Two-view geometry

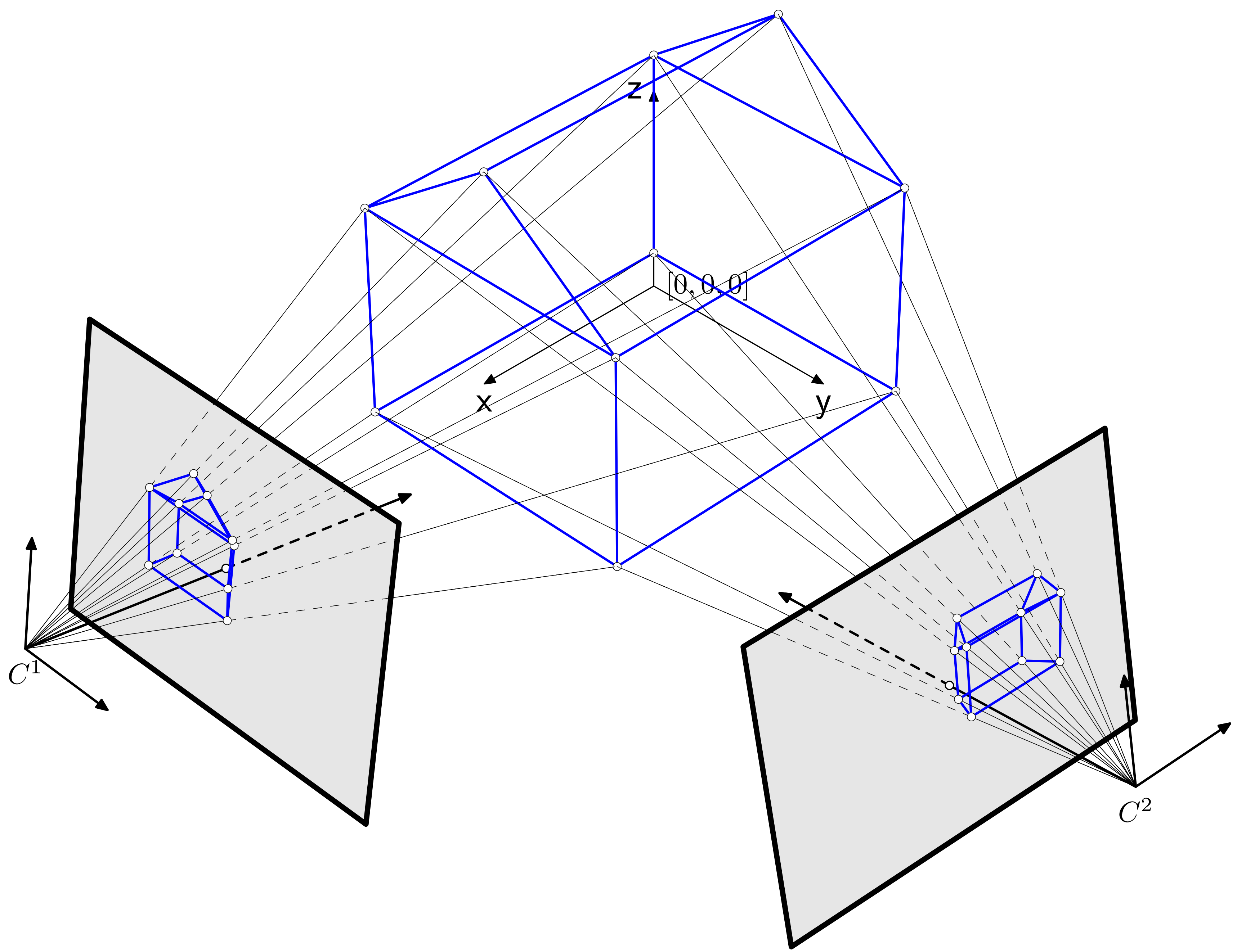
Tomáš Svoboda, svoboda@cmp.felk.cvut.cz
Czech Technical University in Prague, Center for Machine Perception
http://cmp.felk.cvut.cz

Last update: November 9, 2009

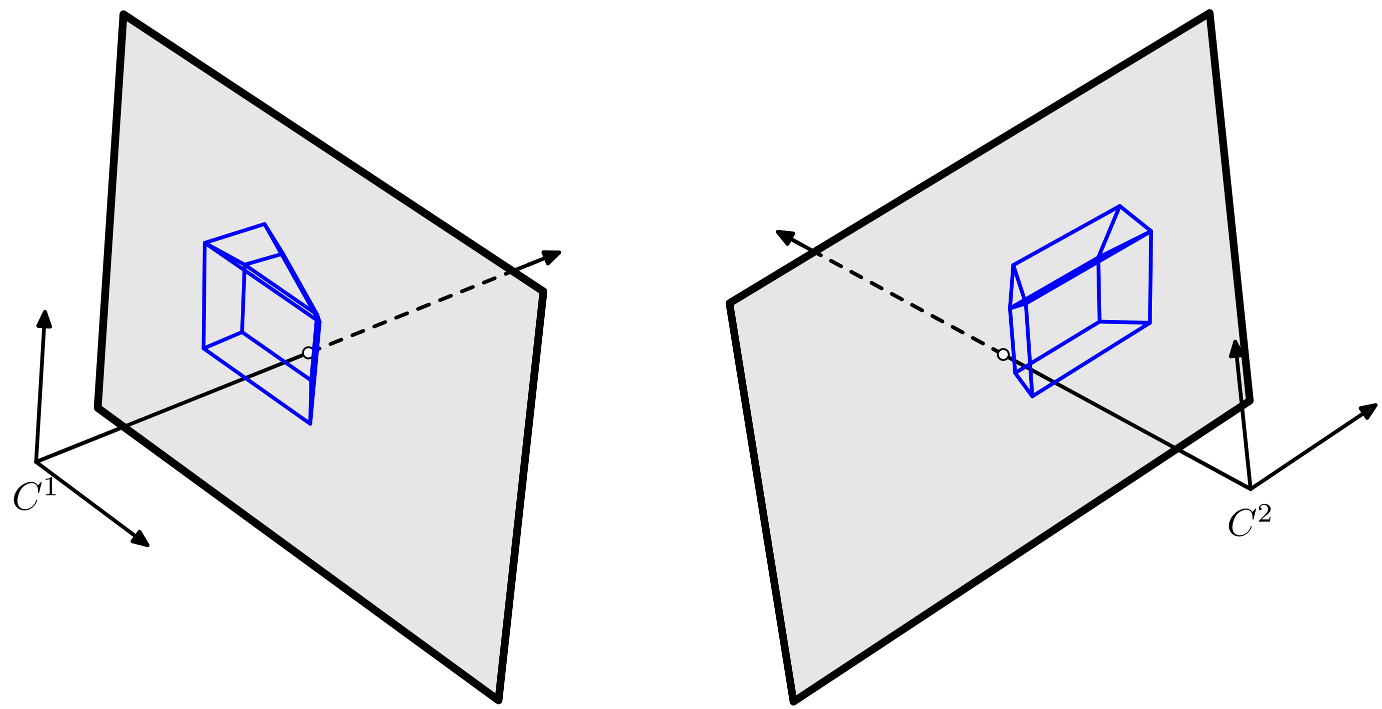
Talk Outline

- Epipolar geometry
- Estimation of the Fundamental matrix
- Camera motion
- Reconstruction of scene structure

Motivation

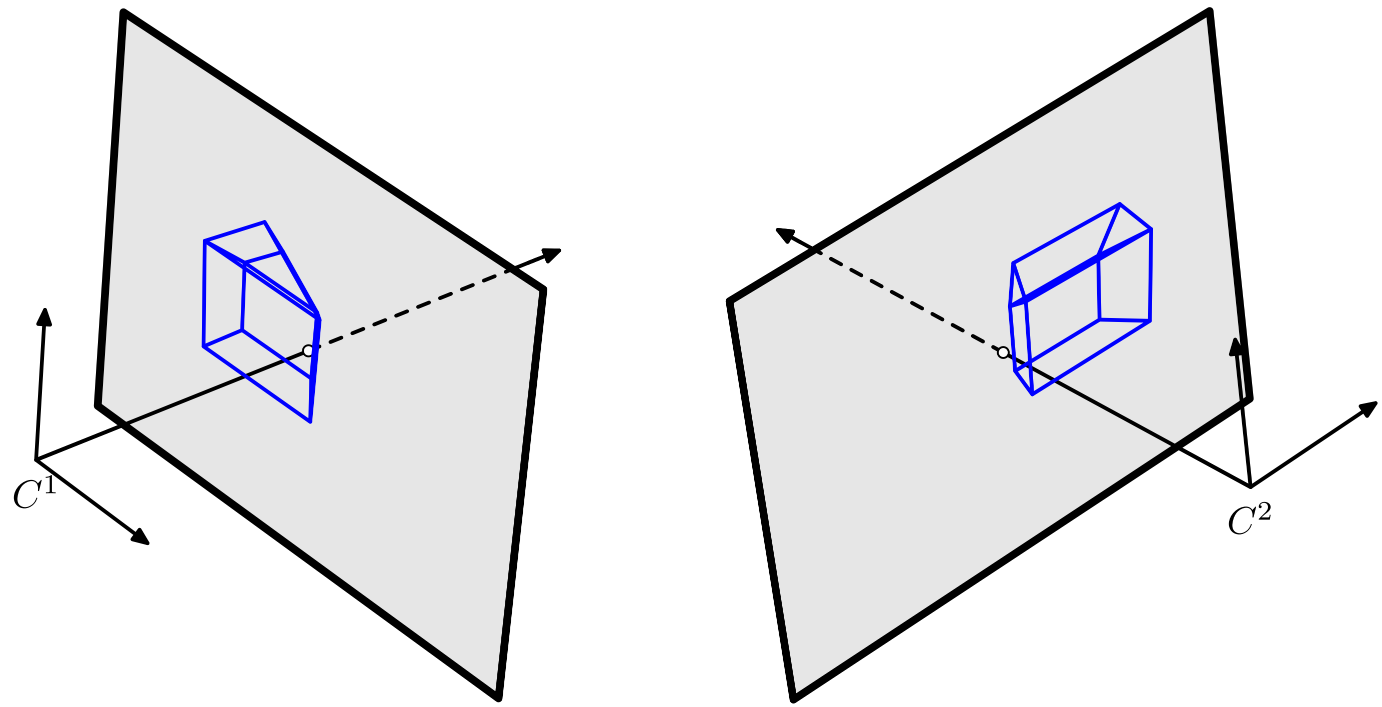


Two projections of a rigid 3D scene



- The projections are clearly different.
- Can the difference tell something about the camera positions?
- and about the scene structure?

Two projections of a rigid 3D scene

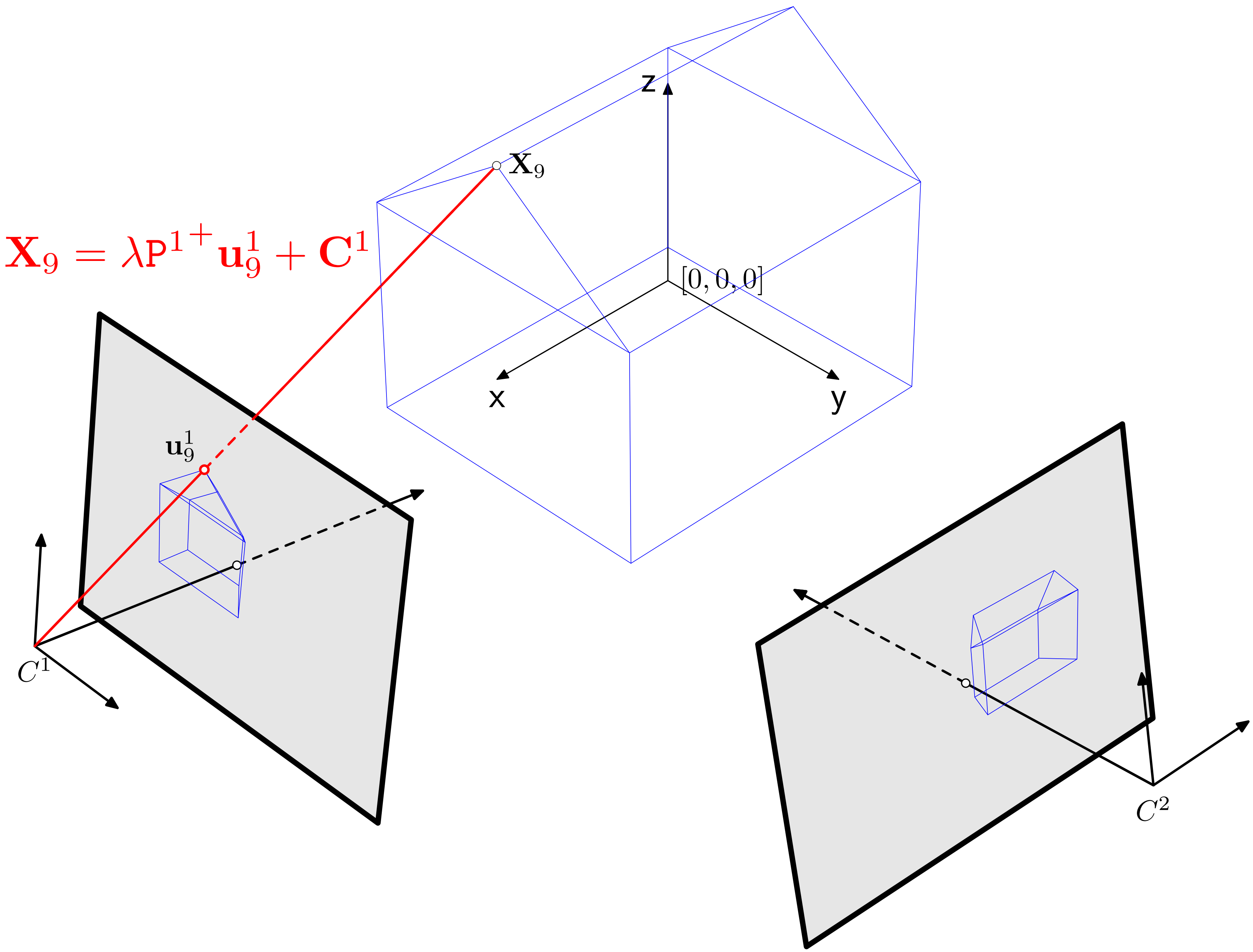


- The projections are clearly different.
- Can the difference tell something about the camera positions?
- and about the scene structure?

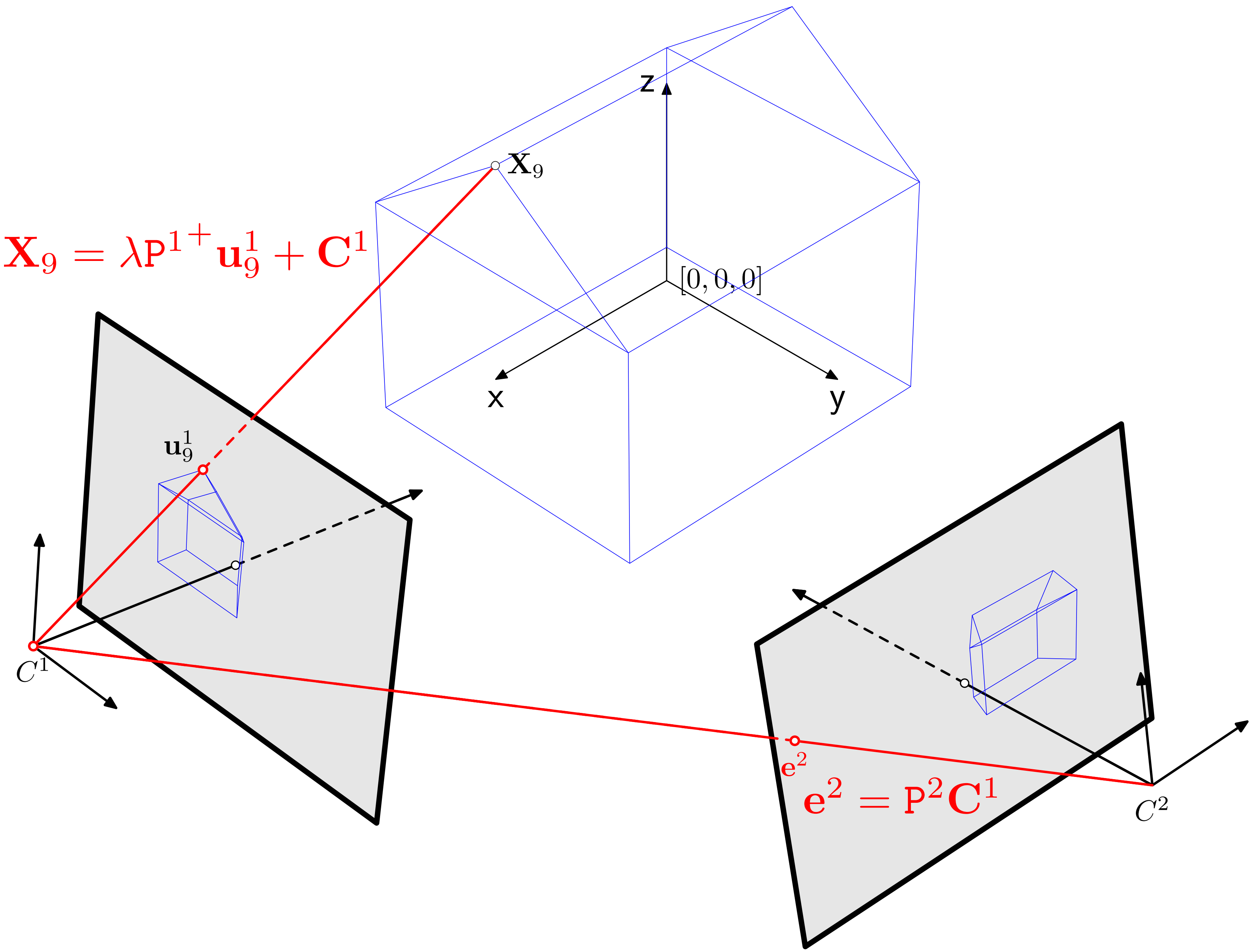
It can! (to both)

Can we find a relation between corresponding projections regardless of the scene structure?

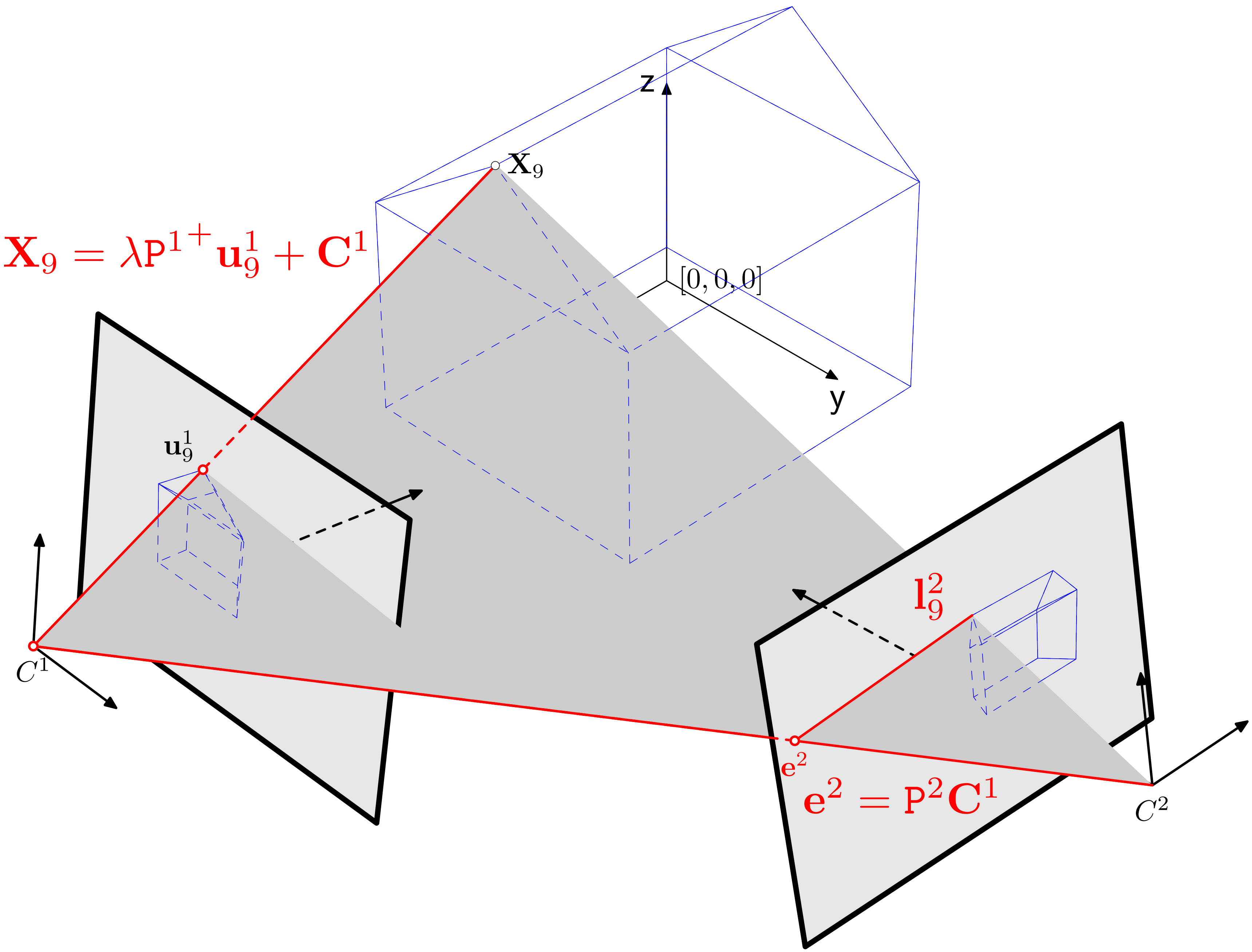
Back project the ray



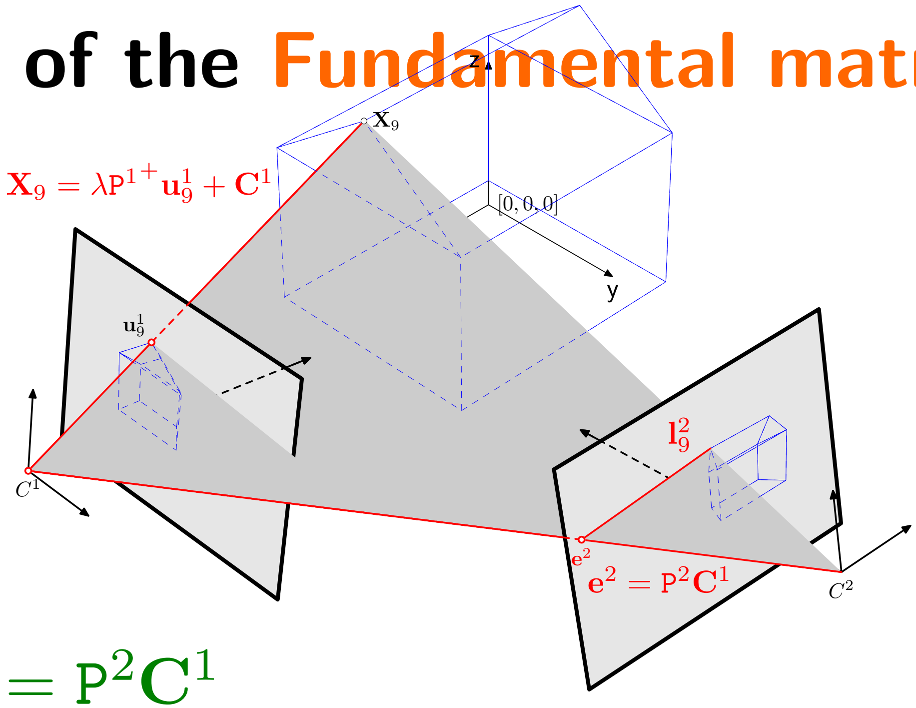
Project the camera center to the second image



The correponding projection must lie on a specific line



Derivation of the Fundamental matrix



We already know: $\mathbf{e}^2 = \mathtt{P}^2\mathbf{C}^1$

Projection to the camera 2: $\mathbf{u}_9^2 = \mathtt{P}^2(\lambda \mathtt{P}^{1+}\mathbf{u}_9^1 + \mathbf{C}^1)$

Line is a cross product of the points lying on it: $\mathbf{e}^2 \times \mathbf{u}_9^2 = \mathbf{l}_9^2$

Putting together: $\mathbf{e}^2 \times (\mathtt{P}^2\lambda\mathtt{P}^{1+}\mathbf{u}_9^1 + \mathtt{P}^2\mathbf{C}^1) = \mathbf{l}_9^2$

Clearly $\mathbf{e}^2 \times \mathtt{P}^2\mathbf{C}^1 = 0$, then: $\mathbf{e}^2 \times \lambda\mathtt{P}^2\mathtt{P}^{1+}\mathbf{u}_9^1 = \mathbf{l}_9^2$

But we also know ${\mathbf{l}_9^2}^\top \mathbf{u}_9^2 = 0$ since the point $\mathbf{u}_9^2$ must lie on the line $\mathbf{l}_9^2$.

Derivation of the Fundamental matrix, cont.

$\mathbf{e}^2 \times \lambda \mathtt{P}^2 \mathtt{P}^{1+} \mathbf{u}_9^1 = \mathbf{l}_9^2$

But we also know ${\mathbf{l}_9^2}^\top \mathbf{u}_9^2 = 0$ since the point $\mathbf{u}_9^2$ must lie on the line.

Introducing a small matrix trick $[\mathbf{e}]_\times = \begin{bmatrix} 0 & -e_3 & e_2 \\ e_3 & 0 & -e_1 \\ -e_2 & e_1 & 0 \end{bmatrix}$

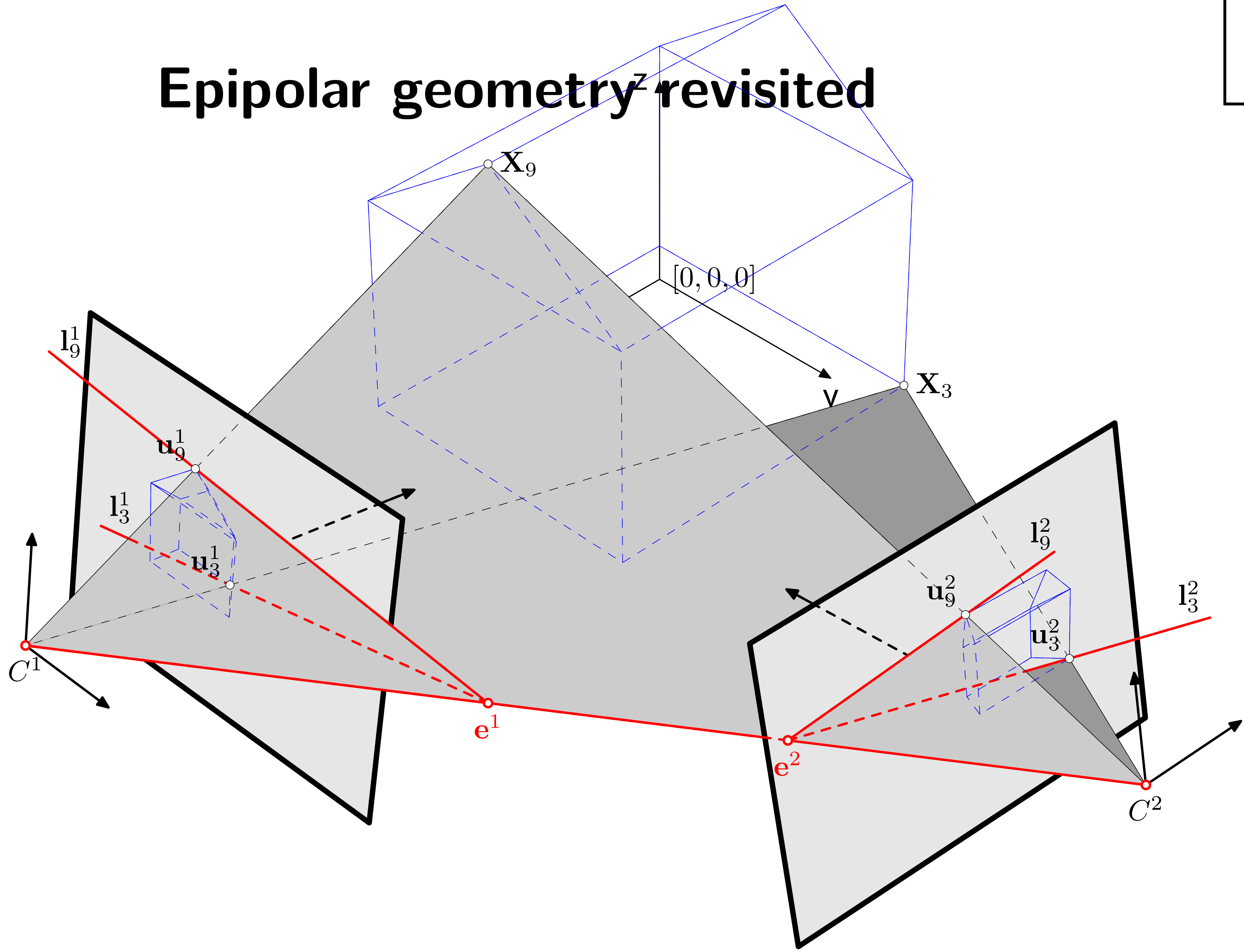
we may rewrite the cross product as a matrix multiplication
$\mathbf{l}_9^2 = \left([\mathbf{e}^2]_\times \lambda \mathtt{P}^2 \mathtt{P}^{1+}\right) \mathbf{u}_9^1$

Inserting into ${\mathbf{l}_9^2}^\top \mathbf{u}_9^2 = 0$ yields:

$${\mathbf{u}_9^1}^\top \underbrace{\left([\mathbf{e}^2]_\times \lambda \mathtt{P}^2 \mathtt{P}^{1+}\right)^\top}_{\mathtt{F}} \mathbf{u}_9^2 = 0$$

$$\boxed{{\mathbf{u}_9^2}^\top \mathbf{F} \mathbf{u}_9^1 = 0}$$

Epipolar geometry revisited

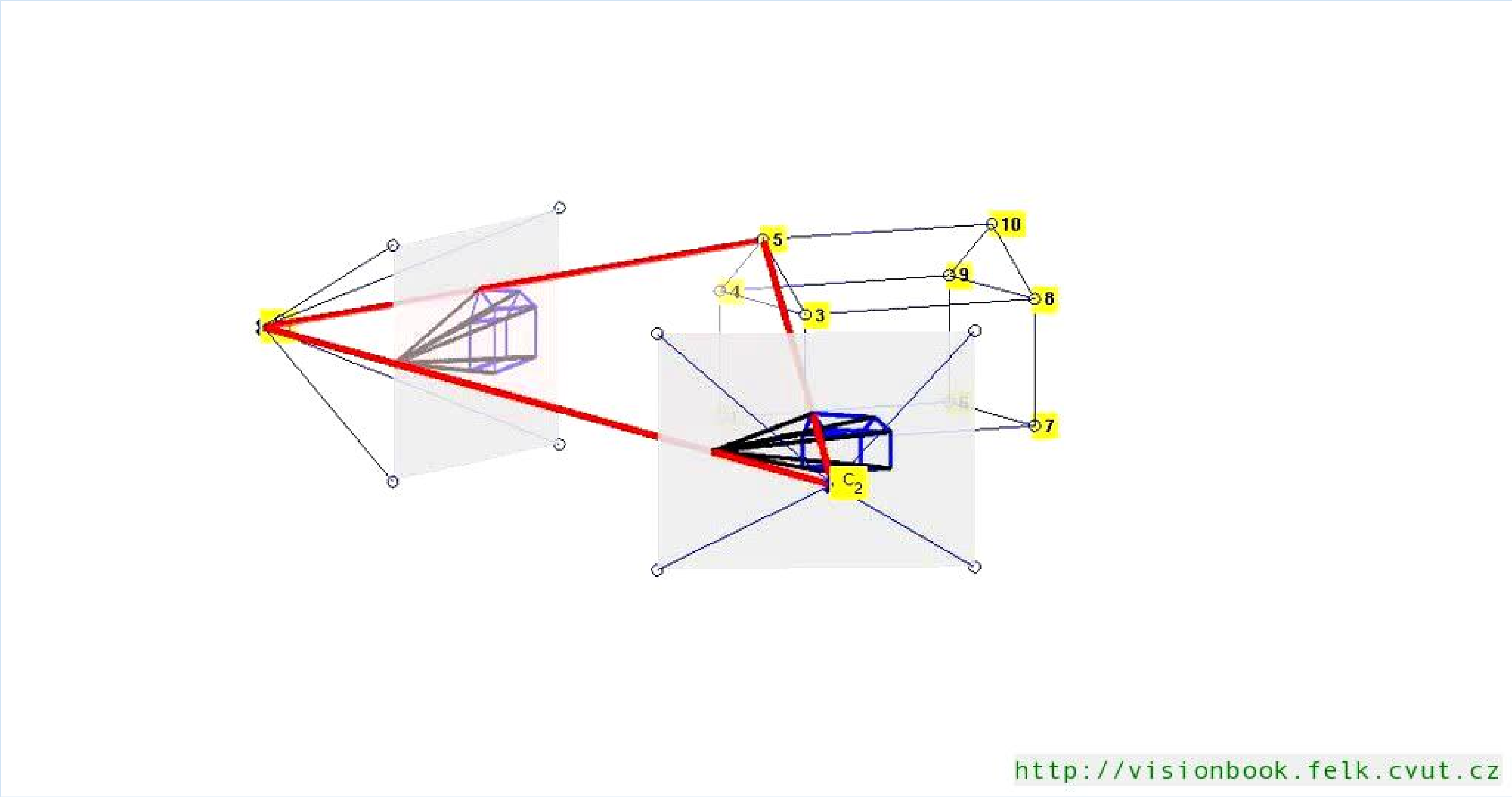


$\mathbf{u}_i^{2\top}\mathbf{F}\mathbf{u}_i^1 = 0$ holds for any corresponding pair $\mathbf{u}_i^1, \mathbf{u}_i^2$.

$\mathbf{F}$ does not depend on the scene structure, only on cameras.

All epipolar lines intersect in epipoles.

Epipolar geometry—overview



video: 3D sketch of Epipolar geometry

Epipolar geometry—what is it good for

Epipolar geometry—what is it good for

Epipolar geometry—what is it good for

Epipolar geometry—what is it good for

Motion and 3D structure is where?

Essential matrix

For the Fundamental matrix we derived

$$\mathbf{u}_i^{1\top} \underbrace{\left([\mathbf{e}^2]_\times \mathtt{P}^2 \mathtt{P}^{1+}\right)^\top}_{\mathtt{F}} \mathbf{u}_i^2 = 0$$

$\mathbf{u}$ denote point coordinates in pixels.

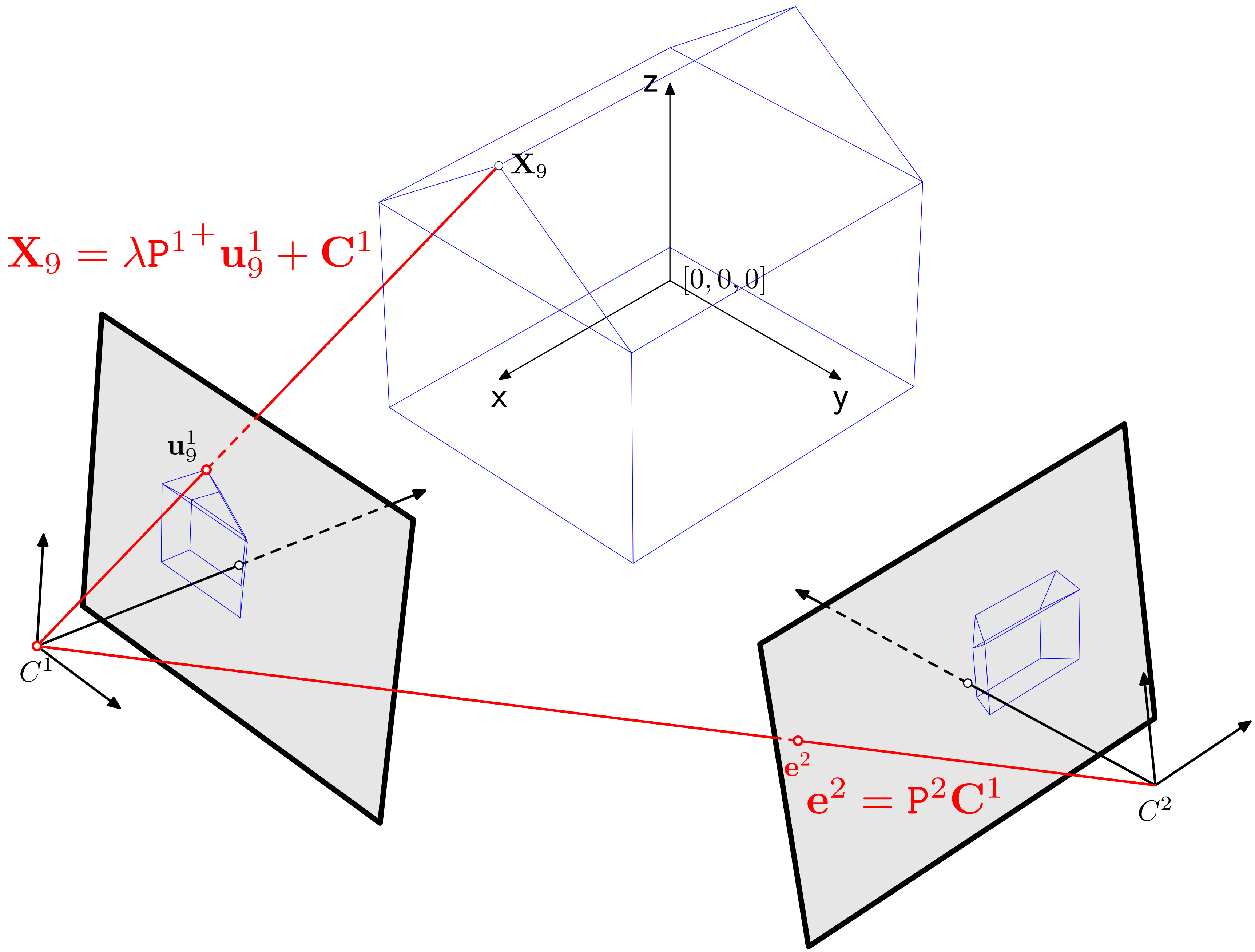
$$\mathbf{u}^1 = \mathtt{K}^1 \begin{bmatrix} \mathtt{R}^1 & \mathbf{t}^1 \end{bmatrix} \mathbf{X} \qquad \mathbf{u}^2 = \mathtt{K}^2 \begin{bmatrix} \mathtt{R}^2 & \mathbf{t}^2 \end{bmatrix} \mathbf{X}$$

Remind the normalized image coordinates $\mathbf{x} = \mathtt{K}^{-1}\mathbf{u}$. We can define normalized cameras $\mathbf{x} = \hat{\mathtt{P}}\mathbf{X}$ and insert the equation above.

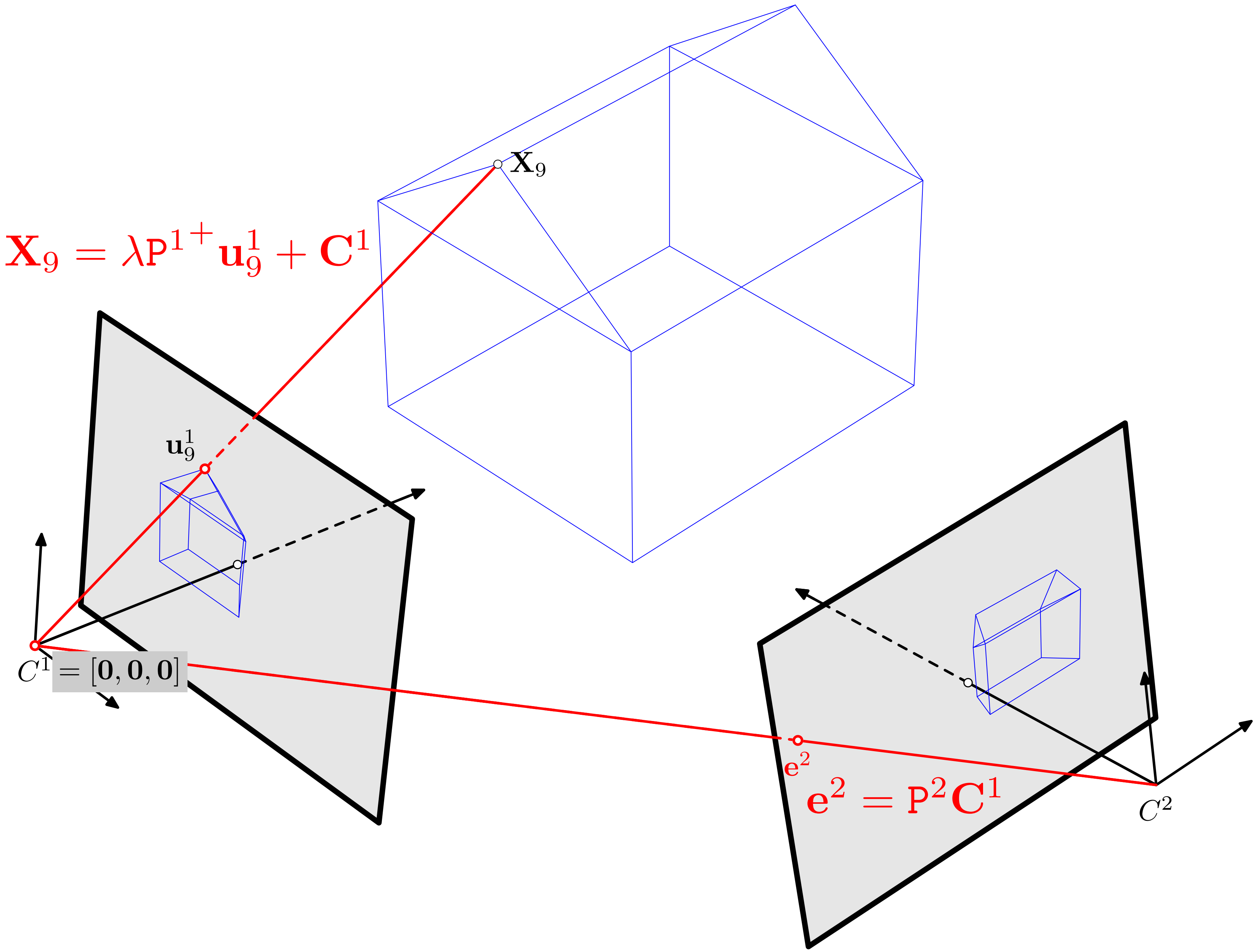
$$\mathbf{x}_i^{1\top} \underbrace{\left([\mathbf{x}_\mathbf{e}^2]_\times \hat{\mathtt{P}}^2 (\hat{\mathtt{P}}^1)^+\right)^\top}_{\mathtt{E}} \mathbf{x}_i^2 = 0$$

where $\mathtt{E}$ is the Essential matrix

Where to set the origin of the world?



Where to set the origin of the world?



What do we gain?

$$\mathbf{u}^1 = \mathtt{K}^1 \left[\begin{array}{cc} \mathtt{R}^1 & \mathbf{t}^1 \end{array} \right] \mathbf{X} \qquad \mathbf{u}^2 = \mathtt{K}^2 \left[\begin{array}{cc} \mathtt{R}^2 & \mathbf{t}^2 \end{array} \right] \mathbf{X}$$

What do we gain?

$$\mathbf{u}^1 = \mathtt{K}^1 \begin{bmatrix} \mathtt{R}^1 & \mathbf{t}^1 \end{bmatrix} \mathbf{X} \qquad \mathbf{u}^2 = \mathtt{K}^2 \begin{bmatrix} \mathtt{R}^2 & \mathbf{t}^2 \end{bmatrix} \mathbf{X}$$

$$\mathbf{u}^1 = \mathtt{K}^1 \begin{bmatrix} \mathtt{I} & \mathbf{0} \end{bmatrix} \mathbf{X} \qquad \mathbf{u}^2 = \mathtt{K}^2 \begin{bmatrix} \mathtt{R} & \mathbf{t} \end{bmatrix} \mathbf{X}$$

What do we gain?

$$\mathbf{u}^1 = \mathtt{K}^1 \left[\begin{array}{cc} \mathtt{R}^1 & \mathbf{t}^1 \end{array} \right] \mathbf{X} \qquad \mathbf{u}^2 = \mathtt{K}^2 \left[\begin{array}{cc} \mathtt{R}^2 & \mathbf{t}^2 \end{array} \right] \mathbf{X}$$

$$\mathbf{u}^1 = \mathtt{K}^1 \left[\begin{array}{cc} \mathtt{I} & \mathbf{0} \end{array} \right] \mathbf{X} \qquad \mathbf{u}^2 = \mathtt{K}^2 \left[\begin{array}{cc} \mathtt{R} & \mathbf{t} \end{array} \right] \mathbf{X}$$

Few variables vanished, $\mathtt{R}$ and $\mathbf{t}$ now denote motion of the camera. One can call it camera displacement, or ego-motion.

Estimation of $\mathtt{R}$ and $\mathbf{t}$ is often called camera tracking.

Essential matrix — cont'd

$$\begin{aligned} \mathtt{E} &= [\mathbf{x}_\mathbf{e}^2]_\times \hat{\mathtt{P}}^2 (\hat{\mathtt{P}}^1)^+ \\ &= [\mathbf{x}_\mathbf{e}^2]_\times \begin{bmatrix} \mathtt{R} & \mathbf{t} \end{bmatrix} \begin{bmatrix} \mathtt{I} & \mathbf{0} \end{bmatrix}^+ \\ &= [\mathbf{x}_\mathbf{e}^2]_\times \mathtt{R} \end{aligned}$$

$$\begin{aligned} \mathbf{x}_\mathbf{e}^2 &= \hat{\mathtt{P}}^2 \mathbf{C}^1 \\ &= \begin{bmatrix} \mathtt{R} & \mathbf{t} \end{bmatrix} \begin{bmatrix} \mathbf{0} \\ 1 \end{bmatrix} \\ &= \mathbf{t} \end{aligned}$$

$$\boxed{\mathtt{E} = [\mathbf{t}]_\times \mathtt{R}}$$

$\mathtt{E}$ comprises the motion between cameras!

after simple manipulation, we see $\mathtt{E} = \mathtt{K}^{2\top} \mathtt{F} \mathtt{K}^1$

Decomposition of the $\mathtt{E}$

Suppose $\mathtt{E} = \mathtt{U}\,\mathrm{diag}(1,1,0)\,\mathtt{V}^\top$ and

$$\mathtt{W} = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \text{and} \quad \mathtt{Z} = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

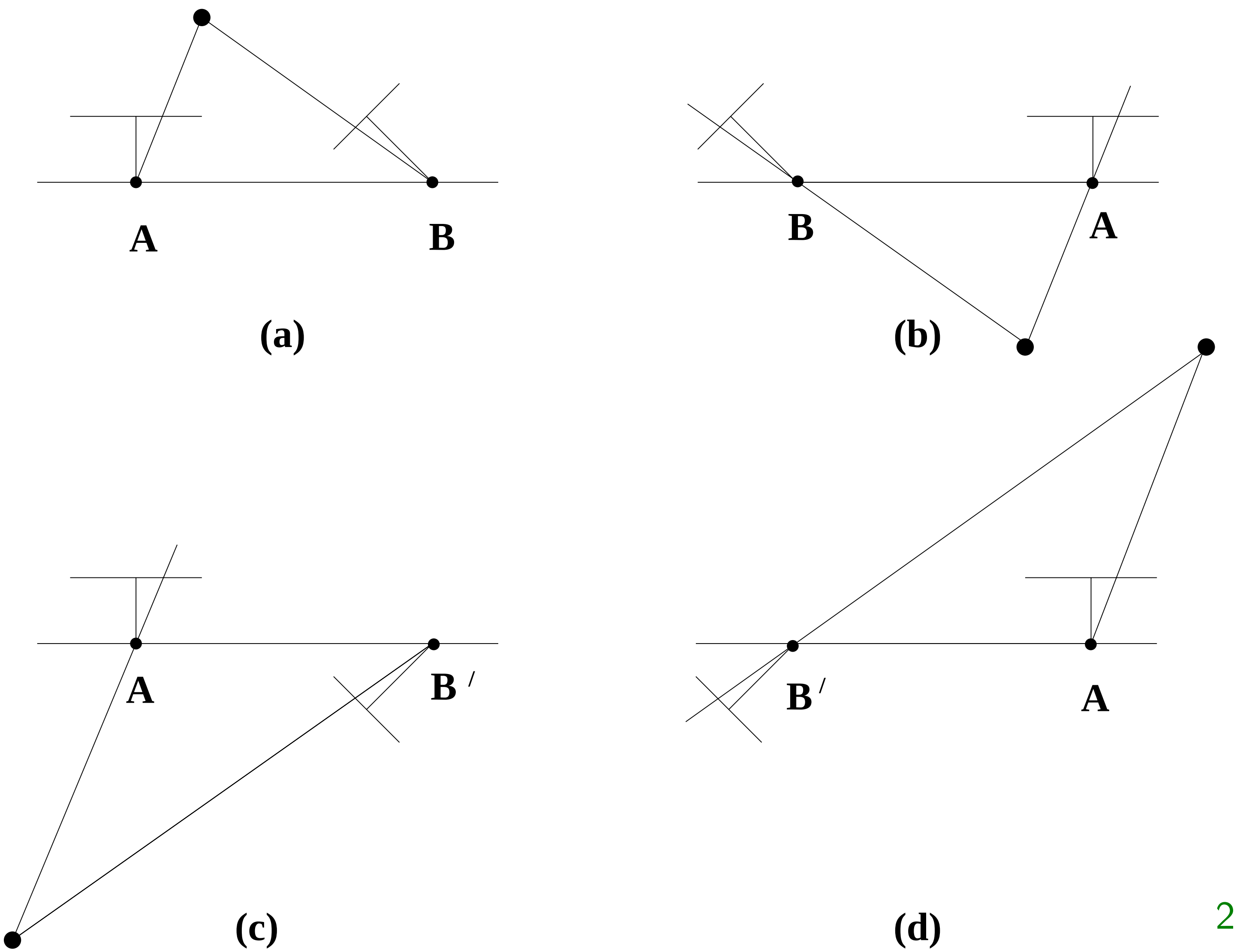
then, for a given $\mathtt{E}$ and $\hat{\mathtt{P}}^1 = [\mathtt{I}|0]$, there are four possible solutions for $\hat{\mathtt{P}}^2$

$$\hat{\mathtt{P}}^2 = [\mathtt{UVW}^\top| + \mathbf{u}_3] \text{ or } [\mathtt{UVW}^\top| - \mathbf{u}_3] \text{ or } [\mathtt{UV}^\top\mathtt{W}^\top| + \mathbf{u}_3] \text{ or } [\mathtt{UV}^\top\mathtt{W}^\top| - \mathbf{u}_3]$$

More details in [3][1].

[1] The relevant chapter 9, is available on the web, http://www.robots.ox.ac.uk/~vgg/hzbook/hzbook2/HZepipolar.pdf, see pages 20-21

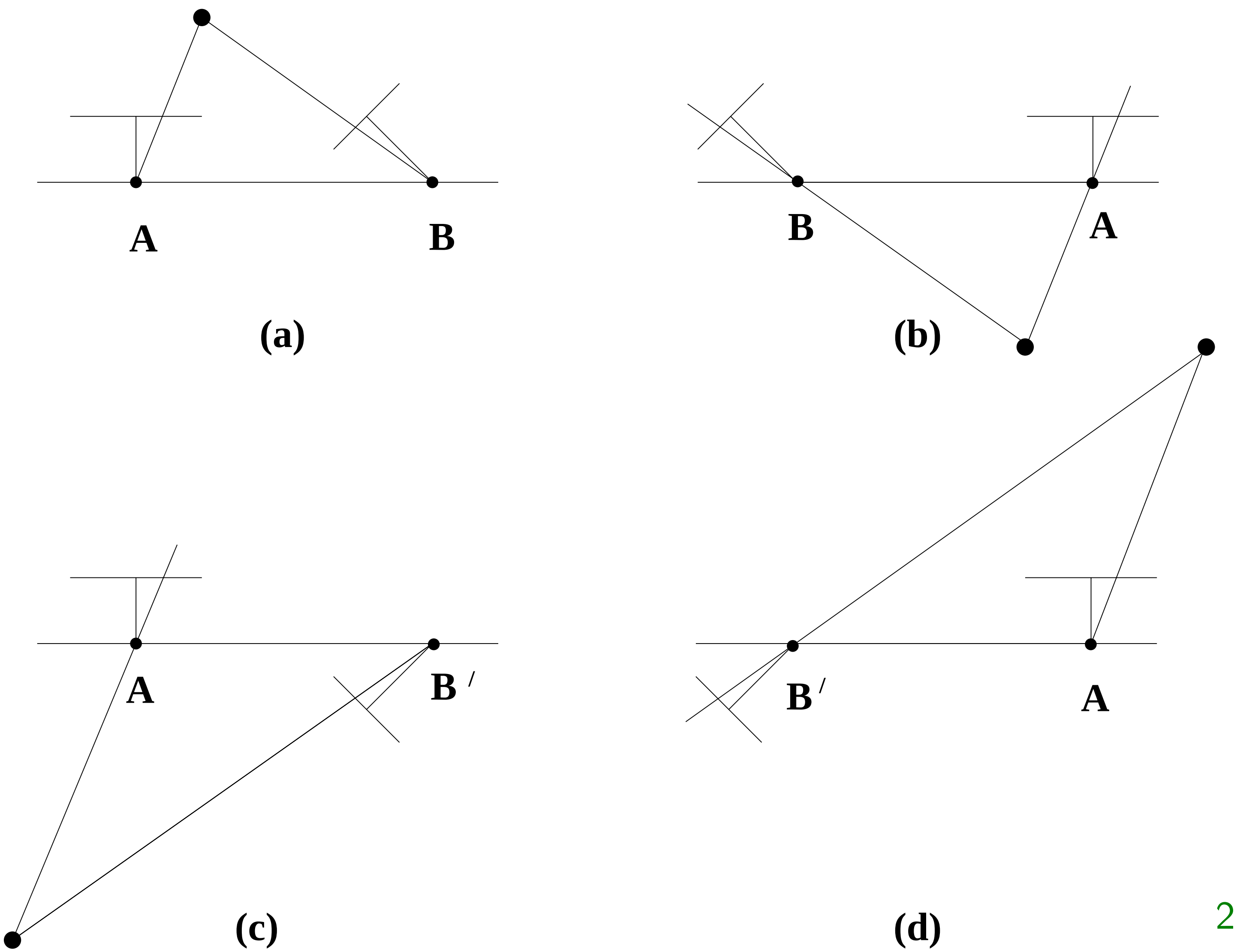
Fourfold ambiguity of the $\mathbb{E}$ decomposition



2

[2] Sketch from [2].

Fourfold ambiguity of the $\mathtt{E}$ decomposition



[2]

Which one?

[2] Sketch from [2].

3D scene reconstruction—Linear method

A scene point $\mathbf{X}$ is observed by two cameras $\mathtt{P}^1$ and $\mathtt{P}^2$. Assume we know its projections $[u^j, v^j]^\top$

$\mathbf{u} = \mathtt{P}\mathbf{X}$, $u = \frac{\mathbf{p}_1^\top \mathbf{X}}{\mathbf{p}_3^\top \mathbf{X}}$, $u(\mathbf{p}_3^\top \mathbf{X}) - \mathbf{p}_1^\top \mathbf{X} = 0$, the same derivation for v and for both cameras:

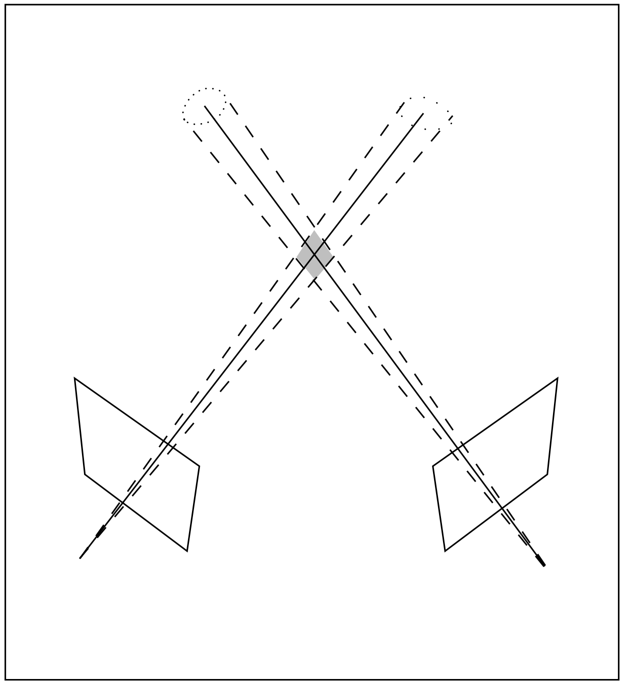
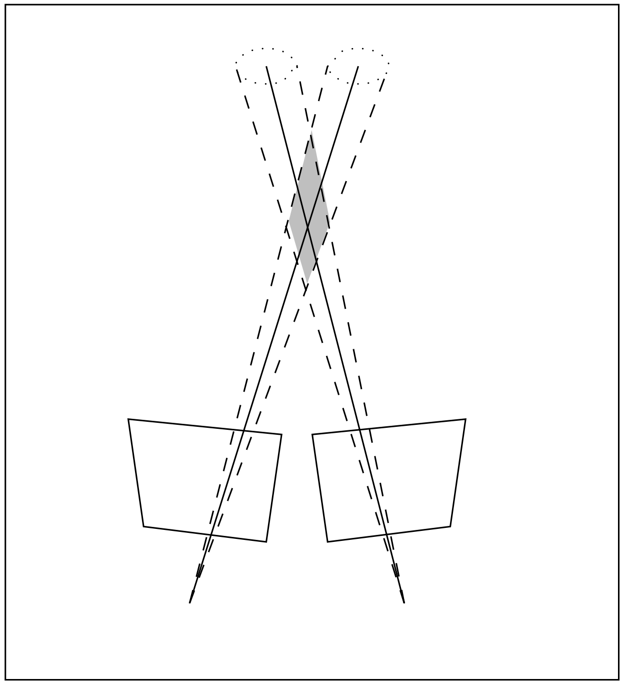
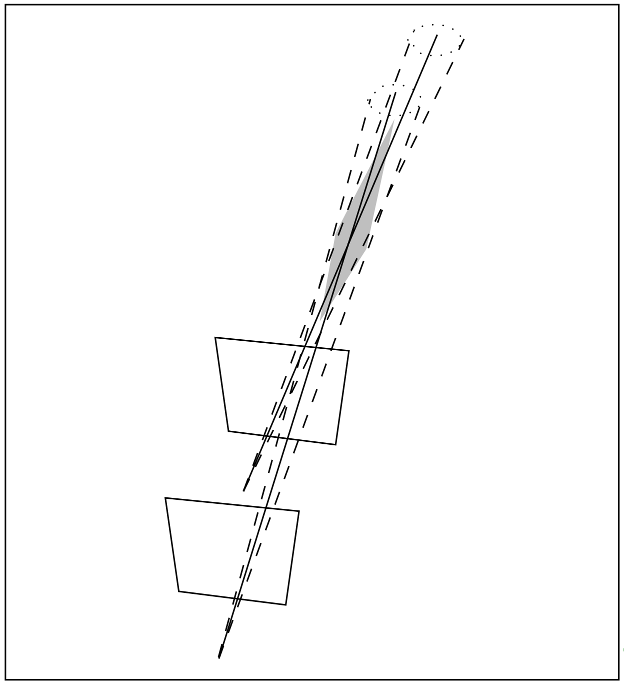
$$\begin{bmatrix} u^1 \mathbf{p}_3^{1\top} - \mathbf{p}_1^{1\top} \\ v^1 \mathbf{p}_3^{1\top} - \mathbf{p}_2^{1\top} \\ u^2 \mathbf{p}_3^{2\top} - \mathbf{p}_1^{2\top} \\ v^2 \mathbf{p}_3^{2\top} - \mathbf{p}_2^{2\top} \end{bmatrix} \begin{bmatrix} \mathbf{X} \end{bmatrix} = \begin{bmatrix} \mathbf{0} \end{bmatrix}$$

Set of linear homogeneous equations. A standard LSQ solution[3] may be used.

Not an optimal solution. It minimizes algebraic not geometric error. More methods can be found in [3, Chapter 12]

[3] http://cmp.felk.cvut.cz/cmp/courses/Y33ROV/Y33ROV_ZS20092010/Lectures/Supporting/constrained_lsq.pdf

Errors in reconstruction



[4]

- the bigger angle between rays the better reconstruction, however ...
- also the more difficult image matching

[4] Sketch borrowed from [2]

Problems with image matching

Good for matching, bad for reconstruction

Problems with image matching

Good for recontruction, bad for matching

Estimation of $\mathtt{F}$ or $\mathtt{E}$ from corresponding point pairs

$$\mathbf{u}_i^{2\top} \mathtt{F} \mathbf{u}_i^1 = 0$$

for any pair of matching points. Each matching pair gives one linear equation

$$u^2 u^1 f_{11} + u^2 v^1 f_{12} + u^2 f_{13} \ldots = 0$$

which may be rewritten an a vector inner product

$$[u^2u^1, u^2v^1, u^2, v^2u^1, v^2v^1, v^2, u^1, v^1, 1]\mathbf{f} = 0$$

A set of n pairs forms a set of linear equations

$$\mathtt{A}\mathbf{f} = \begin{bmatrix} u_1^2 u_1^1 & u_1^2 v_1^1 & u_1^2 & v_1^2 u_1^1 & v_1^2 v_1^1 & v_1^2 & u_1^1 & v_1^1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ u_n^2 u_n^1 & u_n^2 v_n^1 & u_n^2 & v_n^2 u_n^1 & v_n^2 v_n^1 & v_n^2 & u_n^1 & v_n^1 & 1 \end{bmatrix} \mathbf{f} = \mathbf{0}$$

Estimation of $\mathtt{F}$—normalized 8-point algorithm

Solution of

$$\mathtt{A}\mathbf{f} = \begin{bmatrix} u_1^2 u_1^1 & u_1^2 v_1^1 & u_1^2 & v_1^2 u_1^1 & v_1^2 v_1^1 & v_1^2 & u_1^1 & v_1^1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ u_n^2 u_n^1 & u_n^2 v_n^1 & u_n^2 & v_n^2 u_n^1 & v_n^2 v_n^1 & v_n^2 & u_n^1 & v_n^1 & 1 \end{bmatrix} \mathbf{f} = \mathbf{0}$$

is a standard LSQ solution[5]

Point normalization

Consider a point pair $\mathbf{u}^1 = [150, 250, 1]^\top, \mathbf{u}^2 = [250, 350, 1]^\top$. It is clear that row elements in $\mathtt{A}$ are unbalanced.

$$\mathbf{a}^\top = [10^6, 10^6, 10^3, 10^6, 10^6, 10^3, 10^3, 10^3, 10^0]$$

This influences the numerical stability. Solution: normalization of the point coordinates before computation.

[5] `http://cmp.felk.cvut.cz/cmp/courses/Y33ROV/Y33ROV_ZS20092010/Lectures/Supporting/constrained_lsq.pdf`

Estimation of $\mathtt{F}$—normalized 8-point algorithm

Transform the coordinates of points so that the centroid is at the origin of coordinates nad RMS distance is equal to $\sqrt{2}$.

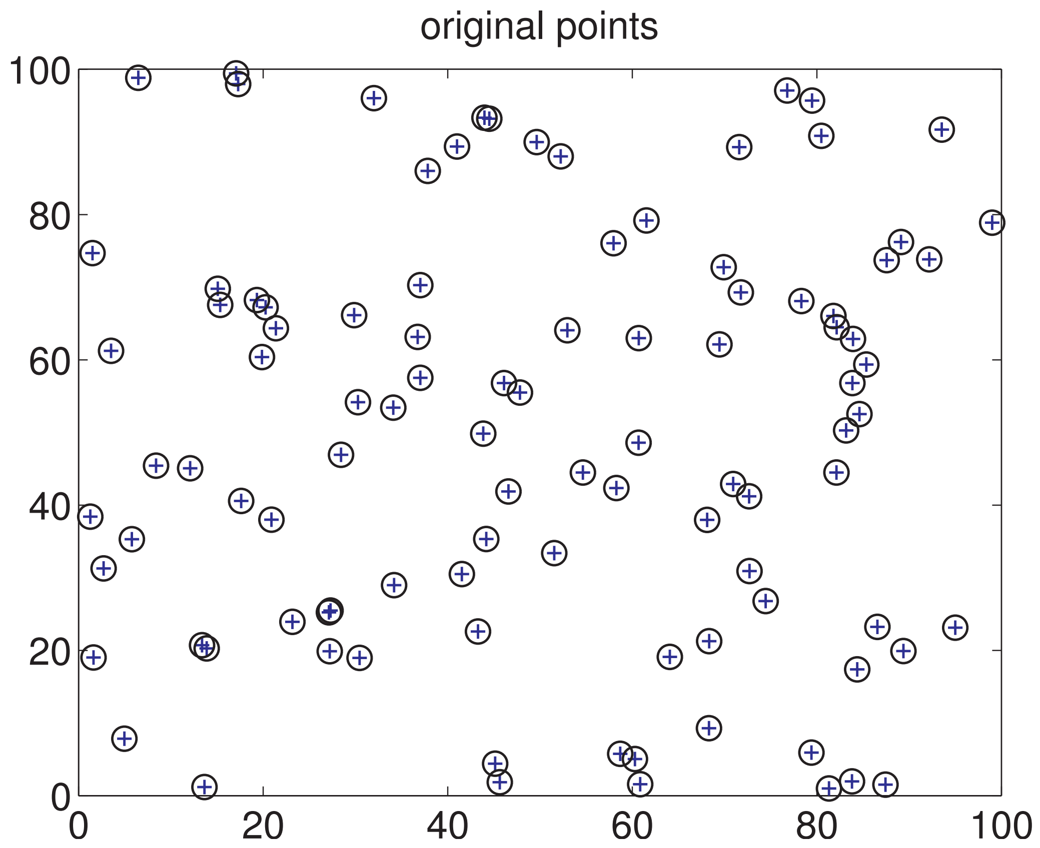
$\hat{\mathbf{u}}^1 = \mathtt{T}^1\mathbf{u}^1$ and $\hat{\mathbf{u}}^2 = \mathtt{T}^2\mathbf{u}^2$, where $\mathtt{T}^i$ are 3×3 normalizing matrices including translation nad scaling.

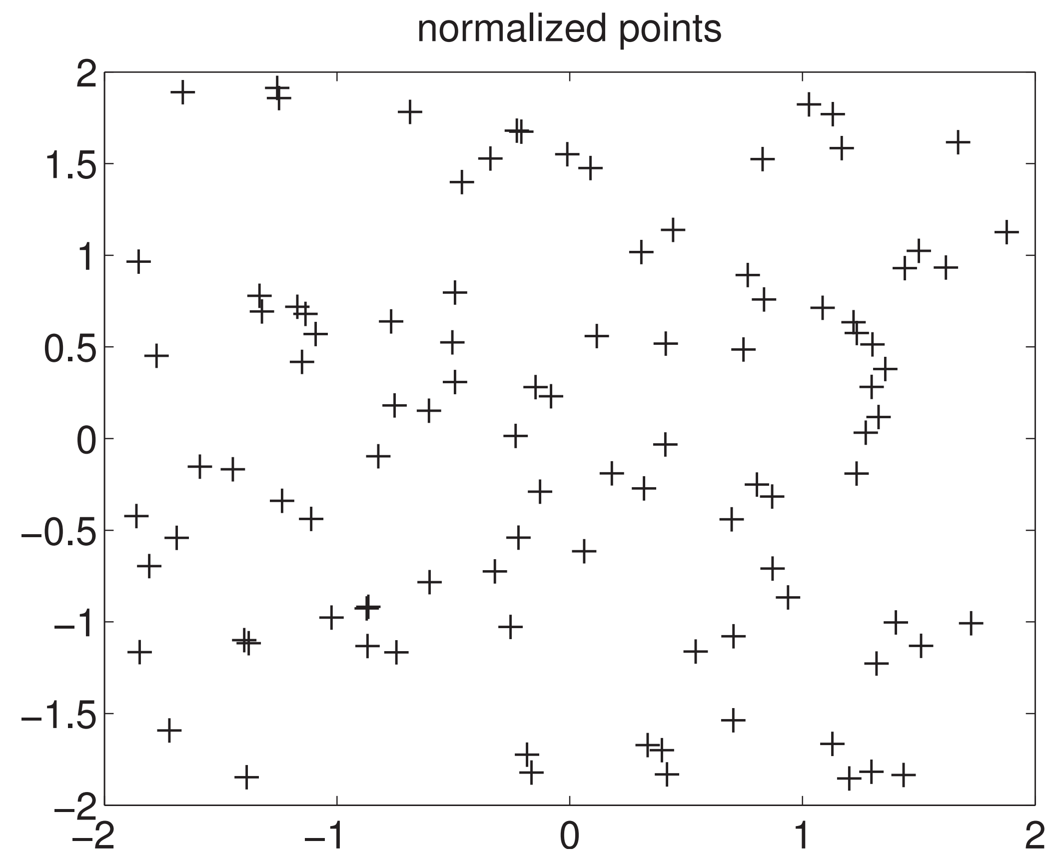
Compute $\hat{\mathtt{F}}$ by using the standard LSQ method, $\hat{\mathbf{u}}^{2\top}\hat{\mathtt{F}}\hat{\mathbf{u}}^1 = 0$. Denormalize the solution $\mathtt{F} = \mathtt{T}^{2\top}\hat{\mathtt{F}}\mathtt{T}^1$

Historical remarks

The linear algorithm for estimation epipolar geometry (calibrated case—essential matrix) was suggest in [5]. The normalization for the uncalibrated case (fundamental matrix) was introduced in [4].

Point normalization



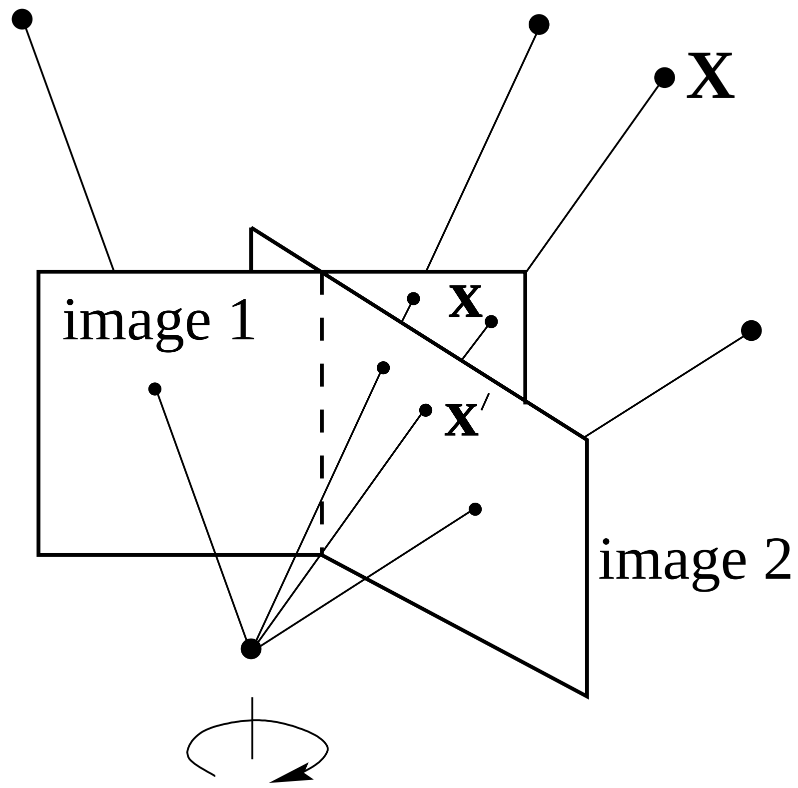


Zero motion

we derived

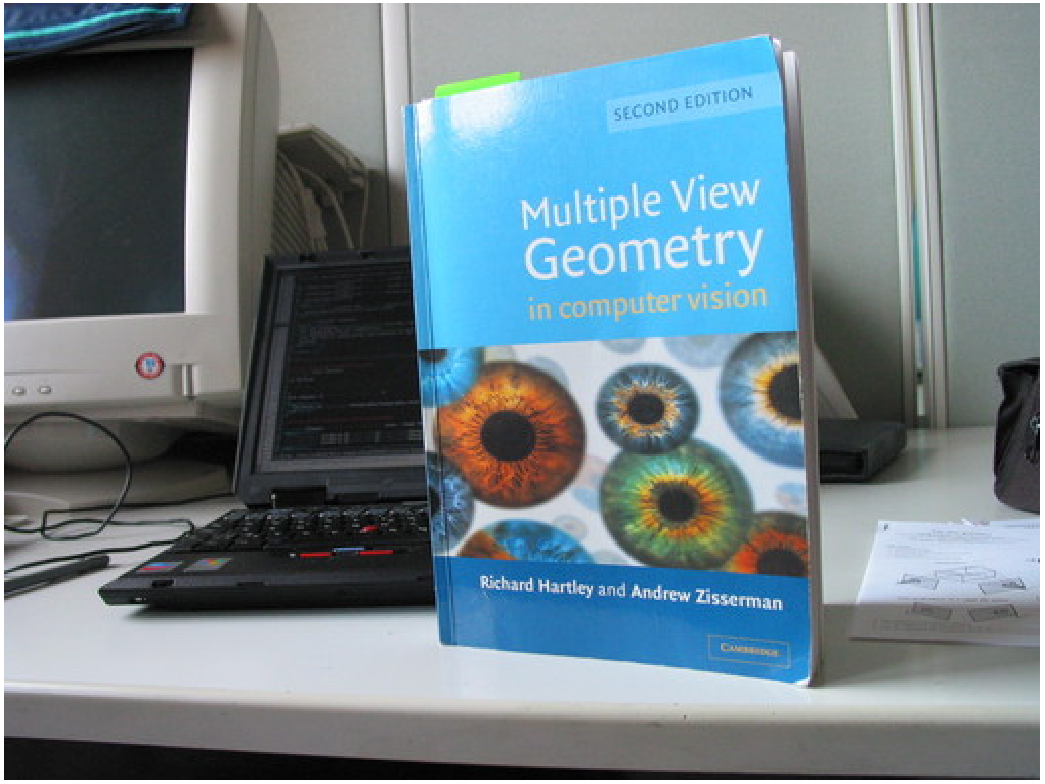
$$E = [\mathbf{t}]_\times R$$

what happens if $\mathbf{t} = \mathbf{0}$?

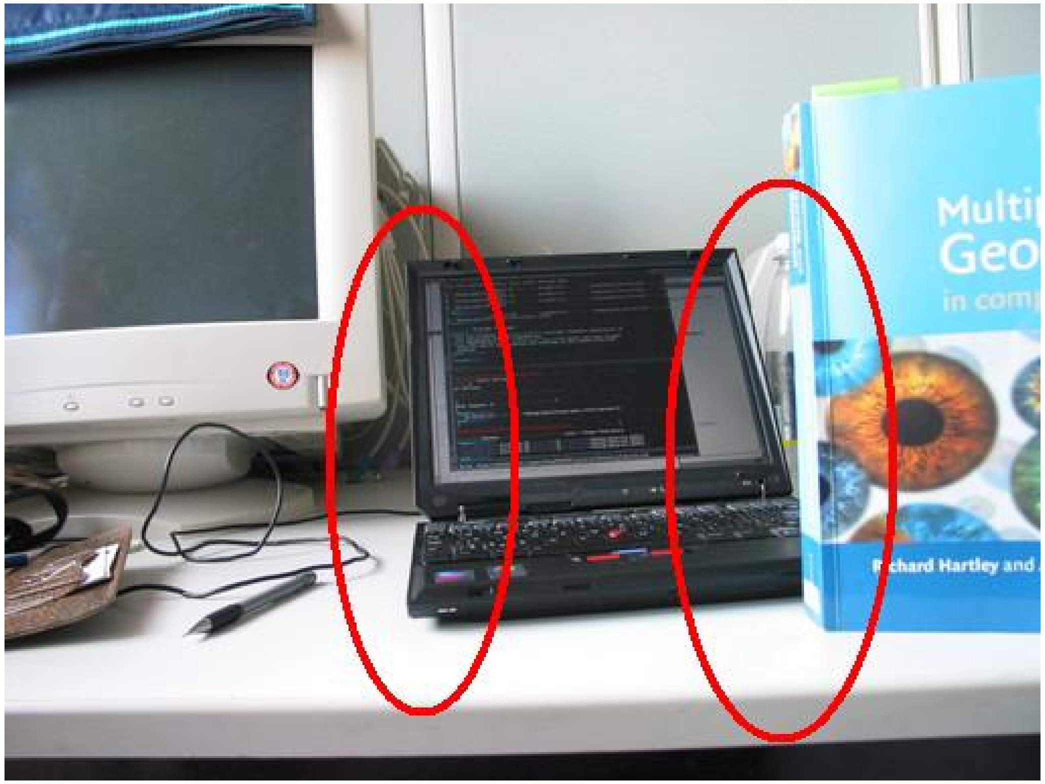


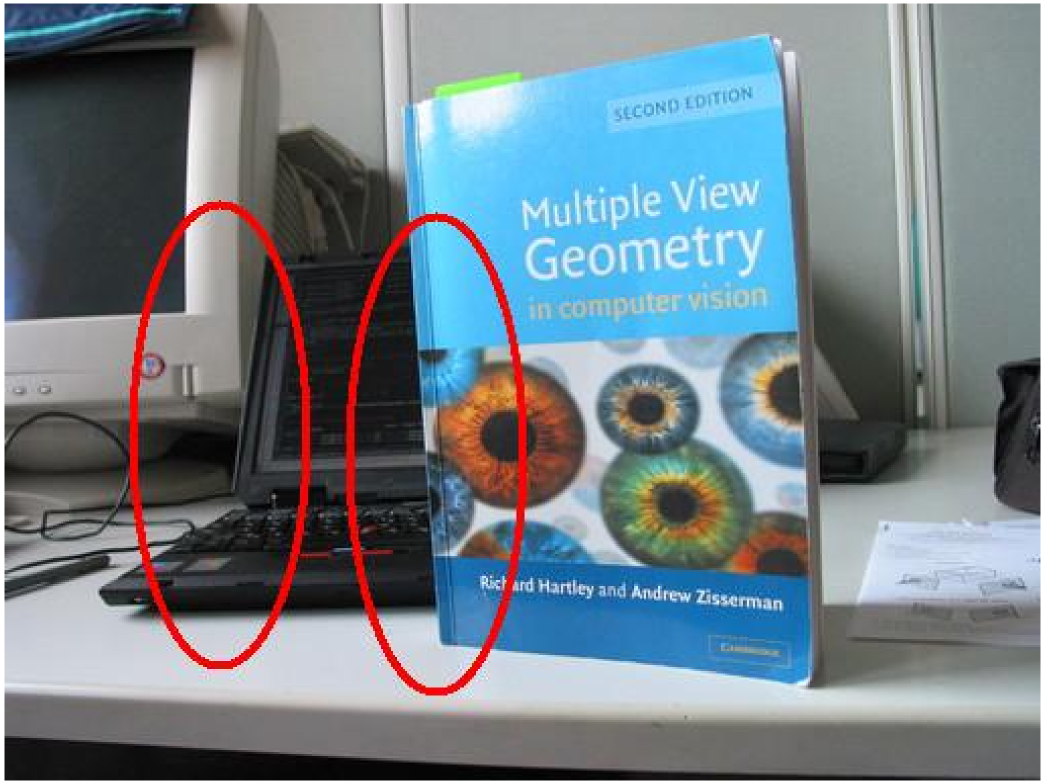
Common $t = 0$ case—Image Panoramas

What are the differences in images
general motion



What are the differences in images
general motion

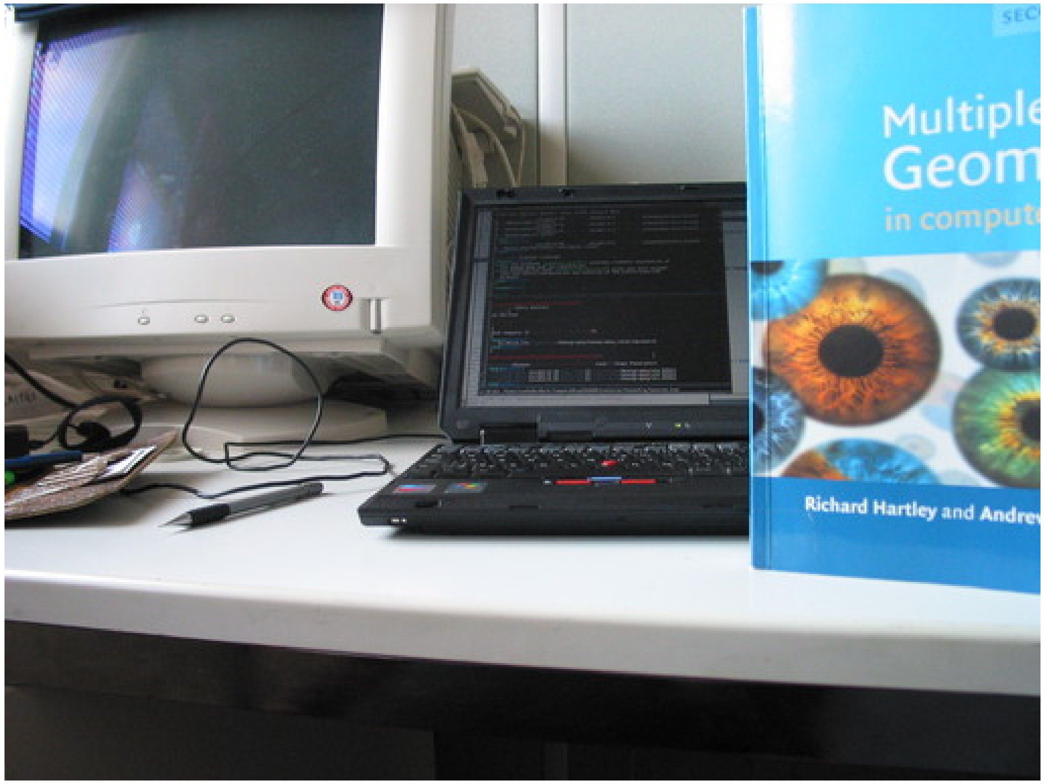


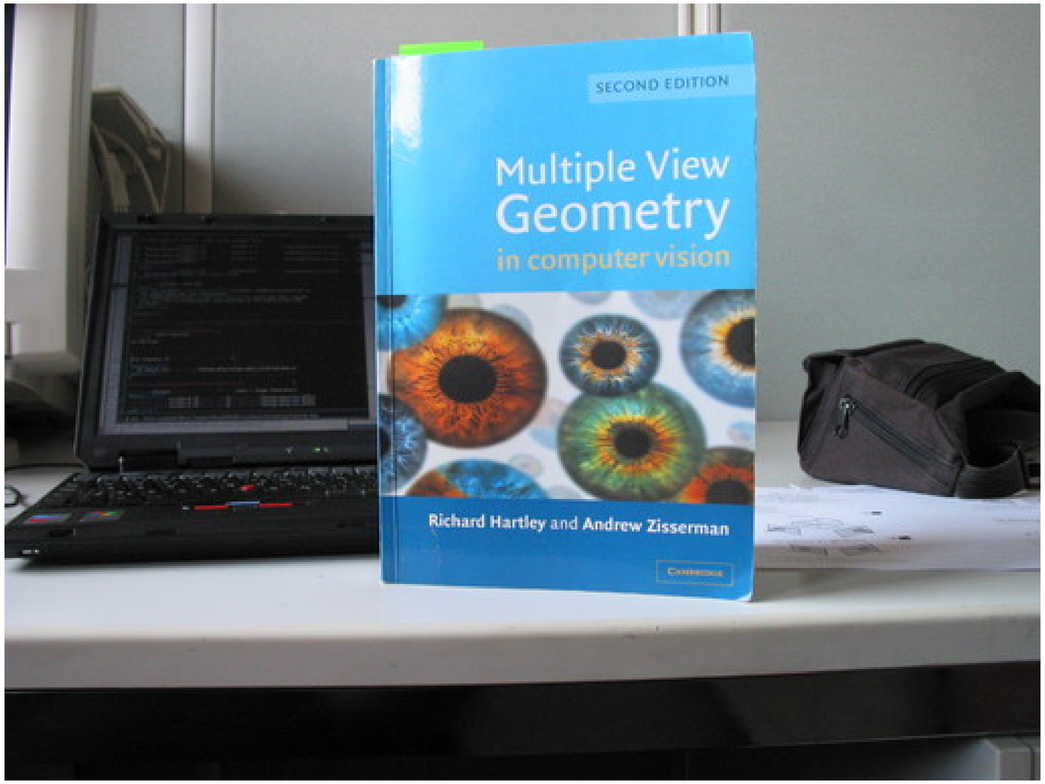


- objects in different depths make occlusions
- the mapping is certainly not 1:1

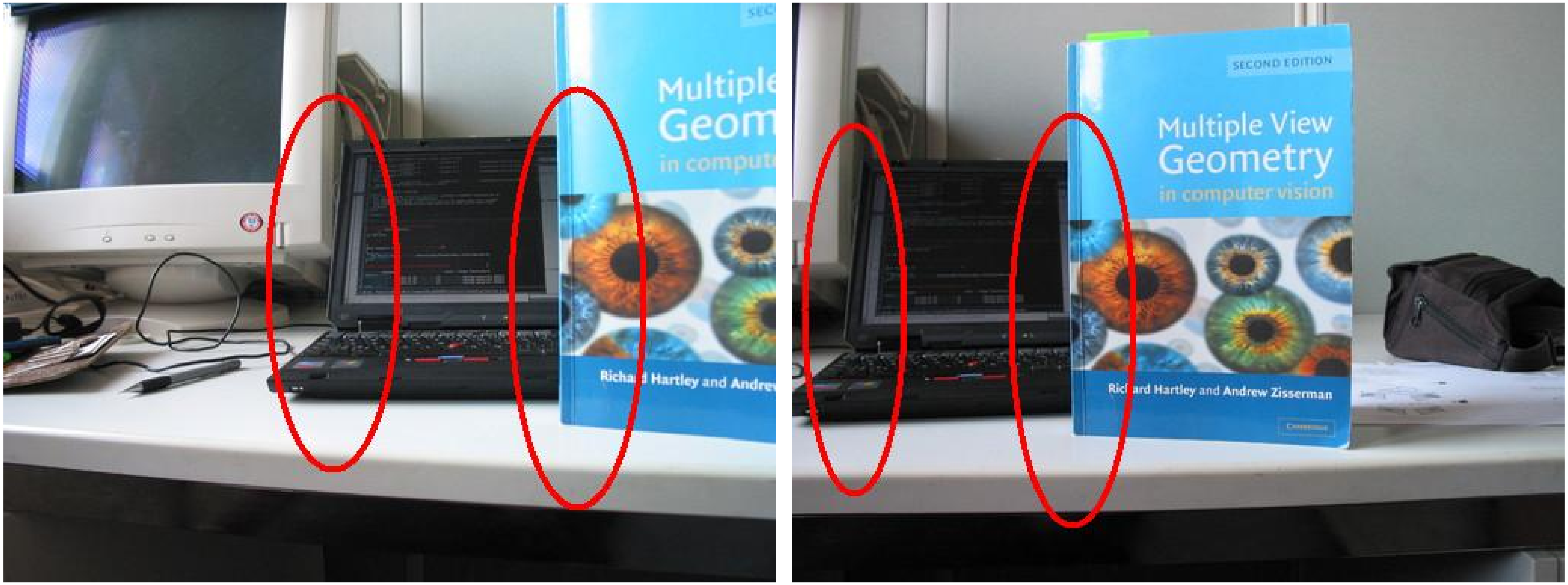
What are the differences in images

rotation



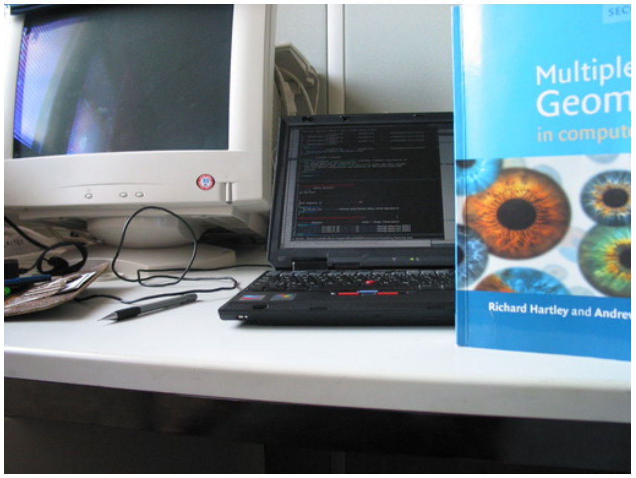


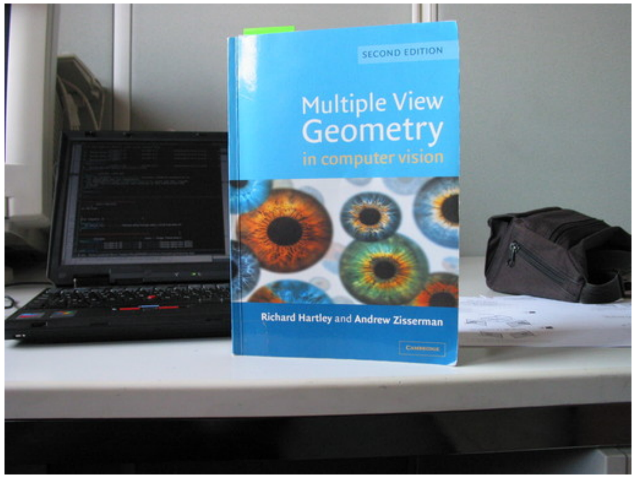
What are the differences in images
rotation

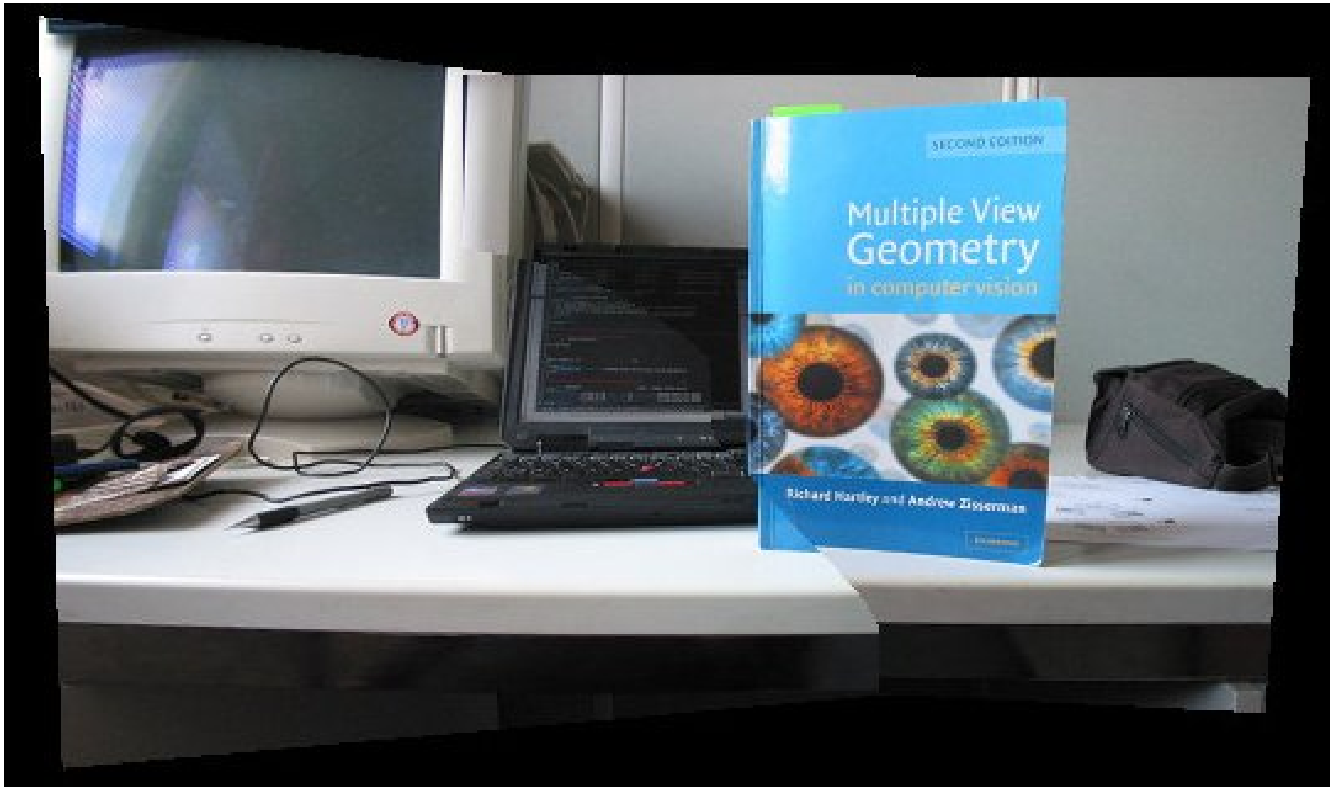


- no occlusions
- the mapping may be 1:1

Mapping between images







References

The book [3] is the ultimate reference. It is a must read for anyone wanting use cameras for 3D computing.

Details about matrix decompositions used throughout the lecture can be found at [1]

[1] Gene H. Golub and Charles F. Van Loan. Matrix Computation. Johns Hopkins Studies in the Mathematical Sciences. Johns Hopkins University Press, Baltimore, USA, 3rd edition, 1996.

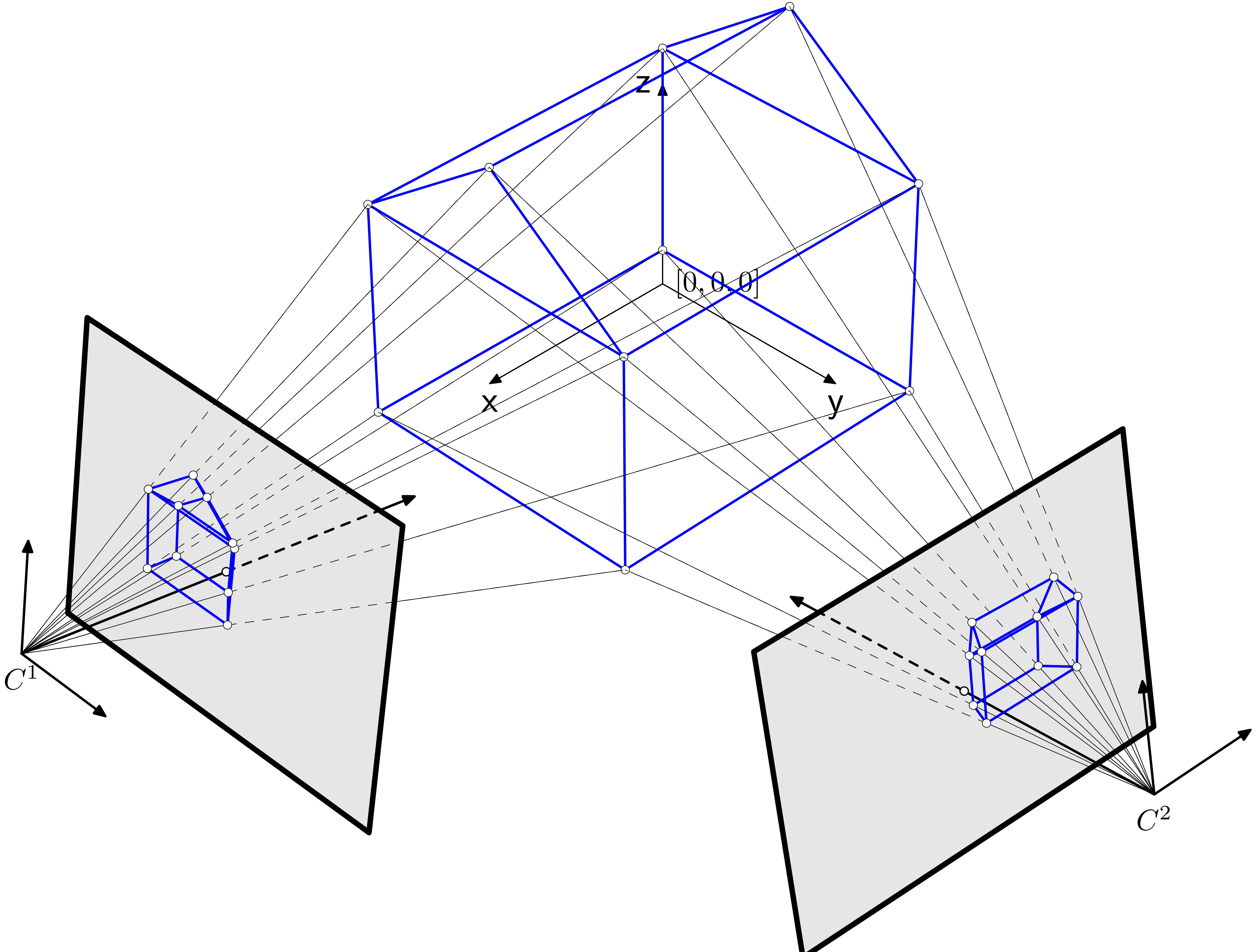
[2] R. Hartley and A. Zisserman. Multiple View Geometry in Computer Vision. Cambridge University Press, Cambridge, UK, 2000. On-line resources at: `http://www.robots.ox.ac.uk/~vgg/hzbook/hzbook1.html`.

[3] Richard Hartley and Andrew Zisserman. Multiple view geometry in computer vision. Cambridge University, Cambridge, 2nd edition, 2003.

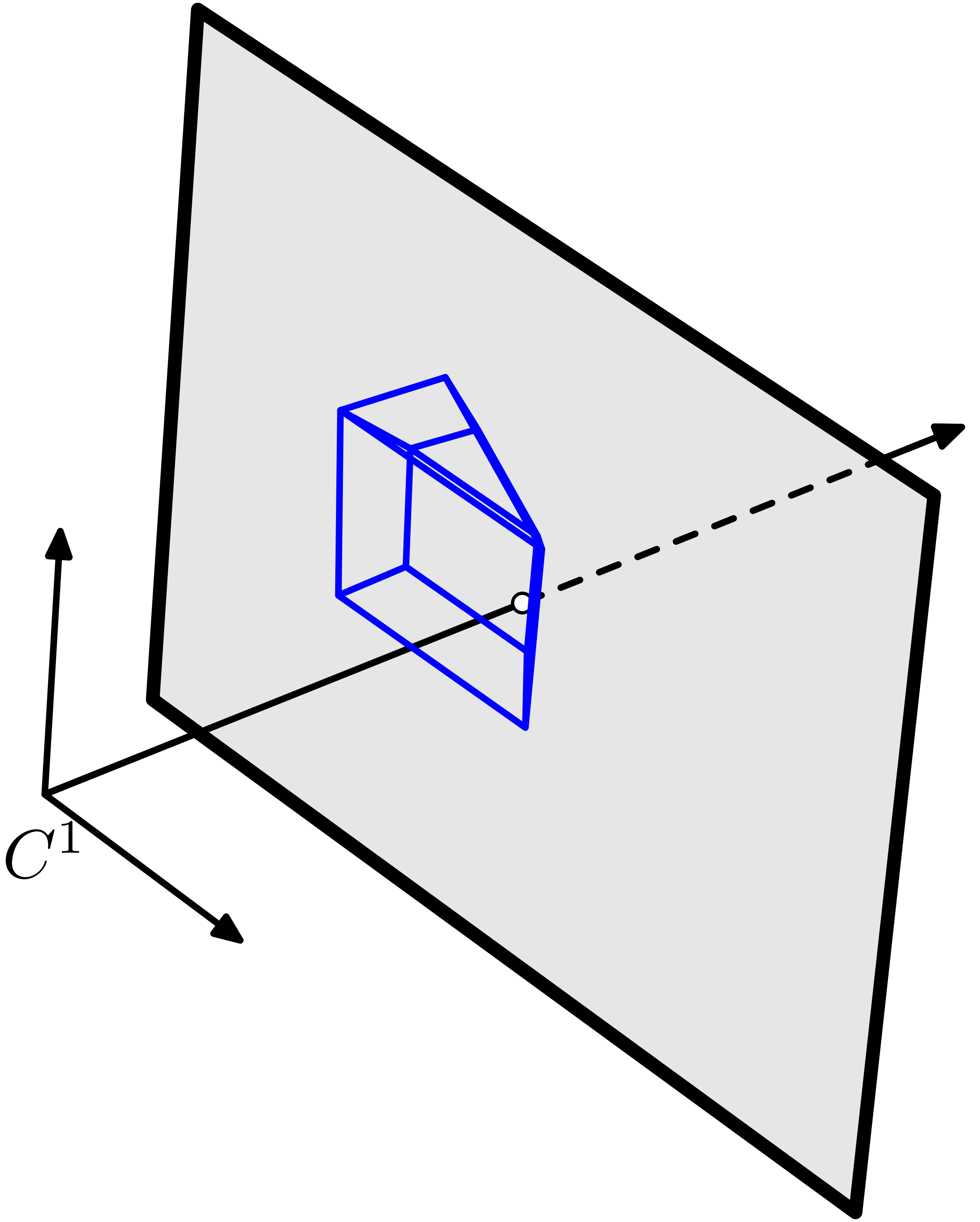
[4] Richard I. Hartley. In defense of the eight-point algorithm. IEEE Transaction on Pattern Analysis and Machine Intelligence, 19(6):580–593, June 1997.

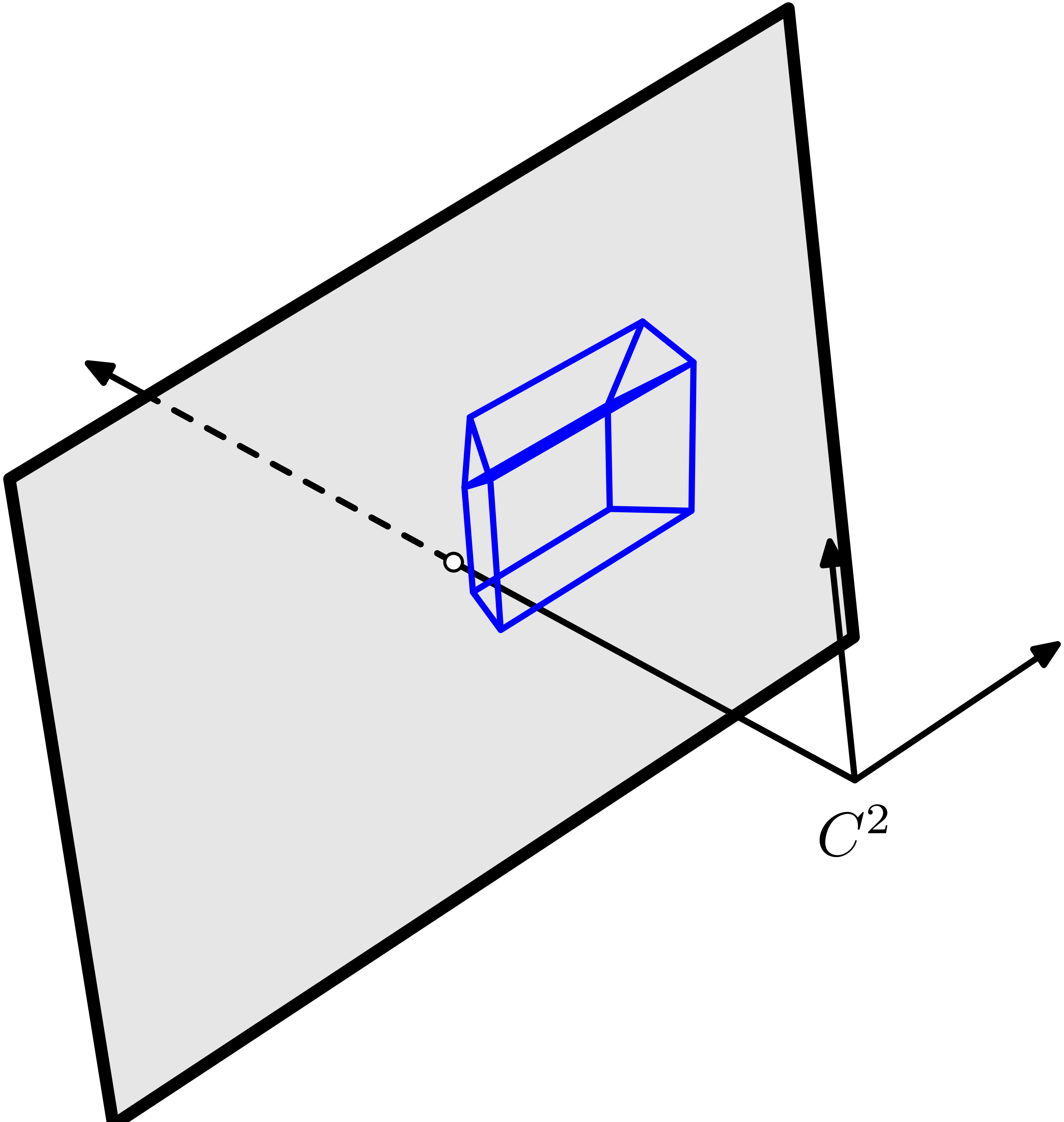
[5] H.C. Longuett-Higgins. A computer algorithm for reconstruction a scene from two projections. Nature, 293:133–135, 1981.

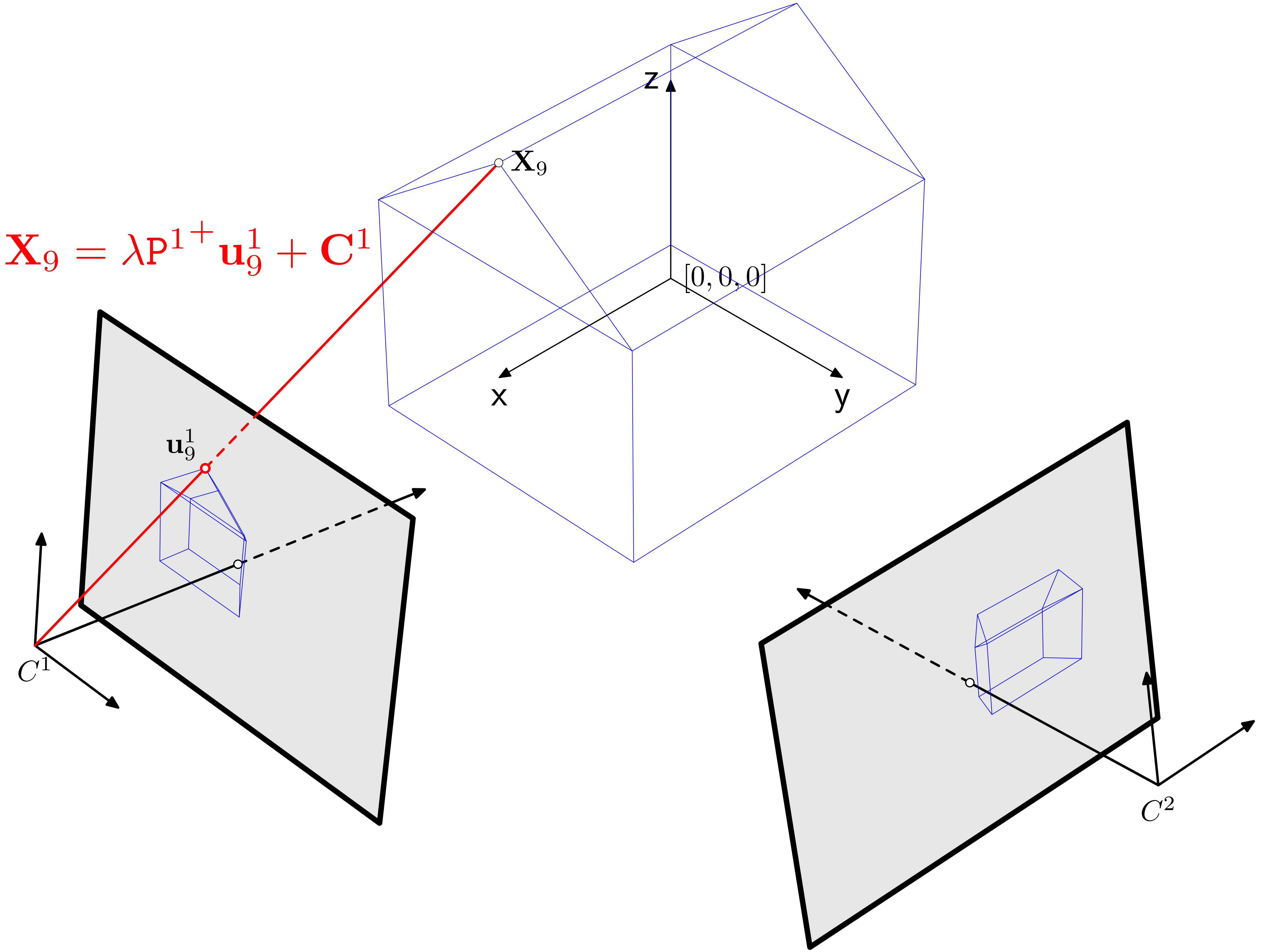
End



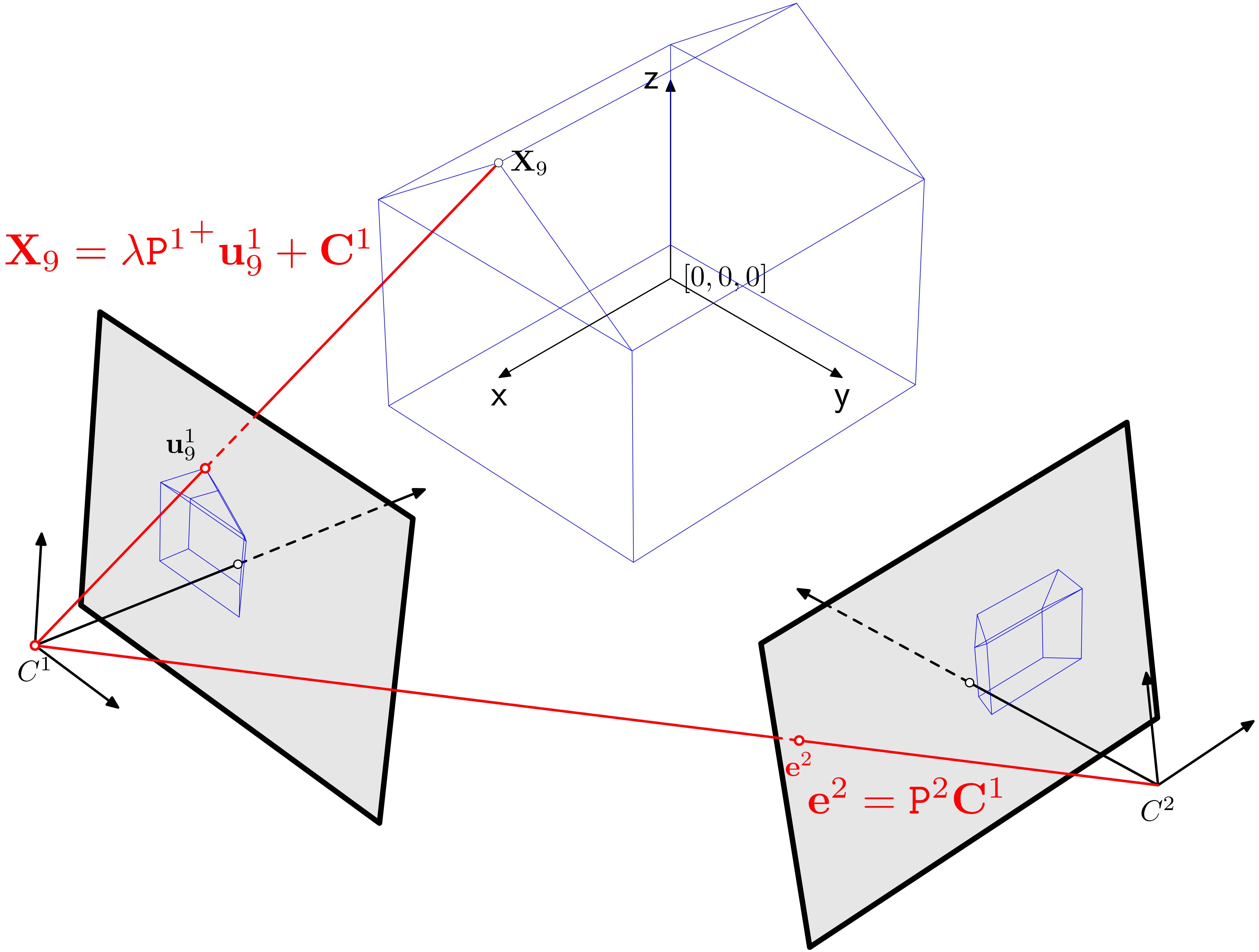
z
x
y
[0, 0, 0]
C^1
C^2


C^1

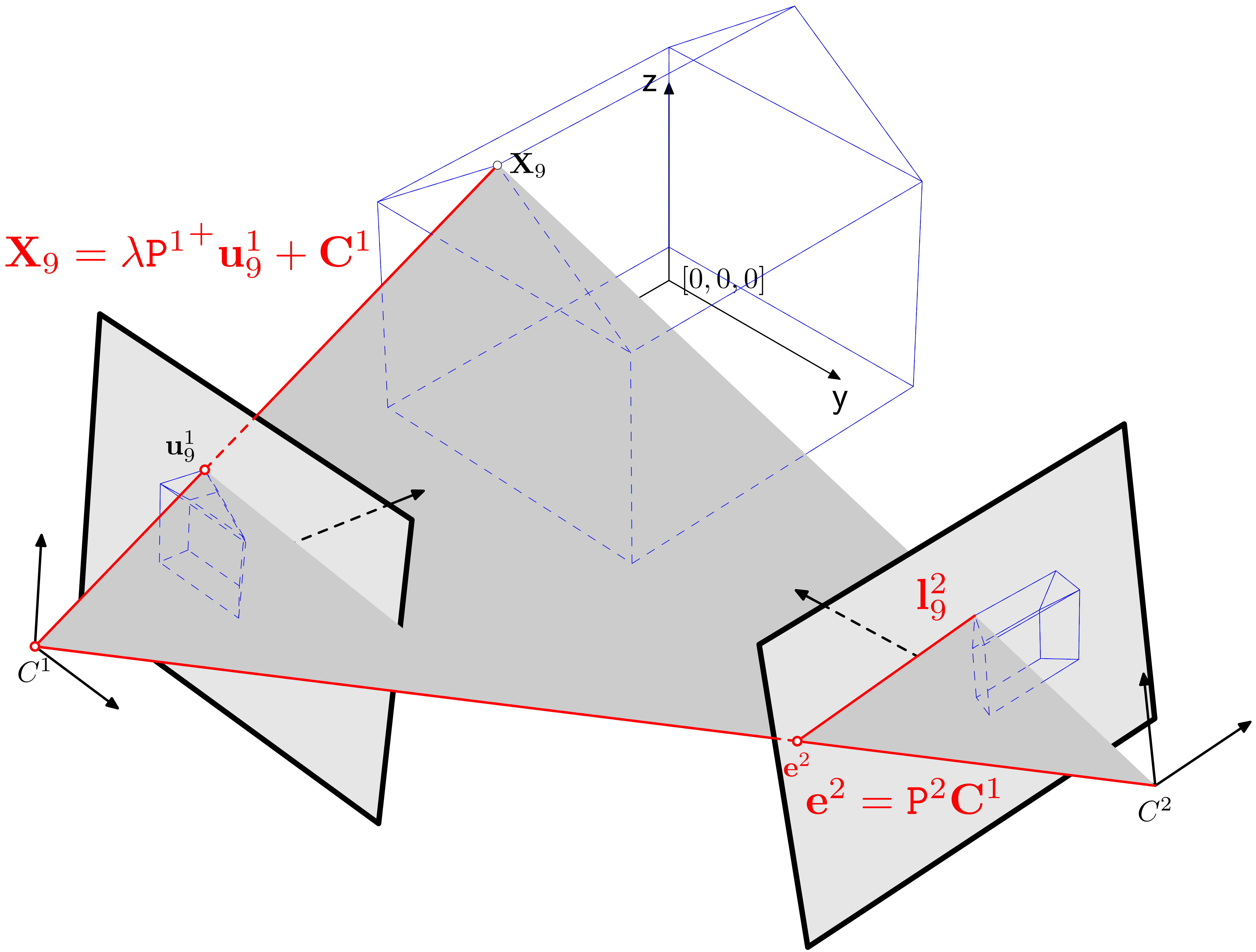

C^2



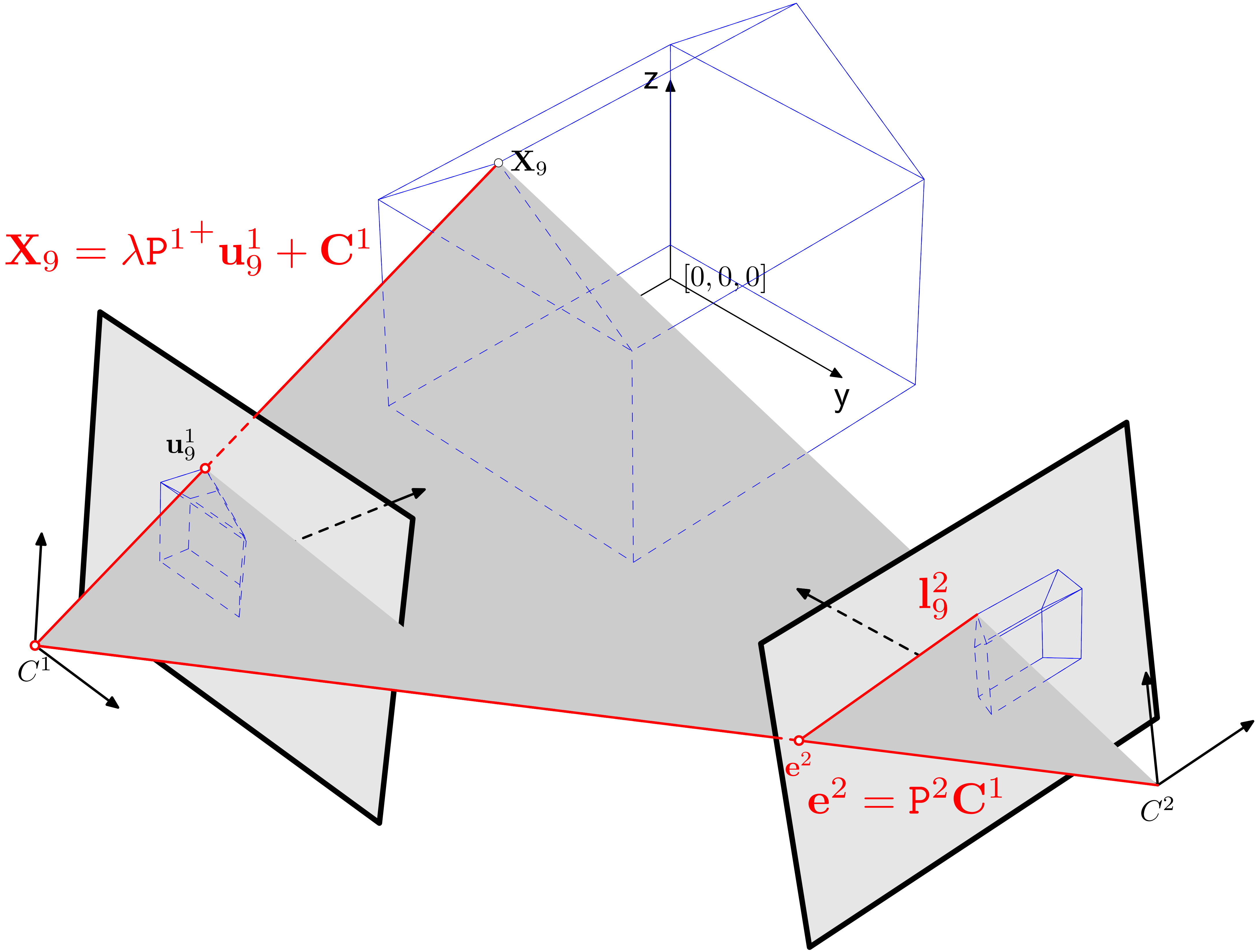
$\mathbf{X}_9 = \lambda \mathtt{P}^{1+} \mathbf{u}_9^1 + \mathbf{C}^1$
$\mathbf{X}_9$
z
x
y
$[0,0,0]$
$\mathbf{u}_9^1$
C^1
C^2



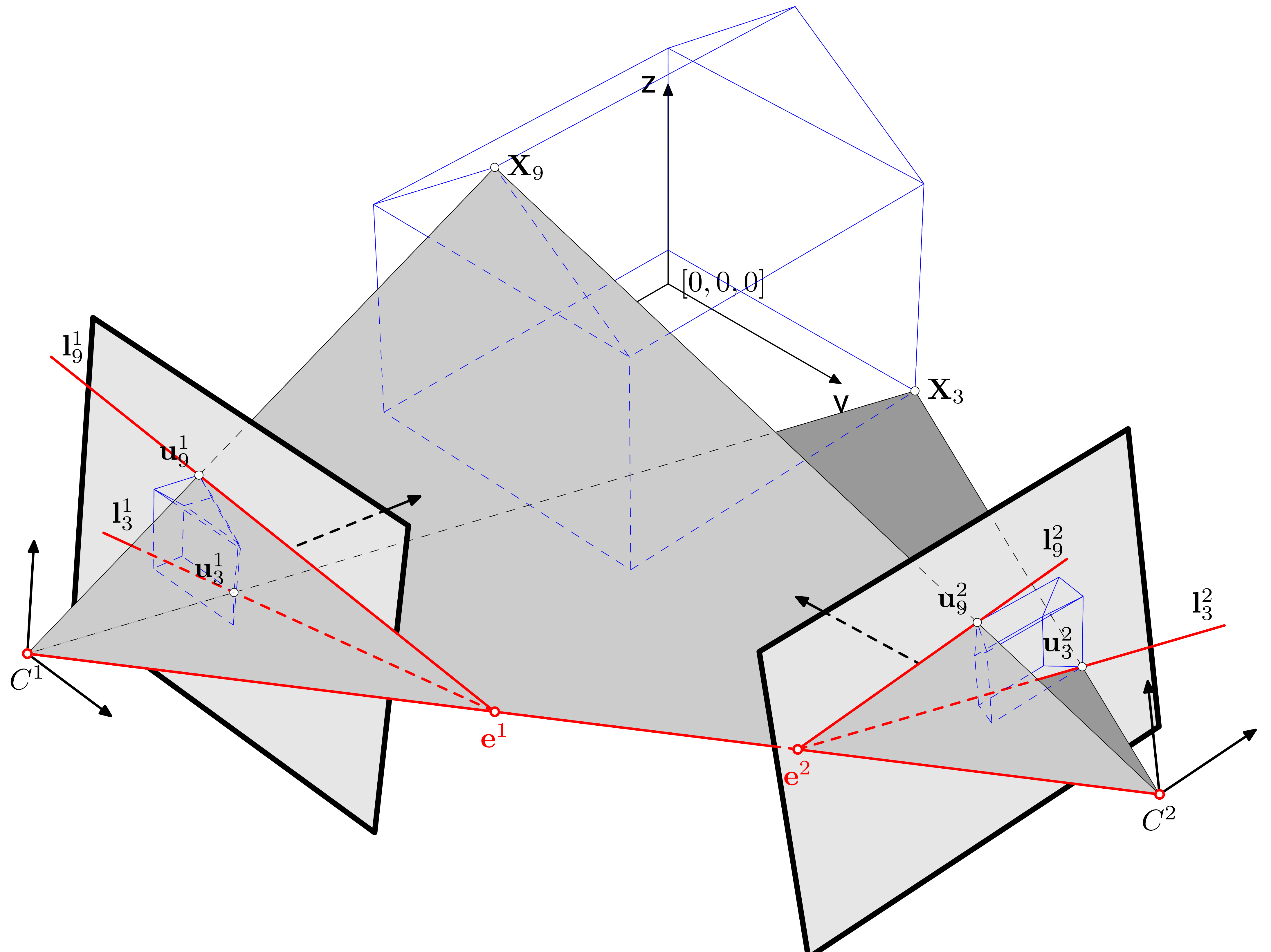
$\mathbf{X}_9 = \lambda \mathtt{P}^{1+} \mathbf{u}_9^1 + \mathbf{C}^1$
$\mathbf{X}_9$
z
$[0, 0, 0]$
x
y
$\mathbf{u}_9^1$
C^1
$\mathbf{e}^2$
$\mathbf{e}^2 = \mathtt{P}^2 \mathbf{C}^1$
C^2



$\mathbf{X}_9 = \lambda \mathtt{P}^{1+} \mathbf{u}_9^1 + \mathbf{C}^1$
$\mathbf{X}_9$
z
$[0,0,0]$
y
$\mathbf{u}_9^1$
C^1
$\mathbf{l}_9^2$
$\mathbf{e}^2$
$\mathbf{e}^2 = \mathtt{P}^2 \mathbf{C}^1$
C^2

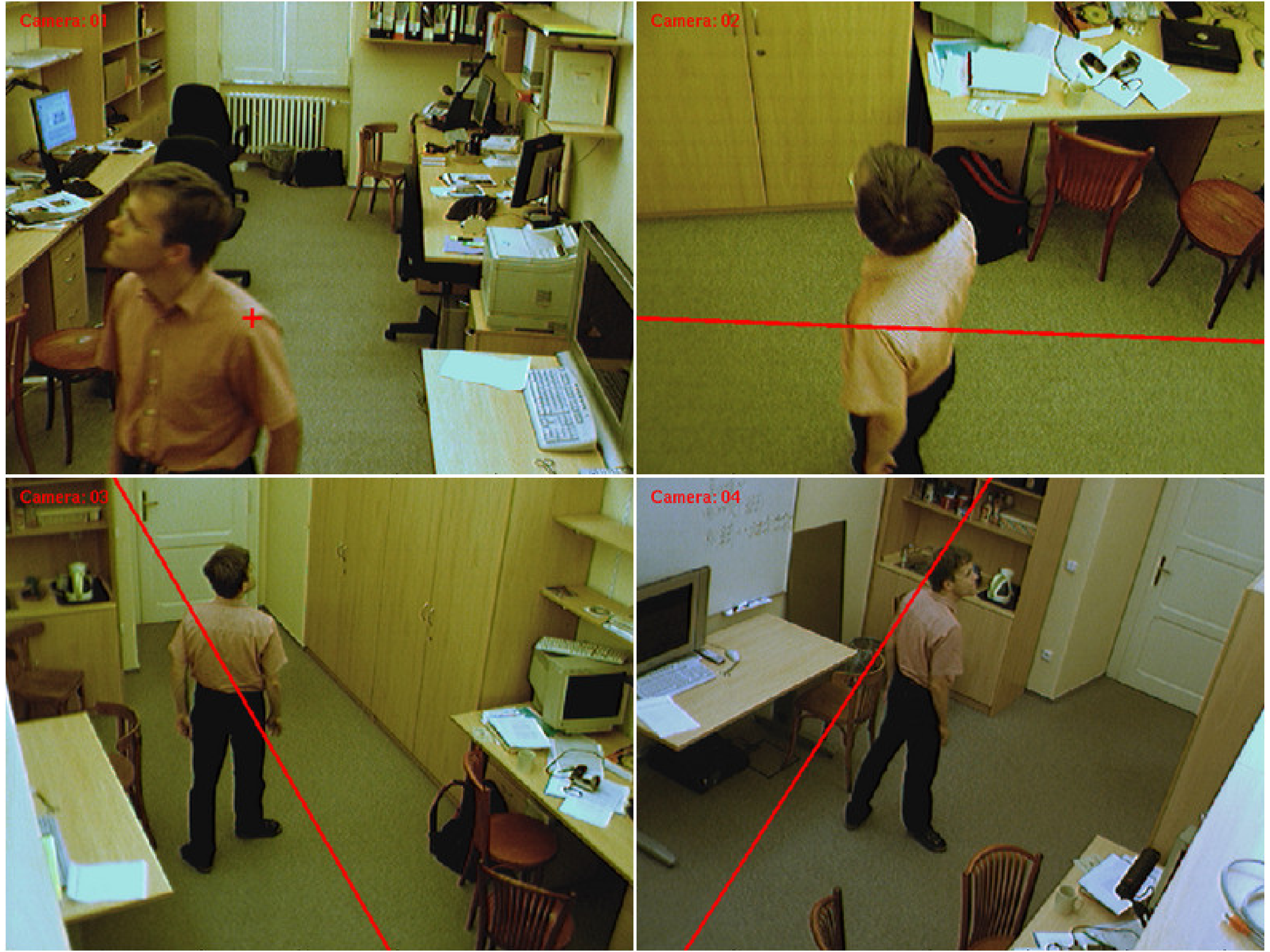


$\mathbf{X}_9 = \lambda \mathtt{P}^{1+}\mathbf{u}_9^1 + \mathbf{C}^1$
z
$\mathbf{X}_9$
$[0,0,0]$
y
$\mathbf{u}_9^1$
C^1
$\mathbf{l}_9^2$
$\mathbf{e}^2$
$\mathbf{e}^2 = \mathtt{P}^2\mathbf{C}^1$
C^2



z
$\mathbf{X}_9$
$[0,0,0]$
y
$\mathbf{X}_3$
$\mathbf{l}_9^1$
$\mathbf{u}_9^1$
$\mathbf{l}_3^1$
$\mathbf{u}_3^1$
C^1
$\mathbf{e}^1$
$\mathbf{l}_9^2$
$\mathbf{u}_9^2$
$\mathbf{l}_3^2$
$\mathbf{u}_3^2$
$\mathbf{e}^2$
C^2

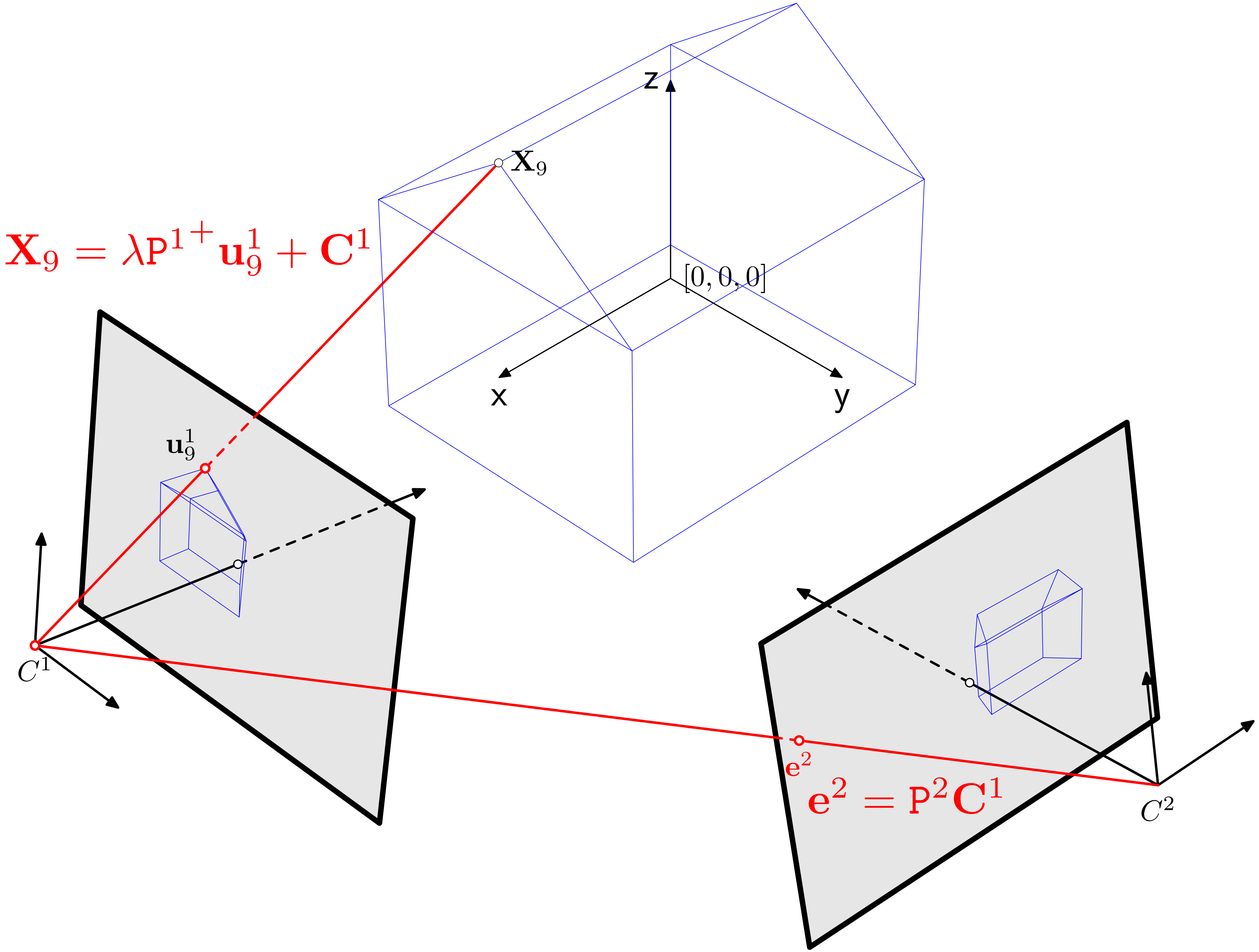

Cam: 01 Frame: 02402
Cam: 02 Frame: 02402
Cam: 03 Frame: 02402
Cam: 04 Frame: 02402



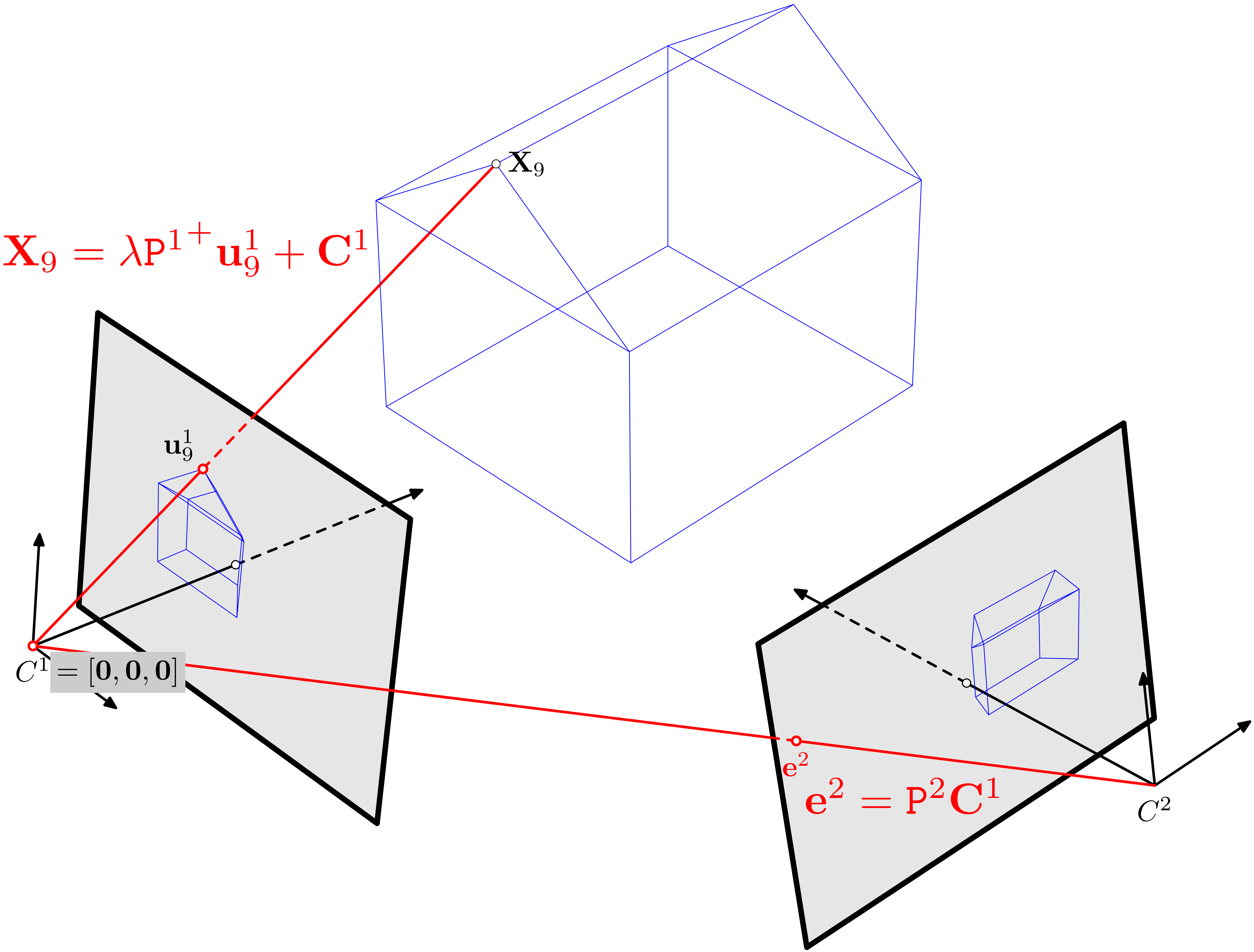
Camera: 01
Camera: 02
Camera: 03
Camera: 04

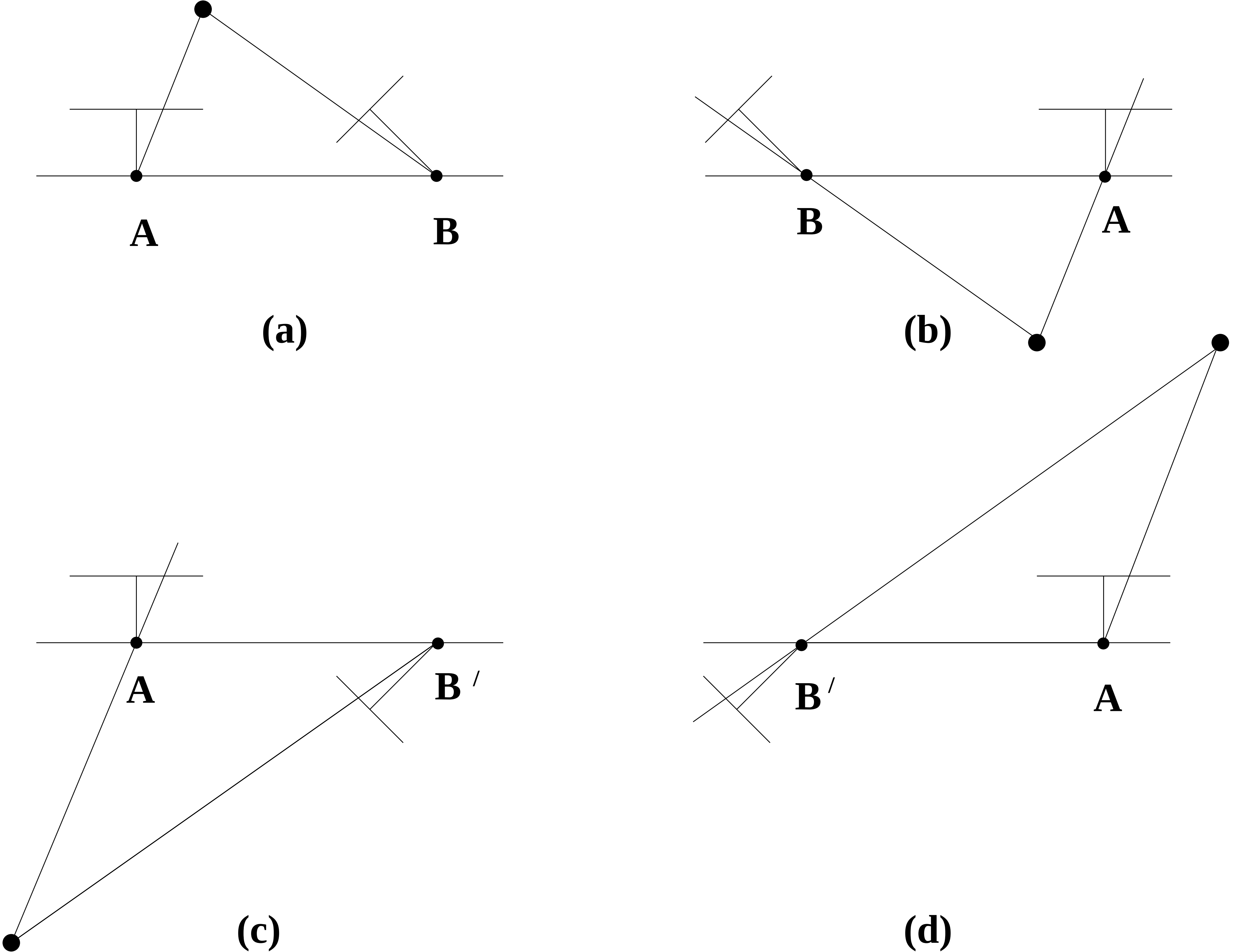

Camera: 01
Camera: 02
Camera: 03
Camera: 04

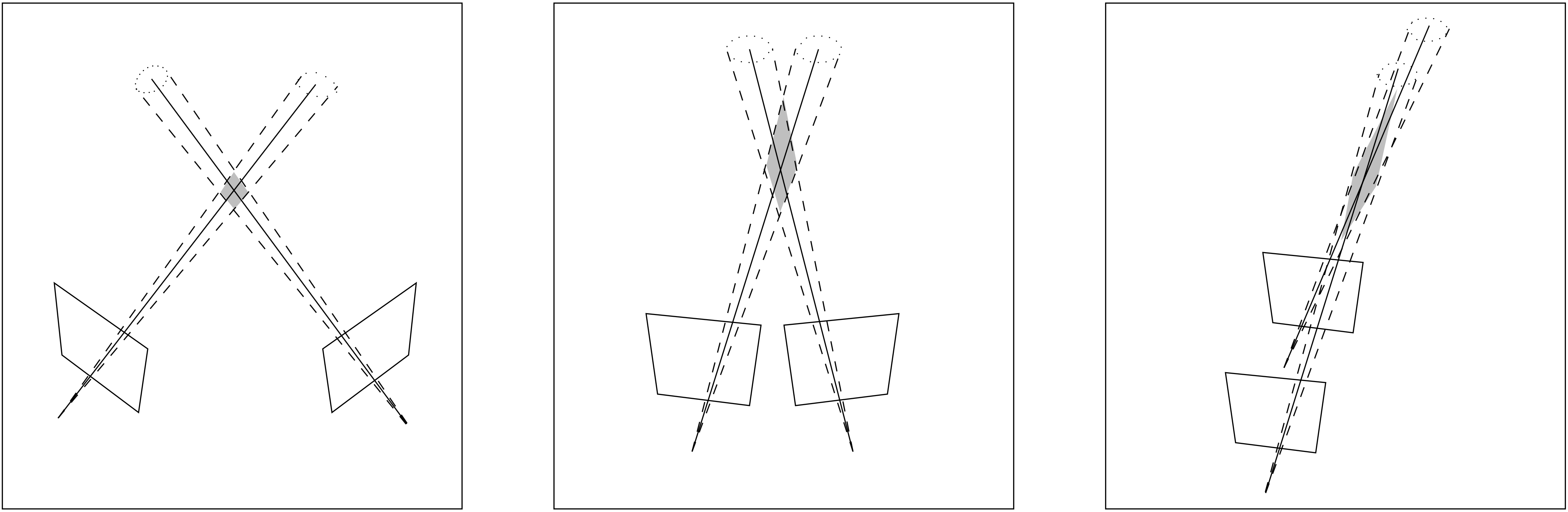

Camera: 01
Camera: 02
Camera: 03
Camera: 04

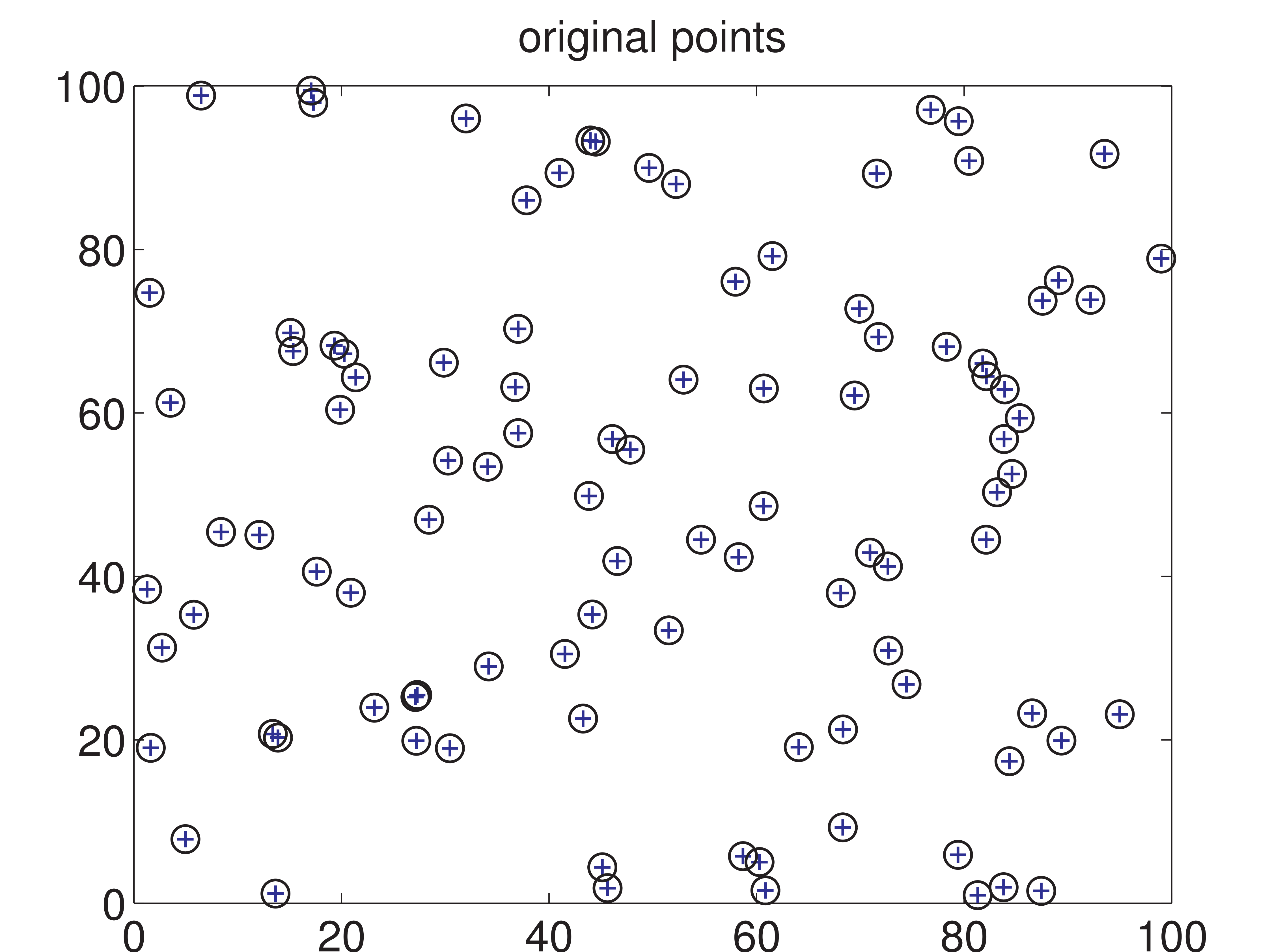


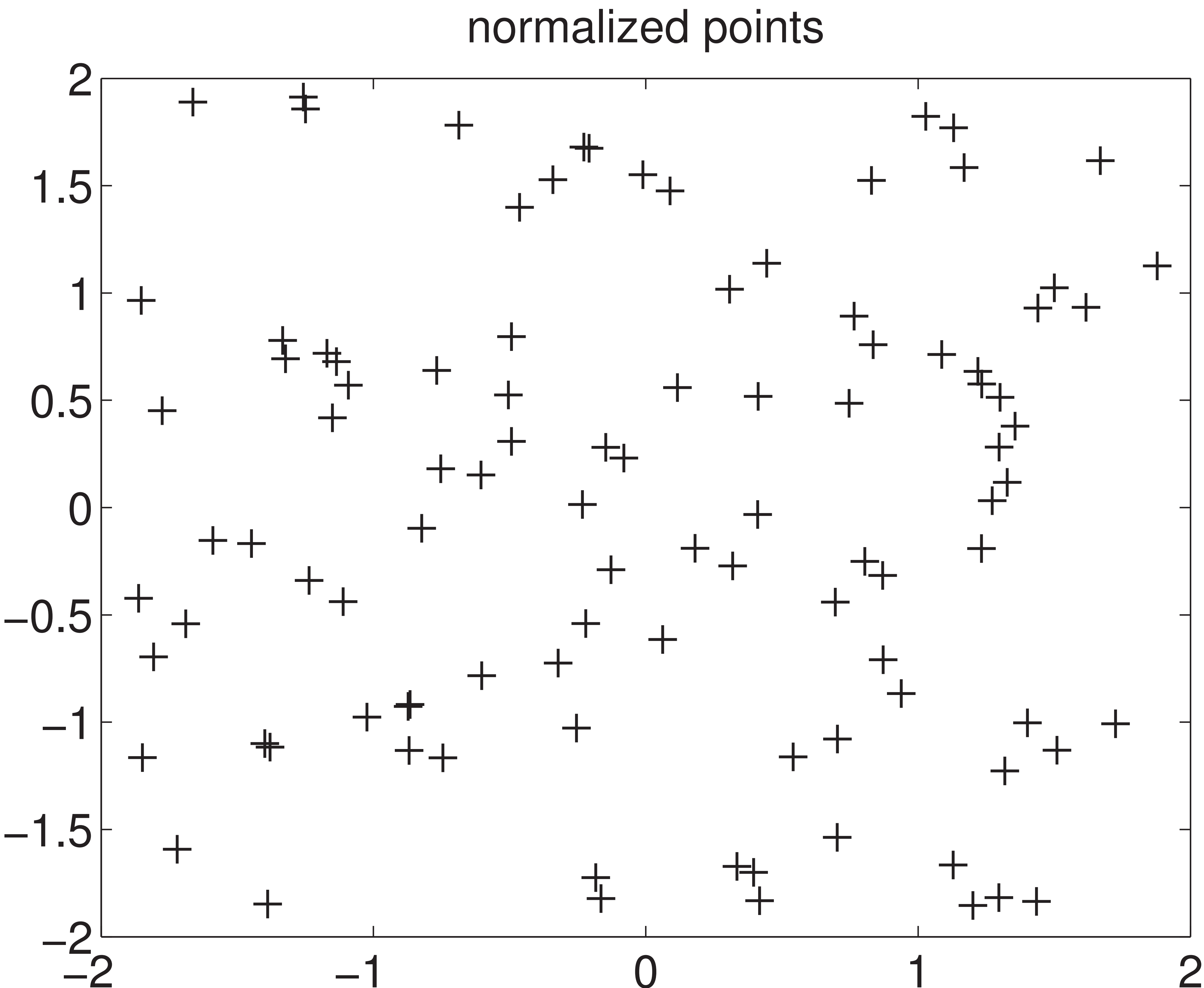
$\mathbf{X}_9 = \lambda \mathtt{P}^{1+} \mathbf{u}_9^1 + \mathbf{C}^1$
$\mathbf{X}_9$
z
$[0,0,0]$
x
y
$\mathbf{u}_9^1$
C^1
$\mathbf{e}^2$
$\mathbf{e}^2 = \mathtt{P}^2 \mathbf{C}^1$
C^2

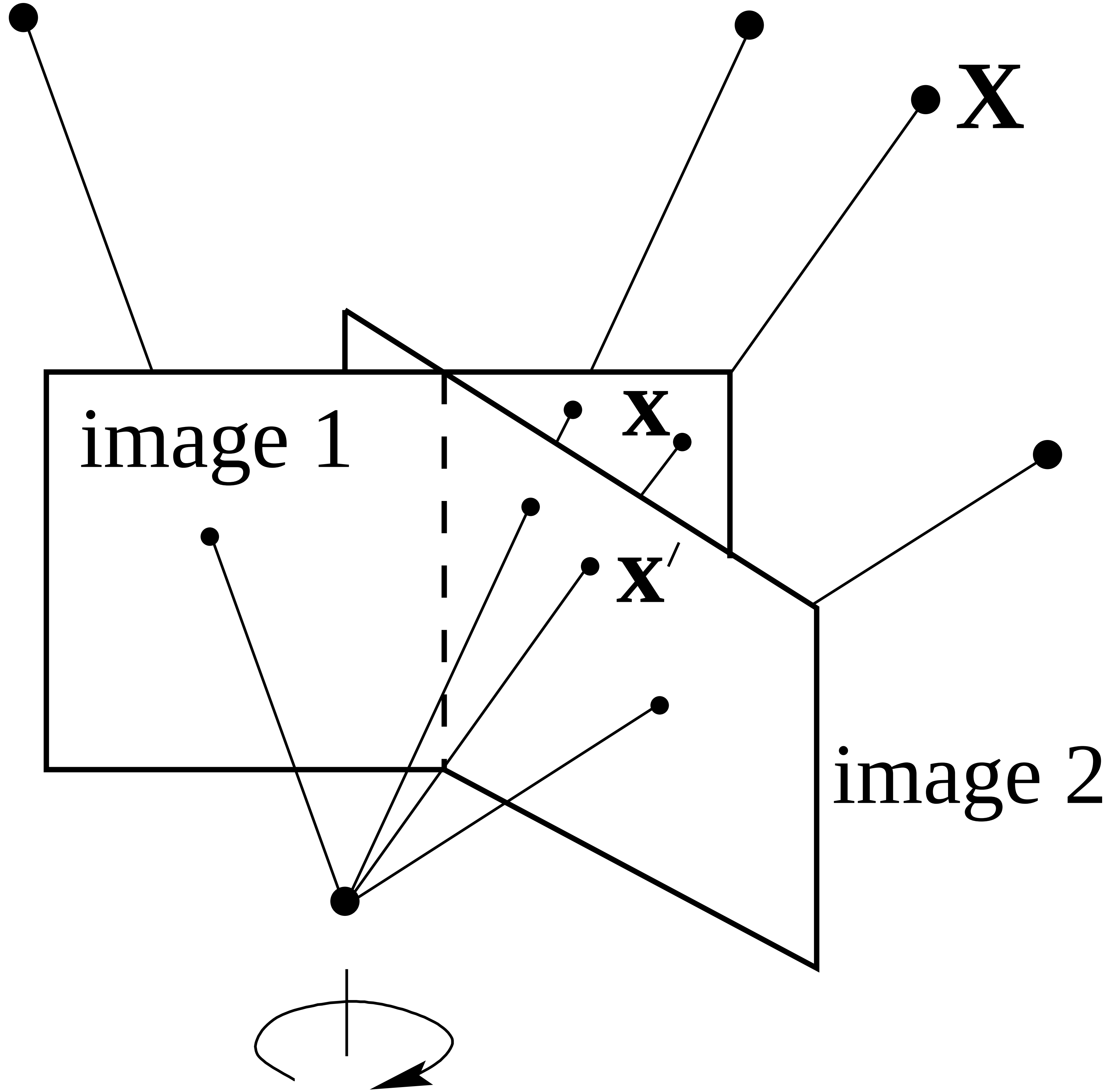

$\mathbf{X}_9$
$\mathbf{X}_9 = \lambda \mathtt{P}^{1+} \mathbf{u}_9^1 + \mathbf{C}^1$
$\mathbf{u}_9^1$
$C^1 = [\mathbf{0}, \mathbf{0}, \mathbf{0}]$
e^2
$\mathbf{e}^2 = \mathtt{P}^2 \mathbf{C}^1$
C^2


A
B
(a)
B
A
(b)
A
B /
(c)
B /
A
(d)

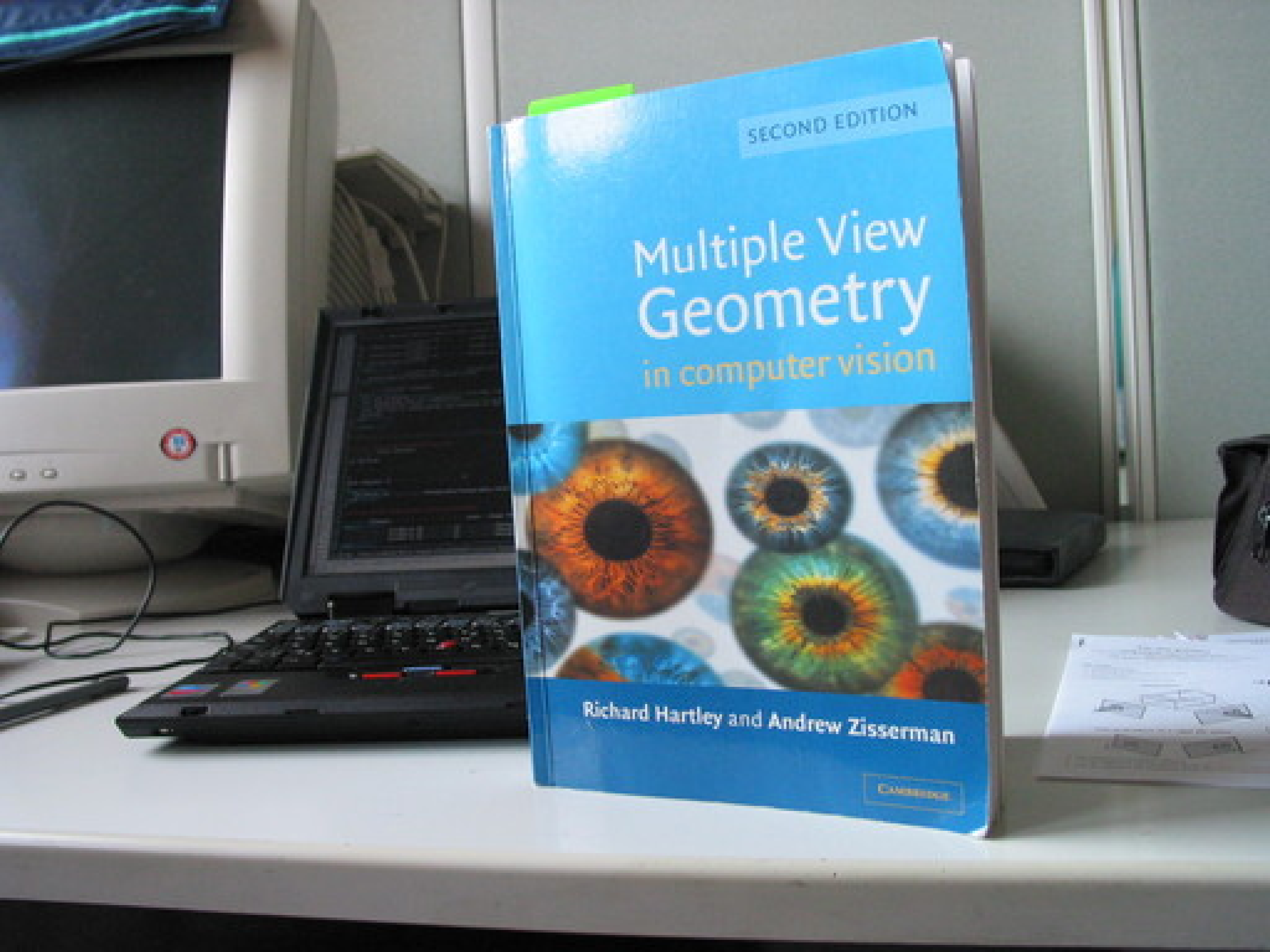



original points
0
20
40
60
80
100
0
20
40
60
80
100

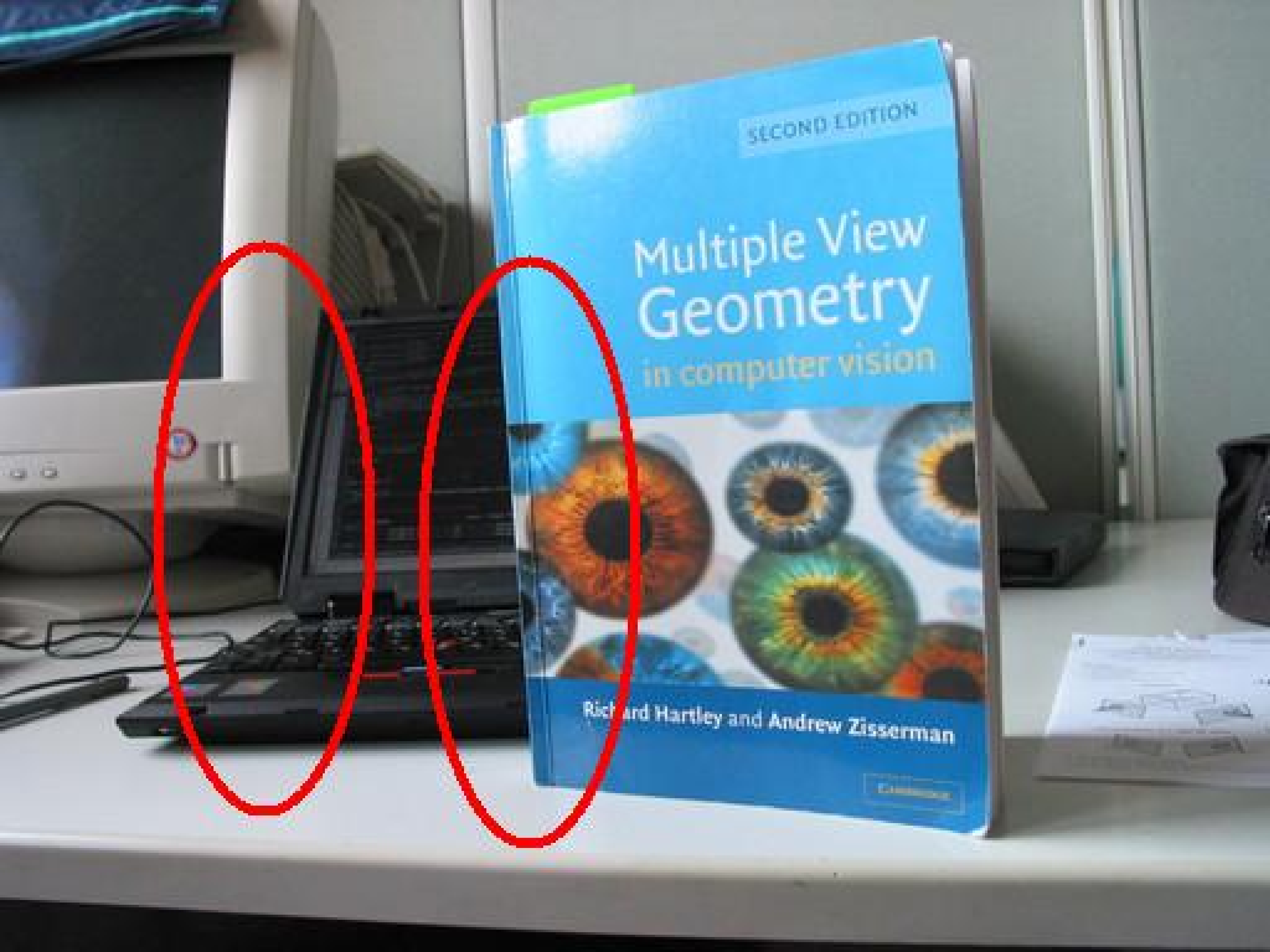

normalized points
2
1.5
1
0.5
0
−0.5
−1
−1.5
−2
−2
−1
0
1
2

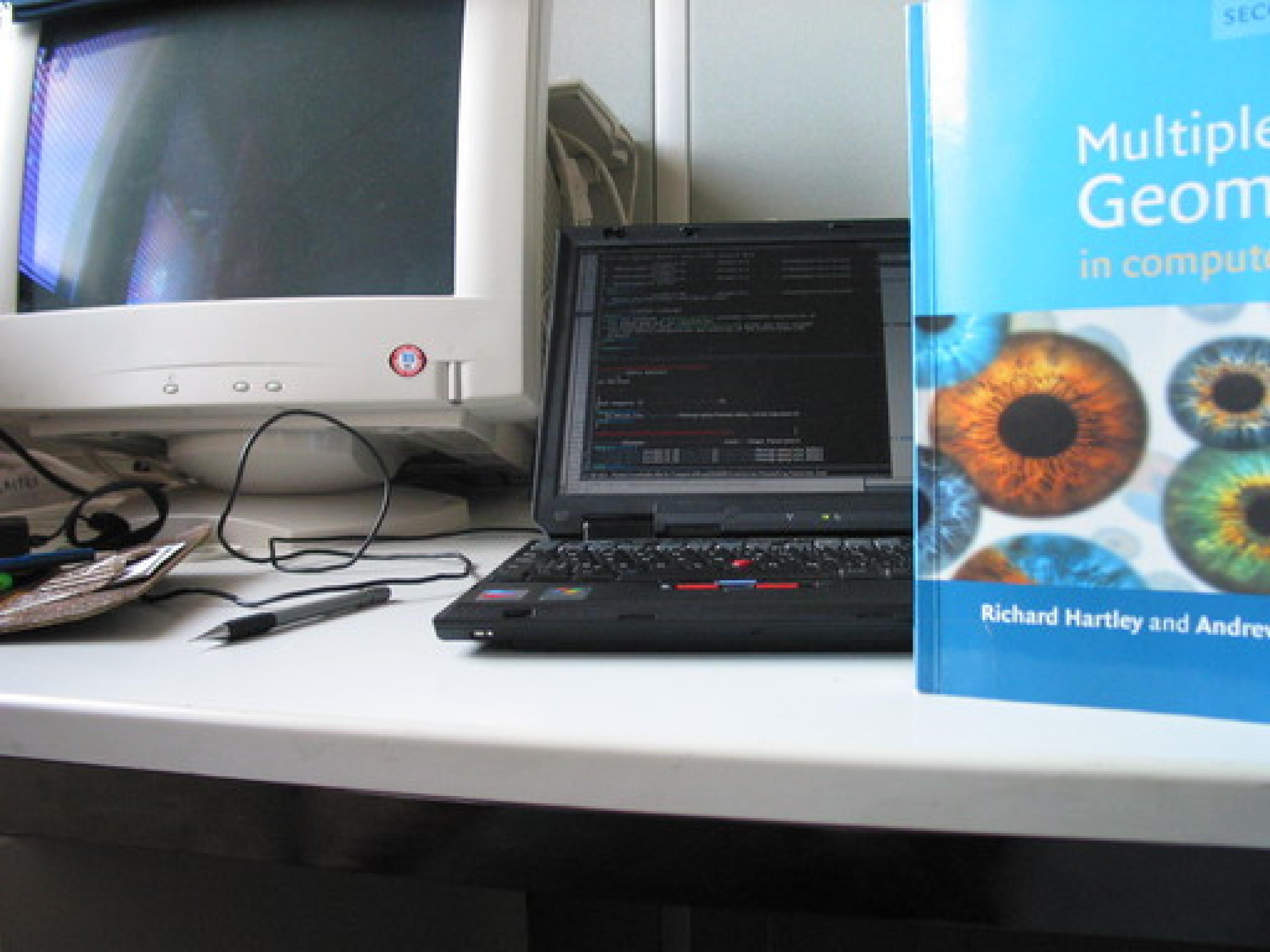

X
x
image 1
x′
image 2

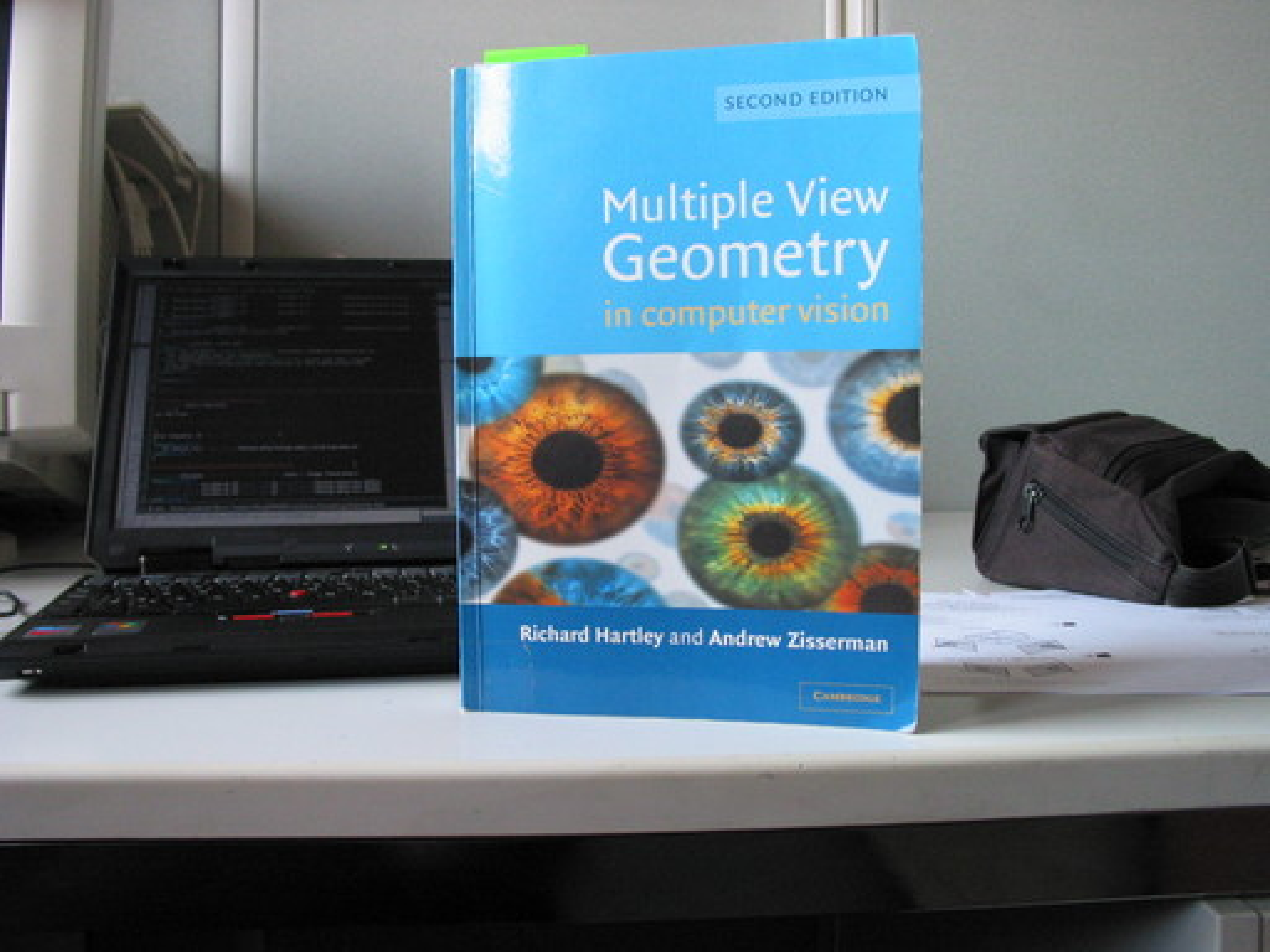

Richard Hartley and

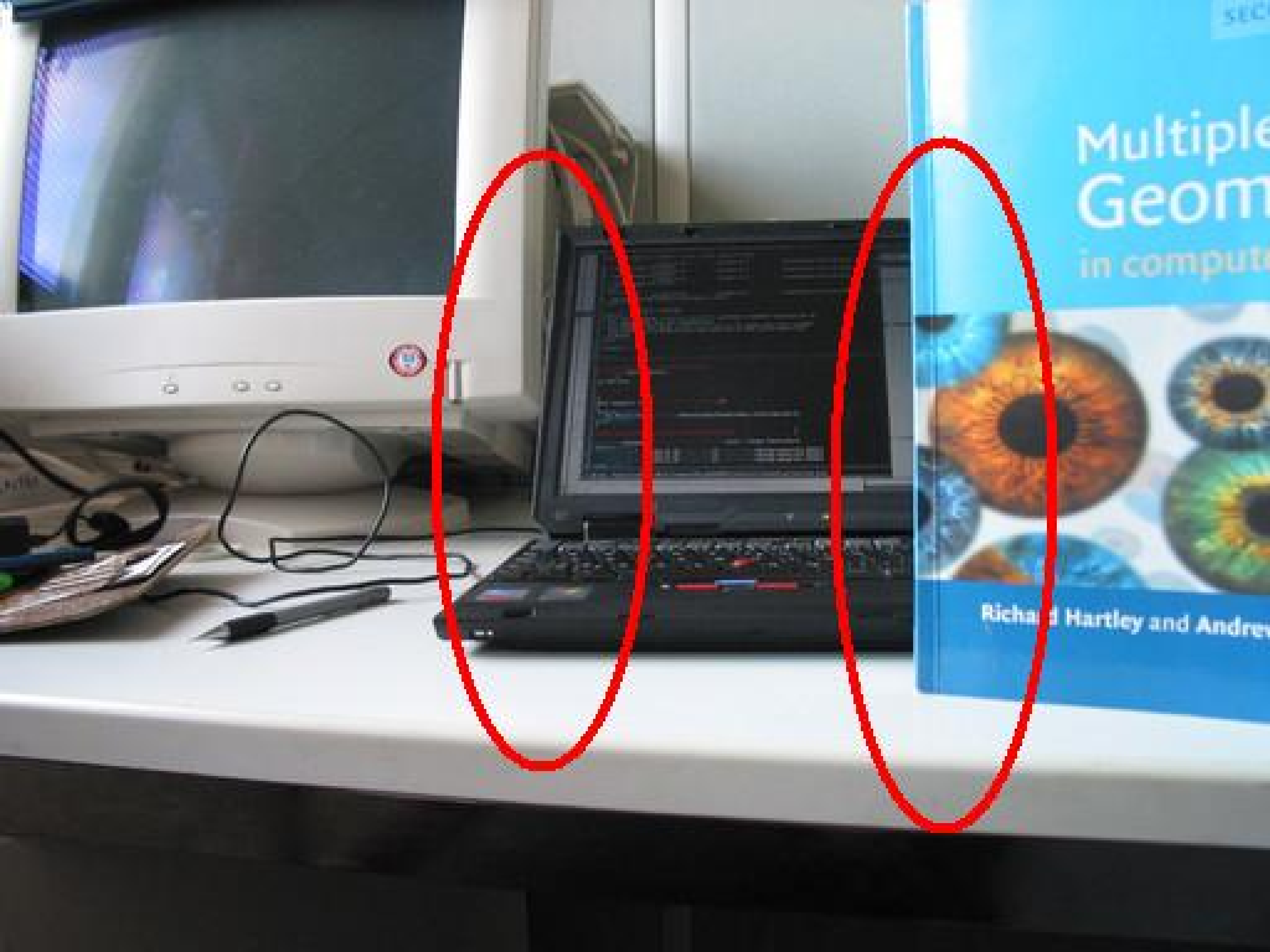

SECOND EDITION
Multiple View
Geometry
in computer vision
Richard Hartley and Andrew Zisserman

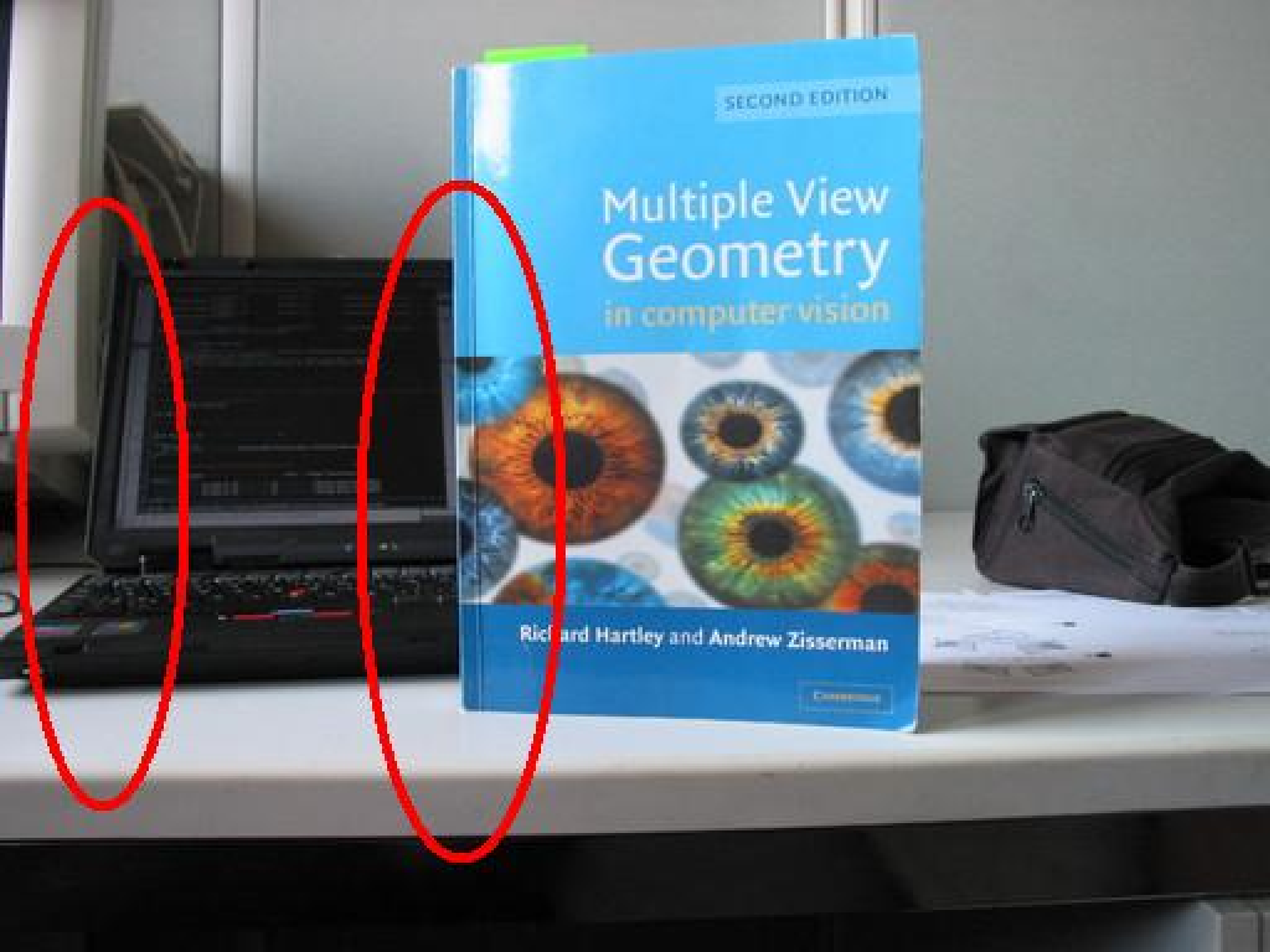

Multi
Geo
in com
hard Hartley and

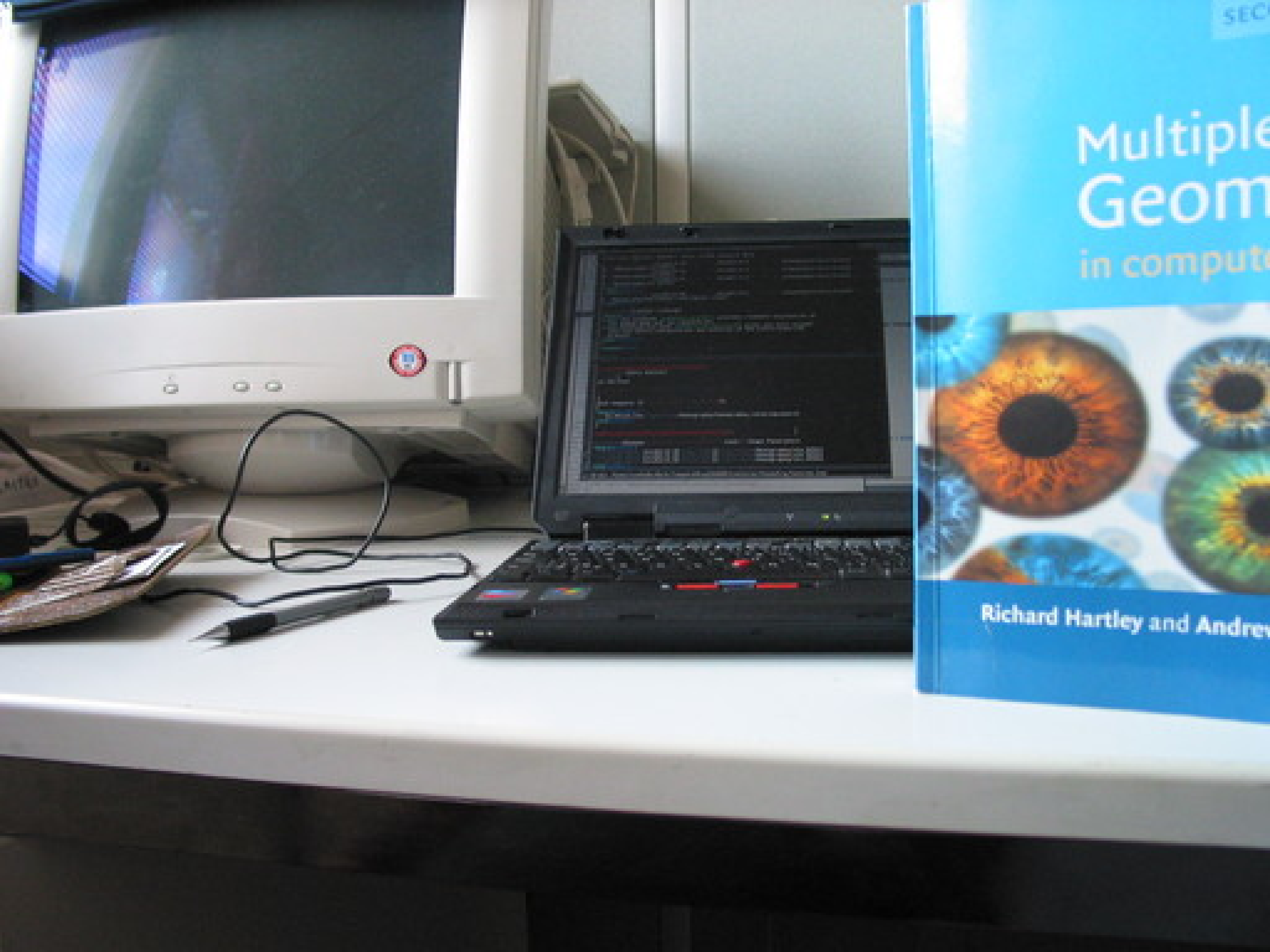

SECOND EDITION
Multiple View Geometry
in computer vision
Richard Hartley and Andrew Zisserman

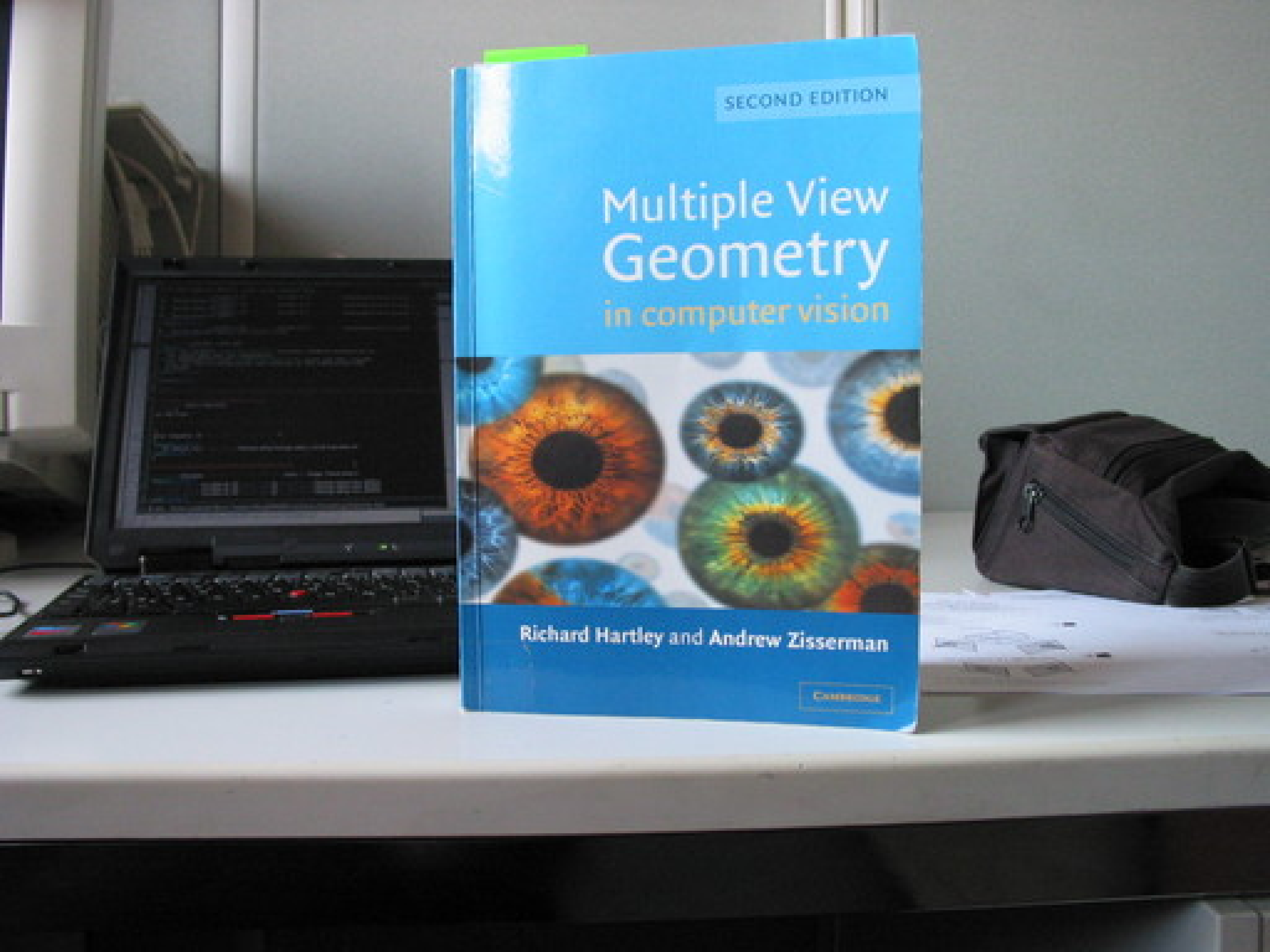

Multiple
Richard Hartley and

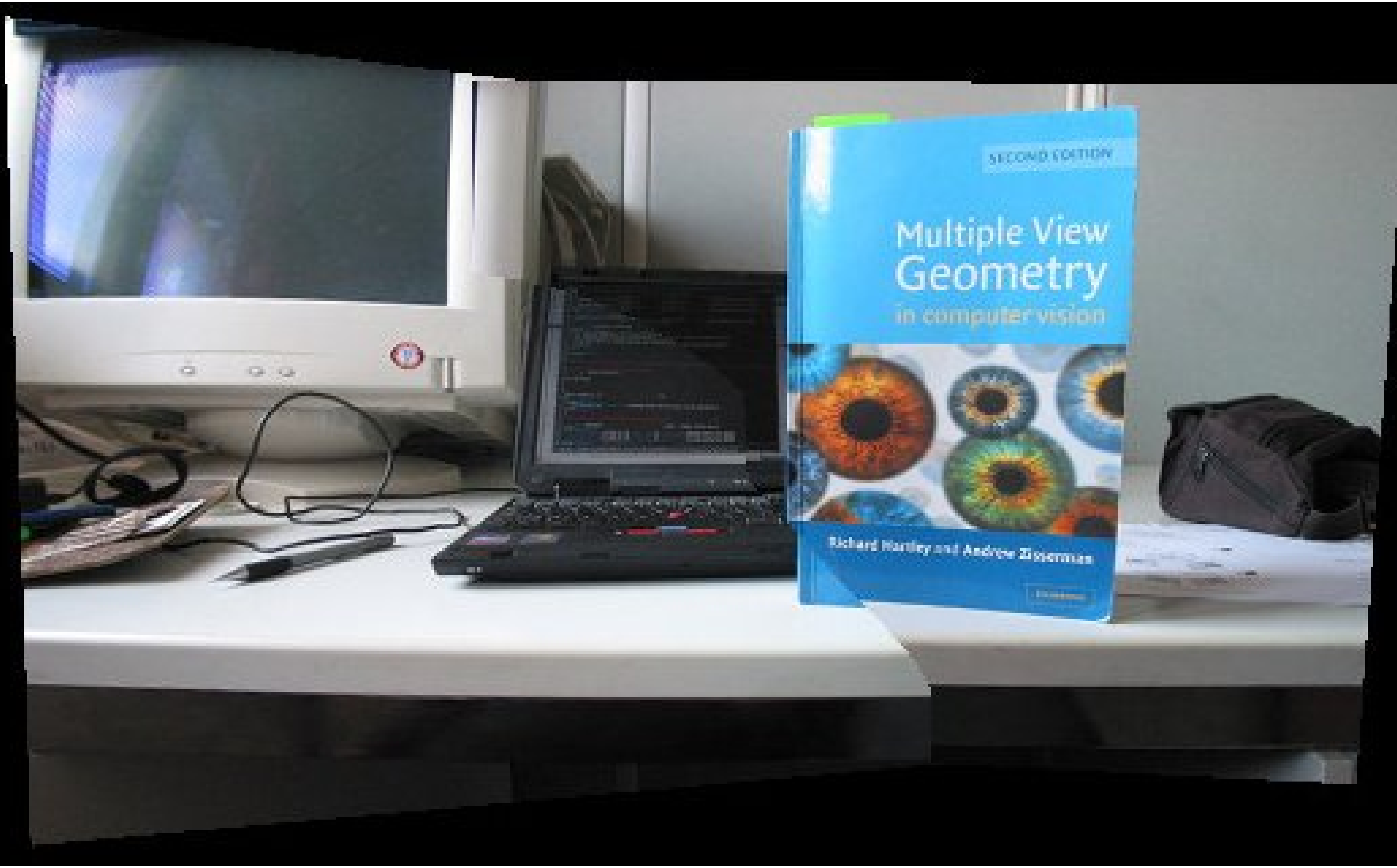

SECOND EDITION
Multiple View
Geometry
in computer vision
Richard Hartley and Andrew Zisserman
CAMBRIDGE


Multiple
Geom
in compute
Richa d Hartley and Andrew


SECOND EDITION
Multiple View
Geometry
in computer vision
Richard Hartley and Andrew Zisserman


Multiple
Richard Hartley and


SECOND EDITION
Multiple View
Geometry
in computer vision
Richard Hartley and Andrew Zisserman


SECOND EDITION
Multiple View Geometry
in computer vision
Richard Hartley and Andrew Zisserman