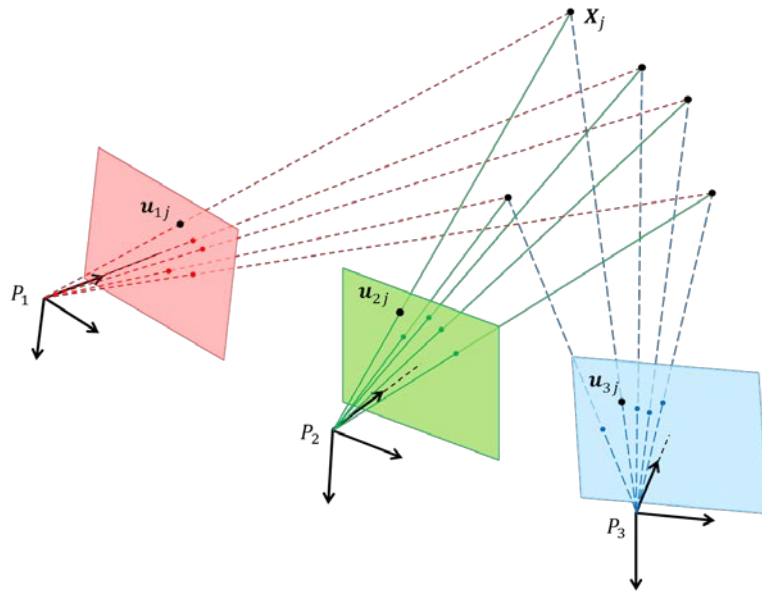


Lecture 8.1

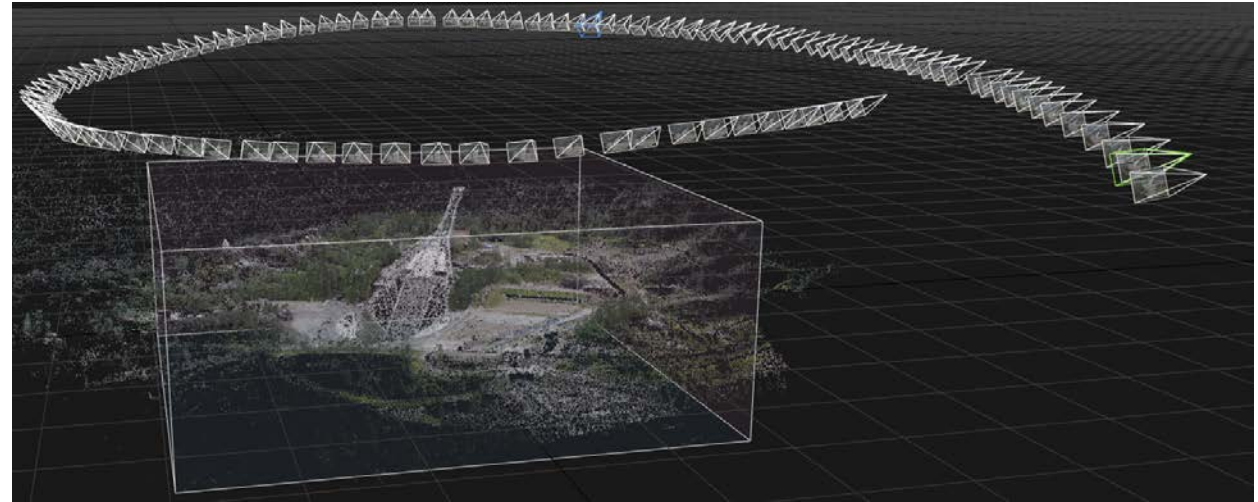
Multiple-View Geometry

Thomas Opsahl

Weekly overview

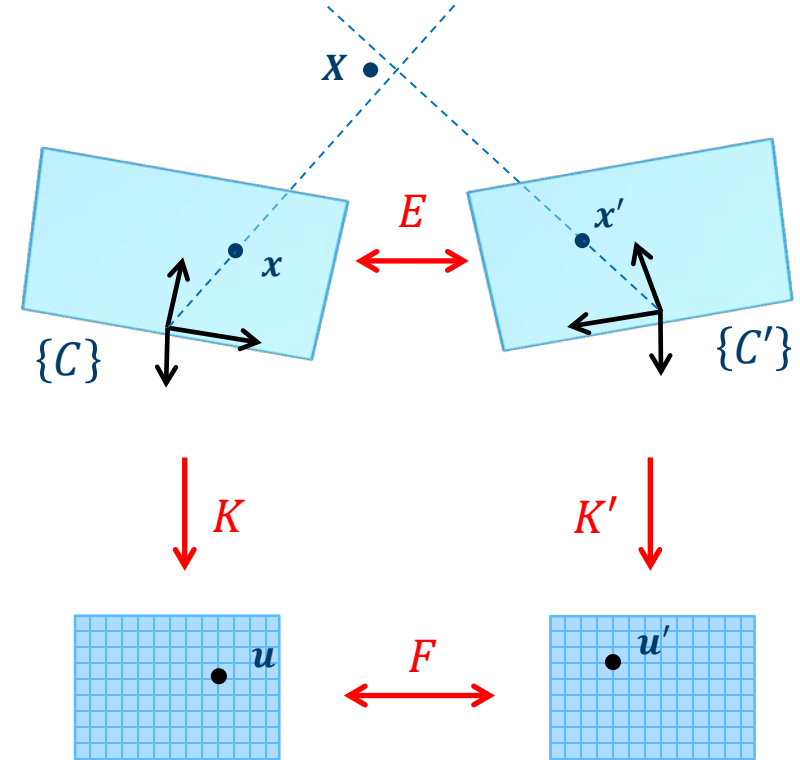


- Multiple-view geometry
 - Correspondences
- Structure from Motion (SfM)
 - Sparse 3D reconstruction
- Multiple-view stereo
 - Dense 3D reconstruction



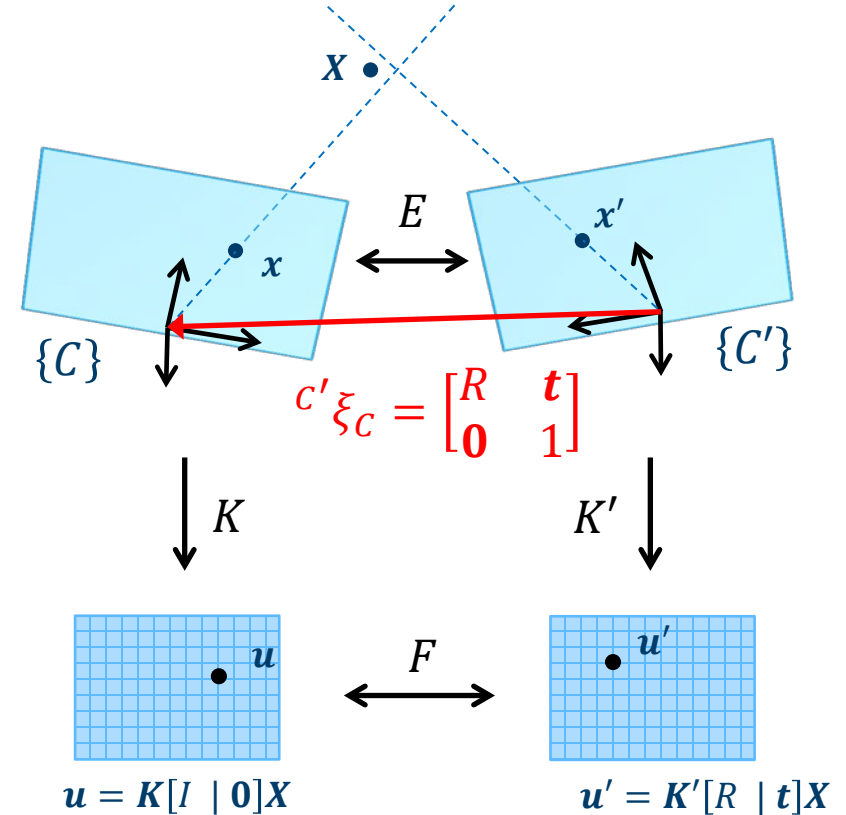
Recap on two-view geometry

- Epipolar geometry
 - The essential matrix $E = [t]_{\times} R$
 $\tilde{x}'^T E \tilde{x} = 0$
 - The fundamental matrix $F = K'^{-T} E K^{-1}$
 $\tilde{u}'^T F \tilde{u} = 0$
 - Estimating F from 7 or 8 correspondences $u_i \leftrightarrow u_i'$
 - Estimating E from 5 correspondences $x_i \leftrightarrow x_i'$



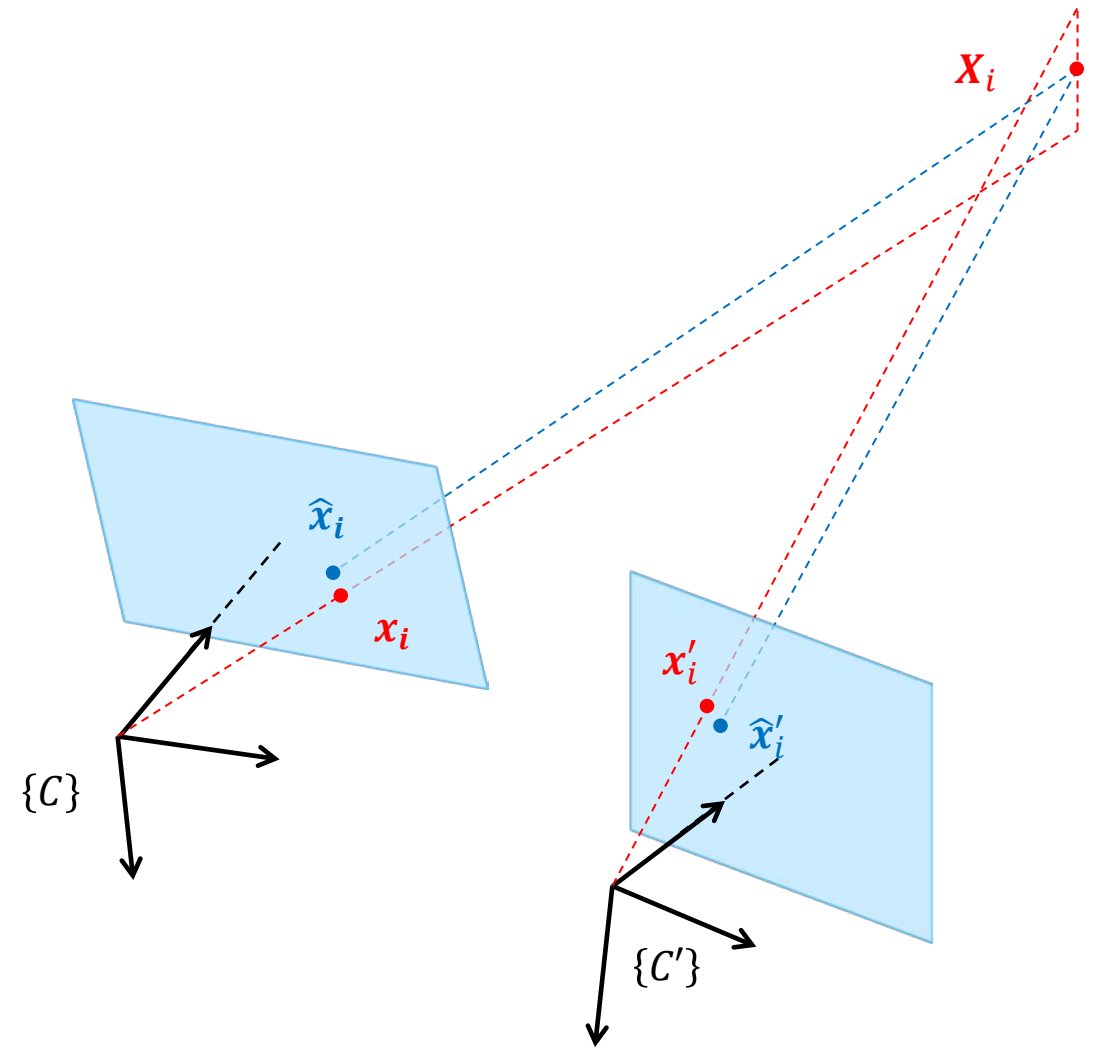
Recap on two-view geometry

- Epipolar geometry
 - The essential matrix $E = [t]_{\times} R$
 $\tilde{x}'^T E \tilde{x} = 0$
 - The fundamental matrix $F = K'^{-T} E K^{-1}$
 $\tilde{u}'^T F \tilde{u} = 0$
 - Estimating F from 7 or 8 correspondences $u_i \leftrightarrow u_i'$
 - Estimating E from 5 correspondences $x_i \leftrightarrow x_i'$
- Pose from epipolar geometry
 - Decomposing E into R and t (up to scale)



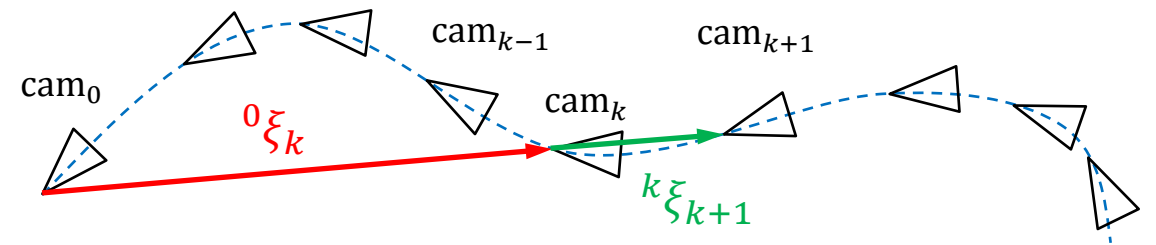
Recap on two-view geometry

- Epipolar geometry
 - The essential matrix $E = [t]_{\times} R$
 $\tilde{x}'^T E \tilde{x} = 0$
 - The fundamental matrix $F = K'^{-T} E K^{-1}$
 $\tilde{u}'^T F \tilde{u} = 0$
 - Estimating F from 7 or 8 correspondences $u_i \leftrightarrow u_i'$
 - Estimating E from 5 correspondences $x_i \leftrightarrow x_i'$
- Pose from epipolar geometry
 - Decomposing E into R and t (up to scale)
- 3D structure from epipolar geometry
 - Triangulation based on known camera matrices



Recap on two-view geometry

- Epipolar geometry
 - The essential matrix $E = [\mathbf{t}]_{\times} R$
 $\tilde{\mathbf{x}}'^T E \tilde{\mathbf{x}} = 0$
 - The fundamental matrix $F = K'^{-T} E K^{-1}$
 $\tilde{\mathbf{u}}'^T F \tilde{\mathbf{u}} = 0$
 - Estimating F from 7 or 8 correspondences $\mathbf{u}_i \leftrightarrow \mathbf{u}_i'$
 - Estimating E from 5 correspondences $\mathbf{x}_i \leftrightarrow \mathbf{x}_i'$
- Pose from epipolar geometry
 - Decomposing E into R and \mathbf{t} (up to scale)
- 3D structure from epipolar geometry
 - Triangulation based on known camera matrices
- Sequential visual odometry
 - $\mathbf{x}_i^{(k)} \leftrightarrow \mathbf{x}_i^{(k+1)} \rightarrow E_{k,k+1} \rightarrow {}^k\xi_{k+1}$
 - ${}^0\xi_{k+1} = {}^0\xi_k \cdot {}^k\xi_{k+1}$



Recap on two-view geometry

Correspondences (matching)

- Correspondences must satisfy the epipolar constraint represented by the fundamental matrix
$$\tilde{\mathbf{u}}'^T F \tilde{\mathbf{u}} = 0$$
- Useful for reducing the number of mismatches

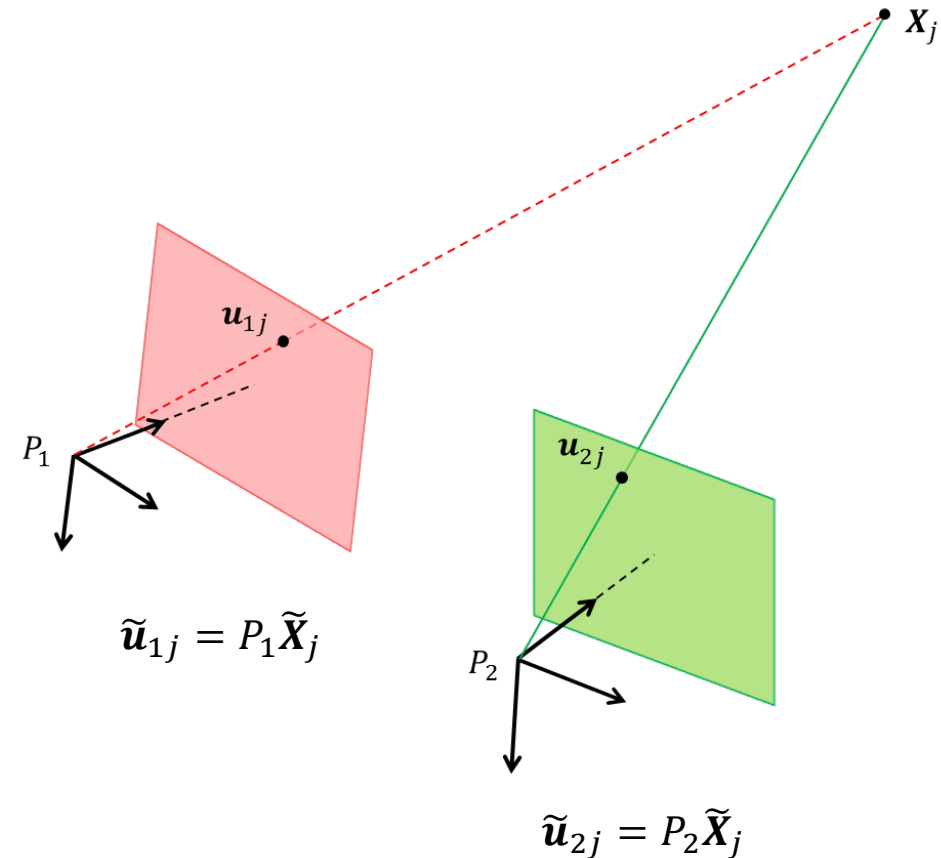
Scene geometry (structure)

- Sparse 3D from triangulating correspondences
- Dense 3D from stereo processing

Camera geometry (motion)

- In the uncalibrated case, the camera matrices P and P' can be estimated from the fundamental matrix F up to a projective ambiguity
- In the calibrated case, the relative pose between cameras can be estimated up to scale by decomposing the essential matrix

$$(\mathbf{x}_j \leftrightarrow \mathbf{x}'_j) \xrightarrow{E=[\mathbf{t}] \times^R} (R, \lambda \mathbf{t})$$



More-than-two-view geometry

Correspondences (matching)

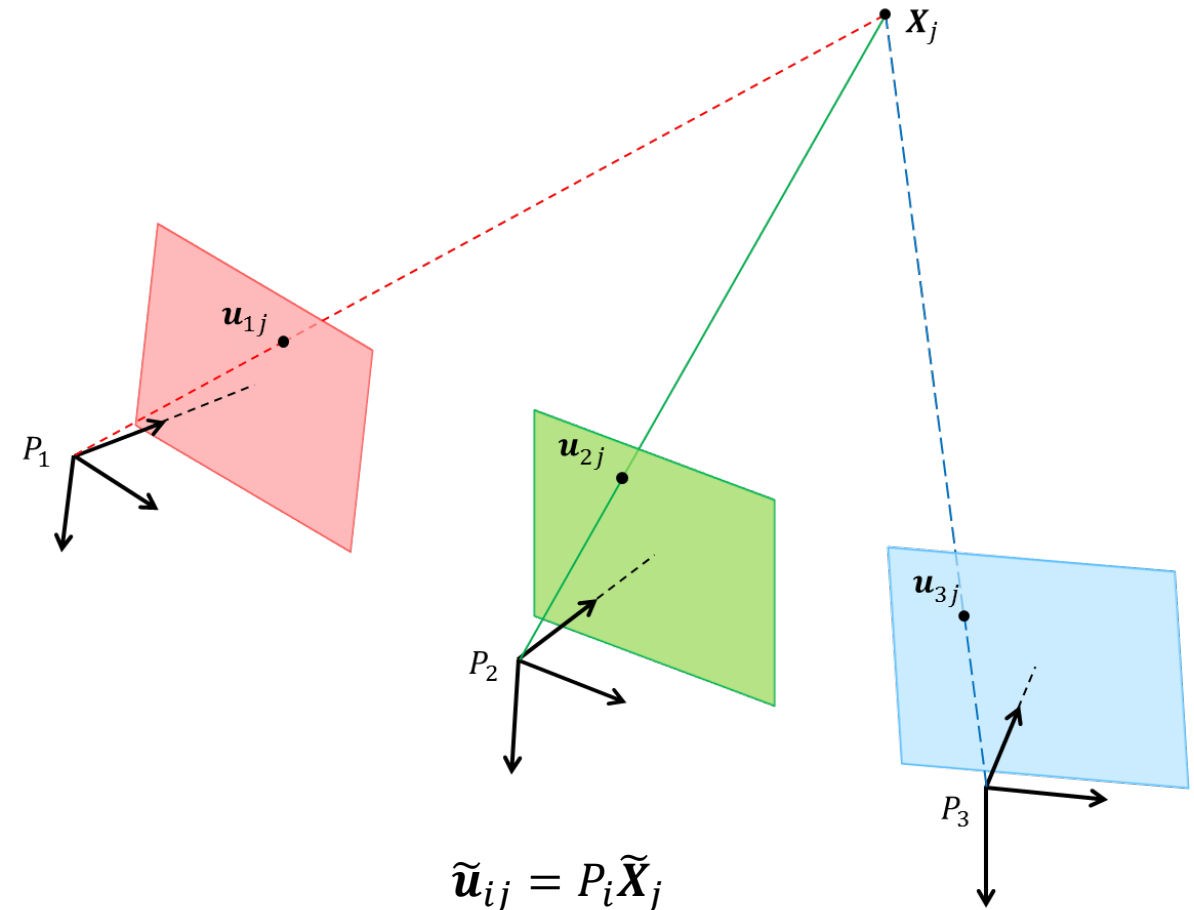
- How does “more-than-two-view geometry” constrain our 2D matches?
- Algebraic description?

Scene geometry (structure)

- Effect of more views on determining the 3D structure of the scene?
- Next lecture

Camera geometry (motion)

- Effect of more views on determining camera poses?
- Next lecture



Correspondences

Two views

- Points \mathbf{u}_1 and \mathbf{u}_2 must satisfy the epipolar constraint

- The fundamental matrix F represents this constraint

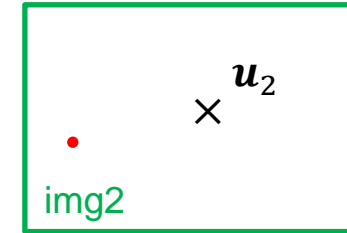
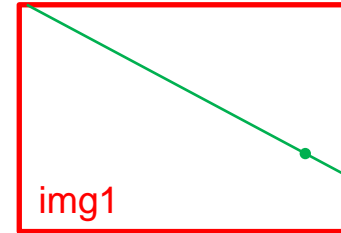
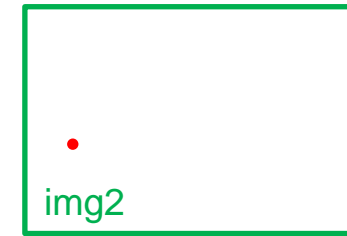
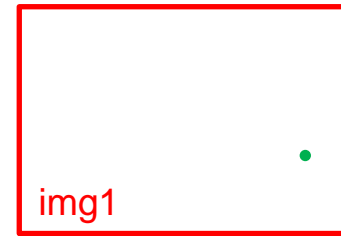
$$\tilde{\mathbf{u}}_2^T F \tilde{\mathbf{u}}_1 = 0$$

- F also describes the correspondence between points and epipolar lines

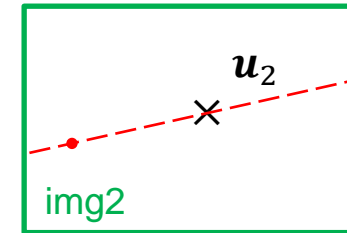
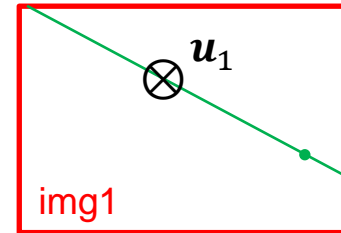
$$\tilde{\mathbf{l}}_2 = F \tilde{\mathbf{u}}_1$$

$$\tilde{\mathbf{l}}_1 = F^T \tilde{\mathbf{u}}_2$$

Epipoles



A point \mathbf{u}_2 in img2 correspond to a line in img1



A point \mathbf{u}_1 in img1 correspond to a line in img2

Correspondences

Three views

Epipoles

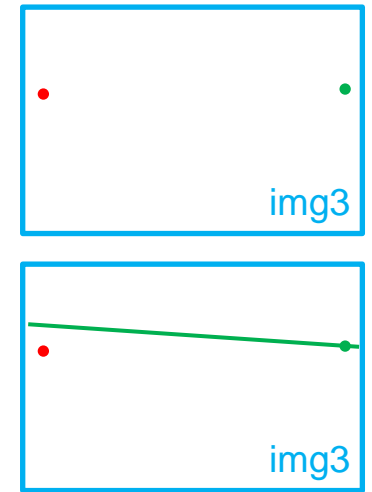
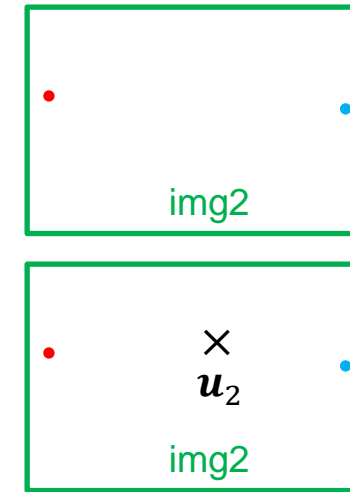
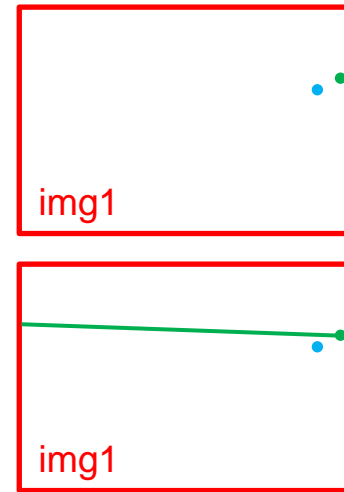


Correspondences

Three views

Epipoles

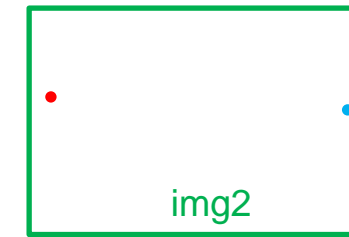
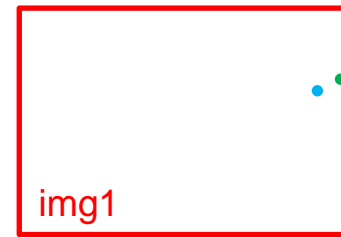
A point u_2 in img2
correspond to lines
in img1 and img3



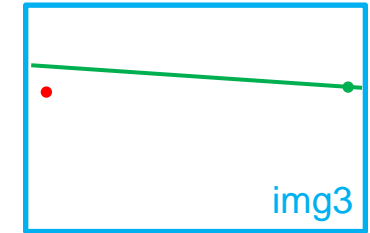
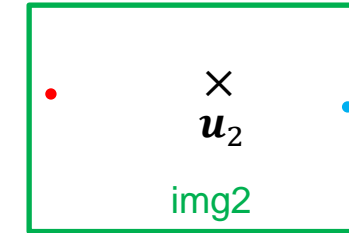
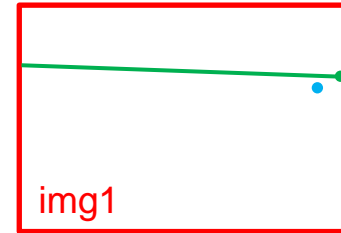
Correspondences

Three views

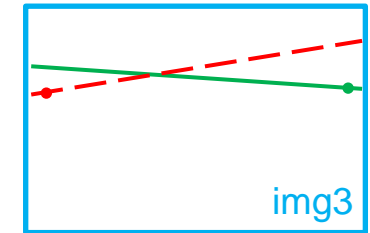
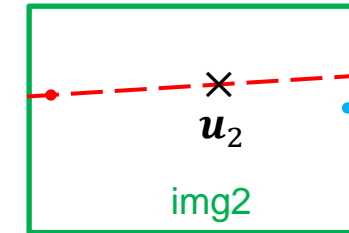
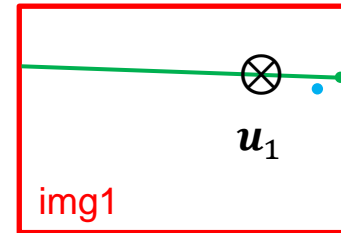
Epipoles



A point u_2 in img2
correspond to lines
in img1 and img3



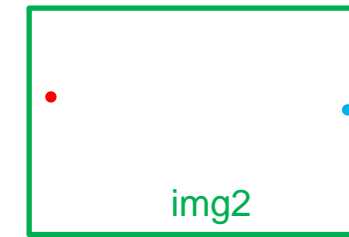
A point u_1 in img1
correspond to lines
in img2 and img3



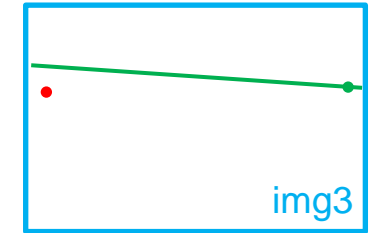
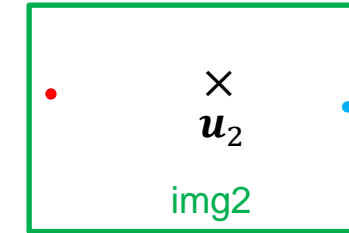
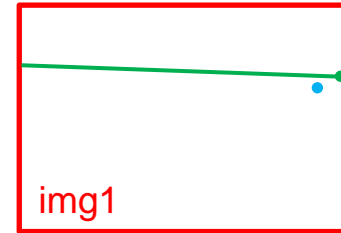
Correspondences

Three views

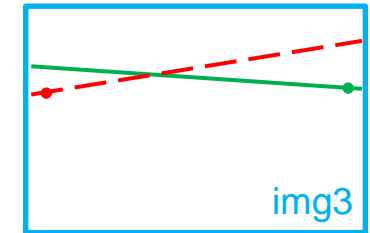
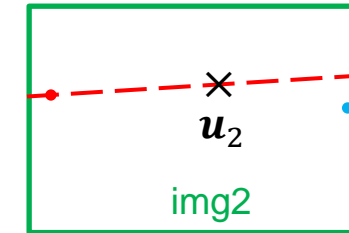
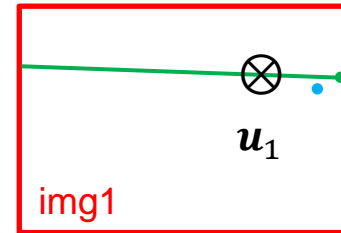
Epipoles



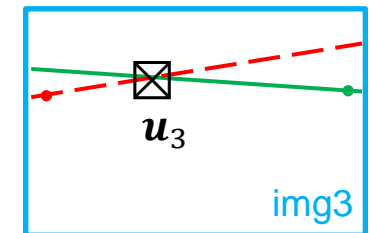
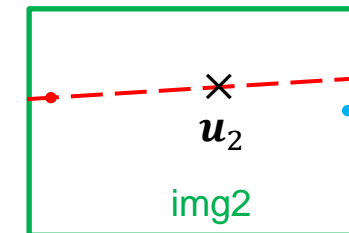
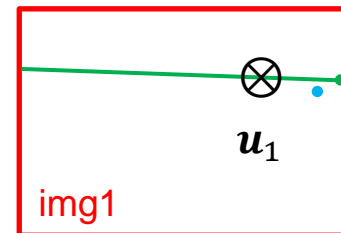
A point u_2 in img2 correspond to lines in img1 and img3



A point u_1 in img1 correspond to lines in img2 and img3



Points u_1 and u_2 define a point u_3 in img3



Correspondences

Three views

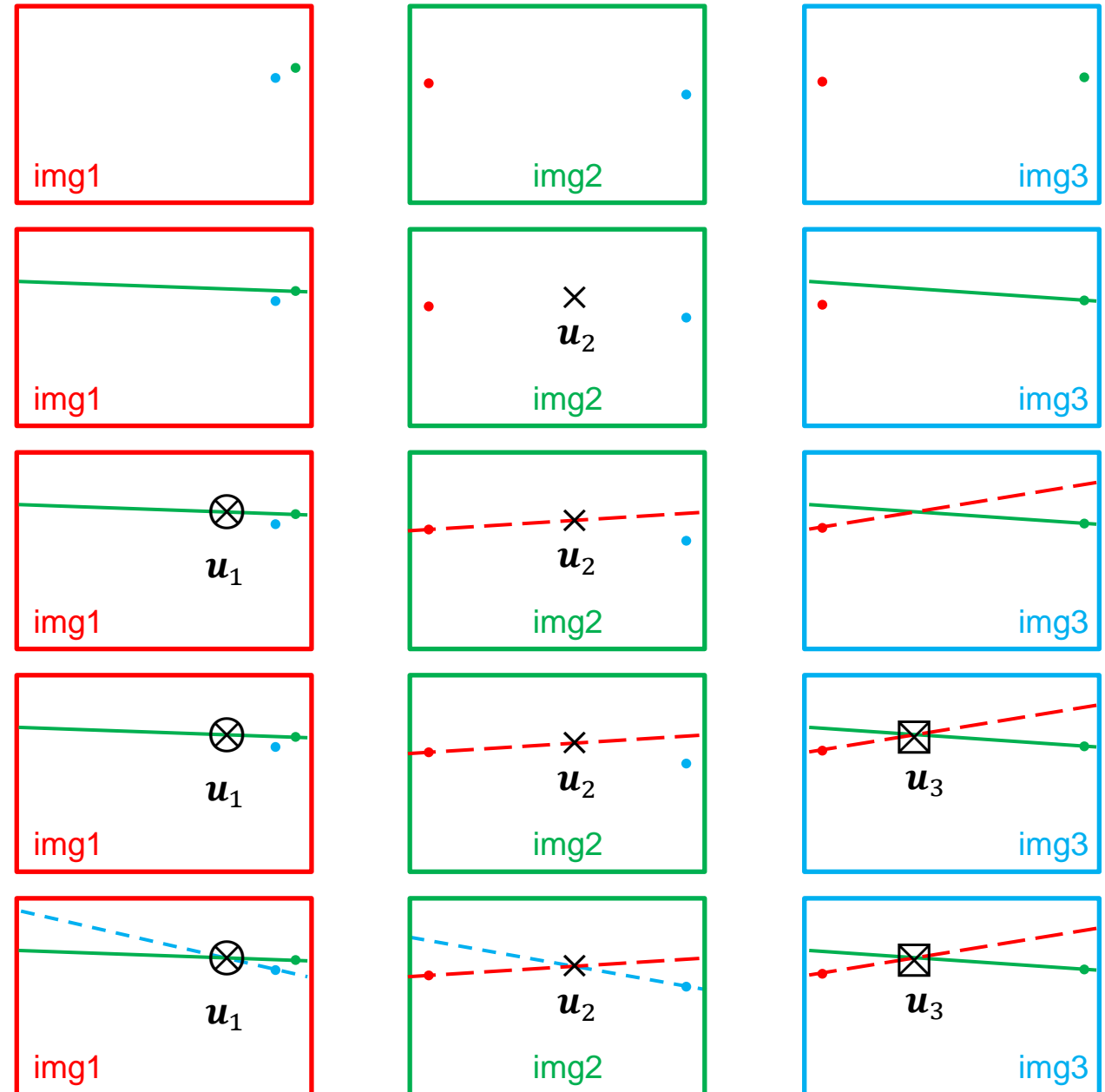
Epipoles

A point u_2 in img2
correspond to lines
in img1 and img3

A point u_1 in img1
correspond to lines
in img2 and img3

Points u_1 and u_2
define a point u_3 in
img3

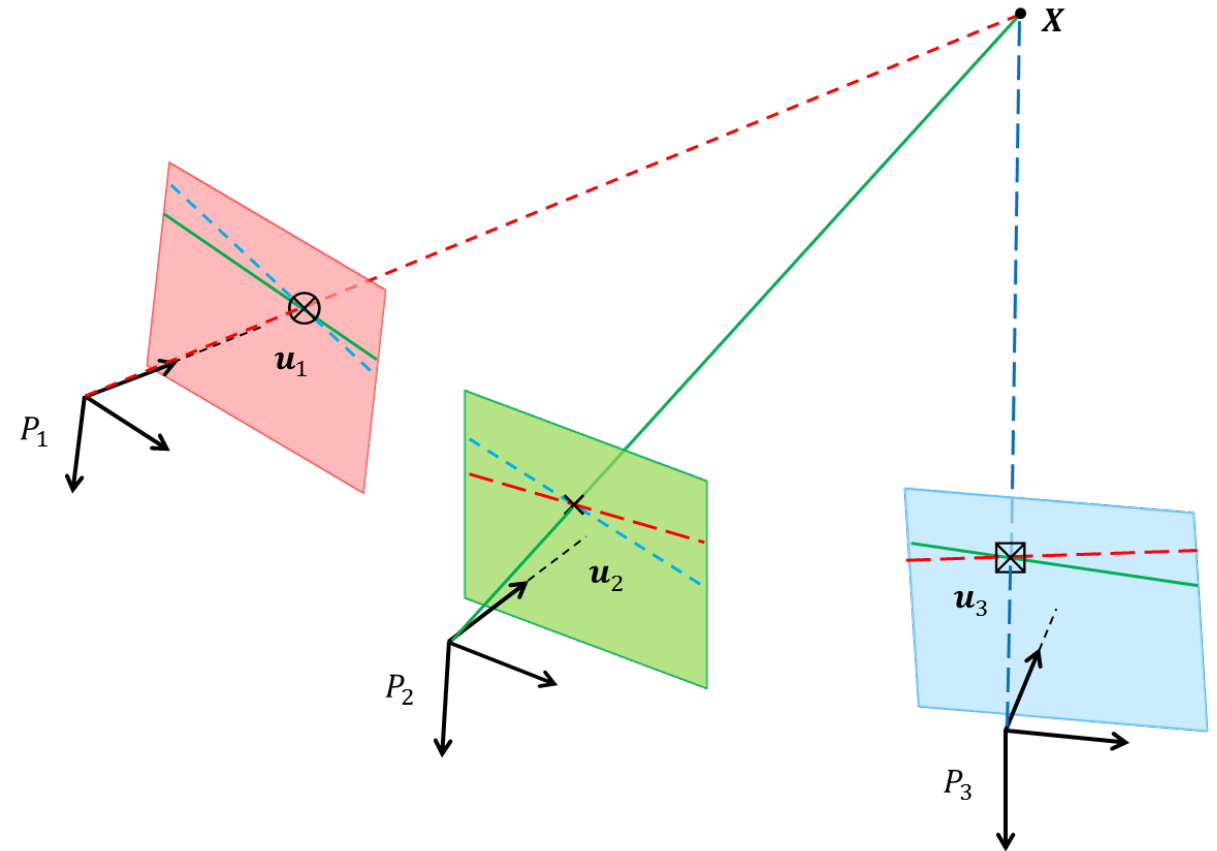
Any two of the
points $\{u_1, u_2, u_3\}$
define the third



Correspondences

Three views

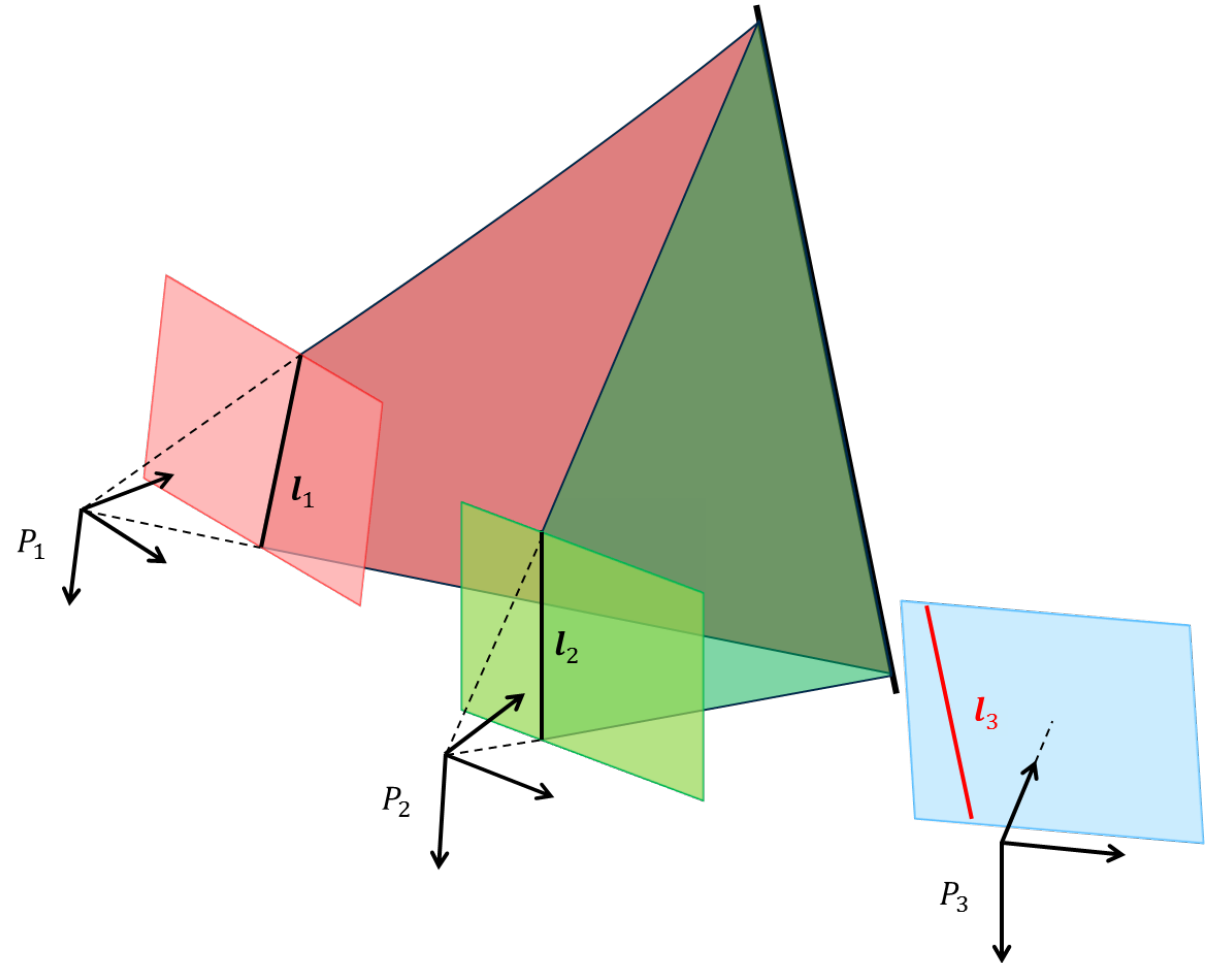
- This construction shows that the three points u_1 , u_2 and u_3 are connected by some geometric constraint
 - Any one of them can be computed from the two others
- But it is not clear if this three-view constraint governs more than the three epipolar constraints put together



Correspondences

Three views

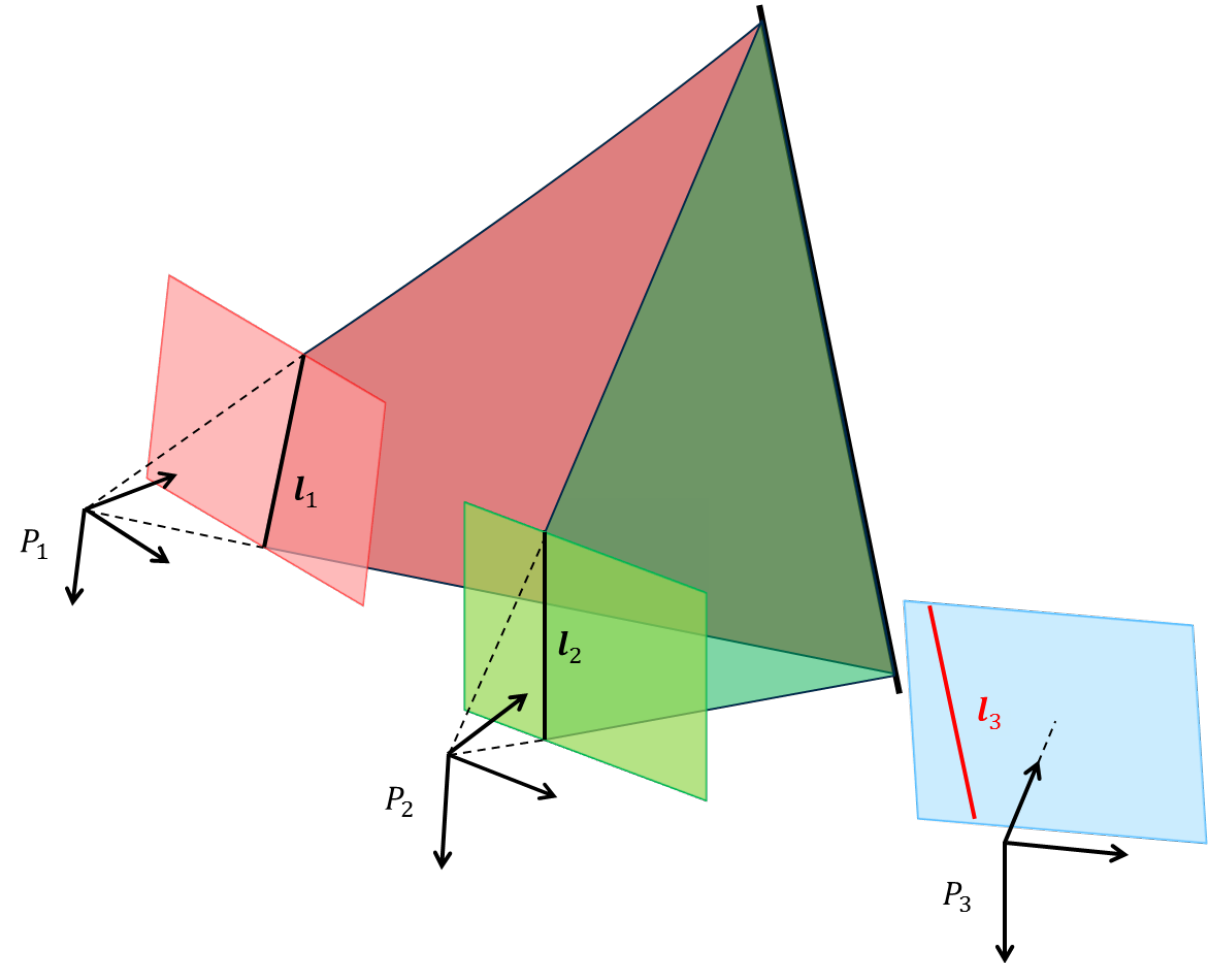
- The difference between two-view geometry and three-view geometry becomes evident if we consider lines instead of points
- In two-view geometry no constraints are available for lines
- In three-view geometry, lines l_1 and l_2 in two views will in general generate a line l_3 in a third view



Correspondences

Three views

- The three view geometry has an algebraic representation known as the trifocal tensor T
 - A $3 \times 3 \times 3$ array with 18dof
- This tensor governs the relationship between points and lines in three views
 - Point-point-point
 - Point-point-line
 - Point-line-line
 - Point-line-point
 - Line-line-line
- It may be used to transfer a two-view point/line correspondence into a point/line in a third view

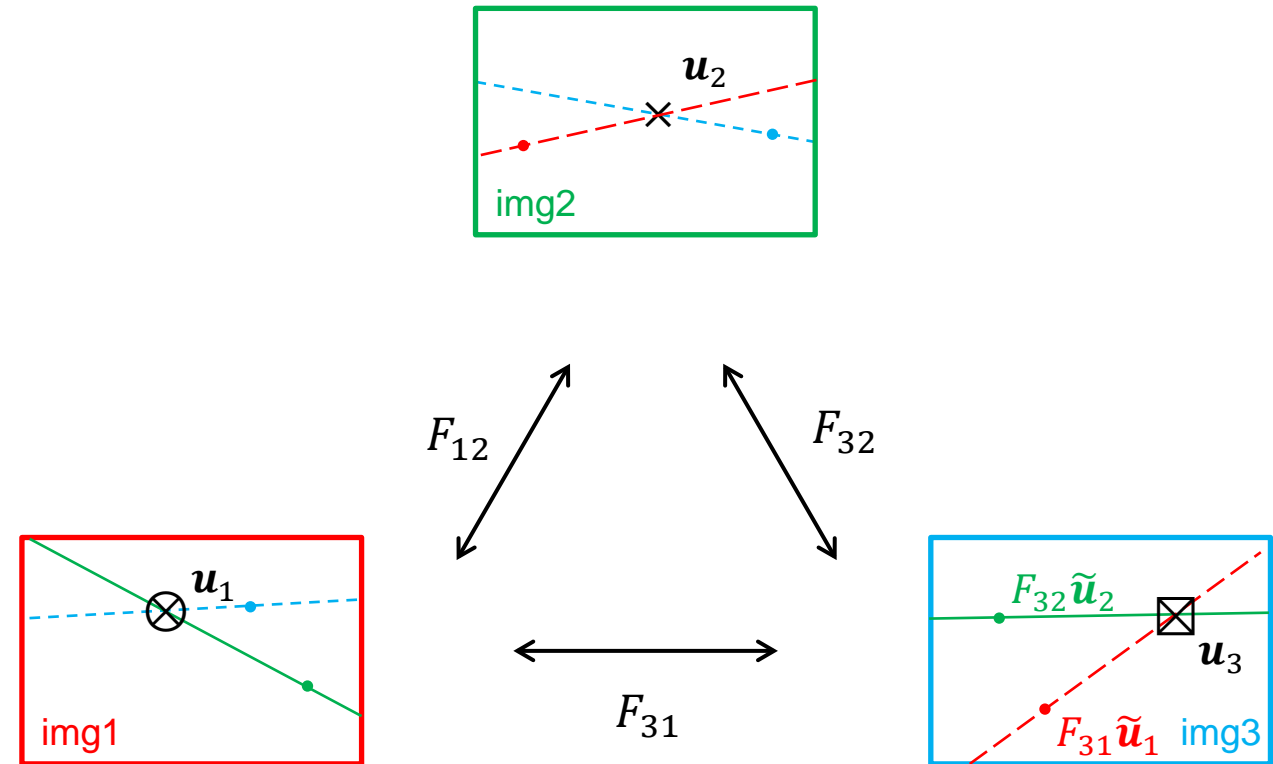


Correspondences

Three views

- As we just saw, point transfer can be done directly from the epipolar constraints

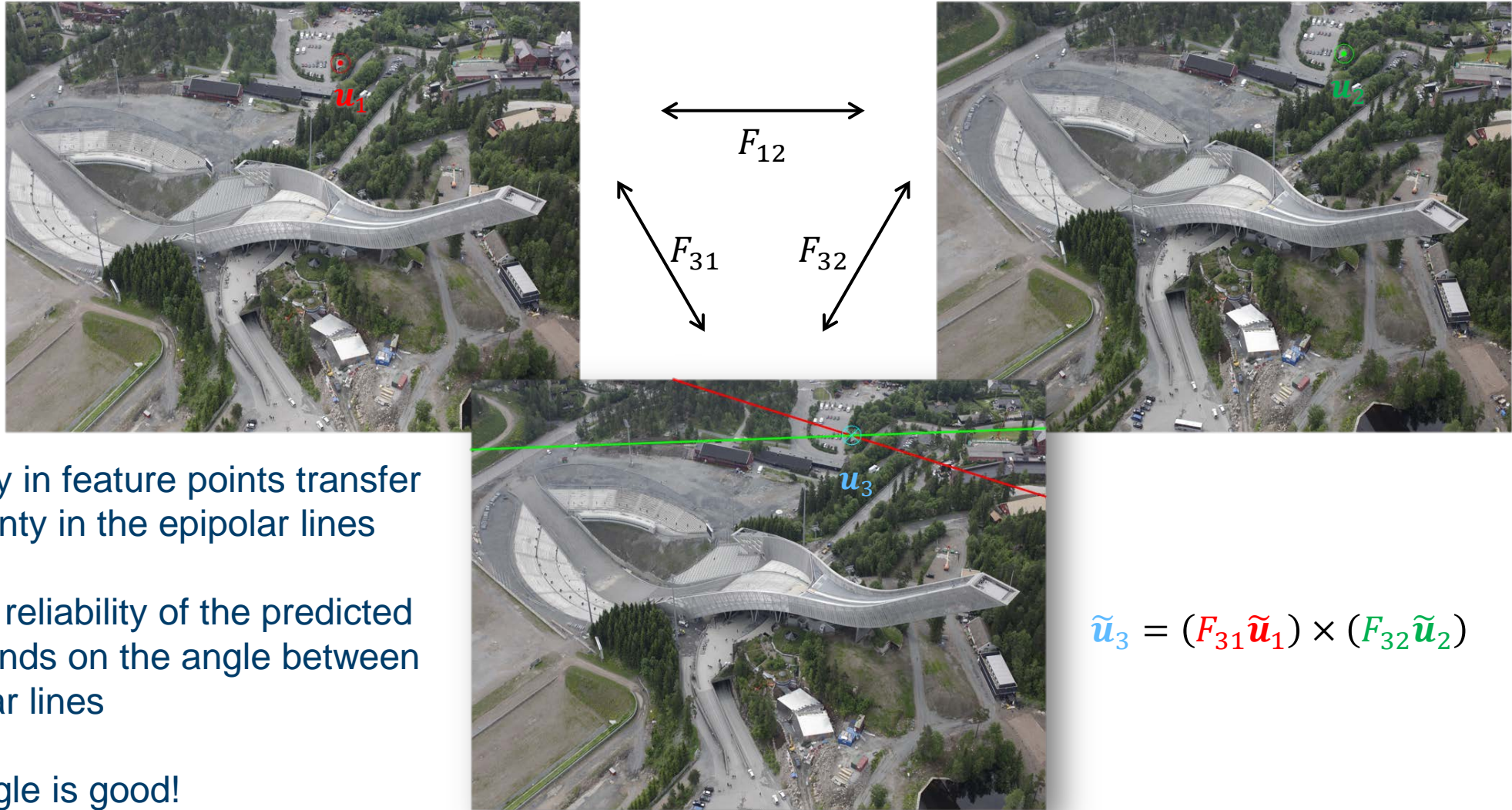
$$\tilde{\mathbf{u}}_3 = (F_{31}\tilde{\mathbf{u}}_1) \times (F_{32}\tilde{\mathbf{u}}_2)$$
- However, this fails for points in the plane defined by the three camera centers – the trifocal plane – since the epipolar lines then will coincide
- The trifocal tensor allows point transfer also for points in the trifocal plane



$$\tilde{\mathbf{u}}_3 = (F_{31}\tilde{\mathbf{u}}_1) \times (F_{32}\tilde{\mathbf{u}}_2)$$

Example

Point transfer based on epipolar constraints



Uncertainty in feature points transfer to uncertainty in the epipolar lines

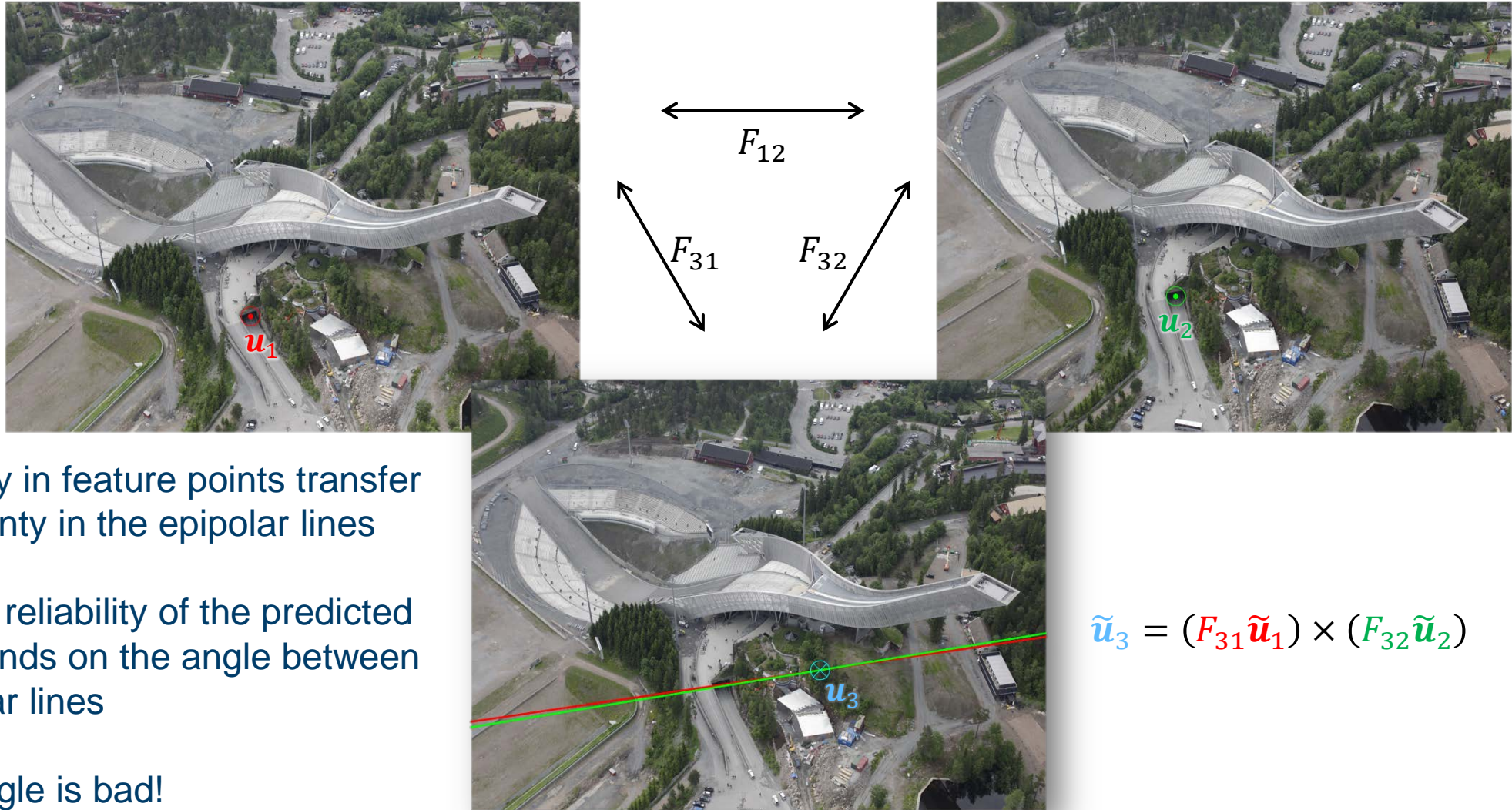
Hence the reliability of the predicted point depends on the angle between the epipolar lines

A large angle is good!

$$\tilde{u}_3 = (F_{31}\tilde{u}_1) \times (F_{32}\tilde{u}_2)$$

Example

Point transfer based on epipolar constraints



Uncertainty in feature points transfer
to uncertainty in the epipolar lines

Hence the reliability of the predicted
point depends on the angle between
the epipolar lines

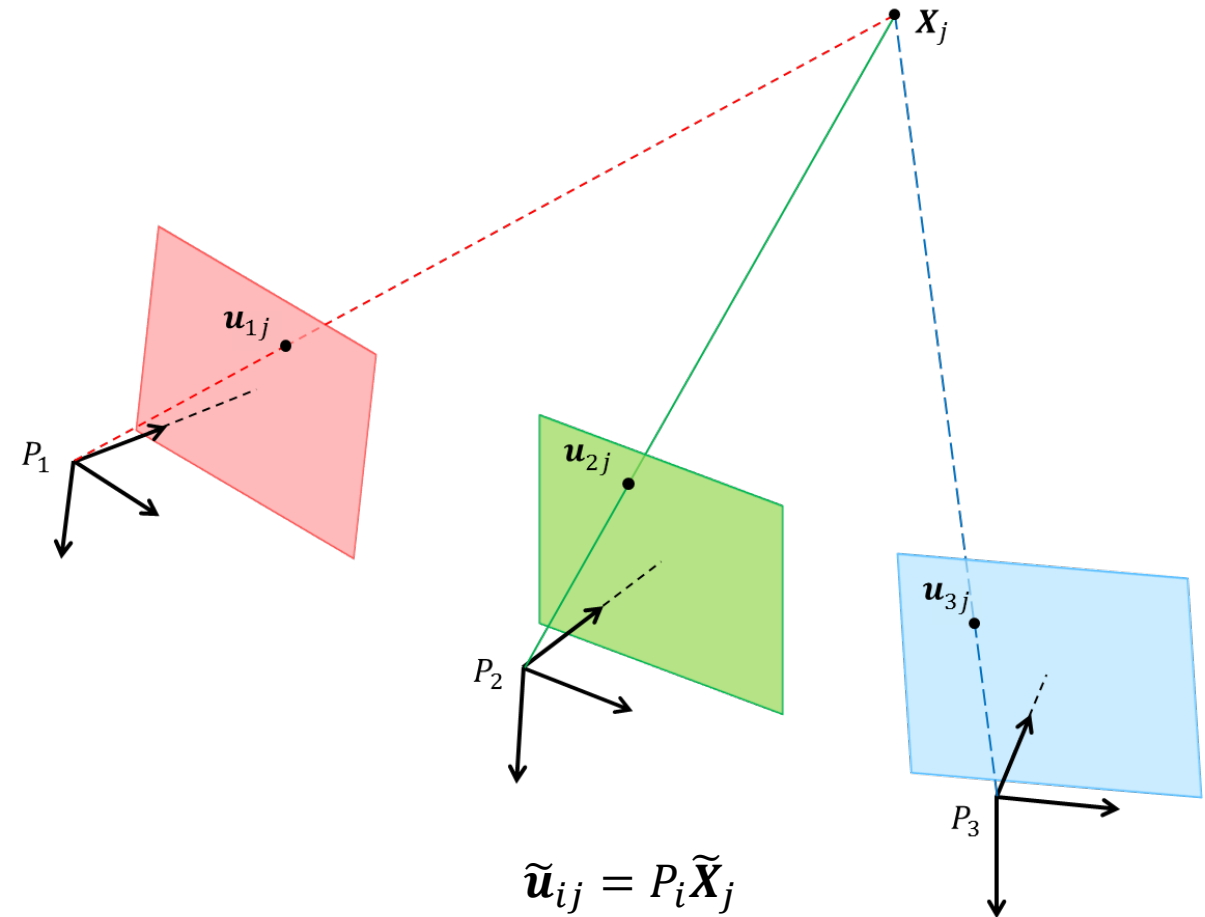
A small angle is bad!

$$\tilde{u}_3 = (F_{31} \tilde{u}_1) \times (F_{32} \tilde{u}_2)$$

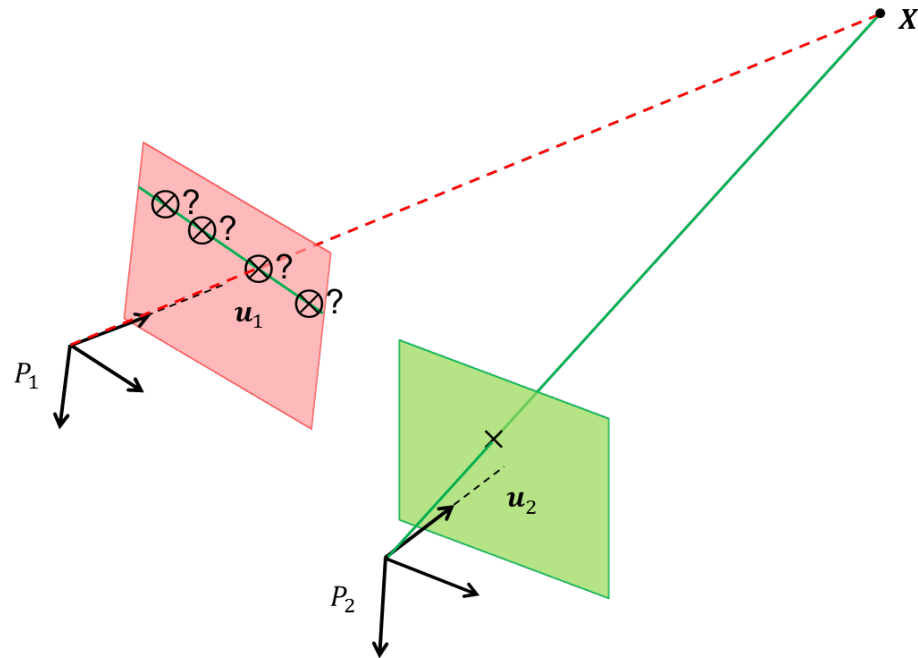
More-than-two-view geometry

Correspondences (matching)

- More views enables us to reveal and remove more mismatches than we can do in the two-view case
- More views also enables us to predict correspondences that can be tested with or without the use of descriptors
- Uncertainties in these predictions will in general decrease with the number of views (up to some point)



Summary



- Multiple-view geometry
- Correspondences
 - Two-view vs Three-view
 - Fundamental matrix vs Trifocal tensor

