UiO : Department of Technology Systems
University of Oslo

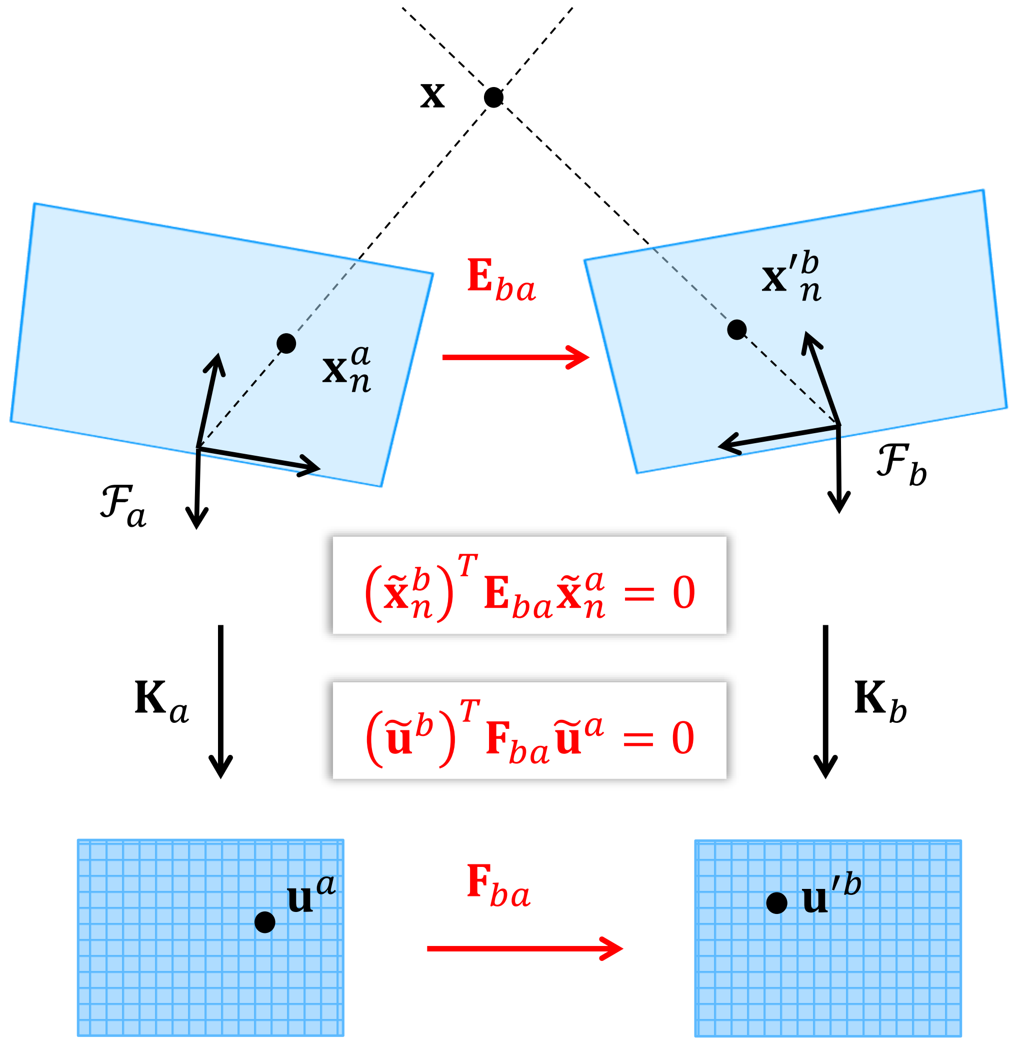
# Multiple-view geometry

Thomas Opsahl

2023

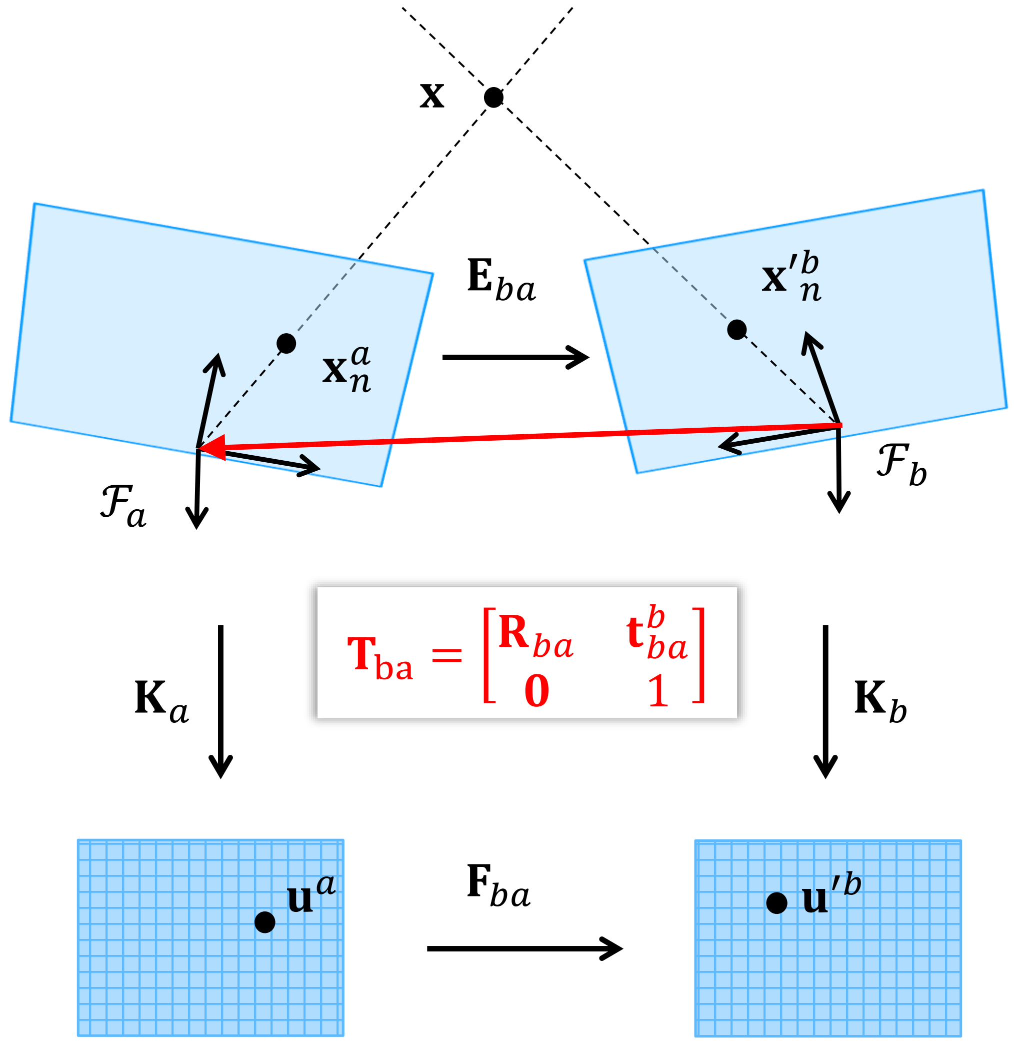
# Recap on two-view geometry

- Epipolar geometry
  - The essential matrix $\mathbf{E}_{ba} = [\mathbf{t}_{ba}^{b}]_{\times}\mathbf{R}_{ba}$
    Estimate from 5 or more correspondences $\mathbf{x}_n^a \leftrightarrow \mathbf{x}'^b_n$
  - The fundamental matrix $\mathbf{F}_{ba} = \mathbf{K}_b^{-T}\mathbf{E}_{ba}\mathbf{K}_a^{-1}$
    Estimate from 7 or more correspondences $\mathbf{u}^a \leftrightarrow \mathbf{u}'^b$



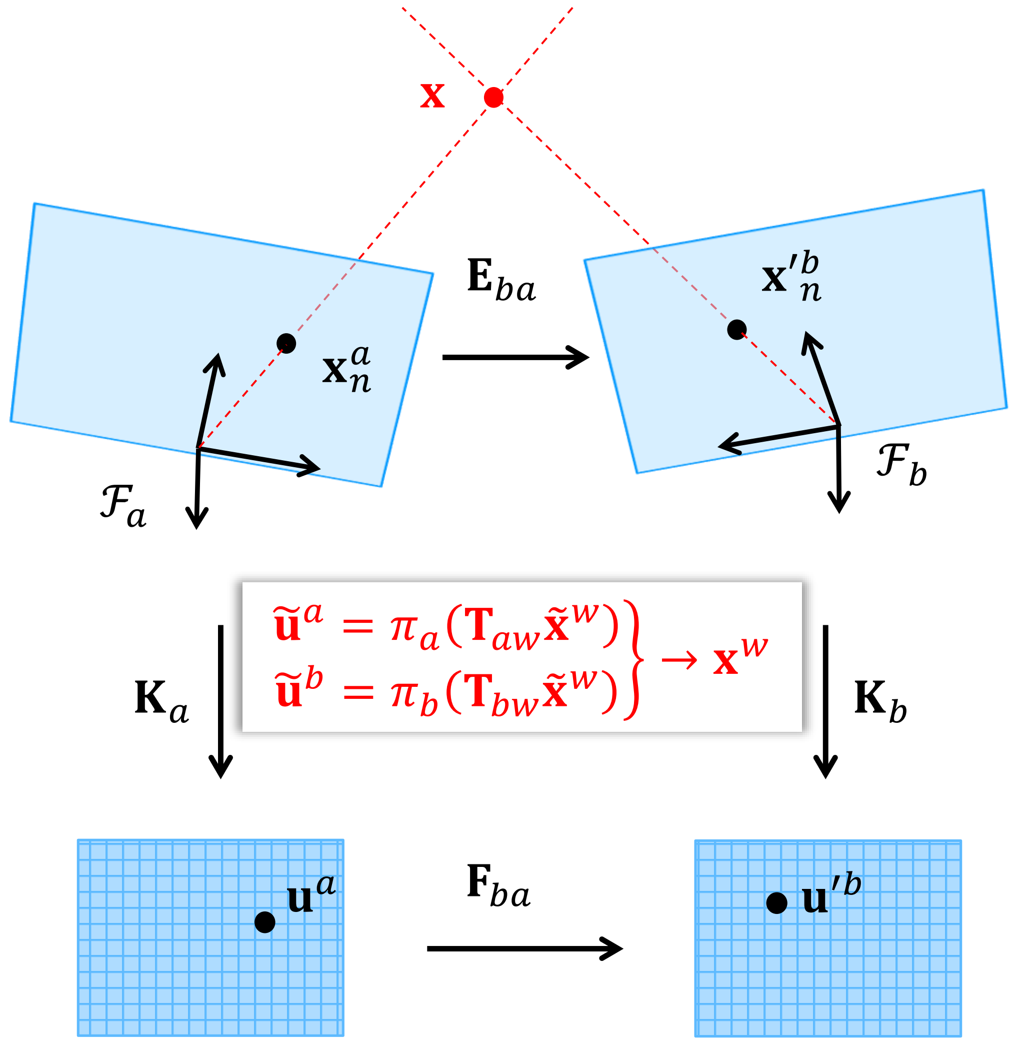
# Recap on two-view geometry

- Epipolar geometry
  - The essential matrix $\mathbf{E}_{ba} = [\mathbf{t}_{ba}^{b}]_{\times}\mathbf{R}_{ba}$
    Estimate from 5 or more correspondences $\mathbf{x}_n^a \leftrightarrow \mathbf{x}'^b_n$
  - The fundamental matrix $\mathbf{F}_{ba} = \mathbf{K}_b^{-T}\mathbf{E}_{ba}\mathbf{K}_a^{-1}$
    Estimate from 7 or more correspondences $\mathbf{u}^a \leftrightarrow \mathbf{u}'^b$
- Pose from epipolar geometry
  - Decompose $\mathbf{E}_{ba}$ into $\mathbf{R}_{ba}$ and $\mathbf{t}_{ba}^{b}$ (up to scale)



# Recap on two-view geometry

- Epipolar geometry
  - The essential matrix $\mathbf{E}_{ba} = \left[\mathbf{t}_{ba}^{b}\right]_{\times} \mathbf{R}_{ba}$
    Estimate from 5 or more correspondences $\mathbf{x}_n^a \leftrightarrow \mathbf{x'}_n^b$
  - The fundamental matrix $\mathbf{F}_{ba} = \mathbf{K}_b^{-T} \mathbf{E}_{ba} \mathbf{K}_a^{-1}$
    Estimate from 7 or more correspondences $\mathbf{u}^a \leftrightarrow \mathbf{u'}^b$
- Pose from epipolar geometry
  - Decompose $\mathbf{E}_{ba}$ into $\mathbf{R}_{ba}$ and $\mathbf{t}_{ba}^{b}$ (up to scale)
- 3D structure from epipolar geometry
  - Triangulation by minimizing algebraic error
    $$\left.\begin{aligned} \tilde{\mathbf{u}}^a &= \mathbf{K}_a \left[\mathbf{R}_{aw} \quad \mathbf{t}_{aw}^{a}\right] \tilde{\mathbf{x}}^w \\ \tilde{\mathbf{u}}'^b &= \mathbf{K}_b \left[\mathbf{R}_{bw} \quad \mathbf{t}_{bw}^{b}\right] \tilde{\mathbf{x}}^w \end{aligned}\right\} \rightarrow \mathbf{A}\mathbf{x}^w = 0 \xrightarrow{SVD} \mathbf{x}^w$$
  - Triangulation by minimizing reprojection error
    $$\varepsilon = \left\|\pi_a(\mathbf{T}_{aw} \cdot \mathbf{x}^w) - \mathbf{u}^a\right\|^2 + \left\|\pi_b(\mathbf{T}_{bw} \cdot \mathbf{x}^w) - \mathbf{u'}^b\right\|^2$$

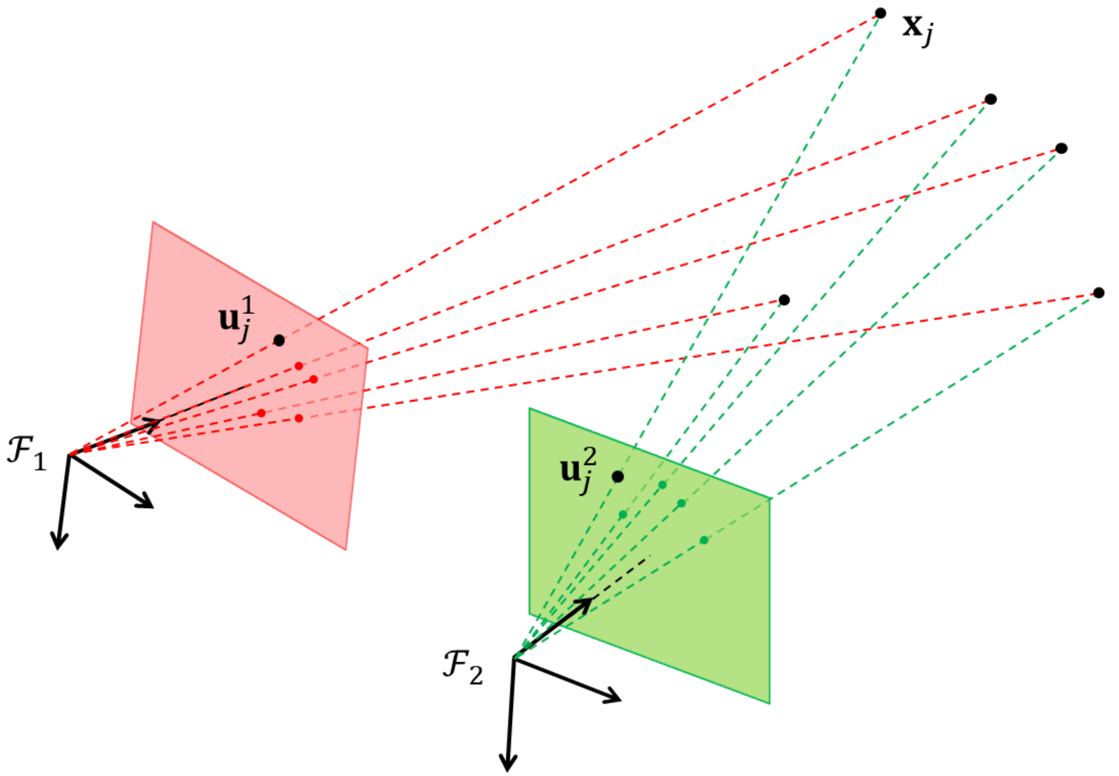


# Recap on two-view geometry

Scene geometry

Pixel correspondences

Camera geometry



# Recap on two-view geometry

### Pixel correspondences (matching)

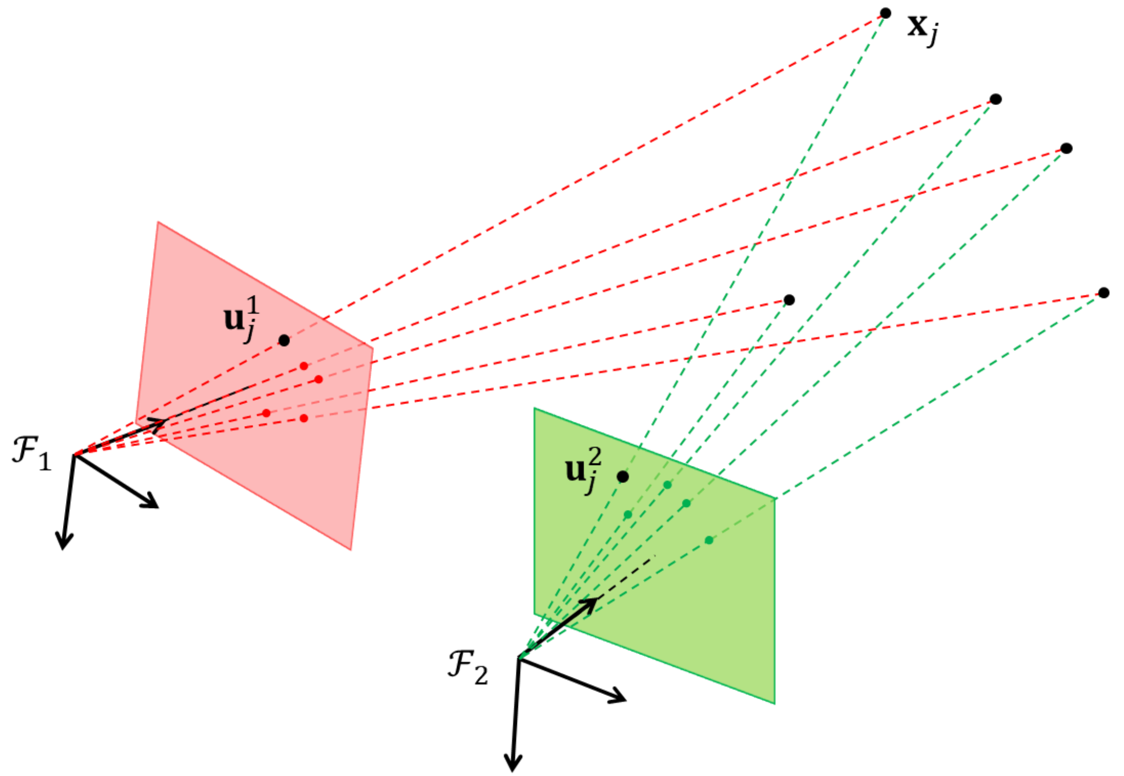
- Correspondences must satisfy the epipolar constraint
- Useful for reducing the number of mismatches
- Useful when searching for correspondences

### Scene geometry (structure)

- Sparse scene geometry from triangulating correspondences
- Refine result by performing BA
- Dense scene geometry from stereo processing

### Camera geometry (motion)

- Camera poses must satisfy the epipolar constraint
- We can decompose the essential matrix to find the relative pose between cameras
- Refine result by performing BA



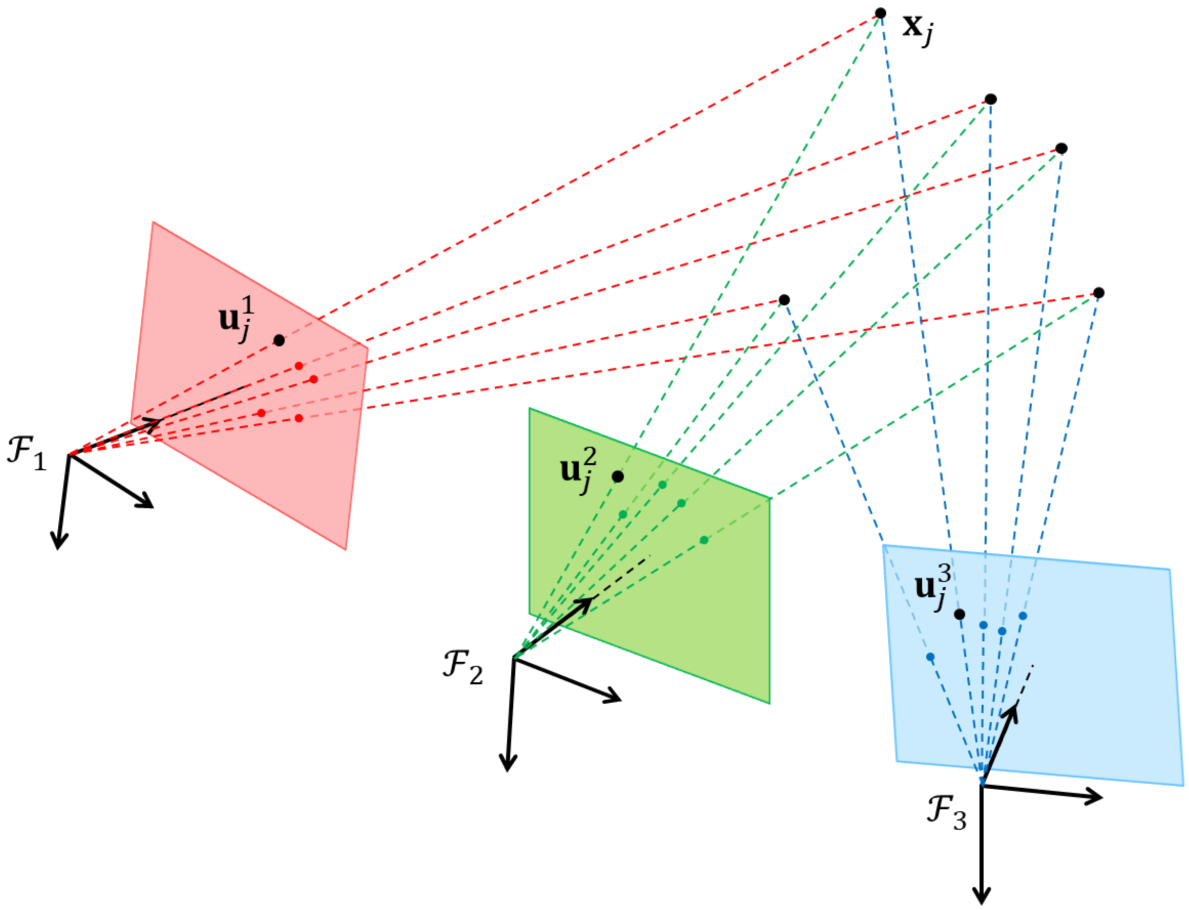
# Multi-view geometry

Scene geometry

Image correspondences

pixel correspondences

Camera geometry



# Multi-view geometry

## Pixel correspondences (matching)

- How does multi-view geometry constrain our pixel correspondences?

## Image correspondences

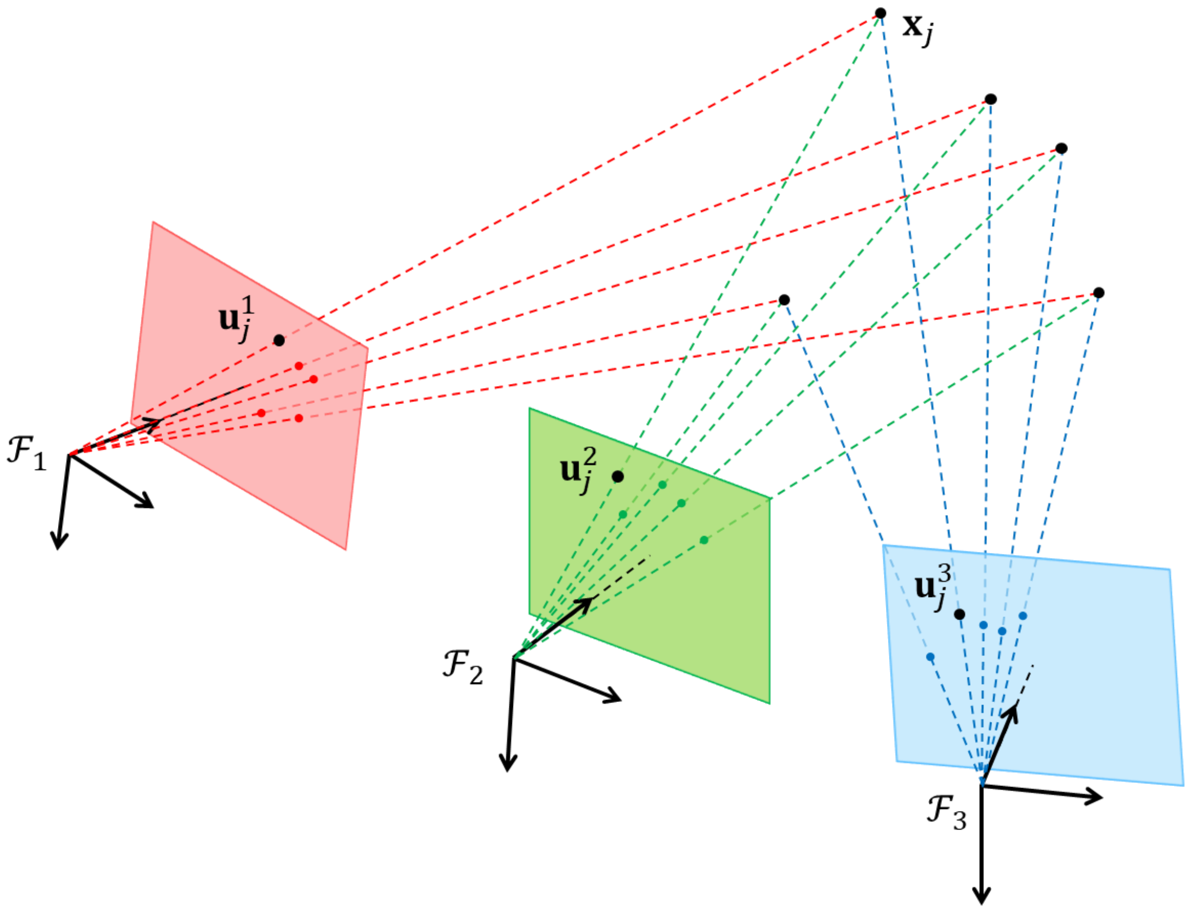
- How do we deal with the problem of images not necessarily overlapping?

## Scene geometry (structure)

- How does multi-view geometry impact the ability to estimate the 3D structure of the scene?

## Camera geometry (motion)

- How does multi-view geometry impact the ability to estimate camera geometry?

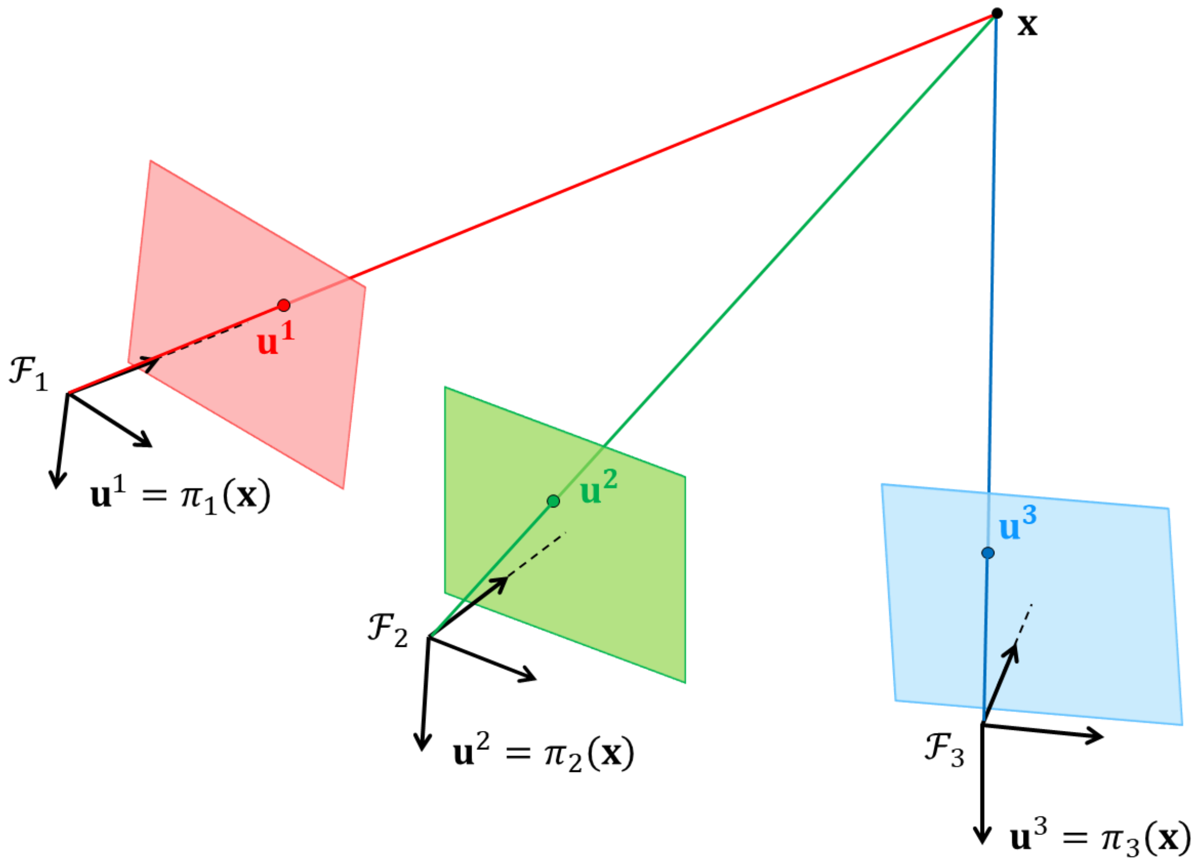


# Multi-view pixel correspondences

**Three-view**

- Three cameras observe the same point $\mathbf{x}$
- We have three two-view geometries

$$(\tilde{\mathbf{u}}^2)^T \mathbf{F}_{21} \tilde{\mathbf{u}}^1 = 0$$
$$(\tilde{\mathbf{u}}^3)^T \mathbf{F}_{31} \tilde{\mathbf{u}}^1 = 0$$
$$(\tilde{\mathbf{u}}^3)^T \mathbf{F}_{32} \tilde{\mathbf{u}}^2 = 0$$



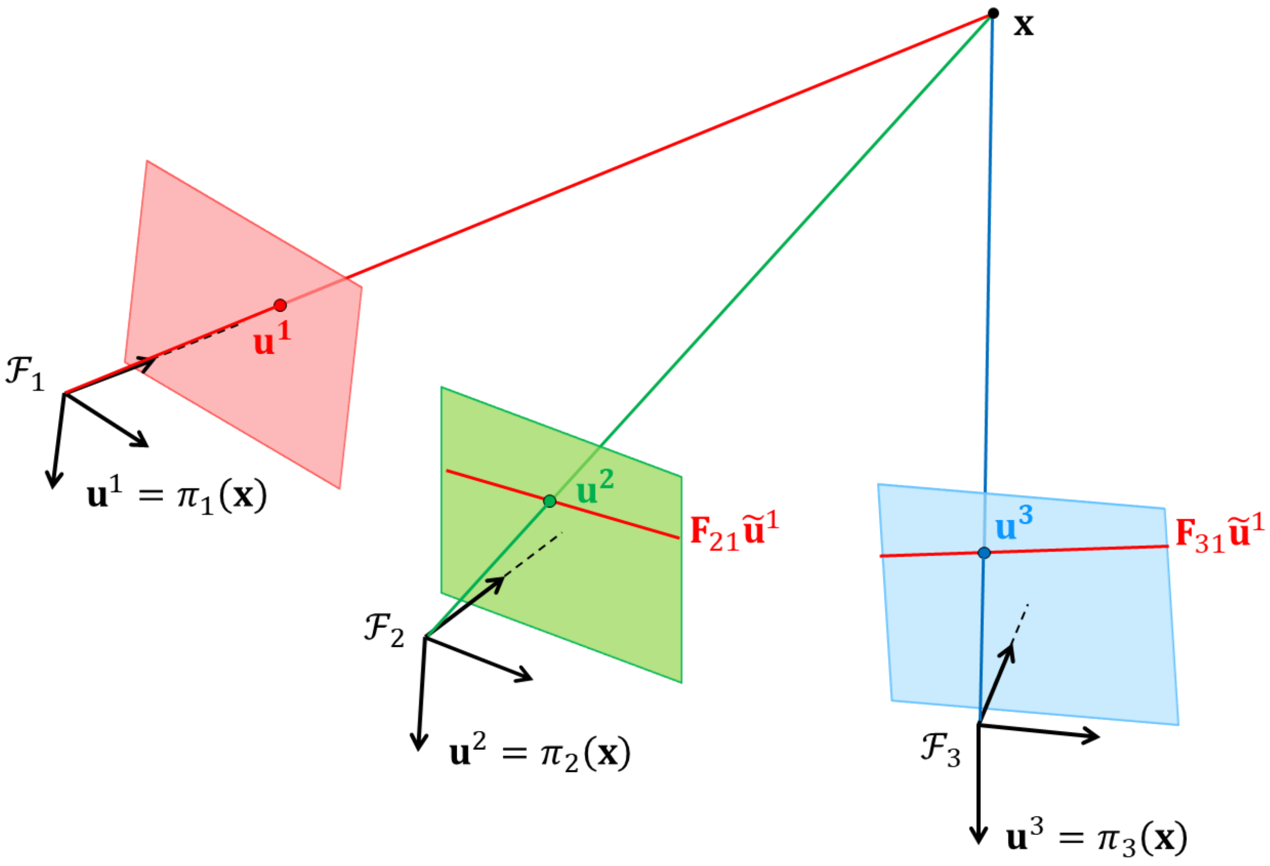
# Multi-view pixel correspondences

**Three-view**

- Three cameras observe the same point $\mathbf{x}$
- We have three two-view geometries

$$(\tilde{\mathbf{u}}^2)^T \mathbf{F}_{21} \tilde{\mathbf{u}}^1 = 0$$
$$(\tilde{\mathbf{u}}^3)^T \mathbf{F}_{31} \tilde{\mathbf{u}}^1 = 0$$
$$(\tilde{\mathbf{u}}^3)^T \mathbf{F}_{32} \tilde{\mathbf{u}}^2 = 0$$

- Each point corresponds to two epipolar lines



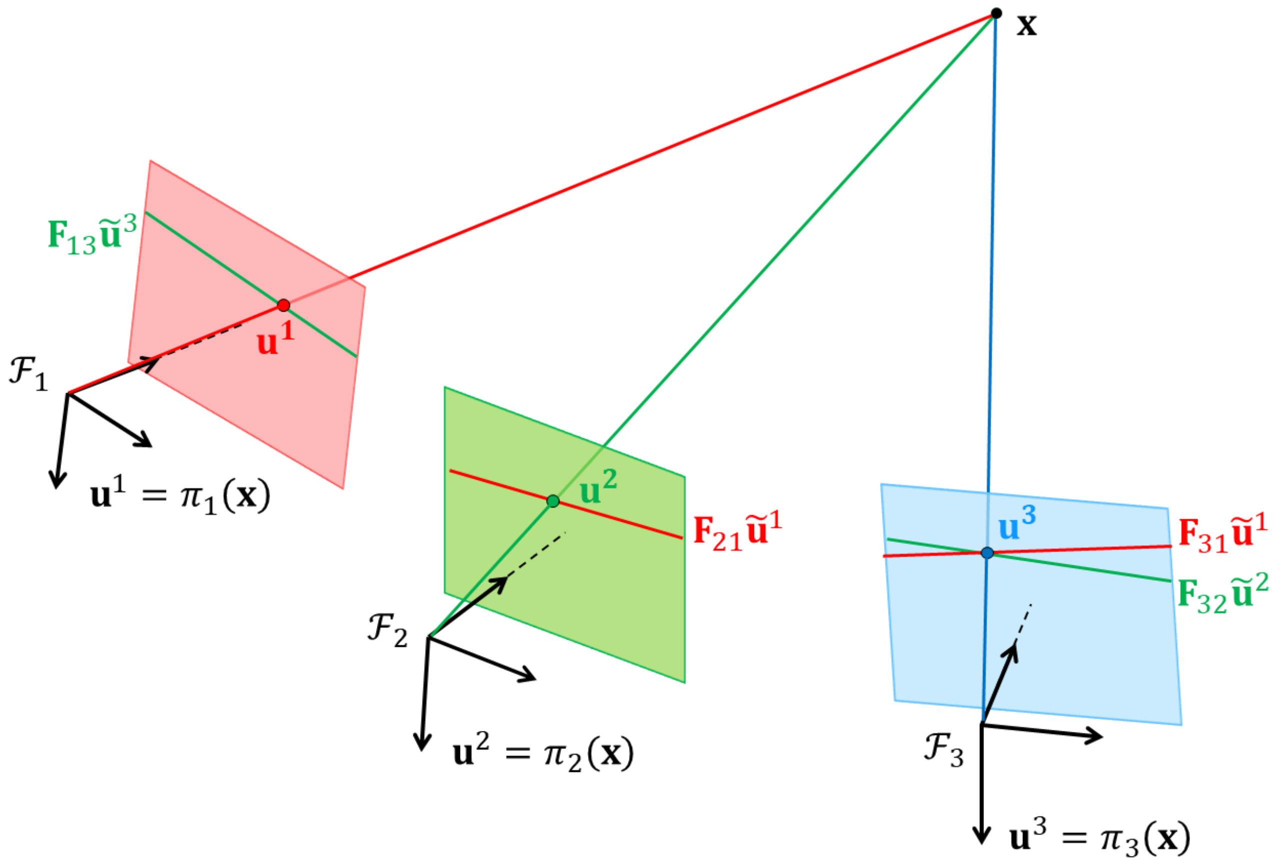
# Multi-view pixel correspondences

**Three-view**

- Three cameras observe the same point $\mathbf{x}$
- We have three two-view geometries

$$(\tilde{\mathbf{u}}^2)^T \mathbf{F}_{21} \tilde{\mathbf{u}}^1 = 0$$
$$(\tilde{\mathbf{u}}^3)^T \mathbf{F}_{31} \tilde{\mathbf{u}}^1 = 0$$
$$(\tilde{\mathbf{u}}^3)^T \mathbf{F}_{32} \tilde{\mathbf{u}}^2 = 0$$

- Each point corresponds to two epipolar lines



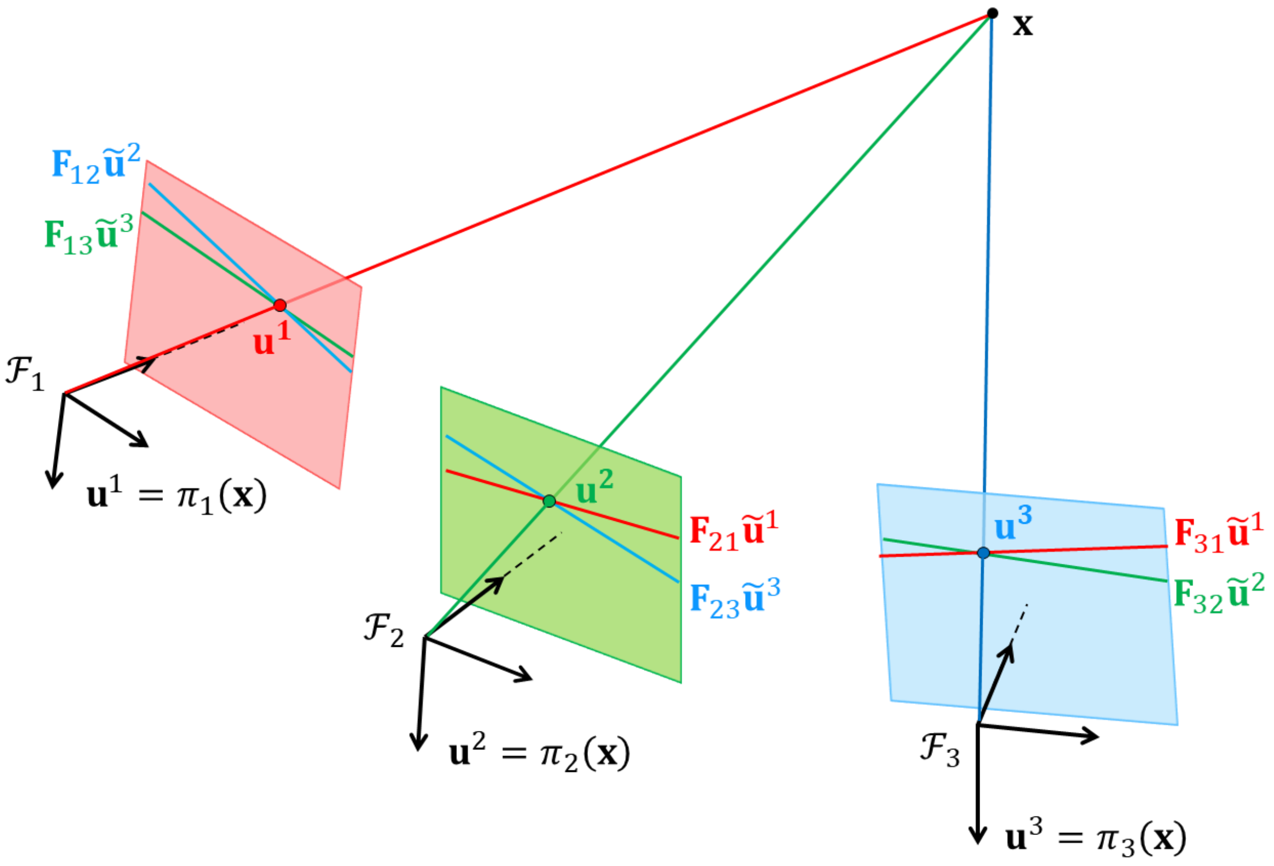
# Multi-view pixel correspondences

**Three-view**

- Three cameras observe the same point $\mathbf{x}$
- We have three two-view geometries

$$(\tilde{\mathbf{u}}^2)^T \mathbf{F}_{21} \tilde{\mathbf{u}}^1 = 0$$
$$(\tilde{\mathbf{u}}^3)^T \mathbf{F}_{31} \tilde{\mathbf{u}}^1 = 0$$
$$(\tilde{\mathbf{u}}^3)^T \mathbf{F}_{32} \tilde{\mathbf{u}}^2 = 0$$

- Each point corresponds to two epipolar lines



# Multi-view pixel correspondences

**Three-view**

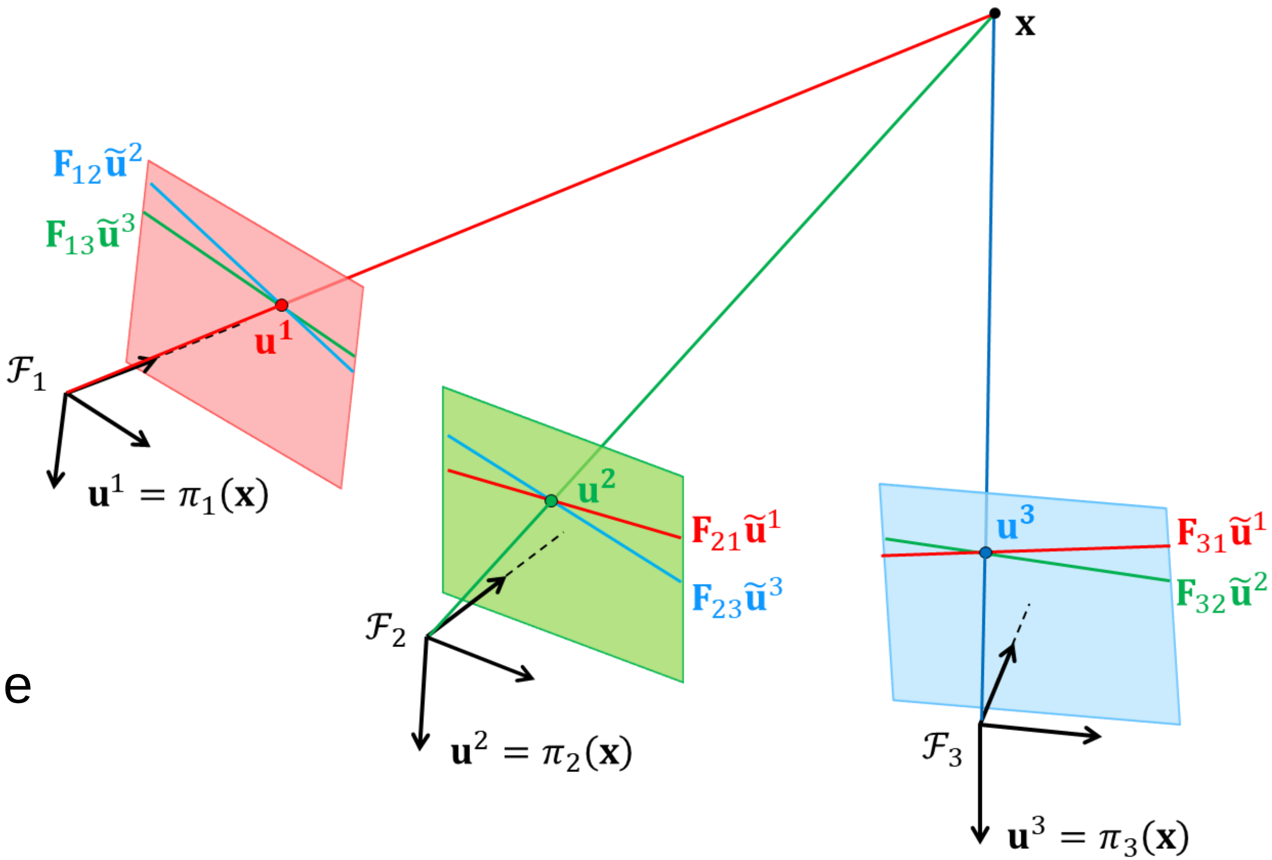
- Three cameras observe the same point $\mathbf{x}$
- We have three two-view geometries

$$(\tilde{\mathbf{u}}^2)^T \mathbf{F}_{21} \tilde{\mathbf{u}}^1 = 0$$
$$(\tilde{\mathbf{u}}^3)^T \mathbf{F}_{31} \tilde{\mathbf{u}}^1 = 0$$
$$(\tilde{\mathbf{u}}^3)^T \mathbf{F}_{32} \tilde{\mathbf{u}}^2 = 0$$

- Each point corresponds to two epipolar lines

- We observe that we can estimate one of the points directly from the two others

$$\boxed{\tilde{\mathbf{u}}^3 = (\mathbf{F}_{31} \tilde{\mathbf{u}}^1) \times (\mathbf{F}_{32} \tilde{\mathbf{u}}^2)}$$

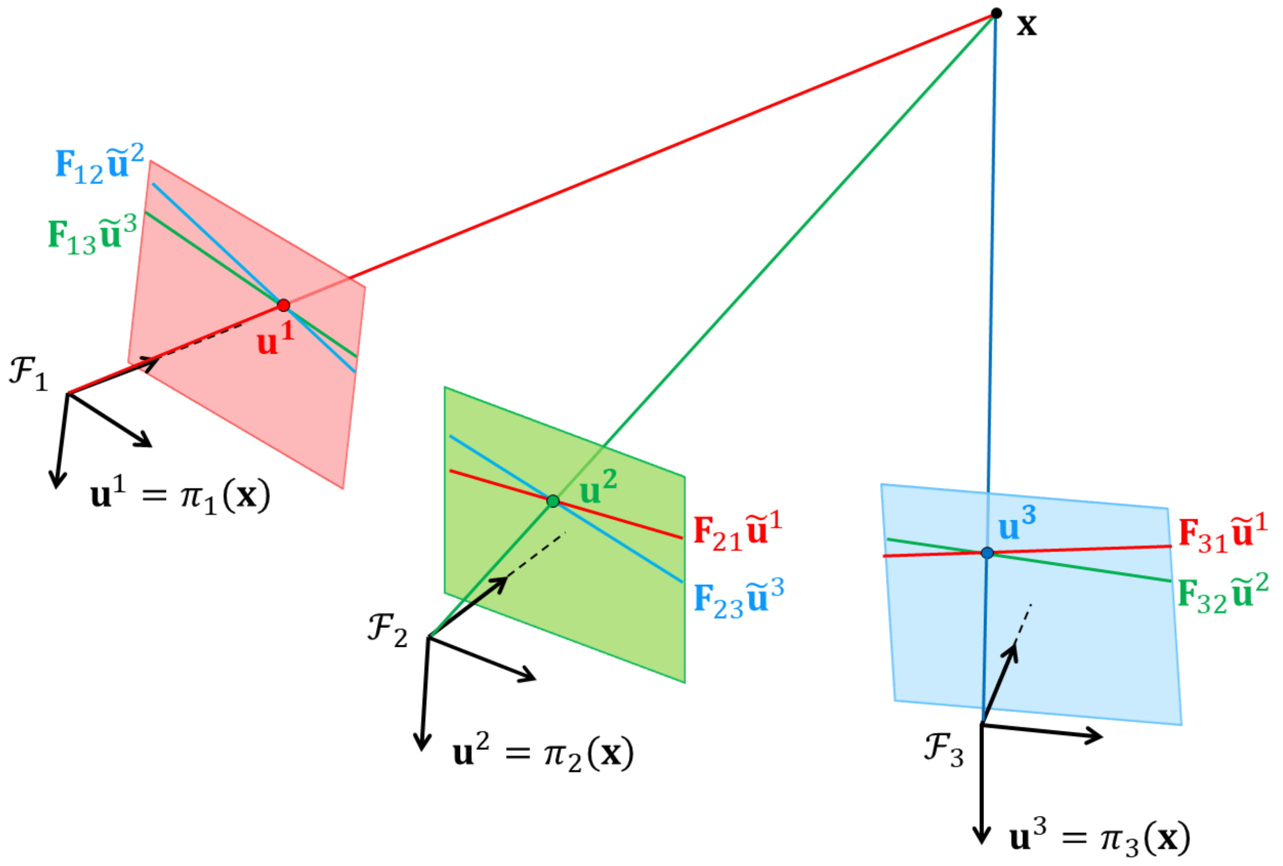


# Multi-view pixel correspondences

**Three-view**

- This observation indicates that the points $\mathbf{u}^1$, $\mathbf{u}^2$ and $\mathbf{u}^3$ are connected by some geometric constraint

- But it is not clear if this three-view constraint governs more than the three two-view constraints combined



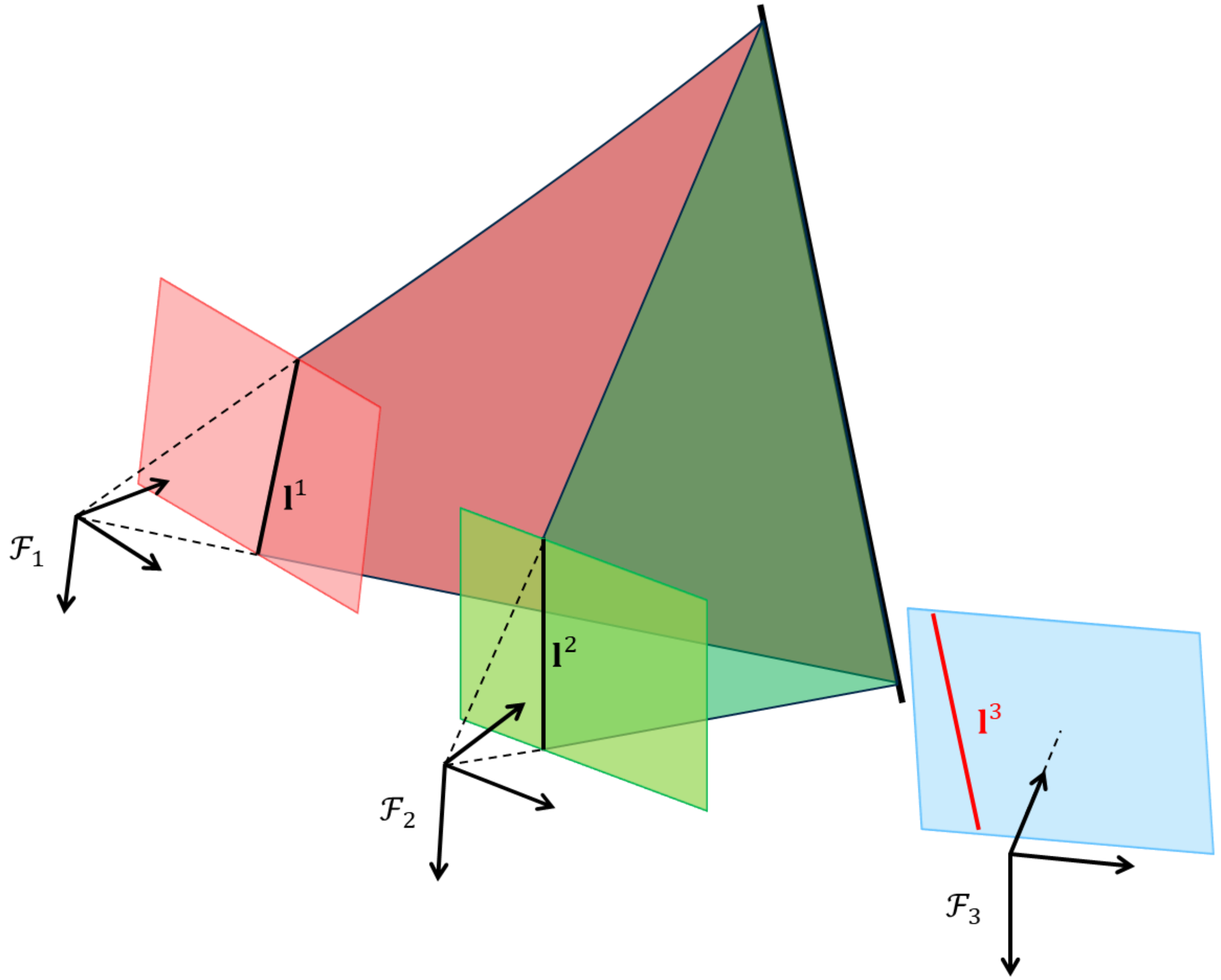
# Multi-view pixel correspondences

**Three-view**

- 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 $\mathbf{l}^1$ and $\mathbf{l}^2$ in two views will in general generate a line $\mathbf{l}^3$ in a third view

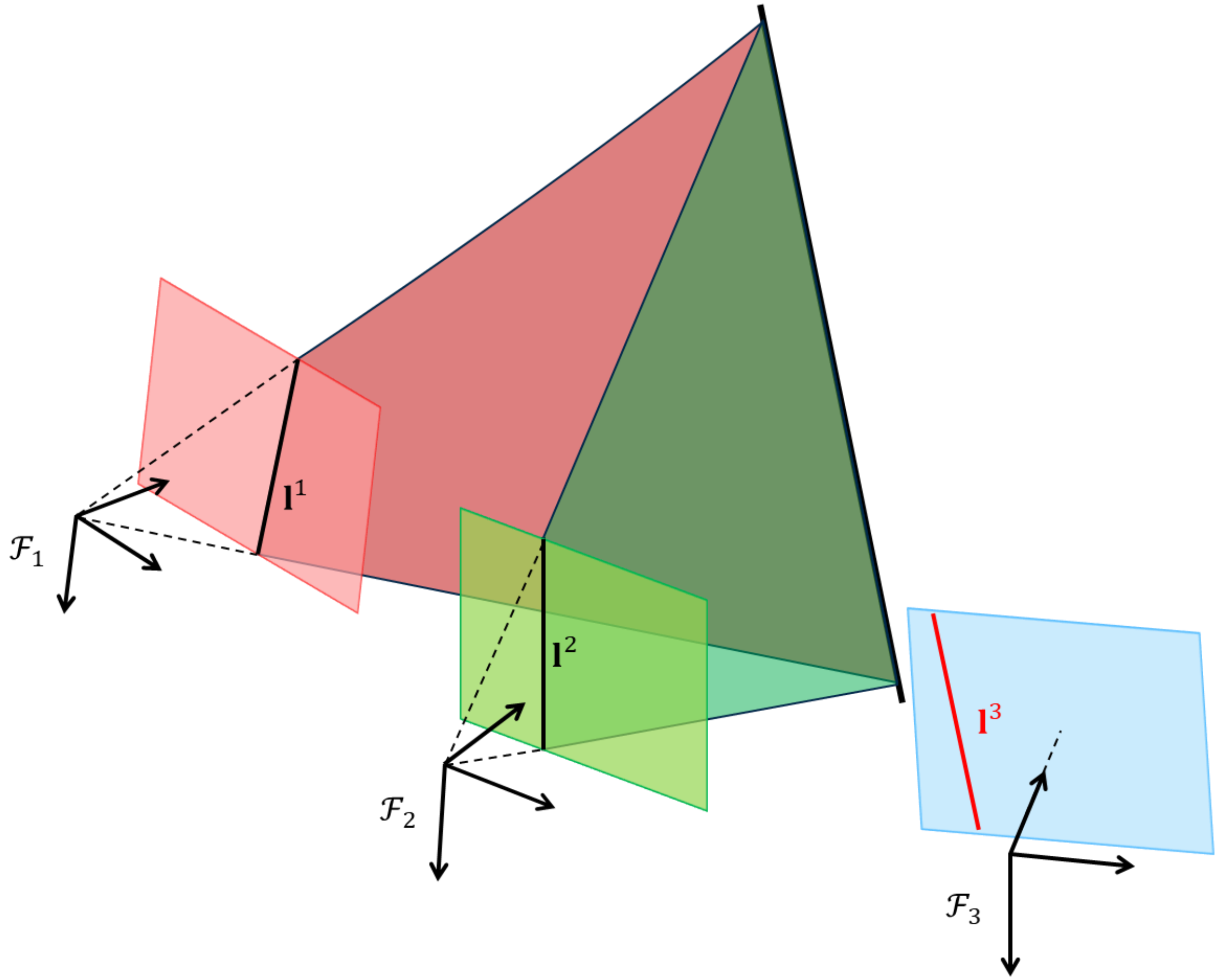


# Multi-view pixel correspondences

**Three-view**

- The three view geometry has an algebraic representation known as the trifocal tensor
  - $3 \times 3 \times 3$ array with 18dof

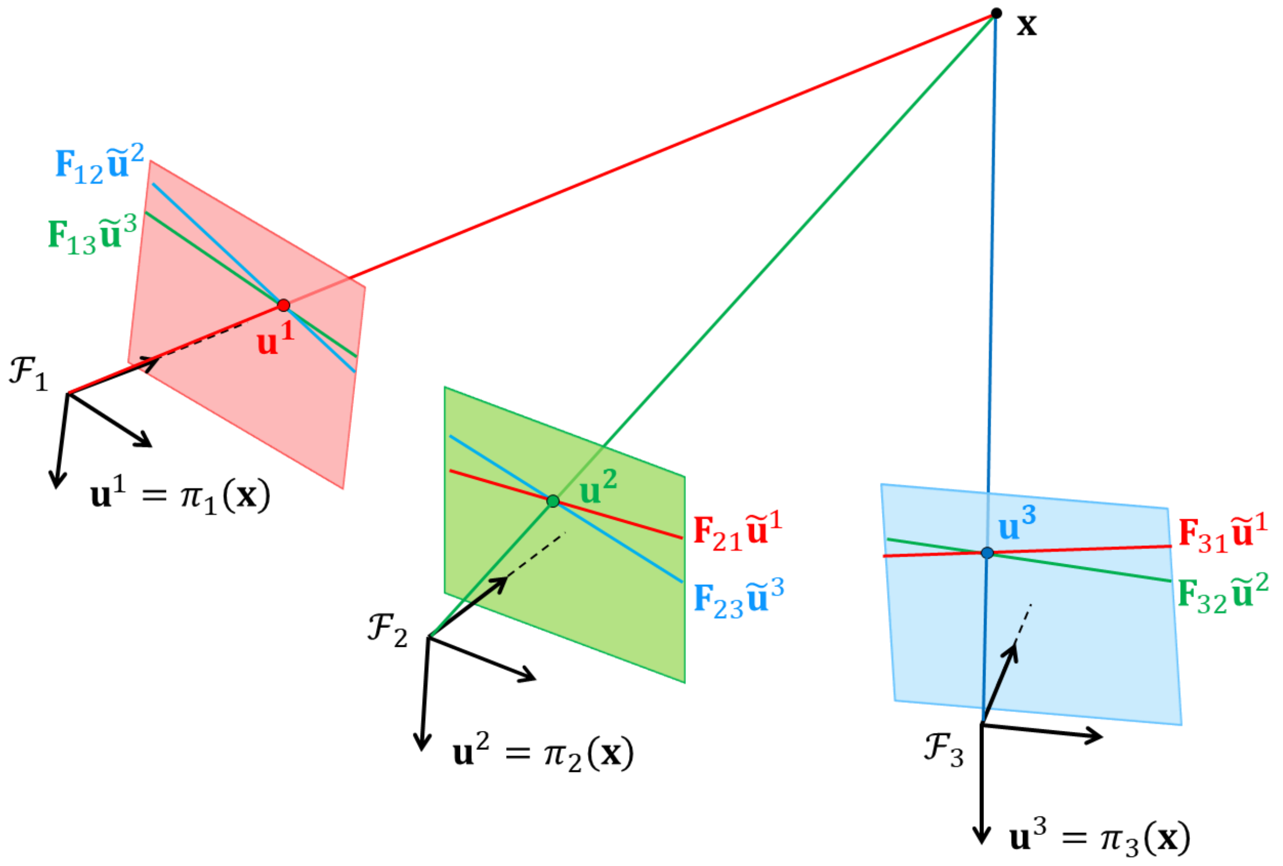
- Describes the relationship between
  - Point-point-point
  - Point-point-line
  - Point-line-line
  - Point-line-point
  - Line-line-line



# Multi-view pixel correspondences

## Multi-view constraints

- Two-view constraint: Fundamental matrix (homography for planar cases)
- Three-view constraint: Trifocal tensor
- Four-view constraint: Quadrifocal tensor
- After that it gets complicated...



# Multi-view pixel correspondences

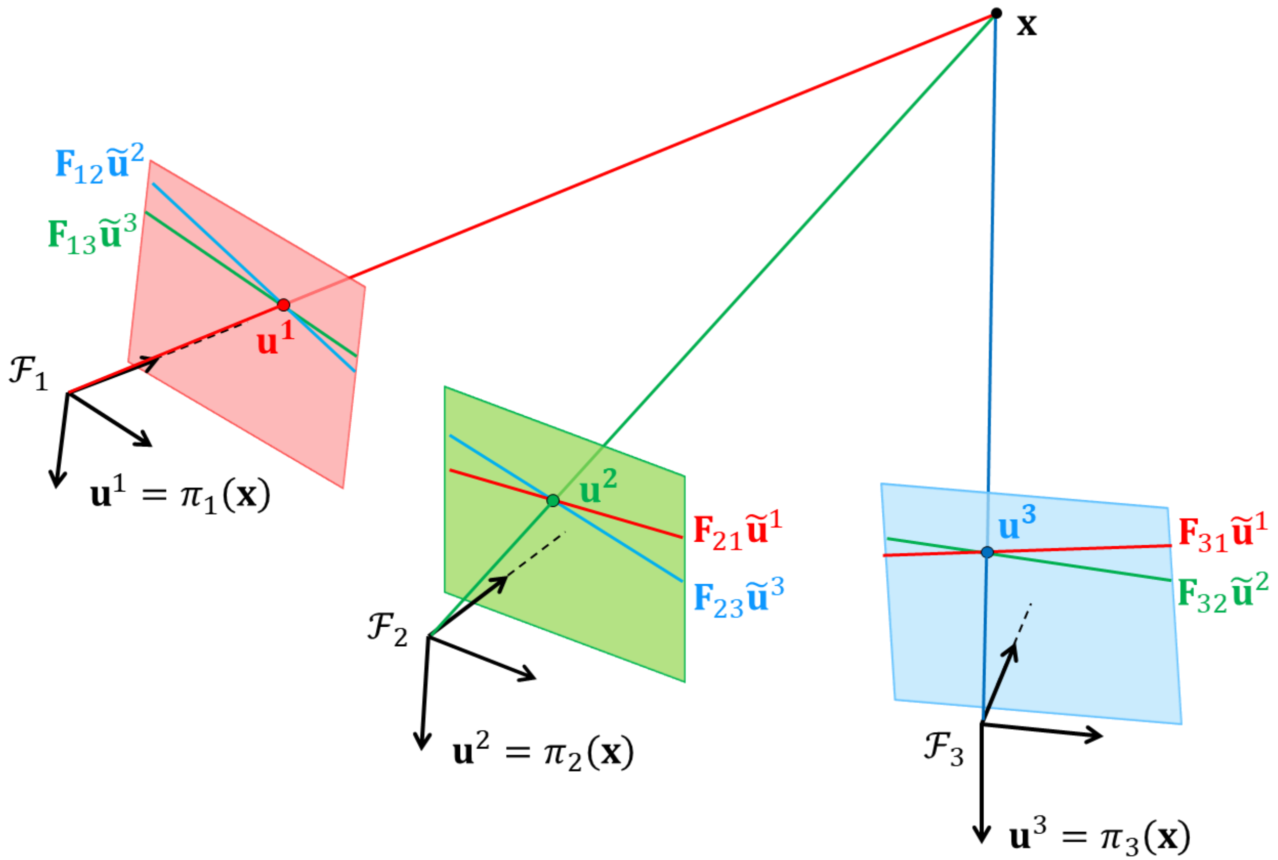
**Multi-view constraints**

- For most applications, the two-view constraint is the goto constraint
  - Fundamental matrix
  - Homography

- Easy to estimate

- Easy to use

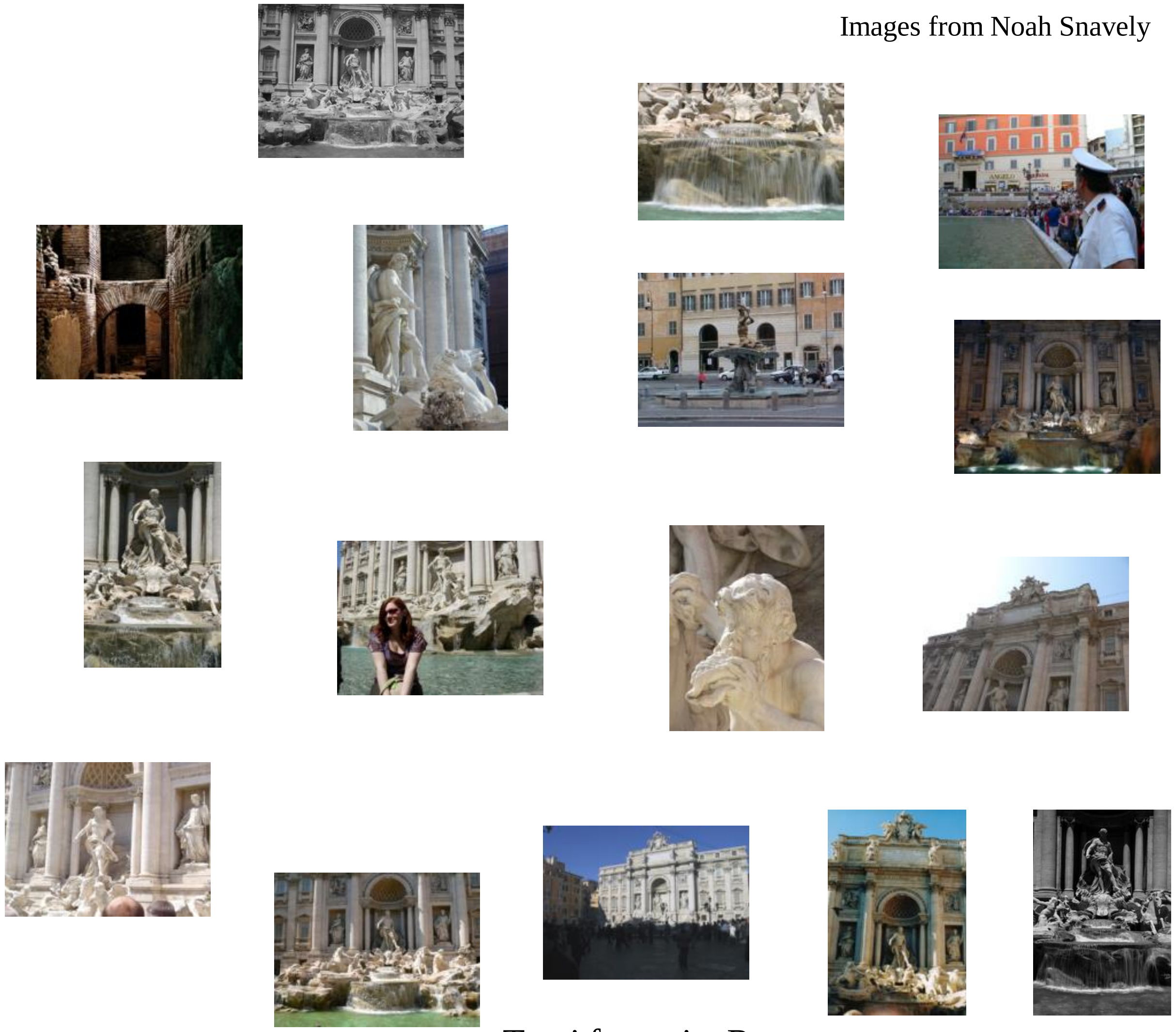
- Also useful for establishing correspondences between images



# Multi-view image correspondences

- Multi-view applications deal with image sets
- Establishing image correspondences can be very difficult and time consuming
- Simplifying factors
  - Ordered image set
  - Known camera model(s)
  - Known camera positions
  - Known camera orientations

Images from Noah Snavely



Trevi fountain, Rome

# Multi-view image correspondences

Images from Noah Snavely

## Establish image correspondences

- Detect features

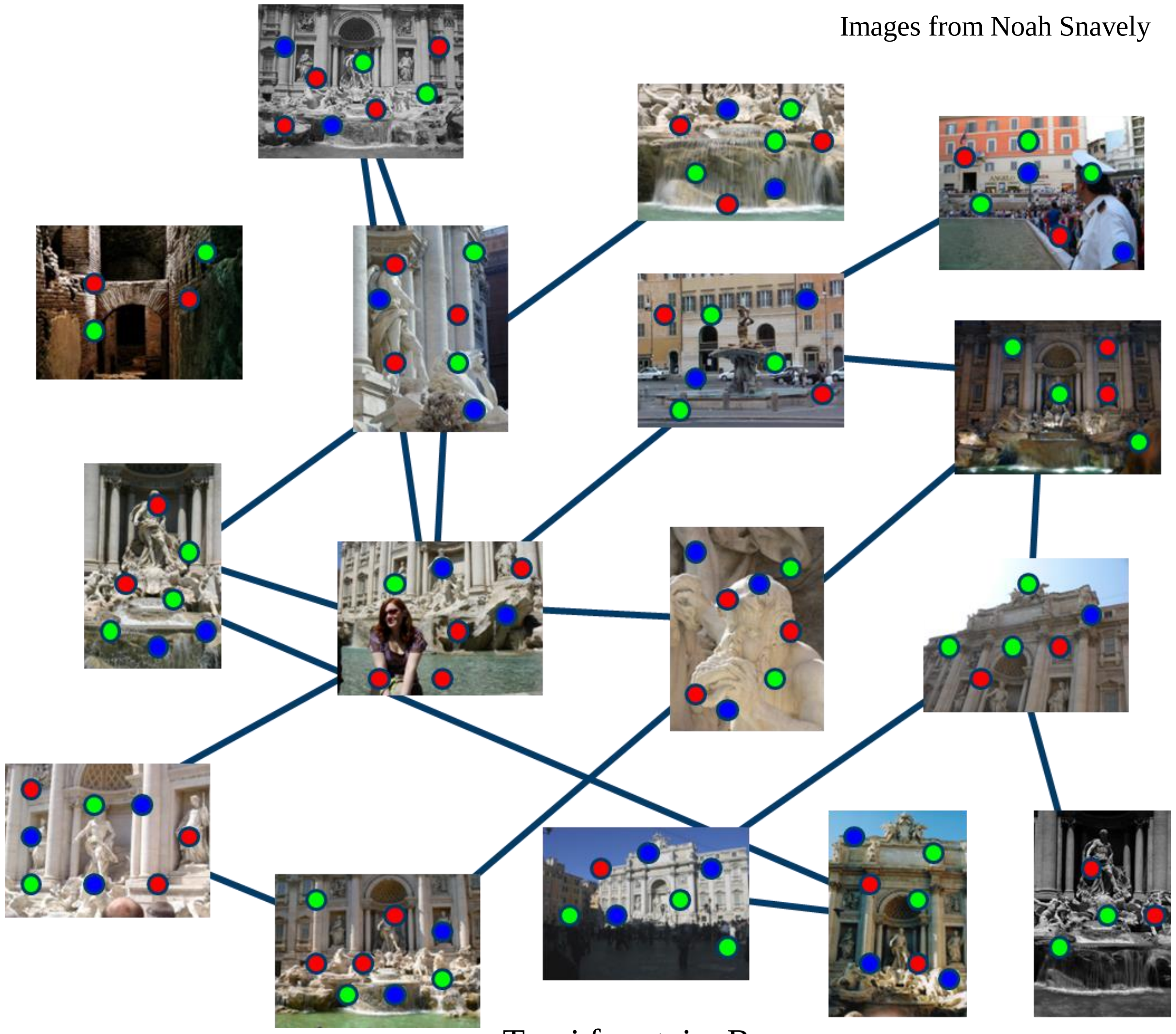
Trevi fountain, Rome

# Multi-view image correspondences

**Establish image correspondences**

- Detect features
- Match features between all image pairs
- Refine matching by estimating $\mathbf{F}$ in a RANSAC scheme

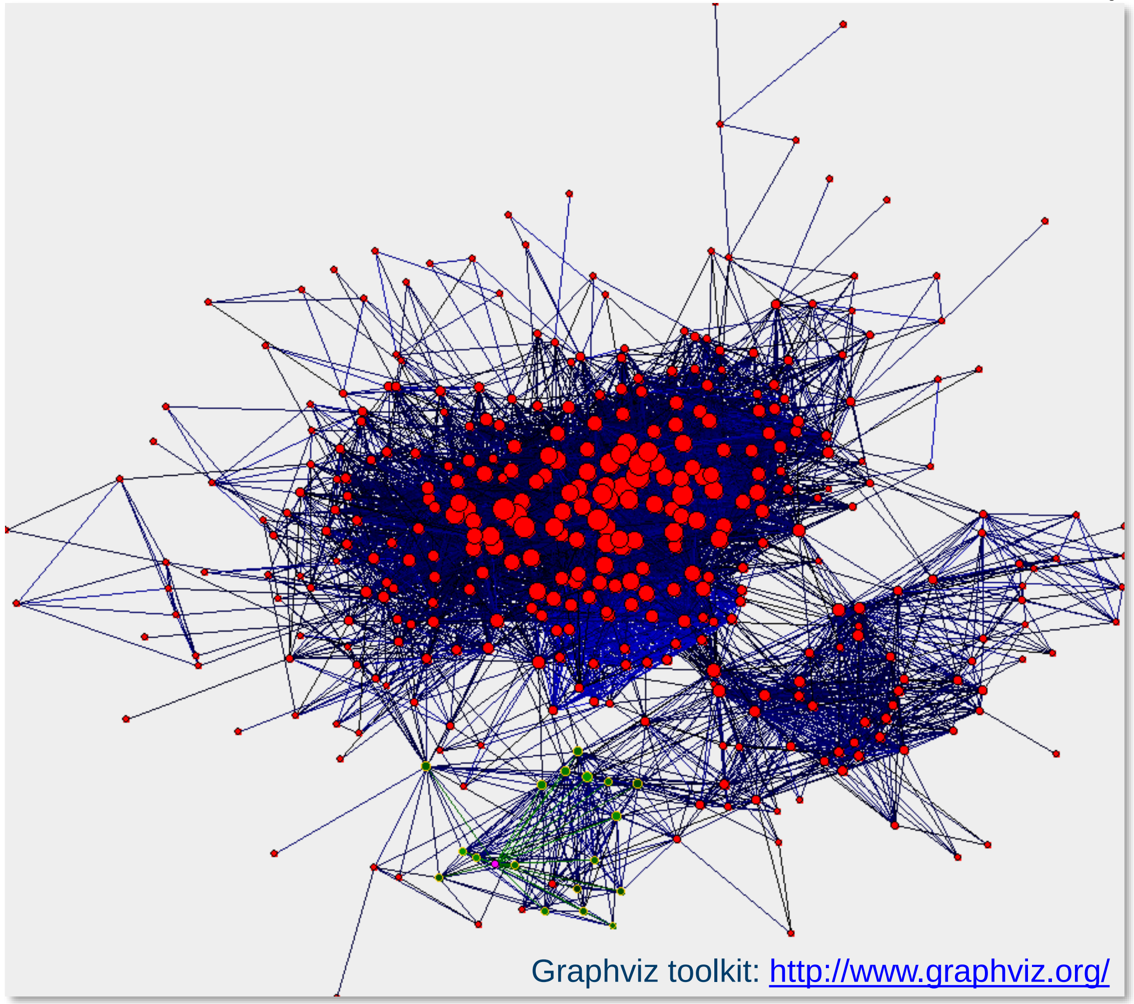
Images from Noah Snavely



Trevi fountain, Rome

# Multi-view image correspondences

Illustration from Noah Snavely




**Establish image correspondences**

- Detect features
- Match features between all image pairs
- Refine matching by estimating $\mathbf{F}$ in a RANSAC scheme
- Graph of image connectivity

# Multi-view image correspondences

**Establish image correspondences**

- Detect features
- Match features between all image pairs
- Refine matching by estimating $\mathbf{F}$ in a RANSAC scheme
- Graph of image connectivity
- Establish connected components

Images from Noah Snavely

# Multi-view geometry

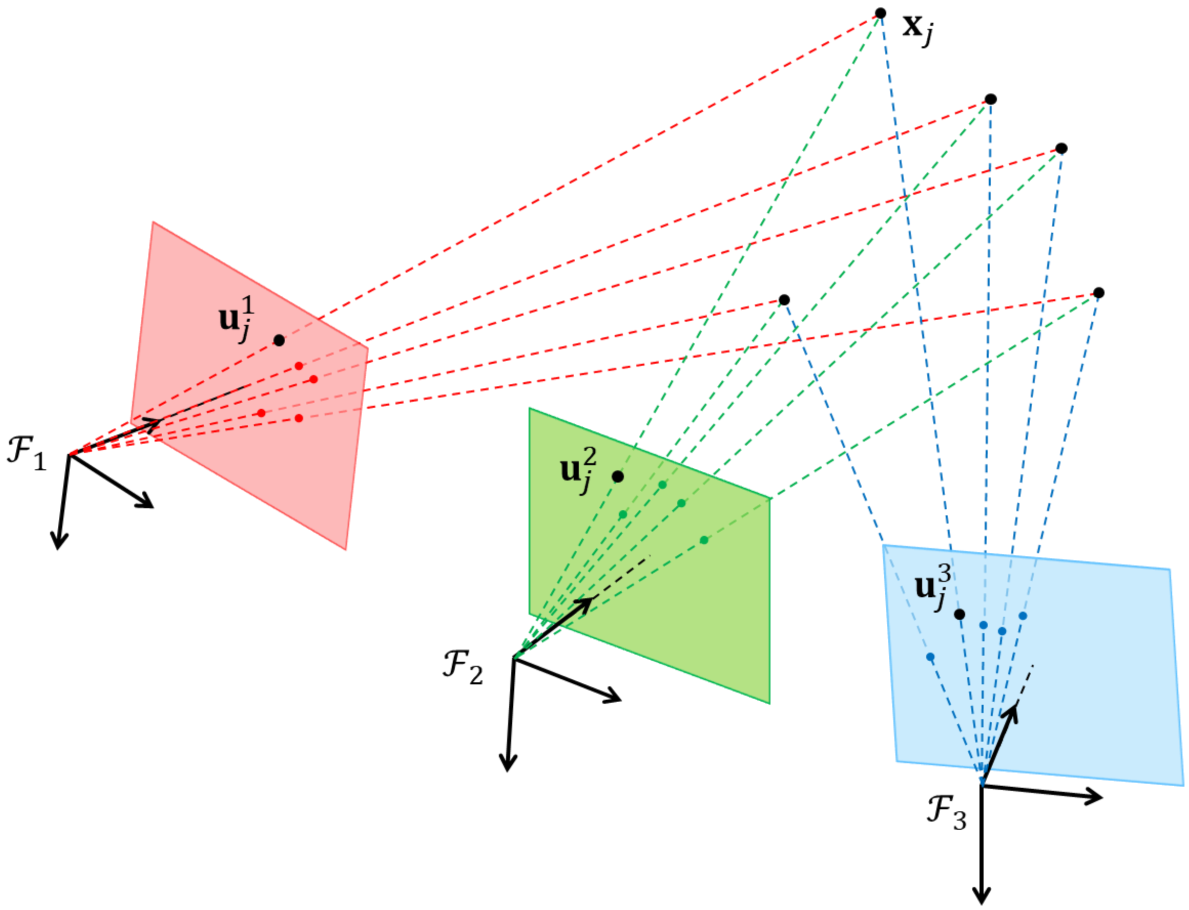
### Multi-view geometry vs multi two-view geometry

The estimated geometry is generally more accurate since we can optimize over all views simultaneously

### Full bundle adjustment (BA)

Estimate scene geometry and camera geometry by minimizing the total reprojection error

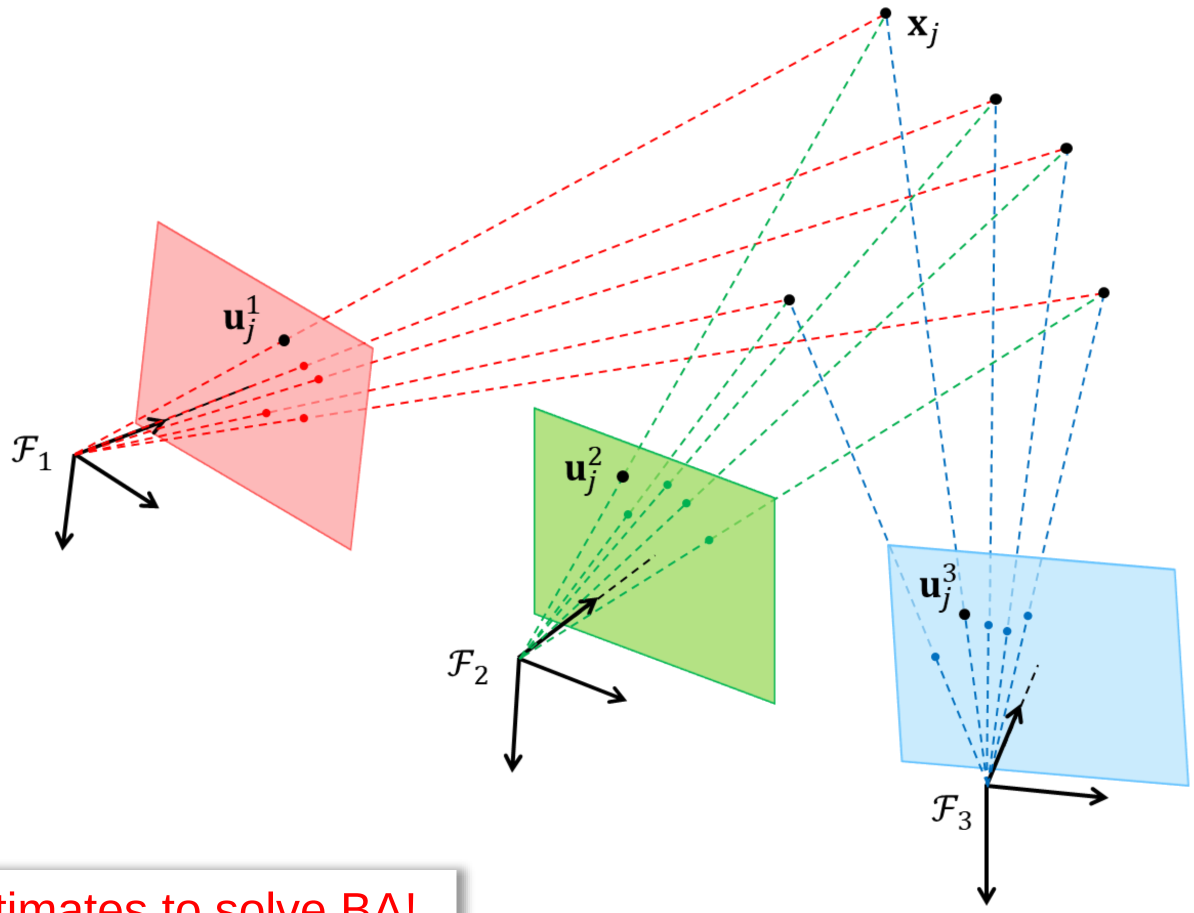
$$\{\mathbf{T}^*_{wc_i}, \mathbf{x}^*_j\} = \underset{\mathbf{T}_{wc_i}, \mathbf{x}_j}{\operatorname{argmin}} \sum_i \sum_j \left\| \pi_i\left(\mathbf{T}^{-1}_{wc_i} \cdot \mathbf{x}^w_j\right) - \mathbf{u}^i_j \right\|^2$$



# Multi-view geometry

## Multi-view geometry vs multi two-view geometry

The estimated geometry is generally more accurate since we can optimize over all views simultaneously




## Full bundle adjustment (BA)

Estimate scene geometry and camera geometry by minimizing the total reprojection error

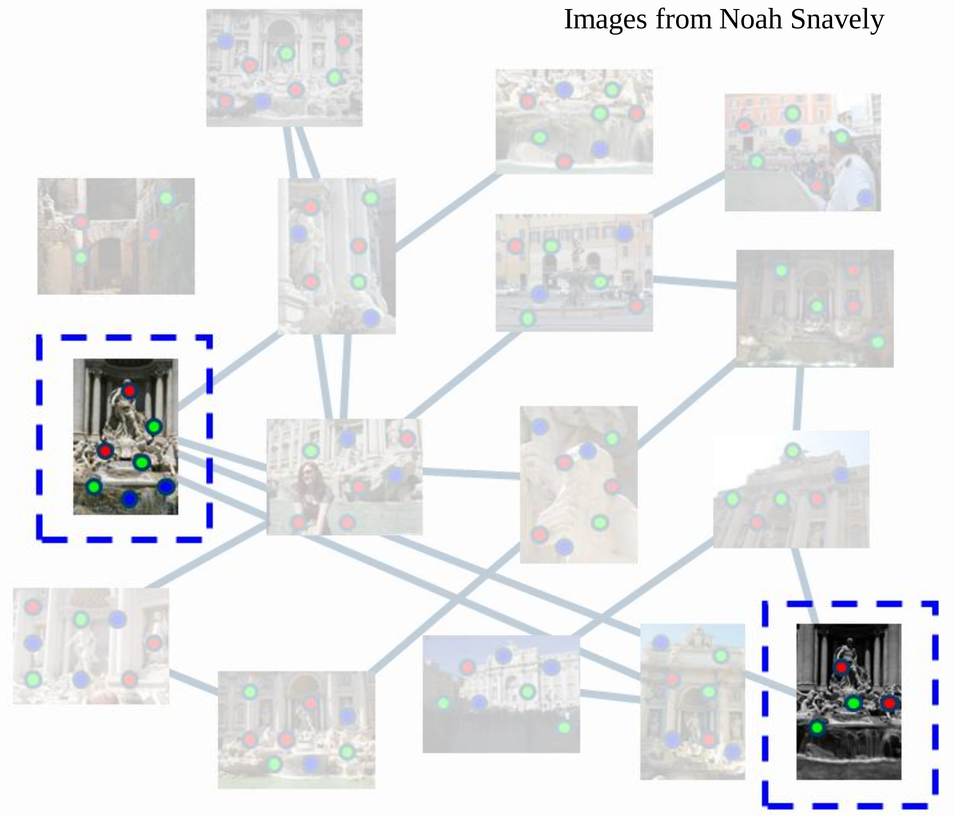
$$\{\mathbf{T}_{wc_i}^*, \mathbf{x}_j^*\} = \underset{\mathbf{T}_{wc_i}, \mathbf{x}_j}{\operatorname{argmin}} \sum_i \sum_j \left\| \pi_i\left(\mathbf{T}_{wc_i}^{-1} \cdot \mathbf{x}_j^w\right) - \mathbf{u}_j^i \right\|^2$$

But we need initial estimates to solve BA!

# Multi-view geometry

## Incremental Structure from Motion (SfM)

- Split images into groups of connected components
- For each component, start with two images
  - Many RANSAC inliers when estimating $\mathbf{F}$
  - Few RANSAC inliers when estimating $\mathbf{H}$
- Estimate camera geometry and scene geometry
  - Refine estimates by BA
- Add connected image
  - Estimate camera geometry
  - Expand scene geometry
  - Update all estimates with BA
- Merge components



Images from Noah Snavely

# Multi-view geometry


Video from Noah Snavely

# Multi-view geometry

Time-lapse reconstruction of Dubrovnik, Croatia

Video from Noah Snavely

# Summary

### Pixel correspondences

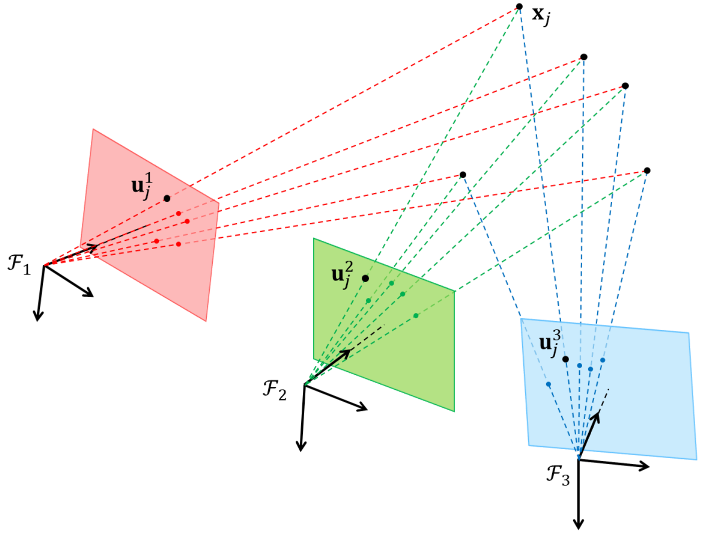
- Geometrical constraints: $\mathbf{F}$ and $\mathbf{H}$ most important

### Image correspondences

- Feature matching + RANSAC estimate of $\mathbf{F}$
- Image connectivity graph
- Connected components

### Scene geometry & camera geometry

- Determine initial estimates based on two-view geometry
- Optimize geometry over all views
- For example incremental SfM



# Supplementary material

**Recommended**

- *Richard Szeliski: Computer Vision: Algorithms and Applications 2nd ed*
  - Chapter 11 “Structure from motion and SLAM”, in particular section 11.4 “Multi-frame structure from motion”

**Other**

- Snavely N., Seitz S. M., *Modeling the World from Internet Photo Collections,* 2007
- Agarwal S. et al., *Building Rome in a Day*, 2011
- Heinly J. et al., Recontstructing the World in Six Days, 2015