Camera Matrices

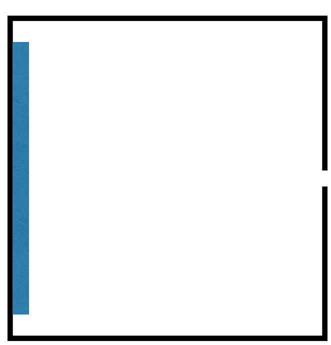
Math Lecture 3

Nicholas Dwork

Central Projective Camera

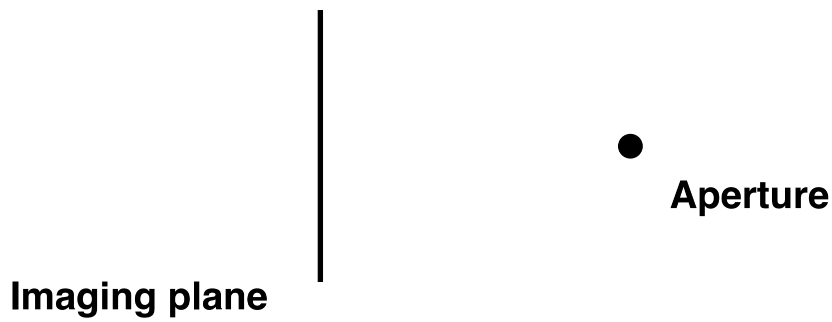
Imaging plane in back collects light.

Extremely tiny hole (called the aperture) lets light in.



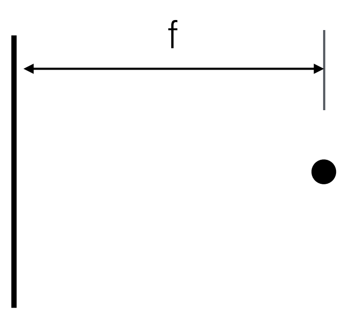
Box prevents stray rays of light from hitting the imaging plane.

We represent the central projective camera with a line and a dot.



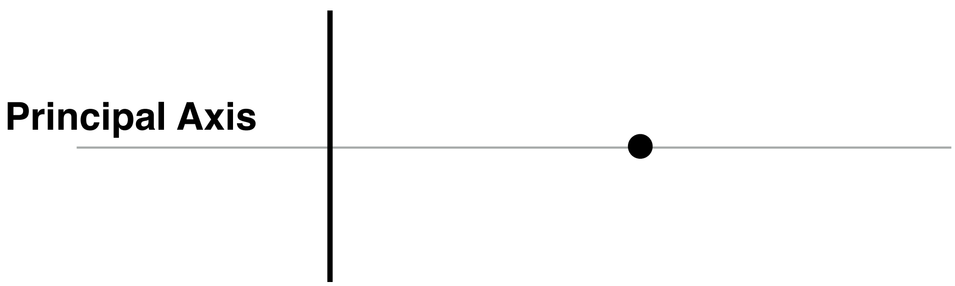
Focal Length

The focal length is the distance between the imaging plane and the aperture.

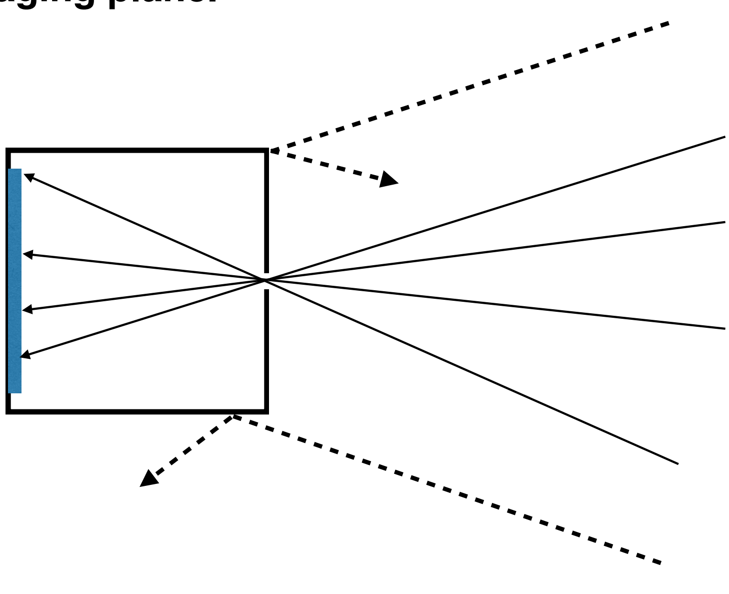


Principal Axis

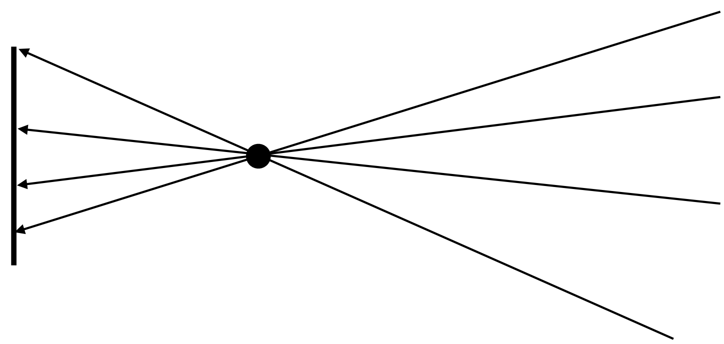
The line that passes through the aperture and is perpendicular to the imaging plane is called the Principal Axis.



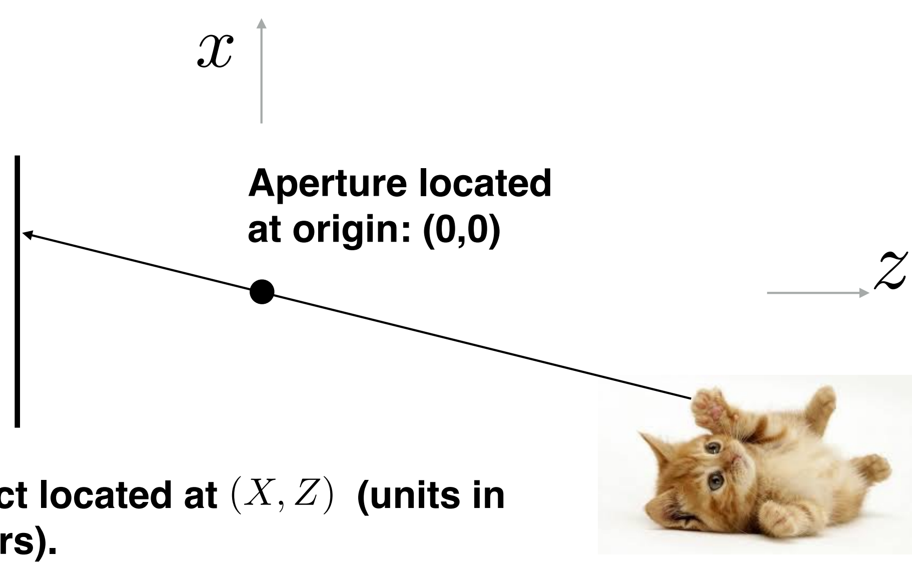
Light rays pass through the aperture and hit the imaging plane.



Light rays pass through the aperture and hit the imaging plane.

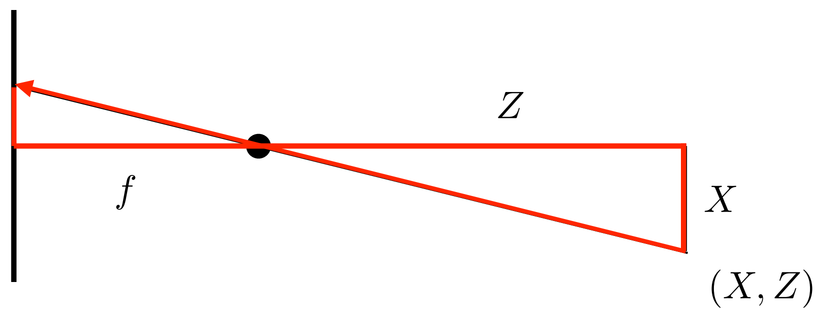


Light ray reflects off of object onto the imaging plane.

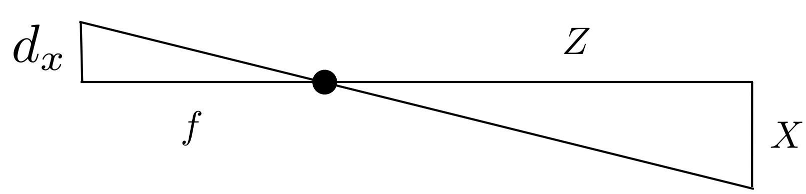


Object located at (X, Z) (units in meters).

Do you see the *similar triangles*?

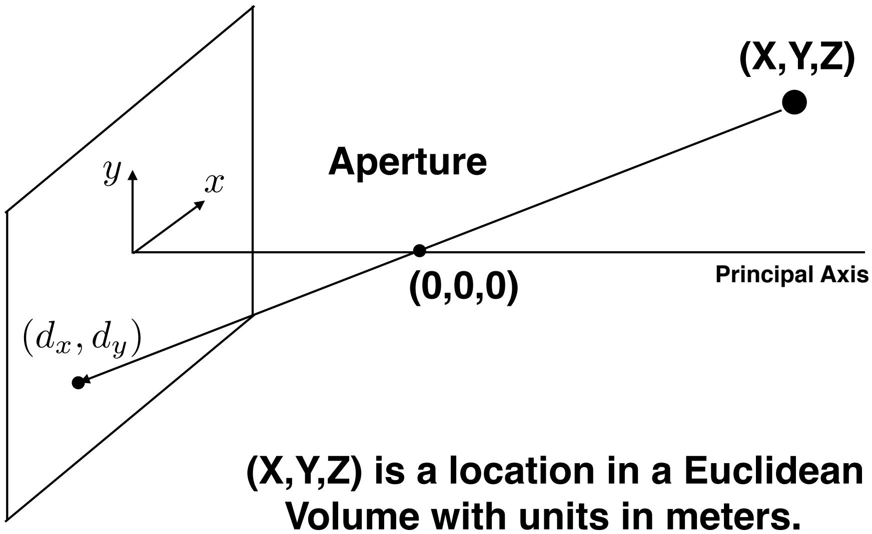


9



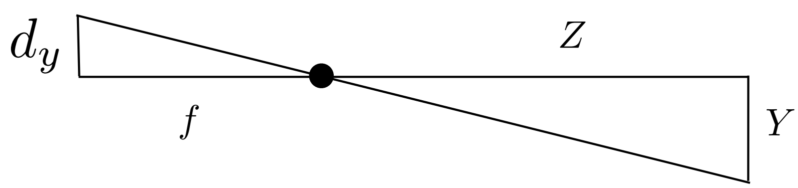
$$\frac{d_x}{f} = \frac{X}{Z} \qquad d_x = f\frac{X}{Z}$$

10



(X,Y,Z) is a location in a Euclidean Volume with units in meters.

(d_x, d_y) **is a location on the image plane in meters.**



$$\frac{d_y}{f} = \frac{Y}{Z} \qquad\qquad d_y = f\frac{Y}{Z}$$

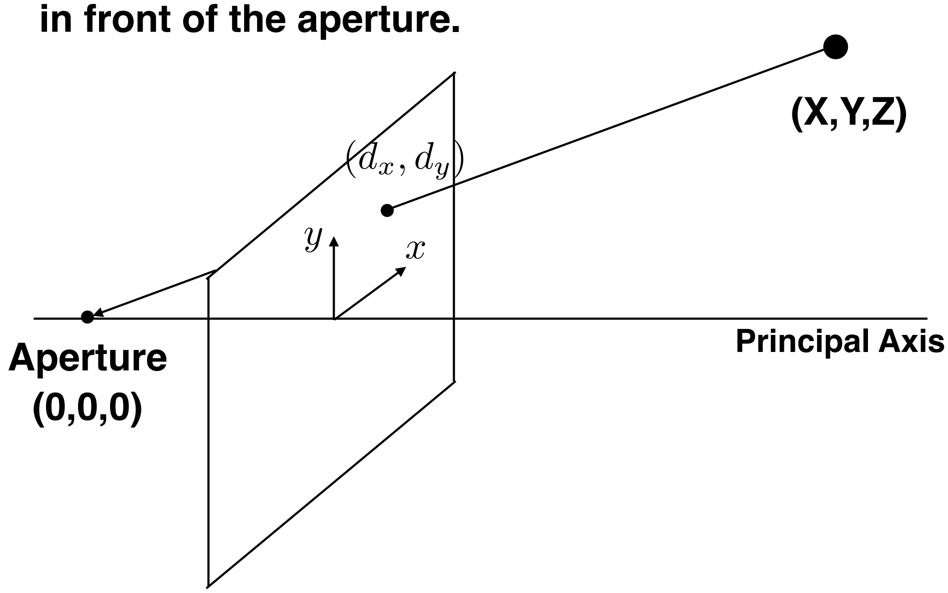
These equations can be combined into a single matrix equation.

$$\begin{bmatrix}\tilde{d}_x \\ \tilde{d}_y \\ \tilde{d}_z\end{bmatrix} = \begin{bmatrix} fX \\ fY \\ Z \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

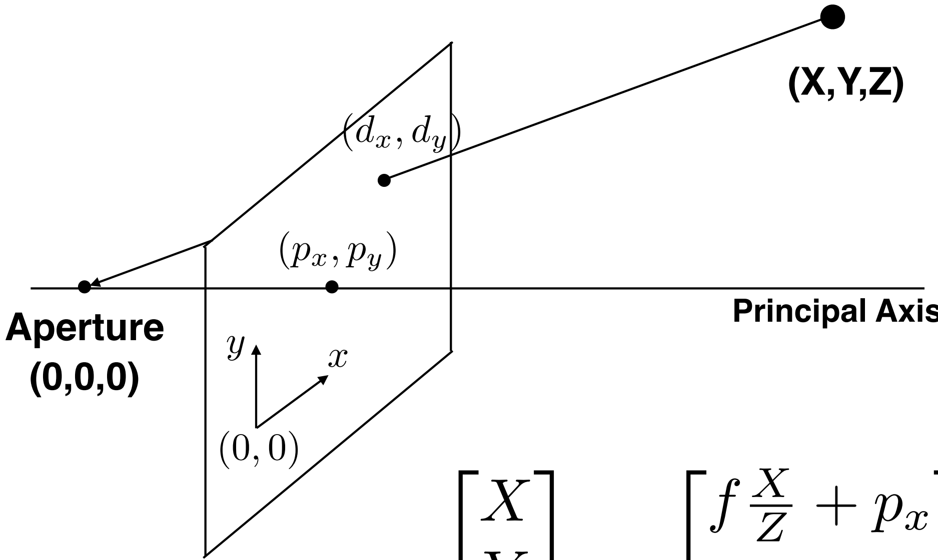
In the above equation, 3D homogeneous coordinates are mapped to 2D homogeneous coordinates.

$$(d_x, d_y) = \left(\frac{\tilde{d}_x}{\tilde{d}_z}, \frac{\tilde{d}_y}{\tilde{d}_z} \right)$$

For convenience, we can move the image plane in front of the aperture.



From now on, we'll assume the image plane is in front of the aperture.



$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \mapsto \begin{bmatrix} f\,\frac{X}{Z} + p_x \\ \\ f\,\frac{Y}{Z} + p_y \end{bmatrix}$$

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \mapsto \begin{bmatrix} f\,\frac{X}{Z} + p_x \\ \\ f\,\frac{Y}{Z} + p_y \end{bmatrix}$$

We can form this mapping as a matrix equation.

$$\begin{bmatrix} \tilde{d}_x \\ \tilde{d}_y \\ \tilde{d}_z \end{bmatrix} = \begin{bmatrix} f & 0 & p_x & 0 \\ 0 & f & p_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

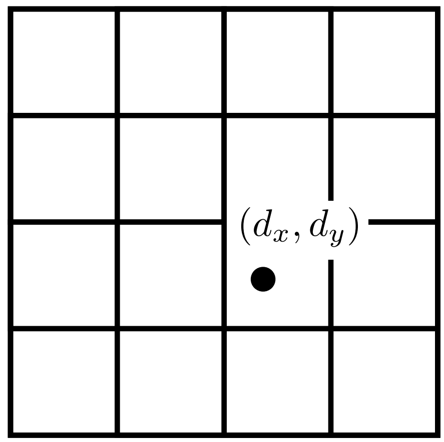
The dimensions of (d_x, d_y) are currently in meters.

We may want to convert to "pixel units".

Suppose (d_x, d_y) equals (0.23,0.14) meters.

Suppose further we know that pixels are 0.1 meters in size horizontally and vertically.

Then we can say that (d_x, d_y) is (2.3,1.4) pixels in size.



To convert to pixel units:

$$\begin{bmatrix} \tilde{d}_x \\ \tilde{d}_y \\ \tilde{d}_z \end{bmatrix} = \begin{bmatrix} m_x & 0 & 0 \\ 0 & m_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} f & 0 & p_x & 0 \\ 0 & f & p_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

where, $m_x = s_x^{-1}$ $\quad$ $m_y = s_y^{-1}$

s_x is the horizontal size of a pixel in meters.

s_y is the vertical size of a pixel in meters.

$$\begin{bmatrix}\tilde{d}_x\\ \tilde{d}_y\\ \tilde{d}_z\end{bmatrix} = \begin{bmatrix} m_x & 0 & 0\\ 0 & m_y & 0\\ 0 & 0 & 1\end{bmatrix} \begin{bmatrix} f & 0 & p_x & 0\\ 0 & f & p_y & 0\\ 0 & 0 & 1 & 0\end{bmatrix} \begin{bmatrix} X\\ Y\\ Z\\ 1\end{bmatrix}$$

$$= \begin{bmatrix} \alpha_x & 0 & x_0 & 0\\ 0 & \alpha_y & y_0 & 0\\ 0 & 0 & 1 & 0\end{bmatrix} \begin{bmatrix} X\\ Y\\ Z\\ 1\end{bmatrix}$$

where

$$\alpha_x = f m_x \quad \alpha_y = f m_y \quad x_0 = m_x p_x \quad y_0 = m_y p_y$$

$$\begin{bmatrix}\tilde{d}_x\\ \tilde{d}_y\\ \tilde{d}_z\end{bmatrix} = \begin{bmatrix} \alpha_x & 0 & x_0 & 0\\ 0 & \alpha_y & y_0 & 0\\ 0 & 0 & 1 & 0\end{bmatrix} \begin{bmatrix} X\\ Y\\ Z\\ 1\end{bmatrix}$$

We can rewrite this expression as follows:

$$\begin{bmatrix}\tilde{d}_x\\ \tilde{d}_y\\ \tilde{d}_z\end{bmatrix} = K \begin{bmatrix} I & 0\end{bmatrix} \begin{bmatrix} X\\ Y\\ Z\\ 1\end{bmatrix}, \quad K = \begin{bmatrix} \alpha_x & 0 & x_0\\ 0 & \alpha_y & y_0\\ 0 & 0 & 1\end{bmatrix}$$

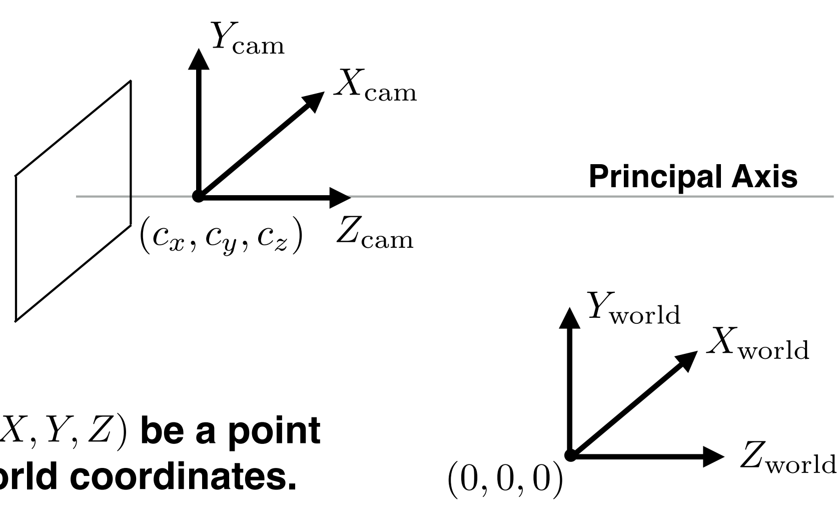
K **is called the intrinsic camera matrix.**

What do we do if the camera is not located at the origin, but rather located at some other location pointed in some other direction?

Answer: we change the coordinates of all the points so that it looks to the camera like the camera is located at the origin pointing along the z axis.

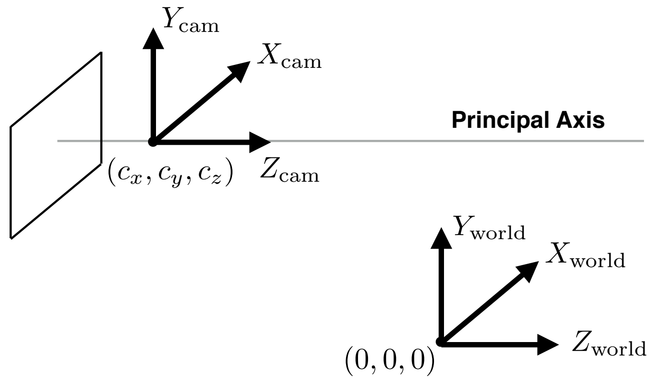
“We change from world coordinates to camera coordinates.”

We translate (or shift) the camera by an amount C.



Let (X, Y, Z) be a point in world coordinates.

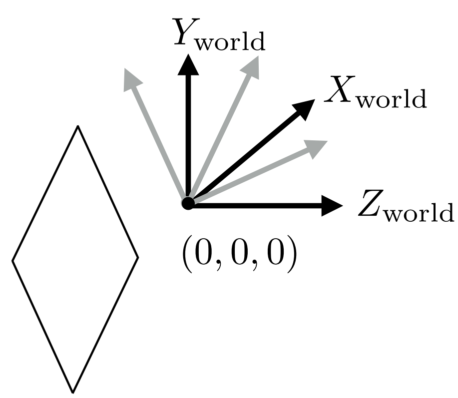
How do we convert from world to camera coordinates?



$$\begin{bmatrix} X_{\text{cam}} \\ Y_{\text{cam}} \\ Z_{\text{cam}} \end{bmatrix} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} - \begin{bmatrix} c_x \\ c_y \\ c_z \end{bmatrix}$$

23

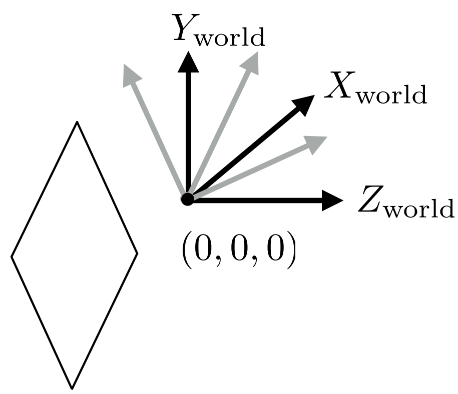
We rotate the camera by some roll, pitch, and yaw.



Let (X, Y, Z) be a point in world coordinates.

How do we convert from world to camera coordinates?

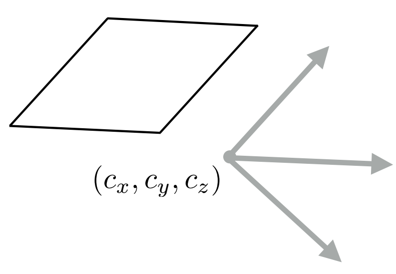
24

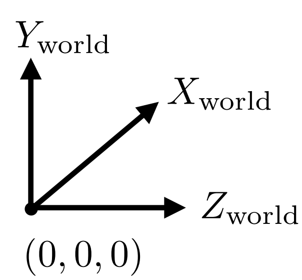


$$\begin{bmatrix} X_{\text{cam}} \\ Y_{\text{cam}} \\ Z_{\text{cam}} \end{bmatrix} = R \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

R is the appropriate rotation matrix.

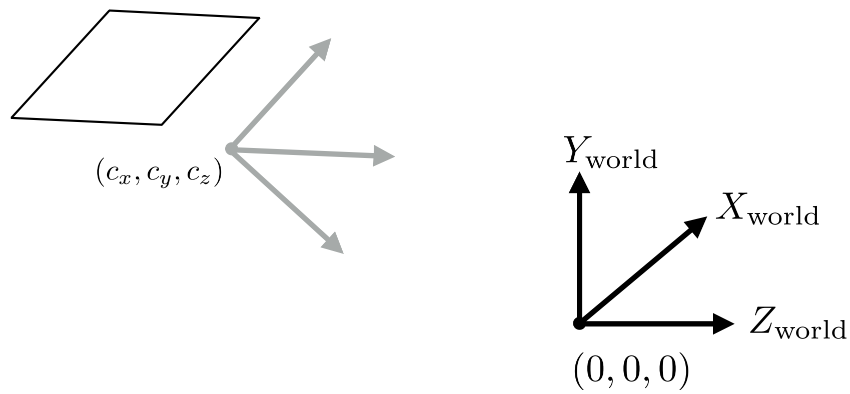
We translate (or shift) and rotate the camera by an amount C.





Let (X, Y, Z) be a point in world coordinates.

How do we convert from world to camera coordinates?



$$\begin{bmatrix} X_{\text{cam}} \\ Y_{\text{cam}} \\ Z_{\text{cam}} \end{bmatrix} = R \left(\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} - \begin{bmatrix} c_x \\ c_y \\ c_z \end{bmatrix} \right)$$

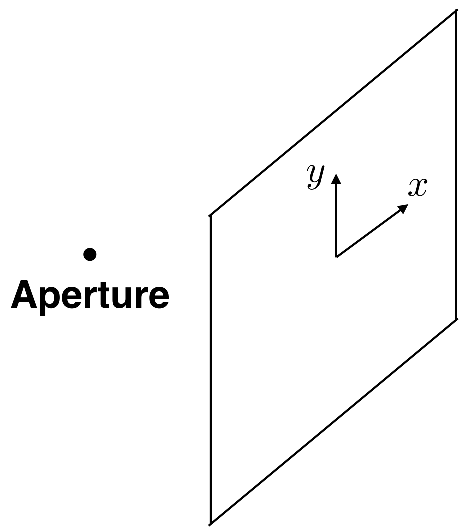
We can combine all equations to convert world coordinates into pixel units

$$P = K\,R\,\begin{bmatrix} I & -C \end{bmatrix}$$

where $C = \begin{bmatrix} c_x \\ c_y \\ c_z \end{bmatrix}$

Final Note

Once you've determined the location (d_x, d_y), you must convert this into computer pixel units.



The way we've define our axis, x points left and y points up.

In Matlab, x points right and y points down. You'll need to do an appropriate conversion.