COMP 558

Lecture 17

Homogeneous Coordinates (finish)
- review
- points at infinity
- 2D

Camera Extrinsics & Intrinsics

Thurs. Nov. 1, 2018

Homogenous Coordinates

To represent a 3D point, (X, Y, Z) we write the point in 4D as $(X, Y, Z, 1)$.

This allows us to represent various transformations in a similar way, namely matrix multiplication.

Translation

$$\begin{bmatrix} X + t_x \\ Y + t_y \\ Z + t_z \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

Rotation

$$\begin{bmatrix} \boxed{\phantom{\begin{matrix} x \\ x \\ x \end{matrix}}} \\ 1 \end{bmatrix} = \begin{bmatrix} \boxed{\begin{matrix} * & * & * \\ * & * & * \\ * & * & * \end{matrix}} & \begin{matrix} 0 \\ 0 \\ 0 \end{matrix} \\ \begin{matrix} 0 & 0 & 0 \end{matrix} & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

3 x 3 rotation matrix

result of 3D rotation

Scaling

$$\begin{bmatrix} \sigma_X X \\ \sigma_Y Y \\ \sigma_Z Z \\ 1 \end{bmatrix} = \begin{bmatrix} \sigma_X & 0 & 0 & 0 \\ 0 & \sigma_Y & 0 & 0 \\ 0 & 0 & \sigma_Z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}.$$

What if we have a value different than 1 in the 4th coordinate?

$$\{ \ (wX, wY, wZ, w) \ \ : \ \ w \neq 0 \ \}$$

No problem. These 4D points all represent the same 3D point, namely (X, Y, Z).

COMP 558

Lecture 17

Homogeneous Coordinates (finish)

- review
- points at infinity
- 2D

Camera Extrinsics & Intrinsics

Thurs. Nov. 1, 2018

Consider a 3D point (X, Y, Z) and scale the coordinates of this point by $s > 0$:

$$(sX, sY, sZ, 1) \equiv (X, Y, Z, \frac{1}{s})$$

For different values s, we get 3D points that all lie along a line from the origin through (X, Y, Z).

$$(sX, sY, sZ, 1) \equiv (X, Y, Z, \frac{1}{s})$$

As $s \rightarrow \infty$, we get a "point at infinity" in direction (X, Y, Z).

$$lim_{s \rightarrow \infty} \, (sX, sY, sZ, 1) = (X, Y, Z, 0)$$

What happens if we apply a rotation or translation or scaling transformation to a point at infinity?

Translating a point at infinity

$$? \quad = \begin{bmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 0 \end{bmatrix}$$

Translating a point at infinity

$$\begin{bmatrix} X \\ Y \\ Z \\ 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 0 \end{bmatrix}$$

Rotating a point at infinity

$$\begin{bmatrix} \boxed{} \\ 0 \end{bmatrix} = \begin{bmatrix} * & * & * & 0 \\ * & * & * & 0 \\ * & * & * & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 0 \end{bmatrix}$$

result of 3D rotation

3 x 3 rotation matrix

So, it behaves similarly to the rotation of a finite point.

Scaling

$$\begin{bmatrix} \sigma_X X \\ \sigma_Y Y \\ \sigma_Z Z \\ 0 \end{bmatrix} = \begin{bmatrix} \sigma_X & 0 & 0 & 0 \\ 0 & \sigma_Y & 0 & 0 \\ 0 & 0 & \sigma_Z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 0 \end{bmatrix}.$$

Note the direction of the point at infinity will be changed when axes are scaled by different amounts.

Exercise:

How are (3D) points at infinity related to vanishing points?

Homogeneous Coordinates in 2D

Translation:

$$\begin{bmatrix} x + t_x \\ y + t_y \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Rotation:

$$\begin{bmatrix} x\cos\theta - y\sin\theta \\ x\sin\theta + y\cos\theta \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

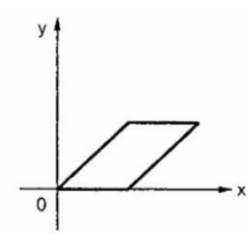
Homogeneous Coordinates in 2D

scaling

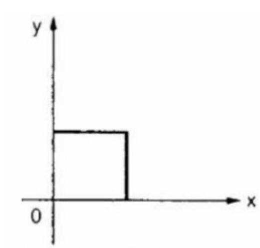
$$\begin{bmatrix} \sigma_x x \\ \sigma_y y \\ 1 \end{bmatrix} = \begin{bmatrix} \sigma_X & 0 & 0 \\ 0 & \sigma_Y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

shear

$$\begin{bmatrix} x + sy \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & s & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$



(b) Object after x shear



(a) Original object

Points at infinity in 2D homogenous coordinates

$$lim_{s \to \infty}(sx, sy, 1) = lim_{s \to \infty}\ (x, y, \frac{1}{s}) = (x, y, 0)$$

You can think of this as a direction vector.

We will use 2D points at infinity in a few weeks.

COMP 558

Lecture 17

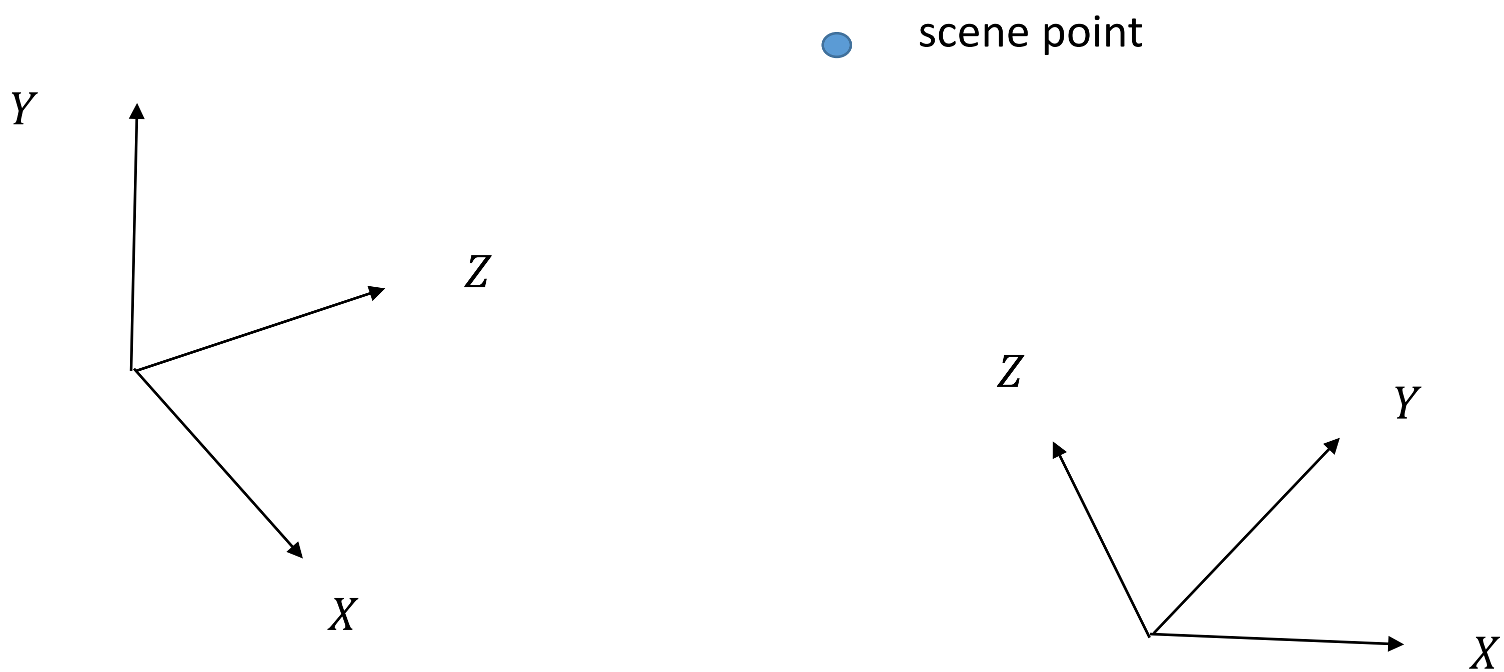
Homogeneous Coordinates (review and finish)

- points at infinity
- 2D

Camera Extrinsics & Intrinsics

Thurs. Nov. 1, 2018

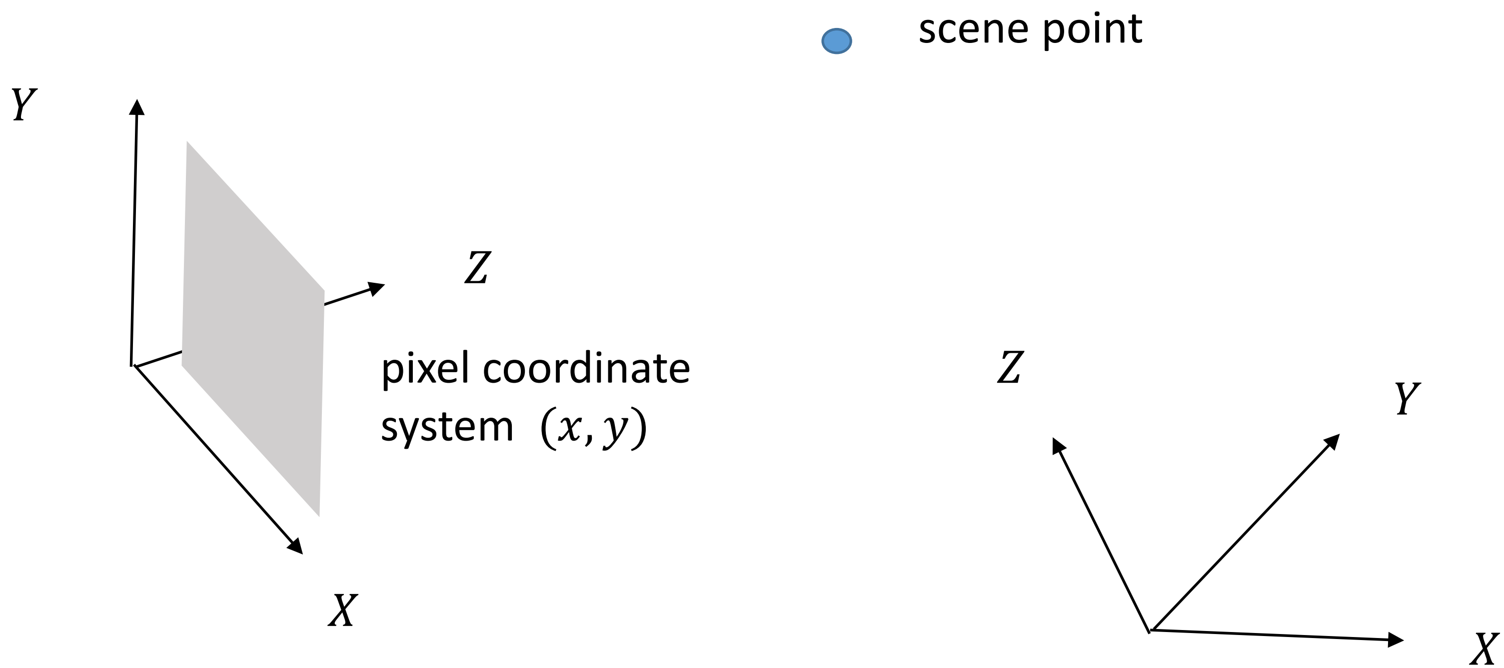
Camera vs. World Coordinates



camera coordinate system

world coordinate system

Pixel vs. Camera vs. World Coordinates



camera coordinate system

world coordinate system

Extrinsic (or external) means written in world coordinates.

Intrinsic (or internal) means written in camera coordinates.

How do 3D scene points (in world coordinates) map to pixel positions ?

How do 3D scene points (in world coordinates) map to pixel positions ?

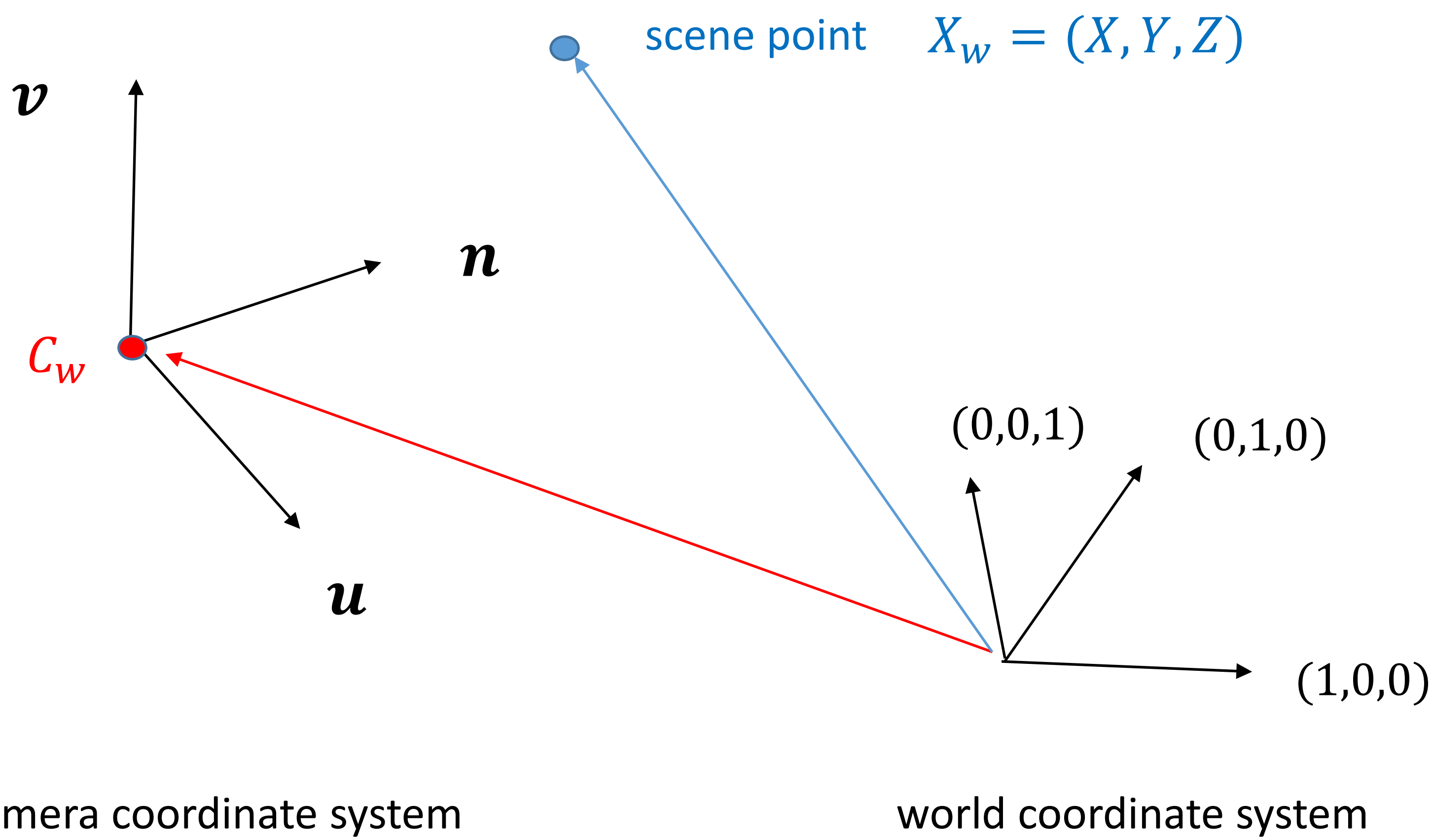
1. Map from world coordinates to camera coordinates

2. Project onto the projection plane.

3. Map from projection plane to pixel coordinates.

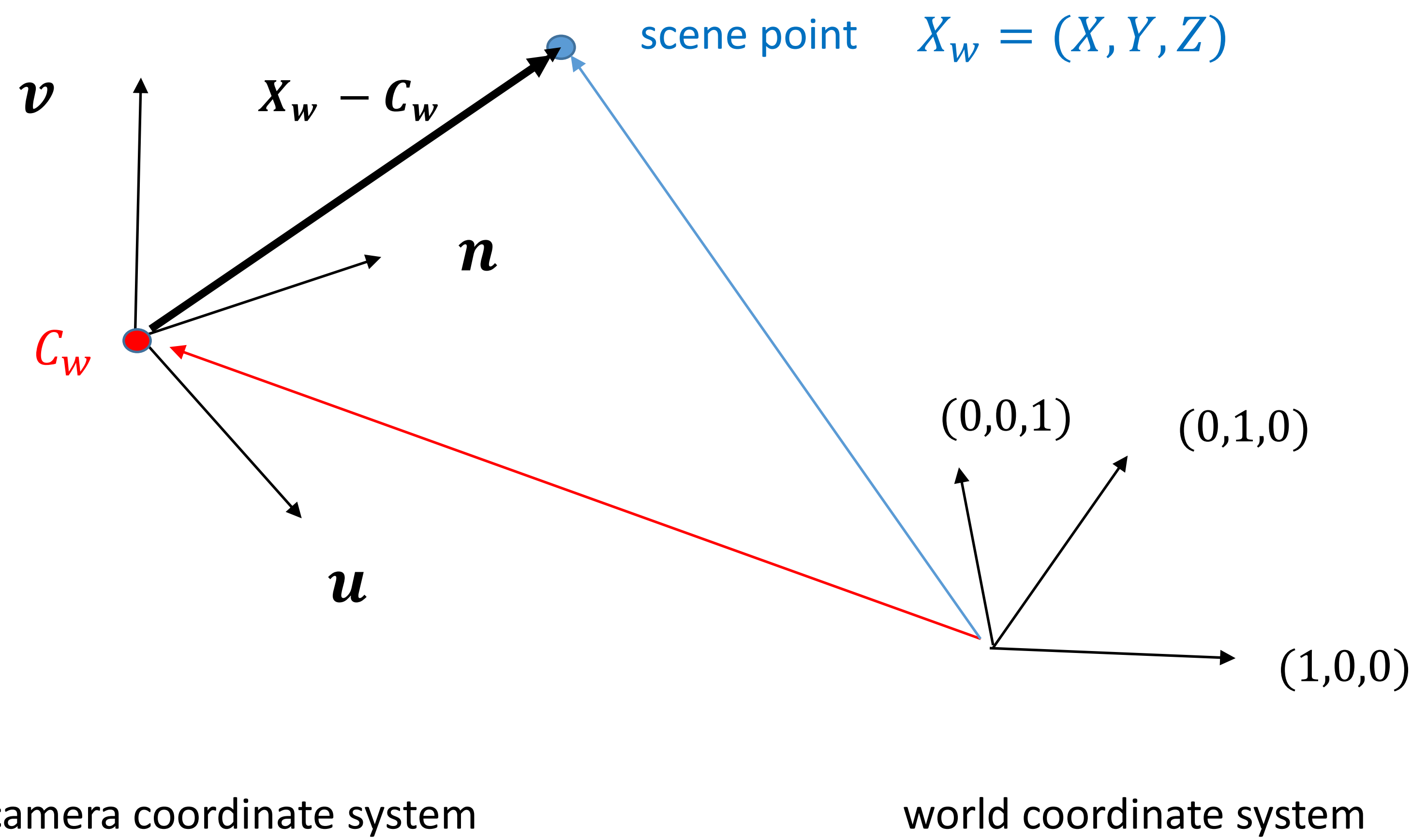
1. Map from world coordinates to camera coordinates

Let $\boldsymbol{X}_w$ be the position of a scene point in world coordinates.
Let C_w be the camera position in world coordinates.



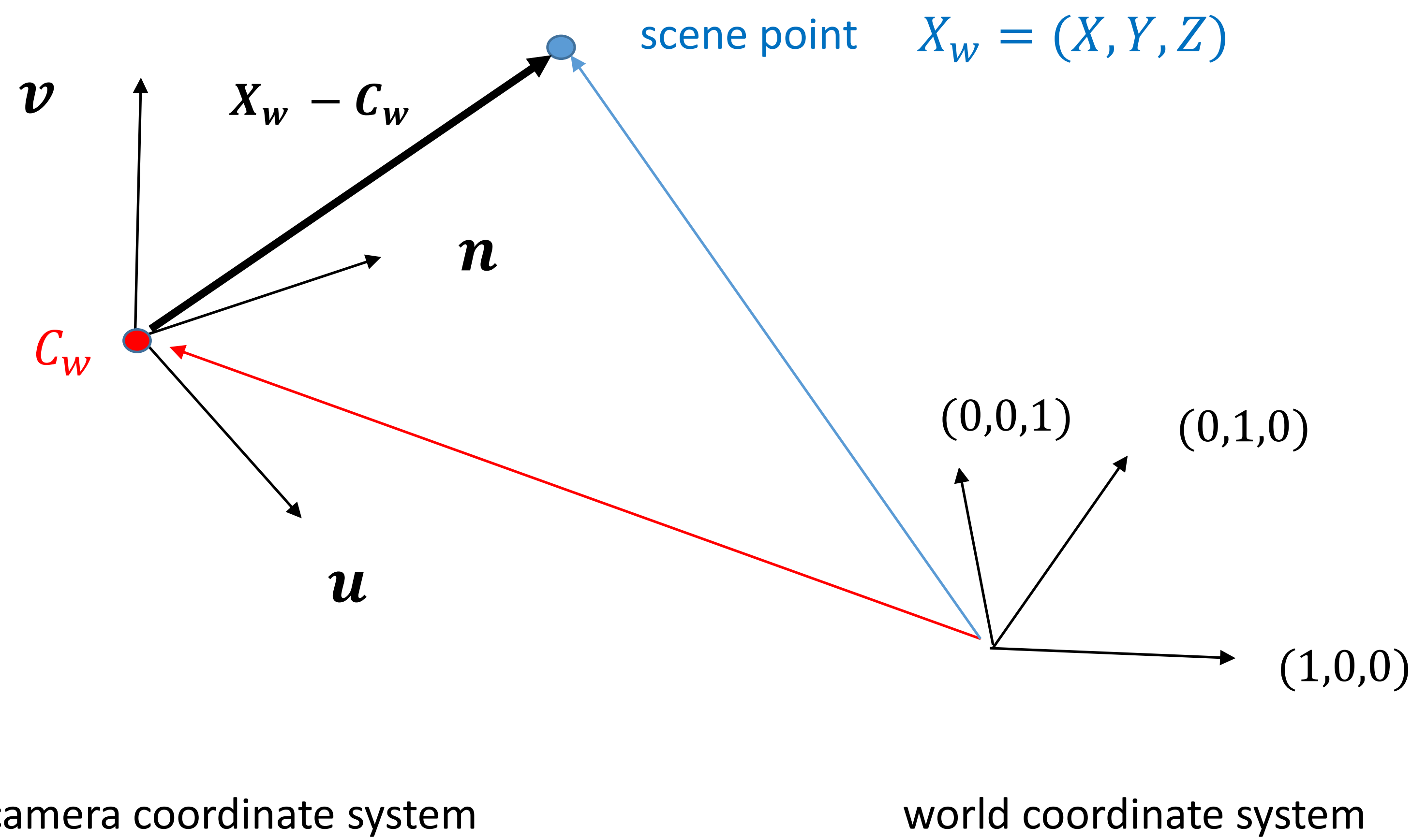
1. Map from world coordinates to camera coordinates

Then, $\boldsymbol{X}_w - C_w$ is the vector from C_w to $\boldsymbol{X}_w$. But this vector is still expressed in terms of world coordinate axes.



1. Map from world coordinates to camera coordinates

We want to write $\boldsymbol{X}_w - C_w$ in terms of camera's coordinate axes.



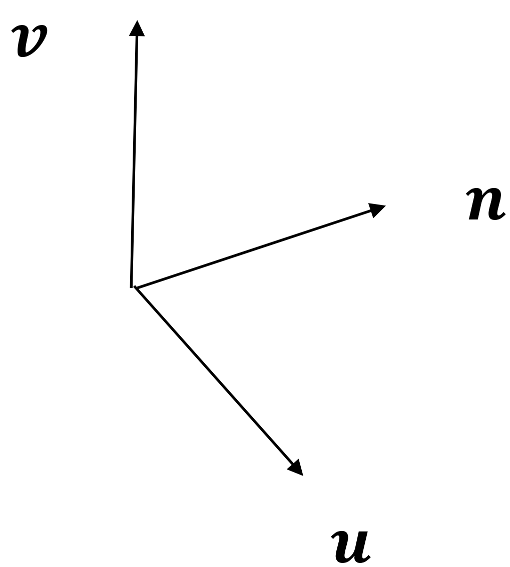
We need to perform a rotation from world coordinate axes to camera coordinate axes.

$$\mathbf{R}_{c \leftarrow w}$$

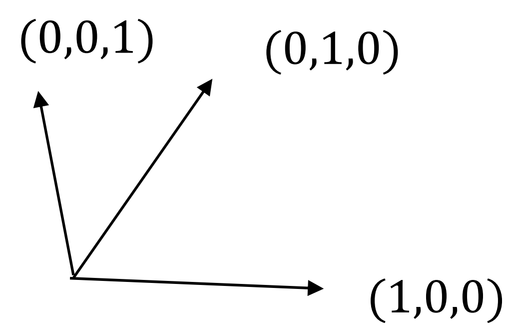
$$\mathbf{X}_c = \mathbf{R}(\mathbf{X}_w - \mathbf{C}_w)$$

$$\mathbf{R}_{c \leftarrow w}$$

The rows of this rotation matrix are camera coordinate axis unit vectors, $\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{n}$ *written in world coordinate system.*



camera coordinate system



world coordinate system

$$\begin{bmatrix} — & \boldsymbol{u} & — \\ — & \boldsymbol{v} & — \\ — & \boldsymbol{n} & — \end{bmatrix} \begin{bmatrix} | & | & | \\ \boldsymbol{u} & \boldsymbol{v} & \boldsymbol{n} \\ | & | & | \end{bmatrix} = I$$

$$RR^T = I$$

Summary: transformation from world to camera coordinates

$$\mathbf{X}_c = \mathbf{R}(\mathbf{X}_w - \mathbf{C}_w)$$

$$= \mathbf{R}\mathbf{X}_w - \mathbf{R}\mathbf{C}_w$$

Using homogeneous coordinates, we write:

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix} = \underbrace{\begin{bmatrix} \mathbf{R} & -\mathbf{R}\mathbf{C}_w \\ 0 & 1 \end{bmatrix}}_{\text{4x4}} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix}$$

$$\mathbf{X}_c = \mathbf{R}(\mathbf{X}_w - \mathbf{C}_w)$$

$$= \mathbf{R}\mathbf{X}_w - \mathbf{R}\mathbf{C}_w$$

Once we have the point in camera (intrinsic) coordinates, we can drop the 4th coordinate. This is equivalent to not bothering with the 4th row of the transformation matrix.

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \\ \cancel{1} \end{bmatrix} = \begin{bmatrix} \mathbf{R} & -\mathbf{R}\mathbf{C}_w \\ \cancel{0} & \cancel{1} \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix}$$

3x4

How do 3D scene points (in world coordinates) map to pixel positions ?

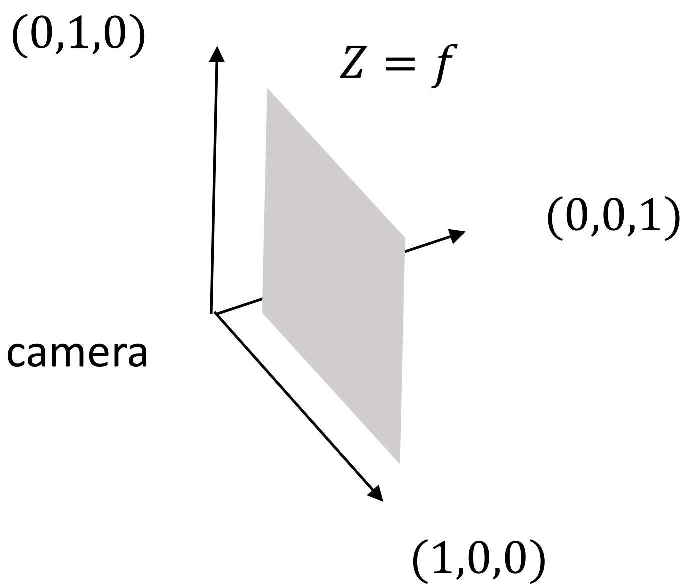
1. Map from world coordinates to camera coordinates.

2. Project onto the projection plane.

3. Map from projection plane to pixel coordinates.

How to project $\boldsymbol{X_c} = (X, Y, Z)$ to projection plane $Z = f$?

Note: *we use only camera coordinates now.*

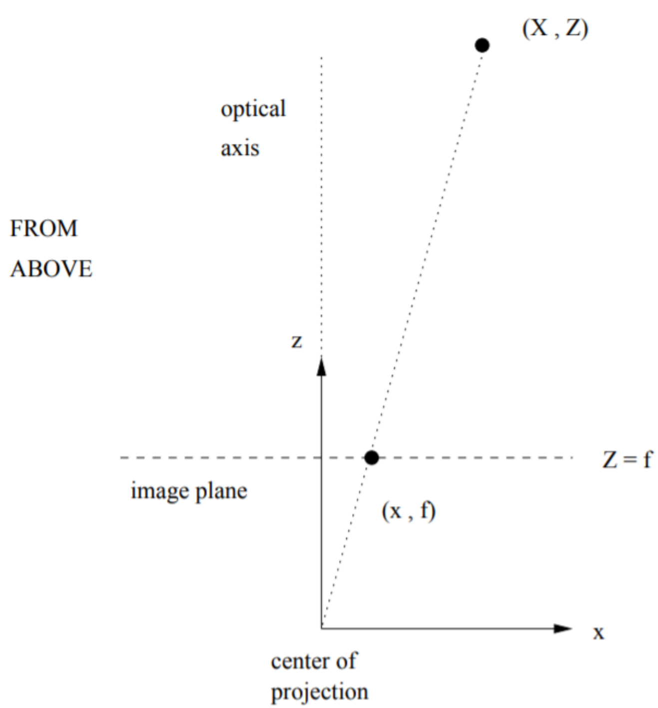


scene point $\boldsymbol{X_c} = (X, Y, Z)$
Represented in camera coordinate system.

How to project $\boldsymbol{X_c} = (X, Y, Z)$ to projection plane $Z = f$?

Recall lecture 15:

$$(x, y) = (f\frac{X}{Z}, f\frac{Y}{Z})$$



How to project $\boldsymbol{X_c} = (X, Y, Z)$ to projection plane $Z = f$?

Recall lecture 15:

$$(x, y) = (f\frac{X}{Z}, f\frac{Y}{Z})$$

Let's rewrite this 2D point in homogeneous coordinates:

$$(x, y, 1) = (f\frac{X}{Z}, f\frac{Y}{Z}, 1)$$

If $Z \neq 0$ then

$$(f\frac{X}{Z}, f\frac{Y}{Z}, 1) \equiv (fX, fY, Z).$$

How to project $\boldsymbol{X_c} = (X, Y, Z)$ to projection plane $Z = f$?

$$\begin{bmatrix} fX \\ fY \\ Z \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

2D homogeneous coordinate representation of point in projection plane

How do 3D scene points (in world coordinates) map to pixel positions ?

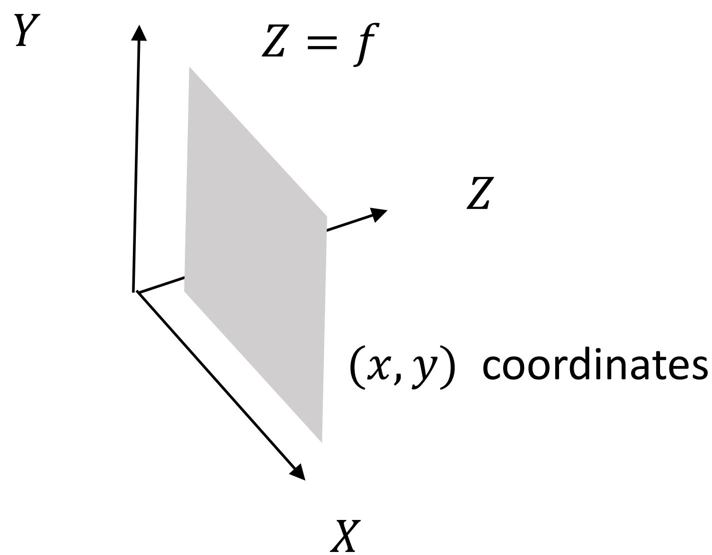
1. Map from world coordinates to camera coordinates.

2. Project onto the projection plane.

3. Map from projection plane coordinates to pixel coordinates.

The units of (x, y) are the same as the units of world and camera coordinates, e.g. millimetres.

We would like to transform them to pixel index coordinates.



projection plane

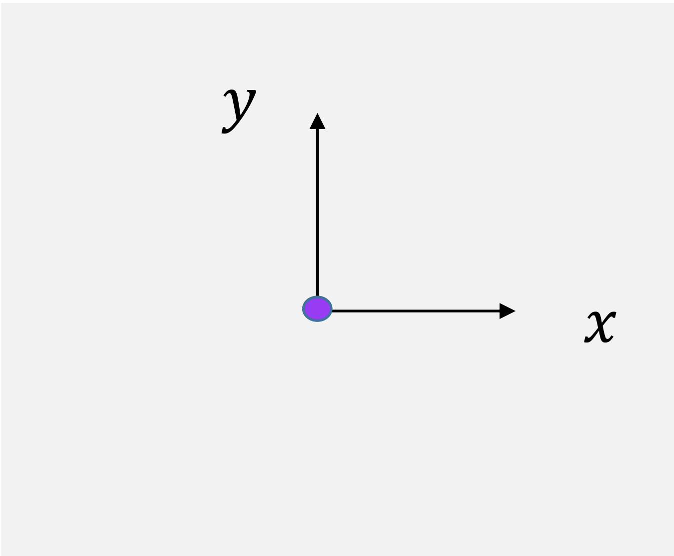
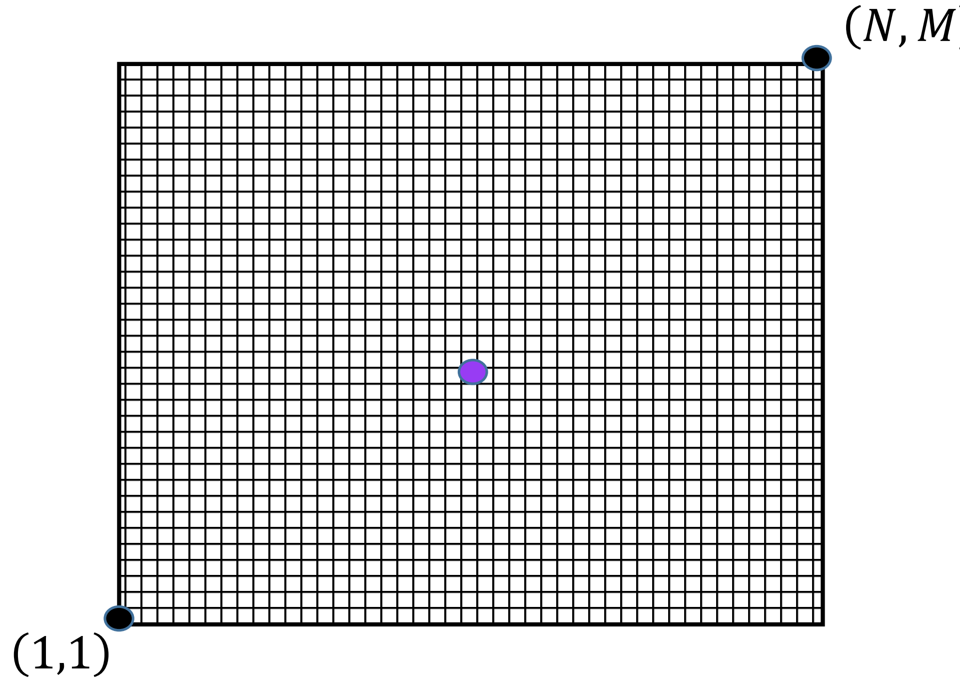


Image pixels



The image pixel positions define a rectangular grid in the projection plane.

We want to define a mapping from (x, y) to pixel positions (x_p, y_p).

We want to define a mapping from (x, y) to pixel positions (x_p, y_p).

Two issues:

- Units are different (mm versus pixel indices)

- The origins (0,0) are different.

We want to define a mapping from (x, y) to pixel positions (x_p, y_p).

Two issues:

- Units are different (mm versus pixel indices)

  Apply a scale transformation.

- The origins (0,0) are different.

  Apply a translation.

Units are different (mm versus pixel indices)

Apply a scale transformation.

$$(x, y) \rightarrow \left(m_x\, x,\ m_y\, y\right)$$

The scale factors m_x and m_y might not be exactly the same, i.e. rectangular rather than square lattice.

Units are different (mm versus pixel indices)

Apply a scale transformation, which would be represented in 2D homogeneous coordinates as:

$$\begin{bmatrix} m_x\, x \\ m_y\, y \\ 1 \end{bmatrix} = \begin{bmatrix} m_x & 0 & 0 \\ 0 & m_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

We want to define a mapping from (x, y) to pixel positions (x_p, y_p).

Two issues:

- Units are different (mm versus pixel indices)

  Apply a scale transformation.

- The origins (0,0) are different.

  Apply a translation.

projection plane

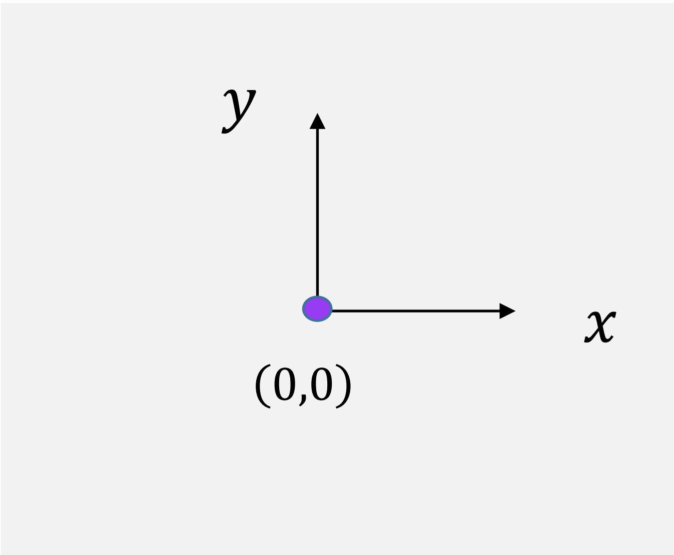
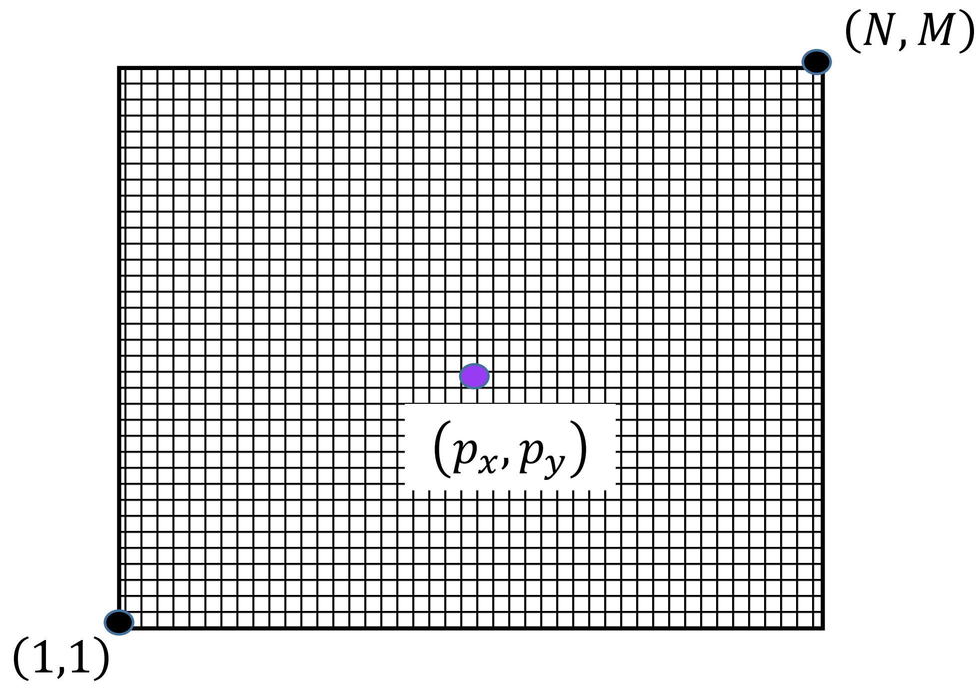


Image pixels



Let (p_x, p_y) be the pixel index that corresponds to $(x, y) = (0,0)$.

This position is called the "principal point". It might not correspond exactly to the center of the pixel grid.

We want to define a mapping from (x, y) to pixel positions (x_p, y_p).

The origins (0,0) are different.

Apply a translation.

$$\begin{bmatrix} m_x\, x + p_x \\ m_y\, y + p_y \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & p_x \\ 0 & 1 & p_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} m_x & 0 & 0 \\ 0 & m_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

(x_p, y_p)

How do 3D scene points (in world coordinates) map to pixel positions ?

1. Map from world coordinates to camera coordinates.

2. Project onto the projection plane.

3. Map from projection plane coordinates to pixel coordinates.

Let’s put these together into one transformation.

translation scaling projection

$$\begin{bmatrix} fm_x & 0 & p_x & 0 \\ 0 & fm_y & p_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & p_x \\ 0 & 1 & p_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} m_x & 0 & 0 \\ 0 & m_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

=

$$\begin{bmatrix} fm_x & 0 & p_x & 0 \\ 0 & fm_y & p_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} = \overset{\text{translation}}{\begin{bmatrix} 1 & 0 & p_x \\ 0 & 1 & p_y \\ 0 & 0 & 1 \end{bmatrix}} \overset{\text{scaling}}{\begin{bmatrix} m_x & 0 & 0 \\ 0 & m_y & 0 \\ 0 & 0 & 1 \end{bmatrix}} \overset{\text{projection}}{\begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}}$$

The 3x3 part is called the "camera calibration matrix" $\boldsymbol{K}$.

It invertible but the 3x4 matrix overall is not invertible since it includes projection.

Collapses the $f\ m_x$ to one constant.

Often used (allows for shear)

$$\mathbf{K} = \begin{bmatrix} \alpha_x & s & p_x \\ 0 & \alpha_y & p_y \\ 0 & 0 & 1 \end{bmatrix}$$

How do 3D scene points (in world coordinates) map to pixel positions ?

1. Map from world coordinates to camera coordinates.

2. Project onto the projection plane.

3. Map from projection plane coordinates to pixel coordinates.

$$\mathbf{P} = \mathbf{K}\,\mathbf{R}\,[\,\mathbf{I}\,|\,-\mathbf{C}\,]$$

3x4 3x3 3x3 3x4

intrinsic
(what are the internal properties of the camera?)

extrinsic
(what are the external properties of the camera ie. position and orientation of camera in world coordinates?)

$$\mathbf{P} = \mathbf{K}\,\overbrace{\mathbf{R}\,[\,\mathbf{I}\,|\,-\mathbf{C}\,]}$$

The matrix $\mathbf{P}$ is called the “finite projective camera” model.