Rectification

Dr. Gerhard Roth

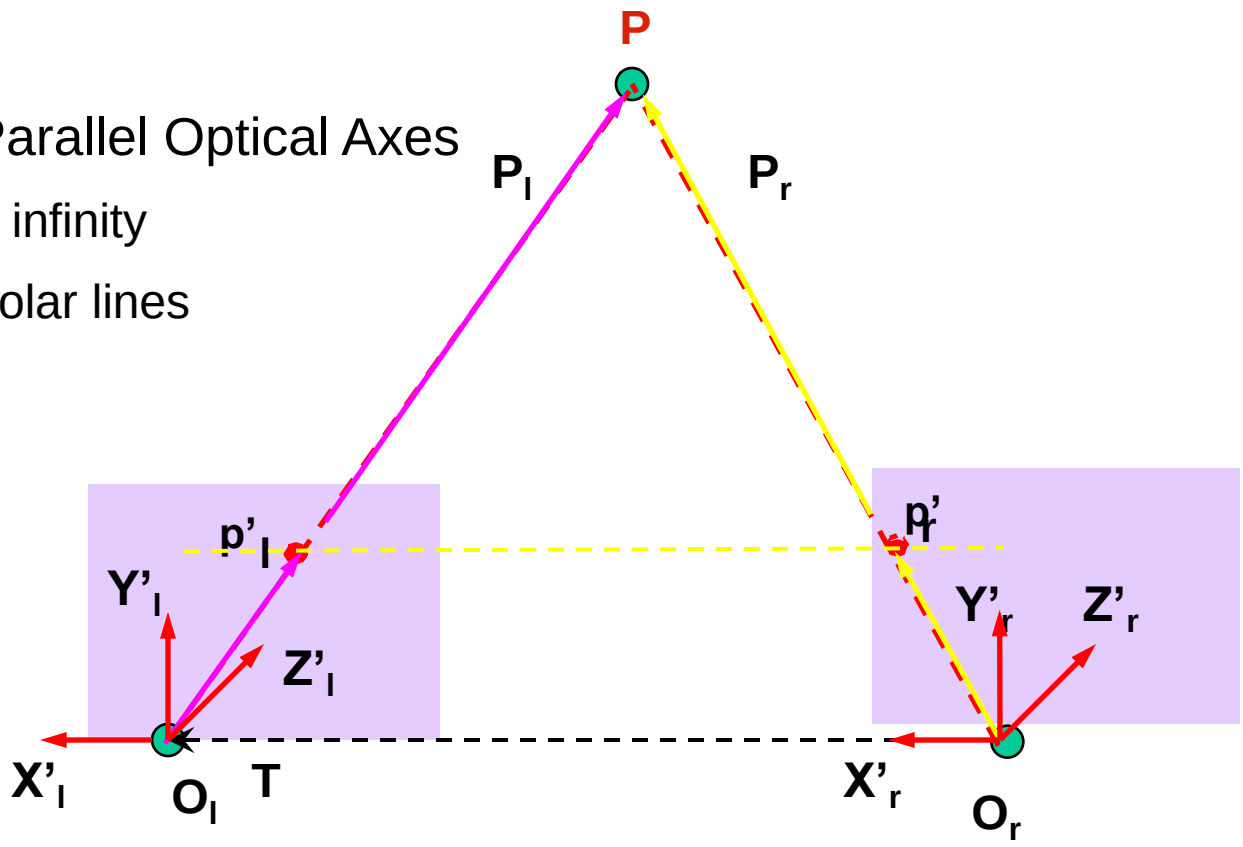
Winter 2012

Problem Definition

- Given a pair of stereo images, the intrinsic parameters of each camera, and the extrinsic parameters of the system, R, and T, compute the image transformation that makes epipolar lines collinear and parallel to horizontal axis
- Basically convert a general stereo configuration to a simple stereo configuration
  - Correlation based correspondence algorithms always assume a simple stereo configuration
  - This speeds up matching because searching horizontal epipolar lines is easier than general epipolar lines
  - So rectification is a good pre-processing step if you want to do correspondence faster using a correlation based algorithm

Stereo Rectification

- Stereo System with Parallel Optical Axes
  - Epipoles are at infinity
  - Horizontal epipolar lines



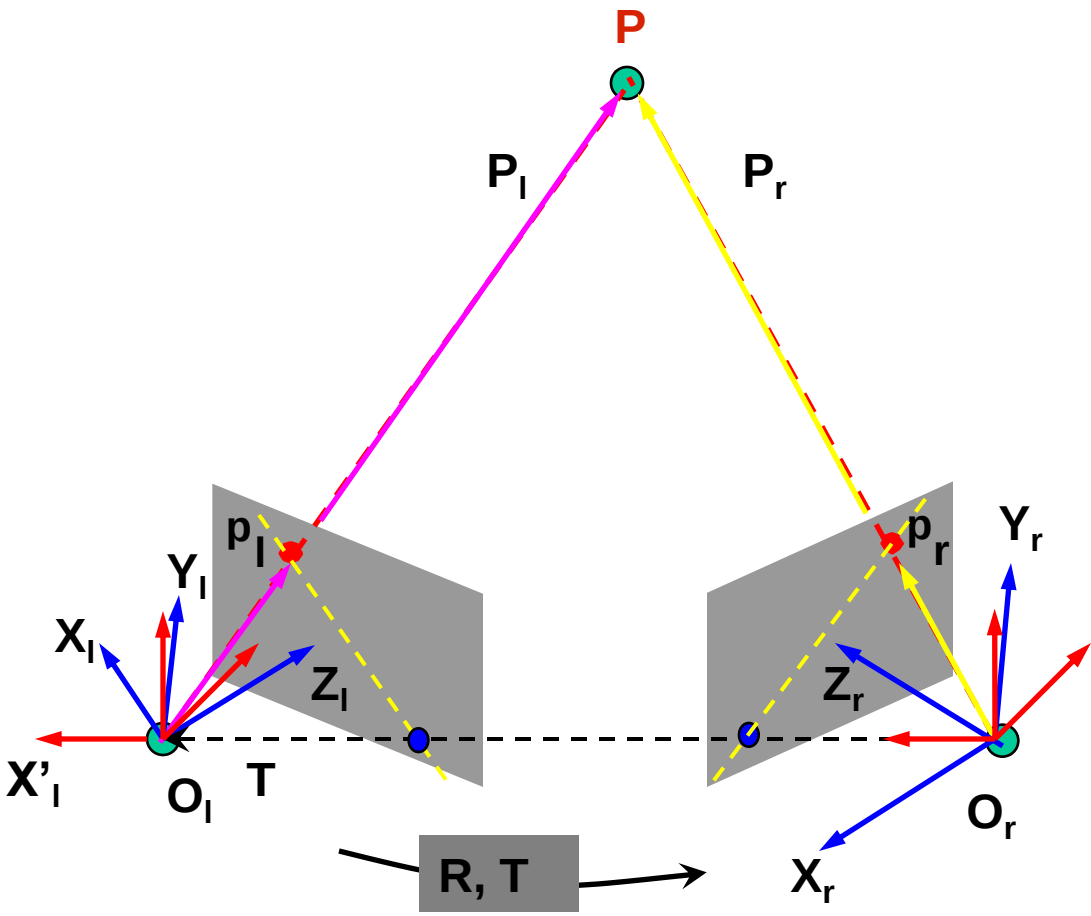
Rectification

- Given a stereo pair, the intrinsic and extrinsic parameters, find the image transformation to achieve a stereo system of horizontal epipolar lines
- A simple algorithm: Assuming calibrated stereo cameras

Stereo Rectification

Algorithm

- Rotate both left and right camera so that they share the same X axis : O_r-O_l = T
- Define a rotation matrix R_{rect} for the left camera
- Rotation Matrix for the right camera is $R_{rect}R^T$
- Rotation can be implemented by image transformation called a homography (see homography.ppt)

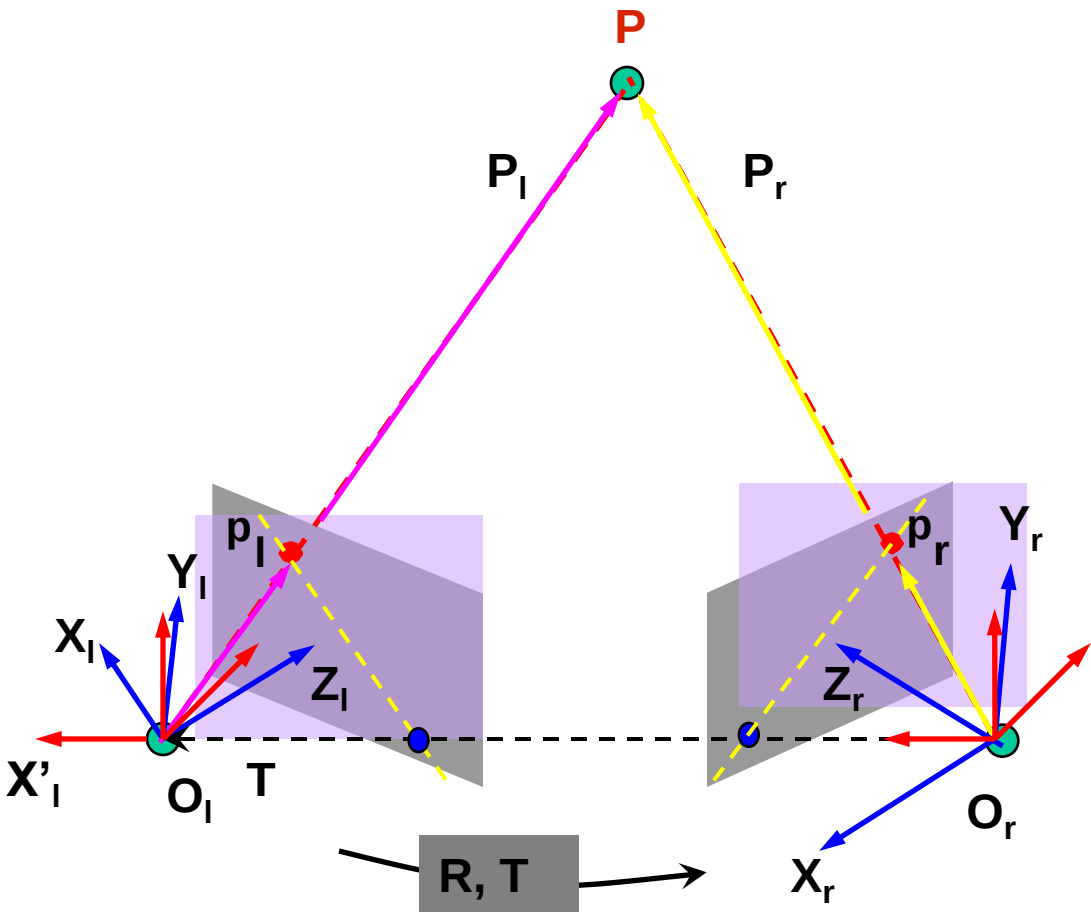


$X_l' = T, \quad Y_l' = X_l' \times Z_l, \quad Z'_l = X_l' \times Y_l'$

Stereo Rectification

Algorithm

- Rotate both left and right camera so that they share the same X axis : O_r-O_l = T
- Define a rotation matrix R_{rect} for the left camera
- Rotation Matrix for the right camera is $R_{rect}R^T$
- Rotation can be implemented by image transformation called a homography (see homography.ppt)

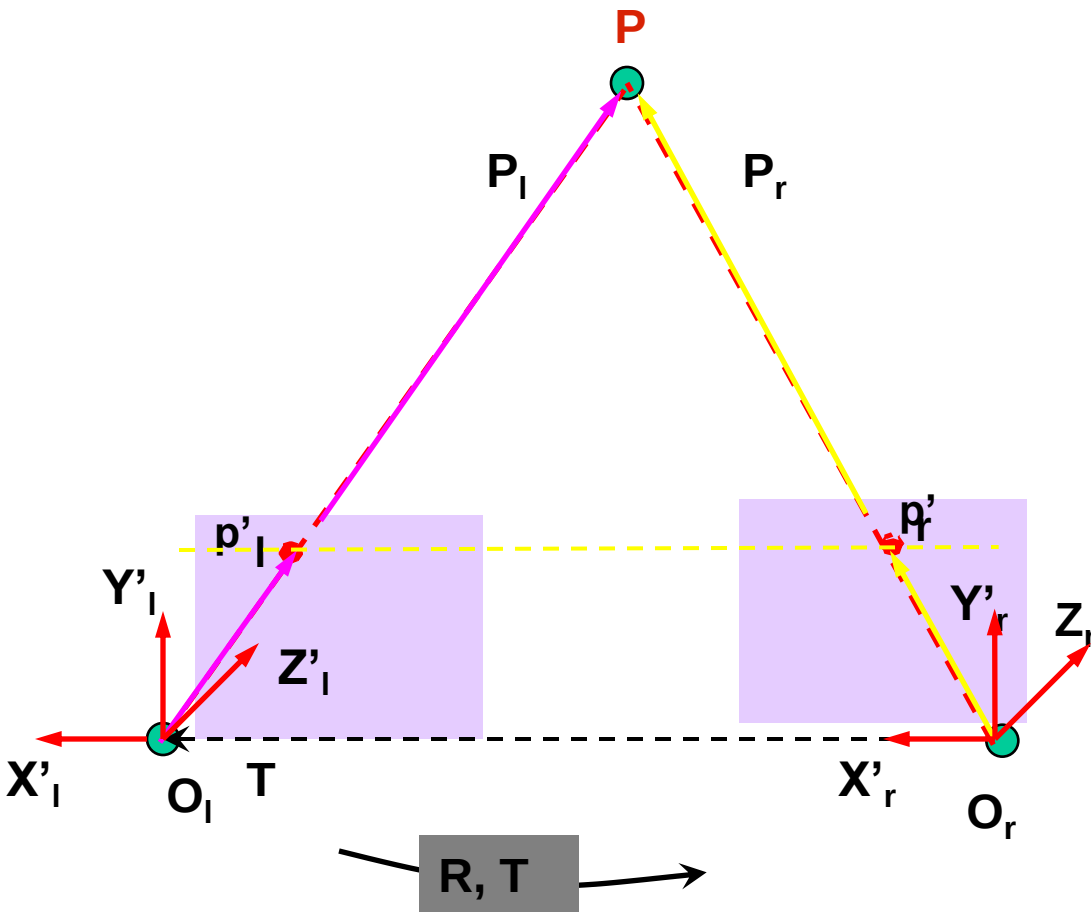


$X_l' = T, \quad Y_l' = X_l' \times Z_l, \quad Z'_l = X_l' \times Y_l'$

Stereo Rectification

Algorithm

- Rotate both left and right camera so that they share the same X axis : O_r-O_l = T
- Define a rotation matrix R_{rect} for the left camera
- Rotation Matrix for the right camera is $R_{rect}R^T$
- Rotation can be implemented by image transformation (see homography.ppt)



$T' = (B, 0, 0)$, $P'_r = P'_l - T'$

Algorithm Rectification

1. Build the matrix R_{rect}
2. Set $R_l = R_{rect}$ and $R_r = R\ R_{rect}$
3. For each left-camera point $p_l = [x,y,f]^T$ compute R_l $p_l = [x',y',z']$ and the corresponding rectified point as $p_l' = f/z'[x',y',z']$ (these are in camera co-ords)
4. Repeat the previous step for the right camera using R_r and p_r

- Rotate the point in camera co-ordinates and and then reproject to create new camera co-ordinates
- In practice, steps 3 and 4 require back projection
- This is easily done with a homography which is computed from the known rotation and calibration

Building matrix R_{rect}

- Make the new x axis along the direction of the baseline (the b vector)
  - This is vector e1
- Make new y axis orthogonal to the new x and the old z which is along the old optical axis (cross product)
  - This is vector e2
- Make new z axis orthogonal to the baseline and the new y axis (cross product)
  - This is vector e3
- Rotation matrix is now complete

$$R_{rect} = \begin{pmatrix} e_1^T \\ e_2^T \\ e_3^T \end{pmatrix}$$

- Rotates left camera so that epipolar lines are parallel

Homography for Rotation

- If camera is rotated (but not translated)
- Home position

$$x = K[I \mid 0]\begin{bmatrix} X \\ 1 \end{bmatrix} = KX$$

- Rotation by a matrix R

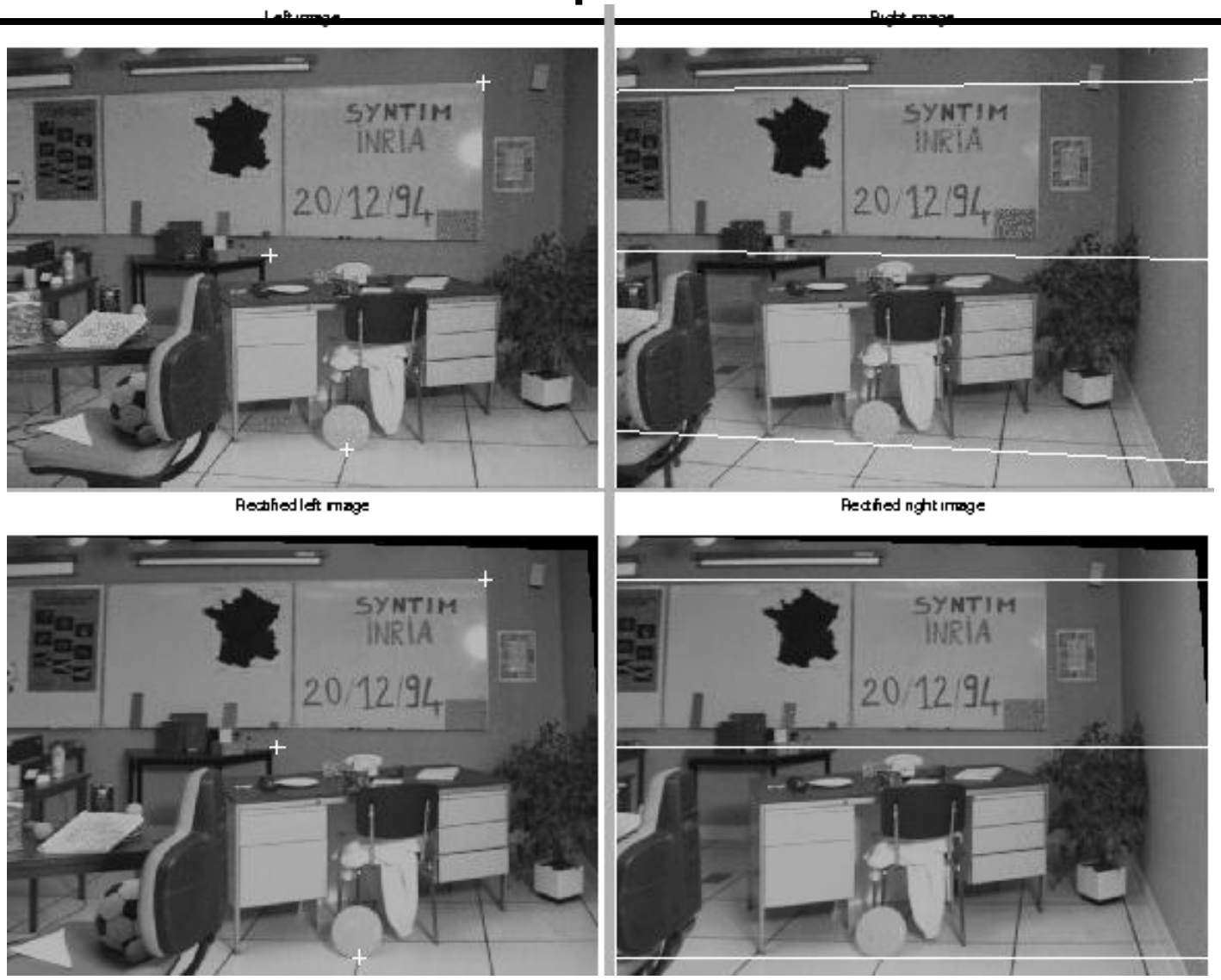
$$x' = K[R \mid 0]\begin{bmatrix} X \\ 1 \end{bmatrix} = KRX$$

- So $x' = KRK^{-1}x$
- Where KRK^{-1} is a 3by3 matrix called a homography

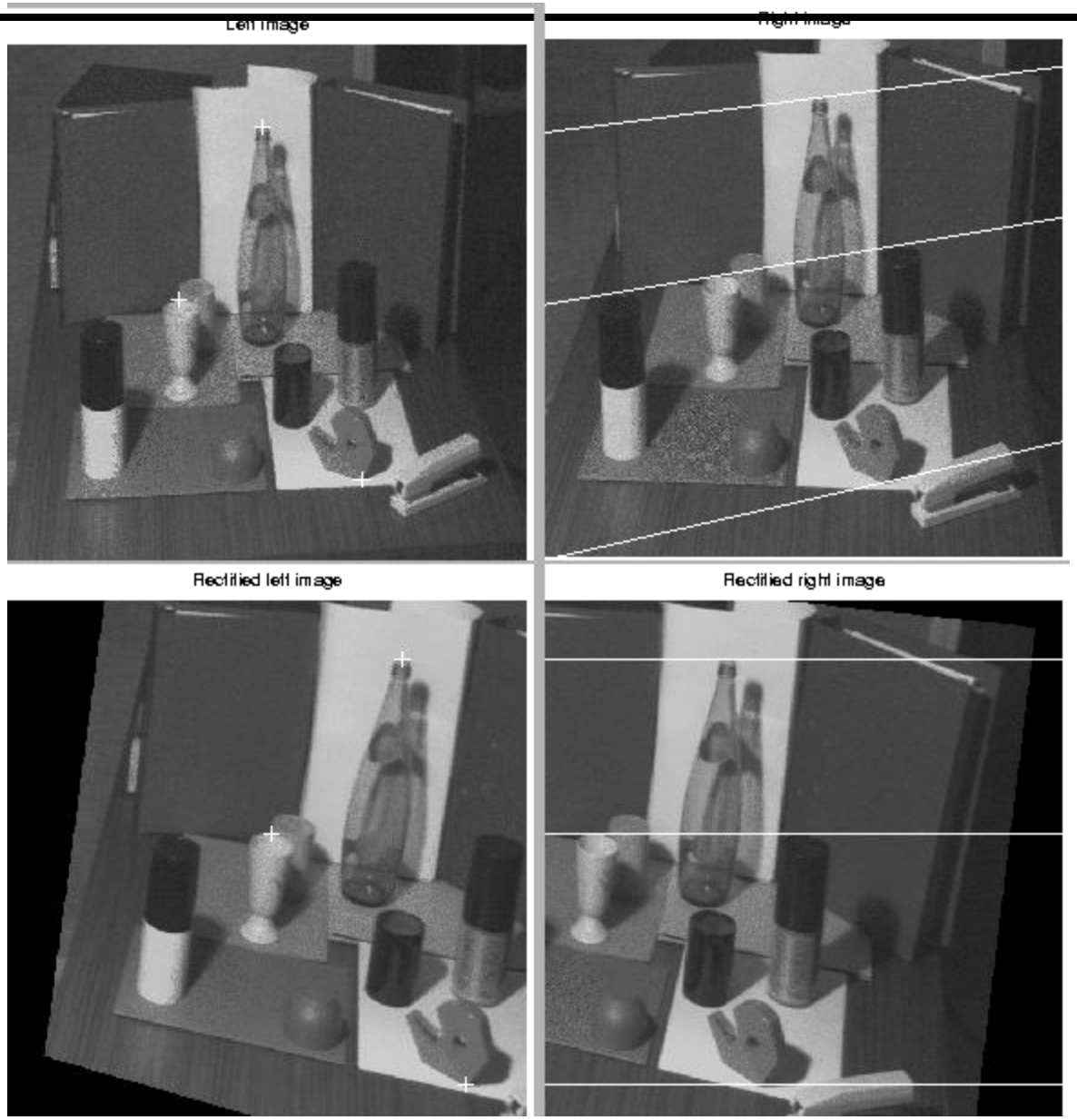
Homography for Rotation

- We know that $\mathbf{x}' = \mathbf{K}\mathbf{R}\mathbf{K}^{-1}\mathbf{x}$
- This means that for any pixel in the new image where x' = [u',v',1] we can compute the pixel x = [u,v,1] in th old image that is mapped to x under this rotation
  - Go through every pixel x' in the new image and put in its place the image at location x in the old image
- This make a new image that is a rotated version of the original old image
- A simple way to apply the rotation to these images
- Note that $x' = M x$ and $x = M^{-1} x'$ where M is the homography that does the rotation
  - So given M (which we can compute) from K and R we can make a new image which looks like a rotated version of the old

Rectification Example



Rectification Example



Rectification Example

Rectification

- Used to transform a general stereo system into a simple stereo system

1. Do the rectification to get two new images
2. Do correspondence in the new simple stereo images
3. Take the 3D points that are found and convert them back into the old image frames (invert rotations)

The result is a set of 3D points, but they are computed more quickly because we do correspondence in simple stereo.