3D Vision
BLG 634E

Professor: Gozde UNAL

Stereo Vision: Epipolar Geometry

Some slides courtesy of
Prof. Greg Slabaugh @ City University London

Stereovision

- In stereovision, *two or more* views of the scene are available from different vantage points, for example
  - From two separate cameras
  - A single camera that is translated
- By finding matches (correspondences) between the images, the relative distance of objects from the camera (depth) can be determined.
- The human visual system achieves stereopsis with binocular vision.
  - Two images of the world are captured by the eyes (~65 mm apart)
  - Receptive fields fuse images and recognise different positions
  - *Disparity* (the amount of displacement) is used to infer depth

Finding correspondences

- Given a point in one image, how do we find its match in the other image?

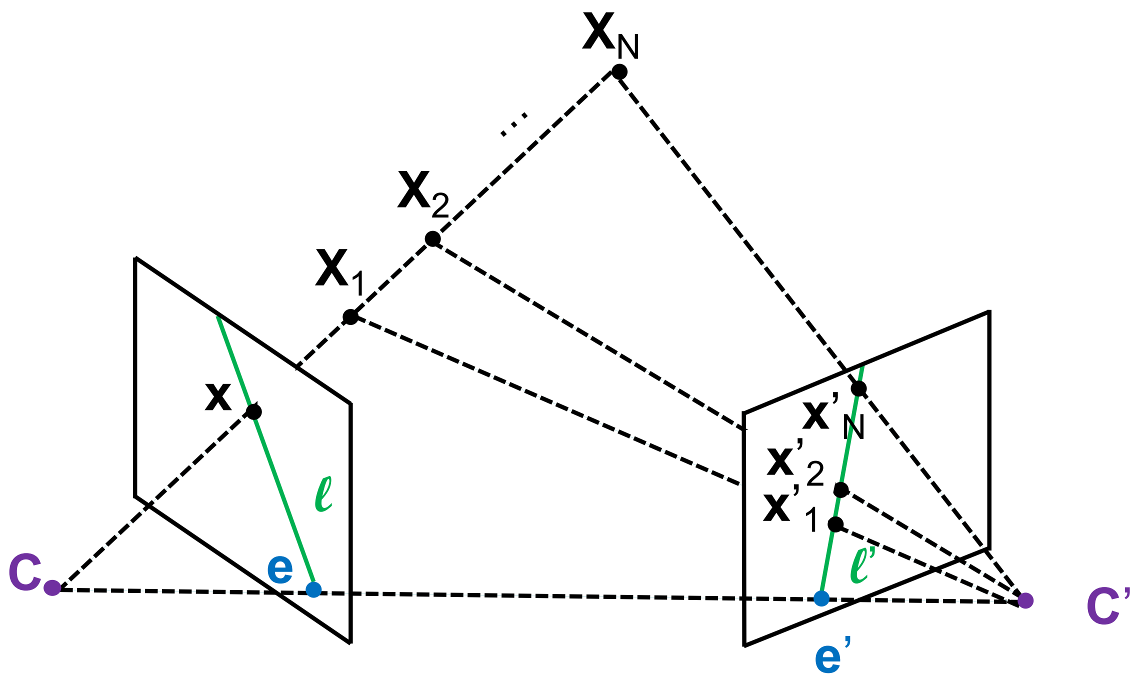
Left image

Right image

- Brute force: search every pixel in the right image to find a match.
- Actually we can do better…

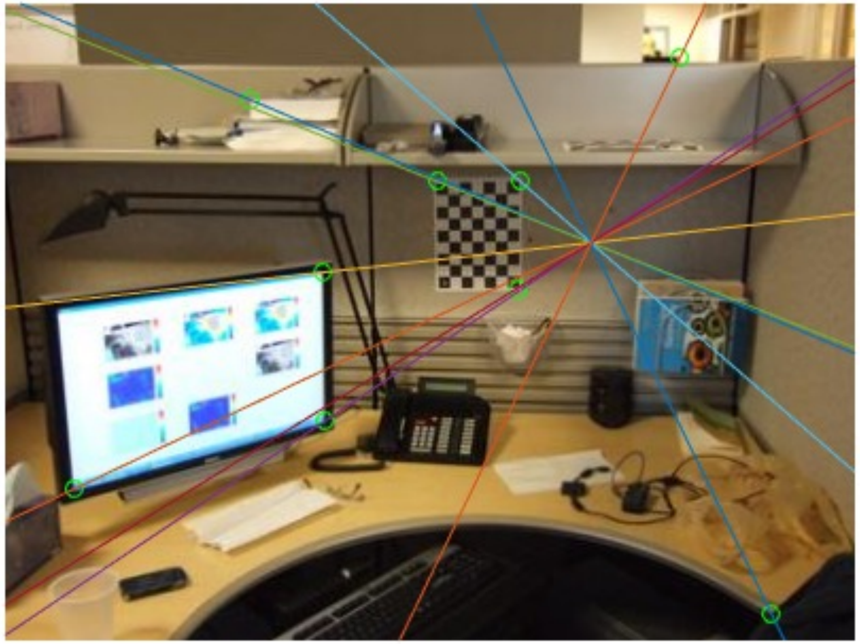
Epipolar geometry

- Epipolar geometry can be used to constrain the search *to a line*.



⇨ Potential matches for **x** must be on the line *ℓ'*.
Similarly, potential matches for **x**' have to be on the line *ℓ*.

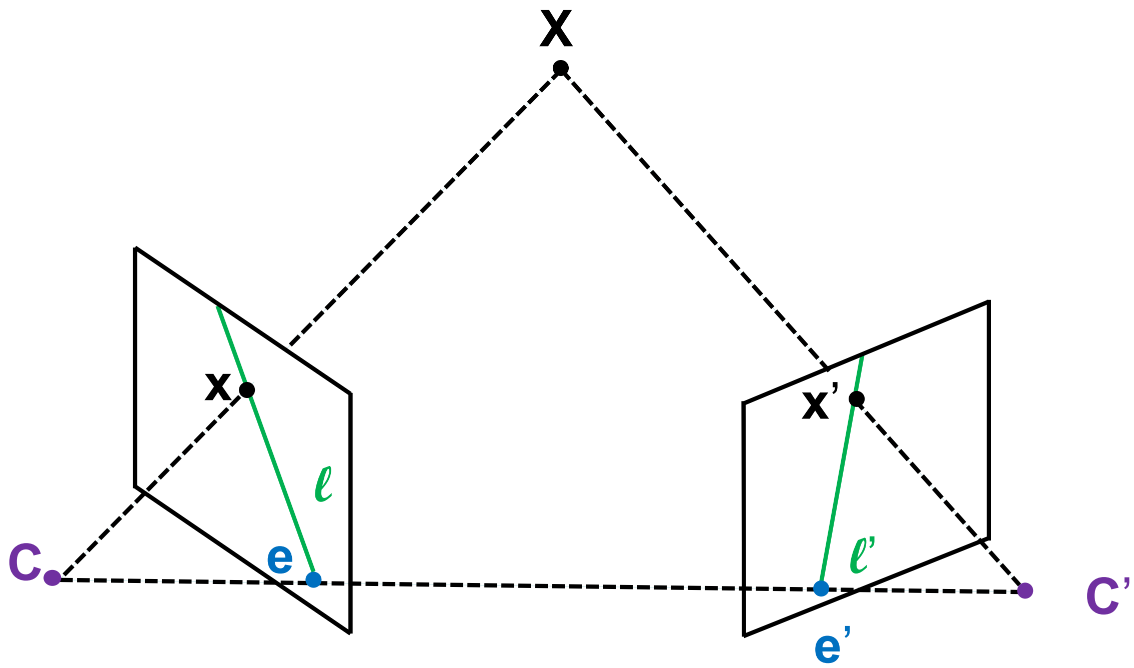
An example



⇨ This example shows epipolar lines in both images

Terminology

- Epipolar geometry can be used to constrain the search *to a line*.



- C, C', and **X** are three 3D points that form the *epipolar plane*.
- The baseline connects the two camera centres. The *epipoles* **e** and **e'** are where the baseline intersects the image plane.
- All epipolar lines in an image will go through the epipole, regardless of **X**.

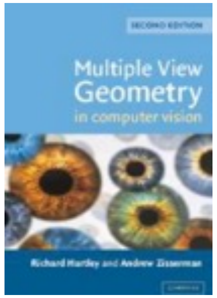
The fundamental matrix

- If the images are uncalibrated, one can compute the *fundamental matrix*, **F**. This 3x3 matrix is the algebraic representation of the epipolar geometry.
- The fundamental matrix allows us to find the epipolar line ℓ' = **Fx**.
- Since **x**' is on the line ℓ', $\mathbf{x}'^{T}\mathbf{F}\mathbf{x} = 0$.

Properties of the fundamental matrix:

- If **F** is the fundamental matrix going from image 1 to image 2, then $\mathbf{F}^{T}$ is the fundamental matrix going from image 2 to image 1.
- $\mathbf{Fe} = \mathbf{0}$
- The fundamental matrix has rank 2

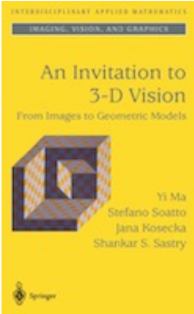
Reading Assignments for Epipolar Geometry



Hartley Zissermann: Multiple View Geometry Book:

Chapter 9:

https://www.robots.ox.ac.uk/~vgg/hzbook/hzbook2/HZepipolar.pdf



Ma, Soatto et al book (An Invitation to 3D Vision):

Chapter 5.1, 5.2, 5.3 and Chapter 6.1, 6.2, 6.A

For 3D Triangulation, see: G. Slabaugh, R. Schafer, and M. Livingston, “Optimal ray intersection for computing 3d points from n-view correspondences,” Technical Report, 2001.

http://www.staff.city.ac.uk/~sbbh653/publications/opray.pdf

C. Loop and Z. Zhang: Computing Rectifying Homographies for Stereo Vision, Technical Report, Microsoft Research, 1999.

Estimating the fundamental matrix

- There are several approaches to estimate **F**. The best-known approach is called the *eight point algorithm*, as it requires eight correspondences.
- One can write out the epipolar constraint as

$$\mathbf{x'}^T F\mathbf{x} = [x' \quad y' \quad 1] \begin{bmatrix} f_{11} & f_{12} & f_{13} \\ f_{21} & f_{22} & f_{23} \\ f_{31} & f_{32} & f_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = 0$$

multiplying this out, one gets

$$x'xf_{11} + x'yf_{12} + x'f_{13} + y'xf_{21} + y'yf_{22} + y'f_{23} + xf_{31} + yf_{32} + f_{33} = 0$$

for each correspondence, we have one such equation.

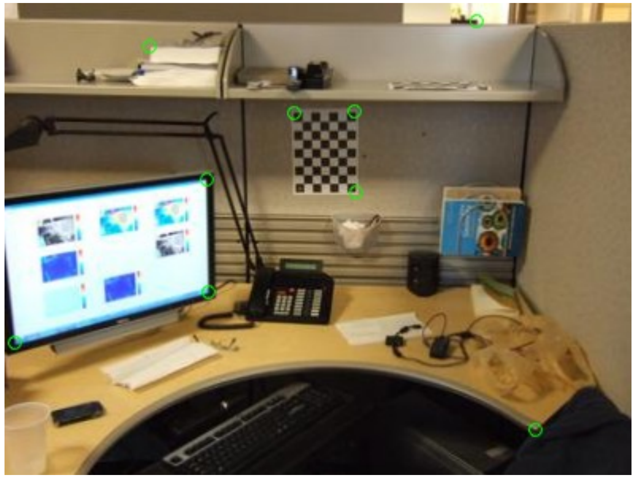
Estimating the fundamental matrix

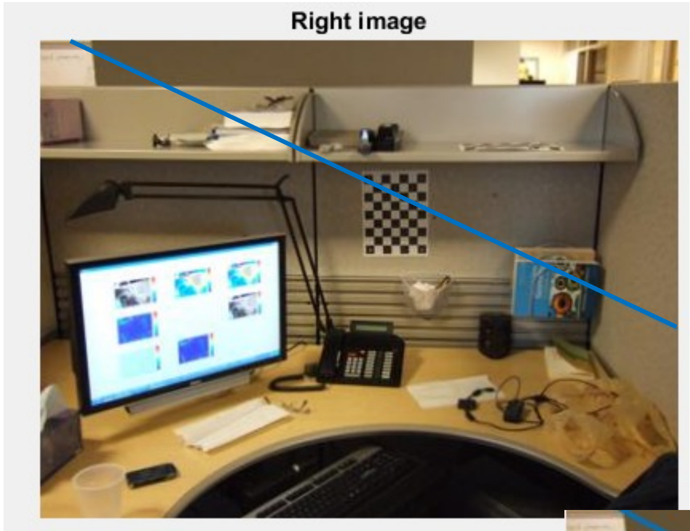
- So with *N* correspondences, we can write

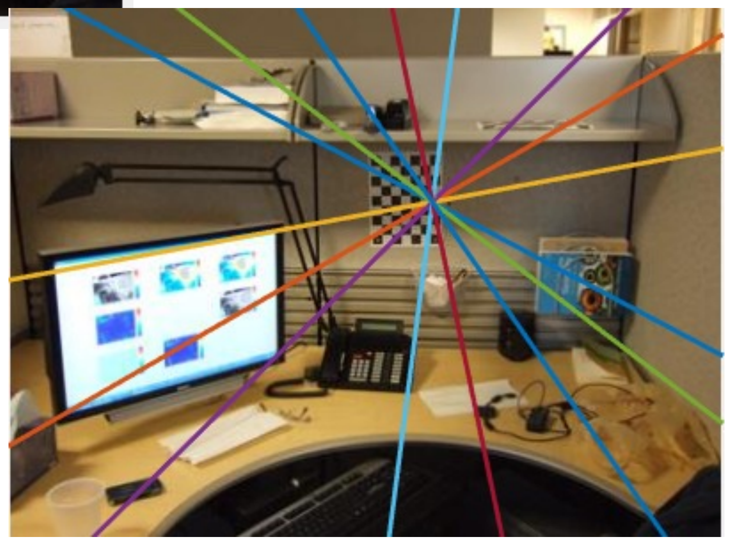
$$\begin{bmatrix} x_1' x_1 & x_1' y_1 & x_1' & x_1 y_1' & y_1 y_1' & y_1' & x_1 & y_1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_N' x_N & x_N' y_N & x_N' & x_N y_N' & y_N y_N' & y_N' & x_N & y_N & 1 \end{bmatrix} \begin{bmatrix} f_{11} \\ f_{12} \\ f_{13} \\ f_{21} \\ f_{22} \\ f_{23} \\ f_{31} \\ f_{32} \\ f_{33} \end{bmatrix} = \mathbf{0}$$

- and solve for **F**. However, we can set ||det(**F**)|| = 1, so this can be reduced to eight correspondences.
- If there are more than eight correspondences, we can use all and solve to get a least squares solution.

Implementation in Matlab







Recall RANSAC

- Note there is a *chicken-and-egg* problem here. To find correspondences, the fundamental matrix is really helpful. But to estimate the fundamental matrix, we need correspondences!
- Also, automation is desired, as manually finding corresponding points may not be practical.
- A common approach is to employ RANSAC, which stands for **RAN**dom **SA**mple **C**onsensus. Essentially, it is a "hypothesize and test" methodology.

Given a set of correspondences (e.g., matching SURF points) found in two images:

1. Randomly subsample a set of correspondences C, selecting eight called C*
2. Estimate the fundamental matrix **F** from C*
3. Score F by determining the fraction of inliers in C using a threshold
4. Go to 1 until a best **F** is found

Method for Fundamental Matrix Estimation

- Use SURF to find feature points in both images
- Match SURF features to generate possible correspondences
- Use RANSAC to find the best F using 8 point algorithm

3D estimation

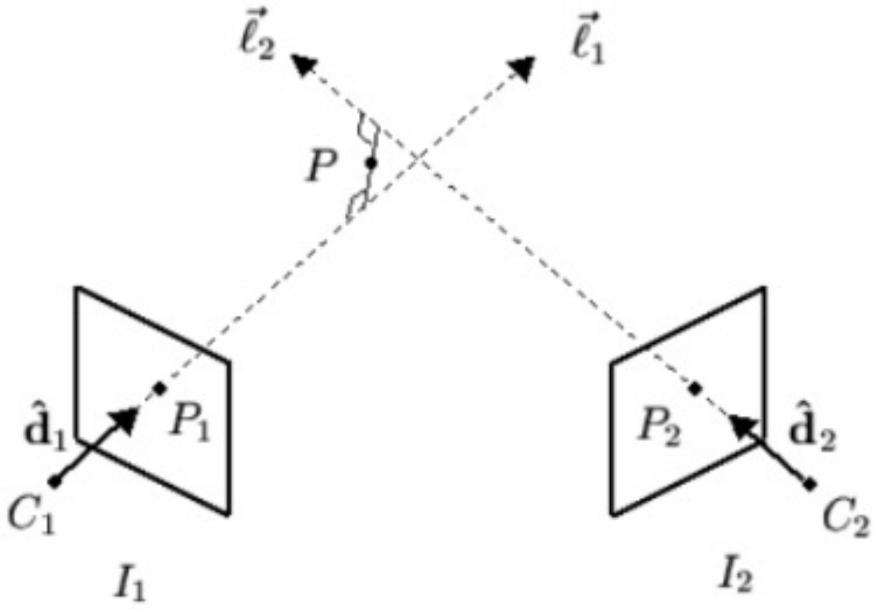
- Compute the fundamental matrix **F** between the two views
- First camera matrix: [**I** | 0]
- If the intrinsics are known , the Matlab function
  - `[R, t] = cameraPose(F, cameraParams, inliers1, inliers2)`

  can estimate the relative rotation and translation between views, and the second camera matrix is then **K** * [**R**, **t**]

- One can then *triangulate* the matching 2D points to compute 3D points **P**.

- How can we triangulate to find the 3D Point P?

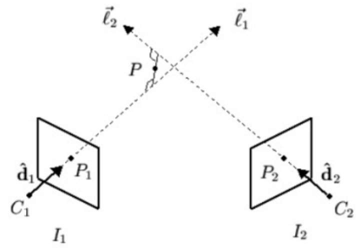
Triangulation



Reference: G. Slabaugh, R. Schafer, and M. Livingston, "Optimal ray intersection for computing 3d points from n-view correspondences," Technical Report, 2001.

http://www.staff.city.ac.uk/~sbbh653/publications/opray.pdf

Triangulation



- There are many ways to triangulate to determine at point **P** give two or more projections given matching points **p** and camera matrices **M** in each image.
- A common approach is to set up a system of linear equations in **P** which can then be solved.

$$\mathbf{p} = \mathbf{M}\mathbf{P} = \begin{bmatrix} \mathbf{m}_1^T \\ \mathbf{m}_2^T \\ \mathbf{m}_3^T \end{bmatrix} \mathbf{P} \qquad w \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{m}_1^T\mathbf{P} \\ \mathbf{m}_2^T\mathbf{P} \\ \mathbf{m}_3^T\mathbf{P} \end{bmatrix}$$

$$\begin{aligned} wx &= \mathbf{m}_1^T\mathbf{P} \\ wy &= \mathbf{m}_2^T\mathbf{P} \\ w &= \mathbf{m}_3^T\mathbf{P} \end{aligned} \qquad \begin{aligned} x\mathbf{m}_3^T\mathbf{P} &= \mathbf{m}_1^T\mathbf{P} \\ y\mathbf{m}_3^T\mathbf{P} &= \mathbf{m}_2^T\mathbf{P} \end{aligned}$$

known

2 equations per image, linear in **P**

- Can solve for two or more images. In Matlab, you can use
- `P = triangulate(matchedPoints1,matchedPoints2,cameraMatrix1,cameraMatrix2)`

Optimisation for Bundle Adjustment

- Optimise over all views and all points by minimising the reprojection error R for each point. This is a non-linear least squares problem, and is known in the literature as *bundle adjustment*. Performed when >2 images.

$$E(\mathbf{M}, \mathbf{P}) = \sum_{i=1}^{M} \sum_{j=1}^{N} R(\mathbf{q}_{ij}, \mathbf{M}_i \mathbf{P}_j)$$