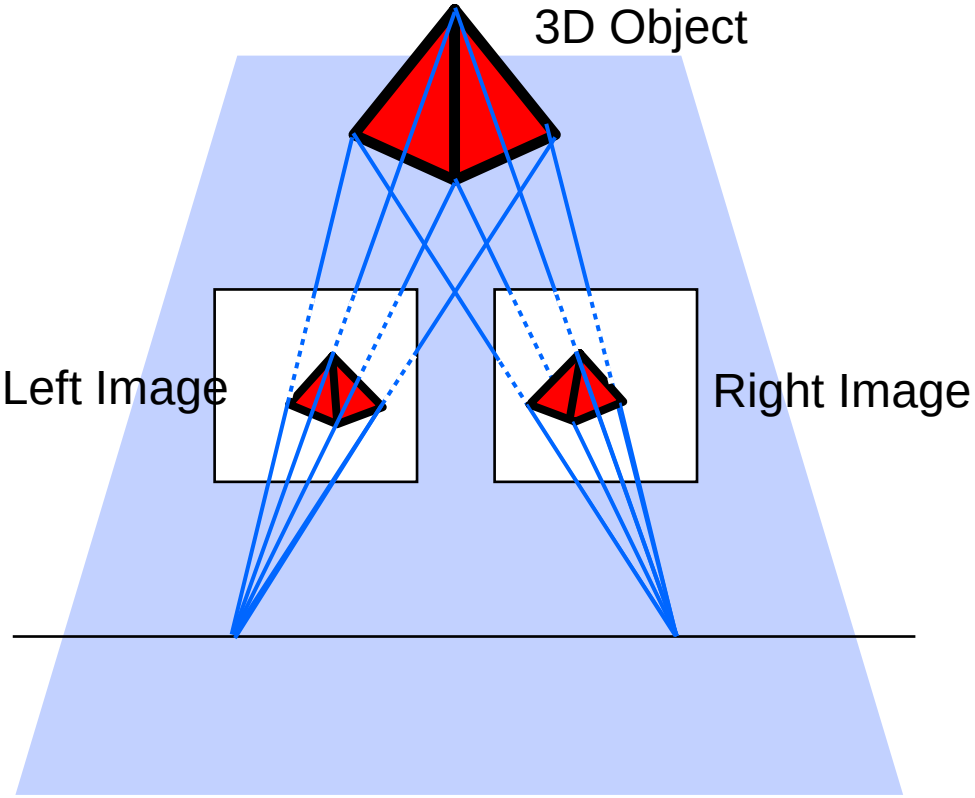
Stereo Vision versus Structure from Motion

- **Stereo vision:**
  is the process of obtaining **depth information** from a pair of images coming from two cameras that look at the same scene from different but **known** positions

- **Structure from Motion:**
  is the process of obtaining **depth and motion information** from a pair of images coming from the same camera that looks at the same scene from different positions

Stereo Vision | working principle

- Observe scene from two different viewpoints and solve for the intersection of the rays and recover the 3D structure



Stereo Vision | simplified case

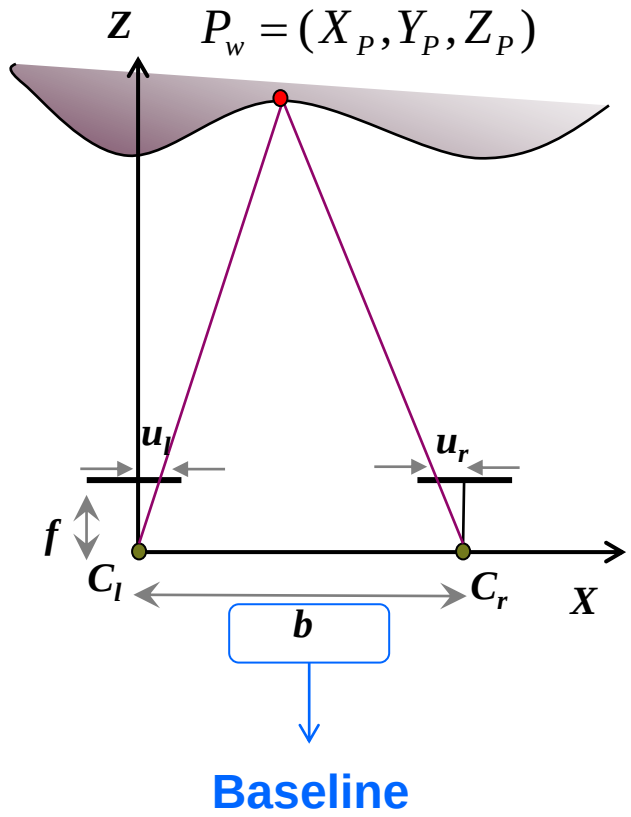
- An ideal, simplified case assumes that both cameras are **identical** and **aligned** with the x-axis
- Can we find an expression for the depth Z_P of point P_W?
- From similar triangles:

$$\frac{f}{Z_P} = \frac{u_l}{X_P}$$

$$\frac{f}{Z_P} = \frac{-u_r}{b - X_P}$$

$$Z_P = \frac{bf}{u_l - u_r}$$

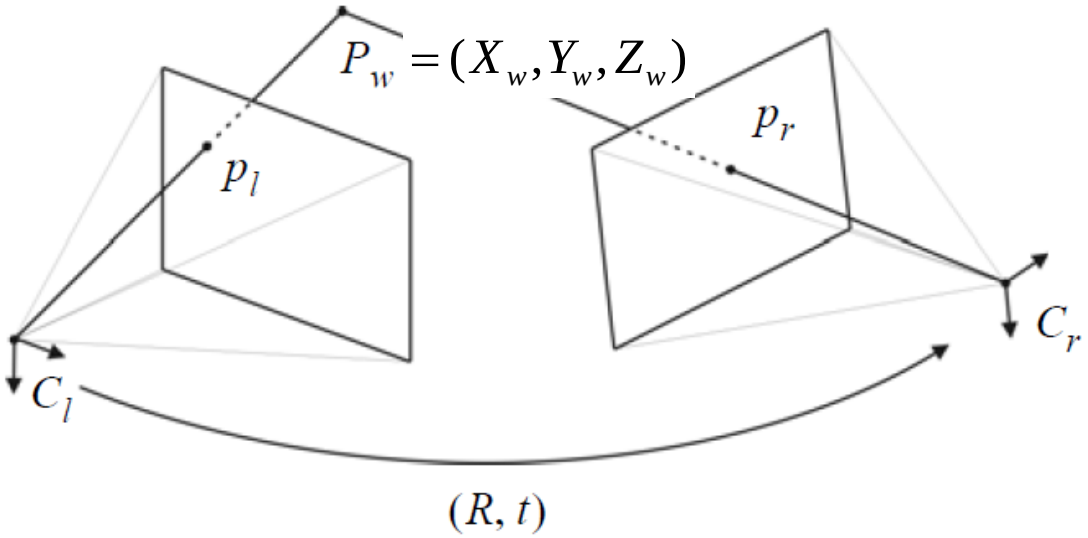
Disparity



- **Disparity** is the difference in image location of the projection of a 3D point in two image planes
- **Baseline** is the distance between the two cameras

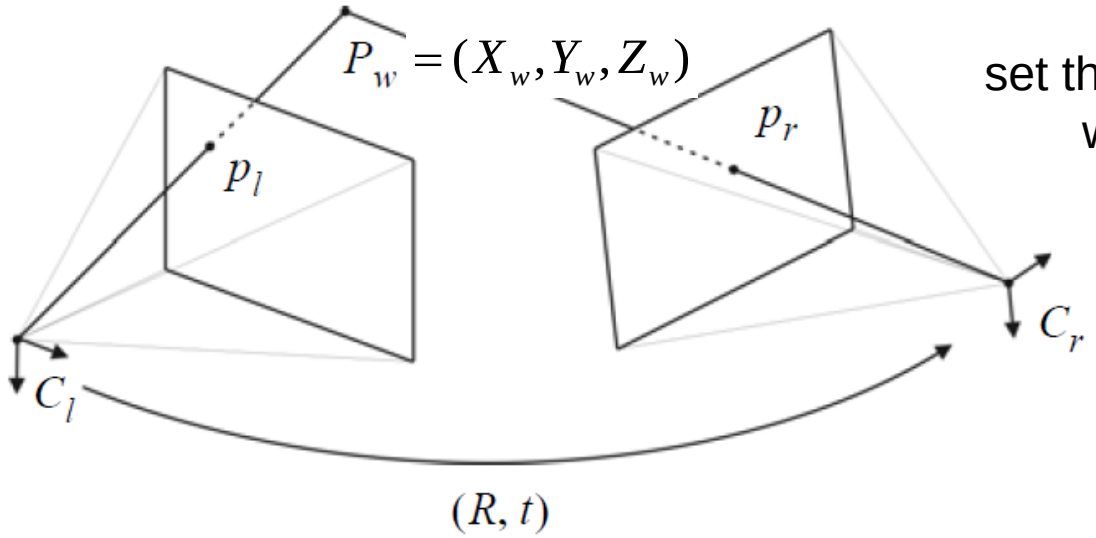
Stereo Vision | general case

- Two identical cameras do not exist in nature!
- Aligning both cameras on a horizontal axis is very difficult
- In order to use a stereo camera, we need to know the intrinsic extrinsic parameters of each camera, that is, the relative pose between the cameras (rotation, translation) ⇒ We can solve for this through camera calibration



Stereo Vision | general case

- To estimate the 3D position of P_W we can construct the system of equations of the left and right camera
- Triangulation is the problem of determining the 3D position of a point given a set of corresponding image locations and known camera poses.



Left camera:
set the world frame to coincide with the left camera frame

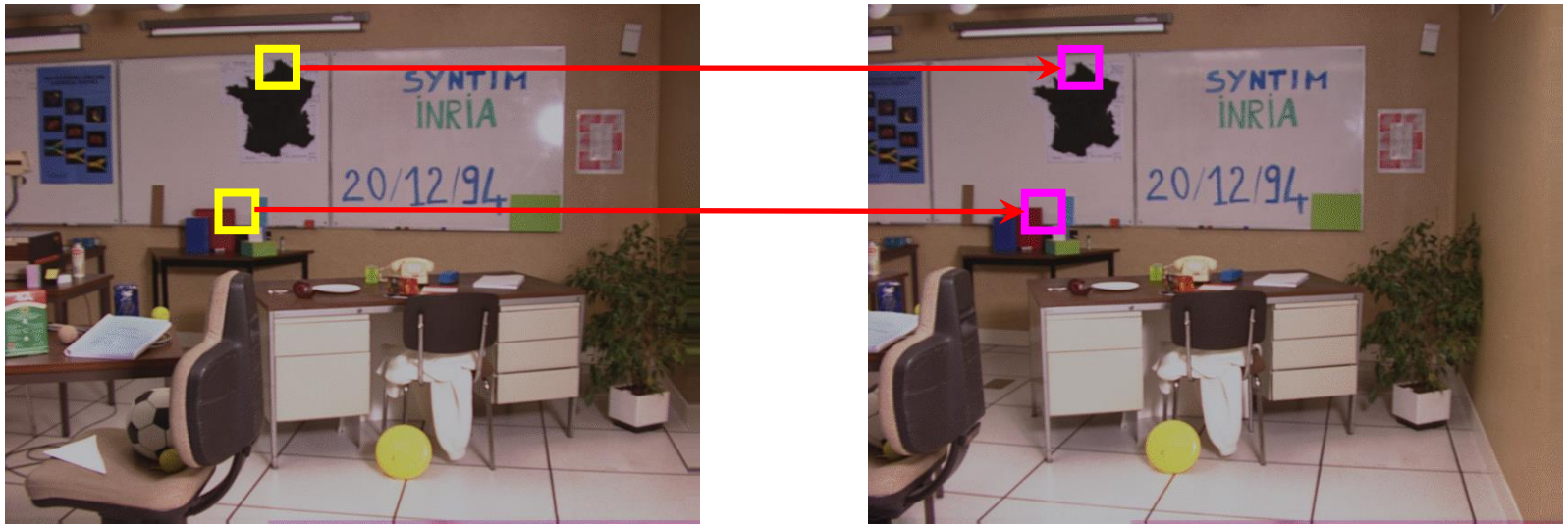
$$\tilde{p}_l = \lambda_l \begin{bmatrix} u_l \\ v_l \\ 1 \end{bmatrix} = K_l \begin{bmatrix} X_w \\ Y_w \\ Z_w \end{bmatrix}$$

Right camera:

$$\tilde{p}_r = \lambda_r \begin{bmatrix} u_r \\ v_r \\ 1 \end{bmatrix} = K_r R \begin{bmatrix} X_w \\ Y_w \\ Z_w \end{bmatrix} + T$$

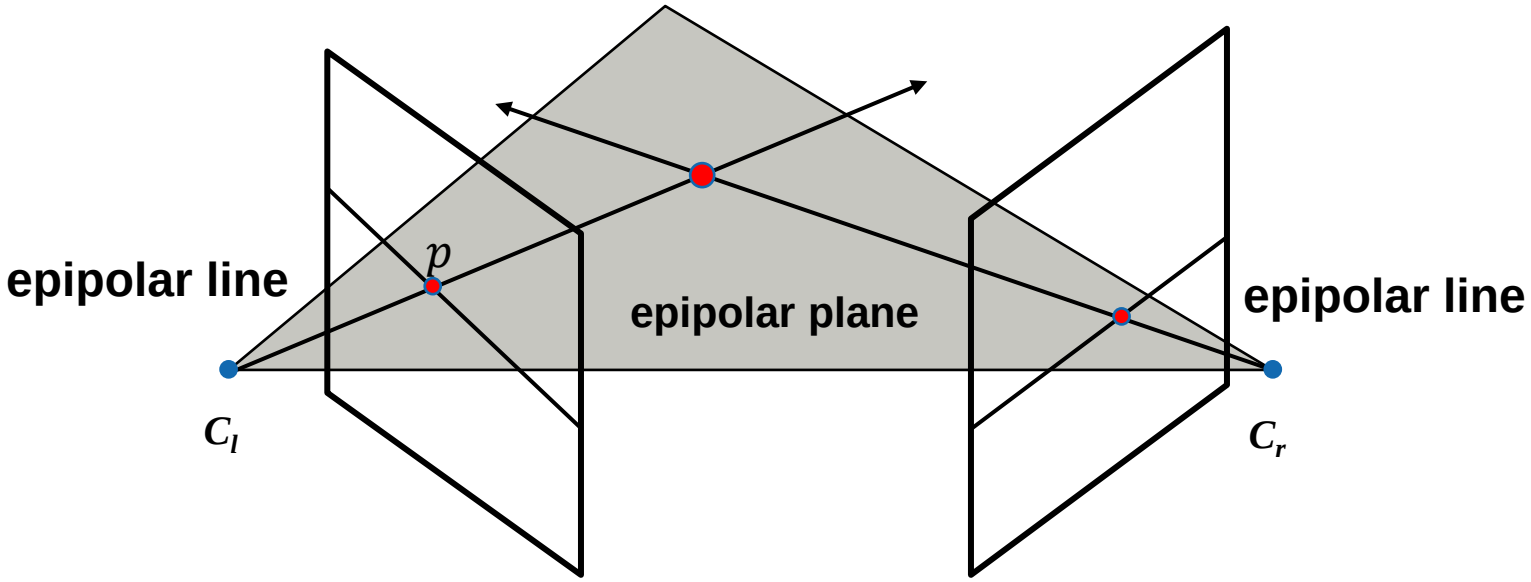
Correspondence Search | the problem

- Goal: identify corresponding points in the left and right images, which are the reprojection of the same 3D scene point
  - Typical similarity measures: Normalized Cross-Correlation (NCC) , Sum of Squared Differences (SSD), Census Transform
  - Exhaustive image search can be computationally very expensive! Can we make the correspondence search in 1D?



Correspondence Search | the epipolar constraint

- The epipolar plane is defined by the image point $\mathbf{p}$ and the optical centers
- Impose the epipolar constraint to aid matching: search for a correspondence along the epipolar line

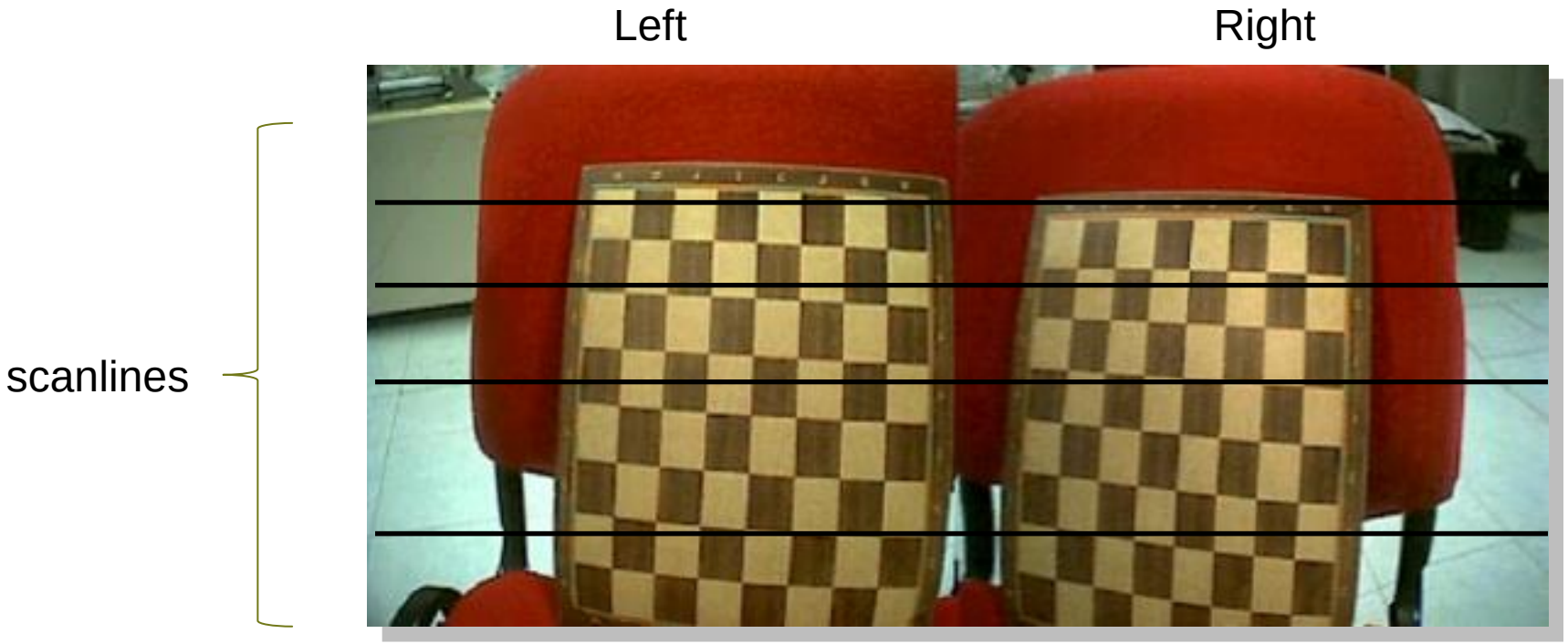


Correspondence Search | the epipolar constraint

- Thanks to the epipolar constraint, corresponding points can be searched for, along epipolar lines ⇒ computational cost reduced to 1 dimension!

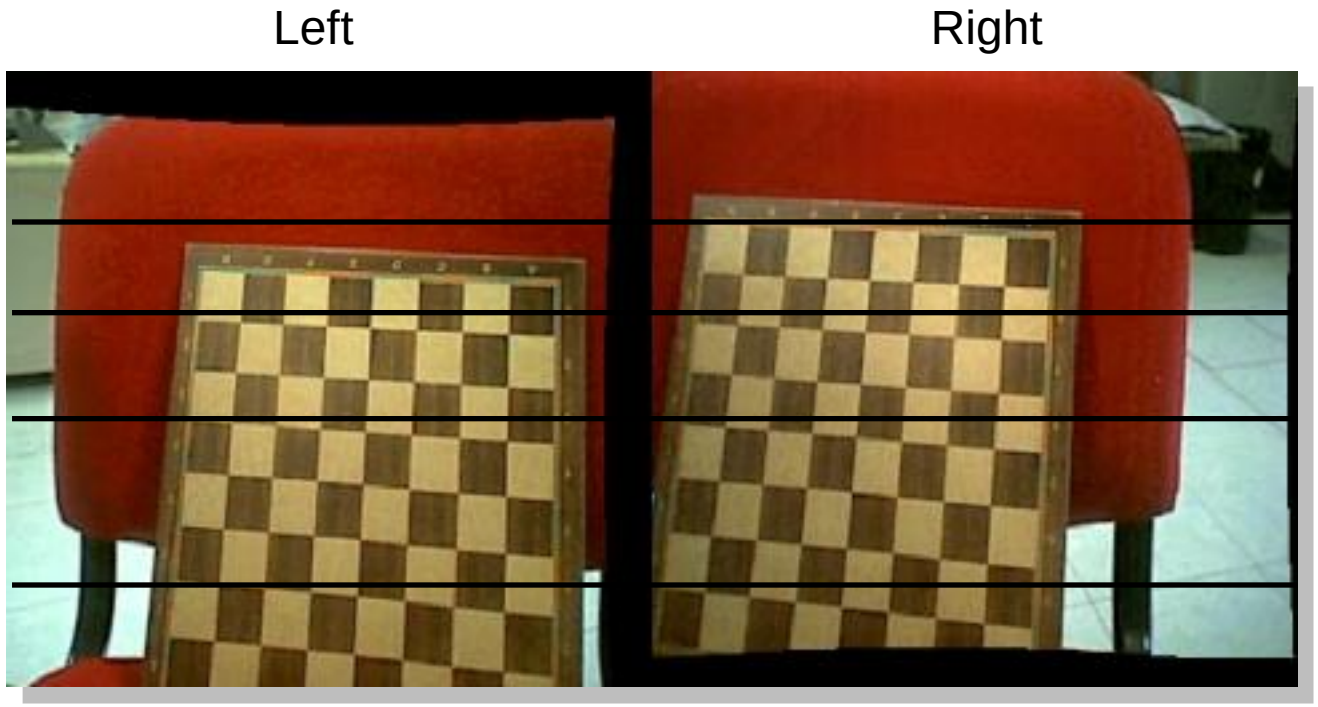
Epipolar Rectification

- Goal: transform the left and right image so that pairs of conjugate epipolar lines become collinear and parallel to one of the image axes (usually the horizontal one)



Epipolar Rectification

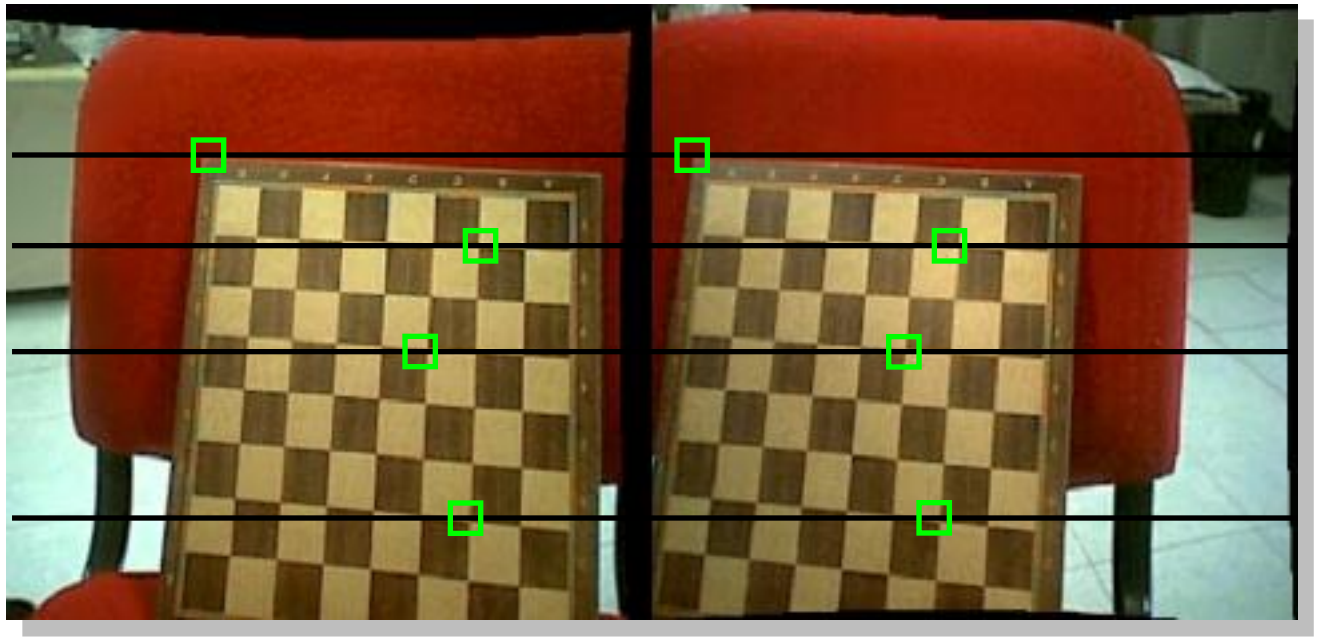
- First, remove radial distortion



Epipolar Rectification

- First, remove radial distortion
- Then, compute homographies and rectify

Left Right



Stereo Vision | disparity map

- The disparity map holds the disparity value at every pixel:
  - Identify correspondent points of all image pixels in the original images
  - Compute the disparity $(u_l - u_r)$ for each pair of correspondences
- Usually visualized in gray-scale images
- Close objects experience bigger disparity; thus, they appear brighter in disparity map

Left image

Right image

Disparity Map

Stereo Vision | disparity map

- The disparity map holds the disparity value at every pixel:
  - Identify correspondent points of all image pixels in the original images
  - Compute the disparity $(u_l - u_r)$ for each pair of correspondences
- Usually visualized in gray-scale images
- Close objects experience bigger disparity; thus, they appear brighter in disparity map
- From the disparity, we can compute the depth Z as:

$$Z = \frac{bf}{u_l - u_r}$$