} = 0$ or: $\mathbf{l}_1 \perp \underline{\mathbf{p}}$

$\mathbf{p}$ is on ℓ_2 likewise: $\mathbf{l}_2 \perp \underline{\mathbf{p}}$

the vector $\mathbf{l}_1 \times \mathbf{l}_2$ is orthogonal to both $\mathbf{l}_1$ and $\mathbf{l}_2$

therefore: $\boxed{\underline{\mathbf{p}} = \mathbf{l}_1 \times \mathbf{l}_2}$

the line connecting two points



given two lines $\mathbf{p}_1$ and $\mathbf{p}_2$, what is their connecting line?

- representation of $\mathbf{p}_1$ is: $\underline{\mathbf{p}}_1 = \begin{bmatrix} a_1 & b_1 & c_1 \end{bmatrix}^\mathrm{T}$
- representation of $\mathbf{p}_2$ is: $\underline{\mathbf{p}}_2 = \begin{bmatrix} a_2 & b_2 & c_2 \end{bmatrix}^\mathrm{T}$
- representation of ℓ is: $\mathbf{l} = \begin{bmatrix} A & B & C \end{bmatrix}^\mathrm{T}$

$\mathbf{p}_1$ is on ℓ, thus: $a_1 A + b_1 B + c_1 C = 0$ or: $\mathbf{l}^\mathrm{T} \underline{\mathbf{p}}_1 = 0$ or: $\mathbf{l} \perp \underline{\mathbf{p}}_1$

$\mathbf{p}_2$ is on ℓ likewise: $\mathbf{l} \perp \underline{\mathbf{p}}_2$

the vector $\underline{\mathbf{p}}_1 \times \underline{\mathbf{p}}_2$ is orthogonal to both $\underline{\mathbf{p}}_1$ and $\underline{\mathbf{p}}_2$

therefore: $\boxed{\mathbf{l} = \underline{\mathbf{p}}_1 \times \underline{\mathbf{p}}_2}$

geometric transforms and homographies

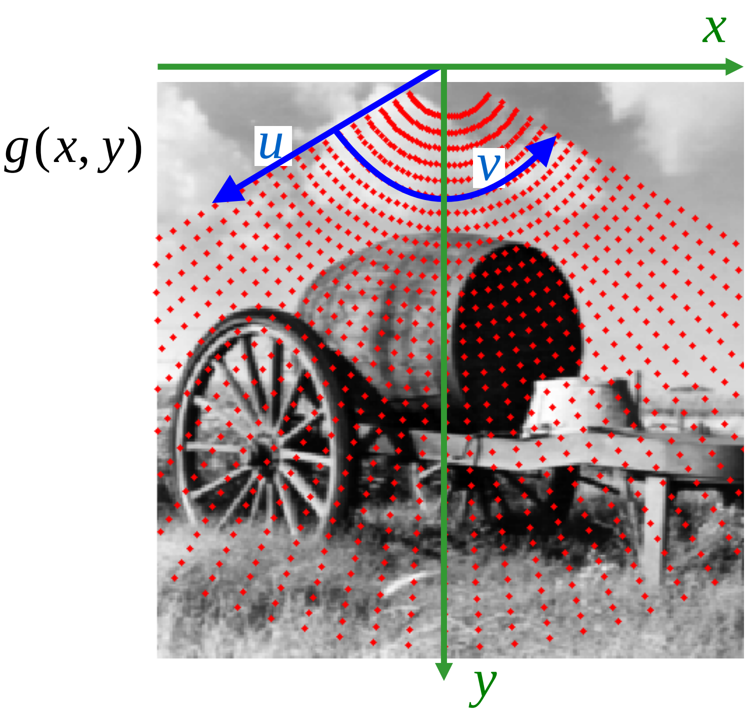
F.vanderHeijden@utwente.nl

geometrical transforms

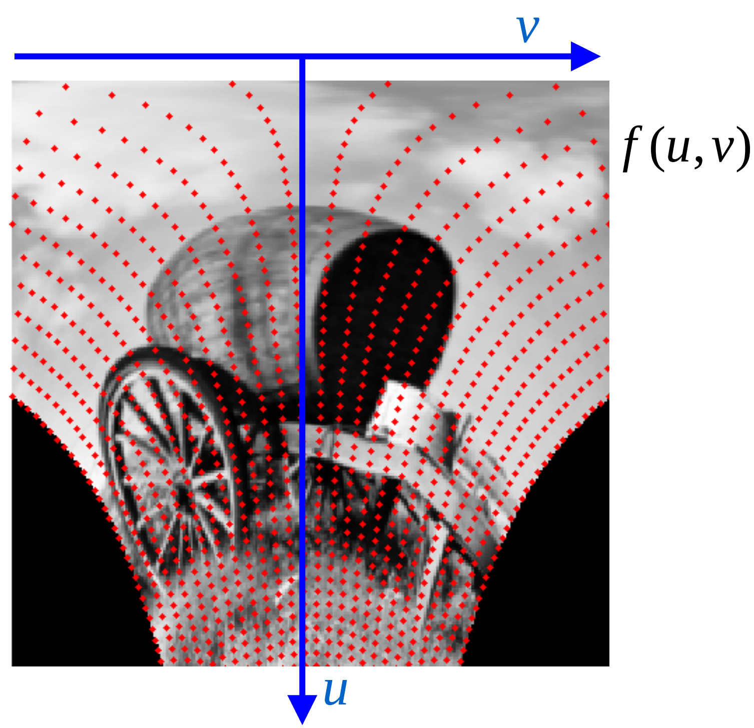
example: fan beam geometry

$$x = a\,u\cos(b\,v) \qquad u = \frac{1}{a}\sqrt{x^2 + y^2}$$

$$y = a\,u\sin(b\,v) \qquad v = \frac{1}{b}\arctan\left(\frac{y}{x}\right)$$



cartesian coordinates



polar coordinates

geometrical transforms

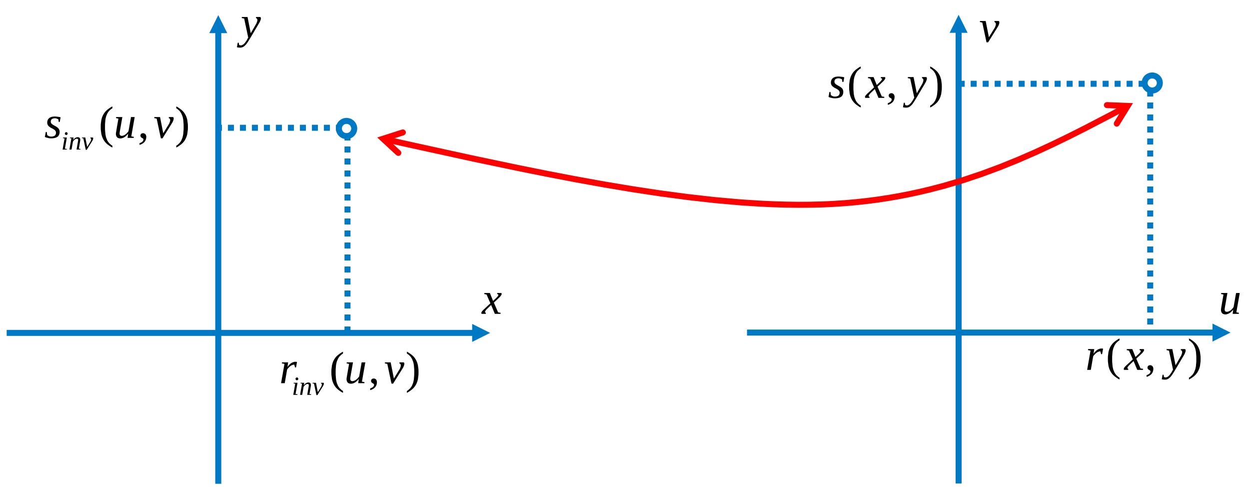
more general:

$$u = r(x,y)$$
$$v = s(x,y)$$

$$x = r_{inv}(u,v)$$
$$y = s_{inv}(u,v)$$

$$f(x,y) = g\big(r(x,y), s(x,y)\big)$$
$$g(u,v) = f\big(r_{inv}(u,v), s_{inv}(u,v)\big)$$

$\boxed{f(x,y)}$ $\boxed{g(u,v)}$

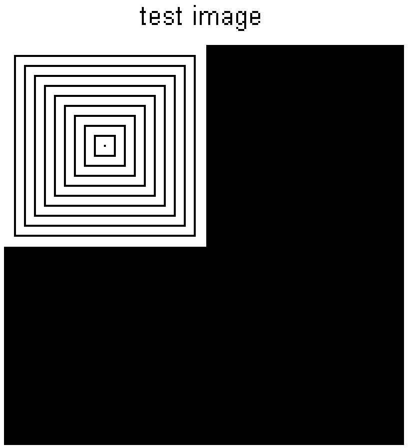


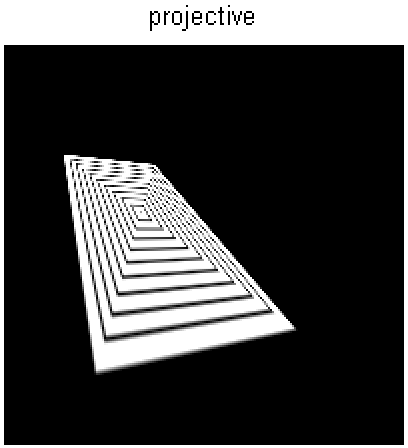
2D projective geometric transformation

projective transform, also called homography :

$$u = \frac{Ax + By + C}{Gx + Hy + 1}$$

$$v = \frac{Dx + Ey + F}{Gx + Hy + 1}$$





- transformed lines are still lines
- lines that share a common vanishing point keep on sharing a vanishing point
- distances are not preserved
- angles are not preserved

2D homography – vector matrix notation

- homography: $$u = \frac{Ax + By + C}{Gx + Hy + 1} \qquad v = \frac{Dx + Ey + F}{Gx + Hy + 1}$$
- define vectors: $\mathbf{x} = \begin{bmatrix} x \\ y \end{bmatrix}$ and $\mathbf{u} = \begin{bmatrix} u \\ v \end{bmatrix}$
- then: $\underline{\mathbf{u}} = \mathbf{H}\underline{\mathbf{x}}$ with: $\mathbf{H} = \begin{bmatrix} A & B & C \\ D & E & F \\ G & H & 1 \end{bmatrix}$
- inverse transform: $\underline{\mathbf{x}} = \mathbf{H}^{-1}\underline{\mathbf{u}}$
- note: $\mathbf{H} \equiv \alpha\mathbf{H}$ for any $\alpha \neq 0$

rectification of a projective transform

undoing a perspective distortion in an image

F.vanderHeijden@utwente.nl

perspective rectification

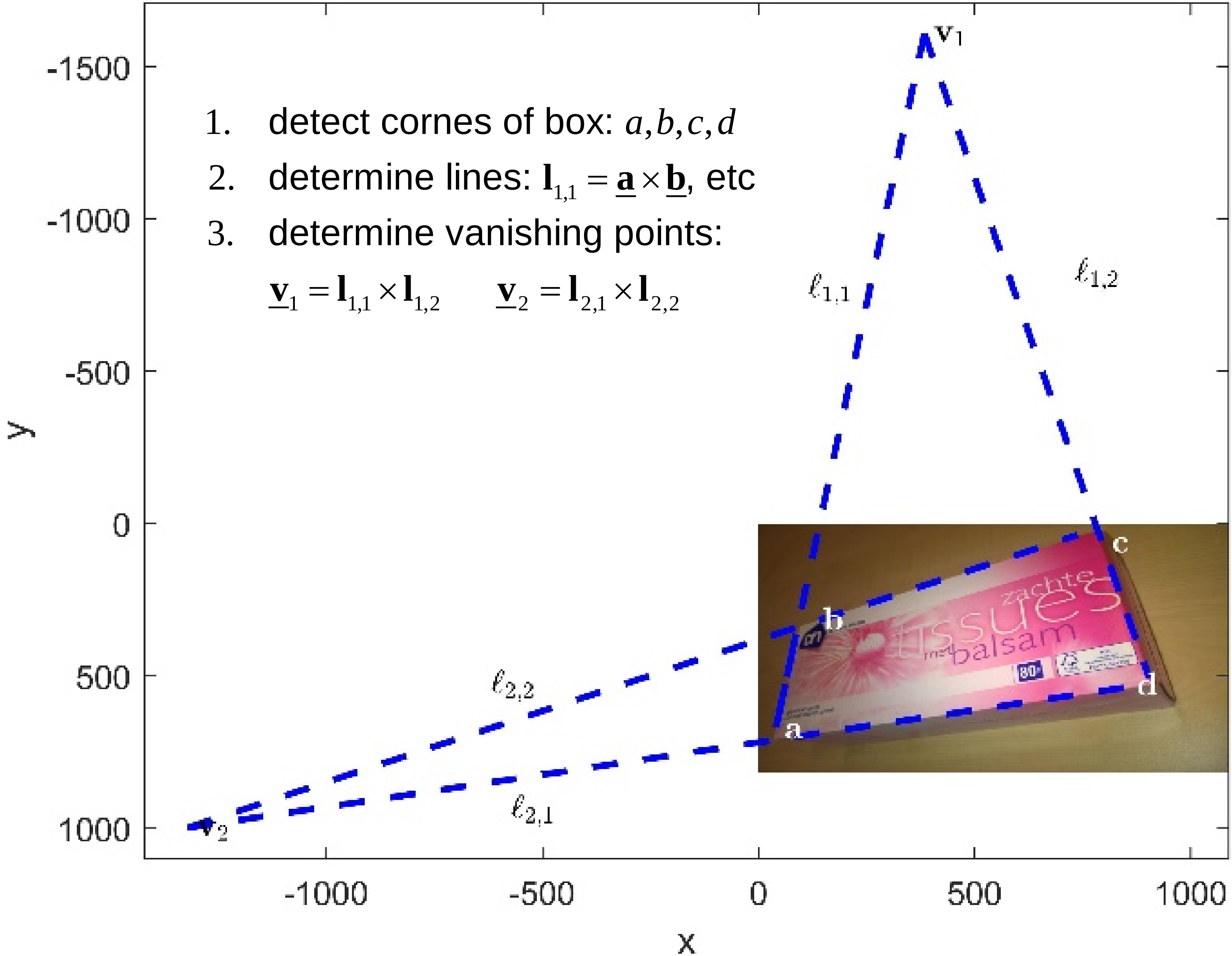
how to transform the image such that the frontside of the box becomes geometrically correct?

- parallel lines
- perpendicular angles

step 1: detection of vanishing points

1. detect cornes of box: a, b, c, d
2. determine lines: $\mathbf{l}_{1,1} = \underline{\mathbf{a}} \times \underline{\mathbf{b}}$, etc
3. determine vanishing points:

$$\underline{\mathbf{v}}_1 = \mathbf{l}_{1,1} \times \mathbf{l}_{1,2} \qquad \underline{\mathbf{v}}_2 = \mathbf{l}_{2,1} \times \mathbf{l}_{2,2}$$



step 2: determining the homography

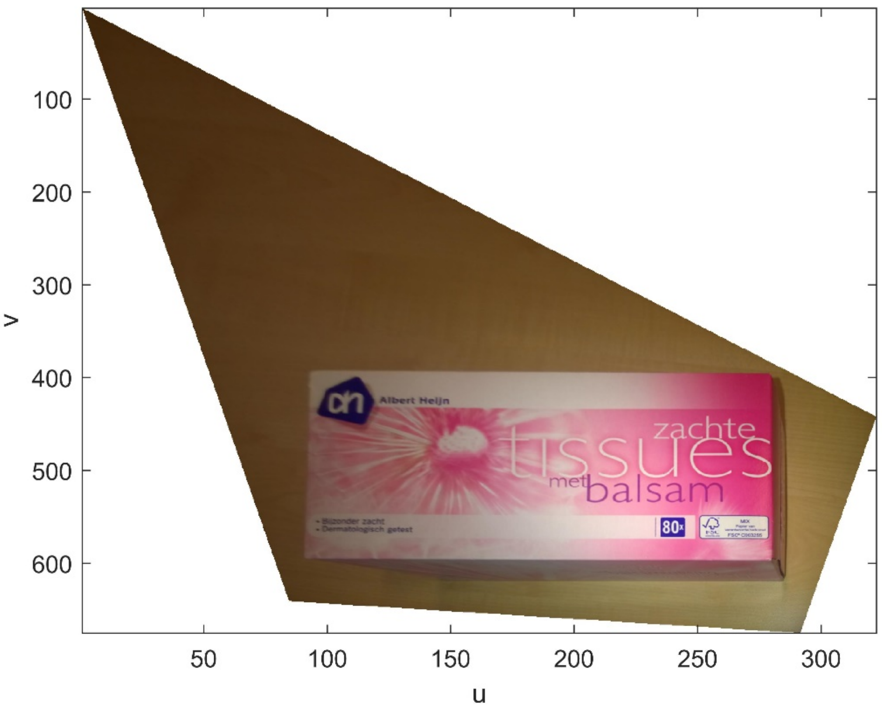
- the 3×3 matrix $\mathbf{H}$ has 8 free parameters
- we need at least 8 equations to solve $\mathbf{H}$

- shift of vanishing points to infinity:
  - $\underline{\mathbf{v}}_1$ must be shifted to $x = 0,\ y = \infty$
  - $\underline{\mathbf{v}}_2$ must be shifted to $x = \infty,\ y = 0$
- the origin must be preserved

this leads to 3 vector-matrix equations:

$$\mathbf{H}\underline{\mathbf{v}}_1 = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \qquad \mathbf{H}\underline{\mathbf{v}}_2 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \qquad \mathbf{H}\begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

from which $\mathbf{H}$ can be solved

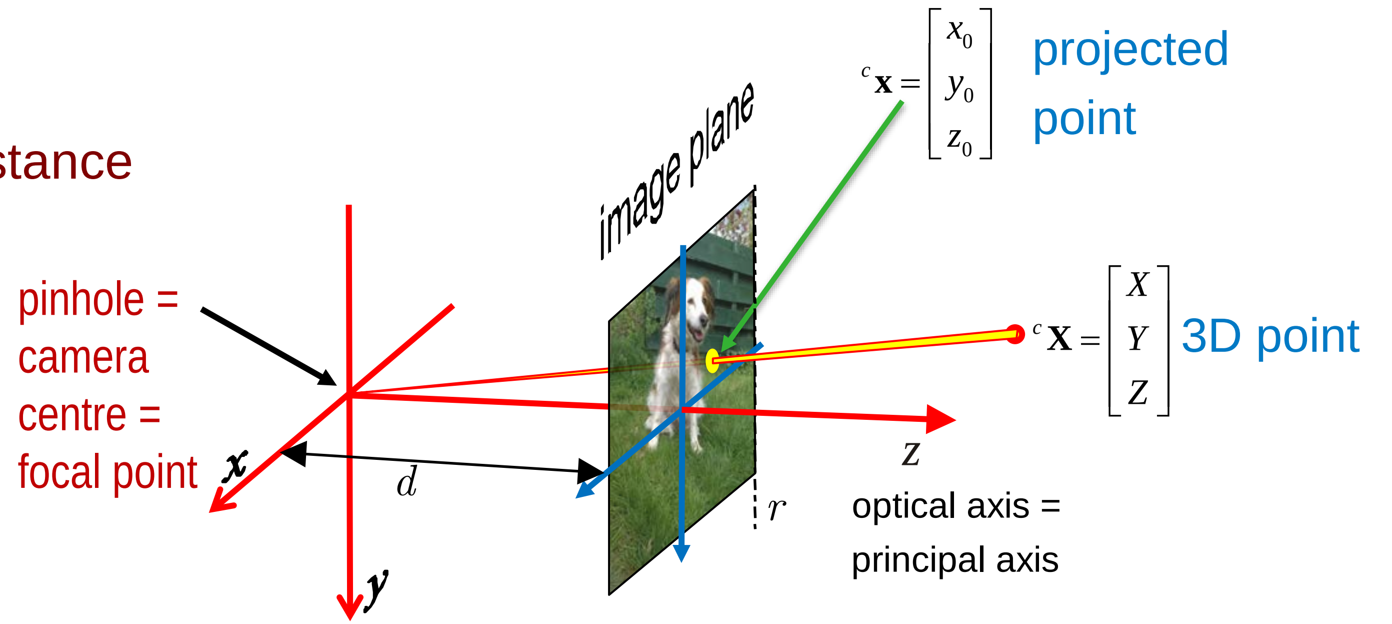


intrinsic camera model

F.vanderHeijden@utwente.nl

pinhole model for image formation in a camera

d : focal distance



perspective projection:

$$x_0 = \frac{Xd}{Z} \qquad y_0 = \frac{Yd}{Z} \qquad z_0 = d$$

image subscripts and image coordinates[1]

image subscripts ≡ array subscripts:	`r,c`:	row and column indices positive integers	`f(r,c)`
linear image index	`i`:	index for vectorized image positive integer	`f(i)`
image coordinates ≡ pixel coordinates:	c, r :	coordinates of the image reals	`c` $= round(c)$ `r` $= round(r)$
camera coordinates:	x_0, y_0 or x_0, y_0, z_0 :	2D or 3D point in the image plane	

Note the reversal of order: `(r,c)` $\triangleq (y, x)$

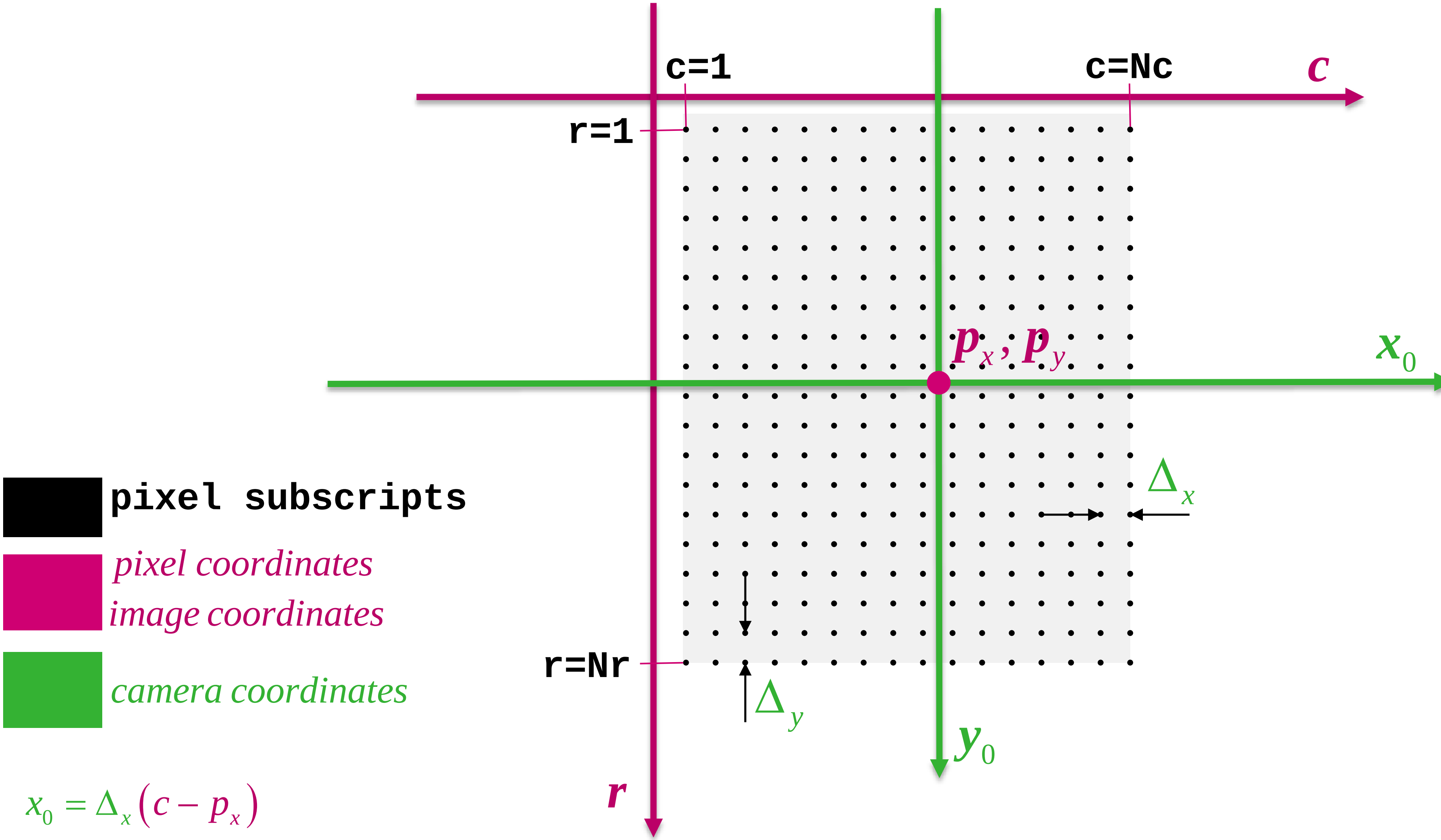
Sampling in the image plane:

$x_0 = \Delta_x (c - p_x)$ Δ_x, Δ_y is pixel pitches in x- and y-direction p_x and p_y is offset

$y_0 = \Delta_y (r - p_y)$

[1] The terminology, here, is valid for camera models.
In other areas, e.g. geometrical transforms, other terminology may apply

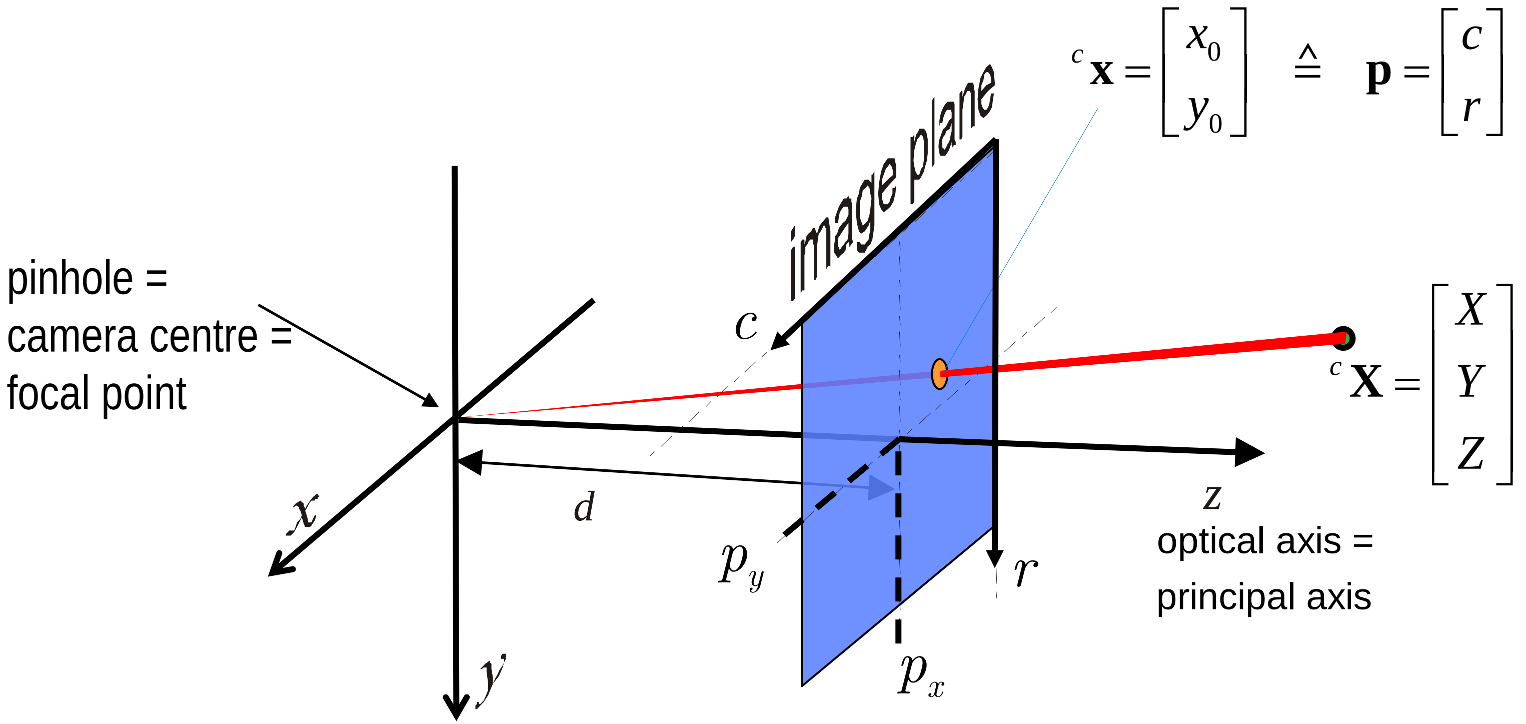
variables in the image plane



$$x_0 = \Delta_x \left(c - p_x \right)$$

$$y_0 = \Delta_y \left(r - p_y \right)$$

from camera coordinates to pixel coordinates



perspective projection:

$$c = \frac{Xd}{Z\Delta_x} + p_x \qquad r = \frac{Yd}{Z\Delta_y} + p_y$$

(r, c): Pixel coordinates

p_x, p_y: Principal point = image center

Δ_x, Δ_y: Pitches (distances between pixels)

Often, d is expressed in units of Δ.

E.g., if $d_x = 2000$, one actually means $d = 2000\Delta_x$

vector-matrix representation of camera pinhole model

- cartesian representation of an imaged 3D point in space: ${}^{c}\mathbf{X} = \begin{bmatrix} X & Y & Z \end{bmatrix}^{\mathrm{T}}$
- homogeneous representation of the 2D point in pixel coordinates: $\underline{\mathbf{p}} = \alpha \begin{bmatrix} c & r & 1 \end{bmatrix}^{\mathrm{T}}$

then†:

$$\alpha c = d_x X + p_x Z \qquad \text{with } d_x = d/\Delta_x$$
$$\alpha r = d_y Y + p_y Z \qquad \text{with } d_y = d/\Delta_y$$
$$\alpha = Z$$

note: d_x is d, but expressed in pixel units

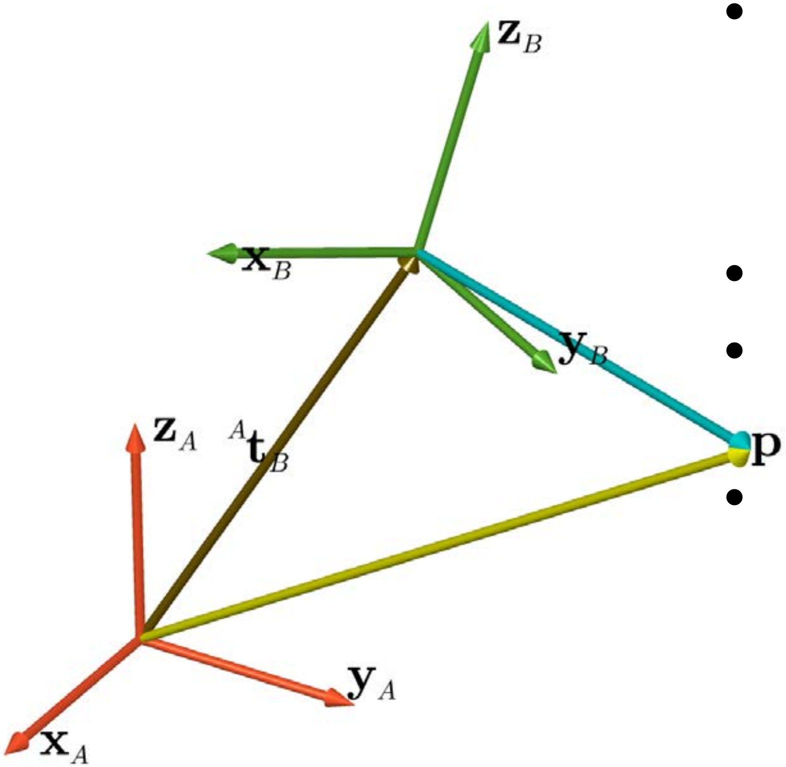
in vector-matrix notation:

$$\underline{\mathbf{p}} = \mathbf{K}\, {}^{c}\mathbf{X} \quad \text{with} \quad \mathbf{K} = \begin{bmatrix} d_x & 0 & p_x \\ 0 & d_y & p_y \\ 0 & 0 & 1 \end{bmatrix}$$

$\mathbf{K}$ is the **calibration matrix**

† because after 'dehomogenization' we have: $c = \dfrac{d_x X}{Z} + p_x \qquad r = \dfrac{d_y Y}{Z} + p_y$

math refresher III: rotated and translated frames



$${}^B\mathbf{R}_A = {}^A\mathbf{R}_B^T$$
$${}^B\mathbf{t}_A = -{}^B\mathbf{R}_A\,{}^A\mathbf{t}_B$$

- Two coordinate systems (frames): A and B
- Point $\mathbf{p}$ has two representations:

$\mathbf{p} = {}^A p_x \mathbf{x}_A + {}^A p_y \mathbf{y}_A + {}^A p_z \mathbf{z}_A$ thus: ${}^A\mathbf{p} = \begin{bmatrix} {}^A p_x & {}^A p_y & {}^A p_z \end{bmatrix}^T$

$\mathbf{p} = {}^B p_x \mathbf{x}_B + {}^B p_y \mathbf{y}_B + {}^B p_z \mathbf{z}_B$ thus: ${}^B\mathbf{p} = \begin{bmatrix} {}^B p_x & {}^B p_y & {}^B p_z \end{bmatrix}^T$

- Frame A is is the reference frame
- Frame B is given by: $\left\{ {}^A\mathbf{R}_B, {}^A\mathbf{t}_B \right\}$
- Conversion:

$${}^A\mathbf{p} = {}^A\mathbf{R}_B\,{}^B\mathbf{p} + {}^A\mathbf{t}_B$$

- Conversely:

$${}^B\mathbf{p} = {}^A\mathbf{R}_B^T\left({}^A\mathbf{p} - {}^A\mathbf{t}_B\right) = {}^B\mathbf{R}_A\,{}^A\mathbf{p} + {}^B\mathbf{t}_A$$

world and camera coordinate systems

a point in world coordinates (wc):

$$ {}^{w}\mathbf{X} = \begin{bmatrix} {}^{w}X & {}^{w}Y & {}^{w}Z \end{bmatrix}^{\mathrm{T}} $$

beware of word confusing:

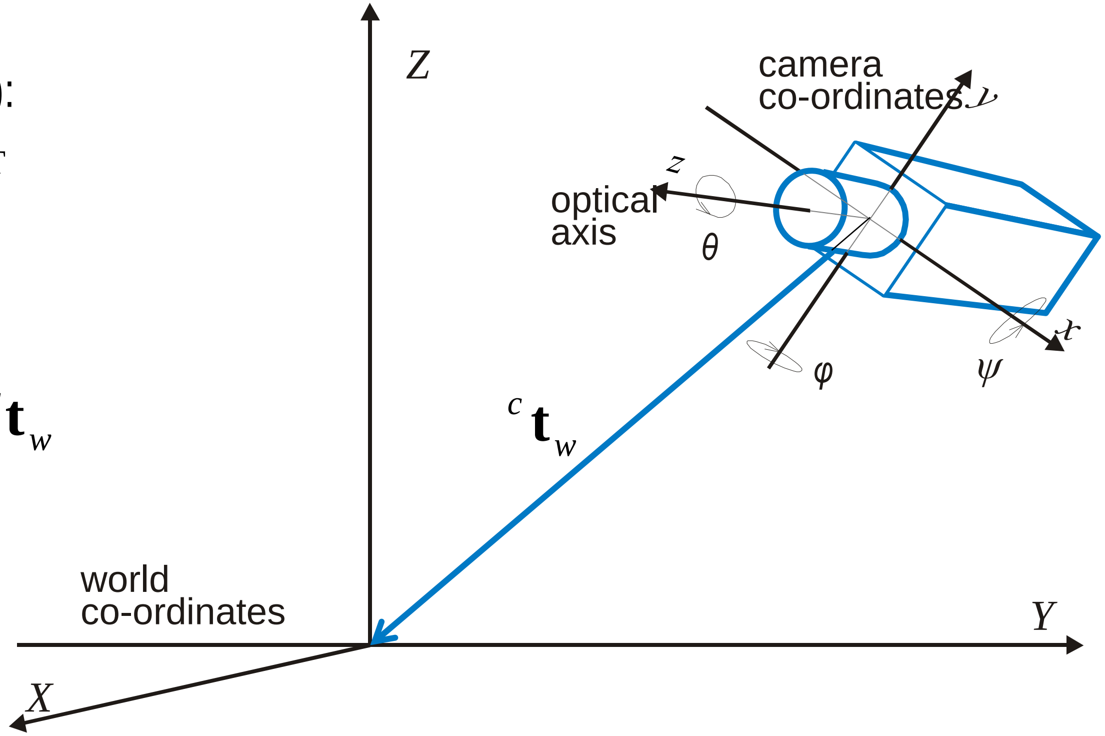
- frame: single image from a video
- frame: coordinate system

the same point in camera coordinates (cc):

$$ {}^{c}\mathbf{X} = \begin{bmatrix} X & Y & Z \end{bmatrix}^{\mathrm{T}} $$

conversion

$$ {}^{c}\mathbf{X} = {}^{c}\mathbf{R}_{w}\,{}^{w}\mathbf{X} + {}^{c}\mathbf{t}_{w} $$



${}^{c}\mathbf{t}_{w}$ position of wc-origin expressed in cc

${}^{c}\mathbf{R}_{w}$ orientation of wc expressed in cc

camera model: putting it altogether

A 3D point in space, from inhomogeneous world coordinates to homogeneous:

$$ {}^{w}\mathbf{X} = \begin{bmatrix} {}^{w}X \\ {}^{w}Y \\ {}^{w}Z \end{bmatrix} \quad \Rightarrow \quad {}^{w}\underline{\mathbf{X}} = \begin{bmatrix} {}^{w}X \\ {}^{w}Y \\ {}^{w}Z \\ 1 \end{bmatrix} $$

From world coordinates to camera coordinates:

$$ {}^{c}\mathbf{X} = {}^{c}\mathbf{R}_{w}\, {}^{w}\mathbf{X} + {}^{c}\mathbf{t}_{w} \quad \Rightarrow \quad {}^{c}\underline{\mathbf{X}} = \begin{bmatrix} {}^{c}\mathbf{R}_{w} & {}^{c}\mathbf{t}_{w} \\ \mathbf{0} & 1 \end{bmatrix} {}^{w}\underline{\mathbf{X}} $$

From camera coordinates to homogeneous image coordinates:

$$ \begin{aligned} \underline{\mathbf{p}} &= \mathbf{K}\left[\mathbf{I} \mid \mathbf{0}\right] {}^{c}\underline{\mathbf{X}} \\ &= \mathbf{K}\left[\mathbf{I} \mid \mathbf{0}\right] \begin{bmatrix} {}^{c}\mathbf{R}_{w} & {}^{c}\mathbf{t}_{w} \\ \mathbf{0} & 1 \end{bmatrix} {}^{w}\underline{\mathbf{X}} \\ &= \mathbf{K}\left[{}^{c}\mathbf{R}_{w} \mid {}^{c}\mathbf{t}_{w}\right] {}^{w}\underline{\mathbf{X}} \end{aligned} $$

From homogenous image coordinates to inhomogeneous:

$$ \mathbf{p} = \begin{bmatrix} c \\ r \end{bmatrix} \quad \Leftarrow \quad \boxed{\underline{\mathbf{p}} = \underbrace{\mathbf{K}}_{3\times3} \underbrace{\left[{}^{c}\mathbf{R}_{w} \mid {}^{c}\mathbf{t}_{w}\right]}_{3\times4} {}^{w}\underline{\mathbf{X}}} $$

nonlinear lens deformation

radial distortion :

parameters: $k_1,\ k_2,$ and k_3

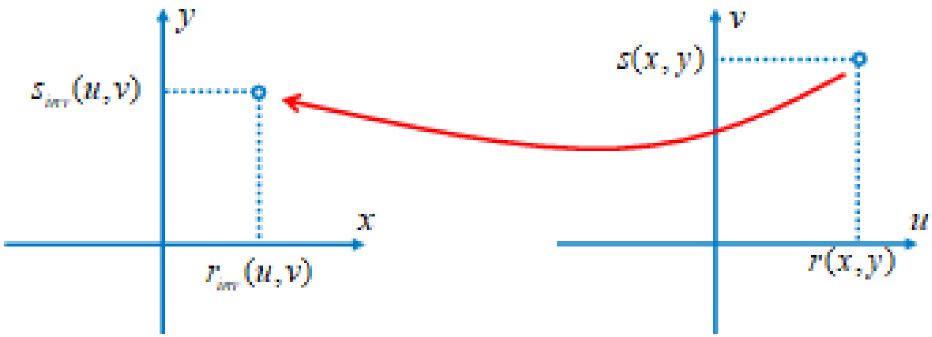
$$u = x(1 + k_1 r^2 + k_2 r^4 + k_3 r^6)$$
$$v = y(1 + k_1 r^2 + k_2 r^4 + k_3 r^6)$$

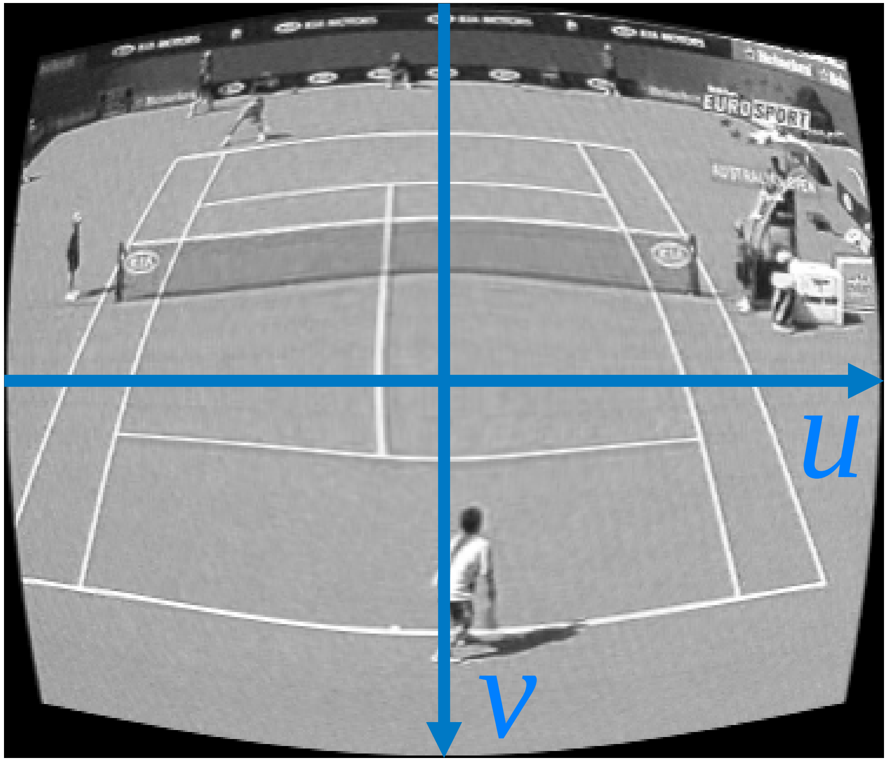
where $r^2 = x^2 + y^2$

tangential distortion :

parameters: $p_1,$ and p_2

$$u = x + \left(2 p_1 xy + p_2\left(r^2 + 2x^2\right)\right)$$
$$v = y + \left(2 p_2 xy + p_1\left(r^2 + 2y^2\right)\right)$$





nonlinear lens deformation

radial distortion :

parameters: $k_1,\ k_2,$ and k_3

$$u = x(1 + k_1 r^2 + k_2 r^4 + k_3 r^6)$$

$$v = y(1 + k_1 r^2 + k_2 r^4 + k_3 r^6)$$

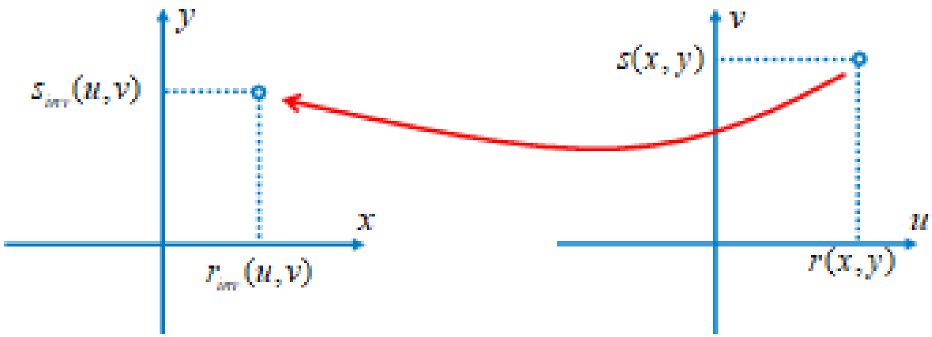
where $r^2 = x^2 + y^2$

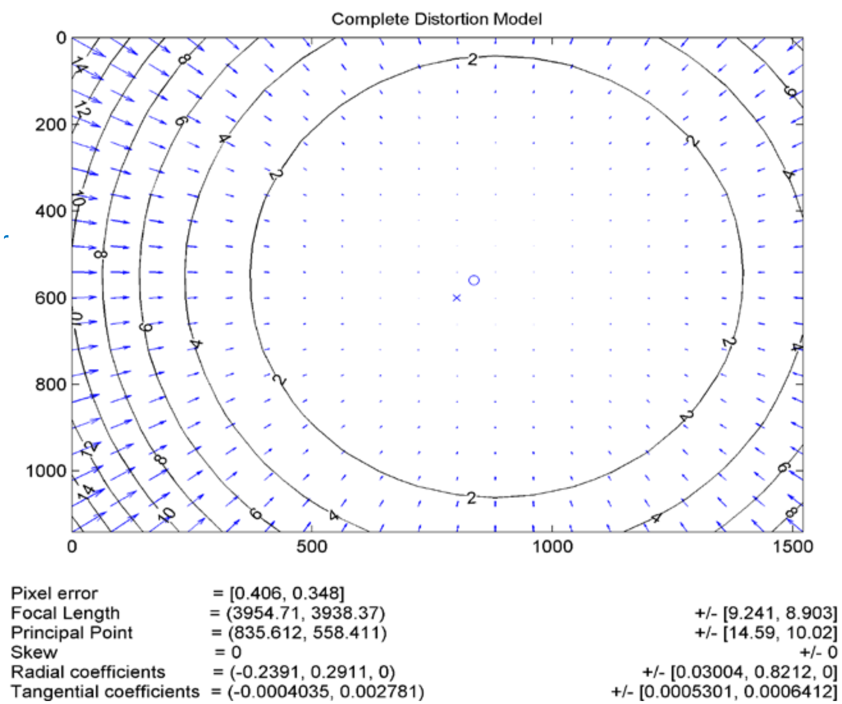
tangential distortion :

parameters: $p_1,$ and p_2

$$u = x + \left(2 p_1 xy + p_2 \left(r^2 + 2x^2\right)\right)$$

$$v = y + \left(2 p_2 xy + p_1 \left(r^2 + 2y^2\right)\right)$$





Camera calibration

Intrinsic camera parameters:

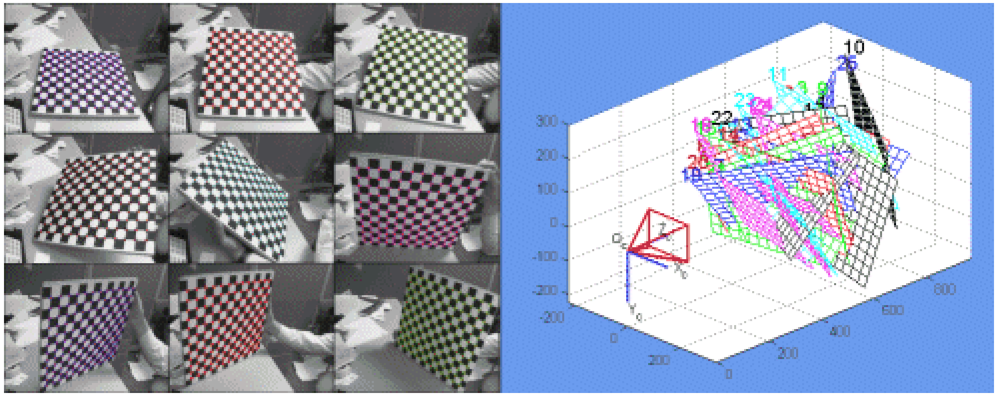
for each camera:

$d_x = d/\Delta_x$	ratio focal distance and pixel period in x direction
$d_y = d/\Delta_y$	ratio focal distance and pixel period in y direction
p_x, p_y	image centre (= principal point)
α	skewness (often set to zero)
$k_1, \cdots, k_5$	parameters describing non-linear lens distortion

Extrinsic parameters

$\mathbf{t}$	baseline vector
$\mathbf{R}$	rotation parameters

In full we need $2 \times 10 + 3 + 3 = 26$ parameters.



Details: See section 5 in syllabus "F. van der Heijden: Camera models"

Matlab's implementation (includes handy GUI): **`cameraCalibrator`**

virtual rotation of a camera

how to process an image such that it looks as if the camera has been rotated?

F.vanderHeijden@utwente.nl

Which geometrical transform is needed?

- Consider a pixel with homogeneous coordinates $\underline{\mathbf{p}} = \begin{bmatrix} c \\ r \\ 1 \end{bmatrix}$
- Suppose $\mathbf{K} = \begin{bmatrix} d_x & 0 & p_x \\ 0 & d_y & p_y \\ 0 & 0 & 1 \end{bmatrix}$, then $\mathbf{K}^{-1} = \begin{bmatrix} 1/d_x & 0 & -p_x/d_x \\ 0 & 1/d_y & -p_y/d_y \\ 0 & 0 & 1 \end{bmatrix}$
- Then: $\mathbf{x} \overset{def}{=} \mathbf{K}^{-1}\underline{\mathbf{p}} = \begin{bmatrix} (c-p_x)/d_x \\ (r-p_y)/d_y \\ 1 \end{bmatrix} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$
- Note that $\mathbf{x}$ is a cartesian 3D point in camera coordinates on the plane $z=1$
- Rotate this 3D point using a rotation matrix: $\mathbf{Rx} = \mathbf{RK}^{-1}\underline{\mathbf{p}}$
- Project this rotated point back to the image plane: $\underline{\mathbf{p}}_{new} = \mathbf{KRK}^{-1}\underline{\mathbf{p}}$

$\Rightarrow$ The required transform is a homography $\underline{\mathbf{p}}_{new} = \mathbf{H}\underline{\mathbf{p}}$ with $\mathbf{H} = \mathbf{KRK}^{-1}$

Exercise 1: virtual rotation of a camera

Measuring the size of a foot:

- The A4 paper can be used as a reference to measure the size of the foot.

Problem:

- The A4 paper has a perspective distortion because the camera is not pointing orthogonally to the floor.

Solution:

- Virtually rotate the camera such that it is pointing orthogonally to the floor

Questions:

- Which geometrical transform is needed?
- Which rotation is needed?

Which rotation matrix is needed?

- After rotation, the A4 must become a rectangle
- Thus, the angles at the corners must be 90^0
- The rotation matrix $\mathbf{R}$ is defined by 3 Euler angles

$\Rightarrow$ Choose the Euler angles such that corner angles become 90^0

How?

- Define an error measure that quantifies the deviation from 90^0
- Minimize this measure by varying the Euler angles.