

Introduction to Cognitive Robotics

Module 5: Robot Vision

Lecture 10: Inverse perspective transformation; stereo vision; epipolar geometry

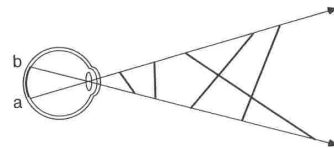
David Vernon
Carnegie Mellon University Africa

www.vernon.eu

Depth Perception: The Inverse Problem

Estimate depth (3D) of world from images

Inverse perspective transformation



Credit: Markus Vincze, Technische Universität Wien

The Inverse Perspective Transformation

- Once the camera model C has been determined, we are in a position to determine an expression for the co-ordinates of a point in the real world in terms of the co-ordinates of its imaged position
- Recalling equations (1) - (5) :

$$C_{11}x + C_{12}y + C_{13}z + C_{14} = u = Ut$$

$$C_{21}x + C_{22}y + C_{23}z + C_{24} = v = Vt$$

$$C_{31}x + C_{32}y + C_{33}z + C_{34} = t$$

The Inverse Perspective Transformation

Substituting the expression for t into the first two equations gives

$$U[C_{31}x + C_{32}y + C_{33}z + C_{34}] = C_{11}x + C_{12}y + C_{13}z + C_{14}$$

$$V[C_{31}x + C_{32}y + C_{33}z + C_{34}] = C_{21}x + C_{22}y + C_{23}z + C_{24}$$

Hence

$$[C_{11} - UC_{31}]x + [C_{12} - UC_{32}]y + [C_{13} - UC_{33}]z + [C_{14} - UC_{34}] = 0$$

$$[C_{21} - VC_{31}]x + [C_{22} - VC_{32}]y + [C_{23} - VC_{33}]z + [C_{24} - VC_{34}] = 0$$

The Inverse Perspective Transformation

Letting

$$a_1 \triangleq C_{11} - UC_{31}$$

$$b_1 \triangleq C_{12} - UC_{32}$$

$$c_1 \triangleq C_{13} - UC_{33}$$

$$d_1 \triangleq C_{14} - UC_{34}$$

and

$$a_2 \triangleq C_{21} - VC_{31}$$

$$b_2 \triangleq C_{22} - VC_{32}$$

$$c_2 \triangleq C_{23} - VC_{33}$$

$$d_2 \triangleq C_{24} - VC_{34}$$

we have

$$a_1x \triangleq b_1y \triangleq c_1z \triangleq d_1 \triangleq 0$$

$$a_2x \triangleq b_2y \triangleq c_2z \triangleq d_2 \triangleq 0$$

The Inverse Perspective Transformation

These are the equations of two planes

The intersection of these planes determines a line comprising the set of real-world points which project onto the image point $\begin{bmatrix} U \\ V \end{bmatrix}$

Solving these plane equations simultaneously (in terms of z)

$$x = \frac{z[b_1c_2 - b_2c_1][b_1d_2 - b_2d_1]}{[a_1b_2 - a_2b_1]}$$
$$y = \frac{z[a_2c_1 - a_1c_2][a_2d_1 - a_1d_2]}{[a_1b_2 - a_2b_1]}$$

The Inverse Perspective Transformation

Thus, for any given z_0 , U and V , we may determine the corresponding x_0 and y_0 , i.e. the real-world co-ordinates

The camera model and the inverse perspective transformation which we have just discussed allow us to compute the x and y real-world co-ordinates corresponding to a given position in the image

However, **we must assume that the z coordinate**, i.e. the distance from the camera, **is known**

The Inverse Perspective Transformation

- For some applications, e.g. where objects lie on a table at a given and constant height (*i.e.* at a given z_0), this is sufficient
- In general, however, we will not know the coordinate of the object in the third dimension and **we must recover it**

The Inverse Perspective Transformation

How we can compute z_0 ?

- If we have a second image of the scene, **taken from another viewpoint**
- If we know the image coordinates of the point of interest in this image
- Then we have two camera models and, hence, two inverse perspective transformations
- Instead of solving two plane equations simultaneously, **we solve four plane equations**

The Inverse Perspective Transformation

In particular, we have

$$a_1x + b_1y + c_1z + d_1 = 0$$

$$a_2x + b_2y + c_2z + d_2 = 0$$

$$p_1x + q_1y + r_1z + s_1 = 0$$

$$p_2x + q_2y + r_2z + s_2 = 0$$

where

$$a_1 = C1_{11} - U1C1_{31} \quad a_2 = C1_{21} - V1C1_{31}$$

$$b_1 = C1_{12} - U1C1_{32} \quad b_2 = C1_{22} - V1C1_{32}$$

$$c_1 = C1_{13} - U1C1_{33} \quad c_2 = C1_{23} - V1C1_{33}$$

$$d_1 = C1_{14} - U1C1_{34} \quad d_2 = C1_{24} - V1C1_{34}$$

$$p_1 = C2_{11} - U2C2_{31} \quad p_2 = C2_{21} - V2C2_{31}$$

$$q_1 = C2_{12} - U2C2_{32} \quad q_2 = C2_{22} - V2C2_{32}$$

$$r_1 = C2_{13} - U2C2_{33} \quad r_2 = C2_{23} - V2C2_{33}$$

$$s_1 = C2_{14} - U2C2_{34} \quad s_2 = C2_{24} - V2C2_{34}$$

$C1_{ij}$ and $C2_{ij}$ are the coefficients of the camera model for the first and second images, respectively.
 $U1, V1$ and $U2, V2$ are the co-ordinates of the point of interest in the first and second images, respectively.

The Inverse Perspective Transformation

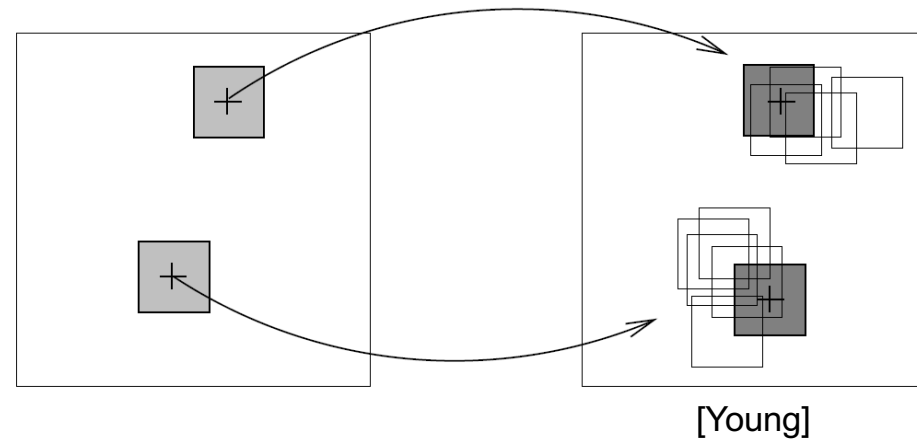
Since we now have four equations and three unknowns,
the system is over-determined

So, we compute a least-square-error solution using the pseudo-inverse technique

The Inverse Perspective Transformation

- It should be noted that the key here is not so much the mathematics which allows us to compute x_0 , y_0 and z_0 but, rather, the **image analysis by which we identify the corresponding point of interest in the two images**
- It is this **correspondence problem** which lies at the heart of most of the difficulties in recovery of depth information

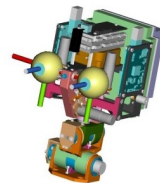
The Inverse Perspective Transformation



Credit: Markus Vincze, Technische Universität Wien

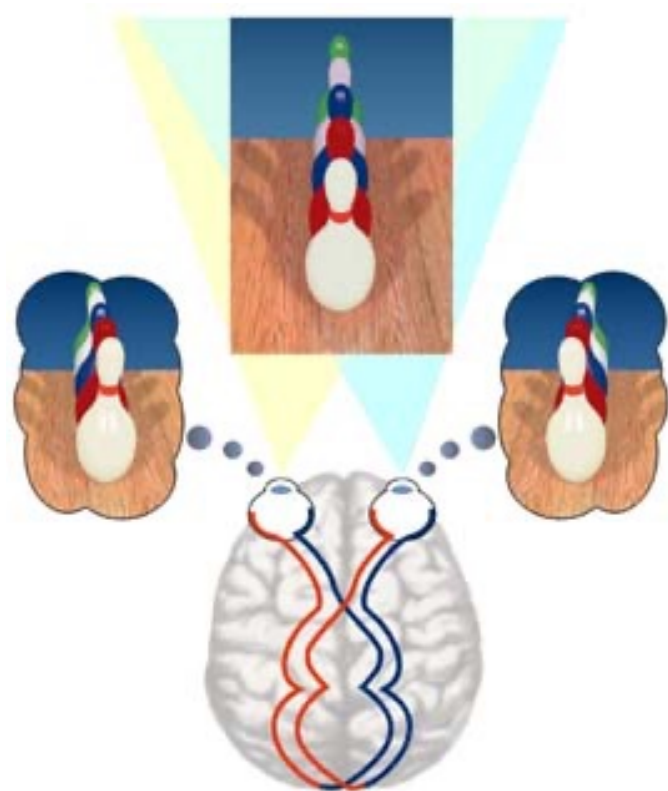
Stereopsis – Stereo Vision

We refer to stereo vision as the problem of inferring 3D information (structure and distances) from two or more images taken from different viewpoints



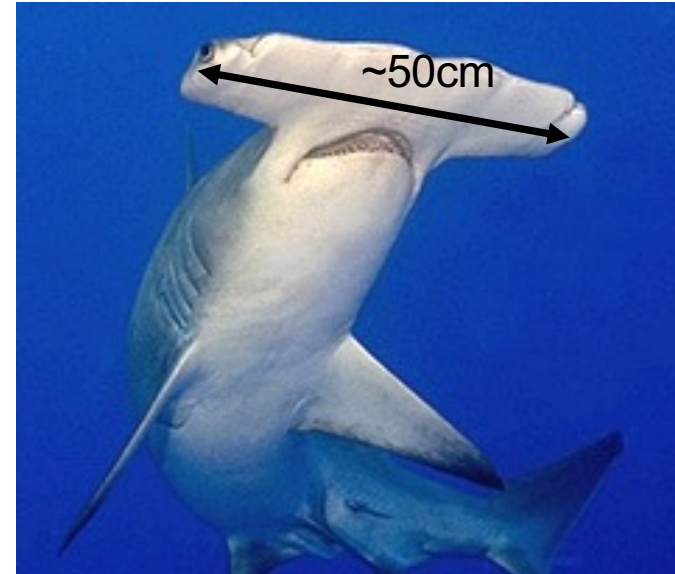
Credit: Francesca Odone, University of Genova

Stereopsis – Stereo Vision



Credit: Markus Vincze, Technische Universität Wien

Stereopsis – Stereo Vision



- Larger baseline increases useful range of depth estimated from stereo
- After a few meters disparity is quite small and depth from stereo is unreliable ...

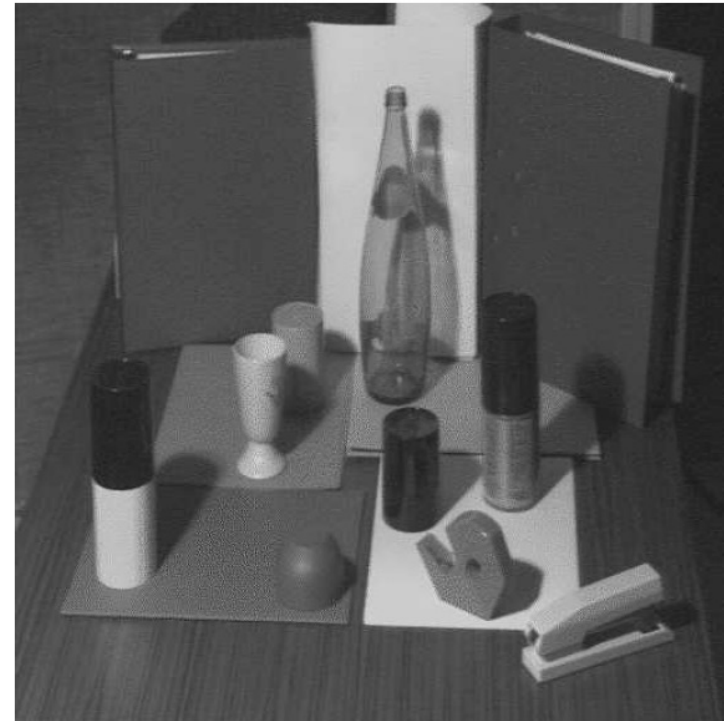
Stereopsis – Stereo Vision



[Young]

Credit: Markus Vincze, Technische Universität Wien

Stereopsis – Stereo Vision



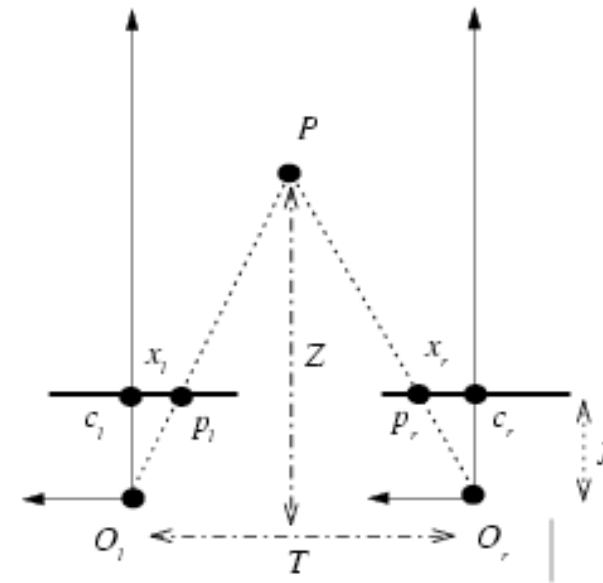
[Young]

Credit: Markus Vincze, Technische Universität Wien

Stereopsis – Stereo Vision

- **Disparity** d is the relative distance between corresponding points (on the image plane)
- **Depth** Z is the distance from a 3D point to the viewing system
- Depth is **inversely proportional** to disparity

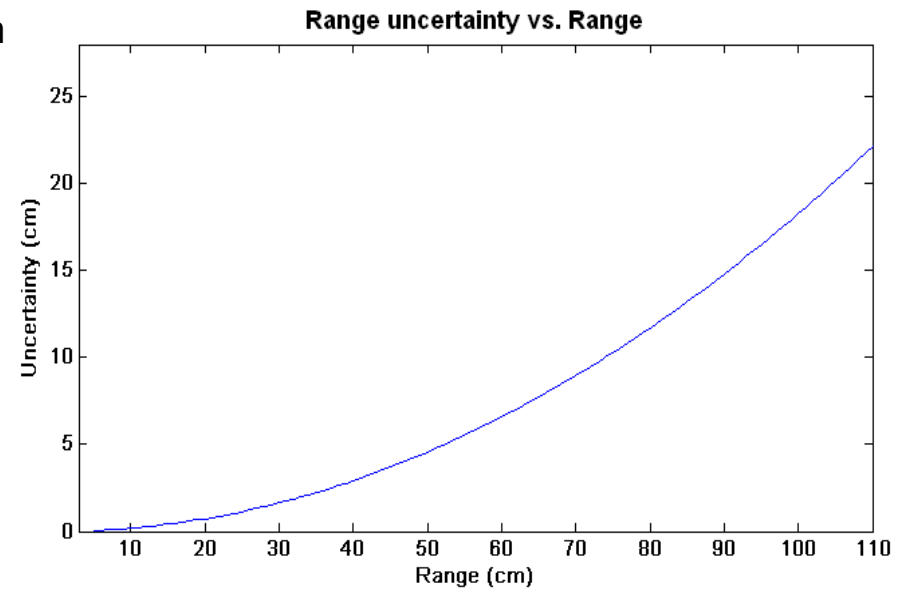
$$Z = \frac{fT}{x_r - x_l} = \frac{fT}{d}$$



Credit: Francesca Odone, University of Genova

Stereopsis – Stereo Vision

- The more distant the object, the larger the depth uncertainty
 - Acute angle: disparity uncertainty grows non-linearly
 - Improve with large focal length and baseline distance
- Humans use stereo only up to arm length
 - Then relative and perspective cues dominate



[S. Ahuja]

Stereopsis – Stereo Vision

- Dense stereo correspondence
- We assume we have two **rectified** images
 - where conjugate points lie on corresponding scanlines of the image (“rows”)
- Our goal is to obtain a **disparity map** giving the relative displacement for each pixel

Stereopsis – Stereo Vision

Assuming a fixation point at infinity, disparity is proportional to the inverse of the distance

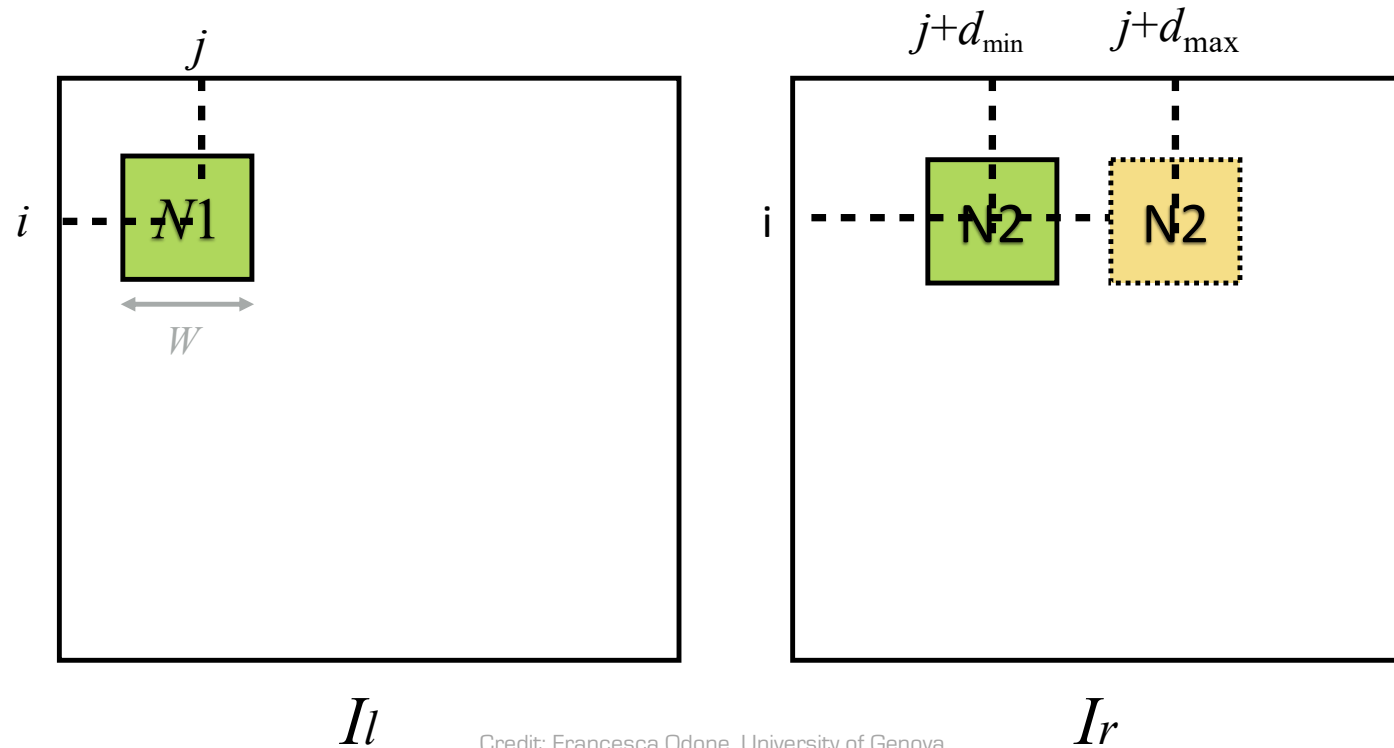
$$Z = \frac{fT}{x_r - x_l} = \frac{fT}{d}$$


Credit: Francesca Odone, University of Genova

Stereopsis – Stereo Vision

Given a stereo pair of **rectified** images I_l and I_r

- size of a correlation window W
- a search range $[d_{\min}, d_{\max}]$



Credit: Francesca Odono, University of Genova

Stereopsis – Stereo Vision

Dense correspondences: left-right consistency



Credit: Francesca Odone, University of Genova

Stereopsis – Stereo Vision

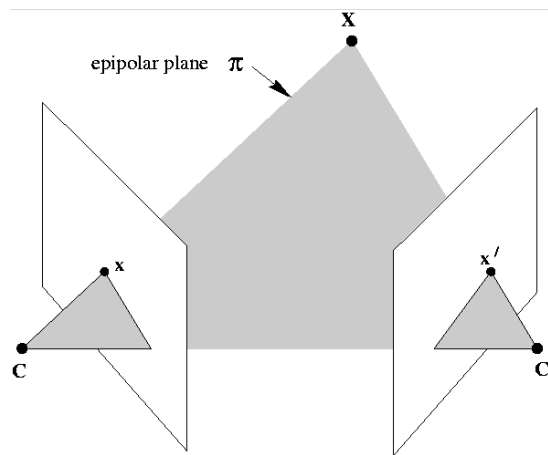
- **Correspondence problem**: finding the *same* point in both images
- Search in entire image is very costly
- Geometry of cameras produces constraints: epipolar plane and epipolar line
 - Limits search to a line in the image
- Finding the same points
 - Correlation (region) or features (edge)

Credit: Markus Vincze, Technische Universität Wien

Stereopsis – Stereo Vision

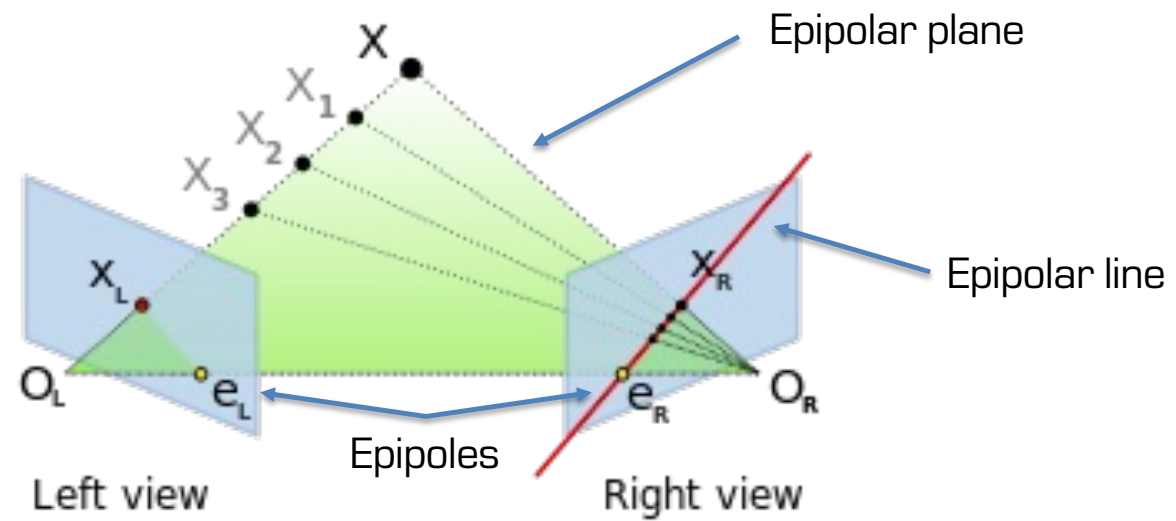
The geometry of a stereo-system is called epipolar geometry

It provides a geometrical prior to the algorithms



Credit: Francesca Odone, University of Genova

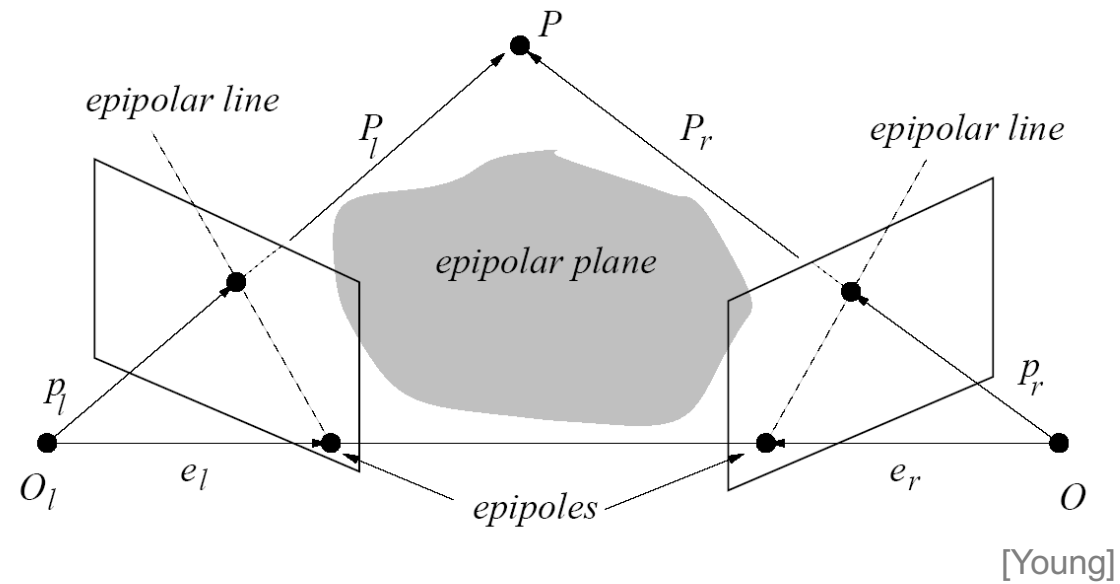
Epipolar Geometry



https://en.wikipedia.org/wiki/Epipolar_geometry

Epipolar Geometry

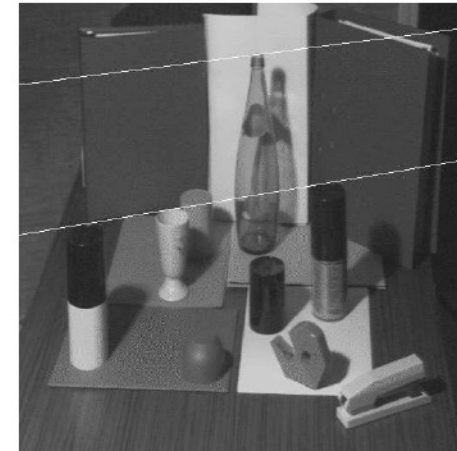
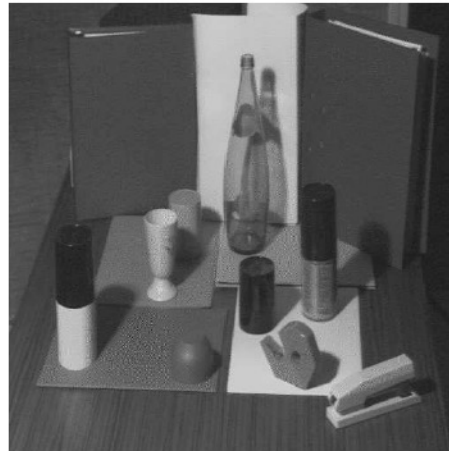
- Epipolar plane: plane of the two visible rays
- Pre-condition: known camera geometry, calibration



Credit: Markus Vincze, Technische Universität Wien

Epipolar Lines

- Each point left defines epipolar line right
- → 1D search for the same feature
- Simplifies correspondence problem

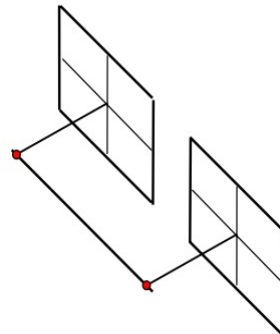


[Young]

Credit: Markus Vincze, Technische Universität Wien

Horizontal Epipolar Lines

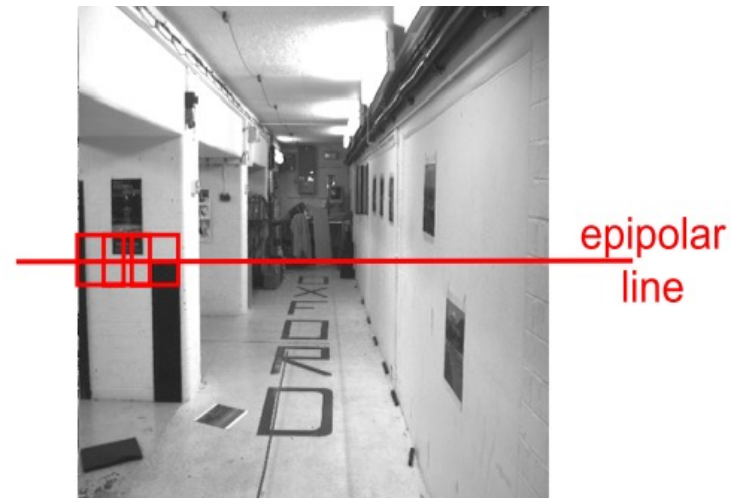
- Simple case: parallel cameras (**fronto-parallel stereo**)
 - In practice not obtainable accurately
- **Rectification** (calibration and elimination of distortions) of the images to obtain epipolar lines on the pixel array of the camera



Credit: Markus Vincze, Technische Universität Wien

Correspondence along a Line

- Search for left image point in the right image
- Dense depth image: correspondence for every point
- Sparse depth image: only distinctive points



Credit: Markus Vincze, Technische Universität Wien

Reading

R. Szeliski, *Computer Vision: Algorithms and Applications*, Springer, 2010.

Section 2.1.5 3D to 2D projections

Section 6.3 Geometric intrinsic calibration

Vernon, D. 1991. *Machine Vision: Automated Visual Inspection and Robot Vision*, Prentice-Hall International; Section 8.6

OpenCV documentation on camera calibration:

http://docs.opencv.org/2.4/modules/calib3d/doc/camera_calibration_and_3d_reconstruction.html

Demo

Read "Camera Modelling and Camera Calibration.pdf"
Then walk through the following example applications:

cameraInvPerspectiveMonocular

cameraInvPerspectiveBinocular