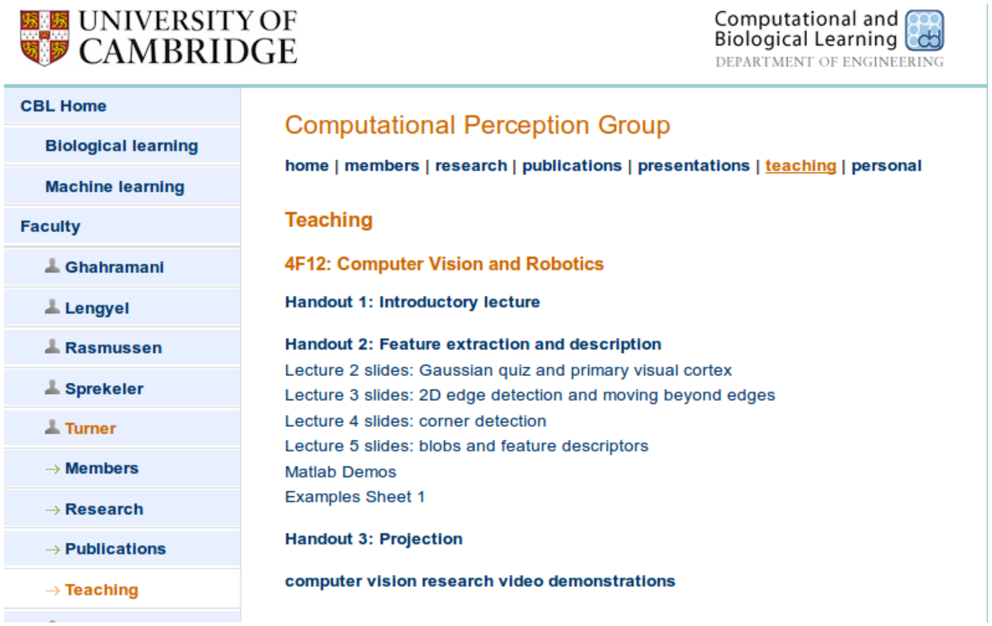
# Lecture 12: stereo vision

Dr. Richard E. Turner (ret26@cam.ac.uk)

November 18, 2013

# House keeping

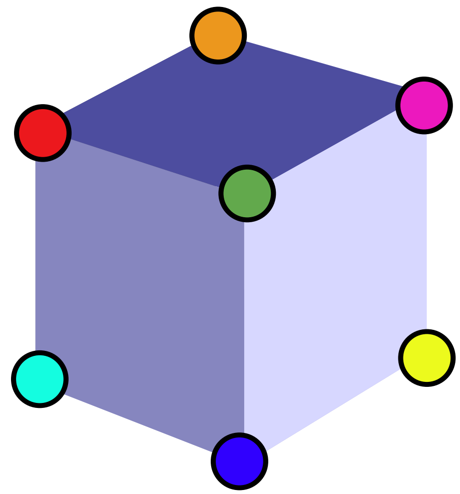



- Hawkeye: Tim Harley `tim.harley@hawkeyeinnovations.co.uk`

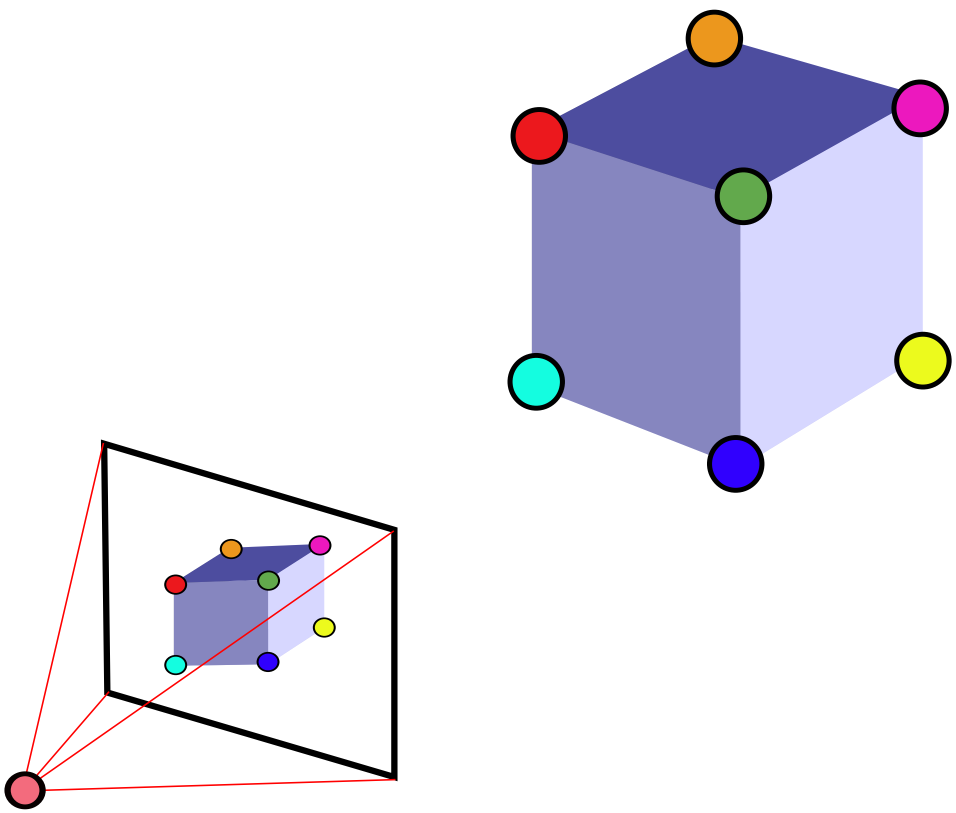
- exam information `http://teaching.eng.cam.ac.uk/content/exam-information-students`

- webpage: `http://cbl.eng.cam.ac.uk/Public/Turner/Teaching`

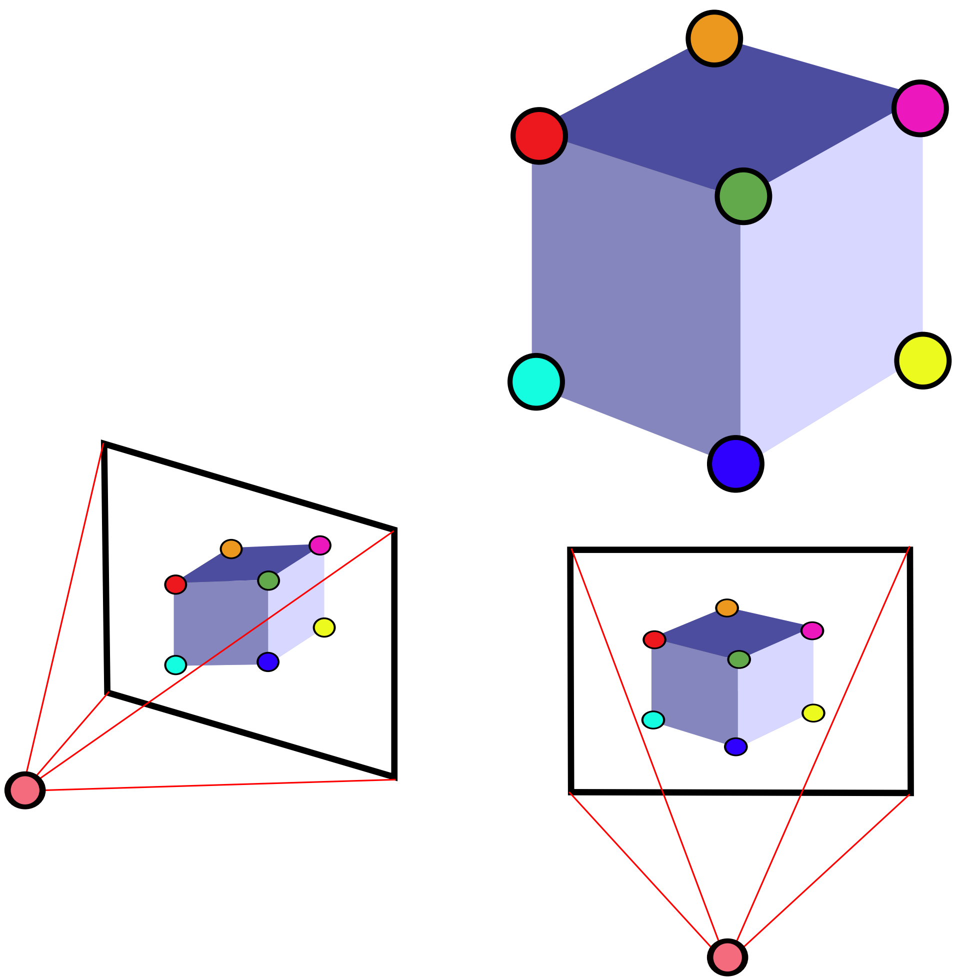
# Main idea: reconstruction from multiple images



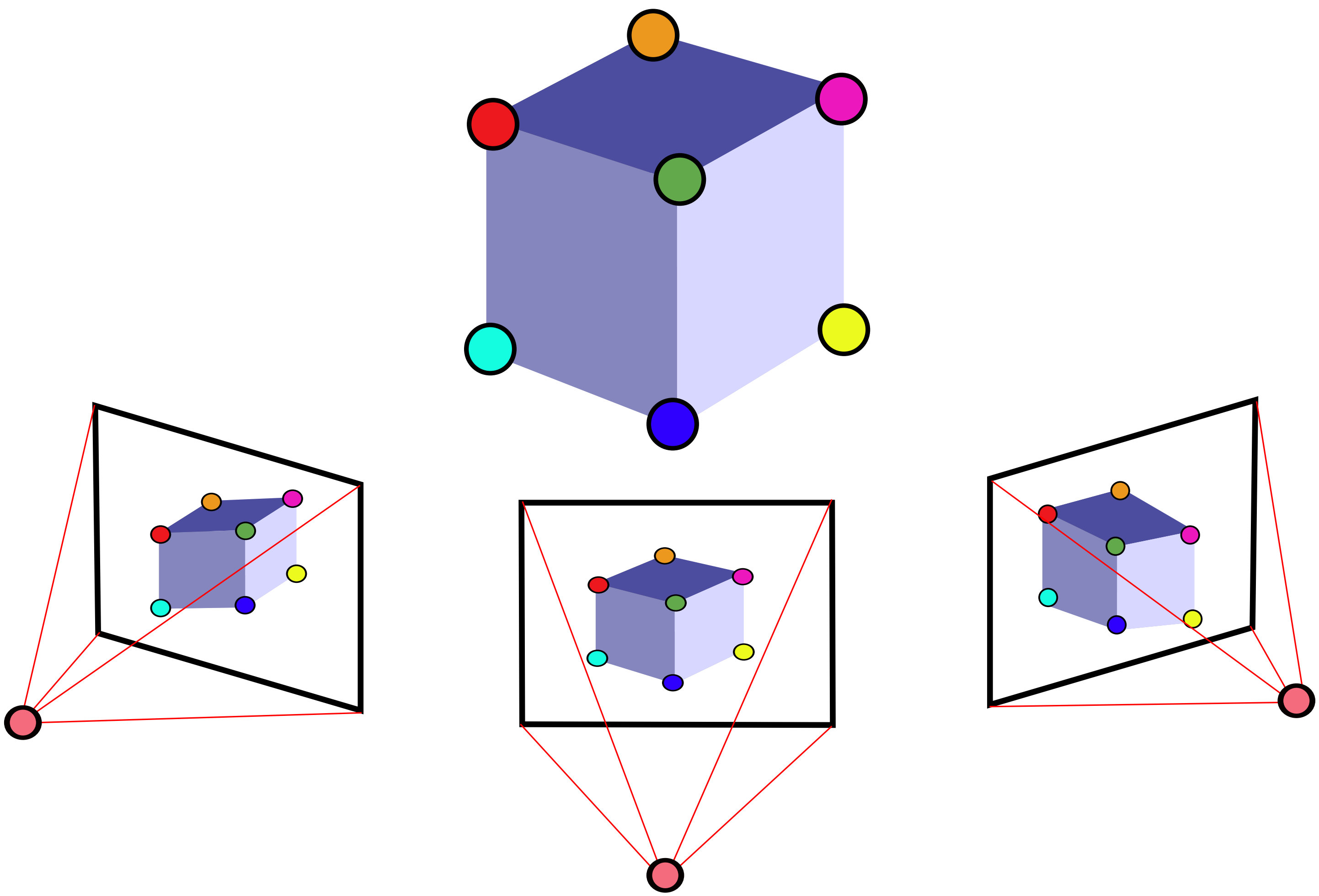
# Main idea: collect images from different views



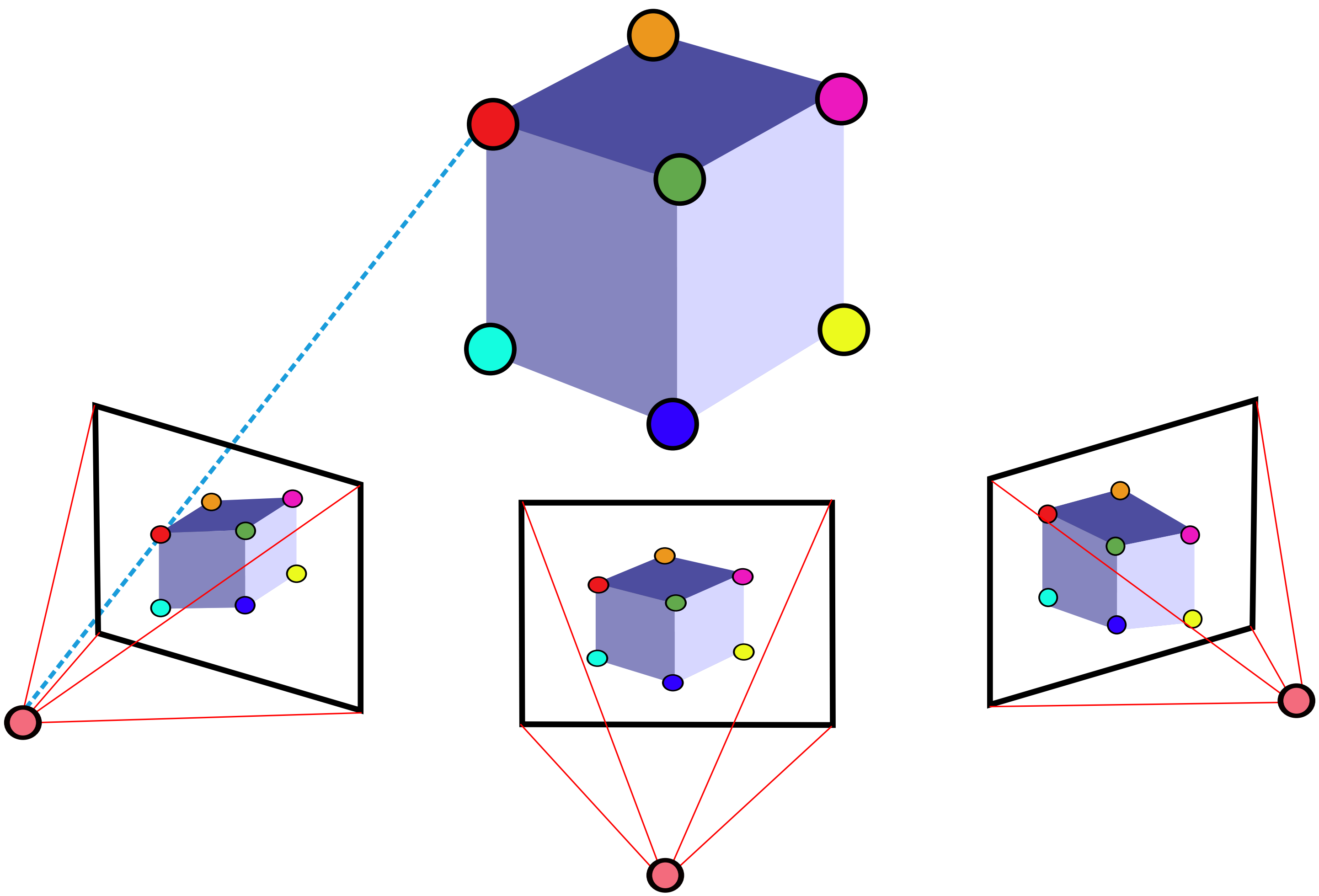
# Main idea: collect images from different views



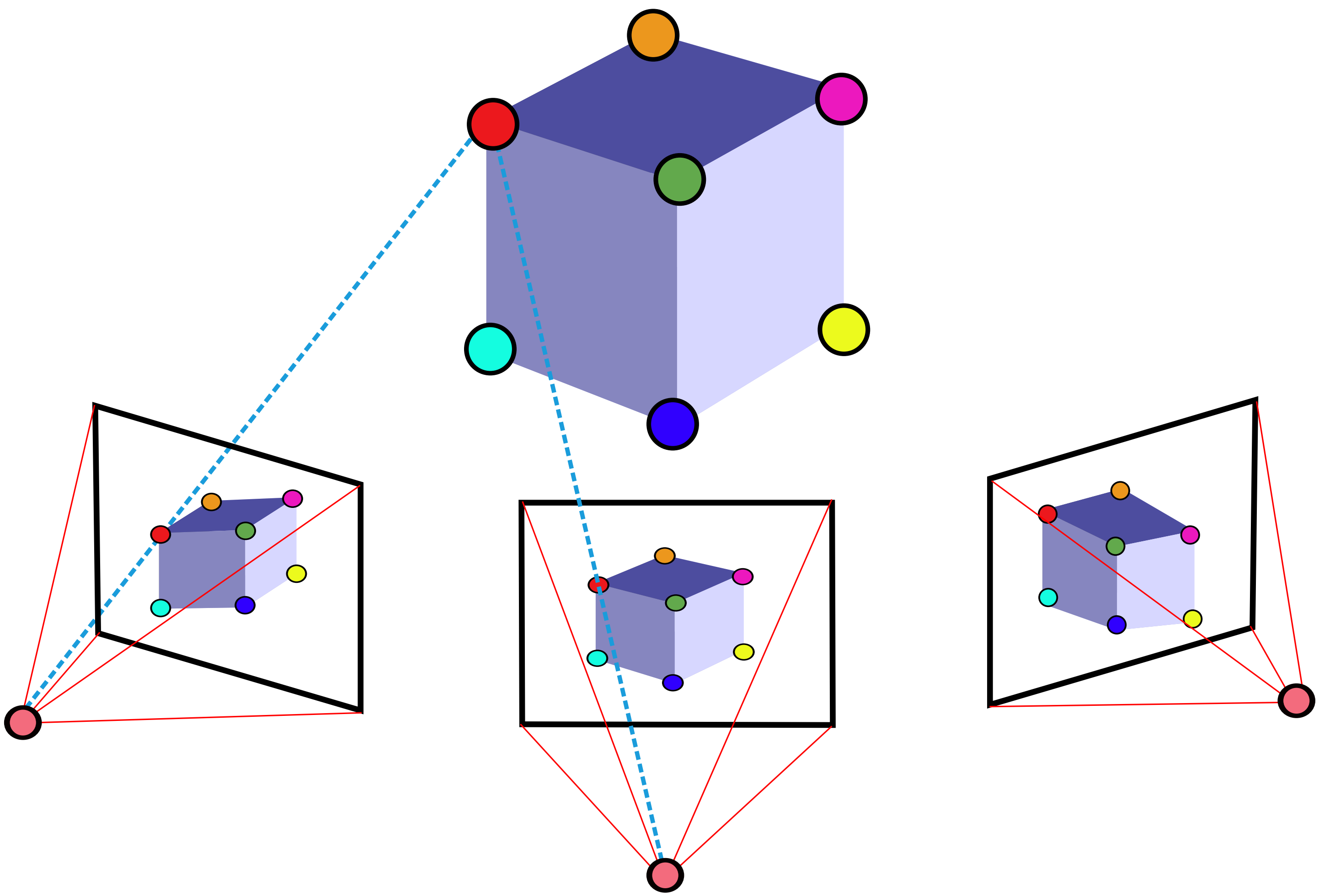
# Main idea: collect images from different views



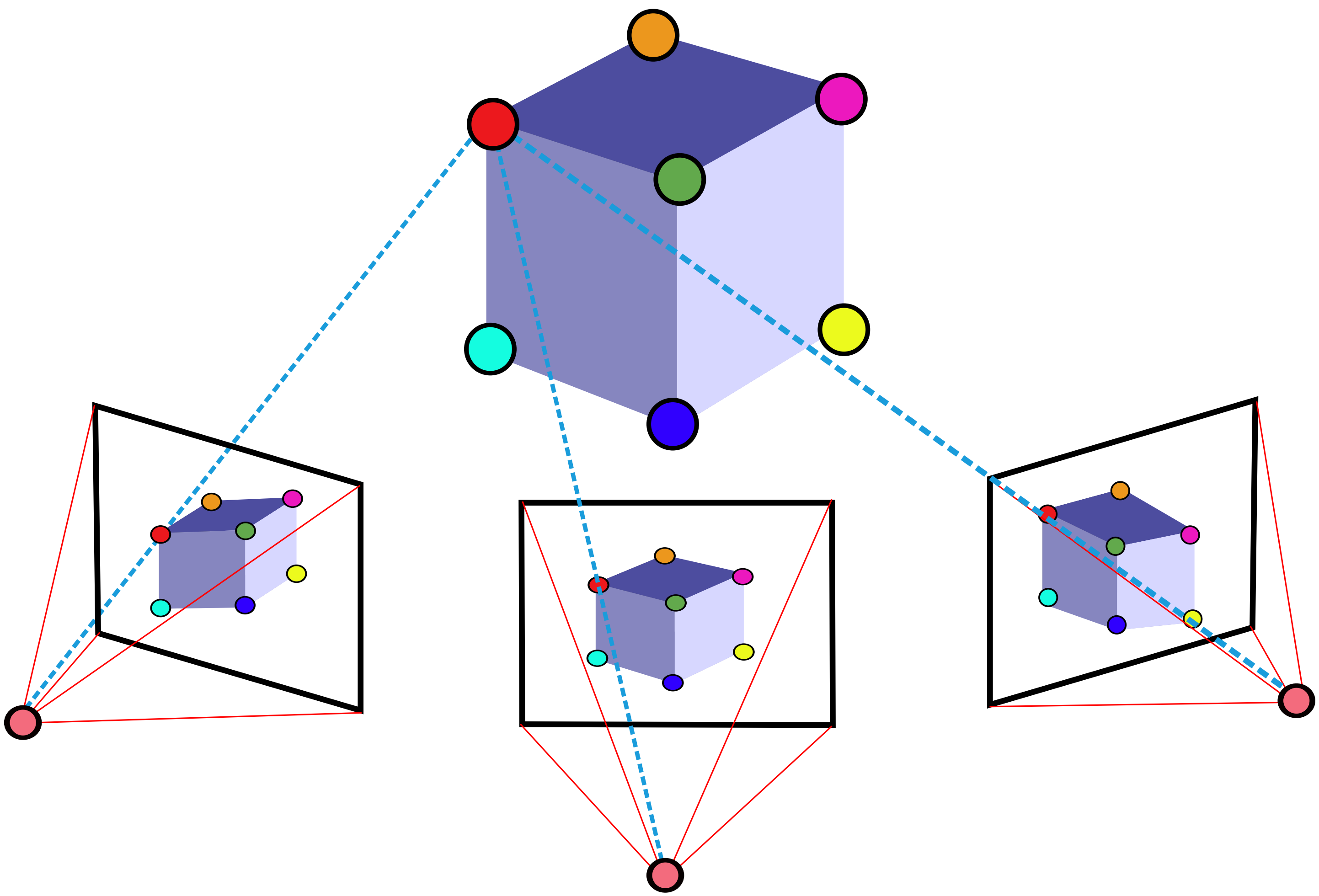
# Main idea: establish correspondences and triangulate



# Main idea: establish correspondences and triangulate



# Main idea: establish correspondences and triangulate



# Main idea: establish correspondences and triangulate

- how do we find the correspondences efficiently
- how do we triangulate
  - to what degree do the cameras need to be calibrated?
  - how do we handle errors?

# Main idea: establish correspondences and triangulate

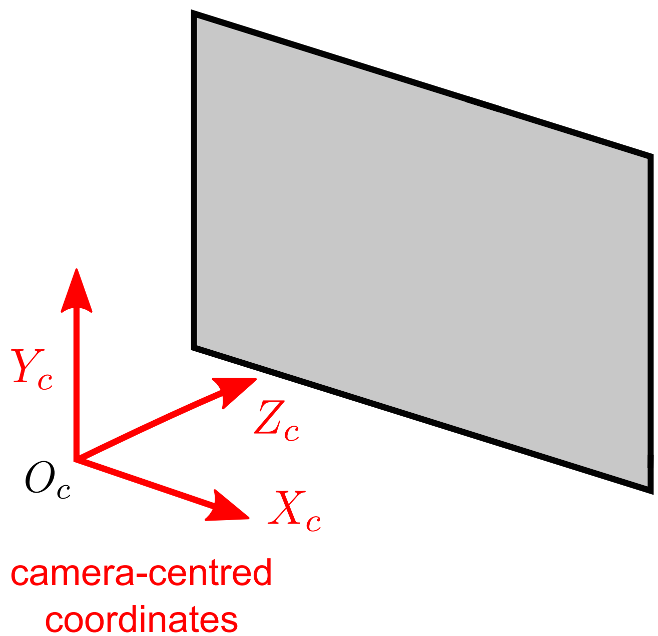
- how do we find the correspondences efficiently
- how do we triangulate
  - to what degree do the cameras need to be calibrated?
  - how do we handle errors?

$\Longrightarrow$ first need to understand the geometry of stereo vision

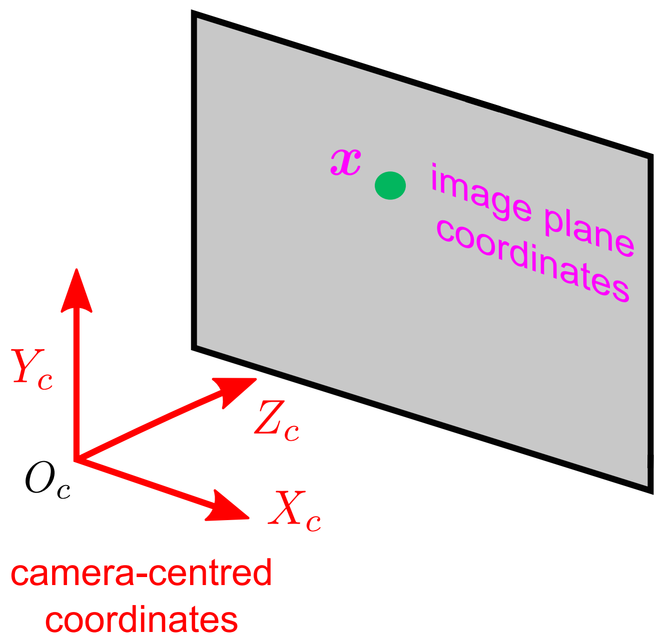
# Outline

- We’re going to figure out the relationship between pairs of points in stereo images
- working in cartesian coordinates (to begin with)
- operate mainly in camera centred coordinates

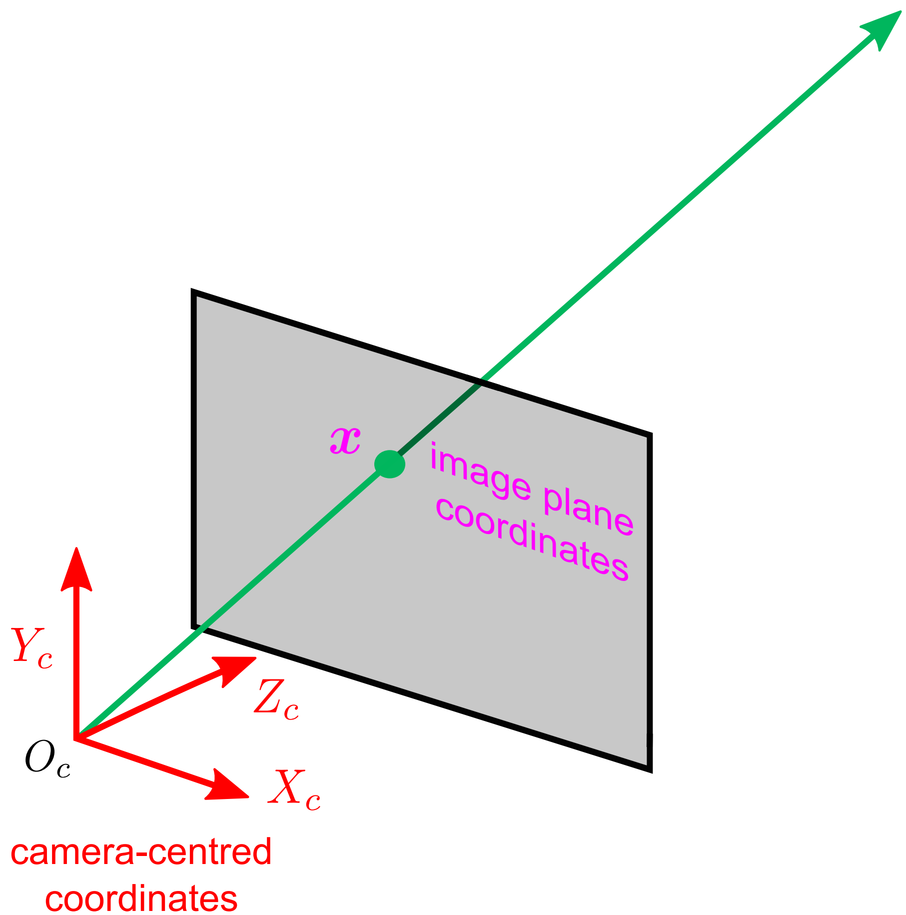
# Rays



# Rays

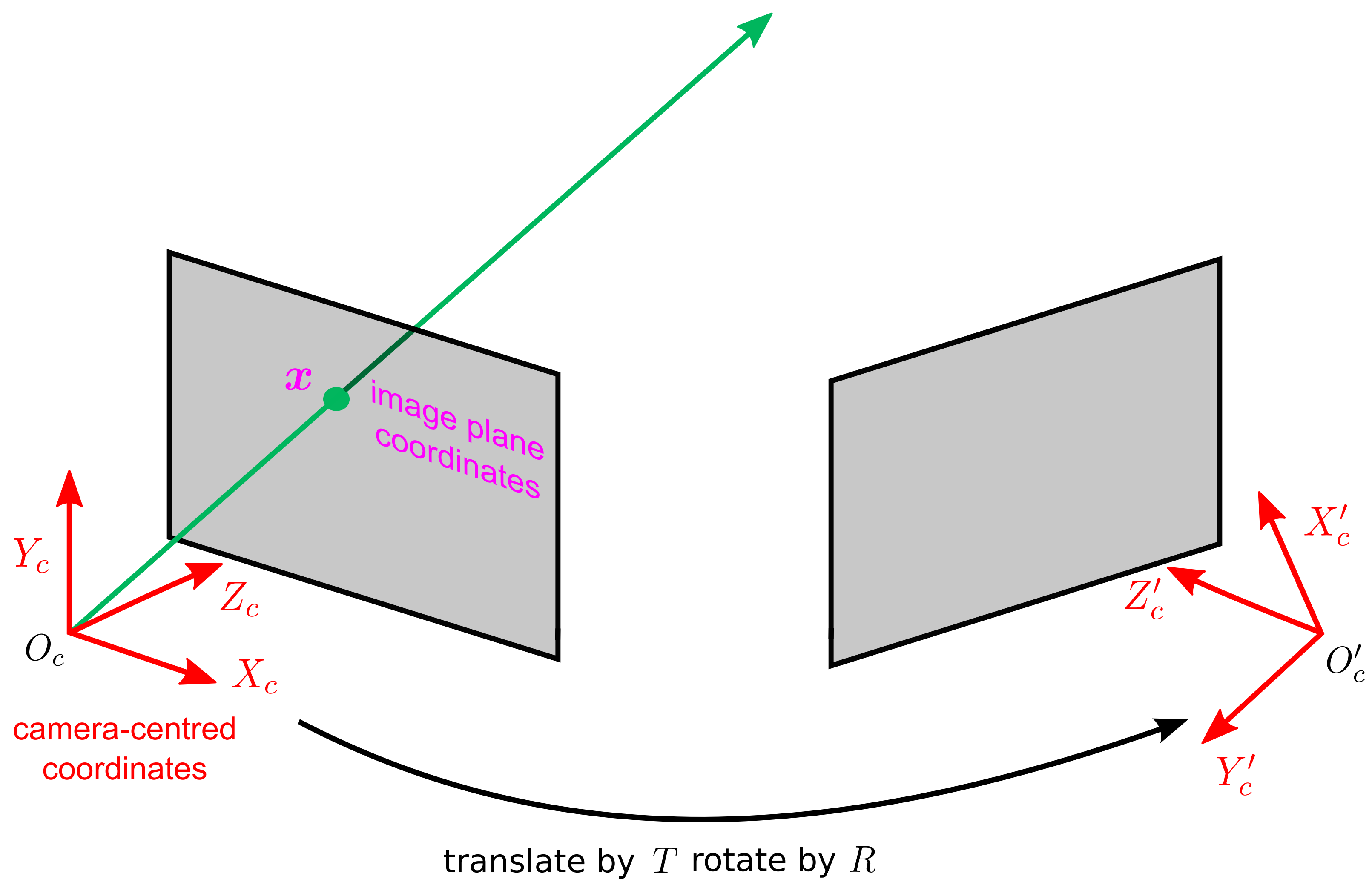

$\boldsymbol{x}$
image plane
coordinates
$Y_c$
$Z_c$
$O_c$
$X_c$
camera-centred
coordinates

# Rays



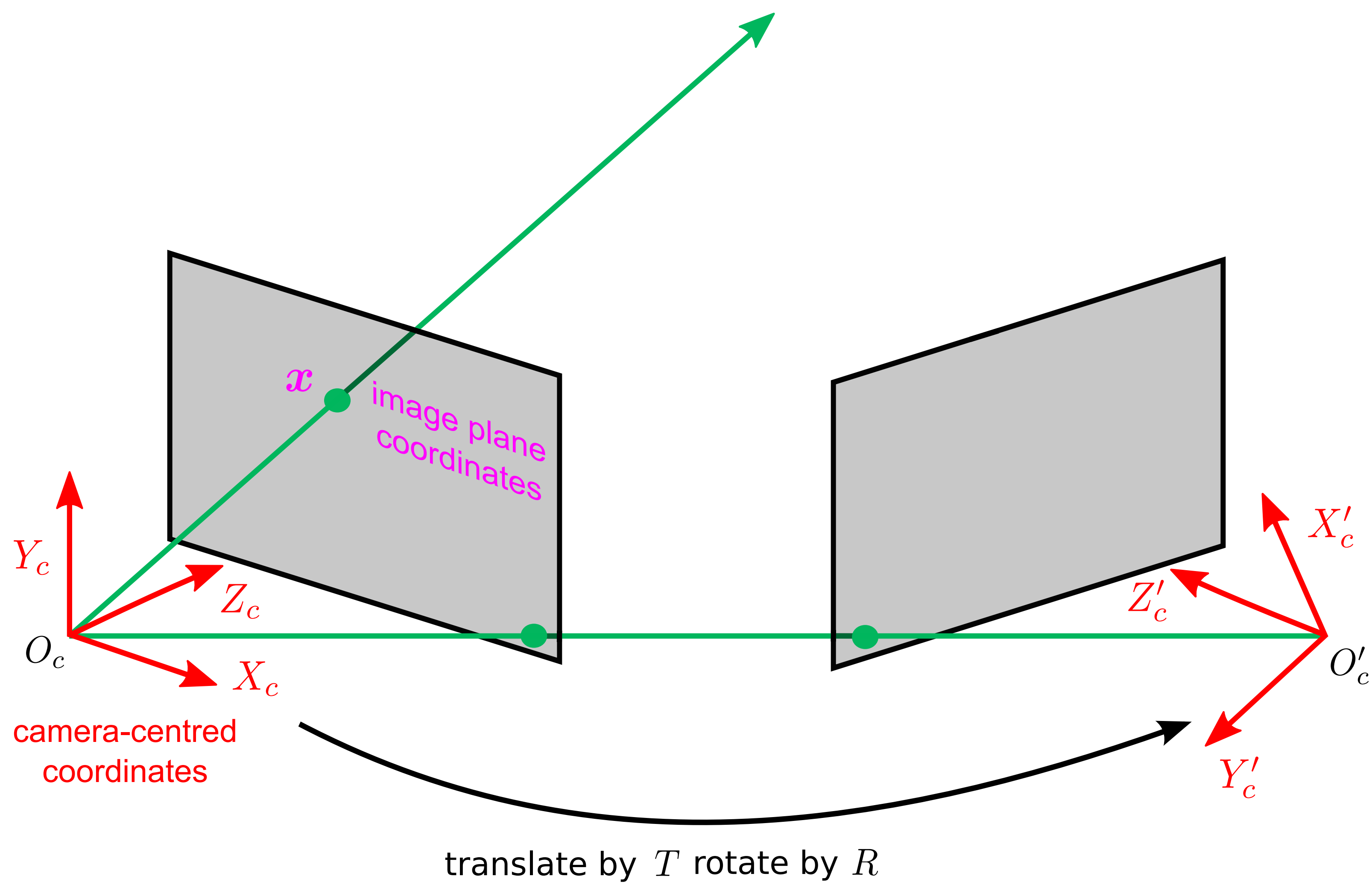
$\boldsymbol{x}$
image plane coordinates
$Y_c$
$Z_c$
$O_c$
$X_c$
camera-centred coordinates

# Rays



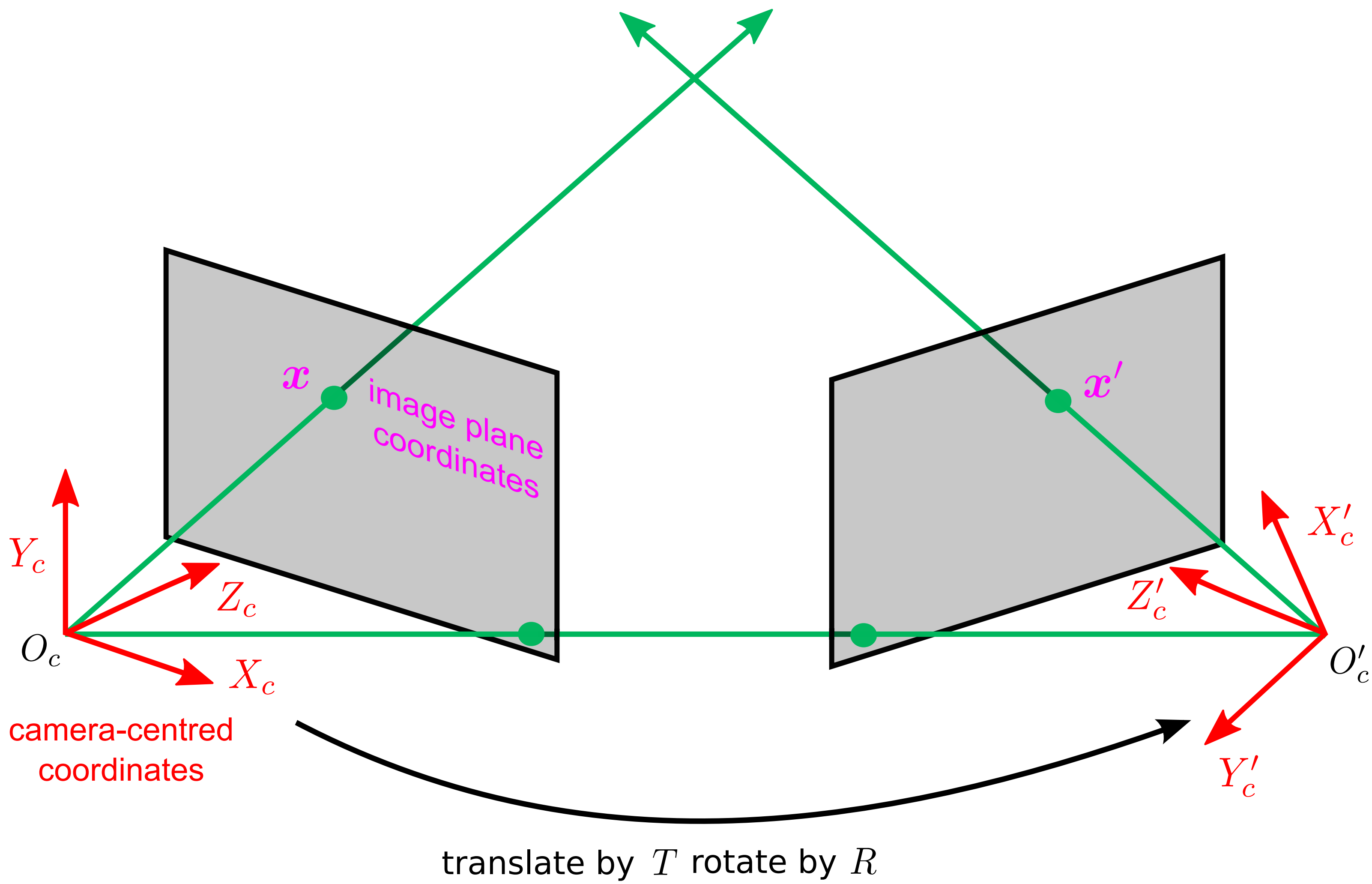
$\boldsymbol{x}$
image plane coordinates
$Y_c$
$Z_c$
$O_c$
$X_c$
camera-centred coordinates
$X'_c$
$Z'_c$
$O'_c$
$Y'_c$
translate by $T$ rotate by $R$

# Rays

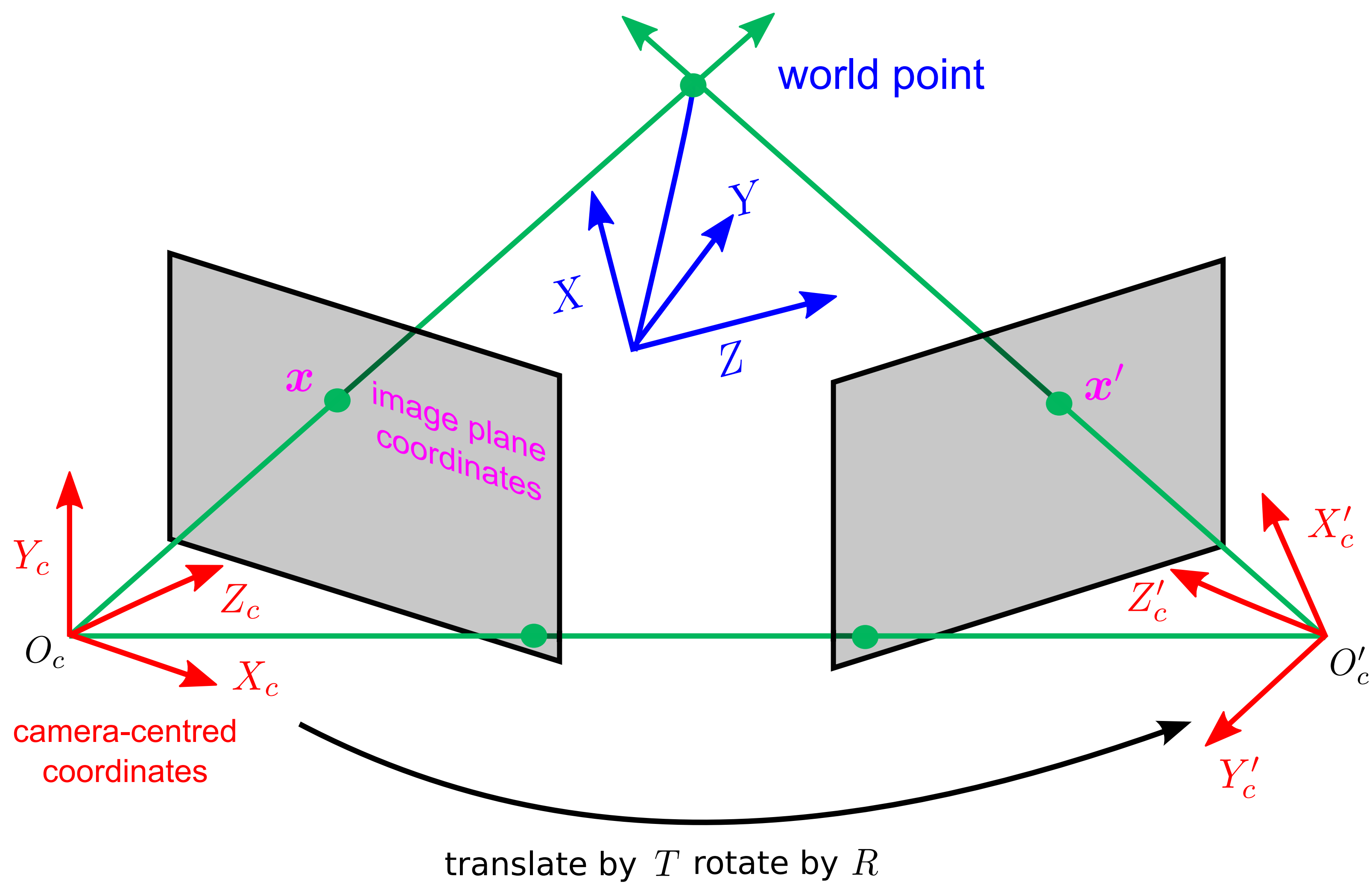


$\boldsymbol{x}$
image plane coordinates
$Y_c$
$Z_c$
$O_c$
$X_c$
camera-centred coordinates
$X'_c$
$Z'_c$
$O'_c$
$Y'_c$
translate by $T$ rotate by $R$

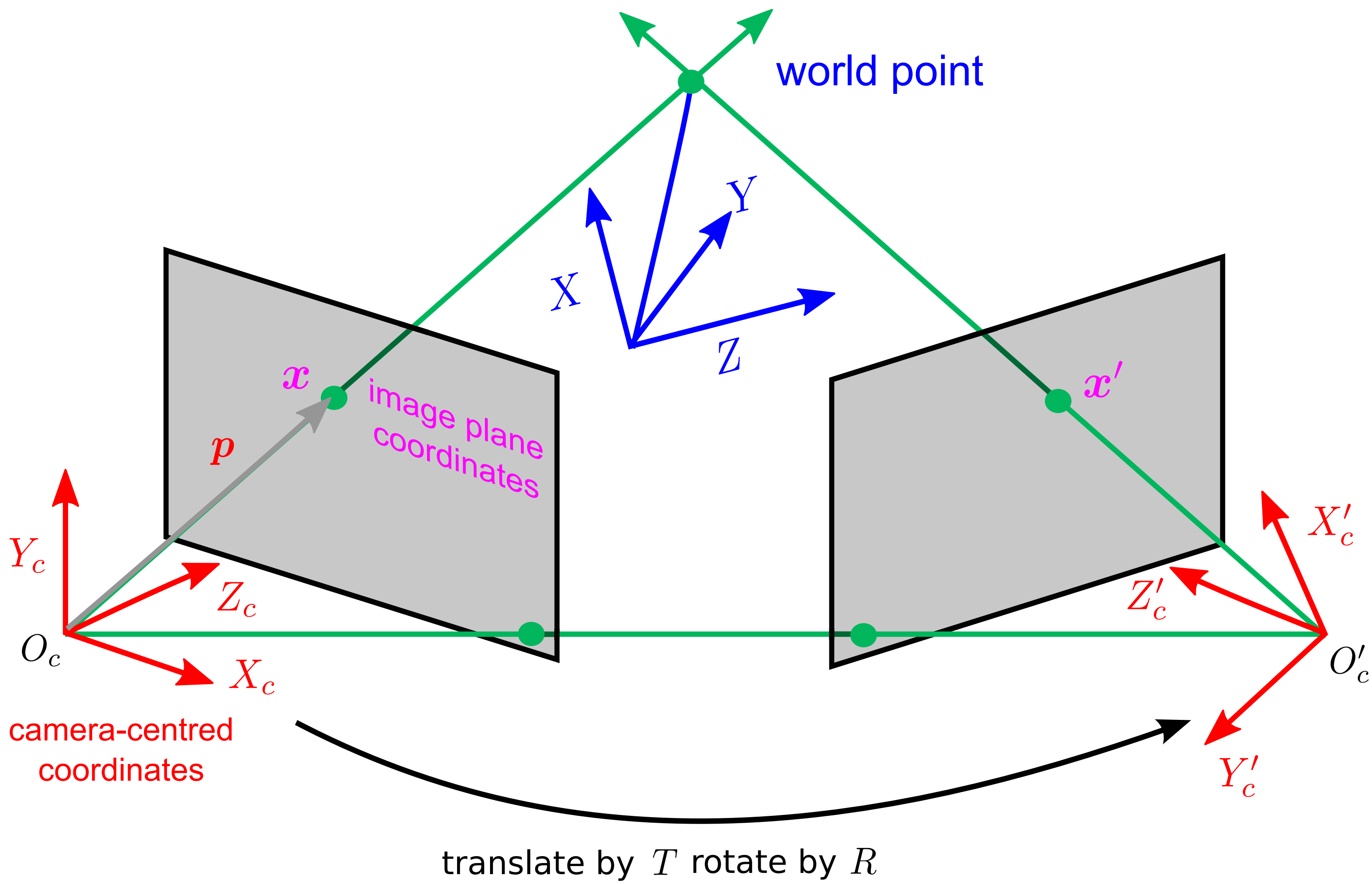
# Rays


$\boldsymbol{x}$
$\boldsymbol{x}'$
image plane
coordinates
$Y_c$
$Z_c$
$X_c$
$O_c$
$X'_c$
$Z'_c$
$O'_c$
$Y'_c$
camera-centred
coordinates
translate by $T$ rotate by $R$

# Rays

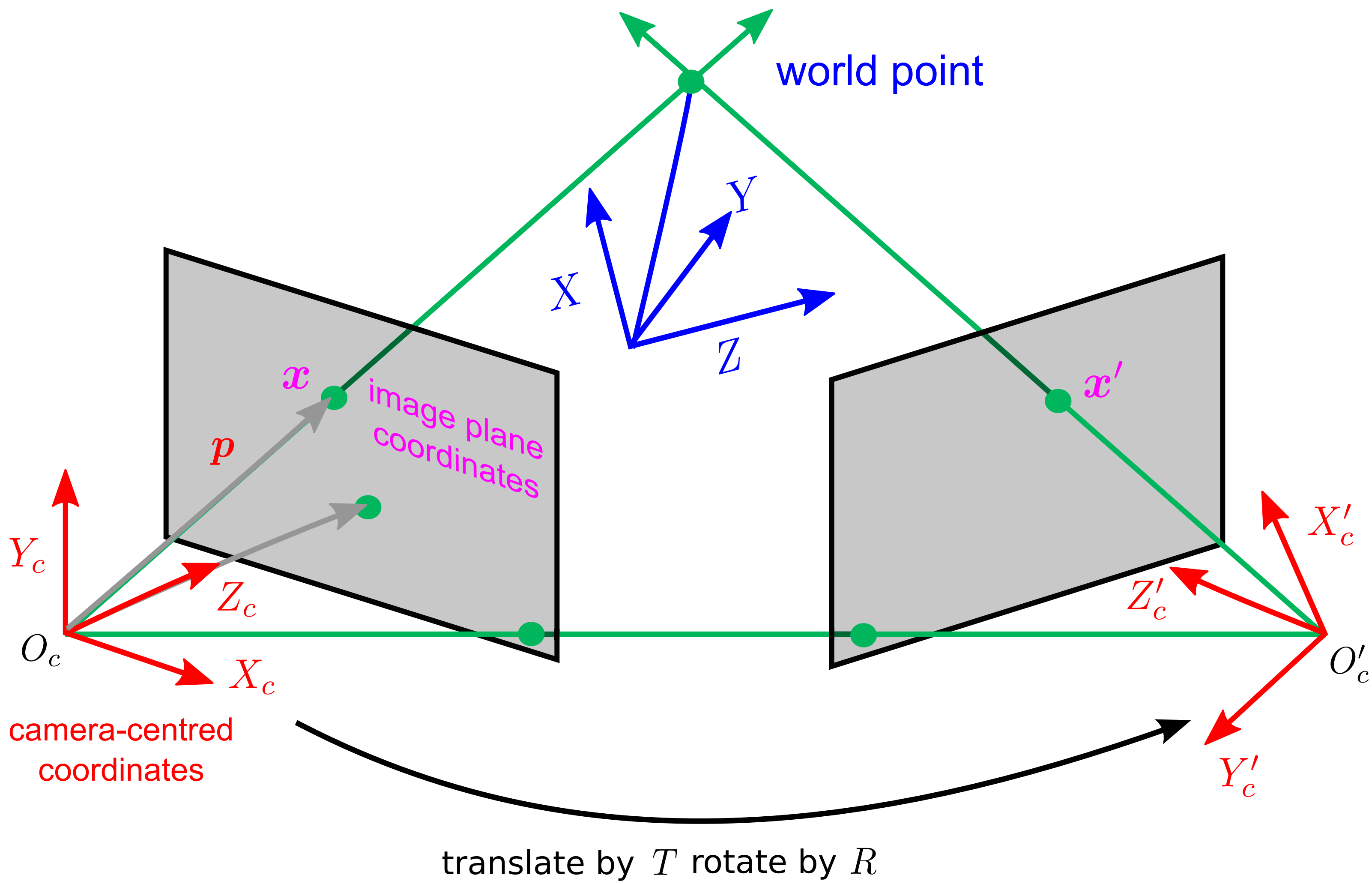

world point
X
Y
Z
$\boldsymbol{x}$
$\boldsymbol{x}'$
image plane coordinates
$Y_c$
$Z_c$
$O_c$
$X_c$
$X'_c$
$Z'_c$
$O'_c$
$Y'_c$
camera-centred coordinates
translate by $T$ rotate by $R$

# Rays



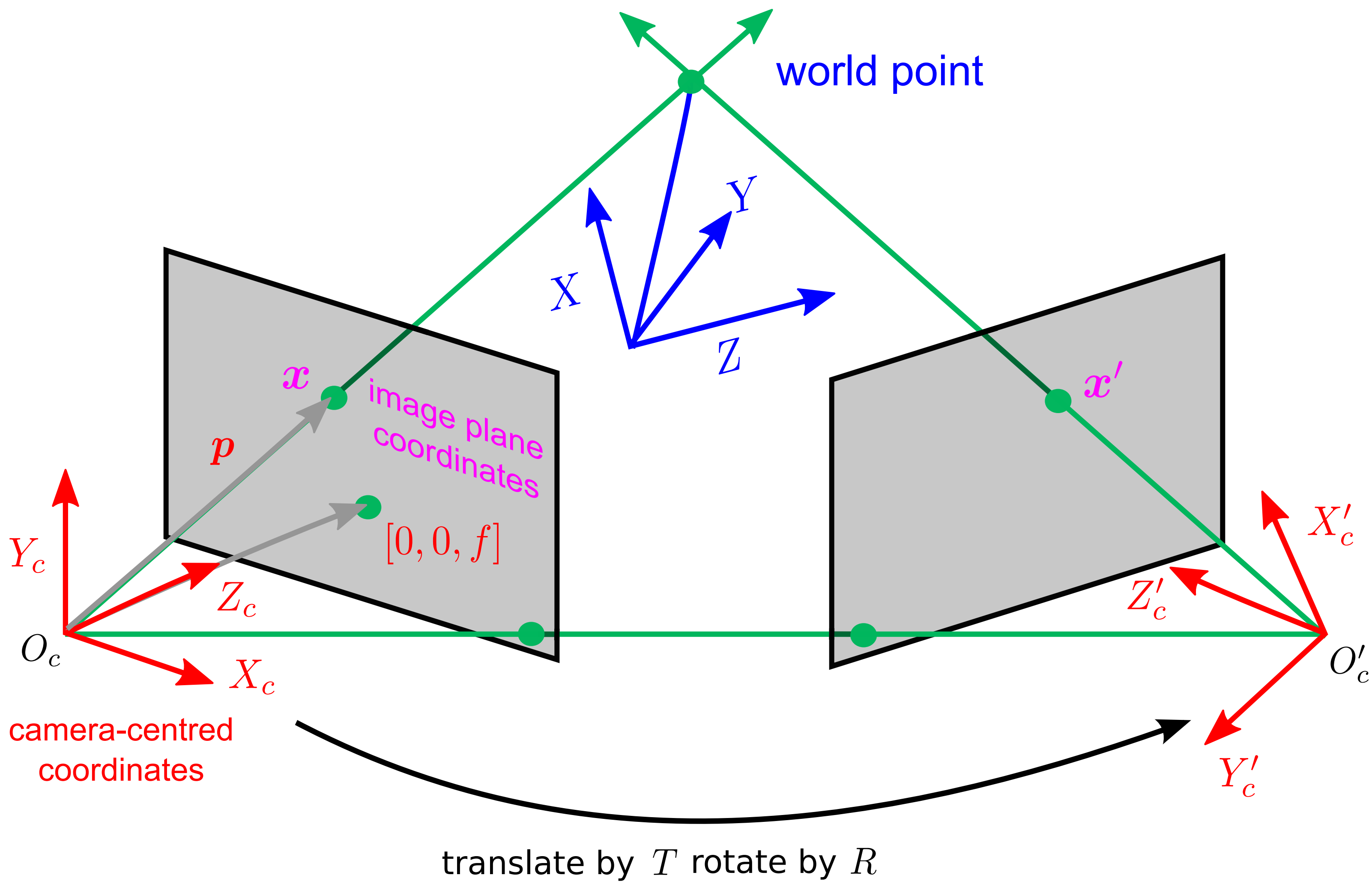
world point
X
Y
Z
$\boldsymbol{x}$
$\boldsymbol{x}'$
image plane coordinates
$\boldsymbol{p}$
$Y_c$
$Z_c$
$O_c$
$X_c$
$X'_c$
$Z'_c$
$O'_c$
$Y'_c$
camera-centred coordinates
translate by $T$ rotate by $R$

# Rays



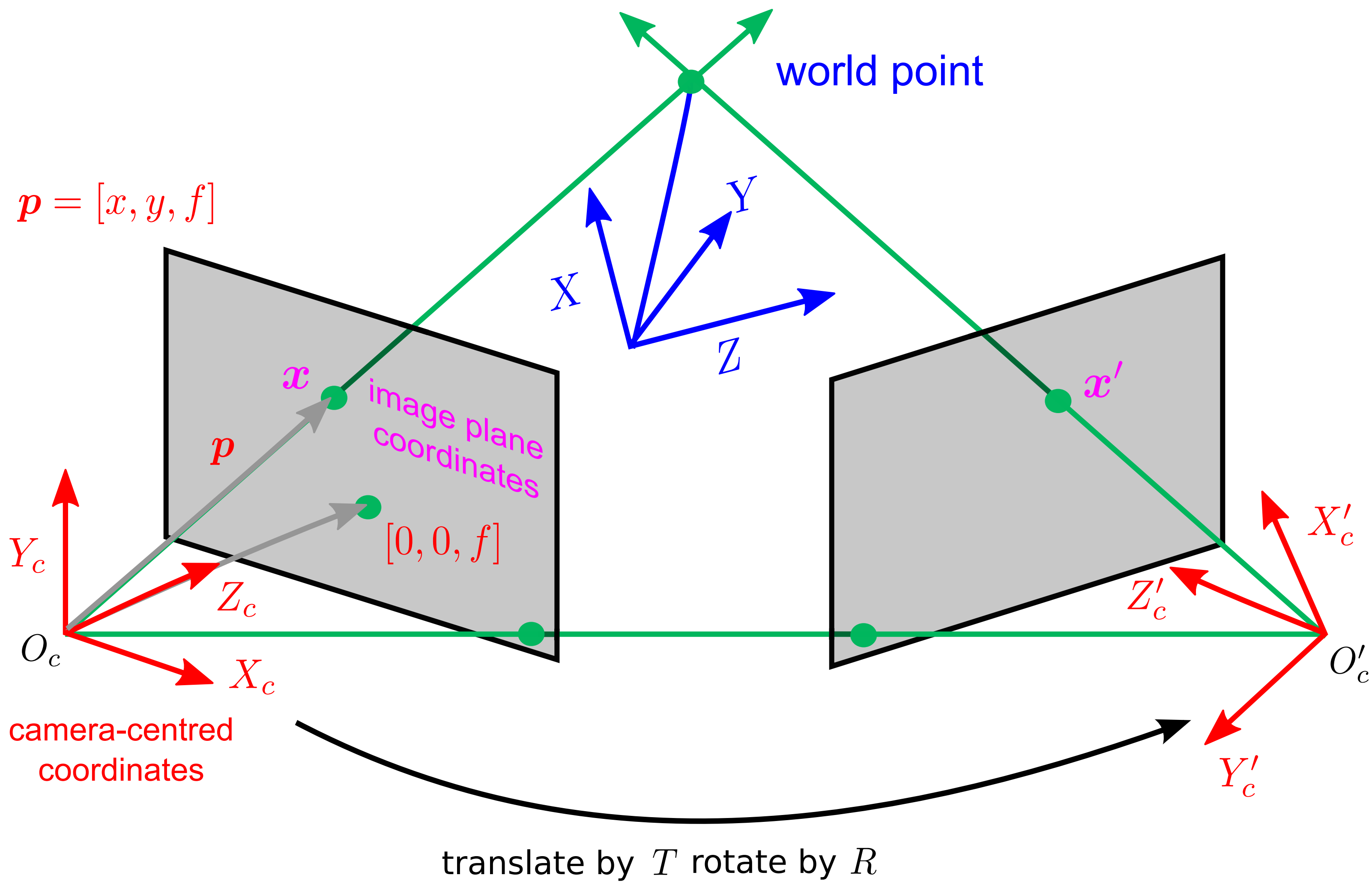
world point
X
Y
Z
$\boldsymbol{x}$
$\boldsymbol{x}'$
image plane
coordinates
$\boldsymbol{p}$
$Y_c$
$Z_c$
$O_c$
$X_c$
$X_c'$
$Z_c'$
$O_c'$
$Y_c'$
camera-centred
coordinates
translate by $T$ rotate by $R$

# Rays



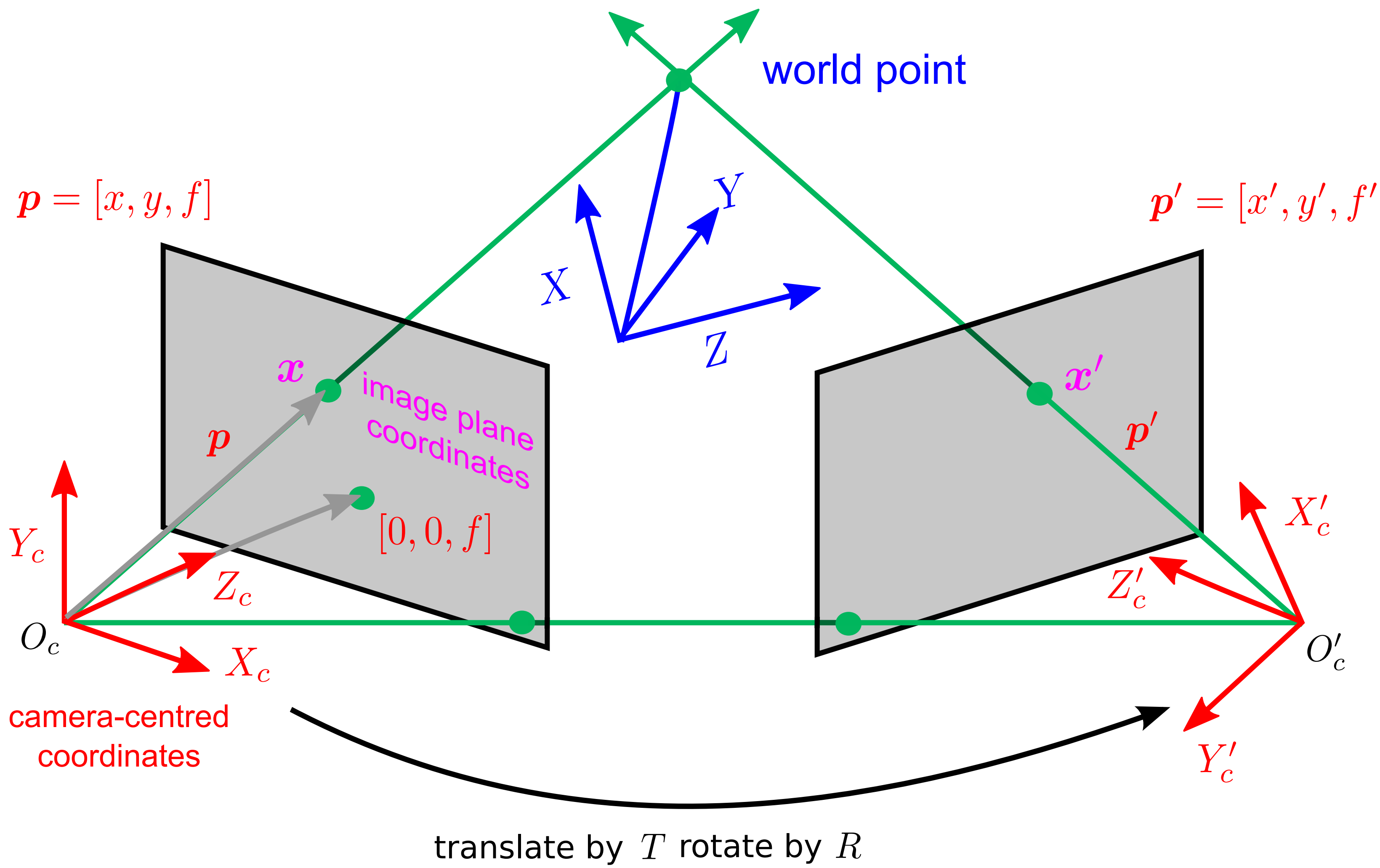
world point
X
Y
Z
$\boldsymbol{x}$
$\boldsymbol{x}'$
image plane coordinates
$\boldsymbol{p}$
$[0, 0, f]$
$Y_c$
$Z_c$
$O_c$
$X_c$
$X'_c$
$Z'_c$
$O'_c$
$Y'_c$
camera-centred coordinates
translate by $T$ rotate by $R$

# Rays



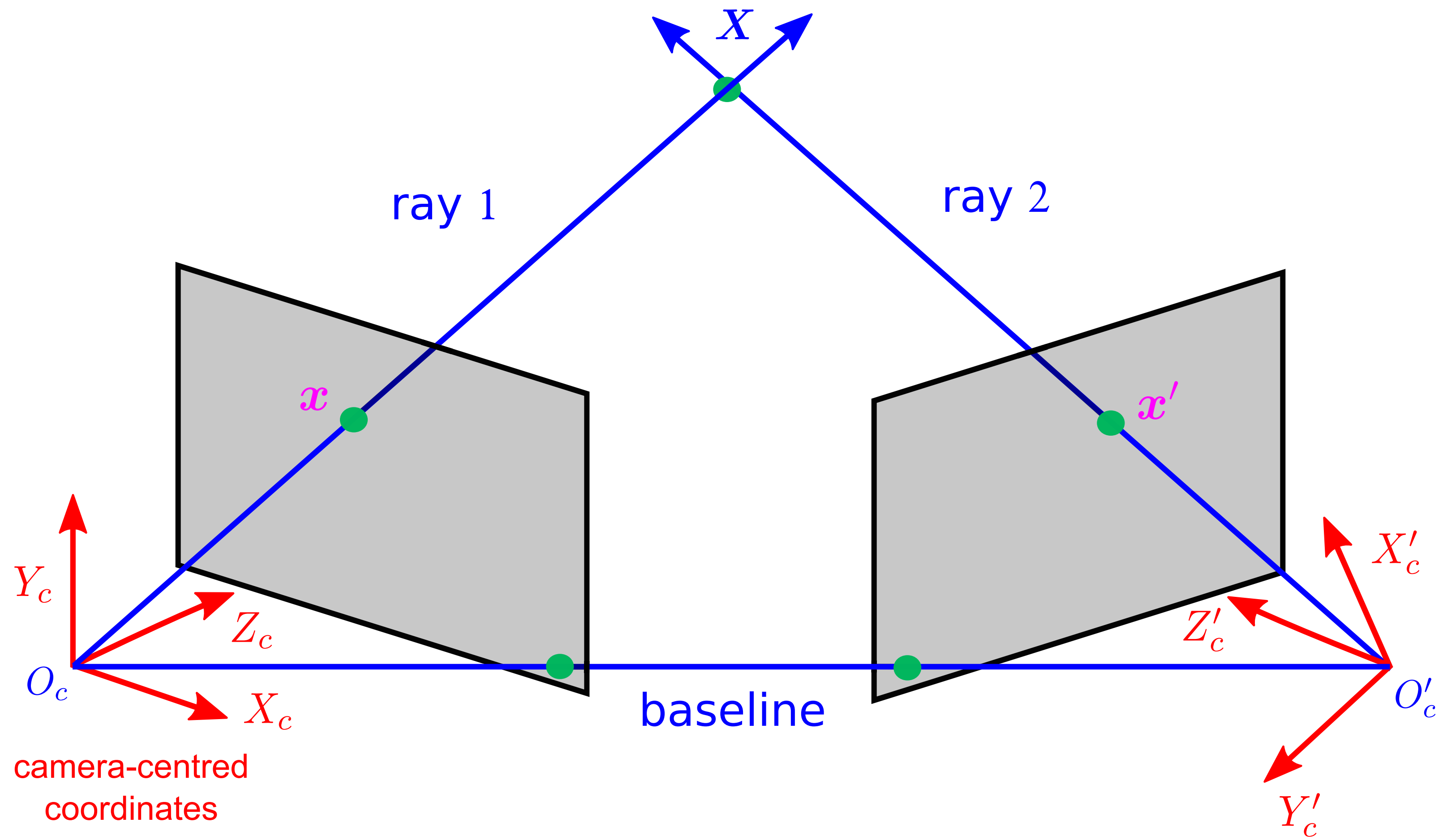
world point
$\boldsymbol{p} = [x, y, f]$
X
Y
Z
$\boldsymbol{x}$
$\boldsymbol{x}'$
image plane coordinates
$\boldsymbol{p}$
$[0, 0, f]$
$Y_c$
$Z_c$
$O_c$
$X_c$
$X'_c$
$Z'_c$
$O'_c$
$Y'_c$
camera-centred coordinates
translate by $T$ rotate by $R$

# Rays



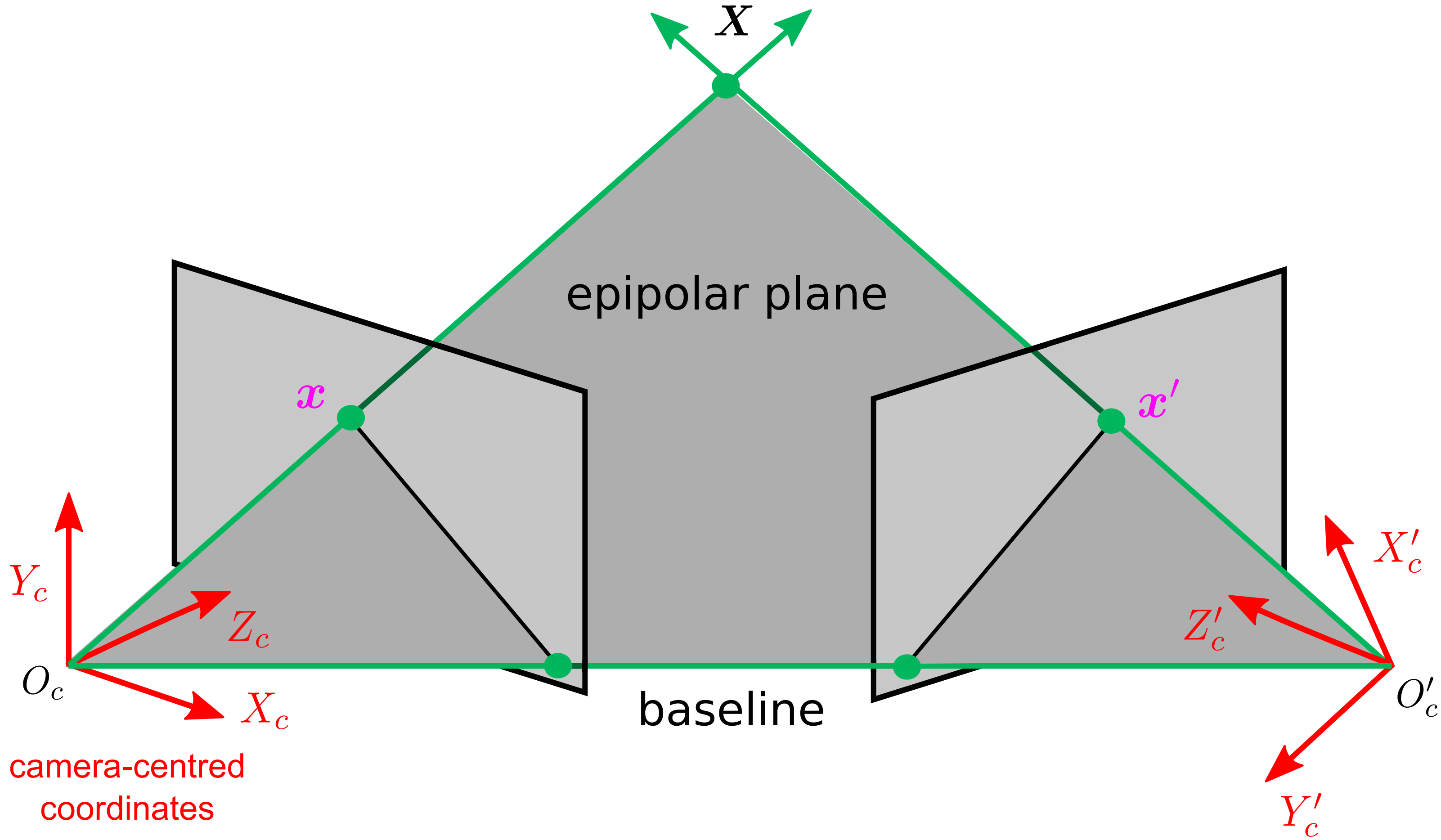
world point
X
Y
Z
$\boldsymbol{p} = [x, y, f]$
$\boldsymbol{p}' = [x', y', f']$
$\boldsymbol{x}$
image plane coordinates
$\boldsymbol{p}$
$[0, 0, f]$
$\boldsymbol{x}'$
$\boldsymbol{p}'$
$Y_c$
$Z_c$
$O_c$
$X_c$
$X'_c$
$Z'_c$
$O'_c$
$Y'_c$
camera-centred coordinates
translate by $T$ rotate by $R$

# Rays

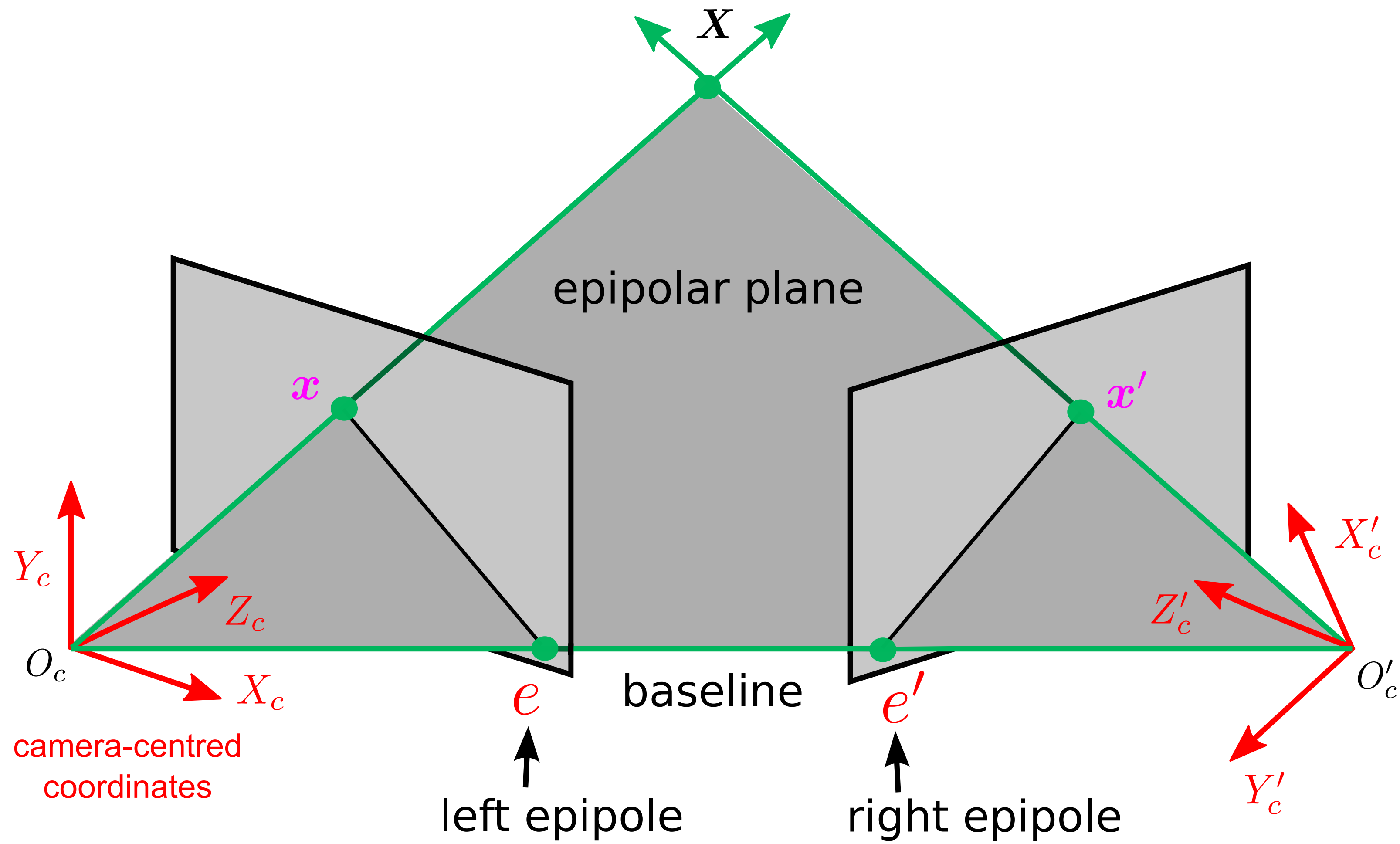


$\boldsymbol{X}$
ray 1
ray 2
$\boldsymbol{x}$
$\boldsymbol{x}'$
$Y_c$
$Z_c$
$X_c$
$O_c$
baseline
$O'_c$
$X'_c$
$Z'_c$
$Y'_c$
camera-centred coordinates

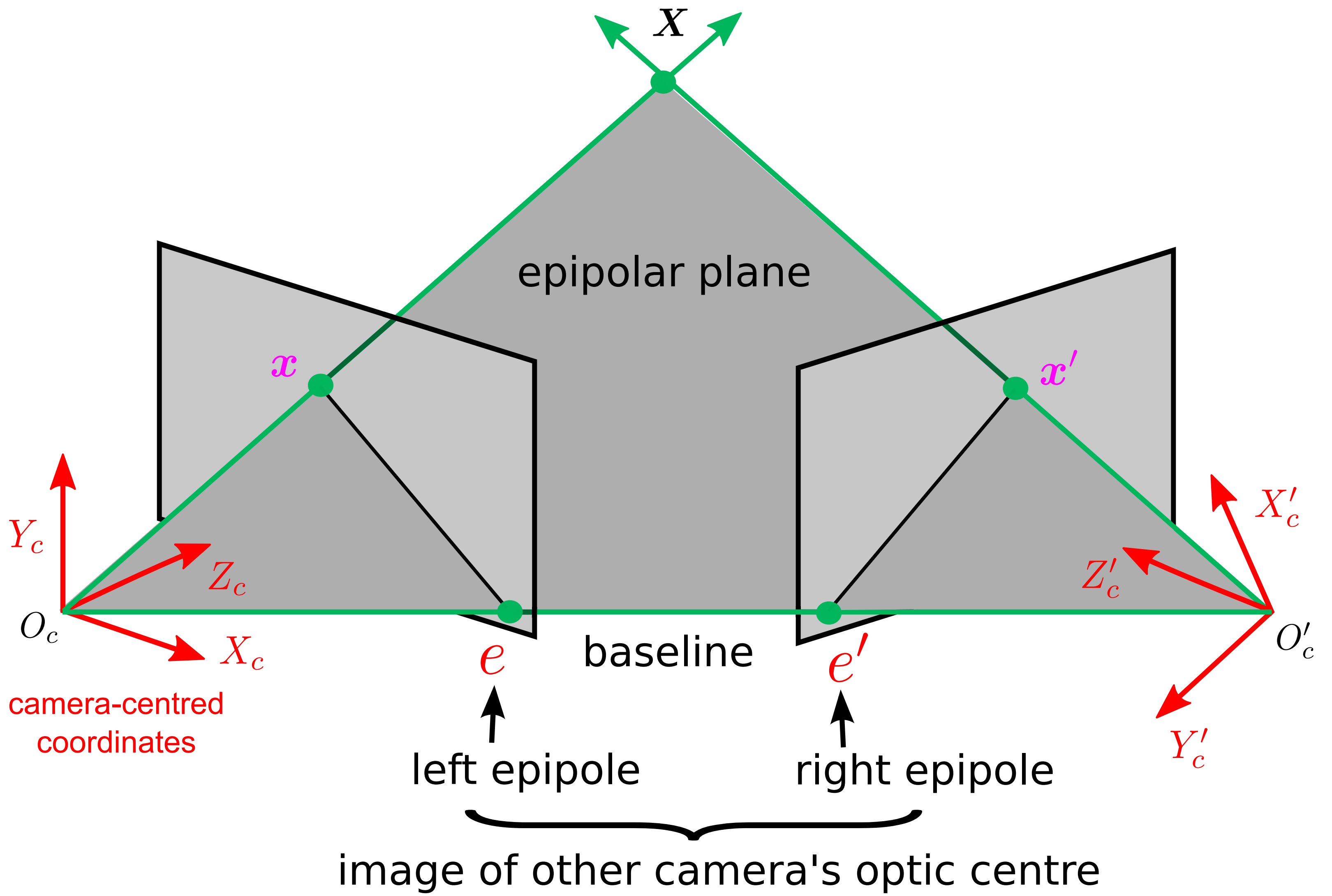
# Epipolar geometry



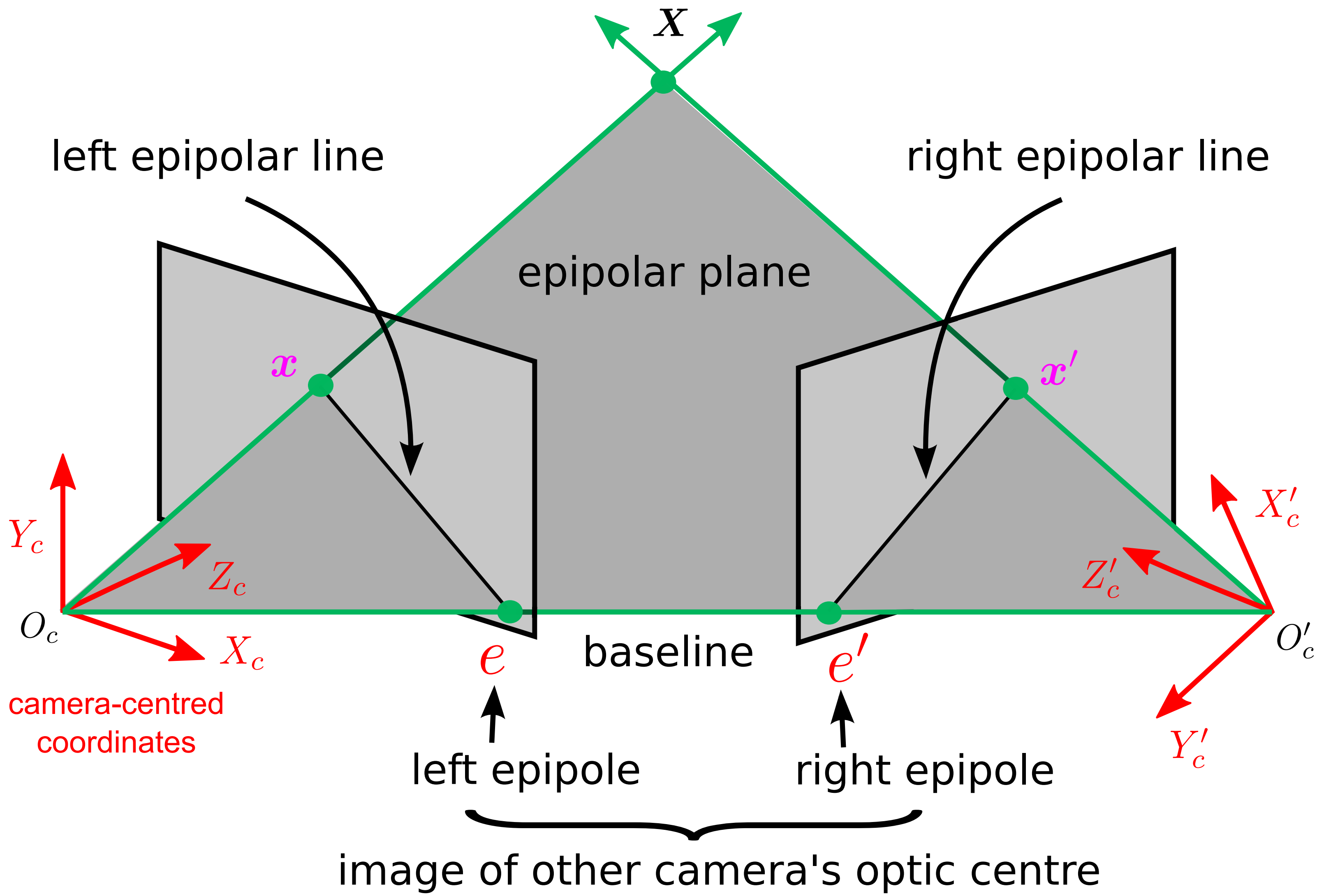
# Epipolar geometry



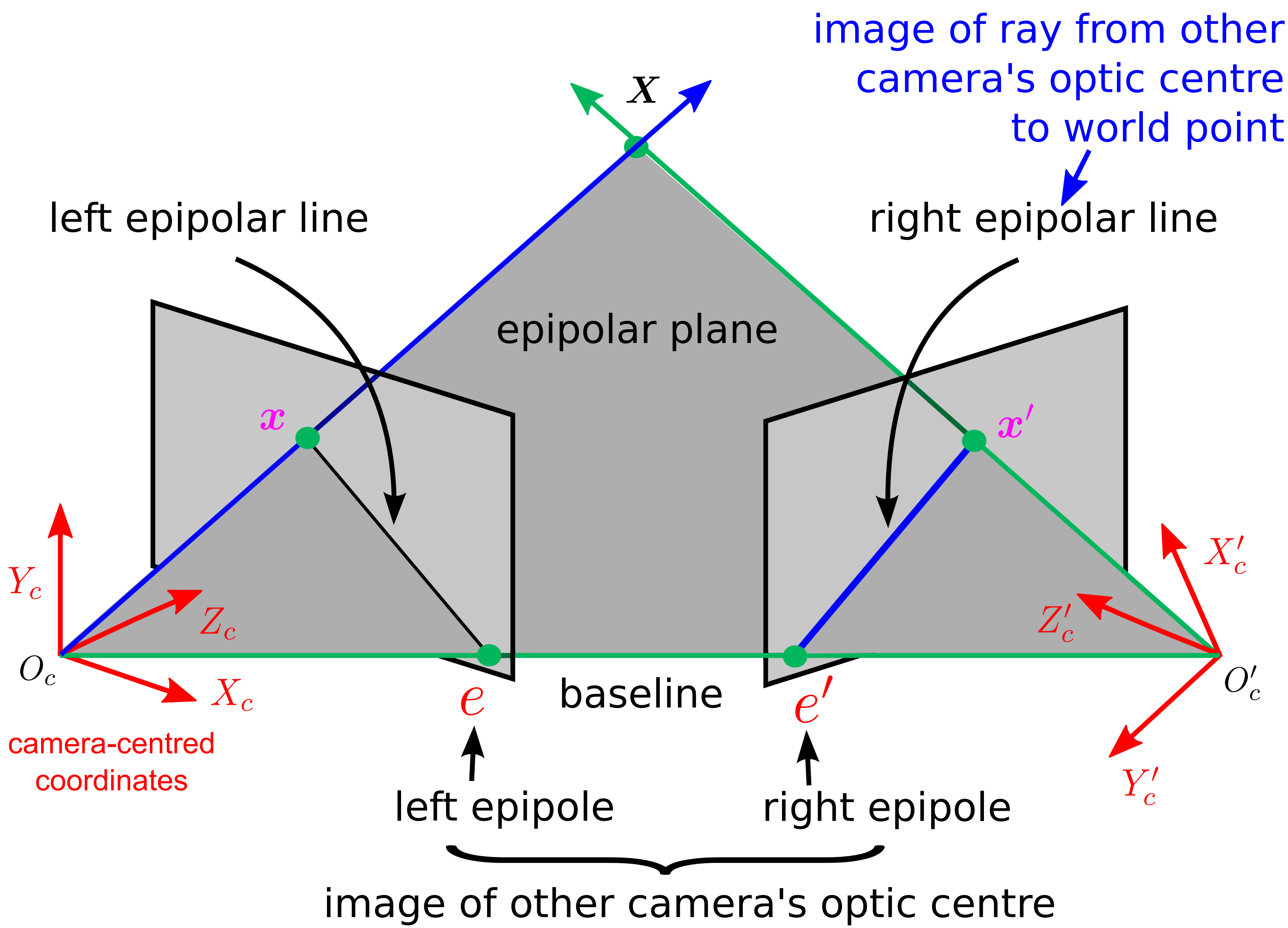
# Epipolar geometry



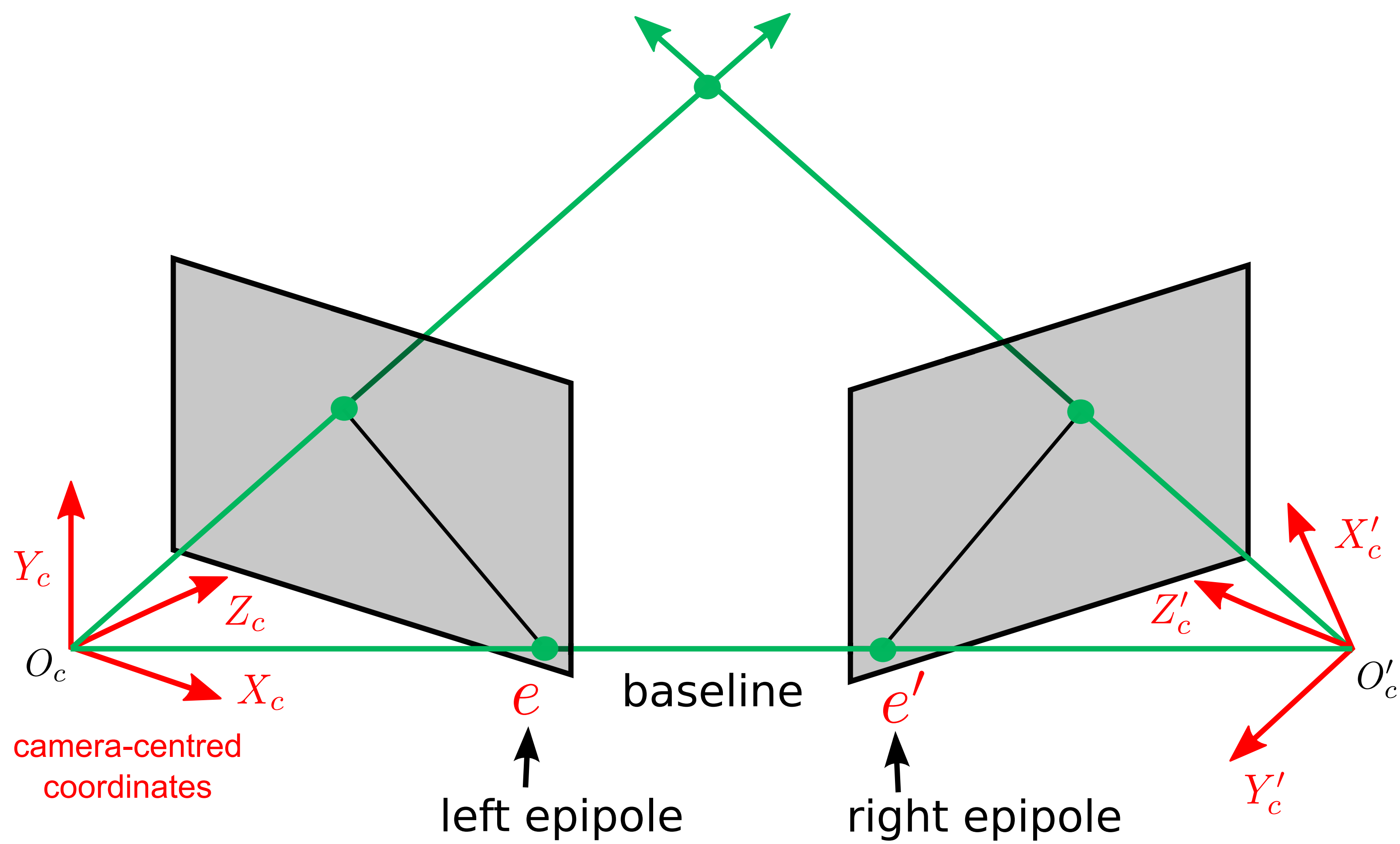
# Epipolar geometry



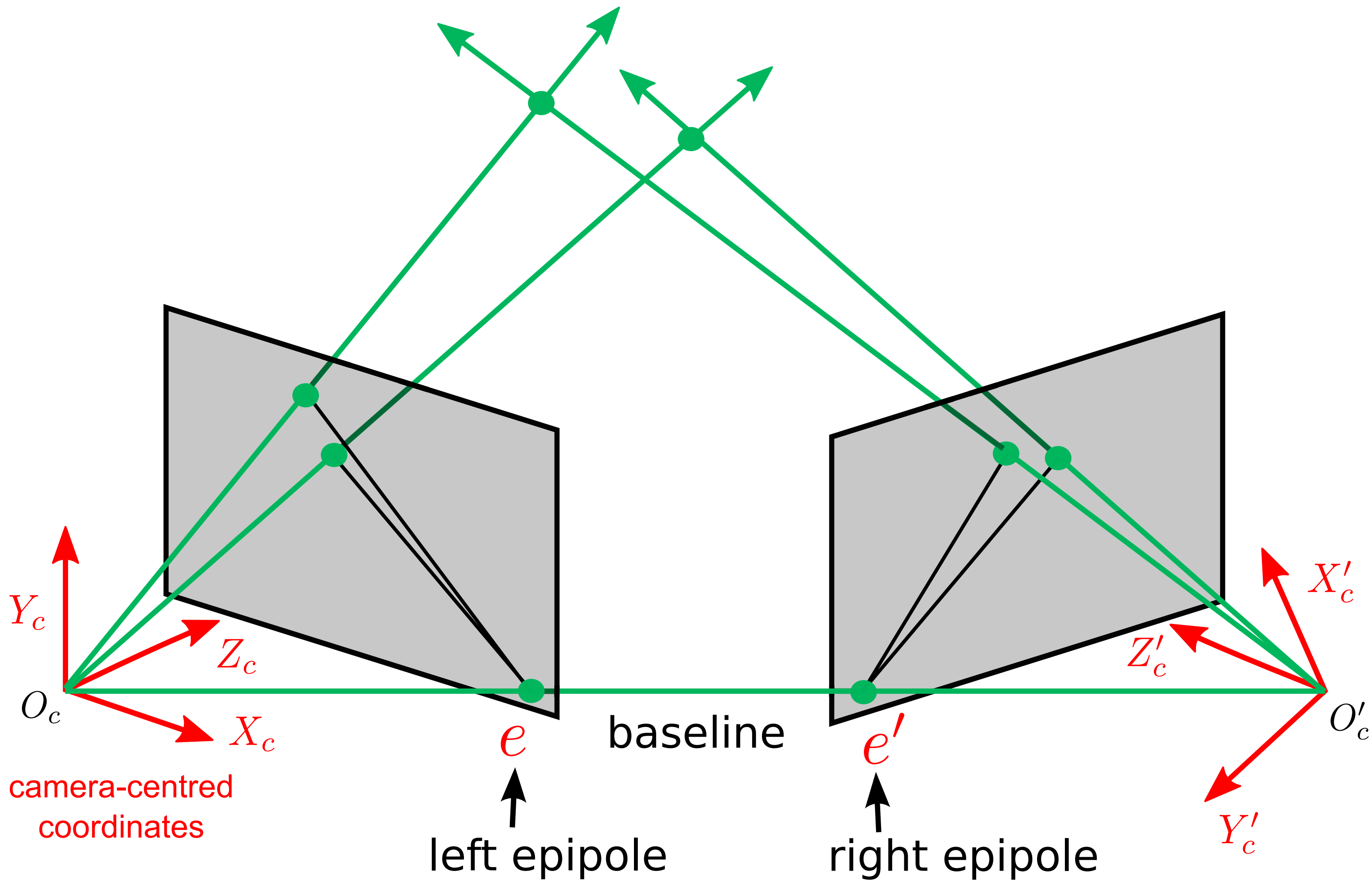
# Epipolar geometry



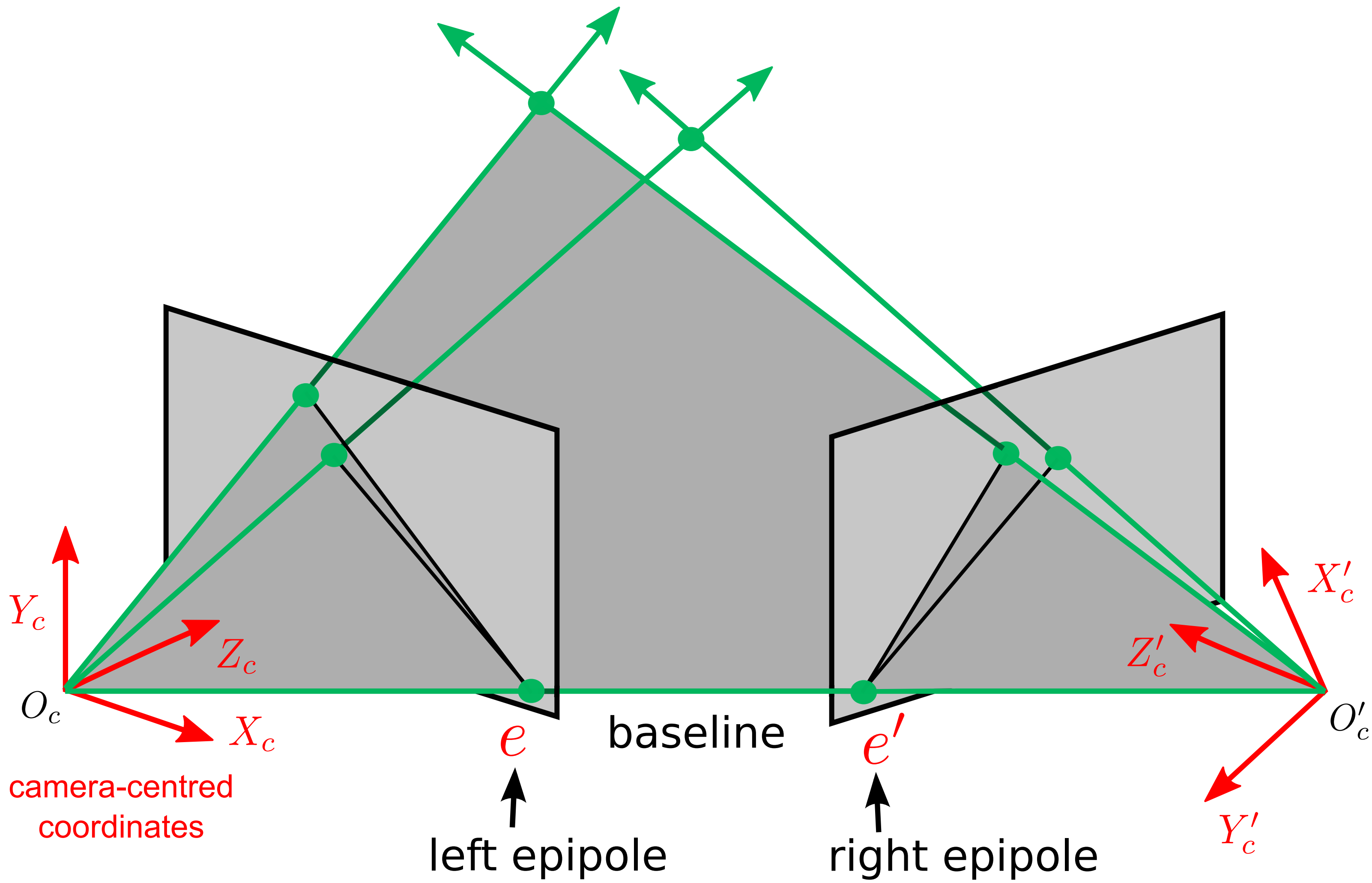
# Epipolar geometry



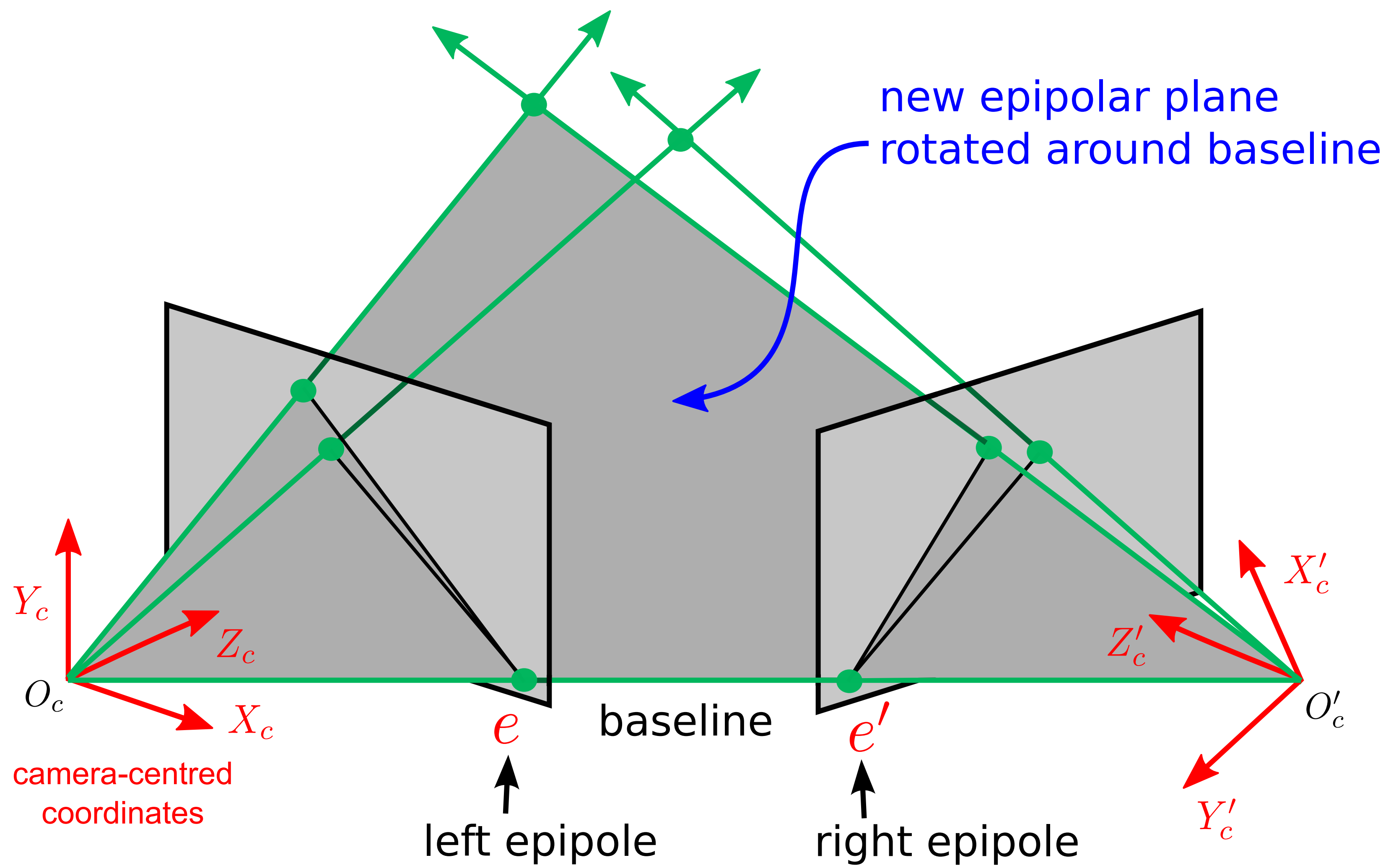
# Epipolar geometry



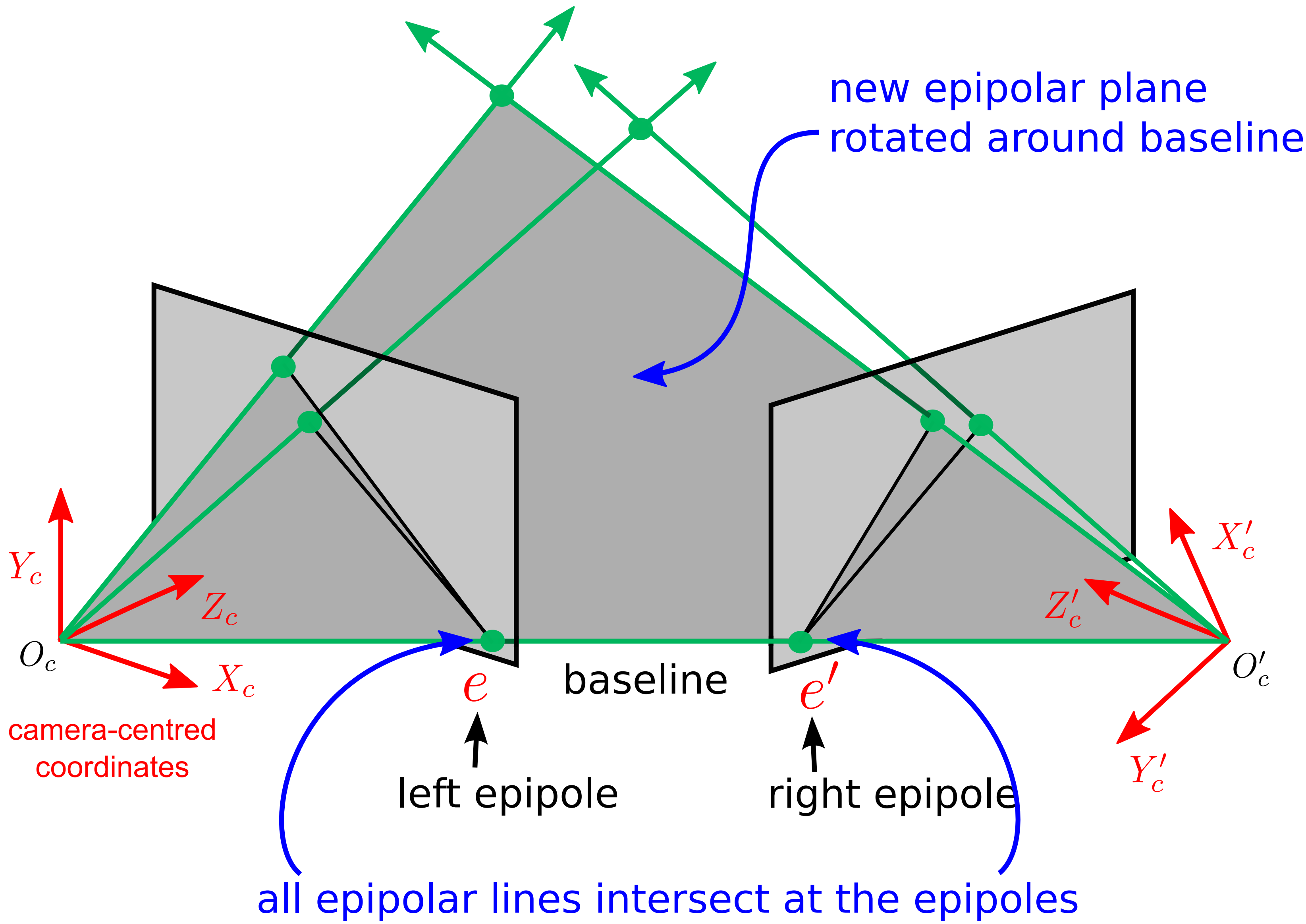
# Epipolar geometry



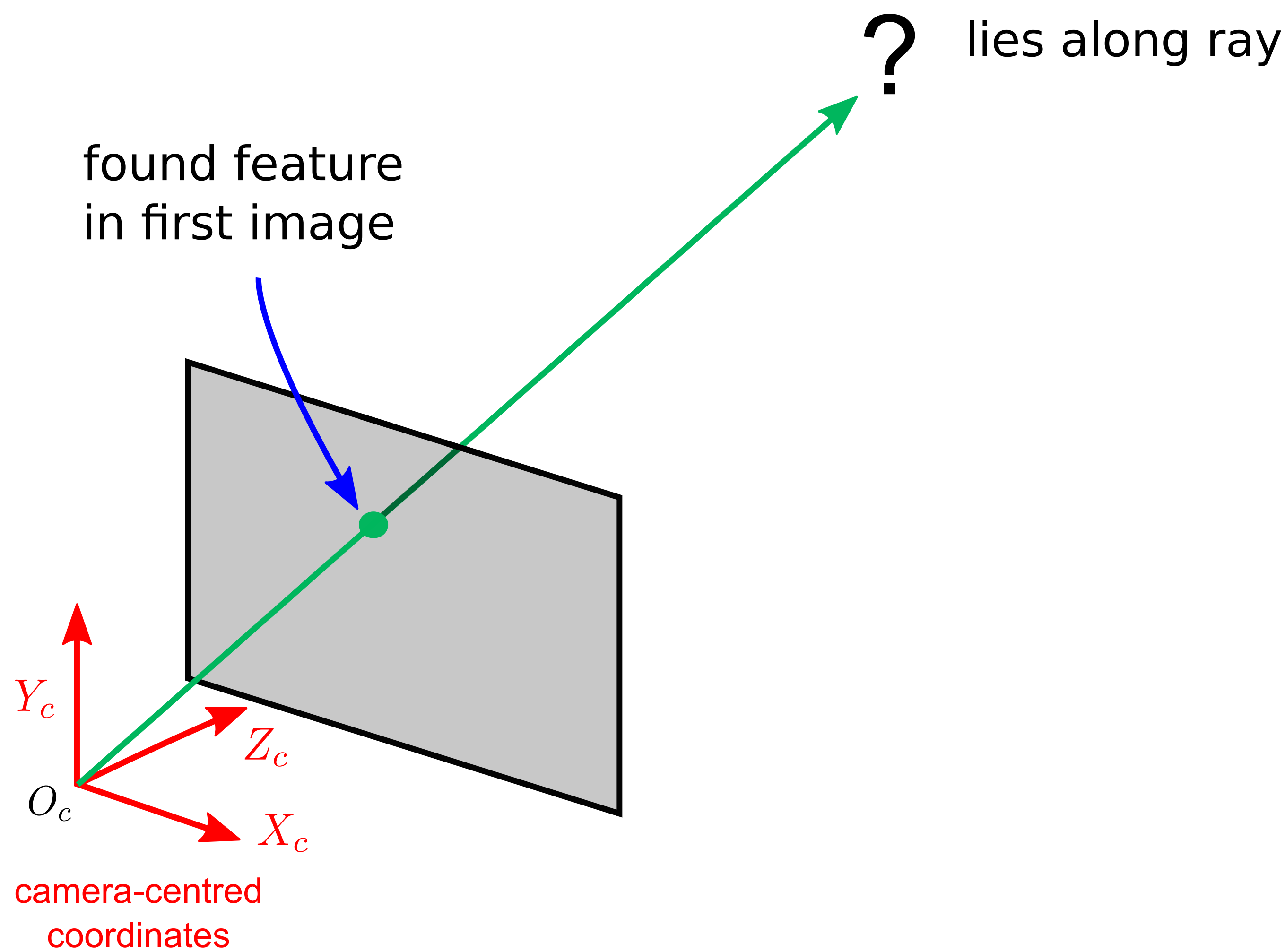
# Epipolar geometry



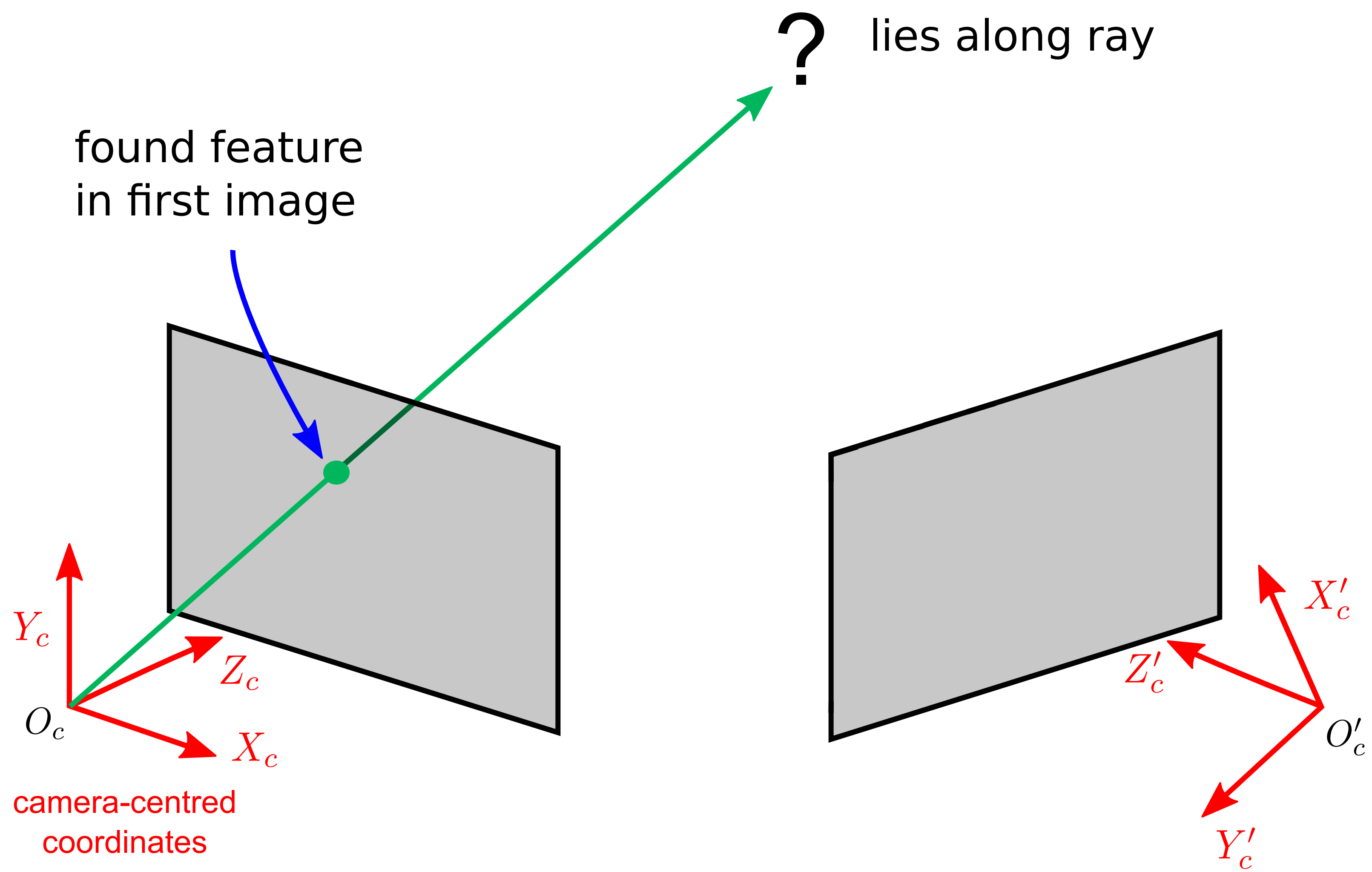
# Epipolar geometry



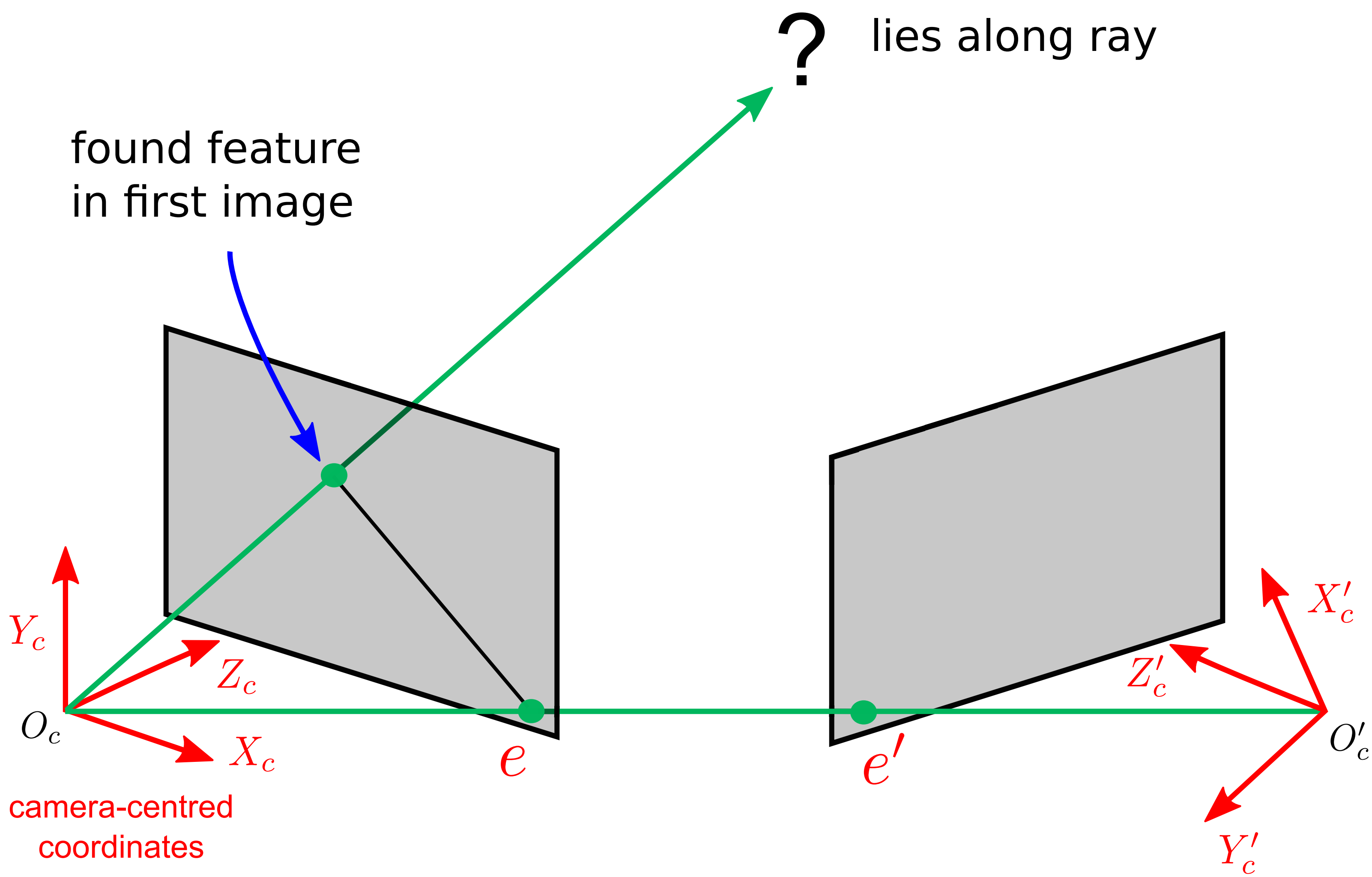
# Epipolar geometry



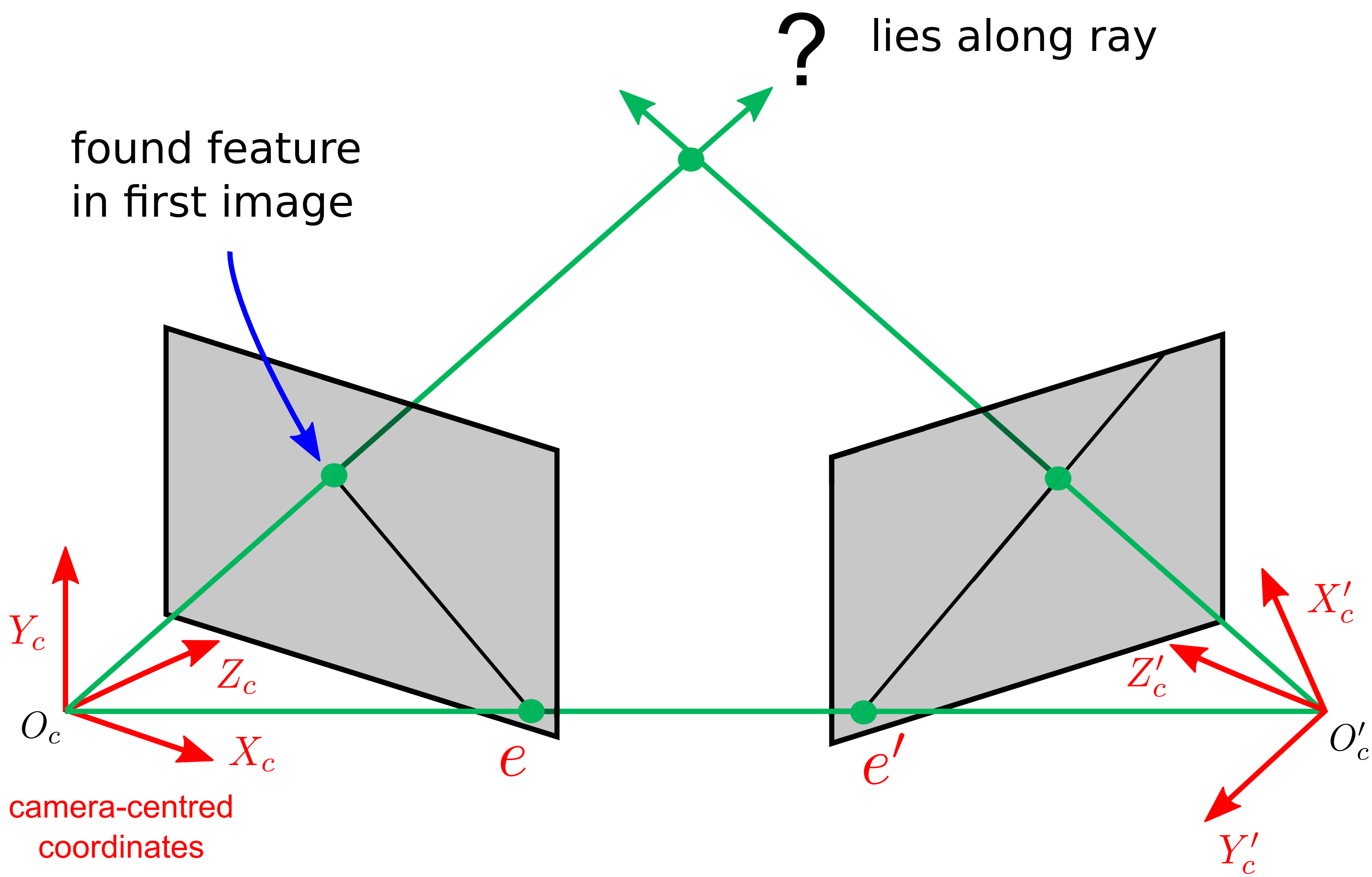
# Epipolar geometry



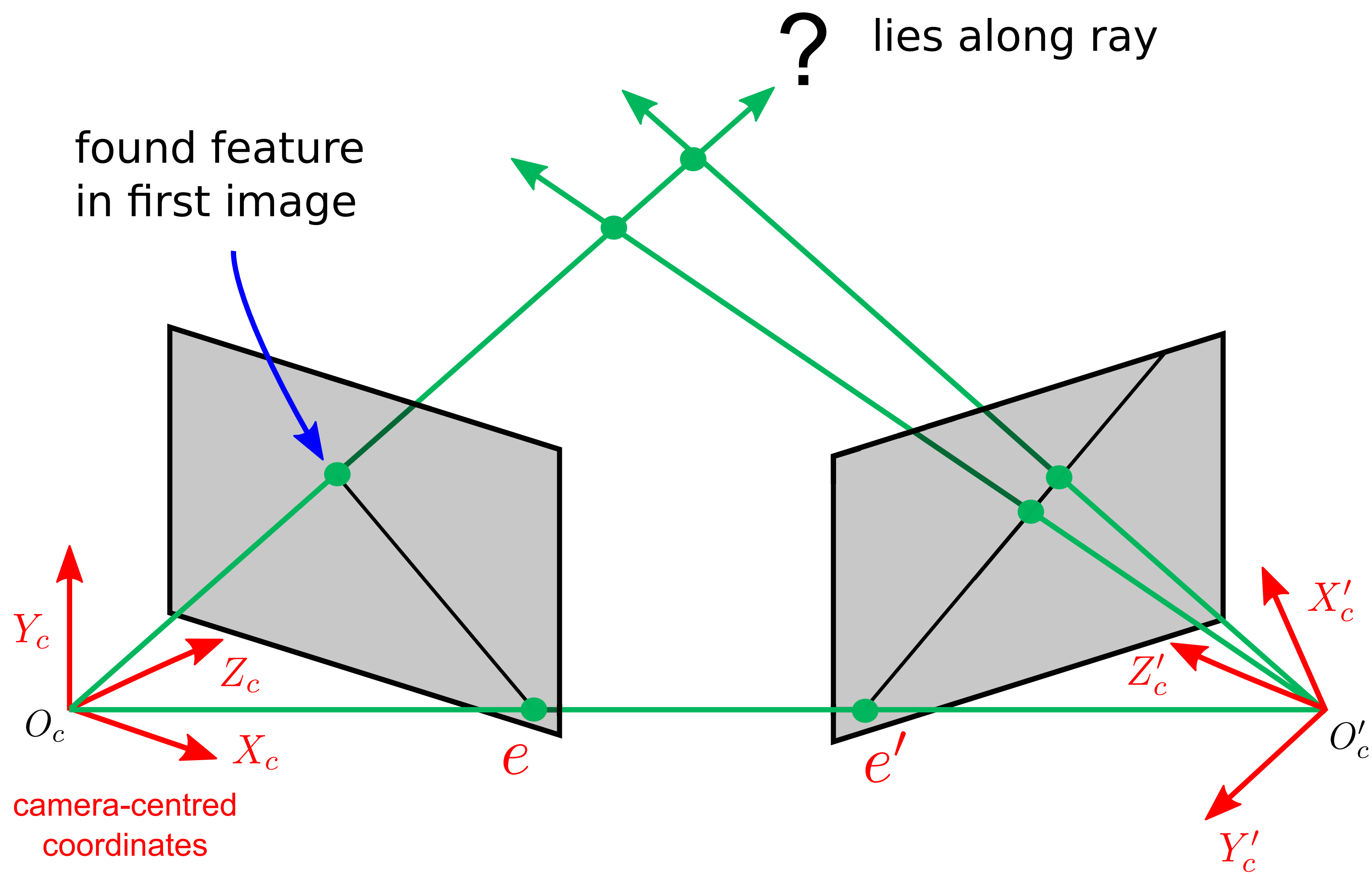
# Epipolar geometry



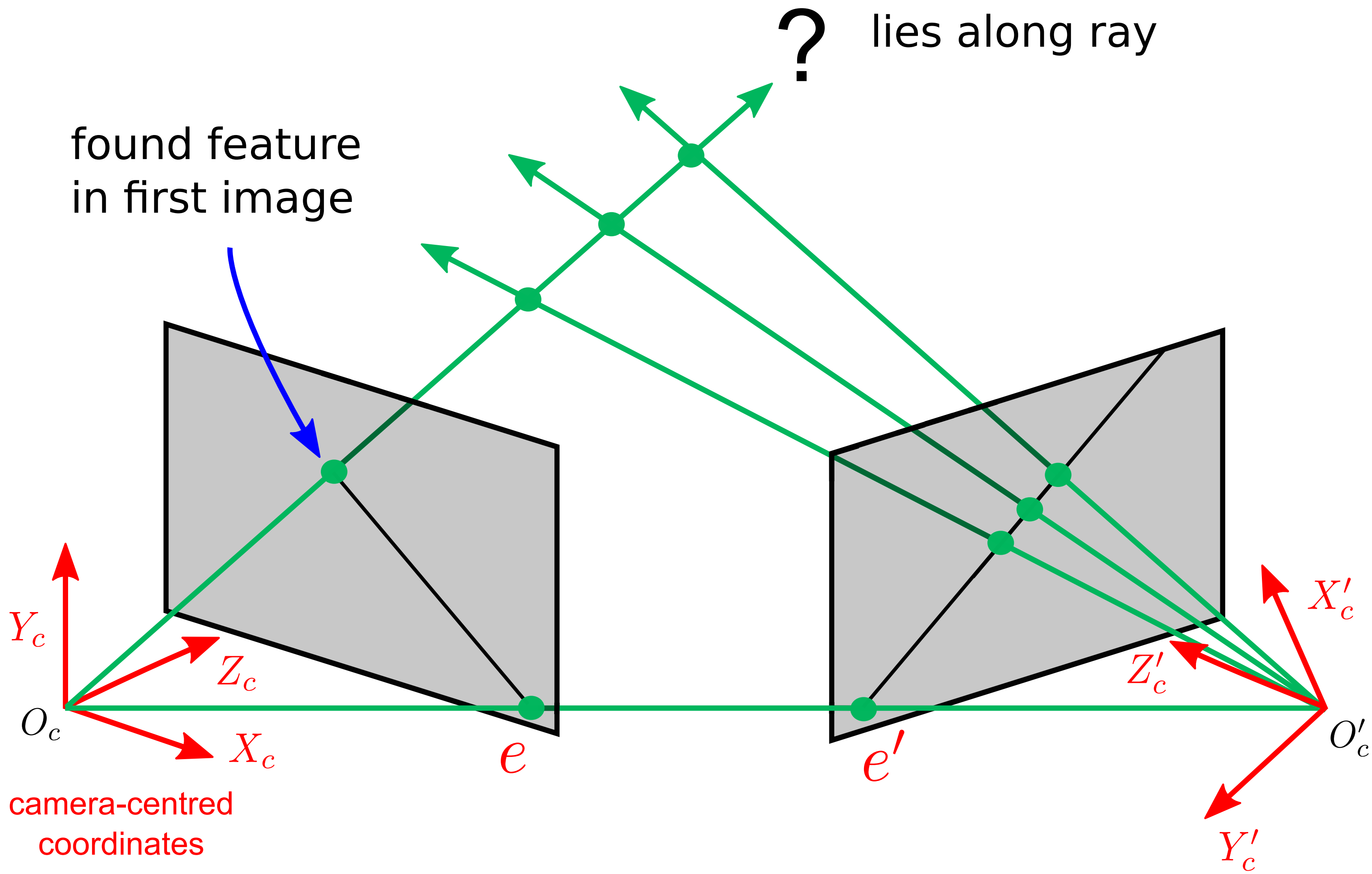
# Epipolar geometry



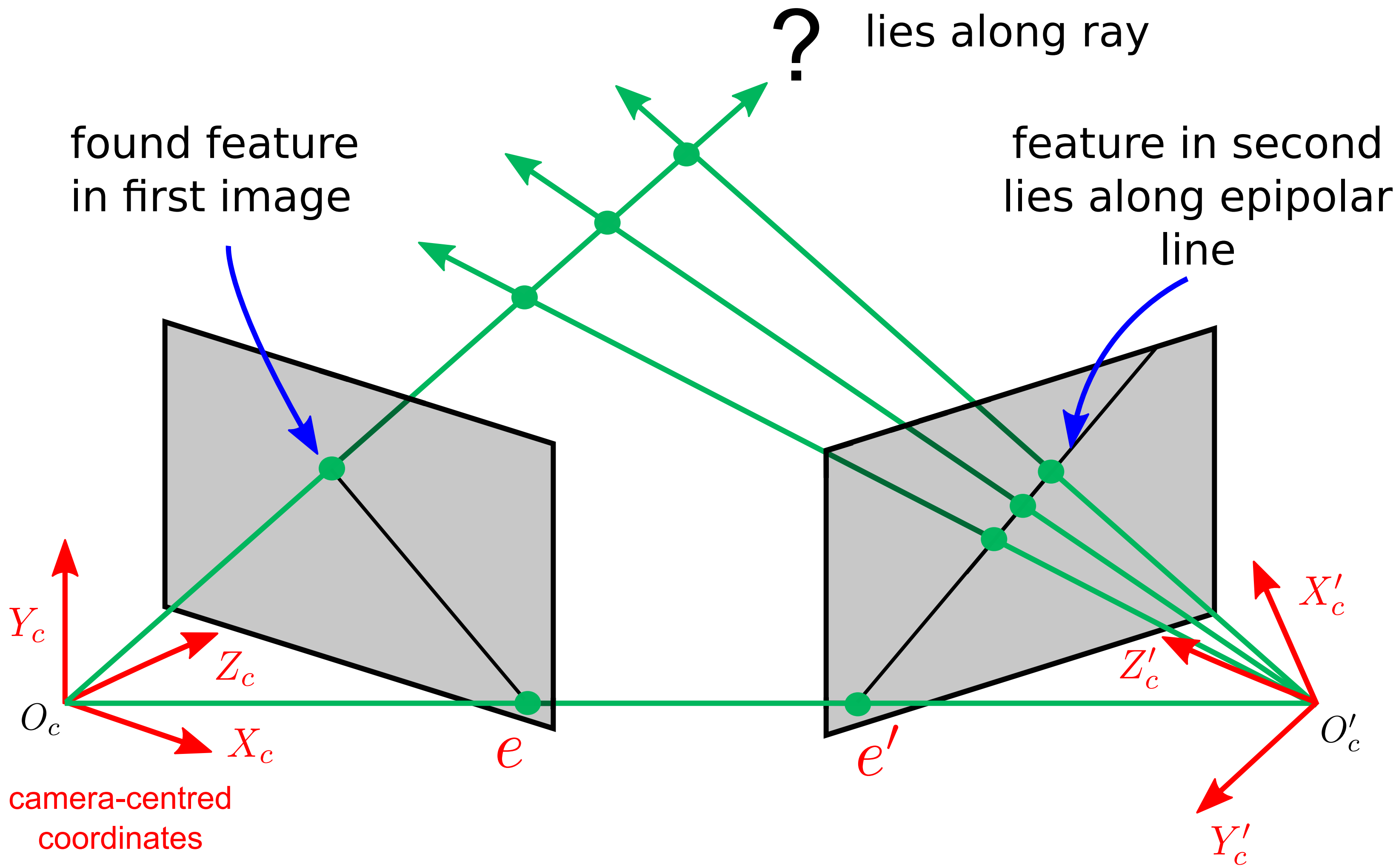
# Epipolar geometry



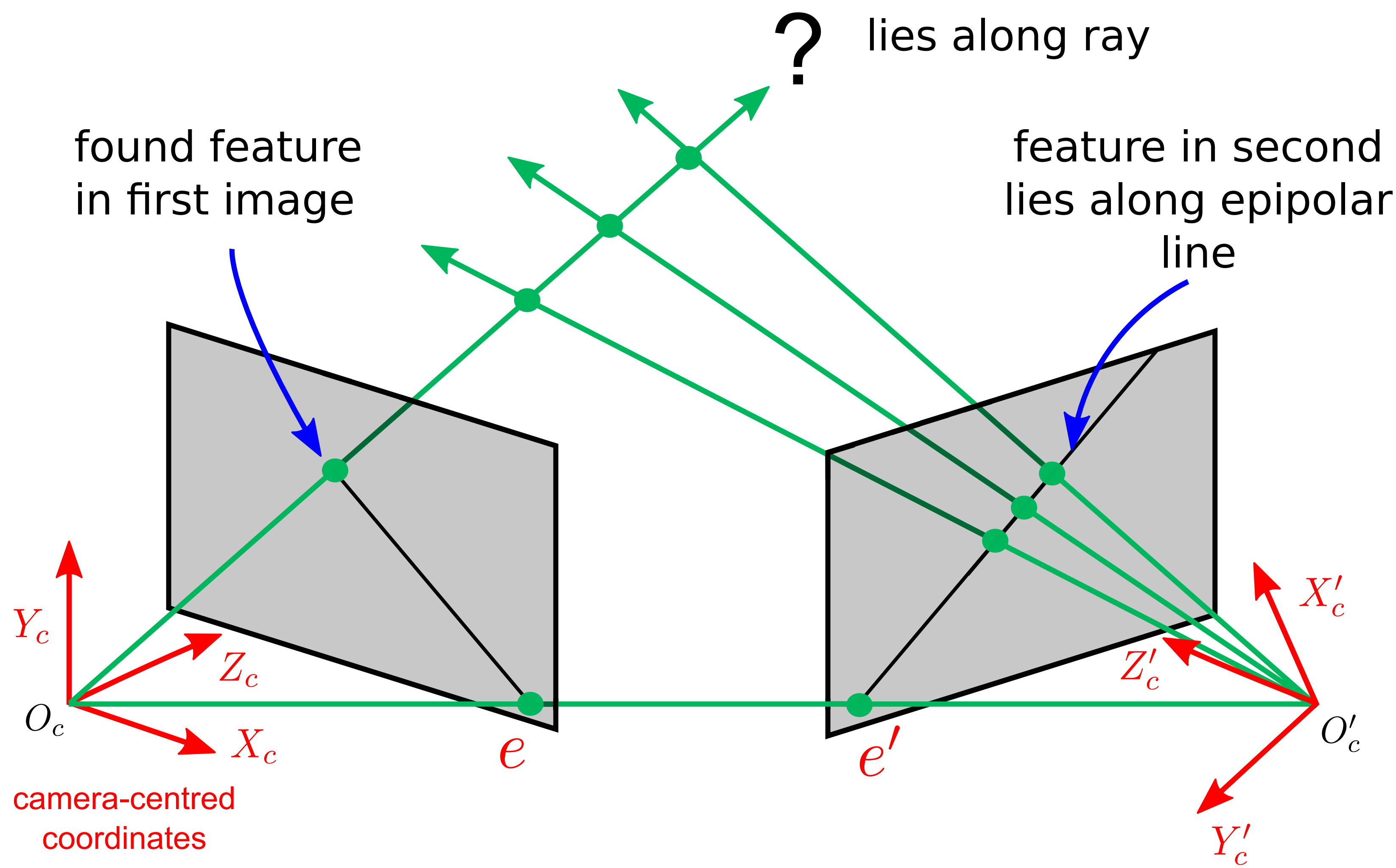
# Epipolar geometry



# Epipolar geometry

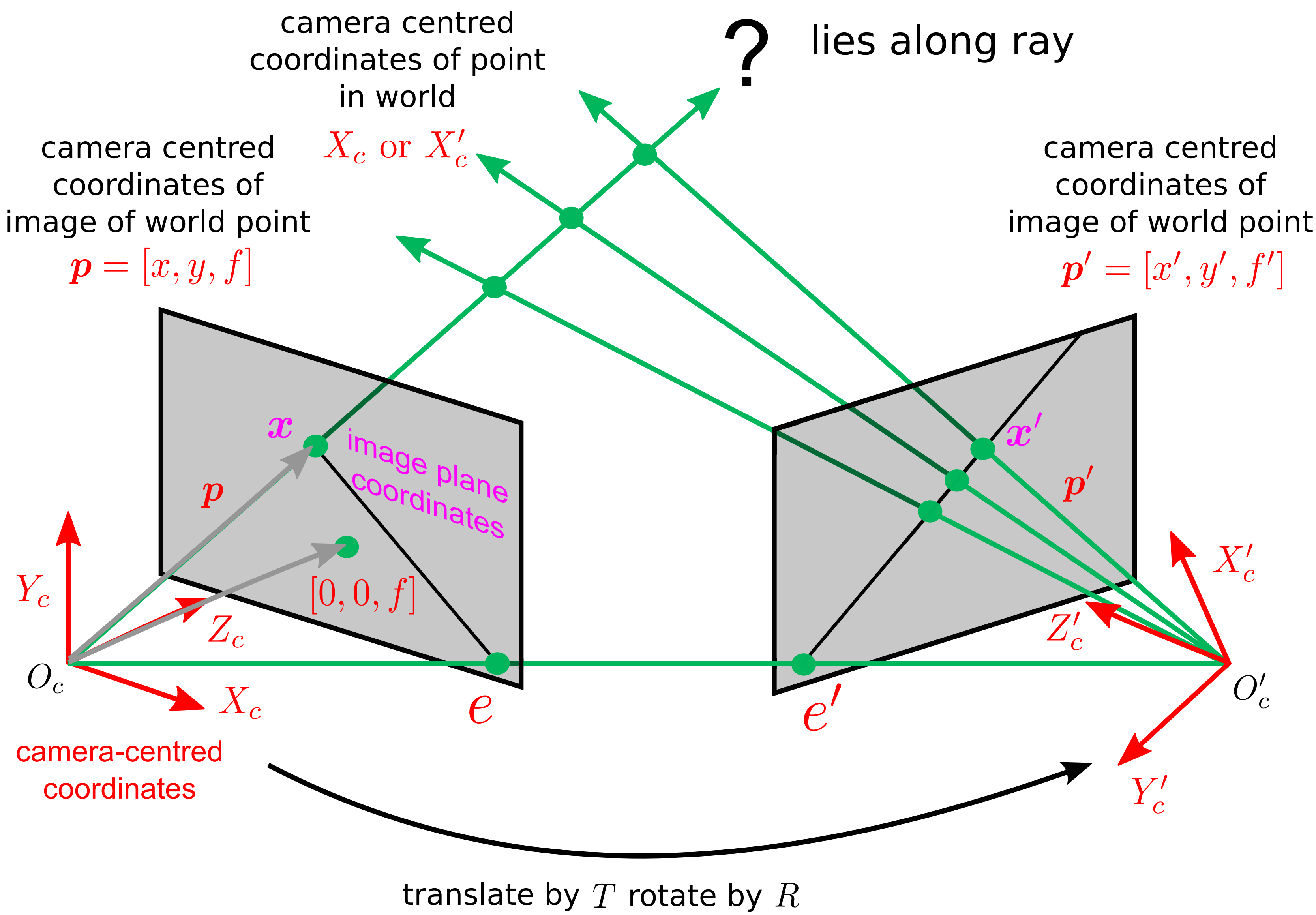


# Epipolar geometry

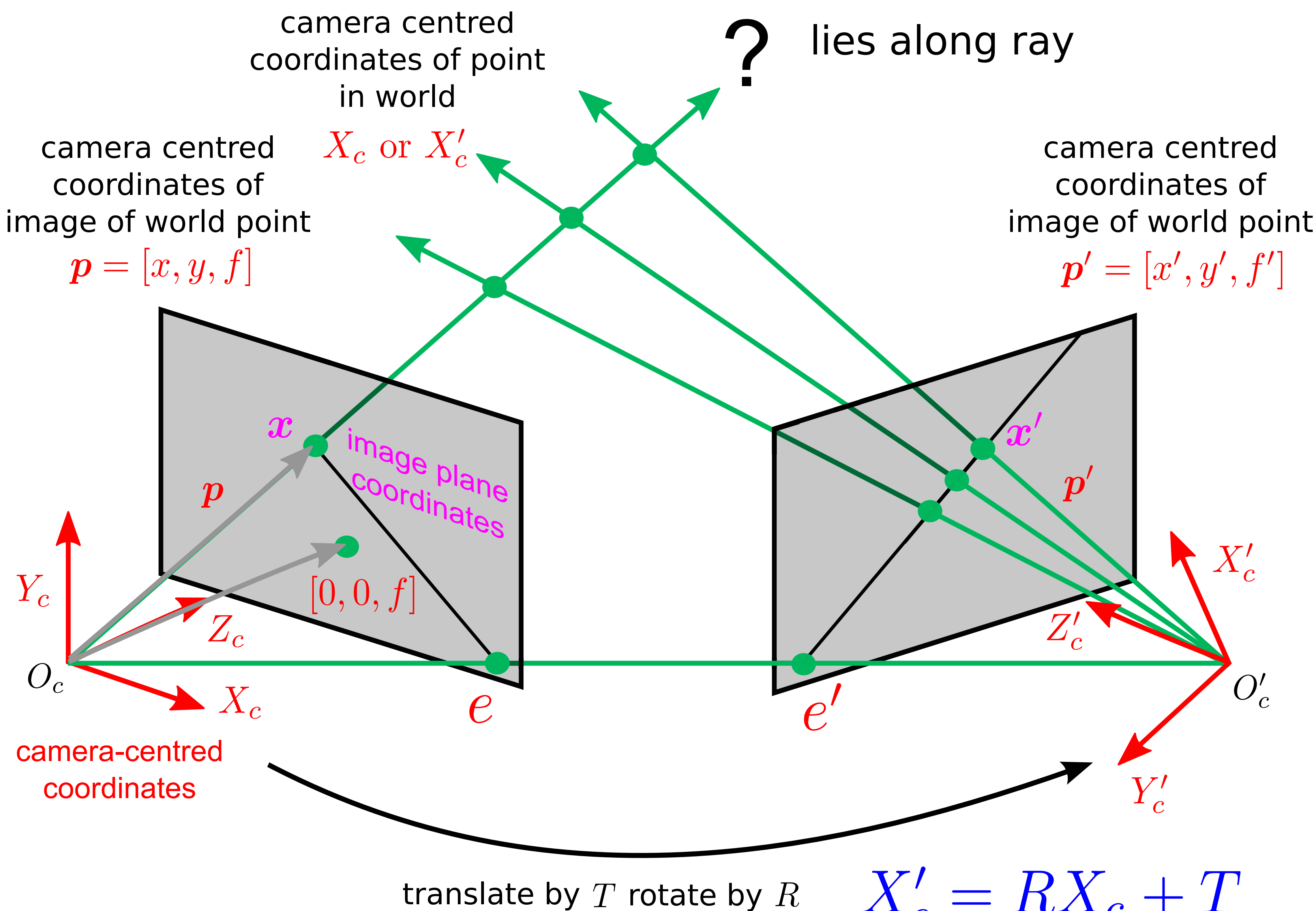



feature matching = one D search

# What's the form of the epipolar line?

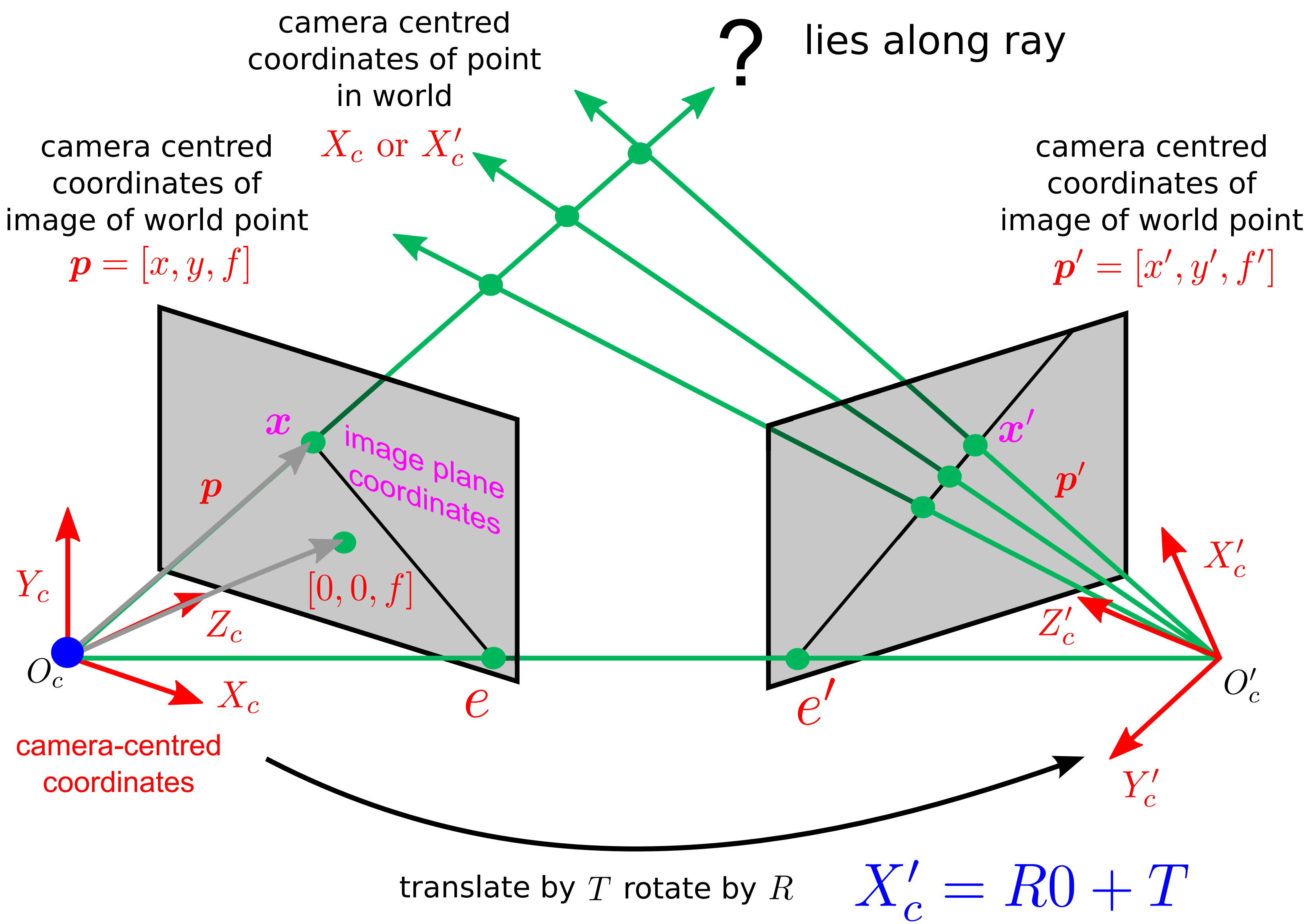


# Epipolar geometry

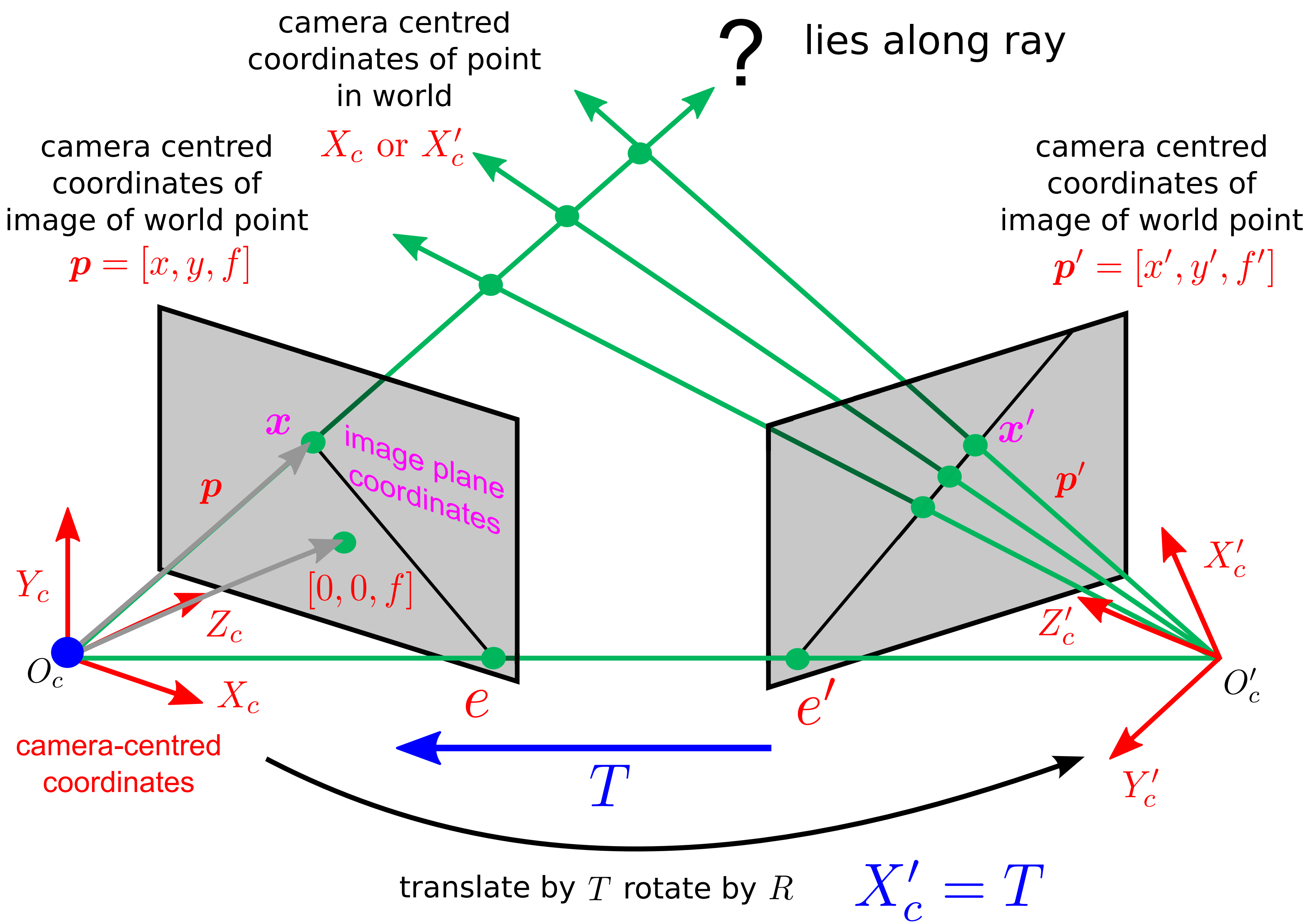



$$X'_c = RX_c + T$$

# Epipolar geometry



# Epipolar geometry

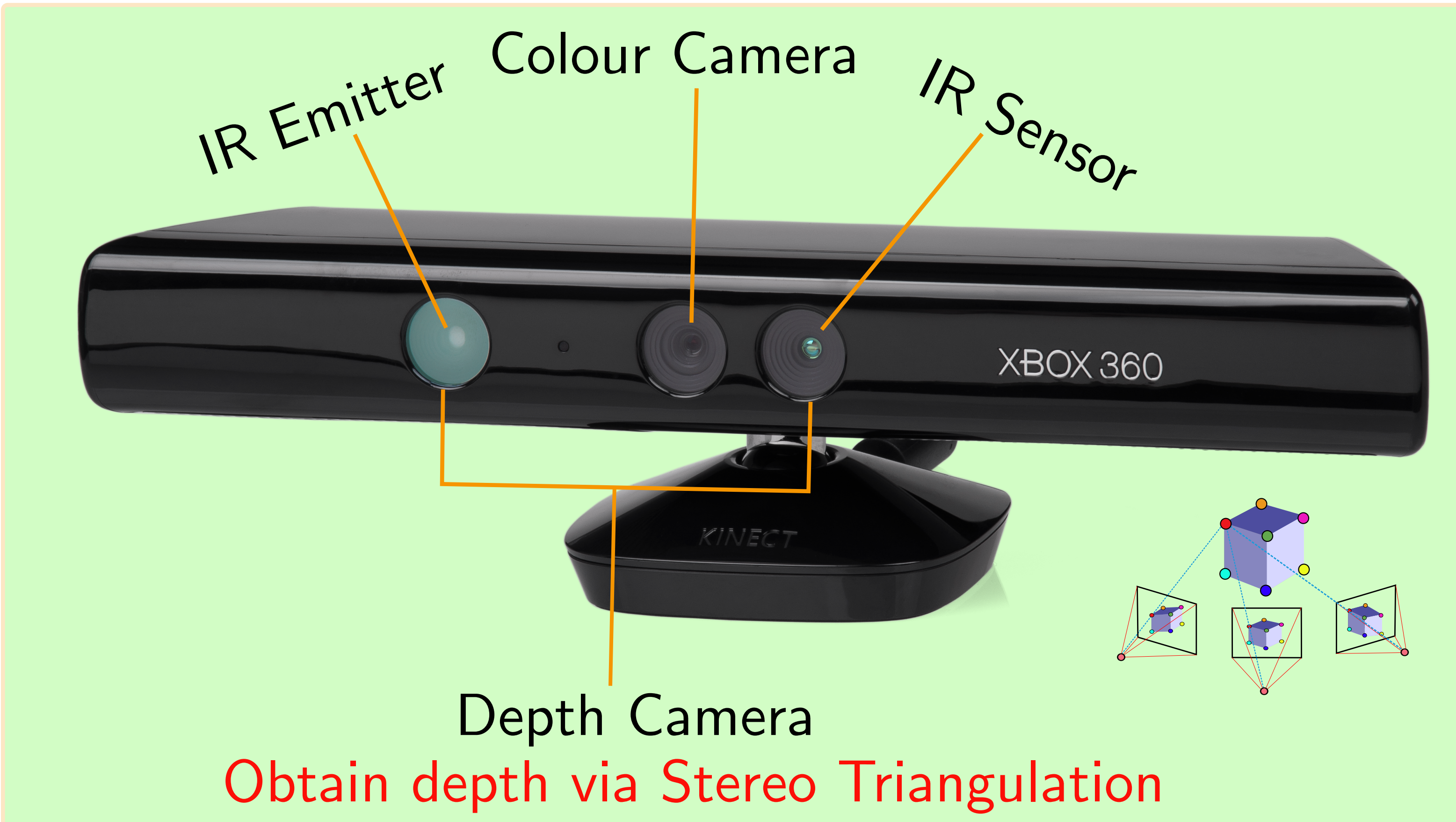


# What's left to do?

- Uncalibrated cameras: to use framework, need to convert from camera centred coordinates to (homogeneous) pixel coordinates
- that's the focus of our next lecture
- for now some demos...

`http://www.3dflow.net/technology/samantha-structure-from-motion/`
`http://www.youtube.com/watch?v=faZSoE1qPxA`
`http://www.acute3d.com/videos/`

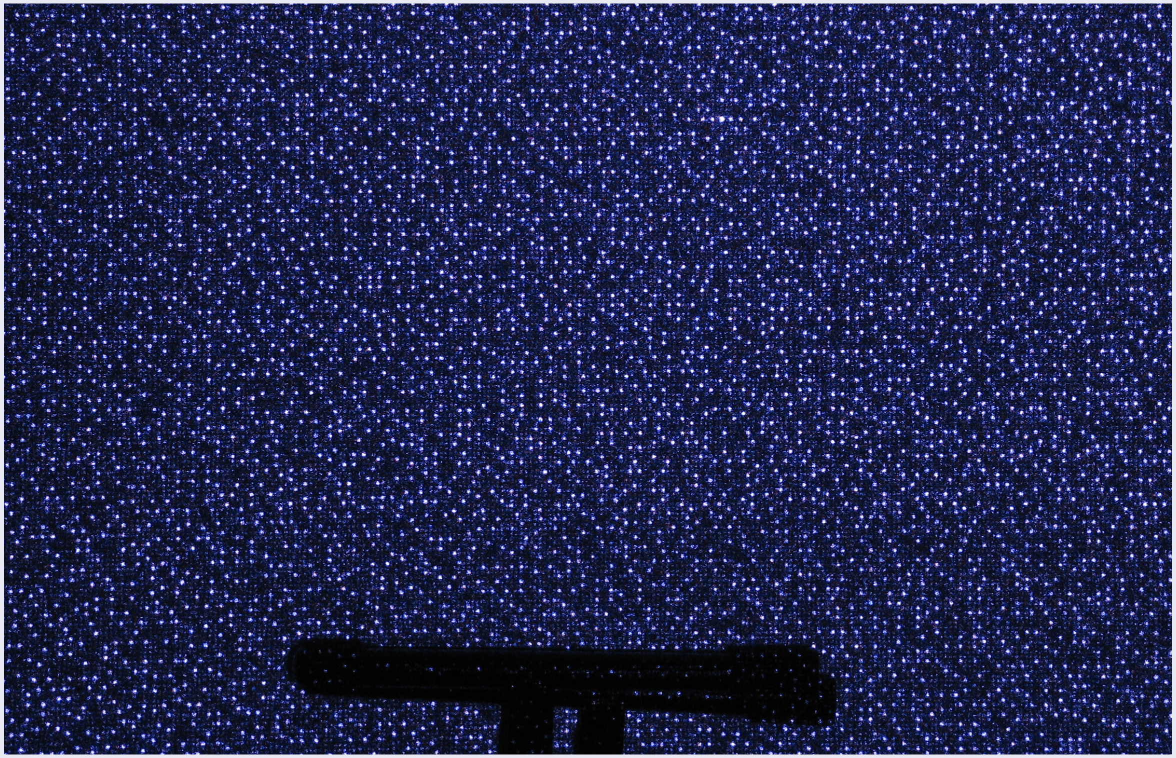
# Kinect




Kinect Sensor: Triangulation is the key to estimating depth.

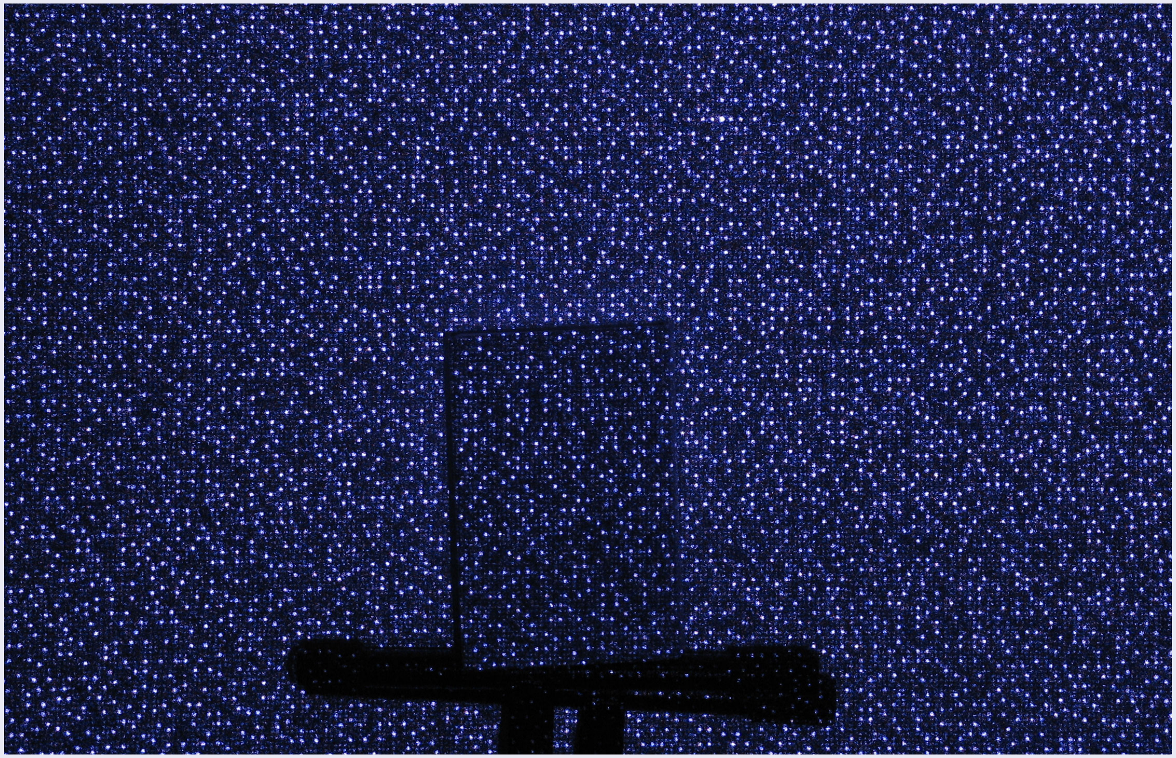
# Kinect

Book vs No Book (image courtesy: http://www.futurepicture.org/?p=97)

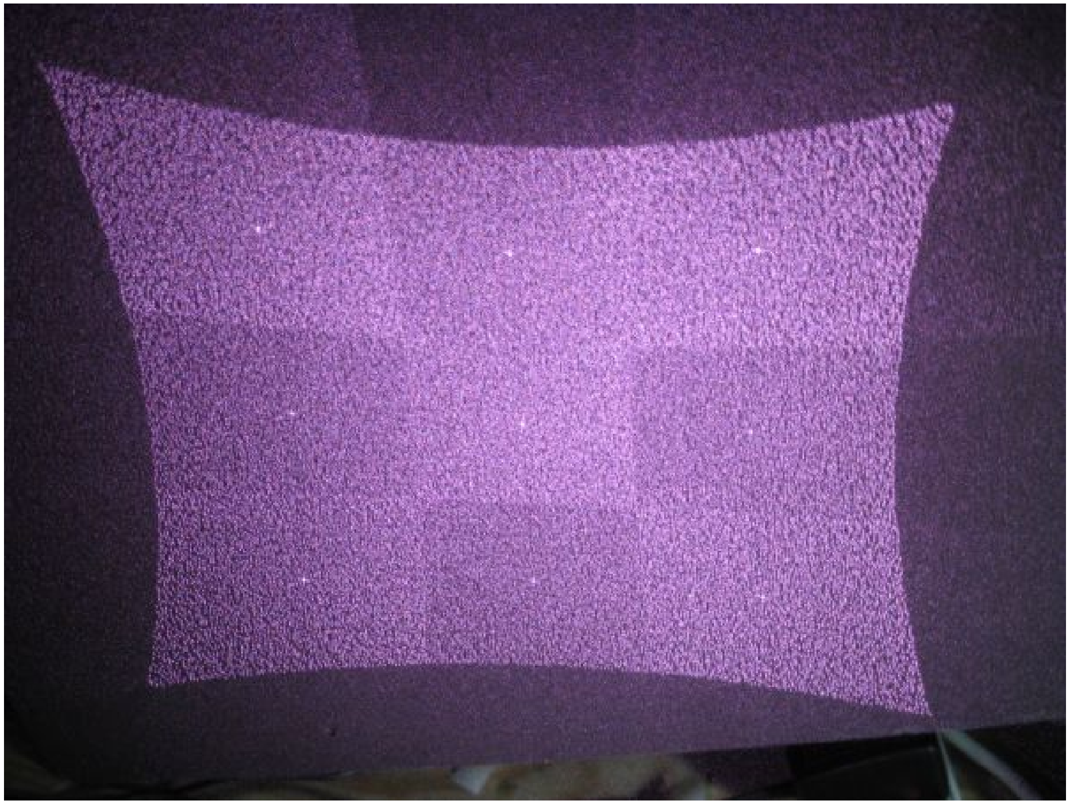


# Kinect

Book vs No Book (image courtesy: `http://www.futurepicture.org/?p=97`)

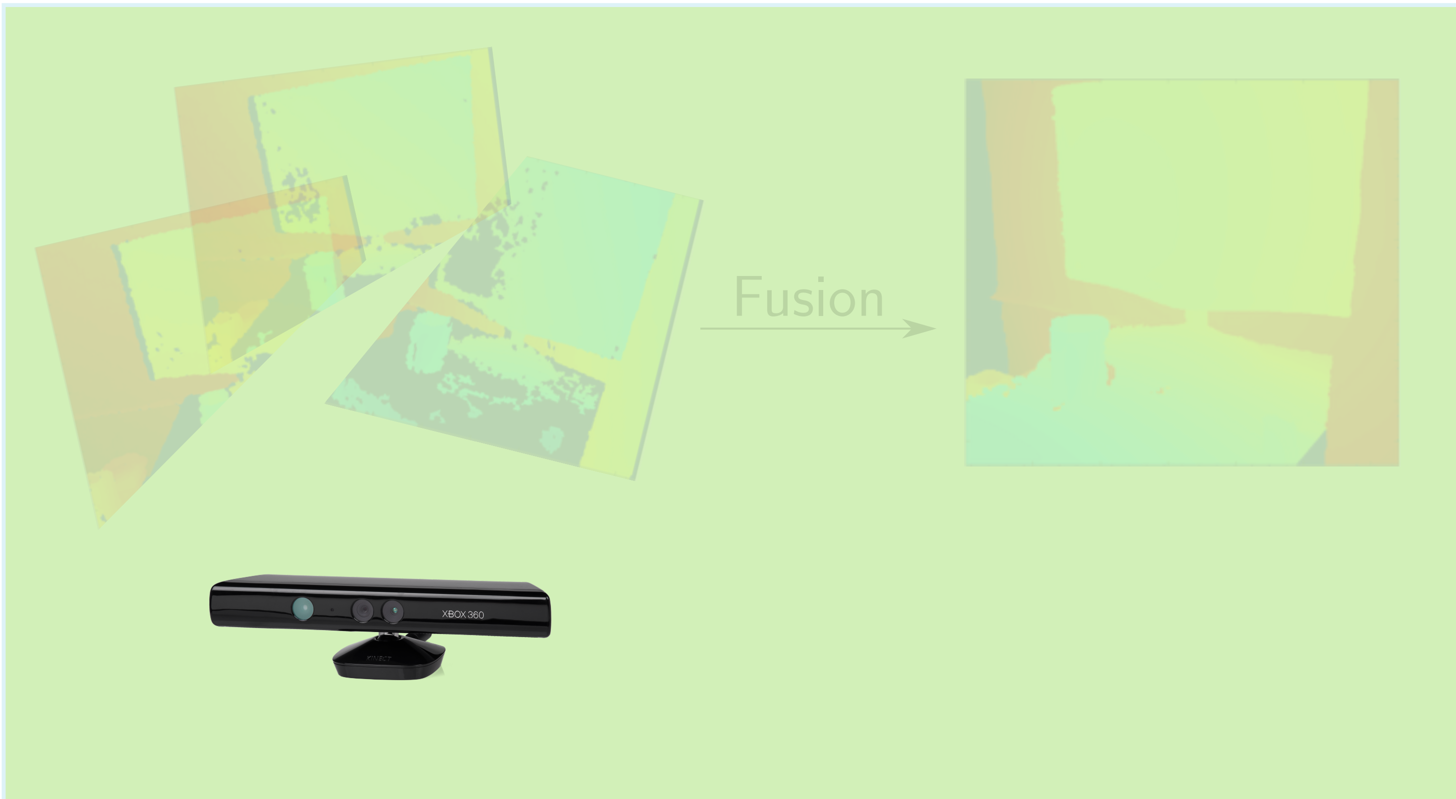


# Kinect



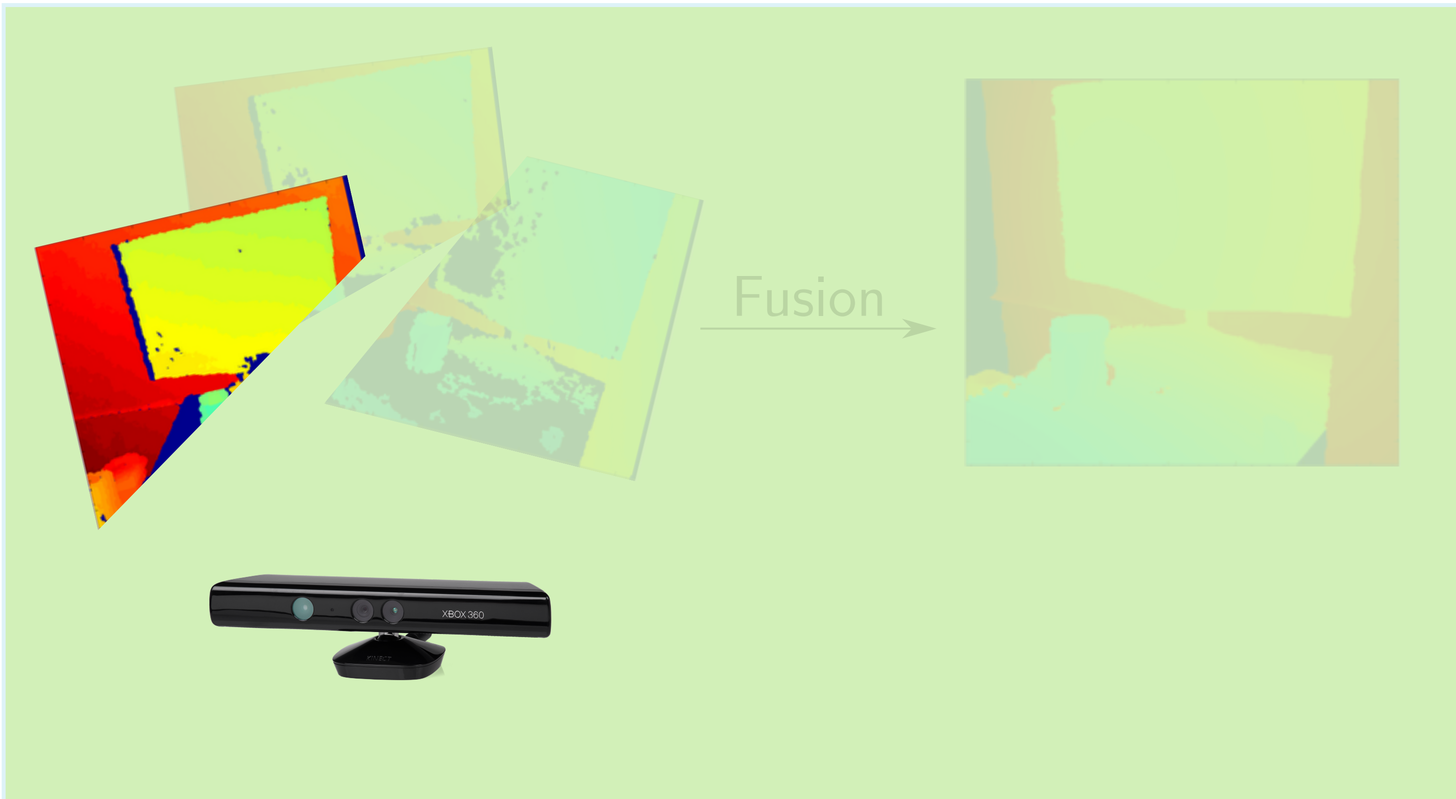
Kinect modifies the pattern a bit and projects a dot-ted tic-tac-toe pattern

# Kinect



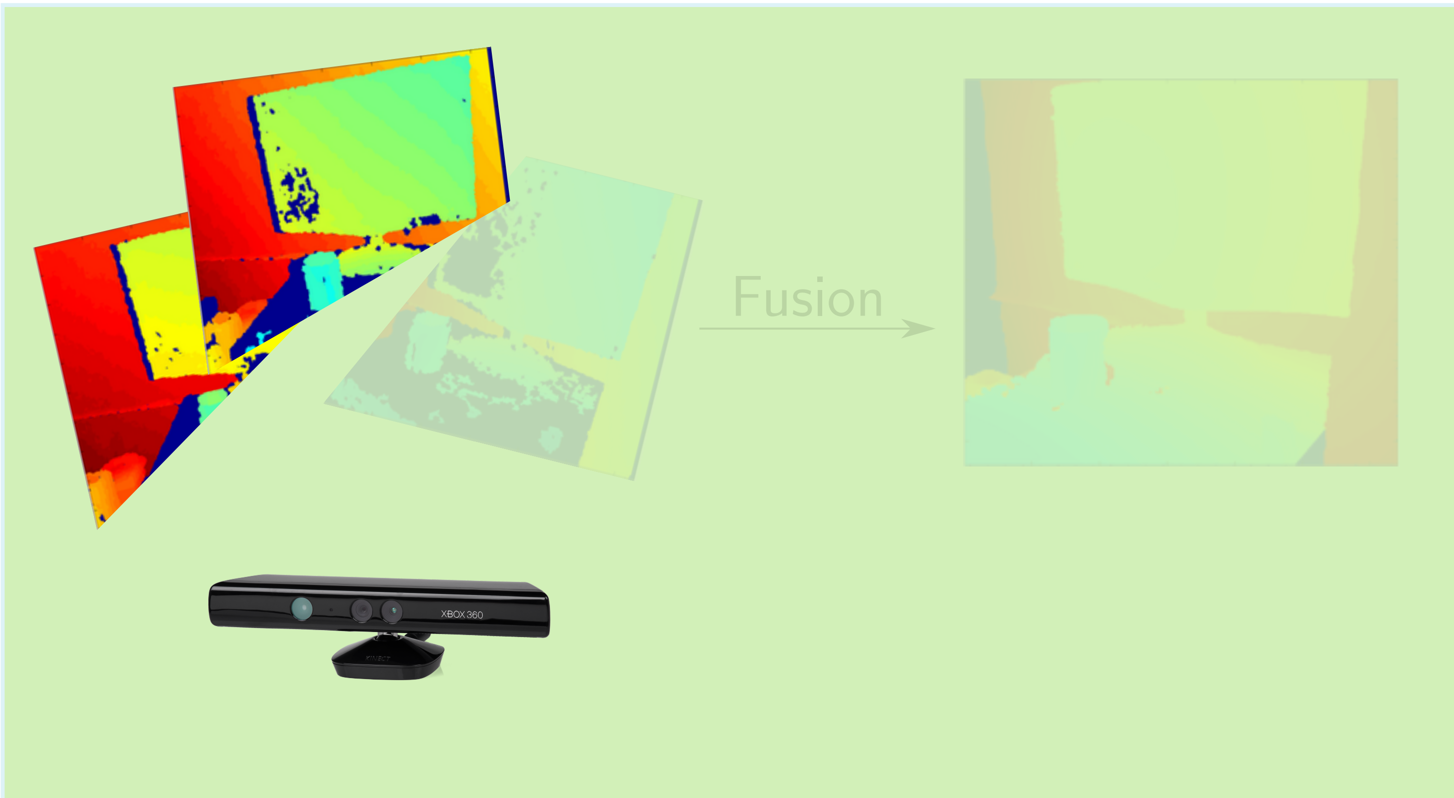

Kinect Sensor

# Kinect



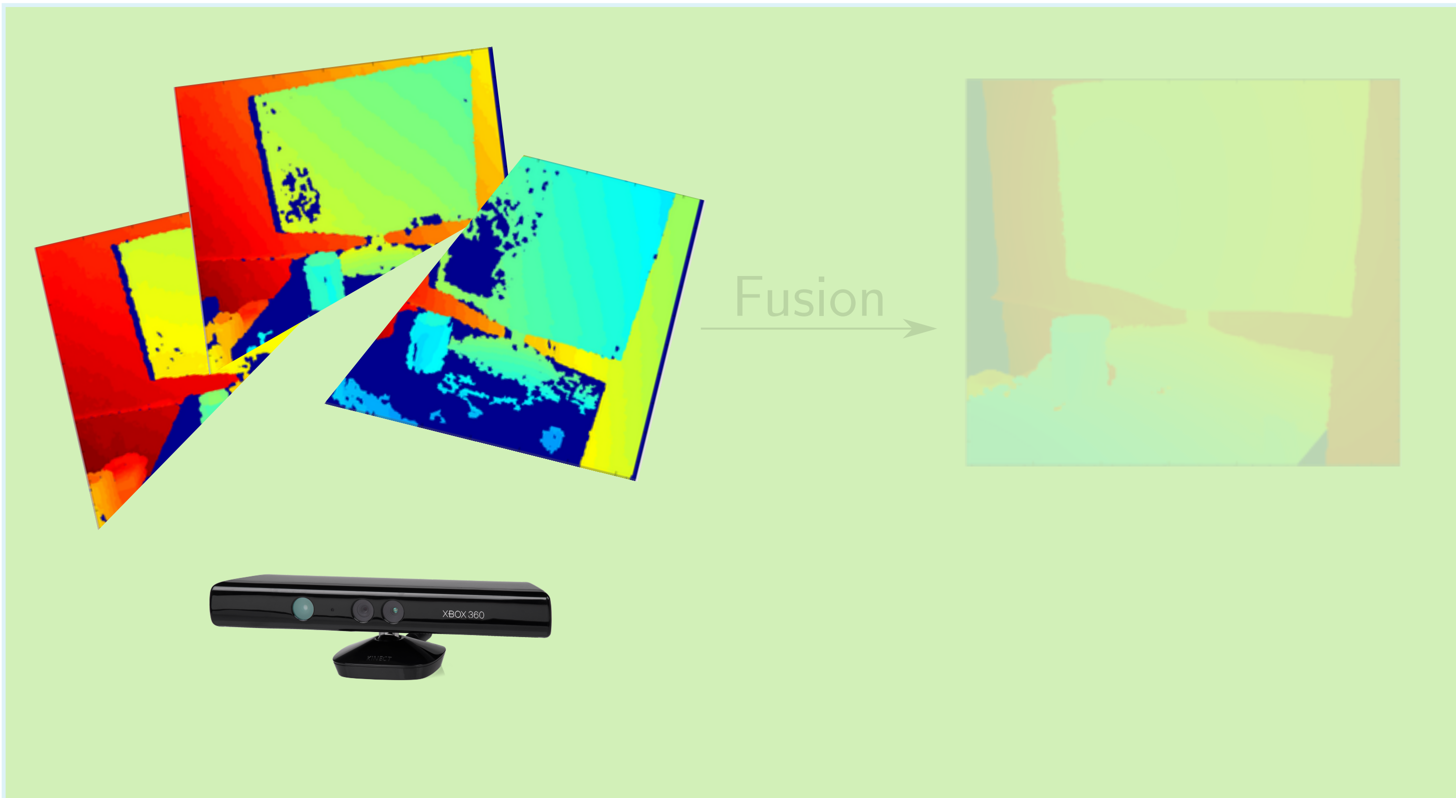

Depth image obtained from the camera: Note the missing values in blue.

# Kinect



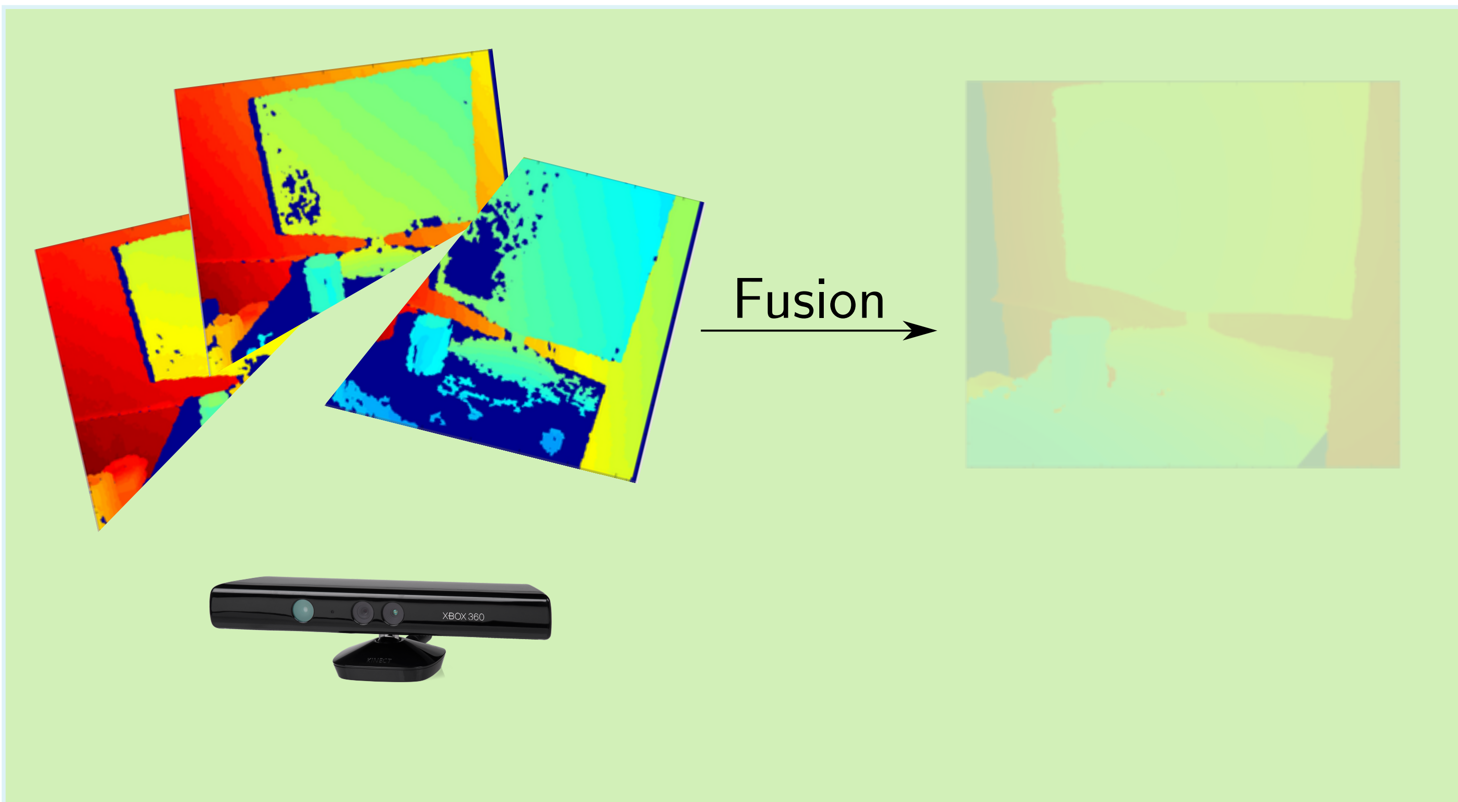

Another image obtained from the camera: Note the missing values in blue.

# Kinect



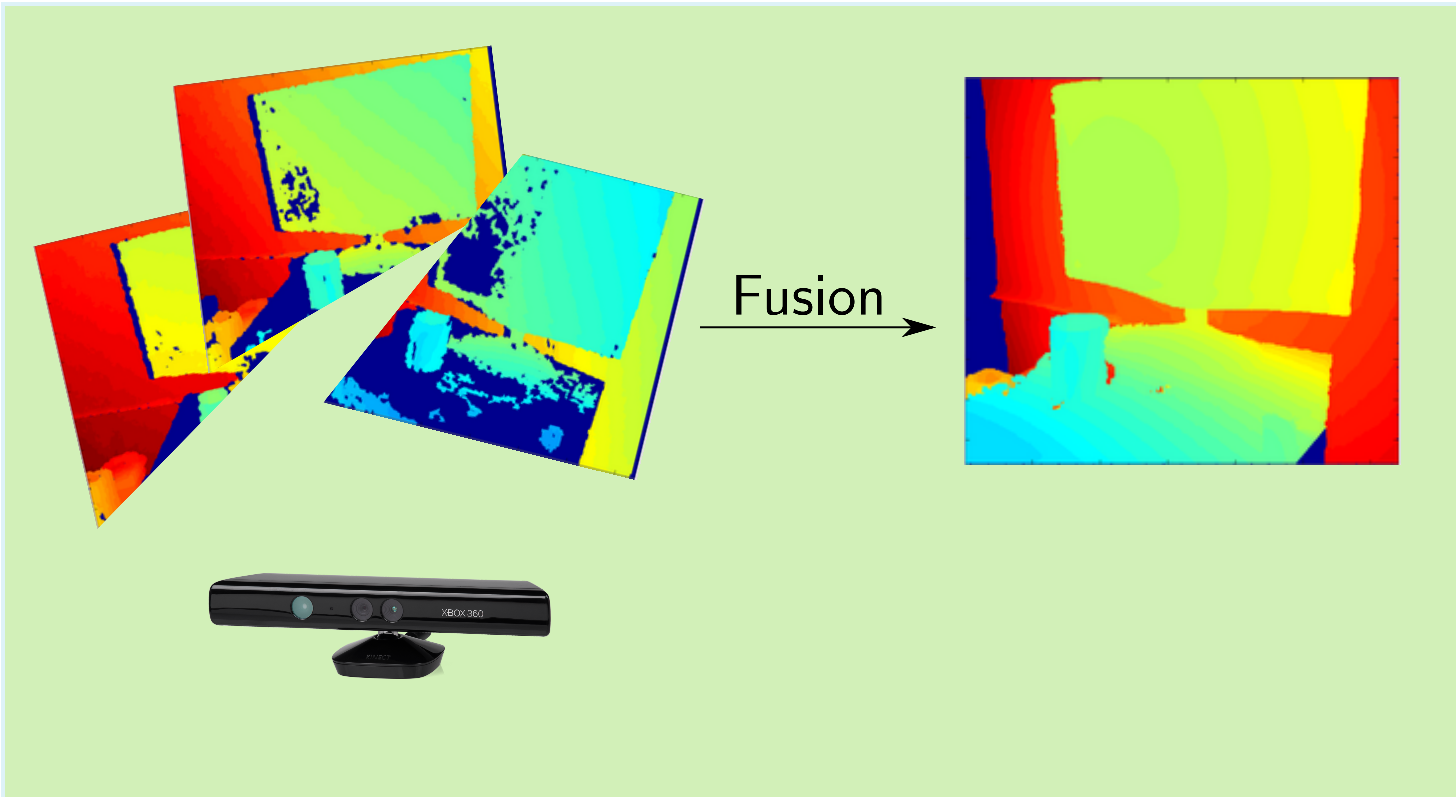

Third image obtained from the camera.

# Kinect



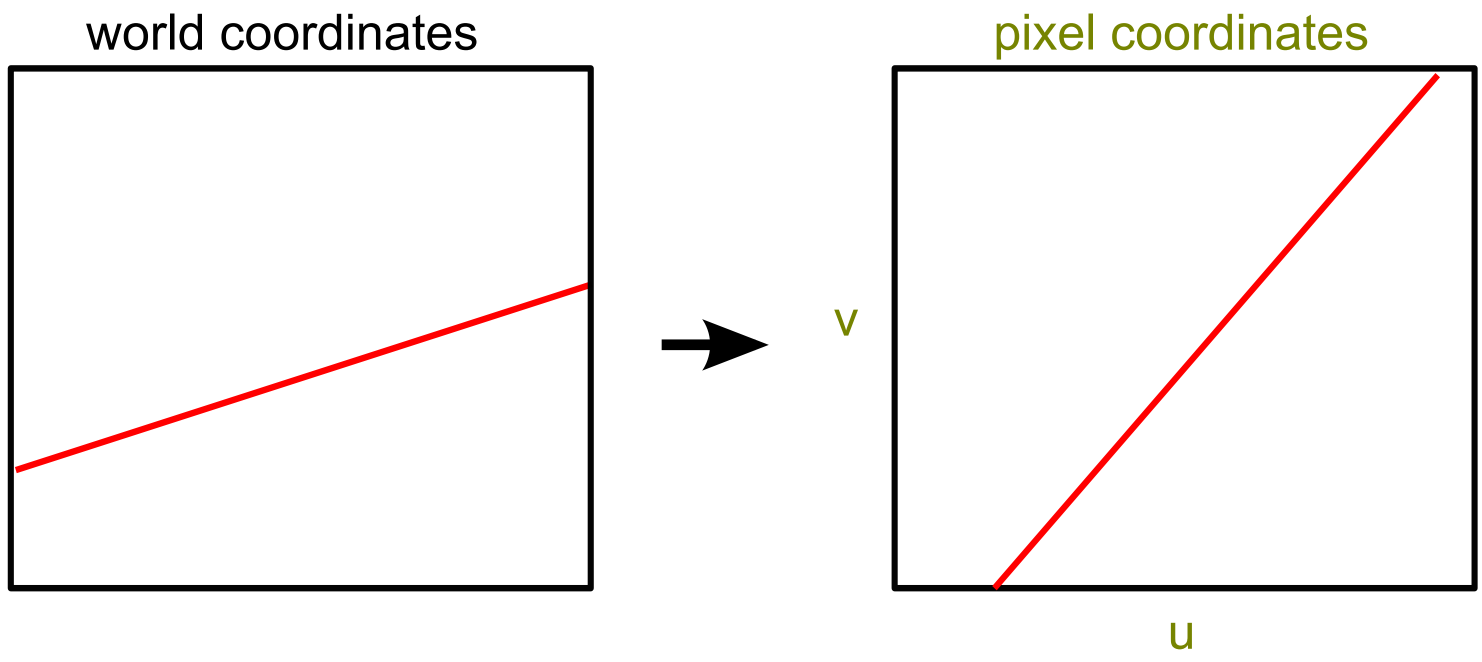

Fuse these images to reconstruct a 3D model.

# Kinect



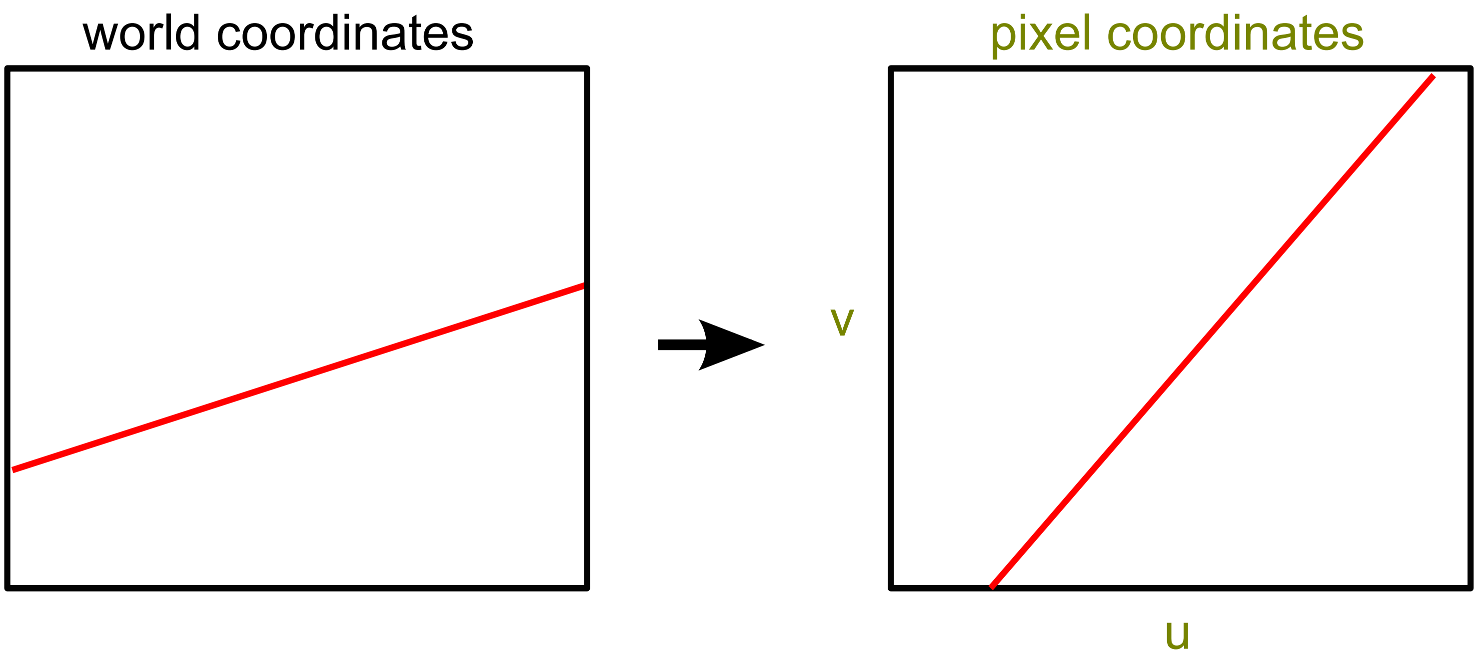

Fusion smooths out the noise and missing values.

# Recap: The cross-ratio




lines in world coordinates project to lines in pixel coordinates

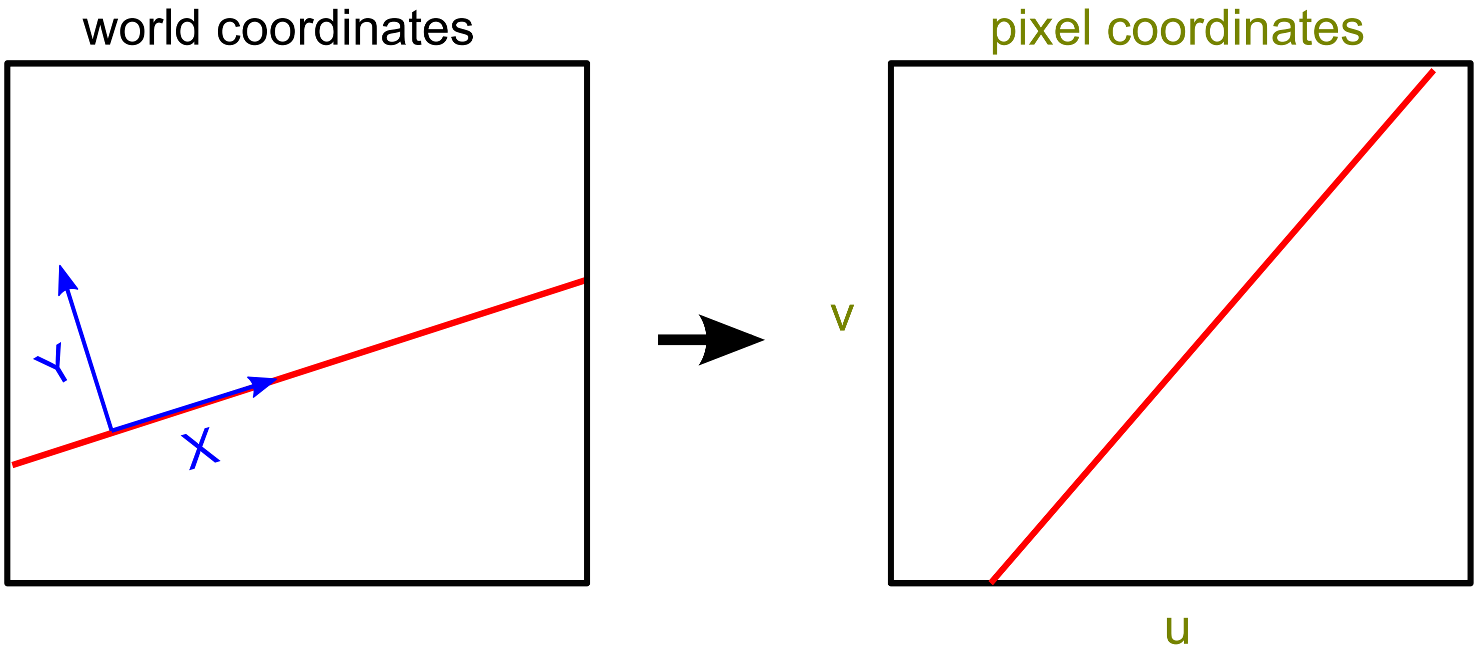
# Recap: The cross-ratio




$$\begin{bmatrix} su \\ sv \\ s \end{bmatrix} = \begin{bmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

projective camera

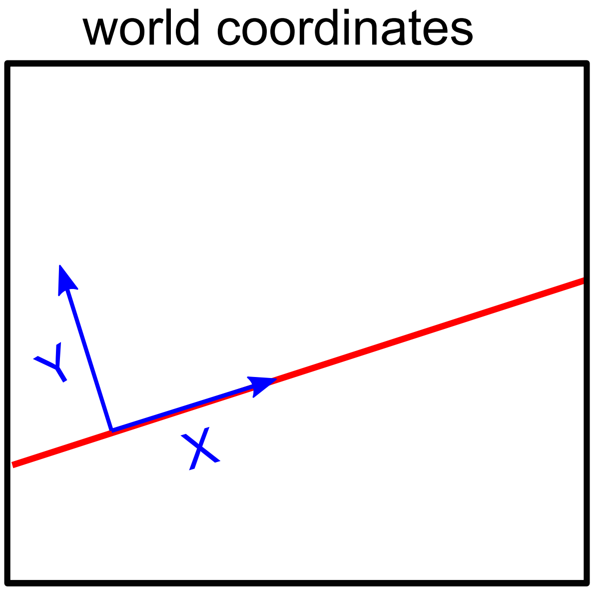
# Recap: The cross-ratio

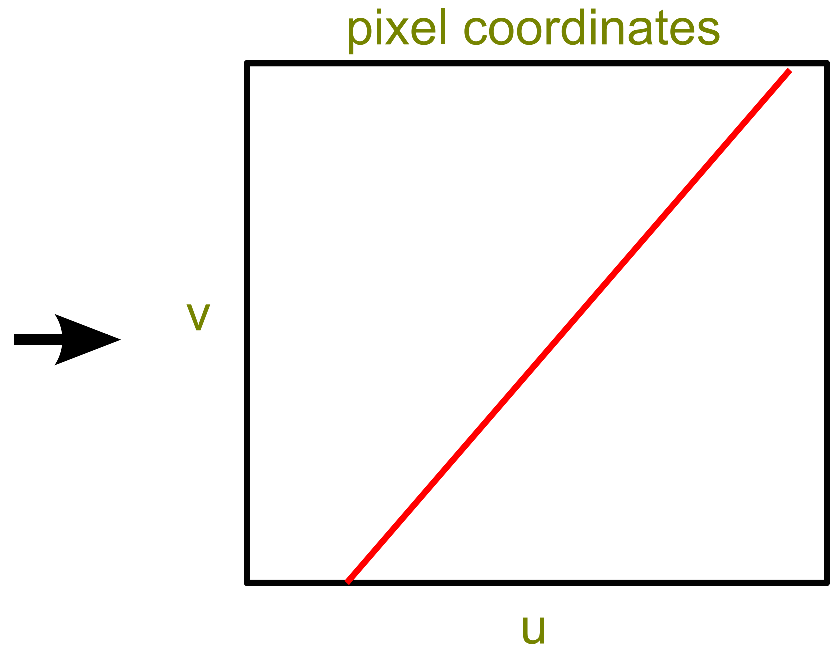



$$\begin{bmatrix} su \\ sv \\ s \end{bmatrix} = \begin{bmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

projective camera

# Recap: The cross-ratio



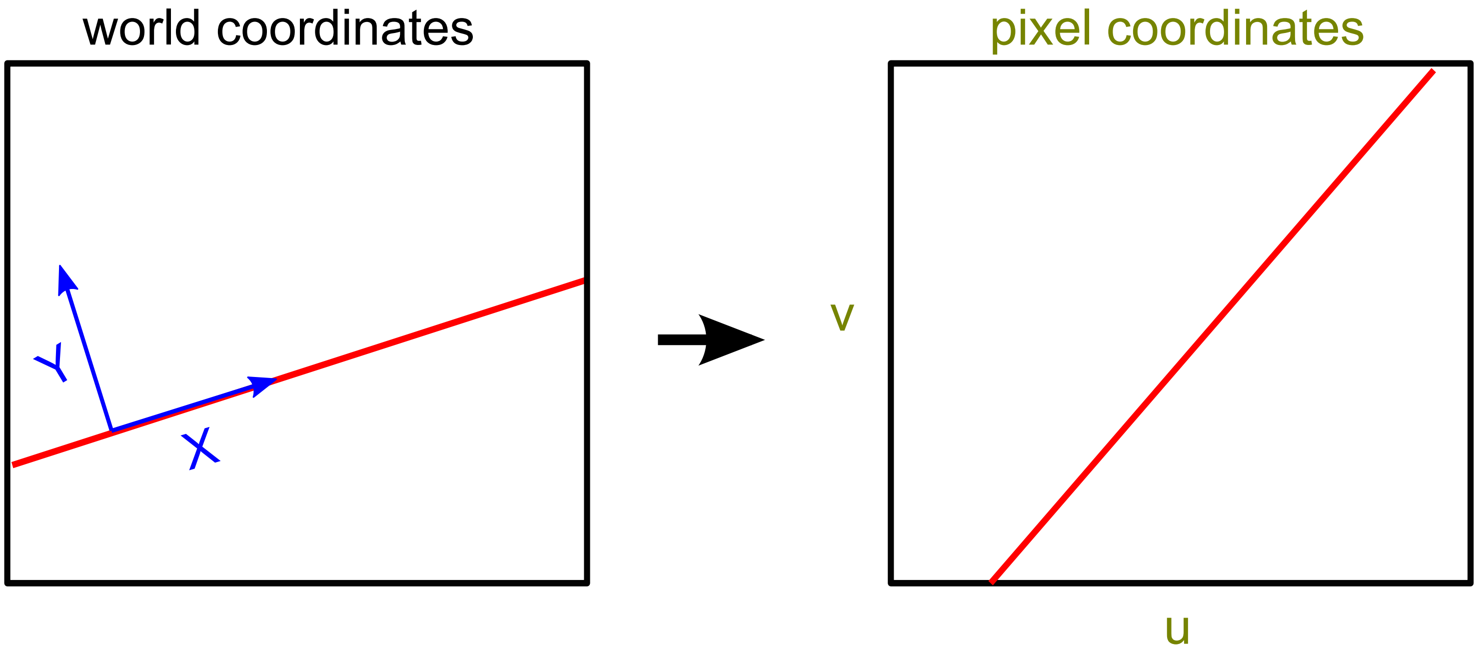

$$\begin{bmatrix} su \\ sv \\ s \end{bmatrix} = \begin{bmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \end{bmatrix} \begin{bmatrix} X \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

wlg can be set to 0

can be removed

wlg align world coordinates with line

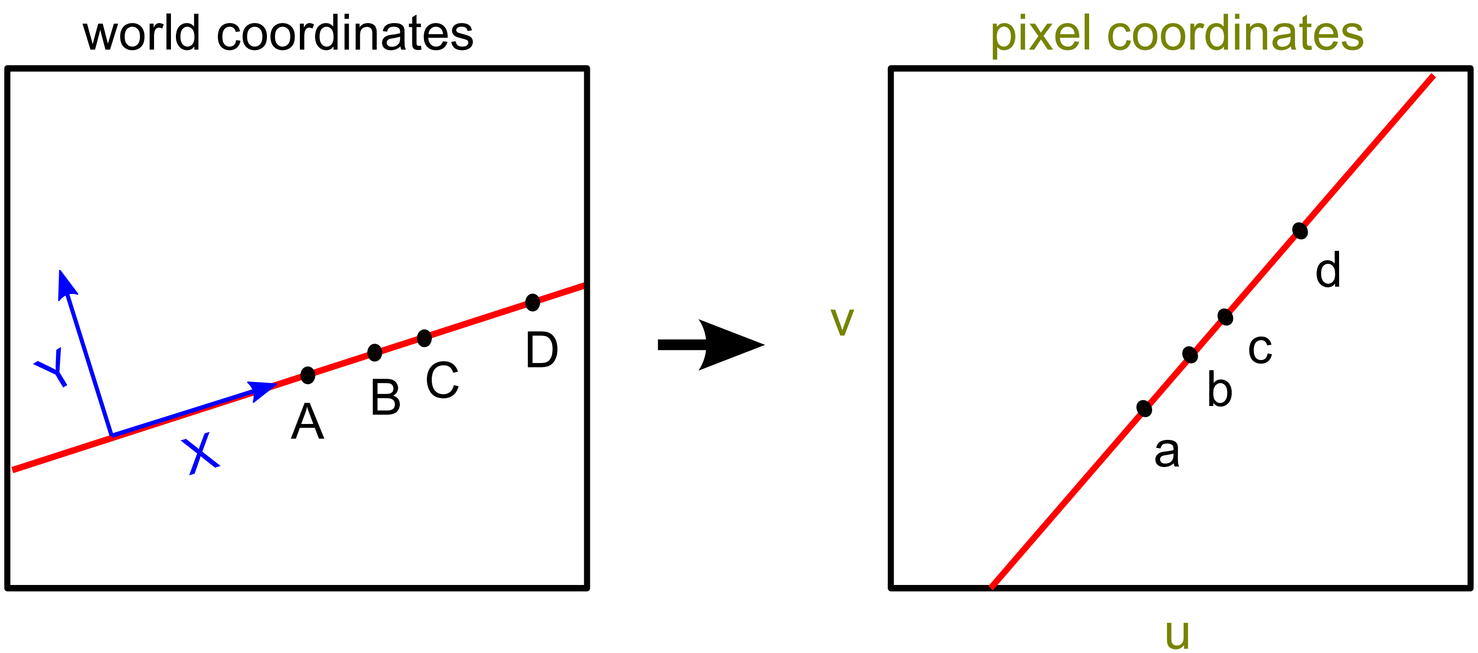
# Recap: The cross-ratio




$$\begin{bmatrix} su \\ sv \\ s \end{bmatrix} = \begin{bmatrix} p_{11} & p_{14} \\ p_{21} & p_{24} \\ p_{31} & p_{34} \end{bmatrix} \begin{bmatrix} X \\ 1 \end{bmatrix}$$

viewing a line

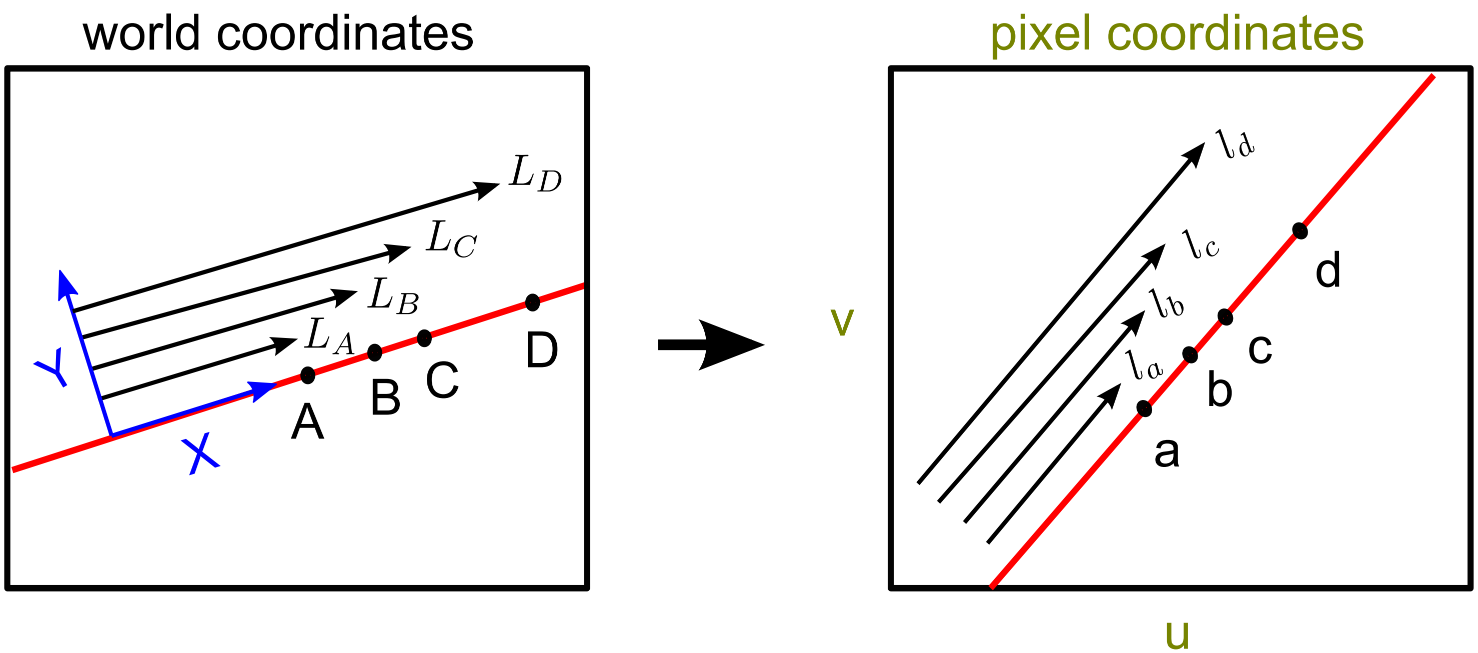
# Recap: The cross-ratio




$$\begin{bmatrix} su \\ sv \\ s \end{bmatrix} = \begin{bmatrix} p_{11} & p_{14} \\ p_{21} & p_{24} \\ p_{31} & p_{34} \end{bmatrix} \begin{bmatrix} X \\ 1 \end{bmatrix}$$

consider colinear points in the world/image

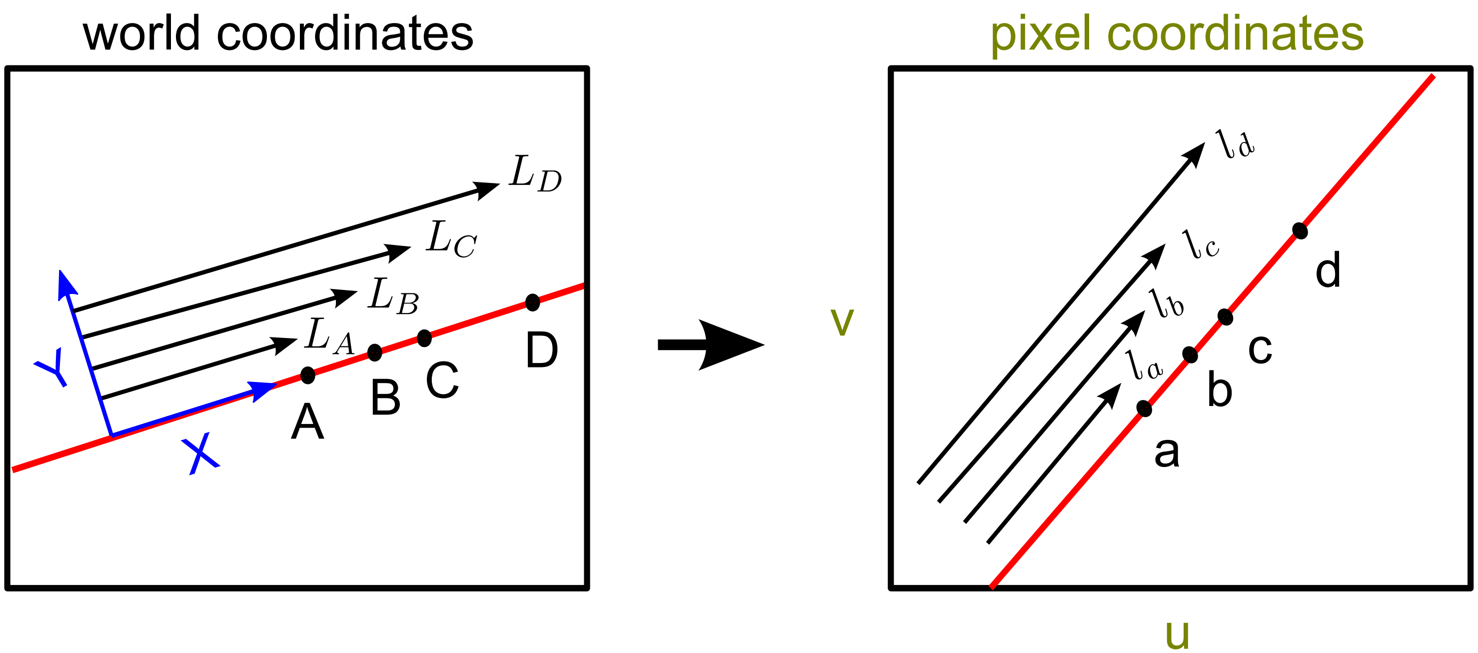
# Recap: The cross-ratio




$$\begin{bmatrix} su \\ sv \\ s \end{bmatrix} = \begin{bmatrix} p_{11} & p_{14} \\ p_{21} & p_{24} \\ p_{31} & p_{34} \end{bmatrix} \begin{bmatrix} X \\ 1 \end{bmatrix}$$

consider colinear points in the world/image

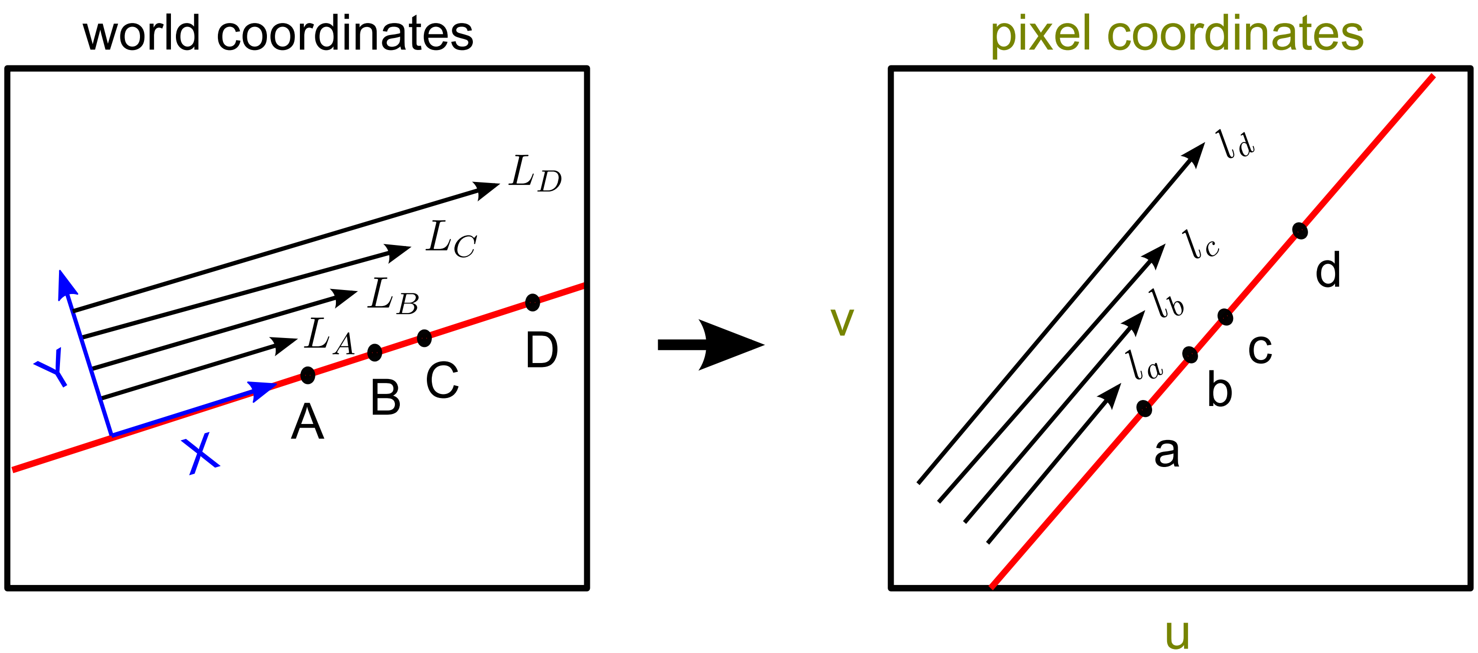
# Recap: The cross-ratio




$$\begin{bmatrix} su \\ sv \\ s \end{bmatrix} = \begin{bmatrix} p_{11} & p_{14} \\ p_{21} & p_{24} \\ p_{31} & p_{34} \end{bmatrix} \begin{bmatrix} X \\ 1 \end{bmatrix} \Longrightarrow \begin{bmatrix} sl_i \\ s \end{bmatrix} = \begin{bmatrix} p & q \\ r & 1 \end{bmatrix} \begin{bmatrix} L_i \\ 1 \end{bmatrix}$$

using the fact that: $l_i \propto u$

# Recap: The cross-ratio

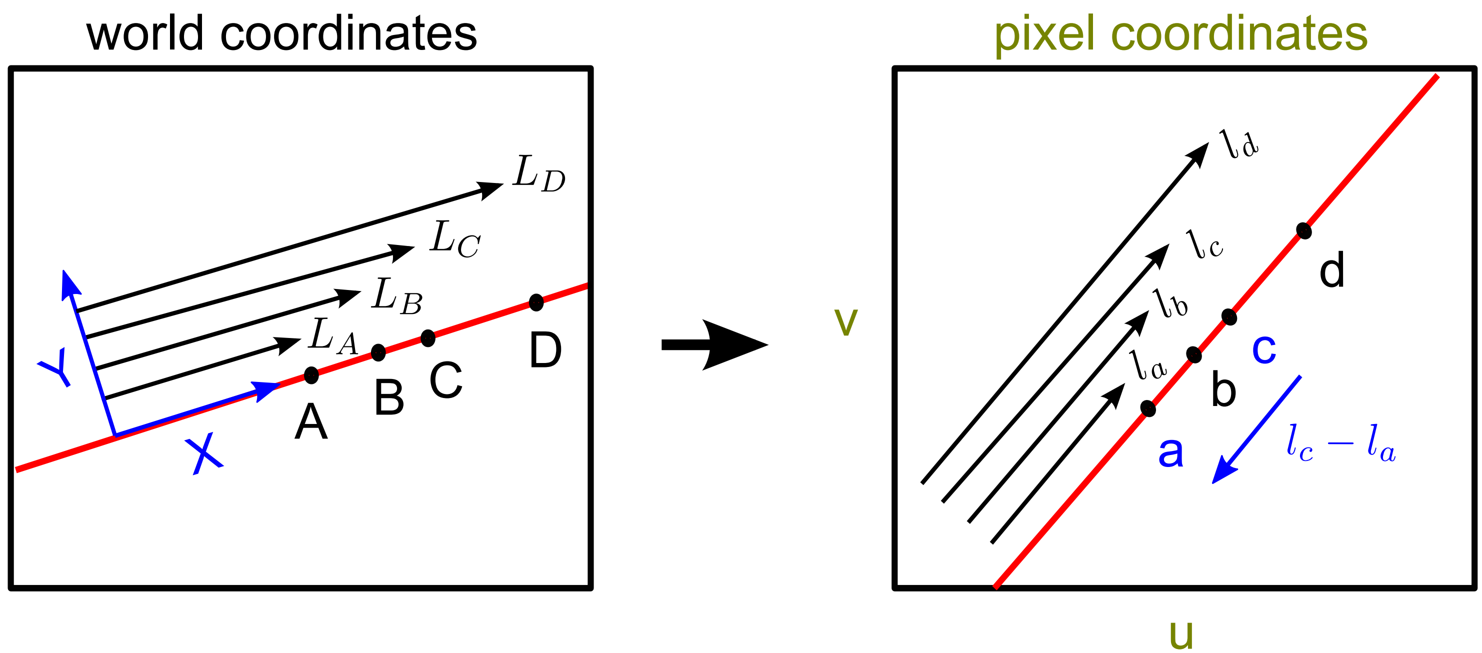



$$\begin{bmatrix} su \\ sv \\ s \end{bmatrix} = \begin{bmatrix} p_{11} & p_{14} \\ p_{21} & p_{24} \\ p_{31} & p_{34} \end{bmatrix} \begin{bmatrix} X \\ 1 \end{bmatrix} \implies \begin{bmatrix} sl_i \\ s \end{bmatrix} = \begin{bmatrix} p & q \\ r & 1 \end{bmatrix} \begin{bmatrix} L_i \\ 1 \end{bmatrix}$$

$$\implies l_i = \frac{pL_i + q}{rL_i + 1}$$

using the fact that: $l_i \propto u$

# Recap: The cross-ratio

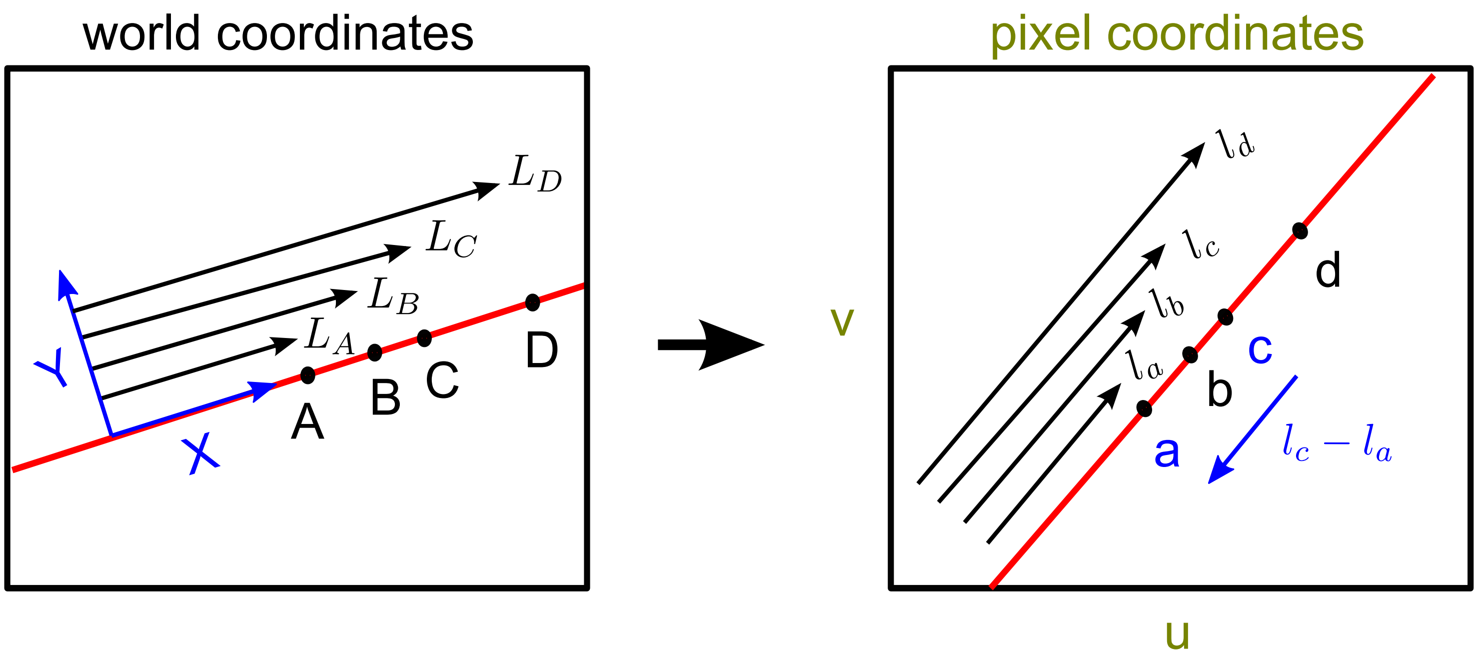



$$\begin{bmatrix} sl_i \\ s \end{bmatrix} = \begin{bmatrix} p & q \\ r & 1 \end{bmatrix} \begin{bmatrix} L_i \\ 1 \end{bmatrix}$$

$$l_i = \frac{pL_i + q}{rL_i + 1}$$

consider ratio of lengths along the line in pixel coordinates

# Recap: The cross-ratio



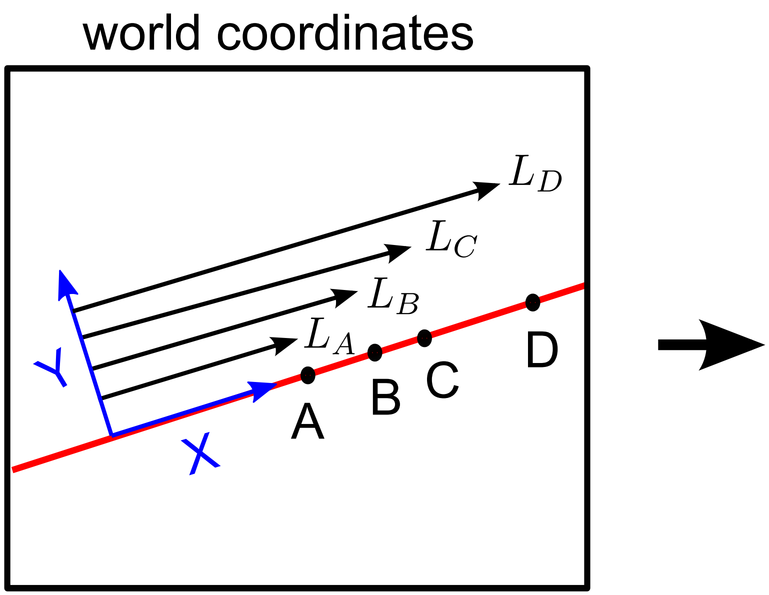

$$l_c - l_a = \frac{pL_c + q}{rL_c + 1} - \frac{pL_a + q}{rL_a + 1}$$

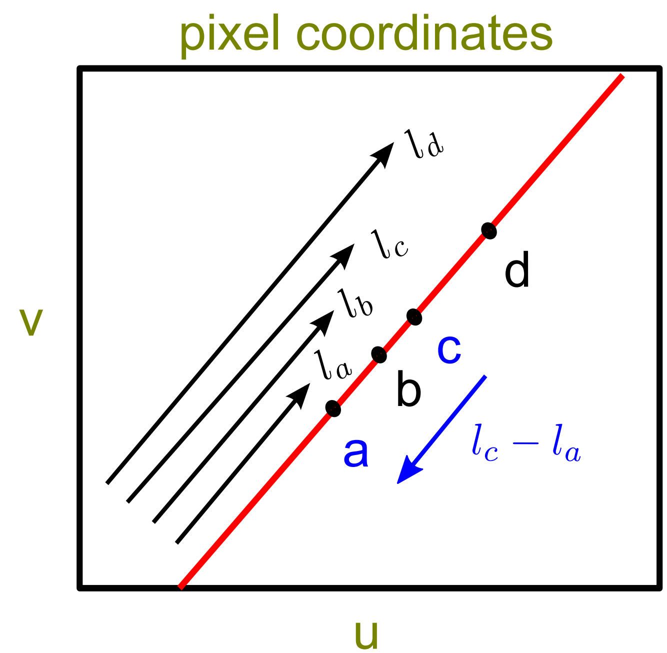
$$\begin{bmatrix} sl_i \\ s \end{bmatrix} = \begin{bmatrix} p & q \\ r & 1 \end{bmatrix} \begin{bmatrix} L_i \\ 1 \end{bmatrix}$$

$$l_i = \frac{pL_i + q}{rL_i + 1}$$

consider ratio of lengths along the line in pixel coordinates

# Recap: The cross-ratio




$$l_c - l_a = \frac{pL_c + q}{rL_c + 1} - \frac{pL_a + q}{rL_a + 1}$$

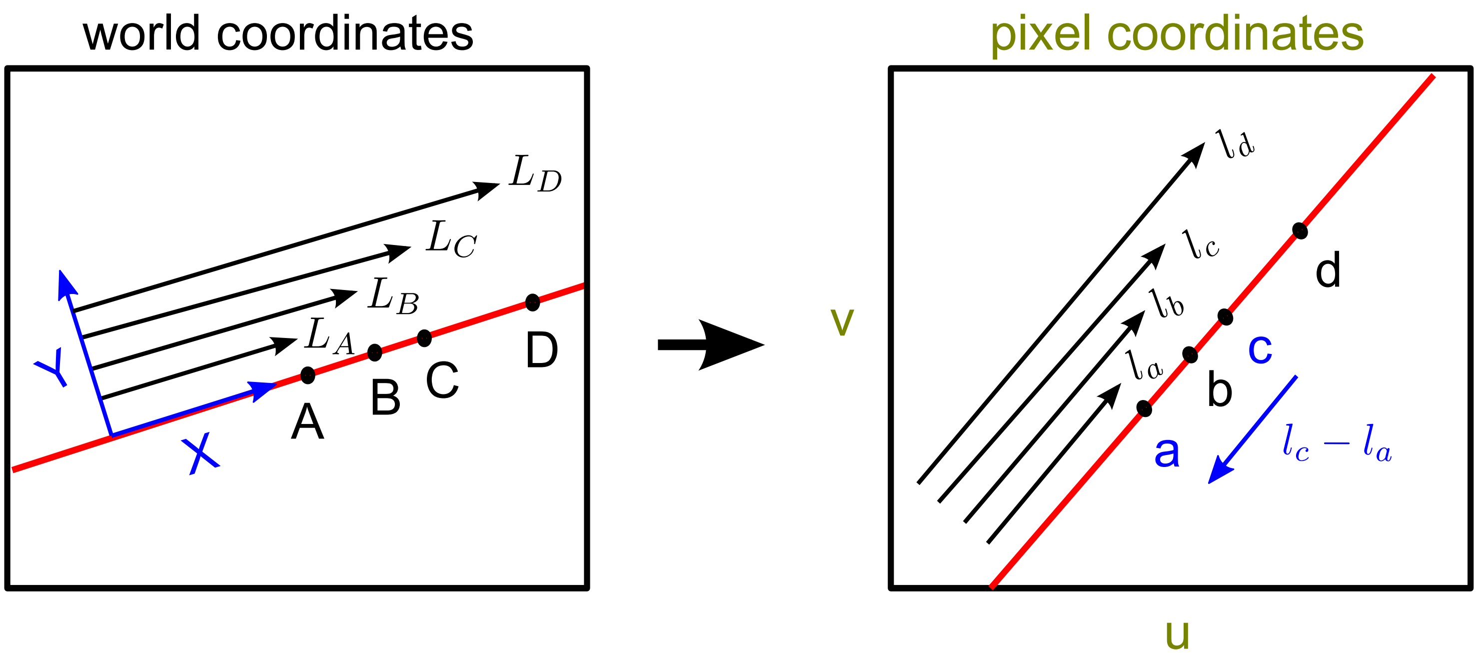
$$= \frac{(pL_c + q)(rL_a + 1) - (pL_a + q)(rL_c + 1)}{(rL_c + 1)(rL_a + 1)}$$

$$\begin{bmatrix} sl_i \\ s \end{bmatrix} = \begin{bmatrix} p & q \\ r & 1 \end{bmatrix} \begin{bmatrix} L_i \\ 1 \end{bmatrix}$$

$$l_i = \frac{pL_i + q}{rL_i + 1}$$

consider ratio of lengths along the line in pixel coordinates

# Recap: The cross-ratio




$$l_c - l_a = \frac{pL_c + q}{rL_c + 1} - \frac{pL_a + q}{rL_a + 1}$$

$$= \frac{(pL_c + q)(rL_a + 1) - (pL_a + q)(rL_c + 1)}{(rL_c + 1)(rL_a + 1)}$$

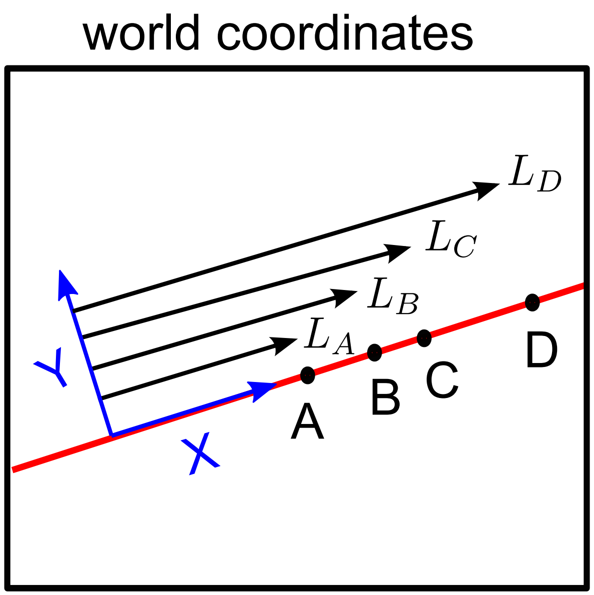
$$= \frac{(L_c - L_a)(p - qr)}{(rL_c + 1)(rL_a + 1)}$$

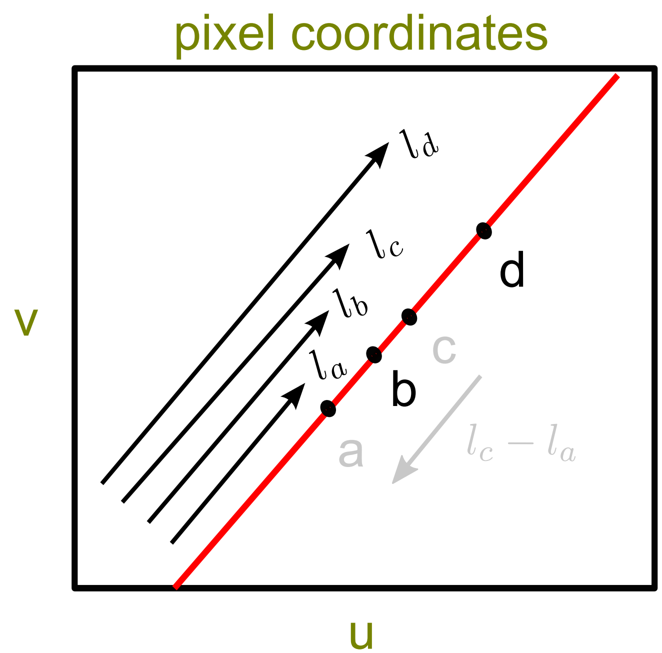
$$\begin{bmatrix} sl_i \\ s \end{bmatrix} = \begin{bmatrix} p & q \\ r & 1 \end{bmatrix} \begin{bmatrix} L_i \\ 1 \end{bmatrix}$$

$$l_i = \frac{pL_i + q}{rL_i + 1}$$

consider ratio of lengths along the line in pixel coordinates

# Recap: The cross-ratio



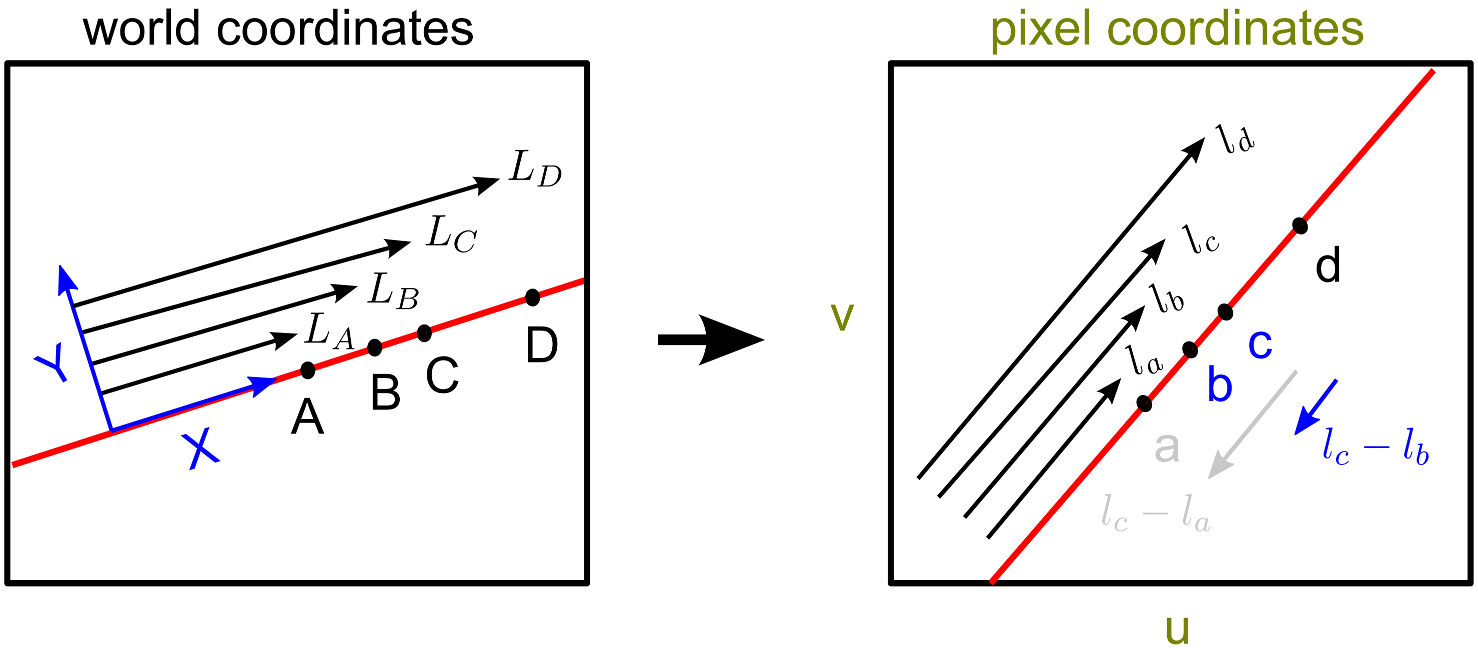

$$l_c - l_a \;=\; \frac{(L_c - L_a)(p - qr)}{(rL_c + 1)(rL_a + 1)}$$

$$\begin{bmatrix} sl_i \\ s \end{bmatrix} = \begin{bmatrix} p & q \\ r & 1 \end{bmatrix} \begin{bmatrix} L_i \\ 1 \end{bmatrix}$$

$$l_i = \frac{pL_i + q}{rL_i + 1}$$

consider ratio of lengths along the line in pixel coordinates

# Recap: The cross-ratio



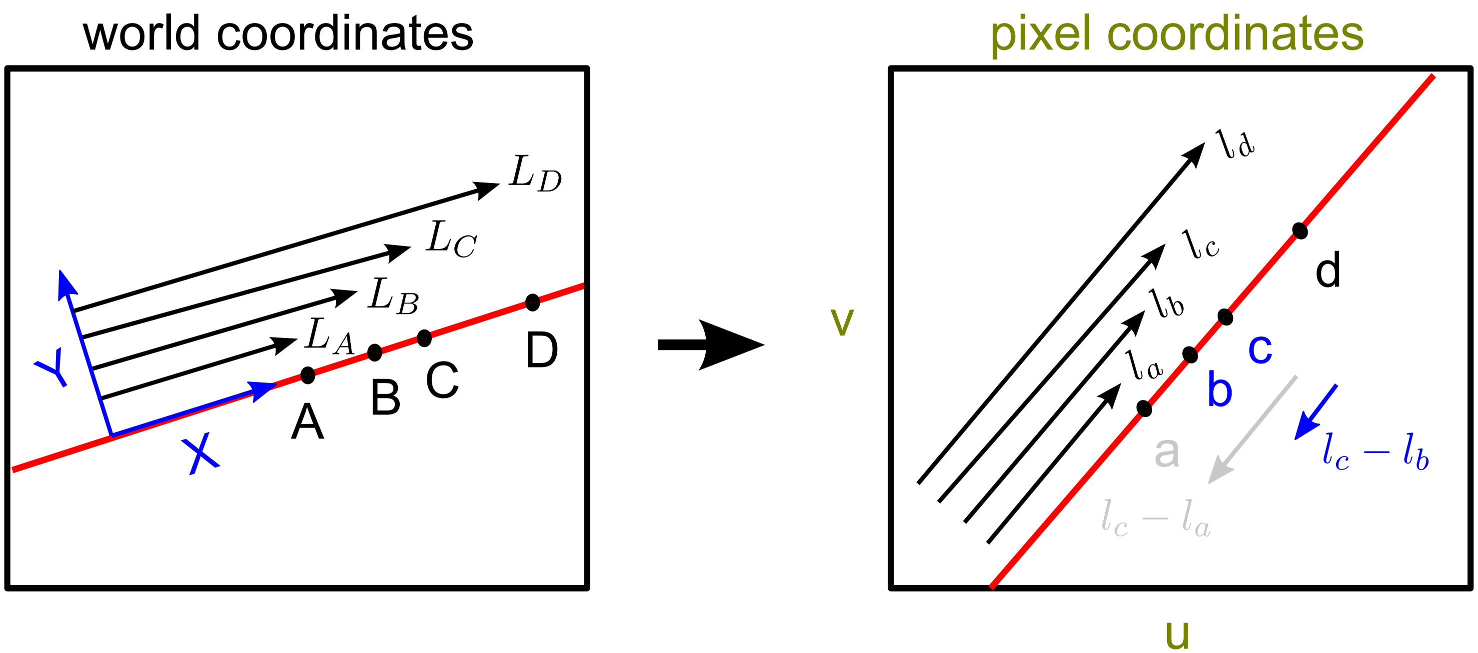

$$l_c - l_a = \frac{(L_c - L_a)(p - qr)}{(rL_c + 1)(rL_a + 1)}$$

$$\begin{bmatrix} sl_i \\ s \end{bmatrix} = \begin{bmatrix} p & q \\ r & 1 \end{bmatrix} \begin{bmatrix} L_i \\ 1 \end{bmatrix}$$

$$l_i = \frac{pL_i + q}{rL_i + 1}$$

consider ratio of lengths along the line in pixel coordinates

# Recap: The cross-ratio




$$l_c - l_a = \frac{(L_c - L_a)(p - qr)}{(rL_c + 1)(rL_a + 1)}$$

$$l_c - l_b = \frac{(L_c - L_b)(p - qr)}{(rL_c + 1)(rL_b + 1)}$$

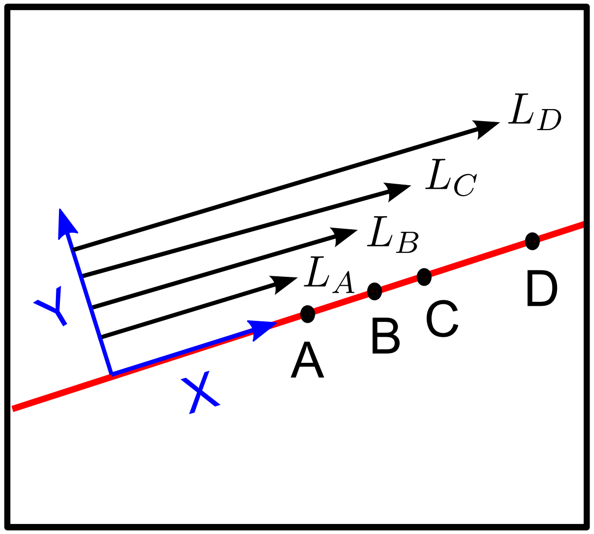
$$\begin{bmatrix} sl_i \\ s \end{bmatrix} = \begin{bmatrix} p & q \\ r & 1 \end{bmatrix} \begin{bmatrix} L_i \\ 1 \end{bmatrix}$$

$$l_i = \frac{pL_i + q}{rL_i + 1}$$

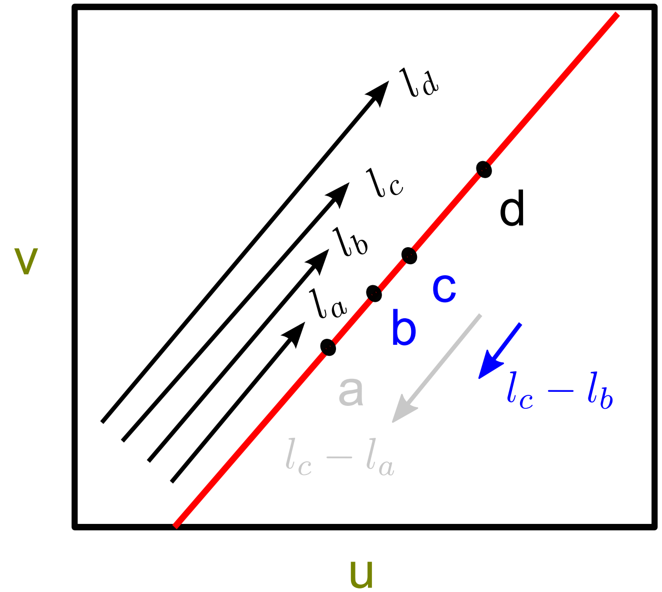
consider ratio of lengths along the line in pixel coordinates

# Recap: The cross-ratio

world coordinates




pixel coordinates




$$l_c - l_a = \frac{(L_c - L_a)(p - qr)}{(rL_c + 1)(rL_a + 1)}$$

$$l_c - l_b = \frac{(L_c - L_b)(p - qr)}{(rL_c + 1)(rL_b + 1)}$$

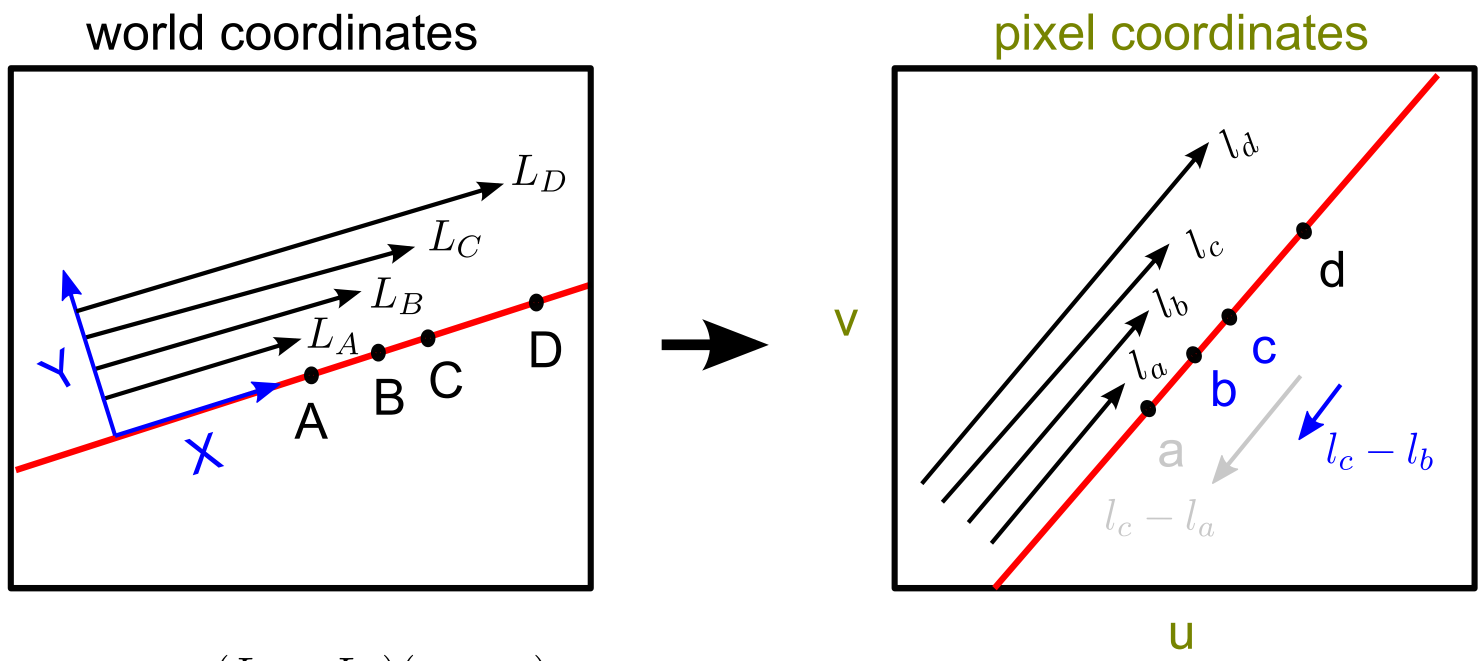
$$\frac{l_c - l_a}{l_c - l_b} = \frac{(L_c - L_a)(rL_b + 1)}{(L_c - L_b)(rL_a + 1)}$$

$$\begin{bmatrix} sl_i \\ s \end{bmatrix} = \begin{bmatrix} p & q \\ r & 1 \end{bmatrix} \begin{bmatrix} L_i \\ 1 \end{bmatrix}$$

$$l_i = \frac{pL_i + q}{rL_i + 1}$$

ratios of lengths are NOT invariant (unlike the affine case)

# Recap: The cross-ratio




$$l_c - l_a = \frac{(L_c - L_a)(p - qr)}{(rL_c + 1)(rL_a + 1)}$$

$$l_c - l_b = \frac{(L_c - L_b)(p - qr)}{(rL_c + 1)(rL_b + 1)}$$

$$\frac{l_c - l_a}{l_c - l_b} = \frac{(L_c - L_a)(rL_b + 1)}{(L_c - L_b)(rL_a + 1)}$$

depends on r

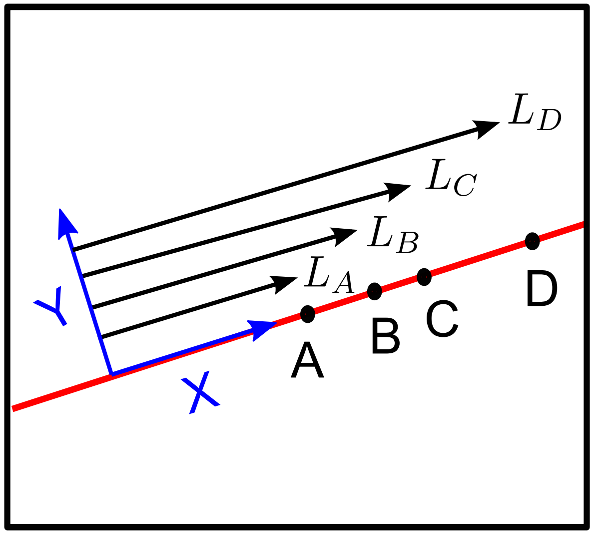
$$\begin{bmatrix} sl_i \\ s \end{bmatrix} = \begin{bmatrix} p & q \\ r & 1 \end{bmatrix} \begin{bmatrix} L_i \\ 1 \end{bmatrix}$$

$$l_i = \frac{pL_i + q}{rL_i + 1}$$

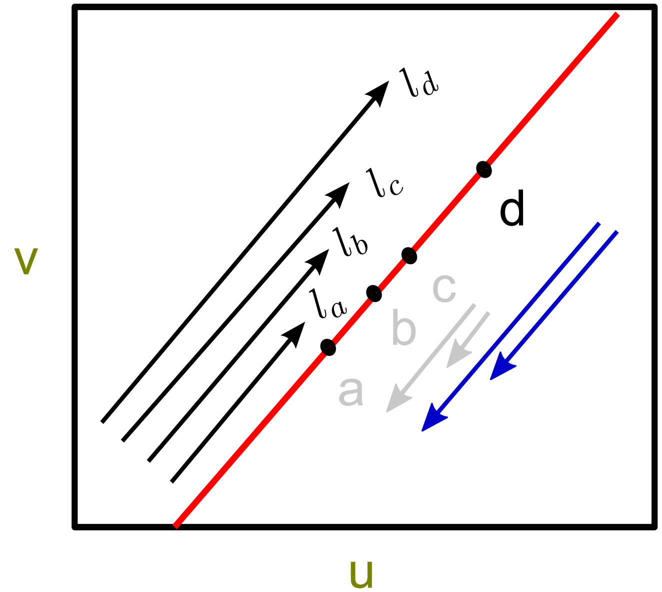
ratios of lengths are NOT invariant (unlike the affine case)

# Recap: The cross-ratio

world coordinates




pixel coordinates




$$l_c - l_a = \frac{(L_c - L_a)(p - qr)}{(rL_c + 1)(rL_a + 1)}$$

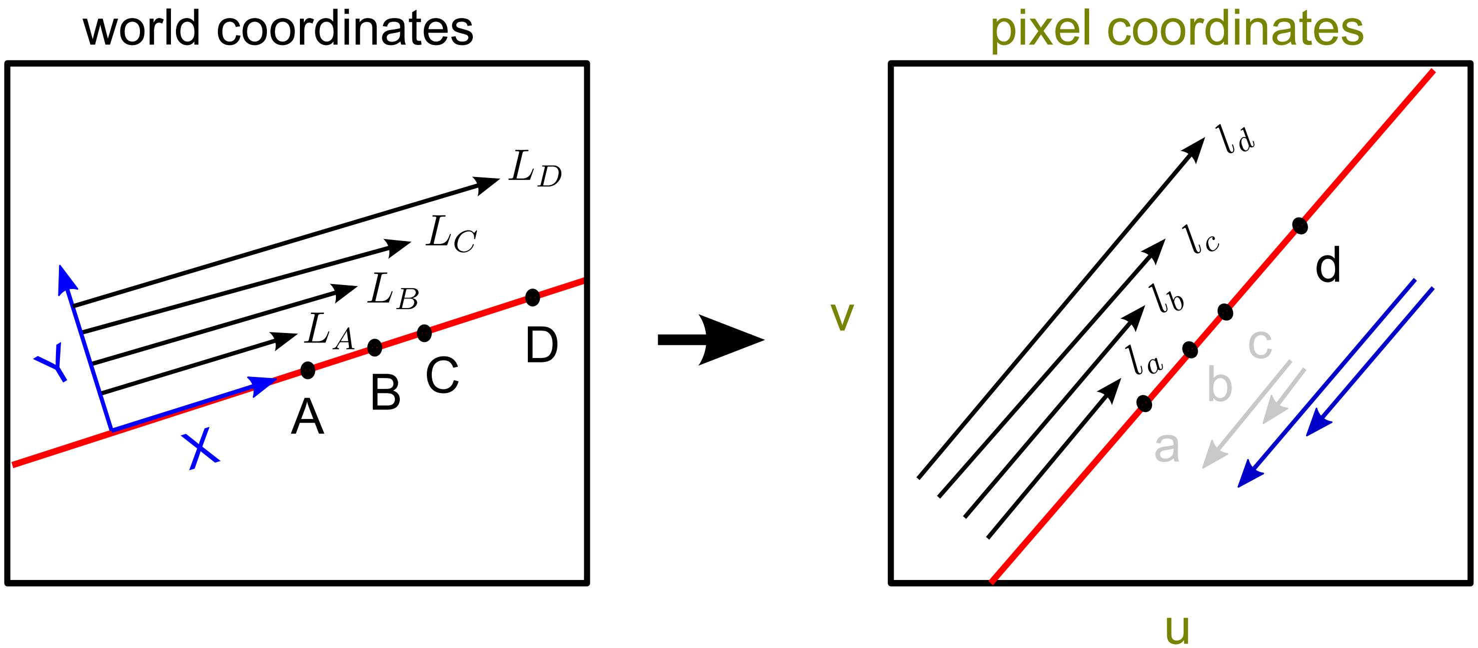
$$l_c - l_b = \frac{(L_c - L_b)(p - qr)}{(rL_c + 1)(rL_b + 1)}$$

$$\frac{l_c - l_a}{l_c - l_b} = \frac{(L_c - L_a)(rL_b + 1)}{(L_c - L_b)(rL_a + 1)}$$

$$\frac{l_d - l_a}{l_d - l_b}$$

BUT now consider ratio derived from point b to points a and b

# Recap: The cross-ratio




$$l_c - l_a = \frac{(L_c - L_a)(p - qr)}{(rL_c + 1)(rL_a + 1)}$$

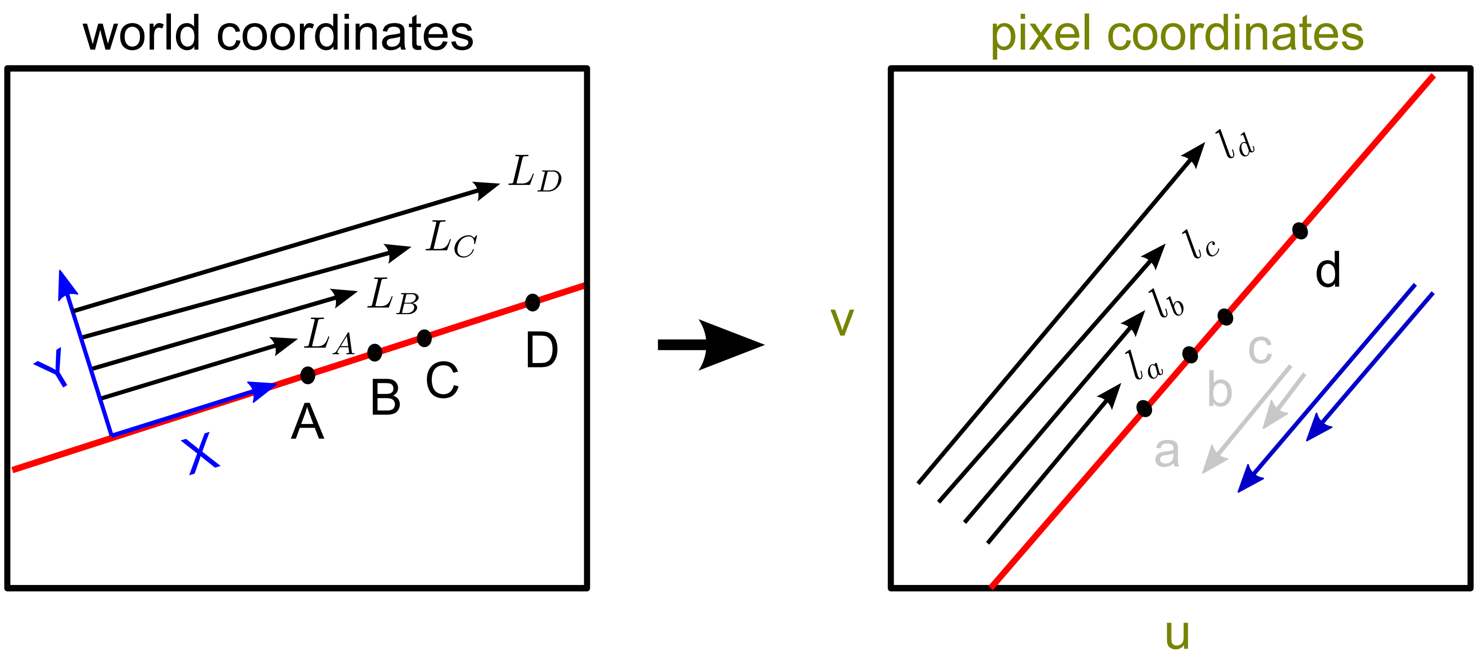
$$l_c - l_b = \frac{(L_c - L_b)(p - qr)}{(rL_c + 1)(rL_b + 1)}$$

$$\frac{l_c - l_a}{l_c - l_b} = \frac{(L_c - L_a)(rL_b + 1)}{(L_c - L_b)(rL_a + 1)}$$

$$\frac{l_d - l_a}{l_d - l_b} = \frac{(L_d - L_a)(rL_b + 1)}{(L_d - L_b)(rL_a + 1)}$$

BUT now consider ratio derived from point b to points a and b

# Recap: The cross-ratio




$$l_c - l_a = \frac{(L_c - L_a)(p - qr)}{(rL_c + 1)(rL_a + 1)}$$

$$l_c - l_b = \frac{(L_c - L_b)(p - qr)}{(rL_c + 1)(rL_b + 1)}$$

$$\frac{l_c - l_a}{l_c - l_b} = \frac{(L_c - L_a)(rL_b + 1)}{(L_c - L_b)(rL_a + 1)}$$

$$\frac{l_d - l_a}{l_d - l_b} = \frac{(L_d - L_a)(rL_b + 1)}{(L_d - L_b)(rL_a + 1)}$$

$\div$

$$\frac{(l_d - l_a)(l_c - l_b)}{(l_d - l_b)(l_c - l_a)} = \frac{(L_d - L_a)(L_c - L_b)}{(L_d - L_b)(L_c - L_a)}$$

the CROSS-RATIO is the ratio of these two quantities